Online Blockchain Voting System

Showcase Presentation

Plagiarism Declaration:

I declare that this is all my own work. Any material I have referred to has been accurately referenced and any contribution of Artificial Intelligence technology has been fully acknowledged. I have read the University's policy on academic misconduct and understand the different forms of academic misconduct. If it is shown that material has been falsified, plagiarised, or I have otherwise attempted to obtain an unfair advantage for myself or others, I understand that I may face sanctions in accordance with the policies and procedures of the University. A mark of zero may be awarded and the reason for that mark will be recorded on my file.

The Problem

LACK OF SECURE SYSTEMS

INEFFICIENT PROCESSES

TRANSPARENCY GAPS

NEED FOR ANONYMITY

SCALABILITY

My Solution


A website to host elections.


Organisers can set up elections with multiple ballot types.


Secure and anonymous voting through blockchain.

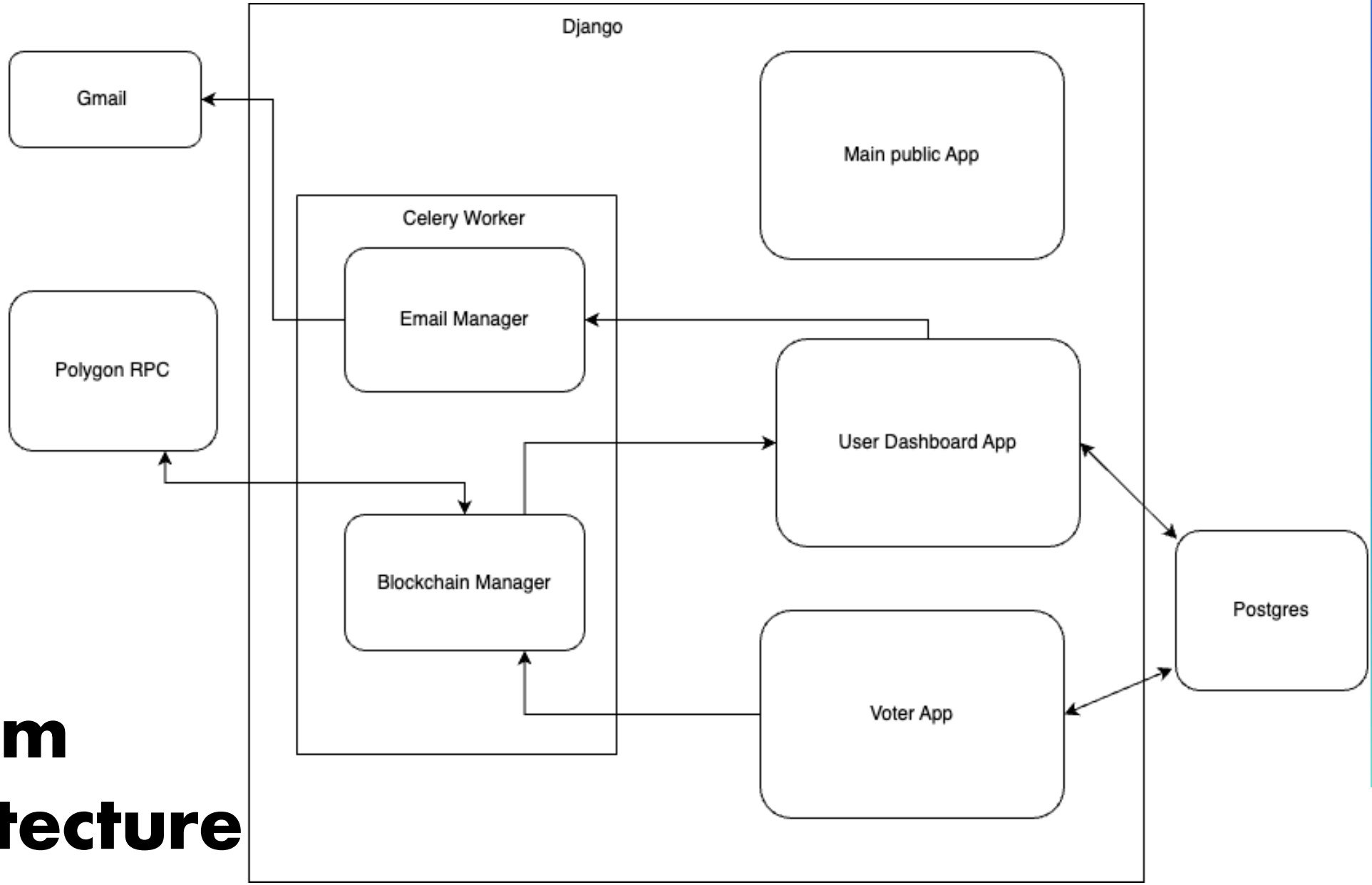

Results are clearly presented after voting ends.


Designed for small-scale use, like organisational or community voting.

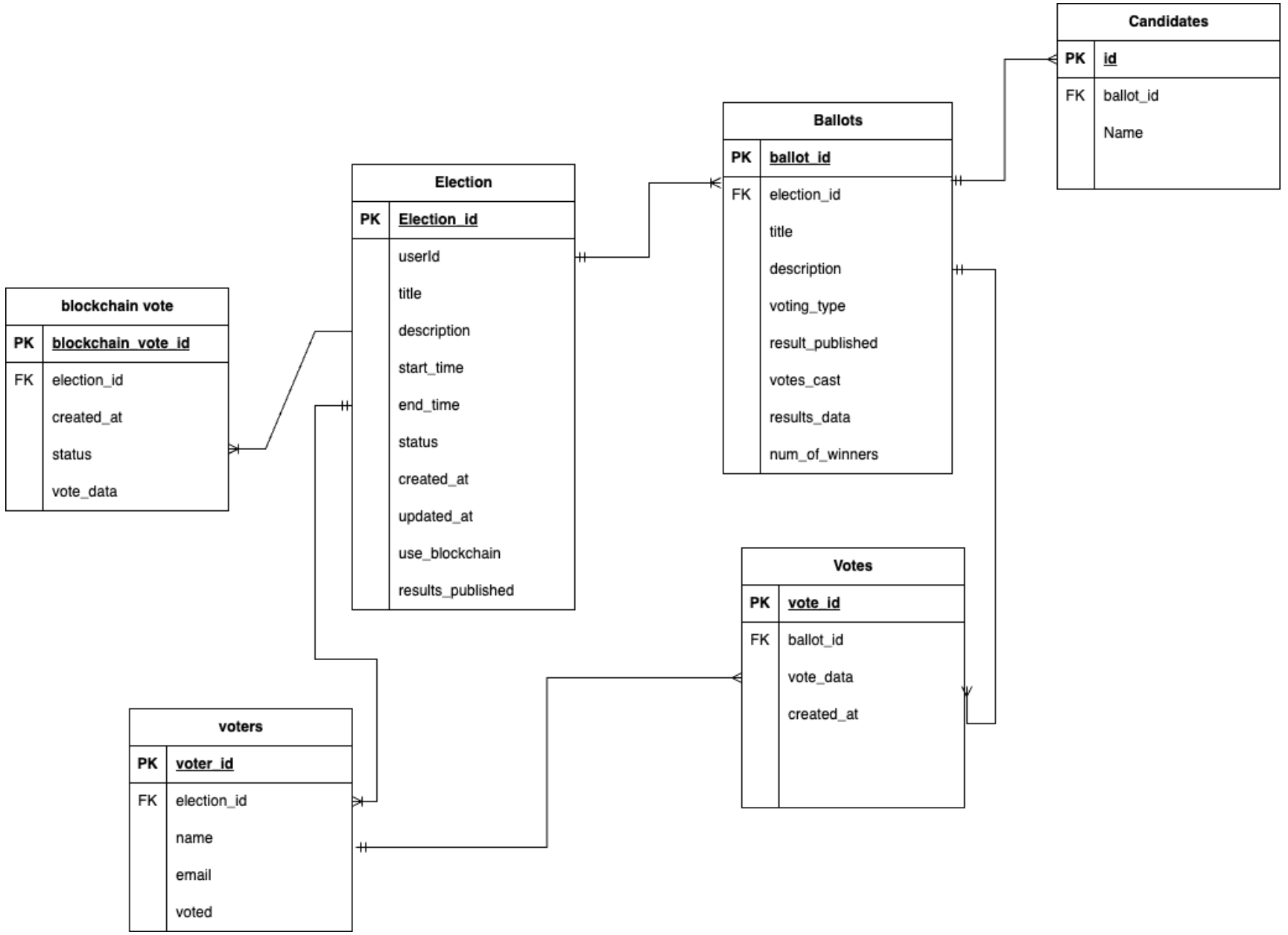
Key Features

- Informational Site for public
- Admin dashboard
- Voting Platform
- Blockchain Voting
- Multiple Voting Types

System Architecture

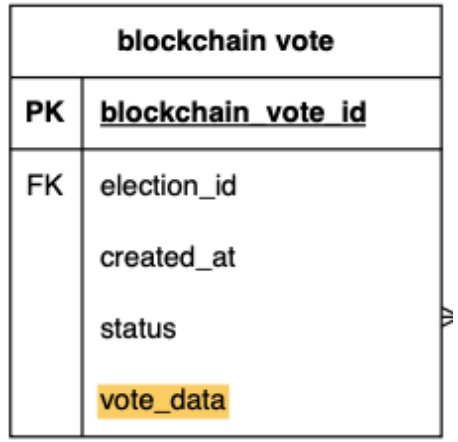


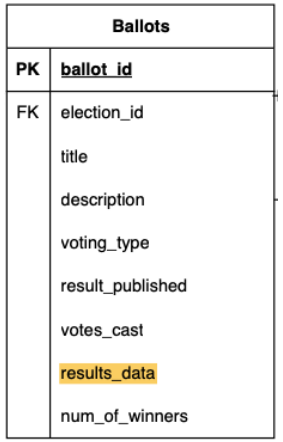
Database Schema

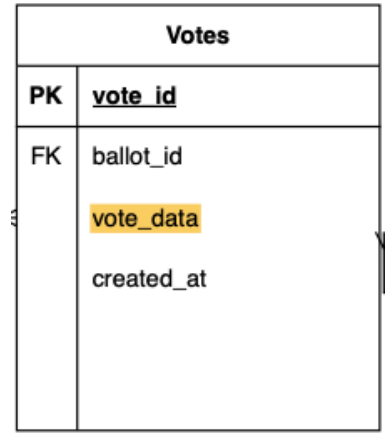


Encryption

- Fernet Python Library
- Uses symmetric Encryption
- Sensitive Database data is encrypted
- Vote data on blockchain encrypted







Counting Algorithms

FPTP voting

Ranked choice

Ranked Choice

1

Count first-choice preferences

2

Set quota

3

Process rounds

Check if quota is met

Eliminate

Redistribute

Repeat

Blockchain Integration

+ Polygon RPC
+ Blockchain Manager Class
+ Async Task
+ Smart contract

```
contract VotingStorage {
    mapping(bytes32 => string) public voteStorage;

    function storeVote(string calldata voteId, string calldata voteData) public {

        bytes32 key = keccak256(abi.encodePacked(voteId));

        voteStorage[key] = voteData;
    }

    function retrieveVote(string calldata voteId) public view returns (string memory) {
        bytes32 key = keccak256(abi.encodePacked(voteId));

        return voteStorage[key];
    }
}
```

Smart contract

Next Steps

- More types of voting
- Weighted votes
- More account personalisation (organisation name etc.)
- Publish results to voters or make public
- More social networking features (voter accounts)

Thank you for listening to my presentation