Flexible Virtual School

Table of Contents

Revision Control History

Version	Date	Description of change

Implementation Consideration

Database:

Oracle Database will be used to store and manage data such as user profiles, course materials, and transaction records.

Server-Side Implementation:

Python will be used to implement the server-side logic, including handling user authentication, managing course content, and processing payments.

Client-Side Implementation:

HTML, CSS, and JavaScript will be used to develop the client-side interface, providing users with a responsive and interactive learning environment.

Mobile Application Development:

The application will be developed using Swift for iPhone mobile phones, providing a native iOS experience for users accessing the virtual school platform on their iPhones.

Integration with Gmail API:

The system will utilize the Gmail API to enable features such as email notifications, announcements, and communication between users and instructors.

Integration with Collabinate API:

The system will integrate with the Collabinate API to facilitate collaborative learning experiences, allowing users to share resources, discuss topics, and collaborate on projects within the virtual classroom environment.

Integration with Moyasar Payment Gateway:

The system will integrate with the Moyasar payment gateway to enable secure online payment processing for course registration fees and other transactions.

Architecture Analysis/Testing

Method and Process:

Reviews:

- Code Reviews: Regular peer reviews and walkthroughs will be conducted to ensure adherence to coding standards and design specifications. This involves inspecting code for correctness, readability, and maintainability.
- Design Reviews: At key milestones, design reviews will be carried out to evaluate architectural design conformance with requirements. This ensures that the system architecture aligns with the specified goals and functionalities.

Verification:

- Unit Testing: Individual units of code will be tested to ensure they function correctly in isolation.

- Integration Testing: Different components/modules will be tested together to ensure they work seamlessly.

- Component Testing: Specific components will be tested to verify their functionality.

- Regression Testing: Previous functionality will be retested to ensure new changes have not introduced defects.

- Code Quality Tools: Utilization of code quality tools will enforce coding standards and facilitate the code review process. Version control systems will also be utilized to track changes and manage codebase effectively.

Validation:

- Requirements Validation: Regular communication with the client will ensure accurate capture and validation of requirements throughout the development process. Feedback from the client will be incorporated to ensure alignment with their needs.

- Security Validation: Penetration testing and vulnerability assessments will be conducted to identify and address security vulnerabilities, ensuring the software is secure and resilient against potential threats.

- Reusability and Expandability Validation: Abstraction and well-defined interfaces will enhance code reusability and expandability. The architecture will be designed to accommodate future enhancements and extensions, ensuring scalability and flexibility.

- Performance Validation: Performance testing will be conducted to evaluate system responsiveness, scalability, and resource utilization, ensuring optimal performance under various conditions.

Acceptance Criteria:

- Procedures: User-friendly installation procedures will be provided in documentation, including step-by-step instructions to facilitate system setup.

- Testing: An acceptance test will be conducted to ensure the system meets quality standards and client requirements, in addition to other tests conducted during development.

- Training: User training sessions and materials, such as manuals and tutorials, will be provided to ensure users are proficient in using the system effectively.

- Documentation: Technical documentation including system architecture diagrams and troubleshooting guides will be provided to aid users and administrators in understanding and maintaining the system.

Deployment and Mobility

Personal Reflection

This project challenged us in several ways, as anticipated in the initial deliverable. Integrating the virtual school platform with existing systems required careful coordination. Security and privacy concerns were crucial, and ensuring customization and flexibility while managing scalability and performance was essential. Despite the challenges, we've learned valuable skills in architectural design, API integration, and security testing. We now feel more comfortable and competent as system designers. The course has emphasized important skills like analyzing problems and understanding non-functional requirements.

Despite the course's extensive material and challenging chapters, it was enlightening for us as software engineering students. It greatly aided our comprehension of our field and furnished us with professional system design architecture skills.