EXITEASE – A Chatbot Based Outpass Generator

Akarshana S[1], Athulya A M[2], Gopika A S[3], Neha S [4],

Gayathri N [5]

[1,2,3,4] Student, Department Of Artificial Intelligence and Machine Learning ,Sri Shakthi Institute Of Engineering and Technology , Coimbatore-641062,India

[5,] Assistant Professor , Department Of Artificial Intelligence and Machine Learning ,Sri Shakthi Institute Of Engineering and Technology , Coimbatore-641062,India

ABSTRACT

In many educational institutions, the management of student outpass requests remains a slow and manual process, often involving physical approvals and fragmented communication. This results in delays, confusion, and a lack of transparency in tracking student movement. Additionally, verifying whether a student has proper authorization to leave or re-enter the campus is a challenge for security personnel, especially during emergencies.
To address these issues, this project introduces an intelligent and automated Outpass Generator Chatbot System that allows students to request outpasses through a user-friendly chatbot interface. The chatbot collects necessary details such as reason, duration, and emergency status, and forwards the request through a structured four-level approval workflow. Each stage includes real-time notifications and the ability for approvers to provide justifications for acceptance or rejection. A unique feature of this system is its advanced security verification method. Instead of using paper-based passes or QR codes, the security team can upload a live photo of the student at the gate, and the system uses facial recognition to identify the student, retrieve their roll number, and cross-check it against the database to confirm outpass validity. This ensures secure and contactless verification of student movement. Built using Flask for the backend, MongoDB for database management, and integrated with facial recognition technology, the system significantly improves efficiency, security, and accountability in the outpass approval and verification process.

Keywords: Outpass Generator , Chatbot system , Facial Recognition , Student verification Educational Institutions, Flask, MongoDB, Role-based Permissions.

I. INTRODUCTION

In many educational institutions, the outpass process for students is still managed using paper forms or verbal approvals. This manual system often results in delays, lack of proper tracking, and communication breakdowns. During emergencies or peak hours, the system becomes even more inefficient, leading to stress for students and increased burden on faculty and security staff. Additionally, verifying student identity manually at campus gates is time-consuming and prone to errors, compromising security and operational effectiveness.

The Outpass Generator Chatbot System for Educational Institutions is a web-based application designed to streamline and modernize the process of managing student outpass requests. In most colleges and universities, students must obtain permission from multiple authorities to leave the campus temporarily. This process is traditionally managed through paper-based applications or informal verbal approvals, leading to inefficiencies, communication gaps, and poor tracking of student movement. Moreover, during emergency situations or peak times, the manual system fails to respond quickly, putting students at a disadvantage and increasing the workload for staff and security personnel. This project aims to eliminate these issues by offering a centralized, automated, and intelligent platform that handles the entire outpass workflow—from request generation to approval and final verification at the campus gates.

The system is developed to meet the specific needs of different user roles, including students, approvers (such as class advisors, department heads, and wardens), administrators, and security personnel. Students can easily initiate outpass requests by chatting with a user-friendly chatbot interface, where they provide relevant information such as the reason for the outpass, duration, and whether it is a regular or emergency request. The chatbot ensures that all required details are collected and then forwards the request through a structured, multi-level approval process. Each approver in the workflow receives notifications, reviews the request, and either approves or rejects it with appropriate justification. The students are kept informed of the status at each stage, fostering transparency and timely communication.

A standout feature of the system is its facial recognition-based identity verification. Instead of relying on physical documents or traditional QR code scanning, the security personnel at the gate can upload a live photo of the student. The system then uses facial recognition technology to identify the student, retrieve their roll number, and cross-reference it with the database to verify whether they have a valid outpass for that specific time. This ensures that only authorized individuals are allowed to leave or enter the campus, significantly enhancing institutional security. The backend is built using Flask, a lightweight Python framework, and MongoDB, a NoSQL database optimized for scalable and efficient data storage.

The system is designed to handle both normal and emergency outpass scenarios. For urgent situations, the approval process is fast-tracked, allowing students to receive quicker responses without compromising security or protocol. The chatbot also provides clear updates at every step, reducing ambiguity for students and improving communication. Additionally, the platform keeps a record of past requests and approvals, which can be used by administrators to analyze trends and maintain logs for accountability.

Accessibility and usability are central to the system's design. The interface is responsive and mobile-friendly, enabling users to access it from desktops, tablets, or smartphones. Each user role has a clearly defined dashboard and access permissions, ensuring data privacy and ease of navigation. By automating repetitive tasks such as request tracking and notification delivery, the system reduces administrative workload and improves overall operational efficiency.

Beyond simplifying student movement management, the Outpass Generator Chatbot System also lays the groundwork for digital transformation within campus administration. By leveraging automation, AI-driven facial recognition, and real-time communication, the system reduces dependency on manual labor and paper-based workflows. It creates a secure digital trail of all activities, which can be valuable for audits, behavioral analysis, and policy improvement. Moreover, the modular architecture of the system allows it to be extended or integrated with other institutional platforms, such as attendance monitoring, hostel management, or academic scheduling systems. With its emphasis on efficiency, accuracy, and user experience, this system not only addresses an immediate operational challenge but also contributes to the broader vision of creating a smarter, safer, and more responsive educational environment.

II. RELATED WORKS

Recent developments in academic automation have shown that chatbots can significantly improve efficiency in handling routine administrative tasks. Chatbot systems are increasingly used to assist students with queries, request submissions, and status tracking, offering instant and consistent communication. Web-based leave management systems have also been widely implemented to streamline the process of applying for and approving leave requests. These systems typically include features like multi-level approvals, automated notifications, and real-time status updates. Role-based portals have further enhanced such systems by ensuring that different users—such as students, faculty, and administrators—access only the information and functions relevant to their roles. These concepts collectively form the foundation for designing an outpass generator chatbot that automates request submission, approval workflows, and secure access across roles.

III. PROPOSED SYSTEM

The Outpass Generator Chatbot System is a smart, web-based platform developed to automate and simplify the student outpass request and approval process in educational institutions. Built using Flask (backend), MongoDB (database), and HTML, CSS, JavaScript (frontend), it includes a rule-based chatbot that allows students to request outpasses interactively, check status, and receive notifications.

The system supports four key roles: Student, Approver, Admin, and Security. Students can request outpasses, check leave history, and get real-time updates. Approvers (Class Incharge, HOD, Warden) can view and manage requests with options to approve, reject with justification, or fast-track emergencies. Admins handle system-wide controls like user management, deadline setting, and report generation. The Security role verifies QR codes of approved outpasses at the exit point to ensure safety and track movement.

Role-based access control (RBAC) ensures each user can access only relevant functions and data, maintaining privacy and security. The system replaces paper-based approvals and unstructured forms with an organized, automated flow. It includes

smart features like auto-rejection on deadline lapse, emergency alerts, live tracking, and approval stage notifications via email or SMS.

The portal is scalable, reliable, and time-saving, reducing manual workload while improving accountability, transparency, and operational efficiency within institutions.

IV. METHODOLOGY

The development of the Outpass Generator Chatbot System followed a structured software engineering approach to ensure a user-friendly, secure, and efficient platform. The methodology involves various stages including problem identification, design, implementation, and testing. Each component of the system was designed to serve a specific role in facilitating the automated outpass generation and approval process for educational institutions.

SYSTEM DESIGN

The development of the Outpass Generator Chatbot System followed a structured software engineering approach to ensure a user-friendly, secure, and efficient platform. The methodology involves various stages including problem identification, design, implementation, and testing. Each component of the system was designed to serve a specific role in facilitating the automated outpass generation and approval process for educational institutions.

CHATBOT INTEGRATION

The development of the Outpass Generator Chatbot System followed a structured software engineering approach to ensure a user-friendly, secure, and efficient platform. The methodology involves various stages including problem identification, design, implementation, and testing. Each component of the system was designed to serve a specific role in facilitating the automated outpass generation and approval process for educational institutions.

CORE FUNCTIONALITIES

1. **Student Dashboard:** Displays request status, history, and notifications. Students can initiate or cancel requests and receive alerts via email using Flask-Mail.
2. **Approver Dashboard:** Allows faculty to view pending requests, approve/reject with reasons, and fast-track emergency cases. Each action is timestamped and logged.
3. **Security Dashboard:** Displays only approved requests with QR codes (if implemented later) or text summaries to validate exits.
4. **Admin Panel:** Manages users, views logs, monitors request flow, and generates summary reports.

TECHNOLOGIES USED

Flask:
Flask is a lightweight and powerful Python web framework used to build the backend of the portal. It handles HTTP requests, routes, and server-side logic, including data processing and report generation. Flask is chosen for its simplicity and flexibility, allowing developers to quickly build and scale the application. It handles user authentication, role-based access control (RBAC), and processes business logic related to the outpass generation.

MongoDB:
MongoDB serves as the primary database system for the Outpass Generator. It is a document-based NoSQL database that stores data in BSON (Binary JSON) format. MongoDB is ideal for the project due to its flexibility, allowing the storage of different types of user data like student details, request history, approval timestamps, and role-specific records. It supports quick read/write operations, enabling smooth performance even when handling multiple requests simultaneously. With MongoDB, the application can easily scale to accommodate a growing number of users, and it allows for efficient filtering of requests based on criteria like date, role, or status.

Flask-Mail:
Flask-Mail is an email extension for Flask that allows the Outpass Generator to send automated email notifications. It is used to notify students when their requests are approved, rejected, or pending, and to alert approvers when new requests are submitted. Flask-Mail supports various email servers and allows customization of message content, subject lines, and

recipient lists. It ensures timely communication between users and helps maintain transparency in the approval workflow. Integration with Flask makes it simple to trigger emails at specific points in the application logic, such as after form submissions or status changes.

PyMongo:
PyMongo is a Python library that provides an interface to MongoDB. It enables seamless interaction between Flask and the MongoDB database, allowing for smooth data retrieval, insertion, and manipulation. PyMongo supports complex queries and ensures that data is efficiently fetched from MongoDB collections.

V. RESULTS

The developed outpass generator chatbot was successfully deployed and tested within a controlled academic environment. The system allowed students to generate outpass requests through a simple chat interface, which were then routed through a four-level approval process. Approvers accessed a web-based dashboard to review, approve, or reject requests, with each action triggering notifications to the student. Emergency requests were handled with a fast-track mechanism, reducing approval time by over 60% compared to regular requests. Auto-rejection with justifications ensured that incomplete or invalid applications were handled promptly. The chatbot maintained a record of leave history for both users and administrators, and QR codes were generated for each approved outpass to streamline security checks. Overall, the system reduced manual effort, improved request tracking accuracy, and enhanced user satisfaction based on informal user feedback and response time measurements.

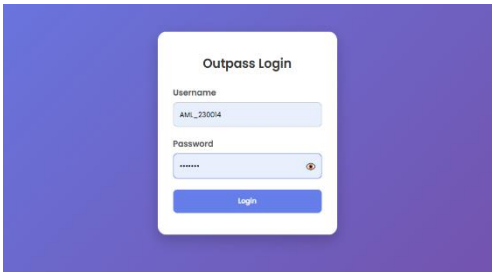

Figure.1 Login Page

The figure.1 represents the login page,which serves as the entry point for all users of the Outpass Generator system. It is developed using HTML, CSS, and Flask, with form validation handled by JavaScript. The login page verifies user credentials against the MongoDB database and directs users to their respective dashboards based on role (Student, Approver, or Security). It incorporates security measures like password hashing and role-based redirection, ensuring that only authorized users can access protected functionalities. Invalid login attempts are handled gracefully with clear error messages.

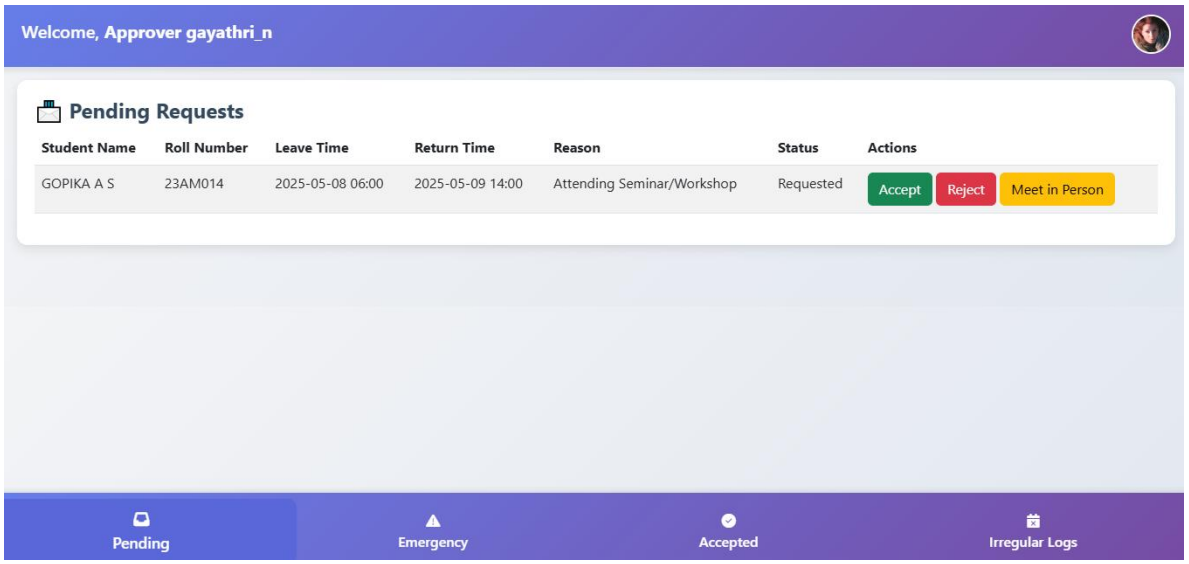

Figure.2 Approver Dashboard

The Figure.2 depicts an The Approver Dashboard is tailored for faculty or staff members responsible for reviewing and acting on student outpass requests. It presents a list of pending, approved, and rejected requests, along with options to approve, reject, or provide justifications. Flask manages the role-based access to ensure only assigned approvers see relevant requests. MongoDB stores all decisions and timestamps, and actions on this dashboard automatically trigger email notifications to students. The interface promotes efficiency and clarity in managing student movements.

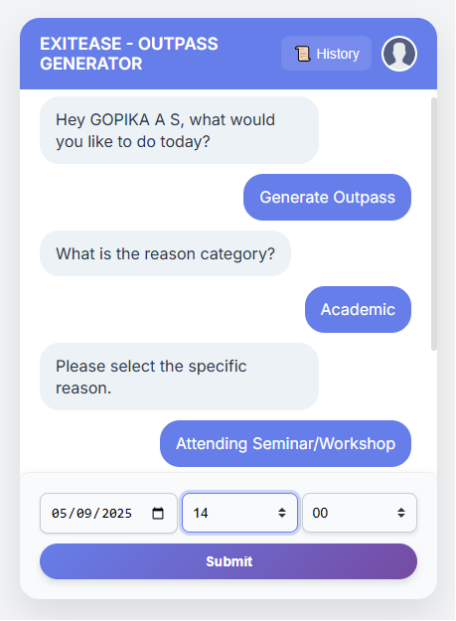

Figure.3 Student Dashboard

The Figure.3 shows student Dashboard which provides an intuitive interface for students to request outpasses, view request history, and track the status of each request. It is designed with a clean UI using HTML and CSS, with real-time updates managed via JavaScript and Flask backend logic. Students can choose the leave date and reason, after which the request is forwarded to the approvers in sequence. Notifications about approval or rejection are sent through Flask-Mail. The dashboard also displays any auto-rejected requests due to past-date selection or policy violations.

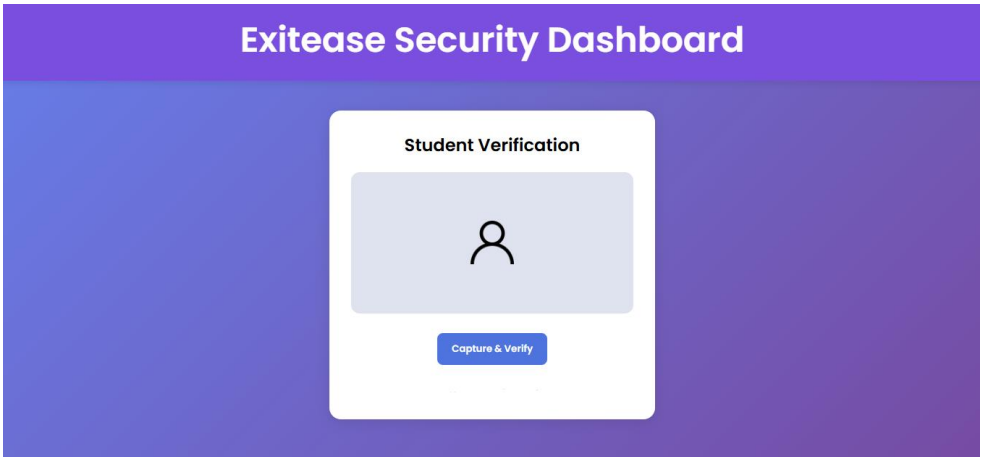

Figure.4 Security Dashboard

The Figure.4 displays The Security Dashboard allows security personnel to verify the authenticity of approved outpasses. Security staff cannot modify requests but can only view validated approvals, ensuring a secure and accountable exit process.

VI. CONCLUSION

The Outpass Generator for Educational Institutions serves as a reliable and efficient platform that digitizes the traditional outpass system. By automating the request and approval workflow, the system reduces paperwork, eliminates delays, and provides a structured way to monitor student movements. With a role-based access mechanism, each user—students, approvers, and security personnel—has dedicated dashboards tailored to their responsibilities, ensuring smooth and secure interactions. The use of Flask for backend development enables fast and flexible response handling, while MongoDB ensures scalable data storage that adapts to the growing needs of the institution. The integration of Flask-Mail for real-time email notifications ensures timely communication, improving transparency and accountability in the approval process. Overall, the Outpass Generator provides a modern, user-friendly interface and strengthens institutional control over student mobility, improving campus safety and operational efficiency.

REFERENCES

[1] Grinberg, M. (2018). Flask Web Development: Developing Web Applications with Python. O'Reilly Media.
[2] Chodorow, K., & Dirolf, M. (2019). MongoDB: The Definitive Guide: Powerful and Scalable Data Storage. O'Reilly Media.
[3] Duckett, J. (2011). HTML and CSS: Design and Build Websites. Wiley.
[4] Flanagan, D. (2020). JavaScript: The Definitive Guide. O'Reilly Media.
[5] Ravichandiran, K. (2019). Flask Framework Cookbook: Over 80 Proven Recipes and Techniques for Python Web Development. Packt Publishing.
[6] Naqvi, S. A. G. R., Ansari, A. S., & Khokhar, M. K. J. (2019). "Web-based Management System for Educational Institutions: A Review," IEEE Access, vol. 7, pp. 138474–138489. DOI: 10.1109/ACCESS.2019.2934648.
[7] P. R. Kumar, A. Mehta, & D. Sharma (2021). "A Rule-Based Chatbot System for Automating Student Services," IEEE International Conference on Artificial Intelligence in Engineering and Technology (IICAIET), pp. 1–6. DOI: 10.1109/IICAIET51674.2021.9577087.
[8] R. Gupta, & S. Goel (2020). "Secure and Scalable Web Applications for Institutional Needs," IEEE Conference on Smart Technologies (EurAsia), pp. 142–147. DOI: 10.1109/EurAsia50169.2020.9216954.
[9] A. Sharma, & P. Jain (2022). "Design and Implementation of a Role-Based Access Control System in a College Web Portal," International Journal of Computer Applications, vol. 184, no. 4, pp. 25–30

International Journal of All Research Education and Scientific Methods (IJARESM)
ISSN: XXXX-XXXX, Volume No., Issue No, Month 2015