University E-voting System utilizing Blockchain Technology and Zero-Knowledge Proofs

LIM AUN XIAN

Bachelor of Computer Science (Computer Systems & Networking) with Honours

UNIVERSITI MALAYSIA PAHANG

UNIVERSITI MALAYSIA PAHANG

DECLARATION OF THESIS AND COPYRIGHT

Author's Full Name	: LIM AUN XIAN
Date of Birth	
Title	: UNIVERSITY E-VOTING SYSTEM UTILIZING BLOCKCHAIN TECHNOLOGY AND ZERO-KNOWLEDGE PROOFS
Academic Session	: 2021/2022

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)	(Supervisor's Signature)
	DR. ZAHIAN BINTI ISMAIL
_______________________	_______________________
New IC/Passport Number	Name of Supervisor
Date: 21 JUNE 2022	Date: 18 JANUARY 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.

Universiti Malaysia PAHANG
Engineering • Technology • Creativity

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 Bachelor of Computer Science (Computer Systems & Networking) with Honours.

[illegible]

____

(Supervisor’s Signature)

Full Name	: DR. ZAHIAN BINTI ISMAIL
Position	: SENIOR LECTURER
Date	: 18 JANUARI 2023

____

(Co-supervisor’s Signature)

Full Name	:
Position	:
Date	:

Universiti Malaysia PAHANG
Engineering • Technology • Creativity

STUDENT'S DECLARATION

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

[illegible]

____

(Student's Signature)

Full Name	: LIM AUN XIAN
ID Number	: CA19089
Date	: 21 JUNE 2022

University E-Voting System utilizing Blockchain Technology and Zero-Knowledge Proofs

LIM AUN XIAN

Thesis submitted in fulfillment of the requirements
for the award of the degree of
Bachelor of Computer Science (Computer Systems & Networking) with Honours

Faculty of Computing

UNIVERSITI MALAYSIA PAHANG

JUNE 2022

ACKNOWLEDGEMENTS

I would like to thank my supervisor Dr. Zahian binti Ismail for her guidance at every step of progress of the project. Her critiques and concise explanations have been extremely helpful in clearing the uncertainty and doubts encountered during the project. I too would like to extend my deepest gratitude to Dr. Danakorn Nincarean A/L Eh Phon, the coordinator of this final year project for session 2021/2022. Without his briefing and constant updates provided, this project would not have been as smooth sailing as it has been.

My grateful thanks to my parents for their support and encouragement during these trying times. Many thanks to University Malaysia Pahang for providing access to countless resources to aid my project.

Lastly, my special thanks to my friends throughout the duration of our degree together who has helped me and provided encouragement. This project would not have reached its completion without your input and advice.

ABSTRAK

Kemunculan dan popularisasi teknologi blockchain telah mengubah teknosfera pada dekad yang lalu, mempenambahbaikan sektor yang berlainan dengan implementasinya dalam infrastruktur sektor tersebut. Dalam pilihan raya majlis perwakilan pelajar Universiti Malaysia Pahang, pelajar-pelajar telah menimbulkan kebimbangan mengenai penghandalan maklumat peribadi kepada peranti yang berpotensi dikompromikan, integriti dan pengesahan keputusan pilihan raya tersebut dan keperluan untuk menghadiri tempat mengundi secara fizikal. Sistem e-voting blockchain menangani masalah dinyata dengan penggunaan internet untuk menjalankan pilihan raya, tidak menangani maklumat peribadi pengguna dan menerbitkan blockchain yang mengandungi setiap undi selepas pilihan raya untuk disahkan oleh semua. Dengan penggunaan rangka kerja DSDM dan FastAPI, sistem e-voting blockchain berasaskan web telah berjaya dihasilkan. Selepas menerima input pengguna melalui UAT, sistem e-voting blockchain telah diterima dengan baik oleh pelajar dan telah mendapat maklum balas positif mengenai kecekapan dan keberkesanannya, menunjukkan pengesahan keputusan dalam sistem yang menggunakan teknologi blockchain.

ABSTRACT

The emergence and popularization of blockchain technology has changed the technosphere in the past decade, improving different sectors with its application. In University Malaysia Pahang's student council elections, students have raised concerns with regards to providing personal information to potentially compromised devices, the integrity and verifiability of the election and the need to physically attend the voting venue. A blockchain e-voting system addresses these issues by allowing users to access the election using internet connection, does not deal with personal information of the user at all and the blockchain is published after the elections for all to verify. By employing the DSDM framework and the FastAPI python framework, a web-based blockchain voting application has been successfully developed. After receiving user input through UAT, the blockchain e-voting system has been well received by students and has gained positive feedback in regard to its efficiency and effectiveness, along with demonstrating the verifiability a system may provide with the use of blockchain.

TABLE OF CONTENT

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS

LIST OF ABBREVIATIONS

ZKP	Zero-Knowledge Proof
AES	Advanced Encryption Standard
DES	Data Encryption Standard
SHA	Secure Hash Algorithm

CHAPTER 1

INTRODUCTION

1.1 Overview

Voting has been used as a medium to elect leaders democratically throughout history. The technology behind voting has been evolving with the times, from the traditional paper voting system to the current electronic voting system. However, electronic voting systems does not instil confidence among voters due to underlying issues. Chapter 1 covers the origins of voting, the issues faced in current voting solutions, methods of resolving those issues, and drafts the objectives and scope of the project.

1.2 Background

Historically, the first occurrence of voting was recorded in Ancient Greece in the year 508 B.C., where male landowners would write the political candidates they most wanted to be exiled (for a period of 10 years) on broken pieces of pots. This was known as a “negative” election and would be the earliest form of democracy in society. Through many iterations and changes, traditional voting would become the paper system that we know today.

Traditional voting using a paper system operates as follows : Voters would choose their desired candidate by marking on a ballot inside polling stations. The ballots are pre-printed with the name of the candidates along with their memorandum. For voters that are unable to physically attend these polling stations, they can opt for mail-in ballots. Once the voting period ends, election officials manually count the ballots and recount the ballots if disputes occur.

With technological advancements, electronic voting (e-voting) has become a reality. In the 1964 USA presidential election, punched card voting systems debuted where votes were counted by machines, which progressed into optical scan voting systems where voter's choices are marked on pieces of paper and goes through a scanner to create a tally for the candidates. The optical scan voting systems are still used today, but has its own set of problems, ranging from paper jams to overheated sensors.

Next came the direct-recording electronic (DRE) system, where a touch screen displays the choices to the voter which allows them to change their choices before casting their vote. These machines provide accessibility to handicapped voters who would otherwise encounter issues with the paper voting systems. Currently, online voting is being tested in elections globally but has encountered security issues in every attempt. For example, an existing internet voting system called OmniBallot allows voters to access the system online but returns the votes in the form of PDF files, making it insecure and could potentially be altered by malicious parties without detection. Thus, upcoming online systems should improve upon the existing solutions and find new ways to improve upon the verifiability of the voting results.

Blockchain technology was first hinted upon in 1982, when cryptographer David Chaum proposed a blockchain-like solution for computer systems where the users does not necessarily trust one another. This concept is further built upon in 1991 by cryptographers Stuart Haber and Scott Stornetta, where the two envisioned creating a chronological chain of hashed data in order to timestamp digital documents for authenticity verification in their paper How to Timestamp a Digital Document published in the Journal of Cryptology. They then further improved upon their system with the help of mathematician Dave Bayer by incorporating Merkle Trees into the design, allowing for several digital documents certification to be included inside one block, increasing the system's efficiency.

In 2008, the blockchain concept was further built upon by a person/group of people going by the pseudonym Satoshi Nakamoto. improved the system by using a proof of work system called Hashcash, which requires computational work to be done in order to validate the block but can be verified efficiently afterwards. By using Hashcash, blocks could then be timestamped at the time of creation without the need of a trusted party and a difficult parameter could be added to the proof of work system to stabilize the rate of blocks mined/added to the chain. This design is then implemented by Nakamoto a year later as the basis of the cryptocurrency bitcoin, a decentralized digital currency which operates without administration or a central bank.

The core concept of blockchain involves a database that is maintained by a network of users, secured and validated through the use of cryptography. New information is added to the database in the form of ledgers stored in data containers called "blocks". When new information is to be added, a new block is made and added to a chain of previously created blocks. A unique ID called the hash is made for the block using the hash of the previous block along with the data stored within the current block using a cryptographic algorithm. This prevents tampering of data in previous blocks as the generated hash would be different if any data was altered.

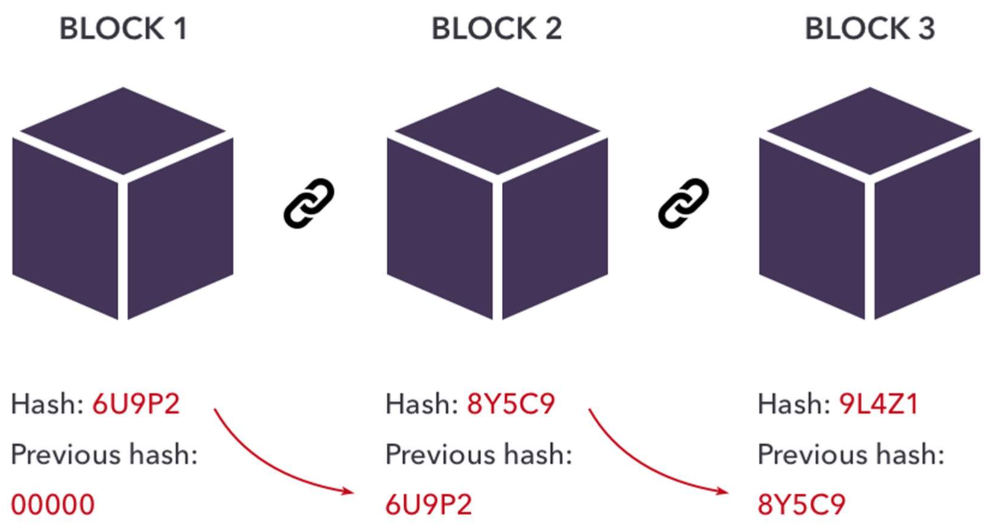


Figure 1.1 Blockchain Technology and its Hashing Function

Contrary to public perception, the applications of blockchain technology are not only limited to cryptocurrencies. In his article Applications of Blockchain Technology beyond Cryptocurrency (2018) published in the Annals of Emerging Technologies in Computing (AETiC), Mahdi H. Miraz talks about storing and verifying legal documents (deeds, certificates), healthcare data, IoT, the Cloud and so forth. One of the purposes of this project is to bring to attention the use of blockchain technology in voting systems.

Due to the sensitivity of voter's personal data in all aspects of digital life, data must be handled in a secure matter to prevent leaks and data breaches. During the 2017 AsiaWorld Expo, two laptops belonging to the Hong Kong's Registration and Electoral Office were stolen. Within those laptops were the personal information of all 3.78 million registered voters in Hong Kong, including their names, addresses, ID card numbers, mobile phone numbers and their respective registered geographical constituencies. While this was a tragic incident, this could've been prevented if access to the data was not made possible.

By employing zero knowledge proofs (ZKP), we could prevent the mentioned incident from ever occurring. ZKPs allows an individual (the prover) to prove to another individual (the verifier) that a given statement is true without disclosing any additional information other than the statement is indeed true through the use of cryptography. By applying ZKP, no exchange of personal data is necessary to prove their identity, mitigating the risk associated with mishandling of their personal data.

The ZK protocol would work as follows in a voting system: users (the prover) would provide their personal information to a trusted party (i.e., the student electoral commission, the government), who would then generate their unique ZKP. This ZKP is then provided to the voting system (the verifier), where the ZKP is ran through a mathematical algorithm where the statement is verified for soundness and completeness.

For the previous student council elections in University Malaysia Pahang, students were required to physically attend the voting venues to cast their votes into the provided e-voting systems after their identity is verified by volunteers present at the venue through their name and matrix ID. During the COVID-19 pandemic, this system has transitioned to submitting Google Forms as ballots for the election process. While the e-voting system works, it can be improved upon by implementing blockchain technology and the zero-knowledge protocol.

1.3 Problem Statement

1.3.1 Data Integrity

In the traditional paper voting system and electronic voting systems (with the exception of online voting), voters must make their way to their respective registered polling stations to cast their votes. Once their vote has been casted, the voter is no longer involved in the process and has no way to ensure that their votes are counted, and the integrity of the count is ensured. There has to be a security element to the election in order to maintain the integrity of the vote and confidence of the voters themselves.

For students of UMP, previous elections were conducted in halls where students attend physically and casted their paper ballots. Subsequently during the COVID-19 pandemic, the student council elections were moved online using Google Forms. However, the possibility remains that results in both methods could be altered without the knowledge of the voters. Online voting would also streamline the vote submission process, the vote tallying and result tabulation while the implementation of blockchain system would address the issue of data integrity.

1.3.2 Data Breach

As previously mentioned, data leaks with regards to the personal information of a nation's voters would have massive repercussions. Compromised data would mean possibly exposing confidential information to foreign parties. Identity theft, financial crimes, nationwide unrest would ensue. Thus, data must be handles in a secure manner, preferably not interacted with at all. In the case of UMP student council elections, the votes are conducted through Google Forms and if one of the accounts responsible for the election is compromised, the results and all other information are at the mercy of the malicious third parties. Information gathered could be used for identity fraud, personal email accounts could be taken over and used for phishing, having their data scraped and the owner possibly facing blackmail from the attackers

In order to prevent such incidents from occurring, zero-knowledge proof should be utilized in the voting system during its registration stage. Zero knowledge proofs allow one party to prove to another party that a given statement is true without disclosing additional information utilizing cryptography. In this scenario, the user can verify their identity to the voting system without the need to submit their personal information to the voting system.

1.3.3 Verification

Another point to mention is the likelihood of voter fraud. In the United States, politicians at all levels of government have repeatedly, and falsely, claimed that the elections conducted in 2016, 2018 and 2020 were marred by large numbers of people voting illegally. While these baseless accusations can be tossed around without proof and has been traced to false claims by the loser of a close race, mischief and administrative or voter error (Minnite, 2007), a method to verify and prove the results of the election is much needed to silence the naysayers.

With the implementation of blockchain technology in voting systems and deployment in actual elections, all casted votes would be stored on the blockchain, vote tallying would occur in real time as the polling progresses and recounts would no longer be needed. The results would be available for all to see and easily verifiable by using their zero-knowledge proof statement and checking the previous hash of all the blocks in the chain. This was previously not available in UMP elections as the results are never published for all to see and kept to the eyes of UMP student council members only.

This project aims to explore the possible applications of blockchain technology in voting systems and the benefits it provides to improving the electoral process by providing transparency and verifiability throughout the entire process.

1.4 Objectives

- To study the fundamentals of blockchain technology and its viability as a real-world solution for problems(e.g., voting), along with the use of zero-knowledge proofs.

- To develop a voting system that utilizes both blockchain technology along with zero knowledge proofs.

- To evaluate the integrity, security, and verifiability of the voting system with the implementation of blockchain technology and zero knowledge proofs, along with public perception towards these emerging technologies.

1.5 Scope

User

- University Malaysia Pahang students.
- University students in need for an electoral solution where all parties involved do not fully trust one another.

Time

- This project will be completed within the span of a year from early 2022 to early 2023.

Technology

- Blockchain technology will be applied in the project to mitigate the issues relating to verifiability and centralization.
- Proof of Work(PoW) blockchains will be used to reduce complexity in implementation.
- Zero knowledge proofs will be applied for user identity authentication.

Programming Language/IDE

- The test project will be written in HTML, CSS, JS and Python utilizing Microsoft Visual Studio, the FastAPI framework and Jinja2 Templates.

Purpose

- Explore the integrity, security, and verifiability of the voting system with the implementation of blockchain technology and zero knowledge proofs, along with public perception towards these emerging technologies.

1.6 Thesis Organization

This thesis is split into six chapters and has been organized in the following manner:

Chapter 1 briefly discusses the history of voting, the evolution of voting systems, the subsequent transition to electronic methods, along with a background of blockchain technology. This chapter also highlights the issues with current solutions, methods to improve upon said issues, objectives, and scope of the project.

Chapter 2 presents the extent of the research performed on the topic, the background, and all other related works. This chapter also brings to light the main features of the blockchain, guidelines and requirements for a functional voting system, the importance of trust in a voting system and previous implementations of the idea.

Chapter 3 converses the research methodology that was applied in this project. The framework, functional requirements and non-functional requirements will be highlighted in this chapter.

Chapter 4 implements the project and analyzes the results obtained from the project, raising discussions about the future of blockchain technology as a whole.

Chapter 5 concludes the project and produces the conclusion reached after the results obtained has been analyzed.

CHAPTER 2

LITERATURE REVIEW

2.1 Introduction

With elections representing a big part of our lives in regard to policies and administration, voting systems have evolved throughout the duration of its existence. E-voting has been the focus of this project and along with it, the implications of blockchain within the system. Chapter 2 explores the advancements in the field by researchers through their publications. The literature review brings focus to electronic voting, otherwise known as E-voting, and its online applications. Sources used are obtained from publications by the Institute of Electrical and Electronics Engineers. While the focus is online voting, previous work done on zero-knowledge proofs was also reviewed, along with implementation of blockchain on 3 previous executions of voting systems within the area of research that highlights the strengths and weaknesses of their respective implementations.

2.2 Requirements of an Electronic Voting System

Reviews of currently available literature is done to obtain the core requirements an electronic voting system must have before implementation is possible. Cetinkaya, O. and Cetinkaya, D. (2007) discusses the core requirements an electronic voting system should have in order to be an adequate solution to electoral needs:

> *Voter Privacy*: The voter's identity must remain private and disassociated with the vote itself. The privacy of the voter must be maintained during the election and post-election as well for the foreseeable future.
> *Eligibility*: Only registered voters should be allowed to cast their vote, with the registration itself occurring before the election.

Fairness: Partial tally of the election results is not to be disclosed before the end of the election to ensure that current tally does not influence voter decision. The electoral committee should not have any insight on the results of the ongoing election.
Uniqueness: A voter should only have the option of casting their vote once and the casted vote should only be counted once.
Uncoercibility: Any coercer (including authorities) should not be able to obtain the values of the vote and coerce voters to cast their vote in their favour. Voters should retain their freedom to vote for whoever.
Accuracy: All the casted votes should be counted. Any vote cannot be altered, copied, deleted, or invalidated. Any sort of attack on the votes should be detected. Uniqueness applies here as well.
Robustness: All parties involved in the election cannot disrupt or influence the final tally and the election itself. Robustness must be ensured to instil confidence in the results of the election.

However, Cetinkaya, O. (2008) further elaborates on these requirements in his next publication. In the next publication, he adds the following requirements to the initial list.

Receipt-freeness: The system would not provide a confirmation of the vote in the form of a receipt. This is to prevent voters from being able to prove their votes to a third party as a deterrent to the practice of vote buying/selling.
Individual Verifiability: The voter should be able to verify that his/her vote has been casted, counted, and tabulated in the final tally. This is proven more difficult in an electronic, remote setting compared to a traditional voting system, as the voter inserts the ballot himself/herself into the ballot box.

The importance of individual verifiability is also discussed. People would not easily trust the e-voting system unless they can personally verify that their votes are casted and counted properly. With this, it can be concluded that individual verifiability is important to raise public trust in the online voting system.

2.3 Types of E-Voting Mechanism

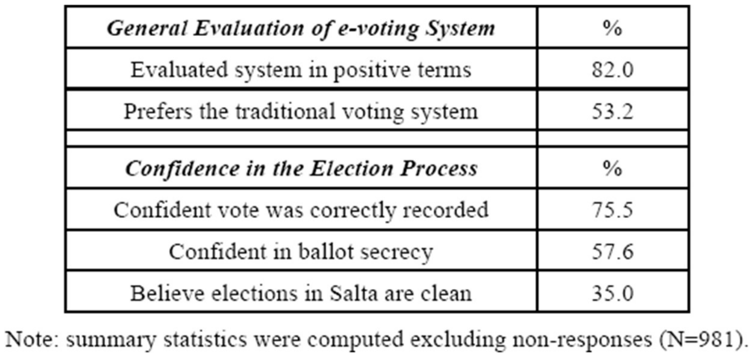
Alvi , S. T. et. al. (2020) explores the existing voting mechanisms and their drawbacks, starting from the Electronic Voting Machines System (EVMs). The EVMs were proposed to provide a less costly and quicker system of accountability, specificity, protection, falsifiability and voting integrity. However, the full process depends on a third party called the presiding officer, opening up the possibilities for vote tampering or insertion of false votes into the system.

E-voting, on the other hand, allows voters to vote from any polling stations for the given election. While voters can vote from anywhere with a voting machine, the system faces issues with regards to confidentiality, honesty, privacy and traceability. The system remains centralized and does not have safety features, making it unsafe to use.

Internet voting, or i-voting, involves multiple computers connected to one centralized server, which introduces a point of failure for the entire system along with the possibility of vote tampering. In conclusion, the author introduces the drawbacks of all voting mechanisms currently available and theoretically proposes a voting mechanism that will ensure protection of the electoral system, which is through the use of blockchain.

2.4 Public Response towards E-Voting

Pomares, J. et. al. (2014) examines the execution of e-voting in the district of Salta, Argentina and subsequently collects the publics experience and confidence towards the voting system/process. Among the key findings is the link between a voter's ability to use the voting machine without assistance and the resulting support for e-voting and positive perceptions of integrity of the election process itself.



General Evaluation of e-voting System	%
Evaluated system in positive terms	82.0
Prefers the traditional voting system	53.2
Confidence in the Election Process	%
Confident vote was correctly recorded	75.5
Confident in ballot secrecy	57.6
Believe elections in Salta are clean	35.0

Note: summary statistics were computed excluding non-responses (N=981).

Figure 2.1 Reception of E-Voting System

In general, it can be seen that the voters received the change in voting systems positively, with only 53.2% of the respondents stating they prefer the traditional system, proving that public sentiment is changing. Based on the election in Salta and the publication, we can conclude that voter confidence is associated with the usability of the voting system.

2.5 Features of Blockchain Technology

Al-Maaitah, S. (2021) discusses the possible opportunity for applying blockchain technology in e-voting systems along with its main features. These are the main features of blockchain technology:

> *Decentralization*: All nodes within the system share the same information, the digital ledger. The nodes handle the transactions and records of the digital ledger without a central authoritative figure.
> *Immutable*: The transactions stored in the ledger of the blockchain cannot be altered easily. Once transactions are committed and pushed on chain, it will not be changed unless all nodes approve of the change, which is near impossible.
> *Distributed*: All the nodes in the network share the same copy of the digital ledger.
> *Open Source*: Anyone can go through the source code to understand and improve upon the existing version by patching bugs and exploits.
> *Secure*: Blockchain technology is secure as it utilizes hash functions, encryption, and decryption algorithms.

The author presents the motivation for implementing blockchain technology in e-voting systems. Traditionally, the database is maintained by a central authority which has complete control over the database. Data can be manipulated and tampered with by the authority if they have a motive to do so. In cases involving financial matters or sensitive data (i.e., votes), it is not wise to give total control of the database to a single authority (Patidar, K., 2019).

To avoid such situations, blockchain technology would be utilized. With blockchain technology, the database is made public, and anyone can store an individual copy of the database to check for any manipulations. The individual copies are to be constantly updated to maintain consistency and to maintain the consistency of the decentralized database, the blockchain will utilized the consensus mechanism. In conclusion, the implementation of blockchain technology within voting systems can overcome the limitations of centralized voting systems.

2.6 Motivations to implement Blockchain Technology

Kamran, M. (2021) discusses some of the most promising designs for a blockchain voting system. Blockchain will be used as a means for casting votes so that the votes remain secure and transparent, while the identity of the voters remain secret. This is under the assumption that the electoral system is correctly set up and the conduction of the election is done fairly by the third party. While at least 50% of the nodes remain honest, the blockchain is secure.

The results are published on a government run website and the blockchain can be explored using a blockchain explorer. This allows voters to verify their own vote and view the total number of votes a candidate obtained. While sound suggestions are raised in this publication, allowing miners to decrypt user votes could possibly pose a security risk.

2.7 Zero Knowledge Proofs

In his publication, Ben-Sasson, E. (2015) showcases a protocol tailored to efficiently support non-interactive zero-knowledge proofs (NIZKs) in order to generate short and easy-to-verify proofs. With previous NIZKs, a party is trusted to i) correctly run a probabilistic algorithm to generate and ii) publish some public parameters to be used by provers and verifiers, without leaking any other information (i.e., the internal randomness employed by the algorithm). By letting a multi-party protocol generate the parameters, such that at least one of the parties is honest, the result is secure and can then be used to generate and verify numerous proofs without any further trust. By employing zero-knowledge proofs, user identity can be authenticated without the exchange of personal information.

2.8 E-Voting in University Setting

Elections of a smaller scale similar to the objective of this project has been performed before, based on a publication by Thiga, M. (2020). In his publication, student elections at Kabarak University in 2018 had low voter turnout and had reported cases of fraud due to human intervention in the voter verification and issuance of login tokens for the system. With this problem, they sought to implement an online election system to increase student turnout and improve system security. The system implemented for their online voting requires the university students to sign up with their personal information, request for a password from the system which will be sent to their provided phone number, then cast their vote.

With the implementation of the new voting system at Kabarak University, student election turnouts the following year increased slightly, of which the author associated to the availability of the voting system online. While the proposed system successfully fulfilled its purpose, it required voters to input their personal information which will be saved in the university's database and voters have to use their university credentials to login to the voting system and cast their vote. The system itself is also centralized and could be susceptible to tampering of the results. While the project provides a great guideline to follow, some aspects have to be improved upon, especially the handling of voter's personal information and the verifiability of the results.

2.9 E-Voting System with implementation on the Ethereum Blockchain

Rewatkar, H. R. et. al. (2021) proposes a model of an electronic voting system using blockchain. The proposed system is to be built on the local blockchain in order to conduct secure and efficient elections for the population with minimal efforts. In the proposed system, voters are issued coins by the government authority for vote casting. This will occur before the election and will not be part of the digital ledger of votes. The voter will then be issued a public and private key, of which the private key is used to access a wallet with the coin. Thus, only those with access to wallets with the coin can vote, verifying that they are real voters.

The proposed model by the author utilizes Ganache, where the truffle framework is used for testing smart contracts and sends them to the blockchain. The wallet for storing the coins will be Metamask and the Ethereum blockchain will be used, as it serves as a platform to host decentralized applications. While the proposal serves as great framework for a blockchain based e-voting system, the use of the Ethereum blockchain is unpractical due to its congested traffic which results in high transaction fees. Another blockchain with low transaction fees should be considered instead.

2.10 University E-Voting with implementation of Blockchain Technology

Singh, A. (2018) attempts to fulfill the security requirements for electronic systems through his design of a more secure and robust electronic voting system for university elections. Through his design, the model would not only conduct the voting procedure, but also provide security against major attacks. With his implementation, the university campus is split into four zones: east, west, north and south zone. Within the zones are multiple colleges, and votes collected in one college would create one block. The blocks of each college in each zone are then chained together to form the blockchain for the zone, a zone level blockchain. Zone level blockchains are then linked together to form the university level blockchain, the complete blockchain.

Through his implementation, the university level blockchain is stored in the voting server. The blocks are secure against different attacks and threats during data transmission. Privacy of the transmitted data (votes) is assured as the data is in hashed and encrypted form. Voter confidentiality is protected through the use of SHA-256 hash algorithm and encryption algorithm. The blocks are tempered so the attacker will not know the vote.

By implementing blockchain technology, forgery and duplication cases during the vote are removed. Unique voter ID is used for identification in the voting process. The blocks within the blockchain contains the hash of the previous block, signature and merkle root hash. The signature is used to prove the authenticity and integrity of the data, the hash of the previous block is included in the current block to maintain data integrity by proving the previous block has not been tampered. The merkle root hash tells the origin of the voter data. The author then concludes that all possible threats and attacks are mitigated with the implementation of electronic voting system in accordance with his proposal. While the author establishes an excellent use of blockchain and its technology, the requirement for voters to be verified after login by election organizers could be avoided entirely with the use of zero knowledge proofs.

Based on the publications of multiple authors, a general idea of the basis of the blockchain electronic voting system can be formed along with the requirements the system must fulfill. We will be applying the knowledge obtained from the review of these publications in the implementation of our blockchain voting system.

2.11 Comparison of 3-Existing Systems

	Publication	**Features**	**Advantages**	**Disadvantages**
Thiga, M. (2020)	Increasing Participation and Security in Student Elections through Online Voting: The Case of Kabarak University	• Voter login • Voter registration • Vote casting • Web interface for desktop and mobile access	• Easy to implement • Protected by firewall • Reduced election costs • Efficient tallying of votes	• Does not employ blockchain technology • Centralized thus resulting in lack of complete trust • Reliant on email and SMS for verification • Votes database unable to be audited
Rewatkar, H. R. et. al. (2021)	Decentralized Voting Application Using Blockchain	• Voter registration • Voter verification • Immutable votes • Privacy of votes • Scalable	• Uses blockchain technology in voting process, • Provably fair, • Instant results tallying • Decentralized vote recording	• Not in suitable context (university scale), • Utilizes the Ethereum network (Expensive gas fees) • Overly complex for the less technologically proficient • Additional database required to store accounts and user's public and private keys.
Singh, A. (2018)	SecEVS : Secure	• Voter registration	• Uses blockchain	• Not in suitable context

	Electronic Voting System Using Blockchain Technology	• Voter login • Duplication and forgery resistant, • Attack resistant, Low storage consumption	technology in voting process • Employs SHA-256 for the hash algorithm • Utilizes AES and DES as the encryption algorithm • Protects privacy of the data • Instant results tally • Publicly verifiable	(university scale), • Does not employ zero-knowledge proofs • Involves parties of authority, not fully decentralized • Requires the storage of voter ID and password • Splitting of zone level and university level blockchains unnecessary

Table 2.1 Comparison for Three Existing Systems

2.12 Summary

In this chapter, the characteristics of the electronic voting system blockchain is explored, along with other features to be implemented like zero-knowledge proofs. Three existing systems are compared, and their respective system features reviewed and taken into consideration for use in the project. The advantages and disadvantages will be considered in the development to improve upon the system's operation and avoid any potential drawbacks.

CHAPTER 3

METHODOLOGY

3.1 Introduction

In the words of the French writer Antoine de Saint-Exupéry, “A goal without a plan is just a wish”. Thus, a plan must be established before the project can proceed. With that said, a plan starts with the framework to implement the project. The framework is a certain approach to project management and software development to improve efficiency. Chapter 3 discusses the methodology used, requirements the system must fulfil, and the framework utilized.

3.2 Framework

The framework chosen for the deployment of this project is the Dynamic Systems Development Method (DSDM). Developed based on the Rapid Application Development (RAD) approach in 1994 due to the demand of project managers seeking more governance and discipline, the method improves upon it by emphasizing the quality of the quality and schedule of the releases. The system functionality is prioritized based on the MoSCoW approach:

- The must have;
- The should have;
- The could have; and
- The won’t have;

DSDM clearly defines the roles and responsibilities of all parties involved and employs techniques that help the project team deliver the required tasks effectively and efficiently. The DSDM framework adheres with the following eight principles to direct the team towards the mindset they should adopt to deliver on their tasks.

- Focus on the business need
- Deliver on time
- Collaborate
- Never compromise quality
- Build incrementally from firm foundations
- Develop iteratively
- Communicate continuously and clearly
- Demonstrate control

Traditional models of project management fix the features needed in the system, while making the time and cost for the project subject to variation. Resources are added if necessary (increases the cost) and the delivery date is delayed if needed (extends the time). However, adding resources to an otherwise late project could further complicate things and could delay deployment even further, while missing the deadline for a project reflects poorly on the company's reputation and competence. With the fear of missing deadlines and increasing costs of the project, quality often takes a hit by taking compromises in aspects such as quality control steps or cutting back on testing.

Unlike traditional models, the DSDM framework takes a different approach by varying the features delivered and maintaining the cost and time for the project. With this approach, projects can be delivered within the set budget and timeframe given by dropping lower priority features/requirements if necessary (with full agreement of the business stakeholders in accordance with the MoSCoW priorities).

The DSDM process works as follows: The one-time phase consists of feature analysis and model development and requirement/feature prioritization and planning, followed by the iterative phase of designing and building iterations. The project is completed at the delivery/implementation phase, where the project is applied into an operational environment.

Feature Analysis and Model Development

In this step, the problem is defined properly, and the technical feasibility of the project is verified to be achievable. Necessities and constraints are outlined, and the project is assessed on whether DSDM framework is applicable. Once the plan for the project has been clearly defined, work is started to build the prototype iteratively and getting it tested by end users to bring out the requirements of the desired system. This is done until a functional model is achieved. Based on the needs for a voting system and the necessary information involved, the blockchain can then be developed first that stores the pre-determined information.

Requirement/Feature Prioritization and Planning

Based on user feedback from the prototype, all features/requirements in the project are given a priority rating and the timeline for the development is mapped out. The priority rating is used to plan which features should be completed first before focus can be moved towards to lower priority features. In the worst-case scenario, lower priority features are dropped as previously mentioned. As a voting system, the voting aspect is the core of the project and has to be completed before quality of life features are added in later iterations of the project.

Design and Build Iterations

This phase ensures the final delivery of the project satisfies the core requirements defined and properly engineered to suit the operational environment. The software components designed in the functional model are refined until a satisfactory standard for deployment in real time environments is achieved. This step of the process is iterative to constantly improve upon the previous build of the project.

Delivery/Implementation

This is the last and final stage of the process. In this stage, users are trained, and the system is put to the test in a real time environment. For this project, a User Acceptance Test (UAT) is prepared and given to users in order to obtain feedback and improve upon the system.

A diagram of the process of the DSDM in context of the development process can be seen below.

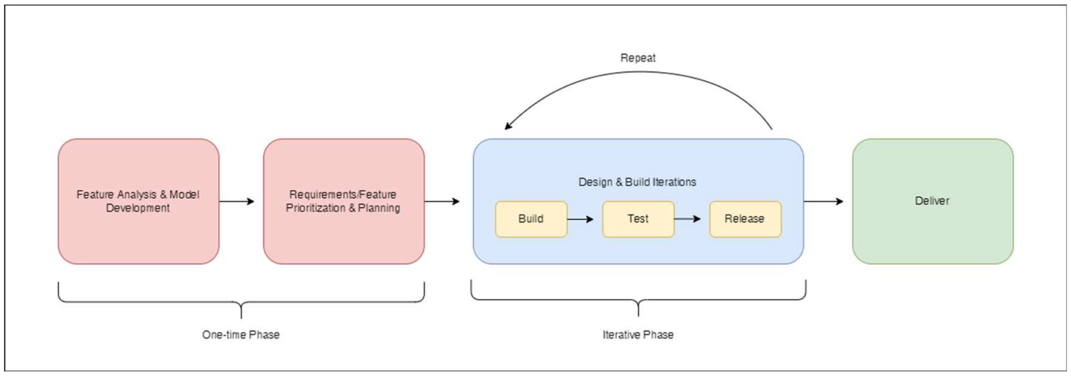

Repeat
Feature Analysis & Model Development
Requirements/Feature Prioritization & Planning
Design & Build Iterations
Build
Test
Release
Deliver
One-time Phase
Iterative Phase

Figure 3.1 DSDM Framework

3.3 Requirements

Functional Requirements

- Users are able to vote for their desired candidates using the system.
- The vote data has to be stored on the blockchain.
- Vote count should be tallied by the system at the end of an election.
- Users should not have to submit their personal information to the system.

Non-Functional Requirements

- Security
  The data on the system must have their integrity maintained and tamper-proof.
- Accessibility

Users with an internet connection should be able to interact with the system, no matter the location.

- Availability
  The system shall be available during a planned election duration and must maintain a 99% up time during that duration.
- Scalability
  The system should be able to accommodate the increase in voters and nodes without any issue.
- Maintainability
  The code for the project must be written in accordance with a set coding standard in order to simplify maintenance and onboarding of other coders.

Constraints/Limitations

For this project, the blockchain voting system will only be operated at a small scale due to financial limitations and the required processing power to run at a larger scale. 2/3 nodes will be deployed using virtual machines for the voting blockchain. With the scope of the project encompassing only an election at the university-level, the additional nodes are not necessary but would be a welcome addition.

This project is limited to the implementation of a Proof of Work (PoW) system. Proof of Stake (PoS) systems have an added layer of complexity in the issuance of cryptocurrency to trusted parties to verify transactions, The population that evaluates the efficiency and effectiveness of the voting system is limited to students at University Malaysia Pahang of age ranging from the early twenties to mid-twenties, hailing from different ethnicities, religions, and financial backgrounds. This is due to the fact that all parties involved in university elections are university students. This would mean that the results of the survey done on the efficiency and effectiveness of the blockchain voting system can only represent the views and opinions of young adults at best.

3.4 Use Case Diagram

A use case diagram is used to summarize how all involved actors interact with the system. For this project, there are 7 use cases and 3 actors involved: the voter, the student electoral commission and the administrator for the blockchain voting system. The voters could register as a voter, submit their zero-knowledge proof (after being registered) to be verified, then submit their vote. The student electoral commission only handles the issuing of zero-knowledge proofs after verifying the voter's identity. The administrator adds voting candidates to the system, issues the command to tally the votes after the election and publishes the votes. All actors can view and verify transactions on the blockchain after the election.

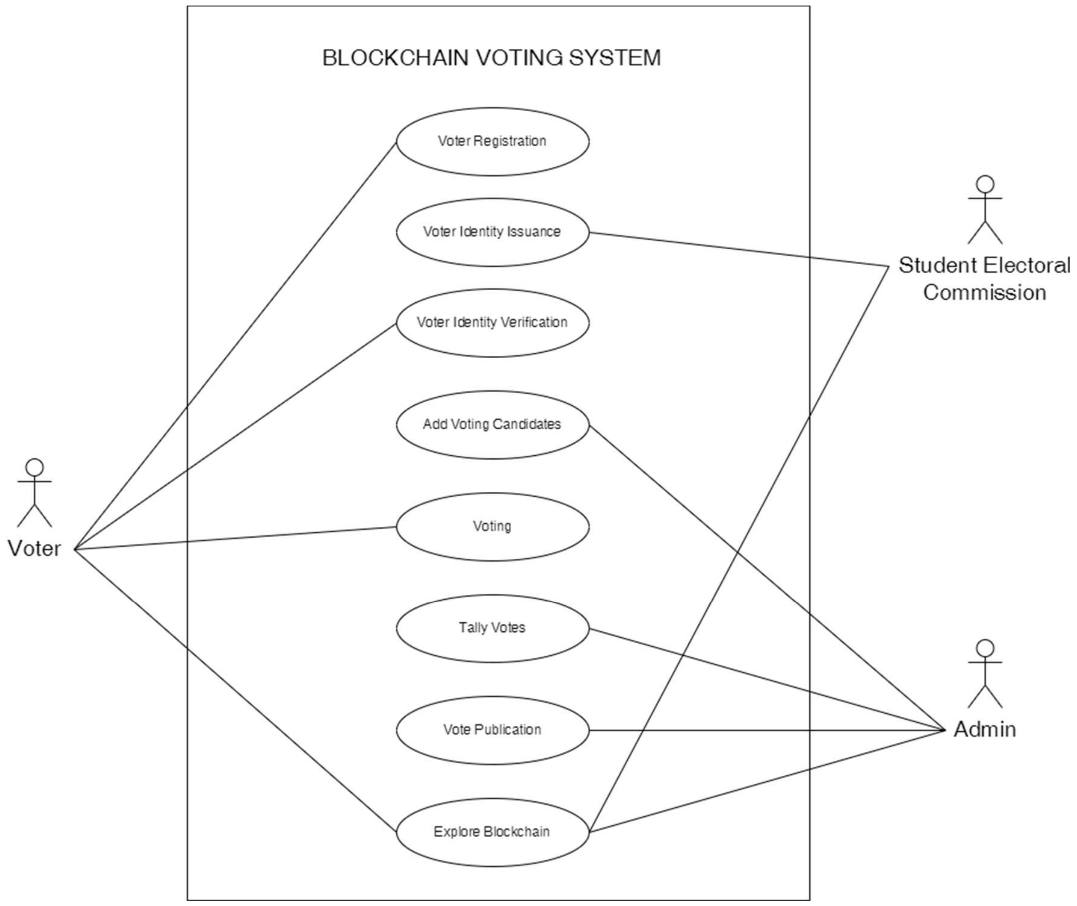


Figure 3.2 Use Case Diagram for Blockchain Voting System

3.5 Context Diagram

Context diagrams focuses on the external factors and events related to the systems requirements and constraints. All external entities are represented in the way they interact with the system. For the blockchain voting system, the interaction between all three actors and the voting system are shown using the context diagram. The voter gets the zero-knowledge proof from the student electoral committee, submits it to the blockchain system to be verified before the voter can submit their vote. The administrator logins to the system, adds voting candidates to the system and tally the votes once the election is finished. The tallied votes can then be viewed by votes.

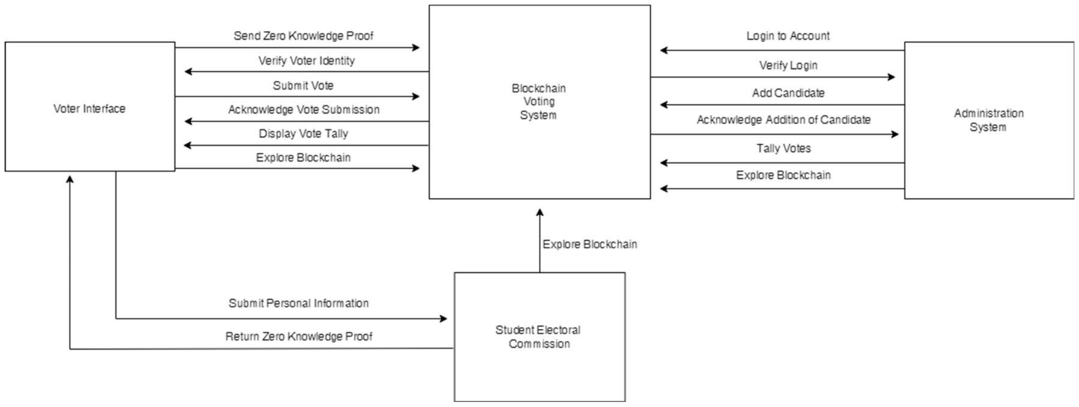


Figure 3.3 Context Diagram for Blockchain Voting System

3.6 Activity Diagram

The activity diagram is used to describe the dynamic aspects of the system. It is an advanced version of the flow chart that models the flow from one activity to another.

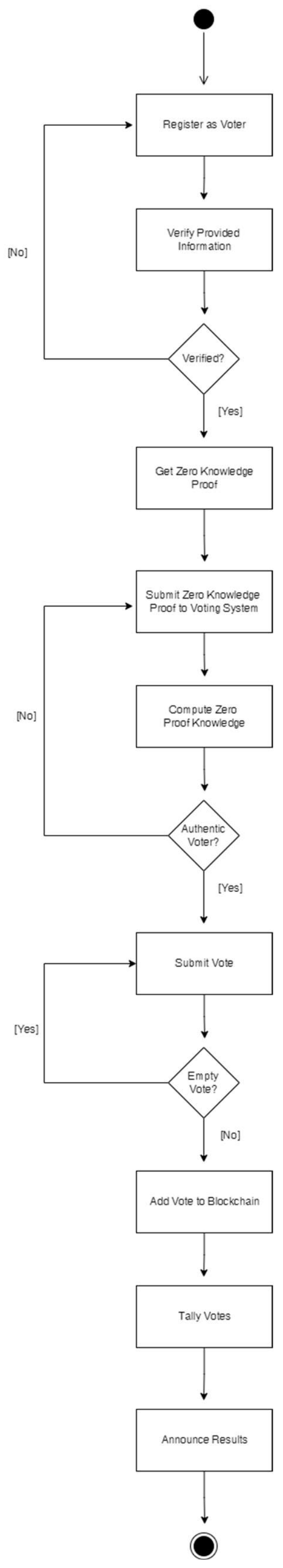


Figure 3.4 Activity Diagram for Blockchain Voting System

3.7 Entity Relationship Diagram

The Entity Relationship Diagram (ERD) is a form of flowchart that serves to illustrate how the internal and external entities relate to one another within a system. The following diagram illustrates the relationships between entities in the blockchain voting system. The voting system utilizes one blockchain, thus the one-to-one relationship. One or more administrators run the voting system along with the blockchain. Voters have 1 StudentID and can have 1 ZKProof only, issued by the only Student Electoral Commission. There are many voting candidates for one election.

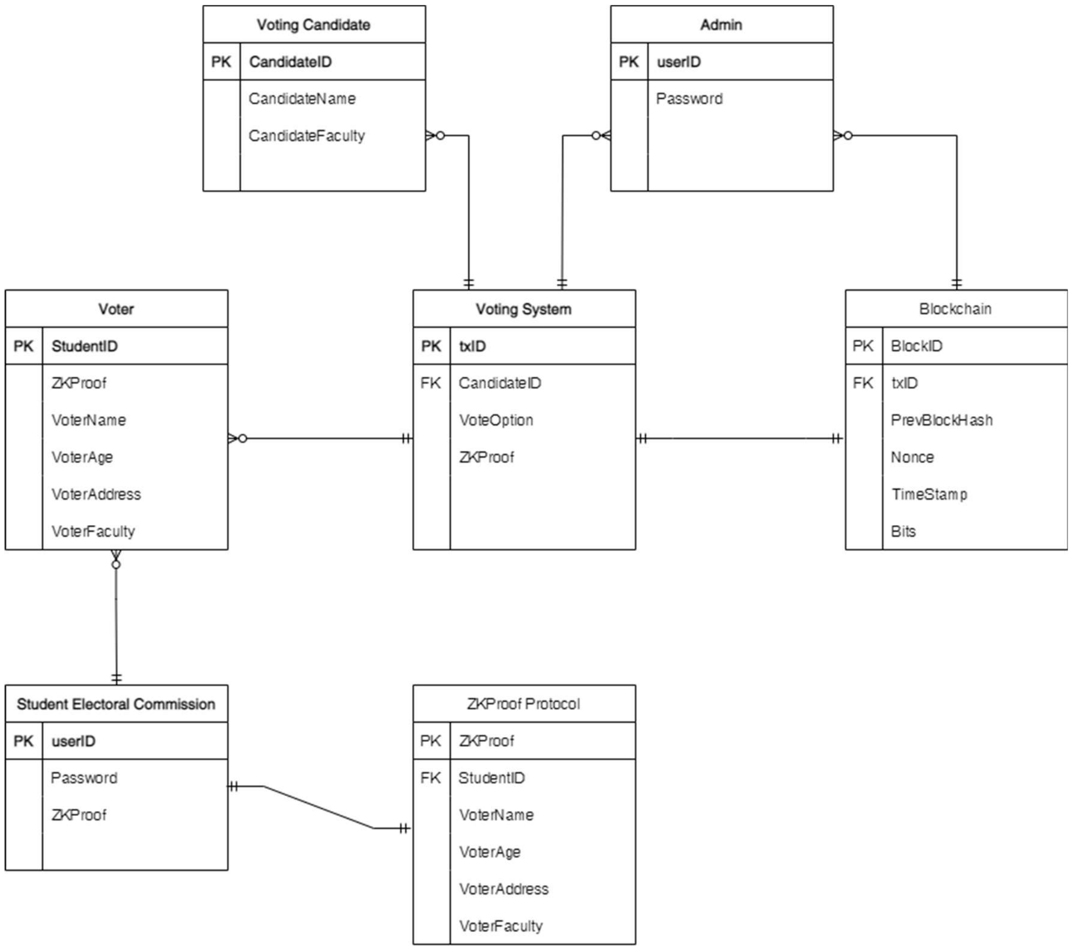


Figure 3.5 Entity Relationship Diagram for Blockchain Voting System

3.8 Database Dictionary

Voter

Field Name	**Data Type**	**Max Field Size**	**Description**	**Example**
StudentID	VARCHAR	7	Unique ID for each individual student	CA19011
VoterName	VARCHAR	64	Name of the student	Ali bin Abdul
VoterAge	INTEGER	2	Age of the student	21
VoterAddress	VARCHAR	64	Address of the student	25, Jalan Ipoh, Taman Johor, Perak
VoterFaculty	VARCHAR	64	Faculty of the student	FKOM
ZKProof	VARCHAR	64	The zero-knowledge proof produced using all above information of the student	90DE95D744785C1A9784591A11EA348A460ABA95B7FC0E357AF56CC3F92A44A5

Table 3.1 Data Dictionary for Voter

Student Electoral Commission

Field Name	**Data Type**	**Max Field Size**	**Description**	**Example**
userID	VARCHAR	7	Unique ID for each individual student	Presiden_Abdul
Password	VARCHAR	64	Name of the student	Password123
ZKProof	VARCHAR	64	The zero-knowledge proof produced using provided information of the student	90DE95D744785C1A9784591A11EA348A460ABA95B7FC0E357AF56CC3F92A44A5

Table 3.2 Data Dictionary for Student Electoral Commission

ZKProof Protocol

Field Name	**Data Type**	**Max Field Size**	**Description**	**Example**
ZKProof	VARCHAR	64	The zero-knowledge proof produced using all above information of the student	90DE95D744785C1A9784591A11EA348A460ABA95B7FC0E357AF56CC3F92A44A5
StudentID	VARCHAR	7	Unique ID for each individual student	CA19011
VoterName	VARCHAR	64	Name of the student	Ali bin Abdul
VoterAge	INTEGER	2	Age of the student	21
VoterAddress	VARCHAR	64	Address of the student	25, Jalan Ipoh, Taman Johor, Perak
VoterFaculty	VARCHAR	64	Faculty of the student	FKOM

Table 3.3 Data Dictionary for ZKProof Protocol

Voting Candidate

Field Name	**Data Type**	**Max Field Size**	**Description**	**Example**
CandidateID	VARCHAR	7	Unique ID for each voting candidate	VOTE001
CandidateName	VARCHAR	64	Name of the student	Chee Kor Yeow
VoterFaculty	VARCHAR	64	Faculty of the student	FIM

Table 3.4 Data Dictionary for Voting Candidate

Admin

Field Name	**Data Type**	**Max Field Size**	**Description**	**Example**
userID	VARCHAR	7	Login ID for the administrator	admin
password	VARCHAR	64	Password of the administrator	admin

Table 3.5 Data Dictionary for Admin

Voting System

Field Name	**Data Type**	**Max Field Size**	**Description**	**Example**
txID	VARCHAR	128	Unique ID for the transaction of each submitted vote	0x7d1821b5544c631f0932acd34201660cac7400999133118d108a69e93daaa1d9
CandidateID	VARCHAR	64	ID of the voting candidate	VOTE001
VoteOption	BINARY	2	Chosen option of the voter	01
ZKProof	VARCHAR	64	The zero-knowledge proof produced using all above information of the student	90DE95D744785C1A9784591A11EA348A460ABA95B7FC0E357AF56CC3F92A44A5

Table 3.6 Data Dictionary for Voting System

Blockchain

Field Name	**Data Type**	**Max Field Size**	**Description**	**Example**
BlockID	INTEGER	32	The ID of the block containing votes	157
txID	VARCHAR	128	Unique ID for the transaction of each submitted vote	0x7d1821b5544c631f0932acd34201660cac7400999133118d108a69e93daaa1d9
PrevBlockHash	VARCHAR	64	The hash value of the previous block	0000000000000000e067a478024addfecdc93628978aa52d91fabd4292982a50
Nonce	VARCHAR	64	Random number/value incremented and added to provide new hash value.	5645689
TimeStamp	INTEGER	64	UNIX format timestamp	1395103695
Bits	INTEGER	64	The difficulty to mine the block	27967152

Table 3.7 Data Dictionary for Blockchain

3.9 Interface Design

3.9.1 Voter Registration Page

On this page, voters will input their personal information to create their unique zero-knowledge proof to verify their identity as a voter. This page is hosted and handled by the Student Electoral Commission, which then provides the zero-knowledge proof to the voter after their identity has been verified by the Student Electoral Commission. This zero-knowledge proof can then be used in the blockchain voting system's page, which does not deal with confidential user information by design.

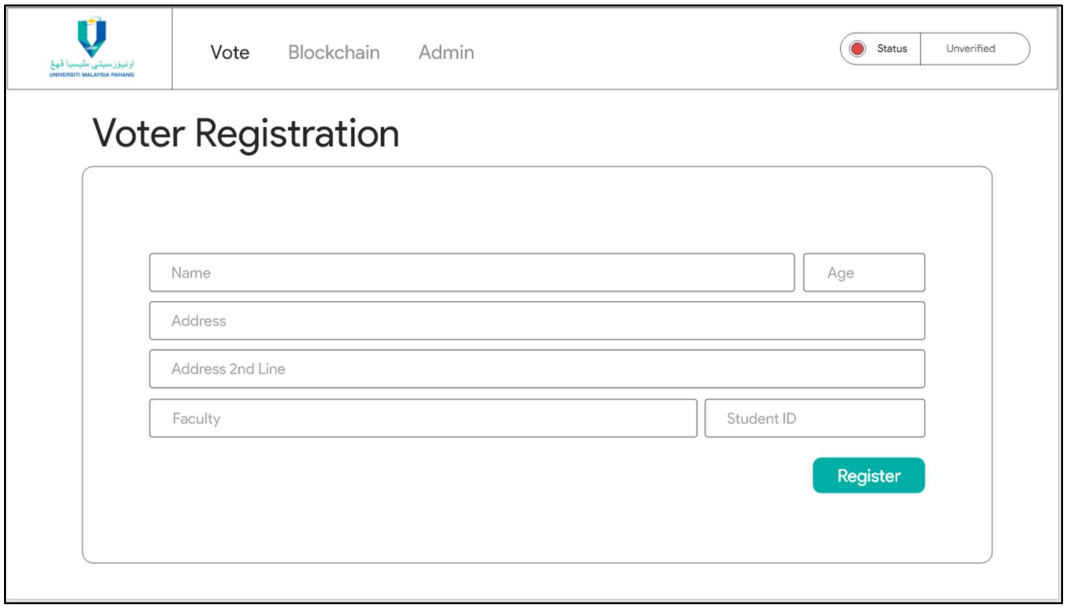


Figure 3.6 Voter Registration Page for Blockchain Voting System

3.9.2 Voter Main Page

Voters are greeted by this page when they open the URL. They will then input their zero-knowledge proof in the provided space. If verification is successful, the status on the top right corner will turn green and verified.

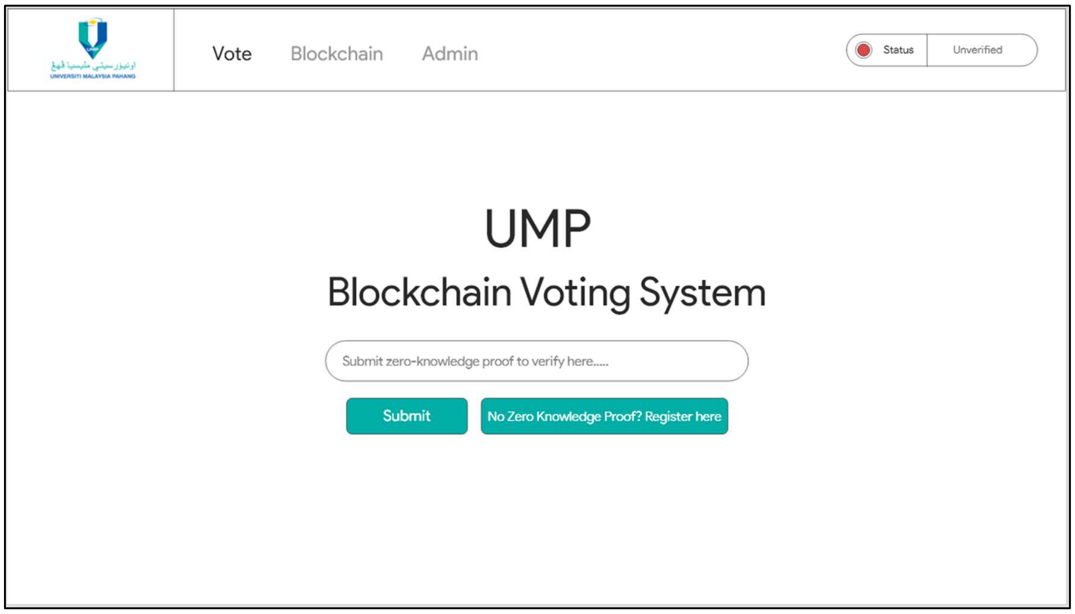

UNIVERSITI MALAYSIA PAHANG
Vote
Blockchain
Admin
Status
Unverified
UMP
Blockchain Voting System
Submit zero-knowledge proof to verify here.....
Submit
No Zero Knowledge Proof? Register here

Figure 3.7 Voter Main Page for Blockchain Voting System

3.9.3 Voting Page

After successful verification, voters will be redirected to the voting page. Here they can view the available candidates and their information. They can then proceed to cast their vote which will produce a confirmation and add the vote/transaction vote to the blockchain.

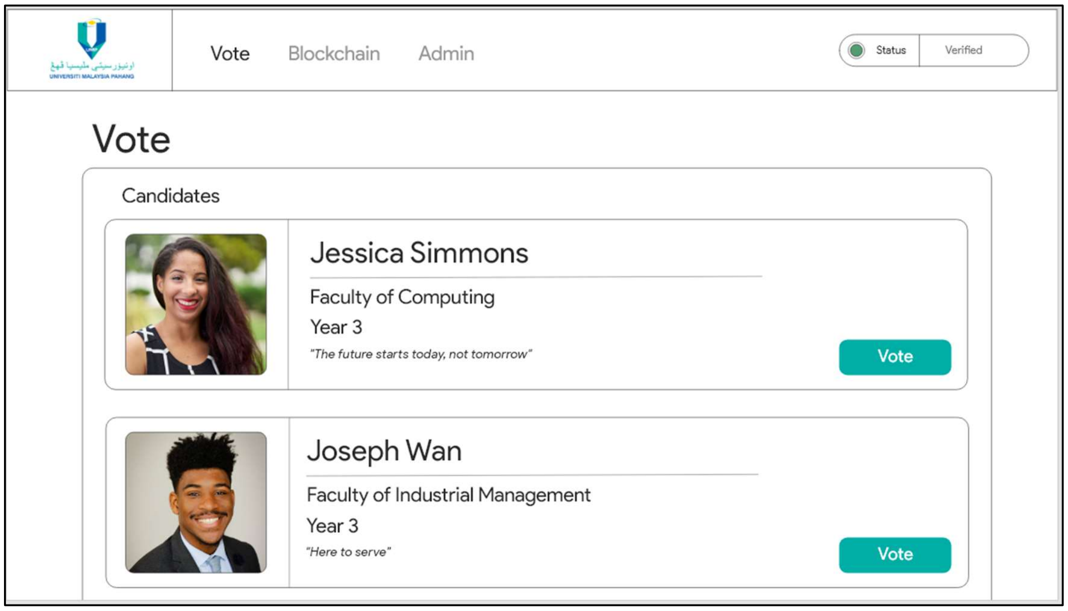


Figure 3.8 Voting Page for Blockchain Voting System

3.9.4 Blockchain Page

After the election has ended, voters can navigate to this page to view there vote on the blockchain. They can also download the whole blockchain if they wish to audit the election.

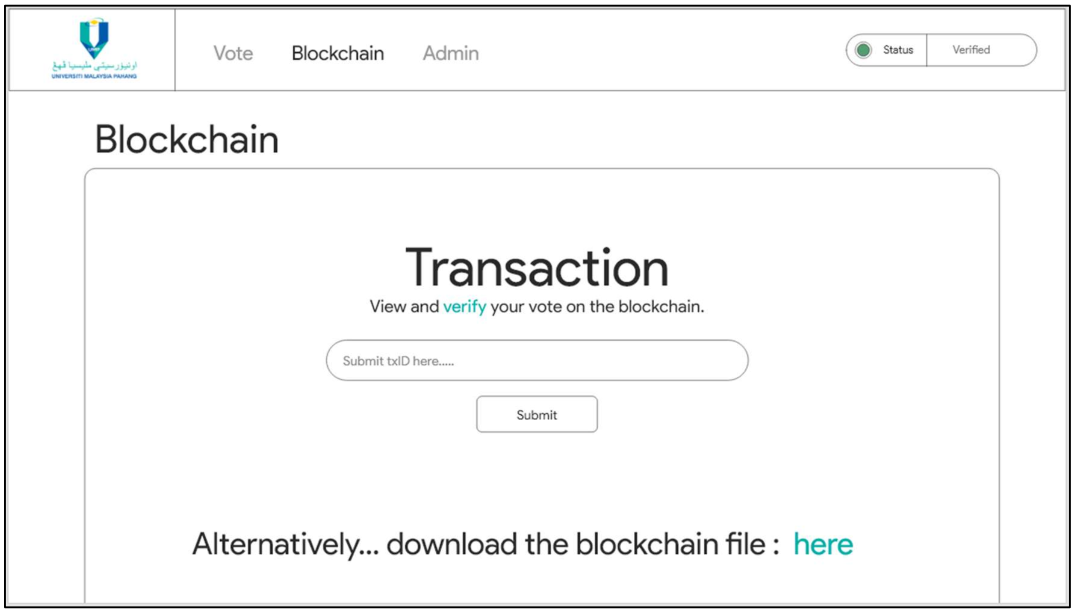

UNIVERSITI MALAYSIA PAHANG
Vote
Blockchain
Admin
Status
Verified
Blockchain
Transaction
View and verify your vote on the blockchain.
Submit txID here.....
Submit
Alternatively... download the blockchain file : here

Figure 3.9 Blockchain Page for Blockchain Voting System

3.9.5 Admin Login Page

The administrator of the voting system will login into the system through this page.

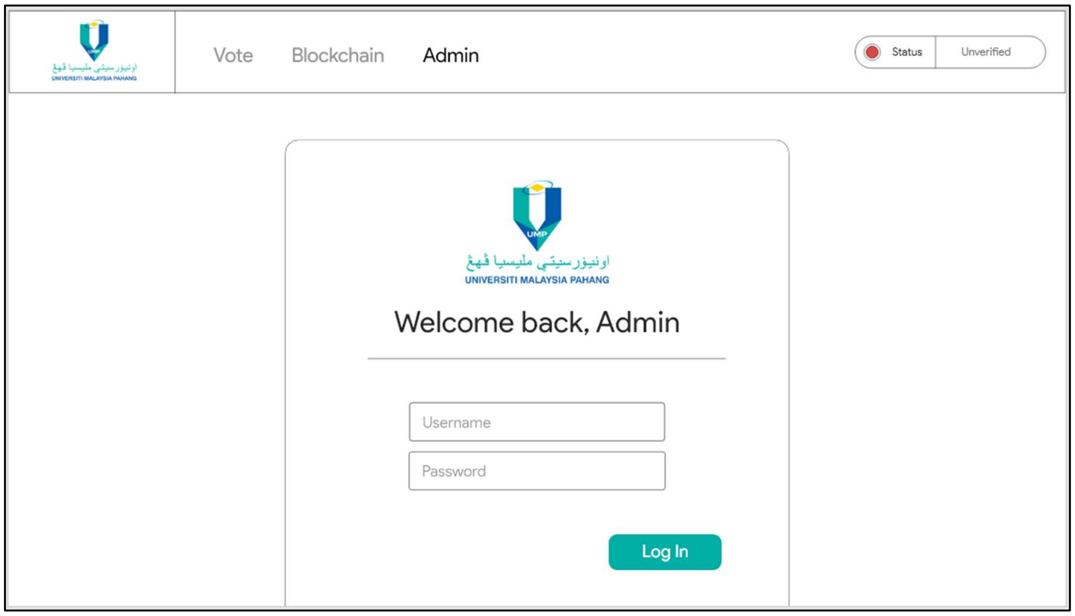


Figure 3.10 Admin Login Page for Blockchain Voting System

3.9.6 Admin Landing Page

Once logged in, the administrator will be presented with multiple actions to perform. These include adding /removing candidates and tallying the votes.

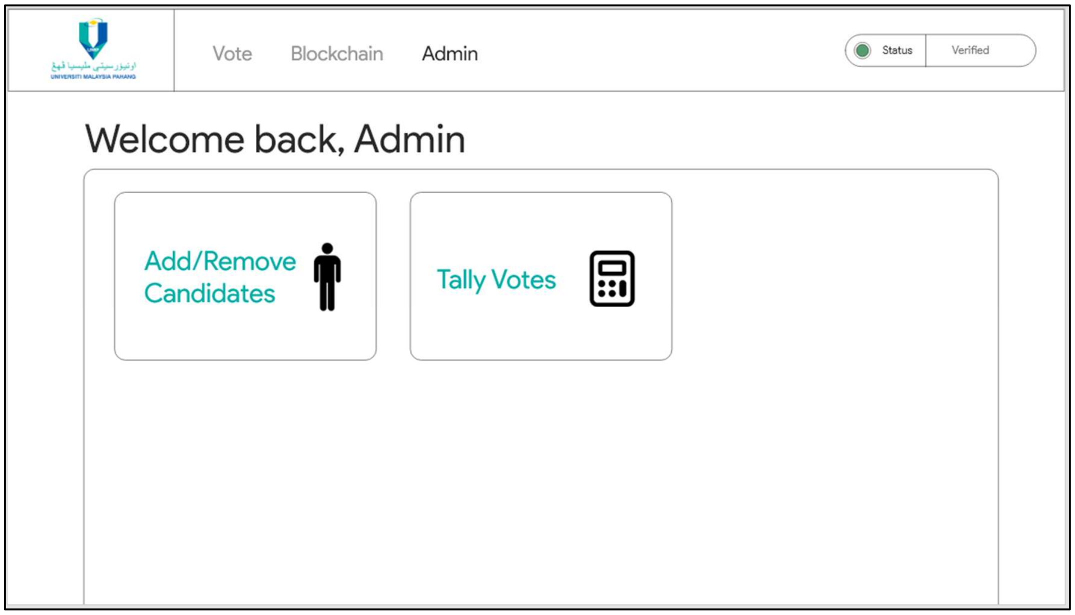


Figure 3.11 Admin Landing Page for Blockchain Voting System

3.9.7 Add Candidate Page

The administrator adds candidates to the election through this page and must provide their picture, name, faculty, and election slogan.

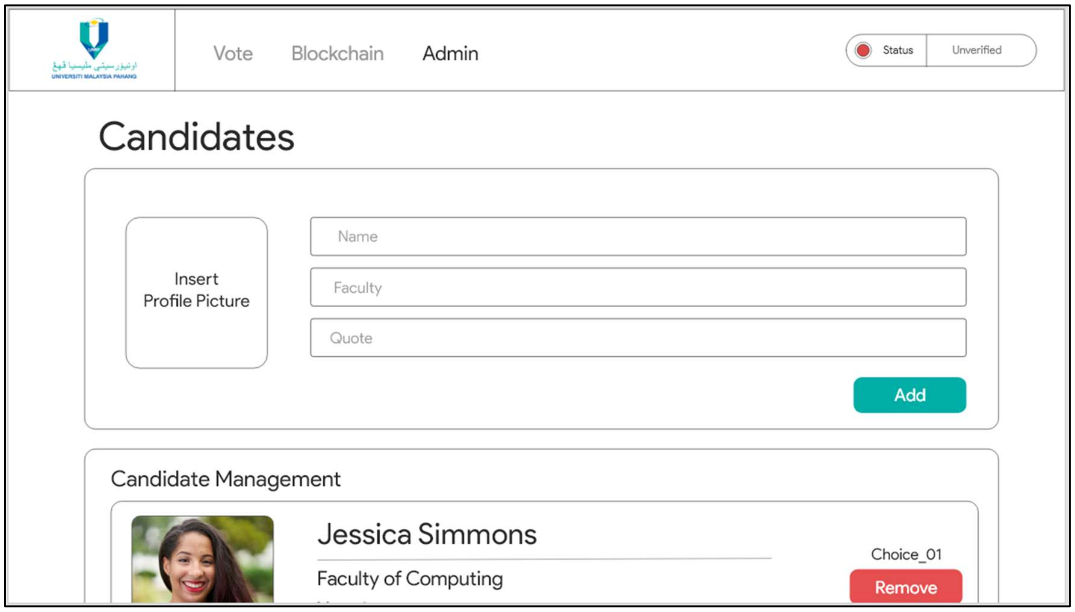


Figure 3.12 Add Candidate Page for Blockchain Voting System

3.9.8 End Election Page

The administrator will be presented with this page after pressing the "End Election" button.

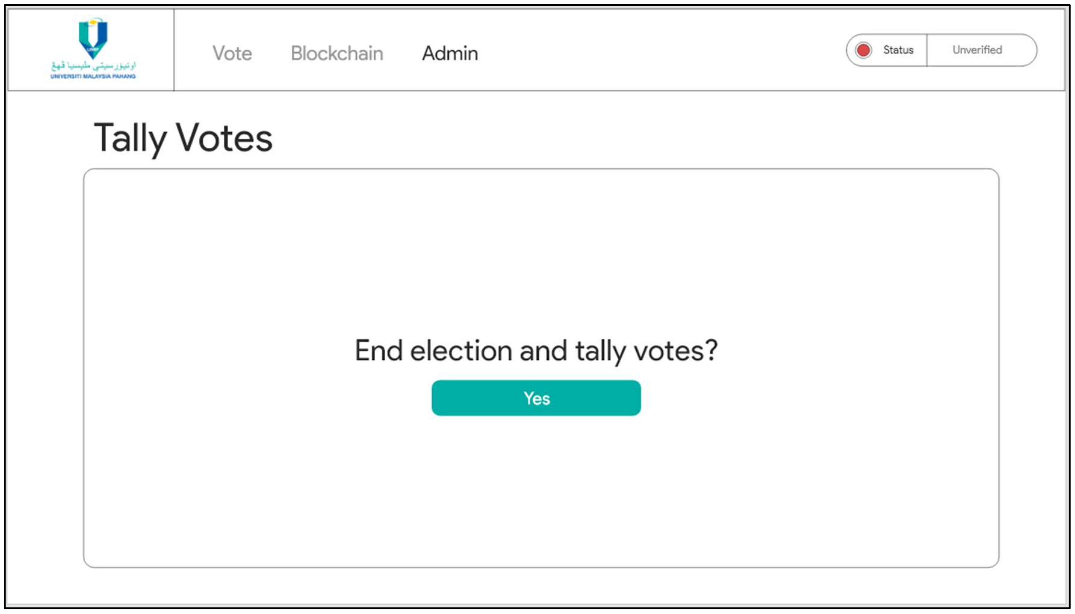
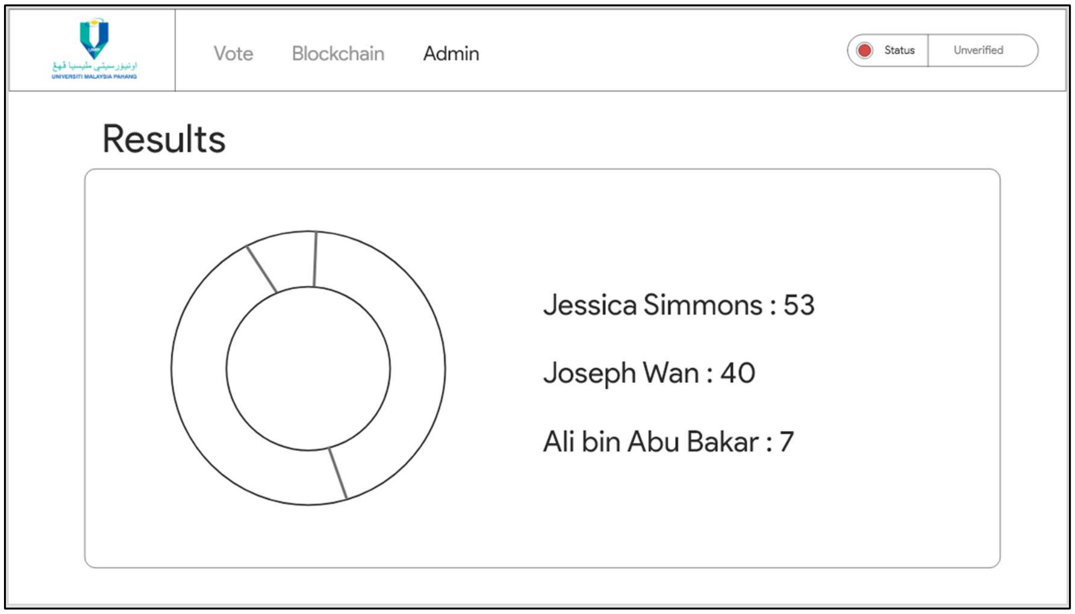


Figure 3.13 End Election Page for Blockchain Voting System

3.9.9 Results Page

After ending the election, the administrator will be redirected to the results page and the final count will be shown for each candidate.



Jessica Simmons : 53

Joseph Wan : 40

Ali bin Abu Bakar : 7

Figure 3.14 Results Page for Blockchain Voting System

3.10 Testing Plan

The testing will be conducted on all aspects of the system from the initial generation of the zero-knowledge proof of the voter, the submission and tallying of the votes and the displaying of the final results. Usability, functionality, performance, and security will be the priority of the testing plan as exploits in the system would be detrimental to the blockchain voting system.

Information regarding usability and functionality will be gathered through tester response, where these testers will be recruited to try out the system before deployment occurs and fill in the User Acceptance Testing (UAT) form based on their experience. As over development framework is iterative, the feedback received will be used to further refine the system until a satisfactory standard is achieved. The UAT form is shown in the next page.

UAT Form

Test Case ID : TC-PROM-XX				Test Designed By : Lim Aun Xian		
Test Priority (Low/Medium/High) :				Test Designed Date : DD/MM/YYYY		
Module Name : XXX				Test Executed By : XXX		
Test Title : XXX				Test Execution Date : DD/MM/YYYY		
Description : XXX						
Precondition : XXX						
Dependencies : XXX						
Steps	Instruction	Test Data	Expected Result	Actual Result	Status (Pass/Fail)	Comments
1.	XXX	XXX	XXX	XXX	XXX	XXX

3.11 Potential Use of Proposed Solution

The successful implementation of the blockchain voting system, flawless execution of the blockchain election along with positive feedback from all parties involved would provide the groundwork for the application of blockchain in our daily life. Blockchain as a concept was conceived with the idea of removing authority from central entities and returning said power to individuals, where opposing opinions cannot be stifled, and transparency of all transactions are open for all to view and verify.

In this project the implementation of a blockchain voting system for UMP Student Council elections is explored. The administrator of the voting system coordinates with the Student Electoral Commission with the issuance of zero-knowledge proofs to the students, who then casts their vote from their respective devices from the comforts of their dormitory. Votes submitted are stored on the blockchain and any attempts at changing the results, albeit very hard, would be easily detected. Announcement of the results would be swift after the election period has ended and the results is available for all to verify.

Moving forward, activities requiring voting could be conducted utilizing blockchain. From a board of directors meeting to a nation's general election, blockchain voting could/should be used to prevent allegations of voter fraud and vote tampering, where all votes are immutable and verifiable. Events such as the allegations of voter fraud presented by President Donald Trump during the 2020 American Presidential Election can be avoided entirely.

Zero-knowledge proofs as a concept has existed since the 1980s, and recently have gained traction in the cryptocurrency community to increase privacy and security. Utilizing zero-knowledge proofs to verify the authenticity of a statement, in this case the identity of the voter, without the exchange of sensitive personal information could be applied in many fields dealing with sensitive information like finance, government offices, social networks and so on.

3.12 Gantt Chart

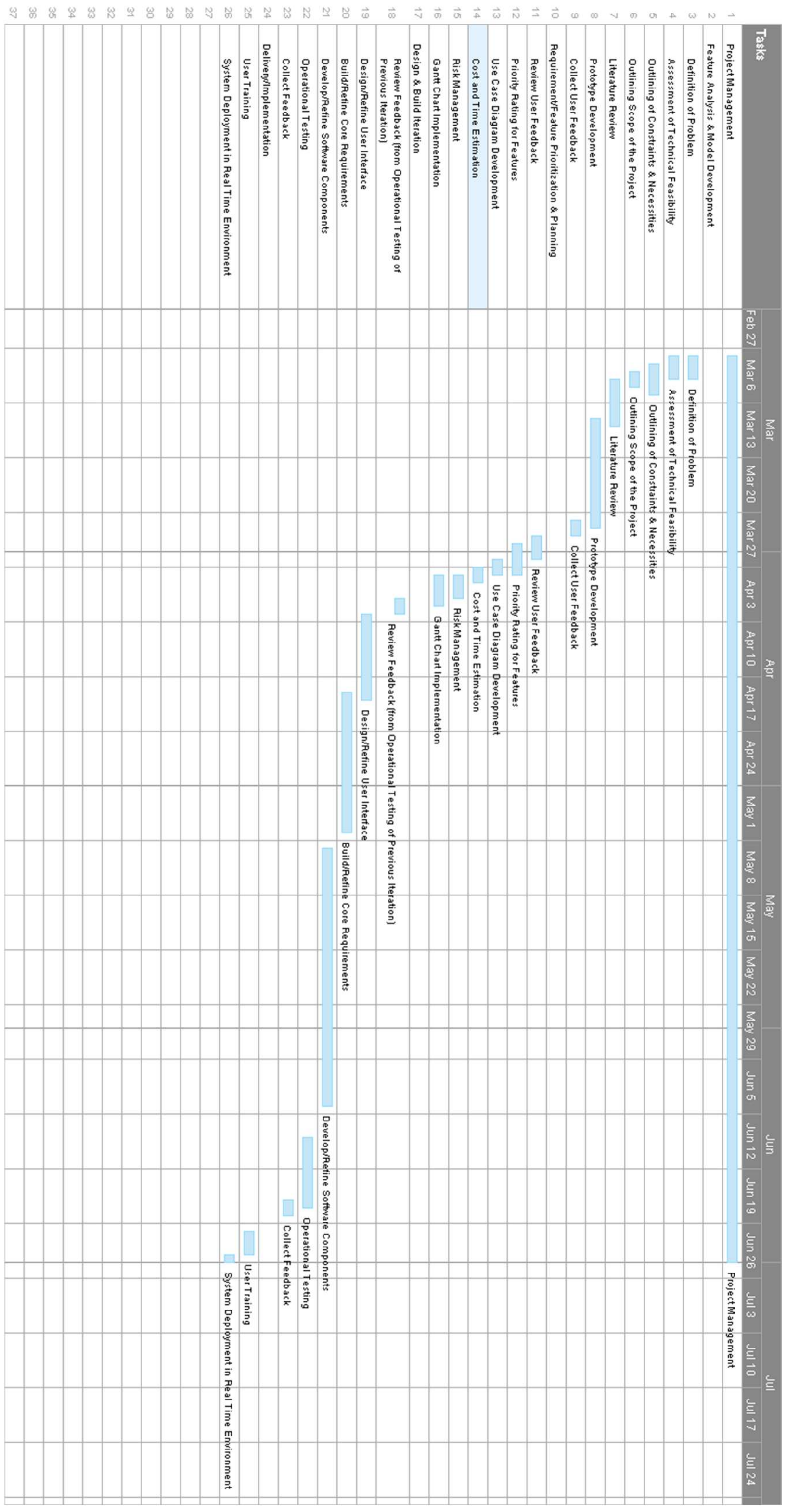


Figure 3.15 Gantt Chart for Blockchain Voting System

3.13 Summary of Chapter 3

Chapter 3 discusses the framework used for this project and the justification for choosing the framework. The DSDM framework modifies the features delivery and maintains the cost and time of production, while keeping the core requirements of the project. Use case diagrams, context diagrams, and other diagrams are used to show the interaction between all entities involved in the system. The user interface and functionalities for all the different webpages are shown. The testing plan is outlined, and the many potentials uses for the proposed solution is elaborated. A Gantt chart is used to schedule all the tasks involved in the successful execution of this project.

CHAPTER 4

RESULTS AND DISCUSSION

4.1 Introduction

With the completion of the initial planning stages and research done with regards to the topic at hand, development of the blockchain project can then be started. Chapter 4 covers the steps taken to develop the project, how the backend was assembled, how the frontend has evolved from previous concepts, justification for methods used during development, flow of the process of voting and the outputs produced at every step of the election.

4.2 Implementation

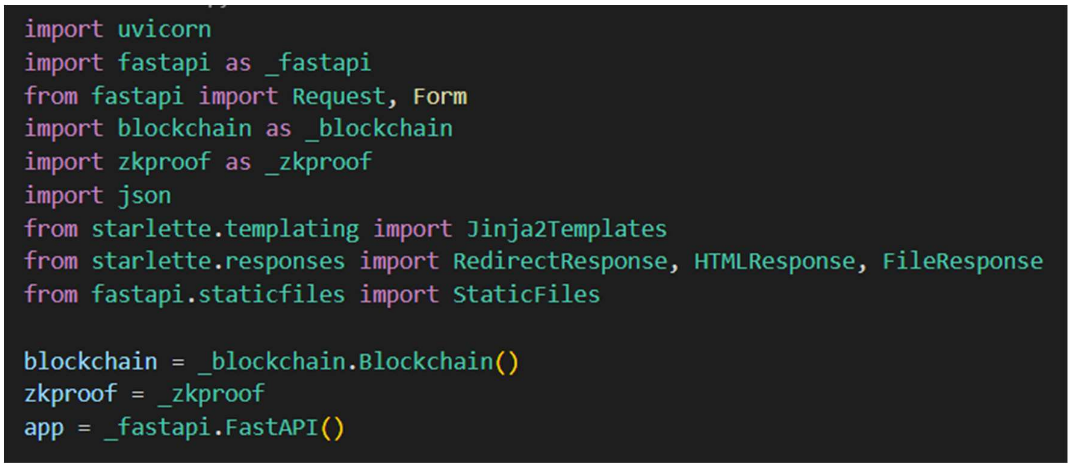
For the project's implementation, it is split into two sections: the backend which handles all data processing and exchanges, and the frontend: which involves the user interface that voters interact with in order to vote. The main functions, along with the other submodules that help make up the backend are showcased and shown in detail. The following two sections discusses the various modules and components that make up the blockchain voting system.

4.3 Backend

The blockchain voting project was developed using the FastAPI web framework to utilize Python as its backend. Three *.py* files are used is this project: the *main.py* for the API implementation and running the web server, the *zkproof.py* for zero knowledge proof and the *blockchain.py* for all blockchain related operations.

4.3.1 Main.py

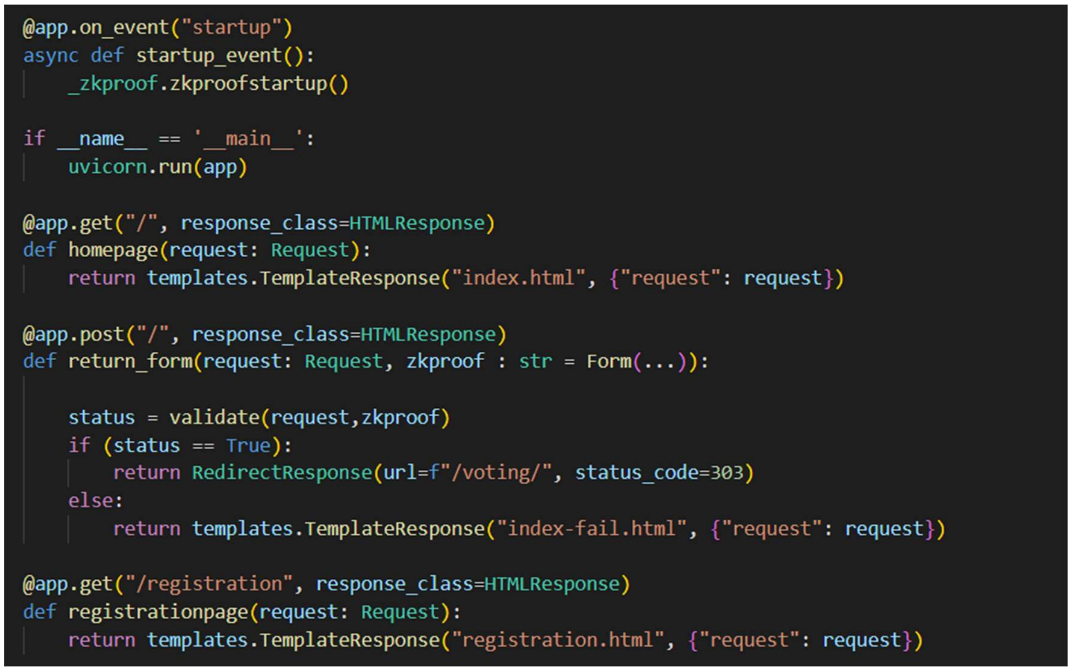
In the main.py file, development is starts by importing the necessary modules and assigning the necessary variables for later use. *Uvicorn* is used to implement web servers in python, *FastAPI* provides the framework and APIs for the project, *json* is used to write to files later on and imports from *starlette* are used for the frontend.



```python
import uvicorn
import fastapi as _fastapi
from fastapi import Request, Form
import blockchain as _blockchain
import zkproof as _zkproof
import json
from starlette.templating import Jinja2Templates
from starlette.responses import RedirectResponse, HTMLResponse, FileResponse
from fastapi.staticfiles import StaticFiles

blockchain = _blockchain.Blockchain()
zkproof = _zkproof
app = _fastapi.FastAPI()
```

Figure 4.1 : imports and assignments for main.py

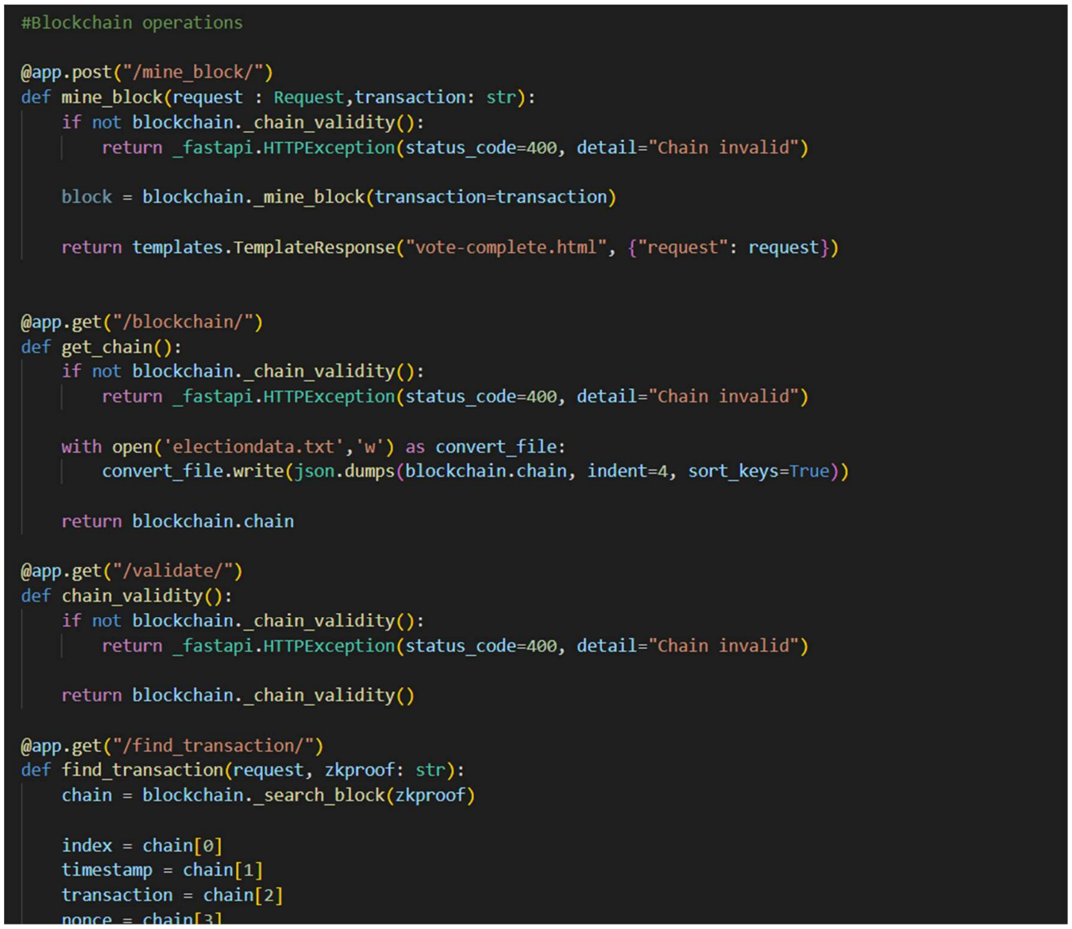
On server startup, the threads used for the client(user) function and server function of ZKProof is started. The *@app.get* and *@app.post* functions are used to navigate and pass data between pages for the project.



```python
@app.on_event("startup")
async def startup_event():
    _zkproof.zkproofstartup()

if __name__ == '__main__':
    uvicorn.run(app)

@app.get("/", response_class=HTMLResponse)
def homepage(request: Request):
    return templates.TemplateResponse("index.html", {"request": request})

@app.post("/", response_class=HTMLResponse)
def return_form(request: Request, zkproof : str = Form(...)):

    status = validate(request,zkproof)
    if (status == True):
        return RedirectResponse(url=f"/voting/", status_code=303)
    else:
        return templates.TemplateResponse("index-fail.html", {"request": request})

@app.get("/registration", response_class=HTMLResponse)
def registrationpage(request: Request):
    return templates.TemplateResponse("registration.html", {"request": request})
```

Figure 4.2 : univorn and FastAPI components

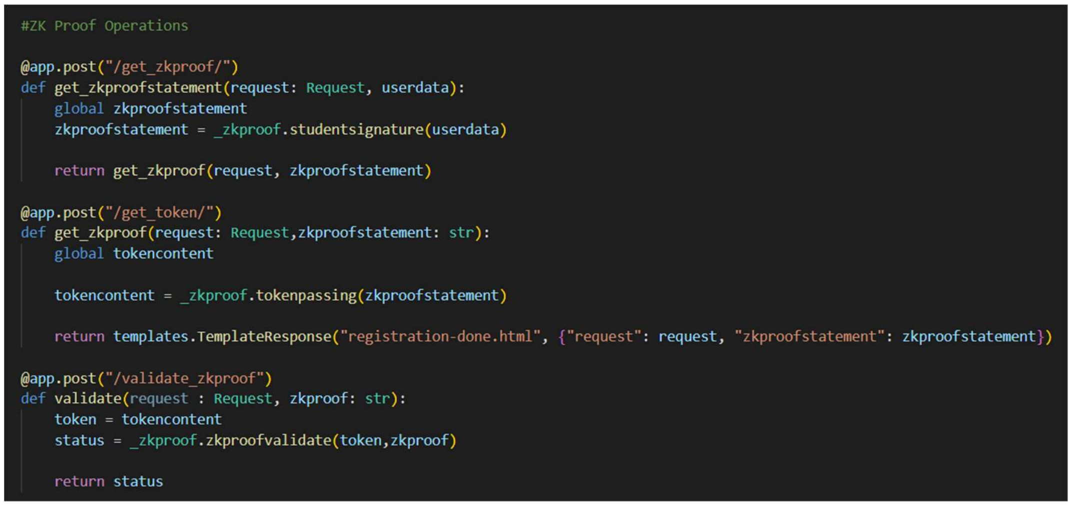
The functions used to utilize the blockchain are also declared here and, with the import of the blockchain module at the start, can then be used by the API to interact with the blockchain.



```python
#Blockchain operations

@app.post("/mine_block/")
def mine_block(request : Request,transaction: str):
    if not blockchain._chain_validity():
        return _fastapi.HTTPException(status_code=400, detail="Chain invalid")

    block = blockchain._mine_block(transaction=transaction)

    return templates.TemplateResponse("vote-complete.html", {"request": request})


@app.get("/blockchain/")
def get_chain():
    if not blockchain._chain_validity():
        return _fastapi.HTTPException(status_code=400, detail="Chain invalid")

    with open('electiondata.txt','w') as convert_file:
        convert_file.write(json.dumps(blockchain.chain, indent=4, sort_keys=True))

    return blockchain.chain

@app.get("/validate/")
def chain_validity():
    if not blockchain._chain_validity():
        return _fastapi.HTTPException(status_code=400, detail="Chain invalid")

    return blockchain._chain_validity()

@app.get("/find_transaction/")
def find_transaction(request, zkproof: str):
    chain = blockchain._search_block(zkproof)

    index = chain[0]
    timestamp = chain[1]
    transaction = chain[2]
    nonce = chain[3]
```

Figure 4.3 : Blockchain functions in main.py

In order to utilize the ZKProof functions as well, methods to call and pass data to them using the FastAPI are also declared here.

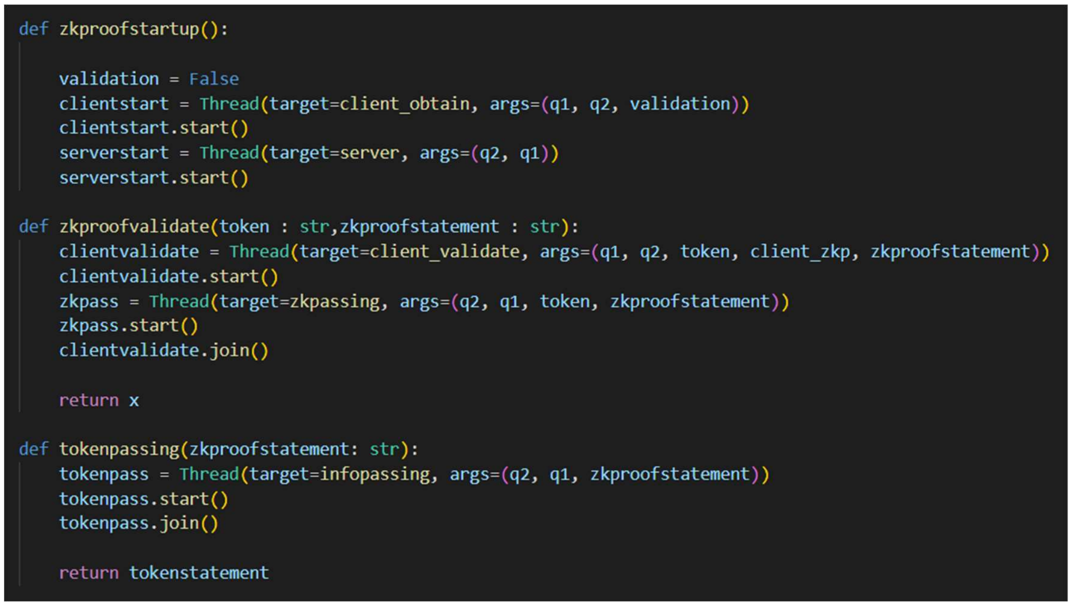


```
#ZK Proof Operations

@app.post("/get_zkproof/")
def get_zkproofstatement(request: Request, userdata):
    global zkproofstatement
    zkproofstatement = _zkproof.studentsignature(userdata)

    return get_zkproof(request, zkproofstatement)

@app.post("/get_token/")
def get_zkproof(request: Request,zkproofstatement: str):
    global tokencontent

    tokencontent = _zkproof.tokenpassing(zkproofstatement)

    return templates.TemplateResponse("registration-done.html", {"request": request, "zkproofstatement": zkproofstatement})

@app.post("/validate_zkproof")
def validate(request : Request, zkproof: str):
    token = tokencontent
    status = _zkproof.zkproofvalidate(token,zkproof)

    return status
```

Figure 4.4 : ZKProof functions in main.py

4.3.2 ZKProof.py

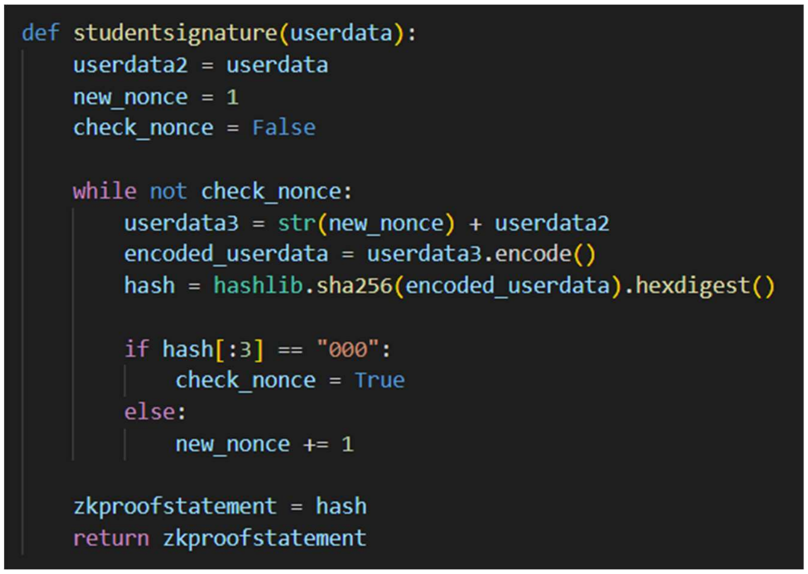
The zkproof.py contains all the necessary functions to produce the ZKProof and the ways to validate said ZKProof. In order to accomplish the operation, threads are started for the client, which sends the information to the server to create the ZKProof and return the ZKProof to the user. Threads are also started for the server to remain operational and return the ZKProof to the user in the event of a submission. The NoKnow implementation is used in order to provide zero knowledge proofs authentication.



```python
def zkproofstartup():

    validation = False
    clientstart = Thread(target=client_obtain, args=(q1, q2, validation))
    clientstart.start()
    serverstart = Thread(target=server, args=(q2, q1))
    serverstart.start()

def zkproofvalidate(token : str,zkproofstatement : str):
    clientvalidate = Thread(target=client_validate, args=(q1, q2, token, client_zkp, zkproofstatement))
    clientvalidate.start()
    zkpass = Thread(target=zkpassing, args=(q2, q1, token, zkproofstatement))
    zkpass.start()
    clientvalidate.join()

    return x

def tokenpassing(zkproofstatement: str):
    tokenpass = Thread(target=infopassing, args=(q2, q1, zkproofstatement))
    tokenpass.start()
    tokenpass.join()

    return tokenstatement
```

Figure 4.5 : Threads for client and server in zkproof.py

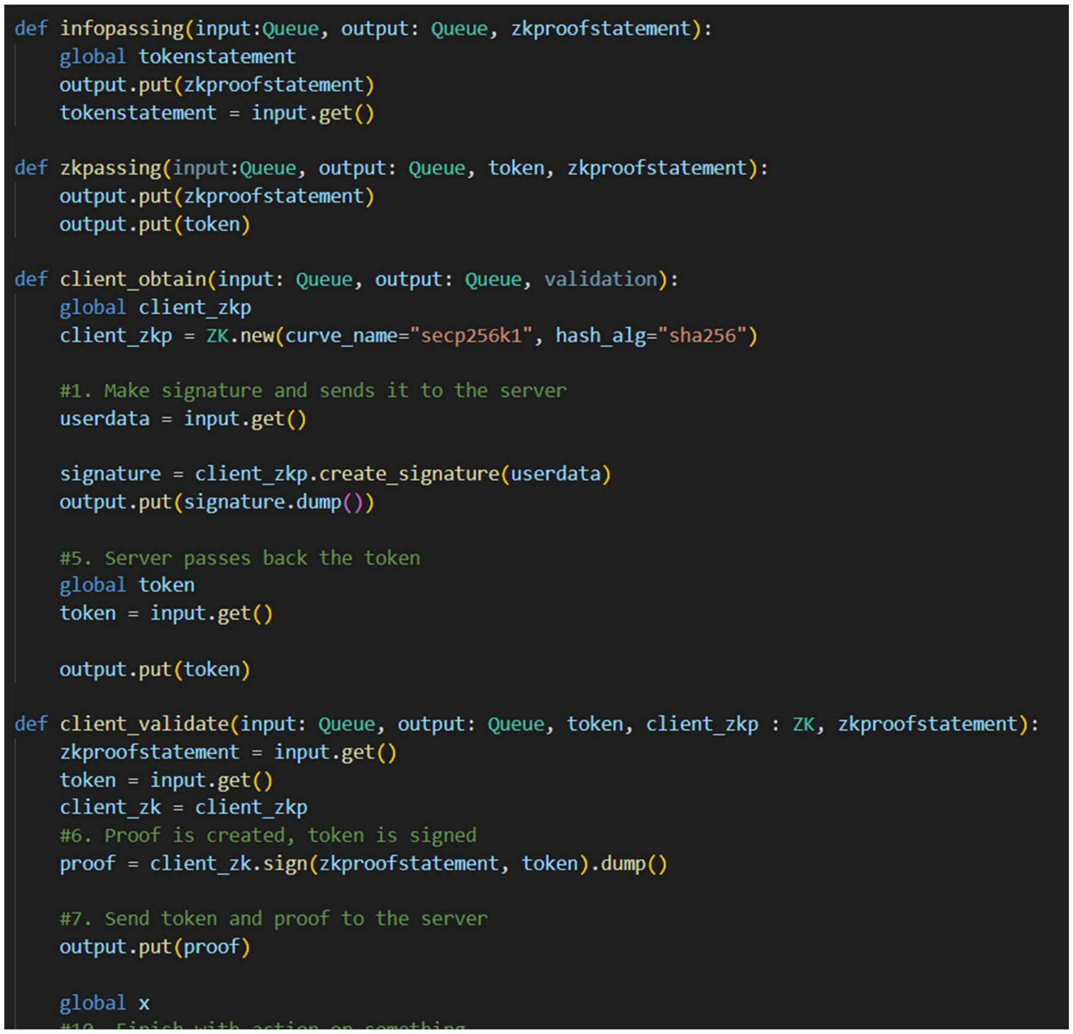
The *studentsignature* function takes the provided user data from user input, combines it into a string, add a nonce then hashes it to produce a statement that starts with 3 zeros. The hashed statement is then used as the ZKProof statement for the user and is then returned to the user.



```python
def studentsignature(userdata):
    userdata2 = userdata
    new_nonce = 1
    check_nonce = False

    while not check_nonce:
        userdata3 = str(new_nonce) + userdata2
        encoded_userdata = userdata3.encode()
        hash = hashlib.sha256(encoded_userdata).hexdigest()

        if hash[:3] == "000":
            check_nonce = True
        else:
            new_nonce += 1

    zkproofstatement = hash
    return zkproofstatement
```

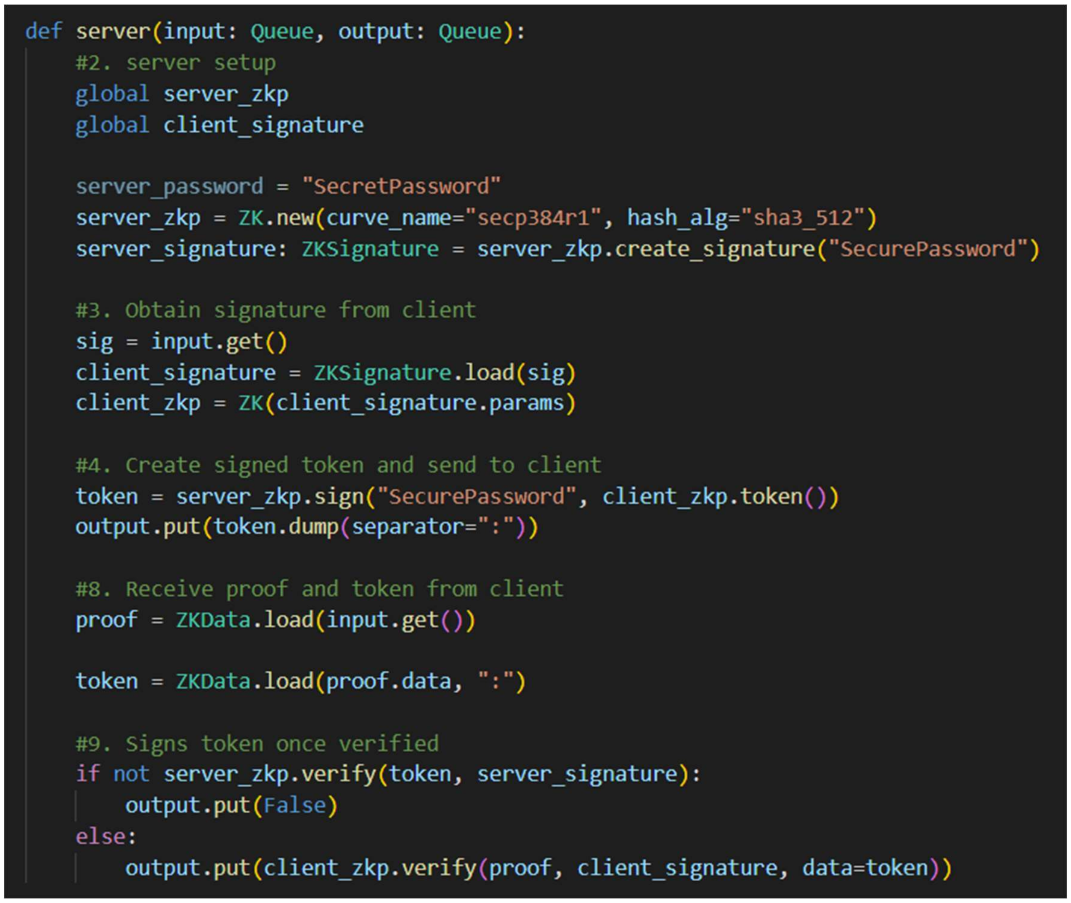
Figure 4.6 : studentsignature function in zkproof.py

The *infopassing* and *zkpassing* functions are used to pass information to the client and server threads during the process of obtaining the ZKProof. The client function is split in two, *client_obtain* to obtain the ZKProof and *client_validate* for the passing and returning of status for said ZKProof. The server function handles the token creation and validation of ZKProof.



```python
def infopassing(input:Queue, output: Queue, zkproofstatement):
    global tokenstatement
    output.put(zkproofstatement)
    tokenstatement = input.get()

def zkpassing(input:Queue, output: Queue, token, zkproofstatement):
    output.put(zkproofstatement)
    output.put(token)

def client_obtain(input: Queue, output: Queue, validation):
    global client_zkp
    client_zkp = ZK.new(curve_name="secp256k1", hash_alg="sha256")

    #1. Make signature and sends it to the server
    userdata = input.get()

    signature = client_zkp.create_signature(userdata)
    output.put(signature.dump())

    #5. Server passes back the token
    global token
    token = input.get()

    output.put(token)

def client_validate(input: Queue, output: Queue, token, client_zkp : ZK, zkproofstatement):
    zkproofstatement = input.get()
    token = input.get()
    client_zk = client_zkp
    #6. Proof is created, token is signed
    proof = client_zk.sign(zkproofstatement, token).dump()

    #7. Send token and proof to the server
    output.put(proof)

    global x
    #10. Finish with action on something
```

Figure 4.7 : passing and client functions in zkproof.py

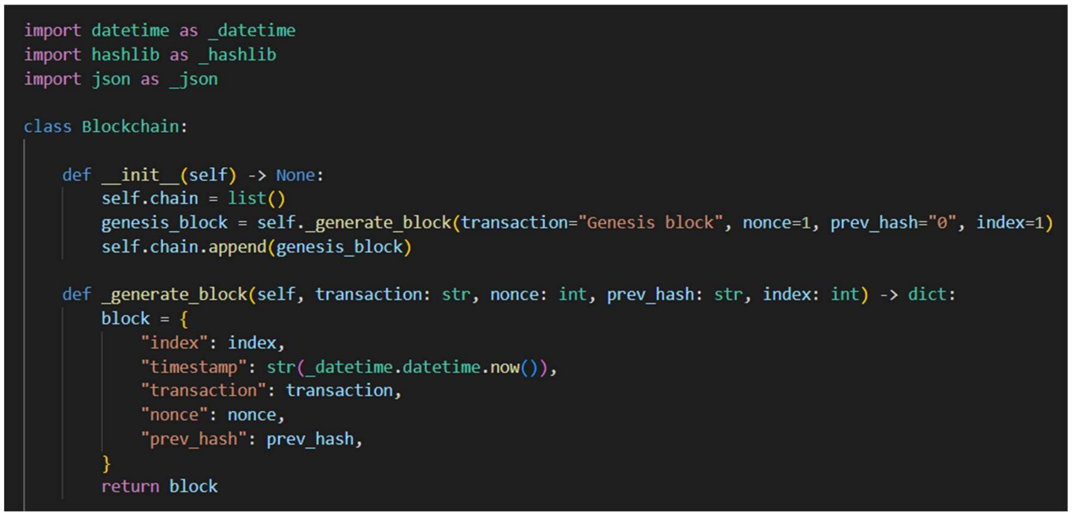


```python
def server(input: Queue, output: Queue):
    #2. server setup
    global server_zkp
    global client_signature

    server_password = "SecretPassword"
    server_zkp = ZK.new(curve_name="secp384r1", hash_alg="sha3_512")
    server_signature: ZKSignature = server_zkp.create_signature("SecurePassword")

    #3. Obtain signature from client
    sig = input.get()
    client_signature = ZKSignature.load(sig)
    client_zkp = ZK(client_signature.params)

    #4. Create signed token and send to client
    token = server_zkp.sign("SecurePassword", client_zkp.token())
    output.put(token.dump(separator=":"))

    #8. Receive proof and token from client
    proof = ZKData.load(input.get())

    token = ZKData.load(proof.data, ":")

    #9. Signs token once verified
    if not server_zkp.verify(token, server_signature):
        output.put(False)
    else:
        output.put(client_zkp.verify(proof, client_signature, data=token))
```

Figure 4.8 : server functions in zkproof.py

4.3.3 Blockchain.py

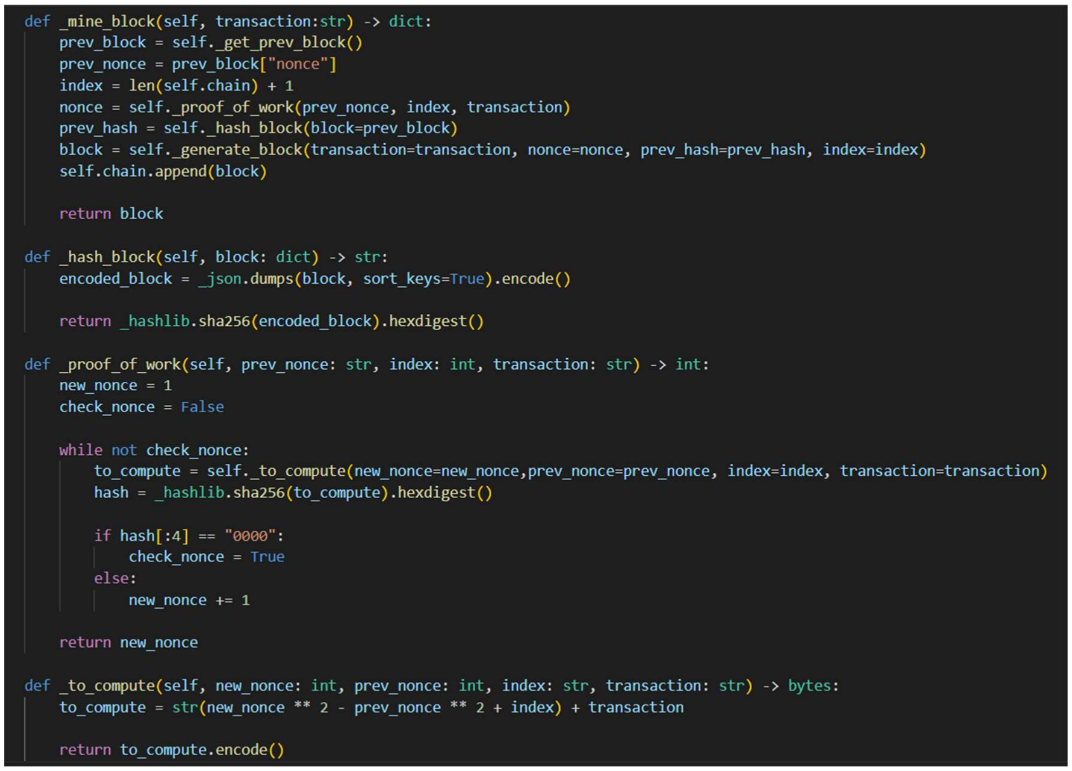
The blockchain modules starts with self-initialization that produces the genesis block(the first block in a blockchain). This can be accomplished by utilizing the *_generate_block* function.



```
import datetime as _datetime
import hashlib as _hashlib
import json as _json

class Blockchain:

    def __init__(self) -> None:
        self.chain = list()
        genesis_block = self._generate_block(transaction="Genesis block", nonce=1, prev_hash="0", index=1)
        self.chain.append(genesis_block)

    def _generate_block(self, transaction: str, nonce: int, prev_hash: str, index: int) -> dict:
        block = {
            "index": index,
            "timestamp": str(_datetime.datetime.now()),
            "transaction": transaction,
            "nonce": nonce,
            "prev_hash": prev_hash,
        }
        return block
```

Figure 4.9 : blockchain init and generation in blockchain.py

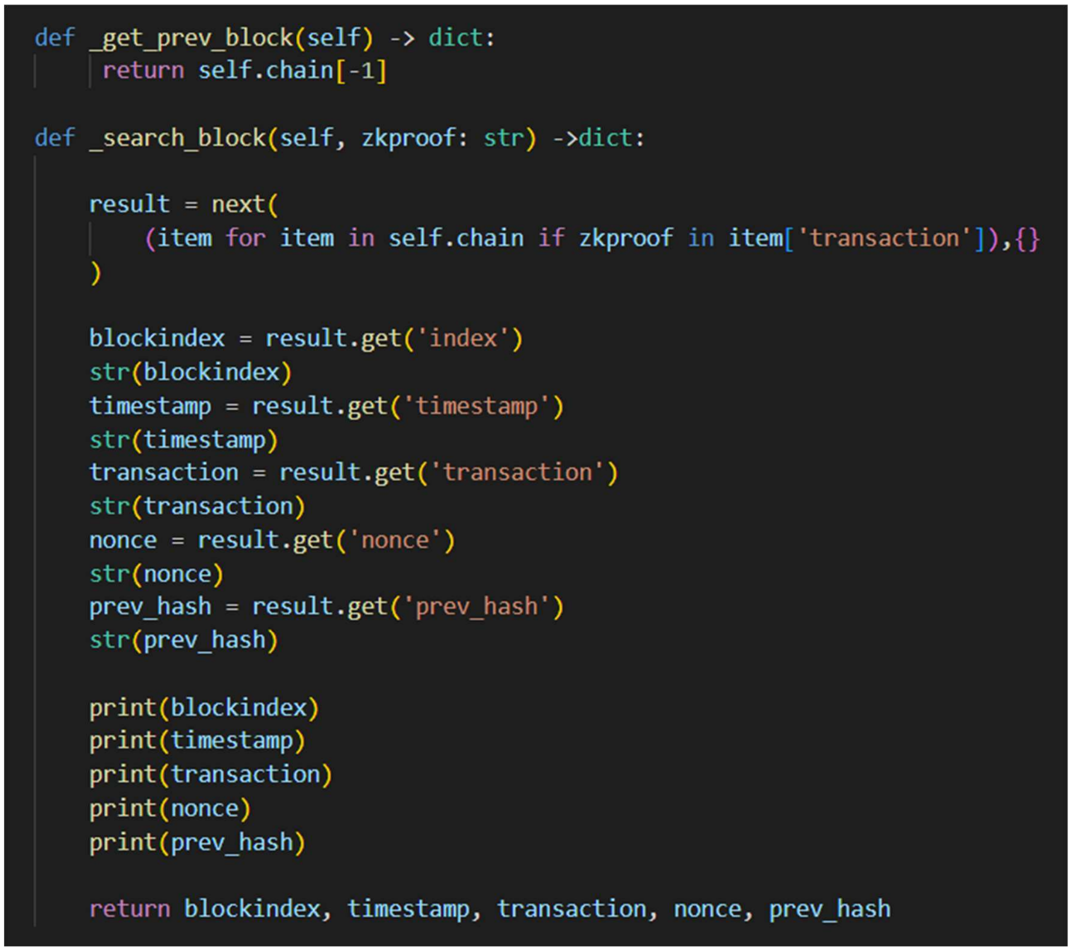
The functions for mining a block, hashing the block and the additional backend operations are also shown here.



```
def _mine_block(self, transaction:str) -> dict:
    prev_block = self._get_prev_block()
    prev_nonce = prev_block["nonce"]
    index = len(self.chain) + 1
    nonce = self._proof_of_work(prev_nonce, index, transaction)
    prev_hash = self._hash_block(block=prev_block)
    block = self._generate_block(transaction=transaction, nonce=nonce, prev_hash=prev_hash, index=index)
    self.chain.append(block)

    return block

def _hash_block(self, block: dict) -> str:
    encoded_block = _json.dumps(block, sort_keys=True).encode()

    return _hashlib.sha256(encoded_block).hexdigest()

def _proof_of_work(self, prev_nonce: str, index: int, transaction: str) -> int:
    new_nonce = 1
    check_nonce = False

    while not check_nonce:
        to_compute = self._to_compute(new_nonce=new_nonce,prev_nonce=prev_nonce, index=index, transaction=transaction)
        hash = _hashlib.sha256(to_compute).hexdigest()

        if hash[:4] == "0000":
            check_nonce = True
        else:
            new_nonce += 1

    return new_nonce

def _to_compute(self, new_nonce: int, prev_nonce: int, index: str, transaction: str) -> bytes:
    to_compute = str(new_nonce ** 2 - prev_nonce ** 2 + index) + transaction

    return to_compute.encode()
```

Figure 4.10 : Mining operations in blockchain.py

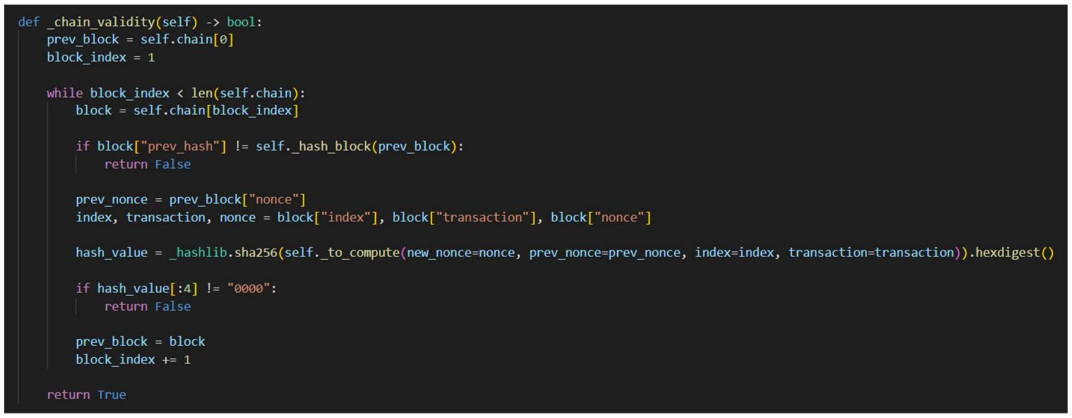
Additional functions to interact with the blockchain like getting the last block and search through the blockchain are also available here.



```
def _get_prev_block(self) -> dict:
    return self.chain[-1]

def _search_block(self, zkproof: str) ->dict:

    result = next(
        (item for item in self.chain if zkproof in item['transaction']),{}
    )

    blockindex = result.get('index')
    str(blockindex)
    timestamp = result.get('timestamp')
    str(timestamp)
    transaction = result.get('transaction')
    str(transaction)
    nonce = result.get('nonce')
    str(nonce)
    prev_hash = result.get('prev_hash')
    str(prev_hash)

    print(blockindex)
    print(timestamp)
    print(transaction)
    print(nonce)
    print(prev_hash)

    return blockindex, timestamp, transaction, nonce, prev_hash
```

Figure 4.11 : Additional functions in blockchain.py

A function that validates and ensures chain integrity by checking every block is also available for use if needed.



```python
def _chain_validity(self) -> bool:
    prev_block = self.chain[0]
    block_index = 1

    while block_index < len(self.chain):
        block = self.chain[block_index]

        if block["prev_hash"] != self._hash_block(prev_block):
            return False

        prev_nonce = prev_block["nonce"]
        index, transaction, nonce = block["index"], block["transaction"], block["nonce"]

        hash_value = _hashlib.sha256(self._to_compute(new_nonce=nonce, prev_nonce=prev_nonce, index=index, transaction=transaction)).hexdigest()

        if hash_value[:4] != "0000":
            return False

        prev_block = block
        block_index += 1

    return True
```

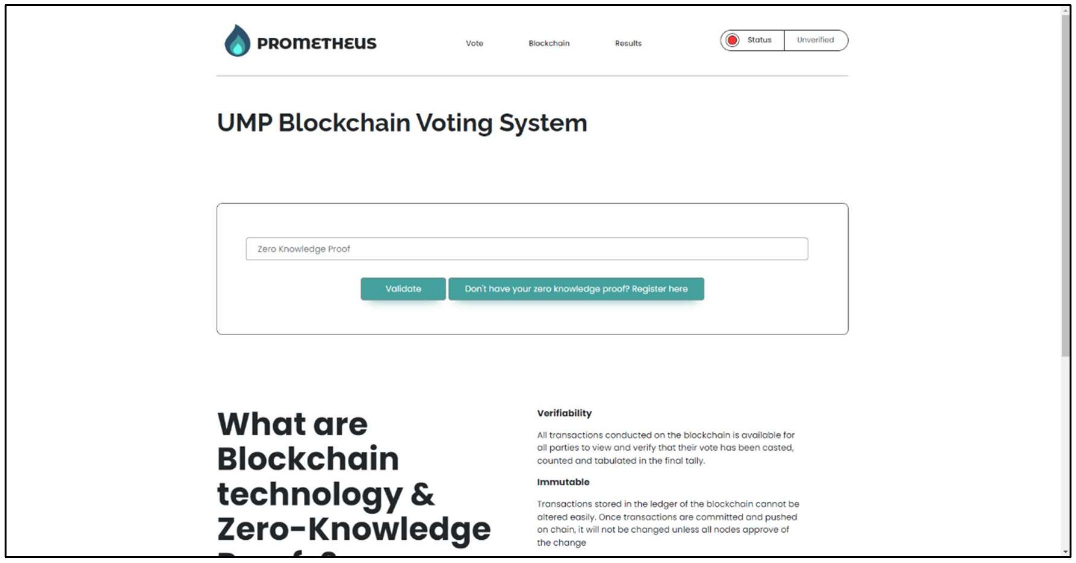
Figure 4.12 : Chain validity function in blockchain.py

4.4 Frontend

The frontend of this blockchain voting project is written entirely in HTML, CSS, and JS, utilizing Jinja2 templates to display the webpages. The blockchain voting system is named and branded as Prometheus, with the inspiration coming from the Titan Prometheus who gave humanity fire in Greek mythology, ushering a new era of innovation and knowledge.

4.4.1 Index

The index page serves as the homepage for and has short paragraphs describing blockchain technology and zero knowledge proofs. It is also the page to submit the ZKProof statement for validation before proceeding to vote.



UMP Blockchain Voting System

What are Blockchain technology & Zero-Knowledge

Verifiability

All transactions conducted on the blockchain is available for all parties to view and verify that their vote has been casted, counted and tabulated in the final tally.

Immutable

Transactions stored in the ledger of the blockchain cannot be altered easily. Once transactions are committed and pushed on chain, it will not be changed unless all nodes approve of the change

Figure 4.13 : Index page of Prometheus

4.4.2 Registration

The registration page is where users sign up to vote and obtain their ZKProof statement.

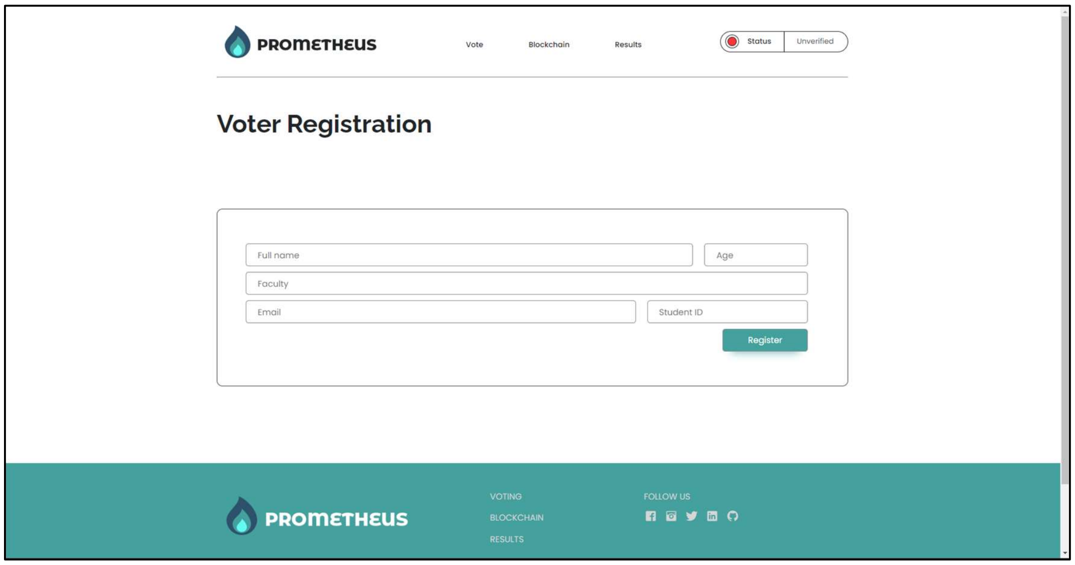


Figure 4.14: Registration page of Prometheus

4.4.3 Voting

The voting page displays all available candidates for the election and submits the vote to the blockchain once the Vote button is pressed.

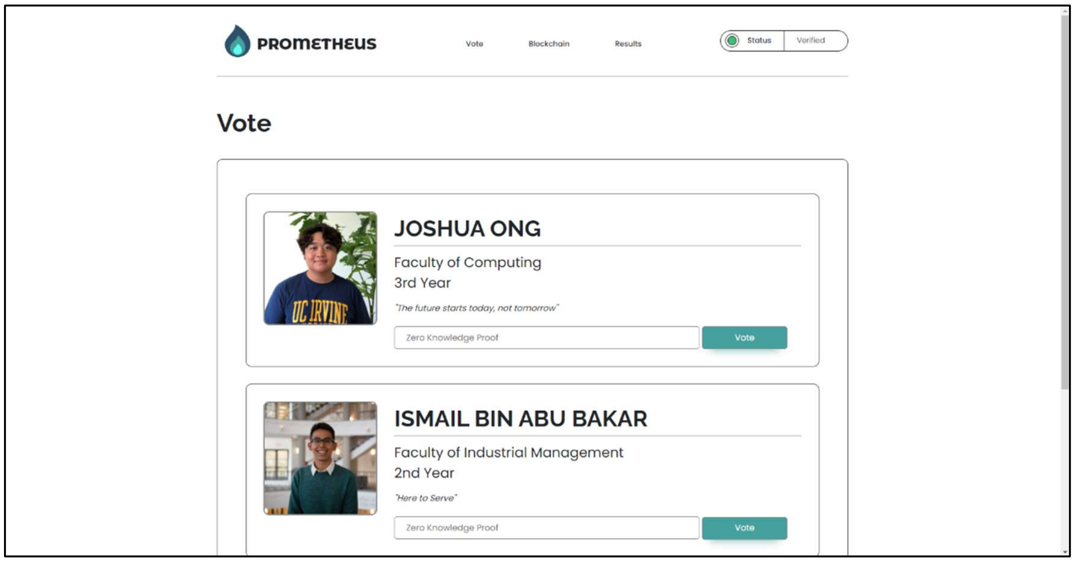


Figure 4.15 : Voting page of Prometheus

4.4.4 Blockchain

The blockchain page allows users to search for their vote within the blockchain or download the entire blockchain file to verify for themselves.

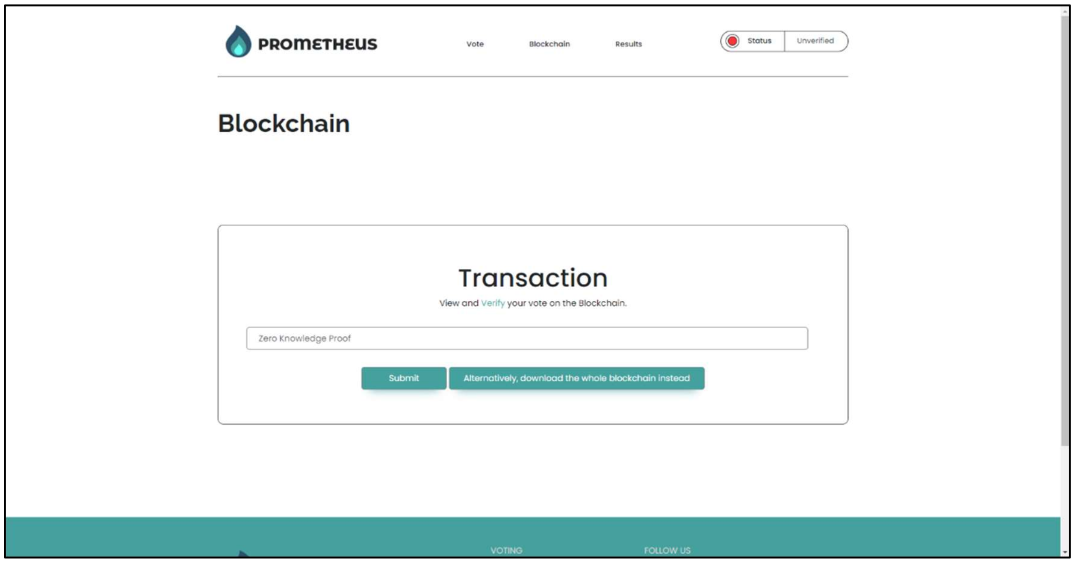


Figure 4.16 : Blockchain page of Prometheus

4.4.5 Results

This page hides the results of the election until the Reveal button is pressed, then it announces the winner.

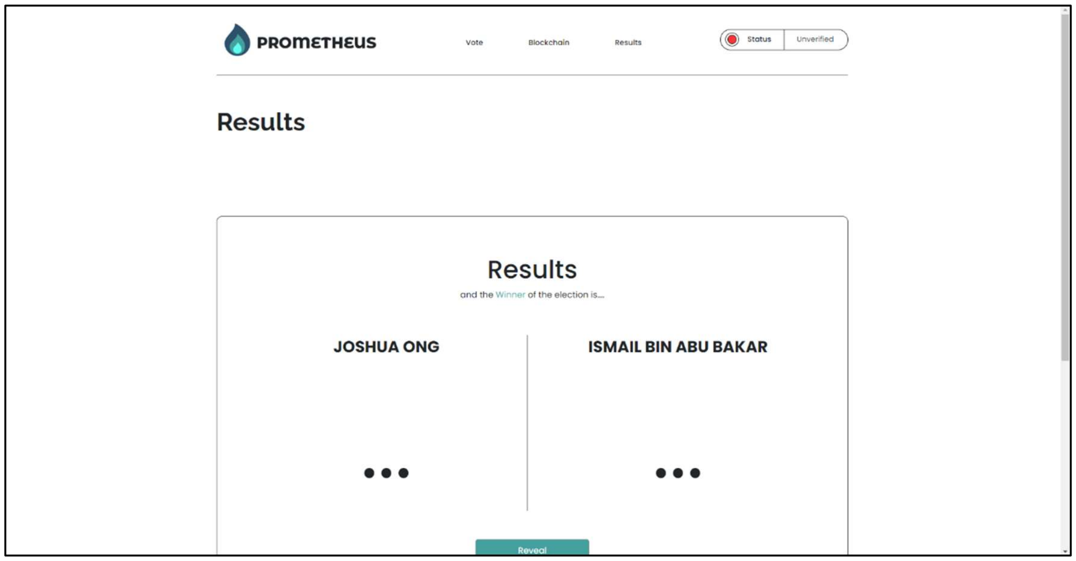


Figure 4.17 : Result page of Prometheus

4.5 Result

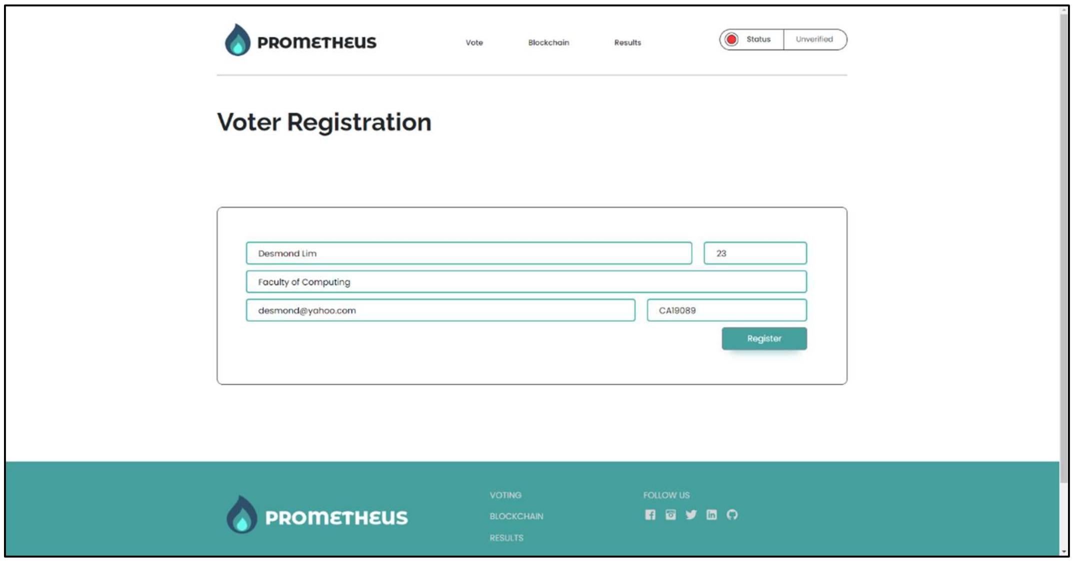
The result section demonstrates how a user will proceed through the system in order to register, vote and verify their casted vote. Outputs produced during the execution of the process is shown in this section.

4.5.1 Flow of the System

The full flow of the Prometheus blockchain voting system, from registration to result announcement, can be seen in the following figures.

4.5.1.1 Register

The user will first register as a voter using their name, age, faculty, email, and StudentID. Once submitted, the inputs will be hashed along with a nonce in order to produce the ZKProof statement that the user will use to vote later on. In the background, the system runs through the ZKProof modules for the first time, passing the ZKProof statement once through the server to produce a token on the server side, which will later be used for verification.

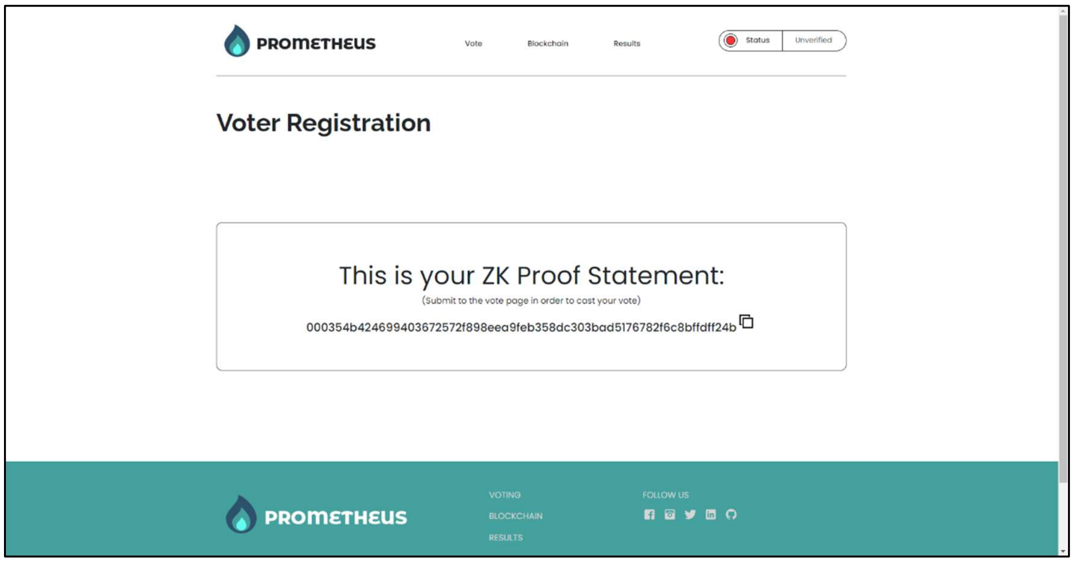


Voter Registration

Desmond Lim
23
Faculty of Computing
desmond@yahoo.com
CA19089

Figure 4.18 Voting Registration step in Prometheus

Once the process is done, the system will redirect to another page where the user can copy their unique ZKProof statement to the clipboard by clicking the statement.



Voter Registration

This is your ZK Proof Statement:

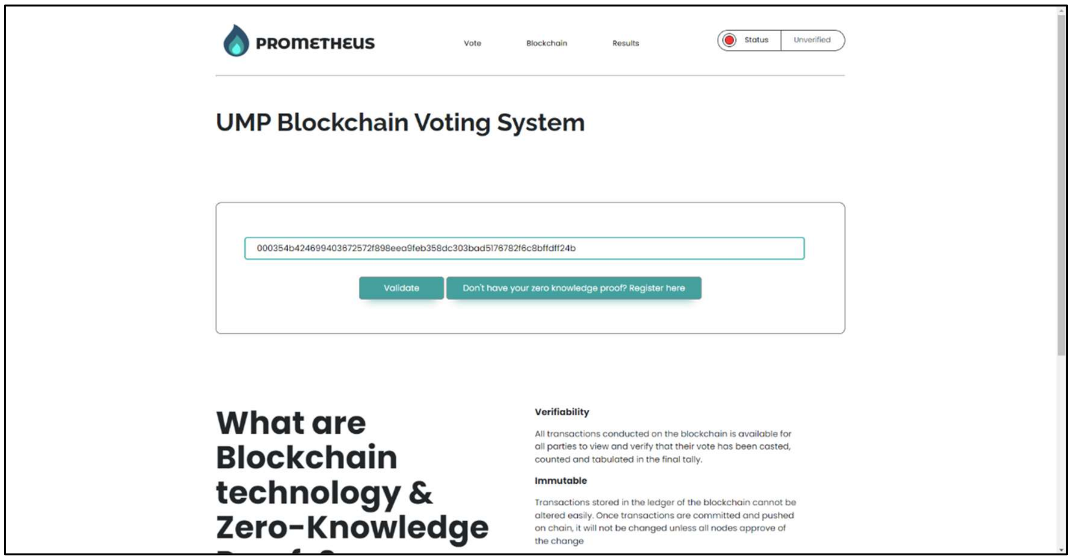
(Submit to the vote page in order to cast your vote)

000354b424699403672572f898eea9feb358dc303bad5176782f6c8bffdff24b

Figure 4.19 Obtain ZKProof statement in Prometheus

4.5.1.2 Submit ZKProof for Verification

Once the ZKProof statement has been copied, user can head back to the main page to start the voting process. Submitting their ZKProof using the input box, the system will verify their ZKProof statement using the token on the server side.



UMP Blockchain Voting System

000354b424699403672572f898eea9feb358dc303bad5176782f6c8bffdff24b

What are Blockchain technology & Zero-Knowledge

Verifiability

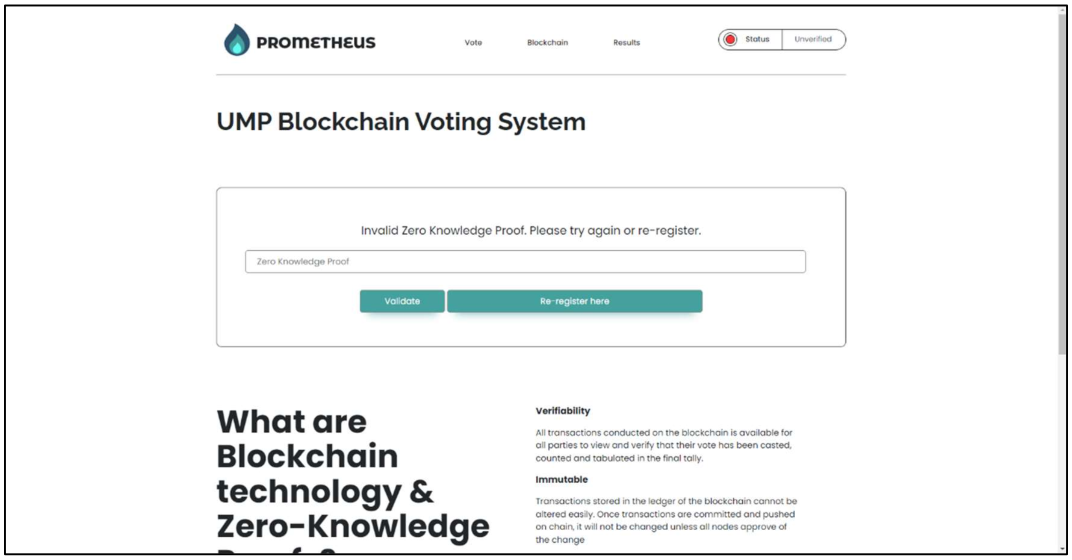
All transactions conducted on the blockchain is available for all parties to view and verify that their vote has been casted, counted and tabulated in the final tally.

Immutable

Transactions stored in the ledger of the blockchain cannot be altered easily. Once transactions are committed and pushed on chain, it will not be changed unless all nodes approve of the change

Figure 4.20 Submit ZKProof in Prometheus

If successful, users will be redirected to the voting page to start submitting their votes. If the verification process ends in failure due to a bad phrase, users will be presented with the following screen and asked to try registering again.



UMP Blockchain Voting System

Invalid Zero Knowledge Proof. Please try again or re-register.

What are Blockchain technology & Zero-Knowledge

Verifiability

All transactions conducted on the blockchain is available for all parties to view and verify that their vote has been casted, counted and tabulated in the final tally.

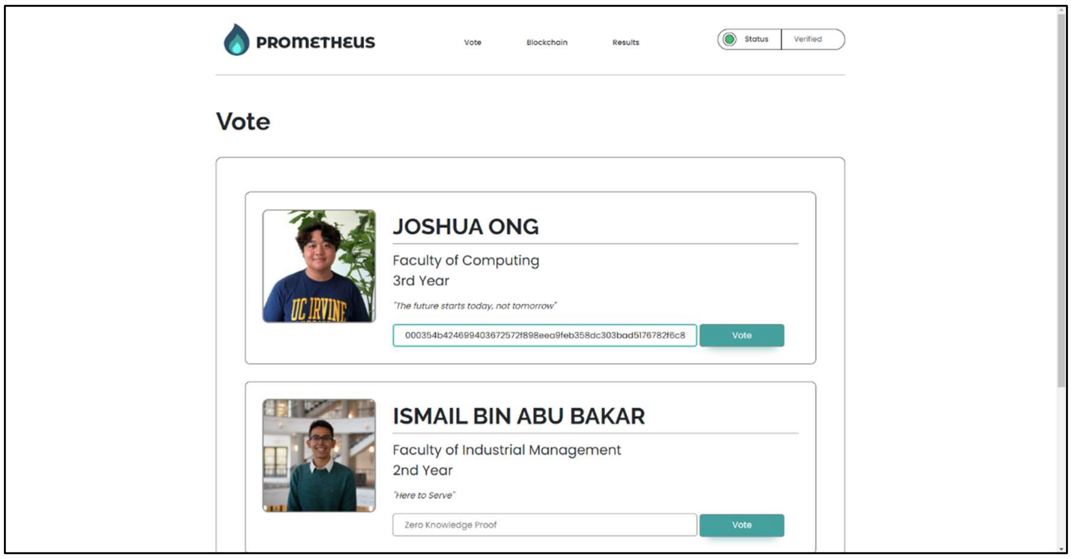
Immutable

Transactions stored in the ledger of the blockchain cannot be altered easily. Once transactions are committed and pushed on chain, it will not be changed unless all nodes approve of the change

Figure 4.21 Failed Verification in Prometheus

4.5.1.3 Voting

Once users have successfully passed verification, they can then cast their vote for their desired candidate. When submitting the vote, users will be required to input their ZKProof statement again as it will be included in the block data for the vote and will be used to find the user’s transaction for verification.



Vote

JOSHUA ONG

Faculty of Computing
3rd Year

"The future starts today, not tomorrow"

000354b424699403672572f898eea9feb358dc303bad5176782f6c8

ISMAIL BIN ABU BAKAR

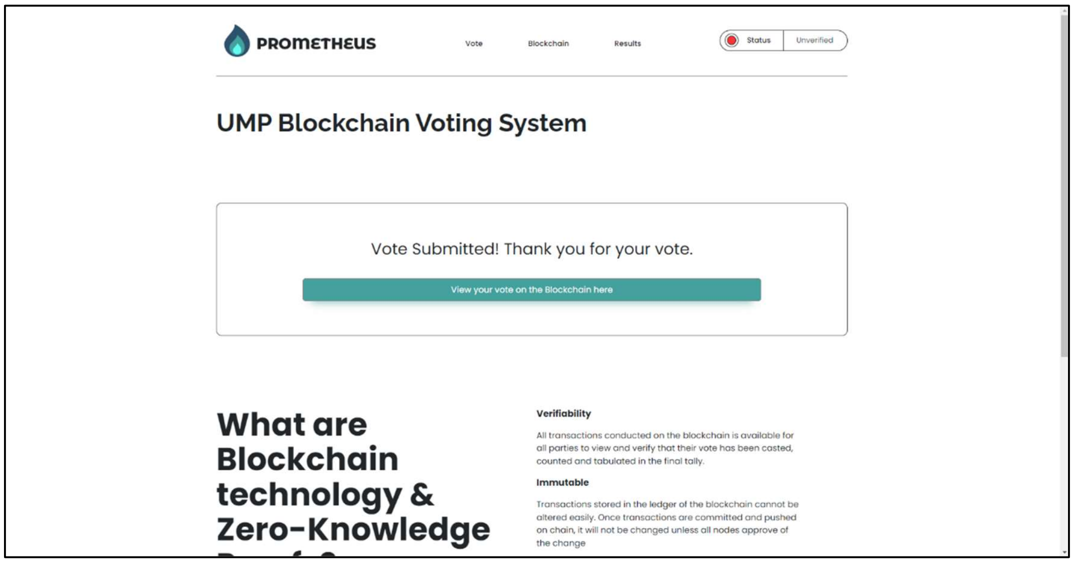
Faculty of Industrial Management
2nd Year

"Here to Serve"

Zero Knowledge Proof

Figure 4.22 Voting process in Prometheus

Once the vote has been submitted, user will be redirected to a new page indicating that their vote has been submitted.



UMP Blockchain Voting System

Vote Submitted! Thank you for your vote.

What are Blockchain technology & Zero-Knowledge

Verifiability

All transactions conducted on the blockchain is available for all parties to view and verify that their vote has been casted, counted and tabulated in the final tally.

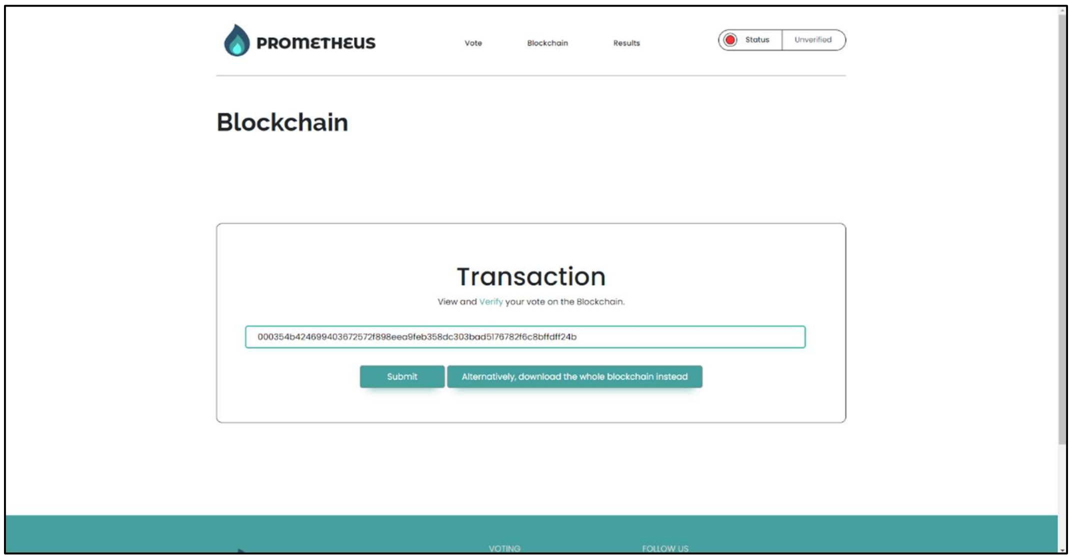
Immutable

Transactions stored in the ledger of the blockchain cannot be altered easily. Once transactions are committed and pushed on chain, it will not be changed unless all nodes approve of the change

Figure 4.23 Vote Submitted in Prometheus

4.5.1.4 Check Blockchain

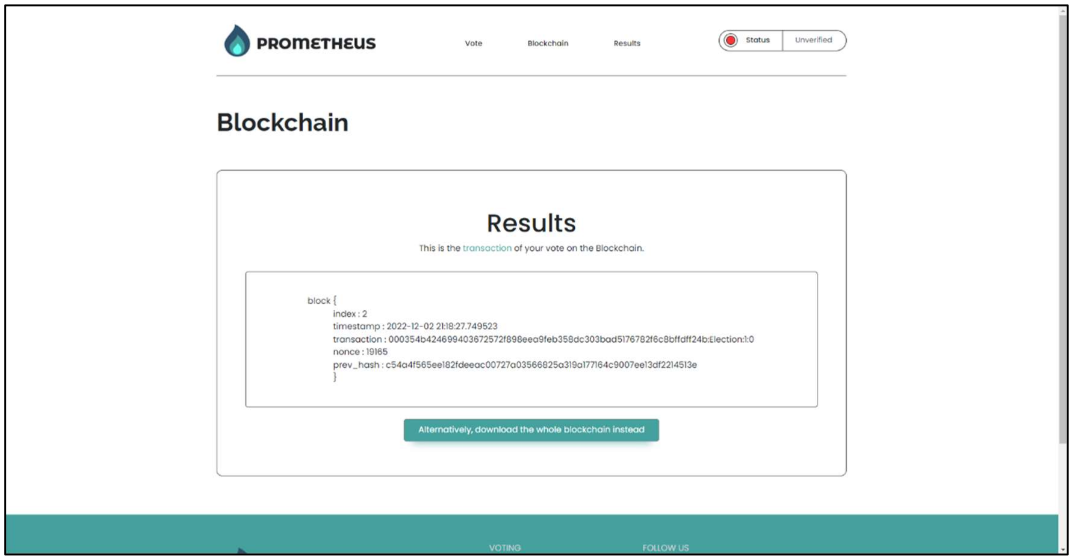
By pressing the “View your vote on the blockchain here” button or navigating to the blockchain tab, users will be presented with the following page where they can submit their ZKProof statement to find their vote in the blockchain.


PROMETHEUS
Vote
Blockchain
Results
Status
Unverified
Blockchain
Transaction
View and Verify your vote on the Blockchain.
Submit
Alternatively, download the whole blockchain instead
VOTING
FOLLOW US

000354b424699403672572f898eea9feb358dc303bad5176782f6c8bffdff24b

Figure 4.24 Search Blockchain in Prometheus

Once submitted, the system will search through the blockchain for the block containing the ZKProof statement in the transaction value. Once found, the data will be presented as follows:



Blockchain

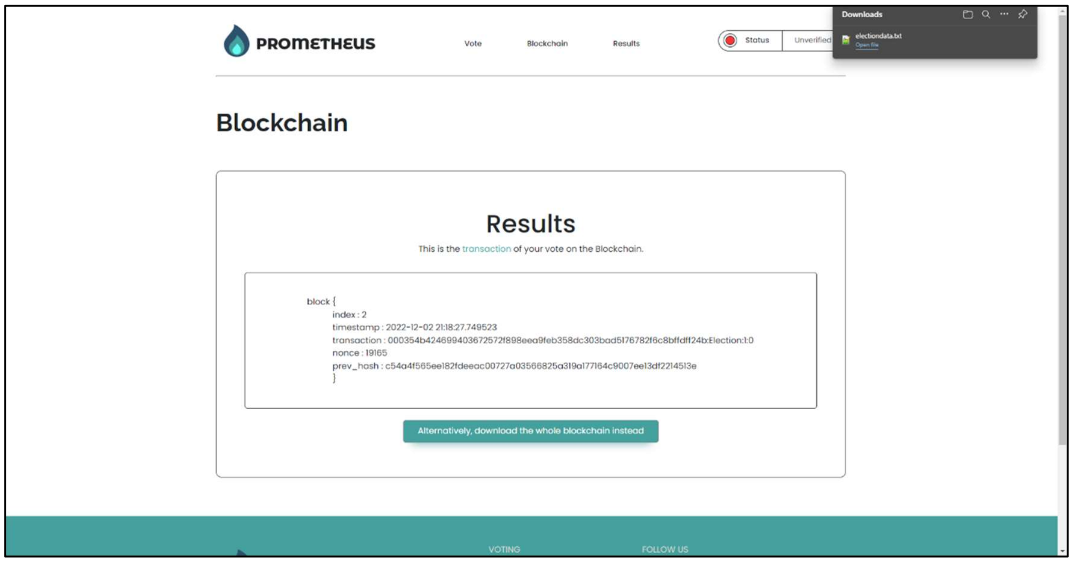
Results

This is the transaction of your vote on the Blockchain.

```
block {
        index : 2
        timestamp : 2022-12-02 21:18:27.749523
        transaction : 000354b424699403672572f898eea9feb358dc303bad5176782f6c8bffdff24b:Election:1:0
        nonce : 19165
        prev_hash : c54a4f565ee182fdeeac00727a03566825a319a177164c9007ee13df2214513e
        }
```

Figure 4.25 Transaction Data in Prometheus

The user can also press the “Alternatively, download the whole blockchain instead” button to download the whole blockchain data to verify the whole election for themselves.



Blockchain

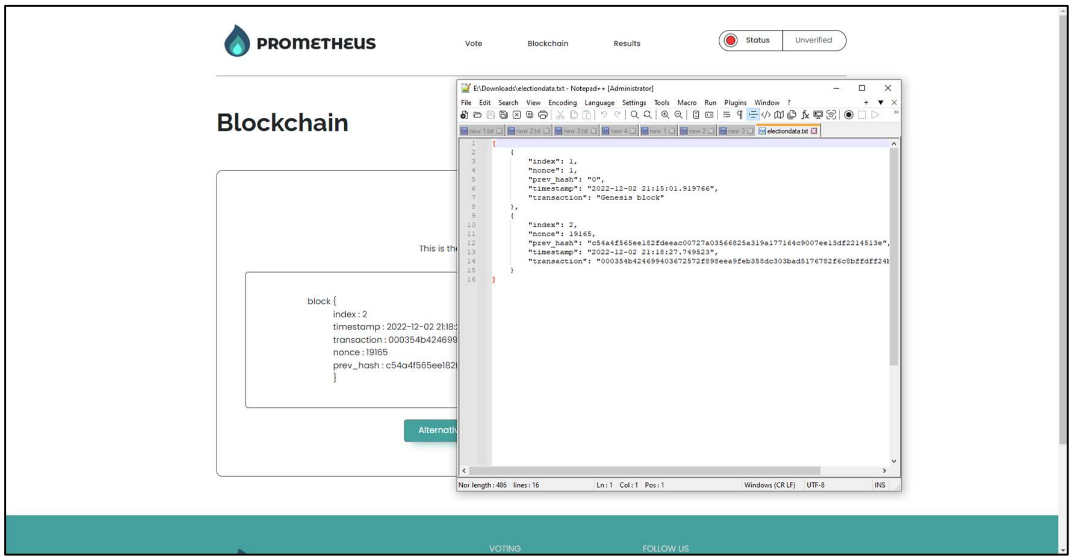
Results

This is the transaction of your vote on the Blockchain.

```
block {
        index : 2
        timestamp : 2022-12-02 21:18:27.749523
        transaction : 000354b424699403672572f898eea9feb358dc303bad5176782f6c8bffdff24b:Election:1:0
        nonce : 19165
        prev_hash : c54a4f565ee182fdeeac00727a03566825a319a177164c9007ee13df2214513e
        }
```

Figure 4.26 Downloaded Blockchain in Prometheus

The downloaded blockchain data will be stored in a text file and the data contained inside is formatted as follows:


PROMETHEUS
Vote
Blockchain
Results
Status
Unverified
Blockchain
E:\Downloads\electiondata.txt - Notepad++ [Administrator]
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window ?
electiondata.txt
Alternati
VOTING
FOLLOW US

```
[
    {
        "index": 1,
        "nonce": 1,
        "prev_hash": "0",
        "timestamp": "2022-12-02 21:15:01.919766",
        "transaction": "Genesis block"
    },
    {
        "index": 2,
        "nonce": 19165,
        "prev_hash": "c54a4f565ee182fdeeac00727a03566825a319a177164c9007ee13df2214513e",
        "timestamp": "2022-12-02 21:18:27.749523",
        "transaction": "000354b424699403672572f898eea9feb358dc303bad5176782f6c8bffdff24b
    }
]
```

This is th

```
block {
        index : 2
        timestamp : 2022-12-02 21:18:
        transaction : 000354b424699
        nonce : 19165
        prev_hash : c54a4f565ee182f
        }
```

```
Nor length : 486   lines : 16        Ln : 1   Col : 1   Pos : 1        Windows (CR LF)   UTF-8        INS
```

Figure 4.27 Blockchain Text File in Prometheus

4.5.1.5 Results

Once the election has ended, users can navigate to the results tab and be greeted with the following screen:

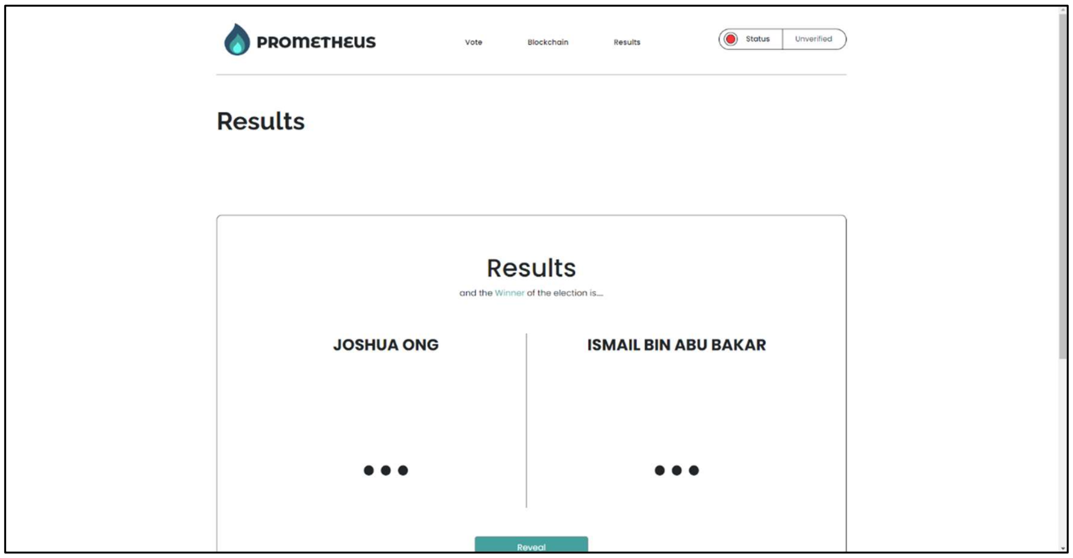


Figure 4.28 Result page in Prometheus

By pressing the reveal button, the results of the election will be displayed. This can be observed in Figure 4.29.

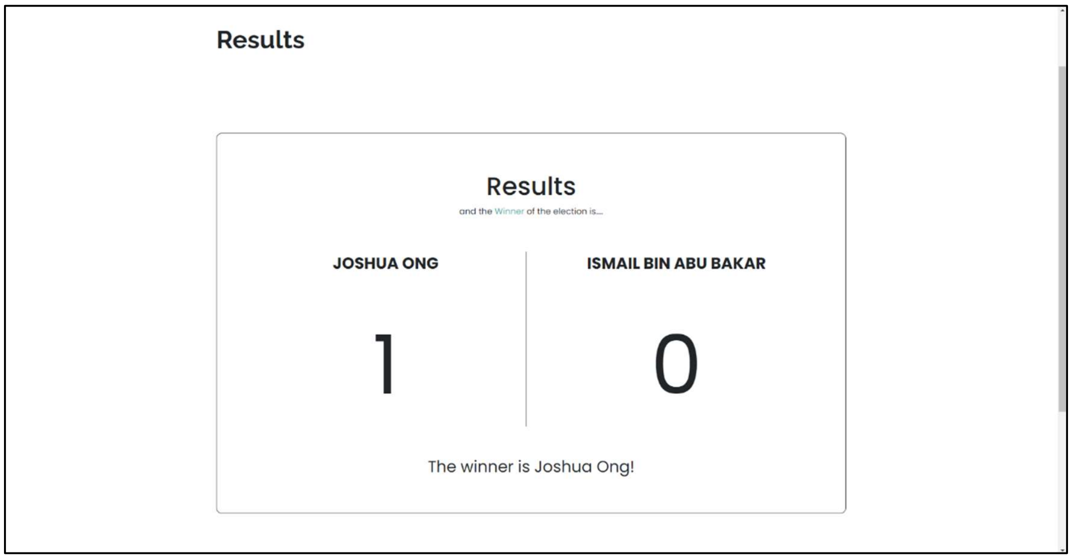

Results
Results
and the Winner of the election is...
JOSHUA ONG
ISMAIL BIN ABU BAKAR
1
0
The winner is Joshua Ong!

Figure 4.29 : Result Show in Prometheus

4.5.1.6 Admin Login

By navigating to the [localhost]/admin, users can access the admin login page. Once their identity has been successfully verified using the admin username and password, they can then gain access to the admin panel.

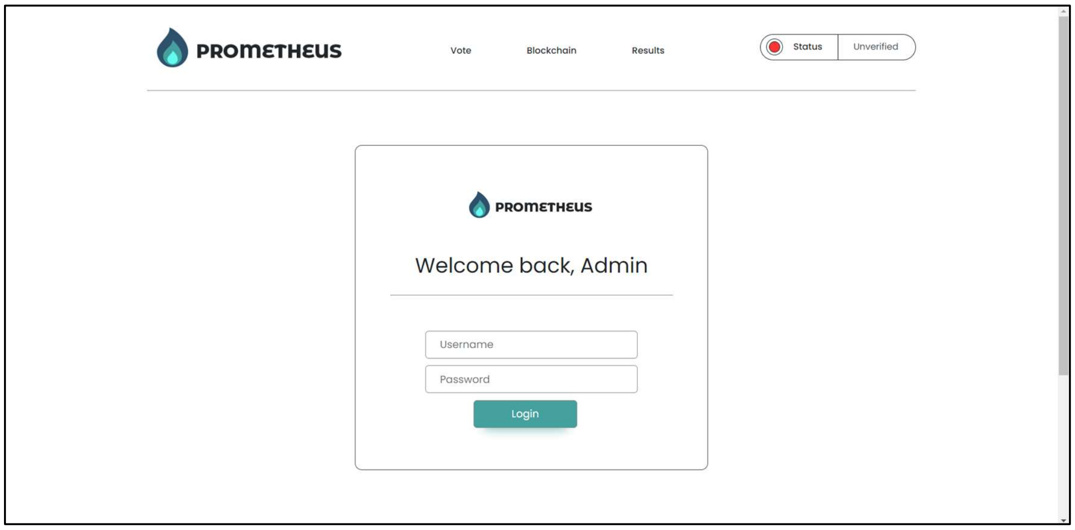


Figure 4.30 : Admin Login in Prometheus

4.5.1.7 Admin Panel

Once logged in, the user then has access to the following functions: starting an election, ending an election, adding candidates to the election, and check the validity of the blockchain of the current election. Based on previous feedback, the UI for admin pages have received slight changes to the footer colour and status indicator to better differentiate it from pages normal users are able to access.

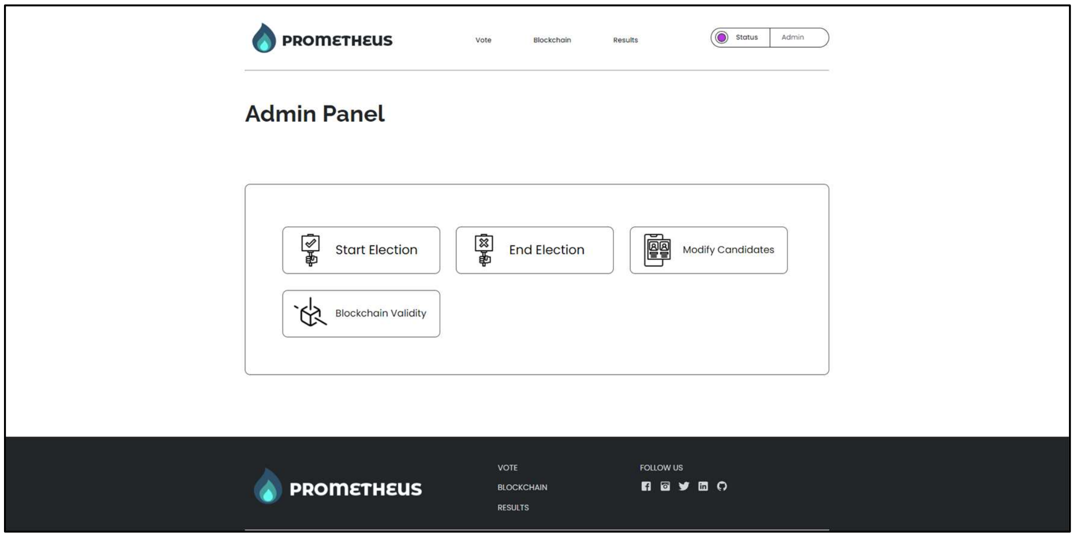
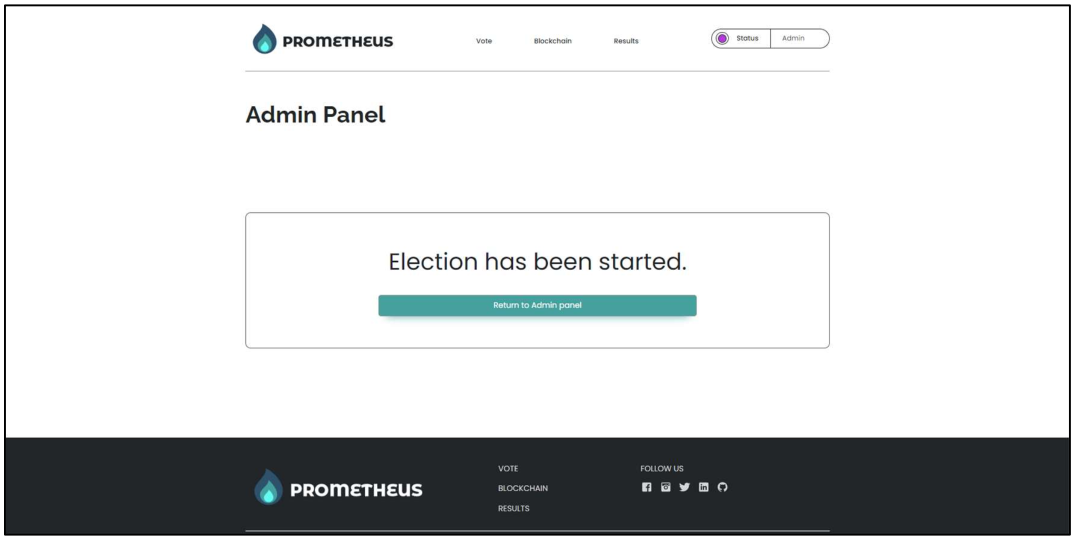


Figure 4.31 : Admin Panel in Prometheus

4.5.1.8 Start Election

When selected, a new election is started. It should be noted that elections cannot be started if candidate information is not provided.



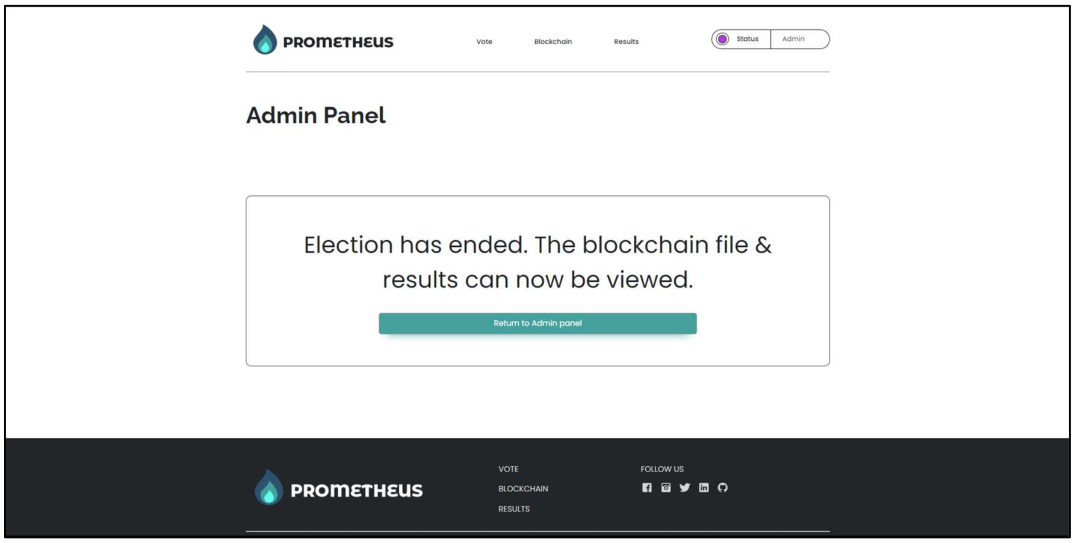
Admin Panel

Election has been started.

Figure 4.32 : Election Start in Prometheus

4.5.1.9 End Election

When an election has been ended, users will be able to gain access to the blockchain and results page. The entire blockchain file can also be downloaded should the user gains interest in verifying the whole election's transaction.



Admin Panel

Election has ended. The blockchain file & results can now be viewed.

Figure 4.33 : Election End in Prometheus

4.5.1.10 Modify/Add Candidates

In the modify candidates page, the administrator can add the profile pictures of the candidates, along with their name, current year of study, faculty and election slogan. Once added, these changes will be reflected in the voting page accessed by the users.

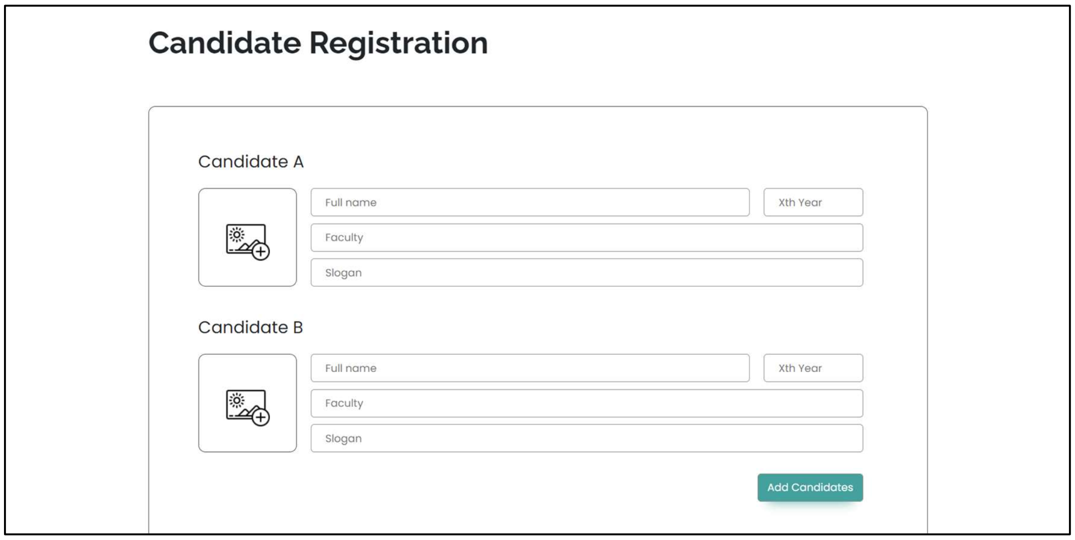


Figure 4.34 Modify Candidate in Prometheus

4.5.1.11 Chain Validity

During an election, the administrator can use the chain validity page to query and test whether the chain is still valid (has not been tampered). If the chain has not been tampered with, the validity will return a true value.

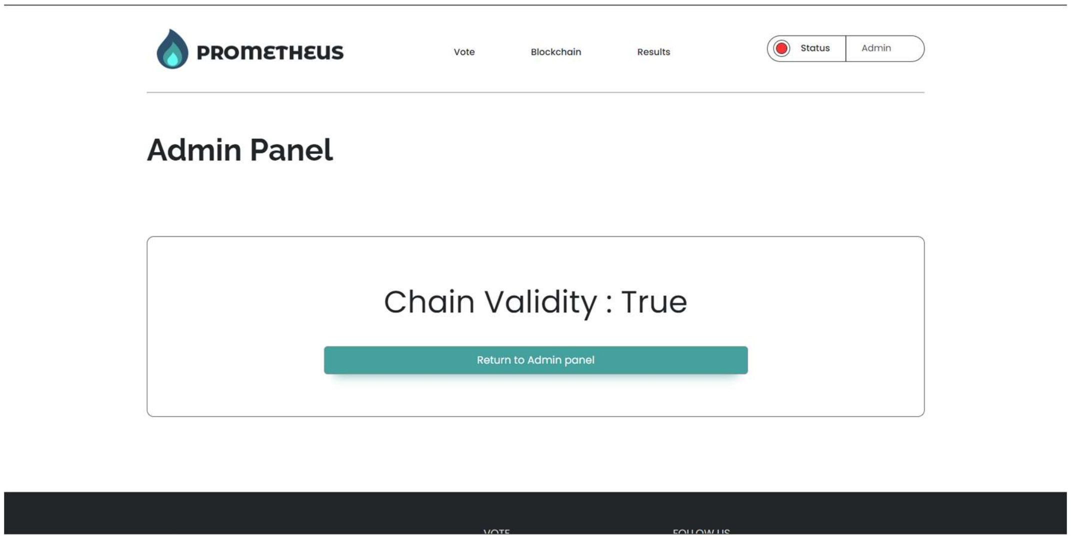


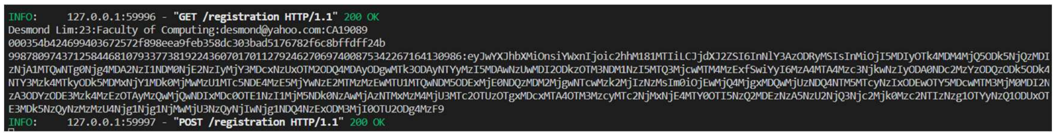
Figure 4.35 Chain Validity in Prometheus

4.5.2 Outputs by the System

In the backend, the system would produce the following information that would not be seen by the users in the frontend.

4.5.2.1 ZKProof Statement

When users submit their information to be validated and return as the ZKProof statement, the backend shows the following:



```
INFO:     127.0.0.1:59996 - "GET /registration HTTP/1.1" 200 OK
Desmond Lim:23:Faculty of Computing:desmond@yahoo.com:CA19089
000354b424699403672572f898eea9feb358dc303bad5176782f6c8bffdff24b
99878097437125844681079337738192243607017011279246270697400875342267164130986:eyJwYXJhbXMiOnsiYWxnIjoic2hhM181MTIiLCJjdXJ2ZSI6InNlY3AzODRyMSIsInMiOjI5MDIyOTk4MDM4MjQ5ODk5NjQzMDI
zNjA1MTQwNTg0Njg4MDA2NzI1NDM0NjE2NzIyMjY3MDcxNzUxOTM2ODQ4MDAyODgwMTk3ODAyNTYyMzI5MDAwNzUwMDI2ODkzOTM3NDM1NzI5MTQ3MjcwMTM4MzExfSwiYyI6MzA4MTA4Mzc3NjkwNzIyODA0NDc2MzYzODQzODk5ODk4
NTY3Mzk4MTkyODk5MDMxNjY1MDk0MjMwMzU1MTc5NDE4MzE5MjYwNzE2MTMzMzEwMTU1MTQwNDM5ODExMjE0NDQzMDM2MjgwNTcwMzk2MjIzNzMsIm0iOjEwMjQ4MjgxMDQwMjUzNDQ4NTM5MTcyNzIxODEwOTY5MDcwMTM3MjM0MDI2N
zA3ODYzODE3Mzk4MzEzOTAyMzQwMjQwNDIxMDc0OTE1NzI1MjM5NDk0NzAwMjAzNTMxMzM4MjU3MTc2OTUzOTgxMDcxMTA4OTM3MzcyMTc2NjMxNjE4MTY0OTI5NzQ2MDEzNzA5NzU2NjQ3Njc2Mjk0Mzc2NTIzNzg1OTYyNzQ1ODUxOT
E3MDk5NzQyNzMzMzU4Njg1Njg1NjMwMjU3NzQyNjIwNjg1NDQ4NzExODM3MjI0OTU2ODg4MzF9
INFO:     127.0.0.1:59997 - "POST /registration HTTP/1.1" 200 OK
```

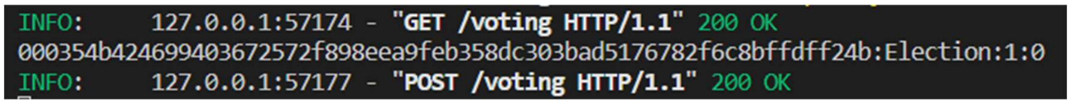
Figure 4.36 ZKProof backend in Prometheus

The first line after the GET statement is the user's information that has been appended into a string, which is then used to produce the next line, the ZKProof Statement. The ZKProof statement is then used by the ZKProof client to produce a token with the ZKProof server, which is the long string of characters before the POST statement.

4.5.2.2 Vote Submission

When users submit their vote, they provide their ZKProof statement as well. The system then appends their ZKProof statement to the transaction for the vote in the following syntax:

(*zkproofstatement*):Election:(*Candidate_A votecount*):(*Candidate_B votecount*)

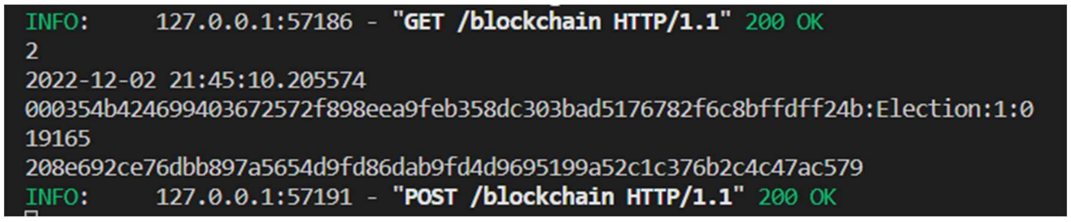


```
INFO:     127.0.0.1:57174 - "GET /voting HTTP/1.1" 200 OK
000354b424699403672572f898eea9feb358dc303bad5176782f6c8bffdff24b:Election:1:0
INFO:     127.0.0.1:57177 - "POST /voting HTTP/1.1" 200 OK
```

Figure 4.37 Vote data in Prometheus

4.5.2.3 Blockchain Search

When users search for their vote transaction in the blockchain, they first submit their ZKProof statement. This statement is the used to search for blocks where the transaction data contains the statement. Once found, the data in that specific block is returned, containing the index, timestamp, transaction data, nonce, and previous hash of that specific block.

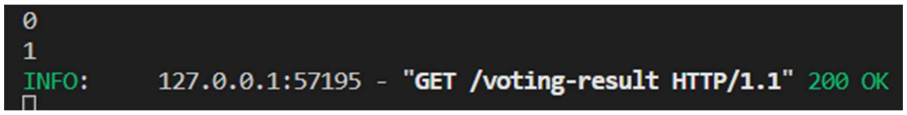


```
INFO:     127.0.0.1:57186 - "GET /blockchain HTTP/1.1" 200 OK
2
2022-12-02 21:45:10.205574
000354b424699403672572f898eea9feb358dc303bad5176782f6c8bffdff24b:Election:1:0
19165
208e692ce76dbb897a5654d9fd86dab9fd4d9695199a52c1c376b2c4c47ac579
INFO:     127.0.0.1:57191 - "POST /blockchain HTTP/1.1" 200 OK
```

Figure 4.38 Blockchain search in Prometheus

4.5.2.4 Vote Result

When the results page is accessed, the data of the last block in the blockchain is retrieved. From the transaction data, the system retrieves the vote count of both candidates by separating the data by the colon for the two last values in the transaction. The vote count is then passed to the results page when accessed, ready to display using JavaScript.



```
0
1
INFO:     127.0.0.1:57195 - "GET /voting-result HTTP/1.1" 200 OK
```

Figure 4.39 Vote Results in Prometheus

4.6 Discussion

Discussions are done to contemplate on current inadequacies and possibly obtain a breakthrough through feedback. This section with cover the reasonings behind the decisions made during the development of this project and any potential limitations the project experiences in its current stage which require improvement. Results from running the user acceptance test are also summarized in this section.

4.6.1 Justifications

4.6.1.1 The Use of Web Application

Despite not being one of the main objectives, a necessary feature that is heavily place upon e-voting systems is accessibility. The decision has been made to make a web application instead of a mobile application. Although most individuals these days has a smartphone, the ability to access the system using both smartphone and desktop computers increases reach immensely. By making the website responsive, this is a non-issue.

4.6.1.2 Python

While there are many other languages that can be used to write blockchain applications and ZKProof libraries, Python remains one of the most supported languages with extensive libraries to import. Due to previous experience working with the language, it was chosen as the main programming language for this project.

4.6.1.3 Why Proof of Work

Proof of work blockchain systems are easier to implement and does not require extensive configurations with regards to nodes, staking, staking rewards and the works.

As a concept, most people are familiar with Bitcoin and explaining proof of work systems would be easier compared to proof of stake systems.

4.6.1.4 Limitations of Current Implementation

In the current system, the blockchain does not have a memory pool to store transactions that are pending. If too many requests/votes are made at an exact moment, forking might occur and cause votes to be lost. For the ZKProof statement verification, the system currently can hold a single session(token) for a single user only, meaning any new registration would reset the token and cause the previous registered voter to be unable to verify their ZKProof statement.

4.6.2 User Acceptance Test

After completing the system, 30 university students are selected at random and asked to test the usability and the ease of navigation of the system. Taking their response, changes and updates can then be made to the system. An example of a change made was users stating that the administrative side of the system was hard to recognize, thus changes was made to the UI of the admin pages for better distinction.

Test Case ID	**Test Case Name**	**Status (Pass/Fail)**
TC-PROM-01	Admin Login	PASS
TC-PROM-02	Add Candidates	PASS
TC-PROM-03	Start Election	PASS
TC-PROM-04	Register as Voter	PASS
TC-PROM-05	Verify ZK Proof	PASS
TC-PROM-06	Vote	PASS
TC-PROM-07	End Election	PASS
TC-PROM-08	View Blockchain Transaction	PASS
TC-PROM-09	Download Blockchain	PASS
TC-PROM-10	View Results	PASS

Table 4.1: UAT Summary for Prometheus

In summary, the system passed the user acceptance test with flying colours and all modules worked as intended. Additional comments were asked of respondents regarding their understanding and perception towards these emerging technologies, which received positive feedback and was generally regarding as innovative and necessary for technology to progress further.

CHAPTER 5

CONCLUSION

5.1 Introduction

For everything that has a beginning, there must also come an end. With the purpose of providing accessibility, security, and verifiability to voting systems, the implementation of blockchain technology and zero-knowledge proofs into the voting system have successfully done so. Developed using Python, FastAPI, HTML, CSS, and JavaScript in accordance with the DSDM framework, the web-based system can be used on all devices with an internet connection. Evaluating the system based on the responses of university students, the system passes all modules checks for design and functionality with modification made to the UI based on feedback. Chapter 5 wraps up the project and revisits the objectives of the project, discusses the constraints affected by the project and provide suggestions for improvements in the future.

5.2 Objectives Revisited

The first objective of the project was to study the fundamentals of blockchain technology and its viability as a real-world solution for problems (e.g., voting), along with the use of zero-knowledge proofs. This was achieved and shown in Chapter 1 and 2, where a brief introduction to blockchain technology and its real-world uses was included after literature review was done on existing projects and publications by academics and researchers.

The next objective was to develop a voting system that utilizes both blockchain technology along with zero knowledge proofs. This was accomplished and discussed in

Chapter 3 and 4, where the methodology was outlined and the process of development, along with the design of the system, was documented from start to finish.

The last objective was to evaluate the accessibility, security, and verifiability of the voting system with the implementation of blockchain technology and zero knowledge proofs, along with public perception towards these emerging technologies. This was done though the User Acceptance Test (UAT) in Chapter 4, where respondents can provide feedback on the features and criteria for the system outlined above, along with their thoughts on blockchain technology and zero-knowledge proofs.

5.3 Research Constraint

5.3.1 Time Limitation

With limited time, exploration into various types of blockchain technology was not feasible. Thus, only the proof of work blockchain was implemented and discussed in this project.

5.3.2 Node Limitation

In a true blockchain system, various nodes are required to provide fairness in validation of the chain itself. Due to time constraints and lack of technical skills, this was not implemented, and the system only runs on one node.

5.3.3 System Complexity

As of its final iteration, the system only supports an election between two candidates. While more candidates can be added with modifications to the system, the project is currently designed for two candidates only due to the added complexity when involving multiple candidates.

5.4 Future Work

Several improvements can be made to the Prometheus blockchain voting system in order to achieve a more cohesive system.

- Adding more nodes to the system to better showcase and illustrate the potential of blockchain technology when applied to voting systems.

- Develop a mobile application for the blockchain voting system to further improve the accessibility of the system.

- Modify the system to allow elections involving more than two candidates to better illustrate real life scenarios.

REFERENCES

Al-Maaitah, S., Qatawneh, M., and Quzmar, A., "E-Voting System Based on Blockchain Technology: A Survey," 2021 International Conference on Information Technology (ICIT), 2021, pp. 200-205, doi: 10.1109/ICIT52682.2021.9491734.

Alvi, S. T., Uddin, M. N., Islam, L. and Ahamed, S., "From Conventional Voting to Blockchain Voting: Categorization of Different Voting Mechanisms," 2020 2nd International Conference on Sustainable Technologies for Industry 4.0 (STI), 2020, pp. 1-6, doi: 10.1109/STI50764.2020.9350399.

Ben-Sasson, E., Chiesa, A., Green, M., Tromer, E. and Virza, M., "Secure Sampling of Public Parameters for Succinct Zero Knowledge Proofs," 2015 IEEE Symposium on Security and Privacy, 2015, pp. 287-304, doi: 10.1109/SP.2015.25.

Cetinkaya, O. and Cetinkaya, D., "Towards Secure E-Elections in Turkey: Requirements and Principles", International Workshop on Dependability and Security in e-Government (ARES'07), Vienna, Austria, pp. 903-907, 10-13 April 2007.

Cetinkaya, O., "Analysis of Security Requirements for Cryptographic Voting Protocols (Extended Abstract)," 2008 Third International Conference on Availability, Reliability and Security, 2008, pp. 1451-1456, doi: 10.1109/ARES.2008.167.

Kamran, M., Nasir H., Imran, M. and Yang, J. S., "Study on E-Voting Systems: A Blockchain Based Approach," *2021 IEEE International Conference on Consumer Electronics-Asia (ICCE-Asia)*, 2021, pp. 1-4, doi: 10.1109/ICCE-Asia53811.2021.9641914.

Minnite, L. (2007). “The Politics of Voter Fraud.” Scotts Valley, CA. CreateSpace Independent Publishing Platform, 2018

Patidar, K., and Jain, S., "Decentralized E-Voting Portal Using Blockchain," 2019 10th International Conference on Computing, Communication and Networking Technologies (ICCCNT), 2019, pp. 1-4, doi: 10.1109/ICCCNT45670.2019.8944820.

Pomares, J., Levin, I., Alvarez, R. M., Mirau, G. L. and Ovejero, T., "From piloting to roll-out: voting experience and trust in the first full e-election in Argentina," 2014 6th International Conference on Electronic Voting: Verifying the Vote (EVOTE), 2014, pp. 1-10, doi: 10.1109/EVOTE.2014.7001136.

Rewatkar, H. R., Agarwal, D., Khandelwal, A. and Upadhyay, S., "Decentralized Voting Application Using Blockchain," 2021 10th IEEE International Conference on Communication Systems and Network Technologies (CSNT), 2021, pp. 735-739, doi: 10.1109/CSNT51715.2021.9509561.

Singh, A. and Chatterjee, K., "SecEVS : Secure Electronic Voting System Using Blockchain Technology," 2018 International Conference on Computing, Power and Communication Technologies (GUCON), 2018, pp. 863-867, doi: 10.1109/GUCON.2018.8675008.

Stapleton, J. (1997). DSDM, dynamic systems development method : the method in practice. Addison-Wesley.

Thiga, M. M., "Increasing Participation and Security in Student Elections through Online Voting: The Case of Kabarak University," 2020 IST-Africa Conference (IST-Africa), 2020, pp. 1-7.

APPENDIX A
SAMPLE APPENDIX 1

PROMETHEUS USER ACCEPTANCE TEST SUMMARY (UAT)

Test Case IDs: TC-PROM-01 - 10	**Test Designed Date:** 02/01/2023
Test Designed By: Lim Aun Xian	
Test Executed By: 30 randomly selected Universiti Malaysia Pahang students	
Test Execution Date: 03/01/2023 – 09/01/2023	
Modules Performance : All modules passed their testing.	
General Critique: Administrative pages are hard to be distinguished, requires update to the user interface.	
Overall : Positive response, showed useful implementation of blockchain technology, zero knowledge proof has potential for many applications.	

Admin Login

Test Case ID: TC-PROM-01	**Test Designed By:** Lim Aun Xian
Test Priority (Low/Medium/High): High	**Test Designed Date:** 02/01/2023
Module Name: Admin Login	**Test Executed By:**
Test Title: Register user account	**Test Execution Date:**
Description: Login using the admin username and password to gain access to the admin panel	
Precondition: None	
Dependencies: None	

Steps	**Test Steps**	**Test Data**	**Expected Result**	**Actual Result**	**Status (Pass / Fail)**	**Comments**
1.	Navigate to admin login page ([localhost]/admin)		System displays login screen			
2.	Enter the username and password	Username: admin Password: blockchain				
3.	Click the login button		System navigate user to the admin panel			

Add Candidates

Test Case ID: TC-PROM-02	**Test Designed By:** Lim Aun Xian
Test Priority (Low/Medium/High): High	**Test Designed Date:** 02/01/2023
Module Name: Add Candidates	**Test Executed By:**
Test Title: Add candidates to election	**Test Execution Date:**
Description: Add new candidates to the election with their profile picture, name, year of study, faculty, and election slogan.	
Precondition: None	
Dependencies: None	

Steps	**Test Steps**	**Test Data**	**Expected Result**	**Actual Result**	**Status (Pass / Fail)**	**Comments**
1.	Click on the "Modify Candidates" button		System navigates to the modify candidate page			
2.	Enter the profile picture, name, year of study, faculty, and election slogan	Profile Picture : Any (1:1 ratio) Name : Any Year of Study : 3rd Year Faculty : Any Election Slogan : Any				
3.	Click the add candidates' button		System navigate user to the admin panel			

Start Election

Test Case ID: TC-PROM-03	**Test Designed By:** Lim Aun Xian
Test Priority (Low/Medium/High): High	**Test Designed Date:** 02/01/2023
Module Name: Start Election	**Test Executed By:**
Test Title: Start a new election	**Test Execution Date:**
Description: Start a new election.	
Precondition: Candidates have been added in Modify Candidates	
Dependencies: None	

Steps	Test Steps	Test Data	Expected Result	Actual Result	Status (Pass / Fail)	Comments
1.	Click on the “Start Election” button		System navigates to the start election page			
2.			System displays confirmation that the election has started.			
3.	Click on the “Return to Admin Panel” button		User is redirected back to the admin panel			

Register as Voter

Test Case ID: TC-PROM-04	**Test Designed By:** Lim Aun Xian
Test Priority (Low/Medium/High): High	**Test Designed Date:** 02/01/2023
Module Name: Voter Registration	**Test Executed By:**
Test Title: Register as Voter	**Test Execution Date:**
Description: Register as a voter using your name, age, faculty, email, and student ID	
Precondition: Start Election	
Dependencies: None	

Steps	Test Steps	Test Data	Expected Result	Actual Result	Status (Pass / Fail)	Comments
1.	Click on the " Don't have your zero-knowledge proof? Register here" button		System navigates to the voter registration page			
2.	Input user information	Name : Any Age : Any Faculty : Any Email : Any Student ID : Any				
3.	Click on the "Register" button		System will display user's ZK Proof statement.			

4.	Click on the ZK Proof Statement		The ZK Proof statement should be copied to the user's clipboard.			

Verify ZK Proof

Test Case ID: TC-PROM-05			**Test Designed By:** Lim Aun Xian			
Test Priority (Low/Medium/High): High			**Test Designed Date:** 02/01/2023			
Module Name: Verify ZK Proof			**Test Executed By:**			
Test Title: ZK Proof Verification			**Test Execution Date:**			
Description: Provide ZK Proof to be validated in order to gain access to the Voting module.						
Precondition: Start Election, Register as Voter						
Dependencies: None						
Steps	**Test Steps**	**Test Data**	**Expected Result**	**Actual Result**	**Status (Pass / Fail)**	**Comments**
1.	Input previously obtained ZK Proof statement	ZK Proof statement	System validates and redirects users to the voting page			
2.	Land in voting page		User is redirected to [localhost]/vote			

Vote

Test Case ID: TC-PROM-06	**Test Designed By:** Lim Aun Xian
Test Priority (Low/Medium/High): High	**Test Designed Date:** 02/01/2023
Module Name: Voting	**Test Executed By:**
Test Title: Vote for Candidate	**Test Execution Date:**
Description: Provide your ZK Proof then vote for desired candidate.	
Precondition: Start Election, Register as Voter, Verify ZK Proof	
Dependencies: None	

Steps	**Test Steps**	**Test Data**	**Expected Result**	**Actual Result**	**Status (Pass / Fail)**	**Comments**
1.	Input ZK Proof	ZK Proof statement				
2.	Click on the “Vote” button		Vote is submitted, user is notified			

End Election

Test Case ID: TC-PROM-07	**Test Designed By:** Lim Aun Xian
Test Priority (Low/Medium/High): High	**Test Designed Date:** 02/01/2023
Module Name: End Election	**Test Executed By:**
Test Title: Election End	**Test Execution Date:**
Description: End an ongoing election.	
Precondition: Start Election	
Dependencies: None	

Steps	**Test Steps**	**Test Data**	**Expected Result**	**Actual Result**	**Status (Pass / Fail)**	**Comments**
1.	Click on the “End Election” button		System shows confirmation			
2.	Click on the “Return to Admin Panel” button		User is redirected back to the admin panel			

View Blockchain Transaction

Test Case ID: TC-PROM-08	**Test Designed By:** Lim Aun Xian
Test Priority (Low/Medium/High): High	**Test Designed Date:** 02/01/2023
Module Name: Blockchain Verify	**Test Executed By:**
Test Title: Verify transaction on blockchain	**Test Execution Date:**
Description: Search for your transaction(vote) on the blockchain	
Precondition: End Election	
Dependencies: None	

Steps	**Test Steps**	**Test Data**	**Expected Result**	**Actual Result**	**Status (Pass / Fail)**	**Comments**
1.	Click on the " Blockchain" navigation button		System navigates to the blockchain page			
2.	Input ZK proof statement	ZK Proof statement				
3.	Click on the "Submit" button		System will display user's transaction in the blockchain.			

Download Blockchain

Test Case ID: TC-PROM-09	**Test Designed By:** Lim Aun Xian
Test Priority (Low/Medium/High): High	**Test Designed Date:** 02/01/2023
Module Name: Blockchain Download	**Test Executed By:**
Test Title: Download Blockchain	**Test Execution Date:**
Description: Download the blockchain file for further verification.	
Precondition: End Election	
Dependencies: None	

Steps	Test Steps	Test Data	Expected Result	Actual Result	Status (Pass / Fail)	Comments
1.	Click on the " Blockchain" navigation button		System navigates to the blockchain page			
2.	Click on the "Download Blockchain" button		The blockchain will be exported and saved as a .txt file			

View Results

Test Case ID: TC-PROM-10	**Test Designed By:** Lim Aun Xian
Test Priority (Low/Medium/High): High	**Test Designed Date:** 02/01/2023
Module Name: Results	**Test Executed By:**
Test Title: View Results	**Test Execution Date:**
Description: View the election results after the election has finished.	
Precondition: End Election	
Dependencies: None	

Steps	**Test Steps**	**Test Data**	**Expected Result**	**Actual Result**	**Status (Pass / Fail)**	**Comments**
1.	Click on the “Results” navigation button		System navigates to the results page			
2.	Click on the “Reveal” button		The results of the election will be displayed, and the winner announced.			