

E-VOTING SYSTEM USING BLOCKCHAIN



A PROJECT REPORT

Submitted by

REG NO	NAME
2011903	KODEESWARAN M
2011907	SARVAN KUMAR M
2011909	TAMARAI SELVAN R

*in partial fulfillment for the award of the degree
Of*

BACHELOR OF COMPUTER SCIENCE AND ENGINEERING

Under the guidance of

Dr. J. NAFEESA BEGUM, M.E., Ph.D.,

Professor and Head
Department of Computer Science and Engineering

**GOVERNMENT COLLEGE OF ENGINEERING
(AUTONOMOUS)**

BARGUR, KRISNAGIRI-635 104.

(Affiliated to Anna University, Accredited by NAAC with a 'B' Grade)

ANNA UNIVERSITY: CHENNAI 600 025

MAY 2023

**GOVERNMENT COLLEGE OF ENGINEERING
(AUTONOMOUS)**

BARGUR, KRISNAGIRI-635 104.

Department of Computer Science and Engineering

BONAFIDE CERTIFICATE

Certified that this project titled **“E-VOTING SYSTEM USING BLOCKCHAIN”** is the bonafide work of **KODEESWARAN M (2011903), SARVAN KUMAR M (2011907)** and **TAMARAI SELVAN R (2011909)** who carried out the project work under my supervision.

**Dr. J. NAFEESA BEGUM, M.E.,Ph.D.,
SUPERVISOR**

Professor [CAS],
Department of CSE,
Govt. College of Engineering,
Bargur-635 104.

**Dr. J. NAFEESA BEGUM, M.E.,Ph.D.,
HEAD OF THE DEPARTMENT**

Professor,
Department of CSE,
Govt. College of Engineering,
Bargur-635 104.

Submitted for the Project Viva -Voce Examination held on _____.

INTERNAL EXAMINER

EXTERNAL EXAMINER

ACKNOWLEDGEMENT

First and foremost, we would like to record our deepest gratitude to **our parents and siblings** for their constant encouragement and support which motivated us to complete our project on time.

We express our sincere gratitude and thanks to our respected Principal **Dr. R. VIJAYAN, M.E., Ph.D.**, and our respected Head of the Department of Computer Science and Engineering **Dr. J. NAFEESA BEGUM, M.E., Ph.D.**, for giving us the opportunity to display our professional skills through this project.

We want to express our deep and wholehearted thanks to our project guide, **Dr. J. NAFEESA BEGUM, M.E., Ph.D.**, Department of Computer Science and Engineering for her valuable guidance and motivation which helped us to complete this project on time.

We would like to thank our project Coordinator, **Prof. R. SUBATHRA, M.E.**, Faculty Advisor of Computer Science and Engineering, who guided us throughout the entire phase of our projects with her esteemed presences. It is her motivation and guidance which made us explore our project.

We again greatly thankful to our project guide **Dr. J. NAFEESA BEGUM, M.E., Ph.D.**, Department of Computer Science and Engineering for her valuable guidance and motivation, which helped us to complete this project on time.

We thank all **our teaching and non-teaching staff** members of Computer Science and Engineering for their passionate support for helping us to identify our mistakes and also for the appreciation they gave us in achieving our goal.

KODEESWARAN M(2011903)

SARVAN KUMAR M (2011907)

TAMARAI SELVAN R (2011909)

ABSTRACT:

Electronic voting or e-voting has been used in varying forms since the 1970s with fundamental benefits over paper-based systems such as increased efficiency and reduced errors. However, there remain challenges to achieving widespread adoption of such systems especially with respect to improving their resilience against potential faults. Block-chain is a disruptive technology of the current era and promises to improve the overall resilience of e-voting systems. This paper presents an effort to leverage the benefits of block-chain such as cryptography foundations and transparency to achieve an effective scheme for e-voting. The proposed scheme conforms to the fundamental requirements for e-voting schemes and achieves end-to-end verifiability. The paper presents details of the proposed e-voting scheme along with its implementation using the Multi chain platform. The paper presents an in-depth evaluation of the scheme which successfully demonstrates its effectiveness to achieve an end-to-end verifiable e-voting scheme.

TABLE OF CONTENTS

ACKNOWLEDGEMENT	i
ABSTRACT.....	ii
LIST OF FIGURES.....	v
LIST OF TABLES.....	v
CHAPTER 1 - INTRODUCTION	1
1.1 EXISTING SYSTEM.....	2
1.2 PROBLEMS WITH EXISTING SYSTEM.....	2
1.3 SOFTWARE DESIGN METHODOLOGIES	2
1.3.1 WATERFALL	3
1.3.2 RAPID APPLICATION DEVELOPMENT	4
1.3.3 CHOOSING WATERFALL FOR THE SOFTWARE DESIGN ..	5
CHAPTER 2 - PROJECT PLANNING	6
2.1 AIM AND OBJECTIVE	6
2.2 SYSTEM DELIVERABLE	7
2.3 RESEARCH TO BE CARRIED OUT	7
2.4 SYSTEM DESIGN	7
2.4.1 UNIFIED MODELING LANGUAGE (UML)	8
CHAPTER 3 - SYSTEM SECURITY	9
3.1 NETWORK SECURITY ATTACKS.....	9
3.1.1 DENIAL OF SERVICE ATTACK (DOS).....	9
3.1.2 MAN-IN-THE-MIDDLE ATTACK (MITM)	10
3.2 AUTHENTICATION	10

3.3	ENCRYPTED COMMUNICATION	11
3.4	SYMMETRIC KEY CRYPTOGRAPHY	11
3.4.1	BLOCK CIPHERS	12
3.5	ASYMMETRIC KEY CRYPTOGRAPHY	12
3.6	ENCRYPTION ALGORITHMS	13
3.6.1	(HOMOMORPHIC) ENCRYPTION ALGORITHM	13
3.6.2	DATA ENCRYPTION STANDARD ALGORITHM (DES)	14
3.6.3	BLOWFISH ALGORITHM	14
3.7	SECURE SOCKET LAYER (SSL)	14
3.8	JAVA CRYPTOGRAPHY EXTENSION (JCE)	15
CHAPTER 4 - SYSTEM DESIGN		17
4.1	SYSTEM REQUIREMENTS	17
4.1.1	FUNCTIONAL REQUIREMENTS	18
4.1.1	NON-FUNCTIONAL REQUIREMENTS	18
4.2	DESIGN TECHNIQUES	19
4.3	WEBSITE INTERFACE DESIGN	19
4.3.1	WEBSITE FORMS	19
4.4	DATABASE DESIGN	20
4.4.1	ENTITIES & ATTRIBUTES	20
4.4.2	DATABASE ENTITY & ATTRIBUTE DIAGRAM	22
4.5	SOFTWARE DESIGN	24
4.5.1	USE CASE MODELING	24
4.5.2	SYSTEM ACCESS DESIGN	26
4.3.1	ADMINISTRATOR & VOTER DESIGN	28

4.5.4	SYSTEM SECURITY DESIGN	29
4.5.5	SYSTEM ARCHITECTURE	30
CHAPTER 5 - IMPLEMENTATION		31
5.1	SERVER SETUP & CONFIGURATION	31
5.1.1	SSL CONFIGURATION ON TOMCAT	32
5.2	SYSTEM IMPLEMENTATION	33
5.3	SERVLET CONFIGURATION	34
5.4	PYTHON CLASS IMPLEMENTATION	35
5.5	ADMIN IMPLEMENTATION	37
5.6	VOTER IMPLEMENTATION	40
5.7	ENCRYPTION CLASS IMPLEMENTATION	41
CHAPTER 6 - TESTING		42
6.1	TEST STRATEGIES	42
6.1.1	TEST PLAN	42
6.2	TEST DATA	43
CHAPTER 7 - SOURCE CODE		46
CHAPTER 8 - SCREENSHOTS		57
CHAPTER 9 - CONCLUSION.....		65
REFERENCES.....		67

LIST OF FIGURES

Figure 1 - Water Fall Model	4
Figure 2 - SSL running above TCP/IP.....	15
Figure 3 - Administrator entity with its attributes.....	22
Figure 4 - Users entity with its attributes.....	22
Figure 5 - Candidate entity with its attributes.....	23
Figure 6 - Voters' entity with its attributes.....	23
Figure 7 - Administrator use case diagram.....	25
Figure 8 - Voter use case diagram.....	25
Figure 9 - Displays the system's access login design.....	26
Figure 10 - Design of the forgotten password section.....	27
Figure 11 - Design of the administrator and voter system features after login ...	28
Figure 12 - Users entity with its attributes.....	22
Figure 13 - Users entity with its attributes.....	22
Figure 15 - Users entity with its attributes.....	22

LIST OF TABLES

Table 1 - Entities used in the database system	21
Table 2 - Security features to be implemented in the system.....	29
Table 3 - Testing features.....	43

CHAPTER 1

INTRODUCTION

The term voting is understood to be a form of choice. This form of expression can be performed through the ballot, or by any other electoral schemes. Electron coting is a way in which votes cast by voters of a specific electronic medium can be retrieved, tallied, and stored electronically.

The project to be produced will be focusing on converting the current paper-based elections system currently being used by the University of Westminster Student Union into an electronic system. The current voting system being used by the student union is currently suffering from poor voter turnout because the system in place is not convenient for most students. The system to be created will address this issue by providing voters with the capability of casting their votes for their chosen candidates via an internet enabinternet-enabled project will focus on the current voting method being used by the student union, and identify a way in which the method can be moledith the internet voting system to be implemented. The system will implement different election mechanisms used for casting votes.

The system will be built to have strict security features. These security features will commence from the point of voter login into the voting system, to casting their vote for their chosen candidate to the point of their exit from the system. The system will have security restrictions preventing the voter from voting more than once for eligible candidates.

The system to be implemented needs to address the issues covering the security needs of a vote being cast over the Internet. Authentication and validation of the users, access rights, information encryption, and vote security need to be looked into in an in-defashionashtoot secure means of voting online.

1.1 EXISTING SYSTEM

The voting system currently being used by the University's student union is a paper-based system, in which the voter simply picks up ballots sheets from electoral officials, ticks off whom they would like to vote for, and then cast their votes by merely handing over the ballot sheet back to the electoral official. The electoral officials gather all the votes being cast into a ballot box. At the end of the elections, the electoral officials converge and count the votes cast for each candidate and determine the winner of each election category.

1.2 PROBLEMS WITH EXISTING SYSTEM

The current system in use today, has several problems my proposed system would aim to correct. The system is highly insecure and prone to election malpractice. The Because can come and fill out a ballot sheet without prior authentication to determine who he/she says they are, is a major concern. The administration of the voting system as a whole is highly inefficient, slow and time consuming, and is highly prone to human error.

1.3 SOFTWARE DESIGN METHODOLOGIES

The most important aspect of software development is; the meticulous planning that takes place before the project can begin. Developing a software system is usually a complex and time-consuming process. In order to control the software system process we try to adhere to some kind of framework that introduces certain degrees of structure to the overall development process.

Software engineering methodologies are the back bone for developing software; the methodologies simply assist one in how one should go on about building a software system which meets its purpose. Various methodologies are used for different types of software development

depending on the scale of the software to be built. Hence one follows the various stages of development methods, such as the planning stage, followed by the requirement stage, design stage, testing, and lastly maintenance stage. These are the type of framework that can minimize time consumption, allow for good control in the process stage and reduce the complexity and uncertainties of the software development.

This project involves building a dynamic web-based voting system. In order to achieve this, an appropriate software design methodology which would suit the project has to be chosen.

- Waterfall
- Rapid Application Development

1.3.1 WATERFALL

The waterfall model is a software development model in which a system's development is viewed as flowing downwards through the phases of the system development process. The waterfall methodology is powerful, precise, and thorough. It has a number of phases that have to be implemented in a sequential manner as shown in figure1-1. The phases which come under the waterfall method are as follows.

- Requirement Analysis
- Design
- Implementation
- Testing
- Maintenance

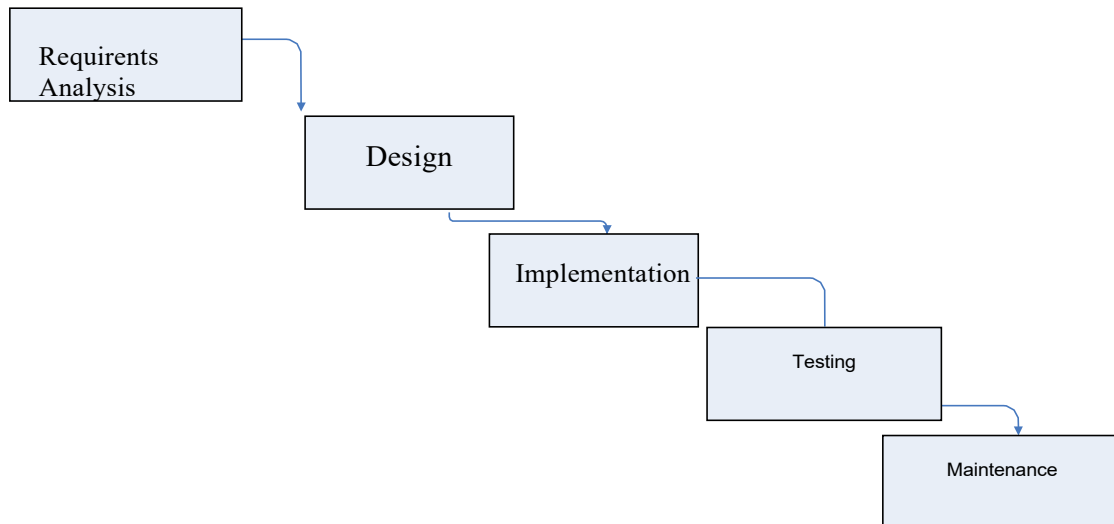


Figure 1-1: Water Fall Model

ADVANTAGES

- Good for large projects
- Waterfall suits a principled approach to design
- Waterfall divides the project into manageable areas
- Waterfall separates the logical and physical

DISADVANTAGES

- Clients are not involved
- Lack of flexibility
- Clients can only see the product when it is finished
- Each stage is completed before moving on to the other stages

1.3.2 RAPID APPLICATION DEVELOPMENT

Rapid Application Development (RAD) is one of the major players in information systems development. RAD is a methodology for compressing the analysis, design, build, and test phases into a series of short, iterative development cycles in order to develop systems at a quicker pace. The two key features of RAD are time boxing and Joint Application

Development (JAD). Time boxing is an approach for fixing the resource allocation for a project. It limits the time available for the refinement of requirements, design, construction and implementation as appropriate. JAD involves both the developers and the customer to identify, what the customer wants from the system that is about to be built. Hence, both parties participate in the building of the system.

ADVANTAGES

- Evolves the user at every level
- Aims to complete 80% of the work in 20% of the time compared with other methodologies
- Very flex able for scope changes

DISADVANTAGES

- Not ideal for critical missions
- Customer could change their mind several times
- Cost can become expensive

1.3.3 CHOOSING WATERFALL FOR THE SOFTWARE DESIGN

Building a web-based voting involves a work of meticulous planning and structuring, it can often be difficult if one does not follow a well structured methodology approach. After an evaluation of the suitability of the most commonly used life-cycle methodologies: Waterfall, RAD and the Prototyping: the waterfall model was chosen. By combining the better features of the other two approaches, the waterfall model is particularly suitable for addressing the needs of this project. Other models such as the RAD and Prototyping models were not as strong and structured as the waterfall model.

CHAPTER-2

PROJECT PLANNING

The project to be undertaken has to have a certain project plan, which would serve as a structured guide for researching, designing and developing the project.

2.1 AIMS AND OBJECTIVES

The aims and objectives of the system to be produced have been stated below:

- To build an online system this would enable voters to cast their votes on chosen candidates.
- Create a secure authentication facility to check validate users logging into the voting system
- Create a database to be used to stored votes, and user information on the system
- Study and implement a security method to be used to ensure that votes being cast in the system will not be compromised and any outside attack
- Enable the system to tally votes cast according to candidate voted for.
- Create a back-end administration section which will be used to enable the administration manage the election system effectively
- Create tools for the administrator to add, delete and update details of voters, candidates and sub administrators on the system
- Display voting results in a graphical fashion for the administer to analyze
- To enable voters to cast their votes for their chosen candidates
- Enable voters to view biographies of the candidates being voted for in the election
- Timestamp votes cast to the database to know when each vote was cast
- Enable administrators to generate reports on the vote results
- Prevent voters from voting more than once for their choose candidates

2.2 SYSTEM DELIVERABLES

The system to be delivered at the end of the implementation and testing phase would consist of an amiable website, which would act as the front-end of the system and also as the main entry point to the system. A Python application in form of Servlets would be produced to facilitate the numerous requests, which would be sent to the web server to be used.

A database would also have to be constructed to store the data to be retrieved of the system's users; it will also be a highly essential tool for authenticating the system's users. Security would be highly prioritised in the building of the voting system, and SSL (Secure Socket Layer) and a mode of password encryption would also be utilised in the construction of the system.

2.3 RESEARCH TO BE CARRIED OUT

In order to progress in the design and paramount construction of the online voting, an extensive form of research has to be carried, to gain more knowledge on the system to be built and to allow analyze different components to be used for constructing the system. The topics of the research to be carried out are listed below.

- Existing electronic voting systems in use
- Website development software
- Server side programming languages
- Databases
- Internet Security

2.4 SYSTEM DESIGN

The voting system's design is an important factor to the usability and durability of the whole system. The system will be engineered in a simple and straight-forward pattern, minimizing complexity and maximizing simplicity, usability and efficient structuring.

2.4.1 UNIFIED MODELING LANGUAGE (UML)

UML was designed to give its users an ‘expressive visual modelling language’ which would allow them to exchange models they developed. It is object-oriented modelling language, allows for specialized extensions, is independent of the programming language used and provides a formal method of interpreting the language. The notation can be used throughout the lifecycle and is not restricted to software projects, although it is optimized for them.

CHAPTER-3

SYSTEM SECURITY

A top priority for any voting system is to maintain the integrity of the votes cast during an election. Online voting systems are only feasible means of carrying out an election, if the system is safe and secure. Voters, who are not confident with the security aspects of the voting system, will not want to cast their votes online. Secure systems are developed so that the rewards retrieved when system is protected outweigh the costs of the system being broken into by a computer hacker. The systems security should be in proportion to what it is protecting. In an online voting system client/server security is an important feature which should be carefully implemented. In order to achieve this goal, an efficient form of authorization and authentication has to be established.

3.1.NETWORK SECURITY ATTACKS

Any web based computer system is susceptible to attacks from system hackers who could attempt to overwhelm a computer system to gain information for illegal use. They could also attempt to crash a system for the aim of sabotaging a Company's business operations. There are a number of system attacks that have been established to sabotage computer systems.

3.1.1 DENIAL OF SERVICE ATTACK (DOS)

A denial of service attack is an attack on a computer or network system that causes the system's users to be deprived of the services which the system provides. The typical loss of service could be the temporary loss of network connectivity which could affect a web-based business considerably due to the fact that the website might have to cease operation to its customers. The DOS attack can come in a number of forms.

3.1.2. MAN-IN-THE-MIDDLE ATTACK (MITM)

MITM attack is an attack in which data being transmitted between two parties on a network is intercepted, read and modified by a system attacker without the communicating parties knowing that their data has been compromised.

To describe the MITM attack process, this form of attack can be explained as Stephanie being the client would like to establish a connection directly with Michael the server.

Stan the attacker would lie in wait for Stephanie to send a request to Michael, upon Stephanie sending the request; Stan would intercept the request, manipulate it and send it to Michael for processing. Michael thinking he is responding to Stephanie directly sends a response which Stan intercepts.

3.2. AUTHENTICATION

Authentication is the process of establishing whether someone or something is who or what it is declared to be. In most internet network systems authentication is generally done through the use of login usernames and passwords.

The user of the system is assumed to know the password in order to get authenticated. Every user is initially registered on the system by a system administrator using an assigned or self-declared password. On each subsequent use, the user must know and use the previously declared password. The main weakness of these kinds of systems that is considerable is that passwords can be guessed, stolen, accidentally revealed, or forgotten by the user. System hackers use password guessing as a simple method of attacking a computer system, be it on a network or offline.

Password guessing requires the hacker to have known usernames and suitable password guesses, by persistently trying the guessed passwords into the system, the attacker could finally break in, and this is mainly due to poor passwords being chosen by users. The best way to protect a system from this form of unwanted intrusion is to prevent users from having an infinite number of login attempts with wrong passwords; the user should be locked out of the system after a specific number of failed login attempts.

3.3. ENCRYPTED COMMUNICATION

The communication process over the internet is intrinsically insecure, due to the fact that data being transferred over the internet medium can be susceptible to attacks and eavesdropping from different points of the transmission route. There is an essential need that online systems which deal with confidential and sensitive data, such as an online voting system, have to provide a means in which data communication between the client to the server is encrypted, thereby making the data being transmitted unusable to a would-be system attacker. There are a number of cryptography algorithms which can be used to encrypt data; algorithms like HOMOMORPHIC, DES, and Blow-fish can all be used at some point of an online system to make it secure. These algorithms are going to be discussed, but the main encryption processing techniques which are behind these algorithms are the Symmetric key cryptography and the Asymmetric key cryptography.

3.4. SYMMETRIC KEY CRYPTOGRAPHY

This form of encryption is also known as the secret key cryptography. Symmetric key cryptography makes use of the same private key while performing an encrypted communication between two users. The same secret key is used for the encryption and decryption of data being transmitted between the two or more users.

This form of cryptography makes use of stream ciphers and block ciphers for encrypting plain text. A stream cipher is an encryption method that is used to encrypt plain text or digits one character at a time while block ciphers encrypts blocks of data.

Symmetric Key cryptography example is the Data Encryption Standard (DES) algorithm.

3.4.1 BLOCK CIPHERS

A block cipher is an encryption method which encrypts large blocks of text; the block cipher regards the input stream for encryption as blocks of fixed sized bytes which can be up to 128bits long.

The block cipher can encrypt a 128bit plaintext and generate a 128bit cipher text as the output result. The block cipher also has a reverse mechanism, which is in form of a decryption function that converts the 128bit ciphertext and decrypts it back to the 128bit plaintext. In order for a block cipher to encrypt data, the function would need a secret key which comes as a string of bits normally 128 to 256 bits long.

3.5. ASYMMETRIC KEY CRYPTOGRAPHY

This form of encryption makes use of one public key which is available to all users and a private key which is known only by the message recipient. The public key can be exchanged between users who can use it to encrypt data being transmitted to another user, the private key which should be kept secret, is used to decrypt the encrypted data to produce the original unencrypted data. This form of key cryptography is used by the Rivets, Shamir, and Adelman (HOMOMORPHIC) encryption algorithm.

3.6. ENCRYPTION ALGORITHMS

Encryption algorithms are used to turn plain text to cipher text. Different forms of encryption algorithms exist and each form has a unique method of generating keys and encrypting input streams.

3.6.1. (HOMOMORPHIC) ENCRYPTION ALGORITHM

The HOMOMORPHIC encryption algorithm is mainly an internet based encryption algorithm, which can be used for authentication. The HOMOMORPHIC encryption algorithm uses the public/private key cryptography technique to encrypt and decrypt data being transported between users.

A complex process of mathematical calculations have to take place to acquire a set of two prime numbers that would represent the public key used to encrypt the data and the private key used to decrypt the data once received by the receiver.

The HOMOMORPHIC algorithm works as follows: take two large primes, p and q , and compute their product $n = pq$; n is called the modulus. Choose a number, e , less than n and relatively prime to $(p-1)(q-1)$, which means e and $(p-1)(q-1)$ have no common factors except 1. Find another number d such that $(ed - 1)$ is divisible by $(p-1)(q-1)$. The values e and d are called the public and private exponents, respectively. The public key is the pair (n, e) ; the private key is (n, d) . The factors p and q may be destroyed or kept with the private key.

The HOMOMORPHIC encryption algorithm is used by the Secure Socket Layer (SSL) for encrypting data being transmitted over a secure connection.

3.6.2.DATA ENCRYPTION STANDARD ALGORITHM (DES)

The DES encryption algorithm uses the symmetric key cryptographic technique, whereby the same secret key is used for encrypting and decrypting data. The encryption algorithm is a block cipher which applies a 56 bit key to each 64 bit block of data to be encrypted.

3.6.3. BLOWFISH ALGORITHM

Blowfish is an encryption algorithm which was created by Bruce Schneier. It is a symmetric block cipher with a block size of 64 bits and a variable length key from 32bits to 448 bits.

3.7. SECURE SOCKET LAYER (SSL)

In the world of electronic commerce, security is a highly essential feature to have in any web system. A socket is a term for a communications port between computers over any interconnection medium using any computer-to-computer protocol.

SSL is a protocol that is used for sending secure encrypted data over the internet. SSL layer is present between TCP/IP protocol and the application layer as shown in figure 3.1. SSL protocol can protect users from “man in the middle attacks”.

The SSL protocol is based on the public key cryptography which has a public and private key pair, the public key can be revealed to everyone but the private key is only known to the recipient of the message being sent. The message is encrypted with the public key and decrypted with the private key.

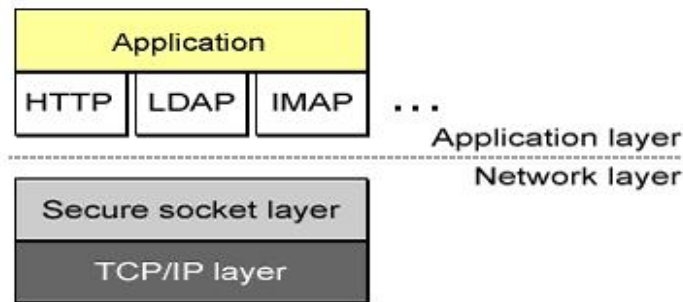


Figure 3.1: SSL running above TCP/IP.

SSL security medium is based on cryptography. Cryptography is the conversion of data into a secret code for transmission over a public network. The plain text is converted into a coded equivalent called cipher text via an encryption algorithm. The cipher text is decrypted at the receiving end and turned back into plaintext. HOMOMORPHIC is the most commonly used internet encryption algorithm.

The Secure Socket Layer protocol has been succeeded by the Transport Layer Security (TLS) protocol, which has similar features to its predecessor SSL. Most internet browser software support TLS.

3.7.JAVA CRYPTOGRAPHY EXTENSION (JCE)

JCE is an implementation of cryptography for Python systems. The JCE package provides a framework for encryption, decryption, key generation, key agreement and Method Authentication Code algorithms on Python platforms. JCE encryption allows symmetric, asymmetric, block, and stream ciphers with additional support for secure streams on the Python platform.

The JCE API was created to support a number of encryption algorithms through a number of Python classes, which a developer could use for implementing security features in a Python based system. The advantage about using the JCE API is that, the developer would not have to understand the logic behind the encrypting algorithms because the details of the encryption algorithm would be managed by the provider. The JCE

Framework provides a service provider that implements the following encryption algorithms.

- BlowFish
- DES

The Python encryption class to be used for the Online Voting System would make of the DES encryption algorithm.

CHAPTER-4

SYSTEM DESIGN

This chapter's goal is to describe the way in which the online voting system is to be built. In order to build an efficient and flexible system, the appropriate system development methodology has to be chosen to suit the system to be created. The waterfall design methodology is to be utilized to design and develop the online voting system.

In order for any form of computer systems to be built in an efficient and user friendly manner, a highly structured and well engineered design has to be created. The design of a software orientated system has to follow certain steps in achieving its end product. The design of a system enables organizations and companies to map out a strategic plan which the system developers would have to follow. The design of a system is very important in the construction of any web based application, and it prevents the occurrence of mistakes and errors during the implementation phase which can be highly costly to the organization funding the specific project.

4.1.SYSTEM REQUIREMENTS

The online voting system to be built will be used by two sides, the students who would be voting and the administrators who are in charge of creating and maintaining information on the system.

The system has to be very secure due to the fact that it is a voting system, the main objective of the voting system is to ensure that votes being cast by voter cannot be rigged or unduly compromised in any shape or form. A high level of user authentication has to be established to maintain security.

The information and usability of the voting system has to be very constructive, efficient and easy to understand by the user. Good systems are easy to utilise, the user should be saved from any form of complexity.

4.1.1. FUNCTIONAL REQUIREMENTS

- Permit users to gain access and utilise information conveyed on the website to be implemented.
- Enable secure form of authenticating users who would like to gain access into the voting system.
- Enable a secure mode of communication between the client and server.
- Permit the system administrator to access and manipulate system users' information.
- Permit system administrator to dynamically add candidates to be voted for in the system.
- Permit voters to login to the system and vote for their chosen candidate.
- Restrict voters from voting more than once.
- Permit a limited number of graphical images on website.

4.1.2. NON-FUNCTIONAL REQUIREMENTS

- The website to be created efficiently utilizing the system resources such as bandwidth, memory etc.
- Implement a comprehensive, adjustable and maintainable voting system.
- Implement a well presented and structured website, which is clearly visible to users.
- Ensure system development and coding is well documented for future use.

4.2. DESIGN TECHNIQUES

In order to design and build a well-structured system, it is highly vital to plan and understand how the data being inputted and outputted would be

conveyed around the system.

There are a number of tools that can be used to plot the construction of the voting system from start to finish. The use of system models would be highly essential in describing and visualizing the way in which the system would be operated.

In the case of the design for the secure online voting system, User Case diagrams would be used to how a graphical representation of how the users will be able to interact and operate the system. Data flow diagrams would be used to showcase the entire architecture of the whole system. This form of design would be very helpful to the system developers and would help in engineering the system in a consistent and efficient manner.

4.3.WEBSITE INTERFACE DESIGN

The website interface design has to be created taking the user of the system in account. The interface should be visibly distinct and precise; the user of the system should find it easy to follow the navigational structure of the website through clearly constructed navigational links on the web site.

4.3.1. WEBSITE FORMS

In order to retrieve data from the system's user's for processing by the web server, forms would have to be used. It is a medium for capturing information from the user, which can then be processed by the server or stored in a database. The forms to be used for the online voting system would be as follows.

- Add candidates : this form will be used for adding the details of the elections candidates
- Add Voter: This form will be used for adding the details of the elections voters into the database
- Add Administrator: This form will be used to add the administrators to the database system

- Voter Login: this form will be used for voter access into the system
- Admin Login: this form will be used for administrator access into the system

4.4. DATABASE DESIGN

In order to develop the online voting system, a database system has to be in place to be used to store all the data retrieved from the users of the system. The database system to be created will also play a major part for enforcing and strengthening the security of the voting system. Authentication of the system's users will rely on the details of the users which would be stored in the database system. MySQL database server has been selected as the database of choice, due to the sheer fact that it is open source which cuts the cost of having to buy database software. MySQL has a very large storage Capacity Which Will Be Essential For Storing The Large Amount Of Data To Be inputted.

4.4.1 ENTITIES & ATTRIBUTES

The database to be constructed will make use of entities and attributes as a form of structure for the database information. The entities take the form of each table to be created in the database. The tables house different fields which take the form of attributes. These attributes can be set to store certain types of data, be it text or integer values. Each entity will have an attribute which will hold a primary key, a primary key is a value that can be used to identify a unique row in a table or entity.

Entity relationship diagrams were created to show the logical structure of the database and the relationships between entities, these diagrams are located in **Appendix C**. The entity table gives a description of the entities used in the database. The entities used in the database system have been described in table 4.1.

ENTITY TABLE

Table 4.1: Entities used in the database system

ENTITY	DESCRIPTION
Administrator	The administrator table will be used to store all the details of the administrators utilizing the system. Each administrator will have a unique username. The attributes utilized in this entity are shown in figure 4-1.
Candidates	The candidates table will be used to store all the details of the elections candidates. Each candidate will have a unique username. The attributes utilized in this entity are shown in figure 4-3.
Users	The user's tables will be used to store the encrypted passwords of the voters and administrators. A field within the table called "type" will also be used to differentiate voters from administrators within the table.
Voters	The voters table will be used to store the details of each voter in the system. Due to the high security measures to be taken when developing the system, the voters table will also contain fields with the records of each candidate voted for by the voter, this design has to be done to prevent the possibility of a voter voting more than one. Through this means if there is any need for suspicion of vote rigging by the elections organizer the database table can prove that each voter voted only once. A field called "voted will also be used to record when each voter has cast their vote by incrementing to 1. A timestamp field will also be added to record the exact time each voter cast their vote.

DATABASE ENTITY & ATTRIBUTE DIAGRAM

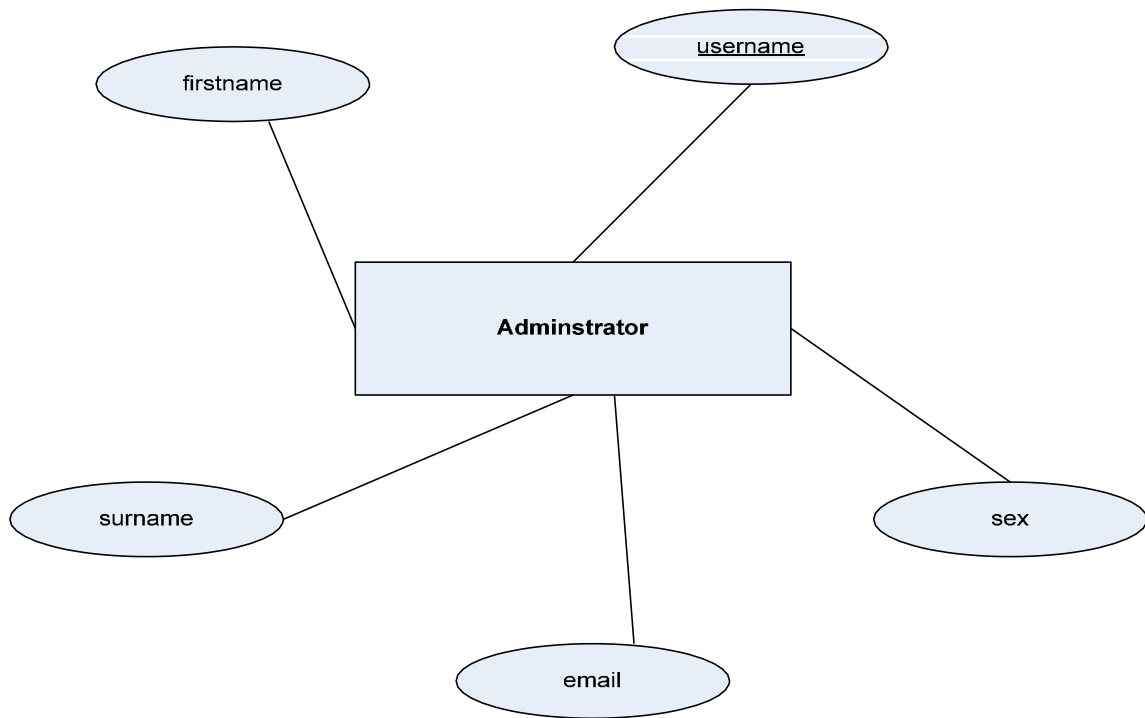


Figure 4.1: Administrator entity with its attributes

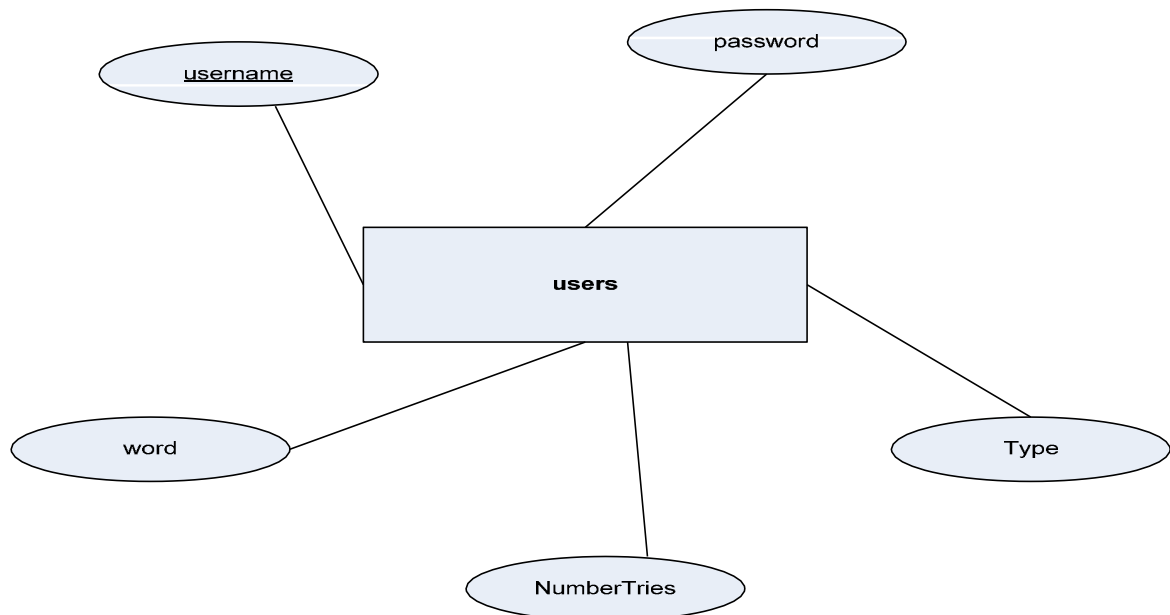


Figure 4.2: Users entity with its attributes

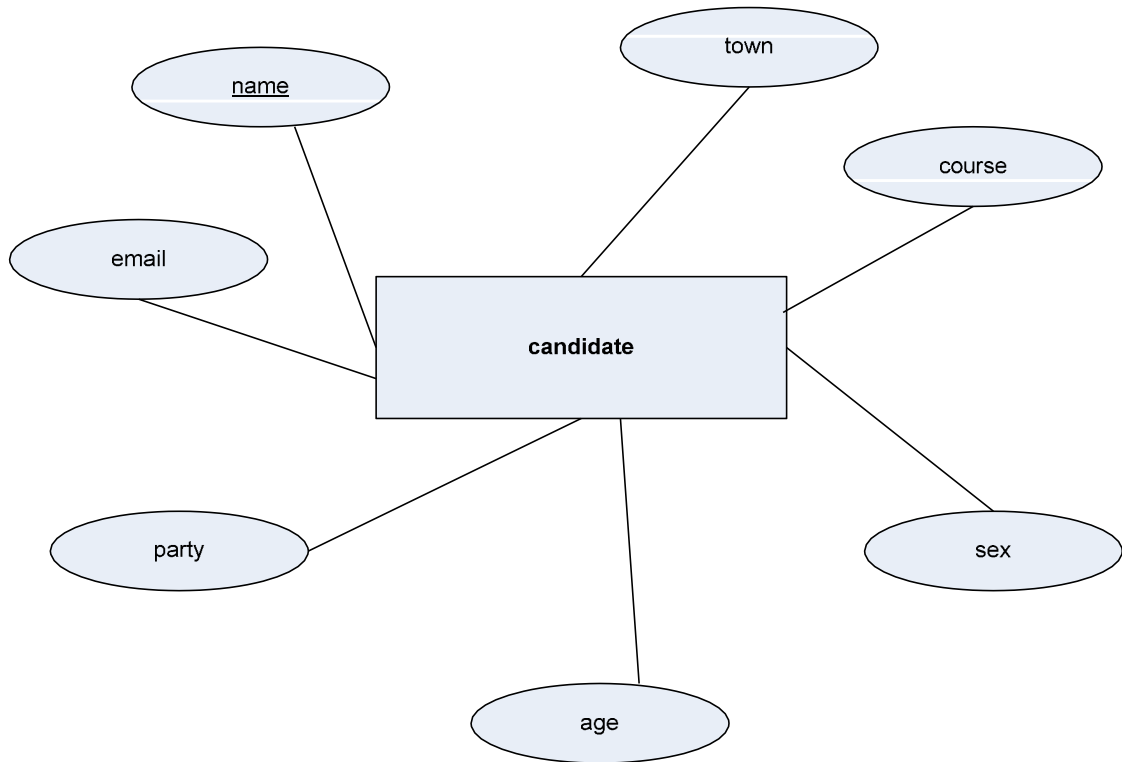


Figure 4.3: Candidate entity with its attributes

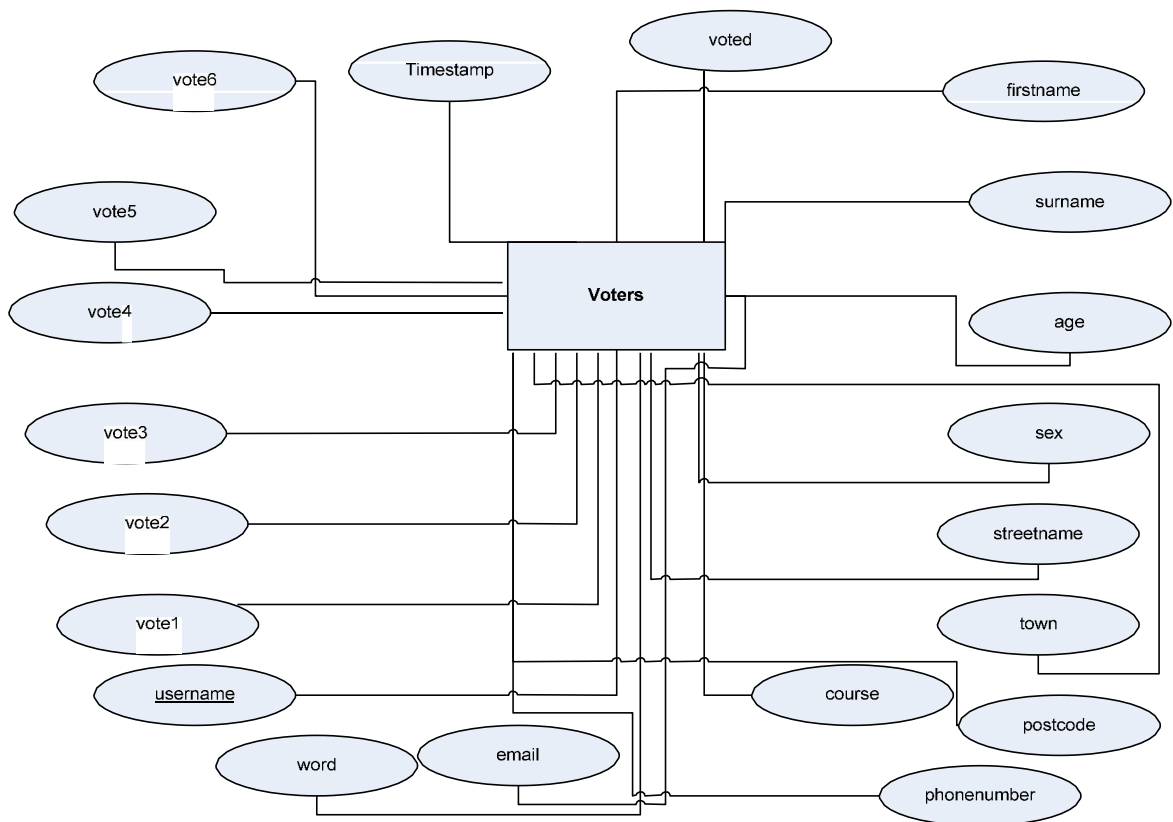


Figure 4.4: Voters' entity with its attributes

4.5. SOFTWARE DESIGN

A software design is a description of the structure of software to be implemented, the data which is part of the system, the interfaces between the user and the system's components and the algorithms utilized. The design of the system software is of top priority to the development of the entire online voting system. The software design will consider the different processes that would be involved in carrying out specific functions within the system. The system design would show the data flow within the system, in order to create a well-structured design, two design methods can be used.

STRUCTURED DESIGN: The structured software modeling is a top down function-oriented approach to software development that decomposes the system into sets of interacting functions in the process known as functional decomposition. The model consists mainly of data flow diagrams to show a system's functional structure.

OBJECT-ORIENTED DESIGN: This is a design method which involved the use of self-contained objects which can communicate with other object; this form of design can decrease the complexity of a project to the developer. Object-Oriented design UML modeling diagrams.

4.5.1. USE CASE MODELING

In order to portray the system's design to an end user like a system analyst, it would be essential to avoid the use of complex technical language but instead adopt a more end- user friendly design approach. The use case modeling technique can be used to visualize the requirements of the online voting system, depicting the scenarios that show how the system would communicate with its users.

The system to be designed will consist of actors who would be interfacing with the system's components; the main actors in the system are the **administrators** whose roles within the system are shown in figure 4.5 and the **voters** whose roles within the system. The use case model will show what system component each actor will be interacting with.

ADMINISTRATOR USE CASE DIAGRAM

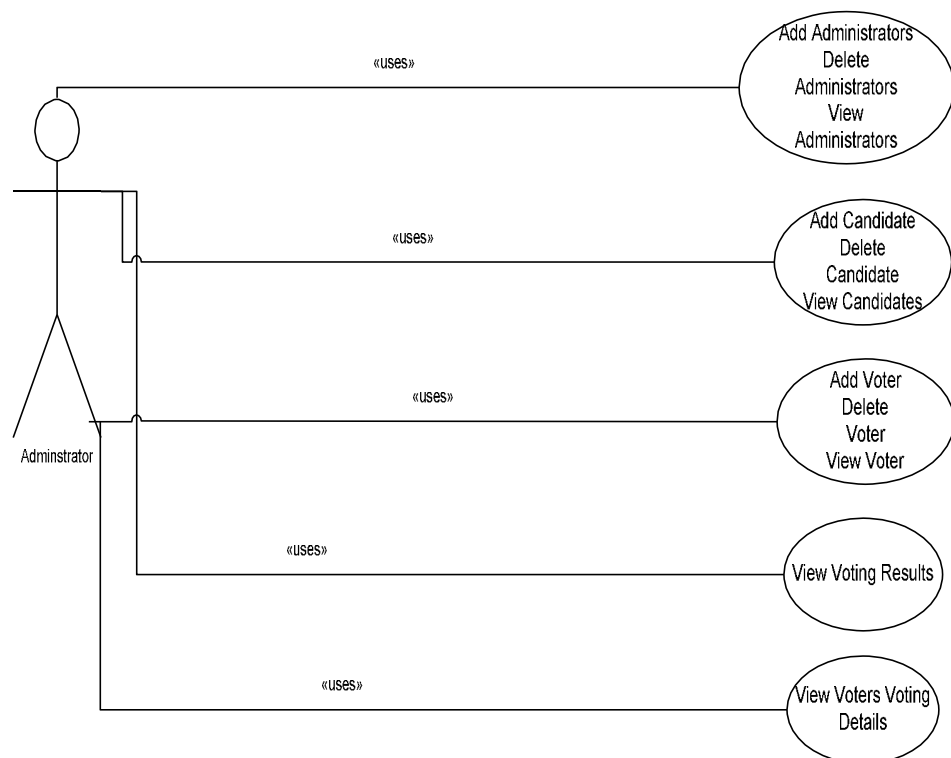


Figure 4.5: Administrator use case diagram

VOTER USE CASE DIAGRAM



Figure 4.6: Voter use case diagram

4.5.1. SYSTEM ACCESS DESIGN

The system to be implemented will need to have strong authentication features. The users of the system need to be authenticated to ensure that they have the right to use the system. For a voting system to be deemed secure, the system's login access must be well designed. The login page for the system must facilitate separate logins for administrators and voters. Each actor using the system will have their own access rights, to ensure that the voters would not be able to gain access to the administrator's page. In designing the login page, a Python class would be implemented for each actor to validate their login criteria with the data stored in the system database through the use of sql statements.

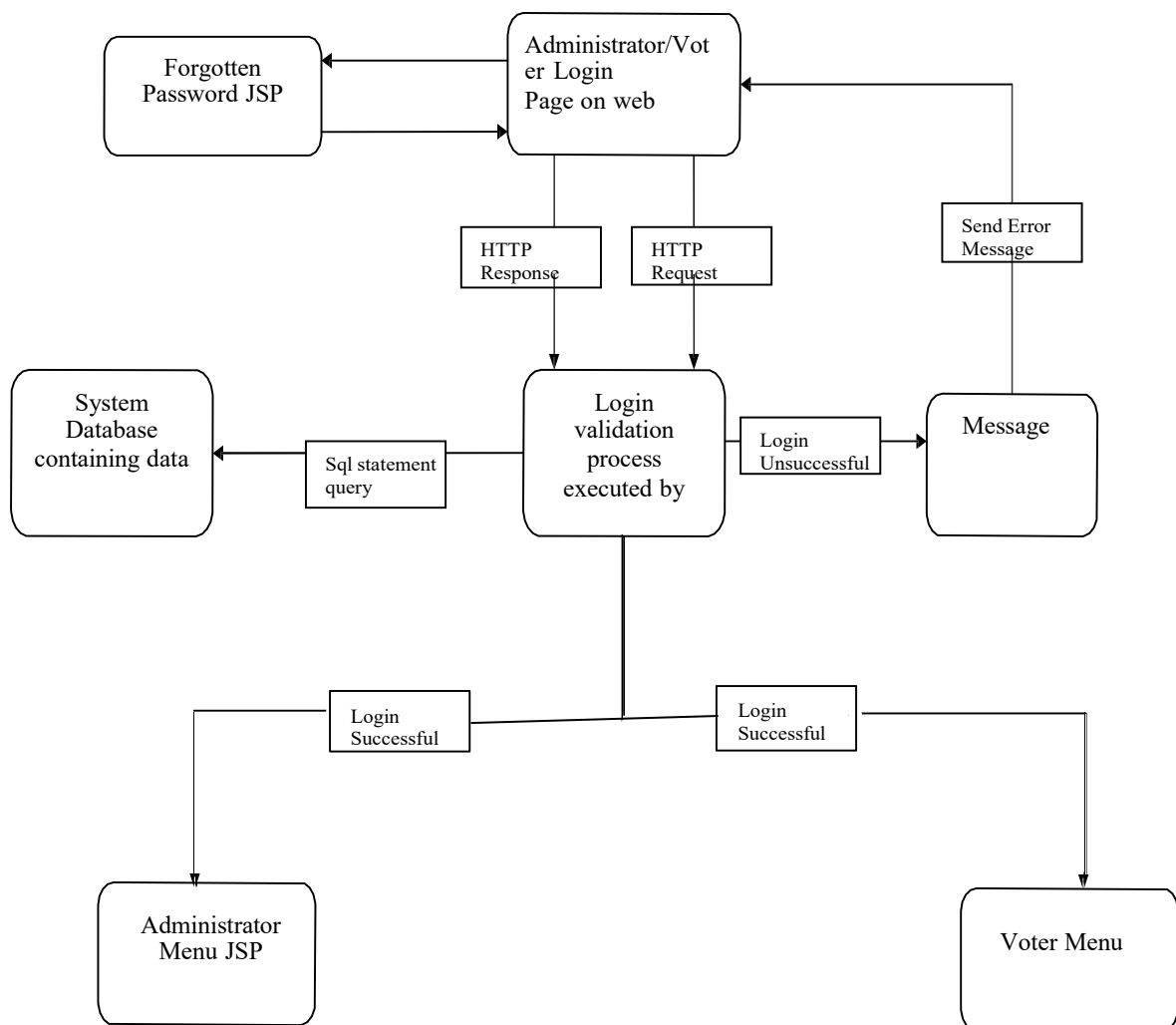


Figure 4-7: Displays the system's access login design

System users will always have a problem with forgetting their chosen passwords, it is fundamental problem which every secure system would have, and an appropriate facility has to be made to deal with this problem. In the design of the voting system, the login page for both the administrators and the voters will be linked to a page for accessing passwords that have been forgotten by the user as shown in figure 4-8, it will be a requirement for every user to have a memorable word which will be used to query the system's database and retrieve the user's password.

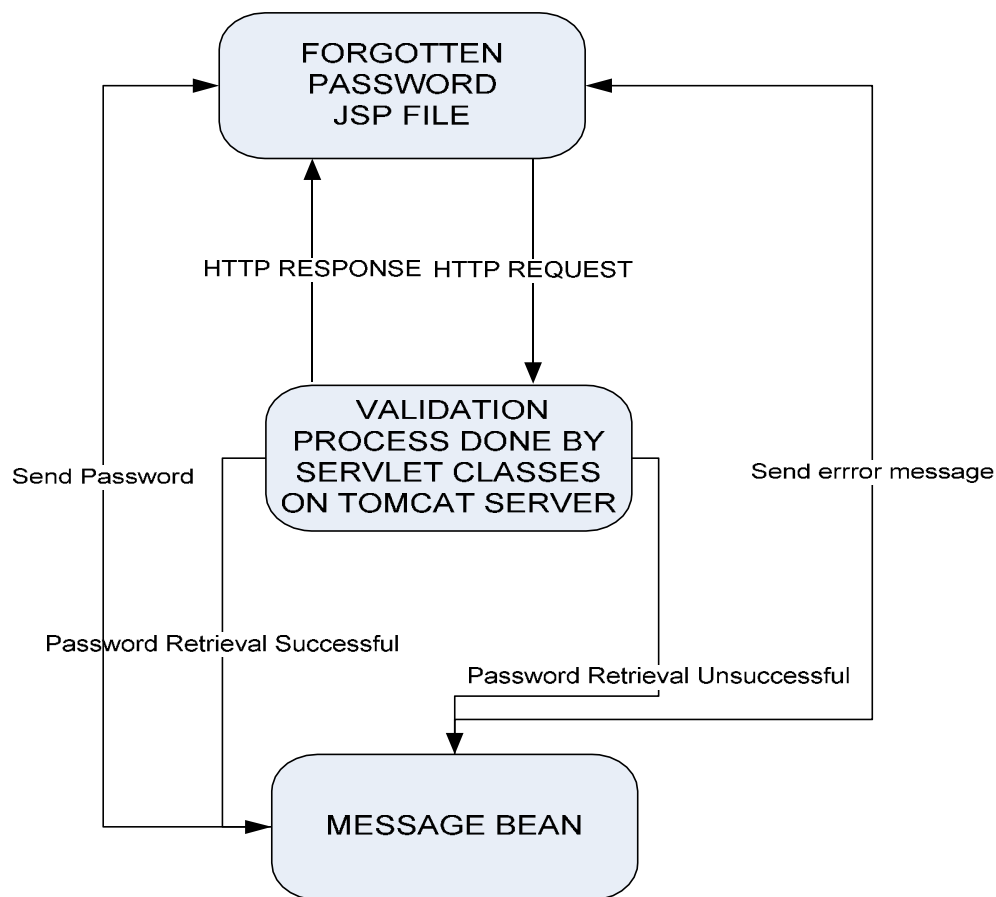


Figure 4-8: Design of the forgotten password section

4.5.2. ADMINISTRATOR & VOTER DESIGN

Once the authentication process has been carried out, the access permission given, the voter and administrator would be able to gain access to their specified facilities as shown in figure 4-9. The voter would be able to select a candidate to vote for and logout of the system, the voter would be blocked from gaining entry to the system after casting their vote. The administrator would have access rights to adding, deleting and viewing candidates, voters and administrators. Addition and deletion of candidate names will also dynamically change the candidate's names on the voter's JSP page.

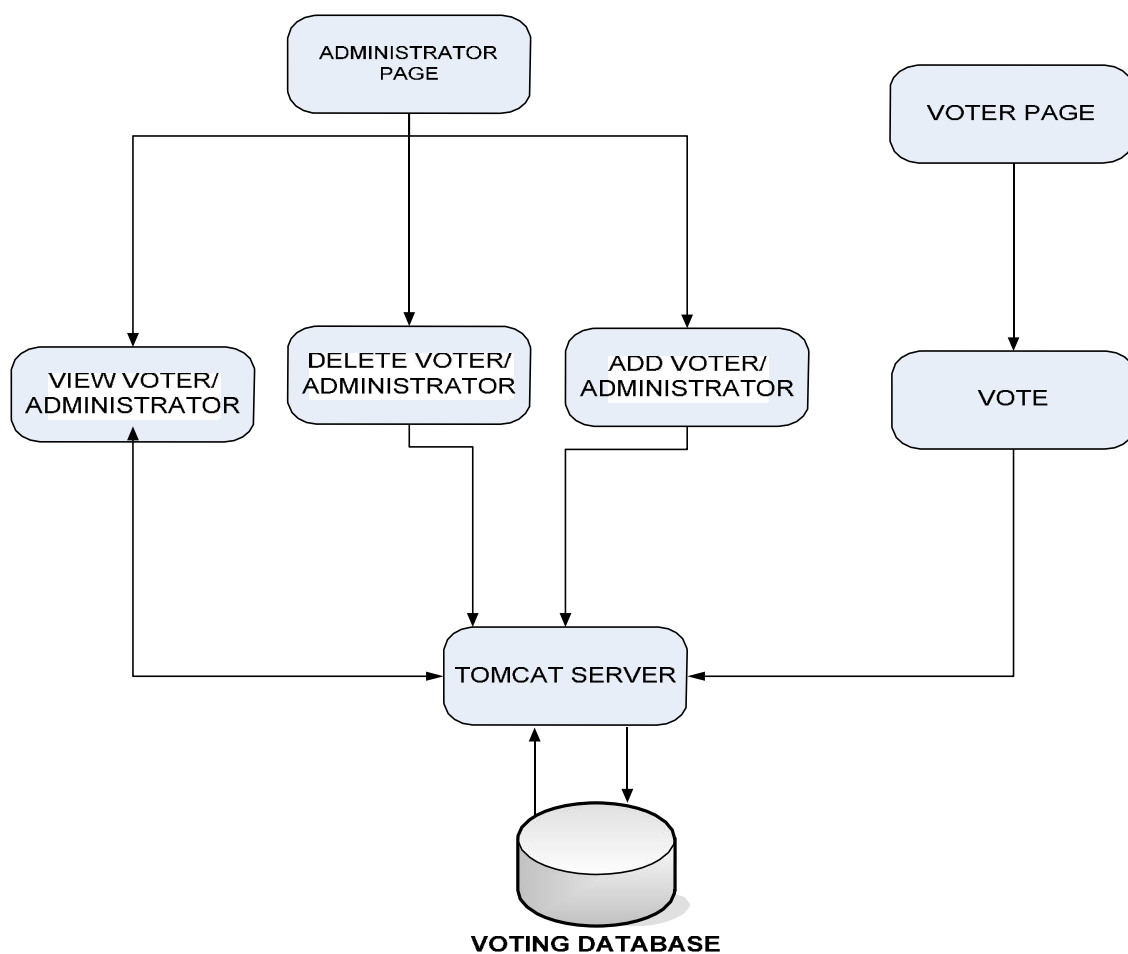


Figure 4-9: Design of the administrator and voter system features after login

4.5.3. SYSTEM SECURITY DESIGN

A top priority requirement for the proposed system is to have highly efficient and secure features, to safe guard the integrity of the voting system. In order to strengthen the system's security, three forms of security measures where engineered into the system to safeguard the data flow within the system and the information being stored in the database. The four measures to be used in addition to the system's login access facility are shown in table 4-2.

Table 4-2: Security features to be implemented in the system

System access attempts log	This security measure would enable the database to record the number of attempts a user tries to log into system with a wrong password. The system should lock the user out, if the attempted tries exceeds a certain number. This measure will be used to prevent password guessing
Password Encryption	This security measure would be used to encrypt password entered into the system through the use of an encryption algorithm, thus in the case of the database being compromised, the password stored would be useless. In order for a user to obtain their password from the forgotten password page, a decryption method would be use to decrypt the encrypted password back to its original form to be displayed to the user
Secure Socket Layer (SSL) Transmission	The form of security would ensure that any data inputted from the users web browser would be encrypted and useful to a hacker who would want to acquire information

4.5.4. SYSTEM ARCHITECTURE

The architecture of the system would comprise of a number client and server side technologies working together.

FRONT END

The front end of the system will represent the user's web browser interface of the voting system; this is where the users will be able to send HTTP requests and receive HTTP responses from the server. In order to build the front end of the system, HTML would be utilized.

BACKEND

The back end of the system will represent the server side of the application; this is where the processing of HTTP requests sent from the client will take place. A Tomcat server engine will be used to load the servlet and jsp's, which can then send requests from the tomcat server engine, the dynamic content will then be sent back to the client in HTML format to enable the client to view the information on the web page.

A database will be utilized to store data sent from the client of the system, MySQL database will be used to store data being used by the system.

In order for the database to be able to retrieve information from the server, a middleware layer has to be established in form of the JDBC API driver which will be used to translate Python methods calls to database API calls.

CHAPTER-5

IMPLEMENTATION

The implementation of the online voting system is the most essential phase of the development of the online voting system. The implementation phase will focus on the technologies and resources used to deploy and develop the online voting system. The implementation phase would be described in sections starting from the point of entry within the system, to the various processes conducted through out the system.

5.1.SERVER SETUP & CONFIGURATION

The first step was to download and configure the web/application server to be used, since tomcat was the chosen server for the project, it had to be downloaded from the apache tomcat web site which permit server downloads for free. There are a number of steps that have to be taken in order to configure the tomcat server to a computer system. The first step is to download the appropriate Python development kit from the Sun Micro systems website. On downloading the JDK software, an environment variable will be set as PYTHON HOME to the path name of the directory in which the specified JDK was installed; the JDK version used for the project was JDK 1.5.0. The environment variables are located in the systems properties box in Windows XP operating systems.

Upon completing the installation of the Python development kit, the next step would be to download and install the Tomcat server. The Apache Software Foundation web site was a wide variety of servers to download with different versions, to be on the safe side a current version Tomcat 5.0 was chosen for download. Once the installation and configuration is completed, the server can be started and stopped by simply clicking on the tomcat option on the start menu bar. The web browser can then be started

up by using the URL <http://localhost:8080>. The number indicates which port the web server is connected on. The port number can be changed on the server.xml file which is located in the config directory.

5.1.1.SSL CONFIGURATION ON TOMCAT

Since security is a top priority for the online voting system, Secure Socket Layer (SSL) transmission is highly vital to the security of the system. A good advantage of using the Tomcat server is that it can be configured to perform SSL, thereby protecting the system user's information from being snooped on by an illicit hacker. SSL is configured by firstly setting up a keystore which would be used to house the keys and certificates used for cryptographic purposes by SSL. The keytool command line utility will be used to create a new keystore from scratch which would contained a self signed certificate that can be viewed by the system user before entering the system's website. The keytool command line is displayed below.

```
C:\tomcat\jakarta-tomcat-5.5.9>keytool -genkey -alias tc-ssl -keyalg  
HOMOMORPHIC keystore - server.keystore.
```

The command line above will create a new file in the tomcat directory which will be named server.keystore. After the key tool command is entered, the user will be prompted for a keystore password. When the password is entered, the user will be prompted to enter a number of other identification related information which would be used in the certificate. The password chosen has to be reflected in the server.xml configuration file as shown in figure 5-1. The remote host connection port number to be used for the secure connection would be 8443. The user would be able to startup the secure connection from their web browser using the URL <https://localhost:8443>.


```
<Connector port="8443" maxHttpHeaderSize="8192"
    maxThreads="150" minSpareThreads="25" maxSpareThreads="75"
    enableLookups="false" disableUploadTimeout="true" acceptCount="100" scheme="https"
    secure="true" clientAuth="false" sslProtocol="TLS" emptySessionPath="true"
    keystoreFile="conf/server.keystore" keystorePass="bondito"/>
```

Figure 5.1. The code above displays server.xml file,

5.2. SYSTEM IMPLEMENTATION

The system was implemented using a combination of JSP pages, Python beans and Python Servlets classes. The Python classes used where in between the JSP web site and the database. A combination of Python servlets and scriptlets which are embedded Python code in a jsp file where used to query the database. The system validation and authentication of the users where implemented through the use of Python classes. The Python servlet class files are the backbone of the system, and form an integral part of the development process of the entire system. Python beans class where also widely used in the system development as a form of temporary data storage.

The description of every step taken to develop the main architecture will be described starting from the user logging into the system, validation and authentication and also the voting process conducted.

5.3.SERVLET CONFIGURATION

Servlets are registered and configured as part of the voting web application. In order for a servlet to be registered, several entries have to be entered into the web application deployment descriptor. A deployment descriptor is used to describe resources in the web application. XML is used to write to these deployment descriptors. The deployment descriptor used for the Online Voting System was the web.xml file located in the WEB-INF directory.

The first entry under the `< servlet >` element defines a name for the servlet and specifies the compiled class that executes the servlet. This element also contains definitions for initialization parameters and any security roles the servlet may have. The second entry under the `< servlet-mapping >` element, defines the URL pattern that calls this servlet, as shown in figure 5.2.

In order to control the way servlets are accessed in the web application, servlet mapping can be utilized.

```
<servlet>
    <description>
    </description>
    <display-name>
    AdminLogin</display-name>
    <servlet-name>AdminLogin</servlet-name>
    <servlet-class>
    evote.AdminLogin</servlet-class>
</servlet>
<servlet-mapping>
    <servlet-name>AdminLogin</servlet-name>
    <url-pattern>/AdminLogin</url-pattern>
</servlet-mapping>
```

Figure 5.2: Servlet Mapping

5.4.PYTHON CLASS IMPLEMENTATION

Due to the fact that the database is the storage component which facilitates loads of functions within the system, it would be essential to describe how the servlet communicate with the database system.

The JDBC driver would be installed in the lib directory found in the WEB-INF folder of the tomcat server.

The name of the class used is `com.mysql.jdbc.Driver`. This method would register the JDBC driver in the JDBC driver manager which is housed in the `databaseconnectivity` class as shown in figure 5.3.

```
// Register JDBC/ODBC Driver in jdbc DriverManager
Class.forName("com.mysql.jdbc.Driver").newInstance();

Connection c =
DriverManager.getConnection("jdbc:mysql:///Voting", "", "");
Statement st = c.createStatement();
if (Username.equalsIgnoreCase("")) {
    message
    .setMessage("Please enter a username");
    return false;
```

Figure 5.3: code from the database connection class displaying database connectivity

The system Python classes where divided into a number of packages. Packages are groups of related classes located in the same directory. Each package housed classes used for the system's processes. The three packages used where as follows

Datasource: Used to group the dataconnection and encryption classes

Bean: Used to group the Python bean classes

Evote: Used to group the Python servlets used for processing http requests and responses.

The system was designed to have two login pages, one for the voter and other for the administrators, each login page was created as a jsp page, once the user enters their login details and clicks the submit button, the users login data is sent to the database connection servlet for authentication and validation as shown in figure 5.4.

```
// Method to add Administrators to database
public static void addAdmin(String name, String surname, String email, String
sex, String username, String password, String word) {
    try {
        //If age not specified, default to 0
        // Register JDBC/ODBC Driver in jdbc DriverManager
        Class.forName("com.mysql.jdbc.Driver").newInstance();
        Connection c =
DriverManager.getConnection("jdbc:mysql:///Voting");
        Statement st = c.createStatement();
        st.executeUpdate("insert into users (username, password, Type,
NumberTries, word) VALUES (" + username + ", " + password + ", 2, 0, " + word +
"");
        st.executeUpdate("insert into administrator (firstname, surname,
email, sex, username) VALUES (" + name + ", " + surname + ", " + email + ", " +
sex + ", " + username + "");
```

Figure 5.4: The code above used for adding administrators into the database

The database connection class methods are also used by other classes for various functions. Due to the vastness of the system implemented, I have broken the processes development for the system into two, the admin implementation process and the login implementation process. These two development processes would be appropriately explained in a concurrent manner.

5.5.ADMIN IMPLEMENTATION

Index.html page: acts as the welcome page which provides links to the voter and admin login pages.

AdminLogin jsp page: provides dynamic access to the system, it is responsible for sending http requests to the adminlogin servlet in form of the

login data retrieved from the administrator, the page is also used to set the value of the properties of the message Python bean in a tag , which displays errors message from the database connection class and admin login class.

Admin Login Servlet class: retrieves http requests from the Admin Login jsp page, gets the username and password parameters, it encrypts the password parameter by calling the Encryption. encrypt method from the Encryption class located in the data source package. The servlet ensures that the user is of type two, which is the type number for the administrators by calling the Authenticate method in the database connection class. If the user is an administrator the user is then permitted access to the Admin Menu jsp page.

Admin Menu JSP page: contains all the links to the add, delete and view candidates, voters and administrator jsp pages. It also contains a link to the welcome page.

Add Voter JSP page: is used to add voters details into the database dynamically. Contains a tag used to declare and instantiate the Python bean class voters Form Bean, the bean id is Voters which is the name of the object. It also contains a message bean which displays errors on the page.

Voters Form Bean class: is used to get the data of voters from the add voter jsp page and set the data with the bean. This data is temporarily stored in the bean to be used by the Add Voter Servlet.

Add voter Servlet: This class imports methods from the data connection, Encryption, message, voter Form Bean classes. It sets the data derived from the Voter Form Bean, and also sets the message Python bean. The Add Voter class conducts validation processes from the data retrieved from the Voter Form Bean, If the password retrieved does not match the repeat password, it sends a message bean error to the Add Voter jsp page.

In order to check if the username chosen by the voter is already taken,

the database is queried by importing the check Username method from the database connection class. If the username is already taken, an error message is sent using the message Python bean and sends the user back to the Add Voter jsp page. If the validation is complete, the Add Voter class calls the encrypt method from the Encryption class and uses it to encrypt the password. The class then calls the method Add Voter from the database connection class to add voters details in the voting database.

Delete Voter JSP page: would be used to display the details of the voters that are to be deleted from the database. Firstly the variables which would be used to encapsulate the data from the Voters table in the Voting database are declared as strings and given null values.

Deleted Voters JSP page: page is used to delete voters from the database through the use of Python code embedded in script tags. Firstly the jsp page requests for the string email from the delete voter page. The JDBC driver is registered to the JDBC driver manager and a connection is made to the voting database. The execute update method is used to execute the DELETE SQL statement which would delete from the Voters table where email field would be the string email. The execute update method is typically used with the INSERT, UPDATE & DELETE SQL statements

Voting Results JSP page: was used to count and display the voting results to the administrator. The variables Vote1="" and count are declared as strings with null values. The JDBC driver is registered and a connection is made to the database. The execute query method is used to select the vote1 record from the voters table and then count the vote1 record using the SQL COUNT statement, the COUNT statement will only count those records in which the table fields in the brackets is NOT NULL. The result of the query is stored in the Result set object "rs1".

All Voting Results JSP page: displays the voter's first name and surname with the names of the candidates they voted for. This is a security

measure, which would enable the administrator to view the voters' details and who they voted for. The JSP uses the execute query method to select the voter's first name, surname and voter's chosen candidates' details from the voters table, and prints to screen.

5.6.VOTER IMPLEMENTATION

The login method used for the voter access is similar to the administrator access, the difference with the authentication used for the voter access. Login servlet class checks the voter to see if they have voted or not, if the voter has not voted and the voted field in the voters table is equal to "0", the voter is permitted to enter the voter menu page, if the voter has voted before and the voted field in the voters table has been flagged as "1", the voter will be blocked from voting and directed to the Not Allowed jsp page.

Voter Menu JSP page: displays the candidate name in a drop down box for the voter to select. An executequery method is used to select the name of the candidate who is under a certain party in the candidate table. The result is stored in the results set object "rs" it is then read by the next() method which prints out the result in a while loop. The voter would then be able to select an option from the drop down box. The option selected is then sent to the vote servlet for processing.

Voter Servlet: gets the http request information from the voter menu jsp page and stores the data as strings. The data is then stored in the Voter Form Bean. Voter is directed to the confirmation jsp page through the use of the request dispatcher method.

Confirmation Vote JSP page: would have the Voter Form Bean tag declared. The execute Update method would be used to set the voter's voted field record in the voters table to one, once the vote has been cast. This is a security measure which would prevent voters from voting more than once. Once the voter has confirmed their choice of candidates, the data in the

voters table is updated to correspond to the new candidate information.

The voter is directed to the thankyou servlet. If the voter wants to change their choice of candidate, the voter can click the back button which would send a http request to the back vote servlet, the servlet in turn would direct the voter back to the voter menu jsp page.

5.7. ENCRYPTION CLASS IMPLEMENTATION

The encryption class was used to encrypt passwords that were fed into the database table users and it was also used for decrypting passwords that were displayed to the user in the forgotten password JSP page. In order to implement the encryption and decryption processes for the voting system, the JCE API was used. The JCE makes use of specific classes to perform encryption and decryption functions. The main class for encryption is the Cipher class, which takes the role of a cryptographic cipher. A transformation is the process of encrypting and decrypting data. A transformation always includes the name of the cryptographic algorithm. The DES symmetric key encryption algorithm would be used for the encryption process.

The getInstance method takes arguments for the transformation process. When the Cipher object is returned by the getInstance method, it must be initialized before it can be used to encrypt or decrypt the password string, in order to perform this function the secret key would be needed. Due to the symmetric nature of the DES encryption algorithm one secret key would be used to encrypt and decrypt the password.

In order to create a DES key, a Key Generator would have to be instantiated for the DES algorithm. The class SecureRandom which provides a cryptographically strong pseudo-random number generator would be used to create a number that would be initialized to the generator object. The generateKey method would be used to generate the secret key. The key would then be stored in a file called OnlineVoting.ser. This method would

now return the key to any method that calls it.

Upon generating a secret key, the cipher object would be initialized using the key from the get key method and set to either ENCRYPT_MODE for encryption or DECRYPT_MODE for decryption.

CHAPTER-6

TESTING

In order to ensure that the system works perfectly, it has to be rigorously tested. The testing procedure would be used to check all the features developed for the online voting system work efficiently, the test procedure would also be used to identify any hidden errors or deficiencies the system may possess.

In order to conduct an efficient testing process on the system, a suitable testing procedure has to be utilized. In choosing an appropriate testing strategy to use, some considerations have to be reviewed in terms of the size and complexity of the system to be tested.

6.1. TEST STRATEGIES

There is a number of testing strategies that can be utilised to conduct adequate testing processes, the black box and white box testing methods are the most popular methods used to test software developed systems.

6.1.1. TEST PLAN

In order to efficiently test the full functional capability of the online voting system, a test plan has to be created. The test plan created would break the testing processes in order to tackle any malfunctioning feature of the online voting system.

The testing process would focus on testing the system's server, database server and web pages on different web browsers. This test has to be carried out to ensure that the system would be able to function on any web browser utilized by the system's users.

The testing process would focus on the system login authentication features; this is an integral part of the system, because it ensures security of the system is upheld against unauthorized access. A test would be carried out

to check if the password being utilized are encrypted and decrypted.

The system's form validation would also have to be tested to ensure the error message to be presented to the user if the forms are not filled correctly is functioning appropriately. The system database engines which connect the application to the database system have to be tested to ensure that information being retrieved from the users are populating the database system

6.2. TEST DATA

Table 6-1: Testing features

Test Ref No	Test Data	Expected Outcome	Final
1	Connect to server	The Client should be able to Connect to	Pass
2	Connect to mysql database	The Client should be able connect to data base	Pass
3	Test SSL connection from web browser	Using the secure local host port number 8443, the user should be able to enter the website over a secure connection	Pass
4	Login validation	Error message should be displayed when inappropriate data is entered	Pass
5	Login process to distinguish voter and administrator	Voter should not be allowed to login into admin page, admin should not be allowed to	Pass

6	Attempt to guess password more than three time during login	System user should be blocked from accessing the system on third attempt	Pass
7	Voter casts votes	The voters table in voting database should be updated with new votes	Pass
8	Login Block for voter	Voter who has voted once should be flagged and blocked from voting again	Pass
9	Voting results page	The votes counted should be updated when new vote is cast	Pass
10	Password encryption	Password entered into database should be encrypted	Pass
12	Administrator should be able to delete voter, candidate and administrator	Details of voter, candidate & administrator should be deleted from database	Pass
13	Password Decryption	Forgotten password request by user should be decrypted before being sent to user	Pass
14	Logoff	User should be able to log off successfully from system	Pass

CHAPTER-7

SOURCE CODE

MANAGE.PY

```
#!/usr/bin/env python
"""Django's command-line utility for administrative tasks."""
import os
import sys

def main():
    os.environ.setdefault('DJANGO_SETTINGS_MODULE', 'Election.settings')
    try:
        from django.core.management import execute_from_command_line
    except ImportError as exc:
        raise ImportError(
            "Couldn't import Django. Are you sure it's installed and "
            "available on your PYTHONPATH environment variable? Did you "
            "forget to activate a virtual environment?"
        ) from exc
    execute_from_command_line(sys.argv)

if __name__ == '__main__':
    main()
```

RSALIBRARY.PY

```
#!/usr/bin/env python3
# encoding: utf-8

import random
import Euclidean
import PrimalityTests
import Modulo
import ExtendedEuclidean
import math
import StringLibrary
import TimeLibrary

# following miller-rabin test
def is_primary(n):
    l = [2, 3]
    if n < 2:
```

```

        return False
    if n in [2,3,5,7,11]:
        return True
    bits = math.ceil(math.log2(n))
    if bits < 2:
        return False
    l.append(random.getrandbits(bits // 2))
    l.append(random.getrandbits(bits // 2))
    l.append(random.getrandbits(bits // 2))

    for a in l:
        if PrimalityTests.is_composite_miller_rabin(n, a):
            return False

    return True

def generate_random_primary(bits=512):
    max_iter = 5000
    while True:
        rnd = random.getrandbits(bits)
        if is_primary(rnd):
            return rnd

        max_iter -= 1
        if max_iter <= 0:
            raise Exception("Cannot find a primary number")

def generate_public_key_e(phi, min_value=3):
    if min_value % 2 == 0:
        min_value += 1
    for i in range(min_value, phi, 2):
        if Euclidean.gcd(i, phi) == 1:
            return i
    raise Exception("Couldn't find E!")

def compute_private_key_d(e, phi):
    g, x, y = ExtendedEuclidean.extended_euclidean(e, phi)

    d = x % phi
    return d

def generate_keys(bits=512, min_e_value=3, return_p_q=False):
    p = generate_random_primary(bits)
    q = generate_random_primary(bits)
    n = p * q
    phi = (p - 1) * (q - 1)

```

```

e = generate_public_key_e(phi, min_value=min_e_value)
d = compute_private_key_d(e, phi)
if return_p_q:
    return n, e, d, p, q
return n, e, d

def encrypt_number(n, e, number):
    if number >= n or number < 0:
        raise Exception("Number should be greater than 0 and less than N")

    return Modulo.pow_mod_binary(number, e, n)

def decrypt_number(n, d, cipher_number):
    return encrypt_number(n, d, cipher_number)

def decrypt_number_crt(d, p, q, cipher_number):
    n = p * q
    mp = Modulo.pow_mod_binary(cipher_number, d % (p - 1), p)
    mq = Modulo.pow_mod_binary(cipher_number, d % (q - 1), q)

    g, yp, yq = ExtendedEuclidean.extended_euclidean(p, q)

    m = (Modulo.mult_mod([mp, yq, q], n) + Modulo.mult_mod([mq, yp, p], n)) % n

    return m

def encrypt_text(n, e, text):
    l = []
    for c in text:
        l.append(encrypt_number(n, e, ord(c)))

    return l

def decrypt_text(n, d, cipher_text_list):
    s = ""
    for cipher_number in cipher_text_list:
        o = decrypt_number(n, d, cipher_number)
        s += chr(o)

    return s

def encrypt_text_to_text(n, e, text, sep="|"):
    l = encrypt_text(n, e, text)
    l = [str(n) for n in l]
    return sep.join(l)

```

```

def decrypt_text_from_text(n, d, cipher_text, sep="|"):
    l = cipher_text.split(sep)
    l = [int(s) for s in l]
    return decrypt_text(n, d, l)

def encrypt_text_v2(n, e, text):
    if len(text) >= math.log2(n) / 8: # length of the text is more than the number of bytes in N
        raise Exception("Text is longer than n({})".format(n))

    plain = StringLibrary.text_to_integer(text)
    return encrypt_number(n, e, plain)

def decrypt_text_v2(n, d, cipher_number):
    if cipher_number >= n:
        raise Exception("Cipher number({}) is longer than n({})".format(cipher_number, n))

    plain = decrypt_number(n, d, cipher_number)
    return StringLibrary.integer_to_text(plain)

def test():
    print("Generating keys...")
    n, e, d, p, q = generate_keys(bits=512, min_e_value=3, return_p_q=True)
    print("n=", n)
    print("e=", e)
    print("d=", d)
    print('-' * 20)

    print("Encrypting a number:")
    number = random.getrandbits(8)
    print("Number to send:", number)

    cipher_number = encrypt_number(n, e, number)
    print("Cipher:", cipher_number)

    decrypted = decrypt_number(n, d, cipher_number)
    print("Decrypted:", decrypted)

    decrypted_crt = decrypt_number_crt(d, p, q, cipher_number)
    print("Decrypted using CRT:", decrypted_crt)

    print('-' * 20)
    print("Normal vs CRT performance:")

    try:
        ts_normal = 0

```



```

ts_crt = 0
for i in range(300, 1000):
    _, t = TimeLibrary.timed_function(decrypt_number, n, d, cipher_number)
    ts_normal += t
    _, t = TimeLibrary.timed_function(decrypt_number_crt, d, p, q, cipher_number)
    ts_crt += t

print("Total time for normal:", TimeLibrary.beautify_time(ts_normal))
print("Total time for CRT:", TimeLibrary.beautify_time(ts_crt))
print("Rate: %f times" % (ts_normal / ts_crt))
except:
    print("Failed!")
    pass

print('-' * 20)

print("Encrypting a text:")
text = "This is a test message..."
print("Text to send:", text)

cipher_text = encrypt_text(n, e, text)
print("Cipher Text:", cipher_text)

decrypted = decrypt_text(n, d, cipher_text)
print("Decrypted Text:", decrypted)
print('-' * 20)

print("Encrypting a text to text:")
text = "Another text message!"
print("Text to send:", text)

cipher_text = encrypt_text_to_text(n, e, text)
print("Cipher Text:", cipher_text)

decrypted = decrypt_text_from_text(n, d, cipher_text)
print("Decrypted Text:", decrypted)

print('-' * 20)
print("Encrypting a text to number (v2):")
text = "This third text is going to be encrypted to a single number."
print("Text to send:", text)

cipher_number = encrypt_text_v2(n, e, text)
print("Cipher Number:", cipher_number)

decrypted = decrypt_text_v2(n, d, cipher_number)
print("Decrypted text:", decrypted)

```

```
#####
```

```
if __name__ == "__main__":
    test()
```

MAIN.PY

```
#!/usr/bin/env python3

from flask import Flask, render_template, request
import sys
from flask import Flask, render_template, request
import jsonify
import requests
import pickle
import numpy as np
import sklearn
from sklearn.preprocessing import StandardScaler
# numpy
import numpy as np
# classifier
from sklearn.feature_extraction.text import CountVectorizer, TfidfVectorizer
import os
from random import shuffle
import pandas
from sklearn import model_selection, preprocessing, naive_bayes
import string
from sklearn.decomposition import LatentDirichletAllocation
from sklearn import svm
from sklearn.metrics import accuracy_score
from sklearn.naive_bayes import GaussianNB
from sklearn.model_selection import cross_val_score
from sklearn.model_selection import KFold
from sklearn.ensemble import RandomForestClassifier
from flask import Flask, render_template, url_for, request
import pandas as pd
import pickle
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.naive_bayes import MultinomialNB
import joblib
app = Flask(__name__)
from flask import Flask, request, jsonify, render_template

sys.path.append('../library')
import RSALibrary
from TimeLibrary import timed_function_beautified

app = Flask(__name__)

n, e, d, p, q = 0, 0, 0, 0, 0
```

```
bits = 512
min_e = 3
```

```
@app.route('/')
@app.route('/first')
def first():
    return render_template('first.html')
@app.route('/login')
def login():
    return render_template('login.html')
```

```
@app.route('/prediction1')
def index():

    global n, e, d, min_e, bits
    context = {
        'n': n,
        'e': e,
        'd': d,
        'min_e': min_e,
        'bits': bits
    }
    return render_template('index.html', **context)
```

```
@app.route('/genkeys_pre')
def generate_keys_pre():
    global n, e, d, min_e, bits, p, q
    min_e_param = request.args.get('min_e', "").strip()
    bits_param = request.args.get('bits', "").strip()
    if min_e_param:
        min_e = int(min_e_param)
    if bits_param:
        bits = int(bits_param)
    # (n, e, d), time_needed = timed_function beautified(RSALibrary.generate_keys, bits, min_e)
    context = {
        'n': n,
        'e': e,
        'd': d,
        'p': p,
        'q': q,
        'min_e': min_e,
        'bits': bits
    }
    return render_template('generate_keys.html', **context)
```

```
@app.route('/genkeys')
```

```

def generate_keys():
    global n, e, d, min_e, bits, p, q
    min_e_param = request.args.get('min_e', "").strip()
    bits_param = request.args.get('bits', "").strip()
    if min_e_param:
        min_e = int(min_e_param)
    if bits_param:
        bits = int(bits_param)
    if bits > 2048:
        raise Exception("Max bits is 2048")
    (n, e, d, p, q), time_needed = timed_function_beautified(RSALibrary.generate_keys, bits, min_e,
True)
    context = {
        'n': n,
        'e': e,
        'd': d,
        'p': p,
        'q': q,
        'min_e': min_e,
        'time_needed': time_needed,
        'bits': bits
    }
    return render_template('generate_keys.html', **context)

```

```

@app.route('/enterkeys', methods=['GET', 'POST'])

```

```

def enter_keys():
    global n, e, d, min_e, bits, p, q
    saved = False
    if request.method == 'POST':
        n = int(request.form.get('n'))
        e = int(request.form.get('e'))
        d = int(request.form.get('d'))
        p = int(request.form.get('p'))
        q = int(request.form.get('q'))
        saved = True

    context = {
        'n': n,
        'e': e,
        'd': d,
        'p': p,
        'q': q,
        'saved': saved
    }
    return render_template('enter_keys.html', **context)

```

```

@app.route('/crypto', methods=['GET', 'POST'])

```

```

def crypto():

```

```

if n <= 0 or (e <= 0 and d <= 0):
    return index()
context = {
    'n': n,
    'e': e,
    'd': d,
    'p': p,
    'q': q,
}
return render_template('crypto.html', **context)

@app.route('/crypto_number', methods=['GET', 'POST'])
def crypto_number():
    if request.method != 'POST':
        return crypto()
    input = int(request.form.get('input')) if request.form.get('input') else 0
    keytype = request.form.get('keytype')
    number output crt, time needed crt = None, None
    if keytype != 'private':
        keytype = 'public'
        # number output = RSALibrary.encrypt number(n, e, input)
        number output, time needed = timed function beautified(RSALibrary.encrypt number, n, e,
input)

    else:
        number output, time needed = timed function beautified(RSALibrary.encrypt number, n, d,
input)
        if p > 0 and q > 0:
            number output crt, time needed crt =
timed function beautified(RSALibrary.decrypt number crt, d, p, q,
input)

    context = {
        'n': n,
        'e': e,
        'd': d,
        'p': p,
        'q': q,
        'input': input,
        'number output': number output,
        'time needed': time needed,
        'number output crt': number output crt,
        'time needed crt': time needed crt,
        'keytype': keytype
    }
    return render_template('crypto.html', **context)

@app.route('/crypto_text', methods=['GET', 'POST'])

```

```

def crypto_text():
    if request.method != 'POST':
        return crypto()
    input_text = request.form.get('input_text')
    keytype = request.form.get('keytype')
    if keytype != 'private':
        keytype = 'public'
        text_output, time_needed = timed_function_beautified(RSALibrary.encrypt_text_v2, n, e,
input_text)
    else:
        text_output, time_needed = timed_function_beautified(RSALibrary.encrypt_text_v2, n, d,
input_text)

    context = {
        'n': n,
        'e': e,
        'd': d,
        'p': p,
        'q': q,
        'input_text': input_text,
        'text_output': text_output,
        'time_needed': time_needed,
        'keytype': keytype
    }
    return render_template('crypto.html', **context)

@app.route('/crypto_text_dec', methods=['GET', 'POST'])
def crypto_text_dec():
    if request.method != 'POST':
        return crypto()
    input_text_dec = int(request.form.get('input_text_dec')) if request.form.get('input_text_dec') else 0
    keytype = request.form.get('keytype')
    print(type(keytype))
    if keytype != 'private':
        keytype = 'public'
        text_output_dec, time_needed = timed_function_beautified(RSALibrary.decrypt_text_v2, n, e,
input_text_dec)
    else:
        text_output_dec, time_needed = timed_function_beautified(RSALibrary.decrypt_text_v2, n, d,
input_text_dec)

    context = {
        'n': n,
        'e': e,
        'd': d,
        'p': p,
        'q': q,
        'input_text_dec': input_text_dec,
        'text_output_dec': text_output_dec,

```

```

        'time_needed': time_needed,
        'keytype': keytype
    }

    return render_template('crypto.html', **context)

@app.route('/chart')
def chart():
    return render_template('chart.html')

if __name__ == "__main__":
    port = 5000
    if len(sys.argv) > 1:
        port = sys.argv[1]
        if port.isnumeric():
            port = int(port)

    app.run(debug=True, port=port) # listen on localhost ONLY
    # app.run(debug=True, host='0.0.0.0') # listen on all public IPs

```

CHAPTER-8

SCREENSHOTS

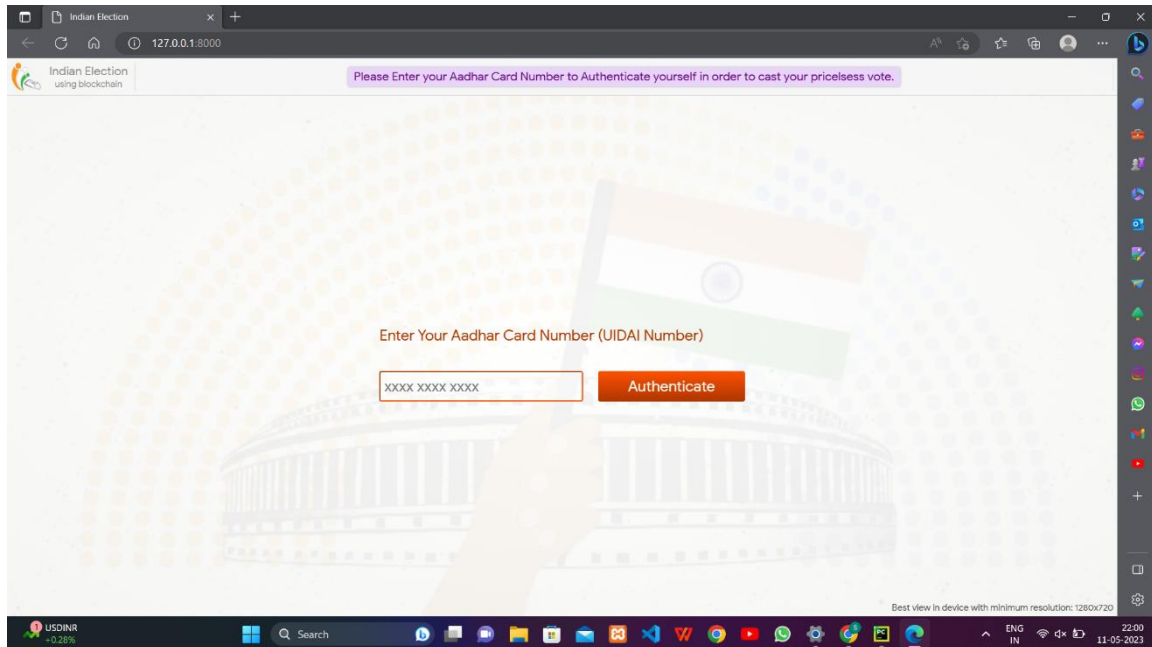


Figure 8.1:Authentication Page

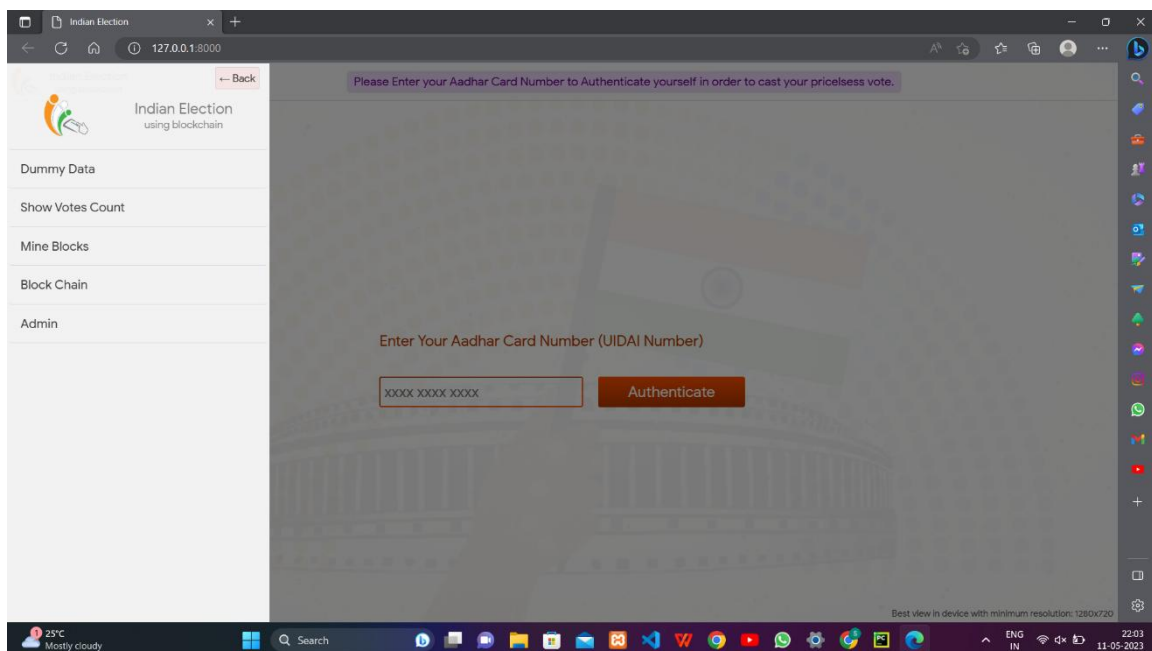


Figure 8.2:Home Dashboard

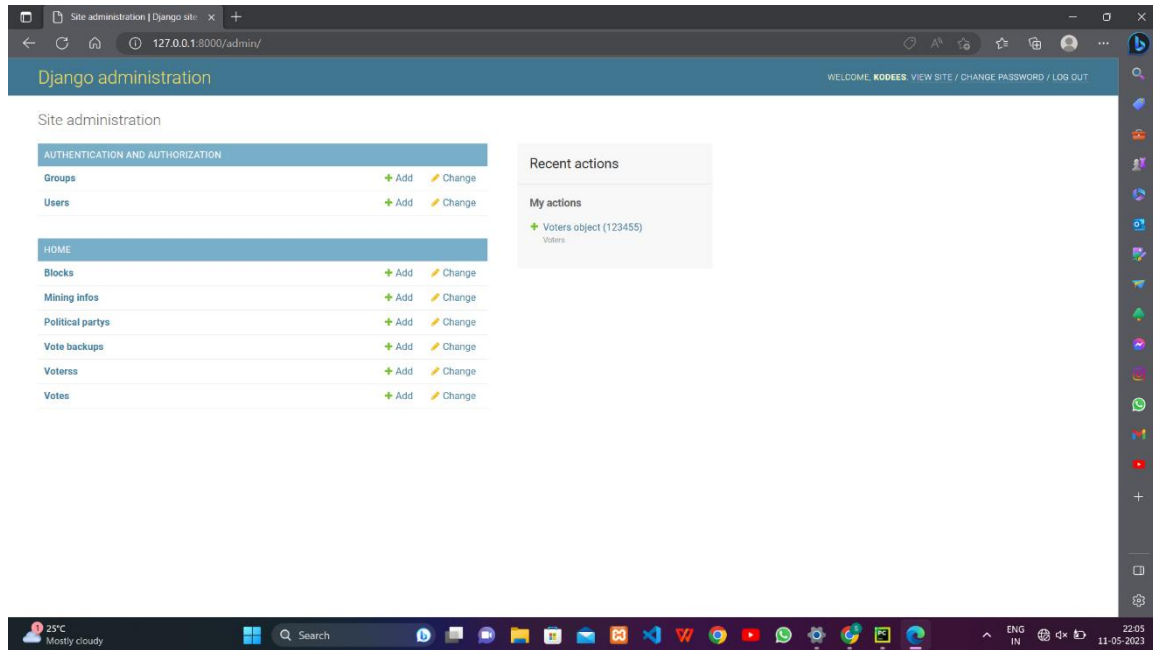


Figure 8.3:Admin Page

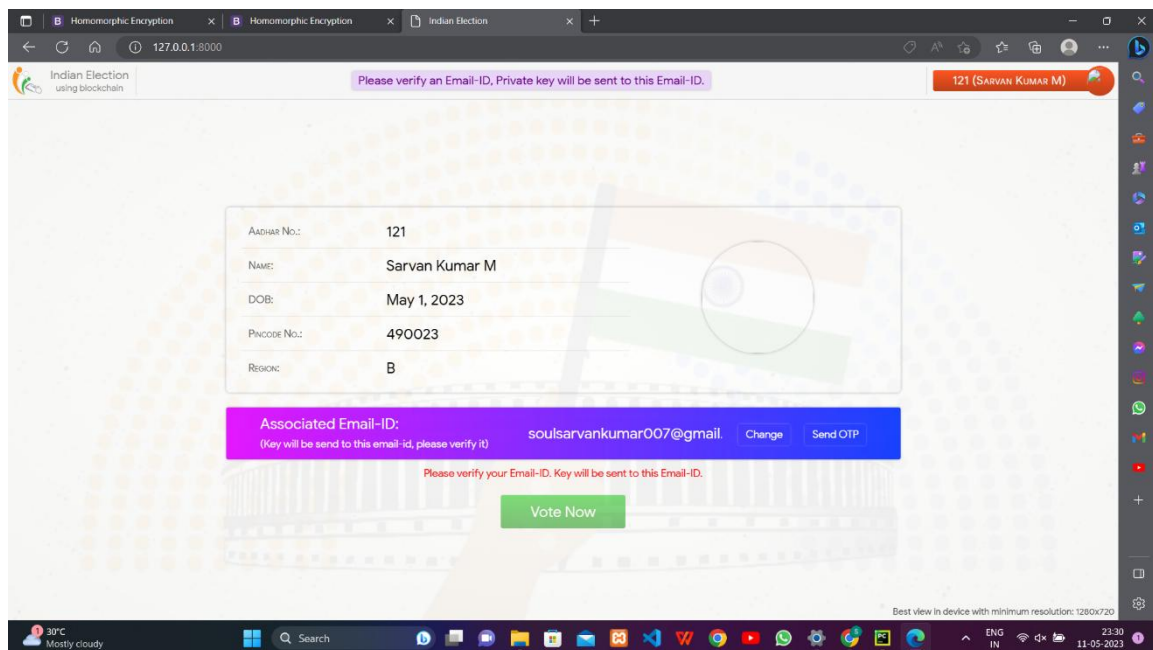


Figure 8.4:Canditate Profile

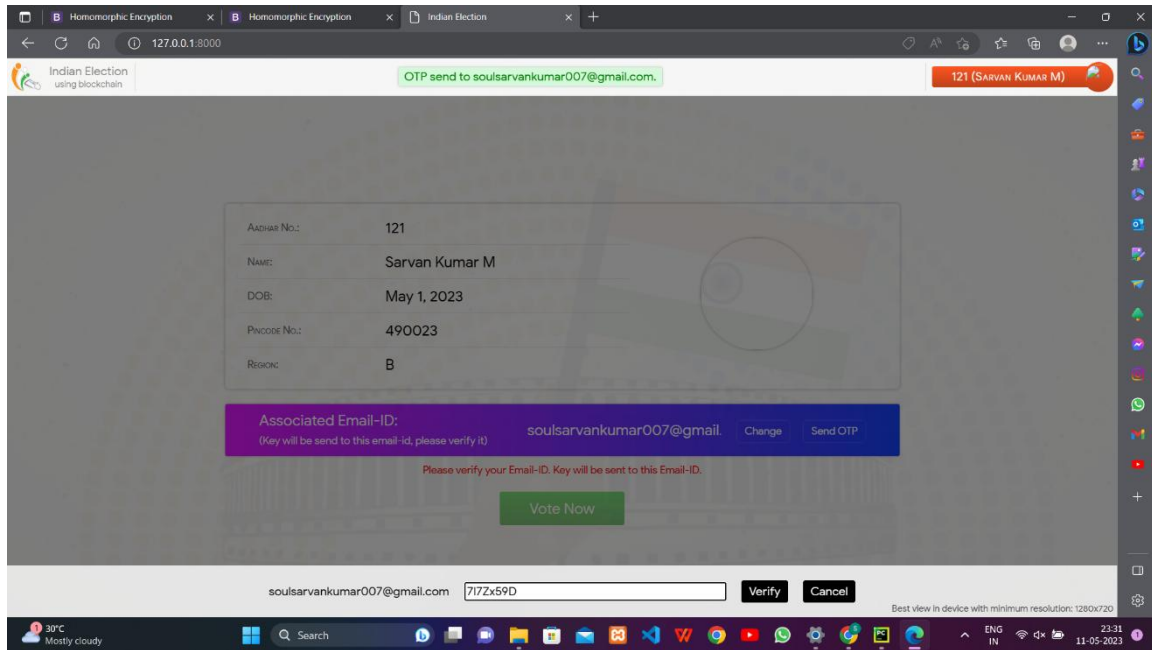


Figure 8.5:Mail Verification

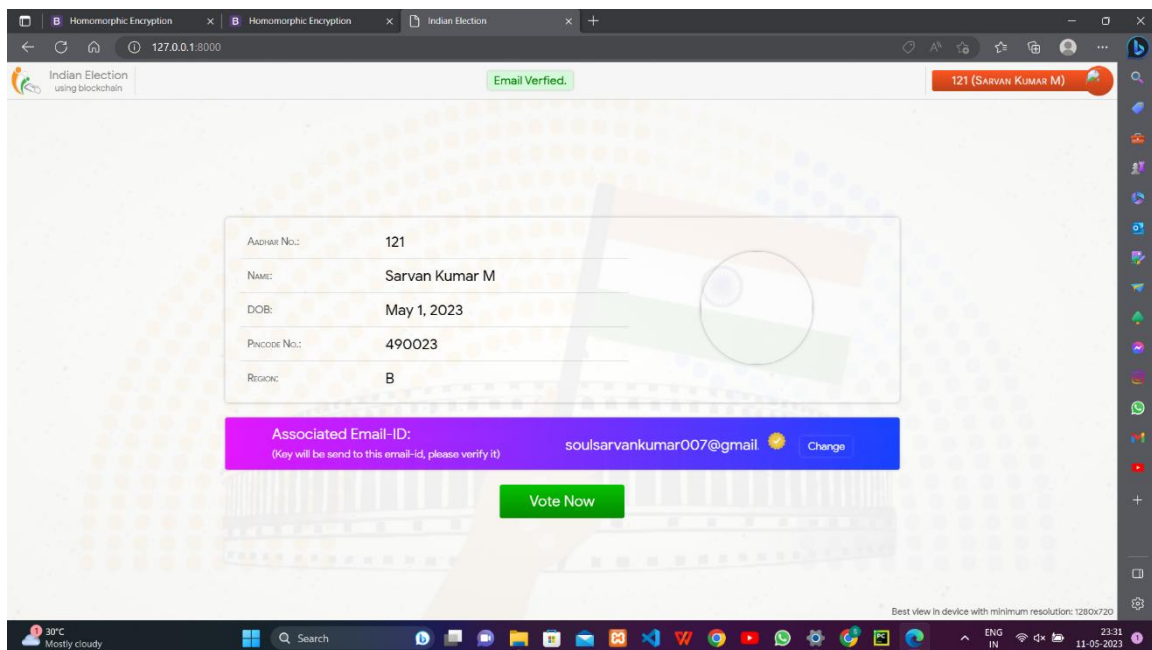


Figure 8.6:Mail Verification Succeed

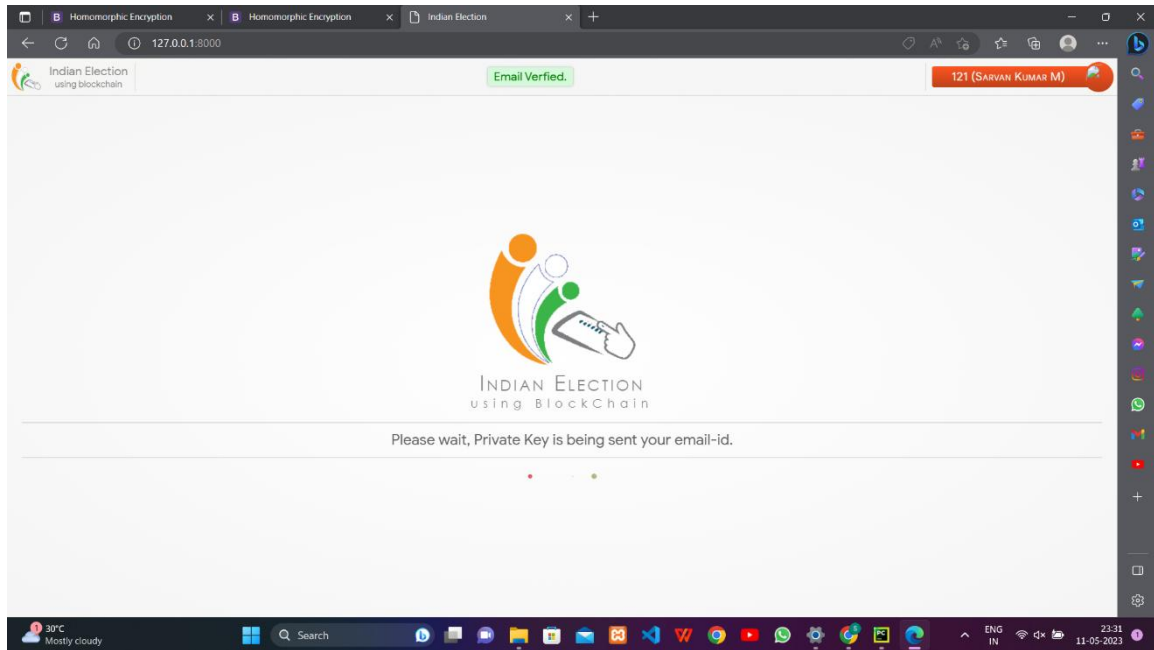


Figure 8.7:Private Key Generation

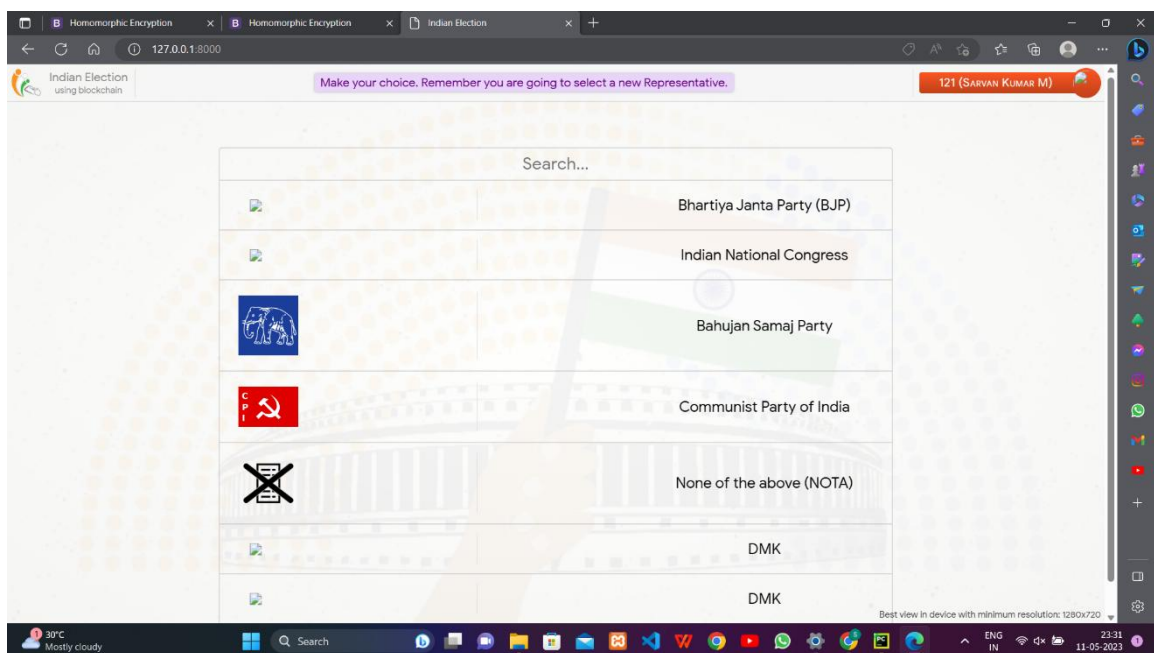


Figure 8.8:List of Parties

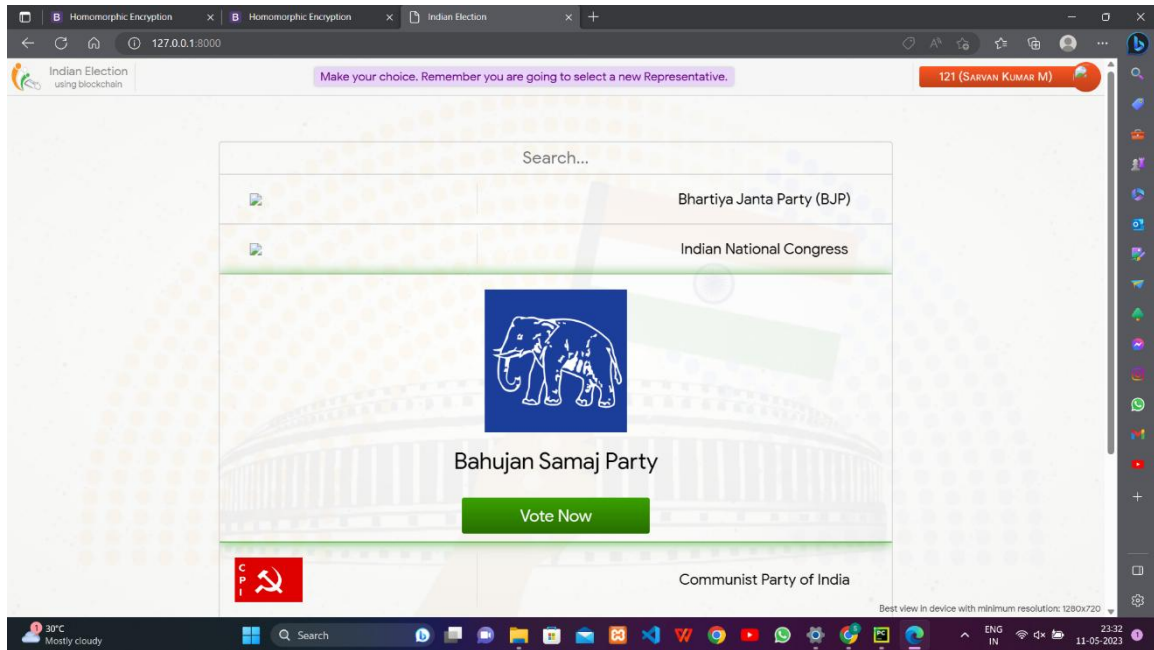


Figure 8.9: Voting Page

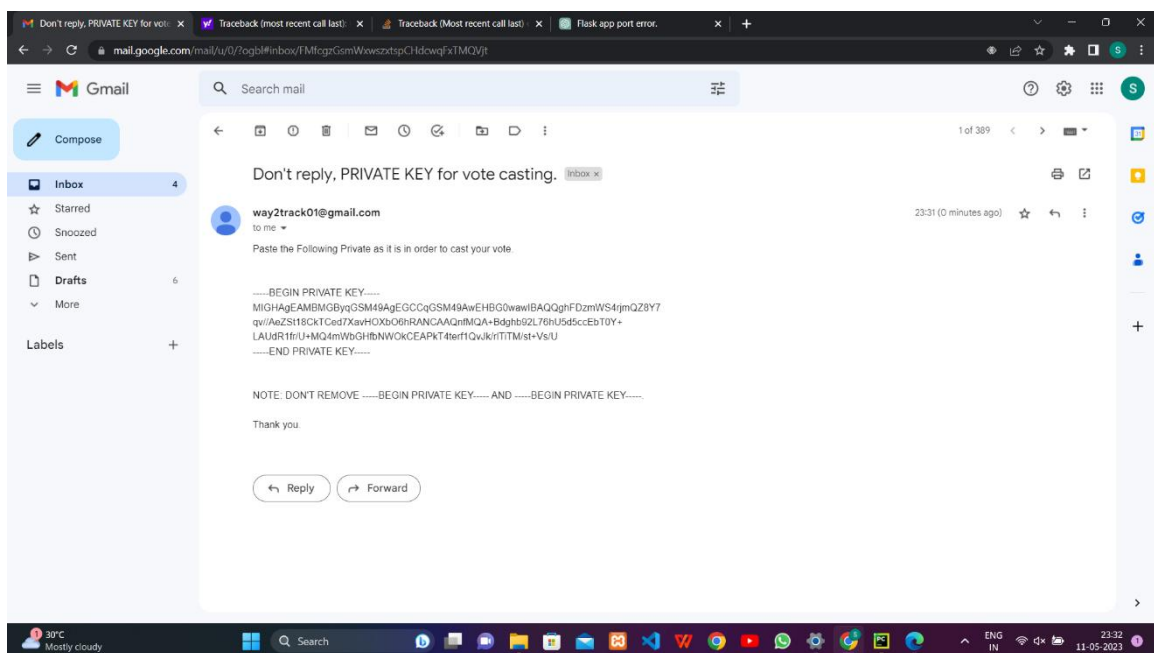


Figure 8.10: Generated Private Key

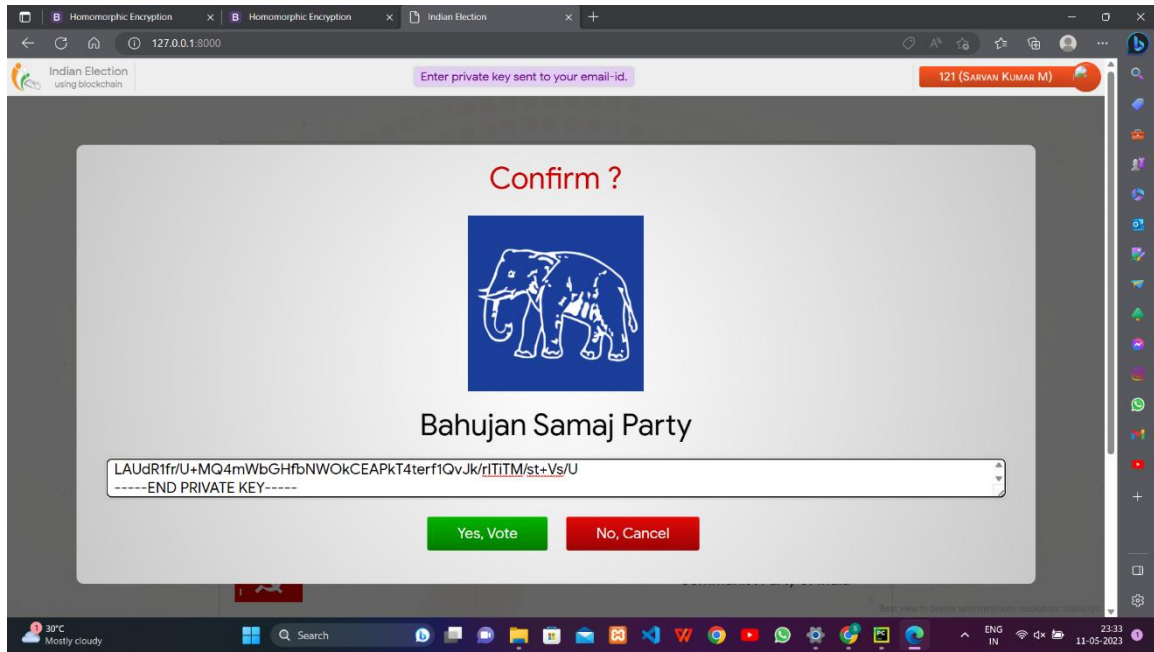


Figure 8.11: Voting Progress

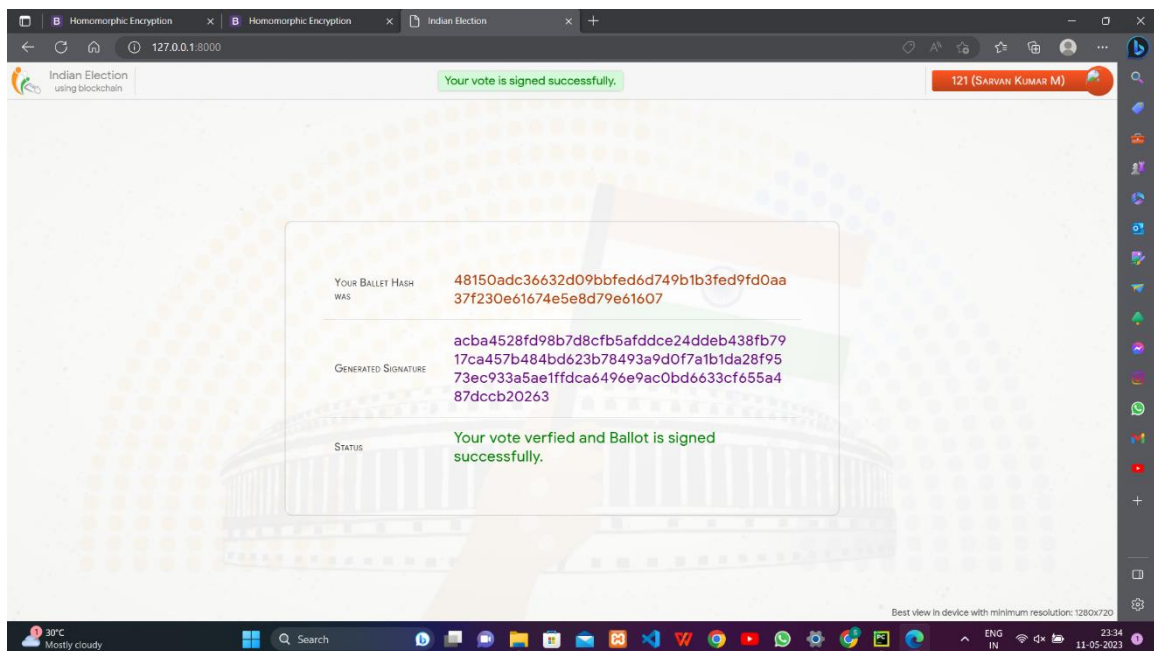


Figure 8.12: Voting Succeed

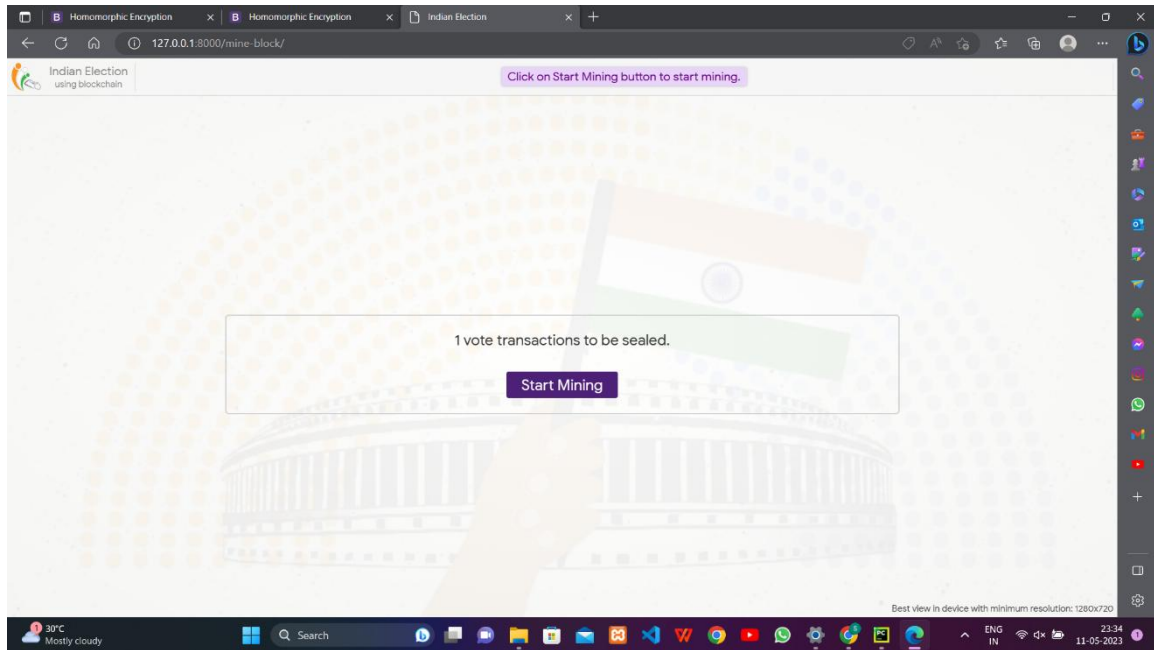


Figure 8.13:Hash code Start Mining

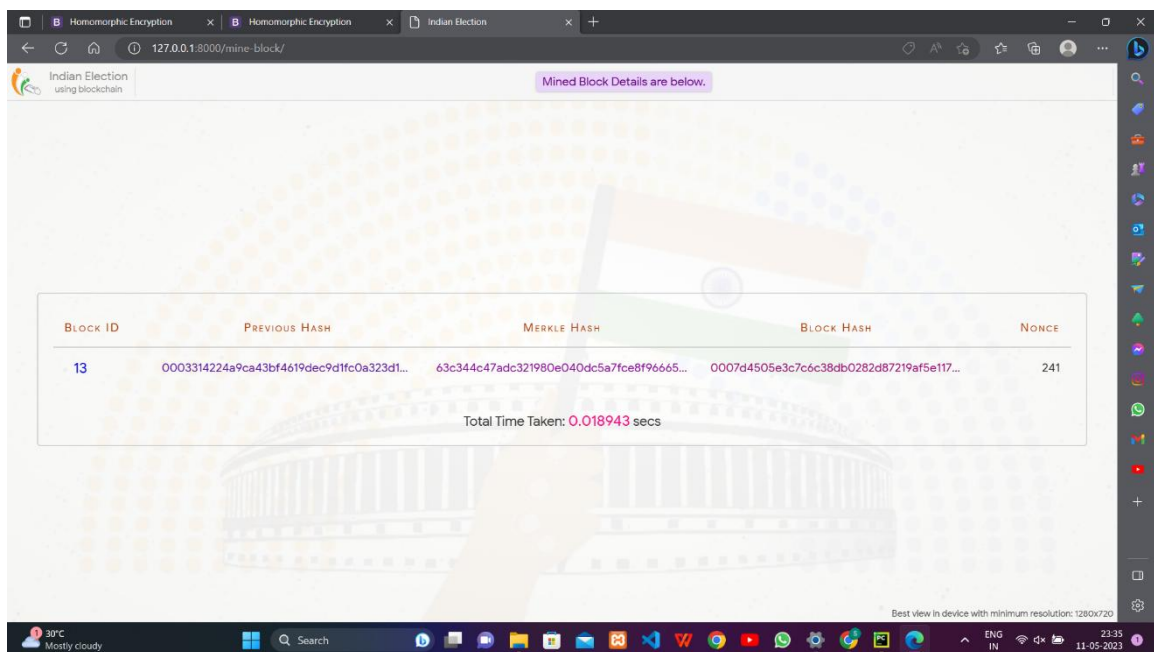


Figure 8.14:Hash code Generated

Indian Election using blockchain

All mined blocks are below. Click on block id to get information about that block.








BLOCK ID	PREVIOUS HASH	MERKLE HASH	BLOCK HASH	NONCE	TIMESTAMP	Verify
1	000000000000000000000000...	1d5ea93982c3a003c67d757d4c2d...	000b80bce929028505dc69277cb...	3543	2022-02-28 10:51:27	Verify
2	000b80bce929028505dc69277cb...	5bdb6852de8be039defb7b741b9c...	00015b89db29675b68fbb0950c9...	5540	2022-02-28 10:51:27	Verify
3	00015b89db29675b68fbb0950c9...	a8f93d51cce8e013012dbfa16139d8...	000b2772aa6623a136354a5e440b...	5019	2022-02-28 10:51:27	Verify
4	000b2772aa6623a136354a5e440b...	dee12e082d54dbf409de614779926...	000bd7da6e9170d9c9f529a0988d...	93	2022-02-28 10:51:27	Verify
5	000bd7da6e9170d9c9f529a0988d...	7f51a41ee43e7d3c3cfffbaeb37d...	000e9da61b879320696786a0a123...	4174	2022-02-28 10:51:27	Verify
6	000e9da61b879320696786a0a123...	c32365492e216a4b71f1240b5e829...	000763d72a62e7e829bbf5849855...	10279	2022-02-28 10:51:27	Verify
7	000763d72a62e7e829bbf5849855...	e095c1e96995d0fc0ab5546ebda1...	00002268b3971be8e9ee9935a986...	305	2022-02-28 10:51:27	Verify
8	00002268b3971be8e9ee9935a986...	13e599e12143ff75fb8274c86a4c07...	0009f1e9c9da197459747ffd71d763...	309	2022-02-28 10:51:27	Verify
13	000314224a9ca43bf4619dec9d1f...	63c344c47adc321980e040dc5a7f...	0007d4505e3c7c6c38db0282d87...	241	2023-05-11 18:05:47	VERIFIED

Best view in device with minimum resolution: 1280x720

Figure 8.15: Encrypted Vote Verification

Indian Election using blockchain

Vote count.

Sr. No.	PARTY SYMBOL	PARTY NAME	VOTE COUNT
1		Bahujan Samaj Party	4
2		Communist Party of India	3
3		None of the above (NOTA)	2
4		Bhartiya Janta Party (BJP)	2
5		Indian National Congress	1
6		DMK	0
7		DMK	0

Best view in device with minimum resolution: 1280x720

Figure 8.16: Vote Counts

CHAPTER-9

CONCLUSION

This chapter will discuss the development of the entire system as a whole. It will give an insight into the general procedures that were taken to accomplish the project. It will also discuss the aims and objectives of the initial proposal that were and the objectives that could not be accomplished. It will cover the drawbacks the project possesses and the necessary work that can be used to enhance the system in the future.

The main project objective was to build a secure online voting system, which would be used. The aim of the project was to convert the current use of paper-based voting to an electronic form of voting, which would enable voters to vote remotely from any location through the use of the internet.

Research was carried out on the different forms of online voting systems that currently exist, noting their features, and how to influence the participation of voters to an election. Various forms of server-side technologies were investigated in order to choose the right programming language to use for the development of the online voting system.

Security issues that may affect the integrity of the online voting system were addressed and counter measures on how to protect the system's security were researched. A number of software development methodologies were reviewed, upon careful consideration, the waterfall methodology was chosen as the most appropriate development method to use for this particular project.

During the design and development of the system, the main effort was focused on designing and developing the system to achieve a solution based on the concepts of the system proposal. This phase provided a clear description of how the system was to be created. The main emphasis was on creating an intuitive user interface for retrieving information, querying the

database by the use of Python classes and scrip lets and ensuring security was of top priority.

The testing phase of the project was used to rigorously exercise the system to expose any deficiencies and short comings which the system may have possessed. The results of the test showed if they system was ready to be delivered to its end users.

The system created met its objectives, by being simple to use and secure, which was important due to the fact that it would be used for the student union electoral process.

9.1. FUTURE WORK

A lot of work could be done to enhance the security features of the present system. The system at the moment permits voters to choose their own passwords when they are being registered by the administrator, it would be more secure to develop a password generator facility which would create a unique password for each voter at random. The passwords would then be emailed to each voter through the use of the email address provided.

The system could also be enhanced by displaying the voting result through the use of 3D graphs, which would help the administrator and elections analysts in reviewing the voting results. Due to time constraints these ideas where not brought to light in the project.

REFERENCES

1. Jayson Falkner, Ben Galbraith, Romin Irani, Casey Kochmer, Sathya Narayana Panduranga, Krishnaraj Perrumal, John Timney, Meeraj Moidoo Kunnumpurath, (2001), Beginning JSP Web Development, Wrox.
2. Peter denHaan, Lance Lavandowska, Sathya Narayana Panduranga, Krishnaraj Perrumal, (2004), Beginning JSP 2 From Novice to Professional, Apress.
3. Aneesha Bakharia, (2001), PythonServerPages, Prima Tech.
4. Bruce W. Perry, (2004), Python Servlet & JSP Cookbook, O'Reilly
5. Simson Garfinkel, Gene Spafford, (1997), Web Security & Commerce, O'Reilly.
6. Time Stamp.
URL: http://whatis.techtarget.com/definition/0,,sid9_gci817089,00.html
7. Maydene Fisher, Jon Ellis, Jonathan Bruce (2003), JDBC API Tutorial and Reference . Third Edition, Sun Microsystems.
8. George Reese, (2000), Database Programming with JDBC and Python. O'Reilly.
9. Laura A. Chappell, Ed Tittel (2004), Guide to TCP/IP. Thomson.
10. Transmission Control Protocol
URL: http://en.wikipedia.org/wiki/Transmission_Control_Protocol
11. Andrew S. Tanenbaum, (1996), Computer Networks. Third Edition. Prentice Hall
12. Mark Andrews: Story of a Servlet
URL: <http://Python.sun.com/products/servlet/articles/tutorial/>
13. Cisco Systems Inc. (2002): Introduction to TCP/IP
URL: <http://www.cisco.com/univercd/cc/td/doc/product/iaabu/centri4/user/scf4ap1.htm#xtocid6>
14. Introduction What is JDBC
URL: <http://Python.sun.com/docs/books/jdbc/intro.html>
15. Bruce Schneier: Analysis of SSL 3.0 Protocol URL:
<http://www.schneier.com/paper-ssl.pdf>
16. SSL Protocol
URL: <http://www.cryptoheaven.com/Security/Presentation/SSL-protocol.htm>

17. Allen (2004): Access vs MySQL
URL:<http://codewalkers.com/tutorialpdfs/tutorial79.pdf>
18. Simon Bennett, Steve McRobb, Ray Farmer, (1996), Object-Oriented System Analysis and Design Using UML
19. William R.Cheswick , Steven M. Bellovin, Aviel D. Rubin, (2003), Firewalls and Internet Security. 2nd Edition, Addison-Wesley.
20. Leszeka A. Maciaszek, Bruc Lee Liong, (2005), Practical Software Engineering. Addison Wesley.