

## Table of Contents

<b>1. Introduction</b>	<b>1</b>
1.1. Introduction	1
1.2. Aims of the Project	2
1.3. Project Scope	2
<b>2. Requirement Specification</b>	<b>3</b>
2.1. Functional Requirements	3
2.1.1. System Components	3
2.2. Non-functional Requirements	5
2.2.1. Performance Requirements	6
2.2.2. Software Quality Attributes	6
2.2.3. Safety Requirements	7
<b>3. Project Design</b>	<b>8</b>
3.1. Methodology	8
3.2. Architecture Overview	8
3.3. Design Description	9
3.4. Behavioral Diagrams	10
3.4.1. Use Case Diagram	11
3.4.2. Activity Diagram	12
3.4.3. Sequence Diagram	13
3.5. Structural Diagrams	14
3.5.1. Class Diagram	15
3.5.2. Component Diagram	16
3.5.3. ERD Diagram	17
<b>4. Implementation and Evaluation</b>	<b>18</b>
4.1. Development Stages	18
4.1.1. Development Phase 1	18
4.1.2. Development Phase 2	18
4.1.3. Development Phase 3	19
4.2. System Integration	19
4.3. User Interface	20
4.3.1. Home Page	21
4.3.2. Other Pages (as relevant)	22
4.4. Evaluation	23
4.5. Unit Testing	24
4.5.1. Registration Use Case Table	25

- 4.5.2. Login Use Case Table ..... 25
- 4.5.3. Other Use Case Tables ..... 26
- 4.6. Functional Testing ..... 27
- 5. **Conclusion & Future Work** ..... 28
  - 5.1. Conclusion ..... 28
  - 5.2. Future Work ..... 29
- 6. **List of Tables** ..... 30
  - Registration Use Case Table ..... 31
  - Login Use Case Table ..... 31
  - (Other tables as needed) ..... 32
- 7. **List of Figures** ..... 33
  - Use Case Diagram ..... 34
  - Activity Diagram ..... 35
  - Sequence Diagram ..... 36
  - Class Diagram ..... 37
  - Component Diagram ..... 38
  - ERD Diagram ..... 39
  - (Additional figures as needed) ..... 40

# Chapter 1

# Introduction

## 1.1 Introduction

The current system that exists now in Pakistan is a paper-based voting system, which requires a lot of manpower and a lot of resources. This voting system also encounters difficulties in the counting process, which is also because it counts manually. As per the recent voting system, symbols of various people parties are used by ballot machines display. People with voting rights may use fake voting cards to vote, which may cause problems. In the current system, participating in elections is a very frustrating task due to the number of the crowd at the election poll, or riots that occur during the election period. One of the main reasons that most people don't want to participate is because there is only one way to vote: by going to the voting booth which takes a huge amount of time and effort to cast a vote. Here EasyVote voting app comes which eliminates manual counting, reduces fake voting risks, and lets people vote from anywhere, avoiding long trips to polling stations.

## 1.2. Aims of the Project

Our project aims to create a secure way for private parties or organizations to conduct their voting process by using a mobile app. To achieve this goal, we have outlined the following aims:

- Develop a modern and secure mobile voting app for private parties and organizations.
- The app is showing all the ongoing elections.
- User can vote to their favorite candidate associated with elections.
- Implement facial recognition technology to enhance the speed and efficiency of the voting process.
- Ensure the integrity of the voting process by preventing the use of fake voting cards.
- Provide convenience by allowing users to vote from any location.
- Create an easy-to-use app that improves the overall voting experience.
- Enable voters to cast their votes easily and quickly.

- Allow voters to view the results in real-time.
- Provide the option for voters to choose and vote for their desired candidates.
- The app will be like a friendly helper for voting, making the whole process smoother and more fun.

### 1.3. Project Scope

The scope of our voting app project outlines what features and functionalities will be included, as well as what aspects will not be covered. Here's a detailed explanation in easy words:

#### Included Features:

- **User Registration and Authentication:** Users can create accounts securely and log in to access voting features.
- **Voting Process:** Voters can cast their votes using the app, which will be verified through facial recognition to prevent fraud.
- **Candidate Information:** Voters can view details about candidates and their profiles before voting.
- **Real-Time Results:** Voters can see live updates of voting results as they are counted.
- **Accessibility:** The app will be designed to be easy to use on mobile devices, ensuring accessibility for all users.
- **Security Measures:** Strong encryption and security protocols will be implemented to protect voter data and ensure the integrity of the voting process.

#### Out of Scope:

- **External Verification:** Verification of voter identities beyond facial recognition (e.g., ID checks by officials).
- **Physical Voting Locations:** The app focuses on remote voting and does not include physical voting stations.
- **Advanced Analytics:** Detailed analytics beyond basic result reporting, such as voter demographics or trends analysis.

## Chapter 2

# Requirement Specification

### 2.1. Functional Requirements

Functional requirements define what the application must do to function correctly. These requirements are the essential features and capabilities that the application must possess for it to operate effectively. Without these key features, the mobile voting app will not function as intended or may fail to work altogether. These requirements are non-negotiable and must be met for the application to be considered successful.

In the context of a development agreement, it is crucial to clearly state these requirements before development begins. The documentation must comprehensively outline each functional requirement, ensuring there is no ambiguity about what is expected from the system. Let's talk about some important practical requirements of our system.

#### 2.1.1 System Components

##### Sign Up:

It is used to register the user and the admin for our system. The system will have a signup feature in order to create the user general account. The signup form will consist of different fields like username, email address, password and role of the user, which will store the information of the user. Once the account has been created, user will be able to access the system.

##### Sign In:

Sign In form will include user email and password of registered user to get login to our system. Type of users who already have an account can directly access their account by providing the valid login details. Otherwise, they will need to perform the Signup process.

**Dashboard:**

Dashboard consist of following items:

- All the ongoing elections will be shown in a slider.
- Below election the associated candidates with elections will be shown.
- User can choose election by sliding the elections
- From dashboard user can go to result page and profile page from bottom navbar.
- After selecting election user will press the “Vote” button displayed below of the screen and will go to next screen to choose candidate.

**Verification:**

After choosing candidate the system will do ID verification of the user by using CNIC of user. If verification become successful it shifts the user towards facial recognition by scanning user face using camera.

**Admin Module**

Admin or owner of the particular organization can manage elections, candidates, organization and voter ID's. Admin can handle that which voter can vote or not. In this way eligible voters can only vote for that particular election. It ensures the integrity of the voting system.

**Result Compilation Module:**

Automatically counts votes and generates real-time results.

**Edit Profile Info**

User can update his/her name, profile image, contact number, CNIC number, and Change Password.

**Sign out:**

Users can sign out from the app.

## 2.2 Non-functional Requirements

Non-functional requirements (NFRs) describe the overall qualities or attributes of the system, focusing on how the system performs rather than specific behaviors or functions. They are contrasted with functional requirements that define specific behavior or functions. The plan for implementing functional requirements is detailed in the system design. The plan for implementing non-functional requirements is detailed in the system architecture, because they are usually architecturally significant requirements.

### 2.2.1 Performance Requirements:

This app requires a good internet connection. If app is connected with good internet then all the real-time elections and candidates will be shown correctly. The result is loaded fast and vice versa because it connects with the firebase cloud server so internet is required to connect the server better connections for better results. The app load and operate quickly, providing responses to user inputs within a specific time frame. The app is able to handle a large number of users simultaneously without performance degradation.

### 2.2.2 Software Quality Attributes:

#### Reliability:

- App is reliable to use. In the event of an error, the administrators are there to maintain and update. This also ensures that all user data is private and secure.
- The app is able to recover gracefully from crashes or errors without losing user data.

**Maintainability:**

- App must be maintained on daily basis or weekly basis. It must be updated as soon as the error be found. Necessary measurements must be taken in this case.
- The app is designed in a modular fashion, making it easier to update or replace components without affecting the whole system.

**Usability:**

- It will provide an easy-to-use interface for users. This feature/requirement provides users with such a user-friendly interface that allows them to easily and efficiently navigate through the application.
- The app have an intuitive and making it easy for users of all skill levels to navigate and vote.

**Availability:**

It will be there whenever users around the world can access it. This feature helps users to access the app from anywhere in the world. Users can vote according to election time at any place while connecting to the internet.

**2.2.3 Safety Requirements:****Data Protection:**

- User information is kept confidential because this information is private. Ensuring the security of user data is also the main objective. No unauthorized user can access user information. It ensures that all user data is confidential and secure.
- The app should ensure that all user data, especially votes, are securely stored and transmitted.

**Authentication and Authorization:**

- App have strong mechanisms to verify the identity of users and ensure that only authorized users can vote.



**Privacy:**

- The app should comply with privacy regulations, ensuring that user data is not misused or shared without consent.

# Chapter 3

## Project Design

### 3.1 Methodology

When reporting a research, writing a methodology is an essential part of presenting your findings. The methodology, the detailed description of reporting process that forms a separate section of report, supports the findings by explaining the research techniques and providing a roadmap of how it arrived at the conclusions. That's why we are going to explain every aspect of our project research in this section. We are developing our application as simple as a novice user can use easily. At first we show all the elections and associated candidates so that user can cast vote in its organization elections. First user have to login or sign up to vote in any election, if the user is unauthentic then application leads to the authentication module like signup or sign in.

### 3.2 Architecture Overview

The architecture of our voting app, utilizing Firebase as the database server, integrates Firebase's comprehensive suite of tools to enhance scalability, real-time capabilities, and ease of development. This architecture follows a client-server model approach.

### 3.2.1 Client-Server Architecture

In this model, the client (mobile app) communicates with Firebase services to perform operations like user authentication, voting, and result retrieval. Firebase acts as the backend server, handling data storage, real-time updates, and user management.

### 3.2.2 Three-Tier Architecture

The voting app is structured into three primary layers: Presentation Layer, Application Layer, and Data Layer, with Firebase services integrated into these layers.

#### 1) Presentation Layer

The Presentation Layer is the front-end of the app where users interact with the system. This includes all the screens and UI components, such as:

- Login and registration screens for user authentication and new user sign-ups.
- Voting screen where users view candidates and cast their votes.
- Results screen to display real time voting.

This layer is implemented using mobile development framework with Flutter development for iOS and Android. Firebase Authentication is used for managing user sign-in and registration.

#### 2) Application Layer

The Application Layer contains the business logic of the voting app, handling user requests and data validation. Key components in this layer include:

- **Authentication and Authorization:**

Firebase manage Authentication, providing secure user sign-in methods, including email/password, phone authentication, and social logins.

- **Voting Management:**

Firebase manage voting operations using Firebase Cloud Firestore for storing votes, Firebase Functions for server-side logic, and Firestore Security Rules for data validation and authorization.

### 3) Data Layer

The Data Layer manages data storage, retrieval, and security. Firebase services involved in this layer include:

- **Firebase Cloud Firestore:**

A NoSQL document database to store user information, votes, candidate details, and election results. Firestore provides real-time data synchronization.

- **Firebase Real-time Database:**

Optionally used for real-time data requirements, offering instant data synchronization.

- **Firebase Storage:**

It stores user and candidate's images.

- **Firestore Security Rules:**

To enforce data validation, access control, and security policies.

### 3.2.3 Security Architecture

Security is a key consideration, with Firebase offering robust security features:

- **Firebase Authentication**

It securely manages user authentication with various sign-in methods.

- **Firestore Security Rules**

It define who can access and modify data in firestore, ensuring data privacy and security.

- **Data Encryption**

Firebase automatically encrypts data in transit using HTTPS and at rest.

### 3.3 Design Description

Project design is a first phase of the project, in which the main features, structure, success criteria and key results of the project are planned. The aim is to develop one or more designs with which the desired project goals can be achieved. The design phase of the project can generate a variety of different outputs, including sketches, flowcharts, floor plans, Code screen layouts, prototypes, photo prints, and more. So In our case we have too many screens performing different tasks. Like we have Login Signup Screen. Forms and Fields will be available on these screens so a user can easily get login and signup on our app. After the authentication module app will lead you to the home screen where you can select election from list of elections to cast a vote and go for further verification. A user will simply come and select election and in the next select favorite candidate to vote. After it the verification process will begin and after successful verification user will elect a vote to its favorite candidate successfully.

#### UML DIAGRAMS:

UML is the abbreviation for Unified Modeling Language. It is a schematic representation of software components. A schematic representation makes the system easier to understand. It is also easy to detect system errors and failures. This modern UML approach is now widely used in documenting various software or business processes that need to be developed. A UML diagram is a diagram that visually represents the main actors, roles, classes or objects for better understanding and documentation.

#### Types of UML diagrams

There are several types of UML diagrams, but the broader categories are Behavioral and Structural Diagrams. Behavioral diagrams show the behavior of the system, its actors and roles. While Structural diagrams are used to analyze and depict the structure of the system. These two broad categories further encompass all other UML Diagrams.

### 3.4 Behavioral Diagrams:

Behavior diagrams describe the behavior of the system, its actors, roles and actions. Shows the system flow. It consists of activity, use case and interaction diagrams, including sequence diagrams.

#### 3.4.1 Use Case Diagram:

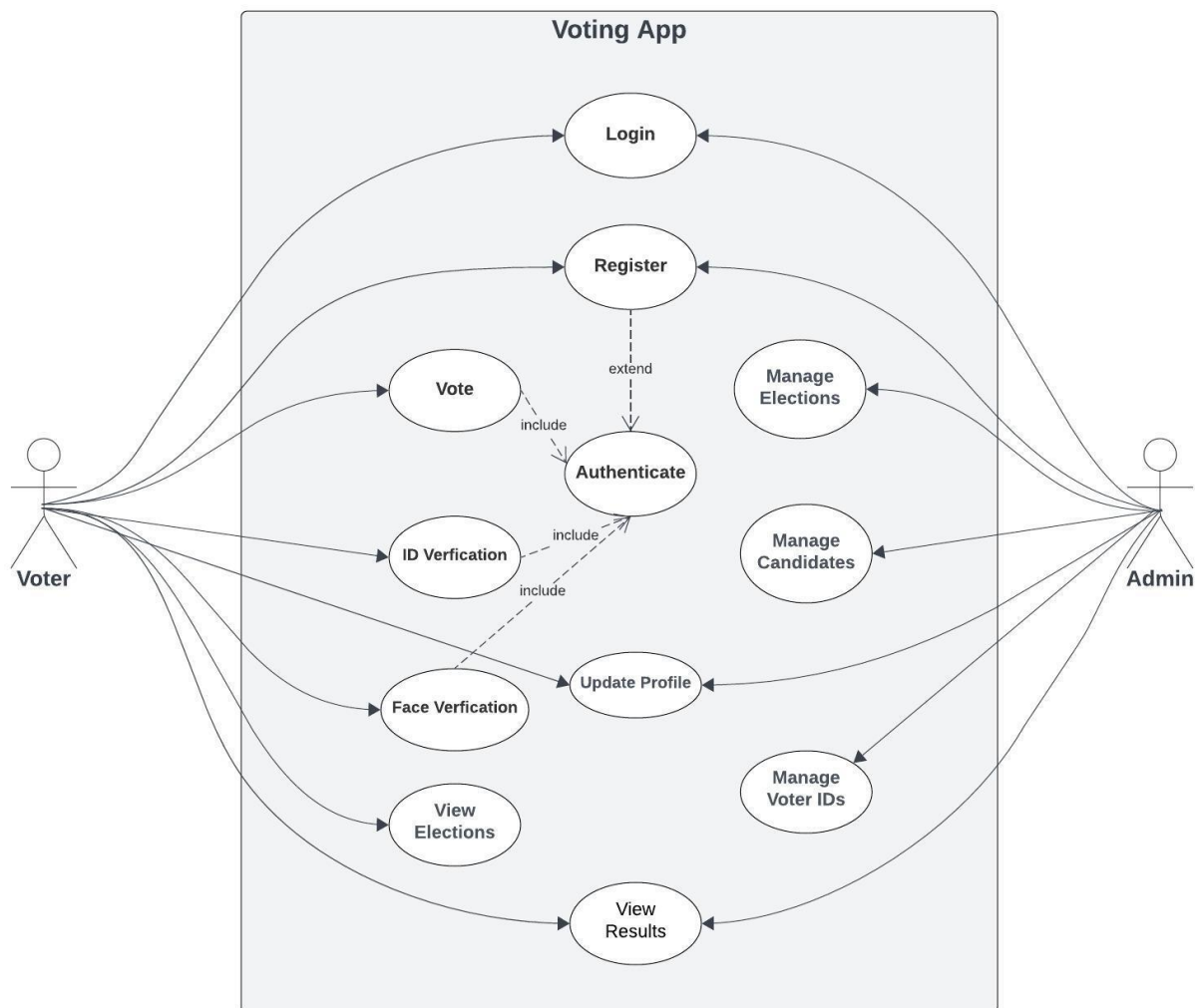


Figure 3.1: Use Case Diagram

### 3.4.2 Activity Diagram:

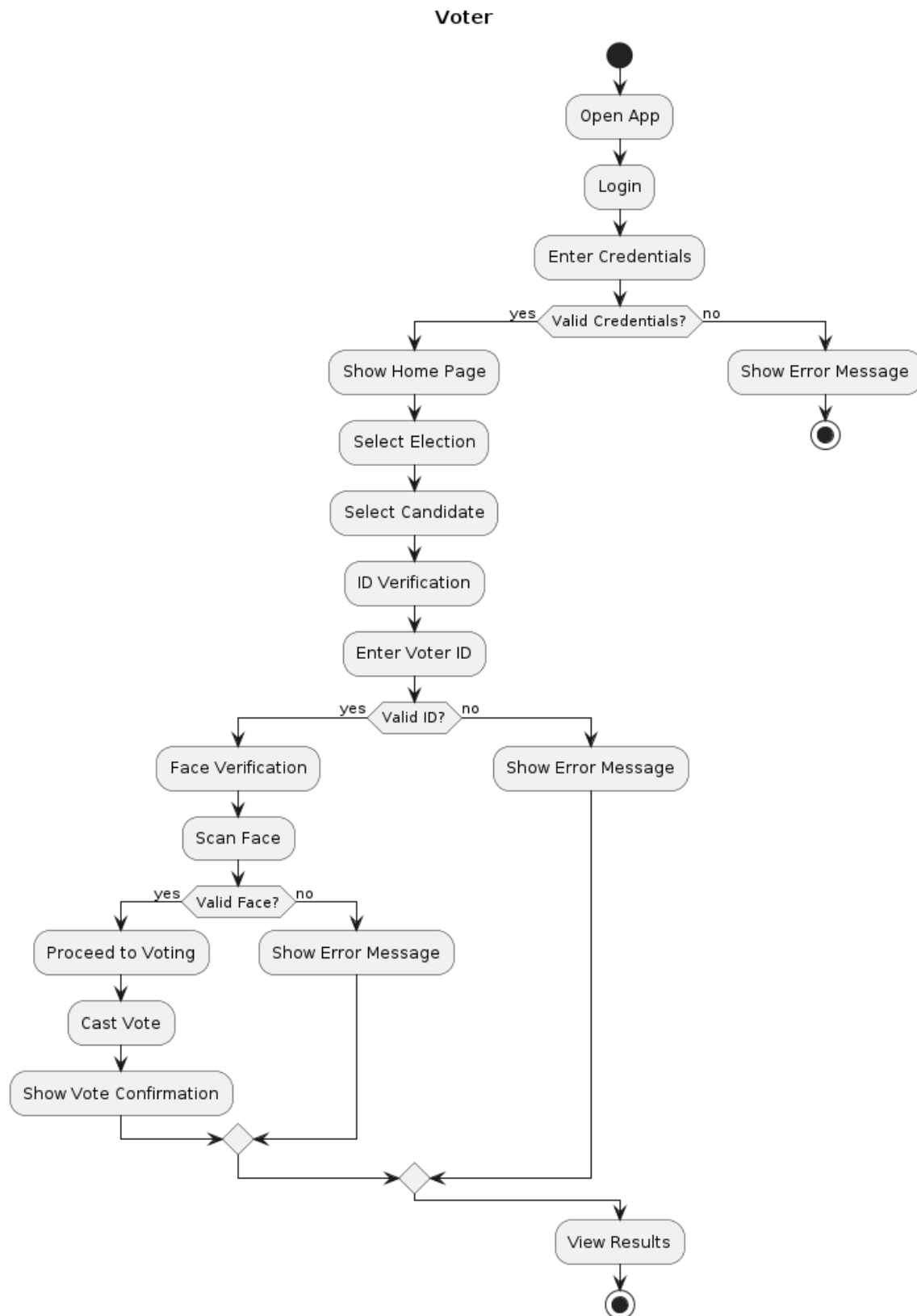


Figure 3.2 (a): Activity Diagram

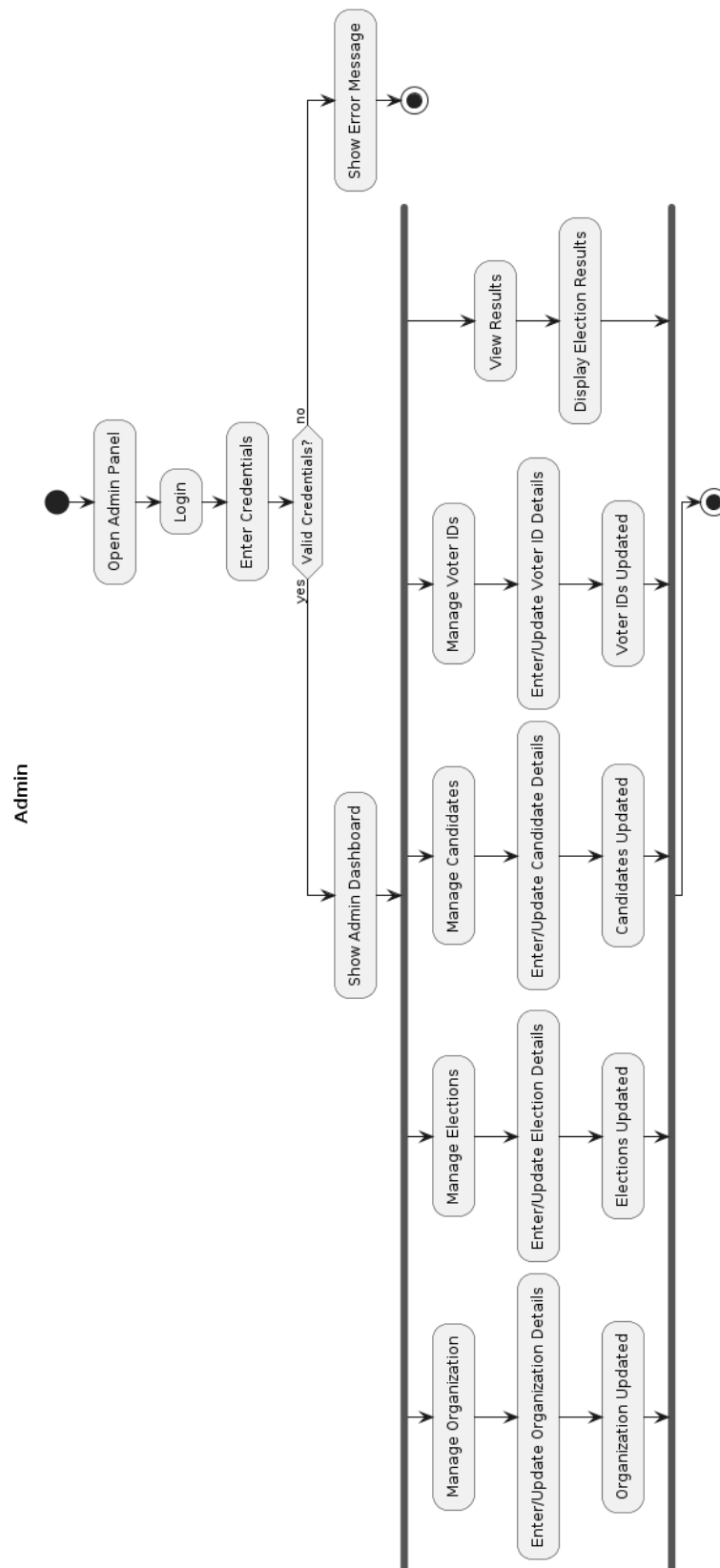


Figure 3.2 (b): Activity Diagram



### 3.4.3 Sequence Diagram:

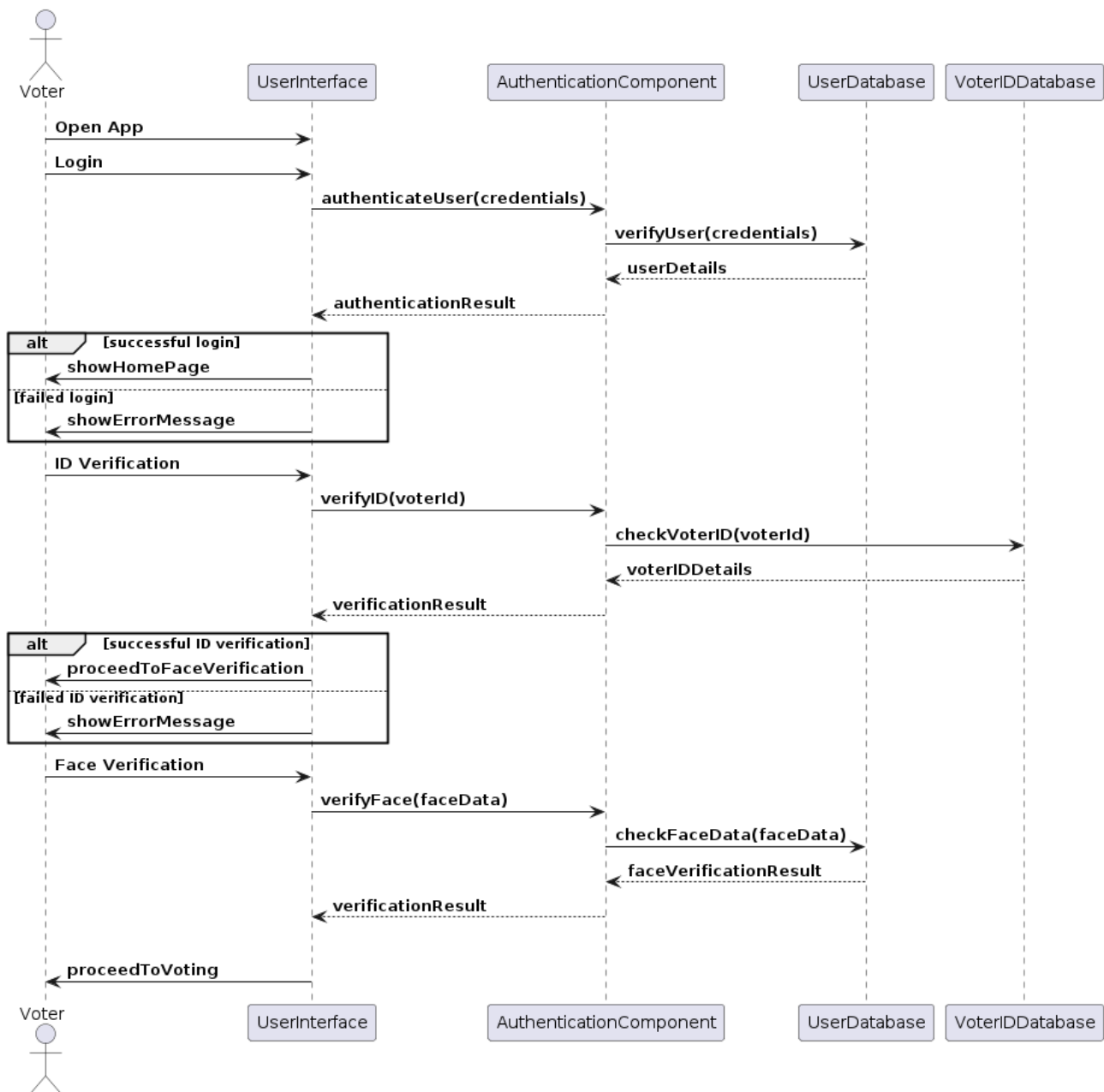


Figure 3.3 (a): Sequence Diagram

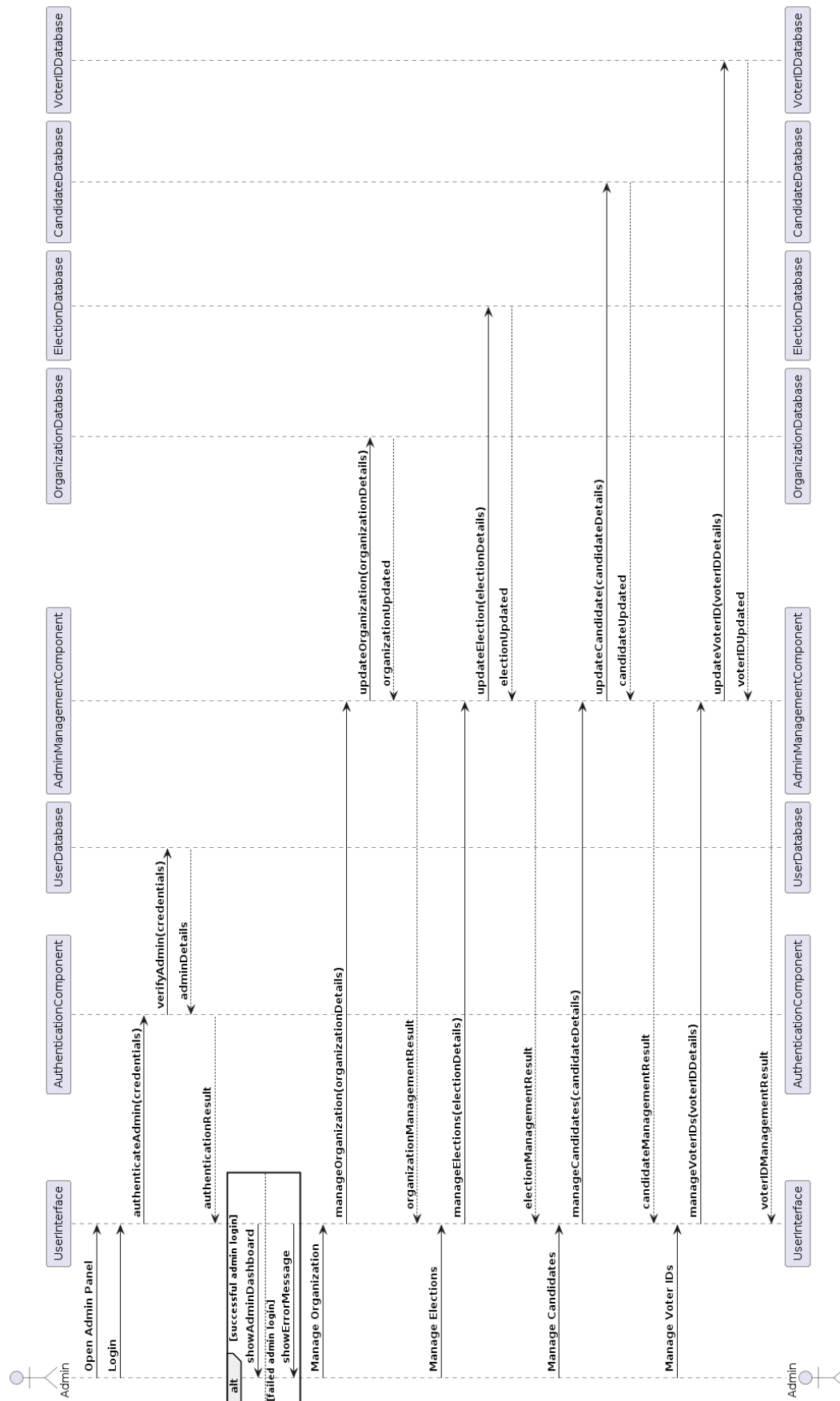


Figure 3.3 (b): Sequence Diagram

## 3.5 Structural Diagrams:

Structural diagrams try to analyze and depict the structure of the system. It gives the clear view of structure of the system. It helps to understand the main aspects of a project. Structural diagrams include Class Diagram, Object Diagram, Component Diagram and some other diagrams

### 3.5.1 Class Diagram

The class diagram shows the static structure of the system. Classes can also be inherited. The inherited ones are called the child class and the inherited one is called the parent class. Classes consist of attributes and methods.

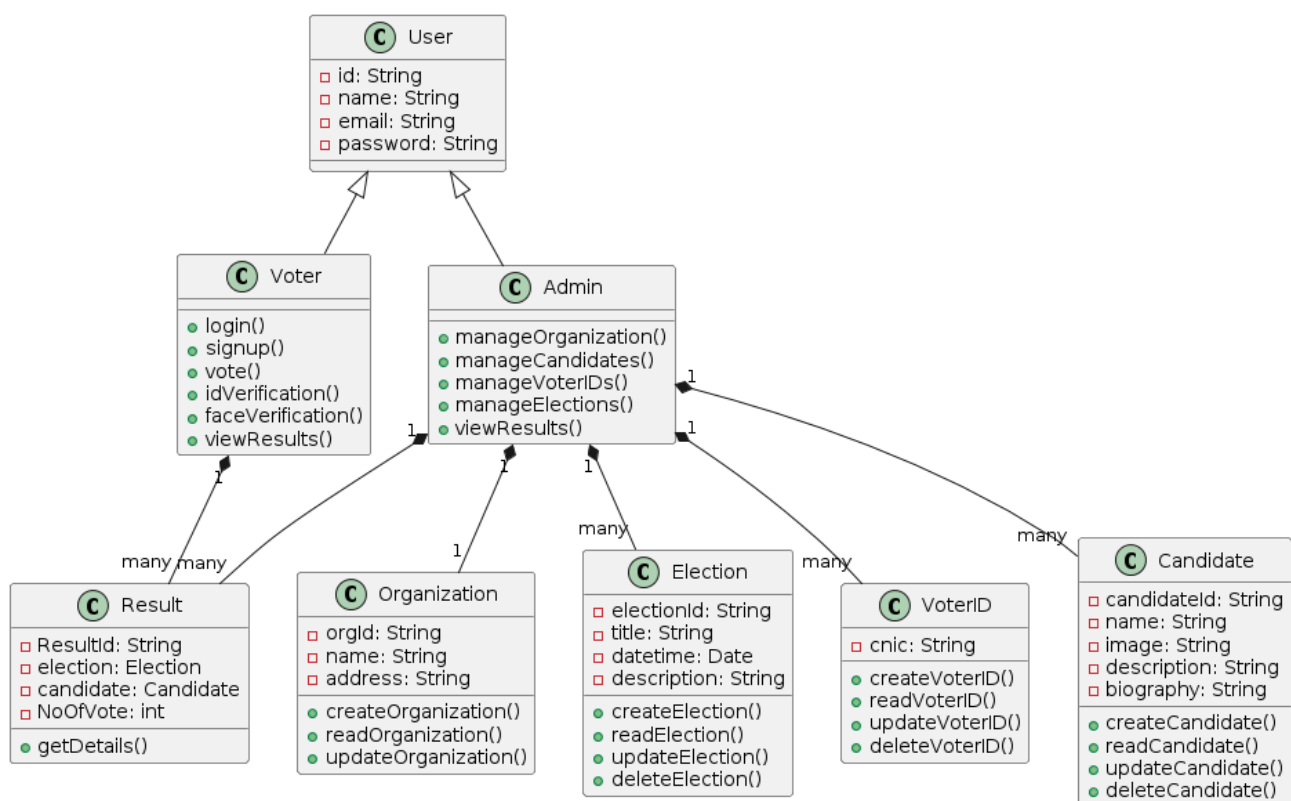


Figure 3.4: Class Diagram

### 3.5.2 Component Diagram

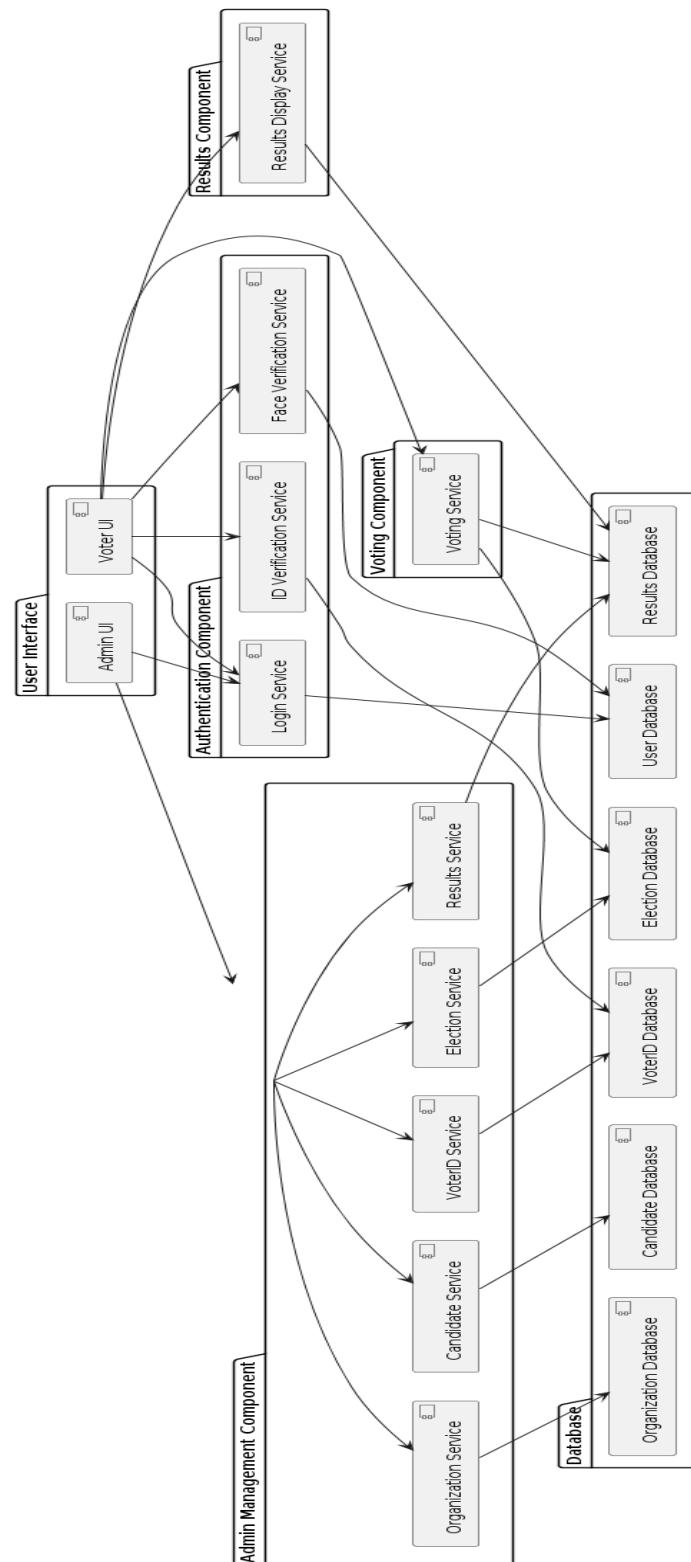


Figure 3.5: Component Diagram

### 3.5.2 ERD Diagram

ERD model, is a type of structural diagram which is mostly use in database design. The diagram is given below:

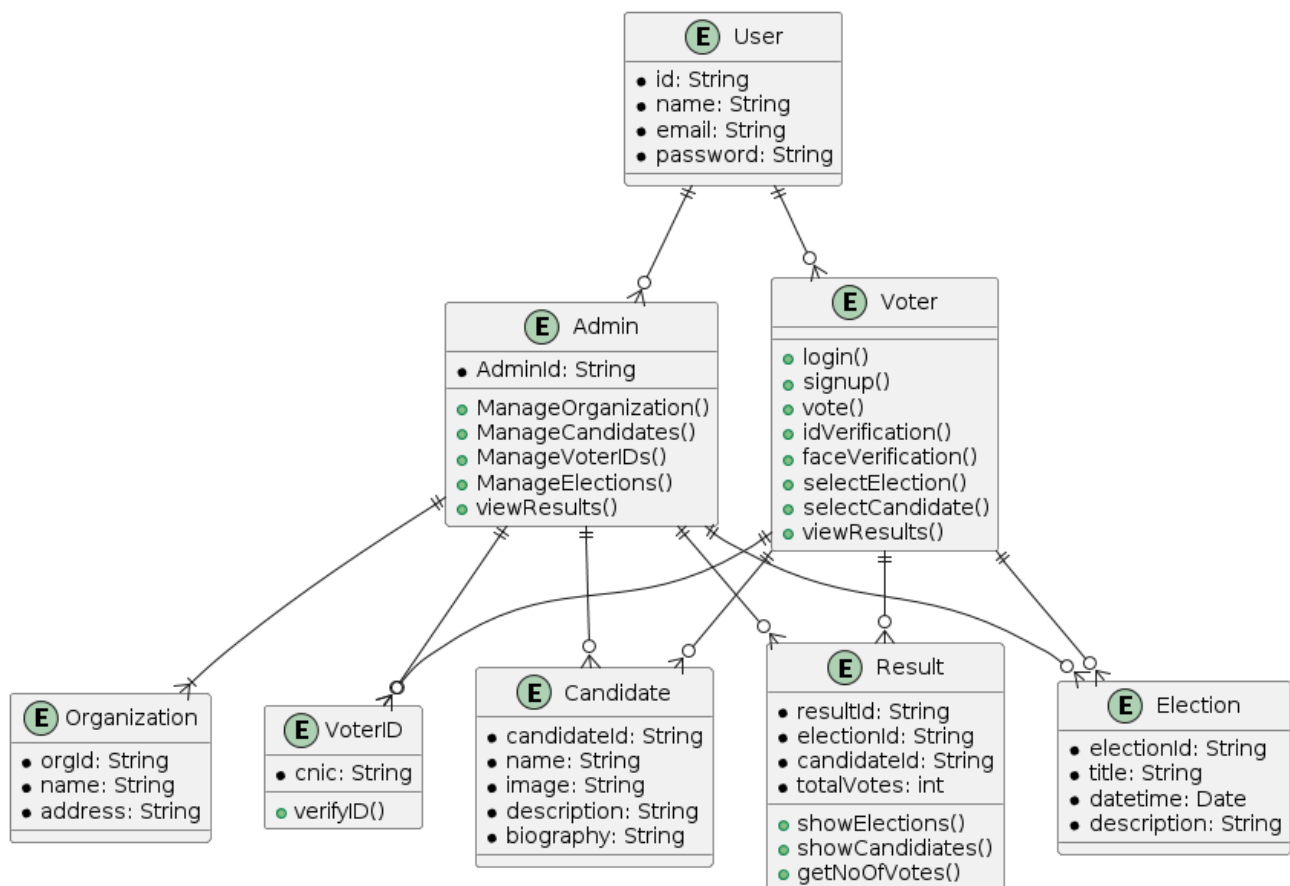


Figure 3.6: ERD Diagram

## Chapter 4

# Implementation and Evaluation

Before starting the development of this project we did analysis all of the requirement to make this app work smoothly. After all this analysis we made a decision to divide this development into phases so we can achieve this in a good manner.

### **4.1 Development Stages**

The development of our voting app was carried out in multiple phases, each focusing on different aspects of the project. The stages are detailed below:

#### **4.1.1 Development Phase 1**

In the first phase of development, our primary focus was on the UI design. We began by creating the UI design using Figma, which allowed us to visualize the layout and user experience. Once the design was finalized, we started converting this design into a mobile app using Flutter. We set up our development environment by installing VS Code, Flutter, and Firebase. We then started building the front end of the app, ensuring that each element matched the Figma design and provided an intuitive user experience.

## 4.1.2 Development Phase 2

After completing the initial UI design and frontend development, we moved on to the second phase, which involved backend development. In this phase, we focused on setting up the server-side components and integrating Firebase. We used Firebase Firestore as our database, which offers real-time synchronization and offline support. We mapped out our database structure using an Entity-Relationship Diagram (ERD) to understand how different pieces of information would relate to each other. We then created collections and documents in Firebase Firestore to store our data and set up Firestore Security Rules to ensure data integrity and security.

## 4.1.3 Development Phase 3

In the third phase, we integrated the frontend and backend components. This involved connecting the Flutter frontend with Firebase to handle data operations such as user authentication, voting, and retrieving results. We ensured that all buttons, forms, and features worked as expected and that data was correctly stored and retrieved from Firestore. We also implemented Firebase Authentication to manage user sign-ins.

## 4.2 System Integration

For our project development, we utilized a combination of Flutter and Firebase to build a robust mobile application. Flutter, with its powerful UI toolkit, allowed us to create a visually appealing and responsive user interface. Firebase served as our backend, providing services such as real-time database (Firestore), authentication, cloud messaging, and storage. VS Code was used as our primary code editor, equipped with various plugins to enhance our development workflow. This integration enabled us to create a dynamic and responsive application that effectively handled both client-side and server-side functionalities.

## 4.3 User Interface

The user interface (UI) is a critical aspect of our voting app, as it directly impacts user experience and satisfaction. We invested significant effort into designing a UI that is intuitive and easy to use, even for novice users. Using Figma, we crafted a design that follows best practices in UI/UX design, ensuring that buttons are placed where users expect them, text is easy to read, and the overall navigation is smooth and logical. Our goal was to create a user-friendly interface that encourages user engagement and makes the voting process straightforward and enjoyable. We are proud of how the UI has turned out, and believe it will provide a positive experience for all users.



### 4.3.1 Initial Screens

#### Splash Screen

This is the splash screen that appears when the app is first launched. It features the app's logo and creates an initial impression of the app

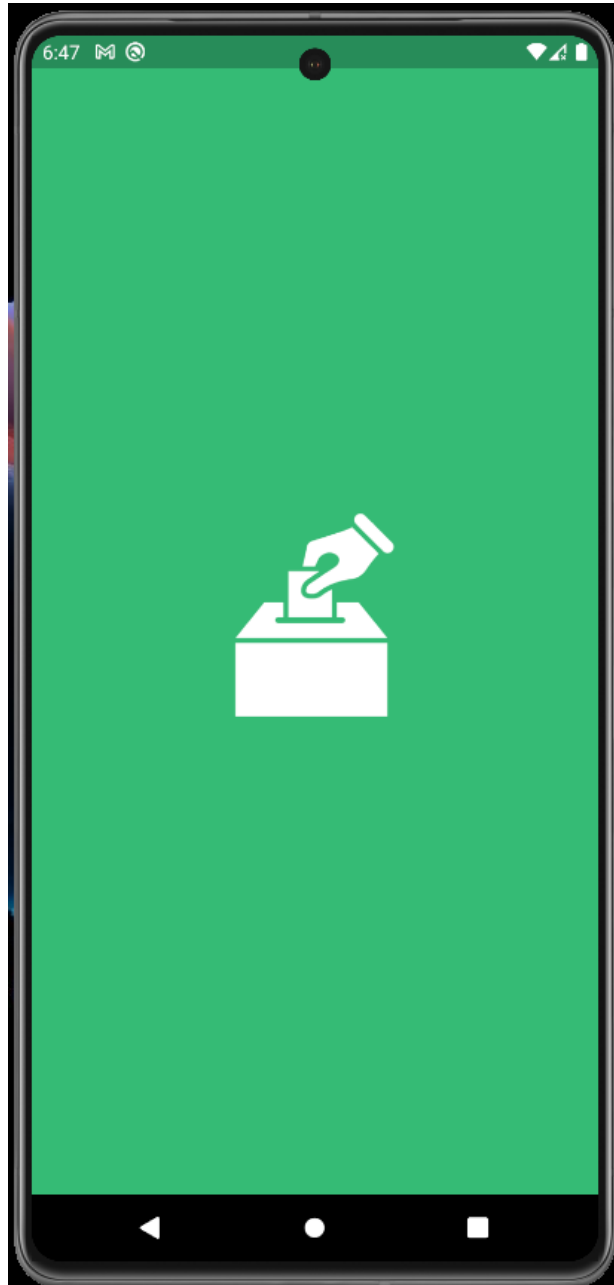


Figure 4.1: Splash Screen

## Auth Screen

This screen provides users with the option to either log in or register. It serves as the gateway to access the app's functionalities.

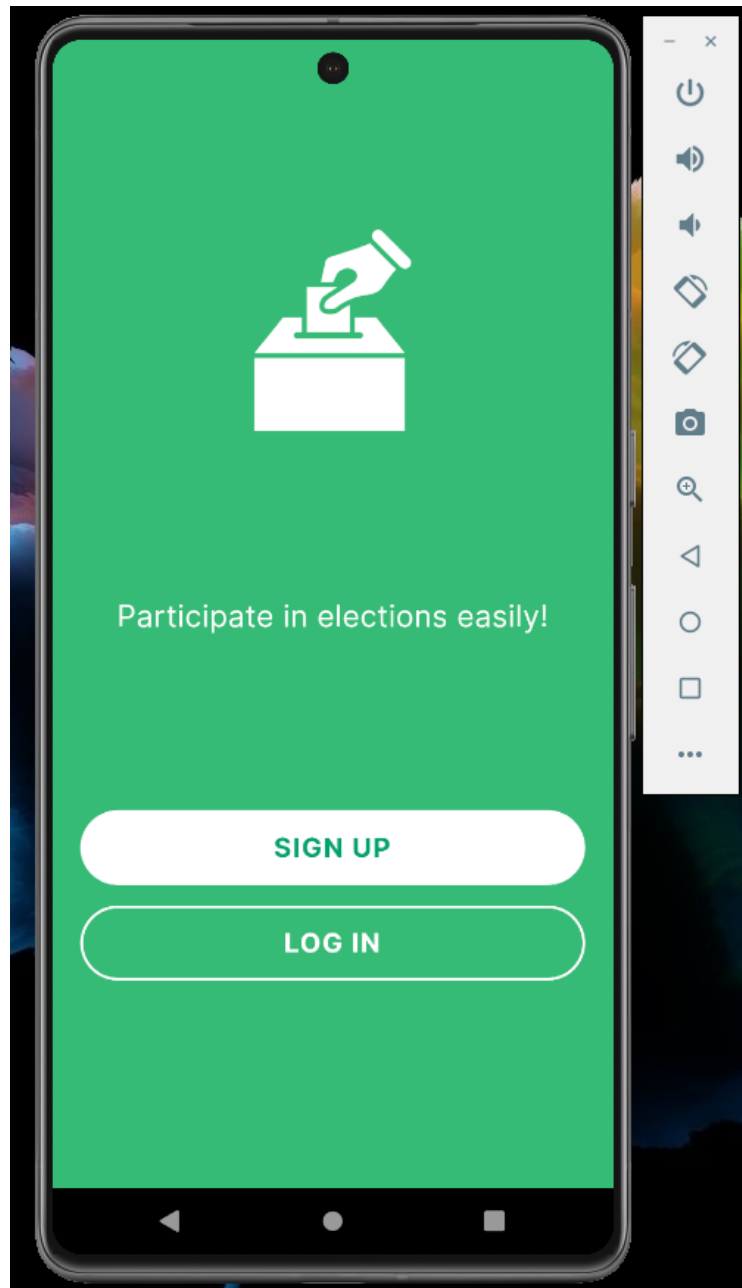


Figure 4.2: Auth Screen

## 4.3.2 Login and Registration Screens

### Login Screen

This is the login screen of the voting app, where users can enter their credentials to access their accounts.

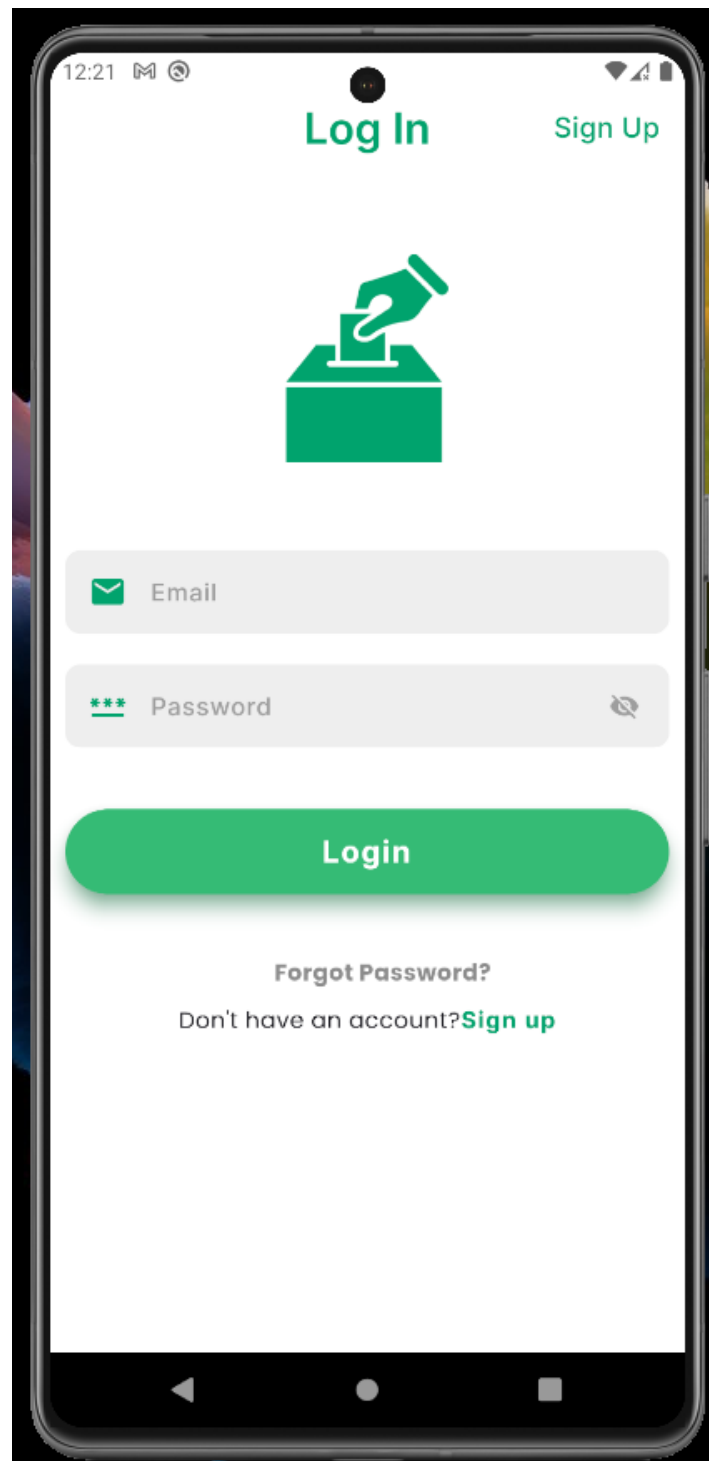


Figure 4.3: Login Screen

## Registration Screen

This is the registration screen of the voting app, where new users can sign up by providing their personal information.

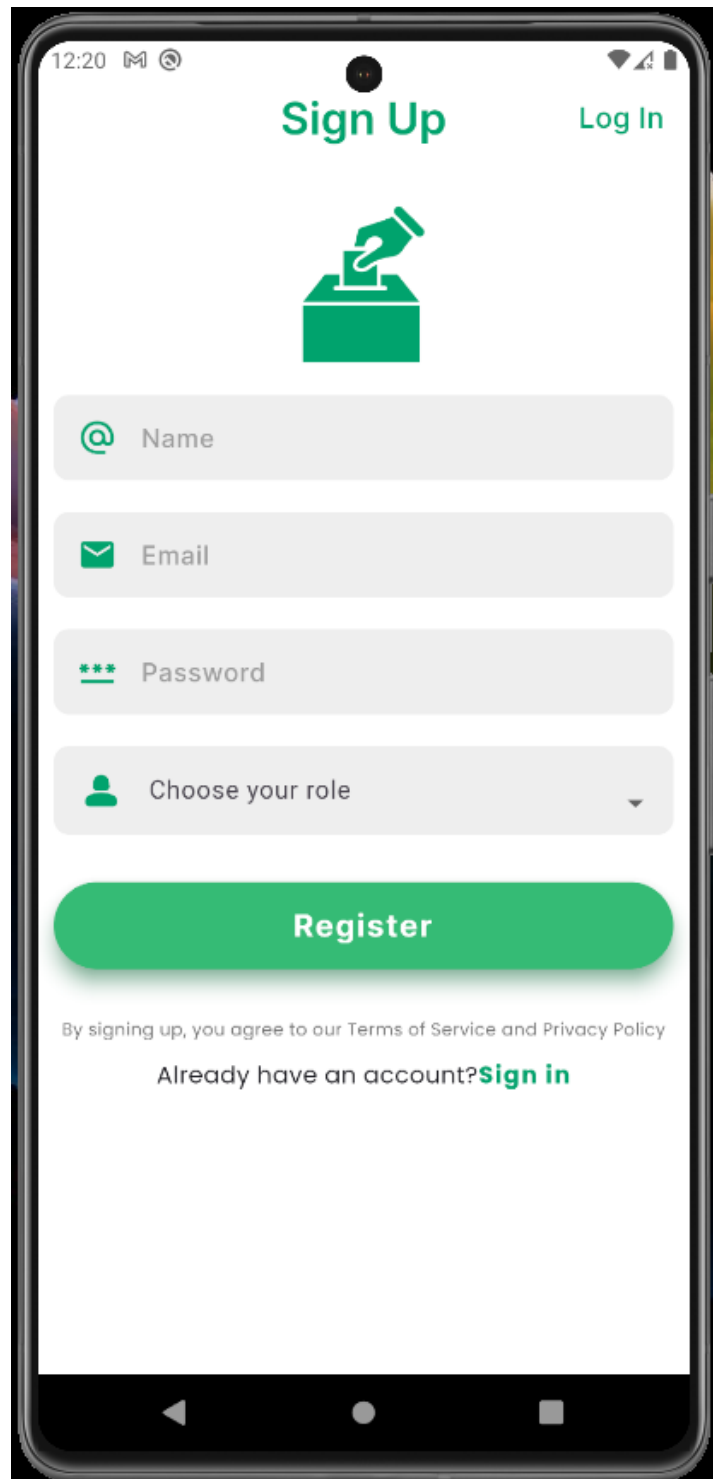


Figure 4.4: Registration Screen

## Forgot Password Screen

The "Forgot Password" screen, where users can enter their email address to receive password reset link.

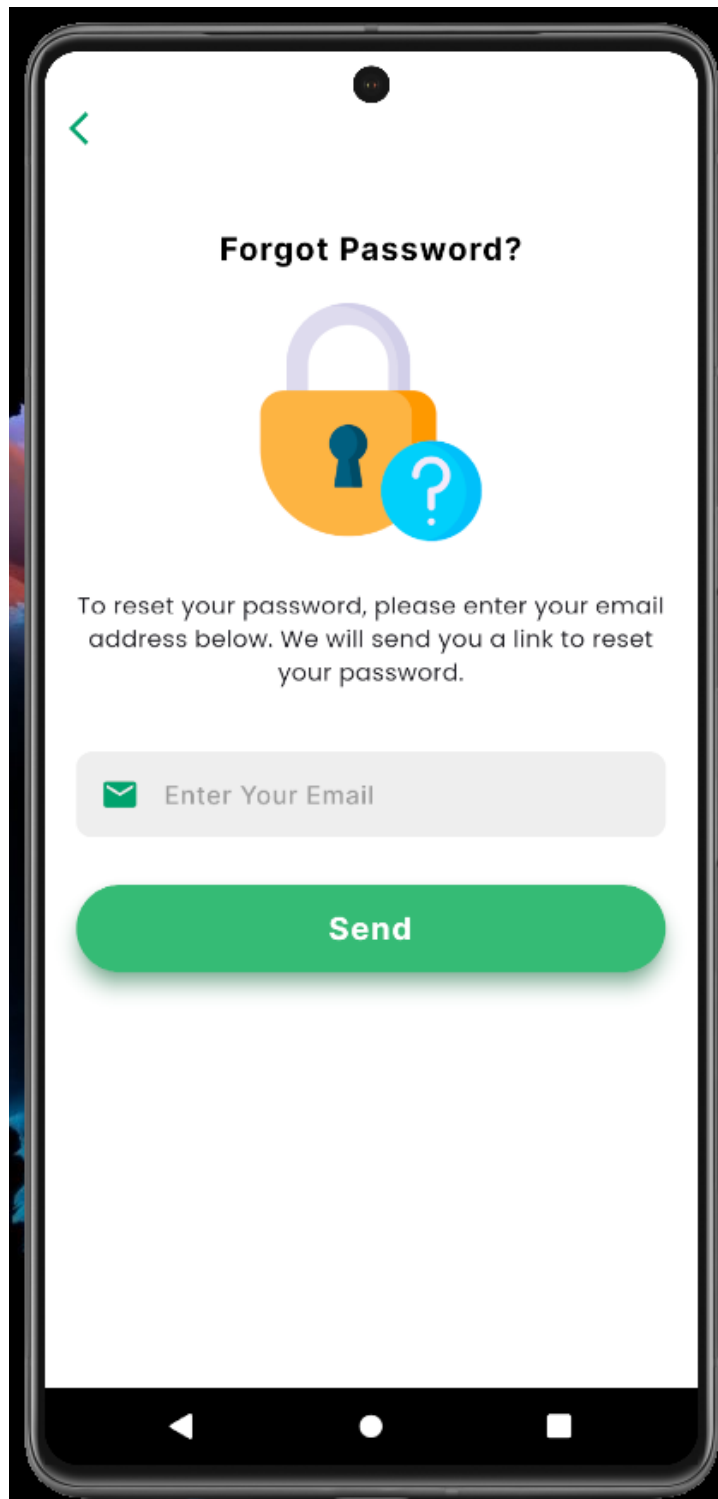
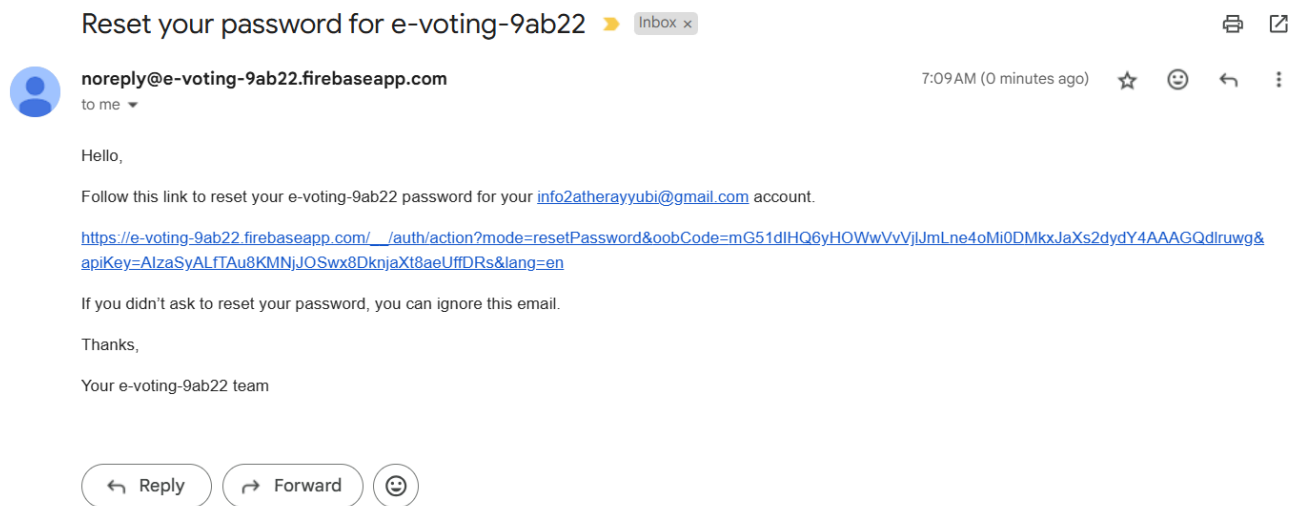


Figure 4.6: Forgot Password Screen

## Reset Password Steps:

### Step 1:

User will receive an email containing the link of password reset as shown below:




### Step 2:

After clicking on link the following box will be shown in the new tab of the browser. After clicking on save button next box will appear to show a message that password has been changed now he/she can login with his/her new password.

Reset your password

for info2atherayyubi@gmail.com

New password  
..... 

SAVE

Password changed

You can now sign in with your new password

### 4.3.3 Admin Dashboard

Admin dashboard contains organization, election, candidate and voter ID screens from here admin can manage all these items. All of these screens shown below:

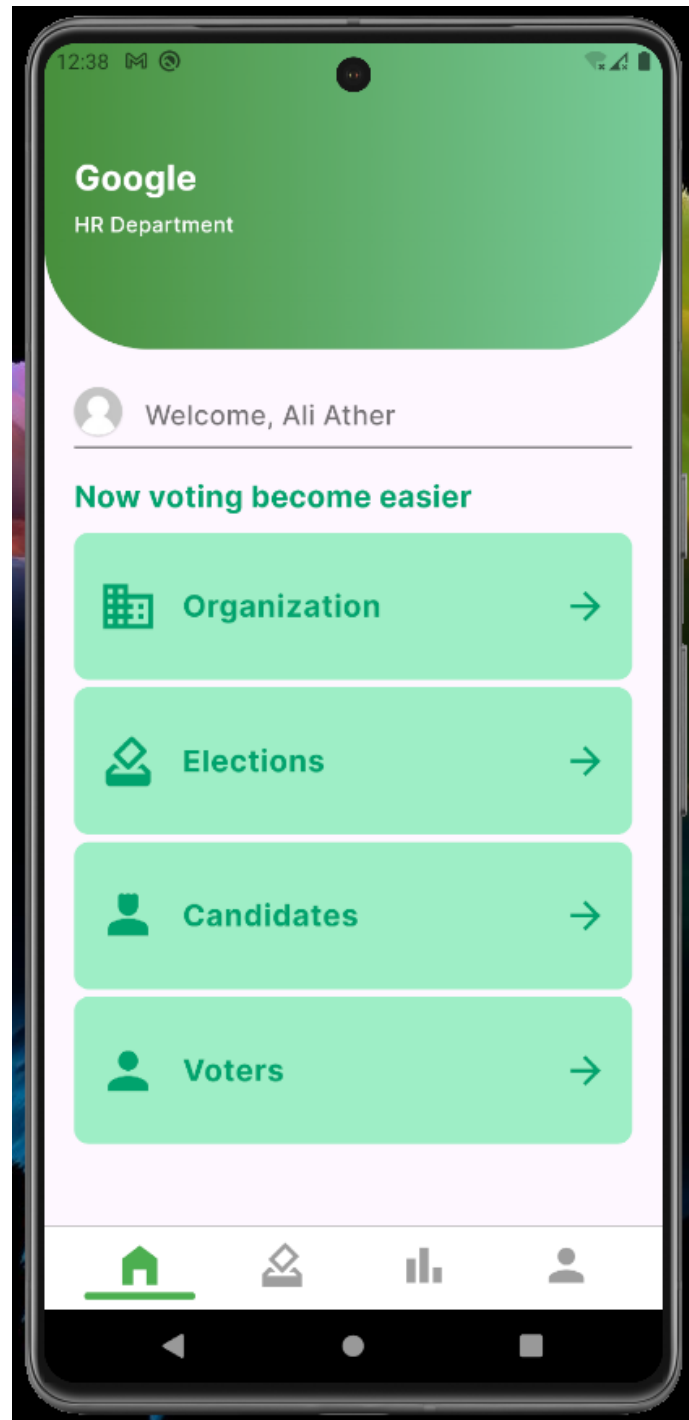


Figure 4.7: Admin Dashboard Screen

### 4.3.3.1 Admin Organization

#### Organization Management Screen

This screen allows administrators to create, view and update organization. Each organization includes details such as name, address, and description. Once the user create organization he can't delete it, he can only edit and read it.

#### Add Organization

This screen shows the form for creating a new organization, where admins can enter the organization's name, address, and description.

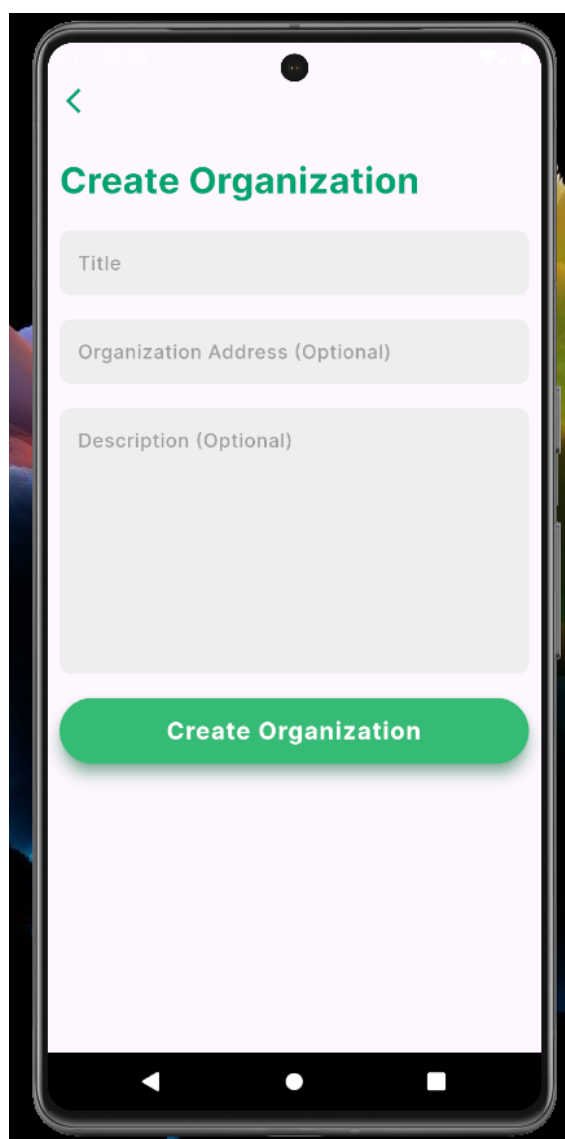


Figure 4.8: Organization Screen



## Edit Organization

Once the user add details of organization after that he can edit details by tapping the edit button and after that admin can edit details as shown below:

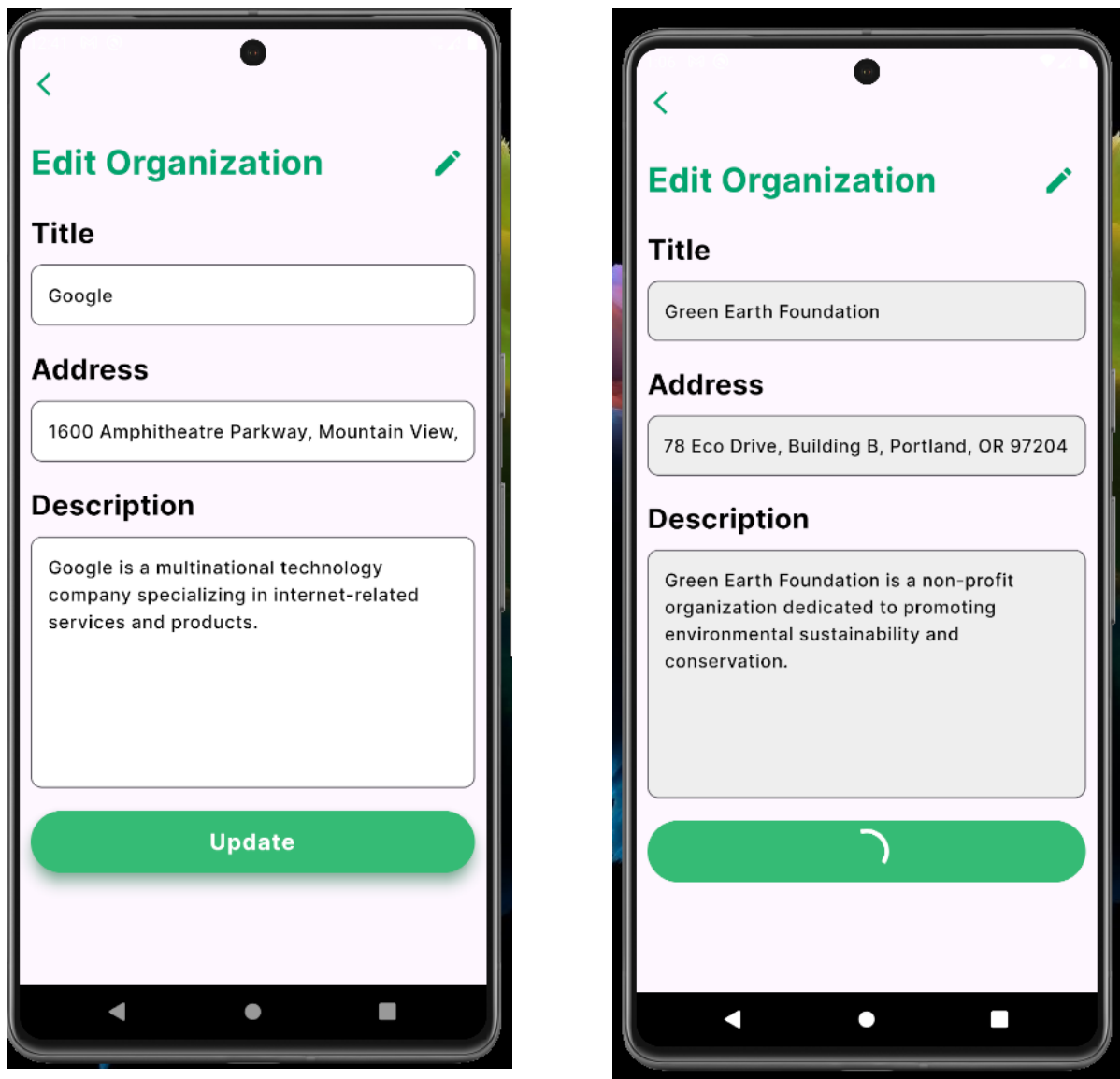


Figure 4.9: Organization Edit Screen

