- Description
- Intended User
- Features
- User Interface Mocks
  - Screen 1
  - Screen 2
- Key Considerations
    - How will your app handle data persistence?
    - Describe any corner cases in the UX.
    - Describe any libraries you'll be using and share your reasoning for including them.
    - Describe how you will implement Google Play Services.
- Next Steps: Required Tasks
  - Task 1: Project Setup
  - Task 2: Implement UI for Each Activity and Fragment
  - Task 3: Your Next Task
  - Task 4: Your Next Task
  - Task 5: Your Next Task

GitHub Username: LucaUburti

Fitness for Diabetics

Description

For diabetic persons, keeping track of their blood sugar levels (glycemia) is paramount. Besides the diet, physical activity also plays a major role in altering the glycemic values.
With this app diabetic persons can log their blood sugars levels throughout the day and complement this information with any workouts done on that particular day.
The app also has access to an online database with the glycemic index of common foods and a chart outlining the glycemic trends of the last few weeks.
All personal glycemic values are stored locally, so no sensitive data is transferred outside the phone.

Intended User

Diabetic people who want to record their glycemic levels throughout the day, together with any workouts done during that day

Features

- Displays glycemia values and workout info of the last few days
- For each day, allows the user to easily enter the glycemia values, the meals and the workouts
- Displays a summary chart of recent glycemic values
- Search an online database with food glycemic indexes

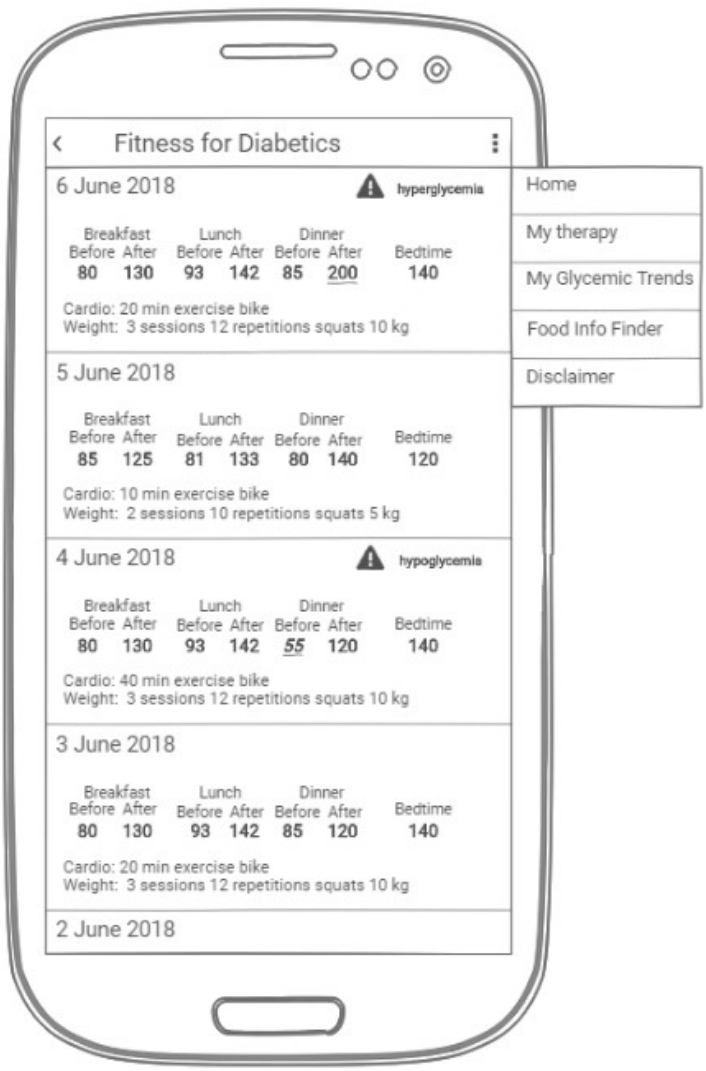
User Interface Mocks

HOME

Summary of the glycemic values and workout information

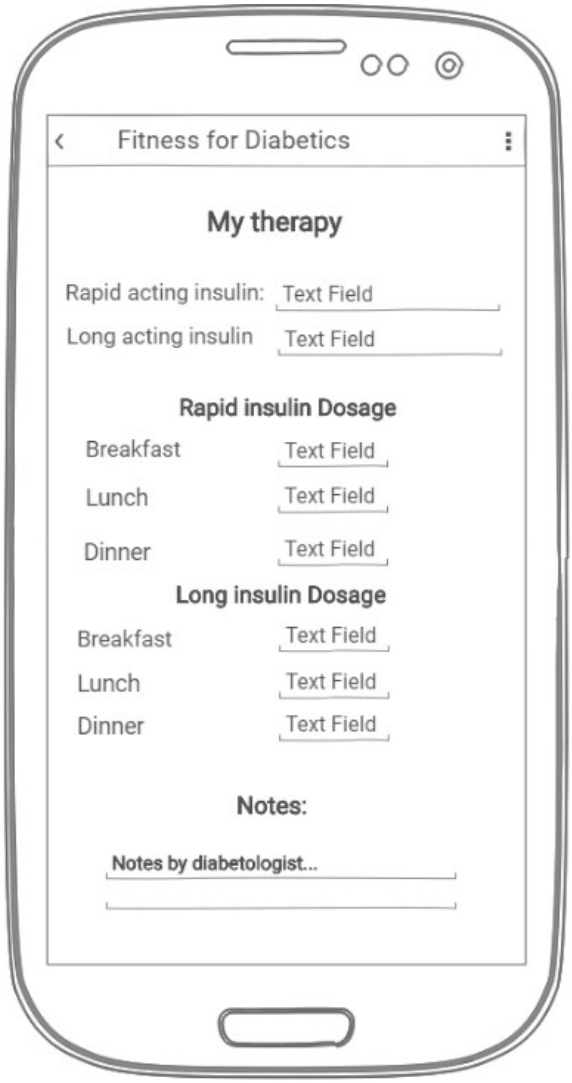
DAY DETAIL

Let the user add and edit the glycemic values before and after each meal, add meal info and any workout done



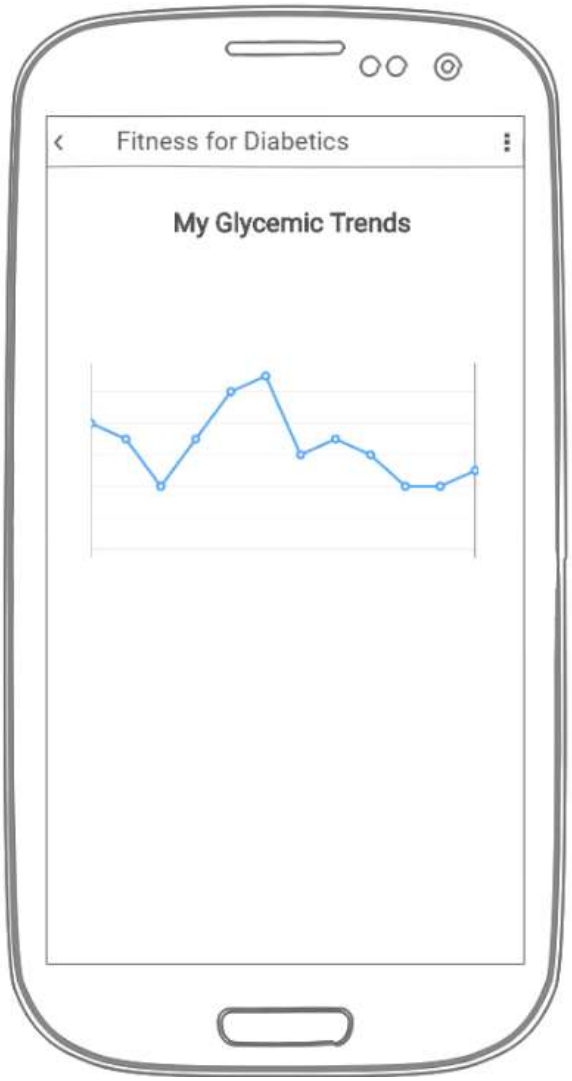
MY THERAPY

Summary of the therapy as prescribed by the diabetologist



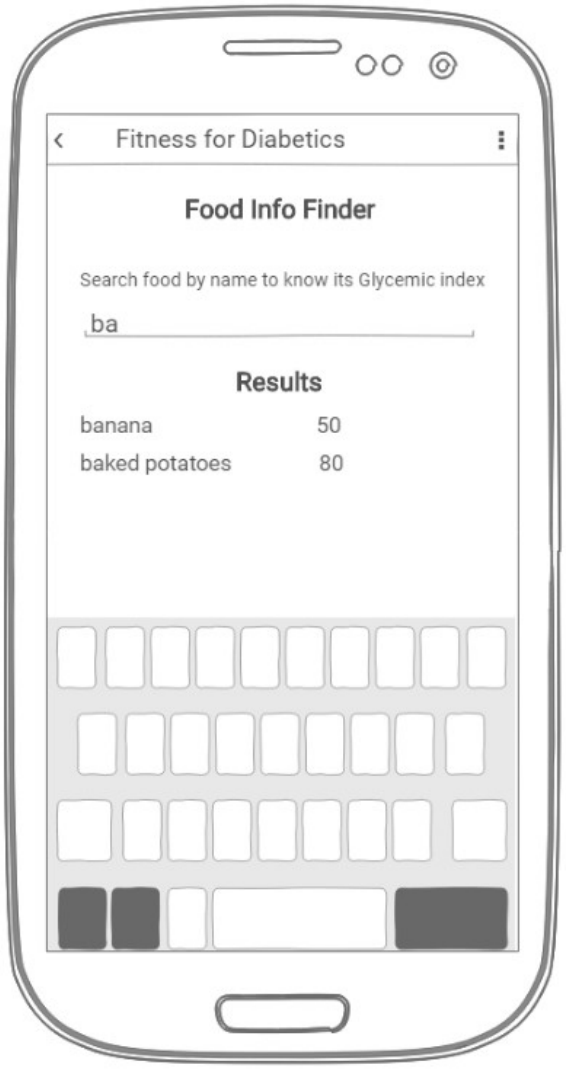
GLYCEMIC TRENDS

A graphical representation of the glycemic levels of the last few days



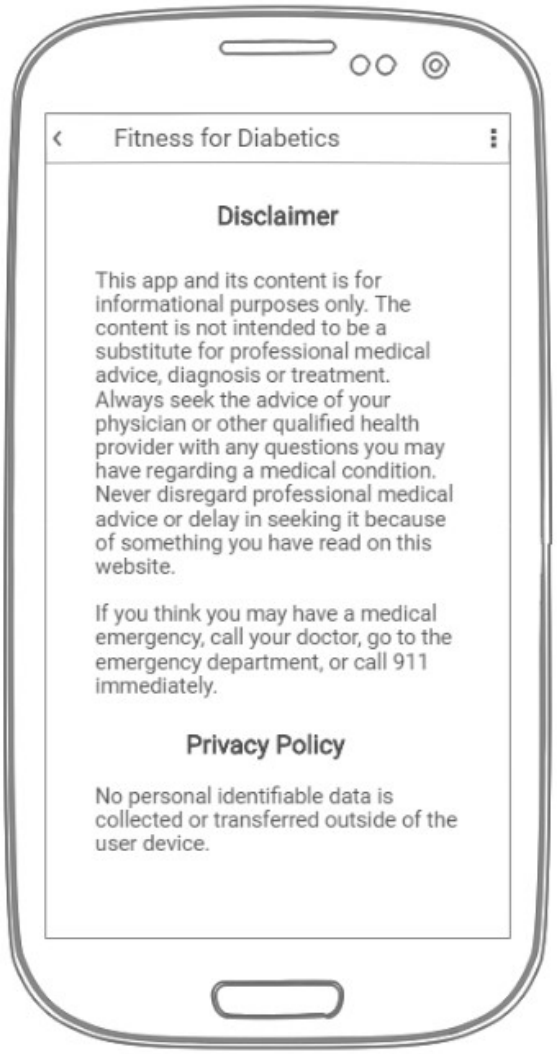
FOOD INFO FINDER

Search an online database for the glycemic index of common foods.



DISCLAIMER AND PRIVACY

Disclaimer info and privacy policy



Widget

Info on latest glycemia reported and last injection

Key Considerations

- App will be written in Java
- App will utilize Android Studio (3.1.3), Gradle (4.4) and the latest stable versions of all required libraries.
- Target and Compile Sdk versions will be 27. Minimum Sdk will be 19
- Room database will be used to store the Glycemic data. LiveData and ViewModel are used when required to avoid unnecessary calls to the database
- An AsyncTask will be used to download the Food Info at runtime
- All strings will be saved in a strings.xml file
- A widget will be used to display data on latest glycemia and injections

How will your app handle data persistence?

All user entered data will be stored locally using Room.
Shared Preferences will be used for any user preferences such as the number of days to display in the Main Activity.
Food Info is not stored locally: it will be downloaded at runtime with an AsyncTask and decoded from a Json table (data hosted on a publicly readable Firebase project)

Describe any edge or corner cases in the UX.

UI representation of both a hyperglycemia and hypoglycemia warnings in the same day summary

How to display blank days with no user data.

Describe any libraries you'll be using and share your reasoning for including them.

Third party libraries will be:

- Butterknife, to eliminate findViewById calls
- Picasso for downloading and caching food pictures
- A line graph library (MPAndroidChart or similar), to create a glycemia graph
- (possibly more if the need arises)

Describe how you will implement Google Play Services or other external services.

Google services used:

- Admob, interstitial on exit
- Firebase Crashlytics
- Firebase Analytics
- (not strictly internal to the App) Food Info Json data is hosted on a publicly readable Firebase project where I will insert mock food data beforehand

Next Steps: Required Tasks

This is the section where you can take the main features of your app (declared above) and break them down into tangible technical tasks that you can complete one at a time until you have a finished app.

Task 1: Project Setup

- Setup the project in AS
- Import the required libraries
- Setup the supporting Firebase project and link it to the AS project

Task 2: Implement UI for Each Activity and Fragment

- Create the Activities classes and for each activity setup a first draft layout

Task 3: Add Google Services

- Setup AdMob to display interstitial ad on exit
- Setup instance of Crashlytics
- Setup instance of Firebase Analytics

Task 4: Define the data structure

- Define the POJO for holding all the information needed in the DayDetail Activity

Task 5: Implement the RecyclerView

- Implement RecyclerView in the Main Activity and its related Adapter.
- Create the Item layout for each day in the Main Activity.

Task 6: Setup Room, LiveData and ViewModel

- Setup room database: make the POJO an Entity, create the necessary Dao, create the db
- Setup ViewModel and LiveData for Query operations
- Setup an Executor class for Insert, Update, Delete operations

Task 7: Implement the main flow logic of the app

- Query with LiveData/ViewModel the requested days to display them in the Main Activity
- Query with LiveData/ViewModelFactory to display the details about a specific day
- Implement the other CRUD operations with the Executor class

Task 8: Implement the FoodInfo AsyncTask

- Populate in the associated Firebase project with a list of food information
- Implement an AsyncTask to download and display food information in the FoodInfo Activity

Task 9: Implement the graph library

- Pick a suitable external graph drawing library
- Implement the graph library in the GlycemicTrends activity.

Task 10: Implement the Widget

- Implement the widget and its related updating service

Task 11: Debug

- Debug on various devices, test all edge cases
- Check that all crash data and app usage info are correctly reported in the Firebase console