

## Project Development Phase and Model Performance Test

Date	04 November 2023
Team ID	NM2023TMID11282
Project Name	Electronic Voting System

### Model Performance Testing:


Project team shall fill the following information when working for blockchain.

S.No.	Parameter	Values	Screenshot
1.	Information gathering	Setup all the Prerequisite:	The Screenshots of information gathering are all given below.
2.	Extract the zip files	Open to vs code	The Screenshots of extract the zip file are all given below.

3.	Remix Idle Platform exploring	<p>Deploy the smart contract code</p> <p>Deploy and run the transaction.</p> <p>Byselecting the environment - inject the MetaMask.</p>	The Screenshots of extract the zip file are all given below.
4	Open file explorer	<p>Open the extracted file and click onthe folder.</p> <p>Open src, and search for utiles.</p> <p>Open cmd enter commands</p> <p>1.npm install</p> <p>2. npm bootstrap</p> <p>3. npm start</p>	The Screenshots of extract the zip file are all given below.
5	LOCALHOST IPADDRESS	Copy the address and open it to chrome so you can see the front end of your project.	The Screenshots of extract the zip file are all given below.

# 1. INFORMATION GATHERING

## Screenshot 1



### Brainstorm & idea prioritization

Use this template in your own brainstorming sessions so your team can unleash their imagination and start shaping concepts even if you're not sitting in the same room.

⌚ 10 minutes to prepare  
🕒 1 hour to collaborate  
👥 2-8 people recommended

**Before you collaborate**

A little bit of preparation goes a long way with this session. Here's what you need to do to get going.

⌚ 10 minutes

---

- Team gathering**  
Define who should participate in the session and send an invite. Share relevant information or pre-work ahead.
- Set the goal**  
Think about the problem you'll be focusing on solving in the brainstorming session.
- Learn how to use the facilitation tools**  
Use the Facilitation Superpowers to run a happy and productive session.


[Open article](#) →

**1**

### Define your problem statement

**PROBLEM**

How can we improve the efficiency and security of electronic voting systems to ensure a reliable and accessible voting process while addressing concerns related to accuracy, transparency, and cybersecurity?



#### Key rules of brainstorming

To run an smooth and productive session

- Stay in topic.
- Defer judgment.
- Go for volume.
- Encourage wild ideas.
- Listen to others.
- If possible, be visual.

## Screenshot 2

**2**

### Brainstorm

Write down any ideas that come to mind that address your problem statement.

⌚ 10 minutes

---

**Person 1**

- Use existing technology to create a secure, accessible, and user-friendly voting system.
- Use existing technology to create a secure, accessible, and user-friendly voting system.
- Use existing technology to create a secure, accessible, and user-friendly voting system.

**Person 2**

- Combine electronic voting with physical voting to create a hybrid system.
- Combine electronic voting with physical voting to create a hybrid system.
- Combine electronic voting with physical voting to create a hybrid system.

**Person 3**

- Use existing technology to create a secure, accessible, and user-friendly voting system.
- Use existing technology to create a secure, accessible, and user-friendly voting system.
- Use existing technology to create a secure, accessible, and user-friendly voting system.

**Person 4**

- Use existing technology to create a secure, accessible, and user-friendly voting system.
- Use existing technology to create a secure, accessible, and user-friendly voting system.
- Use existing technology to create a secure, accessible, and user-friendly voting system.

**3**

### Group ideas

Take turns sharing your ideas while clustering similar or related notes as you go. Once all sticky notes have been grouped, give each cluster a sentence-like label. If a cluster is bigger than six sticky notes, try and see if you can break it up into smaller sub-groups.

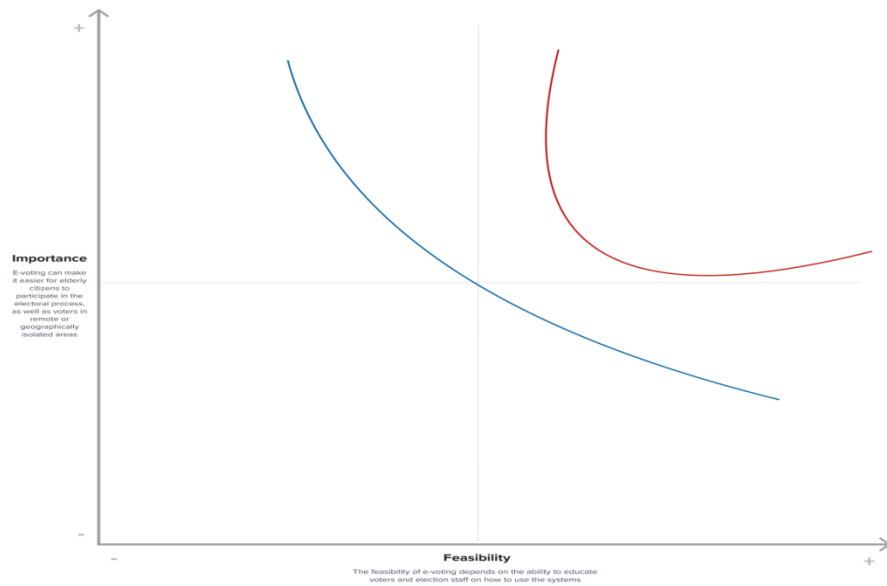
⌚ 20 minutes

---

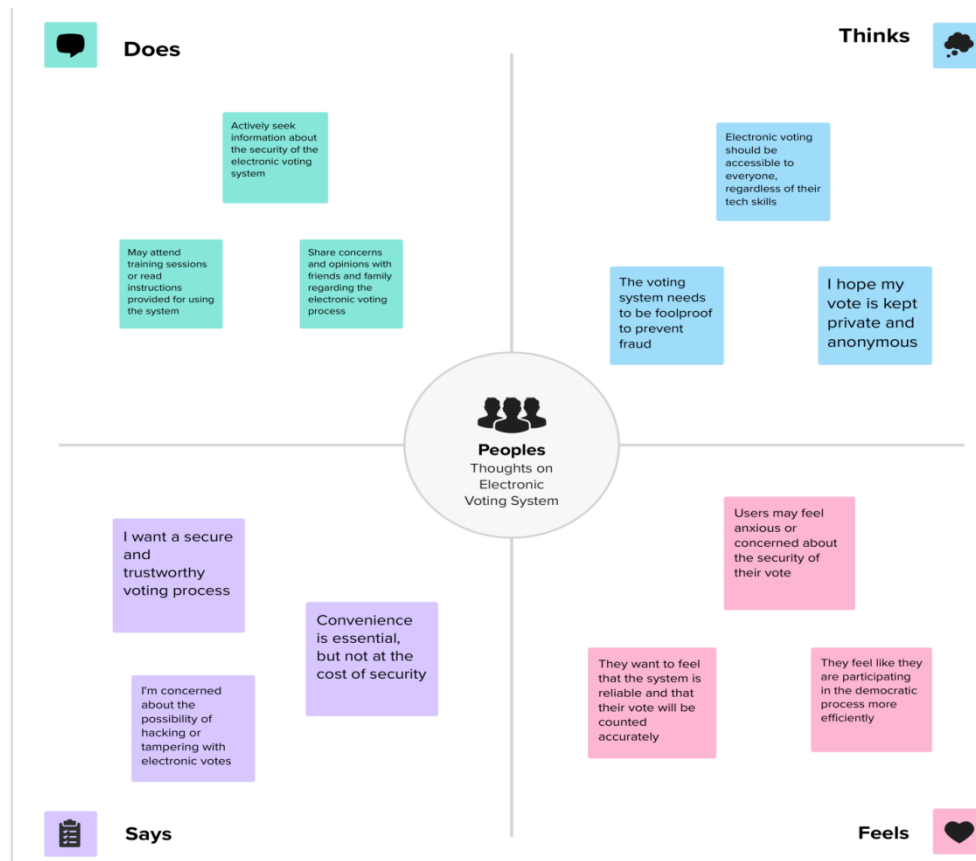
VoteSmart offers educational resources, including articles, videos, and infographics, to help voters understand complex policy issues and the electoral process

Introduce gamification elements to incentivize participation, such as earning badges or points for engaging in discussions, verifying information, and casting vote

Screenshot 3

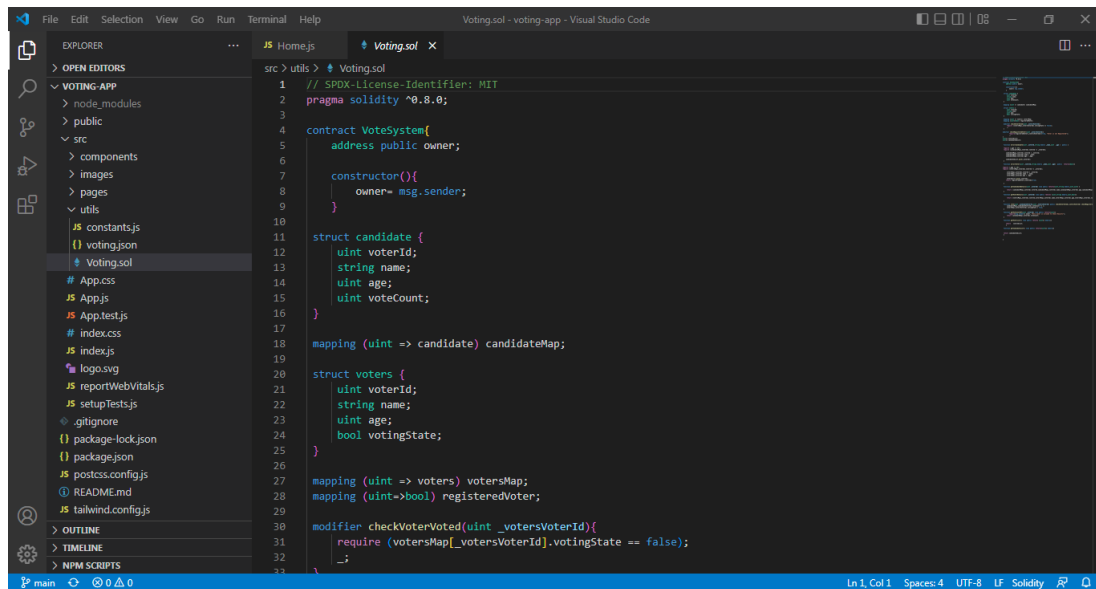


Screenshot 4



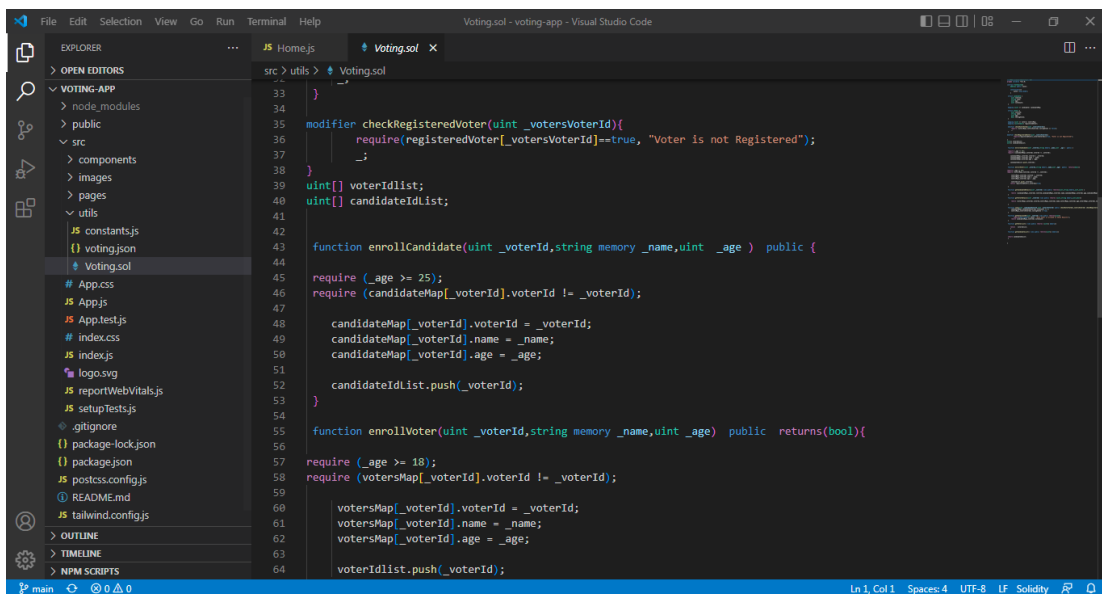
## 2. EXTRACT THE ZIP FILE

Screenshot 1



```
src > utils > Voting.sol
1 // SPDX-License-Identifier: MIT
2 pragma solidity ^0.8.0;
3
4 contract VoteSystem{
5     address public owner;
6
7     constructor(){
8         owner= msg.sender;
9     }
10
11     struct candidate {
12         uint voterId;
13         string name;
14         uint age;
15         uint voteCount;
16     }
17
18     mapping (uint => candidate) candidateMap;
19
20     struct voters {
21         uint voterId;
22         string name;
23         uint age;
24         bool votingState;
25     }
26
27     mapping (uint => voters) votersMap;
28     mapping (uint=>bool) registeredVoter;
29
30     modifier checkVoterVoted(uint _votersVoterId){
31         require (votersMap[_votersVoterId].votingState == false);
32         _;
33     }
```

Screenshot 2



```
src > utils > Voting.sol
33 }
34
35 modifier checkRegisteredVoter(uint _votersVoterId){
36     require(registeredVoter[_votersVoterId]==true, "Voter is not Registered");
37     _;
38 }
39 uint[] voterIdList;
40 uint[] candidateIdList;
41
42
43 function enrollCandidate(uint _voterId,string memory _name,uint _age ) public {
44
45     require (_age >= 25);
46     require (candidateMap[_voterId].voterId != _voterId);
47
48     candidateMap[_voterId].voterId = _voterId;
49     candidateMap[_voterId].name = _name;
50     candidateMap[_voterId].age = _age;
51
52     candidateIdList.push(_voterId);
53 }
54
55 function enrollVoter(uint _voterId,string memory _name,uint _age) public returns(bool){
56
57     require (_age >= 18);
58     require (votersMap[_voterId].voterId != _voterId);
59
60     votersMap[_voterId].voterId = _voterId;
61     votersMap[_voterId].name = _name;
62     votersMap[_voterId].age = _age;
63
64     voterIdList.push(_voterId);
65 }
```

Screenshot 3

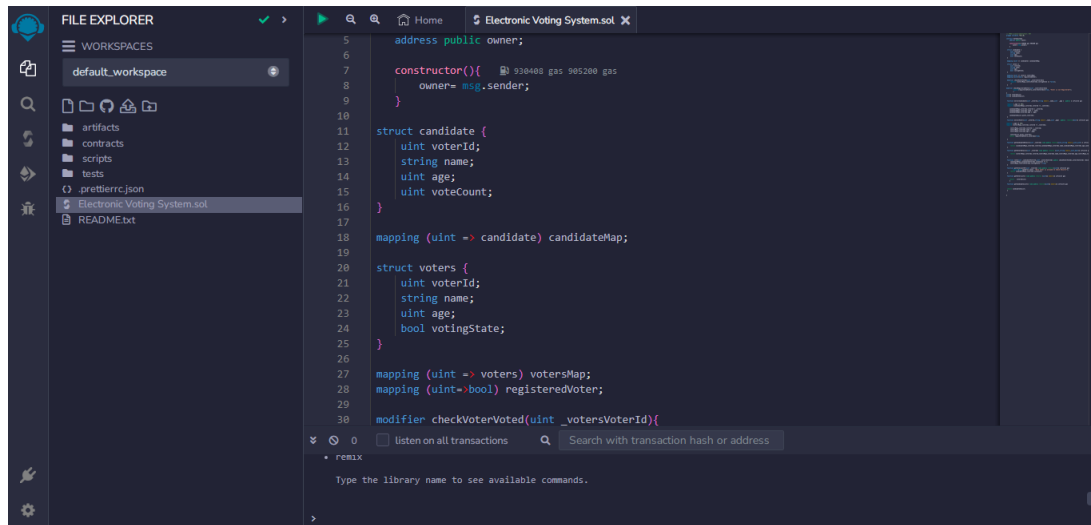
```
src > utils > Voting.sol
64 voterIdList.push(_voterId);
65 return registeredVoter[_voterId]=true;
66
67
68
69
70
71 function getCandidateDetails(uint _voterId) view public returns(uint,string memory,uint,uint) {
72     return (candidateMap[_voterId].voterId,candidateMap[_voterId].name,candidateMap[_voterId].age,candidateMap
73 }
74
75 function getVoterDetails(uint _voterId) view public returns (uint,string memory,uint,bool){
76     return (votersMap[_voterId].voterId,votersMap[_voterId].name,votersMap[_voterId].age,votersMap[_voterId].vc
77 }
78
79
80 function vote(uint _candidateVoterId,uint _votersVoterId) public checkVoterVoted(_votersVoterId) checkRegistere
81     candidateMap[_candidateVoterId].voteCount += 1;
82     votersMap[_votersVoterId].votingState = true;
83 }
84
85 function getVoteCountoff(uint _voterId) view public returns(uint){
86     require(msg.sender== owner, "Only owner is allowed to Check Results");
87     return candidateMap[_voterId].voteCount;
88 }
89
90 function getVoterList() view public returns (uint[] memory){
91     return voterIdList;
92 }
93
94
95 function getCandidateList() view public returns(uint[] memory){
```

Screenshot 4

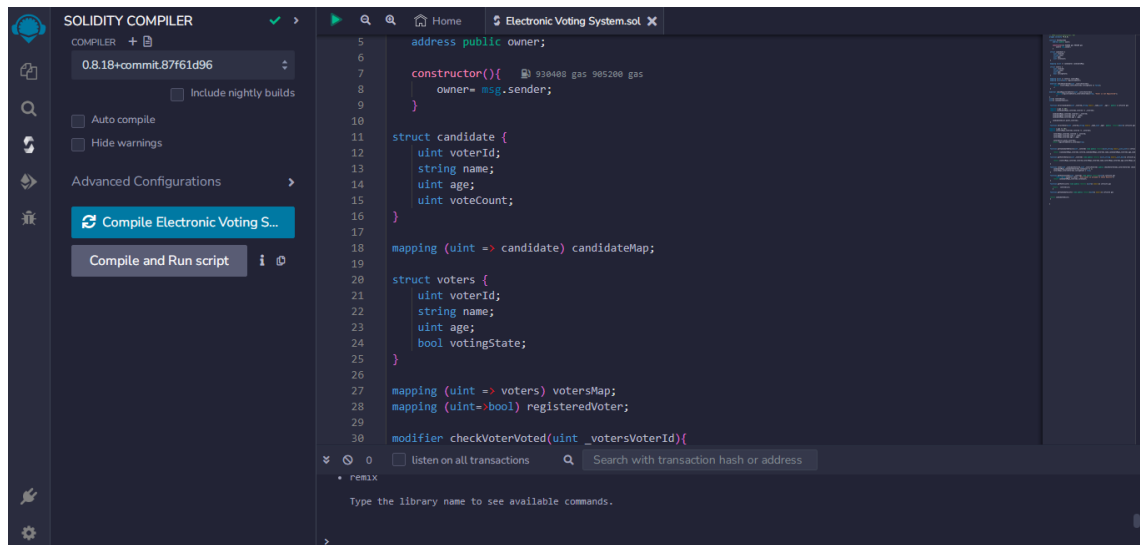
```
src > utils > Voting.sol
96
97
98 return candidateIdList;
99 }
100
101
102
103 }
```

### 3. Remix Ide platform Explorting

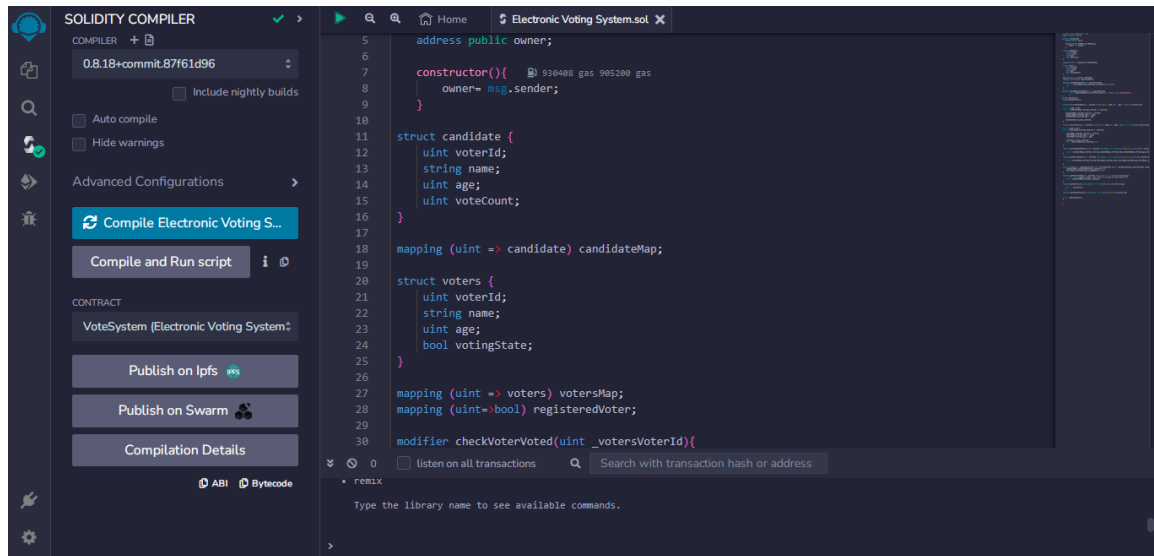
Screenshot 1



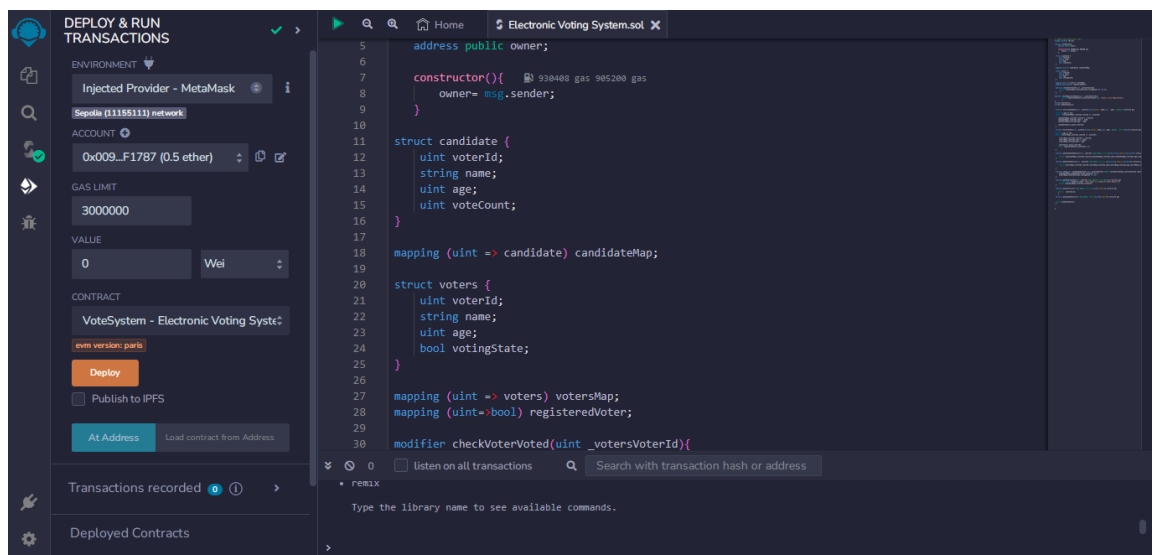
Screenshot 2



Screenshot 3

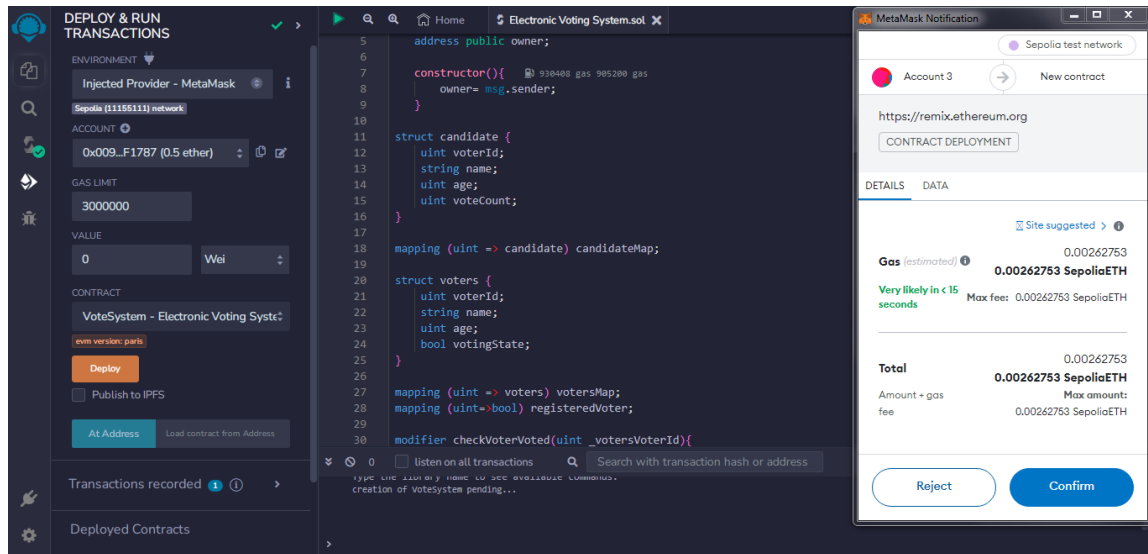


Screenshot 4

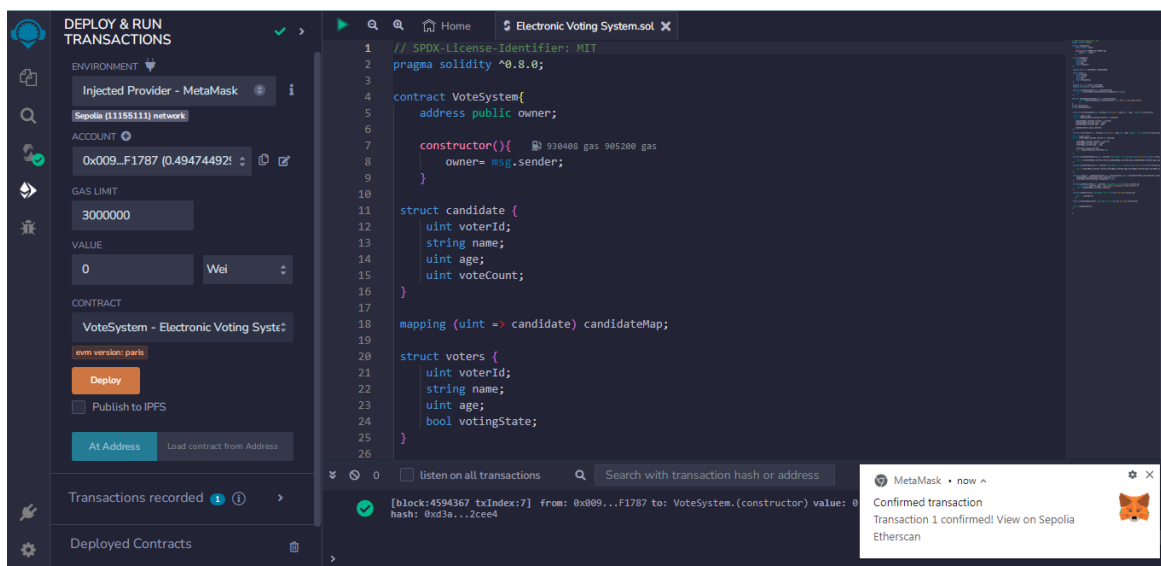




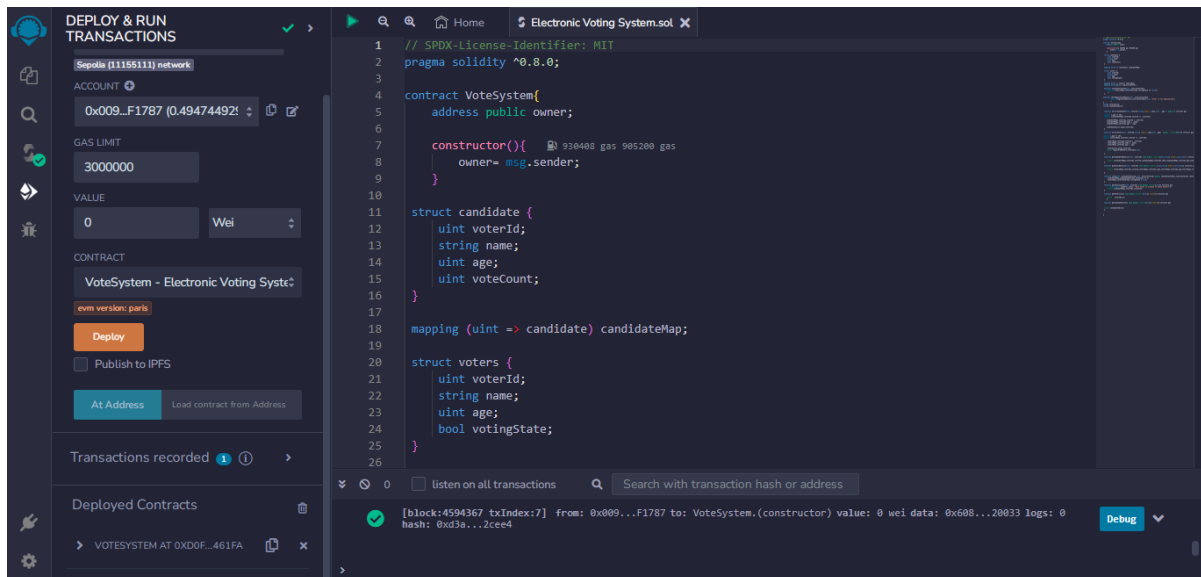
Screenshot 5



Screenshot 6

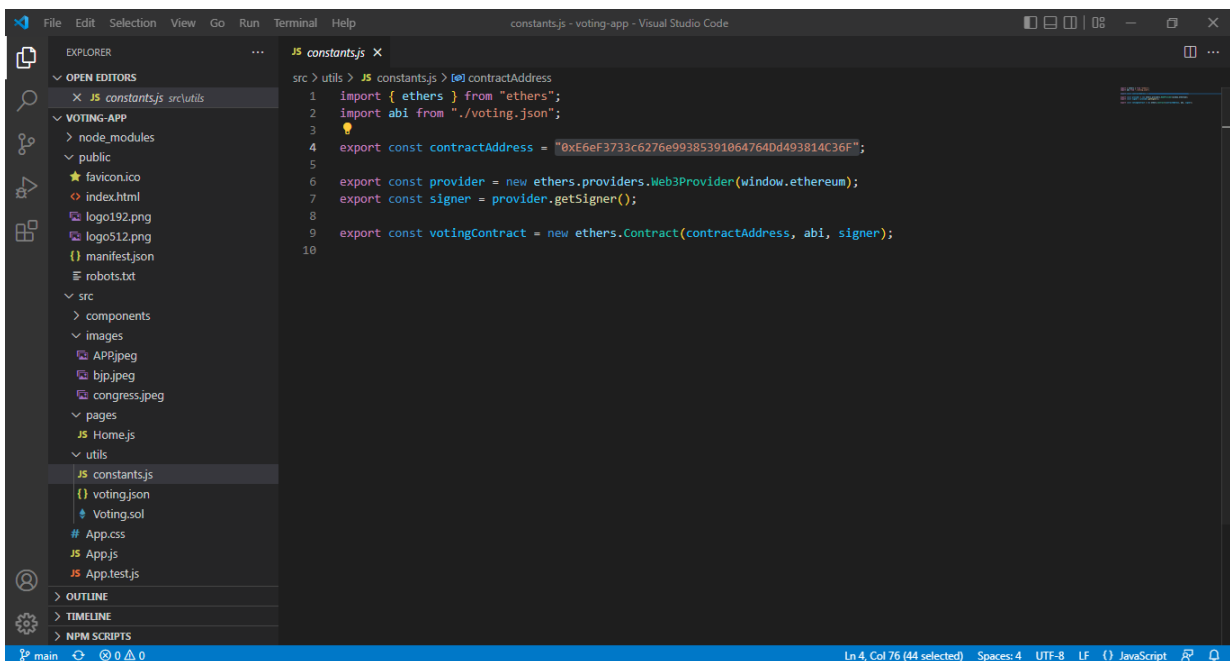


## Screenshot 7

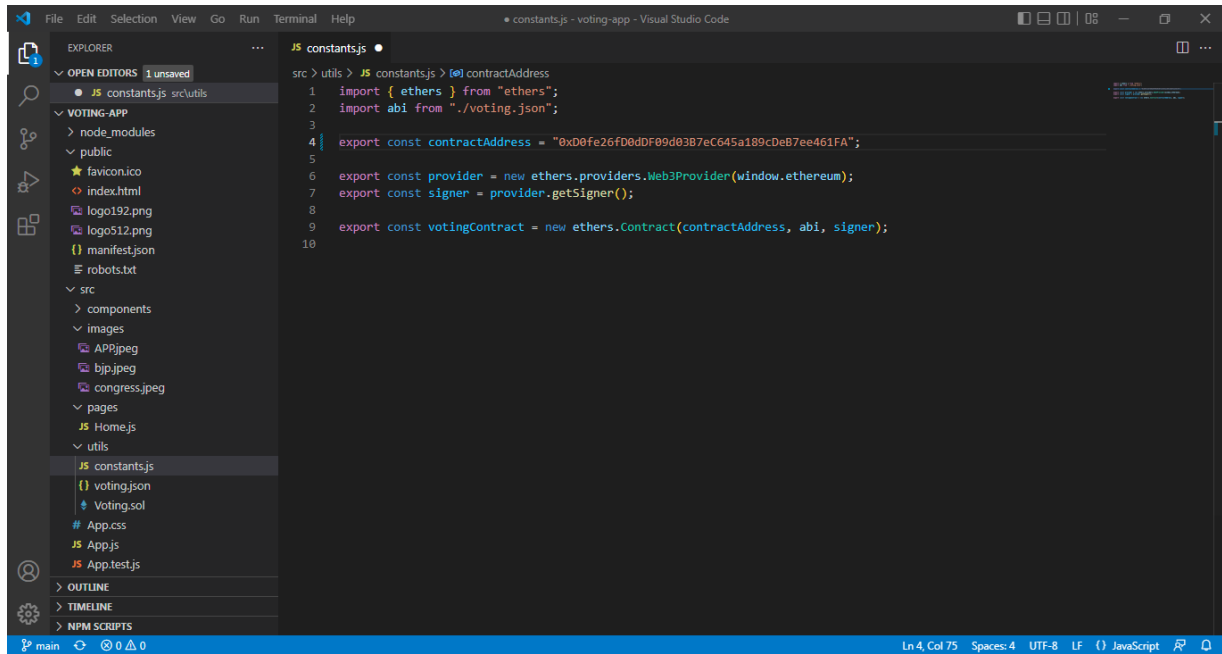


## 4. OPEN THE FILE EXPLORER

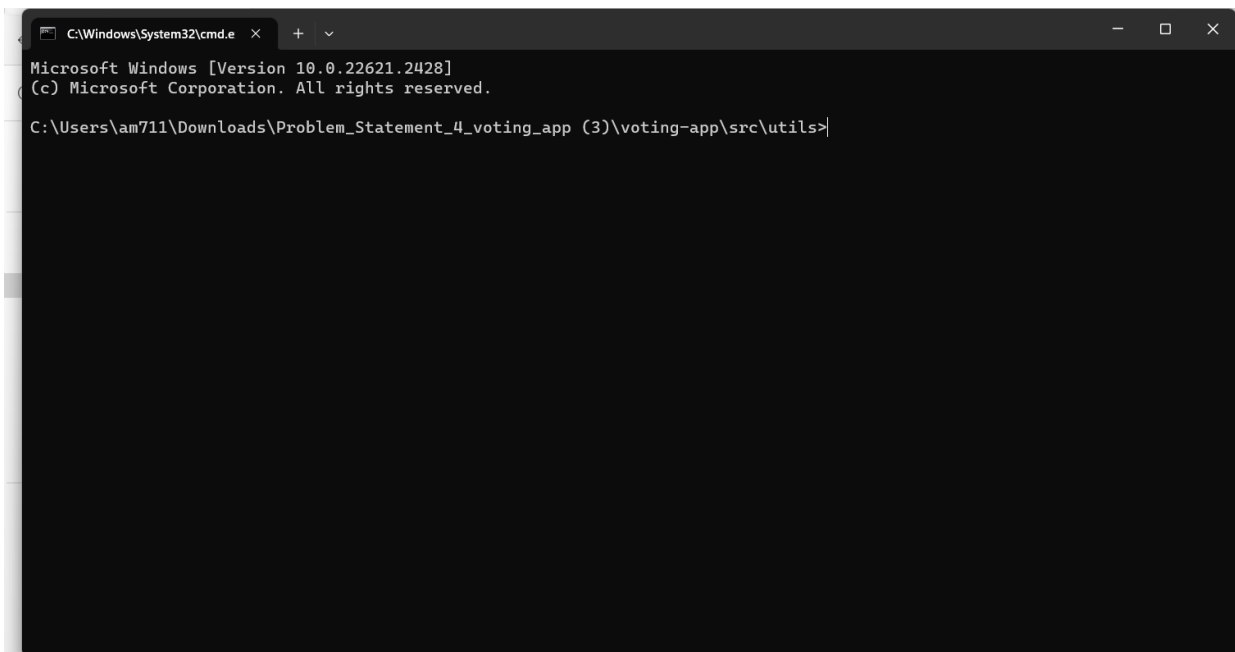
### Screenshot 1



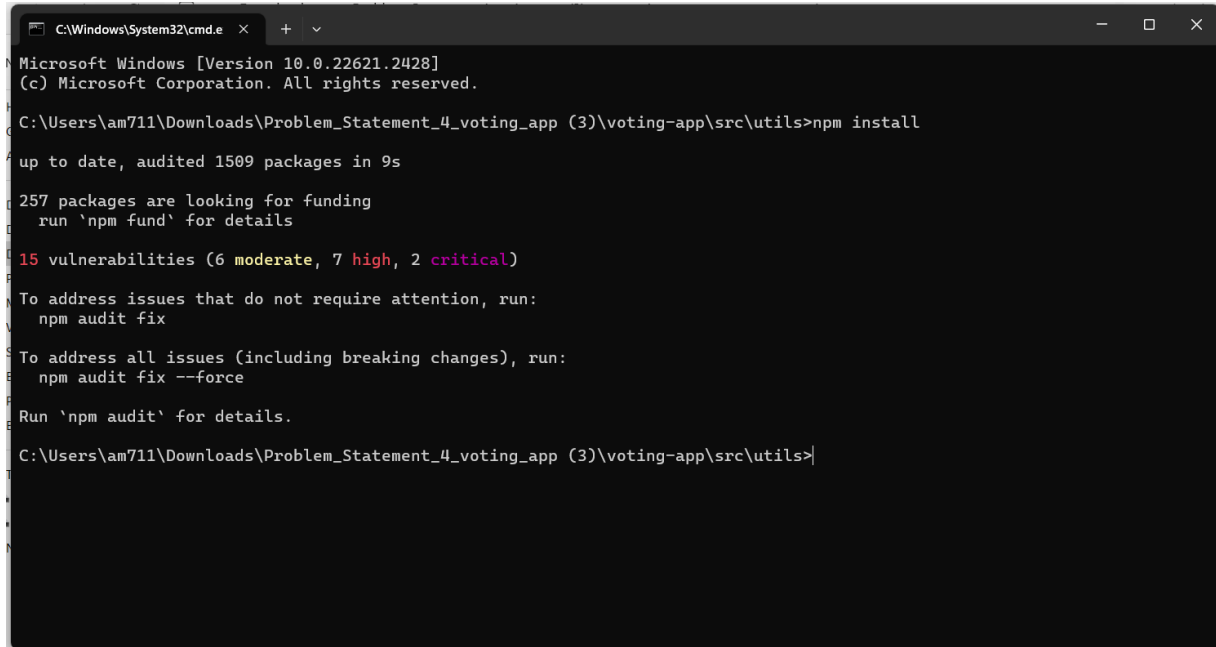
## Screenshot 2



## Screenshot 3



Screenshot 4



```
C:\Windows\System32\cmd.e x + v
Microsoft Windows [Version 10.0.22621.2428]
(c) Microsoft Corporation. All rights reserved.

C:\Users\am711\Downloads\Problem_Statement_4_voting_app (3)\voting-app\src\utils>npm install

up to date, audited 1509 packages in 9s

257 packages are looking for funding
  run `npm fund` for details

15 vulnerabilities (6 moderate, 7 high, 2 critical)

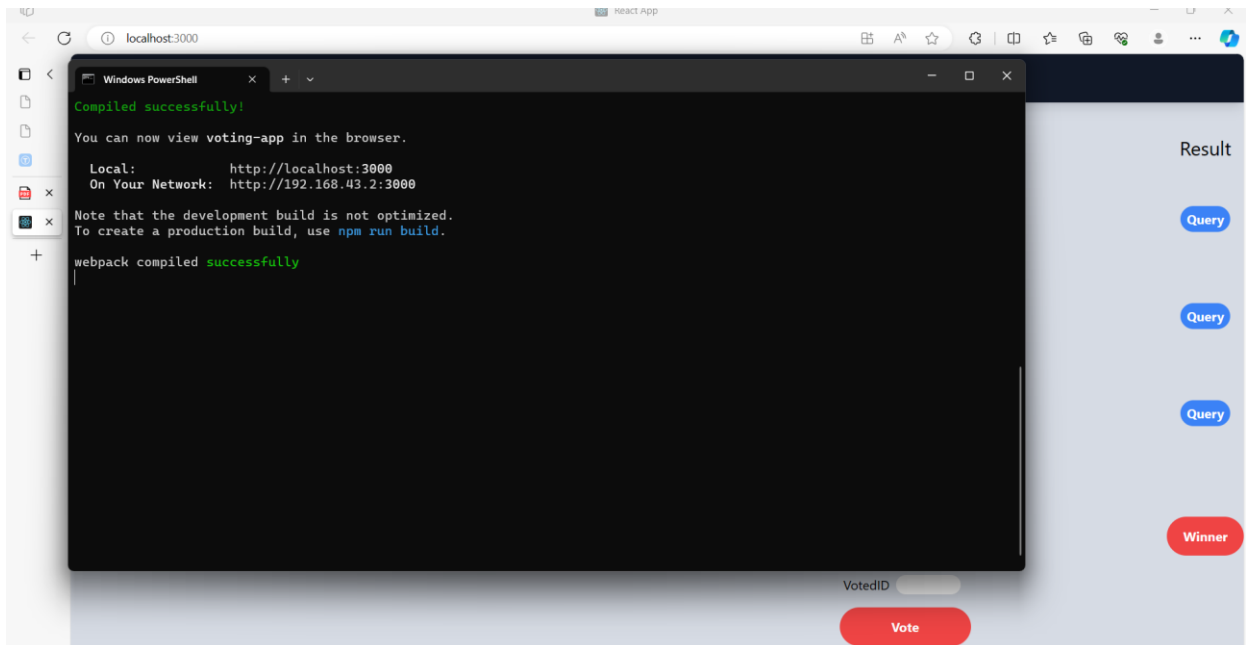
To address issues that do not require attention, run:
  npm audit fix

To address all issues (including breaking changes), run:
  npm audit fix --force

Run `npm audit` for details.

C:\Users\am711\Downloads\Problem_Statement_4_voting_app (3)\voting-app\src\utils>
```

Screenshot 5



## 5. LOCALHOST IP ADDRESS

The screenshot shows a web application titled "Election Commission of India" running on a browser at the address `localhost:3000`. The application has a dark blue header with the title and a "Connect Wallet" link. The main content area is divided into several sections:

- Candidate Registration:** Includes a "Candidate ID" dropdown menu (showing "1"), "Candidate Name" and "Candidate age" input fields, and a blue "Register" button.
- Voter Registration:** Includes "VotedID", "Voter Name", and "Voter Age" input fields, and a blue "Register" button.
- Vote:** Displays three party logos with corresponding radio buttons:
  - BJP ID - 1 (Logo: Lotus flower)
  - TRS ID - 2 (Logo: Telangana Rashtra Samithi)
  - CONGRESS ID - 3 (Logo: Hand holding a torch)Below these is a "VotedID" input field and a red "Vote" button.
- Result:** A vertical column on the right with three blue "Query" buttons and a red "Winner" button at the bottom.

<http://localhost:3000>