

Republic of the Philippines

**Polytechnic University of the Philippines**

**Department of Information Technology**

Santa Maria, Bulacan campus

1st Semester, School Year 2018 - 2019

**Web-Based Voting System for Student Government Elections of Mater Dei Academy**

**Bachelor of Science in Information Technology**

Program

Instructor

**MR. JAYSON R. HERMOGENES**

Proponents

**BRYAN JOEL M. EDURIA**

**ERICA MAE C. GARRIDO**

**CHRISTIAN V. REYES**

**JERICHO IAN E. TORRES**

BSIT 4-2

1. **Project Title**

**WEB-BASED VOTING SYSTEM FOR STUDENT GOVERNMENT**

**ELECTIONS OF MATER DEI ACADEMY**

1. **Project Overview**

Over the years, part of the Mater Dei Academy’s curriculum in developing its students into well-rounded individuals is involving them in extra-curricular activities such as Student Government. This aims to develop the potential and skills of students when it comes to leadership, who in turn, may become future leaders in the different aspects of the society.

This project intends to solve the project beneficiary’s problems in their current system of voting, which is done manually by using paper ballots and manual counting of votes. However, having a voting system done in a manual fashion has its disadvantages. First, the method is quite costly because the school needs to print ballots every year since the names of the candidates in the yearly elections are dynamic. Secondly, the results are prone to fraud because the ballots can be tampered, and the vote counts can be manipulated. And lastly, because the votes are being counted manually, it tends to consume a lot of time and is prone to inaccurate results due to human errors.

Here, the proponents are suggesting the use of a web-based voting system as a replacement for the manual voting system which has been the method that the school has been employing ever since they started conducting elections. A web-based application/system, as defined by Wikipedia, is a type of client-server computer program that runs through a browser, either from a mobile phone, laptop or a desktop PC. The application and the database are being hosted on a server and the user must use a browser and input the correct URL (Universal Resource Locator) to access the website. Alternatively, if the user wants to use the application in a Local Area Network, the software can also be compiled as an executable file which will work just the way a normal desktop application would.

If the proposed system will be implemented, the school does not need to layout and print hundreds of ballots every year whenever there is an election, the canvassing will be much faster, and its output more accurate because every calculation will be done by the computer. Also, instead of having to shade ballots, votes are being done in just a few clicks. By applying the use of modern technology, the proponents believe that it will greatly ease up the school’s election process.

1. **High-level System Components**

The proposed voting system has two sides which will also define the type of its users, the Admin side and the Client side. The Admin side contains the modules that are necessary for the creation, monitoring, and management of the election(s), as well as management the records of voters and accounts of administrators. On the other hand, the Client side is the one that will be utilized by the voters. They will be able to vote, manage their password as well see the results and generate reports about the election when the canvassing is done.

* 1. ADMIN-SIDE APPLICATION

3.1.a Login Module– to access the system internally as an admin, the user must provide certain credentials which will be verified by the system if it is correct or not. If the credentials are correct, then the user will be able to gain access in the system as an admin, otherwise he will not be able to access it.

3.1.b Election Module – this is a collection of submodules that are pertinent to the Creation, Management and Closing of an Election event. The module allows the admin user to create an election, produce a list of candidates, manage and terminate ongoing election events. This module is dynamically designed in such a way that other student related elections aside from the Student Government Organization can be created in this system. During the canvassing of the votes, results are being reflected in the Monitoring Module (see 3.1.c) as statistics. After the canvassing, a tally of votes will be displayed in this module to provide a clear output of the results of the election which can be printed also by the admin user.

3.1.c Monitoring Module – this module allows the admin user to track status of the ongoing election(s) (e.g. who is in the lead right now or at a certain point of time), who is currently voting, as well as keeping and printing records of those who have voted already and who did not.

3.1.d Account Settings and Management – an admin user can change his/her user name and password, and through this module, an admin user can also create, update, and delete admin and client accounts.

3.1.e Archives – this is a repository that contains records of currently enrolled students, voter account and its passwords, vote traces, current and previous election results, and as well as login records from the admin side and the client side. The module also allows the admin user to produce a print out of the records from this module.

3.1.f Configuration Settings – because the system can also be compiled into an .exe file, the admin user may want to configure the settings of the proposed system in such a way that it will run in a LAN – based platform, hence the module.

* 1. CLIENT-SIDE APPLICATION

3.2.a Login Module – to access the system internally as a voter, the user must provide certain credentials which will be verified by the system if it is correct or not. If the credentials are correct, then the user will be able to gain access in the system as a voter, otherwise he will not be able to access it.

**3.2.b Voting Module** – This is the part of the system which enables the voter to vote for his/her desired candidates in for every position. After choosing his/her desired candidates, the system lets the voter his/her choices before finalizing it, which then counts as a vote afterwards.

* 1. **OPTIONAL FUNCTIONAL UNITS**

**3.3.a Backup Module (Admin-side)** – This module allows the admin user to create a backup of system data which is to be stored in a secure location, and input data in the system database just in case the data stored in the system gets deleted.

**3.3.b Election Monitoring (Client-side)** – with some similarities from the Monitoring Module in the Admin-side, this aims to provide the voters a real-time display of the partial results of the election.

**3.3.c Password Settings (Admin/Client-side)** – the user can change his/her password which must meet the system’s requirements for a valid password.

1. **Application Architecture**

The project is using the Enterprise Web Based Architecture as defined by the Woodger Computing Inc., which utilizes the following logical layers: Client Layer, Presentation Layer, Business Logic Layer and the Data Layer. The Client layer of a web application is implemented as a web browser running on the user's client machine.  Its job in a web-based application is to display data and let the user enter/update data. In this project, the semi-intelligent client approach is being used because there are validations and, also, it is generally easier-to-use and requires fewer communications back-and-forth from the server.

The Presentation layer generates webpages and it includes dynamic content in the webpage.  The dynamic content typically originates from a database (e.g. a list of matching products, a list of transaction conducted over the last month, etc.)  The other major job of the presentation layer is to "decode" the webpages coming back from the client (e.g. find the user-entered data and pass that information onto the business logic layer). As for the Business logic, its job is to perform all required calculations and validations, manage workflow (including keeping track of session data), and manage all data access for the presentation tier.

The Data layer is responsible for managing the data.  In the simple case, a data layer may simply be a modern relational database.  However, it may include data access procedures to other data sources like hierarchical databases, legacy flat files, etc. The job of the data layer is to provide the business logic layer with required data when needed and to store data when requested.  Generally-speaking, the architect should aim to have little or no validation/business logic in the data layer since that logic belongs in the business logic layer. However, eradicating all business logic from the data tier is not always the best approach. Not null constraints and foreign key constraints can be considered "business rules" which should only be known to the business logic layer. Most would agree that it is safer/better to include such simple constraints in the database (and to change them, as the business rules evolve).

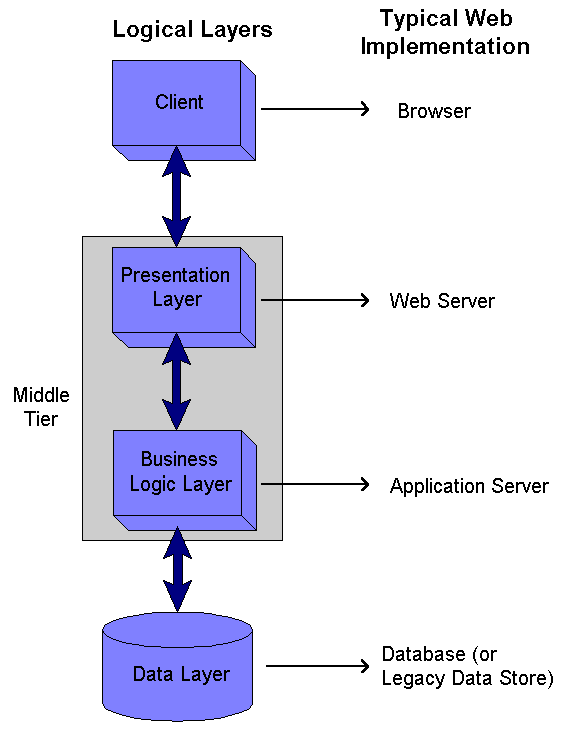


Figure 4.1 Architecture for an Enterprise Web Based Application

1. **Hardware and Software Specifications**

|  |  |
| --- | --- |
| **Recommended System Specifications** | |
| Operating System | Windows 7 or higher |
| Processor | 1 GHz or faster |
| RAM | At least 1 GB |
| Browser | Google Chrome or Mozilla Firefox (latest version) |
| PHP | 7.2 or higher |
| XAMPP | v3.2.2 |

1. **Tools and Technologies Used**

**LANGUAGES/FRAMEWORKS:**

* **FRONT-END**
* HTML and CSS for the visual design/layout
* Bootstrap (front-end framework)
* jQuery for the JavaScript library (Front-end programming language)
* **BACK END**
* PHP 7.2 for server-side programming language
* Laravel 5.6 (PHP framework)
* SQL for the database

**OTHER TOOLS AND TECHNOLOGIES**

* Heroku App for web hosting
* Git for Version Control System
* Adobe Photoshop CS5 for graphic designing
* Visual Studio Code for IDE
* PhpMyAdmin/ MySQL for SQL Database
* PeachPie for the conversion of PHP script to .exe file