ABSTRACT

The ‘Online Voting System’ is the study and implementation of the voting requirements and protocols during an election. It is an online voting system aimed at easing the voting procedure and reducing the election’s time duration. The system is a web application where a person can cast vote after the verification of their personal information and can view the results of the ongoing election. The application is built on the traditional Waterfall Model. The application is built using HTML, CSS and JS for the UI/UX and Python and Django framework for the server-side scripting. The application employs MySQL for storing and retrieving data. A voter, after verification, can view the candidates running for the election along with their personal information, qualification and work done by them. The web application provides simple and intuitive user interface without ambiguity for easy and errorless voting. It can make the election conclude quickly and the money spent to provide wages to vote counters can be saved and used for something more productive.

***Keywords: voting, verification, waterfall, framework, server-side, web application, MySQL***

ACKNOWLEDGEMENT

We would like to express our profound gratitude and apreciation to the Head of Department of BCA and our teacher of Software Enginnering, Ram Babu Mahato, for his invaluable patience and feedback. This endeavor wouldn’t have been possible without him. We would also like to express our deepest appreciation to various Youtubers dedicating their channel to Python tutorials.

We would like to extend our sincere thanks to our colleagues for their impact, inspiration and moral support.

TABLE OF CONTENT

[ABSTRACT i](#_Toc159530892)

[ACKNOWLEDGEMENT ii](#_Toc159530893)

[TABLE OF CONTENT iii](#_Toc159530894)

[LIST OF ABBREVIATIONS iv](#_Toc159530895)

[1. INTRODUCTION 5](#_Toc159530896)

[2. PROBLEM STATEMENT 5](#_Toc159530897)

[3. OBJECTIVES 2](#_Toc159530898)

[4. METHODOLOGY 2](#_Toc159530899)

[4.1 REQUIREMENT IDENTIFICATION 2](#_Toc159530900)

[4.1.1 LITERATURE REVIEW 2](#_Toc159530901)

[4.1.2 REQUIREMENT ANALYSIS 2](#_Toc159530902)

[4.2 FEASIBILTY STUDY 3](#_Toc159530903)

[4.2.1 TECHNICAL FEASIBILITY 3](#_Toc159530904)

[4.2.2 OPERATIONAL FEASIBILITY 3](#_Toc159530905)

[4.2.3 ECONOMIC FEASIBILITY 3](#_Toc159530906)

[4.2.4 SCHEDULE FEASIBILITY 4](#_Toc159530907)

[4.3 SYSTEM DESIGN 4](#_Toc159530908)

[4.4 USE CASE DIAGRAM 5](#_Toc159530909)

LIST OF ABBREVIATIONS

**BCA** Bachelors in Computer Administration

**CSS** Cascading Style Sheet

**HTML** Hypertext Markup Language

**IDE** Integrated Development Environment

**JS** Javascript

**NPR** Nepalese Rupee

**OVS** Online Voting System

**PC** Personal Computer

**SQL** Structured Query Language

**UI** User Interface

**USD** United States Dollar

**UX** User eXperience

1. INTRODUCTION

The ‘Online Voting System’ implementation of the voting requirements and protocols during an election. It is an online voting system aimed at easing the voting procedure and reducing the election’s time duration.

The system is a web application where a person can cast vote after the verification of their personal information and can view the results of the ongoing election. The web application provides simple and intuitive user interface without ambiguity for easy and errorless voting. It can make the election conclude quickly and the money spent to provide wages to vote counters can be saved and used for something more productive.

Online voting system promotes e-governance and brings transparency in the election. This completely mitigates counting errors and theft of ballot boxes during the physical election – which happens often in Nepal during Elections. Along with benefits, it also brings forth problems. For instance, implementation and maintenance cost can be huge, prone to security attacks like hacking, etc. The solutions to these problems are following proper security regulations while building the system and hiring a cyber-security specialist for checking and improving the vulnerabilities. The system can be made short term i.e. making the voting system online only during the election period which can cut the maintenance cost.

1. PROBLEM STATEMENT

Election is the fundamental part of democracy which promotes the democratic decision making. However, Elections are the major financial burden for a nation like Nepal – where local and parliamentary election are conducted every few years. For instance, the local election of 2022 in Nepal brought forth the whopping expenditure of NPR 8.11 billion (USD 63.7 million) for Election Commission which is the highest till date. In essence, every few year millions of dollars are spent during election and the amount rising. Another problem with elections in Nepal is robbery of the ballot boxes which has happened during every election.

OVS aims at resolving these problem to great extent. Implementation of OVS will substantially reduce the cost and time for conducting election. OVS is not the foolproof solution to reduce election expenditure as its implementation and maintenance requires funding and politicians are going to spend above set limits, however, not having to pay wages to the counters, for as long as the day they take to count the votes, can save the amount from the set budget. It is also quite useful in case of the re-elections.

1. OBJECTIVES

* To reduce the time it takes from casting the vote to electing a candidate.
* To reduce the cost of conducting election.
* To improve e-governance index.
* To stop the robbery of ballot boxes.

1. METHODOLOGY

## 4.1 REQUIREMENT IDENTIFICATION

### 4.1.1 LITERATURE REVIEW

### 4.1.2 REQUIREMENT ANALYSIS

Requirement analysis is further divided into two categories: Functional and Non-functional requirement.

1. Functional Requirement:
2. Citizen Information Form: The system will have a form where a citizen can fill up their information, such as voter id number, name, age, etc.
3. Candidate Information: Voters can view the information about the candidates running in the election along with their past work history.
4. Vote Count: Voters can also see the leading candidates based on the vote count.
5. Non-functional Requirement:
6. User Verification: System can verify the eligibility of a voter based on the information provided in the citizen information form.
7. Accident Control: System prompts a pop-up when submitting vote to mitigate accidental votes.
8. Validation: System can validate the inputs provided by the user.
9. Security: System is secured with safety protocols and encryption strategies to protect the user’s information and itself.

## 4.2 FEASIBILTY STUDY

The purpose of this feasibility study is to investigate the practicality of the project and identify potential problems that might arise during the development and/or deployment after taking in all the factors into account. This chapter contains Technical, Operational, Economical and Schedule Feasibility.

### 4.2.1 TECHNICAL FEASIBILITY

This web application is built using HTML, CSS, JS for frontend scripting and Tailwind CSS library for better design whereas Python & Django for backend scripting. The project also requires an IDE like Visual Studio Code to run all these programming languages and libraries. All these tools are free of cost and easy to use. Hence, this project is technically feasible.

### 4.2.2 OPERATIONAL FEASIBILITY

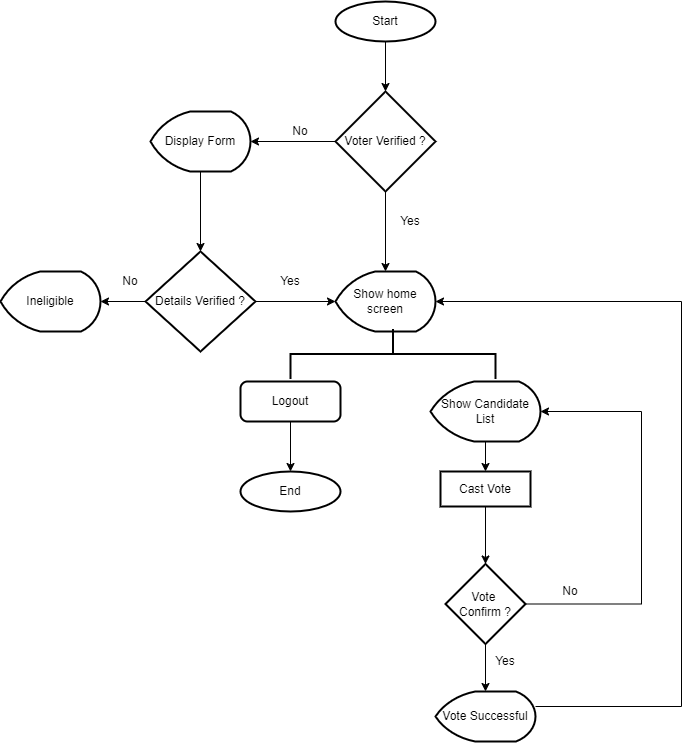
Web application is equipped with helpful resources and hints to avoid ambiguity and eliminate errors. Interface is very user friendly that anyone can use it with ease. Therefore, this project passes the operational feasibility.

### 4.2.3 ECONOMIC FEASIBILITY

The only requirement of the project are developers with PC capable of running the IDE such as Visual Studio Code which is completely free. Since all developers have the computers with good specs and tools and software used are completely free. Only expenses are towards designers and developers, which is not a problem since we are the ones who are fulfilling all the roles. So, the project is economically feasible.

### 4.2.4 SCHEDULE FEASIBILITY

## 4.3 SYSTEM DESIGN



## 4.4 USE CASE DIAGRAM

