

A
PROJECT REPORT
ON

Online Voting System

By

PATEL KUNJ B. (CE-102) (19CEUTG053)
SHAH DIYA A.(CE-119) (19CEUOS102)

B.Tech CE Semester-IV
Subject: Software Project-I

Guided by:

Prof.Pinkal C. Chauhan
Prof.Brijesh B. Bhatt
Prof.Jigar M. Pandya



Faculty of Technology
Department of Computer Engineering
Dharmsinh Desai University



**Faculty of Technology
Department of Computer Engineering
Dharmsinh Desai University**

CERTIFICATE

This is to certify that the practical / term work carried out in the subject of **Software Project-I** and recorded in this journal is the bonafide work of

SHAH DIYA A.(CE-119) (19CEUOS102)

PATEL KUNJ B. (CE-102) (19CEUTG053)

of B.Tech semester **IV** in the branch of **Computer Engineering**
during the academic year **2020-2021**.

Contents:

1. Abstract	4
2. Introduction	5
3. Software Requirement Specifications	7
4. Design	
I) Use Case diagram	11
II) Sequence diagram	12
III) Activity diagram	14
IV) Class diagram	16
V) Data Flow diagram	17
5. Implementation Detail	
I) Modules	20
II) Major Functionality	22
6. Work Flow / Layouts	23
7. Conclusion	28
8. Limitation and Future extension	29
9. Bibliography	30

1. Abstract

The project is mainly aimed at providing a secured and user friendly voting system. The problem of voting is still critical in terms of safety and security. The voting system is managed in a simpler way as all the voters must login by using College ID number and click on his/her favourable candidates to cast the vote by using it provides enough security as one College ID number can be used to cast vote at once only which reduces dummy votes.

On the basis of result , candidates are elected and give them appropriate positions . Using this online system voters can save their time and click on his/her favourable candidates to cast the vote and choose his/her favourable candidate for appropriate position.

2. Introduction

2.1 Brief Introduction

Online Voting Project is a system where the voter is recognized by his/her College ID number. Since the registration number of each student in the college is different, the voter can be easily authenticated. The system allow the voter to vote through his/her registration number. Voter can vote the candidate only once, the system will not allow the candidate to vote for the second time. All voters must have a college ID in order to get the registration number by the comity.

The system will allow admin to add the candidate name and position who are nominated for the election. Admin only has the right to add candidate name and position who are nominated. Admin will register the voters name by verifying voter. Admin will authenticate the user by verifying the user's identity proof and then admin will register the voter. Once the user has got the user id and password from the admin the user can login and vote for the candidate who are nominated. The system will allow the user to vote for only one candidate for one position. The system will allow the user to vote for one time for a particular election. Admin can add any number of candidates when the new election will be announced. Admin can view the election result . Even user can also view the election result any time.

2.2 Tools/Technologies Used

Technologies:

- Django
- Python
- MySQL
- HTML
- CSS

Tools

- GitHub
- Visual Studio Code

Platform

- Local Host

3. Software Requirement Specifications

Admin

R.1.1 Maintain isolation

Description: Admin checks vote integrity here.

R 1.1.1 Vote count

Description-To check the total number of votes casted during the process.

Input: A link will be available named “Result”, and the admin have to click on it.

Output: A count of total number of votes casted till now will be displayed.

R 1.1.2 Check vote integrity

Description- To check number of votes casted by each voter in order to eliminate any discrepancy during the voting process and will show YES if the voter has casted a vote else NO.

Input: A table link will be available named “show voting details”, and the admin have to access it .

Output: A table will be displayed of the voters having 3 columns named,
i)registration number
ii)YES
iii)NO

R 1.2 Mange polling

Description: Admin controls the voting process here

R 1.1.3 Candidate votes

Description- To track number of votes casted for a particular candidate.
Input: A table named “Candidates” will be available on the homepage of the admin, by clicking on it , votes casted for a every candidate will be shown.

Output: A table will be displayed showing names of the candidates and votes gain by each of them.

R 1.2 Manage polling

Description: Admin controls the voting process here

R 1.2.1 Update records in database

Description: If volunteer forgot to store the details of any voter in database then only admin will update the data.

Input: Display form which takes all details of voter .

Output : Display message “Saved successfully.” and store in database at particular section.

R 1.2.2 Delete particular record in database

Description: If any record stored twice in database then admin will delete duplicate data from the database.

Input: Display textbox which takes registration number and voter name also.

Output: Display the message “Delete record successfully”

Admin

R 2.1 Register voters

Description: Admin registers the voters

R 2.1.1 Entering the full name

Description: The voter is required to enter the full name.

Input: Display Textfield for name.

Output: Name taken and stored at temporary storage.

R 2.1.2 Enter the collegeid number

Description: The voter is required to enter the collegeid number.

Input: Display textbox for collegeid number which takes unique id as an input.

Output: collegeid number taken and it generates automatic registration number for further process.

R 2.1.3 Enter current year of studying

Description: The voter is required to enter the year of studying.

Input: Display selection list which takes in range of specific curriculum duration.

Output: From year , it allows the voter to participate in the voting process and display the message “You are eligible for voting”.

R 2.1 Register Candidates

Description: Admin registers the candidates

Input: Display form for adding the details of candidate and click to submit button.

Output: It will display message “Candidate added successfully”.

R 2.2 Voter Database

Description: Storing data in database

R 2.2.1 Voter info

Description: This will show the details entered by the voters during registration to keep a record.

Input: A table link named "Voters" will be available on the homepage of the website, committee or admin have to click on that link.

Output: A table will be displayed having information of voters registered for the process.

R 2.2.2 Login Verification

Description: This will cross check the login details provided by the user to the details recorded by the committee.

Input: Voter have to login using their registration number and email id

Output: If the registration details provided by voter is valid then "login successful" message will be declared.

R 2.2.3 Voting process

Description: The voters will cast their vote to the desired candidates.

Input: The voter have to fill the radio button beside the candidate.

Output: The radio button will be filled.

R 2.2.5 Storing actions in database

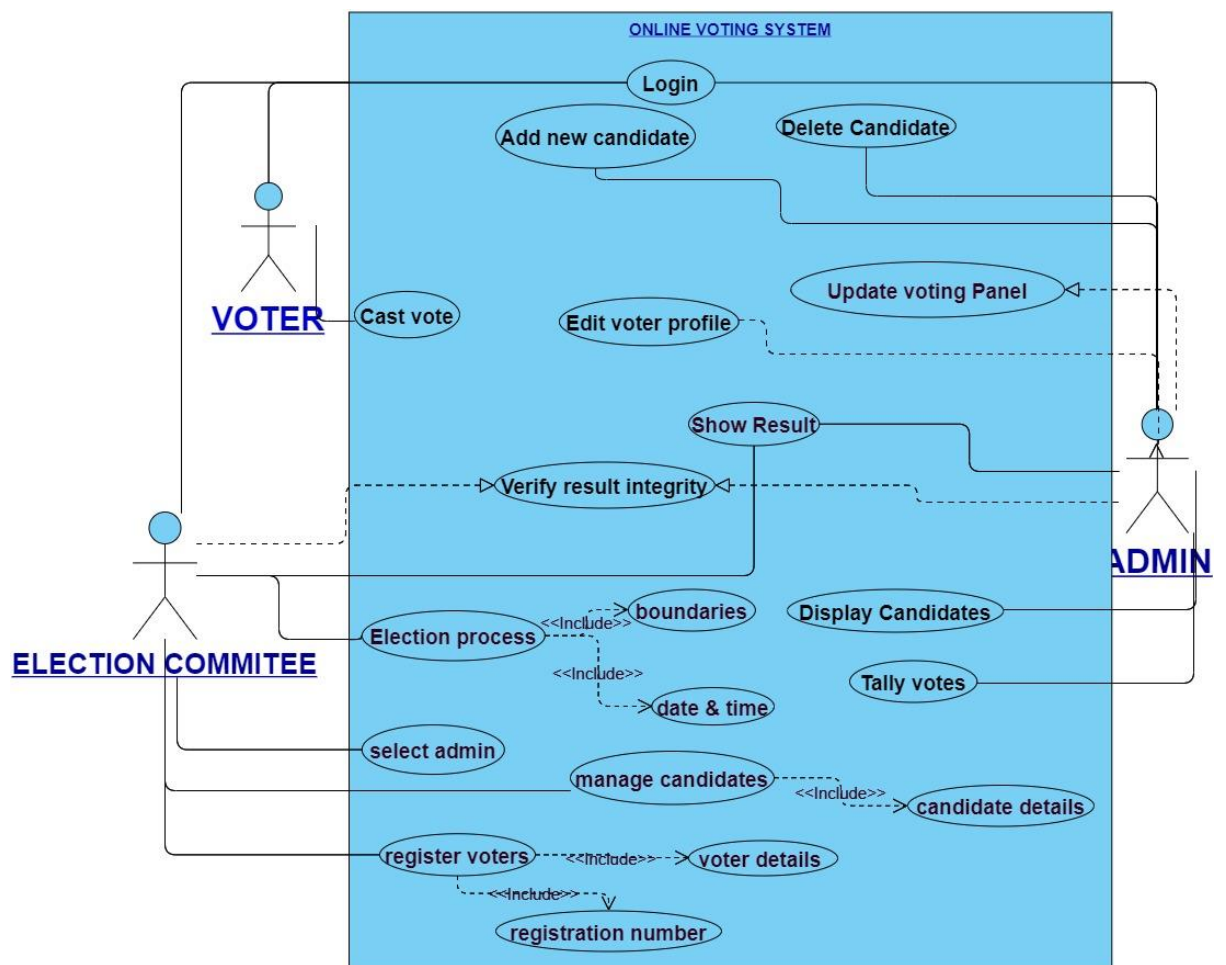
Description: The vote casted by the candidate will be saved in the database and can be viewed by the admin or committee at anytime.

Input: The voter will caste his/her vote to the desired candidate.

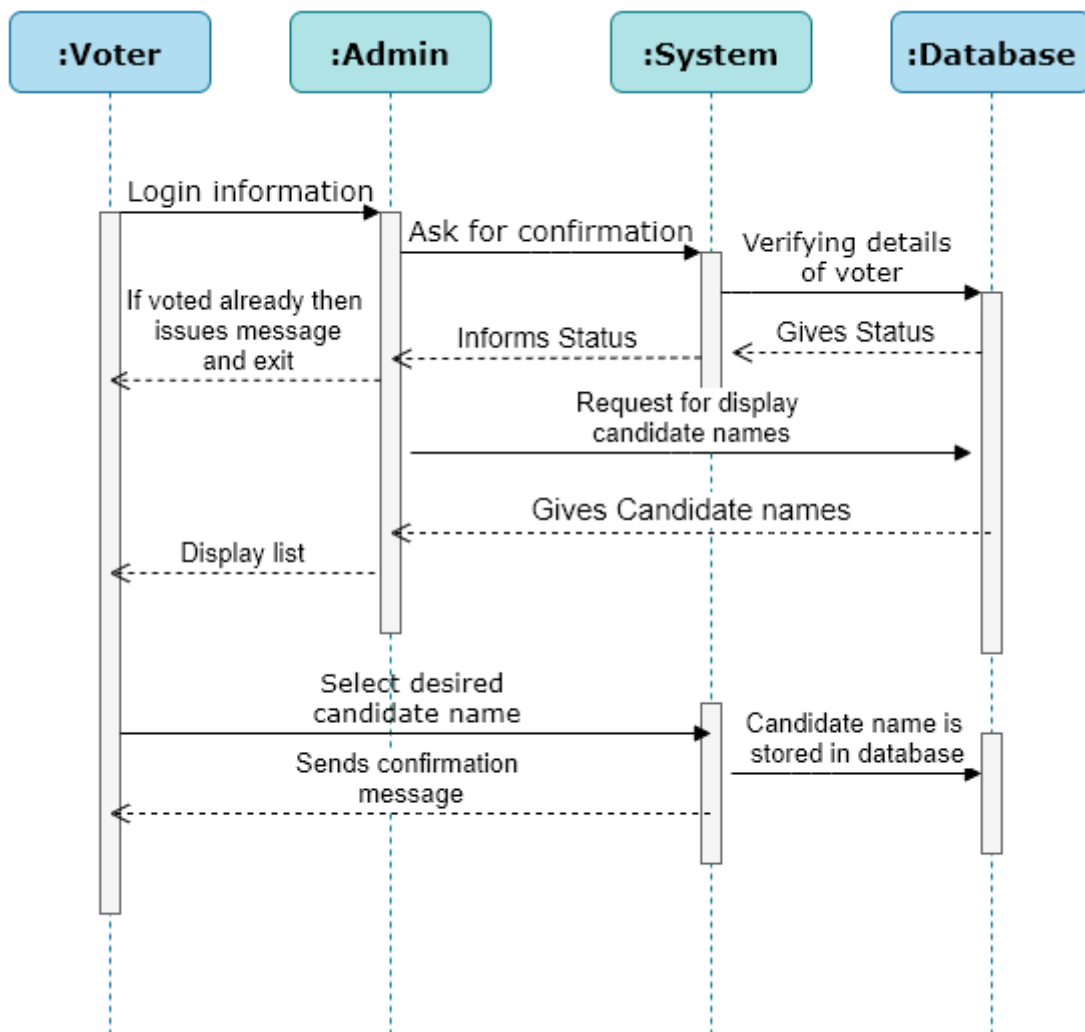
Output: The vote count will be stored in the database of candidate and vote casted will be stored in the database of the voter.

4. Design

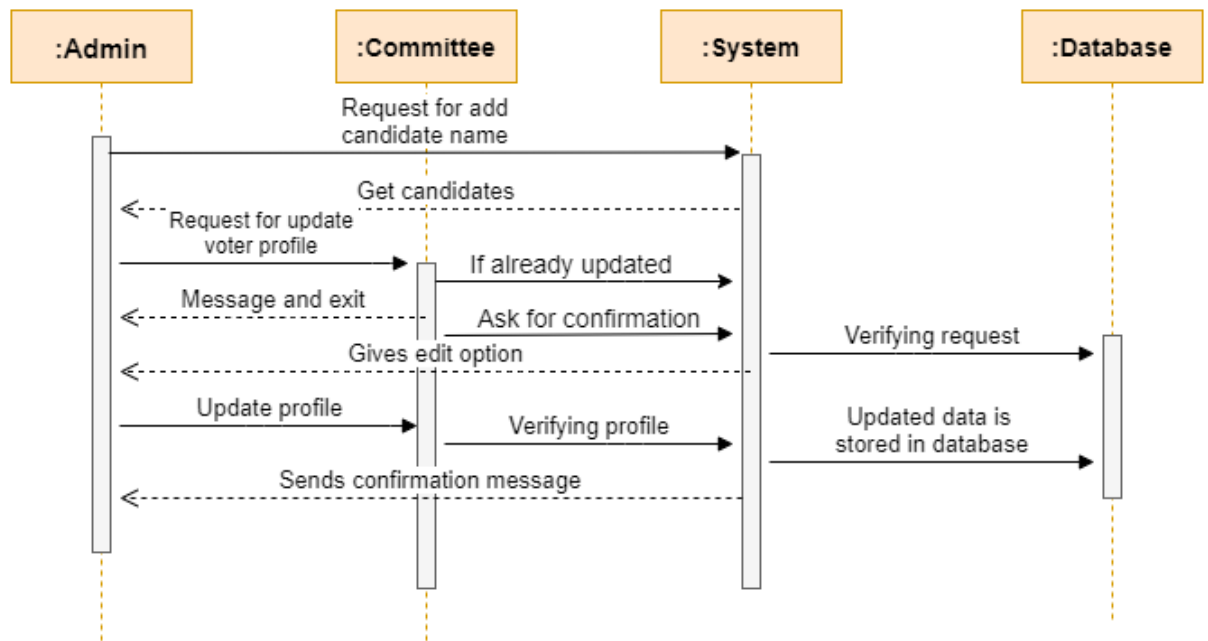
4.1 Use Case Diagram



4.2 Sequence Diagram

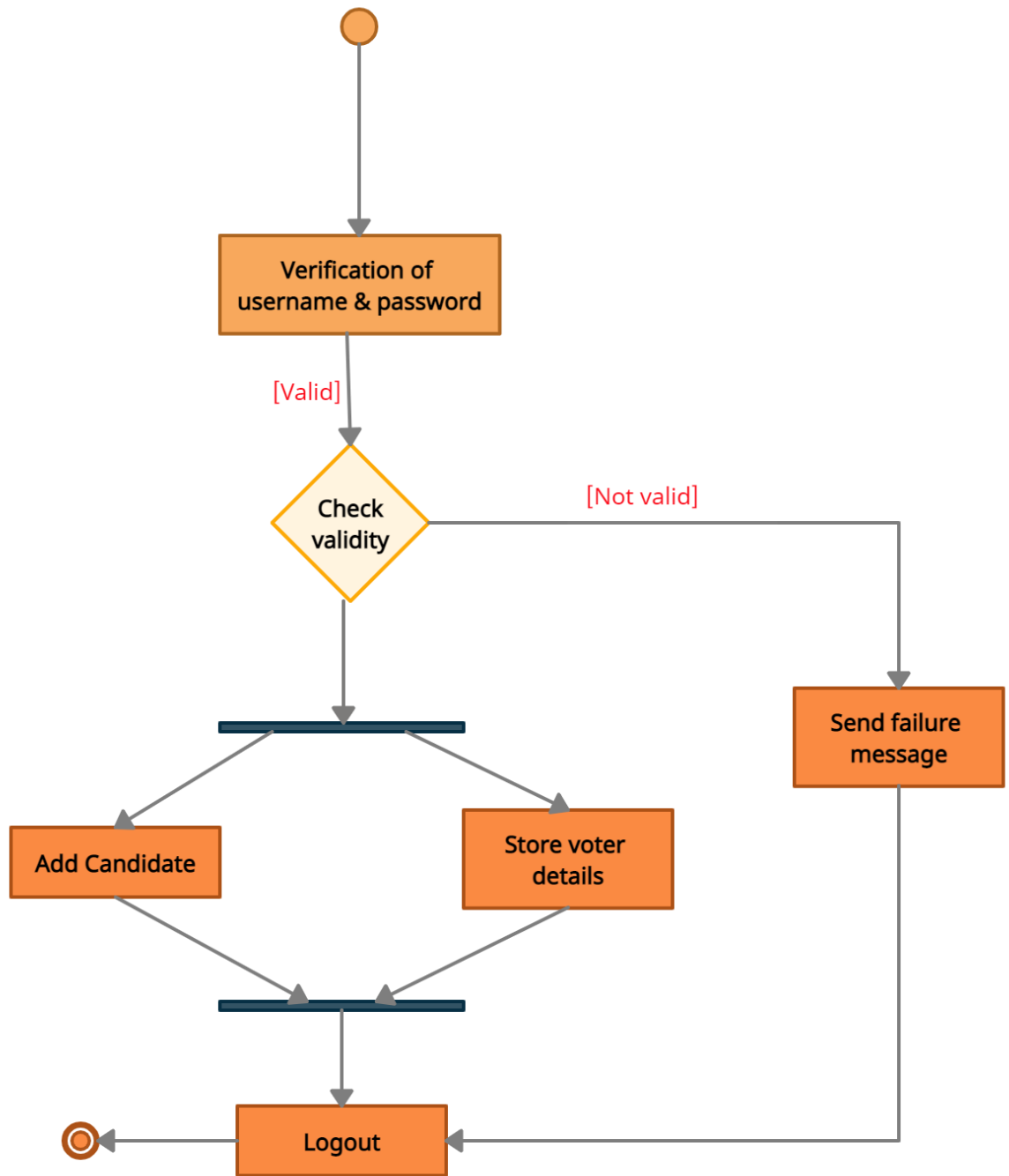


Sequence diagram for voting process

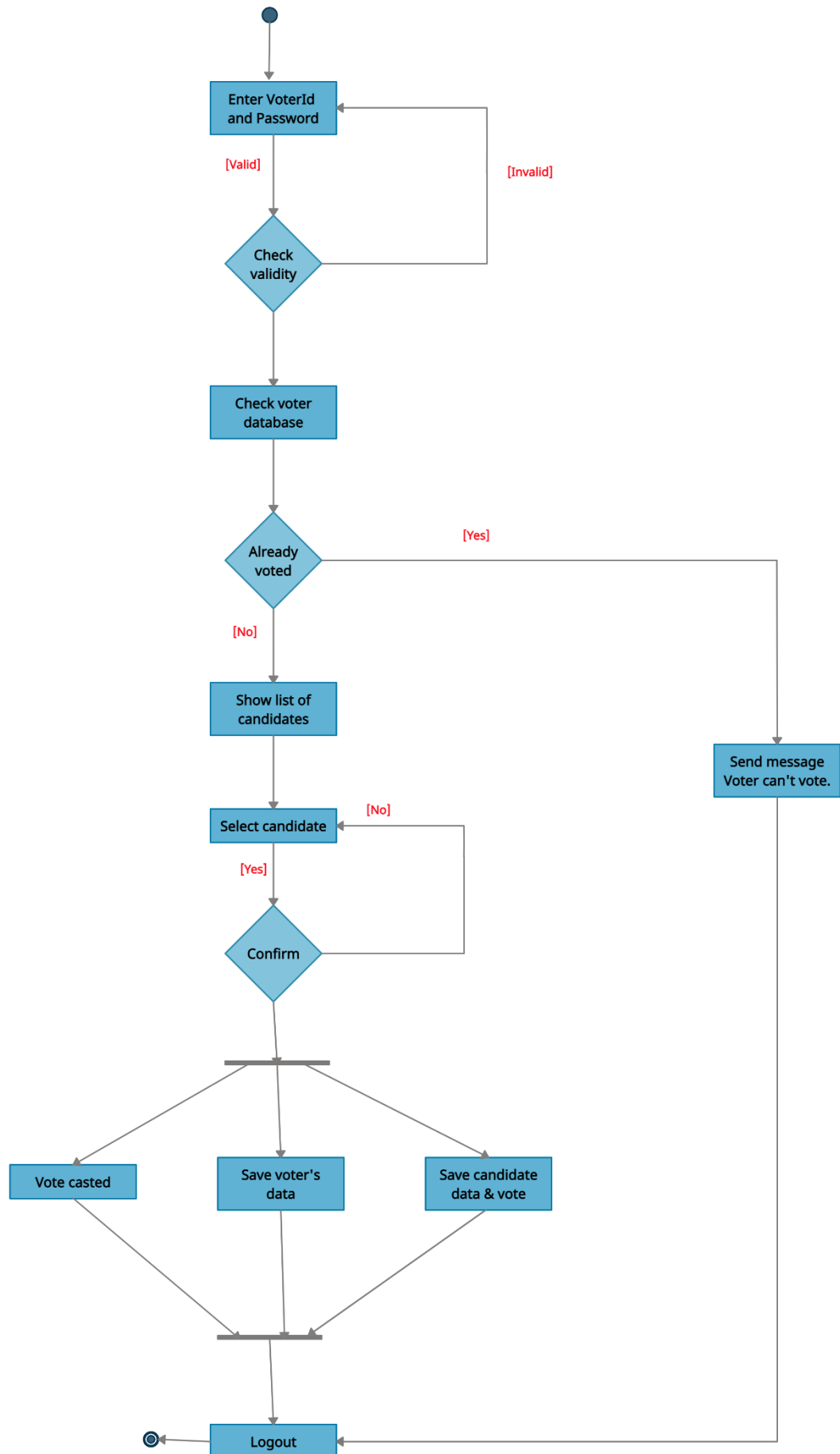


Sequence diagram for add candidate

4.3 Activity Diagram

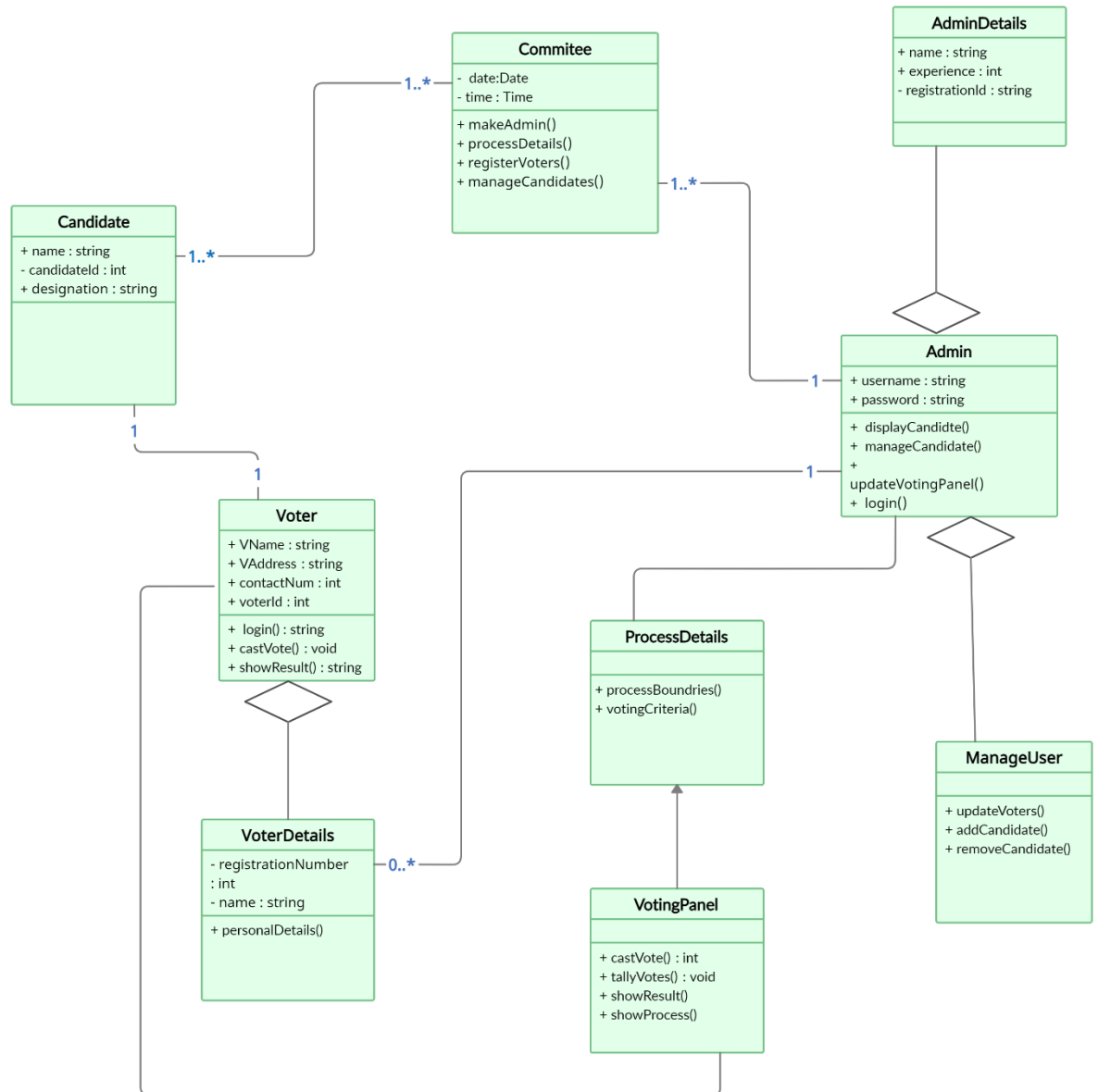


Activity diagram for add candidate and add voter

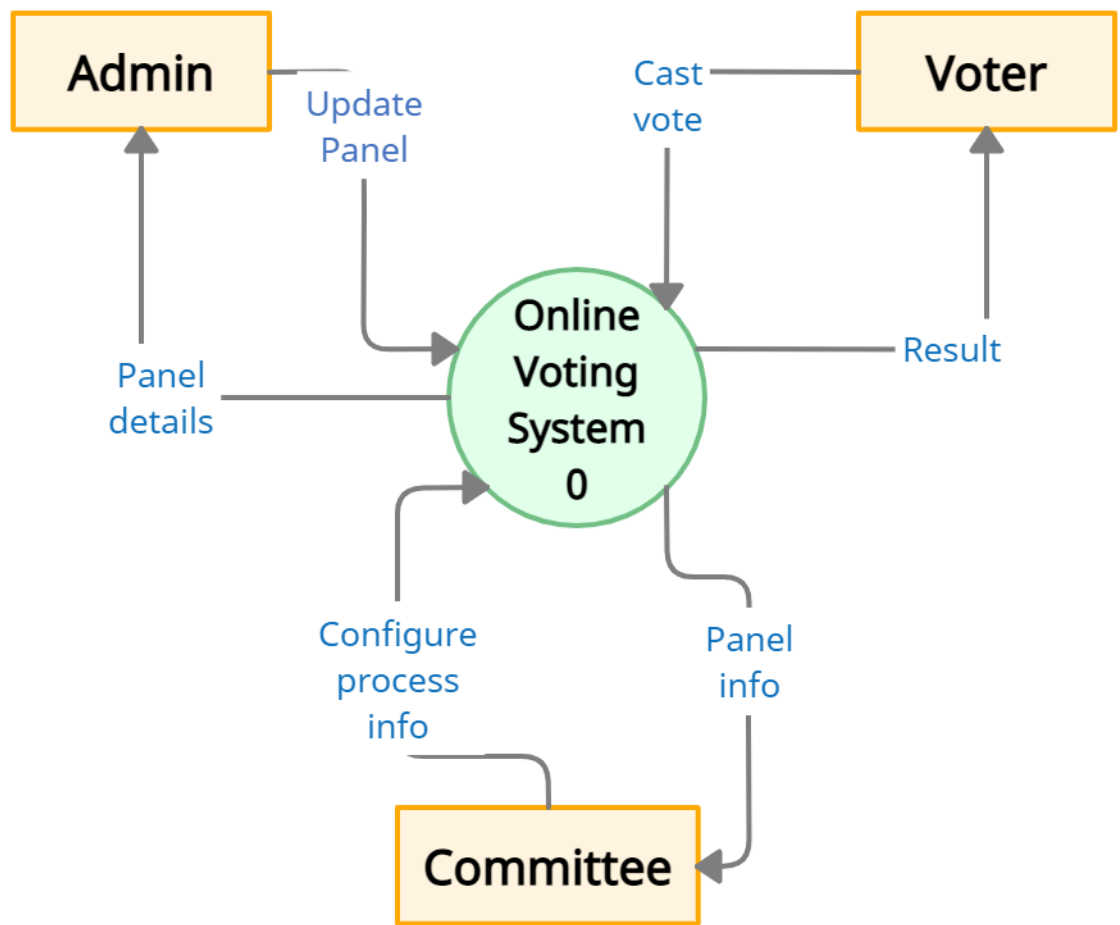


Activity diagram for Voting Process

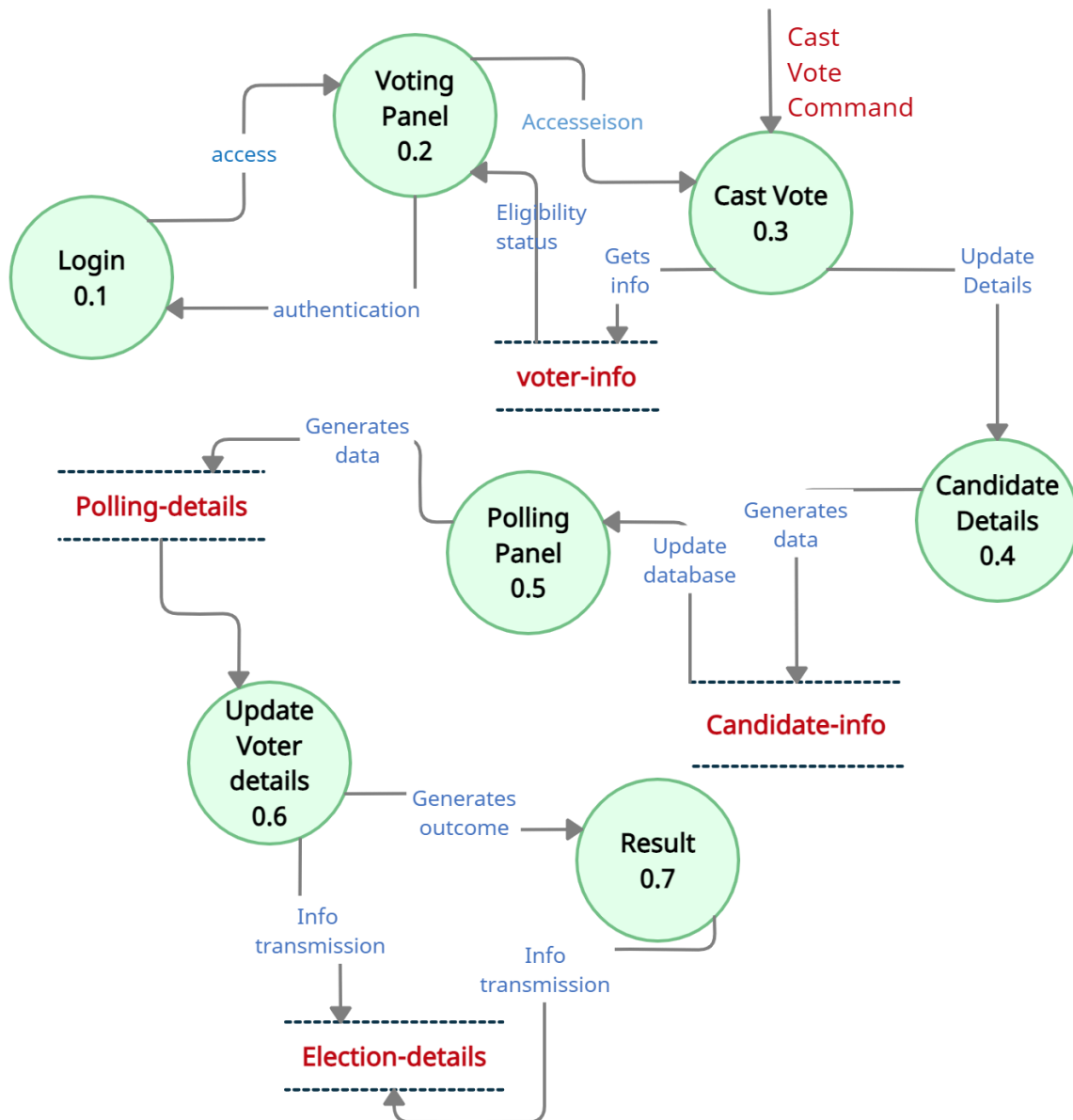
4.4 Class Diagram



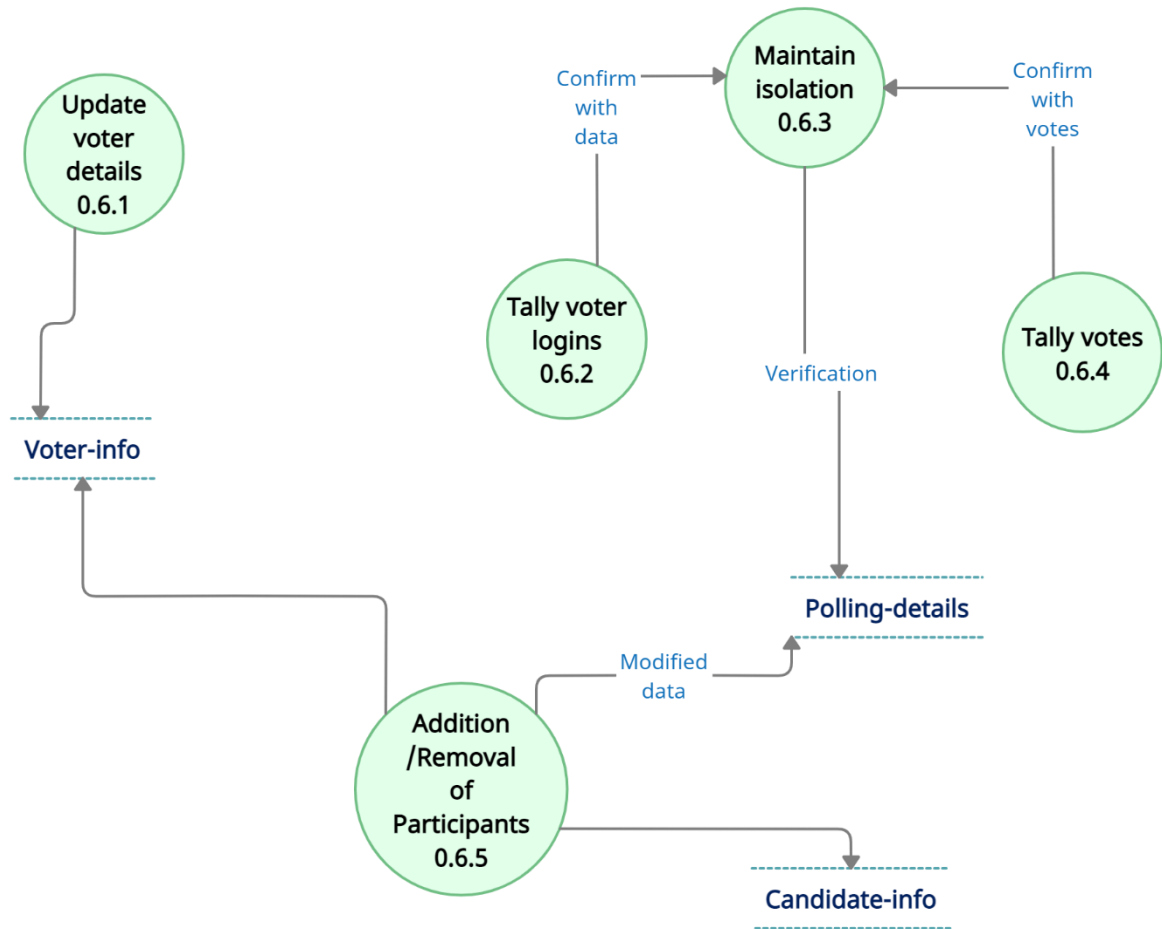
4.5 Data Flow Diagram



(A) Context diagram



(B) Level-1 diagram



5. Implementation Details

The system consists of 4 basic modules namely

1. Voting Module
2. Tallying Module
3. Voter Module
4. Admin Module

Each module consists of several methods to implement the required functionality. Implementation is done using Django. Database used in these modules is MySQL.

Voting Module

- This module does the job of taking votes from the voters.
- During the voting period the login information and vote casted by the voter will be stored in a record.
- Once the vote is casted the user will be automatically returned to the homepage and a message of vote confirmation will be displayed on the screen.
- By this module proper completion of the process on given time will be achieved.

Tallying Module

- This module does the job of vote count for the entire election process.
- It checks the votes registered with the total number of vote casted.
- If there is any discrepancy in the above procedure then it checks the number of votes casted by each voter.
- Then, it compares number of votes of each candidate and generates results of the election process.

Voter Module

- This module is simply a database system that contains the information of all registered voters.

- Prospective voters would first supply their information on the database during the voter registration window from any designated voter registration points.
- Before any prospective voter can cast their vote, the system would have to first confirm that they have been registered to participate in the election.
- This is the voter accreditation phase of voting. The voter information database works with the voter authentication system which connects and relies on the voter information database for the completion of voter authentication.

Admin Module

- This module handles various versions of operation. It allows admin to add candidate , add voter and add position.
- When the voter is entered for voting only those candidates and positions are displayed on the page which are added by admin .
- When the voter is login for casting the vote , first system check that this voter is added by admin or not. If added then only he/she give vote.

5.2 Function prototypes

```
def voterDetails(request):
    if request.method == 'POST':
        voterName=request.POST.get('voterName','')
        collegeId=request.POST.get('id','')
        voterDepartment=request.POST.get('dept','')
        academic_year = request.POST.get('year','')
        email =request.POST.get('email','')
        voting_status=request.POST.get('status','')
        voter = Voter(voterName=voterName,collegeId=collegeId,voterDepartment=voterDepartment,academic_year =academic_year ,email =email ,v
        voter.save()
        username = request.POST.get('email','')
        password = request.POST.get('id','')
        user = User.objects.create_user(username = username , password = password)
        user.save()
        messages.success(request, 'Voter added successfully!!')
        return render(request,'addVoter.html')
```

Add Voter

```

135
136 def addCandidate(request):
137     pos = Position.objects.all()
138     return render(request, 'addCandidate.html', {'positions': pos})
139
140 def candidateDetails(request):
141     if request.method == 'POST':
142         candidateName=request.POST.get('candidateName','')
143         position_id=request.POST.get('position','')
144         position = Position.objects.get(id = position_id)
145         email=request.POST.get('email','')
146         Candidate.objects.create(candidateName = candidateName , position = position , email=email)
147         messages.success(request, 'Candidate added successfully!!')
148         return addCandidate(request)

```

Add Candidate

```

def addPosition(request):
    return render(request, "addPosition.html")

def positionDetails(request):
    if request.method == 'POST':
        title = request.POST.get('name','')
        Position.objects.create(title=title)
        messages.success(request, 'Position added successfully!!')
        return render(request, 'addPosition.html')

```

Add Position

```

def submit(request):
    if request.method == "POST":
        can = request.POST.get('canname')
        c = Candidate.objects.get(candidateName = can)
        pos = c.position_id
        us = request.user
        u = us.id
        c = Candidate.objects.get(candidateName = can)
        candidate_id = c.id
        voter = CountVote.objects.filter(voterId = u , positionId = pos).update(candidateId = candidate_id)
        return render(request, 'submit.html')

```

Store Vote in Database

```
def result(request):
    candidates = Candidate.objects.all()
    results = []

    for candidate in candidates:
        result = {}

        countvote = CountVote.objects.filter(positionId = candidate.position_id , candidateId = candidate.id).count()
        result['can'] = candidate
        result['count'] = countvote
        results.append(result)

    return render(request, 'result.html', { 'results' : results })
```

Showing Result

6. Work Flow

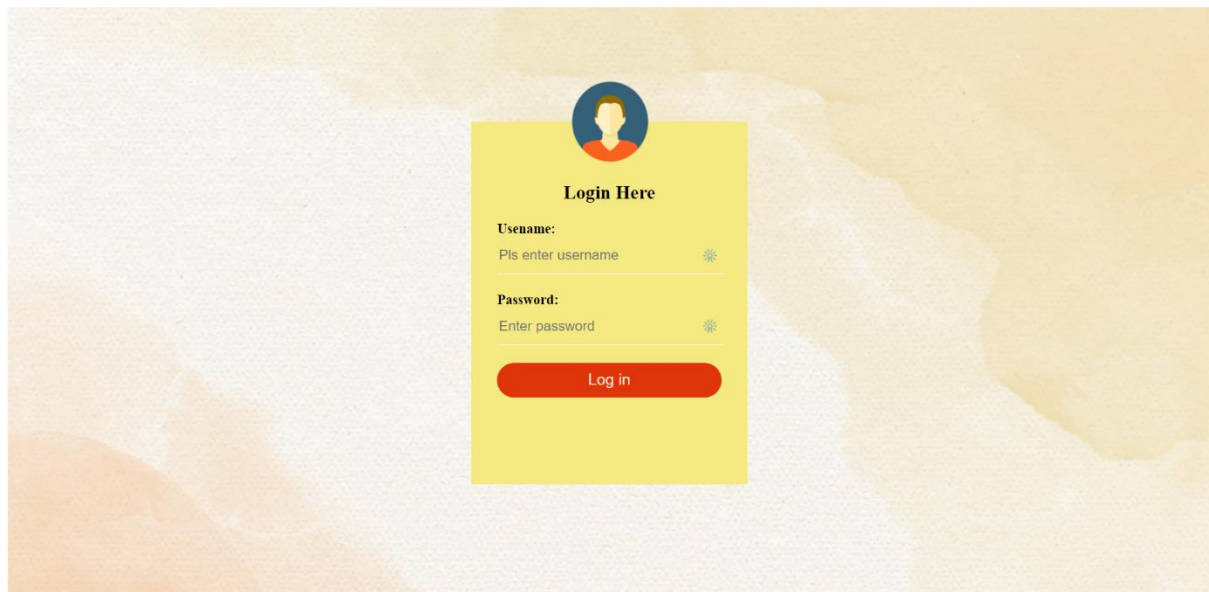
Homepage of website



Admin:

Admin can perform the following activities

1.Login



A login form is centered on a yellow background. At the top is a circular icon of a person with a blue head and red body. Below it, the text "Login Here" is displayed. The form contains two input fields: "Username:" with the placeholder "Pls enter username" and "Password:" with the placeholder "Enter password". Both fields have a small eye icon to the right. At the bottom is a red button labeled "Log in".

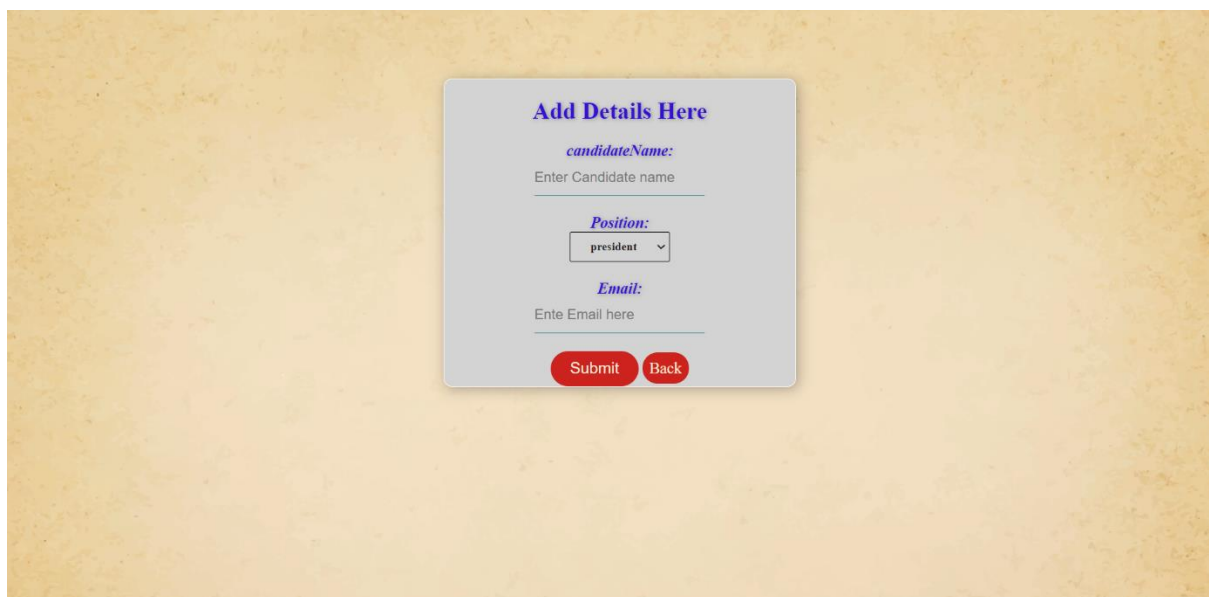
Login Here

Username:
Pls enter username

Password:
Enter password

Log in

2.Add Candidates



An "Add Details" form is centered on a light gray background. It has a title "Add Details Here" in blue. The form includes three input fields: "candidateName:" with the placeholder "Enter Candidate name", "Position:" with a dropdown menu showing "president", and "Email:" with the placeholder "Ente Email here". At the bottom are two red buttons labeled "Submit" and "Back".

Add Details Here

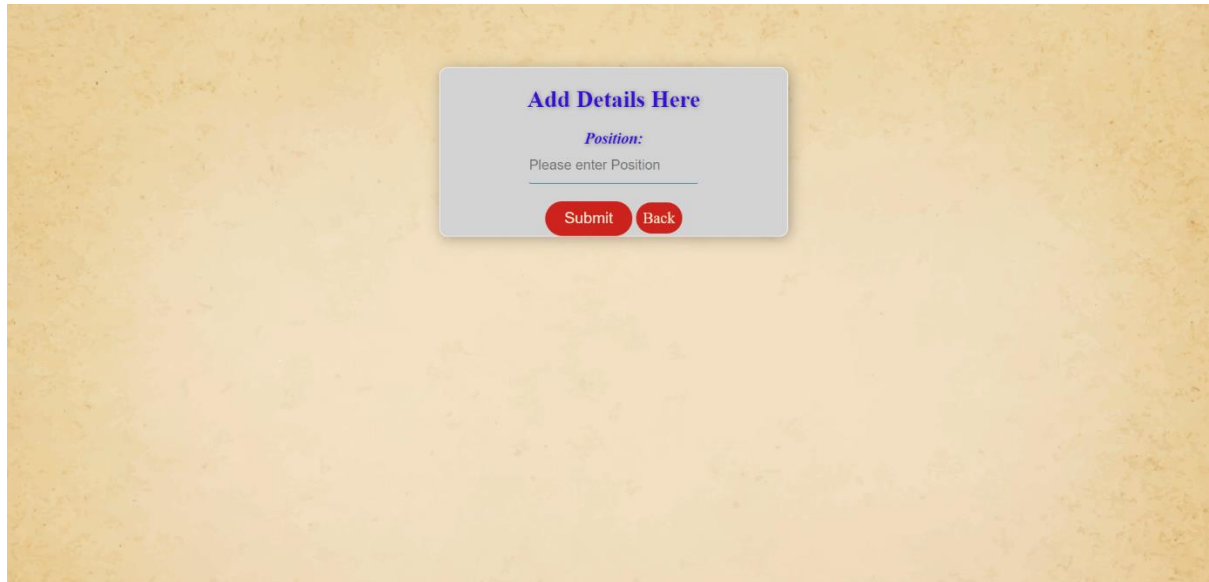
candidateName:
Enter Candidate name

Position:
president

Email:
Ente Email here

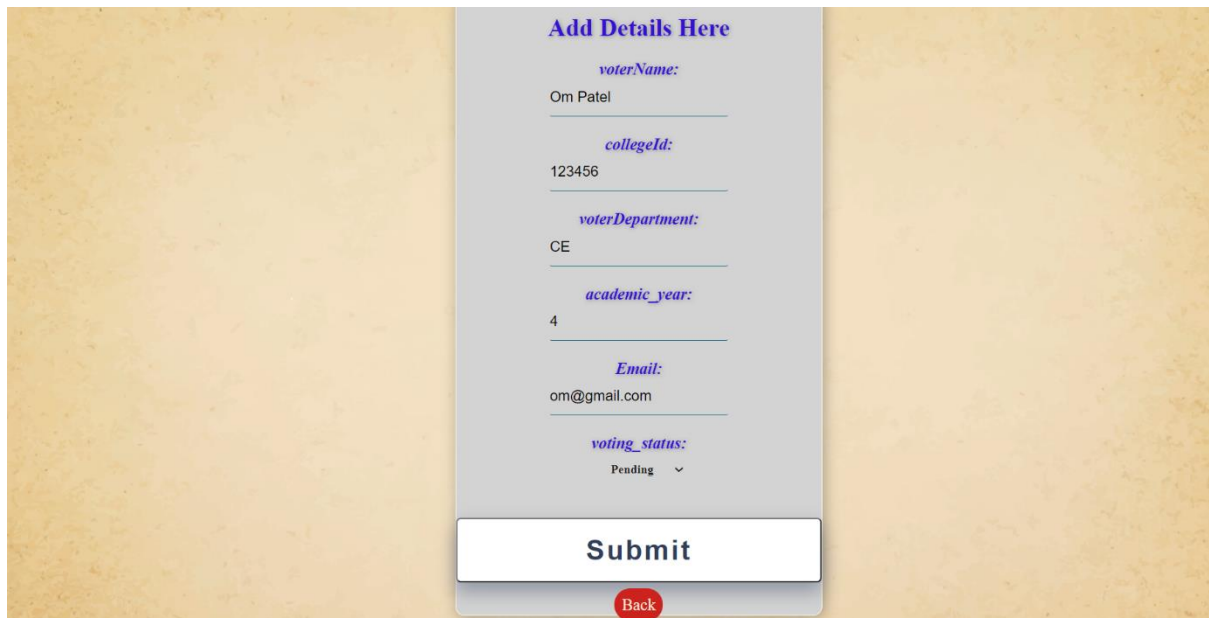
Submit Back

3.Add Position



A screenshot of a web form titled "Add Details Here" in blue. The form is set against a light beige, textured background. It contains a label "Position:" in blue, followed by the text "Please enter Position" in a small grey font. Below this is a text input field. At the bottom of the form are two red buttons: "Submit" and "Back".

4.Add Voters

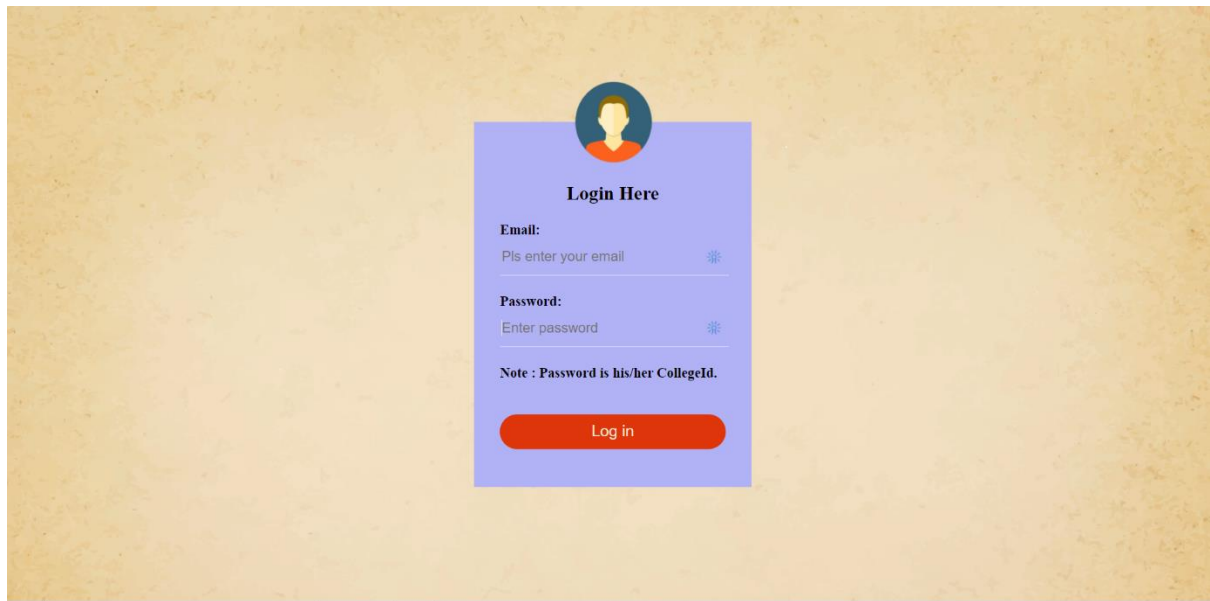


A screenshot of a web form titled "Add Details Here" in blue. The form is set against a light beige, textured background. It contains several fields with blue labels: "voterName:" with the value "Om Patel", "collegeId:" with the value "123456", "voterDepartment:" with the value "CE", "academic_year:" with the value "4", "Email:" with the value "om@gmail.com", and "voting_status:" with a dropdown menu showing "Pending". At the bottom of the form is a large white button labeled "Submit" and a red button labeled "Back".

Voters:

Voters can do the following actions

1.Login



A login form is centered on a light brown, textured background. The form has a light blue header with a circular profile icon of a person with orange hair. Below the icon, the text "Login Here" is displayed. The form contains two input fields: "Email:" with the placeholder "Pls enter your email" and "Password:" with the placeholder "Enter password". Both fields have a small blue asterisk icon to their right. Below the password field, a note reads "Note : Password is his/her Collegeld." At the bottom of the form is a red "Log in" button.

Login Here

Email:
Pls enter your email *

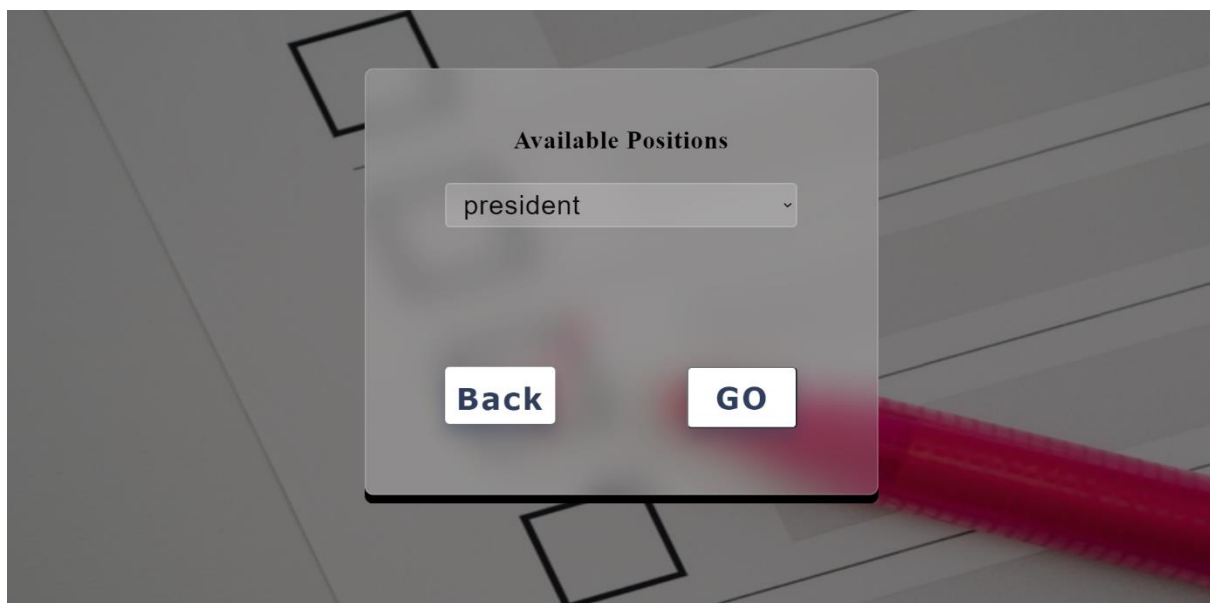
Password:
Enter password *

Note : Password is his/her Collegeld.

Log in

2.Cast Vote

i)Select position for which you want to vote for



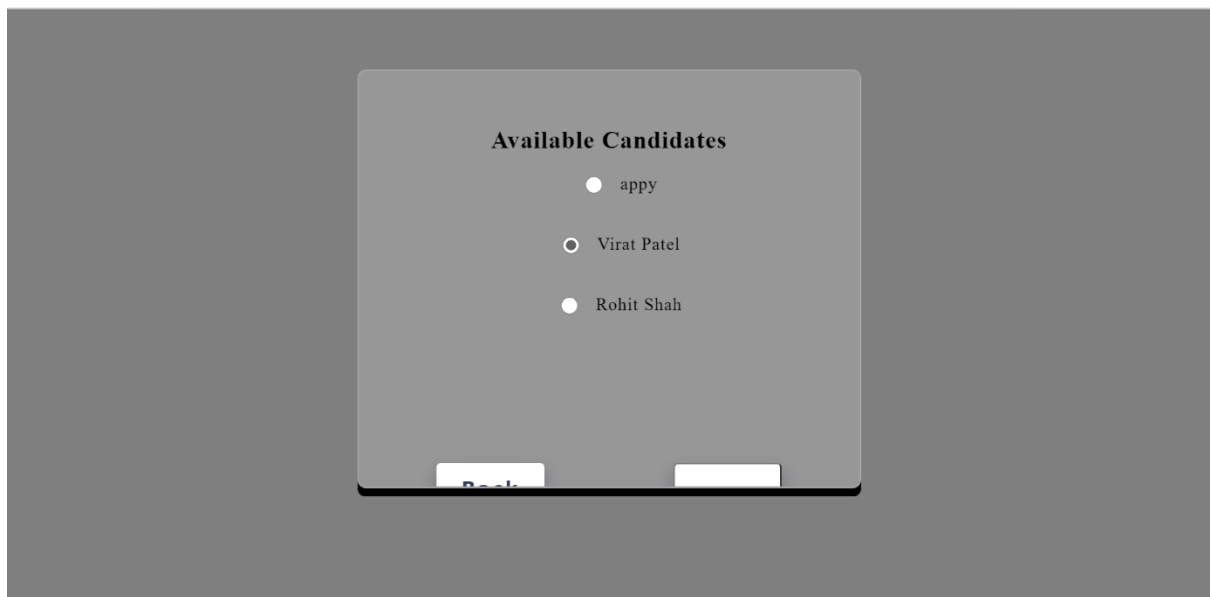
A semi-transparent dialog box titled "Available Positions" is shown over a background of a ballot paper with checkboxes and a red pen. The dialog box contains a dropdown menu with "president" selected. At the bottom are two buttons: "Back" and "GO".

Available Positions

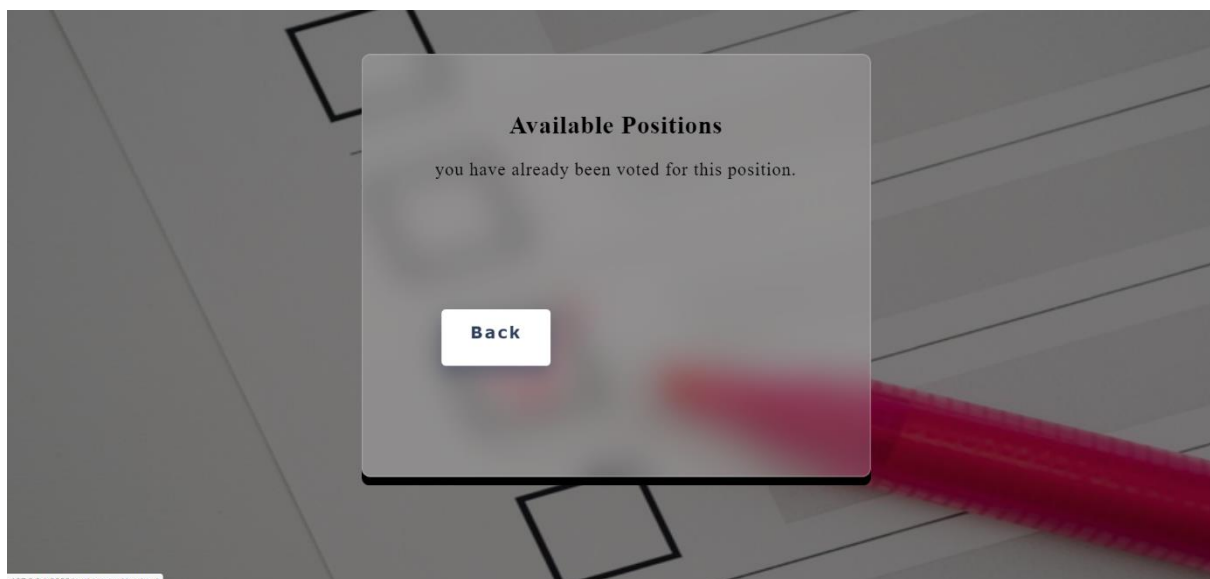
president ▼

Back **GO**

ii)Vote for your desired Candidate



iii) if you have already voted before



3. See results

Result	
Candidate Name	Total Votes
appy	2
Virat Patel	0
Rohit Shah	0

Back

7. Conclusion

The functionalities are implemented in system after understanding all the system modules according to the requirements.

Functionalities that are successfully implemented in the system are:

- Voter registration containing all the necessary validation on field
- Login
- Voter authentication
- Add Candidate , Voter & Position
- Store Vote into Database
- Send Review Result Requests
- Admin Authentication

Hence-forth in this project we have successfully implemented the Admin-side & Voter-side functionality, Admin will add the Voter , Candidate and Position & voter can vote the available candidates at the particular position. Admin and Voter can show result .

8. Limitations and Future Enhancements

We are able to implement the functionality model of the “Online Voting System”. We aim to make this product ready to be used in practical university use cases.

Currently this project works on small scenario election but we will try to extend it’s boundries. The project supports online in-browser syntax highlighting and editing of the code. We can add a feature of in-browser test running of the code. Further extensions involve integrating it with GitHub for suggesting submission to review.

9. Reference / Bibliography

Following links and websites were referred during the development of this project:

stackoverflow.com

docs.djangoproject.com