



Knight Foundation School of Computing and Information Science

Fall 2024 Senior Design Project

StudyQuest

Students: Richard Aguado, Markus Berrios, Alfredo Medina, Christopher Schur, Cesar Vargas
Instructor/Faculty: *Masoud Sadjadi*, Florida International University

PROBLEM

The goal is to create a study platform that maintains a clear focus on the provided topic, offering a robust framework for instant messaging and engaging mechanics. It should include tools that enable users to effectively communicate ideas and opinions while ensuring data is stored and accessed securely, efficiently, and quickly. Additionally, the platform should provide flexible options for managing created rooms and addressing challenges.

CURRENT SYSTEM

Modern messaging apps are typically designed to be subject-agnostic, catering to a wide variety of user groups for broader usability. Communities often bear the responsibility of developing niche, subject-specific features (when feasible) or waiting for developers to implement features based on community requests. This approach allows messaging platforms to remain flexible and adaptable, but it can delay the availability of specialized tools for specific user needs.

REQUIREMENTS

- Functional Requirements:**
- Support for creating and managing study chat rooms
 - Collaboration tools tailored for study sessions
 - Gamified participation features:
 - Points system
 - Challenges
 - Head-to-head competitions
- Non-functional Requirements:**
- Smooth room transitions
 - Real time chat updates
 - A responsive interface

SYSTEM DESIGN

- The system is architected to support a scalable and efficient study messaging platform using a Next.js frontend and Firestore database. Key components include:
- **Frontend:** Built with Next.js to provide a responsive and dynamic UI.
 - **Backend:** Utilizes Firestore for real-time database operations and Firebase authentication for user management.
 - **Core Modules:**
 - Chat Room Management (creation, modification, deletion).
 - Gamification logic for points, challenges, and competitions.
 - Real-time updates for seamless user experiences.

OBJECT DESIGN

- Objects are designed to reflect the key entities and their interactions. Primary objects include:
- **User:**
 - Represents an individual using the platform. This object manages user-specific data, such as personal details, preferences, and progress within the platform. It also facilitates interaction with chat rooms, gamified activities, and other users.
 - **ChatRoom:**
 - A virtual space where users can communicate and collaborate. The ChatRoom object handles the creation and management of rooms, tracks active participants, stores messages, and integrates gamified mechanics to enhance engagement.

IMPLEMENTATION

- Frontend Development:**
- Set up Next.js project structure and integrate CSS for styling.
 - Develop components for chat rooms, user profiles, and gamified features.
- Database Configuration:**
- Structure Firestore collections (e.g., Users, ChatRooms, Challenges).
 - Optimize queries for real-time performance.
- Authentication:**
- Implement Firebase Authentication for secure sign-up/login.
- Git Workflow:**
- Use GitHub for version control, with branching strategies for feature development and bug fixes.

VERIFICATION

- Testing ensures that the system meets functional and non-functional requirements:
- **Unit Testing:** Validate individual components, such as chat room creation or gamified point calculation.
 - **Integration Testing:** Verify data flow between frontend and Firestore.
 - **User Acceptance Testing:** Gather feedback on UI responsiveness, smooth room transitions, and real-time updates.
 - **Performance Testing:** Measure response times and scalability under high user loads.

SUMMARY

The platform combines the dynamic capabilities of Next.js with Firestore’s real-time features to deliver an interactive study messaging tool. Using GitHub for version control and Visual Studio Code as the IDE ensures streamlined collaboration and development. The design supports future scalability, enabling the platform to adapt to growing user demands and feature enhancements.

REFERENCES

- 1. Firebase Documentation:**
<https://firebase.google.com/docs>
- 2. Get to know Cloud Firestore:**
https://www.youtube.com/watch?v=v_hR4K4auoQ&list=PLI-K7zZEsYLLuG5MCVEzXAQ7ACZBCuZgZ
- 3. Next.js Documentation:**
<https://nextjs.org/docs>
- 4. Next.js crash course:**
<https://www.youtube.com/watch?v=ZVnjOPwW4ZA>