|  |  |
| --- | --- |
| **Name** | SUBASH M |
| **Roll no** | 7376222CT152 |
| **Seat no** | 363 |
| **Project ID** | 30 |
| **Problem Statement** | Curriculum automation |

## Technical Components:

|  |  |
| --- | --- |
| **Component** | **Tech Stack** |
| Backend | python |
| Frontend | Html, css, javascript |
| Database | MySQL |
| API | RESTful services |

PROBLEM STATEMENT:

The current curriculum management system in educational institutions is often cumbersome and prone to errors. Manual processes for course code allocation, LTPC (Lecture-Tutorial-Practical-Credit) checking, duplicate avoidance, exam pattern assignment, and syllabus entry are inefficient and time-consuming. Additionally, the process lacks real-time updates and seamless access for stakeholders, such as staff and students. The need for a streamlined, automated system to manage these tasks is essential to reduce complexity and enhance operational efficiency.

INTRODUCTION

This project aims to develop a real-time full-stack web application for college management that automates the curriculum management process. The application will handle course code allocation, LTPC checking, avoidance of duplicate entries, exam pattern assignment, and syllabus entry. It will also provide a platform for staff to upload course materials and for students to view these materials. The system will categorize courses into professional courses, electives, and one-credit courses, generating course codes automatically.

SCOPE OF PROJECT:

* Automation: Automated allocation of course codes, LTPC validation, and duplicate avoidance.
* Stakeholder Access: Separate interfaces for staff and students to manage and view course materials.
* Real-Time Updates: Real-time updates to ensure all stakeholders have the latest information.
* Course Categorization: Automatic generation of course codes categorized by professional courses, electives, and one-credit courses.
* User-Friendly Interface: Intuitive and easy-to-use interfaces for all users.

WORKFLOW:

1.Course Code Allocation

* Input: Staff inputs course details.
* Process: System checks existing course codes to avoid duplicates and assigns a new code.
* Output: Course code is generated and saved.

2.LTPC Checking:

* Input: Staff enters LTPC values for a course.
* Process: System validates LTPC values based on predefined rules.
* Output: Confirmation or error message displayed.

3.Avoiding Duplicates:

* Input: Staff inputs new course information.
* Process: System compares the new entry with existing records.
* Output: Duplicate check result and appropriate action.

4.Exam Pattern Assignment:

* Input: Staff selects an exam pattern.
* Process: System assigns the selected exam pattern to the course.
* Output: Exam pattern saved and linked to the course.

5.Syllabus Entry:

* Input: Staff uploads syllabus documents.
* Process: System links the syllabus to the respective course.
* Output: Syllabus accessible to students.

COURSE MATERIALS MANAGEMENT:

1. Staff Upload:

* Input: Staff uploads course materials.
* Process: Materials are categorized and linked to the relevant course.
* Output: Materials are stored and accessible to students.

2. Student View:

* Input: Students log in to view course materials.
* Process: System retrieves relevant materials based on student enrollment.
* Output: Course materials displayed to students.

3.Stakeholders:

* Staff: Responsible for uploading course materials, entering course details, and managing curriculum components.
* Students: Access course materials and view curriculum-related information.

Conclusion:

The proposed full-stack web application will revolutionize curriculum management in educational institutions by automating critical tasks and providing real-time access to information for all stakeholders. By reducing complexity and enhancing efficiency, the system will contribute significantly to the smooth operation of academic processes.

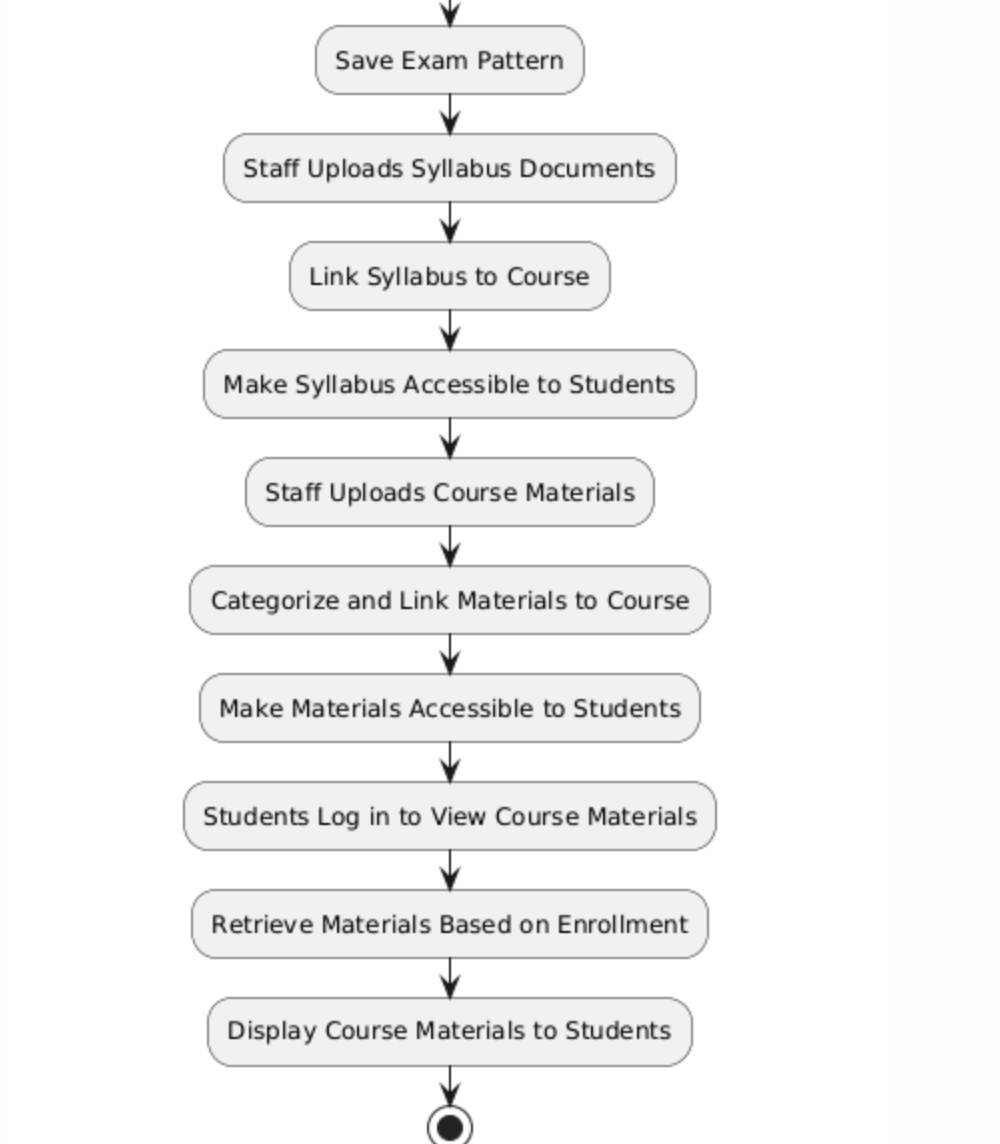
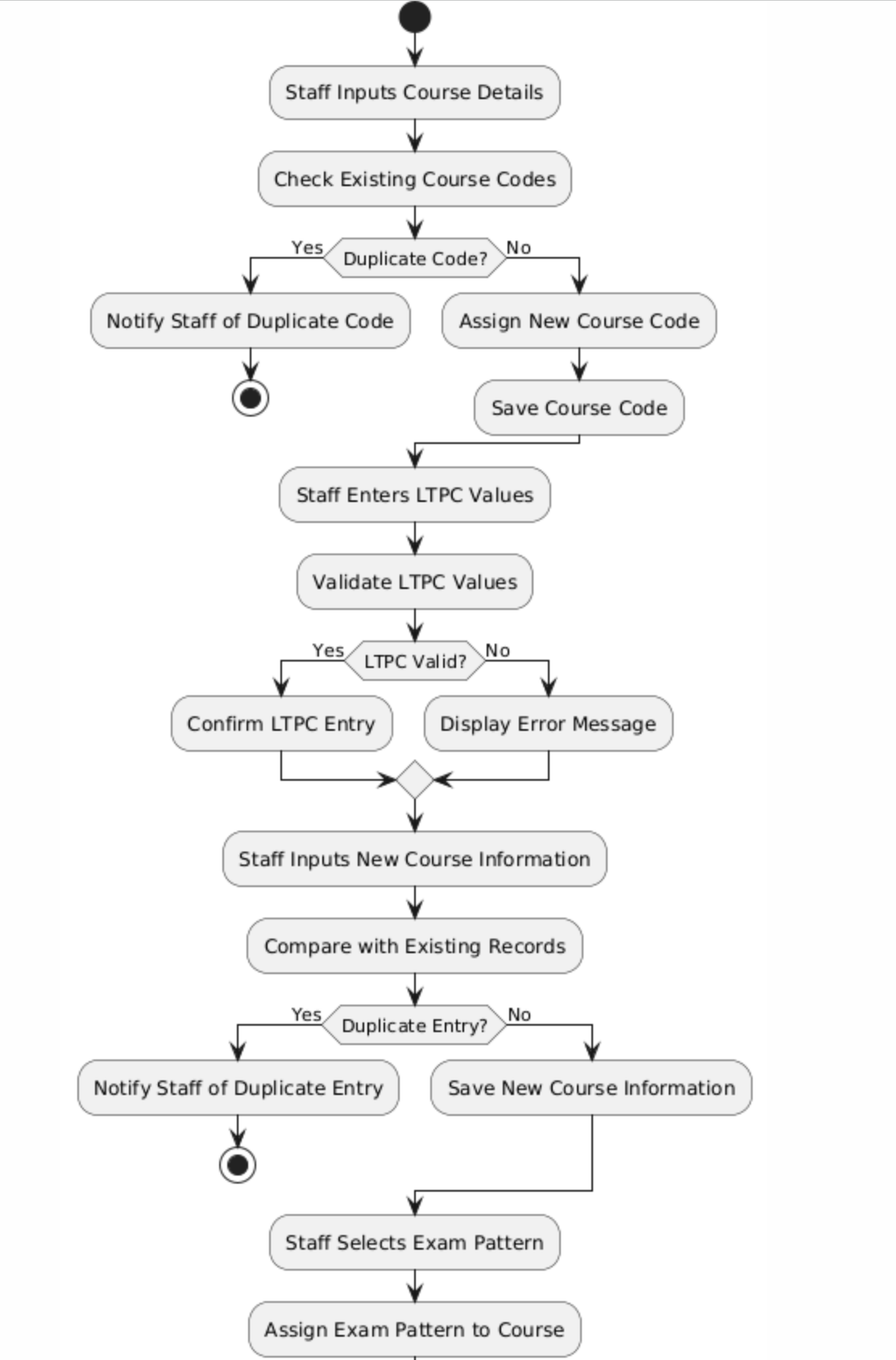
Key Features and Technologies:

* Frontend: HTML, CSS, and JavaScript for dynamic user interfaces.
* Backend: Python with Django for handling server-side logic.
* Database: PostgreSQL and MySQL for robust and scalable data storage.
* API Integration: Open API, SOAP APIs, and RESTful APIs for seamless communication and integration.
* Deployment: Docker for containerization and AWS for cloud deployment.

Implementation Plan:

* Requirement Analysis: Detailed discussion with stakeholders to finalize requirements.
* Design: Create wireframes and system architecture.
* Development: Implement frontend, backend, and database components.
* Testing: Conduct thorough testing to ensure system functionality and reliability.
* Deployment: Deploy the application on a cloud platform for accessibility.
* Training and Support: Provide training to staff and students and offer ongoing support.

This structured approach ensures the successful development and implementation of the college management system, making curriculum management more efficient and effective.

FLOW CHART

ER DAIGRAM: