Ashesi University

DEVELOPMENT OF A BIOMETRIC AUTHENTICATION VOTING SYSTEM FOR SENIOR HIGH SCHOOLS IN GHANA

THESIS

B.Sc. Computer Engineering

Emmanuel Yaw Manu Annor

2020

ASHESI UNIVERSITY

DEVELOPMENT OF A BIOMETRIC AUTHENTICATION VOTING SYSTEM FOR SENIOR HIGH SCHOOLS IN GHANA

CAPSTONE PROJECT

Capstone Project submitted to the Department of Engineering, Ashesi University in partial fulfilment of the requirements for the award of Bachelor Of Science degree in Computer Engineering.

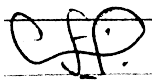
Emmanuel Yaw Manu Annor

2020

DECLARATION

I hereby declare that this capstone is the result of my own original work and that no part of it has been presented for another degree in this university or elsewhere.

Candidate's Signature:



……

Candidate's Name:

Emmanuel Yaw Manu Annor

……

Date:

May 29, 2020

……

I hereby declare that the preparation and presentation of this capstone were supervised in accordance with the guidelines on supervision of capstone laid down by Ashesi University.

Supervisor's Signature:

……

Supervisor's Name:

……

Date:

……

Acknowledgement

Sincere gratitude to all the people who advised and encouraged me to undertake this project. I could not have achieved this feat without your guidance and support.

I am grateful to my father Professor Serekye Yaw Annor and mother Victoria Adu-Owusuah Annor, for their love, prayers, care, and commitment shown to me throughout my undergraduate journey. Without your tremendous emotional support and investment, this journey would not have been a success.

To my able supervisor, Mr. David Amatey Sampah, God bless you a lot for your sacrifices and sleepless nights to review my works. Your guidance and suggestions through the design and implementation process have helped me to build more confidence as a Computer Engineering major to move further and work on several other projects.

My gratitude to Dr. Nathan Amanqauh, Acting Dean of Ashesi Engineering Department, for his further guidance, suggestions, and comments about this project. His immense contributions to this project, and others have impacted my life in my four-year study at Ashesi University.

Abstract

Voting is practised in almost every Senior High School in Ghana to elect school leaders in various schools. In Ghana, several issues are facing senior high schools during voting processes. Some of these issues include: The voting process being costly, taking long hours to finish the voting process, Issues of election fraud (election manipulation or vote rigging), and generally the voting process being tedious, not secured, and trusted. It is required for every voting process to be transparent and able to withstand a variety of fraudulent behaviours. This project presents the design of a highly secured, cost-effective voting system designed with Flask python web framework running on a raspberry pi 3B+ (sever). This system includes a fingerprint authentication as a form of security to prevent fraudulent behaviours. The results from the voting are displayed on an admin page, together with some other voting statistics.

Key Words: biometric authentication, Flask, web server, election fraud, cost-effective.

Table of Content

Contents

List of Tables

List of Figures

Chapter One

1.0 Introduction

1.1 Introduction and Background

Under the Constitution of the Republic of Ghana (Amendment) Act, 1996 Act 527, article 42; Every citizen of Ghana of eighteen years of age or above and of sound mind has the right to vote and is entitled to be registered as a voter for public elections and referenda [1]. According to [2], the Government of Ghana was created as a parliamentary democracy, followed by alternating military and civilian governments. In January 1993, the military government gave way to the Fourth Republic after presidential and parliamentary elections in late 1992. Since then, Ghana has been enjoying a peaceful and fair democracy. During these times, elections were conducted using the traditional system of voting (ballot paper voting). In 2012 the Electoral Commission of Ghana proposed and adopted a new biometric system of voting.

In Ghana, fingerprint verification was used to verify voters by the Electoral Commission before they cast their votes during the 2012 elections. In the face of advancing ICT trends and emerging challenges in manual voter registration, the Electoral Commission of Ghana took the bold decision to replace the current voters' register with a finger biometric register for subsequent elections [3]. Also, some major issues they were averting included [4], preventing multiple voting, preventing voter impersonation, and preventing ballot stuffing.

In every senior high school (SHS) in Ghana, new student leadership is selected at the beginning of a new academic year to forward the interests of students to the schools' administration. When a government is elected, they must plan and organize all activities in the school. Since Ghana is a democratic country, most schools choose student leaders through a voting process. Every student in the school is encouraged to be patriotic and participate in the school's general

elections to elect a new student leadership because every voice matter and needs to be heard during this process.

1.2 Problem identification

An election system must be sufficiently robust to withstand a variety of fraudulent behaviours and must be sufficiently transparent and comprehensible that voters and candidates can accept the result of the election [5]. The researcher conducted ethnographic research by visiting three different senior high schools in Kumasi, Ashanti Region of Ghana, on January 7, 2020. The following are some problems stated by students and staffs of various schools after the research was conducted:

1. The voting process is costly: the cost of voting manually mostly includes money for buying and printing ballot papers and funds to provide incentives to staff to help the election process to be a success.
2. Taking longer hours to finish the voting process hence sometimes classes needs to be cancelled throughout the day.
3. Most students do not trust the voting process because of issues of human errors (counting mistakes, etc.).
4. Issues of election fraud (election manipulation or vote rigging).
5. The process is generally tedious, not secured, and trusted.

According to the Head of Science Education and ICT Unit under the Secondary Education Division of the Ghana Education Service [6], there is a need to develop, implement and popularize science and technology through various activities in Ghana. All schools in Ghana are greatly encouraged to teach and practicalize Information Technology (IT) education to their students because of its wide range of advantages. In this light, it is only beneficial an efficient

system is proposed and implemented in various senior high schools to make voting safe, fair, and transparent.

1.3 Objectives

The aims of carrying out this project are:

1. Designing a hardware implementation to help students in senior high schools vote with ease.
2. Exploring the possibilities of a highly secured voting system to make elections safe, fair, and transparent.

1.4 Motivation and Justification

It is about time students in Ghanaian based colleges took the responsibility of identifying problems around them and solving them. This would give students and graduands the experience to compete in the world of work since they would already have skills of creating and implementing successful projects; in the long term, this will create a positive impact in Ghanaian tertiary institutions.

Also, due to the evolution of technology in the world, it is only feasible to excite teenagers (mainly senior high school students) with such projects to spur their interest in information technology. The use of ICT in the learning environment can bring about a rapid change in the student ‘s performances [7]. Most students will be intrigued and challenged to take their studies seriously so they can excel and build such projects on their own [7]. In the long run, this project can have a positive impact on the nation and urge more of the youths to focus on the technology industry.

1.5 Expected Outcome

This project's implementation is to: design a webpage connected to a fingerprint sensor(scanner), which would be connected to a microcontroller (Raspberry pi 3 B+) to help students vote in their schools. This implementation will allow students to be able to login with their credentials (Student name, Student password, and fingerprint ID) before voting. A system like this must be highly secured. Therefore, considerations such as a multi-factored authentication (password and fingerprint) and other forms of security procedures will be factored into the development. Also, to make sure the election is safe and fair, every student can only vote once since they all have unique fingerprints, which would be checked in a database. The process must be as efficient and fast as possible.

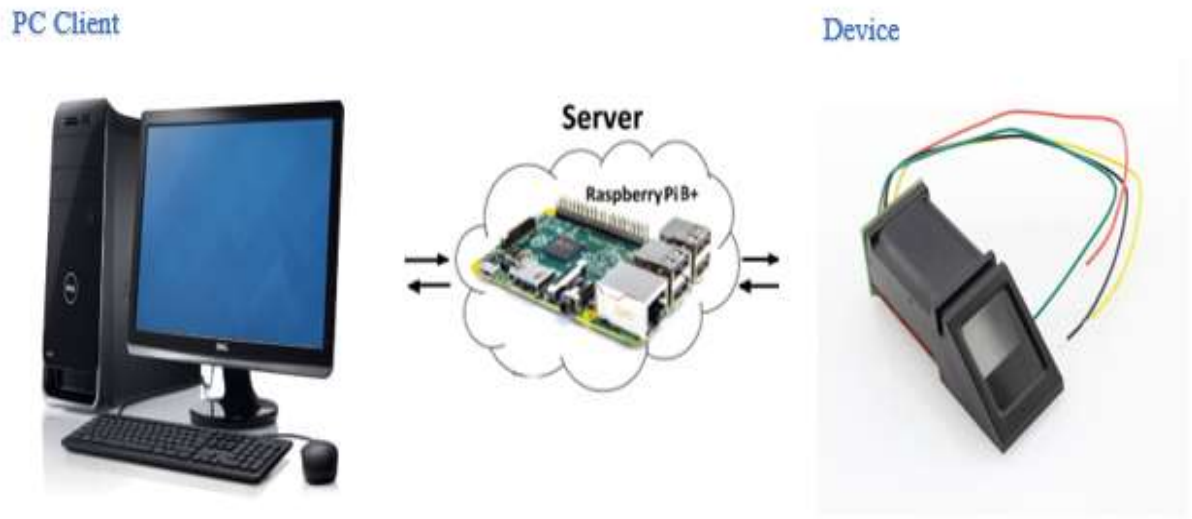


Figure 1: System Architecture

This systems' architecture follows the client-server archetype; that is, a client (in this case execution from the raspberry pi's browser) will make a request to the server; hence the server will reply to the client. The server in the raspberry pi is responsible for managing the system (including the fingerprint sensor attached to the pi).

Chapter Two

2.0 Literature Review

2.1 Available Solutions

Some senior high schools in Ghana have already started the implementation of electronic voting in their schools. Such schools include Christ the King International High School, Mawuko Girls SHS, and Nyankumasi Ahenkro S. H. S. and others. These schools use a somewhat similar system: a combination of basic HTML (HyperText Markup Language) standard markup language for creating web pages and PHP (Hypertext Pre-processor), a programming language for web developers. Students select various aspiring candidates on the webpage, which calculates the results of the election after the election process ends. Various reasons for the implementation of these systems include:

1. [8] Taking out the cumbersome process of sorting out and counting ballots after voting.
2. [9] Making students aware of the various application in the Information Communication Technology space and preparing them for electronic voting systems, which would soon dominate primary elections across nations.
3. [10] For students to be able to vote for preferred candidates in less than a minute.

Figure 2: A student of Christ the King International High School using an electronic voting system

The biometric authentication factor added to the proposed method in this paper will make it more secure than the above-discussed systems because it uses multi-factor authentication and a fingerprint scanner, which will improve the security and authentication of the system. This form of security (biometric) is used across significant elections in various nations.

2.2 Reviewed Articles

1. [11] Electronic voting has been successfully implemented in countries like Australia, Belgium, Estonia, France, and many others. Since the introduction of this voting system, the various countries have benefited, and the system has helped solve most problems associated with manual voting. The benefits Australia enjoys by implementing electronic voting are; provision of a secret ballot for blind and low-vision

voters, more natural delivery of remote voting services, and secure ballot-handling. From this inspiration, a well-thought-through implementation of an electronic voting system can solve the problem identified in senior high schools in Ghana.

2. A Journal by Himanshu Vinod Purandare, Akash Ramswaroop Saini and others on a proposed system: Online Voting System that Utilizes a Facial Recognition Resource

   [12] This voting system uses a facial recognition resource to secure the system by allowing only authentic voters to have access to it. Also, it uses a One Time Password (OTP) to provide extra security to the system. A database was implemented using XAMPP server; hence voters' details can be retrieved whenever needed. This application can run on all android phones, but most android phones do not use the facial recognition software; hence such users cannot use this application.

3. A Journal by Jagdish B. Chakole and P. R. Pardhi on a proposed system: Internet Voting System Designed for Corporate Elections

   [13] To keep the system very secured (providing security to casted votes). This is done by preventing active (tampering casted vote) and passive intruders (access cast votes and make changes) from having access to it. To achieve this, the concept of cryptography and digital signatures were used. Hence votes will be encrypted during voting and decrypted for votes calculations.

4. A Journal by Vinayak Bharadi and Dhvani Shah on a proposed system: IoT Based Biometrics Implementation on Raspberry Pi

   [14] This implementation introduces a secured technology based on information security, shedding more light on how biometrics can leverage cloud's vast computational resources and striking properties of flexibility, scalability, and cost

reduction to reduce the cost of the biometrics system requirements of different computational resources (i.e., processing power or data storage) and to enhance the performance of biometrics systems' processes (i.e., biometric matching). This is done using a Raspberry Pi to build a low-cost biometric system.

Chapter Three

3.0 Requirements and Methodology

3.1 Thesis Design Objective

i. To design a hardware implementation that will help students in senior high schools vote with ease.

ii. To explore the possibilities of a highly secured voting system that will make elections safe, fair, and transparent.

iii. To leverage the cloud's vast computational resources (storage, flexibility, cost efficiency), which would reduce the cost of biometric system requirements.

3.2 Design Decisions

[15] To make the best decision and choose between the alternatives available that will contribute to the development of the voting system, a Pugh matrix and insights from the literature were considered. A Pugh matrix is a decision-making model designed to choose and compare available alternatives using its criteria's:

Table 1: Key for Pugh Matrix

Key	Meaning
S	Same or Neutral
+	Better than
-	Worse than
0	Baseline

Table 2: Pugh Matrix for Microcontrollers

		Baseline	Weight	Node MCU	Atmega8a	FRDM-kl25z
Criteria		Raspberry pi 3B+				
Cost		0	2	+1	+1	+1
Power Consumption		0	4	+1	+1	+1
Processing Power		0	4	-1	-1	-1
Memory segments		0	3	-1	-1	-1
Total Plus (p)			14	6	6	6
Total Minus (m)			0	7	7	7
Total Neutral			0	0	0	0
Total (p + m)			14	-1	-1	-1

From the table above (Table 2), the Raspberry pi 3B+ is the best overall alternative in terms of cost, relatively higher processing power, power consumption, and memory segments.

3.3 System Requirements

3.3.1 Functional Requirements

The functional requirements of the system state precisely what the system will do; this includes:

i. System signup must work successfully.

ii. The system login must work successfully.

iii. The system should be able to send user data into the database

iv. The system must detect the fingerprints of users.

v. The system should be able to calculate the total vote.

3.3.2 Minimum Requirement Specification

The minimum requirements specification needed for the system to function correctly and efficiently includes:

i. Raspberry pi 3B+
ii. Memory card for raspberry pi 3 B+ 2GB
iii. Web browser
iv. 5" 800x480 TFT HDMI display or LCD Screen
v. Fingerprint scanner

3.3.3 Non-functional Requirements

3.3.3.1 Software Quality Attributes

To describe various criteria that judge the final requirements

i. Performance: The system should be user-friendly so, it accomplishes its various tasks effectively; signing up, logging in, voting for a candidate, displaying results after voting, and logging out.
ii. Speed: Publishing data (results) from the database to the system should be fast.
iii. Accuracy: The system will compute its calculations correctly and act accordingly.
iv. Timing: The system should be responsive 24 hours a day during and after the voting process. Except it is undergoing maintenance and upgrades.
v. Security: The system should be protected from unauthorized users. Also, the system should allow its users to vote **once**.

3.3.3.2 Safety Requirements

The following will be considered to provide the best level of safety in the system:

i. The system includes a biometric authentication to provide security to its users before permitting login and vote.

ii. The system should hash voter's information when the vote is cast to prevent votes being traced.

iii. The system should provide a means to log out of the system.

3.3.4 Use Case Scenario

The following explains the interactions between an admin, users, and the voting system.

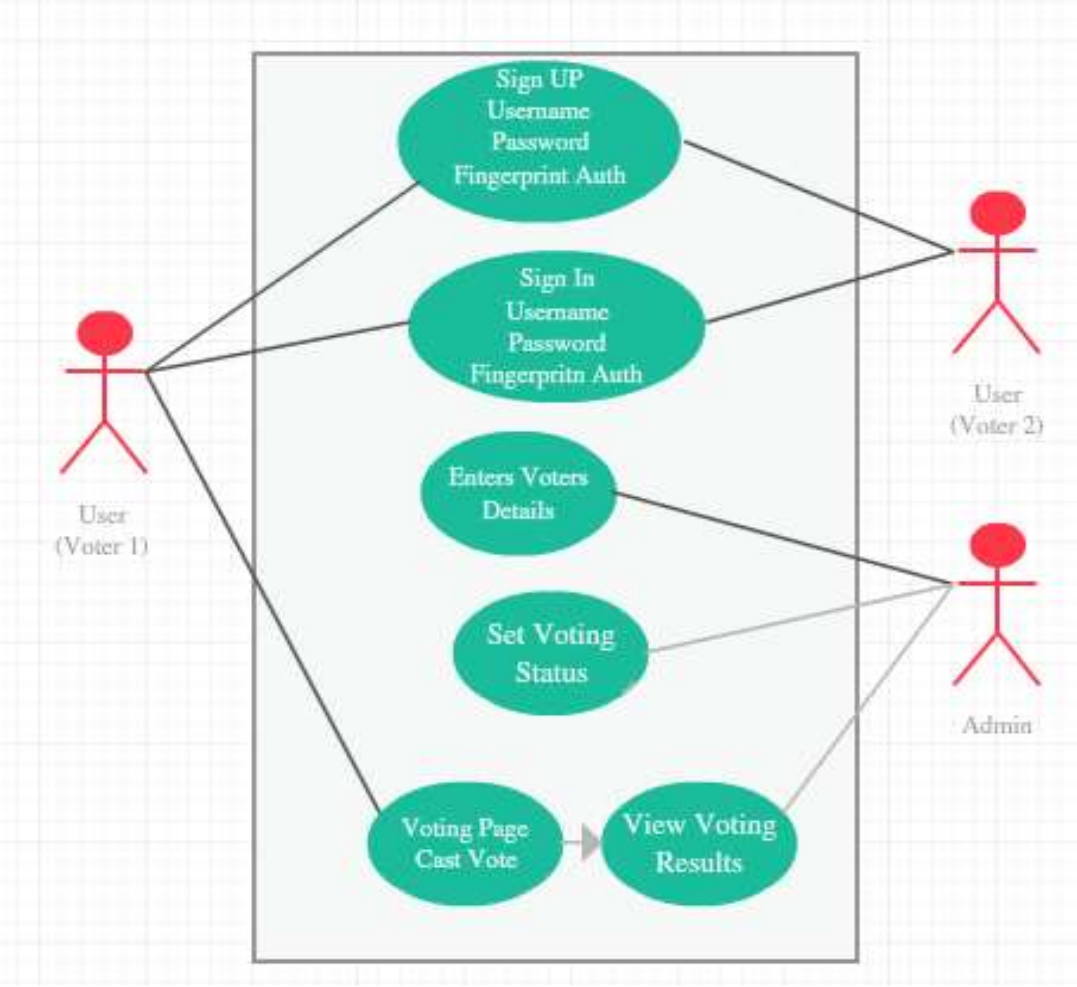


Figure 3: Use Case of the System

3.3.5 Flowchart

The flow chart of the system is illustrated using the diagram below

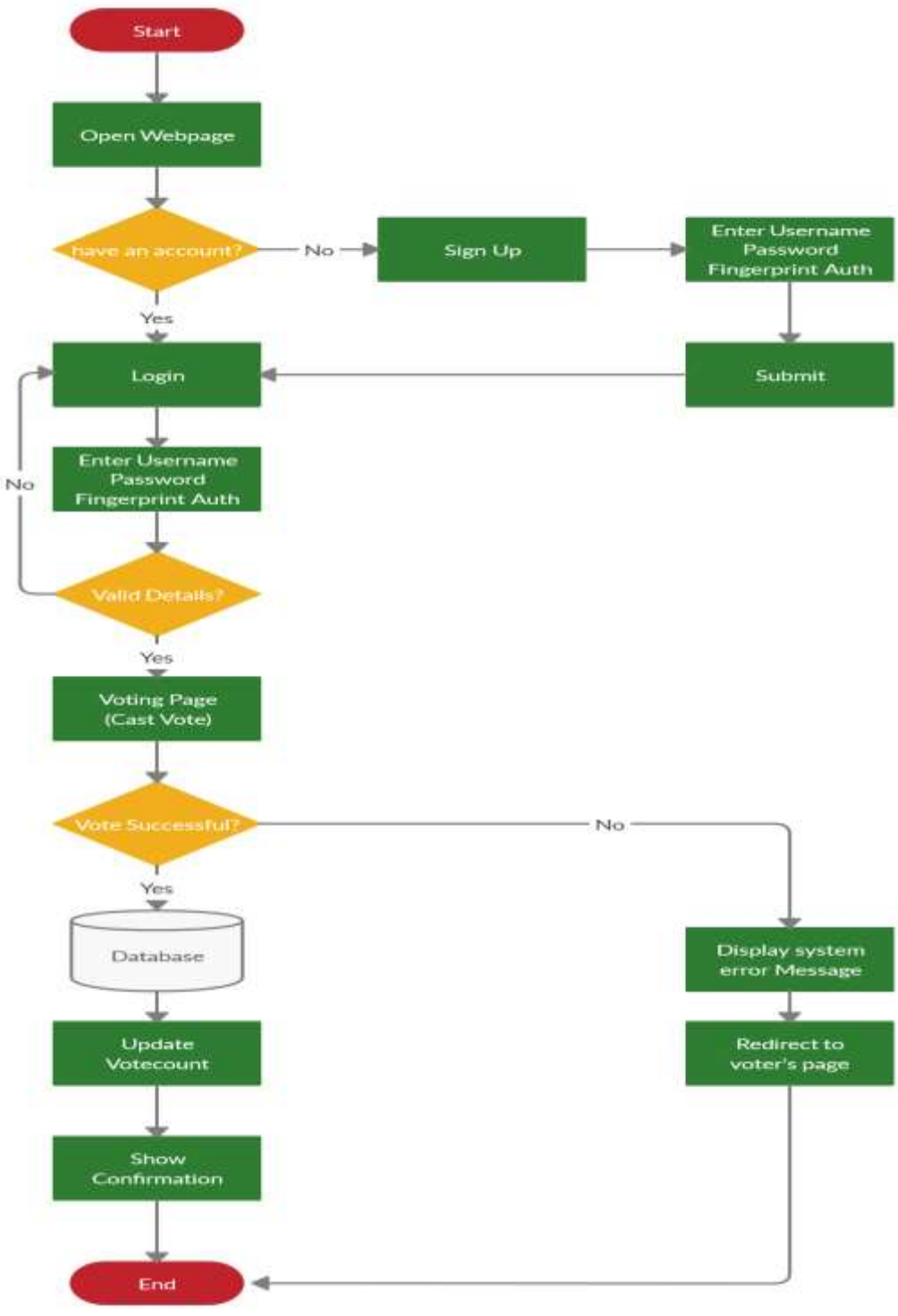


Figure 4: Flowchart of the System

3.4 Architectural Design

The architectural designs of the system explain the main implementation and relationships of both software and hardware in the system. The figure below shows the architectural design of the system:

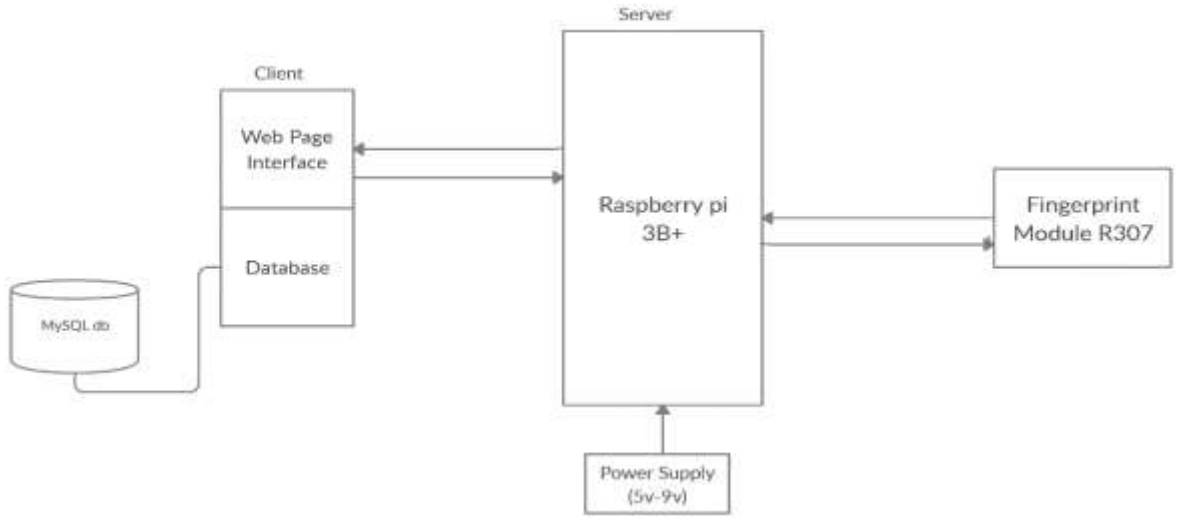


Figure 5: The architectural design of the system

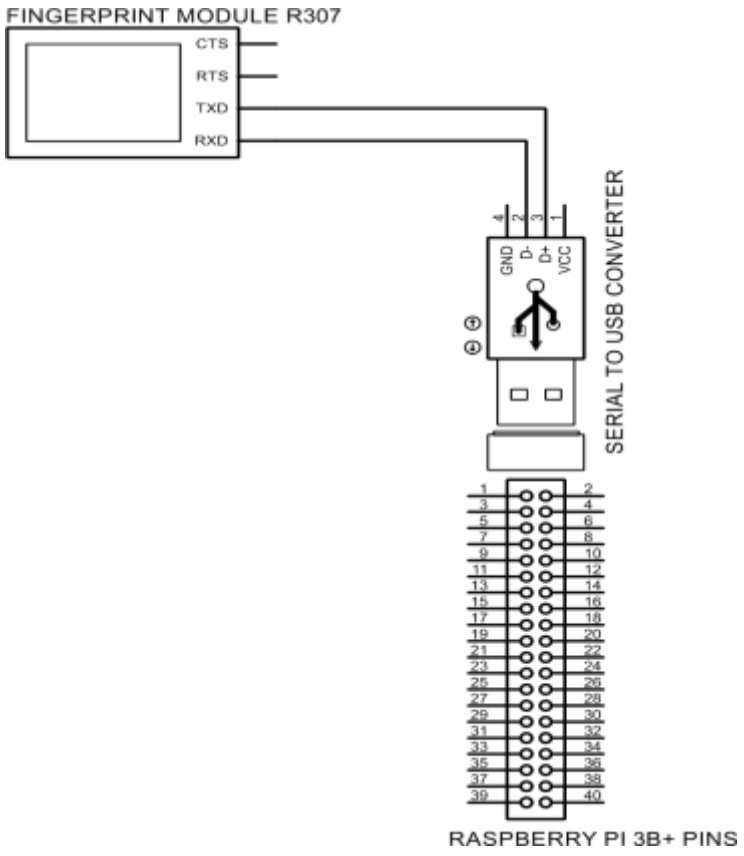


Figure 6: Schematic diagram for raspberry pi connection

3.4.1 Software and Hardware implementation

Figure 6 explains how the various components in the system will be connected. The fingerprint sensor R307 will be connected to the microcontroller (raspberry pi 3B+) via serial to USB converter (as shown in the schematic diagram in figure 4). With the raspberry pi acting as a server, users will interact (vote and view voting results) with the system via the webpage interface displaying on a 5" 800x480 TFT HDMI display screen or LCD Screen. The webpage is connected to MySQL database, which stores all user details and voting statistics. The entire system is powered by an output voltage between 5v-9v.

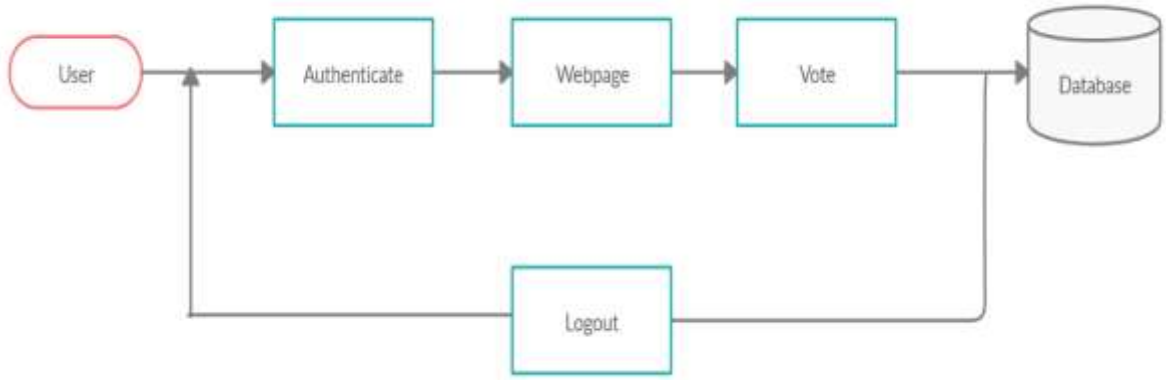


Figure 7: Block diagram of software implementation

3.4.1.1 Software unit implementation

This section is categorized into three subsections: building the python server using Flask, web development with HTML5, CSS, Bootstrap, jQuery, and MySQL database and web deployment with Heroku.

a) To create a simple server-side application that is easily compatible with fingerprint sensor reader R307 library, does not consume a lot of memory, can easily accommodate changes, has better testability, and better performance, Flask was chosen. Flask is a micro web framework server gateway interface written in python. To efficiently and securely build the server-side application, various Flask extensions including Flask-login to create a user session management for Flask, Flask-WTForms to provide an interactive user interface for the login and signup pages, Flask-SQLAlchemy to offer a secure interaction and integration with MySQL database using SQLAlchemy, Flask-Werkzeug security to provide security interface for user data, were used to form the skeleton of the web application. These different extensions and frameworks form wrappers around the Flask framework to facilitate the request and response process. Visual studio code-OSS was the preferred IDE used since there is no official release of visual studio code for ARM devices like the Raspberry pi.

b) [16] To design a responsive, interactive and user-friendly views, client-side frameworks, sheet languages and other open-source frameworks such as HTML5: the standard mark-up language for document rendering in browsers, Bootstrap: a stylesheet language framework used to control the appearance of HTML documents and jQuery: a JavaScript library used to handle events, and Ajax (gives access to asynchronous web applications.) was used.

c) The app will be hosted and deployed on Heroku, a platform as a service (PaaS) application that is compatible with python web framework and MySQL database. It has been in existence for a long time; hence it has a large community that's ready to offer help if needed.

The description of components in the system, how they relate to each other is further explained in the table below. Pictures of the hardware components can be found in appendix A.

Table 3: Hardware and Software components description

Device	Application
Raspberry pi 3B+	[17] A powerful microcontroller (small single-board computer) that has a 1.4GHz 64-bit quad-core processor and 1 GB RAM. It also has a LAN wireless and Bluetooth connectivity to create ideal solutions. In this situation, it is used to host a full website.
5" 800x480 TFT HDMI	[18] This is a mini LCD/HDMI display with a high picture resolution, fast response time, and full-colour display. It will serve as a display for the raspberry pi.

Fingerprint sensor R307	The fingerprint sensor would be used as a means of authentication for users before they can vote. It has an optical green backlight and is USB2.0/UART compatible.
Python Flask	[19] Flask is a micro web framework server gateway interface written in python that makes it easier to scale up to complex applications. To efficiently and securely build the server-side application on the raspberry pi, various Flask extensions be used.
MySQL database	A relational database system that would be used to store user data and voting statistics.
PL2303 USB TTL Cable FTDI	A serial to USB converter that connects the fingerprint sensor to the raspberry pi
Bootstrap	An open-source framework that makes it easier to design web interfaces
Heroku	Web hosting platform

Chapter Four

4.0 Implementation

4.1 Experimental Setup

In developing the voting system, the various parts of the system were modularized into different units, mainly the software and hardware unit. The software unit encompasses building the server, web development, and deployment. The hardware unit comprises the interconnection of all the physical hardware components used in the system. The two units precede a system testing phase.

4.1.1 Hardware unit development

The various components that were used in the development of this system to achieve different functionalities, include raspberry pi 3B+ microcontroller which will serve as the server for the web application, 5" 800x480 TFT HDMI display which will serve as the display for the microcontrollers' operating system (main user interface), Fingerprint Sensor Reader R307 to read the fingerprint of users, 16GB flash memory card that serves as a storage device for the microcontroller, a keyboard and mouse to serve as primary input devices for the raspberry pi microcontroller and a PL2303 USB TTL Cable FTDI which will serve as a gateway connecting the fingerprint sensor to the raspberry pi microcontroller. The following steps explain how all the devices were systematically connected (Hardware unit implementation):

1. Insert a 16GB flash memory with a Raspbian operating system installed into the Raspberry pi 3B+ memory slot.

Figure 8: Inserting an SD Card into the Raspberry pi

2. Connect one side of the PL2303 USB TTL Cable FTDI (Serial to USB converter) to the USB port of the Raspberry pi 3B+ and the other side to the Fingerprint Sensor Reader R307.

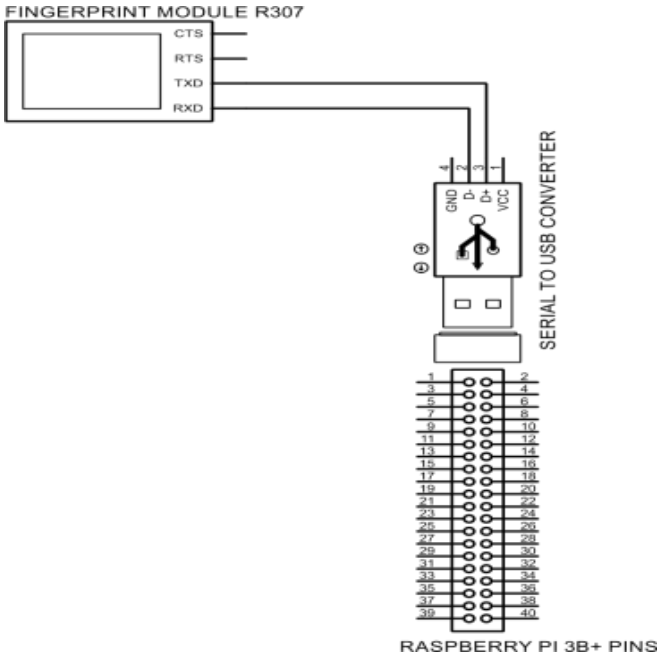


Figure 9: Connecting the fingerprint module to the raspberry pi via PL2303 USB TTL Cable FTDI

3. Connect the 5" 800x480 TFT HDMI display to the HDMI port of the Raspberry pi 3B+.

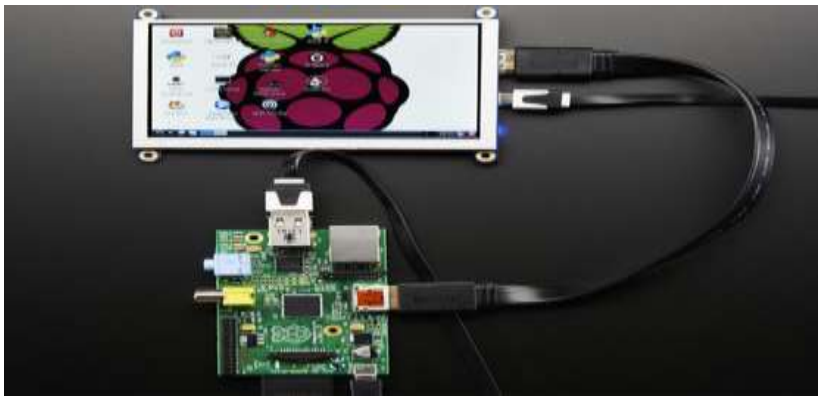


Figure 10: Connecting a raspberry pi to a 5" 800x480 TFT HDMI display

4. Connect a keyboard and mouse to the raspberry pi 3B+ via USB ports.

Figure 11: Connecting keyboard and mouse to the raspberry pi.

5. Switch on the power supply of the Raspberry pi 3B+ to turn it on.

4.1.2 Software unit development

The complete web application has six main interfaces: The **Signup page**: provides users with fingerprint authentication and signup option together with a username and a password. **Login page**: gives access to users to log in to the system and have access to remaining views. **Create Poll page**: provides an administrator with an option to create a voting poll(questions). **Available Polls**: users view available questions or voting polls so they can select and vote accordingly. **The voting page** gives users an option to vote for a candidate once. **Results page**: display the results of the ongoing voting to administrators and users after they vote.

Before building these pages in FLASK:

1. Download Visual studio code-OSS preferred IDE for this project on the raspberry pi 3B+.
    a. The first thing to do is to install a privacy guard key to decrypt and encrypt Code OSS packages that would be installed.

Figure 12: Adding a public GPG (privacy guard) key to verify packages of code OSS that would be installed

b. Install Code OSS with the following syntax in the terminal

```
pi@raspberrypi:~ $ sudo apt-get install code-oss=1.29.0-1539702286
```

Figure 13: Code for installing Code OSS IDE

2. Various files and folders in Appendix C were created
3. From the terminal, python three was installed; also, flask and all its frameworks and extensions mentioned were installed. After installation flask was initialized. Appendix B shows the FLASK server running (executing main.py as a python three file).
4. The main.py file contains all backend scripts used in the project, in this file, flask, all its extensions and framework (including Flask-SQL Alchemy for database purposes) were imported and initialized. Appendix D shows a visual representation of this.
5. Code for all the various pages (signup, login, create a poll, available polls, voting page, and result page) were written in main.py. Also, all HTML and CSS files for the frontend frameworks were written. This can be shown in Appendix E.
6. These processes come together to form the software unit of this system.

Chapter Five

5.0 Results

5.1 Web Application

The web application is hosted on the Raspberry pi 3B+. Several voting tests were done on the system to ensure the accuracy and reliability of the application. In testing the signup and login part of the system: the focus was placed on the accuracy of the fingerprint sensor R307. The more accurate the fingerprint sensor works, the more reliable the data fetched from the database is. The following diagrams show the accuracy score from the fingerprint sensor and the user data stored in the database after signing up in the system.

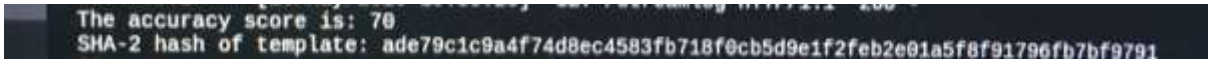


Figure 14: Accuracy score and hash of a fingerprint during the first login test.

Figure 15: Accuracy score and hash of a fingerprint during the second login test

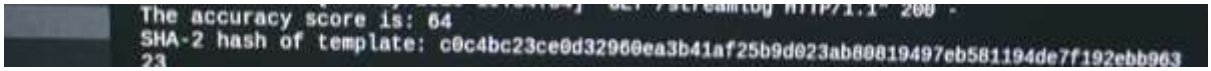


Figure 16: Accuracy score and hash of a fingerprint during the third login test

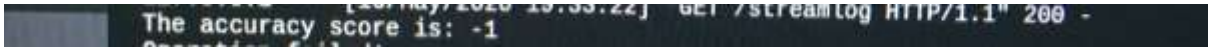


Figure 17: Accuracy score when a user is not found in the database during login

From the test conducted, the average accuracy score after three tests is 75%. Although the fingerprint sensor R307 might be constrained, it can be reliable in terms of reading fingerprints of users.

Before a user can cast a vote on the system, the user needs to select one of the available polls created by an administrator.

Figure 18: An image showing currently available voting polls of the system.

After selecting a poll, a user can vote for a candidate once. This voting is done by selecting one of the available candidates.

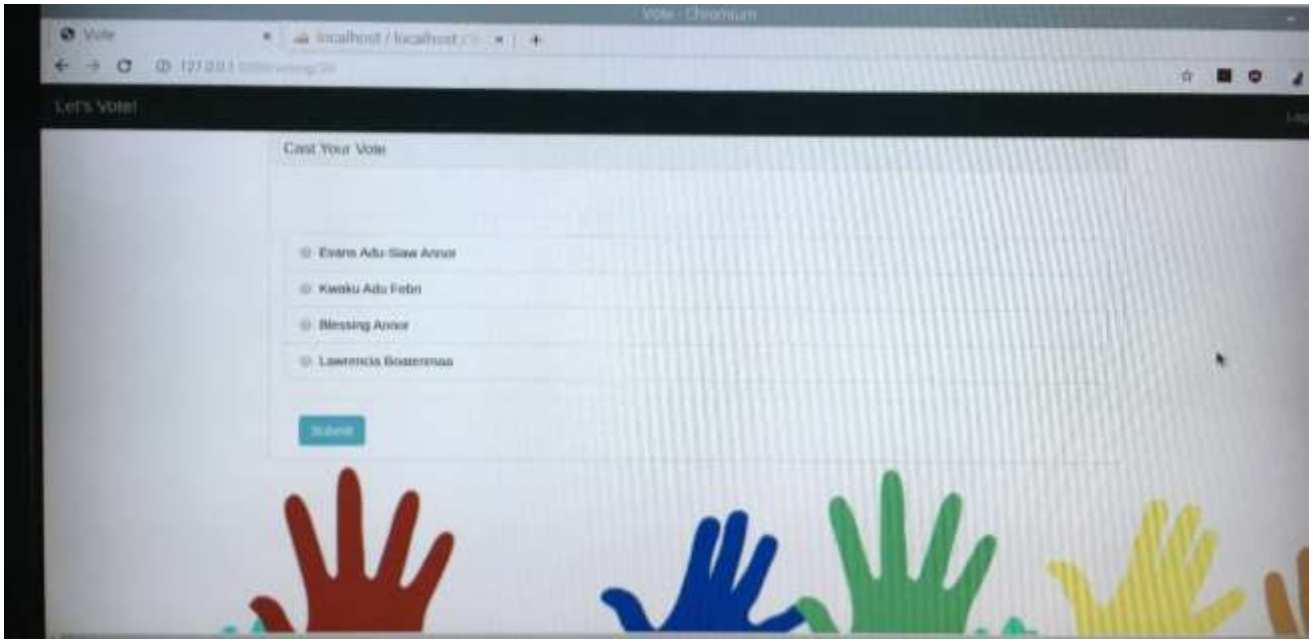


Figure 19: Diagram shows available candidates in the Senior School Prefect poll.

After voting for a candidate, a user can view the results page of the available polls.

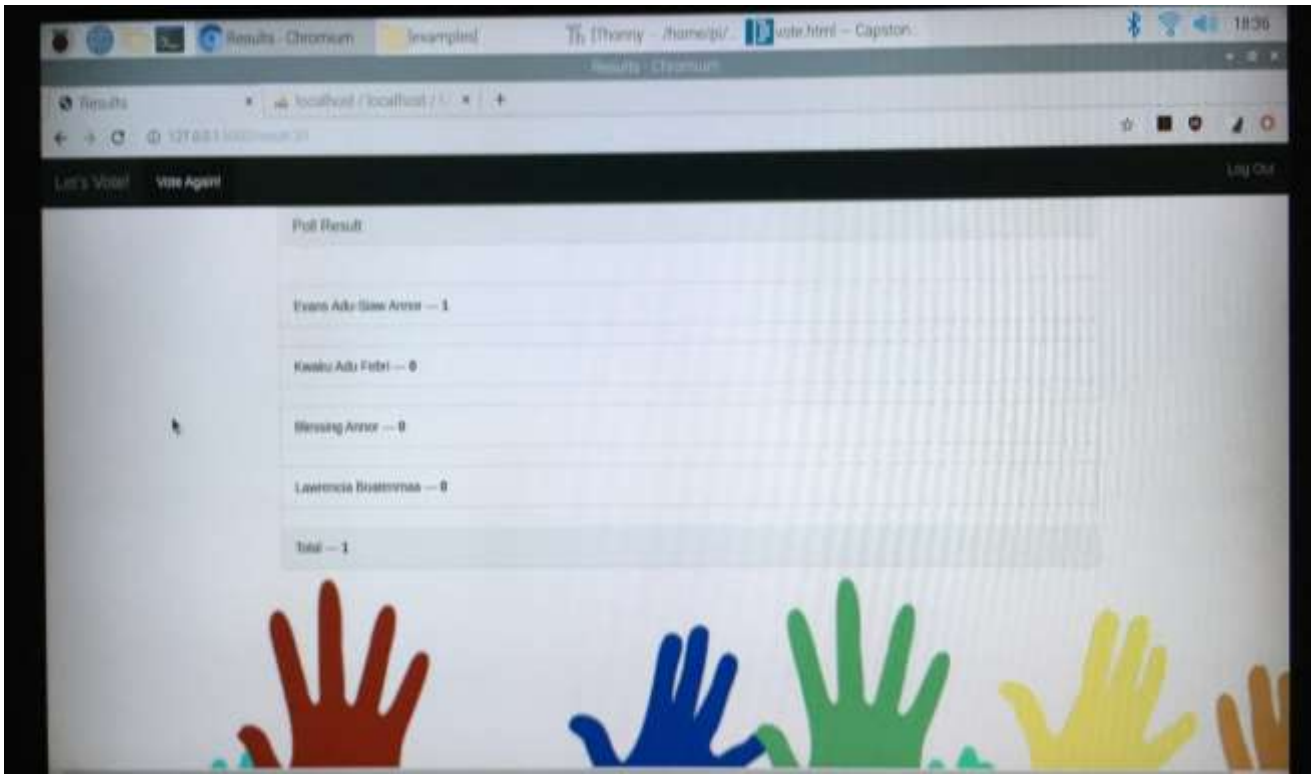


Figure 20: View of the School Prefect poll results page after voting.

On average, it takes approximately about two minutes to signup, login, cast a vote, and view the results.

When the registration (signup) process is already completed, it takes approximately a minute to cast a

vote and view the results. This is possible due to how easy it is to navigate through the pages of the voting system.

5.2 Database

The database is implemented in a way to avoid multiple voting and to make voting anonymous. The following images show two tables created in the database. The question table (which shows descriptions of all polls) and the candidate table (which shows the available candidates in the system and their number of votes cast).

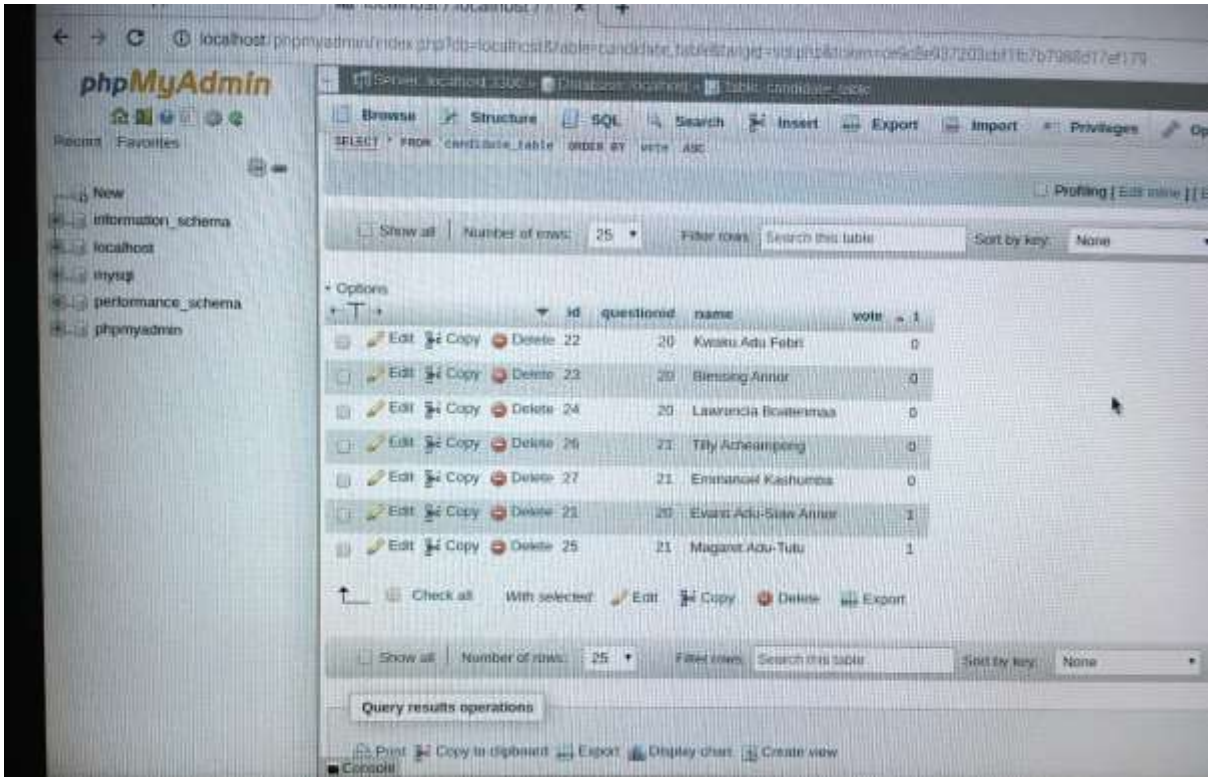


Figure 21: The table shows the candidate table created in the database.

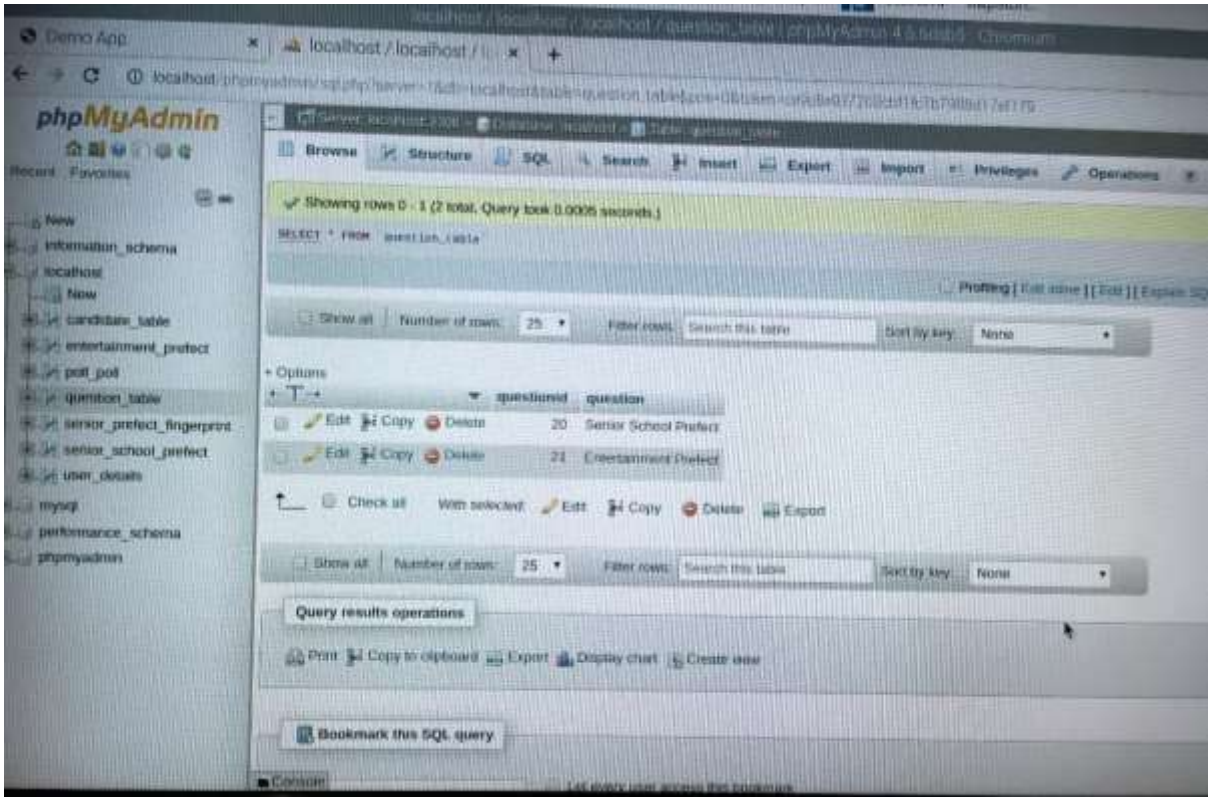


Figure 22: The table shows the questions table created in the database.

Chapter Six

6.0 Conclusion

6.1 Discussions

In this project, a secured, biometric authentication voting system has been presented. This system is safe, fair, and transparent; hence every user is obligated to vote once. The cost of the system is a onetime cost (recurrent cost), that is the cost to acquire all the components used to implement the system. This makes this system relatively cheaper as compared to the cost involved in the traditional voting system. All the equipment used in the project setup was carefully chosen and designed. The expected outcome of this project fulfils the objectives stated:

1. A hardware implementation to help students in senior high schools vote with ease.
2. Explores the possibilities of a highly secured voting system to make elections safe, fair, and transparent.

The result of the analysis proves that users can easily navigate and vote with the system within a shorter period.

6.2 Limitations

The current limitation of this project is that it is not hosted on an online platform yet. The focus was placed on developing a working system that fulfils all the requirement specifications stated.

6.3 Future works

The current system is required to be hosted on an online platform to allow it to be active and live on the internet. This can improve the overall security of the website and make the application readily available to all end users.

References

[1] Constitution of the Republic of Ghana [Ghana], January 7 1993, available at: https://www.refworld.org/docid/3ae6b5850.html [accessed January 27 2020]

[2] Thecommonwealth.org. (n.d.). Ghana: History | The Commonwealth. [online] Available at: https://thecommonwealth.org/our-member-countries/ghana/history [Accessed January 27 2020].

[3] Bafo,A.2011Biometric Voter Registration: What You Should Know, Available at:http://ghanamagazine.com/factual/politics/biometric-voter-registration-what-you-should-know/#.VEqTQPl_sTI [Accessed: Accessed January 17 2020]

[4] Aceproject.org. (2012). *Biometric voter verification in Ghana —*. [online] Available at: http://aceproject.org/electoral-advice/archive/questions/replies/382803236 [Accessed 27 Jan. 2020].

[5] K. Mythil, K. Kanagavalli and B. Shibi, *Ijcsmc.com*, 2014. [Online]. Available: https://ijcsmc.com/docs/papers/February2014/V3I2201499a62.pdf. [Accessed: 28-Jan- 2020].

[6] Ges.gov.gh. (n.d). Secondary Education Division. [online] Available at: https://ges.gov.gh/2019/07/15/secondary-education-division/ [Accessed 17 Jan. 2020].

[7] Fathima, S. (2013). Challenges of ICT in teaching learning process. International Journal of Engineering And Science, 2, 51-54.

[8] Nyavor, G. (2019). Christ, the King Int' l School, breaks ground with electronic voting system. Retrieved 17 January 2020, from https://www.myjoyonline.com/news/2019/April-2nd/christ-the-king-intl-school-breaks-ground-with-electronic-voting-system.php

[9] N. Akpablie, "Mawuko Girls SHS introduces e-voting in school elections," *Citi 97.3 FM - Relevant Radio. Always*, 2018. [Online]. Available: http://citifmonline.com/2018/02/mawuko-girls-shs-introduces-e-voting-school-elections/. [Accessed: 17- Jan- 2020].

[10] D. Kaku, "Nyankumasi Ahenkro S.H.S introduces e-voting in school election," *Ghanaweb.com*, 2018. [Online]. Available: https://www.ghanaweb.com/GhanaHomePage/NewsArchive/Nyankumasi-Ahenkro-S-H-S-introduces-e-voting-in-school-election-636791. [Accessed: 17- Jan- 2020].

[11] Lundie, R. (2019). *Electronic voting at federal elections – Parliament of Australia*. [online] Aph.gov.au. Available at: https://www.aph.gov.au/About_Parliament/Parliamentary_Departments/Parliamentary_Library/pubs/BriefingBook45p/ElectronicVoting [Accessed 13 Oct. 2019].

[12] Purandare, H., Saini, A., Pereira, F., Mathew, B. and Patil, P. (2018). *Application For Online Voting System Using Android Device - IEEE Conference Publication*. [online] Ieeexplore.ieee.org. Available at: https://ieeexplore.ieee.org/document/8537284 [Accessed 9 Nov. 2019].

[13] Chakole, J. and Pardhi, P. (2013). *The Design of Web-Based Secure Internet Voting System for Corporate Election* [online]semanticscholar.org. Available at: https://www.semanticscholar.org/paper/The-Design-of-Web-Based-Secure-Internet-Voting-for-Chakole-Pardhi/2ccf0ca31ed727329177b263d432e2b34afe8b28 [Accessed 9 Nov. 2019].

[14] D. Shah and V. Bharadi, "IoT Based Biometrics Implementation on Raspberry Pi," *Procedia Computer Science*, vol. 79, pp. 328-336, 2016. Available: 10.1016/j.procs.2016.03.043.

[15] G. Dieter and L. Schmidt, Engineering Design, New York: McGraw-Hill, 2009.

[16] Patel, S., 2014. *Developing Responsive Web Applications With AJAX And Jquery*. Birmingham, U.K.: Packt Pub.

[17] Amazon.com. n.d. [online] Available at: https://www.amazon.com/Raspberry-Pi-MS-004-00000024-Model-Board/dp/B01LPLPBS8 [Accessed 30 March 2020].

[18] Amazon.com. n.d. [online] Available at: https://www.amazon.com/Elecrow-800x480-Interface-Supports-Raspberry/dp/B013JECYF2 [Accessed 30 March 2020].

[19] Dwyer, G., Aggarwal, S., and Stouffer, J., 2017. *Flask: building python web services*. Birmingham, United Kingdom: PACKT Publishing Limited.

Appendix

A: Hardware Devices used in the development of the Biometric System

Appendix A: Hardware devices

Raspberry pi 3B+ 	5" 800x480 TFT HDMI 
Fingerprint sensor R307 	PL2303 USB TTL Cable FTDI 

B: Flask server running as localhost to the raspberry pi.

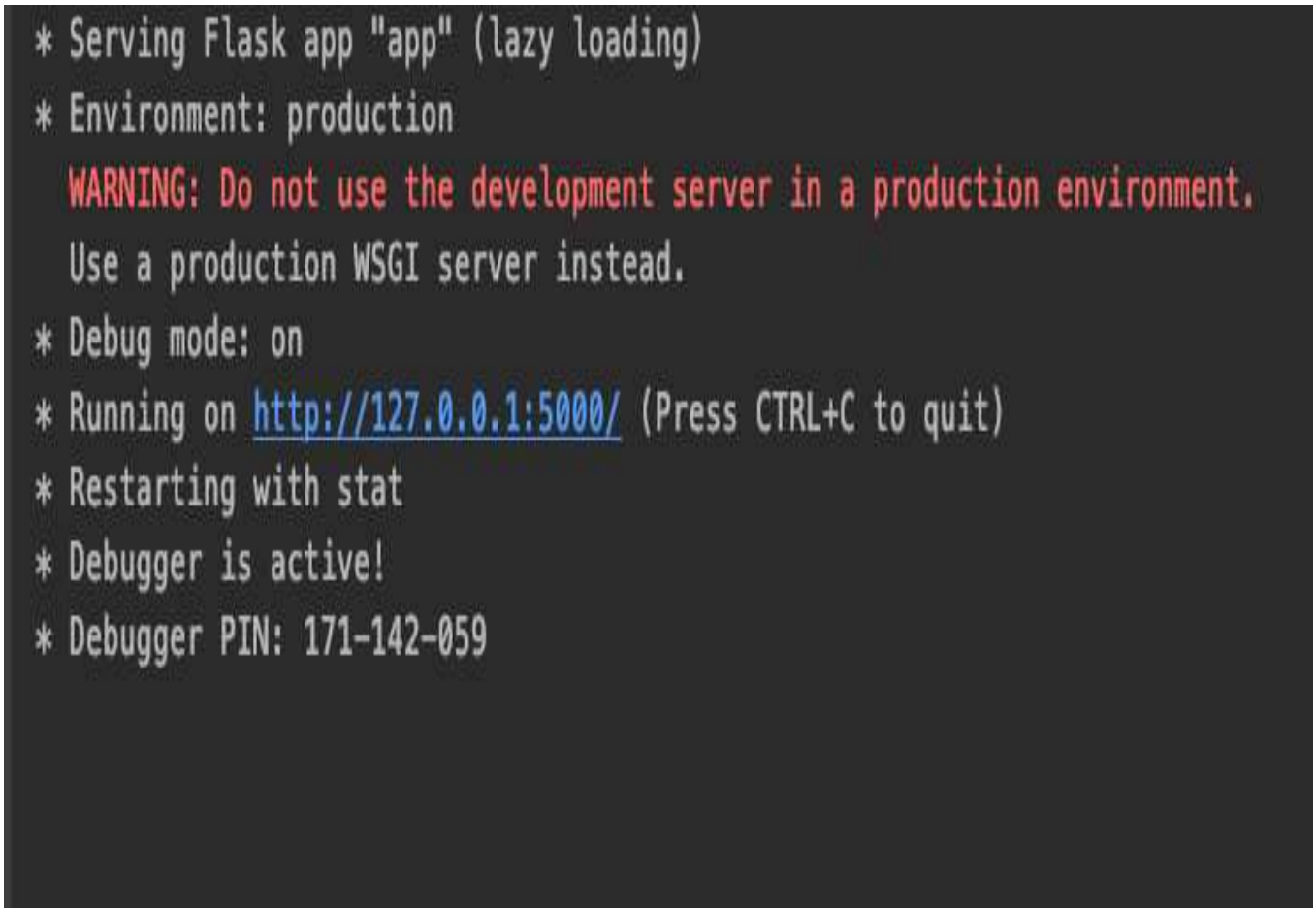


Figure 23: Flask web server running on Raspberry pi

C: Folder tree of the system

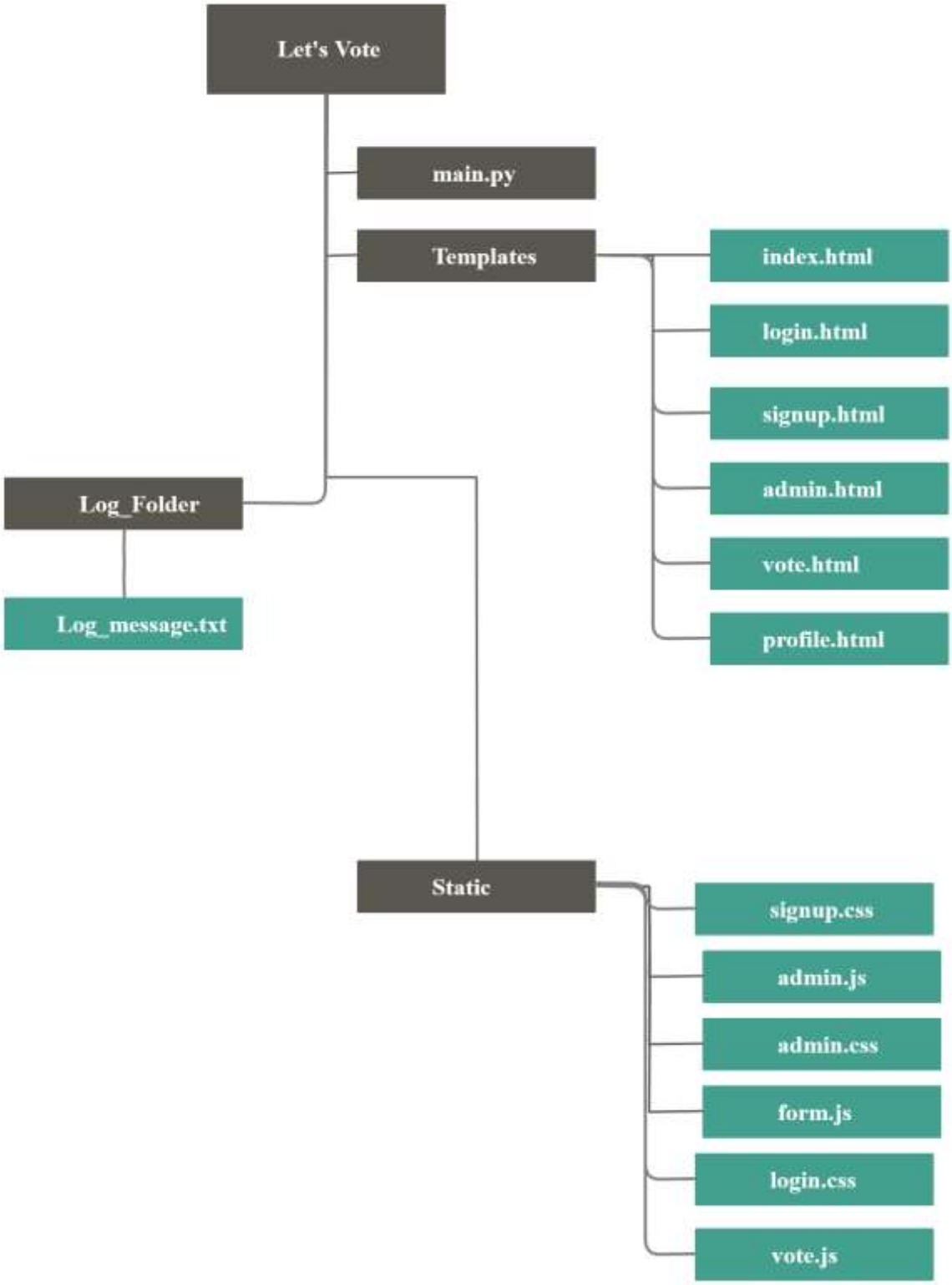


Figure 24: Directory of the project

D: Flask extensions being imported in the project

Figure 25: Image of various flask extensions and frameworks imported

E: Code implementation of the voting system

```
@app.route('/login',methods=['GET','POST'])
def login():
    error=None
    form =LoginForm()

    #if submission succesfull
    if form.validate_on_submit():
        #call fingerscan route
        result=fingersearch()

        if result is None:
            error= 'Invalid Username, Password or FingerprintID Please try again.'

        else:
            #get hashed prints form the result
            prints= result[1]
            ScanHashed_prints= prints
            #quering database for the user
            user=User.query.filter_by(Username=form.username.data).first() #only use first name for checking
            print(user.id)
            admin=User.query.filter_by(Username='admin').first() #only use for checking admin
            user_json = json.dumps(user.id,cls=AlchemyEncoder)
            session["user_json"]=user_json
            print("user_json" in session)
            #if user exist,check password, and fingerprint
            if admin:
                if check_password_hash(user.Password,form.password.data) and (user.Username== 'admin'):
                    login_user(admin,remember=form.remember.data)
                    return redirect(url_for('admin'))
```

Figure 26: Code implementation of the login system

```
            user_json = json.dumps(user.id,cls=AlchemyEncoder)
            session["user_json"]=user_json
            print("user_json" in session)
            #if user exist,check password, and fingerprint
            if admin:
                if check_password_hash(user.Password,form.password.data) and (user.Username== 'admin'):
                    login_user(admin,remember=form.remember.data)
                    return redirect(url_for('admin'))

            if user:
                if check_password_hash(user.Password,form.password.data) and (user.Fingerprint== ScanHashed_prints):
                    login_user(user,remember=form.remember.data)
                    return redirect(url_for('votingpoll'))


                error = 'Invalid Username, Password or FingerprintID Please try again.'
                #if username/password is invalid
                file = os.path.join(
                    app.config['LOG_FOLDER'], session.get('log_file')
                )

                filepath  = os.path.join('LOG_FOLDER','log_messages.txt')
                with open(filepath,'w') as f:
                    pass



    filepath  = os.path.join('LOG_FOLDER','log_messages.txt')
    with open(filepath,'w') as f:
        pass

    return render_template('login.html',form = form)
```

Figure 27: Code implementation of the login system (continues)

```
#for sign up page
@app.route('/signup',methods=['GET','POST'])
def signup():
    error=None
    form = Register()

#if user wants to submit the form...
    if form.validate_on_submit():
        session['Username'] = form.username.data
        result=fingerscan()

        if not result:
            error= 'Fingerprint ID Error, Please try Again'

            file= os.path.join(
                app.config['LOG_FOLDER'], session.get('log_file')
            )

            filepath  = os.path.join('LOG_FOLDER','log_messages.txt')
            with open(filepath,'w') as f:
                pass

        else:
            hashed_prints=result[1]
            #generate hash password
            hashed_password = generate_password_hash(form.password.data,method='sha256')
            #passing parameters of User to new_user to be pushed into the database
           # global table_prints
            table_prints= hashed_prints
            new_user=User(
```

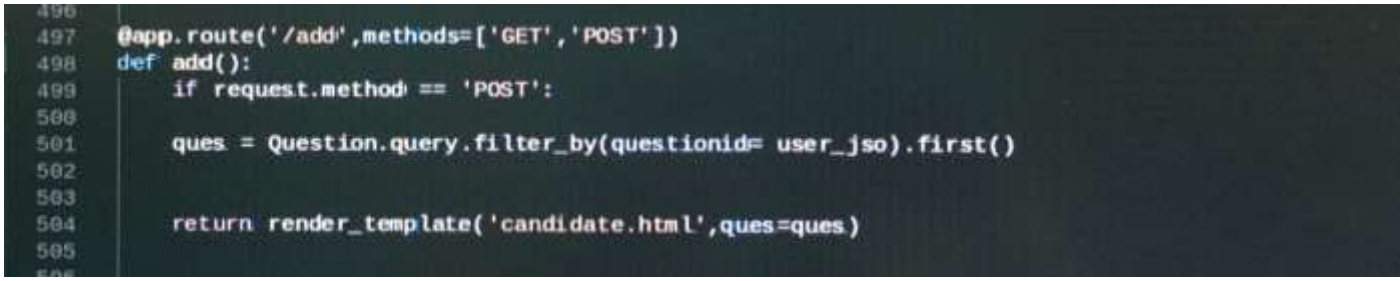
Figure 28: Code implementation of the signup system

```
        else:
            hashed_prints=result[1]
            #generate hash password
            hashed_password = generate_password_hash(form.password.data,method='sha256')
            #passing parameters of User to new_user to be pushed into the database
           # global table_prints
            table_prints= hashed_prints
            new_user=User(
                Username=form.username.data,
                Password=hashed_password,
                Date=form.created,
                Fingerprint=hashed_prints
            )
            db.session.add(new_user)
            db.session.commit()
            file= os.path.join(
                app.config['LOG_FOLDER'], session.get('log_file')
            )

            filepath  = os.path.join('LOG_FOLDER','log_messages.txt')
            with open(filepath,'w') as f:
                pass

            return redirect (url_for('dashboard'))


    filepath  = os.path.join('LOG_FOLDER','log_messages.txt')
    with open(filepath,'w') as f:
        pass

    return render_template('signup.html',error=error,form=form)
```

Figure 29: Code implementation of signup System(continues)

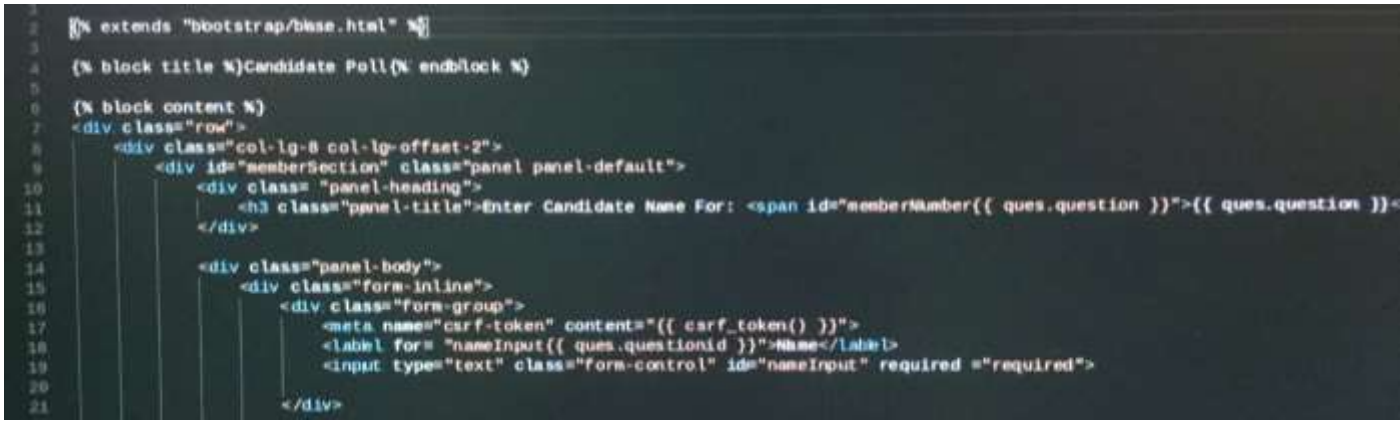
```
@app.route('/create',methods=['GET','POST'])
def create():

    form = CreatePollForms()
    ques = Question.query.filter_by(question=form.question.data).first()
    context ={'form':form}
    if form.validate_on_submit():
        ques = Question.query.filter_by(question=form.question.data).first()

        if ques is None:
            new_ques=Question(question=form.question.data)
            db.session.add(new_ques)
            db.session.commit()
            user_jso = json.dumps(new_ques.questionid,cls=AlchemyEncoder)
            session["user_jso"]=user_jso

            return redirect('add')



    return render_template('create.html',ques=ques,form=form)
```

Figure 30 Code Implementation of creating a new poll (Backend script)



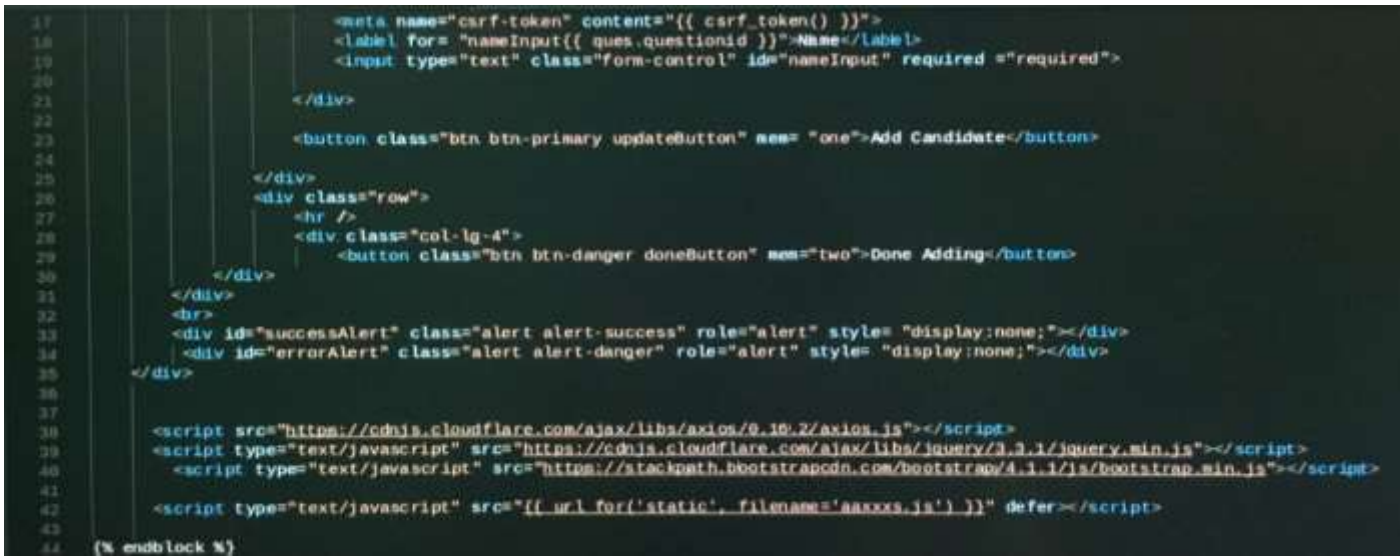
```
@app.route('/add',methods=['GET','POST'])
def add():
    if request.method == 'POST':

    ques = Question.query.filter_by(questionid= user_jso).first()


    return render_template('candidate.html',ques=ques)
```

Figure 31:Code Implementation of adding a new candidate to the poll created (Backend script)



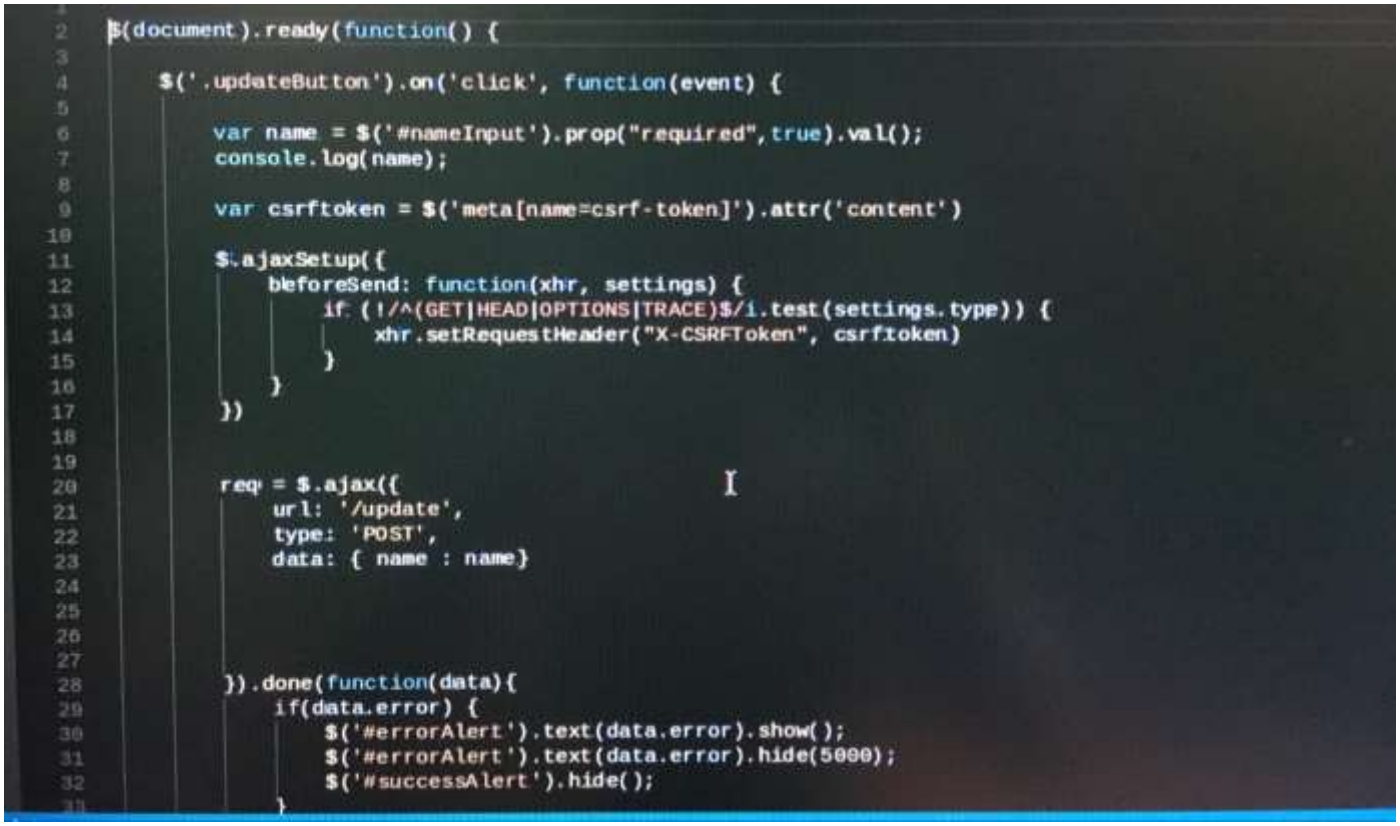
```
{% extends "bootstrap/base.html" %}

{% block title %}Candidate Poll{% endblock %}

{% block content %}
<div class="row">
    <div class="col-lg-8 col-lg-offset-2">
        <div id="memberSection" class="panel panel-default">
            <div class= "panel-heading">
                <h3 class="panel-title">Enter Candidate Name For: <span id="memberNumber{{ ques.question }}">{{ ques.question }}<
            </div>

            <div class="panel-body">
                <div class="form-inline">
                    <div class="form-group">
                        <meta name="csrf-token" content="{{ csrf_token() }}">
                        <label for= "nameInput{{ ques.questionid }}">Name</label>
                        <input type="text" class="form-control" id="nameInput" required ="required">

                    </div>
```

Figure 32: HTML implementation of adding a new candidate to the poll created (Frontend script)



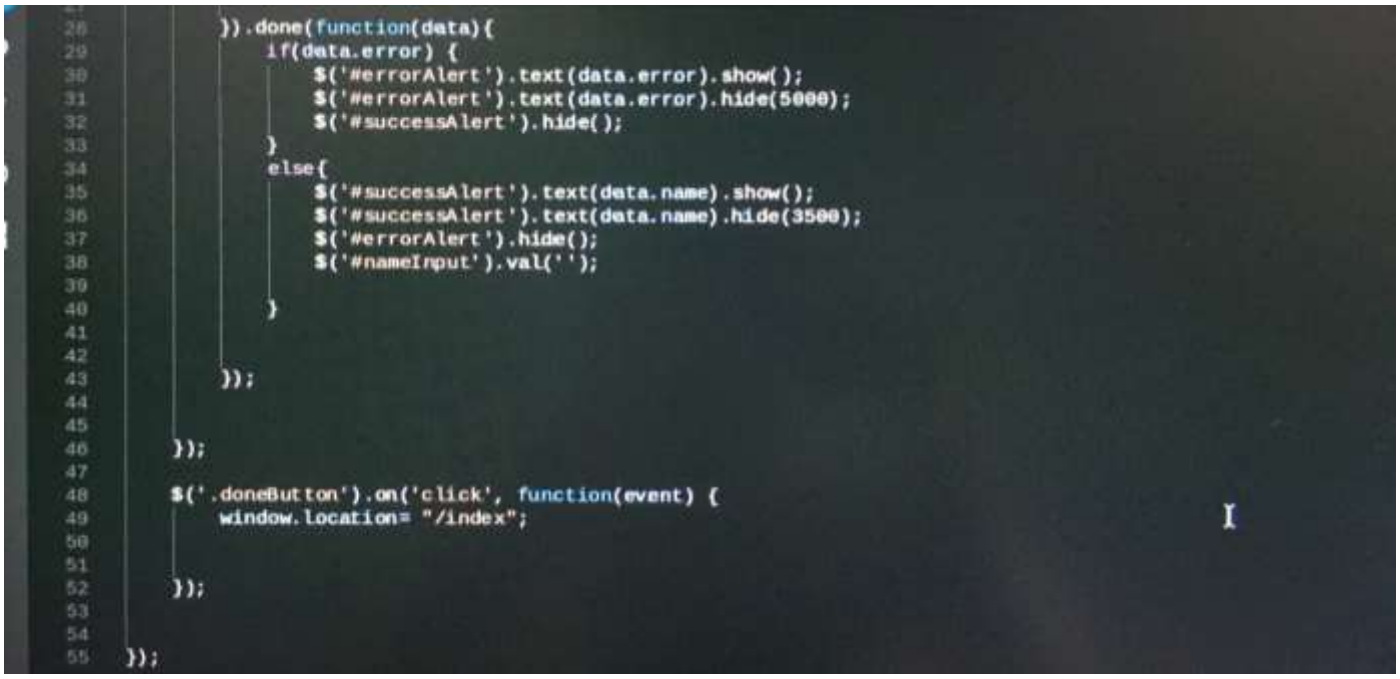
```
                        <meta name="csrf-token" content="{{ csrf_token() }}">
                        <label for= "nameInput{{ ques.questionid }}">Name</label>
                        <input type="text" class="form-control" id="nameInput" required ="required">

                    </div>

                    <button class="btn btn-primary updateButton" mem= "one">Add Candidate</button>

                </div>
                <div class="row">
                    <hr />
                    <div class="col-lg-4">
                        <button class="btn btn-danger doneButton" mem="two">Done Adding</button>
            </div>
        </div>
        <br>
        <div id="successAlert" class="alert alert-success" role="alert" style= "display:none;"></div>
        <div id="errorAlert" class="alert alert-danger" role="alert" style= "display:none;"></div>
    </div>


    <script src="https://cdnjs.cloudflare.com/ajax/libs/axios/0.16.2/axios.js"></script>
    <script type="text/javascript" src="https://cdnjs.cloudflare.com/ajax/libs/jquery/3.3.1/jquery.min.js"></script>
      <script type="text/javascript" src="https://stackpath.bootstrapcdn.com/bootstrap/4.1.1/js/bootstrap.min.js"></script>

    <script type="text/javascript" src="{{ url_for('static', filename='aaxxxs.js') }}" defer></script>

{% endblock %}
```

Figure 33: HTML implementation of adding a new candidate to the poll created (Frontend script) continues



```
$(document).ready(function() {

    $('.updateButton').on('click', function(event) {

        var name = $('#nameInput').prop("required",true).val();
        console.log(name);

        var csrftoken = $('meta[name=csrf-token]').attr('content')

        $.ajaxSetup({
            beforeSend: function(xhr, settings) {
                if (!/^(GET|HEAD|OPTIONS|TRACE)$/i.test(settings.type)) {
                    xhr.setRequestHeader("X-CSRFToken", csrftoken)
                }
            }
        })


        req = $.ajax({
            url: '/update',
            type: 'POST',
            data: { name : name}



        }).done(function(data){
            if(data.error) {
                $('#errorAlert').text(data.error).show();
                $('#errorAlert').text(data.error).hide(5000);
                $('#successAlert').hide();
            }
```

Figure 34: JavaScript implementation of adding a new candidate to the poll created (Frontend script)



```
        }).done(function(data){
            if(data.error) {
                $('#errorAlert').text(data.error).show();
                $('#errorAlert').text(data.error).hide(5000);
                $('#successAlert').hide();
            }
            else{
                $('#successAlert').text(data.name).show();
                $('#successAlert').text(data.name).hide(3500);
                $('#errorAlert').hide();
                $('#nameInput').val('');

            }


        });


    });

    $('.doneButton').on('click', function(event) {
        window.location= "/index";


    });


});
```

Figure 35: JavaScript implementation of adding a new candidate to the poll created (Frontend script) continues

```
459
460  @app.route('/update',methods=['POST'])
461  def update():
462      name = request.form['name']
463      can = Can.query.filter_by(name=name).first()
464      user_jso=session["user_jso"]
465      print(name)
466      print(user_jso)
467      if name:
468          if can is None:
469              new_can=Can(questionid=user_jso,name=name,vote=0)
470              db.session.add(new_can)
471              db.session.commit()
472
473              return jsonify({'name':'Candidate added Succesfully'})
474
475
476      return jsonify({'error': 'Please Enter Candidate Name!'})
```

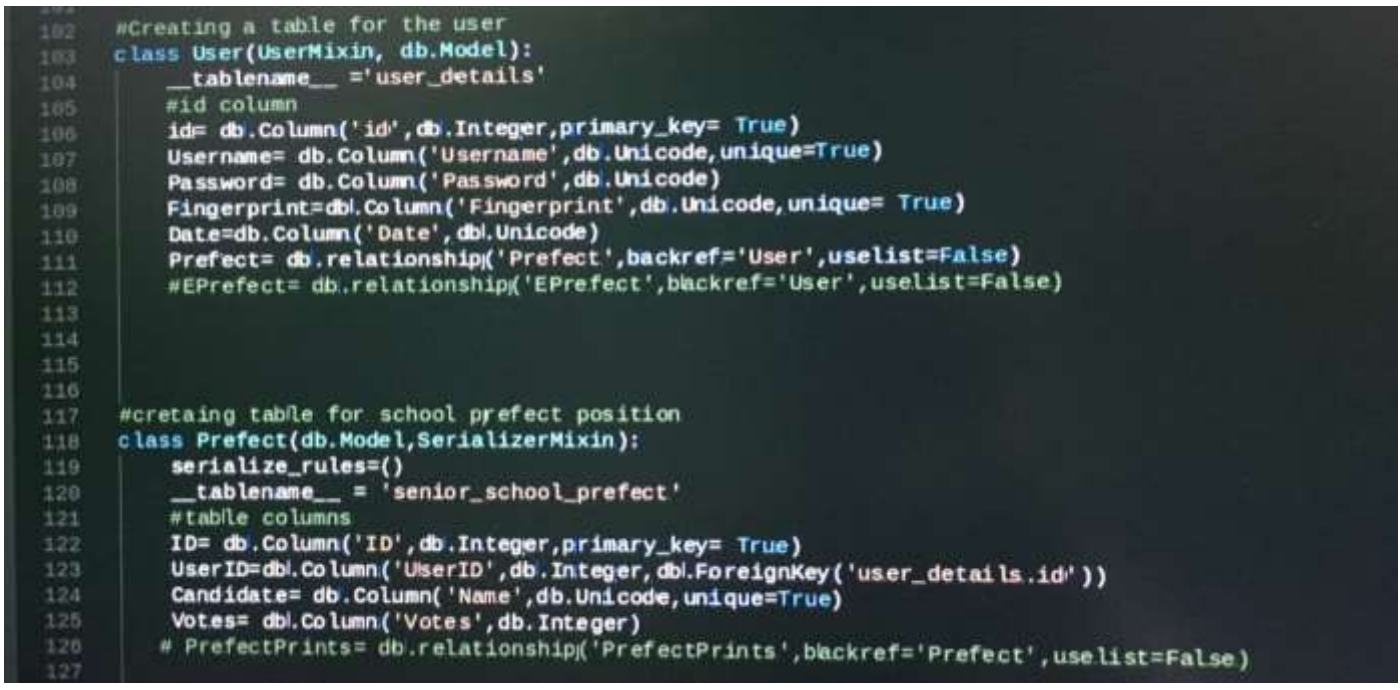
Figure 36: Update route gets information posted from add page and adds it to the database

```
494  @app.route('/voting/<poll_id>',methods=['GET','POST'])
495  @login_required
496  def voting(poll_id):
497      form = VotingForms()
498      context ={'form':form}
499      poll = Can.query.filter_by(questionid=poll_id).all()
500      global id_poll
501      id_poll=poll_id
502
503      if request.method == 'POST':
504          selected_option= request.form['poll']
505          global select
506          select = selected_option
507
508          #query for table, check if user has already voted
509          user_print=PrefectPrints.query.filter_by(UserID=fp).all()
510          name = Can.query.filter_by(name=select).first()
511
512
513
514          if user_print is None:
515              name.vote=name.vote+1
516              db.session.commit()
517
518              new_name = PrefectPrints(questionid=poll_id,UserID=fp)
519
520              db.session.add(new_name)
521              db.session.commit()
522              return redirect(url_for('result',poll_id=poll_id))
```

Figure 37: Code implementation of the voting page (Backend Script)

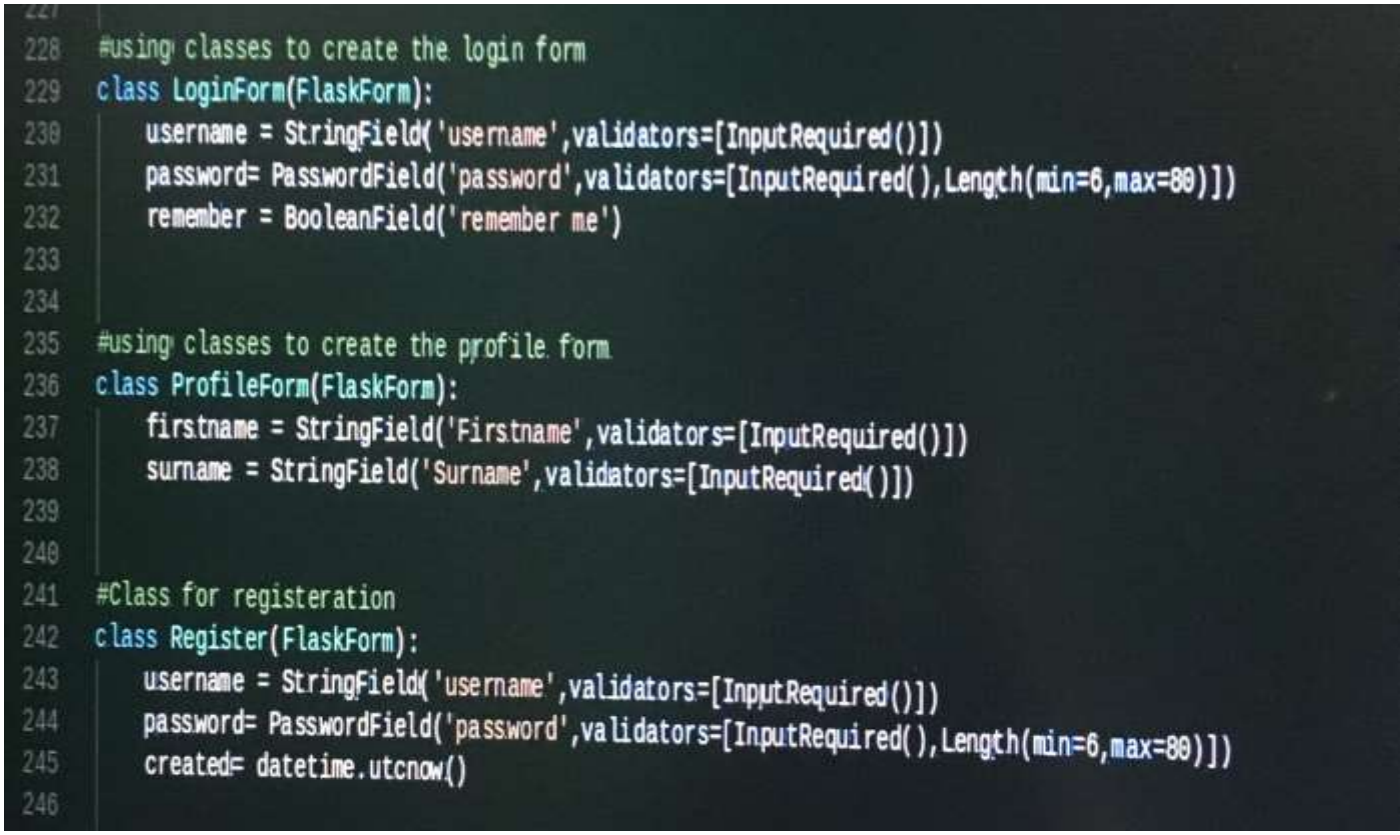
```
        elif user_print is not None:
            l= []
            for i in user_print:
                l.append(i.questionid)

            print(l)

            print(selected_option)
            choose=Can.query.filter_by(name=selected_option).first()
            if (int(poll_id) not in l ):

                name.vote=name.vote+1
                db.session.commit()

                new_name = PrefectPrints(questionid=ppll_id,UserID=fp)


                db.session.add(new_name)
                db.session.commit()
                return redirect(url_for('result',ppll_id=ppll_id))

        else:
            return ('Invalid form',404)

        return redirect(url_for('result',poll_id=poll_id))
    return render_template('vote.html',context=context,form=form)
```

Figure 38: Figure 19 Code implementation of the voting page (Backend Script) continues



```
#Creating a table for the user
class User(UserMixin, db.Model):
    __tablename__ ='user_details'
    #id column
    id= db.Column('id',db.Integer,primary_key= True)
    Username= db.Column('Username',db.Unicode,unique=True)
    Password= db.Column('Password',db.Unicode)
    Fingerprint=db.Column('Fingerprint',db.Unicode,unique= True)
    Date=db.Column('Date',db.Unicode)
    Prefect= db.relationship('Prefect',backref='User',uselist=False)
    #EPrefect= db.relationship('EPrefect',backref='User',uselist=False)



#cretaing table for school prefect position
class Prefect(db.Model,SerializerMixin):
    serialize_rules=()
    __tablename__ = 'senior_school_prefect'
    #table columns
    ID= db.Column('ID',db.Integer,primary_key= True)
    UserID=db.Column('UserID',db.Integer,db.ForeignKey('user_details.id'))
    Candidate= db.Column('Name',db.Unicode,unique=True)
    Votes= db.Column('Votes',db.Integer)
   # PrefectPrints= db.relationship('PrefectPrints',backref='Prefect',uselist=False)
```

Figure 39: Creating database tables as classes in code OSS IDE



```
#using classes to create the login form
class LoginForm(FlaskForm):
    username = StringField('username',validators=[InputRequired()])
    password= PasswordField('password',validators=[InputRequired(),Length(min=6,max=80)])
    remember = BooleanField('remember me')


#using classes to create the profile form
class ProfileForm(FlaskForm):
    firstname = StringField('Firstname',validators=[InputRequired()])
    surname = StringField('Surname',validators=[InputRequired()])


#Class for registeration
class Register(FlaskForm):
    username = StringField('username',validators=[InputRequired()])
    password= PasswordField('password',validators=[InputRequired(),Length(min=6,max=80)])
    created= datetime.utcnow()
```

Figure 40: Creating Flask Forms as class