Department of Computer Science & Engineering

Software Requirements Specification

Mini Project - Software Engineering

UE22CS341A

Project Title:  University Assignment Portal

Submission detail: Deliverable-2

Team Details:


Member 1:

Name: Arjun N R

SRN: PES2UG22CS910

Member 2:

Name: Mohammed Hassan

SRN: PES2UG 22CS317


Date of Submission: 04-10-2024

GitHub Link: Assignment Portal Git

Corse Instructor:

Prof. Rohith Vaidya K

# Table of Contents

---

## 1. Project Lifecycle Model

Model: Agile Scrum

The Agile Scrum methodology has been chosen for the University Assignment Portal project due to its strong focus on adaptability, iterative development, and collaboration. This approach is particularly well-suited for projects in dynamic environments like higher education, where requirements can shift based on stakeholder feedback.

Justification

1. Flexibility:
   Agile's iterative cycles, or sprints, enable the project team to respond swiftly to changing requirements. This adaptability is crucial in a university setting, where the needs of students, professors, and administrators may evolve over time. By focusing on smaller, incremental improvements, the team can implement feedback from each sprint to adjust the project direction as necessary.

2. Regular Deliverables:
   Scrum's structured approach ensures that the team delivers functional increments of the product at the end of each sprint (typically every two weeks). This frequent delivery allows stakeholders to review the current state of the project, assess progress, and provide valuable feedback. As a result, the project remains aligned with user expectations and requirements throughout its development.

3. Collaboration:
   Scrum fosters constant communication among team members, stakeholders, and product owners. Regular meetings, including sprint planning, daily stand-ups, and sprint reviews, ensure that everyone is on the same page regarding project status, challenges, and priorities. This ongoing dialogue helps clarify requirements, address concerns promptly, and maintain a shared understanding of project goals.

4. Risk Management:
   The iterative nature of Agile allows for early identification and mitigation of risks. Continuous delivery and regular testing help uncover potential issues before they escalate into significant problems. By addressing risks in smaller increments, the project team can make informed decisions and pivot strategies as necessary, enhancing the overall quality and success of the final product.

Sprint Structure

The sprint structure within the Agile Scrum methodology includes the following components:

1. Sprint Duration: Each sprint will last for two weeks, providing a focused timeframe for completing specific tasks and features.

2. Sprint Planning: At the beginning of each sprint, the team will hold a planning session to define the objectives for the upcoming sprint. This involves selecting user stories from the product backlog that will be addressed during the sprint.
3. Daily Stand-Ups: The team will conduct short, daily stand-up meetings to discuss progress, identify any blockers, and coordinate efforts. This keeps the team aligned and focused on sprint goals.

4. Sprint Review: At the end of each sprint, the team will hold a review meeting where they will demonstrate the completed work to stakeholders. This session is an opportunity for feedback and to ensure that the project is meeting stakeholder expectations.

5. Sprint Retrospective: Following the sprint review, the team will conduct a retrospective to reflect on the sprint's process. This meeting allows the team to identify successes and areas for improvement, fostering a culture of continuous learning and adaptation.

This Agile Scrum approach will not only streamline the development process for the University Assignment Portal but also enhance stakeholder engagement and satisfaction, ensuring that the final product effectively meets the needs of all users.

---

## 2. Tools for Project Lifecycle

To ensure smooth project execution for the University Assignment Portal, a variety of tools will be utilized across different stages of the project lifecycle, including planning, design, development, and testing. These tools have been selected based on their capabilities, integration potential, and suitability for the project requirements.

Planning Tool: Jira

1. Justification:
   Jira provides a comprehensive platform for agile project management, including functionalities for sprint planning, backlog management, and bug tracking. Its integration capabilities with tools like GitHub and Slack allow for a centralized environment where team members can collaborate effectively. This integration streamlines communication, ensuring that everyone is aligned on project progress and tasks.

2. Usage:
   Tasks and user stories will be created in Jira, forming the basis of the project backlog. Items will be prioritized for each sprint, allowing the team to focus on the most critical features and improvements. The ability to track progress and link bugs directly to user stories enhances transparency and accountability throughout the development process.

Design Tool: Figma

1. Justification:
   Figma is a powerful cloud-based design tool that supports real-time collaboration. Designers and developers can work together to create and iterate on UI/UX designs, which is essential for developing the assignment portal.

2. Usage:
   The UI/UX of the portal will be designed using Figma, including key interfaces such as the student dashboard, professor dashboard, and admin interface. Figma will facilitate the creation of wireframes, mockups, and prototypes, enabling the team to visualize the application and gather feedback from stakeholders early in the design process.

Version Control: Git (GitHub)

1. Justification:
   Git is the industry-standard tool for version control, essential for managing code changes in collaborative environments. GitHub enhances this by providing a platform where developers can work on the same codebase, review changes, and track project progress efficiently.

2. Usage:
   The team will adopt a feature-branch workflow, where each feature or bug fix is developed in a separate branch. After development, changes will undergo a code review process before being merged into the main branch. This practice not only improves code quality but also ensures that all changes are tracked and documented.

Development Tools:

1. Frontend: Visual Studio Code
   a. Justification: Visual Studio Code is a lightweight and powerful IDE that is well-suited for frontend development with HTML, CSS, and JavaScript. It supports a wide range of plugins that enhance productivity and workflow.

   b. Usage: Developers will use Visual Studio Code for writing, debugging, and testing frontend code. The IDE's live server functionality allows for real-time previewing of changes, improving the development process.

2. Backend: Python with Flask
   a. Justification: The backend will be developed using Python with the Flask framework. Flask is a lightweight and flexible web framework that is well-suited for building RESTful APIs and handling backend logic efficiently.
   b. Usage: Developers will use Python with Flask to create the application’s backend, managing user authentication, assignment management, and

database interactions. Flask's simplicity allows for rapid development while maintaining code quality.

Database: SQL

1. Justification: A relational database management system using SQL will be employed to manage the data for the University Assignment Portal. SQL provides robust capabilities for data manipulation and querying.
2. Usage: The database will store user information, assignments, submissions, grades, and other related data. Developers will use SQL to interact with the database, executing queries to retrieve and update data as needed.

Deployment Tool: Nginx

1. Justification: Nginx is a high-performance web server and reverse proxy server that is widely used for serving applications. It efficiently handles concurrent connections and can be configured to serve static files while acting as a reverse proxy for dynamic content generated by the Flask application.

2. Usage: Nginx will be used to deploy the University Assignment Portal. It will manage incoming web traffic, serve static files, and route requests to the Flask application. The combination of Nginx with a WSGI server (like Gunicorn) will ensure optimal performance and reliability of the application in a production environment.

Bug Tracking: Jira

1. Justification:
   Since Jira is already being employed for project management, it is logical to also use it for tracking bugs. This integration allows for seamless tracking and categorization of bugs, facilitating better workflow management.

2. Usage:
   Any bugs identified during the development and testing phases will be logged in Jira. Each bug will be categorized based on severity and assigned to appropriate team members for resolution, linking bugs to relevant user stories for context.

Testing Tools:

1. Selenium: Used for Automated End-to-End (E2E) Testing
   a. Justification: Selenium is a widely adopted testing framework for web applications. It simulates user interactions with the application, verifying that the system functions as intended under real-world conditions.

   b. Usage: Automated tests will be developed using Selenium to simulate scenarios such as student submissions, professor grading, and admin reporting functionalities, ensuring that the application behaves correctly in

various situations.

2. PyTest: Used for Unit Testing in Python
    a. Justification: PyTest is a flexible and powerful testing framework tailored for Python applications. It allows developers to write simple and scalable test cases, supports fixtures and mocking, and is particularly well-suited for testing the Flask backend.

    b. Usage: Unit tests will be created to validate the functionality of various backend modules, including user management, assignment creation, and grading systems. This ensures that individual components work correctly and as expected.

By leveraging these tools throughout the project lifecycle, the team can ensure effective collaboration, maintain high code quality, and deliver a robust University Assignment Portal that meets the needs of students, professors, and administrators. The use of Python and Flask for backend development, combined with SQL for data management and Nginx for deployment, provides a solid foundation for a scalable and maintainable application.

---

## 3.Deliverables Categorization

The project deliverables for the University Assignment Portal are categorized into two main groups: Reuse Components and Build Components. This classification helps streamline development efforts by leveraging existing technologies while also ensuring that custom functionalities meet specific user needs.

Reuse Components

Reuse components are pre-existing libraries and frameworks that will be utilized to accelerate development and ensure reliability. These components are well-established and provide essential functionalities that are critical for the project.

1. Authentication System
    a. Description: The portal will implement user authentication using Flask-Login, a widely-used library for handling user sessions and managing access controls.
    b. Justification:
    Flask-Login simplifies the implementation of user authentication processes, including user registration, login/logout, and session management. By utilizing this library, the project benefits from a secure and standardized method for handling user identities, reducing the complexity of implementing authentication from scratch. This approach not only speeds up development but also adheres to best practices for security.

2. Database Management System: MySQL

a. Description: The project will use MySQL as the relational database management system to store and manage application data.
b. Justification:
MySQL is a robust, widely-used database that supports a wide range of data types and complex queries. Direct SQL queries will be written to perform data operations, providing the flexibility needed for custom interactions with the database. This approach allows developers to have fine-grained control over the database operations, ensuring optimal performance and security without the overhead of an ORM.

3. Frontend Framework (Bootstrap)
    a. Description: The user interface of the portal will be developed using Bootstrap, a responsive front-end framework designed for mobile-first web applications.
    b. Justification:
    Bootstrap provides a collection of pre-designed components, such as buttons, forms, and navigation bars, which help speed up UI development. Its responsive grid system ensures that the application is mobile-friendly and provides a consistent user experience across different devices and screen sizes. By using Bootstrap, the development team can focus on functionality while maintaining a polished look and feel for the application.

Build Components

Build components are custom functionalities specifically designed to meet the unique requirements of the University Assignment Portal. These components will be developed from scratch or customized to provide tailored solutions for users.

1. User Management Module
    a. Description: This module will manage different user roles, including students, professors, and administrators, with specific access privileges and functionalities.
    b. Justification:
    The user management system is crucial for ensuring that users have the appropriate permissions to access certain features of the portal. For instance, professors should be able to create and manage assignments, while students should have access to view their submissions and grades. Customizing this module allows the application to enforce security policies and maintain data integrity by ensuring that users can only perform actions relevant to their roles.

2. Assignment Creation and Management
    a. Description: Professors will have the ability to create, edit, and publish assignments through a user-friendly interface, specifying important details such as deadlines and instructions.
    b. Justification:
    This functionality is the core of the portal, enabling professors to

communicate assignments effectively to students. A well-designed assignment management system ensures that assignments can be easily tracked and that students receive timely notifications. Custom development is necessary to align this functionality with the university's specific requirements for assignment management, including compliance with academic policies and procedures.

3. Submission Handling System
    a. Description: The system will facilitate the uploading, tracking, and validation of assignment submissions made by students.
    b. Justification:
    Handling submissions effectively is vital to the success of the portal. The system must be capable of managing large file uploads, validating file types, and ensuring that submissions are time-stamped to meet deadlines. Furthermore, the system will provide notifications to students upon successful submission, improving the user experience and reducing anxiety related to submission statuses. A custom implementation is required to ensure that this system meets the university's specific criteria for assignment submissions.

4. Grading and Feedback Module
    a. Description: Professors will be able to grade assignments and provide constructive feedback through a secure and organized interface.
    b. Justification:
    This module is essential for maintaining academic standards and supporting student learning. Custom workflows will be developed to accommodate the unique grading criteria and feedback mechanisms established by the university. This may include rubric-based grading, which provides clear expectations for students, and options for professors to give personalized feedback. The ability for students to view their grades and feedback fosters transparency and encourages engagement with their academic progress.

5. Reporting System
    a. Description: The system will generate comprehensive reports on assignment statuses, grades, and overall student performance metrics.
    b. Justification:
    Reporting is critical for both administrative purposes and for providing insights into student performance. Custom reports can be tailored to meet the specific requirements of university administrators, allowing them to monitor academic progress and identify areas needing attention. This module will support data-driven decision-making processes and enable educators to provide better support to students based on their performance analytics.

## 4.Work Breakdown Structure (WBS)

Phase 1: Project Initiation (1 Week)
- 1.1 Define Project Scope
- 1.2 Identify Stakeholders
- 1.3 Create Project Charter
- 1.4 Perform Initial Risk Assessment

Phase 2: Requirements Gathering (3 Weeks)
- 2.1 Conduct Stakeholder Interviews
- 2.2 Document Functional and Non-Functional Requirements
- 2.3 Validate Requirements with Stakeholders

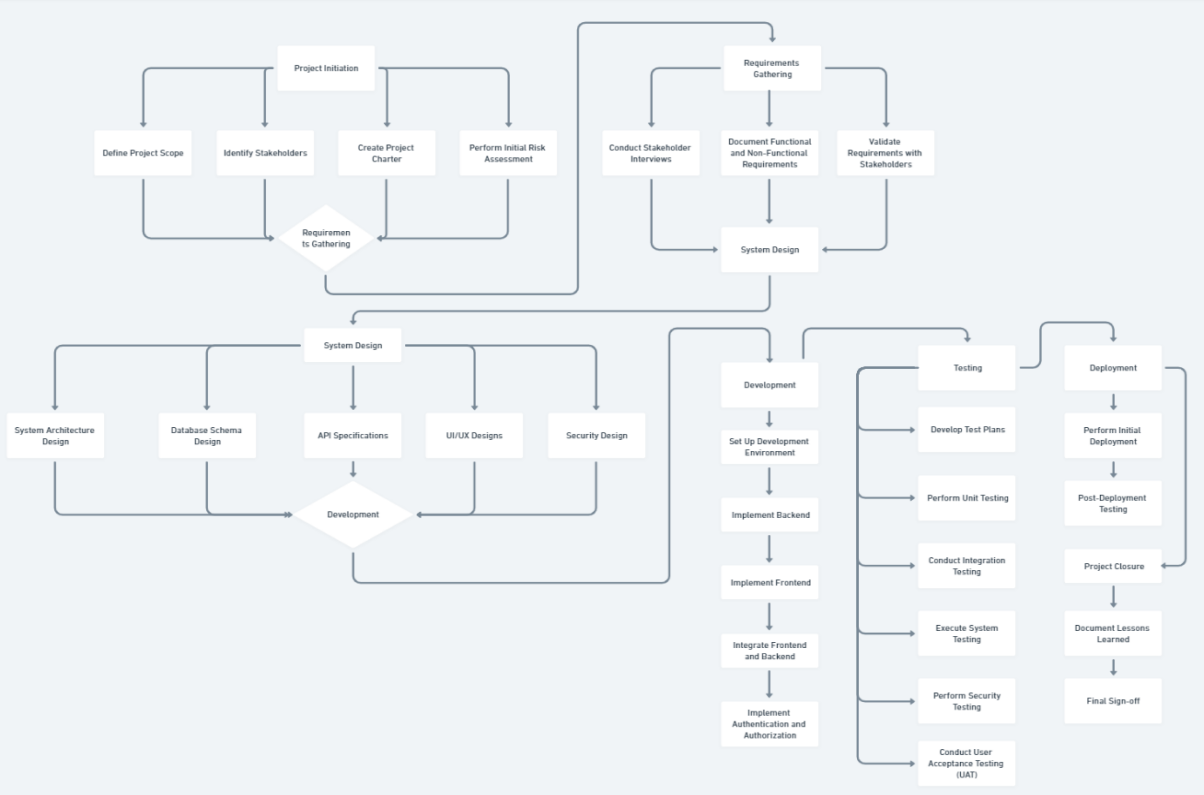
Phase 3: System Design (4 Weeks)
- 3.1 System Architecture Design
- 3.2 Database Schema Design
- 3.3 API Specifications
- 3.4 UI/UX Designs
- 3.5 Security Design

Phase 4: Development (12 Weeks)
- 4.1 Set Up Development Environment
- 4.2 Implement Backend
- 4.3 Implement Frontend
- 4.4 Integrate Frontend and Backend
- 4.5 Implement Authentication and Authorization

Phase 5: Testing (4 Weeks, overlapping with Development)
- 5.1 Develop Test Plans
- 5.2 Perform Unit Testing
- 5.3 Conduct Integration Testing
- 5.4 Execute System Testing
- 5.5 Perform Security Testing
- 5.6 Conduct User Acceptance Testing (UAT)

Phase 6: Deployment (2 Weeks)
- 6.1 Prepare Production Environment
- 6.2 Perform Initial Deployment
- 6.3 Post-Deployment Testing

Phase 7: Project Closure (1 Week)
- 7.1 Conduct Project Review
- 7.2 Document Lessons Learned
- 7.3 Final Sign-off

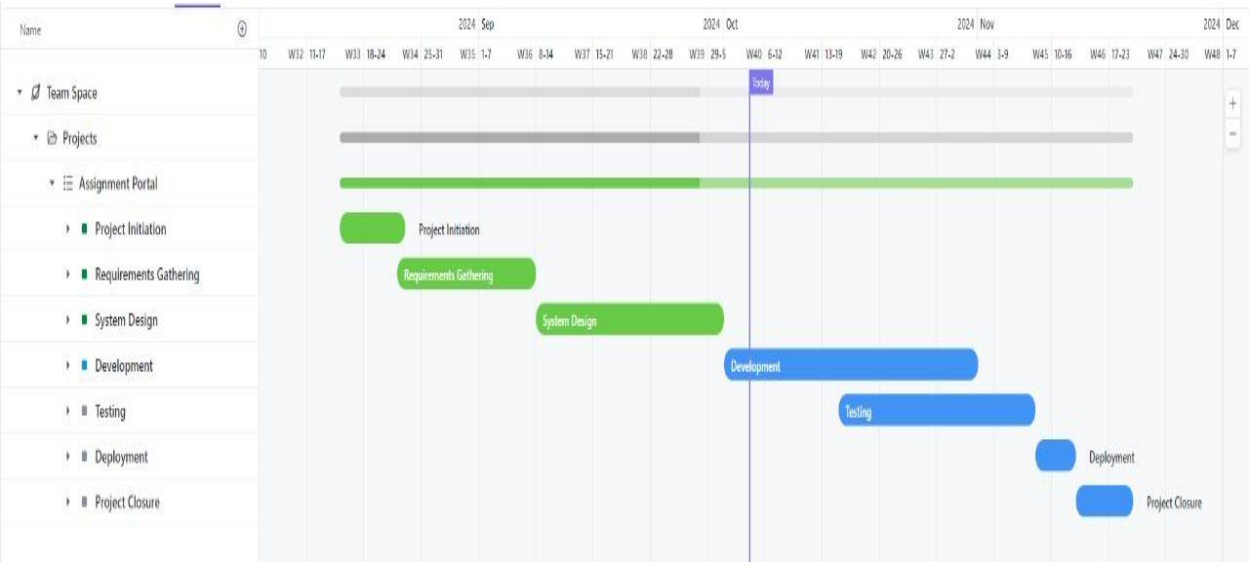



## 4.Effort Estimation and Gantt Chart

### Effort Estimation Overview

The total estimated effort for the University Assignment Portal project is 10.5 person-months. This estimation encompasses all phases of the project, from initiation to closure, and is based on the expected workload associated with each task.

The effort is broken down as follows:



| Phase | Estimated Effort (Person-Months) | Description |
|---|---|---|
| 1. Project Initiation | 0.5 | Involves defining the project scope, identifying stakeholders, and creating the project charter. |
| 2. Requirements Gathering | 1.0 | Includes stakeholder interviews, documenting requirements, and validating these requirements. |
| 3. System Design | 1.5 | Encompasses designing system architecture, database schema, and user interface. |
| 4. Development | 4.0 | The core phase where backend and frontend functionalities are implemented, including all modules. |
| 5. Testing | 2.0 | Involves unit testing, integration testing, and user acceptance testing to ensure software quality. |
| 6. Deployment | 1.0 | Preparing and executing the deployment of the application to the production environment. |
| 7. Project Closure | 1.5 | Conducting project review, documenting lessons learned, and obtaining final sign-off from stakeholders. |

## 6. Coding Details

Frontend Development

1. Languages:
    a. HTML5: The standard markup language for structuring web pages, providing the skeleton for the UI.
    b. CSS3: Used for styling the UI, allowing for responsive designs and visually appealing layouts.
    c. JavaScript (ES6+): The primary scripting language for adding interactivity to web pages. The ES6+ version includes modern features such as arrow functions, classes, and template literals, enhancing code efficiency and readability.

2. Framework:
    a. Bootstrap: A popular front-end framework that facilitates responsive web design. It includes a wide range of pre-designed components (buttons, forms, navigation bars) and a responsive grid system, which helps ensure that the application works well on both desktop and mobile devices. Using Bootstrap significantly speeds up development time and ensures consistency in design across different pages.

3. Build Tool:
    a. Webpack: A powerful module bundler that compiles JavaScript modules and assets into a single bundle for production. It optimizes assets by minifying JavaScript and CSS, enhancing loading times and overall performance. For students, Webpack's rich ecosystem of plugins and loaders allows for handling various file types (such as images and stylesheets) seamlessly.

4. Code Style:
    a. Airbnb JavaScript Style Guide: Following this widely recognized style guide ensures that the code is written consistently, making it easier for team members to read and understand each other's code. Consistent code style also reduces the likelihood of errors and improves maintainability.

---

Backend Development

1. Languages:
   Python 3.8+: A versatile programming language known for its simplicity and readability. Python's extensive libraries and frameworks make it suitable for web development and data manipulation. Using Flask allows for the rapid development of web applications due to its lightweight nature and ease of use.

2. Framework:
   Flask: A micro web framework for Python that is easy to set up and extend. Flask is

suitable for building RESTful APIs, which allows for straightforward communication between the frontend and backend. The minimalistic design of Flask helps BTech students learn and experiment without overwhelming complexity.

3. Database:
   MySQL: A robust relational database management system. MySQL is widely used in industry and academia, making it a great choice for students to learn about data storage and management. It offers a powerful query language (SQL) that enables complex data operations.

4. Code Style:
   PEP 8 Guidelines: Adhering to PEP 8, the style guide for Python code, ensures that code is written in a clean and readable manner. This includes conventions for naming variables, structuring code, and using comments effectively, which are all crucial for collaborative projects among students.

---

## 7. Version Control

Version Control System (VCS)

1. Branching Strategy:
   Feature Branching: Each new feature or bug fix will be developed in a separate branch. This allows for isolated development and makes it easier to manage multiple features simultaneously. BTech students can work on their features independently without interfering with others' work, reducing the chances of merge conflicts.

2. Code Review:
   Pull Requests: All changes must be submitted as pull requests to the main branch. This practice encourages collaboration and knowledge sharing among team members, as each change will be reviewed by another team member. Automated tests must pass before a pull request can be merged, ensuring code quality and stability.

3. Repository:
   GitHub: The primary platform for hosting the code repository. GitHub provides a user-friendly interface for managing code, issues, and project documentation. Its collaborative features, such as pull requests and code reviews, are invaluable for students working in teams. Additionally, GitHub offers free resources and education packs for students, making it accessible and beneficial for learning.

---

## 8. Testing

Testing Frameworks and Processes

1. Backend Testing:
    a. Unit Testing:
    PyTest: A powerful testing framework for Python that allows developers to write simple yet scalable test cases. It supports fixtures and mocking, making it suitable for testing individual components of the backend, such as user management and assignment creation.
    b. Integration Testing:
    This ensures that different backend modules work together correctly and that the application communicates effectively with the MySQL database. Testing database interactions directly helps catch potential issues before deployment.

2. Frontend Testing:
    a. Unit Testing:
    Jest: A popular testing framework for JavaScript that is simple to use and integrates well with React and other frameworks. It is useful for testing individual components like assignment submission forms to ensure they behave as expected.
    b. End-to-End (E2E) Testing:
    Selenium: This testing framework simulates user interactions with the portal, validating the entire flow from assignment creation to submission and grading. E2E testing is critical for ensuring that all components of the application work together seamlessly.

Testing Process:

1. Write Unit Tests: Each module should have a set of unit tests to ensure that individual functions and components operate correctly. This is particularly important for critical functionalities such as user authentication and assignment management.
2. Perform Integration Tests: After unit testing, integration tests will be conducted to verify that various modules (e.g., submission handling and grading) interact correctly and share data as intended. This step is crucial for identifying issues that may not be apparent when testing modules in isolation.
3. Run End-to-End Tests: Selenium will be used to simulate real user scenarios, such as students submitting assignments and professors grading them. This comprehensive testing ensures that the application functions correctly in a production-like environment.

## 9. Security Considerations

Security is a fundamental aspect of the University Assignment Portal, especially since it handles sensitive data related to user accounts, assignments, and grades. The following key security measures will be implemented to protect the application from vulnerabilities:

1. HTTPS:
    a. Description: Secure communication between the frontend and backend will be enforced through HTTPS. This ensures that data transmitted over the network is encrypted, protecting it from interception by unauthorized parties.
    b. Implementation: An SSL certificate will be configured on the Nginx server to secure all HTTP requests, preventing potential man-in-the-middle attacks.

2. Password Hashing:
    a. Description: All user passwords will be securely hashed using bcrypt before storage in the database. This practice ensures that even if the database is compromised, user passwords cannot be easily retrieved.
    b. Implementation: Flask-Bcrypt or a similar library will be utilized to implement secure password hashing and verification processes.

3. Rate Limiting:
    a. Description: API endpoints will be protected against abuse by limiting the number of requests a user can make in a given timeframe. This measure helps prevent brute force attacks and ensures fair usage of server resources.
    b. Implementation: A middleware solution can be implemented in Flask to track request counts and enforce rate limits based on user IP addresses.

4. Data Encryption:
    a. Description: Sensitive data, including assignments and grades, will be encrypted both in transit (using HTTPS) and at rest (in the MySQL database). This measure protects user data from unauthorized access.
    b. Implementation: In addition to HTTPS, data encryption libraries can be used to encrypt sensitive fields in the database.

5. Security Audits:
    a. Description: Regular vulnerability assessments will be conducted to identify potential security flaws and ensure that the portal remains secure from external threats.
    b. Implementation: Automated tools such as OWASP ZAP can be used to perform security scanning on the application, alongside manual code reviews to identify and mitigate vulnerabilities.

## Conclusion

By implementing these coding practices, utilizing student-friendly tools, and prioritizing security considerations, the University Assignment Portal will be developed with a focus on quality, maintainability, and user safety. This comprehensive approach ensures that the application not only meets the functional requirements of its users but also adheres to industry standards for security and performance.