SENG 401
DESIGN DOCUMENT
TEAM 38

Authors & Contributors:
- Adhil Ashraf
- Anfaal Mahbub
- Al Farhana Siddique
- Arvind Krishnaa
- Mehvish Shakeel
- Varshiny Gogulanathan

Last updated: February 6, 2024

Status: Draft

Purpose:

The purpose of this design is to enhance the academic experience of students by providing them with a comprehensive platform, StudySync, which integrates various features to streamline their study routines, foster collaboration, and promote mental well-being.

Context:

StudySync aims to address the challenges students face in managing their academic responsibilities, navigating their courses, and maintaining their mental health. Through prior related work and feedback from students, we identified key areas where students could benefit from additional support and tools.

The features incorporated into StudySync, such as the exam date scheduler for personalized schedules, study group finder, mental health tips, course finder/feedback system, and room finder, are all designed to address specific needs and pain points of students.

By offering personalized exam schedules, StudySync helps students organize their study time effectively, reducing stress and anxiety associated with exam preparation. The study group finder promotes collaboration among students, allowing them to study together and share resources. Mental health tips provide valuable support and resources to help students manage stress and prioritize their well-being.

Additionally, the course finder/feedback system helps students explore and select courses based on feedback from peers, contributing to a more informed decision-making process. The room finder feature simplifies the process of locating available study spaces on campus, optimizing students' study environments.

The goal of StudySync's design is to increase improve student satisfaction, academic success, and retention rates, while also being deeply integrated into broader business objectives. To generate revenue, the platform could employ a subscription model, advertising, partnerships, e-commerce, and data insights, ensuring sustainability while enhancing the student experience.

Detailed Design:

Overall System Architecture:
Our StudySync will adopt a microservices-based architecture, providing modularity, scalability, and maintainability. The front-end will be developed using JavaScript and React.js. At the back-end, we will use Node.js with Express.js and MySQL for structured data storage. Real-time updates will be facilitated through a Pub/Sub pattern using a message broker (e.g., RabbitMQ). We also plan to integrate third-party APIs for mental health tips and room availability.

User Interaction:
Users will access StudySync through a web interface. The platform will offer features such as an exam date scheduler, study group finder, mental health tips, course finder/feedback, and room finder. User

authentication and authorization will be implemented using Passport.js.

Software Components Interaction:
The front-end components will communicate with back-end microservices through a RESTful API. Real-time updates will be achieved through the Pub/Sub pattern.

Data Flow:
User inputs and requests will flow from the front-end to the back end for processing and storage in the MySQL database. Real-time data updates, such as study group notifications and room availability, will flow through the message broker to relevant components.

Implementation Plan:

Technologies We Intend to Use:
- Front-end: JavaScript, React.js, HTML/CSS
- Back-end: Node.js, Express.js, MySQL, RabbitMQ (or alternative message broker)
- Containerization: Docker
- Third-party APIs for external services integration

How We Will Manage Source Code:
We will utilize Git and a version control system for source code management. Our repository on a platform like GitHub or GitLab will facilitate collaboration and version tracking.

How We Will Integrate Components:
We plan to implement containerization and orchestration for efficient component integration and scaling. Clear communication between microservices through the RESTful API will ensure seamless functionality.

How We Intend to Test the System:
Our testing approach includes the development and execution of unit tests, integration tests, and end-to-end tests to ensure system functionality and reliability. Continuous integration/continuous deployment (CI/CD) pipelines using tools like GitHub Actions or GitLab CI will automate testing and deployment.

How We Will Deploy the System:
The system will be deployed to production using container orchestration tools such as Docker to ensure scalability and ease of management. A robust deployment process with rollback mechanisms will be in place to handle potential issues.

Project Planning:

Team Organization:
We have divided into two teams: Front-end and Back-end Development

Front-end Development	**Backend Development**
Adhil Ashraf Al Farhana Siddique Mehvish Shakeel	Anfaal Mahbub Arvind Krishnaa Varshiny Gogulanathan

We split evenly between front-end and back-end tasks. Front-end developers focus on client-side functionality, while back-end developers handle server-side logic and database management. We hold weekly meetings to discuss project progress, address any challenges, and coordinate upcoming tasks. Additionally, we use an organized Discord server with channels for tasks, deadlines, and specific project areas like front-end and back-end.

Schedule:
Week 1-2 (Feb 6 - Feb 19): Detailed Design and Setup
Front-end Team:
- Design UI mockups and wireframes (Feb 6 - Feb 10)

  - Task 2: Set up React.js environment (Feb 10 - Feb 12)
Back-end Team:
  - Design system architecture and database schema (Feb 6 - Feb 10)
  - Set up Node.js, Express.js, and MySQL environment (Feb 10 - Feb 12)
Both Teams:
  - Research and select message broker (e.g., RabbitMQ) (Feb 12 - Feb 14)
  - Set up version control system (e.g., Git/GitHub) (Feb 14 - Feb 19)

Week 3-4 (Feb 20 - Mar 4): Front-end and Back-end Development
Front-end Team:
  - Develop UI components for exam date scheduler and study group finder (Feb 20 - Feb 26)
  - Integrate authentication using Passport.js (Feb 27 - Mar 2)
Back-end Team:
  - Implement RESTful API endpoints for exam date scheduler and study group finder (Feb 20 – Feb 26)
  - Set up MySQL database and implement data storage logic (Feb 27 - Mar 2)
Both Teams:
  - Integrate front-end and back-end components for initial functionality testing (Mar 3 - Mar 4)

Week 5-6 (Mar 5 - Mar 18): Feature Implementation and Testing
Front-end Team:
  - Develop UI components for mental health tips and course finder/feedback (Mar 5 - Mar 11)
  - Conduct UI/UX testing and refinement (Mar 12 - Mar 14)
Back-end Team:
  - Implement RESTful API endpoints for mental health tips and course finder/feedback (Mar 5 - Mar 11)
  - Implement message broker integration for real-time updates (Mar 12 - Mar 14)
Both Teams:
  - Integrate all features for comprehensive testing (Mar 15 - Mar 18)

Week 7-8 (Mar 19 - Apr 1): Testing and Deployment Preparation
  - Develop and execute unit tests, integration tests, and end-to-end tests (Mar 19 - Mar 25)
  - Implement CI/CD pipelines for automated testing and deployment (Mar 26 - Mar 29)
  - Document deployment process and rollback mechanisms (Mar 30 - Apr 1)

Week 9 (Apr 2 - Apr 15): Final Testing, Deployment, and Review
  - Conduct final system testing and bug fixing (Apr 2 - Apr 6)
  - Deploy system to production using Docker and orchestration tools (Apr 7 - Apr 11)
  - Review system performance, gather feedback, and iterate if necessary (Apr 12 - Apr 15)

Milestones/Accomplishments:
- System Design and Architecture Planning: Create the system's architecture and design.
- Front-end Development (Exam date scheduler, Study group finder): Develop front-end components for exam scheduling and study group finding.
- Back-end Development (User authentication, Database setup): Implement user authentication and set up the database.
- Front-end Development (Mental health tips, Course finder/feedback): Develop front-end components for mental health tips and course-related features.
- Back-end Development (RESTful API, Pub/Sub integration): Implement the back-end components including RESTful API and Pub/Sub integration.
- Front-end Development (Room finder): Develop front-end components for finding available rooms.
- Integration with Third-party APIs (Mental health tips, Room availability): Integrate third-party APIs for mental health tips and room availability.
- Testing and Quality Assurance (Unit, integration, and end-to-end testing): Conduct comprehensive testing to ensure system functionality and quality.
- Deployment and Scaling (Containerization and deployment to production): Deploy the system using containerization and prepare for scaling.

Appendix:

- [No artifacts, diagrams, or discussion notes for the moment]