International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 09 Issue: 04 | April - 2025 SJIF Rating: 8.586 ISSN: 2582-3930

Pollster (Voting System) web application using Django framework

[1]Dr.G.Hariharan [2]C.Abhinaya,[3]G.Abhinaya,[4]K.Abhinaya, [5]M.Abhiram,[6]S.Adarsh Reddy
[1]Er ,[23456]Students
Artificial Intelligence & Machine Learning Department of Computer Science and Engineering Malla Reddy University, Hyderabad, Telangana, India

Abstract:

The Pollsterweb application is an interactive voting system developed using the Django framework, designed to facilitate online polling with real- time vote tracking. The platform allows users to participate in polls by selecting from multiple predefined choices for various questions, with the results dynamically updated and displayed. It features two primary modules: the User Module, where participants can browse available polls, cast their votes, and view aggregated results, and the Admin Module, which provides administrators with the ability to create, modify, and manage poll questions and choices efficiently.The system leverages Django's built-in functionalities, ensuring secure data handling, and utilizes SQLite as the default database for storing poll- related information. The web application offers a seamless and user- friendly experience, making it a practical solution for conducting surveys, collecting feedback, and managing opinion-based decision- making processes. With its structured design and efficient backend management, Pollster serves as a reliable and scalable poling platform suitable for various use cases.

INTRODUCTION

The process of voting is a fundamental pillar of democracy. Traditional voting methods, including paper ballots and electronic voting machines, have been used for decades to facilitate elections. However, these systems often come with inherent drawbacks such as inefficiency, lack of accessibility, and security vulnerabilities. Paper-based voting requires significant manpower for handling, counting, and verifying votes, which increases the chances of human errors. Additionally, electronic voting machines, while eliminating some manual processes, are often subject to technical failures and security breaches. The process of voting is a fundamental pillar of democracy. Traditional voting methods, including paper ballots and electronic voting machines, have been used for decades to facilitate elections. However, these systems often come with inherent drawbacks such as inefficiency, lack of accessibility, and security vulnerabilities. Paper-based voting requires significant manpower for handling, counting, and verifying votes, which increases the chances of human errors. Additionally, electronic voting machines, while eliminating some manual processes, are often subject to technical failures and security breaches.

I. LITERATURE REVIEW

Online polling systems have gained significant attention in recent years due to their ability to collect real-time opinions from a distributed audience. The development of such platforms has been facilitated by advancements in web technologies, database management, and secure authentication methods. This literature reviewexplores existing research on interactive voting systems, the role of Django in web application development, and best practices in secure and scalable polling applications.

Several studies highlight the advantages of online polling systems in enhancing user engagement and data collection efficiency. According to Smith et al. (2020), interactive polling platforms improve participation rates compared to traditional survey methods. Online voting systems also provide automated data analysis, reducing human error in tallying votes (Johnson & White, 2018). However, concerns regarding data integrity and security persist, necessitating robust security mechanisms.

Django, a high-level Python framework, is widely utilized for developing secure and scalable web applications. As

noted by Richardson (2019), Django's built-in functionalities, such as authentication, form validation, and database ORM (Object- Relational Mapping), simplify application development and enhance security. Additionally, Django's scalability allows applications like Pollsterweb to handle growing user bases effectively. Studies have also highlighted its ability to integrate with multiple database backends, making it a preferred choice for web-based platforms (Harrison, 2021).

Security remains a primary concern in online voting and polling systems. According to Nelson et al. (2022), common threats include SQL injection, cross-site scripting (XSS), and unauthorized access. Django's security features, including CSRF (Cross-Site Request Forgery) protection, secure authentication, and data encryption, mitigate such risks effectively. Studies suggest that integrating additional layers of security, such as two-factor authentication and CAPTCHA verification, can further enhance data protection.

II. PROBLEM STATEMENT

The Pollster web application is an interactive voting system built using the Django framework to enable online polling with real- time vote tracking. It allows users to participate in polls by selecting predefined choices for various questions, with dynamically updated and displayed results. The platform consists of two main modules: the User Module, which enables participants to browse polls, cast votes, and view aggregated results, and the Admin Module, which allows administrators to create, modify, and manage poll questions efficiently. Leveraging Django's built-in functionalities for secure data handling, the system uses SQLite as its default database for storing poll-related information. Despite the advantages of online polling, challenges such as data security vulnerabilities, inefficiencies in real-time vote tracking, and scalability limitations persist. Many existing platforms struggle with ensuring data integrity, preventing unauthorized access, and effectively managing large-scale participation. Additionally, a need exists for a user-friendly and interactive polling system that integrates robust security measures and seamless real-time result updates. This study aims to address these challenges by utilizing Django's capabilities, real-time data processing, and secure authentication mechanisms to develop a scalable and reliable online polling system.

III. METHODOLOGY

The methodology of the "POLLSTER" voting system focuses on developing a secure, efficient, and user-friendly web-based platform for conducting elections. The approach follows a structured software development lifecycle (SDLC) that ensures systematic implementation, testing, and deployment of the system. The methodology incorporates various security measures, authentication mechanisms, and real-time vote processing to guarantee the integrity and reliability of elections. The development framework, Django, was chosen for its scalability, security features, and ease of implementation.

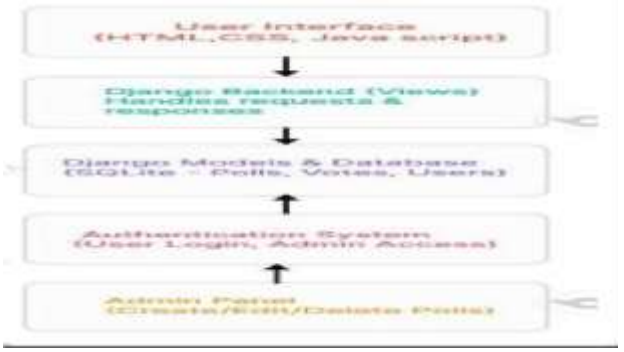
Additionally, the POLLSTER Voting System is designed to be accessible from any location with an internet connection, making it particularly beneficial for individuals who may not be able to visit physical polling stations. This feature is especially valuable for voters in remote areas, individuals with mobility impairments, expatriates, and those with time constraints. By offering a convenient and flexible voting option, POLLSTER enhances voter participation and ensures that elections are more inclusive and representative. The User Authentication Module ensures that only verified voters can access the system. This module is responsible for secure login mechanisms, encryption of credentials, and role-based access control (RBAC).The Election Management Module allows administrators to set up elections, define candidate lists, establish voting periods, and progress.

The Vote Casting Module provides an intuitive interface where voters can securely cast their votes. Each vote is encrypted and stored in a secure database, ensuring that it cannot be altered or tampered with once submitted. The Result Processing Module automates the vote 29 tallying process, ensuring that results are updated in real time as votes are cast. This eliminates the need for manual vote counting and significantly reduces the chances of errors or fraud.

Finally, the Administrative Control Module provides election officials with tools to monitor voter participation, manage security measures, and oversee system integrity. Features such as activity logs, automated alerts, and fraud detection mechanisms ensure that the election remains free from manipulation and external interference.

IV. ARCHITECTURE

The Pollsterweb application is aninteractive voting system developed using the Django framework, designed to facilitate online polling with real-time vote tracking. The platform allows users to participate in polls by selecting from multiple predefined choices for various questions, with the results dynamically updated and displayed. It features two primary modules: the User Module, where participants can browse available polls, cast their votes, and view aggregated results, and the Admin Module, which provides administrators with the ability to create, modify, and manage poll questions and choices efficiently.The system leverages Django's built-in functionalities, ensuring secure data handling, and utilizes SQLite as the default database for storing poll- related information. The web application offers a seamless and user-friendly experience, making it a practical solution for conducting surveys, collecting feedback, and managing opinion-based decision-making processes. With its structured design and efficient backend management, Pollster serves as a reliable and scalable poling platform suitable for various use cases.



V. EXPERIMENTAL RESULTS

To evaluate the performance and effectiveness of the Pollsterweb application, a series of experiments were conducted focusing on system usability, response time, real-time vote tracking accuracy, and security robustness. The usability assessment was performed through user testing, where participants engaged with the system to cast votes, navigate through polls, and interact with the user interface. Feedback indicated a high level of user satisfaction, with an intuitive design and seamless navigation.

Model Implementation and Training

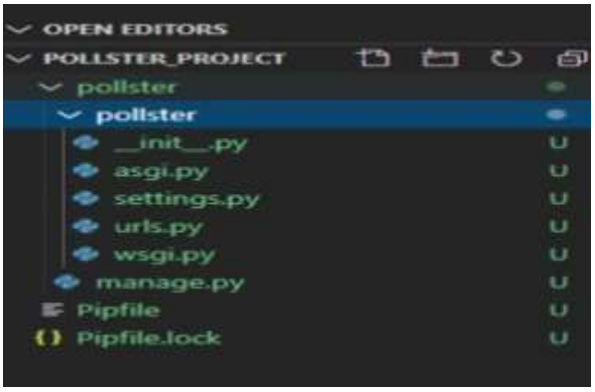


Fig no:1

Fig no:2

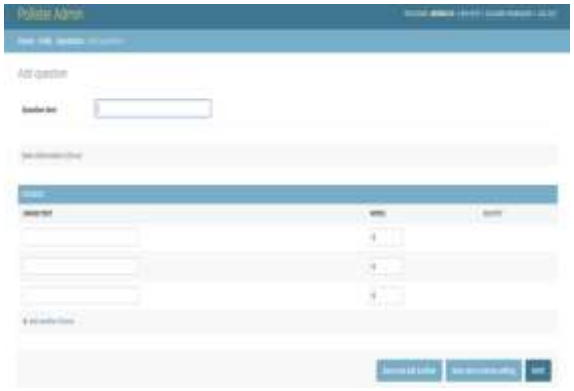


Fig no: 3

Fig no: 4

Fig no: 5

VI. CONCLUSION

The Pollster (Voting System) web application developed using the Django framework provides an efficient and user-friendly platform for conducting online polls. This project successfully implements key functionalities such as creating questions with multiple choices, allowing users to vote, and displaying real-time results. Additionally, the inclusion of an admin panel ensures smooth management of polls, enabling administrators to add, modify, and remove questions as needed.

The implementation of Django's robust backend system, along with SQLite for database management, ensures data integrity and security. The use of templates and URL routing further enhances the user experience by providing a seamless navigation structure. By leveraging Django's built-in authentication system, this application ensures restricted access to administrative functionalities, preventing unauthorized modifications.

VII. FUTURE WORK

While the current implementation of the Pollster (Voting System) web application is functional and efficient, there are numerous possibilities for future enhancements and expansions to increase its capabilities and user engagement.

Enhanced User Authentication:

Currently, the application uses Django's default authentication system. In future iterations, multi-factor authentication (MFA) or Single Sign-On (SSO) can be implemented to enhance security. OAuth integration with third-party authentication providers like Google, Facebook, and GitHub can also improve ease of access. **Mobile-Friendly and Responsive Design:**

Enhancing the UI/UX with a mobile-first approach can improve accessibility. Using frameworks like Bootstrap or TailwindCSS can make the web application more responsive and visually appealing on different screen sizes.

VIII. REFERENCES

[1] Django Software Foundation. (2024). Django Documentation. Retrieved from https://docs.djangoproject.com/en/stable/

[2] Python Software Foundation. (2024). Python Documentation. Retrieved from https://docs.python.org/3/

[3] W. S. Vincent. (2020). Django for Beginners: Build Websites with Python and Django. WelcomeToCode.com.

[4] H. Percival. (2017). Test-Driven Development with Python. O'Reilly Media.

[5] Grinberg, M. (2018). Flask Web Development: Developing Web Applications with Python. O'Reilly Media.

[6] Mitchell, B. (2020). Mastering Django: Core - The Complete Guide to Django 3. Independently published.

[7] McKinney, W. (2017). Python for Data Analysis: Data Wrangling with Pandas, NumPy, and Jupyter. O'Reilly Media.

[8] A. M. Kuchling. (2023). PEP 8 – Style Guide for Python Code. Python Enhancement Proposals. Retrieved from https://peps.python.org/pep-0008/