DEPARTMENT OF INFORMATION TECHNOLOGY

UNIVERSITY INSTITUTE OF ENGINEERING AND

TECHNOLOGY , PANJAB UNIVERSITY, CHANDIGARH

PROJECT – REPORT

B.E[IT] 3rd YEAR [5th SEM]

UIETmate : SMART SOLUTION FOR STUDENTS

SUBMITTED BY

GOURAV [UE238037] , DEVANSHU SINGH [UE238030]

AYUSH CHAUHAN [UE238025] , ANCHAL KAPOOR [UE238011]

BIBHU RAJ [UEM238117] , ARYAN [UEM238116]

ADITYA YATI [UEM238115]

SUBMITTED TO

DR. SUKHVIR SINGH

ASSISTANT PROFESSSOR

Abstract

UIETmate is a proposed AI-driven application that unifies key campus services into a single assistant. The app integrates five modules – an Attendance Tracker & Predictor, an Exam Preparation Assistant, a Career Mentor, a Smart Notice Board, and a Lost & Found AI – to support student needs. Using Agentic AI and autonomous technique, UIETmate automates attendance logging and predicts future trends, generates personalized study aids and quizzes, provides customized career guidance, delivers real-time announcements, and assists in recovering lost items. We outline the system's architecture and methodology (including flowchart), describe the technology stack, and present a development schedule. The expected outcome is a fully functional prototype that streamlines student workflows and enhances engagement. By consolidating scattered services into one intelligent interface, UIETmate aims to reduce administrative overhead and improve academic outcomes.

Introduction and Problem Statement

Modern university campuses often suffer from fragmented technology and manual processes, forcing students to juggle multiple systems for routine tasks. Despite advances in AI, campus technology remained fragmented. In practice, this fragmentation leads to several specific problems:

- **Manual Attendance:** Tracking and recording class attendance is typically paper-based or inconsistent, making it hard to monitor patterns. Since class attendance is strongly linked to academic success, lack of automation means educators cannot easily identify or predict absenteeism.
- **Exam Preparation:** Students often lack personalized study support. Without adaptive resources, learners cannot efficiently target weak areas or gauge readiness, missing out on the benefits of AI-driven tutoring systems that can tailor content to each student's progress.
- **Career Guidance:** Traditional career counseling is generic and limited. Many students are uncertain about job markets and internships.

Research shows AI can act as an interactive virtual mentor, providing personalized career advice based on student interests and performance. Existing solutions rarely offer real-time, individualized mentorship.

- **Information Dissemination:** Campus notice boards (physical or digital) are static and slow. Important announcements (events, deadlines) often get delayed or overlooked. Current static boards lack this efficiency.
- **Lost & Found Management:** Lost items on campus are handled manually, causing stress and wasted time. owners often give up searching, and lost items are rarely reunited with owners.

These issues motivate **UIETmate**: a unified, intelligent web mobile app that addresses the above problems. By integrating Agentic AI and automated workflows, UIETmate aims to make campus life smoother.

Objectives

- Develop a unified web mobile app that integrates multiple campus services into one platform.
- Implement an **Attendance Tracker & Predictor** to automatically record class attendance and forecast future attendance trends using AI models.
- Design an **Exam Preparation Assistant** that offers personalized study materials, practice quizzes, and progress feedback using RAG and adaptive LANGCHAIN techniques.
- Create a **Career Mentor** module that provides tailored career guidance and connects students to relevant internships and job postings based on their profile.
- Build a **Smart Notice Board** that delivers real-time campus announcements.
- Develop a **Lost & Found AI** system that uses image recognition and database matching to help students report lost items and quickly find corresponding found items.
- Ensure the system is secure, user-friendly, and easily maintainable.

System Overview (Modules)

- **Attendance Tracker & Predictor:** A mobile interface allows students to check in and logs this data to a backend. The module aggregates historical attendance records and applies AI technique to predict future attendance patterns. Students can view current attendance and forecast trends. (Accurate attendance data is vital, as studies show class attendance positively affects exam performance.)
- **Exam Prep Assistant:** This module supports students' study needs. The user uploads PYQs, notes , study material, and the app retrieves relevant resources (notes, past questions). An AI engine (e.g., using RAG , Langchain and a GPT-like model) generates or curates concise summaries and also answer user queries. By emulating intelligent tutoring systems, UIETmate can personalize learning, giving "immediate feedback, guided practice, and adaptivity" to improve outcomes.
- **Career Mentor:** This component serves as a virtual career counselor. It collects the student's academic profile, skills, and interests. Using an AI-driven recommendation engine, it scans a database of internships, jobs, and career paths to suggest personalized opportunities. Additionally, it can answer common career questions in natural language, acting as a virtual coach. By integrating with external resources (e.g., LinkedIn APIs or industry data), it keeps advice timely.
- **Smart Notice Board:** This module replaces static bulletin boards. The app fetches and displays important and necessary notices in real time, possibly with push notifications. UIETmate's notice board will integrate with such devices or cloud services to ensure that the latest updates are always available on student devices.
- **Lost & Found AI:** This module automates lost-item recovery. When a student loses an item, they can submit a description and/or photograph through the app. The system uses computer-vision (e.g., an object recognition model, similar to HuskyLens) to compare the submitted item against a database of found items (entered by staff or detected via an AI

camera at a lost-and-found table). The AI continuously monitors new found-item entries; as soon as a match is detected, both the owner and the finder are notified.

Each module interacts with a central backend. User requests are routed to the corresponding service, which processes data and returns results to the app. Common tasks such as user authentication, data storage, and notifications are managed centrally. This modular architecture ensures that each component can be developed and updated independently, yet work together under the UIETmate framework.

Technologies Stack

The system will integrate a range of modern tools and frameworks to ensure efficient development, robust performance, and scalability:

Front-End

JavaScript – Core scripting language for creating interactive user interfaces.

React Native – Cross-platform framework for building mobile applications for both Android and iOS.

Figma – Collaborative UI/UX design tool for prototyping and wireframing application screens.

Back-End

Node.js – Runtime environment for building scalable, event-driven server applications.

FastAPI – High-performance Python web framework for REST API development.

MongoDB – NoSQL database for flexible and efficient data storage.

Agentic AI

Gemini AI – Advanced AI models for natural language understanding and generation.

RAG (Retrieval-Augmented Generation) – AI technique for generating answers with contextual data retrieval.

LangChain – Framework for building applications powered by large language models with integrated data pipelines.

Others

Python – General-purpose programming language for backend services, AI model training, and automation.

REST API – Standardized protocol for communication between the application's modules.

Firebase – Cloud-based platform for authentication, real-time databases, and push notifications.

Proposed Methodology

The development approach is modular and iterative. Key steps include data preparation, model development, integration, and testing for each module. We describe the overall workflow below, followed by a textual flowchart of the user-system interactions:

- **User Authentication:** Students and admins log in via the UI. The system verifies credentials against the university's identity provider.
- **Module Selection:** Upon login, the user selects one of the five modules in the app interface.
- **Module Processing:** Depending on the choice, the request is routed as follows:
- **Attendance Tracker & Predictor:**
    a. *Data Retrieval:* System fetches the student's attendance records from the database.
    b. *Processing:* Clean and preprocess the data (e.g., missing values).
    c. *Prediction:* Run the trained attendance model to forecast future attendance probability.

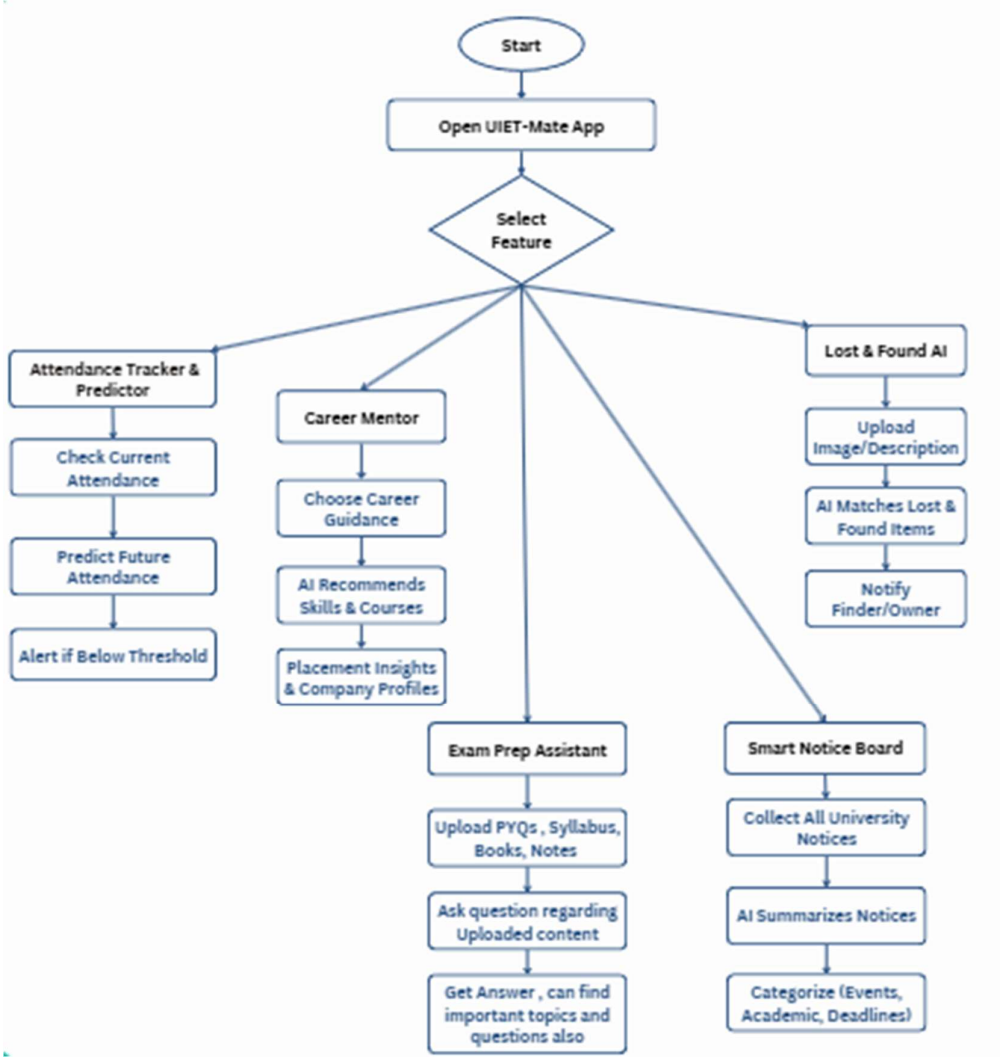
d. *Output:* Display current attendance status and predicted trends on the app dashboard.

- **Exam Prep Assistant:**
    a. *Input:* Student uploads PYQs, Notes , Study Material
    b. *Resource Fetch:* Retrieve relevant course materials and question sets.
    c. *AI Generation:* Use an RAG agent to generate custom practice questions or concise study notes for the topic.
    d. *Interaction:* Provide Summary and answer to user queries using the retrived content and feeding it to the LLM.
- **Career Mentor:**
    a. *Profile Analysis:* Compile the student's academic records and entered skills/interests.
    b. *Information Retrieval:* Query job/internship databases for matches (filter by profile).
    c. *AI Advising:* Use a recommendation algorithm (and optionally a chatbot) to suggest careers, additional skills to acquire, or relevant events.
    d. *Presentation:* Show tailored suggestions and resources, allow follow-up questions from the user.
- **Smart Notice Board:**
    a. *Fetch Notices:* The app polls or is pushed the latest announcements from the admin console.
    b. *Display Content:* Render each notice (text, image, video, or audio) in the app. Provide alerts (e.g., push notifications) for new urgent messages.
- **Lost & Found AI:**
    a. *Report Item:* Student uploads a description and/or photo of a lost item.

b. *AI Matching:* An object recognition model compares the submitted image (or keywords) against the database of found-item images and descriptions.

c. *Outcome:* If a match is found, notify the student (and the finder). Otherwise, log the lost item entry.

- **Central Services:** Tasks like data storage, notification delivery, and logging are handled by centralized backend services, ensuring data consistency across modules.



Work Plan

The project will proceed in phases over an 10-12 week timeline, as outlined below:

• **Week 1–2 (Requirement Analysis):** Performing detailed review and finalize functional requirements for each module.

• **Week 3–4 (Design):** Architect the system (core idea) and designing user interfaces and data schemas.

• **Week 5–6 (Development of Modules):** Collecting placement insights, and saving data in database.

• **Attendance & Exam Modules (Week 5–6):** Build the attendance logging system and predictive model; developing the exam assistant model and pipeline.

• **Career & Notice Modules (Week 5–6):** Create the career advisor backed (profile matching, recommendations) and the notice board interface.

• **Lost & Found Module (Week 6–7):** Integrate image recognition API and database for lost-and-found matching.

• **Week 7-8** (**Development of Modules):** Implement core functionality.

• **Week 9-10 (Integration & Testing):** Integrate all modules into the complete application.

• **Week 10-12 (Evaluation & Documentation)**: Finalize the report, user manual, and presentation materials.

References

- College Project Reports – Sample technology stack and data source structuring ideas from other institutions' published reports.
- OpenAI. Introduction to LangChain for LLM Applications. Retrieved from: https://www.langchain.com
- MongoDB, Inc. MongoDB Documentation. Retrieved from: https://www.mongodb.com/docs/
- FastAPI. FastAPI Official Documentation. Retrieved from: https://fastapi.tiangolo.com

- Google Firebase. Firebase Documentation. Retrieved from: https://firebase.google.com/docs
- Node.js Foundation. Node.js Documentation. Retrieved from: https://nodejs.org/en/docs/
- Gemini AI – Google DeepMind. Gemini Model Overview. Retrieved from: https://deepmind.google/technologies/gemini/
- Wikipedia contributors. “Retrieval-Augmented Generation.” Wikipedia. Retrieved from: https://en.wikipedia.org/wiki/Retrieval-augmented_generation