2. Introduction (About the Project)

a) Define Project Scope

The **Student Management System (Bindo)** is designed to streamline the management of student academic records, particularly focusing on **Class Test (CT) Marks Notification**. The system allows faculty members to enter CT marks, which are securely stored in the database. Students can view their marks through the portal and receive automatic notifications via email and real-time pop-ups. The system will also include role-based access for administrators, faculty, and students to ensure secure and efficient operations.

b) Set Objectives and Goals

The main objectives of this project are:

- To develop a **secure** and **user-friendly** student management system.
- To enable faculty members to **enter and update CT marks** efficiently.
- To ensure students **receive individual CT marks** via **email** and **pop-up notifications**.
- To enhance **data security and privacy** by allowing students to view only their own marks.
- To provide **real-time notifications** for new mark entries and updates.
- To integrate **role-based access control (RBAC)** for system users.
- To generate **academic reports** and **performance analytics** for students and faculty.

3. Defining Requirements

a) Defining All Requirements

The system requires:

- A **database** for storing user information, marks, notifications, and reports.
- **JavaFX UI** for the frontend.
- **JavaMail API** for sending email notifications.
- **Authentication system** for secure access.
- **Automated notification system** for real-time updates.
- **Role-based access control** (Admin, Faculty, Student).

b) Functional Requirements

1. Faculty can **enter, update, and view** CT marks.
2. Students can **view only their own marks**.
3. Marks are **automatically sent via email** to students.
4. A **real-time notification system** alerts students of new marks.
5. Admin can **manage faculty, students, and course assignments**.
6. Reports can be **generated and exported** as PDFs or Excel files.

c) Non-functional Requirements

1. **Security** – Secure login with **two-factor authentication (2FA) for admin and faculty**.
2. **Scalability** – Ability to handle a large number of students and faculty members.
3. **Reliability** – Automated backups for marks and attendance data.
4. **Usability** – Responsive UI optimized for mobile and desktop use.
5. **Performance** – System should handle mark entry and retrieval **within seconds**.

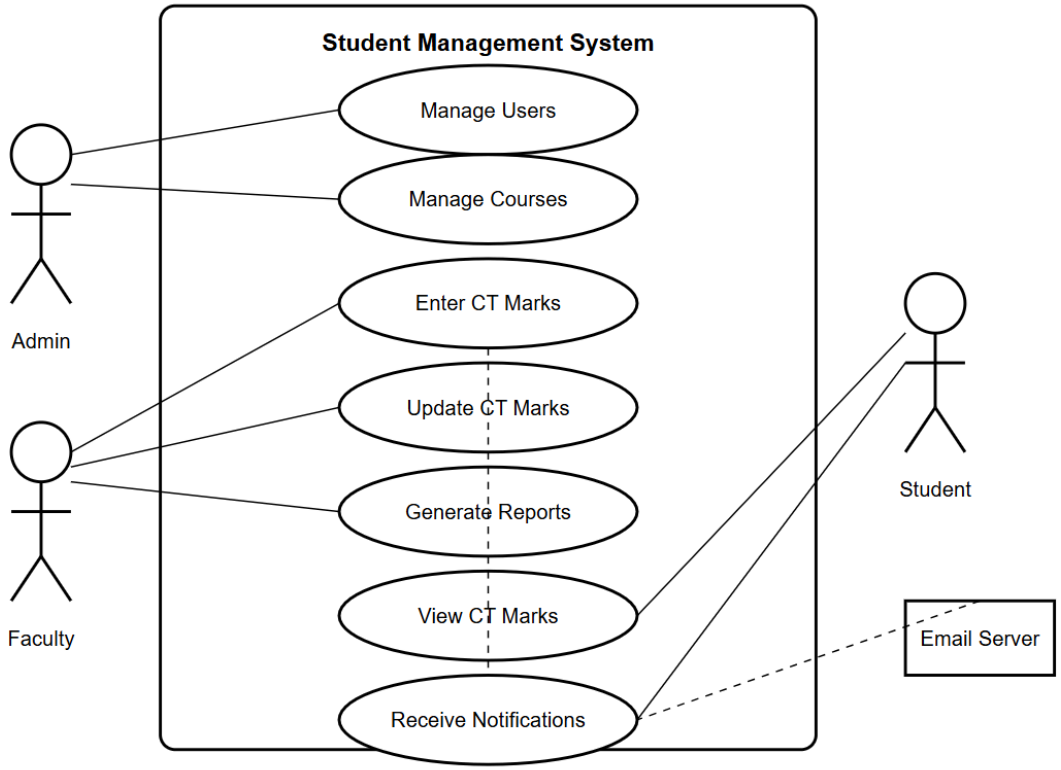
4. Requirement Analysis

a) Scenario-Based Models

i. Use Case Diagram – Including Actors

Actors:

- **Primary Actors:** Faculty, Admin
- **Secondary Actors:** Student, Email Server (for notifications)



ii. Use Case Descriptions – 3 Major Use Cases

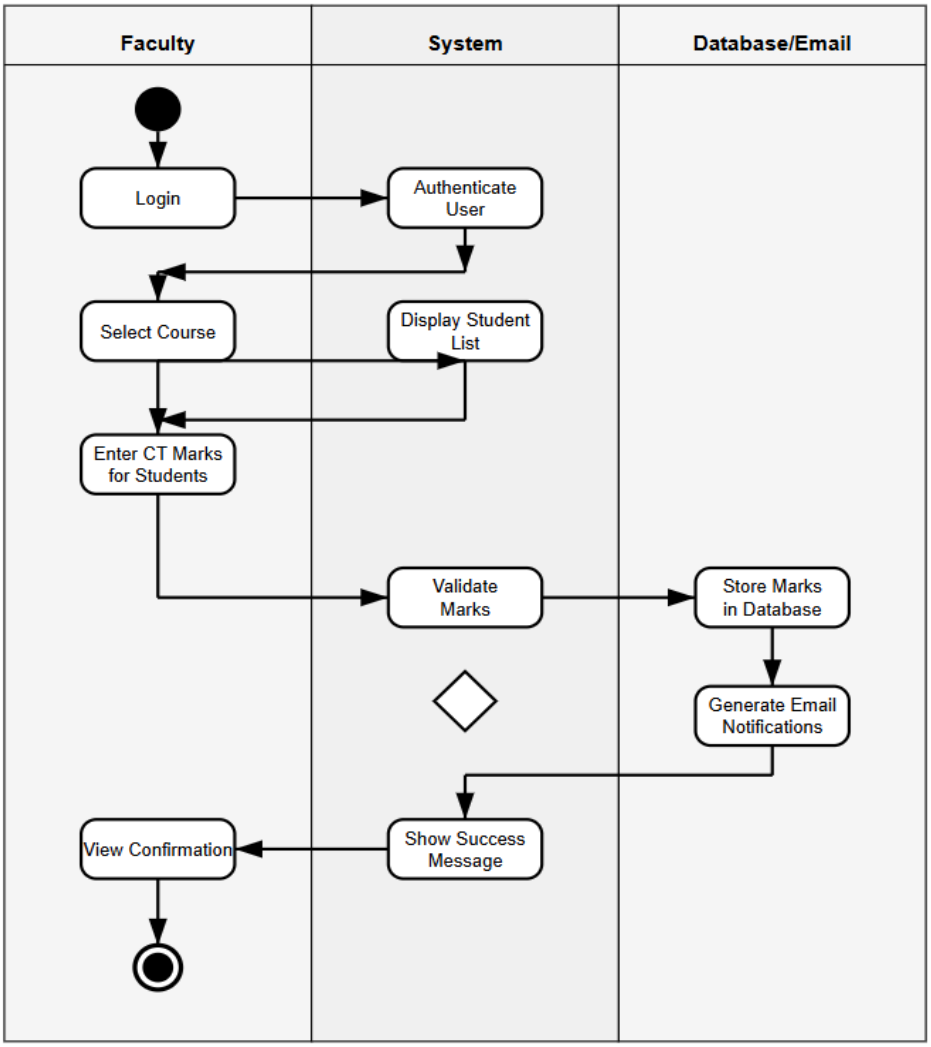
Use Case	Description
Enter CT Marks	Faculty enters CT marks for students, stores them in the database.
Send CT Marks Notification	System sends individual emails to students when CT marks are entered.
View CT Marks	Students log in and see their own marks with pop-up notifications.

b) Behavioral Models

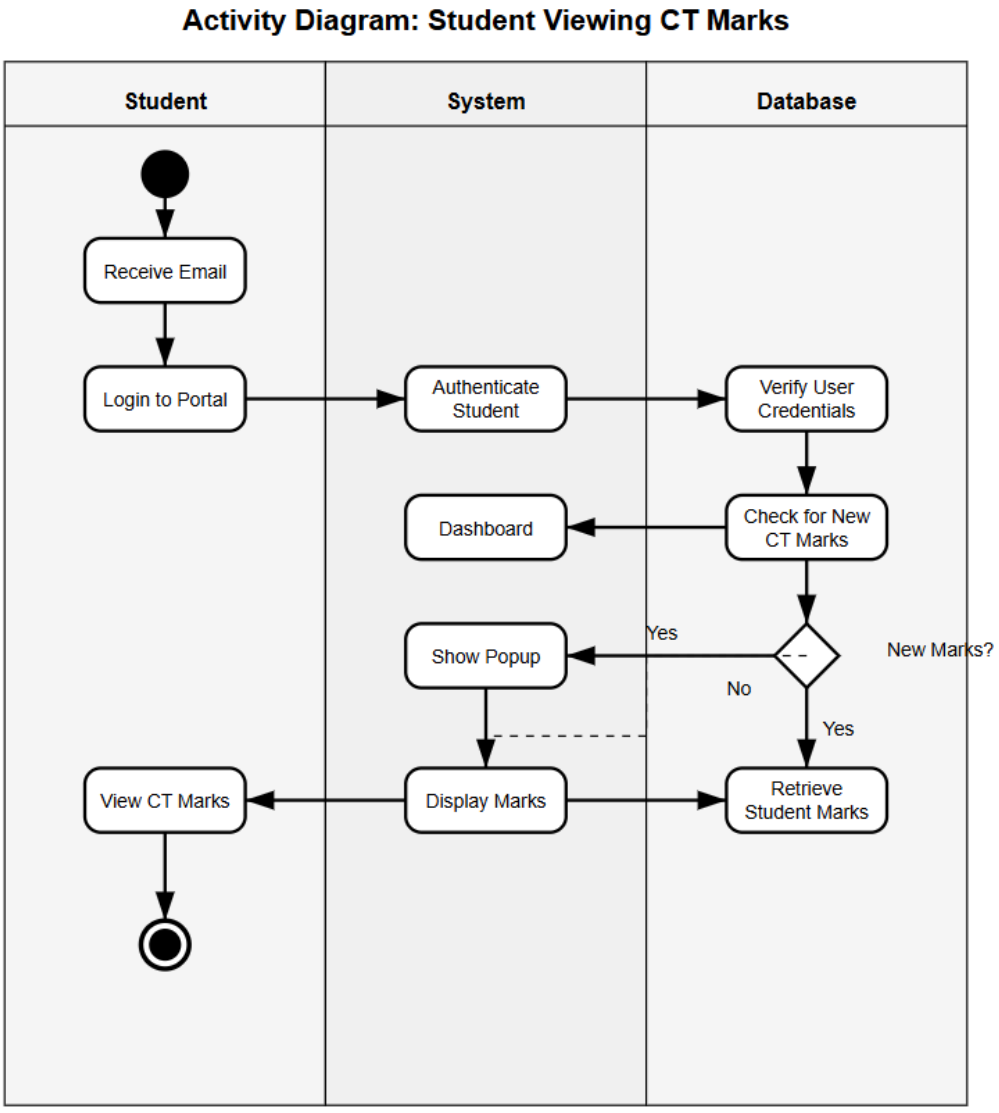
i. Activity Diagram – For 2 Major Processes

1. **Faculty enters CT marks** (Process flow for entering and storing marks)

Activity Diagram: Faculty Entering CT Marks

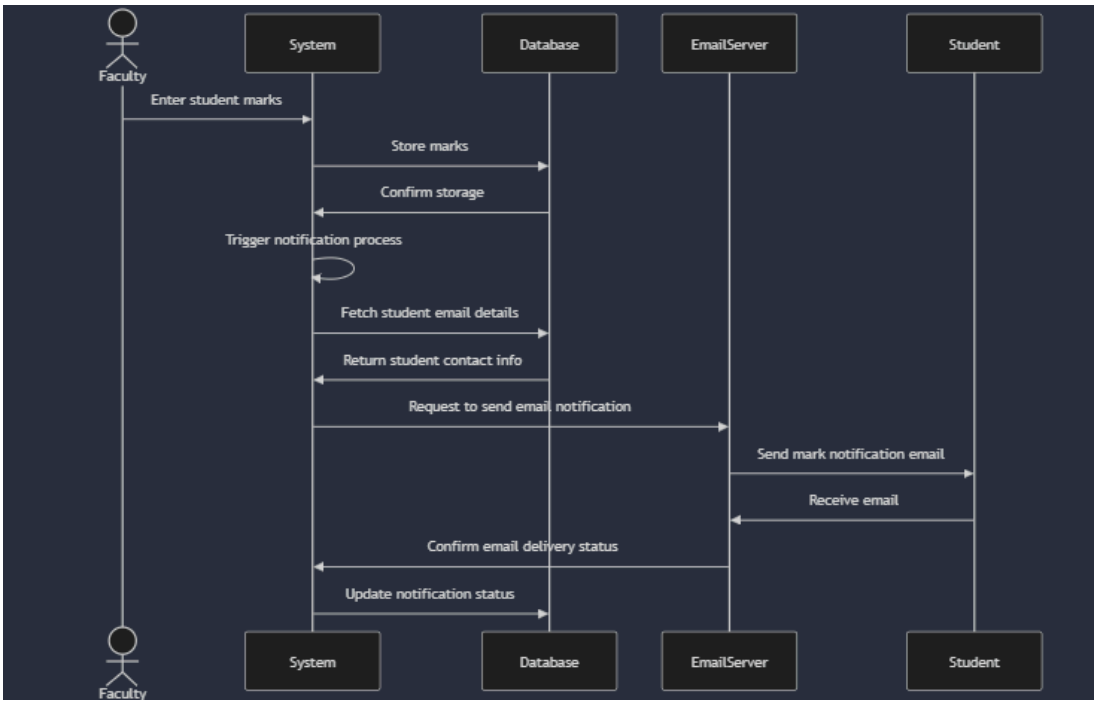


2. **Student views CT marks** (Process flow for viewing marks and receiving notifications)

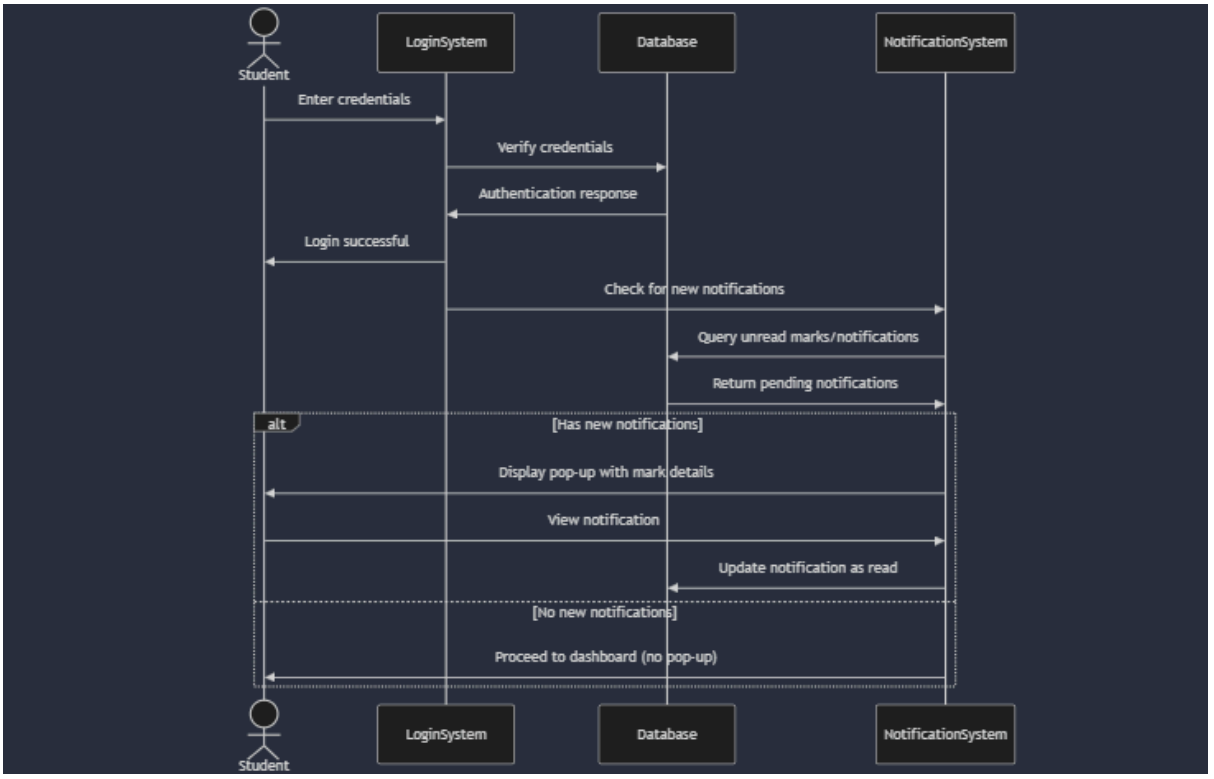


ii. Sequence Diagram – For Other 2 Major Processes

1. **Email notification process** (How the system fetches email and sends marks)



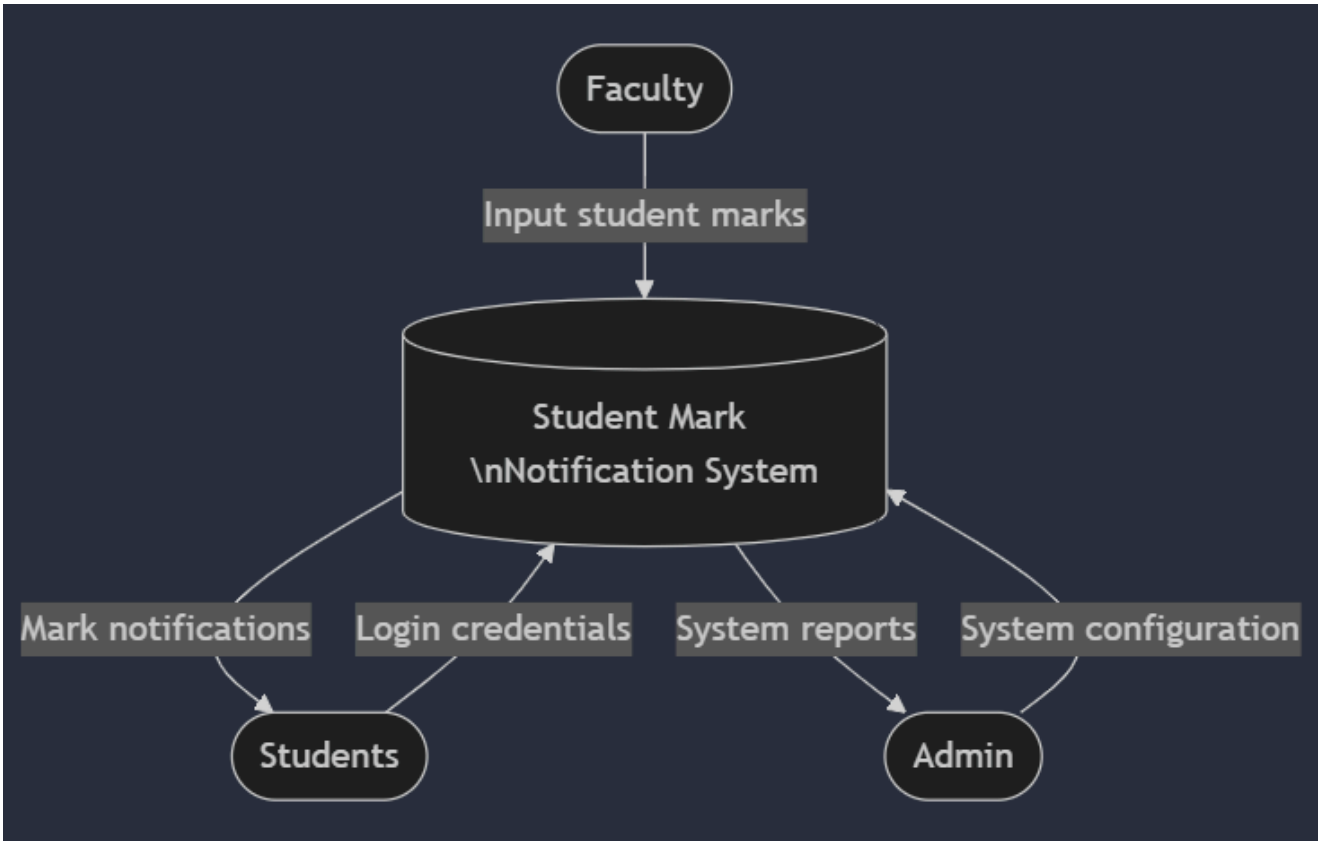
2. **Login and pop-up notification process** (How students receive notifications upon login)



c) Flow Models

i. Level 0 DFD (Context Diagram)

Shows the overall data flow between **Faculty, Students, Admin, Database, and Email Server**.



ii. Level 1 DFD

Breaks down the system into **detailed processes** like:

- **Faculty enters marks** → Stored in **database**.
- **Database triggers email system** → **Email sent to students**.
- **Students log in** → Retrieve **marks from database** → Show **pop-up notification**.