
Software Requirements and Design Document

for E-Voting System

Prepared by

Arrij Fawwad i220755

Hamna Arshad i221098

Maryam Masood i221169

Section-C

Fast Nukes, Islamabad

Nov 23, 2024

Table of Contents

1. Introduction.....	1
1.1 Purpose.....	1
1.2 Product Scope.....	1
1.3 Title.....	1
1.4 Objectives.....	2
1.5 Problem Statement.....	2
2. Overall Description.....	3
2.1 Product Perspective.....	3
2.2 Product Functions.....	3
2.3 List of Use Cases.....	5
Use Case.....	5
2.4 Extended Use Cases.....	6
Use Case 01 - Register Candidate.....	6
Use Case 02 - Manage electoral polling staff accounts.....	7
Use Case 03 - Initiate system.....	8
Use Case 04 - View polling staff Assignments at the Polling Station.....	9
Use Case 05 - Authorize polling staff Entry.....	10
Use Case 06 - Cast Vote.....	11
Use Case 07 - Monitor active systems.....	12
Use Case 08 - Inspect Audit logs.....	13
Use Case 09 - Capture Voter information.....	14
Use Case 10 - Issue election Report.....	15
Use Case 11 - Access Election Forms.....	16
Use Case 12 - Display Election Results.....	17
2.5 Use Case Diagram.....	18
3. Other Nonfunctional Requirements.....	19
3.1 Performance Requirements.....	19
3.2 Safety Requirements.....	19
3.3 Security Requirements.....	19
3.4 Software Quality Attributes.....	20
3.5 Business Rules.....	20
3.6 Operating Environment.....	21
3.7 User Interfaces.....	22
Main Page:.....	22
The layout ensures equal spacing and alignment of elements.....	23
About Page::.....	24
Capture Voter Info:.....	25
1. Interface Overview.....	26
2. Design Standards.....	26

3. Error Handling.....	26
4. Workflow Details.....	27
Polling Station Info:.....	27
1. Interface Overview.....	27
2. Design Standards.....	27
3. Data Handling.....	28
4. Error Handling.....	28
5. Workflow Details.....	28
Cast Vote:.....	29
1. Interface Overview.....	29
Voter Information Display:.....	29
Dynamic Table:.....	29
Voting Action:.....	30
2. Design Standards.....	30
Styling:.....	30
Responsiveness:.....	30
Accessibility:.....	30
3. Data Handling.....	30
Dynamic Row Creation:.....	30
Interactive Elements:.....	30
Vote Submission:.....	30
4. Error Handling.....	30
Validation:.....	30
Null Data Safeguards:.....	30
Backend Notification:.....	31
5. Workflow Details.....	31
Initialization:.....	31
Data Update:.....	31
Vote Casting:.....	31
View Logs:.....	32
Monitor Polling Systems:.....	33
2. Interface Overview.....	33
3. Design Standards.....	34
4. Error Handling.....	34
5. Table Structure.....	34
Admin Menu:.....	34
1. General Layout and Design:.....	35
2. GUI Standards and Components:.....	35
3. Navigation and User Flow:.....	35
4. Error Message Display Standards:.....	35
5. GUI Components Needing a User Interface:.....	36

Add Candidate Interface:.....	36
1. Common UI Elements:.....	36
2. Workflow for Admin Interface (Add Candidate):.....	37
3. Screen Layout and Design Constraints:.....	37
4. Error and Success Handling:.....	37
Add Polling Staff Interface:.....	38
General UI Characteristics:.....	38
Common UI Components:.....	38
Software Components with User Interface:.....	39
Error Message Standards:.....	39
Candidate List Interface:.....	39
Components & Design:.....	40
Error & Feedback:.....	40
Standard Features:.....	40
Sample Screen (Candidate List View):.....	40
Staff Assignment Interface:.....	41
Update/Deactivate Staff Interface:.....	42
Election Forms Interface:.....	43
Election Results Interface:.....	45
Election Area Report Interface:.....	48
4. Domain Model.....	51
5. System Sequence Diagram.....	52
6. Sequence Diagrams.....	64
Capture Voter Info.....	64
Authorize Polling Staff.....	65
Cast Vote.....	66
View Logs.....	67
Monitor Systems.....	68
Add Candidate.....	69
Manage Staff Account.....	70
View Polling Staff Assignments.....	73
Initiate System.....	74
View Election Report.....	75
View Election Form.....	76
View Election Result.....	77
7. Class Diagram.....	77
8. Component Diagram.....	78
9. Package Diagram.....	79
10. Deployment Diagram.....	80

1. Introduction

1.1 Purpose

The Votix e-voting system is designed to transform traditional voting methods by providing a secure, efficient, and user-friendly platform for conducting elections. The software specification outlined here defines the core functionalities, security mechanisms, and real-time monitoring capabilities of Votix, which aims to eliminate manual inefficiencies and enhance transparency in the electoral process.

This document serves as a blueprint for the development and deployment of Votix, focusing on key components such as polling PC operations, admin monitoring through real-time logs, and secure data synchronization via a VPN-based MySQL database. By integrating SQL injection prevention techniques, the system ensures the highest level of security for sensitive election data.

1.2 Product Scope

Votix is a comprehensive digital voting platform that addresses inefficiencies in manual election processes through automation, security, and real-time monitoring. The system's core features include:

- **Real-Time Logs with SSE:** Live updates on polling PC activity, such as votes cast, authentication attempts, and system performance, streamed directly to election administrators.
- **Data Security:** VPN-based encrypted data transfer and centralized database management ensure the integrity and protection of sensitive election data.
- **Transparency and Accountability:** Tools for monitoring activities across all polling stations in real-time ensure complete oversight.

Key Benefits:

- Reduced operational costs and faster election workflows through digitization.
- Enhanced voter trust with tamper-proof data and transparent processes.
- Improved oversight with SSE-driven live updates, reducing response time to anomalies.

1.3 Title

Votix: A Secure, Transparent, and Modern E-Voting Solution with Real-Time Monitoring

1.4 Objectives

The primary objectives of Votix are:

- **Efficiency:** Automate voter authentication, vote casting, and result compilation to minimize manual workflows.
- **Security:** Leverage encryption, VPN connectivity, and centralized database solutions to ensure robust data protection.
- **Transparency:** Enable real-time logs and monitoring of polling PCs via SSE to provide administrators with live operational insights.
- **Usability:** Design an intuitive interface that is accessible and easy to use for all stakeholders.

1.5 Problem Statement

Challenges in Traditional Voting:

Manual voting systems face significant challenges, including:

- Operational inefficiencies lead to delays and increased costs.
- Inaccuracies in result compilation and susceptibility to fraud.
- Limited visibility for administrators in monitoring polling stations during elections.

Votix as the Solution:

Votix resolves these issues by introducing:

- **Automation:** Replacing manual processes with digital workflows, including voter verification and vote casting.
- **Security:** Employing **VPN-based networking** and robust encryption to safeguard sensitive election data.
- **Real-Time Monitoring:** Utilizing **SSE** to stream live logs and updates from polling PCs to administrators, ensuring instant visibility into system activities.

Feasibility:

Votix capitalizes on existing technologies such as JavaFX for a user-friendly interface, MySQL for database management, Tailscale VPN for secure networking, and SSE for real-time event streaming. This integration ensures a reliable and scalable solution that addresses both current and future needs of the electoral process.

2. Overall Description

2.1 Product Perspective

The Votix e-voting system addresses the challenges of traditional election methods still prevalent worldwide. Many countries have adopted e-voting systems to enhance election security and efficiency. For example, **Estonia** offers i-voting with secure digital signatures, while **Switzerland** uses e-voting in select cantons. **India** employs Electronic Voting Machines (EVMs) with paper audit trails. However, these systems often face challenges such as limited accessibility, security concerns, and a lack of real-time monitoring.

In contrast, **Pakistan** currently lacks a nationwide e-voting system, relying instead on manual voting processes that are inefficient, prone to fraud, and slow. This has led to public distrust and challenges in managing large-scale elections.

Votix offers a solution by providing a secure, transparent, and efficient platform for electronic voting. Unlike existing systems, Votix features:

- **End-to-End Security:** Unlike some systems that rely on basic encryption, Votix uses a *VPN-based network* and real-time data synchronization to ensure data integrity and protect against cyber threats.
- **Real-Time Monitoring with SSE:** Votix introduces *Server-Sent Events (SSE)* to provide election administrators with live updates and logs from polling stations, ensuring full transparency and accountability during the voting process.
- **Centralized Database with MySQL:** By centralizing the election data in a secure MySQL database hosted over a VPN, Votix ensures consistency and prevents data tampering or loss.
- **User-Friendly Interface:** Votix's design focuses on ease of use, allowing both voters and administrators to interact with the system intuitively, ensuring wide adoption and usability.

This combination of security, real-time monitoring, and centralized management positions Votix as a unique and efficient solution for modern elections, especially in countries like Pakistan, where the need for a secure, transparent, and scalable e-voting system is urgent.

2.2 Product Functions

The primary functions of the Votix e-voting system are:

- **Voter Authentication:** Securely verify voter identities through digital authentication mechanisms before voting.
- **Vote Casting:** Allow voters to cast their votes in a user-friendly interface, with secure transmission to the central system.
- **Vote Counting and Result Compilation:** Automatically tabulate votes and compile results efficiently and accurately.

- **Real-Time Monitoring:** Enable election administrators to track the status of polling stations and voting progress via real-time logs and updates.
- **Polling PC Activity Logging:** Capture and transmit logs of polling PC activities for live monitoring, using SSE for efficient data streaming.
- **Data Security:** Ensure end-to-end encryption of vote data and secure communication via a VPN, protecting sensitive election information.
- **Admin Interface:** Provide election administrators with tools for monitoring and managing the election process, including real-time access to logs and activity data from polling stations.
- **Centralized Database Management:** Use MySQL database for secure, centralized data storage, ensuring consistency and easy access for election administrators.

2.3 List of Use Cases

Sr.	Use Case
1	Register Candidate
2	Manage Polling Staff Account
3	Initiate System
4	Authorize Polling Staff
5	View Polling Staff Assignment
6	Capture Voter Information
7	Cast Vote
8	Monitor Systems
9	Inspect Logs
10	Issue Election Report
11	Access Election Forms
12	Display Election Results

2.4 Extended Use Cases

Use Case 01 - Register Candidate

Use Case Name	Register Candidate	
Scope	E-Voting System	
Level	User Goal	
Primary Actor	Administrator	
Stakeholders and Interests	Administrator: Ensures candidate eligibility. Election Commission: Validates compliance with election laws. Candidate: Wants to participate in the election.	
Preconditions	The candidate must adhere to all constitutional requirements and criterias.	
PostConditions	The candidate is successfully registered for the General Elections. The system records the registration details.	
Main Success Scenario	Actor Action: 1. Admin logs into the system and registers a candidate. 2. Admin inputs the candidate's personal details and verifies that the candidate meets eligibility criteria. 3. Administrator submits the registration form.	System Responsibility: 4. System processes the registration and confirms the candidate registration. 5. The registration details are stored in the system's database.
Extensions	2a. Candidate is not eligible for elections. <ul style="list-style-type: none"> 2a1. Admin does not register the candidate. 2b. Incorrect candidate details. <ul style="list-style-type: none"> 2b1. Candidate details are entered again. 	

Use Case 02 - Manage electoral polling staff accounts

Use Case Name	Manage electoral polling staff accounts	
Scope	E-Voting System	
Level	User Goal	
Primary Actor	Administrator	
Stakeholders and Interests	Administrator: Needs to create, update, or deactivate staff accounts. Polling staff: Requires access to perform their election duties. Election Commission: Wants only authorized staff to manage election-related tasks.	
Preconditions	Administrator is logged into the system. Polling staff roles and permissions are defined.	
PostConditions	Polling staff accounts are created, updated, activated or deactivated. System records changes in polling staff account status.	
Main Success Scenario	Actor Action: 1. The administrator can perform different actions depending on the situation: Create: Enter details for a new polling staff member. Edit: Modify the details for the existing account. Deactivate: Disable an account.	System Responsibility: 2. The system takes the input provided by the administrator and performs the necessary action. 3. System confirms the successful update.
Extensions	1a. If account details are incomplete. <ul style="list-style-type: none"> 1a1. System prompts to complete fields. 2a. If a system error occurs. <ul style="list-style-type: none"> 2a1. The admin is asked to retry. 	

Use Case 03 - Initiate system

Use Case Name	Initiate system	
Scope	E-Voting System	
Level	User Goal	
Primary Actor	Administrator	
Stakeholders and Interests	Administrator: Needs to initialize the system for election operations. Election Commission: Wants the system to be properly set up to handle the election process. Voters: Rely on a properly initiated system to cast their votes securely and efficiently.	
Preconditions	System is installed and available for initialization. Administrator has the necessary credentials and permissions.	
PostConditions	System is successfully initiated and ready for election processes. All components are properly initialized.	
Main Success Scenario	Actor Action: 1. Administrator configures settings by initializing databases etc.	System Responsibility: 2. System processes the configuration and initializes the necessary modules (voting, candidate management, etc.). 3. System verifies the settings and confirms successful initiation to the administrator.
Extensions	1a. If any configuration is incomplete: <ul style="list-style-type: none"> 1a1. The system prompts the administrator to complete the missing fields. 2a. If system errors occur during initialization: <ul style="list-style-type: none"> 2a1. The system logs the error and prompts the administrator to retry. 	

Use Case 04 - View polling staff Assignments at the Polling Station

Use Case Name	View polling staff Assignments at Polling Station	
Scope	E-Voting System	
Level	User Goal	
Primary Actor	Administrator	
Stakeholders and Interests	Administrator: Needs to view which polling staff members are assigned to specific polling stations. Polling Staff: Requires clarity on their assignments. Election Commission: Ensures proper staffing and operational readiness at all polling stations.	
Preconditions	Polling stations and polling staff assignments have been configured in the system. Administrator is logged into the system.	
PostConditions	Administrator successfully views the assigned polling staff for a particular polling station. The system generates a detailed list of polling staff members and their roles at the selected station.	
Main Success Scenario	Actor Action: 1. Administrator selects the option for viewing Polling Staff. 2. Administrator enters/selects a polling station. 5. Administrator views the list of assigned Polling staff and their roles.	System Responsibility: 4. System retrieves and displays the Polling staff assignment data for the selected polling station.
Extensions	5a. System error occurs <ul style="list-style-type: none"> 5a1. Admin retries. 	

Use Case 05 - Authorize polling staff Entry

Use Case Name	Authorize polling staff Entry	
Scope	E-Voting System	
Level	User Goal	
Primary Actor	Polling staff	
Stakeholders and Interests	<p>Polling staff: Needs secure access to manage the polling process.</p> <p>Administrator: Ensures only authorized polling staff can access the system.</p> <p>Election Commission: Ensures proper authorization and secure access to the e-voting system.</p> <p>Voters: Rely on the system being managed by properly authorized personnel.</p>	
Preconditions	<p>The polling staff has valid login credentials.</p> <p>Polling station setup is complete.</p>	
PostConditions	<p>Polling staff are successfully authorized to manage the polling station's operations within the e-voting system.</p> <p>The system logs the officer's entry for future audits.</p>	
Main Success Scenario	<p>Actor Action:</p> <ol style="list-style-type: none"> 1. The polling staff enters login credentials. 	<p>System Responsibility:</p> <ol style="list-style-type: none"> 2. System verifies the credentials. 3. System verifies whether the MAC address assigned to the polling station matches the MAC address of this PC. 3. A polling staff is granted access to manage the polling station within the system. 4. The corresponding voter list is provided to the staff for the specified station. 5. System logs the successful authorization and allows access to polling management functions.
Extensions	<p>1a. Incomplete fields:</p> <ul style="list-style-type: none"> • 1a1. Officer is requested to fill all the fields. <p>2a. Invalid credentials:</p>	

	<ul style="list-style-type: none"> 2a1. System denies access and prompts re-entry.
--	---

Use Case 06 - Cast Vote

Use Case Name	Cast Vote	
Scope	E-Voting System	
Level	User Goal	
Primary Actor	Voter	
Stakeholders and Interests	<p>Voter: Wants to ensure their vote is cast securely and correctly.</p> <p>Polling staff: Assists voters in the voting process and ensures the system operates correctly.</p> <p>Election Commission: Wants to ensure a fair, transparent, and efficient voting process.</p>	
Preconditions	The voter has been successfully verified and is eligible to vote.	
PostConditions	The vote is successfully cast and recorded in the system.	
Main Success Scenario	<p>Actor Action:</p> <ol style="list-style-type: none"> 1. Voter approaches the system. 3. Voter selects their candidate. 5. Voter confirms their selection. 7. Voter receives a confirmation of their vote. 	<p>System Responsibility:</p> <ol style="list-style-type: none"> 2. System displays the voting interface. 4. System verifies the selection and prepares to record the vote. 6. System records the vote if the voter has not already cast a vote. 8. System updates the total votes for the selected candidate.
Extensions	<p>3a. If the voter attempts to cast a vote without confirming their selection:</p> <ul style="list-style-type: none"> 3a1. System prompts the voter to confirm their choice before proceeding. <p>5a. If the vote cannot be recorded due to a system error:</p> <ul style="list-style-type: none"> 5a1. System informs the voter of the issue and allows them to 	

	retry the vote.
--	-----------------

Use Case 07 - Monitor active systems

Use Case Name	Monitor active systems	
Scope	E-Voting System	
Level	User Goal	
Primary Actor	Admin	
Stakeholders and Interests	Admin: Needs to ensure all systems are functioning properly on polling day. Election Commission: Wants to prevent electoral fraud.	
Preconditions	The admin is logged into the system. Voting systems are operational and connected to the system.	
PostConditions	Admin has a real-time view of all active systems.	
Main Success Scenario	Actor Action: 1. Admin requests the status of active polling stations. 3. Admin reviews the status of each polling station. 5. Admin identifies any polling stations with issues.	System Responsibility: 2. The system retrieves the current status from all polling stations. 4. The system displays the operational status.
Extensions	1a. If the system is unable to retrieve the status from any polling station: <ul style="list-style-type: none"> 1a1. The system notifies the admin of the connection issue and provides details for troubleshooting. 3a. If the admin finds a polling station with a malfunctioning system: <ul style="list-style-type: none"> 3a1. The system allows the admin to initiate a support request to resolve the issue. 	

Use Case 08 - Inspect Audit logs

Use Case Name	Inspect Audit logs	
Scope	E-Voting System	
Level	User Goal	
Primary Actor	Admin	
Stakeholders and Interests	Admin: Requires access to audit logs to monitor system activity and ensure transparency in the voting process. Election Commission: Ensures the integrity and transparency of the electoral process.	
Preconditions	The system logs votes and actions during polling. The admin is logged into the system with sufficient privileges to access audit logs.	
PostConditions	Admin reviews audit logs successfully.	
Main Success Scenario	Actor Action: 1. Admin requests to view the audit logs. 3. Admin reviews the displayed audit logs. 5. Admin filters the logs based on date or event type.	System Responsibility: 2. The system retrieves the relevant audit logs based on the specified criteria. 4. System highlights any anomalies or significant events recorded in the logs. 6. The system updates the display to reflect the filtered logs.
Extensions	1a. If the system cannot retrieve the audit logs: <ul style="list-style-type: none"> 1a1. The system notifies the admin of the issue and provides troubleshooting options. 3a. If the admin identifies a suspicious activity in the logs: <ul style="list-style-type: none"> 3a1. The system allows the admin to initiate an investigation. 5a. If the filtering criteria yield no results: <ul style="list-style-type: none"> 5a1. The system informs the admin that no logs match the specified criteria and prompts them to adjust their search. 	

Use Case 09 - Capture Voter information

Use Case Name	Capture Voter information	
Scope	E-Voting System	
Level	User Goal	
Primary Actor	Polling staff	
Stakeholders and Interests	Polling staff: Needs to ensure voter's information is captured correctly Voter: Needs to have their information verified before voting. Election Commission: Wants to have a record of the votes .	
Preconditions	Voter is present with valid identification at a polling station.	
PostConditions	Voter information is stored in the system.	
Main Success Scenario	Actor Action: 1. Polling staff initiates the process of capturing voter information. 3. Polling staff enters voter information. 5. Polling staff confirms data entry.	System Responsibility: 2. System requests the required voter data. 4. System verifies if the voter belongs to this polling station and records the information.
Extensions	3a. If the voter information entered by the polling staff is incomplete or incorrect: <ul style="list-style-type: none"> 3a1. System displays a warning and asks the polling staff to correct the mistake. 4a. If the voter does not belong to the current polling station: <ul style="list-style-type: none"> 4a1. System alerts the polling staff and provides the voter's registered polling station details, if available. 5a. If the polling staff accidentally confirms incorrect data: <ul style="list-style-type: none"> 5a1. System allows the polling staff to undo or revise the data entry before final submission. 	

Use Case 10 - Issue election Report

Use Case Name	Issue election Report	
Scope	E-Voting System	
Level	User Goal	
Primary Actor	Administrator	
Stakeholders and Interests	Administrator: Needs to generate and distribute a final election report. Election Commission: Requires an accurate and comprehensive report to certify the election. Candidates and Voters: Seek transparency and final confirmation of results.	
Preconditions	The system on all polling stations is terminated, the vote validation has been done and recounts (if any) have concluded.	
PostConditions	A detailed election report is generated, including vote counts, participation statistics, and any relevant observations.	
Main Success Scenario	Actor Action: 1. The administrator selects the option to generate the election report. 4. The administrator reviews and finalizes the report.	System Responsibility: 2. The system compiles the necessary data from the vote tally, voter turnout, and other related information. 3. The system generates and stores the election report for future reference.
Extensions	3a. If the report fails to generate due to system errors, the administrator is prompted to retry	

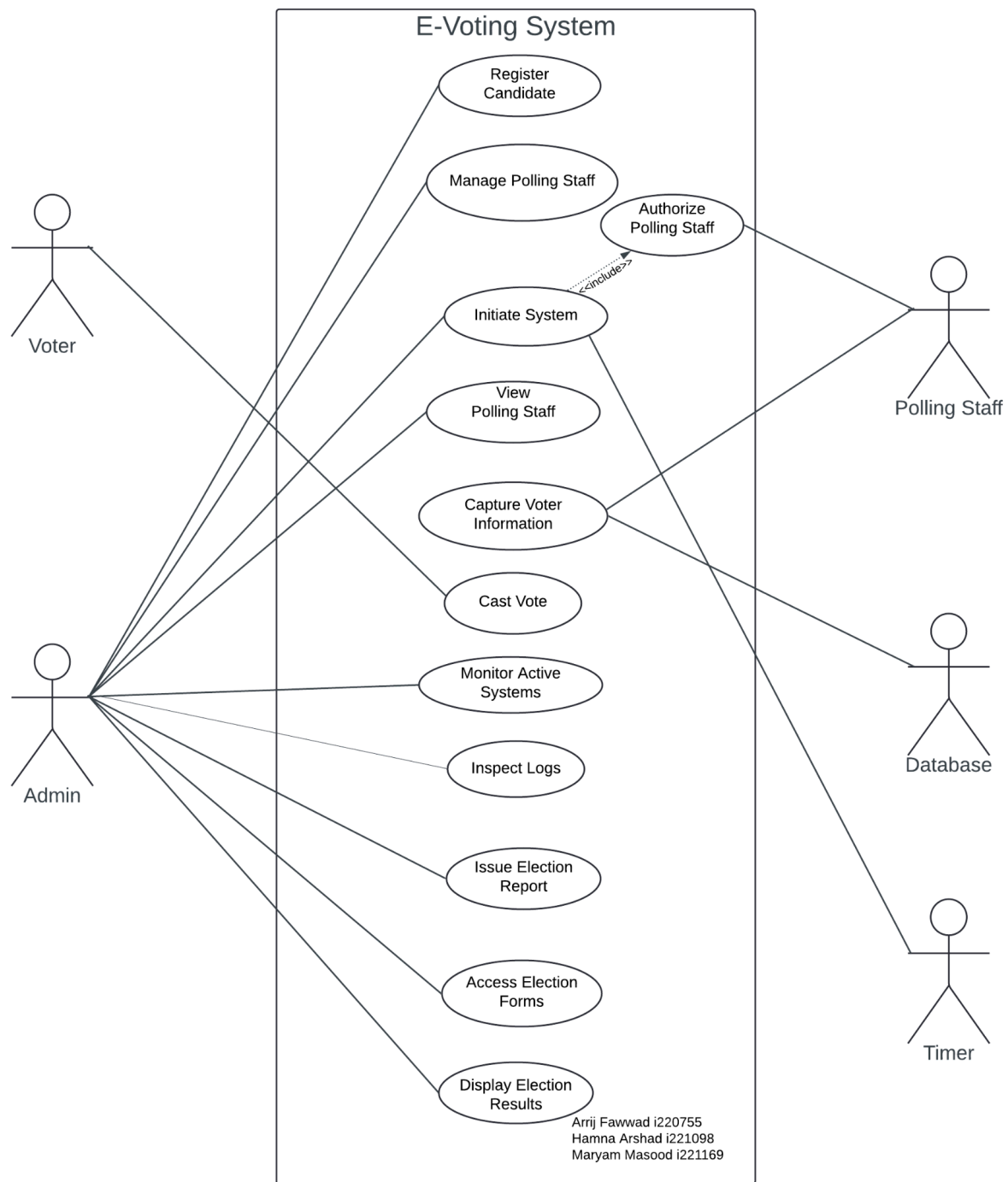
Use Case 11 - Access Election Forms

Use Case Name	Access Election Forms	
Scope	E-Voting System	
Level	User Goal	
Primary Actor	Administrator	
Stakeholders and Interests	<p>Administrator: Needs to review election forms for transparency and to ensure all results are properly documented.</p> <p>Candidates and Parties: Rely on these forms for validating vote counts and official results.</p>	
Preconditions	The system on all polling stations is terminated, the vote validation has been done and recounts (if any) have concluded.	
PostConditions	The administrator successfully accesses any requested form.	
Main Success Scenario	<p>Actor Action:</p> <ol style="list-style-type: none"> 1. The administrator selects the option to access election forms. 3. The administrator chooses the specific form to view. 5. The administrator reviews the contents of the selected form. The administrator exports or prints the form if necessary. 	<p>System Responsibility:</p> <ol style="list-style-type: none"> 2. The system displays the list of available forms. 4. Upon the administrator's selection, the system retrieves and displays the chosen form with all relevant data.
Extensions	<p>2a. No available forms:</p> <ul style="list-style-type: none"> 2a1. The system informs the administrator that no forms are currently available for access. 2a2. The system suggests checking back later after vote validation is completed. <p>4a. Form retrieval error:</p> <ul style="list-style-type: none"> 4a1. The system encounters an error while retrieving the selected form (e.g., file corruption, server issue). 4a2. The system displays an error message, explaining the issue to the administrator. 	

Use Case 12 - Display Election Results

Use Case Name	Display Election Results	
Scope	E-Voting System	
Level	User Goal	
Primary Actor	Administrator	
Stakeholders and Interests	Administrator: Needs to view the final election results (vote counts) after they are verified. Candidates and Voters: Indirectly impacted as they expect a fair and accurate election outcome, but they do not access this internal view.	
Preconditions	The voting process and vote verification have been completed, and there is no recounting in progress. The final results have been consolidated.	
PostConditions	Election results are displayed to the Administrator . The system logs the action for auditing purposes.	
Main Success Scenario	Actor Action: 1. The Administrator selects the option to display the final election results.	System Responsibility: 2. The system retrieves the final, verified results. 4. The system displays the election results on the public interface (e.g., website, announcement board). 5. The system logs the result display event for auditing purposes.
Extensions	1a. If the results have not been verified or finalized: 1a1. The system notifies the Administrator and prevents results from being displayed until the verification is complete. 5a. If there is an issue with the display interface: 5a1. The system alerts the Administrator and provides troubleshooting steps.	

2.5 Use Case Diagram



3. Other Nonfunctional Requirements

3.1 Performance Requirements

The Votix e-voting system must perform efficiently under a range of operational conditions to ensure smooth election processes:

- **System Response Time:** The system should process voter authentication and vote-casting requests within **2 seconds** to maintain user satisfaction and prevent delays during voting.
- **Real-Time Updates:** Election results and polling station data must be synchronized in real-time, with a maximum update delay of **5 seconds**.
- **System Scalability:** The system should handle up to **100,000 concurrent users** without degradation in performance, ensuring it can support elections in larger populations.

These requirements are critical for ensuring that Votix provides a smooth and efficient experience, particularly in high-stakes election environments.

3.2 Safety Requirements

The safety of election data and voting processes is paramount. Key safety measures include:

- **Data Backup and Redundancy:** The system must implement real-time data backups using csv file handling and fault tolerance mechanisms to prevent data loss in the event of hardware failures.
- **Emergency Protocols:** If a polling station experiences technical failure, the system must alert administrators and provide an alternative solution to ensure voters can continue casting their votes.
- **Electoral Integrity:** Any actions that could compromise election integrity, such as tampering with votes or data manipulation, must be prevented with robust security protocols.

These measures are designed to safeguard the election process from physical and digital threats.

3.3 Security Requirements

Security is a critical aspect of the Votix system to protect sensitive electoral data and maintain public trust:

- **Authentication:** The system must require **authentication** for all users, including the presiding officers and election administrators, to verify their identity before accessing the system.
- **Polling PC Access Control:** The polling PCs (laptops) are designated to the **Presiding Officer**. Only the presiding officer can log into the system using their unique credentials.

This ensures that only authorized personnel can manage the voting process at polling stations.

- **MAC Address Validation:** Each polling PC's **MAC address** is stored in the database. This ensures that the application can only run on the designated laptop, preventing unauthorized instances from running on different machines.
- **Audit Trails:** The system must maintain detailed logs of user activity, vote casting, and administrative actions for transparency and post-election audits. These logs should be tamper-proof and available for review if needed.
- **Access Control: Role-based access control (RBAC)** must be enforced, ensuring that only authorized personnel, such as presiding officers and election administrators, can access sensitive functions and data.

These security measures ensure that Votix meets regulatory standards, safeguards electoral data, and prevents unauthorized access, fraud, and data manipulation.

3.4 Software Quality Attributes

Several software quality attributes are crucial for ensuring that the Votix system meets user expectations and functions effectively:

- **Reliability:** The system should have an uptime of **99.9%** during elections, ensuring availability even during peak voting periods.
- **Usability:** The user interface must be intuitive and easy to navigate, with a focus on simplicity for voters and election administrators. This will reduce the learning curve and enhance adoption.
- **Maintainability:** The system should be modular, allowing for easy updates and maintenance without disrupting election processes.
- **Interoperability:** The system must integrate seamlessly with existing databases and other government systems (e.g., voter registration databases).
- **Scalability:** Votix must be scalable to handle an increasing number of users and data volume as the system is deployed in more regions or larger elections.

These attributes ensure that the system is effective, adaptable, and able to perform under various conditions.

3.5 Business Rules

- **Voter Eligibility:** Only registered voters will be allowed to cast their vote, verified by the voter registration database.
- **Role-based Access:** Only election administrators can modify polling station settings, while polling staff can only view their voter status and cast votes.
- **Vote Integrity:** Once a vote is cast, it cannot be altered, and any attempt to tamper with the vote will be flagged and logged for investigation.

- **Election Results:** Only authorized election officials will have access to the final results, and they must follow the prescribed protocol for releasing them after all votes are counted and verified.

These rules ensure the integrity and smooth functioning of the election process.

3.6 Operating Environment

The Votix e-voting system will operate within the following environment:

- **Hardware Platform:** The system will run on servers with a minimum configuration of **8GB RAM, 4-core CPU, and 500GB SSD storage** to ensure optimal performance.
- **Operating System:** The software will be compatible with **Windows Server 2019+** and **Linux-based servers** (e.g., Ubuntu 20.04 or Kali Linux).
- **Database Management System:** The system will use **MySQL 8.x** for secure data storage and management.
- **Client Devices:** PollingStaff will interact with the system via designated Laptops.
- **Network Environment:** The system will operate over a **VPN network (Tailscale)**, ensuring encrypted communication between polling stations and the central server.

These specifications ensure that the Votix system operates reliably in the intended environment, providing both security and performance during elections.

3.7 User Interfaces

Main Page:



The main page serves as the central hub for navigating to the admin and staff login sections or learning more about the application. It features a clean, professional interface with a dark green background (#385B4F) for a cohesive and modern appearance.

Key Interface Elements:

- **Buttons:**
 - **Admin Login:**
 - Positioned at the center-right of the screen.
 - Styled with a gradient background (linear-gradient(to bottom, #385B4F, #2c473d)) and rounded corners for aesthetic appeal.
 - White text in "Century Gothic Bold Italic" font for readability.
 - **Staff Login:**
 - Positioned below the Admin Login button.
 - Shares the same style for consistency.
 - **About:**
 - Positioned below the Staff Login button.
 - Opens a page describing the system's purpose and functionality.
- **Logo and Branding:**
 - Displays the company logo centrally for brand recognition.
 - Includes a tagline graphic positioned above the logo to convey the system's purpose.
- **Navigation Icons:**

- Admin, staff, and about buttons are paired with relevant icons for quick visual identification.

GUI Standards and Style Guides:

- All buttons have uniform dimensions and consistent hover/click animations.
- A horizontal red line adds a sleek, professional touch.

The layout ensures equal spacing and alignment of elements

Login Page:

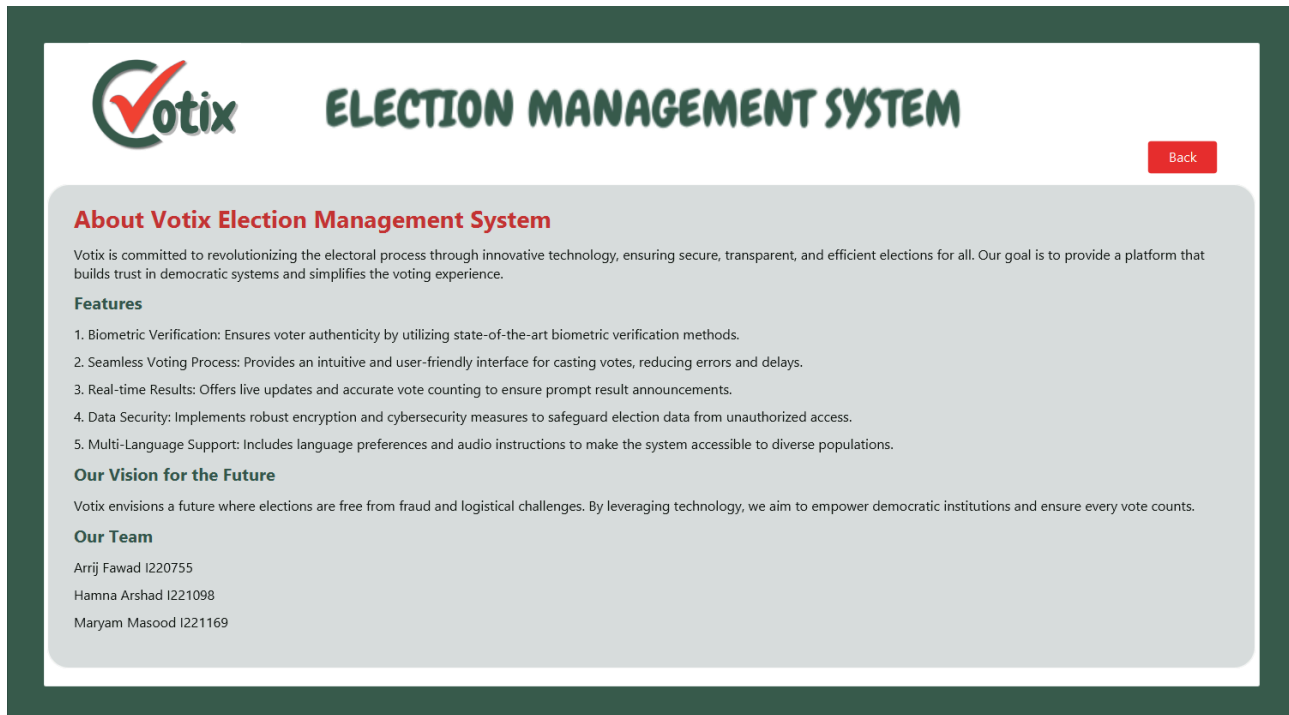


The login page allows admin and staff users to authenticate using their credentials. It emphasizes clarity and simplicity to enhance usability.

Key Interface Elements:

- **Credential Input Fields:**
 - Two fields: one for the username and one for the password.
 - Placeholder text ("Username" and "Password") for guidance.
 - Positioned centrally in a semi-transparent rectangular area for focus.
- **Login Button:**
 - Styled identically to the buttons on the main page.
 - Includes a prominent label "Login".
- **Back Button:**
 - Positioned in the top-right corner to allow users to return to the main page easily.
 - Styled in the same theme as the rest of the application.

About Page::



The About page offers a detailed description of the system's features, purpose, and benefits.

Key Interface Elements:

- **Header:**
 - Includes the logo and a title, reinforcing branding.
- **Content Area:**
 - Uses a VBox for organized text and sections.
 - Displays a brief description of the voting system's goals, such as transparency and efficiency.
 - Styled with bold headers and concise paragraphs for readability.
- **Visual Aids:**
 - Graphical assets and icons complement the text, creating an engaging experience.

Consistency Across Pages:

- Common elements like logo placement, font styles, and background color maintain a unified look and feel.
- Buttons and interactive components follow standard visual cues for intuitiveness.

Software Components Requiring User Interface

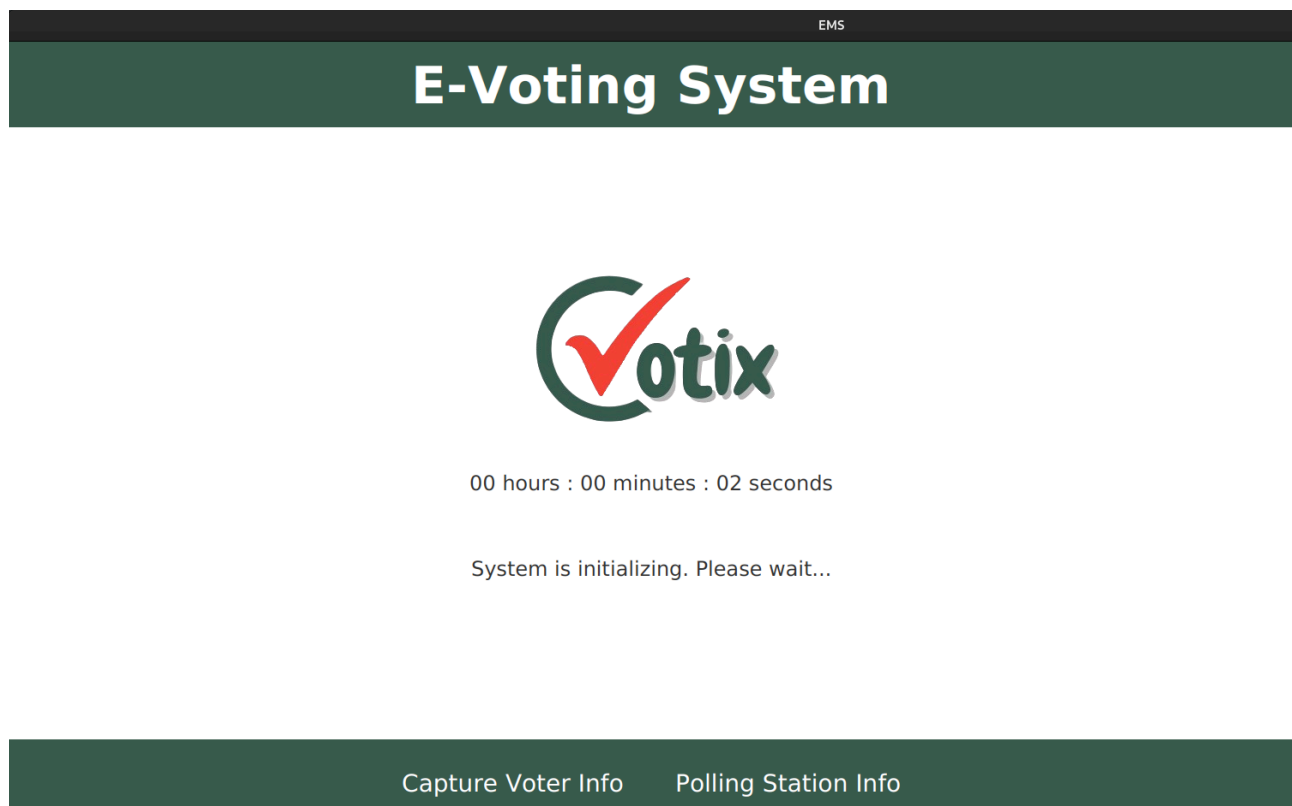
- **Main Page Controller:** Handles navigation to admin login, staff login, and about sections.

- **Login Controller:** Validates user credentials and routes authenticated users to respective dashboards.
- **About Controller:** Displays application details and responds to navigation requests.

Sample Screen Images and Design Documentation

The screen layouts align with the provided FXML files, ensuring a pixel-perfect implementation of the designs. Further refinements, such as keyboard shortcuts (e.g., Tab for navigation and Enter to submit forms) and error message standards, will be outlined in the **User Interface Specification Document**.

Initial Polling PC Screen:



Capture Voter Info:

Capture Voter Information

Enter voter's name:

Enter voter's CNIC:

Submit

Capture Voter Info **Polling Station Info**

Activate Windows
Go to Settings to activate Windows.

This interface facilitates voter information capture by allowing polling staff to validate and process voter details before granting access to cast a vote.

1. Interface Overview

- **Inputs:**
 - **Name Field:** Restricted to alphabets and spaces only.
 - **CNIC Field:** Accepts digits and hyphens.
- **Submission:**
 - Validate inputs and check voter registration and voting status.
 - Redirects to the vote-casting interface for eligible voters.

2. Design Standards

- **Styling:**
Consistent use of `style.css` for layout, ensuring readability and uniform design.
- **Responsiveness:**
Center-aligned form section scales effectively on the primary screen.
- **Accessibility:**
Clear labels and error feedback for incorrect or missing input.

3. Error Handling

- **Invalid Input:**
 - Displays error popups for missing or malformed data.
- **Already Voted:**
 - Prevents duplicate voting with specific error feedback.
- **Unregistered Voter:**
 - Indicates if the voter is not found in the system.

4. Workflow Details

- **Validation:**
 - Real-time field restrictions ensure proper formatting (e.g., letters for names, and numbers for CNIC).
- **Navigation:**
 - On valid submission, voters proceed to the Cast Vote interface.
 - Utilizes **FXMLLoader** for dynamic view changes.

Polling Station Info:

Polling Station Information	
Total Registered: 50	
Voted: 13	
Not Voted: 37	
CNIC	Status
12345-0000001-1	Voted
12345-0000002-1	Voted
12345-0000003-1	Voted
12345-0000004-1	Voted
12345-0000005-1	Not Voted
12345-0000006-1	Voted
12345-0000007-1	Voted
12345-0000008-1	Not Voted
12345-0000009-1	Not Voted
12345-0000010-1	Not Voted
12345-0000011-1	Not Voted
12345-0000012-1	Not Voted
12345-0000013-1	Not Voted
12345-0000014-1	Not Voted
12345-0000015-1	Not Voted
12345-0000016-1	Not Voted

To display voter registration, voting status, and detailed voter information for a polling station, aiding the polling staff in real-time monitoring.

1. Interface Overview

- **Data Display:**
 - **Summary Statistics:**
 - Total Registered Voters
 - Voters Who Have Voted
 - Voters Who Haven't Voted
 - **Voter Details:**
 - CNIC
 - Voting Status (Voted/Not Voted)
- **Dynamic Table:**
 - Uses a scrollable table to show all registered voters, with rows populated dynamically.

2. Design Standards

- **Styling:**

- Consistent design with `style.css` for a clean, professional appearance.
 - Zebra striping for better row distinction in the voter table.
- **Responsiveness:**
 - Centralized table layout for balanced and clear information display.
- **Accessibility:**
 - Labels use concise, descriptive text for clarity.

3. Data Handling

- **Dynamic Row Creation:**
 - Populates rows with CNIC and voting status pulled from the `PollingStation` object.
- **Color Indicators:**
 - **Voted:** Styled with green for easy identification.
 - **Not Voted:** Styled with red for emphasis.

4. Error Handling

- Safeguards against null data:
 - Ensures the `PollingStation` object is valid before processing.
- Prevents crash during row creation or count updates for incomplete data.

5. Workflow Details

- **Initialization:**
 - Calls `setPollingStation` to set the polling station and fetch voter details.
- **Data Update:**
 - `populateVoterTable` dynamically fills the voter table based on polling station data.
 - `updateVoterCounts` calculates and updates statistical labels.

Cast Vote:

Candidate Name	Party Name	Party Symbol	Select
Bilawal Bhutto Zardari	Pakistan Peoples Party		<input type="checkbox"/>
Shehbaz Sharif	Pakistan Muslim League (N)		<input type="checkbox"/>
Shah Mahmood Qureshi	Pakistan Tehreek-e-Insaf		<input type="checkbox"/>
Hafiz Saeed	Milli Muslim League		<input type="checkbox"/>
Siraj-ul-Haq	Jamaat-e-Islami		<input type="checkbox"/>
Saad Hussain Rizvi	Tehreek-e-Labbaik Pakistan		<input type="checkbox"/>
Muhammad Tahir-ul-Qadri	Pakistan Awami Tehreek		<input type="checkbox"/>
Maulana Fazlur Rehman	Jamiat Ulema-e-Islam (F)		<input type="checkbox"/>
Sheikh Rashid Ahmad	Awami Muslim League		<input type="checkbox"/>
Zahid Khan	Awami National Party		<input type="checkbox"/>

Submit

1. Interface Overview

Voter Information Display:

- **Data Display:**
 - Voter Name.
 - CNIC (Unique Identifier).
- **Instructions Section:**
 - Provides clear guidance for voters to select candidates via checkboxes.
- **Candidate Selection:**
 - Dynamically generated table of candidates with the following details:
 - Candidate Name.
 - Party Name.
 - Party Symbol.
 - Selection Checkbox.

Dynamic Table:

- Displays rows for all candidates in a scrollable format.
- Each row is dynamically added based on the data from the Election Management System (EMS).

Voting Action:

- **Submit Button:**
 - Positioned at the bottom center.
 - Triggers vote submission after validation.

2. Design Standards

Styling:

- Consistent styling with `style.css` for a professional look.
- Zebra striping for alternating row colors to enhance readability in the candidate table.
- Labels and buttons styled for clarity and emphasis.

Responsiveness:

- Adaptive layout with proper alignment for different screen sizes.
- Scrollable candidate list ensures the table does not exceed the screen height.

Accessibility:

- Labels and instructions written in concise, clear text.
- High-contrast colors and adequate font sizes for better visibility.

3. Data Handling

Dynamic Row Creation:

- Rows are populated using the `populateCandidates` method, fetching data from the EMS.
- Candidate details are extracted and displayed in their respective columns.

Interactive Elements:

- Each checkbox is mapped to a candidate's ID for easy identification during vote submission.
- Single-selection enforcement ensures only one candidate is selected at a time.

Vote Submission:

- On clicking "Submit," the selected candidate's ID is retrieved and sent to the EMS for processing.

4. Error Handling

Validation:

- Ensures voter details are loaded correctly from the EMS.
- Prevents vote submission if no candidate is selected.

Null Data Safeguards:

- Validates candidate data before dynamically generating rows.

- Handles cases where EMS or voter details are unavailable gracefully.

Backend Notification:

- Sends logs and notifications to the backend server using HTTP POST, with retries in case of failure.

5. Workflow Details

Initialization:

- `setElectionManagementSystem`:
 - Initializes the EMS.
 - Fetches voter details by CNIC and displays them in the interface.
 - Dynamically loads candidates for voting.

Data Update:

- `populateCandidates` dynamically refreshes the candidate list to reflect updates from the EMS.

Vote Casting:

- The selected candidate's ID is passed to `ems.castVote`, updating the voter's status and logging the action.
- After successful submission, the voting window is closed.

View Logs:

Log ID	Action	Time Stamp
40	System 55 restarted successfully	2024-11-26 15:30:00
39	System error occurred on polling PC 55	2024-11-26 15:25:11
38	A vote was cast for CNIC: 12345-0000051-1 at polling PC 55, Station 3, Area 3, Lahore	2024-11-26 15:20:50
37	The Polling PC 12 was closed for station ID: 2 in area: 2, Karachi	2024-11-26 14:15:30
36	A vote was cast for CNIC: 12345-0000044-1 at polling PC 12, Station 2, Area 2, Karachi	2024-11-26 14:10:12
35	System 12 initialized	2024-11-26 14:05:45
41	System 55 initialized	2024-11-26 13:22:46
9	Polling station 10 maintenance completed	2024-11-12 10:40:00
8	Vote count tallying started for Area 5	2024-11-12 10:35:00
7	Database backup completed successfully	2024-11-12 10:30:00
6	System reboot after error at polling station 2	2024-11-12 10:25:00

The **View Logs** interface provides an intuitive and interactive platform for administrators to manage and analyze system logs efficiently. Below are the logical characteristics and design elements of the user interface:

1. Filter Functionality:

- **Search by Keyword:** A text field allows users to filter logs by action keywords.
- **Search by Area or Polling Station ID:** Separate text fields accept numeric inputs for area and polling station IDs, ensuring users can pinpoint specific logs. Validation restricts these fields to numeric entries only.

2. Log Display:

- **Dynamic Table Layout:** Logs are displayed in a vertically scrollable table (VBox), dynamically updated based on the filters applied.
- **Columns and Sorting:**
 - Columns include **Log ID**, **Action**, and **Timestamp**.
 - Logs are sorted by timestamp in descending order for quick access to recent events.

3. Real-Time Updates:

- The interface integrates with an **SSE (Server-Sent Events)** mechanism to update logs in real time when new events are detected.

4. Reusable and Standardized Design:

- Consistent use of styles, such as font sizes and alignment, ensures readability and adherence to a uniform design standard.

- Highlighting mechanisms visually emphasize search results (e.g., bold yellow background for matched terms).
- 5. **Keyboard Shortcuts and Interactivity:**
 - Text fields clear other filters upon gaining focus, streamlining user interactions.
 - Navigation buttons, like the "Back" action, allow users to return to the main menu effortlessly.
- 6. **Error Handling and Feedback:**
 - Invalid inputs are prevented via built-in validators for numeric fields.
 - The system provides visual feedback, ensuring users can detect and resolve issues without confusion.
- 7. **Standard Components:**
 - **Buttons:** Filter and reset buttons are used for clear user commands.
 - **Error Messages:** Validation ensures no invalid data is processed, minimizing manual error correction.

Monitor Polling Systems:

System ID	Station ID	Status	Config	Area Name
1	1	Active	84:14:4D:F7:CD:E4	Islamabad
2	1	Active	00:1A:2B:3C:4D:5F	Islamabad
3	1	Active	00:1A:2B:3C:4D:60	Islamabad
55	1	Active	f4:96:34:28:c6:b6	Islamabad
56	1	Active	10:6f:d9:0d:2a:fb	Islamabad
4	2	Active	00:1A:2B:3C:4D:61	Islamabad
5	2	Active	00:1A:2B:3C:4D:62	Islamabad
6	2	Active	00:1A:2B:3C:4D:63	Islamabad
7	3	Inactive	00:1A:2B:3C:4D:64	Islamabad
8	3	Active	00:1A:2B:3C:4D:65	Islamabad
9	3	Active	00:1A:2B:3C:4D:66	Islamabad

The interface allows administrators to monitor polling systems, filter them by status ("All," "Active," "Inactive"), and view key details such as System ID, Station ID, Status, Configuration, and Area Name.

2. Interface Overview

- **Filtering:** A ComboBox enables filtering systems dynamically based on their status.

- **Dynamic Table:** Displays system details in rows updated dynamically based on filter selection.
- **Back Navigation:** A back arrow icon navigates to the admin menu.

3. Design Standards

- **Styling:** Consistent with `style.css`, with dynamic color codes for system statuses (e.g., green for active, red for inactive).
- **Accessibility:** Clear text, scalable fonts, and high-contrast design.
- **Keyboard Shortcuts:** Use `Tab` to navigate, `Enter` to apply filters.

4. Error Handling

- Defaults to "All" on invalid ComboBox input.
- Real-time updates ensure user feedback without page reloads.

5. Table Structure

- **Columns:** System ID, Station ID, Status, Configuration, Area Name.
- **Dynamic Rows:** Added or removed based on filter criteria.
- **ScrollPane:** Ensures smooth navigation through a large number of systems.

Admin Menu:



1. General Layout and Design:

- **Primary Layout:** The `AdminMenuController` FXML specifies an `AnchorPane` as the root container with a fixed width of 1500px and height of 858px. The UI elements (buttons and labels) are strategically placed to guide admin actions efficiently.
- **Title Bar:**
 - **Style:** The title bar at the top follows a consistent design with the rest of the application, displaying "Votix" prominently.
 - **Functionality:** Provides a constant reference to the system's name, enhancing brand consistency and aiding navigation.
- **Main Content Area:**

The `contentPane` dynamically loads different screens, ensuring the user always remains within the context of the admin dashboard without needing to switch windows.

2. GUI Standards and Components:

- **Color Scheme:**

The interface adheres to a standardized color scheme defined in `style.css` to maintain consistency. Buttons highlight or change appearance upon interaction, following the `selected` class logic.
- **Buttons:**
 - **Standard Buttons:**
 - All key functionalities (e.g., `viewLogs`, `addCand`, `viewStaff`) are represented as buttons.
 - Consistent size and spacing ensure a balanced look, following a grid layout.
 - **Highlighted States:**

The active button is highlighted using a `selected` class to give users a visual cue about their current selection.
- **Standard Functions:**
 - **Logout Functionality:** Accessible through a button and an `ImageView`. Provides a way for admins to securely log out and return to the main page.
 - **Error Handling:** Error messages are printed to the console, indicating exceptions during FXML loading. This behavior should be extended to show user-friendly error pop-ups.

3. Navigation and User Flow:

- **Menu Navigation:**

Users can seamlessly navigate between different functionalities (e.g., viewing candidates, adding polling staff, or monitoring systems). Each button triggers loading an appropriate FXML screen into the `contentPane`.
- **Consistency:**

Each view adheres to the same structure, ensuring that users understand where to find primary actions and information.

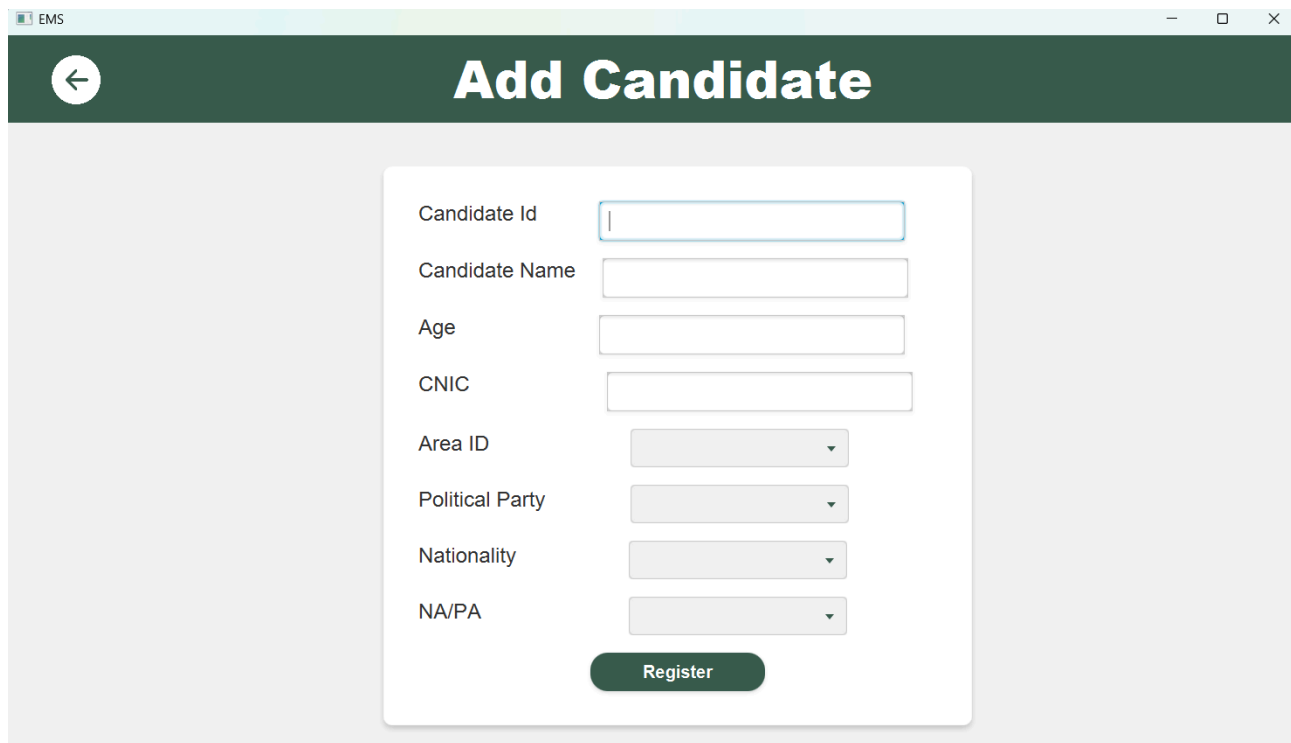
4. Error Message Display Standards:

- **Console Output:**
Currently, errors (such as FXML loading failures) are logged in the console. Future improvements should include on-screen notifications or dialog boxes to inform users about errors gracefully.
- **Validation Feedback:**
Adding error messages within the UI (e.g., input validation failures) enhances the admin's understanding of required actions.

5. GUI Components Needing a User Interface:

- **Dashboard Panels:**
Each feature loaded into the `contentPane` requires a separate, well-defined FXML layout:
 - **Candidate Management Screens** (e.g., add, view, or update candidates).
 - **Staff Management Panels** (e.g., assign polling staff or deactivate staff).
 - **System Monitoring Views** (active system status displays).
 - **Log Viewer** (displays system logs with filtering options).

Add Candidate Interface:



The screenshot displays the 'Add Candidate' interface within the EMS application. The interface has a dark green header bar at the top, which includes a back arrow icon on the left and the title 'Add Candidate' in white text. Below the header, there is a white form card with a light gray border. The form contains the following fields and controls:

- Candidate Id:** A text input field.
- Candidate Name:** A text input field.
- Age:** A text input field.
- CNIC:** A text input field.
- Area ID:** A dropdown menu.
- Political Party:** A dropdown menu.
- Nationality:** A dropdown menu.
- NA/PA:** A dropdown menu.
- Register:** A green button with white text, located at the bottom of the form card.

1. Common UI Elements:

- **Title Bar:** Present on all screens for easy navigation, with a clear system title and a back arrow for returning to the previous screen.
- **Buttons:**
 - **Standard Buttons:** "Register", "Back", and "Submit" are used throughout, with clear labeling.

- **Pop-up Messages:**
 - Success and error popups provide feedback (e.g., "Candidate Added Successfully," "Error: Invalid Data").

2. Workflow for Admin Interface (Add Candidate):

- **Step 1:** Admin navigates to "Add Candidate" from the Admin Menu.
 - **Screen Layout:** The `addCandidateController` screen appears, displaying fields for Candidate ID, Name, Age, CNIC, Area ID, Political Party, Nationality, and Assembly type.
- **Step 2:** Admin enters data for the candidate and clicks "Register."
 - **Form Validation:** The system checks for empty fields, validates data types (e.g., CNIC format), and ensures the candidate meets eligibility (age ≥ 25 , nationality "Pakistani").
- **Step 3:** Submission outcome:
 - **Success:** If validation passes, the system adds the candidate and displays a success popup with candidate details.
 - **Failure:** If validation fails (e.g., incorrect data or empty fields), an error popup appears, guiding the admin to correct the issue (e.g., "Invalid CNIC format" or "Duplicate Candidate ID").
- **Step 4:** Admin can return to the Admin Menu using the back arrow.

3. Screen Layout and Design Constraints:

- **Consistent Layout:** Fields are arranged in a vertical layout with labels on the left and input fields on the right.
- **Dropdown Menus:** Consistent style for dropdown menus (e.g., for Area ID and Political Party), populated dynamically from the database.

4. Error and Success Handling:

- **Success Popups:** After a successful action (e.g., adding a candidate), a popup appears confirming the action, with a summary of the candidate details.
- **Error Handling:** Errors (e.g., invalid input, empty fields) trigger a popup with a clear description of the issue and suggested actions for correction.
- **Input Validation:** The system enforces validation rules (e.g., CNIC format, numeric fields) before allowing form submission.

Add Polling Staff Interface:

The screenshot shows a web application window titled "EMS". The main heading is "Add Polling Staff". The form contains the following fields:

- Staff Id:
- Staff Name:
- Username:
- password:
- Station Id:
- Role:

A "Register" button is located at the bottom of the form.

General UI Characteristics:

1. **Consistency:** The interface follows a consistent style guide for all screens, ensuring a uniform layout, font, and color scheme. This contributes to a cohesive experience across the application.
2. **Navigation:** A back arrow button is always visible at the top, allowing users to return to the previous screen or menu with a single click.
3. **Form Sections:** Forms for adding polling staff include labeled fields such as "Staff ID", "Staff Name", "Username", "Password", "Station ID", and "Role". These input fields are clearly marked and arranged logically for easy data entry.
4. **Error Handling:** The system provides clear error messages when required fields are missing or when input data is in an incorrect format. These messages are shown in popup windows with concise instructions, ensuring users can correct mistakes quickly.
5. **Standard Buttons:** Buttons like "Register" are designed for clarity and perform specific actions (e.g., submit form data). The buttons are visually prominent to encourage user interaction.
6. **Accessibility:** The form fields and buttons are sized appropriately for different screen resolutions, ensuring ease of use. Text fields are wide enough for standard inputs, and labels are large enough to be read easily.

Common UI Components:

1. **Title Bar:** Every screen features a title bar at the top with a clear text label (e.g., "Add Polling Staff"), enhancing the user's understanding of their current location in the system.

2. **Form Section:** The form includes text fields, combo boxes for selecting roles and station IDs, and action buttons, placed in an easy-to-navigate vertical layout. Spacing between fields ensures a clean and organized interface.
3. **Pop-Up Windows:** Upon successful addition of a staff member, a pop-up window informs the user of the success. Error messages also appear in pop-ups with guidance on correcting input errors.

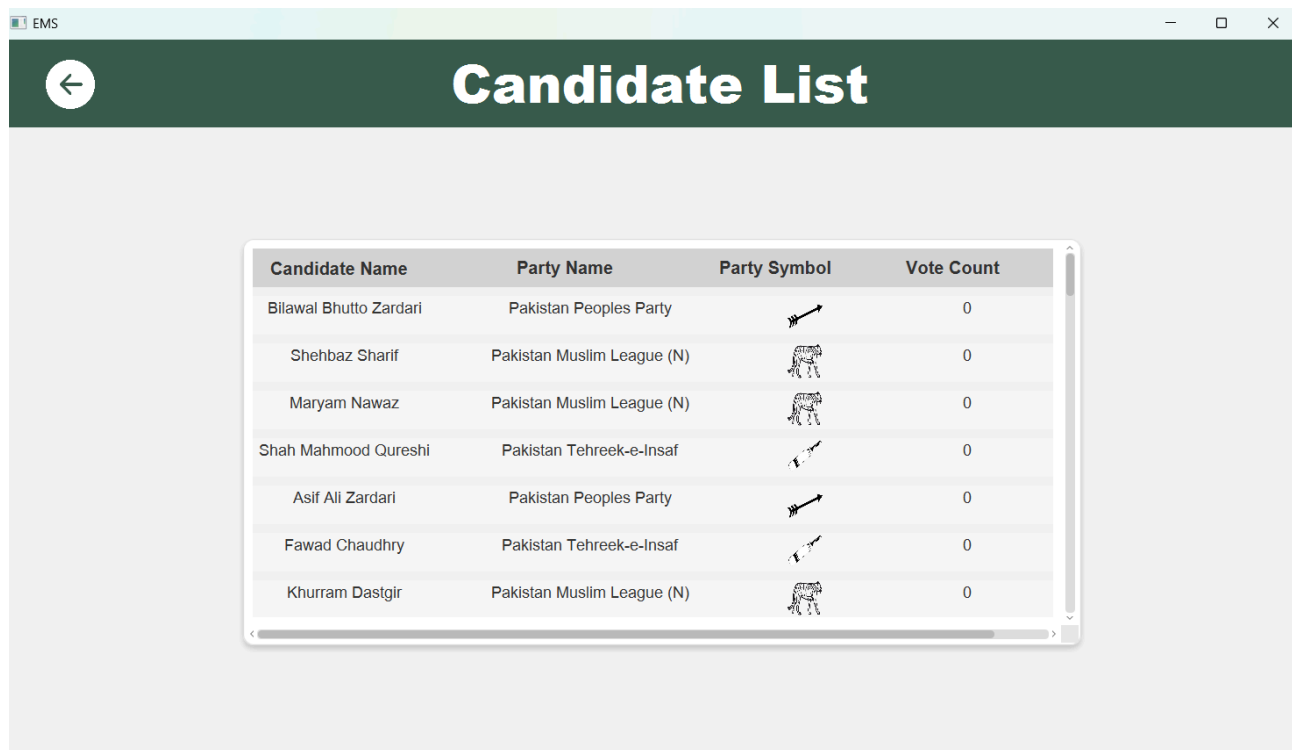
Software Components with User Interface:








- **Polling Staff Management:** A section for adding new polling staff, including entering data (Staff ID, Name, etc.), selecting roles, and assigning station IDs.
- **Error Pop-Ups:** Error messages for missing data or invalid input, are displayed in modal windows.
- **Confirmation Pop-Ups:** After successful staff addition, users are shown a confirmation message in a separate window.

Error Message Standards:

- **Clear & Actionable:** Error messages specify the problem (e.g., "Empty data fields") and offer suggestions for correction.
- **Consistency:** Error message pop-ups have a uniform design, ensuring users know they have encountered an issue and need to act.

Candidate List Interface:



Candidate Name	Party Name	Party Symbol	Vote Count
Bilawal Bhutto Zardari	Pakistan Peoples Party		0
Shehbaz Sharif	Pakistan Muslim League (N)		0
Maryam Nawaz	Pakistan Muslim League (N)		0
Shah Mahmood Qureshi	Pakistan Tehreek-e-Insaf		0
Asif Ali Zardari	Pakistan Peoples Party		0
Fawad Chaudhry	Pakistan Tehreek-e-Insaf		0
Khurram Dastgir	Pakistan Muslim League (N)		0

Components & Design:

- **Buttons & Controls:** Standard buttons (e.g., **Save**, **Cancel**) are used across the system, with consistent sizes, labels, and placements. Actions like **Add Candidate** and **Delete Candidate** are prominent.
- **Dynamic Tables:** For displaying candidate data, tables are used with interactive rows. A scrollbar allows users to navigate lists of candidates. Each row represents a candidate with columns for **Name**, **Party**, **Symbol**, and **Vote Count**.

Error & Feedback:

- **Error Messages:** Any errors are clearly displayed with straightforward messages, ensuring users can understand and correct issues.

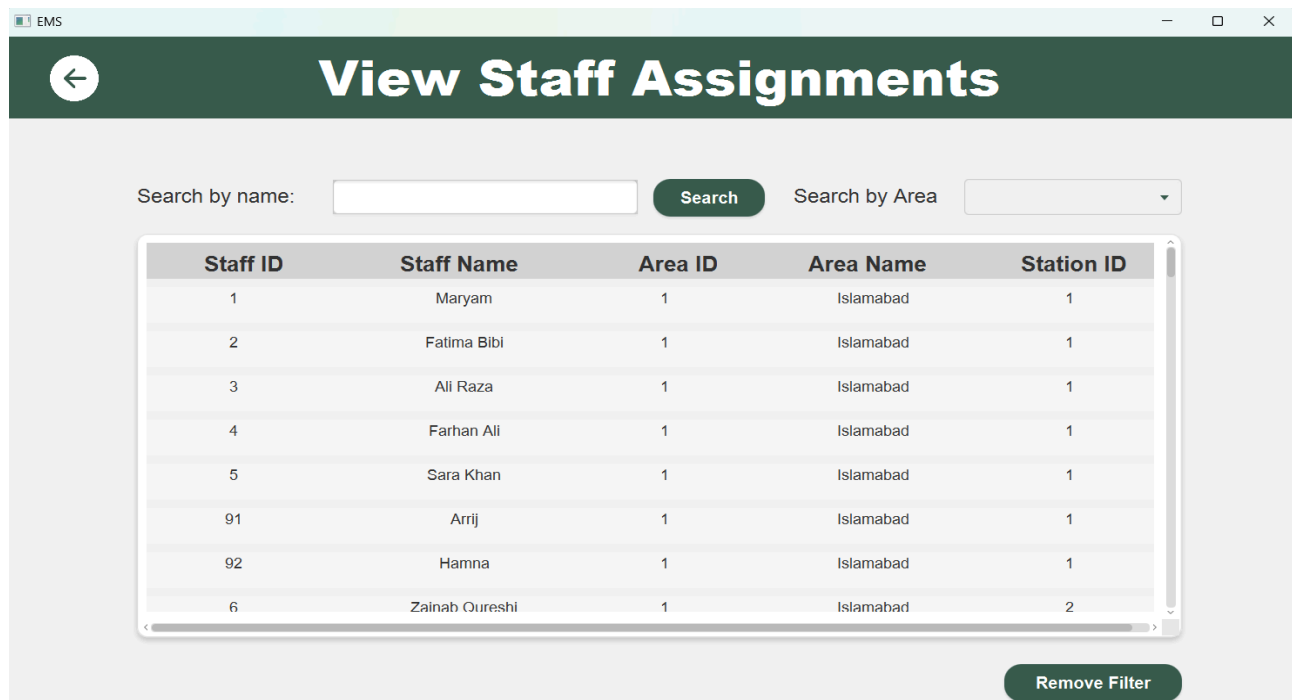
Standard Features:

- **Back Arrow:** A back arrow is present on screens with a hierarchical structure, allowing users to easily return to the previous menu.

Sample Screen (Candidate List View):

- **Table Header:** Displays **Candidate Name**, **Party Name**, **Party Symbol**, and **Vote Count**.
- **Data Row:** Each row dynamically adds candidate data with columns for **Name**, **Party**, **Symbol**, and **Vote Count**.
- **Buttons:** **Add**, **Edit**, **Delete**, and **Back** are visible and easy to access.
- **Navigation:** The **back arrow** allows users to return to the admin menu.

Staff Assignment Interface:

1. **Layout:**

- The main screen is divided into key areas: a **header**, **search inputs**, **staff table**, and **buttons**.
- The **header** includes a title, "View Staff Assignments", and a back arrow icon to navigate back to the admin menu.
- The **staff table** displays staff assignments with columns for Staff ID, Name, Area ID, Area Name, and Station ID, all aligned horizontally in a **VBox** inside a scrollable area.
- The **search** inputs allow for filtering by staff name (via a **TextField**) or area (via a **ComboBox**). A remove filter button clears the filters.

2. **User Interaction:**

- The **back arrow** is interactive, changing the cursor to a hand on hover, and returns the user to the admin menu.
- The **search functionality** allows admins to filter staff based on staff name or area. The search results update the staff table with matching rows.
- The **remove filter button** clears the search filters and reloads the full staff table.

3. **GUI Standards:**

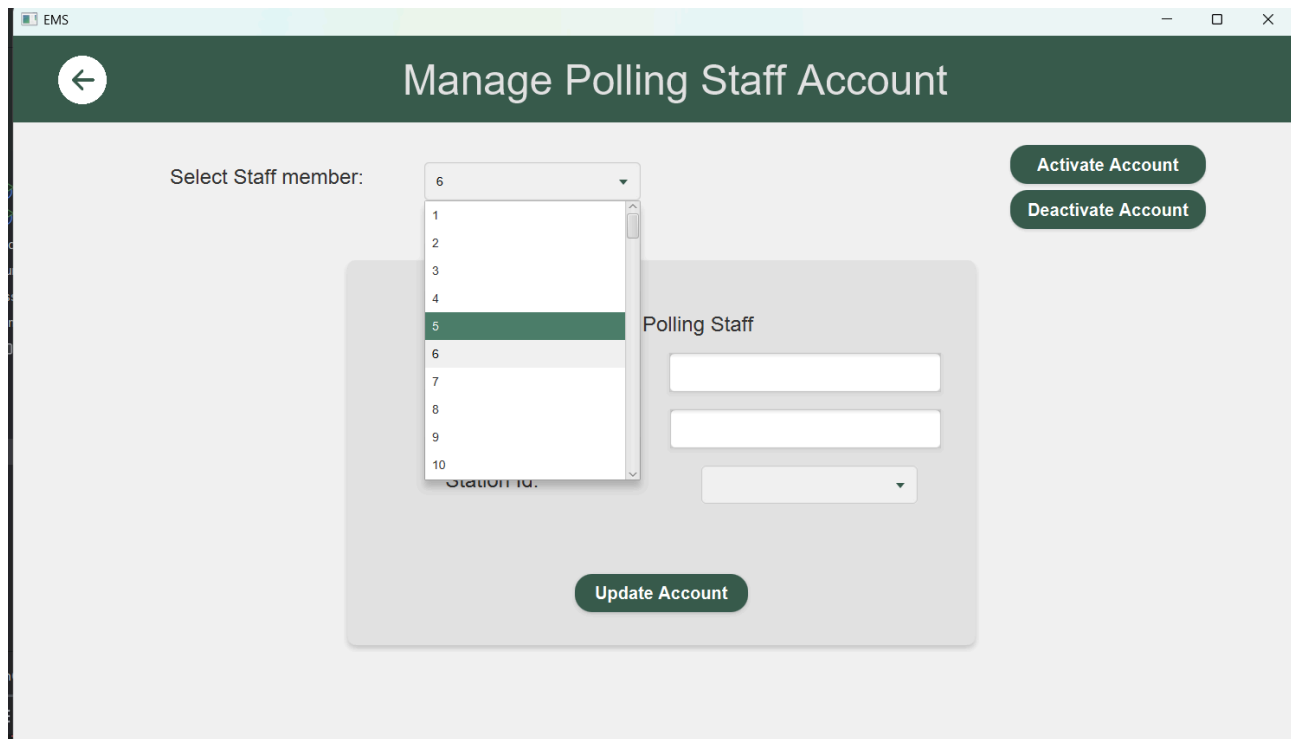
- Standard controls include labels, text fields, buttons, combo boxes, and scrollable areas. The design follows consistent styles with defined spacing between elements for clarity.
- The use of **VBox** and **HBox** ensures a structured layout, while the **table header** maintains alignment consistency for columns.

4. **Feedback:**

- Filtering results appear dynamically as the user interacts with the search fields. The user gets visual confirmation with the update of the staff table.

In terms of **accessibility and usability**, this layout offers clear, easy-to-navigate sections with intuitive controls for filtering and viewing staff assignments. The **back arrow** is prominently placed, offering easy navigation back to the menu.

Update/Deactivate Staff Interface:



The screenshot displays the 'Manage Polling Staff Account' interface. At the top, a dark green header bar contains a back arrow icon on the left and the title 'Manage Polling Staff Account' in the center. Below the header, the main content area is light gray. On the left, there is a 'Select Staff member:' label followed by a dropdown menu showing a list of staff members (1-10). To the right of the dropdown are two buttons: 'Activate Account' and 'Deactivate Account'. Below these is a 'Polling Staff' section with three input fields (username, password, and station ID) and an 'Update Account' button.

Main Screen Layout: The primary screen presents clear options for managing polling staff accounts. A navigation bar at the top includes a "back" button to return to the main menu. Central to the layout is a form section with labeled fields for managing staff details such as username, password, and station ID.

Buttons & Controls: The layout uses standard buttons, including "Update Account," "Activate Account," and "Deactivate Account." The "Update Account" button triggers an action to update staff information. The "Activate" and "Deactivate" buttons enable toggling account statuses. There are also combo boxes to select staff IDs and station IDs. These elements ensure smooth interaction, with well-defined actions for users.

User Feedback: After an action (update, activate, or deactivate), a popup confirms the operation's success or failure. This improves user experience by providing clear, instant feedback. The popup UI is simple, containing a message and an OK button.

GUI Standards: The product follows a consistent and intuitive layout. The use of form fields, buttons, combo boxes, and labels adheres to standard GUI conventions, ensuring familiarity and ease of use for the user.

Navigation Flow: The flow is organized to allow quick transitions between key actions (update, activate, deactivate). The back button at the top helps users navigate easily between screens, while the form section remains the focus for any interaction involving staff account updates.

Style Guides: The product uses a consistent color scheme (soft gray backgrounds, white text, and green buttons), contributing to a professional and cohesive user interface. Font sizes and text placements follow standard design practices for readability.

Election Forms Interface:

EMS

View Election Form

Election Date: 2024-11-26

National / Provisional: National Asse...

Area: Islamabad

View Stations

Select a Station ID

Station ID

1
2
3

Generate Form

Election Form

Candidate Name	Party Name	Votes
Bilawal Bhutto Zardari	Pakistan Peoples Party	80
Shehbaz Sharif	Pakistan Muslim League (N)	133
Shah Mahmood Qureshi	Pakistan Tehreek-e-Insaf	100

Registered Voters: 469

Total Votes: 313

Voter Turnout: 66%

Logical Characteristics of the Election Form Interface

The election form interface provides an interactive and user-friendly GUI for administrative tasks, designed for clarity and accessibility. Below are the logical characteristics, standard features, and guidelines followed in its design:

GUI Standards and Product Family Style Guide

- Color Scheme:**
 - The interface uses a professional color palette, incorporating green gradients for content areas and white backgrounds for text clarity.
 - Text is styled in contrasting colors to enhance readability (e.g., black for primary labels and light gray for titles).
- Font Style and Size:**
 - Headers: Large fonts (48px) for titles such as "View Election Form".
 - Subsections: Medium font sizes (24px) for labels and instructions.
- Screen Layout Constraints:**
 - Left section: Contains navigation tools like the "back" button and contextual inputs (e.g., date and area selection).

- Right section: Displays detailed data in tables and summaries.
- 4. **Standard Buttons and Functions:**
 - Help: Not implemented in this interface but could be added to the top-right corner.
 - Standard Buttons: Generate Form, and Search Area, styled consistently with hover effects.
- 5. **Error Message Display:**
 - Console-based feedback (e.g., "No station selected.") for error handling.
 - Future GUI improvements could integrate pop-up alerts or inline error messages.
- 6. **Keyboard Shortcuts** (Future Feature Scope):
 - Tab navigation between input fields.
 - Enter: Trigger default actions such as search or form generation.

Components of the Interface

1. Election Summary Section:

- **Election Date:** Dynamically displays the current or chosen election date.
- **National/Provincial Assembly Selector:** ChoiceBox for switching between assembly types.

2. Area Search Section:

- **Area Field:** Input for area names.
- **Search Button:** Initiates a lookup for polling stations based on the entered area.

3. Station ID Selector:

- **Station Table:** Displays a list of station IDs fetched based on the search query.
- **Column:** Station ID populated dynamically using PropertyValueFactory.

4. Form Details Section:

- **Table:** Displays fetched election form details, such as candidates, parties, and vote counts.
- **Columns:** Candidate Name, Party Name, and Vote Count.

5. Election Summary Statistics:

- Labels showing registered voters cast votes, and voter turnout percentage.

6. Navigation:

- **Back Arrow:** Returns to the Admin Menu, preserving session data.

Sample Screen Images

Mockup Layout:

1. **Main Interface:**
 - Title bar with "View Election Form".
 - Left-aligned inputs: Date, Assembly Type, and Area.
 - Right-aligned results: Station Table and Election Form details.
2. **Election Form Table:**

- Header Row: Candidate, Party, Votes.
- Aligned data rows for clarity.

Error Handling Standards

- Empty Fields: Show a console warning and prevent form submission.
- Invalid Data: Highlight incorrect inputs (future inline message integration planned).
- No Results Found: Display "No data found for the selected station ID" in a message box.

Software Components Requiring UI

1. **Election Form Display:** TableView for form details.
2. **Station Management:** TableView for station IDs.
3. **Data Inputs:** TextField, ChoiceBox, and buttons for data entry and execution.
4. **Navigation:** ImageView and dynamic scene transitions.

Election Results Interface:

The screenshot displays the 'Election Results' interface. At the top, there's a header with a back arrow and the title 'Election Results'. Below the header, the 'Election Date' is set to '2024-11-26', and the 'National / Provisional' status is 'NA'. There are three input fields for 'Area Name', 'Candidate Name', and 'Party Name', each with a corresponding 'Search' button. Below these, a table lists the election results. The table has four columns: 'Area Name', 'Candidate Name', 'Party Name', and 'Vote Count'. The results are as follows:

Area Name	Candidate Name	Party Name	Vote Count
Islamabad	Bilawal Bhutto Zardari	Pakistan Peoples Party	80
Karachi	Bilawal Bhutto Zardari	Pakistan Peoples Party	80
Peshawar	Bilawal Bhutto Zardari	Pakistan Peoples Party	95
Islamabad	Shehbaz Sharif	Pakistan Muslim League (N)	133
Rawalpindi	Shehbaz Sharif	Pakistan Muslim League (N)	110
Quetta	Shehbaz Sharif	Pakistan Muslim League (N)	90
Lahore	Maryam Nawaz	Pakistan Muslim League (N)	85
Islamabad	Shah Mahmood Qureshi	Pakistan Tehreek-e-Insaf	100
Lahore	Khawaja Saad Rafique	Pakistan Muslim League (N)	75
Lahore	Asad Umar	Pakistan Tehreek-e-Insaf	80
Lahore	Qamar Zaman Kaira	Pakistan Peoples Party	90
Lahore	Hamza Shahbaz	Pakistan Muslim League (N)	85
Lahore	Ali Muhammad Khan	Pakistan Tehreek-e-Insaf	95
Lahore	Nisar Ahmad Khuhro	Pakistan Peoples Party	100
Lahore	Khawaja Asif	Pakistan Muslim League (N)	105

Logical Characteristics of the Election Result Interface

The election result interface for the Votix system concludes the election process by displaying the winners for each area in a user-friendly and intuitive manner. Below is a detailed breakdown of its logical design, including its GUI characteristics:

Key GUI Characteristics

1. Layout and Style:

- **Background Colors:**
 - Primary section: #385B4F (dark green for a formal, professional look).
 - Table section: #FFFFFF (white for clear visibility of data).
 - Header: #385B4F with light-colored text (#E4E4E4) for contrast.
- **Font Styles:**
 - Title: Large and bold (48px), ensuring the "Election Result" heading is prominent.
 - Labels and Table Headers: System default bold font for readability.

2. Standard UI Components:

- **Buttons:**
 - Consistent green background (#385b4f) with hover effects for active feedback.
 - Buttons for searching data:
 - By Area.
 - By Candidate.
 - By Party.
- **Input Fields:**
 - Text fields for entering search criteria (AreaName, CandName, PartyName) with clear placeholder prompts.
- **Table:**
 - Columns for Area Name, Candidate Name, Party Name, and Vote Count, ensuring clarity in data presentation.
- **Dropdown Menu:**
 - Allows users to select between "National Assembly" or "Provincial Assembly" results.

3. Standard Buttons and Functions:

- **Help Button:** Not present in the provided code, but could be added for guidance.
- **Back Arrow Button:** Allows users to return to the main menu.
- **Clear Functionality:** Resets all search fields to default, implemented in clearSearches.

4. Error Messaging Standards:

- Console logging (System.out.println) for debugging purposes during search operations.
- In the final implementation, user-friendly error messages could replace these logs

Interface Functionality

1. Search Options:

- Users can filter results by area, candidate, or party using dedicated text fields and buttons.
- The system performs partial matching and updates the table dynamically.

2. Table View:

- Displays election results, including:
 - **Area Name:** Represents electoral regions.
 - **Candidate Name:** Lists candidates in the area.
 - **Party Name:** Shows associated political parties.
 - **Vote Count:** Indicates the number of votes received.
- Dynamically updates based on search criteria.

3. Final Display:

- Summarizes results by highlighting winners in each area after searches are complete.
 - The table updates based on the selected "National/Provincial" filter.
4. **Navigation:**
- Back navigation provided via the arrow button (backArrow).

Logical Flow for Displaying Election Results

1. **Initialization:**
 - Upon loading, the interface fetches all election results and populates the table.
2. **Search-Based Updates:**
 - Users can search for results based on:
 - Area (handleAreaSearch).
 - Candidate (handleCandSearch).
 - Party (handlePartySearch).
 - Each search filters and updates the table content dynamically.
3. **Displaying Winners:**
 - After conducting searches, the interface highlights the winning candidate for each area based on the maximum vote count.
 - This logic could be implemented by sorting the results within ems and dynamically marking the winner rows in the table (e.g., bold font or green row highlight).
4. **Clear Searches:**
 - Resets all inputs and displays the complete election result dataset.

Future Enhancements

- **Keyboard Shortcuts:** Add shortcuts for quick access (e.g., Ctrl+F for "Search by Candidate").

This interface efficiently concludes the election process by visually summarizing results and providing flexible search capabilities, ensuring a seamless user experience.

Election Area Report Interface:

EMS

Election Area Report

Election Date: 2024-11-26

National / Provisional: National Asse... ▾

Enter Area: Islamabad **Search**

Winning Candidate

Name: Shahbaz Sharif

Party: Pakistan Muslim League (N)

Total Votes: 133

Losing Candidate

Name: Shah Mahmood Qureshi

Party: Pakistan Tehreek-e-Insaf

Total Votes: 100

Summary

Area Name: Islamabad

Total Registered Voters: 1723

Total Votes Cast: 1149

Male Votes Cast: 689

Female Votes Cast: 460

Voter Turnout: 66%

Logical Characteristics of the User Interface

1. Screen Layout and Standards

- **Header Section:**
 - A header pane with a back arrow on the left allows the user to return to the main menu.
 - The title "Election Area Report" is centrally displayed in a large font size for clear identification of the screen.
 - A decorative line separates the header from the main content area.
- **Content Section:**
 - A white background ensures readability.
 - Fields are categorized into logical groups:
 - **Election Information:** Displays the election date and options to filter data (e.g., National/Provisional level, area input).
 - **Candidate Information:** Displays the winning and losing candidates' details (name, party, votes).
 - **Summary Information:** Shows key statistics for the selected area (e.g., voter turnout, total votes cast).

2. Standard Buttons and Components

- **Back Arrow:** A clickable image that navigates to the previous screen.

- **Search Button:** Located near input fields, styled with a green background (#385b4f) and white text. Initiates a search based on user input.
- **ChoiceBox and TextField:**
 - The ChoiceBox allows dropdown selection for election type.
 - Text fields are provided for entering and displaying data like area name, winning candidate details, etc.

3. Navigation and Interactivity

- Mouse events trigger specific actions (e.g., onMouseClicked for back navigation or data search).
- Key areas of interactivity include:
 - Input fields for entering search criteria.
 - Display fields for showing report data.
 - Navigation back to the menu.

4. Error Handling and Feedback

- Input validation can ensure fields like "Enter Area" are not empty.
- Placeholder texts guide users on what information to provide (e.g., "Winner Name," "Enter Area").
- Error messages should be displayed in a user-friendly format near the respective fields when incorrect or insufficient input is provided.

5. Visual Standards

- **Font Styles:**
 - Large bold fonts for headers (e.g., "Summary").
 - Standardized sizes for labels and text fields.
- **Color Palette:**
 - Green-themed buttons and text for headers to align with a consistent style.
 - White background for sections to enhance contrast.

6. Keyboard Shortcuts

- Future enhancements could include:
 - **Tab navigation** for moving between input fields.
 - **Enter key** to trigger the search.

Election Report Overview

The election report provides a comprehensive summary of the winning and losing candidates for a specific area:

1. **Winning Candidate Details:**
 - Name, party, and total votes received are displayed.
2. **Losing Candidate Details:**
 - Name, party, and total votes received are similarly shown.
3. **Area Summary:**
 - Key statistics include:
 - Total registered voters.

- Total votes cast.
- Male and female voter participation.
- Voter turnout percentage.

Description of Files

FXML File:

The FXML file defines the layout of the "Election Area Report" screen using JavaFX. It includes:

- A header section for navigation and title.
- Grouped input fields, buttons, and display areas for interacting with and viewing the report.
- Styles are applied inline and through external stylesheets (style.css).

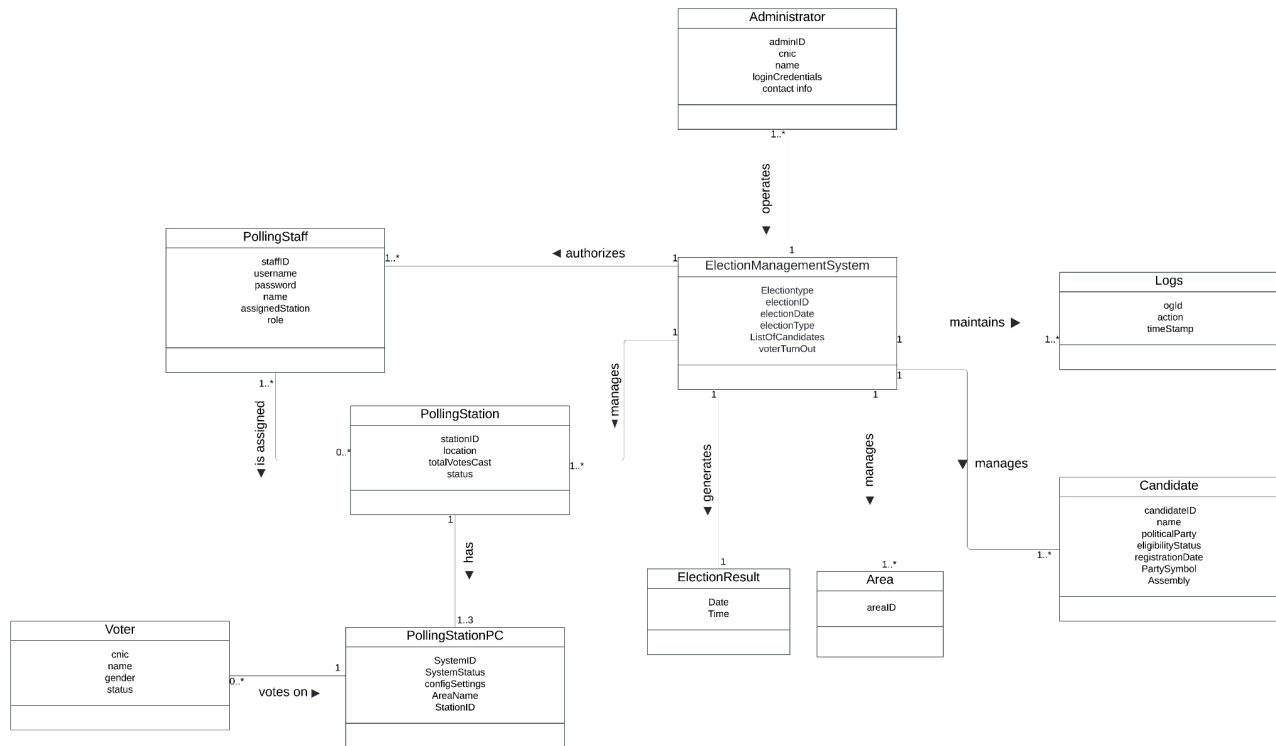
Controller Class (ElectionReportController):

This Java class manages interactions between the UI and underlying logic:

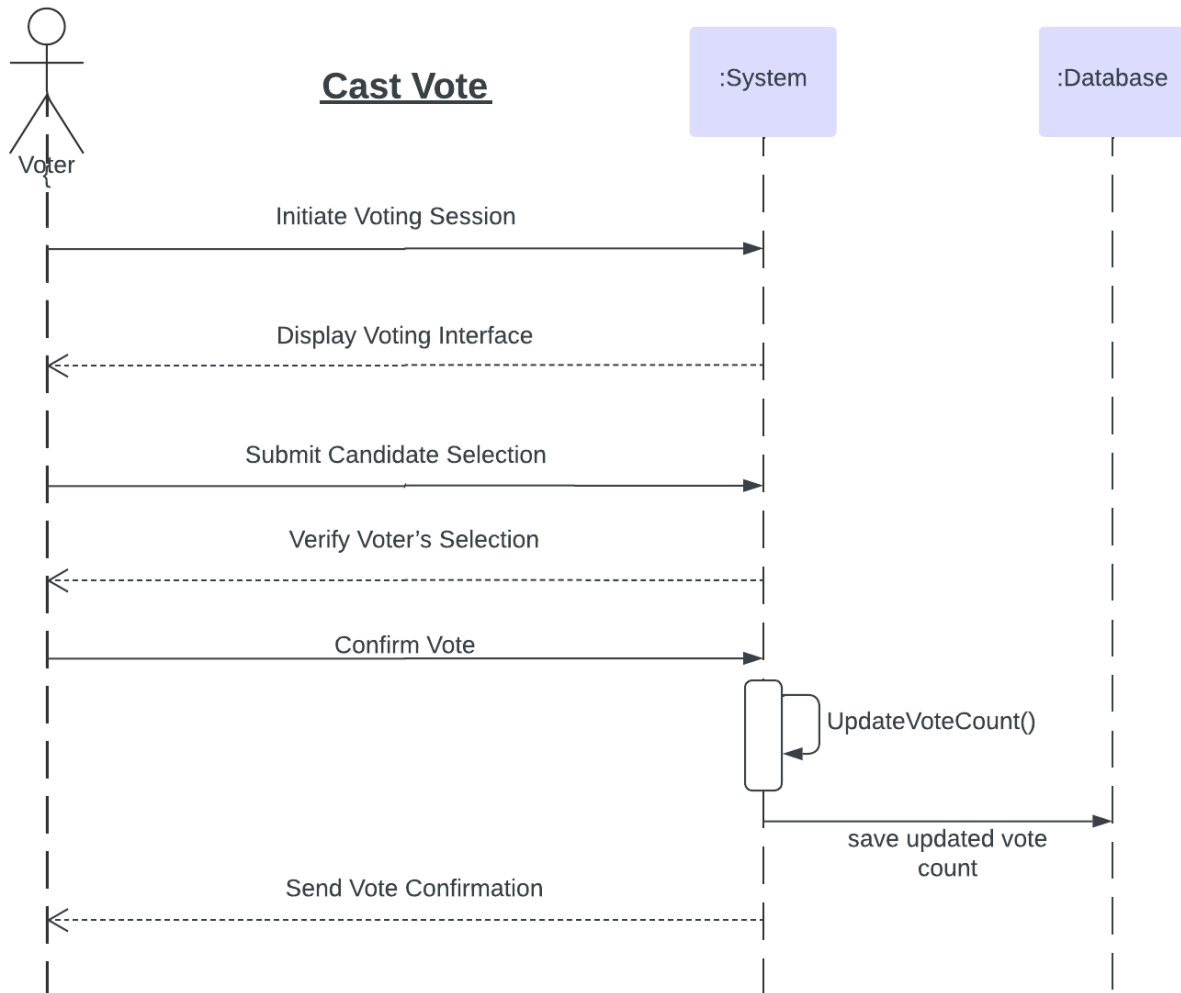
- Handles user actions like search and back navigation.
- Retrieves and displays election data from services like AdminElectionManagementSystem.
- Updates fields dynamically based on user input or fetched results.

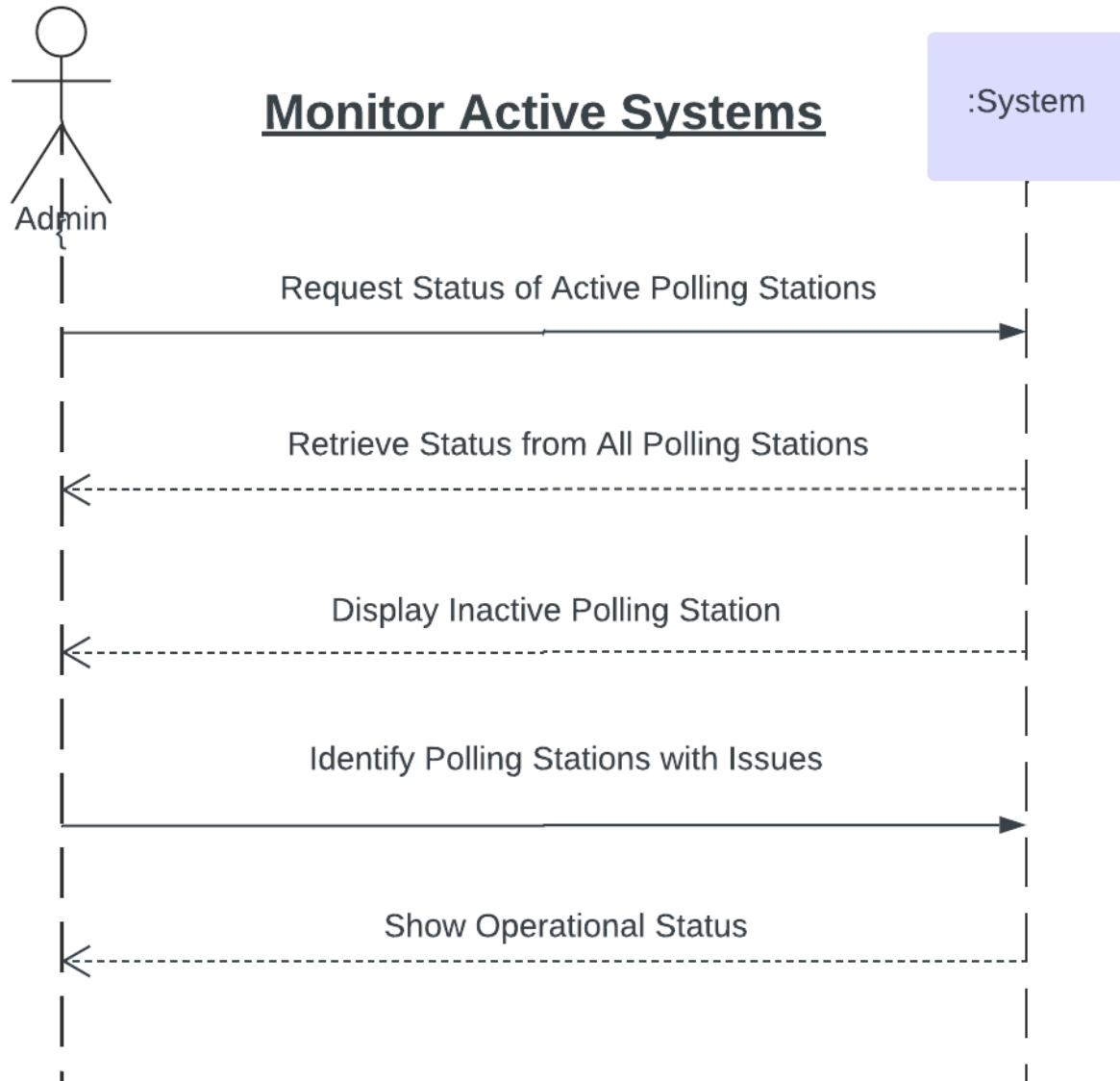
This structured UI design promotes clarity, user-friendliness, and an aesthetic layout for generating and viewing election reports.

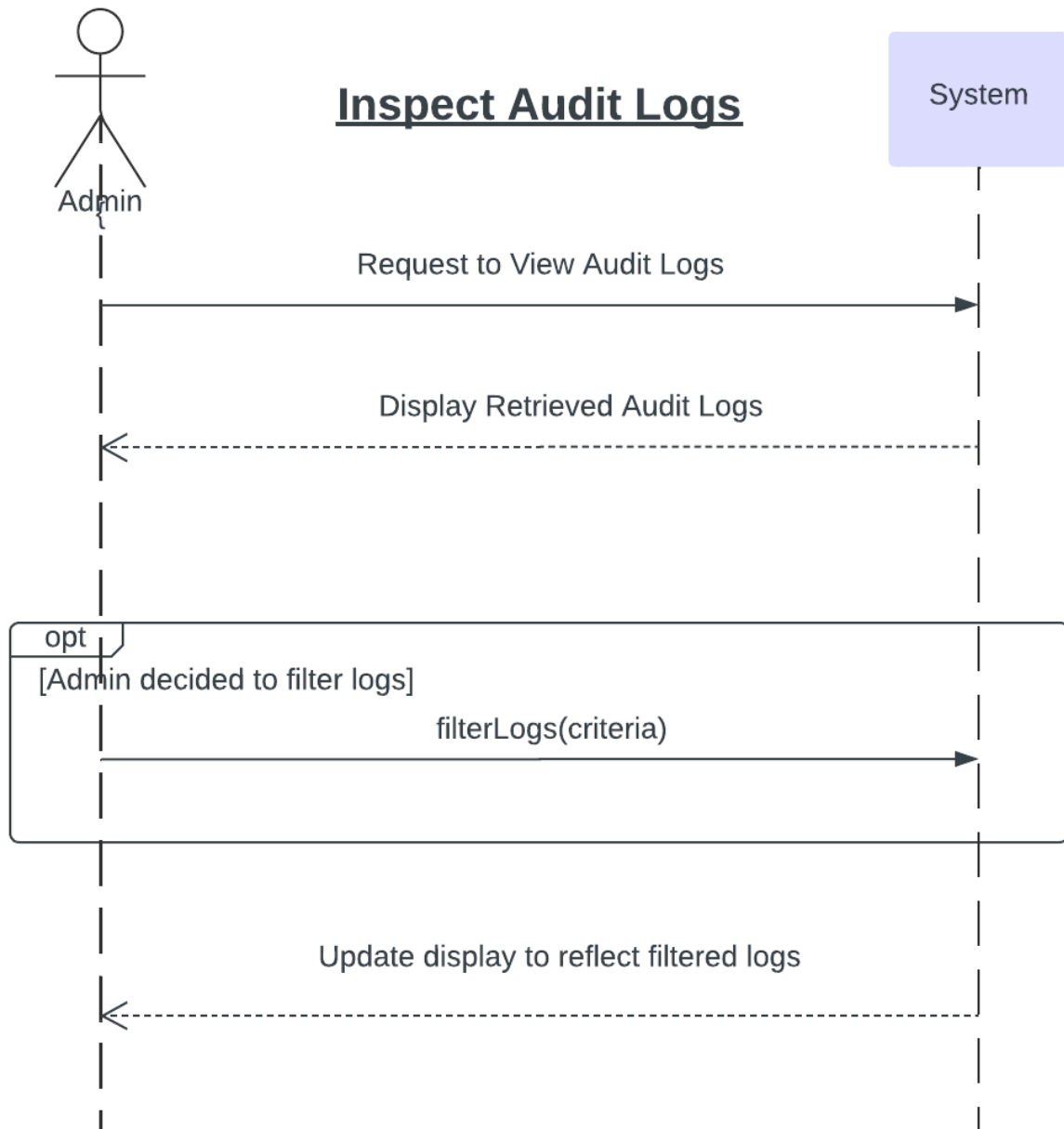
4. Domain Model

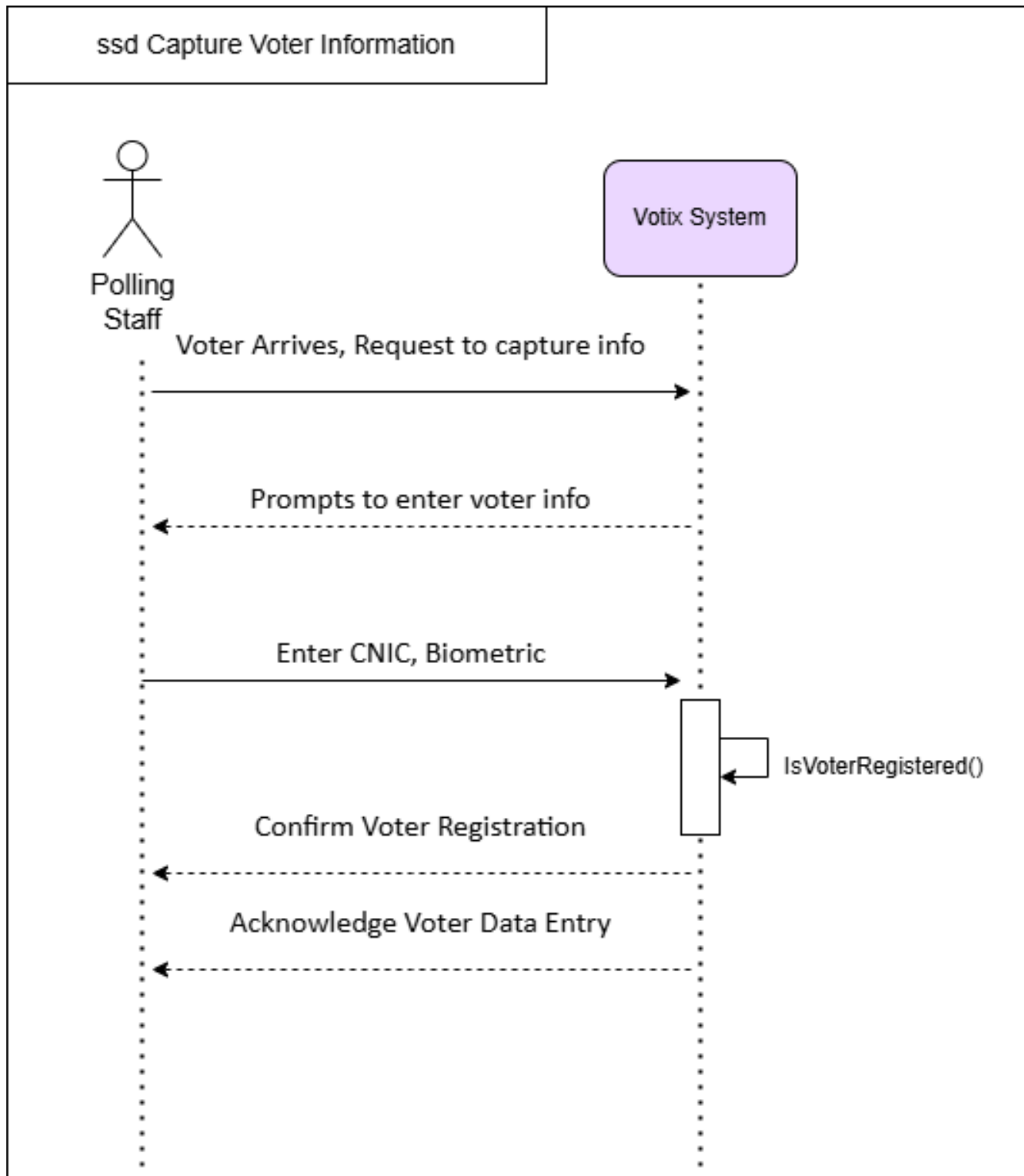


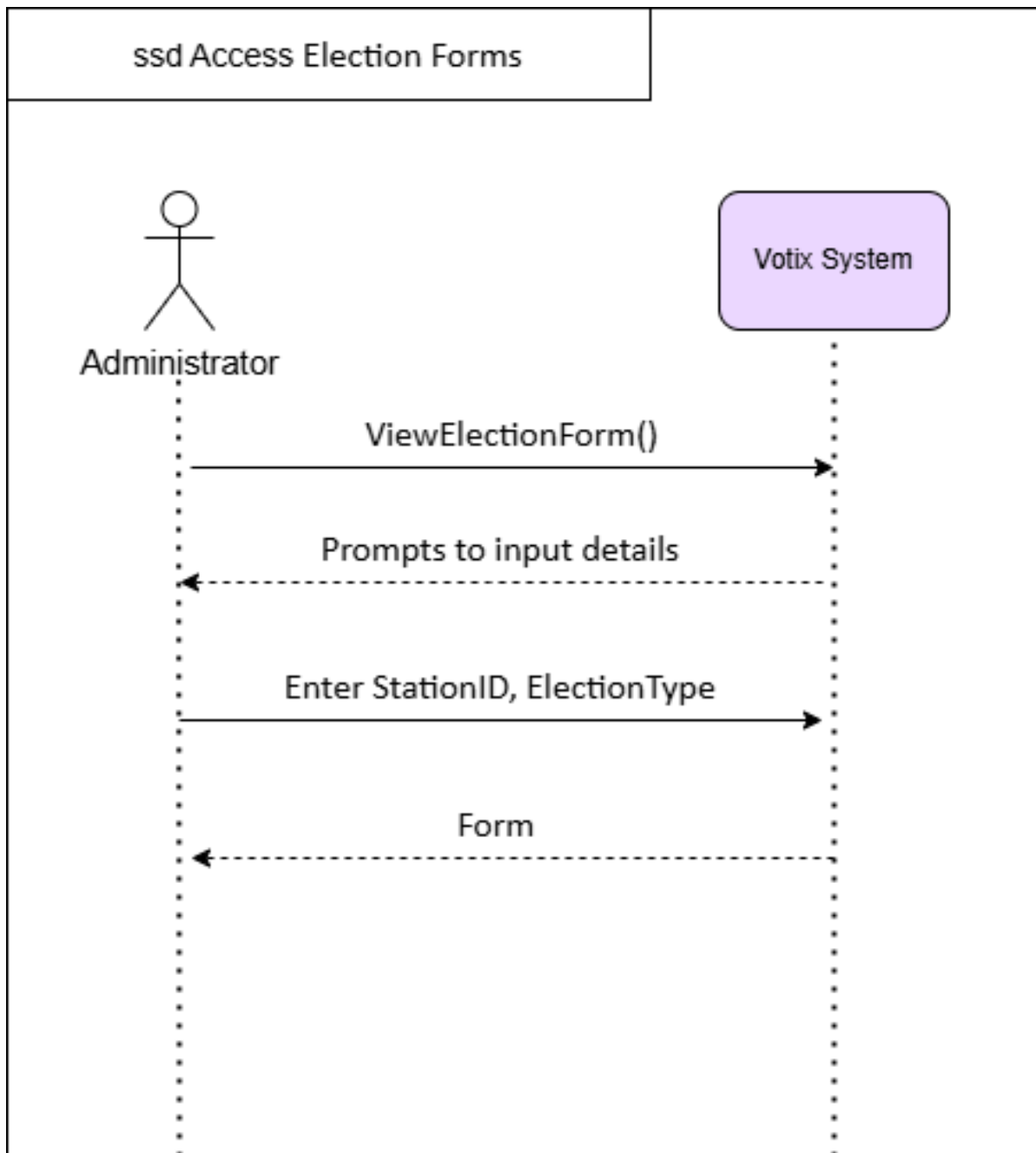
5. System Sequence Diagram

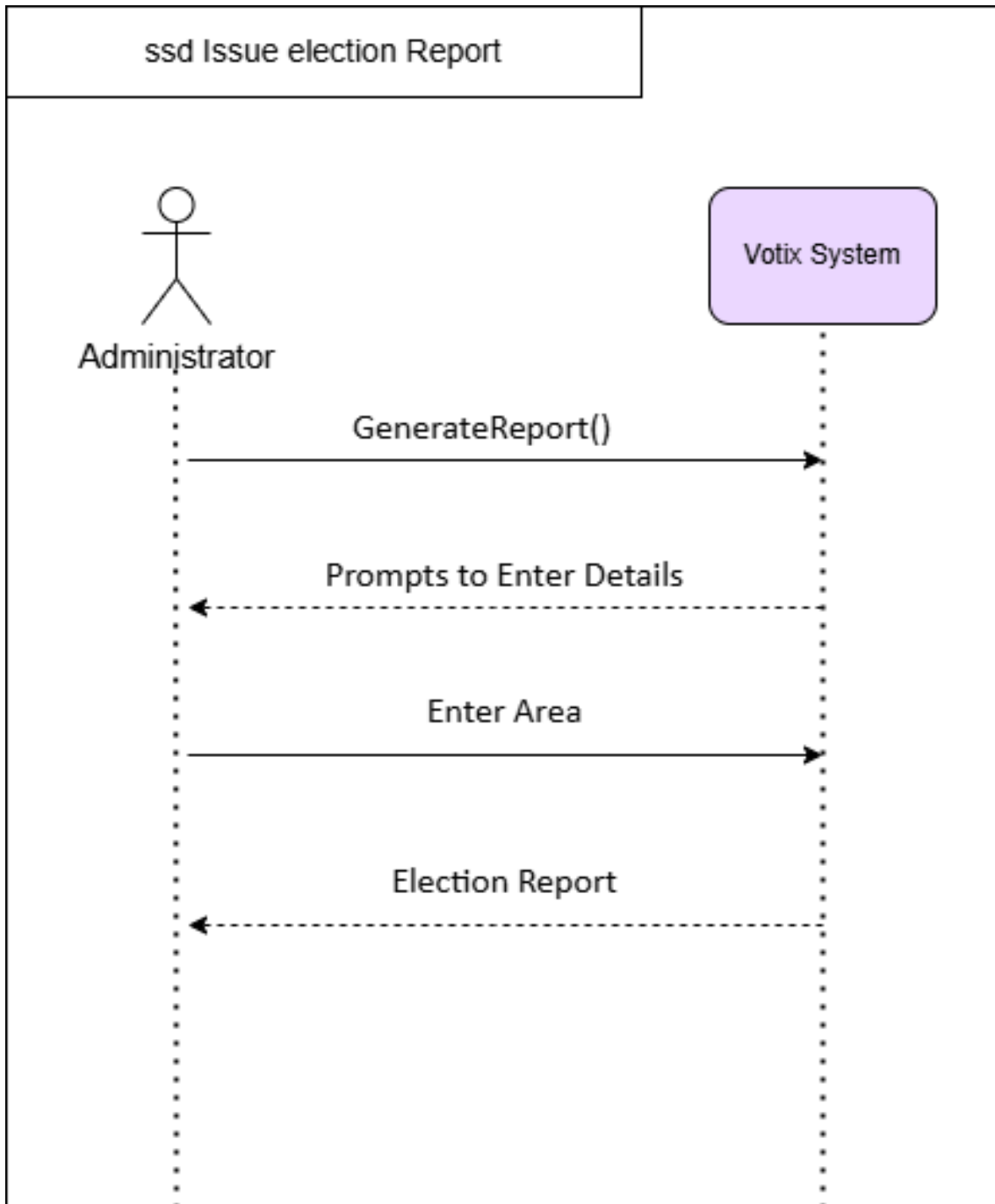


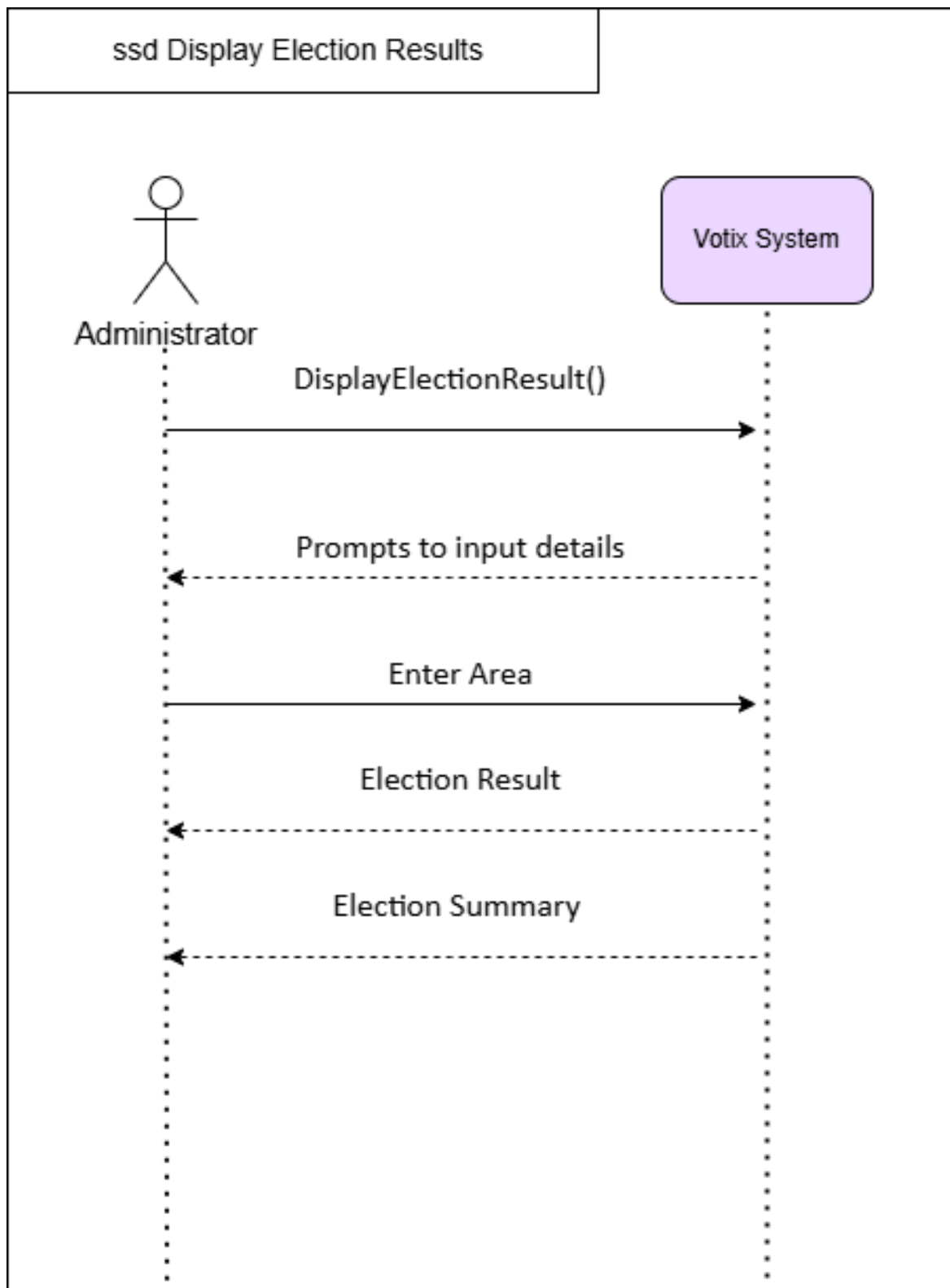


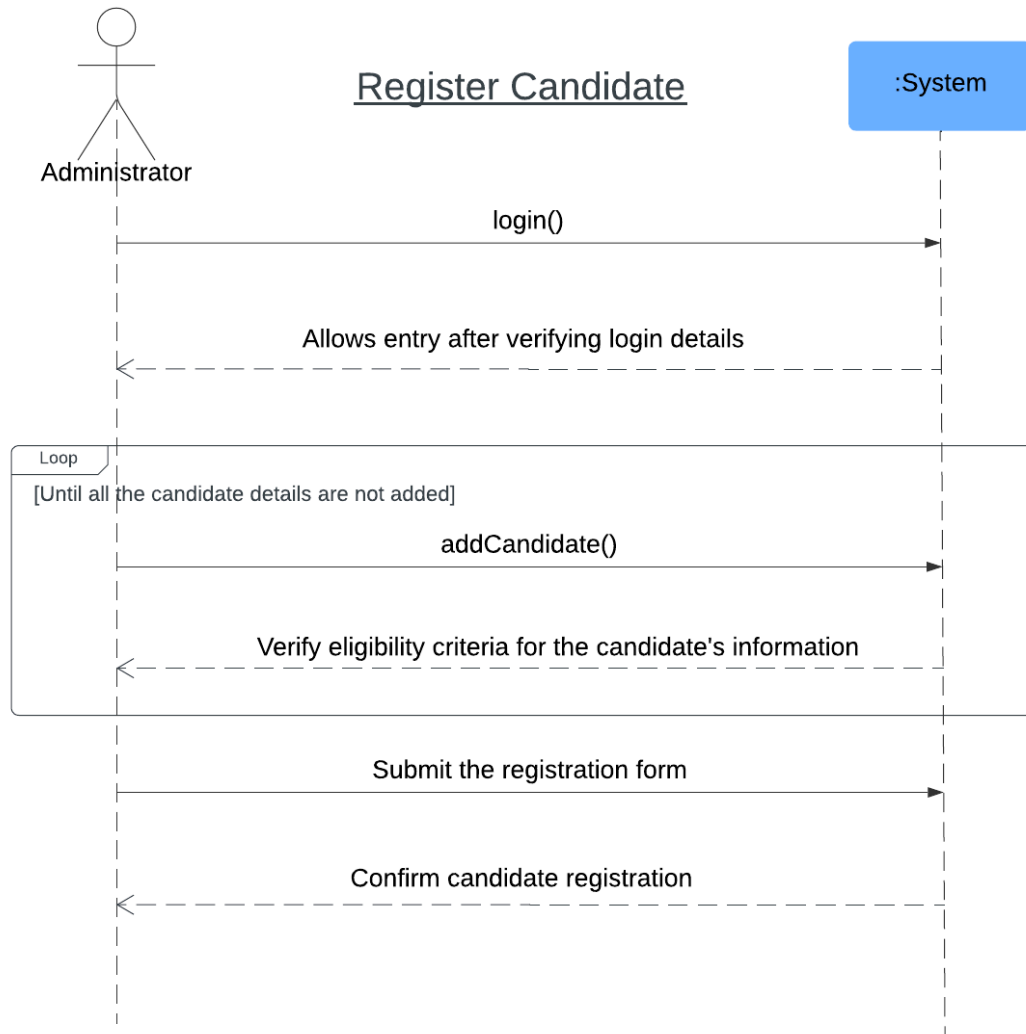


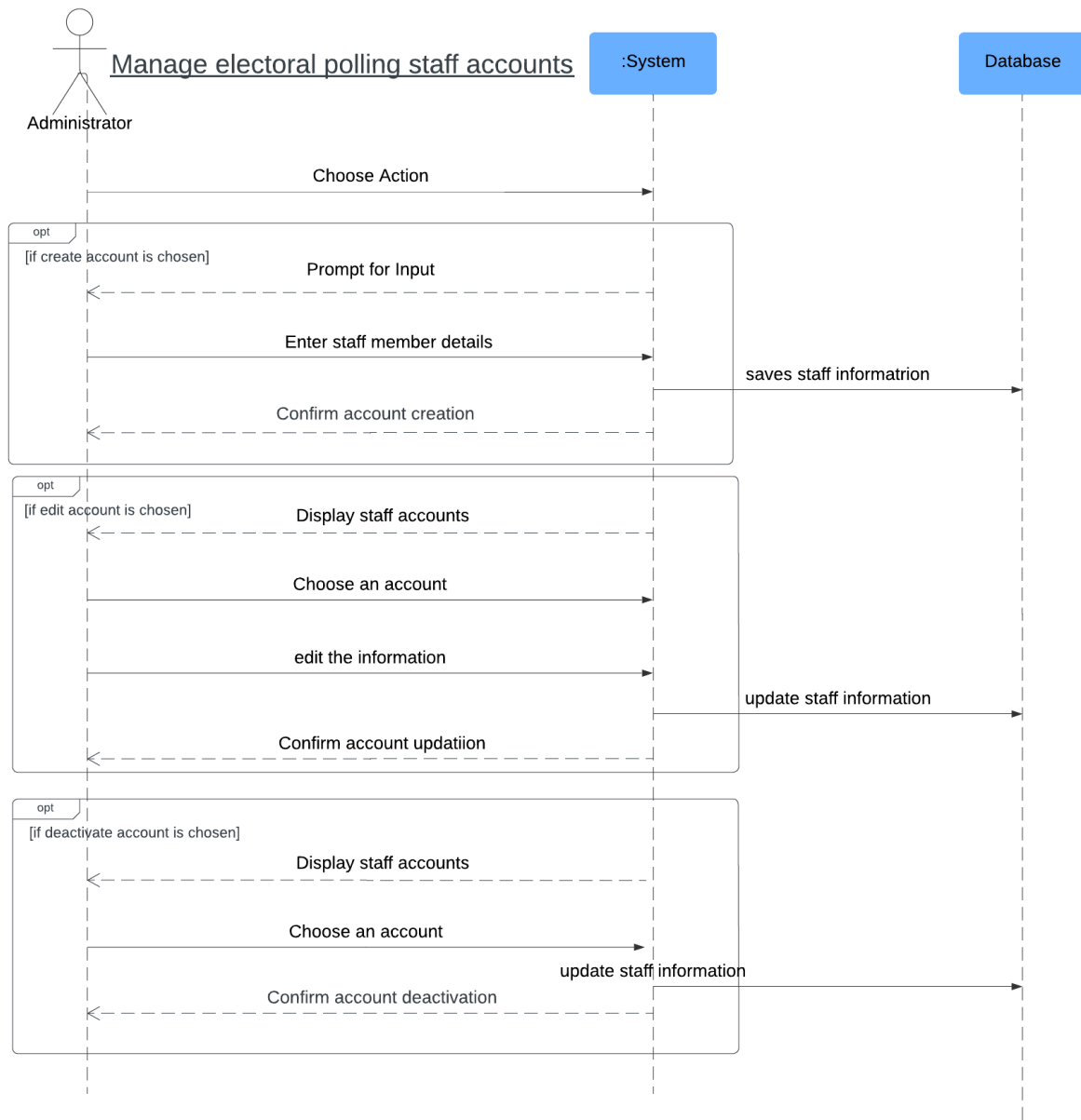


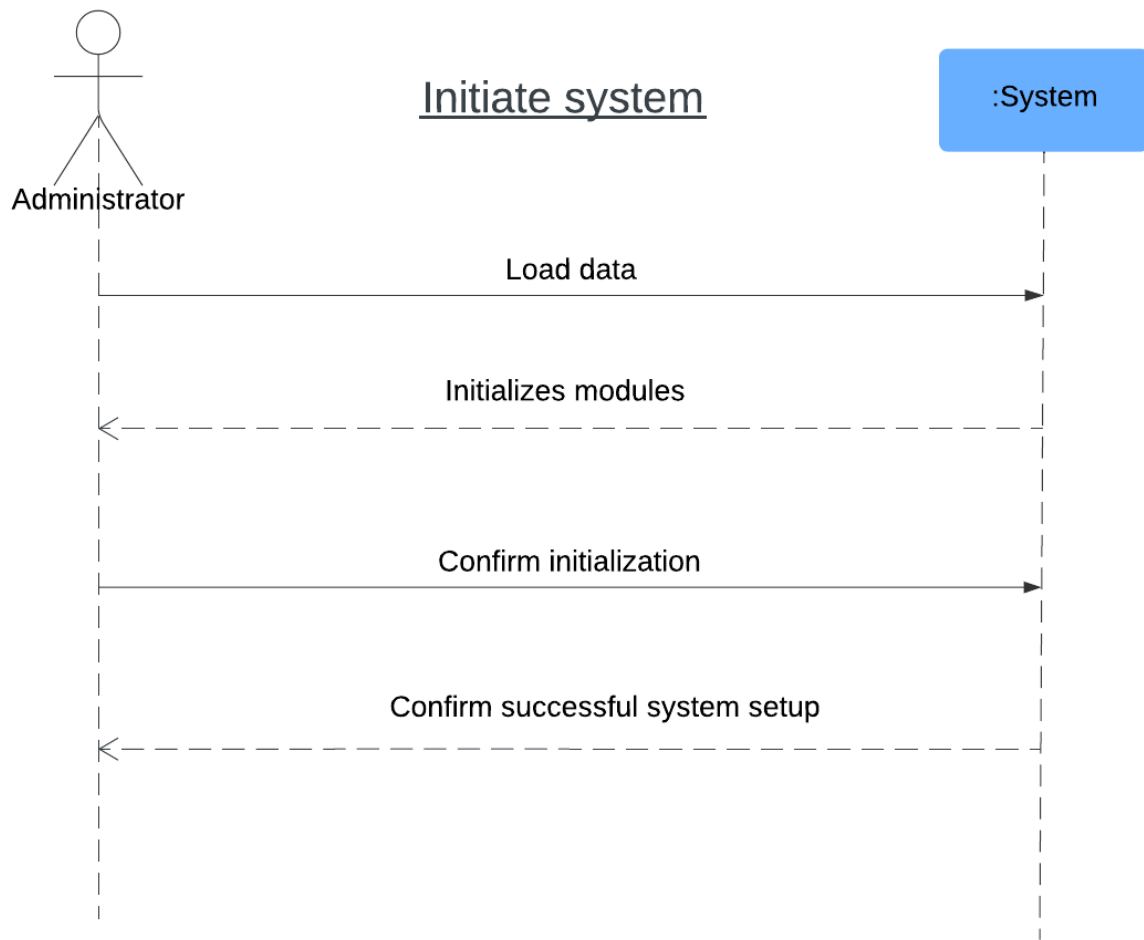


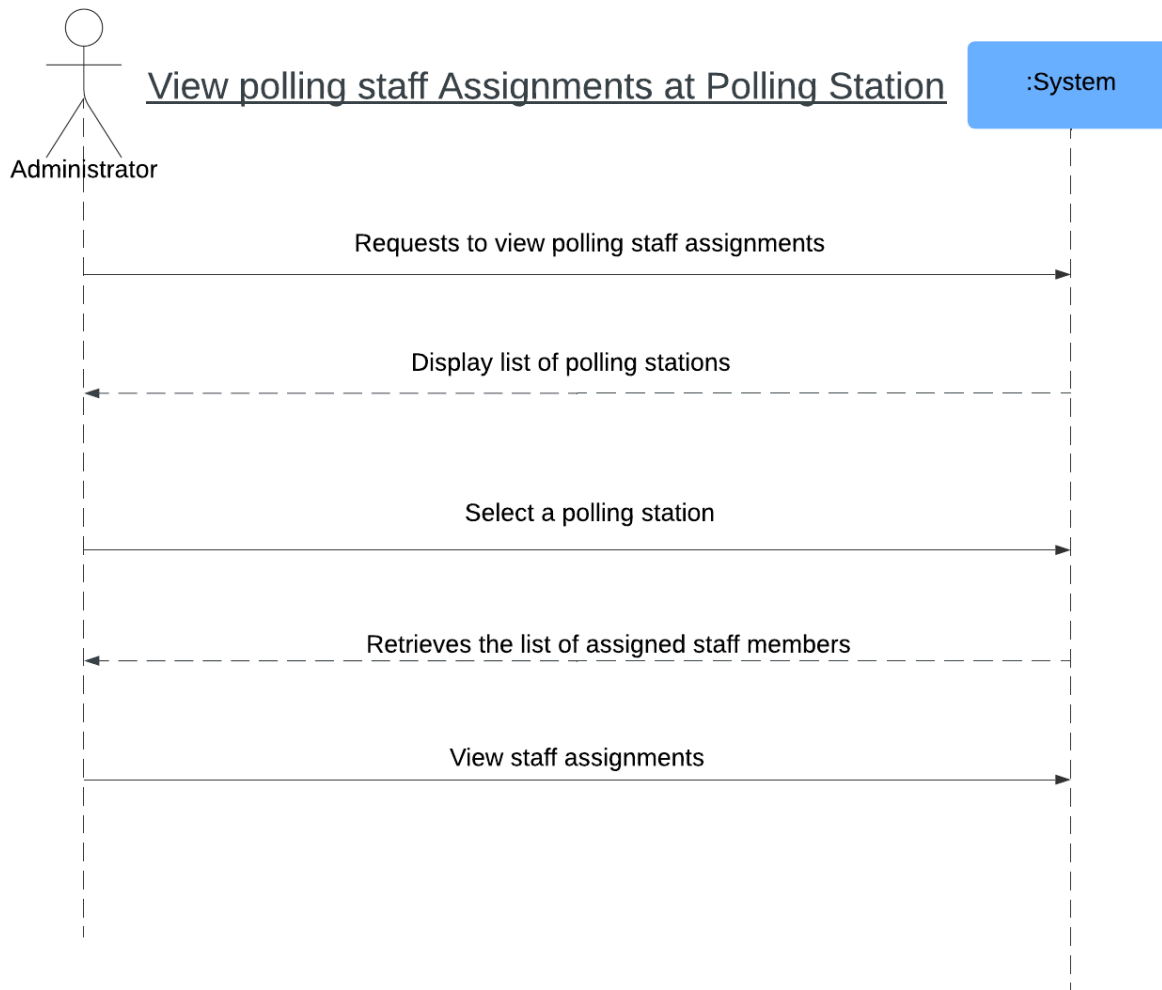


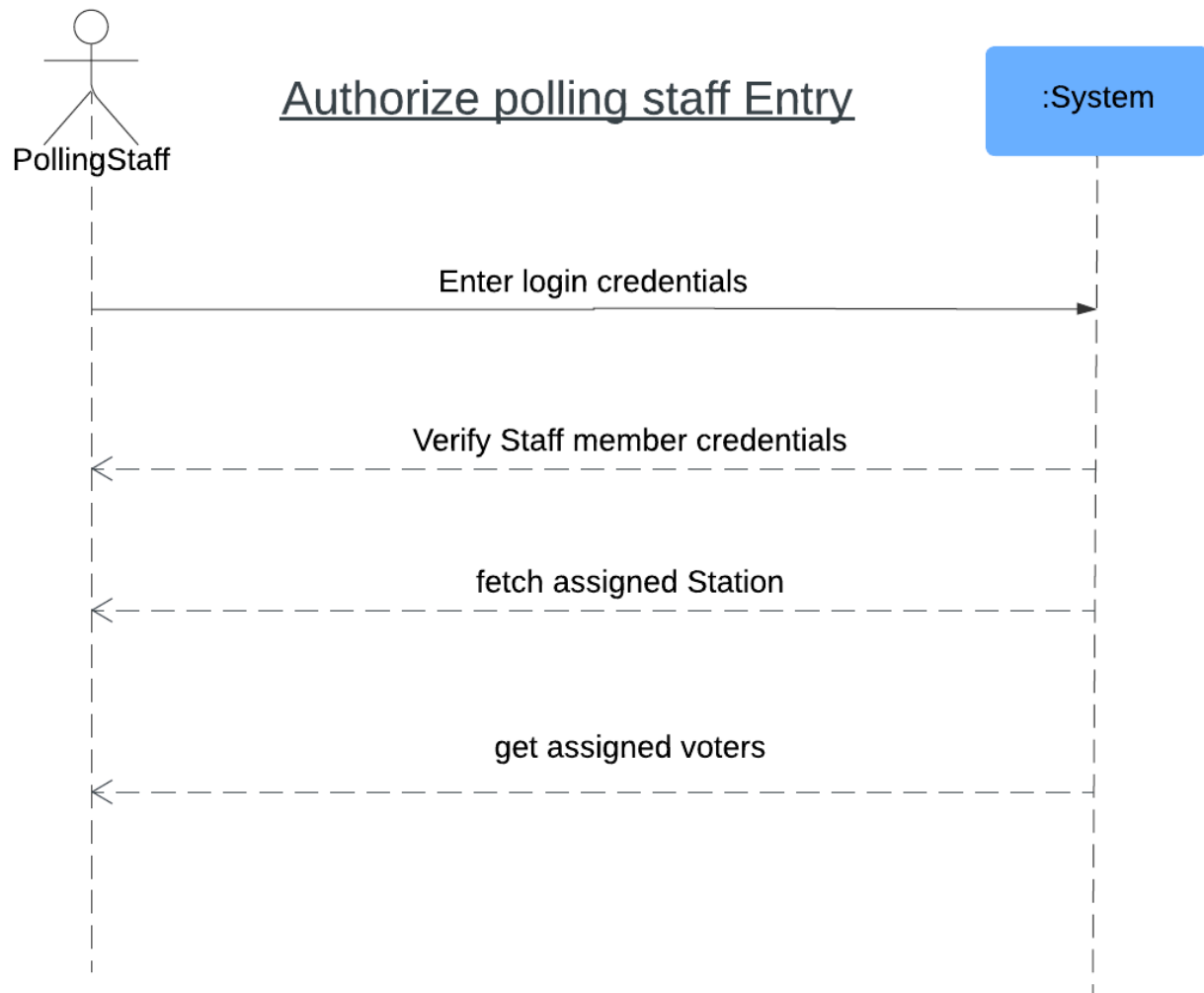




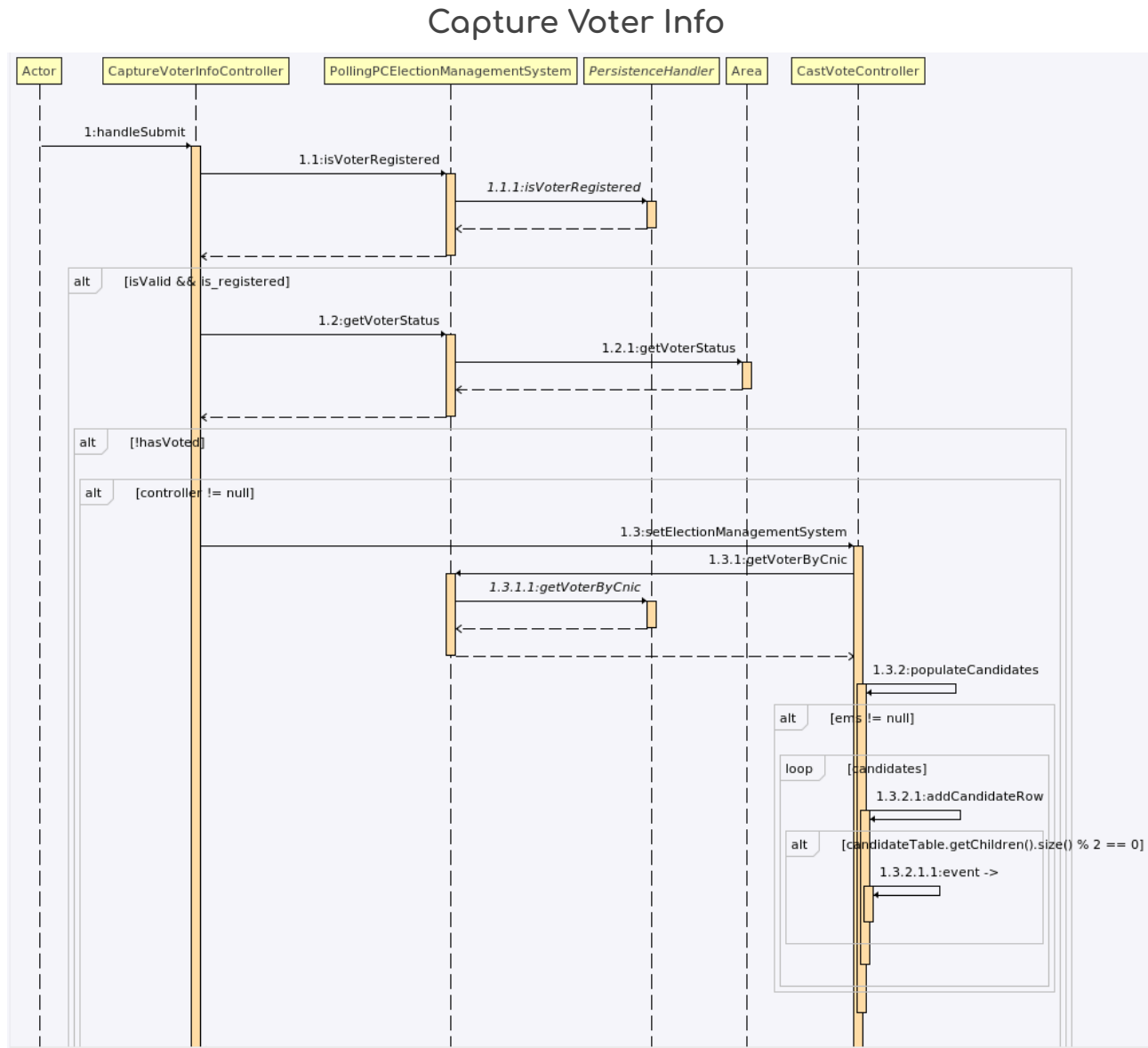




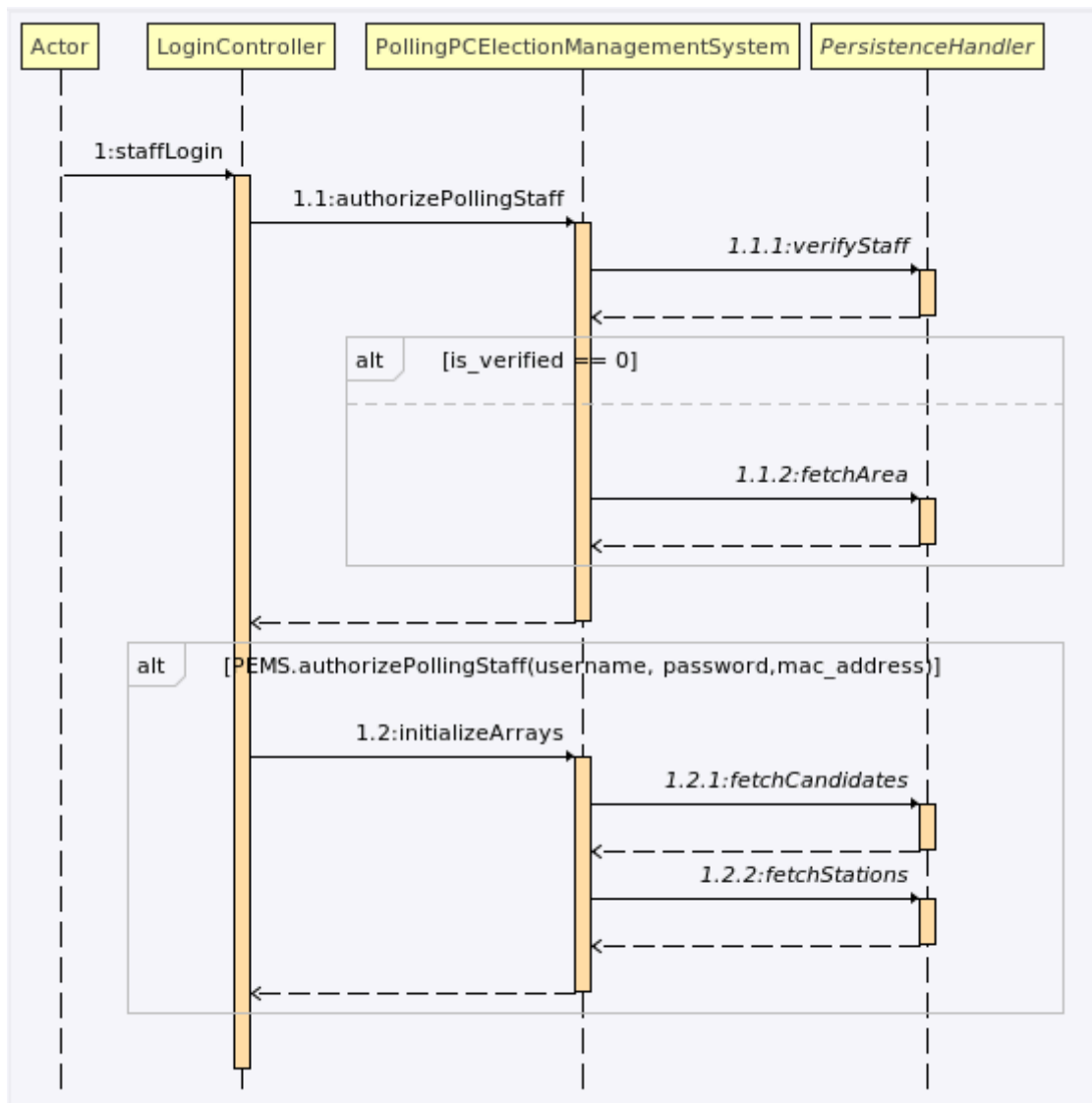




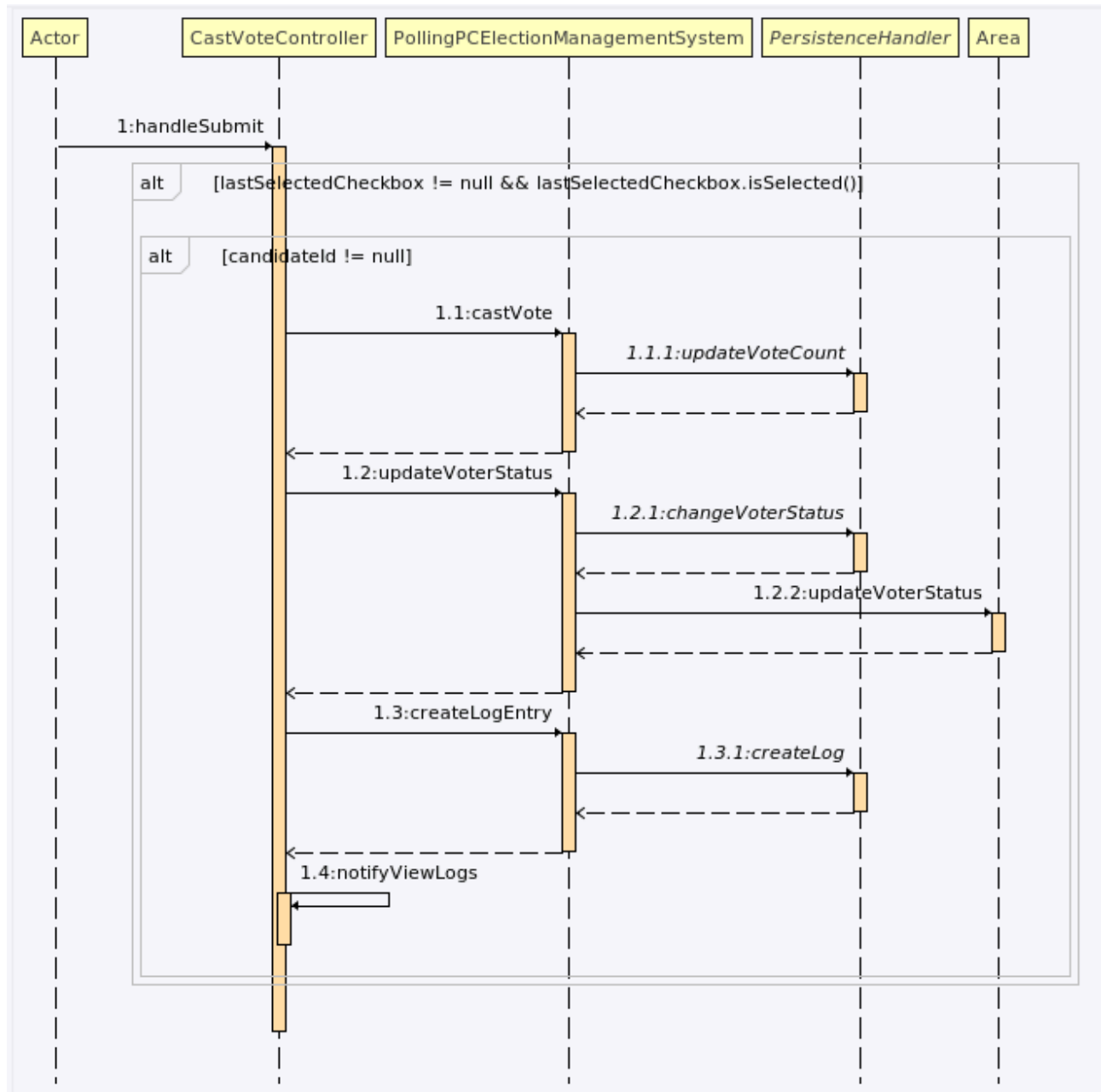
6. Sequence Diagrams



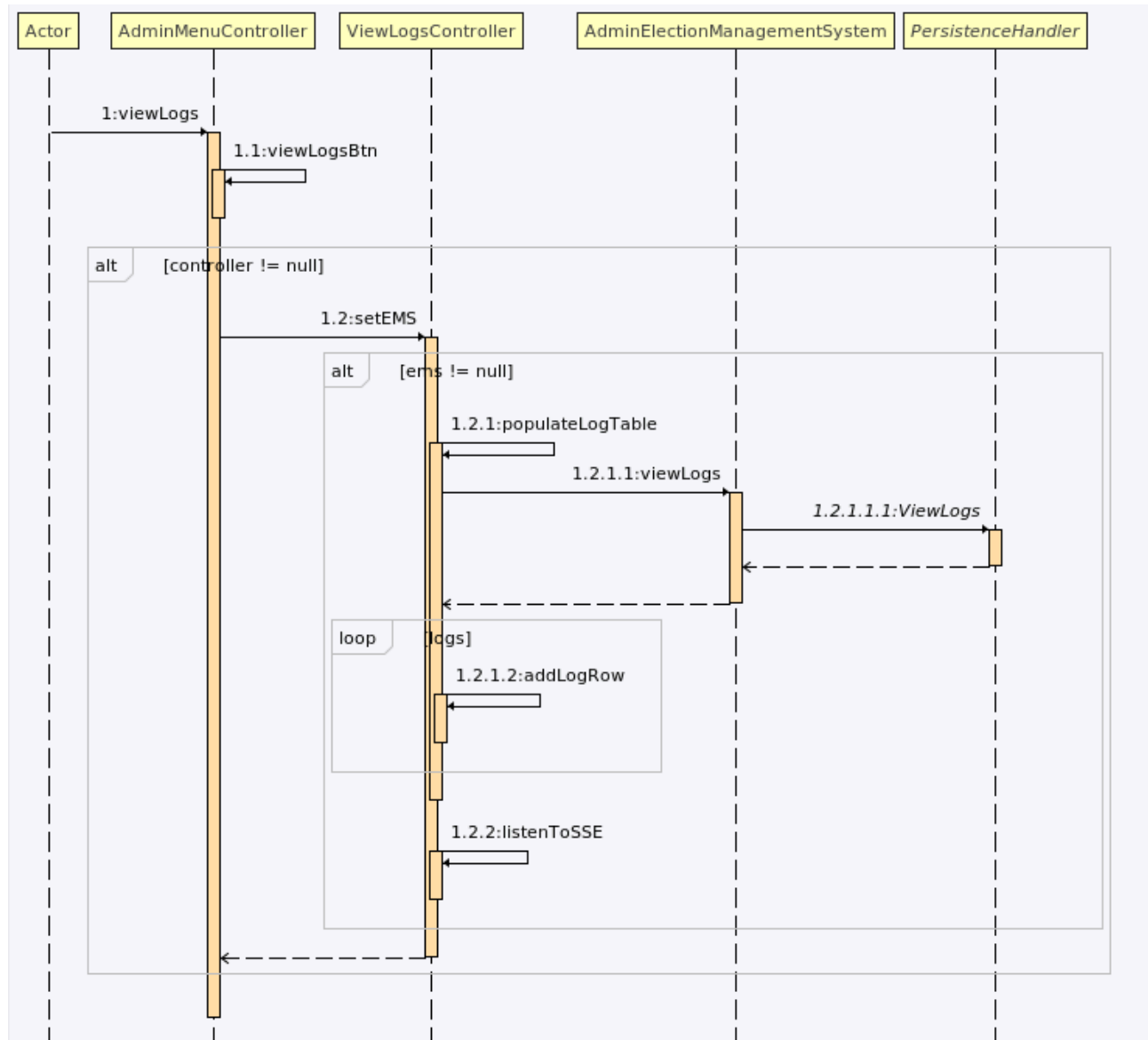
Authorize Polling Staff



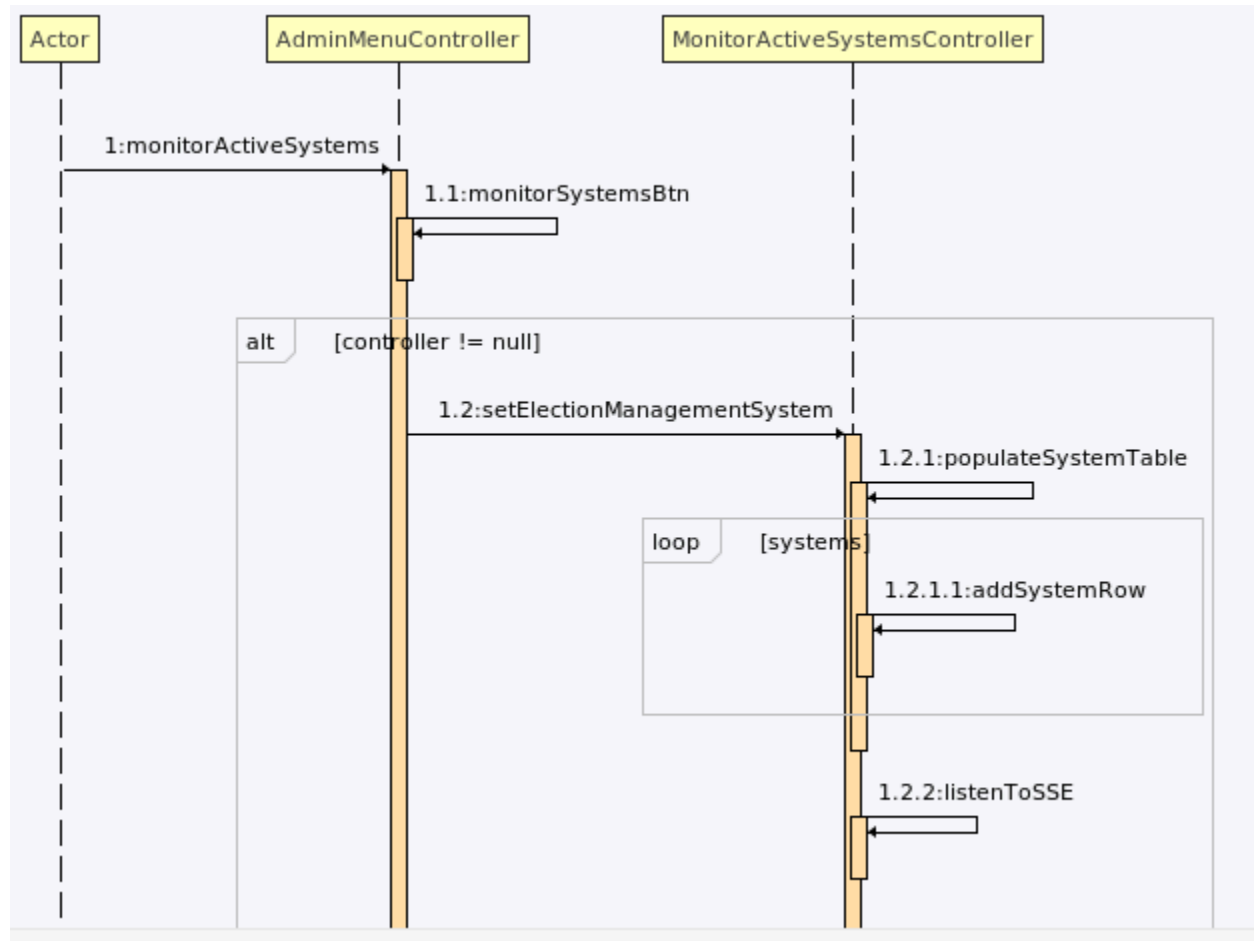
Cast Vote



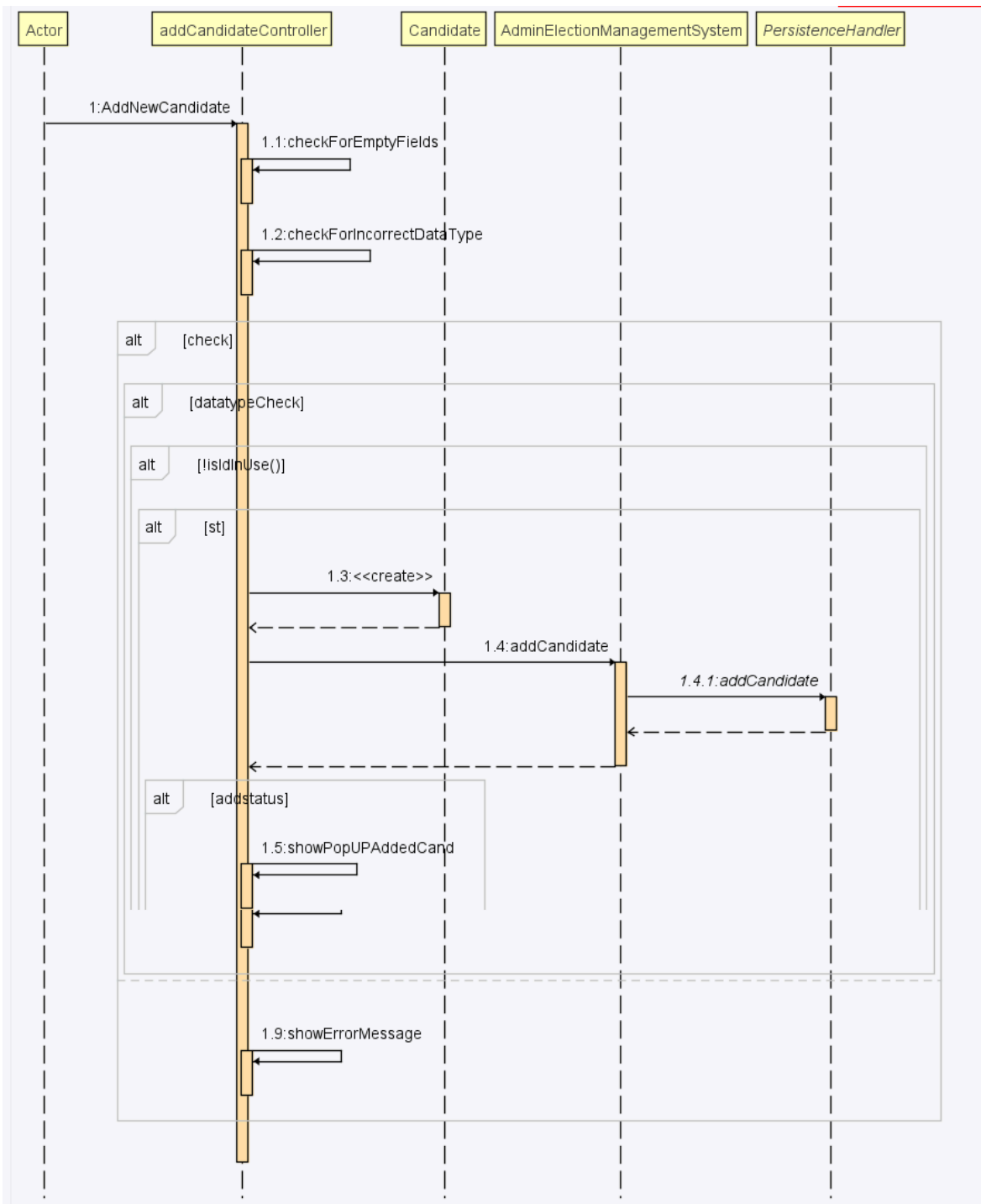
View Logs



Monitor Systems

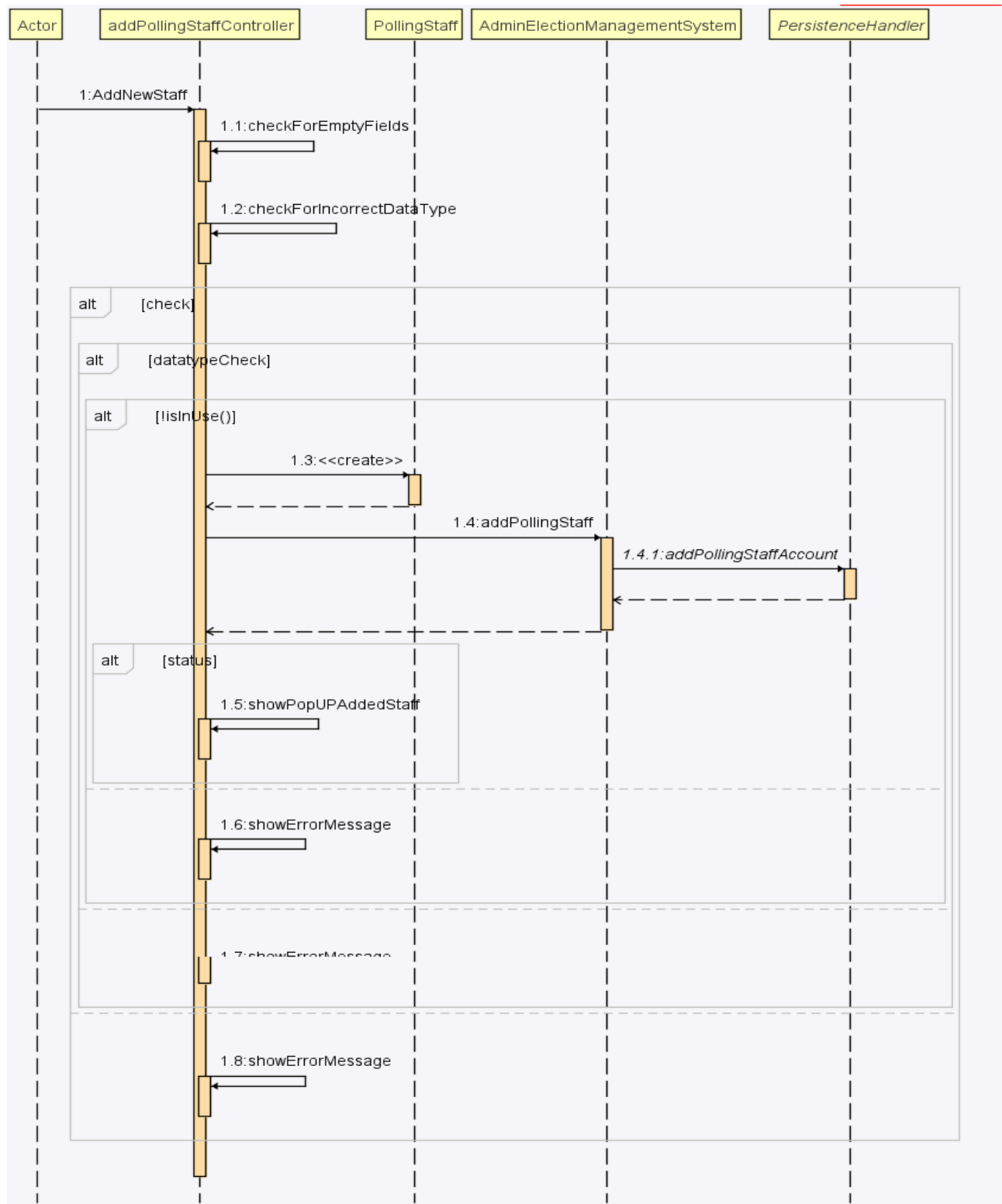


Add Candidate

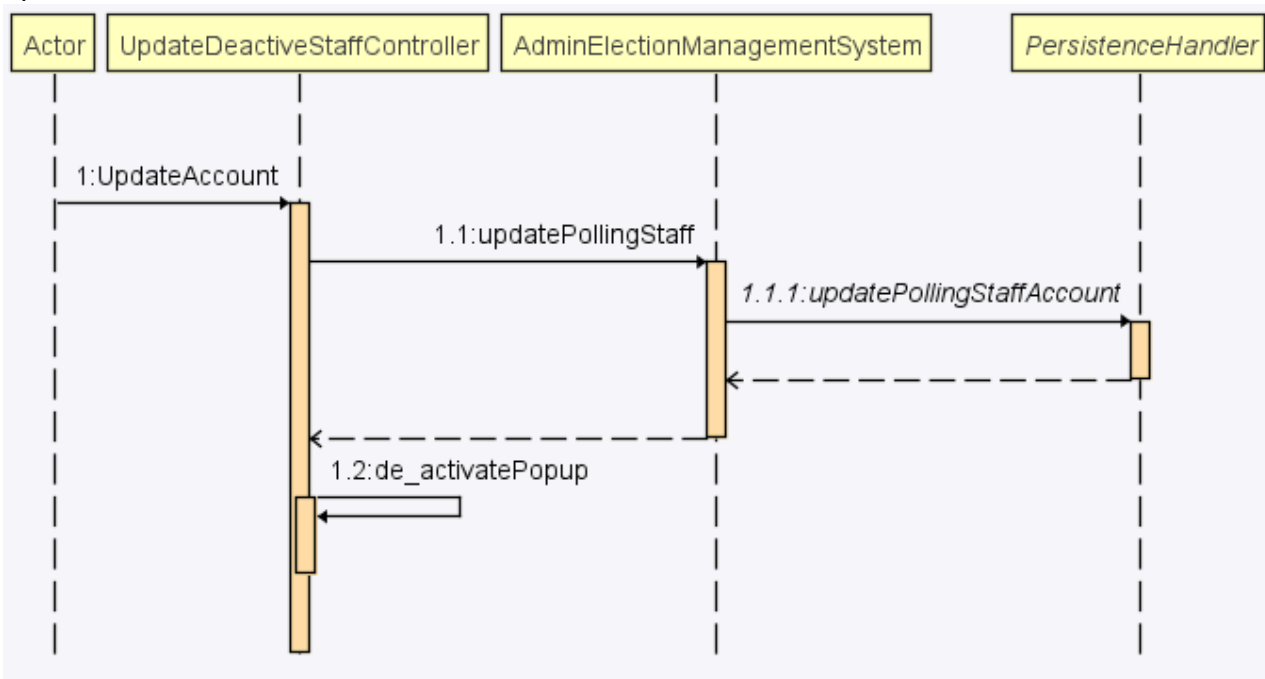


Manage Staff Account

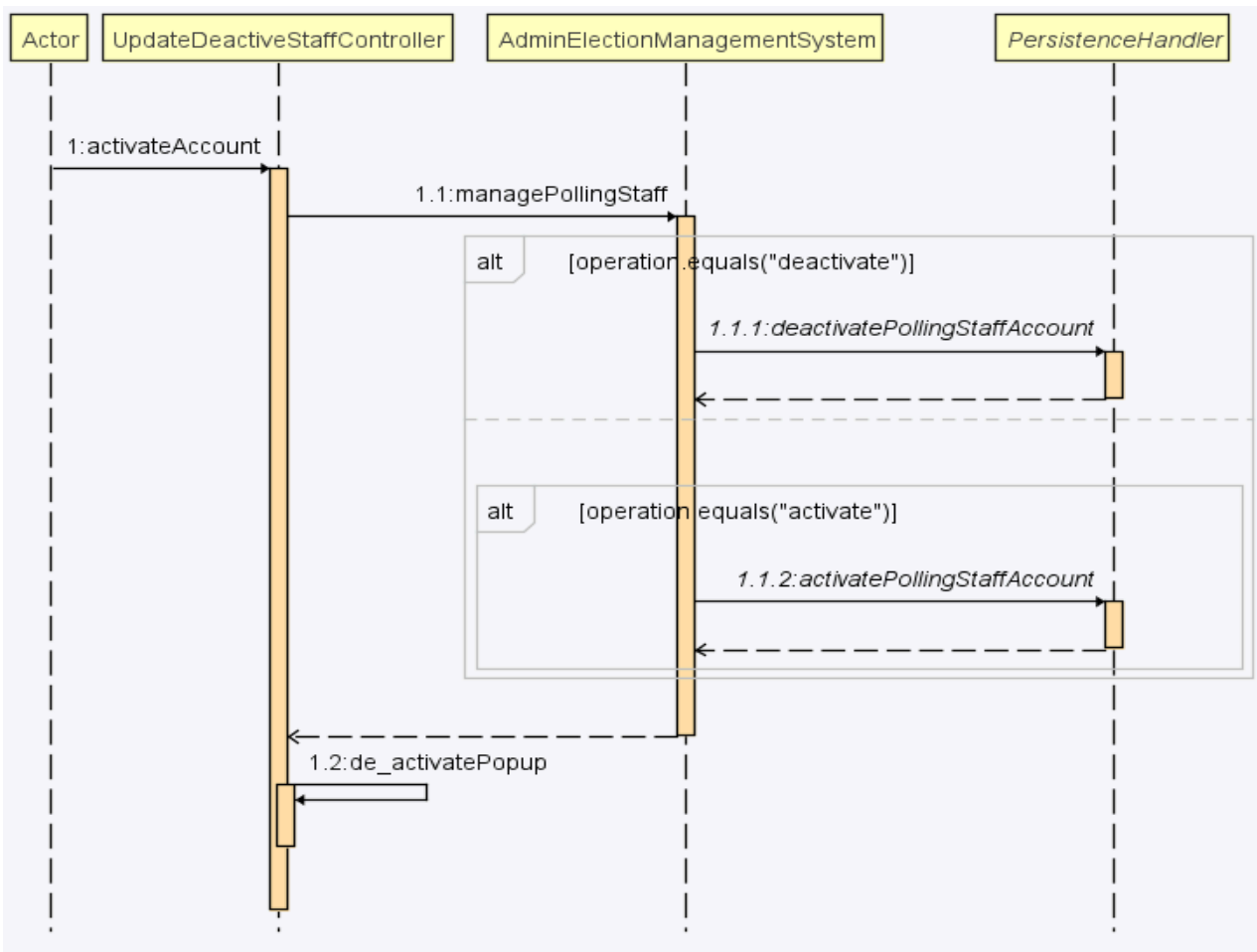
Create Account:



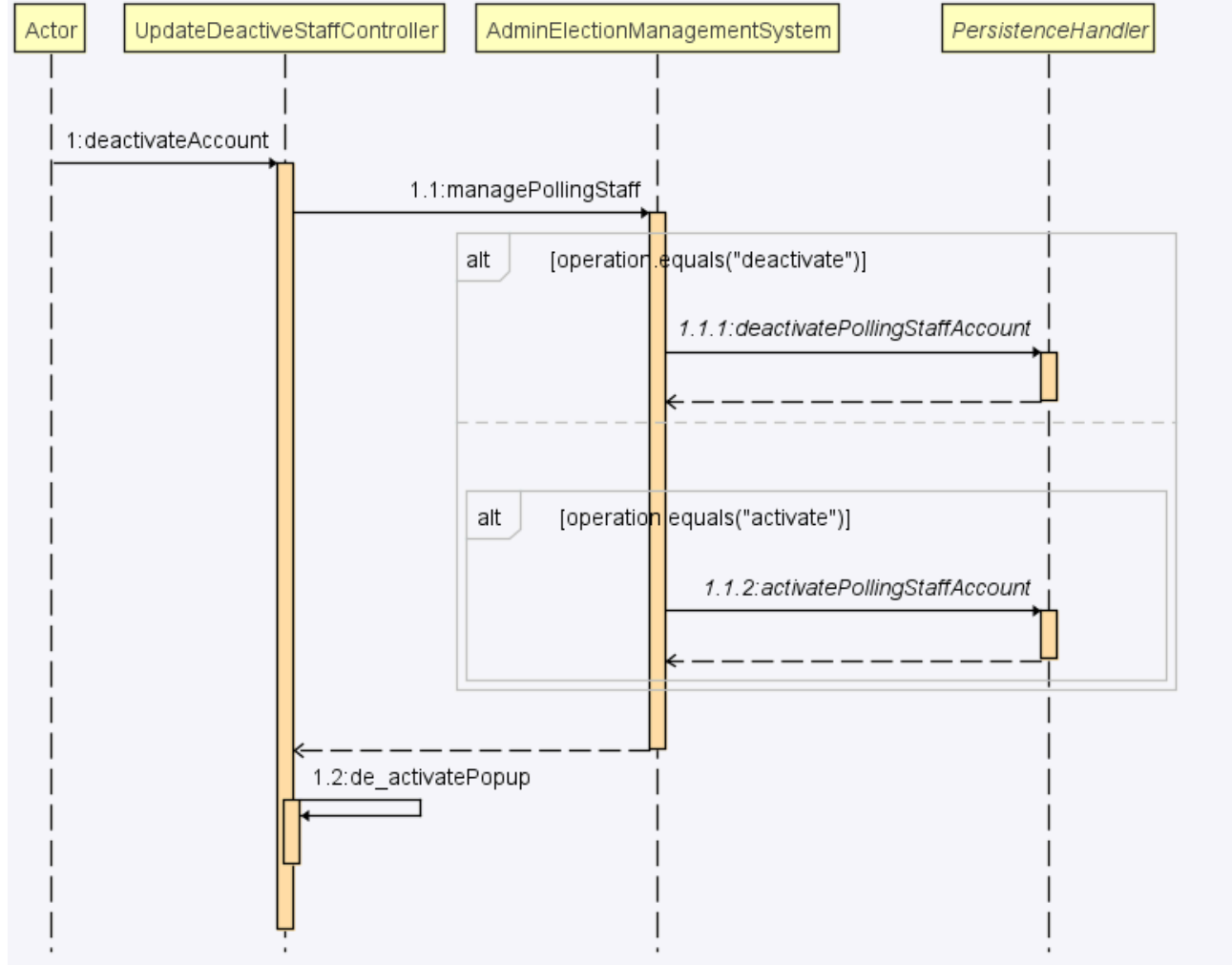
Update Account:



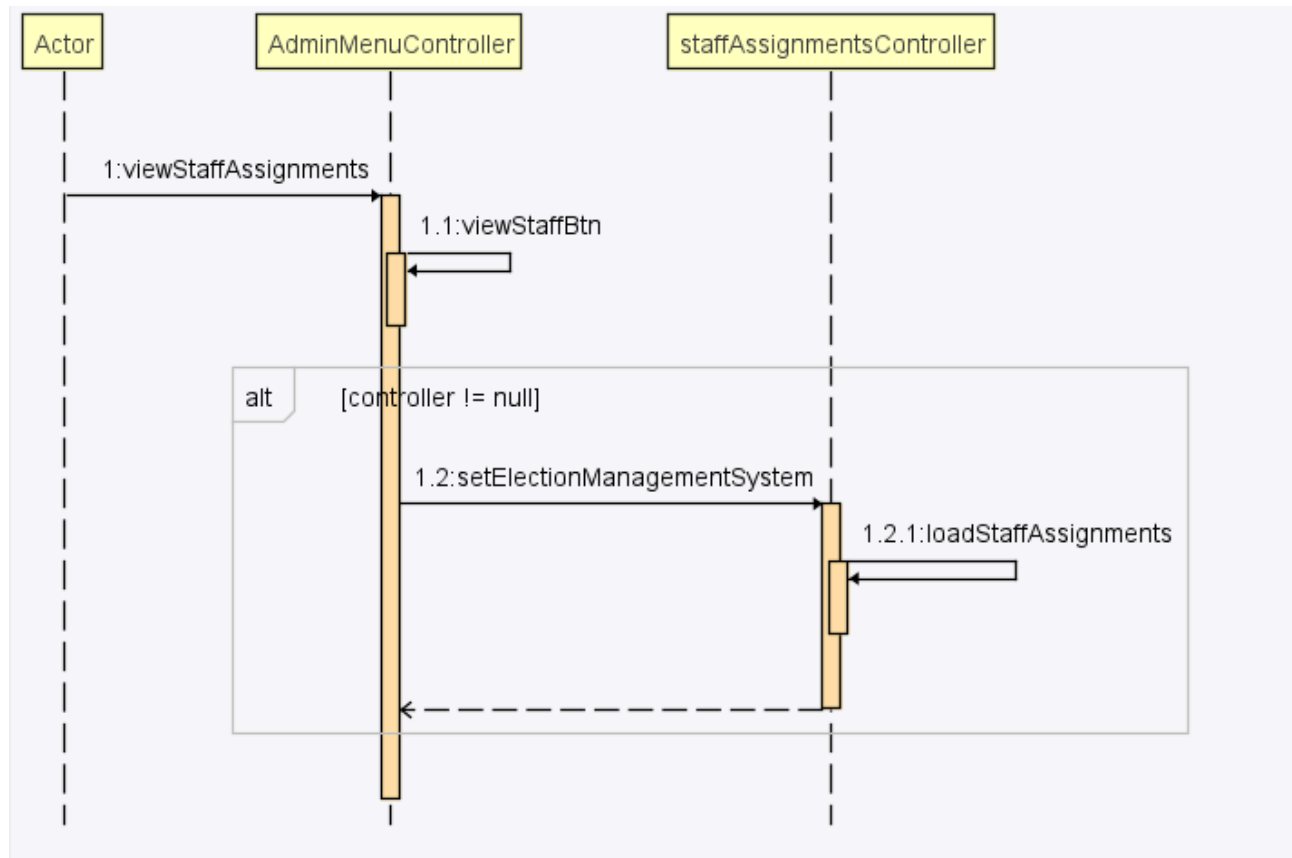
Activate Account:



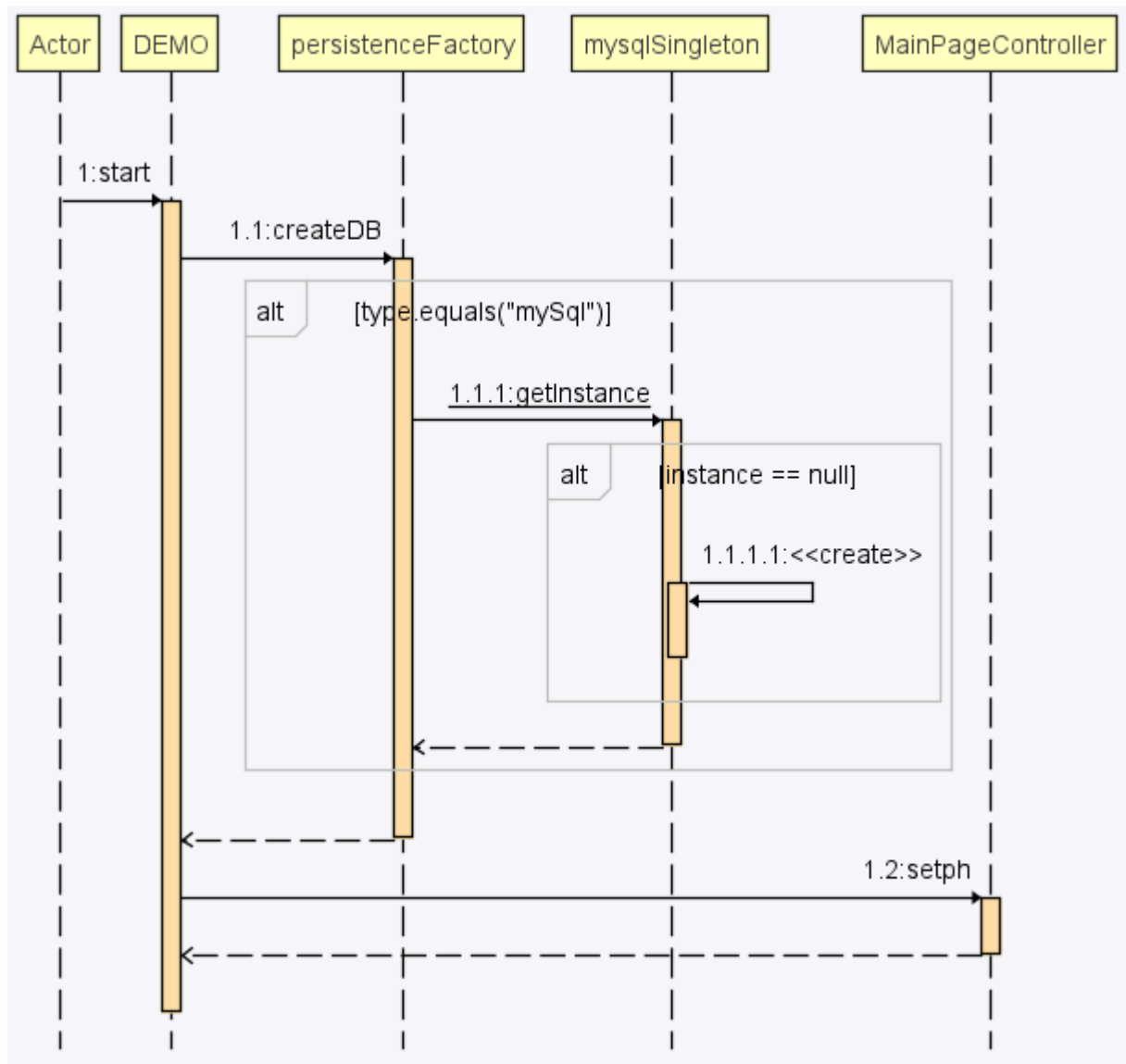
Deactivate Account:



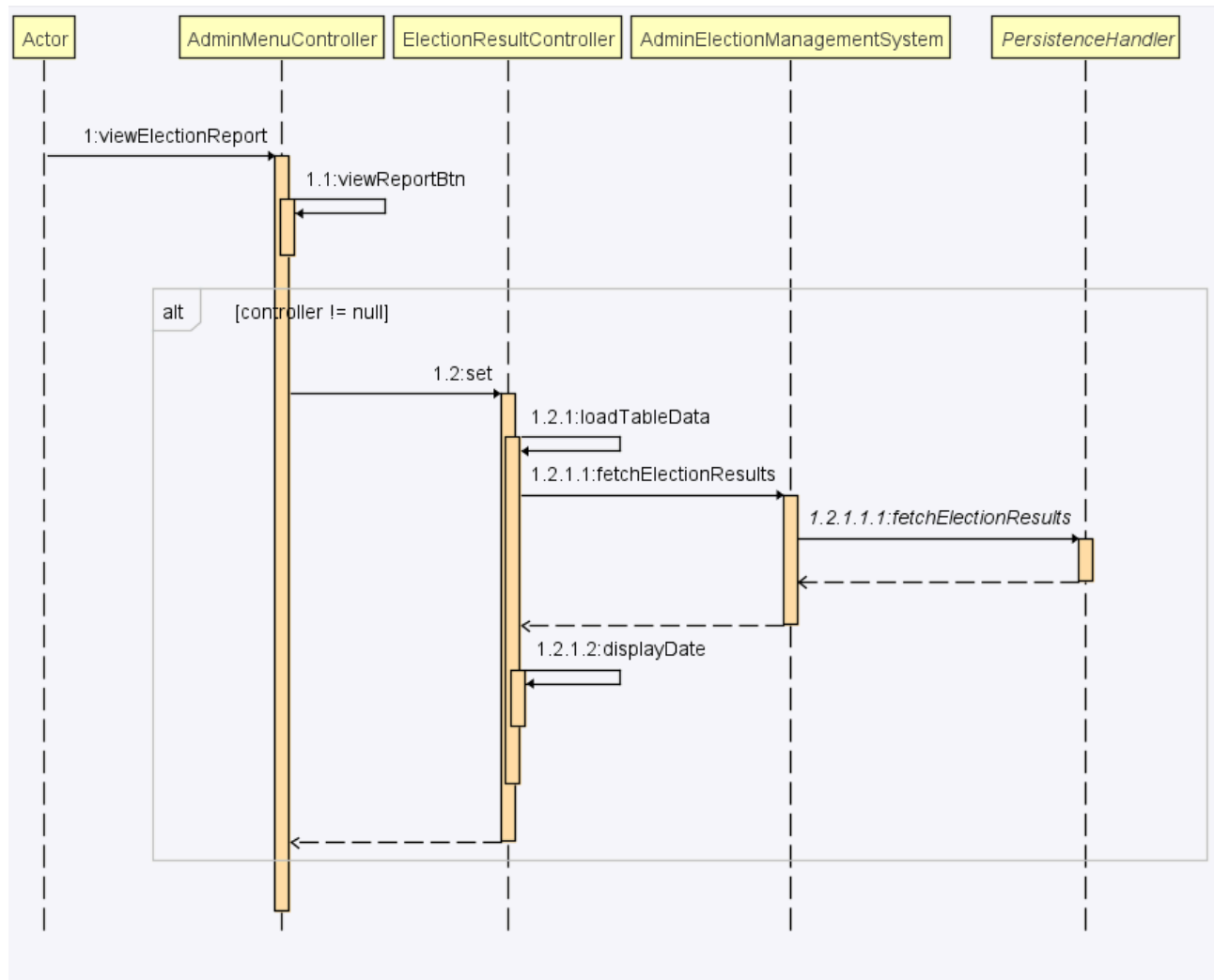
View Polling Staff Assignments



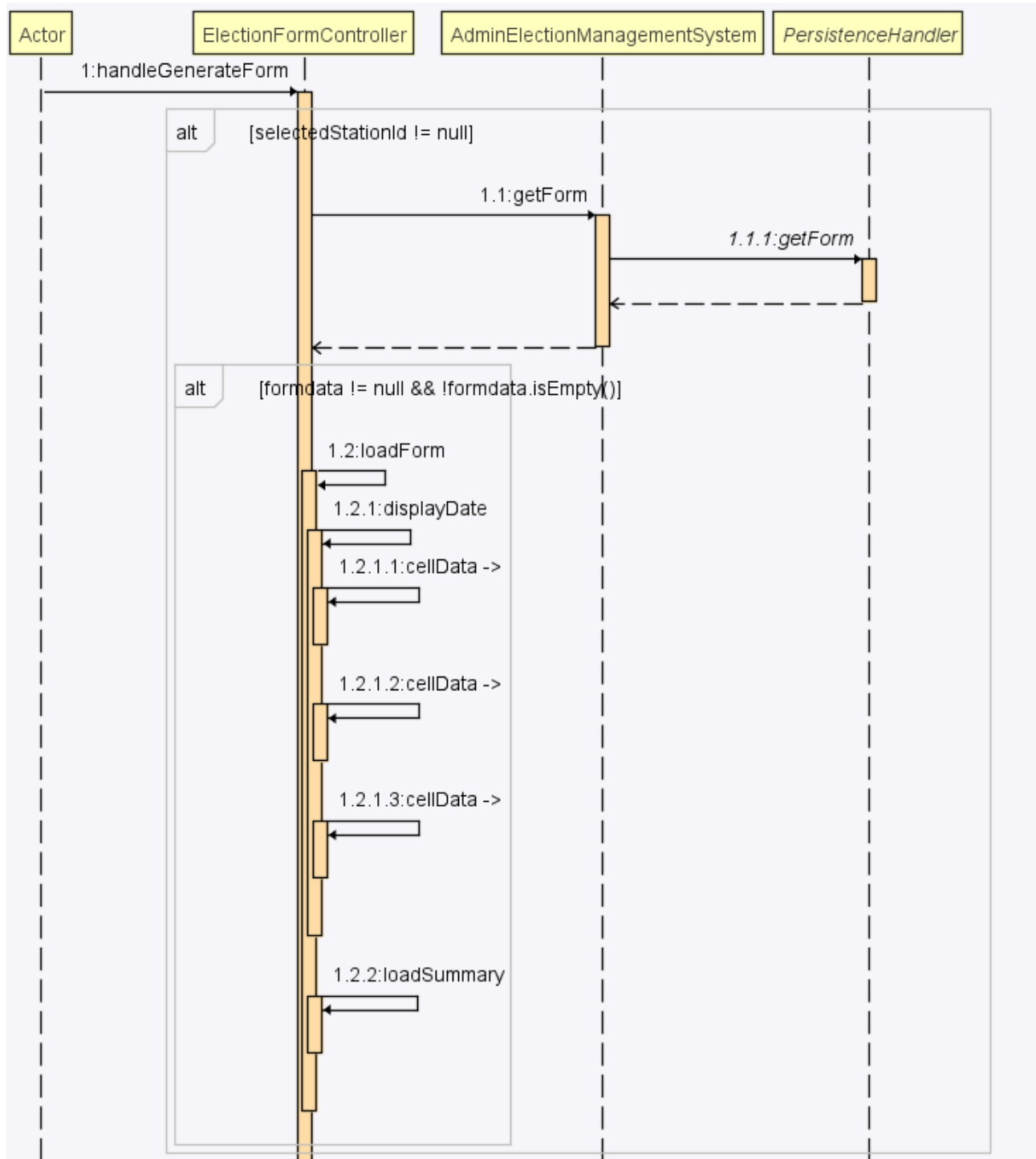
Initiate System



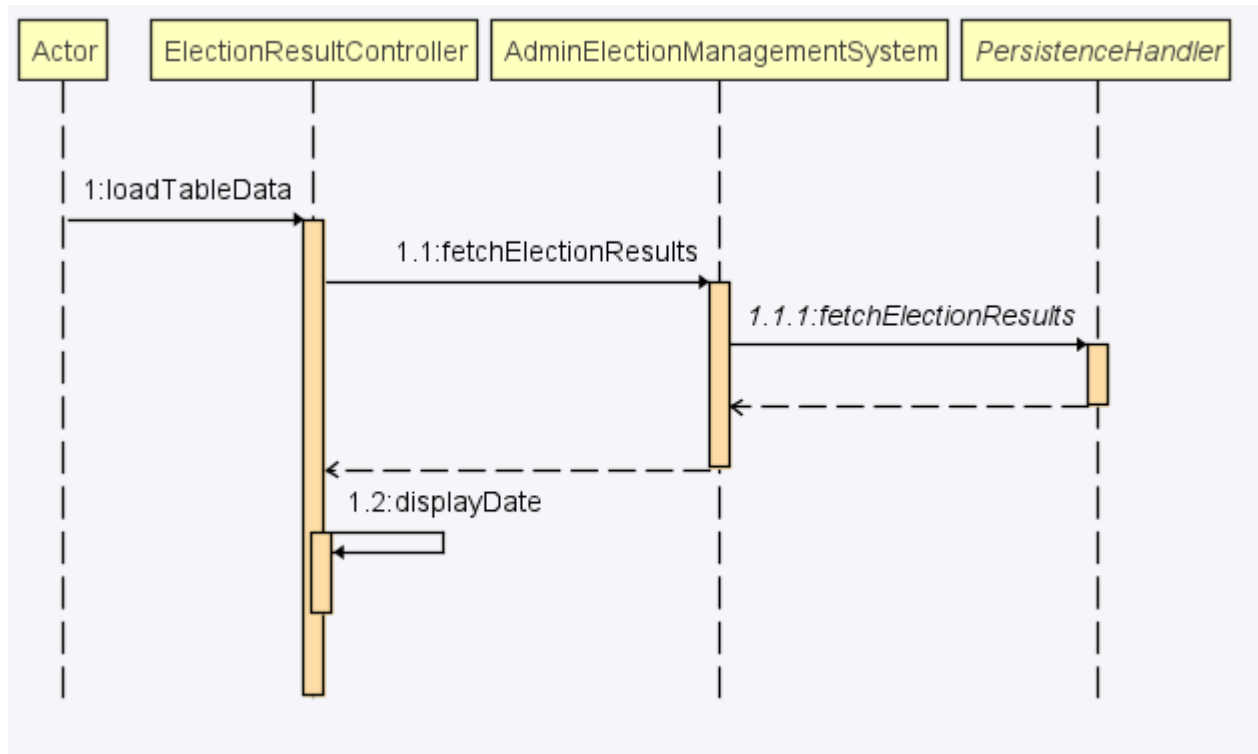
View Election Report



View Election Form



View Election Result

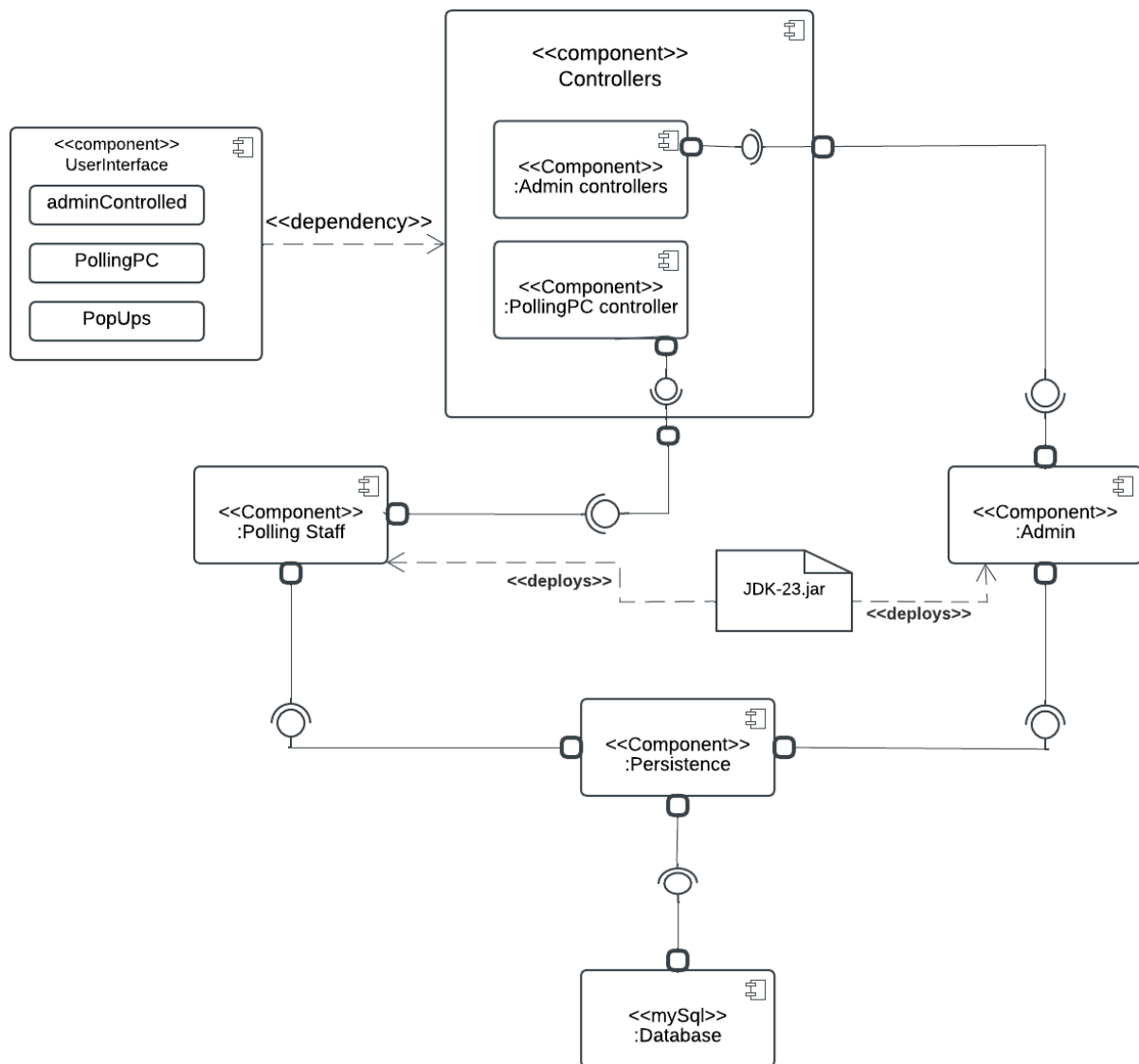


7. Class Diagram

Click to view the diagram

SDA-Project-Votix-ClassDiagram.png

8. Component Diagram



9. Package Diagram



10. Deployment Diagram

