**SOFTWARE REQUIREMENT DOCUMENT: MUST ONLINE VOTING WEBSITE SYSTEM.**

**INTRODUCTION**

**PURPOSE**

The purpose of this document is to build an online system for university voting to ease the election process.

**DOCUMENT CONVENTIONS**

This document uses the following conventions.

|  |  |
| --- | --- |
| DB | Database |
| ER | Entity Relationship |

**INTENDED AUDIENCE AND READING SUGGESTIONS**

This project is a prototype for the online voting system and it is restricted within the college premises. This has been implemented under the guidance of the course instructor. This project is useful for the election process.

**PROJECT SCOPE**

The purpose of the Must online voting system is to ease the Election process and to create a convenient and easy-to-use application for students, trying to vote for their leaders. We will have a database server supporting hundreds of students who will be voting during the election. Above all, we hope to provide a comfortable user experience along with the free pricing available.

**SYSTEM LIMITATION/ CONSTRAINTS**

* An internet connection is required to cast the votes

**TOOL AND TECHNOLOGY**

Following are the tools and technologies that will be used for the development of the web system.

|  |  |  |
| --- | --- | --- |
| **Tools**  **and**  **Technologies** | **Tools** | **Version** |
| Virtual studio code | 1.77 |
| Notepad++ | V8.5.2 |
| **Technology** | **Version** |
| HTML | 5 |
| CSS | 3 |
| PHP | 7.4 |
| JavaScript | ES6 |

**OVERALL DESCRIPTION**

**PRODUCT PERSPECTIVE**

A distributed voting database system stores the following information.

* **User accounts:**

These accounts will contain information about students including course, year of study, and name of the student.

* **Ballots:**

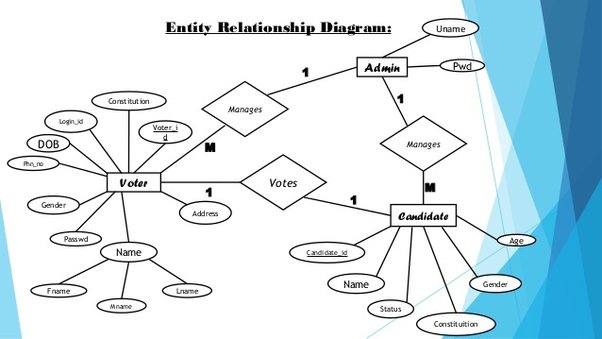
Ballots will include ballot data.

* **Votes:**

Inside the database of votes, the election result will be included.

**PRODUCT FEATURES**

The major features of the MOVS database system as shown in the below entity-relationship (**ER model**)



The diagram shows the layout of the MOVS database system using an entity–relationship model.

**OPERATING ENVIRONMENT**

The operating environment for the MOVS is listed below.

* client/server system
* Operating system: Windows, MAC OS.
* database: sql+ database
* platform: HTML/CSS/PHP /JavaScript.

**EXTERNAL INTERFACE REQUIREMENTS**

**USER INTERFACES**

* Front-end language: HTML, JavaScript, and CSS.
* Back-end: SQL+PHP.

**HARDWARE INTERFACE**

* A browser that supports HTML & JavaScript.
* Windows & Mac OS.
* Computer & Phone with 4RAM+
* Processor – intel core i3+
* Phone processor’s 2GHz+

**SOFTWARE INTERFACES**

The following is the software used for the Must online voting System.

|  |  |
| --- | --- |
| **Software used** | **Description** |
| Operating System | We have chosen Windows and MacOS for their best interfaces and user-friendliness. |
| Database | To save the user info, ballot data, and election results we have chosen an SQL database. |

**COMMUNICATION INTERFACES**

This project supports all types of web browsers. We are using simple languages such as HTML, CSS, PHP, and JavaScript.

**FUNCTIONAL REQUIREMENTS**

The following are the functional requirement of our system:

1. **Authorize User**

It will authorize every user before letting him vote.

1. **Add Nominee**

It will allow adding nominees to the system.

1. **Vote Virtually**

Voters can vote in any part of the world through the web app.

1. **Vote anonymously**

Data of the voter will be safe in Database.

1. **Vote Once**

Any authorized can only vote once in an election.

1. **Accuracy**

The system shall record and count all the votes and shall do so correctly.

**NONFUNCTIONAL REQUIREMENTS**

The following are the nonfunctional requirement of our system:

1. **Registration**

The voter registration shall be done in person only.

1. **Secrecy / Privacy**

No one should be able to determine how any individual voted.

1. **Reliability**

Election systems should work robustly, without loss of any votes, even in the face of  
numerous failures, including failures of voting machines and total loss of network communication.

1. **Simplicity**

The system shall be designed to be extremely simple.

1. **Performance**

Our system will be the fastest vote-casting software.

1. **Support**

As it is written in PHP so it can open on any platform like mobile, laptop, or tab browsers.

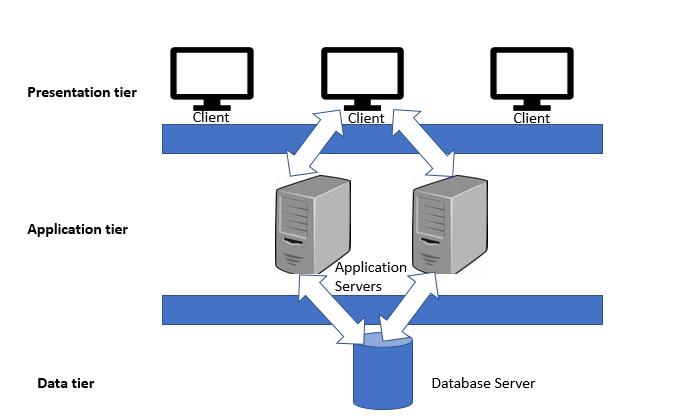
**SOFTWARE DESIGN DOCUMENT: MUST ONLINE VOTING WEBSITE SYSTEM (MOVS)**

**Introduction:** The purpose of this software design document is to outline the system architecture and design for a Must online voting website system. This system will allow registered voters to securely and anonymously cast their votes in various elections.

**Project Requirements:**

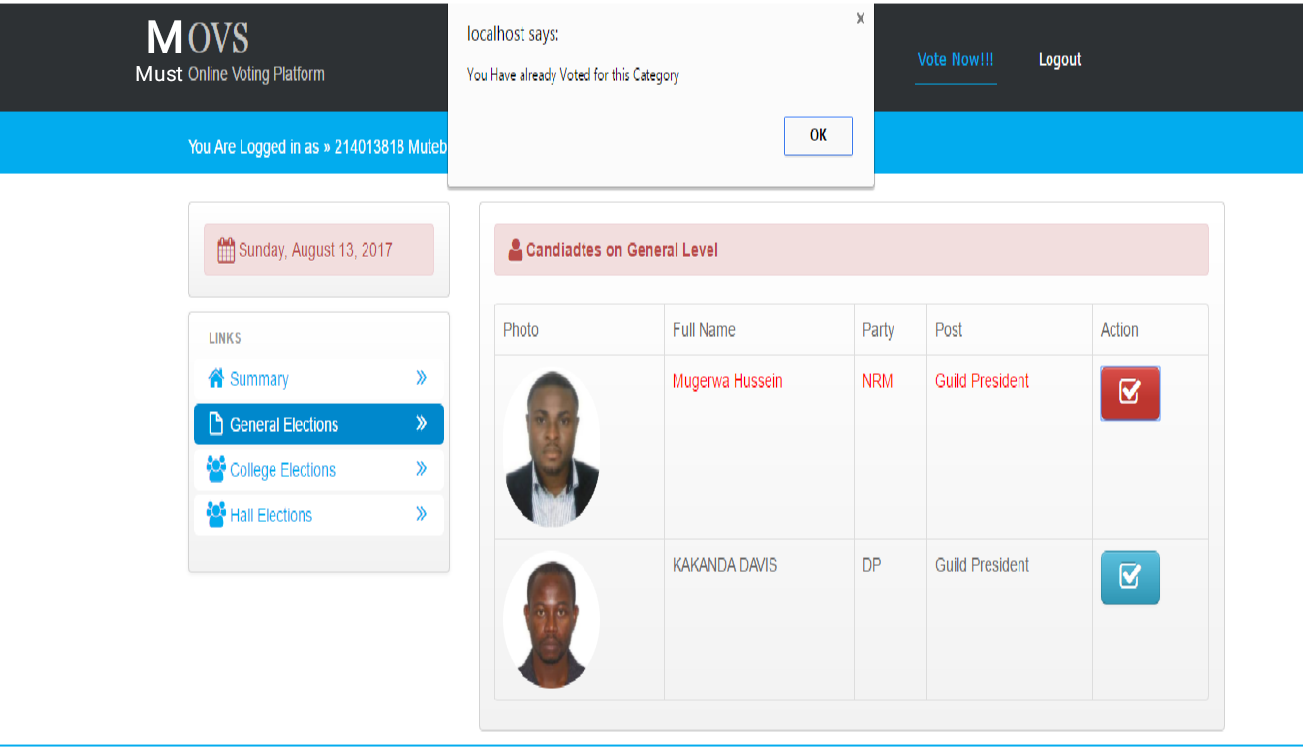
1. User Registration: The system must allow users to register for an account to access the voting system.
2. Voter Authentication: The system must authenticate voters using a secure login process.
3. Ballot Creation: The system must allow authorized users to create ballots for various elections.
4. Vote Casting: The system must allow registered and authenticated voters to cast their votes for elections.
5. Results Reporting: The system must generate accurate and timely results for elections.

**Architecture:** The system will use a three-tier architecture, with the **presentation tier, application tier,** and **data tier.**



* Presentation Tier: The presentation tier will include the website user interface, which will allow voters to interact with the system.
* Application Tier: The application tier will include business logic and server-side programming, which will handle user authentication, ballot creation, vote casting, and results reporting.
* Data Tier: The data tier will include the database that stores user account information, ballot data, and election results.

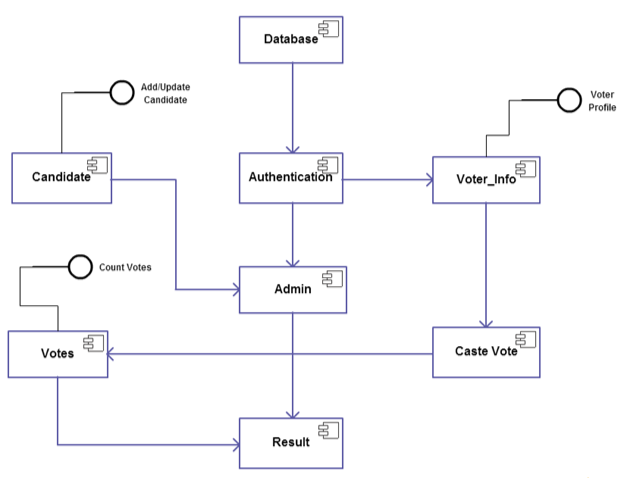
**User Interface:** The website user interface will allow voters to register for an account, authenticate themselves, view available elections, and cast their votes. The user interface will be designed to be simple and intuitive, with clear instructions and error messages.



**Data Design:** The system's database will be designed to store user account information, ballot data, and election results. The database schema will include tables for user accounts, ballots, and votes. The database will use SQL for data storage and retrieval.

**Algorithms and Logic:** The system will use various algorithms and logic to ensure the security and accuracy of the voting process. These will include:

* User authentication: The system will use strong **encryption algorithms** and **secure hashing methods** to protect user account information and prevent unauthorized access.
* Ballot creation: The system will allow authorized users to create ballots using a user-friendly interface, and will enforce strict validation rules to ensure ballot accuracy and consistency.
* Vote casting: The system will use a secure, anonymous, and auditable voting process to ensure that each voter's ballot is counted accurately.
* Results reporting: The system will generate accurate and timely results for elections using a combination of real-time and batch processing methods.



**Testing:** The system will be tested thoroughly using various testing methods, including unit testing, integration testing, and user acceptance testing. The testing process will ensure that the system is secure, reliable, and easy to use.

**Performance:** The system will be optimized for performance and scalability, with a focus on minimizing response times and maximizing throughput. The system will use caching, load balancing, and other performance optimization techniques to ensure that the website can handle a large number of concurrent users.

**Security:** The system will be designed to be highly secure, with a focus on preventing unauthorized access, protecting user data, and ensuring the integrity of the voting process. The system will use strong encryption algorithms, secure authentication methods, and strict access control measures to ensure that the system is secure from malicious attacks.

**Conclusion:** This software design document outlines the architecture and design for a secure and reliable voting website system. The system will be optimized for performance, scalability, and security, and will provide a simple and intuitive user interface for voters to cast their votes securely and anonymously.