HUMAN RESOURCES SYSTEM
FOR SMALL AND MEDIUM
SIZED ENTERPRISES
(SME-HR)

MUHAMMAD TAUFIQ BIN MOHD JASLAN

Bachelor of Computer Science
(Software Engineering) with Honours

UNIVERSITI MALAYSIA PAHANG

UNIVERSITI MALAYSIA PAHANG

DECLARATION OF THESIS AND COPYRIGHT

Author's Full Name : MUHAMMAD TAUFIQ BIN MOHD JASLAN

Date of Birth

Title : SME-HR SYSTEM

Academic Session : SEMESTER 1 2022/2023

I declare that this thesis is classified as:

☐	CONFIDENTIAL	(Contains confidential information under the Official Secret Act 1997)*
☐	RESTRICTED	(Contains restricted information as specified by the organization where research was done)*
☑	OPEN ACCESS	I agree that my thesis to be published as online open access (Full Text)

I acknowledge that Universiti Malaysia Pahang reserves the following rights:

1. The Thesis is the Property of Universiti Malaysia Pahang
2. The Library of Universiti Malaysia Pahang has the right to make copies of the thesis for the purpose of research only.
3. The Library has the right to make copies of the thesis for academic exchange.

Certified by:

(Student's Signature)

New IC/Passport Number
Date: 2 July 2023

(Supervisor's Signature)

Azlina binti Zainuddin
Name of Supervisor
Date: 28 July 2023

NOTE : * If the thesis is CONFIDENTIAL or RESTRICTED, please attach a thesis declaration letter.

SUPERVISOR'S DECLARATION

I/~~We*~~ hereby declare that I/~~We*~~ have checked this thesis/~~project*~~ and in my/~~our*~~ opinion, this thesis/~~project*~~ is adequate in terms of scope and quality for the award of the degree of *~~Doctor of Philosophy/ Master of Engineering/ Master of Science in~~ Bachelor of Computer Science in Software Engineering.

(Supervisor's Signature)

Full Name : AZLINA BINTI ZAINUDDIN
LECTURER
FACULTY OF COMPUTING
COLLEGE OF COMPUTING & APPLIED SCIENCES
UNIVERSITI MALAYSIA PAHANG
26600 PEKAN, PAHANG DARUL MAKMUR
TEL : 09-424 4638 FAX : 09-424 4666

Position :

Date : 28 July 2023

-

(Co-supervisor's Signature)

Full Name :

Position :

Date :

Universiti
Malaysia
PAHANG
Engineering • Technology • Creativity

STUDENT’S DECLARATION

I hereby declare that the work in this thesis is based on my original work except for quotations and citations which have been duly acknowledged. I also declare that it has not been previously or concurrently submitted for any other degree at Universiti Malaysia Pahang or any other institutions.

____

(Student’s Signature)

Full Name : MUHAMMAD TAUFIQ BIN MOHD JASLAN

ID Number : CB20166

Date : 2 JULY 2023

HUMAN RESOURCES SYSTEM
FOR SMALL AND MEDIUM
SIZED ENTERPRISES
(SME-HR)

MUHAMMAD TAUFIQ BIN MOHD JASLAN

Thesis submitted in fulfillment of the requirements
for the award of the degree of
Bachelor of Computer Science (Software Engineering) with Honours

Faculty of Computing

UNIVERSITI MALAYSIA PAHANG

JULY 2023

ACKNOWLEDGEMENTS

First and foremost, I want to thank Allah SWT, the Almighty, for blessing my planned project work and allowing it to be successfully completed.

My supervisor, Ms Azlina Binti Zainuddin, has my deepest gratitude for her insightful and helpful feedback at every stage of my work. Throughout my Final Year Project (FYP), she consistently offered me encouragement and practical guidance. Her steadfast cooperation and perseverance were essential to complete this project.

In addition, I would like to thank Dr Danakorn Nincarean A/L Eh Phon, the coordinator of the Final Year Project (FYP), who provided the information for FYP students to do their Final Year Project (FYP) and the direction for the thesis.

Lastly, I want to express my appreciation to all of my friends and family members who have helped me in any manner to complete my thesis, whether directly or indirectly.

ABSTRAK

Pada era globalisasi kini, kebanyak syarikat di Malaysia mempunyai jabatan sumber manusia dan menggunakan pelbagai kaedah untuk menguruskan tugas mereka. Satu pendekatan yang popular ialah penggunaan sistem Pengurusan Sumber Manusia, yang dapat membantu organisasi untuk memudahkan proses kerja. Sistem-sistem ini boleh didapati di dalam talian, tetapi bayaran akan dikenakan ke atas mengikut pakej-pakej yang telah disediakan. Walaupun kebanyakan syarikat sudah menggunakan sistem-sistem ini, tetapi terdapat juga sesetengah syarikat, terutamanya Perusahaan Kecil dan Sederhana (PKS), masih belum menggunakannya. PKS adalah perniagaan dengan jualan kurang daripada RM20 juta atau kurang daripada 75 pekerja sepenuh masa, termasuk sektor pembuatan, pertanian, pembinaan, perlombongan & kuari, dan perkhidmatan. Selain itu, masih terdapat juga beberapa syarikat PKS yang menggunakan kaedah tradisional, seperti dokumen kertas, untuk menyimpan semua maklumat syarikat, termasuk butiran pekerja, surat cuti sakit, dan resit rujukan bagi tuntutan. Kaedah tradisional ini mempunyai beberapa kelemahan, seperti kemungkinan kehilangan dokumen dan kesukaran mencari rekod tertentu. Ini boleh menyebabkan masalah yang serius bagi pihak PKS jika mereka terus menggunakan kaedah ini. Terdapat beberapa faktor mengapa syarikat PKS ini masih tidak menggunakan sistem Pengurusan Sumber Manusia untuk menguruskan kerja mereka, dan salah satunya adalah harga sistem tersebut. Bagi mengatasi masalah ini, satu sistem yang dicadangkan, iaitu Sistem Sumber Manusia untuk Perusahaan Kecil dan Sederhana (SME-HR), akan menjadi penyelesaian yang sempurna bagi syarikat PKS untuk menyelesaikan masalah mereka. Tujuan system ini dibina ialah untuk memudahkan jabatan sumber manusia di syarikat PKS untuk menguruskan tugas-tugas mereka supaya lebih sistematik dan teratur. Sistem Sumber Manusia ini akan memberi tumpuan khusus kepada PKS dan menyediakan faedah seperti membantu mereka menyusun semua dokumen dan menguruskan kerja mereka dengan cekap. Bagi membantu untuk melancarkan projek ini, satu metodologi telah dipilih iaitu “Rapid Application Development” atau lebih dikenali sebagai metodologi RAD. Oleh yang demikian, ia dapat membantu syarikat-syarikat PKS di luar sana terutamanya rakan usaha sama projek ini iaitu Anh Exora Enterprises untuk memudahkan kerja mereka.

ABSTRACT

Many companies in Malaysia have their own human resources department and use various methods to manage their work. One popular method is third-party Human Resources Management systems, which can help companies organize and ease their work. These systems can be found online, but they require a monthly payment. Even though most companies already use these systems, there are several companies, particularly Small and Medium-Sized Enterprises (SMEs), still do not. SMEs are businesses with a sales turnover of less than RM20 million or less than 75 full-time workers, including manufacturing, agriculture, construction, mining & quarrying, and services. There are a few SMEs that still use traditional, manual methods such as paper documents to save all the company's information, including employee details, medical leave letters, and reference receipts for claims. Traditional methods have several disadvantages, such as the potential loss of documents and difficulty searching for specific records. This can cause severe problems for SMEs if they continue to use these methods. There are several reasons why the SMEs still do not use a third-party system to manage their work, one of the reasons is the system's pricing. To overcome this problem, a proposed system, the Human Resources System for Small and Medium-Sized Enterprises (SME-HR), will be the perfect solution for SME companies to solve their problems. The purpose of this SME-HR system is to help the SME's company to help them to ease their Human Resources workload. Besides that, to ensure this project will be successfully developed, Rapid Application Development or also known as RAD has been chosen. Due to that, it can help the SME company especially Anh Exora Enterprises which is the collaborator of this project to ease their Human Resources workload.

TABLE OF CONTENT

DECLARATION

TITLE PAGE

ACKNOWLEDGEMENTS **ii**

ABSTRAK **iii**

ABSTRACT **iv**

TABLE OF CONTENT **v**

LIST OF TABLES **ix**

LIST OF FIGURES **x**

LIST OF SYMBOLS **xiv**

LIST OF ABBREVIATIONS **xv**

CHAPTER 1 INTRODUCTION **1**

1.1 Introduction 1

1.2 Problem Statement 2

1.3 Objective 3

1.4 Scope 4

1.5 Thesis Organization 4

CHAPTER 2 LITERATURE REVIEW **6**

2.1 Introduction 6

2.2 Existing Systems 6

2.2.1 altHR 6

2.2.2 Ignite Human Resources Management System 9

2.2.3 AuroraHRM 13

2.3 Analysis Comparison of Existing System 15

2.3.1 Analysis of comparison of the existing system with the proposed project 15

2.3.2 Advantages and Disadvantages 18

2.3.3 Relevance of comparison with project title 19

2.4 Survey Data Analysis 19

2.5 Summary 21

CHAPTER 3 METHODOLOGY 22

3.1 Introduction 22

3.2 Project Management Framework/Methodology 22

3.2.1 Requirements Planning 23

3.2.2 User Design 24

3.2.3 Construction 24

3.2.4 Cutover 24

3.3 Project Requirement 25

3.3.1 Functional Requirements 25

3.3.2 Non-Functional Requirements 27

3.3.3 Constraints and Limitations 27

3.4 Proposed Design 28

3.4.1 Context Diagram 28

3.4.2 Use Case Diagram 30

3.4.3 Use Case Description 31

3.4.4 Activity Diagram 62

3.5 Data Design 70

3.5.1 Entity Relationship Diagram 71

3.5.2 Database Dictionary 72

3.6 Proof of Initial Concept 79

3.6.1 Login 80

3.6.2 Manage Employee 81

3.6.3 Manage Leave 85

3.6.4 Manage Claim 90

3.6.5 Manage Payroll 94

3.6.6 Manage EA Form 98

3.7 Testing Plan 103

3.8 Potential Use of Proposed Solution 105

3.9 Gantt Chart 106

CHAPTER 4 IMPLEMENTATION, RESULT AND DISCUSSION 108

4.1 Introduction 108

4.2 Implementation Process 108

4.2.1 Development of Environment 108

4.2.2 Development of Employee Module 110

4.2.3 Development of Payroll Module 115

4.2.4 Development of Claim Module 120

4.2.5 Development of Leave Module 123

4.2.6 Development of EA Form Module 127

4.3 User Acceptance Test 132

CHAPTER 5 CONCLUSION 135

5.1 Introduction 135

5.2 Objective Revisited 135

5.3 Research Constraint 136

5.3.1 Limited Time 136

5.3.2 Development Constraints 137

5.4 Future Work 138

REFERENCES 139

APPENDIX A USER ACCEPTANCE TEST RESPONSES 141

APPENDIX B SOFTWARE REQUIREMENT SPECIFICATION 154

APPENDIX C SOFTWARE DESIGN DOCUMENT 155

LIST OF TABLES

Table 1: Comparison of Three Existing System 15

Table 2: Table of Advantages and Disadvantages 18

Table 3: Non-Functional Requirement 27

Table 4: Login Use Case Description 32

Table 5: Manage Employee Use Case Diagram 34

Table 6: View Employee Use Case Description 37

Table 7: Manage Profile Use Case Description 39

Table 8: Manage Payroll Use Case Description 40

Table 9: View Payroll Use Case Diagram 43

Table 10: Manage Leave Use Case Description 45

Table 11: Manage Claim Use Case Description 51

Table 12: Manage EA Form Use Case Description 57

Table 13: View EA Form Use Case Description 60

Table 14: Staff Data Dictionary 72

Table 15: User Types Data Dictionary 73

Table 16: Positions Data Dictionary 73

Table 17: Salaries Data Dictionary 73

Table 18: Salary Types Data Dictionary 74

Table 19: EA Form Data Dictionary 75

Table 20: Claims Data Dictionary 75

Table 21: Claim Types Data Dictionary 76

Table 22: Leaves Data Dictionary 77

Table 23: Leave Reports Data Dictionary 77

Table 24: Leave Entitlements Data Dictionary 78

Table 25: Leave Types Data Dictionary 79

Table 26: User Acceptance Test (UAT) 103

LIST OF FIGURES

Figure 1: altHR Logo 6
Figure 2: altHR System Mainpage 7
Figure 3: altHR People Module Interface 8
Figure 4: altHR Payroll Module Interface 8
Figure 5: altHR Pring Plan 9
Figure 6: Ignite HRMS Logo 9
Figure 7: Ignite HRMS Interface 10
Figure 8: Ignite HRMS Payroll Module 11
Figure 9: Ignite HRMS Personnel Module 11
Figure 10: Ignite HRMS Leave Module 12
Figure 11: Ignite HRMS Attendance Module 12
Figure 12: Ignite HRMS Claims Module 12
Figure 13: Ignite HRMS Packages 13
Figure 14: AuroraHRM Logo 13
Figure 15: AuroraHRM Interface 14
Figure 16: AuroraHRM Plan 15
Figure 17: Data for Number of Employee for Each Responses 19
Figure 18: Data for the Current Problems That the Company Faces 20
Figure 19: Data for the Important of Human Resources System to SME's Company 21
Figure 20: Rapid Application Development (RAD) Methodology 23
Figure 21: Context Diagram of SME-HR System 29
Figure 22: SME-HR Use Case Diagram 30
Figure 23: Login Use Case Diagram 31
Figure 24: Manage Employee Use Case Diagram 34
Figure 25: View Employee Use Case Diagram 37
Figure 26: Manage Profile Use Case Diagram 39
Figure 27: Manage Payroll Use Case Diagram 40
Figure 28: View Payroll Use Case Diagram 43
Figure 29: Manage Leave Use Case Diagram 45
Figure 30: Manage Claim Use Case Diagram 51
Figure 31: Manage EA Form Use Case Diagram 57
Figure 32: View EA Form Use Case Diagram 60
Figure 33: Login Activity Diagram 63

Figure 34: Manage Employee Activity Diagram 64
Figure 35: Manage Profile Activity Diagram 65
Figure 36: Manage Payroll Activity Diagram 66
Figure 37: Manage Leave Activity Diagram 67
Figure 38: Manage Claim Activity Diagram 68
Figure 39: Manage EA Form Activity Diagram 69
Figure 40: SME-HR System ERD 71
Figure 41: User Login Interface 80
Figure 42: Employee List Page Interface 81
Figure 43: Add Employee Page 81
Figure 44: Add Employee Confirmation Page Interface 82
Figure 45: Add Employee Successful Page Interface 82
Figure 46: Update Employee Page Interface 83
Figure 47: Update Employee Confirmation Page Interface 83
Figure 48: Update Employee Successful Page Interface 84
Figure 49: Delete Employee Confirmation Page Interface 84
Figure 50: Delete Employee Successful Page Interface 85
Figure 51: Leave List Page Interface 85
Figure 52: Add Leave Page Interface 86
Figure 53: Add Leave Confirmation Page Interface 86
Figure 54: Add Leave Successful Page Interface 87
Figure 55: Update Leave Page Interface 87
Figure 56: Update Leave Confirmation Page Interface 88
Figure 57: Update Leave Successful Page Interface 88
Figure 58: Delete Leave Confirmation Page Interface 89
Figure 59: Delete Leave Successful Page Interface 89
Figure 60: Claim List Page Interface 90
Figure 61: Add Claim Page Interface 90
Figure 62: Add Claim Confirmation Page Interface 91
Figure 63: Add Claim Successful Page Interface 91
Figure 64: Update Claim Page Interface 92
Figure 65: Update Claim Confirmation Page Interface 92
Figure 66: Update Claim Successful Page Interface 93
Figure 67: Delete Claim Confirmation Page Interface 93
Figure 68: Delete Claim Successful Page Interface 94

Figure 69: Payroll List Page Interface 94
Figure 70: Add Payroll Page Interface 95
Figure 71: Add Payroll Confirmation Page Interface 95
Figure 72: Add Payroll Successful Page Interface 96
Figure 73: Update Payroll Page Interface 96
Figure 74: Update Payroll Confirmation Page Interface 97
Figure 75: Update Payroll Successful Page Interface 97
Figure 76: EA Form List Page Interface 98
Figure 77: Add EA Form Page Interface 98
Figure 78: Add EA Form Confirmation Page Interface 99
Figure 79: Add EA Form Successful Page Interface 99
Figure 80: Update EA Form Page Interface 100
Figure 81: Update EA Form Confirmation Page Interface 100
Figure 82: Update EA Form Successful Page Interface 101
Figure 83: Delete EA Form Confirmation Page Interface 101
Figure 84: Delete EA Form Successful Page Interface 102
Figure 85: Laravel - PHP Framework 108
Figure 86: Project Structure Environment in Visual Studio Code 109
Figure 87: Add Employee Page Interface 110
Figure 88: Employee View (Add Employee) Snippet Code 111
Figure 89: Employee Model Snippet Code 112
Figure 90: Employee Controller (Register Function) Snippet Code 112
Figure 91: List of Employee Page Interface 113
Figure 92: Employee View (List of Employee) Snippet Code 114
Figure 93: List of Payroll Page Interface 115
Figure 94: Generate Payroll Page Interface 115
Figure 95: View Payroll Page Interface 116
Figure 96: Payroll Model Snippet Code 117
Figure 97: Payroll Controller Snippet Code 117
Figure 98: Payroll Calculation Controller Snippet Code 118
Figure 99: Rate of Contribution for Employees' Social Security Act 1969 (Act 4) 119
Figure 100: Apply Claim Page Interface 120
Figure 101: List of Claim Page Interface 120
Figure 102: Claim View (Apply Claim) Snippet Code 121
Figure 103: Claim Model Snippet Code 122

Figure 104: Claim Controller Snippet Code 122
Figure 105: Apply Leave Page Interface 123
Figure 106: List of Leave Page Interface 123
Figure 107: Leave View (Apply Leave) Snippet Code 125
Figure 108: Leave Model Snippet Code 126
Figure 109: Leave Controller Snippet Code 126
Figure 110: List of EA Form Page Interface 127
Figure 111: Generate EA Form Page Interface 128
Figure 112: EA Form View (Generate EA Form) Snippet Code 129
Figure 113: EA Form Model Snippet Code 130
Figure 114: EA Form Controller Snippet Code 131
Figure 115: User Acceptance Test Form Introduction 132
Figure 116: Respondent Personal Details 133
Figure 117: Declaration Agreement Response 134

LIST OF SYMBOLS

HR	Human Resources
SME	Small and Medium-Sized Enterprises
RAD	Rapid Application Development
UAT	User Acceptance Test
SME-HR	Human Resources System for Small and Medium-Sized Enterprises

LIST OF ABBREVIATIONS

HR	Human Resources
SME	Small and Medium-Sized Enterprises
RAD	Rapid Application Development
UAT	User Acceptance Test
SME-HR	Human Resources System for Small and Medium-Sized Enterprises

CHAPTER 1

INTRODUCTION

1.1 Introduction

In Malaysia, every company can be categorized according to business size. Therefore, each employer must determine their company's business size. There are four commonly classified business size categories, for example, micro-sized business, small-sized business, medium-sized business, and large-sized business. A few indicators need to be followed before determining some organization size. Examples of the indicator is the total number of employees, the company revenue, production, the amount of invested capital and the market capitalization.

In addition, even though every company in the world has a different type of business size, they will have their own organizational structure that will organize specific actions and activities to accomplish their company's goals. These actions and activities are the employee's position, responsibilities, and rules. Every company will have their department, and each department will be separate according to the employees' expertise. Due to that, each employee can work on their knowledge and quickly fulfil and accomplish their task.

In an effort to accomplish their company's goals, each company will have an important role that will handle the organization, which is the human resources position. Human resources, also known as HR, is a significant figure for each company because this HR position deals with various tasks related to business operations. In addition, human resources also play an essential role in creating a healthy business environment and improving employee productivity and engagement.

All of the documents regarding the information of the employee and the company will be handled by human resources. In this era of technology, there are several systems that can help human resources to manage their work efficiently. Examples of the system is attendance software, payroll software, leave software, and claims software. This software can help the human resources department do its work. Therefore, many companies already use this system to organize their work.

Therefore, this project will propose a Human Resources System for Small and Medium-Sized Enterprise (also known as SME-HR) to help SMEs overcome their HR issues.

1.2 Problem Statement

Since all the company in Malaysia have their own human resources department, therefore there will be a few methods that will be used to manage their work. Every company uses a third-party system, such as Human Resources Management system, to organize and ease their work (Shamita, 2022). Several types of Human Resources systems can be found on the internet. For those companies who wants to use this system, they need to pay it per month. This system can help them manage all their work, including managing employees, claims, leave, and payroll.

Even though most companies already use the Human Resources Management system, some companies still do not, especially Small and Medium-Sized Enterprises, also known as SME companies. SMEs are businesses with a sales turnover of less than RM20 million or less than 75 full-time workers. This concept includes all industries, including manufacturing, agriculture, construction, mining & quarrying, and services. Some SMEs companies still use the traditional method, manual methods. For instance, they still use paper or documents to save all the company's information, such as the employee's detail, medical leave letter, and reference receipt, to make a claim.

There are many disadvantages if using these traditional methods. For example, all the documents might be lost, and no backup copy exists. Besides, the paper will be messy, and it will be challenging to search for a specific record (*Disadvantages Of A Manual HR Administration: How They Can Affect Your Organization - SutiSoft*, n.d.). Due to this, it can be a severe problem if this traditional method is still used. There are several reasons why some SMEs still need to use a third-party system to manage their work. Some of the examples are because of the pricing of the system.

In addition, the pricing offered to the SME company to purchase the system is not affordable for them to pay. Based on the research, usually, each company need to pay the range of RM 6.00 to RM 40.00 per month and employee (*Top 10 HR Software in Malaysia - Software Reviews, Pricing, Comparison 2022 | Alternatives*, 2022). The price is relatively high for some SME companies to purchase it. Therefore, some SMEs want to avoid using and buying third-party systems. Besides that, even though a bunch of Human Resources systems can be found in the market, only some are suitable for the SME company. Several functions offered in the system are not appropriate and required for the company. Due to those reasons, that is why some SME companies intend to use the Human Resources system.

Hence, to overcome this problem, a proposed system, the Human Resources System for Small and Medium-Sized Enterprises or also known as SME-HR, will be the perfect solution for the SME company to solve their problem. This Human Resources system will focus more on SME companies which will benefit them. The benefits are that it can help them organize all the documents and efficiently manage their work. In addition, this proposed system will not be expensive because it will use an open resource to develop it. Due to that, it will not be a burden for the SME company to purchase the system.

1.3 Objective

There are three objectives in this project, as stated below:

i) To study the proper method to organize the document from the human resources department for SMEs.

ii) To develop a responsive web-based system for the human resources department for SMEs.

iii) To test the functionality of the purpose system, which is "Human Resources System for Small and Medium-Sizes Enterprises.

1.4 Scope

The project scope is listed as shown below:

User

i) The system will focus on SME company workers, especially in the Human Resources department, to manage their work.

System

i) This system targets Small and Medium-Sized Enterprises, also known as SME companies.

ii) The user, which is the Human Resources Department, can use the system through a responsive web-based platform.

Development

i) The system will be developed using the PHP web framework which is Laravel.

ii) The database that will be used is MySQL.

1.5 Thesis Organization

This thesis consists of five chapters overall. Chapter 1 discusses this project's introduction, problem statement, objectives, scope, and thesis organization. This chapter's main purpose is to briefly discuss the proposed project and related issues.

Chapter 2 shall consist of a literature review of three existing systems similar to the proposed project. This chapter will compare the features of the system. In addition, this chapter shall discuss the relevance of comparing the existing and proposed systems.

Next, Chapter 3 shall focus on the methodology used to accomplish this project. This chapter will also focus on the type of hardware and software used and will consist of the Gantt chart for the whole project.

Chapter 4 of the research paper presents the project's outcomes and activities and an in-depth discussion of the User Acceptance Testing (UAT) report. This chapter comprehensively analyzes the achieved results, including a detailed examination of the project's methodologies and the corresponding outcomes. The UAT report offers valuable insights into the system's acceptance and usability, highlighting user feedback and satisfaction. Its inclusion enhances the credibility and impact of the research findings.

In the final chapter, Chapter 5, the research paper concludes by summarizing the project's findings, presenting future plans and recommendations, and providing an overall conclusion. It offers a concise overview of the research journey, highlights critical contributions, and suggests future exploration and development avenues. This chapter completely closes the research, emphasizing its significance and inspiring further investigation.

CHAPTER 2

LITERATURE REVIEW

2.1 Introduction

This chapter explains the three (3) existing systems similar to the proposed project: the Human Resources System for Small and Medium-Sized Enterprises. This explanation includes detailed information and highlights the advantages and disadvantages of the system. In addition, enhancements and fixes will be implemented based on comparisons with the current system. A better and more efficient approach may be developed from these ideas.

2.2 Existing Systems

Several existing Human Resources systems can be found on the internet, but for this chapter, only three (3) systems are selected. The three (3) systems are altHR, Ignite Human Resources Management System and AuroraHRM. These three systems were chosen based on their criteria and performance.

2.2.1 altHR

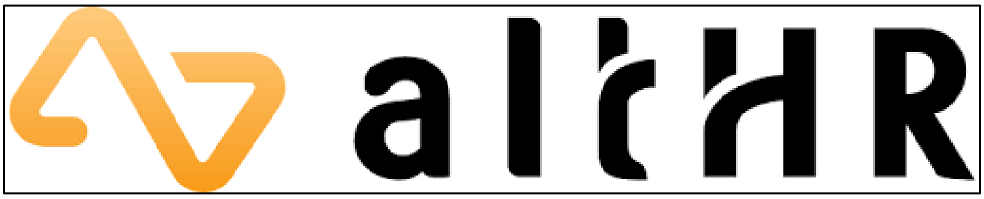
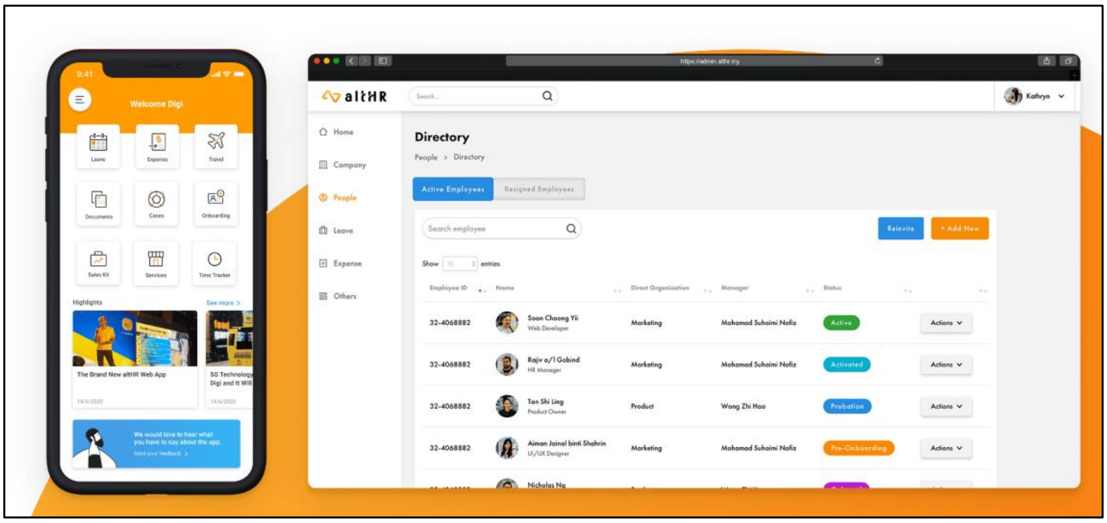


Figure 1: altHR Logo

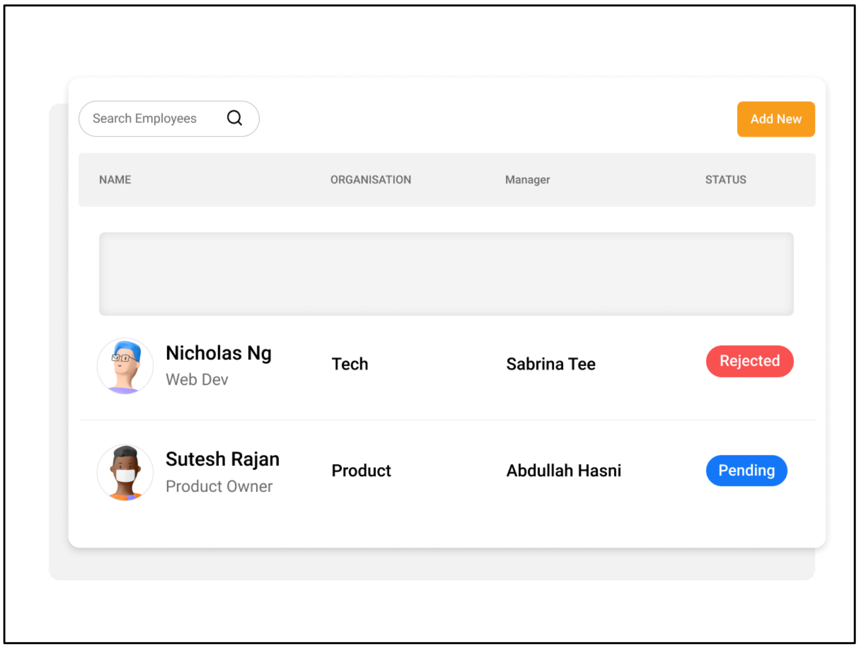
altHR system is a complete services application that provides several functions to help human resources manage their work (*AltHR Pricing | Affordable All-In-One HRMS System For Businesses*, n.d.). It is a comprehensive HR super app that enables businesses of all sizes to handle essential HR activities like leave entitlements, claims, employee profiles, and documentation. The main mission of this system is to help businesses better manage and nurture their most valuable resource, including their employee's information. Thus, this system has been developed based on solution advances in Benefits and Compensation, Employee Engagement, and Performance Management. In addition, the altHR system can assist businesses in better managing remote workers and boosting productivity with Digital Workspace as the digital and flexible work style becomes the norm in the wake of a pandemic.



Employee ID	Name	Direct Organisation	Manager	Status	
32-4068882	Soon Choong Yii Web Developer	Marketing	Mohamad Suhaimi Nafiz	Active	Actions
32-4068882	Rajiv a/l Gobind HR Manager	Marketing	Mohamad Suhaimi Nafiz	Activated	Actions
32-4068882	Tan Shi Ling Product Owner	Product	Wong Zhi Hao	Probation	Actions
32-4068882	Aiman Jainal binti Shahrin UI/UX Designer	Marketing	Mohamad Suhaimi Nafiz	Pre-Onboarding	Actions
	Nicholas Ng				

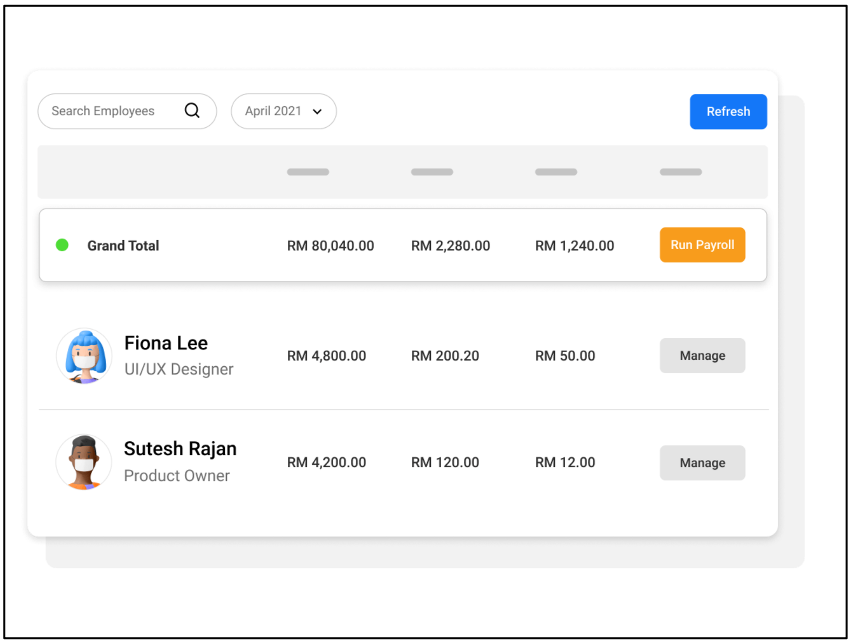
Figure 2: altHR System Mainpage

Several modules are provided in this system. The modules are people, leaves, payroll, directory and many more (*AltHR | Simplified HR Management For Businesses*, n.d.). The basic module provided in this system is the people. People stand for the employee that works under that company. Using this system, the company can manage its workforce's human resource needs in one location. It means fewer points of failure and a more streamlined, streamlined process. Moreover, keeping the team's data centralised is simple, and it can maintain a personnel database for easy access and management. Figure 3 shows the interface for the people module. Besides that, the next powerful module that has been provided is payroll. The unique thing about this system payroll module is that it has been approved by Lembaga Hasil Dalam Negeri Malaysia, also known as LHDN. It is a safe and compliant payroll solution that can streamline payroll and payment procedures. Due to the verification from the LHDN, the company can quickly and easily disburse employee wages. Figure 4 shows the interface for the payroll module.



NAME	ORGANISATION	Manager	STATUS
Nicholas Ng Web Dev	Tech	Sabrina Tee	Rejected
Sutesh Rajan Product Owner	Product	Abdullah Hasni	Pending

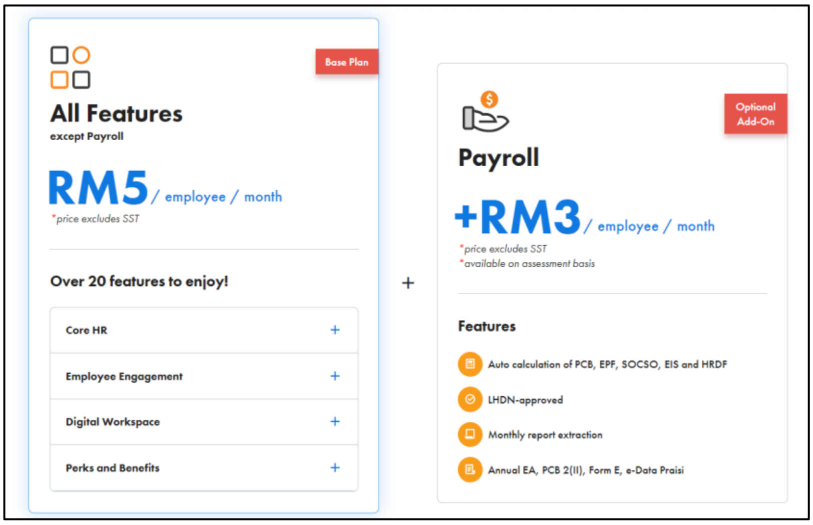
Figure 3: altHR People Module Interface



● Grand Total	RM 80,040.00	RM 2,280.00	RM 1,240.00	Run Payroll
Fiona Lee UI/UX Designer	RM 4,800.00	RM 200.20	RM 50.00	Manage
Sutesh Rajan Product Owner	RM 4,200.00	RM 120.00	RM 12.00	Manage

Figure 4: altHR Payroll Module Interface

The altHR app can be downloaded for free on mobile operating systems such as iOS and Android; however, the altHR admin dashboard can only be accessed through the online browser. The customer may choose between two plans, the first of which is the basic plan and the second of which is an optional add-on plan. Except for payroll, all modules will be covered by the basic plan, and the cost is RM 5.00 per employee and month. In addition, the client also has the option of adding the payroll module as an add-on to the plan. The customer is responsible for paying an additional fee of RM 3.00 per employee each month, bringing the total to RM 8.00 overall. Figure 5 shows the pricing plan for the altHR system.



Base Plan

All Features
except Payroll

RM5 / employee / month
*price excludes SST

Over 20 features to enjoy!

- Core HR +
- Employee Engagement +
- Digital Workspace +
- Perks and Benefits +

+

Optional Add-On

Payroll

+RM3 / employee / month
*price excludes SST
*available on assessment basis

Features

- Auto calculation of PCB, EPF, SOCSO, EIS and HRDF
- LHDN-approved
- Monthly report extraction
- Annual EA, PCB 2(II), Form E, e-Data Praisi

Figure 5: altHR Pring Plan

2.2.2 Ignite Human Resources Management System

Figure 6: Ignite HRMS Logo

Ignite HRMS is a system that BoardRoom Corporation developed. BoardRoom Corporation is a company that offers several business solutions, such as corporate and consulting services, including delivering accurate and dependable corporate solutions for all aspects of the organization. Ignite HRMS is one of the top selling services that BoardRoom Corporation has provided. Ignite HRMS is a system that focuses on five core modules: Leave, Claims, Time & Attendance, Personnel, and Payroll (*Ignite HRMS Payroll System | True Multi-Country HR & Payroll Processing*, n.d.). The Ignite Human Resource Management System (HRMS) is guaranteed to increase productivity and make HR and payroll operations easier for their client. Moreover, this system is designed for the future, removing the need for several payroll systems, and giving exceptional reporting features and an entire online multi-country payroll processing experience.

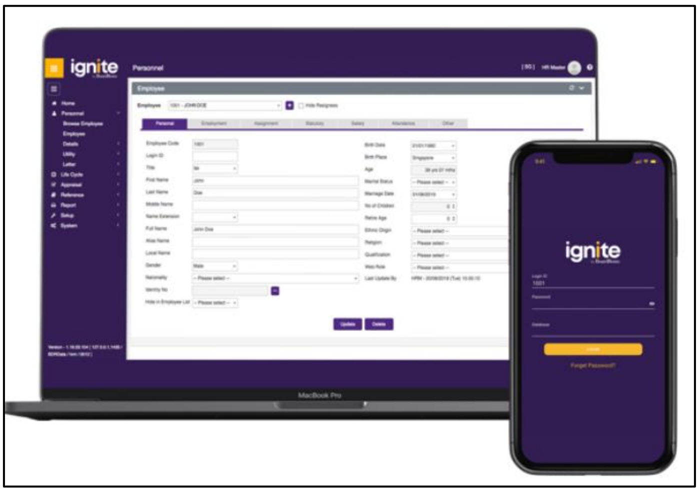
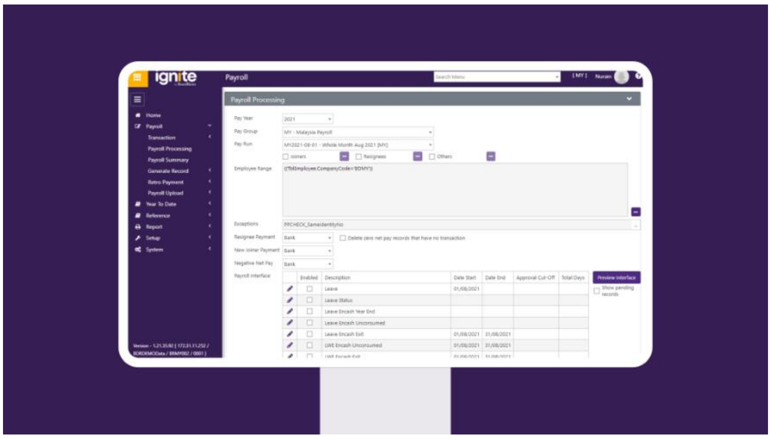


Figure 7: Ignite HRMS Interface

Ignite HRMS is more focused on its payroll module, but there is also another module that supports the system. Processing payroll once a month may be a time-consuming and frustrating task. Using this system, the company can automate the payroll processing tasks and make these a snap. Besides that, other modules are personnel, leave, attendance, and claims. Ignite is a valuable tool for managers and human resource professionals because it allows for the organization and storage of crucial personnel information. Next, employees may apply for annual, sick, maternity, and other types of leave for the leave module and quickly determine their leave balance without involving HR. In addition, the attendance module can monitor shift additions and deletions in real-time and promptly communicate the changes in the staff. With the Ignite app's calendar, the client can see at a glance who is available for a call and how many shifts they have done, making it simple to schedule changes and call-in backup. In the claim module, the client can apply for claims whenever it's convenient for the company, and it even allows them to attach supporting papers, even from their mobile device. Figure 8, Figure 9, Figure 10, Figure 11, and Figure 12 show the interface for the module that has been provided in the system.



Pay Year 2021

Pay Group MY - Malaysia Payroll

Pay Run MY2021-08-01 - Whole Month Aug 2021 [MY]

☐ Joiners ☐ Resignees ☐ Others

Employee Range ([TblEmployee.CompanyCode='BDMY'])

Exceptions PRCHECK_SameIdentityNo

Resignee Payment Bank ☐ Delete zero net pay records that have no transaction

New Joiner Payment Bank

Negative Net Pay Bank

Payroll Interface

Enabled	Description	Date Start	Date End	Approval Cut-Off	Total Days
☐	Leave	01/08/2021			
☐	Leave Status				
☐	Leave Encash Year End				
☐	Leave Encash Unconsumed				
☐	Leave Encash Exit	01/08/2021	31/08/2021		
☐	LWE Encash Unconsumed	01/08/2021	31/08/2021		
☐	LWE Encash Exit	01/08/2021	31/08/2021		

☐ Show pending records

Figure 8: Ignite HRMS Payroll Module

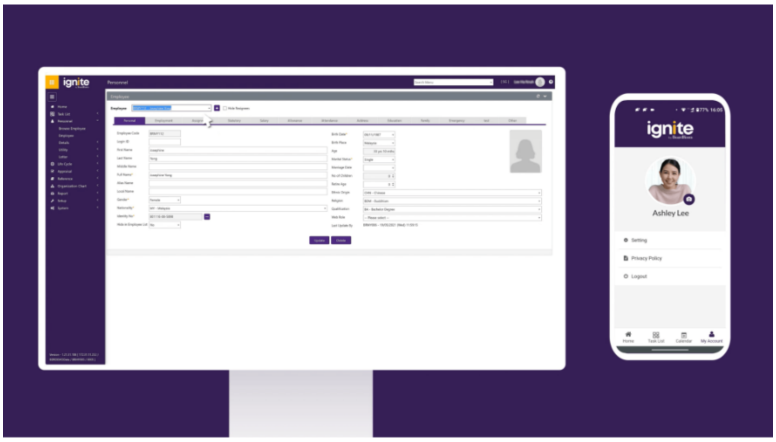
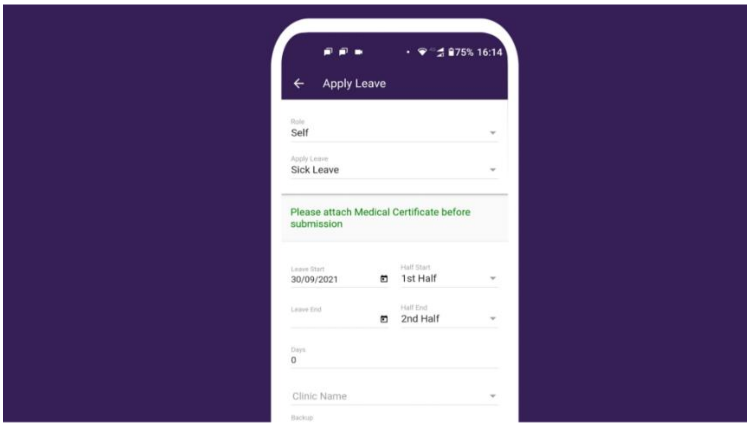


Figure 9: Ignite HRMS Personnel Module



Role Self

Apply Leave Sick Leave

Please attach Medical Certificate before submission

Leave Start 30/09/2021 Half Start 1st Half

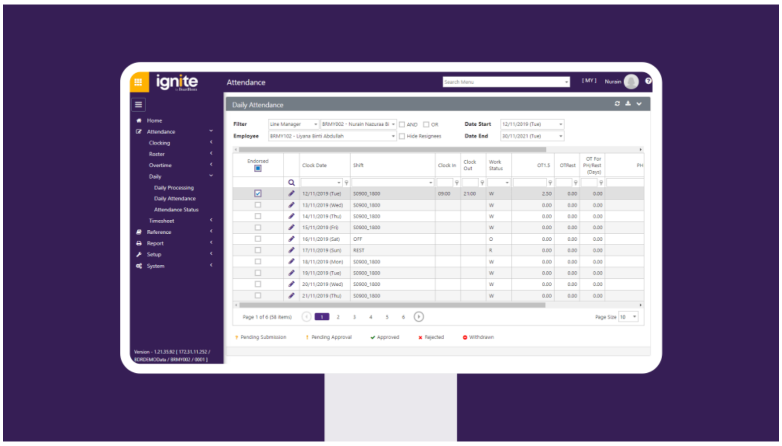
Leave End Half End 2nd Half

Days 0

Clinic Name

Backup

Figure 10: Ignite HRMS Leave Module



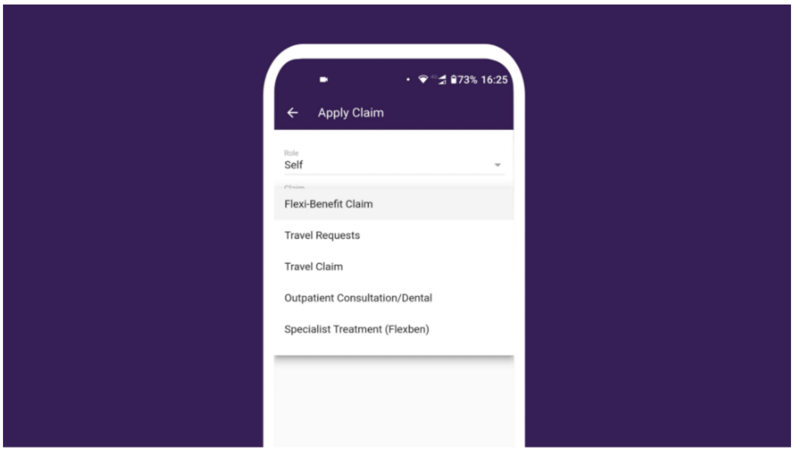
Version - 1.21.35.92 [172.31.11.252 / BDRDEMOData / BRMY002 / 0001]

Filter: Line Manager; BRMY002 - Nurain Nazuraa Bi; □ AND □ OR
Employee: BRMY102 - Liyana Binti Abdullah; □ Hide Resignees
Date Start: 12/11/2019 (Tue)
Date End: 30/11/2021 (Tue)

Endorsed		Clock Date	Shift	Clock In	Clock Out	Work Status	OT1.5	OTRest	OT For PH/Rest (Days)	PH
☑		12/11/2019 (Tue)	S0900_1800	09:00	21:00	W	2.50	0.00	0.00	
□		13/11/2019 (Wed)	S0900_1800			W	0.00	0.00	0.00	
□		14/11/2019 (Thu)	S0900_1800			W	0.00	0.00	0.00	
□		15/11/2019 (Fri)	S0900_1800			W	0.00	0.00	0.00	
□		16/11/2019 (Sat)	OFF			O	0.00	0.00	0.00	
□		17/11/2019 (Sun)	REST			R	0.00	0.00	0.00	
□		18/11/2019 (Mon)	S0900_1800			W	0.00	0.00	0.00	
□		19/11/2019 (Tue)	S0900_1800			W	0.00	0.00	0.00	
□		20/11/2019 (Wed)	S0900_1800			W	0.00	0.00	0.00	
□		21/11/2019 (Thu)	S0900_1800			W	0.00	0.00	0.00	

Page 1 of 6 (58 items) 1 2 3 4 5 6 Page Size 10

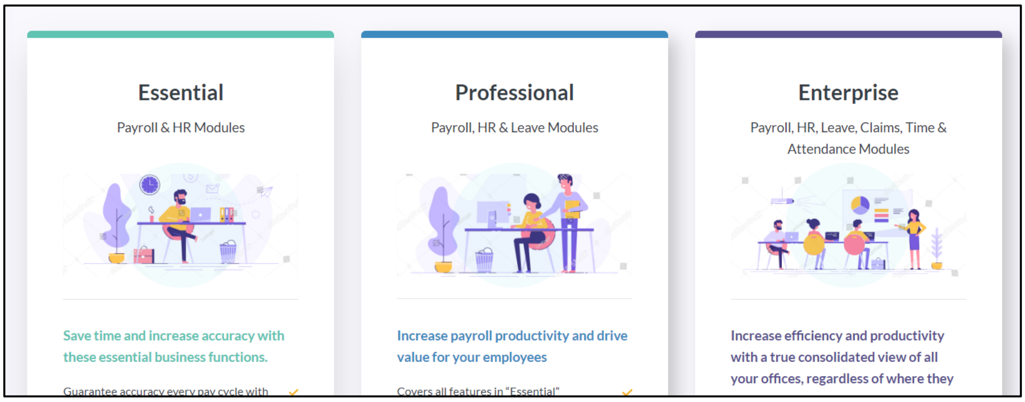
Figure 11: Ignite HRMS Attendance Module



Role: Self

Figure 12: Ignite HRMS Claims Module

This system can use both platforms, using a mobile and a desktop. Moreover, this system is supported by operating systems IOS and Android. Due to that, it can be easier for the staff to install it on their phone. In addition, this system offered three packages for their client. The packages are Essential, Professional, and Enterprise. Essential packages provide two modules, which are Payroll and the HR modules. Next, for the Professional packages, the modules provided are Payroll, HR, and Leave modules, while the last packages, Enterprise packages, provide Payroll, HR, Leave Claims, and Time & Attendance modules. Figure 13 shows the packages that Ignite HRMS offers.



Essential

Payroll & HR Modules

Save time and increase accuracy with these essential business functions.

Guarantee accuracy every pay cycle with ✓

Professional

Payroll, HR & Leave Modules

Increase payroll productivity and drive value for your employees

Covers all features in "Essential" ✓

Enterprise

Payroll, HR, Leave, Claims, Time & Attendance Modules

Increase efficiency and productivity with a true consolidated view of all your offices, regardless of where they

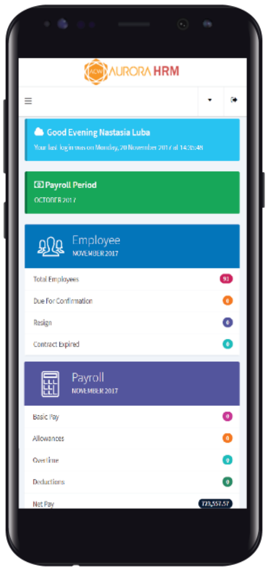
Figure 13: Ignite HRMS Packages

2.2.3 AuroraHRM

Figure 14: AuroraHRM Logo

AuroraHRM, an integrated Human Resource Management system, may facilitate the management of a company's human resources. It has a modular design that comprises more than ten components that combine effortlessly. AuroraHRM is versatile enough to fit the client firm’s demands and scalable enough to accommodate future growth in the company. The administration of human resources is made more straightforward, more accurate, and more productive with each new iteration of the AuroraHRM software (*Product - Aurora Cloud Works*, n.d.).

In order to effectively manage the workforce, AuroraHRM provides a human resource management system that is both extensive and powerful. This system may assist in automating and optimising day-to-day HR processes. The programme was developed specifically for HR experts to provide the HR department with the tools necessary to actively support the company's objectives while enhancing HR competency. Figure 15 shows the interface for the AuroraHRM system.



Good Evening Nastasia Luba

Your last login was on Monday, 20 November 2017 at 14:35:48

Payroll Period

OCTOBER 2017

Employee

NOVEMBER 2017

Total Employees	91
Due For Confirmation	0
Resign	0
Contract Expired	0

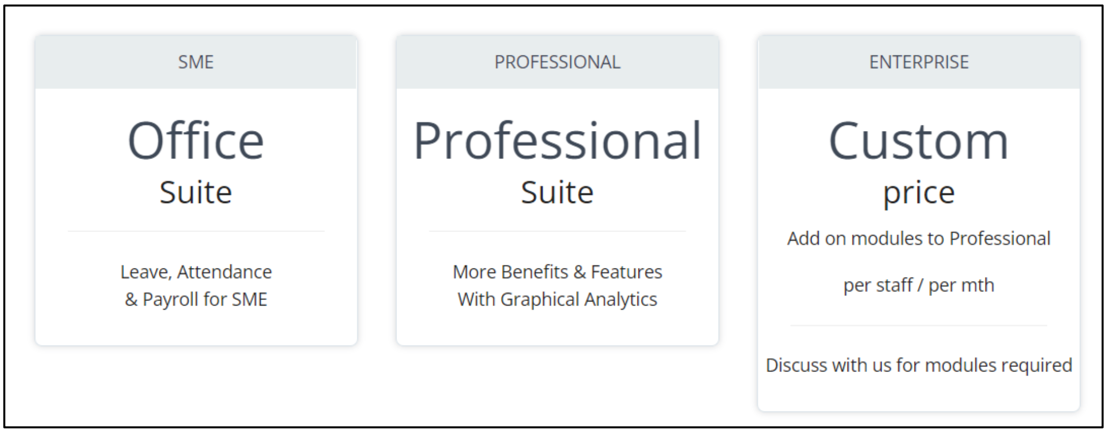
Payroll

NOVEMBER 2017

Basic Pay	0
Allowances	0
Overtime	0
Deductions	0
Net Pay	123,557.57

Figure 15: AuroraHRM Interface

AuroraHRM system provided 18 modules for their client, and the modules are Employee, Payroll, Financial System Integration, Leave, Attendance, Overtime, Benefit, Claim, Training, Performance, Manpower, Job Portal, Recruitment, Succession Plan, Aurora Query Writer, Dashboard, Disciplinary, and Achievement & Award. Each module can provide good benefits to their client company. This system is available in a web browser and mobile application platform. Besides that, the pricing has been categorized into three plans. The plan is SME Suite, Professional Suite, and Custom Price (*Features - Aurora Cloud Works*, n.d.). Each plan consists of a different module; for example, the modules provided are Leave, Attendance and Payroll in SME Suite Plan. Next, for the Professional Suite plan, the modules provided are the same as the SME Suite plan but have additional benefits and features with graphical analytics. For the Custom Plan, the client can customize the module they want in the system. Due to that, the price will depend on what modules the client chooses. Figure 16 shows the plan for AuroraHRM.



SME

Office
Suite

Leave, Attendance
& Payroll for SME

PROFESSIONAL

Professional
Suite

More Benefits & Features
With Graphical Analytics

ENTERPRISE

Custom
price

Add on modules to Professional

per staff / per mth

Discuss with us for modules required

Figure 16: AuroraHRM Plan

2.3 Analysis Comparison of Existing System

2.3.1 Analysis of comparison of the existing system with the proposed project

This section compares the existing systems: altHR, Ignite HRMS, AuroraHRM and the proposed system. Based on the literature review findings, the three current HR systems have been compared. The comparison includes the scope of the project, the platform that the system used, and the features that have been provided in the system. Table 1 shows the comparison of the three existing systems.

Table 1: Comparison of Three Existing System

System Name	**altHR**	**Ignite HRMS**	**AuroraHRM**	**SME-HR**
Scope	Organizations of any size may benefit from using an HR management system.	Organizations of any size may benefit from using an HR management system.	Organizations of any size may benefit from using an HR management system.	Focus on Small and Medium-Sized Enterprises

Platform	• Android • iPhone Operating System (IOS) • Web-based	• Android • iPhone Operating System (IOS) • Web-based	• Android • iPhone Operating System (IOS) • Web-based	• Responsive Web-based system
Features	• Payroll • Leaves • Status check-in • Directory • Rostering • Expenses	• Payroll • Leave • Claim • Attendance • Personnel	• Payroll • Leave • Claim • Employee • Attendance • Financial system integration • Overtime • Benefit • Training • Performance • Manpower • Job portal • Recruitment	• Payroll • Leave • Claim • Employee • EA form

			• Succession plan • Aurora query writer • Dashboard • Disciplinary • Achievement & award	
Programming Language	• Java • Kotlin • Swift • PHP • HTML • SQL • JavaScript	• Dart • PHP • HTML • SQL • JavaScript	• Java • Swift • Dart • PHP • HTML • SQL • JavaScript	• PHP • HTML • SQL • JavaScript

2.3.2 Advantages and Disadvantages

Table 2: Table of Advantages and Disadvantages

System Name	**altHR**	**Ignite HRMS**	**AuroraHRM**	**SME-HR**
Advantages	• Available on all platforms • LHDN approves the payroll module. • Provides training in the form of documentation.	• Available on all platforms. • The user interface is user-friendly.	• Have many features option to choose from. • Available on all platforms. • Provides instruction through printed documents and direct instruction	• The user interface is user-friendly. • Provides training in the form of documentation.
Disadvantages	• The price is quite high. • The UI is not user-friendly. • Does not apply API. • Only provides support via online	• The features are too limited. • Does not apply API	• Does not have 24/7 live support. • Does not apply API	• The features are too limited.

2.3.3 Relevance of comparison with project title

The review of the three existing systems shows that each method has unique features that can help human resources manage their work. This review gives ideas on developing an excellent Human Resources system, especially for Small and Medium-Sized Enterprises. Based on the features that have been offered show that several modules need to be developed in the proposed project. Due to that, implementing this module fulfils all the needs from human resources. Besides that, it also gives a clear idea of what platform the proposed project needs to be developed. In conclusion, doing this study significantly benefits the proposed project.

2.4 Survey Data Analysis

In the upcoming section, the survey will conduct a thorough analysis and processing of survey data collected from small and medium-sized enterprises (SMEs). The survey utilized a set of questionnaires designed to gather information about the current problems and challenges faced by these companies. The data collected will be analyzed in depth to gain a better understanding of the issues faced by SMEs and to identify potential solutions to these problems. The survey will help to make informed decisions and recommendations to help SMEs overcome challenges and improve their performance.

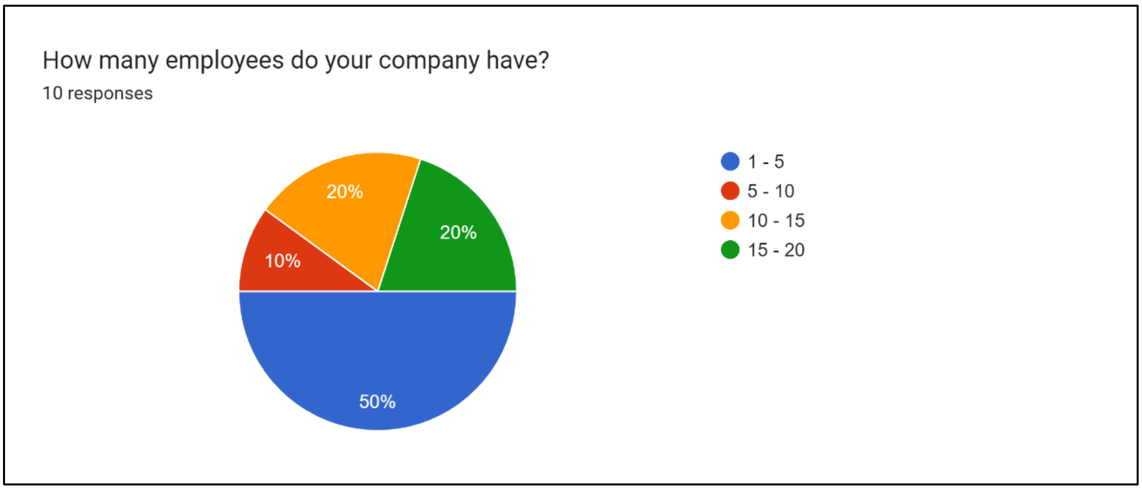


Figure 17: Data for Number of Employee for Each Responses

Based on Figure 17, it is clear that all of the responders in this study are from small and medium-sized enterprises (SMEs). This is indicated by the fact that the number of employees in these companies falls within the range of one (1) to 20 total employees. Additionally, it highlights the importance of understanding SMEs' unique challenges and opportunities in this industry or field of study. It is important to note that SMEs play a significant role in the economy, and it is crucial to understand their specific needs and challenges to support their growth and success.

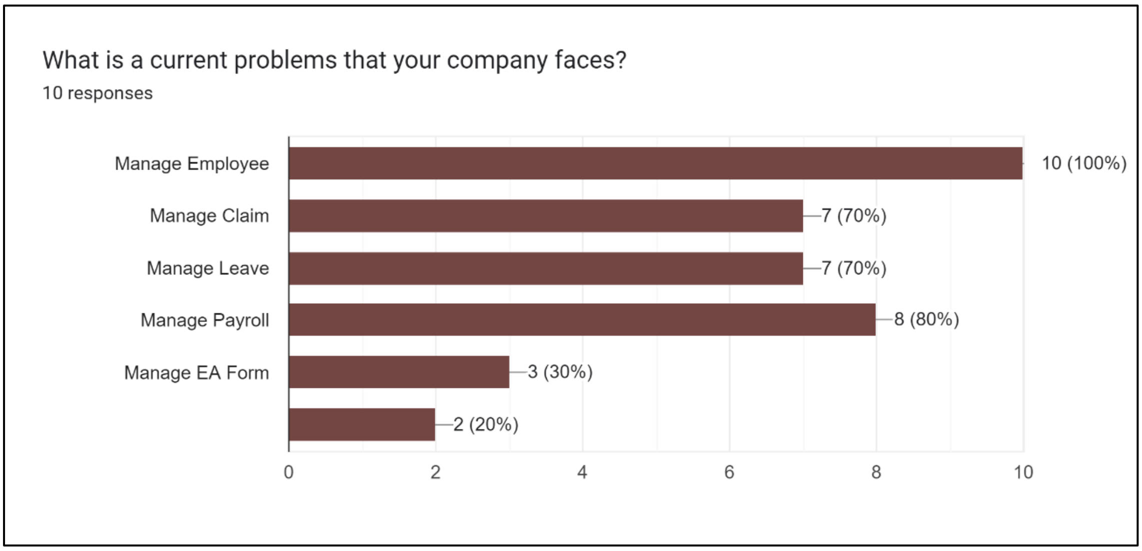


Figure 18: Data for the Current Problems That the Company Faces

Figure 18 shows the collected data for the current problems that the SME company faces. The data shows that managing employees is the primary concern for SME companies, with 100% of the responses indicating that it is an issue. Managing claims, leave, and payroll also received high percentages of responses, with 70% and 80%, respectively. Managing EA form received the lowest responses, with only 30%. Other issues received 20% of the answers. This data suggests that human resource management is a significant concern for SME companies, particularly regarding managing employees, claims, leaves, and payroll. Managing EA forms is less of a problem for these companies.

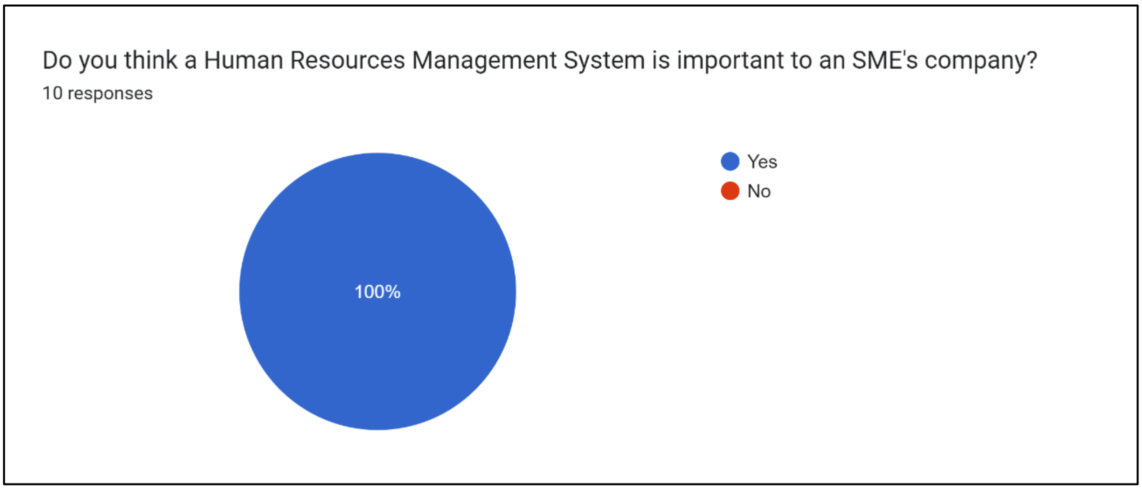


Figure 19: Data for the Important of Human Resources System to SME's Company

Figure 19 shows the collected data responses for the important of Human Resources system to SME’s company. Based on the data collected from 10 SME companies, 100% of the respondents believe that a Human Resources Management System is important for their company. No respondents indicated that they do not think a Human Resources Management System is important. This data collected suggests that most SMEs see the value and importance of implementing a Human Resources Management System in their operations.

2.5 Summary

Each of the three extant systems, the altHR, the Ignite HRMS, and the AuroraHRM, which have been discussed in this chapter, has its unique benefits and drawbacks. There are parallels between the three sets of procedures and the proposed new initiative. This literature study is being carried out to get ideas and serve as a guide for developing a Human Resources System for Small and Medium-Sized Enterprises.

CHAPTER 3

METHODOLOGY

3.1 Introduction

This chapter will describe the strategies and approaches used for the project. The methodology is a framework for organizing, planning, and regulating the process of building an information system. This strategy allows the project to be handled deliberately and methodically, emphasizing achieving the project's goals following its timeline and planning. Otherwise, the selected technique will be used to complete this project from conception to completion before presenting it to stakeholders.

3.2 Project Management Framework/Methodology

RAD (Rapid Application Development) is a methodology that emphasizes the rapid delivery of working software through rapid prototyping and iterative development. The RAD methodology was introduced in the late 1980s and early 1990s in response to the traditional Waterfall method, which was criticized for needing to be faster and more flexible (*The History of How Rapid Application Development Became Agile*, n.d.). RAD methodology focuses on quickly delivering a working version of the software, which can then be refined and improved based on user feedback (Abbas et al., 2008). This approach allows the development team to rapidly test and refine their ideas and make changes as needed in response to user feedback. RAD methodology is well suited to complex and rapidly changing projects, as it allows for constant adaptation and improvement based on user feedback.

One of the key benefits of the RAD methodology is its ability to rapidly deliver working software, which is achieved through rapid prototyping (*The Advantages and Disadvantages of the RAD Model - DistantJob - Remote Recruitment Agency*, n.d.). Creating prototypes early in the development process allows the development team to test and refine their ideas and make necessary changes quickly. This approach reduces the risk of building software that does not meet the stakeholders' requirements, as they are actively involved in the development process and can provide feedback at every stage.

Besides that, another advantage of RAD is that it encourages increased user involvement and feedback, which is crucial for ensuring that the final product meets the needs of the stakeholders (*13 RAD Benefits and Advantages You Could Certainly Expect*, n.d.). The iterative nature of the RAD methodology allows for regular interaction and communication between the development team and the stakeholders, which helps to ensure that the software is developed in a way that meets the stakeholders' needs and expectations.

RAD methodology consists of four phases. The four (4) phases are Requirements Planning, User Design, Construction, and Cutover (*Your Complete Guide To Rapid Application Development (RAD)*, n.d.).

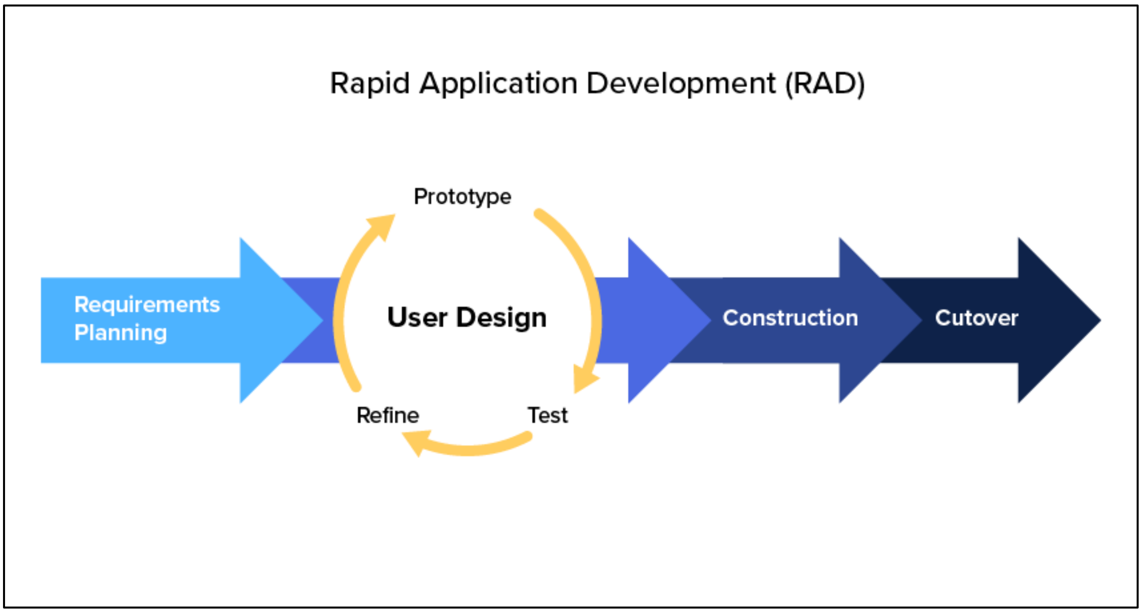


Figure 20: Rapid Application Development (RAD) Methodology

3.2.1 Requirements Planning

The Requirements Planning phase is the first step in the RAD (Rapid Application Development) methodology. In this phase, the developer will work with stakeholders to understand the requirements and needs of the human resources system, especially for the SME company. The goal of this phase is to gather information about the system so that the developer can create a design that meets the needs of the stakeholders. Stakeholders may include HR managers, employees, and other representatives of the SME.

During the Requirements Planning phase, the development team may engage in various activities, including identifying stakeholders, gathering information, documenting requirements, and prioritizing requirements.

3.2.2 User Design

The User Design phase of the RAD methodology is where the development team takes the information gathered during the Requirements Planning phase and uses it to create a preliminary design for the SME-HR system. The preliminary design may include creating wireframes or mockups of the system's user interface and functionalities. This phase aims to create a rough prototype of the system that can be tested and refined based on feedback from stakeholders.

3.2.3 Construction

The Construction phase of RAD methodology involves developing and implementing the human resources system. Based on the design specifications from the User Design phase, a functional software solution is created. The aim is to deliver a working software product that satisfies the SME's requirements and needs. The code is written, tested, debugged, and verified during this phase to meet specifications. Any issues that arise may be addressed by working with stakeholders. This phase is essential for ensuring the final product matches the SME and its stakeholders. The RAD methodology facilitates the delivery of a functioning software product, ensuring that the final product is a valuable asset for the organization.

3.2.4 Cutover

The Cutover phase marks the completion of the software development process and the transition of the human resources system into production use. In this phase, the system is finalized and ready for use by employees. The focus is on integrating the system into the SME's existing IT infrastructure, ensuring it is connected and can be used effectively. This may include training sessions for employees to familiarize them with the system and its features. The Cutover phase is critical for ensuring a seamless integration of the system into the SME's operations. During this phase, the system should be thoroughly tested to ensure it meets all requirements and performs as expected. This phase aims to deliver a high-quality, functional, and valuable human resources system to the SME.

3.3 Project Requirement

Several standards must be met for SME-HR to be successful. This requirement may aid in determining whether or not an application is successful or complete. Observing the specific process in the manual system is one of the several methods for determining the condition. Consequently, it may elicit the needed requirement. Finally, the requirements are acquired by researching the existing Human Resources system.

3.3.1 Functional Requirements

Functional Requirement List.

1. Login
2. Manage Employee
3. Manage Profile
4. Manage Payroll
5. Manage Leave
6. Manage Claim
7. Manage EA Form

Functional Requirement Description

Staff

- The system shall allow the staff to login to the system.

- The system shall allow the staff to manage their profile.
- The system shall allow the staff to generate the pay slip.
- The system shall allow the staff to make claims for specific things.
- The system shall allow the staff to apply for leave.
- The system shall allow the staff to view their claim and leave status.
- The system shall allow the staff to generate their EA form.

Human Resources

- The system shall allow the human resources to login to the system.
- The system shall allow human resources to manage employees.
- The system shall allow human resources to manage the claim.
- The system shall allow human resources to manage leaves.
- The system shall allow human resources to manage payroll.
- The system shall allow human resources to manage the EA of the staff.

Supervisor

- The system shall allow the supervisor to login to the system.
- The system shall allow the supervisor to approve the claim application.
- The system shall allow the supervisor to approve the leave application.
- The system shall allow the supervisor to view the employee’s information.
- The system shall allow the supervisor to claim and leave application status.

3.3.2 Non-Functional Requirements

Table 3: Non-Functional Requirement

Non-Functional Requirements	Description
Usability	The system should be intuitive for the user.
Performance	The system's response time must be below 200 milliseconds to prevent customer dissatisfaction.
Recoverability	To avoid data loss caused by hardware and system failure, the system should facilitate the backup and recovery of user-supplied data.
Privacy	The system should not leak the staff's data.

3.3.3 Constraints and Limitations

1. System compatibility

   - The system only supports Windows 7 and above only.
   - The system supports any latest browsers such as Google Chrome, Apple Safari, Microsoft Edge, and Mozilla Firefox.

2. Internet connectivity

- Users need an internet connection to access the system.

3. System limitation

- Only 100,000 users can access the system simultaneously (real-time users).
- Some of the features are limited to only being used within the Malaysia region.

3.4 Proposed Design

This chapter describes the proposed solution that can be used to develop this SME-HR system. This explanation includes the context diagram, use case diagram, use case description, and activity diagram. A better and more efficient approach may be developed from this proposed design.

3.4.1 Context Diagram

SME-HR (Human Resources for Small and Medium-Sized Enterprises) is a responsive web-based system that can allow the user which is Human Resources department to perform specific tasks regarding human resources management. The user can access the system using their smartphone and desktop since this system is a responsive web-based system.

There are three (3) types of users, which are Administrator, Staff, and Manager. Each of the users has their own responsibility for the system. The system should allow the user to manage the module successfully.

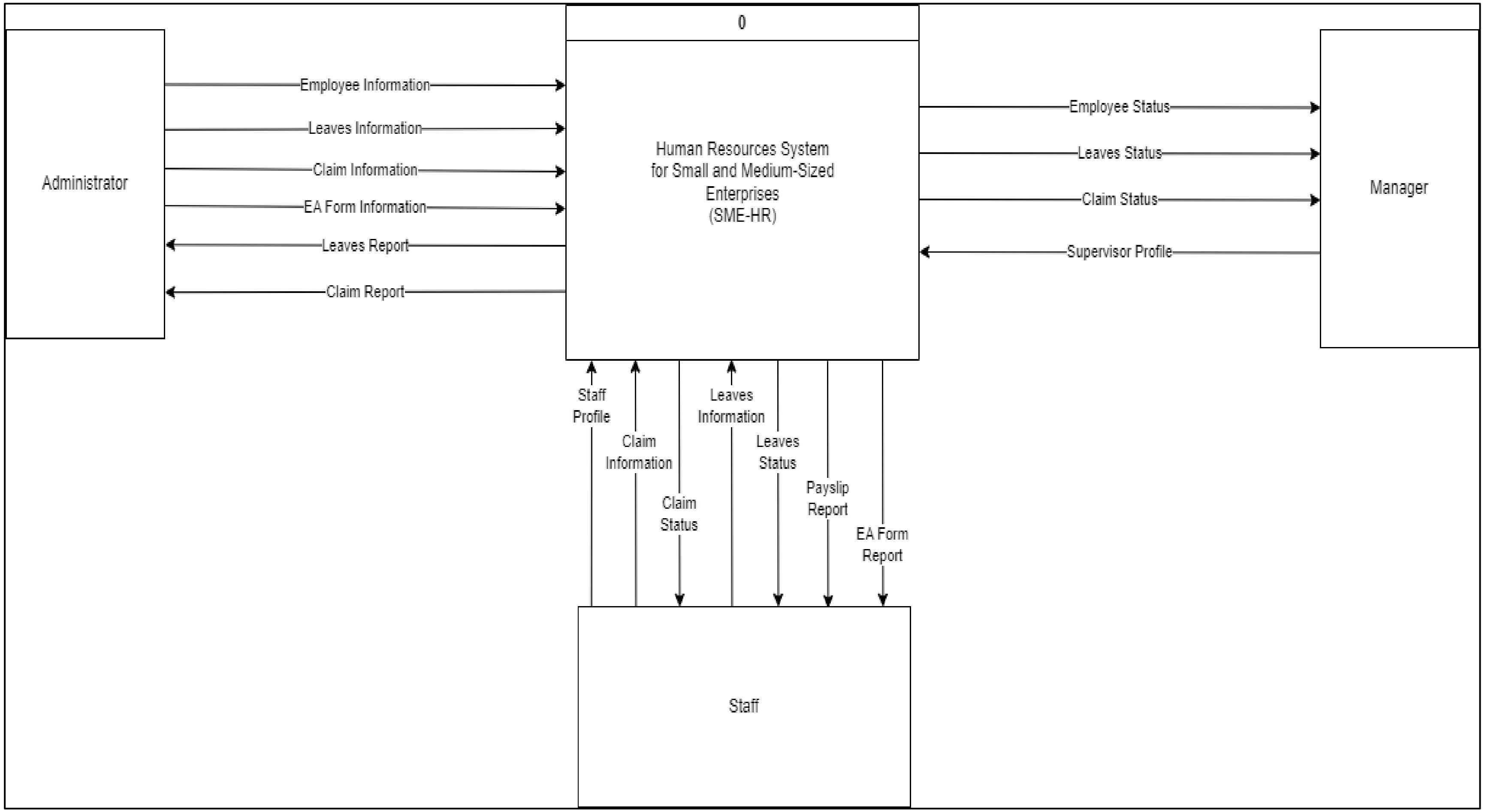


Figure 21: Context Diagram of SME-HR System

3.4.2 Use Case Diagram

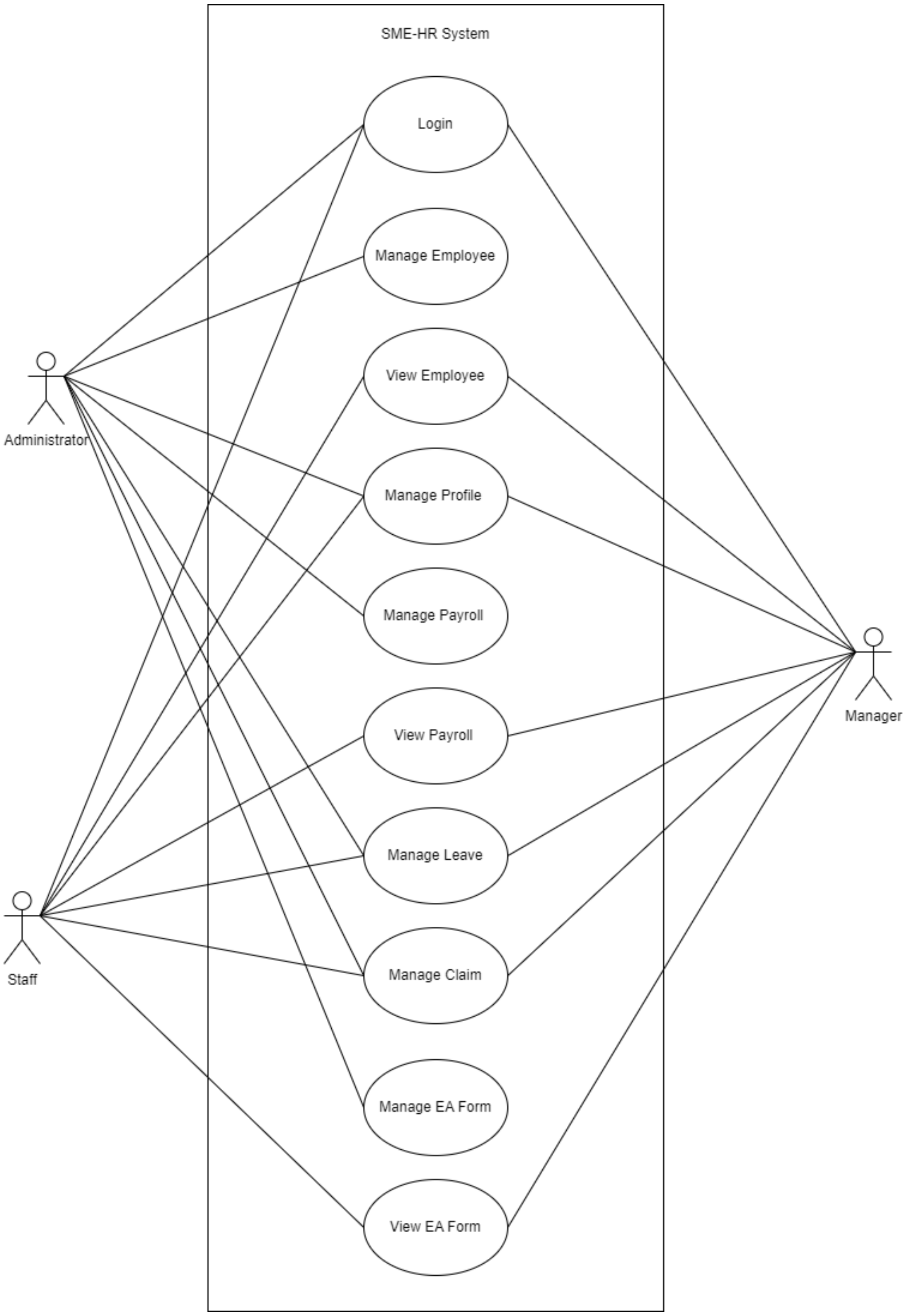


Figure 22: SME-HR Use Case Diagram

Figure 21 shows the use case diagram of SME-HR System. There is a total of 10 use cases:

1) Login
2) Manage Employee
3) View Employee
4) Manage Profile
5) Manage Payroll
6) View Payroll
7) Manage Leave
8) Manage Claim
9) Manage EA Form
10) View EA Form

3.4.3 Use Case Description

3.4.3.1 Login

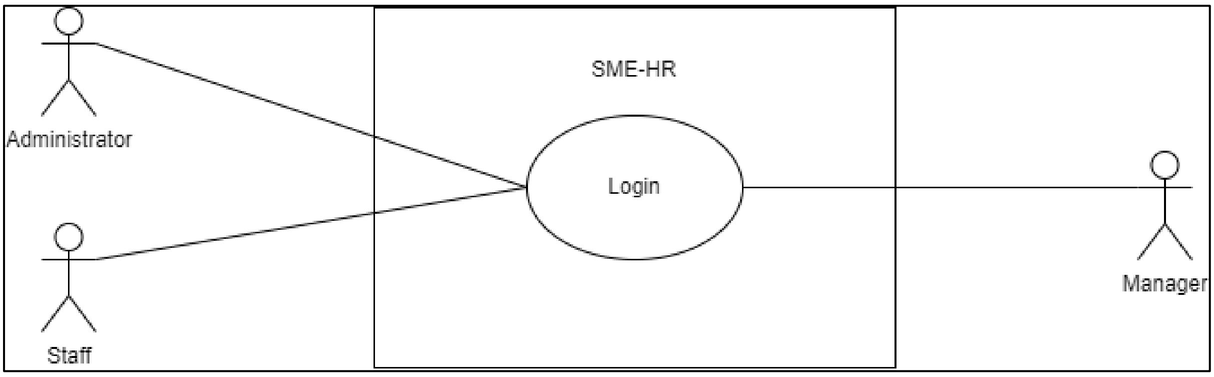


Figure 23: Login Use Case Diagram

Table 4: Login Use Case Description

Brief Description	This use case is initiated by the **Administrator**, **Staff**, and **Manager.** It provides the capability to log in to the system.
Actor	Administrator, Staff, Manager
Pre-Conditions	The administrator, Staff, and Manager must have a registered email and password.
Basic Flow	**User** (**Administrator, Staff, Manager)** 1. The use case begins when the user opens the SME-HR system. 2. If the user wants to login into the system, they need to insert the email and password. 3. The user clicks on the <<Login>> button. 4. The system checks the entered email and password. **[E1: Unregistered email] [E2: Incorrect Password] [R1: Registered Email] [R2: Correct Password]** 5. The system requests the details of the home page from the database. 6. The system retrieves the details of the home page from the database. 7. The system displays the home page. 8. If the user wants to log out, they need to click on the <<Logout>> button. 9. The system displays a login page. 10. The user is able to do the following functions if they forget their password: - **a) [A1: Forgot Password]** 11. The use case ends
Alternative Flow	**A1: Forgot Password** 1. The user clicks on the <<Forgot Password>> button. 2. The system displays re-enter email page. 3. The user enters the registered email.

	4. The system sends the password link to the given email. 5. The user clicks on the verification link. 6. The system displays the reset password page. 7. The user enters a new password. 8. The data is sent to the database. 9. The system displays the confirmation message “Do you want to proceed?” 10. The user clicks on the <<Yes>> button. 11. The data is saved in the database. 12. The system displays a successful message. 13. The use case continues with step 2 in the basic flow.
Exception Flow	**E1: Unregistered Email** 1. The system displays an error message that asks the user to enter a registered email. 2. The use case continues with step 2 in the basic flow. **E2: Incorrect Password** 1. The system displays an error message that asks the user to enter the correct password. 2. The use case continues with step 2 in the basic flow.
Post-Conditions	1. The system displays the updated homepage that is specific to the type of user that is currently logged in.
Rules	**R1: Registered Email** 1. The email must be registered to the system. **R2: Correct Password** 1. The password must be correct.
Constraints	Not applicable

3.4.3.2 Manage Employee

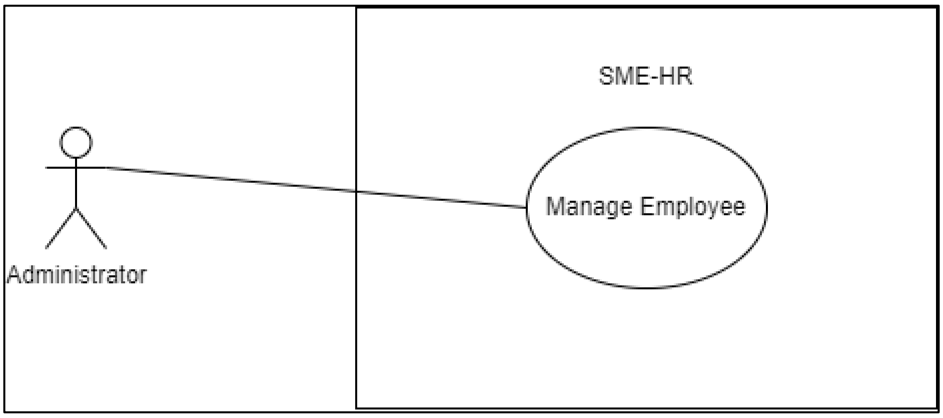


Figure 24: Manage Employee Use Case Diagram

Table 5: Manage Employee Use Case Diagram

Brief Description	This use case is initiated by the **Administrator.** It provides the capability to manage employees.
Actor	Administrator
Pre-Conditions	The user has already login to the system.
Basic Flow	**Administrator** 1. The use case begins when the Administrator clicks on the <<Employee>> button. 2. The system requests the employee page from the database. 3. The system retrieves the employee page from the database. 4. The system displays the Employee page. 5. If the Administrator wants to add a new employee, they need to click on the <<Add>> button. 6. The system requests the add page information from the database. 7. The system retrieves the add page information from the database. 8. The system displays the Add Employee page. 9. The Administrator enters the information of the new

	employee. 10. The Administrator clicks on the <<Save>> button. 11. The data is sent to the database. 12. The system displays the confirmation message “Do you want to proceed?”. 13. The Administrator clicks on the <<Yes>> button. 14. The data is saved in the database. 15. The system displays a successful message. 16. The Administrator is able to do the following functions if they want to view specific employee information: - **a) [A1: Search Employee]** 17. If the Administrator wants to view the detailed information of the specific employee, they need to click on the specific employee row. 18. The system requests detailed information from the database. 19. The system retrieves detailed information about the employee from the database. 20. The system displays the Employee Detail page. 21. If the Administrator wants to update the employee details, they need to click on the <<Update>> button. 22. The system requests the data. 23. The system retrieves the data from the database. 24. The system displays the Update Employee page. 25. The Administrator re-enters the employee information. 26. The Administrator clicks on the <<Save>> button. 27. The data is sent to the database. 28. The system displays the confirmation message “Do you want to proceed?”. 29. The Administrator clicks on the <<Yes>> button.

	30. The data is saved in the database. 31. The system displays a successful message. 32. If the Administrator wants to delete the employee information, they need to click on the <<Delete>> button. 33. The system displays the confirmation message “Do you want to proceed?”. 34. The Administrator clicks on the <<Yes>> button. 35. The data is deleted from the database. 36. The system displays a successful message. 37. The use case end.
Alternative Flow	**A1: Search Employee** 1. The Administrator enters the keyword in the search bar. **[E1: Invalid Keyword] [R1: Valid Keyword]** 2. The Administrator clicks on the <<Search>> button. 3. The system requests data with a similar keyword from the database. 4. The system retrieves the data from the database. 5. The system displays the result search. 6. The use case continues with step 17 in Administrator basic flow.
Exception Flow	**E1: Invalid Keyword** 1. The system displays an error message saying that “No Data Found!”. 2. The use case continues with step 1 in the alternative flow.
Post-Conditions	1. New employee information data appears on the Employee page. 2. The employee information is updated.
Rules	**R1: Valid Keyword** 1. The keyword must be valid and correct to search for specific information.
Constraints	Not applicable

3.4.3.3 View Employee

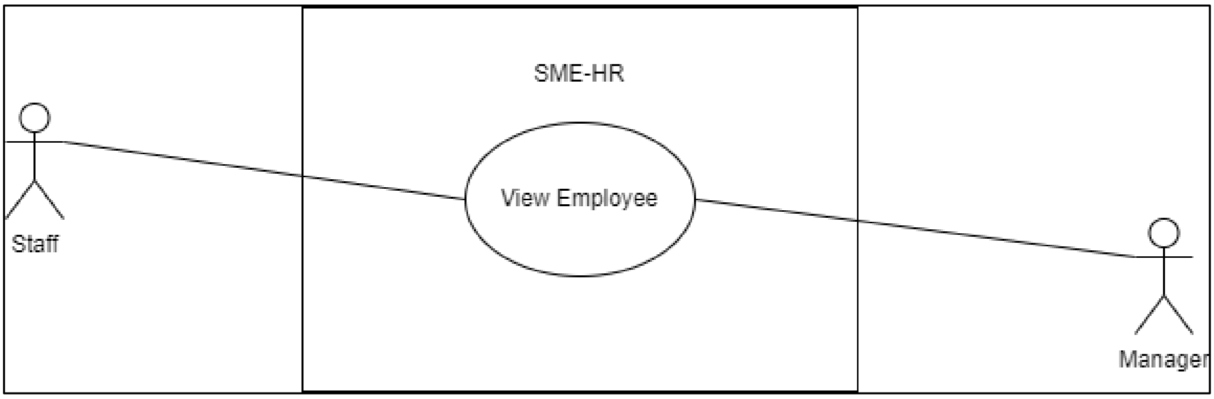


Figure 25: View Employee Use Case Diagram

Table 6: View Employee Use Case Description

Brief Description	This use case is initiated by the **Staff** and **Manager.** It provides the capability to view employees.
Actor	Staff, Manager
Pre-Conditions	The user has already login to the system.
Basic Flow	**Staff** 1. The use case begins when the Staff clicks on the <<Employee>> button. 2. The system requests the employee page from the database. 3. The system retrieves the employee page from the database. 4. The system displays the Employee page. 5. If the Staff want to view their detailed information, they need to click on the information column. 6. The system requests detailed information from the database. 7. The system retrieves detailed information about the employee from the database. 8. The system displays the Employee Detail page.

	9. The use case end. **Manager** 1. The use case begins when the Manager clicks on the <<Employee>> button. 2. The system requests the employee page from the database. 3. The system retrieves the employee page from the database. 4. The system displays the Employee page. 5. If the Manager wants to view the employee detail information, they need to click on the information column. 6. The system requests detailed information from the database. 7. The system retrieves detailed information about the employee from the database. 8. The system displays the Employee Detail page. 9. The use case end.
Alternative Flow	Not applicable
Exception Flow	Not applicable
Post-Conditions	Not Applicable
Rules	Not Applicable
Constraints	Not applicable

3.4.3.4 Manage Profile

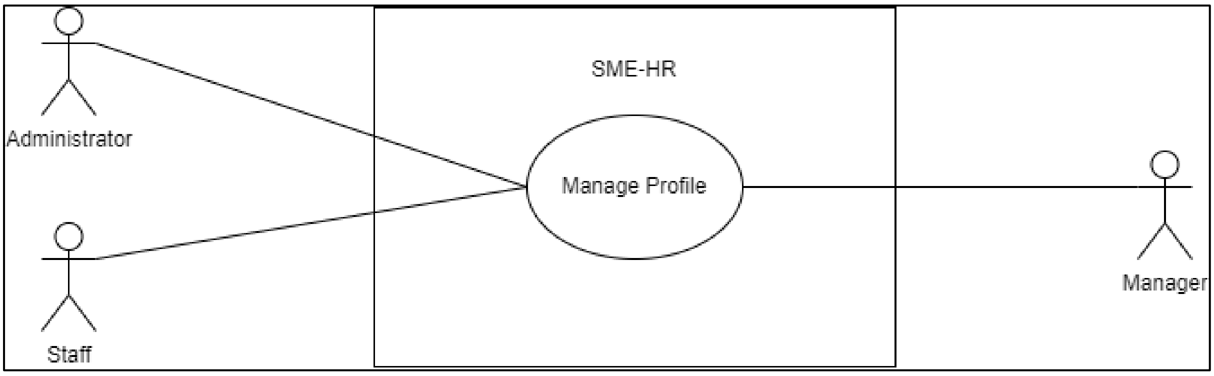


Figure 26: Manage Profile Use Case Diagram

Table 7: Manage Profile Use Case Description

Brief Description	This use case is initiated by the **Administrator**, **Staff**, and **Manager.** It provides the capability to manage profiles.
Actor	Administrator, Staff, Manager
Pre-Conditions	The user has already login to the system.
Basic Flow	**User (Administrator, Staff, Manager)** 1. The use case begins when the user clicks on the <<Profile>> button. 2. The system requests the profile page from the database. 3. The system retrieves the details of the profile page from the database. 4. The system displays all information on the Profile page. 5. If the user what to update their profile information, they can click on the <<Update>> button. 6. The system requests the details of the update page from the database. 7. The system retrieves the details for the update page from the database.

	8. The system displays the Update page. 9. The user re-enters the profile information. 10. The user clicks on the <<Save>> button. 11. The data is sent to the database. 12. The system displays the confirmation message "Do you want to proceed?". 13. The user clicks on the <<Yes>> button. 14. The data is saved in the database. 15. The system displays a successful message. 16. The use case end.
Alternative Flow	Not applicable
Exception Flow	Not applicable
Post-Conditions	1. The profile information is updated.
Rules	Not applicable
Constraints	Not applicable

3.4.3.5 Manage Payroll

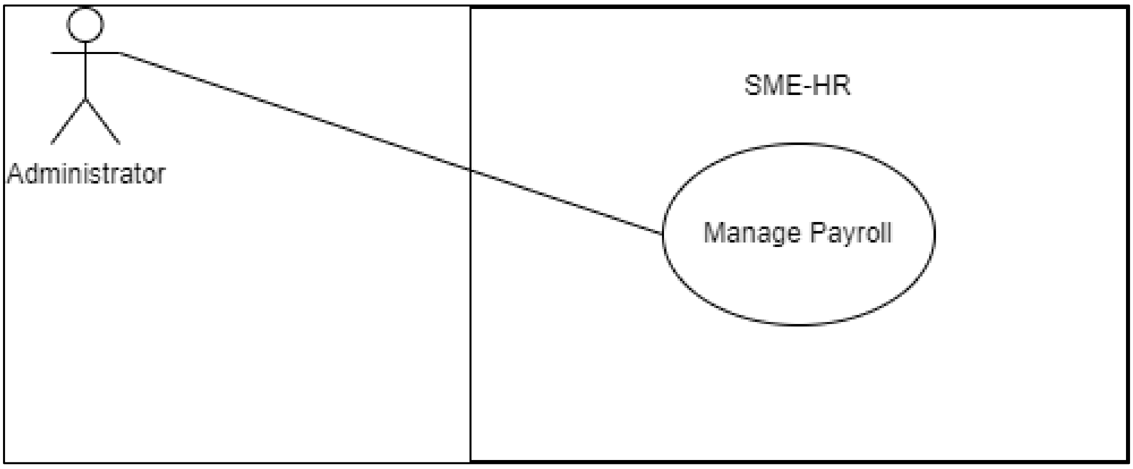


Figure 27: Manage Payroll Use Case Diagram

Table 8: Manage Payroll Use Case Description

Brief Description	This use case is initiated by the **Administrator**. It provides the capability to manage payroll.

Actor	Administrator
Pre-Conditions	The user has already login to the system.
Basic Flow	**Administrator** 1. The use case begins when the Administrator clicks on the <<Payroll>> button. 2. The system requests the payroll page from the database. 3. The system retrieves the payroll page from the database. 4. The system displays the payroll page. 5. If the Administrator wants to add the salary information, they need to click on the specific employee row. 6. The system requests detailed information from the database. 7. The system retrieves detailed information from the database. 8. The system displays the salary page. 9. The Administrator enters the salary information for the employee. 10. The Administrator clicks on the <<Save>> button. 11. The data is sent to the database. 12. The system displays the confirmation message “Do you want to proceed?”. 13. The Administrator clicks on the <<Yes>> button. 14. The data is saved in the database. 15. The system displays a successful message. 16. The Administrator is able to do the following functions if they want to view specific employee salary information: - **a) [A1: Search Payroll]** 17. If the Administrator wants to view the detailed salary information of the specific employee, they

	need to click on the specific employee row. 18. The system requests detailed salary information from the database. 19. The system retrieves detailed salary information from the database. 20. The system displays the Salary Information Page. 21. If the Administrator wants to update the employee salary information, they need to click on the <<Update>> button. 22. The system requests the data from the database. 23. The system retrieves the data from the database. 24. The system displays the Update page. 25. The Administrator re-enters the salary information. 26. The Administrator clicks on the <<Save>> button. 27. The data is sent to the database. 28. The system displays the confirmation message "Do you want to proceed?". 29. The Administrator clicks on the <<Yes>> button. 30. The data is saved in the database. 31. The system displays a successful message. 32. The use case end.
Alternative Flow	**A1: Search Payroll** 1. The Administrator enters the keyword in the search bar. **[E1: Invalid Keyword] [R1: Valid Keyword]** 2. The Administrator clicks on the <<Search>> button. 3. The system requests the data with the similar keyword from the database. 4. The system retrieves the data from the database. 5. The system displays the result search. 6. The use case continues with step 17 in Administrator basic flow.
Exception Flow	**E1: Invalid Keyword** 1. The system displays an error message saying that

	"No Data Found!". 2. The use case continues with step 1 in alternative flow.
Post-Conditions	1. New employee salary information data appears in the payroll page. 2. The employee salary information is updated.
Rules	**R1: Valid Keyword** 1. The keyword must be valid and correct to search for specific information.
Constraints	Not applicable

3.4.3.6 View Payroll

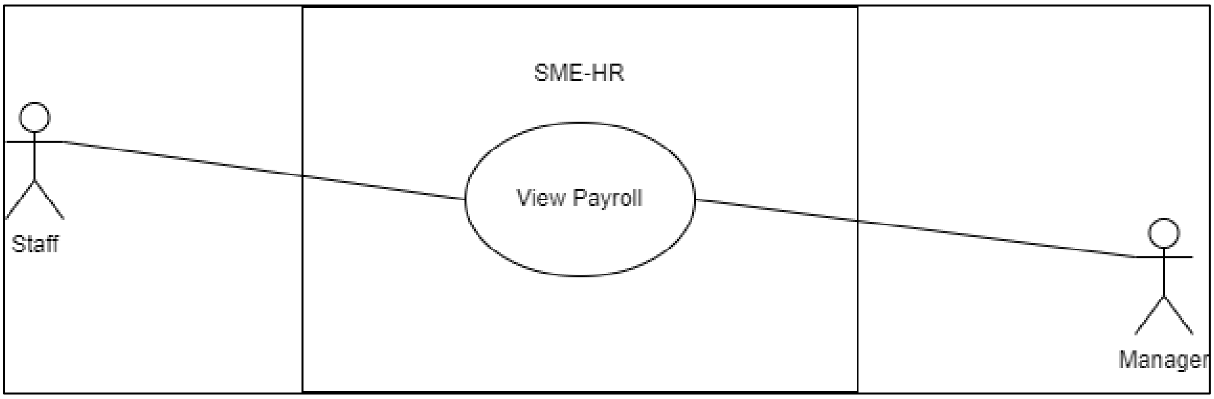


Figure 28: View Payroll Use Case Diagram

Table 9: View Payroll Use Case Diagram

Brief Description	This use case is initiated by the **Staff and Manager**. It provides the capability to view payroll.
Actor	Staff, Manager
Pre-Conditions	The user has already login to the system.
Basic Flow	**User (Staff, Manager)** 1. The use case begins when the user clicks on the <<Payroll>> button.

	2. The system requests the payroll page from the database. 3. The system retrieves the payroll page from the database. 4. The system displays the payroll page. 5. The user is able to do the following functions if they want to filter specific information: - **a) [A1: Search Pay slip]** **b) [A2: Filter Pay slip]** 6. If the user wants to view the pay slip, they need to click on the specific pay slip row. 7. The system requests detailed payroll from the database. 8. The system retrieves detailed payroll information from the database. 9. The system displays the Pay Slip Page. 10. The use case end.
Alternative Flow	**A1: Search Pay Slip** 1. The user enters the keyword in the search bar. **[E1: Invalid Keyword] [R1: Valid Keyword]** 2. The user clicks on the <<Search>> button. 3. The system requests data with a similar keyword from the database. 4. The system retrieves the data from the database. 5. The system displays the resulting search. 6. The use case continues with step 6 in basic flow. **A2: Filter Pay Slip** 1. The user clicks on the <<Filter>> button. 2. The user chooses to filter pay slips by year. 3. The system requests the data related to the filter keyword from the database. 4. The system retrieves the data from the database.

	5. The system displays the result filter. 6. The use case continues with step 6 in basic flow.
Exception Flow	**E1: Invalid Keyword** 1. The system displays an error message saying that "No Data Found!". 2. The use case continues with step 1 in alternative flow.
Post-Conditions	Not applicable
Rules	**R1: Valid Keyword** 1. The keyword must be valid and correct to search for specific information.
Constraints	Not applicable

3.4.3.7 Manage Leave

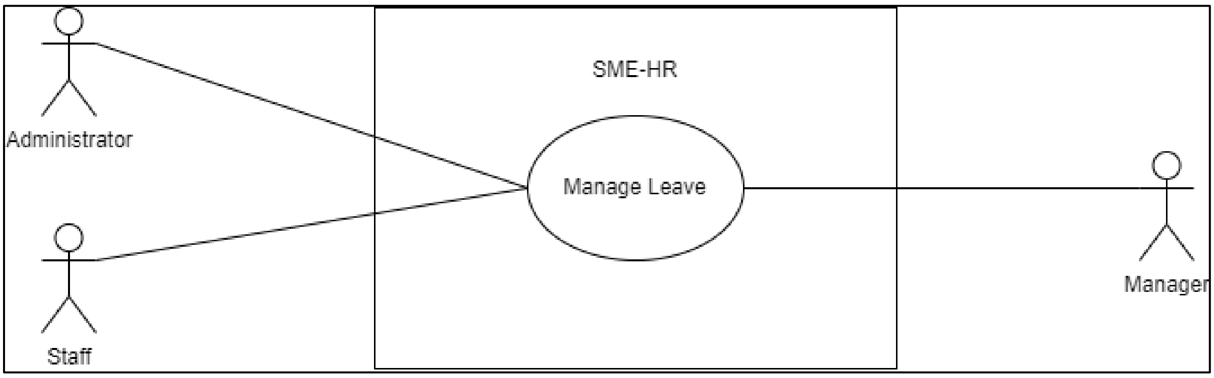


Figure 29: Manage Leave Use Case Diagram

Table 10: Manage Leave Use Case Description

Brief Description	This use case is initiated by the **Administrator, Staff, and Manager**. It provides the capability to manage leave.
Actor	Administrator, Staff, Manager
Pre-Conditions	The user has already login to the system.
Basic Flow	**Administrator**

	1. The use case begins when the Administrator clicks on the <<Leave>> button. 2. The system requests the leave page from the database. 3. The system retrieves the left page from the database. 4. The system displays the leave page. 5. If the Administrator wants to add the leave’s information, they need to click on the <<Add>> button. 6. The system requests the add page information from the database. 7. The system retrieves add page information from the database. 8. The system displays the Add Leave page. 9. The Administrator enters the leave information for the employee. 10. The Administrator clicks on the <<Save>> button. 11. The data is sent to the database. 12. The system displays the confirmation message “Do you want to proceed?”. 13. The Administrator clicks on the <<Yes>> button. 14. The data is saved in the database. 15. The system displays a successful message. 16. If the Administrator wants to apply for leave, they need to click on the <<Apply>> button. 17. The system requests the application leave page information from the database. 18. The system retrieves the application leave page information from the database. 19. The system displays the application leave page information form the database. 20. The Administrator enters the apply leave information.

	21. The Administrator clicks on the <<Apply>> button. 22. The data is sent to the database. 23. The system displays the confirmation message “Do you want to proceed?”. 24. The Administrator clicks on the <<Yes>> button. 25. The data is saved in the database. 26. The system displays a successful message. 27. The Administrator is able to do the following functions if they want to view specific apply leave information: - **a) [A1: Search Leave]** 28. If the Administrator wants to view the detailed leave application information of the specific employee, they need to click on the specific apply row. 29. The system requests detailed leave application information from the database. 30. The system retrieves detailed leave application information from the database. 31. The system displays the Leave Apply Information Page. 32. If the Administrator wants to update the leave information, they need to click on the <<Update>> button. 33. The system requests the data from the database. 34. The system retrieves the data from the database. 35. The system displays the Update page. 36. The Administrator re-enters the leave information. 37. The Administrator clicks on the <<Save>> button. 38. The data is sent to the database. 39. The system displays the confirmation message “Do you want to proceed?”. 40. The Administrator clicks on the <<Yes>> button. 41. The data is saved in the database.

	42. The system displays a successful message. 43. If the Administrator wants to delete the leave information, they need to click on the <<Delete>> button. 44. The system displays the confirmation message “Do you want to proceed?”. 45. The Administrator clicks on the <<Yes>> button. 46. The data is deleted from the database. 47. The system displays a successful message. 48. The use case end. **Staff, Manager** 1. The use case begins when the Staff or Manager clicks on the <<Leave>> button. 2. The system requests the leave page from the database. 3. The system retrieves the leave page from the database. 4. The system displays the leave page. 5. If the Staff or Manager wants to apply for leave, they need to click on the <<Apply>> button. 6. The system requests the application leave page information from the database. 7. The system retrieves the application leave page information from the database. 8. The system displays the application leave page information from the database. 9. The Staff or Manager enters the application leave information. 10. The Administrator clicks on the <<Apply>> button. 11. The data is sent to the database. 12. The system displays the confirmation message “Do you want to proceed?”.

	13. The Staff or Manager clicks on the <<Yes>> button. 14. The data is saved in the database. 15. The system displays a successful message. 16. The Staff or Manager is able to do the following functions if they want to view specific apply leave information: - **a) [A2: Search Leave Application]** 17. If the Staff or Manager wants to view the specific detailed leave application information, they need to click on the specific apply row. 18. The system requests detailed leave application information from the database. 19. The system retrieves detailed leave application information from the database. 20. The system displays the Leave Apply Information Page. 21. If the Staff or Manager wants to update the leave information, they need to click on the <<Update>> button. 22. The system requests the data from the database. 23. The system retrieves the data from the database. 24. The system displays the Update page. 25. The Staff or Manager re-enters the leave information. 26. The Staff or Manager clicks on the <<Save>> button. 27. The data is sent to the database. 28. The system displays the confirmation message “Do you want to proceed?”. 29. The Staff or Manager clicks on the <<Yes>> button. 30. The data is saved in the database. 31. The system displays a successful message. 32. If the Staff or Manager wants to delete the leave

	information, they need to click on the <<Delete>> button. 33. The system displays the confirmation message “Do you want to proceed?”. 34. The Staff or Manager clicks on the <<Yes>> button. 35. The data is deleted from the database. 36. The system displays a successful message. 37. The use case end.
Alternative Flow	**A1: Search Leave** 1. The Administrator enters the keyword in the search bar. **[E1: Invalid Keyword] [R1: Valid Keyword]** 2. The Administrator clicks on the <<Search>> button. 3. The system requests the data with the similar keyword from the database. 4. The system retrieves the data from the database. 5. The system displays the result search. 6. The use case continues with step 28 in Administrator basic flow. **A2: Search Leave Application** 1. The Staff or Manager enters the keyword in the search bar. 2. The Staff or Manager clicks on the <<Search>> button. 3. The system requests the data with the similar keyword from the database. 4. The system retrieves the data from the database. 5. The system displays the result search. 6. The use case continues with step 17 in Staff, Manager basic flow.
Exception Flow	**E1: Invalid Keyword** 1. The system displays an error message saying that “No Data Found!”.

	2. The use case continues with step 1 in alternative flow.
Post-Conditions	1. New employee leave information data appears in the payroll page. 2. The employee leave information is updated.
Rules	**R1: Valid Keyword** 1. The keyword must be valid and correct to search for specific information.
Constraints	Not applicable

3.4.3.8 Manage Claim

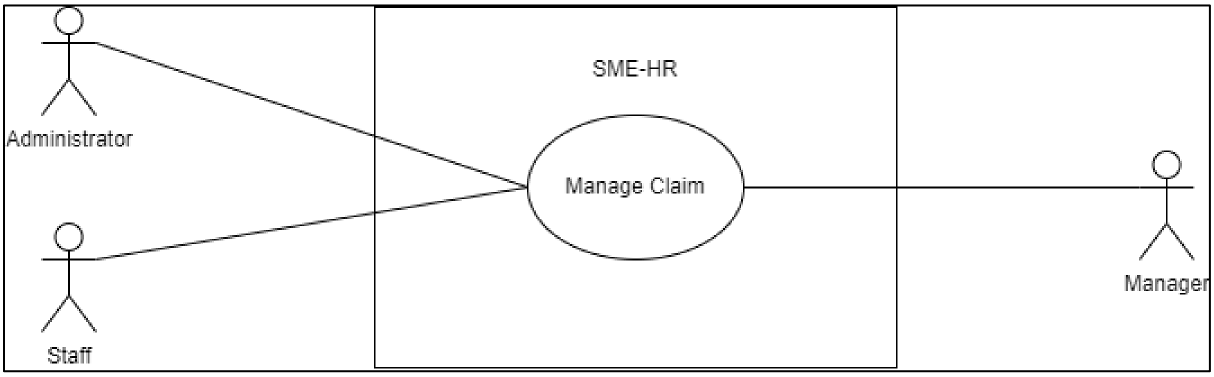


Figure 30: Manage Claim Use Case Diagram

Table 11: Manage Claim Use Case Description

Brief Description	This use case is initiated by the **Administrator, Staff, and Manager**. It provides the capability to manage claim.
Actor	Administrator, Staff, Manager
Pre-Conditions	The user has already login to the system.
Basic Flow	**Administrator** 1. The use case begins when the Administrator clicks on the <<Claim>> button. 2. The system requests the claim page from the database.

	3. The system retrieves the claim page from the database. 4. The system displays the Claim page. 5. If the Administrator wants to add the claim information, they need to click on the <<Add>> button. 6. The system requests the add page information from the database. 7. The system retrieves add page information from the database. 8. The system displays the Add Claim page. 9. The Administrator enters the claim information. 10. The Administrator clicks on the <<Save>> button. 11. The data is sent to the database. 12. The system displays the confirmation message “Do you want to proceed?”. 13. The Administrator clicks on the <<Yes>> button. 14. The data is saved in the database. 15. The system displays a successful message. 16. If the Administrator wants to apply for the claim, they need to click on the <<Apply>> button. 17. The system requests the application leave page information from the database. 18. The system retrieves the apply claim page information from the database. 19. The system displays the apply claim page information from the database. 20. The Administrator enters the apply claim information. 21. The Administrator clicks on the <<Apply>> button. 22. The data is sent to the database. 23. The system displays the confirmation message “Do you want to proceed?”.

	24. The Administrator clicks on the <<Yes>> button. 25. The data is saved in the database. 26. The system displays a successful message. 27. The Administrator is able to do the following functions if they want to view specific apply claim information: - **a) [A1: Search Claim]** 28. If the Administrator wants to view the detailed claim application information of the specific employee, they need to click on the specific row. 29. The system requests detailed claim application information from the database. 30. The system retrieves detailed claim application information from the database. 31. The system displays the Claim Apply Information Page. 32. If the Administrator wants to update the claim information, they need to click on the <<Update>> button. 33. The system requests the data from the database. 34. The system retrieves the data from the database. 35. The system displays the Update page. 36. The Administrator re-enters the claim information. 37. The Administrator clicks on the <<Save>> button. 38. The data is sent to the database. 39. The system displays the confirmation message “Do you want to proceed?”. 40. The Administrator clicks on the <<Yes>> button. 41. The data is saved in the database. 42. The system displays a successful message. 43. If the Administrator wants to delete the claim information, they need to click on the <<Delete>> button.

	44. The system displays the confirmation message “Do you want to proceed?”. 45. The Administrator clicks on the <<Yes>> button. 46. The data is deleted from the database. 47. The system displays a successful message. 48. The use case end. **Staff, Manager** 1. The use case begins when the Staff or Manager clicks on the <<Claim>> button. 2. The system requests the claim page from the database. 3. The system retrieves the claim page from the database. 4. The system displays the claim page. 5. If the Staff or Manager wants to apply for claim, they need to click on the <<Apply>> button. 6. The system requests the apply claim page information from the database. 7. The system retrieves the apply claim page information from the database. 8. The system displays the apply claim page information from the database. 9. The Staff or Manager enters the apply claim information. 10. The Administrator clicks on the <<Apply>> button. 11. The data is sent to the database. 12. The system displays the confirmation message “Do you want to proceed?”. 13. The Staff or Manager clicks on the <<Yes>> button. 14. The data is saved in the database. 15. The system displays a successful message. 16. The Staff or Manager is able to do the following

	functions if they want to view specific apply claim information: - **a) [A2: Search Claim Application]** 17. If the Staff or Manager wants to view the specific detailed claim application information, they need to click on the specific apply row. 18. The system requests detailed claim application information from the database. 19. The system retrieves detailed claim application information from the database. 20. The system displays the Claim Apply Information Page. 21. If the Staff or Manager wants to update the claim information, they need to click on the <<Update>> button. 22. The system requests the data from the database. 23. The system retrieves the data from the database. 24. The system displays the Update page. 25. The Staff or Manager re-enters the claim information. 26. The Staff or Manager clicks on the <<Save>> button. 27. The data is sent to the database. 28. The system displays the confirmation message “Do you want to proceed?”. 29. The Staff or Manager clicks on the <<Yes>> button. 30. The data is saved in the database. 31. The system displays a successful message. 32. If the Staff or Manager wants to delete the claim information, they need to click on the <<Delete>> button. 33. The system displays the confirmation message “Do you want to proceed?”.

	34. The Staff or Manager clicks on the <<Yes>> button. 35. The data is deleted from the database. 36. The system displays a successful message. 37. The use case end.
Alternative Flow	**A1: Search Claim** 1. The Administrator enters the keyword in the search bar. **[E1: Invalid Keyword] [R1: Valid Keyword]** 2. The Administrator clicks on the <<Search>> button. 3. The system requests the data with the similar keyword from the database. 4. The system retrieves the data from the database. 5. The system displays the result search. 6. The use case continues with step 28 in Administrator basic flow. **A2: Search Claim Application** 7. The Staff or Manager enters the keyword in the search bar. 8. The Staff or Manager clicks on the <<Search>> button. 9. The system requests the data with the similar keyword from the database. 10. The system retrieves the data from the database. 11. The system displays the result search. 12. The use case continues with step 17 in Staff, Manager basic flow.
Exception Flow	**E1: Invalid Keyword** 1. The system displays an error message saying that “No Data Found!”. 2. The use case continues with step 1 in alternative flow.
Post-Conditions	1. New employee claim information data appears in the claim page.

	2. The employee claim information is updated.
Rules	**R1: Valid Keyword** 1. The keyword must be valid and correct to search for specific information.
Constraints	Not applicable

3.4.3.9 Manage EA Form

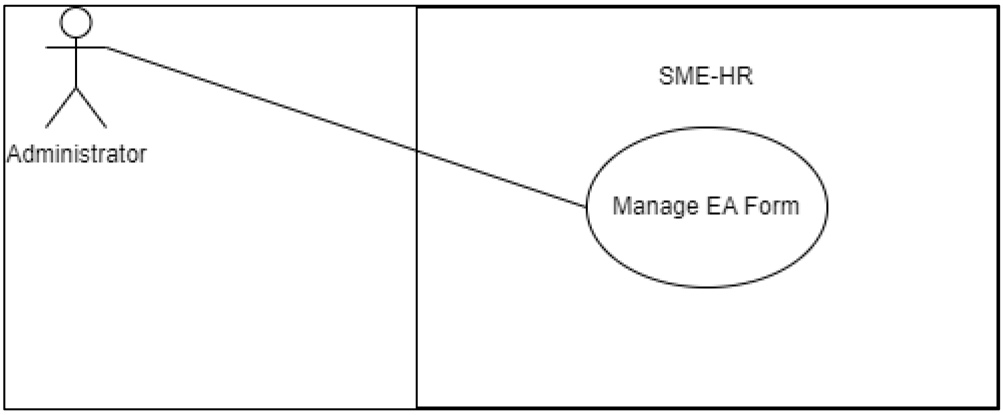


Figure 31: Manage EA Form Use Case Diagram

Table 12: Manage EA Form Use Case Description

Brief Description	This use case is initiated by the **Administrator**. It provides the capability to manage EA form.
Actor	Administrator
Pre-Conditions	The user has already login to the system.
Basic Flow	**Administrator** 1. The use case begins when the Administrator clicks on the <<EA Form>> button. 2. The system requests the EA page from the database. 3. The system retrieves the EA page from the database. 4. The system displays the EA page. 5. If the Administrator wants to add the EA information, they need to click on the <<Add>>

	button. 6. The system requests the add page information from the database. 7. The system retrieves add page information from the database. 8. The system displays the Add EA page. 9. The Administrator enters the EA information. 10. The Administrator clicks on the <<Save>> button. 11. The data is sent to the database. 12. The system displays the confirmation message “Do you want to proceed?”. 13. The Administrator clicks on the <<Yes>> button. 14. The data is saved in the database. 15. The system displays a successful message. 16. The Administrator is able to do the following functions if they want to view specific EA information: - a) **[A1: Search EA Form]** 17. If the Administrator wants to view the detailed EA form information of the specific employee, they need to click on the specific row. 18. The system requests detailed EA form information from the database. 19. The system retrieves detailed EA form information from the database. 20. The system displays the EA From Information Page. 21. If the Administrator wants to update the EA form information, they need to click on the <<Update>> button. 22. The system requests the data from the database. 23. The system retrieves the data from the database. 24. The system displays the Update page. 25. The Administrator re-enters the EA form

	information. 26. The Administrator clicks on the <<Save>> button. 27. The data is sent to the database. 28. The system displays the confirmation message "Do you want to proceed?". 29. The Administrator clicks on the <<Yes>> button. 30. The data is saved in the database. 31. The system displays a successful message. 32. If the Administrator wants to delete the EA form information, they need to click on the <<Delete>> button. 33. The system displays the confirmation message "Do you want to proceed?". 34. The Administrator clicks on the <<Yes>> button. 35. The data is deleted from the database. 36. The system displays a successful message. 37. The use case end.
Alternative Flow	**A1: Search EA Form** 1. The Administrator enters the keyword in the search bar. **[E1: Invalid Keyword] [R1: Valid Keyword]** 2. The Administrator clicks on the <<Search>> button. 3. The system requests the data with the similar keyword from the database. 4. The system retrieves the data from the database. 5. The system displays the result search. 6. The use case continues with step 17 in Administrator basic flow.
Exception Flow	**E1: Invalid Keyword** 1. The system displays an error message saying that "No Data Found!". 2. The use case continues with step 1 in alternative flow.
Post-Conditions	1. New employee EA form information data appears in

	the claim page. 2. The employee EA form information is updated.
Rules	**R1: Valid Keyword** 1. The keyword must be valid and correct to search for specific information.
Constraints	Not applicable

3.4.3.10 View EA Form

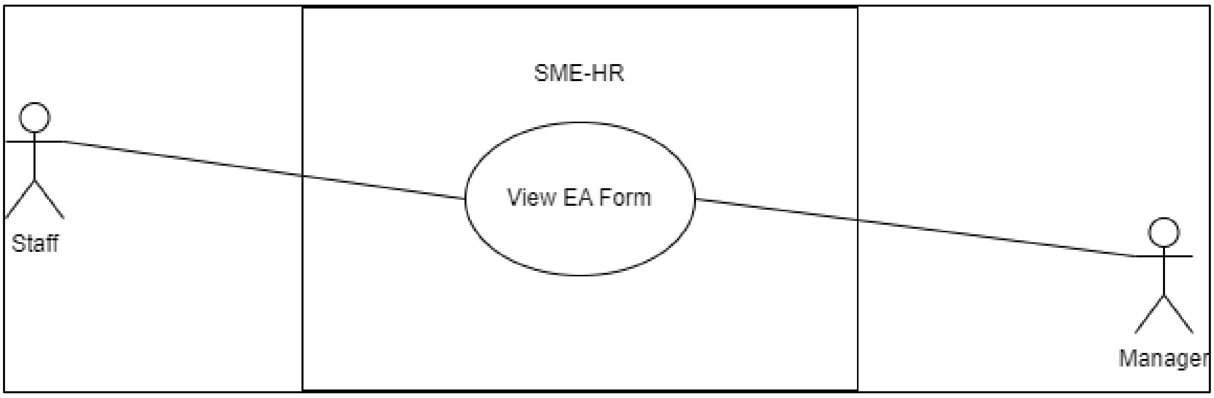


Figure 32: View EA Form Use Case Diagram

Table 13: View EA Form Use Case Description

Brief Description	This use case is initiated by the **Staff and Manager**. It provides the capability to view EA form.
Actor	Staff, Manager
Pre-Conditions	The user has already login to the system.
Basic Flow	**User (Staff, Manager)** 1. The use case begins when the user clicks on the <<EA Form>> button. 2. The system requests the EA form page from the database. 3. The system retrieves the EA form page from the database. 4. The system displays the EA form page.

	5. The user is able to do the following functions if they want to filter specific information: - **a) [A1: Search EA Form]** **b) [A2: Filter EA Form]** 6. If the user wants to view the EA form, they need to click on the specific pay slip row. 7. The system requests detailed EA form from the database. 8. The system retrieves detailed EA form information from the database. 9. The system displays the EA Form Page. 10. The use case end.
Alternative Flow	**A1: Search EA Form** 1. The user enters the keyword in the search bar. **[E1: Invalid Keyword] [R1: Valid Keyword]** 2. The user clicks on the <<Search>> button. 3. The system requests the data with a similar keyword from the database. 4. The system retrieves the data from the database. 5. The system displays the resulting search. 6. The use case continues with step 6 in basic flow. **A2: Filter EA Form** 1. The user clicks on the <<Filter>> button. 2. The user chooses to filter EA form by year. 3. The system requests the data related with the filter keyword from the database. 4. The system retrieves the data from the database. 5. The system displays the result filter. 6. The use case continues with step 6 in basic flow.
Exception Flow	**E1: Invalid Keyword** 1. The system displays an error message saying that "No Data Found!".

	2. The use case continues with step 1 in alternative flow.
Post-Conditions	Not applicable
Rules	**R1: Valid Keyword** 1. The keyword must be valid and correct to search for specific information.
Constraints	Not applicable

3.4.4 Activity Diagram

This section will be explained about the activity diagram that can describe the behaviour and to model the flow of control in SME-HR system. This activity diagram will cover all the functions that need to be developed in the SME-HR system.

3.4.4.1 Login

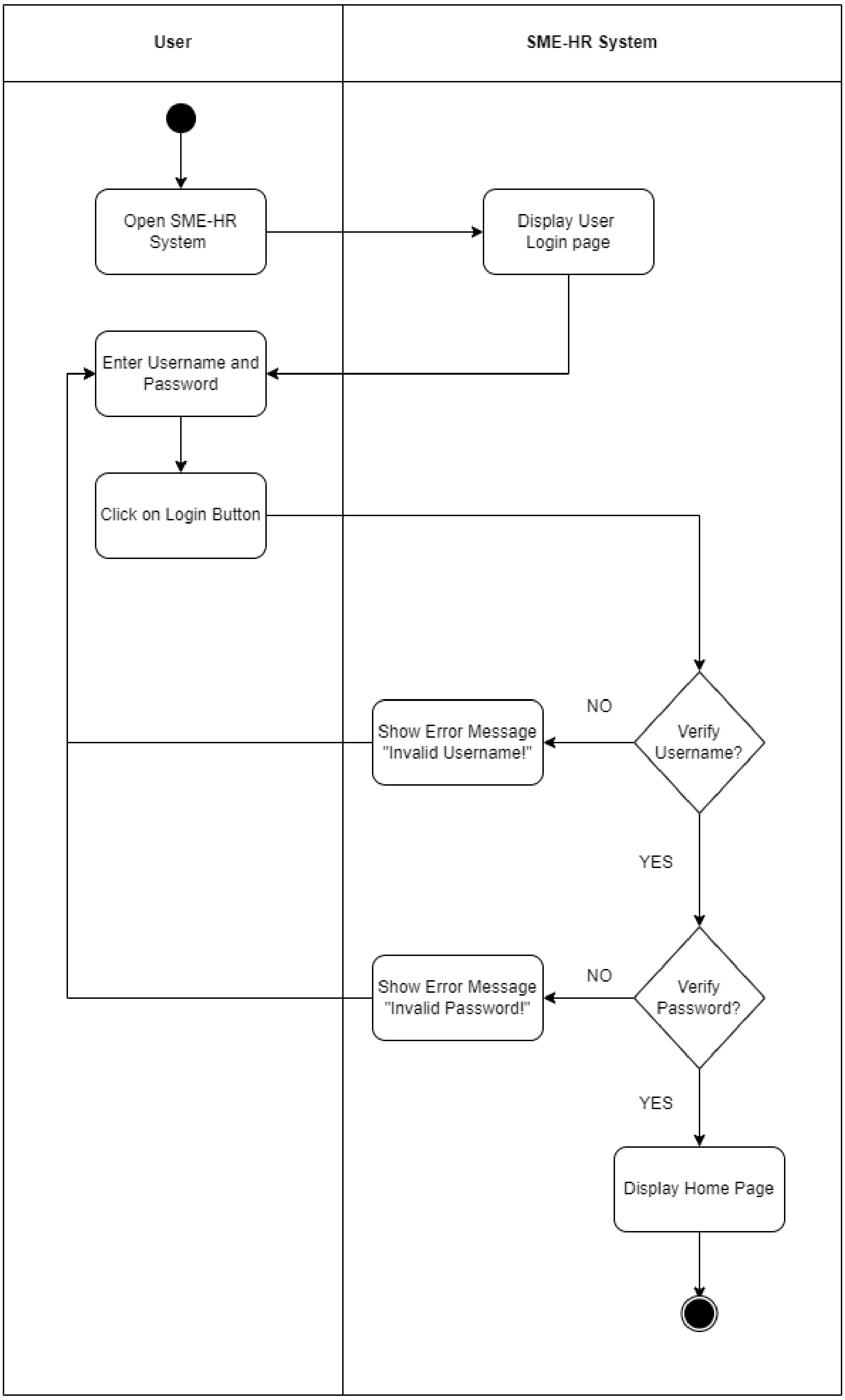


Figure 33: Login Activity Diagram

3.4.4.2 Manage Employee

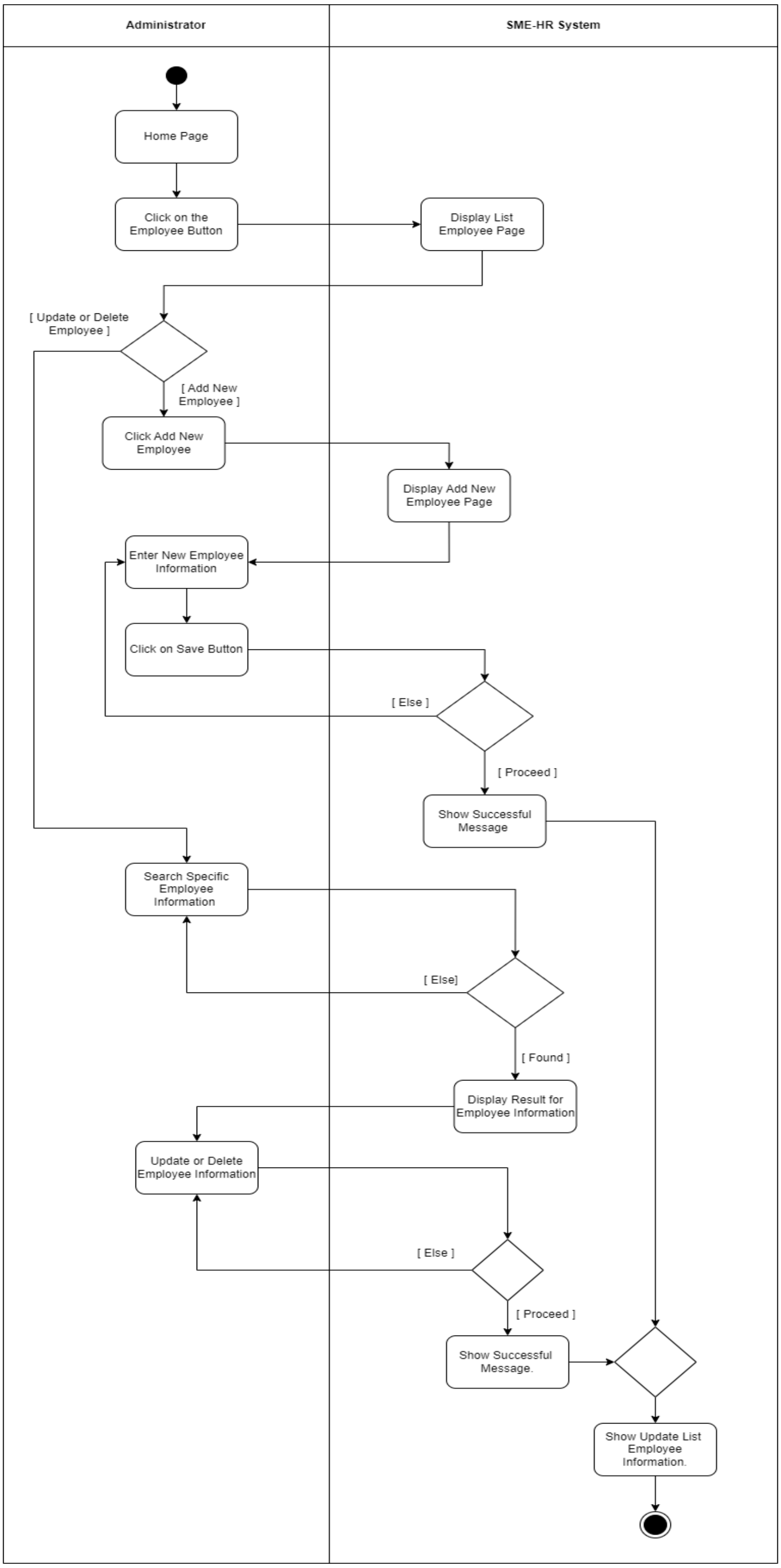

Administrator
SME-HR System
Home Page
Click on the Employee Button
Display List Employee Page
[Update or Delete Employee]
[Add New Employee]
Click Add New Employee
Display Add New Employee Page
Enter New Employee Information
Click on Save Button
[Else]
[Proceed]
Show Successful Message
Search Specific Employee Information
[Else]
[Found]
Display Result for Employee Information
Update or Delete Employee Information
[Else]
[Proceed]
Show Successful Message.
Show Update List Employee Information.

Figure 34: Manage Employee Activity Diagram

3.4.4.3 Manage Profile

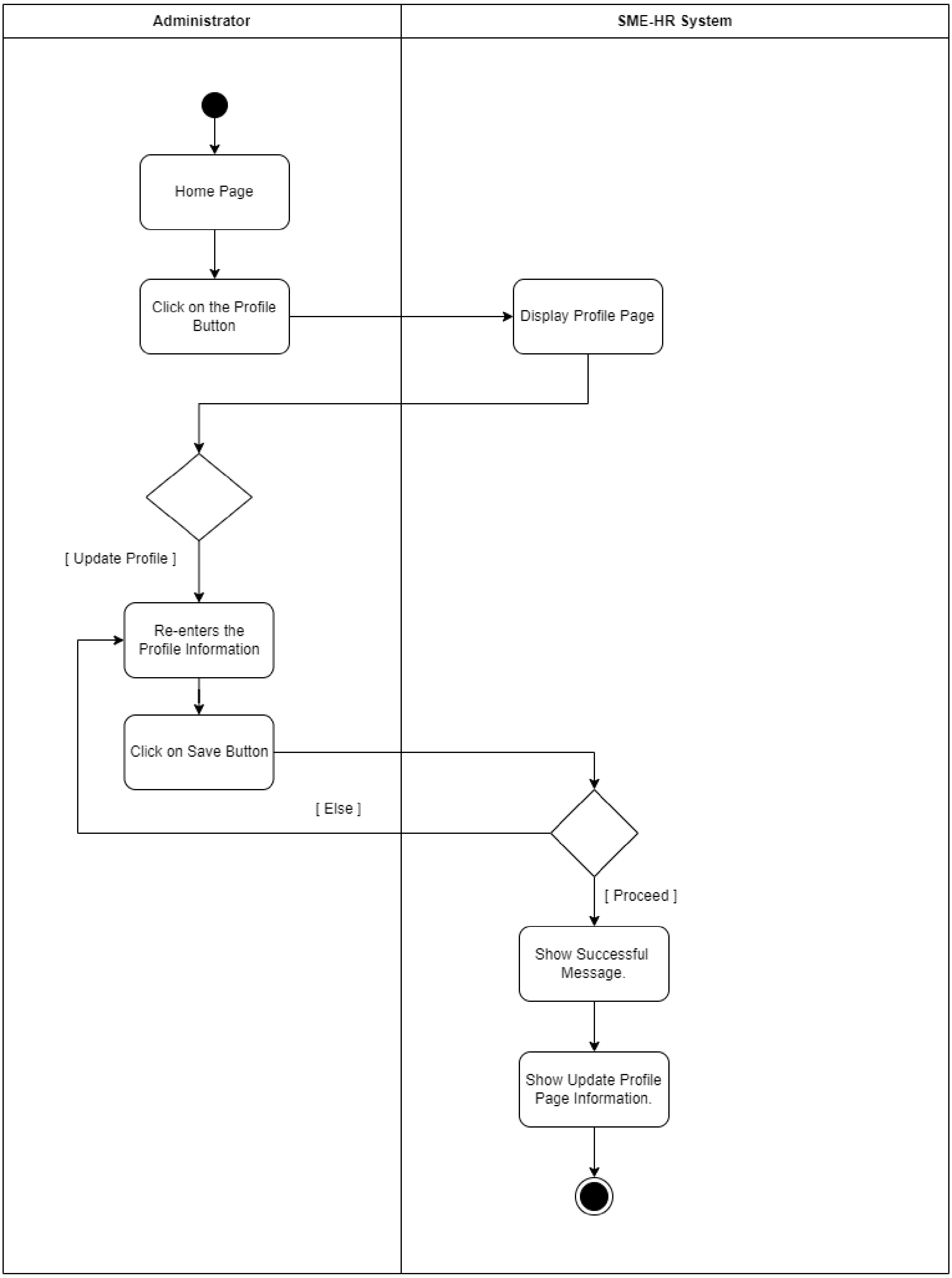


Figure 35: Manage Profile Activity Diagram

3.4.4.4 Manage Payroll

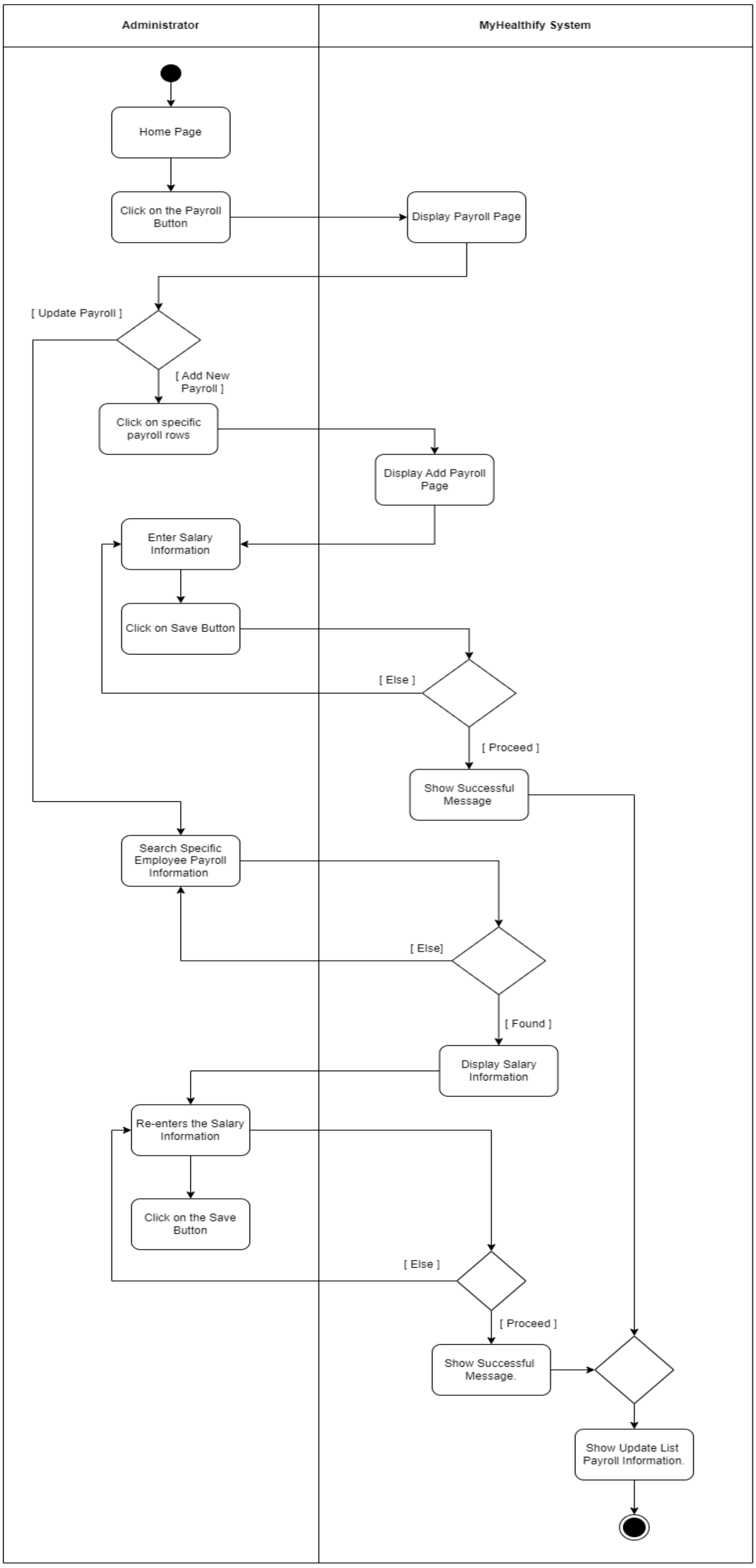


Figure 36: Manage Payroll Activity Diagram

3.4.4.5 Manage Leave

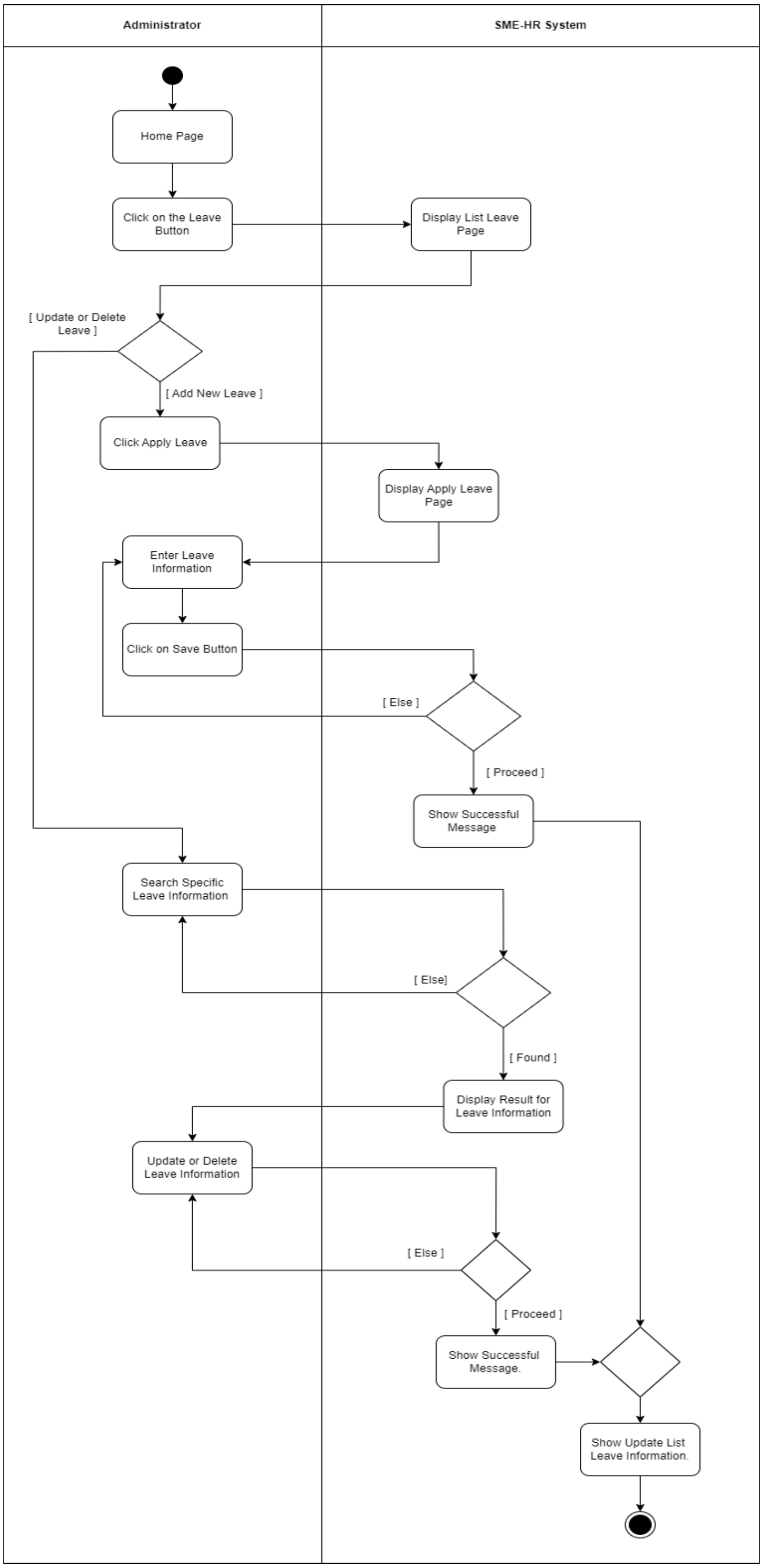


Figure 37: Manage Leave Activity Diagram

3.4.4.6 Manage Claim

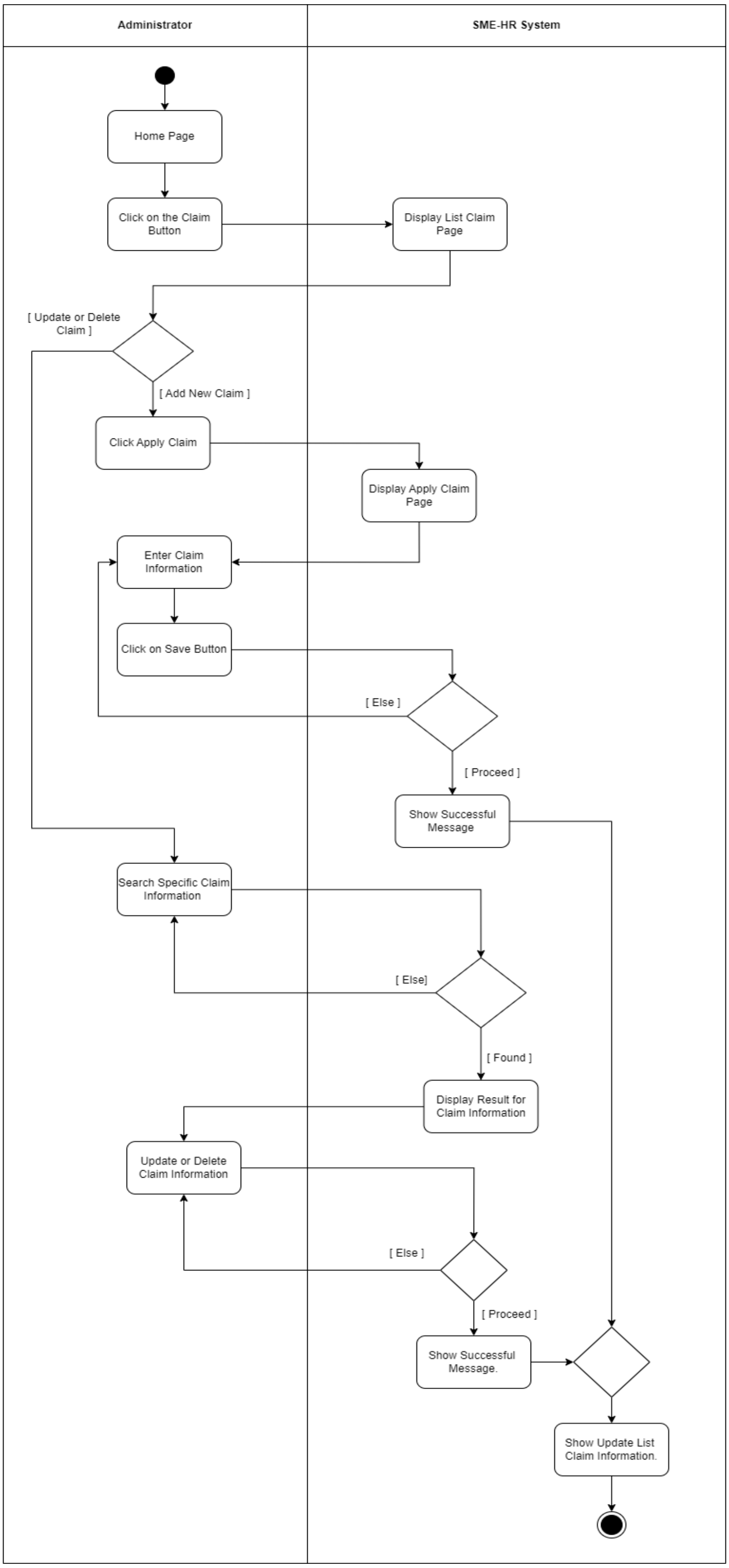


Figure 38: Manage Claim Activity Diagram

3.4.4.7 Manage EA Form

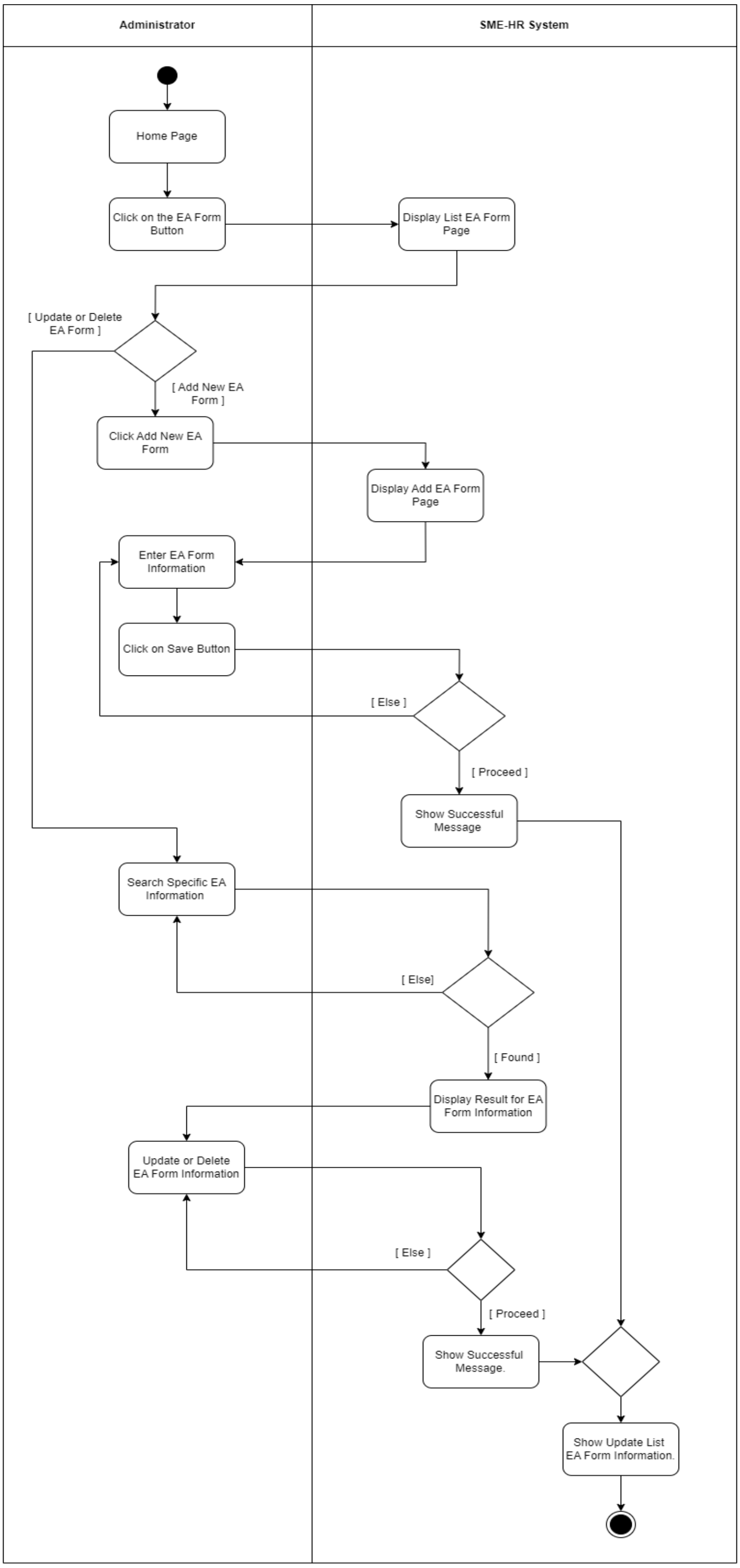

Administrator
SME-HR System
Home Page
Click on the EA Form Button
Display List EA Form Page
[Update or Delete EA Form]
[Add New EA Form]
Click Add New EA Form
Display Add EA Form Page
Enter EA Form Information
Click on Save Button
[Else]
[Proceed]
Show Successful Message
Search Specific EA Information
[Else]
[Found]
Display Result for EA Form Information
Update or Delete EA Form Information
[Else]
[Proceed]
Show Successful Message.
Show Update List EA Form Information.

Figure 39: Manage EA Form Activity Diagram

3.5 Data Design

This chapter explains the details of the data that related to the SME-HR system. The data will cover up the Entity Relationship Diagram (ERD) and the Database Dictionary. These data were chosen based on their criteria for each module.

3.5.1 Entity Relationship Diagram

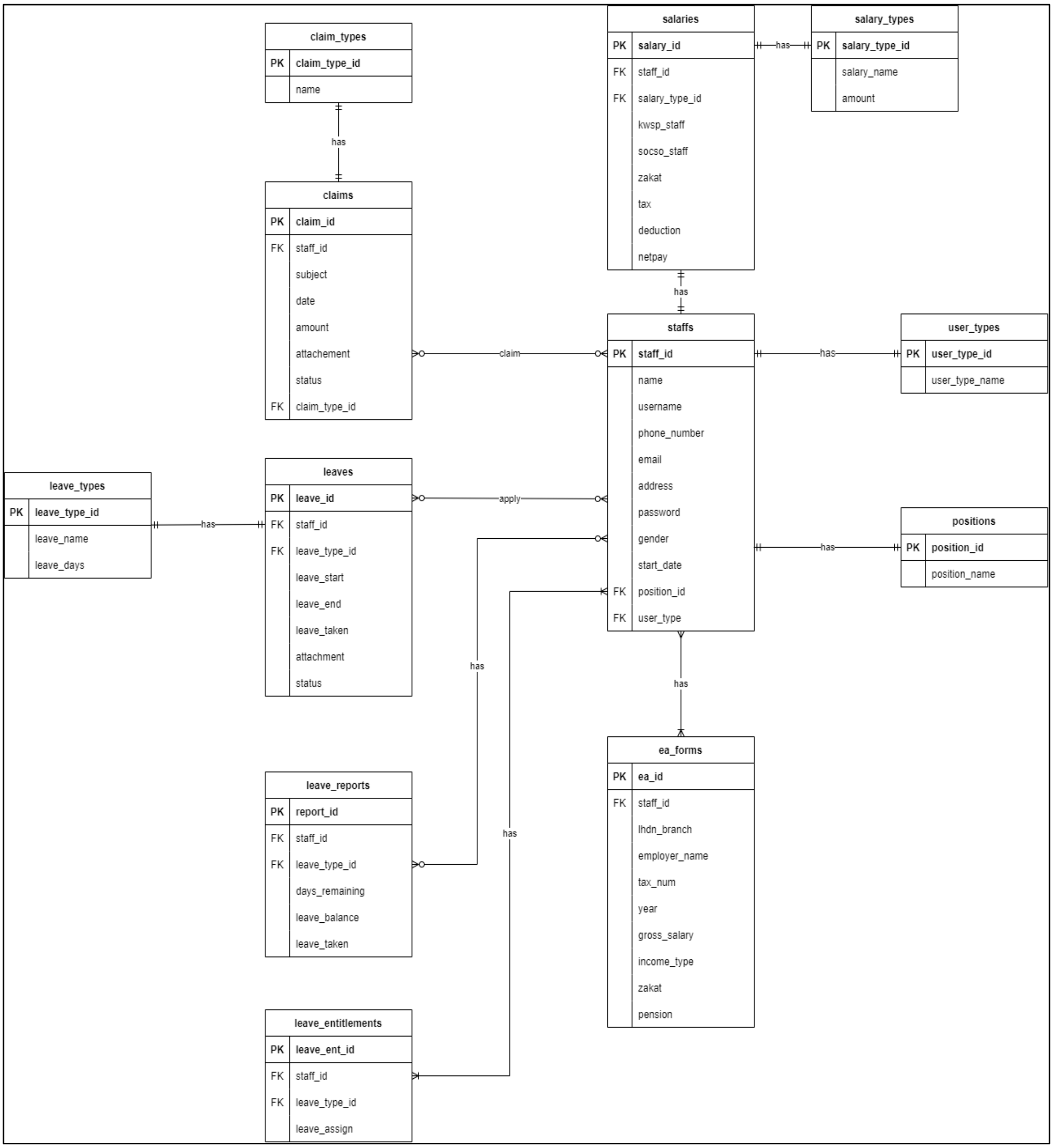


Figure 40: SME-HR System ERD

3.5.2 Database Dictionary

3.5.2.1 Staffs

Table 14: Staff Data Dictionary

Field Name	Description	Data Type	Constraint
staff_id	ID for staff	INT	PK
name	Name of the staff	VARCHAR (50)	
username	Username of the staff	VARCHAR (20)	
phone_number	Phone number of the staff	VARCHAR (10)	
email	Email of the staff	VARCHAR (20)	
address	Address of the staff	VARCHAR (50)	
password	Password for staff	VARCHAR (20)	
gender	Gender of the staff	VARCHAR (10)	
start_date	Date when the staff start to work	DATE	
position_ID	Position of the staff	INT	FK
user_type	User type of the staff	INT	FK

3.5.2.2 User Types

Table 15: User Types Data Dictionary

Field Name	Description	Data Type	Constraint
user_type_id	ID for the user type	INT	PK
user_type_name	Name for the user type	VARCHAR (50)	

3.5.2.3 Positions

Table 16: Positions Data Dictionary

Field Name	Description	Data Type	Constraint
position_id	ID for the position	INT	PK
position_name	Name for the position	VARCHAR (50)	

3.5.2.4 Salaries

Table 17: Salaries Data Dictionary

Field Name	Description	Data Type	Constraint
salary_id	ID for the salary	INT	PK
staff_id	ID for the staff	INT	FK

salary_type_id	ID for the salary type	INT	FK
kwsp_staff	KWSP of the staff	DECIMAL (10,2)	
socso_staff	Socso of the staff	DECIMAL (10,2)	
zakat	Zakat of the staff	DECIMAL (10,2)	
deduction	Deduction of the staff	DECIMAL (10,2)	
netpay	Netpay of the staff	DECIMAL (10,2)	

3.5.2.5 Salary Types

Table 18: Salary Types Data Dictionary

Field Name	Description	Data Type	Constraint
salary_type_id	ID for the salary type	INT	PK
salary_name	Name for the salary type	VARCHAR (50)	
amount	Amount of the salary	DECIMAL (10,2)	

3.5.2.6 EA Form

Table 19: EA Form Data Dictionary

Field Name	Description	Data Type	Constraint
ea_id	ID for the position	INT	PK
staff_id	ID for the position	INT	FK
lhdn_branch	LHDN branch for the company	VARCHAR (50)	
employer_name	Employer name	VARCHAR (50)	
tax_num	Tax number of the staff	VARCAHR (50)	
year	Year for the EA Form	INT	
gross_salary	Gross salary of the staff	DECIMAL (10,2)	
income_type	Income type	VARCHAR (20)	
zakat	Zakat of the staff	DECIMAL (10,2)	
pension	Pension of the staff	DECIMAL (10,2)	

3.5.2.7 Claims

Table 20: Claims Data Dictionary

Field Name	Description	Data Type	Constraint
claim_id	ID for the claim	INT	PK
staff_id	ID for the position	INT	FK
subject	Subject of the claim	VARCHAR (50)	
date	Date for the claim	DATE	
amount	Amount of the claim	DECIMAL (10,2)	
attachment	Attachment or receipt for the claim	VARCHAR (20)	
status	Status for the claim	INT	
claim_type_id	ID for the claim type	INT	FK

3.5.2.8 Claim Types

Table 21: Claim Types Data Dictionary

Field Name	Description	Data Type	Constraint
claim_type_id	ID for the claim type	INT	PK
name	Name for the claim type	VARCHAR (30)	

3.5.2.9 Leaves

Table 22: Leaves Data Dictionary

Field Name	Description	Data Type	Constraint
leave_id	ID for the leave	INT	PK
staff_id	ID for the position	INT	FK
leave_type_id	ID for the leave type	INT	FK
leave_start	Date for the leave start	DATE	
leave_end	Date for the leave end	DATE	
leave_taken	Leave taken day	INT	
attachment	Attachment for the leave	VARCHAR (30)	
status	Status for the leave	INT	

3.5.2.10 Leave Reports

Table 23: Leave Reports Data Dictionary

Field Name	Description	Data Type	Constraint
report_id	ID for the leave report	INT	PK

staff_id	ID for the staff	INT	FK
leave_type_id	ID for leave type	VARCHAR (50)	
days_remaining	Days remaining for the leave	VARCHAR (50)	
leave_balance	Total leave balance	VARCAHR (50)	
leave_taken	Total leave taken	INT	

3.5.2.11 Leave Entitlements

Table 24: Leave Entitlements Data Dictionary

Field Name	Description	Data Type	Constraint
leave_ent_id	ID for the leave entitlement	INT	PK
staff_id	ID for the staff	INT	FK
leave_type_id	ID for the leave type	INT	FK
leave_assign	Total of leave assign	INT	

3.5.2.12 Leave Types

Table 25: Leave Types Data Dictionary

Field Name	Description	Data Type	Constraint
leave_type_id	ID for the leave type	INT	PK
leave_name	Name of the leave	VARCHAR (30)	
leave_days	Total for the leave	INT	

3.6 Proof of Initial Concept

This section will explain the proof and demonstration evidence of the early project for the SME-HR system. The proof of the initial concept will be explained about the design prototype of the SME-HR system. This design prototype consists of all the modules in the system.

3.6.1 Login

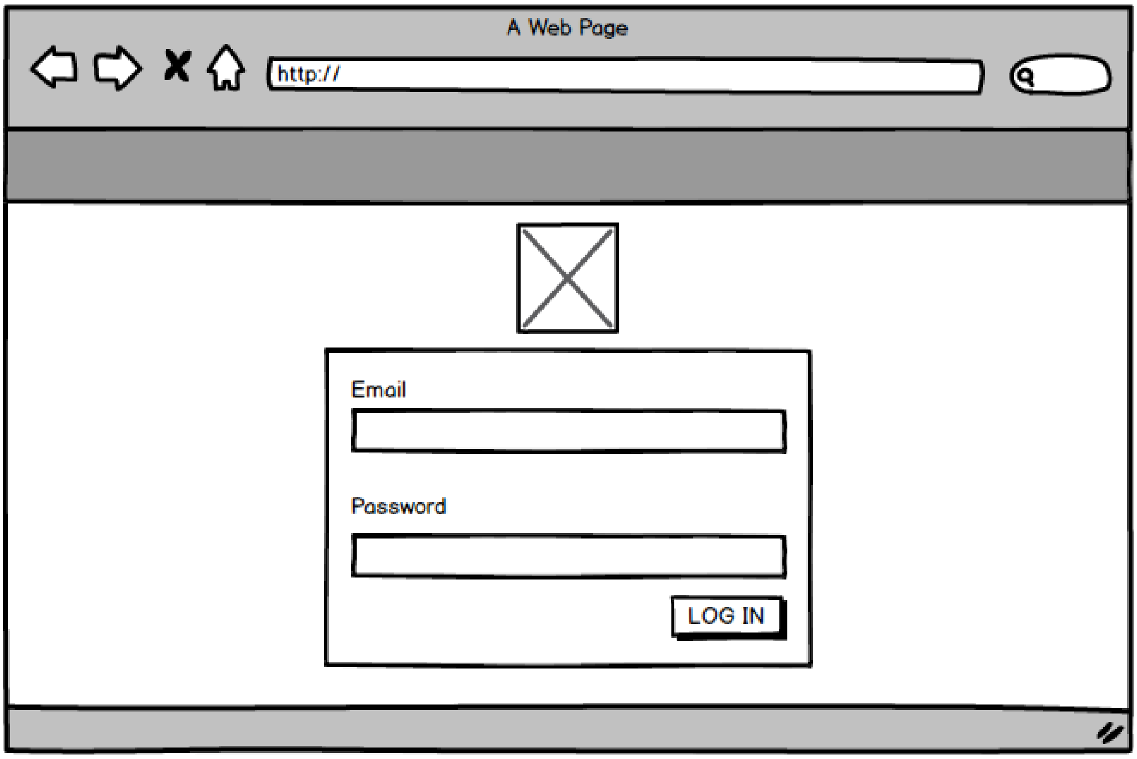
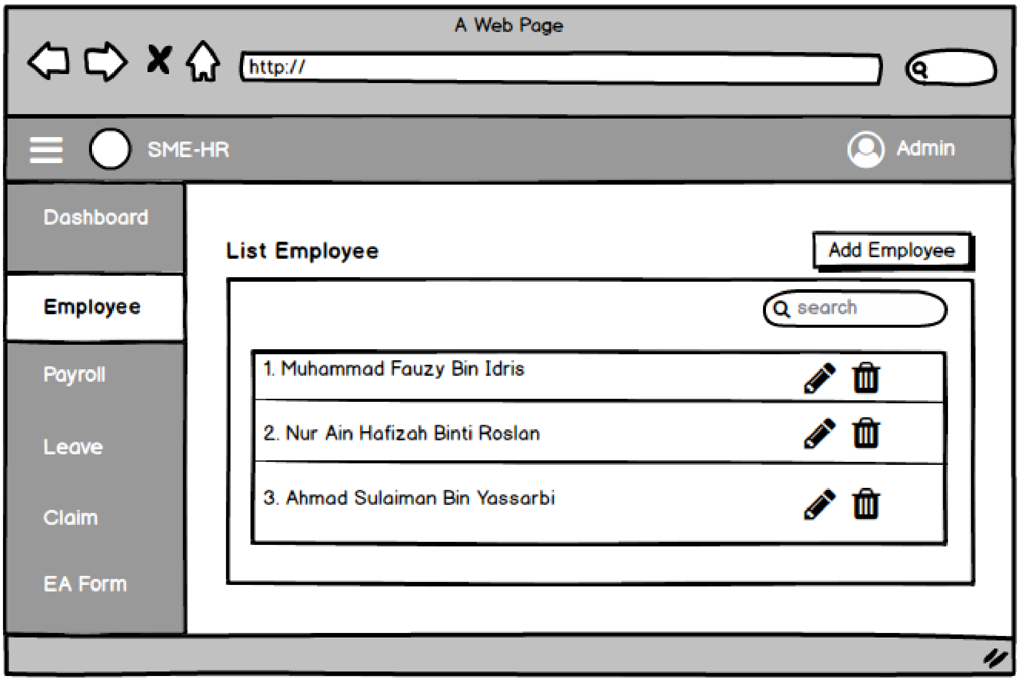


Figure 41: User Login Interface

3.6.2 Manage Employee

3.6.2.1 Employee List Page



1. Muhammad Fauzy Bin Idris
2. Nur Ain Hafizah Binti Roslan
3. Ahmad Sulaiman Bin Yassarbi

Figure 42: Employee List Page Interface

3.6.2.2 Add Employee Page

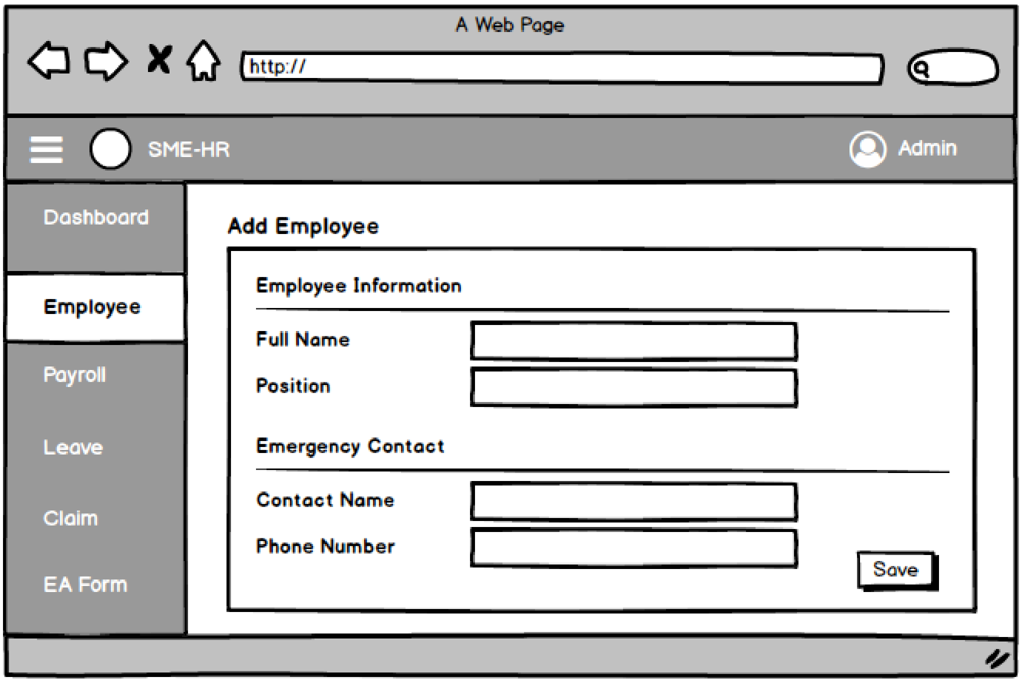


Figure 43: Add Employee Page

3.6.2.3 Add Employee Confirmation Page

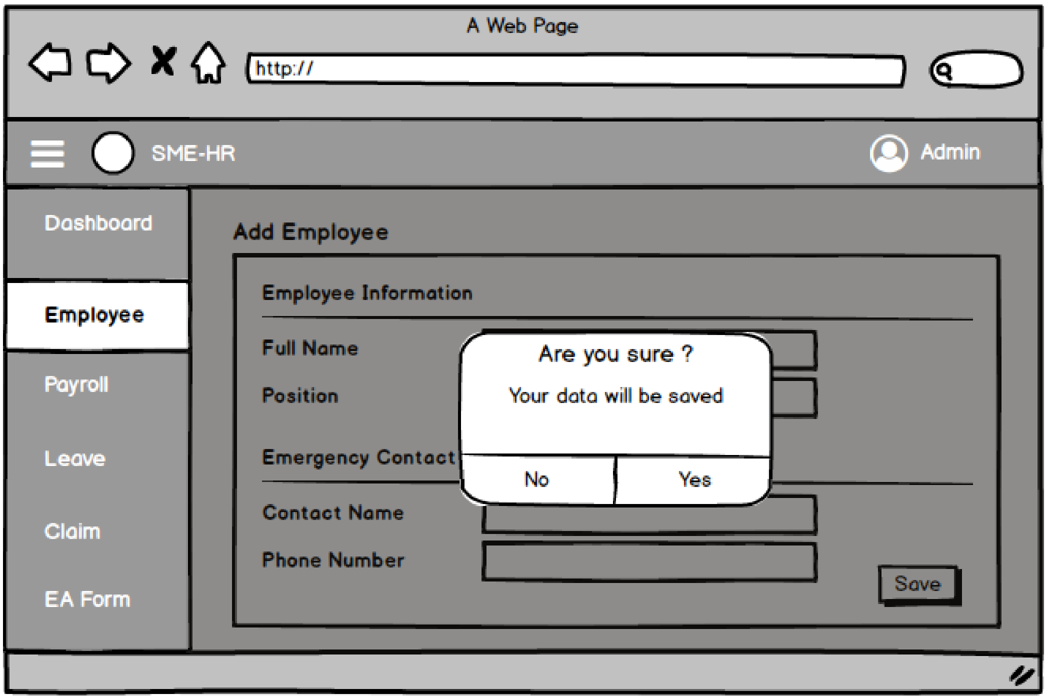


Figure 44: Add Employee Confirmation Page Interface

3.6.2.4 Add Employee Successful Page

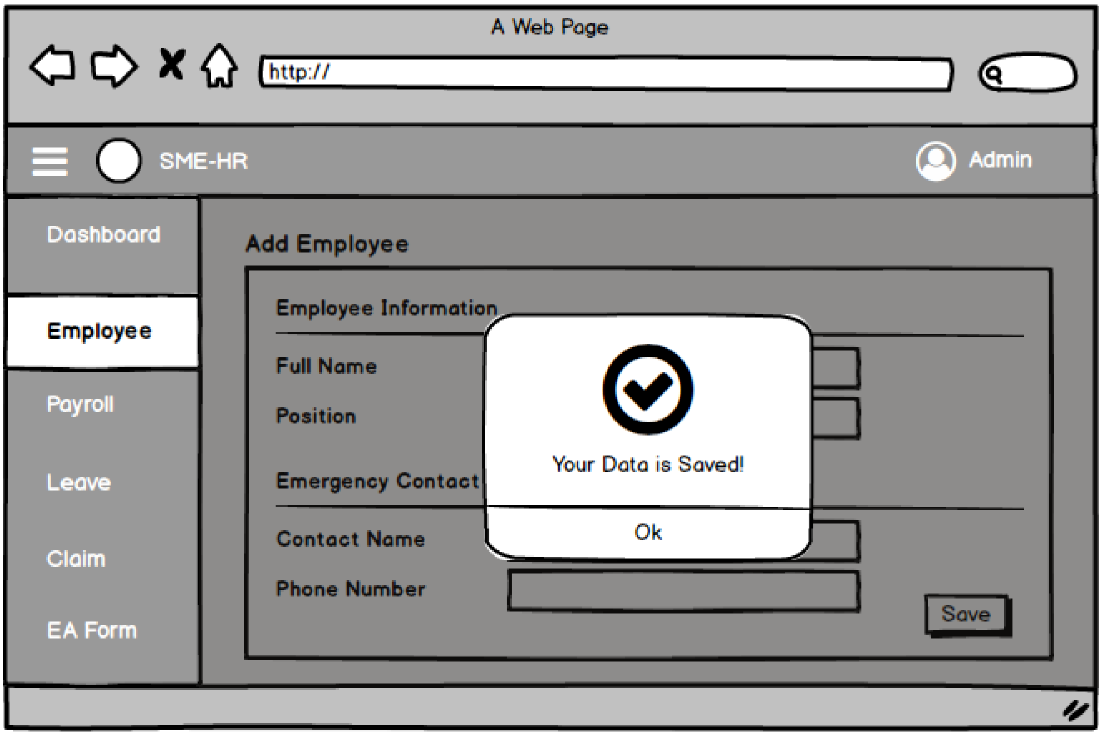
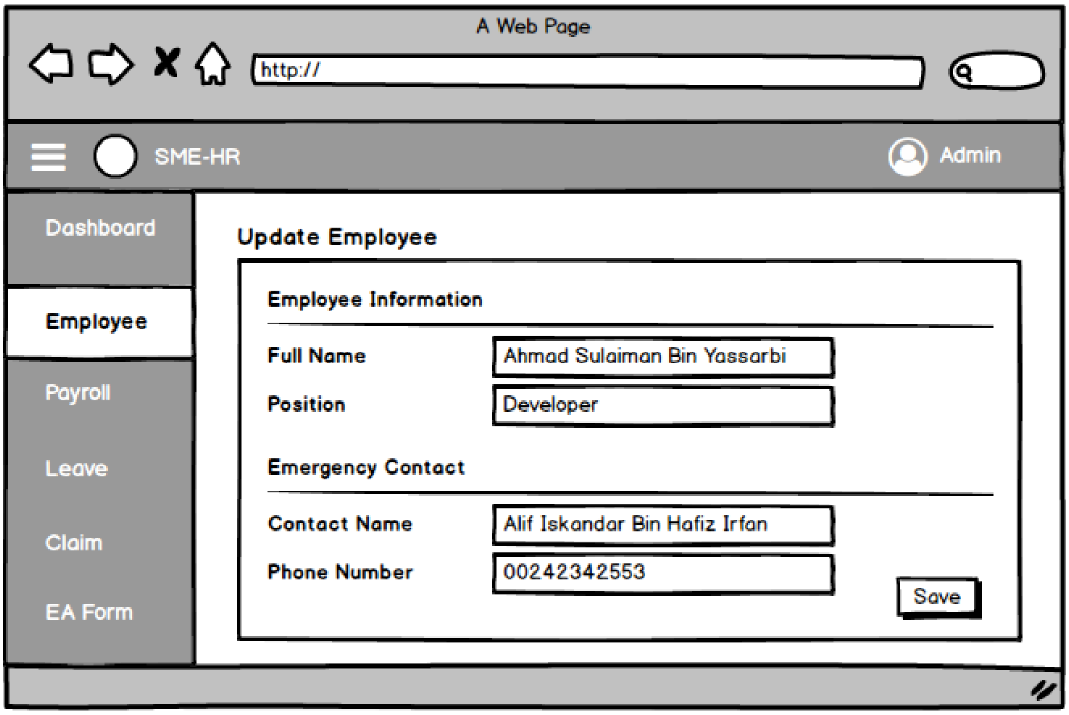


Figure 45: Add Employee Successful Page Interface

3.6.2.5 Update Employee Page



Update Employee

Employee Information

Full Name: Ahmad Sulaiman Bin Yassarbi

Position: Developer

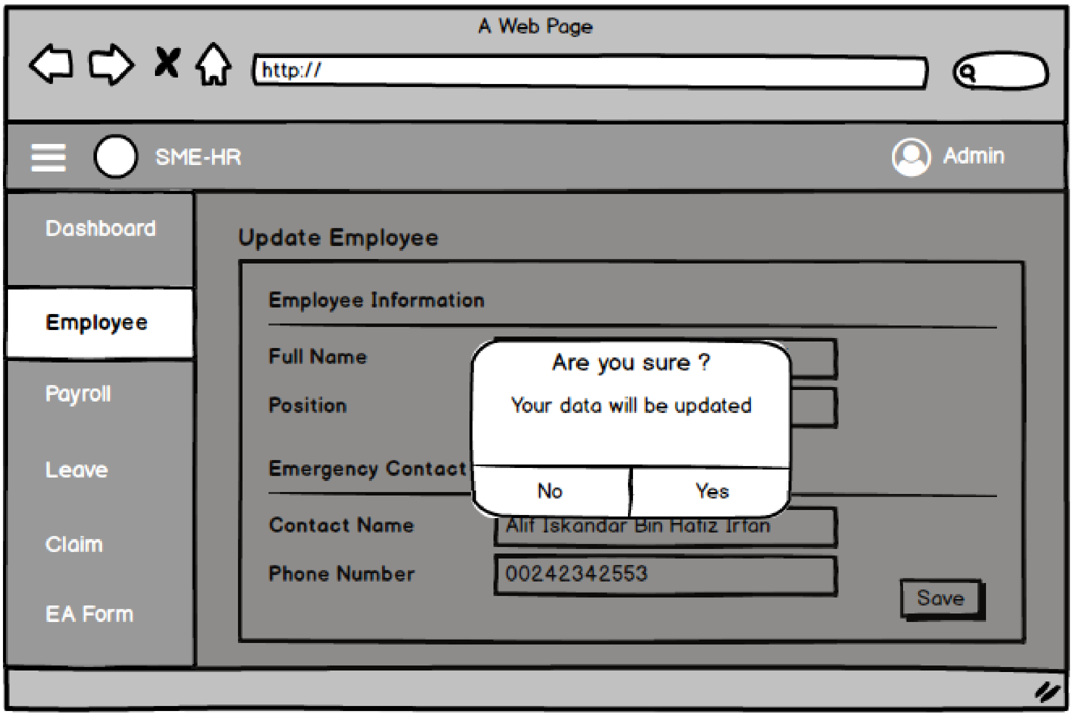
Emergency Contact

Contact Name: Alif Iskandar Bin Hafiz Irfan

Phone Number: 00242342553

Figure 46: Update Employee Page Interface

3.6.2.6 Update Employee Confirmation Page



Update Employee

Employee Information

Full Name

Position

Are you sure ?

Your data will be updated

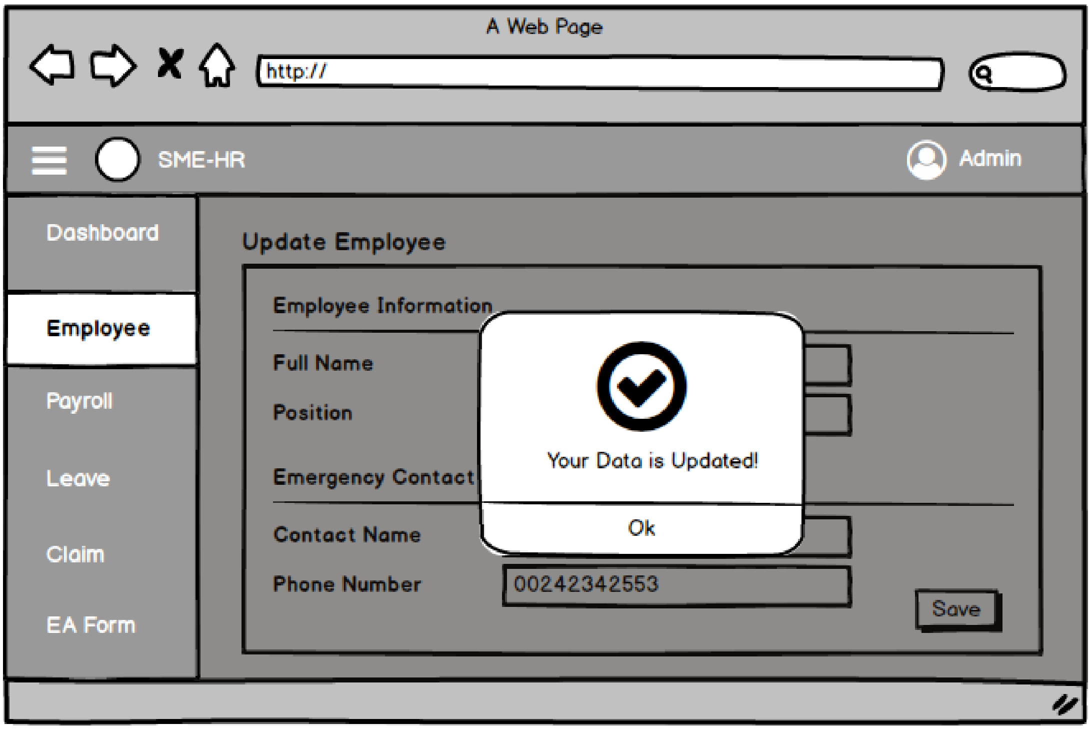
Emergency Contact

Contact Name: Alif Iskandar Bin Hafiz Irfan

Phone Number: 00242342553

Figure 47: Update Employee Confirmation Page Interface

3.6.2.7 Update Employee Successful Page

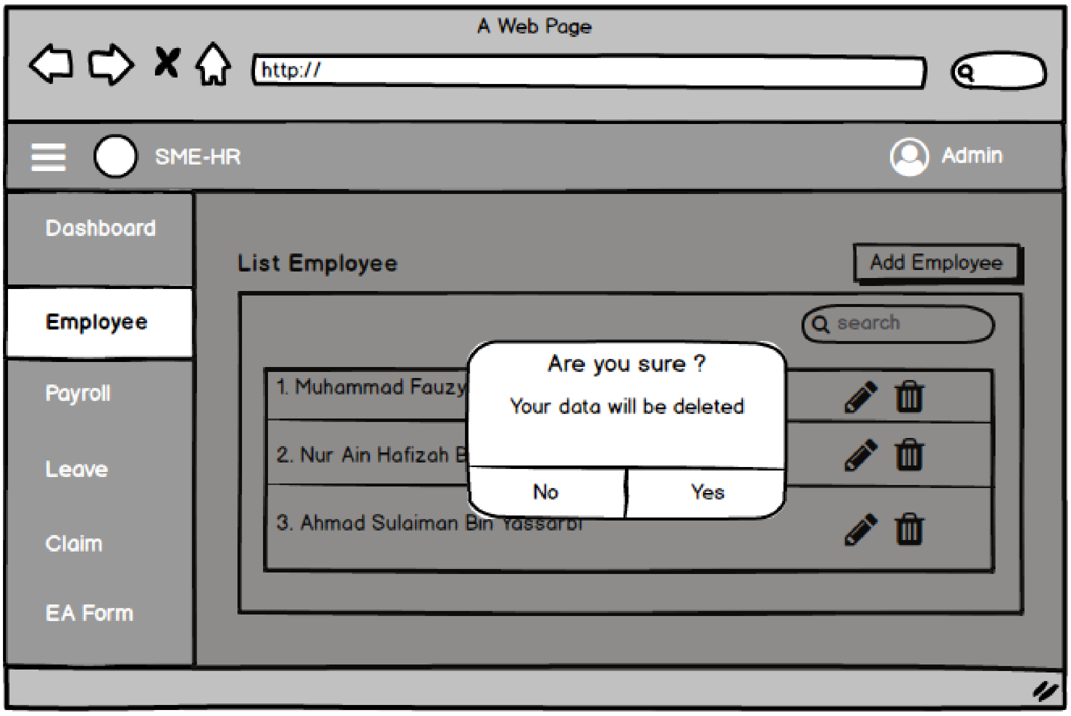


Phone Number: 00242342553

Your Data is Updated!

Figure 48: Update Employee Successful Page Interface

3.6.2.8 Delete Employee Confirmation Page



1. Muhammad Fauzy

2. Nur Ain Hafizah B

3. Ahmad Sulaiman Bin Yassarbi

Are you sure ?

Your data will be deleted

Figure 49: Delete Employee Confirmation Page Interface

3.6.2.9 Delete Employee Successful Page

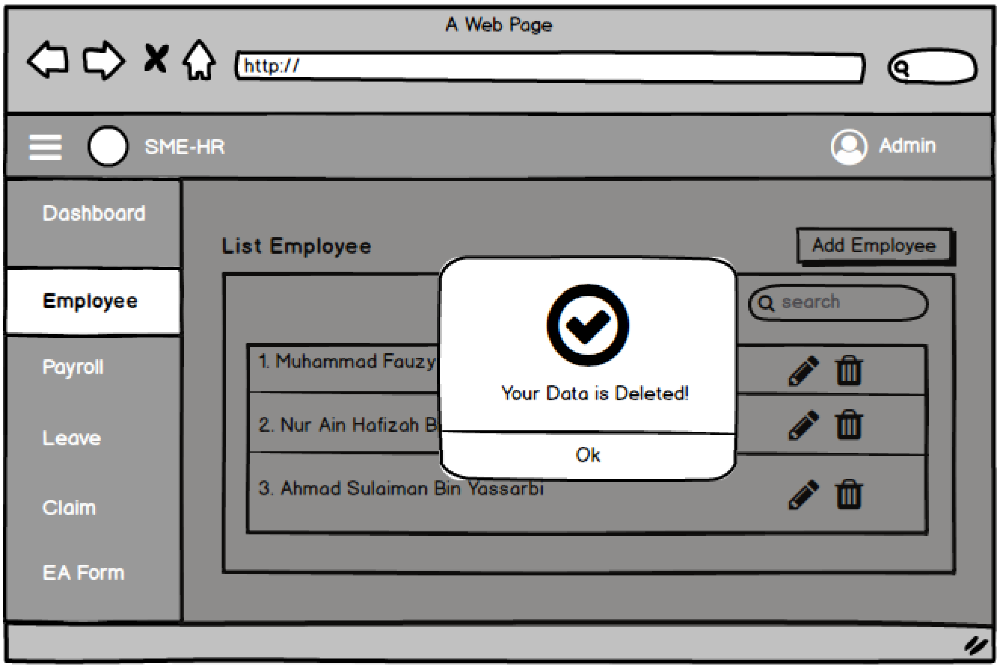


Figure 50: Delete Employee Successful Page Interface

3.6.3 Manage Leave

3.6.3.1 Leave List Page

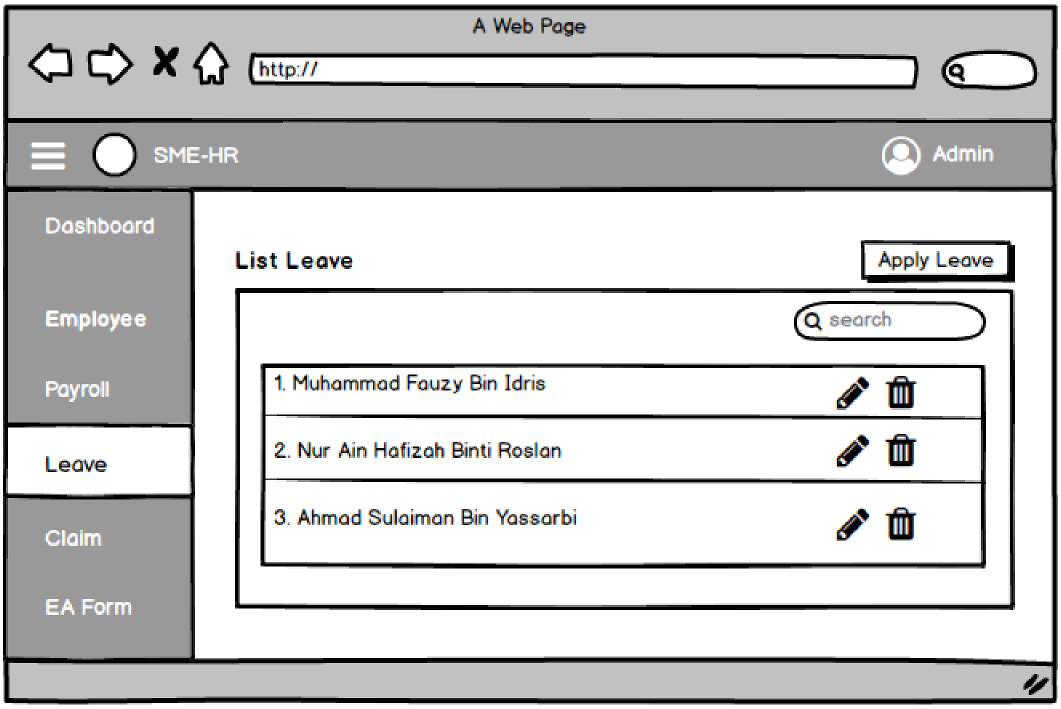


Figure 51: Leave List Page Interface

3.6.3.2 Add Leave Page

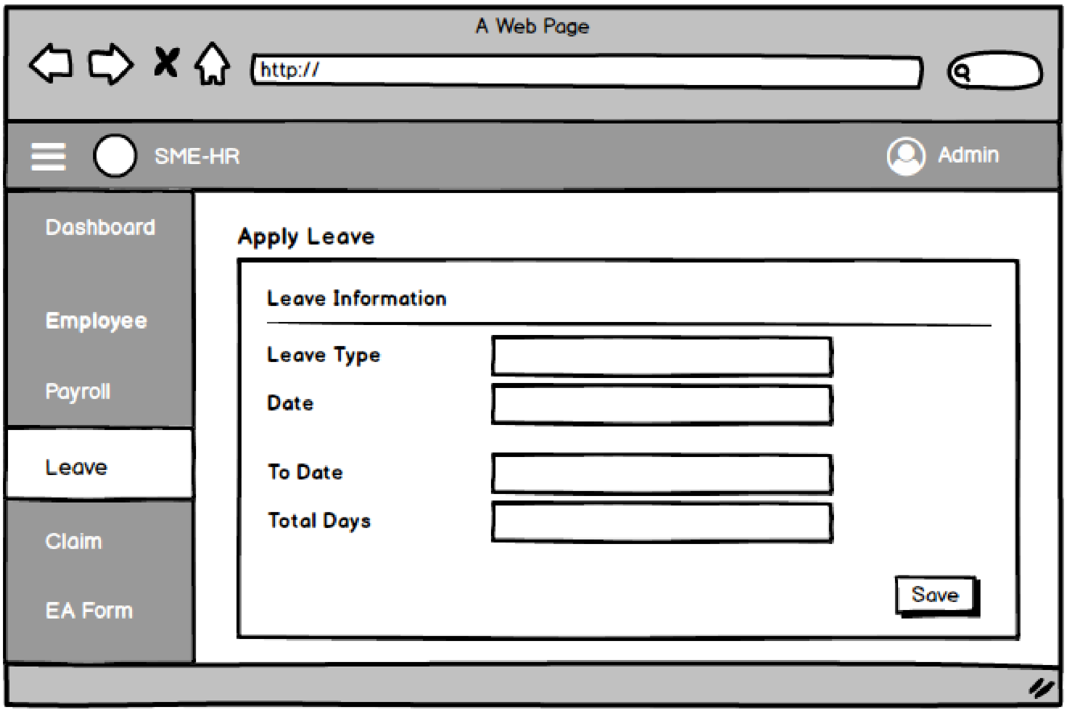
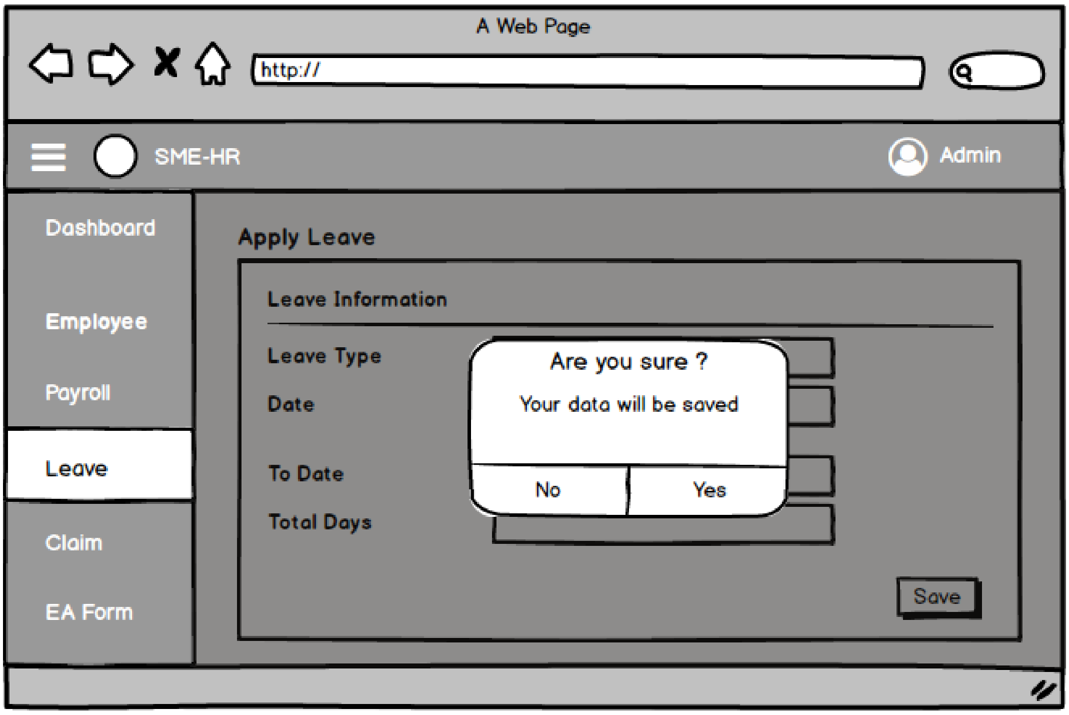


Figure 52: Add Leave Page Interface

3.6.3.3 Add Leave Confirmation Page

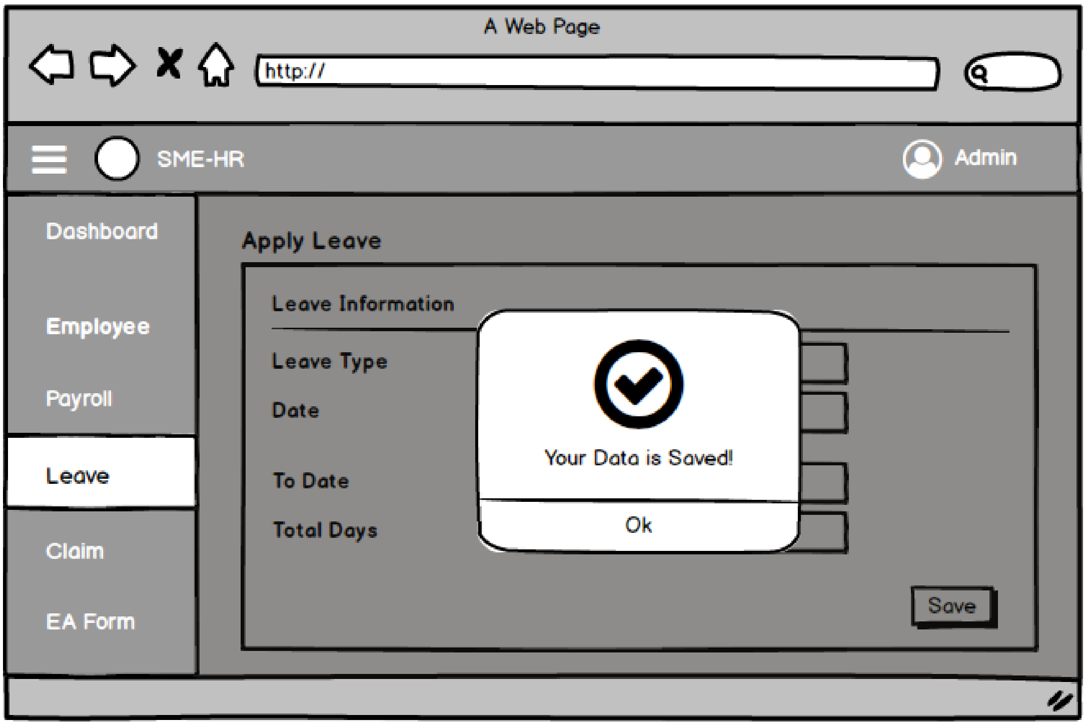


Are you sure ?

Your data will be saved

Figure 53: Add Leave Confirmation Page Interface

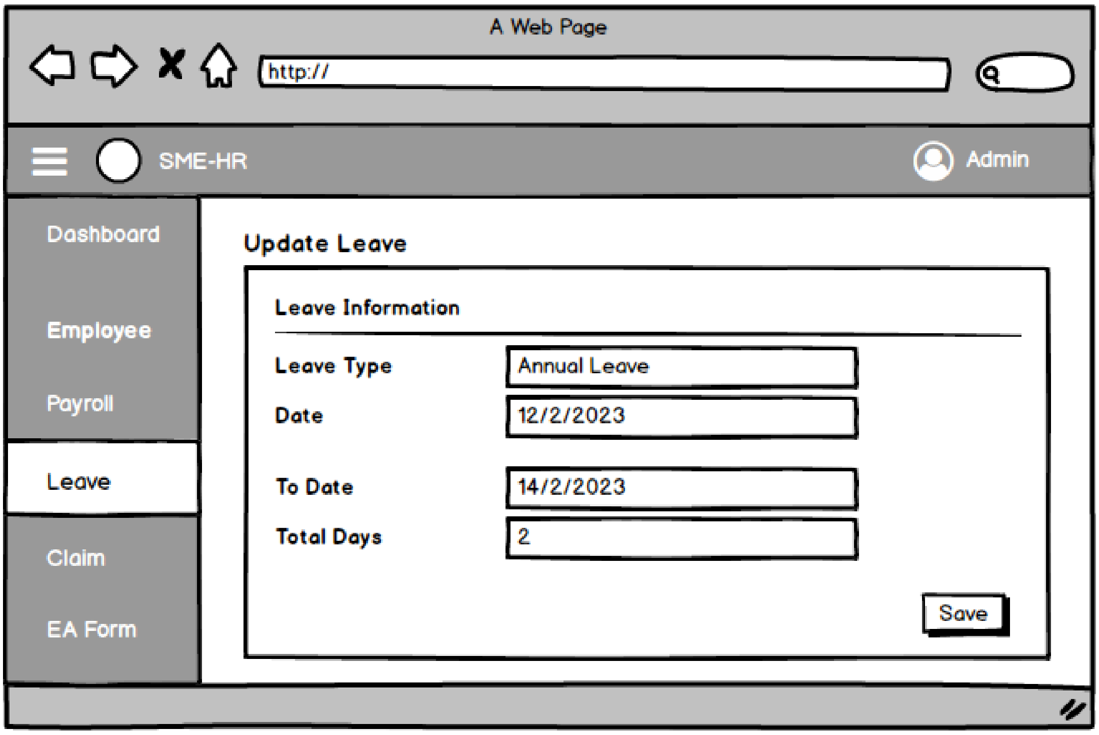
3.6.3.4 Add Leave Successful Page



Your Data is Saved!

Figure 54: Add Leave Successful Page Interface

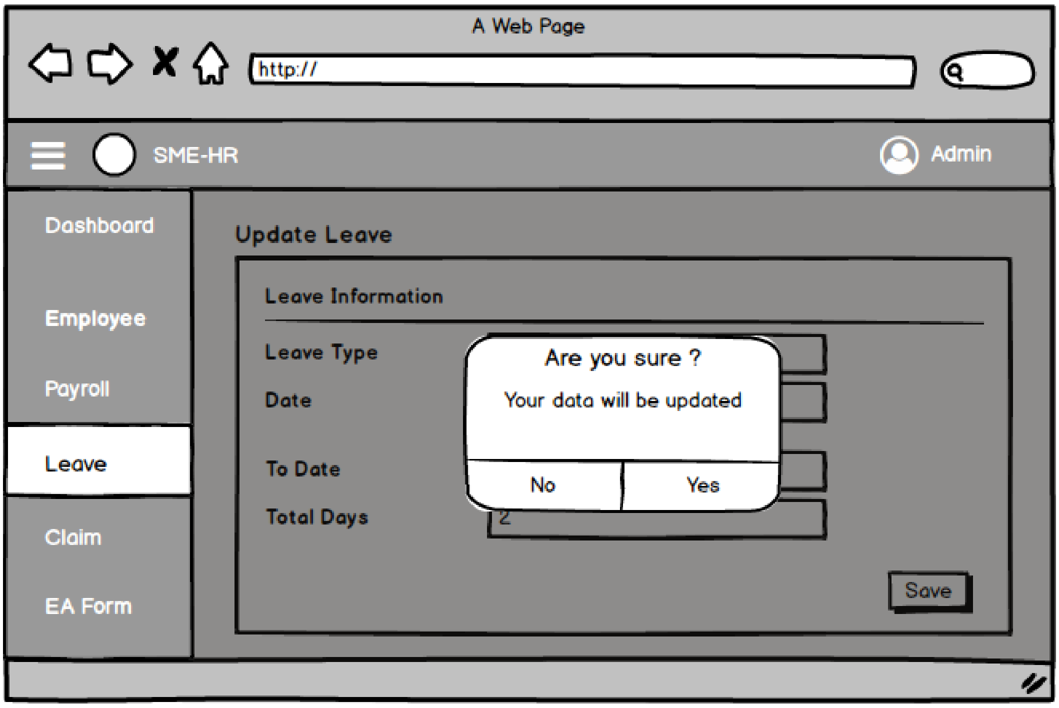
3.6.3.5 Update Leave Page



Leave Type	Annual Leave
Date	12/2/2023
To Date	14/2/2023
Total Days	2

Figure 55: Update Leave Page Interface

3.6.3.6 Update Leave Confirmation Page



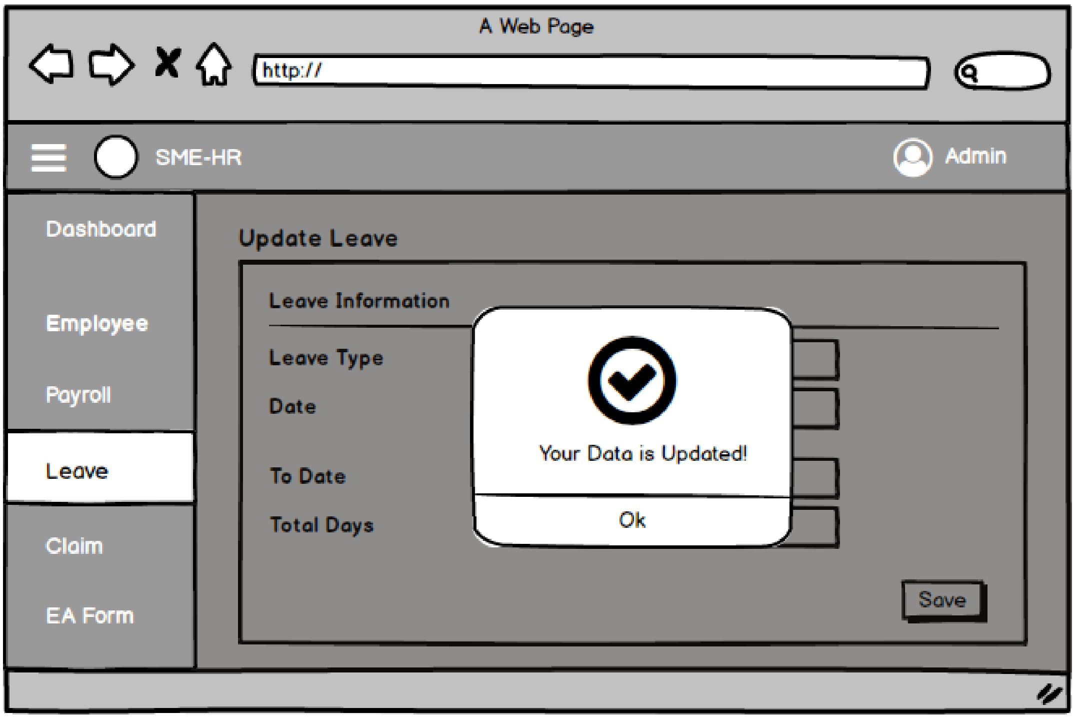
Are you sure ?

Your data will be updated

Total Days: 2

Figure 56: Update Leave Confirmation Page Interface

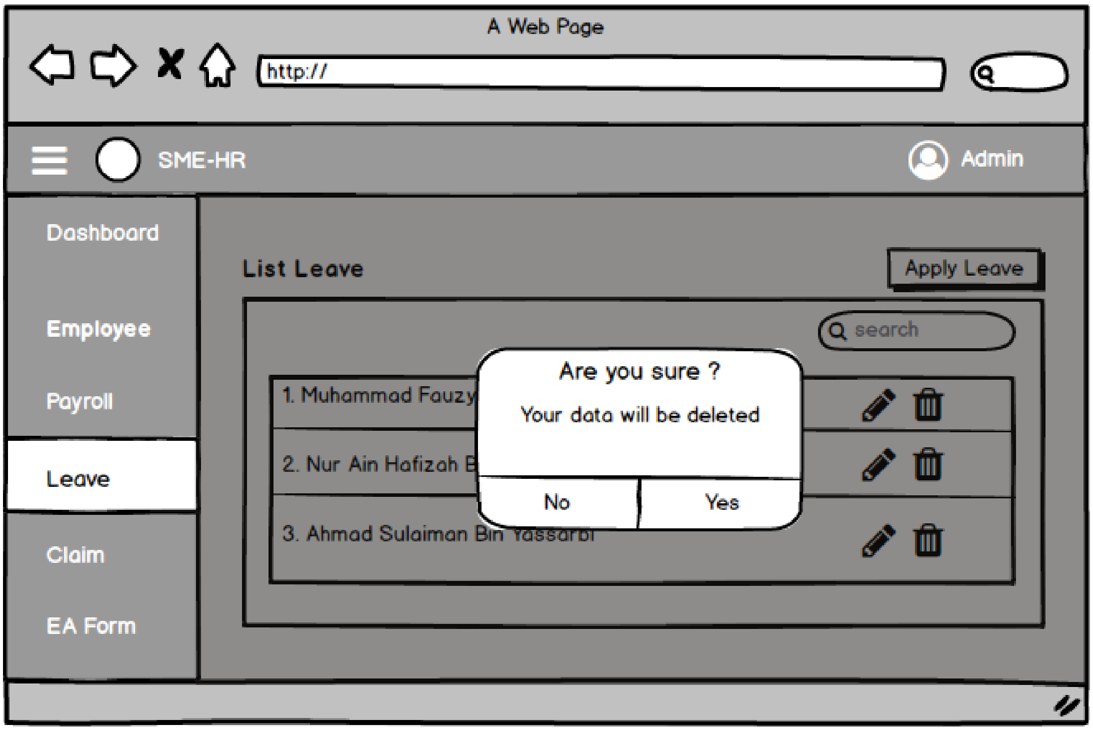
3.6.3.7 Update Leave Successful Page



Your Data is Updated!

Figure 57: Update Leave Successful Page Interface

3.6.3.8 Delete Leave Confirmation Page

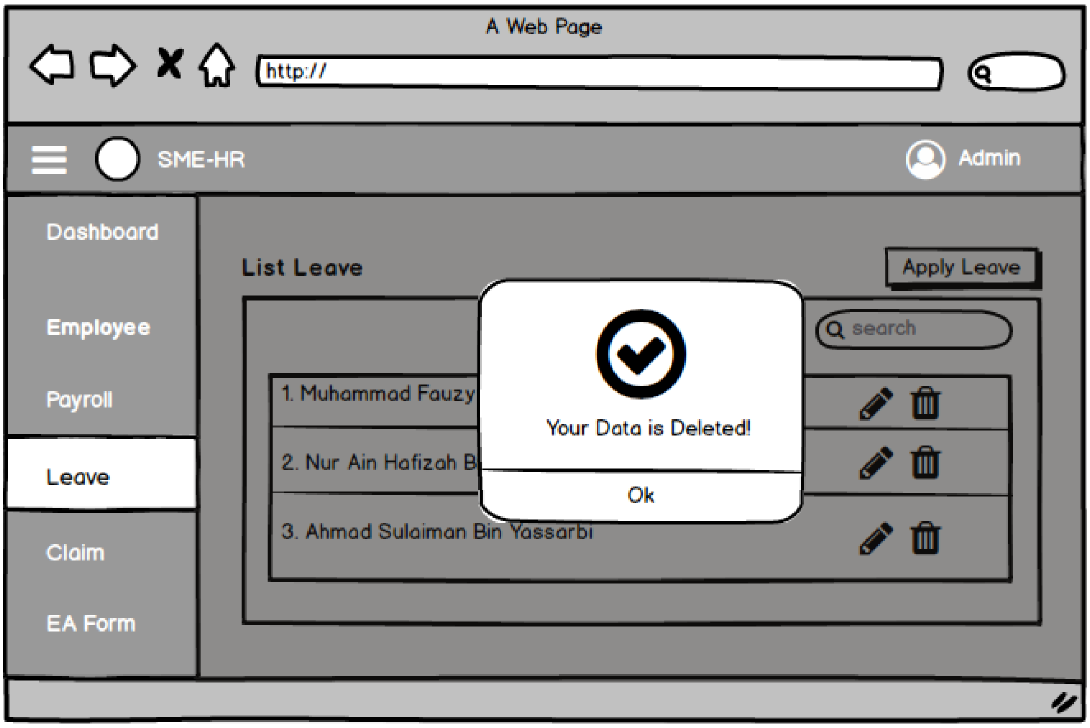


Are you sure ?

Your data will be deleted

Figure 58: Delete Leave Confirmation Page Interface

3.6.3.9 Delete Leave Successful Page

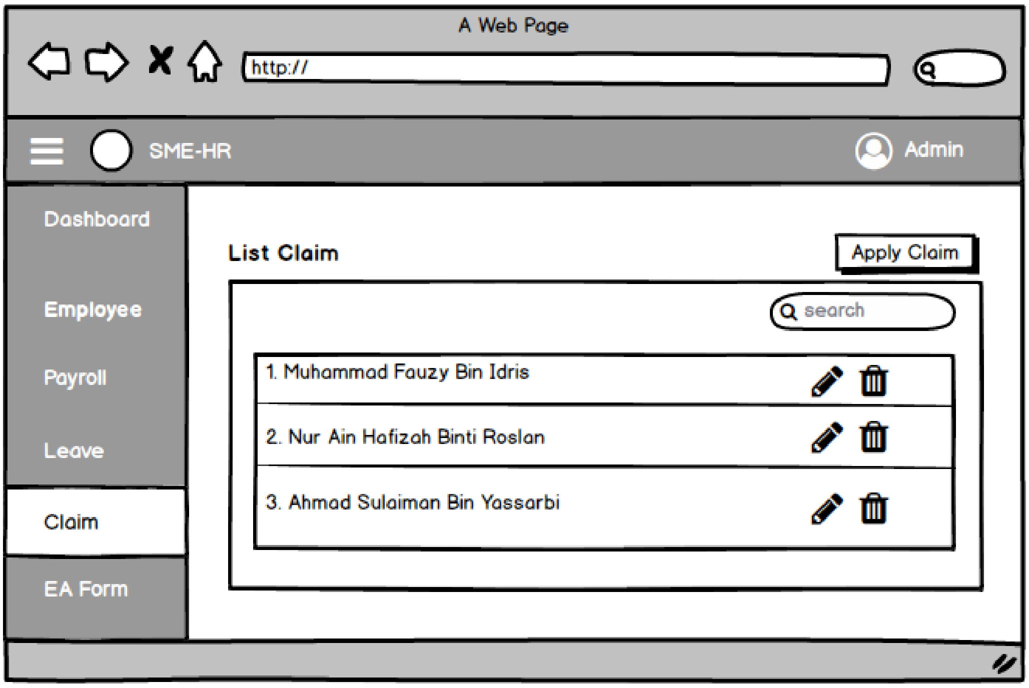


Your Data is Deleted!

Figure 59: Delete Leave Successful Page Interface

3.6.4 Manage Claim

3.6.4.1 Claim List Page



1. Muhammad Fauzy Bin Idris
2. Nur Ain Hafizah Binti Roslan
3. Ahmad Sulaiman Bin Yassarbi

Figure 60: Claim List Page Interface

3.6.4.2 Add Claim Page

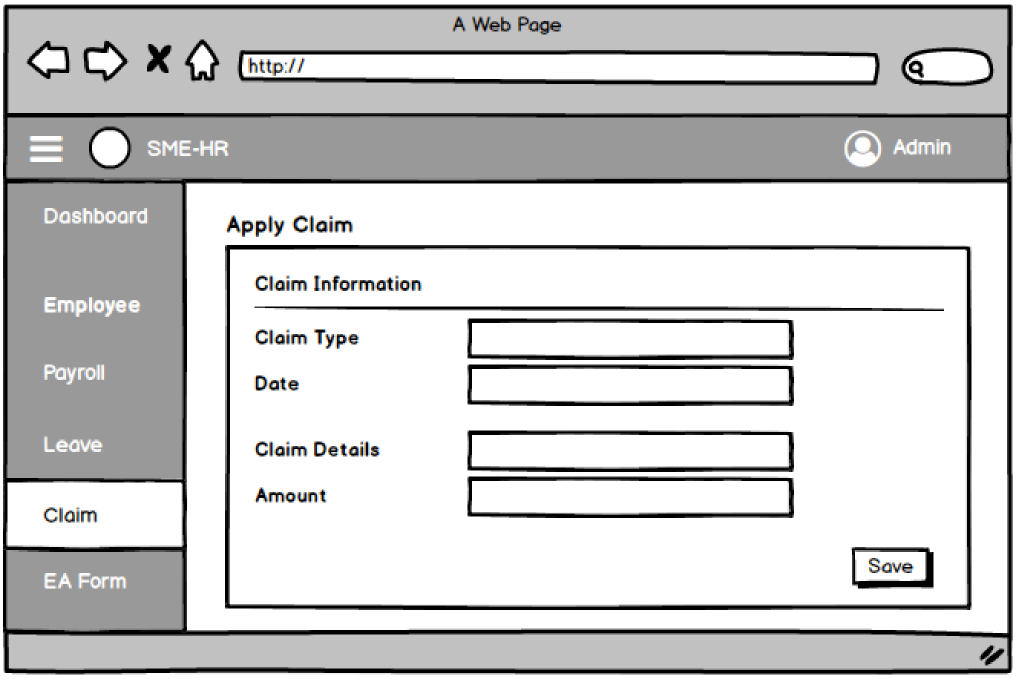
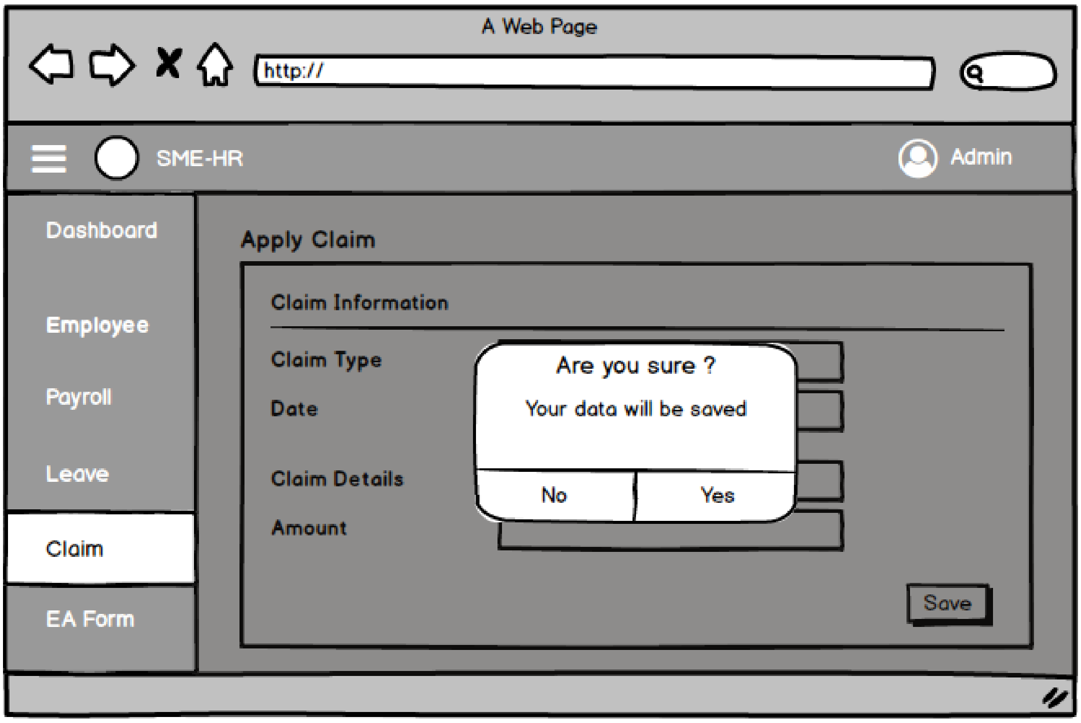


Figure 61: Add Claim Page Interface

3.6.4.3 Add Claim Confirmation Page

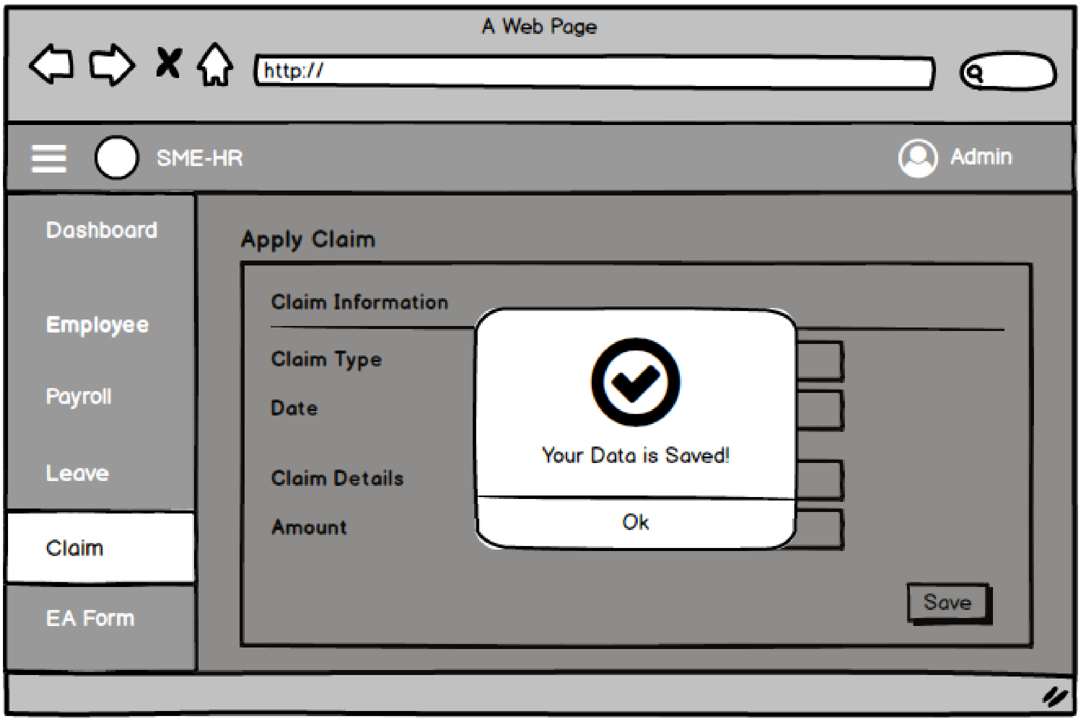


Are you sure ?

Your data will be saved

Figure 62: Add Claim Confirmation Page Interface

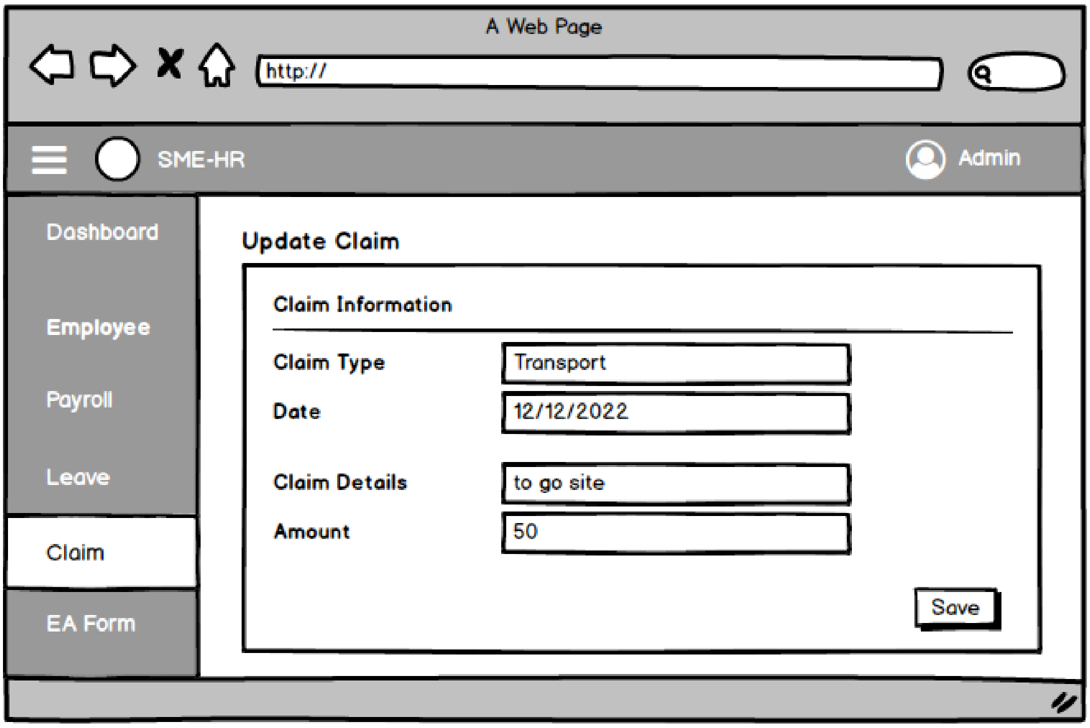
3.6.4.4 Add Claim Successful Page



Your Data is Saved!

Figure 63: Add Claim Successful Page Interface

3.6.4.5 Update Claim Page



Update Claim

Claim Information

Claim Type: Transport

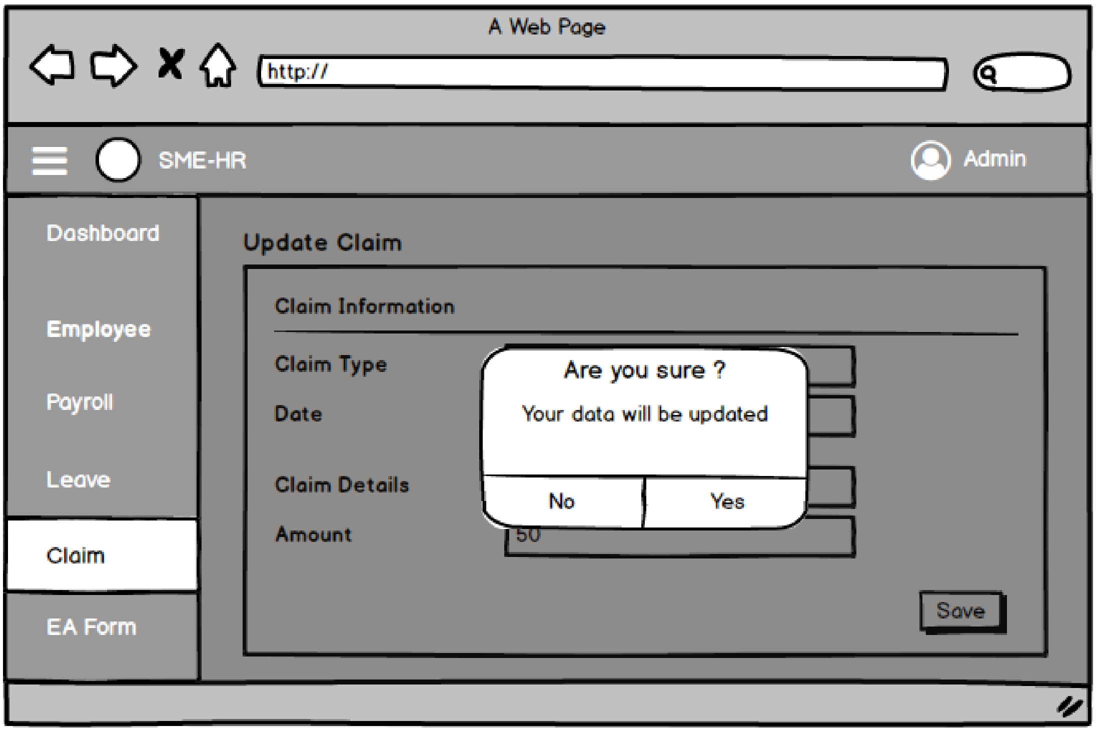
Date: 12/12/2022

Claim Details: to go site

Amount: 50

Figure 64: Update Claim Page Interface

3.6.4.6 Update Claim Confirmation Page



Update Claim

Claim Information

Claim Type

Date

Claim Details

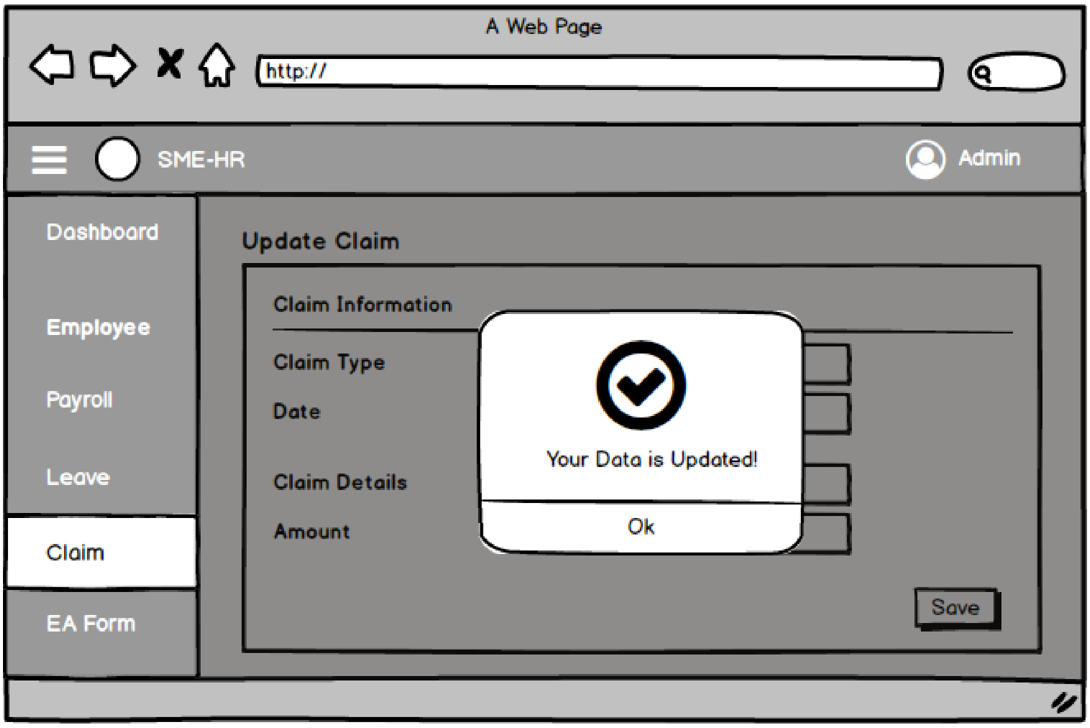
Amount: 50

Are you sure ?

Your data will be updated

Figure 65: Update Claim Confirmation Page Interface

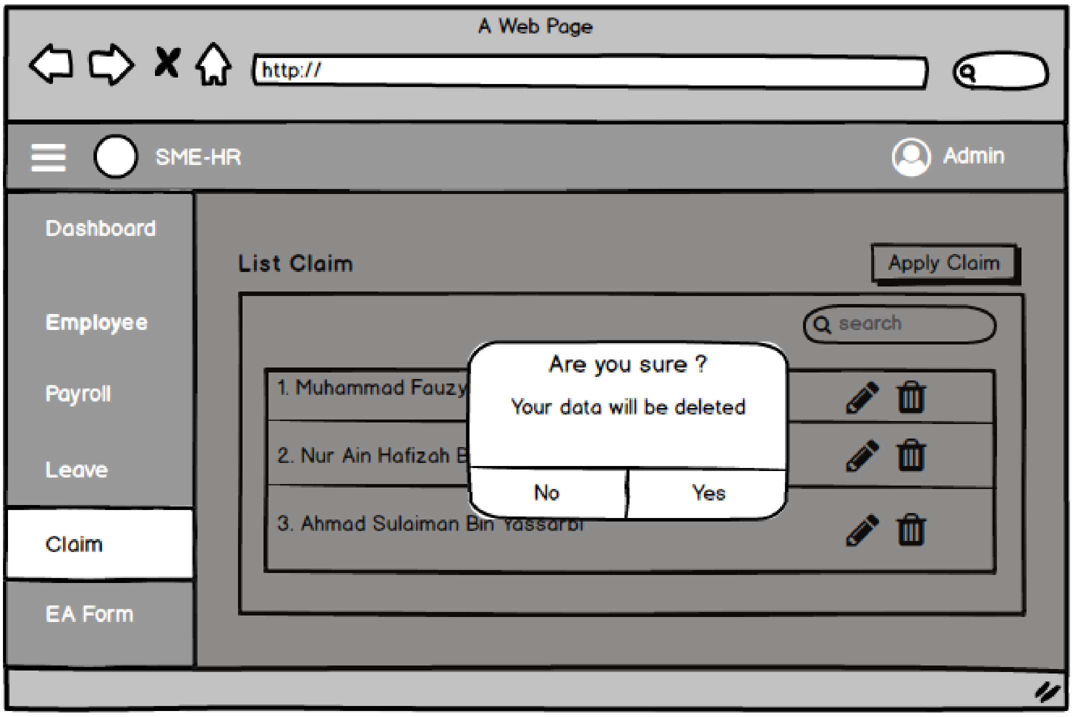
3.6.4.7 Update Claim Successful Page



Your Data is Updated!

Figure 66: Update Claim Successful Page Interface

3.6.4.8 Delete Claim Confirmation Page



1. Muhammad Fauzy

2. Nur Ain Hafizah B

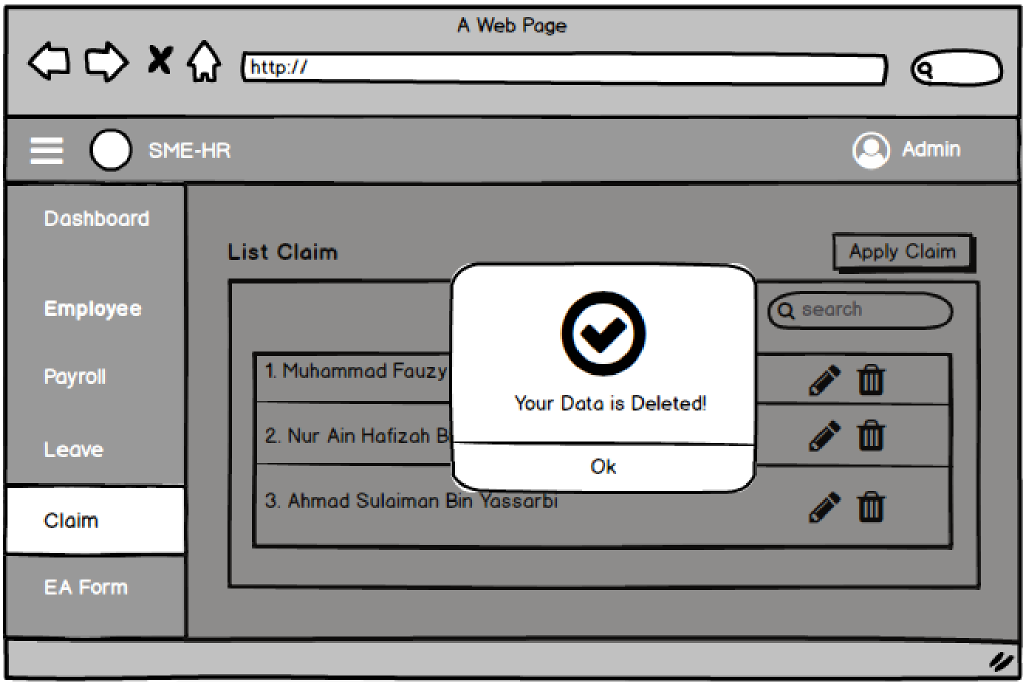
3. Ahmad Sulaiman Bin Yassarbi

Are you sure ?

Your data will be deleted

Figure 67: Delete Claim Confirmation Page Interface

3.6.4.9 Delete Claim Successful Page



1. Muhammad Fauzy

2. Nur Ain Hafizah B

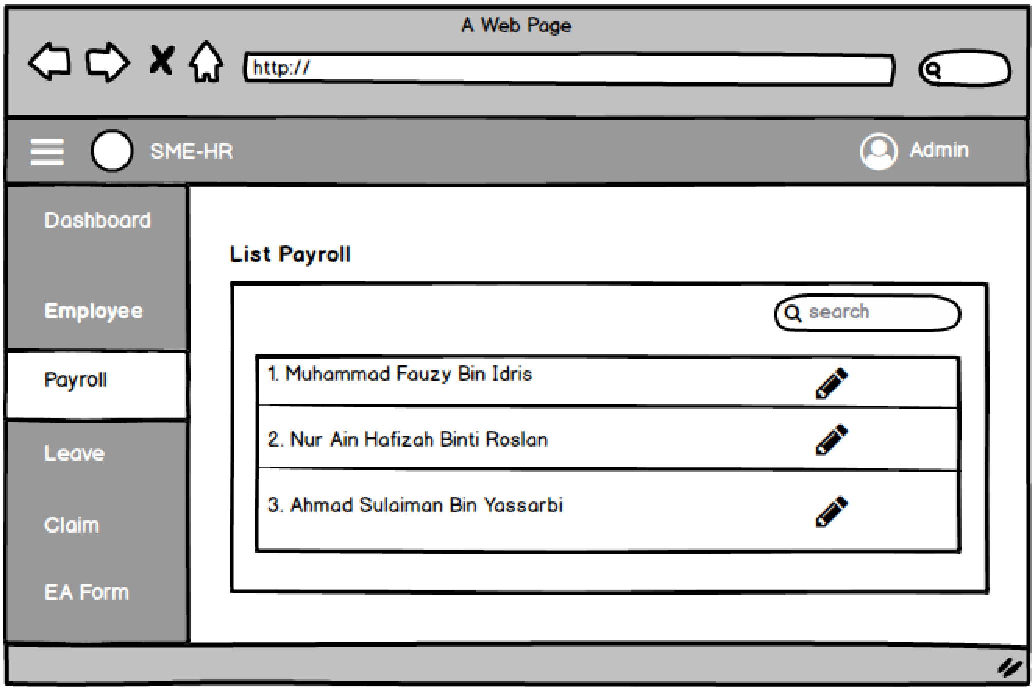
3. Ahmad Sulaiman Bin Yassarbi

Your Data is Deleted!

Figure 68: Delete Claim Successful Page Interface

3.6.5 Manage Payroll

3.6.5.1 Payroll List Page



1. Muhammad Fauzy Bin Idris

2. Nur Ain Hafizah Binti Roslan

3. Ahmad Sulaiman Bin Yassarbi

Figure 69: Payroll List Page Interface

3.6.5.2 Add Payroll Page

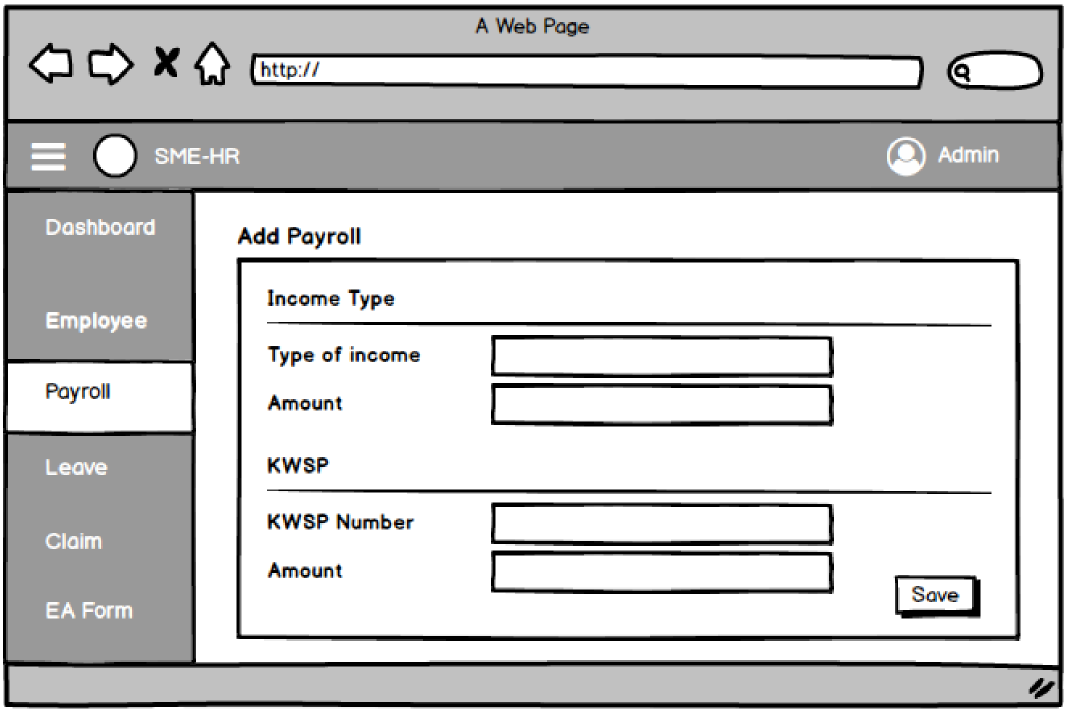


Figure 70: Add Payroll Page Interface

3.6.5.3 Add Payroll Confirmation Page

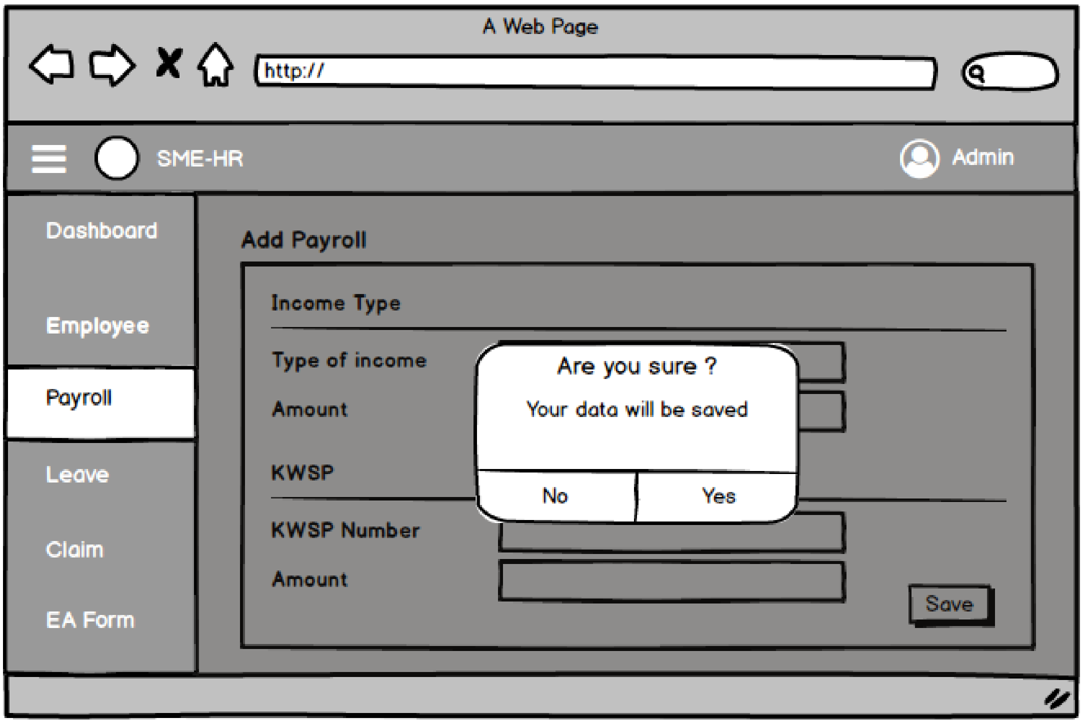


Figure 71: Add Payroll Confirmation Page Interface

3.6.5.4 Add Payroll Successful Page

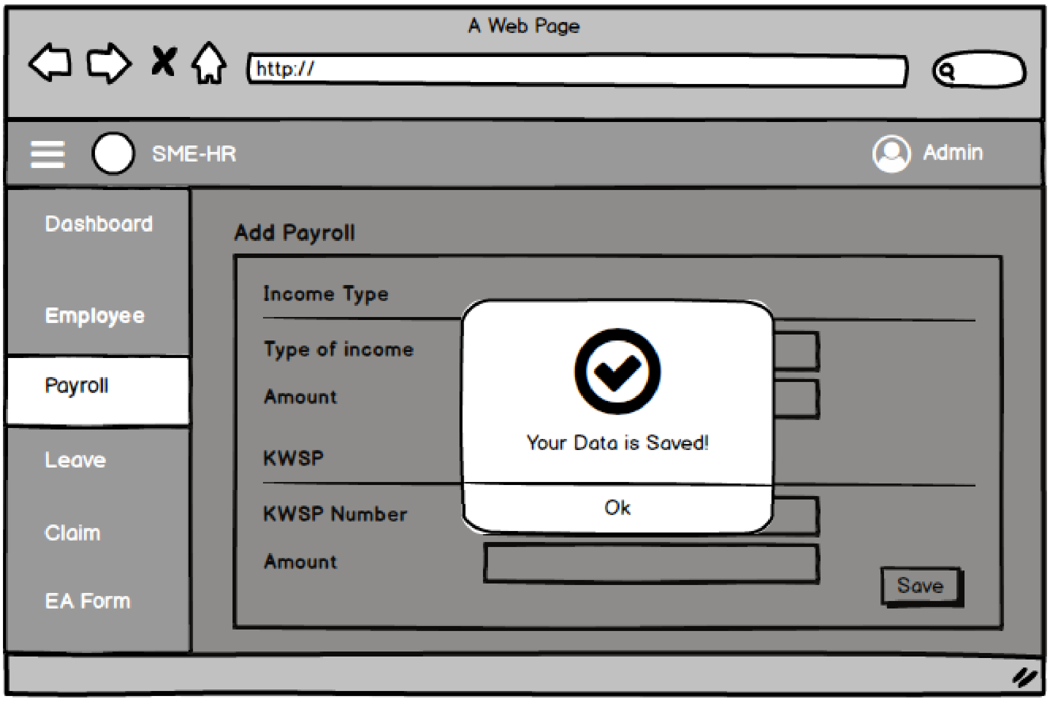
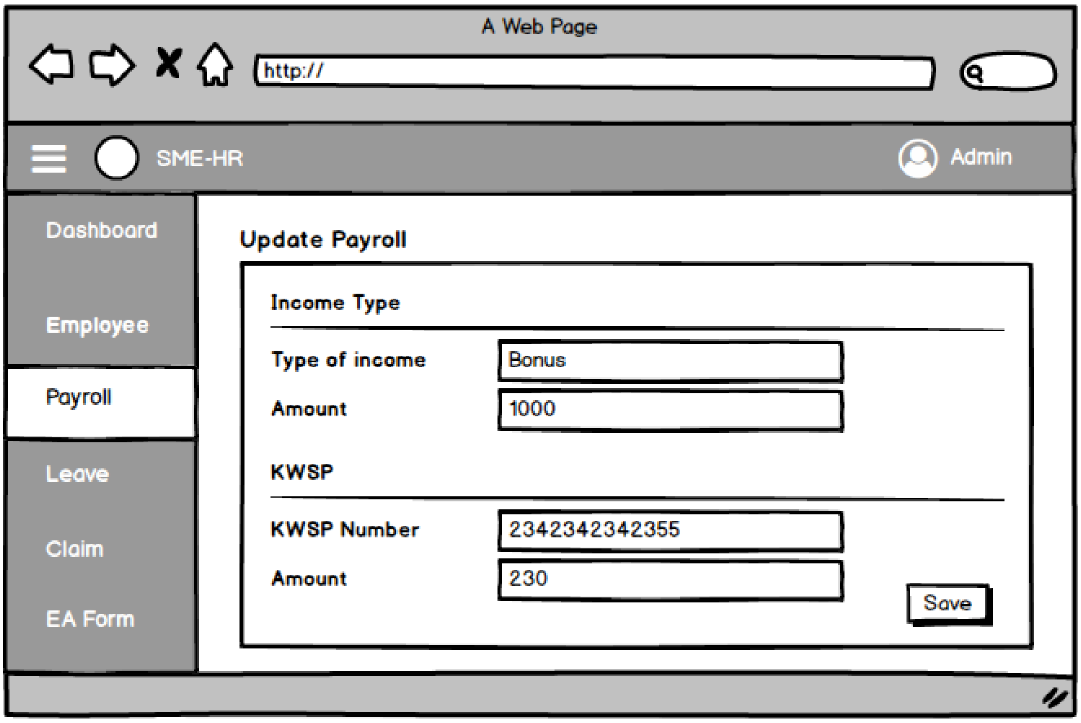


Figure 72: Add Payroll Successful Page Interface

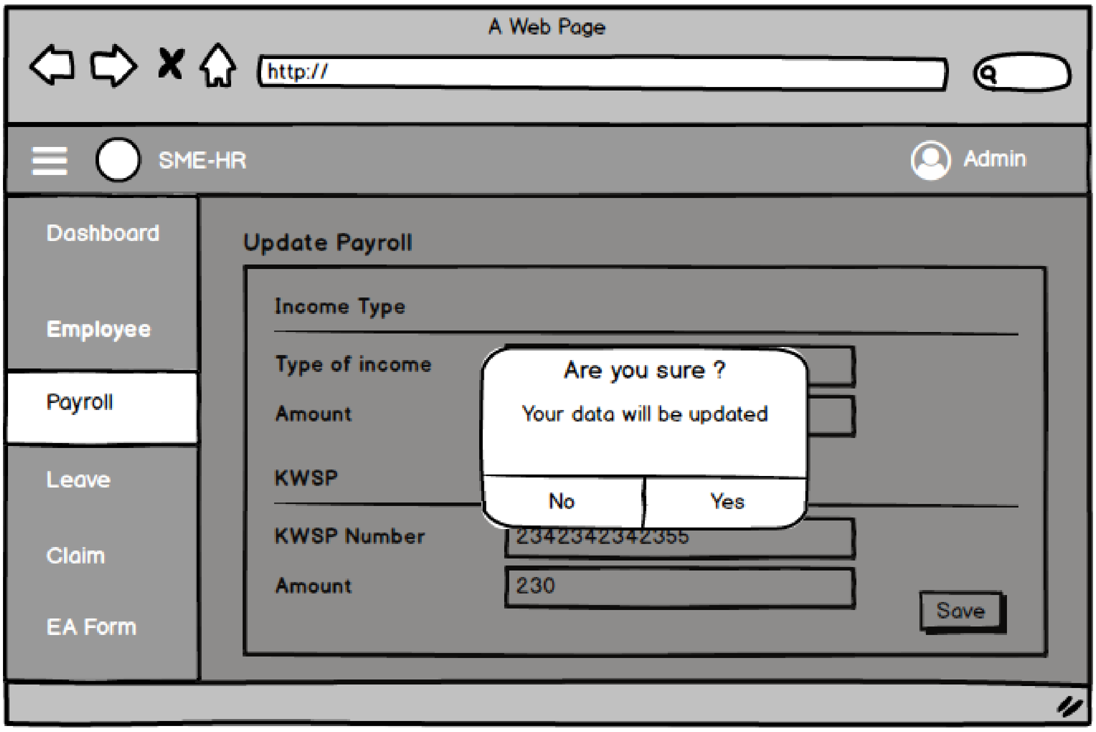
3.6.5.5 Update Payroll Page



Type of income: Bonus
Amount: 1000
KWSP Number: 2342342342355
Amount: 230

Figure 73: Update Payroll Page Interface

3.6.5.6 Update Payroll Confirmation Page



Are you sure ?

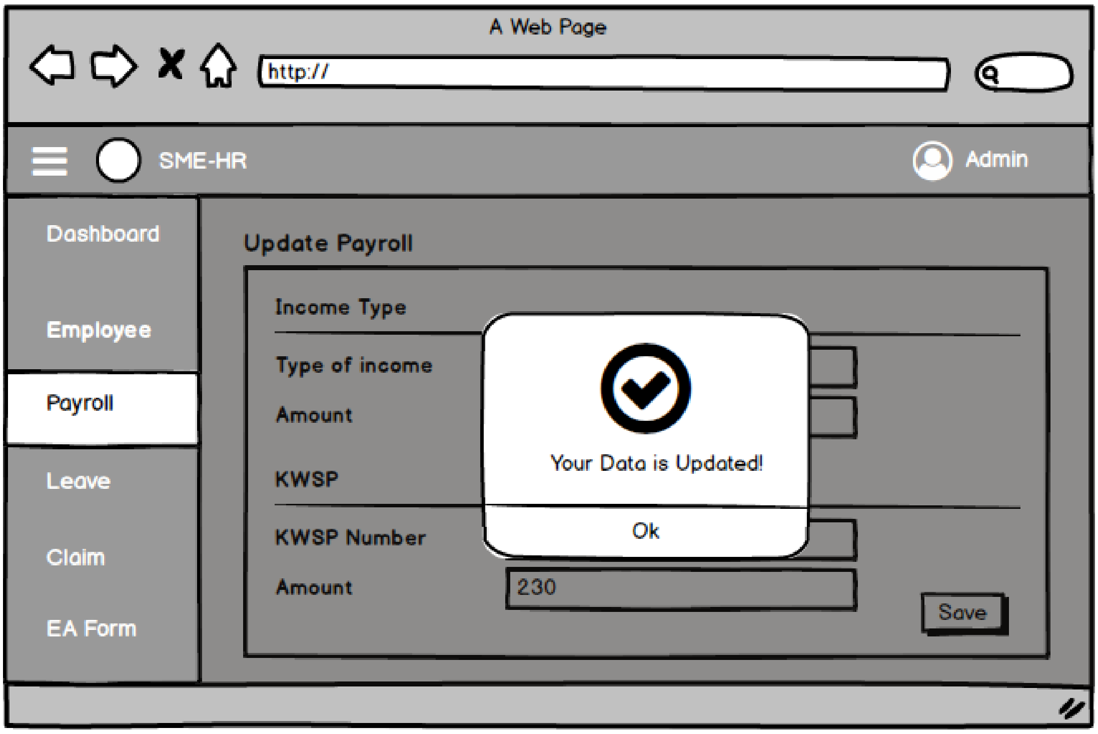
Your data will be updated

KWSP Number: 2342342342355

Amount: 230

Figure 74: Update Payroll Confirmation Page Interface

3.6.5.7 Update Payroll Successful Page



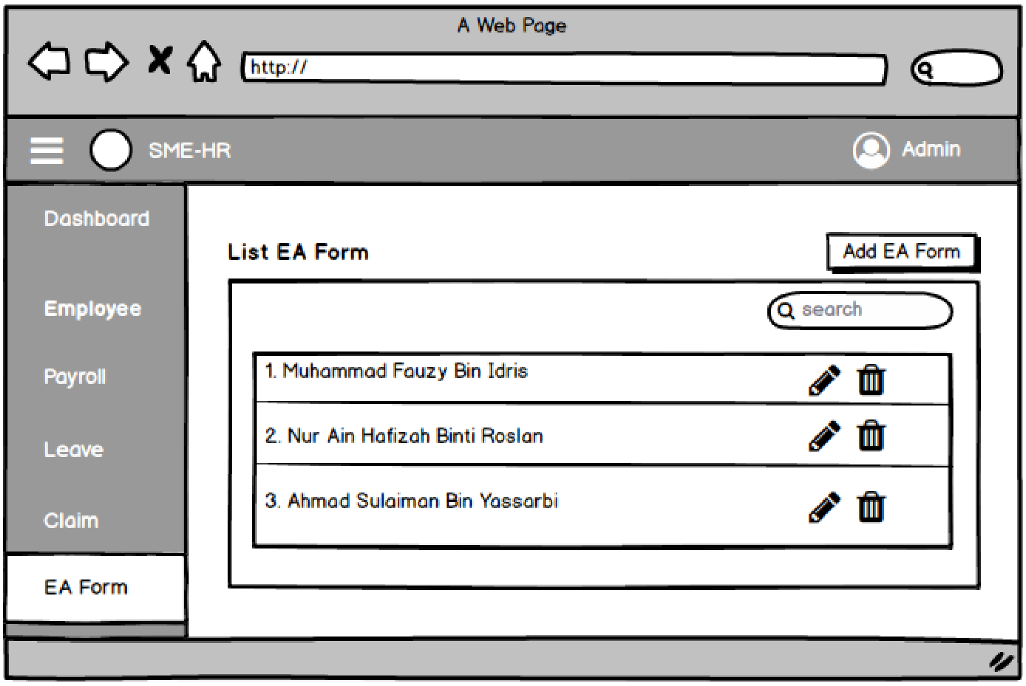
Your Data is Updated!

Amount: 230

Figure 75: Update Payroll Successful Page Interface

3.6.6 Manage EA Form

3.6.6.1 EA Form List Page

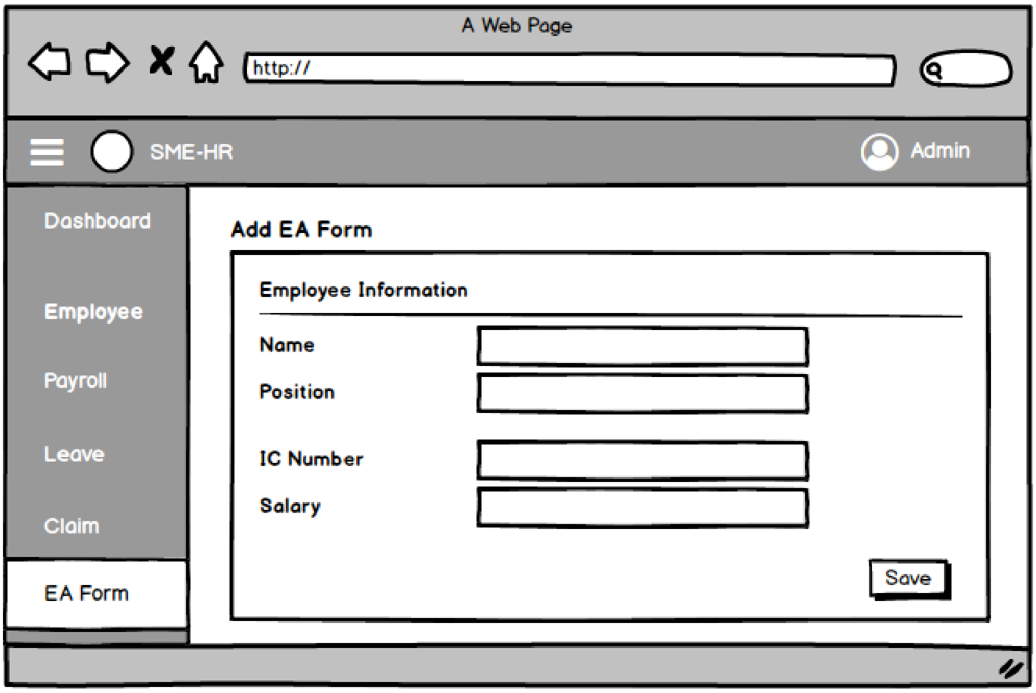


List EA Form

1. Muhammad Fauzy Bin Idris
2. Nur Ain Hafizah Binti Roslan
3. Ahmad Sulaiman Bin Yassarbi

Figure 76: EA Form List Page Interface

3.6.6.2 Add EA Form Page



Add EA Form

Employee Information

Name

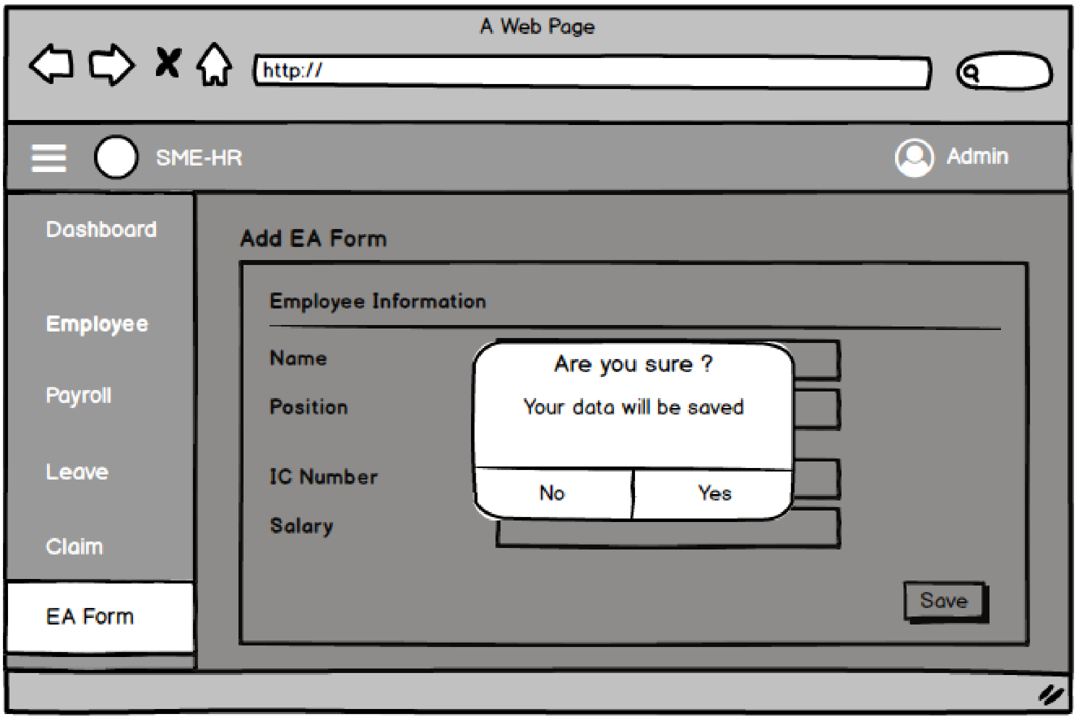
Position

IC Number

Salary

Figure 77: Add EA Form Page Interface

3.6.6.3 Add EA Form Confirmation Page

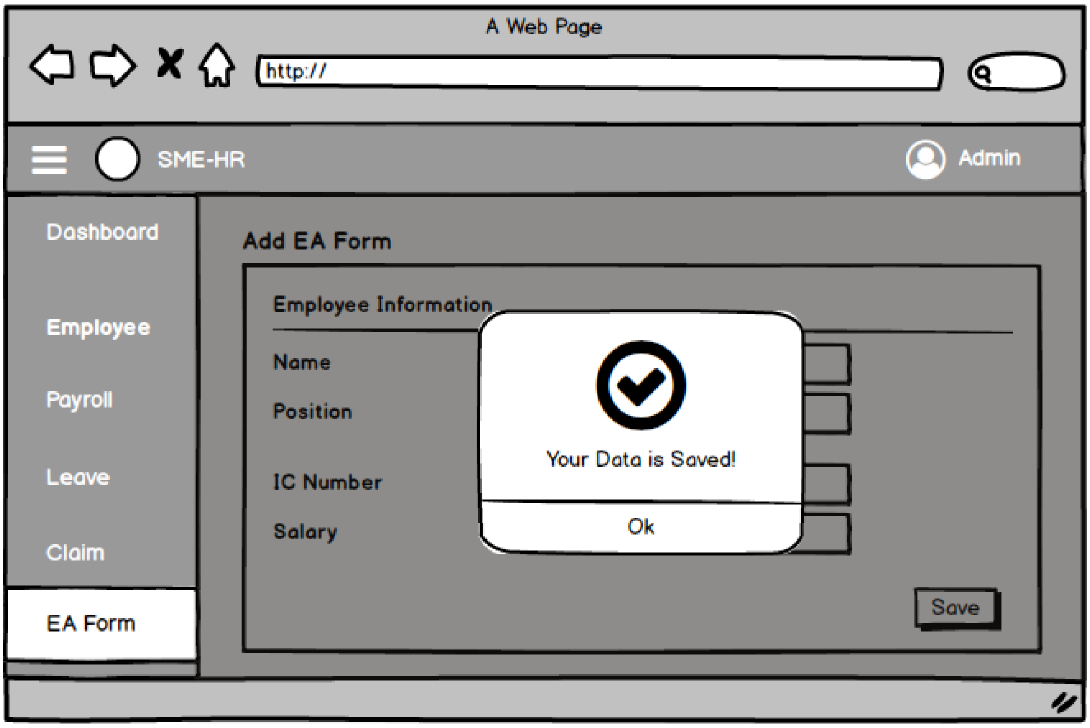


Are you sure ?

Your data will be saved

Figure 78: Add EA Form Confirmation Page Interface

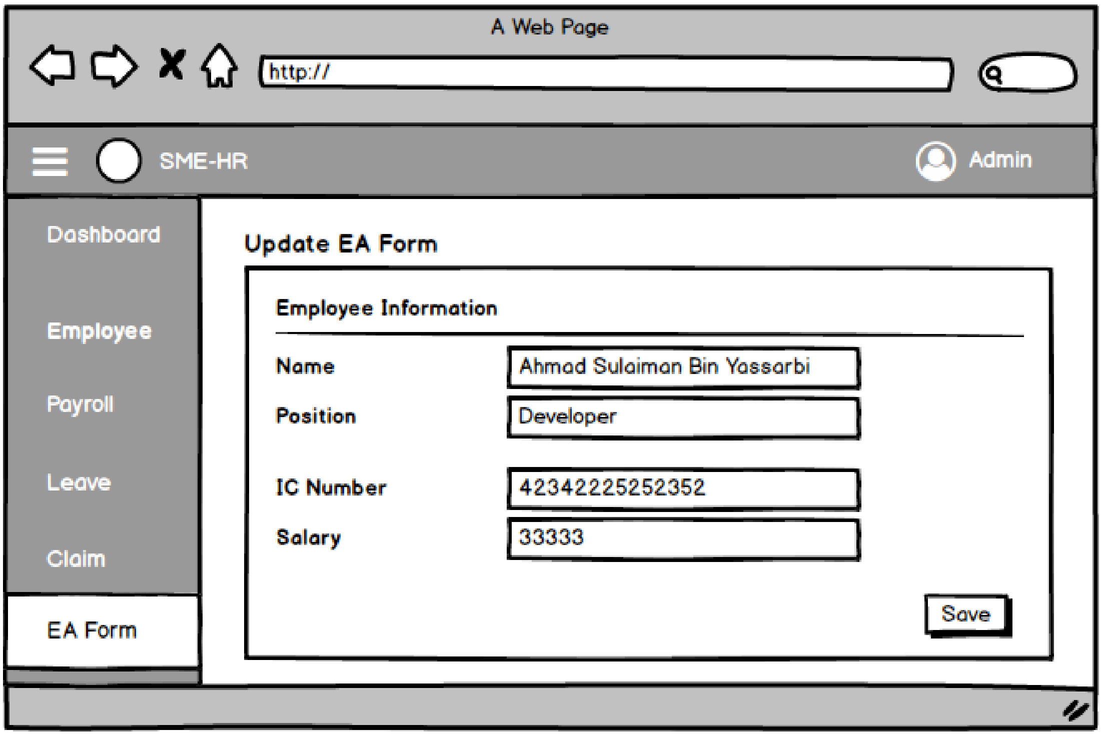
3.6.6.4 Add EA Form Successful Page



Your Data is Saved!

Figure 79: Add EA Form Successful Page Interface

3.6.6.5 Update EA Form Page



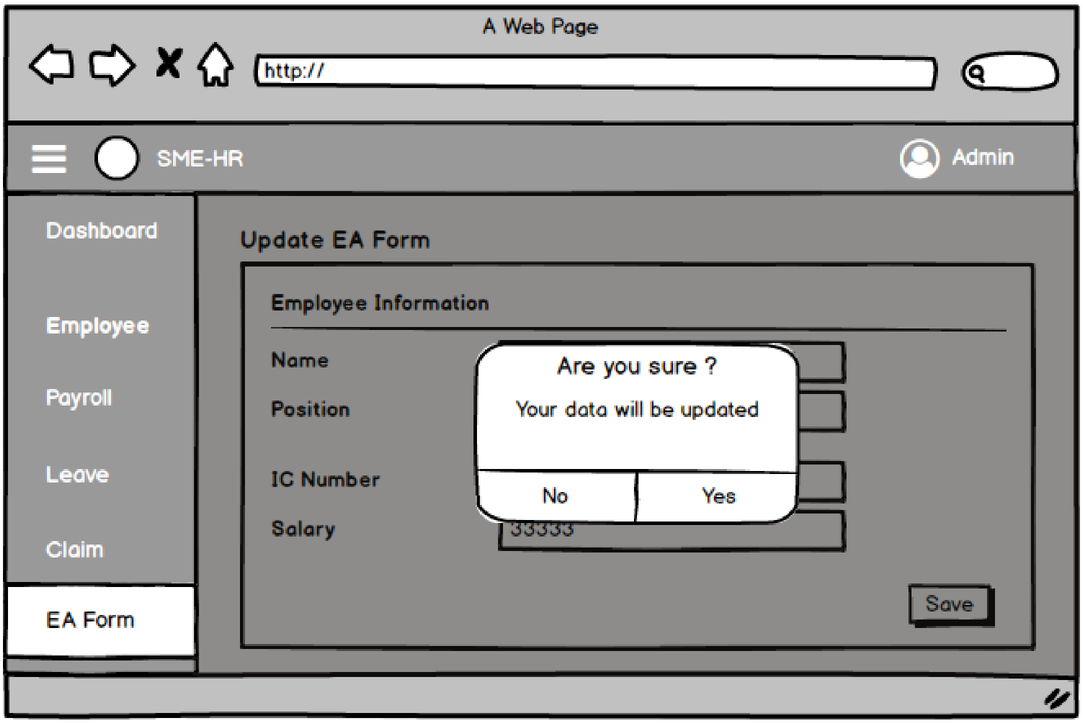
Update EA Form

Employee Information

Name	Ahmad Sulaiman Bin Yassarbi
Position	Developer
IC Number	42342225252352
Salary	33333

Figure 80: Update EA Form Page Interface

3.6.6.6 Update EA Form Confirmation Page



Update EA Form

Employee Information

Name

Position

IC Number

Salary

33333

Are you sure ?

Your data will be updated

Figure 81: Update EA Form Confirmation Page Interface

3.6.6.7 Update EA Form Successful Page

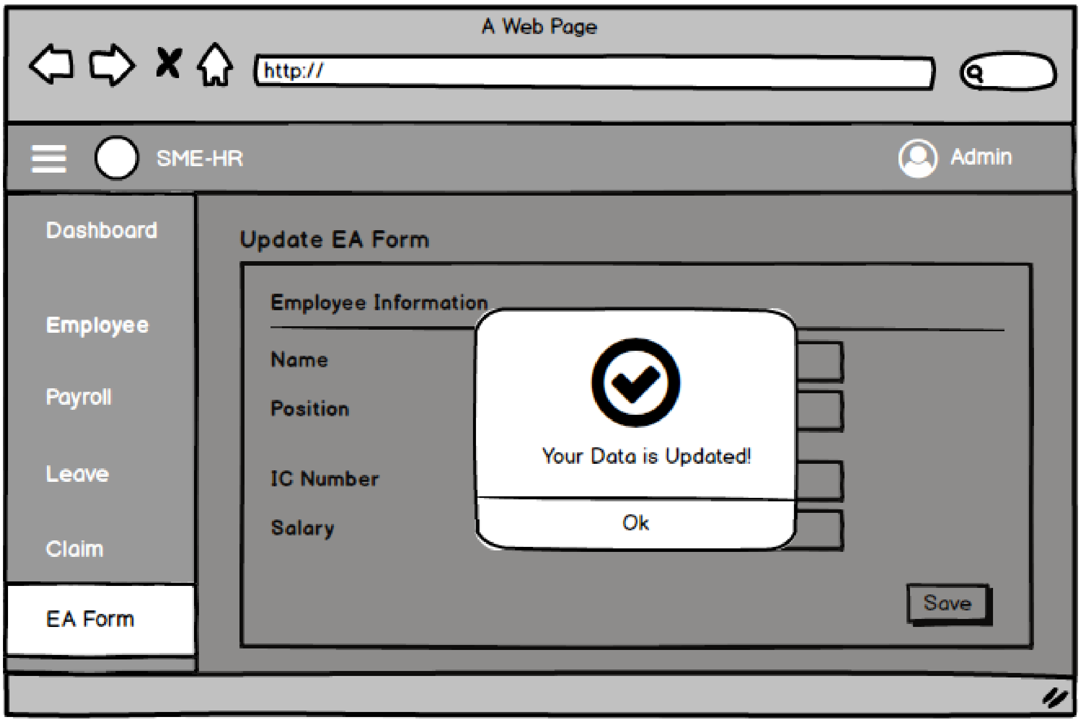


Figure 82: Update EA Form Successful Page Interface

3.6.6.8 Delete EA Form Confirmation Page

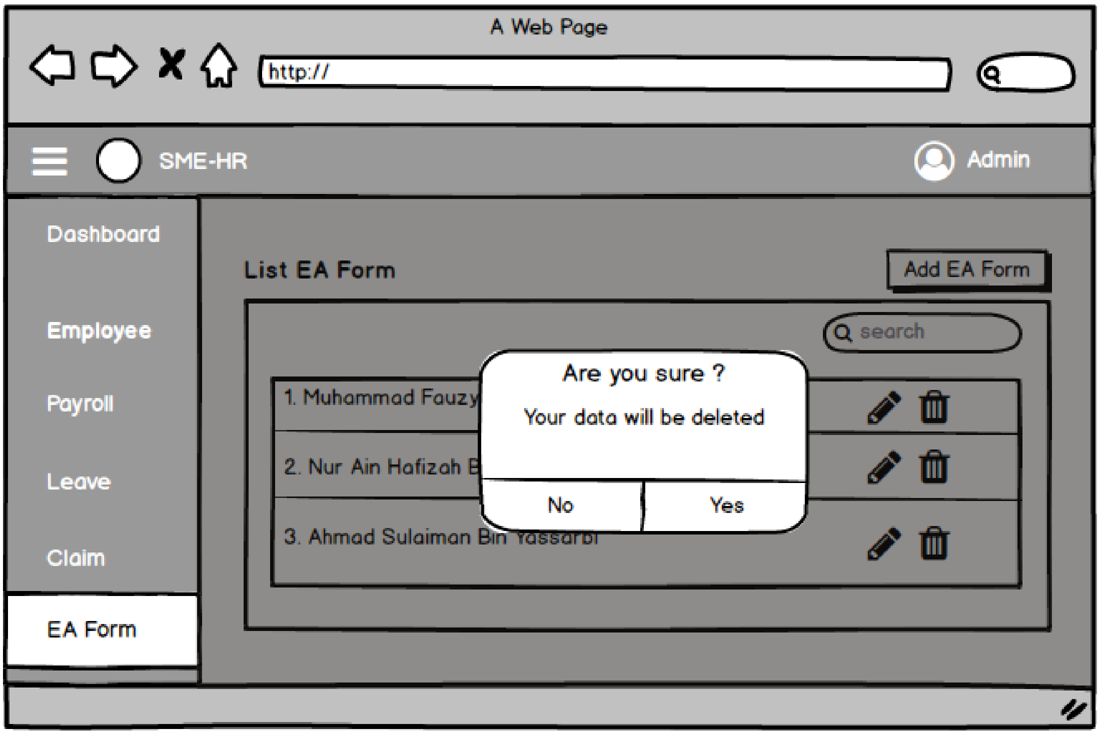


Figure 83: Delete EA Form Confirmation Page Interface

3.6.6.9 Delete EA Form Successful Page

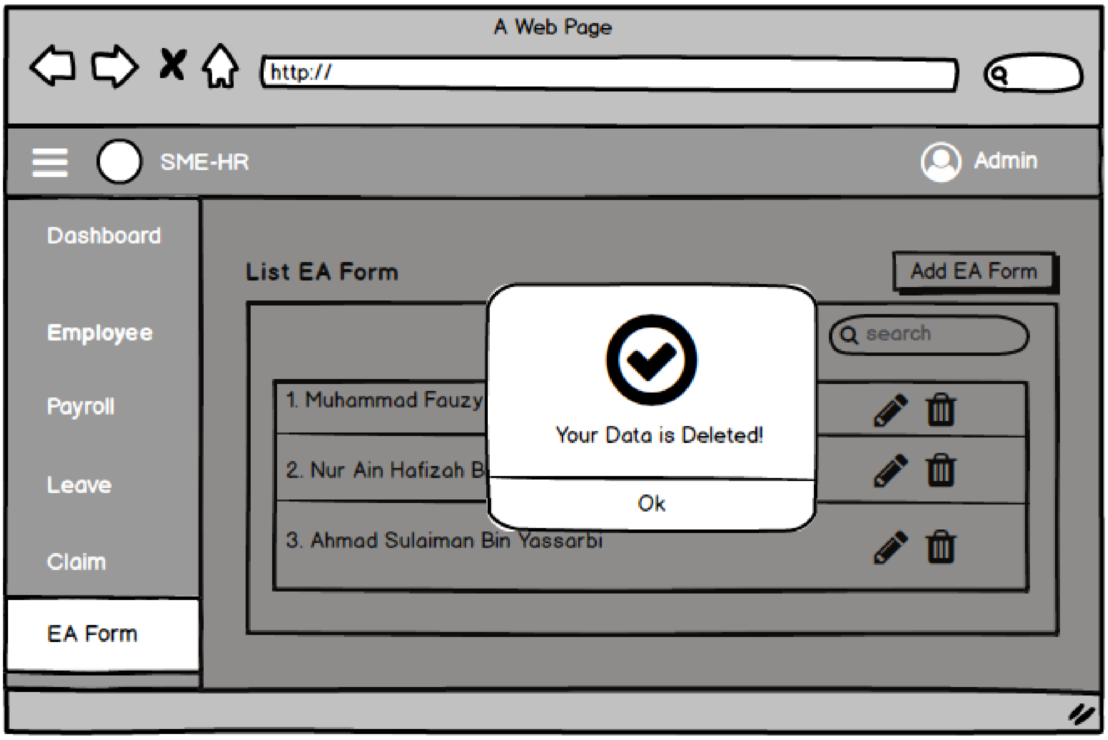


Figure 84: Delete EA Form Successful Page Interface

3.7 Testing Plan

This section explains the testing plan that will be used to test the functionality of SME-HR system in order to achieve the objective of this project. User Acceptance Test (UAT) will be performed by the end user which is Anh Exora Enterprises to measure the capability of all functions in the system.

Table 26: User Acceptance Test (UAT)

No	Module	Activity	Status		Comment
			Pass	Fail	
1.	Login	Login into the system			
2.	Manage Employee	Add new employee			
		View employee information			
		Update employee information			
		Delete employee information			
		Search specific employee			
3.	Manage Profile	View profile information			
		Update profile information			
4.	Manage Payroll	Add new payroll information			
		View payroll information			
		Update payroll information			
		Search specific payroll information			
		Filter specific payroll information			
5.		Add new leave (apply leave)			

	Manage Leave	View leave information			
		Update leave information			
		Delete leave information			
		Search specific leave information			
6.	Manage Claim	Add new claim (apply claim)			
		View claim information			
		Update claim information			
		Delete claim information			
		Search specific claim information			
7.	Manage EA Form	Add new EA form			
		View EA form information			
		Update EA form information			
		Delete EA form information			
		Search specific EA form			
		Filter specific EA form			

This test has been performed by:

Company Name : ____
Client Name : ____
Email : ____
Date : ____

3.8 Potential Use of Proposed Solution

The Human Resources System for Small and Medium-Sized Enterprises (SME-HR) is a powerful tool that can streamline and automate various HR processes within SMEs. It can efficiently manage employees, leave, payroll, claims and EA forms. This system can help SMEs track and manage employee information, such as contact details, job titles, and performance evaluations. It can also track employee leave requests, approvals, and balances, ensuring that the SME has enough staff to meet its needs while reducing the administrative burden of managing leave. Additionally, SME-HR can be used to manage payroll, including calculating and distributing employee salaries and taxes, which can help SMEs to ensure compliance with labour laws and regulations and reduce the administrative burden of managing payroll.

Furthermore, SME-HR can be used to track and manage employee claims, such as medical and reimbursement claims, which can help SMEs ensure compliance with labour laws and regulations and reduce the administrative burden of managing claims. Additionally, SME-HR can be used to manage and keep track of employee EA form, which is mandatory for companies to submit to the government, and it can help SMEs to ensure compliance with labour laws and regulations and reduce the administrative burden of managing EA form. Overall, the SME-HR system can help Small and Medium-Sized Enterprises improve their efficiency, compliance, and decision-making process, leading to a more productive and profitable business.

3.9 Gantt Chart

A Gantt chart is used to schedule the SME-HR project to ensure this project can be finished on time.

Num	Task Description	Time (Day)	Start	Finish	October				November				December				January				February				March					
					1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2
	Requirements Planning																													
1	Define the project scope for the SME-HR system	14	1/10/2022	15/10/2022	■	■																								
2	Conduct stakeholder interviews and gather requirements	14	1/10/2022	15/10/2022	■	■																								
3	Develop a requirements document	21	16/10/2022	6/11/2022			■	■	■																					
4	Prioritize requirements and determine feasibility	21	7/11/2022	28/11/2022						■	■	■																		
5	Create a project plan and schedule	21	1/12/2022	22/12/2022									■	■	■															
	User Design																													
1	Develop wireframes or mockups of the SME-HR system user interface	21	25/12/2022	15/1/2023												■	■	■												
2	Specify system functionalities	21	25/12/2022	15/1/2023												■	■	■												
3	Create a rough prototype of the system	21	25/12/2022	15/1/2023												■	■	■												
4	Test the prototype with stakeholders and gather feedback	21	16/1/2023	6/2/2023															■	■	■									
5	Refine the design based on feedback	21	16/1/2023	6/2/2023															■	■	■									

	Construction																														
1	Develop the SME-HR system based on the design specifications	70	7/2/2023	18/4/2023																		■	■	■	■	■	■	■	■	■	■
3	Developing the core functionality of the SME-HR system	70	7/2/2023	18/4/2023																		■	■	■	■	■	■	■	■	■	■
4	Implement the user interface and user experience (UI/UX)	70	7/2/2023	18/4/2023																		■	■	■	■	■	■	■	■	■	■
5	Review progress of work	70	7/2/2023	18/4/2023																		■	■	■	■	■	■	■	■	■	■
6	Test the system	21	19/4/2023	10/5/2023																											
7	Fix any bugs that are discovered	21	19/4/2023	10/5/2023																											
8	Ensure the system meets specification	7	11/5/2023	18/5/2023																											
9	Work with stakeholders to resolve any issues	14	11/5/2023	25/5/2023																											
	Cutover																														
1	Complete the development of the SME-HR system	28	1/6/2023	29/6/2023																											
2	Conduct training sessions for employess	14	1/7/2023	15/7/2023																											
3	Deploying the SME-HR system to the production	14	1/7/2023	15/7/2023																											
4	Conducting final testing after deployment	14	1/7/2023	15/7/2023																											
5	Ready for User Acceptance Test (UAT)	14	1/7/2023	15/7/2023																											
6	Deliver the final product and provide ongoing support as needed	14	16/7/2023	30/7/2023																											

CHAPTER 4

IMPLEMENTATION, RESULT AND DISCUSSION

4.1 Introduction

This chapter presents a discussion on the implementation and outcomes of the Human Resources System developed specifically for Small and Medium-Sized Enterprises (SME-HR). The intention is to provide a detailed explanation of the system's interfaces and their functions, using figures and comprehensive explanations based on the functionalities previously outlined. The ultimate objective is to offer a comprehensive understanding of the design and operation of the SME-HR system.

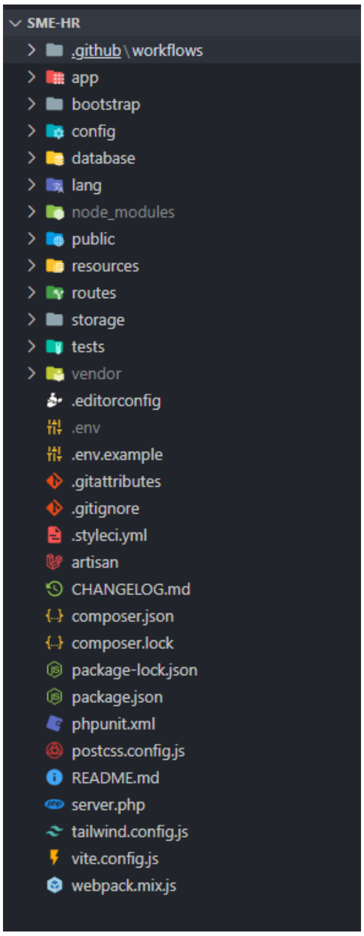
4.2 Implementation Process

The implementation process involves documenting all the steps to develop the SME-HR system. The SME-HR system is a responsive web-based application that can be accessed through both desktop and mobile interfaces. This phase aims to elucidate the development process of all the modules necessary for the system's proper functioning.

4.2.1 Development of Environment

Figure 85: Laravel - PHP Framework

This system has been developed using Laravel. Laravel is a PHP-based, open-source web framework utilized for developing web applications. It was initially launched in 2011 and created by Taylor Otwell (*Exploring the History and Evolution of the Laravel Framework*, n.d.). Laravel aims to provide a more enjoyable and less repetitive web development experience by offering an elegant syntax and various useful tools to simplify common tasks, including routing, authentication, caching, and database management. The framework follows the Model-View-Controller (MVC) architectural pattern and has a substantial and active community contributing to its development and upkeep (Sunardi & Suharjito, 2019).



```
SME-HR
  > .github\workflows
  > app
  > bootstrap
  > config
  > database
  > lang
  > node_modules
  > public
  > resources
  > routes
  > storage
  > tests
  > vendor
    .editorconfig
    .env
    .env.example
    .gitattributes
    .gitignore
    .styleci.yml
    artisan
    CHANGELOG.md
    composer.json
    composer.lock
    package-lock.json
    package.json
    phpunit.xml
    postcss.config.js
    README.md
    server.php
    tailwind.config.js
    vite.config.js
    webpack.mix.js
```

Figure 86: Project Structure Environment in Visual Studio Code

The structure of SME-HR project is shown in Figure 86. The project is compartmentalized into several packages, each serving a particular function within the system. The fundamental functionality required to manage the user input and provide data, including controllers, models, and views, and all these files can be found in the "app" folder. The "public" folder contains static assets, such as photos and CSS stylesheets, which enhance the user interface. Additionally, all interface files for the page are in the "resources" folder. This organizational structure facilitates efficient management and maintenance of the system, ensuring seamless operation and ease of development.

4.2.2 Development of Employee Module

The employee module is a module that involves all users, which is Huma Resources (HR) Administrator, Staff, and Manager. The HR Administrator can create an account for each staff member within the company. Furthermore, the HR Administrator has the privilege of updating employee details. The HR Administrator, Staff, and Manager can all view employee information. However, the HR Administrator alone can update, delete, and add employee information, while Staff and Manager can solely view their respective information.

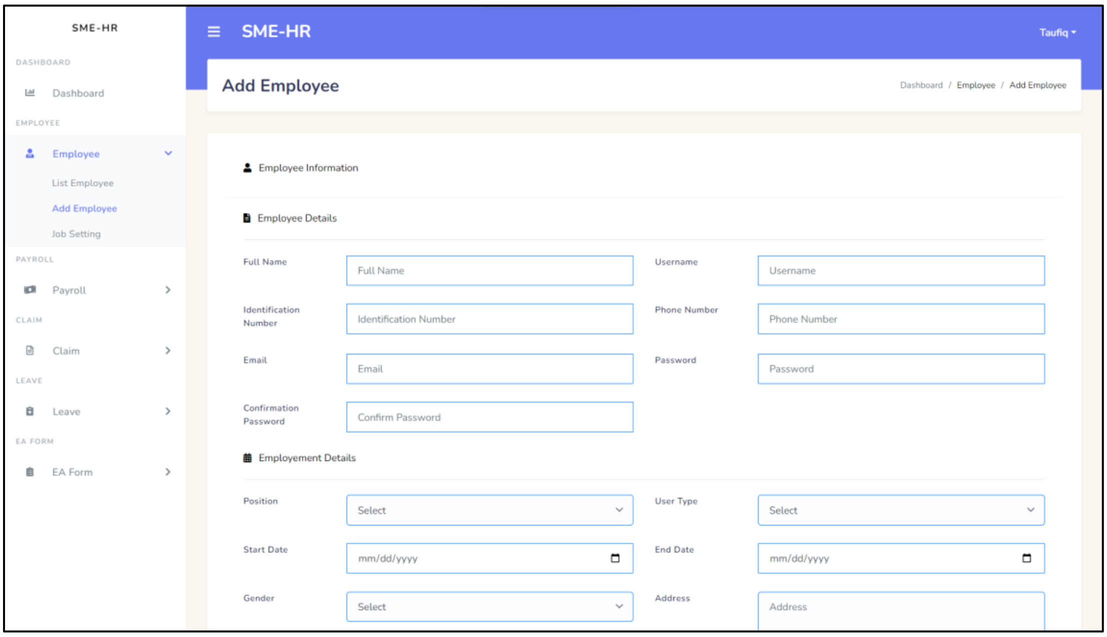
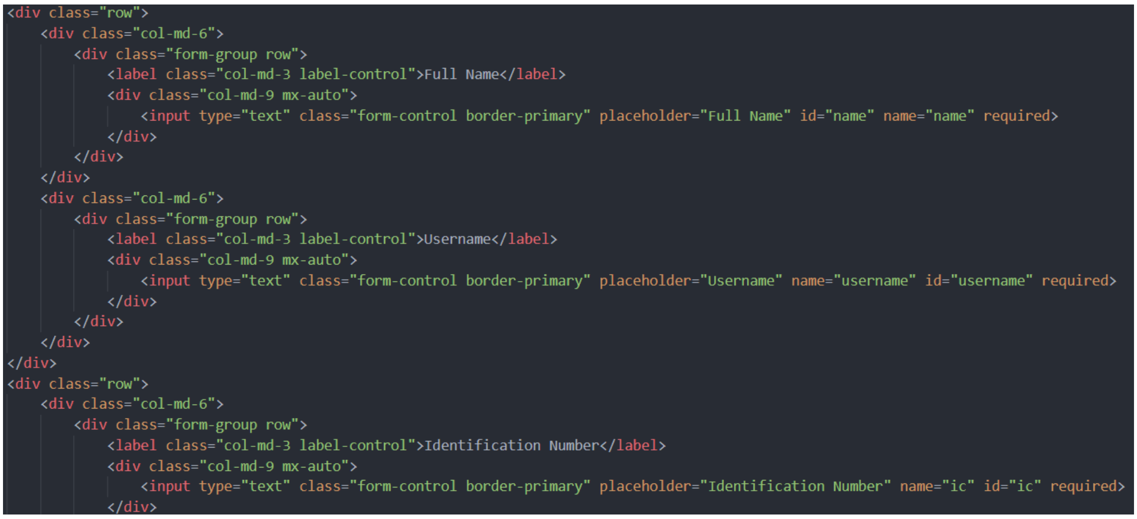


Figure 87: Add Employee Page Interface

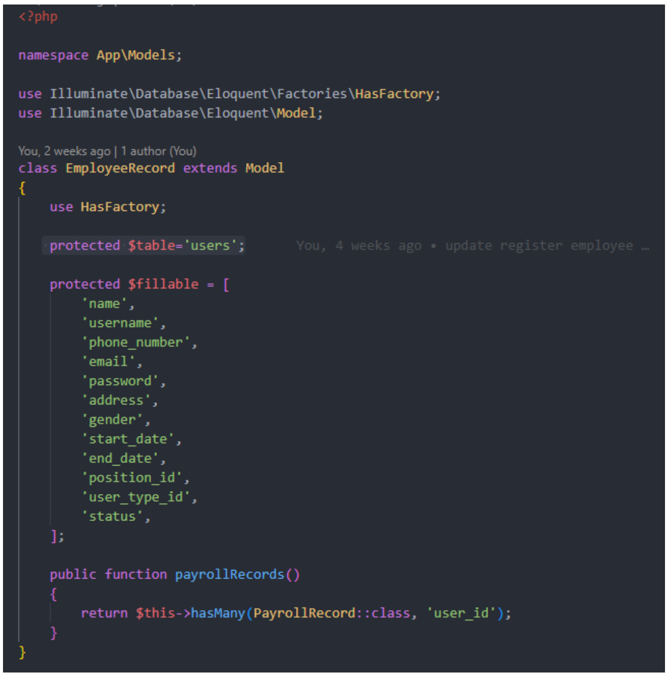
Figure 87 illustrates the interface for the "Add Employee" page, where the HR Administrator can register new staff by inputting specific information in the designated input boxes. In addition, certain information must be provided, including the employee's full name, username, email, and password, along with employment details such as position, user type, address, and start date. This process allows for streamlined and efficient employee registration, ensuring that the necessary information is entered accurately and comprehensively.



```
<div class="row">
    <div class="col-md-6">
        <div class="form-group row">
            <label class="col-md-3 label-control">Full Name</label>
            <div class="col-md-9 mx-auto">
                <input type="text" class="form-control border-primary" placeholder="Full Name" id="name" name="name" required>
            </div>
        </div>
    </div>
    <div class="col-md-6">
        <div class="form-group row">
            <label class="col-md-3 label-control">Username</label>
            <div class="col-md-9 mx-auto">
                <input type="text" class="form-control border-primary" placeholder="Username" name="username" id="username" required>
            </div>
        </div>
    </div>
</div>
<div class="row">
    <div class="col-md-6">
        <div class="form-group row">
            <label class="col-md-3 label-control">Identification Number</label>
            <div class="col-md-9 mx-auto">
                <input type="text" class="form-control border-primary" placeholder="Identification Number" name="ic" id="ic" required>
            </div>
```

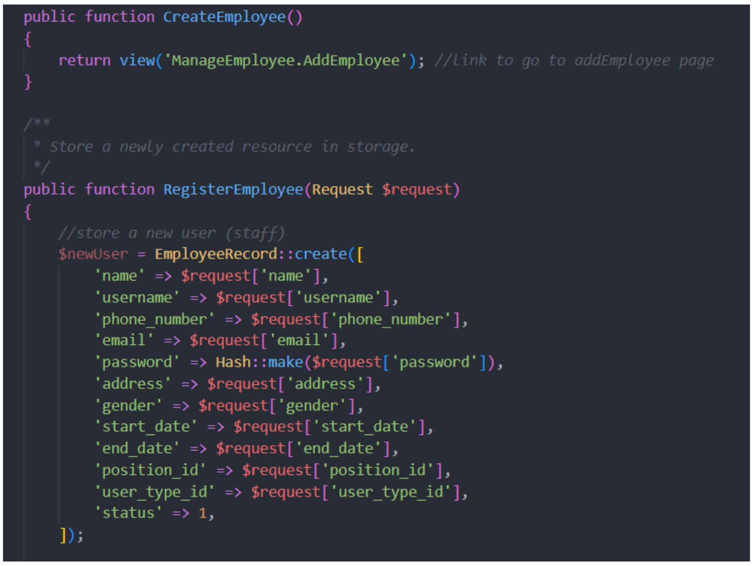
Figure 88: Employee View (Add Employee) Snippet Code

Figure 88 displays a code snippet of the "Add Employee" page, which outlines the structure for a form that facilitates registering new staff information within the system. This code defines several essential fields required for registration, including full name, username, and identification number. Moreover, the form employs a variety of components, such as text input, date input, password, and text area, to ensure the completeness and accuracy of the employee's data. The utilization of these components serves to enhance the user experience and simplify the data input process.



```
<?php

namespace App\Models;

use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Database\Eloquent\Model;

You, 2 weeks ago | 1 author (You)
class EmployeeRecord extends Model
{
    use HasFactory;

    protected $table='users';      You, 4 weeks ago • update register employee …

    protected $fillable = [
        'name',
        'username',
        'phone_number',
        'email',
        'password',
        'address',
        'gender',
        'start_date',
        'end_date',
        'position_id',
        'user_type_id',
        'status',
    ];

    public function payrollRecords()
    {
        return $this->hasMany(PayrollRecord::class, 'user_id');
    }
}
```

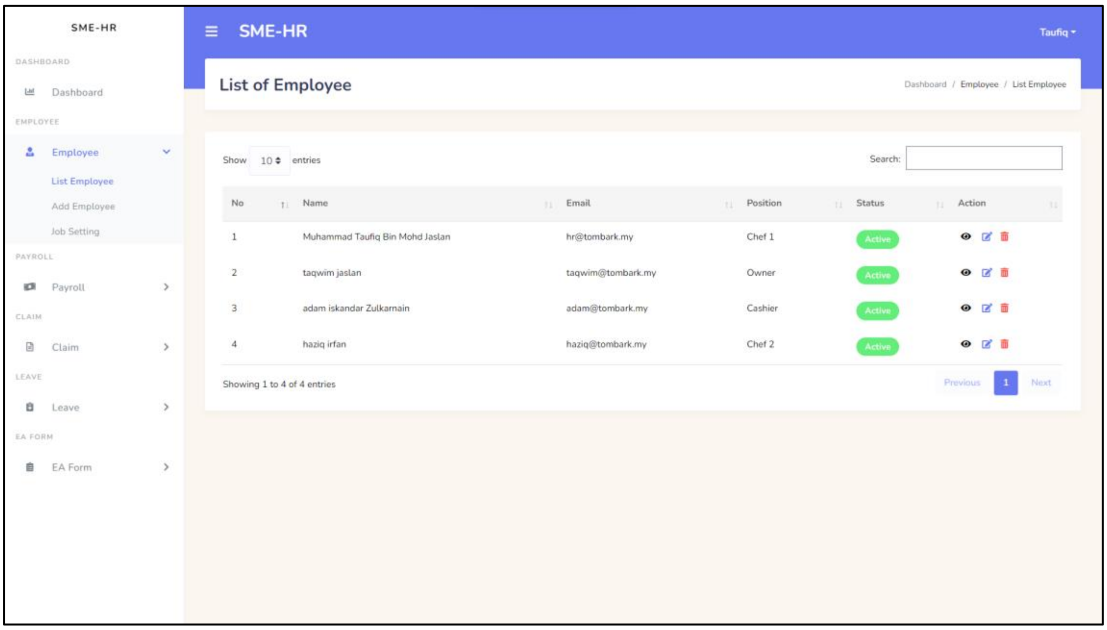
Figure 89: Employee Model Snippet Code



```
public function CreateEmployee()
{
    return view('ManageEmployee.AddEmployee'); //link to go to addEmployee page
}

/**
 * Store a newly created resource in storage.
 */
public function RegisterEmployee(Request $request)
{
    //store a new user (staff)
    $newUser = EmployeeRecord::create([
        'name' => $request['name'],
        'username' => $request['username'],
        'phone_number' => $request['phone_number'],
        'email' => $request['email'],
        'password' => Hash::make($request['password']),
        'address' => $request['address'],
        'gender' => $request['gender'],
        'start_date' => $request['start_date'],
        'end_date' => $request['end_date'],
        'position_id' => $request['position_id'],
        'user_type_id' => $request['user_type_id'],
        'status' => 1,
    ]);
```

Figure 90: Employee Controller (Register Function) Snippet Code

Figure 89 depicts a code snippet related to the employee model, which outlines the attributes contained within the "users" table in the database. This employee model code defines several crucial attributes, including name, username, phone number, email, password, start date, position id, and status. The model serves as a blueprint for the database table, defining the fields and properties the system will utilize to store employee information. By specifying these attributes, the system can efficiently store and manage employee data, providing easy access and maintenance of records. Moreover, Figure 90 illustrates the code relevant to the employee controller, which outlines the methods used to manage data input from the user interface and store it in the database. The controller code defines separate functions for each system functionality, with each function responsible for a specific task, such as adding an employee. For example, the "RegisterEmployee" function stores the employee information entered by the user into the database. These functions allow for streamlined and organized employee data management, ensuring efficient and effective system operation.

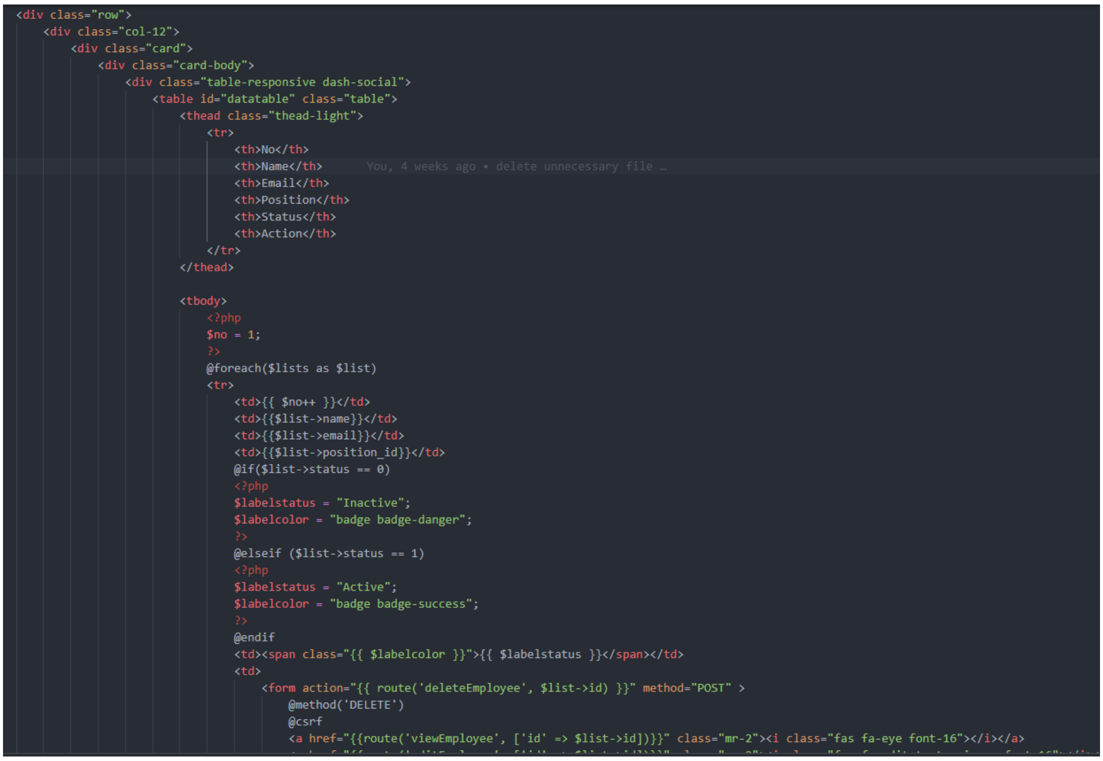


List of Employee

No	Name	Email	Position	Status	Action
1	Muhammad Taufiq Bin Mohd Jaslan	hr@tombark.my	Chef 1	Active	
2	taqwim jaslan	taqwim@tombark.my	Owner	Active	
3	adam iskandar Zulkarnain	adam@tombark.my	Cashier	Active	
4	haziq irfan	haziq@tombark.my	Chef 2	Active	

Showing 1 to 4 of 4 entries

Figure 91: List of Employee Page Interface

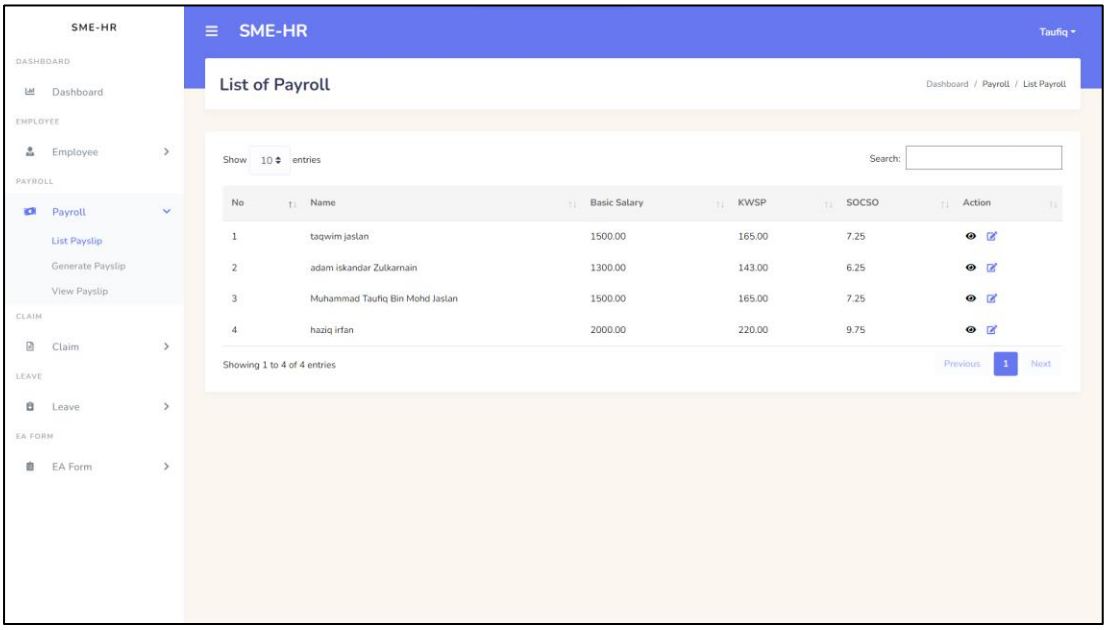


```
<div class="row">
    <div class="col-12">
        <div class="card">
            <div class="card-body">
                <div class="table-responsive dash-social">
                    <table id="datatable" class="table">
                        <thead class="thead-light">
                            <tr>
                                <th>No</th>
                                <th>Name</th>      You, 4 weeks ago • delete unnecessary file …
                                <th>Email</th>
                                <th>Position</th>
                                <th>Status</th>
                                <th>Action</th>
                            </tr>
                        </thead>

                        <tbody>
                            <?php
                            $no = 1;
                            ?>
                            @foreach($lists as $list)
                            <tr>
                                <td>{{ $no++ }}</td>
                                <td>{{$list->name}}</td>
                                <td>{{$list->email}}</td>
                                <td>{{$list->position_id}}</td>
                                @if($list->status == 0)
                                <?php
                                $labelstatus = "Inactive";
                                $labelcolor = "badge badge-danger";
                                ?>
                                @elseif ($list->status == 1)
                                <?php
                                $labelstatus = "Active";
                                $labelcolor = "badge badge-success";
                                ?>
                                @endif
                                <td><span class="{{ $labelcolor }}">{{ $labelstatus }}</span></td>
                                <td>
                                    <form action="{{ route('deleteEmployee', $list->id) }}" method="POST" >
                                        @method('DELETE')
                                        @csrf
                                        <a href="{{route('viewEmployee', ['id' => $list->id])}}" class="mr-2"><i class="fas fa-eye font-16"></i></a>
```

Figure 92: Employee View (List of Employee) Snippet Code

Figure 91 displays the user interface for the list of employee pages, which presents the employee records in tabular form. The table contains a comprehensive list of employee information, including name, email, position, and status, allowing the user to quickly access and retrieve the required data. By presenting this information in a structured and organized format, the list of employee pages enables the system users to review, analyze, and manage employee records efficiently. Besides that, Figure 92 the code snippet for the "List of Employee" page, which defines the critical fields required to display the employee records and the variables that need to be called to retrieve data from the database. The code outlines the necessary elements for presenting the employee records in tabular form, including the name, email, position, and status fields. By utilizing this code, the system can efficiently retrieve and display the required employee data, ensuring optimal system performance and effective management of employee records.

4.2.3 Development of Payroll Module



List of Payroll

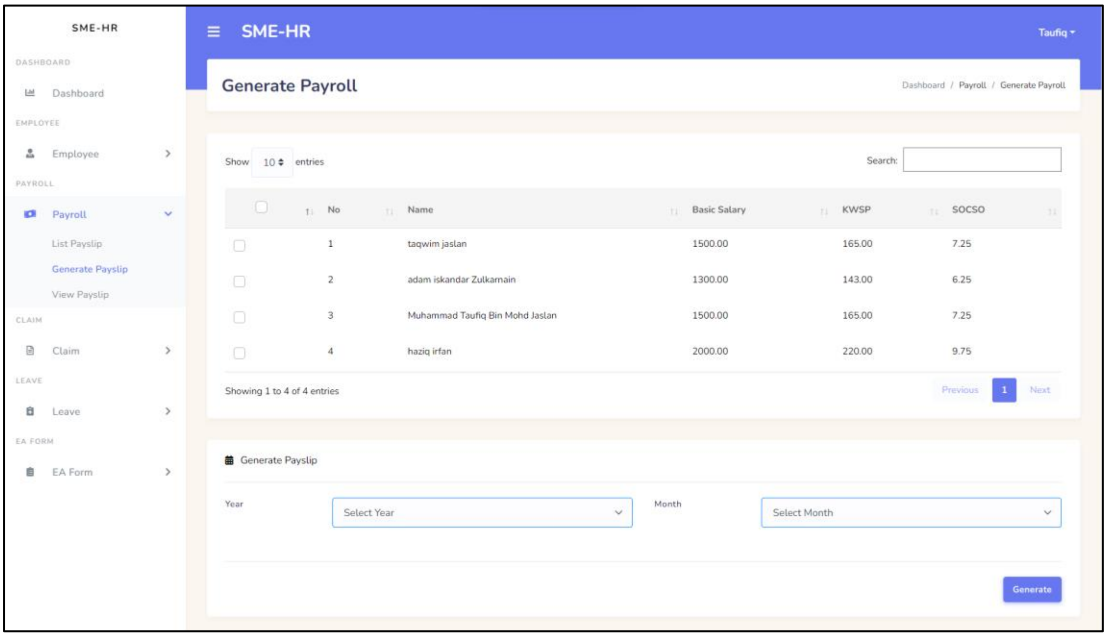
Show 10 entries

Search:

No	Name	Basic Salary	KWSP	SOCSO	Action
1	taqwim jaslan	1500.00	165.00	7.25	
2	adam iskandar Zulkarnain	1300.00	143.00	6.25	
3	Muhammad Taufiq Bin Mohd Jaslan	1500.00	165.00	7.25	
4	haziq irfan	2000.00	220.00	9.75	

Showing 1 to 4 of 4 entries

Figure 93: List of Payroll Page Interface



Generate Payroll

Show 10 entries

Search:

□	No	Name	Basic Salary	KWSP	SOCSO
□	1	taqwim jaslan	1500.00	165.00	7.25
□	2	adam iskandar Zulkarnain	1300.00	143.00	6.25
□	3	Muhammad Taufiq Bin Mohd Jaslan	1500.00	165.00	7.25
□	4	haziq irfan	2000.00	220.00	9.75

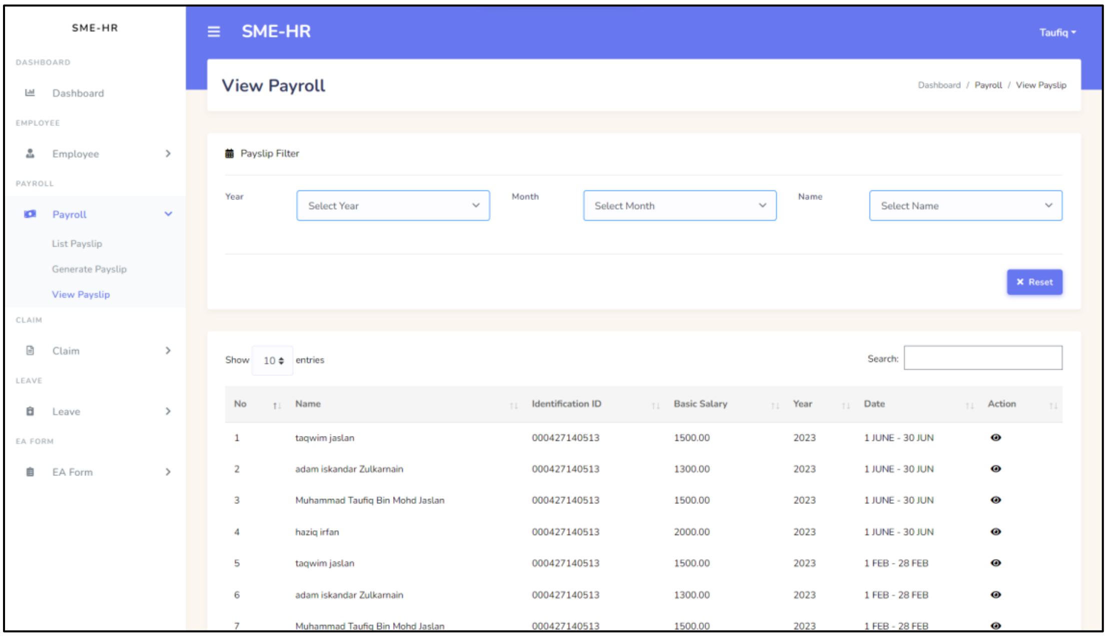
Showing 1 to 4 of 4 entries

Generate Payslip

Year: Select Year

Month: Select Month

Figure 94: Generate Payroll Page Interface



View Payroll

Payslip Filter

Year: Select Year
Month: Select Month
Name: Select Name

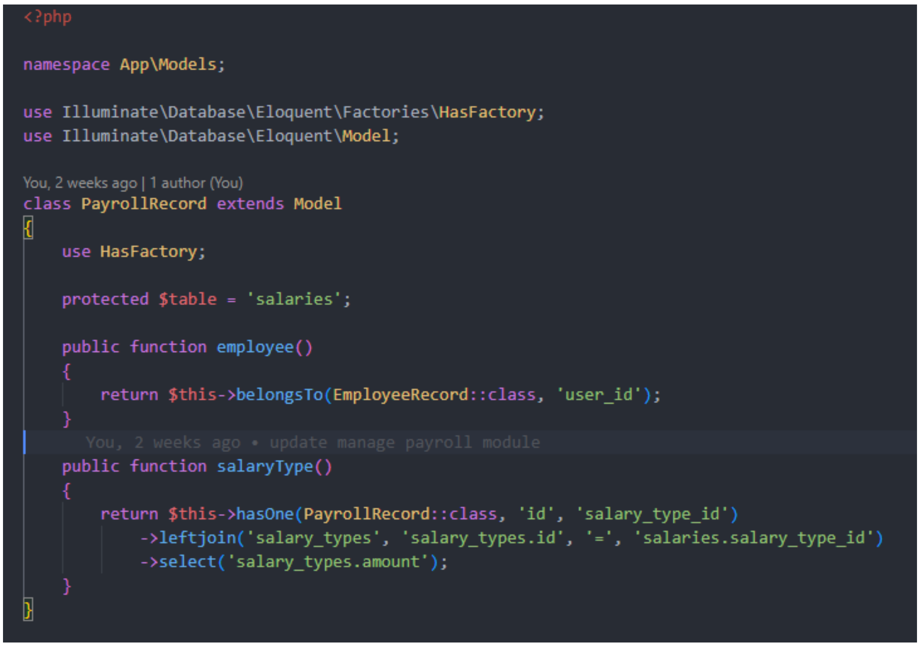
Show 10 entries

Search:

No	Name	Identification ID	Basic Salary	Year	Date	Action
1	taqwim jaslan	000427140513	1500.00	2023	1 JUNE - 30 JUN	👁
2	adam iskandar Zulkarnain	000427140513	1300.00	2023	1 JUNE - 30 JUN	👁
3	Muhammad Taufiq Bin Mohd Jaslan	000427140513	1500.00	2023	1 JUNE - 30 JUN	👁
4	haziq irfan	000427140513	2000.00	2023	1 JUNE - 30 JUN	👁
5	taqwim jaslan	000427140513	1500.00	2023	1 FEB - 28 FEB	👁
6	adam iskandar Zulkarnain	000427140513	1300.00	2023	1 FEB - 28 FEB	👁
7	Muhammad Taufiq Bin Mohd Jaslan	000427140513	1500.00	2023	1 FEB - 28 FEB	👁

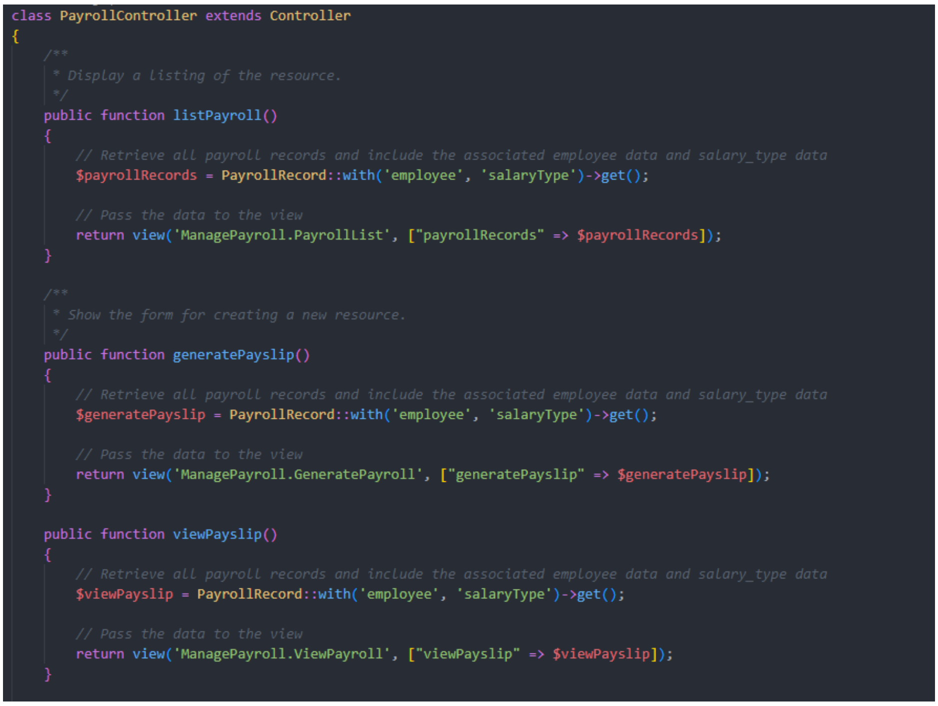
Figure 95: View Payroll Page Interface

Figure 93 displays the interface for the "List of Payroll" page. This page provides a table-form list of payroll information users can access. The table contains detailed information such as the employee's name, position, basic salary, KWSP, and SOCSO. Besides that, Figure 94 displays the interface for the "Generate Payroll" page. The HR Administrator can generate the payslip for all employees in the company through this page. They can input the specific year and month of the payroll they want to generate. Once the HR Administrator generates the payroll, the system will automatically calculate and display the employee's salary based on their basic salary and any applicable deductions. Moreover, Figure 95 displays the interface for the "View Payroll" page. On this page, the user can view the generated payslip by selecting the year and month of the payslip they want to view. The payslip will then be displayed in a table format, containing detailed information such as the employee's name, position, basic salary, KWSP, and SOCSO for the selected period.



```
<?php

namespace App\Models;

use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Database\Eloquent\Model;

You, 2 weeks ago | 1 author (You)
class PayrollRecord extends Model
{
    use HasFactory;

    protected $table = 'salaries';

    public function employee()
    {
        return $this->belongsTo(EmployeeRecord::class, 'user_id');
    }
      You, 2 weeks ago • update manage payroll module
    public function salaryType()
    {
        return $this->hasOne(PayrollRecord::class, 'id', 'salary_type_id')
            ->leftjoin('salary_types', 'salary_types.id', '=', 'salaries.salary_type_id')
            ->select('salary_types.amount');
    }
}
```

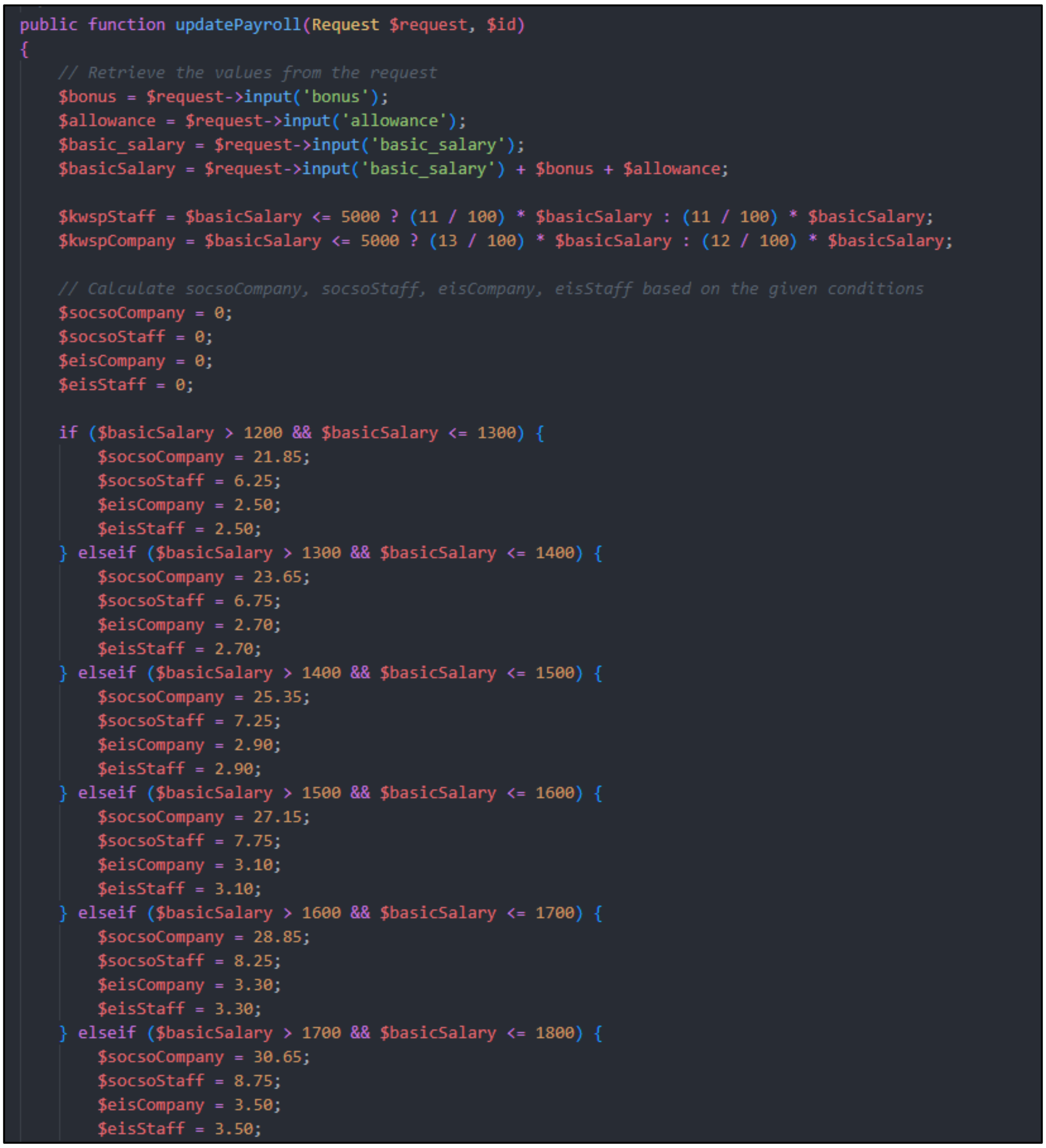
Figure 96: Payroll Model Snippet Code



```
class PayrollController extends Controller
{
    /**
     * Display a listing of the resource.
     */
    public function listPayroll()
    {
        // Retrieve all payroll records and include the associated employee data and salary_type data
        $payrollRecords = PayrollRecord::with('employee', 'salaryType')->get();

        // Pass the data to the view
        return view('ManagePayroll.PayrollList', ["payrollRecords" => $payrollRecords]);
    }

    /**
     * Show the form for creating a new resource.
     */
    public function generatePayslip()
    {
        // Retrieve all payroll records and include the associated employee data and salary_type data
        $generatePayslip = PayrollRecord::with('employee', 'salaryType')->get();

        // Pass the data to the view
        return view('ManagePayroll.GeneratePayroll', ["generatePayslip" => $generatePayslip]);
    }

    public function viewPayslip()
    {
        // Retrieve all payroll records and include the associated employee data and salary_type data
        $viewPayslip = PayrollRecord::with('employee', 'salaryType')->get();

        // Pass the data to the view
        return view('ManagePayroll.ViewPayroll', ["viewPayslip" => $viewPayslip]);
    }
```

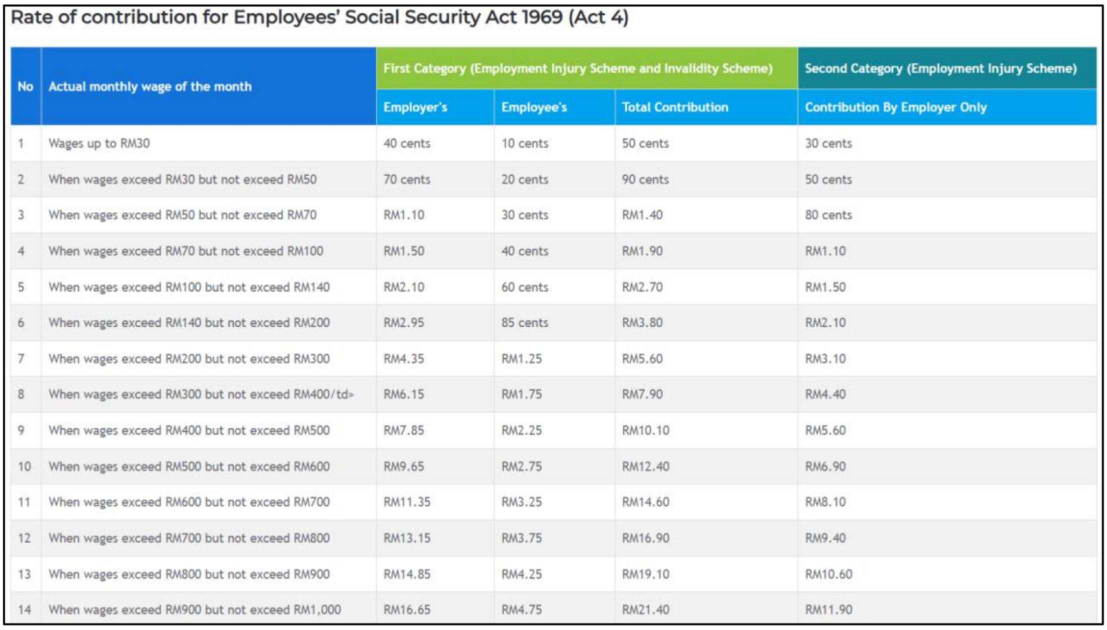
Figure 97: Payroll Controller Snippet Code

Figure 96 presents a code snippet that pertains to the payroll model, outlining the attributes contained in the "salaries" table in the database. This code specifies the essential attributes required to store payroll information, such as basic salary, KWSP, SOCSO, and zakat. By specifying these attributes, the model serves as a blueprint for the database table, enabling the system to store and manage employee data effectively and ensuring easy access and maintenance of records. Similarly, Figure 97 illustrates the code relevant to the payroll controller, specifying the methods utilized to manage data input from the user interface and store it in the database. The controller code defines separate functions for each system functionality, with each function assigned to a specific task, such as adding payroll. For instance, the "listPayroll" function is a function that retrieves all the payroll data from the database.



```php
public function updatePayroll(Request $request, $id)
{
    // Retrieve the values from the request
    $bonus = $request->input('bonus');
    $allowance = $request->input('allowance');
    $basic_salary = $request->input('basic_salary');
    $basicSalary = $request->input('basic_salary') + $bonus + $allowance;

    $kwspStaff = $basicSalary <= 5000 ? (11 / 100) * $basicSalary : (11 / 100) * $basicSalary;
    $kwspCompany = $basicSalary <= 5000 ? (13 / 100) * $basicSalary : (12 / 100) * $basicSalary;

    // Calculate socsoCompany, socsoStaff, eisCompany, eisStaff based on the given conditions
    $socsoCompany = 0;
    $socsoStaff = 0;
    $eisCompany = 0;
    $eisStaff = 0;

    if ($basicSalary > 1200 && $basicSalary <= 1300) {
        $socsoCompany = 21.85;
        $socsoStaff = 6.25;
        $eisCompany = 2.50;
        $eisStaff = 2.50;
    } elseif ($basicSalary > 1300 && $basicSalary <= 1400) {
        $socsoCompany = 23.65;
        $socsoStaff = 6.75;
        $eisCompany = 2.70;
        $eisStaff = 2.70;
    } elseif ($basicSalary > 1400 && $basicSalary <= 1500) {
        $socsoCompany = 25.35;
        $socsoStaff = 7.25;
        $eisCompany = 2.90;
        $eisStaff = 2.90;
    } elseif ($basicSalary > 1500 && $basicSalary <= 1600) {
        $socsoCompany = 27.15;
        $socsoStaff = 7.75;
        $eisCompany = 3.10;
        $eisStaff = 3.10;
    } elseif ($basicSalary > 1600 && $basicSalary <= 1700) {
        $socsoCompany = 28.85;
        $socsoStaff = 8.25;
        $eisCompany = 3.30;
        $eisStaff = 3.30;
    } elseif ($basicSalary > 1700 && $basicSalary <= 1800) {
        $socsoCompany = 30.65;
        $socsoStaff = 8.75;
        $eisCompany = 3.50;
        $eisStaff = 3.50;
```

Figure 98: Payroll Calculation Controller Snippet Code



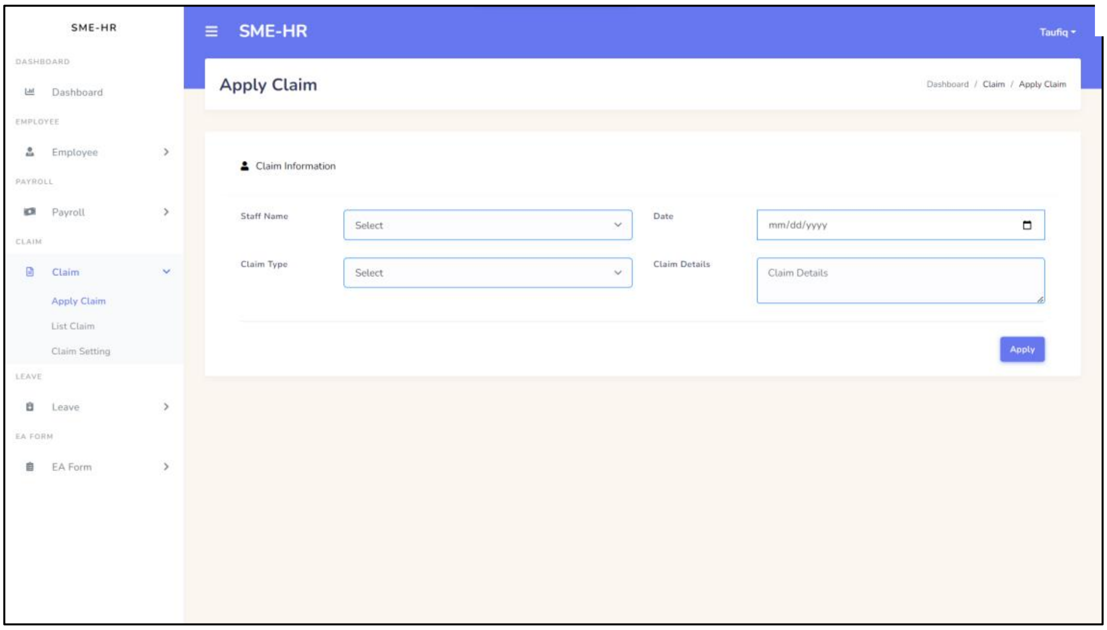
Rate of contribution for Employees' Social Security Act 1969 (Act 4)

No	Actual monthly wage of the month	First Category (Employment Injury Scheme and Invalidity Scheme)			Second Category (Employment Injury Scheme)
		Employer's	Employee's	Total Contribution	Contribution By Employer Only
1	Wages up to RM30	40 cents	10 cents	50 cents	30 cents
2	When wages exceed RM30 but not exceed RM50	70 cents	20 cents	90 cents	50 cents
3	When wages exceed RM50 but not exceed RM70	RM1.10	30 cents	RM1.40	80 cents
4	When wages exceed RM70 but not exceed RM100	RM1.50	40 cents	RM1.90	RM1.10
5	When wages exceed RM100 but not exceed RM140	RM2.10	60 cents	RM2.70	RM1.50
6	When wages exceed RM140 but not exceed RM200	RM2.95	85 cents	RM3.80	RM2.10
7	When wages exceed RM200 but not exceed RM300	RM4.35	RM1.25	RM5.60	RM3.10
8	When wages exceed RM300 but not exceed RM400/td>	RM6.15	RM1.75	RM7.90	RM4.40
9	When wages exceed RM400 but not exceed RM500	RM7.85	RM2.25	RM10.10	RM5.60
10	When wages exceed RM500 but not exceed RM600	RM9.65	RM2.75	RM12.40	RM6.90
11	When wages exceed RM600 but not exceed RM700	RM11.35	RM3.25	RM14.60	RM8.10
12	When wages exceed RM700 but not exceed RM800	RM13.15	RM3.75	RM16.90	RM9.40
13	When wages exceed RM800 but not exceed RM900	RM14.85	RM4.25	RM19.10	RM10.60
14	When wages exceed RM900 but not exceed RM1,000	RM16.65	RM4.75	RM21.40	RM11.90

Figure 99: Rate of Contribution for Employees' Social Security Act 1969 (Act 4)

Figure 98 represents the payroll calculation controller snippet code in the payroll module, which automates the calculation of KWSP, Socso, and Employment Insurance System (EIS) contributions based on the administrator's insertion of the staff's basic salary. The logic formula and algorithm in the code follow the Contribution Amount for Act 4 and Act 800 following the Increment of the Wage Ceiling Limit displayed in Figure 99 (*Rate of Contribution*, n.d.). This automated calculation significantly simplifies the administrator's payroll management task by accurately determining KWSP, Socso, and EIS contributions, streamlining the overall process.

4.2.4 Development of Claim Module



Apply Claim

Claim Information

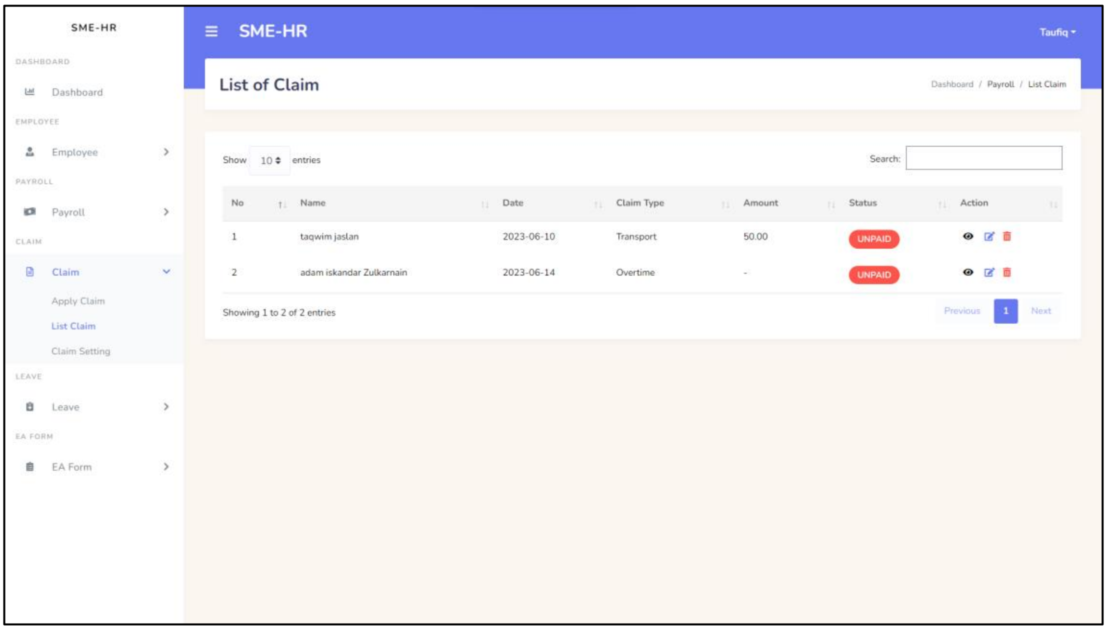
Staff Name: Select

Date: mm/dd/yyyy

Claim Type: Select

Claim Details: Claim Details

Figure 100: Apply Claim Page Interface



List of Claim

Show 10 entries

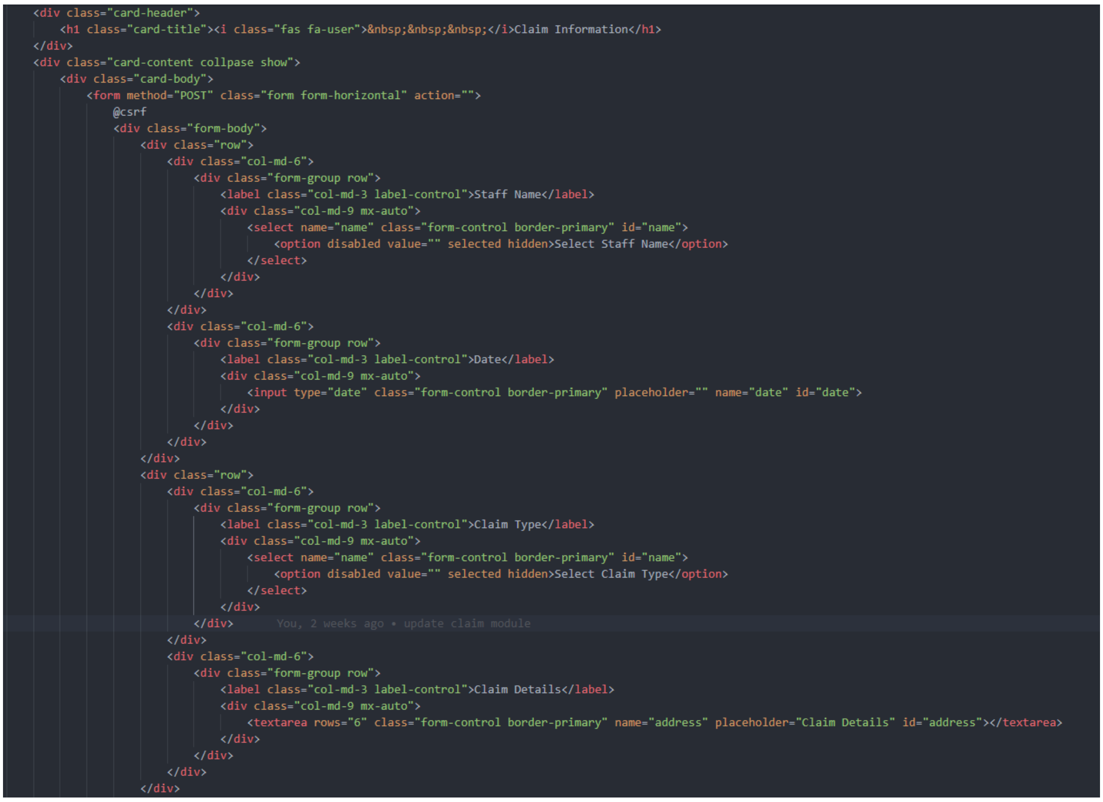
Search:

No	Name	Date	Claim Type	Amount	Status	Action
1	taqwim jaslan	2023-06-10	Transport	50.00	UNPAID	
2	adam iskandar Zulkarnain	2023-06-14	Overtime	-	UNPAID	

Showing 1 to 2 of 2 entries

Figure 101: List of Claim Page Interface

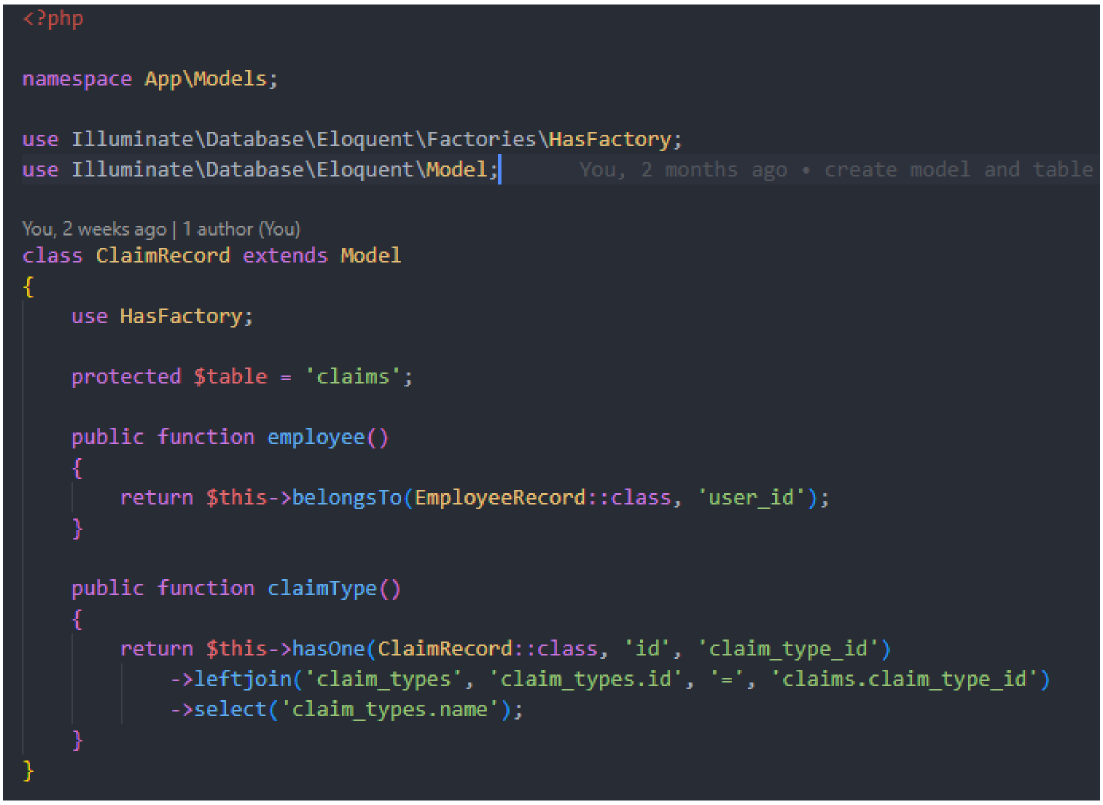
Figure 100 displays the interface for the Apply Claim page, where the user can apply for leave by providing the necessary information. The required information includes the date of the claim, claim type, claim details, attachment file, and amount of the claim. The user can apply for their claim efficiently by filling in this information. Moreover, Figure 101 shows the interface for the List of Claims page. This page displays a table form containing a list of all the claims that have been applied. The table includes all relevant information, such as the date, claim type, claim details, and the status of the claim. Users can quickly determine which claims are pending, approved, or rejected by displaying the claim status in the table.



```
<div class="card-header">
    <h1 class="card-title"><i class="fas fa-user">&nbsp;&nbsp;&nbsp;</i>Claim Information</h1>
</div>
<div class="card-content collpase show">
    <div class="card-body">
        <form method="POST" class="form form-horizontal" action="">
            @csrf
            <div class="form-body">
                <div class="row">
                    <div class="col-md-6">
                        <div class="form-group row">
                            <label class="col-md-3 label-control">Staff Name</label>
                            <div class="col-md-9 mx-auto">
                                <select name="name" class="form-control border-primary" id="name">
                                    <option disabled value="" selected hidden>Select Staff Name</option>
                                </select>
                            </div>
                        </div>
                    </div>
                    <div class="col-md-6">
                        <div class="form-group row">
                            <label class="col-md-3 label-control">Date</label>
                            <div class="col-md-9 mx-auto">
                                <input type="date" class="form-control border-primary" placeholder="" name="date" id="date">
                            </div>
                        </div>
                    </div>
                </div>
                <div class="row">
                    <div class="col-md-6">
                        <div class="form-group row">
                            <label class="col-md-3 label-control">Claim Type</label>
                            <div class="col-md-9 mx-auto">
                                <select name="name" class="form-control border-primary" id="name">
                                    <option disabled value="" selected hidden>Select Claim Type</option>
                                </select>
                            </div>
                        </div>      You, 2 weeks ago • update claim module
                    </div>
                    <div class="col-md-6">
                        <div class="form-group row">
                            <label class="col-md-3 label-control">Claim Details</label>
                            <div class="col-md-9 mx-auto">
                                <textarea rows="6" class="form-control border-primary" name="address" placeholder="Claim Details" id="address"></textarea>
                            </div>
                        </div>
                    </div>
                </div>
```

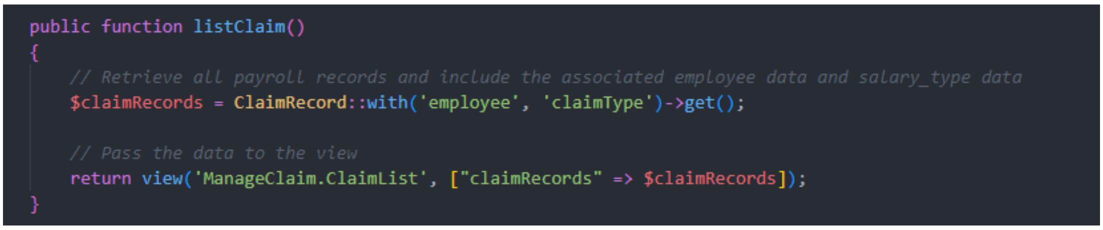
Figure 102: Claim View (Apply Claim) Snippet Code

The code snippet depicted in Figure 102 presents the code structure for the "Apply Claim" page, which facilitates the application of claims by the user. The code defines necessary fields that are mandatory for applying, including the staff name, claim date, claim type, claim details, attachment file, and claim amount. The form has several components, including text input, date input, file input, and text area, ensuring the employee's data is complete and accurate. Implementing these components enhances the user experience and simplifies the data input process.



```php
<?php

namespace App\Models;

use Illuminate\Database\Eloquent\Factories\HasFactory;
use Illuminate\Database\Eloquent\Model;        You, 2 months ago • create model and table

You, 2 weeks ago | 1 author (You)
class ClaimRecord extends Model
{
    use HasFactory;

    protected $table = 'claims';

    public function employee()
    {
        return $this->belongsTo(EmployeeRecord::class, 'user_id');
    }

    public function claimType()
    {
        return $this->hasOne(ClaimRecord::class, 'id', 'claim_type_id')
            ->leftjoin('claim_types', 'claim_types.id', '=', 'claims.claim_type_id')
            ->select('claim_types.name');
    }
}
```

Figure 103: Claim Model Snippet Code



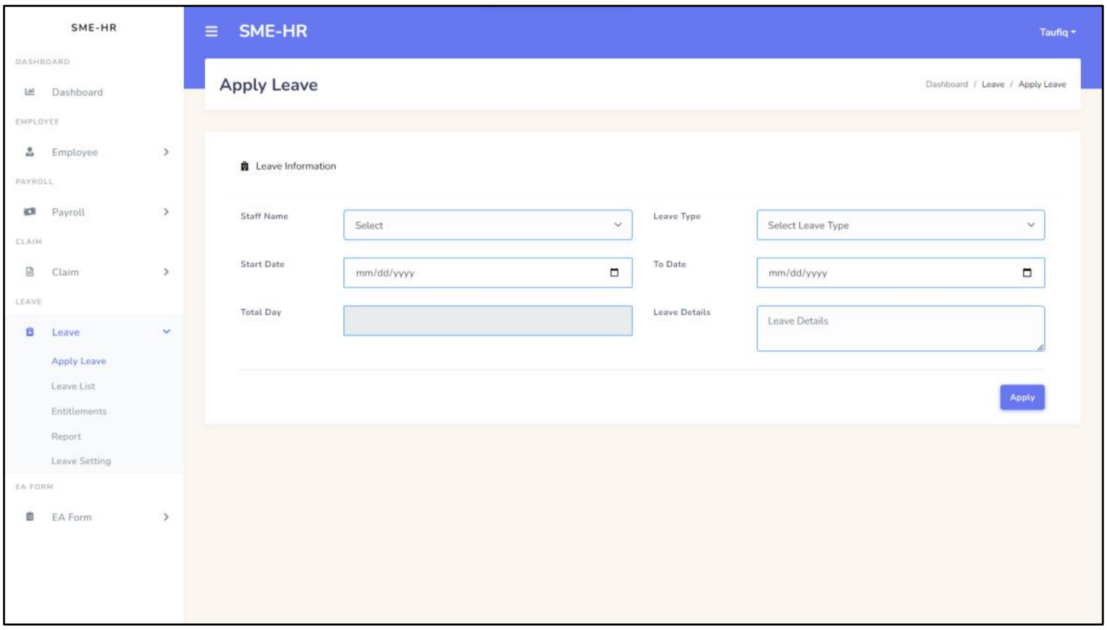
```php
public function listClaim()
{
    // Retrieve all payroll records and include the associated employee data and salary_type data
    $claimRecords = ClaimRecord::with('employee', 'claimType')->get();

    // Pass the data to the view
    return view('ManageClaim.ClaimList', ["claimRecords" => $claimRecords]);
}
```

Figure 104: Claim Controller Snippet Code

Figure 103 presents a code snippet related to the "Claim" model, which outlines the attributes contained within the "claims" table in the database. The claim model code defines several crucial attributes, including claims type and date. Additionally, the "claimType" function contains code that joins two tables, namely the "claims" and "claim types" tables.

Furthermore, Figure 104 illustrates the code relevant to the "Claim" controller, which outlines the methods used to manage data input from the user interface and store it in the database. The controller code defines separate functions for each system functionality, with each function responsible for a specific task, such as displaying all claim data from the database. For example, the "ListClaim" function retrieves all claim data from the database and displays it on the list of claim pages.

4.2.5 Development of Leave Module



Apply Leave

Leave Information

Staff Name: Select

Leave Type: Select Leave Type

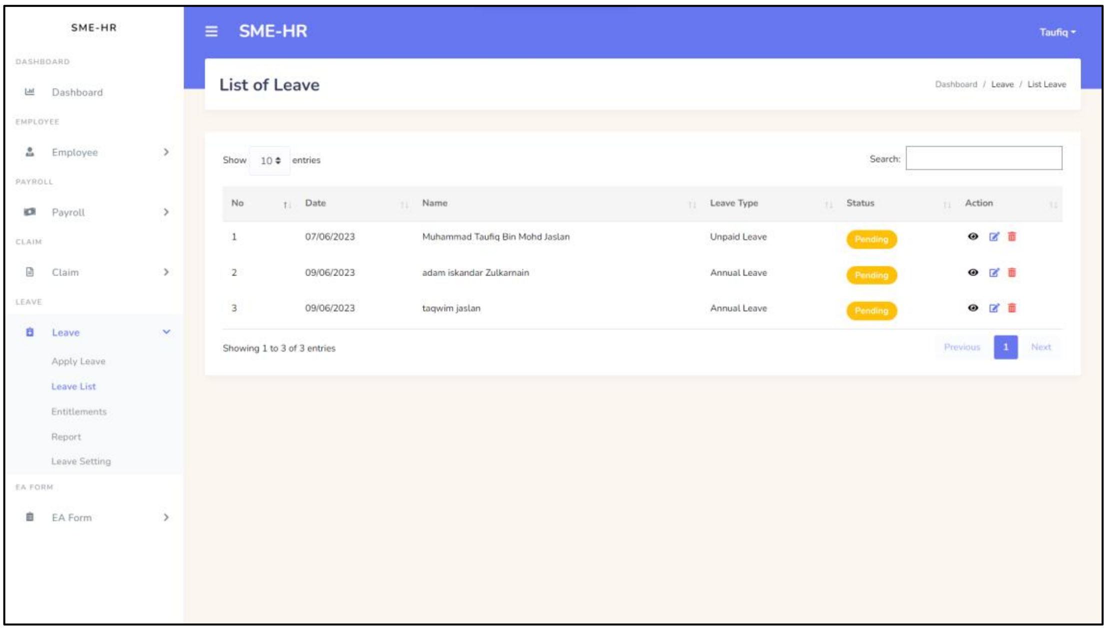
Start Date: mm/dd/yyyy

To Date: mm/dd/yyyy

Total Day:

Leave Details: Leave Details

Figure 105: Apply Leave Page Interface



List of Leave

Show 10 entries

Search:

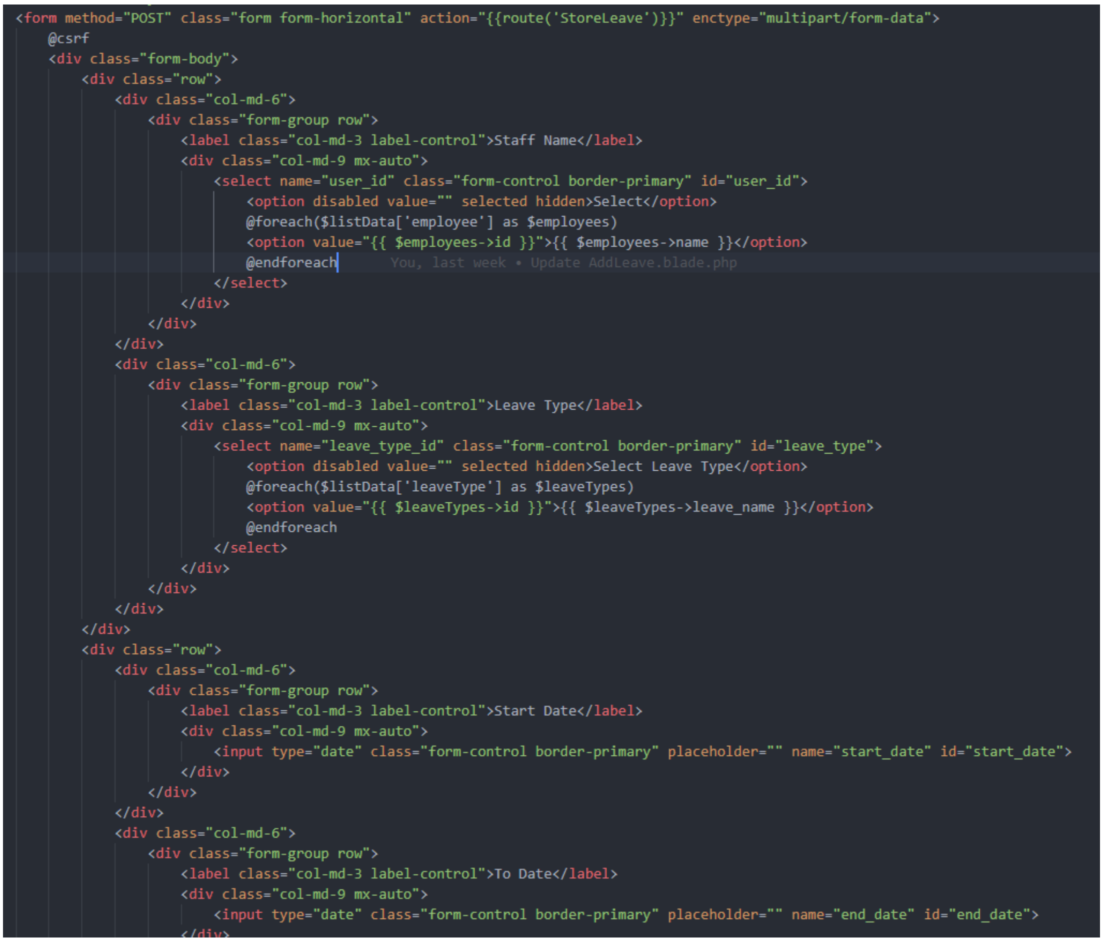
No	Date	Name	Leave Type	Status	Action
1	07/06/2023	Muhammad Taufiq Bin Mohd Jaslan	Unpaid Leave	Pending	
2	09/06/2023	adam iskandar Zulkarnain	Annual Leave	Pending	
3	09/06/2023	taqwim jaslan	Annual Leave	Pending	

Showing 1 to 3 of 3 entries

Figure 106: List of Leave Page Interface

Figure 105 illustrates the interface for the "Apply Leave" page, which is accessible to both the HR Administrator and other staff members. This page allows users to apply for leave by filling in the required information. The interface would include fields such as leave type, start date, end date, reason for leave, and possibly additional details related to the leave application. All of the information on this page is mandatory and must be filled in by the user before submitting the application. This ensures that all necessary details are provided for the leave request.

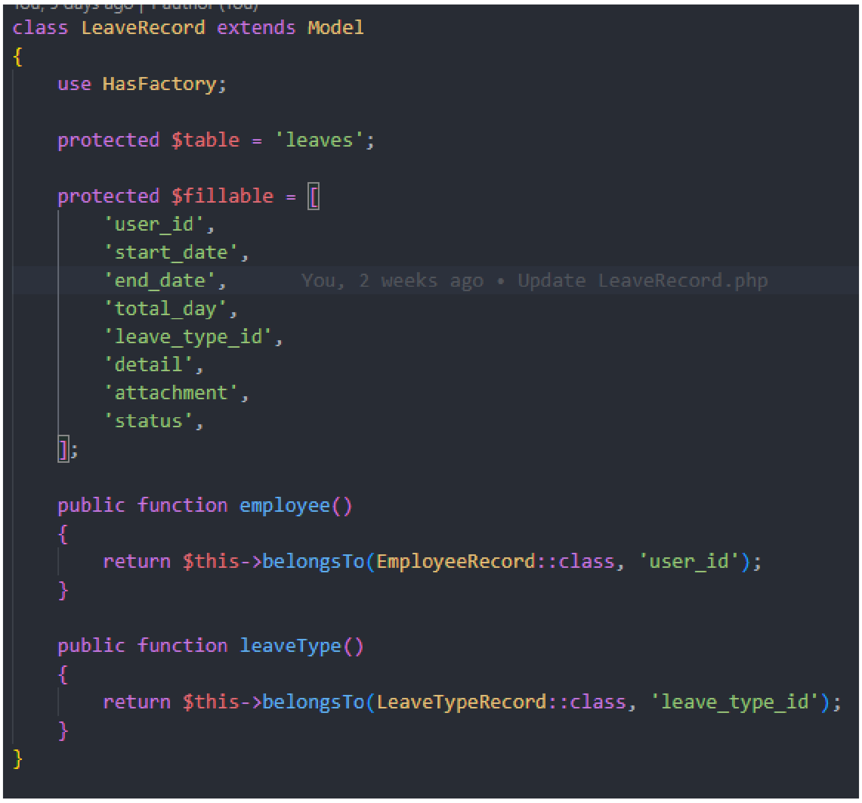
On the other hand, Figure 106 displays the interface for the "List of Leave" page. This page presents a list of all the applied leave applications and includes information about the status of each leave request. The list would typically include details such as employee names, leave types, start dates, end dates, and the current status of each application. The status field informs users whether the leave has been approved, is pending approval, or has been rejected. This allows users, including the HR Administrator, to easily track and manage leave applications and view the status of each request.



```
<form method="POST" class="form form-horizontal" action="{{route('StoreLeave')}}" enctype="multipart/form-data">
    @csrf
    <div class="form-body">
        <div class="row">
            <div class="col-md-6">
                <div class="form-group row">
                    <label class="col-md-3 label-control">Staff Name</label>
                    <div class="col-md-9 mx-auto">
                        <select name="user_id" class="form-control border-primary" id="user_id">
                            <option disabled value="" selected hidden>Select</option>
                            @foreach($listData['employee'] as $employees)
                            <option value="{{ $employees->id }}">{{ $employees->name }}</option>
                            @endforeach      You, last week • Update AddLeave.blade.php
                        </select>
                    </div>
                </div>
            </div>
            <div class="col-md-6">
                <div class="form-group row">
                    <label class="col-md-3 label-control">Leave Type</label>
                    <div class="col-md-9 mx-auto">
                        <select name="leave_type_id" class="form-control border-primary" id="leave_type">
                            <option disabled value="" selected hidden>Select Leave Type</option>
                            @foreach($listData['leaveType'] as $leaveTypes)
                            <option value="{{ $leaveTypes->id }}">{{ $leaveTypes->leave_name }}</option>
                            @endforeach
                        </select>
                    </div>
                </div>
            </div>
        </div>
        <div class="row">
            <div class="col-md-6">
                <div class="form-group row">
                    <label class="col-md-3 label-control">Start Date</label>
                    <div class="col-md-9 mx-auto">
                        <input type="date" class="form-control border-primary" placeholder="" name="start_date" id="start_date">
                    </div>
                </div>
            </div>
            <div class="col-md-6">
                <div class="form-group row">
                    <label class="col-md-3 label-control">To Date</label>
                    <div class="col-md-9 mx-auto">
                        <input type="date" class="form-control border-primary" placeholder="" name="end_date" id="end_date">
                    </div>
```

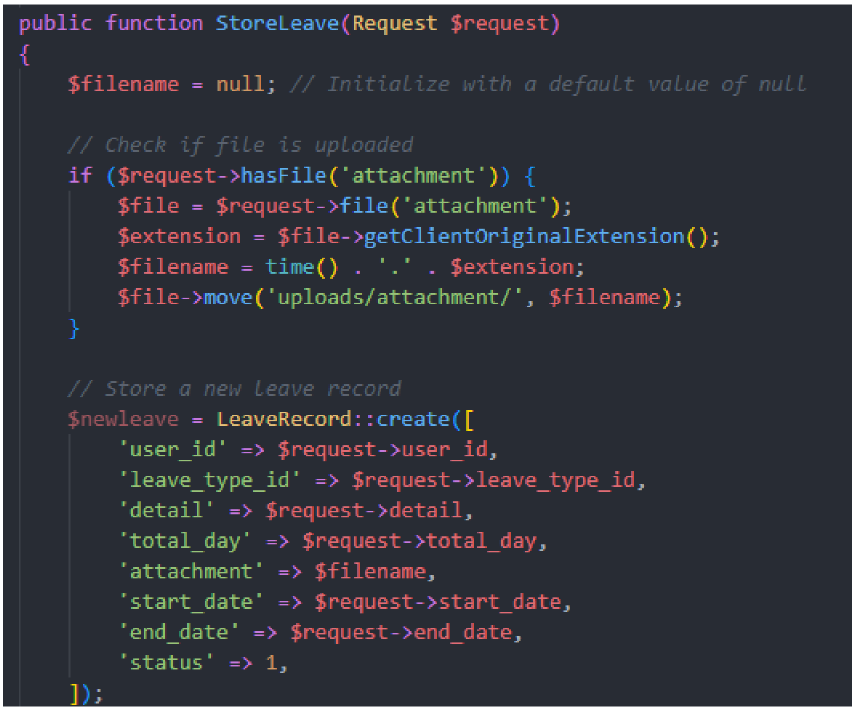
Figure 107: Leave View (Apply Leave) Snippet Code

Figure 107 illustrates a code snippet for the "Apply Leave" page, presenting the structure of a form that facilitates leave applications. The code defines several required fields, including full name, leave type, start date, end date, details, and attachment. These fields are typically implemented using various components, including text inputs, date inputs, file upload options, and text areas. The code snippet represents the HTML elements and attributes necessary to create the form structure, with accompanying JavaScript or server-side programming logic to handle form submission and validation.



```
You, 3 days ago | 1 author (You)
class LeaveRecord extends Model
{
    use HasFactory;

    protected $table = 'leaves';

    protected $fillable = [
        'user_id',
        'start_date',
        'end_date',        You, 2 weeks ago • Update LeaveRecord.php
        'total_day',
        'leave_type_id',
        'detail',
        'attachment',
        'status',
    ];

    public function employee()
    {
        return $this->belongsTo(EmployeeRecord::class, 'user_id');
    }

    public function leaveType()
    {
        return $this->belongsTo(LeaveTypeRecord::class, 'leave_type_id');
    }
}
```

Figure 108: Leave Model Snippet Code



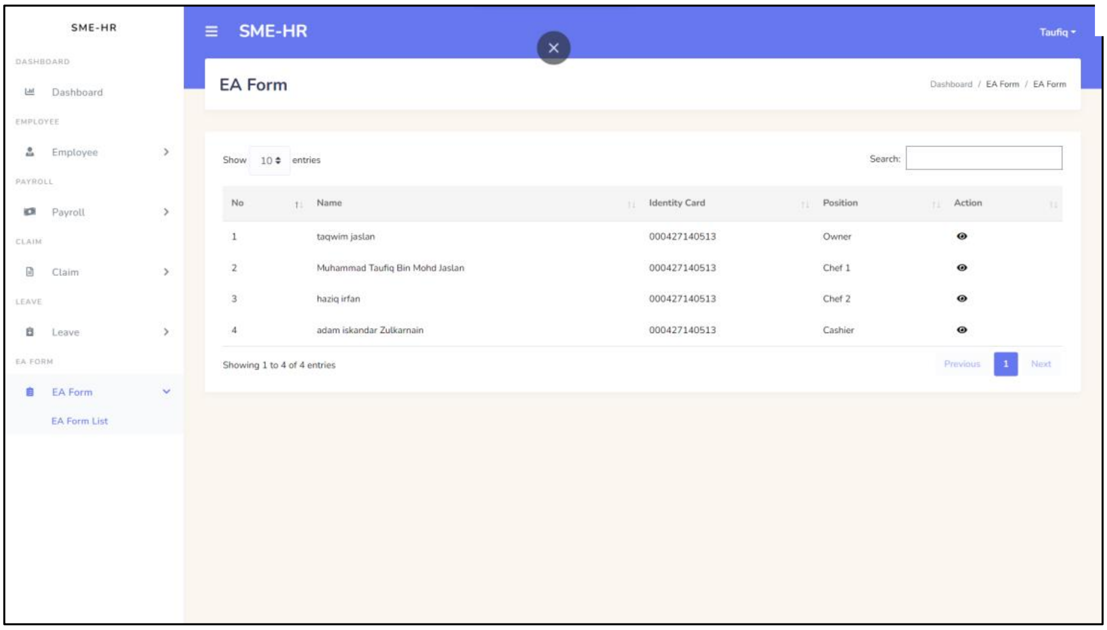
```
public function StoreLeave(Request $request)
{
    $filename = null; // Initialize with a default value of null

    // Check if file is uploaded
    if ($request->hasFile('attachment')) {
        $file = $request->file('attachment');
        $extension = $file->getClientOriginalExtension();
        $filename = time() . '.' . $extension;
        $file->move('uploads/attachment/', $filename);
    }

    // Store a new leave record
    $newleave = LeaveRecord::create([
        'user_id' => $request->user_id,
        'leave_type_id' => $request->leave_type_id,
        'detail' => $request->detail,
        'total_day' => $request->total_day,
        'attachment' => $filename,
        'start_date' => $request->start_date,
        'end_date' => $request->end_date,
        'status' => 1,
    ]);
```

Figure 109: Leave Controller Snippet Code

Figure 108 presents a code snippet relevant to the leave model, specifically about the attributes within the "leaves" table in the database. This code snippet defines the attributes in the table, including fields such as leave ID, employee ID, leave type ID, start date, end date, reason, and other pertinent information related to leave management. Additionally, the code snippet includes a "fillable" function, which specifies the attributes from the "leaves" table that can be mass-assigned during the creation or update of leave records.

Furthermore, Figure 109 illustrates the code relevant to the leave controller. This code provides methods and functions that handle the input information for leave applications and facilitate their storage in the database. For example, the "storeLeave" function is responsible for gathering and validating user input before persisting it as a new leave record in the database. The leave controller code ensures effective handling of user requests, interaction with the leave model, and seamless integration with the underlying database.

4.2.6 Development of EA Form Module



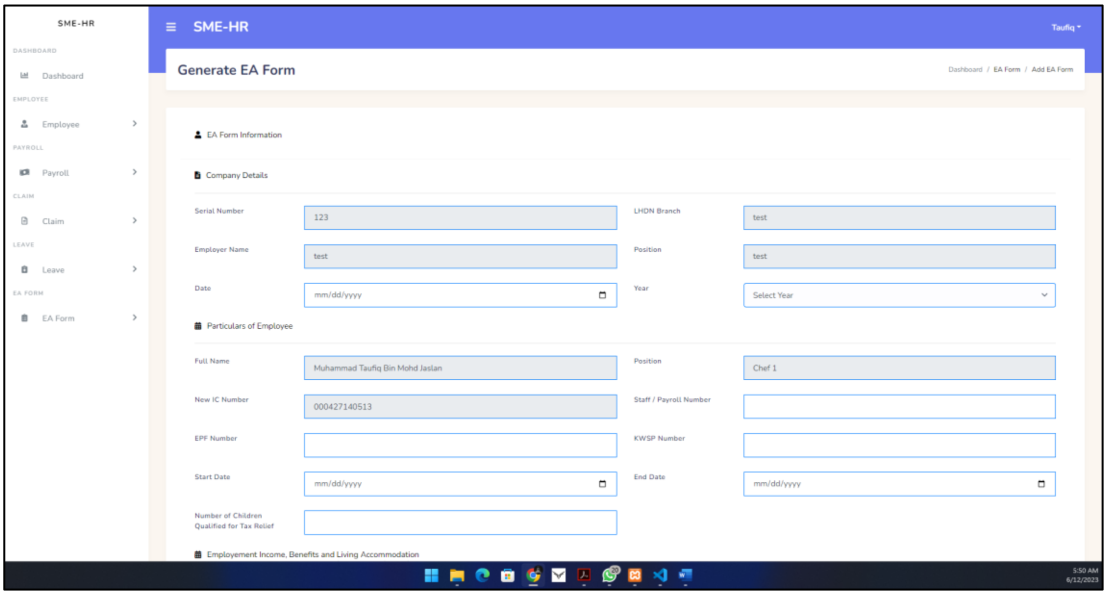
Show 10 entries

Search:

No	Name	Identity Card	Position	Action
1	taqwim jaslan	000427140513	Owner	👁
2	Muhammad Taufiq Bin Mohd Jaslan	000427140513	Chef 1	👁
3	haziq irfan	000427140513	Chef 2	👁
4	adam iskandar Zulkarnain	000427140513	Cashier	👁

Showing 1 to 4 of 4 entries

Figure 110: List of EA Form Page Interface



Generate EA Form

EA Form Information

Company Details

Serial Number	123	LHDN Branch	test
Employer Name	test	Position	test
Date	mm/dd/yyyy	Year	Select Year

Particulars of Employee

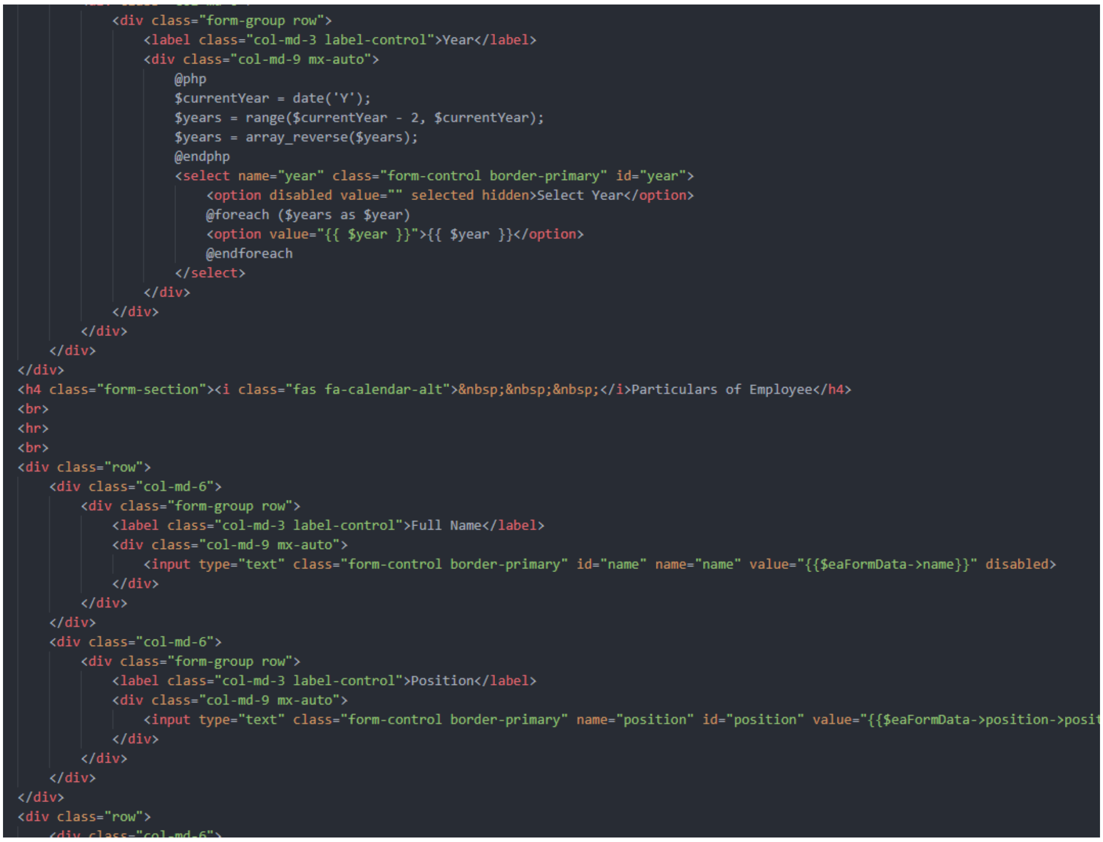
Full Name	Muhammad Taufiq Bin Mohd Jaslan	Position	Chef 1
New IC Number	000427140513	Staff / Payroll Number	
EPF Number		KWSP Number	
Start Date	mm/dd/yyyy	End Date	mm/dd/yyyy
Number of Children Qualified for Tax Relief			

Employement Income, Benefits and Living Accommodation

Figure 111: Generate EA Form Page Interface

Figure 110 showcases the interface for the "List of EA Form" page. This page is designed to display a comprehensive list of EA forms for each employee. By accessing this interface, users, particularly the HR Administrator, can easily view and manage the EA forms of all employees. The list presents pertinent details related to the EA form, providing a centralized overview of the information.

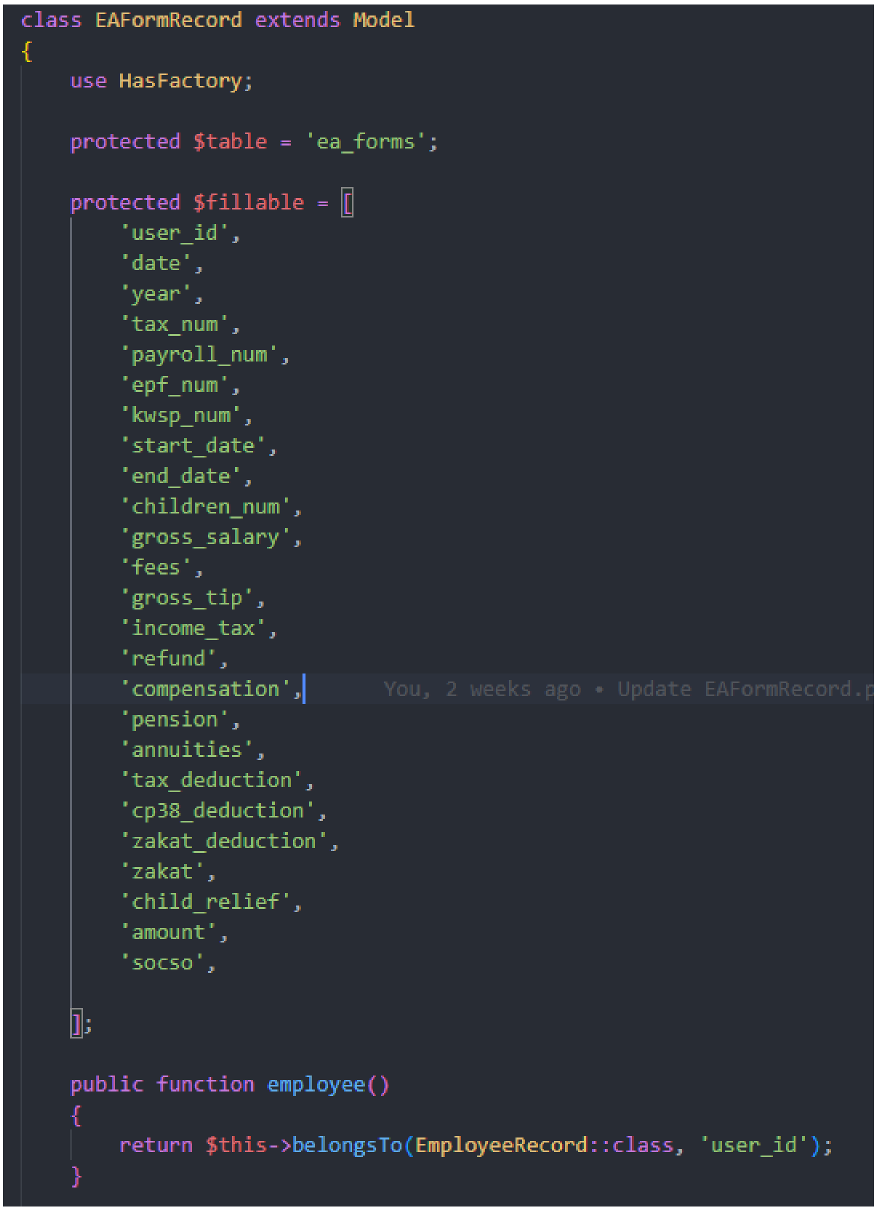
On the other hand, Figure 111 presents the interface for the "Generate EA Form" page. This page empowers the HR Administrator to input all the necessary information required for generating an EA form. The interface offers fields and options to fill in crucial details such as the KWSP number, EPF number, SOCSO number, and basic salary for each employee. The HR Administrator can accurately generate the EA form by providing this essential information, ensuring compliance with legal and regulatory requirements.



```
            <div class="form-group row">
                <label class="col-md-3 label-control">Year</label>
                <div class="col-md-9 mx-auto">
                    @php
                    $currentYear = date('Y');
                    $years = range($currentYear - 2, $currentYear);
                    $years = array_reverse($years);
                    @endphp
                    <select name="year" class="form-control border-primary" id="year">
                        <option disabled value="" selected hidden>Select Year</option>
                        @foreach ($years as $year)
                        <option value="{{ $year }}">{{ $year }}</option>
                        @endforeach
                    </select>
                </div>
            </div>
        </div>
    </div>
</div>
<h4 class="form-section"><i class="fas fa-calendar-alt">&nbsp;&nbsp;&nbsp;</i>Particulars of Employee</h4>
<br>
<hr>
<br>
<div class="row">
    <div class="col-md-6">
        <div class="form-group row">
            <label class="col-md-3 label-control">Full Name</label>
            <div class="col-md-9 mx-auto">
                <input type="text" class="form-control border-primary" id="name" name="name" value="{{$eaFormData->name}}" disabled>
            </div>
        </div>
    </div>
    <div class="col-md-6">
        <div class="form-group row">
            <label class="col-md-3 label-control">Position</label>
            <div class="col-md-9 mx-auto">
                <input type="text" class="form-control border-primary" name="position" id="position" value="{{$eaFormData->position->posit
            </div>
        </div>
    </div>
</div>
<div class="row">
    <div class="col-md-6">
```

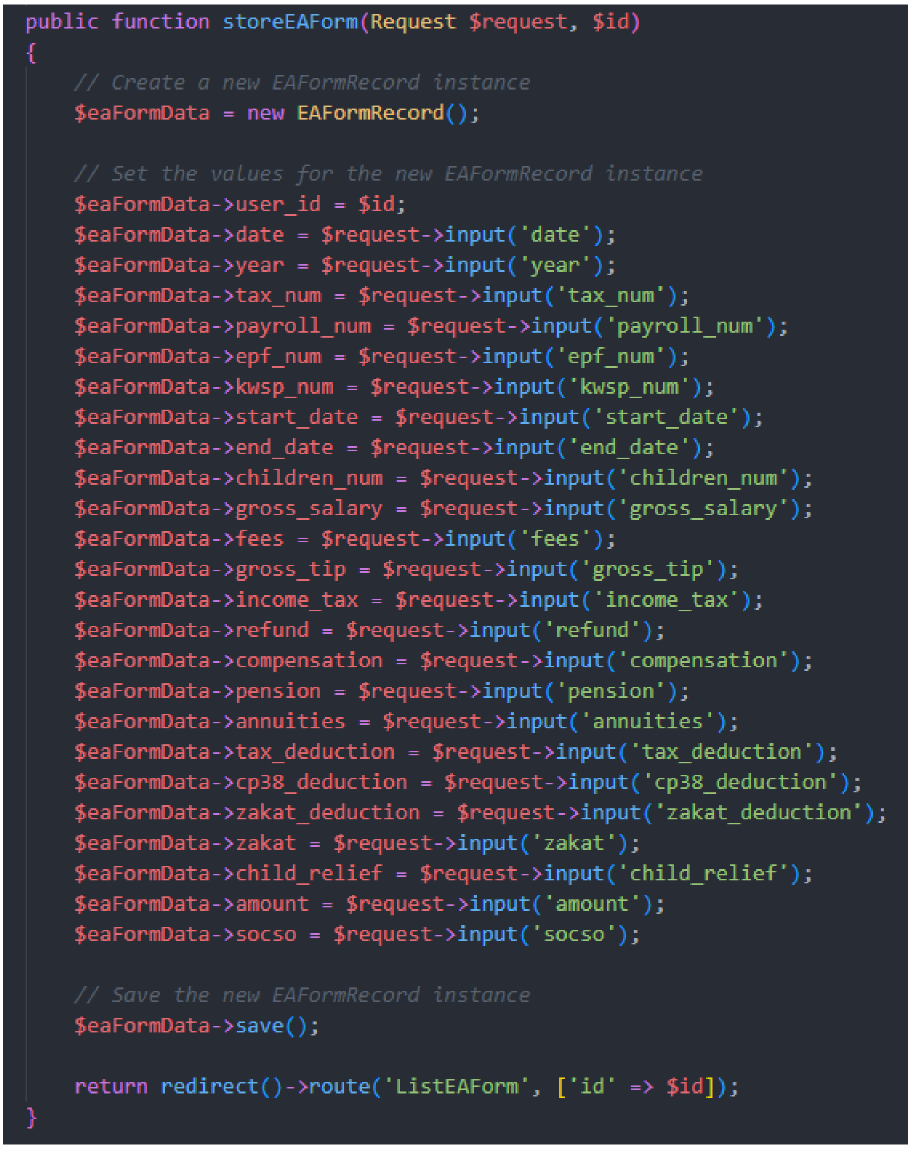
Figure 112: EA Form View (Generate EA Form) Snippet Code

Figure 112 showcases a code snippet for the "Generate EA Form" page, outlining a form structure to facilitate the generation of EA forms. The code defines essential fields such as full name, position, start date, end date, KWSP number, and SOCSO number. These fields use text inputs, date inputs, file uploads, and text areas. The code snippet enables users, particularly HR Administrators, to accurately input the required information for generating EA forms. The form's HTML elements, attributes, and JavaScript or server-side programming logic handle form submission and validation. Overall, the code snippet streamlines the process of generating EA forms by providing a user-friendly interface for capturing necessary detail.



```
class EAFormRecord extends Model
{
    use HasFactory;

    protected $table = 'ea_forms';

    protected $fillable = [
        'user_id',
        'date',
        'year',
        'tax_num',
        'payroll_num',
        'epf_num',
        'kwsp_num',
        'start_date',
        'end_date',
        'children_num',
        'gross_salary',
        'fees',
        'gross_tip',
        'income_tax',
        'refund',
        'compensation',      You, 2 weeks ago • Update EAFormRecord.p
        'pension',
        'annuities',
        'tax_deduction',
        'cp38_deduction',
        'zakat_deduction',
        'zakat',
        'child_relief',
        'amount',
        'socso',

    ];

    public function employee()
    {
        return $this->belongsTo(EmployeeRecord::class, 'user_id');
    }
```

Figure 113: EA Form Model Snippet Code



```php
public function storeEAForm(Request $request, $id)
{
    // Create a new EAFormRecord instance
    $eaFormData = new EAFormRecord();

    // Set the values for the new EAFormRecord instance
    $eaFormData->user_id = $id;
    $eaFormData->date = $request->input('date');
    $eaFormData->year = $request->input('year');
    $eaFormData->tax_num = $request->input('tax_num');
    $eaFormData->payroll_num = $request->input('payroll_num');
    $eaFormData->epf_num = $request->input('epf_num');
    $eaFormData->kwsp_num = $request->input('kwsp_num');
    $eaFormData->start_date = $request->input('start_date');
    $eaFormData->end_date = $request->input('end_date');
    $eaFormData->children_num = $request->input('children_num');
    $eaFormData->gross_salary = $request->input('gross_salary');
    $eaFormData->fees = $request->input('fees');
    $eaFormData->gross_tip = $request->input('gross_tip');
    $eaFormData->income_tax = $request->input('income_tax');
    $eaFormData->refund = $request->input('refund');
    $eaFormData->compensation = $request->input('compensation');
    $eaFormData->pension = $request->input('pension');
    $eaFormData->annuities = $request->input('annuities');
    $eaFormData->tax_deduction = $request->input('tax_deduction');
    $eaFormData->cp38_deduction = $request->input('cp38_deduction');
    $eaFormData->zakat_deduction = $request->input('zakat_deduction');
    $eaFormData->zakat = $request->input('zakat');
    $eaFormData->child_relief = $request->input('child_relief');
    $eaFormData->amount = $request->input('amount');
    $eaFormData->socso = $request->input('socso');

    // Save the new EAFormRecord instance
    $eaFormData->save();

    return redirect()->route('ListEAForm', ['id' => $id]);
}
```

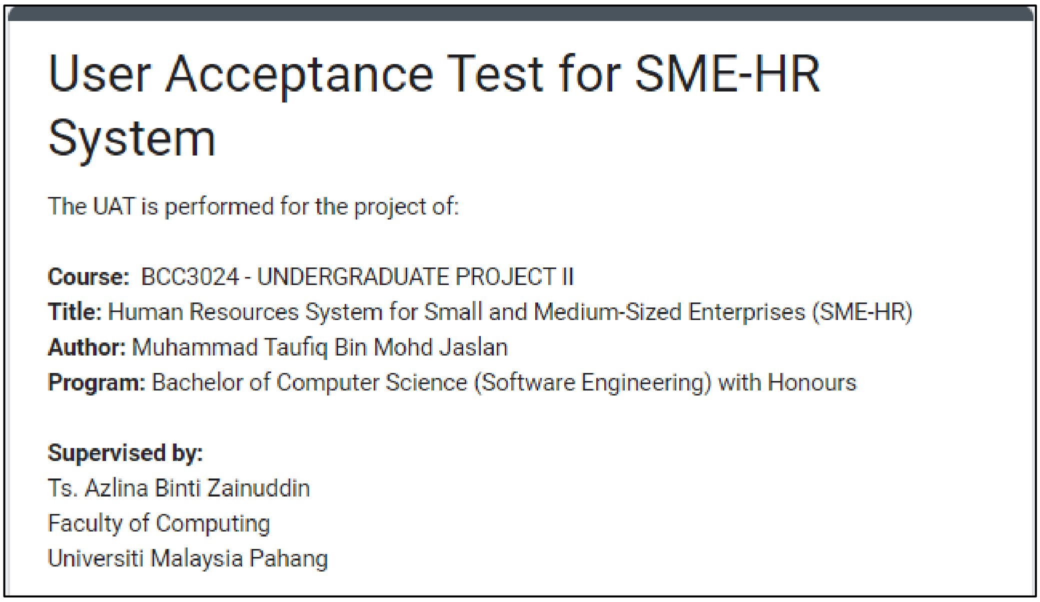
Figure 114: EA Form Controller Snippet Code

Figure 113 presents a code snippet relevant to the EA Form model, specifically about the attributes within the "ea_forms" table in the database. This code snippet defines the attributes contained in the table. It may include fields such as EA form ID, employee ID, position, start date, end date, KWSP number, and other relevant information associated with EA forms. Additionally, the code snippet likely includes a "fillable" function that lists all the attributes from the "ea_forms" table, specifying which attributes can be mass-assigned during the creation or update of EA form records. Furthermore, the "employee" function signifies an essential foreign reference to the related employee table.

In addition to the EA Form model, Figure 114 illustrates the code relevant to the EA Form controller. This code outlines the method functions responsible for managing and storing the input information in the database. For instance, the "storeEAForm" function would typically collect and validate user input before persisting it as a new EA form record in the database. The EA Form controller code effectively handles user requests, facilitates interaction with the EA Form model, and seamlessly integrates with the underlying database.

4.3 User Acceptance Test

User Acceptance Testing (UAT) plays a vital role in software development, allowing end-users to actively test the system's functionality and usability. For this project, UAT involves a series of meticulously designed tests conducted by Anh Exora Enterprises to ensure that the developed system meets their specific needs and expectations. Through UAT, Anh Exora Enterprises can thoroughly evaluate the system, provide valuable feedback, and identify any issues or areas for improvement. This collaborative process fosters effective communication between the development team and end-users, resulting in a high-quality system that aligns with user requirements and delivers a satisfactory user experience.



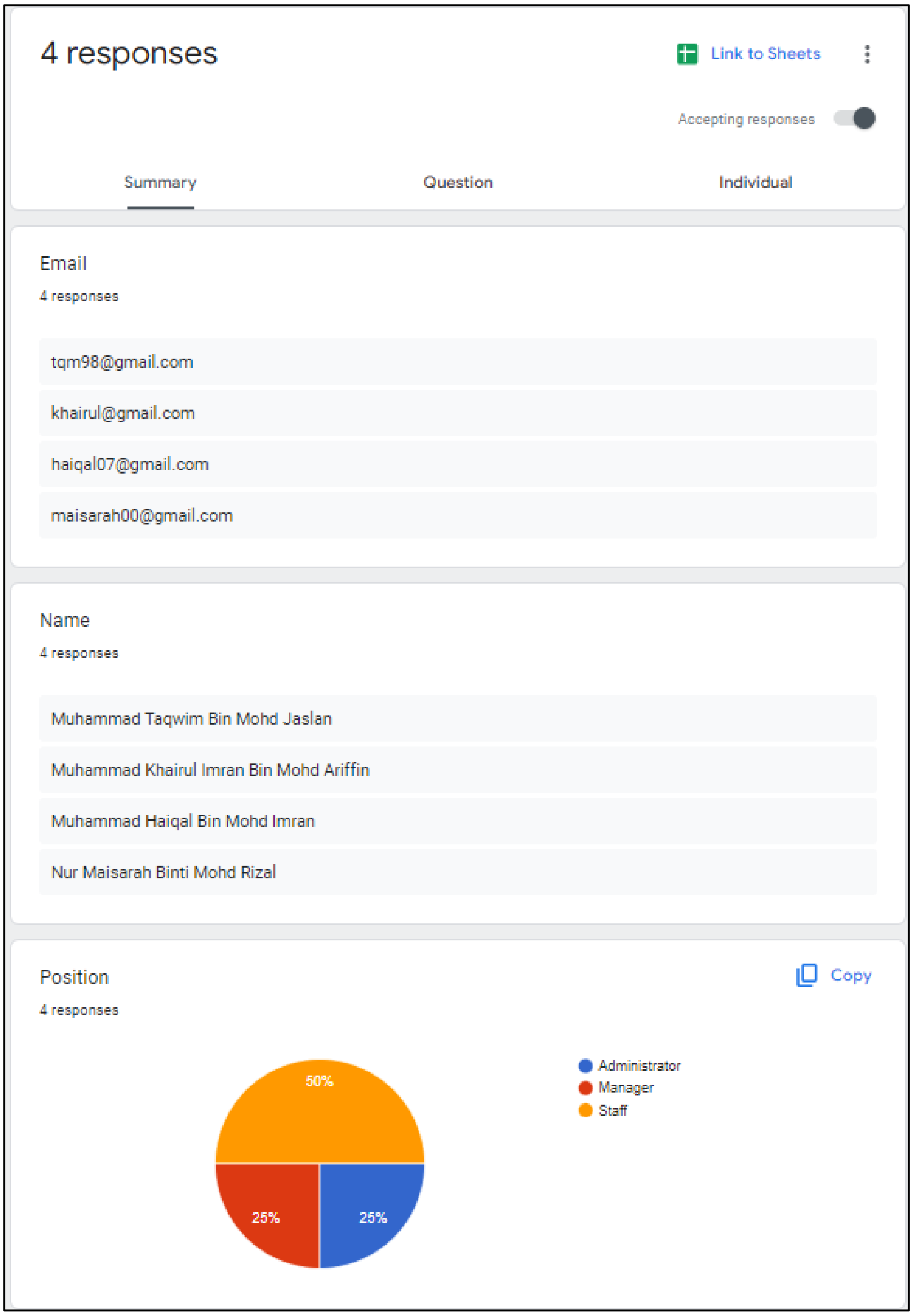
User Acceptance Test for SME-HR System

The UAT is performed for the project of:

Course: BCC3024 - UNDERGRADUATE PROJECT II
Title: Human Resources System for Small and Medium-Sized Enterprises (SME-HR)
Author: Muhammad Taufiq Bin Mohd Jaslan
Program: Bachelor of Computer Science (Software Engineering) with Honours

Supervised by:
Ts. Azlina Binti Zainuddin
Faculty of Computing
Universiti Malaysia Pahang

Figure 115: User Acceptance Test Form Introduction



4 responses

Email

4 responses

tqm98@gmail.com

khairul@gmail.com

haiqal07@gmail.com

maisarah00@gmail.com

Name

4 responses

Muhammad Taqwim Bin Mohd Jaslan

Muhammad Khairul Imran Bin Mohd Ariffin

Muhammad Haiqal Bin Mohd Imran

Nur Maisarah Binti Mohd Rizal

Position

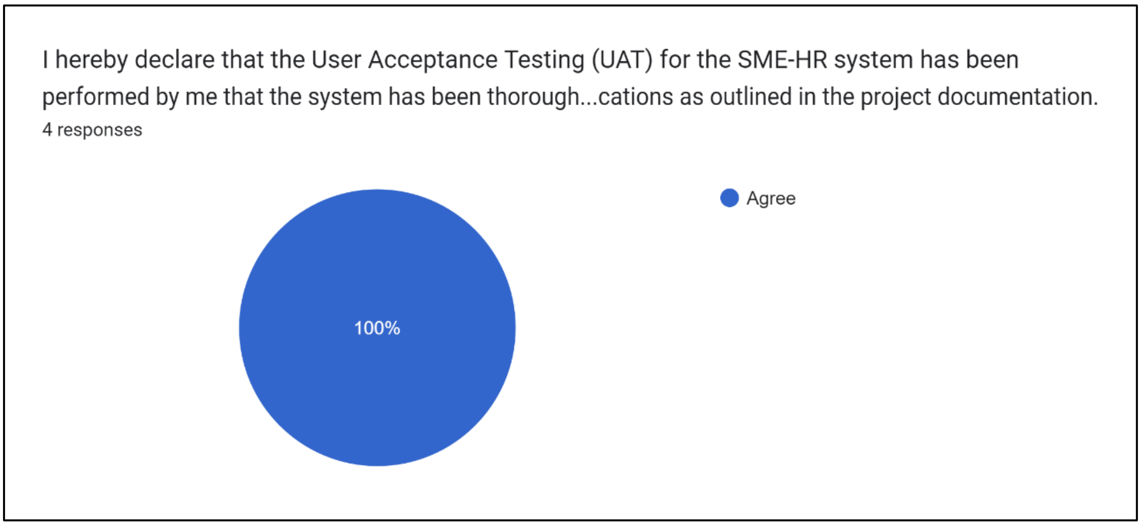
4 responses

Figure 116: Respondent Personal Details

Figure 115 illustrates the introduction of the User Acceptance Test (UAT) form, providing brief details about the author and the supervisor involved in the process. The form is crucial in assessing the system's acceptance by end-users.

Furthermore, the Figure 116 presents the results of the UAT, displaying comprehensive information about the participating respondents. The details encompass their names, email addresses, and respective positions within Ahn Exora Enterprises. The UAT process involved four respondents from the organization, comprising one Administrator, one Manager, and two Staff members.

For a comprehensive and in-depth understanding of the UAT responses, readers are encouraged to refer to Appendix A, which contains the complete responses collected during the user acceptance test. These responses provide valuable insights into the system's usability, functionality, and overall user satisfaction.


I hereby declare that the User Acceptance Testing (UAT) for the SME-HR system has been performed by me that the system has been thorough...cations as outlined in the project documentation.

4 responses

Figure 117: Declaration Agreement Response

Figure 117 showcases the inclusion of a tester declaration, which formally acknowledges the completion and submission of the User Acceptance Test (UAT) form. This documentation plays a crucial role in officially recording and monitoring the progress of the UAT process. The tester declaration signifies that the testers have diligently evaluated the system, provided valuable feedback, and affirmed their confidence in its deployment preparedness. This declaration is a significant milestone, ensuring that the developer can proceed with assurance, knowing that the system has undergone comprehensive testing and is deemed ready for implementation.

CHAPTER 5

CONCLUSION

5.1 Introduction

This chapter concludes the research on the SME-HR system, which aims to meet the human resources management needs of small and medium-sized enterprises (SMEs). The project utilized the Rapid Application Development (RAD) methodology and focused on rapid software development and iterative improvements.

It presents conclusive findings, demonstrating the SME-HR system's effectiveness in addressing SMEs' HR management requirements and aligning with project objectives. The chapter also evaluates the collected data's relevance and integration into the system's functionalities.

Furthermore, it reflects on the implementation of RAD, highlighting its strengths and limitations in developing the SME-HR system. Challenges faced during implementation are addressed, and recommendations for future enhancements and advancements are provided.

In summary, this chapter consolidates the research findings, assesses data suitability, reflects on the RAD methodology, and offers valuable recommendations for the future development of the SME-HR system.

5.2 Objective Revisited

The objectives of this project have been successfully accomplished, as evidenced by the comprehensive study conducted on existing HR systems, including altHR, Ignite Human Resources Management System, and AuroraHRM. This study thoroughly examined the limitations of current human resources systems, revealing their inadequacy for SME companies. Consequently, this study laid the groundwork for developing an SME-HR system explicitly tailored to the needs of SME companies.

Moreover, the proposed system underwent meticulous development and rigorous testing. The test results demonstrate the system's robust functionality in effectively managing various aspects of human resources work, with particular emphasis on the payroll module, which incorporates an automatic payroll calculator. The system encompasses essential modules such as employee management, claims management, payroll management, leave management, and EA form management, enabling streamlined HR operations for SME companies.

In conclusion, this project has achieved its objectives, culminating in creating the SME-HR system explicitly designed for SME companies. This system will prove invaluable to Ahn Exora Enterprises and other similar organizations, empowering them to efficiently manage their HR tasks while addressing the unique requirements and challenges associated with SME operations.

5.3 Research Constraint

This section aims to comprehensively analyze the limitations and challenges faced during this project, which hindered its successful completion. These constraints encompassed a range of factors that presented obstacles and demanded meticulous attention and effective management.

5.3.1 Limited Time

The proposed system for SME-HR has several limitations that warrant careful consideration. Firstly, time constraints played a significant role in shaping the project's progress. One of the key challenges was the collection of surveys and requirements from SME companies. Despite proactive efforts, some companies did not provide timely responses, impeding the gathering of essential information. Furthermore, the comprehensive development of the system required an in-depth understanding of the intricacies of human resources workflows, particularly the intricacies of the payroll module. Extensive research was necessary to comprehend the calculation of essential elements such as KWSP (Employees' Provident Fund), SOCSO (Social Security Organization), and EIS (Employment Insurance Scheme), which further contributed to the time limitations of the project.

Additionally, resource availability posed a constraint to the project's completion. Securing a skilled workforce with expertise in human resources management and software development proved challenging. The project necessitated individuals who possessed a deep understanding of both HR practices and software development methodologies. This constraint required careful allocation and utilization of available expertise to ensure optimal progress.

In summary, the proposed SME-HR system encountered limitations primarily related to time constraints, including difficulties in obtaining timely survey responses and the need for extensive research on human resources workflows and payroll calculations. Additionally, resource availability posed a challenge in securing the right expertise for the project. Despite these limitations, proactive measures were taken to mitigate their impact and ensure the successful completion of the system.

5.3.2 Development Constraints

One of the significant challenges and limitations encountered during the development process of the SME-HR system pertained to development constraints. The primary challenge revolved around creating a logical calculation algorithm for the payroll and leave modules. These modules required intricate calculations that adhered to specific contribution rates stipulated by Perkeso, including the Rates of Contribution for Employees' Social Security Act 1969 (Act 4) and the Rates of Contribution for the Employment Insurance System (Act 800).

Developing an accurate and efficient algorithm to calculate payroll and leave entitlements necessitated a thorough understanding of the relevant legislative acts and their corresponding contribution rates. The algorithm needed to incorporate complex calculations to ensure compliance with the prescribed rates and accurately determine the employees' social security and employment insurance contributions.

Despite the complexity of this task, the process development process remains committed to developing a robust and reliable system that would streamline HR processes for SMEs. By overcoming the development constraints and successfully implementing the logic calculation algorithm, the SME-HR system aimed to provide accurate and efficient payroll and leave management functionalities to enhance HR operations within SMEs.

5.4 Future Work

Regarding future plans for the system, several key considerations exist to enhance its functionality and user experience. Firstly, there is a plan to introduce additional modules, such as a performance module. This module would allow managers to monitor and track individual staff members' work progress and performance. By incorporating this module, the system would enable managers to gain valuable insights into employee productivity and identify areas for improvement.

Furthermore, a potential development in the future is integrating the system into a mobile application platform. Given the widespread use of smartphones in today's technology-driven era, providing users with a mobile application would enhance accessibility and convenience. Users could access the system and perform tasks directly from their smartphones, facilitating efficient management of human resources even while on the go.

In conclusion, the future plan for this system entails optimizing the user experience and catering to users' evolving needs and preferences by expanding its capabilities by incorporating additional modules and embracing the mobile application platform. Through these enhancements, managers will be empowered with comprehensive monitoring tools to effectively track and manage various aspects of employee performance. Simultaneously, users will benefit from a seamless and user-friendly system that they can conveniently access through their preferred devices. This strategic approach is expected to significantly enhance productivity and efficiency in human resources management processes.

REFERENCES

13 RAD Benefits and Advantages You Could Certainly Expect. (n.d.). Retrieved February 11, 2023, from https://kissflow.com/application-development/rad/benefits-of-rapid-application-development/

Abbas, N., Gravell, A. M., & Wills, G. B. (2008). Historical roots of agile methods: Where did "Agile thinking" come from? *Lecture Notes in Business Information Processing*, *9 LNBIP*, 94–103. https://doi.org/10.1007/978-3-540-68255-4_10

altHR | Simplified HR Management For Businesses. (n.d.). Retrieved November 20, 2022, from https://althr.my/features/core-hr

altHR Pricing | Affordable All-In-One HRMS System For Businesses. (n.d.). Retrieved November 20, 2022, from https://althr.my/pricing

Disadvantages Of A Manual HR Administration: How They Can Affect Your Organization - SutiSoft. (n.d.). Retrieved February 11, 2023, from https://www.sutisoft.com/blog/disadvantages-of-a-manual-hr-administration-how-they-can-affect-your-organization/

Exploring the History and Evolution of the Laravel Framework. (n.d.). Retrieved May 1, 2023, from https://www.w3schools.in/laravel/history

Features - Aurora Cloud Works. (n.d.). Retrieved November 20, 2022, from https://auroracw.com/price/

Ignite HRMS Payroll System | True Multi-Country HR & Payroll Processing. (n.d.). Retrieved November 20, 2022, from https://ignitepayroll.boardroomlimited.com/

Product - Aurora Cloud Works. (n.d.). Retrieved November 20, 2022, from https://auroracw.com/product/

Rate of Contribution. (n.d.). Retrieved June 10, 2023, from https://www.perkeso.gov.my/en/rate-of-contribution.html

Shamita, G. (2022). *Implementation of Human Resources System In The Company*. https://www.researchgate.net/publication/364335598

Sunardi, A., & Suharjito. (2019). MVC architecture: A comparative study between laravel framework and slim framework in freelancer project monitoring system web based. *Procedia Computer Science*, *157*, 134–141. https://doi.org/10.1016/j.procs.2019.08.150

The Advantages and Disadvantages of the RAD Model - DistantJob - Remote Recruitment Agency. (n.d.). Retrieved February 7, 2023, from https://distantjob.com/blog/rad-model-advantages-and-disadvantages/

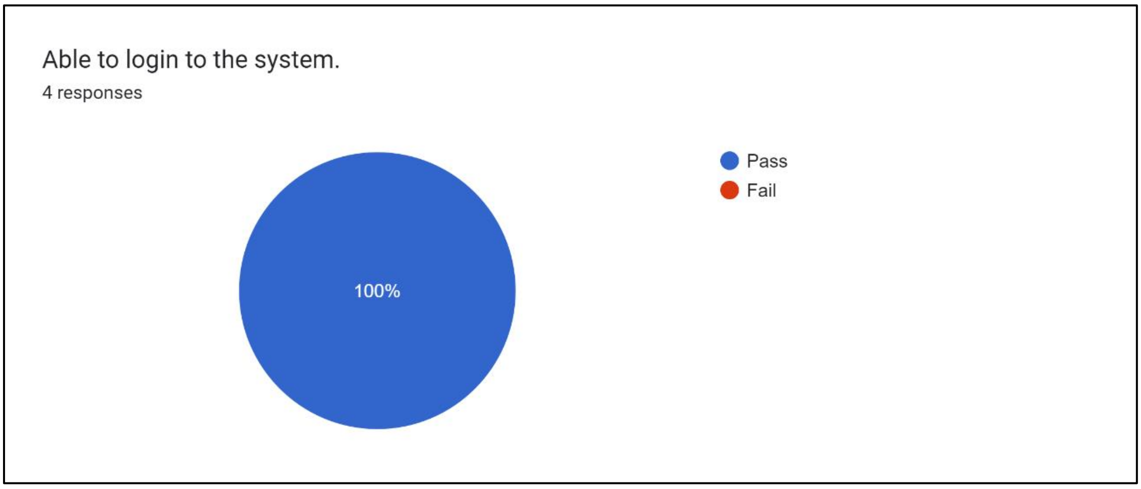
The history of how Rapid Application Development became Agile. (n.d.). Retrieved February 7, 2023, from https://www.onpathtesting.com/blog/history-of-rapid-application-development-and-agile

Top 10 HR Software in Malaysia - Software Reviews, Pricing, Comparison 2022 | Alternatives. (2022, September 2). https://www.wesuggestsoftware.com/top-10-hr-software-in-malaysia/

Your Complete Guide To Rapid Application Development (RAD). (n.d.). Retrieved February 11, 2023, from https://marutitech.com/rapid-application-development/

APPENDIX A
USER ACCEPTANCE TEST RESPONSES

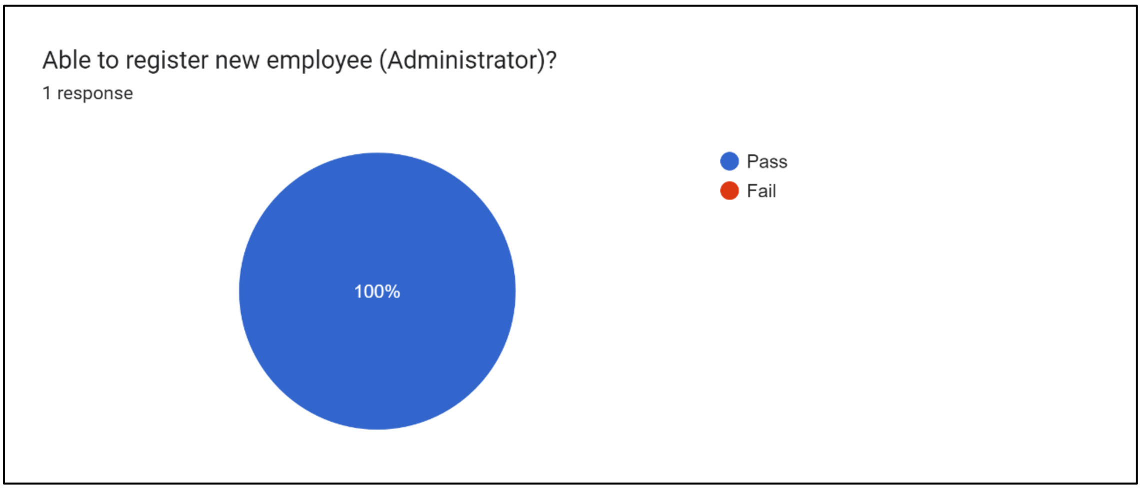
Login



Able to login to the system.

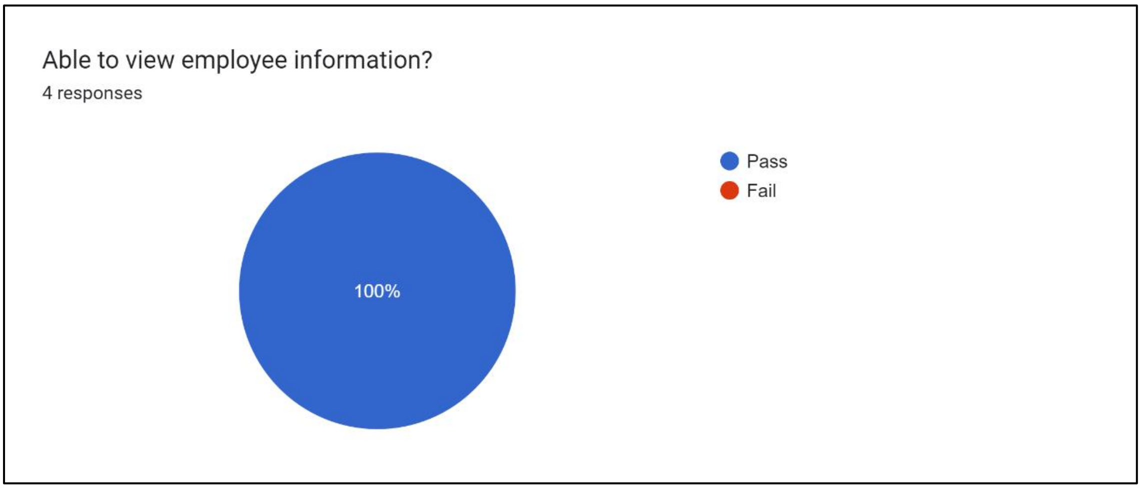
4 responses

Manage Employee



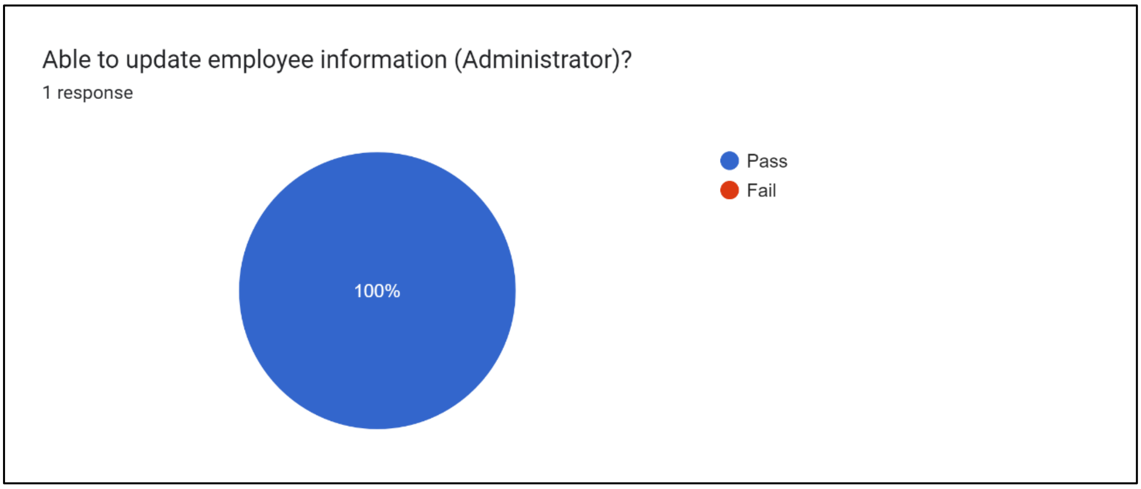
Able to register new employee (Administrator)?

1 response



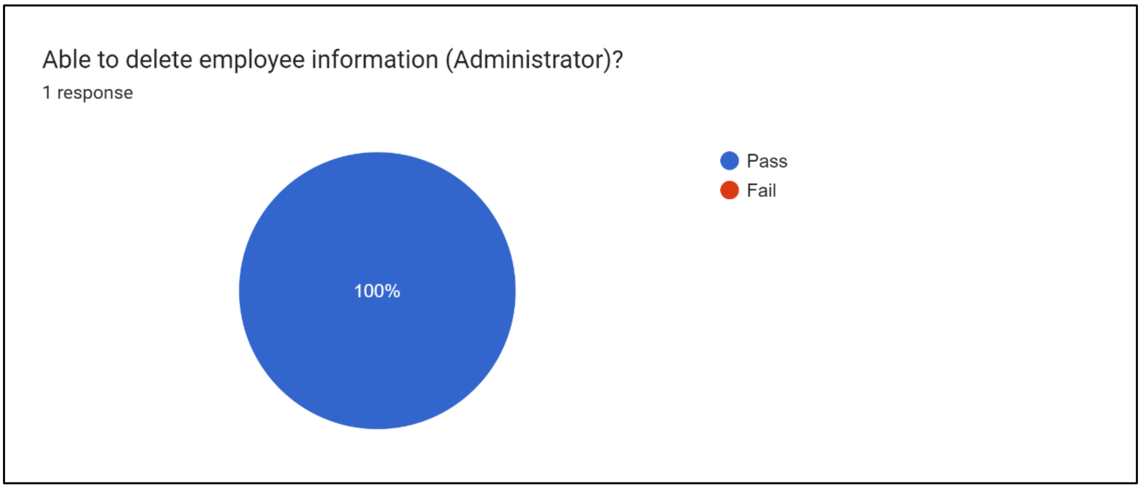
Able to view employee information?

4 responses



Able to update employee information (Administrator)?

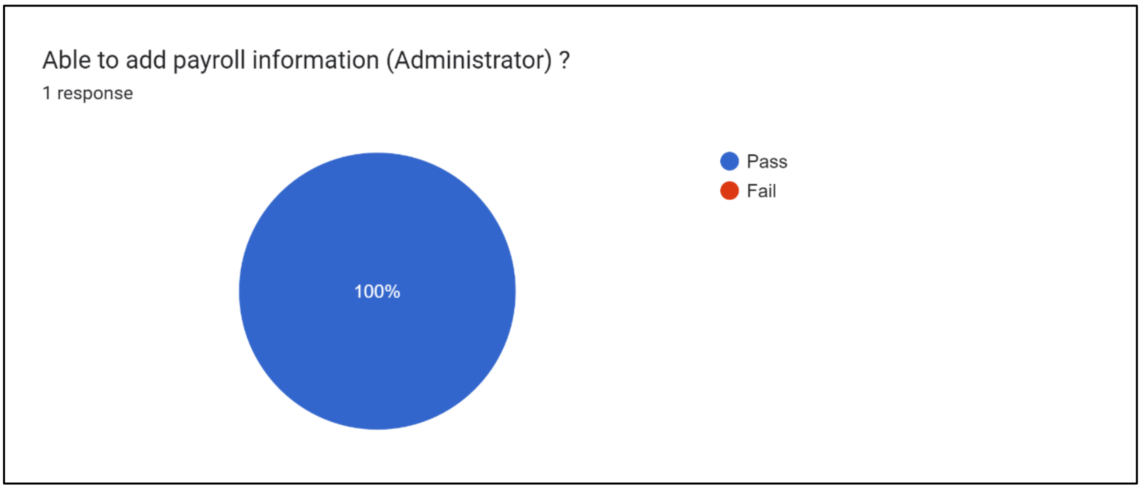
1 response



Able to delete employee information (Administrator)?

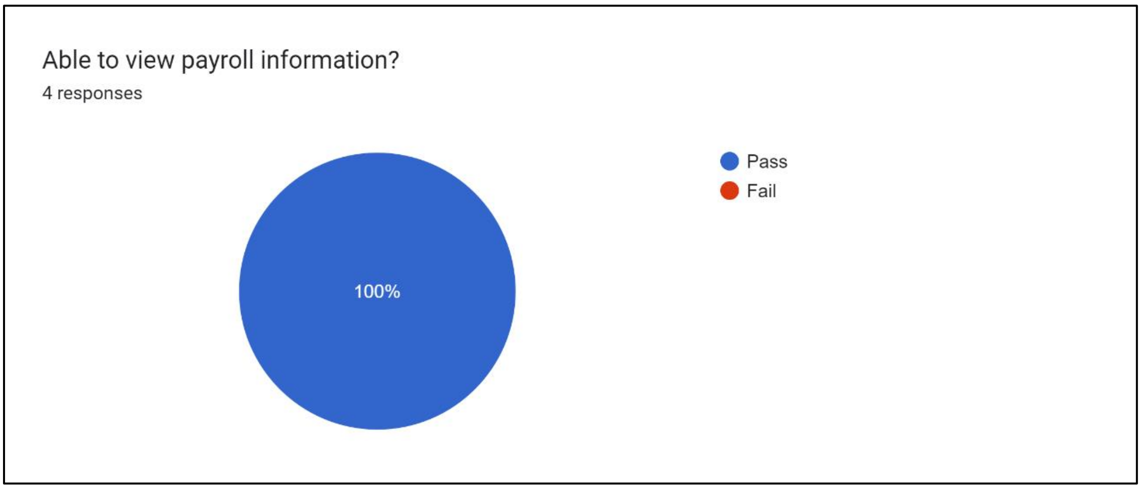
1 response

Manage Payroll



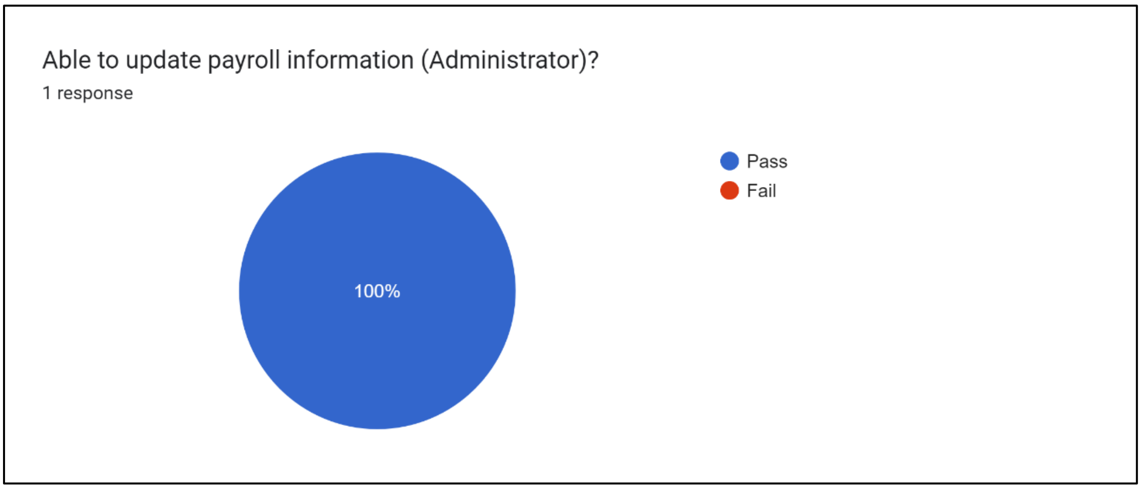
Able to add payroll information (Administrator) ?

1 response



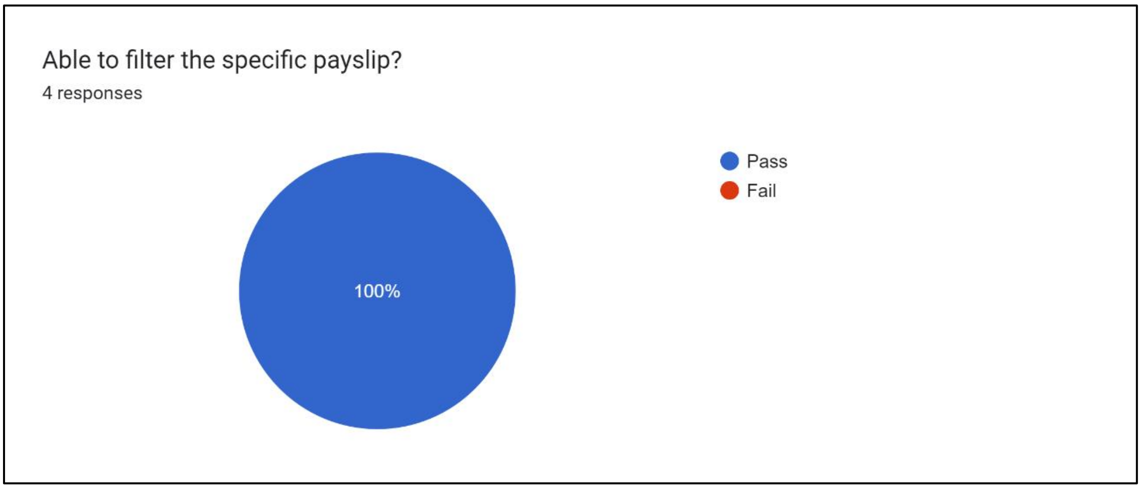
Able to view payroll information?

4 responses



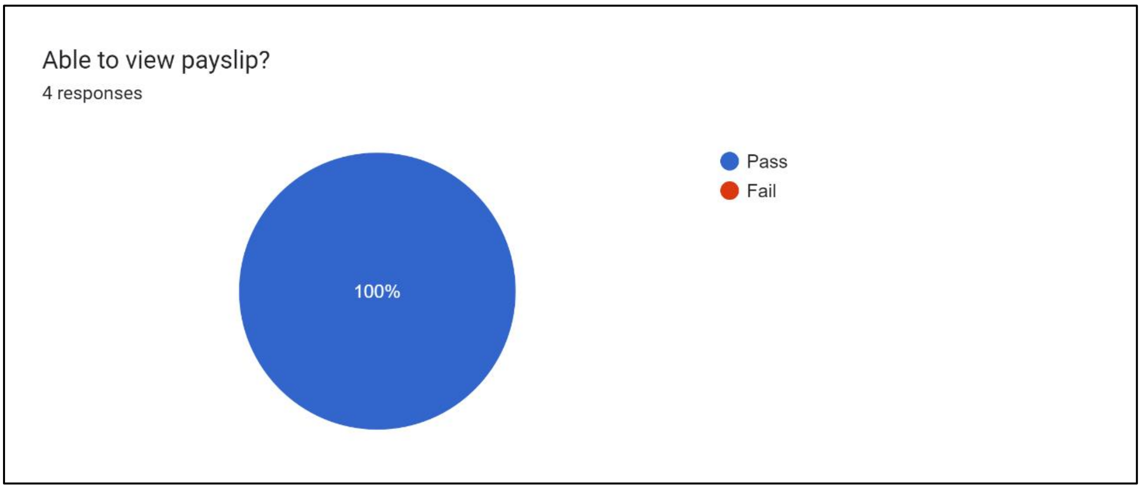
Able to update payroll information (Administrator)?

1 response



Able to filter the specific payslip?

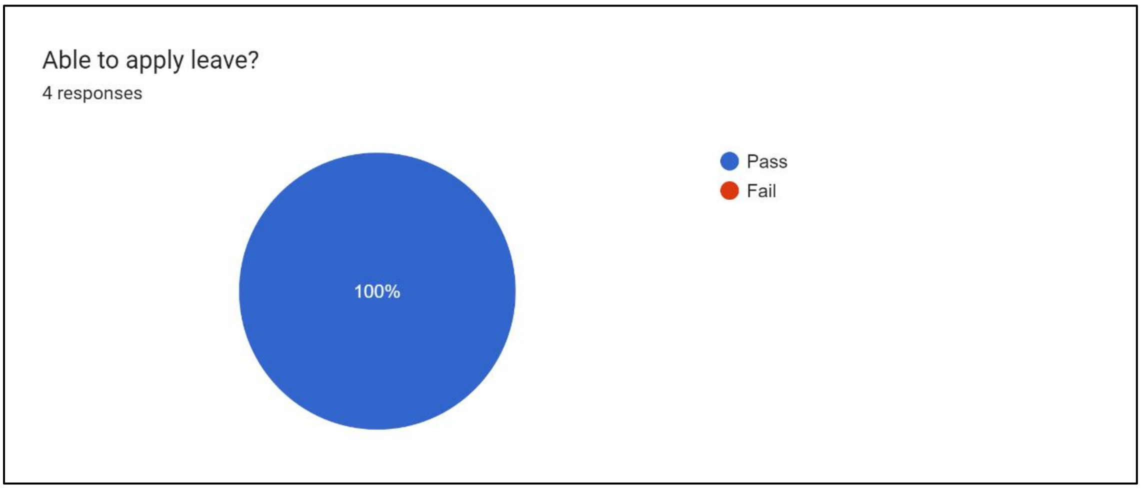
4 responses



Able to view payslip?

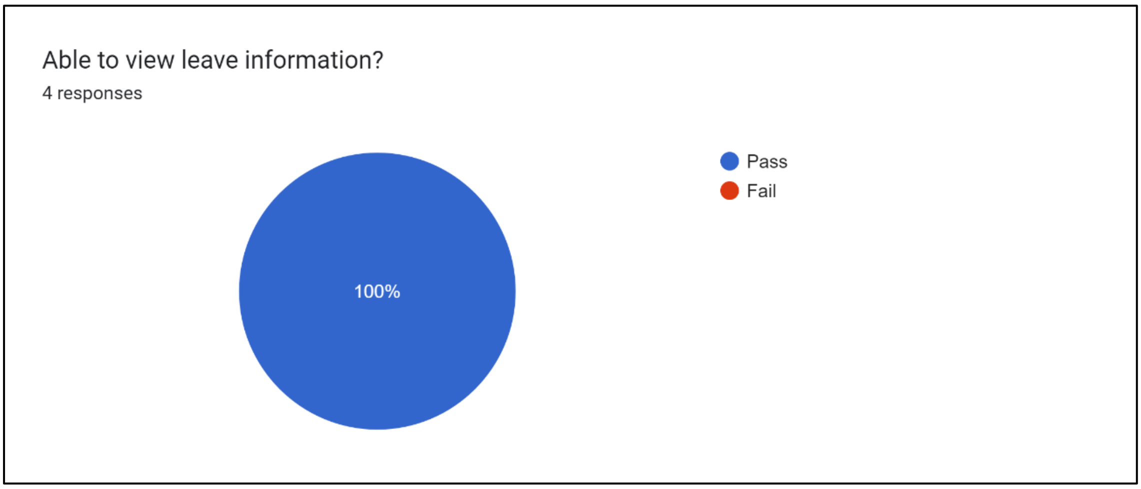
4 responses

Manage Leave



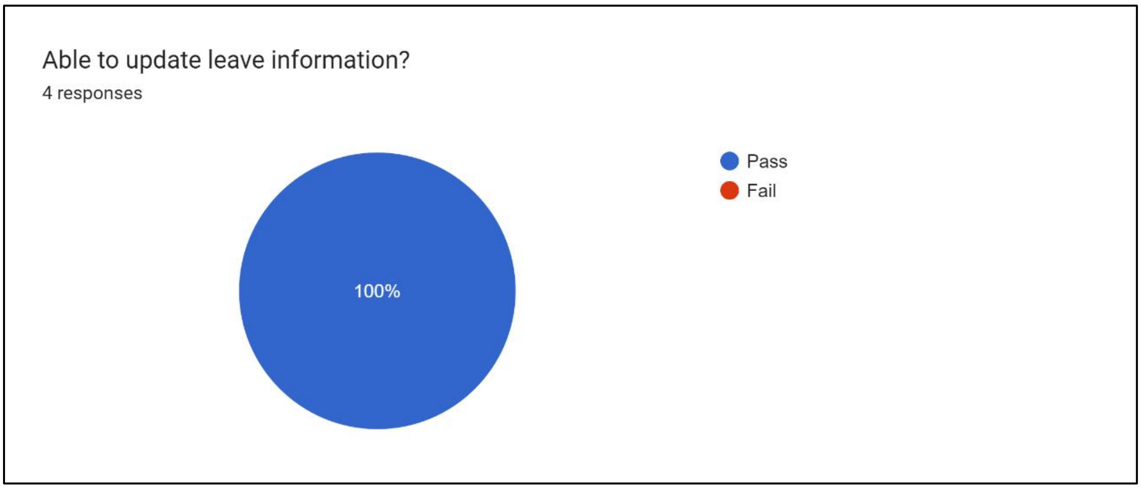
Able to apply leave?

4 responses



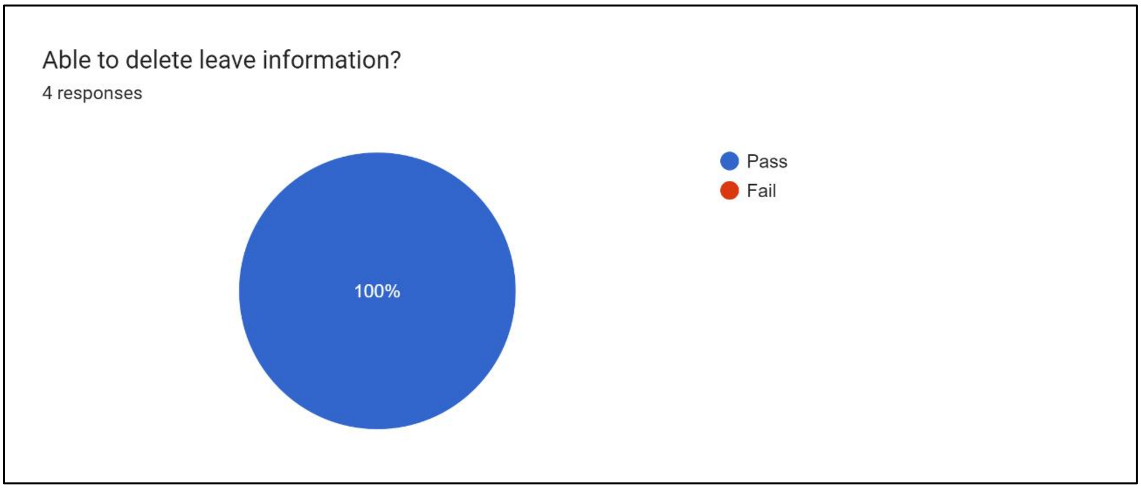
Able to view leave information?

4 responses



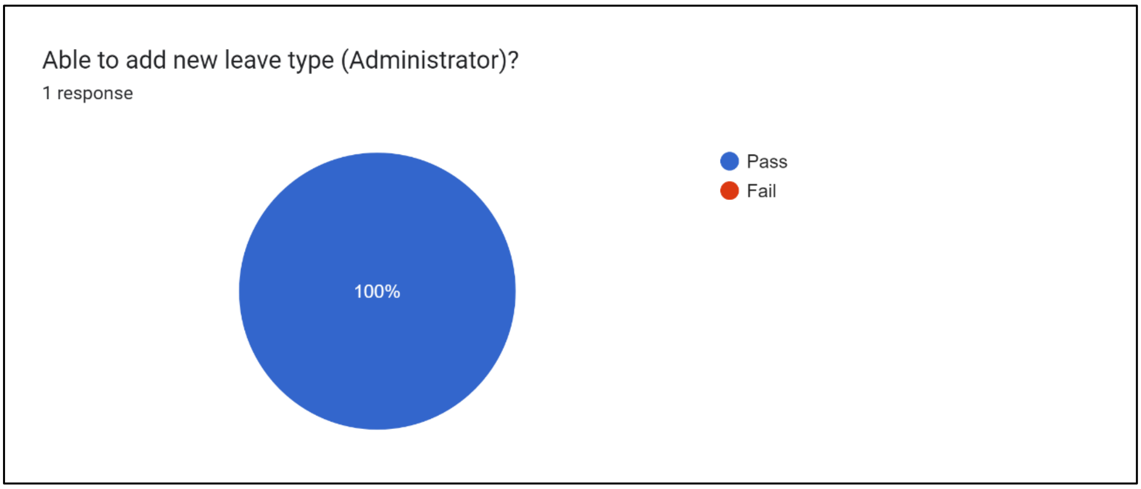
Able to update leave information?

4 responses



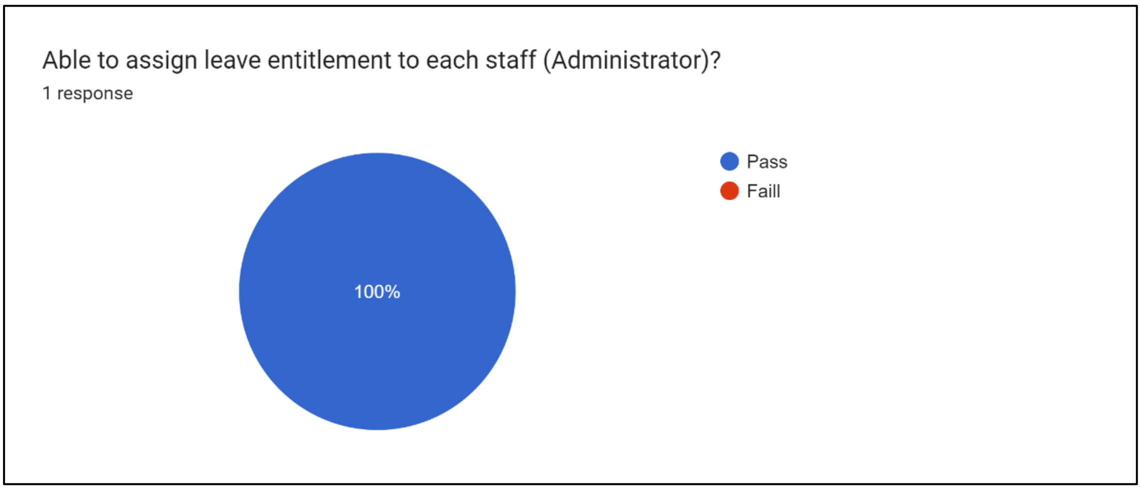
Able to delete leave information?

4 responses



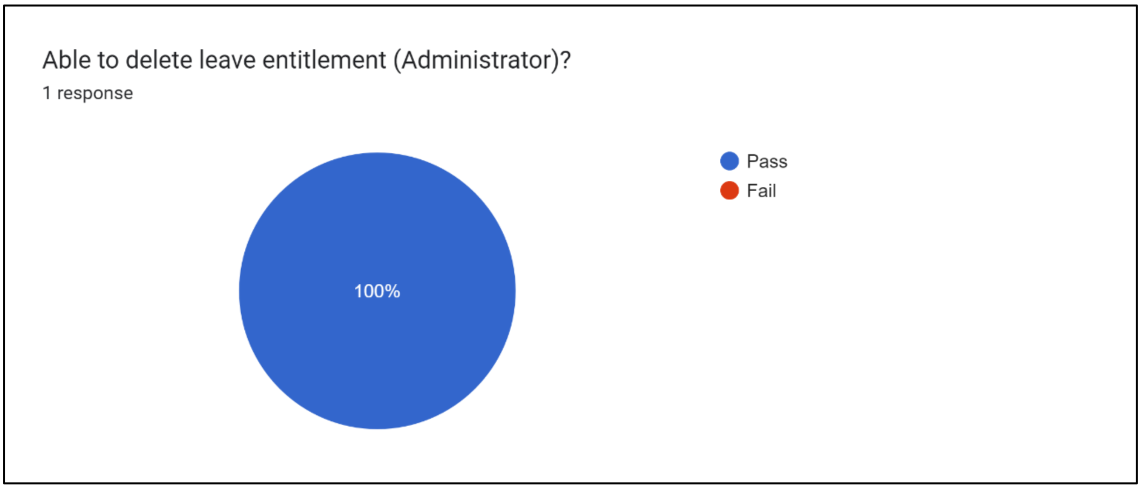
Able to add new leave type (Administrator)?

1 response



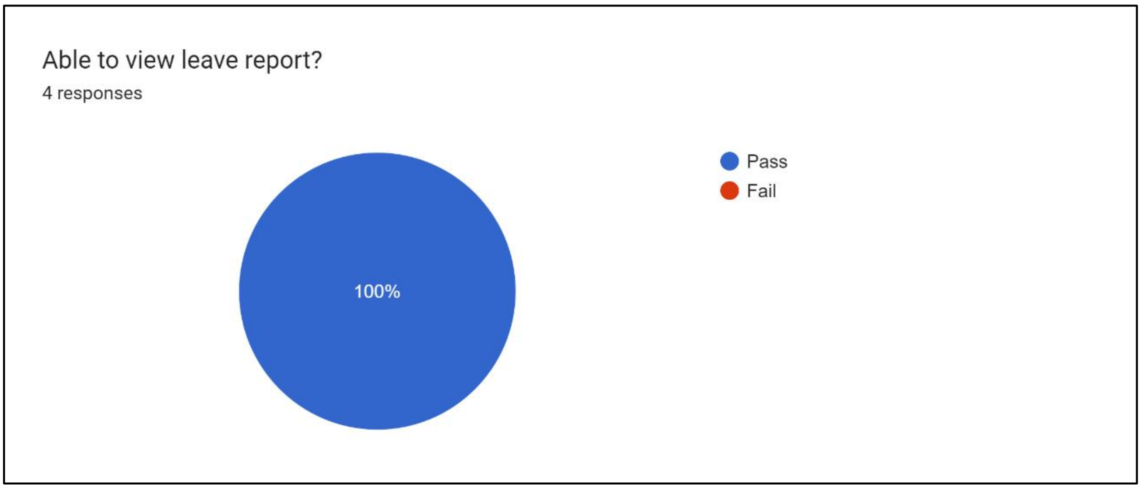
Able to assign leave entitlement to each staff (Administrator)?

1 response



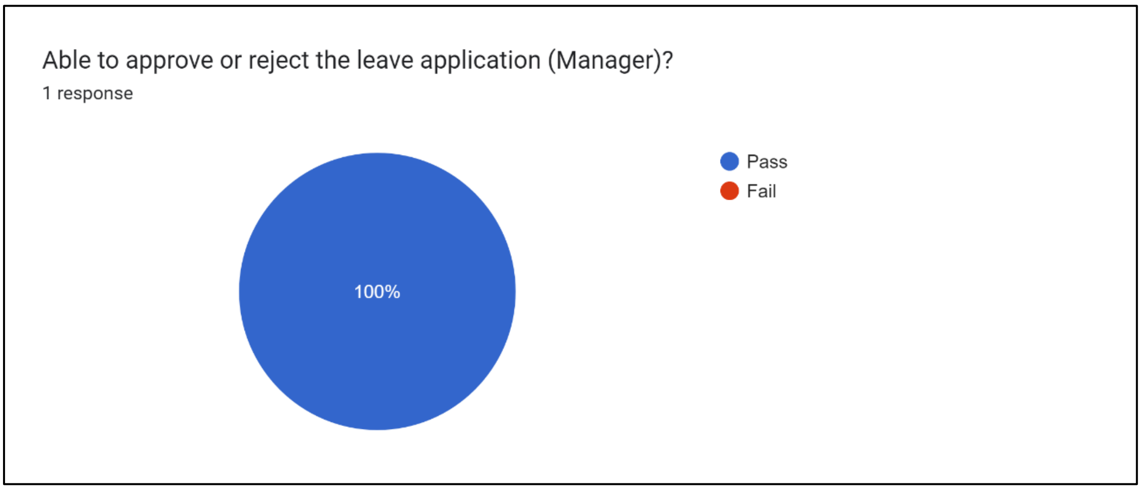
Able to delete leave entitlement (Administrator)?

1 response



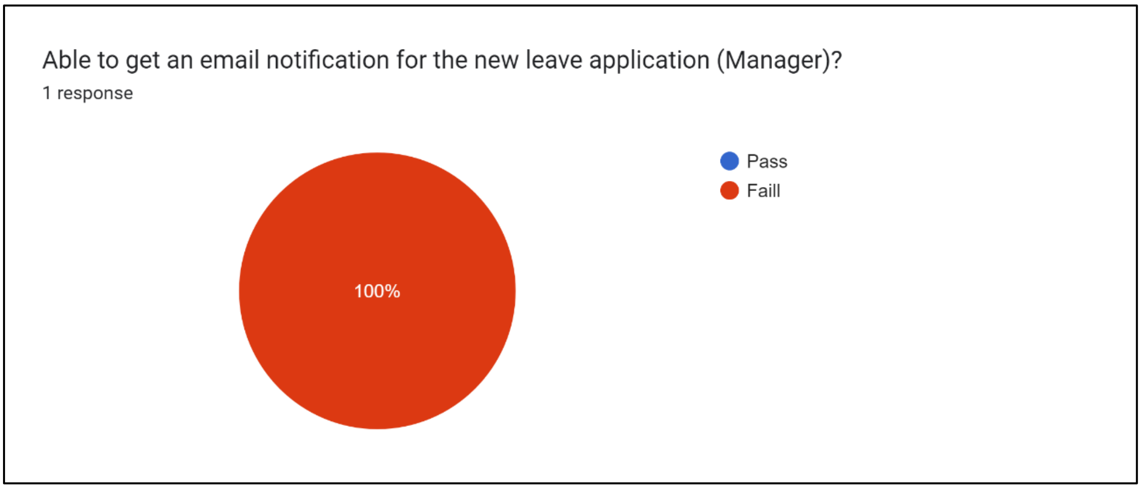
Able to view leave report?

4 responses



Able to approve or reject the leave application (Manager)?

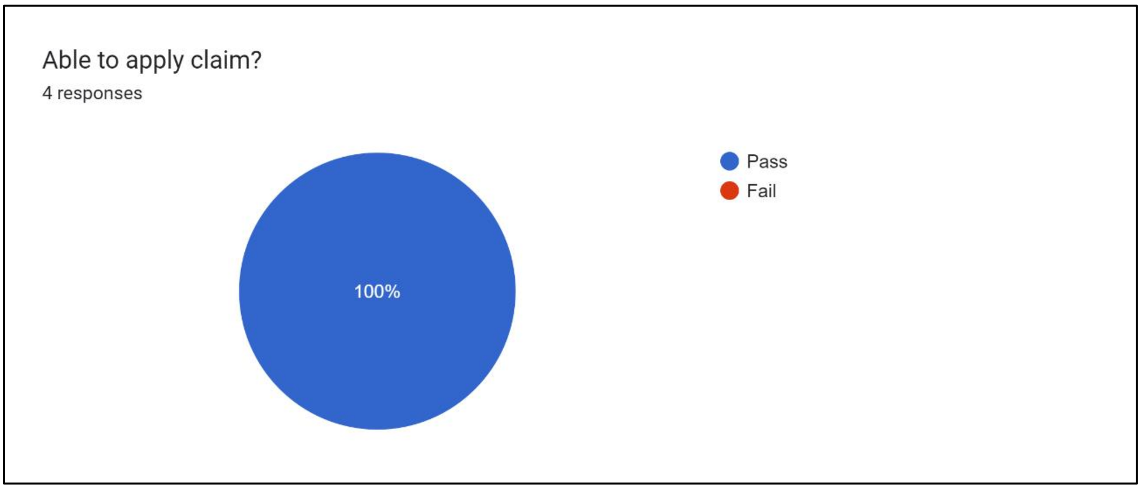
1 response

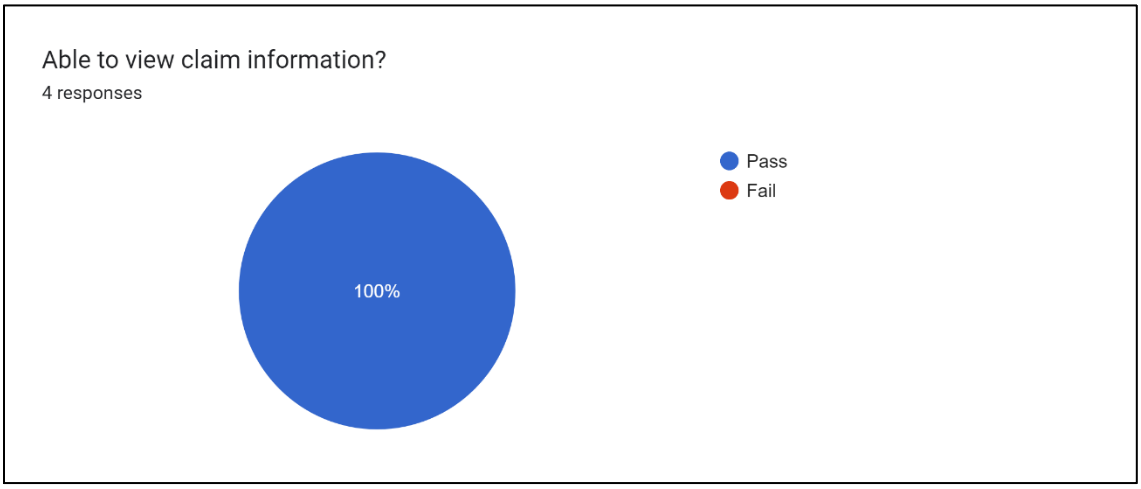


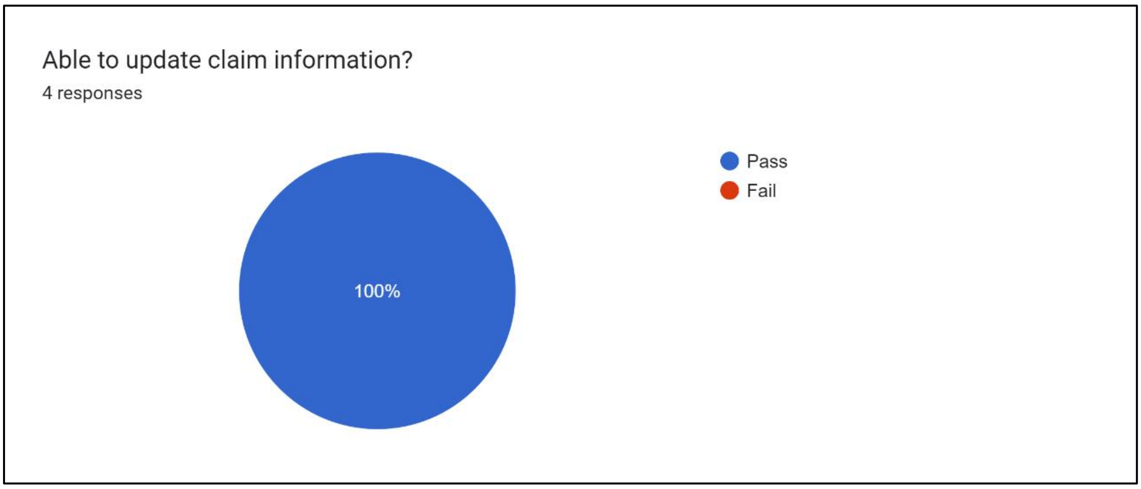
Able to get an email notification for the new leave application (Manager)?

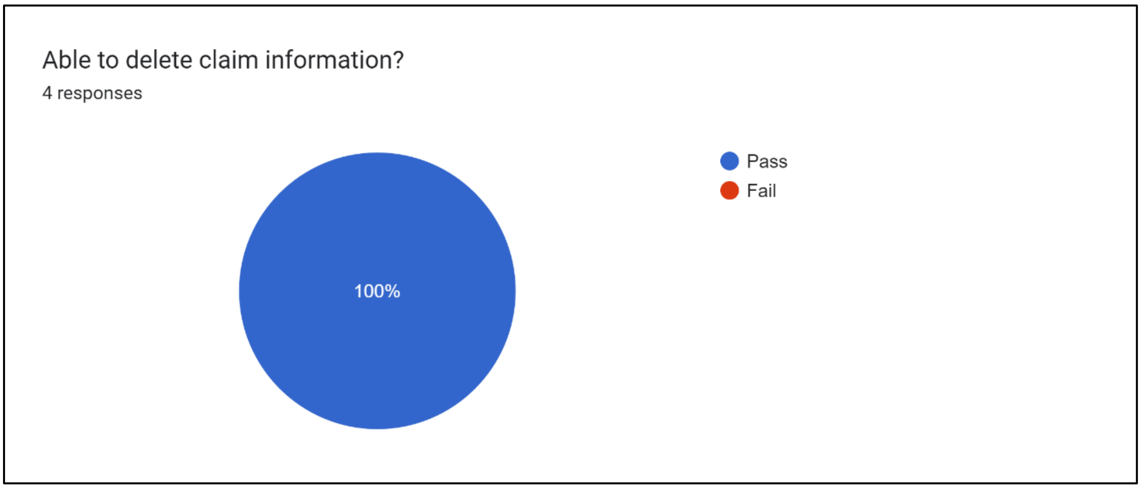
1 response

Manage Claim


Able to apply claim?
4 responses

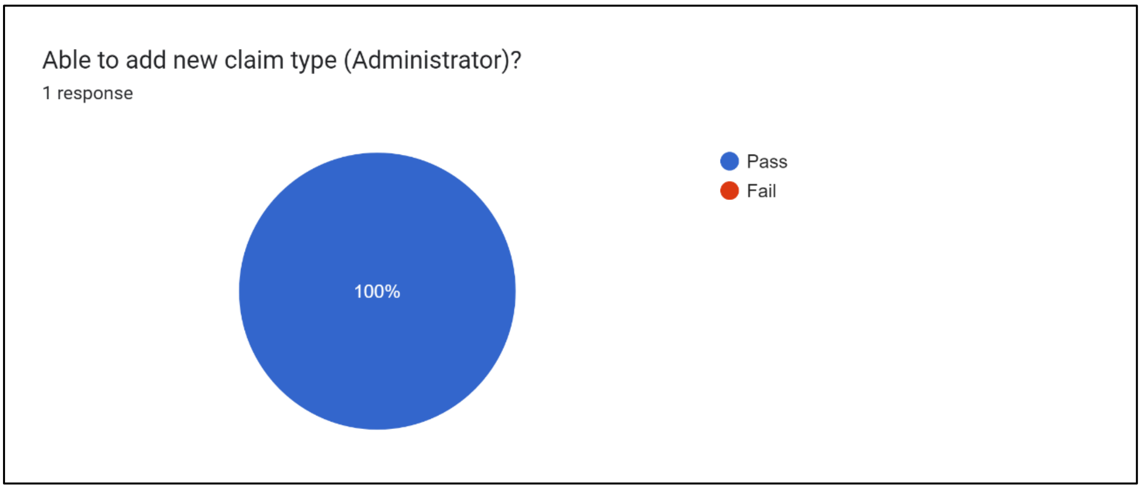

Able to view claim information?
4 responses


Able to update claim information?
4 responses



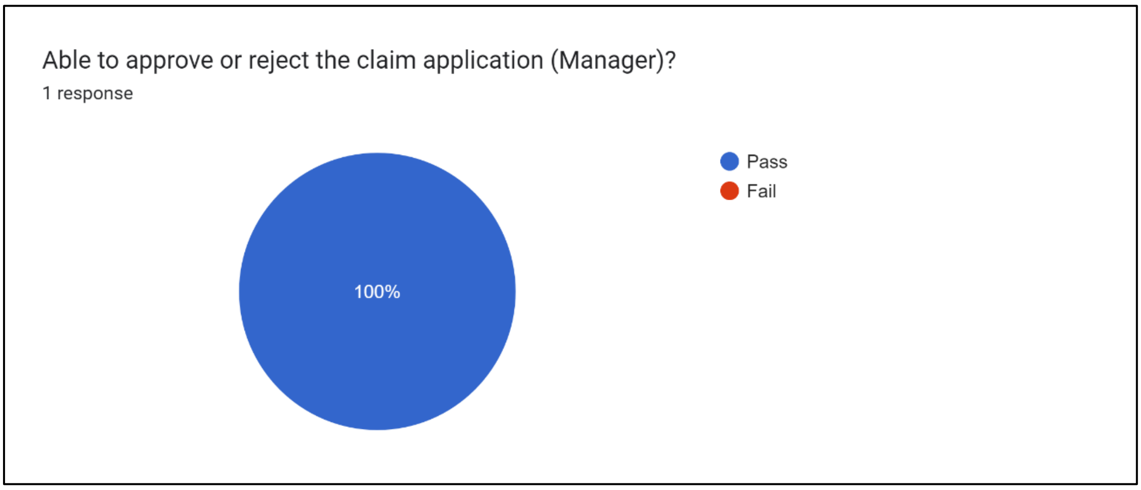
Able to delete claim information?

4 responses



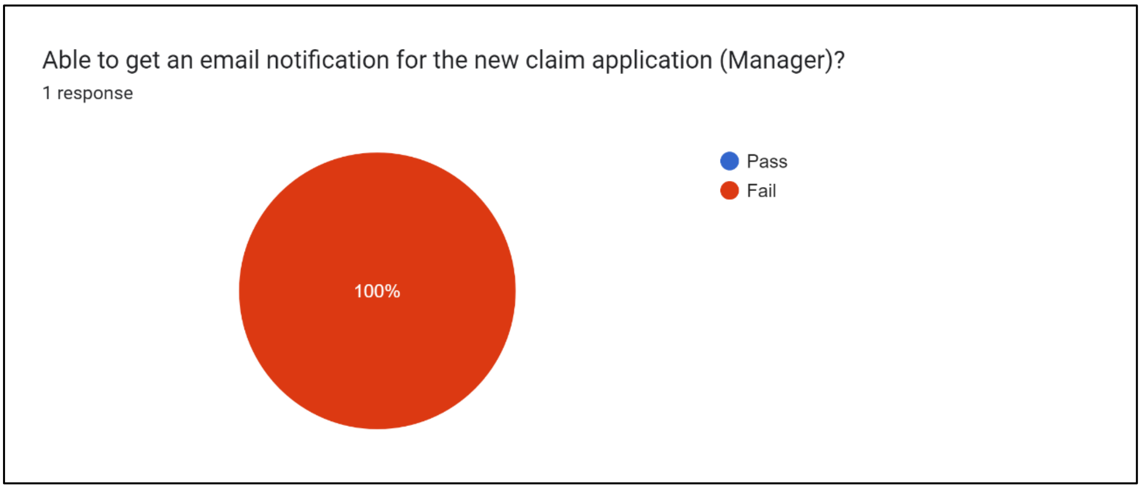
Able to add new claim type (Administrator)?

1 response



Able to approve or reject the claim application (Manager)?

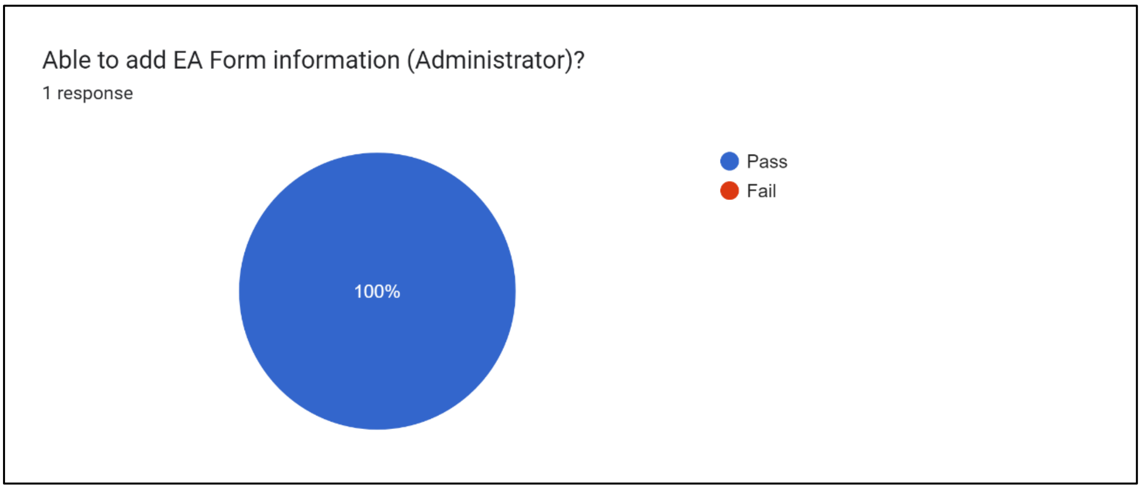
1 response



Able to get an email notification for the new claim application (Manager)?

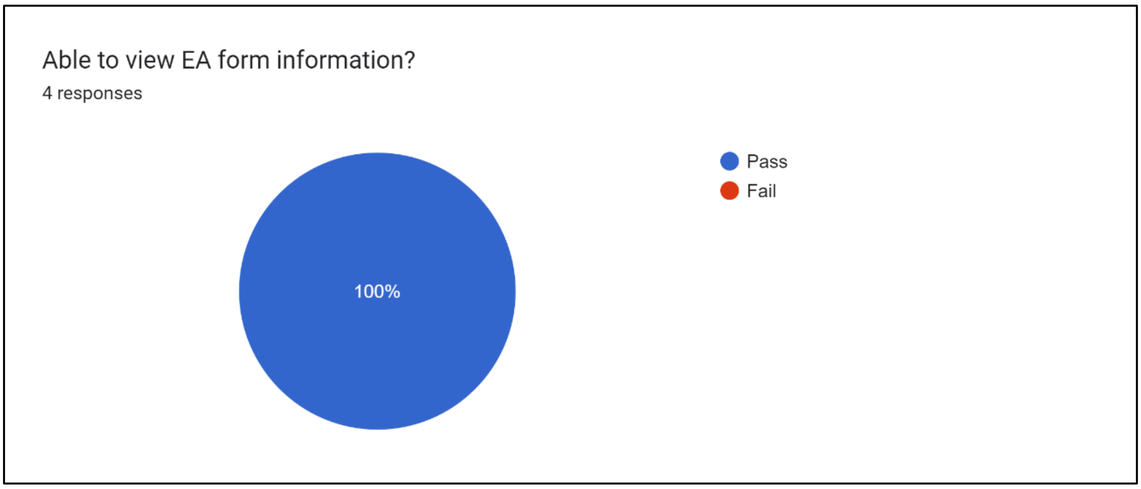
1 response

Manage EA Form



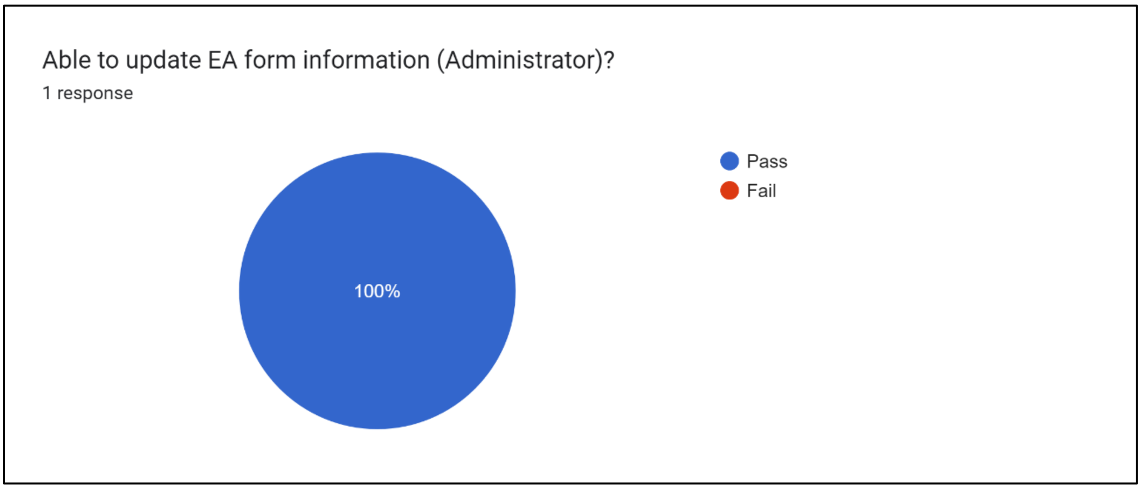
Able to add EA Form information (Administrator)?

1 response



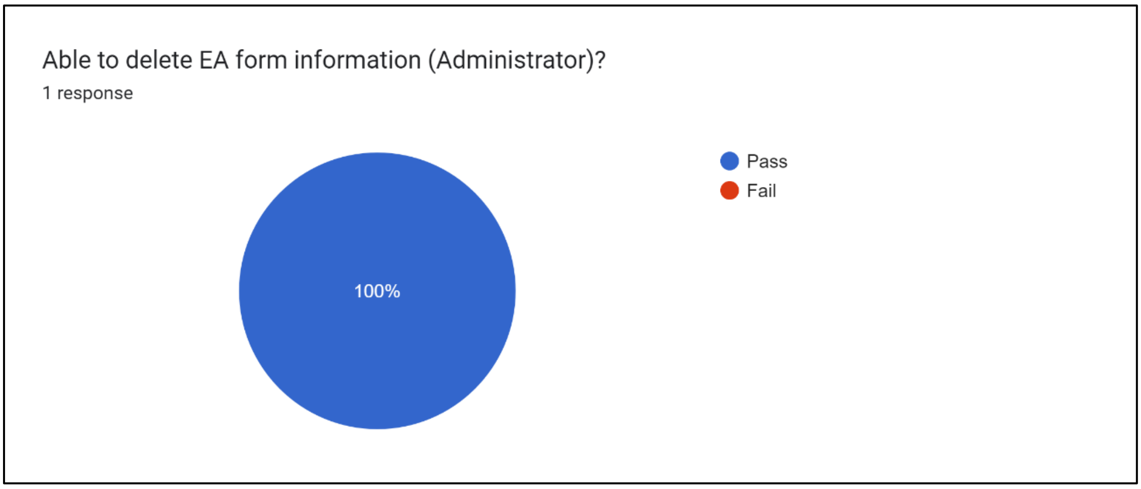
Able to view EA form information?

4 responses



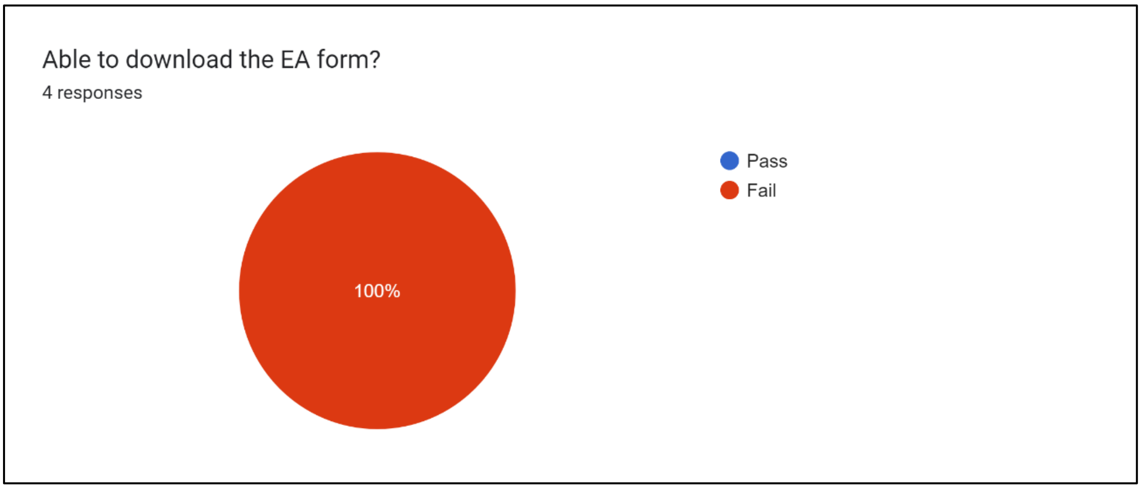
Able to update EA form information (Administrator)?

1 response



Able to delete EA form information (Administrator)?

1 response



Able to download the EA form?

4 responses

APPENDIX B
SOFTWARE REQUIREMENT SPECIFICATION

2023

SOFTWARE REQUIREMENT SPECIFICATION (SRS)

SME-HR

DOCUMENT APPROVAL

	Name	Date
Authenticated by: ________________________ Project Leader	MUHAMMAD TAUFIQ BIN MOHD JASLAN	2 JULY 2023
Approved by: ________________________ Project Manager	MUHAMMAD TAUFIQ BIN MOHD JASLAN	2 JULY 2023
Approved by: ________________________ Client	MUHAMMAD IMRAN BIN SALLEH	2 JULY 2023

Software	: Microsoft Word, Google Docs, Draw.io, Mendeley, Balsamiq Mockup
Archiving Place	:
Copies Available	: Softcopy

TABLE OF CONTENTS

DOCUMENT APPROVAL ii

TABLE OF CONTENTS iii

LIST OF FIGURES iv

LIST OF TABLES iv

LIST OF APPENDIXES vi

1. INTRODUCTION 1

1.1 PURPOSE 1

1.2 SYSTEM IDENTIFICATION 1

1.3 SYSTEM OVERVIEW 1

1.4 REFERENCES 3

1.5 DOCUMENT OVERVIEW 3

2. PRODUCT DESCRIPTION 4

2.1 Product Perspective 4

2.2 Product Functions 5

2.3 User Characteristics 6

2.4 Constraints 6

2.6 Assumptions and Dependencies 7

3. SPECIFIC REQUIREMENTS 8

3.1 Software Product Features 8

3.1.1 Login 8

3.1.2 Manage Employee 10

3.1.3 View Employee 13

3.1.4 Manage Profile 15

3.1.5 Manage Payroll 16

3.1.6 View Payroll 19

3.1.7 Manage Leave 21

3.1.8 Manage Claim 27

3.1.9 Manage EA Form 33

3.1.10 View EA Form 36

4. REQUIREMENT TRACEABILITY 39

5. ACRONYMS AND ABBREVIATION 44

LIST OF FIGURES

Figure 1: SME-HR Context Diagram 4

Figure 2: SME-HR Use Case Diagram 5

Figure 3: Login Use Case Diagram....................8

Figure 4: Manage Employee Use Case Diagram 10

Figure 5: View Employee Use Case Diagram 13

Figure 6: Manage Profile Use Case Diagram....................15

Figure 7: Manage Payroll Use Case Diagram 16

Figure 8: Manage Payroll Use Case Diagram 19

Figure 9: Manage Leave Use Case Diagram....................21

Figure 10: Manage Claim Use Case Diagram 27

Figure 11: Manage EA Form Use Case Diagram 33

Figure 12: View EA Form Use Case Diagram....................36

LIST OF TABLES

Table 1: SME-HR User Characteristics 6

Table 2: Login Use Case Description 8

Table 3: Manage Employee Use Case Description...10

Table 4: View Employee Use Case Description...14

Table 5: Manage Profile Use Case Description 15

Table 6: Manage Payroll Use Case Description 16

Table 7: Manage Payroll Use Case Description 19

Table 8: Manage Leave Use Case Description 21

Table 9: Manage Claim Use Case Description 27

Table 10: Manage EA Form Use Case Description...33

Table 11: View EA Form Use Case Description 37

Table 12: Requirement Traceability for Use Case Login 39

Table 13: Requirement Traceability for Use Case Manage Employee 39

Table 14: Requirement Traceability for Use Case View Employee.......................................40

Table 15: Requirement Traceability for Use Case Manage Profile..40

Table 16: Requirement Traceability for Use Case Manage Payroll 40

Table 17: Requirement Traceability for Use Case View Payroll 41

Table 18: Requirement Traceability for Use Case Manage Leave 41

Table 19: Requirement Traceability for Use Case Manage Claim 42

Table 20: Requirement Traceability for Use Case Manage EA Form.....................................42

Table 21: Requirement Traceability for Use Case View EA Form 43

LIST OF APPENDIXES

Appendix A-1.1: Basic Flow 45

Appendix A-1.2: Alternative Flow [A1]: Forgot Password 46

Appendix A-2.1: Basic Flow 47

Appendix A-2.2: Alternative Flow [A1]: Search Employee 48

Appendix A-3.1: Basic Flow 48

Appendix A-4.1: Basic Flow 49

Appendix A-5.1: Basic Flow 50

Appendix A-5.2: Alternative [A1]: Search Payroll 51

Appendix A-6.1: Basic Flow 51

Appendix A-6.2: Alternative [A1]: Search Pay Slip..52

Appendix A-6.3: Alternative [A2]: Filter Pay Slip 52

Appendix A-7.1: Basic Flow 54

Appendix A-7.2: Alternative [A1]: Search Leave...55

Appendix A-8.1: Basic Flow 56

Appendix A-8.2: Alternative [A2]: Search Leave Application 57

Appendix A-9.1: Basic Flow 58

Appendix A-9.2: Alternative [A1]: Search Claim 59

Appendix A-10.1: Basic Flow 60

Appendix A-10.2: Alternative [A2]: Search Claim Application 61

Appendix A-11.1: Basic Flow 62

Appendix A-11.2: Alternative [A1]: Search EA Form 63

Appendix A-12.1: Basic Flow 63

Appendix A-12.2: Alternative [A1]: Search EA Form 64

Appendix A-12.3: Alternative [A2]: Filter EA Form 64

1. INTRODUCTION

1.1 PURPOSE

The purpose behind doing this Software Requirement Specification (SRS) is used to gather and analyze all assorted ideas that have come up to develop the SME-HR system. These requirements will regard consumers and will provide a detailed overview, parameters, the scope of work, and the goals of the SME-HR System. This document also describes this system's target audience and its user interface, hardware, and software requirements.

This document is likewise used to establish the basis of the details for an agreement among clients and software developers on how the software should work. It characterizes how our client, team, and audience see the item and its functionality. It gives a reference to software designers including all highlights to client requirements. In any case, it helps any designer and developer to diminish later upgrades and prevent software project failure.

1.2 SYSTEM IDENTIFICATION

This Software Requirement Specification (SRS) belongs to the "SME-HR System" (SME-HR)

System title	: SME-HR System
System abbreviation	: SME-HR
System identification number	: S01-SME-HR-2023
Version number	: V01
Release number	: 2023

The SME-HR is used to record system abbreviation, which is the abbreviation for the system name, SME-HR system which is Small and Medium-Sized Enterprises Human Resources System. At the same time, Version 1 is the format for recording the version number for this system, which is the first system, and 2023 is the format to document the release number, which is the year 2023. In addition, the S01-SME-HR-2023 format is used to record the system identification number. S01 stands for system 1, the first system. Besides that, SME-HR stands for Small and Medium-Sized Enterprises Human Resources System, which is the project system name. Meanwhile, 2023 is the release year for this system.

1.3 SYSTEM OVERVIEW

SME-HR is a system developed to manage Human Resources department activities in a systematic way. Thus, there are five modules available in the SME-HR system and are stated as below:

- Module 1: Manage Employee.
- Module 2: Manage Payroll.
- Module 3: Manage Leave.

- Module 4: Manage Claim.
- Module 5: Manage EA Form.

SME-HR system will be developed in responsive web-based system. The web-based system is for admin use only, while the mobile application system is for Malaysian Citizens and Malaysian visitors use only.

For Module 1, which is manage employee, the Administrator can create an account for each staff in the company. Besides, the Administrator can also update information about the employee detail. As for the Administrator, they can view, update, delete and add employee's information meanwhile for Staff and Manager, they can view their own information.

For Module 2, which is manage payroll, the Administrator can add the salary information for each staff. Besides that, the Administrator also can update and delete the payroll information if the information is not correct. In addition, the Administrator can generate the pay slip by month for each employee. Next for Staff and Manager, they can view their own pay slip and print it.

For Module 3, which is manage leave, the Administrator can add the leave information such as the leave entitlement, leave type. Moreover, the administrator can apply for the leave. Administrator also can update and delete all the information about the leave. Next, Staff can apply for the leave. Besides that, the Staff also can update and delete their apply leave information. In addition, the Manager can approve or reject the leave application for each employee.

For Module 4, which is manage claim. The Administrator can add the claim information such as claim type. The Administrator also can apply for the claim, and they also can edit and delete the claim information. Besides that, in this module, Staff can apply for the claim, and they also can edit and delete their claim application. In addition, for Manager, they can view all the claim application and can approve or reject the application.

For Module 5, which is manage EA form. The Administrator can add the EA form for each employee. In addition, they also can delete and update all the EA form information if there is a mistake. Besides that, in this module, the Staff and Manager is able to view their own EA form.

1.4 REFERENCES

1. SOFTWARE REQUIREMENT SPECIFICATION (SRS) FK ITEM NUMBER VERSION NUMBER (Example SDP ABC 2008 VERSION 1.0). (n.d.).
2. Custom Software Requirements Specification Document - Belitsoft. (n.d.). Retrieved November 27, 2021, from https://belitsoft.com/custom-application-development-services/software-requirements-specification-document-example-international-standard
3. Shamita, G. (2022). Implementation of Human Resources System In The Company. https://www.researchgate.net/publication/364335598

1.5 DOCUMENT OVERVIEW

This document consists of five chapters that provide a general description of the SME-HR system. Chapter 1 discusses the purpose of this report, system identification of the system, system overview between Administrator, Staff, and Manager, and references used for the SME-HR system requirements specification.

Next, the product perspective, system interface, product function, user characteristics, constraints, and assumptions and dependencies will be explained in Chapter 2. The product perspective will be using a context diagram, and the product function will be using a use case diagram of the system.

Furthermore, Chapter 3 will focus on software product features: use case description, sequence diagram, external interface requirements. In addition, the hardware interfaces and software interfaces also will be described in detail in chapter 3 under the user interface section.

Moreover, Chapter 4 will explain the requirement traceability from the SME-HR system. The last chapter, Chapter 5, will explain the acronyms and abbreviations that describe the definitions of all the terms used in the software requirement specification document, such as SME stands for Small and Medium-Sized Enterprises.

2. PRODUCT DESCRIPTION

2.1 Product Perspective

The SME-HR (Human Resources for Small and Medium-Sized Enterprises) is a responsive web-based system that can allow the user which is Human Resources department to perform specific tasks regarding human resources management. The user can access the system using their smartphone and desktop since this system is a responsive web-based system.

There are three (3) type of user, which are Administrator, Staff, and Manager. Each of the user has their own responsibility for the system. The system should allow the user to manage of the module successfully.

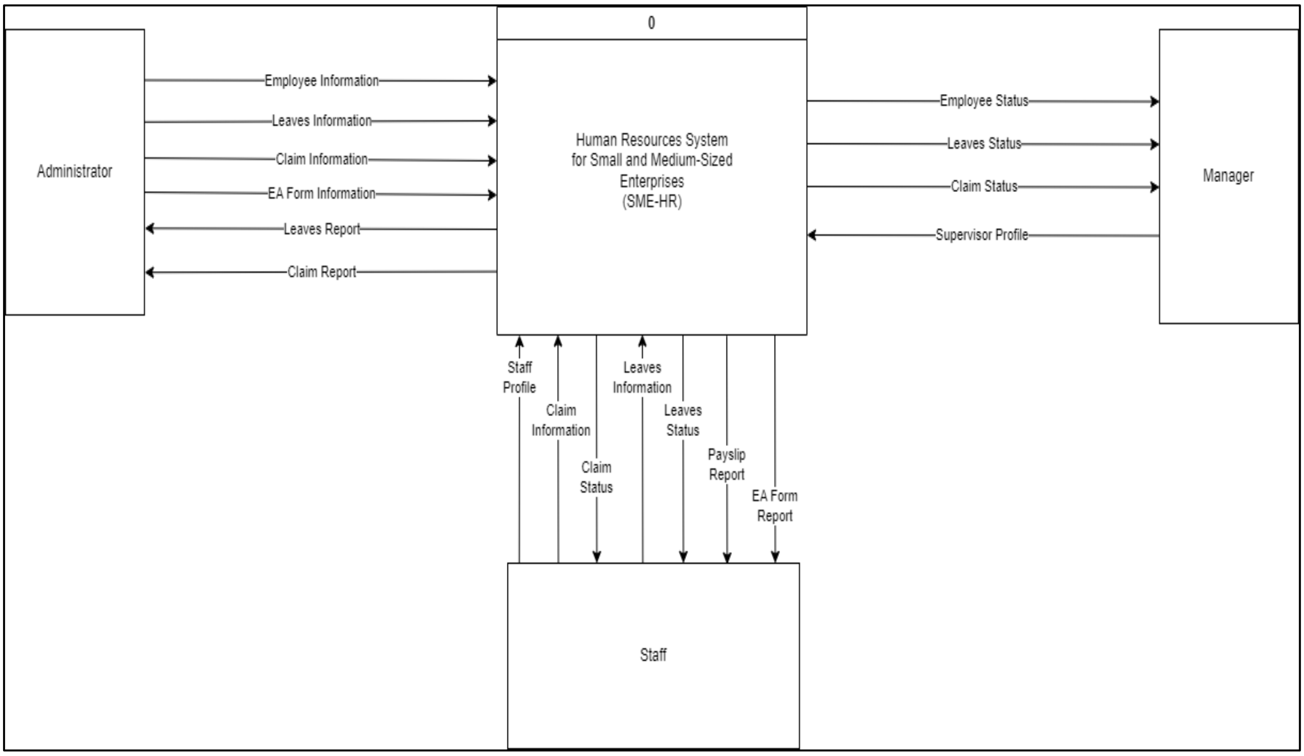


Figure 1: SME-HR Context Diagram

2.2 Product Functions

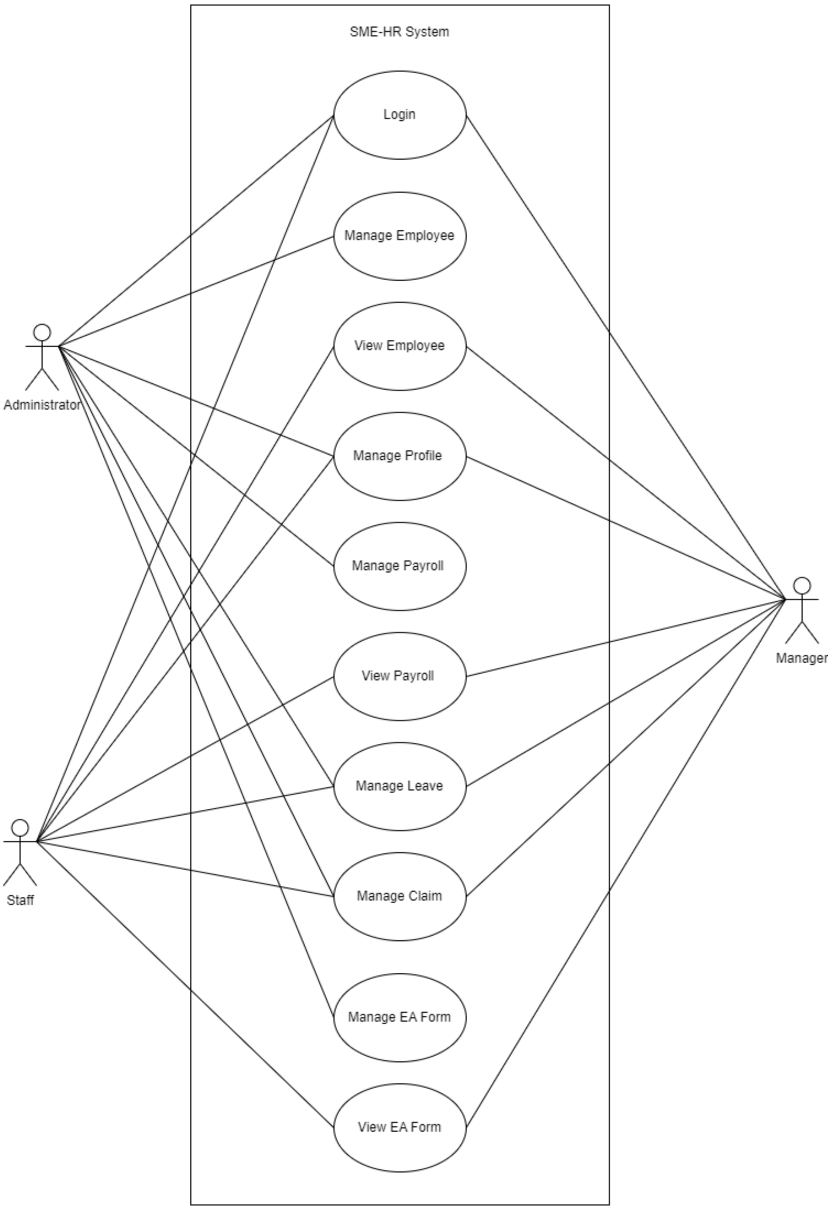


Figure 2: SME-HR Use Case Diagram

Use Case 1: Login
Use Case 2: Manage Employee
Use Case 3: View Employee
Use Case 4: Manage Profile
Use Case 5: Manage Payroll
Use Case 6: View Payroll
Use Case 7: Manage Leave
Use Case 8: Manage Claim
Use Case 9: Manage EA Form
Use Case 10: View EA Form

2.3 User Characteristics

Table 1: SME-HR User Characteristics

USER	EDUCATION LEVEL	BACKGROUND EXPERIENCE
Administrator	• Bachelor’s degree or diploma in any relevant field such as business administration, management, or human resources. • Excellent communication and interpersonal skills. • Proficiency in computer skills, particularly in the use of office software such as Microsoft Office.	• Strong organizational and time. management skills. • Have the knowledge about the finance and administrator field.
Staff	• Bachelor’s degree or equivalent. • Vocational or technical diploma. • Proficiency in computer skills, particularly in the use of office software such as Microsoft Office.	• Strong communication and interpersonal skills • Relevant work experience in the field.
Manager	• Bachelor’s degree or diploma in a relevant field such as business administration, management, finance, human resources or engineering.	• Strong leadership skills. • Strong problem-solving and decision-making abilities. • Experience in managing and leading teams.

2.4 Constraints

There are some constraints that would be considered in developing the system. The constraints that are stated here are the most important to take care of, which is:

- System compatibility
  - The system only supports Windows 7 and above only.
- Internet connectivity
  - Users need an internet connection to access the system.
- System Limitation
  - Only 10,000 users can access the system at a time.

2.6 Assumptions and Dependencies

- It is assumed that the user scope of SME-HR System will not be changed throughout the development. If it is changing, it will affect the whole Software Requirement Specification.
- It is assumed that a new function of SME-HR System could be added if it is not changing the module in Software Requirement Specification completely.
- It is assumed that the user of SME-HR system has a stable internet connection. Without a stable internet connection, the user might not be able to access the system.
- It is assumed that the login password should be strong which consists of uppercase letters, lowercase letters, numbers, and special characters. If it is not, the security of the user's account will be easily threatened.

3. SPECIFIC REQUIREMENTS

3.1 Software Product Features

3.1.1 Login

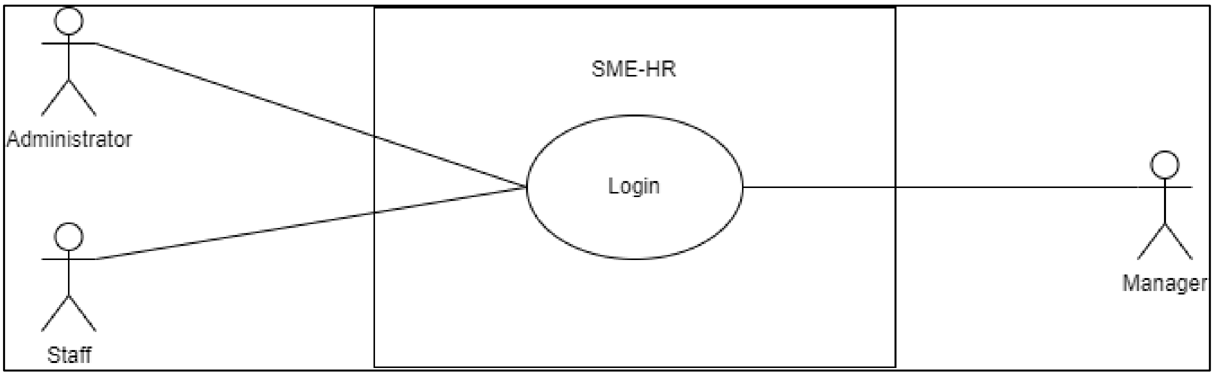


Figure 3: Login Use Case Diagram

Table 2: Login Use Case Description

Use Case ID	SME-HR-REQ-001
Brief Description	This use case is initiated by the **Administrator**, **Staff**, and **Manager.** It provides the capability to log in to the system.
Actor	Administrator, Staff, Manager
Pre-Conditions	The administrator, Staff, and Manager must have a registered email and password.
Basic Flow	**User** (**Administrator, Staff, Manager)** 1. The use case begins when the user opens the SME-HR system. 2. If the user wants to login into the system, they need to insert the email and password. 3. The user clicks on the <<Login>> button. 4. The system checks the entered email and password. **[E1: Unregistered email] [E2: Incorrect Password] [R1: Registered Email] [R2: Correct Password]** 5. The system requests the detail of the home page from the database.

	6. The system retrieves the detail of the home page from the database. 7. The system displays the home page. 8. If the user wants to log out, they need to click on the <<Logout>> button. 9. The system displays a login page. 10. The user is able to do the following functions if they forget their password: - **a) [A1: Forgot Password]** 11. The use case ends
Alternative Flow	**A1: Forgot Password [SME-HR-REQ-001-A01]** 1. The user clicks on the <<Forgot Password>> button. 2. The system displays re-enter email page. 3. The user enters the registered email. 4. The system sends the password link to the given email. 5. The user clicks on the verification link. 6. The system displays the reset password page. 7. The user enters a new password. 8. The data is sent to the database. 9. The system displays the confirmation message "Do you want to proceed?" 10. The user clicks on the <<Yes>> button. 11. The data is saved in the database. 12. The system displays a successful message. 13. The use case continues with step 2 in the basic flow.
Exception Flow	**E1: Unregistered Email [SME-HR-REQ-001-E01]** 1. The system displays an error message that asks the user to enter a registered email. 2. The use case continues with step 2 in the basic flow. **E2: Incorrect Password [SME-HR-REQ-001-E02]** 1. The system displays an error message that asks the

	user to enter the correct password. 2. The use case continues with step 2 in the basic flow.
Post-Conditions	1. The system displays the updated homepage that is specific to the type of user that is currently logged in.
Rules	**R1: Registered Email [SME-HR-REQ-001-R01]** 1. The email must be registered to the system. **R2: Correct Password [SME-HR-REQ-001-R02]** 1. The password must be correct.
Constraints	Not applicable
Sequence Diagram	Refer Appendix A-1.1: Sequence Diagram - Basic Flow A-1.2: Sequence Diagram - Alternative Flow [A1]

3.1.2 Manage Employee

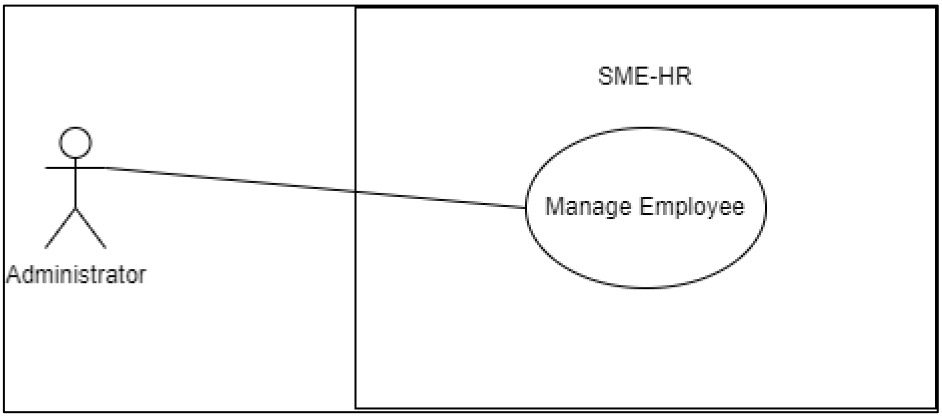


Figure 4: Manage Employee Use Case Diagram

Table 3: Manage Employee Use Case Description

Use Case ID	SME-HR-REQ-002
Brief Description	This use case is initiated by the **Administrator.** It provides the capability to manage employees.

Actor	Administrator
Pre-Conditions	The user has already login to the system.
Basic Flow	**Administrator** 1. The use case begins when the Administrator clicks on the <<Employee>> button. 2. The system requests the employee page from the database. 3. The system retrieves the employee page from the database. 4. The system displays the Employee page. 5. If the Administrator wants to add a new employee, they need to click on the <<Add>> button. 6. The system requests the add page information from the database. 7. The system retrieves the add page information from the database. 8. The system displays the Add Employee page. 9. The Administrator enters the information of the new employee. 10. The Administrator clicks on the <<Save>> button. 11. The data is sent to the database. 12. The system displays the confirmation message "Do you want to proceed?". 13. The Administrator clicks on the <<Yes>> button. 14. The data is saved in the database. 15. The system displays a successful message. 16. The Administrator is able to do the following functions if they want to view specific employee information: - **a) [A1: Search Employee]** 17. If the Administrator wants to view the detailed information of the specific employee, they need to click on the specific employee row.

	18. The system requests detailed information from the database. 19. The system retrieves detailed information about the employee from the database. 20. The system displays the Employee Detail page. 21. If the Administrator wants to update the employee detail, they need to click on the <<Update>> button. 22. The system requests the data. 23. The system retrieves the data from the database. 24. The system displays the Update Employee page. 25. The Administrator re-enters the employee information. 26. The Administrator clicks on the <<Save>> button. 27. The data is sent to the database. 28. The system displays the confirmation message "Do you want to proceed?". 29. The Administrator clicks on the <<Yes>> button. 30. The data is saved in the database. 31. The system displays a successful message. 32. If the Administrator wants to delete the employee information, they need to click on the <<Delete>> button. 33. The system displays the confirmation message "Do you want to proceed?". 34. The Administrator clicks on the <<Yes>> button. 35. The data is deleted from the database. 36. The system displays a successful message. 37. The use case end.
Alternative Flow	**A1: Search Employee [SME-HR-REQ-002-A01]** 1. The Administrator enters the keyword in the search bar. **[E1: Invalid Keyword] [R1: Valid Keyword]** 2. The Administrator clicks on the <<Search>> button. 3. The system requests the data with a similar keyword

	from the database. 4. The system retrieves the data from the database. 5. The system displays the result search. 6. The use case continues with step 17 in Administrator basic flow.
Exception Flow	**E1: Invalid Keyword [SME-HR-REQ-002-E01]** 1. The system displays an error message saying that “No Data Found!”. 2. The use case continues with step 1 in the alternative flow.
Post-Conditions	1. New employee information data appears on the Employee page. 2. The employee information is updated.
Rules	**R1: Valid Keyword [SME-HR-REQ-002-R01]** 1. The keyword must be valid and correct to search for specific information.
Constraints	Not applicable
Sequence Diagram	Refer Appendix A-2.1: Sequence Diagram - Basic Flow A-2.2: Sequence Diagram - Alternative Flow [A1]

3.1.3 View Employee

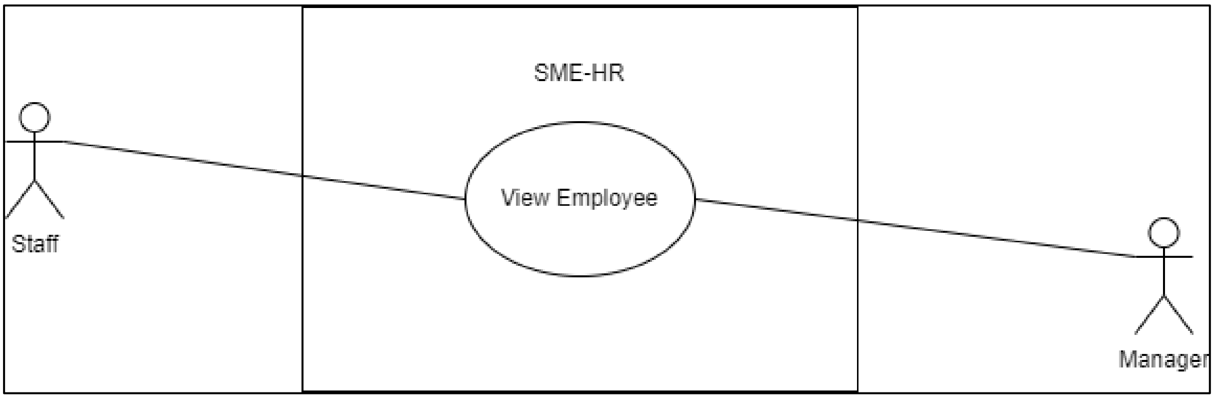


Figure 5: View Employee Use Case Diagram

Table 4: View Employee Use Case Description

Use Case ID	SME-HR-REQ-003
Brief Description	This use case is initiated by the **Staff** and **Manager.** It provides the capability to view employees.
Actor	Staff, Manager
Pre-Conditions	The user has already login to the system.
Basic Flow	**Staff, Manager** 1. The use case begins when the Staff clicks on the <<Employee>> button. 2. The system requests the employee page from the database. 3. The system retrieves the employee page from the database. 4. The system displays the Employee page. 5. If the Staff want to view their detailed information, they need to click on the information column. 6. The system requests detailed information from the database. 7. The system retrieves detailed information about the employee from the database. 8. The system displays the Employee Detail page. 9. The use case end.
Alternative Flow	Not applicable
Exception Flow	Not applicable
Post-Conditions	Not Applicable
Rules	Not Applicable
Constraints	Not applicable
Sequence Diagram	Refer Appendix A-3.1: Sequence Diagram - Basic Flow

3.1.4 Manage Profile

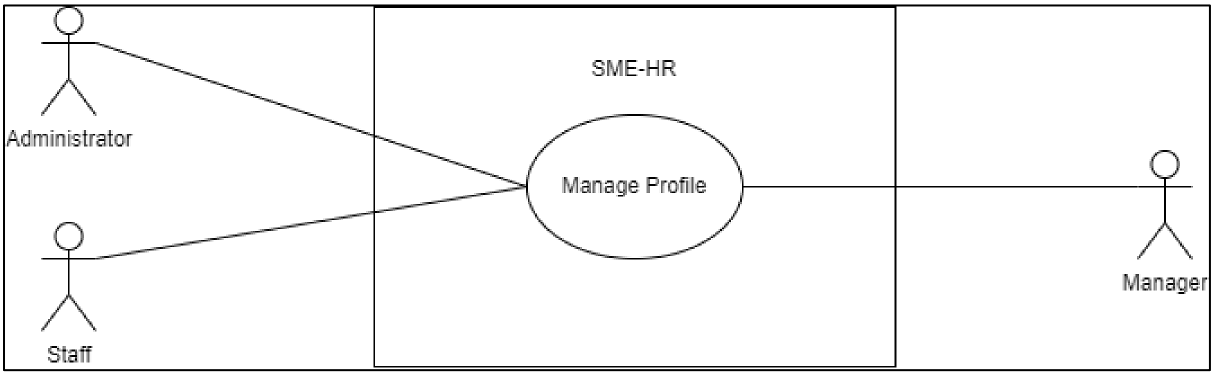


Figure 6: Manage Profile Use Case Diagram

Table 5: Manage Profile Use Case Description

Use Case ID	SME-HR-REQ-004
Brief Description	This use case is initiated by the **Administrator**, **Staff**, and **Manager.** It provides the capability to manage profiles.
Actor	Administrator, Staff, Manager
Pre-Conditions	The user has already login to the system.
Basic Flow	**User (Administrator, Staff, Manager)** 1. The use case begins when the user clicks on the <<Profile>> button. 2. The system requests the profile page from the database. 3. The system retrieves the detail of the profile page from the database. 4. The system displays all information on the Profile page. 5. If the user what to update their profile information, they can click on the <<Update>> button. 6. The system requests the detail of the update page from the database. 7. The system retrieves the detail for the update page from the database.

	8. The system displays the Update page. 9. The user re-enters the profile information. 10. The user clicks on the <<Save>> button. 11. The data is sent to the database. 12. The system displays the confirmation message "Do you want to proceed?". 13. The user clicks on the <<Yes>> button. 14. The data is saved in the database. 15. The system displays a successful message. 16. The use case end.
Alternative Flow	Not applicable
Exception Flow	Not applicable
Post-Conditions	1. The profile information is updated.
Rules	Not applicable
Constraints	Not applicable
Sequence Diagram	Refer Appendix A-4.1: Sequence Diagram - Basic Flow

3.1.5 Manage Payroll

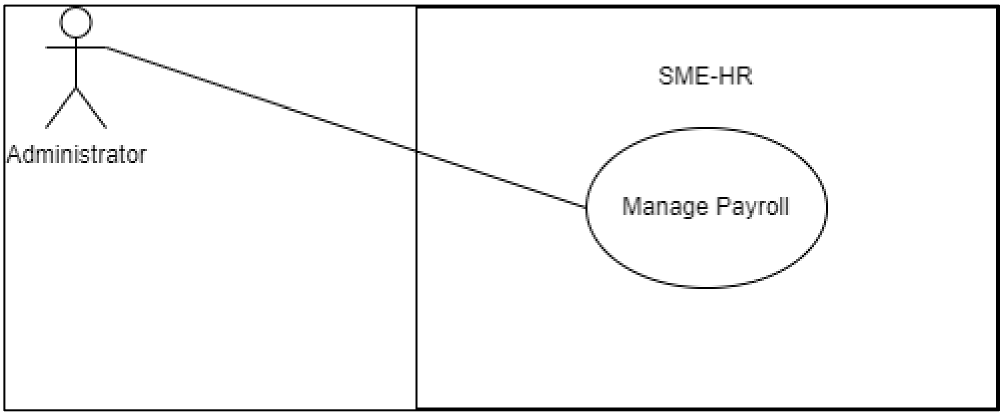


Figure 7: Manage Payroll Use Case Diagram

Table 6: Manage Payroll Use Case Description

Use Case ID	SME-HR-REQ-005

Brief Description	This use case is initiated by the **Administrator**. It provides the capability to manage payroll.
Actor	Administrator
Pre-Conditions	The user has already login to the system.
Basic Flow	**Administrator** 1. The use case begins when the Administrator clicks on the <<Payroll>> button. 2. The system requests the payroll page from the database. 3. The system retrieves the payroll page from the database. 4. The system displays the payroll page. 5. If the Administrator wants to add the salary information, they need to click on the specific employee row. 6. The system requests detailed information from the database. 7. The system retrieves detailed information from the database. 8. The system displays the salary page. 9. The Administrator enters the salary information for the employee. 10. The Administrator clicks on the <<Save>> button. 11. The data is sent to the database. 12. The system displays the confirmation message “Do you want to proceed?”. 13. The Administrator clicks on the <<Yes>> button. 14. The data is saved in the database. 15. The system displays a successful message. 16. The Administrator is able to do the following functions if they want to view specific employee salary information:- **a) [A1: Search Payroll]** 17. If the Administrator wants to view the detailed salary information of the specific employee, they need to click on the specific employee row.

	18. The system requests detailed salary information from the database. 19. The system retrieves detailed salary information from the database. 20. The system displays the Salary Information Page. 21. If the Administrator wants to update the employee salary information, they need to click on the <<Update>> button. 22. The system requests the data from the database. 23. The system retrieves the data from the database. 24. The system displays the Update page. 25. The Administrator re-enters the salary information. 26. The Administrator clicks on the <<Save>> button. 27. The data is sent to the database. 28. The system displays the confirmation message “Do you want to proceed?”. 29. The Administrator clicks on the <<Yes>> button. 30. The data is saved in the database. 31. The system displays a successful message. 32. The use case end.
Alternative Flow	**A1: Search Payroll [SME-HR-REQ-005-A01]** 1. The Administrator enters the keyword in the search bar. **[E1: Invalid Keyword] [R1: Valid Keyword]** 2. The Administrator clicks on the <<Search>> button. 3. The system requests the data with the similar keyword from the database. 4. The system retrieves the data from the database. 5. The system displays the result search. 6. The use case continues with step 17 in Administrator basic flow.
Exception Flow	**E1: Invalid Keyword [SME-HR-REQ-005-E01]** 1. The system displays an error message saying that “No Data Found!”.

	2. The use case continues with step 1 in alternative flow.
Post-Conditions	1. New employee salary information data appears in the payroll page. 2. The employee salary information is updated.
Rules	**R1: Valid Keyword [SME-HR-REQ-005-R01]** 1. The keyword must be valid and correct to search for specific information.
Constraints	Not applicable
Sequence Diagram	Refer Appendix A-5.1: Sequence Diagram - Basic Flow A-5.2: Sequence Diagram - Alternative Flow [A1]

3.1.6 View Payroll

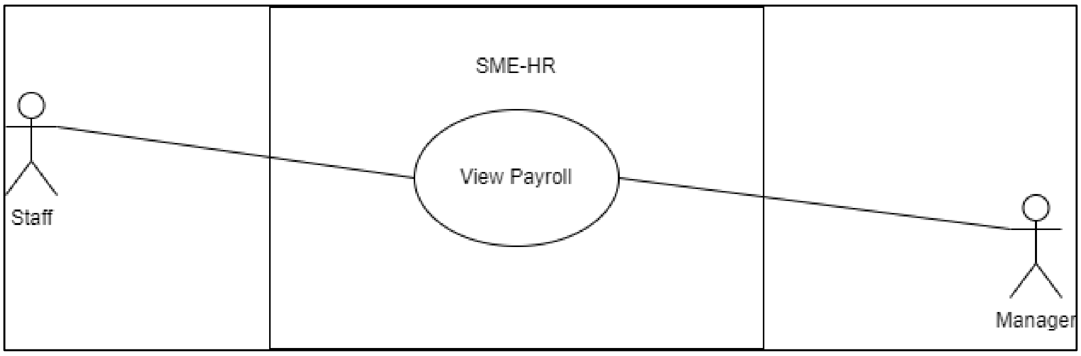


Figure 8: Manage Payroll Use Case Diagram

Table 7: Manage Payroll Use Case Description

Use Case ID	SME-HR-REQ-006
Brief Description	This use case is initiated by the **Staff and Manager**. It provides the capability to view payroll.
Actor	Staff, Manager
Pre-Conditions	The user has already login to the system.
Basic Flow	**User (Staff, Manager)** 1. The use case begins when the user clicks on the <<Payroll>> button. 2. The system requests the payroll page from the database. 3. The system retrieves the payroll page from the database.

	4. The system displays the payroll page. 5. The user is able to do the following functions if they want to filter specific information: - **a) [A1: Search Pay slip]** **b) [A2: Filter Pay slip]** 6. If the user wants to view the pay slip, they need to click on the specific pay slip row. 7. The system requests detailed payroll from the database. 8. The system retrieves detailed payroll information from the database. 9. The system displays the Pay Slip Page. 10. The use case end.
Alternative Flow	**A1: Search Pay Slip [SME-HR-REQ-006-A01]** 1. The user enters the keyword in the search bar. **[E1: Invalid Keyword] [R1: Valid Keyword]** 2. The user clicks on the <<Search>> button. 3. The system requests the data with a similar keyword from the database. 4. The system retrieves the data from the database. 5. The system displays the resulting search. 6. The use case continues with step 6 in basic flow. **A2: Filter Pay Slip [SME-HR-REQ-006-A02]** 1. The user clicks on the <<Filter>> button. 2. The user chooses to filter pay slips by year. 3. The system requests the data related with the filter keyword from the database. 4. The system retrieves the data from the database. 5. The system displays the result filter. 6. The use case continues with step 6 in basic flow.
Exception Flow	**E1: Invalid Keyword [SME-HR-REQ-006-E01]** 1. The system displays an error message saying that “No Data Found!”.

	2. The use case continues with step 1 in alternative flow.
Post-Conditions	Not applicable
Rules	**R1: Valid Keyword [SME-HR-REQ-006-R01]** 1. The keyword must be valid and correct to search for specific information.
Constraints	Not applicable
Sequence Diagram	Refer Appendix A-6.1: Sequence Diagram - Basic Flow A-6.2: Sequence Diagram - Alternative Flow [A1] A-6.3: Sequence Diagram - Alternative Flow [A2]

3.1.7 Manage Leave

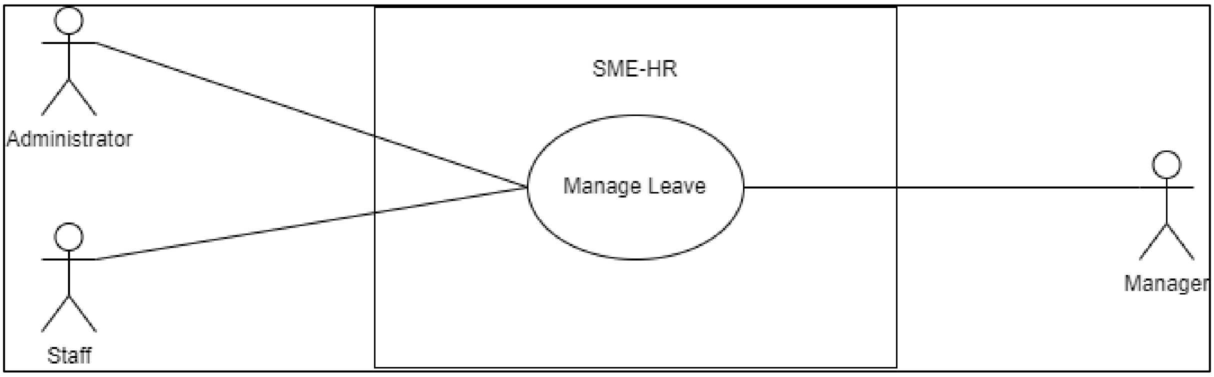


Figure 9: Manage Leave Use Case Diagram

Table 8: Manage Leave Use Case Description

Use Case ID	SME-HR-REQ-007
Brief Description	This use case is initiated by the **Administrator, Staff, and Manager**. It provides the capability to manage leave.
Actor	Administrator, Staff, Manager
Pre-Conditions	The user has already login to the system.
Basic Flow	**Administrator** 1. The use case begins when the Administrator clicks on the <<Leave>> button.

	2. The system requests the leave page from the database. 3. The system retrieves the leave page from the database. 4. The system displays the leave page. 5. If the Administrator wants to add the leave's information, they need to click on the <<Add>> button. 6. The system requests the add page information from the database. 7. The system retrieves add page information from the database. 8. The system displays the Add Leave page. 9. The Administrator enters the leave information for the employee. 10. The Administrator clicks on the <<Save>> button. 11. The data is sent to the database. 12. The system displays the confirmation message "Do you want to proceed?". 13. The Administrator clicks on the <<Yes>> button. 14. The data is saved in the database. 15. The system displays a successful message. 16. If the Administrator wants to apply for the leave, they need to click on the <<Apply>> button. 17. The system requests the apply leave page information from the database. 18. The system retrieves the apply leave page information form the database. 19. The system displays the apply leave page information form the database. 20. The Administrator enters the apply leave information. 21. The Administrator clicks on the <<Apply>> button.

	22. The data is sent to the database. 23. The system displays the confirmation message “Do you want to proceed?”. 24. The Administrator clicks on the <<Yes>> button. 25. The data is saved in the database. 26. The system displays a successful message. 27. The Administrator is able to do the following functions if they want to view specific apply leave information: - **a) [A1: Search Leave]** 28. If the Administrator wants to view the detailed leave application information of the specific employee, they need to click on the specific apply row. 29. The system requests detailed leave application information from the database. 30. The system retrieves detailed leave application information from the database. 31. The system displays the Leave Apply Information Page. 32. If the Administrator wants to update the leave information, they need to click on the <<Update>> button. 33. The system requests the data from the database. 34. The system retrieves the data from the database. 35. The system displays the Update page. 36. The Administrator re-enters the leave information. 37. The Administrator clicks on the <<Save>> button. 38. The data is sent to the database. 39. The system displays the confirmation message “Do you want to proceed?”. 40. The Administrator clicks on the <<Yes>> button. 41. The data is saved in the database. 42. The system displays a successful message.

	43. If the Administrator wants to delete the leave information, they need to click on the <<Delete>> button. 44. The system displays the confirmation message “Do you want to proceed?”. 45. The Administrator clicks on the <<Yes>> button. 46. The data is deleted from the database. 47. The system displays a successful message. 48. The use case end. **Staff, Manager** 1. The use case begins when the Staff or Manager clicks on the <<Leave>> button. 2. The system requests the leave page from the database. 3. The system retrieves the leave page from the database. 4. The system displays the leave page. 5. If the Staff or Manager wants to apply for leave, they need to click on the <<Apply>> button. 6. The system requests the apply leave page information from the database. 7. The system retrieves the apply leave page information form the database. 8. The system displays the apply leave page information form the database. 9. The Staff or Manager enters the apply leave information. 10. The Administrator clicks on the <<Apply>> button. 11. The data is sent to the database. 12. The system displays the confirmation message “Do you want to proceed?”. 13. The Staff or Manager clicks on the <<Yes>> button.

	14. The data is saved in the database. 15. The system displays a successful message. 16. The Staff or Manager is able to do the following functions if they want to view specific apply leave information: - **a) [A2: Search Leave Application]** 17. If the Staff or Manager wants to view the specific detailed leave application information, they need to click on the specific apply row. 18. The system requests detailed leave application information from the database. 19. The system retrieves detailed leave application information from the database. 20. The system displays the Leave Apply Information Page. 21. If the Staff or Manager wants to update the leave information, they need to click on the <<Update>> button. 22. The system requests the data from the database. 23. The system retrieves the data from the database. 24. The system displays the Update page. 25. The Staff or Manager re-enters the leave information. 26. The Staff or Manager clicks on the <<Save>> button. 27. The data is sent to the database. 28. The system displays the confirmation message “Do you want to proceed?”. 29. The Staff or Manager clicks on the <<Yes>> button. 30. The data is saved in the database. 31. The system displays a successful message. 32. If the Staff or Manager wants to delete the leave information, they need to click on the <<Delete>>

	button. 33. The system displays the confirmation message “Do you want to proceed?”. 34. The Staff or Manager clicks on the <<Yes>> button. 35. The data is deleted from the database. 36. The system displays a successful message. 37. The use case end.
Alternative Flow	**A1: Search Leave [SME-HR-REQ-007-A01]** 1. The Administrator enters the keyword in the search bar. **[E1: Invalid Keyword] [R1: Valid Keyword]** 2. The Administrator clicks on the <<Search>> button. 3. The system requests the data with the similar keyword from the database. 4. The system retrieves the data from the database. 5. The system displays the result search. 6. The use case continues with step 28 in Administrator basic flow. **A2: Search Leave Application [SME-HR-REQ-007-A02]** 1. The Staff or Manager enters the keyword in the search bar. 2. The Staff or Manager clicks on the <<Search>> button. 3. The system requests the data with the similar keyword from the database. 4. The system retrieves the data from the database. 5. The system displays the result search. 6. The use case continues with step 17 in Staff, Manager basic flow.
Exception Flow	**E1: Invalid Keyword [SME-HR-REQ-007-E01]** 1. The system displays an error message saying that “No Data Found!”.

	2. The use case continues with step 1 in alternative flow.
Post-Conditions	1. New employee leave information data appears in the payroll page. 2. The employee leave information is updated.
Rules	**R1: Valid Keyword [SME-HR-REQ-007-R01]** 1. The keyword must be valid and correct to search for specific information.
Constraints	Not applicable
Sequence Diagram	Refer Appendix A-7.1: Sequence Diagram - Basic Flow A-7.2: Sequence Diagram - Alternative Flow [A1] A-8.1: Sequence Diagram - Basic Flow A-8.2: Sequence Diagram - Alternative Flow [A2]

3.1.8 Manage Claim

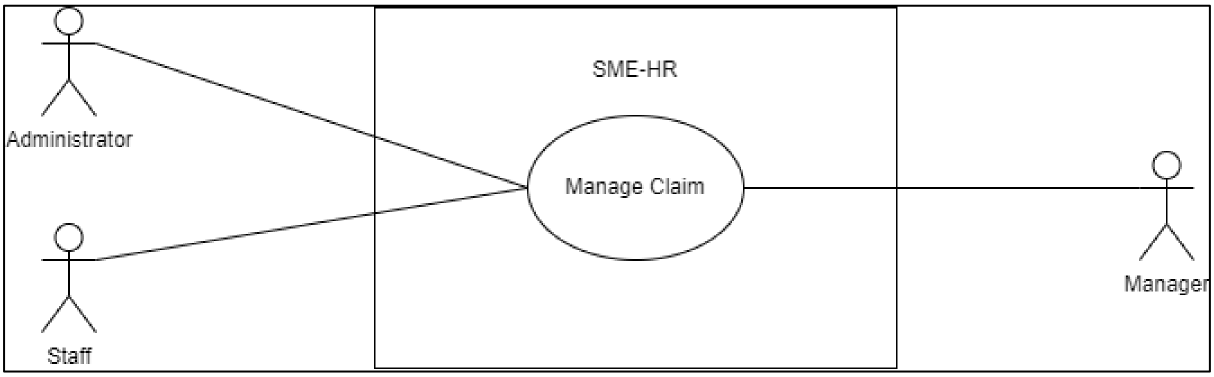


Figure 10: Manage Claim Use Case Diagram

Table 9: Manage Claim Use Case Description

Use Case ID	SME-HR-REQ-008
Brief Description	This use case is initiated by the **Administrator, Staff, and Manager**. It provides the capability to manage claim.
Actor	Administrator, Staff, Manager

Pre-Conditions	The user has already login to the system.
Basic Flow	**Administrator** 1. The use case begins when the Administrator clicks on the <<Claim>> button. 2. The system requests the claim page from the database. 3. The system retrieves the claim page from the database. 4. The system displays the Claim page. 5. If the Administrator wants to add the claim information, they need to click on the <<Add>> button. 6. The system requests the add page information from the database. 7. The system retrieves add page information from the database. 8. The system displays the Add Claim page. 9. The Administrator enters the claim information. 10. The Administrator clicks on the <<Save>> button. 11. The data is sent to the database. 12. The system displays the confirmation message “Do you want to proceed?”. 13. The Administrator clicks on the <<Yes>> button. 14. The data is saved in the database. 15. The system displays a successful message. 16. If the Administrator wants to apply for the claim, they need to click on the <<Apply>> button. 17. The system requests the apply leave page information from the database. 18. The system retrieves the apply claim page information form the database. 19. The system displays the apply claim page information form the database.

	20. The Administrator enters the apply claim information. 21. The Administrator clicks on the <<Apply>> button. 22. The data is sent to the database. 23. The system displays the confirmation message “Do you want to proceed?”. 24. The Administrator clicks on the <<Yes>> button. 25. The data is saved in the database. 26. The system displays a successful message. 27. The Administrator is able to do the following functions if they want to view specific apply claim information: - **a) [A1: Search Claim]** 28. If the Administrator wants to view the detailed claim application information of the specific employee, they need to click on the specific row. 29. The system requests detailed claim application information from the database. 30. The system retrieves detailed claim application information from the database. 31. The system displays the Claim Apply Information Page. 32. If the Administrator wants to update the claim information, they need to click on the <<Update>> button. 33. The system requests the data from the database. 34. The system retrieves the data from the database. 35. The system displays the Update page. 36. The Administrator re-enters the claim information. 37. The Administrator clicks on the <<Save>> button. 38. The data is sent to the database. 39. The system displays the confirmation message “Do you want to proceed?”.

	40. The Administrator clicks on the <<Yes>> button. 41. The data is saved in the database. 42. The system displays a successful message. 43. If the Administrator wants to delete the claim information, they need to click on the <<Delete>> button. 44. The system displays the confirmation message "Do you want to proceed?". 45. The Administrator clicks on the <<Yes>> button. 46. The data is deleted from the database. 47. The system displays a successful message. 48. The use case end. **Staff, Manager** 1. The use case begins when the Staff or Manager clicks on the <<Claim>> button. 2. The system requests the claim page from the database. 3. The system retrieves the claim page from the database. 4. The system displays the claim page. 5. If the Staff or Manager wants to apply for claim, they need to click on the <<Apply>> button. 6. The system requests the apply claim page information from the database. 7. The system retrieves the apply claim page information form the database. 8. The system displays the apply claim page information form the database. 9. The Staff or Manager enters the apply claim information. 10. The Administrator clicks on the <<Apply>> button. 11. The data is sent to the database.

	12. The system displays the confirmation message “Do you want to proceed?”. 13. The Staff or Manager clicks on the <<Yes>> button. 14. The data is saved in the database. 15. The system displays a successful message. 16. The Staff or Manager is able to do the following functions if they want to view specific apply claim information: - **a) [A2: Search Claim Application]** 17. If the Staff or Manager wants to view the specific detailed claim application information, they need to click on the specific apply row. 18. The system requests detailed claim application information from the database. 19. The system retrieves detailed claim application information from the database. 20. The system displays the Claim Apply Information Page. 21. If the Staff or Manager wants to update the claim information, they need to click on the <<Update>> button. 22. The system requests the data from the database. 23. The system retrieves the data from the database. 24. The system displays the Update page. 25. The Staff or Manager re-enters the claim information. 26. The Staff or Manager clicks on the <<Save>> button. 27. The data is sent to the database. 28. The system displays the confirmation message “Do you want to proceed?”. 29. The Staff or Manager clicks on the <<Yes>> button. 30. The data is saved in the database.

	31. The system displays a successful message. 32. If the Staff or Manager wants to delete the claim information, they need to click on the <<Delete>> button. 33. The system displays the confirmation message “Do you want to proceed?”. 34. The Staff or Manager clicks on the <<Yes>> button. 35. The data is deleted from the database. 36. The system displays a successful message. 37. The use case end.
Alternative Flow	**A1: Search Claim [SME-HR-REQ-008-A01]** 1. The Administrator enters the keyword in the search bar. **[E1: Invalid Keyword] [R1: Valid Keyword]** 2. The Administrator clicks on the <<Search>> button. 3. The system requests the data with the similar keyword from the database. 4. The system retrieves the data from the database. 5. The system displays the result search. 6. The use case continues with step 28 in Administrator basic flow. **A2: Search Claim Application [SME-HR-REQ-008-A02]** 7. The Staff or Manager enters the keyword in the search bar. 8. The Staff or Manager clicks on the <<Search>> button. 9. The system requests the data with the similar keyword from the database. 10. The system retrieves the data from the database. 11. The system displays the result search. 12. The use case continues with step 17 in Staff, Manager basic flow.

Exception Flow	**E1: Invalid Keyword [SME-HR-REQ-008-E01]** 1. The system displays an error message saying that “No Data Found!”. 2. The use case continues with step 1 in alternative flow.
Post-Conditions	1. New employee claim information data appears in the claim page. 2. The employee claim information is updated.
Rules	**R1: Valid Keyword [SME-HR-REQ-008-R01]** 1. The keyword must be valid and correct to search for specific information.
Constraints	Not applicable
Sequence Diagram	Refer Appendix A-9.1: Sequence Diagram - Basic Flow A-9.2: Sequence Diagram - Alternative Flow [A1] A-10.1: Sequence Diagram - Basic Flow A-10.2: Sequence Diagram - Alternative Flow [A2]

3.1.9 Manage EA Form

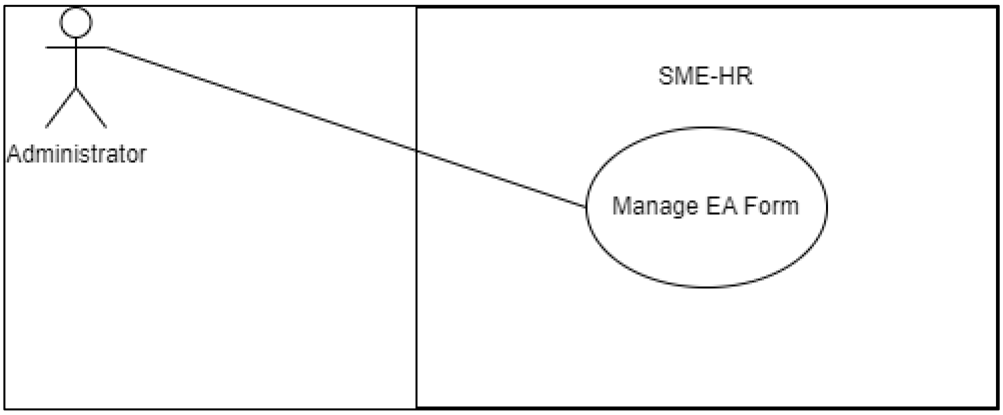


Figure 11: Manage EA Form Use Case Diagram

Table 10: Manage EA Form Use Case Description

Use Case ID	SME-HR-REQ-009

Brief Description	This use case is initiated by the **Administrator**. It provides the capability to manage EA form.
Actor	Administrator
Pre-Conditions	The user has already login to the system.
Basic Flow	**Administrator** 1. The use case begins when the Administrator clicks on the <<EA Form>> button. 2. The system requests the EA page from the database. 3. The system retrieves the EA page from the database. 4. The system displays the EA page. 5. If the Administrator wants to add the EA information, they need to click on the <<Add>> button. 6. The system requests the add page information from the database. 7. The system retrieves add page information from the database. 8. The system displays the Add EA page. 9. The Administrator enters the EA information. 10. The Administrator clicks on the <<Save>> button. 11. The data is sent to the database. 12. The system displays the confirmation message "Do you want to proceed?". 13. The Administrator clicks on the <<Yes>> button. 14. The data is saved in the database. 15. The system displays a successful message. 16. The Administrator is able to do the following functions if they want to view specific EA information: - **a) [A1: Search EA Form]** 17. If the Administrator wants to view the detailed EA form information of the specific employee, they

	need to click on the specific row. 18. The system requests detailed EA form information from the database. 19. The system retrieves detailed EA form information from the database. 20. The system displays the EA From Information Page. 21. If the Administrator wants to update the EA form information, they need to click on the <<Update>> button. 22. The system requests the data from the database. 23. The system retrieves the data from the database. 24. The system displays the Update page. 25. The Administrator re-enters the EA form information. 26. The Administrator clicks on the <<Save>> button. 27. The data is sent to the database. 28. The system displays the confirmation message "Do you want to proceed?". 29. The Administrator clicks on the <<Yes>> button. 30. The data is saved in the database. 31. The system displays a successful message. 32. If the Administrator wants to delete the EA form information, they need to click on the <<Delete>> button. 33. The system displays the confirmation message "Do you want to proceed?". 34. The Administrator clicks on the <<Yes>> button. 35. The data is deleted from the database. 36. The system displays a successful message. 37. The use case end.
Alternative Flow	**A1: Search EA Form [SME-HR-REQ-009-A01]** 1. The Administrator enters the keyword in the search bar. **[E1: Invalid Keyword] [R1: Valid Keyword]**

	2. The Administrator clicks on the <<Search>> button. 3. The system requests the data with the similar keyword from the database. 4. The system retrieves the data from the database. 5. The system displays the result search. 6. The use case continues with step 17 in Administrator basic flow.
Exception Flow	**E1: Invalid Keyword [SME-HR-REQ-009-E01]** 1. The system displays an error message saying that “No Data Found!”. 2. The use case continues with step 1 in alternative flow.
Post-Conditions	1. New employee EA form information data appears in the claim page. 2. The employee EA form information is updated.
Rules	**R1: Valid Keyword [SME-HR-REQ-009-R01]** 1. The keyword must be valid and correct to search for specific information.
Constraints	Not applicable
Sequence Diagram	Refer Appendix A-11.1: Sequence Diagram - Basic Flow A-11.2: Sequence Diagram - Alternative Flow [A1]

3.1.10 View EA Form

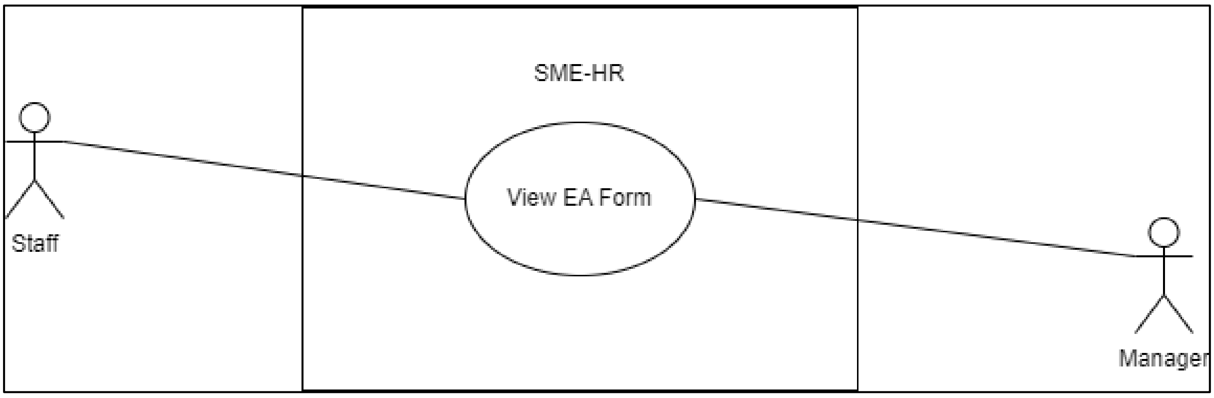


Figure 12: View EA Form Use Case Diagram

Table 11: View EA Form Use Case Description

Use Case ID	SME-HR-REQ-010
Brief Description	This use case is initiated by the **Staff and Manager**. It provides the capability to view EA form.
Actor	Staff, Manager
Pre-Conditions	The user has already login to the system.
Basic Flow	**User (Staff, Manager)** 1. The use case begins when the user clicks on the <<EA Form>> button. 2. The system requests the EA form page from the database. 3. The system retrieves the EA form page from the database. 4. The system displays the EA form page. 5. The user is able to do the following functions if they want to filter specific information: - **a) [A1: Search EA Form]** **b) [A2: Filter EA form]** 6. If the user wants to view the EA form, they need to click on the specific EA form row. 7. The system requests detailed EA form from the database. 8. The system retrieves detailed EA form information from the database. 9. The system displays the EA Form Page. 10. The use case end.
Alternative Flow	**A1: Search EA Form [SME-HR-REQ-010-A01]** 1. The user enters the keyword in the search bar. **[E1: Invalid Keyword] [R1: Valid Keyword]** 2. The user clicks on the <<Search>> button. 3. The system requests the data with a similar keyword from the database.

	4. The system retrieves the data from the database. 5. The system displays the resulting search. 6. The use case continues with step 6 in basic flow. **A2: Filter EA Form [SME-HR-REQ-010-A02]** 1. The user clicks on the <<Filter>> button. 2. The user chooses to filter EA form by year. 3. The system requests the data related with the filter keyword from the database. 4. The system retrieves the data from the database. 5. The system displays the result filter. 6. The use case continues with step 6 in basic flow.
Exception Flow	**E1: Invalid Keyword [SME-HR-REQ-010-E01]** 1. The system displays an error message saying that “No Data Found!”. 2. The use case continues with step 1 in alternative flow.
Post-Conditions	Not applicable
Rules	**R1: Valid Keyword [SME-HR-REQ-010-R01]** 1. The keyword must be valid and correct to search for specific information.
Constraints	Not applicable
Sequence Diagram	Refer Appendix A-12.1: Sequence Diagram - Basic Flow A-12.2: Sequence Diagram - Alternative Flow [A1] A-12.3: Sequence Diagram - Alternative Flow [A2]

4. REQUIREMENT TRACEABILITY

Table 12: Requirement Traceability for Use Case Login

Requirements	***Description***
SME-HR-REQ-001	Login Provides Administrator, Staff, and Manager the ability to log in to the system.
SME-HR-REQ-001-A01	Forgot Password Administrator, Staff, and Manager click on the <<Forgot Password>> button to reset button.
SME-HR-REQ-001-E01	Unregistered Email The entered email to log in is not registered to the system.
SME-HR-REQ-001-E02	Incorrect Password The entered password to log in is incorrect.
SME-HR-REQ-001-R01	Registered Email The entered username or email to log in must be registered to the system.
SME-HR-REQ-001-R02	Correct Password The entered password must be correct to log in to the system.

Table 13: Requirement Traceability for Use Case Manage Employee

Requirements	***Description***
SME-HR-REQ-002	Manage Employee Provides Administrator the ability to manage employee.
SME-HR-REQ-002-A01	Search Employee Provides Administrator the ability to search with keyword on employee list page.
SME-HR-REQ-002-E01	Invalid Keyword The entered keyword is not valid.

SME-HR-REQ-001-R01	Valid Keyword The entered keyword to search the employee must be valid.

Table 14: Requirement Traceability for Use Case View Employee

Requirements	***Description***
SME-HR-REQ-003	View Employee Provides Staff and Manager the ability to view employee information.

Table 15: Requirement Traceability for Use Case Manage Profile

Requirements	***Description***
SME-HR-REQ-004	Manage Profile Provides Administrator, Staff, and Manager the ability to manage profile.

Table 16: Requirement Traceability for Use Case Manage Payroll

Requirements	***Description***
SME-HR-REQ-005	Manage Payroll Provides Administrator the ability to manage payroll.
SME-HR-REQ-005-A01	Search Payroll Provides Administrator the ability to search with keyword on payroll list page.
SME-HR-REQ-005-E01	Invalid Keyword The entered keyword is not valid.
SME-HR-REQ-005-R01	Valid Keyword The entered keyword to search the payroll must be valid.

Table 17: Requirement Traceability for Use Case View Payroll

Requirements	***Description***
SME-HR-REQ-006	View Payroll Provides Staff and Manager the ability to view payroll.
SME-HR-REQ-006-A01	Search Pay Slip Provides Staff and Manager the ability to search with keyword on payroll list page.
SME-HR-REQ-006-A02	Filter Pay Slip Provides Staff and Manager the ability to filter the payroll list.
SME-HR-REQ-006-E01	Invalid Keyword The entered keyword is not valid.
SME-HR-REQ-005-R01	Valid Keyword The entered keyword to search the pay slip must be valid.

Table 18: Requirement Traceability for Use Case Manage Leave

Requirements	***Description***
SME-HR-REQ-007	Manage Leave Provides Administrator, Staff and Manager the ability to manage leave.
SME-HR-REQ-007-A01	Search Leave Provides Administrator the ability to search with keyword on leave list page.
SME-HR-REQ-007-A02	Search Leave Application Provides Staff and Manager the ability to search with keyword on the leave application page.
SME-HR-REQ-007-E01	Invalid Keyword The entered keyword is not valid.
SME-HR-REQ-007-R01	Valid Keyword The entered keyword to search the leave must be valid.

Table 19: Requirement Traceability for Use Case Manage Claim

Requirements	***Description***
SME-HR-REQ-008	Manage Claim Provides Administrator, Staff and Manager the ability to manage claim.
SME-HR-REQ-008-A01	Search Claim Provides Administrator the ability to search with keyword on claim list page.
SME-HR-REQ-008-A02	Search Claim Application Provides Staff and Manager the ability to search with keyword on the claim application page.
SME-HR-REQ-008-E01	Invalid Keyword The entered keyword is not valid.
SME-HR-REQ-008-R01	Valid Keyword The entered keyword to search the claim must be valid.

Table 20: Requirement Traceability for Use Case Manage EA Form

Requirements	***Description***
SME-HR-REQ-009	Manage EA Form Provides Administrator the ability to manage EA form.
SME-HR-REQ-009-A01	Search EA Form Provides Administrator the ability to search with keyword on EA form list page.
SME-HR-REQ-009-E01	Invalid Keyword The entered keyword is not valid.
SME-HR-REQ-009-R01	Valid Keyword The entered keyword to search the EA form must be valid.

Table 21: Requirement Traceability for Use Case View EA Form

Requirements	***Description***
SME-HR-REQ-010	View EA Form Provides Staff and Manager the ability to view EA form.
SME-HR-REQ-010-A01	Search EA Form Provides Administrator the ability to search with keyword on EA form list page.
SME-HR-REQ-010-A02	Filter EA Form Provides Administrator the ability to filter the EA form list.
SME-HR-REQ-010-E01	Invalid Keyword The entered keyword is not valid.
SME-HR-REQ-010-R01	Valid Keyword The entered keyword to search the EA form must be valid.

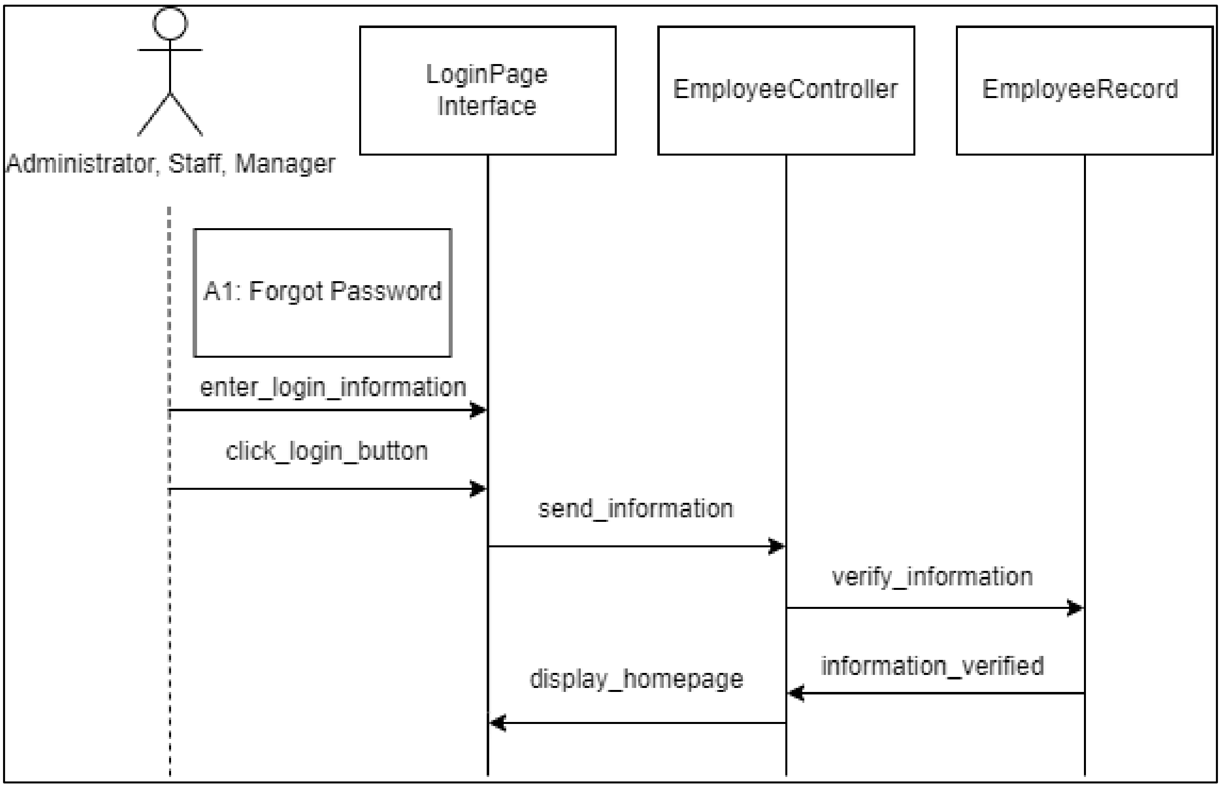
5. ACRONYMS AND ABBREVIATION

SME-HR	Small and Medium-Sized Enterprises Human Resources System
SRS	Software Requirement Specification
SME-HR-REQ-xx	SME-HR System-Requirement-xx
SME-HR-REQ-xx-Axx	SME-HR System-Requirement-xx-Alternative xx
SME-HR-REQ-xx-Exx	SME-HR System-Requirement-xx-Exception xx
SME-HR-REQ-xx-Rxx	SME-HR System-Requirement-xx-Rules xx
XX	Referring to the Specific Use Case, Alternative, Exception Rules Number

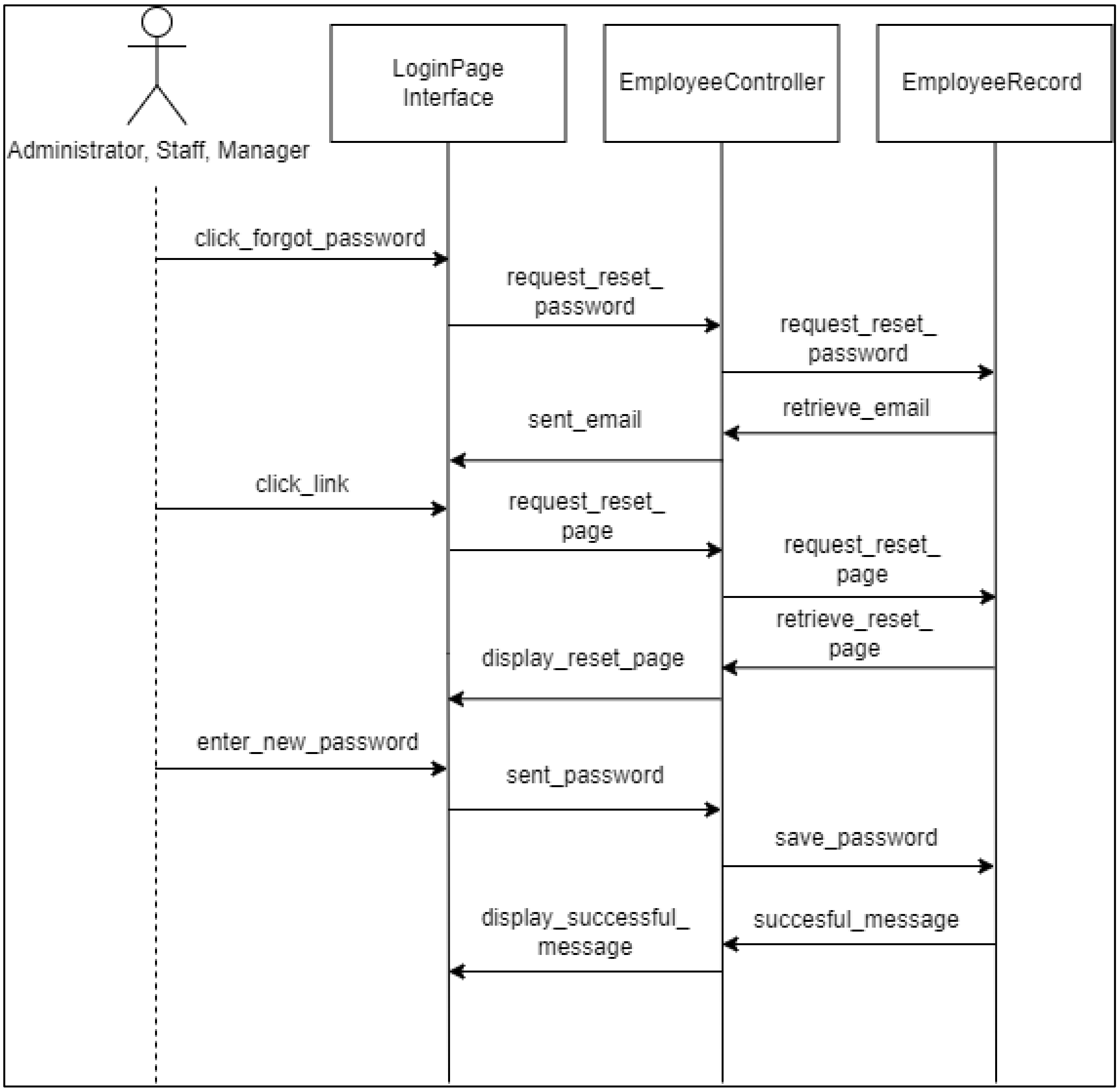
APPENDIX A

Sequence Diagram

Appendix A-1: Login Sequence Diagram (Administrator/Staff/Manager)

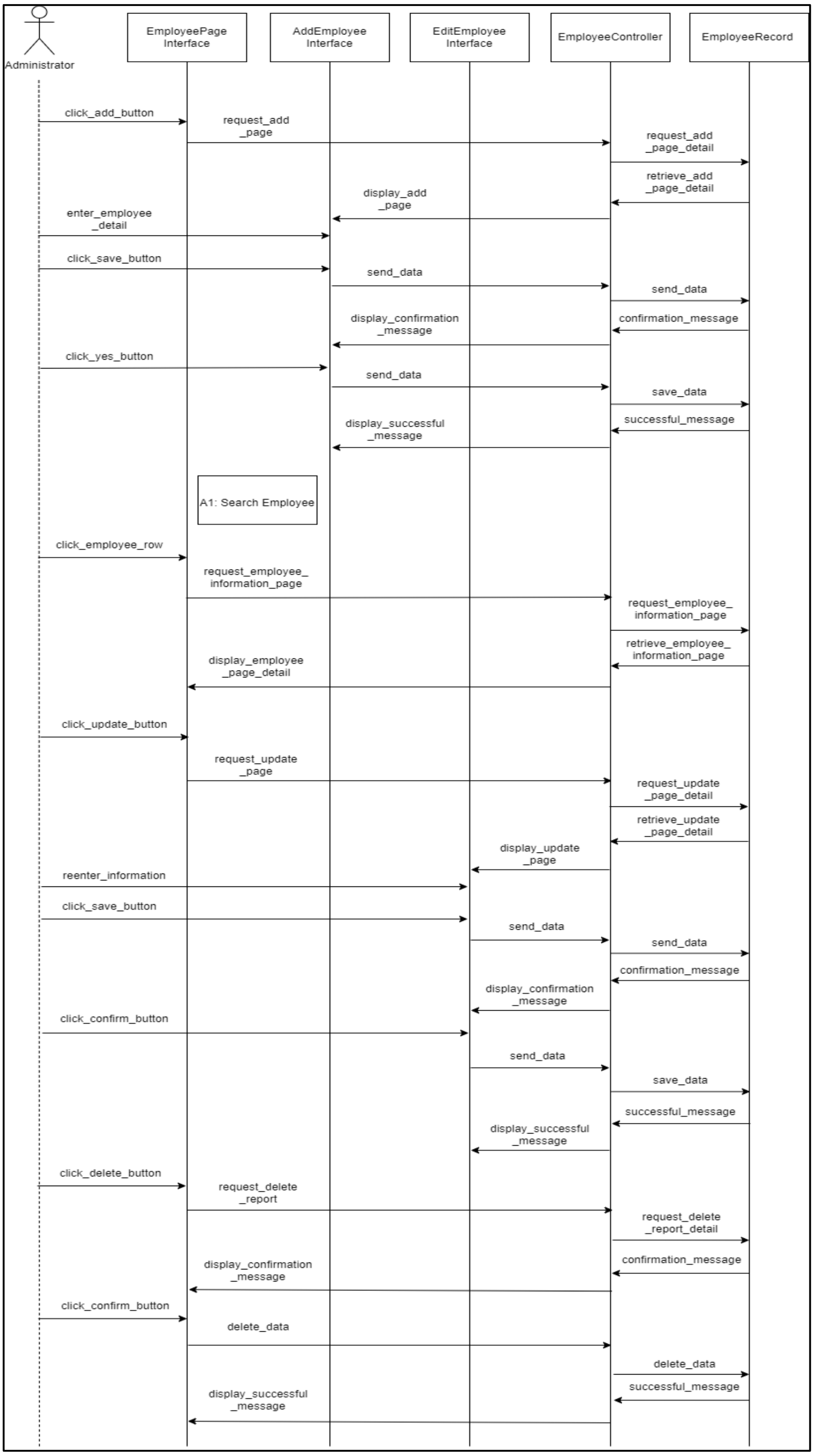


Appendix A-1.1: Basic Flow

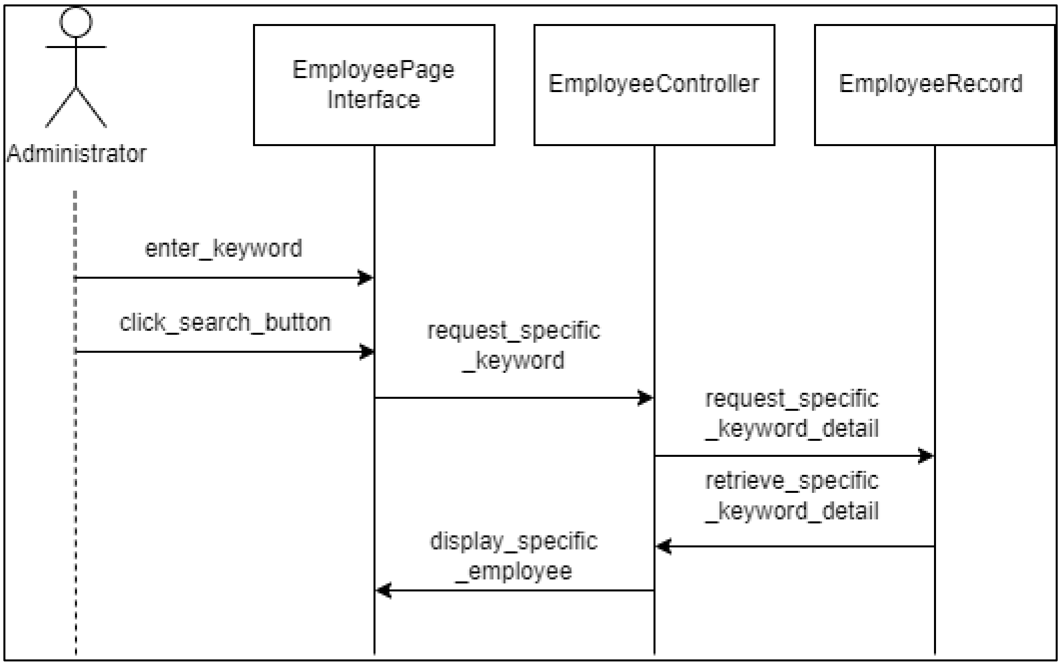


Appendix A-1.2: Alternative Flow [A1]: Forgot Password

Appendix A-2: Manage Employee Sequence Diagram (Administrator)

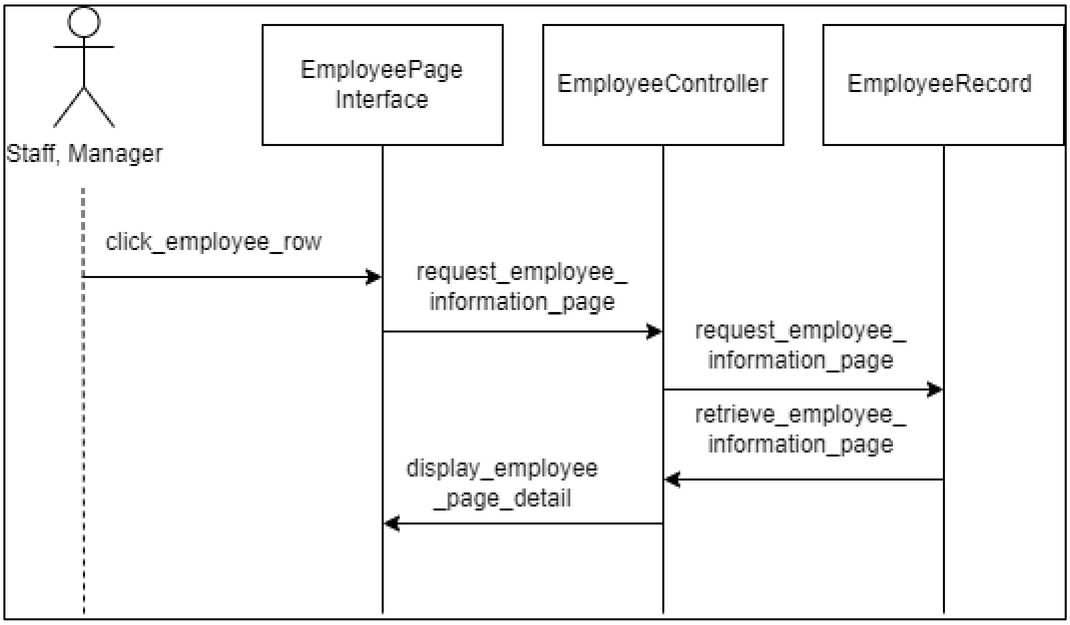


Appendix A-2.1: Basic Flow


Administrator
EmployeePage Interface
EmployeeController
EmployeeRecord
enter_keyword
click_search_button
request_specific _keyword
request_specific _keyword_detail
retrieve_specific _keyword_detail
display_specific _employee

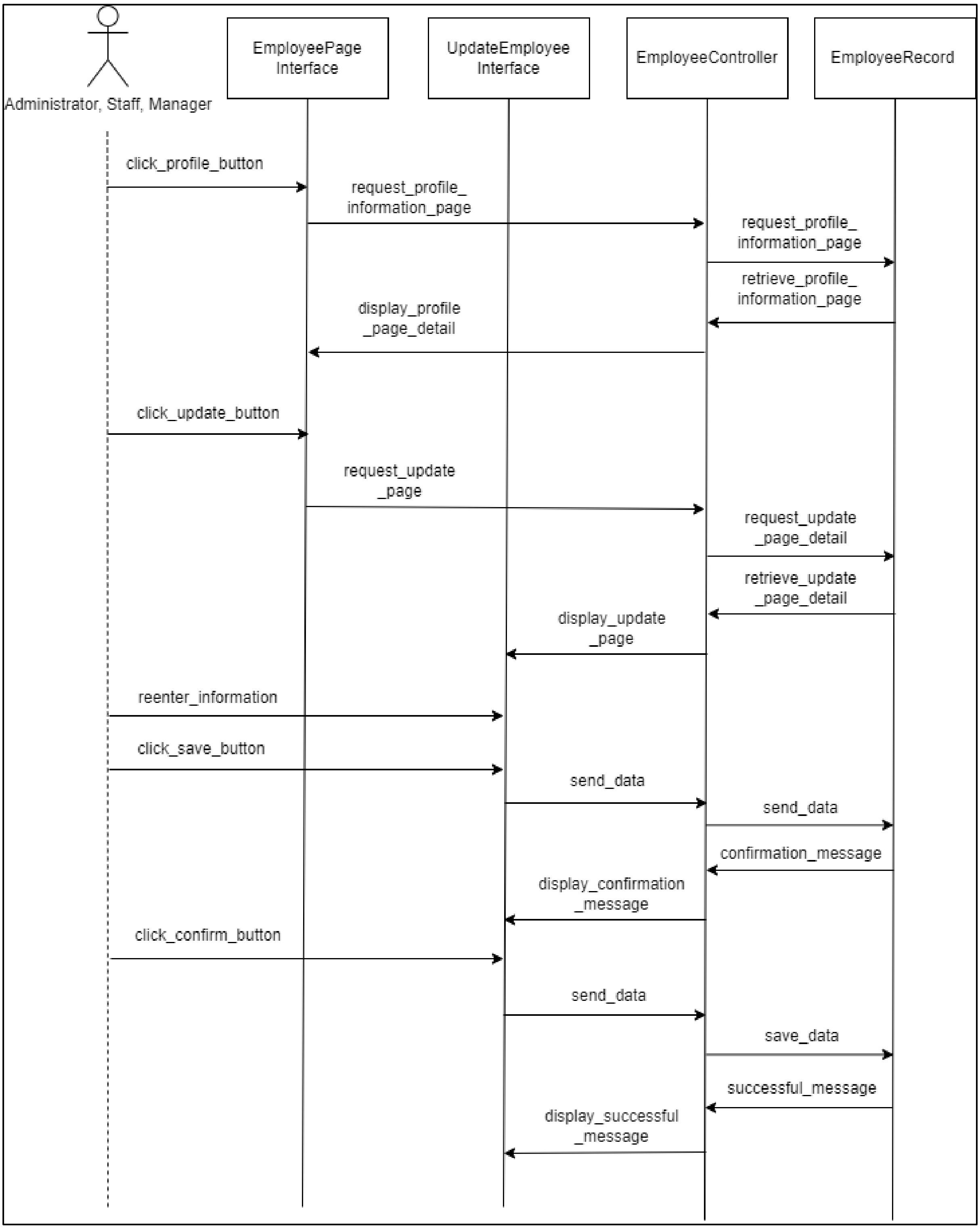
Appendix A-2.2: Alternative Flow [A1]: Search Employee

Appendix A-3: View Employee Sequence Diagram (Staff/Manager)


Staff, Manager
EmployeePage Interface
EmployeeController
EmployeeRecord
click_employee_row
request_employee_ information_page
request_employee_ information_page
retrieve_employee_ information_page
display_employee _page_detail

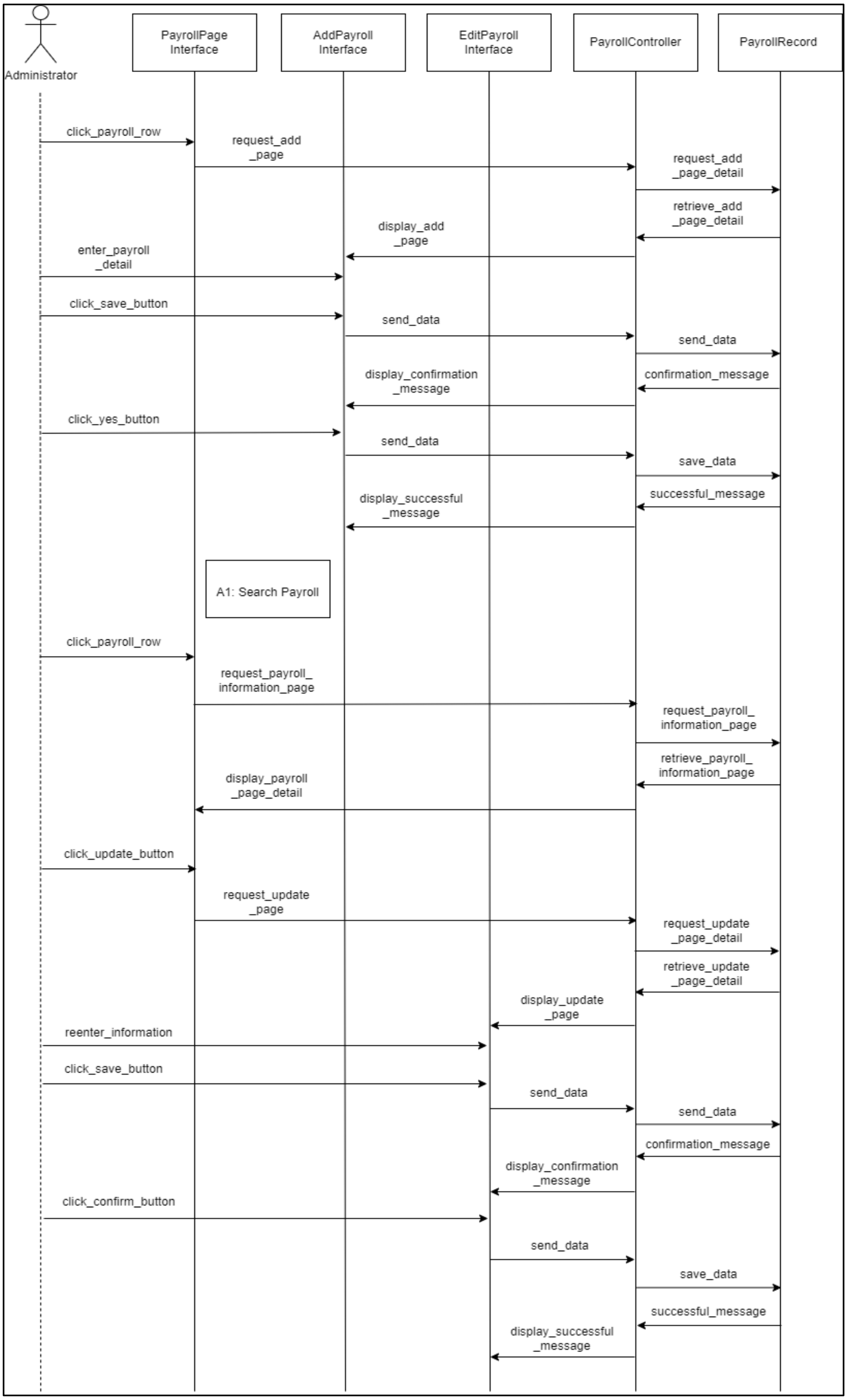
Appendix A-3.1: Basic Flow

Appendix A-4: Manage Profile Sequence Diagram (Administrator/Staff/Manager)

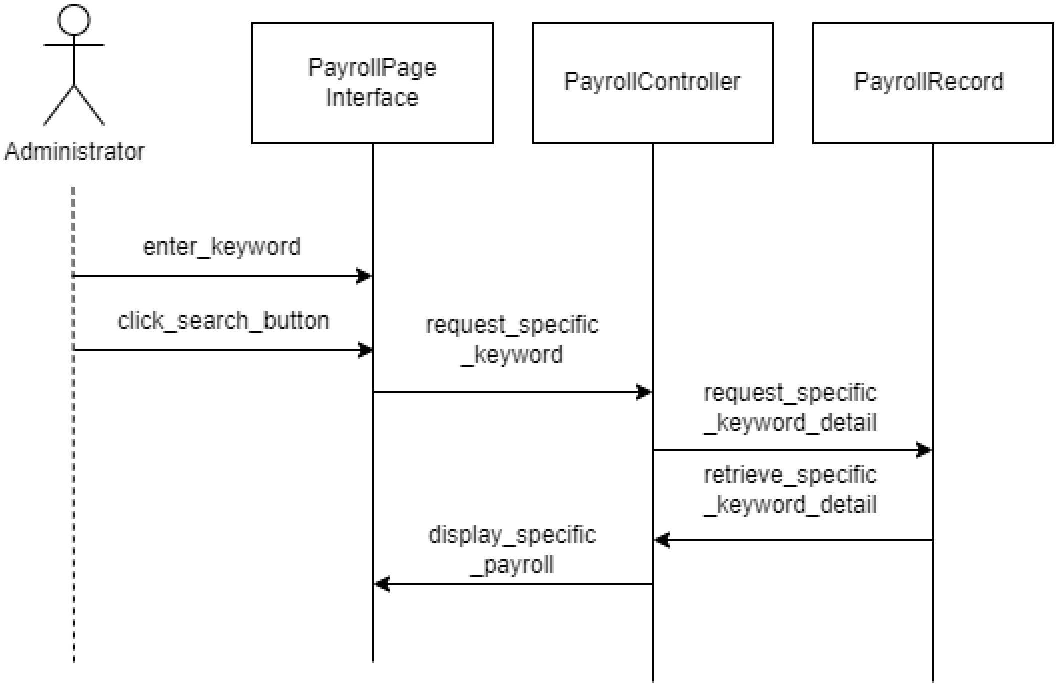


Appendix A-4.1: Basic Flow

Appendix A-5: Manage Payroll Sequence Diagram (Administrator)

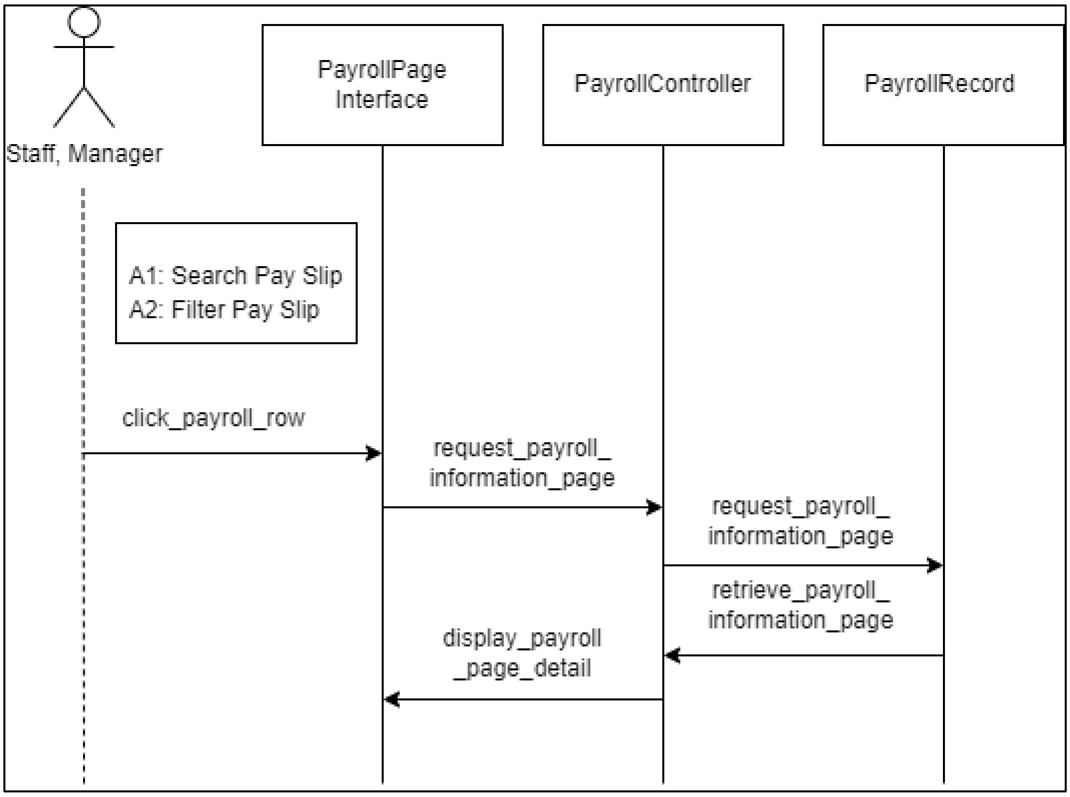


Appendix A-5.1: Basic Flow

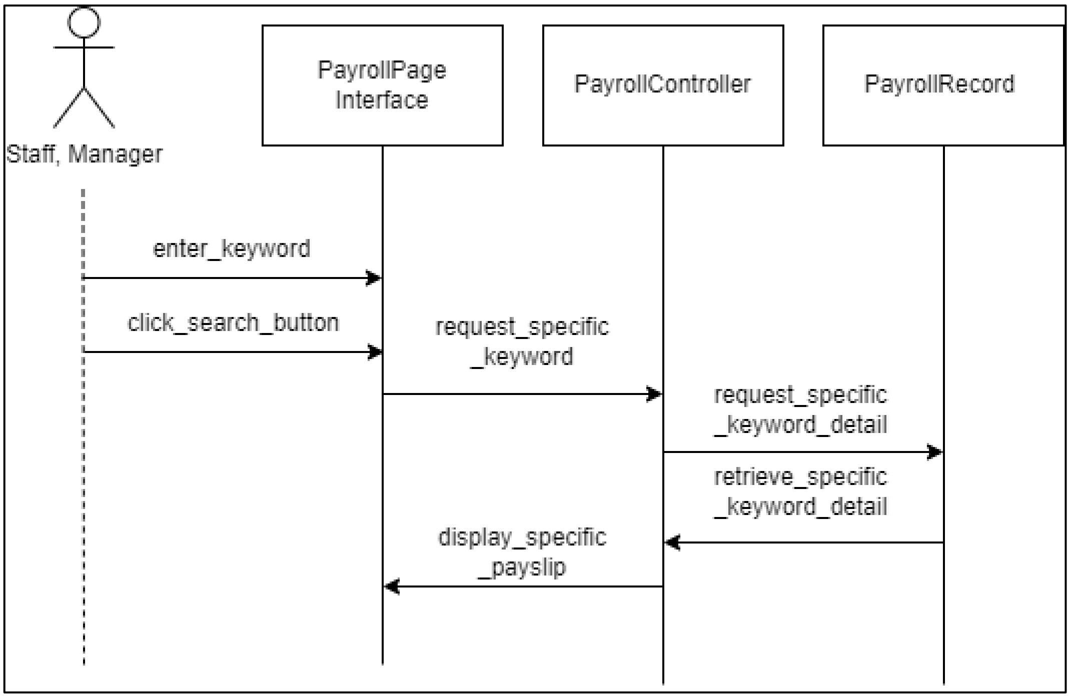

Administrator
PayrollPage Interface
PayrollController
PayrollRecord
enter_keyword
click_search_button
request_specific _keyword
request_specific _keyword_detail
retrieve_specific _keyword_detail
display_specific _payroll

Appendix A-5.2: Alternative [A1]: Search Payroll

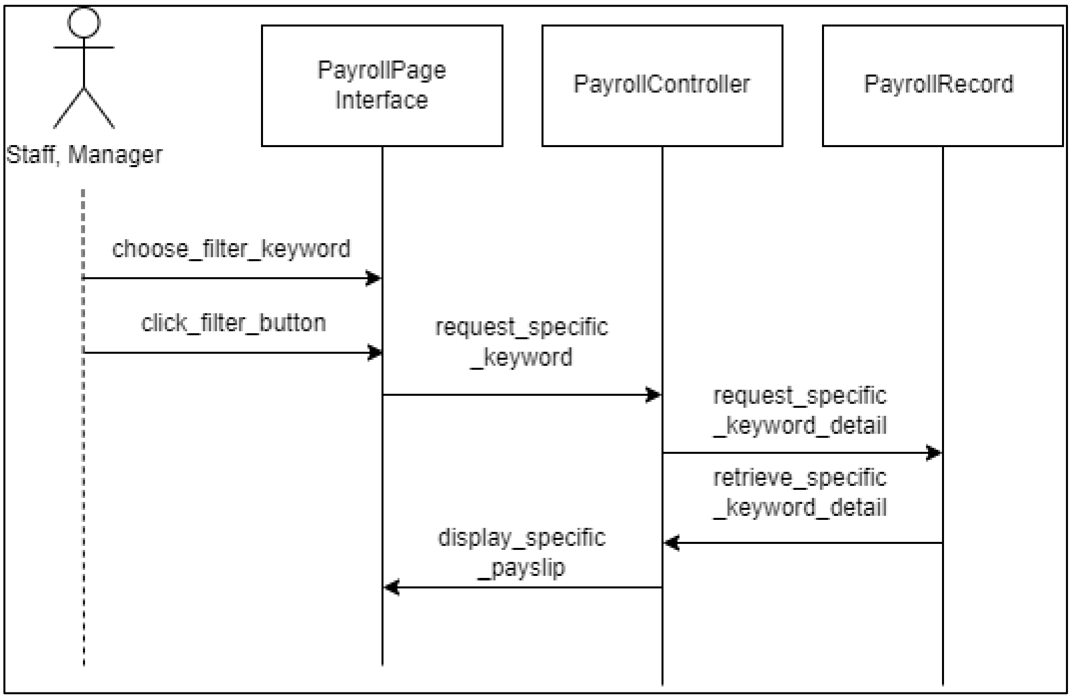
Appendix A-6: View Payroll Sequence Diagram (Staff/Manager)


Staff, Manager
PayrollPage Interface
PayrollController
PayrollRecord
A1: Search Pay Slip
A2: Filter Pay Slip
click_payroll_row
request_payroll_ information_page
request_payroll_ information_page
retrieve_payroll_ information_page
display_payroll _page_detail

Appendix A-6.1: Basic Flow

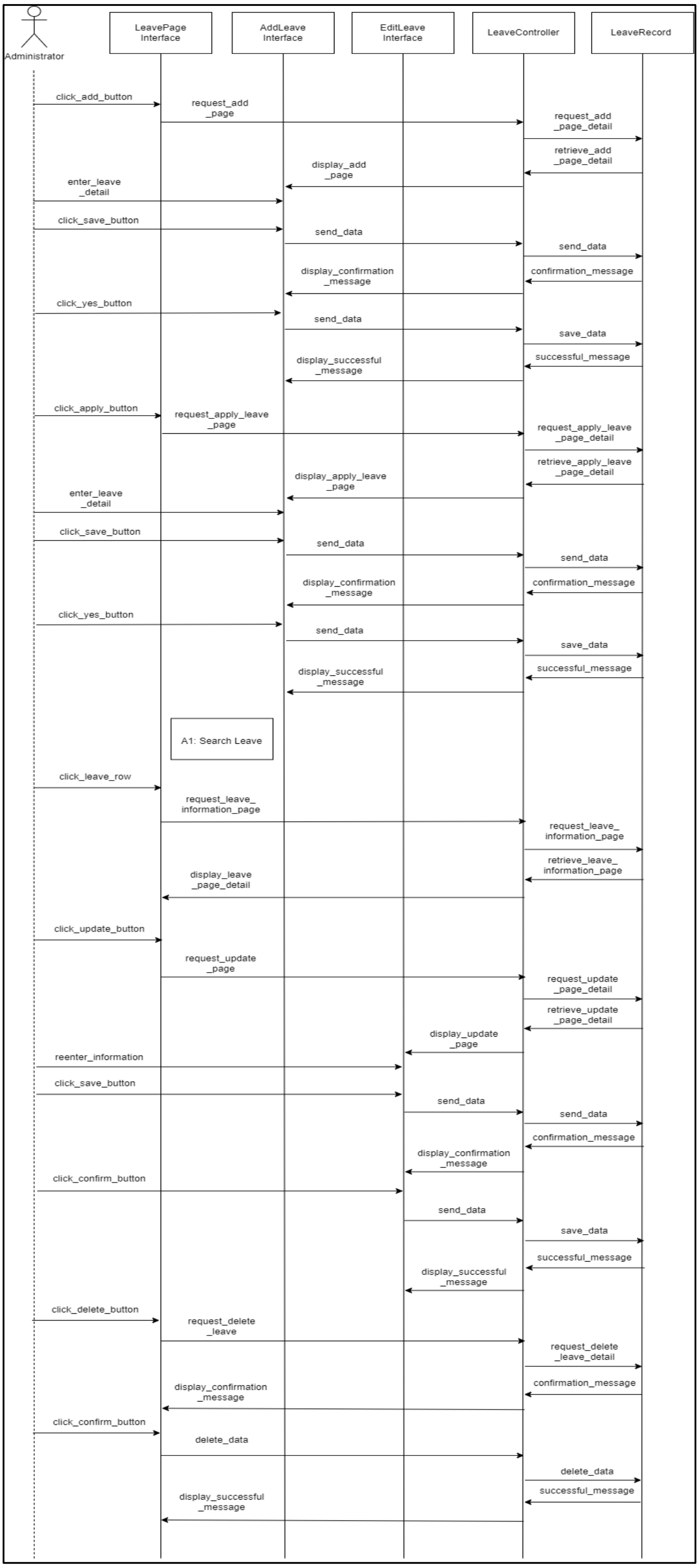

Staff, Manager
PayrollPage Interface
PayrollController
PayrollRecord
enter_keyword
click_search_button
request_specific _keyword
request_specific _keyword_detail
retrieve_specific _keyword_detail
display_specific _payslip

Appendix A-6.2: Alternative [A1]: Search Pay Slip

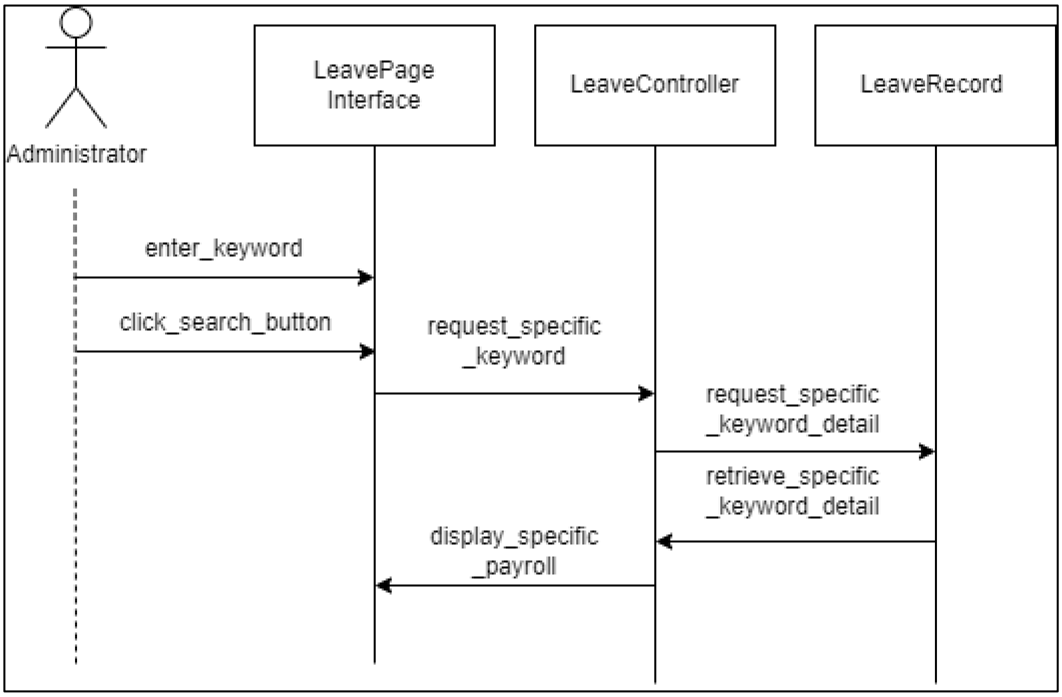

Staff, Manager
PayrollPage Interface
PayrollController
PayrollRecord
choose_filter_keyword
click_filter_button
request_specific _keyword
request_specific _keyword_detail
retrieve_specific _keyword_detail
display_specific _payslip

Appendix A-6.3: Alternative [A2]: Filter Pay Slip

Appendix A-7: Manage Leave Sequence Diagram (Administrator)

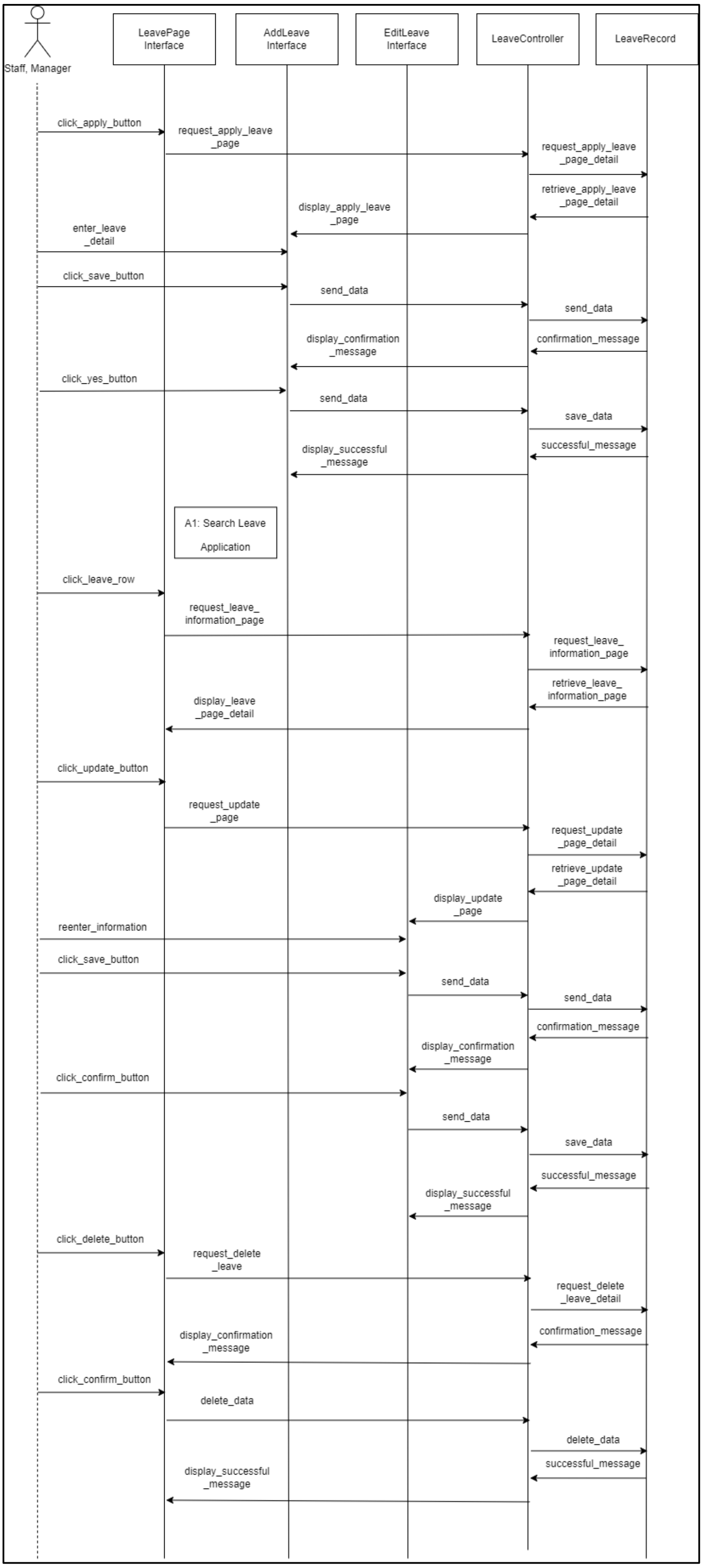


Appendix A-7.1: Basic Flow

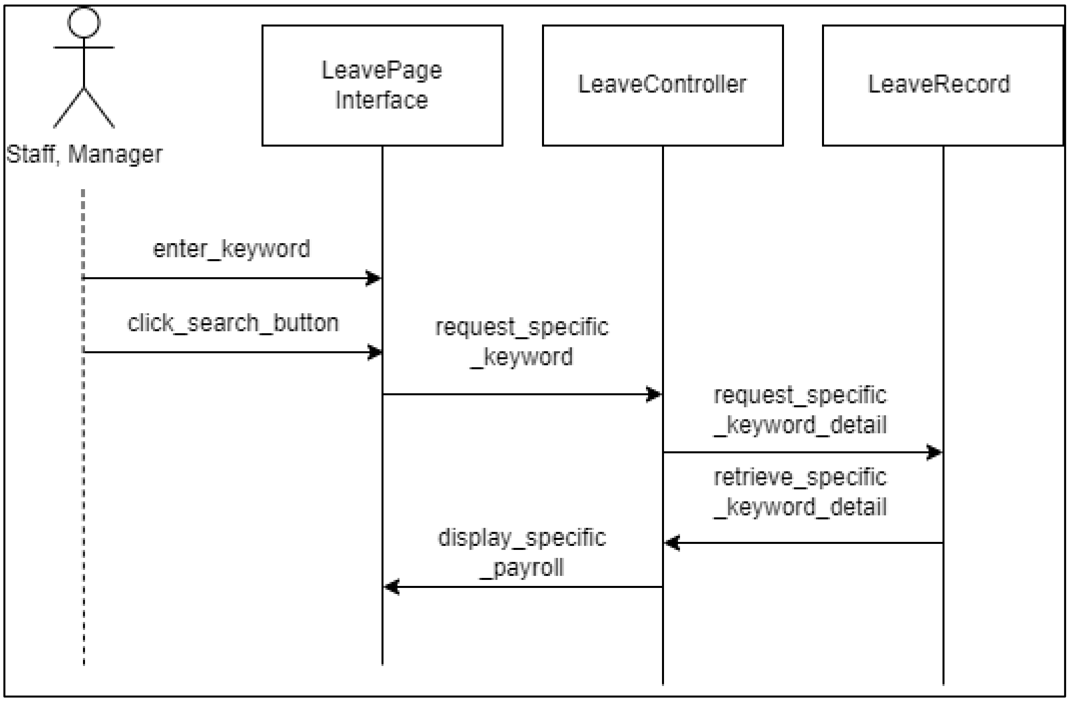

Administrator
LeavePage Interface
LeaveController
LeaveRecord
enter_keyword
click_search_button
request_specific _keyword
request_specific _keyword_detail
retrieve_specific _keyword_detail
display_specific _payroll

Appendix A-7.2: Alternative [A1]: Search Leave

Appendix A-8: Manage Leave Sequence Diagram (Staff/Manager)

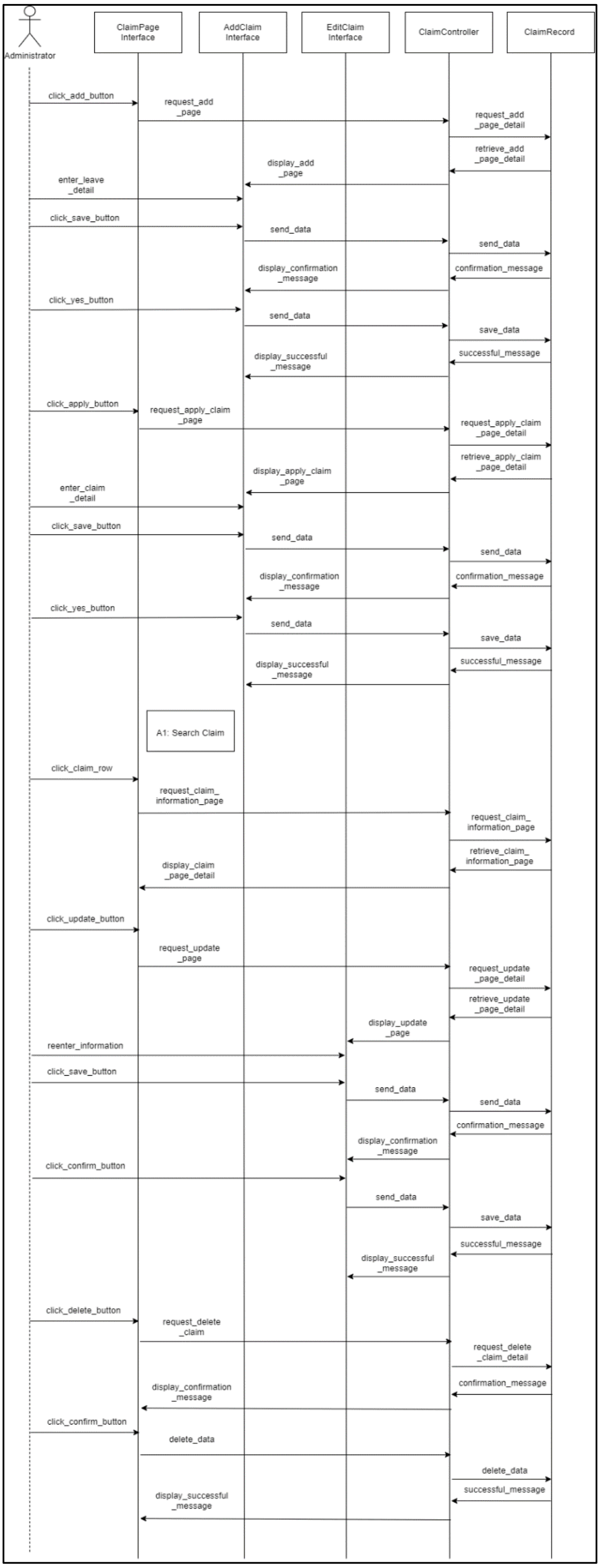


Appendix A-8.1: Basic Flow

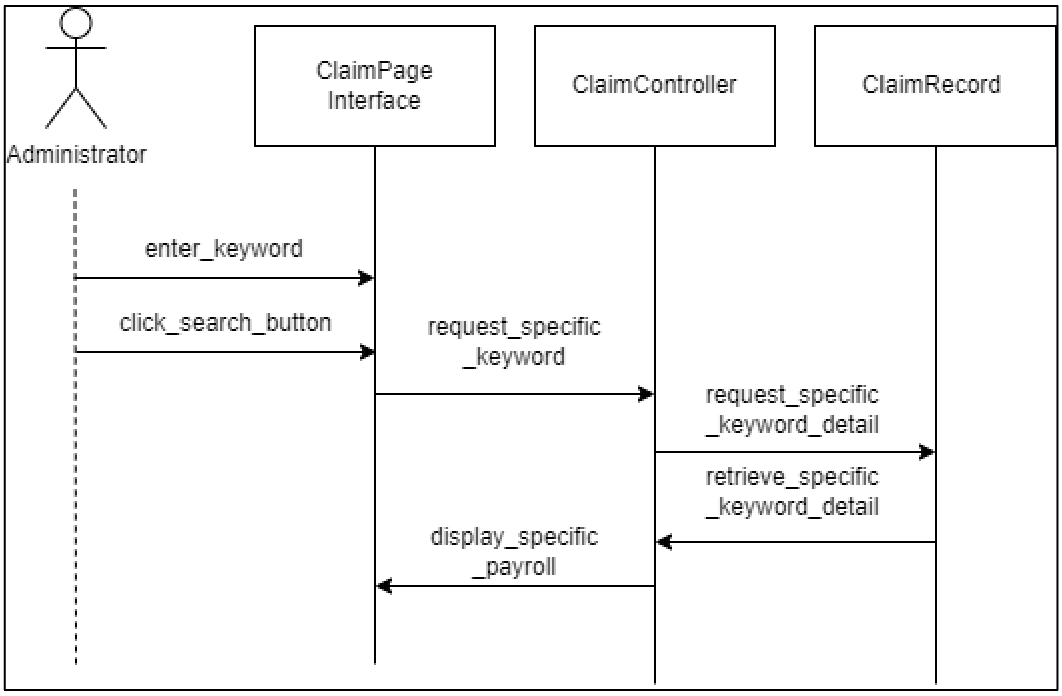

Staff, Manager
LeavePage Interface
LeaveController
LeaveRecord
enter_keyword
click_search_button
request_specific _keyword
request_specific _keyword_detail
retrieve_specific _keyword_detail
display_specific _payroll

Appendix A-8.2: Alternative [A2]: Search Leave Application

Appendix A-9: Manage Claim Sequence Diagram (Administrator)

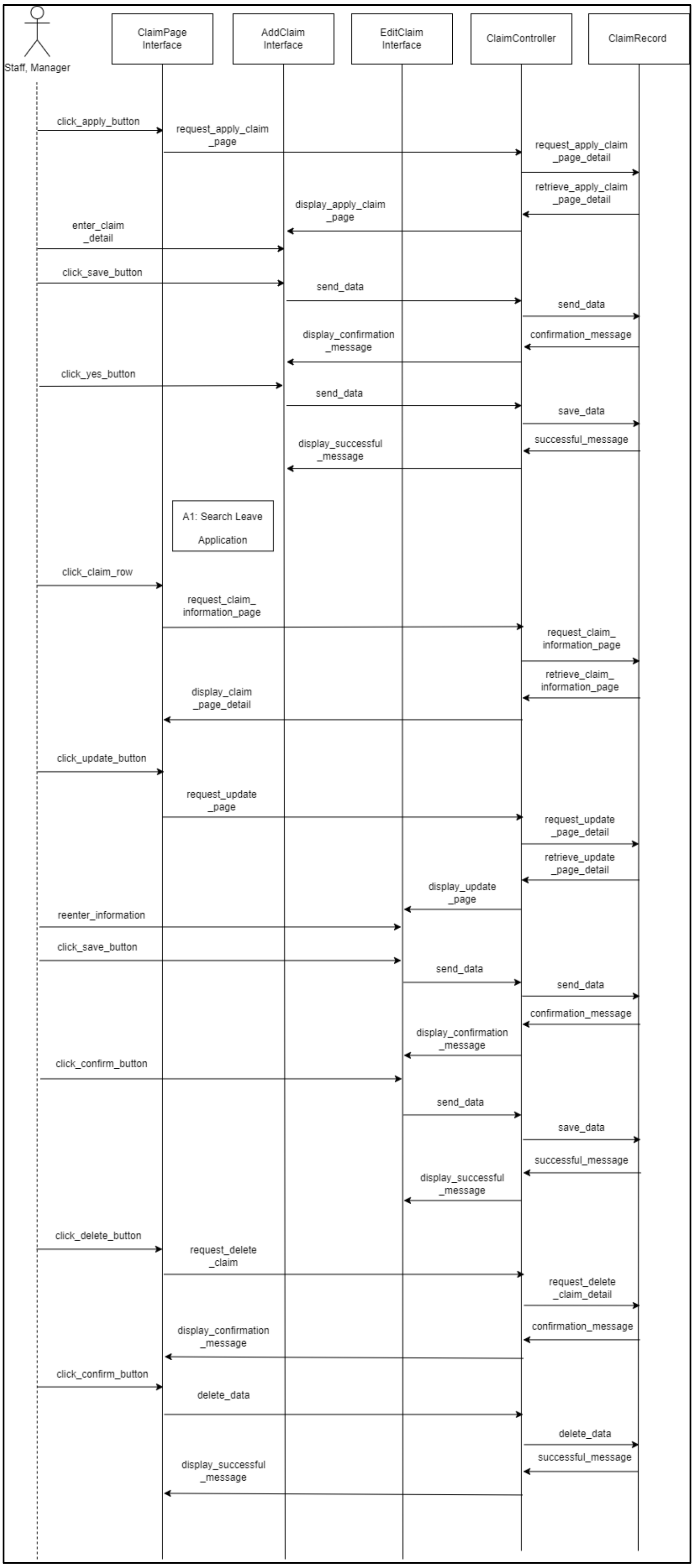


Appendix A-9.1: Basic Flow

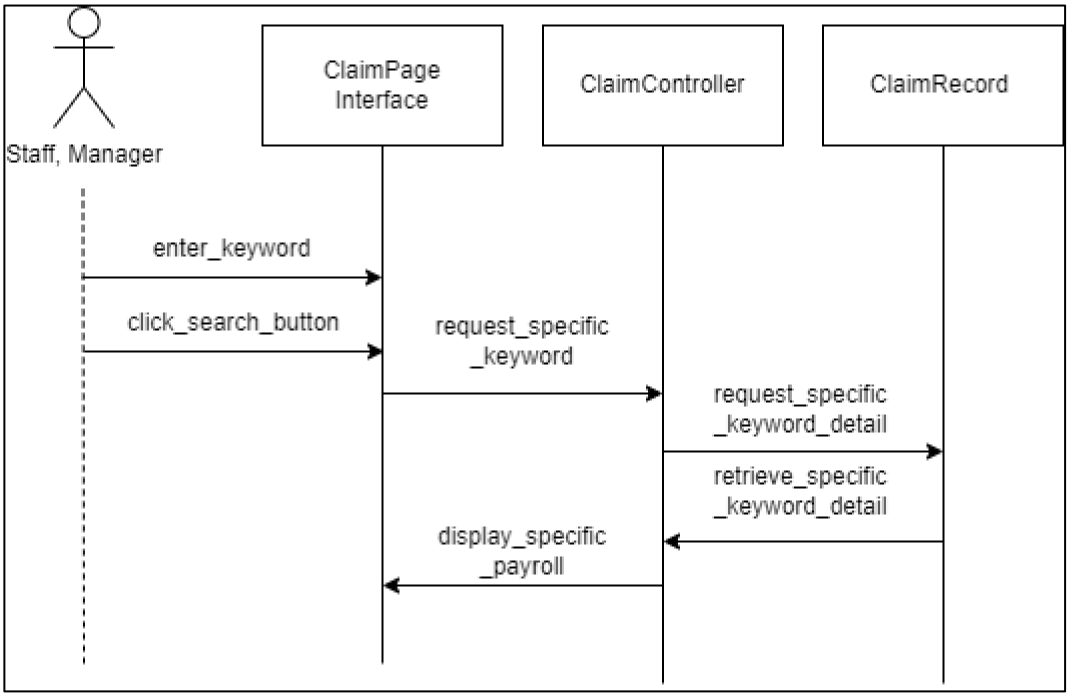

Administrator
ClaimPage Interface
ClaimController
ClaimRecord
enter_keyword
click_search_button
request_specific _keyword
request_specific _keyword_detail
retrieve_specific _keyword_detail
display_specific _payroll

Appendix A-9.2: Alternative [A1]: Search Claim

Appendix A-10: Manage Claim Sequence Diagram (Staff/Manager)

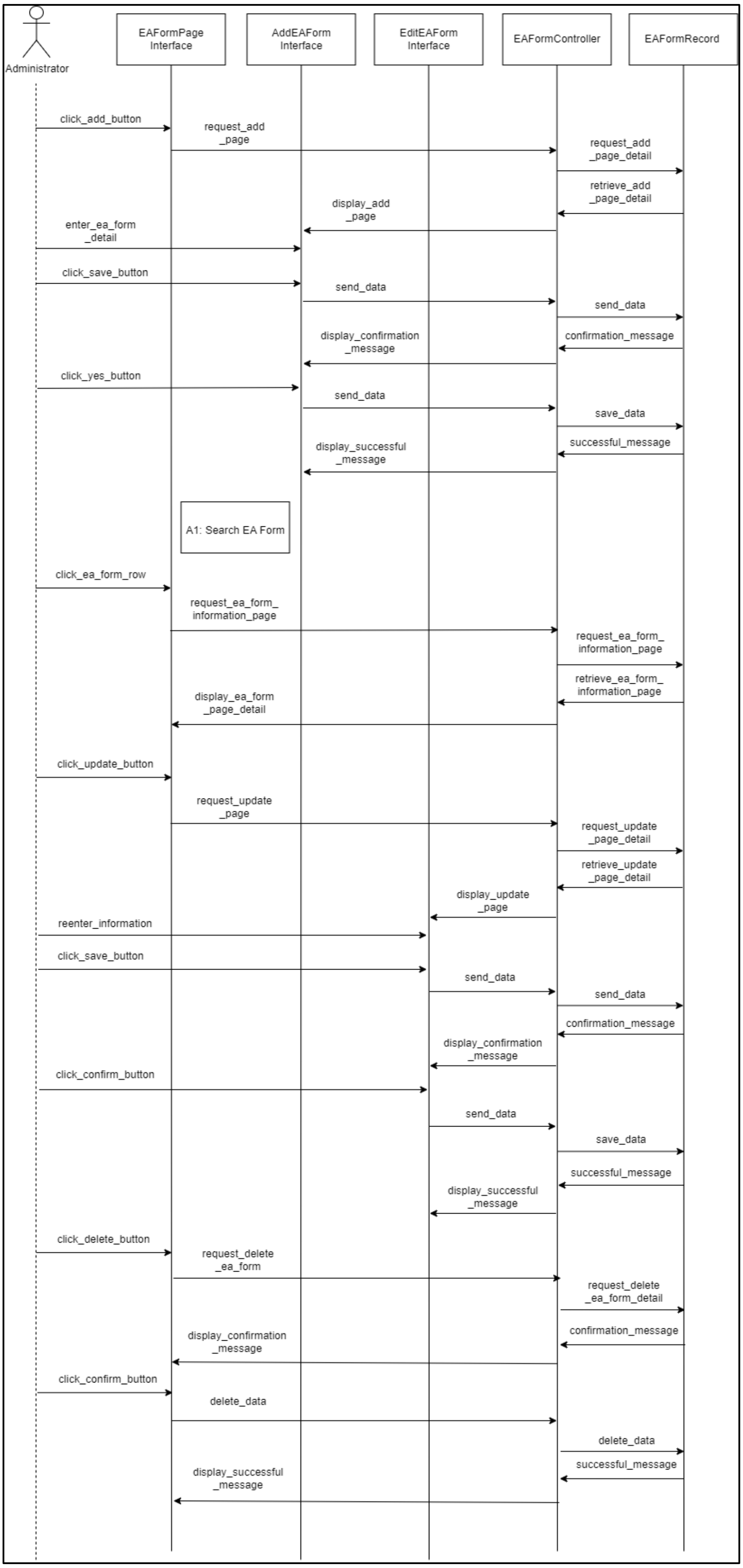


Appendix A-10.1: Basic Flow

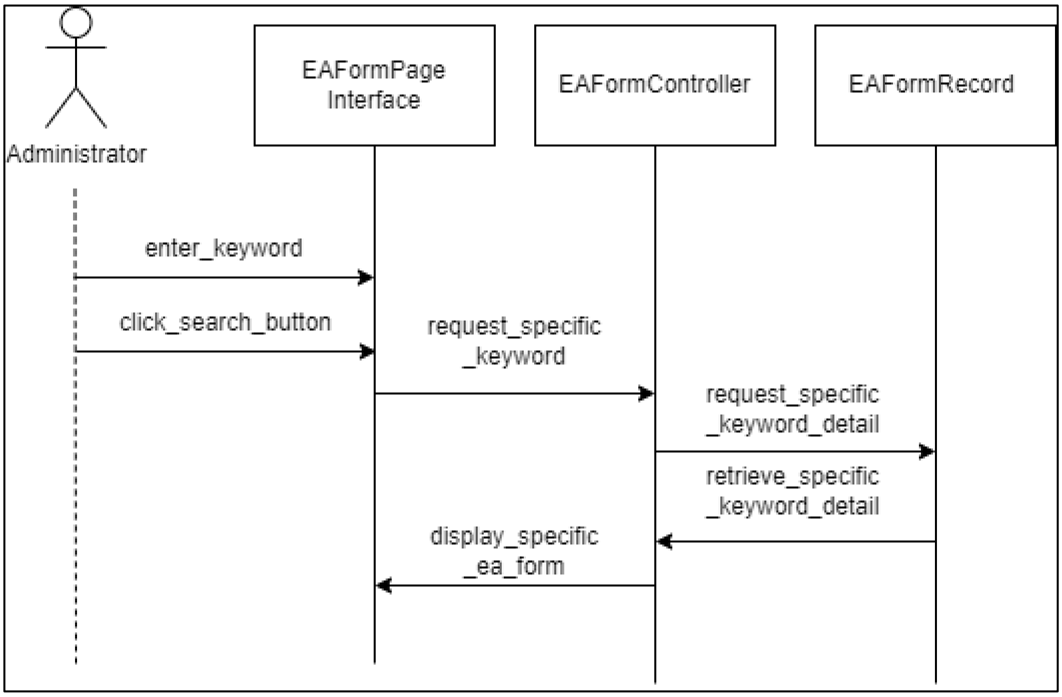

Staff, Manager
ClaimPage Interface
ClaimController
ClaimRecord
enter_keyword
click_search_button
request_specific _keyword
request_specific _keyword_detail
retrieve_specific _keyword_detail
display_specific _payroll

Appendix A-10.2: Alternative [A2]: Search Claim Application

Appendix A-11: Manage EA Form Sequence Diagram (Administrator)

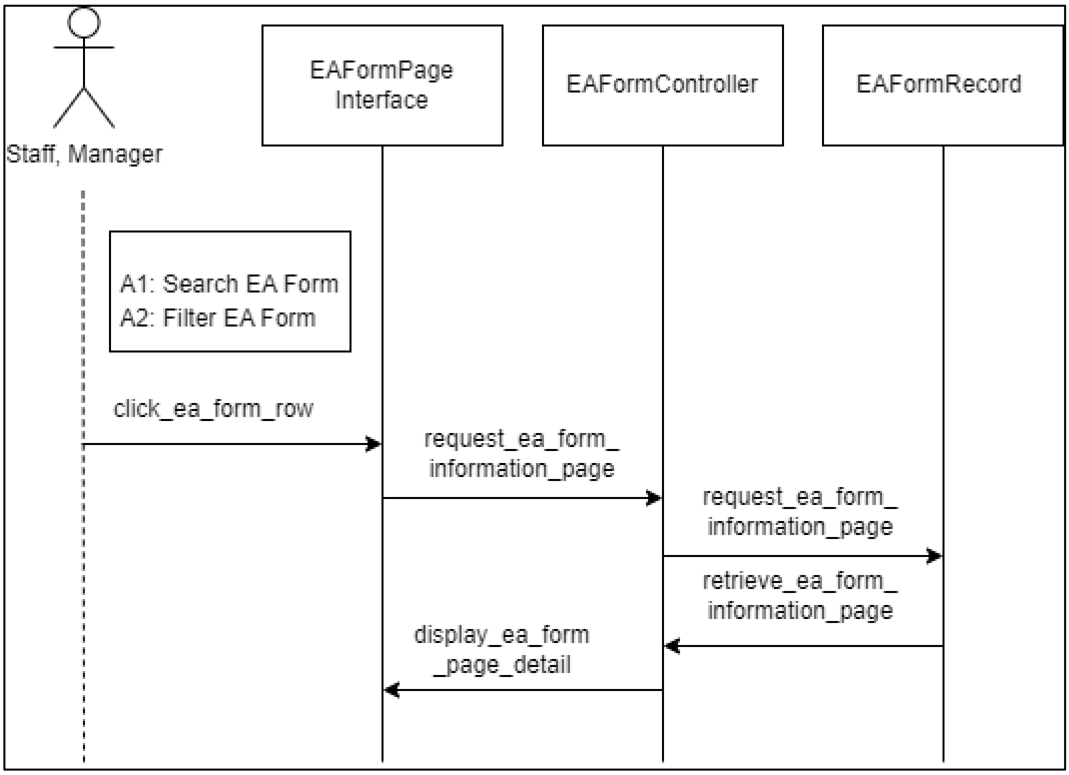


Appendix A-11.1: Basic Flow

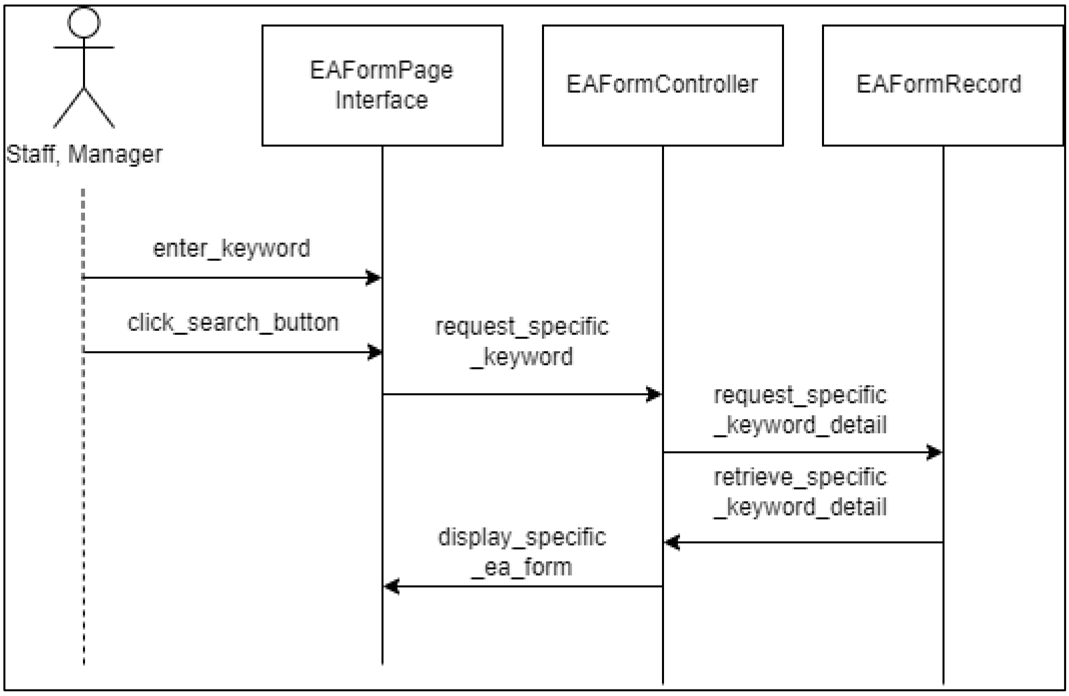

Administrator
EAFormPage Interface
EAFormController
EAFormRecord
enter_keyword
click_search_button
request_specific _keyword
request_specific _keyword_detail
retrieve_specific _keyword_detail
display_specific _ea_form

Appendix A-11.2: Alternative [A1]: Search EA Form

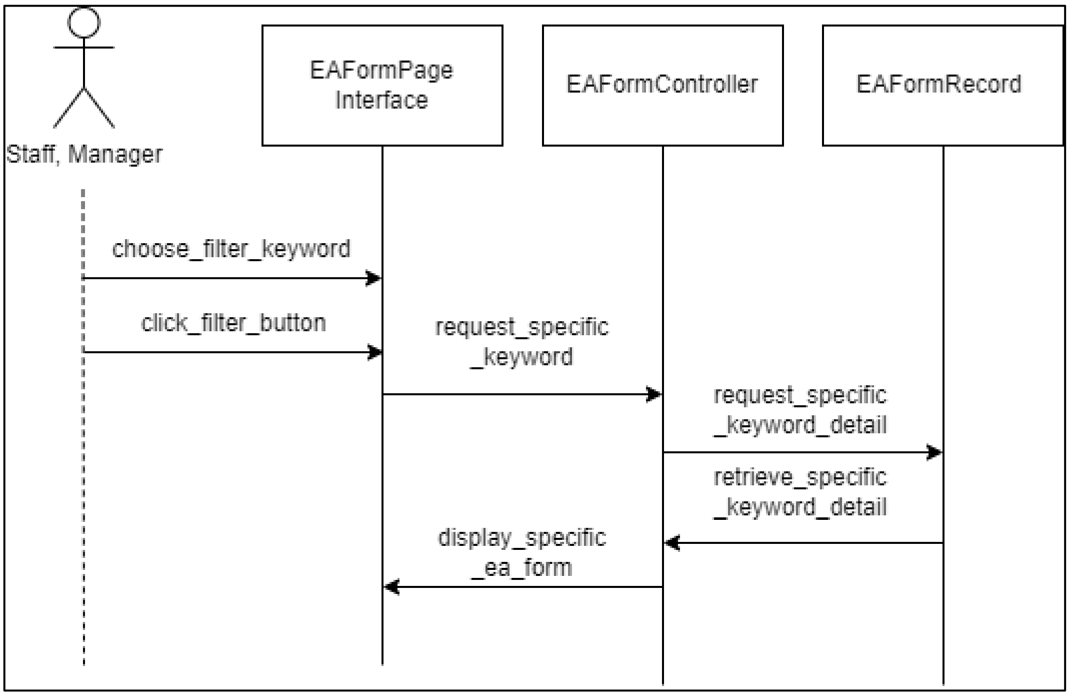
Appendix A-12: View EA Form Sequence Diagram (Staff/Manager)


Staff, Manager
EAFormPage Interface
EAFormController
EAFormRecord
A1: Search EA Form
A2: Filter EA Form
click_ea_form_row
request_ea_form_ information_page
request_ea_form_ information_page
retrieve_ea_form_ information_page
display_ea_form _page_detail

Appendix A-12.1: Basic Flow


Staff, Manager
EAFormPage Interface
EAFormController
EAFormRecord
enter_keyword
click_search_button
request_specific _keyword
request_specific _keyword_detail
retrieve_specific _keyword_detail
display_specific _ea_form

Appendix A-12.2: Alternative [A1]: Search EA Form

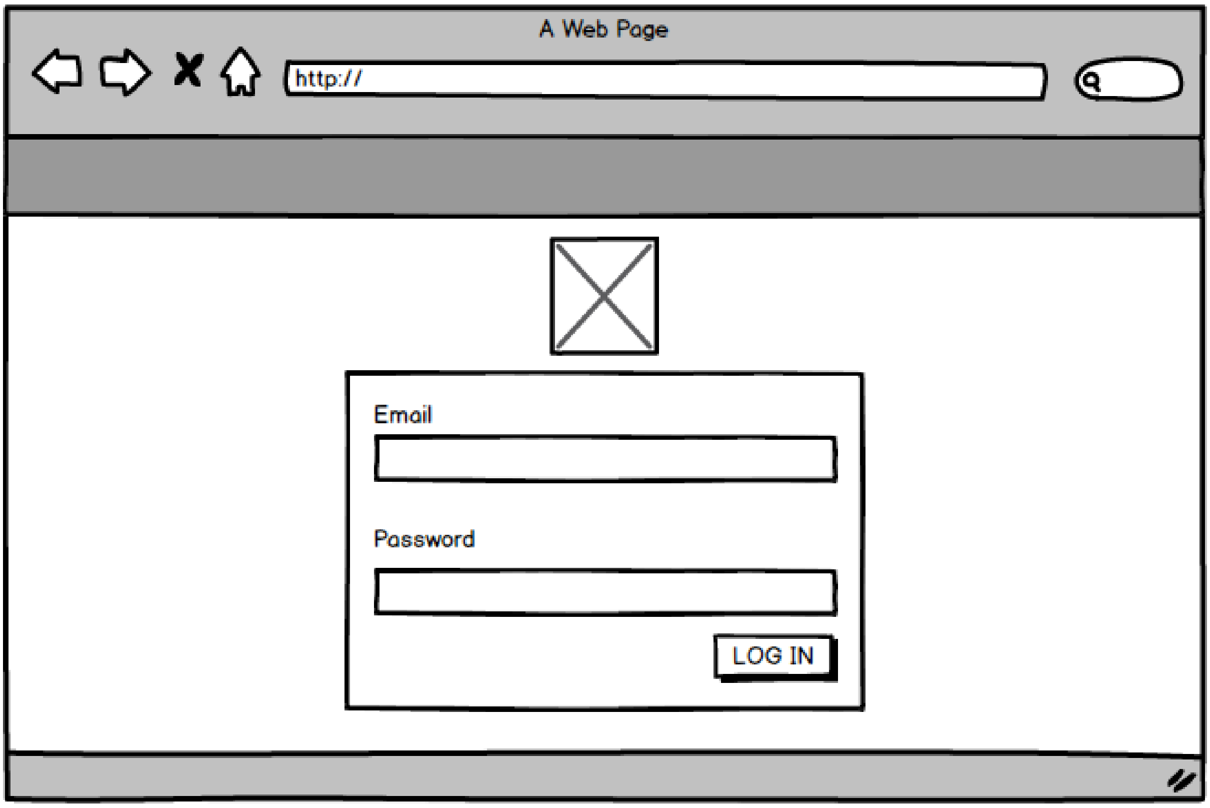

Staff, Manager
EAFormPage Interface
EAFormController
EAFormRecord
choose_filter_keyword
click_filter_button
request_specific _keyword
request_specific _keyword_detail
retrieve_specific _keyword_detail
display_specific _ea_form

Appendix A-12.3: Alternative [A2]: Filter EA Form

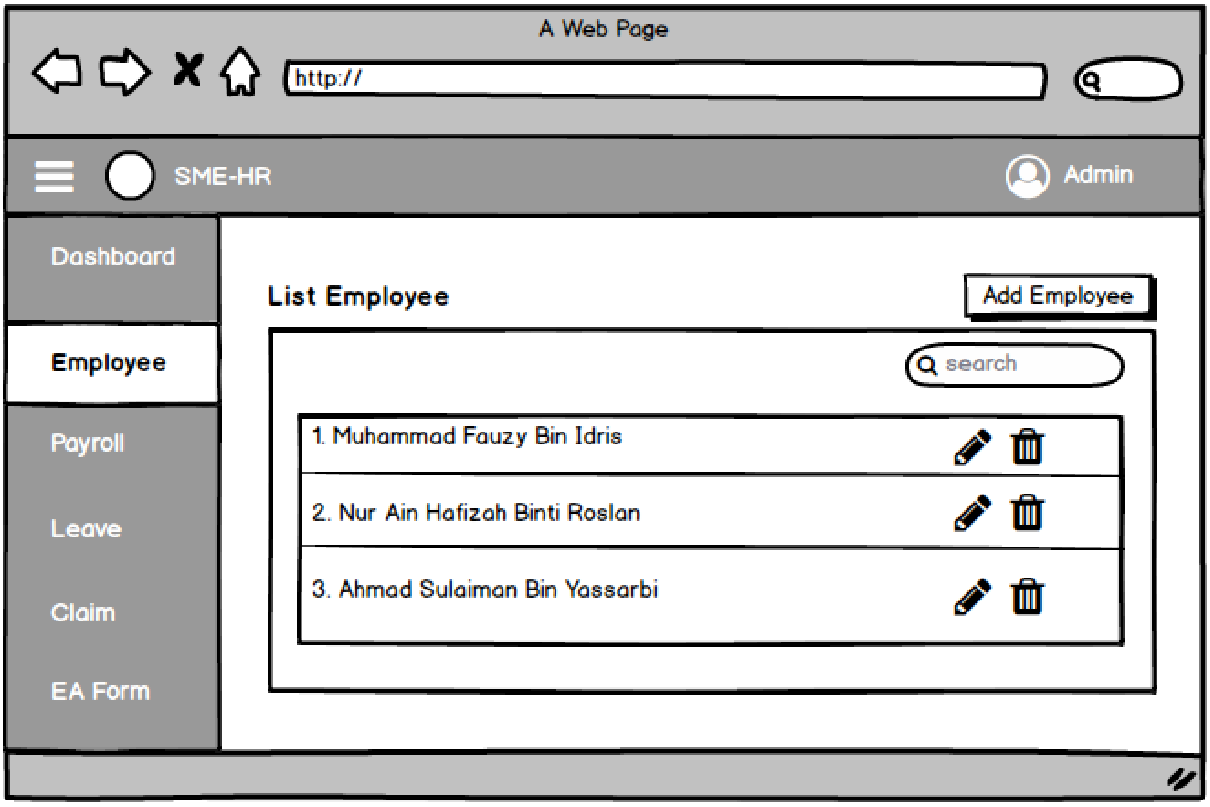
APPENDIX B

User Interfaces

Appendix B-1: Mockup Interface

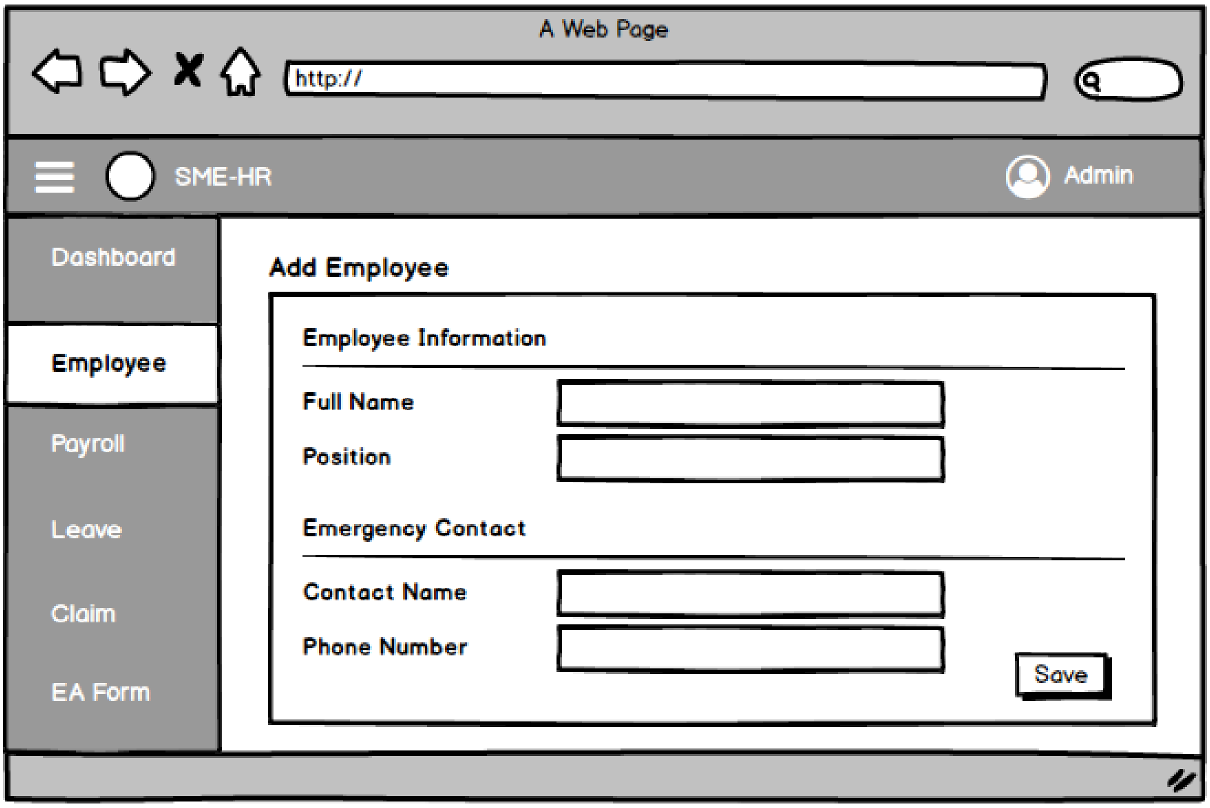


Appendix B-1.1: Login Interface



1. Muhammad Fauzy Bin Idris
2. Nur Ain Hafizah Binti Roslan
3. Ahmad Sulaiman Bin Yassarbi

Appendix B-1.2: Manage Employee



Employee Information

Full Name

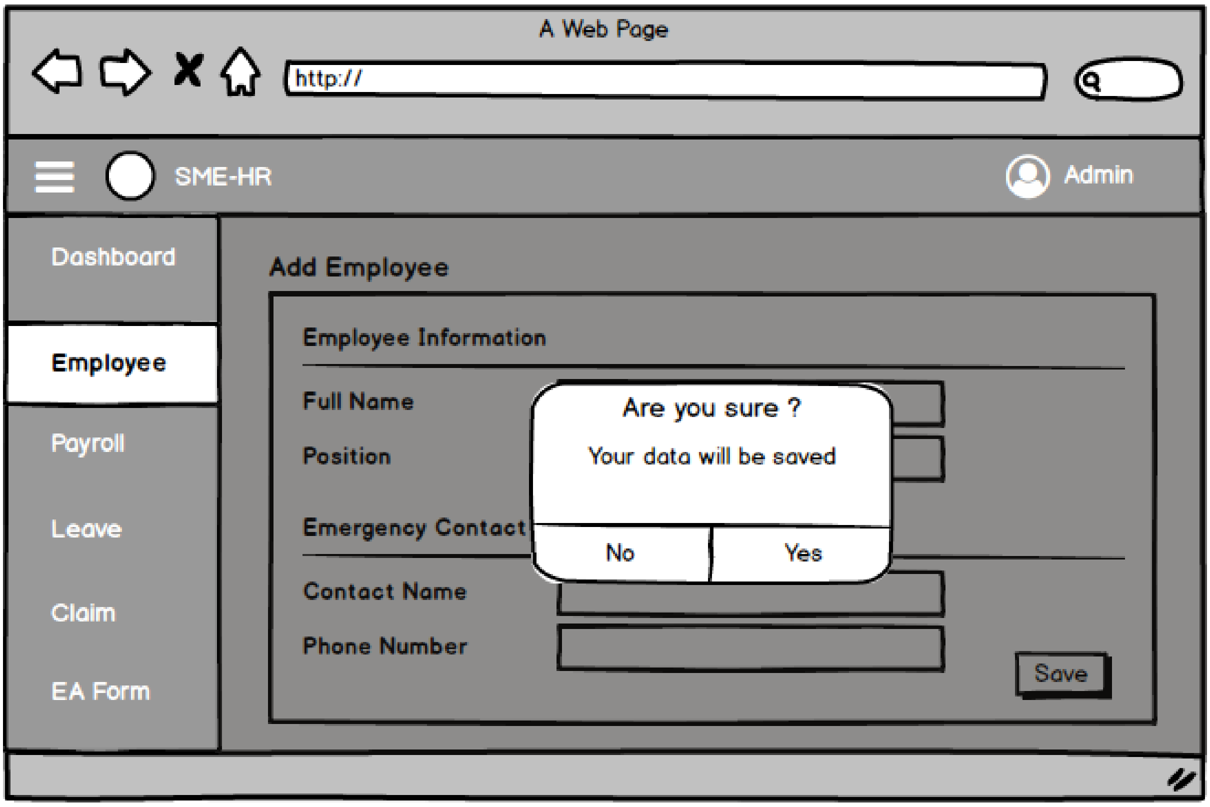
Position

Emergency Contact

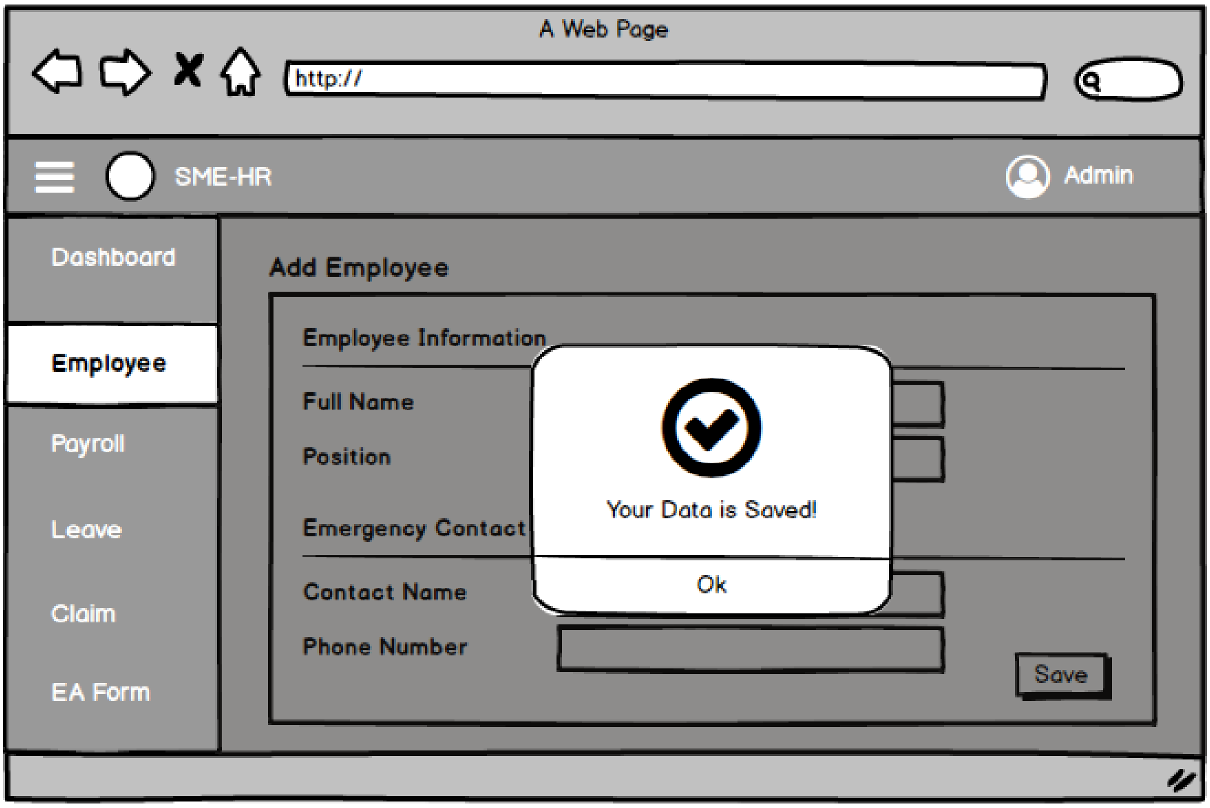
Contact Name

Phone Number

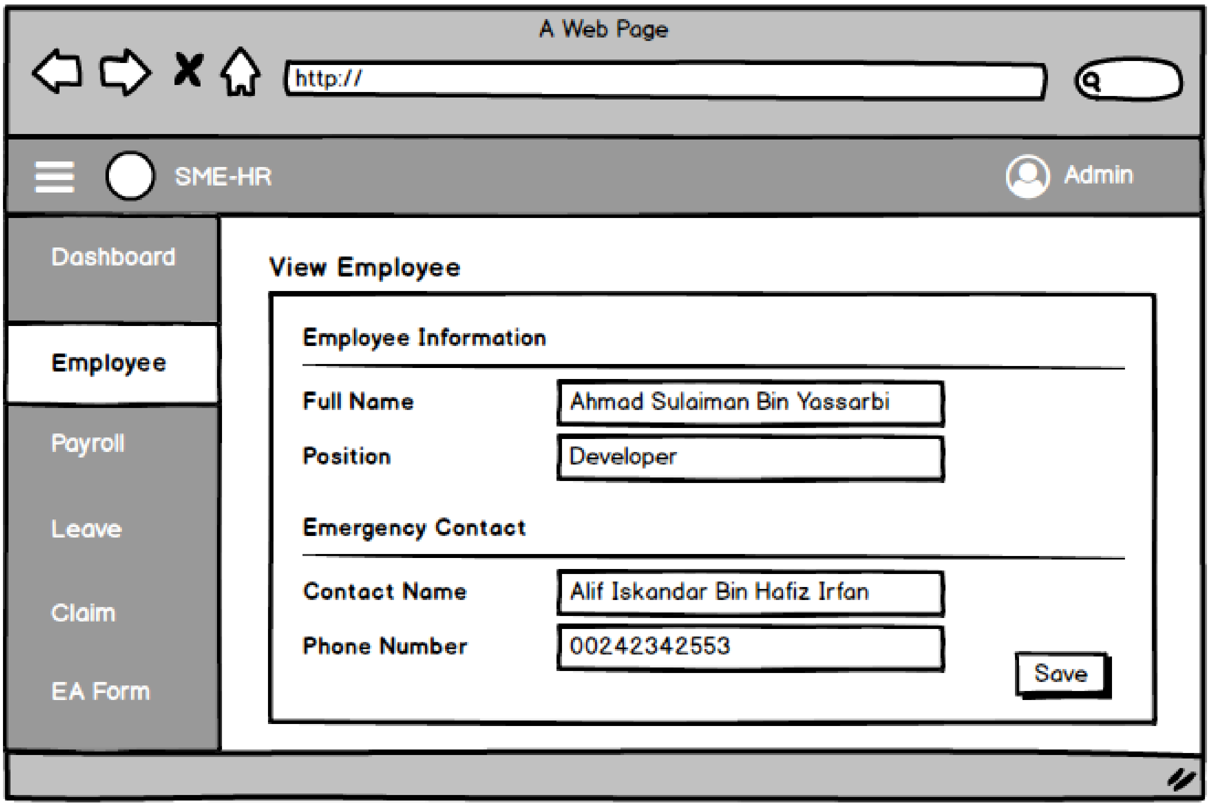
Appendix B-1.3: Add Employee



Appendix B-1. 4: Add Employee Confirmation



Appendix B-1. 5: Add Employee Successful



Employee Information

Full Name: Ahmad Sulaiman Bin Yassarbi

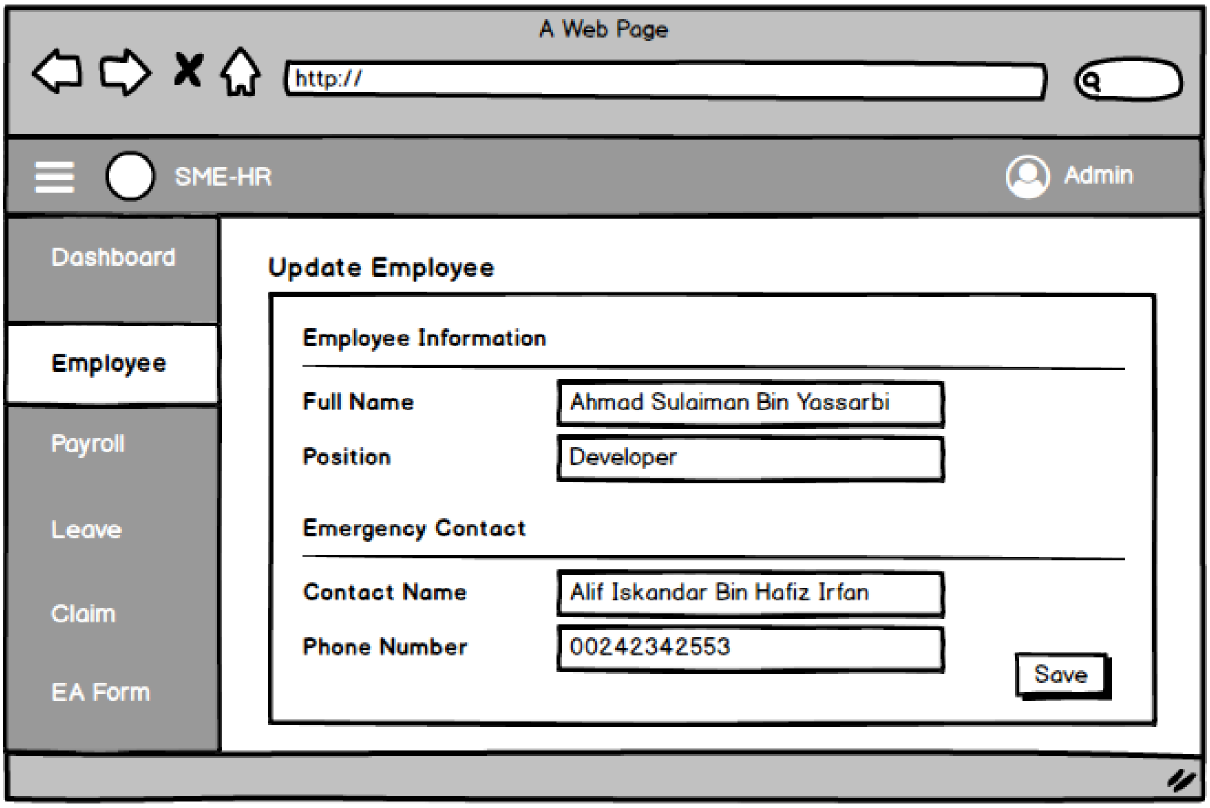
Position: Developer

Emergency Contact

Contact Name: Alif Iskandar Bin Hafiz Irfan

Phone Number: 00242342553

Appendix B-1.6: View Employee



Employee Information

Full Name: Ahmad Sulaiman Bin Yassarbi

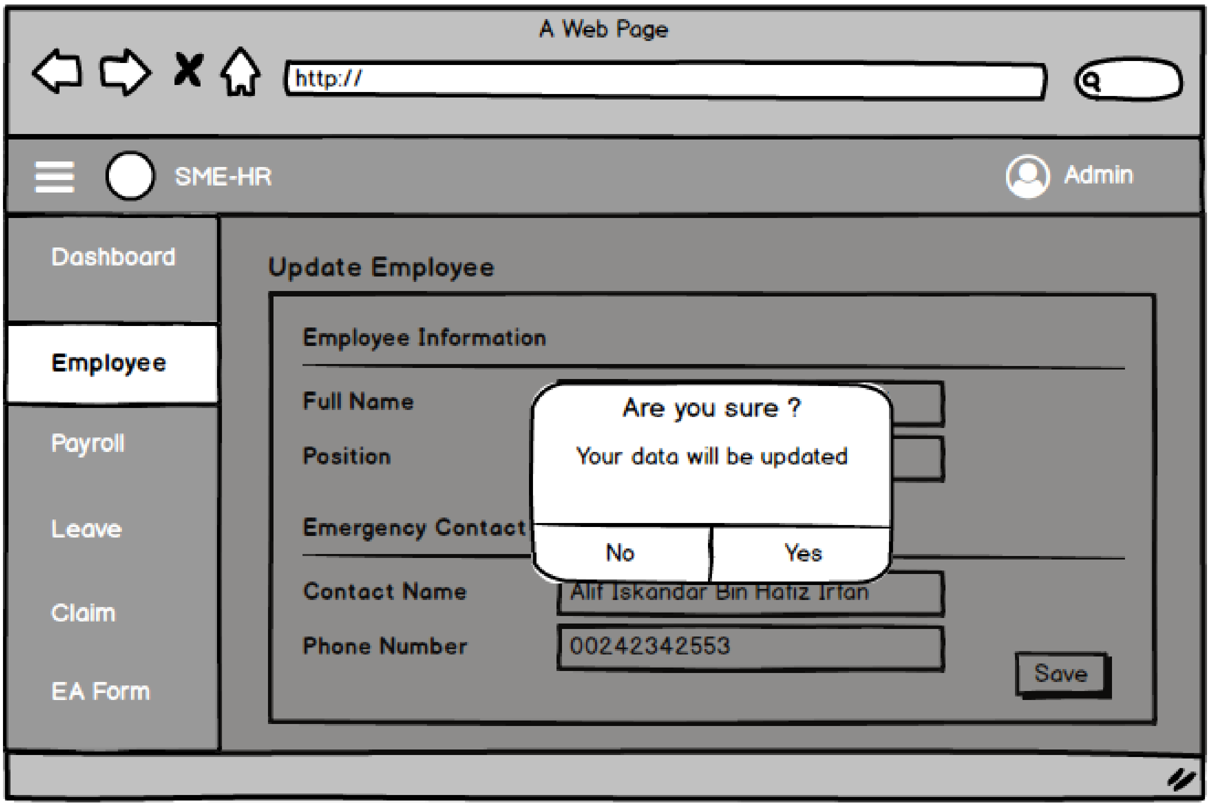
Position: Developer

Emergency Contact

Contact Name: Alif Iskandar Bin Hafiz Irfan

Phone Number: 00242342553

Appendix B-1.7: Update Employee


A Web Page

http://

SME-HR

Admin

Update Employee

Employee Information

Full Name

Position

Emergency Contact

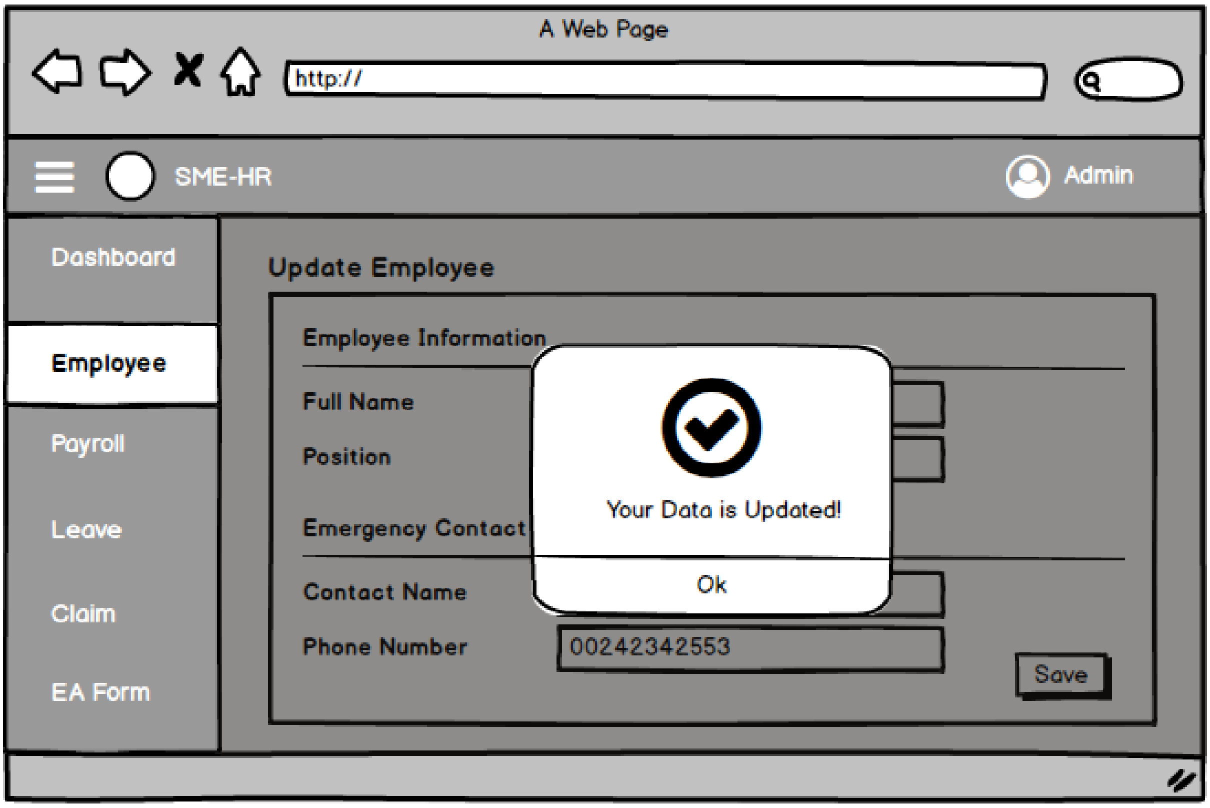
Contact Name: Alif Iskandar Bin Hafiz Irfan

Phone Number: 00242342553

Are you sure ?

Your data will be updated

Appendix B-1.8: Update Employee Confirmation


A Web Page

http://

SME-HR

Admin

Update Employee

Employee Information

Full Name

Position

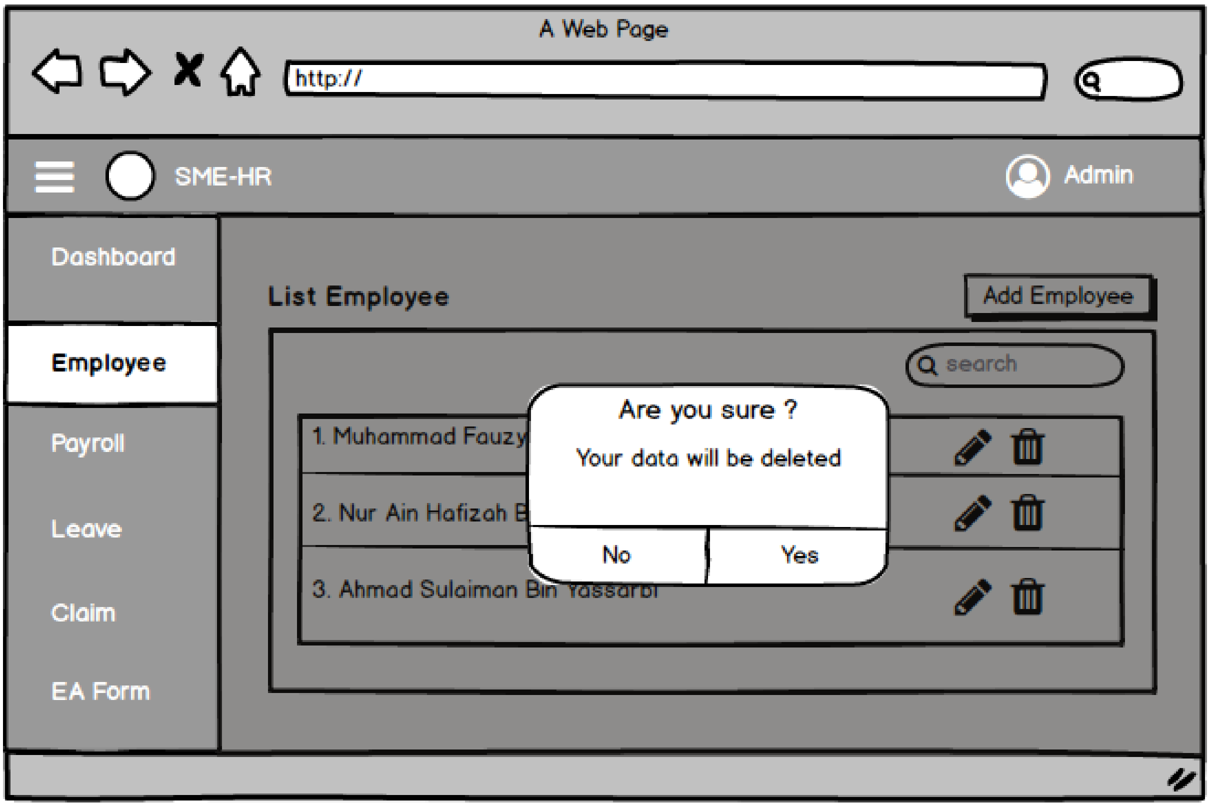
Emergency Contact

Contact Name

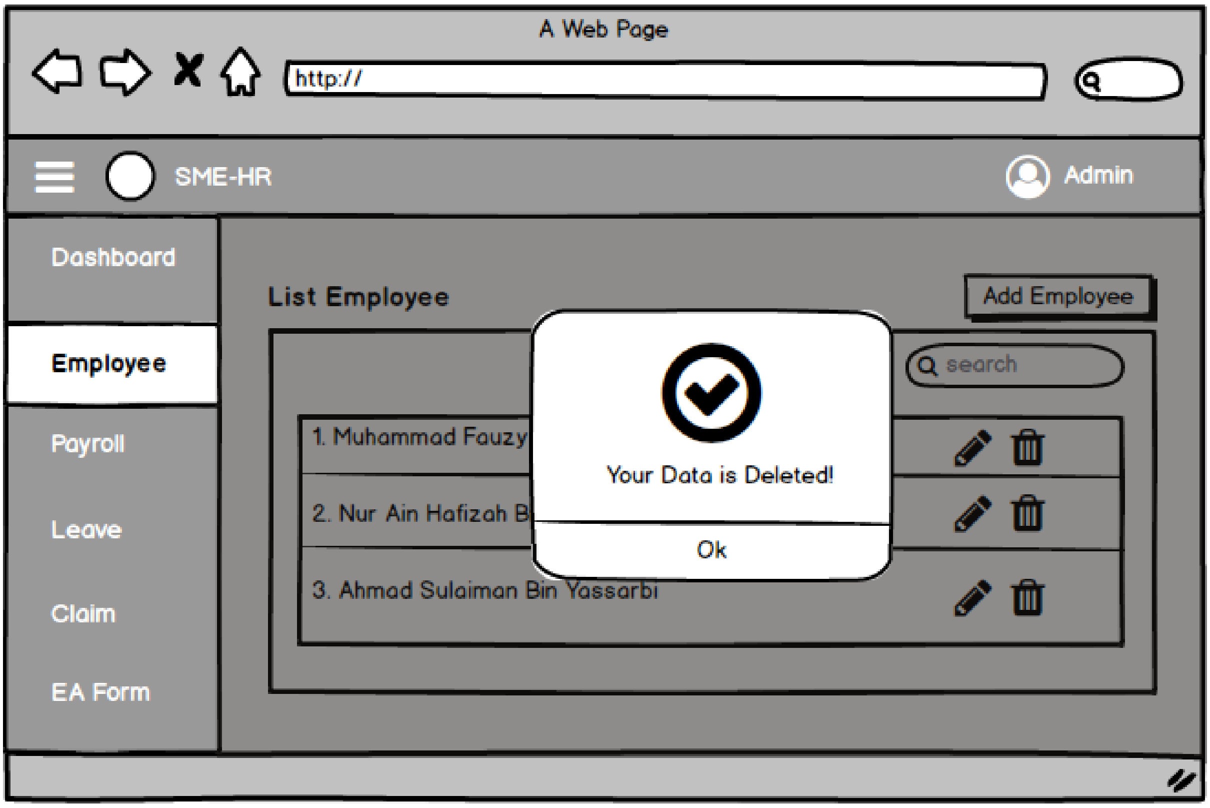
Phone Number: 00242342553

Your Data is Updated!

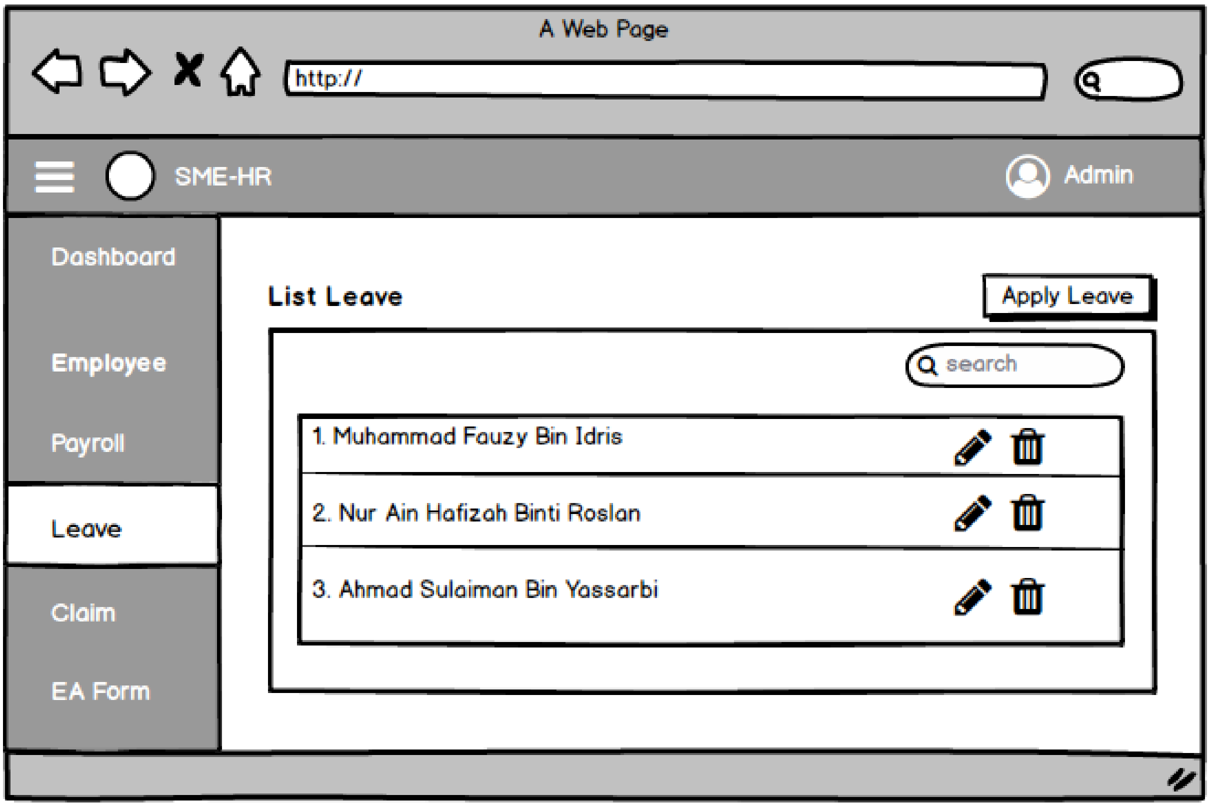
Appendix B-1. 9: Update Employee Success



Appendix B-1.10: Delete Employee Confirmation

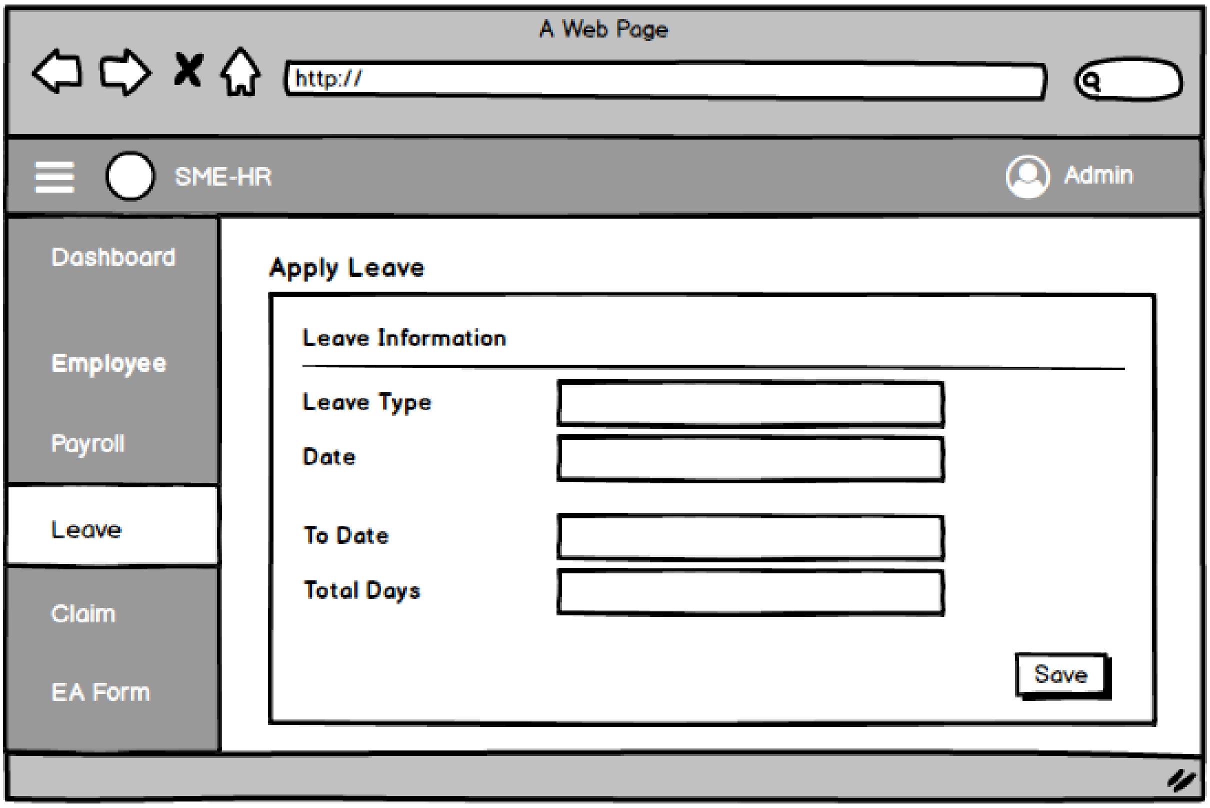


Appendix B-1.11: Delete Employee Success



1. Muhammad Fauzy Bin Idris
2. Nur Ain Hafizah Binti Roslan
3. Ahmad Sulaiman Bin Yassarbi

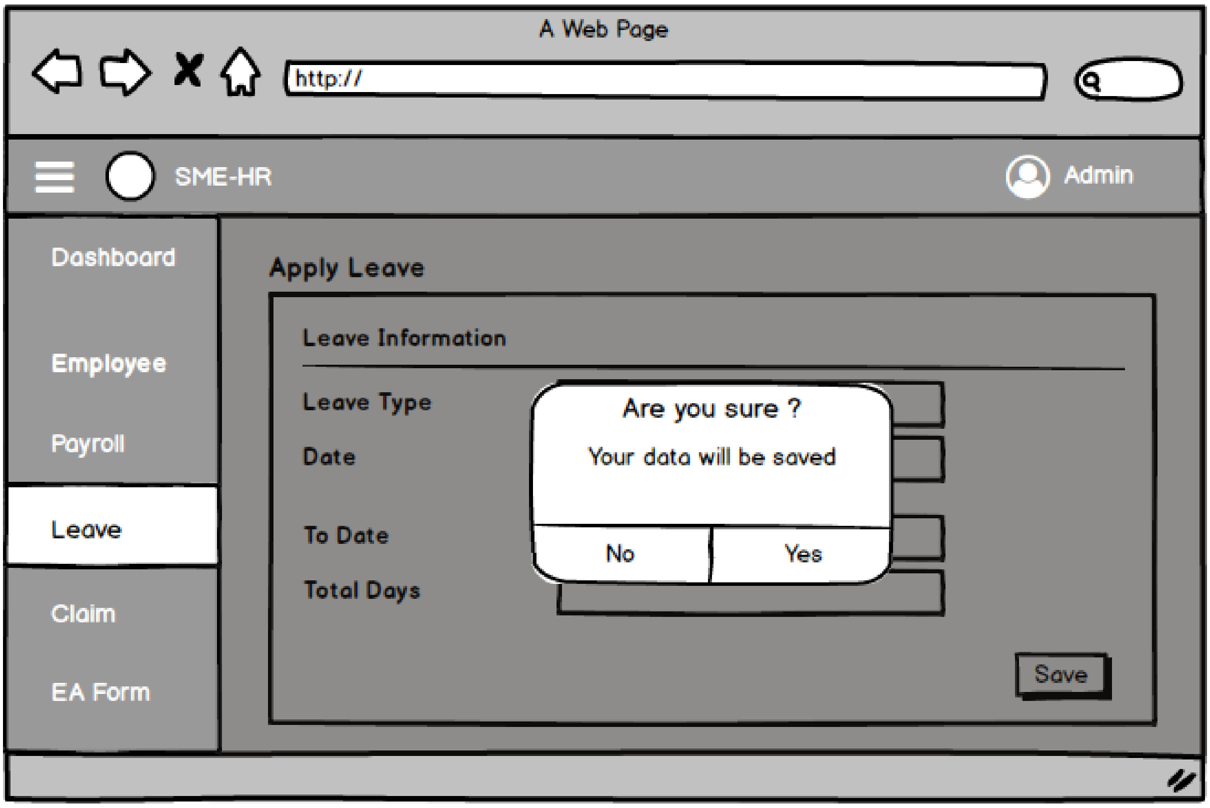
Appendix B-1.12: Leave Page



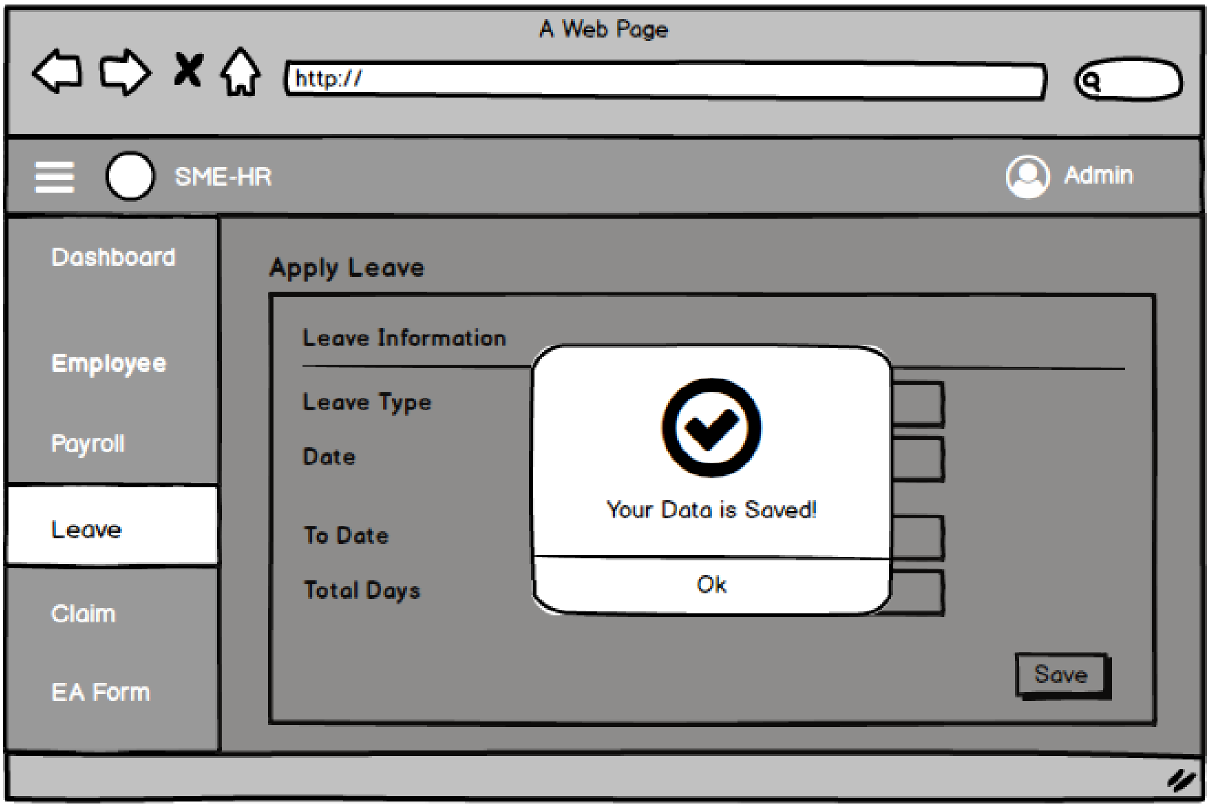
Leave Information

Leave Type	
Date	
To Date	
Total Days	

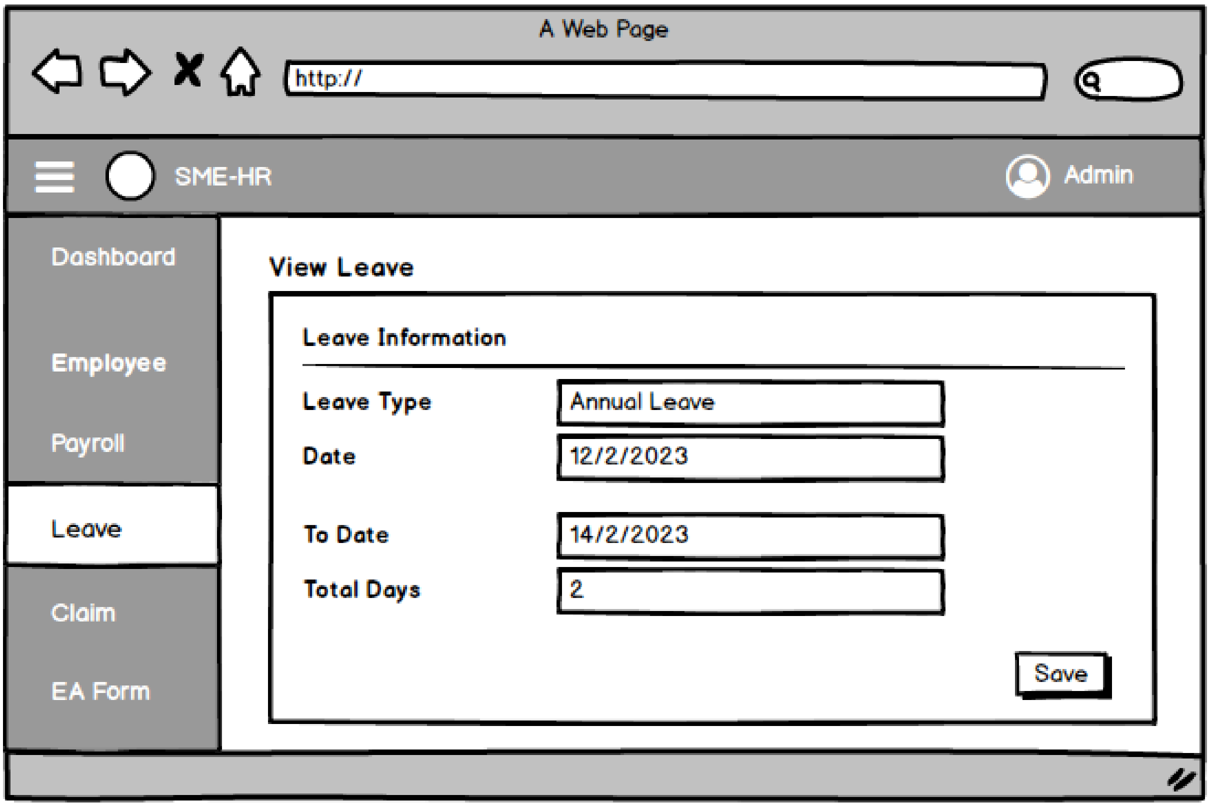
Appendix B-1.13: Add Leave Page



Appendix B-1.14: Add Leave Confirmation



Appendix B-1.15: Add Leave Success

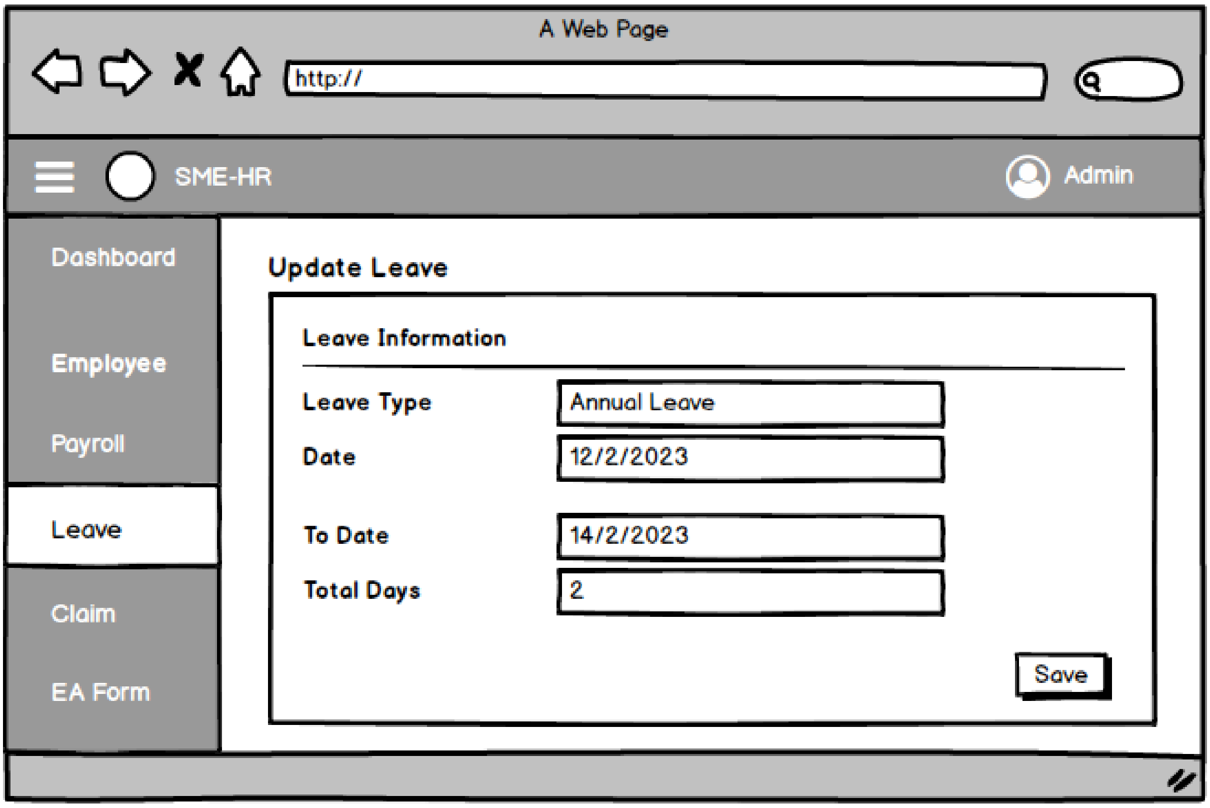


View Leave

Leave Information

Leave Type	Annual Leave
Date	12/2/2023
To Date	14/2/2023
Total Days	2

Appendix B-1.16: View Leave

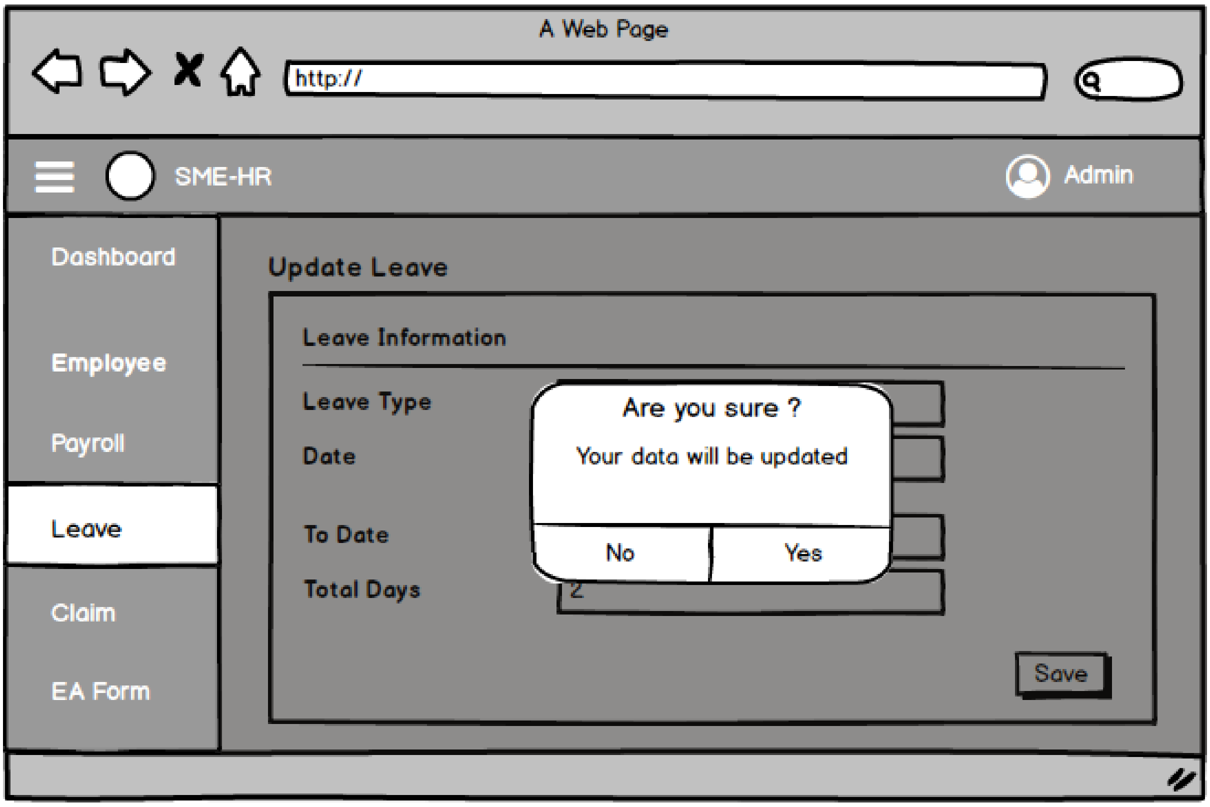


Update Leave

Leave Information

Leave Type	Annual Leave
Date	12/2/2023
To Date	14/2/2023
Total Days	2

Appendix B-1.17: Update Leave

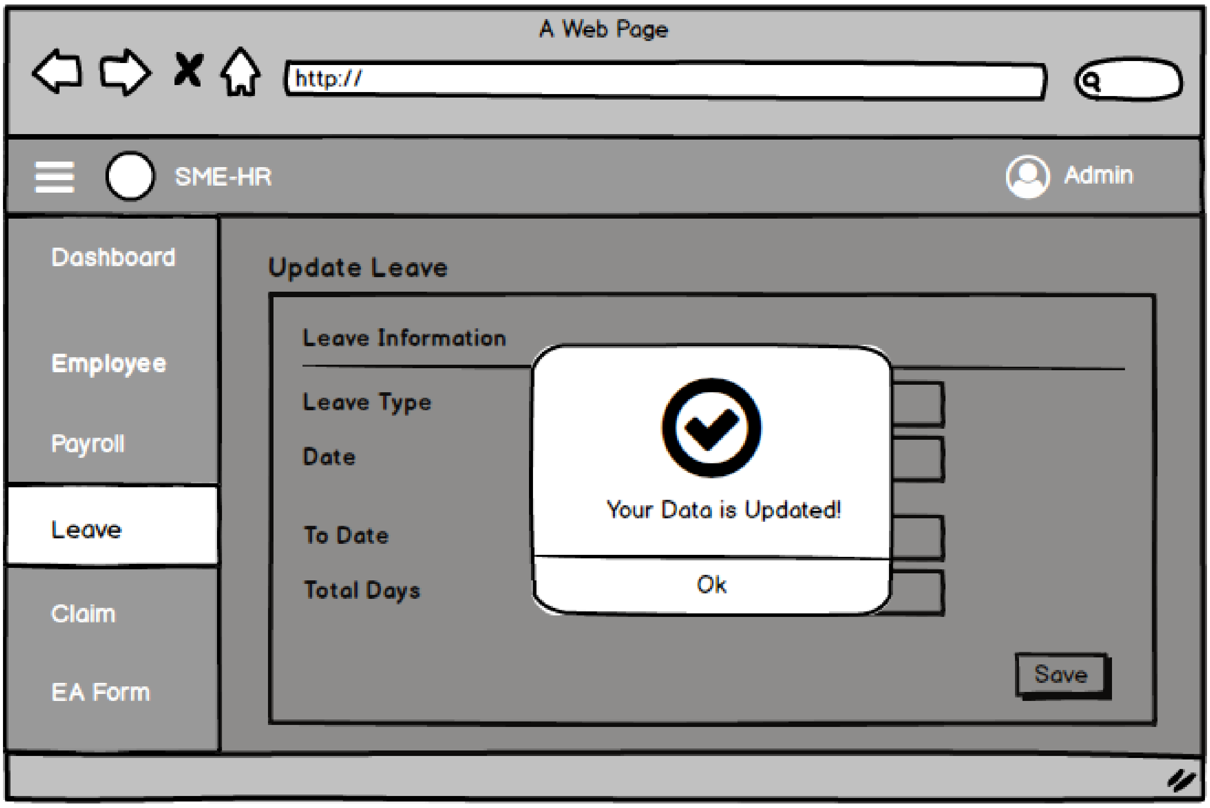


Are you sure ?

Your data will be updated

No Yes

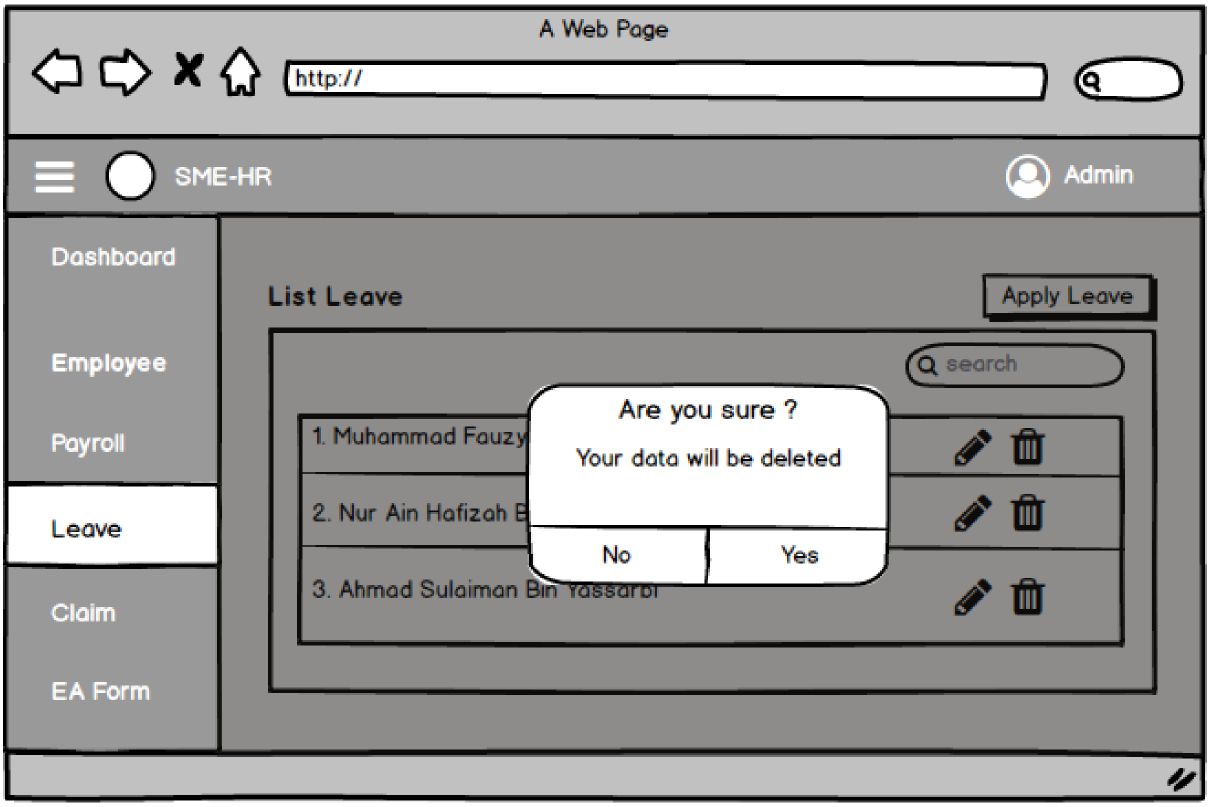
Appendix B-1.18: Update Leave Confirmation



Your Data is Updated!

Ok

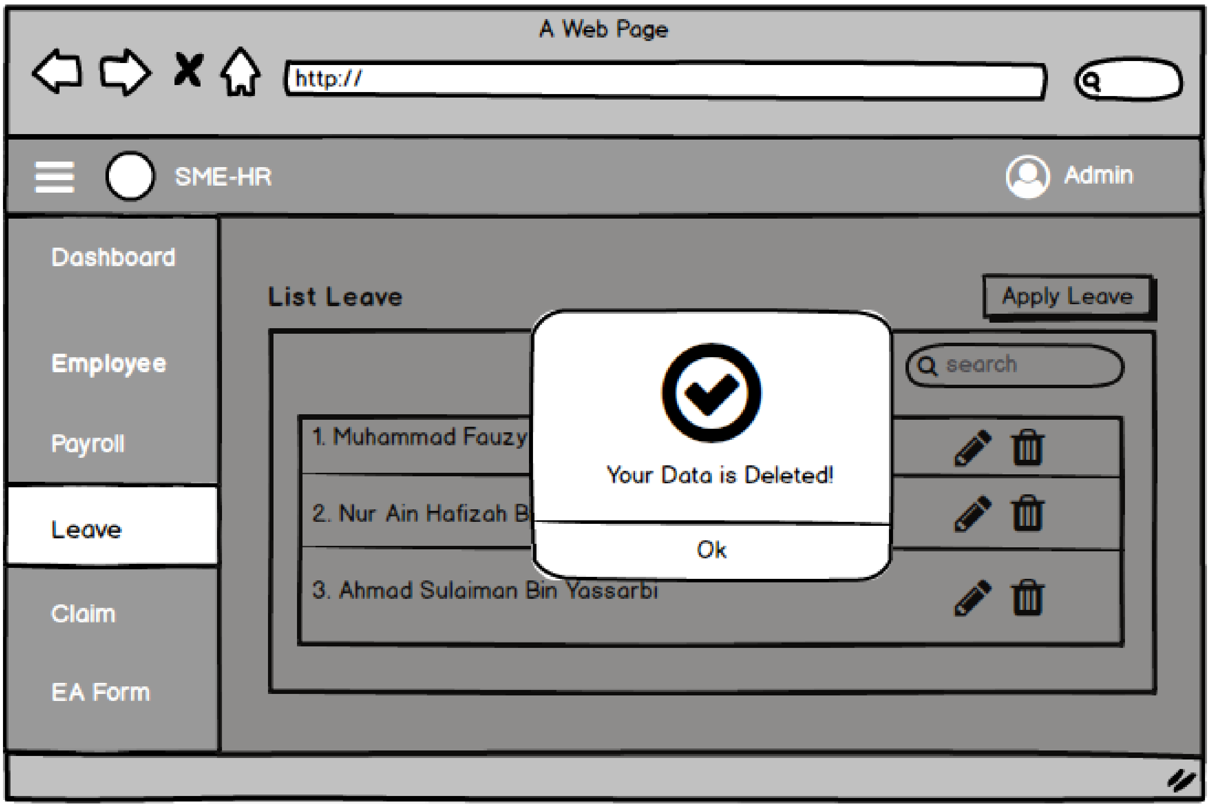
Appendix B-1.19: Update Leave Success



Are you sure ?

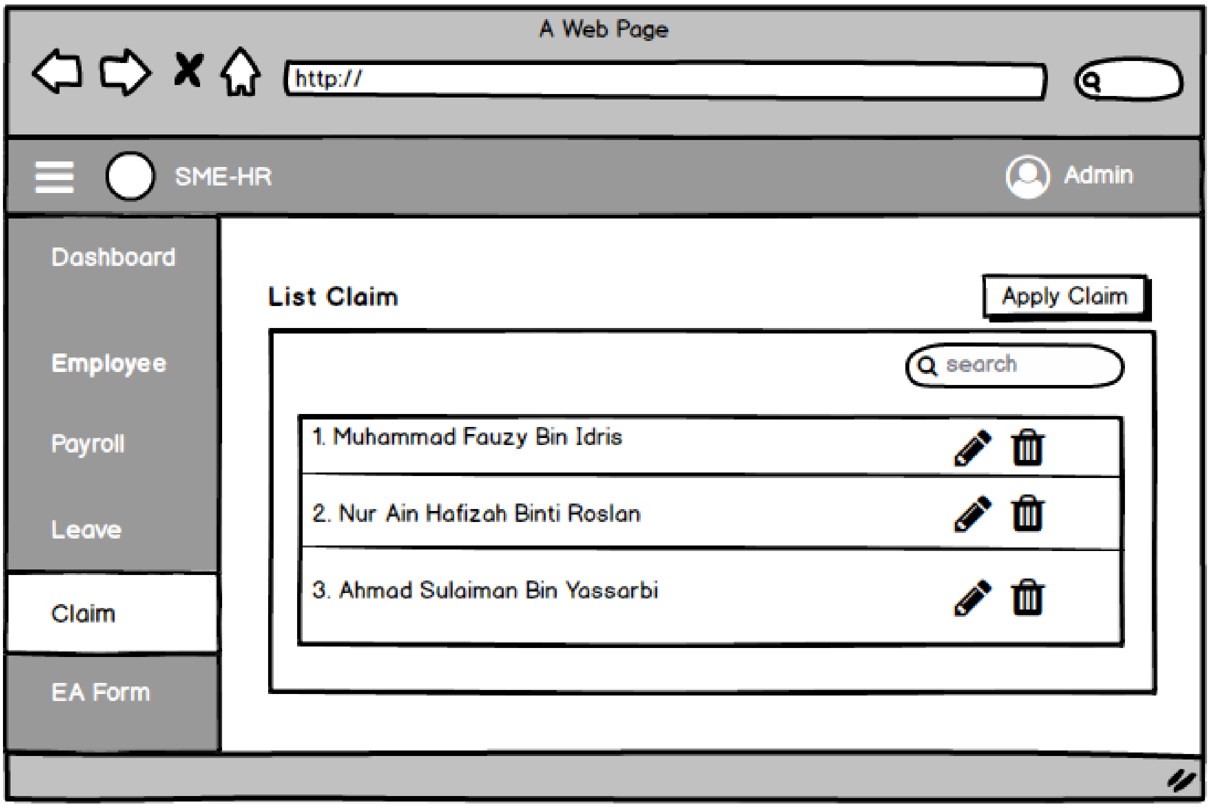
Your data will be deleted

Appendix B-1.20: Delete Leave Confirmation



Your Data is Deleted!

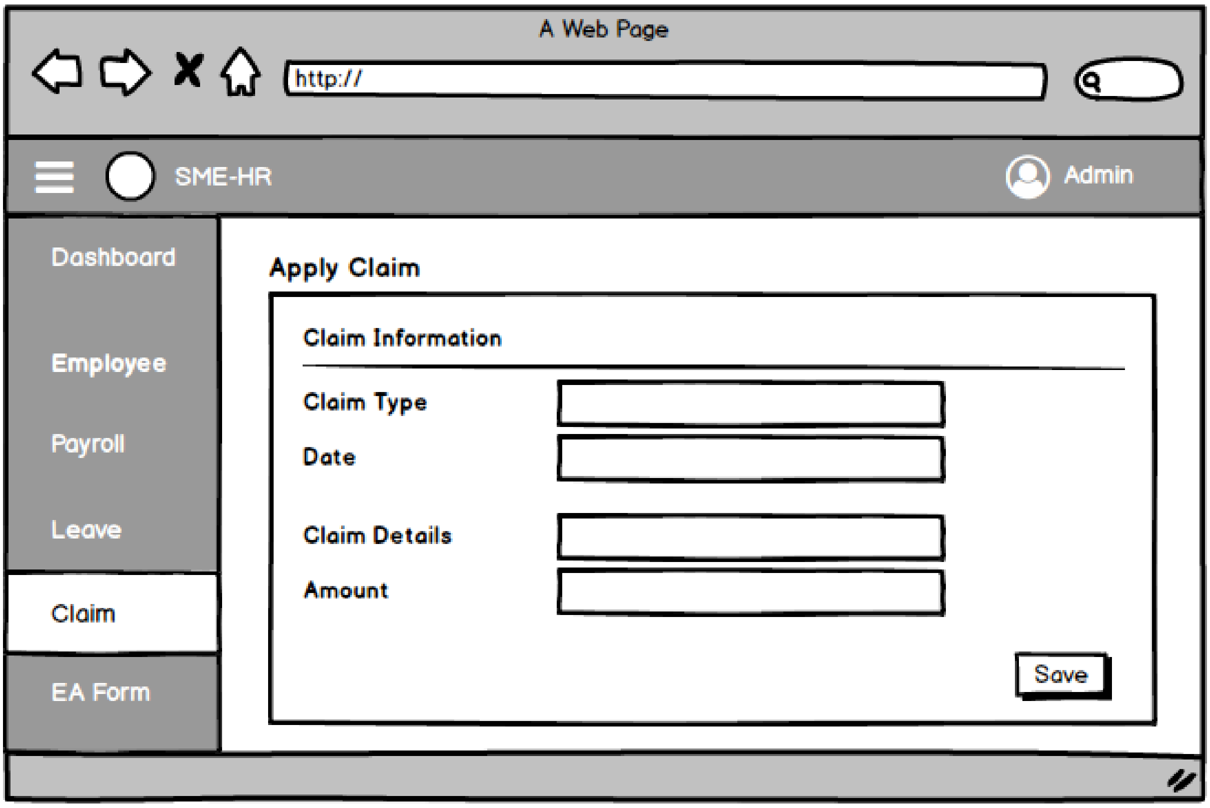
Appendix B-1.21: Delete Leave Success



List Claim

1. Muhammad Fauzy Bin Idris
2. Nur Ain Hafizah Binti Roslan
3. Ahmad Sulaiman Bin Yassarbi

Appendix B-1.22: Claim Page



Apply Claim

Claim Information

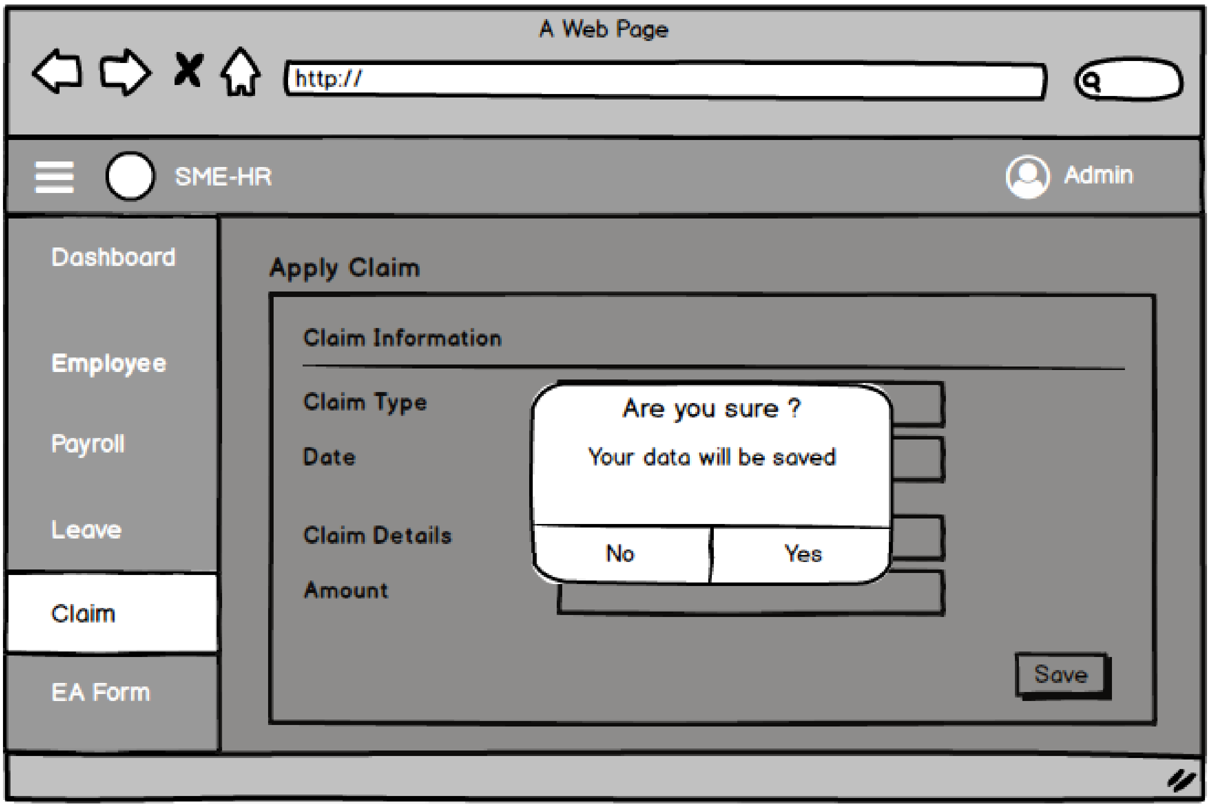
Claim Type

Date

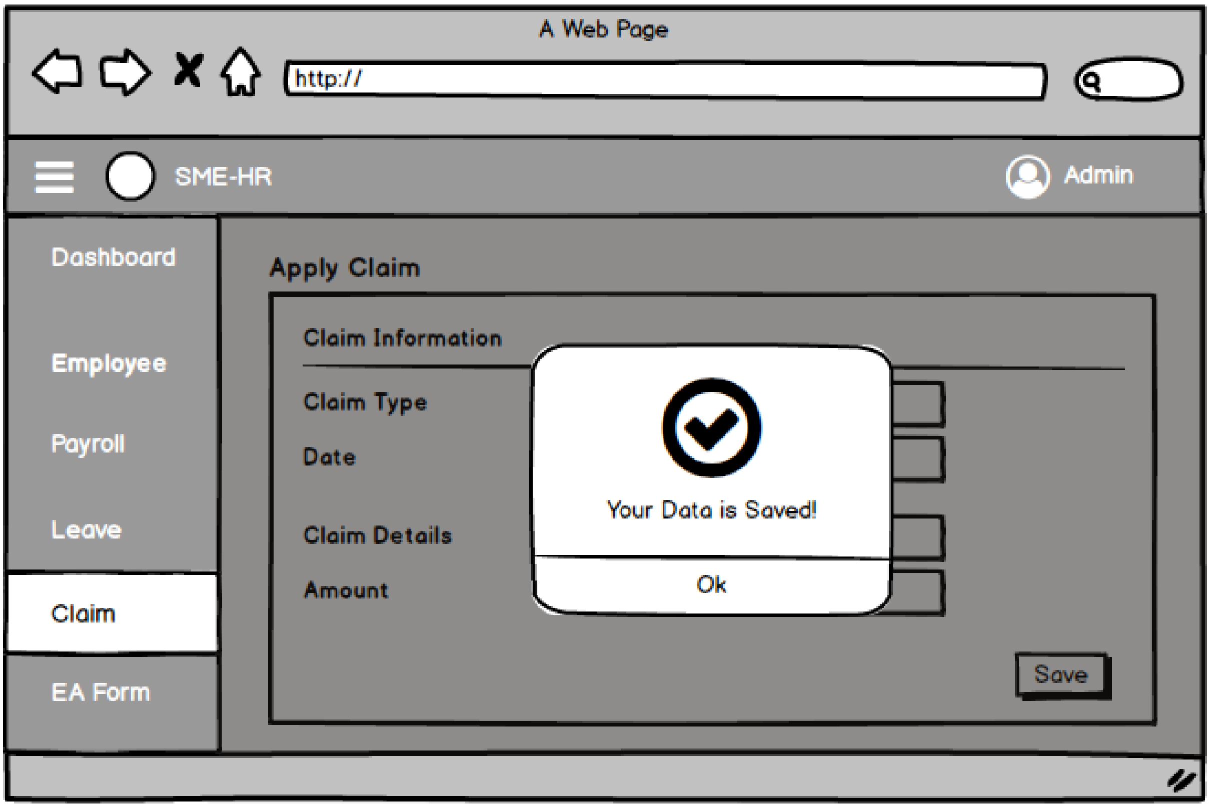
Claim Details

Amount

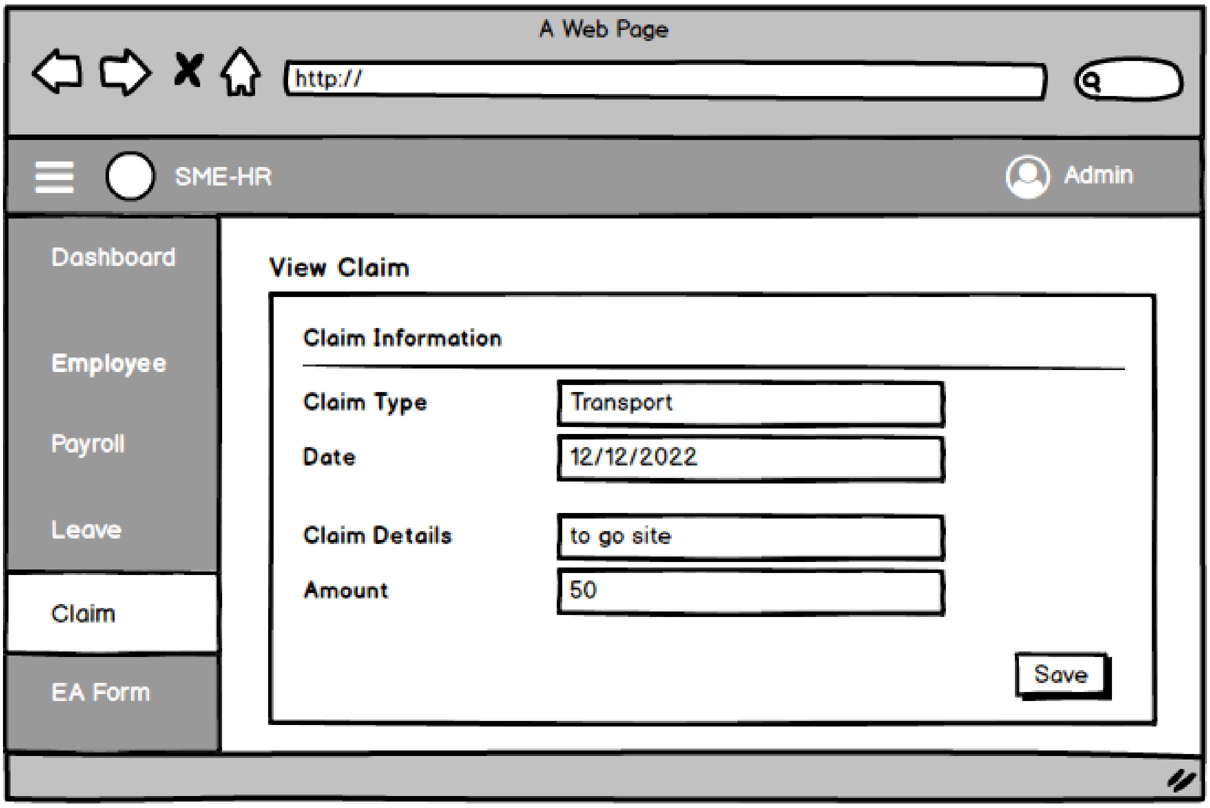
Appendix B-1.23: Add Claim Page



Appendix B-1.24: Add Claim Confirmation



Appendix B-1.25: Add Claim Success

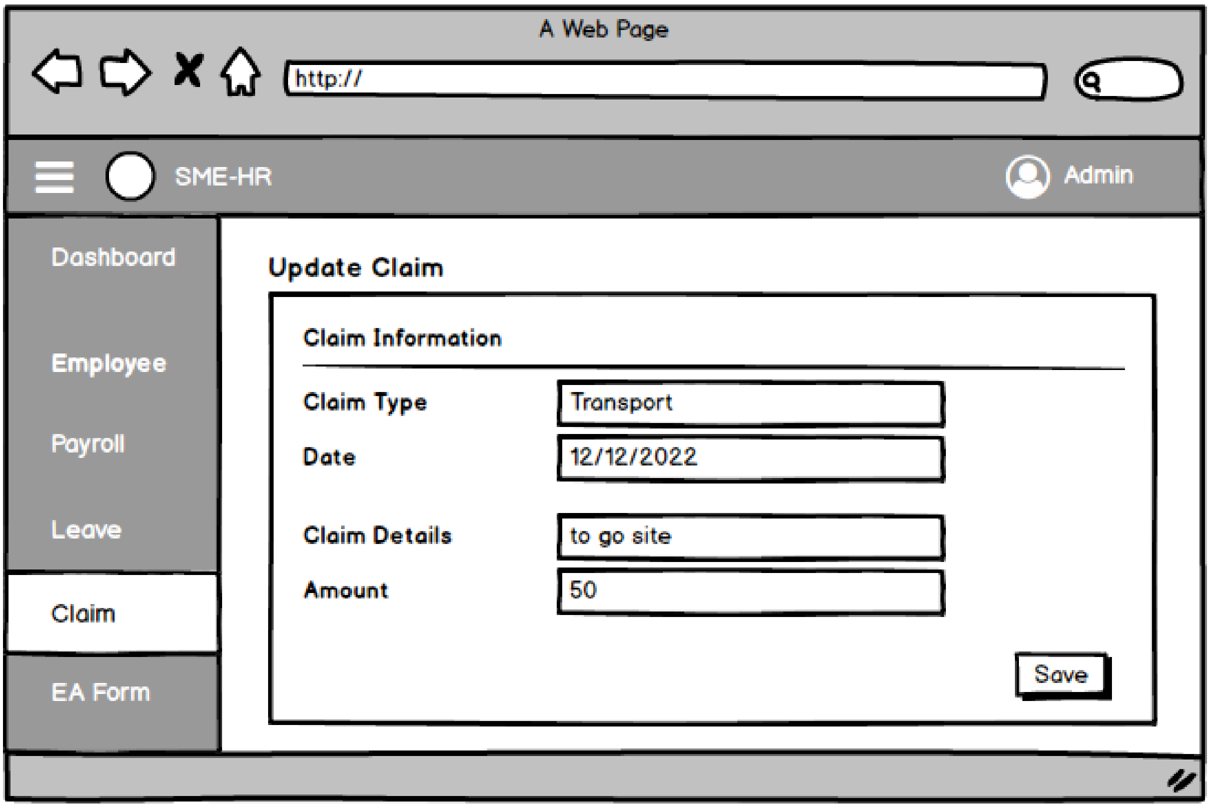


View Claim

Claim Information

Claim Type	Transport
Date	12/12/2022
Claim Details	to go site
Amount	50

Appendix B-1.26: View Claim

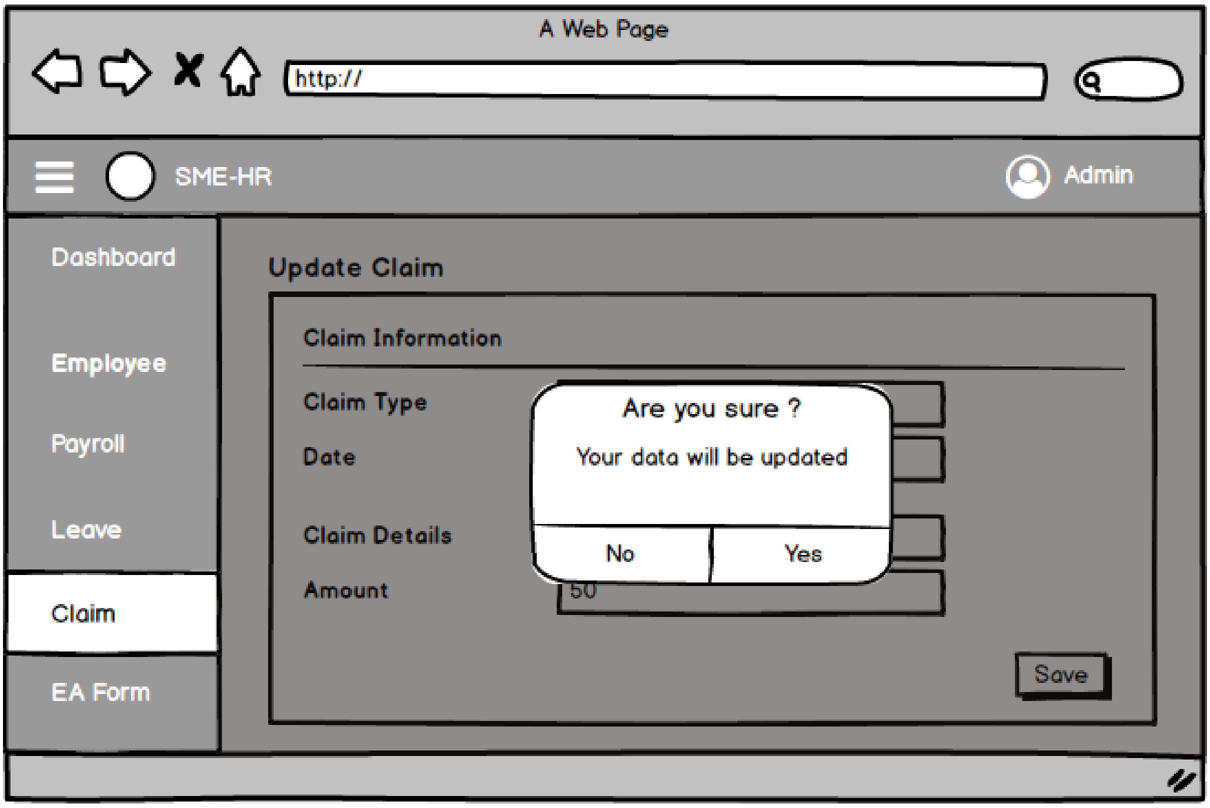


Update Claim

Claim Information

Claim Type	Transport
Date	12/12/2022
Claim Details	to go site
Amount	50

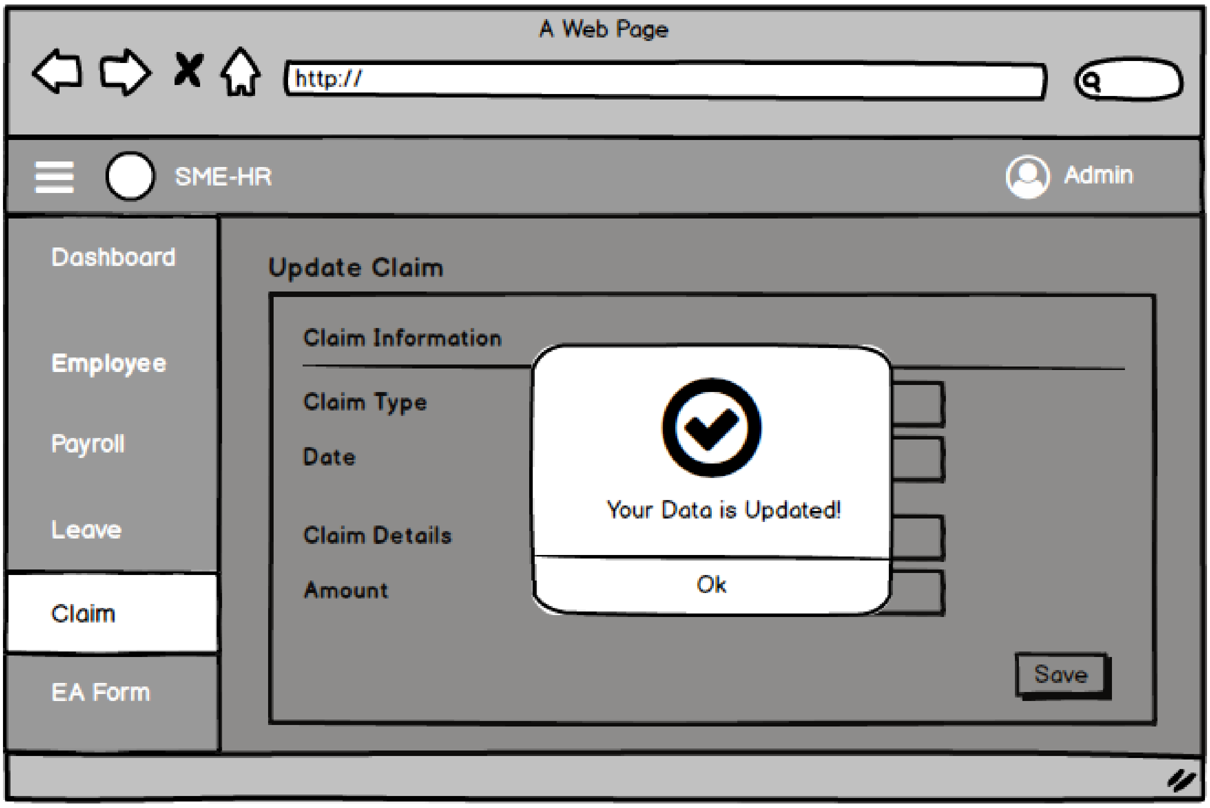
Appendix B-1.27: Update Claim Page



Are you sure ?

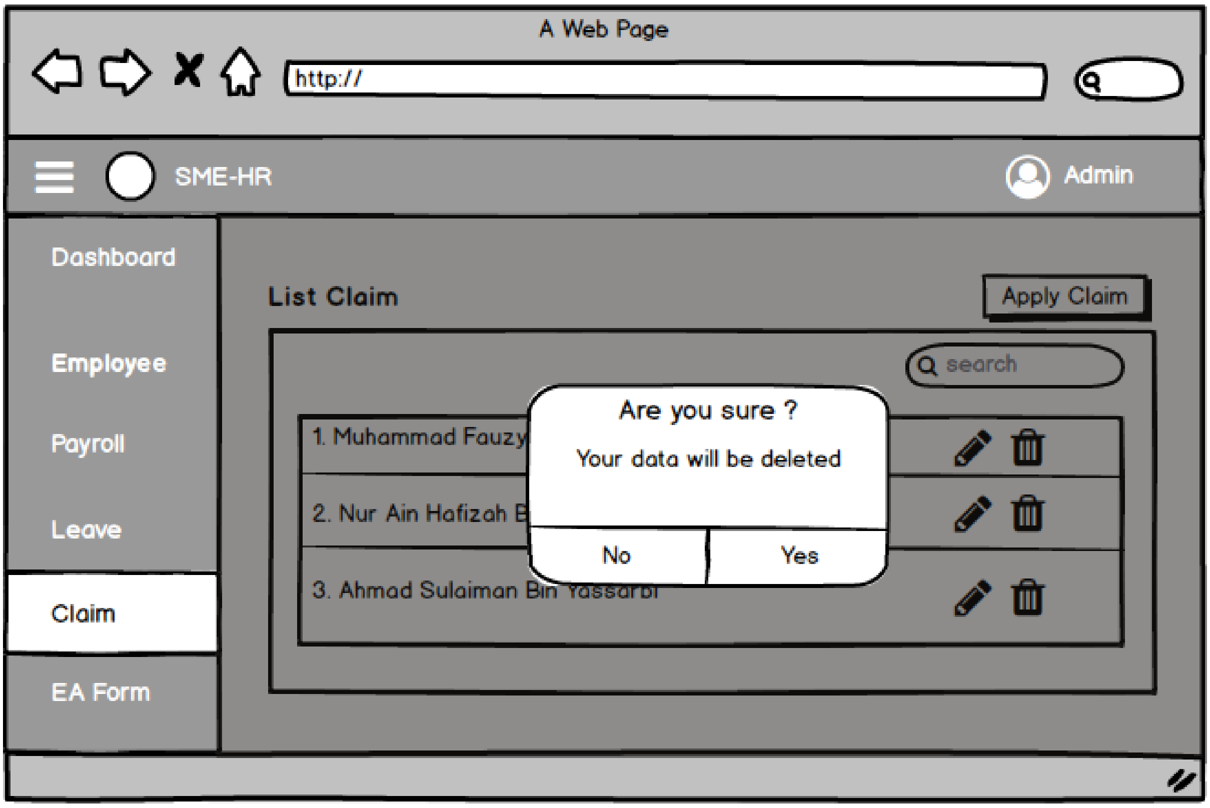
Your data will be updated

Appendix B-1.28: Update Claim Confirmation



Your Data is Updated!

Appendix B-1.29: Update Claim Success



1. Muhammad Fauzy

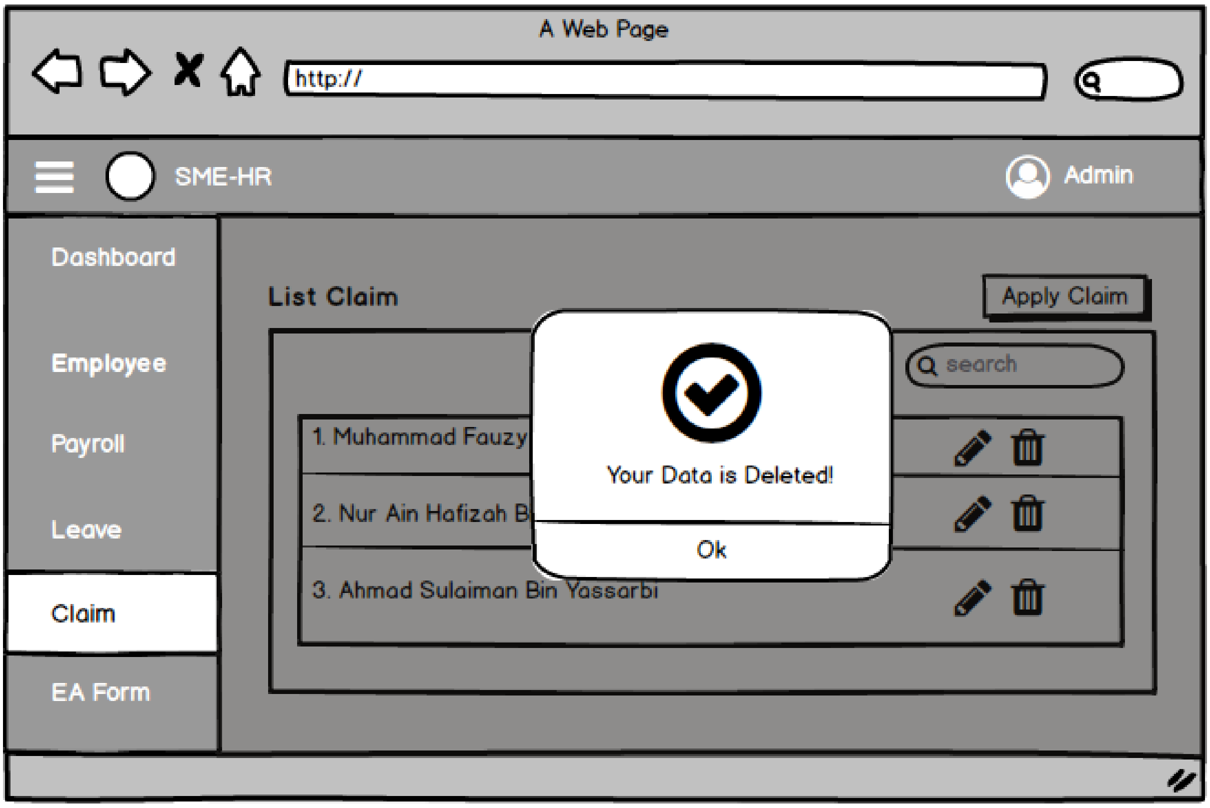
2. Nur Ain Hafizah B

3. Ahmad Sulaiman Bin Yassarbi

Are you sure ?

Your data will be deleted

Appendix B-1.30: Delete Claim Confirmation



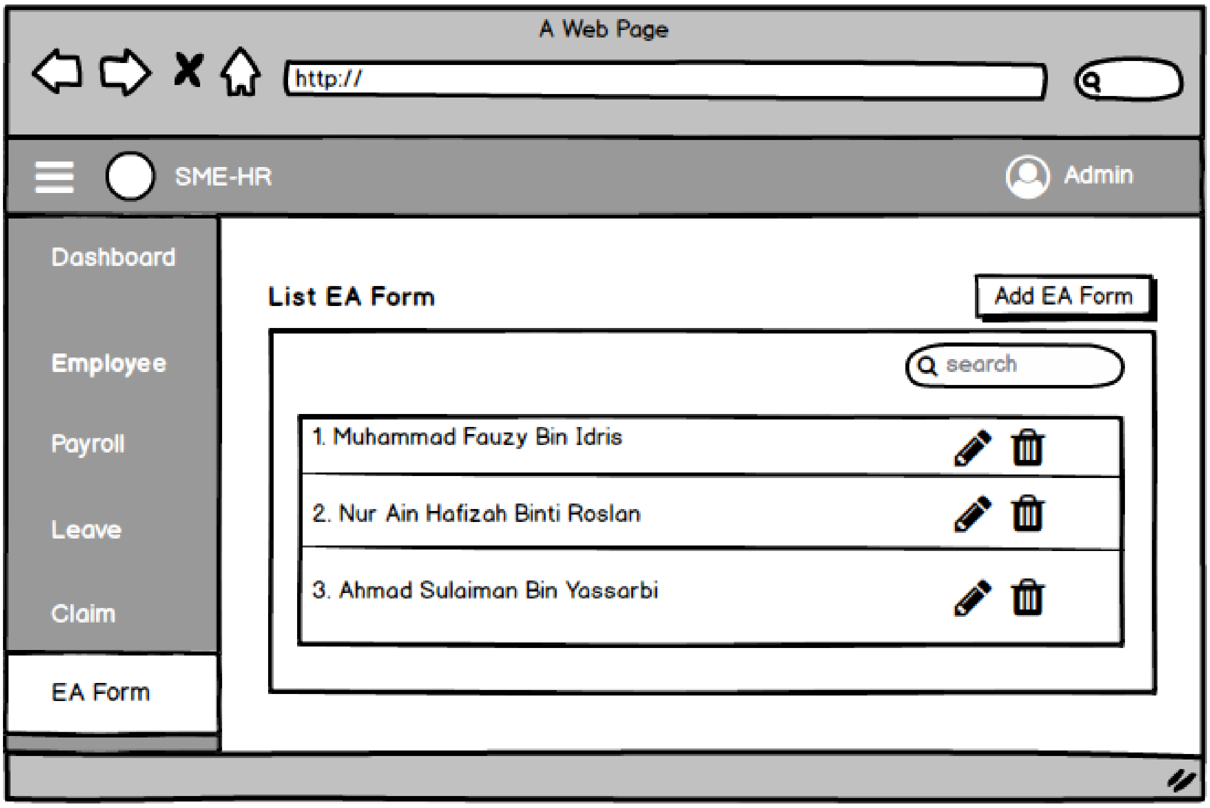
1. Muhammad Fauzy

2. Nur Ain Hafizah B

3. Ahmad Sulaiman Bin Yassarbi

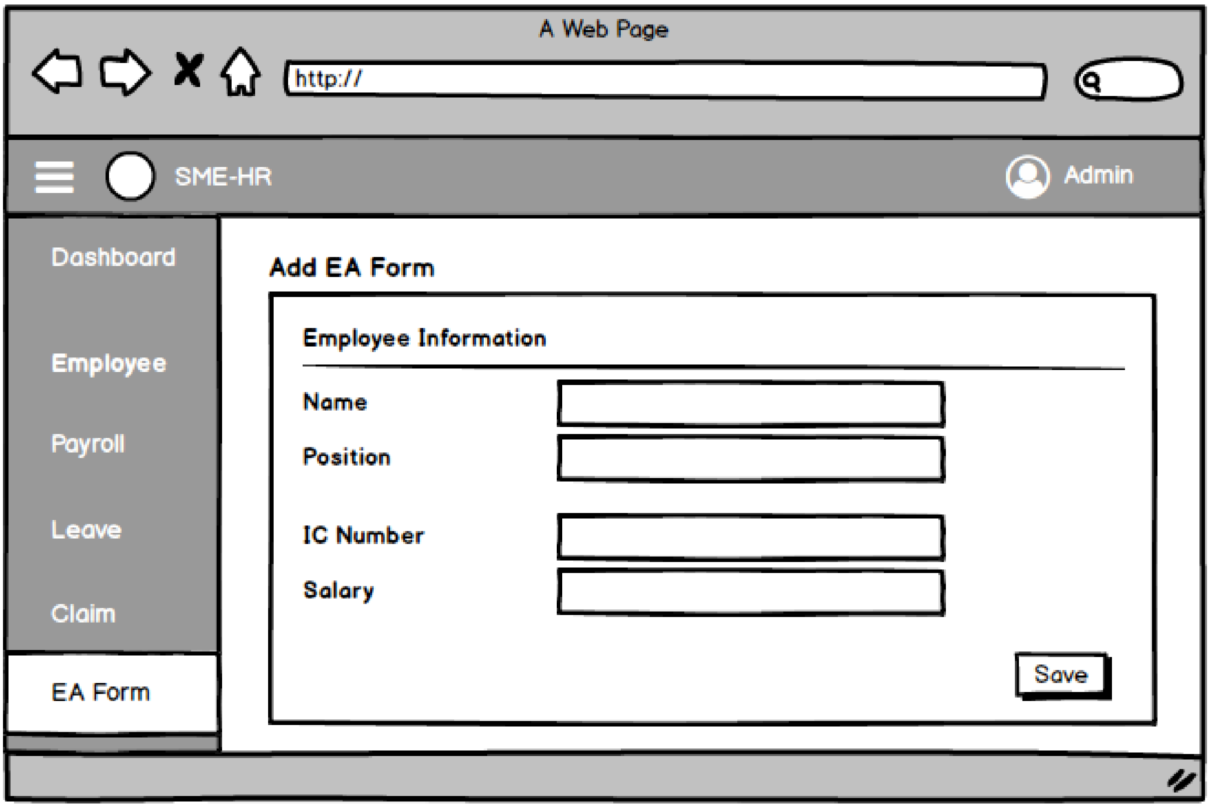
Your Data is Deleted!

Appendix B-1.31: Delete Claim Success



1. Muhammad Fauzy Bin Idris
2. Nur Ain Hafizah Binti Roslan
3. Ahmad Sulaiman Bin Yassarbi

Appendix B-1.32: EA Form Page

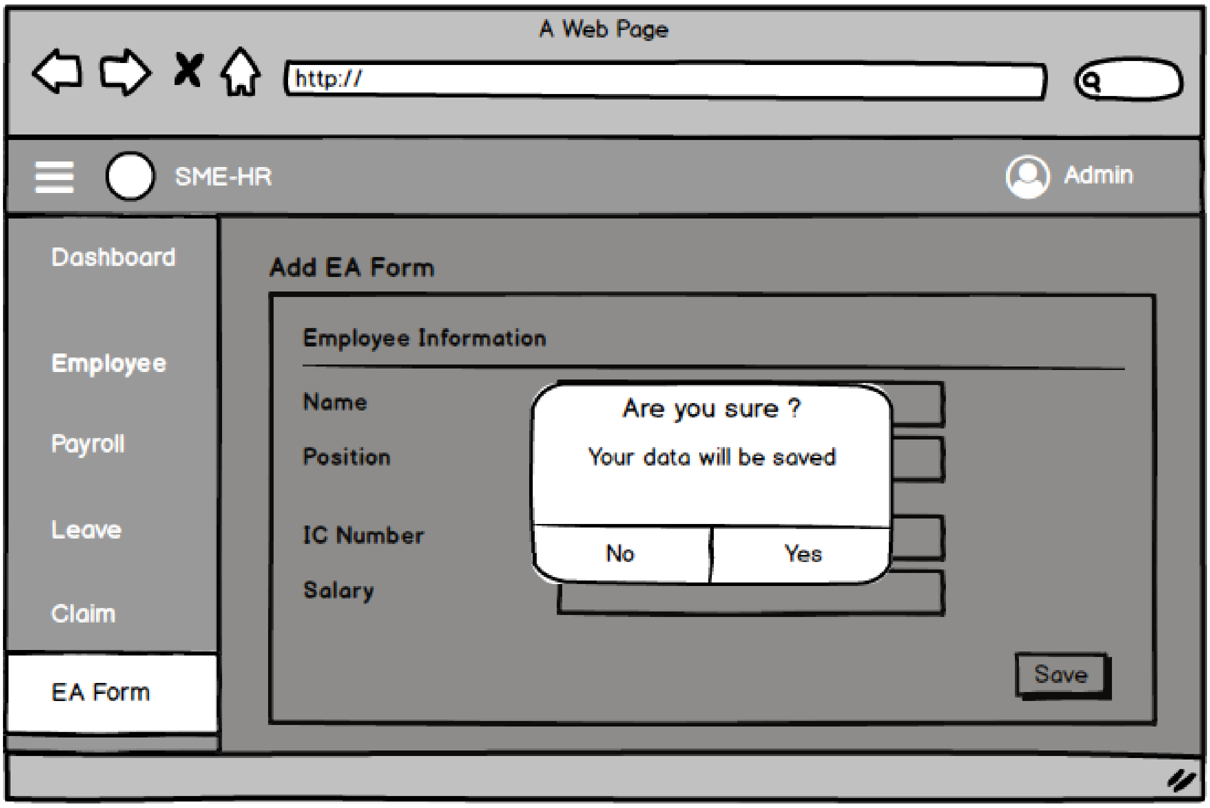


Add EA Form

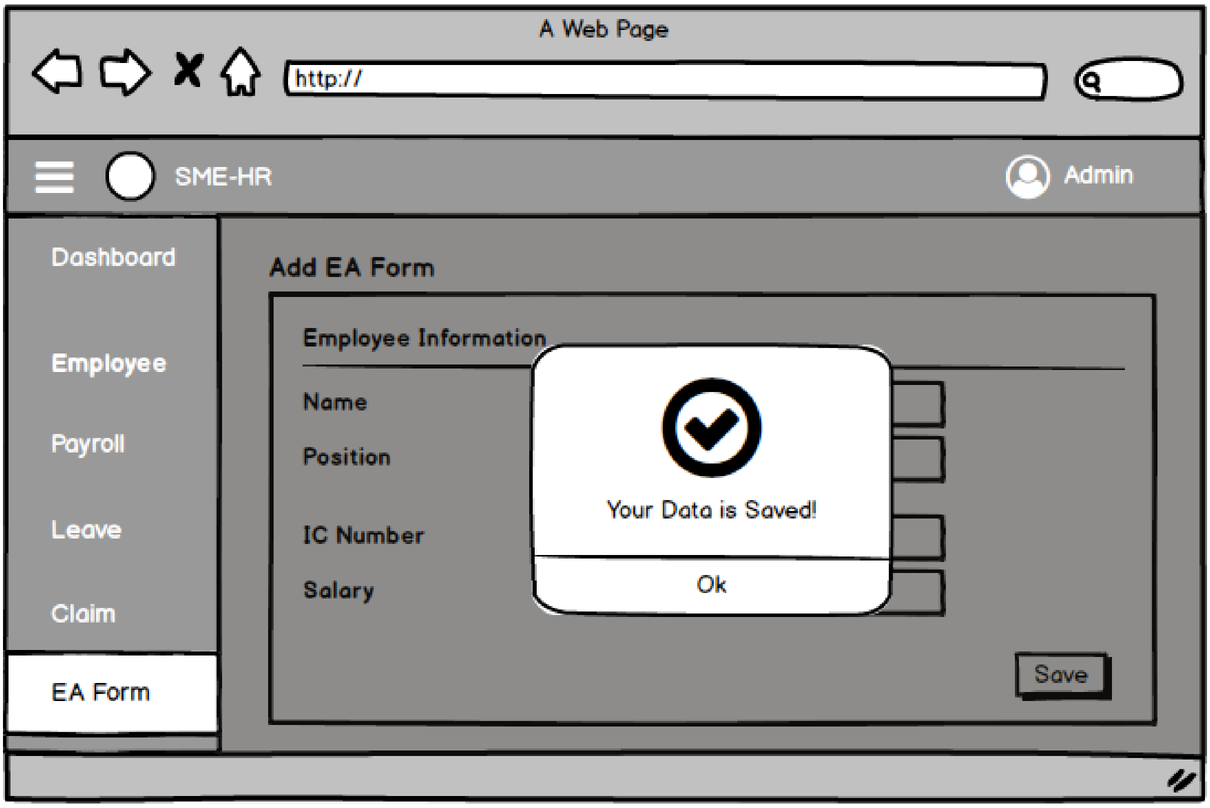
Employee Information

Name
Position
IC Number
Salary

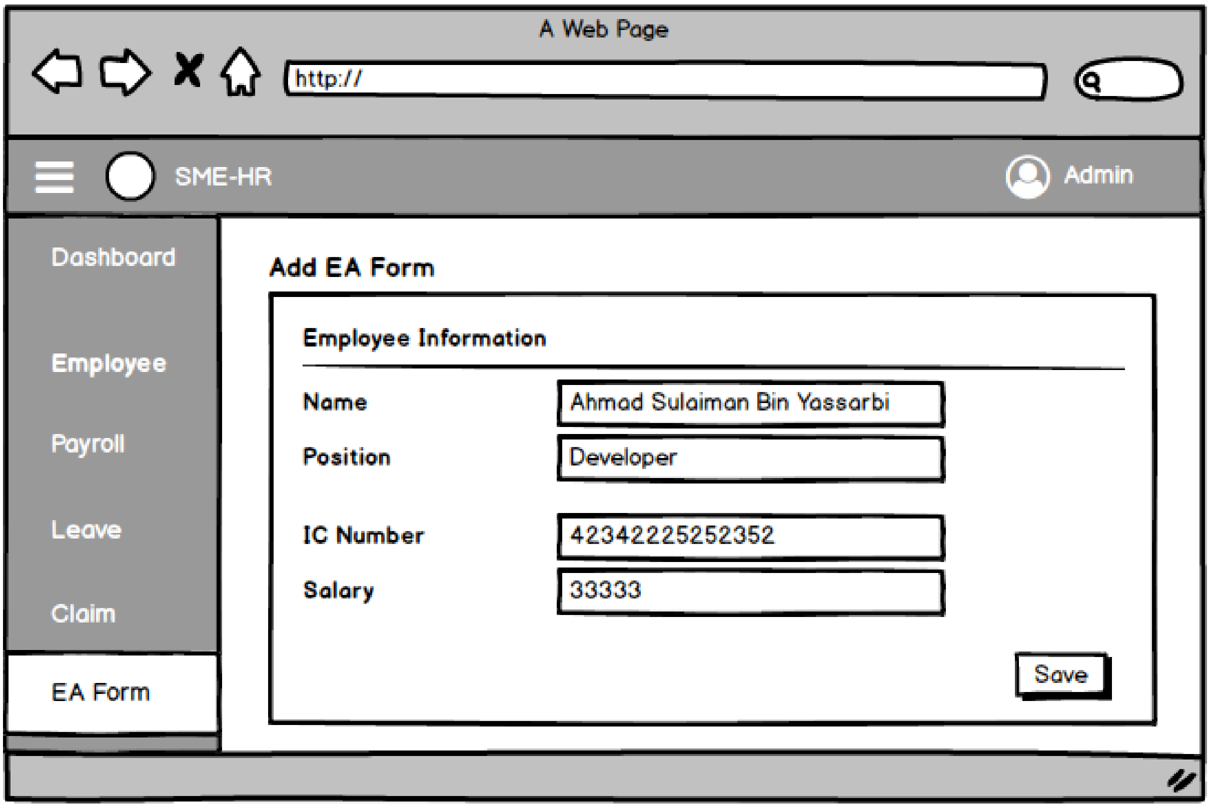
Appendix B-1.33: Add EA Form Page



Appendix B-1.34: Add EA Form Confirmation



Appendix B-1.35: Add EA Form Success

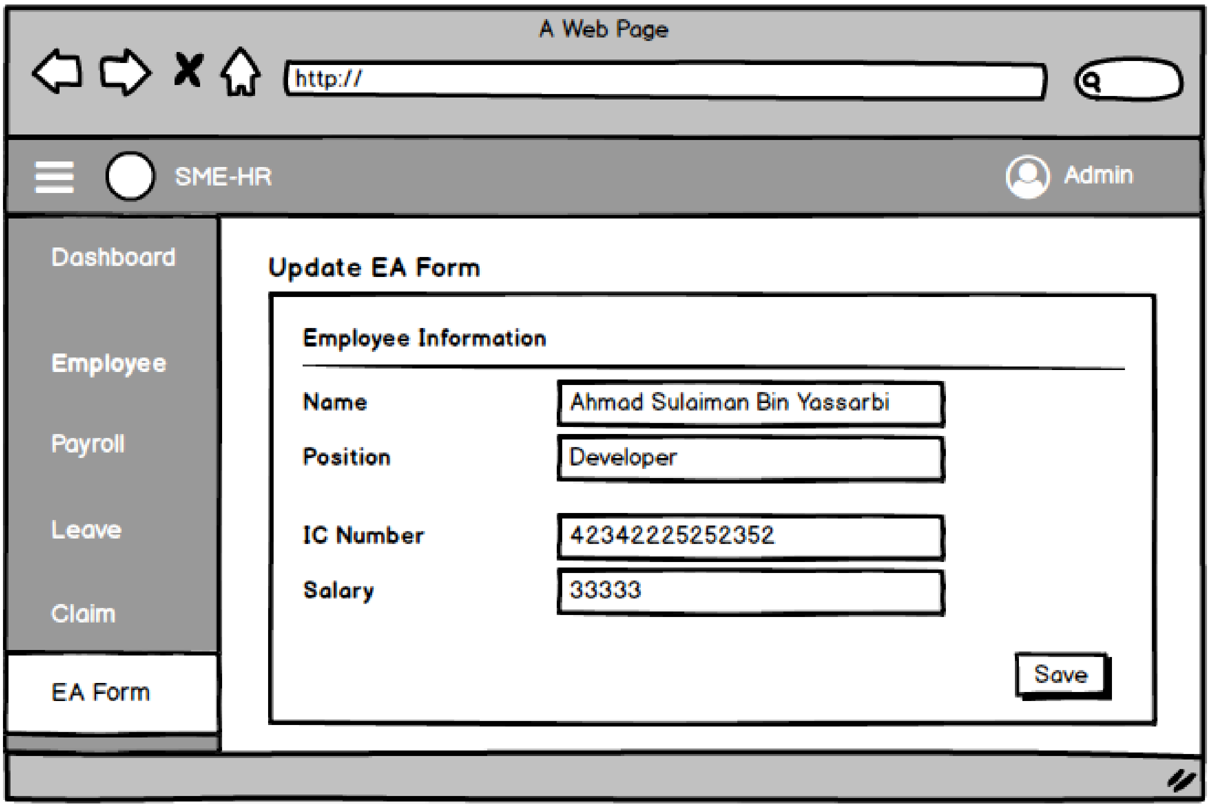


Add EA Form

Employee Information

Name	Ahmad Sulaiman Bin Yassarbi
Position	Developer
IC Number	42342225252352
Salary	33333

Appendix B-1.36: View EA Form

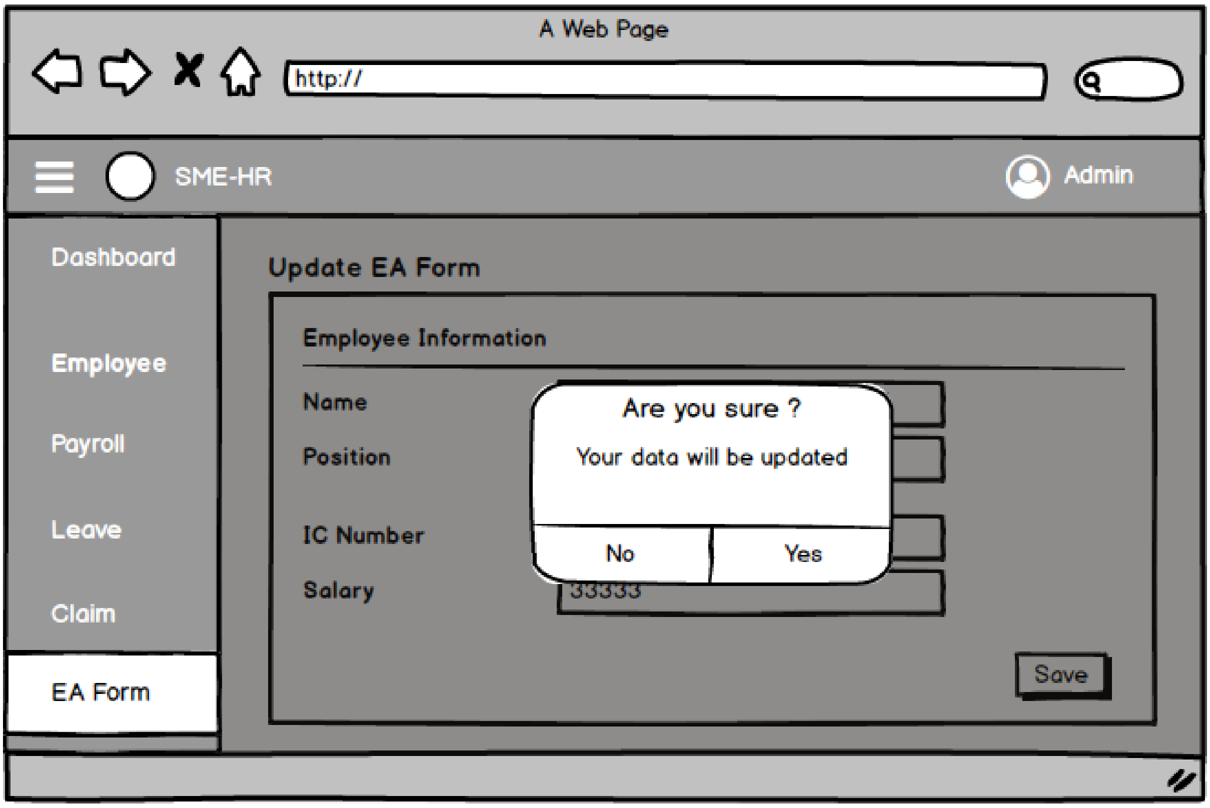


Update EA Form

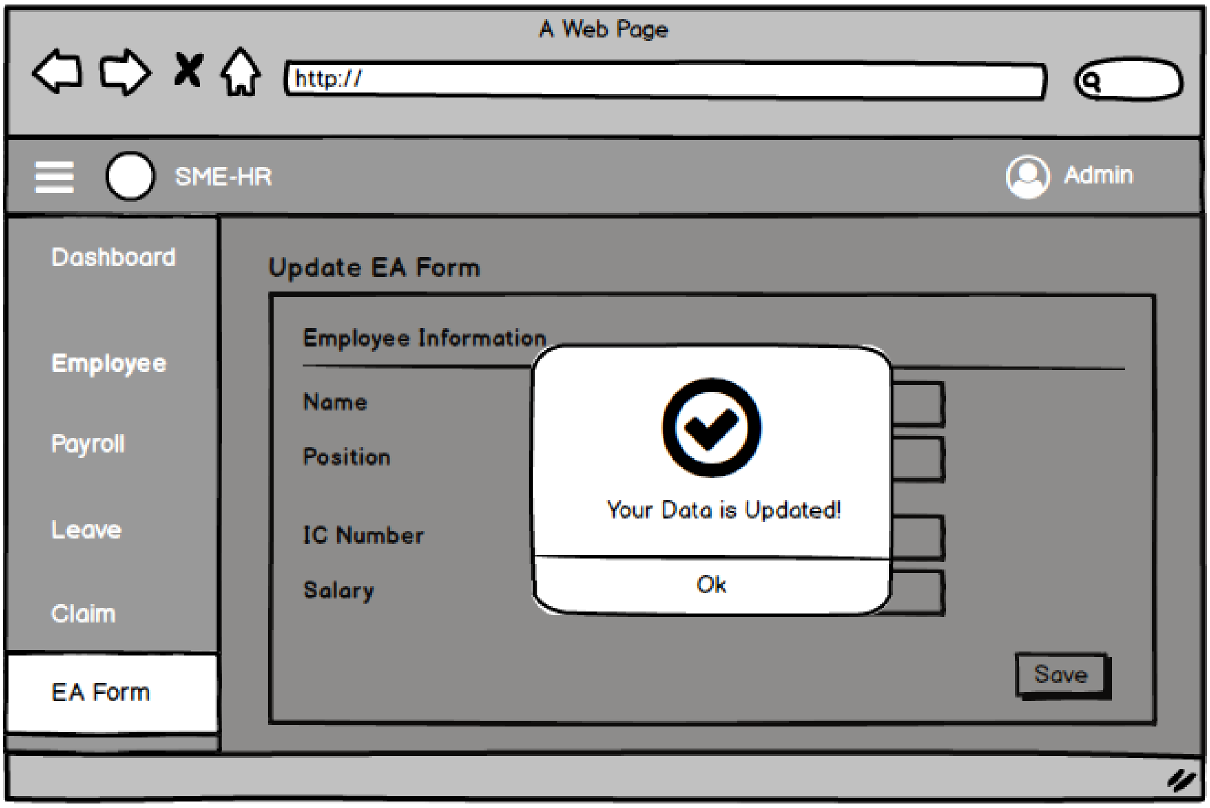
Employee Information

Name	Ahmad Sulaiman Bin Yassarbi
Position	Developer
IC Number	42342225252352
Salary	33333

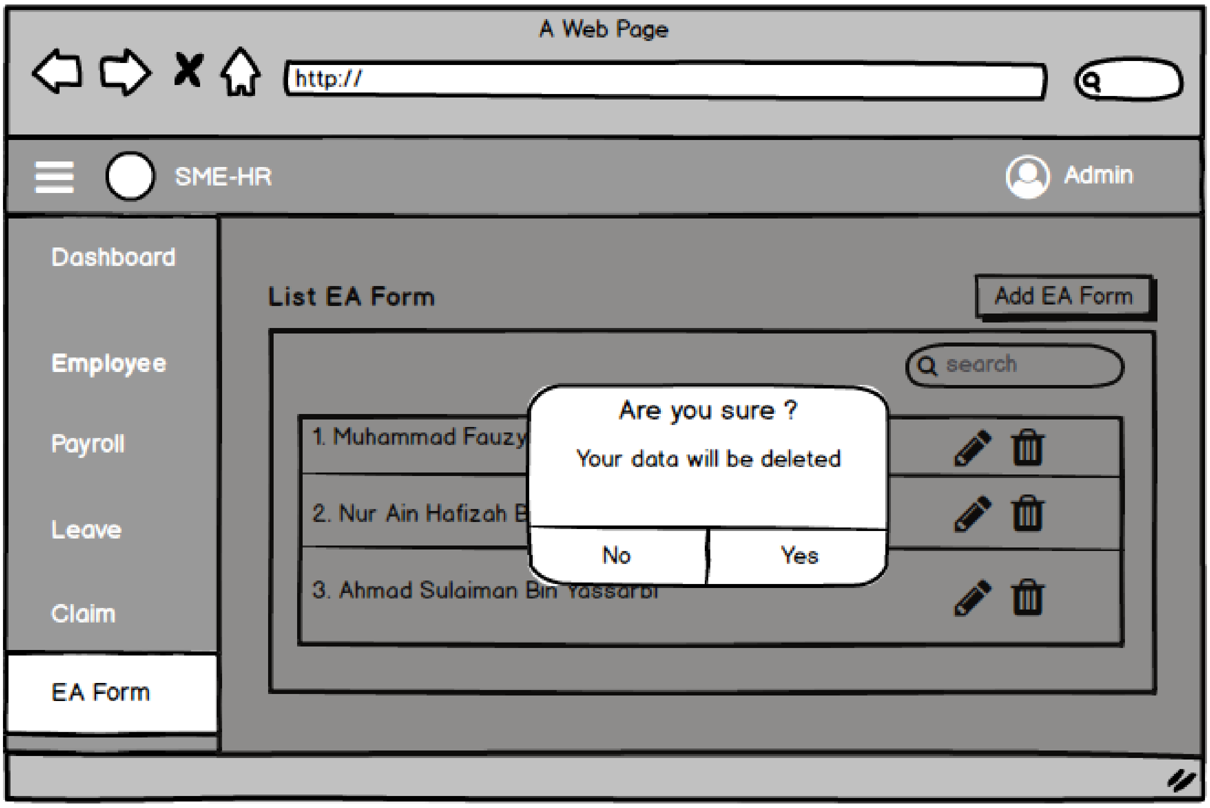
Appendix B-1.37: Update EA Form Page



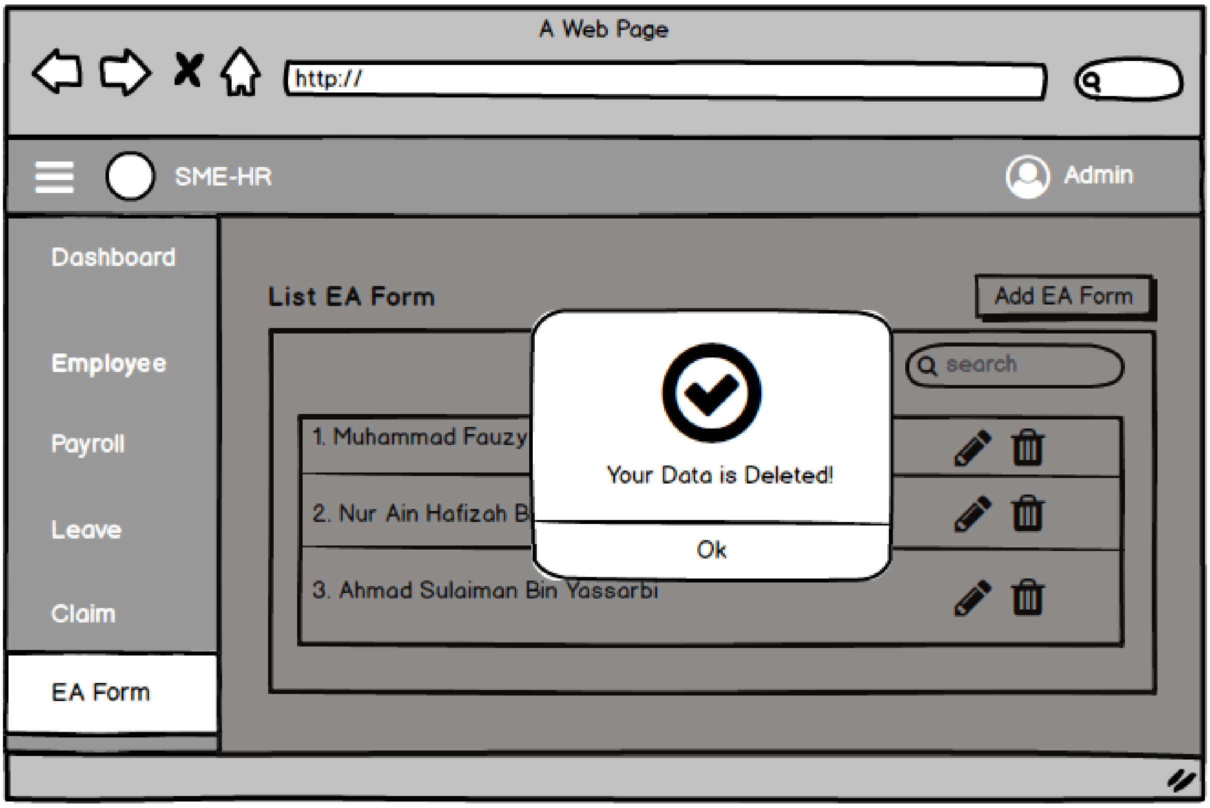
Appendix B-1.38: Update EA Form Confirmation



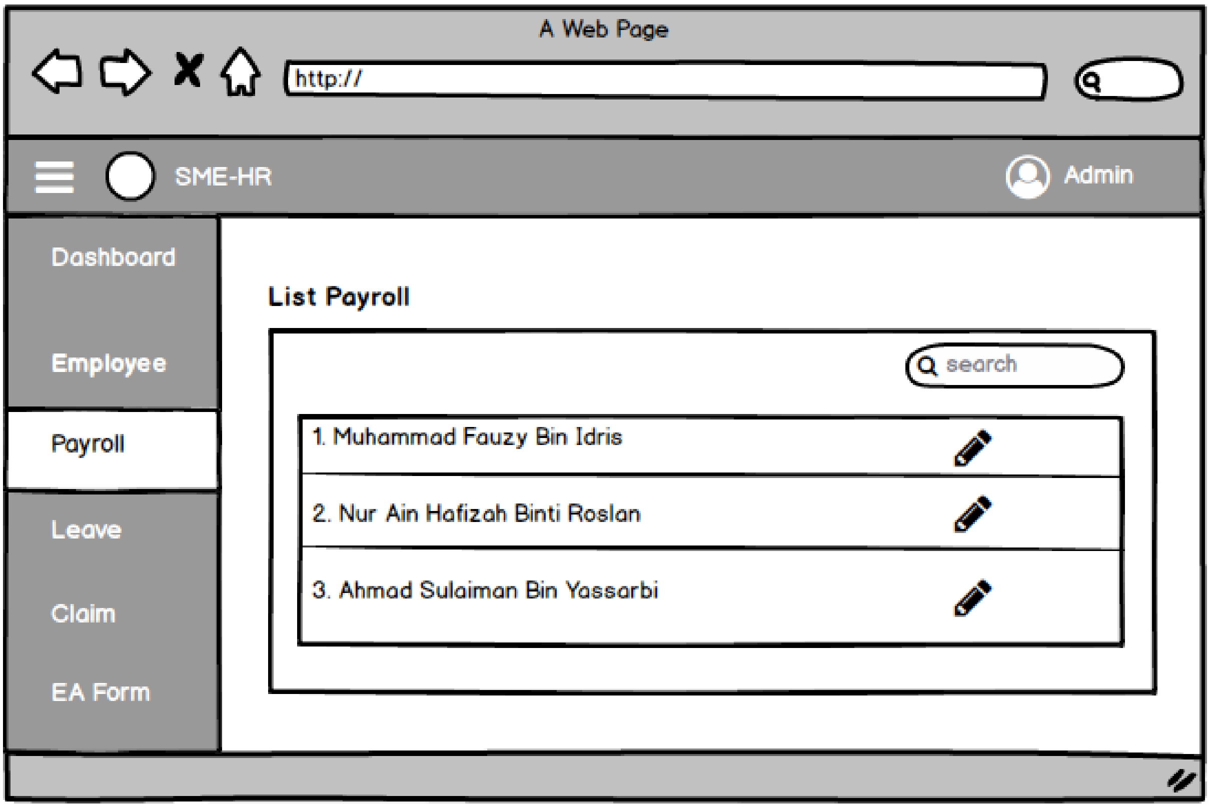
Appendix B-1.39: Update EA Form Success



Appendix B-1.40: Delete EA Form Confirmation



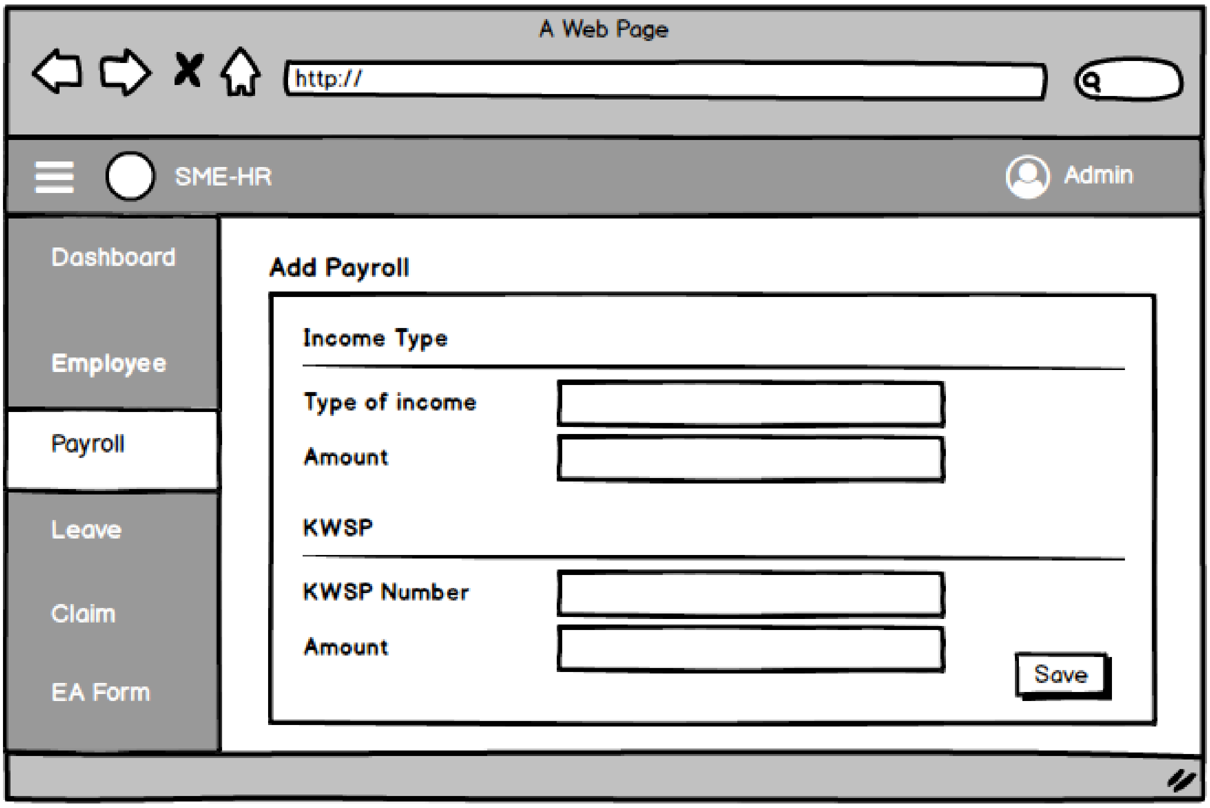
Appendix B-1.41: Delete EA Form Success



List Payroll

1. Muhammad Fauzy Bin Idris
2. Nur Ain Hafizah Binti Roslan
3. Ahmad Sulaiman Bin Yassarbi

Appendix B-1.42: Payroll Page



Add Payroll

Income Type

Type of income

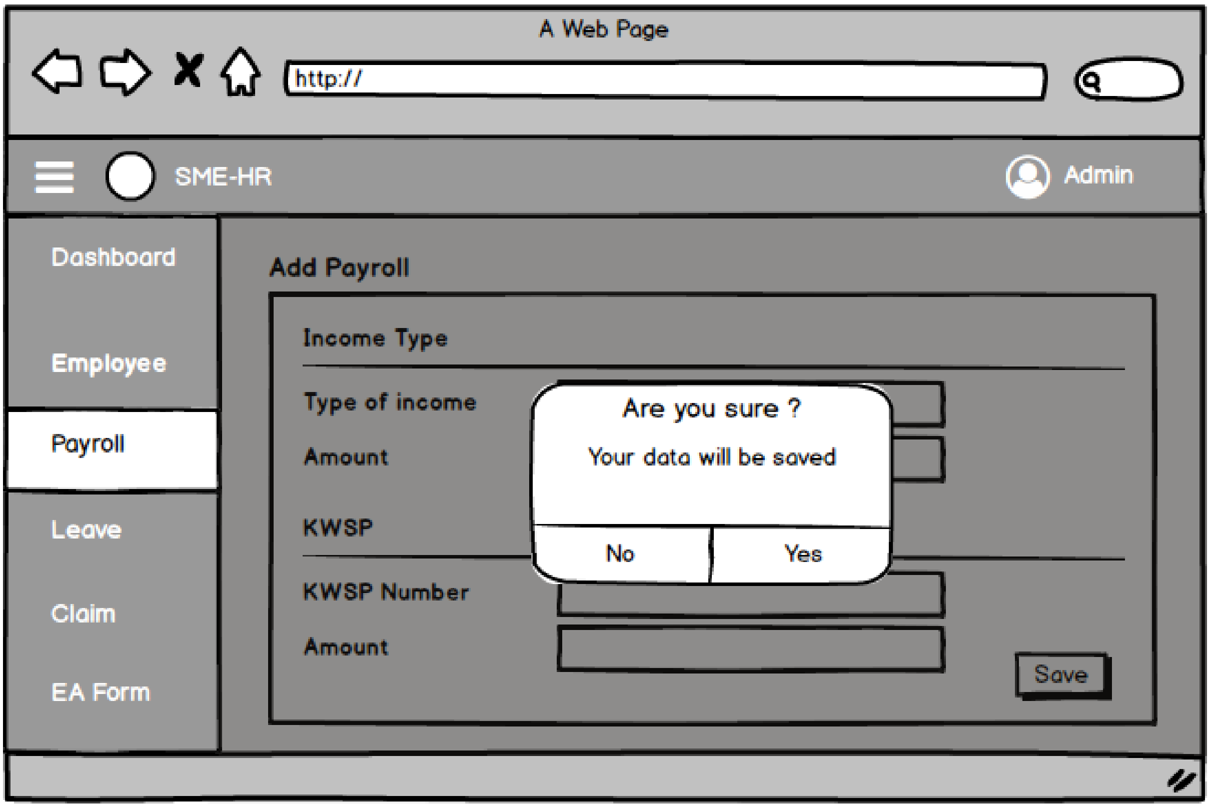
Amount

KWSP

KWSP Number

Amount

Appendix B-1.43: Add Payroll Page

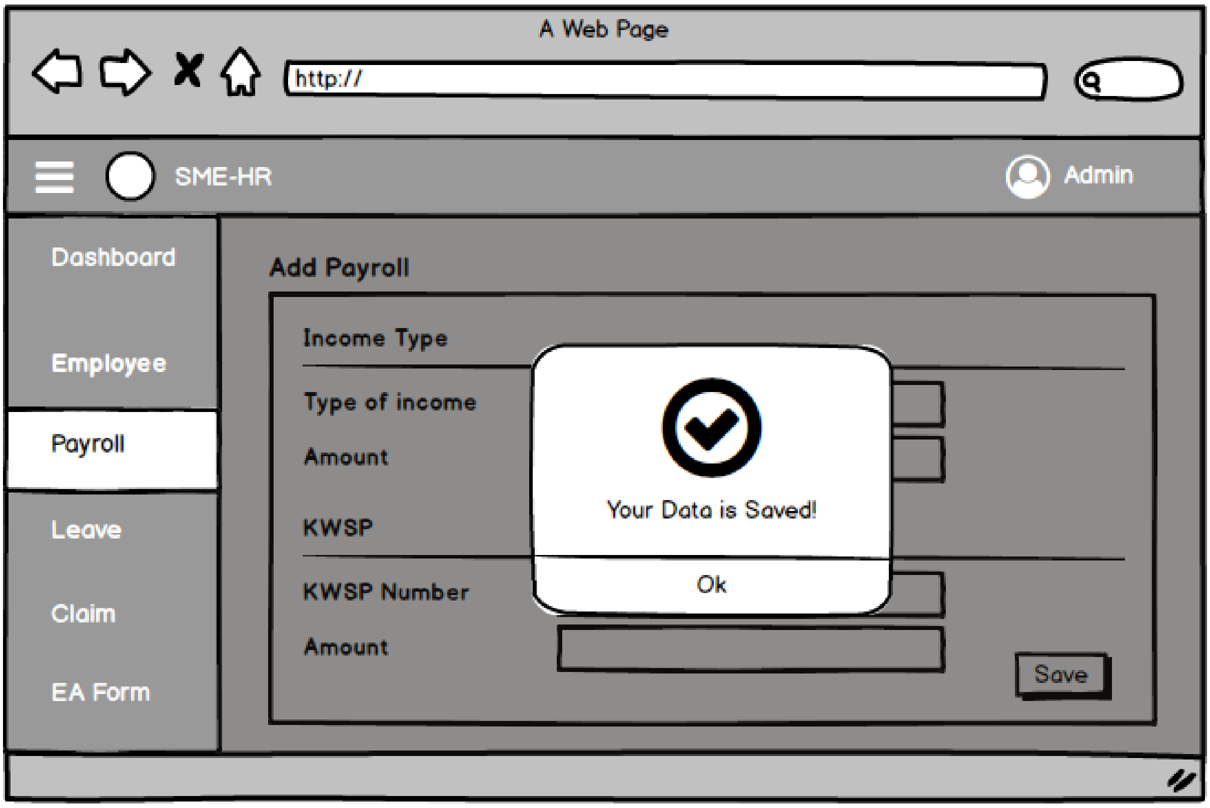


Are you sure ?

Your data will be saved

No Yes

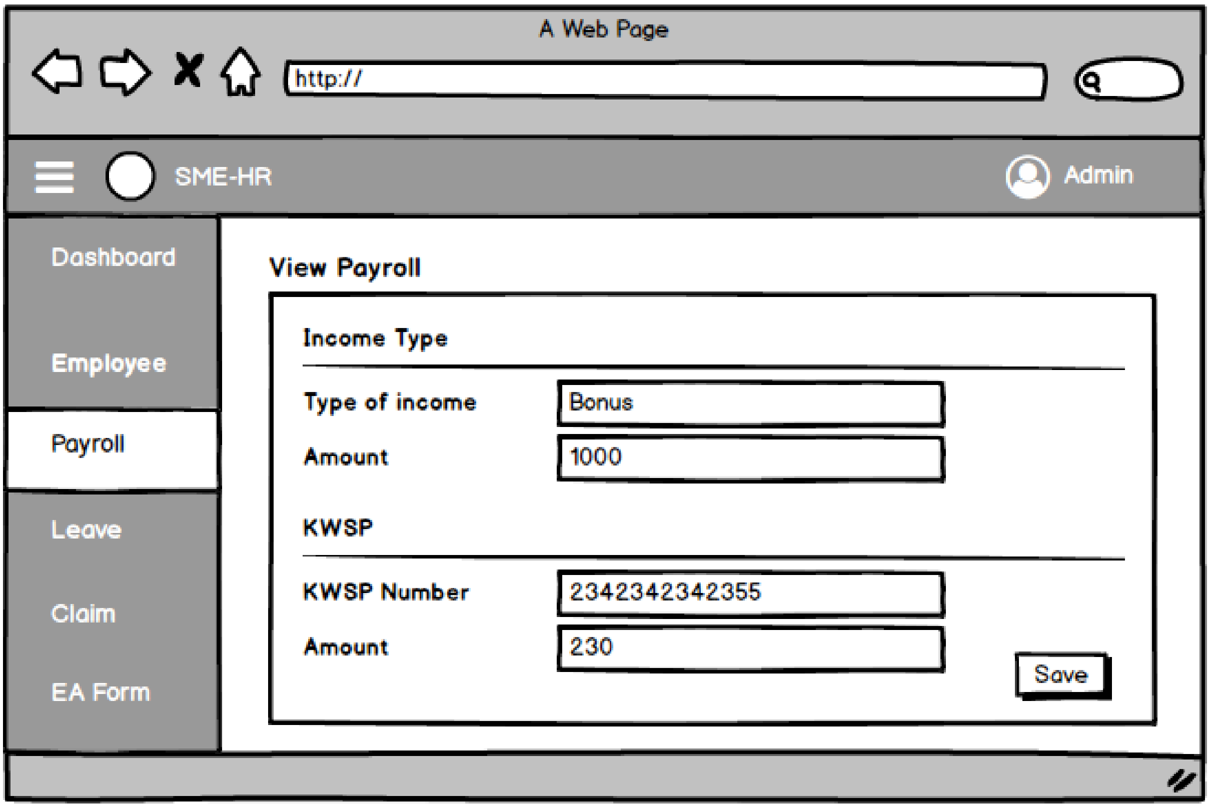
Appendix B-1.44: Add Payroll Confirmation



Your Data is Saved!

Ok

Appendix B-1.45: Add Payroll Success



View Payroll

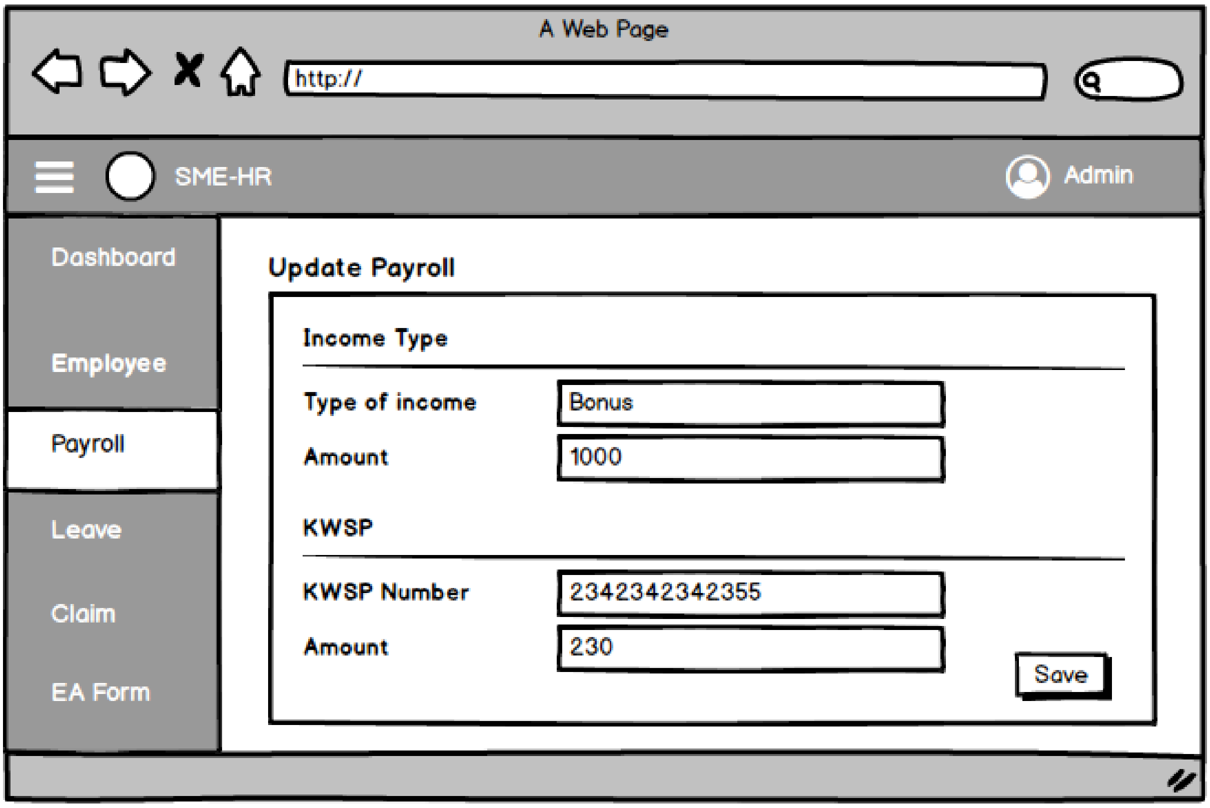
Income Type

Type of income	Bonus
Amount	1000

KWSP

KWSP Number	2342342342355
Amount	230

Appendix B-1.46: View Payroll



Update Payroll

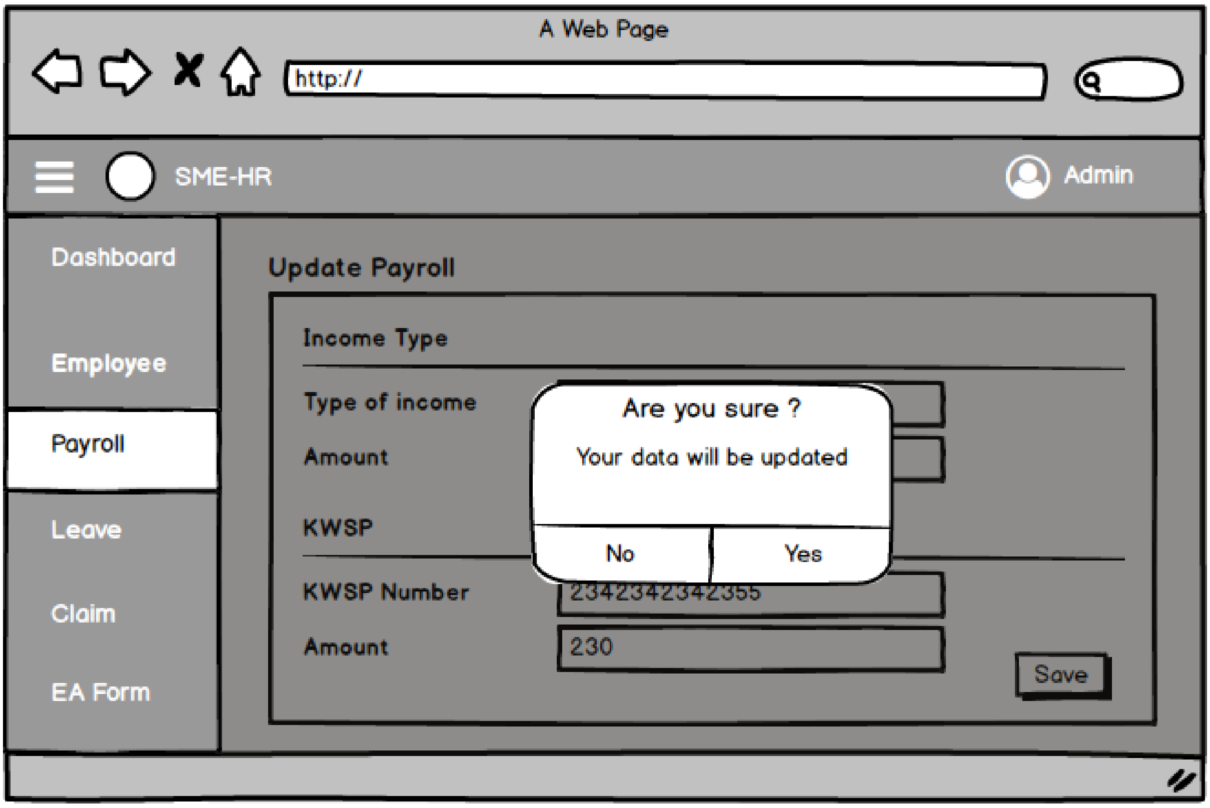
Income Type

Type of income	Bonus
Amount	1000

KWSP

KWSP Number	2342342342355
Amount	230

Appendix B-1.47: Update Payroll



Update Payroll

Income Type

Type of income

Amount

KWSP

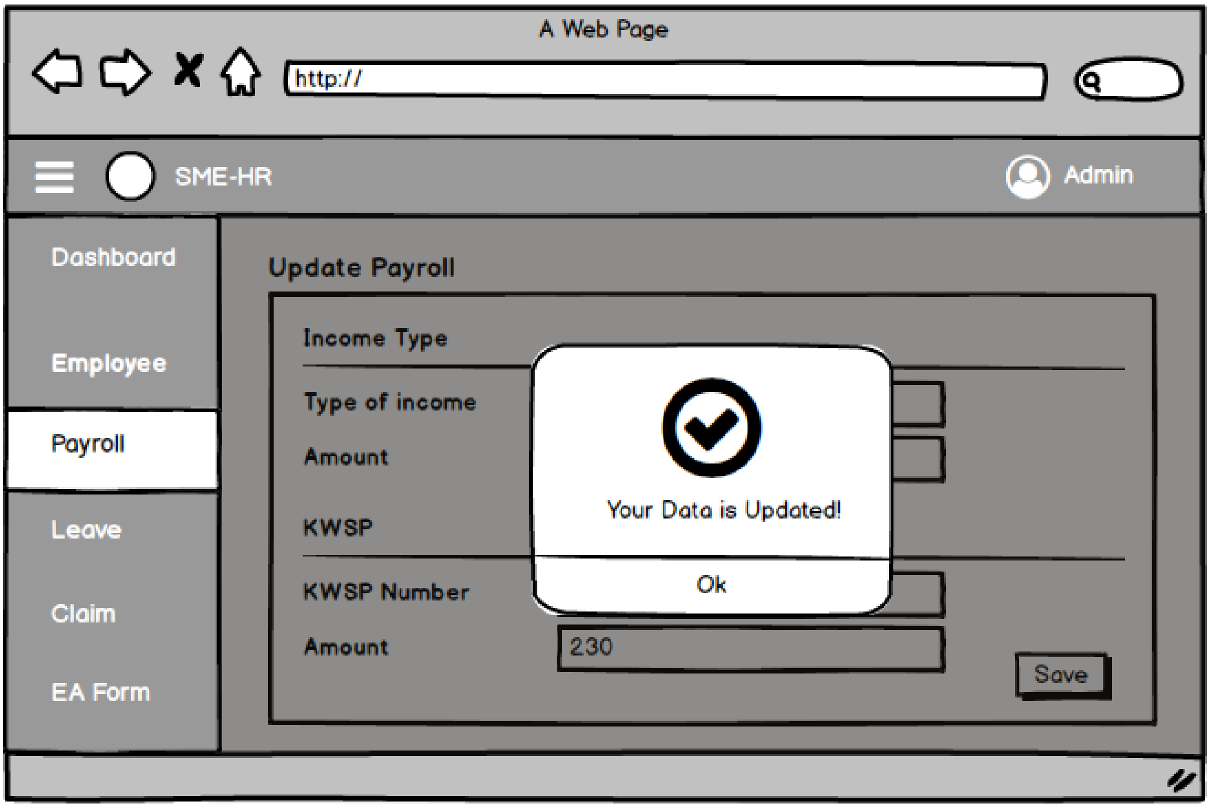
KWSP Number 2342342342355

Amount 230

Are you sure ?

Your data will be updated

Appendix B-1.48: Update Payroll Confirmation



Update Payroll

Income Type

Type of income

Amount

KWSP

KWSP Number

Amount 230

Your Data is Updated!

Appendix B-1.49: Update Payroll Success

APPENDIX C
SOFTWARE DESIGN DOCUMENT

Universiti Malaysia PAHANG

Engineering • Technology • Creativity

SDD Document
SEM I 20222023

SME-HR System

Name

1. MUHAMMAD TAUFIQ BIN MOHD JASLAN [CB20166]

TABLE OF CONTENTS

1. INTRODUCTION 3

1.1 Purpose 3

1.2 System Identification 3

1.3 System Overview 3

1.4 Referenced Document 4

2. DATA DESIGN 5

2.1 Entity Relationship Diagram (ERD) 5

2.2 Data Dictionary 6

3. GENERAL ARCHITECTURE 1

3.1 Layered Architecture 1

3.1.1 View Package 1

3.1.2 Controller Package 6

3.1.3 Model Package 6

4. DETAIL DESIGN 8

4.1 Manage Employee [SDD-REQ-100] 8

4.2 Manage Leave [SDD-REQ-200] 15

4.3 Manage Claim [SDD-REQ-300] 25

4.4 Manage Payroll [SDD-REQ-400] 32

4.5 Manage EA Form [SDD-REQ-500] 40

5. REQUIREMENT TRACEABILITY 47

5.1 Login [SME-HR-REQ-001] 47

5.2 Manage Employee [SME-HR-REQ-002] 47

5.3 View Employee [SME-HR-REQ-003] 47

5.4 Manage Profile [SME-HR-REQ-004] 48

5.5 Manage Payroll [SME-HR-REQ-005] 48

5.6 View Payroll [SME-HR-REQ-006] 48

5.7 Manage Leave [SME-HR-REQ-007] 49

5.8 Manage Claim [SME-HR-REQ-008] 50

5.9 Manage EA Form [SME-HR-REQ-009] 50

5.10 View EA Form [SME-HR-REQ-010] 51

1. INTRODUCTION

1.1 Purpose

The purpose behind doing this Software Design Document (SDD) is used to gather and analyze all assorted ideas that have come up to develop the SME-HR system. These requirements will regard consumers and will provide a detailed overview, parameters, the scope of work, and the goals of the SME-HR System. This document also describes this system's target audience and its user interface, hardware, and software requirements.

This document is likewise used to establish the basis of the details for an agreement among clients and software developers on how the software should work. It characterizes how our client, team, and audience see the item and its functionality. It gives a reference to software designers including all highlights to client requirements. In any case, it helps any designer and developer to diminish later upgrades and prevent software project failure.

1.2 System Identification

This Software Requirement Specification (SRS) belongs to the “SME-HR System” (SME-HR)

System title	: SME-HR System
System abbreviation	: SME-HR
System identification number	: S01-SME-HR-2023
Version number	: V01
Release number	: 2023

The SME-HR is used to record system abbreviation, which is the abbreviation for the system name, SME-HR system which is Small and Medium-Sized Enterprises Human Resources System. At the same time, Version 1 is the format for recording the version number for this system, which is the first system, and 2023 is the format to document the release number, which is the year 2023. In addition, the S01-SME-HR-2023 format is used to record the system identification number. S01 stands for system 1, the first system. Besides that, SME-HR stands for Small and Medium-Sized Enterprises Human Resources System, which is the project system name. Meanwhile, 2023 is the release year for this system.

1.3 System Overview

SME-HR is a system developed to manage Human Resources department activities in a systematic way. Thus, there are five modules available in the SME-HR system and are stated as below:

- Module 1: Manage Employee.
- Module 2: Manage Payroll.
- Module 3: Manage Leave.
- Module 4: Manage Claim.
- Module 5: Manage EA Form

MySafe system will be developed in both web-based system and mobile applications. The web-based system is for admin and staff use only, while the mobile application system is for Malaysian Citizens and Malaysian Visitors use only.

For Module 1, which is manage employee, the Administrator can create an account for each staff in the company. Besides, the Administrator can also update information about the employee detail. As for the Administrator, they can view, update, delete and add employee's information meanwhile for Staff and Manager, they can view their own information.

For Module 2, which is manage payroll, the Administrator can add the salary information for each staff. Besides that, the Administrator also can update and delete the payroll information if the information is not correct. In addition, the Administrator can generate the pay slip by month for each employee. Next for Staff and Manager, they can view their own pay slip and print it.

For Module 3, which is manage leave, the Administrator can add the leave information such as the leave entitlement, leave type. Moreover, the administrator can apply for the leave. Administrator also can update and delete all the information about the leave. Next, Staff can apply for the leave. Besides that, the Staff also can update and delete their apply leave information. In addition, the Manager can approve or reject the leave application for each employee.

For Module 4, which is manage claim. The Administrator can add the claim information such as claim type. The Administrator also can apply for the claim, and they also can edit and delete the claim information. Besides that, in this module, Staff can apply for the claim, and they also can edit and delete their claim application. In addition, for Manager, they can view all the claim application and can approve or reject the application.

For Module 5, which is manage EA form. The Administrator can add the EA form for each employee. In addition, they also can delete and update all the EA form information if there is a mistake. Besides that, in this module, the Staff and Manager is able to view their own EA form.

1.4 Referenced Document

1. *Shamita, G. (2022). Implementation of Human Resources System In The Company. https://www.researchgate.net/publication/364335598*

2. *SOFTWARE REQUIREMENT SPECIFICATION (SRS) FK ITEM NUMBER VERSION NUMBER (Example SDP ABC 2008 VERSION 1.0)*. (n.d.). Retrieved December 19, 2021, from https://kalam.ump.edu.my/pluginfile.php/108257/mod_folder/content/0/SRS%20TEMPLATE%20SEM%20I%2020212022.pdf?forcedownload=1

3. Custom Software Requirements Specification Document - Belitsoft. (n.d.). Retrieved November 27, 2021, from https://belitsoft.com/custom-application-development-services/software-requirements-specification-document-example-international-standard

2. DATA DESIGN

2.1 Entity Relationship Diagram (ERD)

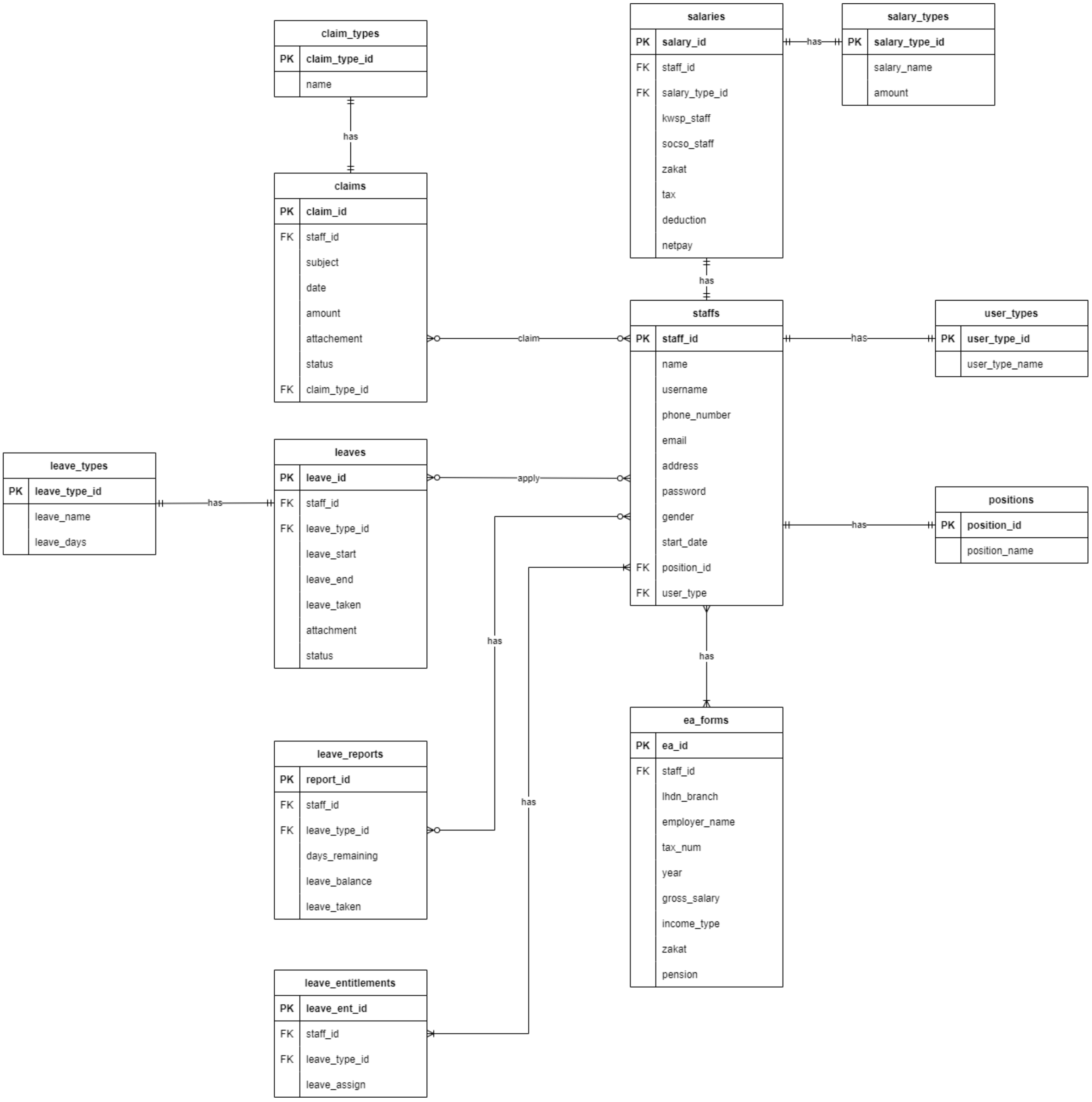


Figure 1: SME-HR System ERD

2.2 Data Dictionary

2.2.1 Staffs

Table 1: Staffs Data Dictionary

Field Name	Description	Data Type	Constraint
staff_id	ID for staff	INT	PK
name	Name of the staff	VARCHAR (50)	
username	Username of the staff	VARCHAR (20)	
phone_number	Phone number of the staff	VARCHAR (10)	
email	Email of the staff	VARCHAR (20)	
address	Address of the staff	VARCHAR (50)	
password	Password for staff	VARCHAR (20)	
gender	Gender of the staff	VARCHAR (10)	
start_date	Date when the staff start to work	DATE	
position_ID	Position of the staff	INT	FK
user_type	User type of the staff	INT	FK

2.2.2 User Types

Table 2: User Types Data Dictionary

Field Name	Description	Data Type	Constraint
user_type_id	ID for the user type	INT	PK
user_type_name	Name for the user type	VARCHAR (50)	

2.2.3 Positions

Table 3: Positions Data Dictionary

Field Name	Description	Data Type	Constraint
position_id	ID for the position	INT	PK
position_name	Name for the position	VARCHAR (50)	

2.2.4 Salaries

Table 4: Salaries Data Dictionary

Field Name	Description	Data Type	Constraint
salary_id	ID for the salary	INT	PK
staff_id	ID for the staff	INT	FK
salary_type_id	ID for the salary type	INT	FK
kwsp_staff	KWSP of the staff	DECIMAL (10,2)	
socso_staff	Socso of the staff	DECIMAL (10,2)	
zakat	Zakat of the staff	DECIMAL (10,2)	

deduction	Deduction of the staff	DECIMAL (10,2)	
netpay	Netpay of the staff	DECIMAL (10,2)	

2.2.5 Salary Types

Table 5: Salary Types Data Dictionary

Field Name	Description	Data Type	Constraint
salary_type_id	ID for the salary type	INT	PK
salary_name	Name for the salary type	VARCHAR (50)	
amount	Amount of the salary	DECIMAL (10,2)	

2.2.6 EA Forms

Table 6: EA Forms Data Dictionary

Field Name	Description	Data Type	Constraint
ea_id	ID for the position	INT	PK
staff_id	ID for the position	INT	FK
lhdn_branch	LHDN branch for the company	VARCHAR (50)	
employer_name	Employer name	VARCHAR (50)	
tax_num	Tax number of the staff	VARCAHR (50)	
year	Year for the EA Form	INT	
gross_salary	Gross salary of the staff	DECIMAL (10,2)	

income_type	Income type	VARCHAR (20)	
zakat	Zakat of the staff	DECIMAL (10,2)	
pension	Pension of the staff	DECIMAL (10,2)	

2.2.7 Claims

Table 7: Claims Data Dictionary

Field Name	Description	Data Type	Constraint
claim_id	ID for the claim	INT	PK
staff_id	ID for the position	INT	FK
subject	Subject of the claim	VARCHAR (50)	
date	Date for the claim	DATE	
amount	Amount of the claim	DECIMAL (10,2)	
attachment	Attachment or receipt for the claim	VARCHAR (20)	
status	Status for the claim	INT	
claim_type_id	ID for the claim type	INT	FK

2.2.8 Claim Types

Table 8: Claim Types Data Dictionary

Field Name	Description	Data Type	Constraint
claim_type_id	ID for the claim type	INT	PK
name	Name for the claim type	VARCHAR (30)	

2.2.9 Leaves

Table 9: Leaves Data Dictionary

Field Name	Description	Data Type	Constraint
leave_id	ID for the leave	INT	PK
staff_id	ID for the position	INT	FK
leave_type_id	ID for the leave type	INT	FK
leave_start	Date for the leave start	DATE	
leave_end	Date for the leave end	DATE	
leave_taken	Leave taken day	INT	
attachment	Attachment for the leave	VARCHAR (30)	
status	Status for the leave	INT	

2.2.10 Leave Reports

Table 10: Leave Reports Data Dictionary

Field Name	Description	Data Type	Constraint
report_id	ID for the leave report	INT	PK
staff_id	ID for the staff	INT	FK
leave_type_id	ID for leave type	VARCHAR (50)	

days_remaining	Days remaining for the leave	VARCHAR (50)	
leave_balance	Total leave balance	VARCAHR (50)	
leave_taken	Total leave taken	INT	

2.2.11 Leave Entitlements

Table 11: Leave Entitlements Data Dictionary

Field Name	**Description**	**Data Type**	**Constraint**
leave_ent_id	ID for the leave entitlement	INT	PK
staff_id	ID for the staff	INT	FK
leave_type_id	ID for the leave type	INT	FK
leave_assign	Total of leave assign	INT	

2.2.12 Leave Types

Table 12: Leave Types Data Dictionary

Field Name	**Description**	**Data Type**	**Constraint**
leave_type_id	ID for the leave type	INT	PK
leave_name	Name of the leave	VARCHAR (30)	
leave_days	Total for the leave	INT	

3. GENERAL ARCHITECTURE

3.1 Layered Architecture

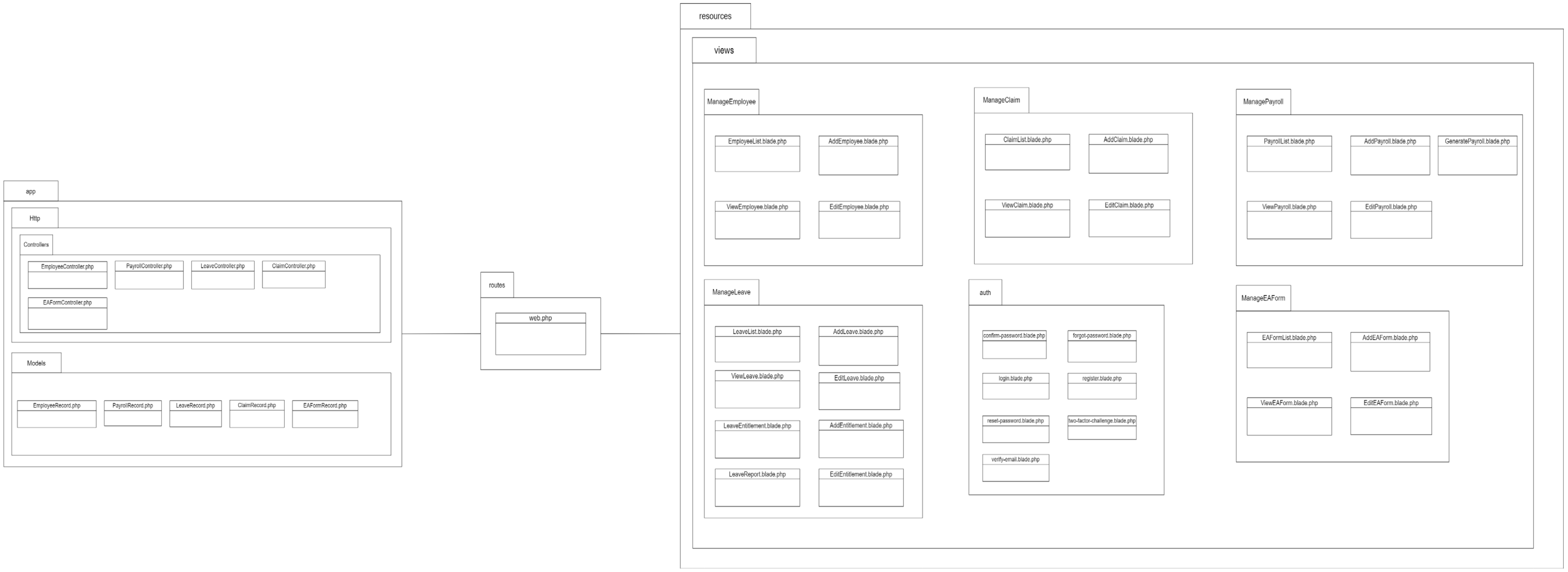
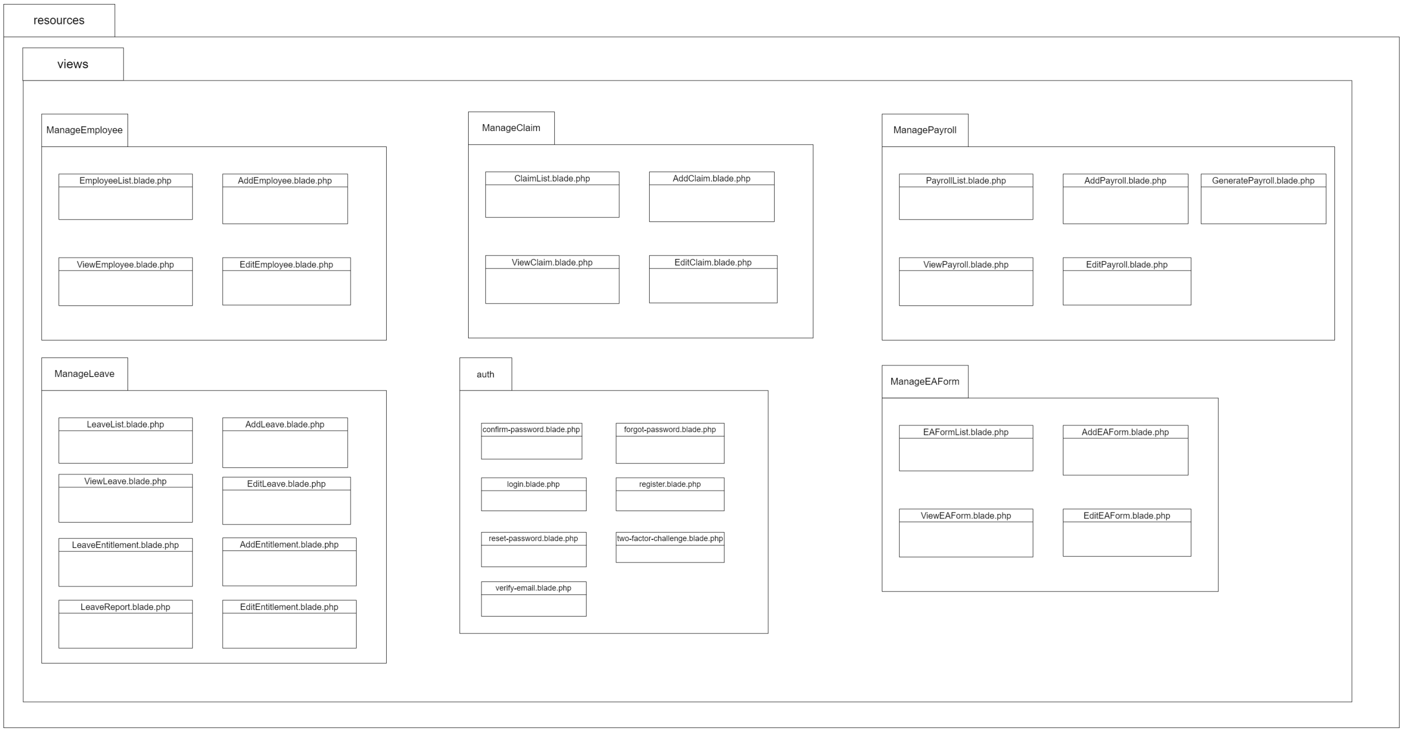
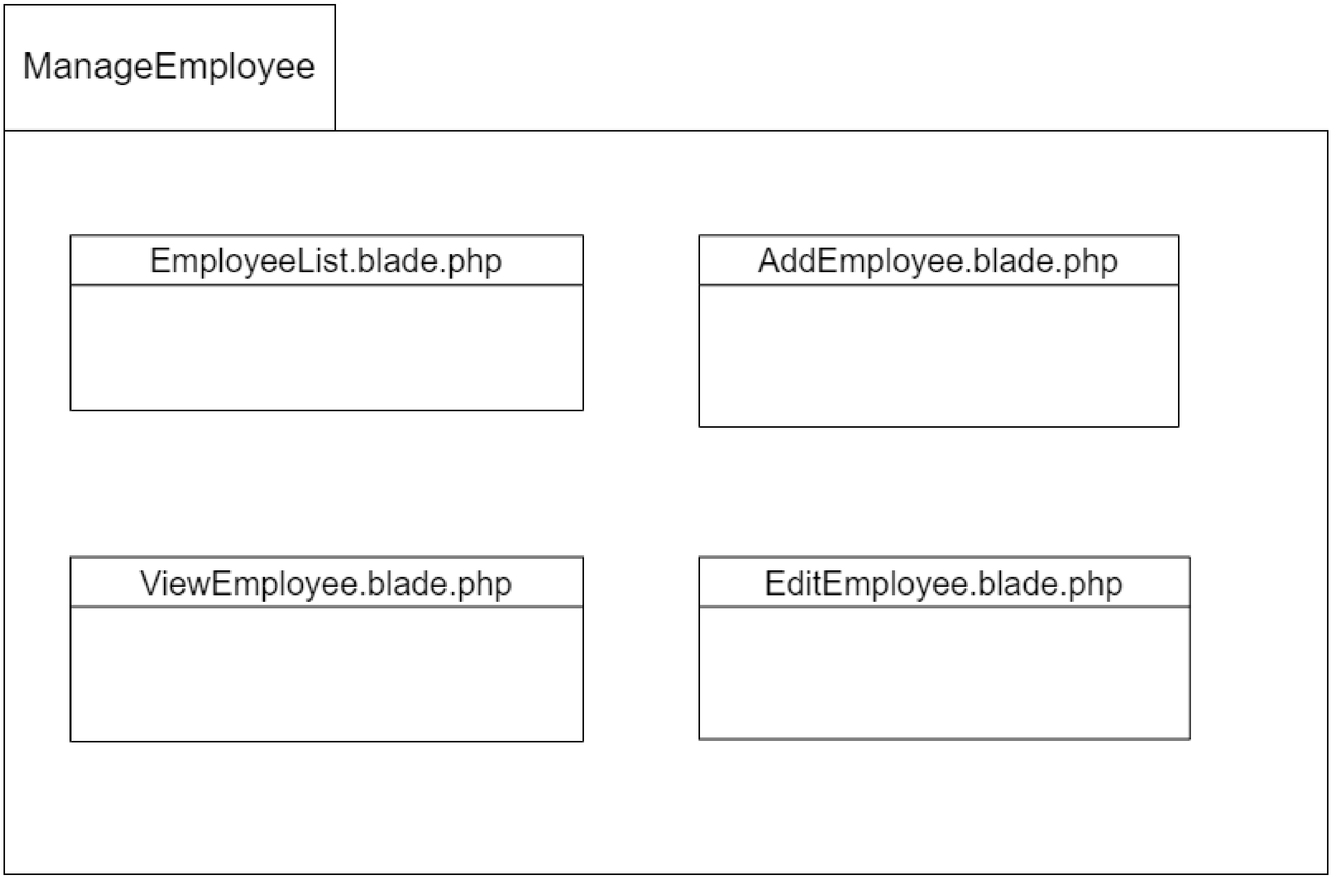


Figure 2: SME-HR System General Package

3.1.1 View Package

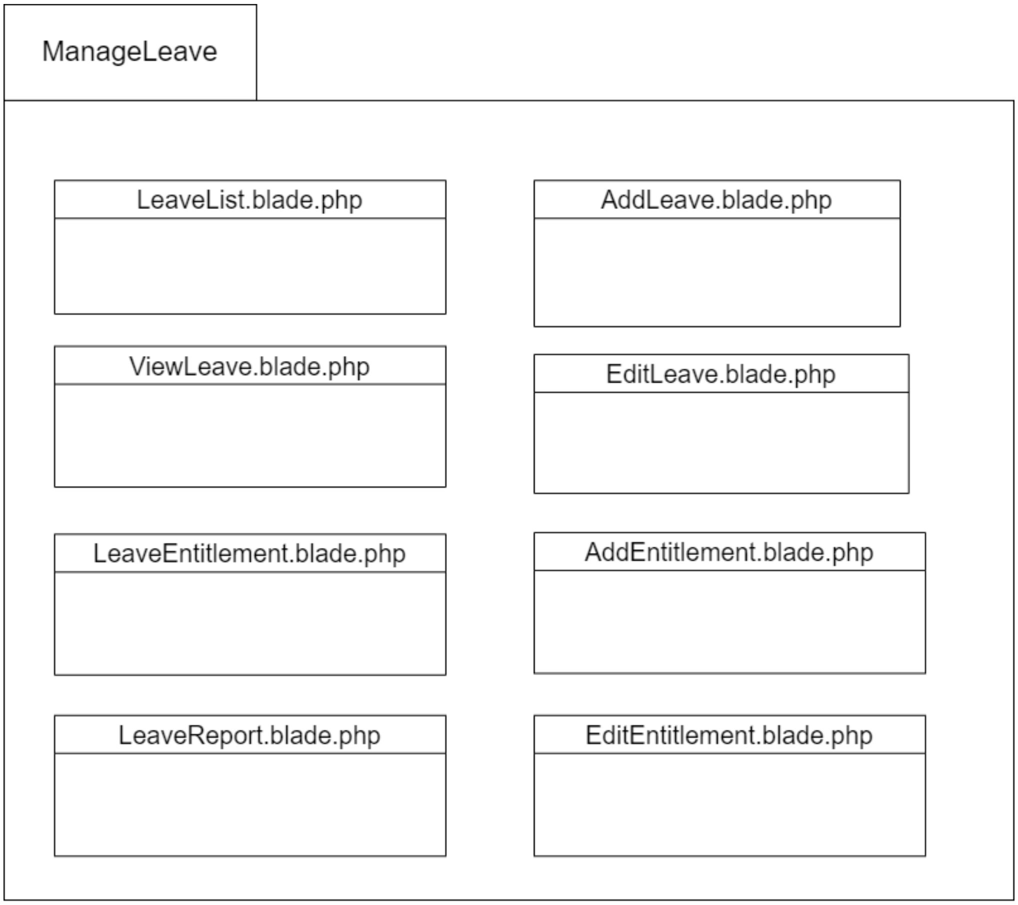


3.1.1.1 Manage Employee [SDD-REQ-100]



Class Name	Description
EmployeeList.blade.php [SDD-REQ-101]	This interface will allow the user to display all the list of employees.
AddEmployee.blade.php [SDD-REQ-102]	This interface allows the user to add new employee information.
ViewEmployee.blade.php [SDD-REQ-103]	This interface allows the user to view the detail of employee.
EditEmployee.blade.php [SDD-REQ-104]	This interface allows the user to edit the employee details.

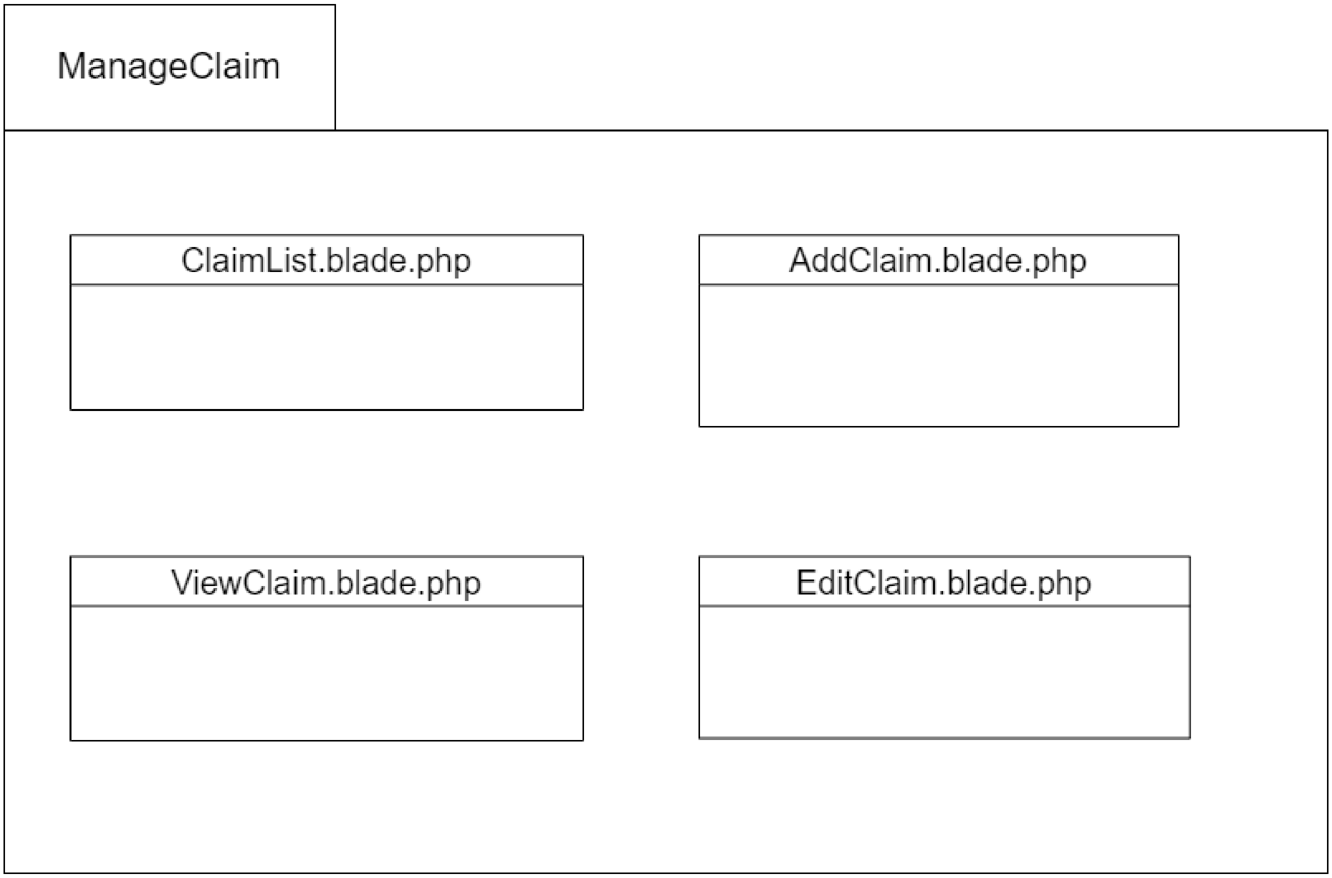
3.1.1.2 Manage Leave [SDD-REQ-200]



Class Name	Description
LeaveList.blade.php [SDD-REQ-201]	This interface allows the user to display all the list of leave.
AddLeave.blade.php [SDD-REQ-202]	This interface allows the user to add leave information.

Viewleave.blade.php [SDD-REQ-203]	This interface allows the user to display the leave information.
EditLeave.blade.php [SDD-REQ-204]	This interface allows the user to edit the leave information.
LeaveEntitlement.blade.php [SDD-REQ-205]	This interface allows the user to display all the list of leave entitlement.
AddEntitlement.blade.php [SDD-REQ-206]	This interface allows the user to add the leave entitlement information.
EditEntitlement.blade.php [SDD-REQ-207]	This interface allows the user to edit the leave entitlement information.
LeaveReport.blade.php [SDD-REQ-208]	This interface allows the user to display all the list of leave report.

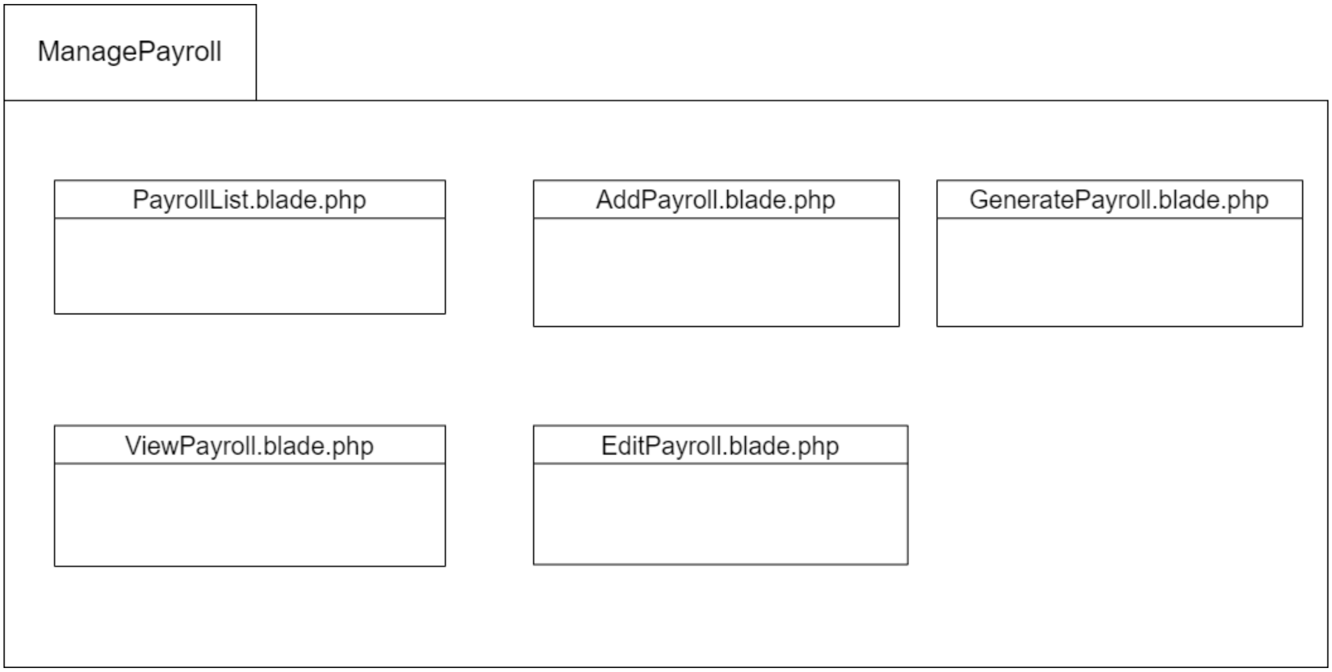
3.1.1.3 Manage Claim [SDD-REQ-300]



Class Name	Description
ClaimList.blade.php [SDD-REQ-301]	This interface allows the user to display all the list of claims.

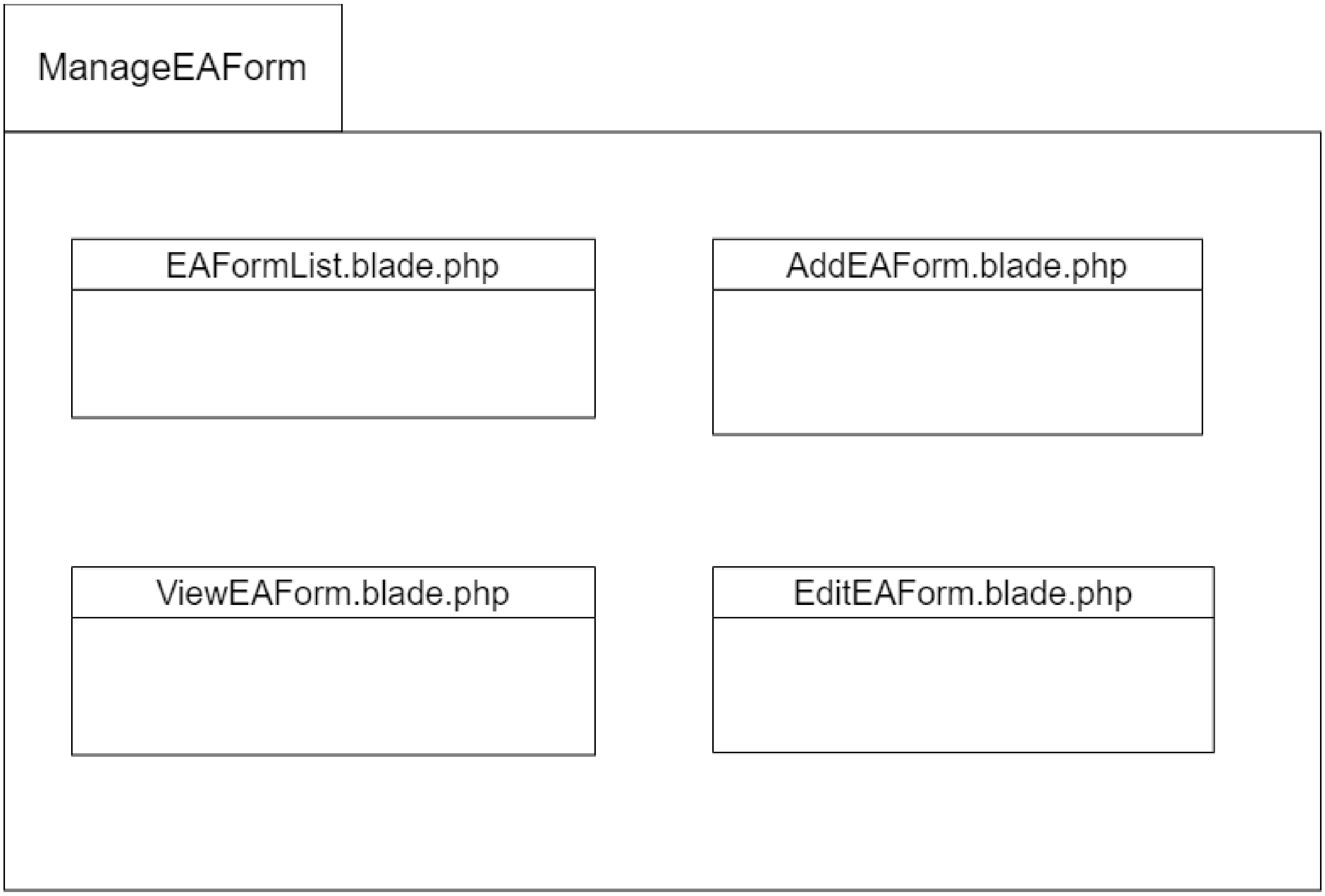
AddClaim.blade.php [SDD-REQ-302]	This interface allows the user to add claim information.
ViewClaim.blade.php [SDD-REQ-303]	This interface allows the user to display the claim information.
EditClaim.blade.php [SDD-REQ-304]	This interface allows the user to edit the claim information.

3.1.1.4 Manage Payroll [SDD-REQ-400]



Class Name	Description
PayrollList.blade.php [SDD-REQ-401]	This interface allows the user to display all the list of payrolls.
AddPayroll.blade.php [SDD-REQ-402]	This interface allows the user to add payroll information.
ViewPayroll.blade.php [SDD-REQ-403]	This interface allows the user to display the payroll information.
EditPayroll.blade.php [SDD-REQ-404]	This interface allows the user to edit the payroll information.
GeneratePayroll.blade.php [SDD-REQ-405]	This interface allows the user to generate the payroll information.

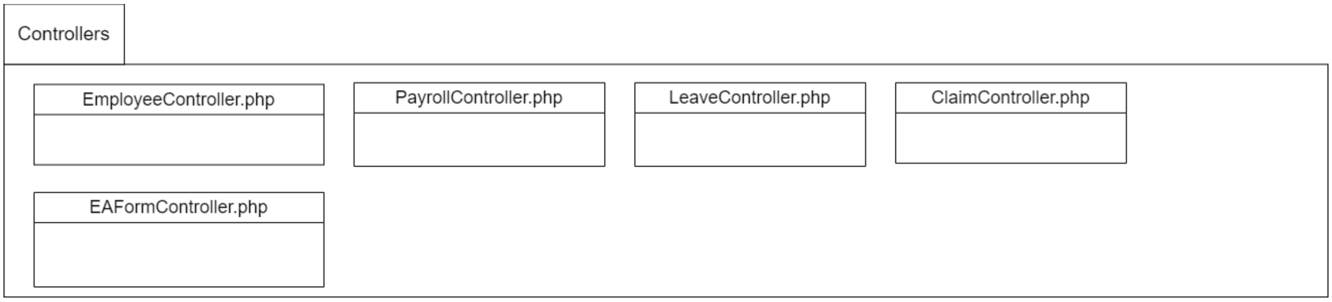
3.1.1.5 Manage EA Form [SDD-REQ-500]



Class Name	Description
EAFormList.blade.php [SDD-REQ-501]	This interface allows the user to display all the list of EA form.
AddEAForm.blade.php [SDD-REQ-502]	This interface allows the user to add EA form information.
ViewEAForm.blade.php [SDD-REQ-503]	This interface allows the user to display the EA form information.
EditEAForm.blade.php [SDD-REQ-504]	This interface allows the user to edit the EA form information.

3.1.2 Controller Package

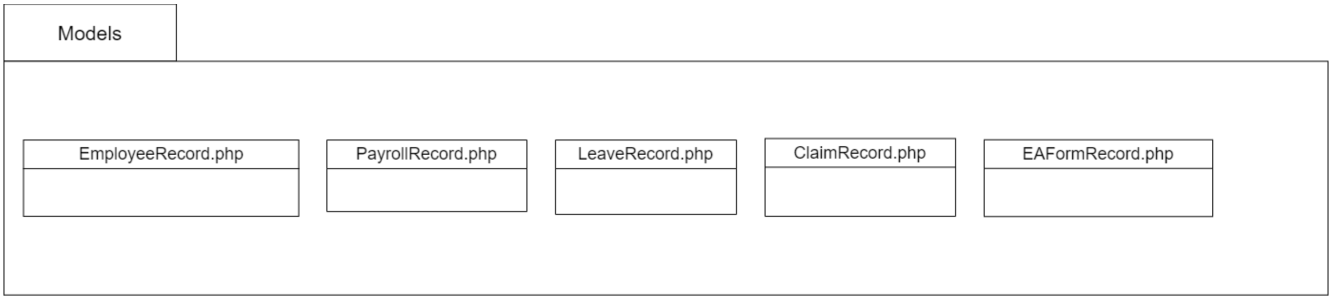
3.1.2.1 Controller [SDD-REQ-CT-100]



Class Name	Description
EmployeeController.php [SDD-REQ-CT-101]	This class allows the system to retrieve or validate data from the database and handle user request in employee page.
PayrollController.php [SDD-REQ-CT-102]	This class allows the system to retrieve or validate data from the database and handle user request in payroll page.
LeaveController.php [SDD-REQ-CT-103]	This class allows the system to retrieve or validate data from the database and handle user request in leave page.
ClaimController.php [SDD-REQ-CT-104]	This class allows the system to retrieve or validate data from the database and handle user request in claim page.
EAFormController.php [SDD-REQ-CT-105]	This class allows the system to retrieve or validate data from the database and handle user request in EA form page.

3.1.3 Model Package

3.1.3.1 Model [SDD-REQ-MD-100]

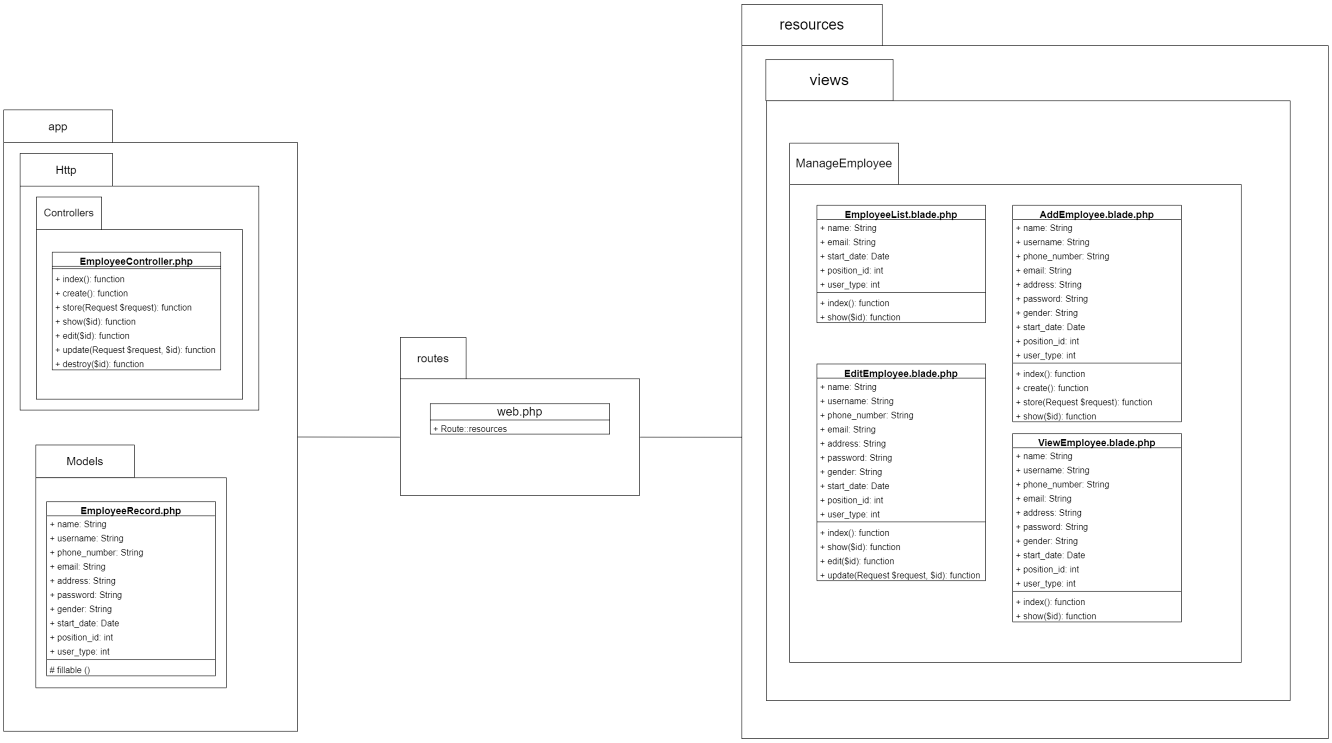


Class Name	Description
EmployeeRecord.php [SDD-REQ-MD-101]	This model is used to store data that is related to employee.
PayrollRecord.php	This model is used to store data that is related to payroll.

[SDD-REQ-MD-102]	
LeaveRecord.php [SDD-REQ-MD-103]	This model is used to store data that is related to leave.
ClaimRecord.php [SDD-REQ-MD-104]	This model is used to store data that is related to claim.
EAFormRecord.php [SDD-REQ-MD-105]	This model is used to store data that is related to EA form.

4. DETAIL DESIGN

4.1 Manage Employee [SDD-REQ-100]



4.1.1 EmployeeList.blade.php [SDD-REQ-101]

Class Type	Boundary class	
Responsibility	An interface to display the all of the employee list.	
Attributes	Attributes Name	Attributes Type
	name email start_date position_id user_type	String String Date Int Int
Methods	Method Name	Description
	index ( ) show ($id)	To go to the employee list page. To display specific data from the database.
Algorithm	**index ( )** **START** index () is called Return view to employeelist.blade.php	

```
    END
show ($id)
    START
        show () is called
        Retrieve specific data from database
        Return view
    END
```

4.1.2 AddEmployee.blade.php [SDD-REQ-102]

Class Type	Boundary class	
Responsibility	An interface allows the user to add new employee information.	
Attributes	Attributes Name	Attributes Type
	name	String
	username	String
	phone_number	String
	email	String
	address	String
	password	String
	gender	String
	start_date	Date
	position_id	Int
	user_type	Int
Methods	Method Name	Description
	create ( )	To go to the addEmployee.blade.php page
	store (Request $request)	To store the data from the user input in database
	show ($id)	To display specific data from the database

Algorithm

```
create ( )
    START
        create () is called
        Return view to AddEmployee.blade.php
    END

store (Request $request)
    START
        store (Request $request) is called
        insert attributes
        connect to database
        Return view
    END

show ($id)
    START
        show () is called
        Retrieve specific data from database
        Return view
    END
```

4.1.3 ViewEmployee.blade.php [SDD-REQ-103]

Class Type	Boundary class	
Responsibility	An interface allows the user to view the employee information.	
Attributes	Attributes Name	Attributes Type
	name	String
	username	String
	phone_number	String
	email	String
	address	String
	password	String
	gender	String
	start_date	Date
	position_id	Int

	user_type	Int
Methods	Method Name	Description
	index ( ) show ($id)	To go to the ViewEmployee.blade.php page To display specific data from the database
Algorithm	**index ( )** **START** index () is called Return view to ViewEmployee.blade.php **END** **show ($id)** **START** show () is called Retrieve specific data from database Return view **END**	

4.1.4 EditEmployee.blade.php [SDD-REQ-104]

Class Type	Boundary class	
Responsibility	An interface allows the user to edit the employee information.	
Attributes	Attributes Name	Attributes Type
	name username phone_number email address password gender start_date position_id user_type	String String String String String String String Date Int Int
Methods	Method Name	Description
	edit ($id)	To go to the EditEmployee.blade.php page

	update (Request $request, $id)	To update data from the database
Algorithm	**edit ($id)** **START** edit ($id) is called Return view to EditEmployee.blade.php **END** **update (Request $request, $id)** **START** update (Request $request, $id) is called Retrieve specific data from database Insert new information Sent data to the database Return view **END**	

4.1.5 EmployeeController.php [SDD-REQ-CT-101]

Class Type	Controller class	
Responsibility	This class allows the system to retrieve or validate data from the database and handle user request in employee page.	
Attributes	Attributes Name	Attributes Type
	Not Applicable	Not Applicable
Methods	Method Name	Description
	index ( ) create ( ) show ( ) store(Request $request) edit ($id) update(Request $request, $id) destroy ($id)	To display all the data from the database To accepts an array of attributes, creates a model, and inserts it into the database To display specific data from the database To display a form to apply changes To store a newly created resources in the database To update the data from the database To delete the data from the database
Algorithm	**index()**	

```
    START
        index () is called
        retrieve all data from database
        return view
    END

create ()
    START
        create () is called
        Return view
    END

show ()
    START
        show () is called
        retrieve specific data from the database
        Return view
    END

store (Request $request)
    START
        store () is called
        insert attributes
        connect to database
        Return view
    END

edit ($id)
    START
        edit ($id) is called
        change the new information in the update page
        Return view
    END
```

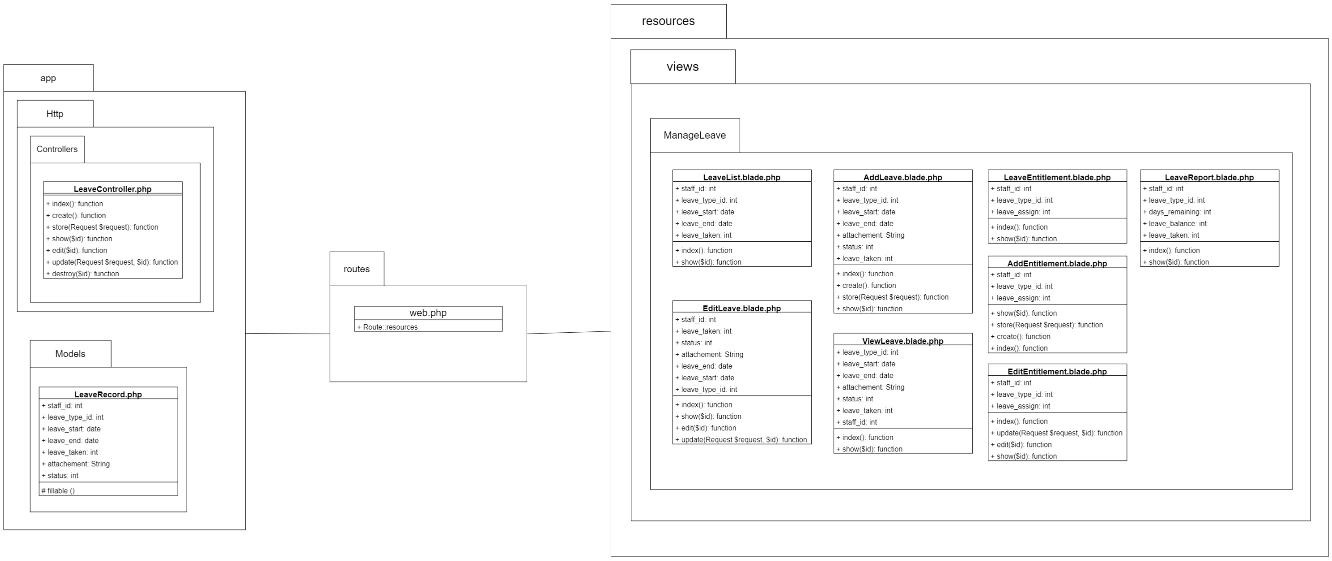
```
update (Request $request, $id)
    START
        update (Request $request, $id) is called
        retrieve data from database
        insert new information
        connect to database
        Return view
    END

destroy ($id)
    START
        destroy ($id) is called
        delete data from database
        Return view
    END
```

4.1.5 EmployeeRecord.php [SDD-REQ-MD-101]

Class Type	Model class	
Responsibility	This model is used to store the data for the employee into the database.	
Attributes	Attributes Name	Attributes Type
	name username phone_number email address password gender start_date position_id user_type	String String String String String String String Date Int Int
Methods	Method Name	Description
	fillable ( )	To take care of defining which fields are to be considered when the user insert or update data

Algorithm	**fillable ()** **START** fillable () is called insert attributes return this **END**

4.2 Manage Leave [SDD-REQ-200]



4.2.1 LeaveList.blade.php [SDD-REQ-201]

Class Type	Boundary class	
Responsibility	An interface to display all of the leave list.	
Attributes	Attributes Name	Attributes Type
	staff_id leave_type_id leave_start leave_end leave_taken	Int Int Date Date Int
Methods	Method Name	Description
	index ( ) show ($id)	To go to the leave list page. To display specific data from the database.
Algorithm	**index ( )** **START**	

```
                index () is called
                Return view to LeaveList.blade.php
        END
show ($id)
        START
                show () is called
                Retrieve specific data from database
                Return view
        END
```

4.2.2 AddLeave.blade.php [SDD-REQ-202]

Class Type	Boundary class	
Responsibility	An interface allows the user to add new employee information.	
Attributes	Attributes Name	Attributes Type
	staff_id	Int
	leave_type_id	Int
	leave_start	Date
	leave_end	Date
	attachment	String
	status	Int
	leave_taken	Int
Methods	Method Name	Description
	create ( )	To go to the AddLeave.blade.php page
	store (Request $request)	To store the data from the user input in database
	show ($id)	To display specific data from the database

Algorithm	**create ( )** **START** create () is called Return view to AddLeave.blade.php **END** **store (Request $request)** **START** store (Request $request) is called insert attributes connect to database Return view **END** **show ($id)** **START** show () is called Retrieve specific data from database Return view **END**

4.2.3 ViewLeave.blade.php [SDD-REQ-203]

Class Type	Boundary class	
Responsibility	An interface allows the user to view the leave information.	
Attributes	Attributes Name	Attributes Type
	staff_id	Int
	leave_type_id	Int
	leave_start	Date
	leave_end	Date
	attachment	String
	status	Int
	leave_taken	Int
Methods	Method Name	Description
	index ( )	To go to the ViewLeave.blade.php page

	show ($id)	To display specific data from the database
Algorithm	**index ( )** **START** index () is called Return view to ViewLeave.blade.php **END** **show ($id)** **START** show () is called Retrieve specific data from database Return view **END**	

4.2.4 EditLeave.blade.php [SDD-REQ-204]

Class Type	Boundary class	
Responsibility	An interface allows the user to edit the leave information.	
Attributes	Attributes Name	Attributes Type
	staff_id	Int
	leave_type_id	Int
	leave_start	Date
	leave_end	Date
	attachment	String
	status	Int
	leave_taken	Int
Methods	Method Name	Description
	edit ($id)	To go to the EditLeave.blade.php page
	update (Request $request, $id)	To update data from the database
Algorithm	**edit ($id)** **START** edit ($id) is called Return view to EditLeave.blade.php **END**	

	update (Request $request, $id) **START** update (Request $request, $id) is called Retrieve specific data from database Insert new information Sent data to the database Return view **END**

4.2.5 LeaveEntitlement.blade.php [SDD-REQ-205]

Class Type	Boundary class	
Responsibility	An interface to display all of the leave entitlement list.	
Attributes	Attributes Name	Attributes Type
	staff_id leave_type_id leave_assign	Int Int Int
Methods	Method Name	Description
	index ( ) show ($id)	To go to the leave list page. To display specific data from the database.
Algorithm	**index ( )** **START** index () is called Return view to LeaveEntitlement.blade.php **END** **show ($id)** **START** show () is called Retrieve specific data from database Return view **END**	

4.2.6 AddEntitlement.blade.php [SDD-REQ-206]

Class Type	Boundary class	
Responsibility	An interface allows the user to add new employee information.	
Attributes	Attributes Name	Attributes Type
	staff_id	Int
	leave_type_id	Int
	leave_assign	Int
Methods	Method Name	Description
	create ( )	To go to the AddEntitlement.blade.php page
	store (Request $request)	To store the data from the user input in database
	show ($id)	To display specific data from the database
Algorithm	**create ( )** **START** create () is called Return view to AddEntitlement.blade.php **END** **store (Request $request)** **START** store (Request $request) is called insert attributes connect to database Return view **END** **show ($id)** **START** show () is called Retrieve specific data from database Return view **END**	

4.2.7 EditEntitlement.blade.php [SDD-REQ-207]

Class Type	Boundary class	
Responsibility	An interface allows the user to edit the leave information.	
Attributes	Attributes Name	Attributes Type
	staff_id leave_type_id leave_assign	Int Int Int
Methods	Method Name	Description
	edit ($id) update (Request $request, $id)	To go to the EditEntitlement.blade.php page To update data from the database
Algorithm	**edit ($id)** **START** edit ($id) is called Return view to EditEntitlement.blade.php **END** **update (Request $request, $id)** **START** update (Request $request, $id) is called Retrieve specific data from database Insert new information Sent data to the database Return view **END**	

4.2.8 LeaveReport.blade.php [SDD-REQ-208]

Class Type	Boundary class	
Responsibility	An interface to display all of the leave report list.	
Attributes	Attributes Name	Attributes Type
	staff_id leave_type_id days_remaining leave_balance	Int Int Int Int

	leave_taken	Int
Methods	Method Name	Description
	index ( ) show ($id)	To go to the leave report page. To display specific data from the database.
Algorithm	**index ( )** **START** index () is called Return view to LeaveReport.blade.php **END** **show ($id)** **START** show () is called Retrieve specific data from database Return view **END**	

4.2.9 LeaveController.php [SDD-REQ-CT-103]

Class Type	Controller class	
Responsibility	This class allows the system to retrieve or validate data from the database and handle user request in leave page.	
Attributes	Attributes Name	Attributes Type
	Not Applicable	Not Applicable
Methods	Method Name	Description
	index ( ) create ( ) show ( ) store(Request $request) edit ($id) update(Request $request, $id) destroy ($id)	To display all the data from the database To accepts an array of attributes, creates a model, and inserts it into the database To display specific data from the database To display a form to apply changes To store a newly created resources in the database To update the data from the database To delete the data from the database

Algorithm

```
index()
    START
        index () is called
        retrieve all data from database
        return view
    END

create ()
    START
        create () is called
        Return view
    END

show ()
    START
        show () is called
        retrieve specific data from the database
        Return view
    END

store (Request $request)
    START
        store () is called
        insert attributes
        connect to database
        Return view
    END

edit ($id)
    START
        edit ($id) is called
        change the new information in the update page
        Return view
    END
```

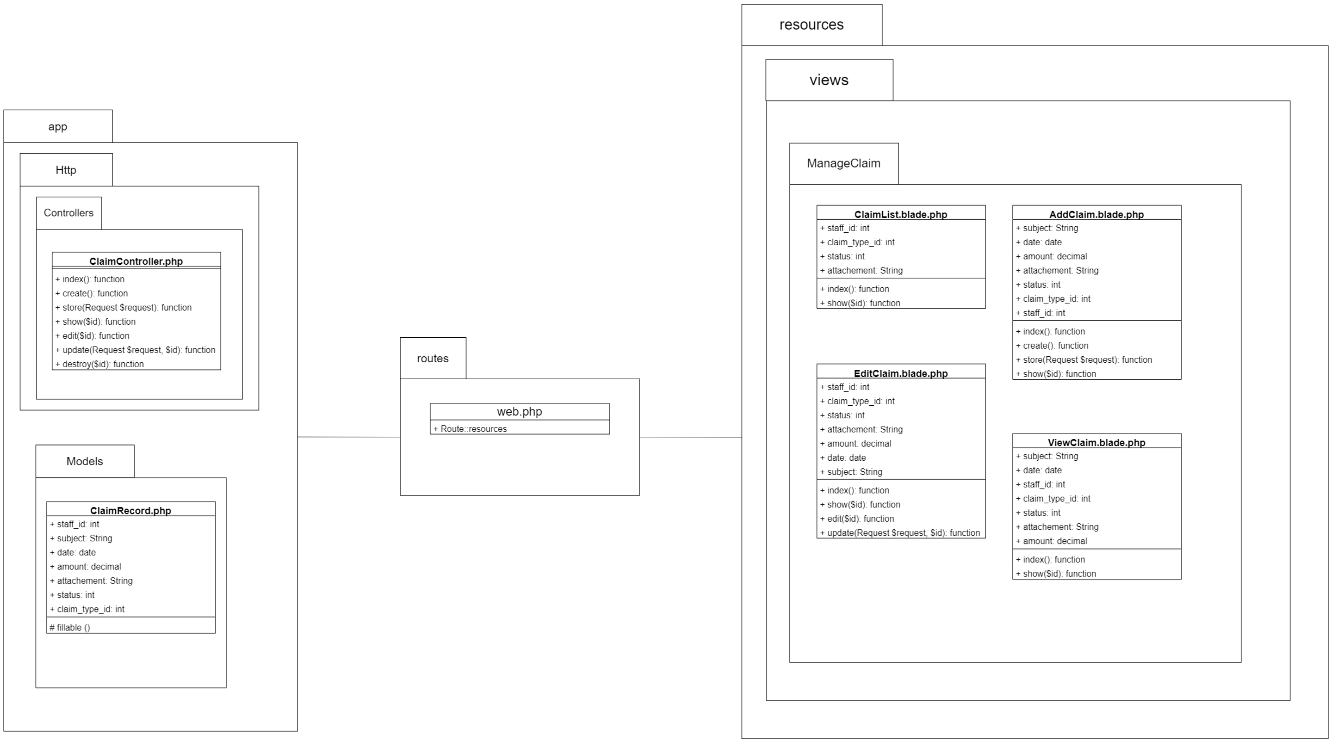
	update (Request $request, $id) **START** update (Request $request, $id) is called retrieve data from database insert new information connect to database Return view **END** **destroy ($id)** **START** destroy ($id) is called delete data from database Return view **END**

4.2.10 LeaveRecord.php [SDD-REQ-MD-103]

Class Type	Model class	
Responsibility	This model is used to store the data for the employee into the database.	
Attributes	Attributes Name	Attributes Type
	staff_id leave_type_id leave_start leave_end attachment status leave_taken	Int Int Date Date String Int Int
Methods	Method Name	Description
	fillable ( )	To take care of defining which fields are to be considered when the user insert or update data
Algorithm	**fillable ()** **START** fillable () is called	

	insert attributes
	return this
	END

4.3 Manage Claim [SDD-REQ-300]



4.3.1 ClaimList.blade.php [SDD-REQ-301]

Class Type	Boundary class	
Responsibility	An interface to display all of the claim list.	
Attributes	Attributes Name	Attributes Type
	staff_id claim_type_id status attachment	Int Int Int String
Methods	Method Name	Description
	index ( ) show ($id)	To go to the claim list page. To display specific data from the database.
Algorithm	**index ( )** **START** index () is called	

```
            Return view to ClaimList.blade.php
        END
show ($id)
        START
            show () is called
            Retrieve specific data from database
            Return view
        END
```

4.3.2 AddClaim.blade.php [SDD-REQ-302]

Class Type	Boundary class	
Responsibility	An interface allows the user to add new employee information.	
Attributes	Attributes Name	Attributes Type
	subject	String
	date	Date
	amount	Decimal
	staff_id	Int
	claim_type_id	Int
	status	Int
	attachment	String
Methods	Method Name	Description
	create ( )	To go to the addClaim.blade.php page
	store (Request $request)	To store the data from the user input in database
	show ($id)	To display specific data from the database

Algorithm	**create ( )** **START** create () is called Return view to AddClaim.blade.php **END** **store (Request $request)** **START** store (Request $request) is called insert attributes connect to database Return view **END** **show ($id)** **START** show () is called Retrieve specific data from database Return view **END**

4.3.3 ViewClaim.blade.php [SDD-REQ-303]

Class Type	Boundary class	
Responsibility	An interface allows the user to view the employee information.	
Attributes	Attributes Name	Attributes Type
	subject date amount staff_id claim_type_id status attachment	String Date Decimal Int Int Int String
Methods	Method Name	Description
	index ( )	To go to the ViewClaim.blade.php page

	show ($id)	To display specific data from the database

Algorithm	**index ( )**
	    START
	        index () is called
	        Return view to ViewClaim.blade.php
	    END
	
	show ($id)
	    START
	        show () is called
	        Retrieve specific data from database
	        Return view
	    END	

4.3.4 EditClaim.blade.php [SDD-REQ-304]

Class Type	Boundary class	
Responsibility	An interface allows the user to edit the employee information.	
Attributes	Attributes Name	Attributes Type
	subject date amount staff_id claim_type_id status attachment	String Date Decimal Int Int Int String
Methods	Method Name	Description
	edit ($id) update (Request $request, $id)	To go to the EditClaim.blade.php page To update data from the database

Algorithm	**edit ($id)**
	    START
	        edit ($id) is called
	        Return view to EditClaim.blade.php
	    END	

	update (Request $request, $id) **START** update (Request $request, $id) is called Retrieve specific data from database Insert new information Sent data to the database Return view **END**

4.3.5 ClaimController.php [SDD-REQ-CT-104]

Class Type	Controller class	
Responsibility	This class allows the system to retrieve or validate data from the database and handle user request in claim page.	
Attributes	Attributes Name	Attributes Type
	Not Applicable	Not Applicable
Methods	Method Name	Description
	index ( ) create ( ) show ( ) store(Request $request) edit ($id) update(Request $request, $id) destroy ($id)	To display all the data from the database To accepts an array of attributes, creates a model, and inserts it into the database To display specific data from the database To display a form to apply changes To store a newly created resources in the database To update the data from the database To delete the data from the database
Algorithm	**index()** **START** index () is called retrieve all data from database return view **END** **create ()**	

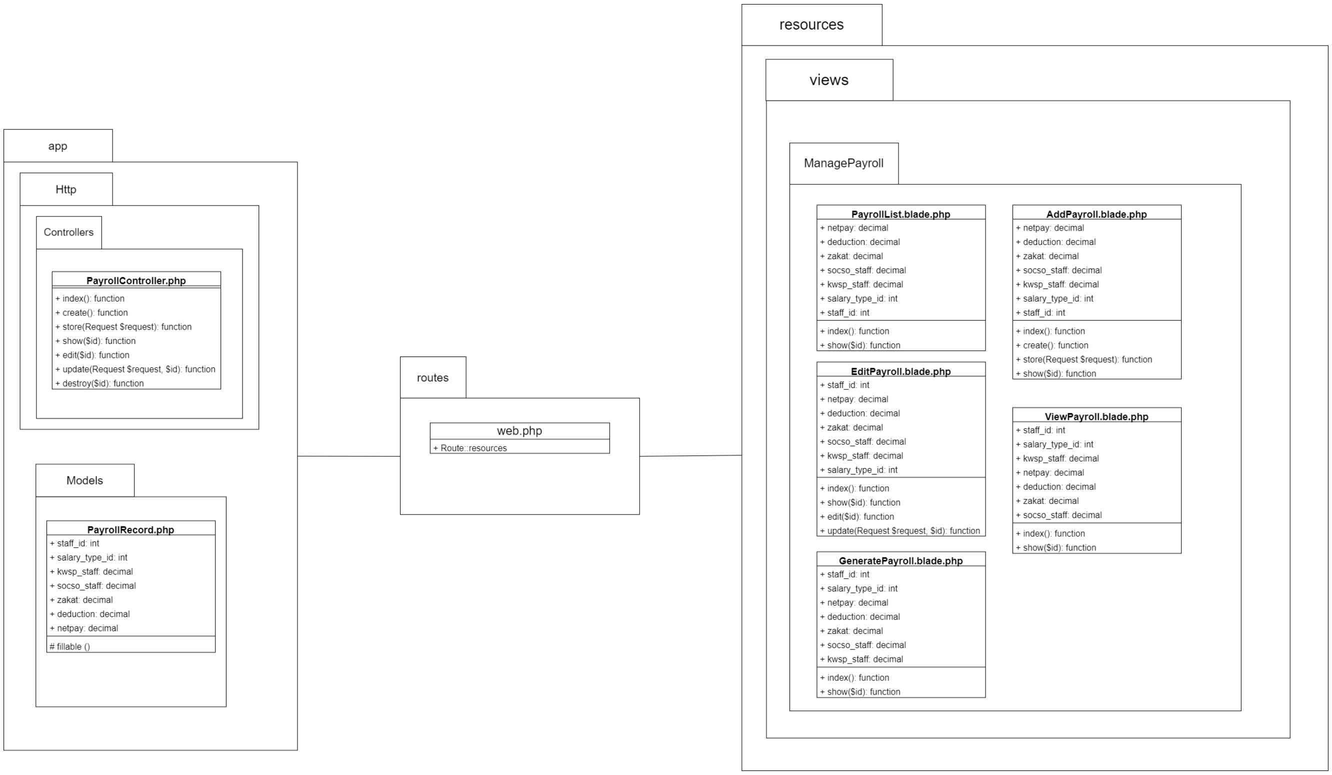
```
        START
                create () is called
                Return view
        END

show ()
        START
                show () is called
                retrieve specific data from the database
                Return view
        END

store (Request $request)
        START
                store () is called
                insert attributes
                connect to database
                Return view
        END

edit ($id)
        START
                edit ($id) is called
                change the new information in the update page
                Return view
        END



update (Request $request, $id)
        START
                update (Request $request, $id) is called
                retrieve data from database
                insert new information
                connect to database
                Return view
```

	END **destroy ($id)** **START** destroy ($id) is called delete data from database Return view **END**

4.3.4 ClaimRecord.php [SDD-REQ-MD-104]

Class Type	Model class	
Responsibility	This model is used to store the data for the employee into the database.	
Attributes	Attributes Name	Attributes Type
	subject date amount staff_id claim_type_id status attachment	String Date Decimal Int Int Int String
Methods	Method Name	Description
	fillable ( )	To take care of defining which fields are to be considered when the user inserts or update data
Algorithm	**fillable ()** **START** fillable () is called insert attributes return this **END**	

4.4 Manage Payroll [SDD-REQ-400]



4.4.1 PayrollList.blade.php [SDD-REQ-401]

Class Type	Boundary class	
Responsibility	An interface to display all of the payroll list.	
Attributes	Attributes Name	Attributes Type
	netpay deduction zakat socso_staff kwsp_staff salary_type_id staff_id	Decimal Decimal Decimal Decimal Decimal Int Int
Methods	Method Name	Description
	index ( ) show ($id)	To go to the payroll list page. To display specific data from the database.
Algorithm	**index ( )** **START** index () is called Return view to PayrollList.blade.php	

```
        END
show ($id)
        START
                show () is called
                Retrieve specific data from database
                Return view
        END
```

4.4.2 AddPayroll.blade.php [SDD-REQ-402]

Class Type	Boundary class	
Responsibility	An interface allows the user to add new payroll information.	
Attributes	Attributes Name	Attributes Type
	netpay	Decimal
	deduction	Decimal
	zakat	Decimal
	socso_staff	Decimal
	kwsp_staff	Decimal
	salary_type_id	Int
	staff_id	Int
Methods	Method Name	Description
	create ( )	To go to the AddPayroll.blade.php page
	store (Request $request)	To store the data from the user input in database
	show ($id)	To display specific data from the database

Algorithm	**create ( )** **START** create () is called Return view to AddPayroll.blade.php **END** **store (Request $request)** **START** store (Request $request) is called insert attributes connect to database Return view **END** **show ($id)** **START** show () is called Retrieve specific data from database Return view **END**

4.4.3 ViewPayroll.blade.php [SDD-REQ-403]

Class Type	Boundary class	
Responsibility	An interface allows the user to view the payroll information.	
Attributes	Attributes Name	Attributes Type
	netpay	Decimal
	deduction	Decimal
	zakat	Decimal
	socso_staff	Decimal
	kwsp_staff	Decimal
	salary_type_id	Int
	staff_id	Int
Methods	Method Name	Description
	index ( )	To go to the ViewPayroll.blade.php page

	show ($id)	To display specific data from the database
Algorithm	**index ( )** **START** index () is called Return view to ViewPayroll.blade.php **END** **show ($id)** **START** show () is called Retrieve specific data from database Return view **END**	

4.4.4 EditPayroll.blade.php [SDD-REQ-404]

Class Type	Boundary class	
Responsibility	An interface allows the user to edit the employee information.	
Attributes	Attributes Name	Attributes Type
	netpay	Decimal
	deduction	Decimal
	zakat	Decimal
	socso_staff	Decimal
	kwsp_staff	Decimal
	salary_type_id	Int
	staff_id	Int
Methods	Method Name	Description
	edit ($id)	To go to the EditPayroll.blade.php page
	update (Request $request, $id)	To update data from the database
Algorithm	**edit ($id)** **START** edit ($id) is called Return view to EditPayroll.blade.php **END**	

	update (Request $request, $id) **START** update (Request $request, $id) is called Retrieve specific data from database Insert new information Sent data to the database Return view **END**

4.4.5 GeneratePayroll.blade.php [SDD-REQ-405]

Class Type	Boundary class	
Responsibility	An interface to generate payroll.	
Attributes	Attributes Name	Attributes Type
	netpay	Decimal
	deduction	Decimal
	zakat	Decimal
	socso_staff	Decimal
	kwsp_staff	Decimal
	salary_type_id	Int
	staff_id	Int
Methods	Method Name	Description
	index ( )	To go to the generate payroll page.
	show ($id)	To display specific data from the database.
Algorithm	**index ( )** **START** index () is called Return view to GeneratePayroll.blade.php **END** **show ($id)** **START** show () is called Retrieve specific data from database Return view	

	END

4.4.6 PayrollController.php [SDD-REQ-CT-102]

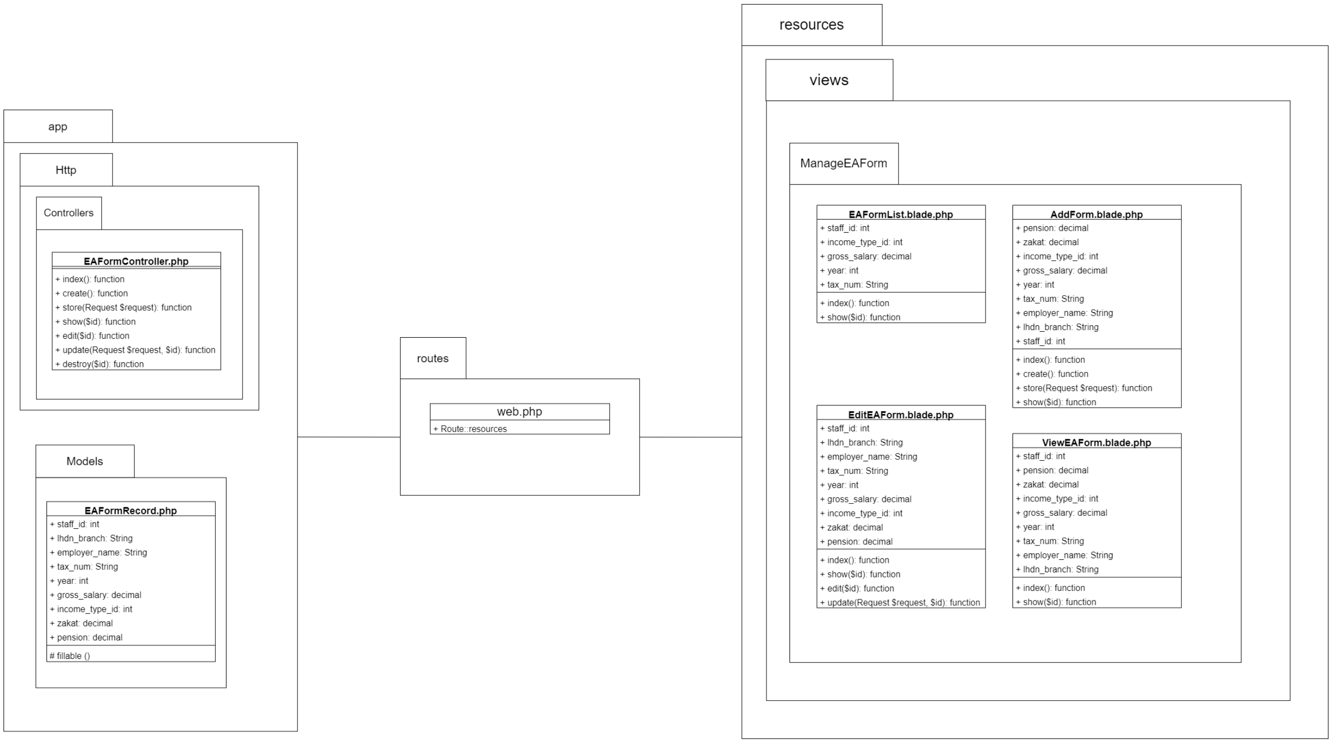
Class Type	Controller class	
Responsibility	This class allows the system to retrieve or validate data from the database and handle user request in payroll page.	
Attributes	Attributes Name	Attributes Type
	Not Applicable	Not Applicable
Methods	Method Name	Description
	index ( ) create ( ) show ( ) store(Request $request) edit ($id) update(Request $request, $id) destroy ($id)	To display all the data from the database To accepts an array of attributes, creates a model, and inserts it into the database To display specific data from the database To display a form to apply changes To store a newly created resources in the database To update the data from the database To delete the data from the database
Algorithm	**index()** **START** index () is called retrieve all data from database return view **END** **create ()** **START** create () is called Return view **END** **show ()** **START** show () is called	

```
                retrieve specific data from the database
                Return view
        END

store (Request $request)
        START
                store () is called
                insert attributes
                connect to database
                Return view
        END

edit ($id)
        START
                edit ($id) is called
                change the new information in the update page
                Return view
        END

update (Request $request, $id)
        START
                update (Request $request, $id) is called
                retrieve data from database
                insert new information
                connect to database
                Return view
        END

destroy ($id)
        START
                destroy ($id) is called
                delete data from database
                Return view
        END
```

4.4.7 ClaimRecord.php [SDD-REQ-MD-102]

Class Type	Model class	
Responsibility	This model is used to store the data for the employee into the database.	
Attributes	Attributes Name	Attributes Type
	netpay	Decimal
	deduction	Decimal
	zakat	Decimal
	socso_staff	Decimal
	kwsp_staff	Decimal
	salary_type_id	Int
	staff_id	Int
Methods	Method Name	Description
	fillable ( )	To take care of defining which fields are to be considered when the user insert or update data
Algorithm	**fillable ()** **START** fillable () is called insert attributes return this **END**	

4.5 Manage EA Form [SDD-REQ-500]



4.5.1 EAFormList.blade.php [SDD-REQ-501]

Class Type	Boundary class	
Responsibility	An interface to display all of the EA Form list.	
Attributes	Attributes Name	Attributes Type
	staff_id	Int
	income_type_id	Int
	gross_salary	Decimal
	year	Int
	tax_num	String
Methods	Method Name	Description
	index ( )	To go to the EA form list page.
	show ($id)	To display specific data from the database.
Algorithm	**index ( )** **START** index () is called Return view to EAFormList.blade.php **END** **show ($id)**	

	START show () is called Retrieve specific data from database Return view **END**

4.5.2 AddEAForm.blade.php [SDD-REQ-502]

Class Type	Boundary class	
Responsibility	An interface allows the user to add new EA form information.	
Attributes	Attributes Name	Attributes Type
	pension	Decimal
	zakat	Decimal
	income_type	Int
	gross_salary	Decimal
	year	Int
	tax_num	String
	employer_name	String
	lhdn_branch	String
	staff_id	Int
Methods	Method Name	Description
	create ( )	To go to the addEAForm.blade.php page
	store (Request $request)	To store the data from the user input in database
	show ($id)	To display specific data from the database

Algorithm	**create ( )** **START** create () is called Return view to AddEAForm.blade.php **END** **store (Request $request)** **START** store (Request $request) is called insert attributes connect to database Return view **END** **show ($id)** **START** show () is called Retrieve specific data from database Return view **END**

4.5.3 ViewEAForm.blade.php [SDD-REQ-503]

Class Type	Boundary class	
Responsibility	An interface allows the user to view the EA form information.	
Attributes	Attributes Name	Attributes Type
	pension	Decimal
	zakat	Decimal
	income_type	Int
	gross_salary	Decimal
	year	Int
	tax_num	String
	employer_name	String
	lhdn_branch	String
	staff_id	Int
Methods	Method Name	Description
	index ( )	To go to the ViewEAForm.blade.php page

	show ($id)	To display specific data from the database
Algorithm	**index ( )** **START** index () is called Return view to ViewEAForm.blade.php **END** **show ($id)** **START** show () is called Retrieve specific data from database Return view **END**	

4.5.4 EditEAForm.blade.php [SDD-REQ-504]

Class Type	Boundary class	
Responsibility	An interface allows the user to edit the EA Form information.	
Attributes	Attributes Name	Attributes Type
	pension	Decimal
	zakat	Decimal
	income_type	Int
	gross_salary	Decimal
	year	Int
	tax_num	String
	employer_name	String
	lhdn_branch	String
	staff_id	Int
Methods	Method Name	Description
	edit ($id)	To go to the EditEAForm.blade.php page
	update (Request $request, $id)	To update data from the database
Algorithm	**edit ($id)** **START** edit ($id) is called	

	Return view to EditEAForm.blade.php **END** **update (Request $request, $id)** **START** update (Request $request, $id) is called Retrieve specific data from database Insert new information Sent data to the database Return view **END**

4.4.7 EAFormController.php [SDD-REQ-CT-105]

Class Type	Controller class	
Responsibility	This class allows the system to retrieve or validate data from the database and handle user request in EA form page.	
Attributes	Attributes Name	Attributes Type
	Not Applicable	Not Applicable
Methods	Method Name	Description
	index ( ) create ( ) show ( ) store(Request $request) edit ($id) update(Request $request, $id) destroy ($id)	To display all the data from the database To accepts an array of attributes, creates a model, and inserts it into the database To display specific data from the database To display a form to apply changes To store a newly created resources in the database To update the data from the database To delete the data from the database

Algorithm	**index()**
	START
	index () is called
	retrieve all data from database
	return view
	END	

```
create ()
    START
        create () is called
        Return view
    END

show ()
    START
        show () is called
        retrieve specific data from the database
        Return view
    END

store (Request $request)
    START
        store () is called
        insert attributes
        connect to database
        Return view
    END

edit ($id)
    START
        edit ($id) is called
        change the new information in the update page
        Return view
    END

update (Request $request, $id)
    START
        update (Request $request, $id) is called
        retrieve data from database
        insert new information
        connect to database
        Return view
    END

destroy ($id)
    START
        destroy ($id) is called
        delete data from database
        Return view
    END
```

4.4.6 EAFormRecord.php [SDD-REQ-MD-105]

Class Type	Model class	
Responsibility	This model is used to store the data for the EA form into the database.	
Attributes	Attributes Name	Attributes Type
	pension zakat income_type gross_salary year tax_num employer_name lhdn_branch staff_id	Decimal Decimal Int Decimal Int String String String Int
Methods	Method Name	Description
	fillable ( )	To take care of defining which fields are to be considered when the user insert or update data
Algorithm	**fillable ()** **START** fillable () is called insert attributes return this **END**	

5. REQUIREMENT TRACEABILITY

5.1 Login [SME-HR-REQ-001]

Requirement ID	Description	Design ID
SME-HR-REQ-001	Provides Administrator, Staff, Manager the ability to log in to the system	SDD-REQ-101 SDD-REQ-CT-101 SDD-REQ-MD-101
SME-HR-REQ-001-A01	Provides Administrator, Staff, Manager the ability to click on the forgot password	SDD-REQ-101 SDD-REQ-CT-101 SDD-REQ-MD-101

5.2 Manage Employee [SME-HR-REQ-002]

Requirement ID	Description	Design ID
SME-HR-REQ-002	Provides Administrator the ability to manage employee.	SDD-REQ-101 SDD-REQ-102 SDD-REQ-103 SDD-REQ-104 SDD-REQ-CT-101 SDD-REQ-MD-101
SME-HR-REQ-002-A01	Provides Administrator the ability to search specific employee.	SDD-REQ-101 SDD-REQ-CT-101 SDD-REQ-MD-101

5.3 View Employee [SME-HR-REQ-003]

Requirement ID	Description	Design ID
SME-HR-REQ-003	Provides Staff and Manager the ability to view employee.	SDD-REQ-101 SDD-REQ-103 SDD-REQ-CT-101 SDD-REQ-MD-101

5.4 Manage Profile [SME-HR-REQ-004]

Requirement ID	Description	Design ID
SME-HR-REQ-004	Provides Administrator, Staff, and Manager the ability to manage profile.	SDD-REQ-101 SDD-REQ-102 SDD-REQ-CT-101 SDD-REQ-MD-101

5.5 Manage Payroll [SME-HR-REQ-005]

Requirement ID	Description	Design ID
SME-HR-REQ-005	Provides Administrator the ability to manage payroll.	SDD-REQ-401 SDD-REQ-402 SDD-REQ-403 SDD-REQ-404 SDD-REQ-405 SDD-REQ-CT-102 SDD-REQ-MD-102
SME-HR-REQ-005-A01	Provides Administrator the ability to search specific payroll.	SDD-REQ-401 SDD-REQ-405 SDD-REQ-CT-102 SDD-REQ-MD-102

5.6 View Payroll [SME-HR-REQ-006]

Requirement ID	Description	Design ID
SME-HR-REQ-006	Provides Staff and Manager the ability to view payroll.	SDD-REQ-401 SDD-REQ-403 SDD-REQ-CT-102 SDD-REQ-MD-102
SME-HR-REQ-006-A01	Provides Administrator the ability to search specific pay slip.	SDD-REQ-401 SDD-REQ-CT-102 SDD-REQ-MD-102

SME-HR-REQ-006-A02	Provides Administrator the ability to filter specific pay slip.	SDD-REQ-401 SDD-REQ-CT-102 SDD-REQ-MD-102

5.7 Manage Leave [SME-HR-REQ-007]

Requirement ID	Description	Design ID
SME-HR-REQ-007	Provides Administrator, Staff and Manager the ability to manage leave.	SDD-REQ-201 SDD-REQ-202 SDD-REQ-203 SDD-REQ-204 SDD-REQ-205 SDD-REQ-206 SDD-REQ-207 SDD-REQ-208 SDD-REQ-CT-103 SDD-REQ-MD-103
SME-HR-REQ-007-A01	Provides Administrator the ability to search specific leave.	SDD-REQ-201 SDD-REQ-205 SDD-REQ-208 SDD-REQ-CT-103 SDD-REQ-MD-103
SME-HR-REQ-007-A02	Provides Staff and Manager the ability to search specific leave.	SDD-REQ-201 SDD-REQ-205 SDD-REQ-208 SDD-REQ-CT-103 SDD-REQ-MD-103

5.8 Manage Claim [SME-HR-REQ-008]

Requirement ID	Description	Design ID
SME-HR-REQ-008	Provides Administrator, Staff and Manager the ability to manage claim.	SDD-REQ-301 SDD-REQ-302 SDD-REQ-303 SDD-REQ-304 SDD-REQ-CT-104 SDD-REQ-MD-104
SME-HR-REQ-008-A01	Provides Administrator the ability to search specific claim.	SDD-REQ-301 SDD-REQ-CT-104 SDD-REQ-MD-104
SME-HR-REQ-008-A02	Provides Staff and Manager the ability to search specific claim.	SDD-REQ-301 SDD-REQ-CT-104 SDD-REQ-MD-104

5.9 Manage EA Form [SME-HR-REQ-009]

Requirement ID	Description	Design ID
SME-HR-REQ-009	Provides Administrator the ability to manage EA form.	SDD-REQ-501 SDD-REQ-502 SDD-REQ-503 SDD-REQ-504 SDD-REQ-CT-105 SDD-REQ-MD-105
SME-HR-REQ-009-A01	Provides Administrator the ability to search specific EA form.	SDD-REQ-501 SDD-REQ-CT-105 SDD-REQ-MD-105

5.10 View EA Form [SME-HR-REQ-010]

Requirement ID	Description	Design ID
SME-HR-REQ-010	Provides Staff and Manager the ability to view EA form.	SDD-REQ-501 SDD-REQ-503 SDD-REQ-CT-105 SDD-REQ-MD-105
SME-HR-REQ-010-A01	Provides Staff and Manager the ability to search specific EA form.	SDD-REQ-501 SDD-REQ-CT-105 SDD-REQ-MD-105
SME-HR-REQ-010-A02	Provides Staff and Manager the ability to filter specific EA form.	SDD-REQ-501 SDD-REQ-CT-105 SDD-REQ-MD-105