TASK PORTAL
PROJECT ID – 06

TECH STACK:PYTHON STACK (AI)

NAME : PAWAN G
ROLL NUMBER : 7376222AL180

Problem Statement:

To develop a web-based task management portal for a college that facilitates task assignment and communication between the Head of the Department (HOD), faculty members, and students. The portal will include features for task assignment, acceptance, management, and dashboards for each user type.

Scope of the Project:

The scope includes developing a task management system that allows the HOD to assign tasks to faculty members and students, provides a feature for faculty to accept or decline tasks, enables task reassignment to students by faculty, and supports communication across all user roles.

Goals:

- Efficient task assignment and management between HOD, faculty, and students.
- Secure communication system for all users.
- Role-based dashboards for task tracking and management.
- Notification system for task updates and deadlines.

Functional Requirements:

- **User Authentication:** Secure login for HOD, faculty, and students.
- **Task Assignment:**
    - HOD can assign tasks to faculty and students.
    - Faculty can assign tasks to students.
- **Task Management:**
    - Faculty can accept or decline tasks.
    - Faculty and students can update task status.
- **Dashboards:**
    - HOD, faculty, and students each have personalized dashboards to manage tasks.
- **Communication System:**

    - Secure messaging feature for communication between HOD, faculty, and students.
- **Notifications:**
    - System-generated notifications for task updates, deadlines, and communication.

Non-Functional Requirements:

1. **Performance:** The system should support concurrent users with minimal latency.
2. **Security:** Implement secure data storage and encrypted communication.
3. **Scalability:** Design to accommodate growing numbers of users and tasks.
4. **Compatibility:** Ensure cross-browser and cross-device accessibility.

User Stories:

1. **HOD Login:** As an HOD, I want to securely log in to the portal to assign and monitor tasks.
2. **Faculty Login:** As a faculty member, I want to securely log in to manage tasks assigned by HOD and assign tasks to students.
3. **Student Login:** As a student, I want to securely log in to view and update the tasks assigned to me.
4. **Task Assignment:** As an HOD, I want to assign tasks to faculty and students.
5. **Task Acceptance:** As a faculty member, I want to accept or decline tasks assigned by the HOD.
6. **Task Reassignment:** As a faculty member, I want to assign tasks to students.
7. **Task Dashboard:** As a user, I want a personalized dashboard to manage my tasks.
8. **Messaging:** As a user, I want to communicate with other users securely.
9. **Receive Notifications:** As a user, I want to receive notifications for task updates and deadlines.

Workflow:

1. Planning Phase

- Identify the problem statement, scope, goals, and objectives.
- Stakeholder meetings to finalize initial requirements.

2. Requirements Phase

- Gather detailed requirements from stakeholders (HOD, faculty, and students).
- Define user stories and use cases.
- Document functional and non-functional requirements.

3. Architecture Phase

- Select appropriate technologies: Django for the backend, React for the frontend, MySQL for the database.
- Define system architecture and database schema.
- Plan RESTful API structure for frontend-backend communication.

4. Development Phase

- **Frontend Development:**
  - Design and develop user interfaces using React, HTML, CSS, and JavaScript.
  - Create role-based dashboards and task management views.
- **Backend Development:**
  - Set up Django project structure and implement models, views, and controllers.
  - Integrate with MySQL for data storage and implement Django ORM.
  - Develop APIs for task management, user authentication, and communication.
- **Integration:**
  - Integrate frontend with backend via RESTful APIs.
  - Implement the communication system and notification feature.

5. Testing and Deployment Phase

- Conduct unit testing, integration testing, and user acceptance testing (UAT).
- Optimize performance and security measures.
- Deploy the application to a production server.
- Provide training and documentation for users.

Implementation:

1. Web Page Design:

- Use design tools like Figma to create wireframes.
- Discuss designs with stakeholders for feedback and approval.

2. Frontend Development:

- Implement responsive and interactive UI using React.
- Develop role-based dashboards and task management interfaces.

3. Database Design and Implementation:

- Design the database schema in MySQL.
- Implement database interactions using Django ORM.

4. Backend Development:

- Set up the Django framework and develop backend logic.
- Create and secure APIs for frontend communication.
- Implement notification system and task assignment logic.

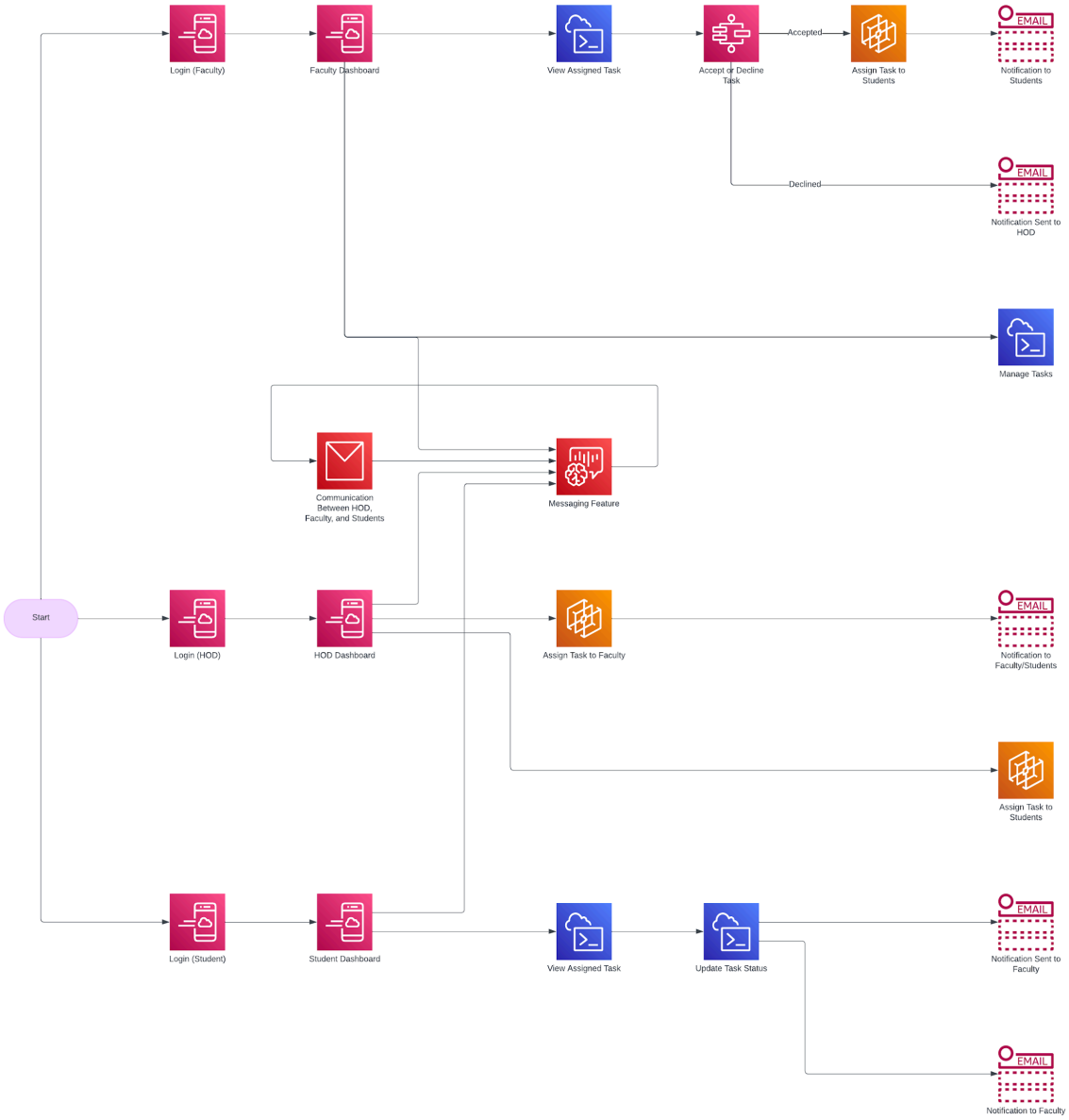
5. Testing and Deployment:

- Perform testing across all user roles and functionalities.
- Deploy the application to a cloud server for college-wide access.
- Set up a maintenance plan for ongoing support.

System Architecture:

Front End	HTML, CSS, JavaScript
Backend	Python, Django
Database	MySQL
API	OpenAPI, SOAP APIs, RESTful APIs

Flowchart:


Start
Login (Faculty)
Faculty Dashboard
View Assigned Task
Accept or Decline Task
Accepted
Assign Task to Students
Notification to Students
Declined
Notification Sent to HOD
Manage Tasks
Communication Between HOD, Faculty, and Students
Messaging Feature
Login (HOD)
HOD Dashboard
Assign Task to Faculty
Notification to Faculty/Students
Assign Task to Students
Login (Student)
Student Dashboard
View Assigned Task
Update Task Status
Notification Sent to Faculty
Notification to Faculty