

ONLINE ELECTION SYSTEM

(CS-165 Software Engineering Lab Semester Project Fall 2023)



Session: 2021-2025

Project Supervisor

Sir Laeeq Khan Niazi

Project Developer

Ali Haider

2021-CS-38

Contents

1	Introduction	1
1.1	Background	1
1.2	Objectives	1
1.3	Scope	1
1.4	Features	1
2	Diagrams with Explanation	3
2.1	Use Case Diagram	3
2.2	Class Diagram	3
2.3	Database Schema Diagram	4
2.4	Sequence Diagram	4
2.5	DFD Level 0	5
2.6	DFD Level 1	5
2.7	Architecture Diagram	5
3	Project Implementation Details	6
3.1	Best Practices Adopted	6
3.1.1	Naming Convention	6
3.1.2	Model-Controller-Route (MVR) Structure	6
3.1.3	Commenting Guidelines	7
3.2	Communication Tools	7
3.2.1	Collaboration Platforms	7
3.2.2	Meeting Minute	8
3.3	Version Control	9
3.3.1	Git Repository Setup	9
3.3.2	Commit Guidelines	9
3.4	Project Management	10
3.4.1	Task Breakdown	10
4	Application Working	11
4.1	User Registration	11
4.1.1	Screenshot	11
4.1.2	Code Snippet	11
4.2	User Login	12
4.2.1	Screenshot	12
4.2.2	Code Snippet	12
4.3	Admin Dashboard	13
4.3.1	Screenshot	13
4.3.2	Code Snippet	13
4.4	Admin Election	14
4.4.1	Screenshot	14
4.4.2	Code Snippet	14
4.5	Admin Party	15
4.5.1	Screenshot	15
4.5.2	Code Snippet	15
4.6	Admin Candidate	16
4.6.1	Screenshot	16
4.6.2	Code Snippet	16

4.7	Admin Election Candidate	17
4.7.1	Screenshot	17
4.7.2	Code Snippet	17
4.8	Admin Verification	18
4.8.1	Screenshot	18
4.8.2	Code Snippet	18
4.9	Admin Account	19
4.9.1	Screenshot	19
4.9.2	Code Snippet	19
4.10	User Dashboard	20
4.10.1	Screenshot	20
4.10.2	Code Snippet	20
4.11	User Cast Vote	21
4.11.1	Screenshot	21
4.11.2	Code Snippet	21

5	Conclusion	22
----------	-------------------	-----------

List of Figures

1	Use Case Diagram	3
2	Class Diagram	3
3	Database Schema Diagram	4
4	Sequence Diagram	4
5	DFD Level 0	5
6	DFD Level 1	5
7	Architecture Diagram	6
8	Naming Convention	6
9	MVR	7
10	Comments Guidelines	7
11	Slack	8
12	Two Meeting Minutes	8
13	Github	9
14	Commit Style	9
15	Clickup	10
16	User Registration	11
17	User Registration	11
18	User Registration	12
19	User Login	12
20	Admin Dashboard	13
21	Admin Dashboard Code	13
22	Admin Election	14
23	Admin Election Code	14
24	Admin Party	15
25	Admin Party Code	15
26	Admin Candidate	16
27	Admin Candidate Code	16
28	Admin Election Candidate	17
29	Admin Candidate Code	17
30	Admin Verification	18

31	Admin Verification Code	18
32	Admin Account	19
33	Admin Account Code	19
34	User Dashboard	20
35	User Dashboard Code	20
36	User Cast Vote	21
37	User Cast Vote Code	21

1 Introduction

1.1 Background

The Online Election System project aims to revolutionize the electoral process, providing a user-friendly platform for voters and candidates. This web application strives to create a seamless connection between users, offering a feature-rich environment for candidacy registration, voter information, and secure voting. Advanced functionalities are integrated to ensure transparency, trust, and integrity of the electoral process within a digital framework.

1.2 Objectives

The objective of the Online Election System project is to revolutionize the electoral experience by creating an intuitive and accessible web platform. This system aims to simplify candidacy registration, provide comprehensive voter information, and facilitate a secure and transparent voting process. By employing robust security measures, ensuring ease of use, and fostering transparency, the project strives to enhance participation, engagement, and trust in the democratic process while offering a seamless digital framework for elections.

1.3 Scope

The scope of the Online Election System project encompasses the development of a comprehensive web-based platform facilitating candidate registration, voter information dissemination, and a secure voting mechanism. It includes the creation of user-friendly interfaces for candidates to register their candidacies securely, a database system for storing and presenting voter information, and the implementation of encryption and authentication protocols to ensure the integrity and confidentiality of the voting process. Additionally, the project involves creating administrative tools for system management, monitoring, and result tabulation while prioritizing scalability, accessibility, and adherence to legal and ethical standards governing elections within the digital landscape.

1.4 Features

1. User Authentication and Profile Management:

- Users can register and log into the system securely, manage their profiles, and modify account information.

2. Election Management (CRUD Operations):

- Admin has full access to create, read, update, and delete (CRUD) operations for managing elections.

3. Candidate Management (CRUD Operations):

- Admin can perform CRUD operations for managing candidates participating in elections.

4. Election Results and Statistics:

- Admin can view election details, voter lists, candidate statistics, and check winners.

5. System Activity Logs:

- Admin can maintain logs of all activities within the system.

6. **Voter Eligibility Verification:**
 - Admin verifies voter eligibility based on age, citizenship, and residence.
7. **Multilingual Support:**
 - Users can choose their preferred language for the website interface.
8. **Ongoing Elections Details:**
 - Users can view information about ongoing elections, including details and candidate lists.
9. **Candidate Details and Voting:**
 - Users can access candidate details and cast their votes securely.
10. **Vote Confirmation Receipt:**
 - Users can generate a receipt confirming their vote.
11. **Chatbot Assistance:**
 - Users have access to a chatbot for assistance or queries regarding the election process.
12. **Profile Language Selection:**
 - Users can choose their preferred language for the website interface.
13. **Password Modification (User):**
 - Users can change their passwords securely.
14. **View Election Logs (Admin):**
 - Admin can access and view logs of all system activities related to elections and user interactions.
15. **Voter Helpdesk (User):**
 - Users have access to a helpdesk for assistance during the voting process.
16. **Real-time Voter Status (Admin):**
 - Admin can check the real-time status of voters, including who has cast their votes and who hasn't.
17. **Comprehensive User Activity Tracking (Admin):**
 - Admin can track and maintain logs of all user activities within the system.

2 Diagrams with Explanation

2.1 Use Case Diagram

The use case diagram illustrates the various interactions between users and the system. It provides a high-level view of the system's functionalities and the actors involved. Refer to Figure 1 for the visual representation.

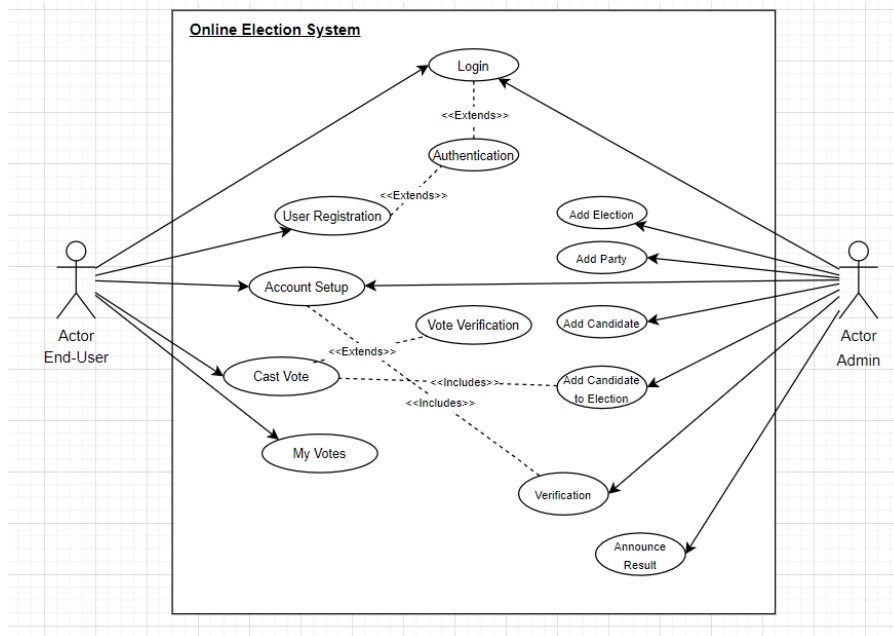


Figure 1: Use Case Diagram

2.2 Class Diagram

The class diagram models the system's structure by depicting the classes, their attributes, and the relationships between them. Figure 2 visually represents the class diagram.

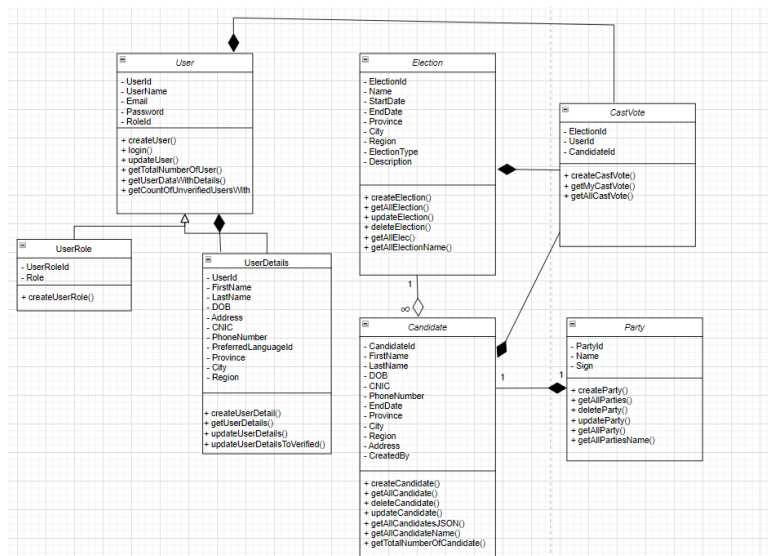


Figure 2: Class Diagram

2.3 Database Schema Diagram

The database schema diagram depicts the structure of the database, including tables, relationships, and key constraints. Figure 3 provides a visual representation of the project database schema.

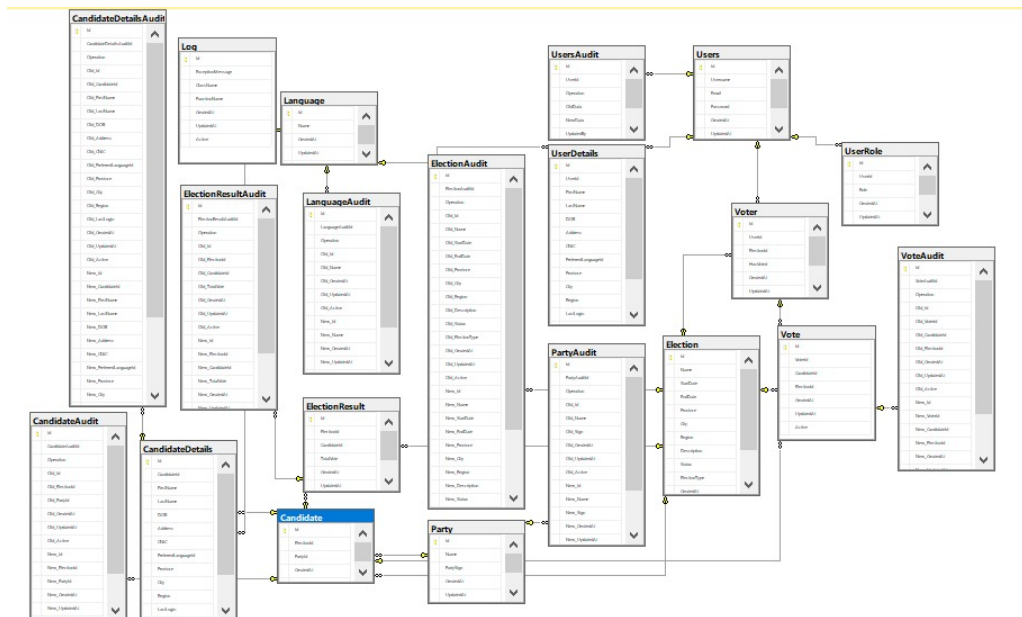


Figure 3: Database Schema Diagram

2.4 Sequence Diagram

The sequence diagram captures the dynamic behavior of the system by illustrating the sequence of interactions between objects over time. See Figure 4 for a visual representation.

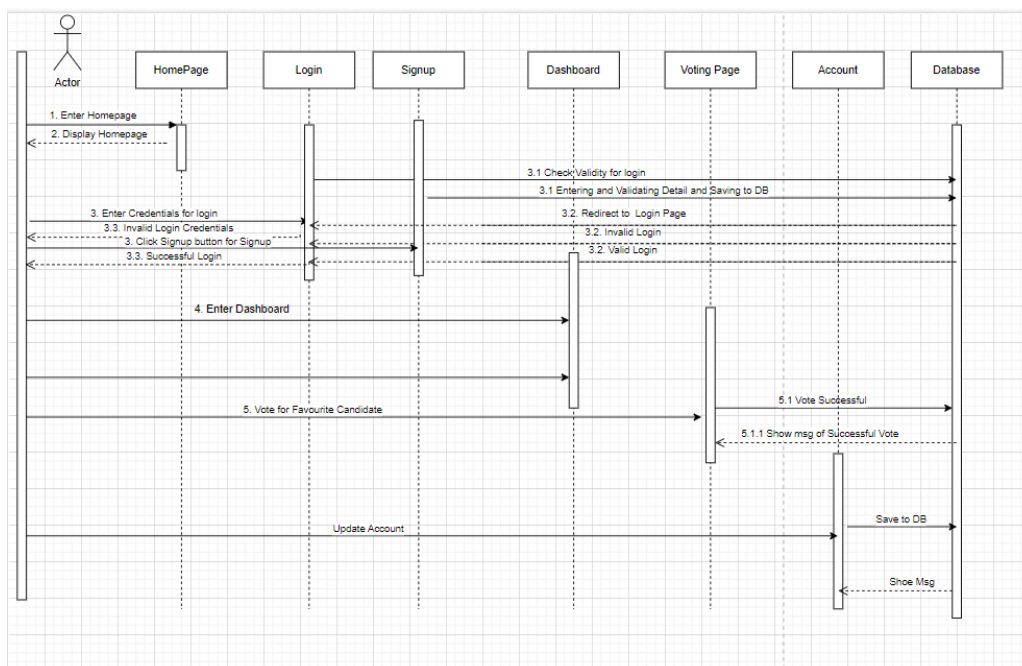


Figure 4: Sequence Diagram

2.5 DFD Level 0

The Level 0 Data Flow Diagram provides a holistic view of the system, showcasing the major processes and data flows. Figure 5 visually represents this high-level overview.

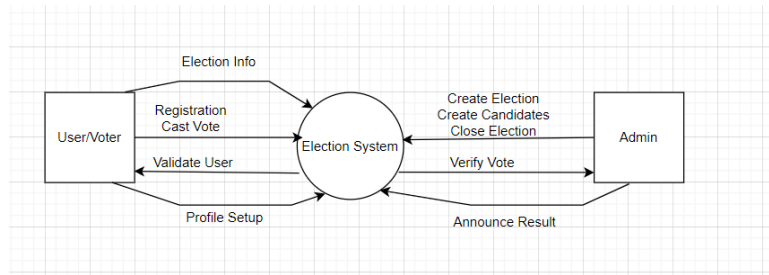


Figure 5: DFD Level 0

2.6 DFD Level 1

The Level 1 Data Flow Diagram delves into more detail, breaking down processes into sub-processes and illustrating the data flows between them. Refer to Figure 6 for the visual representation.

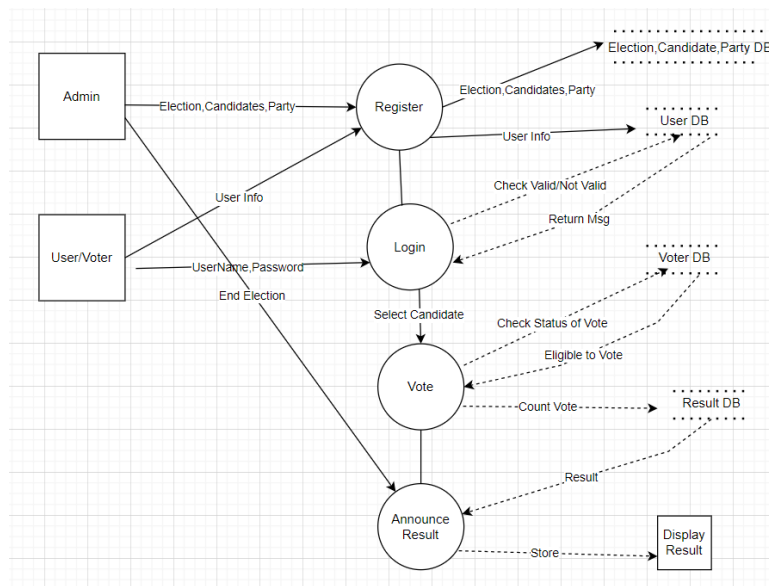


Figure 6: DFD Level 1

2.7 Architecture Diagram

The architecture diagram provides an overview of the system's structure and how its components interact. Figure 7 visually represents the architecture of the PakCars system.

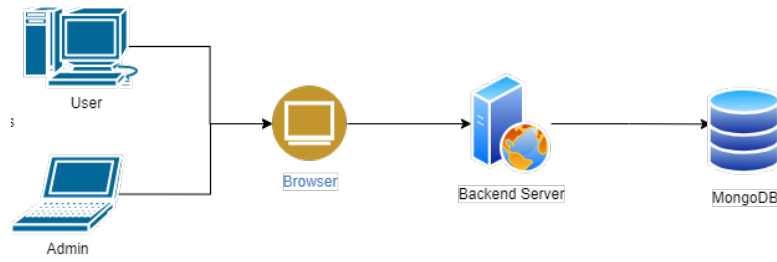


Figure 7: Architecture Diagram

3 Project Implementation Details

3.1 Best Practices Adopted

3.1.1 Naming Convention

The codebase maintains clear and organized naming rules to make code easier to read and manage. This involves using descriptive names for variables, functions, and classes. In JavaScript, consistency with CamelCase helps keep everything uniform across the entire project.

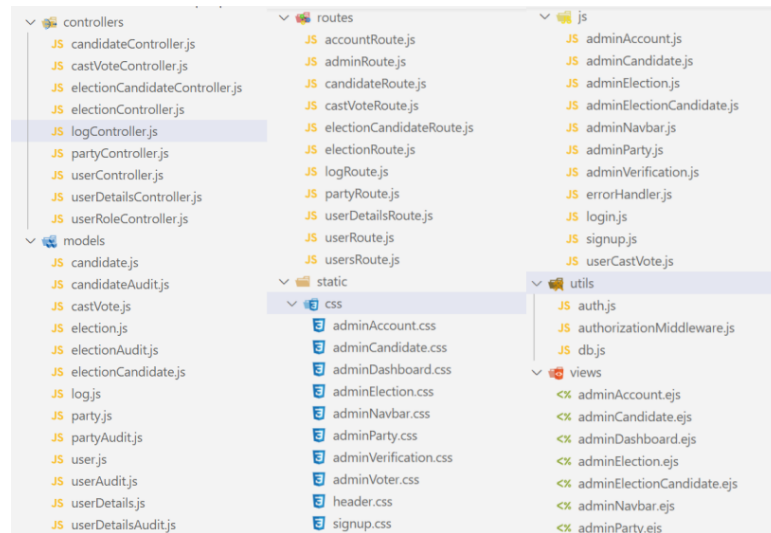


Figure 8: Naming Convention

3.1.2 Model-Controller-Route (MVR) Structure

The project follows a logical organization called the Model-Controller-Route (MVR) structure. This structure neatly separates data models (Model), data controllers (View), and API routes (Routes). By doing so, it improves code modularity and makes maintaining the project simpler.

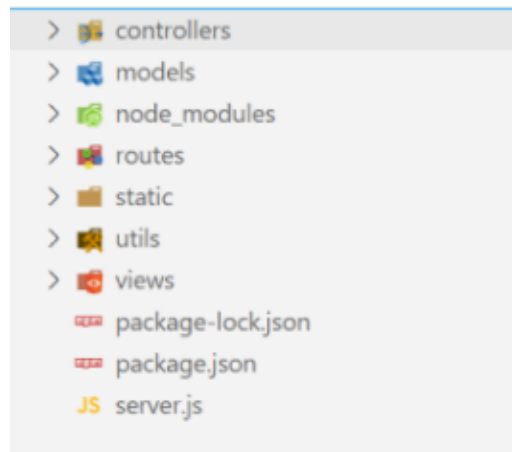


Figure 9: MVR

3.1.3 Commenting Guidelines

The codebase includes thorough and informative comments to clarify the functionality of different components. Inline comments help explain intricate algorithms, while block comments offer insights into the overall purpose of files. These comments foster collaboration among developers and enhance comprehension of the codebase

```
const getAllElectionName = async () => {
  try {
    const elections = await Election.find({}).populate('CreatedBy', 'UserName');

    // Mapping the elections to create an array of objects
    const formattedElections = elections.map(election => ({
      id: election._id, // Assuming _id is the ID field of the Election model
      formattedValue: `${election.Name},${election.ElectionType}`
    }));

    return formattedElections; // Return the array of objects
  } catch (err) {
    throw new Error(err.message);
  }
};
```

Figure 10: Comments Guidelines

3.2 Communication Tools

3.2.1 Collaboration Platforms

Slack is used as the primary collaboration platform to facilitate real-time communication. Slack provides a centralized space for project members to communicate efficiently, share updates, and engage in discussions. Channels are organized based on project modules and functionalities, ensuring focused and effective communication.

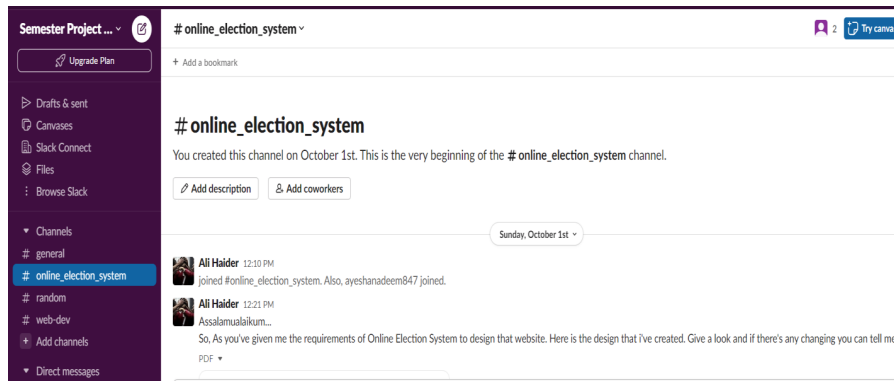


Figure 11: Slack

3.2.2 Meeting Minute

Two meetings are conducted physically to address specific needs during critical project phases. These short and focused meetings aim to gather requirements and discuss key aspects of the project efficiently. The first meeting was dedicated to Requirements Gathering, ensuring a clear understanding of project objectives. The second meeting focused on discussing the final technology stack and, asking some queries.

Meeting Minutes

Date: September 11, 2023. Time: 1:30PM to 1:45PM (15Minutes)

Project Title: Online Election System

Members:

Registration No	Name	Attendance/ Remark
2021-CS-09	Ayesha Nadeem	Present Owner
2021-CS-38	Ali Haider	Present Developer

Type of Meeting: Physical
Location: Lab 3 Computer Science Department UET Lahore

Agenda:

- Information gathering
- Admin can add other admin.
- Should be responsive.
- User authentication
- Notification or email should be sent.
- Report.
- History should be saved.
- Any web technology can be used.
- Discussed issues and concerns.
- Discuss the specific goals and objectives of the project.

Discussion or Feedback:

Admin will be responsible for controlling the application. He can add others admin to help. After authentication, the user will be redirected to their modules (Admin or user). Admin will be responsible for adding new election and candidate to the election. End-users can select an election and vote for their favorite candidate. End-users can only vote one candidate per election. Admin will be responsible for declaring the result and after announcing the result notification will be sent to the user who has voted. End-user voting history will also be saved. Admin can also generate different reports. Also, the application should be responsible for meeting and module. Any web technology can be used.

Signature:

Owner

Ayesha Nadeem

Developer

Ali Haider

Meeting Minutes

Date: 08/11/2023 Time: 11:00 pm to 11:10 pm

Project Title: Online Election System

Members:

Registration No	Name	Attendance/ Remark
2021-CS-09	Ayesha Nadeem	Present Owner
2021-CS-38	Ali Haider	Present Developer

Type of Meeting: Physical
Location: Lecture Theatre Computer Science Department UET Lahore

Agenda:

- Take only necessary information and then give option to complete their profile.
- Decided to make project in node, express and MongoDB.
- Frontend in HTML, CSS, and bootstrap
- Added option to verify user when user complete their profile in admin module.

Signature:

Owner

Developer

Figure 12: Two Meeting Minutes

3.3 Version Control

3.3.1 Git Repository Setup

The project source code is hosted on a Git repository, providing a centralized location for version control. The repository is structured with separate branches for development, testing, and production. Access controls are implemented to manage contributions effectively.

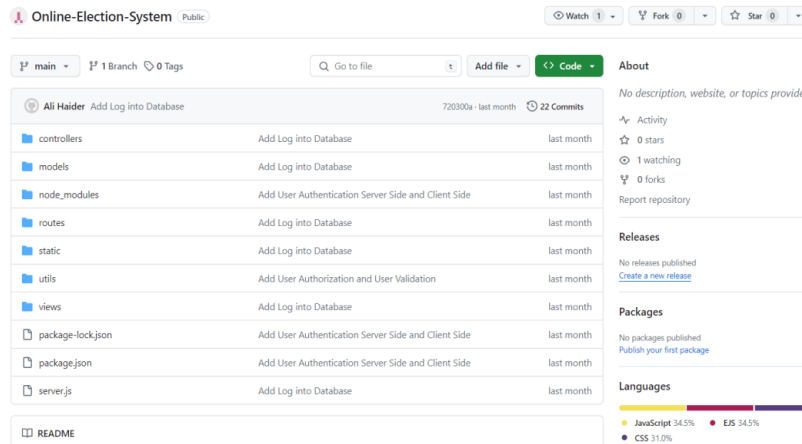


Figure 13: Github

3.3.2 Commit Guidelines

A standardized commit message format is followed to maintain a clear and descriptive version history. Each commit message includes a concise summary of the changes, an optional detailed description, and references to relevant issues or tasks.

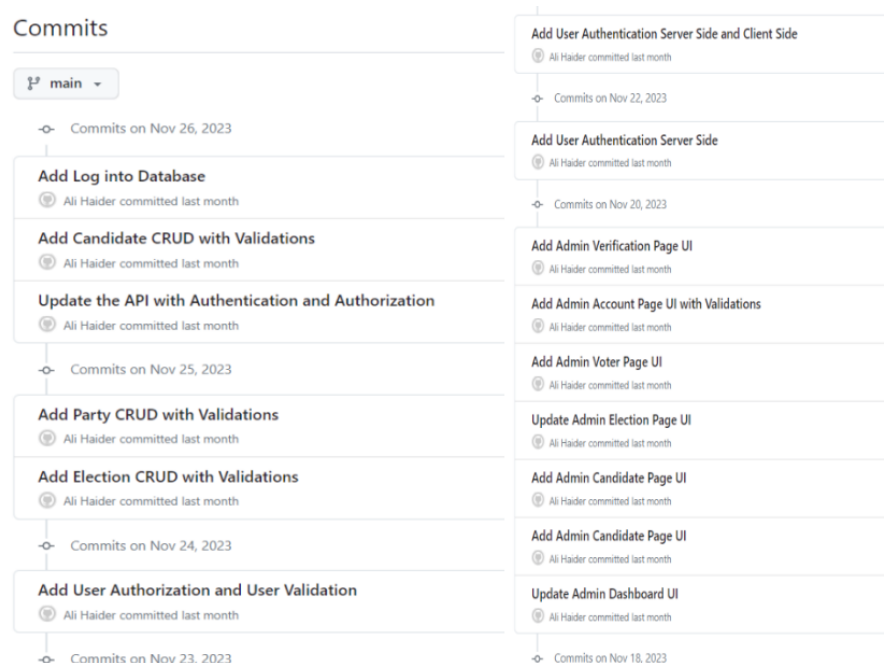


Figure 14: Commit Style

3.4 Project Management

3.4.1 Task Breakdown

Tasks are broken down into manageable units to facilitate efficient development. Each task is assigned a priority, estimated time, and responsible team member. This breakdown ensures a clear understanding of project progress and individual responsibilities.

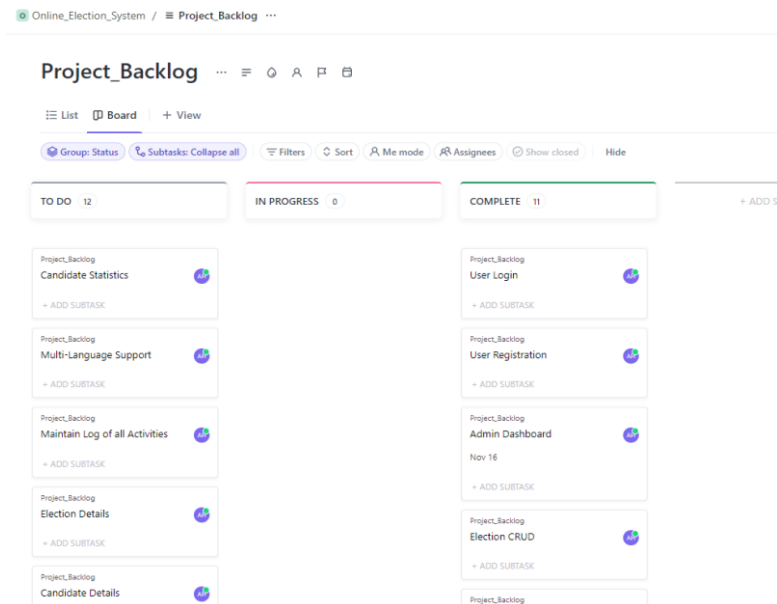


Figure 15: Clickup

4 Application Working

4.1 User Registration

4.1.1 Screenshot

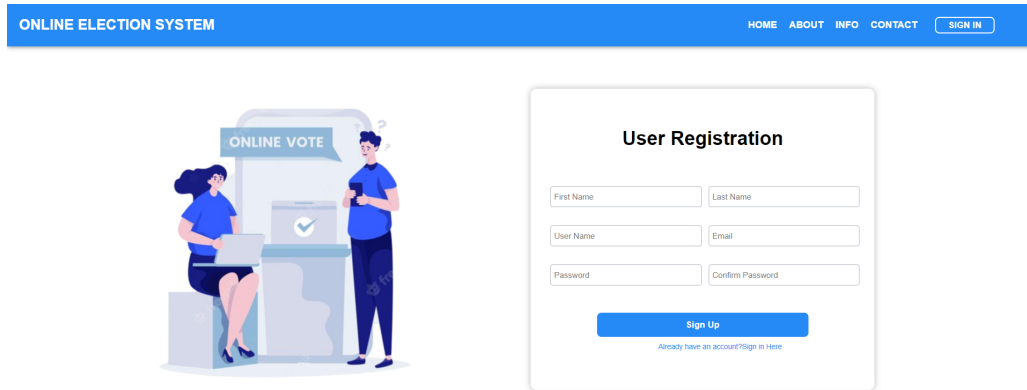


Figure 16: User Registration

4.1.2 Code Snippet

```
async function createUser(req, res) {
  try {
    const { firstName, lastName, userName, email, password, confirmPassword } = req.body;
    console.log(firstName, lastName, userName, email, password, confirmPassword)
    // Check if passwords match
    if (password !== confirmPassword) {
      return res.status(400).json({ error: 'Passwords do not match' });
    }

    // Create a new user instance based on the User schema
    const newUser = new User({
      UserName: userName,
      Email: email,
      Password: password
    });

    // Save the new user to the database
    await newUser.save();
    // const userId = newUser._id;

    const newDetails = new UserDetail({
      UserId: newUser._id,
      FirstName: firstName,
      LastName: lastName
    });
    await newDetails.save();

    setTimeout(() => {
      res.redirect("/");
    }, 5000);
  } catch (err) {
    res.status(500).json({ error: err.message });
  }
}
```

Figure 17: User Registration

4.2 User Login

4.2.1 Screenshot

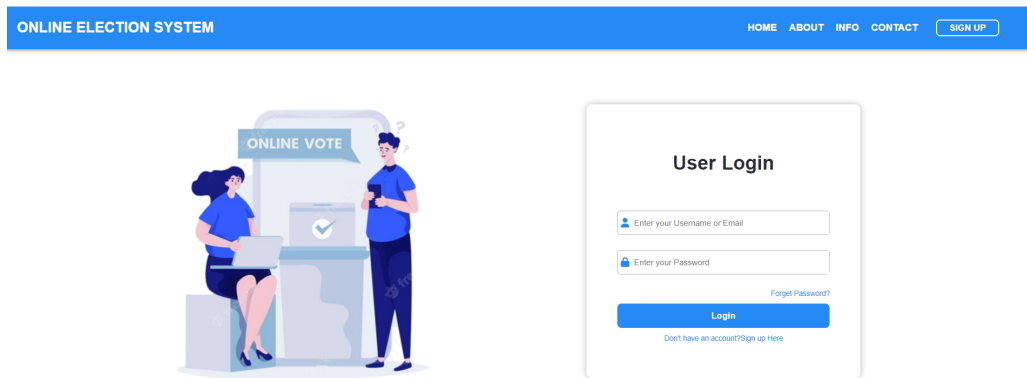


Figure 18: User Registration

4.2.2 Code Snippet

```
async function login(req, res) {
  const { UserName, Password } = req.body;
  console.log(UserName, Password)

  try {
    // Search for the user by username or email
    const user = await User.findOne({
      $or: [{ UserName }, { Email: UserName }]
    });

    if (!user) {
      return res.status(404).json({ error: 'User not found' });
    }

    // Check if the passwords match
    if (user.Password !== Password) {
      // return res.status(404).json({ error: 'User not found' });
      return res.render('LandingNavbar', { page: 'login', navbarButtonText: 'Sign In' });
    }

    // If passwords match, generate a token
    //const token = GenerateToken(user);

    // Prepare response data with token and user info
    let responseData = {
      message: 'Logged in successfully',
    };
    if (user.RoleId === 1) {
      // Redirect to admin dashboard
      user.Role = 'User';
      responseData.isAdmin = false;
      responseData.Role = 'User';
    }
  }
}
```

Figure 19: User Login

4.3 Admin Dashboard

4.3.1 Screenshot

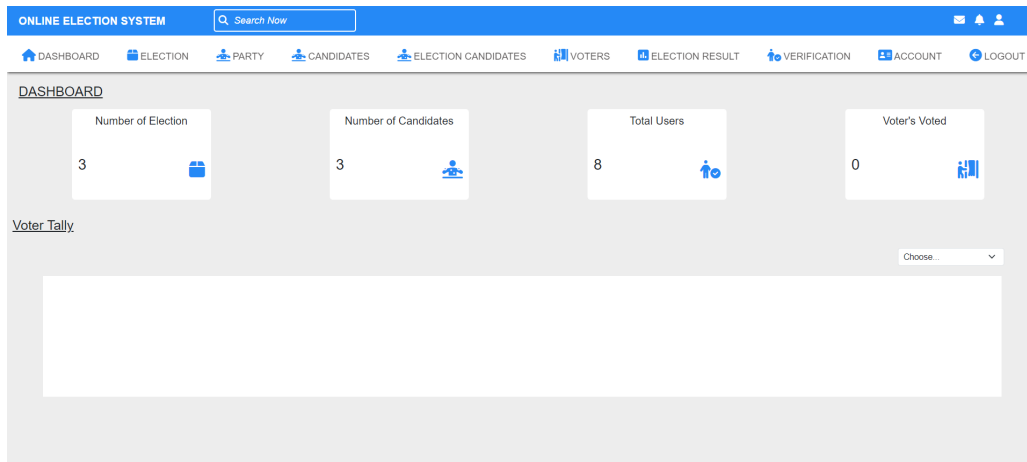


Figure 20: Admin Dashboard

4.3.2 Code Snippet

```
<link rel="stylesheet" href="/css/adminDashboard.css">
<section class="box-data">
  <h1 class="headingText">DASHBOARD</h1>
  <div class="boxes">
    <div class="box">
      <div class="boxHeading">Number of Election</div>
      <div class="Number-icon">
        <p id="NumberOfElection" class="tNumber"><%= telections %></p>
        <i class="fa-solid fa-box" style="color: #2589f6; font-size: 30px;"></i>
      </div>
    </div>
    <div class="box">
      <div class="boxHeading">Number of Candidates</div>
      <div class="Number-icon">
        <p id="NumberOfCandidate" class="tNumber"><%= telections %></p>
        <i class="fa-solid fa-person-dots-from-line" style="color: #2589f6; font-size: 30px;"></i>
      </div>
    </div>
    <div class="box">
      <div class="boxHeading">Total Users</div>
      <div class="Number-icon">
        <p id="NumberOfVoter" class="tNumber"><%= tUser %></p>
        <i class="fa-solid fa-person-circle-check" style="color: #2589f6; font-size: 30px;"></i>
      </div>
    </div>
    <div class="box">
      <div class="boxHeading">Voter's Voted</div>
      <div class="Number-icon">
        <p id="NumberOfVoterVoted" class="tNumber">0</p>
        <i class="fa-solid fa-person-booth" style="color: #2589f6; font-size: 30px;"></i>
      </div>
    </div>
  </div>
</section>
```

Figure 21: Admin Dashboard Code

4.4 Admin Election

4.4.1 Screenshot

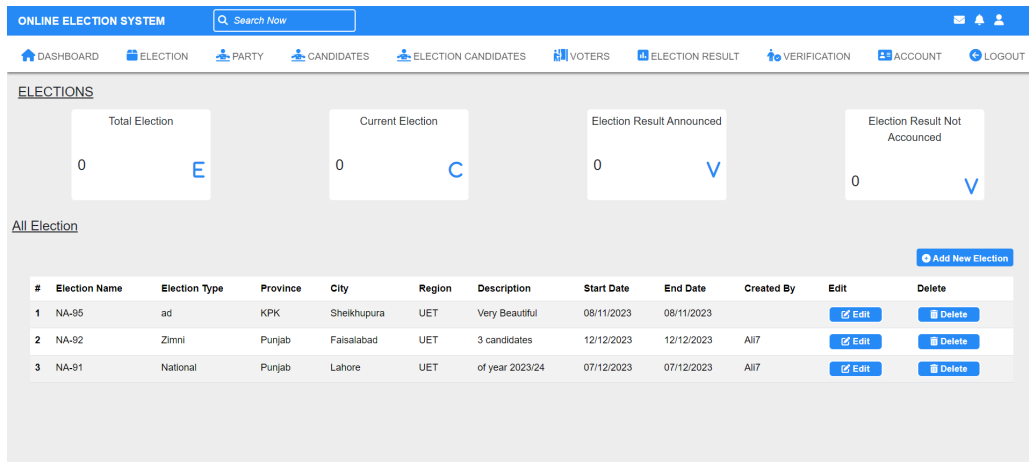


Figure 22: Admin Election

4.4.2 Code Snippet

```
<link rel="stylesheet" href="/css/adminElection.css">

<section class="box-data">
  <h1 class="headingText">ELECTIONS</h1>
  <div class="boxes">
    <div class="box">
      <div class="boxHeading">Total Election</div>
      <div class="Number-icon">
        <p id="totalElection" class="tNumber">0</p>
        <i class="fa-solid fa-e" style="color: #2589f6; font-size: 30px;"></i>
      </div>
    </div>
    <div class="box">
      <div class="boxHeading">Current Election</div>
      <div class="Number-icon">
        <p id="currentElection" class="tNumber">0</p>
        <i class="fa-solid fa-c" style="color: #2589f6; font-size: 30px;"></i>
      </div>
    </div>
    <div class="box">
      <div class="boxHeading">Election Result Announced</div>
      <div class="Number-icon">
        <p id="electionResultAnnounced" class="tNumber">0</p>
        <i class="fa-solid fa-v" style="color: #2589f6; font-size: 30px;"></i>
      </div>
    </div>
    <div class="box">
      <div class="boxHeading">Election Result Not Announced</div>
      <div class="Number-icon">
        <p id="electionResultNotAnnounced" class="tNumber">0</p>
        <i class="fa-solid fa-v" style="color: #2589f6; font-size: 30px;"></i>
      </div>
    </div>
  </div>
</div>
```

Figure 23: Admin Election Code

4.5 Admin Party

4.5.1 Screenshot

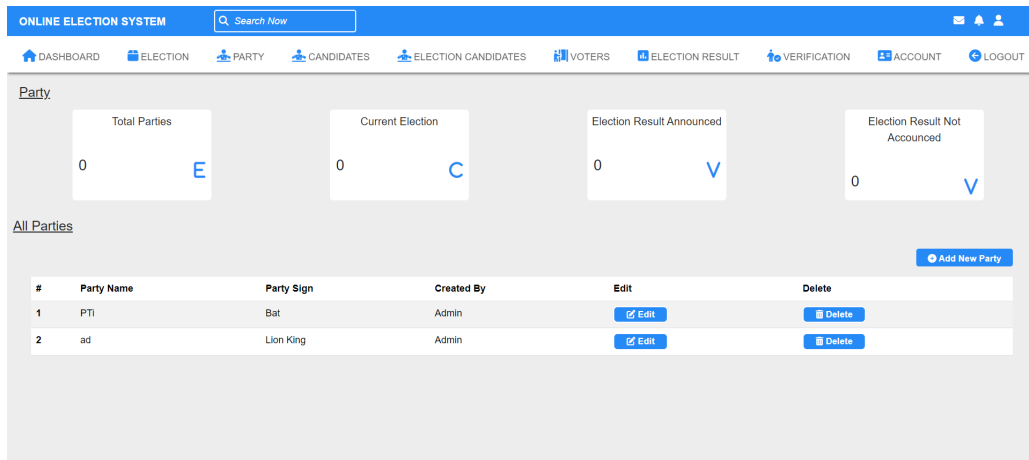


Figure 24: Admin Party

4.5.2 Code Snippet

```
<link rel="stylesheet" href="/css/adminParty.css">

<section class="box-data">
  <h1 class="headingText">Party</h1>
  <div class="boxes">
    <div class="box">
      <div class="boxHeading">Total Parties</div>
      <div class="Number-icon">
        <p id="totalParties" class="tNumber">0</p>
        <i class="fa-solid fa-e" style="color: #2589f6; font-size: 30px;"></i>
      </div>
    </div>
    <div class="box">
      <div class="boxHeading">Current Election</div>
      <div class="Number-icon">
        <p id="currentElection" class="tNumber">0</p>
        <i class="fa-solid fa-c" style="color: #2589f6; font-size: 30px;"></i>
      </div>
    </div>
    <div class="box">
      <div class="boxHeading">Election Result Announced</div>
      <div class="Number-icon">
        <p id="electionResultAnnounced" class="tNumber">0</p>
        <i class="fa-solid fa-v" style="color: #2589f6; font-size: 30px;"></i>
      </div>
    </div>
    <div class="box">
      <div class="boxHeading">Election Result Not Announced</div>
      <div class="Number-icon">
        <p id="electionResultNotAnnounced" class="tNumber">0</p>
        <i class="fa-solid fa-v" style="color: #2589f6; font-size: 30px;"></i>
      </div>
    </div>
  </div>
</div>
```

Figure 25: Admin Party Code

4.6 Admin Candidate

4.6.1 Screenshot

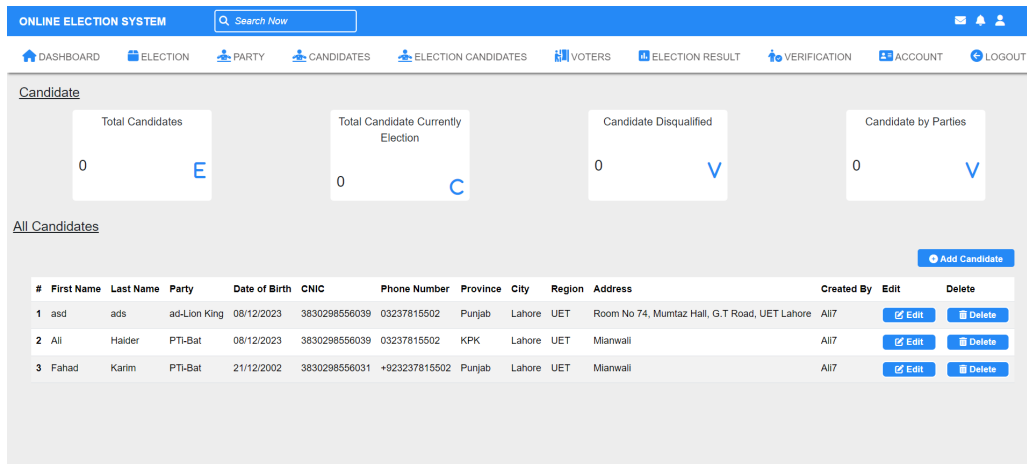


Figure 26: Admin Candidate

4.6.2 Code Snippet

```
<link rel="stylesheet" href="/css/adminCandidate.css">

<section class="box-data">
  <h1 class="headingText">Candidate</h1>
  <div class="boxes">
    <div class="box">
      <div class="boxHeading">Total Candidates</div>
      <div class="Number-icon">
        <p id="totalElection" class="tNumber">0</p>
        <i class="fa-solid fa-e" style="color: #2589f6; font-size: 30px;"></i>
      </div>
    </div>
    <div class="box">
      <div class="boxHeading">Total Candidate Currently Election</div>
      <div class="Number-icon">
        <p id="currentElection" class="tNumber">0</p>
        <i class="fa-solid fa-c" style="color: #2589f6; font-size: 30px;"></i>
      </div>
    </div>
    <div class="box">
      <div class="boxHeading">Candidate Disqualified</div>
      <div class="Number-icon">
        <p id="electionResultAnnounced" class="tNumber">0</p>
        <i class="fa-solid fa-v" style="color: #2589f6; font-size: 30px;"></i>
      </div>
    </div>
    <div class="box">
      <div class="boxHeading">Candidate by Parties</div>
      <div class="Number-icon">
        <p id="electionResultNotAnnounced" class="tNumber">0</p>
        <i class="fa-solid fa-v" style="color: #2589f6; font-size: 30px;"></i>
      </div>
    </div>
  </div>
</section>

<section class="voterTally">
  <h1 class="headingText">All Candidates</h1>
  <div class="container">
    <button type="button" class="btn btn-primary btn-sm id="btnAddNewCandidate"><i class="fa-solid fa-circle-plus" style="color: #ffffff;"></i> Add Candidate</button>
  </div>
</section>
```

Figure 27: Admin Candidate Code

4.7 Admin Election Candidate

4.7.1 Screenshot

#	Candidate Name	Election Name	Election Type	Province	City	Region	Description	Start Date	End Date	Created By	Edit	Delete
1		NA-95	ad	KPK	Sheikhupura	UET	Very Beautiful	08/11/2023	11/11/2023	All7	Edit	Delete
2	Fahad Karim	NA-92	Zimni	Punjab	Faisalabad	UET	3 candidates	12/12/2023	25/12/2023	All7	Edit	Delete

Figure 28: Admin Election Candidate

4.7.2 Code Snippet

```
<link rel="stylesheet" href="/css/adminElection.css">

<section class="box-data box-data2">
  <form class="row g-3 needs-validation" action="" method="post" novalidate>
    <div class="col-md-12">
      <label for="validationCustom01" class="form-label">Election</label>
      <select class="form-select select-text" name="election" id="validationCustom01" required>
        <option selected disabled value="">Choose...</option>
        <% elections.forEach(election=> { %>
          <option value="<%= election.id %%">
            <%= election.formattedValue %>
          </option>
        <% }); %>
      </select>
      <div class="invalid-feedback">
        Please select a valid Election.
      </div>
    </div>

    <div class="col-md-12">
      <label for="validationCustom02" class="form-label">Candidate</label>
      <select class="form-select select-text" name="candidate" id="validationCustom02" required>
        <option selected disabled value="">Choose...</option>
        <% candidates.forEach(candidate=> { %>
          <option value="<%= candidate.id %%">
            <%= candidate.formattedName %>
          </option>
        <% }); %>
      </select>
      <div class="invalid-feedback">
        Please select a valid Candidate.
      </div>
    </div>

    <div class="col-12">
      <div class="text-end">
        <button onclick="addElectionCandidates()" class="btn btn-primary new-btn" id="btnSave" type="submit">Add</button>
      </div>
    </div>
  </form>
</section>
<div class="tableElection tableElectionCandidate">
  <table class="table table-striped">
```

Figure 29: Admin Candidate Code

4.8 Admin Verification

4.8.1 Screenshot

ONLINE ELECTION SYSTEM

Search Now

DASHBOARD ELECTION PARTY CANDIDATES ELECTION CANDIDATES VOTERS ELECTION RESULT VERIFICATION ACCOUNT LOGOUT

VERIFICATION

Verification Pending: 2

#	Name	UserName	Email	DOB	CNIC	Phone Number	Province	City	Region	Address	Province
1	Fahad Karim	Fahad123	Fahad568761@gmail.com								<button>✓ Verify</button>
2	Tahir Ali	Tahir123	Tahir568761@gmail.com								<button>✓ Verify</button>

Figure 30: Admin Verification

4.8.2 Code Snippet

```
<link rel="stylesheet" href="/css/adminVerification.css">

<section class="box-data">
  <h1 class="headingText">VERIFICATION</h1>
  <div class="tableVerification">
    <h1 class="headingText headingTextV">Verification Pending: <span id="tVerificationPending">{%= tUsers %}</span></h1>

    <table class="table table-striped">
      <thead>
        <tr>
          <th scope="col">#</th>
          <th scope="col">Name</th>
          <th scope="col">UserName</th>
          <th scope="col">Email</th>
          <th scope="col">DOB</th>
          <th scope="col">CNIC</th>
          <th scope="col">Phone Number</th>
          <th scope="col">Province</th>
          <th scope="col">City</th>
          <th scope="col">Region</th>
          <th scope="col">Address</th>
          <th scope="col">Province</th>
        </tr>
      </thead>
      <tbody>
        <% users.forEach((user, index) => { %>
          <tr id="userRow_<%= user._id %">
            <th scope="row"><%= index + 1 %></th>
            <td><%= user.Firstname + ' ' + user.Lastname %></td>
            <td><%= user.UserId && user.UserId.Username ? user.UserId.Username : '' %></td>
            <td><%= user.UserId && user.UserId.Email ? user.UserId.Email : '' %></td>
            <td><%= user.DOB ? user.DOB.toLocaleDateString() : '' %></td>
            <td><%= user.CNIC ? user.CNIC : '' %></td>
            <td><%= user.PhoneNumber ? user.PhoneNumber : '' %></td>
            <td><%= user.Province ? user.Province : '' %></td>
            <td><%= user.City ? user.City : '' %></td>
            <td><%= user.Region ? user.Region : '' %></td>
            <td><%= user.Address ? user.Address : '' %></td>
            <td>
              <button onclick="verifyUser('<%= user._id %>')" class="btn btn-primary btn-sm"><i class="fa-solid fa-check"></i> Veri
            </td>
          </tr>
        <% >
      </tbody>
    </table>
  </div>
</section>
```

Figure 31: Admin Verification Code

4.9 Admin Account

4.9.1 Screenshot

ONLINE ELECTION SYSTEM

Search Now

DASHBOARD ELECTION PARTY CANDIDATES ELECTION CANDIDATES VOTERS ELECTION RESULT VERIFICATION ACCOUNT LOGOUT

Account Information

First Name: Ali Last Name: Haider Date of Birth: 14/12/2023

Username: @ Ali7 Password: 12345

Email: haider5568761@gmail.com Present Address: Room No 74, Mumtaz Hall, G.T Road, UET Lahore

CNIC: 38302 Phone Number: 0323

Province: KPK City: Sargodha

Region: UET

Save

Figure 32: Admin Account

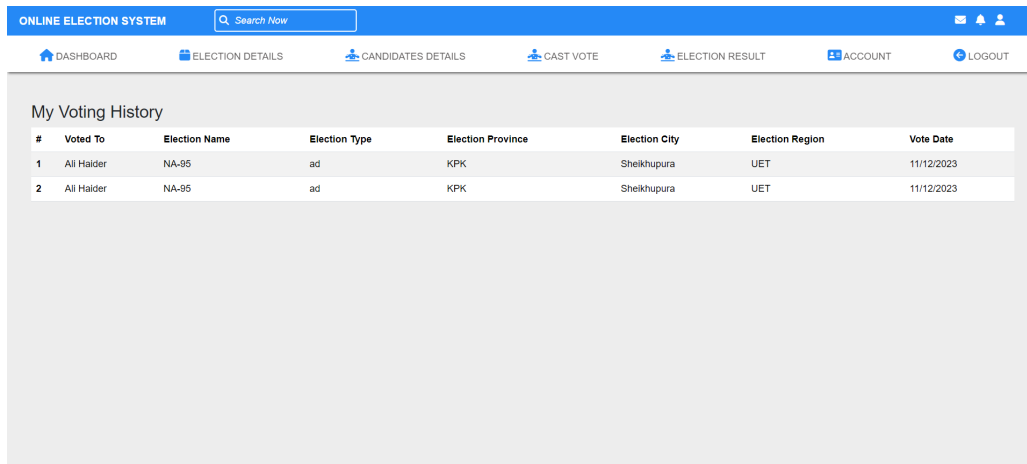
4.9.2 Code Snippet

```
<link rel="stylesheet" href="/css/adminAccount.css">
<section class="box-data">
  <h1 class="headingText">Account Information</h1>
  <div class="formContainer">
    <form class="row g-3 needs-validation" novalidate>
      <div class="col-md-4">
        <label for="validationCustom01" class="form-label">First Name</label>
        <input type="text" class="form-control" name="firstName" id="validationCustom01" value="% details[0].FirstName %" required>
        <div class="invalid-feedback">
          Please provide the first name.
        </div>
      </div>
      <div class="col-md-4">
        <label for="validationCustom02" class="form-label">Last Name</label>
        <input type="text" class="form-control" name="lastName" id="validationCustom02" value="% details[0].LastName %" required>
        <div class="invalid-feedback">
          Please provide the last name.
        </div>
      </div>
      <div class="col-md-4">
        <label for="validationCustom03" class="form-label">Date of Birth</label>
        <input type="date" class="form-control" name="dob" id="validationCustom03" value="% details[0].DOB %" required>
        <div class="invalid-feedback">
          Please provide the date of birth.
        </div>
      </div>
      <div class="col-md-6">
        <label for="validationCustomUsername" class="form-label">Username</label>
        <div class="input-group has-validation">
          <span class="input-group-text" id="inputGroupPrepend">@</span>
          <input type="text" class="form-control" name="userName" id="validationCustomUsername" value="% details[0].UserName %" required>
          <div class="invalid-feedback">
            Please choose a username.
          </div>
        </div>
      </div>
      <div class="col-md-6">
        <label for="validationCustom04" class="form-label">Password</label>
        <input type="password" class="form-control" name="password" id="validationCustom04" value="% details[0].Password %" required>
        <div class="invalid-feedback">
          Please provide a password.
        </div>
      </div>
    </form>
  </div>
</section>
```

Figure 33: Admin Account Code

4.10 User Dashboard

4.10.1 Screenshot



The screenshot shows the 'ONLINE ELECTION SYSTEM' dashboard. The top navigation bar includes links for DASHBOARD, ELECTION DETAILS, CANDIDATES DETAILS, CAST VOTE, ELECTION RESULT, ACCOUNT, and LOGOUT. The main content area is titled 'My Voting History' and contains a table with the following data:

#	Voted To	Election Name	Election Type	Election Province	Election City	Election Region	Vote Date
1	Ali Haider	NA-95	ad	KPK	Sheikhupura	UET	11/12/2023
2	Ali Haider	NA-95	ad	KPK	Sheikhupura	UET	11/12/2023

Figure 34: User Dashboard

4.10.2 Code Snippet

```
<link rel="stylesheet" href="/css/adminElection.css">
<div class="tableElection tableElectionCandidate">
  <h3>My Voting History</h3>
  <table class="table table-striped">
    <thead>
      <tr>
        <th scope="col">#</th>
        <th scope="col">Voted To</th>
        <th scope="col">Election Name</th>
        <th scope="col">Election Type</th>
        <th scope="col">Election Province</th>
        <th scope="col">Election City</th>
        <th scope="col">Election Region</th>
        <th scope="col">Vote Date</th>
      </tr>
    </thead>
    <tbody>
      <% votes.forEach((vote, index) => { %>
        <tr id="voteRow_<%= vote._id %">
          <th scope="row"><%= index + 1 %></th>
          <td><%= vote.CandidateId.FirstName + ' ' + vote.CandidateId.LastName %></td>
          <td><%= vote.ElectionId.Name %></td>
          <td><%= vote.ElectionId.ElectionType %></td>
          <td><%= vote.ElectionId.Province %></td>
          <td><%= vote.ElectionId.City %></td>
          <td><%= vote.ElectionId.Region %></td>
          <td><%= vote.createdAt.toLocaleDateString() %></td>
        </tr>
      <% }) %>
    </tbody>
  </table>
```

Figure 35: User Dashboard Code

4.11 User Cast Vote

4.11.1 Screenshot

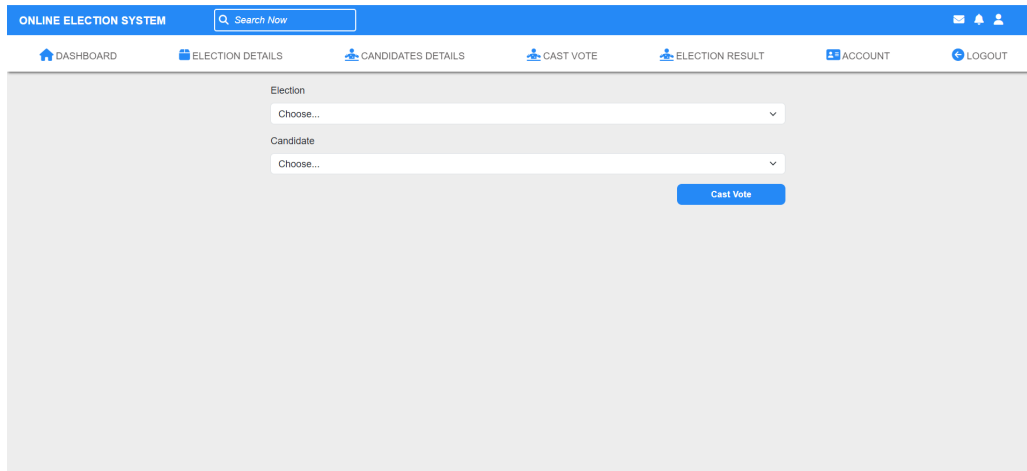


Figure 36: User Cast Vote

4.11.2 Code Snippet

```
<link rel="stylesheet" href="/css/adminElection.css">

<section class="box-data box-data2">
  <form class="row g-3 needs-validation" action="" method="post" novalidate>
    <div class="col-md-12">
      <label for="validationCustom01" class="form-label">Election</label>
      <select class="form-select select-text" name="election" id="validationCustom01" required>
        <option selected disabled value="">Choose...</option>
        <% elections.forEach(election => { %>
          <option value="<%= election.id %>"><%= election.formattedValue %></option>
        <% }); %>
      </select>
      <div class="invalid-feedback">
        Please select a valid Election.
      </div>
    </div>

    <div class="col-md-12">
      <label for="validationCustom02" class="form-label">Candidate</label>
      <select class="form-select select-text" name="candidate" id="validationCustom02" required>
        <option selected disabled value="">Choose...</option>
        <% candidates.forEach(candidate => { %>
          <option value="<%= candidate.id %>"><%= candidate.formattedName %></option>
        <% }); %>
      </select>
      <div class="invalid-feedback">
        Please select a valid Candidate.
      </div>
    </div>

    <div class="col-12">
      <div class="text-end">
        <button onClick="castVote()" class="btn btn-primary new-btn" id="btnSave" type="submit">Cast Vote</button>
      </div>
    </div>
  </form>
</section>

<script src="/js/userCastVote.js"></script>
```

Figure 37: User Cast Vote Code

5 Conclusion

Throughout our journey in crafting the Online Election System, we've achieved notable milestones. We've effectively integrated user registration capabilities, empowering individuals to register accounts and interact effortlessly with the platform. Our profile management system offers a personalized touch, enabling users to display and modify their details. Facilitating the creation and administration of election campaigns marks a pivotal addition to our system, enriching its capabilities. The introduction of a user-friendly dashboard and a dedicated area for overseeing electoral campaigns contributes to a resilient and user-oriented system.