Software Design and Analysis Document Report For

E-Voting System

I21-0433 Nida Azam

I21-1520 Arsal Miraj

I20-0461 Basit Rauf

Startex

26 November,2024

Table of Content

1. Introduction

1.1 Purpose

The purpose of the E-Voting system is to make the voting process more robust, secure, and hassle-free. Arranging physical voting systems can be challenging due to logistical complexities, high costs, and time-consuming procedures. Additionally, traditional voting methods are prone to issues such as vote stealing, tampering, and inaccurate results. This e-voting system addresses these challenges by providing a streamlined, digital solution that ensures security, accuracy, and convenience for both voters and election administrators.

1.2 Product scope

The system is not designed to handle nationwide elections but is well-suited for conducting votes at an inter-regional level. It provides essential functionalities such as voter registration, vote casting, withdrawal, and management of voters and candidates. Additionally, the system includes robust features for tallying votes accurately and generating comprehensive reports, ensuring transparency and reliability. These capabilities make it a practical and secure solution for small-scale elections.

1.3 Objective

The objective of the E-Voting System is to provide a secure and reliable platform for inter-regional elections, ensuring the integrity of votes and preventing tampering or inaccuracies. It aims to streamline election processes through efficient voter registration, vote casting, candidate management, accurate vote tallying, and comprehensive report generation, making elections more transparent, accessible, and hassle-free.

1.4 Problem Statement

Traditional voting methods face challenges such as logistical complexities, high costs, vote tampering, and inaccurate results. These issues highlight the need for a secure and efficient solution that simplifies the voting process, ensures accuracy, and enhances trust in election outcomes.

2. Requirements

2.1 Functional Requirements

The E-Voting System includes key functional requirements such as **User Management** for registering and managing voters and candidates, and **Election Setup** for configuring election details. It allows users to **cast votes**, **withdraw** votes if necessary, and ensures **vote tallying** is accurate and transparent. Additionally, the system generates **reports** to provide clear insights into election results, ensuring accountability and ease of access to important data.

2.2 Non-Functional Requirements

The E-Voting System must ensure **Performance** by providing fast, responsive interactions even under heavy loads. **Scalability** is essential, allowing the system to grow and handle increasing voter numbers or multiple elections without compromising efficiency. **Security** is a priority to safeguard voter data and prevent tampering, while **Availability** ensures the system is accessible at all times with minimal downtime. The system must also adhere to relevant **Compliance** standards and provide an intuitive, user-friendly interface for ease of use and better **Usability** for both voters and administrators.
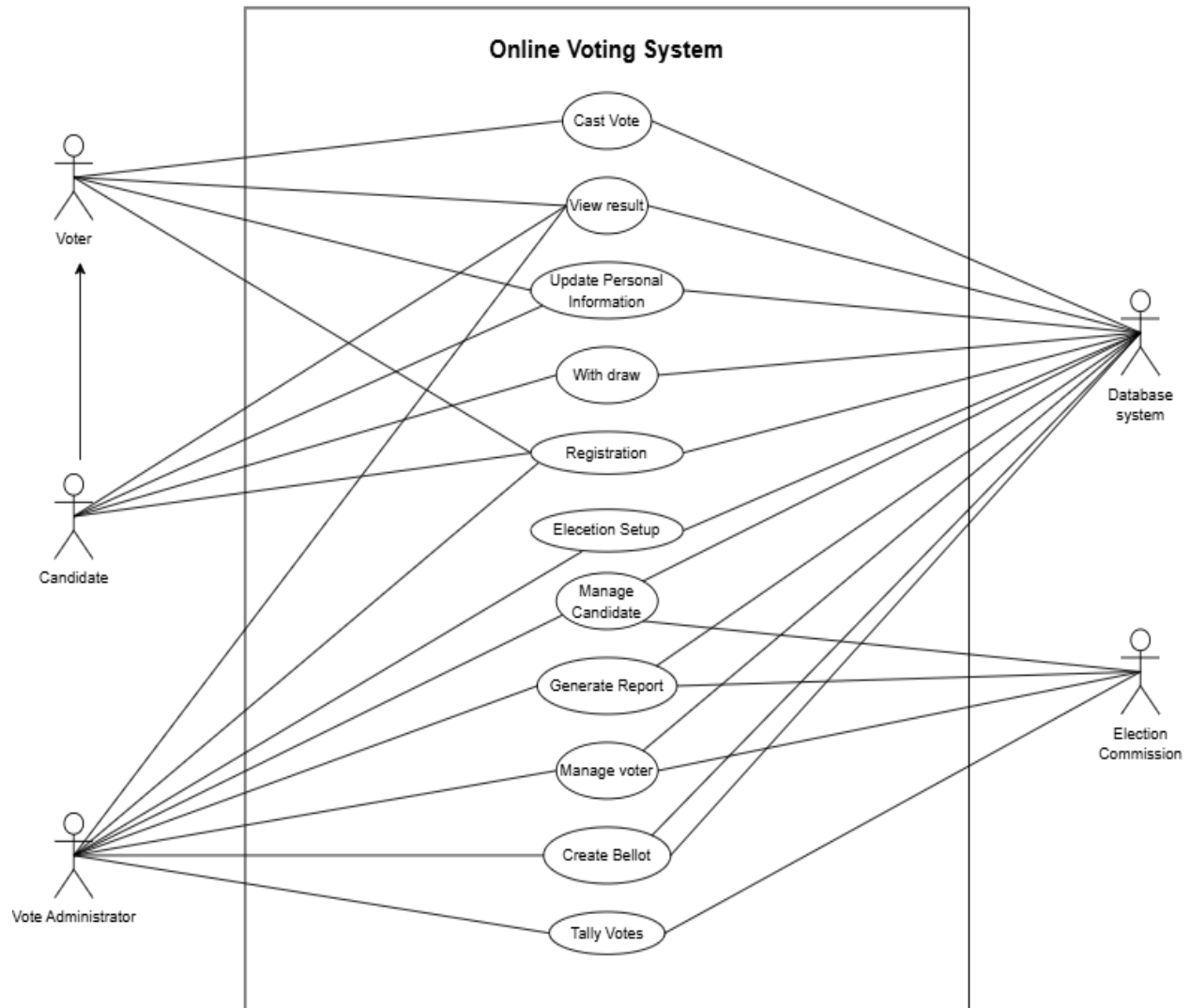
3. System Design Model
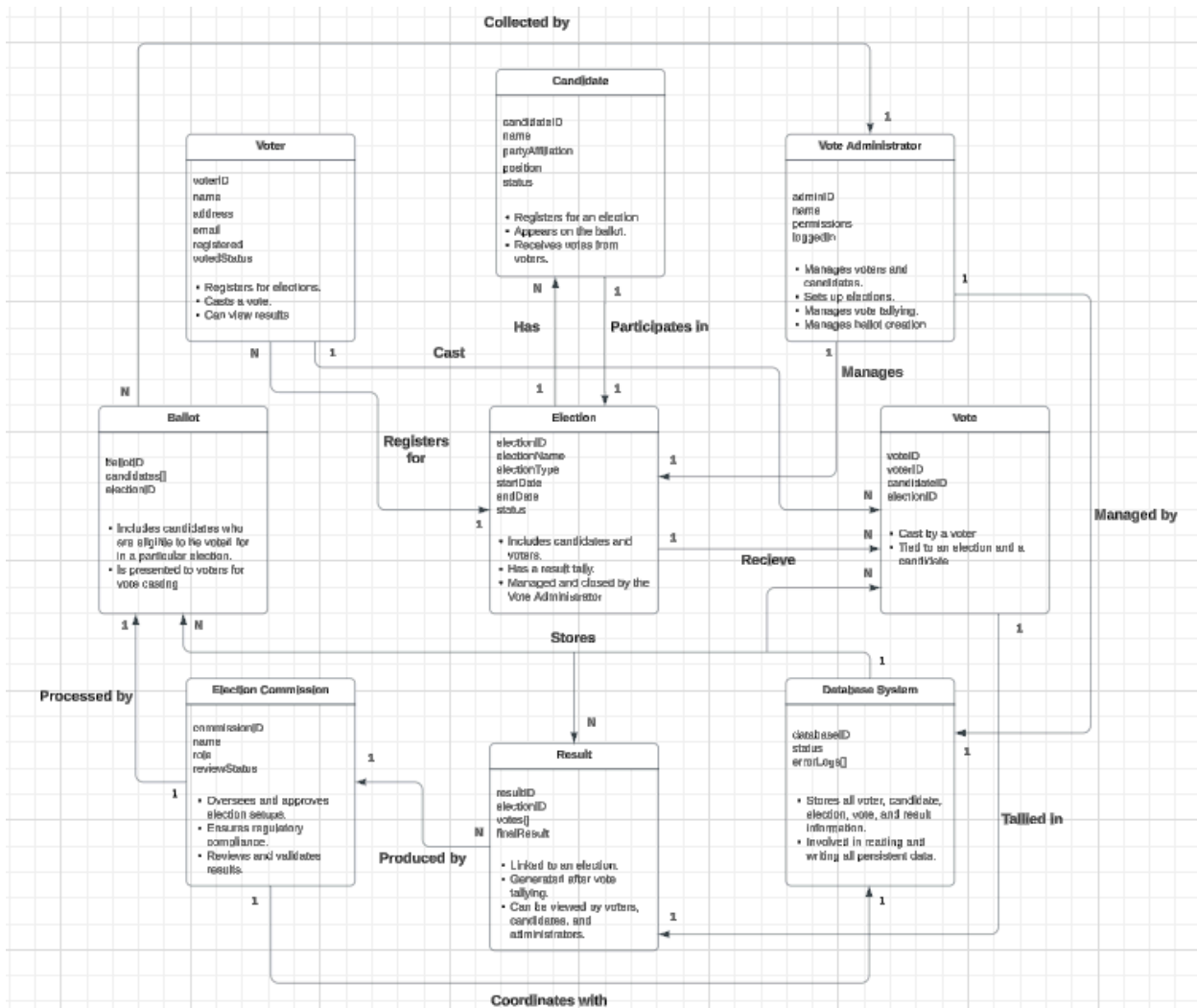
1.1 Use cases

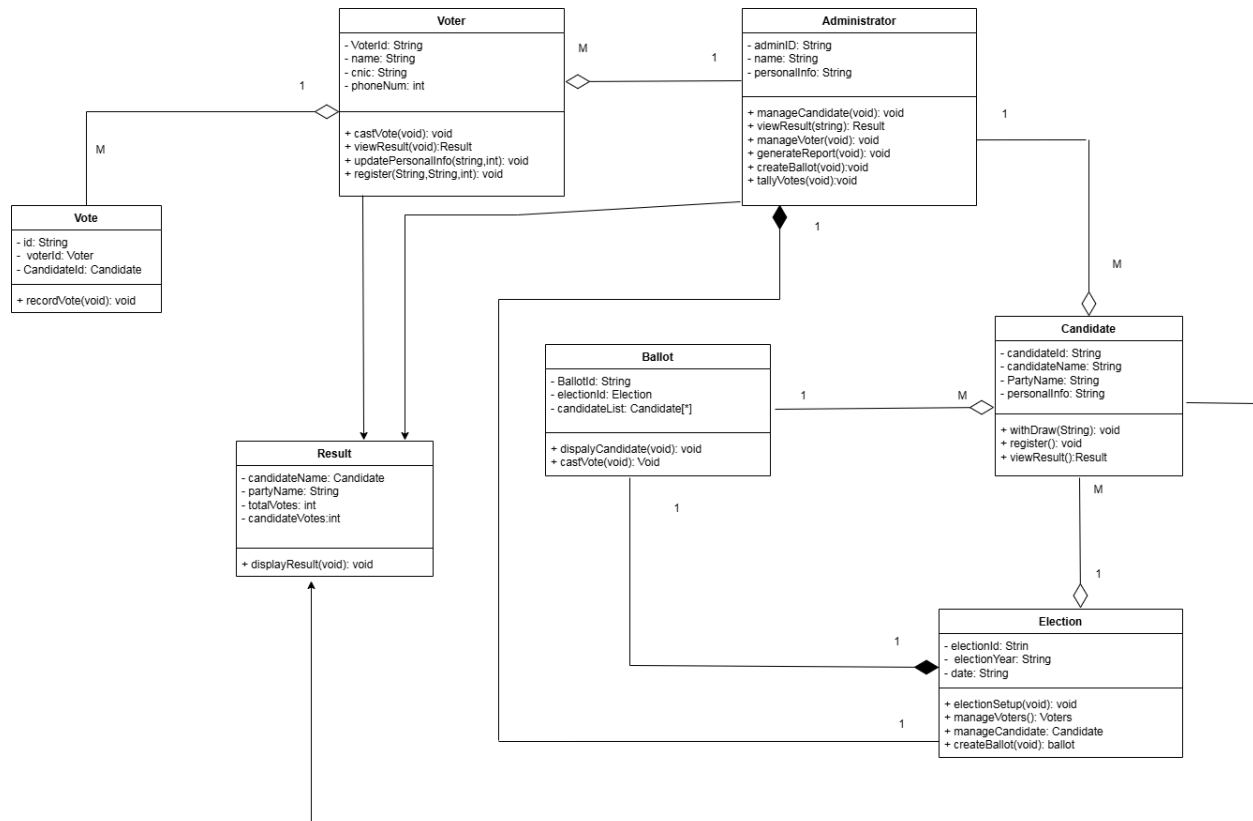**Group Members**

21i-0433 Nida Azam

21i-1520 Arsal Mairaj

20i-0461 Basit Rauf

1.2 Domain Model

1.3 Class Diagram

Voter

- VoterId: String
- name: String
- cnic: String
- phoneNum: int

+ castVote(void): void
+ viewResult(void):Result
+ updatePersonalInfo(string,int): void
+ register(String,String,int): void

Administrator

- adminID: String
- name: String
- personalInfo: String

+ manageCandidate(void): void
+ viewResult(string): Result
+ manageVoter(void): void
+ generateReport(void): void
+ createBallot(void):void
+ tallyVotes(void):void

Vote

- id: String
- voterId: Voter
- CandidateId: Candidate

+ recordVote(void): void

Ballot

- BallotId: String
- electionId: Election
- candidateList: Candidate[*]

+ dispalyCandidate(void): void
+ castVote(void): Void

Candidate

- candidateId: String
- candidateName: String
- PartyName: String
- personalInfo: String

+ withDraw(String): void
+ register(): void
+ viewResult():Result

Result

- candidateName: Candidate
- partyName: String
- totalVotes: int
- candidateVotes:int

+ displayResult(void): void

Election

- electionId: Strin
- electionYear: String
- date: String

+ electionSetup(void): void
+ manageVoters(): Voters
+ manageCandidate: Candidate
+ createBallot(void): ballot
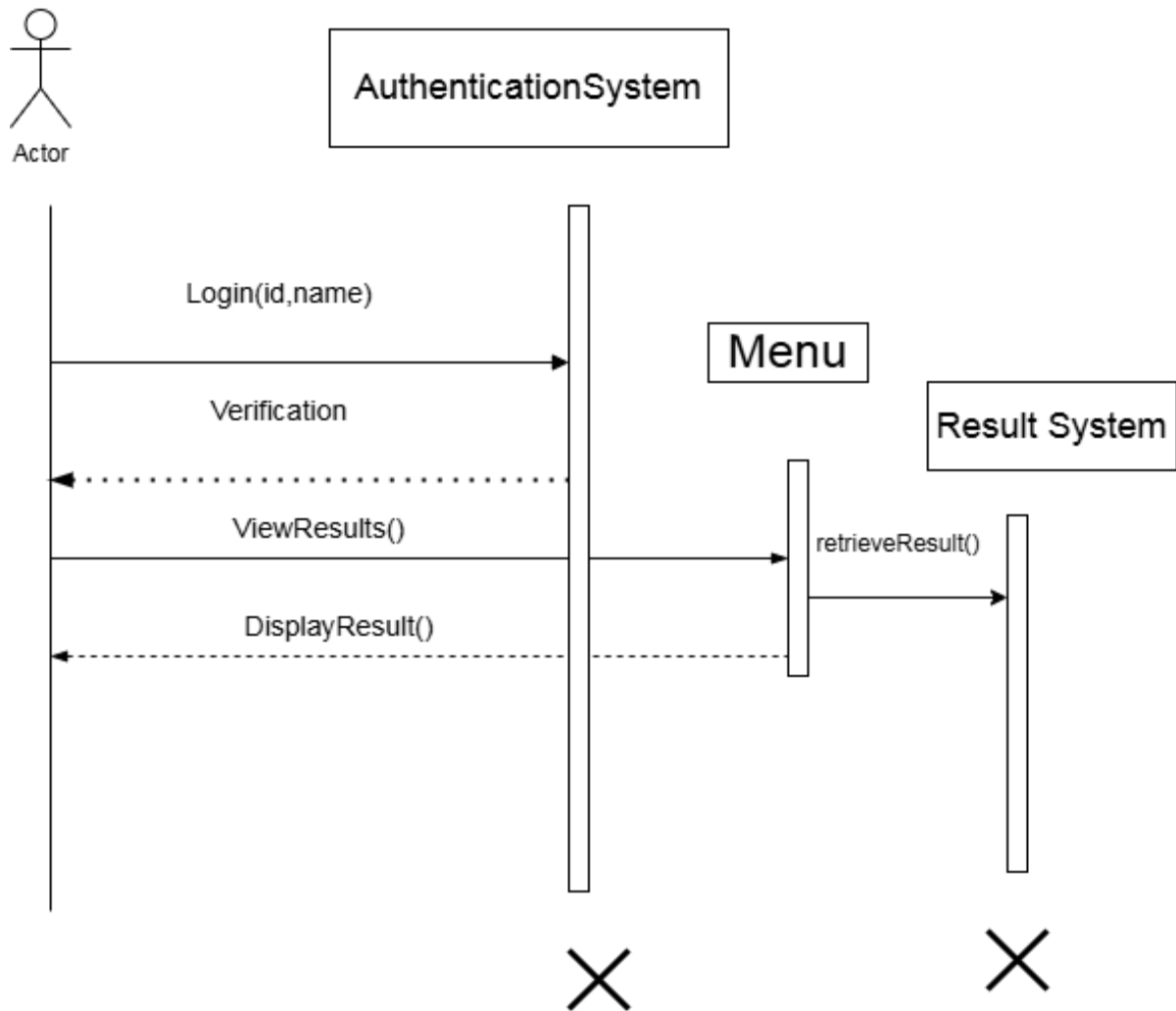
M   1

1

M

1

M

1

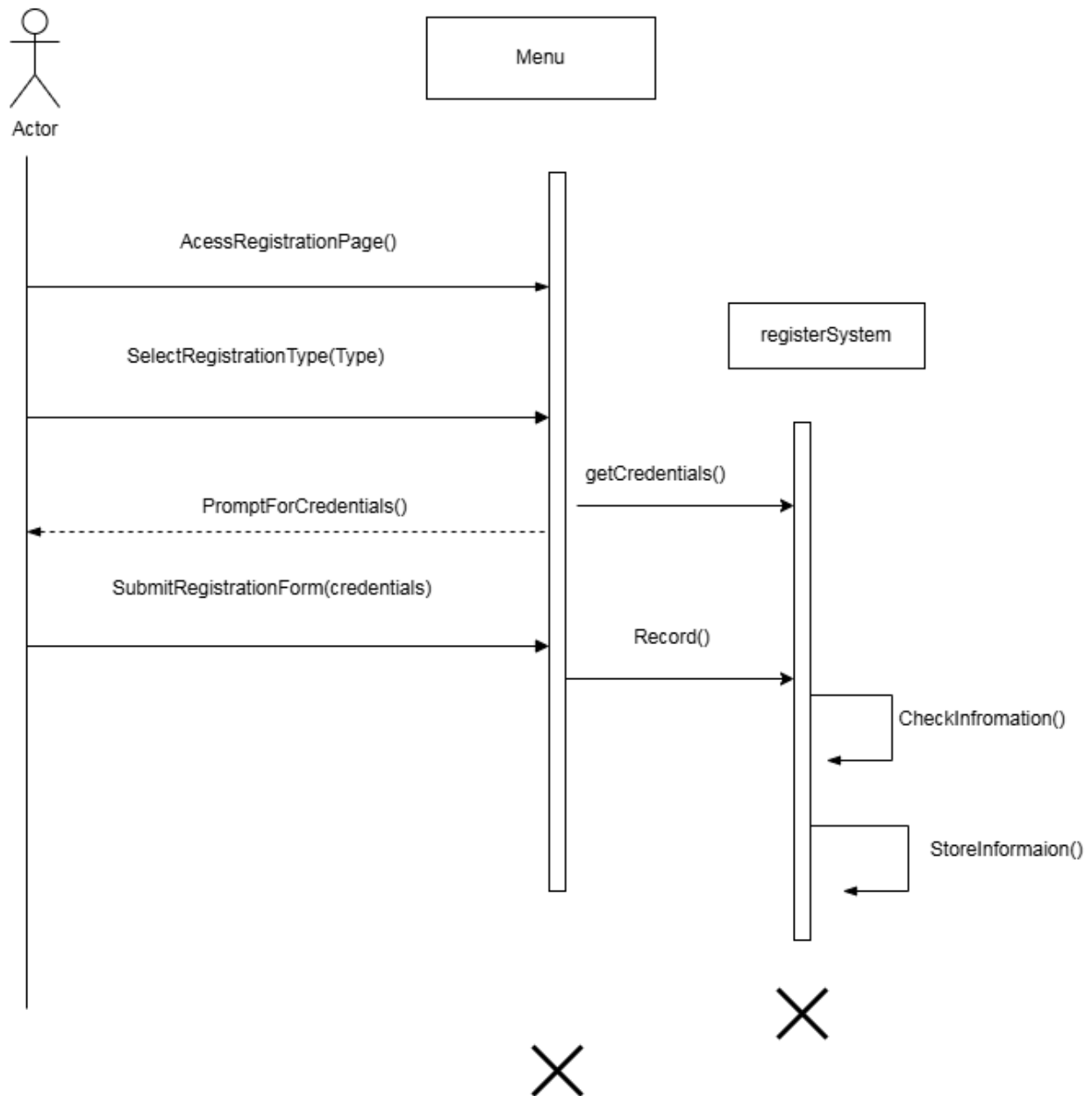M

1

1

M

1

M

1

1

1

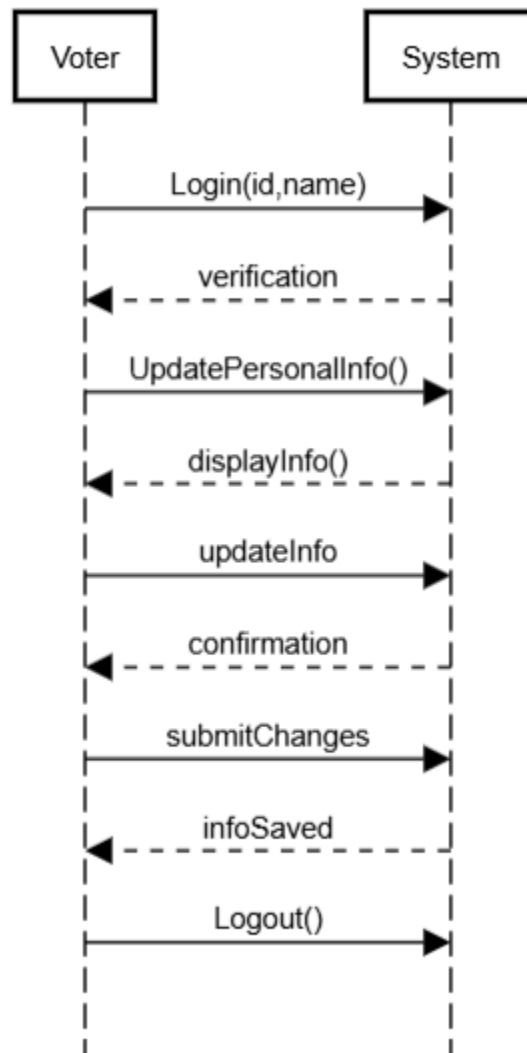## 1.4 System Sequence Diagram
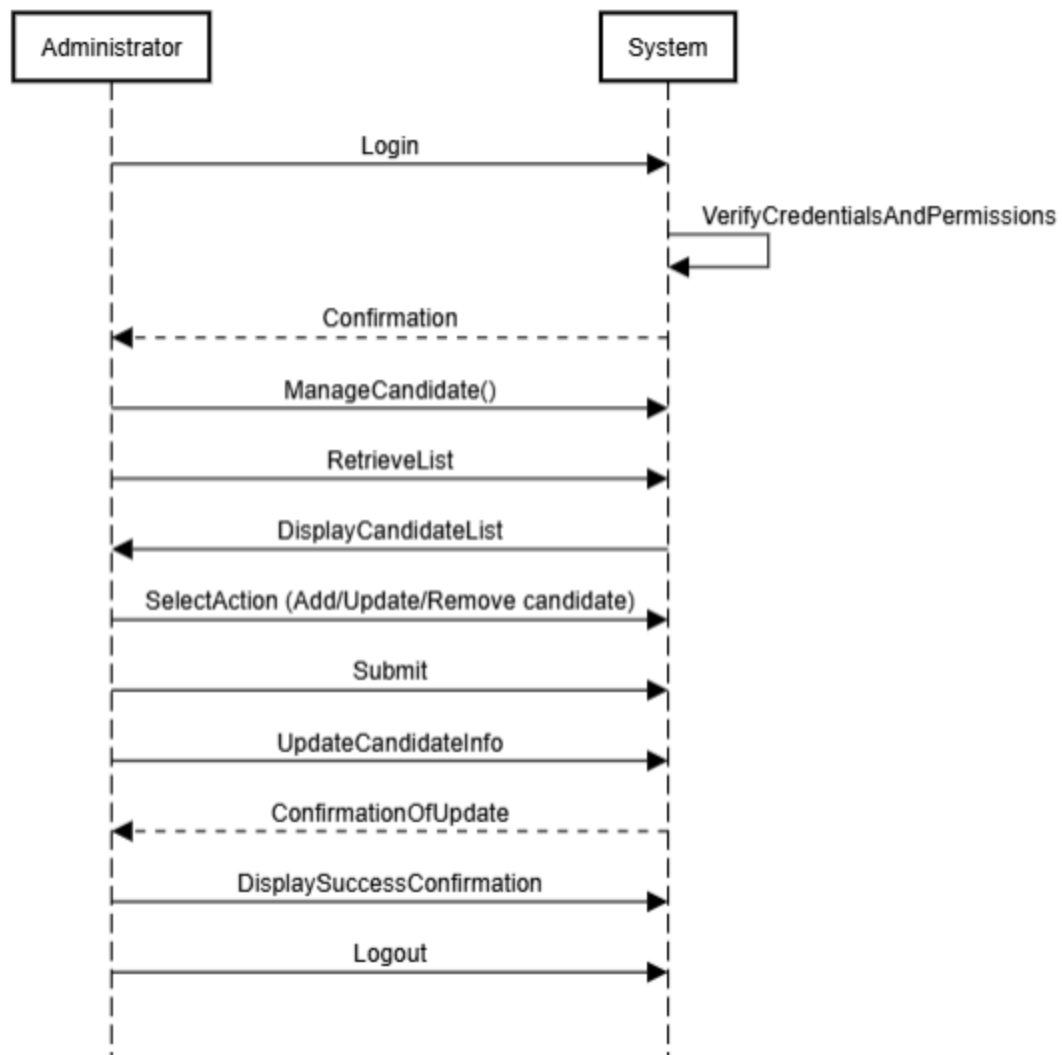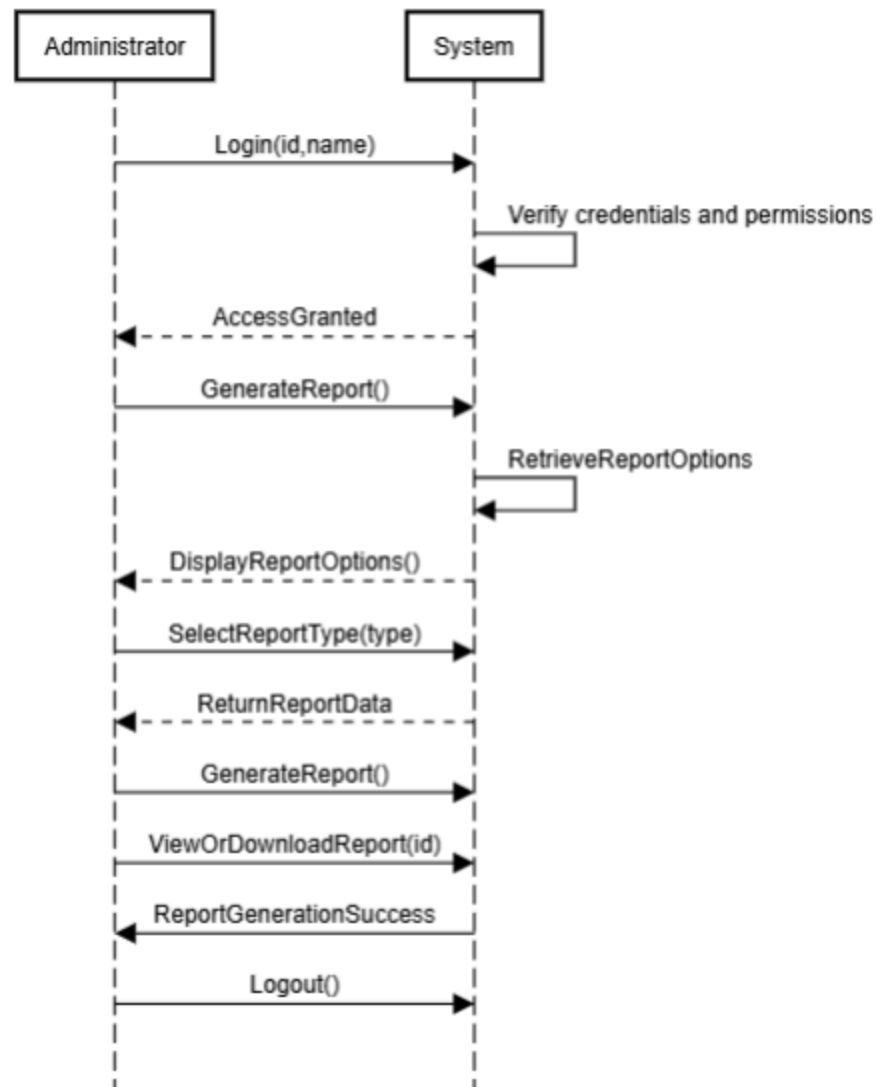

Cast Vote
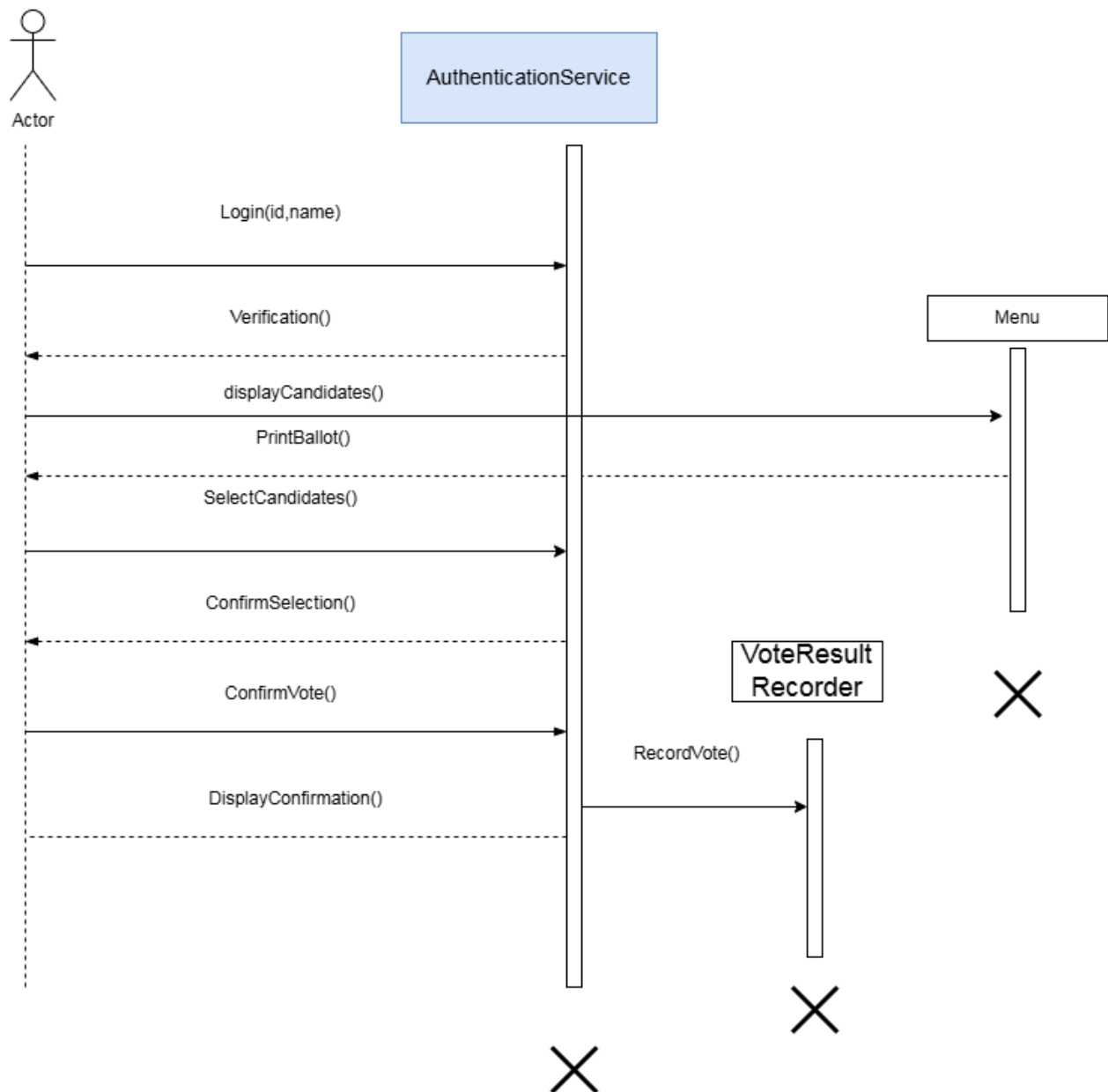
View Result

Registration

Update Personal Information

Manage Candidates

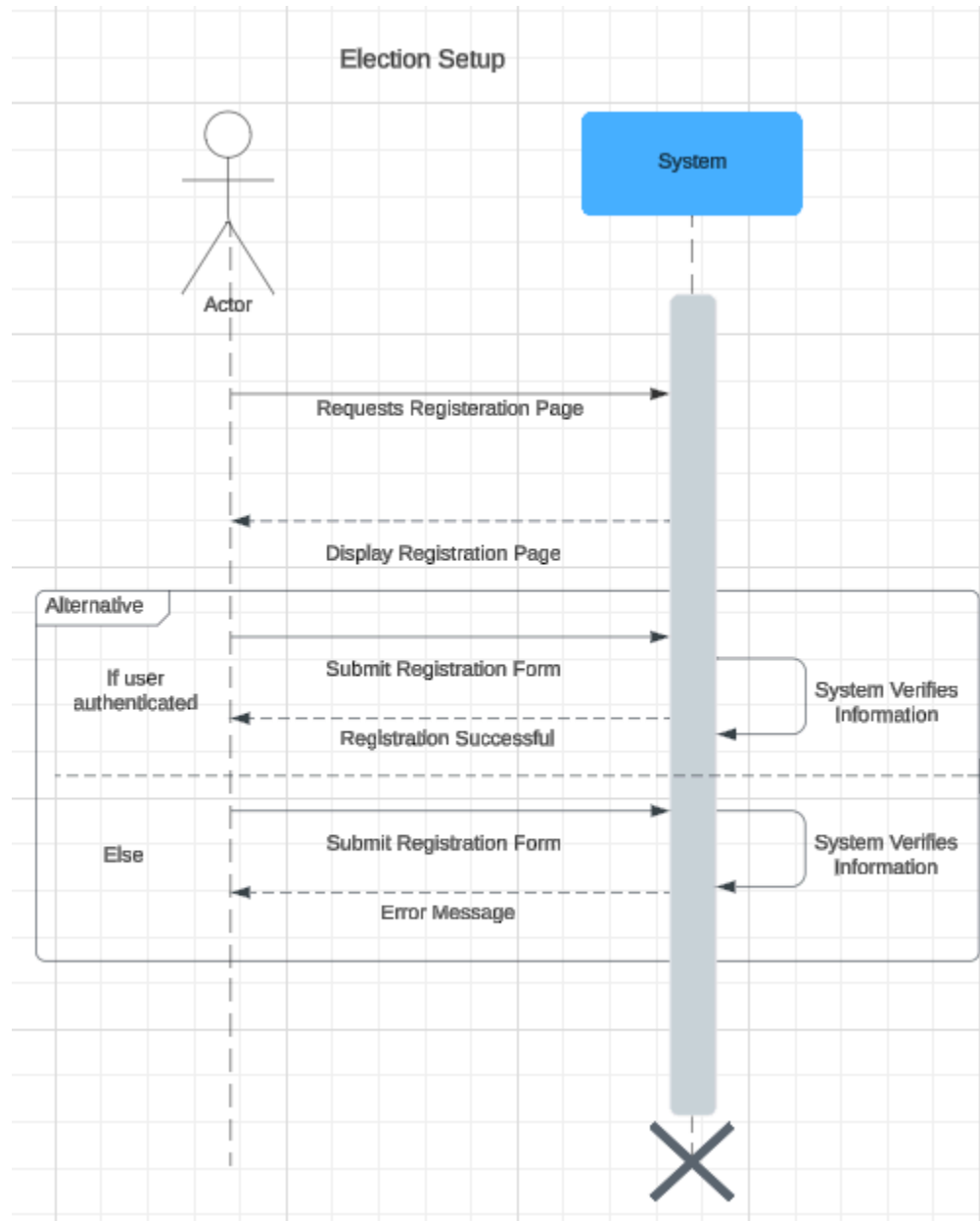Generate Report

Create Ballot

Election Setup

Election Setup

Actor

System

Requests Registeration Page

Display Registration Page

Alternative

If user authenticated

Submit Registration Form

System Verifies Information

Registration Successful

Else

Submit Registration Form

System Verifies Information

Error Message

Manage Voter

Manage Voter

System

Administraor

Selects Manage Voter

Displays Manage Options

Selects Managing Option

Enter Required Information

Alternative

Submits Required Information

System Validates Information

If information is valid

Information Valid

Else

Submits Required Information

System Validates Information

Promts Admin to Enter Correct Information

Admin Confirms

Displays Success Message

Tally Votes

Tally Votes

System

Administraor

Selects Tally Votes

Presents a List of Available Elections

Chooses The Specific Election

Retrieves all Votes For The Selected Election

Confirms Vote Tallying

System processes the votes

Generates Tally Report

Reviews Report

Displays Final Decision

Confirms The Report Results

System stores the results

Marks Elections "Closed"