SafeWorkout Documentation

[G10] - André Neves 84948, Dany Costa 85097

Context

With the pandemic we are living in, there are a lot of people that became sedentary or are losing their motivation to work out. Gyms are still a place that most are afraid to attend because it is a common place where people tend to gather. For that reason, working out at home or outside becomes a reasonable solution and probably healthier.

Application purpose

The goal of SafeWorkout is to motivate the users to workout at home or outside. Our app will have a section dedicated to social interaction where users can see the evolution and training plans of their friends. This will bring not only motivation but also some kind of competition. Moreover, we intend to improve the users' health.

Features

- Check exercises by category ✔️
- Add exercises to your favorites ✔️
- Workout at home with videos ✔️
- Discover public spaces based on your current location ✔️
- Share info ✔️
- Set goals and watch your progress over time ❌
- Track your eating habits (e.g. calories) ❌
- Sedentary alerts ❌
- Health and nutrition tips ❌

Sensors & mobile functionalities

- **GPS:** use the user's location to advise public spaces nearby for working out
- **Camera:** take pictures of those public places to share with friends(not implemented)
- **QR Code:** share workout plans or other information
- **Image and Video:** show exercise tutorials in the format of video / image
- **Accelerometer:** measure average speed to find out if users are stopped for long periods of time (not implemented)

Architecture & technical options

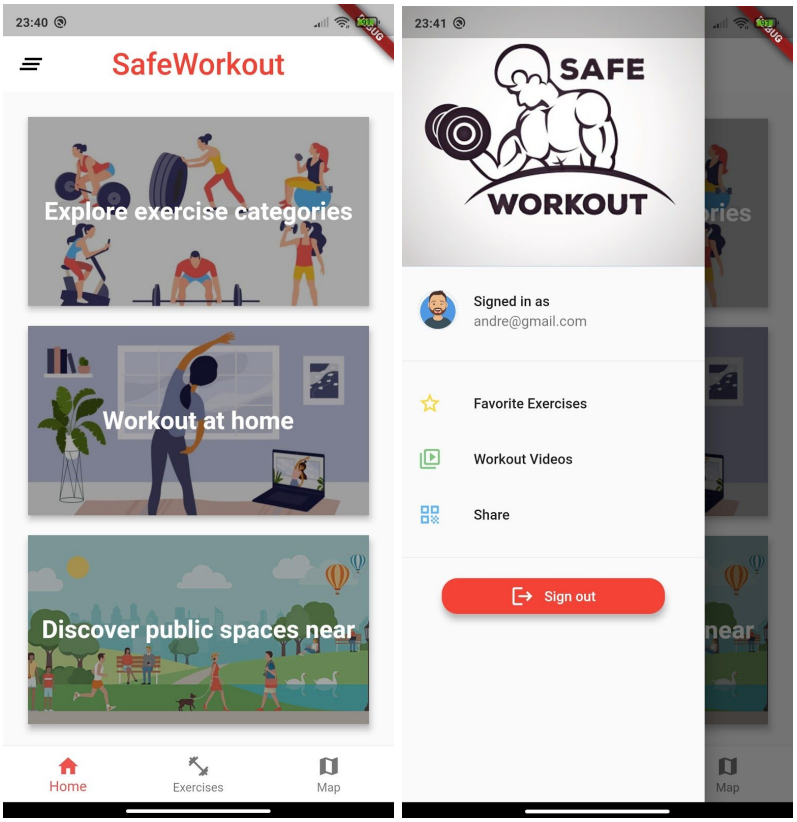
Requirements

- Flutter: Install flutter and dart plugins
- Android SDK
- VS Code (recommended)

Views / Screens

Overview

In terms of UI, our app is organized in a home page and 5 main pages (and sub-pages) that implement our features. The bottom navigation bar holds the home page, exercises page and map page, whereas the drawer contains favorite exercises page, workout videos page and share page. These are functionalities accessed by registered users only, which bring an exclusive, personal touch to SafeWorkout.

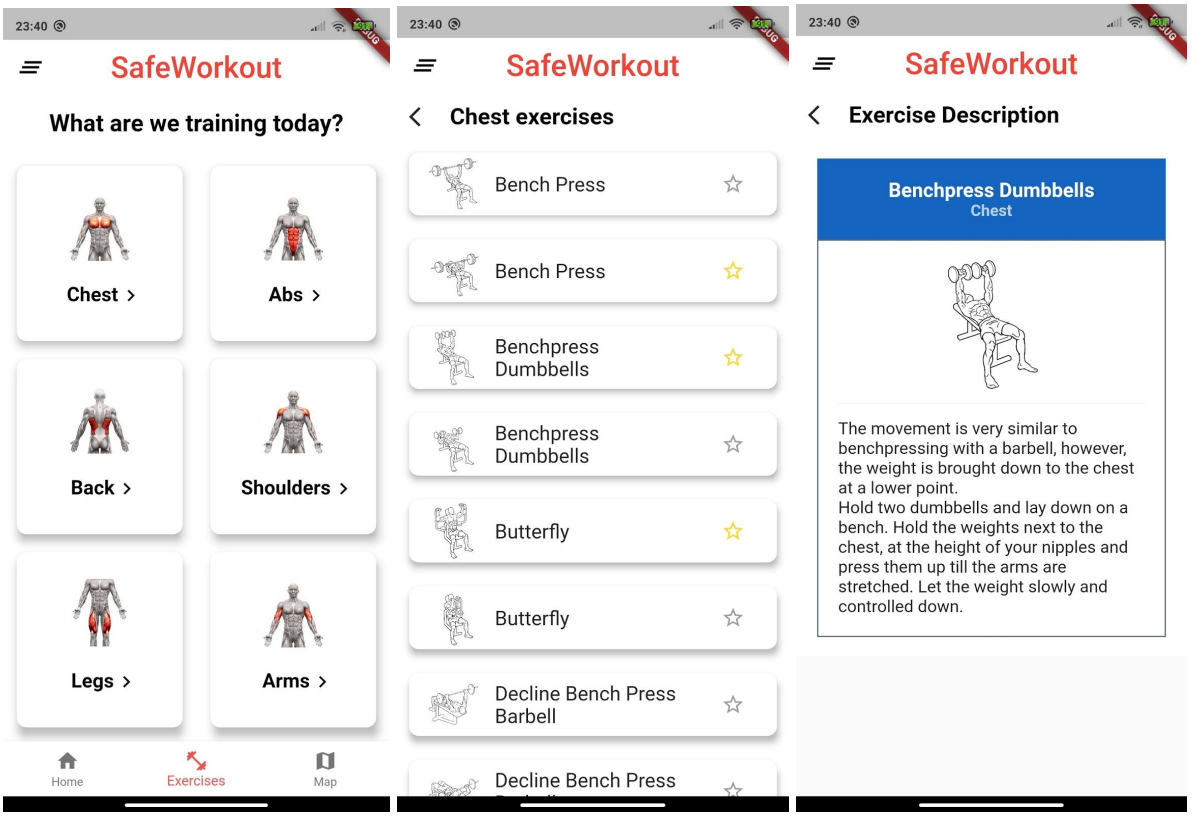


Structure overview: Home page with bottom navigation bar and drawer

Bottom Navigation Bar

The user is welcomed at the **home page** where we display an overview of the goal and usability of SafeWorkout. This is an information only screen so the users have a starting point.

The **exercises page** shows 7 categories arranged on a grid view. This is where the user can explore exercises by category that he/she can perform at a gym with the appropriate equipment. Inside a category, those exercises are organized in a listview with image and name. Depending on whether the user is logged in or is a guest, each exercise will have a favorite icon so that it can be added to his/her favorites. Finally, for each exercise there is another sub page with proper description on how to perform it.

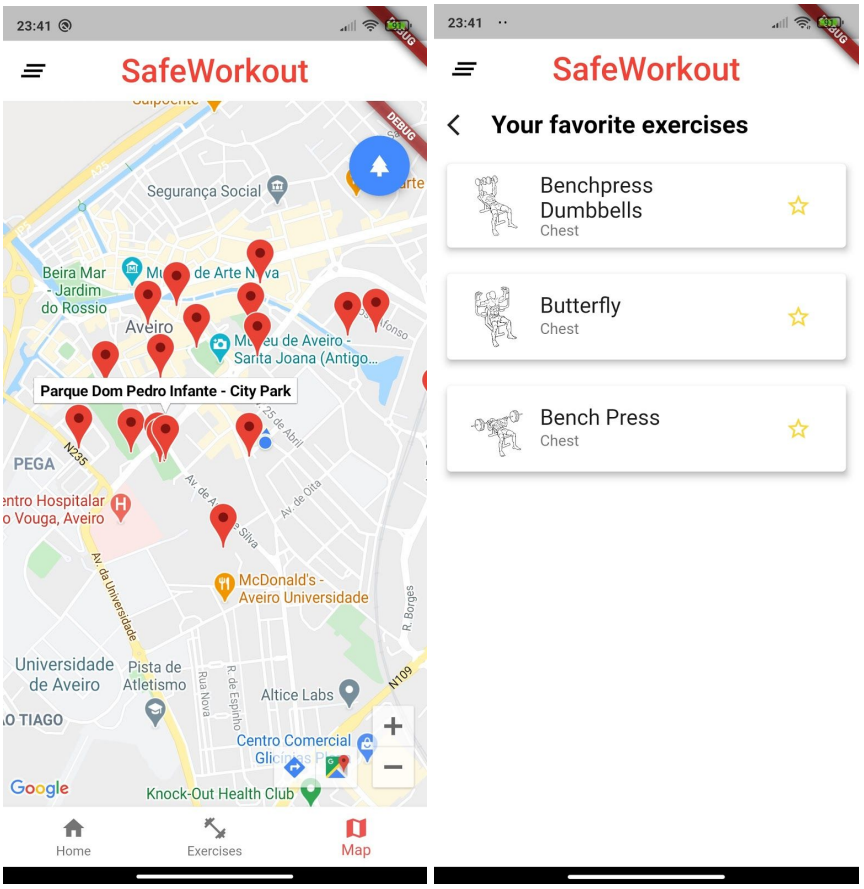


Bottom navigation bar: Exercises page

The **map page** shows a map centered at the user's current location in order to advise public spaces near (e.g. parks). To do that instead of using google maps search package or google Maps places we make a request "nearby search" with the current user location and with a search type of parks. After that, we receive a response with all the parks near the user in a radius we want (in this case 1000m). To see/unsee those parks it's just a question of tapping in the FloatingActionButton and all the parks will appear as we can see in the image below. We also added two buttons that redirect the user to the google maps application when he tapped on a specific marker to get the directions and the path to that marker.

Drawer

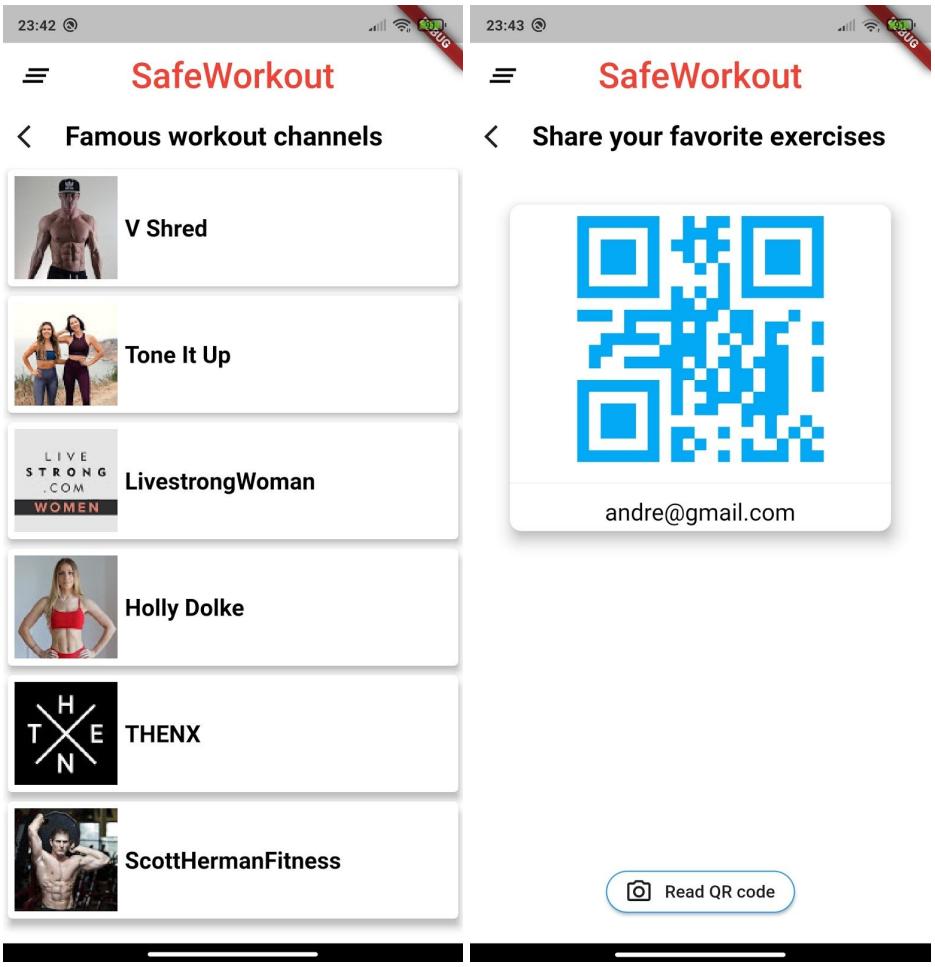
As mentioned before, registered users can add exercises to their favorites. The **favorite exercises page** has the same layout as the Exercises Page with the same functionalities such as detailed description and favorite option, in case they want to remove some item from that list. We used a **firebase realtime database** to store the current user's favorite exercises so that the user itself can have their own exercises saved and take benefits of being logged in.



Map page (bottom navigation bar) and Favorites page (drawer)

For those who do not intend to go to a gym we created a page dedicated to home workouts: **workout videos page.** Opposed to the exercises already described that require gym equipment, this option is the opposite in a way that the user can perform these workouts at home or outside with the help of youtube videos and without any or little equipment. This page shows a list of popular channels and its videos which can be played in the app itself, instead of being redirected to youtube. Like favorite exercises page the workout video page only appears to users that have an account.

Finally, in order to share the user's favorite exercises, in the **share page** is generated a QR code with that data. Additionally, it is also possible to scan someone else's code using the camera.



Workout videos page and Share page

Widget tree

Basically our app has two parts: the **user** and **guest** part.

- **Guests** , as mentioned before, are able to see only the exercises and use the map so their tree is basically:
    - Main Page -- LoginPage -- [3 possibilities]:
        - Home/Exercises/Map
- **Users** are able to have their own favorite list, see the videos section and share info:
    - Main Page -- LoginPage -- [5 possibilities]:
        - Home/Exercises/Map/Favorites/VideoPage
        - Inside Exercises we have:
            - ExerciseListPage->DescriptionPage

We didn't build a specific widget tree because in our app after LogIn user can navigate in different pages the way he want

BLOC Pattern

The data displayed concerning exercises is fetched using an API. Because the amount of data is quite big, we decided to adopt a BLOC pattern strategy. First we fetch the data with http requests that will be transformed into objects in a repository class. Then the BLOC uses **streams** to transfer the relevant data into the UI where we need it. This process occurs for each access to a category in order to list the exercises. Some processing is done in the backend to handle some details, such as category and image association for each exercise.

Firebase

Firebase was used for the **authentication** and to **store** favorite exercise lists in each user:

- **Authentication**
  - We implemented two users (andre@gmail.com / dany@gmail.com with the same password)
- **Real Time Database :**
  - the current user has his own favorite exercise list stored in firebase realtime database that can be displayed when he logIn and

The purpose of using firebase was to differentiate a user with an account (that has more perks) and a guest who can only see the exercises and use the map

Assessment

Overall, we did not fulfill every objective that we first thought of. We were able to implement 5 key features listed above and use sensors and mobile functionalities such as QR code, camera, GPS, display images and play videos.

Main **backend issues**:

- Exercises API:
  - In order to have a list of exercises with images, we need to make 2 HTTP requests: one for the actual list of exercises and a second one for the corresponding images. This brings performance issues because not only are we taking more time to fetch the data but also even more time as we need to merge these images with the exercise list.
- Youtube Page
  - We had some problems in terms of google API key because it has a limit of quotas and we all the requests we did the limit of quotas were constantly being reached
- Google Maps :
  - Problems making requests to search near parks
  - Location problems
  - Mainly API problems
- LogIn
  - Using firebase to authenticate the users we had a big issue in terms of exceptions because the IDE was catching an exception that was not

supposed to and too solved that we spent a lot of time searching for the solution (when basically there was no solution)

In a general way we consider that the final product turned out to be very satisfactory given our experience with Flutter and time constraints.

Contribution

André Neves 84948 (50%):

- Front-end:
  - Home page
  - Every exercise related page: Exercises categories page, exercises list page, exercises description page
  - Drawer design
  - Favorite exercises page
  - Share page
- Back-end:
  - Implemented BLOC pattern to fetch data from API, along with data processing
  - QR code generation and scan features

Dany Costa 85097 (50%):

- Front-end:
  - Map page
  - Every video related page: Workout youtube channels page and videos page.
  - Login page
- Back-end:
  - Firebase authentication used to login

- Firebase Cloud Firestore to store user’s info along with his favorite exercises list
- Youtube API
- Google Maps API