Requirements Document for Academic and Administrative Website (AAW)

Project Overview

The Academic and Administrative Website (AAW) is an end-to-end platform designed to streamline the educational experience for students, faculty, and administrators. The system centralizes course management, assignments, grading, and announcements, providing an organized and interactive learning environment.

Stakeholders

- Students: Access course materials, submit assignments, take quizzes, and view grades.
- Faculty: Manage course content, track student progress, and communicate with students.
- Administrators: Oversee course assignments, track faculty and student activities, and ensure the smooth operation of the educational institution.

Scope

The AAW will provide the following functionalities:

1. Student Functionality:
   - View enrolled courses (current and previous semesters).
   - Access published course content.
   - View quizzes and assignments.
   - View personal grades.
   - Set profile information and notifications.

2. Faculty Functionality:
   - View homepage with a list of courses (current and previous semesters).
   - Manage courses (publish/unpublish).
   - Add content to syllabus.

- View student lists and grades.

- Assign grades.

- Add assignments and quizzes.

- Post announcements.

3. Admin Functionality:

- View courses by faculty by semester.

- Assign courses to faculty for new semesters.

- View student lists (without grades).

4. System-wide Functionality:

- User authentication and authorization.

- Report generation and analytics.

- Announcement system.

Tech Stack

- Frontend: Next.js, HTML, Tailwind CSS

- Backend: Flask

- Database: MySQL

Functional Requirements

1. User Authentication and Authorization:

- Secure login and signup.

- Role-based access control (Student, Faculty, Admin).

2. Course Management:

- CRUD operations for courses.

- Faculty can manage courses and students can view their enrolled courses.

3. Assignment and Quiz Management:

- Faculty can create and grade assignments/quizzes.

- Students can submit and view assignments/quizzes.

4. Grade Management:

- Real-time grade assignment by faculty.

- Students can view their grades.

5. Announcement System:

- Faculty/Admin can post announcements.

- Students receive notifications.

6. Reporting and Analytics:

- Generate reports on student performance and engagement.

Non-Functional Requirements

1. Security:

- Data encryption for sensitive information.

- Secure APIs to prevent unauthorized access.

2. Performance:

- Efficient data retrieval and updates.

- Scalability to handle a large number of users.

3. Usability:

- Intuitive and user-friendly interface.
- Accessible from various devices and platforms.

4. Reliability:

- High availability and minimal downtime.
- Regular backups and data recovery mechanisms.

Design and Architecture

1. OOP Principles:

- Encapsulation: Course class encapsulates course details.
- Inheritance: User class with derived classes for Faculty, Student, and Admin.
- Polymorphism: ViewCourses method with different implementations for Faculty and Student.
- Abstraction: Simplified interfaces for complex operations.

2. MVC Framework:

- Model: Manages core data and business rules.
- View: Provides the user interface.
- Controller: Processes user inputs and updates the view.

3. Middleware:

- Bridges frontend and backend.
- Manages user requests, business rules, and data flow.

UI Mockups

1. Login / SignUp Page

2. Student Page

   - View assigned grades.

3. Admin Page

   - Assign Faculty to courses.

Implementation Plan

1. Phase 1: Requirements Gathering and Analysis

   - Conduct stakeholder meetings.

   - Document detailed requirements.

2. Phase 2: Design

   - Create wireframes and mockups.

   - Design database schema and API endpoints.

3. Phase 3: Development

   - Implement frontend and backend.

   - Integrate with the database.

4. Phase 4: Testing

   - Perform unit, integration, and system testing.

   - Conduct user acceptance testing (UAT).

5. Phase 5: Deployment and Maintenance

   - Deploy the system to the production environment.

   - Provide ongoing support and maintenance.

Conclusion

The AAW aims to enhance the educational experience by providing a centralized platform for course management, assignments, grading, and communication. The project will be implemented using modern web technologies, adhering to best practices in security, performance, and usability.