CAPSTONE 4ZP6B PROJECT

Design Document

Version 1

FINDDIT

GROUP 19

Prakarsh Kamal

Japnit Singh

Kanwar Sandhu

Hriday Jham

Purpose of the Project

Purpose:

The purpose of building this application is to enhance the social experience and decision-making process for groups of users when it comes to choosing a restaurant. The app aims to increase end-user happiness when making a collective decision in a group by streamlining the decision-making process through an interactive stack-based swipeable UI and a cost optimization function considering user preferences.

Goals:

The goals of building this application are to increase user happiness, reduce decision-making fatigue, and facilitate enjoyable group outings through an interactive application.

Purpose of the Document

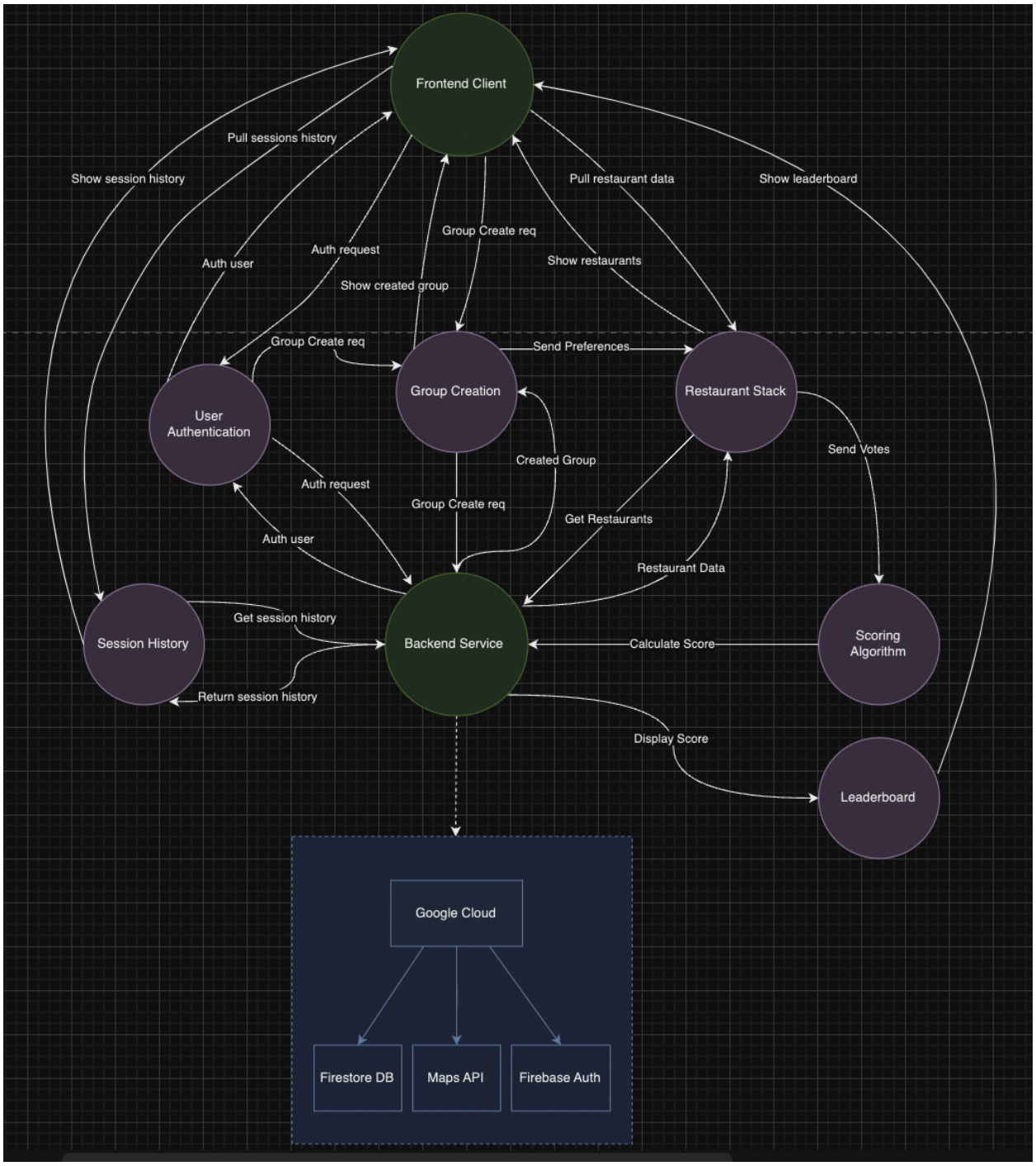
Purpose:

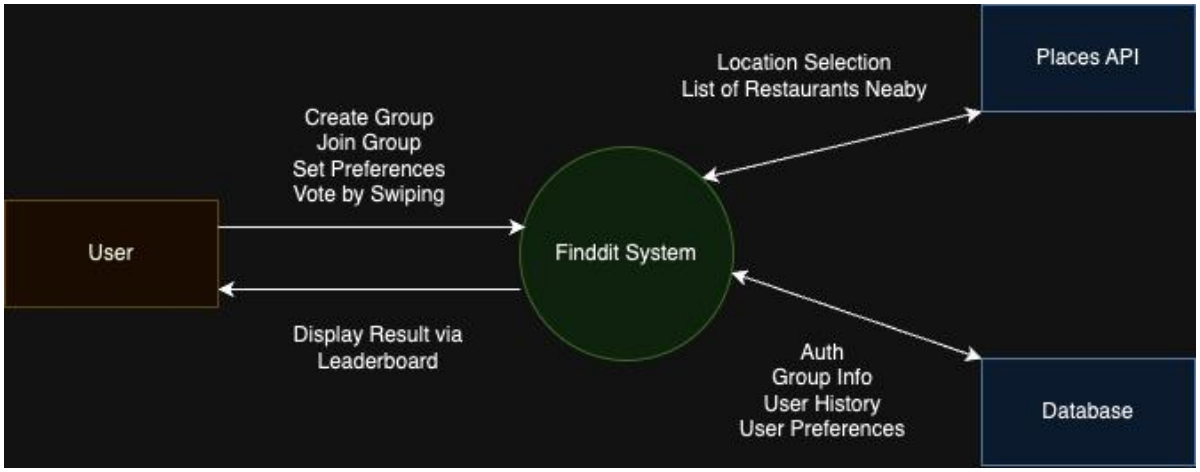
The purpose of this document is to decompose the system into components and provide a Modular Interface Specification for each component in the system. This document will be utilized as a basis for the implementation work to be completed in the upcoming months.

Scope:

The designed system serves the purpose of simplifying the decision-making complexities that often arise when determining a restaurant within a group setting. Its core functionality allows users to create a group effortlessly, add members to it, and facilitate a virtual voting process through a user-friendly swipeable stack interface. The system takes into account individual user preferences and votes, employing an algorithm model to generate a comprehensive leaderboard. This leaderboard, in turn, serves as the conclusive factor in determining the chosen restaurant, with the overarching goal of minimizing user dissatisfaction throughout the decision-making process.

4 Component Diagram & Relationships





5 Component Requirements

Component	Requirement
User authentication component	• FR 1: User profile creation • NFR - Privacy: Account Management • FR 1: User registration page
Group creation component	• FR 2.1: Creating a group • FR 2.1: setting preferences • adding users
Restaurant stack component	• Get restaurant list • FR 4.1: Display restaurant cards • FR 4.2: visualize right swipe/left swipe
Session history component	• FR 6.1Track session history • NFR - Accuracy: Store session preferences • FR 3.3: Display active sessions • FR 3.1: Enable users to join sessions • FR 3.2: Store results of inactive and active sessions
Leaderboard component	• FR 5.1: Display results • NFR - Reliability: Rank restaurants based on score
Scoring algorithm component	• FR 7.1: Calculate votes • FR 7.1: Generate score for all restaurants • FR 7.1: Rank restaurants based on the score

6.1 User authentication component

Normal Behavior	The user inputs the credentials to the component. Existing User is authenticated and logged in. New user is able to sign up and a new profile is created. User is able to log out and switch profiles.
Implementation	Implemented using Firebase Auth API
Potential Undesired Behaviour	User login/signup fails. Firebase Authentication not working as expected. Logged in user is logged out before the expected login credential time to live (TTL).

API Name	**Input**	**Output**	**Comments**
sendSignupRequest	firstName, lastName, email, iconID	userID	Takes in user information and creates a new user in the database
createUserWithEmail AndPassword	FIREBASE_AUTH, email, password	firebaseResponseObject	External API call to firebase to sign up a new user
signInWithEmailAnd Password	FIREBASE_AUTH, email, password	firebaseResponseObject	External API call to firebase to sign in a new user
signOut	FIREBASE_AUTH,	firebaseResponseObject	External API call to firebase to sign out a user

6.2 Group creation component

Normal Behavior	The component is used by the user to create a group by

	selecting a name, adding desired members and selecting preferences as to get desired responses.
Implementation	A react component in the frontend takes in user inputs, creates a group table in the firestore database and sends the inputted data to fetch restaurants using the Nearby Restaurants API.
Potential Undesired Behaviour	Group creation fails. Unable to add logged in users to the group. Group does not show selected name or icon. Group preferences do not match the intended preferences.

API Name	**Input**	**Output**	**Comments**
createNewGroup	groupName, groupIcondID, groupAdminEmail, groupMemeberEmails, votingDeadline, isActive, adminPreferences	Response.CODE	API call to create a new Group object in the database.
findUserByEmailOrName	emailOrName,loggedInUser	List[] userID	API call to get logged-in users by name or email.
getCardDataFromGroup	groupID	json{} restaurantData	API call to get restaurant data for a particular group ID.

6.3 Restaurant stack component

Normal Behavior	The stack displays the restaurants as cards in a deck along with other restaurant data which is used by the user to make a decision. The stack visualizes the right and left swiping of

	the cards.
Implementation	Implemented using a component called 'react-native-deck-swiper' which captures user gestures and processes corresponding actions.
Potential Undesired Behaviour	No cards to display. Cards displayed have an incorrect UI. Unable to visualize swipes. Unable to present the restaurant cards. Unable to swipe left or right. Unable to register user actions.

API Name	**Input**	**Output**	**Comments**
registerRightSwipe	userID, groupID, restaurantID	Response.CODE	API call to register right swipe on a restaurant card
registerLeftSwipe	userID, groupID, restaurantID	Response.CODE	API call to register left swipe on a restaurant card
registerUpSwipe	userID, groupID, restaurantID	Response.CODE	API call to register up swipe on card

6.4 Session history component

Normal Behavior	The component stores all the sessions created along with the preferences, members and results of each group. The user can also view any previous session and add users to a session if still active
Implementation	Implemented using an API to the Firestore database.
Potential Undesired Behaviour	Unable to store or retrieve a previous group from the history.

API Name	**Input**	**Output**	**Comments**

getActiveGroupsForUser	userEmail	List[] activeGroups	API call to fetch groups for the given user.
getInactiveGroupsForUser	userEmail	List[] inactiveGroups	API call to fetch previous groups for the given user
getCheckedInUsersForGroup	groupID	List[] userID	API call to fetch all checked in users for the group.

6.5 Leaderboard components

Normal Behavior	The component displays the restaurants as a ranked list based on the score from the scoring algorithm.
Implementation	Implemented using node.js UI frameworks
Potential Undesired Behaviour	Unable to display the results.

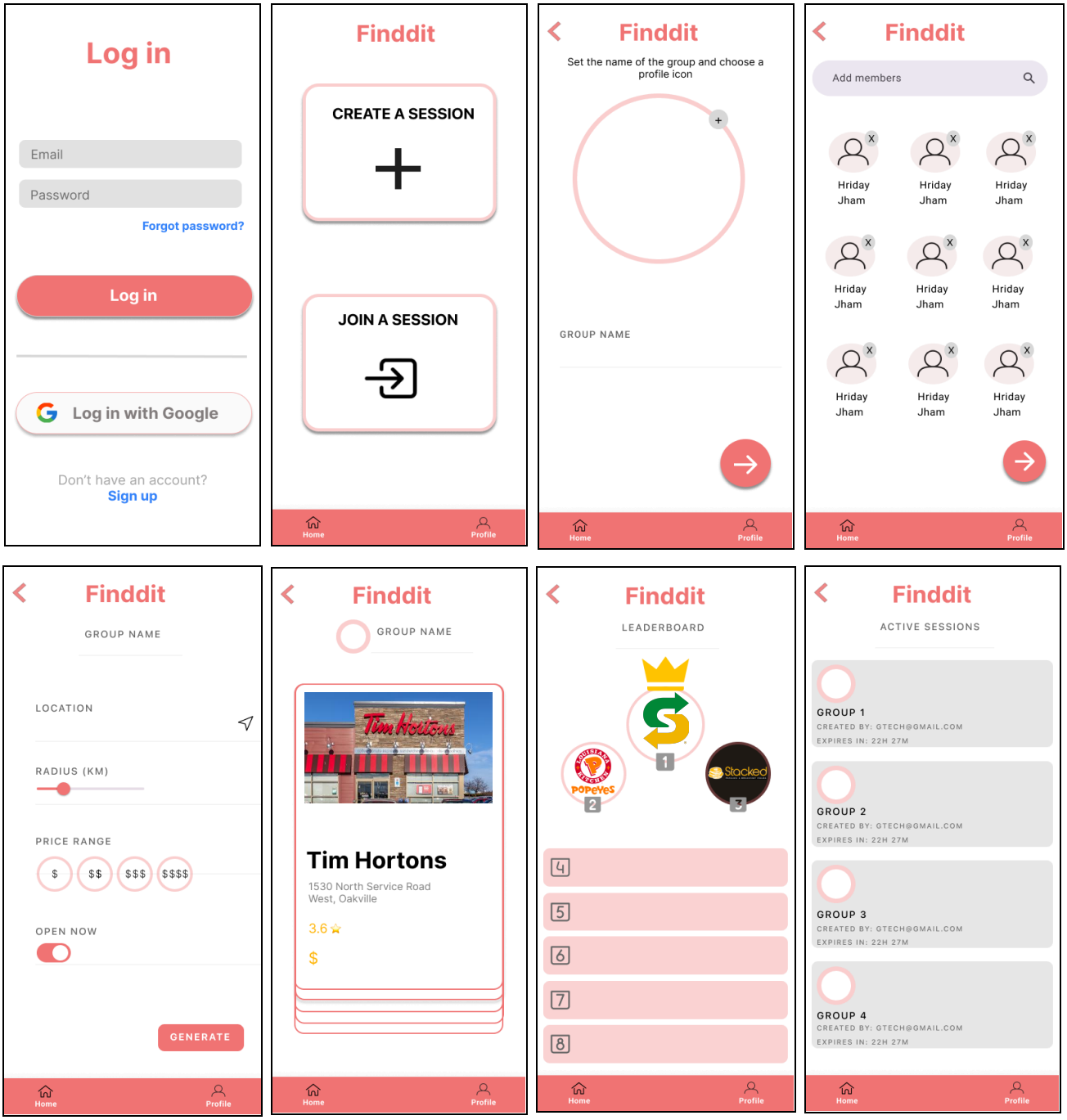
API Name	**Input**	**Output**	**Comments**
generateLeaderboard	groupID	json{}leaderBoardObject	API call to generate leaderboard based on the restaurant scores after voting has finished

6.6 Scoring algorithm component

Normal Behavior	The component counts the votes from each user of a group and generates a score for each restaurant in the results of the group. A leaderboard then ranks the restaurants based on the corresponding scores.
Implementation	Implemented using a cost function-based algorithm that aims at reducing user unhappiness by optimizing restaurants based on how distant they are and the restaurant parameters from the user preferences.
Potential Undesired	Failure to rank the restaurants. Miscalculation of restaurant scores.

Behaviour	

7.1 Figma wireframes



7.2 User interaction

The above wireframes are created in Figma and showcase the user interaction and flow in the app. Initially, the user logs in (or creates an account) and is authenticated successfully. On the next screen, the user can "Create a session" or "Join a session". Clicking Create session takes the user to the next screen where they can set a name and select an icon for the group. The user is then taken to add users on the following screen. They can search for users and add them. Added users can also be removed by clicking the "X" icon. On the next screen, the user can set group preferences such as selecting the location, radius, and price range, and open now toggle. Then, the swipeable restaurant cards show up and the user can swipe left or right to vote. Once all votes have been collected, the scoring algorithm weighs them and displays the most voted restaurants on the leaderboard screen. The top 3 restaurants are highlighted. There is also a bottom tab navigator for "Home" and "Account". Navigating to the Home page, the user can see the 2 calls to action create/join session buttons. When the user clicks Join session, they can see the current active group sessions and join them. From the Account page, the user can sign out.

7.3 Design choices

We wanted to achieve a minimal look with our app's design. We believe this improves the appearance and also the user experience. Our primary color is #F27575 which is a mixture of dark pink and light red that creates a warm hue. To complement this, we went with white as the background color and black/gray as the text colors. For the font, we used a readable and clean style. We imported numerous icons and user components (such as buttons, modals, input fields, and switches) into the front end which enhances the user experience and provides a clear understanding of features.