Software Requirements Specification

Maths Club SAT

Version 1.0.3 approved

Prepared by Garv Shah

Software Development 3/4

31/5/202

Table of Contents

Revision History

Name	Date	Reason For Changes	Version
Link Update	15/6/22	Outdated links in references	1.0.1
Use Case Explanation	16/6/22	Use Case Diagram could be clarified	1.0.2
Formatting	17/6/22	Formatting was not consistent	1.0.3

1. Introduction

1.1 Purpose

The purpose of this SRS is to create an app for the CGS Maths Club that will be able to make it easier to access questions, easier to create questions, and more fun to answer them. These three main objectives are what each feature should connect to, allowing for better community interaction and making the process of running and being part of the Maths Club more streamlined.

1.2 Intended Audience

The intended audience for this solution would be members of the CGS Maths Club, that allows for an archive of posts and problems which students can comment on, collaborate on, and discuss with each other. Namely, the app has two target clients, the general users who submit and view quizzes/posts, and the content creators/administrators who manage users and content on the app. The experience for both these types of users should be streamlined.

1.3 Product Scope

1.3.1 In Scope

Search Engine

- Search for certain sections or individual quizzes/posts
- Ability to open and enter different parts of the app from the search engine
- Relatively fast with increasing size

Quiz System (Answer Online)

- Have posts that can remain as quizzes for a certain period and then get archived
- While posts are an active quiz, members will be able to do the quiz and get points for questions answered
- Solutions only to be released after the quiz period
- Members should be able to answer questions within the quiz interface
- Should be able to differentiate between different math being equal or the same

Divided Section for Junior/Senior etc

- The GUI should be able to provide easy separation for sections such as Junior and Senior, or different filters
- There will be a "group" system, where you can create different groups and assign users to them, which gives them certain access to different parts of the app. These may include "admin", "junior" or "senior"

Leaderboards

- There should be a leaderboards page, where points from quizzes can be viewed
- Clients should be able to click a button to automatically scroll to their leaderboard position
- Badge System: members should be able to get badges that will be manually assigned

Optional Hints

- Question curators should be able to optionally provide hints that users view

Forums/Ability to Comment on Posts

- Members should be able to comment on posts, facilitating community interaction
- Members who are not approved (i.e., people not part of the CGS community) should not be able to post messages (dependant on "group" system)

1.3.2 Out of Scope

- Ability to search for other users
- Ability to message users
- Ability to get notifications for posts (would be nice as an extra, not necessary at all for a working system)
- Ability to scan paper answers and have them be processed by the app

2. Overall Description

2.1 Constraints

- **Time:** must have the design finished by the end of Term 2; product by the end of Term 3
- **Cost:** Won't be able to spend money on the app, so there should be no fees despite scaling
- **Hardware:** Ideally, the developer won't be running their own server for this, as this produces un unnecessary reliability and would significantly decrease the usability of the project especially in this case, where the project should continue well after developer involvement ceases.
- **Software:** everything must be created and be possible in Flutter, for at least MacOS, iOS, Android, and the Web. Windows and Linux are not priorities, though it would be nice if they were supported.
- **Social:** the app must not be big and complicated, because then it would be overwhelming for the average maths club member. The design and UX/UI must stay simple and will not be able to go overboard.
- **Legal:** privacy laws must be respected, namely in a school environment when this app is available to the public, so it may be an issue if a school endorses this platform, and anyone can contact school students
- **Ability:** the app is constrained by the developer's ability to program, so ideas must stay within what can feasibly be done by the developer's ability and/or expertise

2.2 Product Functions

2.2.1 Functional Requirements

Search Engine

- Search for certain sections or individual quizzes/posts
- Ability to open and enter different parts of the app from the search engine

Quiz System (Answer Online)

- Have posts that can remain as quizzes for a certain period and then get archived
- While posts are an active quiz, members will be able to do the quiz and get points for questions answered
- Solutions only to be released after the quiz period
- Members should be able to answer questions within the quiz interface
- Should be able to differentiate between different math being equal or the same

Divided Section for Junior/Senior etc

- There will be a "group" system, where you can create different groups and assign users to them, granting permissions.

Leaderboards

- There should be a leaderboards page, where points from quizzes can be viewed
- Clients should be able to click a button to automatically scroll to their leaderboard position
- Badge System: members should be able to get badges that will be manually assigned

Optional Hints

- Question curators should be able to optionally provide hints that users view

Forums/Ability to Comment on Posts

- Members should be able to comment on posts, facilitating community interaction
- Non-approved members should not be able to comment

2.2.2 Non-Functional Requirements

Usability
- Search Engine should remain relatively fast with increasing size
- Search Engine should make it easier to navigate the app and get to where you want to
- Quiz System should feel fair, promoting a healthy competition, and be easy to use
- The GUI should be able to provide easy separation for sections such as Junior and Senior, which should feel intuitive
- The leaderboards must include a sense of competition that encourages people to partake in Maths Club more often

Reliability
- Should reliably be able to handle large file sizes without limits or expiration

Portability
- The software solution should work on most of the intended audience's devices

Robustness
- The system should be able to handle invalid account creation or user input in the quizzes

Maintainability
- The software solution should have enough documentation and tests written such that maintaining the software after it is written does not require rewriting parts of the software

Security
- The forums should protect and consider the privacy of school students

2.3 Stack

For the app, the intention is to use the F^3 stack: Firebase, Flutter and Fastlane.

Along with this, Notion will be used as a CMS for content, combined with Firestore. This combination should provide the scalability needed (with unlimited file storage and a manageable content backend) combined with the security and flexibility of Firebase.

Flutter will be the frontend, a cross-platform framework that can develop iOS, MacOS, Web and Android apps, along with many more. This makes portability ideal, and being made by Google, the platform has a good integration with Firebase, providing a suitable interop with the BaaS.

Firebase will be the backend, along with the auth and analytics. Firebase acts as a Swiss Army knife, so most of what is needed from the backend should be able to be done by Firebase, apart from file storage, since this would get very expensive very fast with the service. Otherwise, it will be used for live updates, an authentication workflow and profile pictures.

Finally, Fastlane is an app deployment platform which will allow for CI/CD into the project, ideal for the scrum/agile, with continuous iterations based on client feedback. Fastlane should automate a few mundane jobs and make things a lot faster so clients can get updates faster.

In summary, my stack will be F^3 + Notion

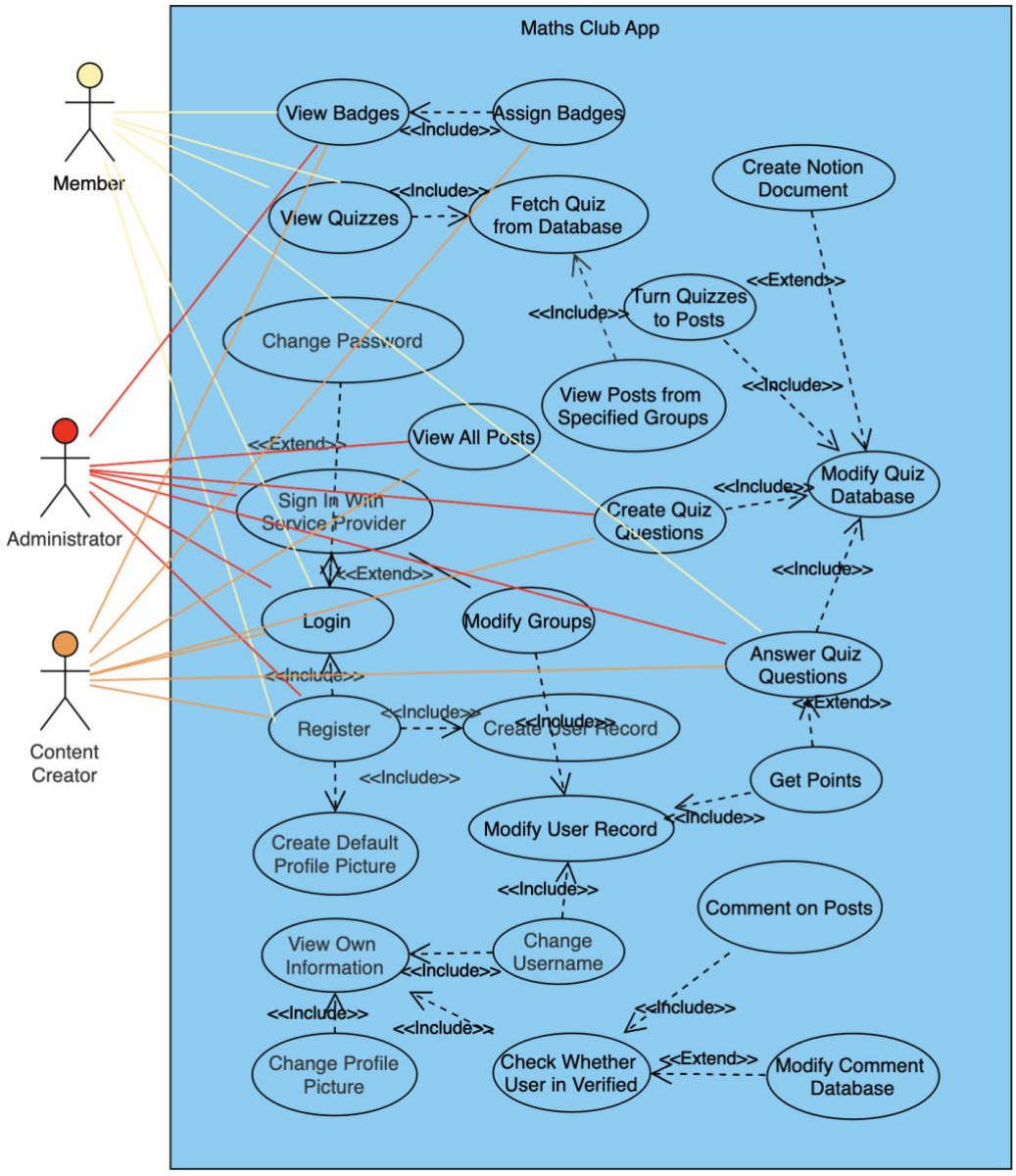
2.4 Operating Environment

The solution will be implemented in the following environments:
- Website
- MacOS
- iOS/iPadOS
- Android

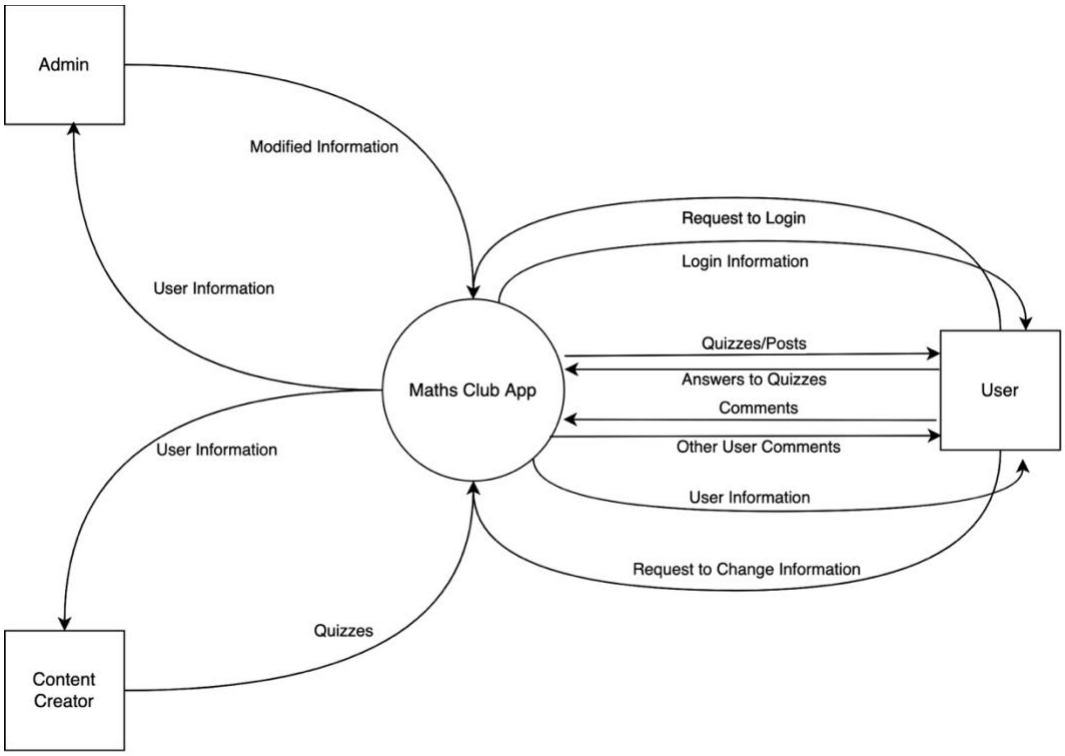
3. Analysis

3.1 Use Case Diagram

The Maths Club system combines various smaller subsystems into one larger application. As such, the Use Case Diagram below is quite complicated, as it includes the interactions of 3 different systems into one. The DFDs have proven to be more useful, as in essence, they provide zoomed in views of each of the systems, showing isolated data flow in comparison to use cases. Due to the nature of the app, it has been deemed that though the below Use Case Diagram is complicated and doesn't provide a view of individual functions, it provides a broad overarching view of the application and mostly serves as the basis for interactions between systems, that the more detailed views of DFDs do not provide. Each system is separate but unified, as the diagram shows.

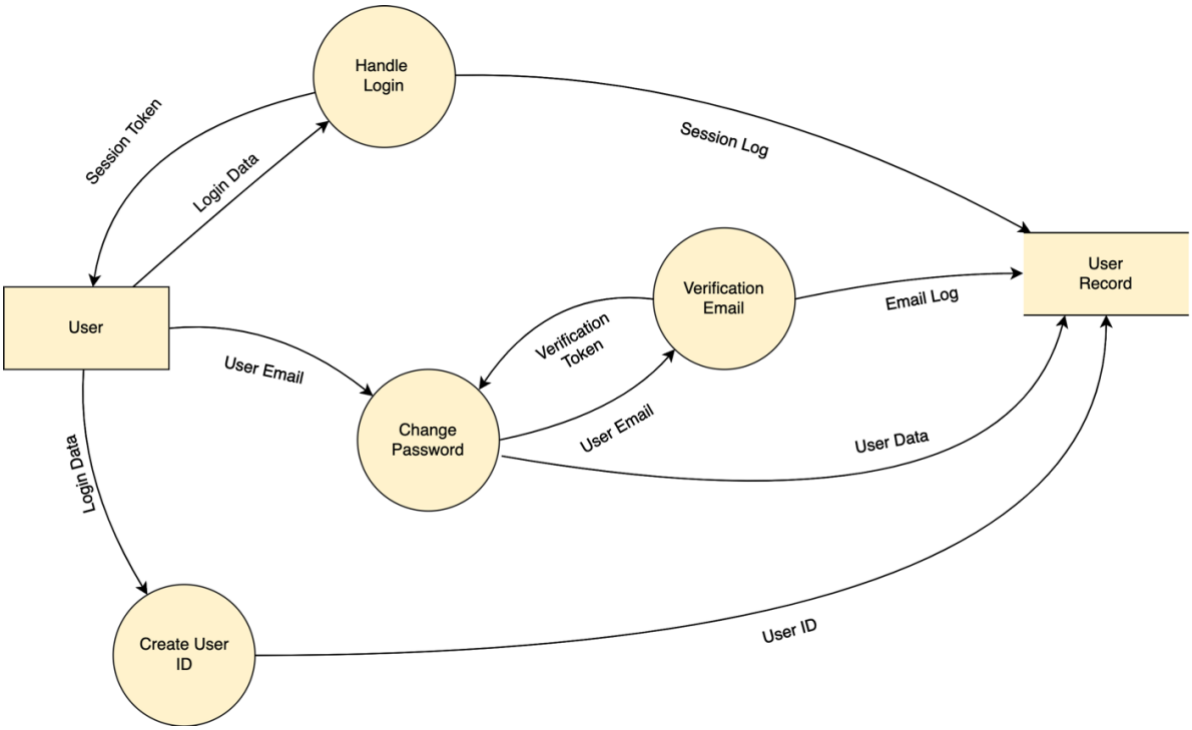


3.2 Context Diagram

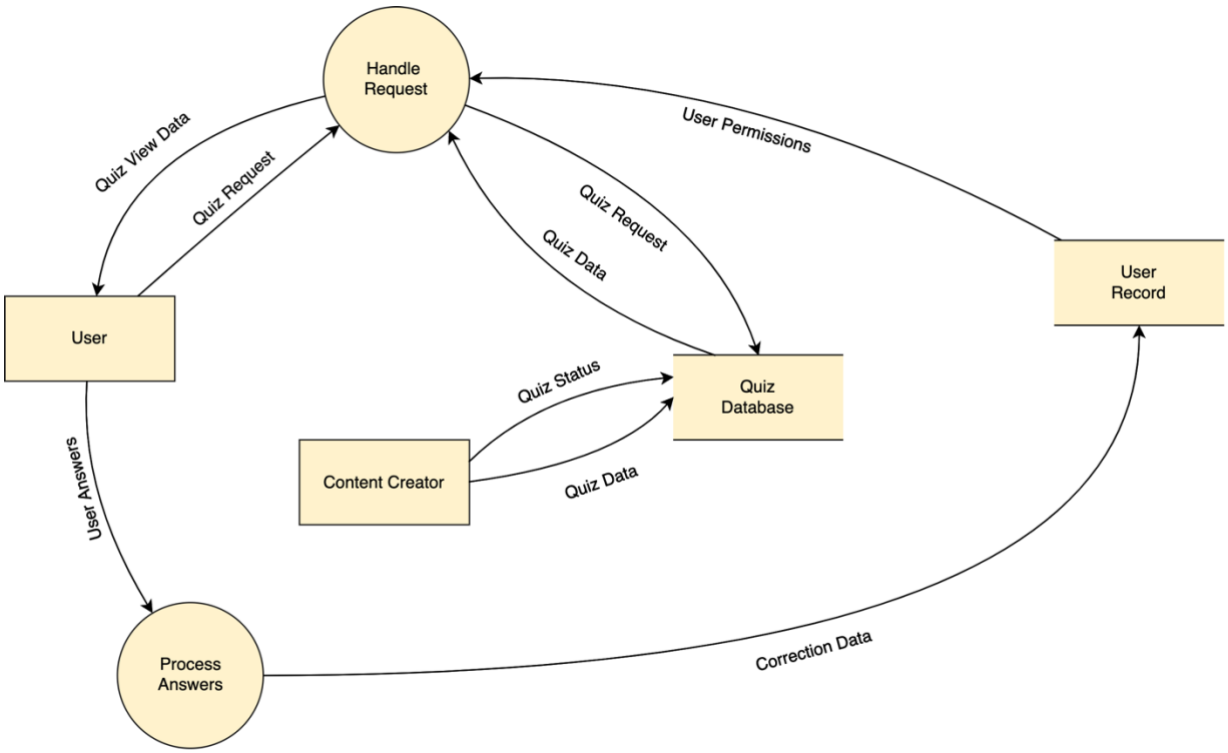


3.3 DFDs

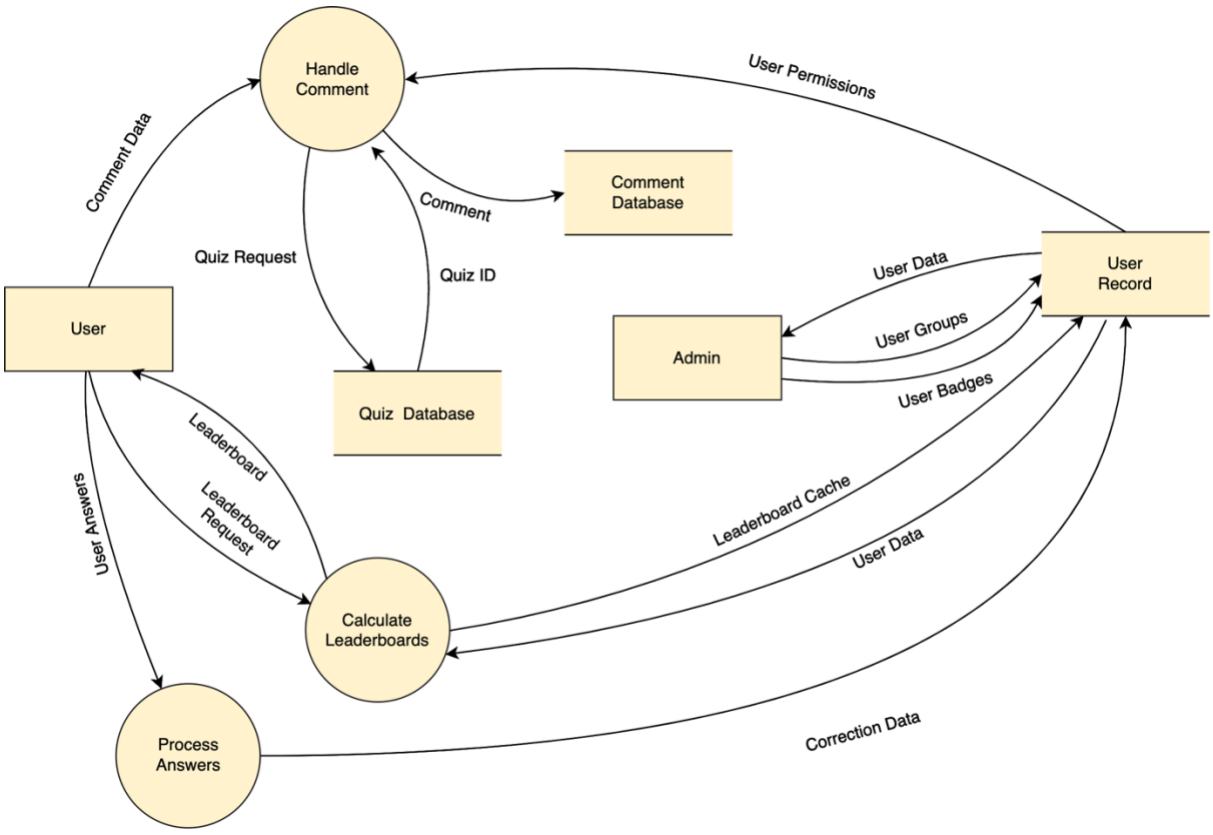
3.3.1 Login System



3.3.2 Quiz System



3.3.3 Misc. Systems



Appendix

3.4 References

This document refers to https://github.com/garv-shah/software-dev, as this address is where a variety of work on the project exists, including interviews, ideas, and documentation. https://garv-shah.vercel.app is also where project updates live, under the "Software Development 3&4" tag.