Project Proposal

Group: 38

Q1. Name of your product - (Potential Names)

- Peerspace
- Speers
- Study Sync
- Epic Study Buddy Finder Application 360

Q2. Briefly describe the core concept of your application in terms of user functionality.
The app is about providing students with a one-stop platform that integrates features to enhance their academic experience.
Features:

1. Exam date scheduler for personalised schedules
2. Study group finder utilising the given study schedules
3. Mental health tips
4. Course finder/feedback
5. Room finder (to find classes/ available study spaces)
6. Assignment/Grade tracker

Q3. List names of each person in project group

- Adhil Ashraf
- Anfaal Mahbub
- Al Farhana Siddique
- Arvind Krishnaa
- Mehvish Shakeel
- Varshiny Gogulanathan

Q4. List programming languages and web development frameworks that you know or want to use

Front-end:

- Developed using JavaScript and React.js, providing a responsive user interface.
- Includes separate components for features like the exam date scheduler, study group finder, mental health tips, course finder/feedback, and room finder.
- Utilizes HTML/CSS for page structuring and styling.

Back-end:

- Built on Node.js with Express.js for server-side development.
- Utilizes MySQL as the relational database for structured data storage.
- Implements user authentication and authorization using Passport.js.
- Provides a RESTful API using Express.js, allowing communication between front-end and back-end components.
- Containerized using Docker for efficient deployment and scaling.

External Services:

- Integrates with third-party APIs for services such as mental health tips or geolocation services for the room finder feature.

Q5. Provide a rough diagram or description of your system architecture, which project requirements you strive to implement, and how this fulfils the learning goals for this course

System Architecture Overview:

The system architecture for the student platform is designed as a modular, microservices-based application that leverages modern web technologies to provide a comprehensive academic experience for students. It consists of both front-end and back-end components that communicate through a RESTful API. Additionally, real-time features are supported through a Pub/Sub pattern.

Pub/Sub for Real-time Updates:

Utilises a message broker system (e.g., RabbitMQ) for implementing the Pub/Sub pattern. Enables real-time updates and notifications for study groups, notifications, and room availability.

Project Requirements Fulfilled:

Exam Date Scheduler: Implemented as a feature within the front-end and utilises the back-end to personalise schedules for individual students.

Study Group Finder: Realised through the study group finder component, which uses the provided study schedules and supports real-time updates for study group interactions.

Mental Health Tips: Integrated as part of the front-end, offering mental health tips and resources, possibly fetched from third-party APIs.

Course Finder/Feedback: Provided as part of the front-end and back-end components, allowing users to search for courses and provide feedback, with data stored in the MySQL database.

Room Finder: Implemented as a feature utilising external APIs or services for room availability information, and supporting real-time updates through the Pub/Sub pattern.

Alignment with Learning Goals:

Understanding of Software Architecture Models and Documentation: The architecture includes comprehensive documentation, such as component diagrams, interaction diagrams, and deployment diagrams, demonstrating a deep understanding of architectural models.

Understanding of Non-functional Requirements: The architecture considers non-functional requirements like scalability (through microservices and Docker), security, and performance, showcasing practical knowledge of non-functional requirements.

Knowledge of Architectural Styles: The microservices architecture style is applied, demonstrating the ability to select an architectural style fitting to the problem.

Knowledge of Common Software Patterns: The use of the MVC pattern, Singleton, Observer, and Pub/Sub patterns illustrates practical knowledge of common software patterns.

Knowledge of Software Reuse Techniques: Reusable components and libraries for user authentication and authorization showcase knowledge of software reuse techniques.

Effective Use of Development Tools: Implementation of CI/CD pipelines using GitHub Actions or GitLab CI demonstrates effective use of industry-standard development tools.

Architectural Evaluation Techniques: Architectural evaluations are integrated into the development process through code reviews, inspections, and CI/CD quality checks, fulfilling the learning goal of using architectural evaluation techniques.

In summary, the system architecture aligns with the project requirements by providing the desired features for enhancing the academic experience of students. It also aligns with the

learning goals by demonstrating a deep understanding of software architecture models, non-functional requirements, architectural styles, common software patterns, software reuse techniques, effective use of development tools, and the incorporation of architectural evaluation techniques into the development process