System Design for Flick Picker

Team 7, 7eam
Talha Asif - asift
Jarrod Colwell - colwellj
Madhi Nagarajan - nagarajm
Andrew Carvalino - carvalia
Ali Tabar - sahraeia

April 5, 2023

1 Revision History

Date	Version	Notes
January 18	0.1	Added content to section 6.4, some potential content to 6.1 - Jarrod
January 18	0.2	Completed Sections 3, 4, 6.1, 6.2, 6.4 - Jarrod
January 18	0.3	Completed Sections 5, 6.3, 8, 12 - Madhi
January 18	0.4	Editing and adding to reflection - All
April 5	1.0	Minor edits to diagrams, content - Madhi

2 Reference Material

This section records information for easy reference.

2.1 Abbreviations and Acronyms

symbol	description
Flick Picker	A program to find Movies, TV Shows, or Anime for individuals or groups
UI	User Interface
MG	Module Guide
MIS	Module Interface Specification
API	Application Programming Interface

Contents

List of Tables

List of Figures

3 Introduction

In order to continue with the implementation of Flick Picker, it is important to create a System Design document. This will help in outlining the design decisions that were made so as to not cause confusion and increased work in the implementation process. It is also important to outline the tasks, both in regard to the work that needs to be done and with whom the work has been assigned. This document, including these design decisions and timeline, will be complimented by the MG and the MIS to further outline the specifications of Flick Picker.

4 Purpose

The purpose of this design documentation is to cover the behaviour of our system under a variety of conditions, detail the components of the system, discuss any connections between design choices and the requirements, user interface designs, and a timeline of when tasks should be complete and who should be completing them. Information in this design document should support or further explain the information found in the MG and MIS.

5 Scope

The system will allow the user to interact with the application, in order to find appropriate Movie/Anime recommendations for themselves and/or their group. The system will also allow the user to securely login and manage their user profile and their groups. External APIs are also made use of to retrieve movie and anime data.

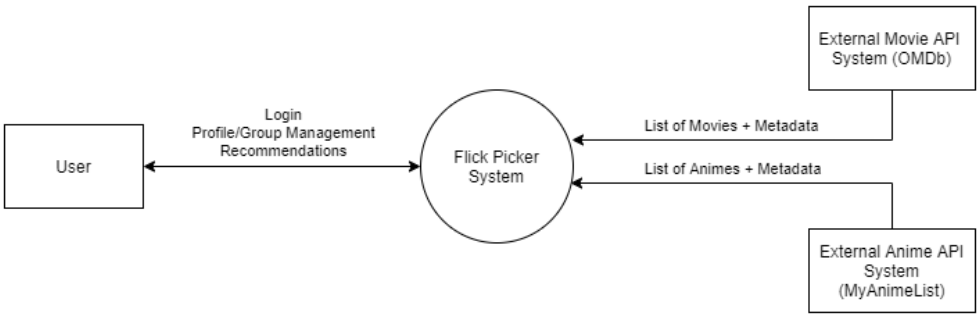

Figure 1: System Context of the entire system and its environment

6 Project Overview

6.1 Normal Behaviour

- **Native Login Module:** Provided the email and password, this module will communicate with the database module to authorize or deny the user's login.
- **OAuth Login Module:** Provided the authorization from an OAuth login, this module will communicate with the database module to authorize the user's login or account creation.
- **Database Module:** Communicates with the database to verify logins, store Movie, TV Show, and Anime data, and to retrieve user data.
- **Profile Module:** Communicates with the Database module to get a user's information to provide to the UI. Allows for users to add friends and create or join groups.
- **Friends Module:** Allows for users to see their friends, accept friend invites, and invite their friends to groups.
- **Group Module:** Compiles preferences of users and sends this information to the Matching Algorithm Module. Presents the received recommendations to the users of the group.
- **Matching Algorithm Module:** Receives the set of preferences and queries the API Module using these preferences to create a set of Movies, TV Shows, and/or Anime recommendations. Sends this set of recommendations back to the group module.
- **API Module:** Queries external APIs daily for the most popular Movies, TV Shows, and Anime. Sends this information to the database. Receives query calls from the Matching Algorithm Module and returns information to it.

6.2 Undesired Event Handling

- **API Module - No Match Found:** If the Matching Algorithm queries the API Module and it cannot find Movies, TV Shows, or Anime that match the filters, the API module will send requests to the external APIs with these filters.
- **Various Modules - Rate Limiting:** If an individual attempts to overload the system with certain requests, they will be timed out via rate limiting.

6.3 Component Diagram

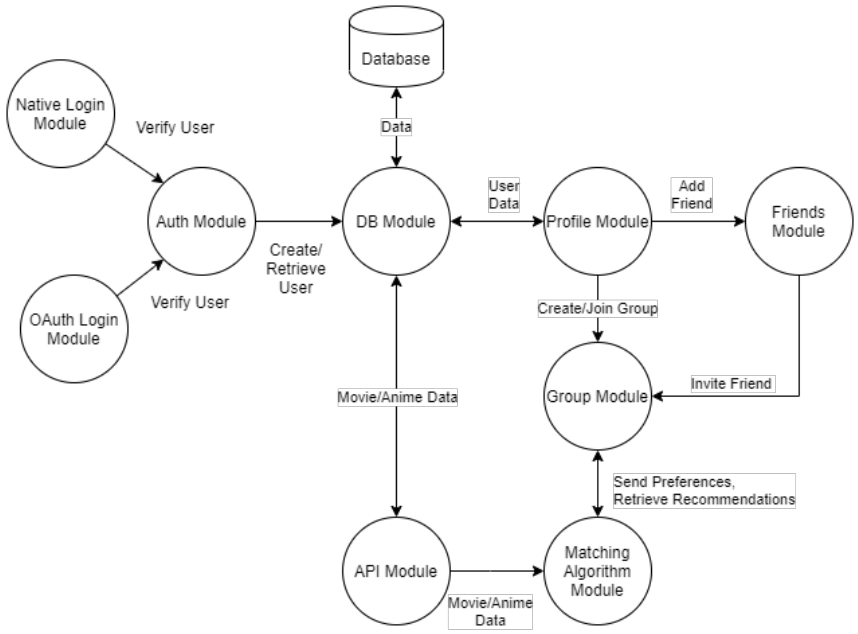

Figure 2: Component Diagram showing all modules

6.4 Connection Between Requirements and Design

6.4.1 Connection Between Authentication Requirements and Design

For email and password authentication, the user's email and a hashed form of their password are stored in the database. Upon login request, the inputted email and password will be compared to the pairs in the database and the login will either succeed or fail depending on whether a match is found.

Google OAuth will provide authorization and an email to Flick Picker which will have / create an account with this email. If an account with this email already exists using OAuth, the user is logged in successfully. If an account using the previous method (email and password) exists, the user will be prompted to login using their email and password as above. If an account with this email does not exist, the user will be logged in and a new OAuth account will be created. They will then be brought to the new user page (creation of username, preferences, etc.).

Upon clicking the logout button, the user will be brought to the login screen and their token is removed.

6.4.2 Connection Between Profile/Group Requirements and Design

There will be a page for users to modify their information, including username and password if the account is not an OAuth account. Additionally, there will be a page for

users to access and modify their preferences (e.g. genre, show type, etc.). A separate page will enable users to create and join groups. Group invites will be provided through the invited user's email or friends page. The owner of a group can view the group members, join requests, and other information regarding group management on the group management page.

6.4.3 Connection Between Recommendation Requirements and Design

Flick Picker will retrieve information about the top 3000 most popular Movies, TV Shows, and Anime and store that information for quick access. This information will be used in the vast majority of all recommendations. If a recommendation cannot be found within these Movies, TV Shows, or Anime, additional queries will be sent to the relevant APIs to find a recommendation.

During the recommendation process for a group, individual users reflect their desire to watch a recommendation using the 'like', 'neutral', or 'dislike' buttons. This information will be stored alongside their preferences to aid in future recommendations for both the group and the individuals.

Given user permission, Flick Picker will send emails to all members in a group once a recommendation has been chosen.

7 System Variables

7.1 Monitored Variables

N/A

7.2 Controlled Variables

N/A

7.3 Constants Variables

N/A

8 User Interfaces

Mockups of the user interface were designed in Figma. Below are the mockups, organized by the module they fall under.

Figure 3: UI mockups of the Login Module. Note that both the Native Login Module and Auth Module are represented in the Login Page

Figure 4: UI mockups of the Profile & Friend Modules

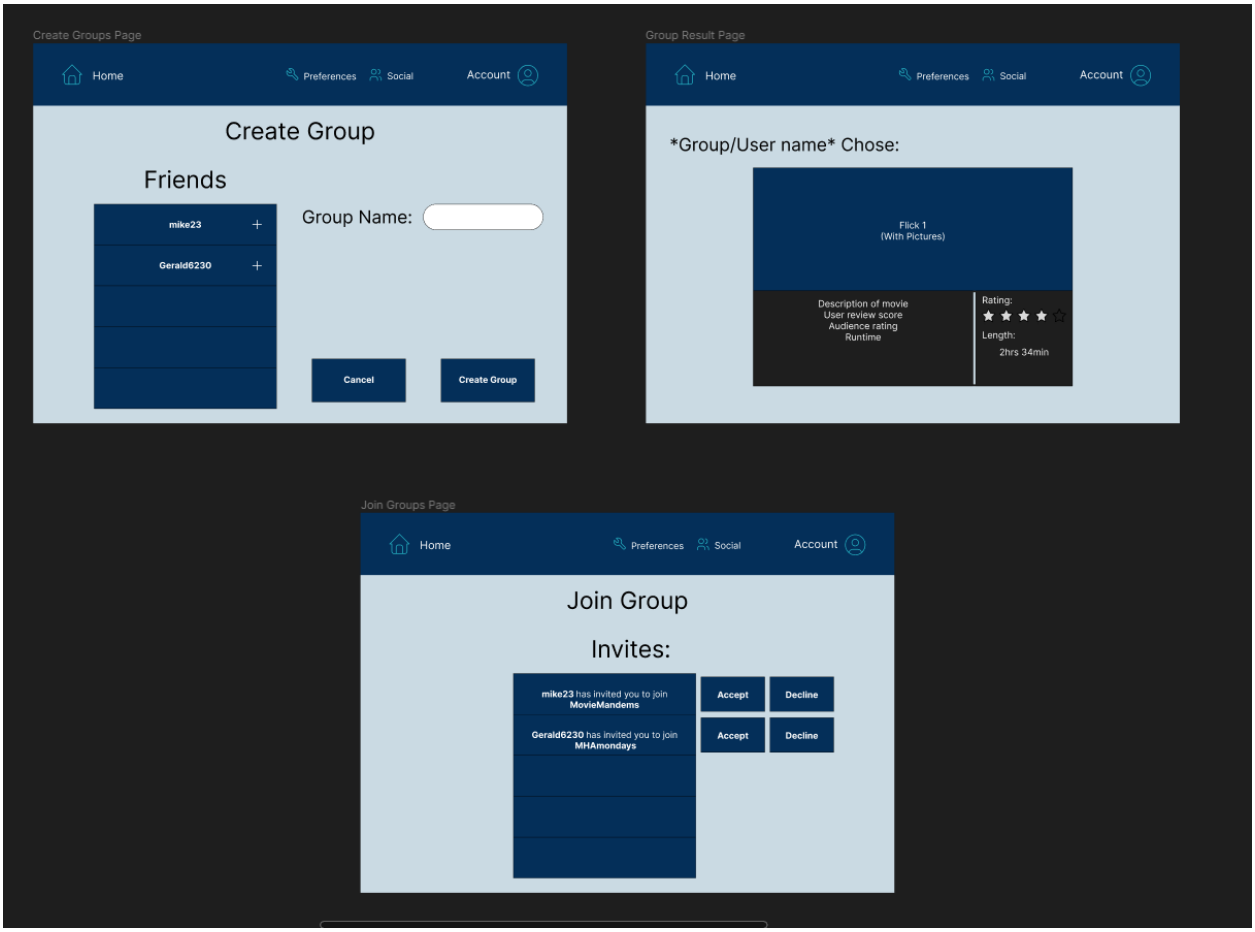

Figure 5: UI mockups of the Group Module

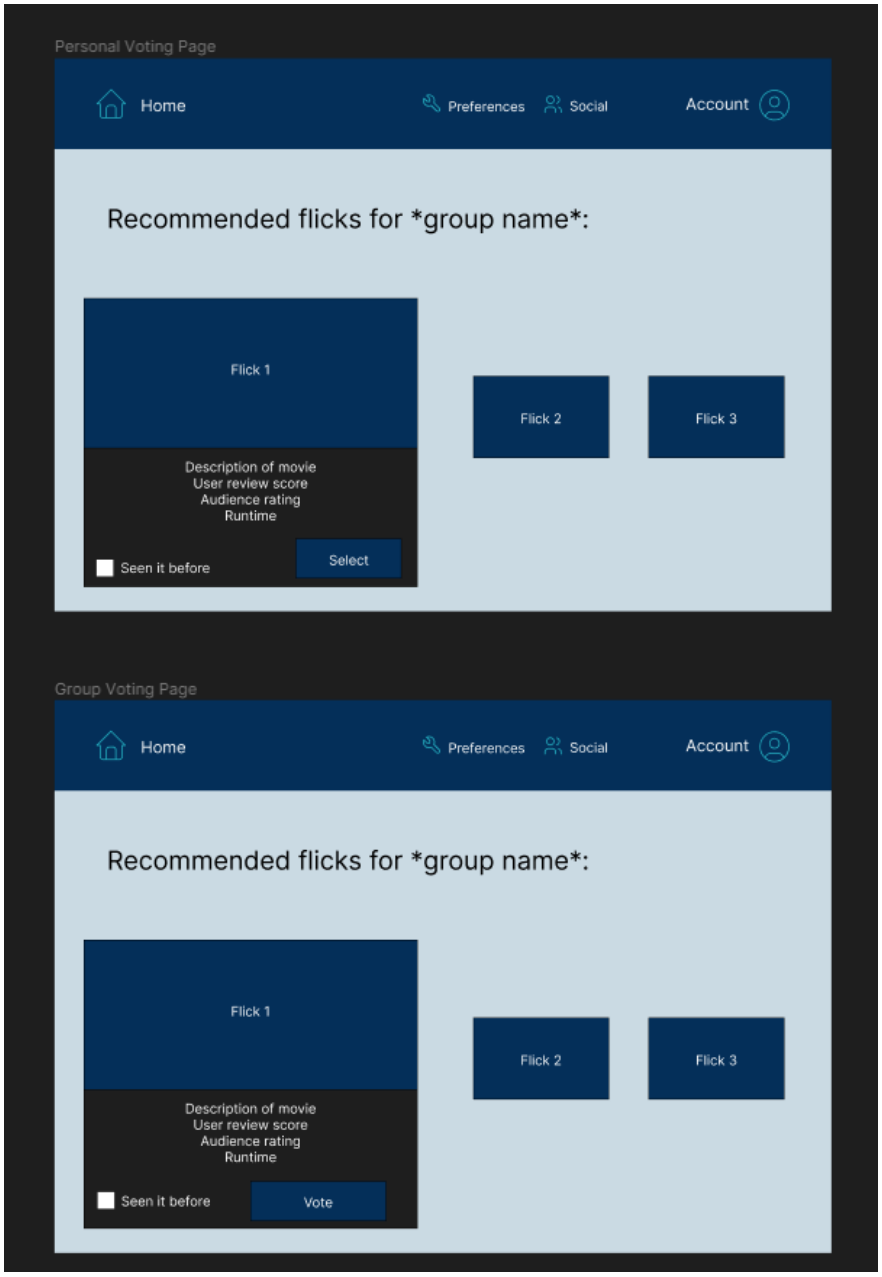

Figure 6: UI mockups of the Matching Algorithm Module

9 Design of Hardware

N/A

10 Design of Electrical Components

N/A

11 Design of Communication Protocols

N/A

12 Timeline

Task	**Module**	**Task Due Date**	**Assigned To**
Create Login & Password Reset Pages	Login (Native & Auth)	Jan 23	Talha
Integrate External APIs to backend	API	Jan 23	Madhi
Setup Firebase DB	API, DB	Jan 23	Jarrod
Create Home & Profile Pages	Profile	Jan 25	Andrew
Create Preference Pages	Profile	Jan 25	Ali
Add native login functionality	Native Login	Jan 30	Jarrod
Add Auth functionality	Auth Login	Jan 30	Madhi
Create Groups Page	Group	Feb 2	Ali
Create Friends Page	Friend	Feb 2	Andrew
Create Recommendation Pages	Friend	Feb 3	Talha
Hookup Backend to Frontend	All	Feb 5	Talha, Jarrod
Deploy Frontend and Backend	All	Feb 12	Madhi
Manual Testing	All	continuous	Team
Integration Tests	All	Feb 15	Team

A Interface

[Include additional information related to the appearance of, and interaction with, the user interface —SS]

B Mechanical Hardware

N/A

C Electrical Components

N/A

D Communication Protocols

N/A

E Reflection

The information in this section will be used to evaluate the team members on the graduate attribute of Problem Analysis and Design. Please answer the following questions:

1. What are the limitations of your solution? Put another way, given unlimited resources, what could you do to make the project better? (LO_ProbSolutions)

2. Give a brief overview of other design solutions you considered. What are the benefits and tradeoffs of those other designs compared with the chosen design? From all the potential options, why did you select documented design? (LO_Explores)

E.1 Group Reflection

1. Our solution's two most significant limitations are the number of developers and time constraints. As 7eam only has five developers with solely software engineering experience, we need to learn how to market a product. We also need multiple years of development experience to avoid pitfalls that individuals who have worked in the industry would know to avoid. If we were given an unlimited number of individuals to make an entire development team with dedicated teams per dev-op, frontend, backend, and business teams, the application would see a much more significant influx of users and have an incredibly refined design. Furthermore, we are constrained by time, not just by the due dates but by how much content we five individual developers can put in. We could refine the application incredibly well with infinite time and no other resources.

Outside of those limitations, the APIs also proved to be limiting. There are some ideas that the team had that we were unable to go through with due to the lacking APIs, especially on the Anime side of things. If these APIs were available or if this information magically appeared in our database, we could have developed a more complex algorithm and potentially some cool new features.

2. Another design solution was a minor decision about pairing users and preferences together. The benefit is that they go hand-in-hand, where users are directly linked to their preferences. The downside being it adds an overhead to updating the preferences list, and since we are going to take into account user feedback for the preferences, it could go through a handful of updates. Therefore splitting it off into a module in isolation and linking the user is a safer implementation.

Another design decision that we ended up not going with was to have, alongside each preference, a scale to say how important that preference is to a user, potentially on a scale of 1-10 or 1-3. We felt like this would help people who might not care much about the type of entertainment (Movie, TV Show, Anime) in comparison to the genre (e.g. they really love action and comedy but hate romance). Ultimately this idea was placed on a potential addition list since it doesn't add enough value to the system in comparison to the complication it poses to both the users and the developers.