SPECIAL NEEDS CHILDREN ONBOARDING MANAGEMENT SYSTEM

NURIN IMAN BINTI MOHAMMAD AZMI

Bachelor of Degree

UNIVERSITI MALAYSIA PAHANG

UNIVERSITI MALAYSIA PAHANG

DECLARATION OF THESIS AND COPYRIGHT

Author's Full Name : NURIN IMAN BINTI MOHAMMAD AZMI

Date of Birth _____

Title : SPECIAL NEEDS CHILDREN ONBOARDING _____

MANAGEMENT SYSTEM _____

_____

Academic Session : SEMESTER II 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)

000720-10-0262
New IC/Passport Number
Date: 12/06/2023

_____ (Supervisor's Signature)

DR ZAFRIL RIZAL BIN M. AZMI
Name of Supervisor
Date: 04/07/2023

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

THESIS DECLARATION LETTER (OPTIONAL)

Librarian,
Perpustakaan Universiti Malaysia Pahang,
Universiti Malaysia Pahang,
Lebuhraya Tun Razak,
26300, Gambang, Kuantan.

Dear Sir,

CLASSIFICATION OF THESIS AS RESTRICTED

Please be informed that the following thesis is classified as RESTRICTED for a period of three (3) years from the date of this letter. The reasons for this classification are as listed below.

Author's Name
Thesis Title

Reasons (i)

(ii)

(iii)

Thank you.

Yours faithfully,

_____
(Supervisor's Signature)

Date:

Stamp:

Note: This letter should be written by the supervisor, addressed to the Librarian, *Perpustakaan Universiti Malaysia Pahang* with its copy attached to the thesis.

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/ Bachelor in Computer Science (Software Engineering) with Honors.

_____

(Supervisor's Signature)

Full Name : DR ZAFRIL RIZAL BIN M. AZMI

Position : SENIOR LECTURER

Date : 04/07/2023

_____

(Co-supervisor's Signature)

Full Name :

Position :

Date :

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 : NURIN IMAN BINTI MOHAMMAD AZMI

ID Number : CB20180

Date : 12/06/2023

SPECIAL NEEDS CHILDREN ONBOARDING MANAGEMENT SYSTEM

NURIN IMAN BINTI MOHAMMAD AZMI

Thesis submitted in fulfillment of the requirements
for the award of the degree of
Bachelor of Degree of Software Engineering

Faculty of Computing
UNIVERSITI MALAYSIA PAHANG

JUNE 2023

ACKNOWLEDGEMENTS

In the name of Allah, the Most Merciful and Gracious.

Allah is praised and blessed for the completion of this final-year project. I want to express my gratitude and appreciation to everyone who contributed to completing my final year project, "Special Needs Children Onboarding Management System." This endeavour would not have been possible without numerous individuals' and organisations' support, guidance, and assistance.

Before anything else, I would like to thank my project supervisor, Dr Zafril Rizal Bin M. Azmi, for his invaluable guidance, constant encouragement, and insightful suggestions throughout this project's development. Their knowledge, tolerance, and unwavering support were essential in shaping and improving the overall quality of this system.

Also, I would like to thank MZR Global Sdn. Bhd staff for their assistance and cooperation during the data collection phase of this project. Their willingness to provide access to resources and data has substantially improved the system's functionality and efficiency.

I would also like to express my deepest gratitude to my parents, family, and friends for their support and understanding throughout this project. Their unwavering encouragement, motivation, and confidence in my abilities have inspired me constantly.

Lastly, I would like to express my deepest gratitude to all those who have provided support, advice, and encouragement in various capacities but who may not be explicitly mentioned. Your contributions have significantly contributed to the accomplishment of this endeavor.

In conclusion, I am genuinely grateful for the opportunity to undertake this final-year project and for the support and guidance I have received along the way. The knowledge and skills I have gained through this endeavor will undoubtedly shape my future work and contribute to the betterment of special needs children and their onboarding processes.

ABSTRAK

Little Ones Eduworld (LOE), sebuah pusat terapi dan pembelajaran untuk kanak-kanak berkeperluan khas, organisasi ini memerlukan Sistem Pengurusan Onboarding Kanak-Kanak Berkeperluan Khas. Hal ini kerana, prosedur onboarding manual di LOE memepunyai beberapa kelamahan seperti memakan masa, terdedah kepada salah faham dan kehilangan atau kerosakan dokumen. Projek ini menggunakan rangka kerja Laravel untuk menyelesaikan kebimbangan ini dan membina sistem yang berkesan dan automatik. Matlamat termasuk menganalisis sistem semasa, membangunkan Sistem Pengurusan Onboarding Kanak-kanak Berkeperluan Khas, dan menilai fungsinya. Bab pertama menekankan keperluan untuk sistem pengurusan induksi yang dipertingkatkan dengan menonjolkan kesukaran yang berkaitan dengan proses manual. Bab kedua mengkaji kepentingan sistem sedemikian untuk individu berkeperluan khas dengan membandingkan system yang sedia ada. Bab ini menganalisis tiga sistem pengurusan onboarding yang digunakan dalam konteks lain. Disebabkan persamaan mereka, pendekatan manual dan sistem Pengurusan Onboarding Pekerja—SentricHR, Zoho dan ClearCompany dipilih untuk dibandingkan. Sistem projek yang dicadangkan juga diterangkan, dibandingkan dan dianalisis berasama dengan system yang sedia ada yang lain. Bab ini juga membandingkan sistem sedia ada untuk memebina Sistem Pengurusan Onboarding Kanak-Kanak Berkeperluan Khas yang baik. Selain itu, bab ini juga memberi penerangan tentang pengurusan onboarding dan membantu memilih kaedah dan alatan yang sesuai untuk projek yang dicadangkan iaitu Sistem Pengurusan Onboarding Kanak-kanak Berkeperluan Khas. Bab ketiga menerangkan metodologi, bab ini menumpukan pada reka bentuk yang dicadangkan melalui beberapa rajah dan mengguna pakai rangka kerja scrum tangkas sebagai kitaran hayat pembangunan perisian (Software Development Lifecycle). Rangka kerja scrum tangkas digunakan kerana ia menyokong pembangunan dan pelarasan pantas. Manakala bab keempat pula mengkaji pelaksanaan dan ujian sistem, membentangkan keputusan dan perbincangan daripada ujian kebolehterimaan pengguna. Sepuluh modul merangkumi Sistem Pengurusan Onboarding Kanak-Kanak Berkeperluan Khas ini. Pembangunan sistem ini menggunakan Laravel, XAMPP, Mailtrap dan MySQL. Bab ini juga membincangkan keputusan ujian penerimaan pengguna dan ulasan pihak berkepentingan. Bab 4 menunjukkan bahawa sistem berfungsi dengan berkesan, tetapi system ini memerlukan peningkatan dan penambah baikkan. Akhir sekali, bab 5 membincangkan kekangan dan potensi kemajuan. Sistem Pengurusan Penyertaan Kanak-kanak Berkeperluan Khas ialah penyelesaian berharga yang meningkatkan kecekapan dan pengurusan keseluruhan LOE. Projek ini secara berkesan melaksanakan rangka kerja scrum tangkas, menunjukkan potensi pengembangan dan pembangunannya.

ABSTRACT

Little Ones Eduworld (LOE), a therapy and learning centre for special needs children, requires a Special Needs Children Onboarding Management System. The manual onboarding procedure at LOE is time-consuming, prone to misunderstandings, and causes the loss or damage of documents. The project employs the Laravel framework to resolve these concerns and build an effective and automated system. The goals include analyzing the current system, developing the Special Needs Children Onboarding Management System, and assessing its functionality. The first chapter emphasizes the need for an enhanced induction management system by highlighting the difficulties associated with the manual process. The second chapter examines the significance of such a system for individuals with special needs by comparing existing systems. The chapter analyzes three onboarding management systems used in other contexts. Due to their similarities, the manual approach and Employee Onboarding Management systems—SentricHR, Zoho, and ClearCompany are compared. The proposed project's systems are described, compared, and analyzed. This chapter compares existing systems to establish a good Special Needs Children Onboarding Management System. It sheds light on onboarding management and helps choose the proper methods and tools for the project. The third chapter describes the methodology, concentrating on the proposed design through several diagrams and adopting the agile scrum framework. The agile scrum framework is used because it supports rapid development and adjustments. The fourth chapter examines the system's implementation and testing, presenting the results and discussions from user acceptability testing. Ten modules cover onboarding management. The system development uses Laravel, XAMPP, Mailtrap and MySQL. The chapter also discusses user acceptance testing results and stakeholder comments. Chapter 4 shows that the system works effectively, but it needs upgrades. Lastly, chapter 5 discusses constraints and potential progress. The Special Needs Children Onboarding Management System is a valuable solution that improves LOE's overall efficiency and management. The project effectively implements the agile scrum framework, demonstrating its expansion and development potential.

TABLE OF CONTENT

DECLARATION

TITLE PAGE

ACKNOWLEDGEMENTS **ii**

ABSTRAK **iii**

ABSTRACT **iv**

TABLE OF CONTENT **v**

LIST OF TABLES **viii**

LIST OF FIGURES **ix**

LIST OF SYMBOLS **xv**

LIST OF ABBREVIATIONS **xvi**

CHAPTER 1 INTRODUCTION **1**

1.1 Introduction 1

1.2 Problem Statement 3

1.3 Objective 5

1.4 Scope 5

1.5 Thesis Organization 9

CHAPTER 2 LITERATURE REVIEW **10**

2.1 Introduction 10

2.2 Existing Systems/Works 11

2.2.1 Manual approach 11

2.2.2 System A [SentricHR Employee Onboarding Software] 12

2.2.3 System B [Zoho Employee Onboarding System] 13

2.2.4 System C [ClearCompany Employee Onboarding Software] 14

2.3 Analysis / Comparison of Existing System 15

2.3.1 Analysis of comparison on existing system 15

CHAPTER 3 METHODOLOGY 23

3.1 Introduction 23

3.2 Project Management Framework/Methodology 24

3.3 Project Requirement 27

3.3.1 Functional Requirement 27

3.3.2 Non-Functional Requirement 27

3.3.3 Constraints 28

3.3.4 Limitation 28

3.4 Propose Design 29

3.4.1 Flowchart 29

3.4.2 Context Diagram 34

3.4.3 Use Case Diagram and Description 35

3.4.4 Activity Diagram 83

3.4.5 Storyboard 102

3.5 Data Design 152

3.5.1 Entity Relationship Diagram (ERD) 152

3.5.2 Database Dictionary 153

3.6 Proof of Initial Concept 167

3.6.1 Interface Design 167

3.7 Testing/Validation Plan 168

3.8 Potential Use of Proposed Solution 181

3.9 Gantt Chart 182

CHAPTER 4 RESULTS AND DISCUSSION 183

4.1 Introduction 183

4.2 Implementation 183

4.2.1 Development tools 184

4.2.2 Setup Laravel Environment 186

4.2.3 System Interface and Functionality Development 189

4.2.4 Coding Implementation 209

4.2.5 Database Implementation 252

4.3 Results / Findings 256

4.4 Discussions 256

CHAPTER 5 CONCLUSION 257

5.1 Introduction 257

5.2 Objective Revisited 258

5.3 Limitations and Constrains 260

5.4 Recommendations Future Works 261

REFERENCES 262

APPENDIX A 264

APPENDIX B 323

LIST OF TABLES

Table 2.3.1: Analysis of comparison on existing system 15
Table 3.4.1: Use case description Login (UC0001) 36
Table 3.4.2: Use case description Manage User System (UC0002) 38
Table 3.4.3: Use case description Manage Report (UC0003) 41
Table 3.4.4: Use case description Manage IEP (UC0004) 59
Table 3.4.5: Use case description Manage Pre-screening (UC0005) 62
Table 3.4.6: Use case description Manage Payment (UC0006) 66
Table 3.4.7: Use case description Register (UC0007) 68
Table 3.4.8: Use case description Manage Appointment (UC0008) 70
Table 3.4.9: Use case description Manage Profile (UC0009) 75
Table 3.4.10: Use case description Manage Parent Initial Info (M3) (UC0010) 80
Table 3.5.1: User table 153
Table 3.5.2: Parent table 154
Table 3.5.3: Staff table 155
Table 3.5.4: Child table 156
Table 3.5.5: Appointment table 157
Table 3.5.6: IEP table 158
Table 3.5.7: Payment table 164
Table 3.5.8: Pre-screening table 165
Table 3.5.9: M3 table 166
Table 3.7.1: Testing Plan 168
Table 4.2.1: Development tools used and descriptions of the tools 184
Table 5.2.1: Objectives and Results 258

LIST OF FIGURES

Figure 1.2.1: Flowchart of the onboarding management manual process 3
Figure 2.2.1: Assessment result and observation of APB 11
Figure 2.2.2: SentricHR Employee Onboarding Software 12
Figure 2.2.3: Zoho Employee Onboarding System 13
Figure 2.2.4: ClearCompany Employee Onboarding Software 14
Figure 3.1.1: Agile Scrum SDLC 24
Figure 3.4.1: Flowchart for Parent 29
Figure 3.4.2: Flowchart for LOE Staff 30
Figure 3.4.3: Flowchart for OT 31
Figure 3.4.4: Flowchart for Consultant 32
Figure 3.4.5: Flowchart for Admin 33
Figure 3.4.6: Context diagram 34
Figure 3.4.7: Use case diagram 35
Figure 3.4.8: Use case diagram Login (UC0001) 36
Figure 3.4.9: Use case diagram Manage User System (UC0002) 38
Figure 3.4.10: Use case diagram Manage Report (UC0003) 41
Figure 3.4.11: Use case diagram Manage IEP (UC0004) 59
Figure 3.4.12: Use case diagram Manage Pre-screening (UC0005) 62
Figure 3.4.13: Use case diagram Manage Payment (UC0006) 66
Figure 3.4.14: Use case diagram Register (UC0007) 68
Figure 3.4.15: Use case diagram Manage Appointment (UC0008) 70
Figure 3.4.16: Use case diagram Manage Profile (UC0009) 75
Figure 3.4.17: Use case diagram Manage Parent Initial Info (M3) (UC0010) 80
Figure 3.4.18: Activity diagram Login 83
Figure 3.4.19: Activity diagram Manage User System 84
Figure 3.4.20: Activity diagram Manage Report (Admin) 85
Figure 3.4.21: Activity diagram Manage Report (LOE staff) 86
Figure 3.4.22: Activity diagram Manage Report (Parent) 87
Figure 3.4.23: Activity diagram Manage Report (OT & Consultant) 88
Figure 3.4.24: Activity diagram Manage Report (Consultant) 89
Figure 3.4.25: Activity diagram Manage IEP 90
Figure 3.4.26: Activity diagram Manage Pre-screening (Parent) 91
Figure 3.4.27: Activity diagram Manage Pre-screening (LOE staff and OT) 92

Figure 3.4.28: Activity diagram Manage Pre-screening (Admin) 93
Figure 3.4.29: Activity diagram Manage Parent Initial Info (M3) (Parent) 94
Figure 3.4.30: Activity diagram Manage Parent Initial Info (M3) (Admin) 95
Figure 3.4.31: Activity diagram Mange Payment 96
Figure 3.4.32: Activity diagram Register 97
Figure 3.4.33: Activity diagram Manage Appointment (Parent) 98
Figure 3.4.34: Activity diagram Manage Appointment (Admin) 99
Figure 3.4.35: Activity diagram Manage Appointment (LOE staff) 100
Figure 3.4.36: Activity diagram Manage Profile 101
Figure 3.4.37: Login page for Parent 102
Figure 3.4.38: Register page 103
Figure 3.4.39: Parent dashboard page 104
Figure 3.4.40: My Profile page 105
Figure 3.4.41: Edit My Profile page 106
Figure 3.4.42: Change password page 107
Figure 3.4.43: Child Profile page 108
Figure 3.4.44: Edit Child Profile page 109
Figure 3.4.45: Add Child Profile page 110
Figure 3.4.46: Appointment page 111
Figure 3.4.47: Book appointment page 112
Figure 3.4.48: Make Payment page 113
Figure 3.4.49: Questionnaire dashboard page 114
Figure 3.4.50: Pre-screening page 115
Figure 3.4.51: Parent Initial Info page 116
Figure 3.4.52: Parent Initial Info page 2 117
Figure 3.4.53: Report dashboard page 118
Figure 3.4.54: IEP report 119
Figure 3.4.55: IEP report details 120
Figure 3.4.56: Payment report page 121
Figure 3.4.57: Payment details report page 122
Figure 3.4.58: Pre-screening report page 123
Figure 3.4.59: Pre-screening report details page 124
Figure 3.4.60: Child Development screening report page 125
Figure 3.4.61: Child Development screening details report page 126
Figure 3.4.62: Parent Initial Info report page 127

Figure 3.4.63: Parent Initial Info details report page 128
Figure 3.4.64: Login page for Admin and LOE staff 129
Figure 3.4.65: Manage User page 130
Figure 3.4.66: Add User page 131
Figure 3.4.67: Edit user page 132
Figure 3.4.68: Appointment page 133
Figure 3.4.69: Child Development Screening report page 134
Figure 3.4.70: Child Development Screening details report page 135
Figure 3.4.71: IEP report page 136
Figure 3.4.72: IEP report details page 137
Figure 3.4.73: Payment report page 138
Figure 3.4.74: Parent Child Info page 139
Figure 3.4.75: Parent Child Info details page 140
Figure 3.4.76: M3 report page 141
Figure 3.4.77: M3 report details page 143
Figure 3.4.78: Manage Child IEP report page (Staff) 143
Figure 3.4.79: Manage Child IEP report details page 144
Figure 3.4.80: Add IEP page (staff) 145
Figure 3.4.81: Edit IEP page 147
Figure 3.4.82: Pre-screening page (staff) 148
Figure 3.4.83: Pre-screening details page (staff) 149
Figure 3.4.84: Appointment page 150
Figure 3.4.85: Appointment details page 151
Figure 3.5.1: ERD 152
Figure 3.9.1: PSM I Gantt Chart 182
Figure 3.9.2: PSM II Gantt Chart 182
Figure 4.2.1: MVC (Model and Controller) 187
Figure 4.2.2: MVC (Views) 188
Figure 4.2.3: Registration page 189
Figure 4.2.4: Email verification interface 190
Figure 4.2.5: Received email for verification 190
Figure 4.2.6: Login interface 191
Figure 4.2.7: Manage My Account interface 192
Figure 4.2.8: Manage My Profile interface 192
Figure 4.2.9: Manage Child Profile interface 193

Figure 4.2.10: Add Child Profile interface 193
Figure 4.2.11: Edit Child Profile interface 194
Figure 4.2.12: Manage User System interface 194
Figure 4.2.13: Add User interface 195
Figure 4.2.14: Edit User interface 195
Figure 4.2.15: View User details interface 196
Figure 4.2.16: Manage Pre-screening (M1) interface 197
Figure 4.2.17: Update Child LOE Programme interafce (LOE staff [Manager/Principal] and OT) 198
Figure 4.2.18: Manage Appointment interface (Admin) 199
Figure 4.2.19: Manage Appointment interface (Parent) 199
Figure 4.2.20: Book Appointment interface (Parent) 200
Figure 4.2.21: Manage Appointment interface (LOE staff [Manager/Principal]) 200
Figure 4.2.22: Update PIC interface (LOE staff [Manager/Principal]) 201
Figure 4.2.23: Manage Parent Initial Info (M3) interface 202
Figure 4.2.24: Manage Payment interface 203
Figure 4.2.25: Manage IEP (admin) interface 203
Figure 4.2.26: Add IEP interface 204
Figure 4.2.27: View IEP interface 205
Figure 4.2.28: Pre-screening Report interface 205
Figure 4.2.29: M1 details interface 206
Figure 4.2.30: Payment Report interface 206
Figure 4.2.31: Child and Parent Info Report interface 207
Figure 4.2.32: Child and Parent Info details interface 207
Figure 4.2.33: Appointment Report interface 207
Figure 4.2.34: M3 Report interface 208
Figure 4.2.35: Fillable User Model 209
Figure 4.2.36: Relationship in User Model 209
Figure 4.2.37: ParentRecord Model 210
Figure 4.2.38: StaffRecord Model 211
Figure 4.2.39: ChildRecord Model 212
Figure 4.2.40: M1Record Model 213
Figure 4.2.41: AppointmentRecord Model 213
Figure 4.2.42: M3Record Model 214
Figure 4.2.43: PaymentRecord Model 215

Figure 4.2.44: Home Controller 216
Figure 4.2.45: User Controller (index, show, create function) 217
Figure 4.2.46: User Controller (store function) 218
Figure 4.2.47: User Controller (edit, editProfile, update function) 219
Figure 4.2.48: User Controller (updateProfile, destroy function) 220
Figure 4.2.49: Child Controller (index function) 221
Figure 4.2.50: Child Controller (profile function) 222
Figure 4.2.51: Child Controller (create, edit, store function) 223
Figure 4.2.52: Child Controller (update, destroy function) 224
Figure 4.2.53: M1 Controller (index, childList function) 225
Figure 4.2.54: M1 Controller (m1Details function) 226
Figure 4.2.55: M1 Controller (store function) 227
Figure 4.2.56: M1 Controller (update function) 227
Figure 4.2.57: Appointment Controller (index, create function) 228
Figure 4.2.58: Appointment Controller (store, edit, update, destroy function) 229
Figure 4.2.59: M3 Controller (index function) 230
Figure 4.2.60: M3 Controller (store function) 231
Figure 4.2.61: Payment Controller (index, report, store function) 232
Figure 4.2.62: Registration blade 233
Figure 4.2.63: Registration blade (2) 233
Figure 4.2.64: Login blade 234
Figure 4.2.65: Add User blade 235
Figure 4.2.66: Edit User blade 236
Figure 4.2.67: Manage User Account blade 237
Figure 4.2.68: Manage User Profile blade 238
Figure 4.2.69: Manage User Profile blade (2) 238
Figure 4.2.70: Manage User Profile blade (3) 239
Figure 4.2.71: Manage Pre-screening (M1) blade 240
Figure 4.2.72: Manage Pre-screening (M1) blade (2) 241
Figure 4.2.73: Manage Pre-screening (M1) (Update LOE Programme) blade 242
Figure 4.2.74: Manage Pre-screening (M1) (Update LOE Programme) blade (2) 243
Figure 4.2.75: Manage Appointment blade 244
Figure 4.2.76: Add Appointment blade 245
Figure 4.2.77: Assign PIC for Appointment blade 246
Figure 4.2.78: Manage Parent Initial Info (M3) 247

Figure 4.2.79: Manage Payment 248
Figure 4.2.80: M1 report blade 249
Figure 4.2.81: Payment report blade 250
Figure 4.2.82: Child and Parent Info blade 251
Figure 4.2.83: web.php 252
Figure 4.2.84: Appointments table 252
Figure 4.2.85: Childs table 252
Figure 4.2.86: M1s table 253
Figure 4.2.87: M3s table 254
Figure 4.2.88: Parents table 254
Figure 4.2.89: Payments table 254
Figure 4.2.90: Staffs table 255
Figure 4.2.91: Users table 255

LIST OF SYMBOLS

None.

LIST OF ABBREVIATIONS

LOE	Little Ones Eduworld
UAT	User Acceptance Test
PDPA	Personal Data Protection Act 2010
OT	Occupational Therapy
M1	Pre-screening Test
M3	Parent Initial Info
P1	Perjumpaan Pertama (First Meeting)
P2	Perjumpaan Kedua (Second Meeting)
UNICEF	United Nations International Children's Emergency Fund
ADHD	Attention Deficit Hyperactivity Disorder
USA	United State of America
Inc,	Incorporated
SDN BHD	Sendirian Berhad
IEP	Individualized Education Plan
GARS-2	Gilliam Autism Rating Scale
DDST II	Denver Developmental Screening Test
SSP	Short Sensor Profile
IDE	Integrated Development Environment
PHP	Hypertext Preprocessor
MySQL	My Structured Query Language
HR	Human Resources
ACA	Accelerated Capital Allowance
HRMS	Human Resources Management System
MCO	Movement Control Order
COVID-19	Corona Virus Disease 2019
SDLC	Software Development Lifecycle
PK	Primary Key
FK	Foreign Key
MVC	Model View Controller
HTML	HyperText Markup Language
CSS	Cascading Style Sheets

CHAPTER 1

INTRODUCTION

1.1 Introduction

Children are the pillar of hope for the country's aspirations for the future. Children are an essential asset to the development and will inherit the country's leadership (*Kanak-kanak Aset Penting Bangunkan Negara*, 2018). Therefore, parents need to work hard to build their children's potential by giving them the best education so that they can evaluate something, whether it is good or otherwise (mt961013, 2018). This shows that education is critical in the foundation of the formation of intellectual children. According to Preston et al. (1996), quality education is very effective in fighting poverty, building democracy, and forming a prosperous society. This is no exception for special children because they also play an important role in the nation. According to a study by UNICEF Malaysia (2018), 10% of respondents think that children with disabilities do not need to go to school. This matter is unreasonable and should be resolved because every child has the right to education. However, the educational approach for children with disabilities may be slightly different compared to other normal children. This is because, according to a UNICEF Malaysia study (2018), more than 70% think that children with mental disabilities should not be in the same schools as children without disabilities.

Therefore, a centre called Little One Eduworld (LOE) was established. LOE is a centre that offers therapy and learning programmes for special needs children such as dyslexia, Attention Deficit Hyperactivity Disorder (ADHD) and speech delay. This centre is vital to cater to special children's needs in providing education. Therefore, LOE requires a system that can facilitate the management of onboarding and monitoring of the child's development. The Onboarding Management system is an improved approach to acquainting students with the institution or therapy centre, the processes, and the requirements they would need to satisfy to have a better overall experience as a student (Alban, 2022). LOE has done this process manually, which is no longer appropriate

nowadays. It will cause various problems, such as being time-consuming, misunderstandings, miscommunication, and many more. Even worse, these issues can diminish the quality of management and the reputation of LOE. Hence, this project, Special Needs Children Onboarding Management System, is a system that will replace the manual process to assist the LOE staff and parents in managing the onboarding process for new students more efficiently.

1.2 Problem Statement

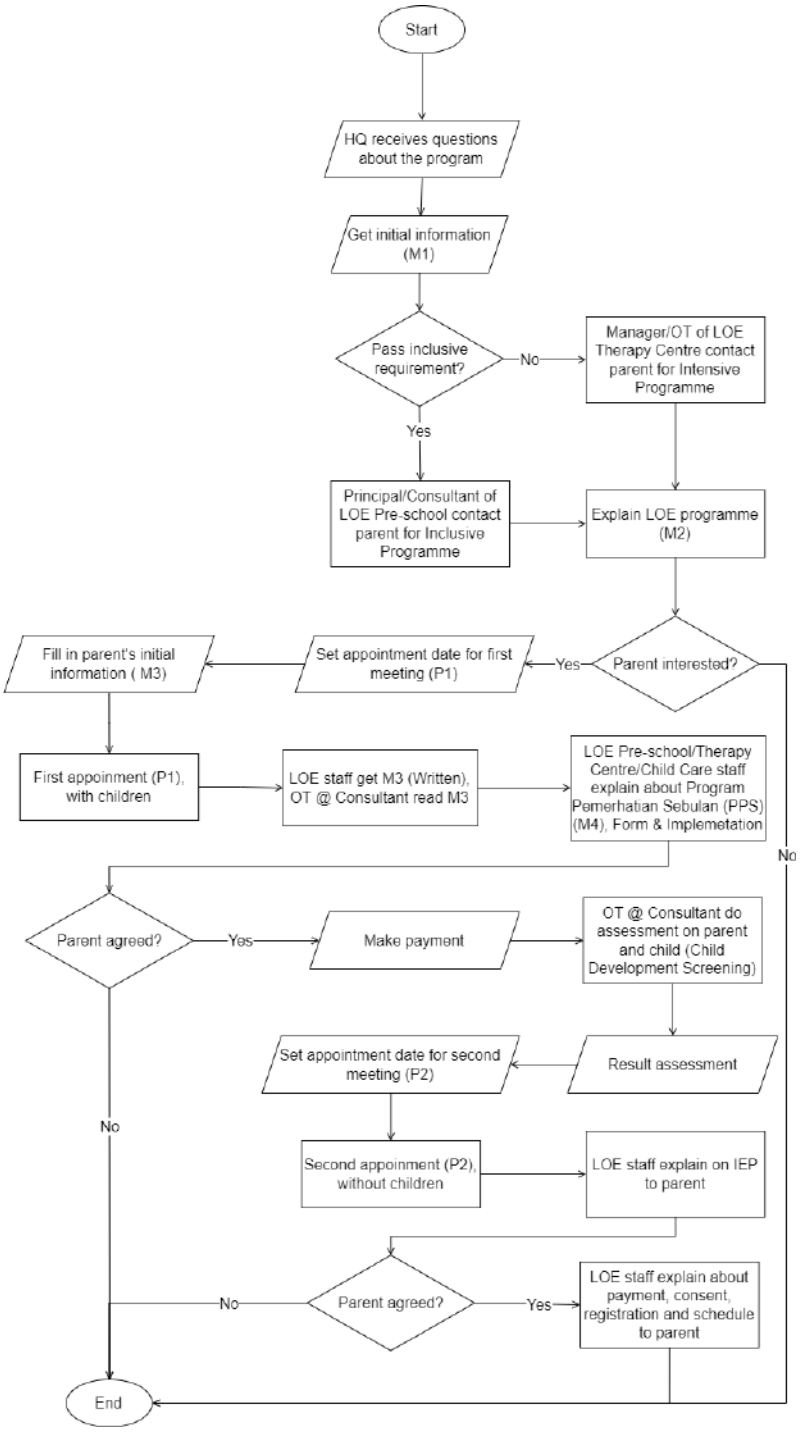


Figure 1.2.1: Flowchart of the onboarding management manual process

Onboarding Management System is essential for a therapy centre like the LOE centre to ensure that these special needs children can familiarize themselves with the learning environment at the centre, and the LOE staff can make observations and

assessments of the children. However, the manual approach of LOE's onboarding management process presents some issues. Figure 1.2.1 shows a flowchart of the onboarding manual process.

Among the problems encountered during the manual process carried out by LOE is that it is very time-consuming to manage to onboard manually because the number of new student admission applications to the centre annually is numerous and unpredictable. In addition, there is a lot of paperwork to be managed, which will make finding the necessary documents difficult when time is pressed. Because of that, parents will not be able to receive the latest information from the LOE centre.

Next, the occurrence of misunderstanding and miscommunication in understanding the writing on the paper. The survey of 2,000 Americans, conducted by OnePoll on behalf of Bic USA Inc., also found that the average respondent had experienced in their working life where their handwriting led to miscommunication (Sadlier, 2019). This can affect the observation of special needs children to a satisfactory level due to misunderstandings in reading the description of the needs of special students.

In addition, the use of paper during getting initial information (M1), initial information for parents for the first meeting (M3) and the first-month observation program (P1) often happens missing and damaged. This can prevent the management process of students to the next level. Therefore, the occurrence of delays and the onboarding processes are interrupted. This negligence significantly impacts the monitoring process and the development of special needs children at the centre.

Also, these can cause neglect of management during the process happens. Thus, the Special Needs Children Onboarding Management System will help them to store, secure and manage the data efficiently. Besides, the users can easily track the needed documents by searching them in the system. Furthermore, the system shall generate reports quickly for the user. In conclusion, this system gives a lot of benefits to the users to work more seamlessly and simultaneously reduce paper usage to save the environment.

1.3 Objective

The objective of this project are:

i. To study the existing system related to Special Needs Children Onboarding Management System.

ii. To develop a Special Needs Children Onboarding Management System using the Laravel framework.

iii. To evaluate the functionality of the Special Needs Children Onboarding Management System.

1.4 Scope

The scope of this project are listed as below:

i. User scope:

   a. Admin

   b. LOE staff (Principal/Managers)

   c. Parent

   d. Consultant

   e. Occupational Therapy (OT)

ii. Features:

   a. Admin:

      i. Login

      ii. Manage user system

      iii. Manage pre-screening

      iv. Manage parent initial info (M3)

      v. Manage appointment

vi. Manage report

b. LOE staff (Principal/Managers):

i. Login

ii. Manage pre-screening

iii. Manage appointment

iv. Manage report

c. Parent:

i. Register

ii. Login

iii. Manage profile

iv. Manage pre-screening

v. Manage appointment

vi. Manage payment

vii. Manage parent initial info (M3)

viii. Manage report

d. Consultant

i. Login

ii. Manage child development screening, Gilliam Autism Rating Scale (GARS-2). (will be redirected to an external system named Child Developmental Screening Test & Progress Monitoring System)

iii. Manage report

e. Occupational Therapy (OT)

i. Login

ii. Manage pre-screening

iii. Manage child Individualized Education Plan (IEP)

iv. Manage child development screening, Denver Developmental Screening Test (DDST II), Short Sensor Profile (SSP). (will be redirected to an external system named Child Developmental Screening Test & Progress Monitoring System)

v. Manage report

iii. Hardware:

a. Monitor

b. Keyboard

c. Mouse

d. 4GB RAM

e. Intel core i5 processor

iv. Software:

a. Framework: Laravel

b. Integrated Development Environment (IDE): Visual Code

c. Language: PHP, HTML, CSS, JavaScript

d. Database: MySQL

e. Version control: Git Hub

1.5 Thesis Organization

This thesis consists of five chapters. Chapter 1 discusses the introduction of the project. Chapter 2 generally explains the literature review and comparison with the existing system. Next, chapter 3 discusses the methodology that is used for the project and describes project's requirement through proposed design such as Context Diagram, Use Case Diagram, Activity Diagram, and flowchart. Then, chapter 4 shall explain the implementation and testing of the system. Lastly, chapter 5 will conclude all the chapters and provide some recommendations.

CHAPTER 2

LITERATURE REVIEW

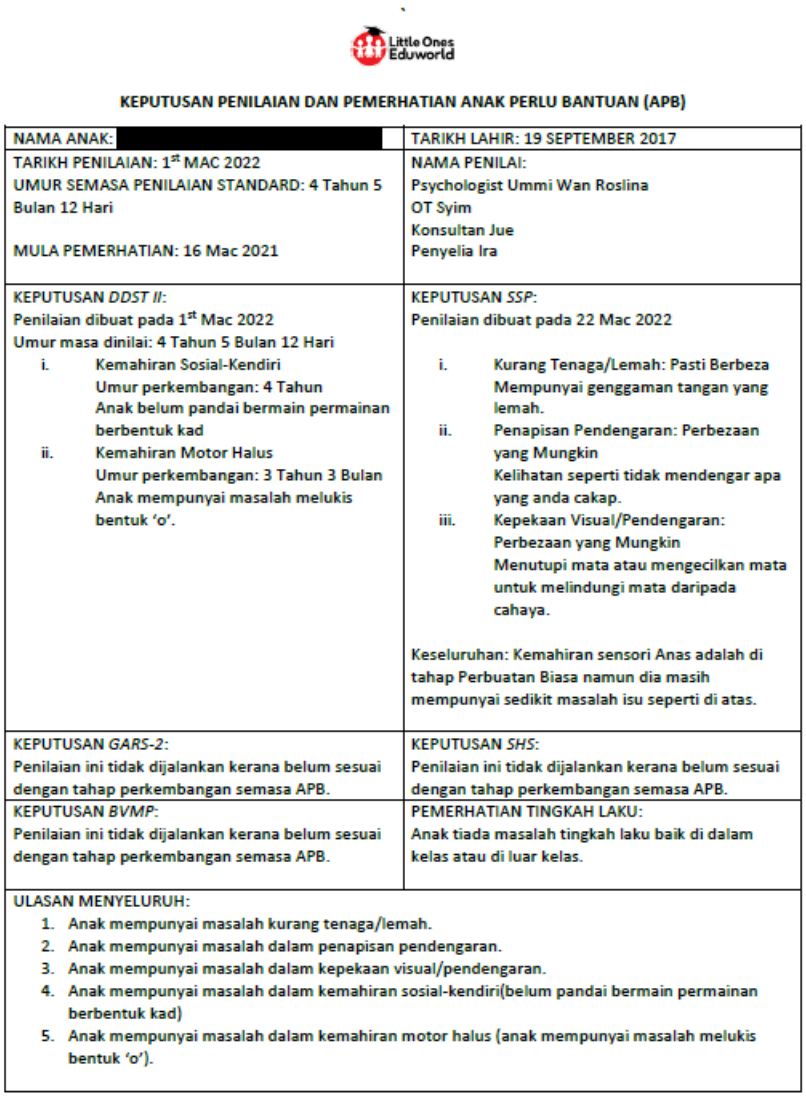
2.1 Introduction

The onboarding management system is essential, particularly for children with special needs who wish to acquire an education. This approach can evaluate these special needs children and assist them in adapting to an appropriate educational setting. Consequently, parents play a significant role in the growth of these children by giving them the necessary education to assess their children's abilities. In addition, experts, such as occupational therapists and consultants, are also required to observe and evaluate the children to receive an appropriate education. Traditional working methods and several similar systems exist to facilitate onboarding management. However, since open-source onboarding management systems for special needs children are not yet widely available, comparison resources are limited. Therefore, since the Employee Onboarding Management system has almost the same characteristics as the proposed system, a decision has been taken to use the Employee Onboarding Management system as a comparison.

Thus, this chapter will define the manual approach and three existing onboarding management systems: the SentricHR Employee Onboarding Software, Zoho Employee Onboarding system and ClearCompany Employee Onboarding Software. It consists of descriptions and comparisons of the project and existing systems. This chapter also explains the advantages, disadvantages, techniques, methods, tools and technologies that can be adapted to the project.

2.2 Existing Systems/Works

2.2.1 Manual approach



Little Ones Eduworld

KEPUTUSAN PENILAIAN DAN PEMERHATIAN ANAK PERLU BANTUAN (APB)

NAMA ANAK: [redacted]	TARIKH LAHIR: 19 SEPTEMBER 2017
TARIKH PENILAIAN: 1st MAC 2022 UMUR SEMASA PENILAIAN STANDARD: 4 Tahun 5 Bulan 12 Hari MULA PEMERHATIAN: 16 Mac 2021	NAMA PENILAI: Psychologist Ummi Wan Roslina OT Syim Konsultan Jue Penyelia Ira
KEPUTUSAN *DDST II*: Penilaian dibuat pada 1st Mac 2022 Umur masa dinilai: 4 Tahun 5 Bulan 12 Hari i. Kemahiran Sosial-Kendiri Umur perkembangan: 4 Tahun Anak belum pandai bermain permainan berbentuk kad ii. Kemahiran Motor Halus Umur perkembangan: 3 Tahun 3 Bulan Anak mempunyai masalah melukis bentuk 'o'.	KEPUTUSAN *SSP*: Penilaian dibuat pada 22 Mac 2022 i. Kurang Tenaga/Lemah: Pasti Berbeza Mempunyai genggaman tangan yang lemah. ii. Penapisan Pendengaran: Perbezaan yang Mungkin Kelihatan seperti tidak mendengar apa yang anda cakap. iii. Kepekaan Visual/Pendengaran: Perbezaan yang Mungkin Menutupi mata atau mengecilkan mata untuk melindungi mata daripada cahaya. Keseluruhan: Kemahiran sensori Anas adalah di tahap Perbuatan Biasa namun dia masih mempunyai sedikit masalah isu seperti di atas.
KEPUTUSAN *GARS-2*: Penilaian ini tidak dijalankan kerana belum sesuai dengan tahap perkembangan semasa APB.	KEPUTUSAN *SHS*: Penilaian ini tidak dijalankan kerana belum sesuai dengan tahap perkembangan semasa APB.
KEPUTUSAN *BVMP*: Penilaian ini tidak dijalankan kerana belum sesuai dengan tahap perkembangan semasa APB.	PEMERHATIAN TINGKAH LAKU: Anak tiada masalah tingkah laku baik di dalam kelas atau di luar kelas.
ULASAN MENYELURUH: 1. Anak mempunyai masalah kurang tenaga/lemah. 2. Anak mempunyai masalah dalam penapisan pendengaran. 3. Anak mempunyai masalah dalam kepekaan visual/pendengaran. 4. Anak mempunyai masalah dalam kemahiran sosial-kendiri(belum pandai bermain permainan berbentuk kad) 5. Anak mempunyai masalah dalam kemahiran motor halus (anak mempunyai masalah melukis bentuk 'o').	

1

Figure 2.2.1: Assessment result and observation of APB

Little Ones Eduworld currently manage special needs children onboarding manually. The parent, LOE staff, Occupational Therapy and Consultant need to manually fill in all the form by handwriting and type it on the computer using Microsoft Word. They are flexible in writing it in English or Malay as long as it is understandable. Figure 2.2.1

shows the result of the child development screening process. The user needs to key in all the information, such as the children's name, DDST II result, SSP result etc., on their own. There are some advantages when managing the onboarding manually, such as giving it much more comfortable for the computer illiterate to manage it and flexibility to write or draw in it. On the other hand, there are some disadvantages to managing it manually, such as difficulty finding certain documents and a lot of paper usage because the document needs to print out to get the parent's signature.

2.2.2 System A [SentricHR Employee Onboarding Software]

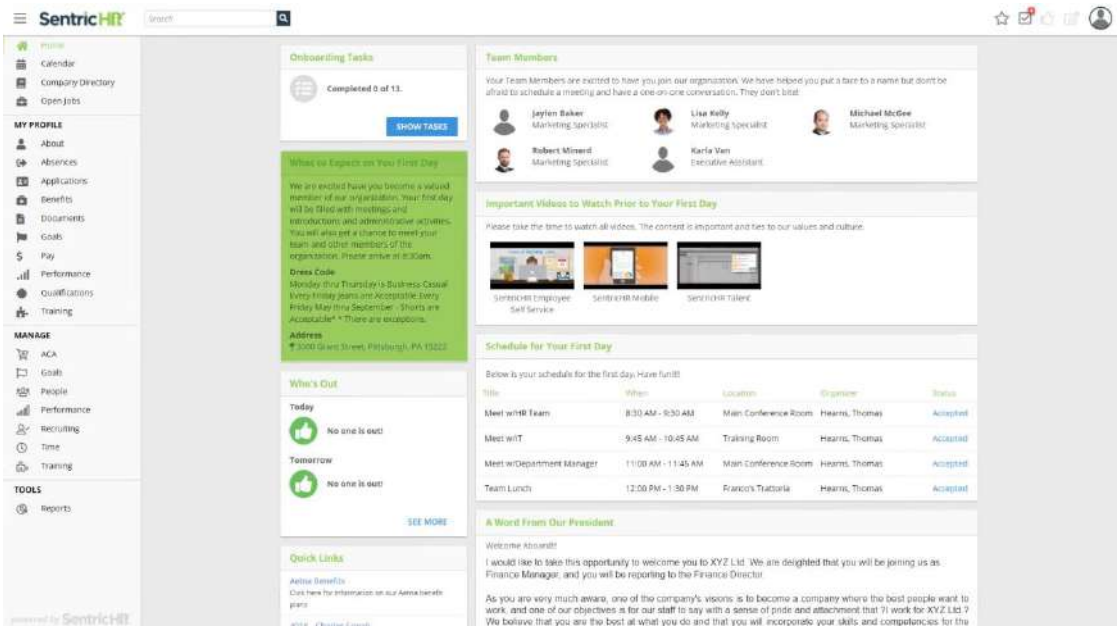


Figure 2.2.2: SentricHR Employee Onboarding Software

SentricHR developed an onboarding system named Employee Onboarding System (as shown in Figure 2.2.2). This software is used to monitor the development of new employees and improve their skills. And it is similar to the proposed system, where the Special Needs Children Onboarding Management System is used to see the development of the children and improve their abilities by going through several assessments and therapies. This SentricHR software avoids redundant data entry and fills in blanks with information already on file for the user. In addition, electronic forms and ESIGN-compliant signatures allow recruits to sign important papers before entering the office. It can be done anywhere because this system can be accessed by smartphone. This software replaces stacks of paper records with digital files to make employee onboarding

more efficient, accurate, and compliant. Also, it enables new employees to digitally meet their team members to increase employee engagement (SentriHR, 2022). However, the language provided is limited to English, and some of the software features are not user-friendly (Matchr, 2022).

2.2.3 System B [Zoho Employee Onboarding System]

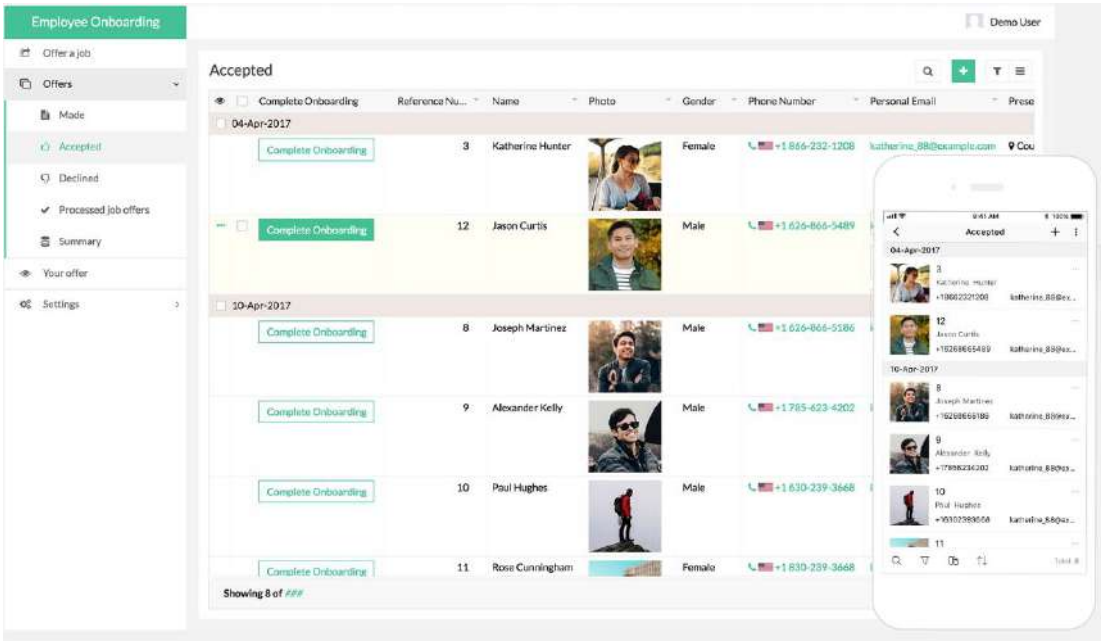


Figure 2.2.3: Zoho Employee Onboarding System

Zoho Employee Onboarding System (as shown in Figure 2.2.3) is very special because it comes with a mobile application. So, it is easy for them to access the system everywhere. The digital onboarding procedure directs candidate communications and quickens their transfer into new roles. This online employee onboarding programme is fully customizable, which is the finest feature. Zoho Employee Onboarding System is also integrated with many tools to create a seamless flow. This system can also customize the portal to suit users' needs. This system provides many exciting features, such as an automated hiring process, auto notification, and electronically signing forms. However, some features need to be enhanced, especially the system's performance.

2.2.4 System C [ClearCompany Employee Onboarding Software]

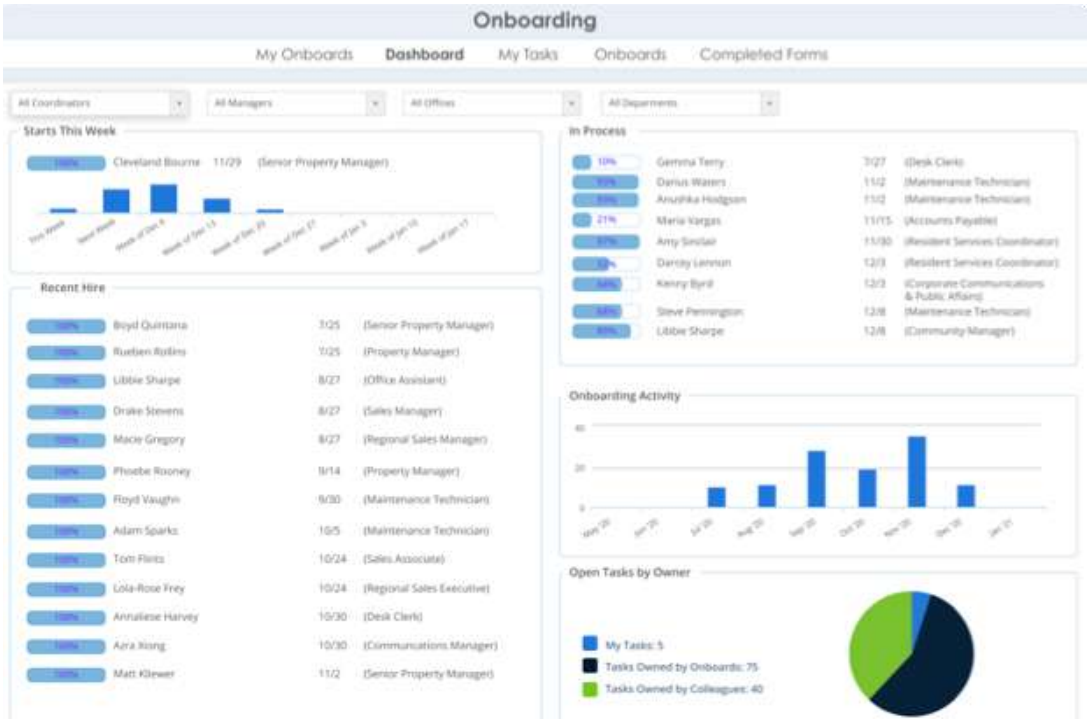


Figure 2.2.4: ClearCompany Employee Onboarding Software

ClearCompany develops Employee Onboarding Software (as shown in Figure 2.2.4) that provides a user-friendly option for tracking applicants. As a result of integrating the assessment and recruiting procedures into a single platform, candidate screening will now be considerably less time-consuming. In addition, the system provides solutions for keeping employees' documents safe and secure while making them easily accessible. The automated processes allow employees to keep track of new hire paperwork and collect it. This may be accomplished through the use of user-friendly and smart forms. Because it is so intelligent, employees do not need to repeatedly enter the information that an employee has already supplied in the past. ClearCompany Employee Onboarding Software, on the other hand, will pre-fill the data for employees, validate it, and precisely format it (*ClearCompany Reviews*, 2021).

2.3 Analysis / Comparison of Existing System

2.3.1 Analysis of comparison on existing system

Table 2.3.1: Analysis of comparison on existing system

Features/Systems	**Manual approach**	**SentricHR Employee Onboarding Software**	**Zoho Employee Onboarding system**	**ClearCompany Employee Onboarding Software**
Tools/Technology used	Microsoft Word	Accelerated Capital Allowance (ACA)	Not mentioned	Not mentioned
Language	Malay/English	English	English	English

Accessed platform	Paper / Word processing software	Smartphone PC/Desktop Laptop Tablet	Web based & Mobile apps, Smartphone PC/Desktop Laptop	Web based, PC/Desktop
Integrates	-	Payroll system HR software Etc.	HRMS Payroll system Etc.	Paycom Kronos Workday Etc.

Technique/Method		Typing/Write	Not mentioned	Not mentioned	Not mentioned
Features	Childs/Employee Info	/	/	/	/
		Child info such as name and date of birth	Employee info such as name, email, phone number and position	Employee info such as name, email, phone number and position	Employee info such as name, email, phone number and position
	Virtual meet	/	/	x	/
		During COVID-19 Movement Control Order (MCO) using Google Meet/Whatsapp	Meet team virtually	-	Engage with employees virtually

		Video Call platform			
	Automated features	x	/	/	/
		-	Auto notification to inform employee	Automated hiring process Auto notification to inform employee	Automated online delivery, tracking, collection, and storage of all the necessary onboarding documents and forms (Doctor, 2021) Auto notification to inform employee

	Track Progress	/	/	/	/
		Child Pre-screening Child Development Screening	Track employee progress	Track new-hire applicant status	Track recruiting
	E-Signatures	x	/	/	x
		-	Electronically sign forms	Electronically sign forms	-
	Planning/Scheduler	/	x	/	/
		Children IEP	-	Scheduler and new hire checklist	Workforce planning

	Report	/	/	/	/
		Child pre-screening report Child Development Screening report	New hire report	Employee progress report	Employee grade and progress report
Pros		Suitable for computer illiterate Flexible to write and draw	Paperless process Improve employee engagement. Auto notification alert to make new employee stay alert.	Receive automatic welcome email. Automatically notify new hires to submit document or key in details. Personalized employee portal	The software saves money, time, and errors through the software's smart forms. Can also be accessed from any device. Automated workflows let employee easily set

		Customize onboarding templates. Safely view and sign document from own devices.	Completing tasks on the go	deadlines and reminders
Cons	A lot of paperwork Hard to find paperwork	The report is difficult to learn. Limited language provided. (*SentricHR Reviews*, 2022)	Lack of performance No error displayed when fail to check in/out. (*Zoho People Reviews*, 2022)	Confusing interfaces and features (*ClearCompany Reviews*, 2021)

2.4 Summary

Based on the comparison that has been made on the three existing systems, it can be concluded that the system must be user-friendly and minimize the features. Instantly, it can improve the system's performance to work more efficiently. Therefore, the findings that have been discovered in this chapter will be used to develop the Special Needs Children Onboarding Management System.

CHAPTER 3

METHODOLOGY

3.1 Introduction

This chapter discuss about the overall approach of Special Needs Children Onboarding Management System project. This project will be developed using Agile Scrum Software Development Life Cycle (SDLC) (as shown in Figure 3.1.1) which consist of Product Backlog, Sprint Backlog and Increment/Demo. Scrum framework is suitable to be used for this project since it facilitates collaborative teamwork, and its execution can be adjusted to meet the specific requirements of the organization (Atlassian & Drumond, 2018).

The content for this chapter contains: Introduction, Project Management Framework, Project Requirement, Propose Design, Data Design, Proof of Initial Concept, Testing Plan, Potential Use of Proposed Solution and Gantt Chart.

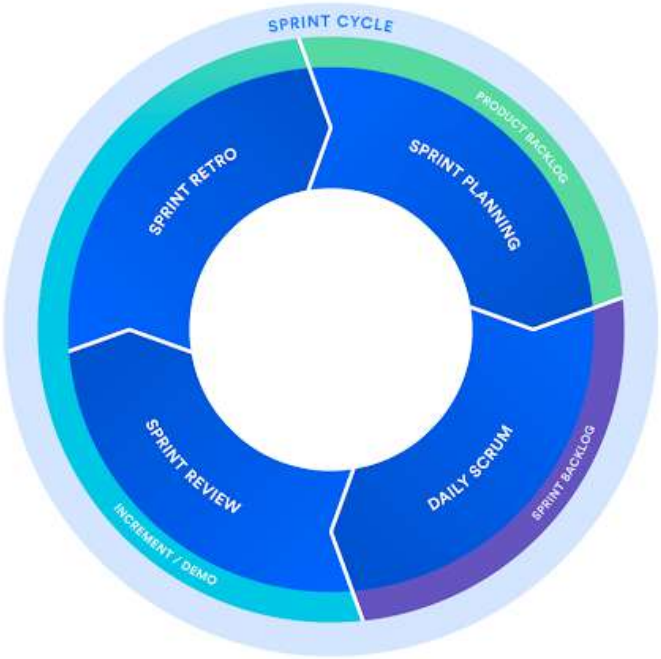


Figure 3.1.1: Agile Scrum SDLC

3.2 Project Management Framework/Methodology

This project utilises the agile scrum framework since it is ideal for teams working together more efficiently. Agile and scrum are methods that combine agile philosophy and scrum framework. Agile allows teams to develop projects in small increments. Scrum, on the other hand, is a technique recognised for dividing tasks into sizeable chunks called "sprints". This framework is appropriate for whom, wish to complete specific projects swiftly. This is because this structure supports rapid project development. After all, each set of objectives must be accomplished within the sprint time. Additionally, this framework is flexible. The teams can quickly and easily update the product objectives if stakeholders make any changes (Peek, 2022). Since this project is involved and related to other team members with different projects individually, the agile scrum framework is considered to be chosen as the methodology throughout the project development.

The phases and process of the agile scrum framework used throughout the development of Special Needs Children Onboarding Management system are:

i. Product Backlog

The product owner or related stakeholders are involved in managing the product backlog. The product owner will provide all necessary plans and task details for the project's development. The features, requirements, objectives, and flow throughout the project's development will be produced and addressed in collaboration with the scrum master before the sprint planning meeting when they will be conveyed to the development team.

a. Sprint Planning Meeting

Google Meet is used to host an online meeting session for the sprint planning meeting with the product owner or other stakeholders from MZR Global. Involved in the online meeting are the scrum master, team member supervisor, and development team. The scrum master assigned the project to each member of the four-person development team on an individual basis. During this meeting, the team will decide on the target for the sprint. The stories must always align with the goal, and the scrum team must agree that it will be possible to implement them throughout the sprint. The tasks are documented on a Gantt chart, making tracking their progress much more accessible. After the planning meeting, each scrum team member must clearly understand what can be delivered during the sprint and how the increment may be delivered.

ii. Sprint Backlog

The needed tasks from the sprint backlog will be specified and executed by a development team member. The tasks must be accomplished within the sprint timeframe. Each development team member determined and planned to complete the tasks within the sprint period independently. A sprint backlog could be flexible and might undergo change while a sprint is in progress. However, the core purpose of the sprint, which is what the team hopes to accomplish with this particular sprint, must not be compromised.

a. Daily Scrum or stand up

A daily stand-up or scrum is a meeting held every two weeks for thirty minutes through an online meeting using the Google Meet platform. Members of the development team, the scrum master, and the supervisors attended the meeting. Every two weeks, each

development team will provide the scrum master with an update on their progress during the daily scrum meeting. The development team member will also describe any obstacles or limitations encountered during the task's progression and the planned task's progression for the following weeks. The scrum master also guided and encouraged the development team member to resolve the encountered restrictions and problems and commit to the sprint target.

iii. Increment / Demo

The Increment is the aggregate of all sprint-developed tasks, use cases, user stories, product backlogs, and any other aspect that will be made accessible to the end user in the form of software (Singh, 2022). During the Increment or Sprint Goal phase, the development team will demonstrate what was accomplished during the sprint.

a. Sprint Review

At the end of the Sprint, an online meeting will be held between the product owner, scrum master, development team members, and supervisors to view the demo of, or examine, the increment. A member of the development team will give a presentation on the completed project's overall results. Then, the product owner will give feedback and will decide whether to release the increment or not.

b. Sprint Retrospective

During the retrospective, the team development gets together with the product owner, scrum master and supervisor to document and discuss the aspects of a sprint that went well and those that needed improvement. The objective of the sprint is to identify potential process enhancements and develop a plan for implementing them in the following sprint.

3.3 Project Requirement

3.3.1 Functional Requirement

i. The LOE Onboarding Management system shall allow the users to login into the system.
ii. The LOE Onboarding Management system shall provide the admin with ability to manage user system.
iii. The LOE Onboarding Management system should able to display the reports.
iv. The LOE Onboarding Management system should allow the OT to manage child IEP.
v. The parent should be able to click on submit button to submit the pre-screening questionnaires (M1).
vi. The parent should be able to choose date and time and click submit button to set the appointment date.
vii. The parent should be able to click on submit button to submit the parent's initial information (M3).
viii. The LOE Onboarding Management system should be able to display total payment for parent.
ix. The LOE Onboarding Management system should allow the LOE staff and OT to update child programme type.
x. The parent should be able to enter payment details and click on button to make a payment.
xi. The LOE Onboarding Management system shall provide the parent with ability to register an account.
xii. The LOE Onboarding Management system shall allow the users to receive notifications.

3.3.2 Non-Functional Requirement

i. Security requirement
    - The login session expires after 120 minutes if in the idle state.
    - The users required to create a strong password when register an account. The password must be more than 8 characters.
    - The user account information must be Personal Data Protection Act (PDPA) 2010 compliant.
ii. Usability requirement
    - The LOE Onboarding Management system user interface is user friendly and easy to use.

iii. Availability requirement

- The LOE Onboarding Management system is required to have 99.99% avaiability.

iv. Localization requirement

- The date format must be as follows: day / month / year
- The time format must be as follows: hour : minute

v. Performance requirement

- The LOE Onboarding Management system must able to respond within or less than 3 seconds.

3.3.3 Constraints

i. Parent is unable to proceed with child development process if they do not make payment and do pre-screening.

ii. Parents are unable to set appointments for second meeting if the result assessment is not released yet.

iii. LOE staff, OT and consultant must be registered by admin only.

iv. Parents must register by themselves to create an account.

3.3.4 Limitation

i. Users must connect to the internet to be able to use the system.

ii. The system can be operated through website only.

iii. Users are recommended to use Chrome browser with the latest version.

3.4 Propose Design

3.4.1 Flowchart

3.4.1.1 Flowchart for user Parents

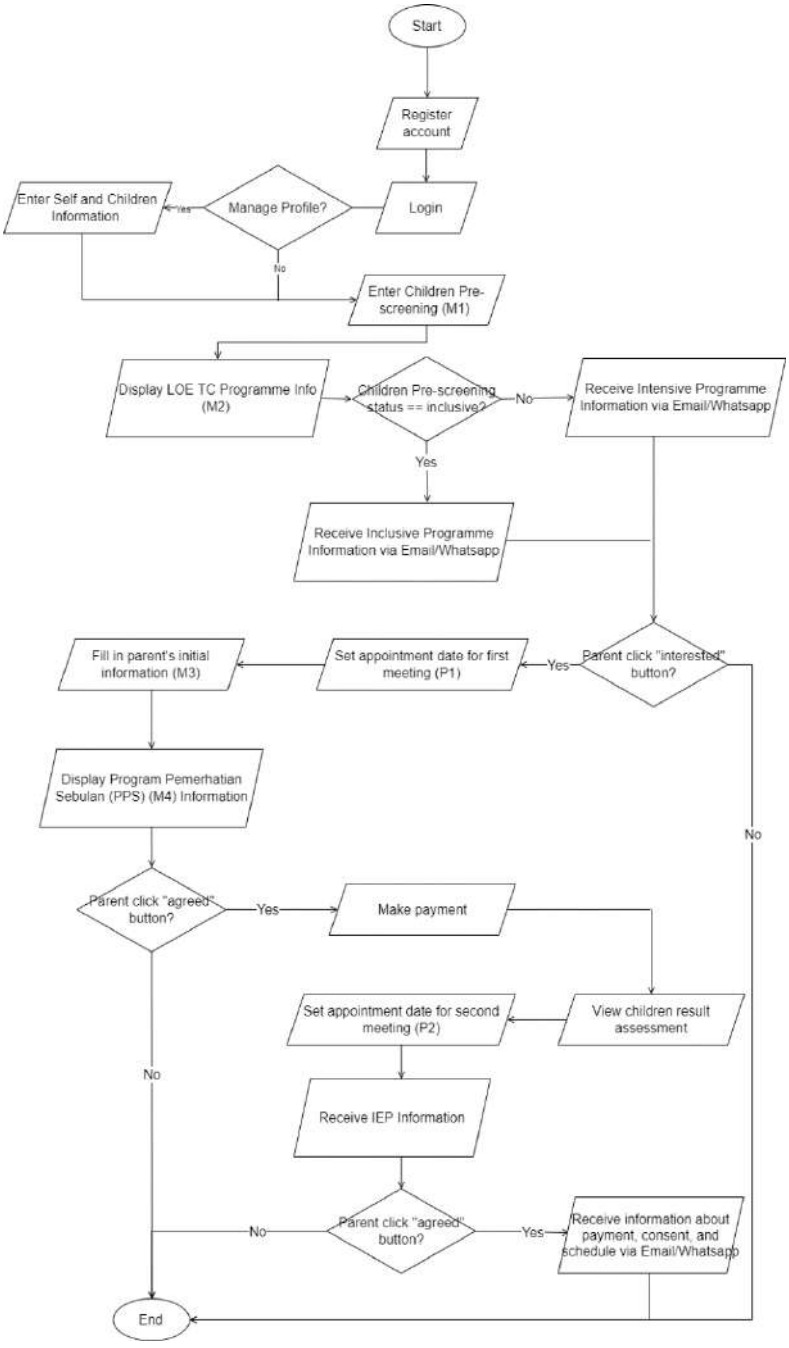


Figure 3.4.1: Flowchart for Parent

3.4.1.2 Flowchart for user LOE staffs

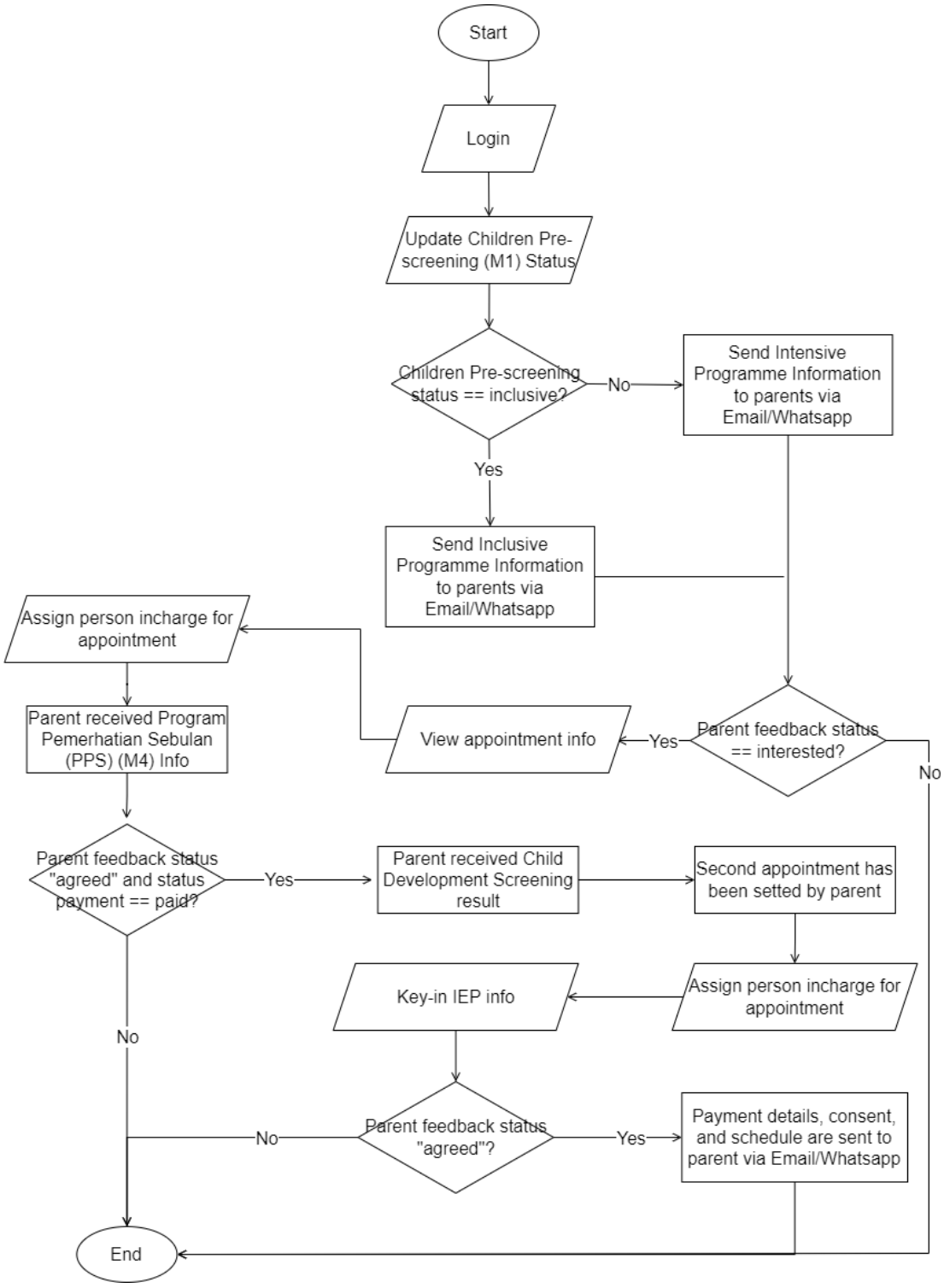


Figure 3.4.2: Flowchart for LOE Staff

3.4.1.3 Flowchart for user OT

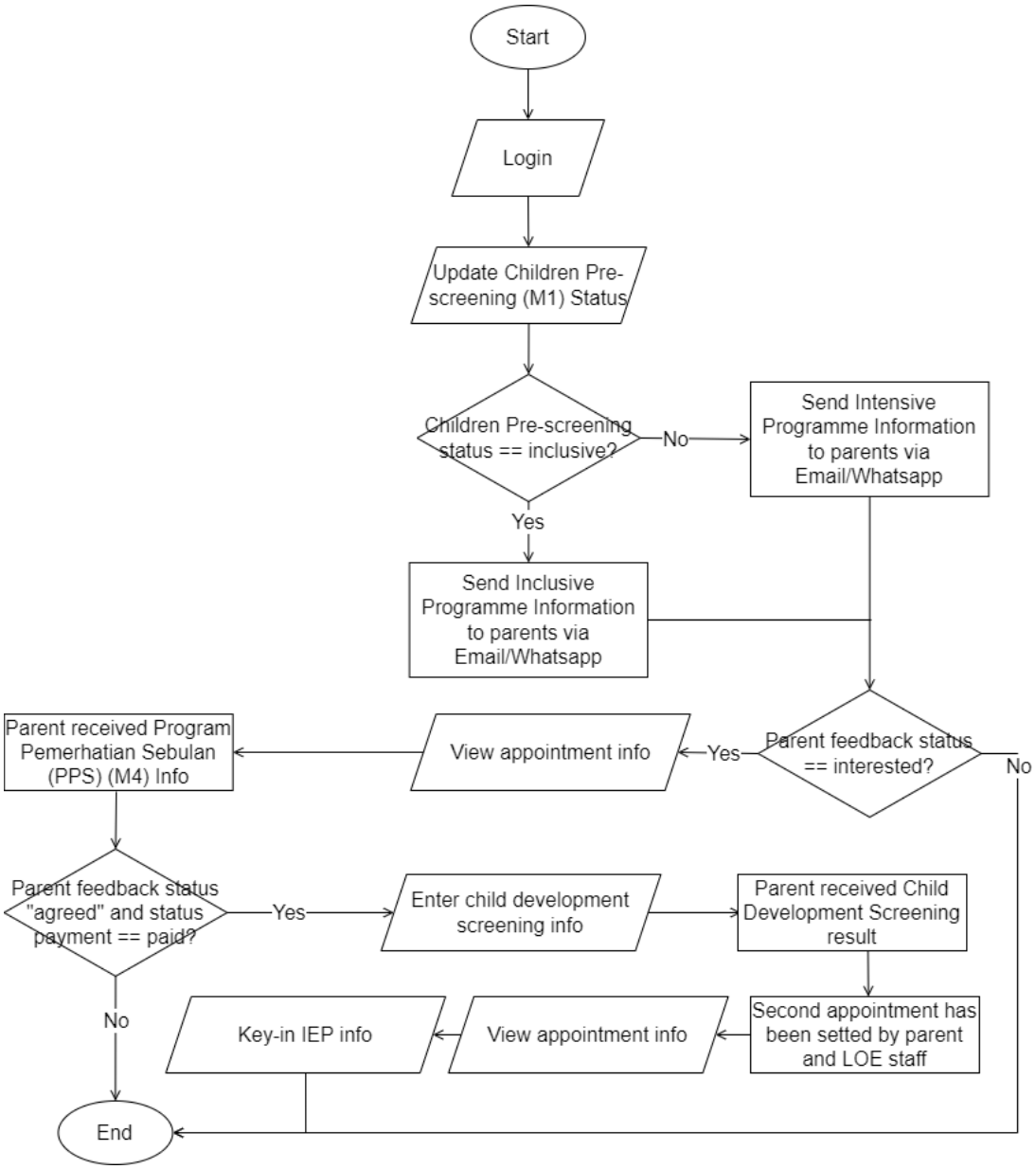


Figure 3.4.3: Flowchart for OT

3.4.1.4 Flowchart for user Consultant

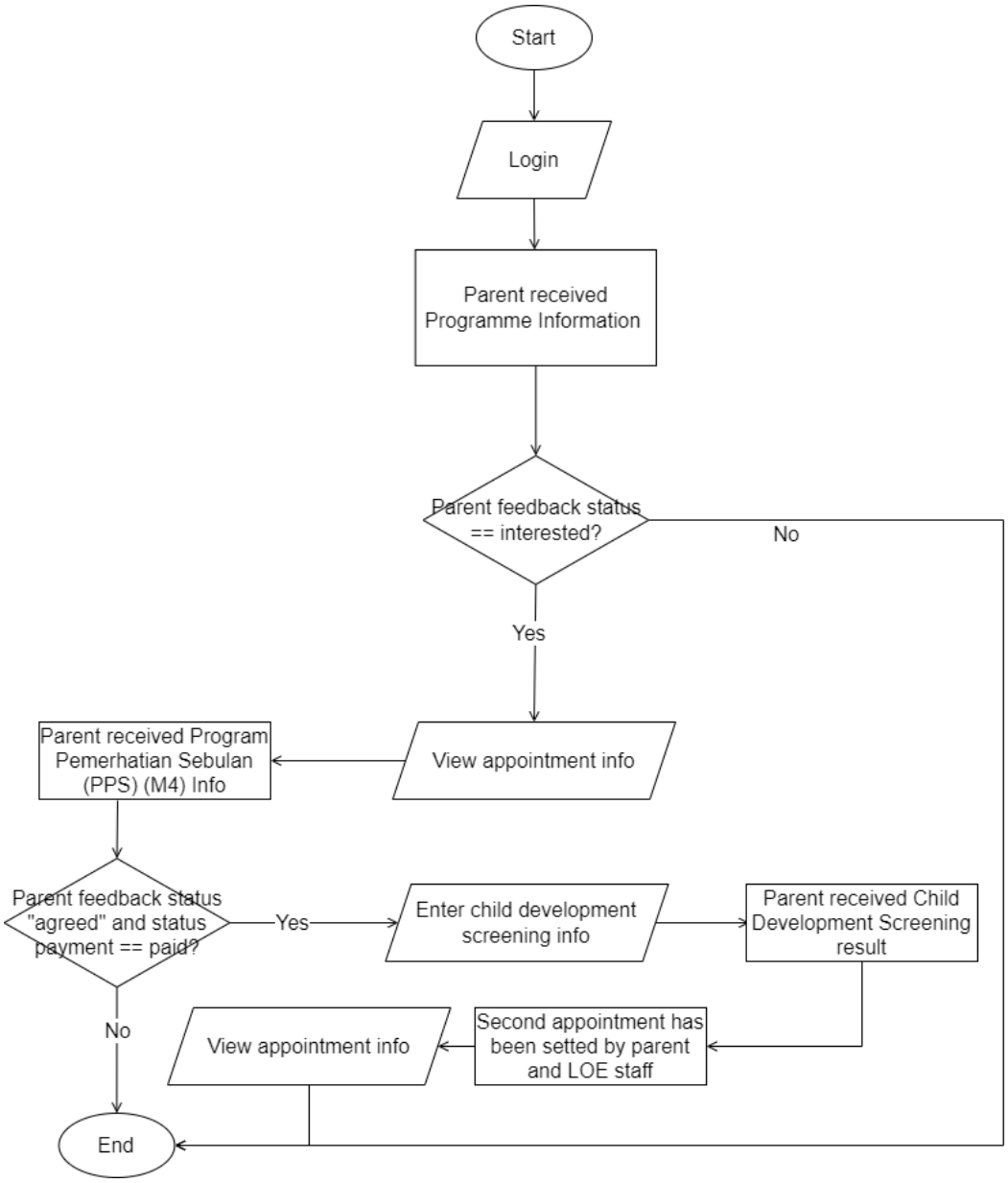


Figure 3.4.4: Flowchart for Consultant

3.4.1.5 Flowchart for user Admin

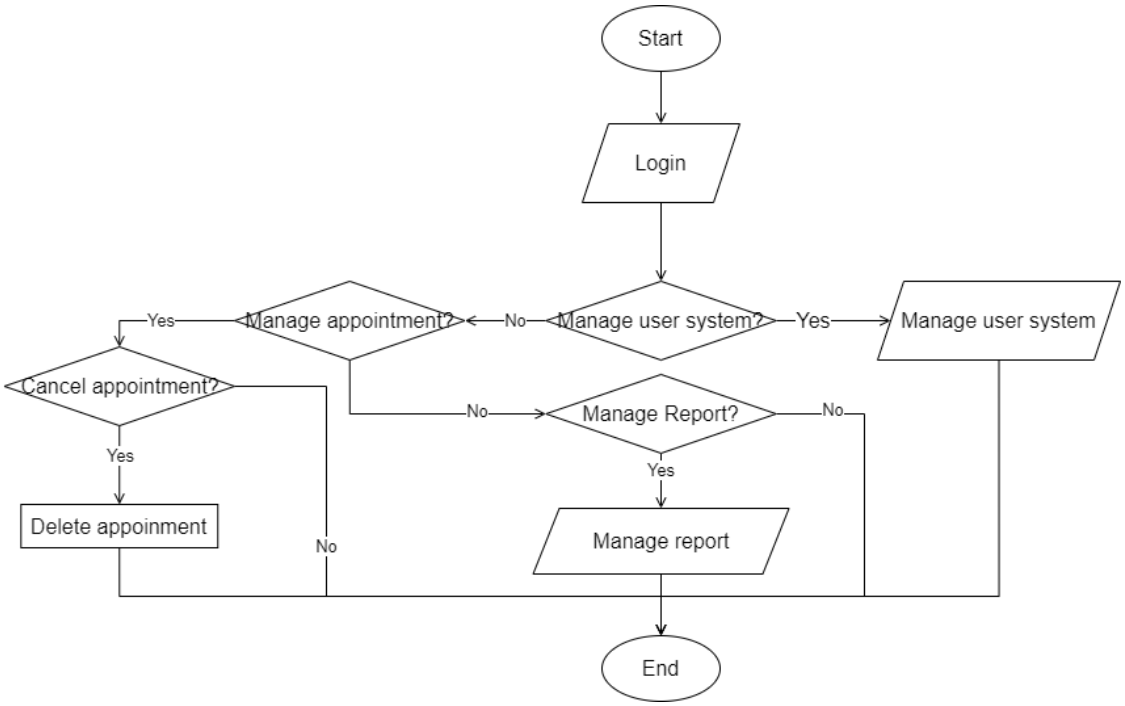


Figure 3.4.5: Flowchart for Admin

3.4.2 Context Diagram

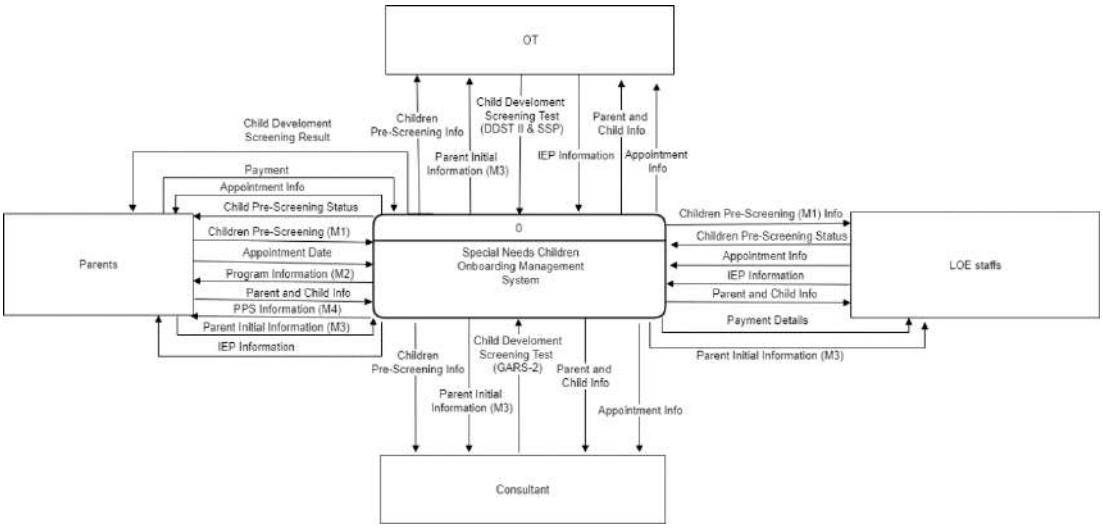


Figure 3.4.6: Context diagram

3.4.3 Use Case Diagram and Description

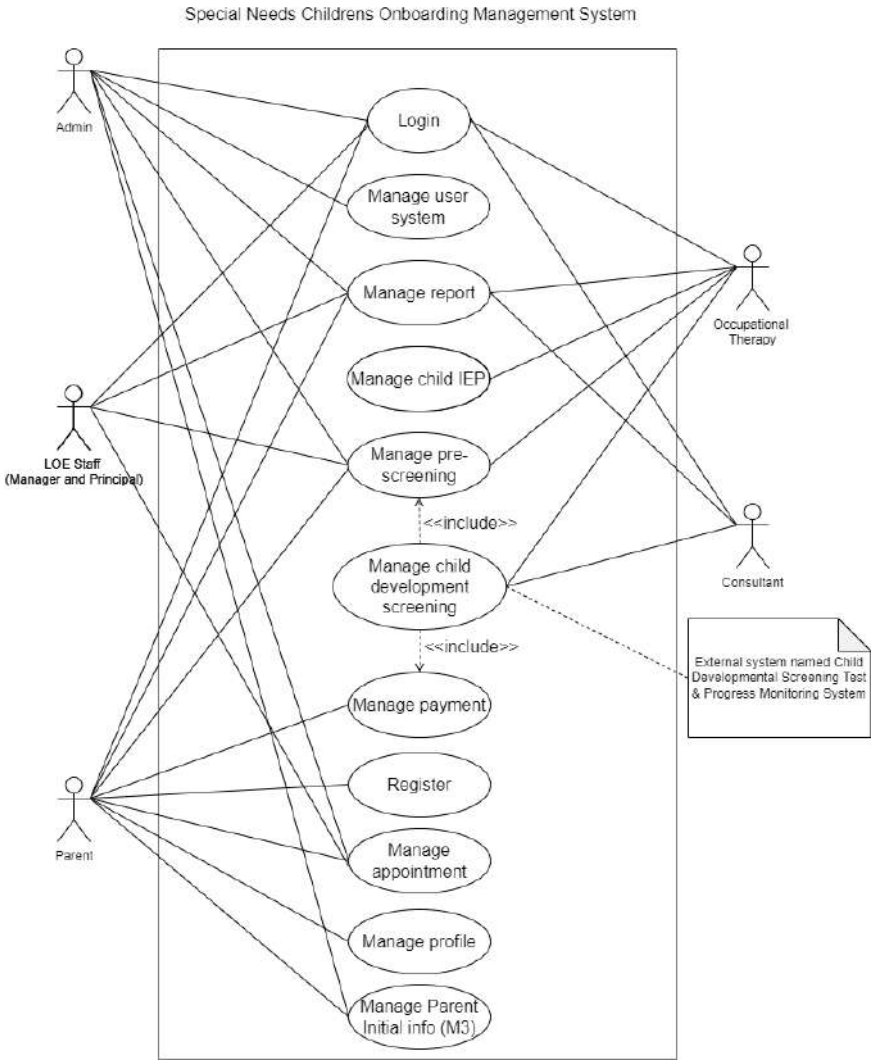


Figure 3.4.7: Use case diagram

3.4.3.1 Login (UC0001)

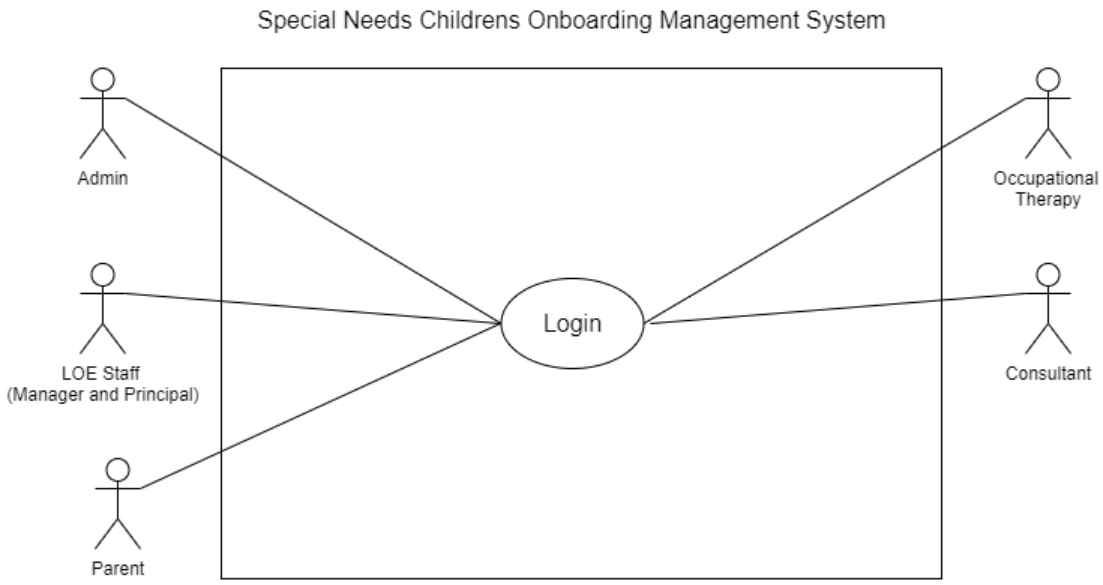


Figure 3.4.8: Use case diagram Login (UC0001)

Table 3.4.1: Use case description Login (UC0001)

Use case ID	UC0001
Use case Name	Login
Brief Description	This use case is used for users to login into the system using their registered account.
Actor	Admin, LOE Staff, Parent, OT, Consultant
Pre-Conditions	The user must be registered into the system first.
Basic Flow	1. The use case begins when the users open the Login page. 2. User key-in email and password.

	3. User clicks on “Login” button. 4. System validates the user authentication. **[E1: Invalid input]** 5. System display user’s dashboard page. 6. The use case ends.
Alternative Flow	None
Exception Flow	**E1: Invalid input** 1. System displays an error message 2. User clicks on “OK” button. 3. The use case continues with steps 6 in basic flow.
Post-Conditions	User able to log in to the system.

3.4.3.2 Manage User System (UC0002)

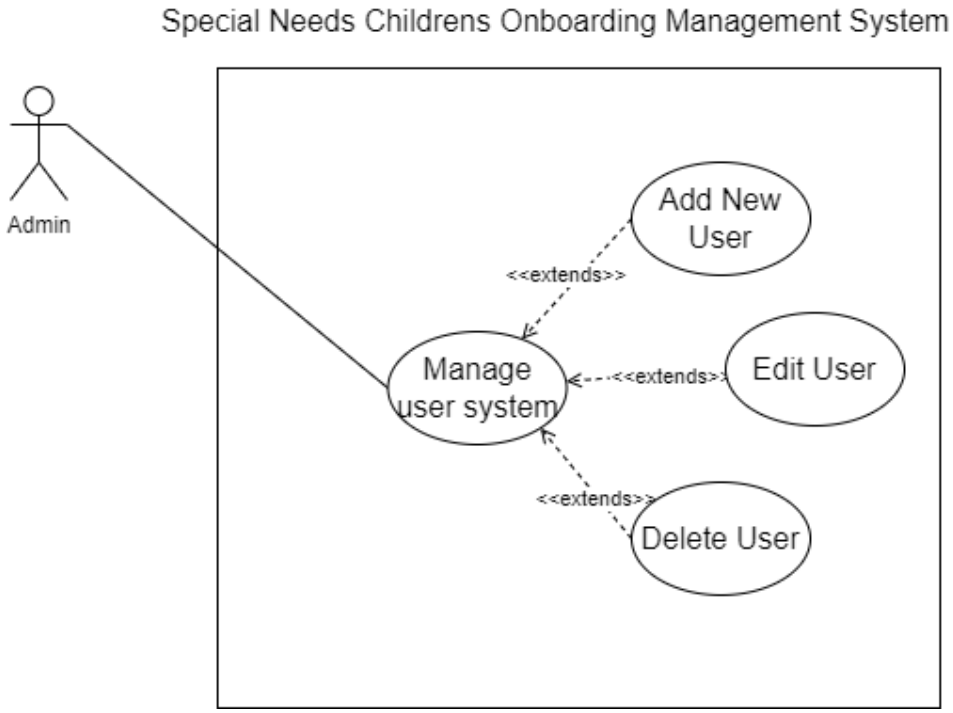


Figure 3.4.9: Use case diagram Manage User System (UC0002)

Table 3.4.2: Use case description Manage User System (UC0002)

Use case ID	UC0002
Use case Name	Manage user system
Brief Description	This use case is used to manage user system such as add user, delete user and update user info.
Actor	Admin
Pre-Conditions	The user must have logged in to system.
Basic Flow	1. The use case begins when the user clicks on “Manage User” menu.

	2. System displays Manage User System page and list of users. 3. User can choose following options: **[A1: Add new user]** **[A2: Edit user account]** **[A3: Delete user account]** 4. The use case ends.
Alternative Flow	**[A1: Add new user]** 1. User clicks on "Add new user" button. 2. System displays the page. 3. User key-in user's details. 4. User clicks on "Add" button. **[E1: Invalid input]** 5. System adds the data into the database. 6. System displays successful message. 7. The use case continues to step 4 in basic flow. **[A2: Edit user account]** 1. User clicks on "Edit" link. 2. System displays the page. 3. User key-in user's details. 4. User clicks on "Save" button. **[E2: Invalid input]** 5. System display confirmation box. 6. User clicks on "Yes" button. 7. System updates the data in the database. 8. System displays successful message.

	9. The use case continues to step 4 in basic flow. **[A3: Delete user account]** 1. User clicks on “Delete” link. 2. System displays confirmation box. 3. User clicks on “Yes” button. 4. System deletes the data in the database. 5. System displays successful message. 6. The use case continues to step 4 in basic flow.
Exception Flow	**E1: Invalid input** 1. System displays an error message 2. User clicks on “OK” button. 3. The use case continues with steps 4 in basic flow. **E2: Invalid input** 1. System displays an error message 2. User clicks on “OK” button. 3. The use case continues with steps 4 in basic flow.
Post-Conditions	Data updated in database.

3.4.3.3 Manage Report (UC0003)

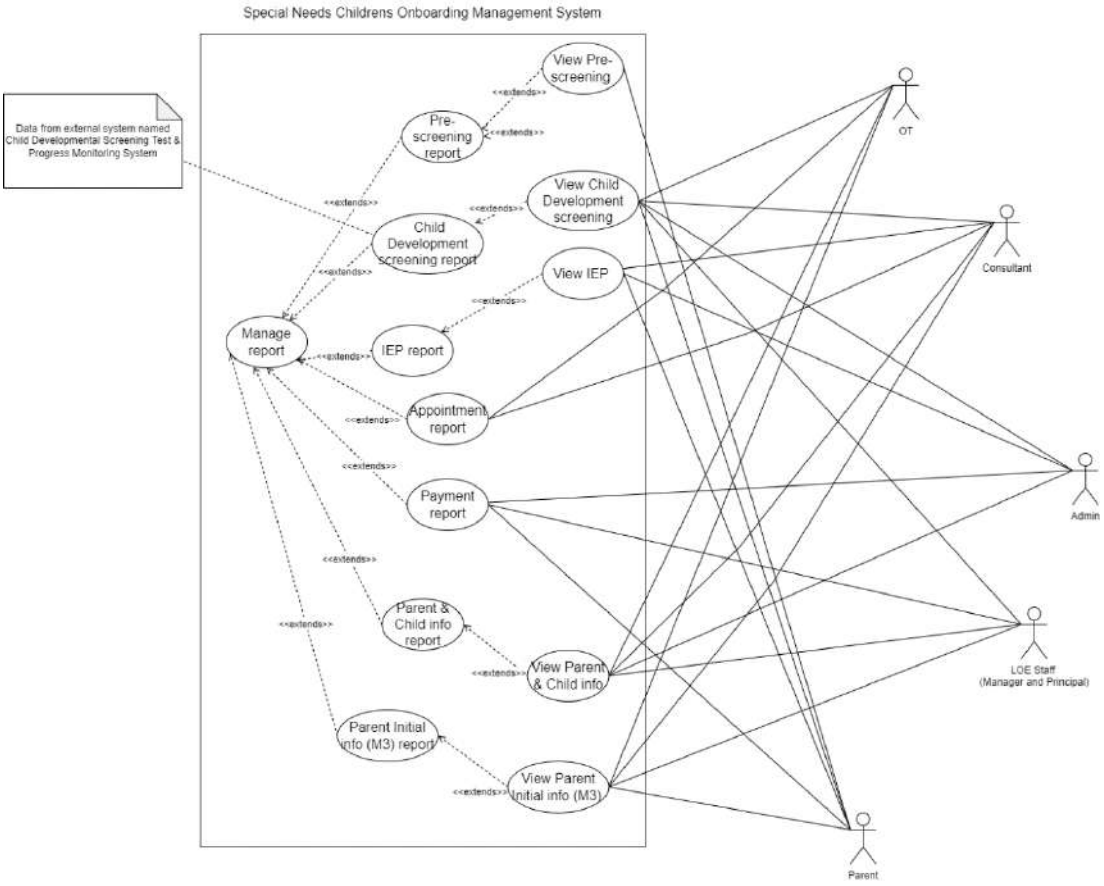


Figure 3.4.10: Use case diagram Manage Report (UC0003)

Table 3.4.3: Use case description Manage Report (UC0003)

Use case ID	UC0003
Use case Name	Manage report
Brief Description	This use case is used to manage and generate children report based on child's observation assessments and child development screenings results for the use of documentation and observation. Parent can view child pre-screening report.

	Users can view Child development screen result report. Admin, Consultant and Parents can view IEP report. OT and consultant can view appointment report. Admin, LOE staff and parent can view payment report. LOE staff, OT and consultant can view Parent & Child info report. Users can view Parent Initial info report.
Actor	Admin, LOE Staff, Parent, OT, Consultant
Pre-Conditions	• Users must have logged in to the system. • Parents must do a pre-screening test. • OT and Consultant must do child development assessments.
Basic Flow	**Admin:** 1. The use case begins when the user clicks on “Report” menu. 2. System displays Report page and list of report types.

	3. User can choose following options: **[A1: Child development screening report]** **[A2: IEP report]** **[A3: Payment report]** **[A4: Parent & Child info report]** 4. The use case ends. **LOE staff:** 1. The use case begins when the user clicks on "Report" menu. 2. System displays Report page and list of report types. 3. User can choose following options: **[A5: Child development screening report]** **[A6: Payment report]** **[A7: Parent & Child info report]** **[A8: Parent Initial info report (M3)]** 4. The use case ends. **Parent:** 1. The use case begins when the user clicks on "Report" menu. 2. System displays Report page and list of report types. 3. User can choose following options: **[A9: Pre-screening report]** **[A10: Child development screening report]**

	[A11: IEP report] **[A12: Payment report]** **[A13: Parent Initial info report (M3)]** 4. The use case ends. **OT:** 1. The use case begins when the user clicks on “Report” menu. 2. System displays Report page and list of report types. 3. User can choose following options: **[A14: Child development screening report]** **[A15: Appointment report]** **[A16: Parent & Child info report]** **[A17: Parent Initial info report (M3)]** 4. The use case ends. **Consultant:** 1. The use case begins when the user clicks on “Report” menu. 2. System displays Report page and list of report types. 3. User can choose following options: **[A18: Child development screening report]** **[A19: Appointment report]** **[A20: IEP report]** **[A21: Parent & Child info report]** **[A22: Parent Initial info report (M3)]**

	4. The use case ends.
Alternative Flow	**Admin:** **[A1: Child development screening report]** 1. User clicks on “Child Development Screening” menu button. 2. System displays the list child development screening report page. 3. User can choose following options: **[A23: View Child Development Screening details]** 4. The use case continues to step 4 in basic flow. **[A2: IEP report]** 1. User clicks on “Individualized Education Programs (IEP)” menu button. 2. System displays the list of IEP report page. 3. User can choose following options: **[A24: View IEP details]** 4. The use case continues to step 4 in basic flow.

	[A3: Payment report] 1. User clicks on “Payment Report” menu button. 2. System displays the list of payment report page. 3. The use case continues to step 4 in basic flow. **[A4: Parent & Child info report]** 1. User clicks on “Parent and Child Info” menu button. 2. System displays the list of parent and child info report page. 3. User can choose following options: **[A25: View Parent & Child info details]** 4. The use case continues to step 4 in basic flow. **LOE staff:** **[A5: Child development screening report]** 1. User clicks on “Child Development Screening” menu button. 2. System displays the list child development screening report page. 3. User can choose following options:

	[A26: View Child Development Screening details] 4. The use case continues to step 4 in basic flow. **[A6: Payment report]** 1. User clicks on “Payment Report” menu button. 2. System displays the list of payment report page. 3. The use case continues to step 4 in basic flow. **[A7: Parent & Child info report]** 1. User clicks on “Parent and Child Info” menu button. 2. System displays the list of parent and child info report page. 3. User can choose following options: **[A27: View Parent & Child info details]** 4. The use case continues to step 4 in basic flow. **[A8: Parent Initial info report (M3)]** 1. User clicks on “Parent Initial Info (M3)” menu button.

	2. System displays the list of parent initial info (M3) report page. 3. User can choose following options: **[A28: View Parent Initial info (M3) details]** 4. The use case continues to step 4 in basic flow. **Parent:** **[A9: Pre-screening report]** 1. User clicks on “Pre-screening” menu button. 2. System displays the list of pre-screening report page. 3. User can choose following options: **[A29: View Pre-Screening details]** 4. The use case continues to step 4 in basic flow. **[A10: Child development screening report]** 1. User clicks on “Child Development Screening” menu button. 2. System displays the list child development screening report page. 3. User can choose following options: **[A30: View Child Development Screening details]**

	4. The use case continues to step 4 in basic flow. **[A11: IEP report]** 1. User clicks on “Individualized Education Programs (IEP)” menu button. 2. System displays the list of IEP report page. 3. User can choose following options: **[A31: View IEP details]** 4. The use case continues to step 4 in basic flow. **[A12: Payment report]** 1. User clicks on “Payment Report” menu button. 2. System displays the list of payment report page. 3. The use case continues to step 4 in basic flow. **[A13: Parent Initial info report (M3)]** 1. User clicks on “Parent Initial Info (M3)” menu button. 2. System displays the list of parent initial info (M3) report page. 3. User can choose following options: **[A32: View Parent Initial info (M3) details]** 4. The use case continues to step 4 in basic flow.

	OT: ***Child Developmental Screening Test & Progress Monitoring System [External system]*** **[A14: Child development screening report]** 1. User clicks on “Child Development Screening” menu button. 2. System displays the list child development screening report page. 3. User can choose following options: **[A33: View Child Development Screening details]** 4. The use case continues to step 4 in basic flow. **[A15: Appointment report]** 1. User clicks on “Appointment Report” menu button. 2. System displays the list of appointment report page. 3. The use case continues to step 4 in basic flow. **[A16: Parent & Child info report]** 1. User clicks on “Parent and Child Info” menu button. 2. System displays the list of parent and child info report page.

	3. User can choose following options: **[A34: View Parent & Child info details]** 4. The use case continues to step 4 in basic flow. **[A17: Parent Initial info report (M3)]** 1. User clicks on “Parent Initial Info (M3)” menu button. 2. System displays the list of parent initial info (M3) report page. 3. User can choose following options: **[A35: View Parent Initial info (M3) details]** 4. The use case continues to step 4 in basic flow. **Consultant:** ***Child Developmental Screening Test & Progress Monitoring System [External system]*** **[A18: Child development screening report]** 1. User clicks on “Child Development Screening” menu button. 2. System displays the list child development screening report page.

	3. User can choose following options: **[A36: View Child Development Screening details]** 4. The use case continues to step 4 in basic flow. **[A19: Appointment report]** 1. User clicks on “Appointment Report” menu button. 2. System displays the list of appointment report page. 3. The use case continues to step 4 in basic flow. **[A20: IEP report]** 1. User clicks on “Individualized Education Programs (IEP)” menu button. 2. System displays the list of IEP report page. 3. User can choose following options: **[A37: View IEP details]** 4. The use case continues to step 4 in basic flow. **[A21: Parent & Child info report]** 1. User clicks on “Parent and Child Info” menu button. 2. System displays the list of parent and child info report page. 3. User can choose following options:

	[A38: View Parent & Child info details] 4. The use case continues to step 4 in basic flow. **[A22: Parent Initial info report (M3)]** 1. User clicks on “Parent Initial Info (M3)” menu button. 2. System displays the list of parent initial info (M3) report page. 3. User can choose following options: **[A39: View Parent Initial info (M3) details]** 4. The use case continues to step 4 in basic flow. **Admin:** **[A23: View Child Development Screening details]** 1. User clicks on “View” link. 2. System displays the child development screening details page. 3. The use case continues to step 4 in alternative flow. **[A24: View IEP details]** 1. User clicks on “View” link.

	2. System displays the IEP details page. 3. The use case continues to step 4 in alternative flow. **[A25: View Parent & Child info details]** 1. User clicks on “View” link. 2. System displays the parent & child info details page. 3. The use case continues to step 4 in alternative flow. **LOE staff:** **[A26: View Child Development Screening details]** 1. User clicks on “View” link. 2. System displays the child development screening details page. 3. The use case continues to step 4 in alternative flow. **[A27: View Parent & Child info details]** 1. User clicks on “View” link. 2. System displays the parent & child info details page. 3. The use case continues to step 4 in alternative flow.

	[A28: View Parent Initial info report (M3) details] 1. User clicks on “View” link. 2. System displays the parent initial info details page. 3. The use case continues to step 4 in alternative flow. **Parent:** **[A29: View Pre-Screening details]** 1. User clicks on “View” link. 2. System displays the pre-screening details page. 3. The use case continues to step 4 in alternative flow. **[A30: View Child Development Screening details]** 1. User clicks on “View” link. 2. System displays the child development screening details page. 3. The use case continues to step 4 in alternative flow. **[A31: View IEP details]** 1. User clicks on “View” link. 2. System displays the IEP details page.

	3. The use case continues to step 4 in alternative flow. **[A32: View Payment details]** 1. User clicks on “View” link. 2. System displays the payment details page. 3. The use case continues to step 4 in alternative flow. **[A33: View Parent Initial info (M3) details]** 1. User clicks on “View” link. 2. System displays the parent initial info details page. 3. The use case continues to step 4 in alternative flow. **OT:** ***Child Developmental Screening Test & Progress Monitoring System [External system]*** **[A34: View Child Development Screening details]** 1. User clicks on “View” link. 2. System displays the child development screening details page. 3. The use case continues to step 4 in alternative flow.

	[A35: View Parent & Child info details] 1. User clicks on “View” link. 2. System displays the parent & child info details page. 3. The use case continues to step 4 in alternative flow. **[A36: View Parent Initial info (M3) details]** 1. User clicks on “View” link. 2. System displays the parent initial info details page. 3. The use case continues to step 4 in alternative flow. **Consultant:** ***Child Developmental Screening Test & Progress Monitoring System [External system]*** **[A37: View Child Development Screening details]** 1. User clicks on “View” link. 2. System displays the child development screening details page. 3. The use case continues to step 4 in alternative flow. **[A38: View IEP details]** 1. User clicks on “View” link.

	2. System displays the IEP details page. 3. The use case continues to step 4 in alternative flow. **[A39: View Parent & Child info details]** 1. User clicks on “View” link. 2. System displays the parent & child info details page. 3. The use case continues to step 4 in alternative flow.
Exception Flow	-
Post-Conditions	Data retrieved from database.

3.4.3.4 Manage IEP (UC0004)

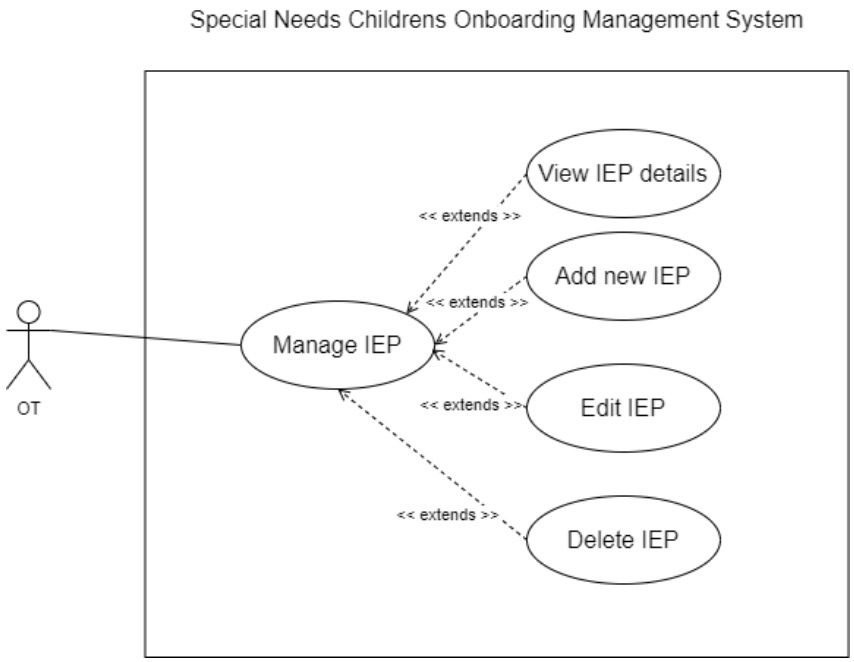


Figure 3.4.11: Use case diagram Manage IEP (UC0004)

Table 3.4.4: Use case description Manage IEP (UC0004)

Use case ID	UC0004
Use case Name	Manage child IEP
Brief Description	This use case is used to manage children IEP
Actor	OT
Pre-Conditions	Parent and child have completed child development screening phase
Basic Flow	1. The use case begins when the user clicks on “Child IEP” menu. 2. System displays Child IEP page.

	3. User can choose following options: **[A1: Add New IEP]** **[A2: View IEP]** **[A3: Edit IEP]** **[A4: Delete IEP]** 4. The use case ends.
Alternative Flow	**[A1: Add New IEP]** 1. User clicks on "Add New IEP" button. 2. System displays the page. 3. User key-in details. 4. User clicks on "Add" button. **[E1: Invalid Input]** 5. System adds data into the database. 6. The use case continues to step 4 in basic flow. **[A2: View IEP]** 1. User clicks on "View" link. 2. System displays the IEP details page. 3. The use case continues to step 4 in basic flow. **[A3: Edit IEP]** 1. User clicks on "Edit" link. 2. System displays the page. 3. User key-in details. 4. User clicks on "Save" button. **[E2: Invalid Input]**

	5. System display confirmation box. 6. User clicks on “Yes” button. 7. System updates data in the database. 8. The use case continues to step 4 in basic flow. **[A4: Delete IEP]** 1. User clicks on “Delete” link. 2. System displays the confirmation message. 3. User clicks on “Yes” button. 4. System deletes data in the database. 5. The use case continues to step 4 in basic flow.
Exception Flow	**E1: Invalid input** 1. System displays an error message. 2. User clicks on “OK” button. 3. The use case continues with steps 6 in alternative flow. **E2: Invalid input** 1. System displays an error message. 2. User clicks on “OK” button. 3. The use case continues with steps 8 in alternative flow.
Post-Conditions	Data updated in database.

3.4.3.5 Manage Pre-screening (UC0005)

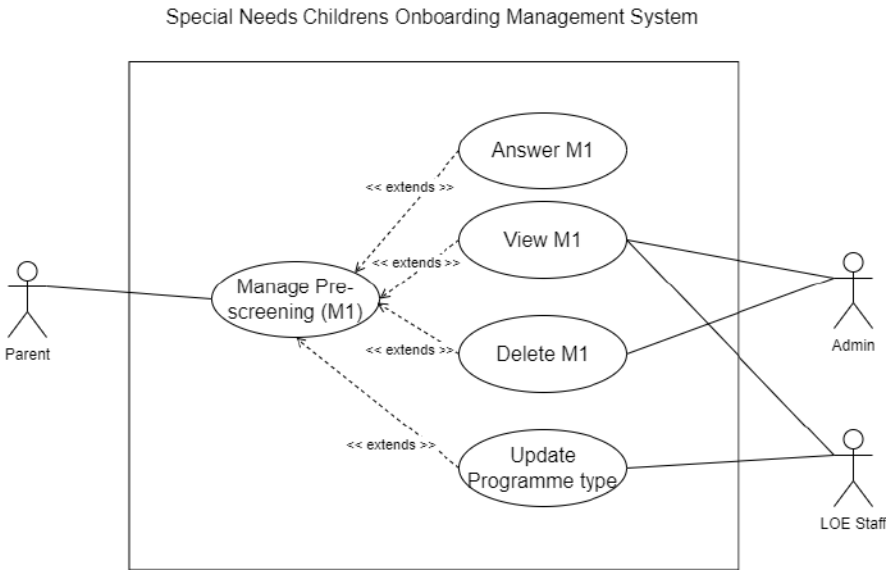


Figure 3.4.12: Use case diagram Manage Pre-screening (UC0005)

Table 3.4.5: Use case description Manage Pre-screening (UC0005)

Use case ID	UC0005
Use case Name	Manage pre-screening
Brief Description	This use case is for Parent to answer the pre-screening questionnaire. LOE staff, and OT can view the answered questionnaire and update child programme type Admin can view and delete M1 data if parent is request to re-submit M1.
Actor	Admin, Parent, LOE staff, OT

Pre-Conditions	• The user must have logged into the system. • Parent must answer the questionnaire.
Basic Flow	**Parent:** 1. The use case begins when the user clicks on “Pre-screening” menu. 2. System displays Pre-screening page. 3. User key-in the answer. 4. User tick on the “I acknowledge that all my answers are accurate, and all information provided is private and confidential.” checkbox. 5. User clicks on “Submit” button to submit the form**. [E1: Invalid Input] [E2: Untick Checkbox]** 6. System adds data into the database. 7. System displays successful message and notifies LOE staff and OT. 8. The use case ends. **LOE staff/OT:** 1. The use case begins when the user clicks on “Pre-screening” menu. 2. System displays the list of answered pre-screening. 3. Users clicks on “Update” link. 4. System displays the page. 5. Users choose programme type. 6. Users click on “Update” button.

	7. System updates data into database. 8. System displays successful message and notifies parent. 9. Use case end. **Admin** 1. The use case begins when the user clicks on “Pre-screening” menu. 2. System displays the list of answered pre-screening. 3. Users can choose following options: **[A1: View M1]** **[A2: Delete M1]** 4. Use case end.
Alternative Flow	**[A1: View M1]** 1. The use case begins when the user clicks on “View” link. 2. System displays the answered pre-screening. 3. The use case continues to step 4 in basic flow. **[A2: Delete M1]** 1. The use case begins when the user clicks on “Delete” link. 2. The system displays confirmation box. 3. User clicks on “Yes” button. 4. System deletes the data. 5. The use case continues to step 4 in basic flow.

Exception Flow	**E1: Invalid input** 1. System displays an error message. 2. User clicks on “OK” button. 3. The use case continues with steps 8 in basic flow. **E2: Untick Checkbox** 1. System displays an error message. 2. User clicks on “OK” button. 3. The use case continues with steps 8 in basic flow.
Post-Conditions	Data added into the database.

3.4.3.6 Manage Payment (UC0006)

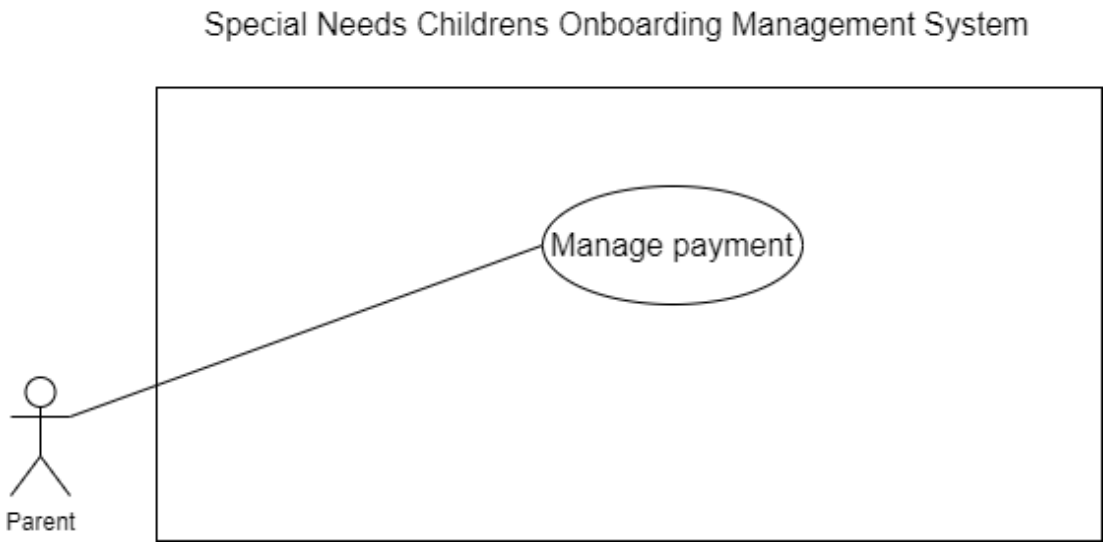


Figure 3.4.13: Use case diagram Manage Payment (UC0006)

Table 3.4.6: Use case description Manage Payment (UC0006)

Use case ID	UC0006
Use case Name	Manage payment
Brief Description	This use case is for parent to manage payment.
Actor	Parent
Pre-Conditions	• The users must have logged into the system. • The user must have bank account to make payment.
Basic Flow	1. The use case begins when the user clicks on "Payment" menu. 2. System displays Payment page.

	3. User chooses Fee Type, Payment Type and Bank Type. 4. User clicks on “Pay” button. 5. System displays successful message. 6. System updates data in the database. 7. System notifies LOE staff. 8. The use case ends.
Alternative Flow	-
Exception Flow	-
Post-Conditions	Data updated in database.

3.4.3.7 Register (UC0007)

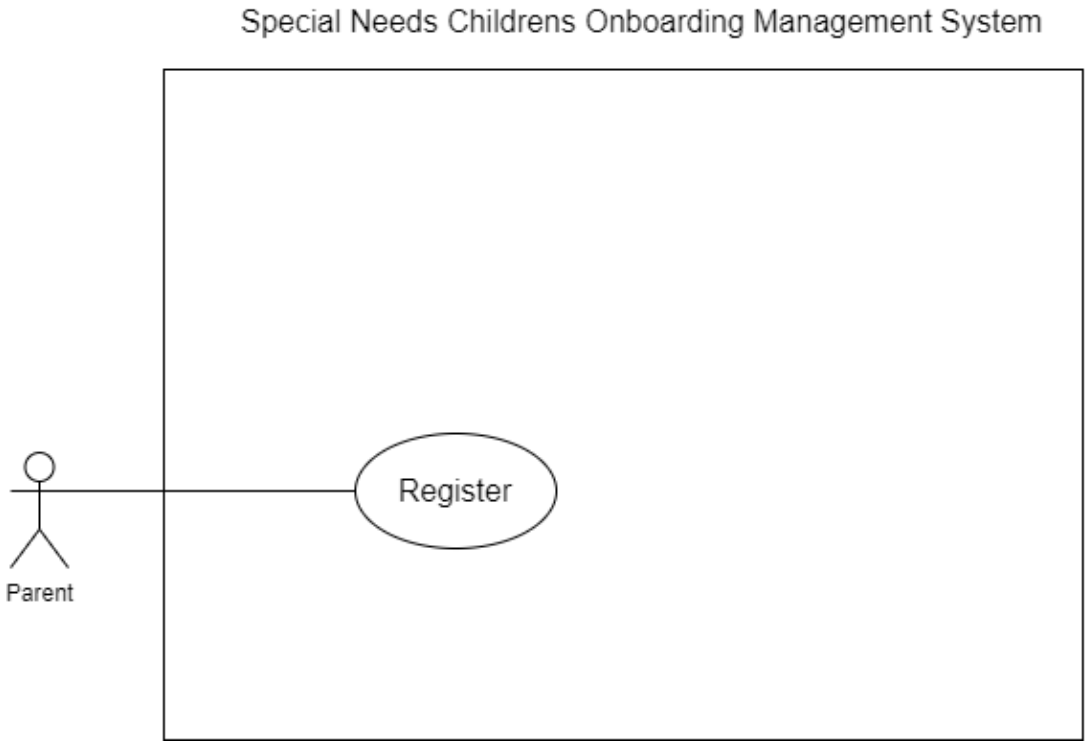


Figure 3.4.14: Use case diagram Register (UC0007)

Table 3.4.7: Use case description Register (UC0007)

Use case ID	UC0007
Use case Name	Register
Brief Description	This use case is used to create parent's account
Actor	Parent
Pre-Conditions	-
Basic Flow	1. The use case begins when the user clicks on "Sign up" button.

	2. System displays Sign up page. 3. User enters their information. 4. User enters email and password. 5. User clicks on “Create Account” button. **[E1: Invalid Input]** 6. System inserts data into the database. 7. System displays successful message. 8. The use case ends.
Alternative Flow	-
Exception Flow	**E1: Invalid input** 1. System displays an error message 2. User clicks on “OK” button. 3. The use case continues with steps 8 in basic flow.
Post-Conditions	Data updated in database.

3.4.3.8 Manage Appointment (UC0008)

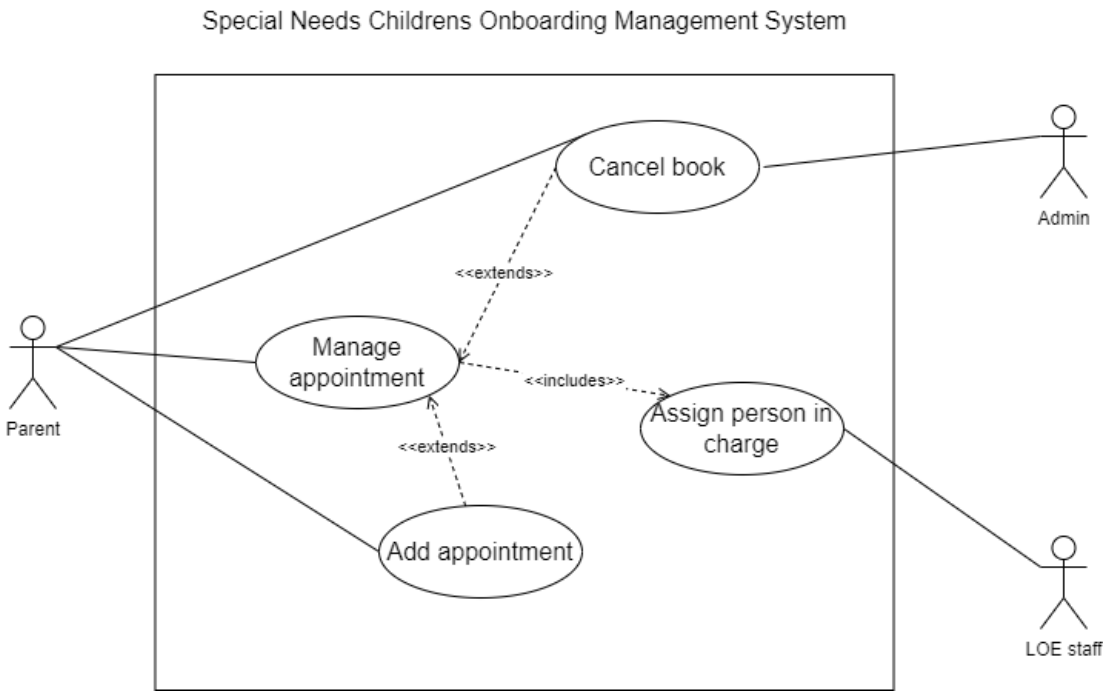


Figure 3.4.15: Use case diagram Manage Appointment (UC0008)

Table 3.4.8: Use case description Manage Appointment (UC0008)

Use case ID	UC0008
Use case Name	Manage appointment
Brief Description	This use case is for users to manage appointment. Parent can add and cancel appointment. Admin can cancel appointment if there is some urgent matter happens.

	LOE staff can assign person in charge for the appointment.
Actor	Parent, Admin, LOE staff
Pre-Conditions	• The user must have logged into the system • The user must have made a payment before booking for second appointment.
Basic Flow	**Parent:** 1. The use case begins when the user clicks on "Appointment" menu. 2. System displays Appointment page. 3. User can choose the following option: **[A1: Book Appointment]** **[A2: Cancel Appointment]** (If appointment have been made) 4. The use case ends. **Admin:** 1. The use case begins when the user clicks on "Appointment" menu. 2. System displays Appointment list page. 3. User can choose the following option: **[A3: Cancel Appointment]** (If appointment have been made by parent) 4. The use case ends.

	LOE staff: 1. The use case begins when the user clicks on “Appointment” menu. 2. System displays Appointment list page. 3. Users clicks on “View” link 4. System displays page 5. User assign person in charge by click on dropdown menu. 6. User clicks on “Update” button. 7. System updates data in database. 8. System notifies Parent and OT/Consultant in charge. 9. The use case ends.
Alternative Flow	**Parent:** **[A1: Book Appointment]** 1. User clicks on “Add Appointment” button. 2. System displays page. 3. User chooses Appointment Type. 4. User chooses available date and time slot. 5. User clicks on “Book” button. **[E1: No input data]** 6. System inserts data into the database. 7. System displays successful message. 8. System notify LOE staff, OT and Consultant.

	9. The use case continues to step 4 in basic flow. **[A2: Cancel Appointment]** 1. User clicks on “Cancel” link. 2. System displays confirmation message. 3. User clicks on “OK” button. 4. System deletes data in the database. 5. System displays successful message. 6. System notify LOE staff, OT and Consultant. 7. The use case continues to step 4 in basic flow. **Admin:** **[A3: Cancel Appointment]** 1. User clicks on “Cancel” link. 2. System displays confirmation message. 3. User clicks on “OK” button. 4. System deletes data in the database. 5. System displays successful message. 6. System notify LOE staff, OT and Consultant. 7. The use case continues to step 4 in basic flow.
Exception Flow	**[E1: No input data]**

	1. System displays an error message 2. The use case continues with step 9 in alternative flow.
Post-Conditions	Data updated in database.

3.4.3.9 Manage Profile (UC0009)

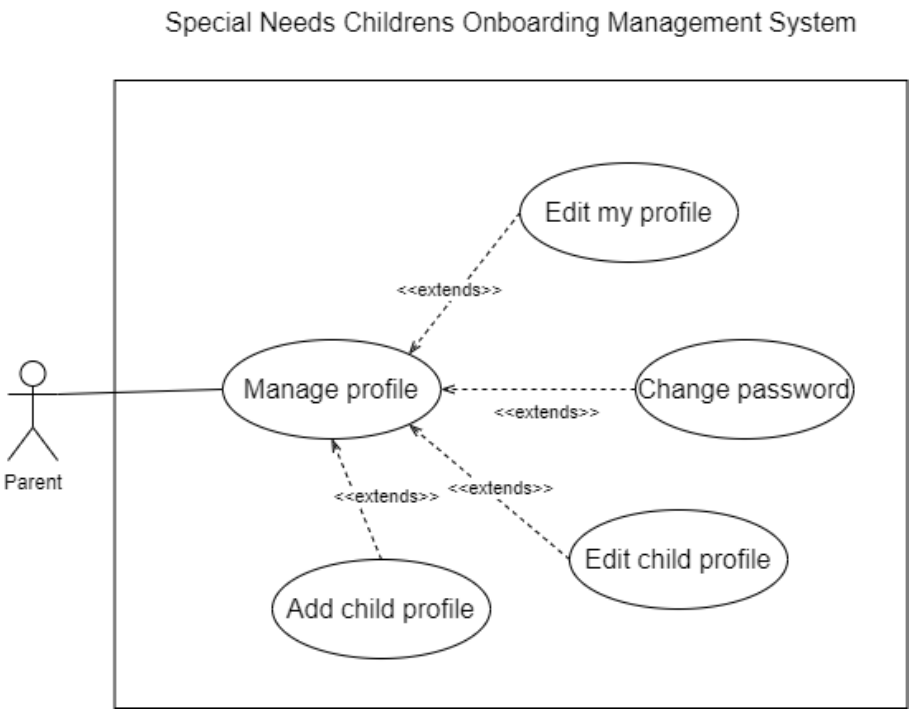


Figure 3.4.16: Use case diagram Manage Profile (UC0009)

Table 3.4.9: Use case description Manage Profile (UC0009)

Use case ID	UC0009
Use case Name	Manage profile
Brief Description	This use case is used to manage user profile
Actor	Parent
Pre-Conditions	Users logged in into the system
Basic Flow	1. The use case begins when the user clicks on “Manage Profile” menu. 2. System displays Profile page.

	3. User can choose following options: **[A1: Edit My Profile]** **[A2: Change Password]** **[A3: Children's Profile]** 4. The use case ends.
Alternative Flow	**[A1: Edit My Profile]** 1. User clicks on "Edit Profile" button. 2. System displays the page. 3. User key-in details. 4. User clicks on "Save" button. **[E1: Invalid Input]** 5. System displays confirmation message. 6. User clicks "OK" button. 7. System updates data in the database. 8. System displays successful message. 9. The use case continues to step 4 in basic flow. **[A2: Change Password]** 1. User clicks on "Change Password" menu. 2. System displays the page. 3. User key-in details. 4. User clicks on "Save" button. **[E2: Input Value Not Same]**

	5. System displays confirmation message. 6. User clicks “OK” button. 7. System updates data in the database. 8. System displays successful message. 9. The use case continues to step 4 in basic flow. **[A3: Children’s Profile]** 1. User clicks on “Children’s Profile” menu. 2. System displays the page. 3. User can choose following options: **[A4: Edit Children’s Profile]** **[A5: Add Children’s Profile]** 4. The use case continues to step 4 in basic flow. **[A4: Edit Children’s Profile]** 1. User clicks on “Edit Children’s Profile” button. 2. System displays the page. 3. User key-in details. 4. User clicks on “Update” button. **[E3: Invalid Input]** 5. System displays confirmation message. 6. User clicks “OK” button. 7. System updates data in the database.

	8. System displays successful message. 9. The use case continues to step 4 in alternative flow. **[A5: Add Children's Profile]** 1. User clicks on "Add Children's Profile" button. 2. System displays the page. 3. User key-in details. 4. User clicks on "Add" button. **[E4: Invalid Input]** 5. System add data in the database. 6. System displays successful message. 7. The use case continues to step 4 in alternative flow.
Exception Flow	**E1: Invalid input** 1. System displays an error message. 2. User clicks on "OK" button. 3. The use case continues with steps 9 in alternative flow. **E2: Input Value Not Same** 1. System displays an error message. 2. User clicks on "OK" button. 3. The use case continues with steps 9 in alternative flow.

	E3: Invalid input 1. System displays an error message. 2. User clicks on “OK” button. 3. The use case continues with steps 9 in alternative flow. **E4: Invalid input** 1. System displays an error message. 2. User clicks on “OK” button. 4. The use case continues with steps 7 in alternative flow.
Post-Conditions	Data updated in database.

3.4.3.10 Manage Parent Initial Info (M3) (UC0010)

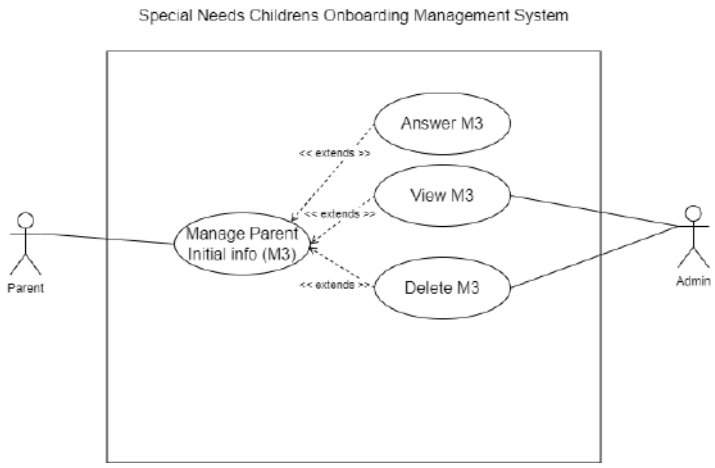


Figure 3.4.17: Use case diagram Manage Parent Initial Info (M3) (UC0010)

Table 3.4.10: Use case description Manage Parent Initial Info (M3) (UC0010)

Use case ID	UC0010
Use case Name	Manage Parent Initial Info (M3)
Brief Description	This use case is used for parent to answer the M3 questionnaire. Admin is able to view and delete the data if parent is requested to re-submit M3.
Actor	Parent and Admin
Pre-Conditions	• Users logged in into the system. • Users must book appointment first

Basic Flow	**Parent** 1. The use case begins when the user clicks on “Parent Initial Info” menu. 2. System displays page. 3. User key-in the answer. 4. User clicks on “Submit” button to submit the form**. [E1: Invalid Input]** 5. System adds data into the database. 6. System displays successful message. 7. The use case ends. **Admin** 1. The use case begins when the user clicks on “Parent Initial Info (M3)” menu. 2. System displays the list of M3. 3. Users can choose following options: **[A1: View M3]** **[A2: Delete M3]** 4. Use case end.
Alternative Flow	**[A1: View M3]** 1. The use case begins when the user clicks on “View” link. 2. System displays the M3 details. 3. The use case continues to step 4 in basic flow. **[A2: Delete M3]** 1. The use case begins when the user clicks on “Delete” link. 2. The system displays confirmation box.

	3. User clicks on “Yes” button. 4. System deletes the data. 5. The use case continues to step 4 in basic flow.
Exception Flow	**E1: Invalid input** 1. System displays an error message 2. User clicks on “OK” button. 3. The use case continues with steps 7 in alternative flow.
Post-Conditions	Data add in database.

3.4.4 Activity Diagram

3.4.4.1 Login

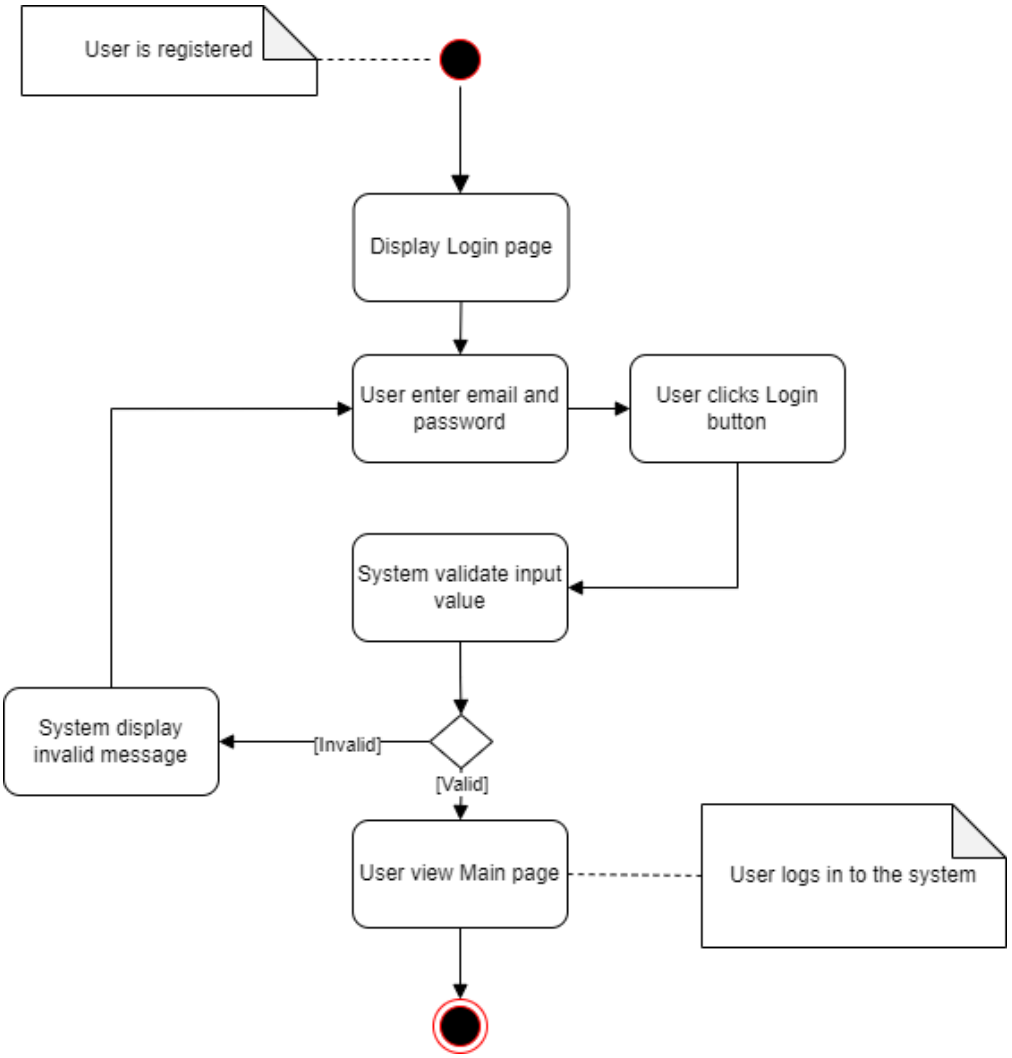


Figure 3.4.18: Activity diagram Login

3.4.4.2 Manage User System

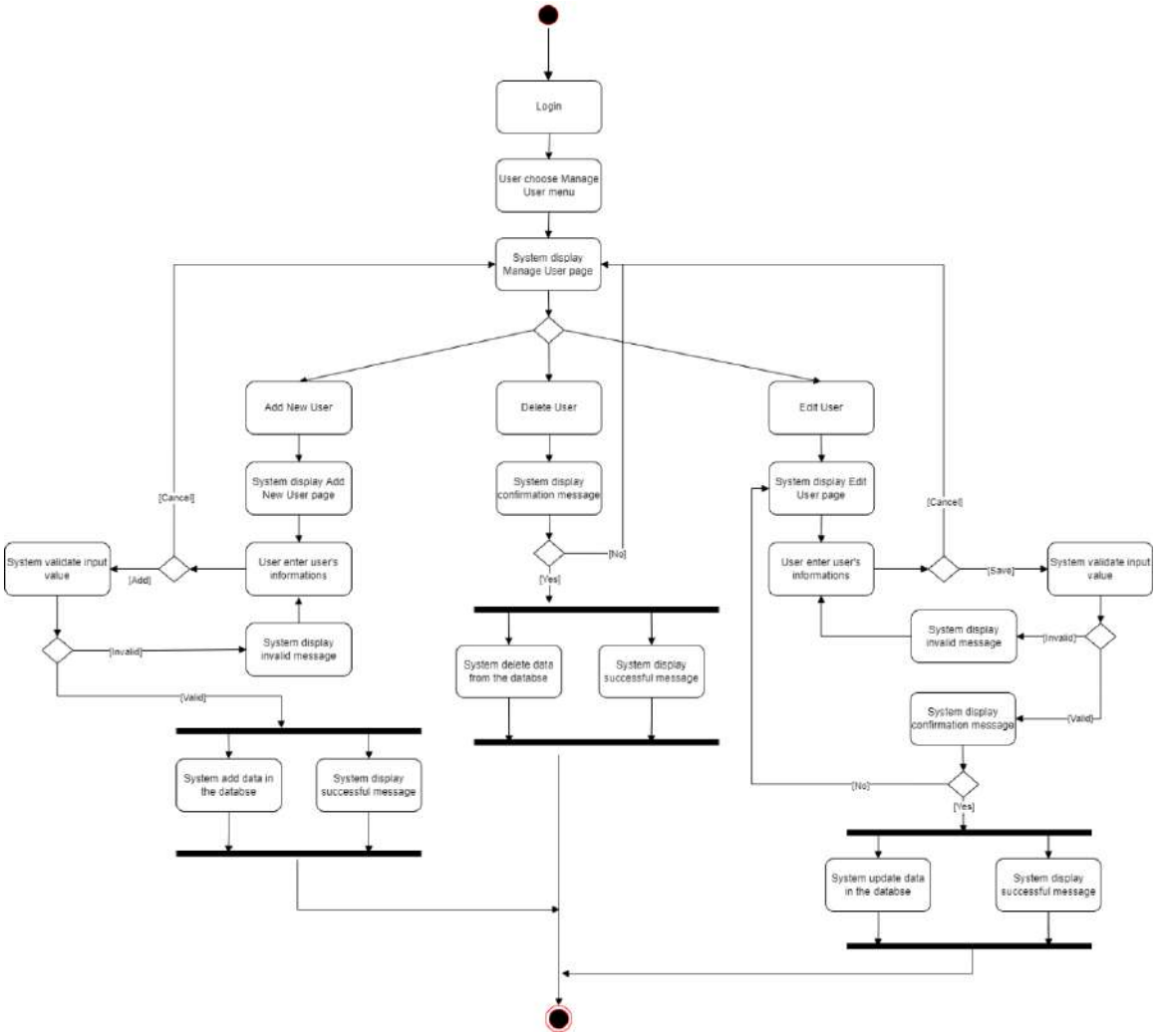


Figure 3.4.19: Activity diagram Manage User System

3.4.4.3 Manage Report

Admin

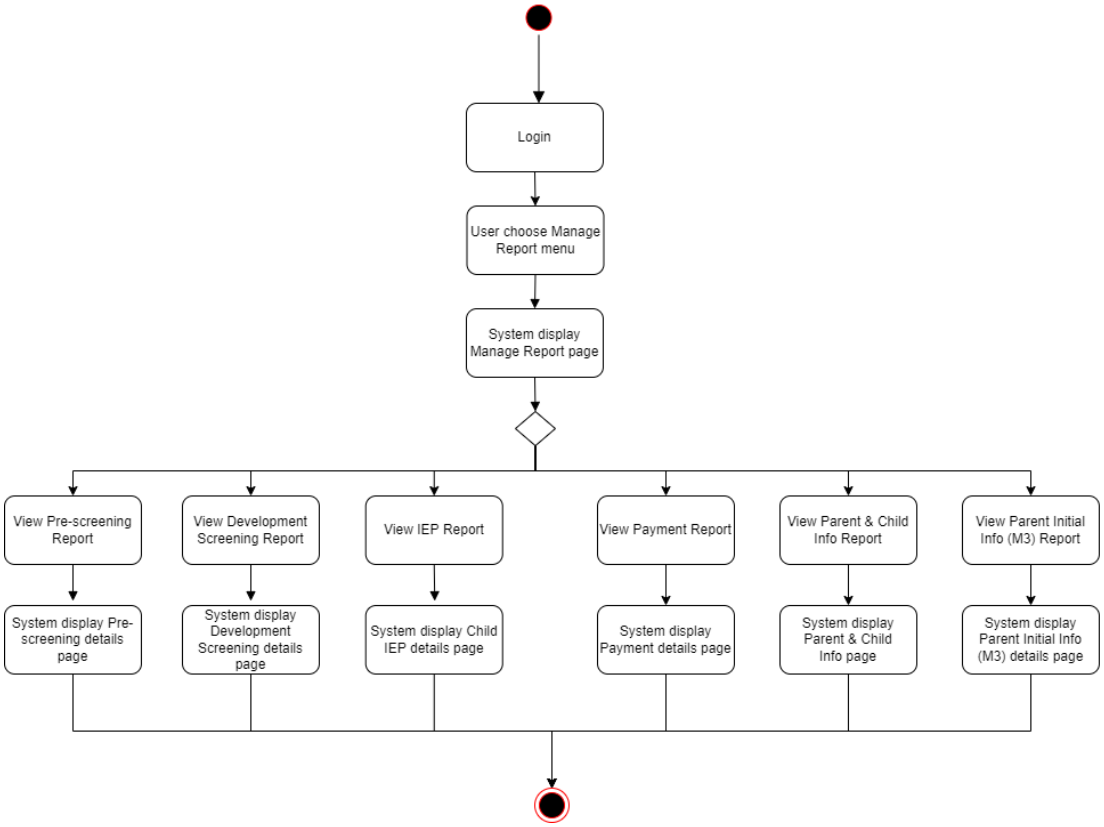


Figure 3.4.20: Activity diagram Manage Report (Admin)

LOE staff

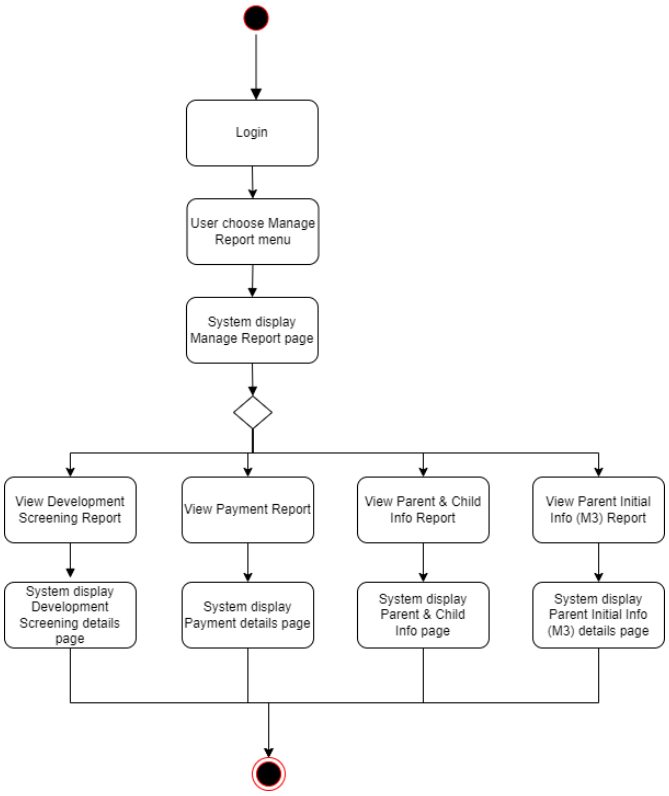


Figure 3.4.21: Activity diagram Manage Report (LOE staff)

Parent

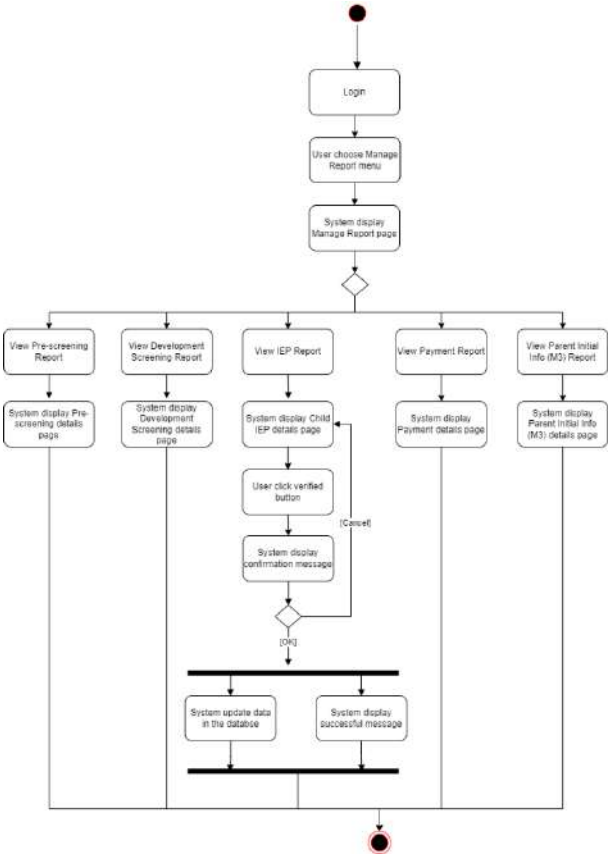


Figure 3.4.22: Activity diagram Manage Report (Parent)

OT

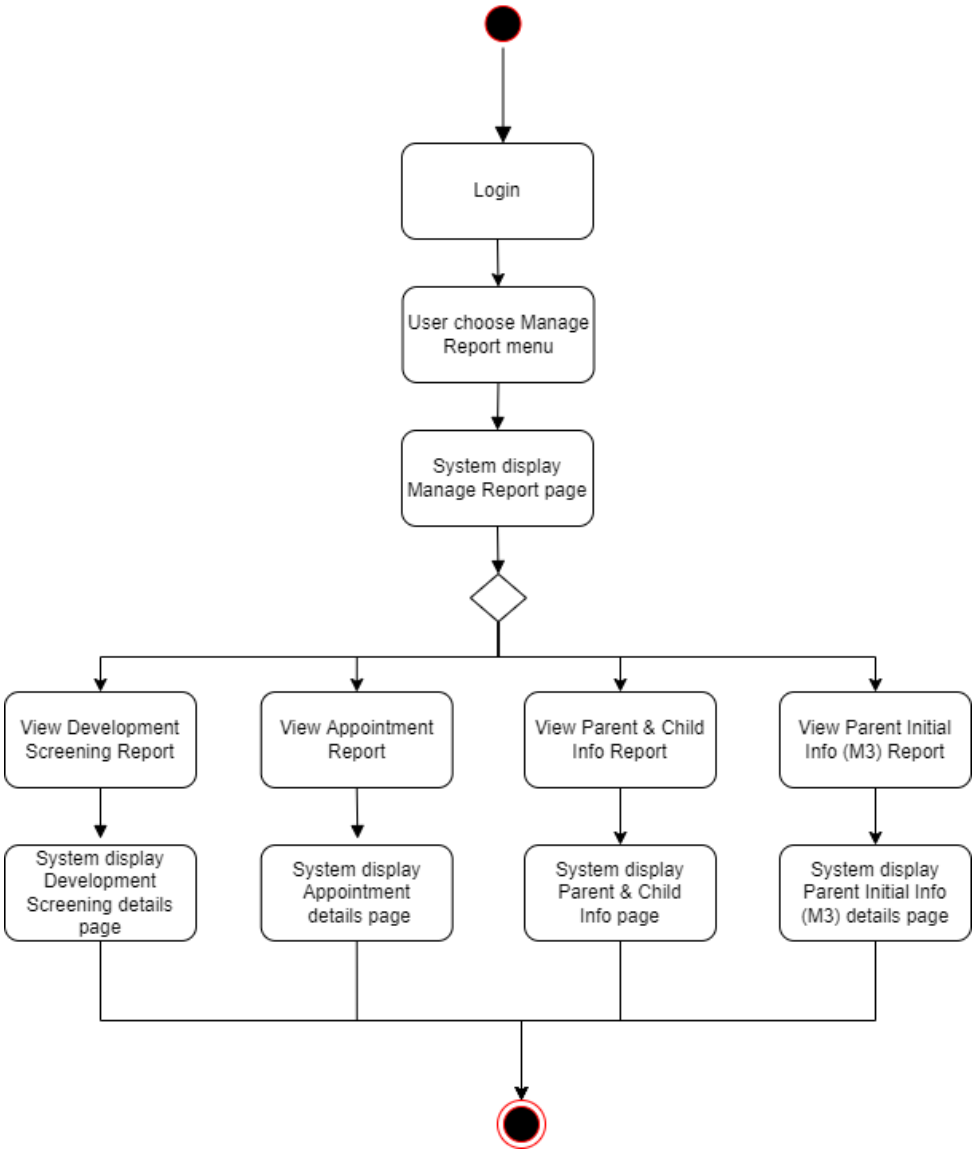


Figure 3.4.23: Activity diagram Manage Report (OT & Consultant)

Consultant

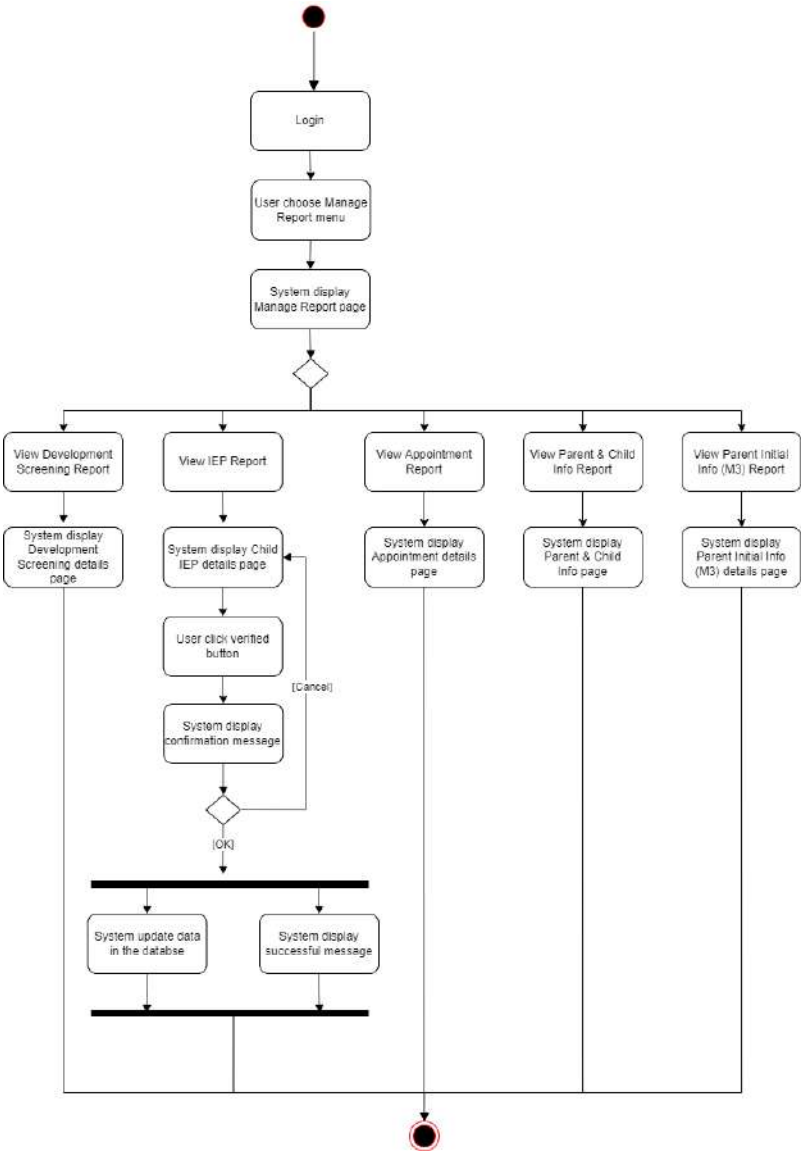


Figure 3.4.24: Activity diagram Manage Report (Consultant)

3.4.4.4 Manage Child IEP

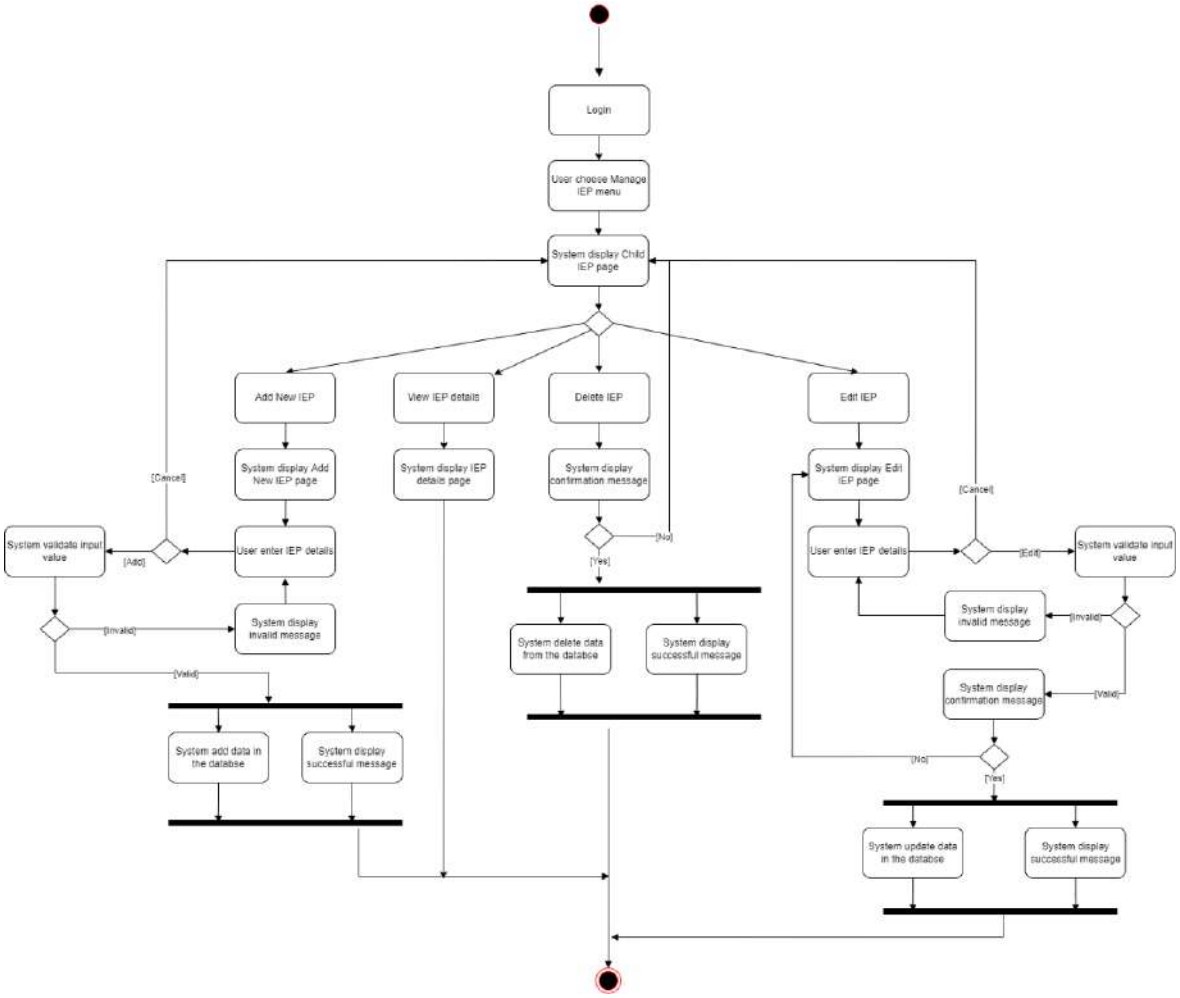


Figure 3.4.25: Activity diagram Manage IEP

3.4.4.5 Manage Pre-screening

Parent

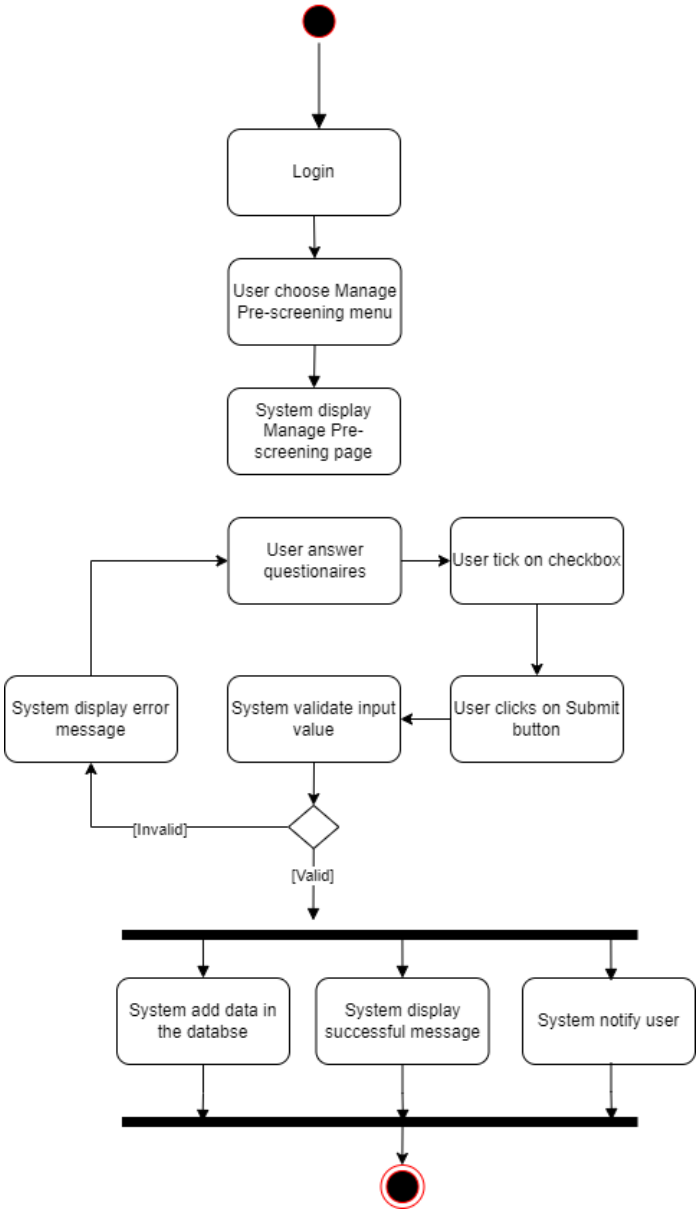


Figure 3.4.26: Activity diagram Manage Pre-screening (Parent)

LOE staff and OT

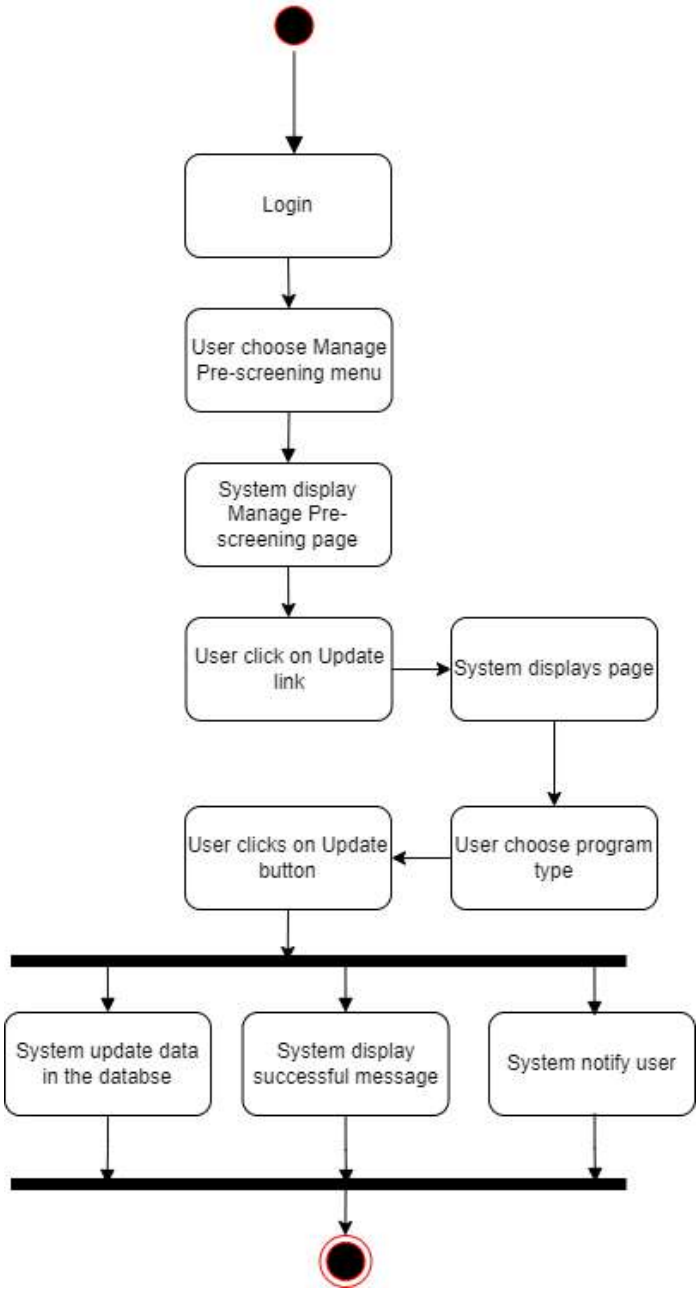


Figure 3.4.27: Activity diagram Manage Pre-screening (LOE staff and OT)

Admin

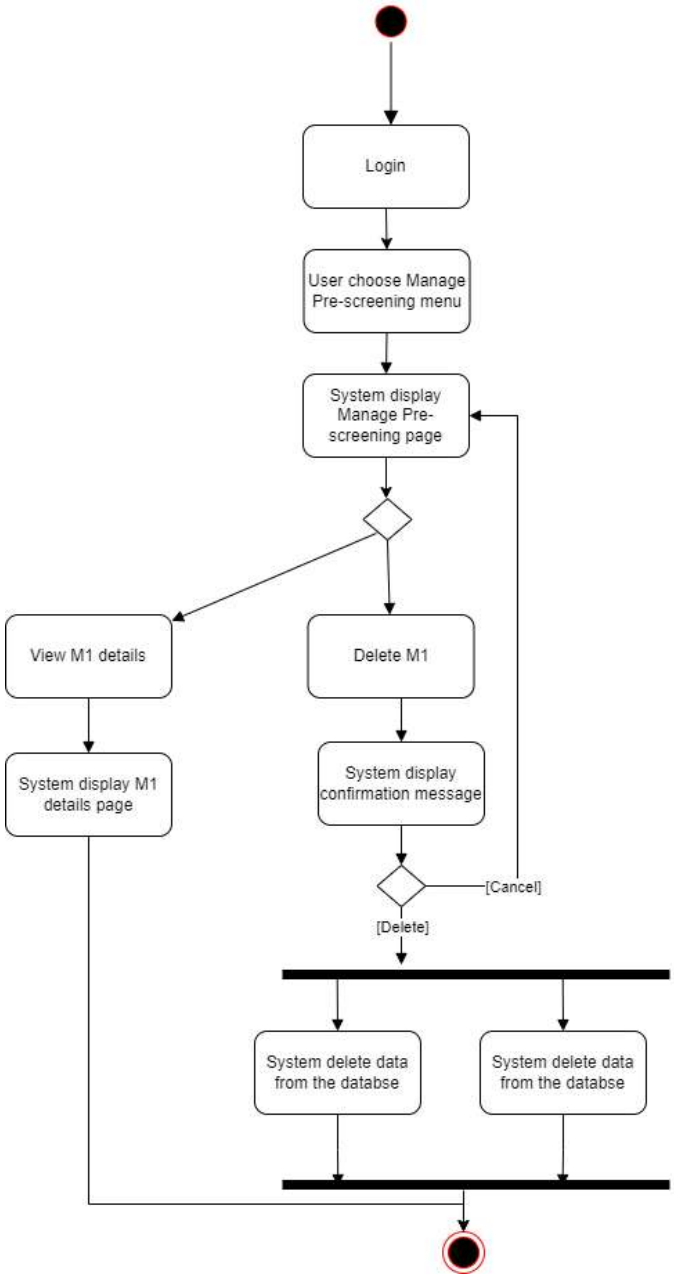


Figure 3.4.28: Activity diagram Manage Pre-screening (Admin)

3.4.4.6 Manage Parent Initial Info (M3)

Parent

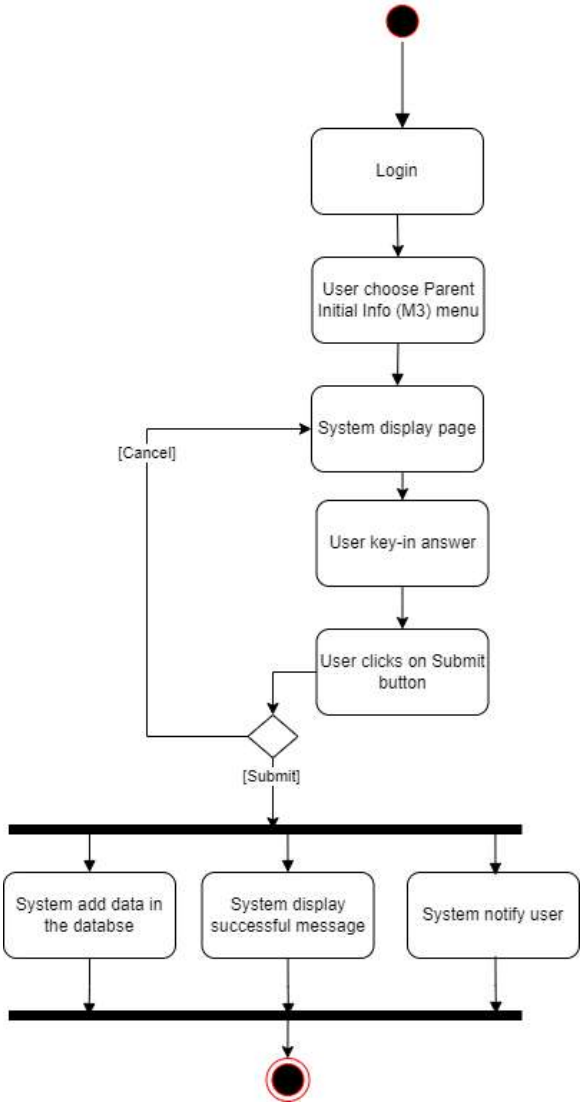


Figure 3.4.29: Activity diagram Manage Parent Initial Info (M3) (Parent)

Admin

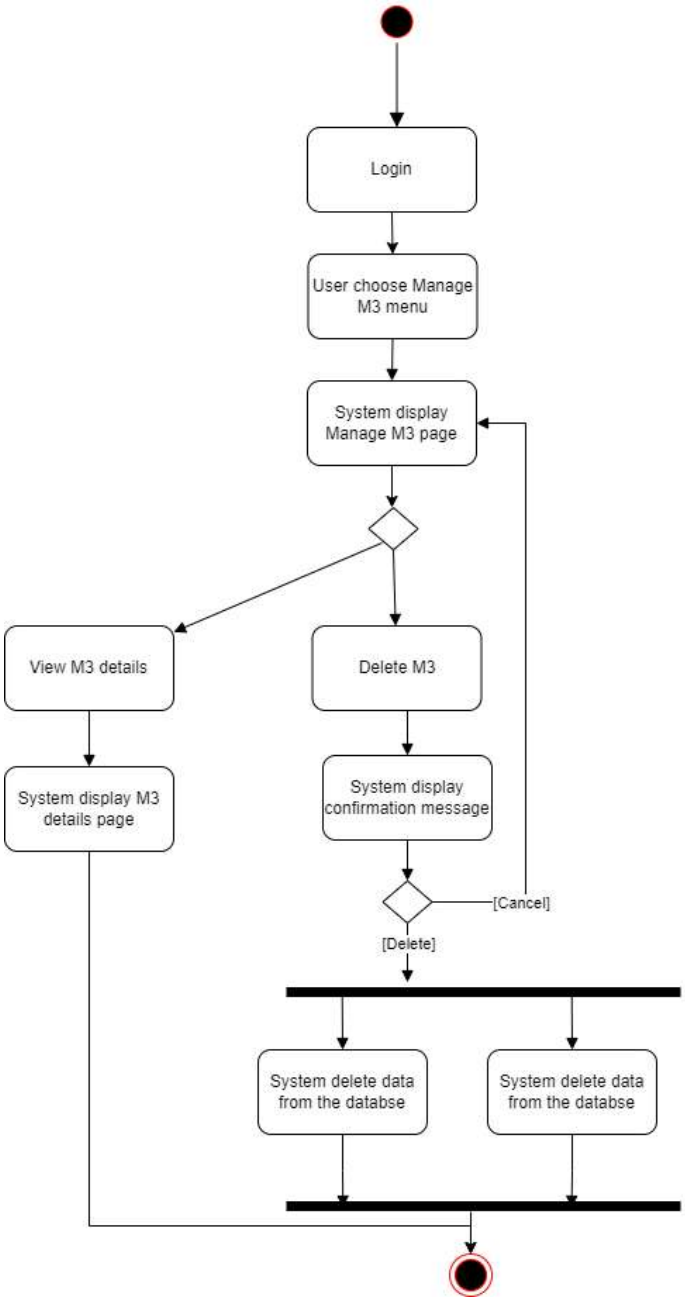


Figure 3.4.30: Activity diagram Manage Parent Initial Info (M3) (Admin)

3.4.4.7 Manage Payment

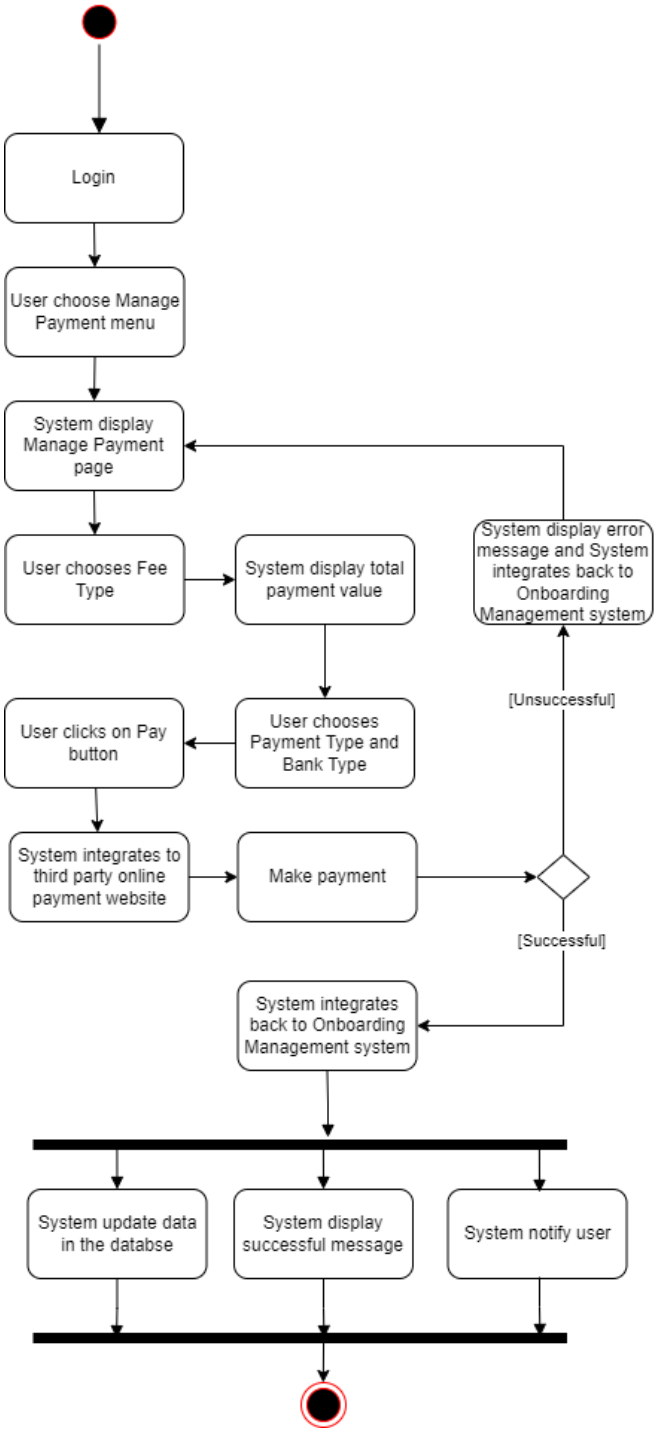


Figure 3.4.31: Activity diagram Mange Payment

3.4.4.8 Register

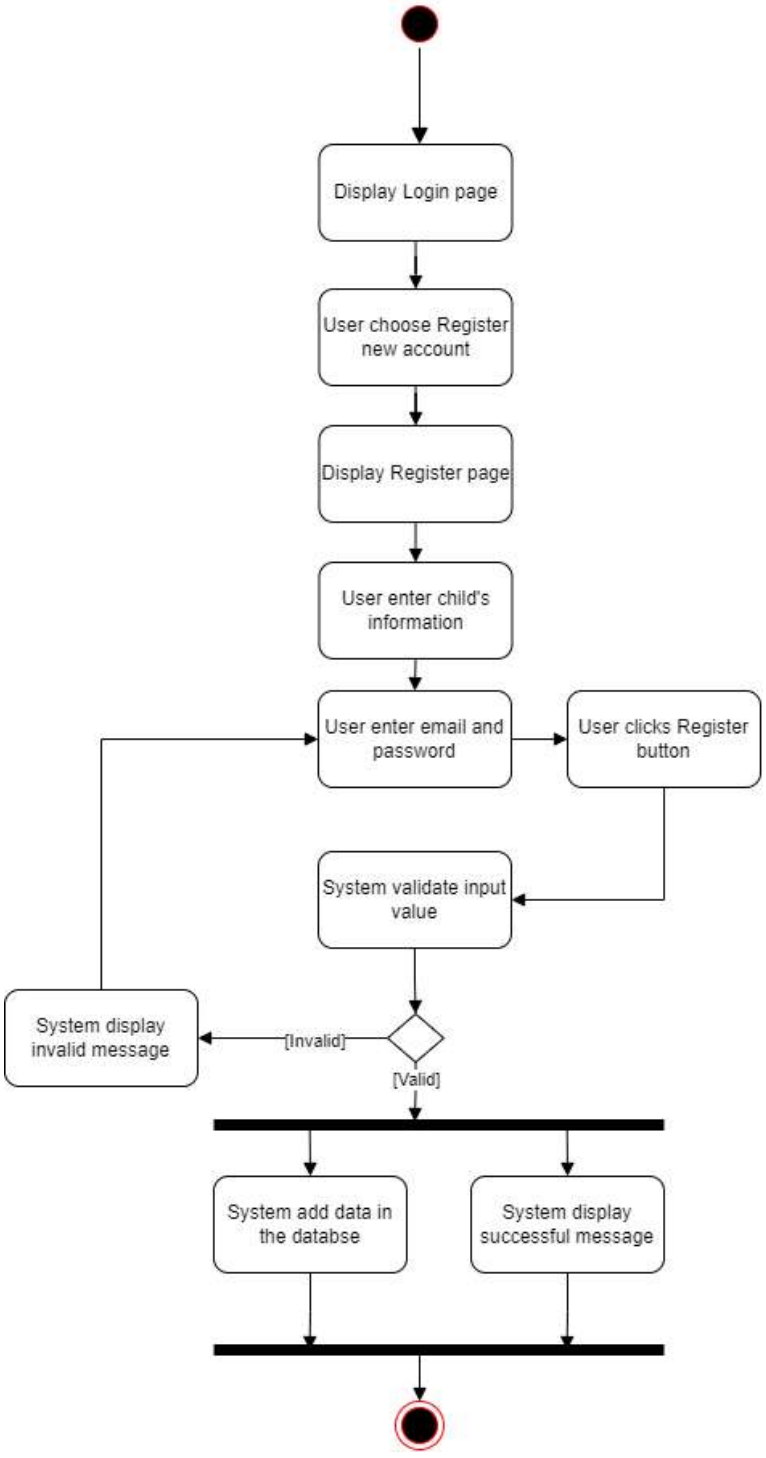


Figure 3.4.32: Activity diagram Register

3.4.4.9 Manage Appointment

Parent

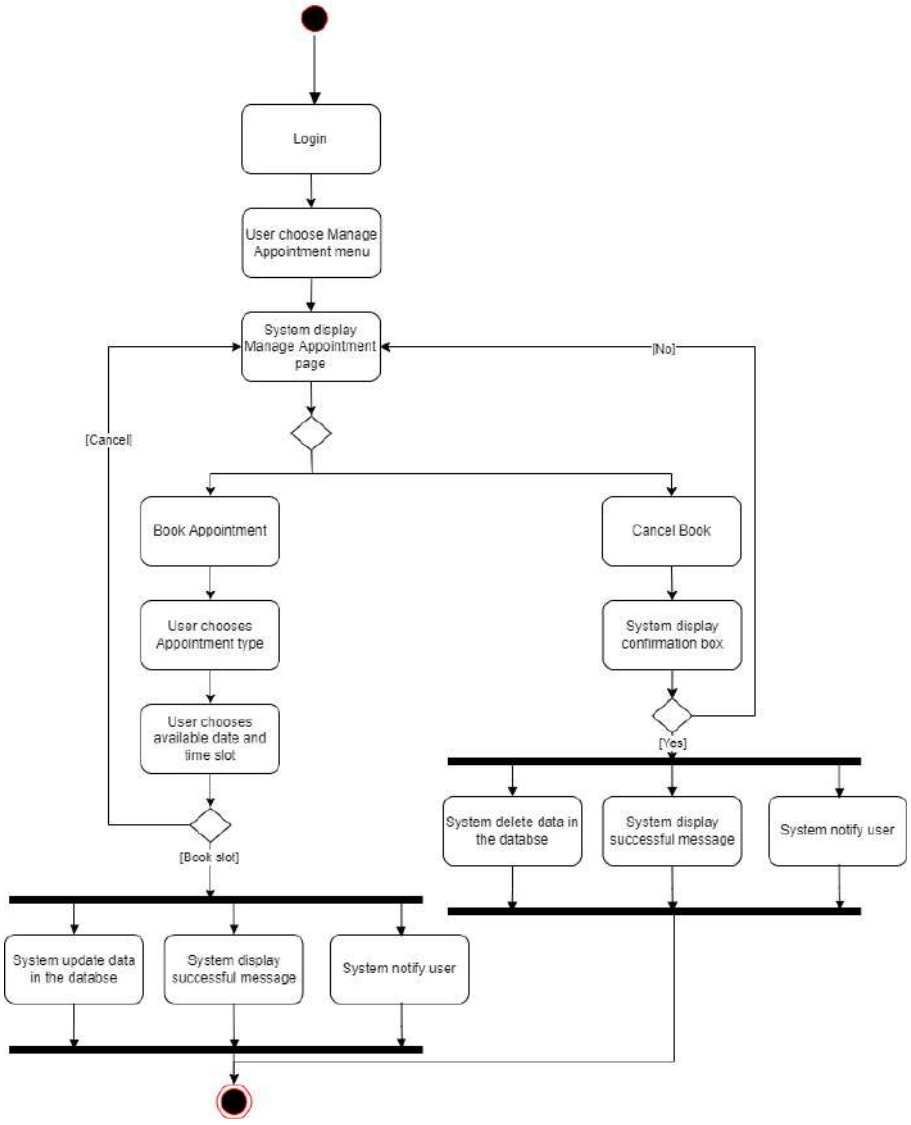


Figure 3.4.33: Activity diagram Manage Appointment (Parent)

Admin

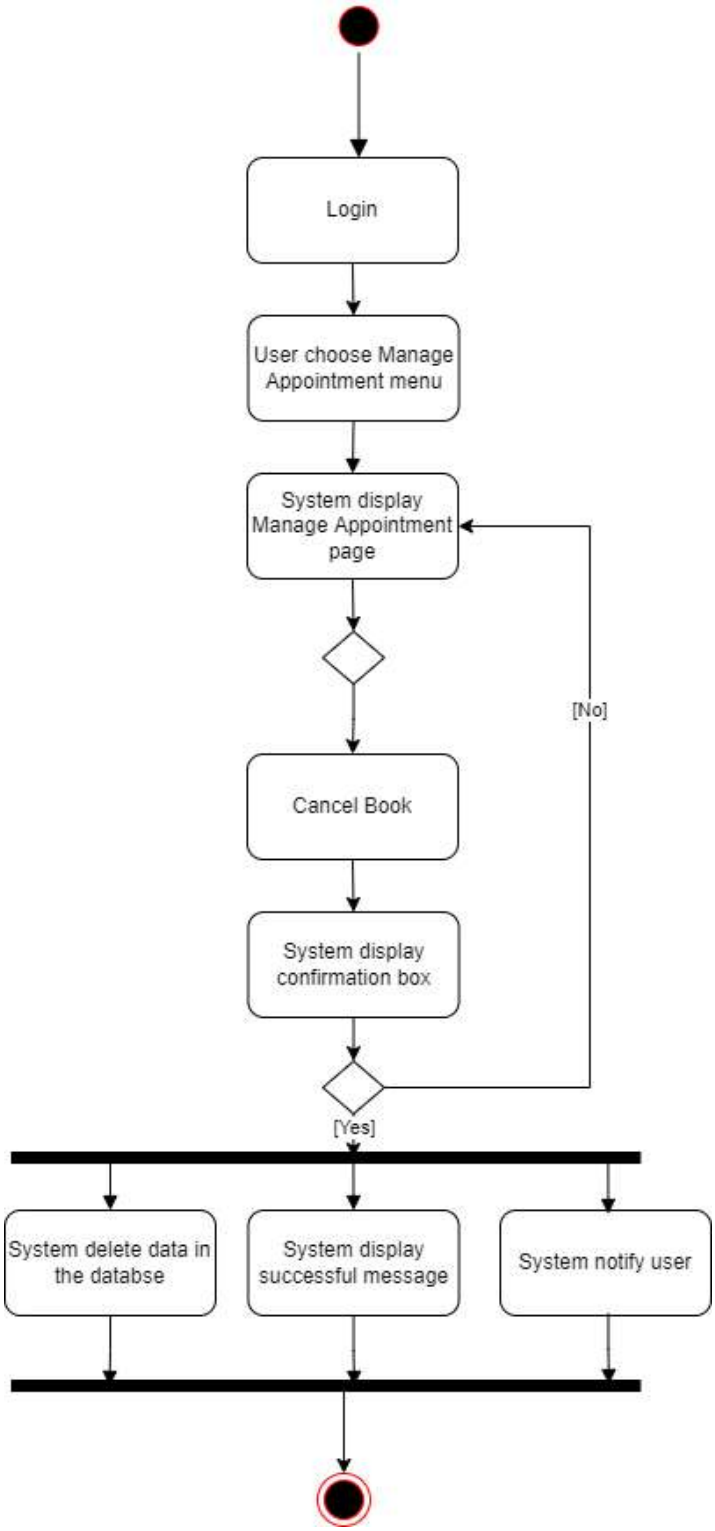


Figure 3.4.34: Activity diagram Manage Appointment (Admin)

LOE staff

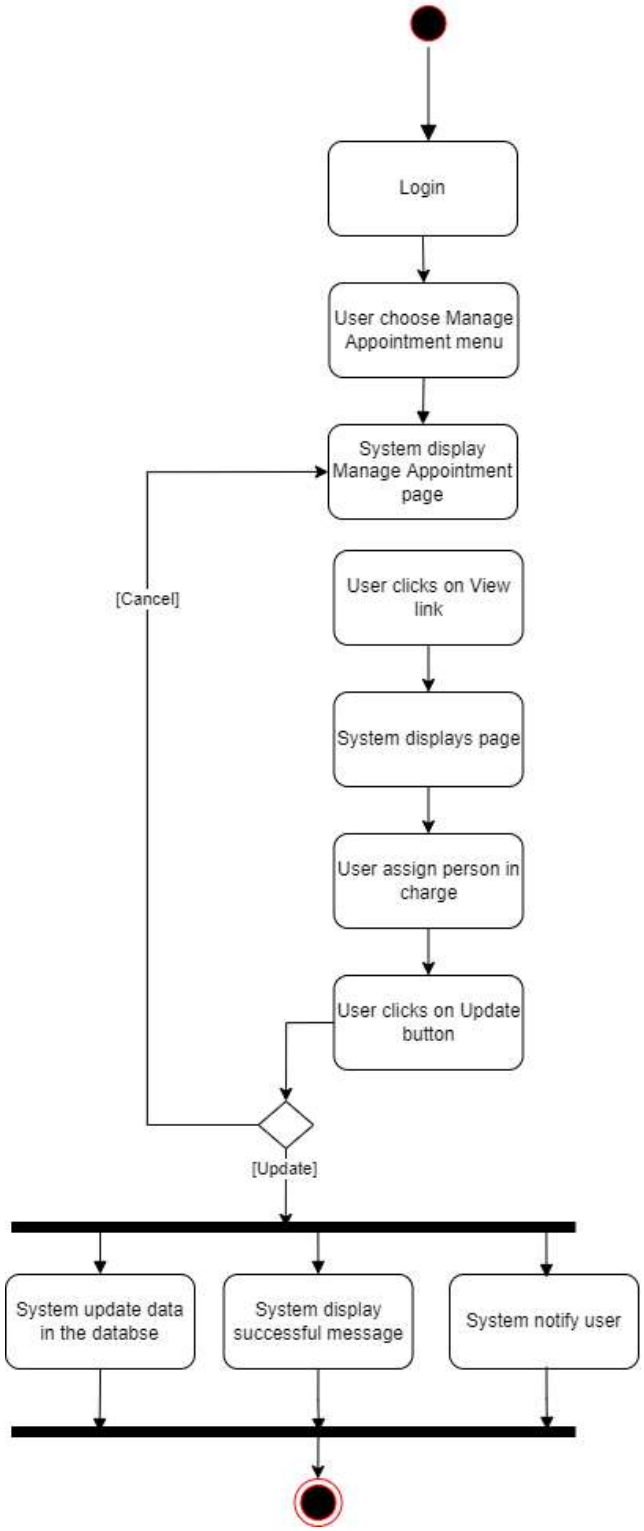


Figure 3.4.35: Activity diagram Manage Appointment (LOE staff)

3.4.4.10 Manage Profile

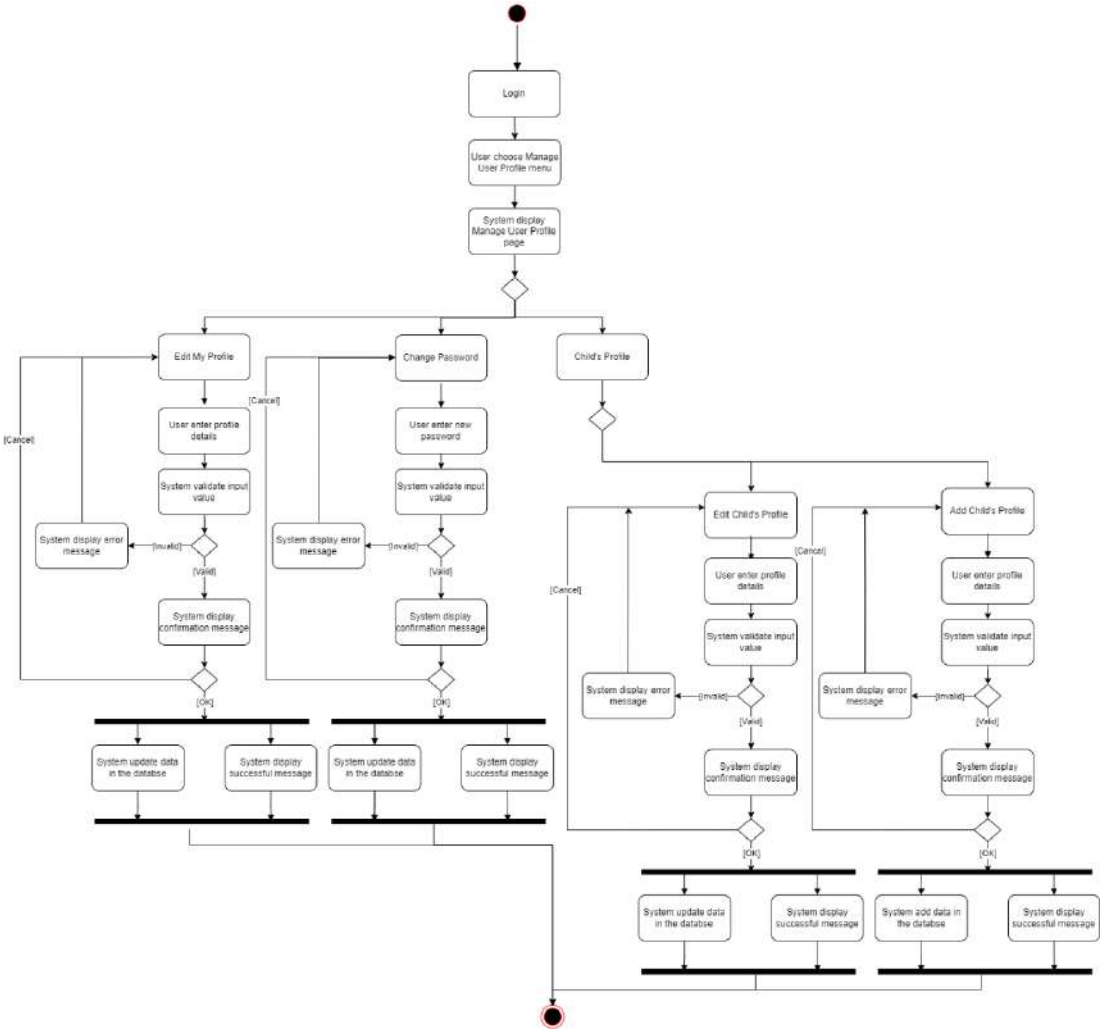


Figure 3.4.36: Activity diagram Manage Profile

3.4.5 Storyboard

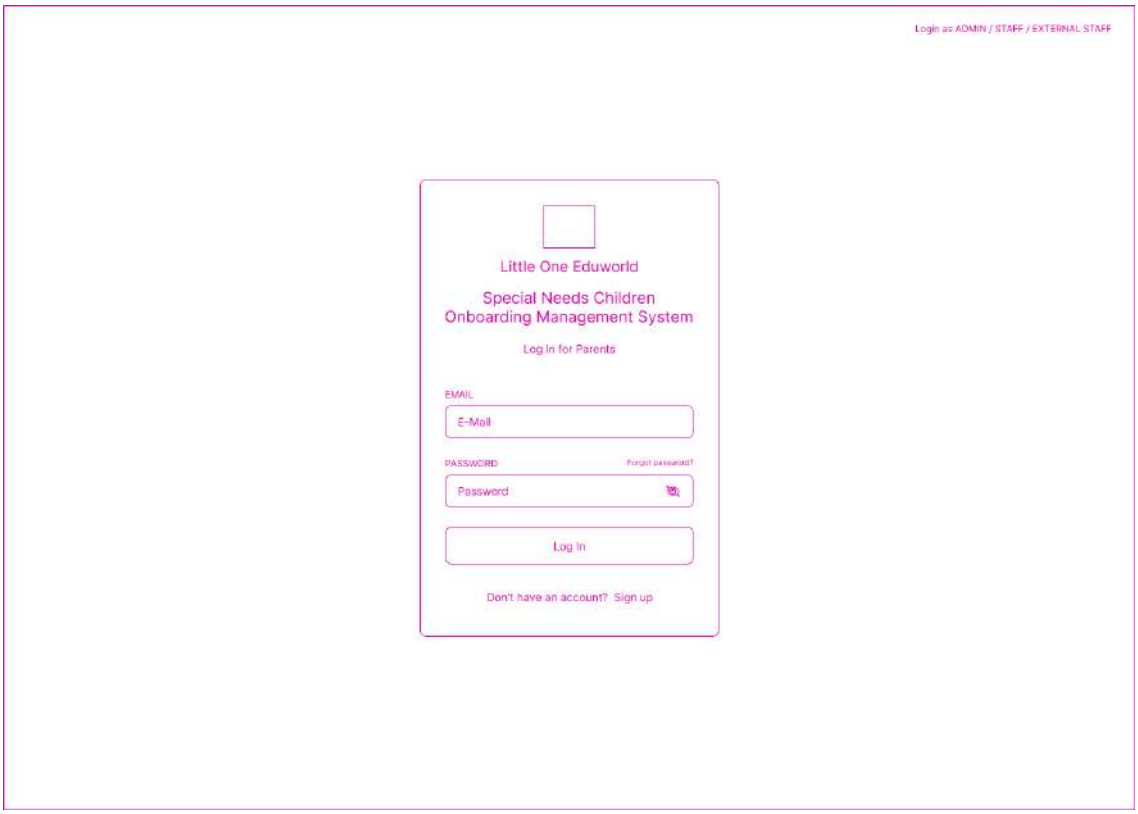


Figure 3.4.37: Login page for Parent

- This is the login page for the parent.
- Users can click “Sign up” link to register new account.
- Users must enter registered email and password.
- Users click the login button to login and navigate to the parent dashboard page.

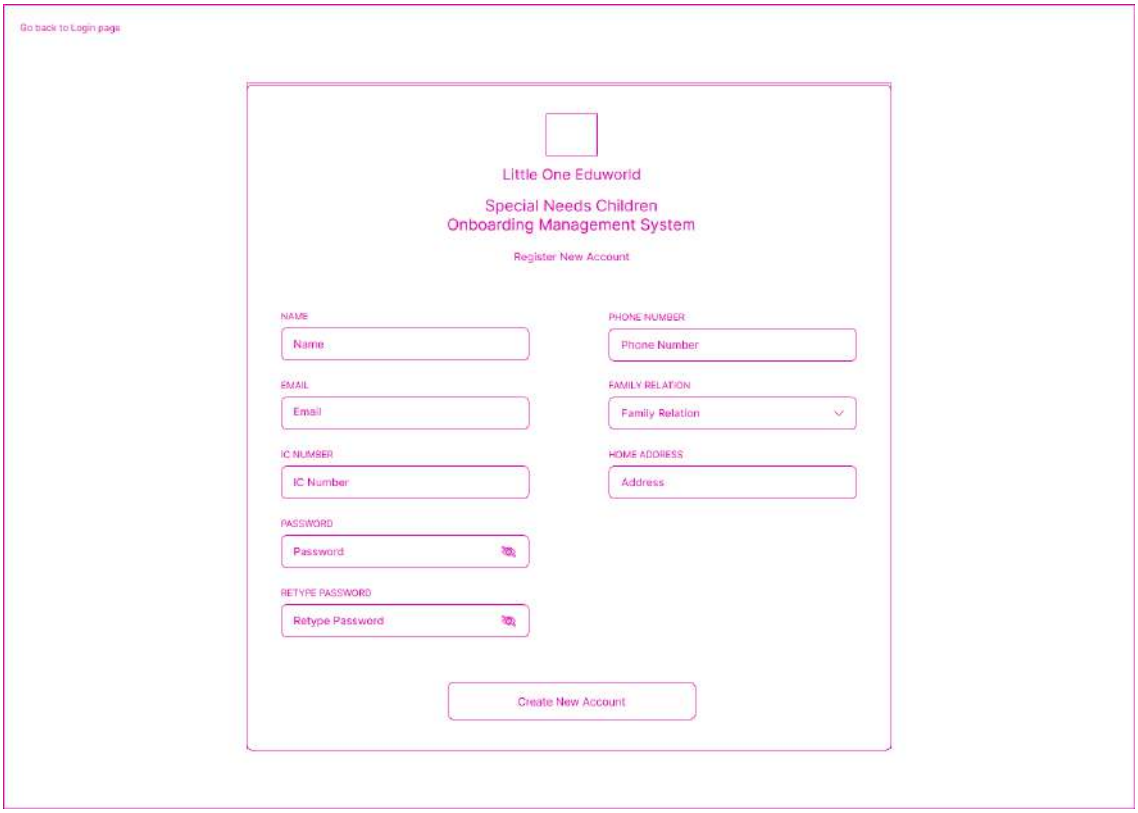


Figure 3.4.38: Register page

- This is the register account page for the parent.
- Users must enter all the required information.
- Users click the “Create New Account” button to create account and navigate to the parent login page.

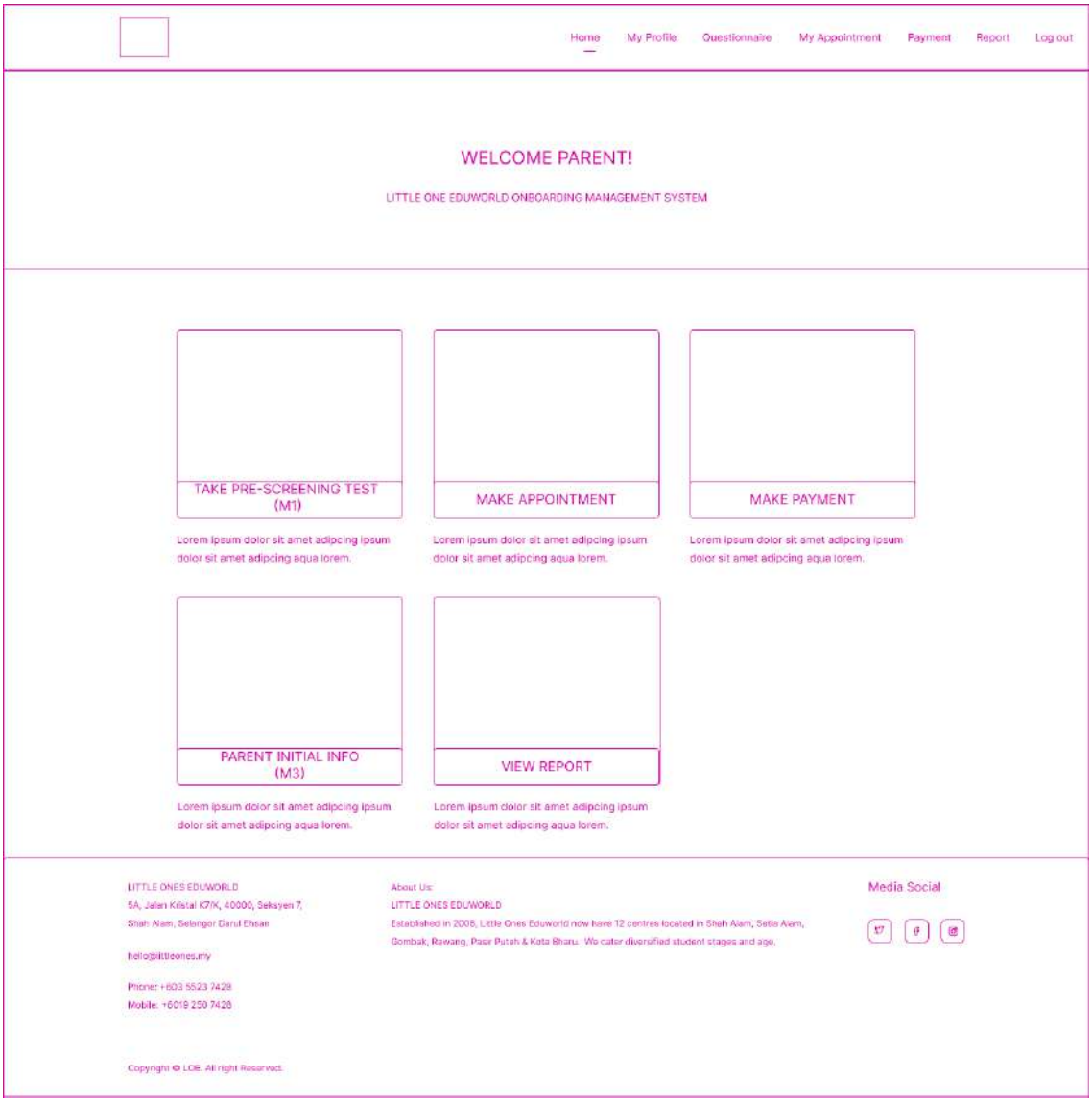


Figure 3.4.39: Parent dashboard page

- This is the dashboard page for the parent.
- Users can choose any menu in dashboard page.
- Users click the “Take Pre-screening Test” menu, it will navigate to the Pre-screening page.
- Users click the “Make appointment” menu, it will navigate to the Appointment page.

- Users click the “Make Payment” menu, it will navigate to the Payment page.
- Users click the “Parent Initial Info” menu, it will navigate to the Parent Initial Info page.
- Users click the “View Report” menu, it will navigate to the Report dashboard page.

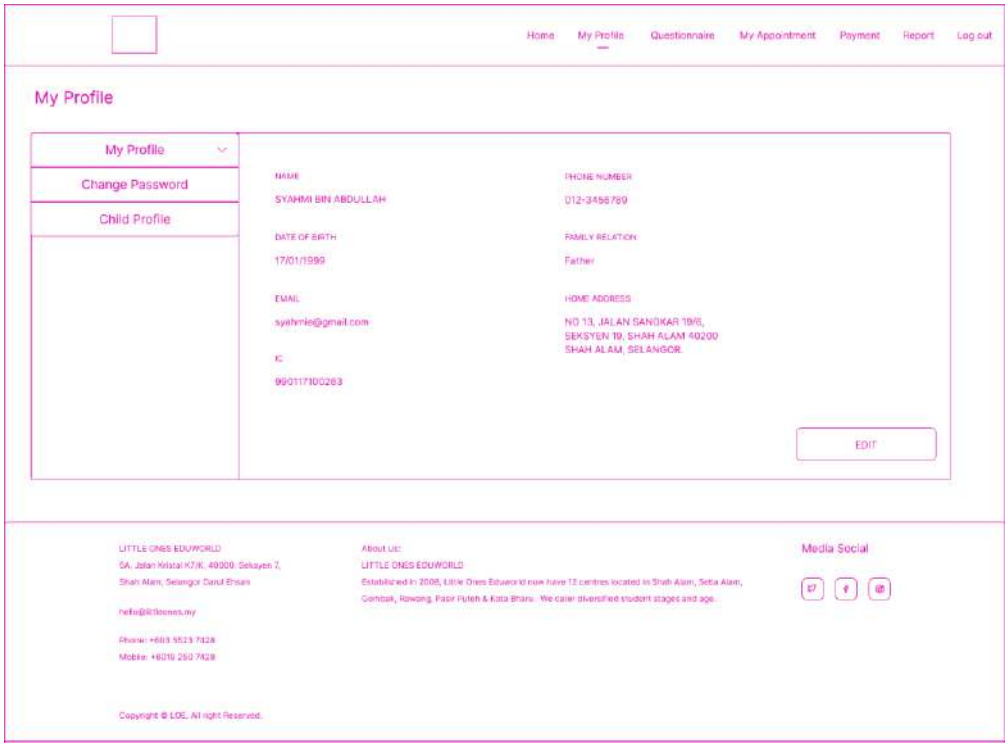


Figure 3.4.40: My Profile page

- This is the My Profile page for the parent.
- Users can choose any menu in this page.
- Users click the “Edit” button, it will navigate to the Edit My Profile page.
- Users can change password by click on “Change Password” menu.
- Users can view, add, edit Child Profile by click on “Child Profile” menu.

Figure 3.4.41: Edit My Profile page

- This is the Edit My Profile page for the parent.
- Users must key in all the required fields.
- Users click the “Save” button, it will navigate to the My Profile page.

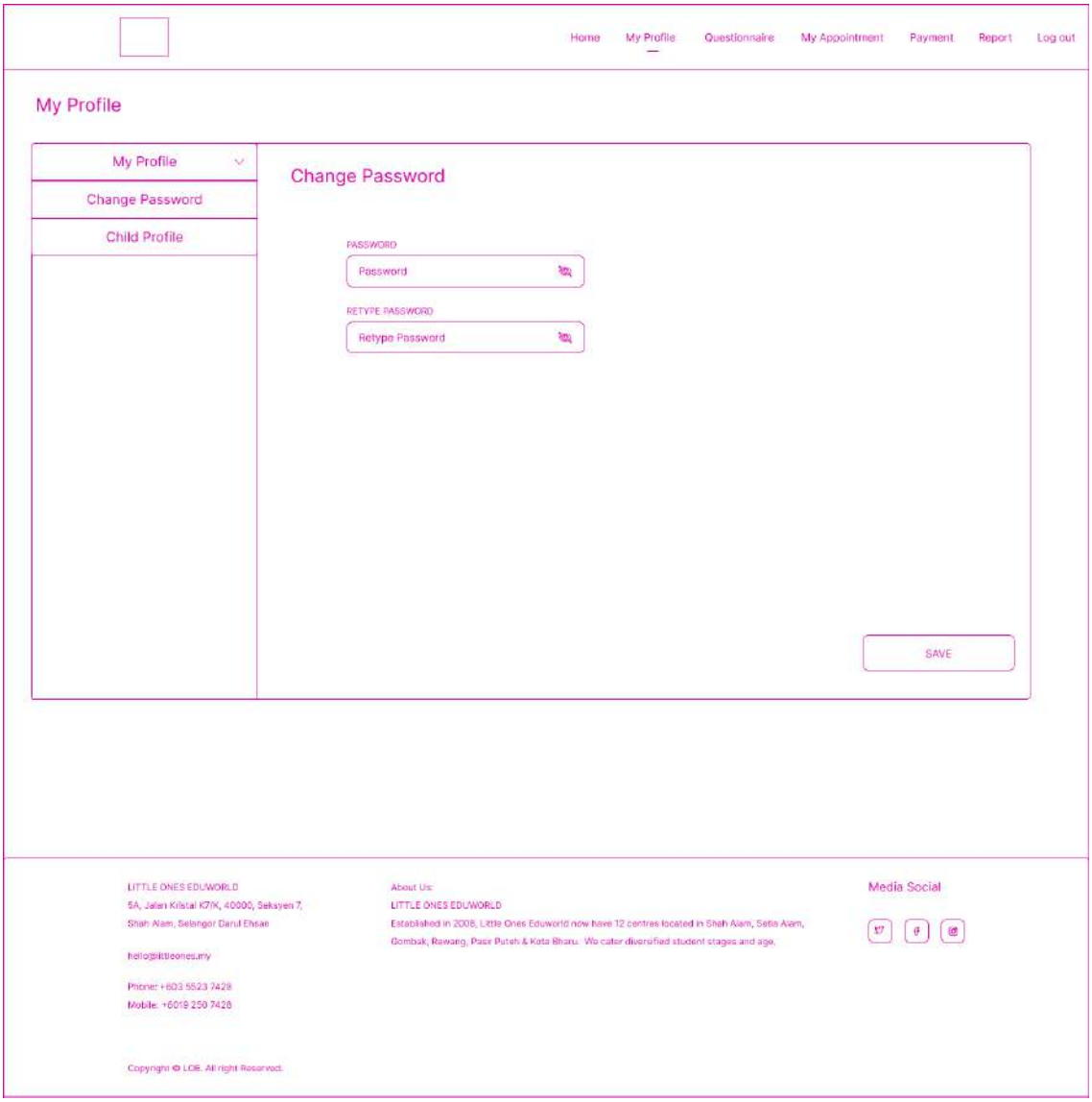


Figure 3.4.42: Change password page

- This is the Change Password page for the parent.
- Users must key in all the required fields.
- Users click the “Save” button to save the changes, it will logged out and navigate to the Login page.

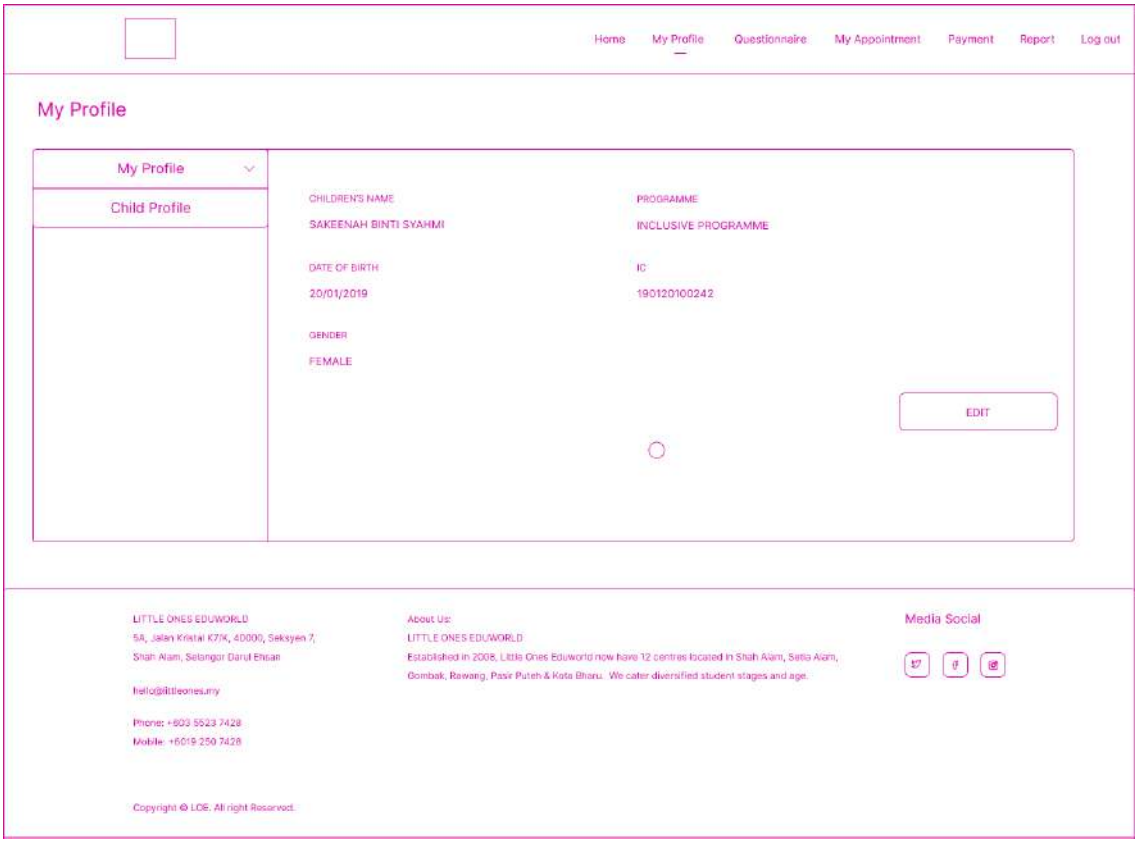


Figure 3.4.43: Child Profile page

- This is the Child Profile page for the child.
- Users click the “Edit” button to edit, it will navigate to the Edit Child Profile page.
- Users can click plus icon to Add Child Profile, it will navigate to Add Child Profile page.

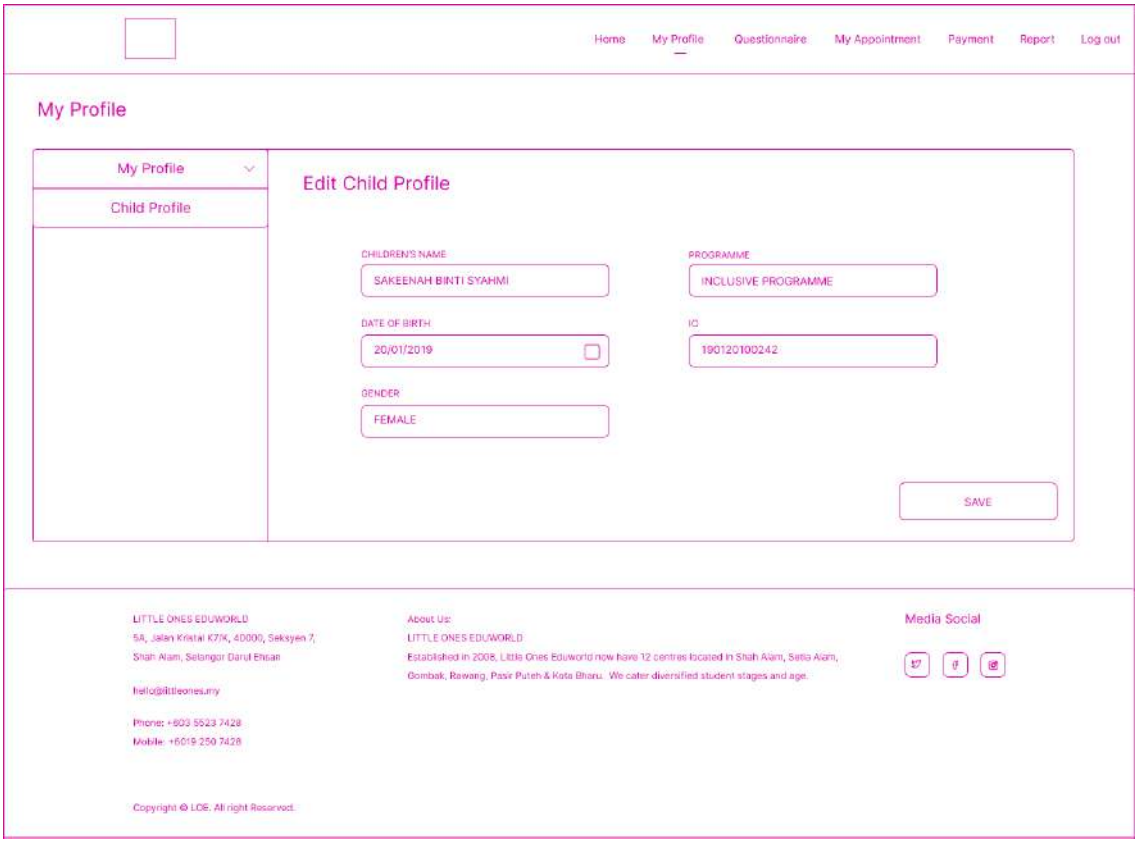


Figure 3.4.44: Edit Child Profile page

- This is the Edit Child Profile page for the child.
- Users must key in all the required fields.
- Users click the “Save” button, it will navigate to the Child Profile page.

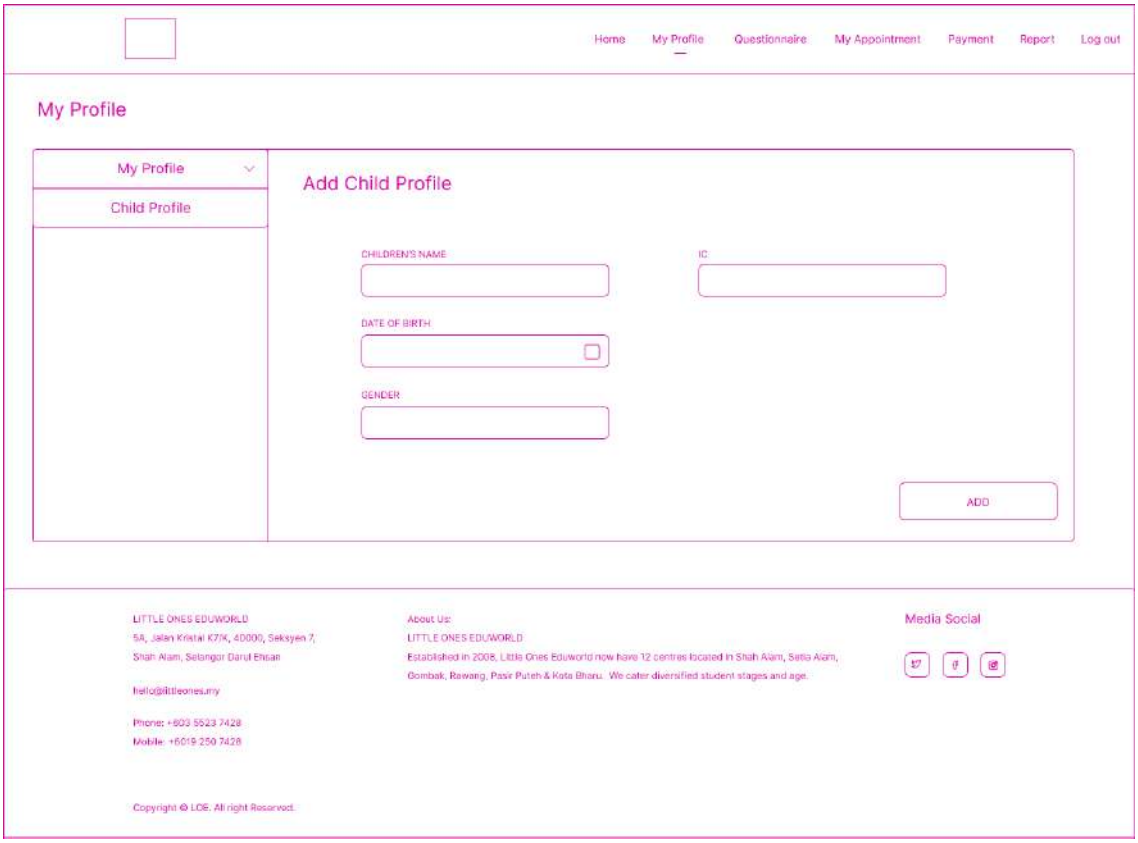


Figure 3.4.45: Add Child Profile page

- This is the Edit Add Child Profile page.
- Users must key in all the required fields.
- Users click the “Add” button, it will navigate to the Child Profile page.

Figure 3.4.46: Appointment page

- This is the Appointment page for the parent.
- Users can cancel book by click on Cancel Book link.
- Users click the “Book Appointment” button, it will navigate to the Book Appointment page.

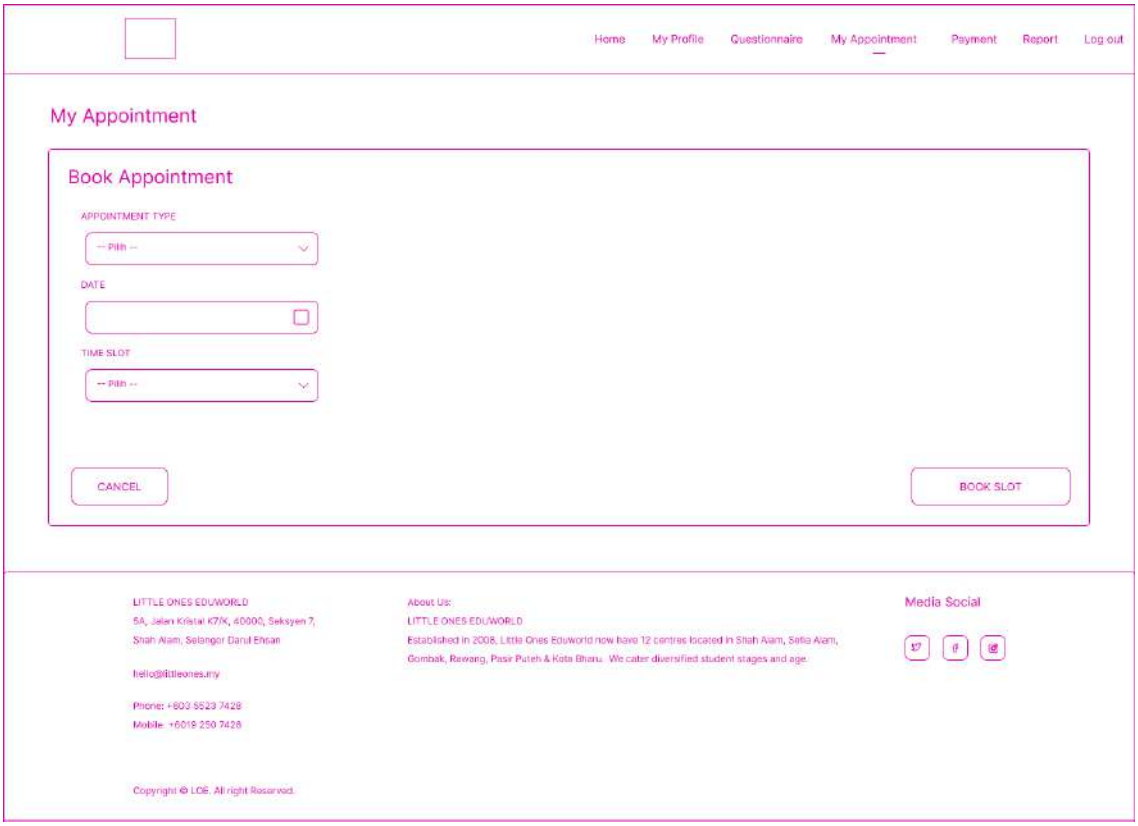


Figure 3.4.47: Book appointment page

- This is the Book Appointment page for the parent.
- Users must key in all the required fields.
- Users click the “Book slot” button, it will navigate to the Appointment page.

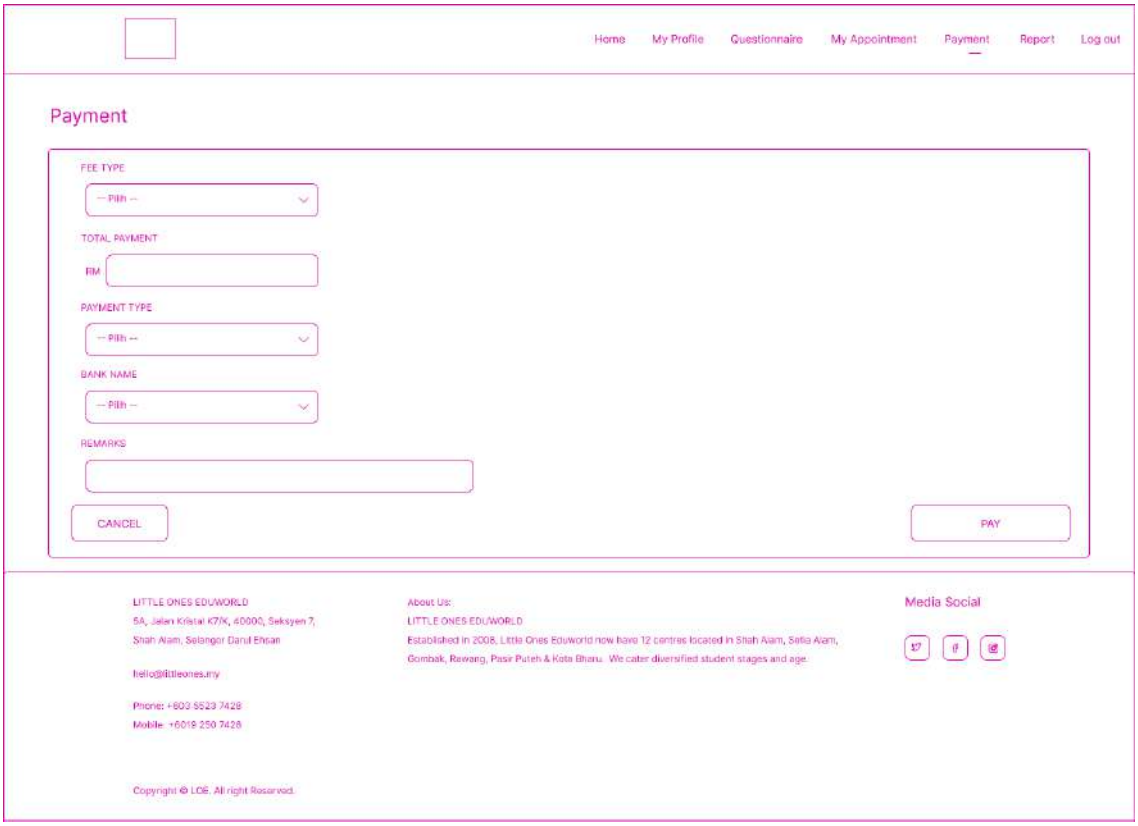


Figure 3.4.48: Make Payment page

- This is the Make payment page for the parent.
- Users must key in all the required fields.
- Users click the “Pay” button, it will navigate to the external system then, dashboard page.

Figure 3.4.49: Questionnaire dashboard page

- This is the Questionnaire dashboard page for the parent.
- Users can choose any menu in the dashboard.
- Users click the “Take Pre-screening Test” menu, it will navigate to the Pre-screening page.
- Users click the “Parent Initial Info” menu, it will navigate to the Parent Initial Info page.

Figure 3.4.50: Pre-screening page

- This is the Pre-screening page for the parent.
- Users must key in all the required fields.
- Users click the “Submit” button, it will navigate to the questionnaire dashboard page.

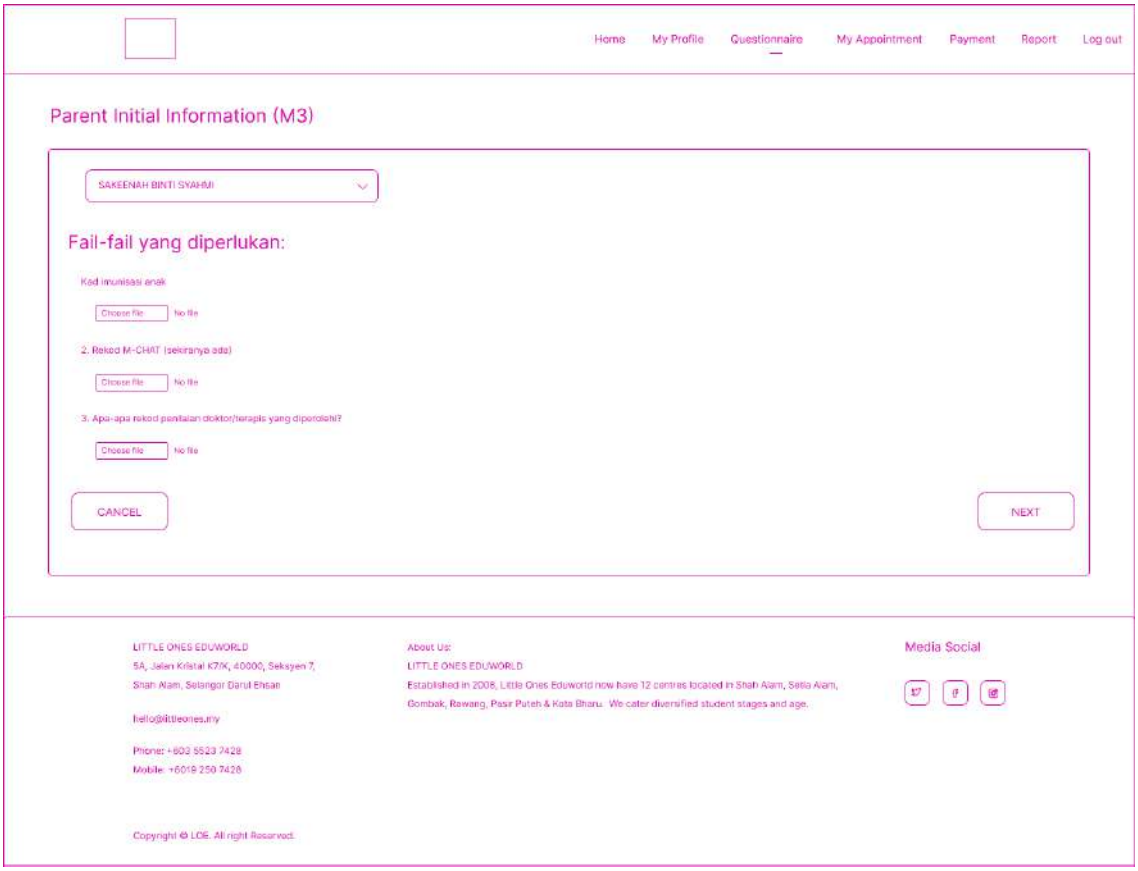


Figure 3.4.51: Parent Initial Info page

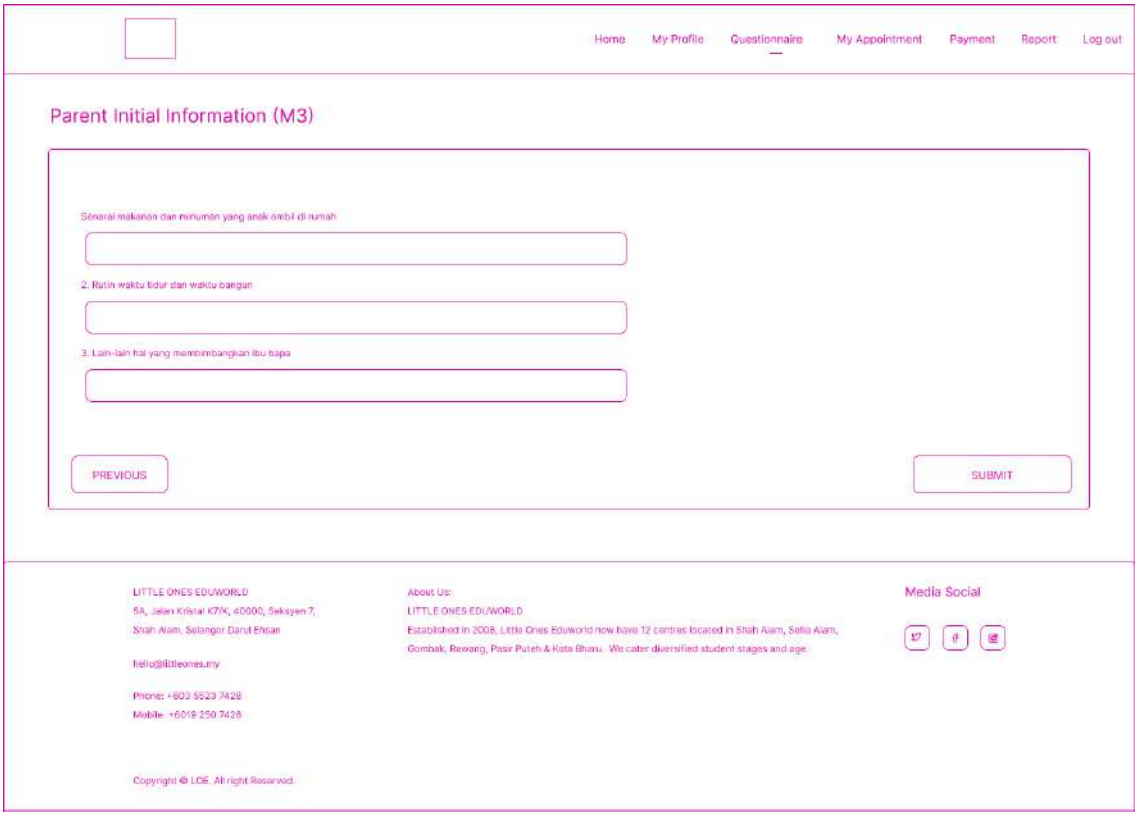


Figure 3.4.52: Parent Initial Info page 2

- These are the Parent Initial Info page for the parent.
- Users must key in all the required fields.
- Users click the “Submit” button, it will navigate to the questionnaire dashboard page.

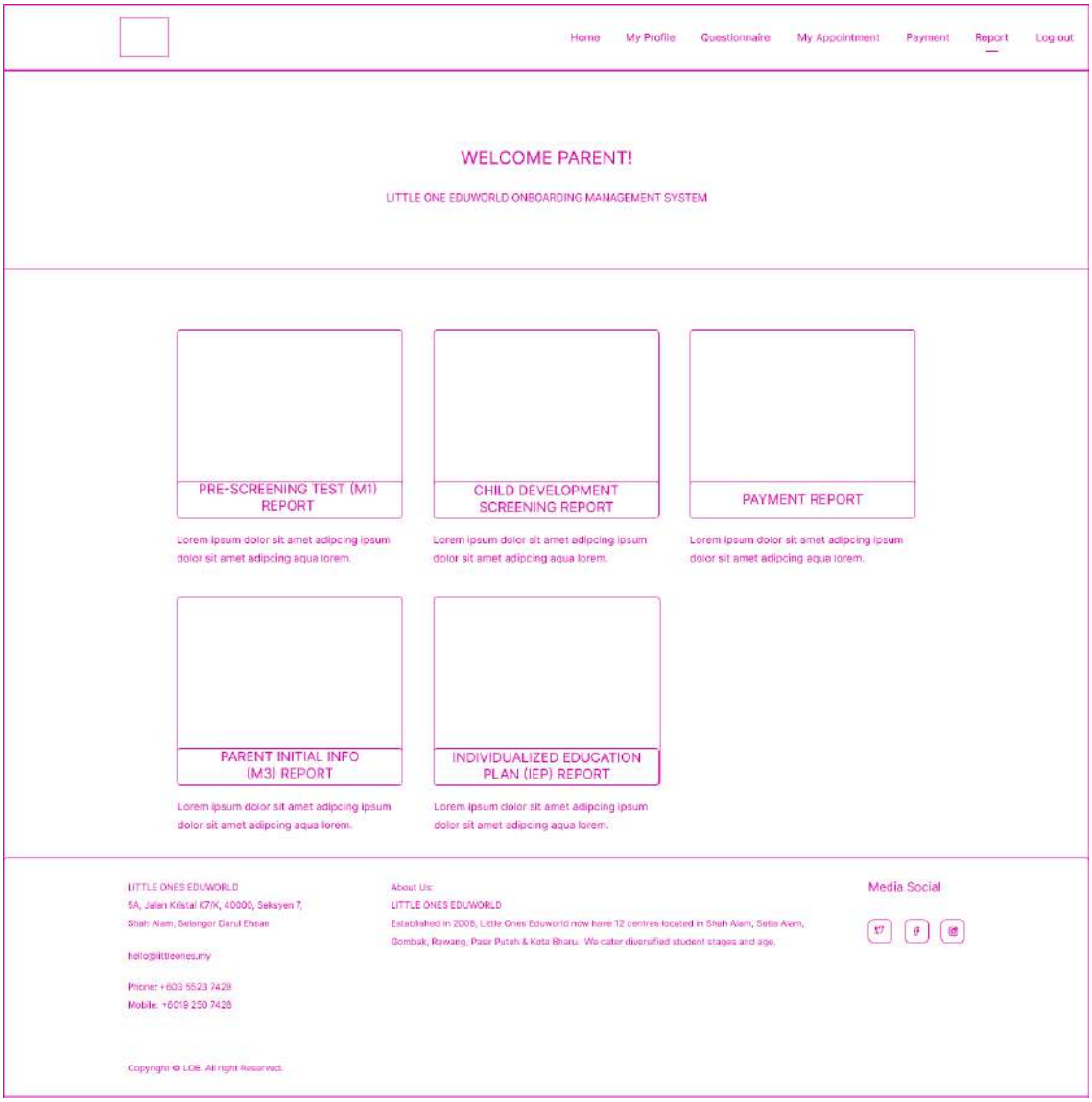


Figure 3.4.53: Report dashboard page

- This is the Report dashboard page for the parent.
- Users can choose any menu on the dashboard.
- Users click the “Pre-screening Test report” menu, it will navigate to the Pre-screening report page.
- Users click the “Child Development screening Test report” menu, it will navigate to the Child Development screening Test report page.

- Users click the “Payment report” menu, it will navigate to the Payment report page.
- Users click the “IEP report” menu, it will navigate to the IEP report page.
- Users click the “Parent Initial Info” menu, it will navigate to the Parent Initial Info page.

Figure 3.4.54: IEP report

- This is the IEP Report page for the parent.
- Users click the “View” link, it will navigate to the IEP report details page.

Figure 3.4.55: IEP report details

- This is the IEP Report details page.
- Users click the “back” button, it will navigate to the IEP report page.

Figure 3.4.56: Payment report page

- This is the Payment Report page.
- Users click the “View” link, it will navigate to the Payment report details page.

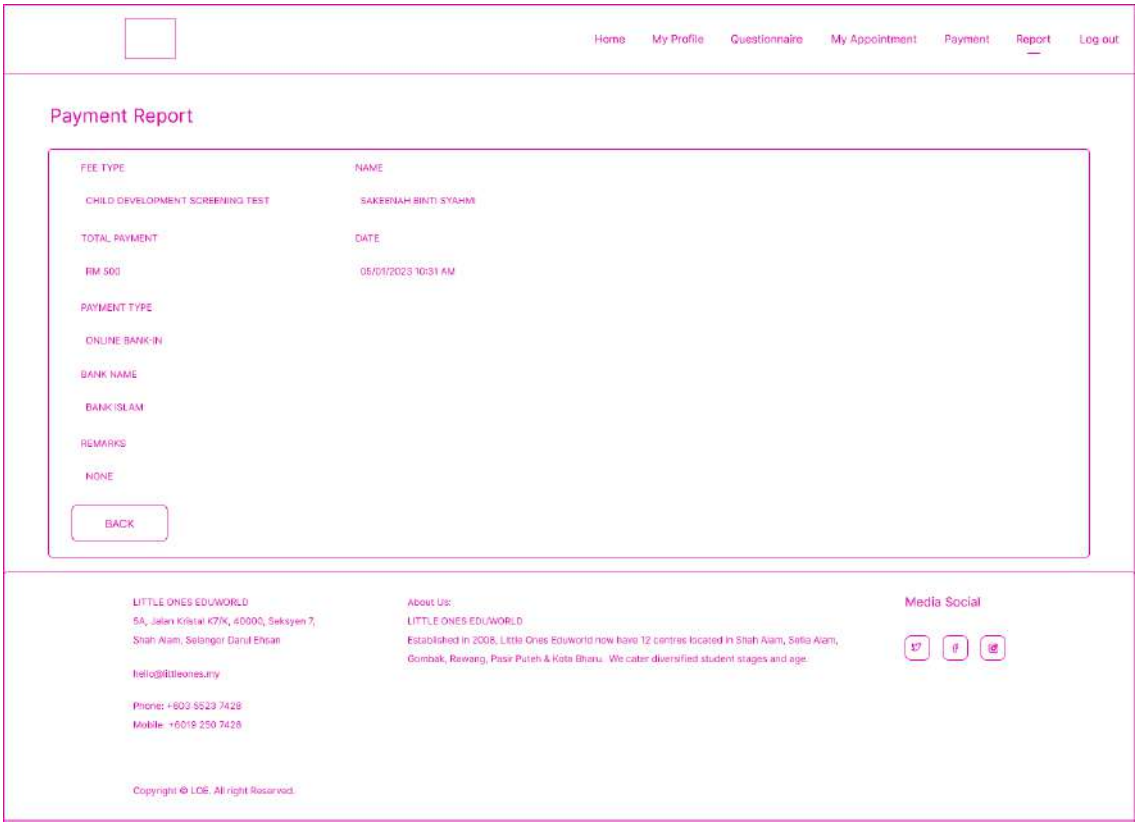


Figure 3.4.57: Payment details report page

- This is the Payment Report details page.
- Users click the “back” button, it will navigate to the Payment report page.

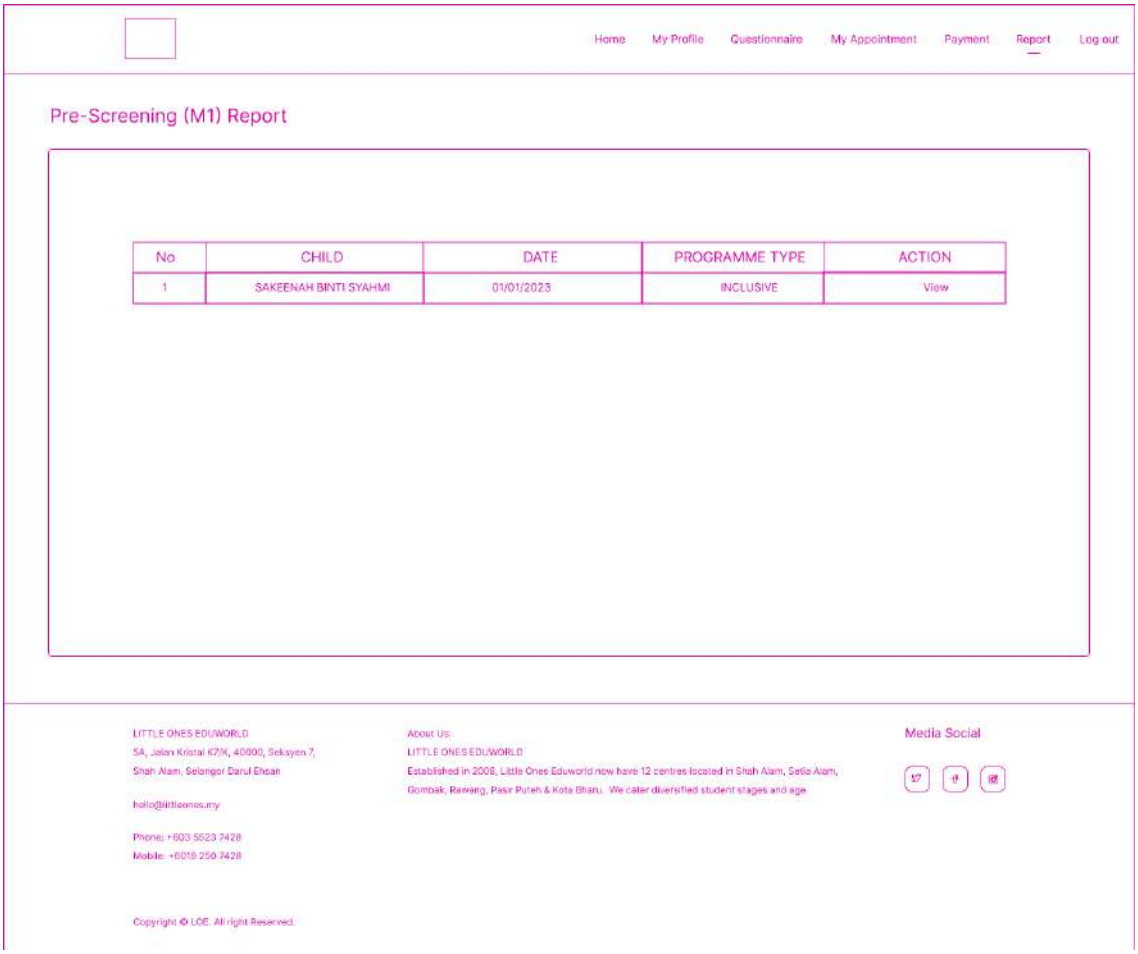


Figure 3.4.58: Pre-screening report page

- This is the Pre-screening Report page.
- Users click the “View” link, it will navigate to the Pre-screening report details page.

Figure 3.4.59: Pre-screening report details page

- This is the Pre-screening Report details page.
- Users click the “back” button, it will navigate to the Pre-screening report page.

Figure 3.4.60: Child Development screening report page

- This is the Child Development screening Report page.
- Users click the “View” link, it will navigate to the Child Development screening report details page.

Figure 3.4.61: Child Development screening details report page

- This is the Child Development screening Report details page.
- Users click the “back” button, it will navigate to the Child Development screening report page.

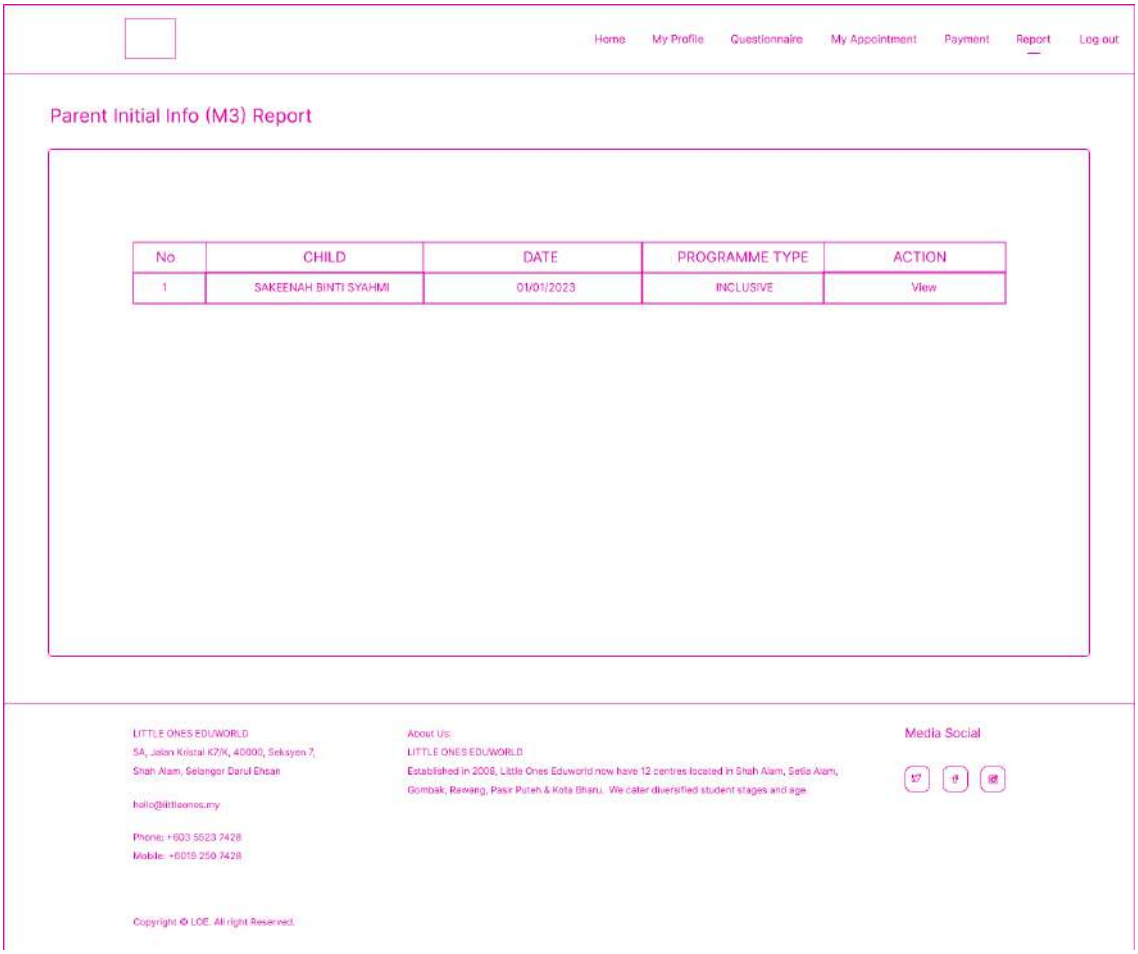


Figure 3.4.62: Parent Initial Info report page

- This is the Parent Initial Info Report page.
- Users click the “View” link, it will navigate to the M3 report details page.

Figure 3.4.63: Parent Initial Info details report page

- This is the M3 Report details page.
- Users click the “back” button, it will navigate to the Parent Initial Info report page.

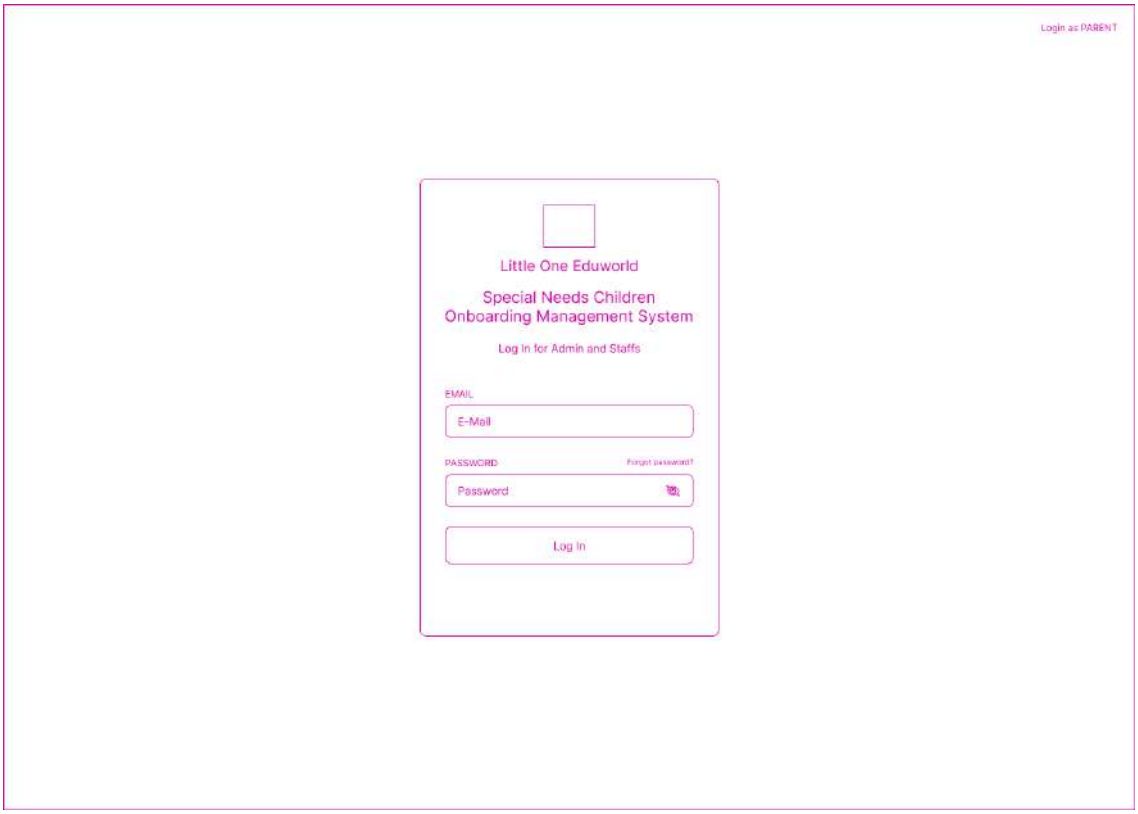


Figure 3.4.64: Login page for Admin and LOE staff

- This is the login page for the admin and LOE staff.
- Users must enter registered email and password.
- Users click the login button to login and navigate to the dashboard page.

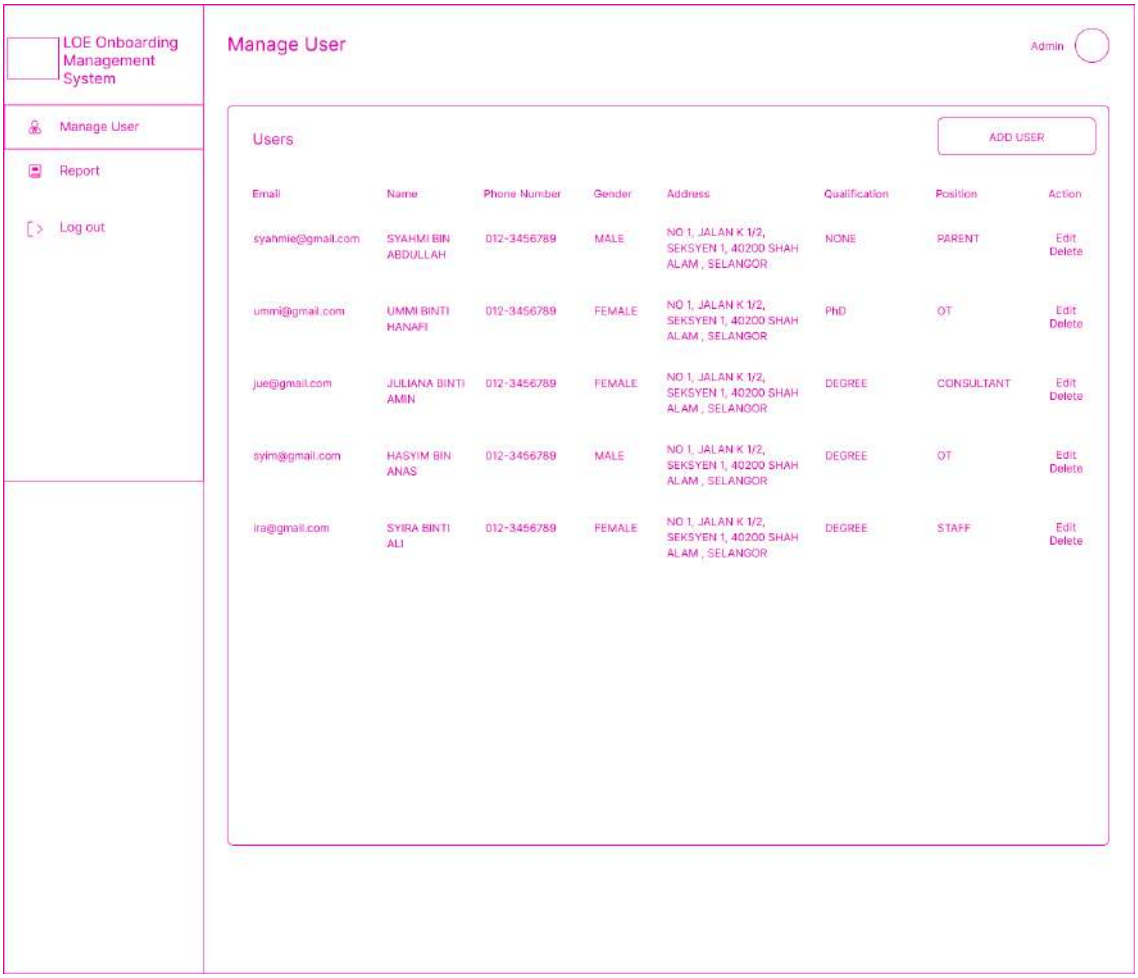


Figure 3.4.65: Manage User page

- This is the manage user page.
- Users click the Add User button to add user and navigate to the Add New User page.
- Users click the Edit link to edit user details and navigate to the Edit User page.
- Users click the Delete link to delete user and refresh the page.

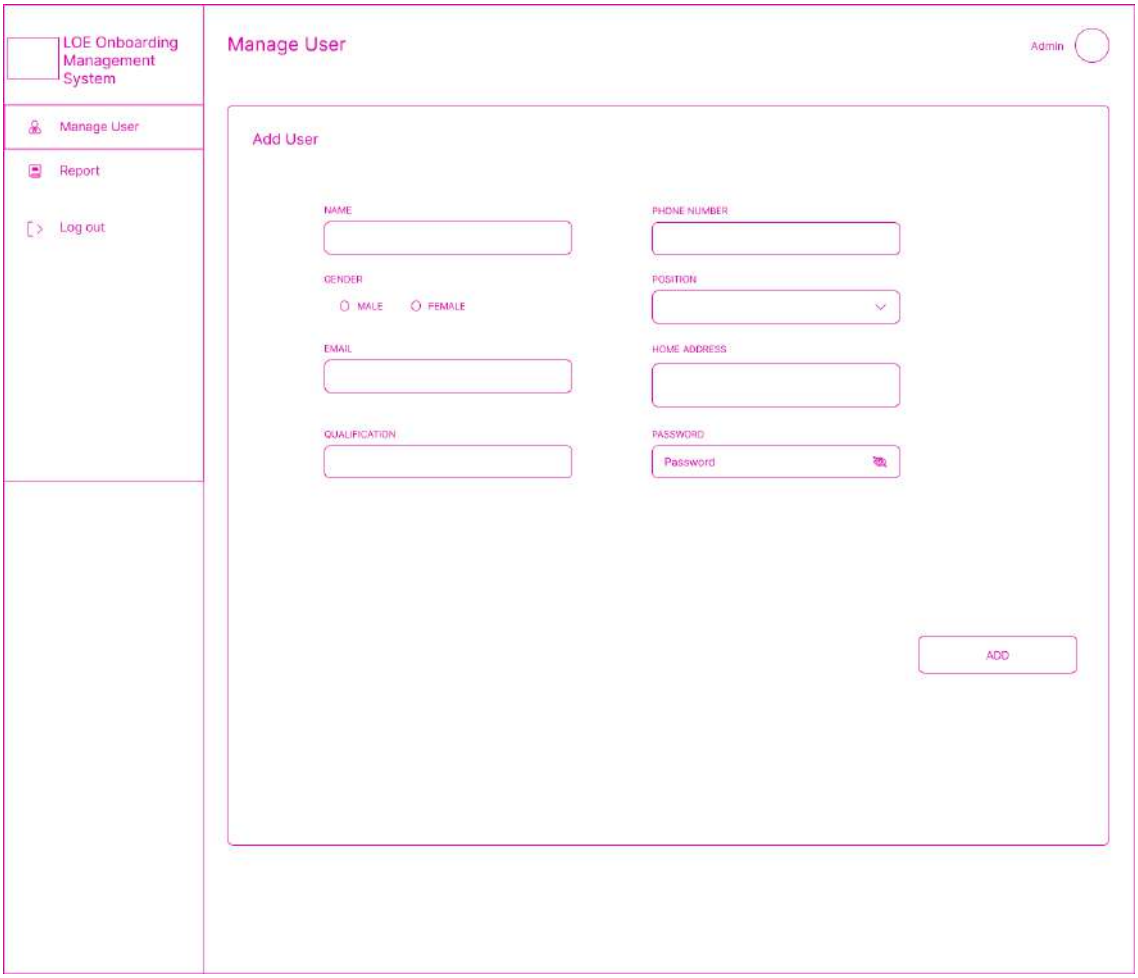


Figure 3.4.66: Add User page

- This is the Add User page
- Users must key in all the required fields.
- Users click the “Add” button, it will navigate to the Manage User page.

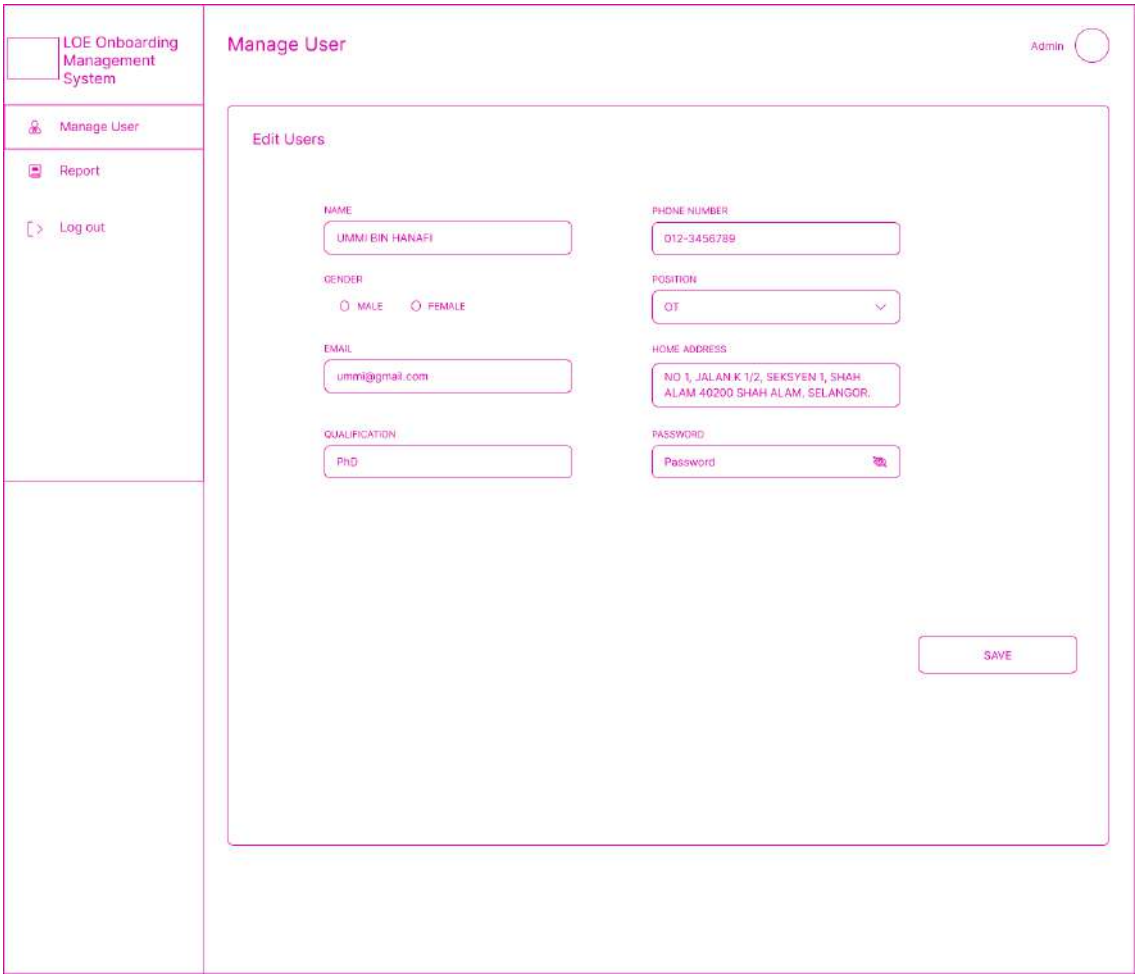


Figure 3.4.67: Edit user page

- This is the Edit User page
- Users must key in all the required fields.
- Users click the “Save” button, it will navigate to the Manage User page.

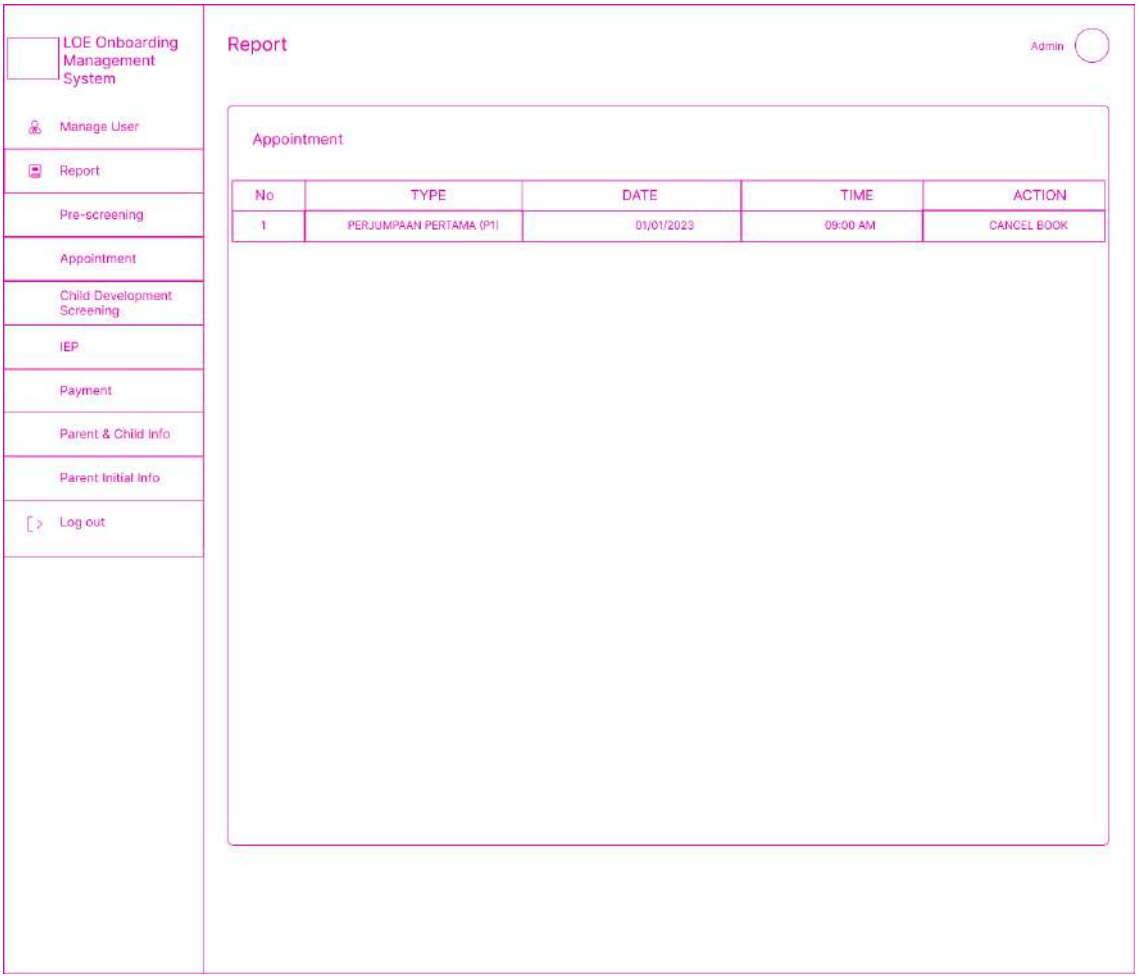


Figure 3.4.68: Appointment page

- This is the Appointment page.
- Users can cancel book by click on Cancel Book link and it will delete the data and refresh the page

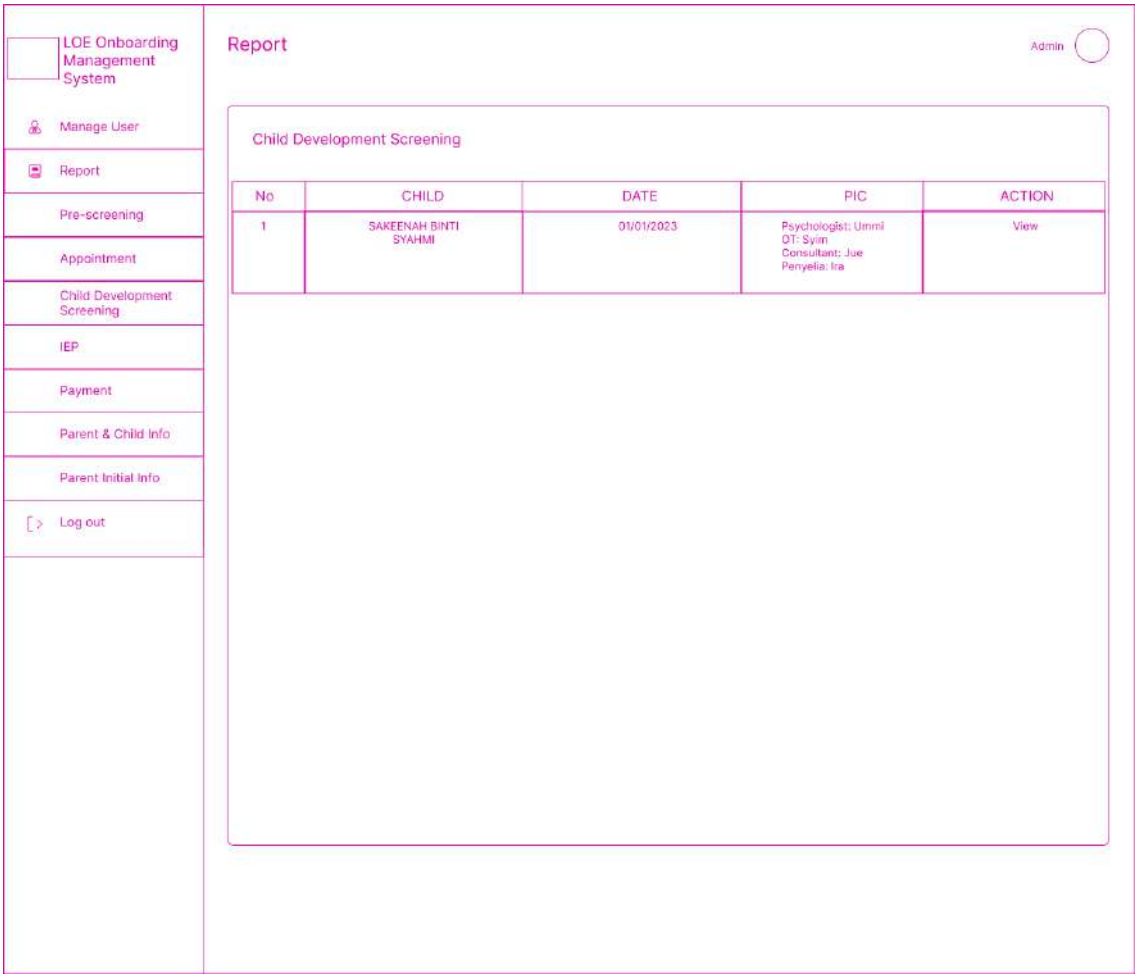


Figure 3.4.69: Child Development Screening report page

- This is the Child Development screening Report page.
- Users click the “View” link, it will navigate to the Child Development screening report details page.

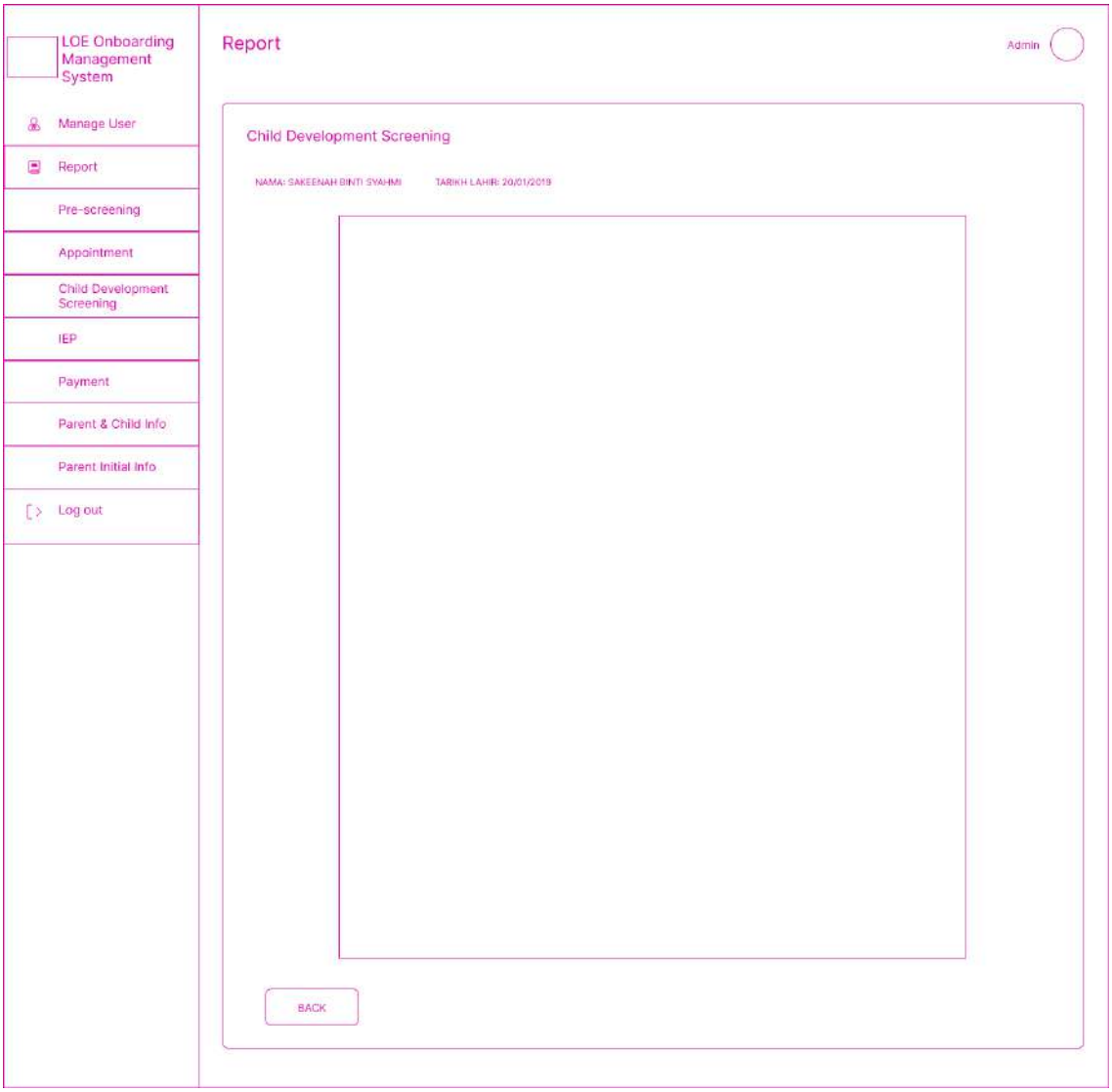


Figure 3.4.70: Child Development Screening details report page

- This is the Child Development screening Report details page.
- Users click the “back” button, it will navigate to the Child Development screening report page.

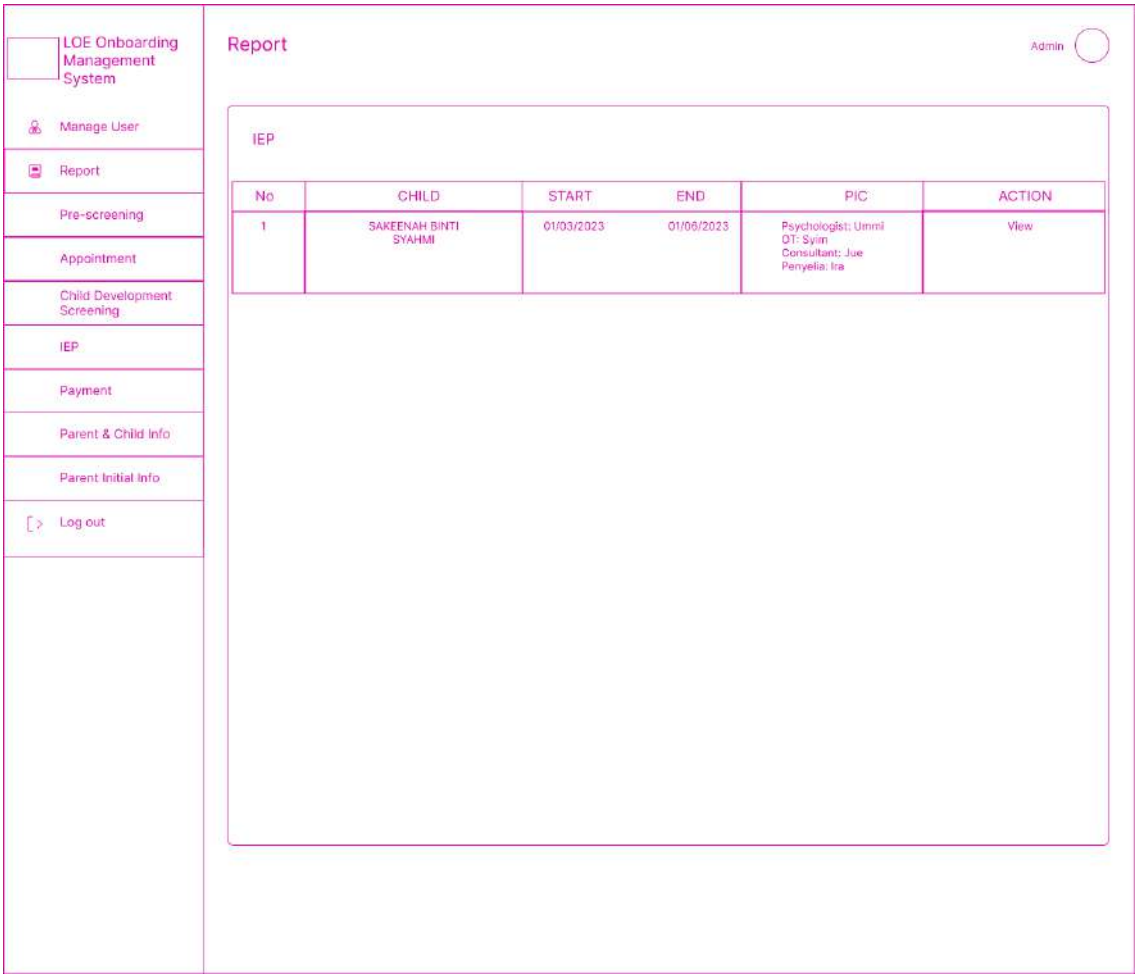


Figure 3.4.71: IEP report page

- This is the IEP Report page for the parent.
- Users click the “View” link, it will navigate to the IEP report details page.

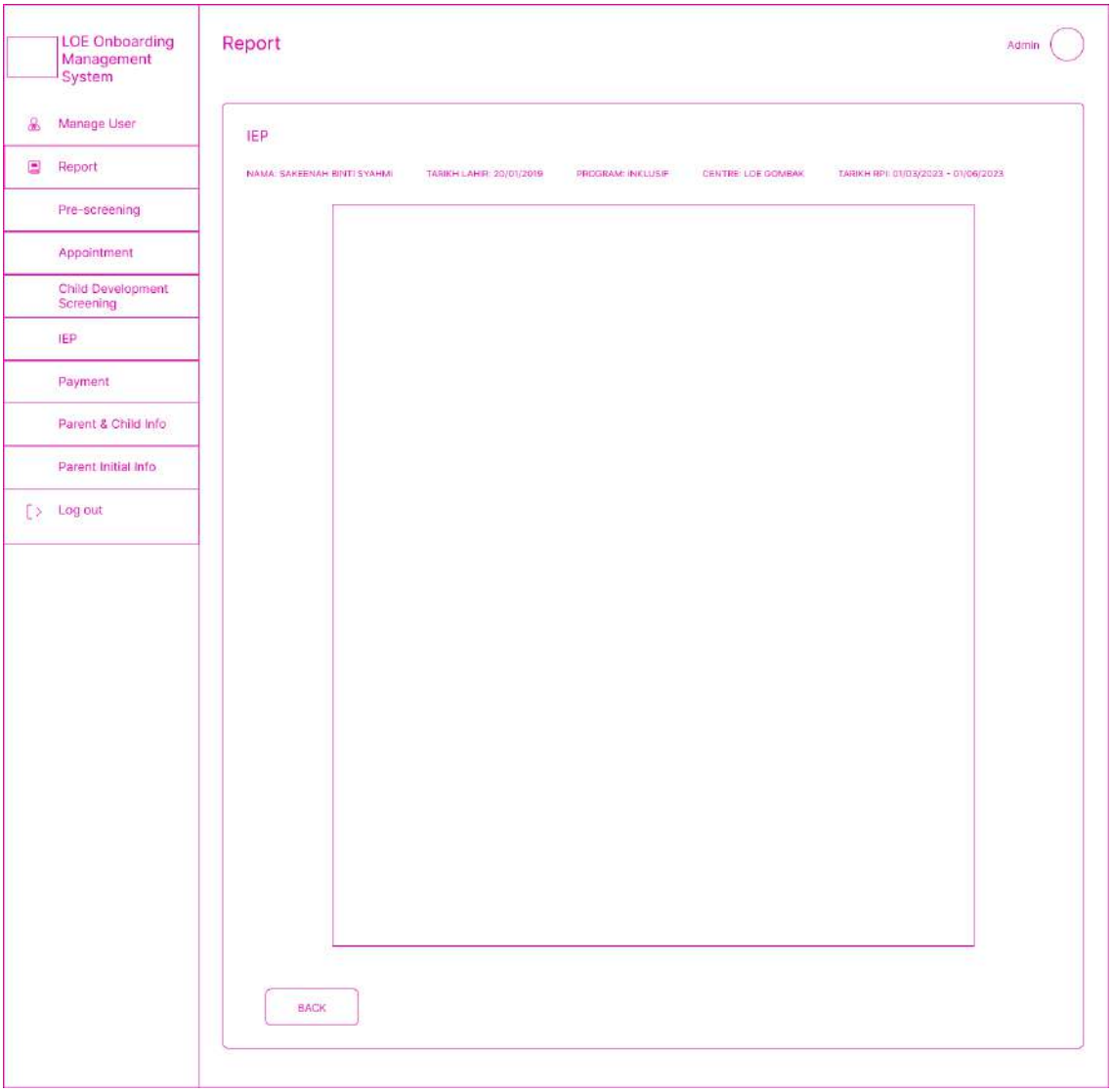


Figure 3.4.72: IEP report details page

- This is the IEP Report details page.
- Users click the “back” button, it will navigate to the IEP report page.

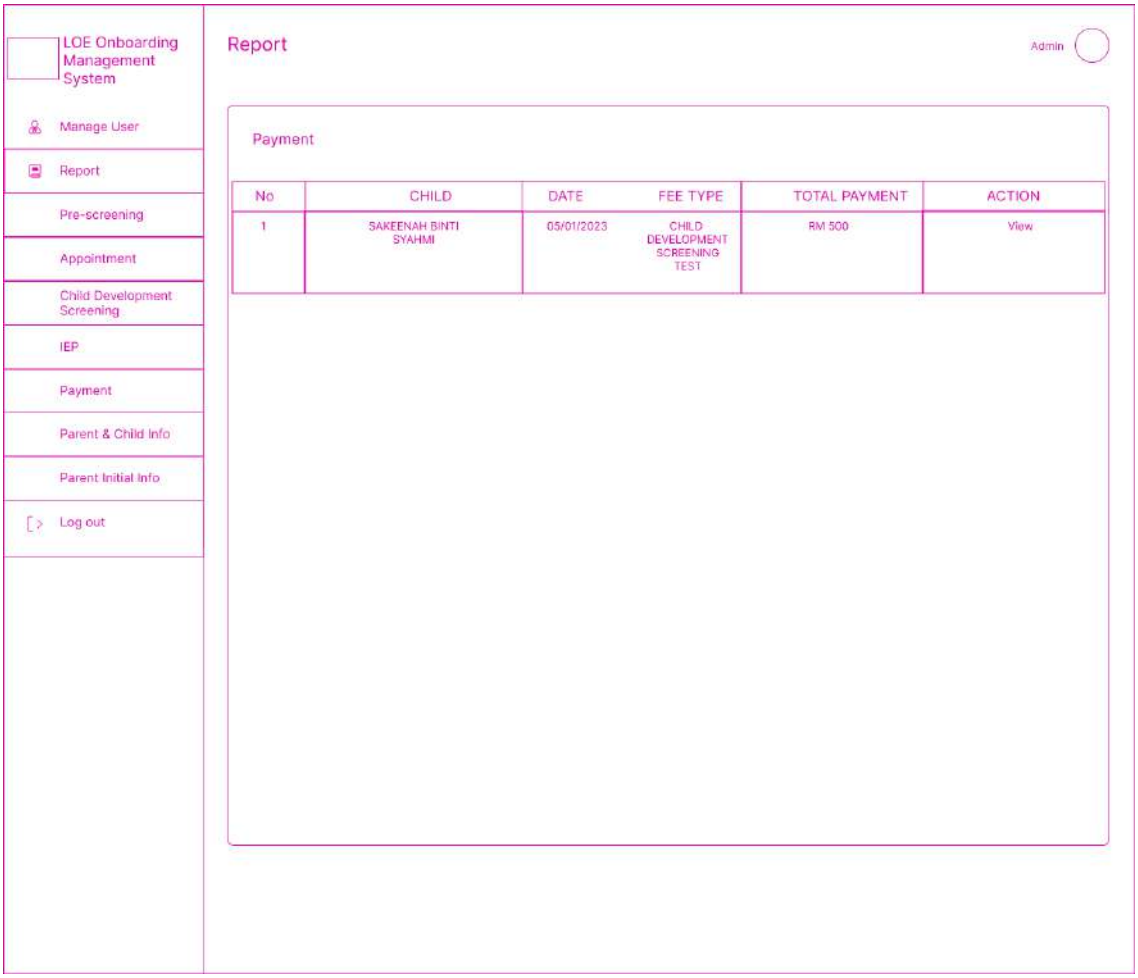


Figure 3.4.73: Payment report page

- This is the Payment Report page.

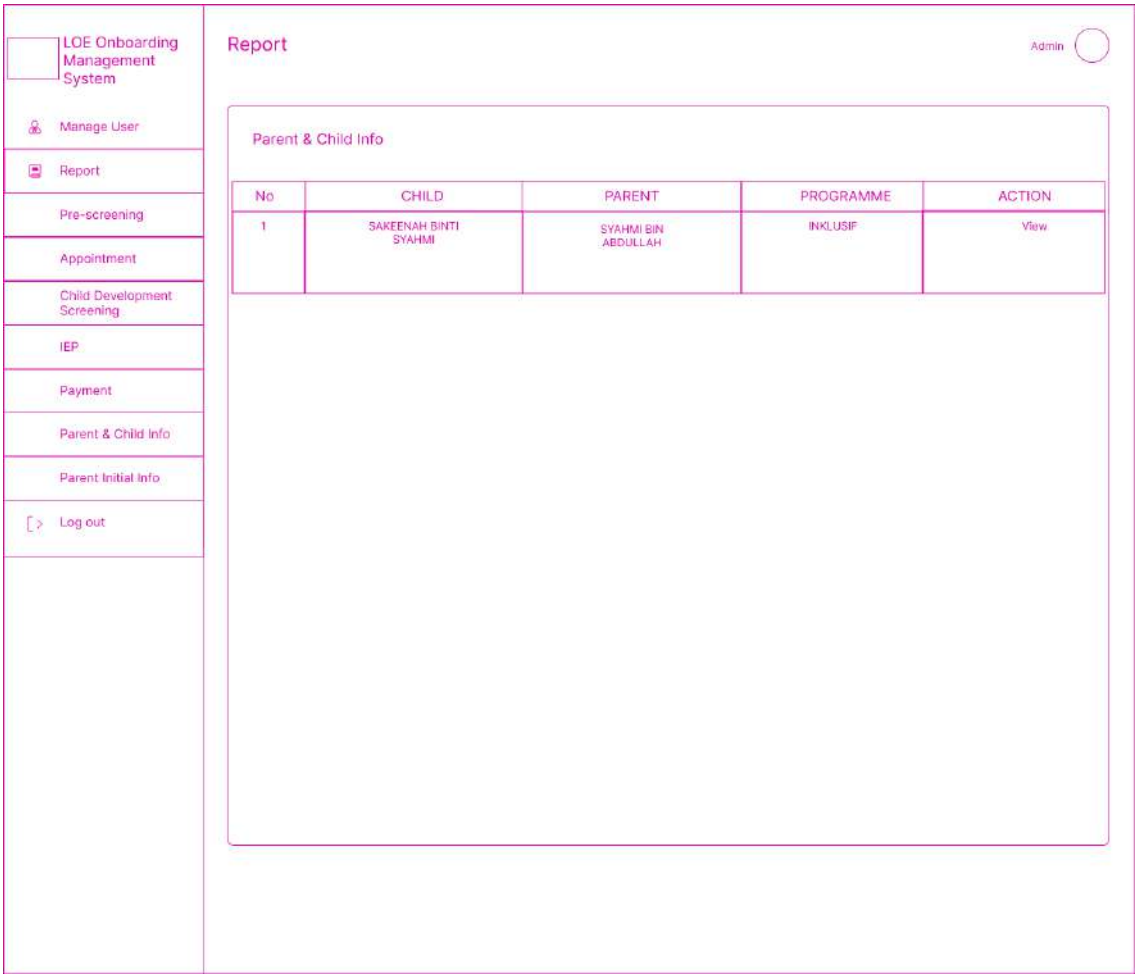


Figure 3.4.74: Parent Child Info page

- This is the Parent Child Info Report page.
- Users click the “View” link, it will navigate to the Parent Child Info report details page.

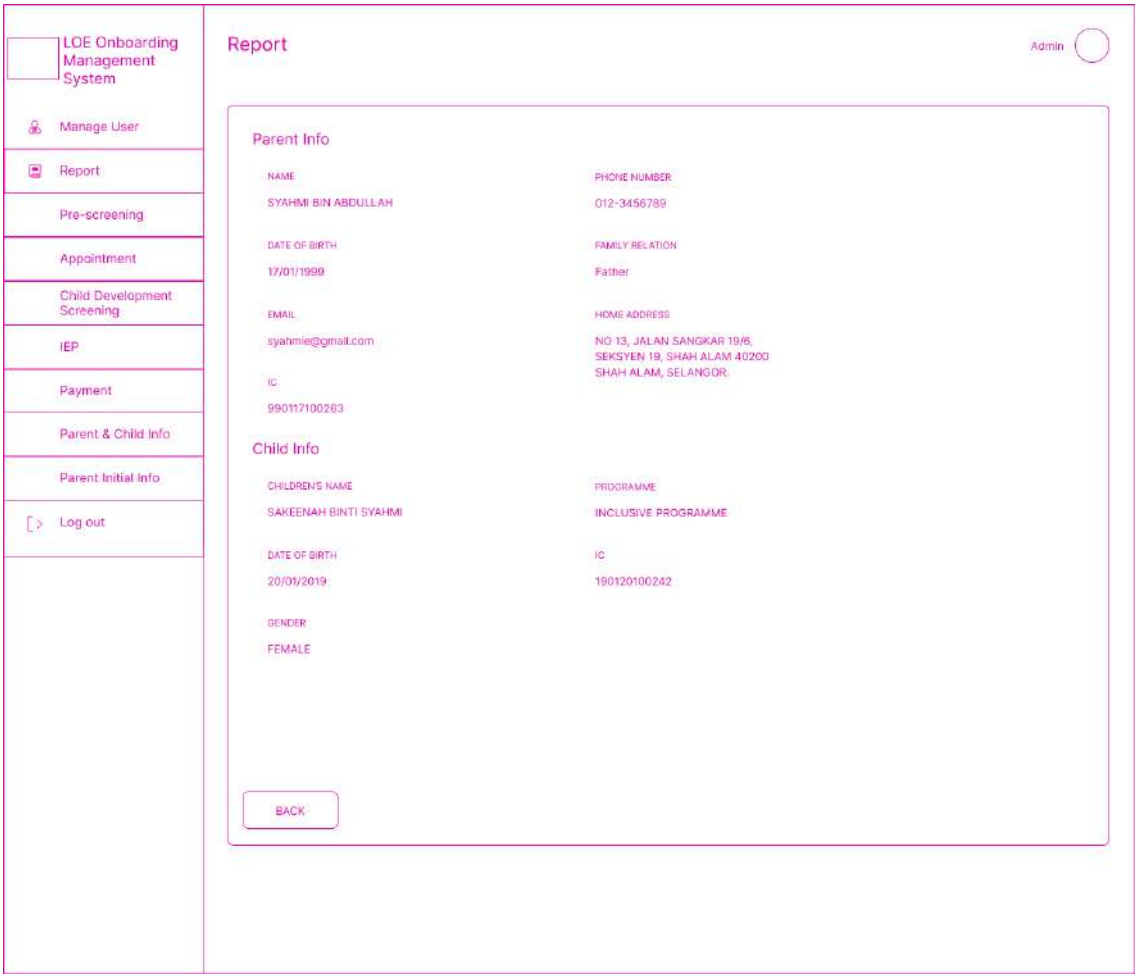


Figure 3.4.75: Parent Child Info details page

- This is the Parent Child Info Report details page.
- Users click the “back” button, it will navigate to the Parent Child Info report page.

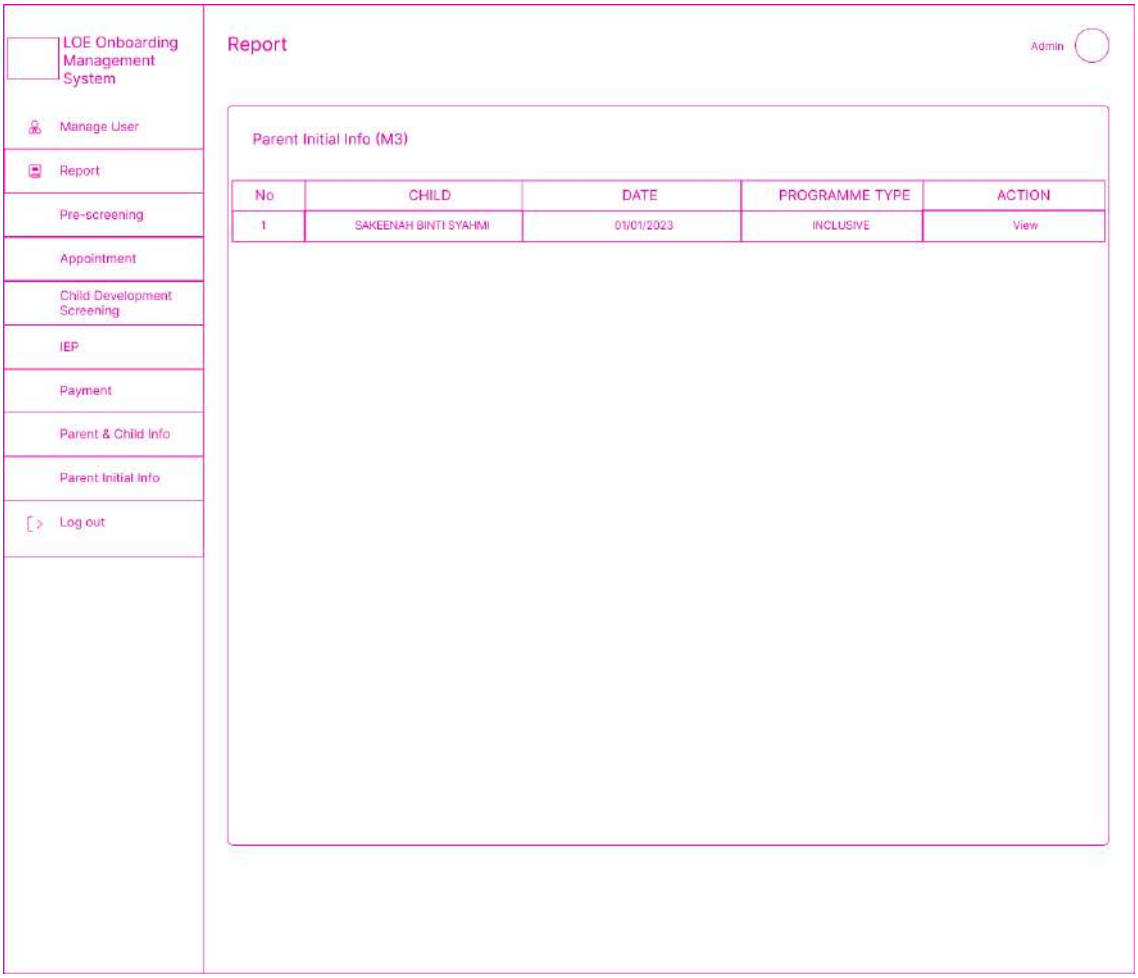


Figure 3.4.76: M3 report page

- This is the Parent Initial Info Report page.
- Users click the “View” link, it will navigate to the M3 report details page.

LOC Onboarding Management System
Manage User
Report
Pre-onboarding
Appointment
Child Development Screening
IEP
Payment
Parent & Child Info
Parent Educ Info
Log out
Report
Fail-fail yang diperlukan:
Sebelum Anak Lahir:
Semasa Melahirkan:
Pengalaman dalam Pantang:
Sejarah Pembesaran anak:
Tingkah laku anak ketika di rumah sendiri:
Penggunaan gajet:
Tingkah laku anak ketika di luar rumah:
BACK

Figure 3.4.77: M3 report details page

- This is the M3 Report details page.
- Users click the “back” button, it will navigate to the Parent Initial Info report page.

Figure 3.4.78: Manage Child IEP report page (Staff)

- This is the IEP Report page for the parent.
- Users click the “View” link, it will navigate to the IEP report details page.

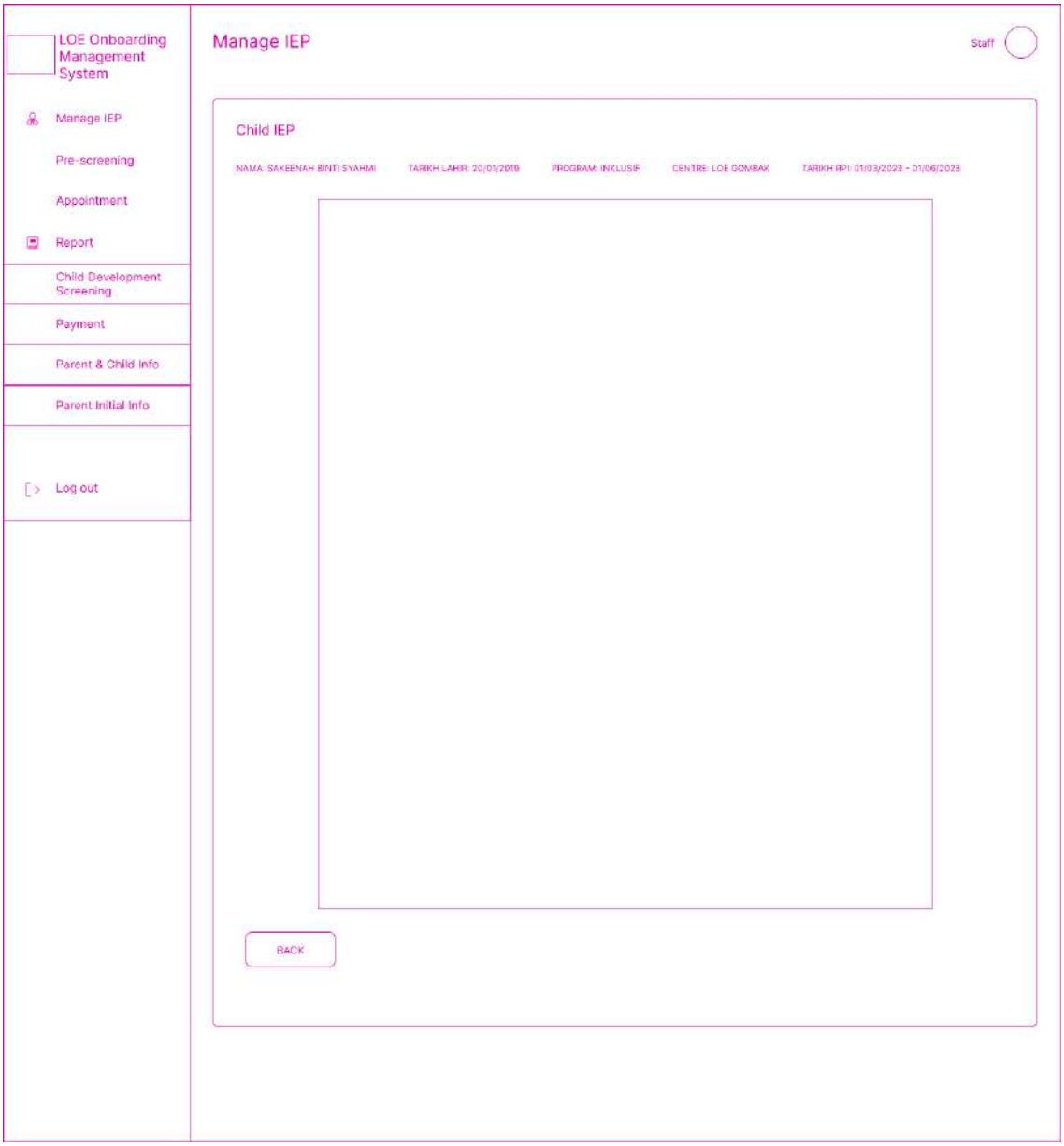


Figure 3.4.79: Manage Child IEP report details page

- This is the IEP Report details page.
- Users click the “back” button, it will navigate to the IEP report page.

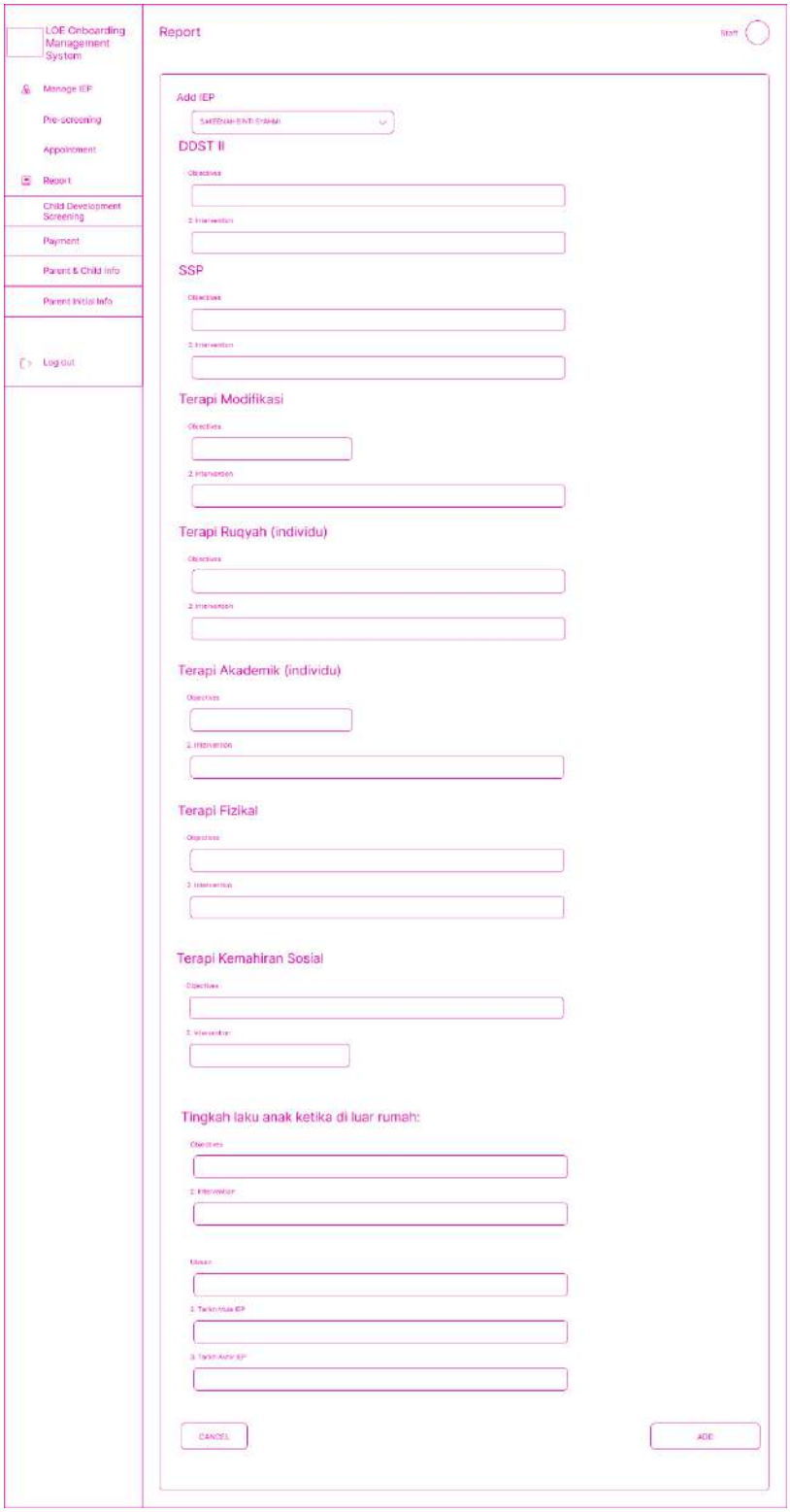


Figure 3.4.80: Add IEP page (staff)

- This is the Add IEP page
- Users must key in all the required fields.
- Users click the “Add” button, it will navigate to the Manage IEP page.

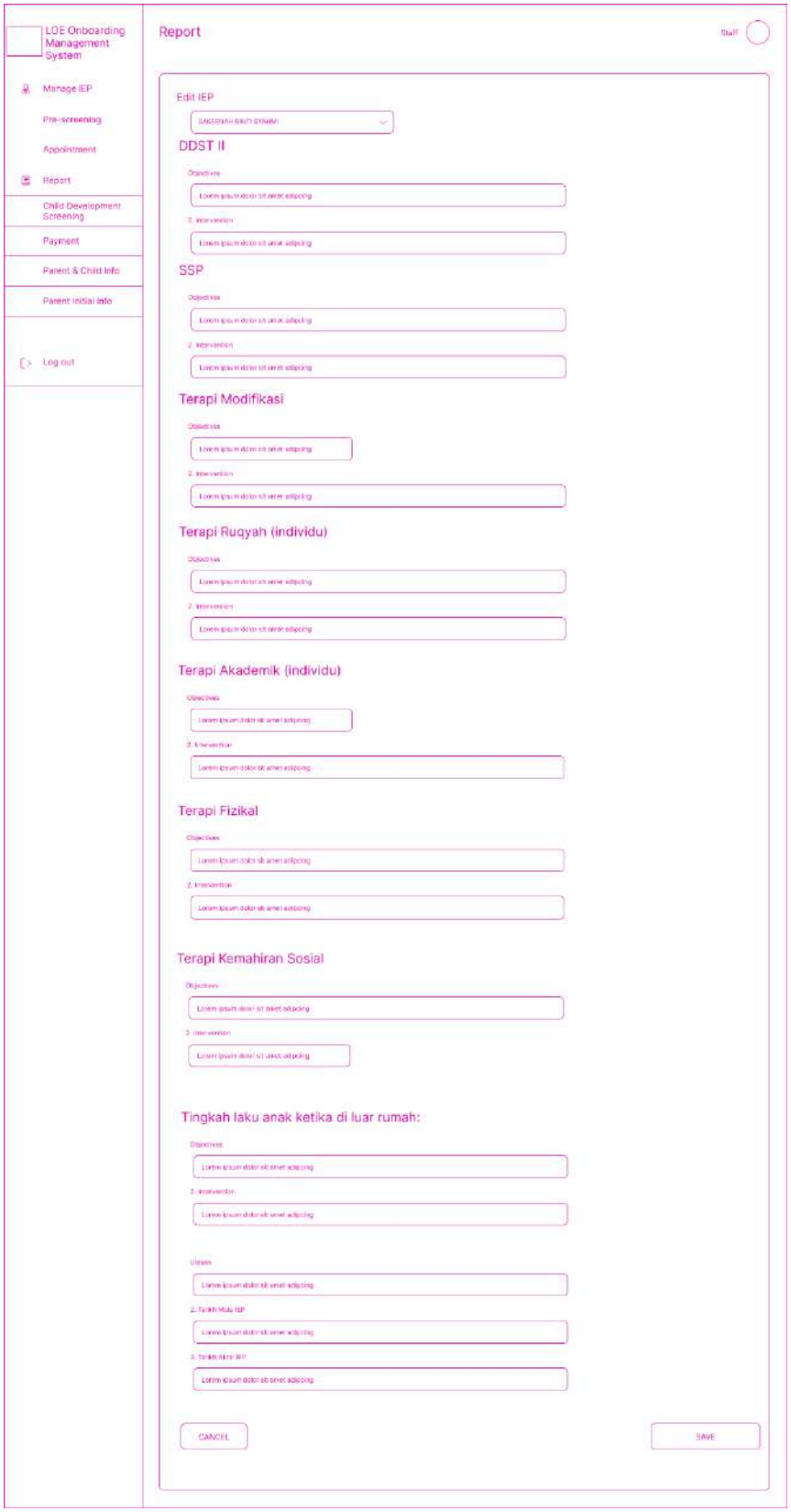


Figure 3.4.81: Edit IEP page

- This is the Edit IEP page
- Users must key in all the required fields.
- Users click the “Save” button, it will navigate to the Manage IEP page.

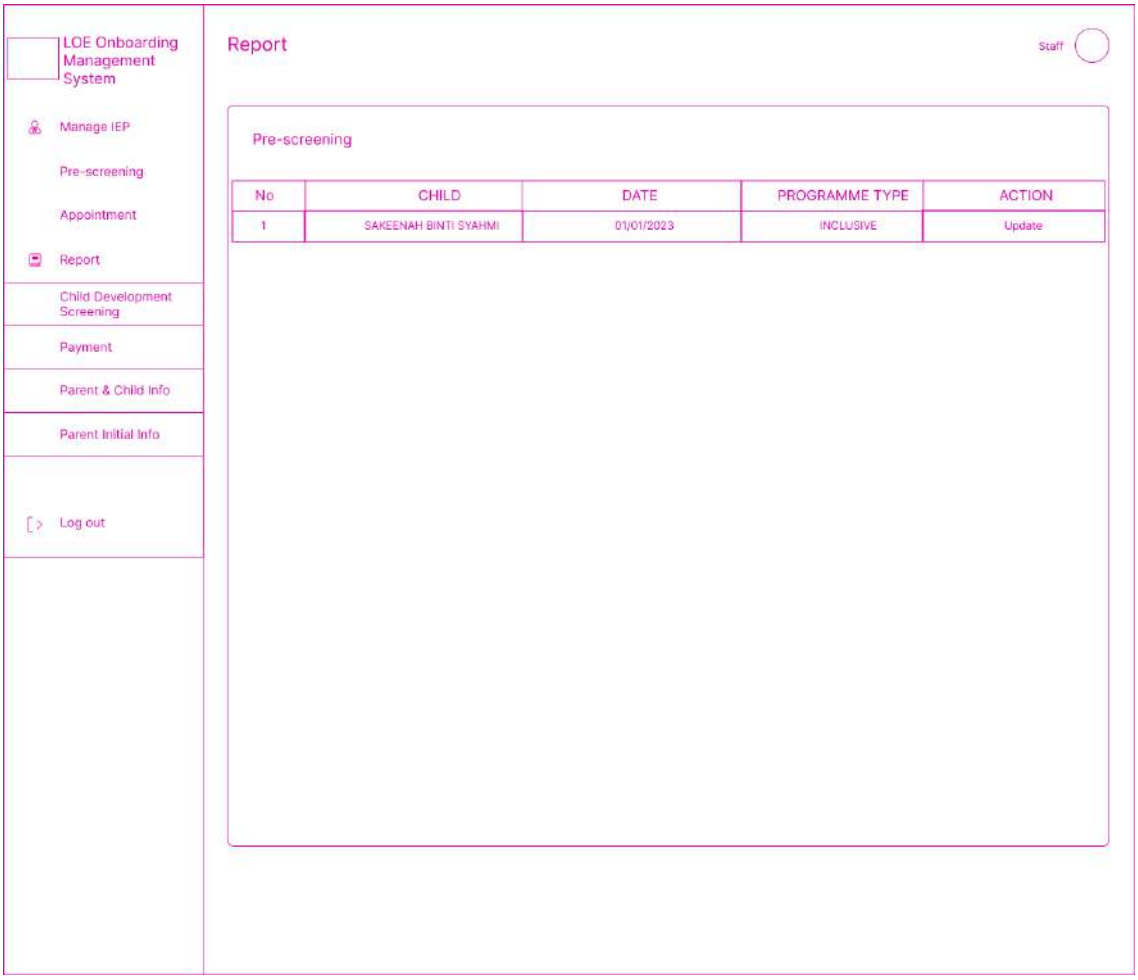


Figure 3.4.82: Pre-screening page (staff)

- This is the Pre-screening page.
- Users click the “View” link, it will navigate to the Pre-screening details page.

Figure 3.4.83: Pre-screening details page (staff)

- This is the Pre-screening details page.
- Users must choose program type and click Update button and it will navigate to Pre-screening page

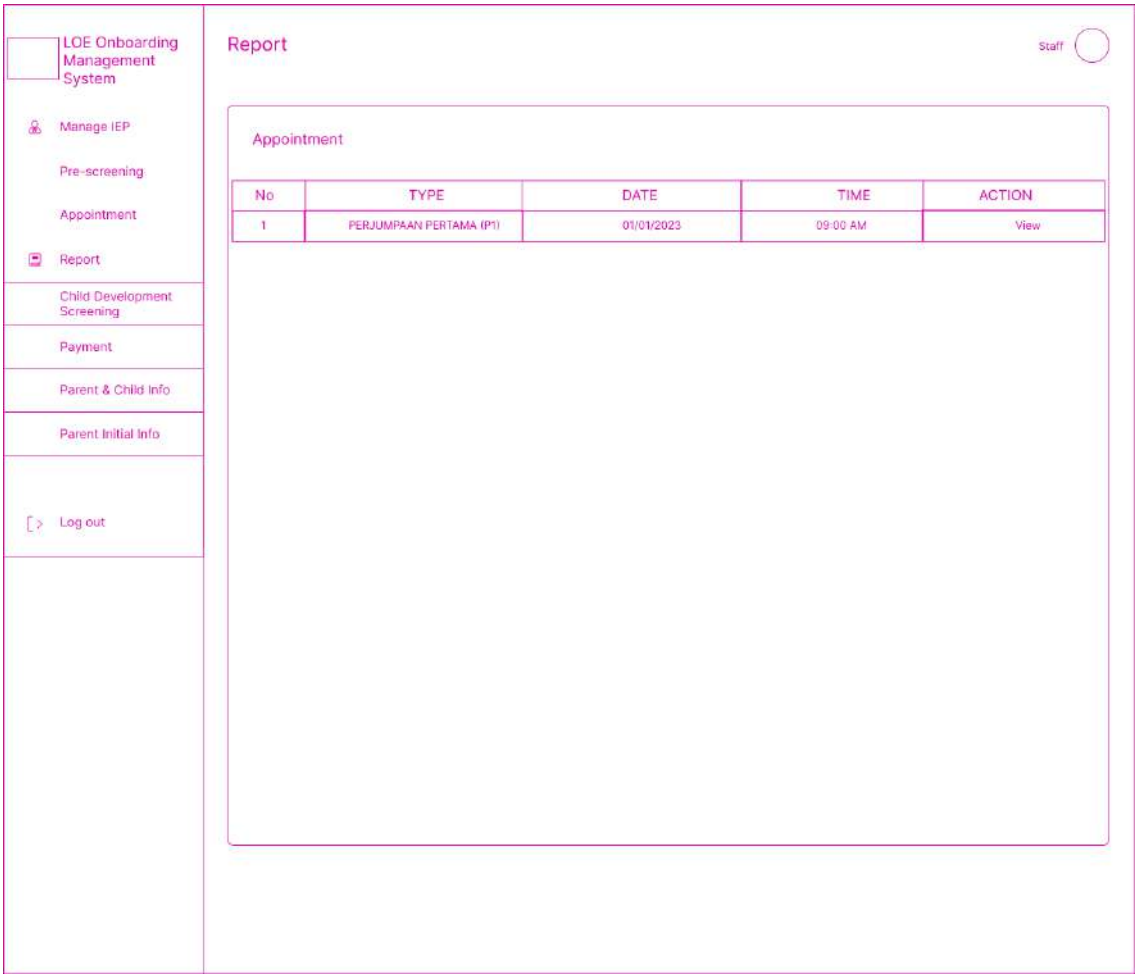


Figure 3.4.84: Appointment page

- This is the appointment page.
- Users click the “View” link, it will navigate to the Appointment details page.

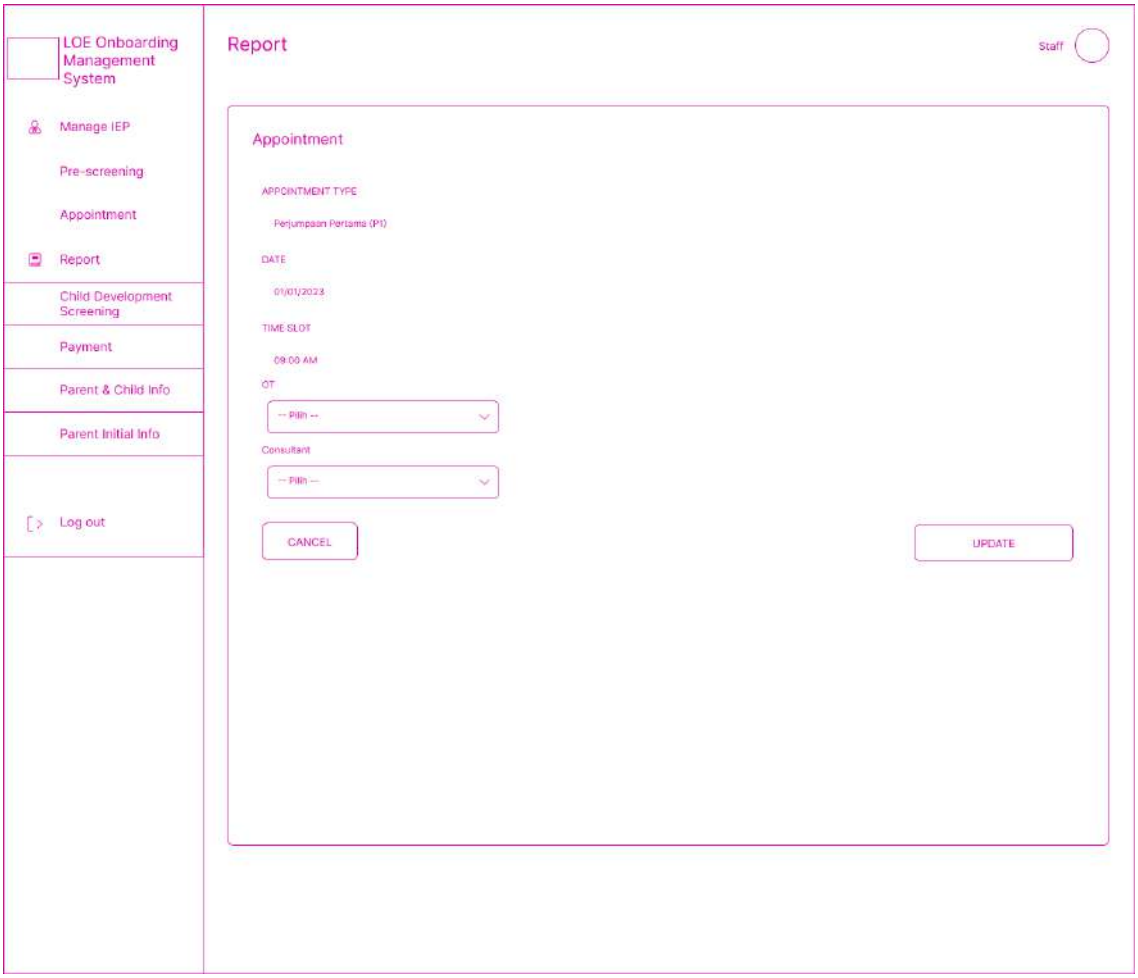


Figure 3.4.85: Appointment details page

- This is the Appointment details page.
- Users must choose person in charge and click Update button and it will navigate to Appointment page

3.5 Data Design

3.5.1 Entity Relationship Diagram (ERD)

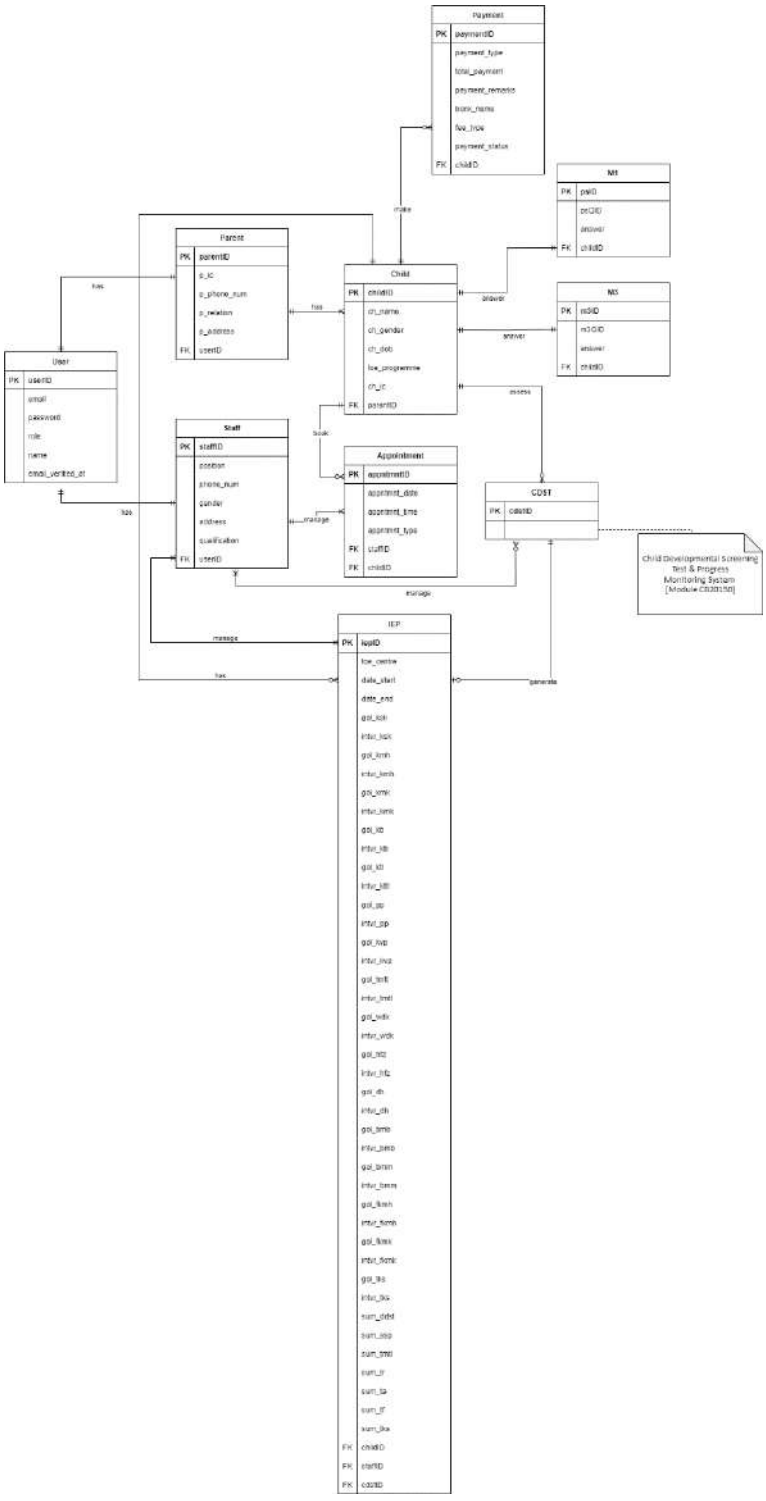


Figure 3.5.1: ERD

3.5.2 Database Dictionary

3.5.2.1 User

Table 3.5.1: User table

Attribute Name	**Type**	**Description**	**Constraint**
userID	int	User ID	PK, Not Null
name	Varchar(255)	User's name	Not Null
email	Varchar(255)	User's email	Not Null
password	Varchar(255)	User's password	Not Null
role	Varchar(255)	User's role (eg, Admin, Staff, Parent)	Not Null
email_verified_at	timestamp	User's email verified datetime	Null

3.5.2.2 Parent

Table 3.5.2: Parent table

Attribute Name	Type	Description	Constraint
parentID	int	Parent ID	PK, Not Null
p_ic	Varchar(255)	Parent's IC number	Not Null, Unique
p_phone_num	Varchar(255)	Parent's phone number	Not Null
p_relation	Varchar(255)	Parent's relation with children	Not Null
p_address	Varchar(255)	Parent's home address	Not Null
userID	int	User ID	FK, Not Null

3.5.2.3 Staff

Table 3.5.3: Staff table

Attribute Name	**Type**	**Description**	**Constraint**
staffID	int	Staff ID	PK, Not Null
ic	Varchar(255)	Staff's IC	Not Null, Unique
gender	Varchar(255)	Staff's gender	Not Null
phone_num	Varchar(255)	Staff's phone number	Not Null
address	Varchar(255)	Staff's home address	Not Null
position	Varchar(255)	Staff's position (Manager, Principal, OT, Consultant)	Not Null
qualification	Varchar(255)	Staff's qualification	Not Null
userID	int	User ID	FK, Not Null

3.5.2.4 Child

Table 3.5.4: Child table

Attribute Name	**Type**	**Description**	**Constraint**
childID	int	Child ID	PK, Not Null
ch_name	Varchar(255)	Child's name	Not Null
ch_gender	Varchar(255)	Child's gender	Not Null
ch_dob	date	Child's date of birth	Not Null
ch_programme	Varchar(255)	Child's programme at LOE centre	Null
ch_ic	Varchar(255)	Child's IC number	Not Null
parentID	int	Parent ID	FK, Not Null

3.5.2.5 Appointment

Table 3.5.5: Appointment table

Attribute Name	**Type**	**Description**	**Constraint**
appntmntID	int	Appointment ID	PK, Not Null
appntmnt_date	date	Appointment's date	Not Null
appntmnt_time	Varchar(255)	Appointment's time	Not Null
appntmnt_type	Varchar(255)	Appointment's type (such as P1 or P2)	Not Null
staffID	int	Staff's ID for person in charge during gathering with parents.	FK, Null
childID	int	Child's ID	FK, Not Null

3.5.2.6 IEP

Table 3.5.6: IEP table

Attribute Name	Type	Description	Constraint
iepID	int	IEP ID	PK, Not Null
loe_centre	Varchar(255)	Loe centre child name	Not Null
date_start	Varchar(255)	Date start assessment	Not Null
date_end	Varchar(255)	Date end assessment	Not Null
gol_ksk	longtext	Gol Kemahiran Sosial Kendiri	Null
intvr_ksk	longtext	Intervensi Kemahiran Sosial Kendiri	Null
gol_kmh	longtext	Gol Kemahiran Motor Halus	Null

intvr_kmh	longtext	Intervensi Kemahiran Motor Halus	Null
gol_kmk	longtext	Gol Kemahiran Motor Kasar	Null
intvr_kmk	longtext	Intervensi Kemahiran Motor Kasar	Null
gol_kb	longtext	Gol Kemahiran Bahasa	Null
intvr_kb	longtext	Intervensi Kemahiran Bahasa	Null
gol_ktl	longtext	Gol Kemahiran Tenaga/Lemah	Null
intvr_ktl	longtext	Intervensi Kemahiran Tenaga/Lemah	Null
gol_pp	longtext	Gol Penapisan Pendengaran	Null
intvr_pp	longtext	Intervensi Penapisan Pendengaran	Null

gol_kvp	longtext	Gol Kepekaan Visual/Pendengaran	Null
intvr_kvp	longtext	Intervensi Kepekaan Visual/Pendengaran	Null
gol_tmtl	longtext	Gol Terapi Modifikasi Tingkah Laku	Null
intvr_tmtl	longtext	Intervensi Terapi Modifikasi Tingkah Laku	Null
gol_wdk	longtext	Gol Wuduk	Null
intvr_wdk	longtext	Intervensi Wuduk	Null
gol_hfz	longtext	Gol Hafazan Al-Quran	Null
gol_hfz	longtext	Intervensi Hafazan Al-Quran	Null
gol_dh	longtext	Gol Doa Harian	Null
intvr_dh	longtext	Intervensi Doa Harian	Null

gol_bmb	longtext	Gol Kemahiran Bahasa Melayu Baca	Null
intvr_bmb	longtext	Intervensi Kemahiran Bahasa Melayu Baca	Null
gol_bmm	longtext	Gol Kemahiran Bahasa Melayu Tulis	Null
intvr_bmm	longtext	Intervensi Kemahiran Bahasa Melayu Tulis	Null
gol_fkmh	longtext	Gol Terapi Fizikal Kemahiran Motor Halus	Null
intvr_fkmh	longtext	Intervensi Terapi Fizikal Kemahiran Motor Halus	Null
gol_fkmk	longtext	Gol Terapi Fizikal Kemahiran Motor Kasar	Null
intvr_fkmk	longtext	Intervensi Terapi Fizikal Kemahiran Motor Kasar	Null

gol_tks	longtext	Gol Terapi Fizikal Kemahiran Sosial	Null
intvr_tks	longtext	Intervensi Terapi Fizikal Kemahiran Sosial	Null
sum_ddst	longtext	Summary DDST II	Null
sum_ssp	longtext	Summary SSP	Null
sum_tmtl	longtext	Summary Terapi Modifikasi Tingkah Laku	Null
sum_tr	longtext	Summary Terapi Ruqyah	Null
sum_ta	longtext	Summary Terapi Akedemik	Null
sum_tf	longtext	Summary Terapi Fizikal	Null
sum_tks	longtext	Summary Terapi Kemahiran Sosial	Null
staffID	int	Staff ID	FK, Not Null

childID	int	Child ID	FK, Not Null
cdstID	int	Development Screening Test ID	FK, Not Null

3.5.2.7 Payment

Table 3.5.7: Payment table

Attribute Name	**Type**	**Description**	**Constraint**
paymentID	int	Payment ID	PK, Not Null
payment_type	Varchar(100)	Payment type	Not Null
fee_type	Varchar(100)	Payment type	Not Null
total_payment	double	Total payment need to be paid	Not Null
payment_remarks	Varchar(255)	Payment remarks	Null
payment_status	Varchar(255)	Payment status (such as Paid)	Not Null
bank_name	Varchar(255)	Bank's name	Not Null
childID	int	Child ID	FK, Not Null

3.5.2.8 M1 (Pre-screening)

Table 3.5.8: Pre-screening table

Attribute Name	Type	Description	Constraint
id	int	Child ID	PK, Not Null
psQID	Varchar(255)	Question ID of pre-screening questionnaire	Not Null
answer	Varchar(255)	Answer of pre-screening questionnaire	Not Null
child_id	int	Child ID	FK, Not Null

3.5.2.9 M3

Table 3.5.9: M3 table

Attribute Name	Type	Description	Constraint
m3ID	int	Child ID	PK, Not Null
m3QID	Varchar(255)	Question ID of M3 questionnaire	Not Null
answer	Varchar(255)	Answer of M3 questionnaire	Null
childID	int	Child ID	FK, Not Null

3.6 Proof of Initial Concept

3.6.1 Interface Design

Refer to Appendix A.

3.7 Testing/Validation Plan

The testing plan describes the activities to ensure the Special Needs Onboarding Management system's functionality and any errors or problems found in the system for improvement. The table below shows the testing plan.

Table 3.7.1: Testing Plan

No.	Module	Activities	Expected Result	Actual Result	Status		Comments
					Pass	Fail	
1	Register	Enter valid email format	Account successfully registered.				
		Enter invalid email format	Error message displayed				
		Click register button	Account successfully registered and receive email verification.				

		Enter valid parent's name format	Account successfully registered.				
		Enter invalid parent's name format	Error message displayed				
		Enter different value in confirmation password input field with password input field	Error message displayed				
		Enter same value in confirmation password input field with password input field	Account successfully registered.				
		Enter valid IC number format	Account successfully registered.				

		Enter invalid IC number format	Error message displayed				
		Enter valid phone number format	Account successfully registered.				
		Enter invalid phone number format	Error message displayed				
		Enter input value for strings in any case	The input field will show uppercase only				
2	Login	Enter valid email address	User successfully logged in.				
		Enter invalid email address	Error message displayed				

		Enter valid password	User successfully logged in.				
		Enter invalid password	Error message displayed				
		Login without an account	Error message displayed				
3	Manage User System	Enter valid email format	Account successfully registered.				
		Enter invalid email format	Error message displayed				
		Enter valid user’s name format	Account successfully added.				

		Enter invalid user's name format	Error message displayed				
		Enter valid user's IC format	Account successfully added.				
		Enter invalid user's IC format	Error message displayed				
		Enter valid user's qualification format	Account successfully added.				
		Enter invalid user's qualification format	Error message displayed				
		Enter valid phone number format	Account successfully registered.				

		Enter invalid phone number format	Error message displayed				
		Delete respective user	Correct user deleted and message displayed				
4	Manage Account and Profile	Enter valid name format	Successfully edited and message box displayed				
		Enter invalid name format	Error message displayed				
		Enter valid IC number format	Successfully edited and message box displayed				
		Enter invalid IC number format	Error message displayed				

		Enter valid phone number format	Successfully edited and message box displayed				
		Enter invalid phone number format	Error message displayed				
		Enter different value in confirmation password input field with password input field	Error message displayed				
		Enter same value in confirmation password input field with password input field	Successfully edited and message box displayed				
		Enter valid email address format	Successfully edited and message box displayed				

		Enter invalid email address format	Error message displayed				
		Enter valid child name (Parent)	Successfully edited and message box displayed				
		Enter invalid child name format (Parent)	Error message displayed				
		Enter valid child IC format (Parent)	Successfully edited and message box displayed				
		Enter invalid child IC format (Parent)	Error message displayed				
		Delete respective child (Parent)	Correct child deleted and message displayed				

5	Manage Pre-Screening	Display respective child name	Answer successfully store with correct id				
		Update LOE child programme (Manager, Principal, OT)	Successfully updated and message box displayed				
		Notify user after click on Submit button	Data stored and User received notification				
6	Manage Payment	Display correct value of total payment	Successfully pay				
		Notify user after click on Pay button	Data stored and User received notification				
7	Manage Appointment	Notify user after click on Book button	Data stored and User received notification				

		Notify user after click on Update button	Data updated and User received notification				
		View appointment list (Parent)	Display correct child data under parent id				
		View appointment list (Admin)	Display correct all child data that book appointment				
		Cancel respective appointment (Admin)	Correct data deleted and message displayed				
		Cancel respective appointment (Parent)	Correct data deleted and message displayed				
		Display respective child name	Answer successfully store with correct id				

		Display respective PIC name (Manager, Principal and OT)	Data successfully store with correct id				
8	Manage Child IEP	Add IEP	Data added into database				
		Edit IEP	Data updated in database and message displayed				
		Delete IEP	Data deleted from database and message displayed				
9	Manage Report	View Pre-Screening (M1)	Data displayed				
		View Child Development Screening	Data displayed				

		View IEP	Data displayed				
		View Appointment	Data displayed				
		View Payment	Data displayed				
		View Parent & Child Info	Data displayed				
		View Parent Initial Info (M3)	Data displayed				
10	Manage Parent Initial Info (M3)	Notify user after click on Submit button	Data stored and User received notification				
		Display respective child name	Answer successfully store with correct id				

6	Manage Child Development (External system)	-	-	-	-	-	-

Verified by:

_____

Name :

Date : 09/06/2023

3.8 Potential Use of Proposed Solution

LOE has traditionally handled this procedure manually, which is no longer practical. In addition, it will take more time than necessary, and there'll be plenty of potential for confusion and miscommunication. In order to help the LOE staff and parents manage the onboarding process for new students more effectively, this project aims to develop a Special Needs Children Onboarding Management System to replace the current manual procedure.

Special Needs Children Onboarding Management System can assist LOE staff in managing the onboarding process of special needs children quickly and smoothly in terms of assessment of children's development, management of children's individual education plans and scheduling meetings between parents, children with LOE staff and experts such as OT. With the existence of this system, it will be easier for LOE staff to save and locate vital data promptly and securely.

In addition, this system allows parents to monitor their children's development during the onboarding process to see if their child can adapt to the new environment at the centre. Through this system, it is not only easy for parents to make payments but also to schedule meetings. Also, the notification function included in the Special Needs Children Onboarding Management System makes it possible for parents to obtain information immediately.

3.9 Gantt Chart

				Sprint 1		Sprint 2		Sprint 3		Sprint 4	MidTerm Break	Sprint 5		Sprint 6		Sprint 7			Presentation/Study Week
				Week 1	Week 2	Week 3	Week 4	Week 5	Week 6	Week 7	Week 8	Week 9	Week 10	Week 11	Week 12	Week 13	Week 14	Week 15	Week 16
				2022	2022	2022	2022	2022	2022	2022	2022	2022	2022	2022/2023	2023	2023	2023	2023	2023
Task	START	END	DURATION	17.10 - 23.10	24.10 - 30.10	31.10 - 06.11	07.11 - 13.11	14.11 - 20.11	21.11 - 27.11	28.11 - 04.12	05.12 - 11.12	12.12 - 18.12	19.12 - 25.12	26.12 - 01.01	02.01 - 08.01	09.01 -15.01	16.01 - 22.01	23.01 - 29.01	30.01 - 05.02
PSM I																			
Project Requirement																			
Software Requirements	14/10/2022	16/11/2022	33																
Functional Requirements	14/10/2022	16/11/2022	33																
Non-Functional Requirements	14/10/2022	16/11/2022	33																
Proposed Design																			
Context Diagram	17/11/2022	20/11/2022	3																
Use Case Diagram	20/11/2022	27/11/2022	7																
Use Case Description	27/11/2022	10/12/2022	13																
Activity Diagram	17/11/2022	20/11/2022	3																
Flowchart	17/11/2022	20/11/2022	3																
Interface	11/12/2022	18/1/2023	38																
Data Design																			
Entity Relationship Diagram (ERD)	18/12/2022	24/1/2022	6																
Database Dictionary	23/12/2022	4/1/2023	12																
Proof of Initial Concept																			
Prototype	11/12/2022	18/1/2023	38																
Testing Planning																			
Testing Plan	11/1/2023	15/1/2023	4																
Others																			
Research and Setup Framework	11/1/2023	5/2/2023	25																

Figure 3.9.1: PSM I Gantt Chart

				Cuti	Cuti	Cuti	PSM II						MidTerm Break								Study Week	
				Week 17	Week 18	Week 19	Week 1	Week 2	Week 3	Week 4	Week 5	Week 6	Week 7	Week 8	Week 9	Week 10	Week 11	Week 12	Week 13	Week 14	Week 15	Week 16
				2023	2023	2023	2023	2023	2023	2023	2023	2023	2023	2023	2023	2023	2023	2023	2023	2023	2023	2023
Task	START	END	DURATION	20.02 - 26.02	27.02 - 05.03	06.03 - 12.03	13.03 - 19.03	20.03 - 26.03	27.03 - 02.04	03.04 - 09.04	10.04 - 16.04	17.04 - 23.04	24.04 - 30.04	01.05 - 07.05	08.05 - 14.05	15.05 - 21.05	22.05 - 28.05	29.05 - 04.06	05.06 - 11.06	12.06 - 18.06	19.06 - 25.06	26.06 - 02.07
PSM II																						
Development																						
Module 1: Register	26/2/2023	11/3/2023	13																			
Module 2: Login	26/2/2023	11/3/2023	13																			
Module 3: Manage Profile	11/3/2023	25/3/2023	13																			
Module 4: Manage User System	11/3/2023	25/3/2023	13																			
Module 5: Manage Child Report	26/3/2022	8/4/2022	13																			
Module 6: Pre-Screening	26/3/2023	8/4/2023	13																			
Module 7: Manage Payment	9/4/2023	22/4/2023	13																			
Module 8: Manage Parent Initial Info MS	9/4/2023	22/4/2023	13																			
Module 9: Manage Child MS	23/4/2023	20/5/2023	27																			
Module 10: Manage Appointment	7/5/2023	20/5/2023	13																			
Testing																						
System Testing and Maintenance	21/5/2023	10/6/2023	20																			

Figure 3.9.2: PSM II Gantt Chart

CHAPTER 4

RESULTS AND DISCUSSION

4.1 Introduction

This chapter will discuss the system implementation, results and discussion of the Special Needs Children On-boarding Management System. This system includes ten modules: Registration, Login, Manage User, Manage Profile, Manage Pre-screening, Manage Appointment, Manage Payment, Manage IEP, Manage Parent Initial Info (M3) and Manage Report. This system will be developed using the Laravel web application framework and MySQL. Finally, the results and discussions will cover the testing of the system and discussion on the interpretation of the result for future development.

4.2 Implementation

This section describes how the system was implemented, includes the method, tools and the software used in developing the system.

4.2.1 Development tools

Table 4.2.1:Development tools used and descriptions of the tools

Tools	Images	Descriptions
Laravel		Laravel is an open-source web application framework for developing web applications using PHP. Laravel provides many tools and features that make building scalable, maintainable, and secure web applications easier. Some of the key features of Laravel include a robust routing system and a powerful templating engine called Blade.
VS Code		Visual Studio Code is a free and open-source code editor developed by Microsoft. It is designed to be a lightweight, fast, and flexible developer tool. VS Code provides various features to support development across multiple programming languages, including syntax highlighting, code completion, debugging, source control integration, and more. It also includes a built-in terminal for running commands and scripts and a rich ecosystem of extensions that can be used to customize the editor and add additional functionality.

XAMPP		XAMPP comprises an Apache HTTP server, MySQL database, and PHP programming language. Developers may easily set up a local web server for testing and development with XAMPP. Web developers use XAMPP to build and test web apps locally before deploying them.
phpMyAdmin		phpMyAdmin is a free and open-source web-based administration tool for managing MySQL databases. It is written in PHP and provides a graphical user interface (GUI) for database administration tasks like creating databases, tables, and users, managing database content, and executing SQL queries.
Mailtrap		Mailtrap is a simulated SMTP server used to test and stage email systems. It enables developers to evaluate the email functionality of their application (such as email verification) without sending emails to actual users. Mailtrap intercepts email messages sent by a developer's application and displays them in a web-based inbox. The developer can then examine the email, verify the correctness of the message's content, and examine its formatting.

4.2.2 Setup Laravel Environment

The Special Needs On-boarding Management System is developed using Laravel framework. Below are the steps to setup the environment:

1. Download and install XAMPP.
2. Install composer, which is a PHP package manager that is used to manage Laravel dependencies.
3. Open VS code and its terminal to install Laravel and create Laravel project.
4. Setup database name at ".env.example" file and copy Copy the ".env.example" file to ".env" file.
5. Start Apache and MySQL in XAMPP and click "Admin" button at MySQL to open the phpMyAdmin to create a new database.
6. Run the migration to manage the tables and attributes or fields involved.
7. Create the necessary controllers and models for processing HTTP requests and generating responses, and for manage system database.
8. Create the necessary interfaces to create and display the interface of the system.
9. Create the necessary routes to map the URLs to specific controllers and actions.
10. Start the development server by using the command "php artisan serve" at the terminal in VS code.

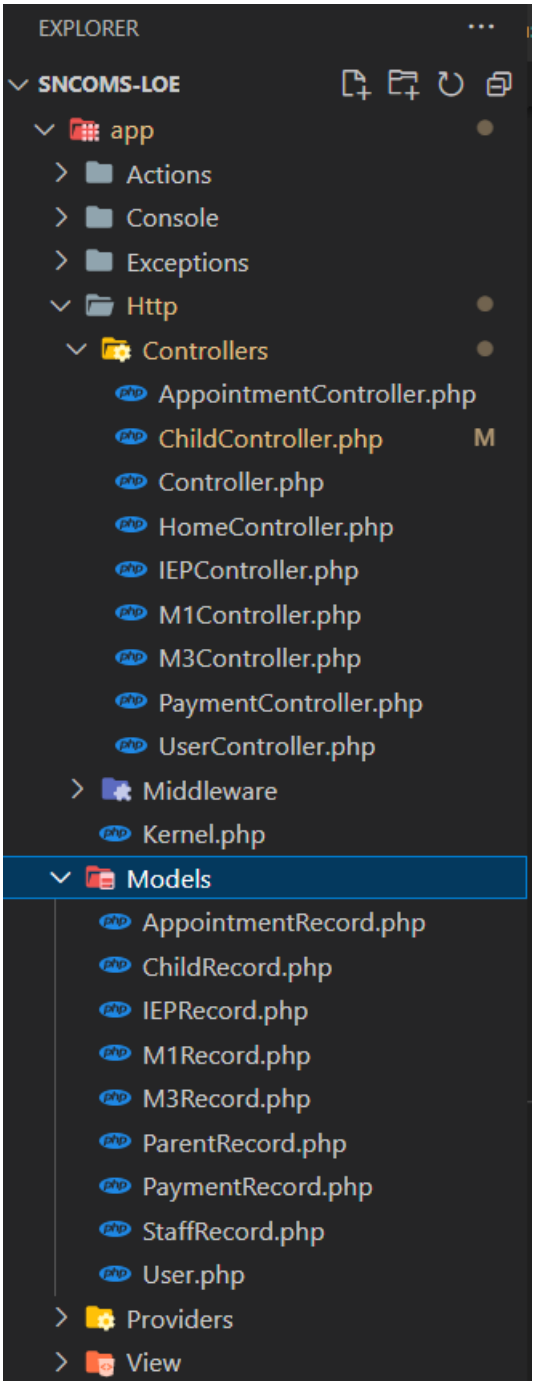


Figure 4.2.1: MVC (Model and Controller)

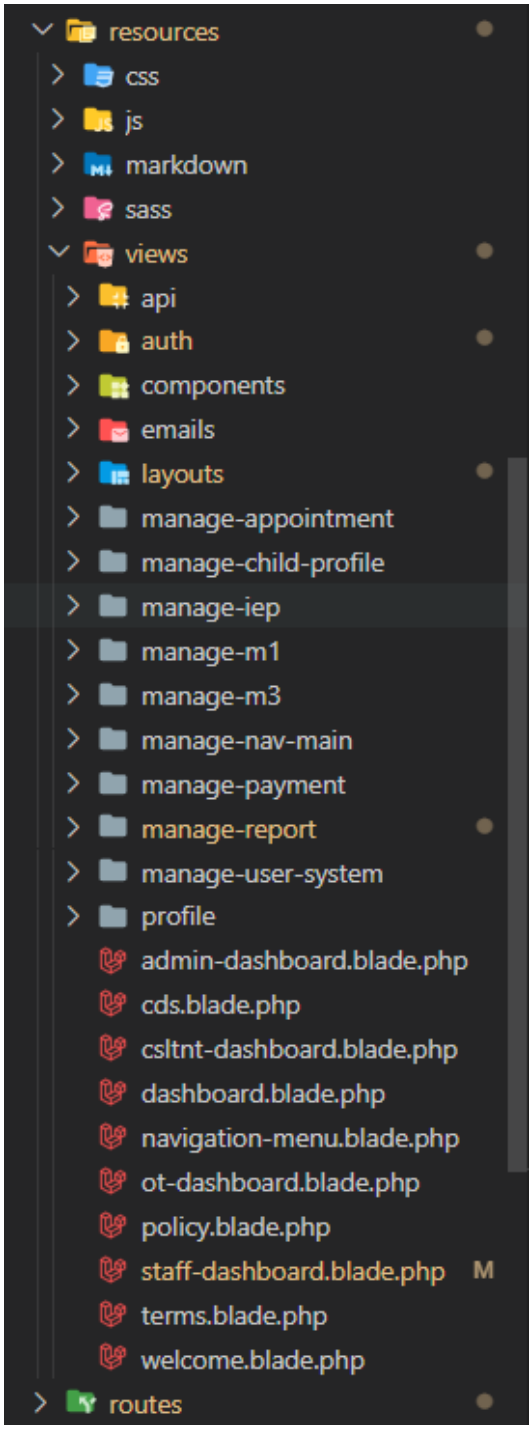


Figure 4.2.2: MVC (Views)

4.2.3 System Interface and Functionality Development

4.2.3.1 Registration

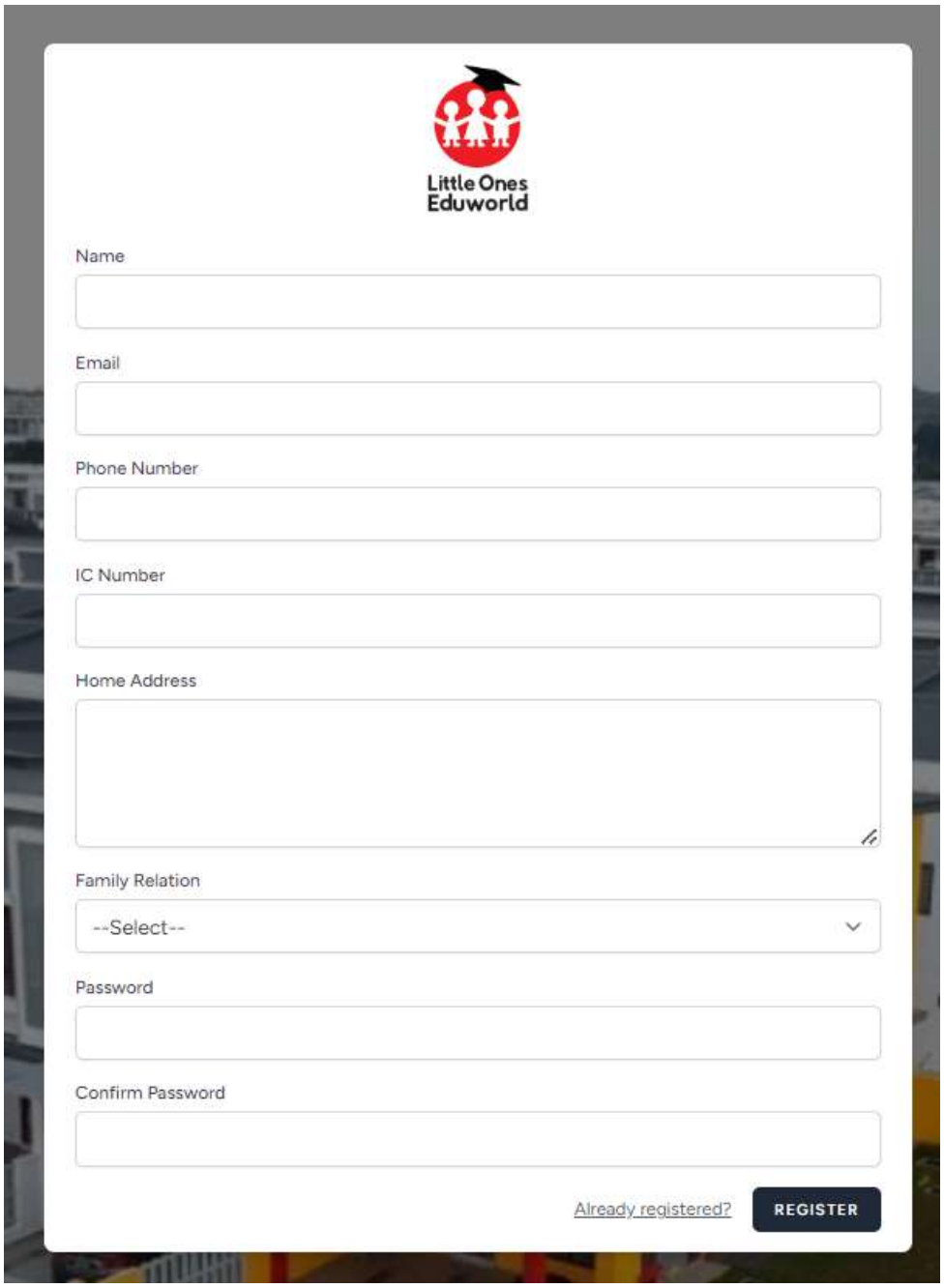


Figure 4.2.3: Registration page

The user needs to key in all the information such as name, email, phone number, IC number, home address, family relation and password before registering a new account as

a parent. After clicking on the “Register” button (as shown in Figure 4.2.3), the system will redirect the user for performing email verification as shown in Figure 4.2.4.

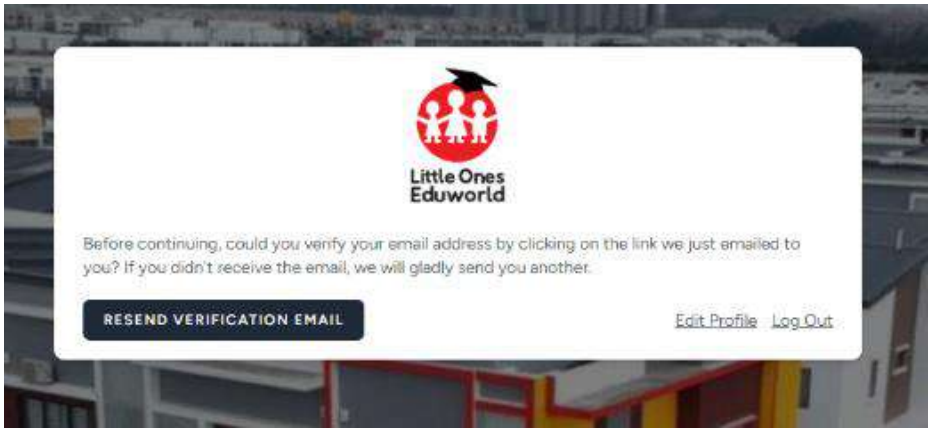


Figure 4.2.4: Email verification interface

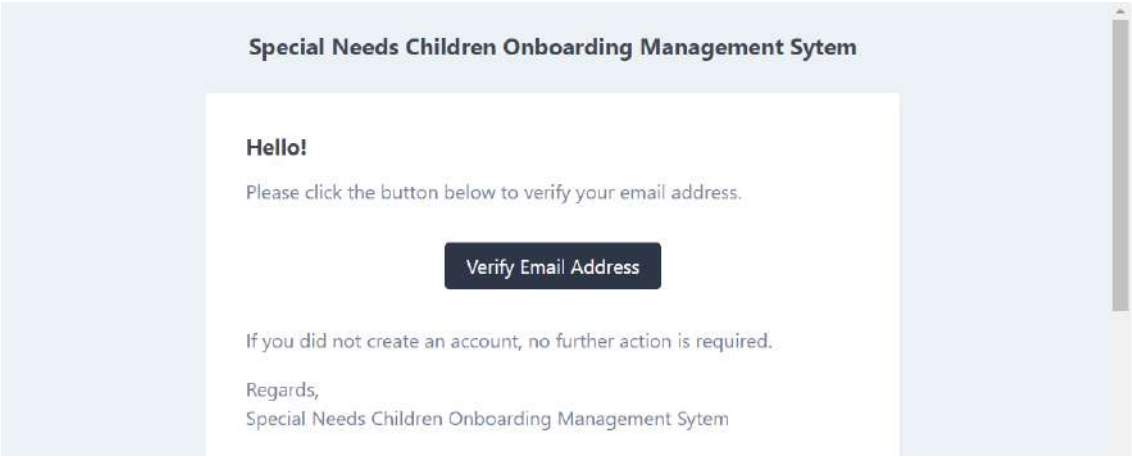


Figure 4.2.5: Received email for verification

Then the user will receive an email for verification. Users need to click on the “Verify Email Address” button in the email as shown in Figure 4.2.5, then the system will redirect the user to the parent dashboard.

4.2.3.2 Login

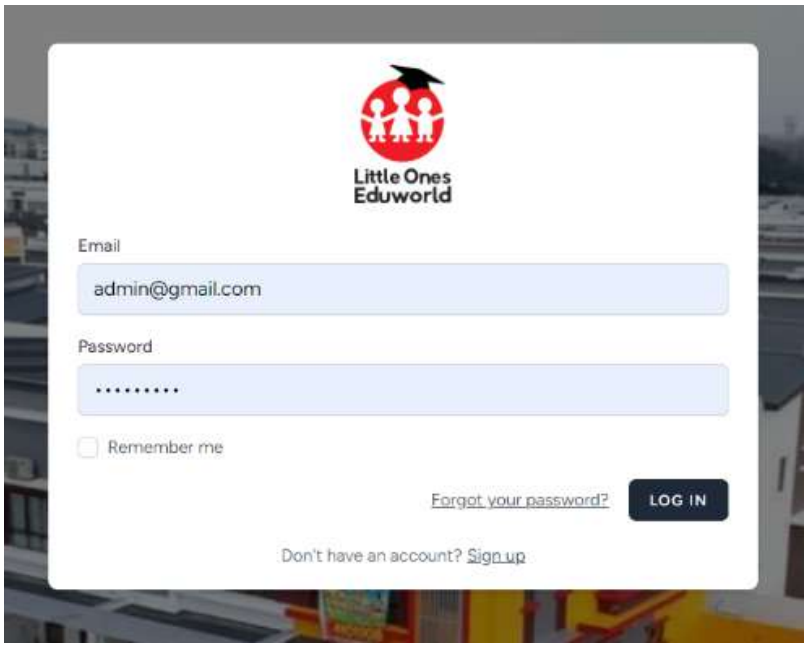


Figure 4.2.6: Login interface

The users need to key in the email and password before clicking on “Login” button. Then the system will redirect the user to the user’s dashboard. If the user (parent) do not have an account, they can click on the “Sign up” link (as shown in Figure 4.2.6) to register an account.

4.2.3.3 Manage Profile

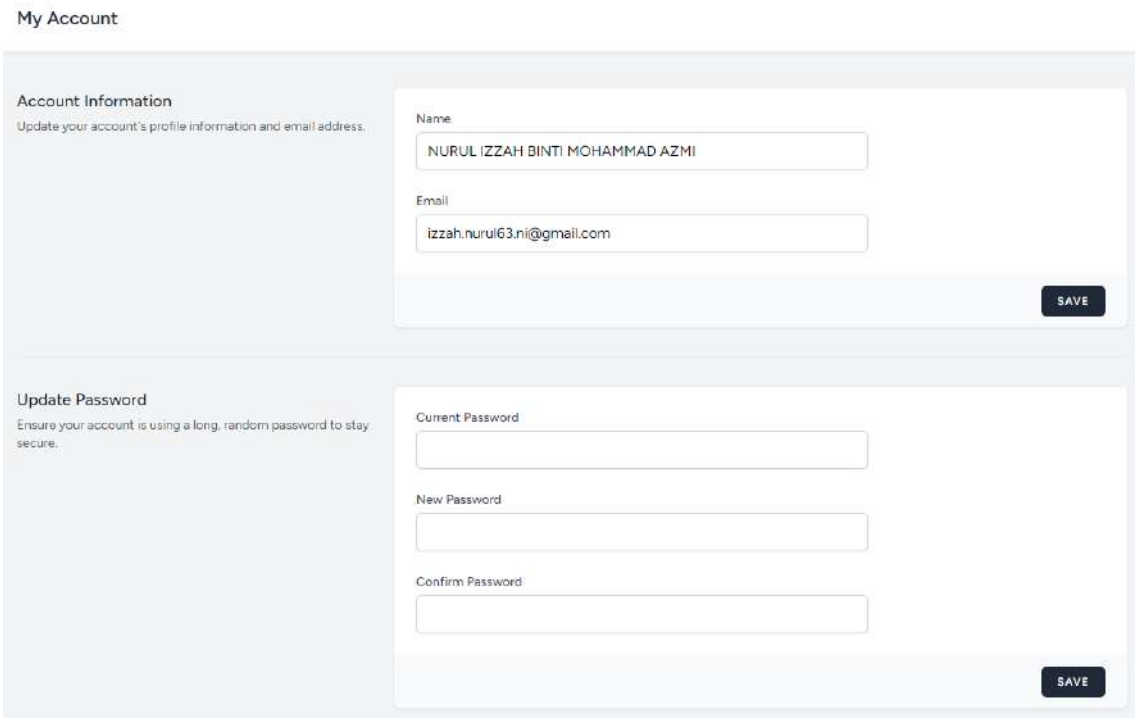


Figure 4.2.7: Manage My Account interface

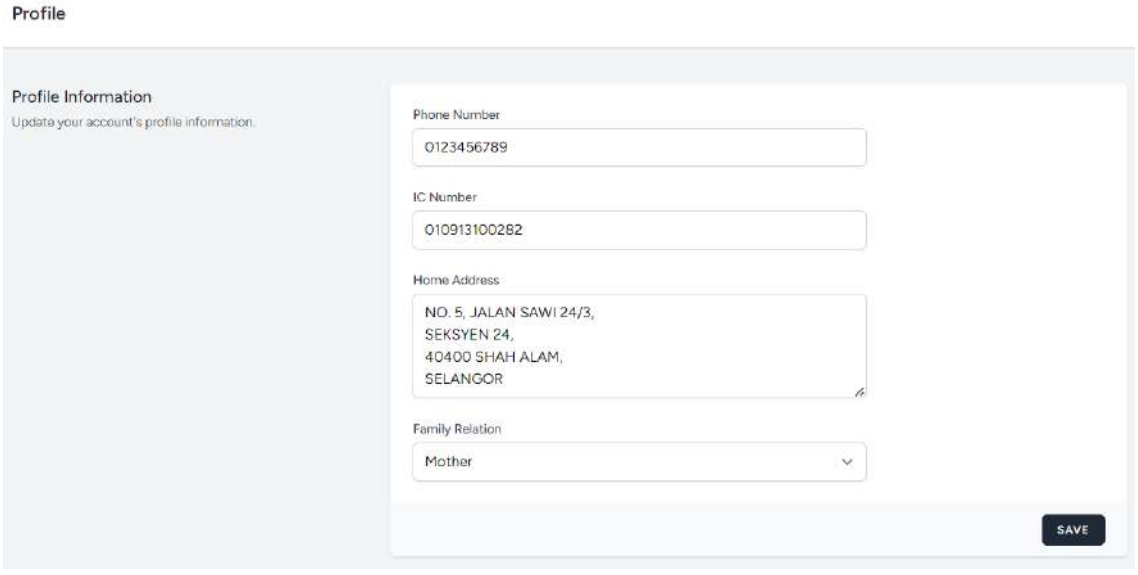


Figure 4.2.8: Manage My Profile interface

As shown in Figure 4.2.7 and Figure 4.2.8, users able to edit their profile information and click on “Save” button to save the changes.

Figure 4.2.9: Manage Child Profile interface

Figure 4.2.10: Add Child Profile interface

The users can click on “Add Child” button, as shown in Figure 4.2.9 to add child profile. Then the system will redirect to the Add Child page and the users need to key in all the information needed as shown in Figure 4.2.10.

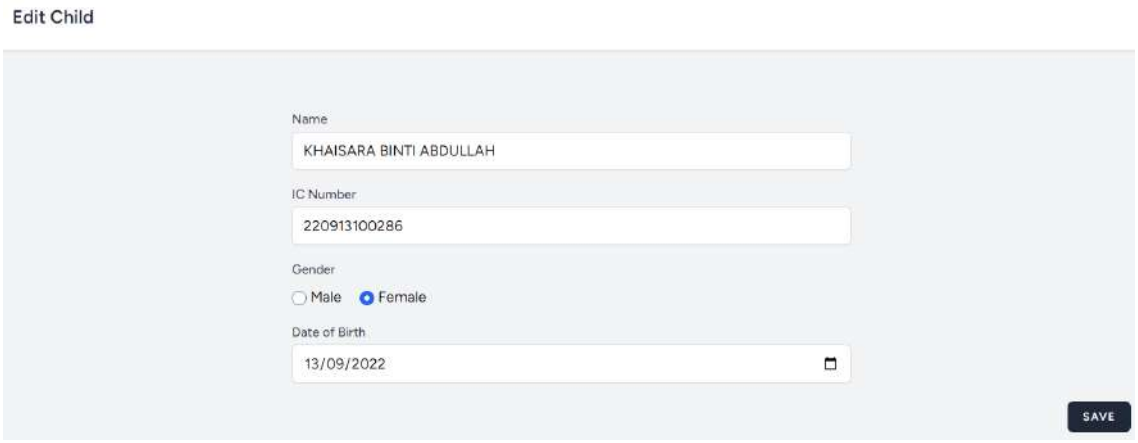


Figure 4.2.11: Edit Child Profile interface

The users also can edit the profile by clicking on the "Edit" link as shown in Figure 4.2.9, then the system will redirect to the Edit Child Profile page. Click on "Save" button to save the changes.

4.2.3.4 Manage User System

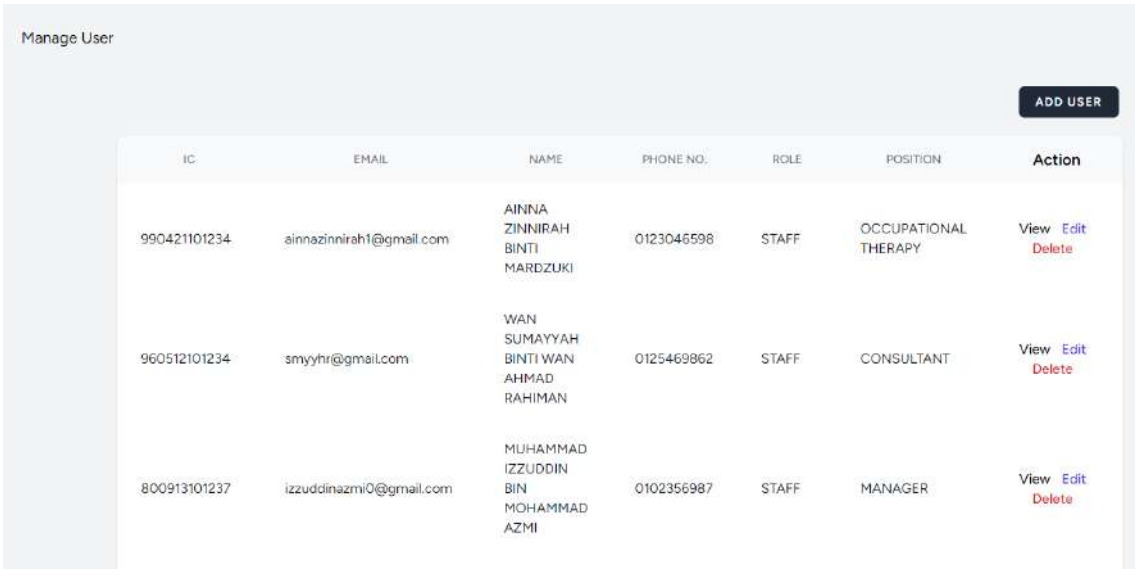


Figure 4.2.12: Manage User System interface

The users can click on "Add User" button, as shown in Figure 4.2.12 to add user account. Then the system will redirect to the Add User page and admin need to key in all

the information needed as shown in Figure 4.2.13. Admin also can delete the users account by clicking on the “Delete” link (as shown in Figure 4.2.12).

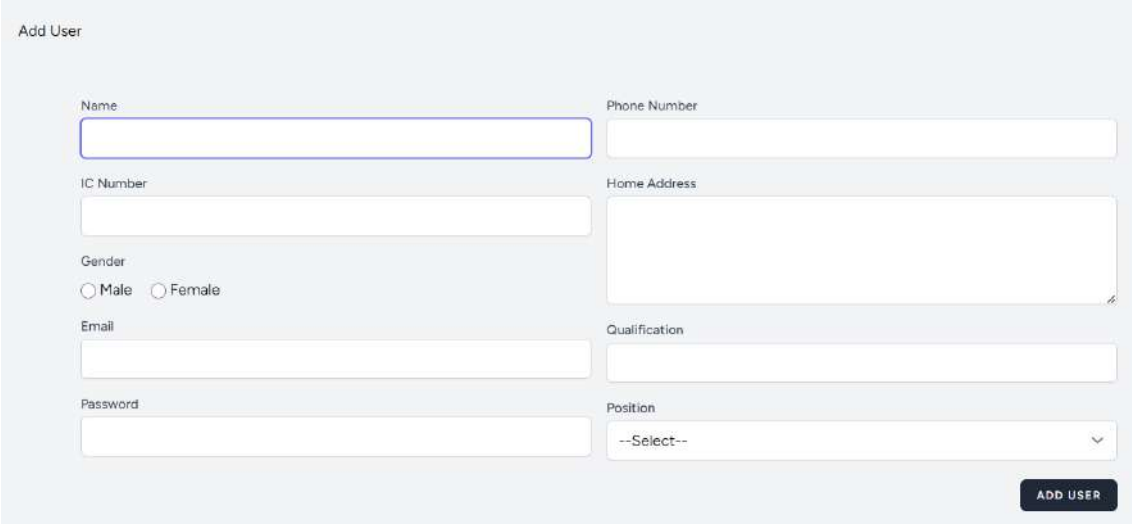


Figure 4.2.13: Add User interface

Figure 4.2.14: Edit User interface

The users also can edit the profile by clicking on the “Edit” link as shown in Figure 4.2.14, then the system will redirect to the Edit User Profile page. Click on “Save” button to save the changes.

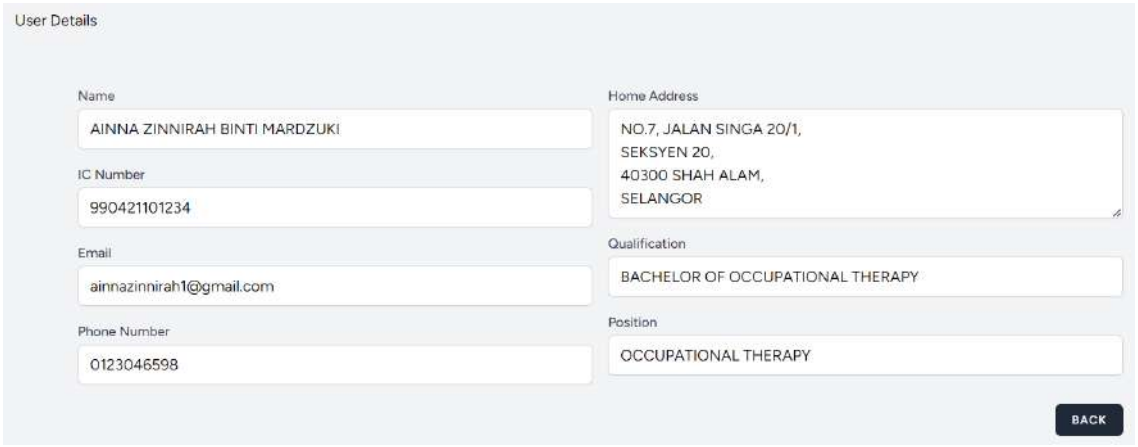


Figure 4.2.15: View User details interface

The users also can view the profile by clicking on the “View” link as shown in Figure 4.2.12, then the system will redirect to the View User Profile Details page. Click on “Back” button to go to previous page.

4.2.3.5 Manage Pre-screening (M1)

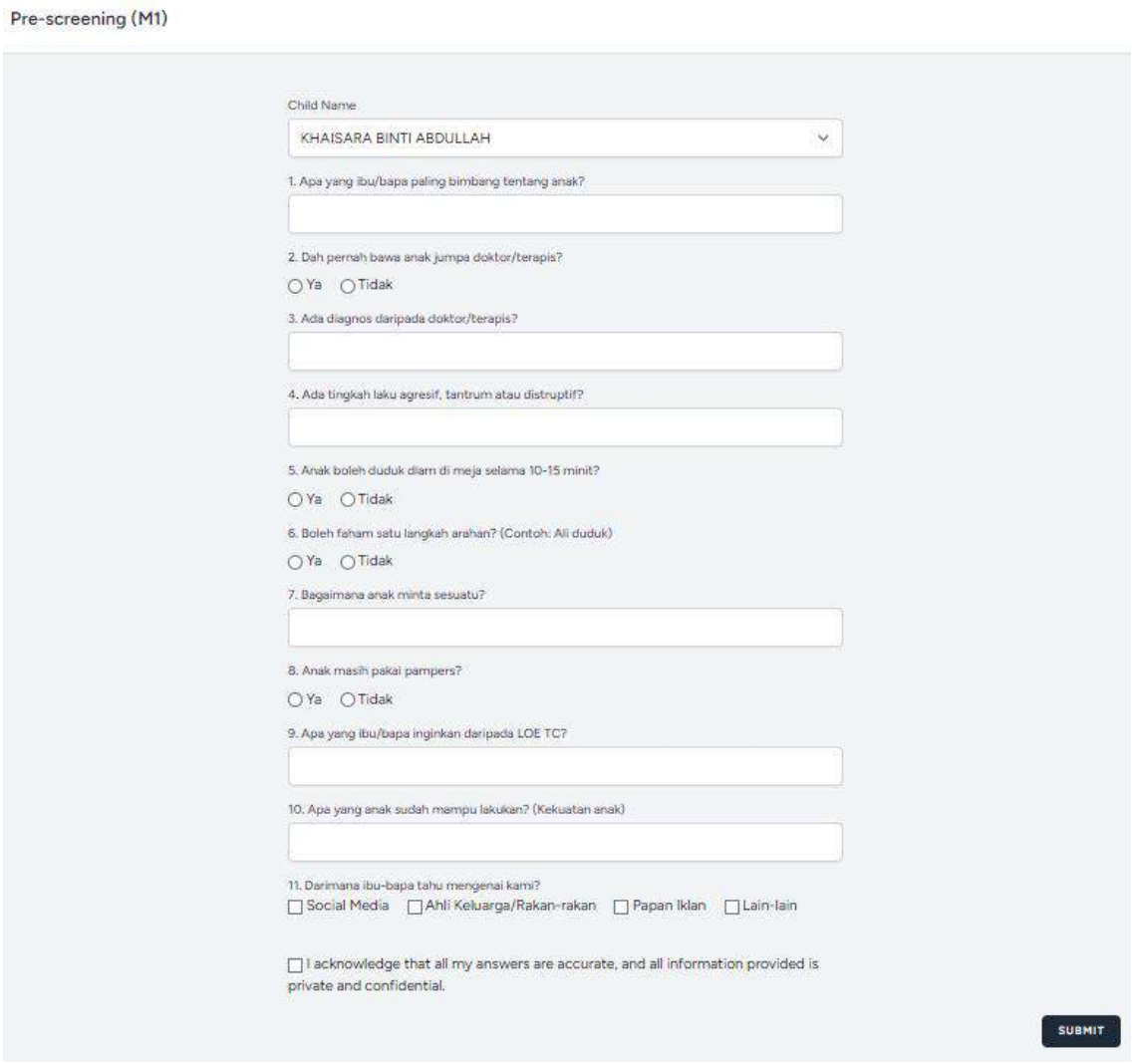


Figure 4.2.16: Manage Pre-screening (M1) interface

Parents need to answer all the questionnaire and submitting the form by clicking in the "Submit" button as shown in Figure 4.2.16.

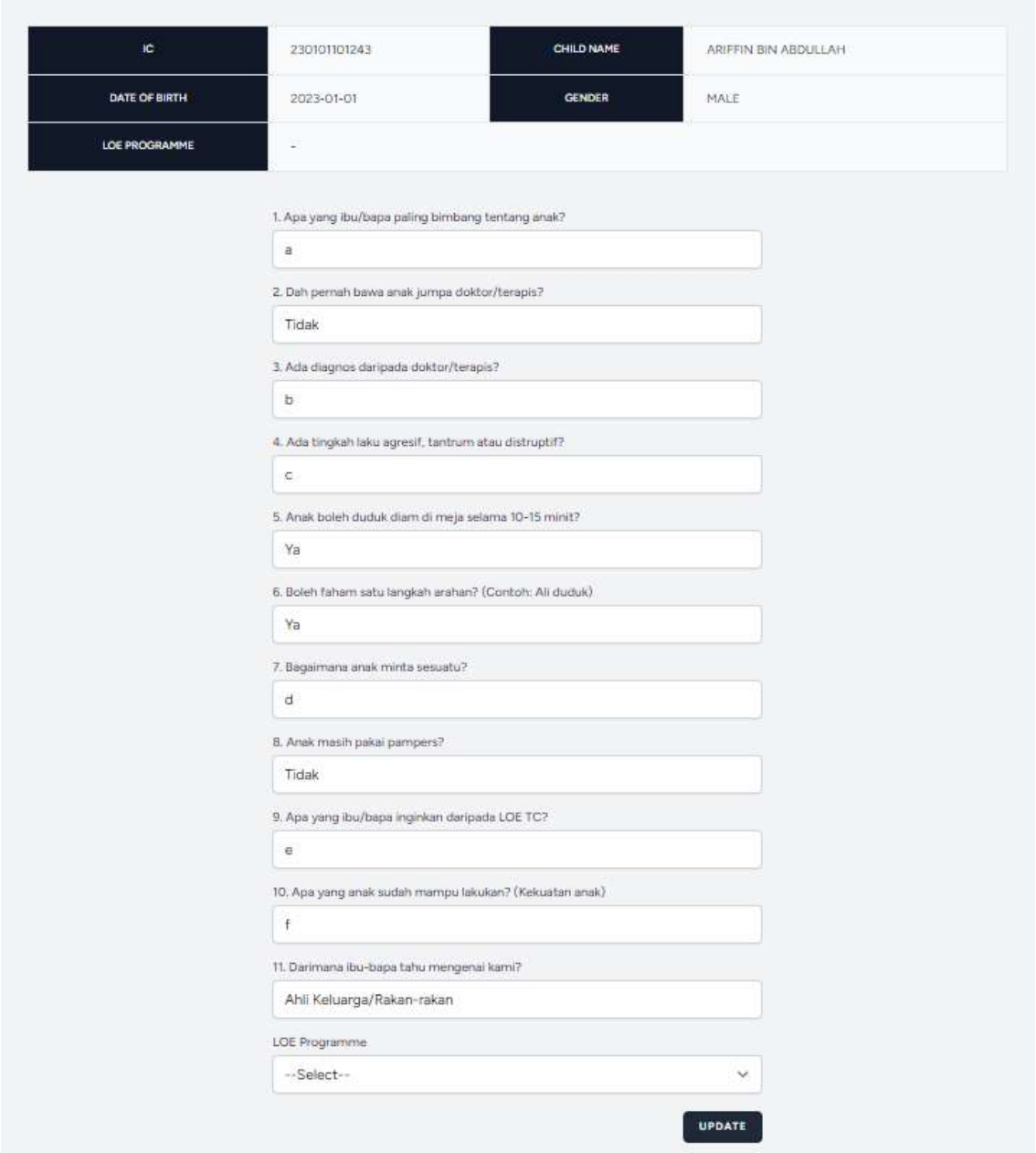


Figure 4.2.17: Update Child LOE Programme interafce (LOE staff [Manager/Principal] and OT)

LOE staff are able to see the pre-screening test of the child and update their LOE programme by selecting the dropdown menu and then click on the “Update” button as shown in Figure 4.2.17.

4.2.3.6 Manage Appointment

Figure 4.2.18: Manage Appointment interface (Admin)

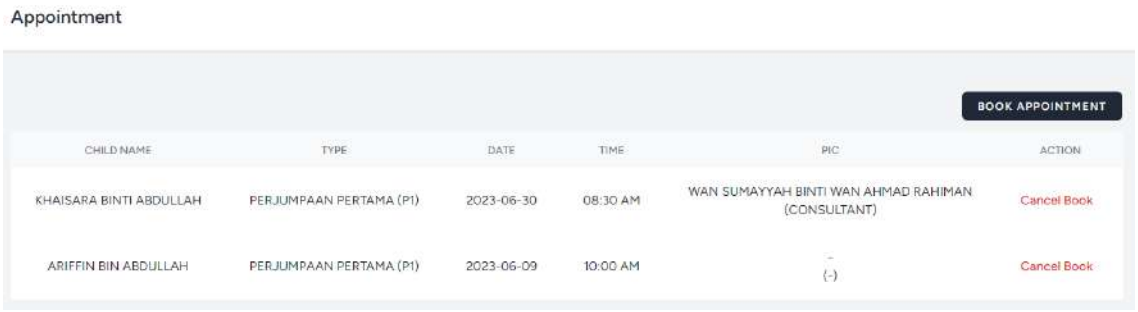


Figure 4.2.19: Manage Appointment interface (Parent)

The parent and admin are able to cancel book by clicking on the “Cancel Book” link (as shown in Figure 4.2.18 and Figure 4.2.19).

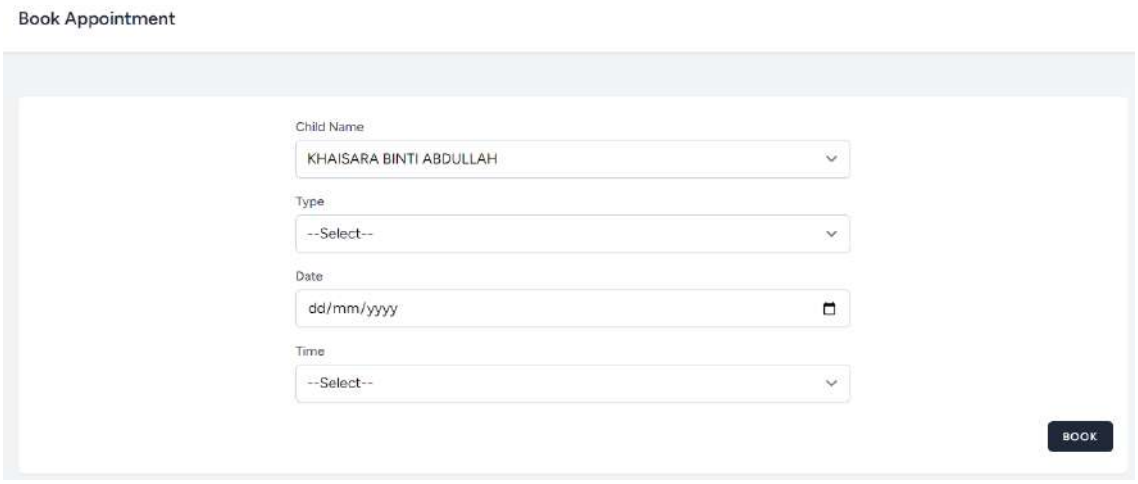


Figure 4.2.20: Book Appointment interface (Parent)

To book an appointment, the parent needs to click on the “Book Appointment” button (as shown in Figure 4.2.19) and key in all the information as shown in Figure 4.2.20. Then, click on the “Book” button to submit it.

Figure 4.2.21: Manage Appointment interface (LOE staff [Manager/Principal])

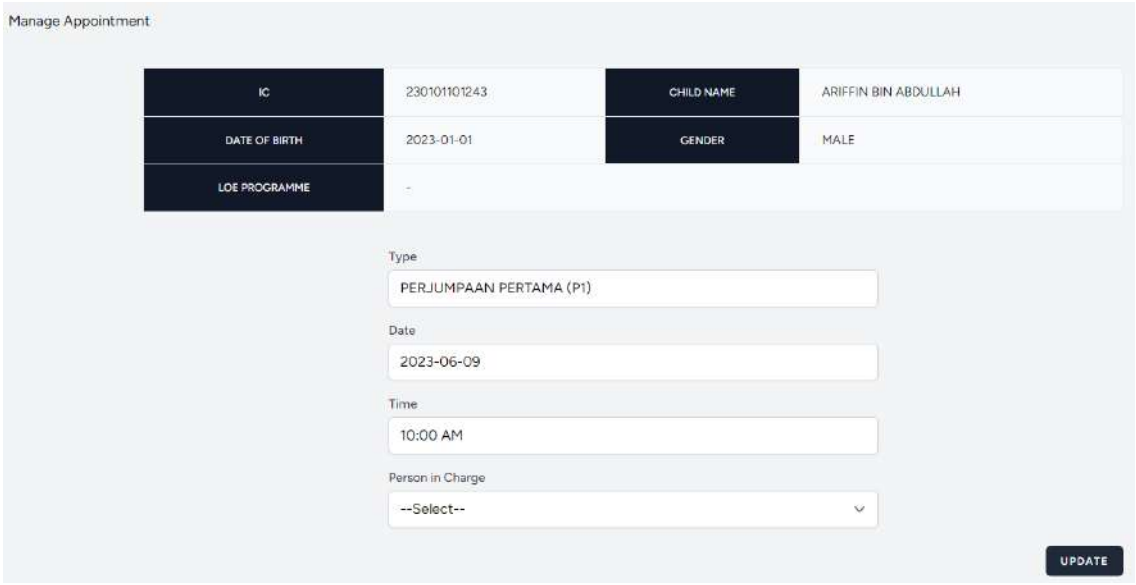


Figure 4.2.22: Update PIC interface (LOE staff [Manager/Principal])

The LOE staff is able to update the person in charge by clicking on the “Update PIC” link as shown in Figure 4.2.21. then the user will be redirected to the Update PIC page as shown in Figure 4.2.22 to update the person in charge of appointment.

4.2.3.7 Manage Parent Initial Info (M3)

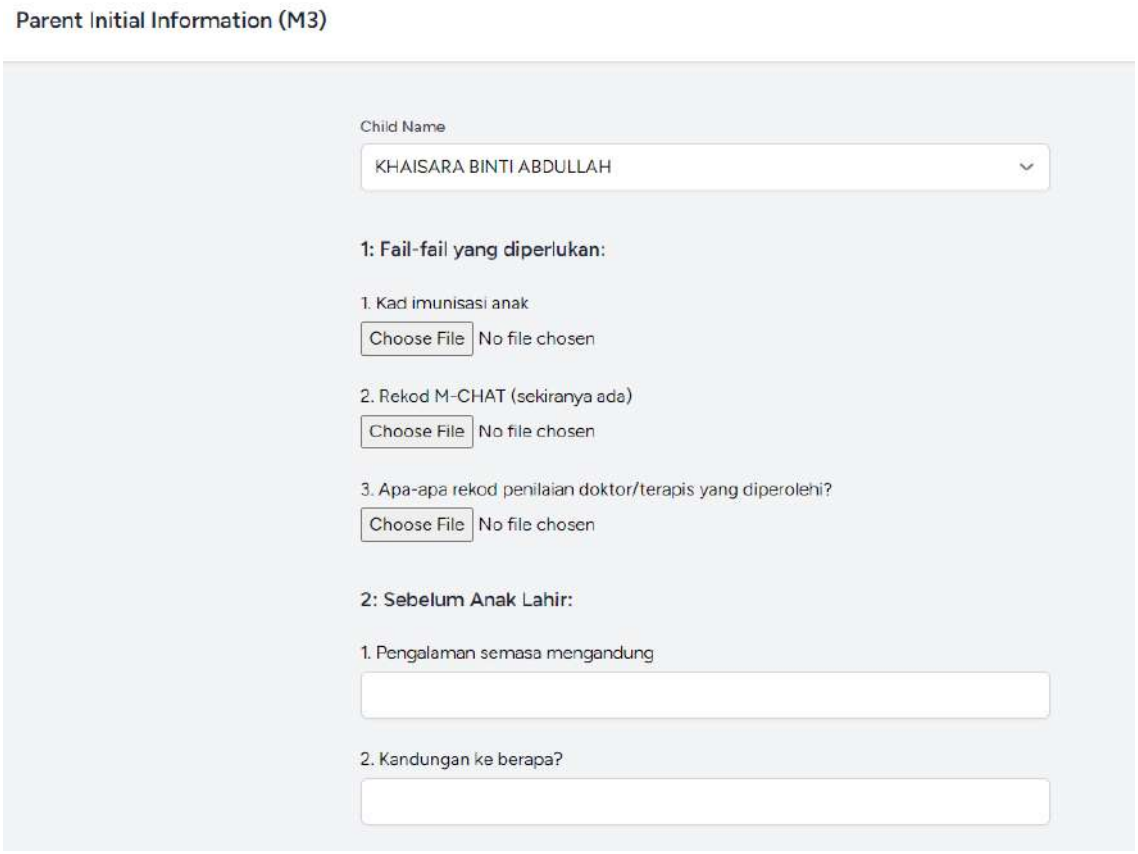


Figure 4.2.23: Manage Parent Initial Info (M3) interface

Parents need to answer all the questionnaire and submitting the form by clicking in the "Submit" button as shown in Figure 4.2.23.

4.2.3.8 Manage Payment

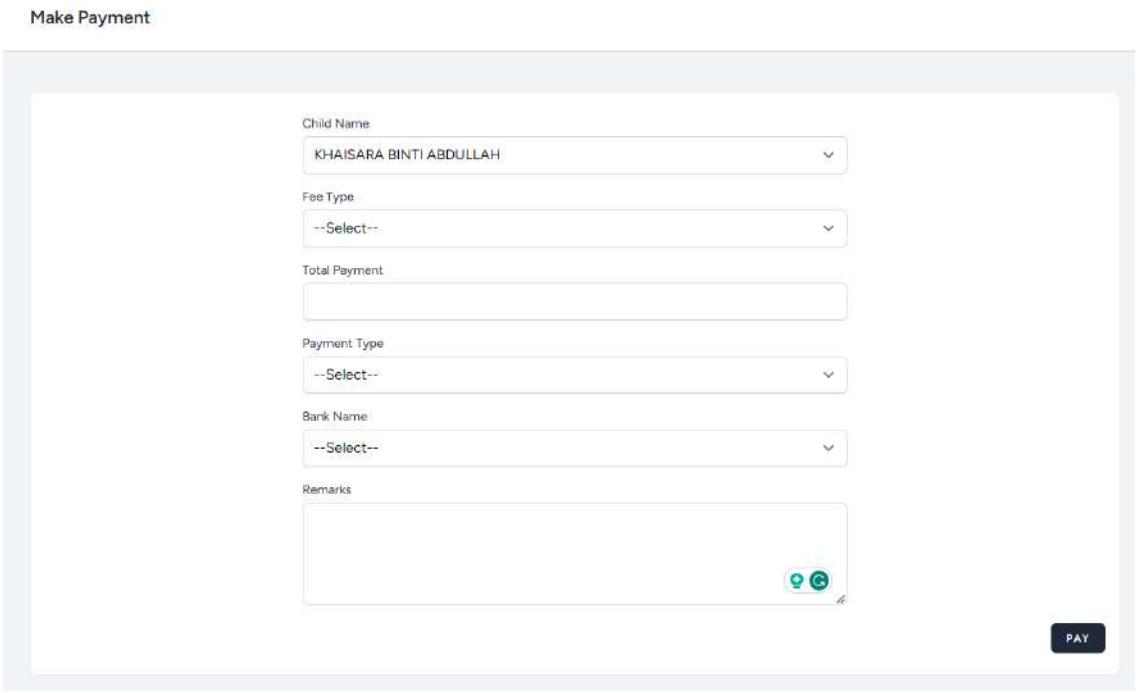


Figure 4.2.24: Manage Payment interface

Parents need to key in all the information and submitting the form by clicking in the "Pay" button as shown in Figure 4.2.24 to do payment.

4.2.3.9 Manage Individualize Education Plan (IEP)

Figure 4.2.25: Manage IEP (admin) interface

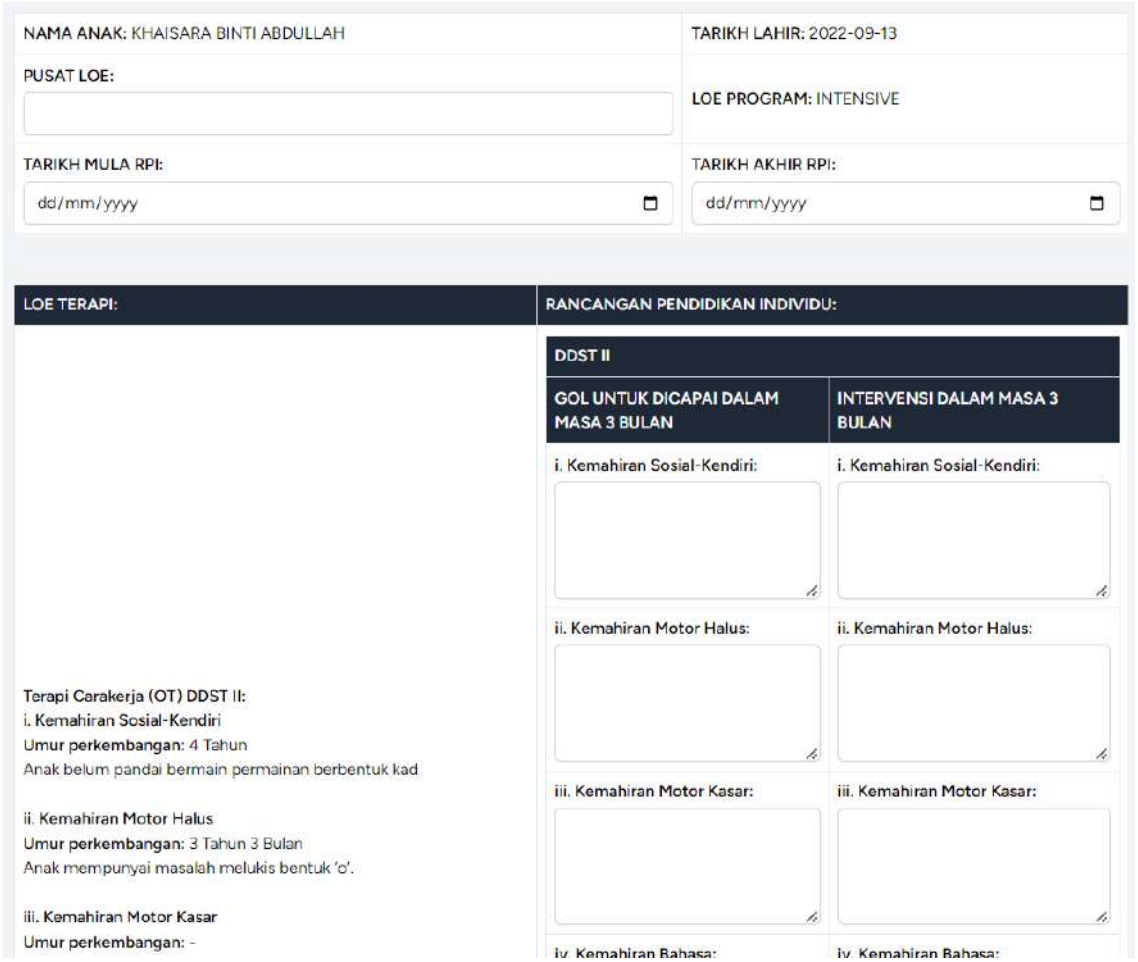


Figure 4.2.26: Add IEP interface

User need to key in all the information and submitting the form by clicking in the "Add" button as shown in Figure 4.2.26 to add new child's IEP.

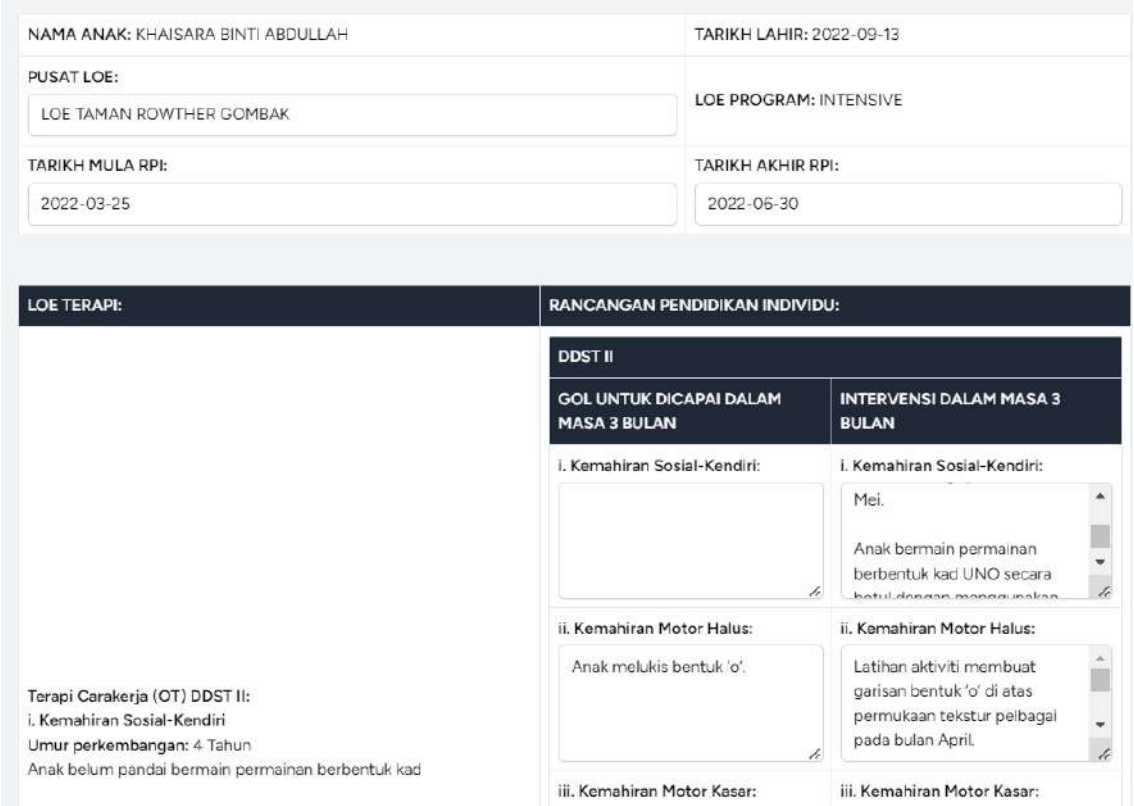


Figure 4.2.27: View IEP interface

4.2.3.10 Manage Report

Figure 4.2.28: Pre-screening Report interface

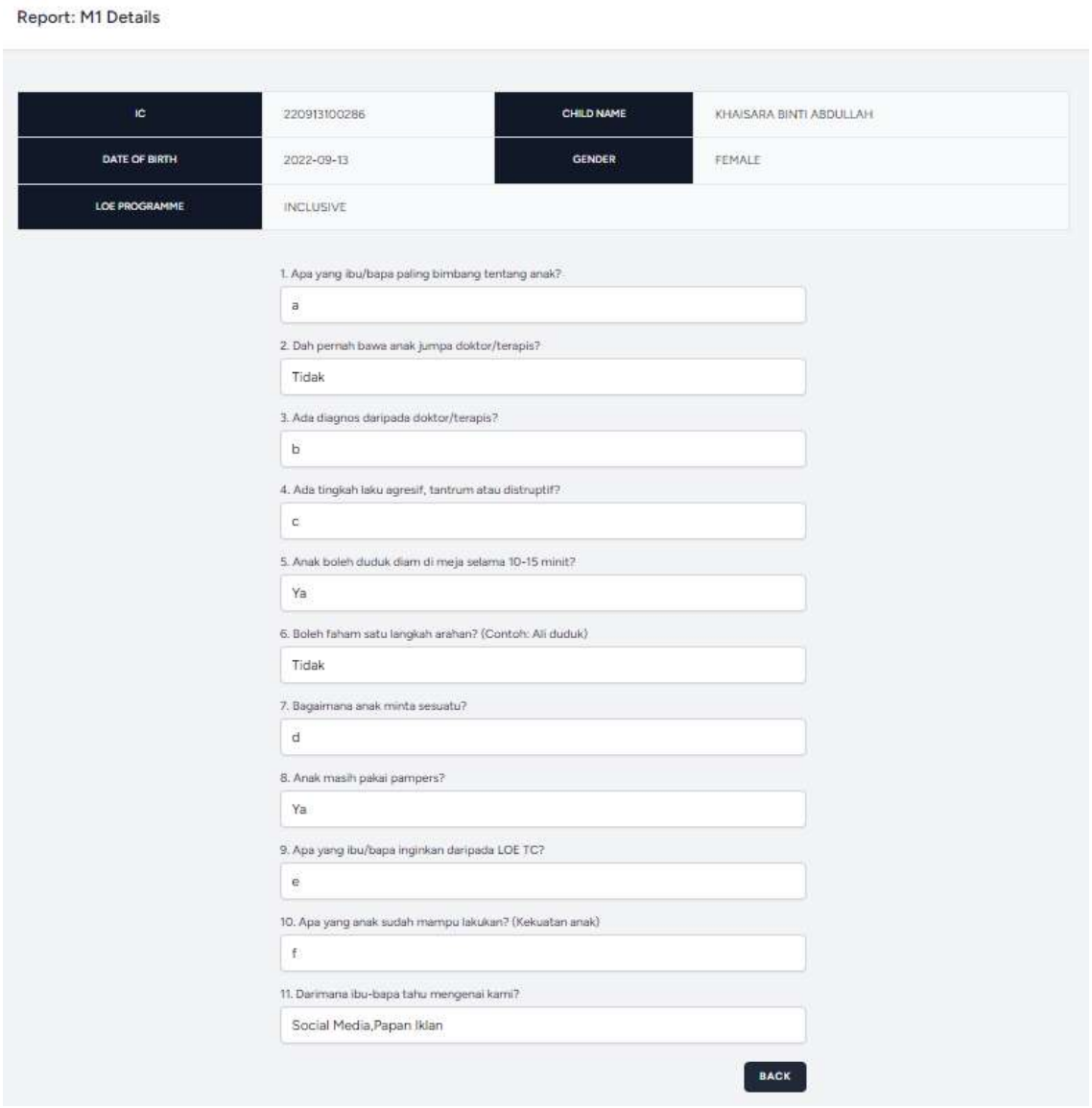


Figure 4.2.29: M1 details interface

Figure 4.2.30: Payment Report interface

Figure 4.2.31: Child and Parent Info Report interface

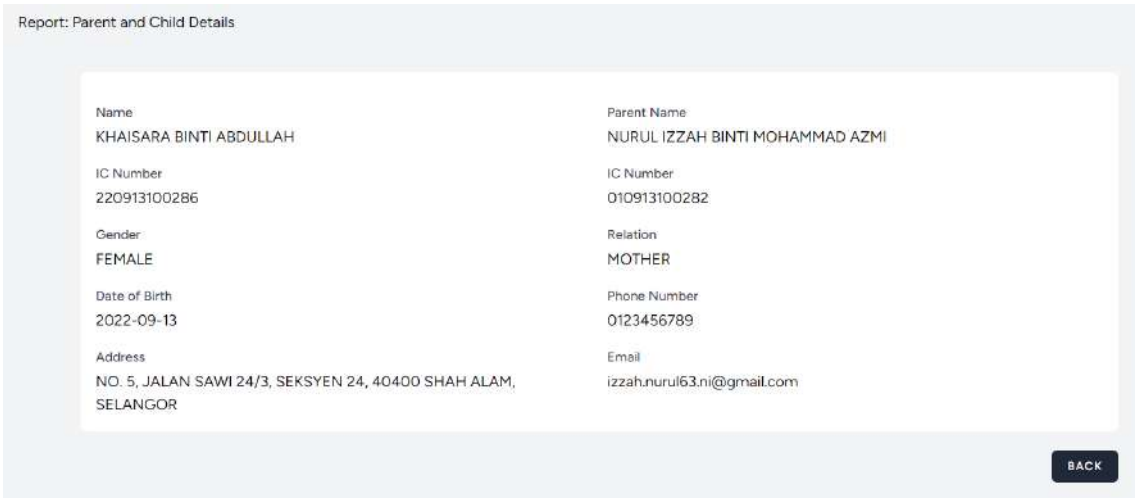
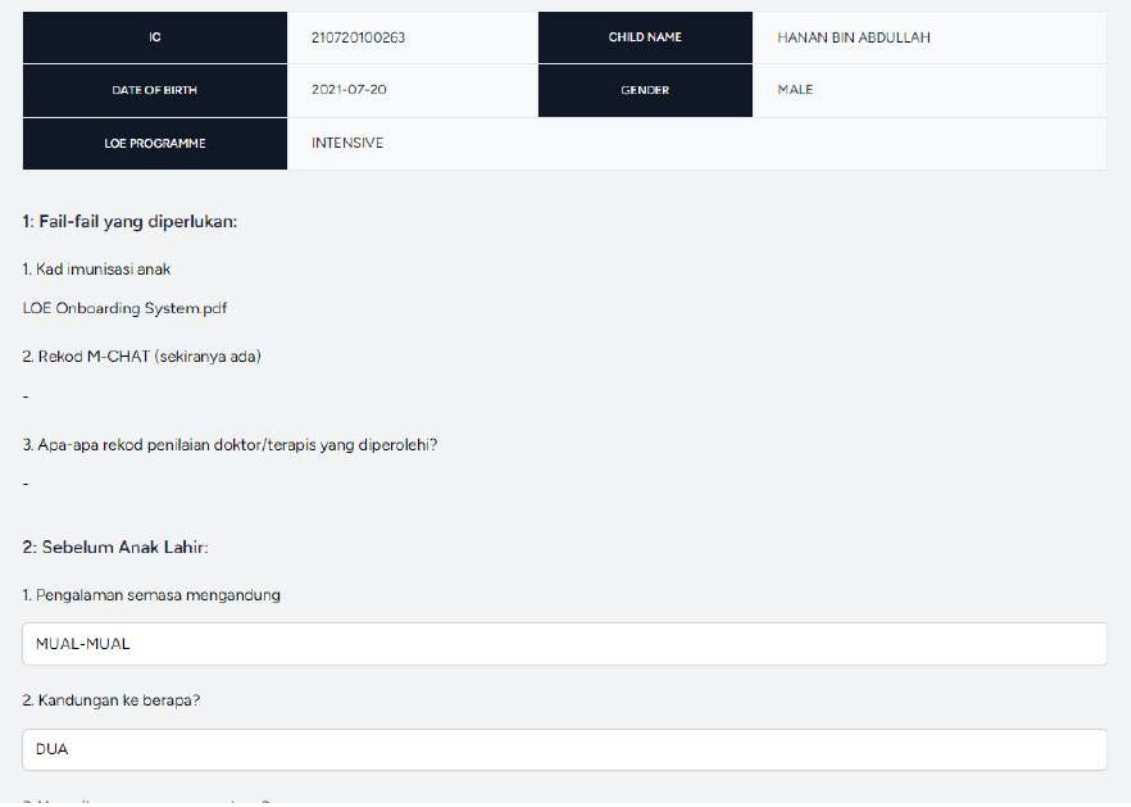


Figure 4.2.32: Child and Parent Info details interface

Figure 4.2.33: Appointment Report interface



IC	210720100263	CHILD NAME	HANAN BIN ABDULLAH
DATE OF BIRTH	2021-07-20	GENDER	MALE
LOE PROGRAMME	INTENSIVE		

1: Fail-fail yang diperlukan:

1. Kad imunisasi anak

LOE Onboarding System.pdf

2. Rekod M-CHAT (sekiranya ada)

-

3. Apa-apa rekod penilaian doktor/terapis yang diperolehi?

-

2: Sebelum Anak Lahir:

1. Pengalaman semasa mengandung

MUAL-MUAL

2. Kandungan ke berapa?

DUA

Figure 4.2.34: M3 Report interface

The users are able to view the reports only.

4.2.4 Coding Implementation

4.2.4.1 Model

```
protected $table = 'users';
protected $primaryKey = 'id';

protected $fillable = [
    'name',
    'email',
    'password',
    'role',
];
```

Figure 4.2.35: Fillable User Model

```
public function parent(){
    return $this->hasOne(ParentRecord::class, 'user_id', 'id');
}

public function staff(){
    return $this->hasOne(StaffRecord::class, 'user_id', 'id');
}
```

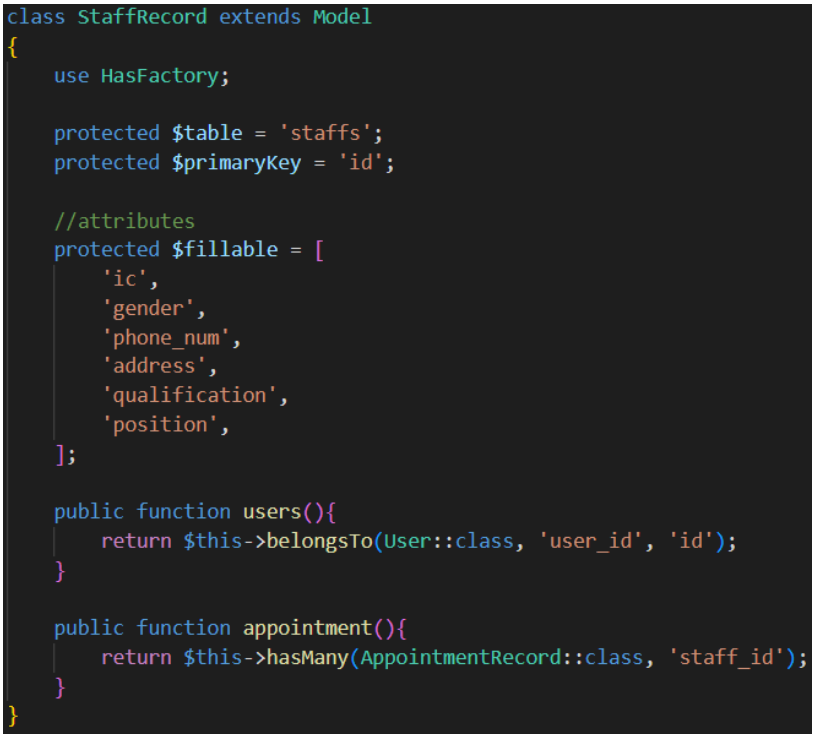
Figure 4.2.36: Relationship in User Model

This model (Figure 4.2.35 and Figure 4.2.36) is connected with the users table to store, retrieve, edit and delete users' information. This model is mostly used in the Registration, Login, Manage User System and Manage Profile modules. This model has one-to-one relation with parents table and staffs table.

```php
class ParentRecord extends Model
{
    use HasFactory;

    protected $table = 'parents';
    protected $primaryKey = 'id';

    //attributes
    protected $fillable = [
        'p_phone_num',
        'p_ic',
        'p_address',
        'p_relation',
    ];

    public function users(){
        return $this->belongsTo(User::class);
    }

    public function child(){
        return $this->hasMany(ChildRecord::class, 'parent_id', 'id');
    }
}
```

Figure 4.2.37: ParentRecord Model

This model (Figure 4.2.37) is connected with the parents table to store, retrieve, edit and delete parents' information. This model is mostly used in the Registration, Login and Manage Profile modules. This model has one-to-many relation with childs table and belongs to users table.



```php
class StaffRecord extends Model
{
    use HasFactory;

    protected $table = 'staffs';
    protected $primaryKey = 'id';

    //attributes
    protected $fillable = [
        'ic',
        'gender',
        'phone_num',
        'address',
        'qualification',
        'position',
    ];

    public function users(){
        return $this->belongsTo(User::class, 'user_id', 'id');
    }

    public function appointment(){
        return $this->hasMany(AppointmentRecord::class, 'staff_id');
    }
}
```

Figure 4.2.38: StaffRecord Model

This model (Figure 4.2.38) is connected with the staffs table to store, retrieve, edit and delete staffs' information. This model is mostly used in the Manage Profile and Manage Users modules. This model has one-to-many relation with appointments table and belongs to users table.

```php
class ChildRecord extends Model
{
    use HasFactory;
    protected $table = 'childs';
    protected $primaryKey = 'id';

    //attributes
    protected $fillable = [
        'ch_name',
        'ch_gender',
        'ch_dob',
        'ch_ic',
        'loe_programme',
        'parent_id',
    ];

    public function parent(){
        return $this->belongsTo(ParentRecord::class, 'parent_id', 'id');
    }

    public function m1(){
        return $this->hasMany(M1Record::class, 'child_id');
    }

    public function appointment(){
        return $this->hasMany(AppointmentRecord::class, 'child_id');
    }

    public function m3(){
        return $this->hasMany(M3Record::class, 'child_id');
    }

    public function payment(){
        return $this->hasMany(PaymentRecord::class, 'child_id');
    }
}
```

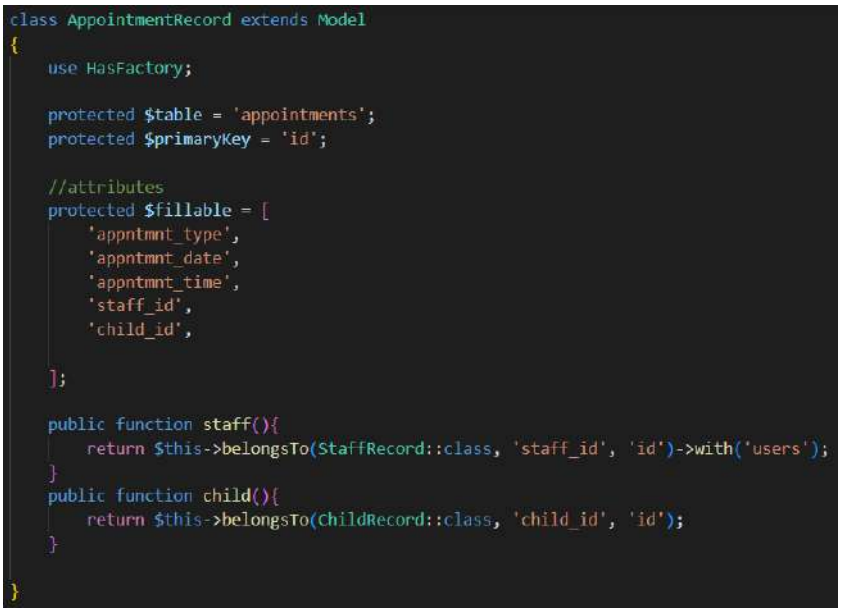
Figure 4.2.39: ChildRecord Model

This model (Figure 4.2.39) is connected with the childs table to store, retrieve, edit and delete childs' information. This model is mostly used in the Manage Child Profile and Manage M1, Manage M3, Manage Payment and most modules in the system. This model has one-to-many relation with appointments table, m3s table, payments table and m1s table. This model also belongs to parents table.

```php
class M1Record extends Model
{
    use HasFactory;
    protected $table = 'm1s';
    protected $primaryKey = 'id';

    //attributes
    protected $fillable = [
        'psQID',
        'answer',
        'child_id',
    ];

    public function child(){
        return $this->belongsTo(ChildRecord::class, 'child_id', 'id');
    }

}
```

Figure 4.2.40: M1Record Model

This model (Figure 4.2.40) is connected with the m1s table to store and retrieve m1s information. This model is mostly used in the Manage M1 and Manage Report modules. Also, this model has relation belongs to childs table.



```php
class AppointmentRecord extends Model
{
    use HasFactory;

    protected $table = 'appointments';
    protected $primaryKey = 'id';

    //attributes
    protected $fillable = [
        'appntmnt_type',
        'appntmnt_date',
        'appntmnt_time',
        'staff_id',
        'child_id',

    ];

    public function staff(){
        return $this->belongsTo(StaffRecord::class, 'staff_id', 'id')->with('users');
    }
    public function child(){
        return $this->belongsTo(ChildRecord::class, 'child_id', 'id');
    }

}
```

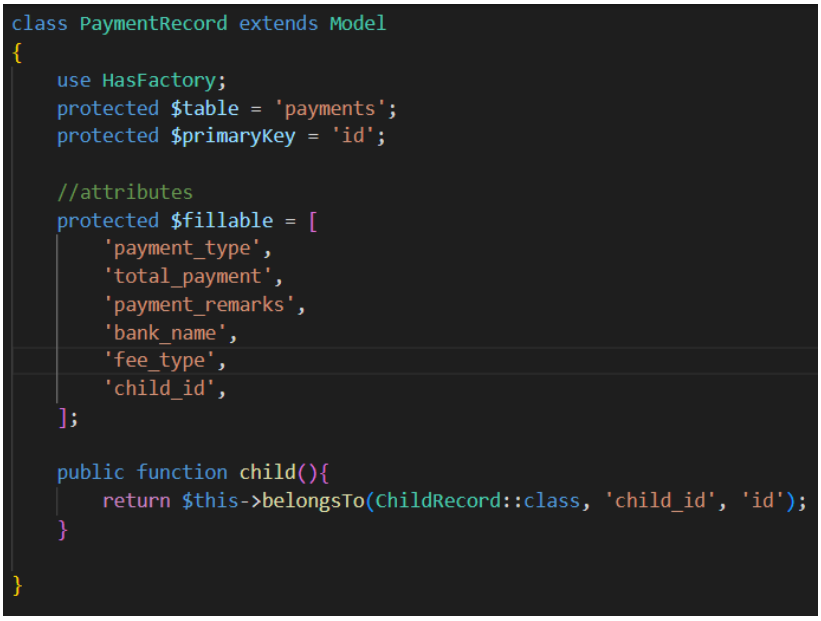
Figure 4.2.41: AppointmentRecord Model

This model (Figure 4.2.41) is connected with the appointments table to store, retrieve, edit and delete appointments' information. This model is mostly used in the Manage Appointment and Manage Report modules. This model has relation belongs to staffs and childs table.

```php
class M3Record extends Model
{
    use HasFactory;
    protected $table = 'm3s';
    protected $primaryKey = 'id';

    //attributes
    protected $fillable = [
        'm3QID',
        'answer',
        'child_id',
    ];

    public function child(){
        return $this->belongsTo(ChildRecord::class, 'child_id', 'id');
    }

}
```

Figure 4.2.42: M3Record Model

This model (Figure 4.2.42) is connected with the m3s table to store and retrieve m3s information. This model is mostly used in the Manage M3 and Manage Report modules. This model has relation belongs to childs table.



```php
class PaymentRecord extends Model
{
    use HasFactory;
    protected $table = 'payments';
    protected $primaryKey = 'id';

    //attributes
    protected $fillable = [
        'payment_type',
        'total_payment',
        'payment_remarks',
        'bank_name',
        'fee_type',
        'child_id',
    ];

    public function child(){
        return $this->belongsTo(ChildRecord::class, 'child_id', 'id');
    }

}
```

Figure 4.2.43: PaymentRecord Model

This model (Figure 4.2.43) is connected with the payments table to store and retrieve payment' information. This model is mostly used in the Manage Payment and Manage Report modules. This model has relation belongs to childs table.

4.2.4.2 Controller

```php
class HomeController extends Controller
{
    //Check User Role
    public function index()
    {

        if (!Auth::user()) {
            return redirect()->route('login');
        }
        if (Auth::user()->role === 'ADMIN') {
            return redirect()->route('admin.dashboard');
        }
        if (Auth::user()->role === 'STAFF') {
            $staffRecord = StaffRecord::where('user_id', Auth::user()->id)->first();
            if ($staffRecord && $staffRecord->position === 'OCCUPATIONAL THERAPY') {
                return redirect()->route('child.list');
            } elseif ($staffRecord && $staffRecord->position === 'CONSULTANT') {
                return redirect()->route('consultant.dashboard');
            } elseif ($staffRecord && in_array($staffRecord->position, ['MANAGER', 'PRINCIPAL'])) {
                return redirect()->route('staff.dashboard');
            }
        } else
            return redirect()->route('dashboard');
    }


}
```

Figure 4.2.44: Home Controller

This controller (Figure 4.2.44) is used to redirect the users from login page to the respective user dashboard pages.

```php
class UserController extends Controller
{
    public function index()
    {
        $users = User::with(['staff', 'parent'])->where('role', '!=', 'ADMIN')->get();
        // Pass the users data to a view
        return view('admin-dashboard', compact('users'));
    }


    /**
     * Display the specified resource.
     *
     * @param  int  $id
     * @return \Illuminate\Http\Response
     */

    public function show($id)
    {
        // page view/manage-user-system.view-user
        $users = User::where('id', $id)->get();
        return view('manage-user-system.view-user', ['users' => $users]);
    }

    public function create(Request $request)
    {
        // page view/manage-user-system.add-user
        //proses submit dekat form bila tkan bttn
        // ikut migration
        $inputUser = new User();
        $inputUser->name = $request->name;
        $inputUser->email = $request->email;
        $inputUser->password = Hash::make($request->password);
        $inputUser->role = 'STAFF';
        $inputUser->save();

        $inputStaff = new StaffRecord();
        $inputStaff->ic = $request->ic;
        $inputStaff->gender = $request->gender;
        $inputStaff->phone_num = $request->phone_num;
        $inputStaff->address = $request->address;
        $inputStaff->qualification = $request->qualification;
        $inputStaff->position = $request->position;
        $inputStaff->user_id = $inputUser->id;
        $inputStaff->save();

        return redirect()->route('admin.dashboard')->with('success', 'Successfully Add');
    }
```

Figure 4.2.45: User Controller (index, show, create function)

```php
public function store(array $input): User
{
    //untuk masukkan data
    //add values to multiple tables
    DB::transaction(function () use ($input, &$user) {
        Validator::make($input, [
            'name' => ['required', 'string', 'max:255'],
            'email' => ['required', 'string', 'email', 'max:255', 'unique:users'],
            'password' => $this->passwordRules(),
            'terms' => Jetstream::hasTermsAndPrivacyPolicyFeature() ? ['accepted', 'required'] : '',
            //Staff
            'ic' => ['required', 'regex:/\d{12}/'],
            'gender' => ['required', 'in:MALE,FEMALE'],
            'phone_num' => ['required', 'string', 'regex:/\d{10,11}/'],
            'address' => ['required', 'string', 'max:255'],
            'qualification' => ['required', 'string', 'max:255'],
            'position' => ['required', 'string', 'max:255'],
        ])->validate();

        $user = User::create([
            'name' => $input['name'],
            'email' => $input['email'],
            'password' => Hash::make($input['password']),
        ]);

        //Create staff (Child table of User table)
        $staff = new StaffRecord([
            'ic' => $input['ic'],
            'gender' => $input['gender'],
            'phone_num' => $input['phone_num'],
            'address' => $input['address'],
            'qualification' => $input['qualification'],
            'position' => $input['position'],
        ]);

        $user->parent()->save($staff);
        event(new Registered($user));
    });
    return $user;

    //kembali ke interface asal

    return redirect('admin.dashboard')
        ->with('success', 'New User Successfully Added');
}
```

Figure 4.2.46: User Controller (store function)

```php
public function edit($id)
{
    $users = User::where('id', $id)->get();
    return view('manage-user-system.edit-user', ['users' => $users]);
}

public function editProfile()
{
    // $users = User::where('id', $id)->get();
    return view('profile.update-user-details-form');
}

/**
 * Update the specified resource in storage.
 *
 * @param  \Illuminate\Http\Request  $request
 * @param  int  $id
 * @return \Illuminate\Http\Response
 */

public function update(Request $request, $id)
{
    $users = User::find($id);

    $users->name = $request->input('name');
    $users->email = $request->input('email');

    if ($users->role === "STAFF") {
        $users->staff->address = $request->input('address');
        $users->staff->phone_num = $request->input('phone_num');
        $users->staff->ic = $request->input('ic');
        $users->staff->qualification = $request->input('qualification');
        $users->staff->position = $request->input('position');
        $users->staff->gender = $request->input('gender');
        $users->staff->update();
    }
    if ($users->role === "PARENT") {
        $users->parent->p_ic = $request->input('ic');
        $users->parent->p_phone_num = $request->input('phone_num');
        $users->parent->p_address = $request->input('address');
        $users->parent->p_relation = $request->input('p_relation');
        $users->parent->update();
    }

    // Check if the password field is not empty
    if ($request->filled('password')) {
        $users['password'] = Hash::make($request->input('password'));
    }

    $users->update();


    return redirect()->route('admin.dashboard')->with('success', 'Successfully Update');
}
```

Figure 4.2.47: User Controller (edit, editProfile, update function)

```php
public function updateProfile(Request $request, $id)
{
    $users = User::find($id);

    if ($users->role === "STAFF") {
        $users->staff->address = $request->input('address');
        $users->staff->phone_num = $request->input('phone_num');
        $users->staff->ic = $request->input('ic');
        $users->staff->qualification = $request->input('qualification');
        $users->staff->position = $request->input('position');
        $users->staff->gender = $request->input('gender');
        $users->staff->save();
    }
    if ($users->role === "PARENT") {
        $users->parent->p_ic = $request->input('p_ic');
        $users->parent->p_phone_num = $request->input('p_phone_num');
        $users->parent->p_address = $request->input('p_address');
        $users->parent->p_relation = $request->input('p_relation');
        $users->parent->save();
    }

    // dd($request->all());

    // Refresh the page
    return redirect()->route('profile.edit')->with('success', 'Successfully Update');
}

/**
 * Remove the specified resource from storage.
 *
 * @param  int  $id
 * @return \Illuminate\Http\Response
 */

public function destroy($user_id)
{
    $user = User::findOrFail($user_id);
    // delete related parent data first
    // Delete related staffs records
    $user->staff()->delete();
    $user->parent()->delete();

    // then delete the user data
    $user->delete();

    return redirect()->route('admin.dashboard')
        ->with('success', 'User Successfully Deleted');
}
```

Figure 4.2.48: User Controller (updateProfile, destroy function)

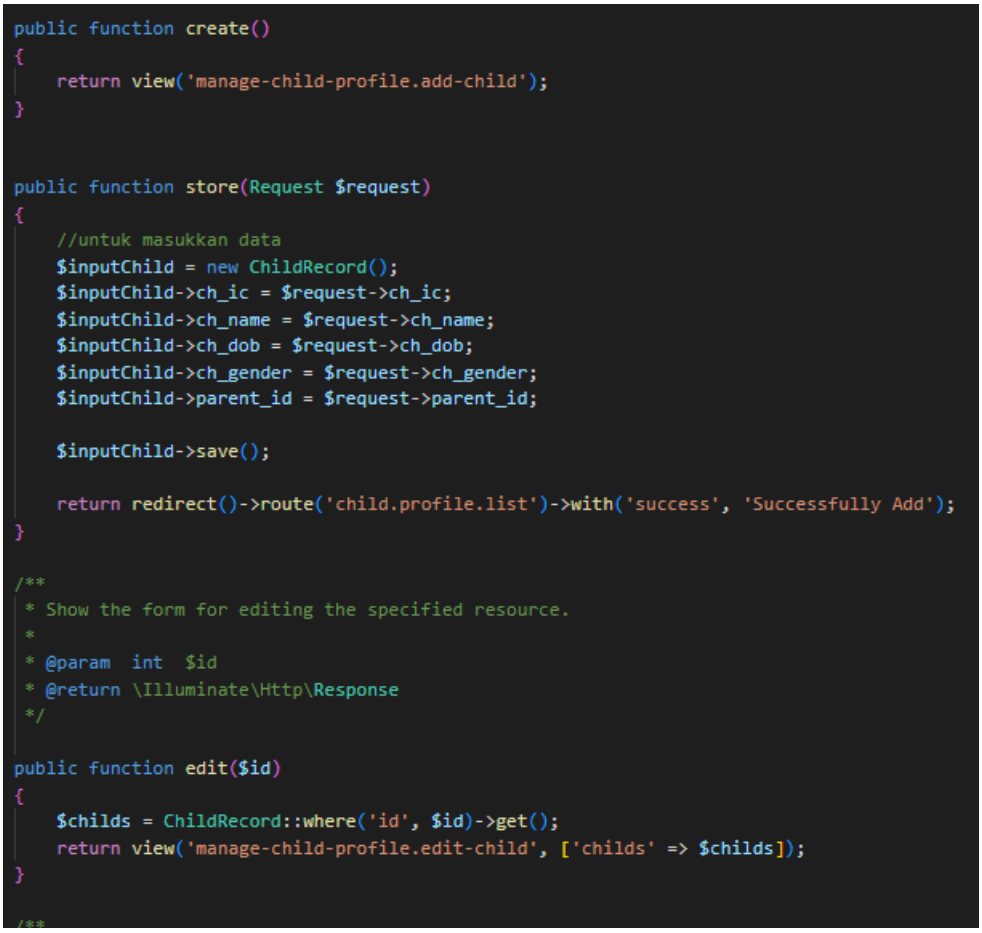
This controller (Figure 4.2.45, Figure 4.2.46, Figure 4.2.47, Figure 4.2.48) is used to view, store, edit, update, and delete the users' information.

```php
class ChildController extends Controller
{
    //
    public function index()
    {
        if (Auth::user()->role === 'PARENT') {
            $users = auth()->user();
            $childs = $users->parent->child;

            // Pass the users data to a view
            return view('manage-child-profile.list-child', compact('childs'));
        }
        if (Auth::user()->role === 'ADMIN') {
            $childs = ChildRecord::all();
            // Pass the users data to a view
            return view('manage-report.admin.info', compact('childs'));
        }
        if (Auth::user()->role === 'STAFF') {
            $staffRecord = StaffRecord::where('user_id', Auth::user()->id)->first();
            if ($staffRecord && in_array($staffRecord->position, ['MANAGER', 'PRINCIPAL'])) {
                // Get the childs for the parent's children
                $childs = ChildRecord::all();

                // Pass the users data to a view
                return view('manage-report.admin.info', compact('childs'));
            }
            if ($staffRecord && in_array($staffRecord->position, ['OCCUPATIONAL THERAPY'])) {
                // Get the childs for the parent's children
                $childs = ChildRecord::all();

                // Pass the users data to a view
                return view('manage-report.ot.info', compact('childs'));
            }
            if ($staffRecord && in_array($staffRecord->position, ['CONSULTANT'])) {
                // Get the childs info for the parent's children
                $childs = ChildRecord::all();

                // Pass the users data to a view
                return view('manage-report.csltnt.info', compact('childs'));
            }
        }
    }
```

Figure 4.2.49: Child Controller (index function)

```
public function profile($id)
{
    if (Auth::user()->role === 'ADMIN') {
        $childs = ChildRecord::with('parent.users')
            ->join('parents', 'parents.id', '=', 'childs.parent_id')
            ->join('users', 'users.id', '=', 'parents.user_id')
            ->where('childs.id', $id)->get();
        // Pass the users data to a view
        return view('manage-report.admin.info-details', compact('childs'));
    }
    if (Auth::user()->role === 'STAFF') {
        $staffRecord = StaffRecord::where('user_id', Auth::user()->id)->first();
        if ($staffRecord && in_array($staffRecord->position, ['MANAGER', 'PRINCIPAL'])) {
            // Get the childs for the parent's children
            $childs = ChildRecord::with('parent.users')
                ->join('parents', 'parents.id', '=', 'childs.parent_id')
                ->join('users', 'users.id', '=', 'parents.user_id')
                ->where('childs.id', $id)->get();

            // Pass the users data to a view
            return view('manage-report.admin.info-details', compact('childs'));
        }
        if ($staffRecord && in_array($staffRecord->position, ['OCCUPATIONAL THERAPY'])) {
            // Get the childs for the parent's children
            $childs = ChildRecord::with('parent.users')
                ->join('parents', 'parents.id', '=', 'childs.parent_id')
                ->join('users', 'users.id', '=', 'parents.user_id')
                ->where('childs.id', $id)->get();

            // Pass the users data to a view
            return view('manage-report.ot.info-details', compact('childs'));
        }
        if ($staffRecord && in_array($staffRecord->position, ['CONSULTANT'])) {
            // Get the childs info for the parent's children
            $childs = ChildRecord::with('parent.users')
                ->join('parents', 'parents.id', '=', 'childs.parent_id')
                ->join('users', 'users.id', '=', 'parents.user_id')
                ->where('childs.id', $id)->get();

            // Pass the users data to a view
            return view('manage-report.csltnt.info-details', compact('childs'));
        }
    }
}
```

Figure 4.2.50: Child Controller (profile function)



```php
public function create()
{
    return view('manage-child-profile.add-child');
}


public function store(Request $request)
{
    //untuk masukkan data
    $inputChild = new ChildRecord();
    $inputChild->ch_ic = $request->ch_ic;
    $inputChild->ch_name = $request->ch_name;
    $inputChild->ch_dob = $request->ch_dob;
    $inputChild->ch_gender = $request->ch_gender;
    $inputChild->parent_id = $request->parent_id;

    $inputChild->save();

    return redirect()->route('child.profile.list')->with('success', 'Successfully Add');
}

/**
 * Show the form for editing the specified resource.
 *
 * @param  int  $id
 * @return \Illuminate\Http\Response
 */

public function edit($id)
{
    $childs = ChildRecord::where('id', $id)->get();
    return view('manage-child-profile.edit-child', ['childs' => $childs]);
}

/**
```

Figure 4.2.51: Child Controller (create, edit, store function)

```php
public function update(Request $request, $id)
{
    $childs = ChildRecord::find($id);

    $childs->ch_ic = $request->input('ch_ic');
    $childs->ch_name = $request->input('ch_name');
    $childs->ch_dob = $request->input('ch_dob');
    $childs->ch_gender = $request->input('ch_gender');
    $childs->parent_id = $request->input('parent_id');
    $childs->update();

    return redirect()->route('child.profile.list')->with('success', 'Successfully Edit');
}

/**
 * Remove the specified resource from storage.
 *
 * @param  int  $id
 * @return \Illuminate\Http\Response
 */

public function destroy($id)
{
    $childs = ChildRecord::findOrFail($id);

    // then delete the user data
    $childs->delete();

    return redirect()->route('child.profile.list')
        ->with('success', 'Child Successfully Deleted')->with('success', 'Successfully Delete');
}
```

Figure 4.2.52: Child Controller (update, destroy function)

This controller (Figure 4.2.49, Figure 4.2.50, Figure 4.2.51, Figure 4.2.52) is used to view, store, edit, update, and delete the child' information.

```php
class M1Controller extends Controller
{
    //
    public function index()
    {
        $user = auth()->user();
        $parent = $user->parent;
        $childs = $parent->child;

        // Pass the users data to a view
        return view('manage-m1.parent.m1-question', compact('childs'));
    }

    //staff
    public function childList()
    {
        if (Auth::user()->role === 'STAFF') {
            $childs = ChildRecord::whereHas('m1')
                ->get();

            $staffRecord = StaffRecord::where('user_id', Auth::user()->id)->first();
            if ($staffRecord && in_array($staffRecord->position, ['MANAGER', 'PRINCIPAL'])) {
                // Pass the users data to a view
                return view('manage-m1.staff.m1-list', compact('childs'));
            } else {
                // Pass the users data to a view
                return view('ot-dashboard', compact('childs'));
            }
        }
        if (Auth::user()->role === 'ADMIN') {
            $childs = ChildRecord::whereHas('m1')
                ->get();

            return view('manage-report.admin.m1', compact('childs'));
        }
        if (Auth::user()->role === 'PARENT') {
            // Get the authenticated user (parent)
            $parent = Auth::user()->parent;

            // Get the appointments for the parent's children
            $childs = ChildRecord::whereHas('m1')->where('parent_id', $parent->id)->get();

            return view('manage-report.parent.m1', compact('childs'));
        }
    }
```

Figure 4.2.53: M1 Controller (index, childList function)

```php
//staff
public function m1Details($id)
{
    $m1s = M1Record::with(['child'])
        ->where('child_id', $id)->get();

    $childs = ChildRecord::find($id);

    $questions = [
        1 => "1. Apa yang ibu/bapa paling bimbang tentang anak?",
        2 => "2. Dah pernah bawa anak jumpa doktor/terapis?",
        3 => "3. Ada diagnos daripada doktor/terapis?",
        4 => "4. Ada tingkah laku agresif, tantrum atau distruptif?",
        5 => "5. Anak boleh duduk diam di meja selama 10-15 minit?",
        6 => "6. Boleh faham satu langkah arahan? (Contoh: Ali duduk)",
        7 => "7. Bagaimana anak minta sesuatu?",
        8 => "8. Anak masih pakai pampers?",
        9 => "9. Apa yang ibu/bapa inginkan daripada LOE TC?",
        10 => "10. Apa yang anak sudah mampu lakukan? (Kekuatan anak)",
        11 => "11. Darimana ibu-bapa tahu mengenai kami?",

    ];

    if (Auth::user()->role === 'STAFF') {
        $staffRecord = StaffRecord::where('user_id', Auth::user()->id)->first();
        if ($staffRecord && in_array($staffRecord->position, ['MANAGER', 'PRINCIPAL'])) {
            // Pass the users data to a view
            return view('manage-m1.staff.update-program', compact('m1s', 'childs', 'questions'));
        } else {
            // Pass the users data to a view
            return view('manage-m1.ot.update-program', compact('m1s', 'childs', 'questions'));
        }
    }
    if (Auth::user()->role === 'ADMIN') {
        // Pass the users data to a view
        return view('manage-report.admin.m1-details', compact('m1s', 'childs', 'questions'));
    }
    if (Auth::user()->role === 'PARENT') {

        return view('manage-report.parent.m1-details', compact('m1s', 'childs', 'questions'));
    }
}
```

Figure 4.2.54: M1 Controller (m1Details function)

```php
public function store(Request $request)
{
    // Loop through the questions and store the answers
    for ($i = 1; $i <= 11; $i++) {
        $answer = $request->input('q' . $i);

        // Checkbox input
        if ($i == 11) {
            $q11 = $request->input('q11');
            if ($q11 && is_array($q11)) {
                $answer = implode(',', $q11);
            }
        }

        // Create a new instance of the M1s model
        $m1s = new M1Record();

        // Assign values from the form inputs to the model attributes
        $m1s->psQID = $i;
        $m1s->answer = $answer;
        $m1s->child_id = $request->input('child_id');

        // Save the model to the database
        $m1s->save();
    }

    // Perform any additional actions or return a response as needed
    return redirect()->route('dashboard')->with('success', 'Successfully Submit');
}
```

Figure 4.2.55: M1 Controller (store function)

```php
public function update(Request $request, $id)
{
    $childs = ChildRecord::find($id);

    $childs->loe_programme = $request->input('loe');
    $childs->save();

    if (Auth::user()->role === 'STAFF') {
        $staffRecord = StaffRecord::where('user_id', Auth::user()->id)->first();
        if ($staffRecord && in_array($staffRecord->position, ['MANAGER', 'PRINCIPAL'])) {
            // Pass the users data to a view
            return redirect()->route('child.list')->with('success', 'Successfully Update');
        } else {
            // Pass the users data to a view
            return redirect()->route('child.list')->with('success', 'Successfully Update');
        }
    }
}
```

Figure 4.2.56: M1 Controller (update function)

This controller (Figure 4.2.53, Figure 4.2.54, Figure 4.2.55, Figure 4.2.56) is used to view and store M1 information. Also, to edit and update child's LOE programme.

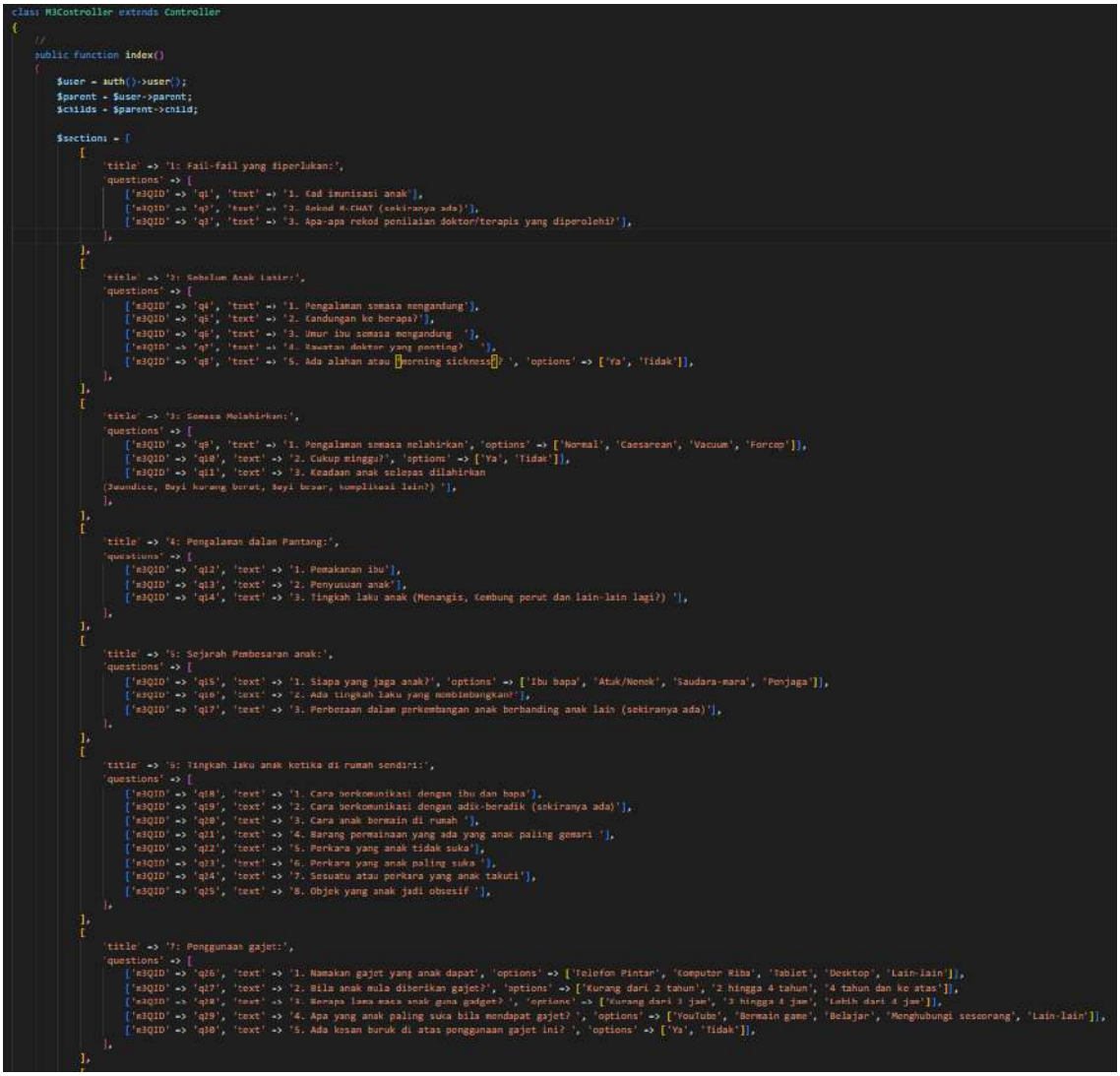
```php
class AppointmentController extends Controller
{
    //
    public function index()
    {
        if (Auth::user()->role === 'ADMIN') {

            // Get the appointments for the parent's children
            $appointments = AppointmentRecord::with('staff.users', 'child')->get();

            // Pass the users data to a view
            return view('manage-appointment.admin.cancel-book', compact('appointments'));
        }
        if (Auth::user()->role === 'STAFF') {
            $staffRecord = StaffRecord::where('user_id', Auth::user()->id)->first();
            if ($staffRecord && in_array($staffRecord->position, ['MANAGER', 'PRINCIPAL'])) {
                // Get the appointments for the parent's children
                $appointments = AppointmentRecord::with('staff.users', 'child')->get();

                // Pass the users data to a view
                return view('manage-appointment.staff.list-appointment', compact('appointments'));
            }
            if ($staffRecord && in_array($staffRecord->position, ['OCCUPATIONAL THERAPY'])) {
                // Get the appointments for the parent's children
                $appointments = AppointmentRecord::where('staff_id', $staffRecord->id)
                ->get();

                // Pass the users data to a view
                return view('manage-report.ot.appointment', compact('appointments'));
            }
            if ($staffRecord && in_array($staffRecord->position, ['CONSULTANT'])) {
                // Get the appointments for the parent's children
                $appointments = AppointmentRecord::where('staff_id', $staffRecord->id)
                ->get();

                // Pass the users data to a view
                return view('manage-report.csltnt.appointment', compact('appointments'));
            }
        } else {
            // Get the authenticated user (parent)
            $parent = Auth::user()->parent;

            // Get the parent's child IDs
            $childs = $parent->child->pluck('id')->toArray();

            // Get the appointments for the parent's children
            $appointments = AppointmentRecord::with('staff.users', 'child')
                ->whereIn('child_id', $childs)
                ->get();
            // dd($appointments);

            // Pass the users data to a view
            return view('manage-appointment.parent.list-appointment', compact('appointments'));
        }
    }

    public function create()
    {
        $user = auth()->user();
        $parent = $user->parent;
        $childs = $parent->child;

        return view('manage-appointment.parent.add-appointment', compact('childs'));
    }
```

Figure 4.2.57: Appointment Controller (index, create function)

```php
public function store(Request $request)
{
    // Create a new instance of the appointment model
    $ap = new AppointmentRecord();

    // Assign values from the form inputs to the model attributes
    $ap->appntmnt_type = $request->type;
    $ap->appntmnt_time = $request->time;
    $ap->appntmnt_date = $request->date;
    $ap->child_id = $request->input('child_id');

    // Save the model to the database
    $ap->save();

    return redirect()->route('appointment.list')->with('success', 'Successfully Book');
}

public function edit($id)
{
    $appointment = AppointmentRecord::findOrFail($id);
    $staffs = StaffRecord::all();
    $selectedStaff = StaffRecord::find($appointment->staff_id);

    return view('manage-appointment.staff.assign-pic', compact('appointment', 'staffs', 'selectedStaff'));
}

public function update(Request $request, $id)
{
    $appointment = AppointmentRecord::find($id);

    // Update the staff_id with the selected value from the dropdown
    $appointment->staff_id = $request->input('pic');
    $appointment->save();

    return redirect()->route('appointment.list')->with('success', 'Successfully updated.');

}

public function destroy($ap_id)
{
    if (Auth::user()->role === 'ADMIN') {
        $appointments = AppointmentRecord::findOrFail($ap_id);

        // delete the data
        $appointments->delete();

        // Pass the users data to a view
        return redirect()->route('appointment.list')->with('success', 'Successfully Deleted');
    }
    if (Auth::user()->role === 'PARENT') {
        $appointments = AppointmentRecord::findOrFail($ap_id);

        // delete the data
        $appointments->delete();

        // Pass the users data to a view
        return redirect()->route('appointment.list')->with('success', 'Successfully Deleted');
    }
}
```

Figure 4.2.58: Appointment Controller (store, edit, update, destroy function)

This controller (Figure 4.2.57, Figure 4.2.58) is used to view, store, edit, update and delete appointment info.



```
class M3Controller extends Controller
{
    //
    public function index()
    {
        $user = auth()->user();
        $parent = $user->parent;
        $childs = $parent->child;

        $sections = [
            [
                'title' => '1: Fail-fail yang diperlukan:',
                'questions' => [
                    ['m3QID' => 'q1', 'text' => '1. Kad imunisasi anak'],
                    ['m3QID' => 'q2', 'text' => '2. Rekod M-CHAT (sekiranya ada)'],
                    ['m3QID' => 'q3', 'text' => '3. Apa-apa rekod penilaian doktor/terapis yang diperolehi?'],
                ],
            ],
            [
                'title' => '2: Sebelum Anak Lahir:',
                'questions' => [
                    ['m3QID' => 'q4', 'text' => '1. Pengalaman semasa mengandung'],
                    ['m3QID' => 'q5', 'text' => '2. Kandungan ke berapa?'],
                    ['m3QID' => 'q6', 'text' => '3. Umur ibu semasa mengandung  '],
                    ['m3QID' => 'q7', 'text' => '4. Rawatan doktor yang penting?   '],
                    ['m3QID' => 'q8', 'text' => '5. Ada alahan atau Morning sickness? ', 'options' => ['Ya', 'Tidak']],
                ],
            ],
            [
                'title' => '3: Semasa Melahirkan:',
                'questions' => [
                    ['m3QID' => 'q9', 'text' => '1. Pengalaman semasa melahirkan', 'options' => ['Normal', 'Caesarean', 'Vacuum', 'Forcep']],
                    ['m3QID' => 'q10', 'text' => '2. Cukup minggu?', 'options' => ['Ya', 'Tidak']],
                    ['m3QID' => 'q11', 'text' => '3. Keadaan anak selepas dilahirkan
                (Jaundice, Bayi kurang berat, Bayi besar, kumplikasi lain?) '],
                ],
            ],
            [
                'title' => '4: Pengalaman dalan Pantang:',
                'questions' => [
                    ['m3QID' => 'q12', 'text' => '1. Pemakanan ibu'],
                    ['m3QID' => 'q13', 'text' => '2. Penyusuan anak'],
                    ['m3QID' => 'q14', 'text' => '3. Tingkah laku anak (Menangis, Kembung perut dan lain-lain lagi?) '],
                ],
            ],
            [
                'title' => '5: Sejarah Pembesaran anak:',
                'questions' => [
                    ['m3QID' => 'q15', 'text' => '1. Siapa yang jaga anak?', 'options' => ['Ibu bapa', 'Atuk/Nenek', 'Saudara-mara', 'Penjaga']],
                    ['m3QID' => 'q16', 'text' => '2. Ada tingkah laku yang membimbangkan?'],
                    ['m3QID' => 'q17', 'text' => '3. Perbezaan dalam perkembangan anak berbanding anak lain (sekiranya ada)'],
                ],
            ],
            [
                'title' => '6: Tingkah laku anak ketika di rumah sendiri:',
                'questions' => [
                    ['m3QID' => 'q18', 'text' => '1. Cara berkomunikasi dengan ibu dan bapa'],
                    ['m3QID' => 'q19', 'text' => '2. Cara berkomunikasi dengan adik-beradik (sekiranya ada)'],
                    ['m3QID' => 'q20', 'text' => '3. Cara anak bermain di rumah '],
                    ['m3QID' => 'q21', 'text' => '4. Barang permainan yang ada yang anak paling gemari '],
                    ['m3QID' => 'q22', 'text' => '5. Perkara yang anak tidak suka'],
                    ['m3QID' => 'q23', 'text' => '6. Perkara yang anak paling suka '],
                    ['m3QID' => 'q24', 'text' => '7. Sesuatu atau perkara yang anak takuti'],
                    ['m3QID' => 'q25', 'text' => '8. Objek yang anak jadi obsesif '],
                ],
            ],
            [
                'title' => '7: Penggunaan gajet:',
                'questions' => [
                    ['m3QID' => 'q26', 'text' => '1. Namakan gajet yang anak dapat', 'options' => ['Telefon Pintar', 'Komputer Riba', 'Tablet', 'Desktop', 'Lain-lain']],
                    ['m3QID' => 'q27', 'text' => '2. Bila anak mula diberikan gajet?', 'options' => ['Kurang dari 2 tahun', '2 hingga 4 tahun', '4 tahun dan ke atas']],
                    ['m3QID' => 'q28', 'text' => '3. Berapa lama masa anak guna gadget? ', 'options' => ['Kurang dari 1 jam', '1 hingga 4 jam', 'Lebih dari 4 jam']],
                    ['m3QID' => 'q29', 'text' => '4. Apa yang anak paling suka bila mendapat gajet? ', 'options' => ['YouTube', 'Bermain game', 'Belajar', 'Menghubungi seseorang', 'Lain-lain']],
                    ['m3QID' => 'q30', 'text' => '5. Ada kesan buruk di atas penggunaan gajet ini? ', 'options' => ['Ya', 'Tidak']],
                ],
            ],
```

Figure 4.2.59: M3 Controller (index function)

Figure 4.2.60: M3 Controller (store function)

This controller (Figure 4.2.59, Figure 4.2.60) is used to view and store M3 information.

```php
class PaymentController extends Controller
{
    public function index()
    {

        $user = auth()->user();
        $parent = $user->parent;
        $childs = $parent->child;

        // Pass the users data to a view
        return view('manage-payment.payment', compact('childs'));

    }

    public function report()
    {
        if (Auth::user()->role === 'PARENT') {
        // $user = auth()->user();
        // $parent = $user->parent;
        // $childs = $parent->child;
        $parent = Auth::user()->parent;

        // $payments = PaymentRecord::where('child_id', $childs->child_id)->get();
        $childs = ChildRecord::whereHas('payment')
        ->with('payment')
        ->where('parent_id', $parent->id)->get();

        // Pass the users data to a view
        return view('manage-report.parent.payment', compact('childs'));
        }
        if (Auth::user()->role === 'STAFF') {
            $staffRecord = StaffRecord::where('user_id', Auth::user()->id)->first();
            if ($staffRecord && in_array($staffRecord->position, ['MANAGER', 'PRINCIPAL'])) {
                $payments = PaymentRecord::with('child')->get();

                // Pass the users data to a view
                return view('manage-report.staff.payment', compact('payments'));
            }
        }
        if (Auth::user()->role === 'ADMIN') {
            $payments = PaymentRecord::with('child')->get();

            return view('manage-report.admin.payment', compact('payments'));
        }
    }

    public function store(Request $request)
    {
        // Create a new instance of the appointment model
        $pay = new PaymentRecord();

        // Assign values from the form inputs to the model attributes
        $pay->fee_type = $request->fee;
        $pay->total_payment = $request->total;
        $pay->payment_type = $request->pay_type;
        $pay->bank_name = $request->bank_name;
        $pay->payment_remarks = $request->remarks;
        $pay->child_id = $request->input('child_id');
        // Save the model to the database
        $pay->save();

        return redirect()->route('dashboard')->with('success', 'Successfully Pay');
    }
}
```

Figure 4.2.61: Payment Controller (index, report, store function)

This controller (Figure 4.2.61) is used to view and store Payment information.

4.2.4.3 View

The View component of the MVC architecture in the Laravel project of Special Needs Children On-boarding Management system uses HTML, CSS, and JavaScript to retrieve and display data. It provides the GUI code required to show and interact with the system's various features.

Figure 4.2.62: Registration blade

Figure 4.2.63: Registration blade (2)

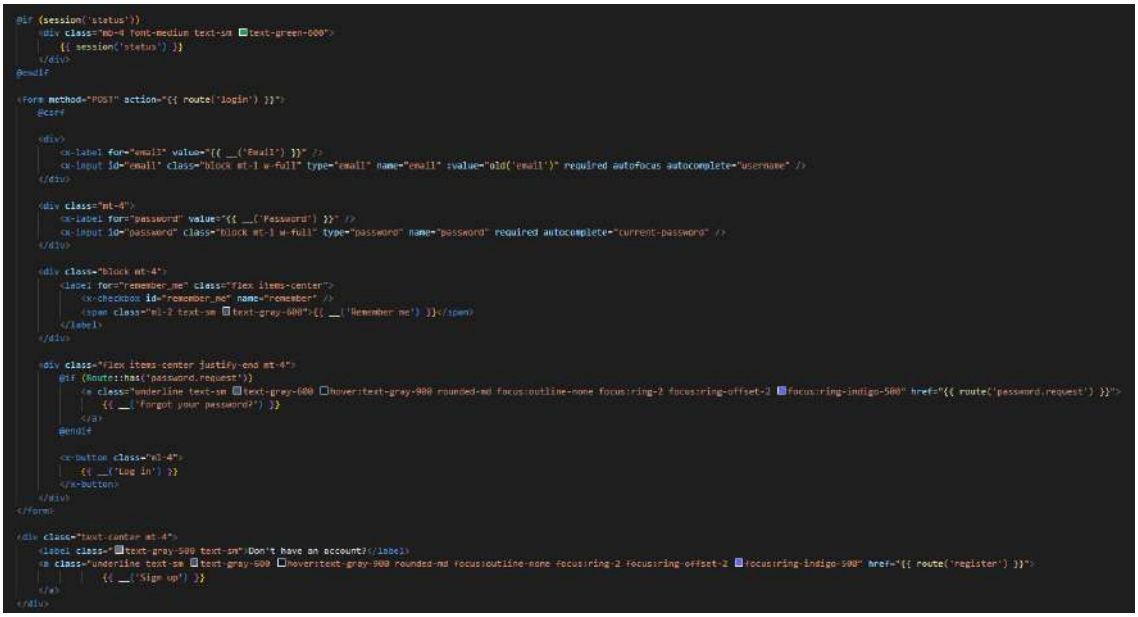


Figure 4.2.64: Login blade

Figure 4.2.65: Add User blade

```
@foreach ($users as $user)
    <form method="POST" action="{{ route('user.update', ['id' => $user->id]) }}">
        @csrf
        <div class="flex gap-4 justify-center">
            <div class="flex-auto w-1/2">

                <div class="mt-4">
                    <x-label for="name" value="{{ __('Name') }}" />
                    <x-input id="name" oninput="validateInputStrings(this)"
                        class="uppercase block mt-1 w-full" type="text" name="name"
                        value="{{ $user->name }}" autofocus autocomplete="name" />
                </div>

                <div class="mt-4">
                    <x-label for="ic" value="{{ __('IC Number') }}" />
                    <x-input id="ic" oninput="validateInputIC(this)" class="block mt-1 w-full"
                        type="text" name="ic"
                        value="{{ $user->staff->ic ?? $user->parent->p_ic }}" autofocus
                        autocomplete="ic" />
                </div>
                @if ($user->role === 'STAFF')
                    <div class="mt-4">
                        <x-label value="{{ __('Gender') }}" />
                        <div class="pt-2">
                            <input id="male" type="radio" name="gender" value="MALE"
                                {{ $user->staff->gender === 'MALE' ? 'checked' : '' }}
                                @if (old('gender') == 'MALE') checked @endif />
                            <label class="pr-4" for="male"> Male </label>
                            <input id="female" type="radio" name="gender" value="FEMALE"
                                {{ $user->staff->gender === 'FEMALE' ? 'checked' : '' }}
                                @if (old('gender') == 'FEMALE') checked @endif />
                            <label class="pr-4" for="female"> Female </label>
                        </div>
                    </div>
                @endif

                @if ($user->role === 'PARENT')
                    <div class="mt-4">
                        <x-label for="p_relation" value="{{ __('Family Relation') }}" />
                        <select id="p_relation"
                            class="block mt-1 w-full border-gray-300 focus:border-indigo-500 focus:ring-indigo-500 rounded-md shadow-sm"
                            name="p_relation" required>
                            <option value="MOTHER"
                                {{ $user->parent->p_relation == 'MOTHER' ? 'selected' : '' }}
                                @if (old('p_relation') == 'MOTHER') selected @endif>Mother</option>
                            <option value="FATHER"
                                {{ $user->parent->p_relation == 'FATHER' ? 'selected' : '' }}
                                @if (old('p_relation') == 'FATHER') selected @endif>Father</option>
                            <option value="GRANDPARENT"
                                {{ $user->parent->p_relation == 'GRANDPARENT' ? 'selected' : '' }}
                                @if (old('p_relation') == 'GRANDPARENT') selected @endif>Grandparent
                            </option>
                            <option value="GUARDIAN"
                                {{ $user->parent->p_relation == 'GUARDIAN' ? 'selected' : '' }}
                                @if (old('p_relation') == 'GUARDIAN') selected @endif>Guardian</option>
                        </select>
                    </div>
                @endif
```

Figure 4.2.66: Edit User blade

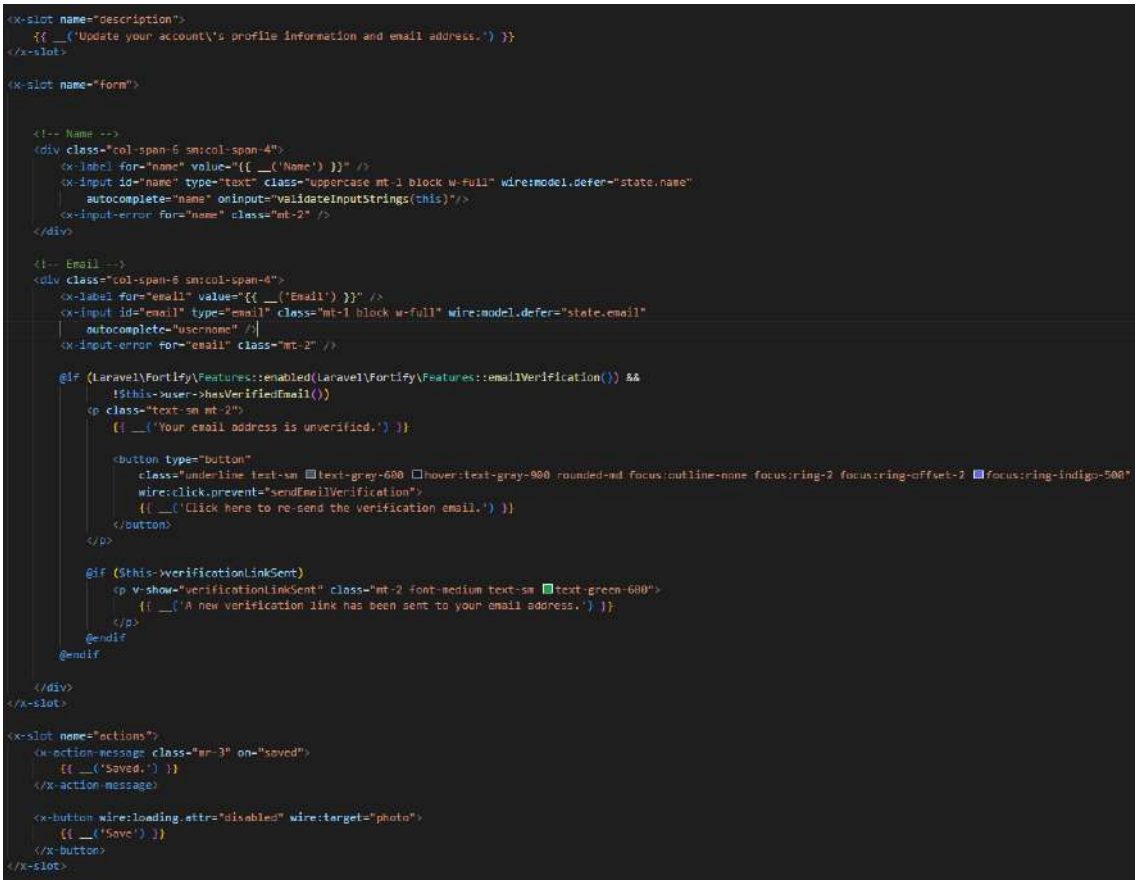


Figure 4.2.67: Manage User Account blade

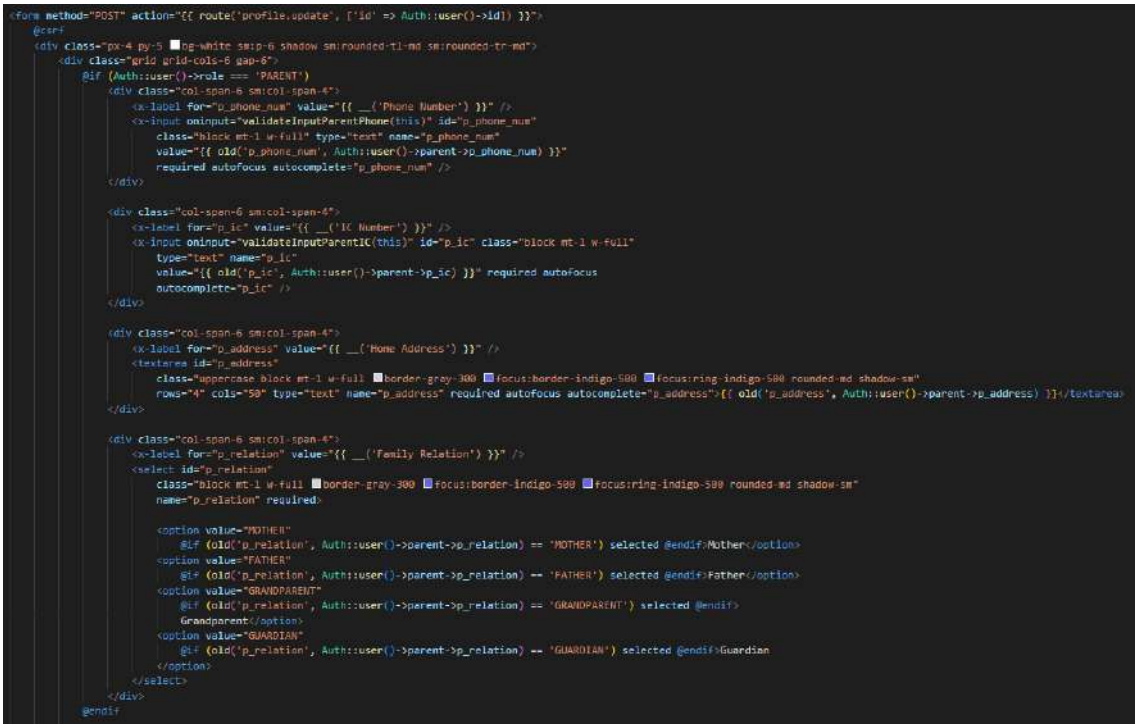


Figure 4.2.68: Manage User Profile blade

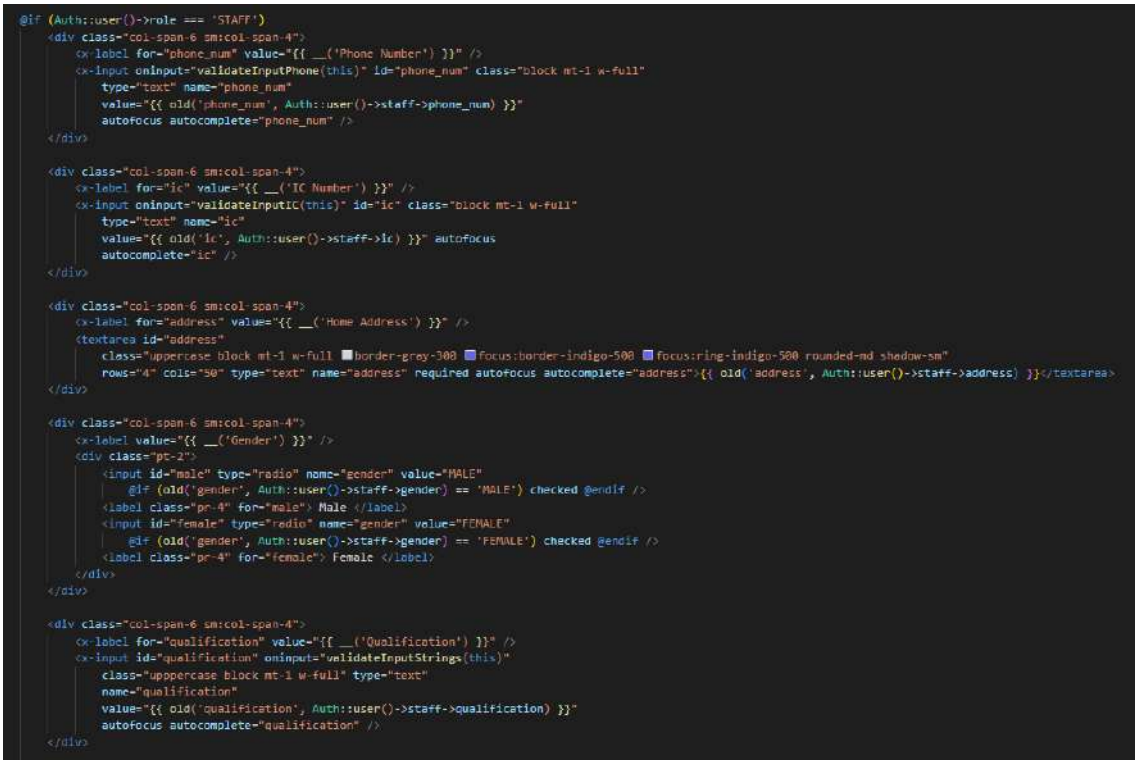


Figure 4.2.69: Manage User Profile blade (2)

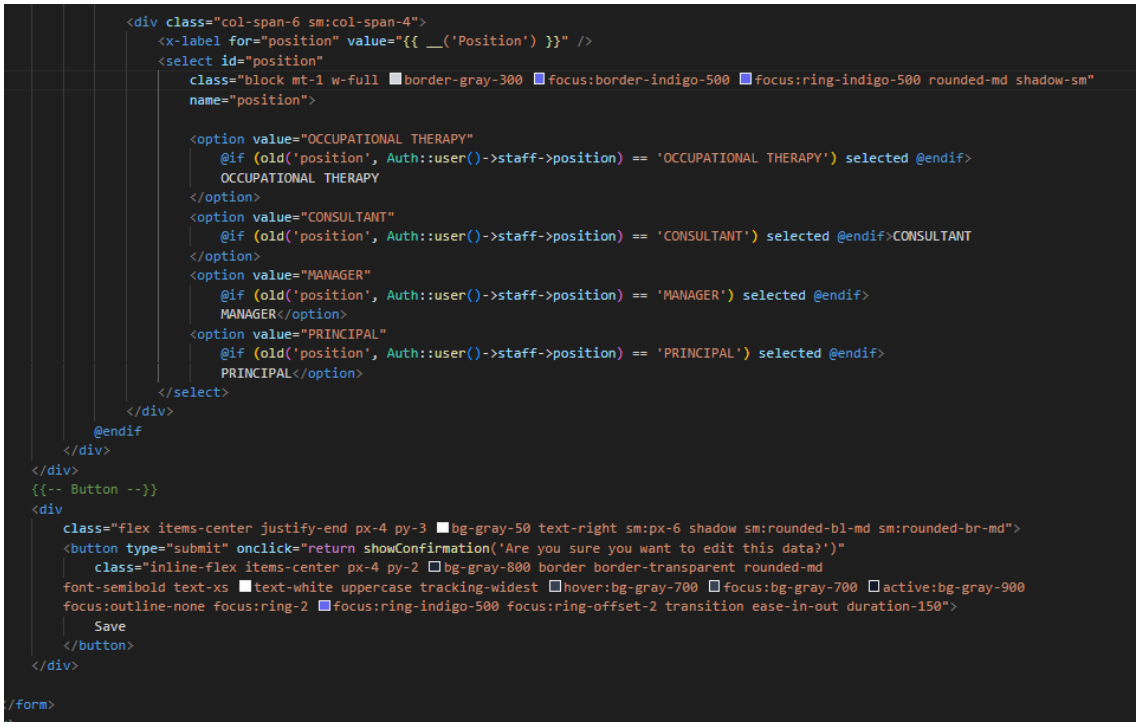
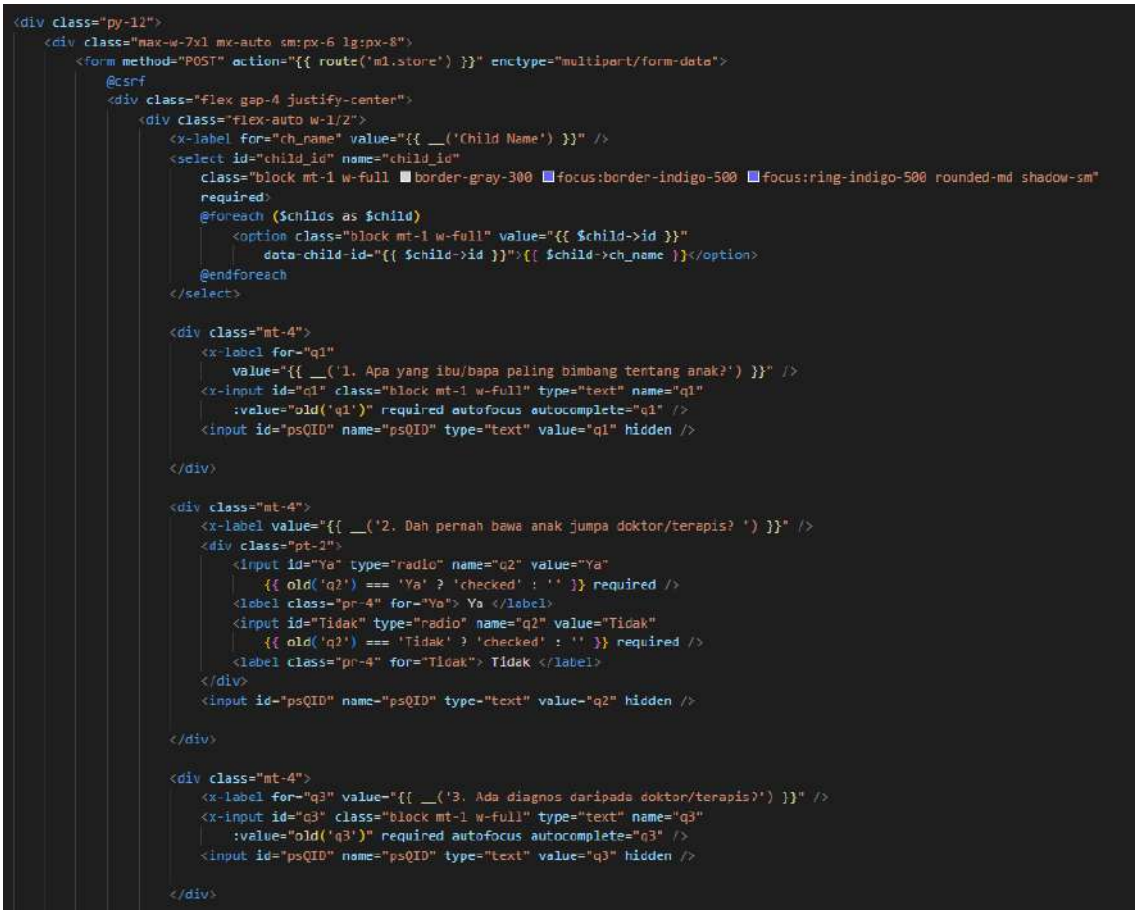


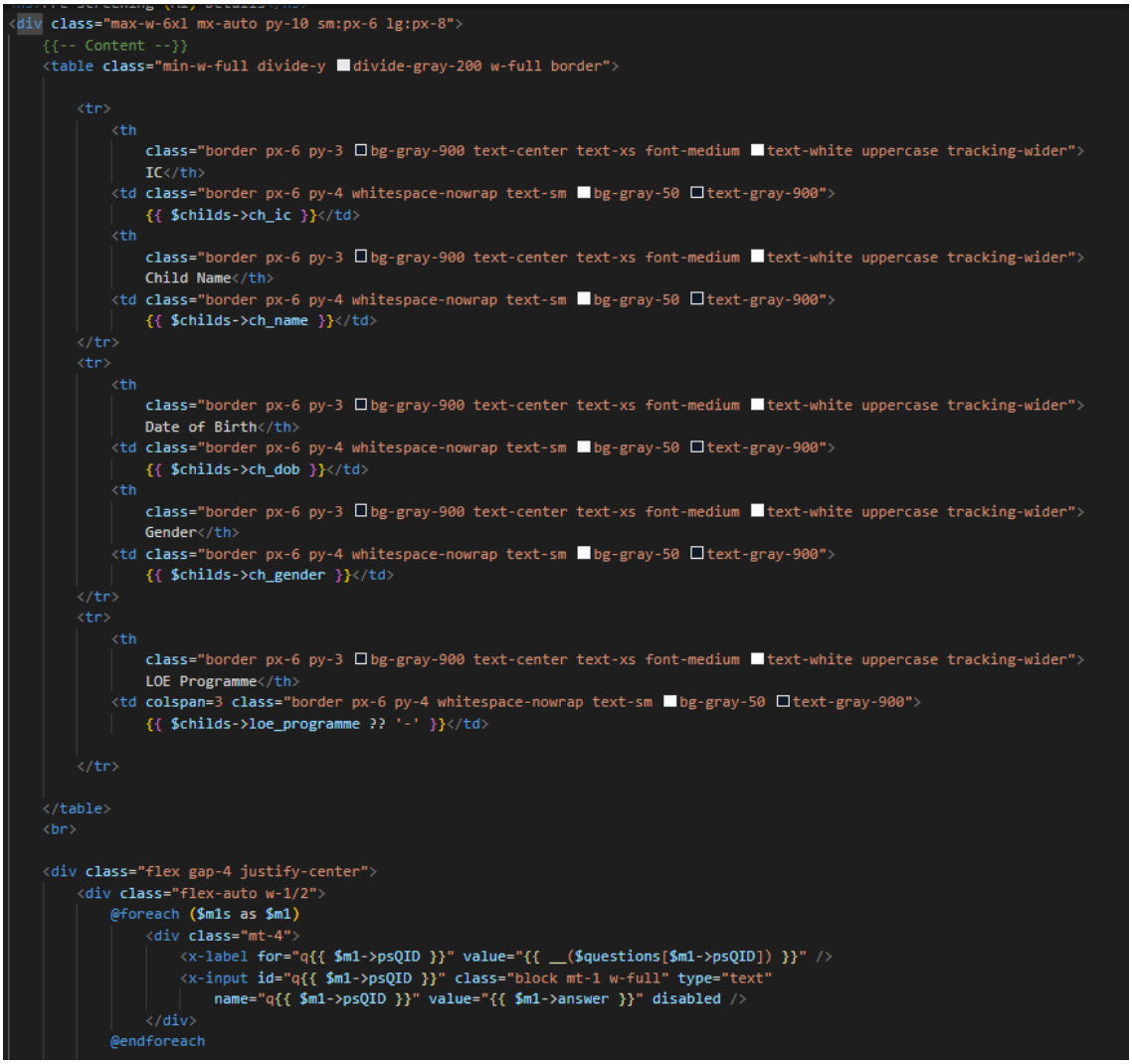
Figure 4.2.70: Manage User Profile blade (3)



```
<div class="py-12">
    <div class="max-w-7xl mx-auto sm:px-6 lg:px-8">
        <form method="POST" action="{{ route('m1.store') }}" enctype="multipart/form-data">
            @csrf
            <div class="flex gap-4 justify-center">
                <div class="flex-auto w-1/2">
                    <x-label for="ch_name" value="{{ __('Child Name') }}" />
                    <select id="child_id" name="child_id"
                        class="block mt-1 w-full border-gray-300 focus:border-indigo-500 focus:ring-indigo-500 rounded-md shadow-sm"
                        required>
                        @foreach ($childs as $child)
                            <option class="block mt-1 w-full" value="{{ $child->id }}"
                                data-child-id="{{ $child->id }}">{{ $child->ch_name }}</option>
                        @endforeach
                    </select>

                    <div class="mt-4">
                        <x-label for="q1"
                            value="{{ __('1. Apa yang ibu/bapa paling bimbang tentang anak?') }}" />
                        <x-input id="q1" class="block mt-1 w-full" type="text" name="q1"
                            :value="old('q1')" required autofocus autocomplete="q1" />
                        <input id="psQID" name="psQID" type="text" value="q1" hidden />

                    </div>

                    <div class="mt-4">
                        <x-label value="{{ __('2. Dah pernah bawa anak jumpa doktor/terapis? ') }}" />
                        <div class="pt-2">
                            <input id="Ya" type="radio" name="q2" value="Ya"
                                {{ old('q2') === 'Ya' ? 'checked' : '' }} required />
                            <label class="pr-4" for="Ya"> Ya </label>
                            <input id="Tidak" type="radio" name="q2" value="Tidak"
                                {{ old('q2') === 'Tidak' ? 'checked' : '' }} required />
                            <label class="pr-4" for="Tidak"> Tidak </label>
                        </div>
                        <input id="psQID" name="psQID" type="text" value="q2" hidden />

                    </div>

                    <div class="mt-4">
                        <x-label for="q3" value="{{ __('3. Ada diagnos daripada doktor/terapis?') }}" />
                        <x-input id="q3" class="block mt-1 w-full" type="text" name="q3"
                            :value="old('q3')" required autofocus autocomplete="q3" />
                        <input id="psQID" name="psQID" type="text" value="q3" hidden />

                    </div>
```

Figure 4.2.71: Manage Pre-screening (M1) blade

Figure 4.2.72: Manage Pre-screening (M1) blade (2)



```
<div class="max-w-6xl mx-auto py-10 sm:px-6 lg:px-8">
    {{-- Content --}}
    <table class="min-w-full divide-y divide-gray-200 w-full border">

        <tr>
            <th
                class="border px-6 py-3 bg-gray-900 text-center text-xs font-medium text-white uppercase tracking-wider">
                IC</th>
            <td class="border px-6 py-4 whitespace-nowrap text-sm bg-gray-50 text-gray-900">
                {{ $childs->ch_ic }}</td>
            <th
                class="border px-6 py-3 bg-gray-900 text-center text-xs font-medium text-white uppercase tracking-wider">
                Child Name</th>
            <td class="border px-6 py-4 whitespace-nowrap text-sm bg-gray-50 text-gray-900">
                {{ $childs->ch_name }}</td>
        </tr>
        <tr>
            <th
                class="border px-6 py-3 bg-gray-900 text-center text-xs font-medium text-white uppercase tracking-wider">
                Date of Birth</th>
            <td class="border px-6 py-4 whitespace-nowrap text-sm bg-gray-50 text-gray-900">
                {{ $childs->ch_dob }}</td>
            <th
                class="border px-6 py-3 bg-gray-900 text-center text-xs font-medium text-white uppercase tracking-wider">
                Gender</th>
            <td class="border px-6 py-4 whitespace-nowrap text-sm bg-gray-50 text-gray-900">
                {{ $childs->ch_gender }}</td>
        </tr>
        <tr>
            <th
                class="border px-6 py-3 bg-gray-900 text-center text-xs font-medium text-white uppercase tracking-wider">
                LOE Programme</th>
            <td colspan=3 class="border px-6 py-4 whitespace-nowrap text-sm bg-gray-50 text-gray-900">
                {{ $childs->loe_programme ?? '-' }}</td>

        </tr>

    </table>
    <br>

    <div class="flex gap-4 justify-center">
        <div class="flex-auto w-1/2">
            @foreach ($m1s as $m1)
                <div class="mt-4">
                    <x-label for="q{{ $m1->psQID }}" value="{{ __($questions[$m1->psQID]) }}" />
                    <x-input id="q{{ $m1->psQID }}" class="block mt-1 w-full" type="text"
                        name="q{{ $m1->psQID }}" value="{{ $m1->answer }}" disabled />
                </div>
            @endforeach
```

Figure 4.2.73: Manage Pre-screening (M1) (Update LOE Programme) blade

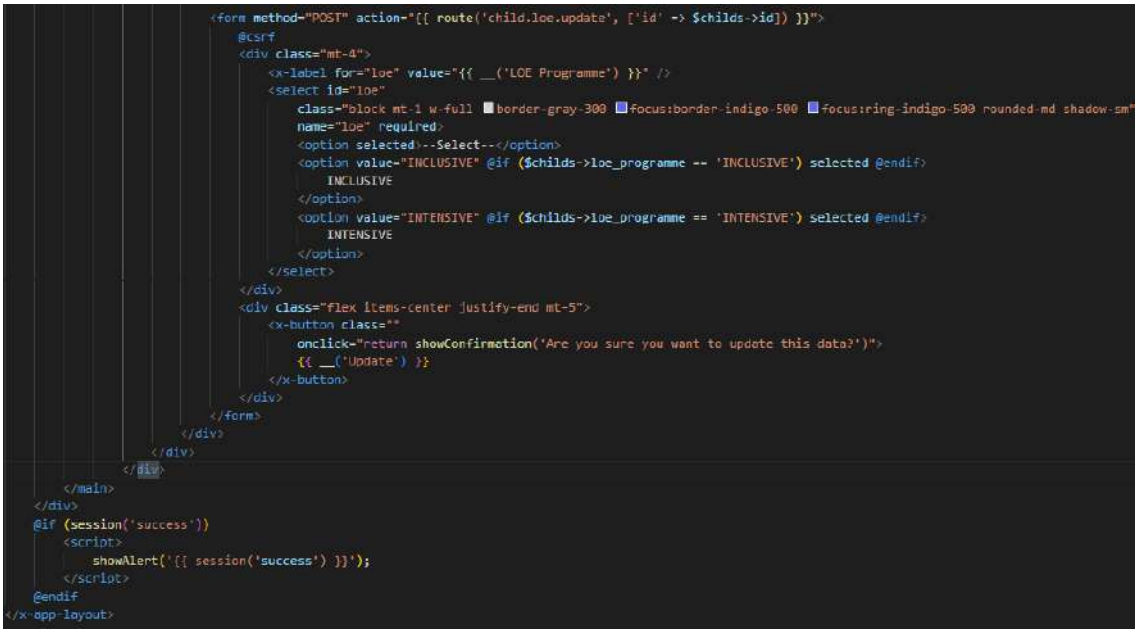


Figure 4.2.74: Manage Pre-screening (M1) (Update LOE Programme) blade (2)

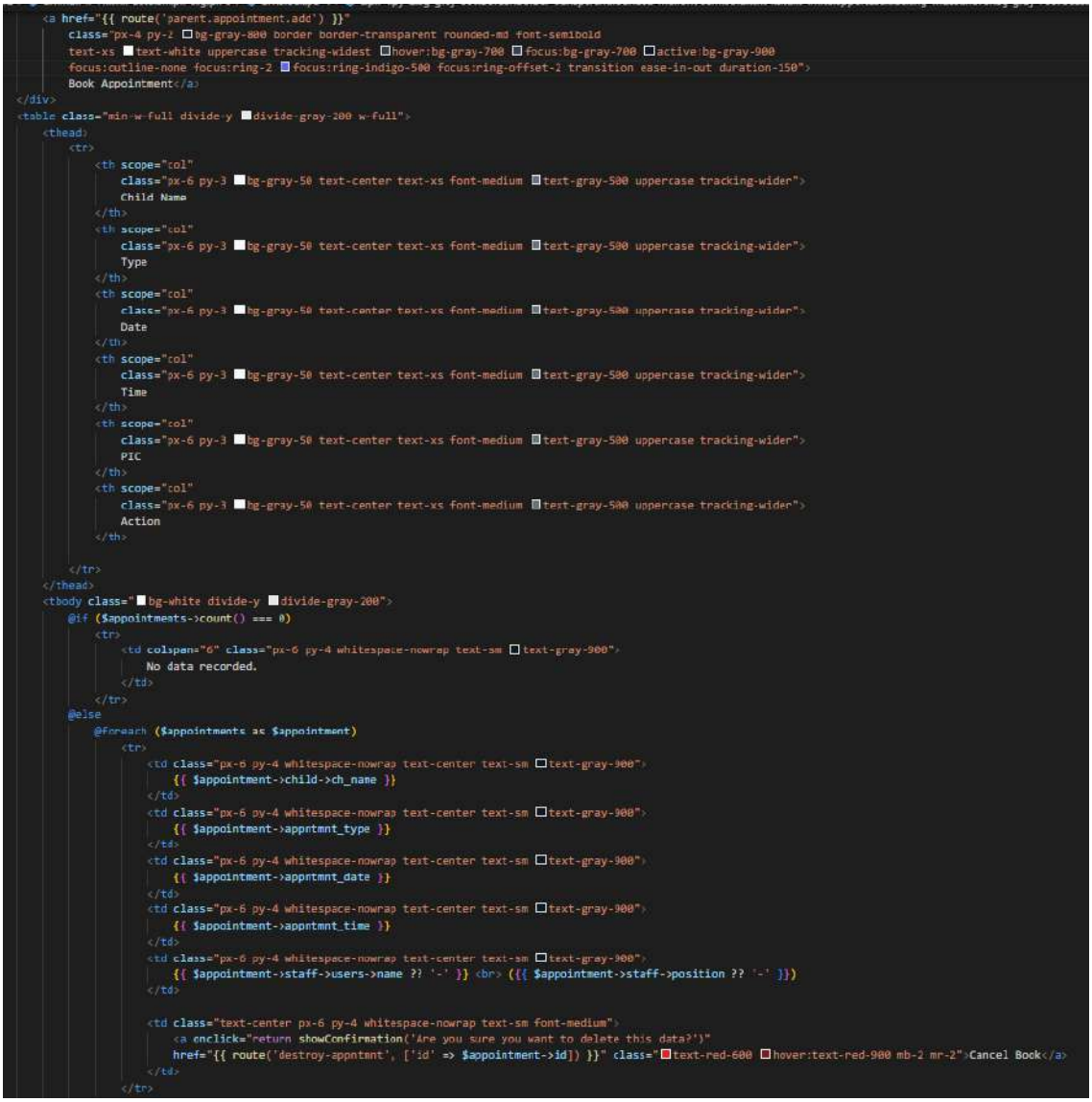


Figure 4.2.75: Manage Appointment blade

```
<form method="POST" action="{{ route('parent.appointment.store') }}">
    @csrf
    <div class="flex gap-4 justify-center">
        <div class="flex-auto w-1/2">
            <br>
            <x-label for="ch_name" value="{{ __('Child Name') }}" />
            <select id="child_id" name="child_id"
                class="block mt-1 w-full border-gray-300 focus:border-indigo-500 focus:ring-indigo-500 rounded-md shadow-sm"
                required>
                @foreach ($childs as $child)
                    <option class="block mt-1 w-full" value="{{ $child->id }}"
                        data-child-id="{{ $child->id }}">{{ $child->ch_name }}</option>
                @endforeach
            </select>

            <div class="mt-4">
                <x-label for="type" value="{{ __('Type') }}" />
                <select id="type"
                    class="block mt-1 w-full border-gray-300 focus:border-indigo-500 focus:ring-indigo-500 rounded-md shadow-sm"
                    name="type" required>
                    <option selected>--Select--</option>
                    <option value="PERJUMPAAN PERTAMA (P1)"
                        @if (old('type') == 'PERJUMPAAN PERTAMA (P1)') selected @endif>PERJUMPAAN PERTAMA (P1)
                    </option>
                    <option value="PERJUMPAAN KEDUA (P2)"
                        @if (old('type') == 'PERJUMPAAN KEDUA (P2)') selected @endif>PERJUMPAAN KEDUA (P2)</option>
                </select>
            </div>

            <div class="mt-4">
                <x-label for="date" value="{{ __('Date') }}" />
                <x-input id="date" class="block mt-1 w-full" type="date" name="date"
                    :value="old('date')" required autofocus autocomplete="date" />
            </div>

            <div class="mt-4">
                <x-label for="time" value="{{ __('Time') }}" />
                <select id="time"
                    class="block mt-1 w-full border-gray-300 focus:border-indigo-500 focus:ring-indigo-500 rounded-md shadow-sm"
                    name="time" required>
                    <option selected>--Select--</option>
                    <option value="08:30 AM" @if (old('time') == '08:30 AM') selected @endif>08:30 AM
                    </option>
                    <option value="09:00 AM" @if (old('time') == '09:00 AM') selected @endif>09:00 AM
                    </option>
                    <option value="09:30 AM" @if (old('time') == '09:30 AM') selected @endif>09:30 AM
                    </option>
                    <option value="10:00 AM" @if (old('time') == '10:00 AM') selected @endif>10:00 AM
                    </option>
                    <option value="10:30 AM" @if (old('time') == '10:30 AM') selected @endif>10:30 AM
                    </option>
                    <option value="11:00 AM" @if (old('time') == '11:00 AM') selected @endif>11:00 AM
                    </option>
                    <option value="02:30 PM" @if (old('time') == '02:30 PM') selected @endif>02:30 PM
                    </option>
                    <option value="03:00 PM" @if (old('time') == '03:00 PM') selected @endif>03:00 PM
                    </option>
                    <option value="03:30 PM" @if (old('time') == '03:30 PM') selected @endif>03:30
                        PM</option>
                    <option value="04:00 PM" @if (old('time') == '04:00 PM') selected @endif>04:00
                        PM</option>
                </select>
            </div>
```

Figure 4.2.76: Add Appointment blade

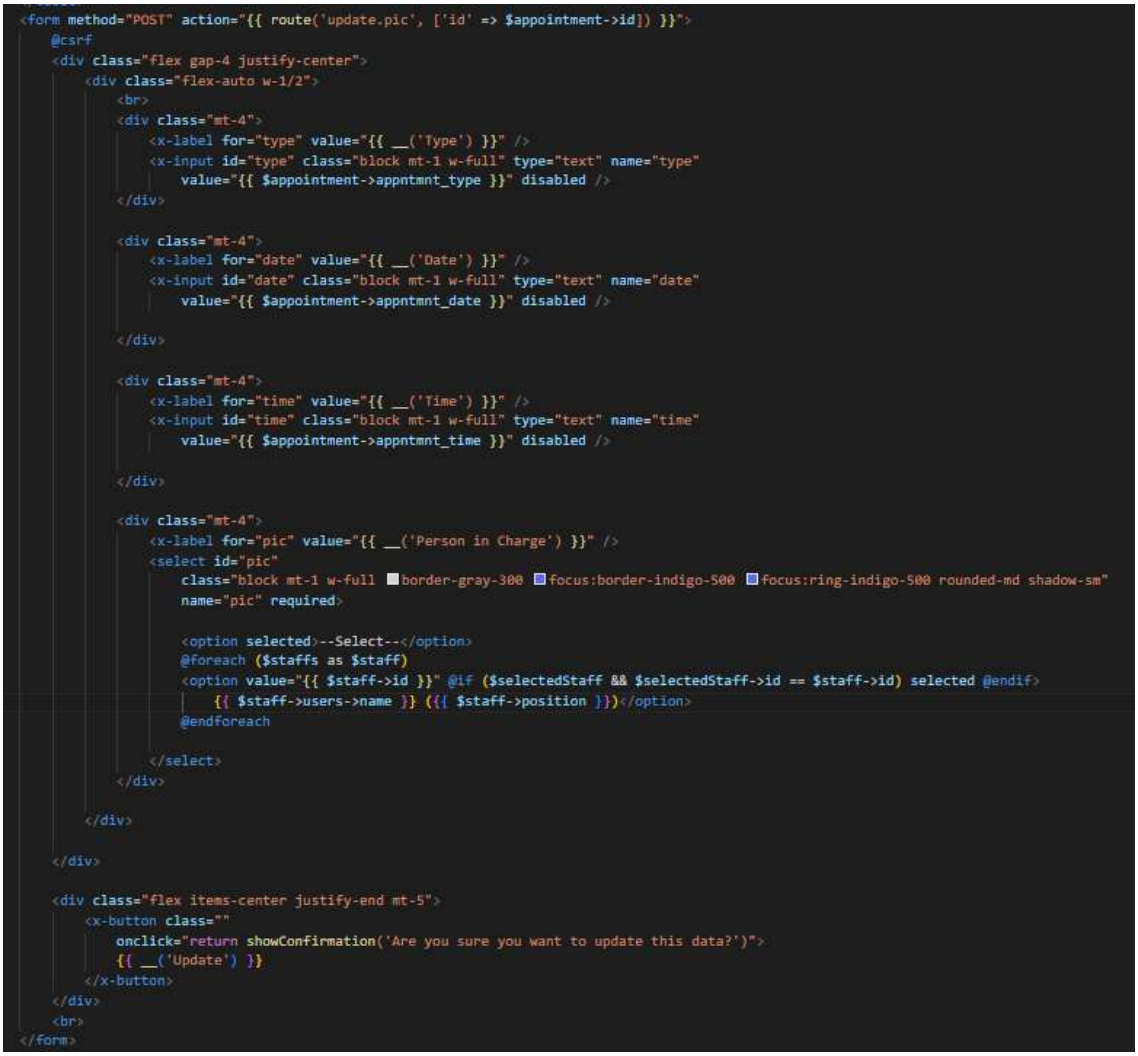


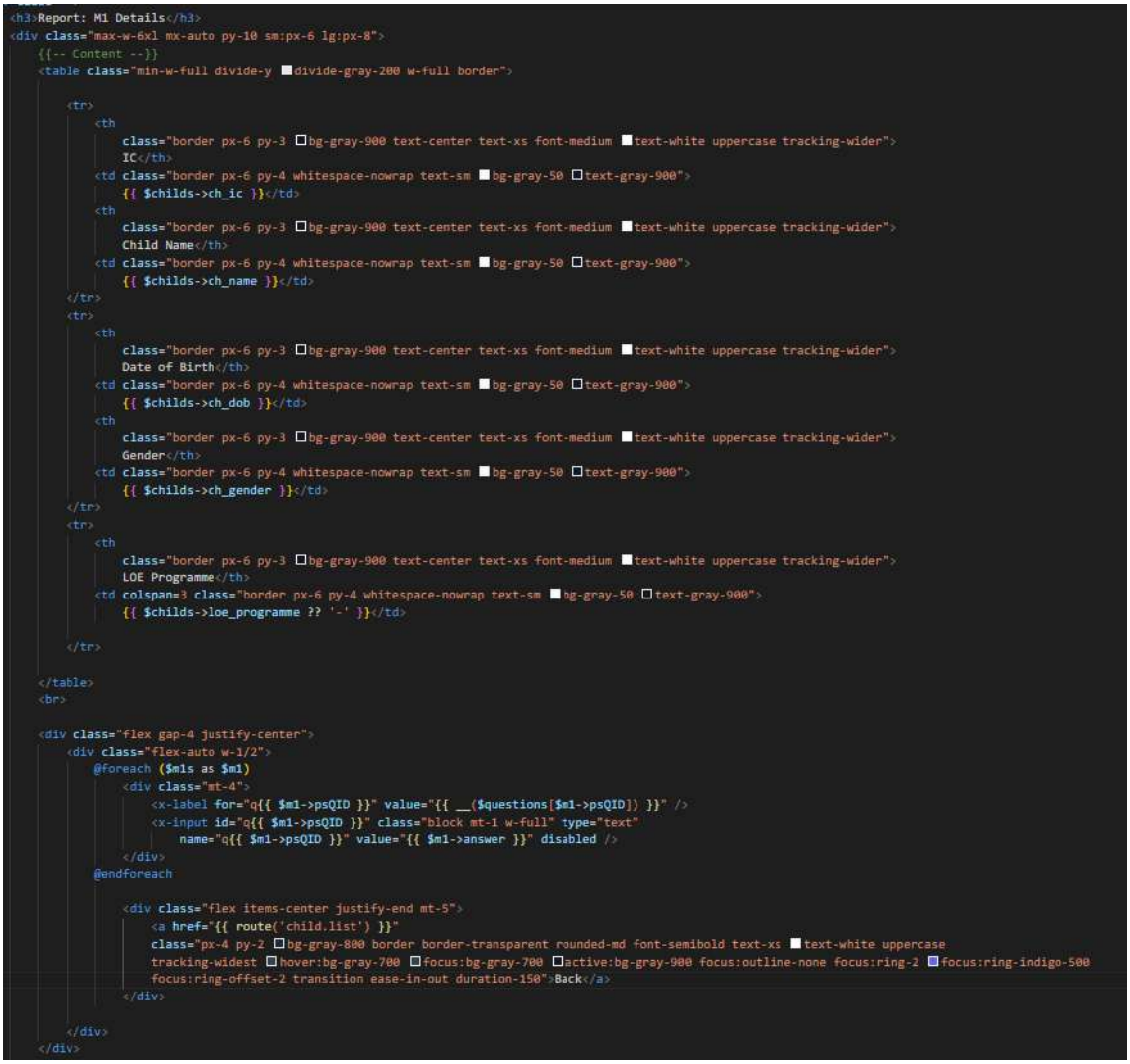
Figure 4.2.77: Assign PIC for Appointment blade

```
<form method="POST" action="{{ route('m3.store') }}" enctype="multipart/form-data">
    @csrf
    <div class="flex gap-4 justify-center">
        <div class="flex-auto w-1/2">
            <x-label for="ch_name" value="{{ __('Child Name') }}" />
            <select id="child_id" name="child_id"
                class="block mt-1 w-full border-gray-300 focus:border-indigo-500 focus:ring-indigo-500 rounded-md shadow-sm"
                required>
                @foreach ($childs as $child)
                    <option class="block mt-1 w-full" value="{{ $child->id }}"
                        data-child-id="{{ $child->id }}">{{ $child->ch_name }}</option>
                @endforeach
            </select>
            <br>

            @foreach ($sections as $section)
                <div class="mt-4">
                    <h4 class="font-semibold text-lg text-gray-800 leading-tight">{{ $section['title'] }}
                    </h4>
                    <br>
                    @foreach ($section['questions'] as $question)
                        <div class="question">
                            <x-label for="question_{{ $question['m3QID'] }}" /> {{ $question['text'] }}
                            <br>
                            @if (in_array($question['m3QID'], ['q1']))
                                <!-- File input -->
                                <input type="file" class="block mt-1 w-full"
                                    id="question_{{ $question['m3QID'] }}" name="{{ $question['m3QID'] }}"
                                    required />
                            @elseif (in_array($question['m3QID'], ['q2', 'q3']))
                                <!-- File input -->
                                <input type="file" class="block mt-1 w-full"
                                    id="question_{{ $question['m3QID'] }}"
                                    name="{{ $question['m3QID'] }}" />
                            @elseif (in_array($question['m3QID'], ['q8', 'q10', 'q27', 'q28', 'q30']))
                                <!-- Radio button input -->
                                @foreach ($question['options'] as $option)
                                    <div class="pt-4 inline">
                                        <input type="radio" id="question_{{ $question['m3QID'] }}"
                                            name="{{ $question['m3QID'] }}" value="{{ $option }}"
                                            required />
                                        <label class="pr-4"
                                            for="question_{{ $question['m3QID'] }}">{{ $option }}</label>
                                    </div>
                                @endforeach
                            @elseif (in_array($question['m3QID'], ['q9', 'q15']))
                                <!-- Dropdown menu -->
                                <select
                                    class="block mt-1 w-full border-gray-300 focus:border-indigo-500 focus:ring-indigo-500 rounded-md shadow-sm"
                                    id="question_{{ $question['m3QID'] }}" name="{{ $question['m3QID'] }}"
                                    required>
                                    @foreach ($question['options'] as $option)
                                        <option value="{{ $option }}">{{ $option }}</option>
                                    @endforeach
                                </select>
                            @elseif (in_array($question['m3QID'], ['q26', 'q29']))
                                <!-- Checkbox input -->
                                @foreach ($question['options'] as $option)
                                    <div class="pt-4 inline">
                                        <input type="checkbox"
                                            id="question_{{ $question['m3QID'] }}_{{ $loop->index }}"
                                            name="{{ $question['m3QID'] }}[]"
                                            value="{{ $option }}" />
                                        <label class="pr-4"
                                            for="question_{{ $question['m3QID'] }}_{{ $loop->index }}">{{ $option }}</label>
                                    </div>
                                @endforeach
```

Figure 4.2.78: Manage Parent Initial Info (M3)

```
<form method="POST" action="{{ route('payment.store') }}">
    @csrf
    <div class="flex gap-4 justify-center">
        <div class="flex-auto w-1/2">
            <br>
            <x-label for="ch_name" value="{{ __('Child Name') }}" />
            <select id="child_id" name="child_id"
                class="block mt-1 w-full border-gray-300 focus:border-indigo-500 focus:ring-indigo-500 rounded-md shadow-sm"
                required>
                @foreach ($childs as $child)
                    <option class="block mt-1 w-full" value="{{ $child->id }}"
                        data-child-id="{{ $child->id }}">{{ $child->ch_name }}</option>
                @endforeach
            </select>

            <div class="mt-4">
                <x-label for="fee" value="{{ __('Fee Type') }}" />
                <select id="fee" onchange="updateTotalPayment()"
                    class="block mt-1 w-full border-gray-300 focus:border-indigo-500 focus:ring-indigo-500 rounded-md shadow-sm"
                    name="fee" required>
                    <option selected>--Select--</option>
                    <option value="PROGRAM PEMERHATIAN SEBULAN (PPS)"
                        @if (old('fee') == 'PROGRAM PEMERHATIAN SEBULAN (PPS)') selected @endif>PROGRAM PEMERHATIAN SEBULAN (PPS)
                    </option>
                    <option value="YURAN"
                        @if (old('fee') == 'YURAN') selected @endif>YURAN</option>
                </select>
            </div>

            <div class="mt-4">
                <x-label for="total" value="{{ __('Total Payment') }}" />
                <x-input id="total" class="block mt-1 w-full" type="text" name="total" :value="old('total')" readonly
                    required />
            </div>

            <div class="mt-4">
                <x-label for="pay_type" value="{{ __('Payment Type') }}" />
                <select id="pay_type"
                    class="block mt-1 w-full border-gray-300 focus:border-indigo-500 focus:ring-indigo-500 rounded-md shadow-sm"
                    name="pay_type" required>
                    <option selected>--Select--</option>
                    <option value="CREDIT/DEBIT CARD"
                        @if (old('pay_type') == 'CREDIT/DEBIT CARD') selected @endif>CREDIT/DEBIT CARD
                    </option>
                    <option value="ONLINE BANK-IN"
                        @if (old('pay_type') == 'ONLINE BANK-IN') selected @endif>ONLINE BANK-IN</option>
                </select>
            </div>

            <div class="mt-4">
                <x-label for="bank_name" value="{{ __('Bank Name') }}" />
                <select id="bank_name"
                    class="block mt-1 w-full border-gray-300 focus:border-indigo-500 focus:ring-indigo-500 rounded-md shadow-sm"
                    name="bank_name" required>
                    <option selected>--Select--</option>
                    <option value="BIMB" @if (old('bank_name') == 'BIMB') selected @endif>BIMB
                    </option>
                    <option value="MAYBANK" @if (old('bank_name') == 'MAYBANK') selected @endif>MAYBANK
                    </option>
                    <option value="RHB" @if (old('bank_name') == 'RHB') selected @endif>RHB
                    </option>
                    <option value="CIMB" @if (old('bank_name') == 'CIMB') selected @endif>CIMB
                    </option>
                    <option value="BSN" @if (old('bank_name') == 'BSN') selected @endif>BSN
```

Figure 4.2.79: Manage Payment



```
<h3>Report: M1 Details</h3>
<div class="max-w-6xl mx-auto py-10 sm:px-6 lg:px-8">
    {{-- Content --}}
    <table class="min-w-full divide-y divide-gray-200 w-full border">

        <tr>
            <th
                class="border px-6 py-3 bg-gray-900 text-center text-xs font-medium text-white uppercase tracking-wider">
                IC</th>
            <td class="border px-6 py-4 whitespace-nowrap text-sm bg-gray-50 text-gray-900">
                {{ $childs->ch_ic }}</td>
            <th
                class="border px-6 py-3 bg-gray-900 text-center text-xs font-medium text-white uppercase tracking-wider">
                Child Name</th>
            <td class="border px-6 py-4 whitespace-nowrap text-sm bg-gray-50 text-gray-900">
                {{ $childs->ch_name }}</td>
        </tr>
        <tr>
            <th
                class="border px-6 py-3 bg-gray-900 text-center text-xs font-medium text-white uppercase tracking-wider">
                Date of Birth</th>
            <td class="border px-6 py-4 whitespace-nowrap text-sm bg-gray-50 text-gray-900">
                {{ $childs->ch_dob }}</td>
            <th
                class="border px-6 py-3 bg-gray-900 text-center text-xs font-medium text-white uppercase tracking-wider">
                Gender</th>
            <td class="border px-6 py-4 whitespace-nowrap text-sm bg-gray-50 text-gray-900">
                {{ $childs->ch_gender }}</td>
        </tr>
        <tr>
            <th
                class="border px-6 py-3 bg-gray-900 text-center text-xs font-medium text-white uppercase tracking-wider">
                LOE Programme</th>
            <td colspan=3 class="border px-6 py-4 whitespace-nowrap text-sm bg-gray-50 text-gray-900">
                {{ $childs->loe_programme ?? '-' }}</td>

        </tr>

    </table>
    <br>

    <div class="flex gap-4 justify-center">
        <div class="flex-auto w-1/2">
            @foreach ($m1s as $m1)
                <div class="mt-4">
                    <x-label for="q{{ $m1->psQID }}" value="{{ __($questions[$m1->psQID]) }}" />
                    <x-input id="q{{ $m1->psQID }}" class="block mt-1 w-full" type="text"
                        name="q{{ $m1->psQID }}" value="{{ $m1->answer }}" disabled />
                </div>
            @endforeach

                <div class="flex items-center justify-end mt-5">
                    <a href="{{ route('child.list') }}"
                    class="px-4 py-2 bg-gray-800 border border-transparent rounded-md font-semibold text-xs text-white uppercase
                    tracking-widest hover:bg-gray-700 focus:bg-gray-700 active:bg-gray-900 focus:outline-none focus:ring-2 focus:ring-indigo-500
                    focus:ring-offset-2 transition ease-in-out duration-150">Back</a>
                </div>

        </div>
    </div>
```

Figure 4.2.80: M1 report blade

```
<table class="min-w-full divide-y divide-gray-200 w-full">
    <thead>
        <tr>
            <th scope="col"
                class="px-6 py-3 bg-gray-50 text-center text-xs font-medium text-gray-500 uppercase tracking-wider">
                IC
            </th>
            <th scope="col"
                class="px-6 py-3 bg-gray-50 text-center text-xs font-medium text-gray-500 uppercase tracking-wider">
                Name
            </th>
            <th scope="col"
                class="px-6 py-3 bg-gray-50 text-center text-xs font-medium text-gray-500 uppercase tracking-wider">
                Fee Type
            </th>
            <th scope="col"
                class="px-6 py-3 bg-gray-50 text-center text-xs font-medium text-gray-500 uppercase tracking-wider">
                Total Amount
            </th>
            <th scope="col"
                class="px-6 py-3 bg-gray-50 text-center text-xs font-medium text-gray-500 uppercase tracking-wider">
                Payment Type
            </th>
            <th scope="col"
                class="px-6 py-3 bg-gray-50 text-center text-xs font-medium text-gray-500 uppercase tracking-wider">
                Bank Name
            </th>
            <th scope="col"
                class="px-6 py-3 bg-gray-50 text-center text-xs font-medium text-gray-500 uppercase tracking-wider">
                Remarks
            </th>

        </tr>
    </thead>
    <tbody class="bg-white divide-y divide-gray-200">
        @if ($payments->isEmpty())
            <tr>
                <td colspan="7" class="px-6 py-4 whitespace-nowrap text-sm text-gray-900">
                    No data recorded.
                </td>
            </tr>
        @else
            @foreach ($payments as $payment)
                <tr>
                    <td class="px-6 py-4 whitespace-nowrap text-center text-sm text-gray-900">
                        {{ $payment->child->ch_ic }}
                    </td>
                    <td class="px-6 py-4 whitespace-nowrap text-center text-sm text-gray-900">
                        {{ $payment->child->ch_name }}
                    </td>
                    <td class="px-6 py-4 whitespace-nowrap text-center text-sm text-gray-900">
                        {{ $payment->fee_type }}
                    </td>
                    <td class="px-6 py-4 whitespace-nowrap text-center text-sm text-gray-900">
                        RM {{ $payment->total_payment }}.00
                    </td>
                    <td class="px-6 py-4 whitespace-nowrap text-center text-sm text-gray-900">
                        {{ $payment->payment_type }}
                    </td>
                    <td class="px-6 py-4 whitespace-nowrap text-center text-sm text-gray-900">
                        {{ $payment->bank_name }}
                    </td>
                    <td class="px-6 py-4 whitespace-nowrap text-center text-sm text-gray-900">
                        {{ $payment->payment_remarks }}
                    </td>
```

Figure 4.2.81: Payment report blade

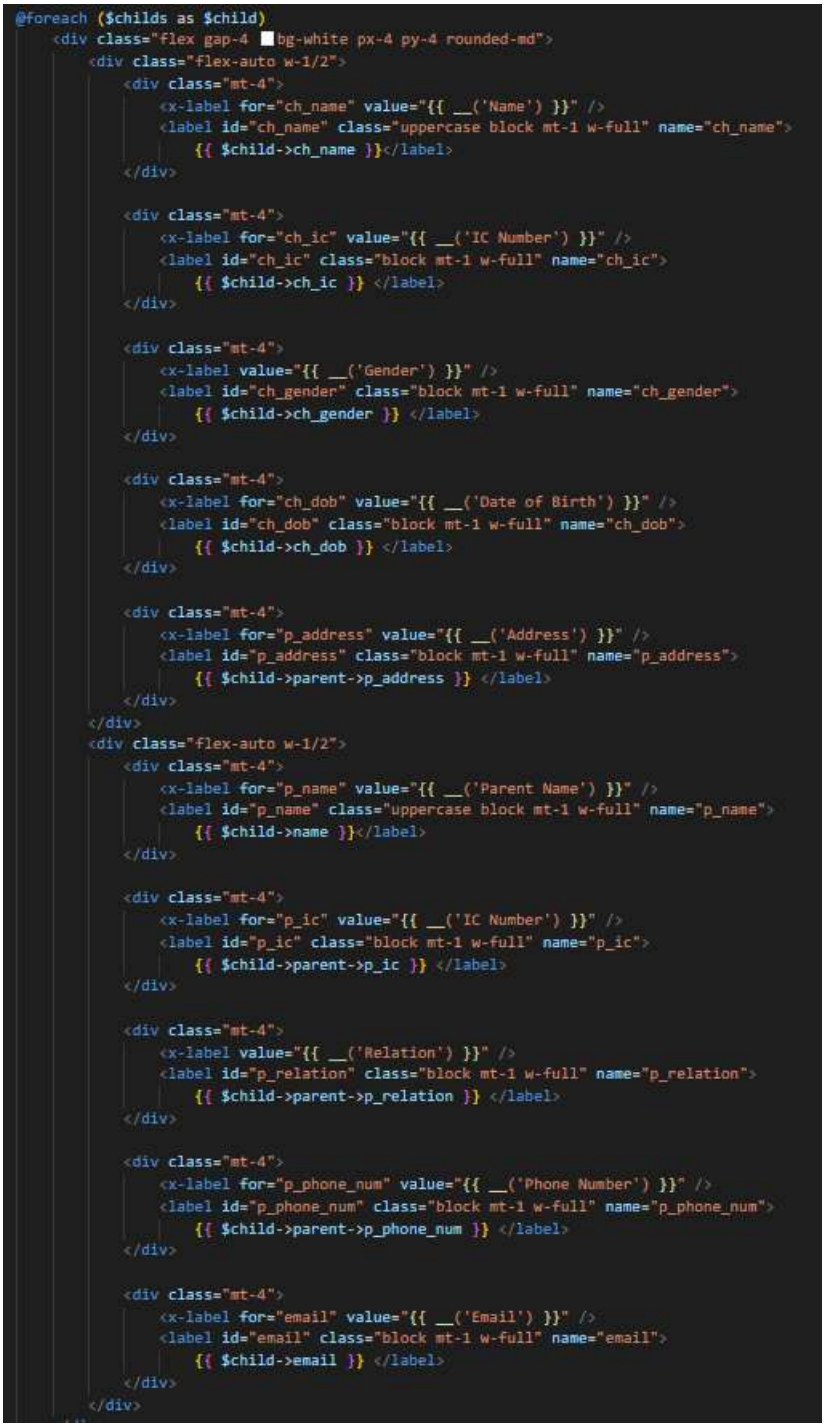


Figure 4.2.82: Child and Parent Info blade

4.2.4.4 Route

```php
Route::get('/admin/dashboard', [UserController::class,'index'])->name('admin.dashboard');
Route::post('/manage-user-system/admin/add-user', [UserController::class,'create'])->name('user.add');
Route::get('/manage-user-system/admin/view-user/{id}', [UserController::class,'show'])->name('admin.user.view');
Route::get('/manage-user-system/admin/edit-user/{id}', [UserController::class,'edit'])->name('admin.user.edit');
Route::post('/manage-user-system/admin/edit-user/update/{id}', [UserController::class,'update'])->name('user.update');
Route::get('destroy-user/{id}', [UserController::class,'destroy'])->name('destroy-user');

// go to add user page
Route::get('/manage-user-system/admin/add-user', function () {
    return view('manage-user-system/add-user');
})->name('admin.user.add');

// Manage m1

Route::get('/manage-m1/parent/m1-question', [M1Controller::class, 'index'])->name('parent.m1');
Route::post('/manage-m1/parent/m1-question-store', [M1Controller::class, 'store'])->name('m1.store');
Route::post('/manage-m1/child-loe/{id}', [M1Controller::class, 'update'])->name('child.loe.update');
Route::get('/manage-m1/child-list', [M1Controller::class, 'childList'])->name('child.list');
Route::get('/manage-m1/m1-details/{id}', [M1Controller::class, 'm1Details'])->name('m1.details');

// Manage appointment

Route::get('/manage-appointment/list-appointment', [AppointmentController::class, 'index'])->name('appointment.list');
Route::post('/manage-appointment/parent/add-appointment/store', [AppointmentController::class, 'store'])->name('parent.appointment.store');
Route::get('/manage-appointment/parent/add-appointment', [AppointmentController::class, 'create'])->name('parent.appointment.add');
Route::get('destroy-appntmnt/{id}', [AppointmentController::class,'destroy'])->name('destroy-appntmnt');

Route::get('edit-appntmnt/{id}', [AppointmentController::class,'edit'])->name('edit-appntmnt');
Route::post('/manage-appointment/appointment/{id}', [AppointmentController::class, 'update'])->name('update.pic');

// Manage payment

Route::post('/manage-payment/store', [PaymentController::class, 'store'])->name('payment.store');
Route::get('/manage-payment', [PaymentController::class, 'index'])->name('payment.page');
Route::get('/manage-report/payment', [PaymentController::class, 'report'])->name('payment.report');

// Manage m3

Route::get('/manage-m3/parent/m3-question', [M3Controller::class, 'index'])->name('parent.m3');
Route::post('/manage-m3/parent/m3-question-store', [M3Controller::class, 'store'])->name('m3.store');
```

Figure 4.2.83: web.php

4.2.5 Database Implementation

	id	appntmnt_type	appntmnt_date	appntmnt_time	child_id	staff_id	created_at	updated_at
Edit Copy Delete	1	PERJUMPAAN PERTAMA (P1)	2023-06-30	08:30 AM	2	2	2023-06-02 14:17:12	2023-06-08 08:38:26
Edit Copy Delete	7	PERJUMPAAN PERTAMA (P1)	2023-06-09	10:00 AM	7	NULL	2023-06-07 07:29:02	2023-06-07 07:29:02
Edit Copy Delete	10	PERJUMPAAN PERTAMA (P1)	2023-06-12	09:00 AM	9	1	2023-06-09 08:00:48	2023-06-09 08:03:09

Figure 4.2.84: Appointments table

	id	ch_name	ch_gender	ch_dob	ch_ic	loe_programme	parent_id	created_at	updated_at
Edit Copy Delete	2	KHAISARA BINTI ABDULLAH	FEMALE	2022-09-13	220913100286	INCLUSIVE	6	2023-04-05 20:11:44	2023-06-08 09:01:32
Edit Copy Delete	7	ARIFFIN BIN ABDULLAH	MALE	2023-01-01	230101101243	NULL	6	2023-06-07 01:43:40	2023-06-07 01:43:40
Edit Copy Delete	8	ADAM IDRIS BIN NUH SOLEH	MALE	2020-07-20	200720100263	NULL	7	2023-06-07 04:09:06	2023-06-07 04:09:06
Edit Copy Delete	9	HANAN BIN ABDULLAH	MALE	2021-07-20	210720100263	INTENSIVE	24	2023-06-09 07:50:29	2023-06-09 07:56:48

Figure 4.2.85: Childs table

	id	psQID	answer	child_id	created_at	updated_at
Edit Copy Delete	145	1	a	2	2023-06-03 00:15:25	2023-06-03 00:15:25
Edit Copy Delete	146	2	Tidak	2	2023-06-03 00:15:25	2023-06-03 00:15:25
Edit Copy Delete	147	3	b	2	2023-06-03 00:15:25	2023-06-03 00:15:25
Edit Copy Delete	148	4	c	2	2023-06-03 00:15:25	2023-06-03 00:15:25
Edit Copy Delete	149	5	Ya	2	2023-06-03 00:15:25	2023-06-03 00:15:25
Edit Copy Delete	150	6	Tidak	2	2023-06-03 00:15:25	2023-06-03 00:15:25
Edit Copy Delete	151	7	d	2	2023-06-03 00:15:25	2023-06-03 00:15:25
Edit Copy Delete	152	8	Ya	2	2023-06-03 00:15:25	2023-06-03 00:15:25
Edit Copy Delete	153	9	e	2	2023-06-03 00:15:25	2023-06-03 00:15:25
Edit Copy Delete	154	10	f	2	2023-06-03 00:15:25	2023-06-03 00:15:25
Edit Copy Delete	155	11	Social Media,Papan Iklan	2	2023-06-03 00:15:25	2023-06-03 00:15:25
Edit Copy Delete	156	1	a	7	2023-06-07 07:07:25	2023-06-07 07:07:25
Edit Copy Delete	157	2	Tidak	7	2023-06-07 07:07:25	2023-06-07 07:07:25
Edit Copy Delete	158	3	b	7	2023-06-07 07:07:25	2023-06-07 07:07:25
Edit Copy Delete	159	4	c	7	2023-06-07 07:07:25	2023-06-07 07:07:25
Edit Copy Delete	160	5	Ya	7	2023-06-07 07:07:25	2023-06-07 07:07:25
Edit Copy Delete	161	6	Ya	7	2023-06-07 07:07:25	2023-06-07 07:07:25
Edit Copy Delete	162	7	d	7	2023-06-07 07:07:25	2023-06-07 07:07:25
Edit Copy Delete	163	8	Tidak	7	2023-06-07 07:07:25	2023-06-07 07:07:25
Edit Copy Delete	164	9	e	7	2023-06-07 07:07:25	2023-06-07 07:07:25
Edit Copy Delete	165	10	f	7	2023-06-07 07:07:25	2023-06-07 07:07:25

Figure 4.2.86: M1s table

	id	m3QID	answer	child_id	created_at	updated_at
Edit Copy Delete	63	1	LOE Onboarding System.pdf	2	2023-06-04 17:57:16	2023-06-04 17:57:16
Edit Copy Delete	64	2	*NULL*	2	2023-06-04 17:57:16	2023-06-04 17:57:16
Edit Copy Delete	65	3	*NULL*	2	2023-06-04 17:57:16	2023-06-04 17:57:16
Edit Copy Delete	66	4	a	2	2023-06-04 17:57:16	2023-06-04 17:57:16
Edit Copy Delete	67	5	b	2	2023-06-04 17:57:16	2023-06-04 17:57:16
Edit Copy Delete	68	6	c	2	2023-06-04 17:57:16	2023-06-04 17:57:16
Edit Copy Delete	69	7	d	2	2023-06-04 17:57:16	2023-06-04 17:57:16
Edit Copy Delete	70	8	Tidak	2	2023-06-04 17:57:16	2023-06-04 17:57:16
Edit Copy Delete	71	9	Normal	2	2023-06-04 17:57:16	2023-06-04 17:57:16
Edit Copy Delete	72	10	Tidak	2	2023-06-04 17:57:16	2023-06-04 17:57:16
Edit Copy Delete	73	11	e	2	2023-06-04 17:57:16	2023-06-04 17:57:16
Edit Copy Delete	74	12	f	2	2023-06-04 17:57:16	2023-06-04 17:57:16
Edit Copy Delete	75	13	g	2	2023-06-04 17:57:16	2023-06-04 17:57:16
Edit Copy Delete	76	14	h	2	2023-06-04 17:57:16	2023-06-04 17:57:16
Edit Copy Delete	77	15	Ibu bapa	2	2023-06-04 17:57:16	2023-06-04 17:57:16
Edit Copy Delete	78	16	i	2	2023-06-04 17:57:16	2023-06-04 17:57:16
Edit Copy Delete	79	17	j	2	2023-06-04 17:57:16	2023-06-04 17:57:16
Edit Copy Delete	80	18	k	2	2023-06-04 17:57:16	2023-06-04 17:57:16
Edit Copy Delete	81	19	l	2	2023-06-04 17:57:16	2023-06-04 17:57:16
Edit Copy Delete	82	20	m	2	2023-06-04 17:57:16	2023-06-04 17:57:16
Edit Copy Delete	83	21	n	2	2023-06-04 17:57:16	2023-06-04 17:57:16

Console

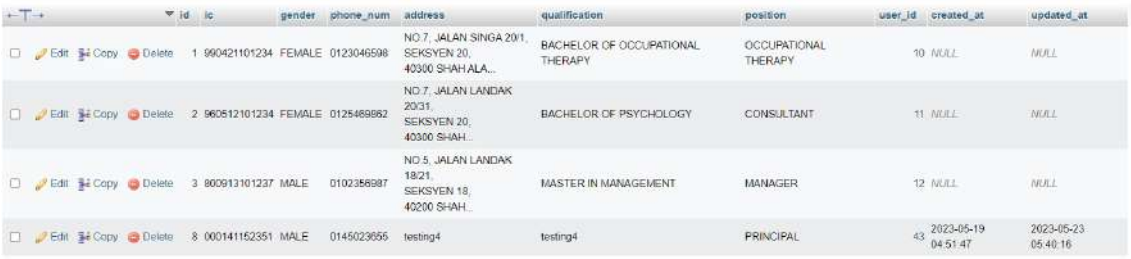
Figure 4.2.87: M3s table

	id	p_ic	p_phone_num	p_relation	p_address	user_id	created_at	updated_at
Edit Copy Delete	6	010913100282	0123456789	MOTHER	NO. 5, JALAN SAWI 24/3, SEKSYEN 24, 40400 SHAH ALA...	13	2023-04-05 20:11:44	2023-04-05 20:11:44
Edit Copy Delete	7	010913100283	0123456790	GUARDIAN	NO. 5, JALAN SAWI 24/3, SEKSYEN 24, 40400 SHAH A...	18	2023-04-05 20:11:44	2023-05-23 05:49:17
Edit Copy Delete	24	000720100262	0123038788	MOTHER	NO 48, JALAN RINTING, 18/29, SEKSYEN 18, 40200 SHA...	47	2023-06-09 07:44:50	2023-06-09 07:49:58

Figure 4.2.88: Parents table

	id	payment_type	total_payment	payment_remarks	payment_status	bank_name	fee_type	child_id	created_at	updated_at
Edit Copy Delete	1	ONLINE BANK-IN	375.00	-	PAID	MAYBANK	PROGRAM PEMERHATIAN SEBULAN (PPS)	2	2023-06-08 17:46:40	2023-06-08 17:46:40
Edit Copy Delete	5	ONLINE BANK-IN	375.00	none	PAID	BIMB	PROGRAM PEMERHATIAN SEBULAN (PPS)	9	2023-06-09 08:05:28	2023-06-09 08:05:28

Figure 4.2.89: Payments table



	id	ic	gender	phone_num	address	qualification	position	user_id	created_at	updated_at
Edit Copy Delete	1	990421101234	FEMALE	0123046598	NO.7, JALAN SINGA 20/1, SEKSYEN 20, 40300 SHAH ALA...	BACHELOR OF OCCUPATIONAL THERAPY	OCCUPATIONAL THERAPY	10	NULL	NULL
Edit Copy Delete	2	960512101234	FEMALE	0125469862	NO.7, JALAN LANDAK 20/31, SEKSYEN 20, 40300 SHAH...	BACHELOR OF PSYCHOLOGY	CONSULTANT	11	NULL	NULL
Edit Copy Delete	3	800913101237	MALE	0102356987	NO.5, JALAN LANDAK 18/21, SEKSYEN 18, 40200 SHAH...	MASTER IN MANAGEMENT	MANAGER	12	NULL	NULL
Edit Copy Delete	8	000141152351	MALE	0145023655	testing4	testing4	PRINCIPAL	43	2023-05-19 04:51:47	2023-05-23 05:40:16

Figure 4.2.90: Staffs table

	id	name	email	role	email_verified_at	password	two_factor_secret	two_factor_recovery_codes
Edit Copy Delete	9	ADMIN	admin@gmail.com	ADMIN	NULL	$2y$10$Brl5IDzdUueTkh6wsaHPouD2irFYCYLj8MDx1czjF1W...	NULL	NULL
Edit Copy Delete	10	AINNA ZINNIRAH BINTI MARDZUKI	ainnazinnirah1@gmail.com	STAFF	NULL	$2y$10$PYKjseWBSbiAxArD52370.HwbolyuRhAbt7nGMzjUF...	NULL	NULL
Edit Copy Delete	11	WAN SUMAYYAH BINTI WAN AHMAD RAHIMAN	smyyhr@gmail.com	STAFF	NULL	$2y$10$evZZdMQoik8ut6RJw9wXEOlMwJ7I5RhIPVuecNd4rYUZ...	NULL	NULL
Edit Copy Delete	12	MUHAMMAD IZZUDDIN BIN MOHAMMAD AZMI	izzuddinazmi6@gmail.com	STAFF	NULL	$2y$10$USgvKvmcl0x3gvDABRikf.fGQEZCyYmiJDNxU1gKS6xG...	NULL	NULL
Edit Copy Delete	13	NURUL IZZAH BINTI MOHAMMAD AZMI	izzah.nurul63.ni@gmail.com	PARENT	2023-05-23 15:08:57	$2y$10$Shbh1f0qRZQFBWl8/TrRVefKuSJzsrcaCSQ1RYay7z8...	NULL	NULL
Edit Copy Delete	18	NUR ATIQAH BINTI KAMAL	nuratiqah1306@gmail.com	PARENT	2023-05-01 03:33:17	$2y$10$1zo13uunchJ49e8jULmGp.j84pZSivCP6ElnsydZqzC...	NULL	NULL
Edit Copy Delete	43	ABU AHMAD BIN ALI	abu@gmail.com	STAFF	2023-05-20 15:52:45	$2y$10$Xrma/BrwqyHY8sM/MadY6b.bAfChds9OWy9/0tWMZcTHe...	NULL	NULL
Edit Copy Delete	47	NURIN IMAN BINTI AZMI	nuriniman93@gmail.com	PARENT	2023-06-09 07:45:11	$2y$10$rhDTlEcc2QwSC4kvaYWzYuxqetvOjTEZlkk9DZlA3FN...	NULL	NULL

Figure 4.2.91: Users table

4.3 Results / Findings

In this section, the results of the testing, as well as the comments and suggestions from users and other stakeholders, are presented. User Acceptance Testing (UAT) is used to determine whether the application's features are functioning as intended. UAT aims to ensure that the software or system satisfies the requirements of end users, is free of critical defects, and provides a satisfactory user experience. It reduces the likelihood of deployment problems, user dissatisfaction, and costly revisions after releasing the software. All system modules are subjected to testing for the benefit of prospective clients. Regarding the system development project, testing and validation were conducted via online meetings through Google Meet with the client, MZR Global Sdn Bhd on 9th June 2023, Friday, from 3 PM to 5 PM. The assessment is performed in Appendix B.

4.4 Discussions

This section analyses the results and discusses the system's limitations, areas for future development, and overall project success. Based on the UAT, some improvement needs to be made, such as implementing email notifications to ensure the user is always notified of any changes or updated info. Also, minor changes must be made to fulfil the client's requirements. Due to a lack of knowledge of the Laravel framework and constraints, two modules could not finish on time during the first testing phase. Overall, the system is working well, and the stakeholders are satisfied with the existing modules that have been completed.

CHAPTER 5

CONCLUSION

5.1 Introduction

The establishment of Little One Eduworld (LOE) and the need for a Special Needs Children Onboarding Management System emphasizes the significance of providing adequate education and rehabilitation programs for children with special needs. The manual onboarding process at LOE has presented numerous difficulties, such as time-consuming tasks, misunderstandings, miscommunication, and the loss or destruction of vital documents. These issues can considerably impact LOE's management quality and reputation.

Using the Laravel framework, this project intends to develop a Special Needs Children Onboarding Management System to resolve these issues. The system seeks to increase productivity by automating processes, managing data securely, facilitating document retrieval, and rapidly generating reports. By implementing this system, LOE staff and parents will be able to effectively manage the onboarding process for new students, thereby improving the overall experience for the children and enhancing their development monitoring.

The project adheres to an agile scrum framework, enabling effective collaboration among team members and adaptability to shifting project goals. This methodology facilitates rapid development and permits rapid updates to accommodate stakeholder needs.

The implementation of the system presents ten modules that cover various aspects of onboarding management. The system uses the MySQL database and the Laravel web application framework. Results and discussions include user acceptance testing (UAT)

and stakeholder feedback, emphasizing the system's functionality, limitations, and future development opportunities. Based on the results of the UAT, the system exhibits effective performance. However, specific enhancements are required to meet client requirements. Despite constraints and limited knowledge of the Laravel framework, stakeholders are pleased with the completed modules, demonstrating the successful completion of the project.

Overall, the Special Needs Children Onboarding Management System provides a valuable solution to the problems encountered by LOE, enhancing the efficiency of onboarding processes and the overall management of special needs children. With continued development and the incorporation of stakeholder input, this system can potentially expand and enhance.

5.2 Objective Revisited

Table 5.2.1: Objectives and Results

No.	Objective	Result
1	To study the existing system related to Special Needs Children Onboarding Management System.	The study of existing systems, including the SentricHR Employee Onboarding Software, Zoho Employee Onboarding System, and ClearCompany Employee Onboarding Software, provided valuable insights into onboarding management practices and served as a reference for developing the Special Needs Children Onboarding Management System. Also, the requirements are clearly defined for the Special Needs Children Onboarding system with client participation.
2	To develop a Special Needs Children Onboarding Management	The development of the Special Needs Children Onboarding Management System was successfully carried out

	System using the Laravel framework.	using the Laravel web application framework. The system comprises ten modules, including registration, login, manage user, manage profile, manage appointment, manage payment, and report. The system was designed and developed as a simplified, visually appealing, and user-friendly web-based system.
3	To evaluate the functionality of the Special Needs Children Onboarding Management System.	The functionality of the Special Needs Children Onboarding Management System was evaluated through user acceptance testing (UAT) by MZR Global Sdn Bhd. The system was tested for its features, functionality, and performance to validate and verify the integrity of functionality implementation, the system's efficacy, and usability. Feedback and suggestions from users and stakeholders were collected and analyzed to assess the system's effectiveness and identify areas for improvement.

5.3 Limitations and Constrains

Throughout the system's development, certain limitations and constraints are encountered during the project's completion.

- **Time constraints**

  Insufficient time to implement a system with more appealing interfaces and more precise functions within the project's timeline. Developing a complete and robust system within a specified timeframe can be difficult. The project had strict deadlines, which placed pressure on the development process.

- **Lack of experience and knowledge of the Laravel framework**

  My limited prior exposure to the Laravel framework made learning and adapting to its functionalities challenging. This limitation may have impacted the development schedule and the ability to utilize the framework's features completely. The system is developed using the Laravel framework, requiring more knowledge, experience, and familiarity with the architecture. Knowledge and familiarity gaps with the Laravel framework create difficulties in comprehending code structure and managing errors within the project's deadline.

- **Technical constraints**

  During the development process, specific technical restrictions or limitations were encountered. These include browser compatibility issues, database constraints, and regulations imposed by the selected development tools. To overcome these technical limitations, additional effort and troubleshooting were required.

- **Participation of stakeholders and communication**

  Communication and participation of stakeholders, including the management team and end consumers, are essential for project success. Possible obstacles to the development process and the ability to fully meet the stakeholders' needs and requirements were a lack of stakeholder engagement or communication gaps. The

involvement and engagement of clients affect the system's development, as adjustments and modifications to the requested functions and requirements impact the project schedule.

5.4 Recommendations Future Works

The Special Needs Onboarding Management System is still deficient in functionality, validation, user interface design, and additional enhancements for future works.

- Additional features or improvements can be added to the system based on comments and recommendations from users and stakeholders. This may include features such as messaging notifications/email notifications to keep users informed of changes or reminders. Further usability testing and user feedback collection can help identify areas where the system's user interface and user experience can be improved. Enhancing the system's intuitiveness, navigation, and overall efficacy will facilitate a more streamlined user onboarding procedure.

- Explore integrating the Special Needs Children Onboarding Management System with other relevant platforms or systems. For instance, integration with the Child Development Screening System and Payment System can expedite data exchange and improve overall management and collaboration processes.

- Consider creating a mobile application variant of the system to increase user accessibility and convenience. Using their mobile devices, parents and staff can access the system, receive notifications, and interact with the system via a mobile application on the go.

- Establish a procedure for ongoing system maintenance and updates to resolve potential issues or bugs. Review and update the system regularly to ensure conformity with evolving requirements, technological advances, and emergent best practices in onboarding management.

REFERENCES

Preston, S.H. dan Elo, I.T. (1996). Survival after age 80: letter to the editor. New England Journal of Medicine, 334 (8), 537.

mt961013 (mt961013) wrote, 2018-11-29 20:03:00 mt961013 mt961013 2018-11-29 20:03:00. (2018, November 29). *Khutbah Jumaat: "Kanak-Kanak Harapan Negara."* LiveJournal. https://mt961013.livejournal.com/1401805.html

Kanak-kanak aset penting bangunkan negara. (2018, November 17). Utusan Borneo Online. https://www.utusanborneo.com.my/2018/11/17/kanak-kanak-aset-penting-bangunkan-negara

Alban, S. (2022, February 22). *Creating a Smooth Student Onboarding Process in the Middle of a Pandemic*. UserGuiding. https://userguiding.com/blog/student-onboarding-process/

Kanak-kanak Kurang Upaya di Malaysia Kajian mengenai pengetahuan, sikap dan amalan. (2018). *UNICEF*. https://www.unicef.org/malaysia/media/951/file/KAP%20booklet%20(BM).pdf

Sadlier, S. (2019, January 25). *Bad handwriting causing a big problem in US workplaces*. New York Post. https://nypost.com/2019/01/25/bad-handwriting-causing-a-big-problem-in-us-workplaces/

Atlassian, & Drumond. (2018). *What is Scrum?* Atlassian. https://www.atlassian.com/agile/scrum

Peek, S. (2022, June 29). *What Is Agile Scrum Methodology?* Business News Daily. https://www.businessnewsdaily.com/4987-what-is-agile-scrum-methodology.html

Singh, M. (2022, June 3). *What Is Scrum Methodology? & Scrum Project Management*. Digite. https://www.digite.com/agile/scrum-methodology/

SentricHR. (2022, March 22). *Simplified Employee Onboarding Software for Businesses*. All-In-One HR and Payroll Software - SentricHR. https://sentrichr.com/software/onboarding/

SentricHR *Reviews*. (2022). GetApp. https://www.getapp.com/hr-employee-management-software/a/sentricworkforce/reviews/

Matchr. (2022, July 1). *SentricHR Software - Free Demo and Reviews*. https://matchr.com/hris-software/software-list/sentrichr/

Zoho People Reviews. (2022). GetApp. https://www.getapp.com/hr-employee-management-software/a/zoho-people/reviews/

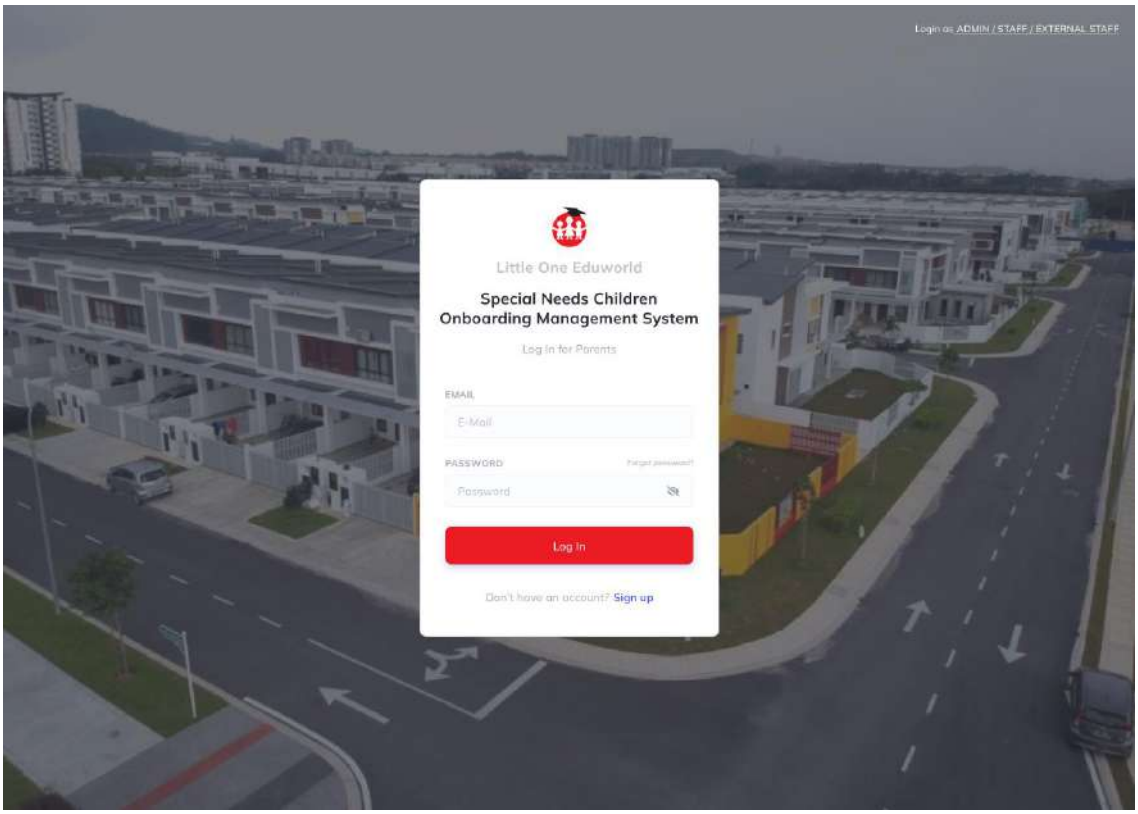
Free employee onboarding software | Customisable HR onboarding. (n.d.). Zoho. https://www.zoho.com/creator/apps/employee-onboarding-software.html

ClearCompany. (n.d.). *Leading ATS Software and Talent Management System | ClearCompany*. https://www.clearcompany.com/

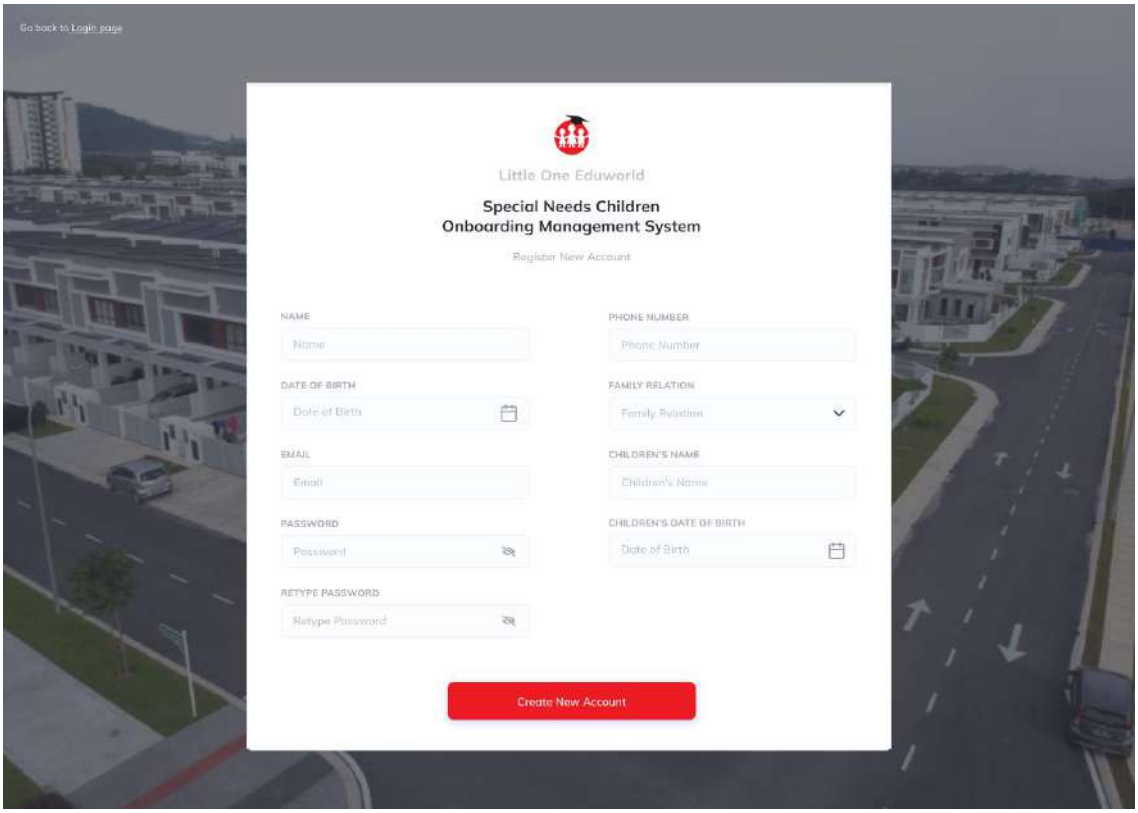
Doctor, T. (2021, July 29). *21 Best Employee Onboarding Software Tools for Remote Workers*. Time Doctor Blog. https://www.timedoctor.com/blog/employee-onboarding-software/

ClearCompany Reviews. (2021). GetApp. https://www.getapp.com/hr-employee-management-software/a/clearcompany-hrm/reviews/

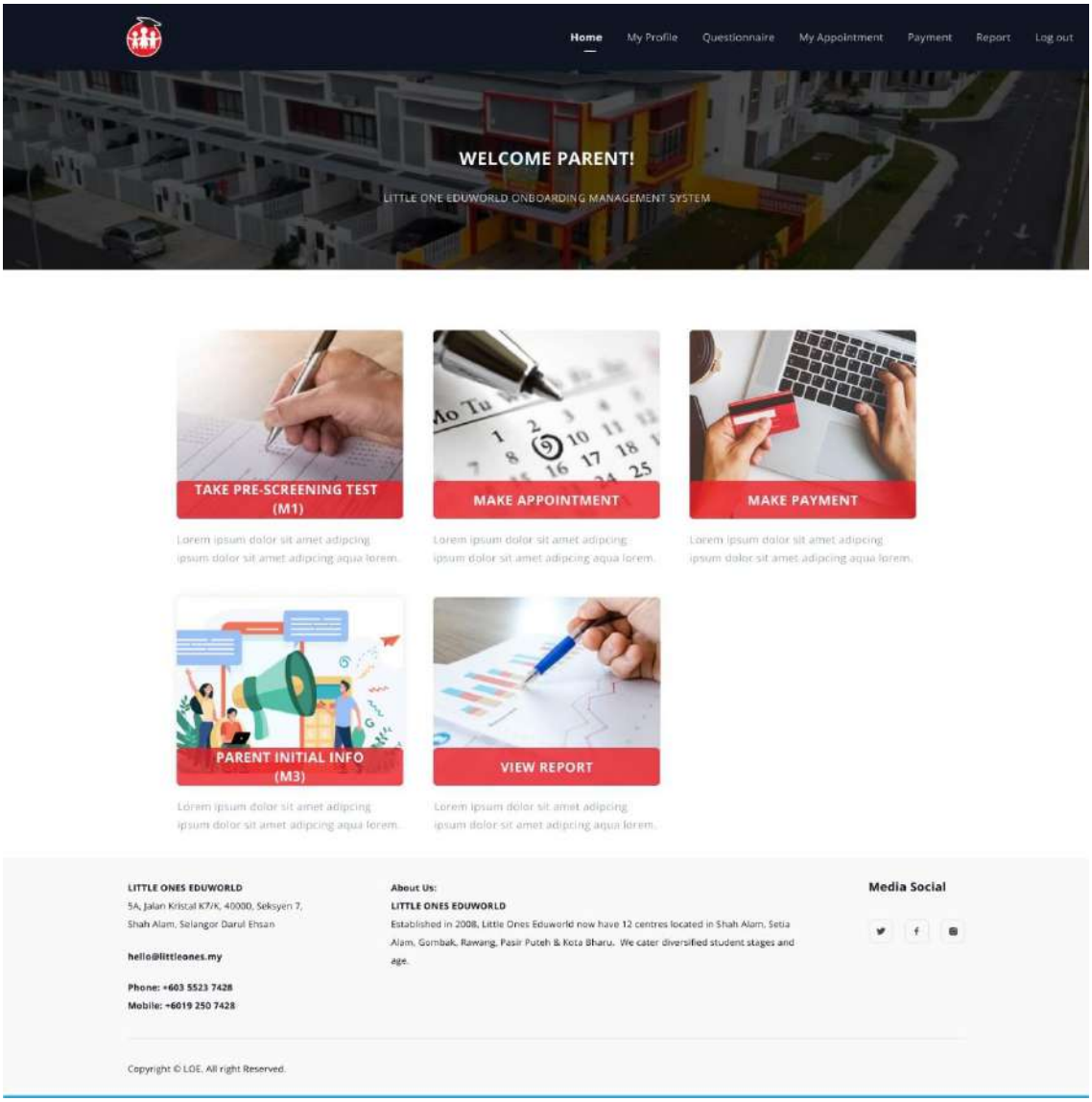
APPENDIX A



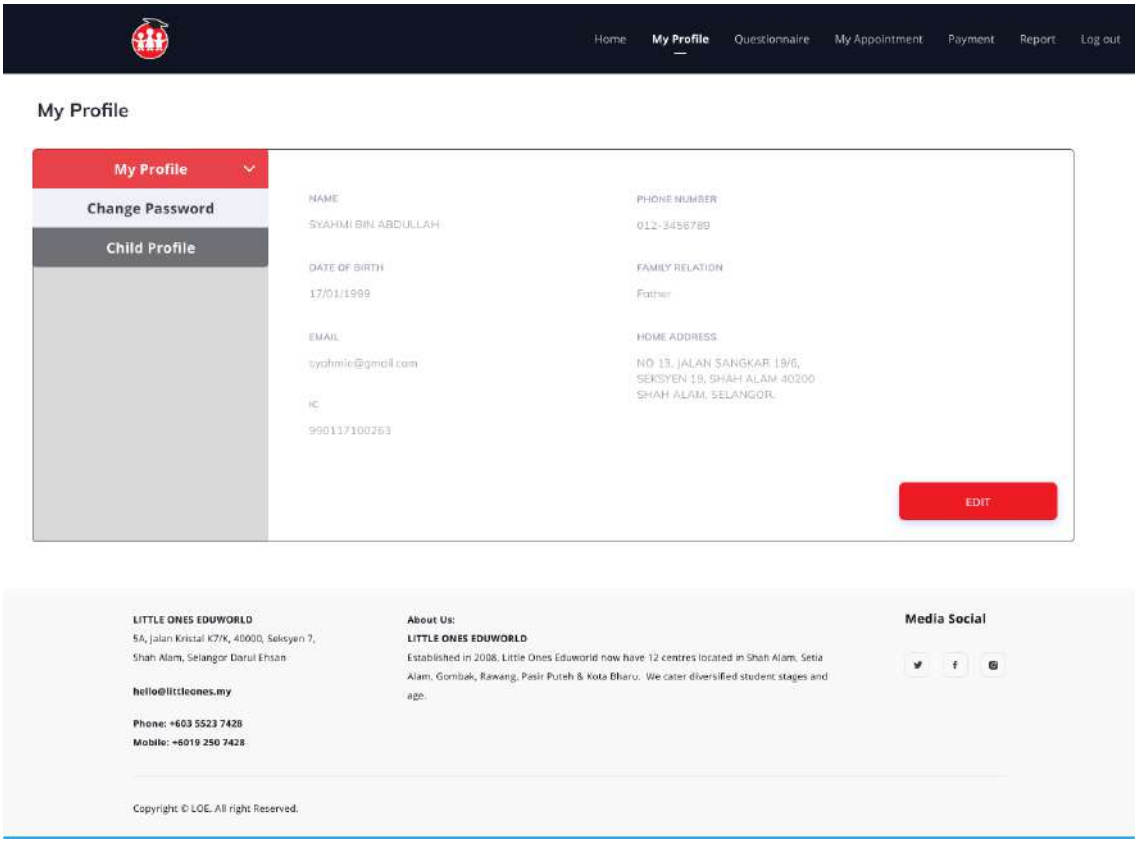
Appendix A. 1: Login page Parent



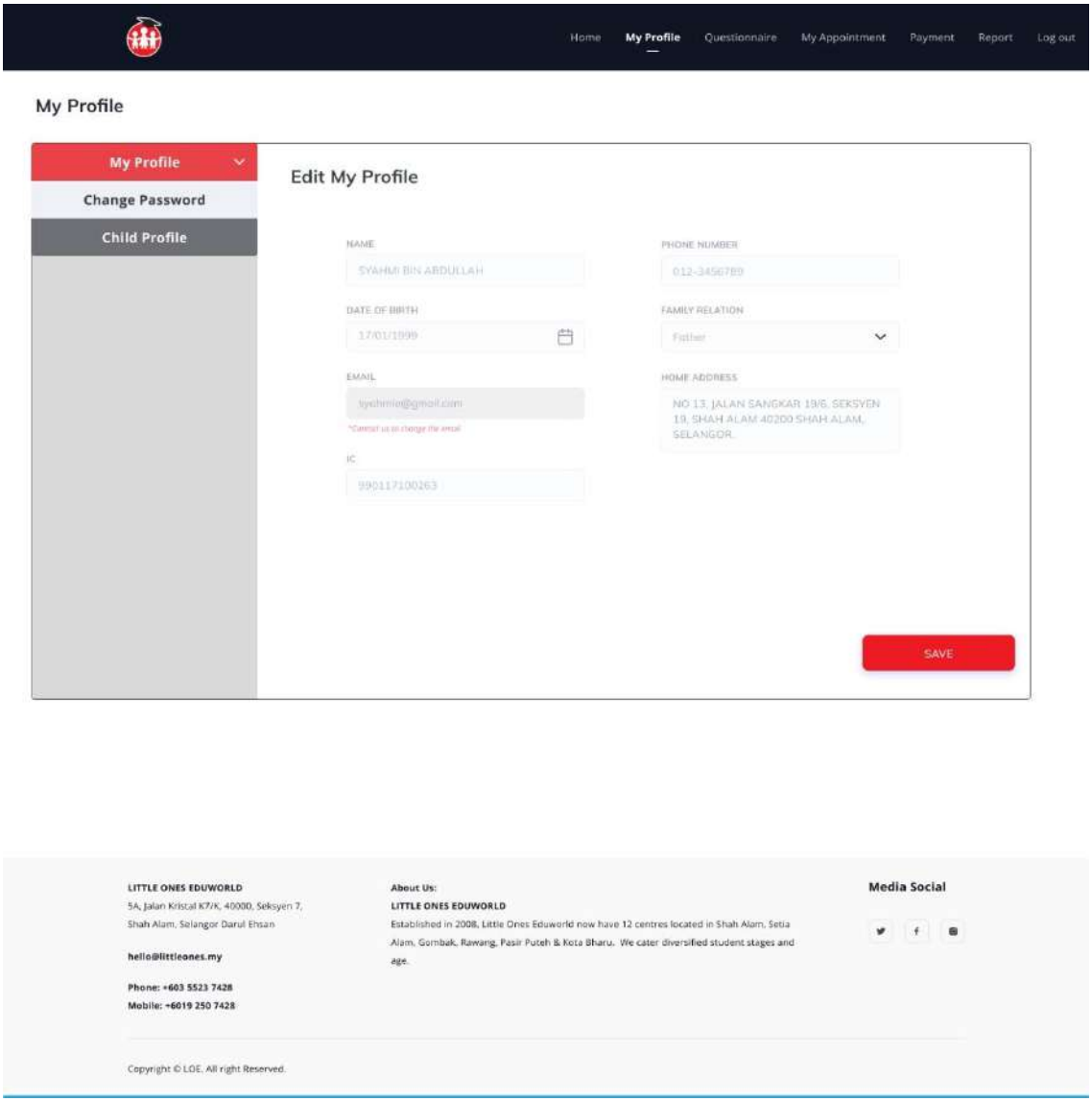
Appendix A. 2: Register page Parent



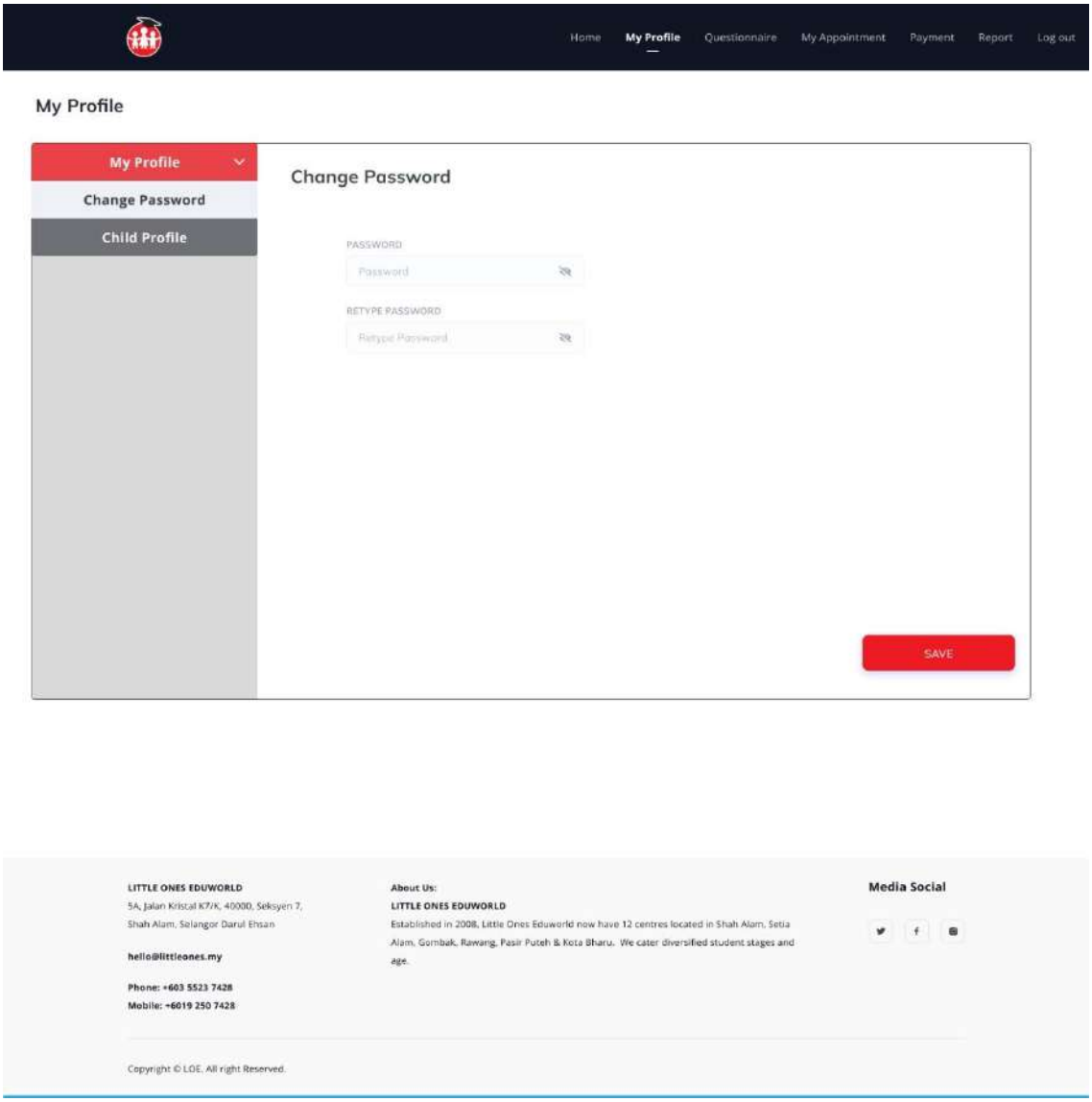
Appendix A. 3: Parent dashboard page



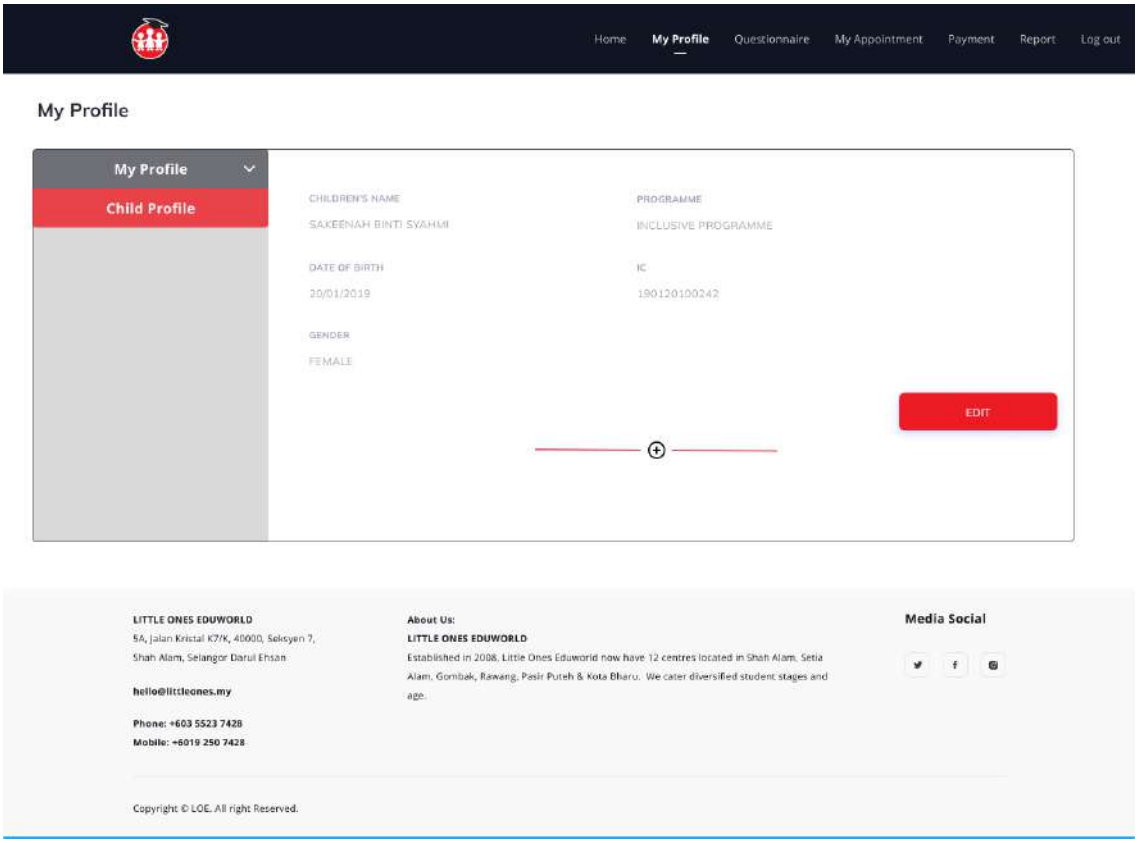
Appendix A. 4: My Profile page



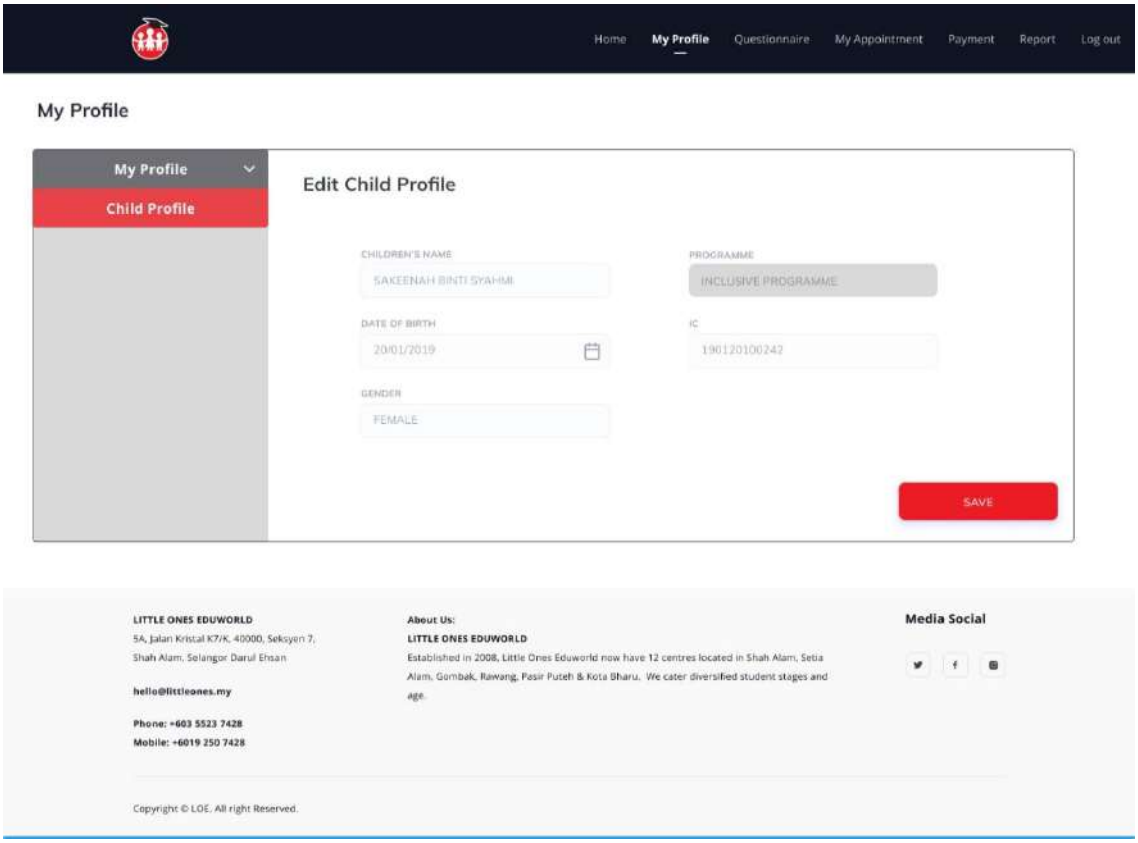
Appendix A. 5: Edit My Profile page



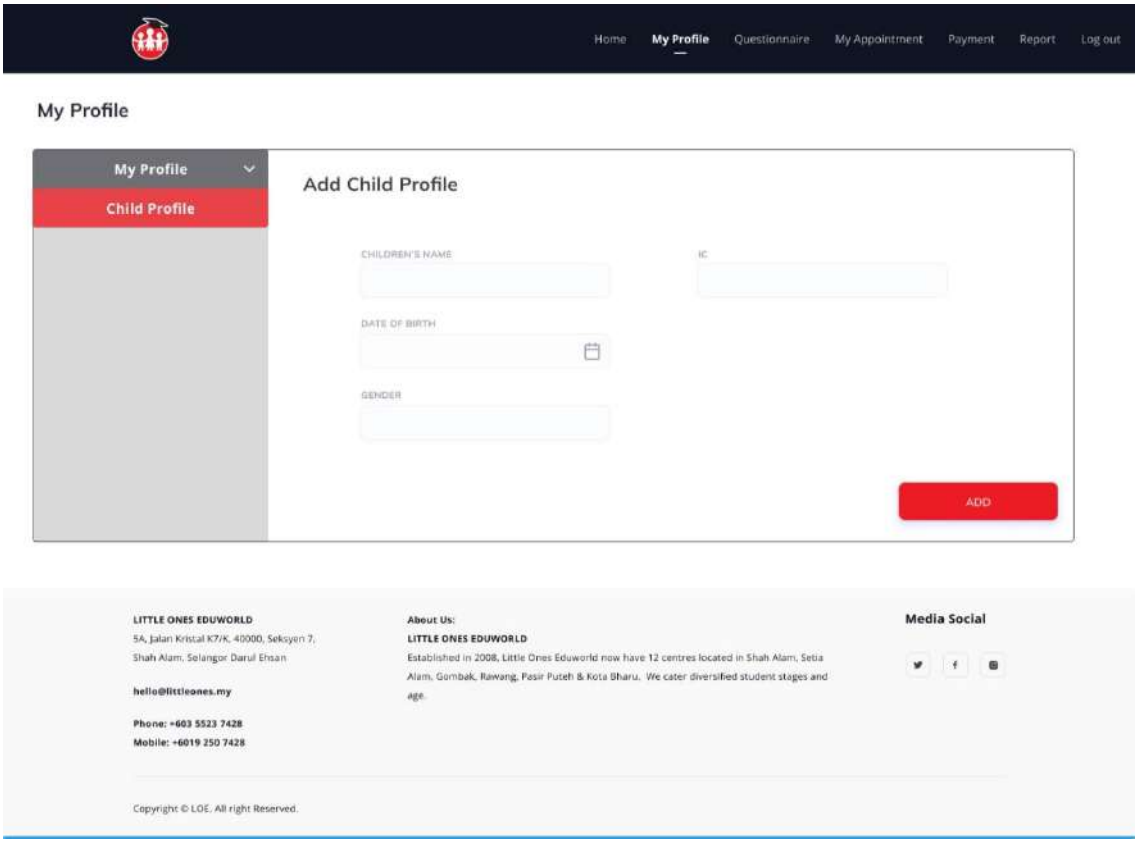
Appendix A. 6: Change password page



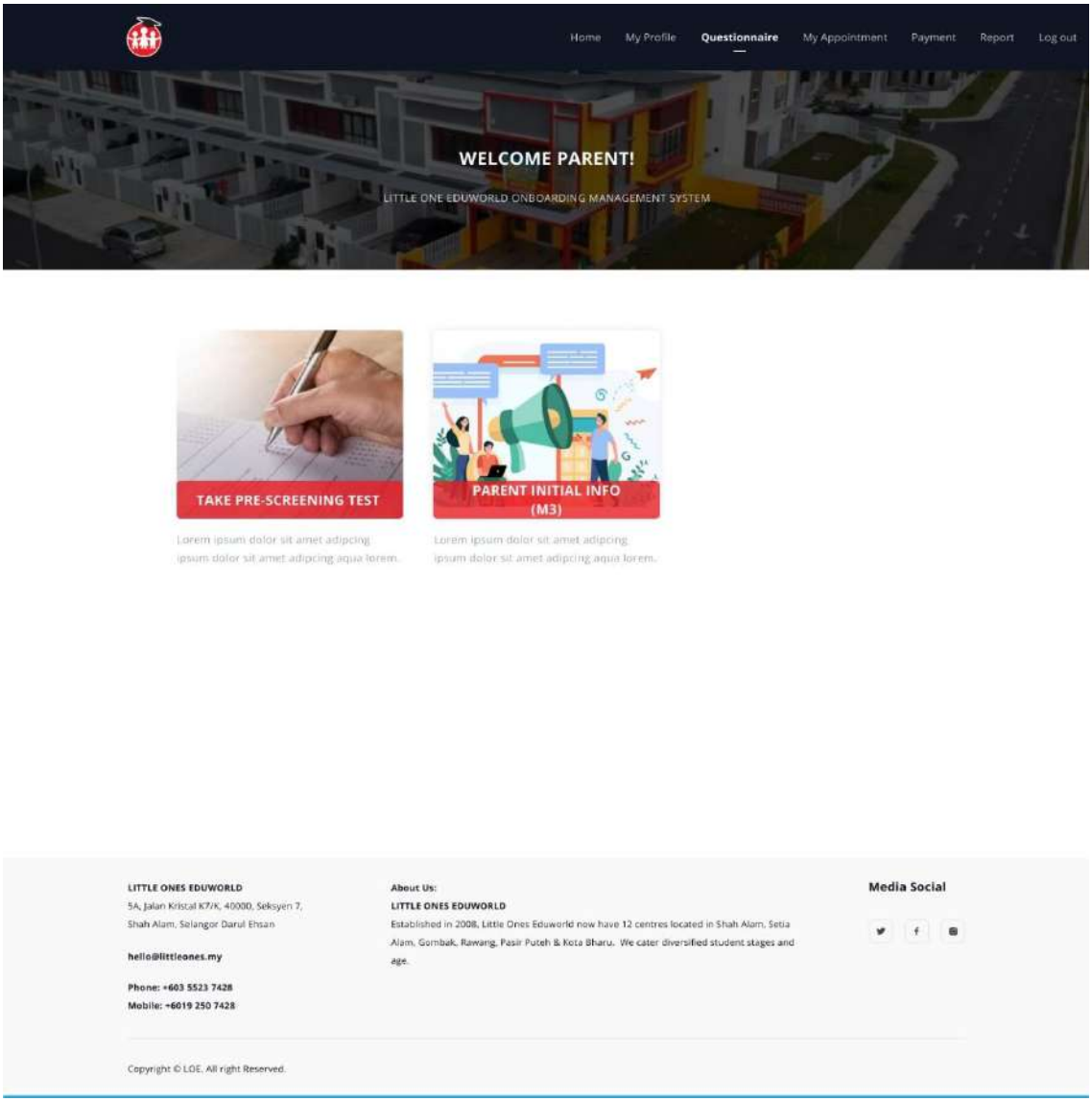
Appendix A. 7:Child Profile page



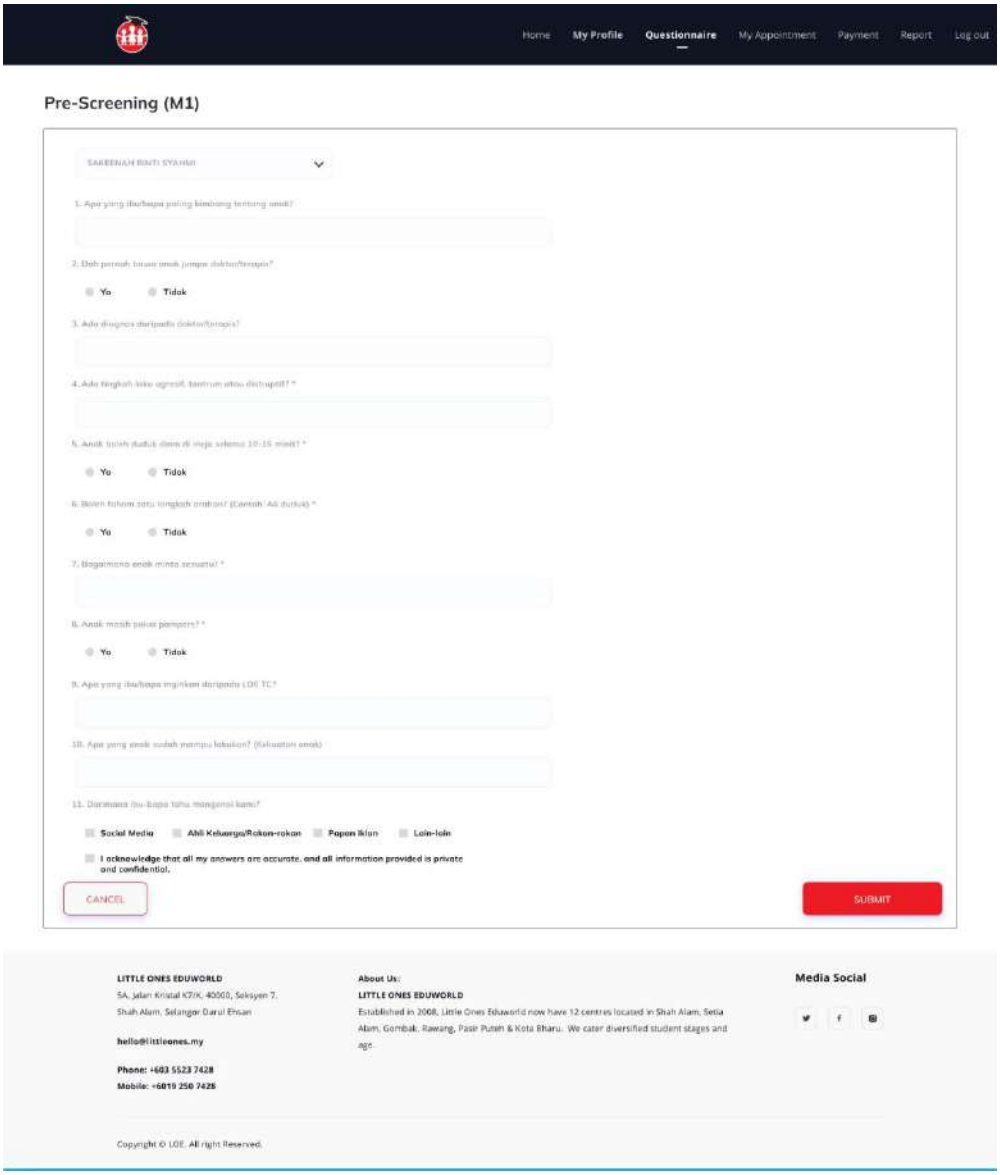
Appendix A. 8: Edit Child Profile page



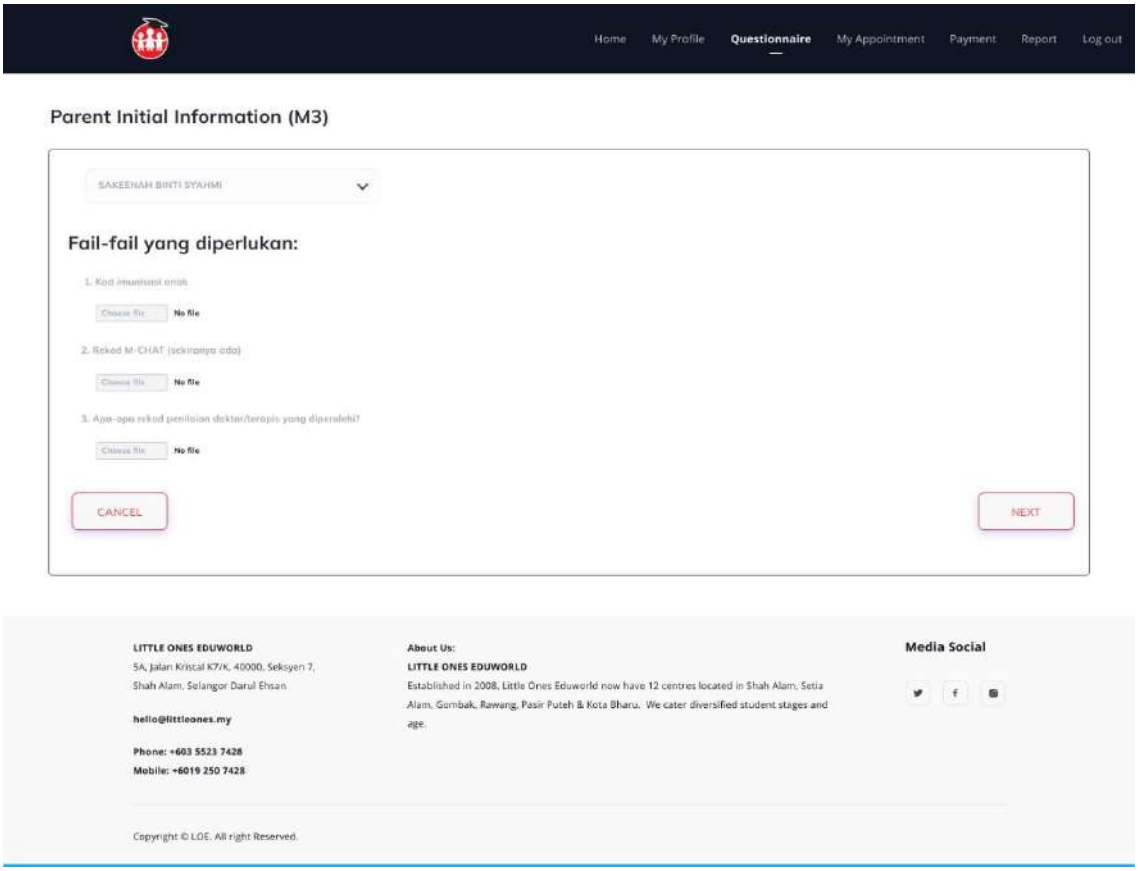
Appendix A. 9: Add Child Profile page



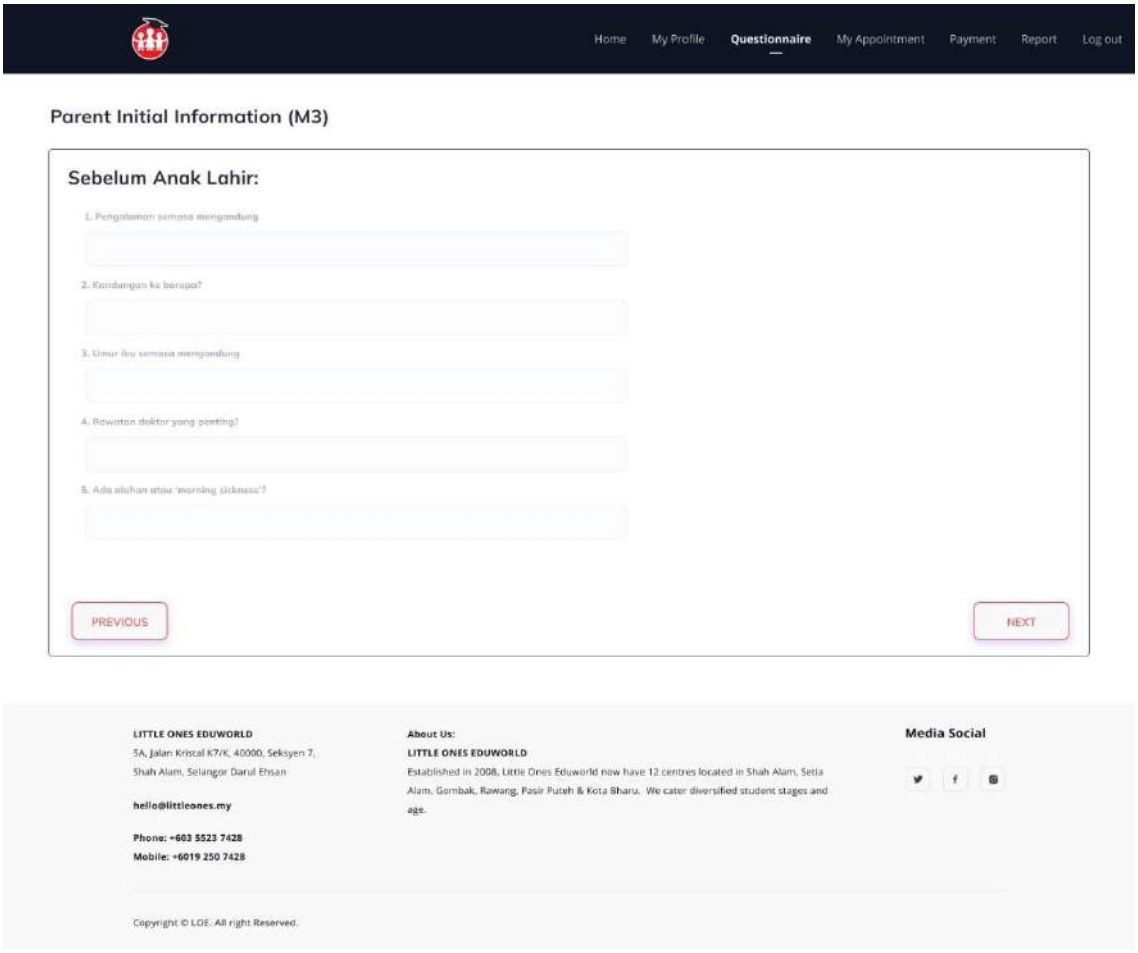
Appendix A. 10: Questionnaire dashboard page



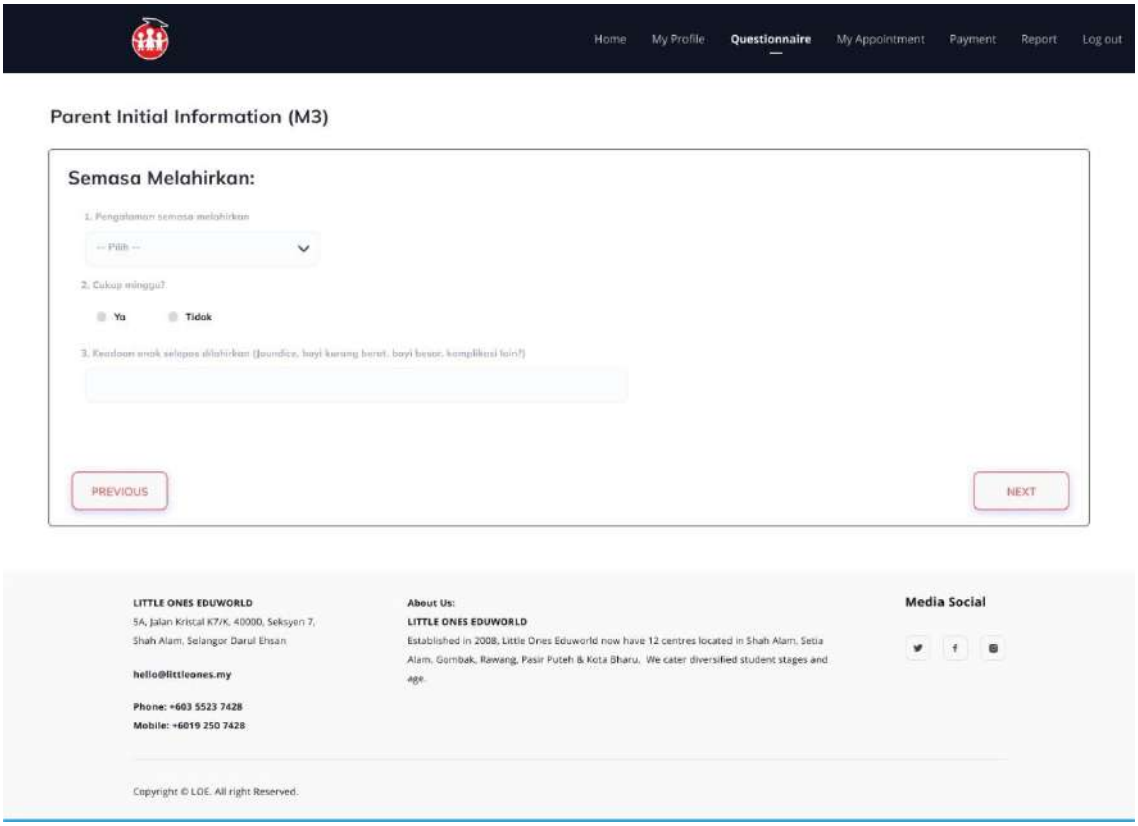
Appendix A. 11: Pre-screening page



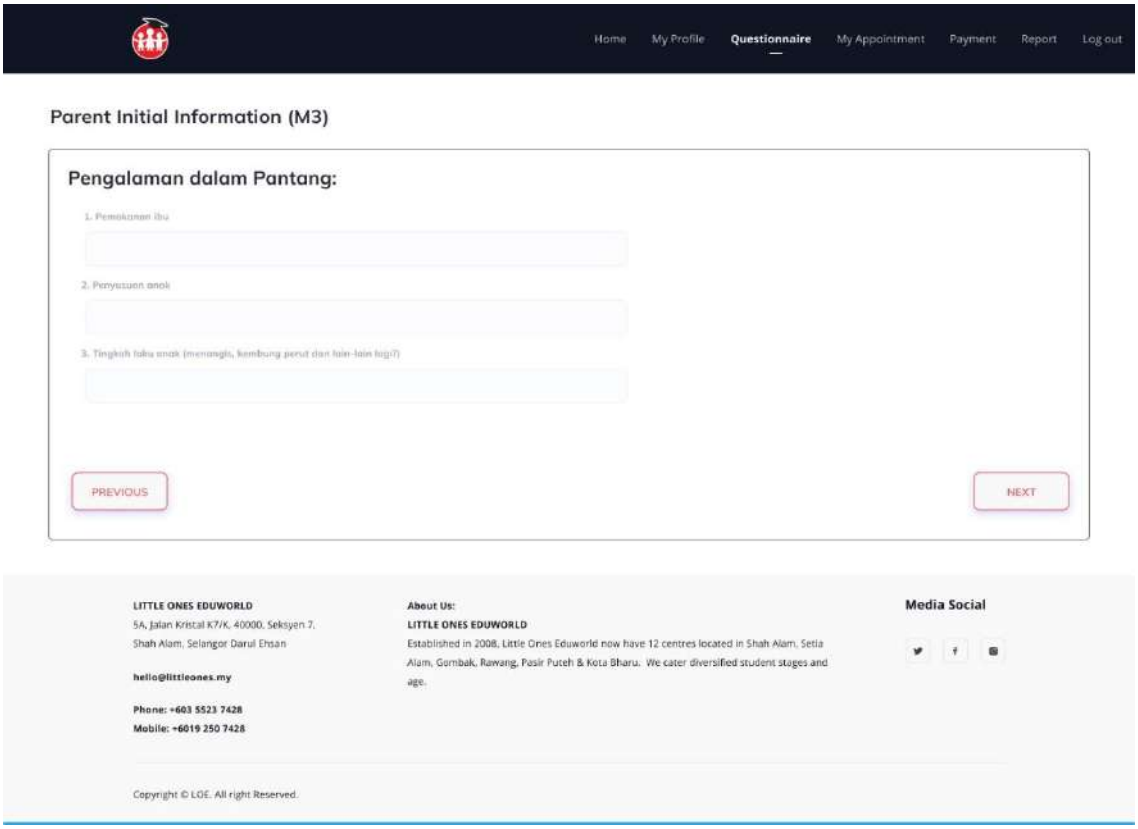
Appendix A. 12: Parent Initial Info Page 1



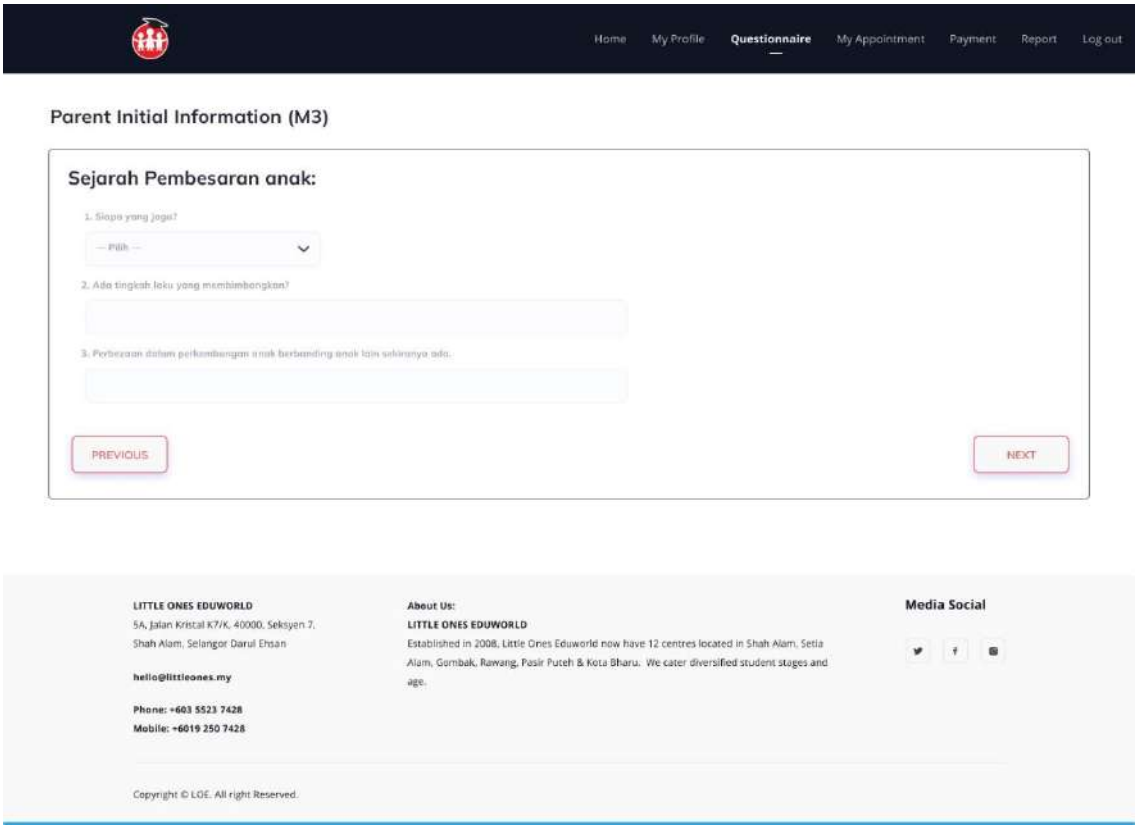
Appendix A. 13: Parent Initial Info Page 2



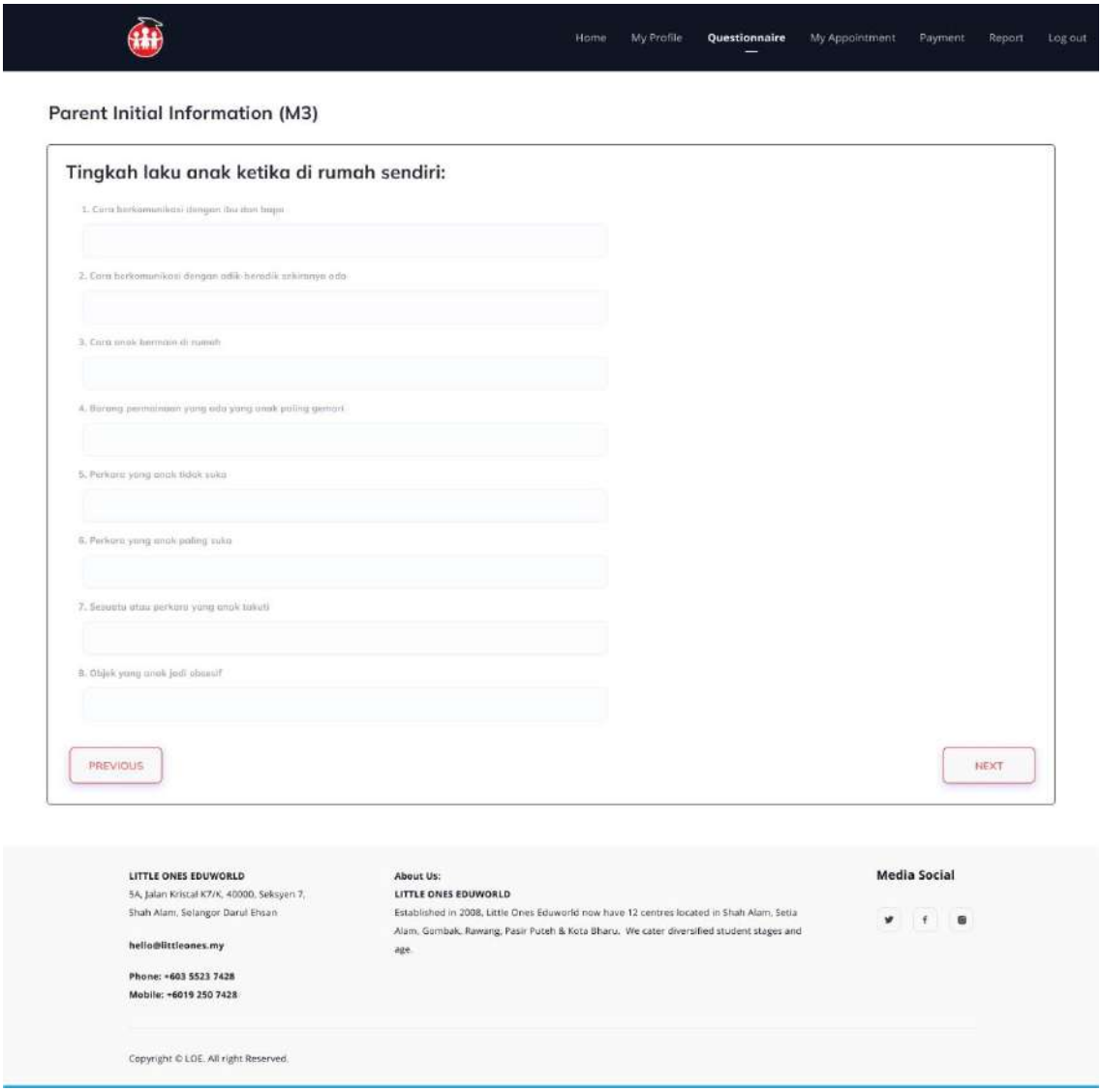
Appendix A. 14: Parent Initial Info Page 3



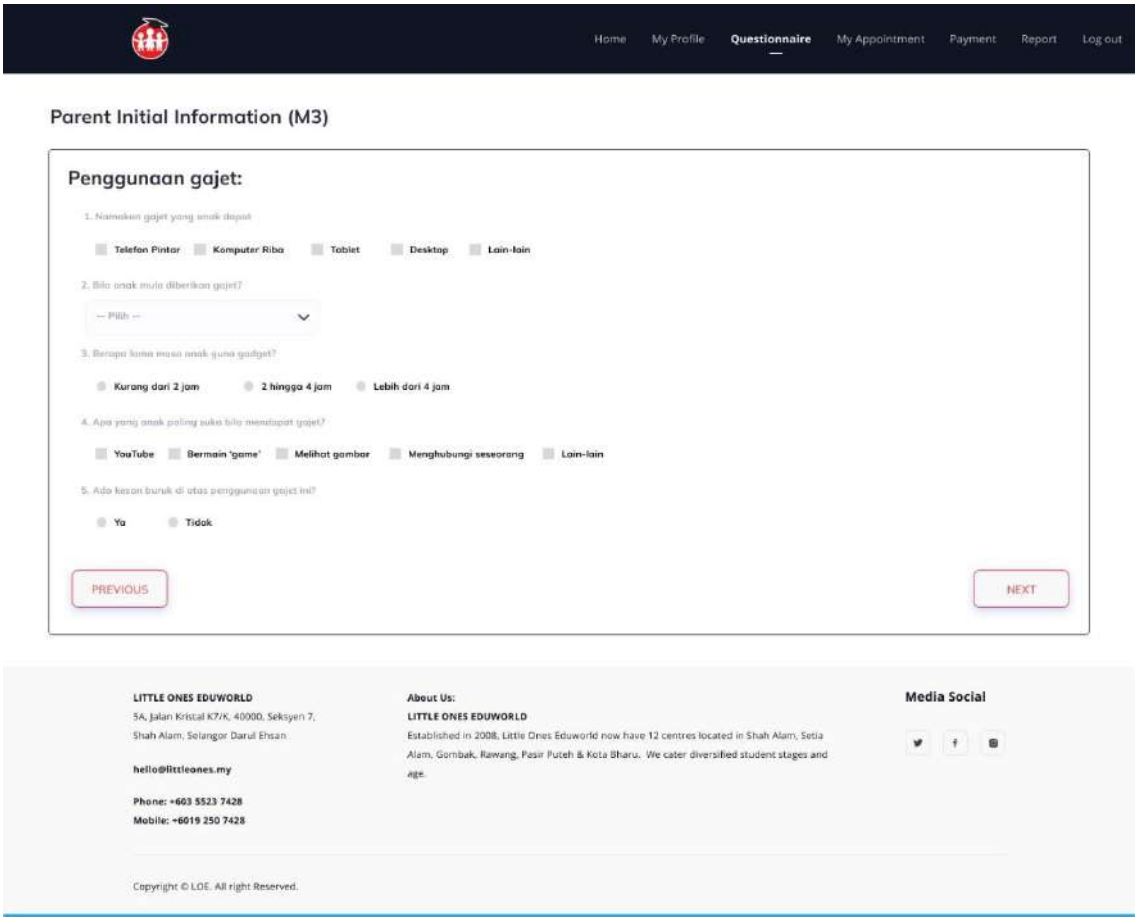
Appendix A. 15: Parent Initial Info Page 4



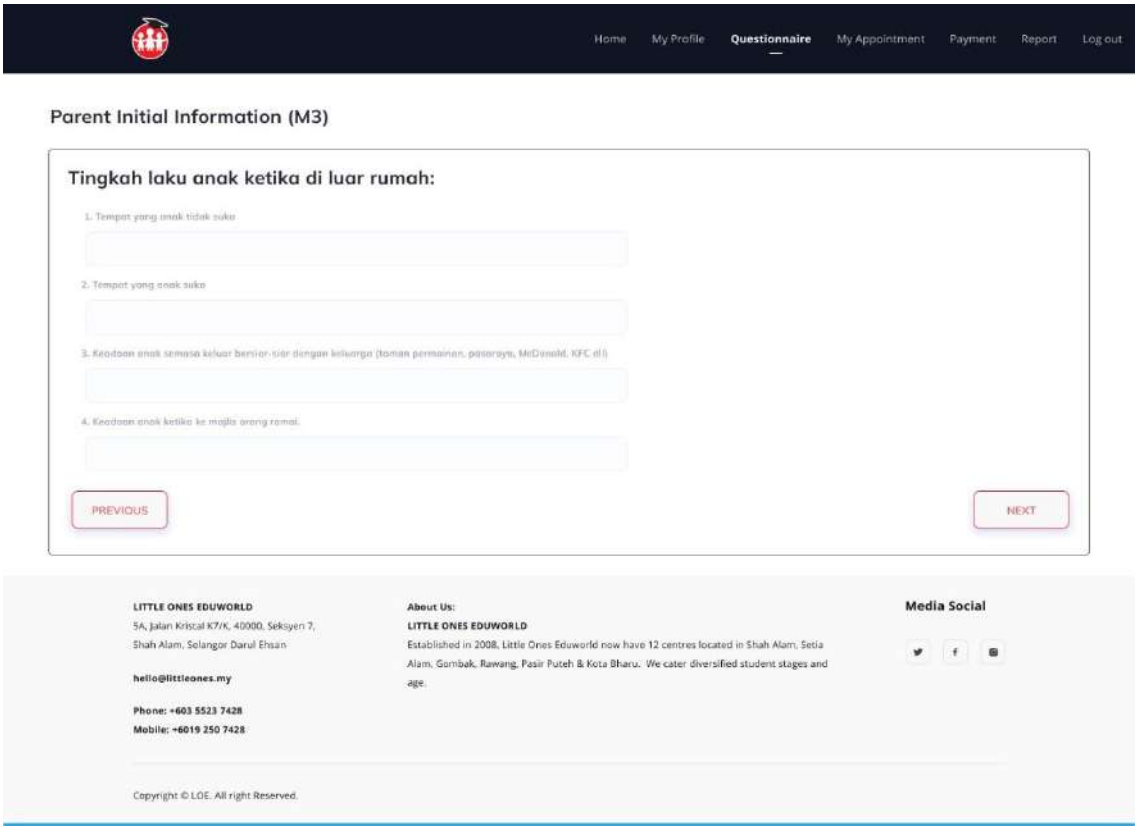
Appendix A. 16: Parent Initial Info Page 5



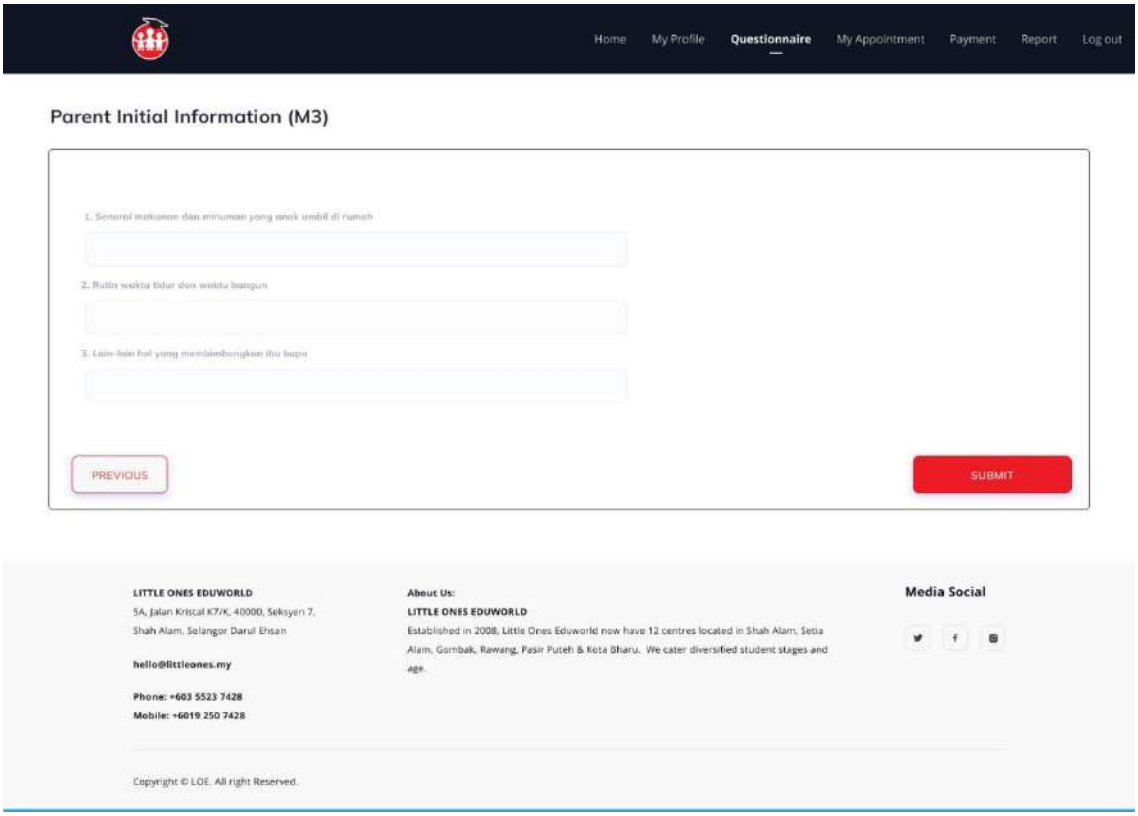
Appendix A. 17: Parent Initial Info Page 6



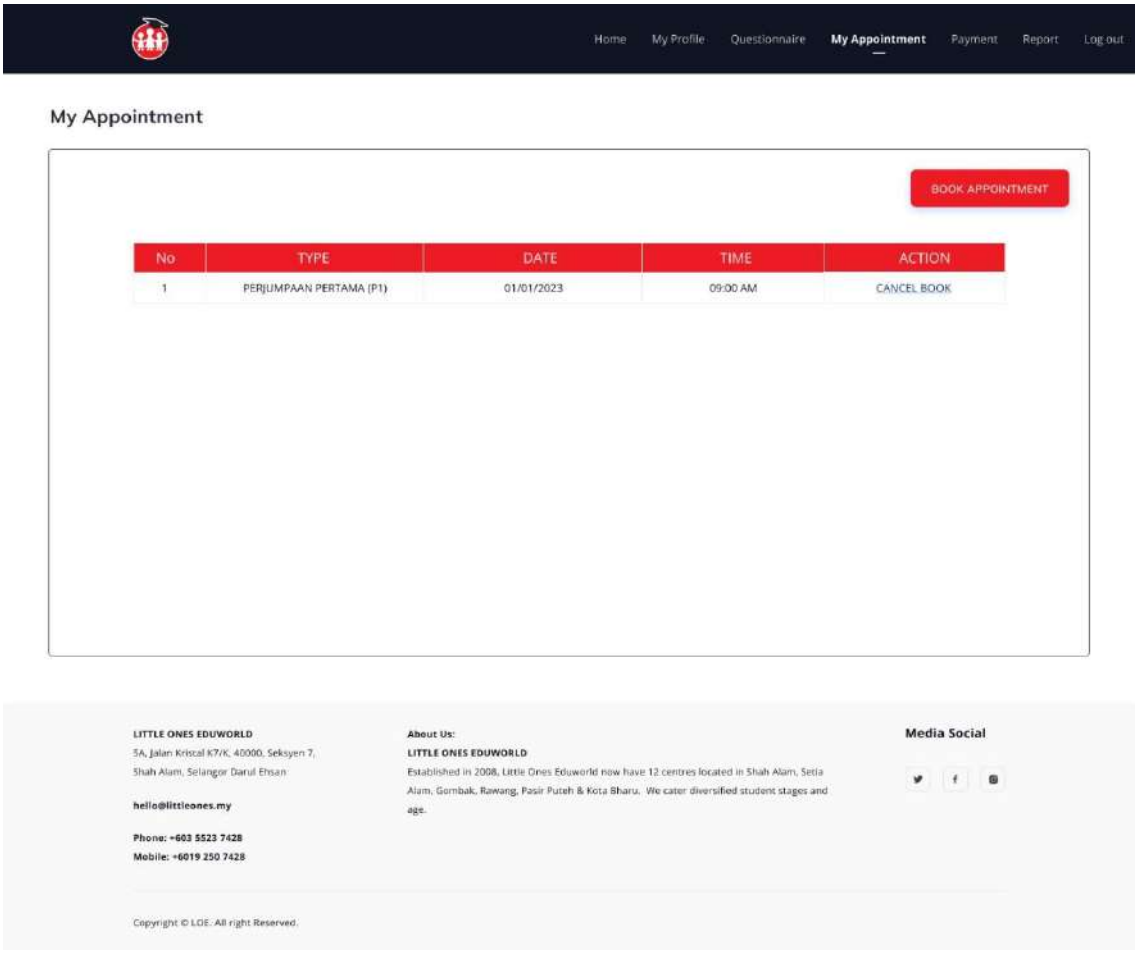
Appendix A. 18: Parent Initial Info Page 7



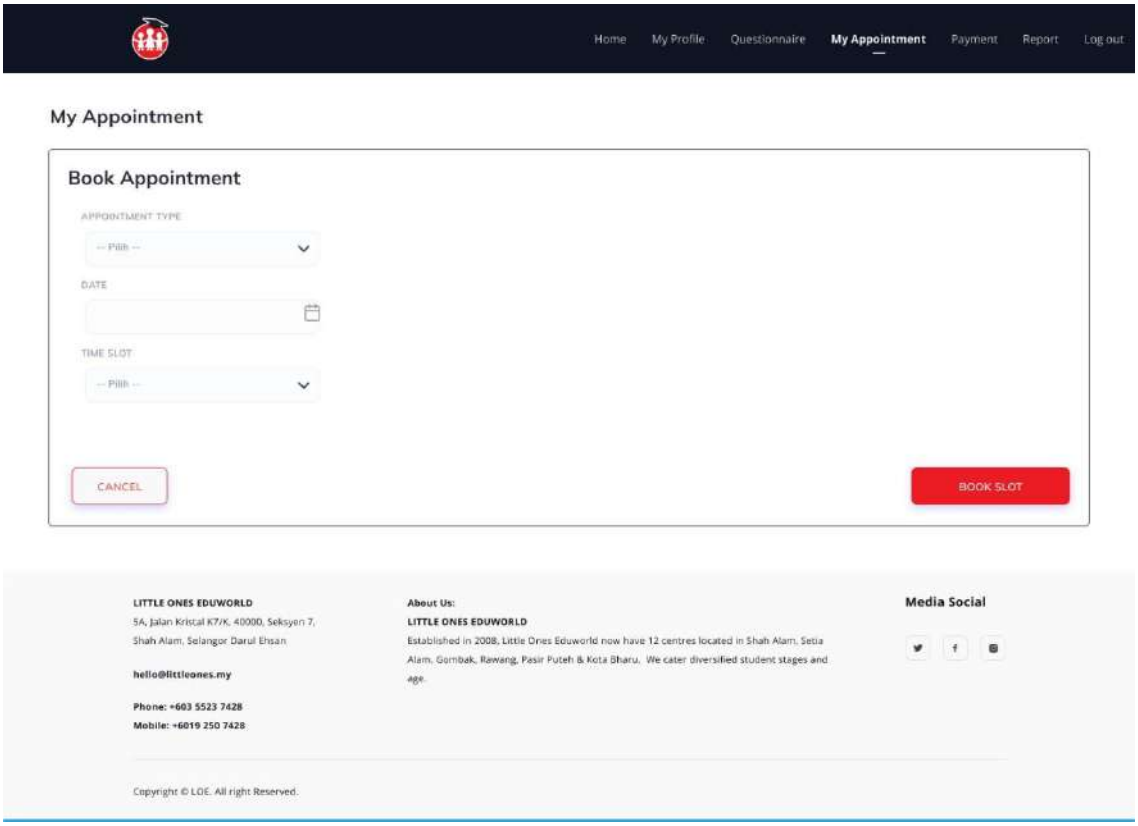
Appendix A. 19: Parent Initial Info Page 8



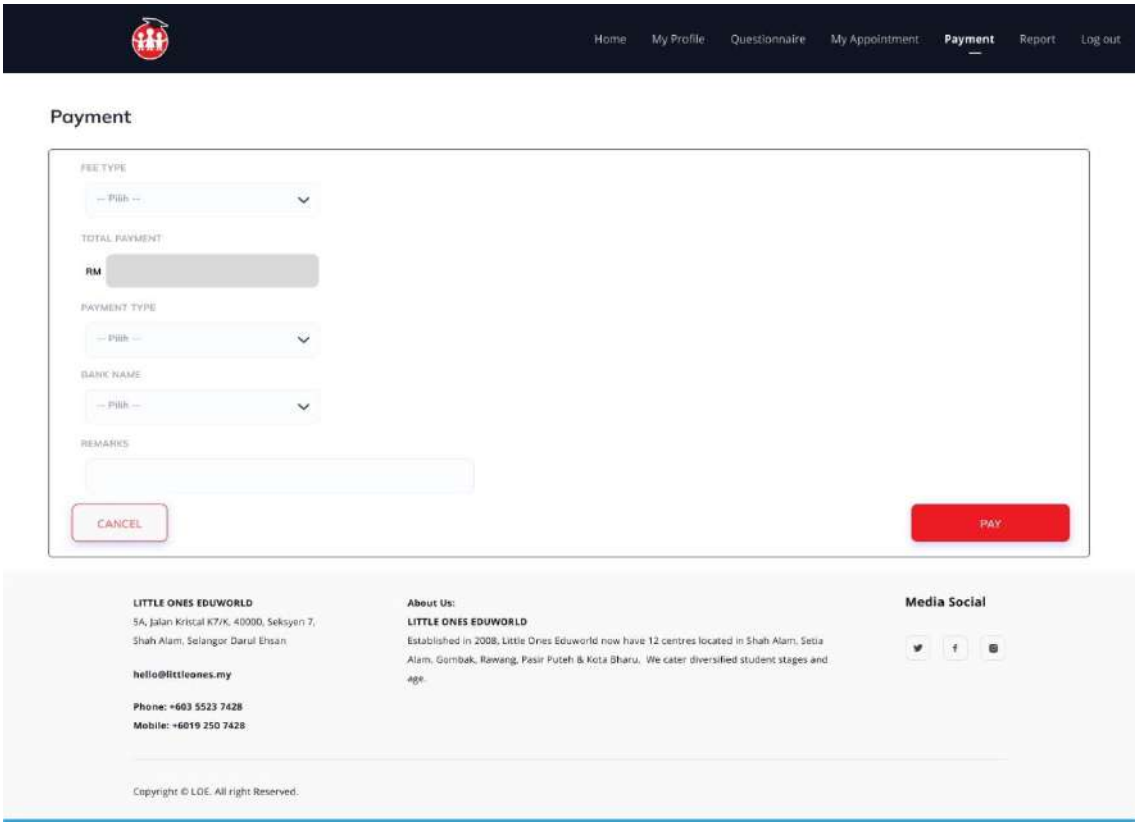
Appendix A. 20: Parent Initial Info Page 9



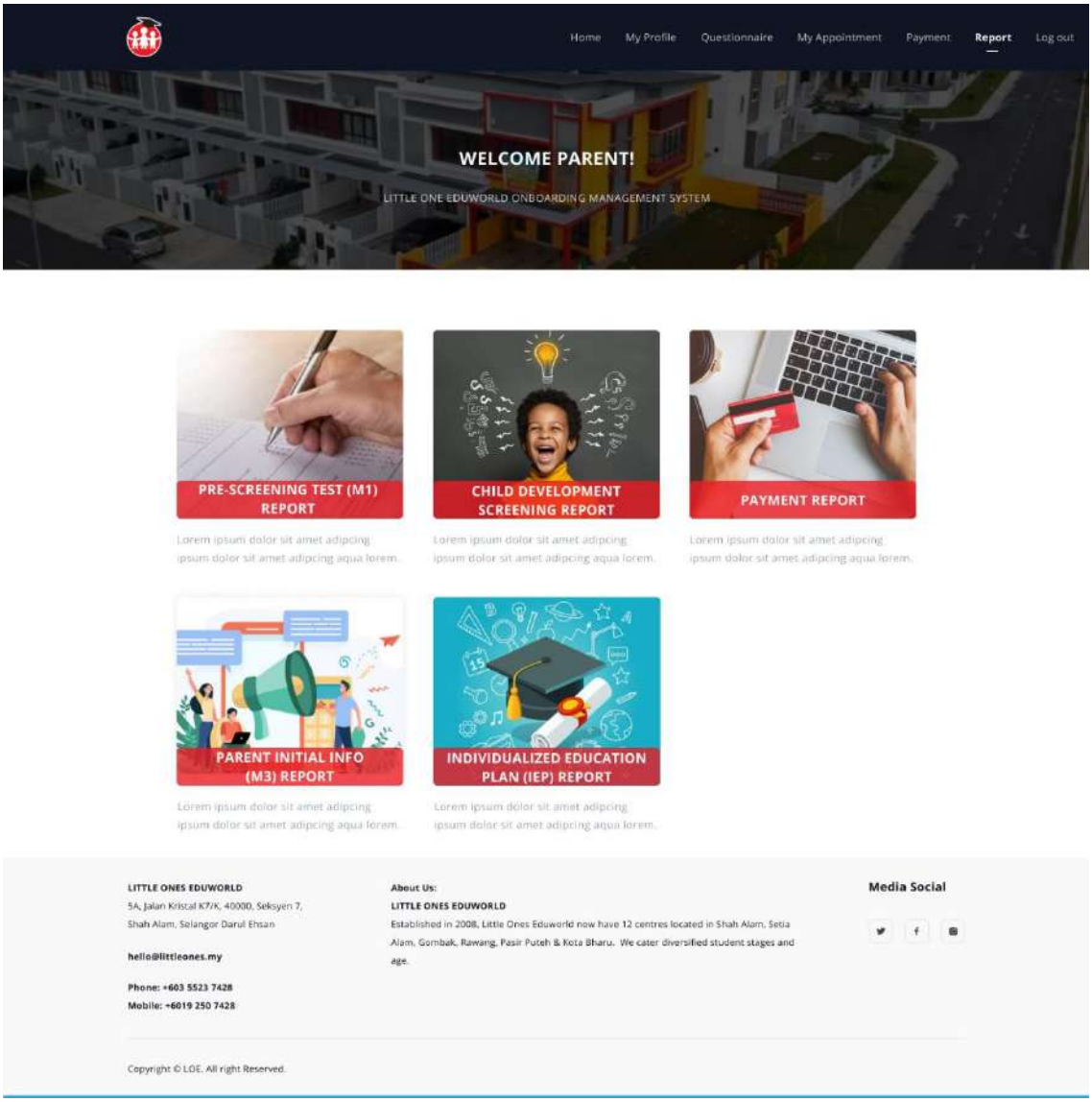
Appendix A. 21: Appointment list page



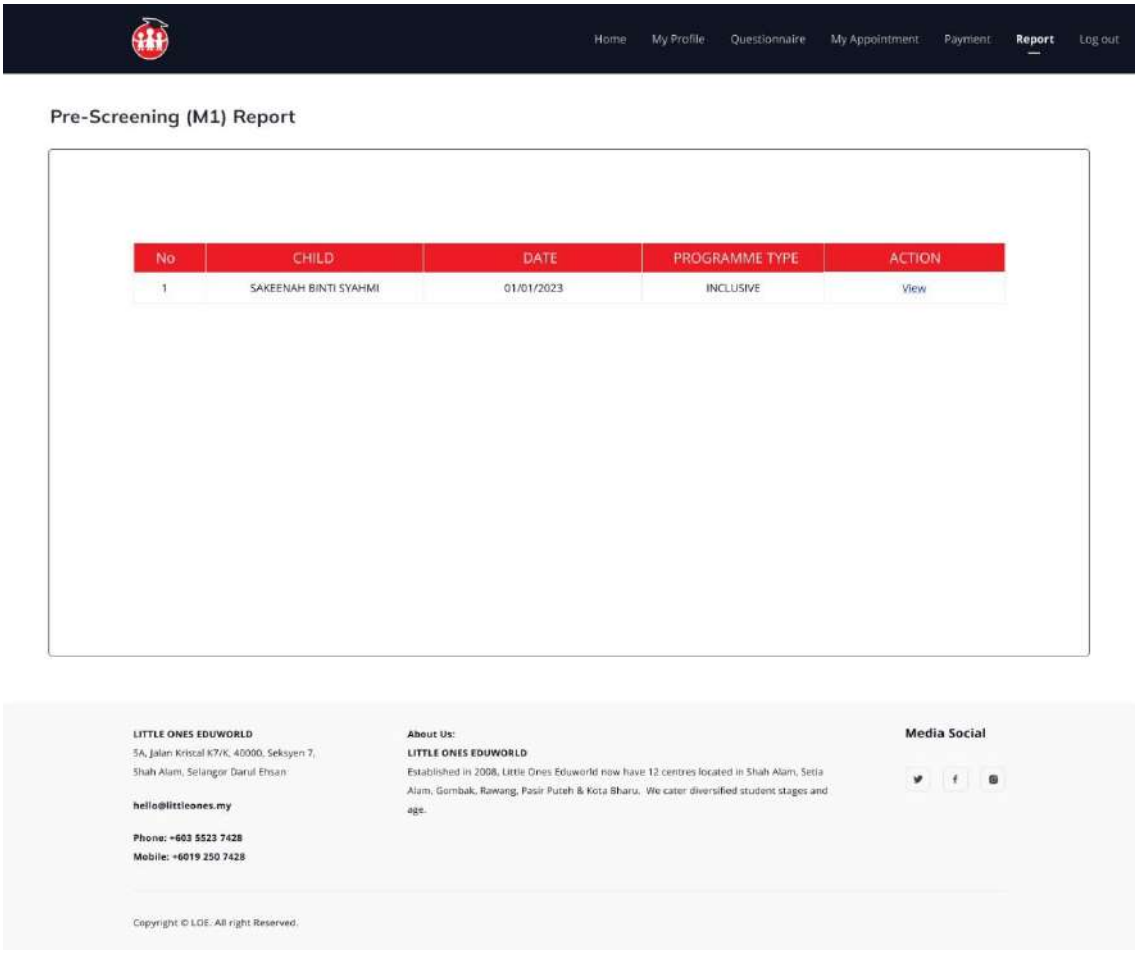
Appendix A. 22: Book Appointment page



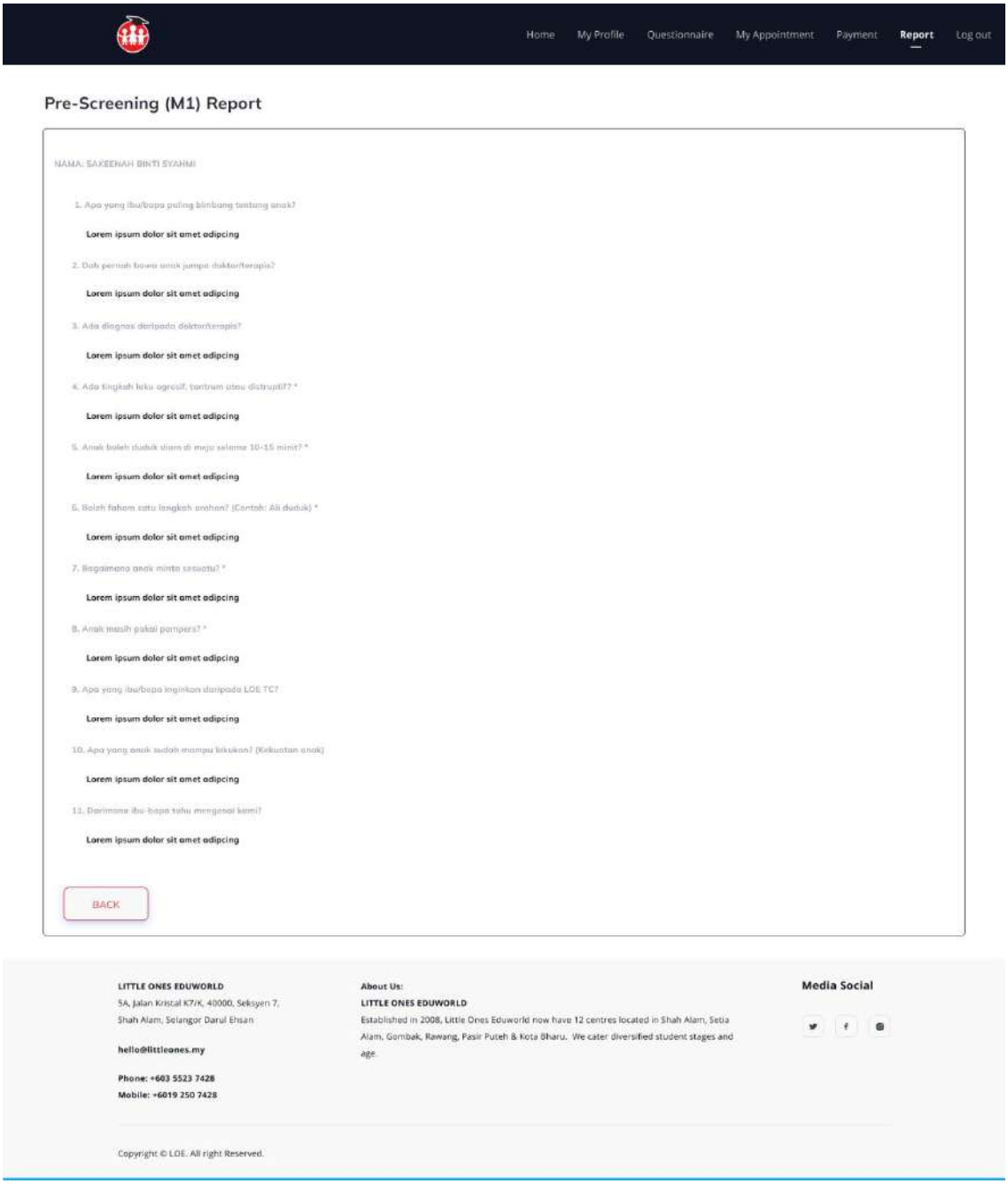
Appendix A. 23: Payment page



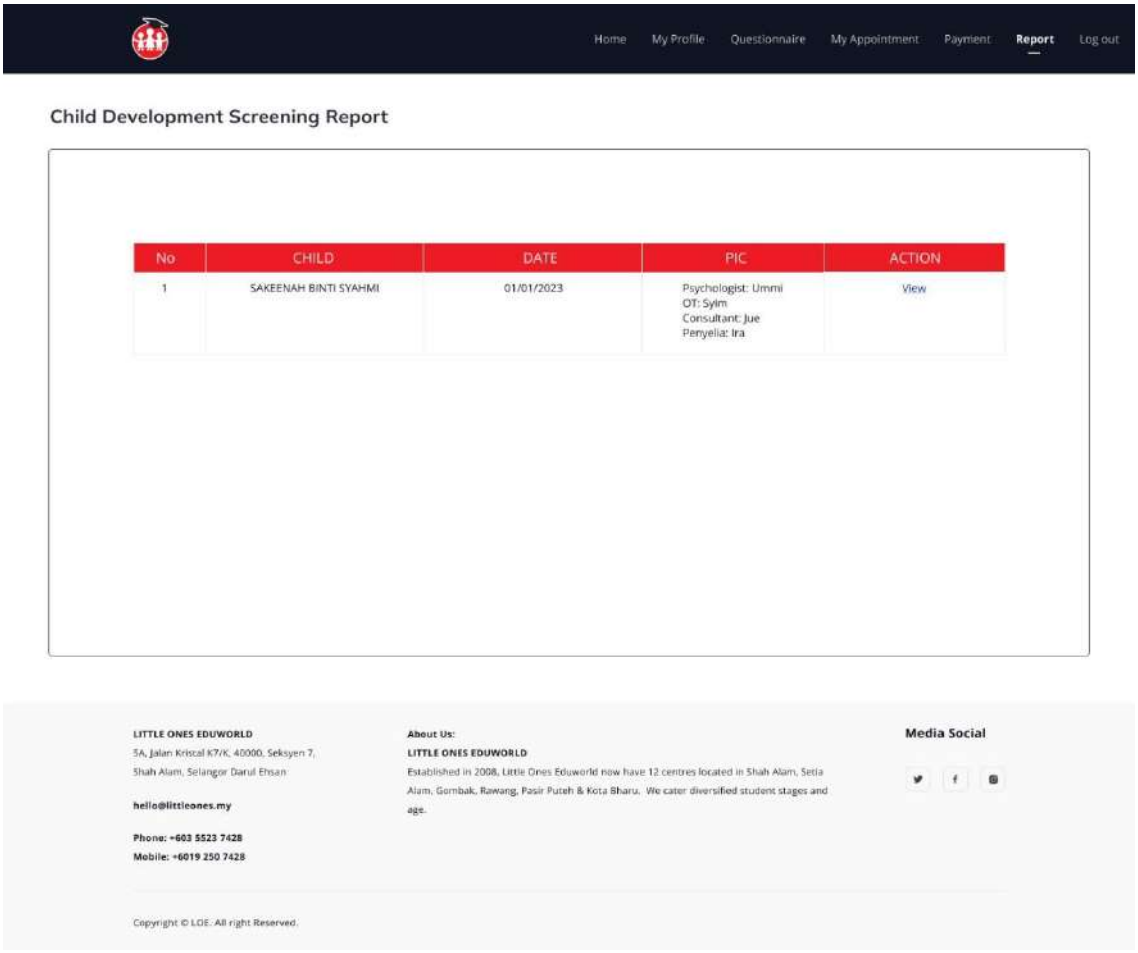
Appendix A. 24: Parent dashboard page



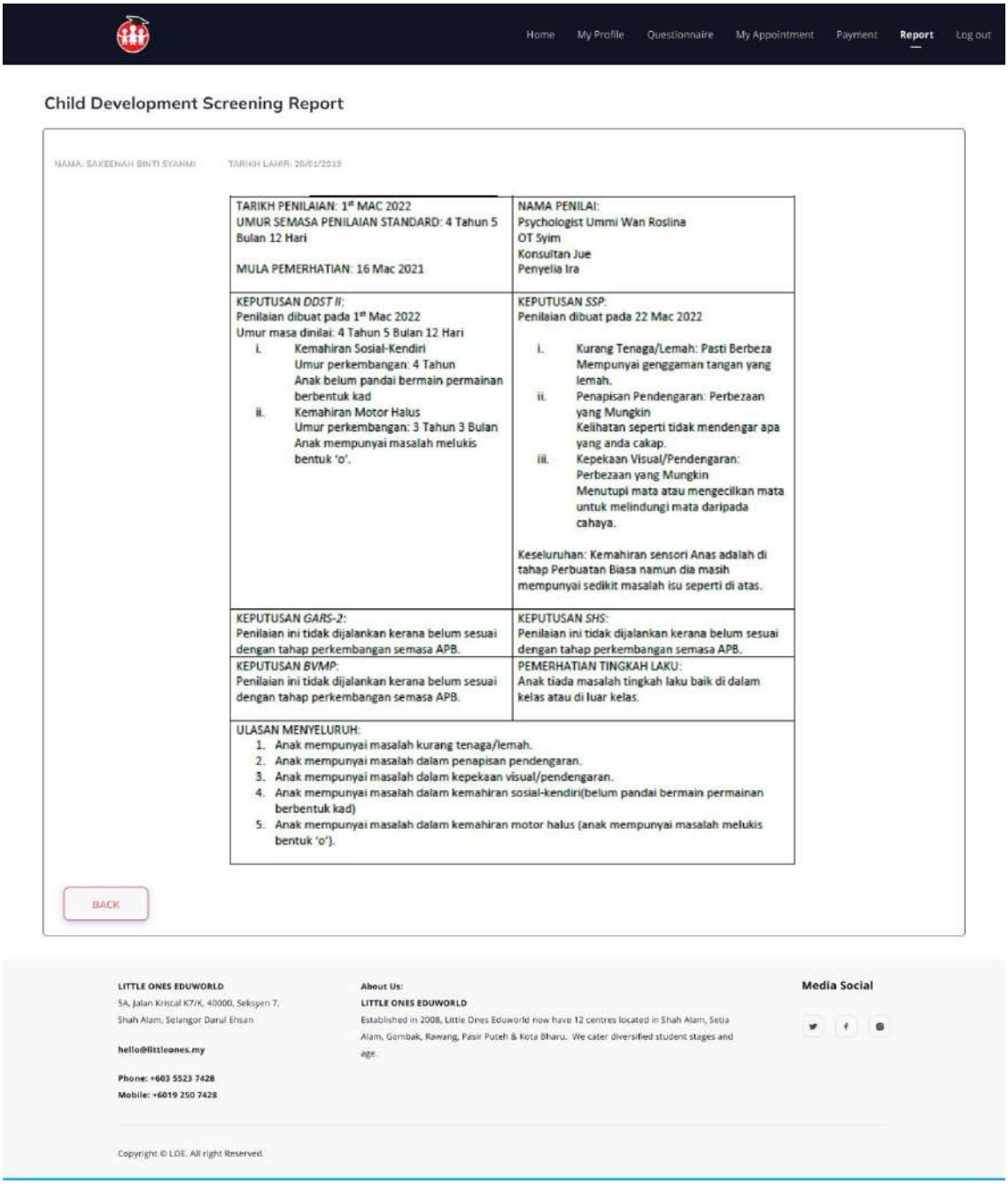
Appendix A. 25: Pre-screening report list page



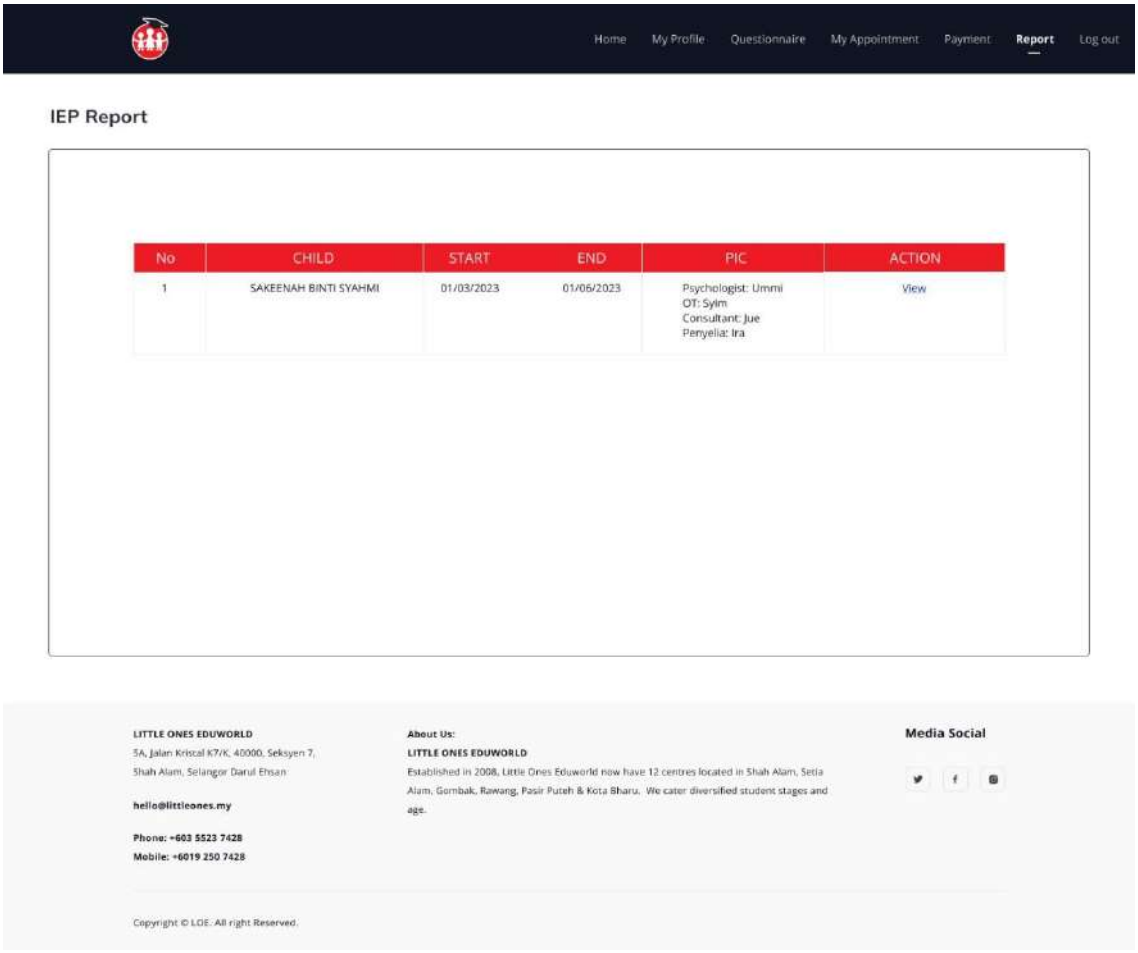
Appendix A. 26: Pre-screening report details page



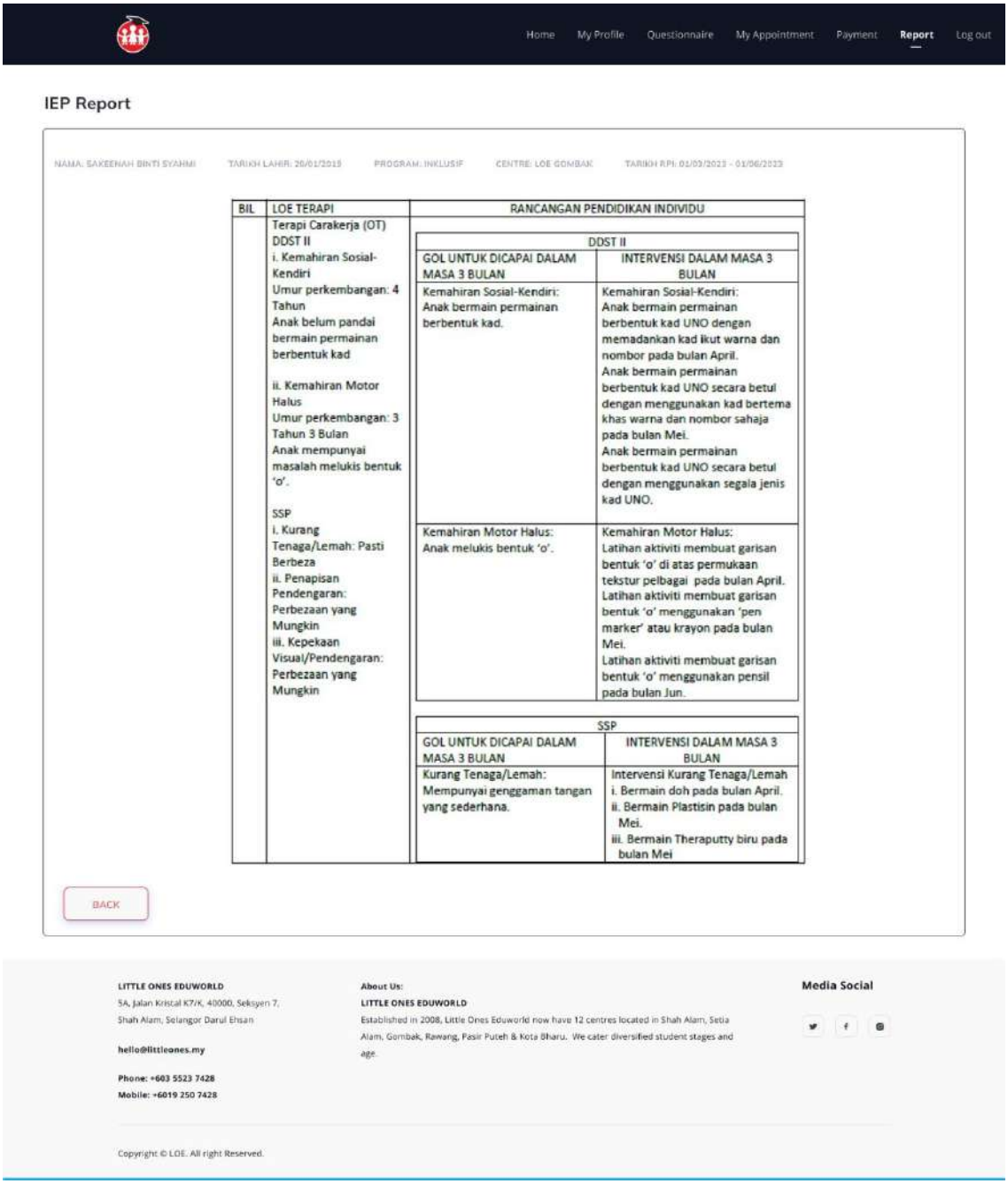
Appendix A. 27: Child development screening report list page



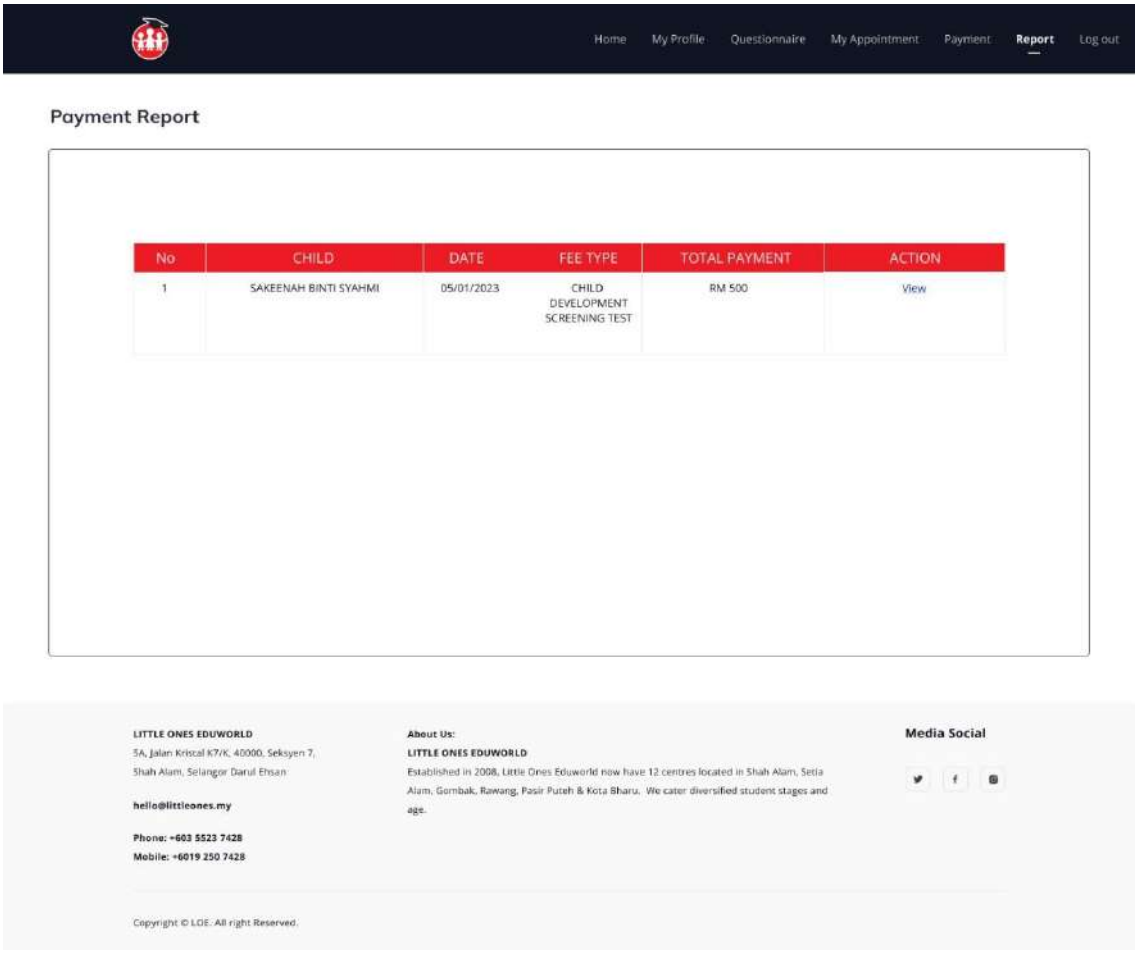
Appendix A. 28: Child development screening report details page



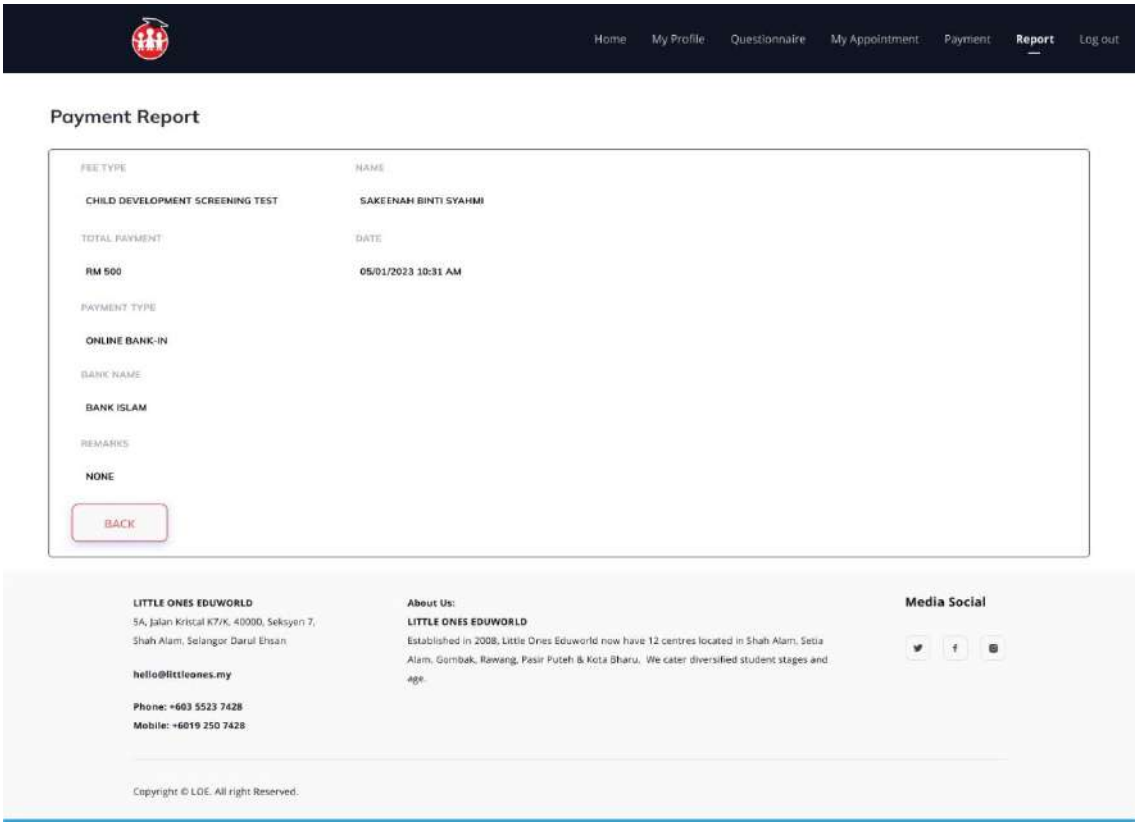
Appendix A. 29: IEP report list page



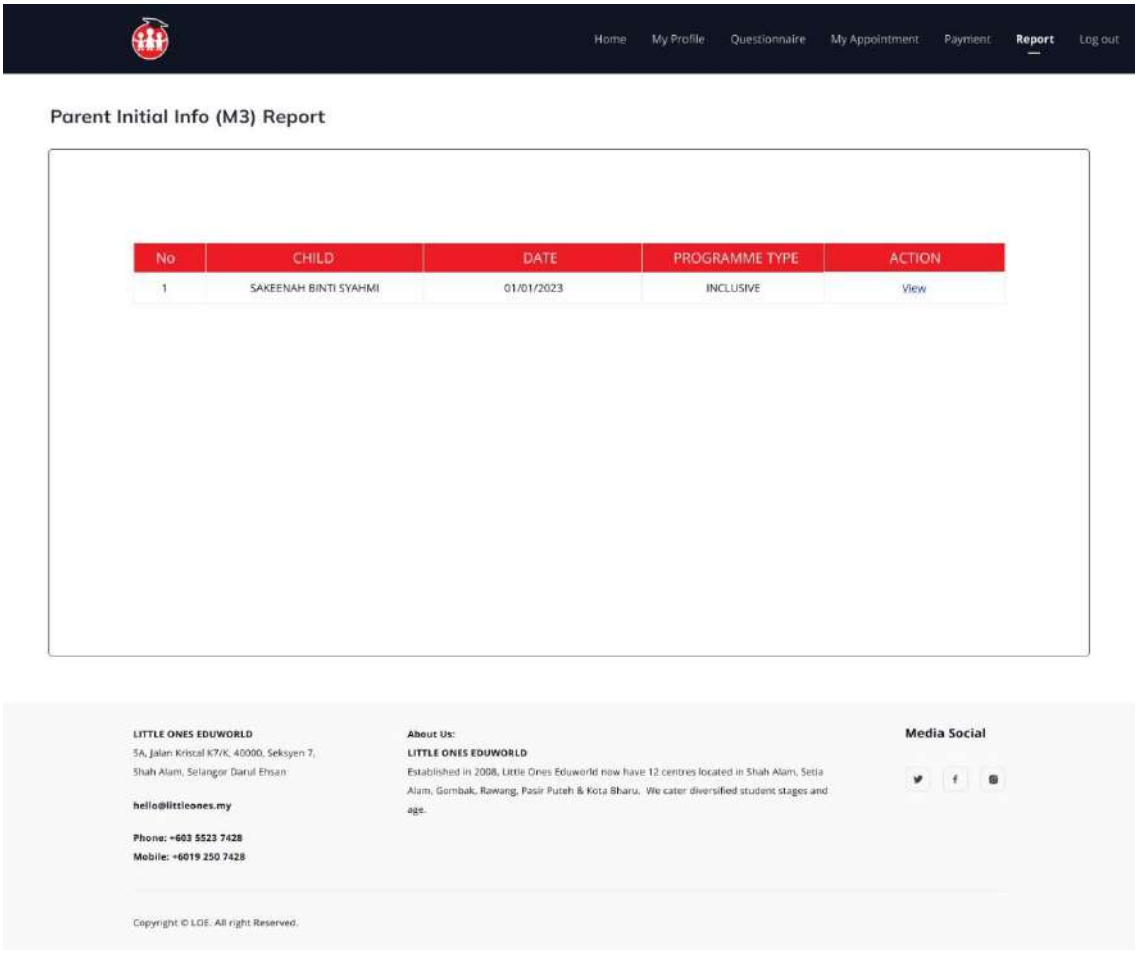
Appendix A. 30: IEP report details page



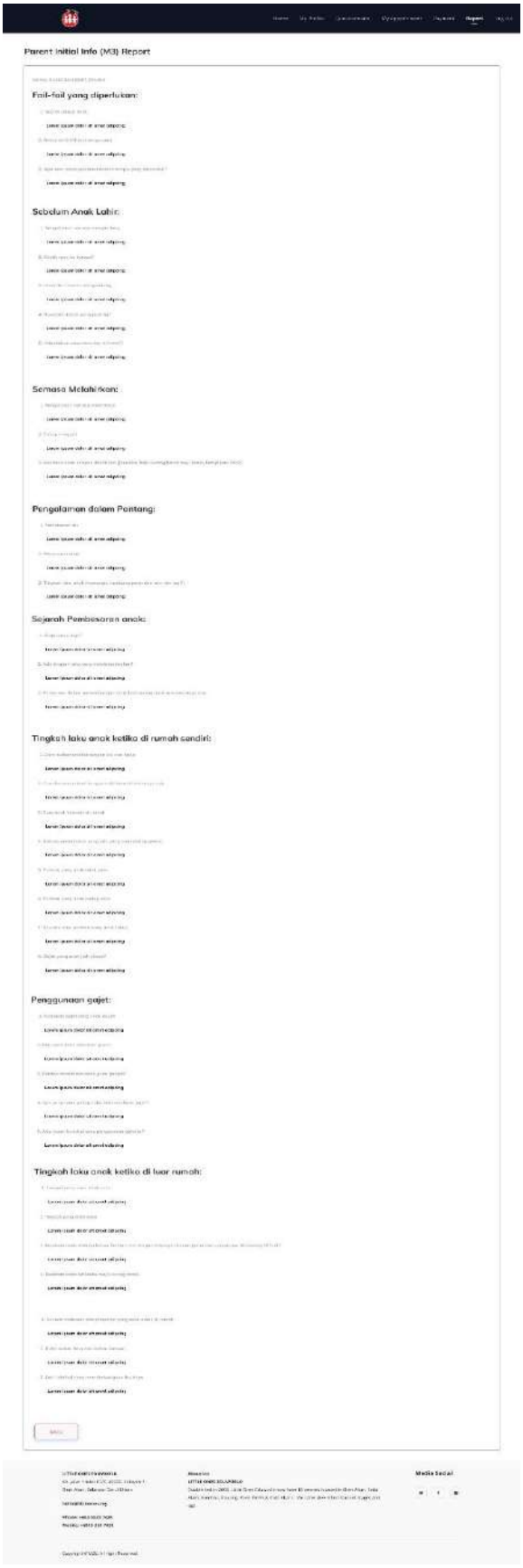
Appendix A. 31: Payment report list page



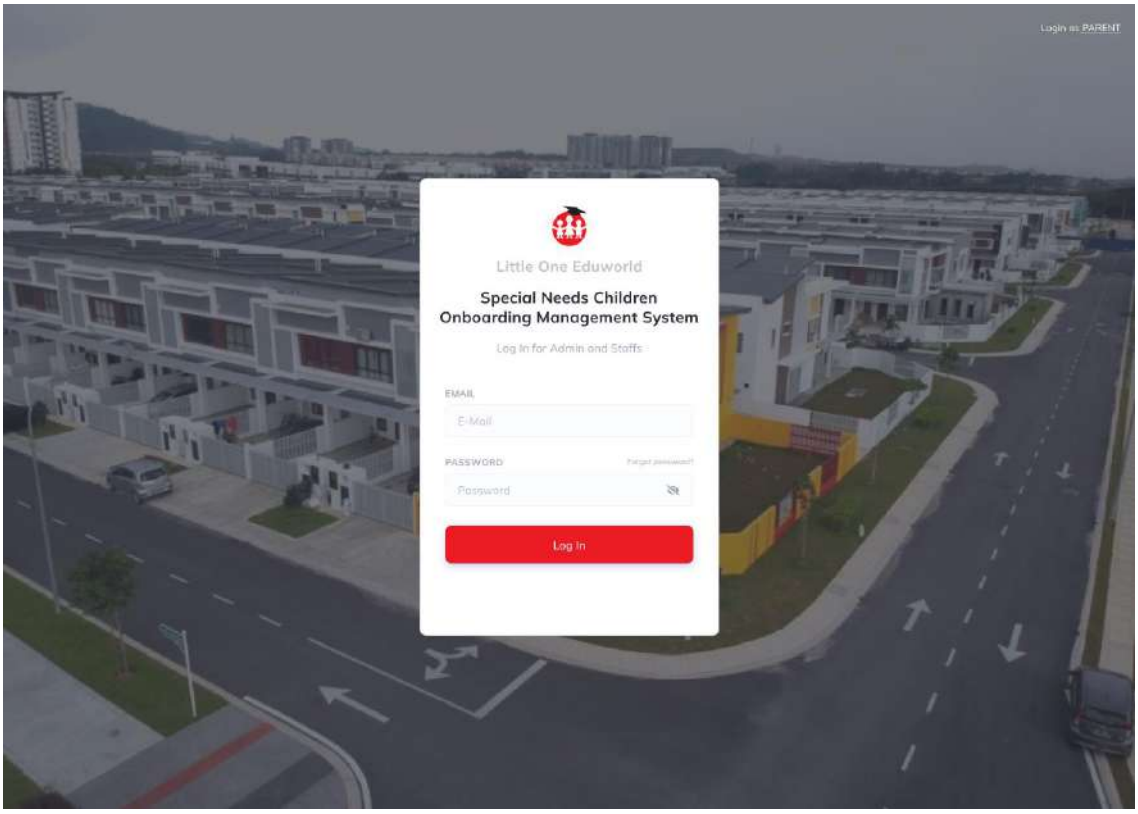
Appendix A. 32: Payment report details page



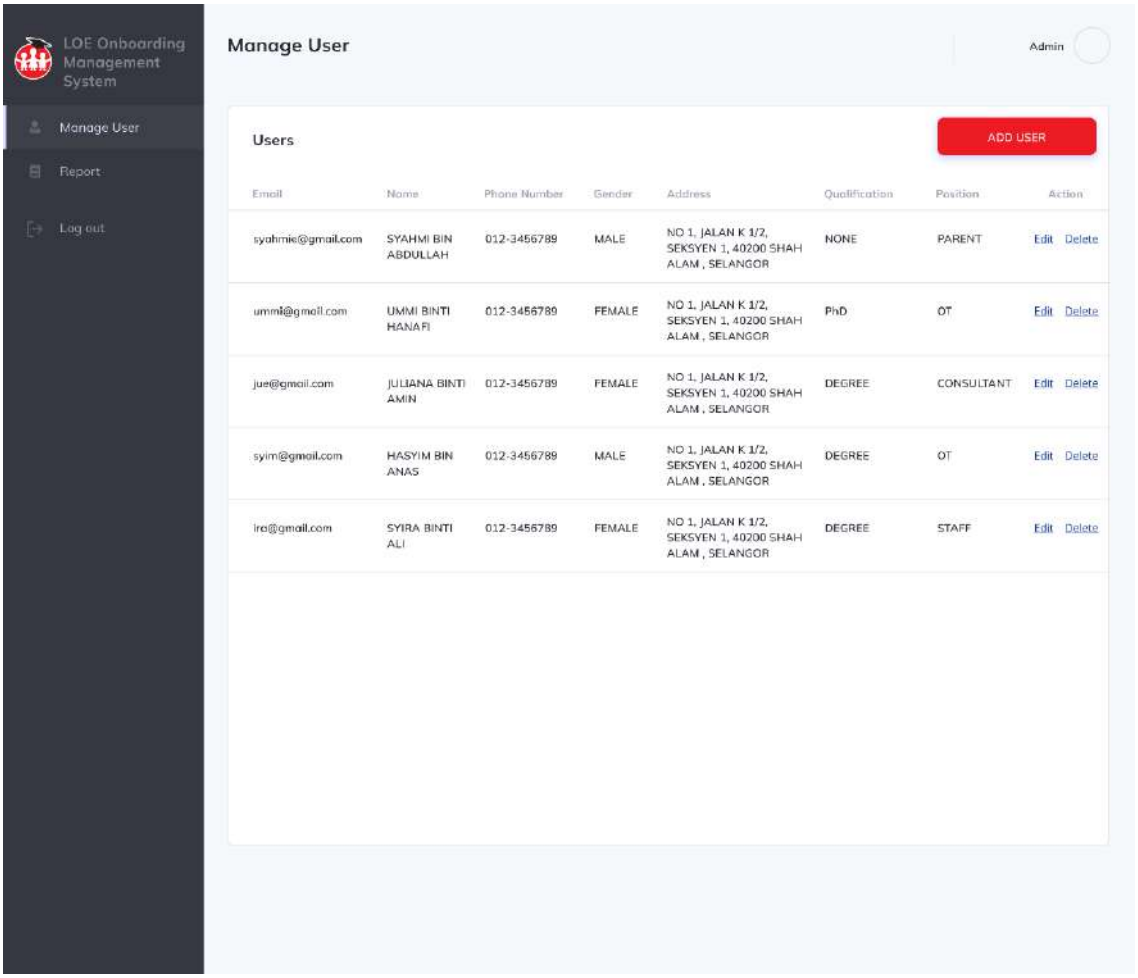
Appendix A. 33: Parent Initial Info report list page



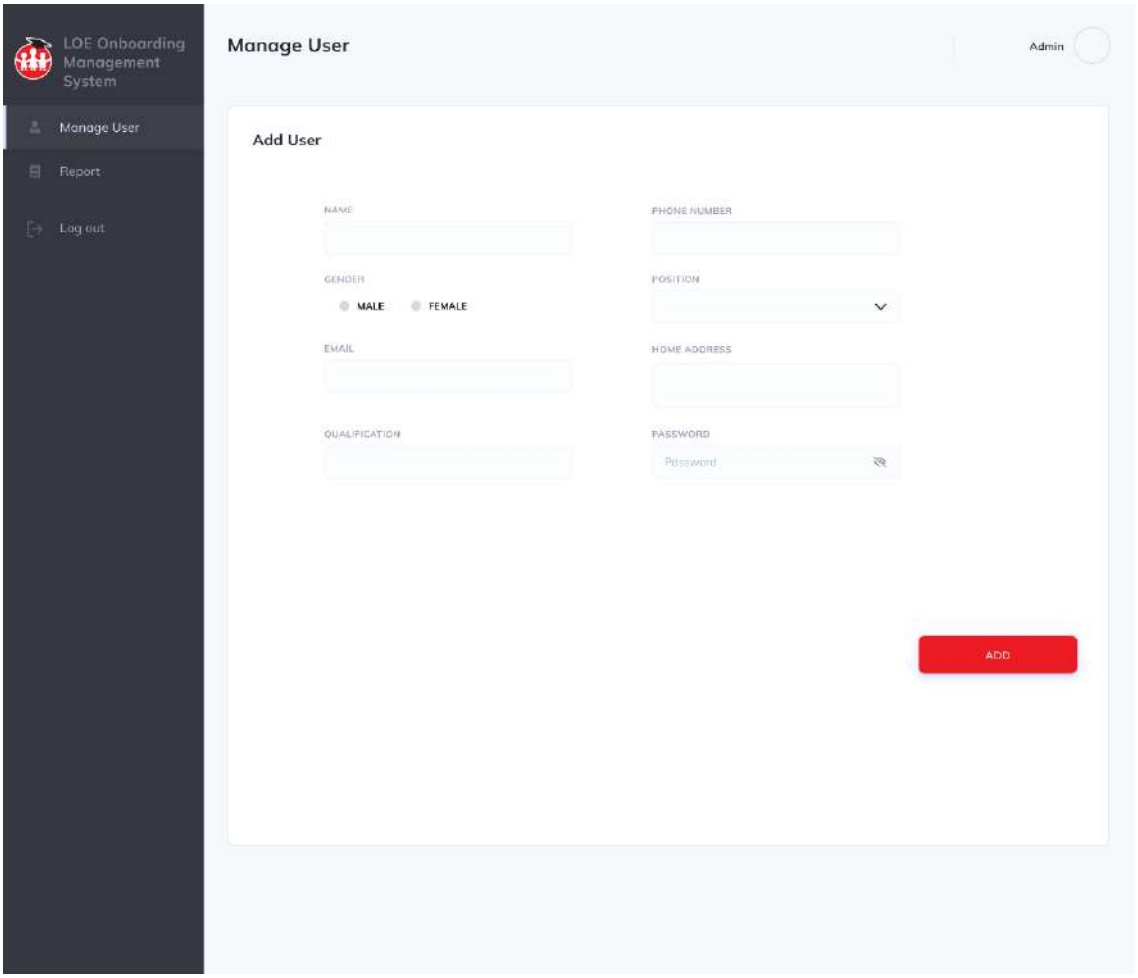
Appendix A. 34: Parent Initial Info report details page



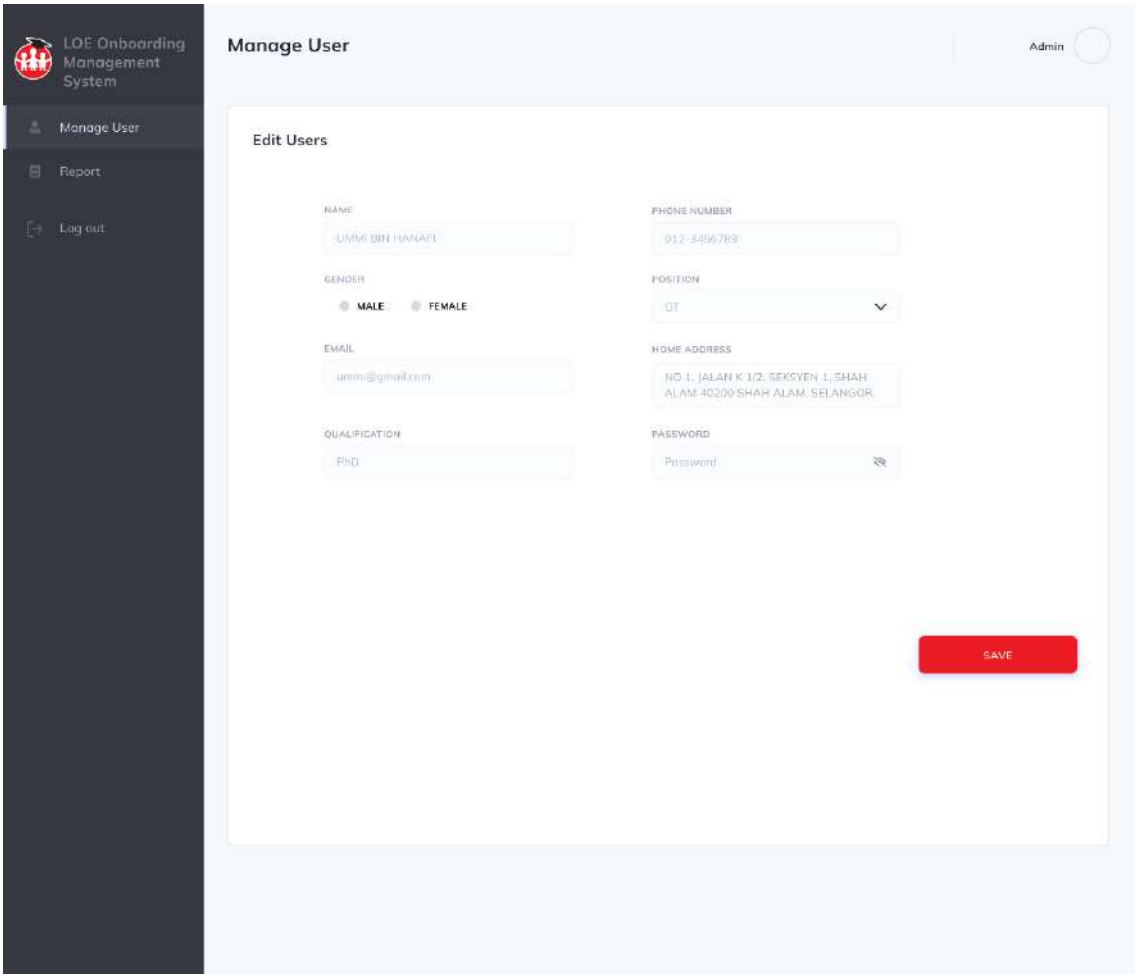
Appendix A. 35: Login page Admin/LOE Staff



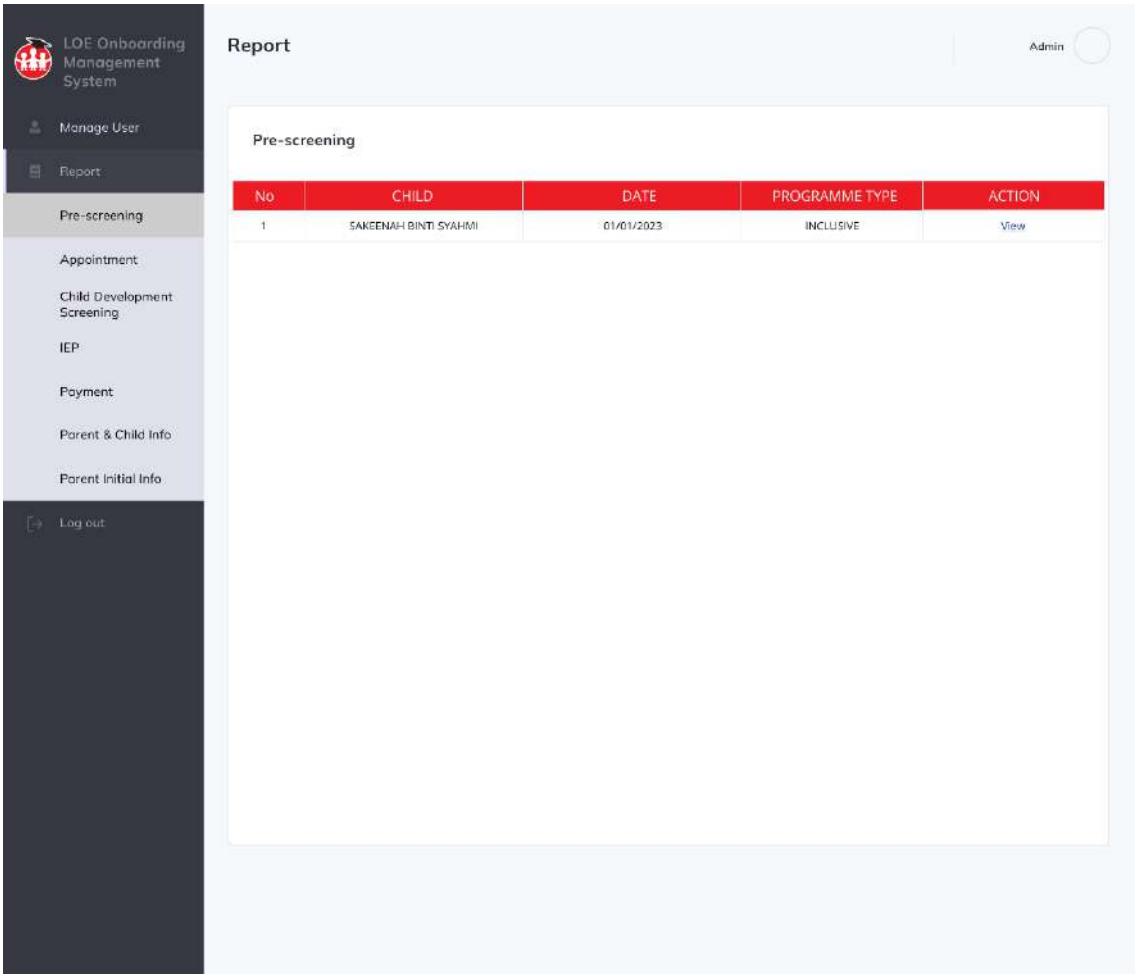
Appendix A. 36: Manage User page



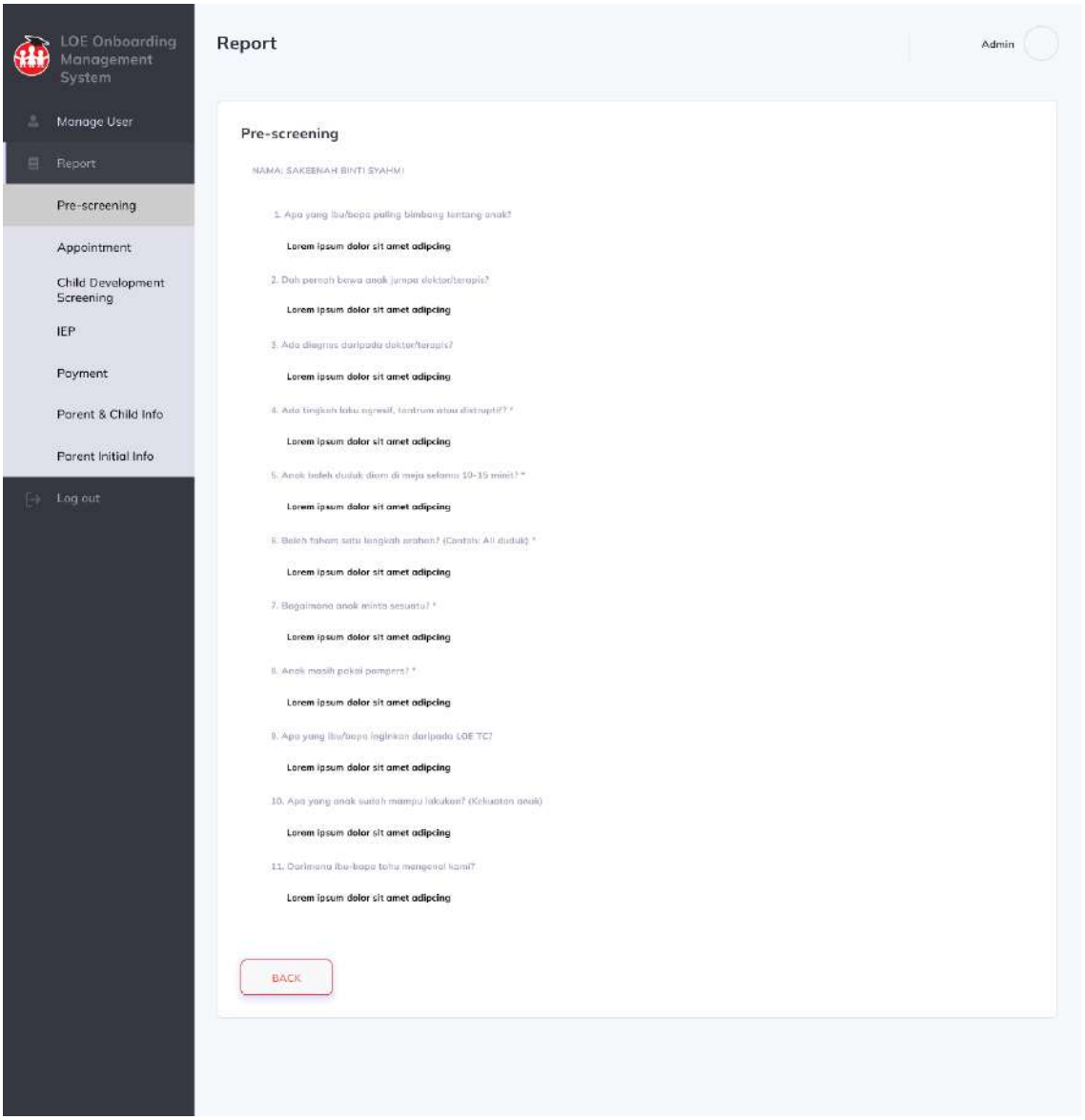
Appendix A. 37: Add user page



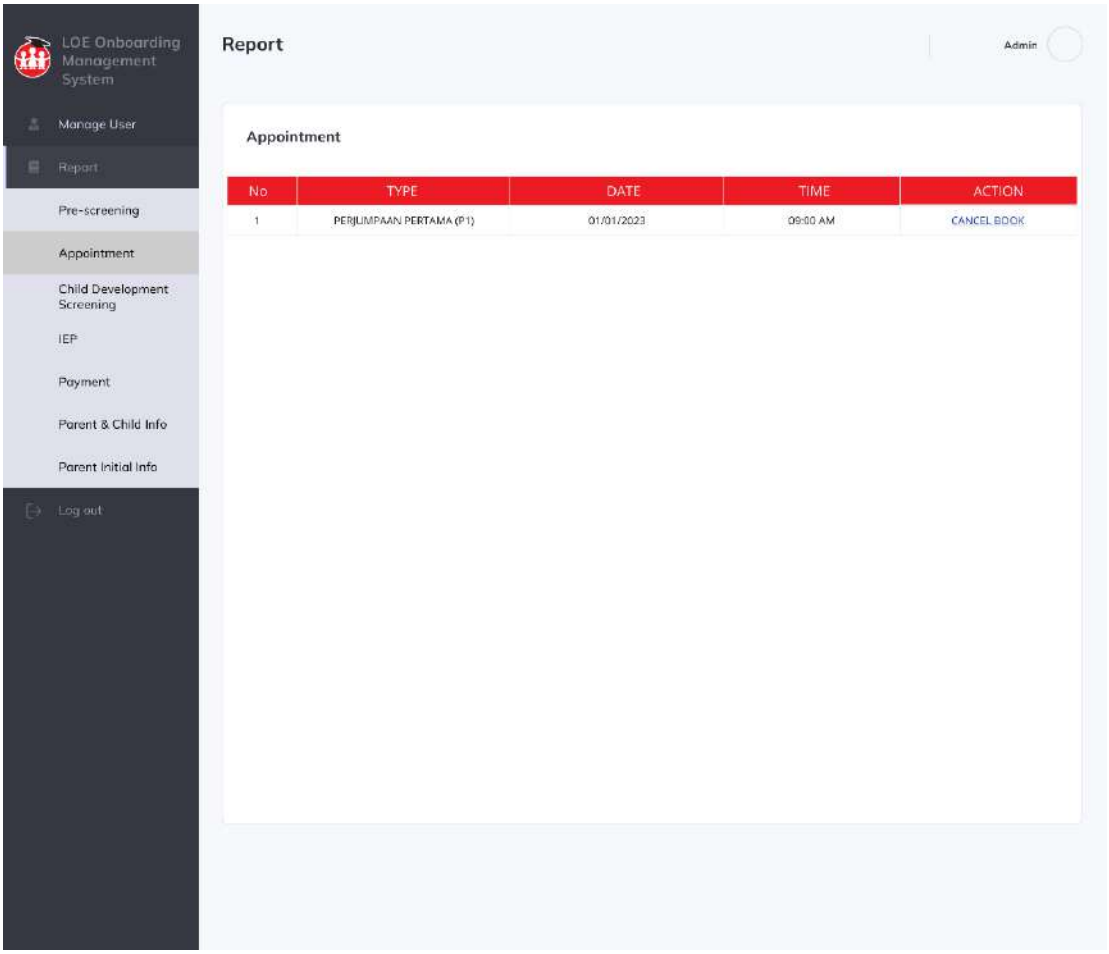
Appendix A. 38: Edit user page



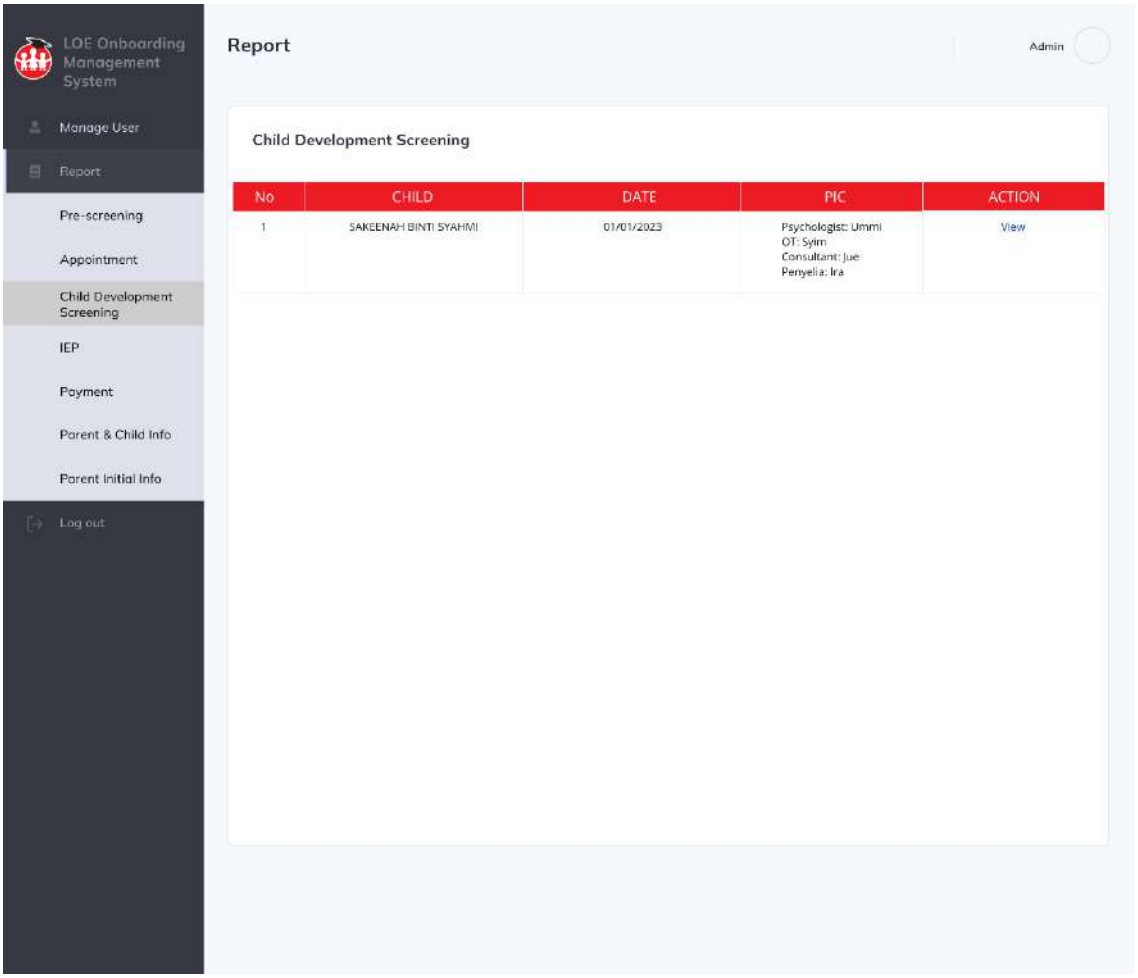
Appendix A. 39: Pre-screening report list page Admin



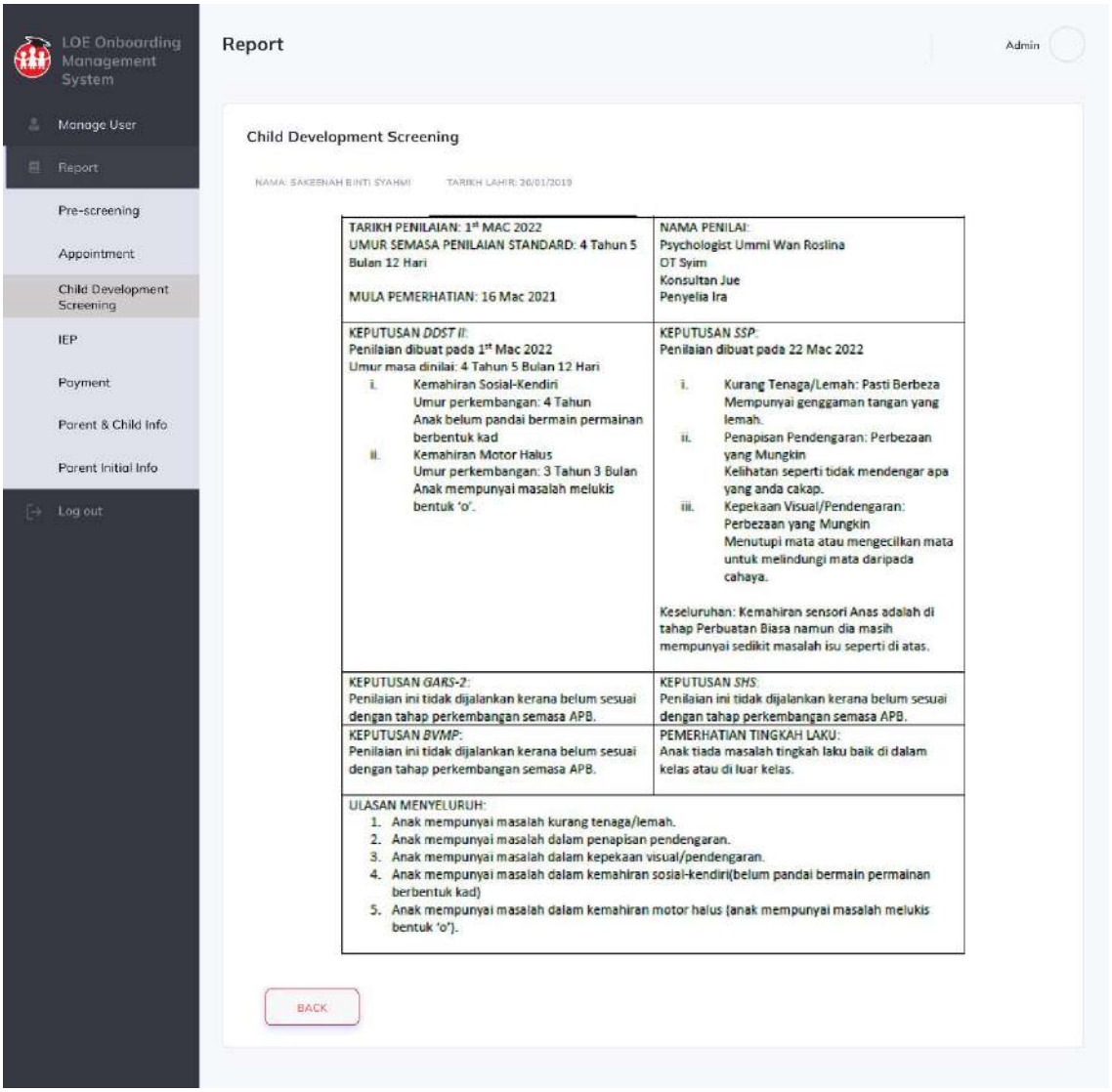
Appendix A. 40: Pre-screening report details page Admin



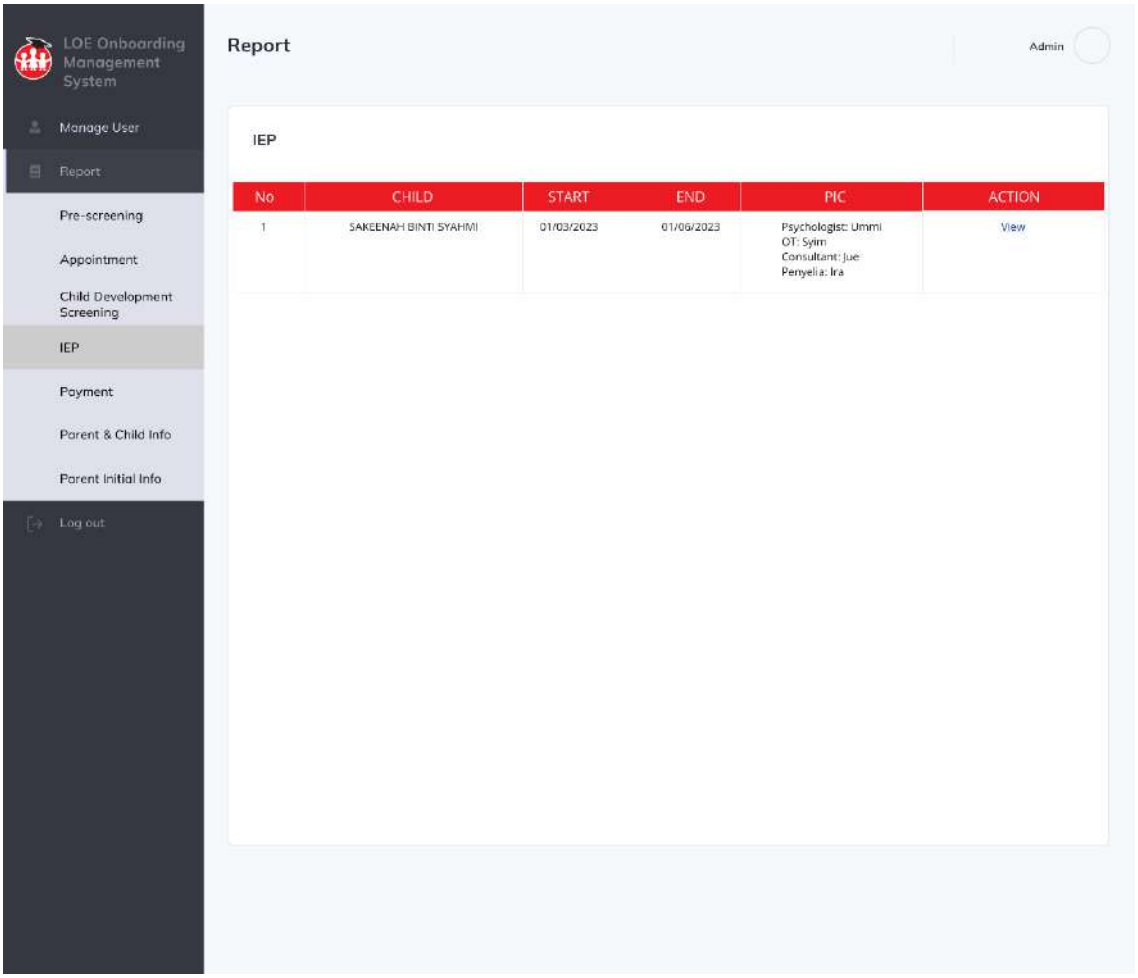
Appendix A. 41: Appointment list page Admin



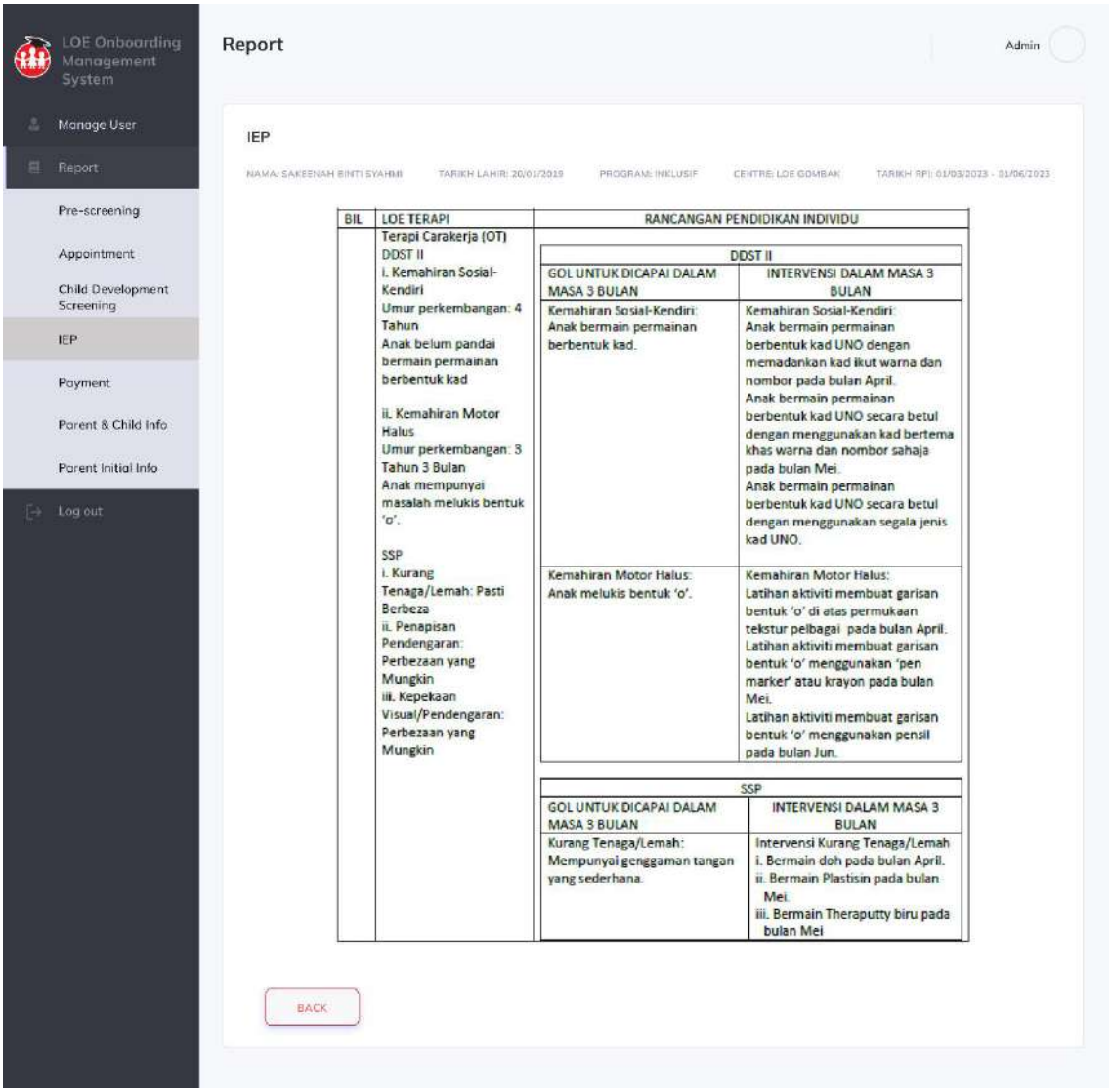
Appendix A. 42: Child Development screening report list page Admin & LOE staff



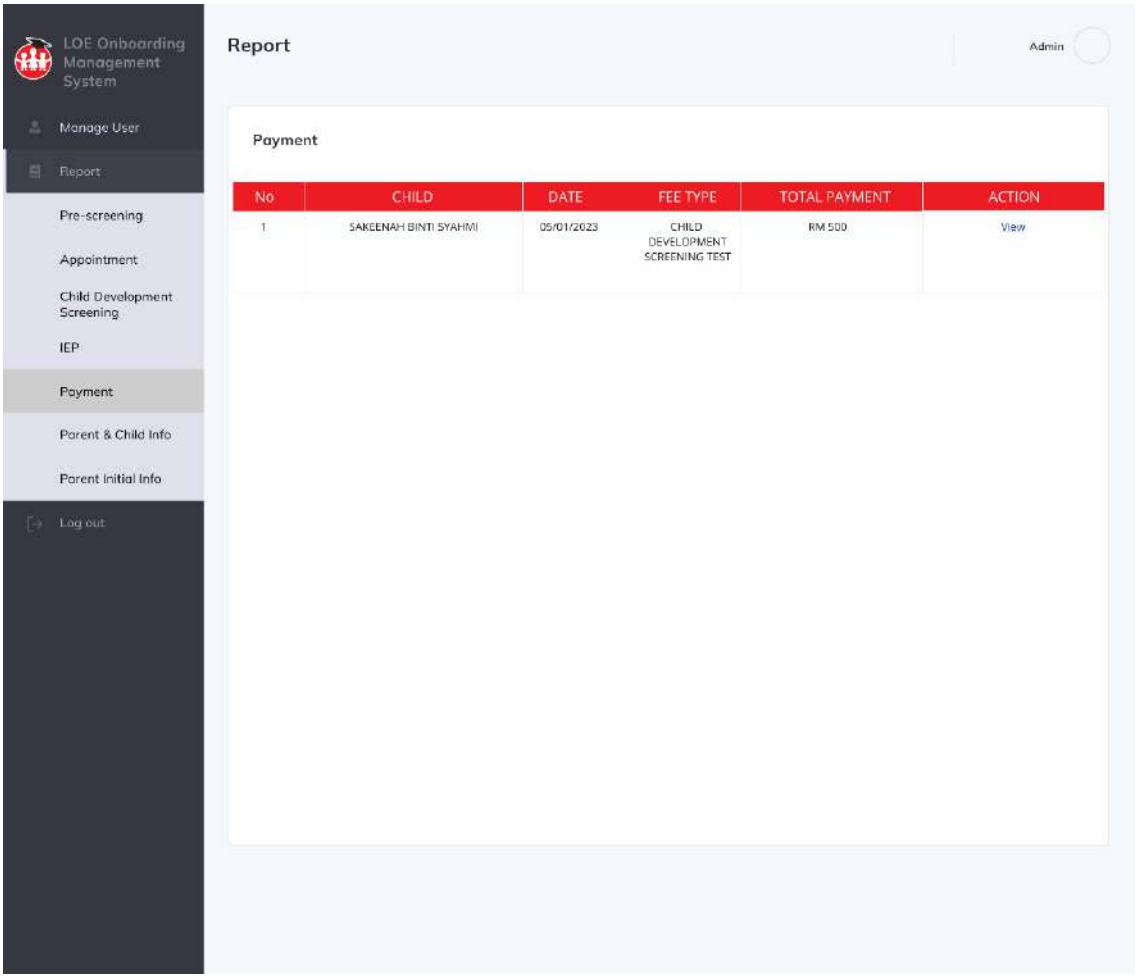
Appendix A. 43: Child Development screening report details page Admin & LOE staff



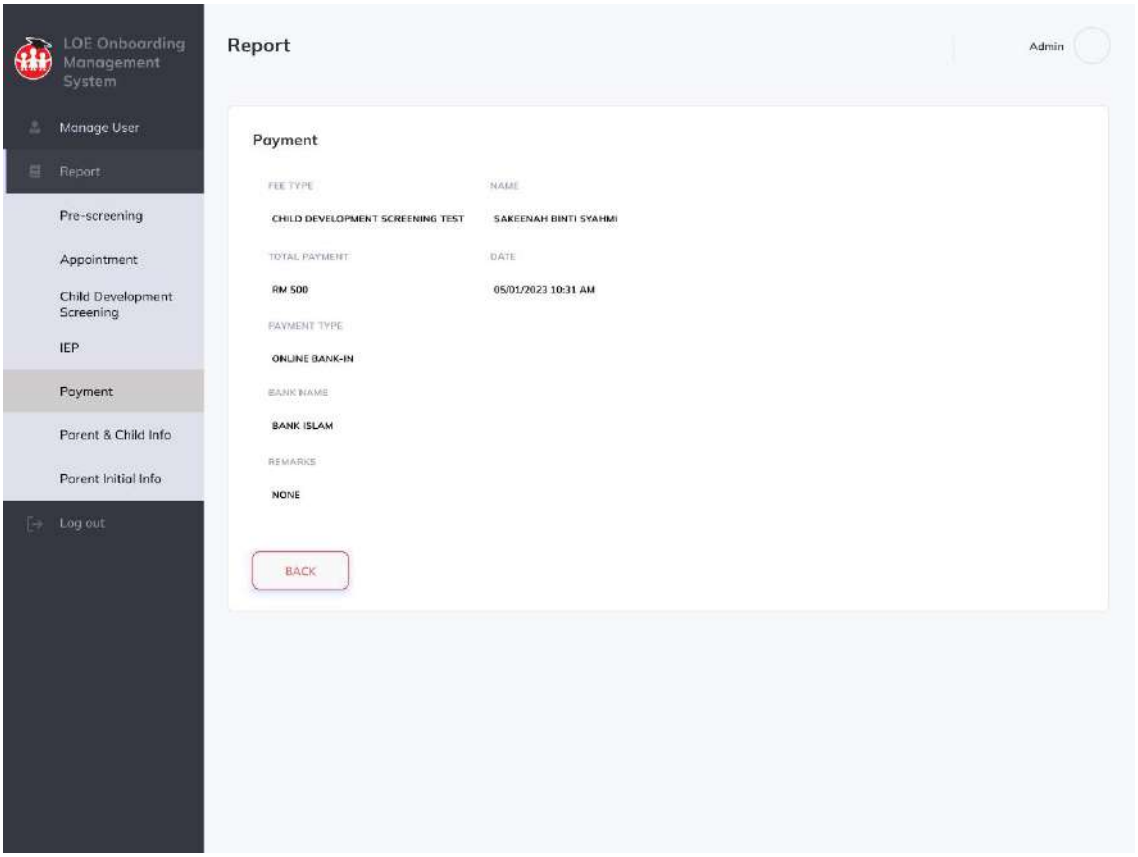
Appendix A. 44: IEP report list page Admin



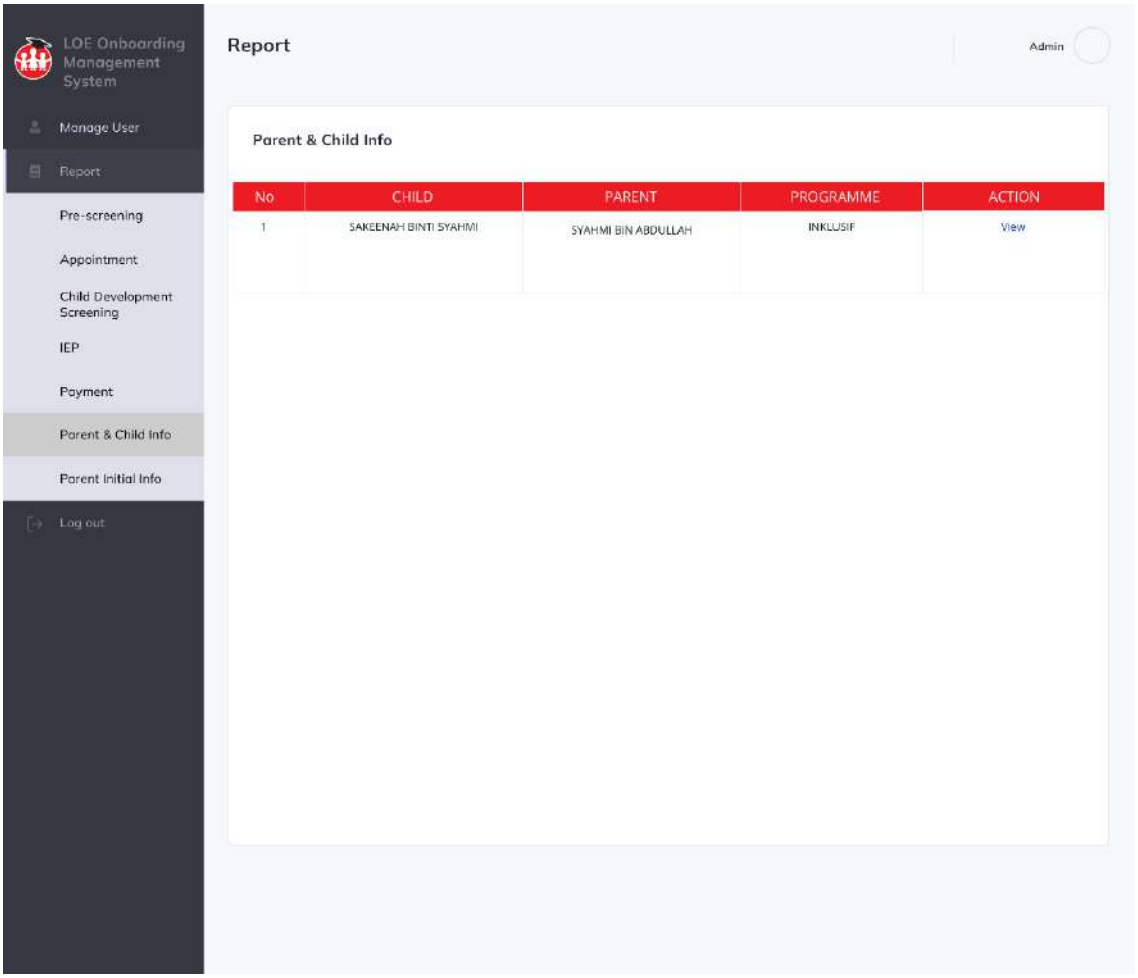
Appendix A. 45: IEP report details page Admin



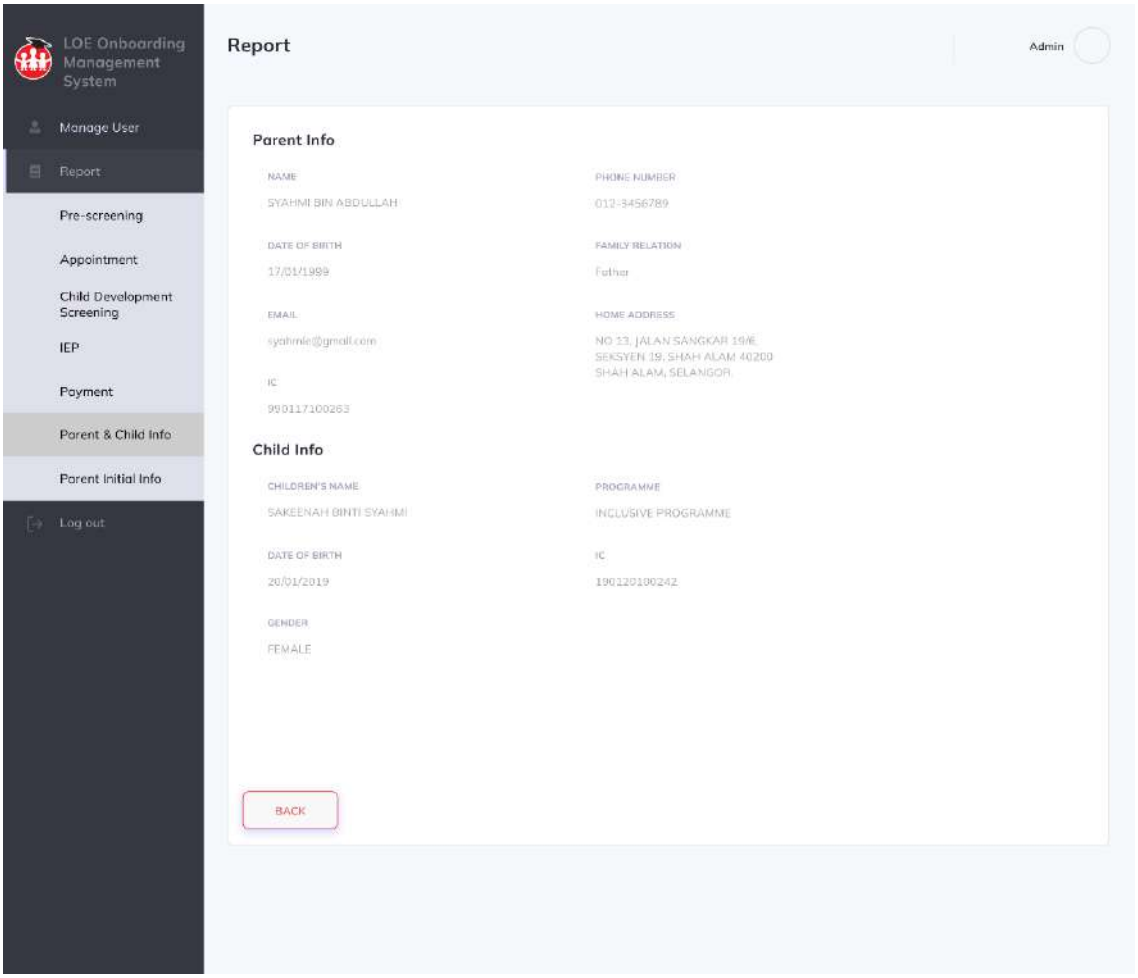
Appendix A. 46: Payment report list page Admin



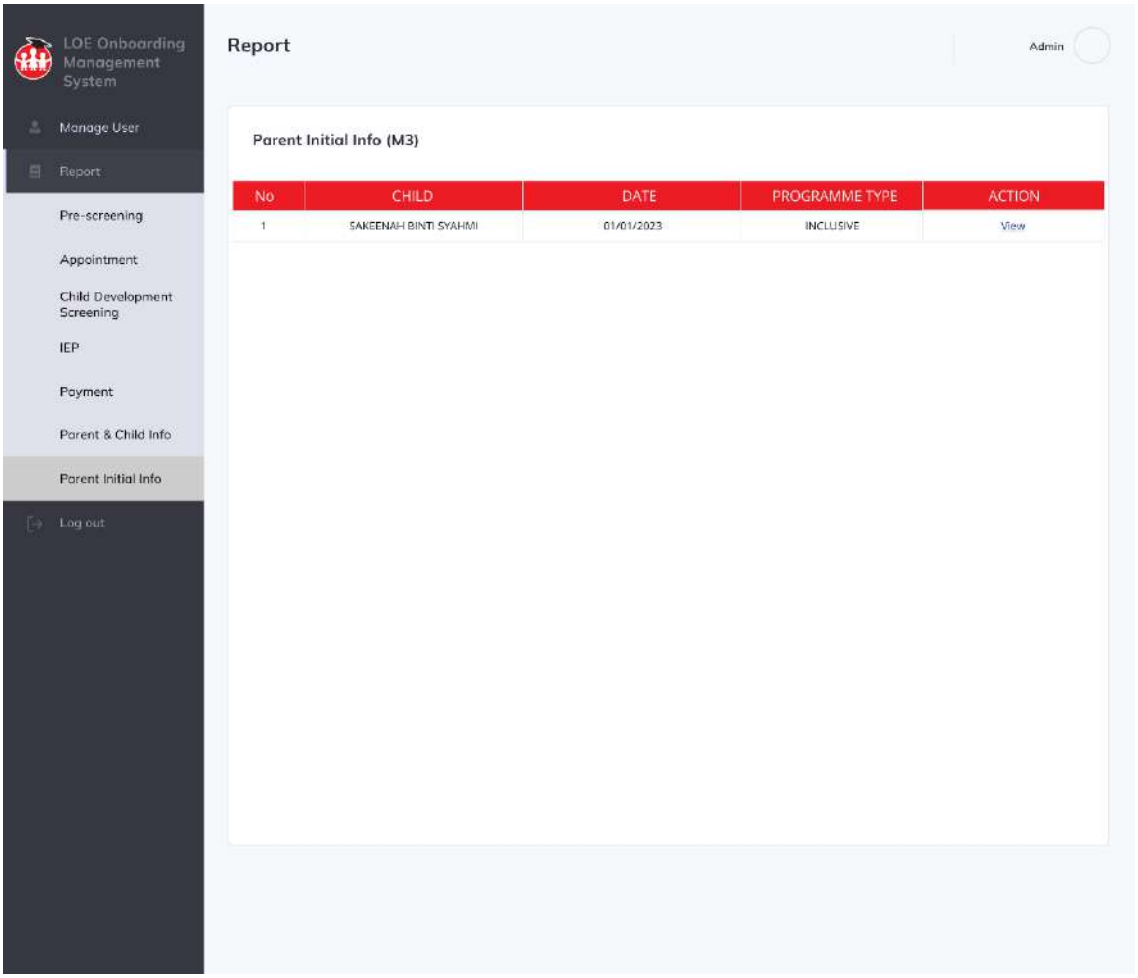
Appendix A. 47: Payment report details page Admin



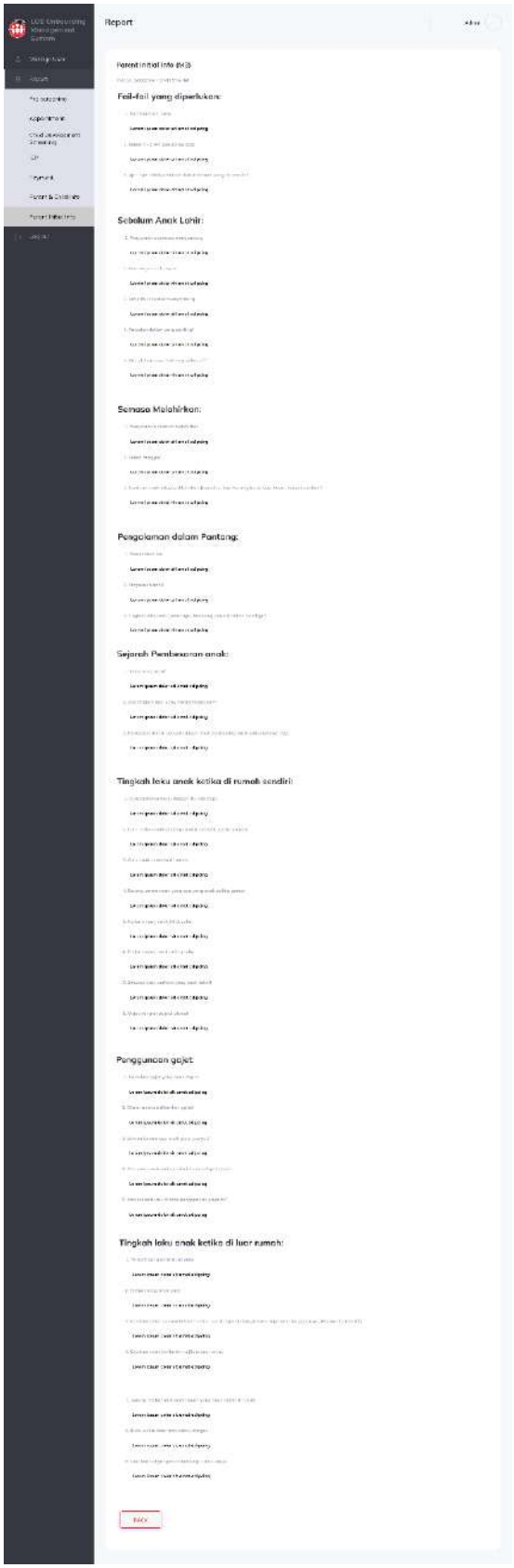
Appendix A. 48: Parent Child Info report list page Admin & LOE staff



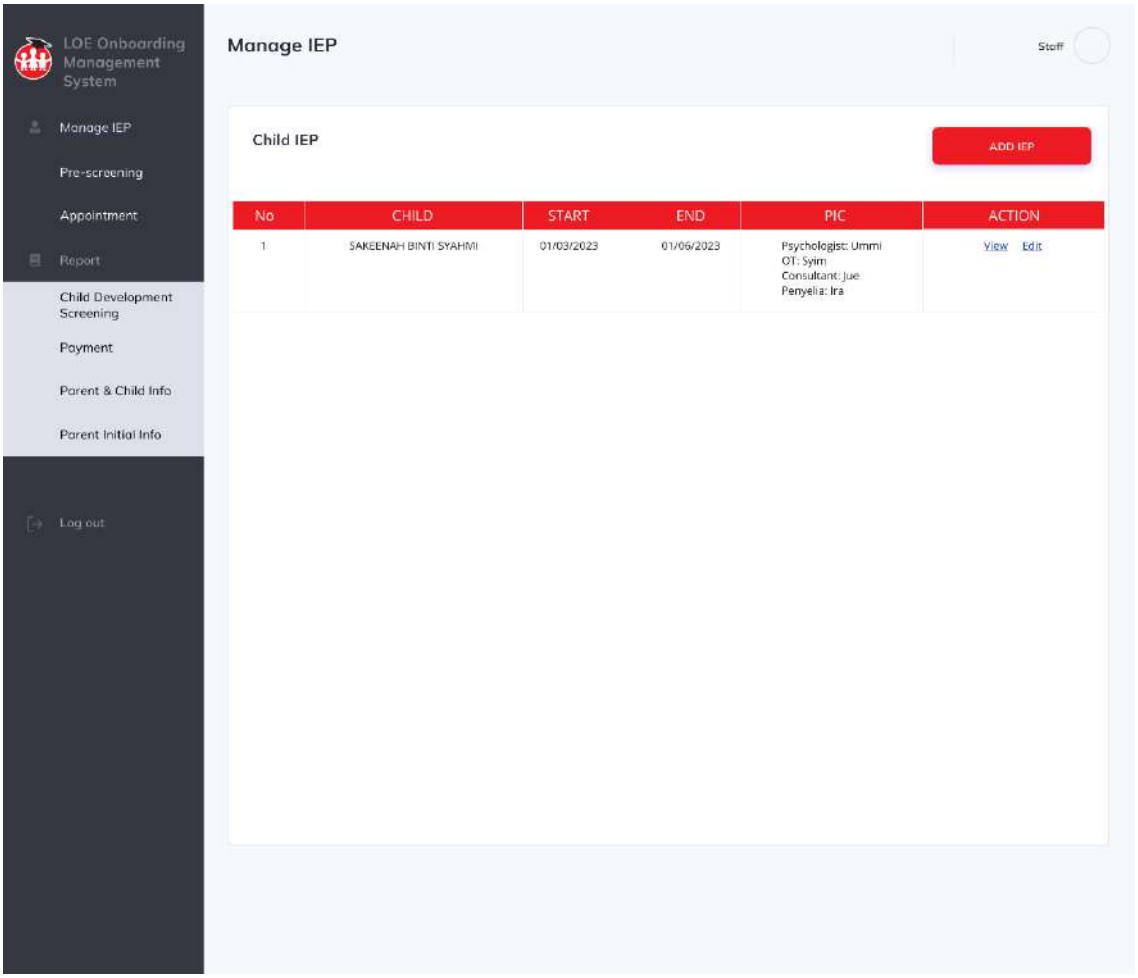
Appendix A. 49: Parent Child Info report details page Admin & LOE staff



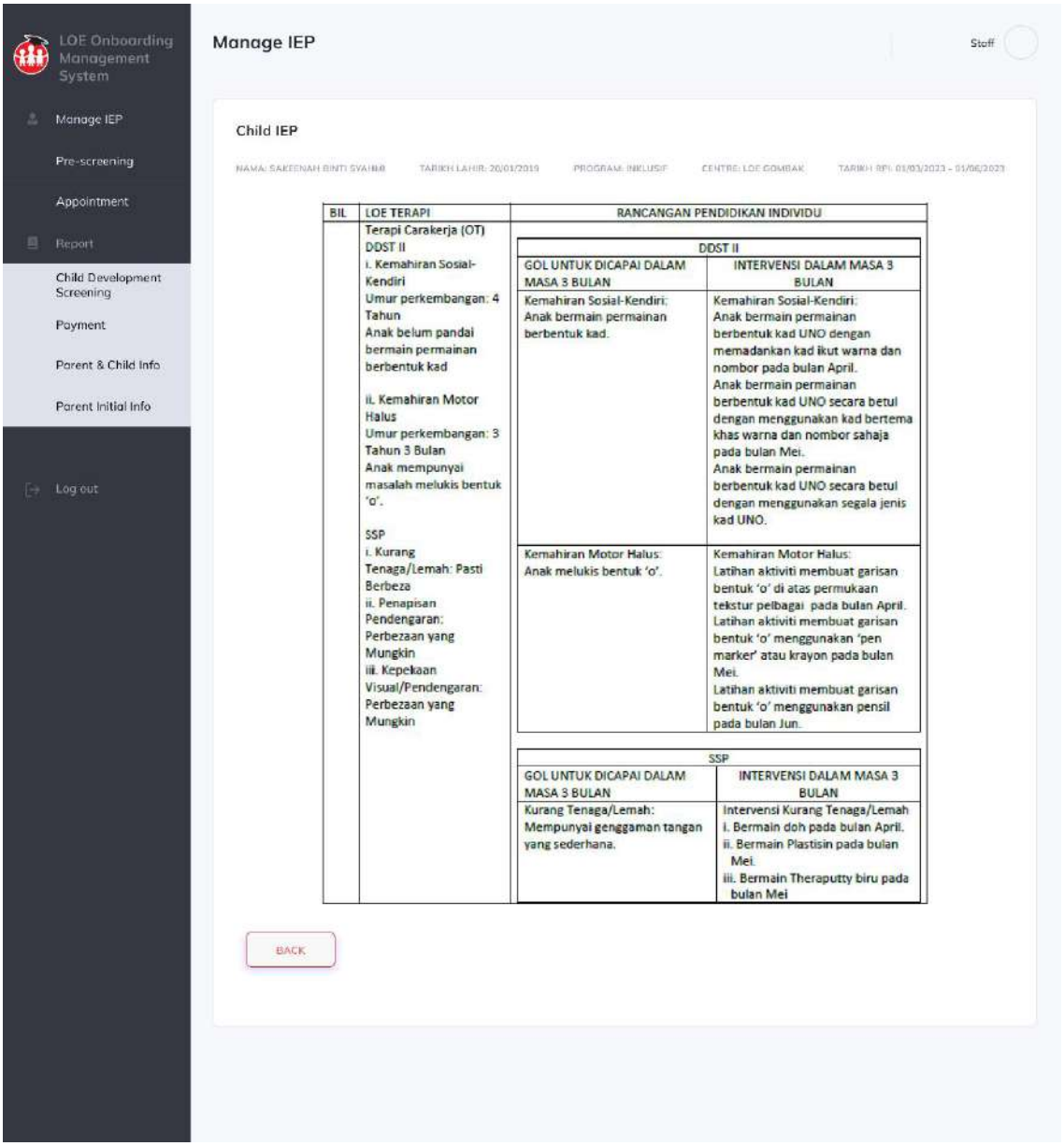
Appendix A. 50: Parent Initial Info report list page Admin & LOE staff



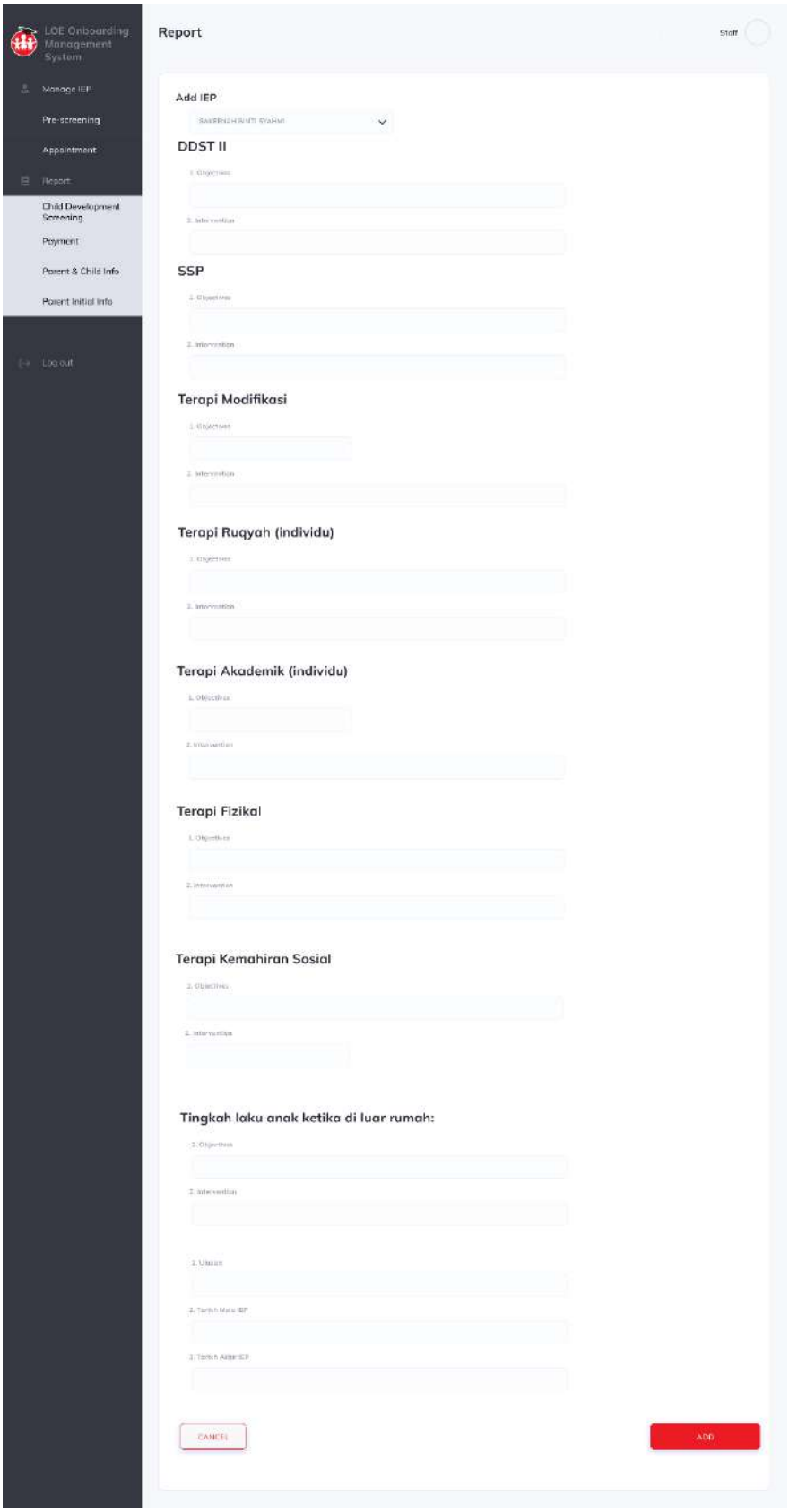
Appendix A. 51: Parent Initial Info report details page Admin & LOE staff



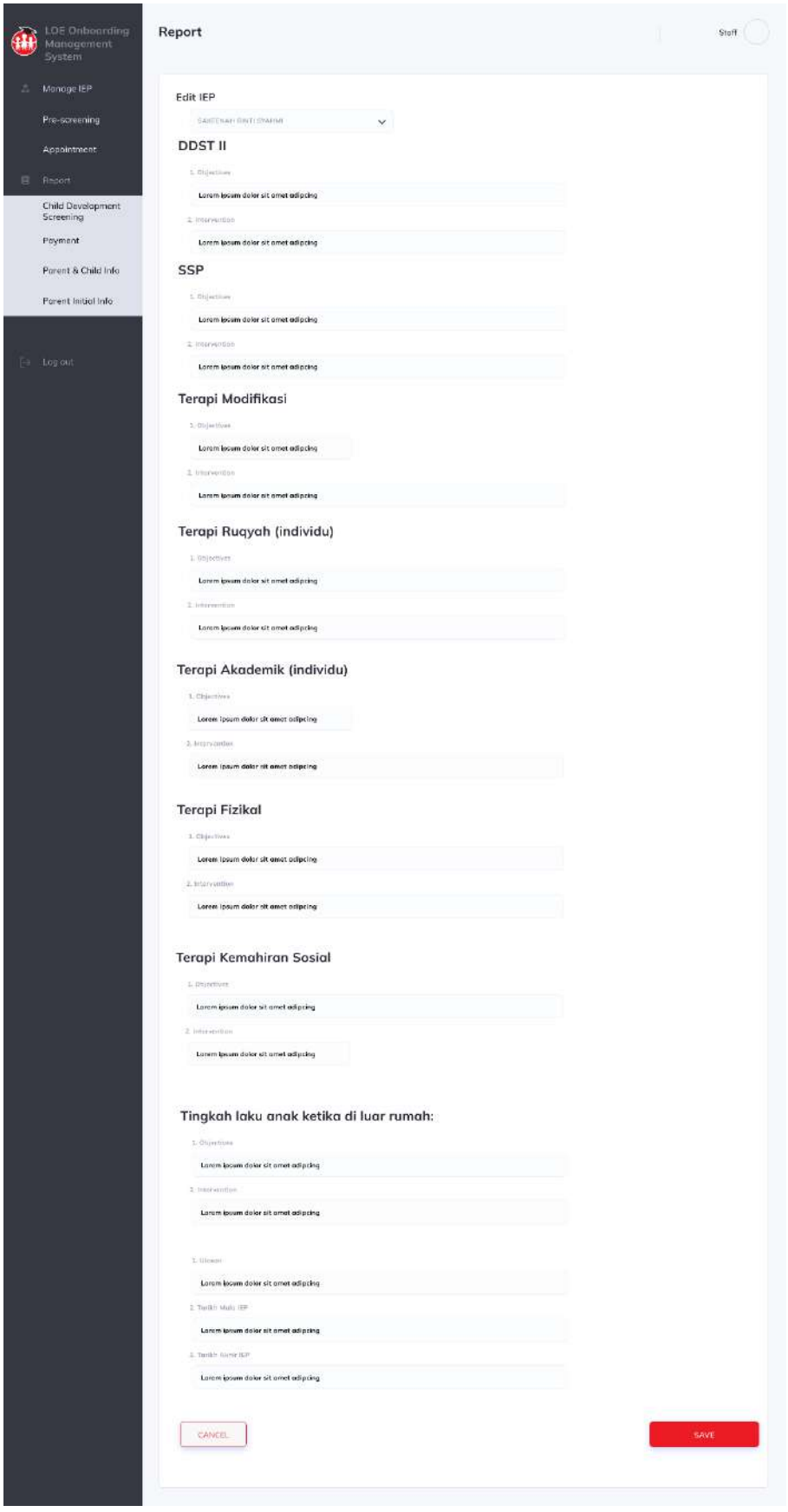
Appendix A. 52: Manage child IEP page



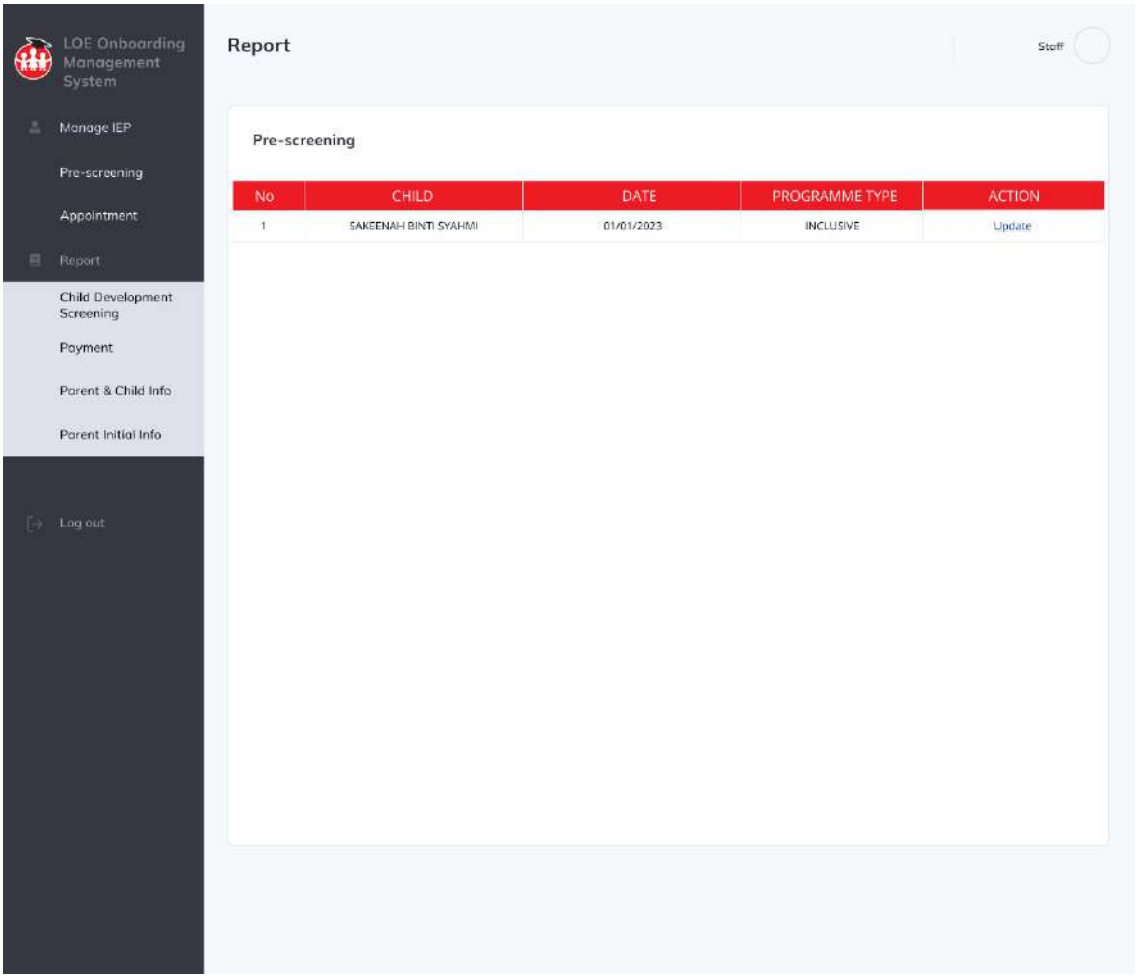
Appendix A. 53: View Child IEP details page



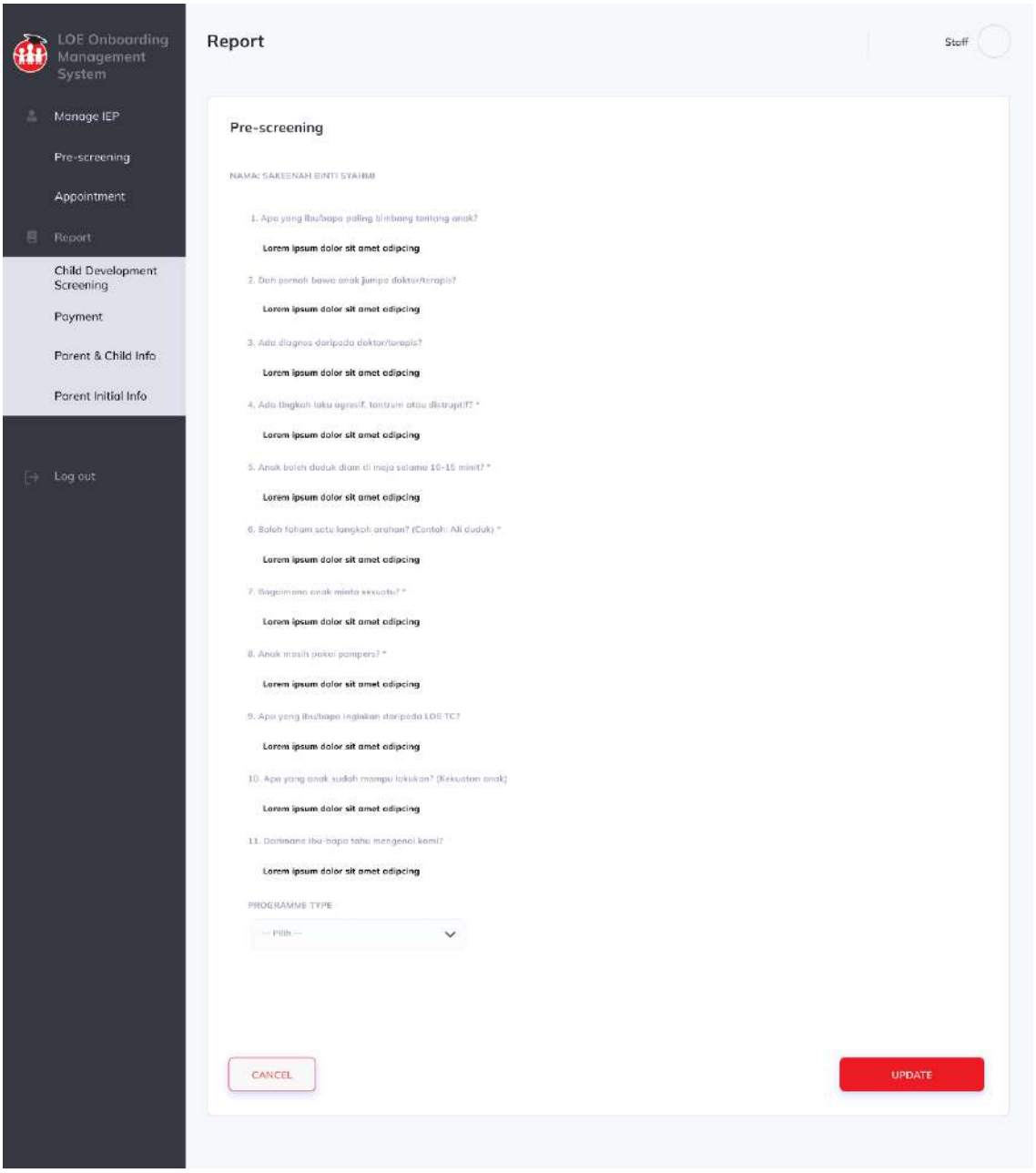
Appendix A. 54: Add child IEP page



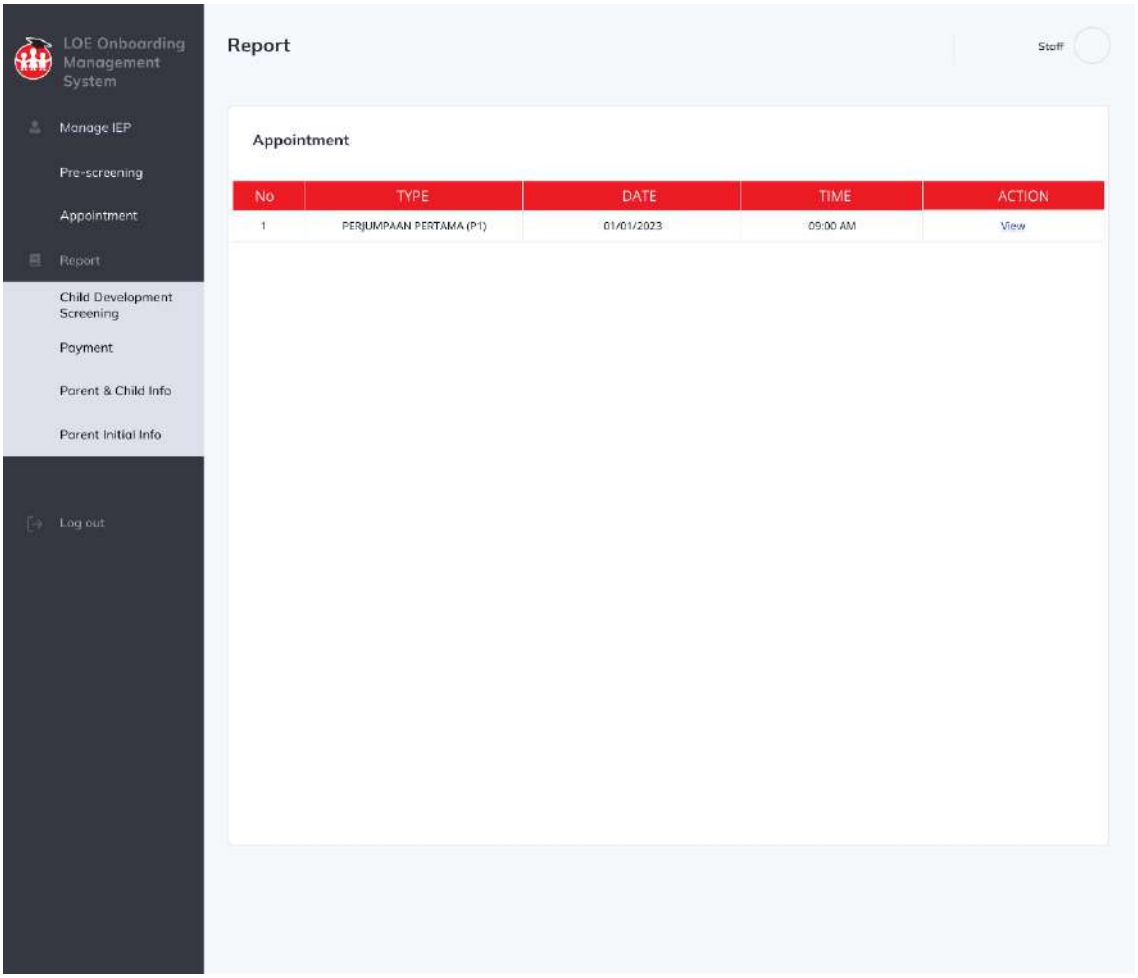
Appendix A. 55: Edit child IEP page



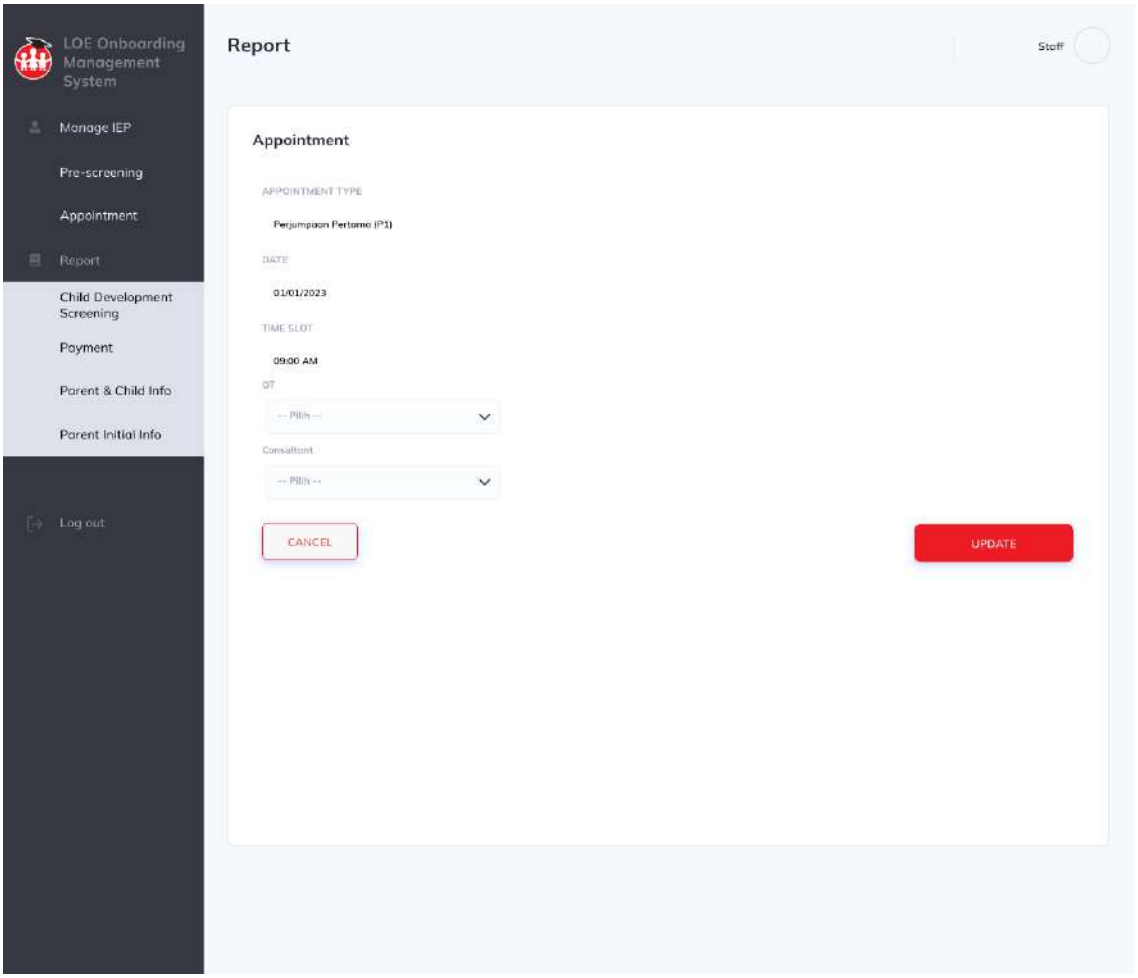
Appendix A. 56: Pre-screening list page



Appendix A. 57: Update programme type page



Appendix A. 58: Appointment list page LOE staff



Appendix A. 59: Assign Person in Charge for appointment page

APPENDIX B

No.	Module	Activities	Expected Result	Actual Result	Status		Comments
					Pass	Fail	
1	Register	Enter valid email format	Account successfully registered.	Account successfully registered.			OK
		Enter invalid email format	Error message displayed	Error message displayed			OK

		Click register button	Account successfully registered and receive email verification.	Account successfully registered and receive email verification.			OK
		Enter valid parent's name format	Account successfully registered.	Account successfully registered.			OK
		Enter invalid parent's name format	Error message displayed	Error message displayed			OK
		Enter different value in confirmation password input field with password input field	Error message displayed	Error message displayed			OK

		Enter same value in confirmation password input field with password input field	Account successfully registered.	Account successfully registered.			OK
		Enter valid IC number format	Account successfully registered.	Account successfully registered.			OK
		Enter invalid IC number format	Error message displayed	Error message displayed			OK. Error message displayed when user enter strings value. Need to display message box for duplicate input because IC number must be unique.
		Enter valid phone number format	Account successfully registered.	Account successfully registered.			OK

		Enter invalid phone number format	Error message displayed	Error message displayed			OK
		Enter input value for strings in any case	The input field will show uppercase only	The input field will accept uppercase only			It displayed in uppercase, but stored in lower case if the user key-in lower case alphabets.
2	Login	Enter valid email address	User successfully logged in.	User successfully logged in.			OK
		Enter invalid email address	Error message displayed	Error message displayed			OK
		Enter valid password	User successfully logged in.	User successfully logged in.			OK
		Enter invalid password	Error message displayed	Error message displayed			OK

		Login without an account	Error message displayed	Error message displayed			OK
3	Manage User System	Enter valid email format	Account successfully registered.	Account successfully registered.			OK
		Enter invalid email format	Error message displayed	Error message displayed			OK
		Enter valid user’s name format	Account successfully added.	Account successfully added.			OK
		Enter invalid user’s name format	Error message displayed	Error message displayed			OK
		Enter valid user’s IC format	Account successfully added.	Account successfully added.			OK

		Enter invalid user's IC format	Error message displayed	Error message displayed			OK
		Enter valid user's qualification format	Account successfully added.	Account successfully added.			OK
		Enter invalid user's qualification format	Error message displayed	Error message displayed			OK
		Enter valid phone number format	Account successfully registered.	Account successfully registered.			OK
		Enter invalid phone number format	Error message displayed	Error message displayed			OK
		Delete respective user	Correct user deleted and message displayed	Correct user deleted and message displayed			OK

4	Manage Account and Profile	Enter valid name format	Successfully edited and message box displayed	Successfully edited and message box displayed			OK
		Enter invalid name format	Error message displayed	Error message displayed			OK
		Enter valid IC number format	Successfully edited and message box displayed	Successfully edited and message box displayed			OK
		Enter invalid IC number format	Error message displayed	Error message displayed			OK. Error message displayed when user enter strings value. Need to display message box for duplicate input because IC number must be unique.

		Enter valid phone number format	Successfully edited and message box displayed	Successfully edited and message box displayed			OK
		Enter invalid phone number format	Error message displayed	Error message displayed			OK
		Enter different value in confirmation password input field with password input field	Error message displayed	Error message displayed			OK
		Enter same value in confirmation password input field with password input field	Successfully edited and message box displayed	Successfully edited and message box displayed			OK
		Enter valid email address format	Successfully edited and message box displayed	Successfully edited and message box displayed			OK

		Enter invalid email address format	Error message displayed	Error message displayed			OK
		Enter valid child name (Parent)	Successfully edited and message box displayed	Successfully edited and message box displayed			OK
		Enter invalid child name format (Parent)	Error message displayed	Error message displayed			OK
		Enter valid child IC format (Parent)	Successfully edited and message box displayed	Successfully edited and message box displayed			OK
		Enter invalid child IC format (Parent)	Error message displayed	Error message displayed. But when user mistakenly add not unique IC number, improper error message displayed.			OK. Error message displayed when user enter strings value. Need to display message box for duplicate input

							because IC number must be unique.
		Delete respective child (Parent)	Correct child deleted and message displayed	Correct child deleted and message displayed			OK
5	Manage Pre-Screening	Display respective child name	Answer successfully store with correct id	Answer successfully store with correct id			OK Add “Asterisk (*)” for required input fields
		Update LOE child programme (Manager, Principal, OT)	Successfully updated and message box displayed	Successfully updated and message box displayed			OK
		Notify user after click on Submit button	Data stored and User received notification				Need to display a message box for and send email notification.

6	Manage Payment	Display correct value of total payment	Successfully pay	Successfully pay			OK Remarks input value need to change to enter receipt number/upload bank receipt
		Notify user after click on Pay button	Data stored and User received notification				Need to display a message box for and send email notification.
7	Manage Appointment	Notify user after click on Book button	Data stored and User received notification				OK
		Notify user after click on Update button	Data updated and User received notification	Data updated and message box displayed			OK. Message box displayed. Need to send email notification.

		View appointment list (Parent)	Display correct child data under parent id	Display correct child data under parent id			OK
		View appointment list (Admin)	Display correct all child data that book appointment	Display correct all child data that book appointment			OK
		Cancel respective appointment (Admin)	Correct data deleted and message displayed	Correct data deleted and message displayed			OK
		Cancel respective appointment (Parent)	Correct data deleted and message displayed	Correct data deleted and message displayed			OK
		Display respective child name	Answer successfully store with correct id	Answer successfully store with correct id			OK

		Display respective PIC name (Manager, Principal and OT)	Data successfully store with correct id	Data successfully store with correct id			OK Able to add more PIC.
8	Manage Child IEP	Add IEP	Data added into database	-			OK
		Edit IEP	Data updated in database and message displayed	-			OK
		Delete IEP	Data deleted from database and message displayed	-			OK
9	Manage Report	View Pre-Screening (M1)	Data displayed	Data displayed			OK Add generate report to PDF

		View Child Development Screening	Data displayed	-			OK Add generate report to PDF
		View IEP	Data displayed	-			OK Add generate report to PDF
		View Appointment	Data displayed	Data displayed			OK Add generate report to PDF
		View Payment	Data displayed	Data displayed			OK Add generate report to PDF
		View Parent & Child Info	Data displayed	-			OK. Add generate report to PDF

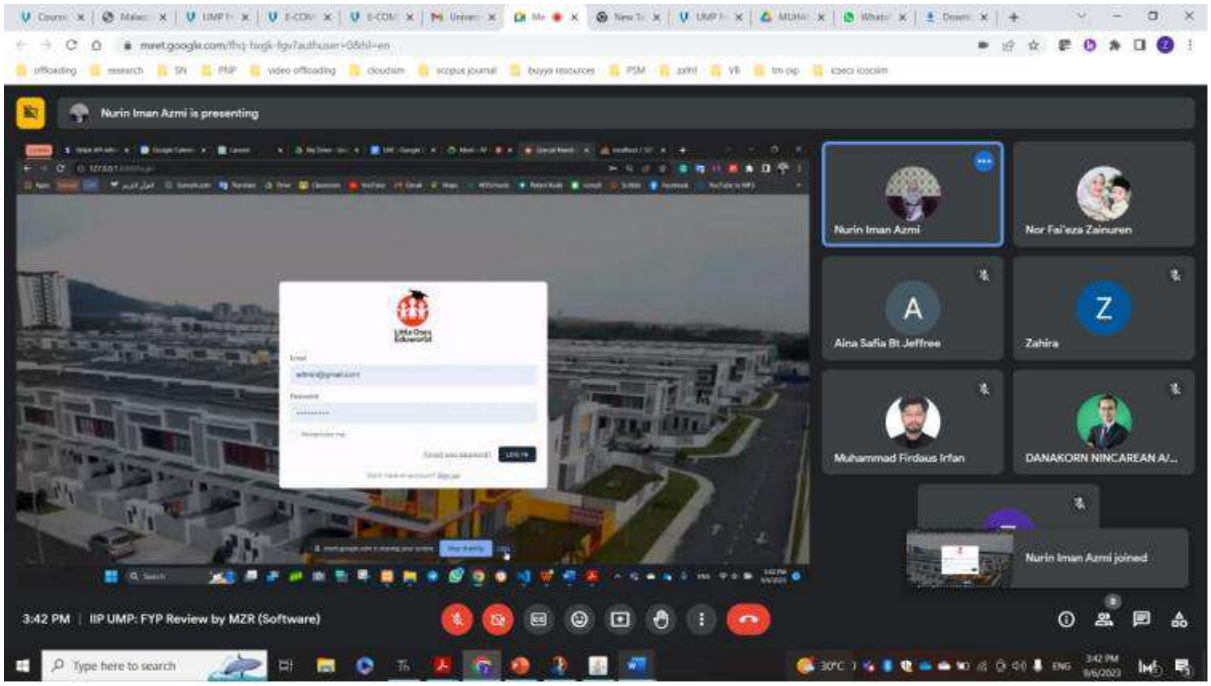
		View Parent Initial Info (M3)	Data displayed	-			OK Add generate report to PDF
10	Manage Parent Initial Info (M3)	Notify user after click on Submit button	Data stored and User received notification	-			Need to complete it as soon as possible
		Display respective child name	Answer successfully store with correct id	Answer successfully store with correct id			OK Add “Asterisk (*)” for required input fields
11	Manage Child Development (External system)	-	-	-	-	-	-

Appendix B. 1

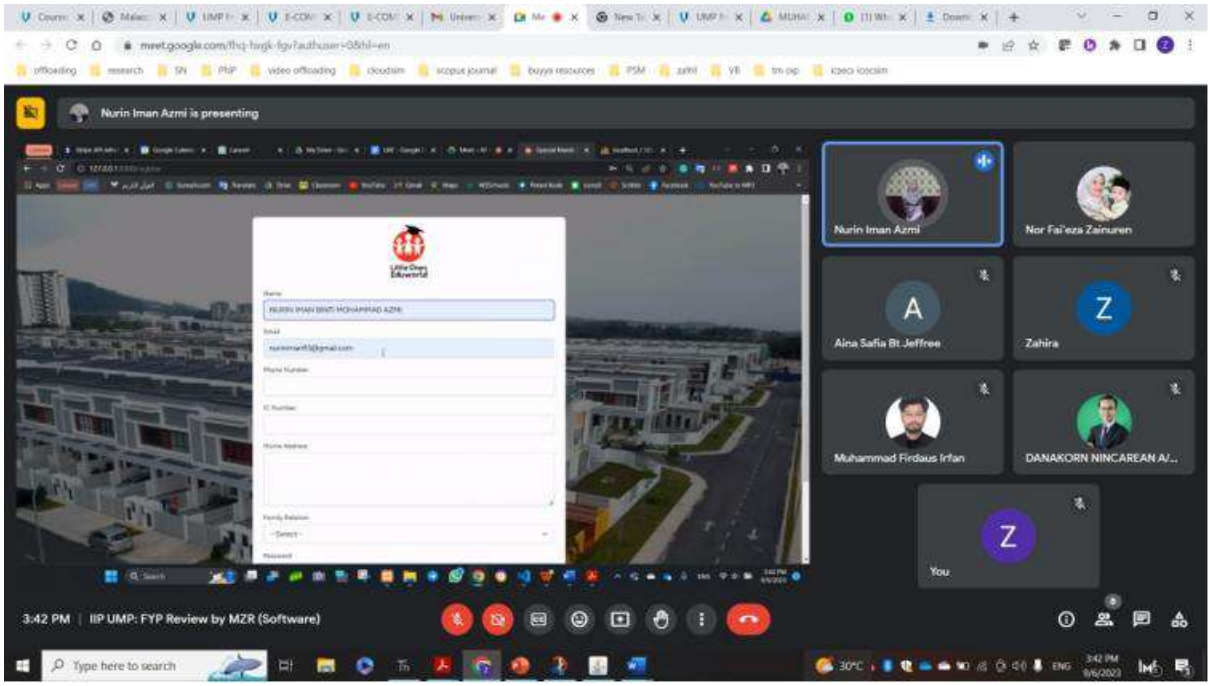
Verified by:

Name : Nor Fai'eza Binti Zainuren
Date : 09/06/2023

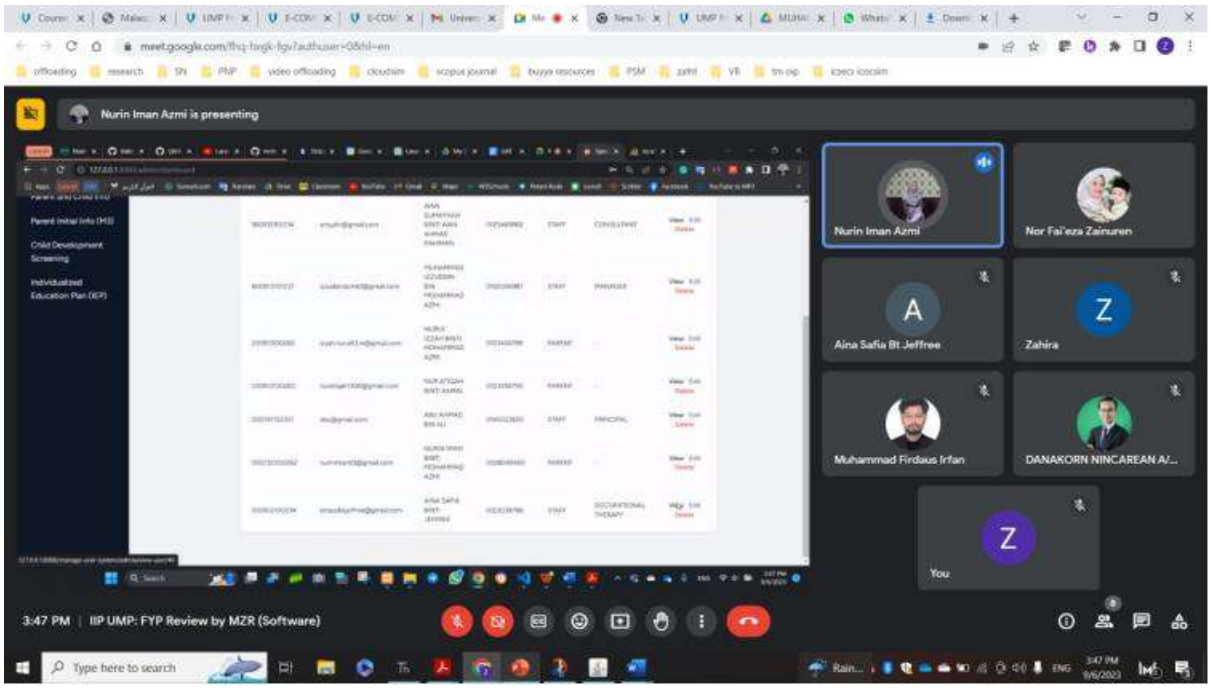
System Review with MZR Global Sdn.Bhd



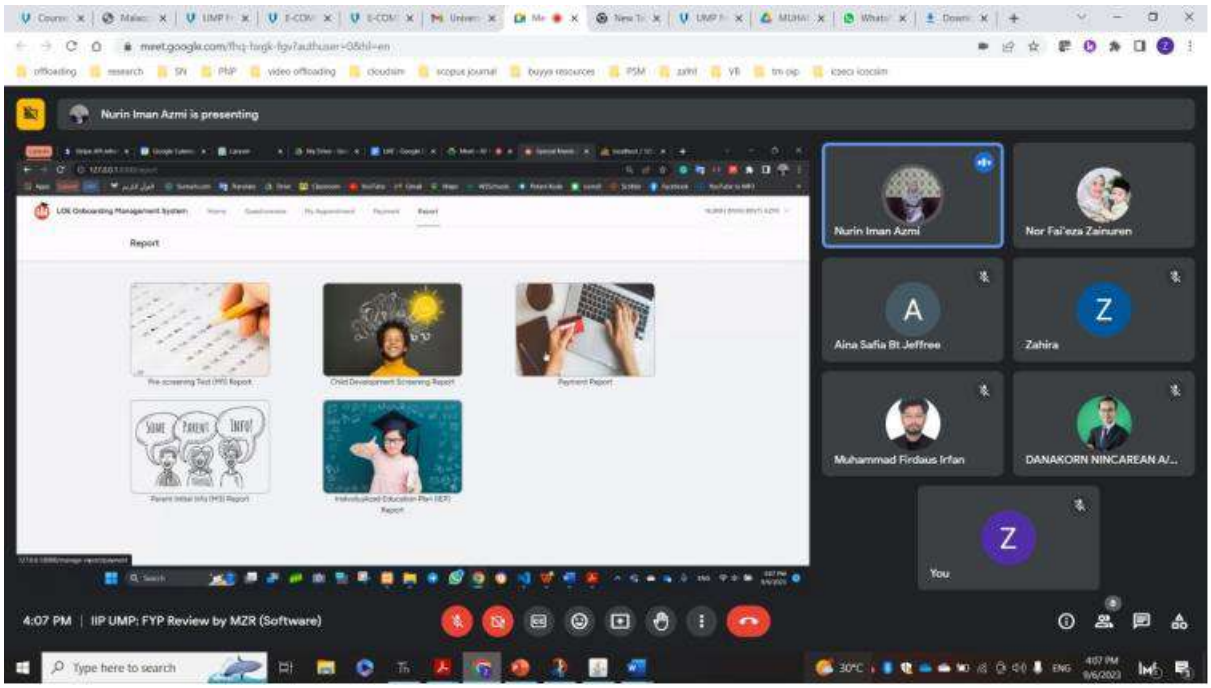
Appendix B. 2



Appendix B. 3



Appendix B. 4



Appendix B. 5