**ONLINE VOTEING SYSTEM USING BLOCKCHAIN**

**ABSTRACT:**

The Online Voting System Using Blockchain is a cutting-edge solution designed to address the vulnerabilities inherent in traditional voting systems. Leveraging blockchain technology, the system introduces a decentralized and secure platform for conducting elections. Users undergo a seamless registration and Aadhaar verification process, ensuring the authenticity of participants. Each vote is recorded as a tamper-resistant block in the blockchain, providing an immutable and transparent ledger of the entire voting process. The system employs advanced cryptographic techniques to safeguard user data and communication, enhancing the overall security of the platform. An intuitive admin panel empowers election authorities to manage voter lists, monitor the voting process, and ensure a fair and efficient election. The results, displayed through a transparent view of the blockchain, allow voters to independently verify and validate the outcome, fostering trust in the democratic process. Overall, the Online Voting System Using Blockchain presents a robust and trustworthy solution that revolutionizes election systems by promoting transparency, security, and integrity.In summary, this innovative system addresses the challenges of traditional voting systems through the integration of blockchain technology, providing a secure and transparent platform for conducting elections.

**CHAPTER-1**

**INTRODUCTION:**

Elections form the bedrock of democratic societies, providing citizens with a means to express their political will and shape the course of governance. However, the traditional methods of conducting elections have faced persistent challenges, including concerns related to transparency, security, and the potential for fraud. In response to these issues, the Online Voting System Using Blockchain emerges as a pioneering solution that harnesses the transformative capabilities of blockchain technology to redefine the electoral landscape. Blockchain, renowned for its decentralized and tamper-resistant nature, offers a novel approach to ensuring the integrity and transparency of the voting process. This system aims to revolutionize how elections are conducted, instilling trust and confidence in voters by providing a secure, auditable, and verifiable platform.The Online Voting System Using Blockchain encompasses a comprehensive set of features designed to address the shortcomings of traditional voting systems. From secure user registration and Aadhaar verification to the recording of votes in an immutable blockchain, each element of the system contributes to fostering a reliable and transparent electoral environment. This introduction sets the stage for a detailed exploration of how this innovative voting system leverages blockchain technology to overcome existing challenges, ensuring the democratic process is robust, trustworthy, and reflective of the collective voice of the electorate.

**EXISTING SYSTEM:**

The traditional paper-based voting system has been the cornerstone of democratic elections for decades. In this conventional approach, voters physically visit polling stations to cast their votes using paper ballots. The process involves manual counting and tallying of votes, with election authorities responsible for overseeing the accuracy and fairness of the election. While this system has been widely used and is familiar to many, it is not without its challenges.One of the primary drawbacks of the traditional voting system is the potential for human error and inefficiencies in the manual counting process. The reliance on paper ballots makes the system susceptible to miscounts, accidental loss of ballots, and delays in announcing results. Additionally, the physical nature of the process limits accessibility for certain segments of the population, such as individuals with disabilities or those residing in remote areas. In light of these challenges, the need for a more technologically advanced and secure voting system has become evident. The existing paper-based system, while time-tested, is ripe for improvement to enhance efficiency, accuracy, and overall confidence in the democratic process. This recognition has led to the exploration and development of innovative solutions, such as the Online Voting System Using Blockchain, to address the shortcomings of the traditional approach.

**DISADVANTAGES:**

* Despite the potential for increased accessibility, the Online Voting System Using Blockchain may exacerbate existing digital divides. Citizens without reliable internet access or those unfamiliar with technology may face barriers to participation. This could result in a biased representation of the electorate, as segments of the population may be excluded from the voting process. Ensuring equitable access to the online platform becomes crucial to prevent disenfranchisement.
* While blockchain is known for its security features, the online voting system is not immune to cybersecurity threats. Malicious actors may attempt to compromise the system through various means, such as Distributed Denial of Service (DDoS) attacks or exploiting vulnerabilities in the voting platform. Ensuring robust cybersecurity measures and constant monitoring are essential to safeguard the integrity of the election and maintain public trust.
* Establishing a secure and user-friendly authentication process is challenging. The system must verify voters' identities effectively while preserving their privacy. Striking the right balance between authentication and privacy protection is crucial to prevent fraudulent activities and unauthorized access. Issues related to the handling of personal data and the potential for data breaches must be carefully addressed to maintain the confidentiality of voter information.

**PROPOSED SYSTEM:**

The proposed Online Voting System Using Blockchain seeks to revolutionize the electoral process by leveraging cutting-edge technologies to address the shortcomings of traditional voting methods. Harnessing the power of blockchain, each vote is securely recorded in an immutable and transparent ledger, ensuring the integrity of the election results. The decentralized architecture enhances security, making it resistant to tampering or malicious interference. A user-friendly interface prioritizes accessibility, accommodating voters of varying technological proficiency and promoting inclusivity. Aadhaar verification adds an additional layer of authenticity, mitigating the risks associated with fraudulent voting. Real-time monitoring through an admin panel empowers election officials to manage the process efficiently, while a transparent view of the blockchain allows voters to independently verify the results. This proposed system not only enhances the security and transparency of elections but also strives to create a more accessible and trustworthy democratic voting experience.

**ADVANTAGES:**

* Utilizing blockchain technology, the online voting system offers a significantly higher level of security compared to traditional voting methods. Each vote is recorded as a block in the blockchain, creating a tamper-resistant and immutable ledger. The decentralized and cryptographic nature of blockchain ensures that once a vote is cast, it cannot be altered or manipulated. Blockchain's transparent and decentralized nature contributes to increased transparency in the voting system. Every participant in the election process, including voters, can have access to a transparent view of the blockchain to independently verify and audit the election results. The ability to trace every vote back to its origin adds a layer of accountability, making the entire process more robust and credible.
* The adoption of an online voting system simplifies and streamlines the entire voting process. Voters can cast their ballots remotely, reducing the need for physical polling stations and minimizing logistical challenges. This results in increased efficiency, as the election authorities can manage voter registration, monitor real-time voting trends, and process results more expeditiously.
* The online voting system using blockchain transcends geographical constraints, allowing eligible voters to participate in elections from virtually anywhere with internet access.The system promotes inclusivity by accommodating a diverse range of voters, including those with disabilities, thereby fostering a more democratic and participatory electoral environment.

**SOFTWARE SPECIFICATION:**

**HARDWARE CONFIGURATION:**

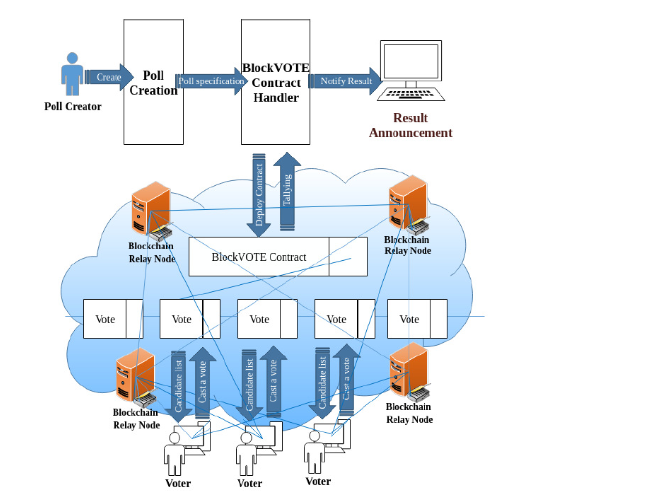
* Processor - i5
* Speed - 3 GHz
* RAM - 4 GB(min)S
* Hard Disk - 500 GB
* Key Board - Standard Windows Keyboard
* Mouse - Two or Three Button Mouse
* Monitor - SVGA

**SOFTWARE CONFIGURATION:**

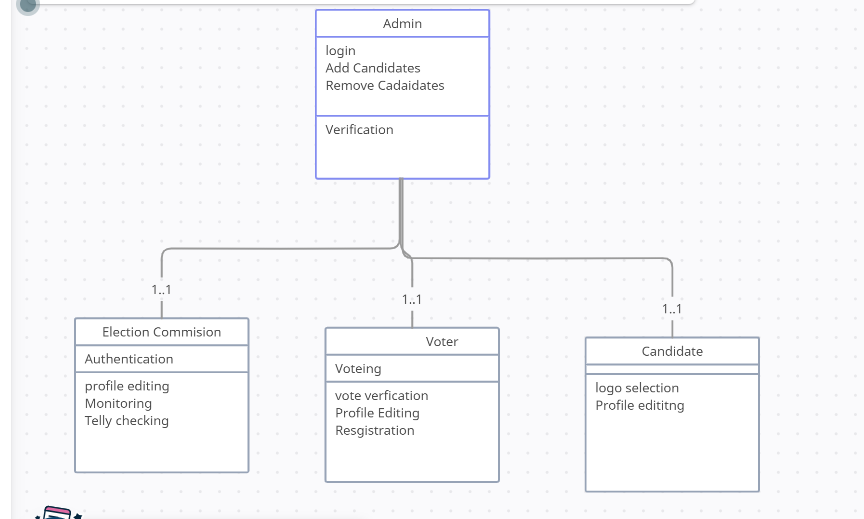
* Operating System: Linux, Windows/7/10
* Tools: Anaconda, Jupiter, vscode

Server side Script: Python , AIML

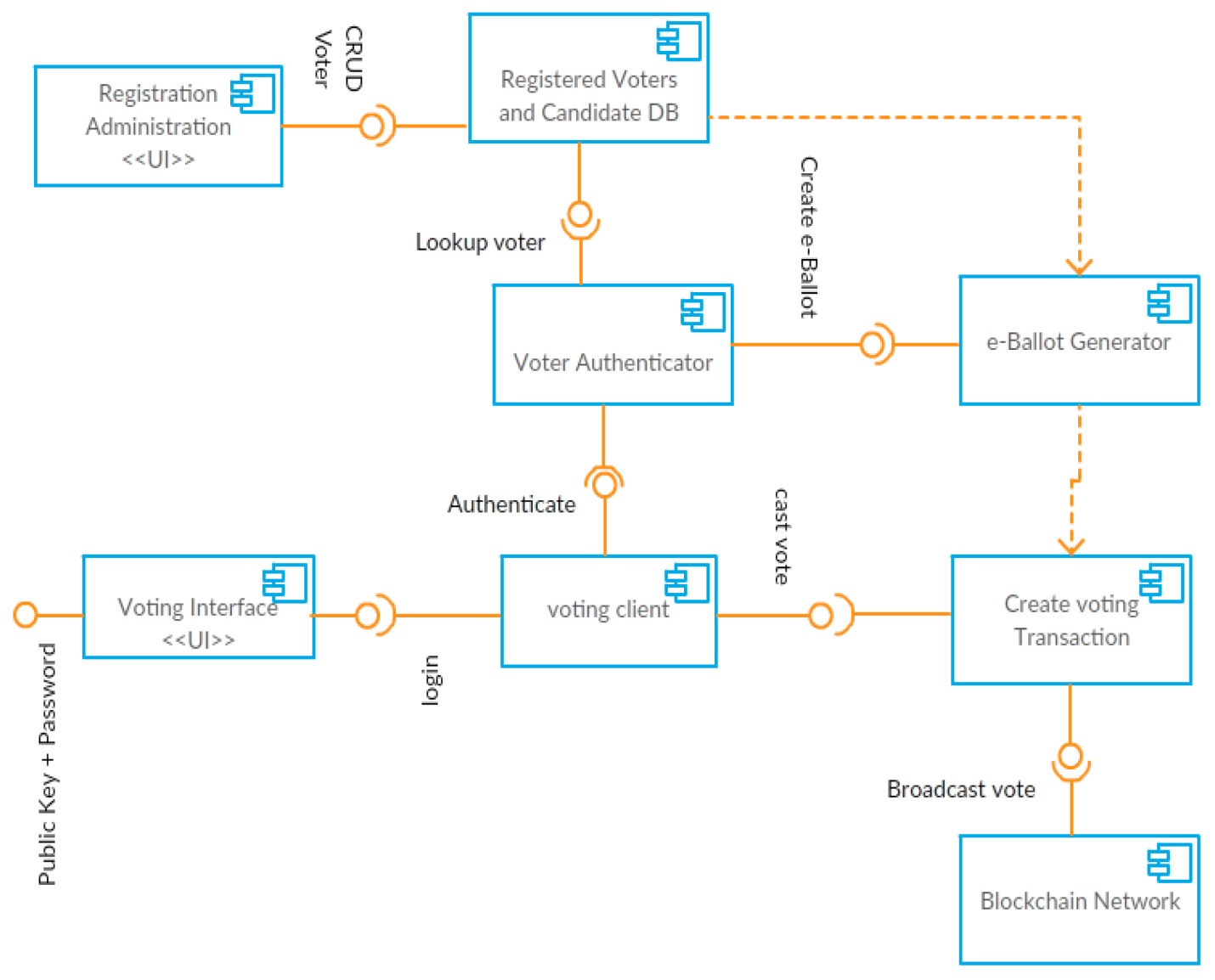
**SYSTEM ARCHITECTURE DIAGRAM:**



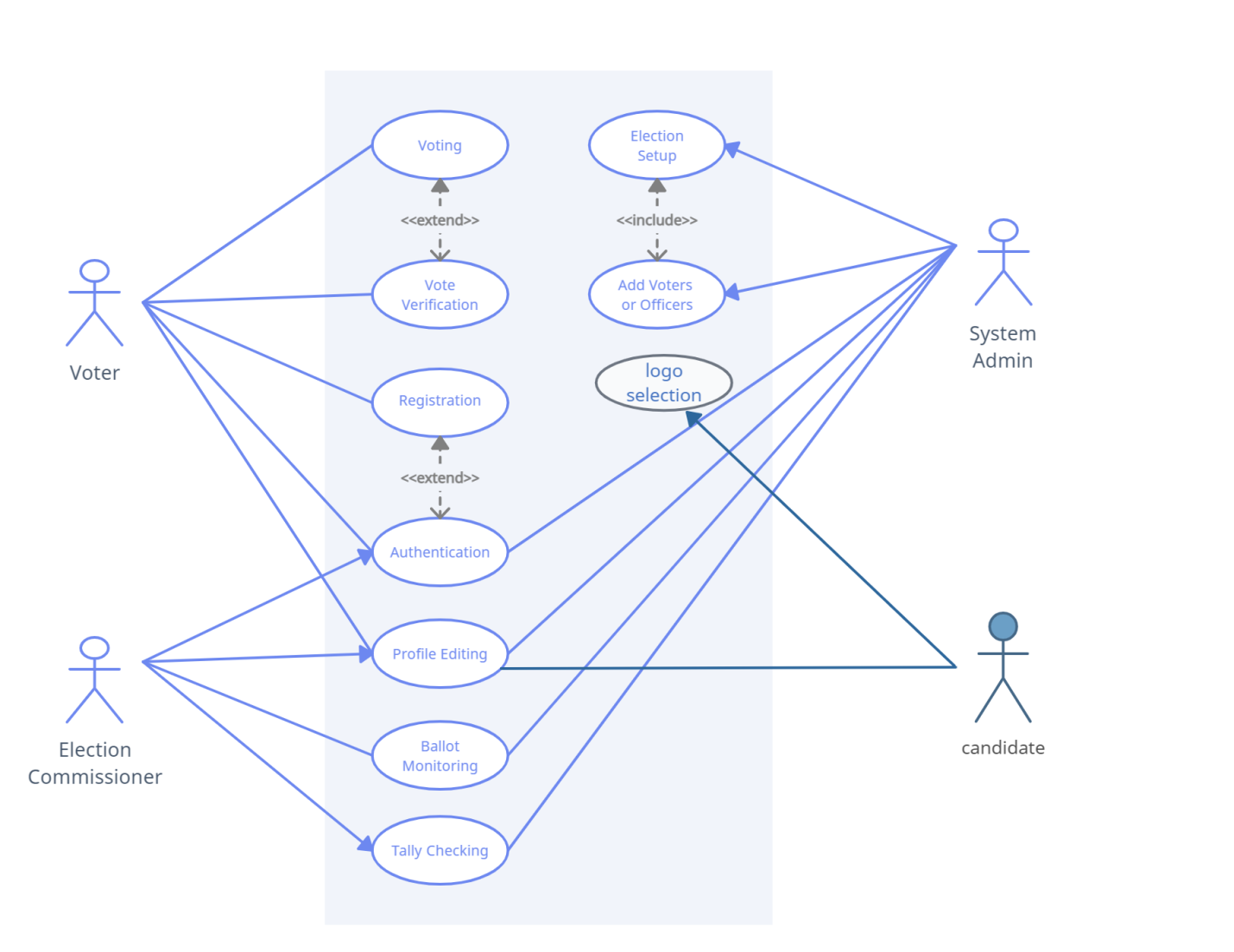
**CLASS DIAGRAM:**



**DATAFLOW DIAGRAM:**



**USE CASE DIAGRAM:**



**ER DIAGRAM:**

Election commision

Candidate

Authentication

and Result

Registration

Admin

Voter

System maintanence

Result

**CHAPTER-2**

**RELATED WORK:**

Several online voting systems and blockchain-based solutions have been explored to address the challenges and improve the efficiency of electoral processes. One notable related work is "Follow My Vote," an online voting platform that integrates blockchain technology to enhance transparency and security. Follow My Vote employs blockchain for voter authentication and to create an immutable ledger of votes, ensuring a tamper-resistant record. Voatz has been implemented in various pilot projects and elections, aiming to provide a convenient and accessible voting experience while maintaining the integrity of the electoral process through blockchain encryption. Furthermore, Estonia's e-Government system stands out as a pioneering example. While not exclusively focused on elections, Estonia has successfully integrated blockchain into its digital infrastructure for secure e-governance, including aspects of online voting. The country's innovative approach demonstrates the potential for blockchain to enhance the overall functioning of a government's digital services.These related works highlight the ongoing efforts to leverage blockchain in the realm of online voting, emphasizing security, transparency, and accessibility as key objectives in the evolution of electoral systems.

**LITERATURE SURVEY REFERENCES:**

# Title: “Online Voting System”

**Authors:** [Ketan](https://ieeexplore.ieee.org/author/37089670629), [Shanu Khare](https://ieeexplore.ieee.org/author/37089671454), [Maninder Kaur](https://ieeexplore.ieee.org/author/37089674669)

**Year:** 2022

**Abstract:** This analysis has shown that candidate selection often involves politics, and that whenever voters vote for their elections, they end up with a typical exploitation and manual approach. Manually selecting it can usually lead to incorrect behavior. Therefore, it is necessary to introduce an online voting system. This particular analysis envisions implementing an online voting system with options such as: B. A system implemented by each party and supported by options that tend to participate in voting. The main reason to switch from a traditional voting system to an online voting system is to save time and allow you to vote online from anywhere.

# Title: “Smart Online Voting System”

**Authors:** [S Ganesh Prabhu](https://ieeexplore.ieee.org/author/37088880986), [A Nizarahammed.](https://ieeexplore.ieee.org/author/37088881464)

**Year:** 2021

**Abstract:** India has only offline voting system which is not effective and upto the mark as it requires large man force and it also requires more time to process and publish the results. Therefore, to be made effective, the system needs a change, which overcomes these disadvantages. The new method does not force the person's physical appearance to vote, which makes the things easier. This paper focusses on a system where the user can vote remotely from anywhere using his/her computer or mobile phone and doesn't require the voter to got to the polling station through two step authentication of face recognition and OTP system. The face scanning system is used to record the voters face prior to the election and is useful at the time of voting. This system also enables the user the citizens to see the results anytime which can avoid situations that pave way to vote tampering.

# Title: “Online voting system for India based on AADHAAR ID”

**Authors:** [Himanshu Agarwal](https://ieeexplore.ieee.org/author/37086669610), [G. N. Pandey](https://ieeexplore.ieee.org/author/37086667460)

**Year:** 2013

**Abstract:** An online voting system for Indian election is proposed for the first time in this paper. The proposed model has a greater security in the sense that voter high security password is confirmed before the vote is accepted in the main database of Election Commission of India. The additional feature of the model is that the voter can confirm if his/her vote has gone to correct candidate/party. In this model a person can also vote from outside of his/her allotted constituency or from his/her preferred location. In the proposed system the tallying of the votes will be done automatically, thus saving a huge time and enabling Election Commissioner of India to announce the result within a very short period.

# Title: “Implementation of authenticated and secure online

# voting system”

**Authors:** [Srivatsan Sridharan](https://ieeexplore.ieee.org/author/37073068400)

**Year:** 2013

**Abstract:** This paper aims at creation of a voting system by providing a cost effective solution to the government along with ensuring non-traceability and integrity of the votes cast while providing great convenience to voters. This system is developed robustly to ensure that all eligible voters having a Universal Identification Number of their country (For Example the Smart Card in USA) is allowed to cast their respective vote. The voters, who cast multiple votes during the process of voting is ensured to be prevented. Also to ensure the maintenance of authenticity, any biometric identification of the voters could be used for accessing the terminal to cast their vote and restricting them to cast again. The process of online voting could be deployed with three phases - the voter registration online vote capturing and the instant online counting and result declaration. A Secret Voting Password provided to voter during registration acts as an authentication mechanism which enables the voters to securely cast their vote along with their captured biometric identification. A Simulation result of implementation of the same is described in this paper by describing the robustness of this system.

# Title: “Online Voting System Based on IoT and Ethereum Blockchain”

**Authors:** Awsan A. H. Othman, [Emarn A. A. Muhammed](https://ieeexplore.ieee.org/author/37088842237), [Haneen K. M. Mujahid](https://ieeexplore.ieee.org/author/37088845227), [Hamzah](https://ieeexplore.ieee.org/author/37088843261)

**Year:** 2021

**Abstract:** Voting process is a democratic practice that has been used over the years as a primary method used by people in democratic countries to express their opinions on issues and discussions that concern them. This paper seeks to facilitate and protect the voting process by making an online voting system for elections and referendums connected with voting devices. The IoT and Blockchain have been used with this system to ensure that users' data are protected from theft and prevent eavesdropping or vote tampering to guarantee the integrity of the voting. The blockchain encrypts votes in order to protect every vote from forgery. This system is not directed to governments only, but to all governmental and private agencies. For example, governments can establish referendums or elections, and anyone who has reached the legal age and has a voting card issued by the government will be able to vote, thus we get rid of the traditional methods and dispense with ballot boxes, standing in long queues and delay counting the votes that cost governments a lot of time, effort and money. The system assists the concerned authorities in obtaining results quickly without delay, taking into account the differences in voting process between government and private organizations.

**CHAPTER– 3**

**MODULE:**

* **Authentication and User Management**
* **Aadhaar Enrollment**
* Voting
* **Admin Panel**
* **Blockchain and Voting Results**
* Email and Messaging
* Testing Process (commented out)
* Miscellaneous

**MODULE DISCRIPTION:**

1. **Authentication and User Management:**

register: Handles user registration using a registration form.

login\_page: Manages user login with authentication.

logout\_page: Logs out the authenticated user.

home: Renders the home page.

1. **Aadhaar Enrollment:**

aadhaar: Manages the enrollment of Aadhaar numbers, including sending OTP for verification.

aadhaar\_next: Handles the next steps in Aadhaar enrollment, including OTP verification.

1. **Voting:**

voters: Displays a list of voters.

voterlist: Manages the creation of a list of voters.

voter\_details: Displays details of a specific voter.

mine\_block: Processes a vote and adds a new block to the

blockchain-like structure.

send\_hash: Sends a secret key (hash) to the user via email.

1. **Admin Panel:**

admin1: Handles login for the admin panel.

Blockchain and Voting Results:

view\_chain: Displays the blockchain structure and voting results.

1. **Email and Messaging:**

send\_otp: Sends OTP for verification.

send\_hash: Sends a secret key (hash) to the user.

Testing Process (commented out):

Includes commented-out code for testing and alternative approaches.

1. **Miscellaneous:**

Import statements for necessary modules and libraries.CSRF exemption for specific views.Various utility functions for email and messaging.

**CHAPTER 4**

**LANGUAGE DESCRIPTION:**

**PYTHON:**

Python is an interpreter, object-oriented, high-level programming language with dynamic semantics. Its high-level built in data structures, combined with dynamic typing and dynamic binding; make it very attractive for Rapid Application Development, as well as for use as a scripting or glue language to connect existing components together. Python's simple, easy to learn syntax emphasizes readability and therefore reduces the cost of program maintenance. Python supports modules and packages, which encourages program modularity and code reuse. The Python interpreter and the extensive standard library are available in source or binary form without charge for all major platforms, and can be freely distributed.

Often, programmers fall in love with Python because of the increased productivity it provides. Since there is no compilation step, the edit-test-debug cycle is incredibly fast. Debugging Python programs is easy: a bug or bad input will never cause a segmentation fault. Instead, when the interpreter discovers an error, it raises an exception. When the program doesn't catch the exception, the interpreter prints a stack trace. A source level debugger allows inspection of local and global variables, evaluation of arbitrary expressions, setting breakpoints, stepping through the code a line at a time, and so on. The debugger is written in Python itself, testifying to Python's introspective power. On the other hand, often the quickest way to debug a program is to add a few print statements to the source: the fast edit-test-debug cycle makes this simple approach very effective.It ranges from simple automation tasks to gaming, web development, and even complex enterprise systems. These are the areas

where this technology is still the king with no or little competence: Machine learning as it has a plethora of libraries implementing machine learning algorithms.Python is a one-stop shop and relatively easy to learn, thus quite popular now. What other reasons exist for such universal popularity of this programming language and what companies have leveraged its opportunities to the max? Let’s talk about that. Python technology is quite popular among programmers, but the practice shows that business owners are also Python development believers and for good reason. Software developers love it for its straightforward syntax and reputation as one of the easiest programming languages to learn. Business owners or CTOs appreciate the fact that there’s a framework for pretty much anything – from web apps to machine learning. Moreover, it is not just a language but more a technology platform that has come together through a gigantic collaboration from

thousands of individual professional developers forming a huge and peculiar community of aficionados. So what is python used for and what are the tangible benefits the language brings to those who decided to use it? Below we’re going to discover that. Productivity and Speed It is a widespread theory within development circles that developing Python applications is approximately up to 10 times faster than developing the same application in Java or C/C++. The impressive benefit in terms of time saving can be explained by the clean object-oriented design, enhanced process control capabilities, and strong integration and text processing capacities. Moreover, its own unit testing framework contributes substantially to its speed and productivity.

**PYCHARM:**

PyCharm is a dedicated Python Integrated Development Environment (IDE) providing a wide range of essential tools for Python developers, tightly integrated to create a convenient environment for productive Python, web, and data science development.

Choose the best PyCharm for you﻿

**PyCharm is available in three editions:**

* Community (free and open-sourced): for smart and intelligent Python development, including code assistance, refactorings, visual debugging, and version control integration.
* Professional (paid) : for professional Python, web, and data science development, including code assistance, refactorings, visual debugging, version control integration, remote configurations, deployment, support for popular web frameworks, such as Django and Flask, database support, scientific tools (including Jupyter notebook support), big data tools.
* Edu (free and open-sourced): for learning programming languages and related technologies with integrated educational tools.

|  |  |  |
| --- | --- | --- |
| **Requirement** | **Minimum** | **Recommended** |
| RAM | 4 GB of free RAM | 8 GB of total system RAM |
| CPU | Any modern CPU | Multi-core CPU. PyCharm supports multithreading for different operations and processes making it faster the more CPU cores it can use. |
| Disk space | 2.5 GB and another 1 GB for caches | SSD drive with at least 5 GB of free space |
| Monitor resolution | 1024x768 | 1920×1080 |
| Operating system | Officially released 64-bit versions of the following:   * Microsoft Windows 8 or later * macOS 10.13 or later * Any Linux distribution that supports Gnome, KDE, or Unity DE. PyCharm is not available for some Linux distributions, such as RHEL6 or CentOS6, that do not include [GLIBC](https://ftp.gnu.org/gnu/libc/) 2.14 or later.   Pre-release versions are not supported | Latest 64-bit version of Windows, macOS, or Linux (for example, Debian, Ubuntu, or RHEL) |

|  |  |  |
| --- | --- | --- |
|  |  |  |

**Supported languages﻿:**

To start developing in Python with PyCharm you need to download and install Python from python.org depending on your platform.

**SUPPORTED PLATFORMS﻿:**

[**Jupyter Notebook**](https://doc.cocalc.com/jupyter.html#id13)**:**

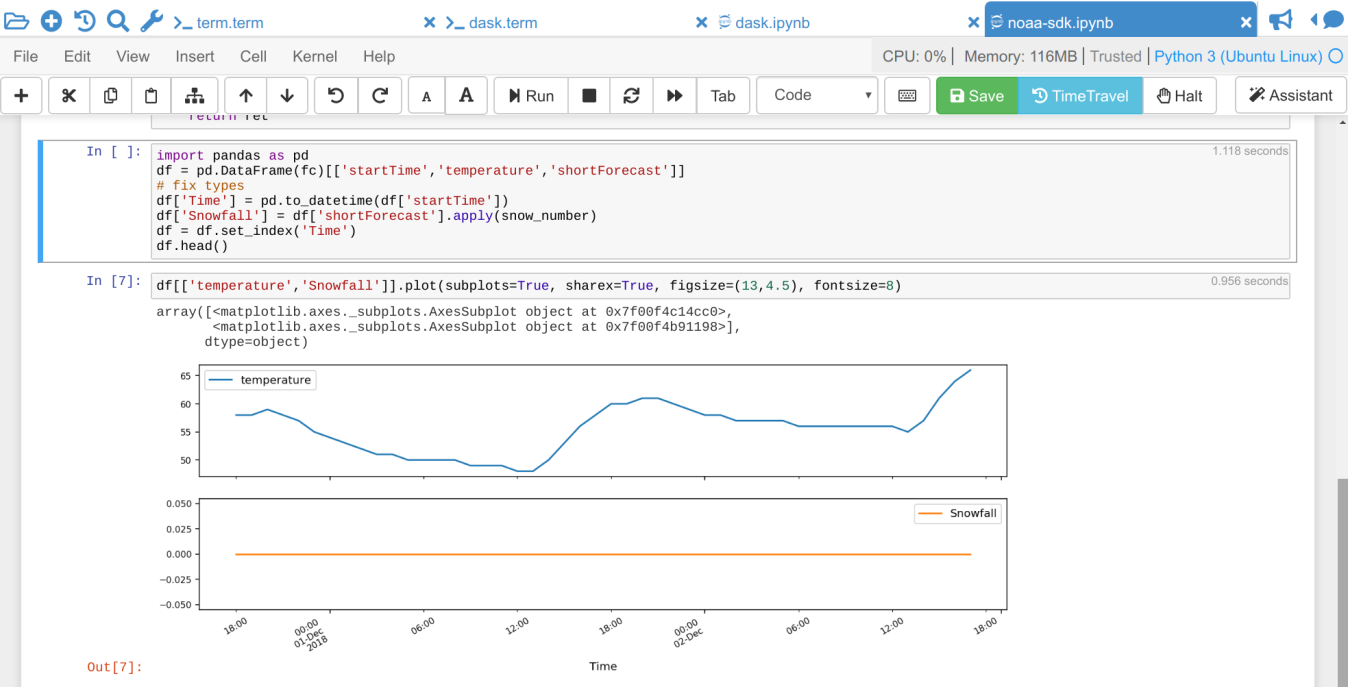
A Jupyter notebook is a specific filetype with the ending .ipynb, which records an interactive session with a **Kernel**. It made up of cells, which can either store one or more lines of code or formatted text. When you run a cell – which evaluates the piece of code in the cell via the active kernel session – you can see its output after the calculation is done. This combination of communicating back and forth with a kernel and adding descriptive text makes this form of document very attractive.

## [Jupyter Kernels](https://doc.cocalc.com/jupyter.html#id14):

You can choose the programming language and environment by selecting a Jupyter kernel for the notebook. Popular choices are [Python3](https://docs.python.org/3/), [SageMath](https://www.sagemath.org/), and [R](https://www.r-project.org/about.html). There many others. Our page on [Jupyter Kernel Selection](https://doc.cocalc.com/howto/jupyter-kernel-selection.html) shows how to set the kernel.

**JUPYTER NOTEBOOK BASIC:**

By default, a Jupyter notebook on CoCalc has all CoCalc’s core features, including real-time collaboration, side chat, and TimeTravel. Read more in our [blogpost](http://blog.sagemath.com/jupyter/2017/05/05/jupyter-rewrite-for-smc.html). The basic user interface looks like the following:

[](https://doc.cocalc.com/_images/jupyter-notebook-cocalc-1.png)

* Above the main area is a menu bar and a button row:
* The menu bar contains all commands, and in particular the Kernel menu is for changing it if necessary.
* The button row gives you a one-click access to Run the current cell (otherwise press your Shift+Return keys), a way to restart the kernel (which clears the current session) and a Save button to make sure CoCalc has stored the file. The Time Travel button allows you to see previous versions of that notebook, such that you can go back in time to recover from a bad change
* Active cell: in the screenshot above, the blue bar on the left and a blue border around a cell indicates that this is the currently active one. Actions like Run, Delete Cell, etc. operate on the currently selected cell. It is also possible to select more than one cell.
* Execution counter: On the left of each cell, there is an execution counter The number increases each time a cell is being run. After the kernel stopped and restarted, that counter starts again at 1.
* The output of code cells is below the input cell. For example, is the output of cell In the right hand corner of the input cell is some information about how long it took to calculate the result.
* Text cells are slightly different. Select “Markdown” in the dropdown menu in the button bar to change a code cell to such a markdown text cell. There, you can use Markdown to format the text. Similar to code-cells, either Run these text cells to see the processed Markdown code or press Shift+Return. To edit a text cell, either double click it or press your Return key.
* Saving: more general, the nice things about Jupyter Notebooks is that they save all your intput and output in one single file. This means you can download or publish the notebook as it is, and everyone else sees it in exactly the same way

**ANACONDA PYTHON:**

Anaconda® is a package manager, an environment manager, a Python/R data science distribution, and a collection of [over 7,500+ open-source packages](https://docs.anaconda.com/anaconda/packages/pkg-docs/). Anaconda is free and easy to install, and it offers [free community support](https://groups.google.com/a/anaconda.com/forum/?fromgroups#!forum/anaconda).

Get the Anaconda Cheat Sheet and then [download Anaconda](https://www.anaconda.com/downloads).

Want to install conda and use conda to install just the packages you need? Get [Miniconda](http://conda.pydata.org/miniconda.html).

**Anaconda Navigator or conda?**

After you install Anaconda or Miniconda, if you prefer a desktop graphical user interface (GUI) then use [Navigator](https://docs.anaconda.com/anaconda/navigator/). If you prefer to use Anaconda prompt (or terminal on Linux or macOS), then use that and conda. You can also switch between them.

You can install, remove, or update any Anaconda package with a few clicks in Navigator, or with a single conda command in Anaconda Prompt (terminal on Linux or macOS).

* **To try Navigator**, after installing Anaconda, click the Navigator icon on your operating system’s program menu, or in Anaconda prompt (or terminal on Linux or macOS), run the command anaconda navicator
* **To try conda**, after installing Anaconda or Miniconda, take the [20-minute conda test drive](https://conda.io/projects/conda/en/latest/user-guide/getting-started.html) and download a [conda cheat sheet](https://docs.conda.io/projects/conda/en/latest/user-guide/cheatsheet.html)

**Packages available in Anaconda:**

* Over 7,500 additional open-source packages (including R) can be individually installed from the Anaconda repository with the conda install command.
* Over [250 packages](https://docs.anaconda.com/anaconda/packages/pkg-docs/) are automatically installed with Anaconda
* Thousands of other packages are available from [Anaconda.org](https://anaconda.org/).
* You can download other packages using the pip install command that is installed with Anaconda. [Pip packages](https://conda.io/projects/conda/en/latest/user-guide/tasks/manage-pkgs.html#installing-non-conda-packages) provide many of the features of conda packages and in some cases they can work together. However, the preference should be to install the conda package if it is available.
* You can also make your own [custom packages](https://conda.io/projects/conda-build/en/latest/) using the conda build command, and you can share them with others by uploading them to [Anaconda.org](http://anaconda.org/), PyPI, or other repositories.

**Previous versions:**

* Previous versions of Anaconda are available in the [archive](https://repo.anaconda.com/archive/). For a list of packages included in each previous version, see [Old package lists](https://docs.anaconda.com/anaconda/packages/oldpkglists/).
* Anaconda2 includes Python 2.7 and Anaconda3 includes Python 3.7. However, it does not matter which one you download, because you can create new environments that include any version of Python packaged with conda. See [Managing Python with conda](https://conda.io/projects/conda/en/latest/user-guide/tasks/manage-python.html).

**CHAPTER-5**

**SYSTEM TESTING AND MAINTENANCE:**

Testing is vital to the success of the system. System testing makes a logical assumption that if all parts of the system are correct, the goal will be successfully achieved. In the testing process we test the actual system in an organization and gather errors from the new system operates in full efficiency as stated. System testing is the stage of implementation, which is aimed to ensuring that the system works accurately and efficiently. In the testing process we test the actual system in an organization and gather errors from the new system and take initiatives to correct the same. All the front-end and back-end connectivity are tested to be sure that the new system operates in full efficiency as stated. System testing is the stage of implementation, which is aimed at ensuring that the system works accurately and efficiently. The main objective of testing is to uncover errors from the system. For the uncovering process we have to give proper input data to the system. So we should have more conscious to give input data. It is important to give correct inputs to efficient testing. After testing all the modules, the modules are integrated and testing of the final system is done with the test data, specially designed to show that the system will operate successfully in all its aspects conditions. Thus the system testing is a confirmation that all is correct and an opportunity to show the user that the system works. Inadequate testing or non-testing leads to errors that may appear few months later.

This will create two problems, Time delay between the cause and appearance of the problem. The effect of the system errors on files and records within the system. The purpose of the system testing is to consider all the likely variations to which it will be suggested and push the system to its limits. The testing process focuses on logical intervals of the software ensuring that all the statements have been tested and on the function intervals (i.e.,) conducting tests to uncover errors and ensure that defined inputs will produce actual results that agree with the required results. Testing has to be done using the two common steps Unit testing and Integration testing. In the project system testing is made as follows: The procedure level testing is made first. By giving improper inputs, the errors occurred are noted and eliminated. This is the final step in system life cycle. Here we implement the tested error-free system into real-life environment and make necessary changes, which runs in an online fashion. Here system maintenance is done every months or year based on company policies, and is checked for errors like runtime errors, long run errors and other maintenances like table verification and reports.Integration Testing is a level of software testing where individual units are combined and tested as a group.The purpose of this level is to expose faults in the interaction between integrated units. Test drivers and test stubs are used to assist in Integration testing

**Unit Testing:**

Unit testing is a foundational practice in software development where individual units of code, such as functions or methods, are systematically tested in isolation to ensure their correctness. During unit testing, developers write test cases that evaluate specific functionalities of a unit, providing known inputs and validating expected outputs. The primary goal is to verify that each unit behaves as intended and to catch any defects early in the development process. Unit tests are typically automated, allowing for frequent and rapid execution, which aids in identifying regressions quickly. This practice fosters code reliability, facilitates easier maintenance, and contributes to overall code quality. By embracing unit testing, developers can confidently make changes to the codebase,

**Integration Testing:**

Integration testing is a vital phase in software testing that focuses on verifying the collaboration and interaction between different modules or components within a software system. Unlike unit testing, which evaluates individual units in isolation, integration testing assesses the combined functionality of these units when integrated. The goal is to uncover defects arising from the interfaces and interactions between components, ensuring that they work seamlessly together. Integration tests identify issues such as incorrect data flow, communication problems, or inconsistencies that may arise during the integration of various modules. This testing phase plays a crucial role in confirming that the software system as a whole meets its functional and performance requirements, providing a comprehensive. Successful integration testing contributes to a more reliable and stable software product, reducing the risk of defects that might emerge when individual components are combined.

**Functional Testing:**

Functional testing is a key aspect of software testing that focuses on verifying the correctness and functionality of a software application based on its specified requirements. This testing phase assesses whether the software system performs its intended operations accurately, adheres to functional specifications, and meets the expectations outlined in the project documentation. Functional testing covers various aspects such as user interface interactions, data manipulation, and overall system functionality. It involves executing test cases that simulate real-world scenarios, inputting valid and invalid data, and validating the system's responses. Functional testing ensures that each feature, from basic functionalities to complex business processes, operates as expected, providing a comprehensive validation of the software's functional capabilities. Successful functional testing not only confirms the reliability of the software but also helps identify and address any deviations from the desired behavior, contributing to the overall quality and usability of the application.

**Performance Testing:**

Performance testing is a crucial aspect of software testing aimed at evaluating the speed, responsiveness. This type of testing assesses how well the system performs in terms of speed, scalability, and stability, especially when subjected to different levels of load and stress. Performance testing includes various subtypes such as load testing, stress testing, and scalability testing. The primary goal of performance testing is to identify and eliminate bottlenecks, ensuring that the software can handle the expected workload without degradation in performance. This testing phase is instrumental in guaranteeing a positive user experience, minimizing downtimes, and optimizing the software's efficiency under varying conditions.

**Security Testing:**

Security testing is a critical component of software testing that focuses on identifying vulnerabilities and weaknesses within a software application's security infrastructure. The primary goal is to assess the system's resilience against potential threats and unauthorized access, ensuring the confidentiality, authorization controls, data encryption, and protection against common security threats such as SQL injection, cross-site scripting, and other forms of malicious attacks. By simulating real-world cyber threats, security testing helps in fortifying the software against potential breaches and exploits. Successful security testing not only safeguards sensitive information but also fosters user trust, compliance with regulatory standards, and overall resilience against evolving cybersecurity risks.

**Usability Testing:**

Usability testing is a crucial aspect of software testing that focuses on evaluating the user-friendliness and overall user experience of a software application. The primary goal is to assess how easily and effectively users can interact with the system to accomplish their tasks. Usability testing typically involves real end-users performing specific tasks within the application while providing feedback on their experience. Key aspects evaluated include the clarity of user interfaces, intuitiveness of navigation, efficiency in completing tasks, and overall satisfaction. This testing phase helps identify potential user interface issues, areas for improvement in design, and any obstacles that might hinder the user's ability to use the software effectively. Successful usability testing contributes to enhanced user satisfaction, increased productivity, and a positive perception of the software, ultimately leading to a more successful and user-friendly product.

**Compatibility Testing:**

Compatibility testing is a crucial aspect of software testing that focuses on ensuring that a software application functions correctly across different environments, configurations, and devices. The primary goal is to verify that the application is compatible with various operating systems, browsers, network environments, and hardware configurations that users may employ. Compatibility testing aims to identify any issues related to functionality, performance, or display that may arise when the software is accessed on different platforms. This testing phase is particularly important in the era of diverse devices and technologies to guarantee a consistent user experience across a wide range of setups. By conducting compatibility testing, organizations can ensure that their software is accessible to a broader audience and performs reliably in diverse contexts, contributing to customer satisfaction and reducing the risk of post-release issues.

**Regression Testing:**

Regression testing is a vital software testing process that involves re-running previously executed test cases to ensure that recent changes or enhancements in the software codebase have not negatively impacted existing functionalities. The primary goal of regression testing is to detect and prevent the introduction of new defects or bugs due to code modifications. Whenever there are updates, bug fixes, or new features added to the software, regression testing is performed to confirm that the alterations do not unintentionally disrupt the previously tested aspects of the application. Automated testing tools are often employed to streamline the regression testing process, allowing for quick and frequent testing cycles, especially in agile development environments. Regression testing plays a crucial role in maintaining the reliability and integrity of the software throughout its development life cycle.

**CHAPTER – 6**

**METHODOLOGY:**

1. **User Authentication**:

The application uses Django's built-in authentication system

for user registration, login, and logout. The authenticate, login, and logout

functions from django.contrib.auth are used for user authentication.

1. **OTP Verification:**

The send\_otp function utilizes the smtplib library to send OTPs to

users via email.OTPs are randomly generated using a combination

of letters and digits.

1. **Aadhaar Enrollment:**

The aadhaar view handles the enrollment of Aadhaar numbers using

a Django form (num).Entered Aadhaar numbers are saved in the

number1 model.

1. **Voting System:**

The application allows users to vote for a leader, and the voted data

is stored in the votedata model.

The mine\_block view adds a new block to the blockchain, representing

a vote for a specific leader.

1. **Admin Panel:**

There's an admin panel (admin1 view) for administrative

purposes, with a hardcoded username and password for authentication.

1. **Email Notifications:**

Email notifications are sent for OTPs and secret keys using the

smtplib library.

1. **Security Measures:**

CSRF protection is applied to certain views using the

@csrf\_exempt decorator.

1. **Data Models:**

Django models such as number1, aadhaarnumber, voteringlist,

and votedata are used to define the structure of the database.

**CHAPTER – 7**

**CONCLUSION:**

In conclusion, the developed Django application serves as a comprehensive platform for conducting secure and verifiable online voting. It successfully integrates user registration, authentication, and Aadhaar enrollment, ensuring a reliable and authenticated user base. The OTP verification mechanism adds an extra layer of security to the enrollment process. The innovative use of blockchain technology for storing vote data in the votedata model enhances the transparency and integrity of the voting system. Each vote is represented as a block, creating an immutable and traceable ledger. The application promotes a secure and tamper-resistant environment for recording and verifying election results.The inclusion of an admin panel (admin1 view) allows for administrative control, although further considerations for dynamic admin credentials and enhanced security practices would be beneficial. However, to make the application more robust, it is recommended to implement comprehensive testing methodologies, including unit tests and integration tests, to ensure the correctness and reliability of the code. Additionally, further security audits and considerations for protecting sensitive information would be essential in a real-world scenario. Overall, the application lays a foundation for a secure and technologically advanced online voting system, with potential for expansion and improvement based on evolving requirements and best practices in software development.

**CHAPTER - 8**

**REFERENCE:**

1. Online Voting System (Year: 2022)

Authors: Ketan, Shanu Khare, Maninder Kaur

1. Smart Online Voting System (Year: 2021)

Authors: S Ganesh Prabhu, A Nizarahammed

1. Online voting system for India based on AADHAAR ID (Year: 2013)

Authors: Himanshu Agarwal, G. N. Pandey

1. Implementation of authenticated and secure online voting system (Year: 2013)

Authors: Srivatsan Sridharan

1. Online Voting System Based on IoT and Ethereum Blockchain (Year: 2021)

Authors: Awsan A. H. Othman, Emarn A. A. Muhammed, Haneen K. M. Mujahid, Hamzah

**CODEING**

from django.shortcuts import render,redirect,get\_object\_or\_404

from .forms import num,reg\_page,Voteing

from django.contrib.auth import authenticate,login,logout

from django.http import HttpResponse,JsonResponse

from .models import number1,aadhaarnumber,voteringlist,votedata

from time import sleep

import smtplib

from email.mime.text import MIMEText

from email.mime.multipart import MIMEMultipart

import random

import string

from django.views.decorators.csrf import csrf\_exempt

from django.db.models import Sum

from django.contrib import messages

# Create your views here.

def home(request):

return render(request,'web/index.html')

def register(request):

reg=reg\_page()

# print(reg\_page)

if request.method=='POST':

reg=reg\_page(request.POST)

if reg.is\_valid():

reg.save()

print("Register Successfully !!!")

return redirect('login')

return render(request,'web/reg.html',{'form':reg\_page})

def login\_page(request):

if request.user.is\_authenticated:

return redirect('login')

else:

if request.method=="POST":

name=request.POST.get('username')

pwd=request.POST.get('password')

user=authenticate(request,username=name,password=pwd)

if user is not None:

login(request,user)

return redirect('aadhaar')

else:

message="Username are Password incorrect"

print(message)

return render(request,'web/login.html',{'message':message})

return render(request,'web/login.html')

def logout\_page(request):

if request.user.is\_authenticated:

logout(request)

print("Logout")

return redirect('home')

def send\_otp(recipient\_email):

sender\_email = 'hariviki7895@gmail.com'

sender\_password = 'kmvw rwph njsf amtu'

recipient\_email = 'matlabfabhost2023@gmail.com'

otp = ''.join(random.choices(string.ascii\_letters + string.digits, k=6))

msg = MIMEMultipart()

msg['From'] = sender\_email

msg['To'] = recipient\_email

msg['Subject'] = 'OTP for Verification'

body = f"Your OTP is: {otp}. Please use this code to verify your identity."

msg.attach(MIMEText(body, 'plain'))

server = smtplib.SMTP('smtp.gmail.com', 587)

server.starttls()

server.login(sender\_email, sender\_password)

text = msg.as\_string()

server.sendmail(sender\_email, recipient\_email, text)

server.quit()

return otp

def aadhaar(request):

if request.user.is\_authenticated:

print("Welcome")

aadhaar\_form = num()

entered\_aadhaar = None

message = None

if request.method == "POST":

aadhaar\_form = num(request.POST)

if aadhaar\_form.is\_valid():

aadhaar\_instance = aadhaar\_form.save(commit=False)

aadhaar\_instance.save()

message = 'Aadhaar number successfully enrolled.'

# entered\_aadhaar = aadhaar\_form.cleaned\_data['aadhaar']

# if number1.objects.filter(aadhaar=entered\_aadhaar).exists():

# message = 'Aadhaar number already enrolled.'

# otp = send\_otp(request.user.email)

# request.session['otp'] = otp

# request.session['aadhaar'] = entered\_aadhaar

# return redirect('aadhaar1')

return render(request, 'web/aadhaar\_next.html', {'message': message})

else:

# otp = send\_otp(request.user.email)

# request.session['otp'] = otp

# request.session['aadhaar'] = entered\_aadhaar

# return redirect('aadhaar1')

aadhaar\_instance = aadhaar\_form.save(commit=False)

aadhaar\_instance.save()

message = 'Aadhaar number successfully enrolled.'

return render(request, 'web/aadhaar\_next.html', {'message': message})

return render(request, 'web/aadhaar.html', {'data': aadhaar\_form, 'number': entered\_aadhaar, 'message': message})

else:

print("Please Login")

message='Login User Account'

return render(request, 'web/index.html', {'message': message})

@csrf\_exempt

def aadhaar\_next(request):

if request.method == "POST":

aadhaar = request.POST.get('number')

# print(aadhaar)

data = number1.objects.filter(aadhaar=aadhaar)

print(data)

if data:

num\_queryset = number1.objects.filter(aadhaar=aadhaar)

# print(num\_queryset)

if num\_queryset:

num = num\_queryset.first()

# print(num)

num\_list = list(num\_queryset.values())

request.session['aadhaar'] = num\_list

otp = send\_otp(request.user.email)

request.session['otp'] = otp

print("Hello")

return render(request, 'web/aadhaar\_next.html', {'data': data, 'num': num})

else:

message = 'Aadhaar number not found in aadhaarnumber.'

return render(request, 'web/aadhaar\_next.html', {'message': message})

else:

message = 'Aadhaar number not found in Database.'

return render(request, 'web/aadhaar\_next.html', {'message': message})

else:

return render(request, 'web/aadhaar\_next.html')

@csrf\_exempt

def verify\_otp(request):

if request.method == "POST":

entered\_otp = request.POST.get('otp')

stored\_otp = request.session.get('otp')

aadhaar = request.session.get('aadhaar')

print("Entered OTP:", entered\_otp)

print("Stored OTP:", stored\_otp)

print("Aadhaar from session:", aadhaar)

aadhaar\_number = aadhaar[0]['aadhaar']

print("Aadhaar Number:", aadhaar\_number)

user\_aadhaar = number1.objects.get(aadhaar=aadhaar\_number)

print(user\_aadhaar)

if entered\_otp == stored\_otp:

aadhaarnumber.objects.create(useraadhaar=user\_aadhaar, otp=stored\_otp)

# Clear session variables

# del request.session['otp']

# del request.session['aadhaar']

return redirect('voters')

else:

message = 'Invalid OTP. Please try again.'

print(message)

return render(request, 'web/aadhaar\_next.html', {'message': message})

return render(request, 'web/aadhaar\_next.html')

def voters(request):

print("Election team list")

if request.user.is\_authenticated:

voters\_list = voteringlist.objects.all()

return render(request, 'web/voters.html', {'voters\_list': voters\_list})

else:

message = "Login to Your Account"

print(message)

return render(request, 'web/index.html', {'message': message})

def voterlist(request):

if request.method == 'POST':

form = Voteing(request.POST, request.FILES)

if form.is\_valid():

form.save()

return redirect('voters')

else:

form = Voteing()

return render(request, 'web/voterlist.html', {'form': form})

**SCREENSHOTS**

