San Francisco State University
CSC 648 - 04
Milestone 2
Section 04 Team 06 [Check With Manuel]
StudyBuddy
03/20/2024

Team Members

Ashley Ching	Team Leader
Yuquan Xu	Front End Lead
Kent Nguyen	Back End Lead
Nhan Nguyen	Git Master
Brenden Lapuz	Scrum Master
Pierre Harbin	Team Member

History Revision Table

Version	**Date:**	**Notes:**

Data definition V2

Our choice of database for the project is MongoDB. A group of entities in a non-relational database is known as a collection and each entity is known as a document. We have three different types of collections: users, groups, and messages. Their schemas are listed below:

1. Users
    a. Username
    b. Password
    c. Email
    d. Tags
    e. Friends
    f. History
    g. isVerified
    h. isGuess
    i. ID
2. Groups
    a. Name
    b. Members
    c. ID
3. Messages
    a. Participants
    b. Contents
    c. ID

Each user document holds information about that specific user. Each group will have a group name, a list of group members, and the group's ID. Each message will have a list of participants, their chat content, and message ID.

Initial List of Functional Requirements V2

- R.1 - User Management
    - R.1.1 - Users should be allowed to create an account using an email/ password
    - R.1.1 - Users should be allowed to log into their account using an email/ password
    - R.1.1 - Users should be able to use the site without creating an account
    - R.1.1 - Users should be able to create and edit personal profiles, preferred study methods, and interests like school, major, or class subjects.
    - R.1.2 - Users should be verified through the school email verification system to confirm users' affiliations.
- R.2 - Social Management
    - R.2.1 - Users should be able to view the list of other users and public groups.
    - R.2.1 - Users should be able to chat with other users during study sessions
    - R.2.1 - Users should be able to chat with other users outside of study sessions through direct messaging
    - R.2.2 - Users should be able to send a group message that each member of a study group will receive
    - R.2.2 - Registered users should be able to create private and public groups for study sessions
    - R.2.2 - Registered users should be able to add people to study groups and save these groups
    - R.2.2 - Users should be able to schedule and join public study sessions
    - R.2.2 - Registered users should be able to schedule and join private study sessions
    - R.2.3 - Users should be able to integrate sessions with external calendars
- R.3 - Discovery Management
    - R.3.1 - Users should be able to search for study groups or sessions based on filters like school, major, or class subjects
    - R.3.1 - Users should be able to tag and categorize their study preferences on their profile for better matchmaking with study groups or sessions.
- R.4 - Safety and Privacy Management
    - R.4.2 - Users should be able to report inappropriate behavior or content
    - R.4.1 - Users should have the option to block/ unblock other users to maintain personal safety and privacy
        - ■
- R.5 - Future Features/ Functionality

- R.5.3 - Users should be able to share images and documents within group or user discussions
- R.5.3 - Users should have access to live video function for real-time study sessions
    - Video function could be done through Zoom
    - Perhaps a Zoom room would be created through Study Buddy
- R.5.1 - Users should receive some form of verification badge or status upon successful email verification, giving access to special features
- R.5.3 - Users should be able to review and rate their experience with study sessions or groups, contributing to a community reputation system.

List of Non-Functional Requirements V2

Performance:

- Response Time: Ensure the application responds to user interactions within 5 seconds under normal usage conditions.

Usability:

- Intuitive Design: The interface should be user-friendly, allowing new users to navigate and use the app without requiring a tutorial.
    - This means easy to read pages, all the options and menus being displayed in a clear way
    - Properly color coordinated text and backgrounds
    - Default zoom level will have the text large enough for most people to read clearly
    - Smooth UX flow so users don’t go in wild goose changes doing an action on the site
- Accessibility: Aim for basic accessibility features, such as clear labels for buttons and the ability to zoom text.

Storage:

- Efficient Use of Space: Optimize storage of user data, ensuring the database does not exceed 1GB under the project's scope.

Compatibility:

- Device Support: The application should work seamlessly on the latest versions of Android and iOS for mobile, and Chrome and Firefox for web.
- Responsive Design: Ensure the web interface adjusts to different screen sizes, from mobile phones to desktop monitors.

Security:

- Password Protection: Implement hashed passwords and basic encryption for user data storage.
- Data Privacy: Ensure user data is not shared without consent, adhering to basic data protection principles.

Code Management:

- Version Control: Use Git for version control, with a simple branching model that includes a development branch and feature branches.
- Code Reviews: Encourage peer code reviews before merging feature branches to the development branch.

Scalability:

- Simplicity Over Scale: Design the application to support up to 100 concurrent users, focusing on functionality over scalability for the project's scope.

Reliability:

- Basic Error Handling: Implement error handling to provide users with clear messages in case something goes wrong.

  Maintainability:
- Code Documentation: Document major functions and modules within the codebase to facilitate understanding and future modifications.

UI/UX Flow

The UX flow is useful because it gives a general design of how the final website would function. The

The UX flow is usable because it connects all the needed pages together to show how a visitor of the page would navigate through the web page.

The UX flow is desirable because it is simple to use and intuitive.

The UX flow is accessible because each button performs a unique action with no additional actions.

The UX flow is findable because the buttons are big and placed in common locations. There are many ways to go to the same page and it is consistent throughout the pages.

The UX flow is credible because it shows that the user is in control with who they are friends with and who they group up with. With school email verification, it will allow users to trust people who have the school emails go to that school.

High Level Architecture, Database Organization

DB Organization:

- Our choice of database for the project is MongoDB. A group of entities in a non-relational database is known as a collection and each entity is known as a document. We have three different types of collections: users, groups, and messages. Their schemas are listed below:

1. Users
    a. Username
    b. Password
    c. Email
    d. Tags
    e. Friends
    f. History
    g. isVerified
    h. isGuess
    i. ID
2. Groups
    a. Name
    b. Members
    c. ID
3. Messages
    a. Participants
    b. Contents
    c. ID

Each user document holds information about that specific user. Each group will have a group name, a list of group members, and the group's ID. Each message will have a list of participants, their chat content, and message ID.

Add/Delete/Search architecture:

- When a user registers, it creates a user document in the table. Since we have guest users, when these users are on, a user document is still created but won't have the things verified accounts have. Guest users are deleted after a specified amount of time.
- The time when the DB is searched is when the user logs in or looks at the history
- Another time is is when the user is looking through filters and trying to find people to study with

APIs

- Major endpoints: register, login, dashboard
- Register endpoint - post request where it receives form data from the front end page
- Login endpoint - also a post request
- Dashboard endpoint - get request
- Message - handled by socket io, handles real time chat
- Axios - handles the get and post request between front and back end
- Decrypt - meant to encrypt passwords (No front-end or back-end communication)
- Mongoose - library that handles MongoDB (specifically for back-end)

Key Risks

- Skills Risks and Mitigation Plan
    - Risk: Team members may lack expertise in specific technologies used in the project (Express, OCI,...).
    - Mitigation: Implement a continuous learning plan, including online tutorials, coding workshops, and regular knowledge-sharing sessions among team members. Utilize pair programming for complex tasks to spread knowledge and expertise.
- Schedule Risks
    - Risk: Missed deadlines due to underestimation of tasks or unforeseen complications.
    - Mitigation: Utilize Agile methodologies to allow for flexible scheduling and regular reassessment of tasks and priorities. Implement a project management tool (Jira) for transparent task tracking and updates. Schedule weekly sprints to assess progress and adjust tasks as needed.
- Teamwork Risks
    - Risk: Inconsistent participation in meetings or uneven workload distribution among team members.
    - Mitigation: Establish clear roles and responsibilities from the outset. Hold regular team meetings to ensure everyone is on track and any issues are promptly addressed. Implement a buddy system to ensure no team member falls behind and to foster a supportive team environment.
- Legal/Content Risks
    - Risk: Utilizing software or content without the proper licensing or violating copyright laws.
    - Mitigation: Conduct thorough research on the licensing of all software and third-party content used in the project. Opt for open-source solutions with permissible licenses when possible. Consult legal advice for any uncertain situations to ensure compliance.
- Key to Risk Resolution
    - The key to resolving these risks is proactive management and open communication. By identifying potential issues early on and implementing the mitigation strategies outlined, the team can navigate challenges more effectively.
    - Regular team meetings and the use of project management tools will ensure transparency and accountability, allowing for timely adjustments to the project plan as necessary.
    - Encouraging a culture of continuous learning and mutual support will help in overcoming skill gaps and ensuring project success.

Project Management

Our team has a Google Drive folder that is shared with all group members. Inside this Google Drive folder, we have different milestone folders. Inside milestone folders, we have Google Docs for the milestone document as well as study documents and anything else that we would need to share. The scrum meeting notes is on Google Docs shared in the Google Drive we all have access to.

We use Jira as a tool to manage each member's tasks to make it transparent on who is working on what. We will move tasks into different categories (eg: In progress, Done, Reviewing) as we proceed with our work to help everyone on the team understand where we are at in the process of finishing.