IAT359 Course Project

Fitness 359
Final Report

Team 14

Table of Contents

Project name

FITNESS 359

Abstract / Summary

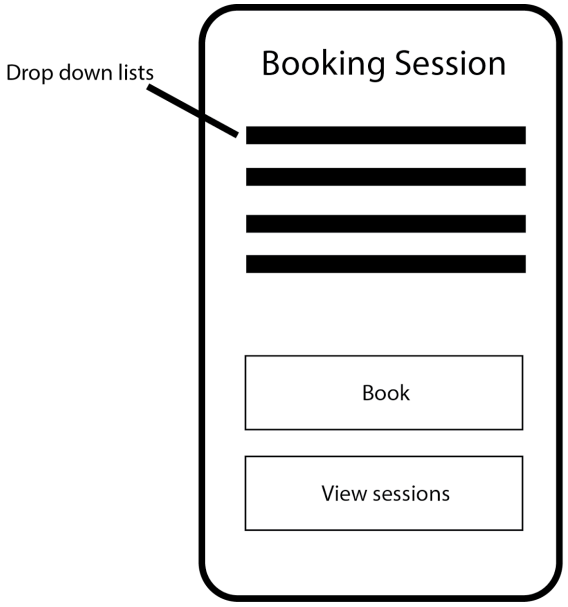
FITNESS 359 performs as a virtual membership card that supports its members during the inconvenience of the contemporary pandemic. This application allows members to book their workout sessions online. This process is to help local gyms with the new restrictions and guidelines to prevent the spread of COVID-19. This application allows gyms to be ready for their members' arrival. We implemented an authentication process that checks a user's account through an internal SQLite database. Implementation of the external database is considered for future submission. Similar to the process of many fitness centers, users are given username and password registered by the center themselves thus, there will be no registration form. After logging in, members have the option to book workout sessions on their own time at their preferred location. FITNESS 359's users can also change the general theme of the UI, which is stored on SharedPreferences. An embedded map is implemented to show gyms and to facilitate the booking process. Users can monitor their health while working out. This application can track users' steps and calories burnt.

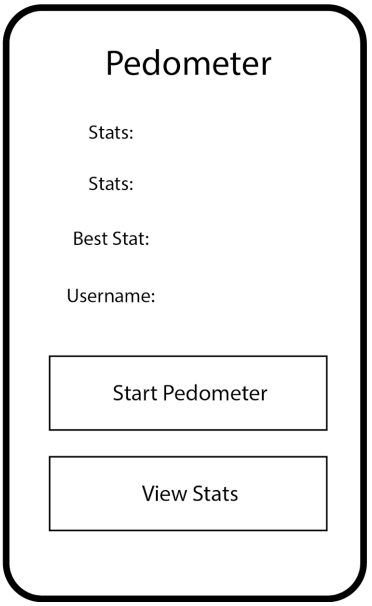
Completion Report

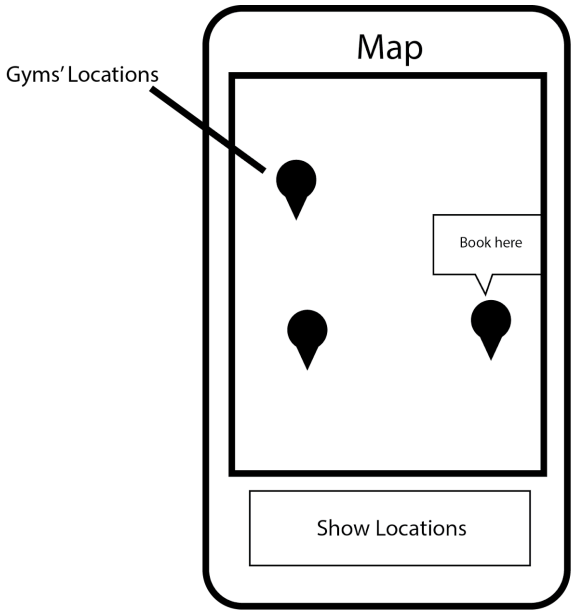
Application has a logo image (https://www.flaticon.com/free-icon/weightlifter_1468372) so users can easily find the application on their device after installation. The UI is developed emphasizing the theme and characteristics of the application's nature. An embedded map to facilitate the booking process. Users can look at their location as the application tracks their current location. They can also select a gym to book. FITNESS 359 is database-powered. We have put the authentication and session booking processes into an SQLite workflow. Users are able to see and delete their booking data. SharedPreferences is utilized to offer the user a customizable UI, manage login sessions and storing personal records on the pedometer.

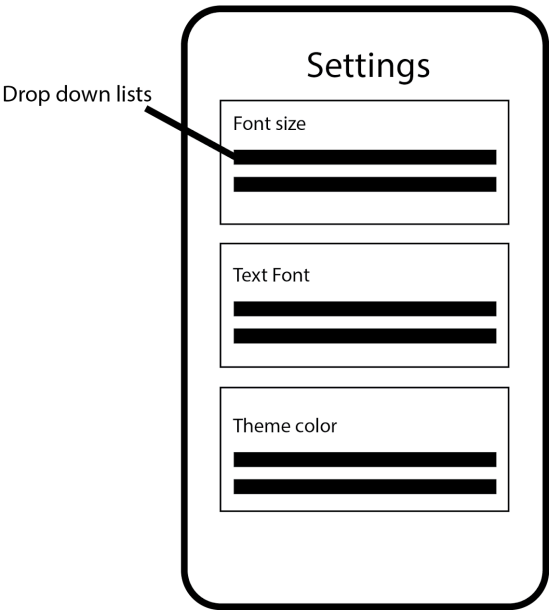
UI Design - Sketches and Implementations

Sketches

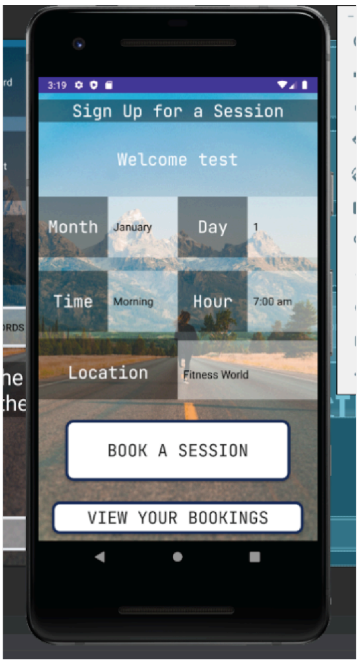




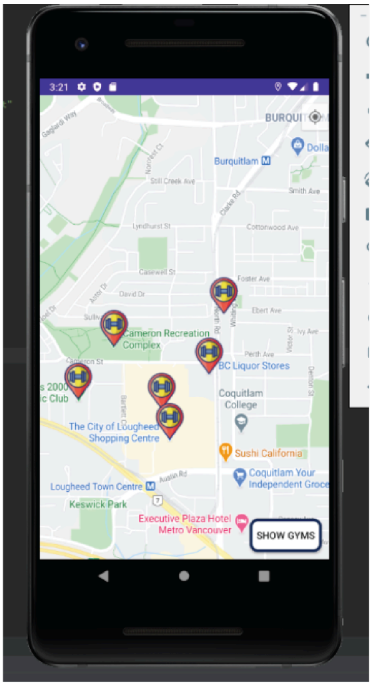


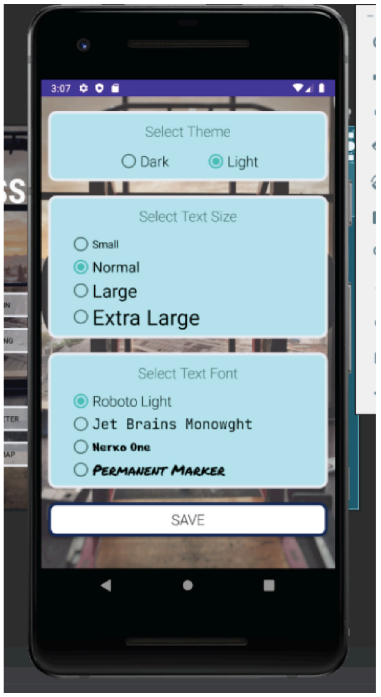


Implementations



Booking & Pedometer Activities





Map View & SettingsActivities

Project Requirements and Details

FITNESS 359 currently have 6 activities:

- *MainActivity*: This is the first activity the user sees in this application. The user can choose how they want to navigate in the application. Buttons on *MainActivity* allow users to go to the *AccelerometerActivity*, *MapsActivity*, *SignInActivity* and *SignUpActivity*. Users can change the UI settings by clicking on the gear icon.
- *AccelerometerActivity*: Allows users to monitor their health while working out using the device's accelerometer (see Device Hardware Usage).
- *MapsActivity*: This activity has Google Map integration to display a map on the device's screen (see Device Hardware Usage), tracks user's device and allows users to select a gym to book in.
- *SignInActivity*: User can sign in to the application or create an account. This uses an external MySQL database to store user's login information. PHP scripts are used to authenticate the log in information then parse the log in result back to Android. (see External SQLite Database).
- *SignUpActivity*: This activity allows users to book a session with their gym of choice. The user has a choice in month, day, hour and location. This information is sent to a PHP page that simultaneously inserts a data row in an online MySQL (see External SQLite Database).
- *ViewBookingActivity*: This activity is accessed through the *SignUpActivity*. ViewBooking allows the user to view their booked appointment and cancel their appointment if they wish to. The data retrieved from an external MySQL database is displayed on a RecyclerView. By clicking on the blue-color location text, the user will be directed to their device's map application via an implicit intent (see Implicit Intent).
- *SettingActivity*: Using shared preferences, users can change the font size, font family, and the theme of the entire application (see Shared Preferences).

Explicit Intents

Explicit intents were used throughout the application to ensure a seamless navigation between activities.

Implicit Intents

For this milestone, we implement implicit intents in the ViewBookingActivity. These intents direct the user to their device's map application to show the

location of their booked session. Verification steps are implemented to ensure the device is capable of handling the implicit intent.

Shared Preferences

- Usage of shared preferences to store user's preferences in the UI. This allows the user to change the theme, text size and font of the entire application. Each selection is saved and displays the appropriate theme, text size and font.
- Shared preferences are also used to register the current user and log in status (gym member or not). Hence, the current username will be used to register for booking sessions and retrieving booked sessions. When the user logs out, the username shared preference will be registered as a null.
- Another use for shared preferences is keeping a personal best record of the Pedometer. While the pedometer stores records at an Internal SQLite Database, the best record will be kept in shared preferences for display and comparison purposes.

Internal SQLite Database

Internal SQLite Database is used to store the Pedometer Activity's statistics, hence using that to find out the best statistics achieved by the user. The statistics will be shared by all users of the same device, regardless of being a member or not. Here are some specifications about the database:

- Table name: records
- Columns:
  - user, calories, steps, data as fields

External MySQL Database Connected via PHP Pages

This database is used to store User Accounts' Information and Booked Sessions' Details. By creating customized Listener and utilizing Android's Volley library, we manage to connect the application to PHP scripts that handle MySQL operations. By storing these data online, it allows every device, once given the right permission, through logging in, can access the information.
(Android to MySQL via PHP tutorial:
https://www.androidhive.info/2012/05/how-to-connect-android-with-php-mysql/)

Device Hardware Usage

Accelerometer

- Accelerometer is used to display the step count and calories burnt of the user while the user is engaged in physical activity. The user must select the 'Accelerometer' button in the MainActivity to access this feature. The device must be on the user for this feature to work.

(Basic pedometer algorithm reference: https://programmerworld.co/android/how-to-create-walking-step-counter-app-using-accelerometer-sensor-and-shared-preference-in-android/)

GPS

- GPS functionality used to track device's location and facilitate the booking process. Ideally, the application locates the gyms nearby the user's current location. Due to the price attached to every request to the Google Places API, this was not possible (see Challenges). The map view changes so the user can see their current location on the map as they move. This is with the help of public class FusedLocationProviderClient by Google (https://developers.google.com/android/reference/com/google/android/gms/location/FusedLocationProviderClient). This public class helped with tracking the device's current location and checks for changes in location.
- This application does not use the device's location without the user's permission. A permission request is asked when the user uses the application for the first time and when the user has not granted permission to use their device.

Map

- The map uses Google's Maps SDK for Android (https://developers.google.com/maps/documentation/android-sdk/overview). This is helping with the UI of the MapsActivity. The map shows the device's location with a small blue dot. There is a current location button on the top right of the screen to show the device's current location. This is for the user when they look around the map and are no longer looking at their current location. They can click on the current location button and the map will display their current location.
  Once the 'Show Gyms' button is clicked, the map's view stops showing the user's location to display gyms by Lougheed Mall in Burnaby. This was done to accommodate the usability of the application and prevent the usage of Google's Places API.
- Custom gym markers (https://www.flaticon.com/free-icon/fitness-gym_2038640) were hard-coded to display on the map with the help of the Gym class. This class helps store basic information of a gym (gym's name, latitude and longitude of the gym). When the user clicks on the marker, an info box appears above the marker displaying the name of the gym and "Book now!". This was done with the help of CustomInfoWindowAdapter class. Clicking this info box takes users to the SignUpActivity. This sends the gym's name, latitude and longitude to the next activity.

Challenges

These are some challenges we faced during the development of this Milestone:

- There were some troubles using GitLab as a method to collaborate. Dealing with merge conflicts occasionally ended up with destructive results (i.e. some code went missing, corrupted codes after merging, incompatible API and Plug-ins, etc.). However, through various means of communications, we managed to fix most of the problems and get the application to work smoothly.
- Initially, we tried to use Internal SQLite Database to store User Accounts and Booked Sessions. However, it soon posed some problems that a new device's database will be empty, meaning there is no account to be logged into. To tackle this problem, we manage to establish a web server that stores the database and PHP scripts, which communicate with Android using Volley.
- Implementing a complex Pedometer requires deep understanding of Kinesiology, not to mention copying the entire algorithm of the researchers, which is quite unfavorable. Nonetheless, for this project, we utilized a fairly uncomplicated algorithm that is easily understood. Even though the results might not be extremely precise, it can get the job done when run on a Samsung device.
- We wanted to implement displaying all the gyms near the user's current location. Unfortunately, using the nearby search with Google Places API costs money. Testing one night alone, the search requests were a total of 334 requests. Classes that supported the nearby search were GetNearbyPlacesData, DownloadUrl and DataParser. After seeking some guidance, it was best to hardcode the gym location onto the map and set a button to guide the user to the gyms. This application displays gyms by Lougheed Mall in Burnaby. The call that references the method showNearbyGyms is commented out to prevent any new requests.