How to Use this Template

1. Create a new document, and copy and paste the text from this template into your new document [Select All → Copy → Paste into new document]
2. Name your document file: "**Capstone_Stage1**"
3. Replace the text in green

GitHub Username: UmutFlash

OpenActivity

Description

The OpenActivity App gives users the possibility to find outdoor spots nearby without having to pay for sport activities. For e.g: volleyball, table tennis, crossfit/fitness...

The user can find spots himself and has the possibility to add location, pictures and descriptions.

Intended User

Any consumer interested in looking for a possibility to do sports activity in the vicinity for free.

Features

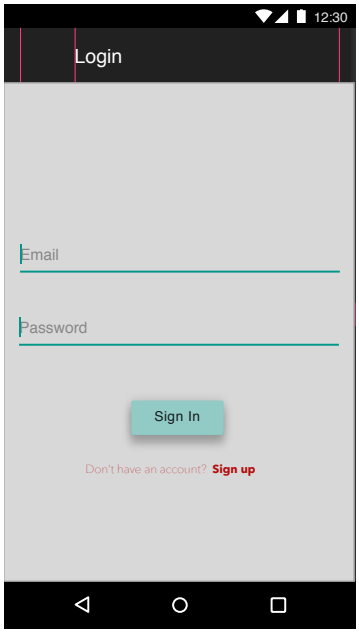
The App is written solely in the Java Programming Language.

- Saves information
- Takes pictures
- Save pictures
- Save Location
- Show on Map Makers (Places for sport activity)
- Show User Location
- Login
- Authentication

User Interface Mocks

Screen 1

Login



If there is no network available, the user cannot log in and an error message appears.

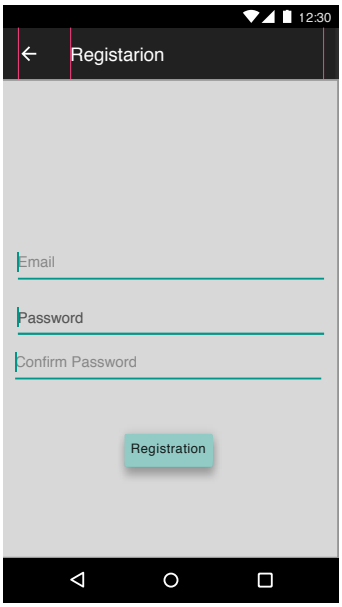
The user can enter his login data in this view. To do this, the user clicks in the corresponding fields e-mail and password, taps on the Button „Sign in“ and is then navigated to the Main View.

If an error occurs in this process, the user receives a meaningful error message.

If the user has not yet registered to use the app, he can start the required process by clicking on the red "Sign up" text and is navigated to the registration view.

Screen 2

Registarion



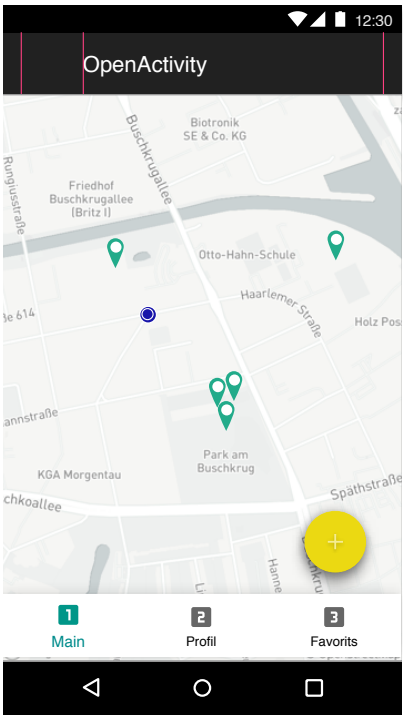
If there is no network available, the user cannot register and receives an error message.

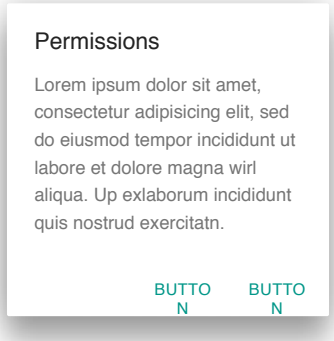
A click on the back icon navigates the User back to the Login View in case he wants to cancel his registration.

On this view the user enters his email address, his password and his confirmation password and can complete the registration by clicking on the button Registration. Then the user receives a success dialog and is navigated to the Login View.

Screen 3

Mainview





The first time the user logs in to the app, he will be asked to agree to the required location permissions before the page can be displayed. A dialog with the buttons Yes and No will be displayed. The app cannot be used without permission.

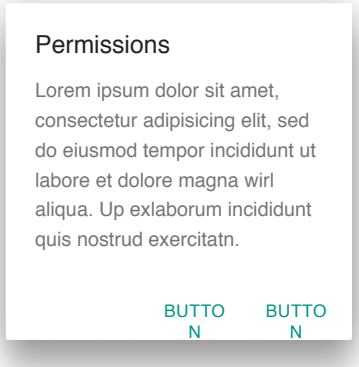
The Main View consists primarily of a map. On the map, all activity spots in its vicinity are displayed.

The user can simply click on a location marker and then get to a detail view which presents the title of the location, the appropriate category, photos and a description.

Alternatively he can add a location himself. Click on the add icon and navigate to the corresponding view.

Screen 4

Add Spot View



If there is no net, you cannot send anything and the button is disabled.

On this view the user can add information to a new location and then create it. The required information is entered in the fields Category, Title and Description. In addition, photos can be added using the "Camera" button.

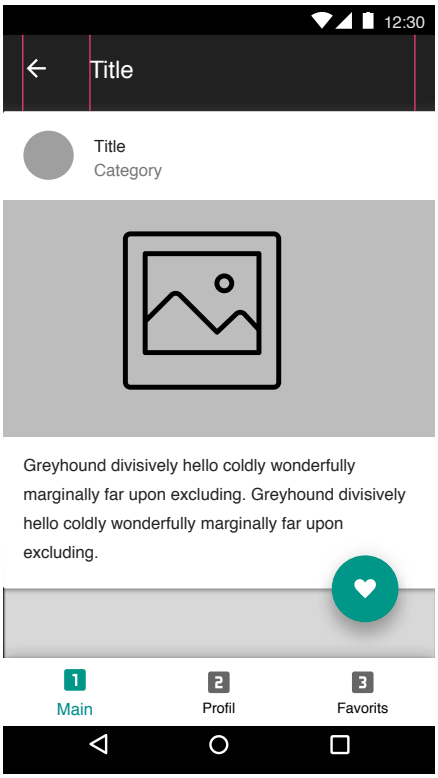
The Title and Description fields have to be filled in by activating the Submit button!

Once the user has entered the information, he can send it by clicking on "Submit". The contents will then be saved in firebase and a new location will appear visible in the app.

-> if it is successful, there is a success message
-> if there is an error, an error message is displayed

Screen 5

Detail-View



The Detail View displays the content of the spot selected in the Main View.

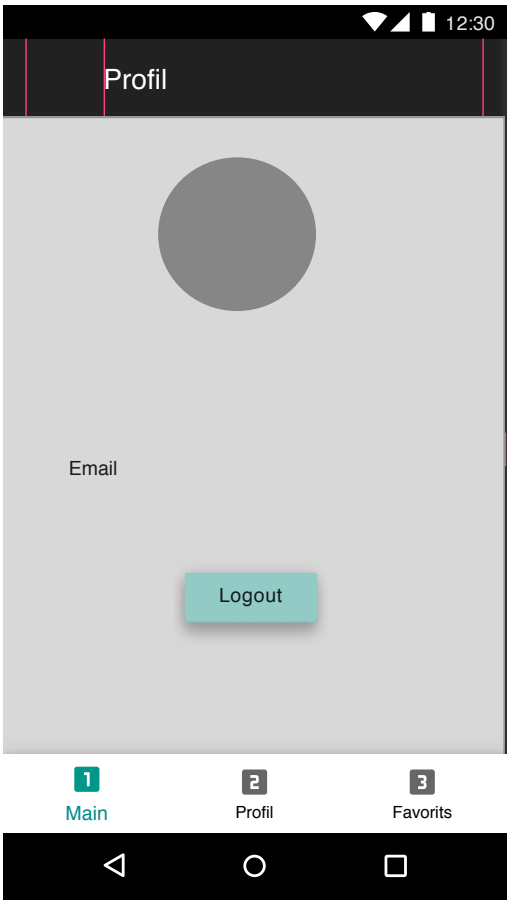
If the user clicks on the heart icon, the spot is added to its favorites list and the icon becomes an X with a red background.

If the user clicks on the X button, the entry is deleted from the favorites list again. In this view the X icon changes back to the heart icon of the list and the icon becomes a heart.

If he user clicks on the back arrow you get to the Main View.

Screen 6

Profile-View

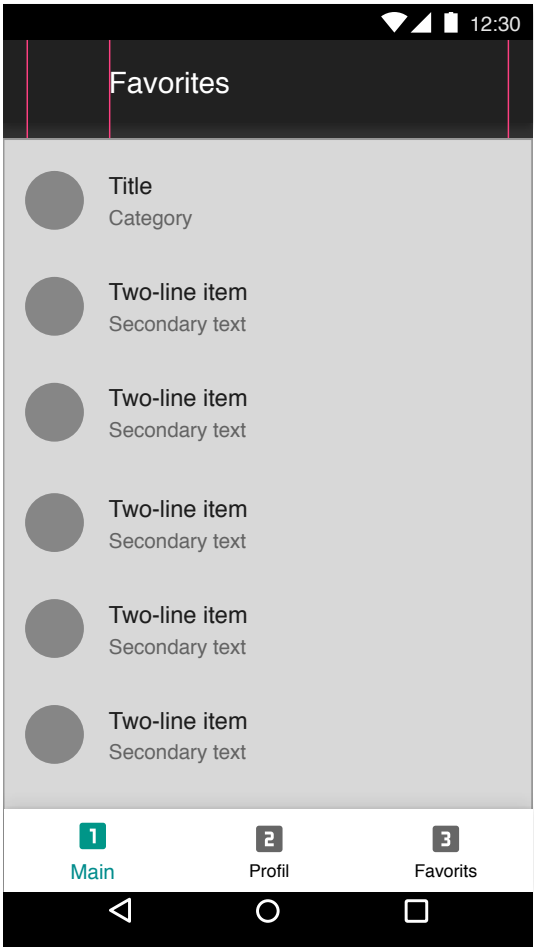


On this view the user can see all stored information about his profile.

If the user clicks on the Logout button, he is logged out and navigated back to the Sign in View.

Screen 7

Favorites View



This view displays all of the user's favorites.

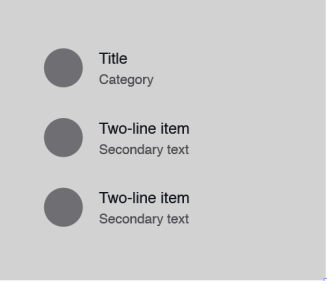
If the user clicks on an entry in the list, the user is navigated to the detail view of the entry.

If he user clicks on the back arrow you get to the Main View.

8. Bottom Navigation

A click on the Main icon navigates the User to Main View
A click on the Profil icon navigates the User to Profil View
A click on the Favorites icon navigates the User to Favorites View

9. Widget



This small view displays all of the user's favorites.

Key Considerations

How will your app handle data persistence?

Rooms And Firebase Realtime Database
I use for the favorites rooms and for other data (Spots and User Data) Firebase Realtime Database.

Describe any edge or corner cases in the UX.

The user cannot be pw and change email in the first expansion stage!

There won't be different layout versions for phones or tablets.

The user is notified with the snacker when there is no Internet connection, and while he will no longer be able to retrieve new Marker on the map, he can still show the locally stored favorites in favorites view.

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

Firebase Authentication - for the authentication.
Firebase Realtime Database - for store and sync data.
Firebase FireStore -Cloud Firestore is a flexible, scalable database for mobile, web, and server development from Firebase and Google Cloud Platform.
Firebase Storage - Developers use the Firebase SDKs for Cloud Storage to upload and download files directly from clients.
Butter Knife - for field and method binding for views.
GeoFire - allows you to store and query a set of keys based on their geographic location.
Retrofit turns your HTTP API into a Java interface

Glide to handle the loading and caching of images.
Gson can be used to convert Java Objects into their JSON representation

Libraries	Version
Firebase Authentication	19.1.0
Firebase Realtime Database	19.1.0
Firebase FireStore	21.1.1
Firebase Storage	19.1.0
Glide	4.10.0
Butter Knife	10.2.0
GeoFire	3.0.0
retrofit	2.6.2
Gson	2.8.6

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

AdMob for monetizing the app with banner ads.
FirBase for the Backend.

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

- Configure libraries
- Configure Firebase Auth
- Configure FireBase Realtime Database
- Build connectivity to Firebase

Task 2: Implement UI for Registration Activity, Login Activity, Profil

- Build UI Registration-Activity
- Build UI Login-Activity
- Build UI Main-Activity
- Build UI Profile-Activity
- Build UI Detail-Activity
- Build UI AddSpot-Activity
- Refer all the hardcoded strings from the strings.xml file.
- Enables RTL layout switching

Task 3: Implement Authentication with Firebase Auth

- Implement authentication with Firebase Auth
- Store user data in FireBase Realtime Database
- Implement offlinehandling
- Handle errors
- Handle permissions

Task 4: Implement MainView(Google Map)

- Create key in Firebase for Google Map
- Create GoogleMapActivity
- Build connectivity to Google Map
- Implement add spot function(navigate to AddSpotActivity)
- Implement show location markers(all Spots nearby) function
- Implement show Location function
- Implement onClick on a location marker to navigate to detail view function
- Handle errors
- Implement offlinehandling

Task 5: Store Spots-Object to FireBase Realtime Database

- Create FireBase Realtime Database
- Create FireBase Storage
- Create a Spots-Object
- Store Spots-Object in FireBase Realtime Database
- Store data FireBase using Retrofit or AsyncTaskLoader
- Handle errors
- Implement offlinehandling

Task 5: Store Spots-Object to FireBase Realtime Database

- Create FireBase Realtime Database
- Create FireBase Storage
- Create a Spots-Object
- Store Spots-Object in FireBase Realtime Database

- Store Image FireBase Storage
- Handle errors
- Implement offlinehandling

Task 6: Store User-Object to FireBase Realtime Database

- Create a User-Object
- Store User-Object in FireBase Realtime Database
- Store Image FireBase Storage
- Handle errors
- Implement offlinehandling

Task 7: Store Favorites to Rooms and Displayed the Favorites

- Create a Favorites-Object
- Implement delete functionality
- Implement update functionality
- Store Favorites-Object in Room
- Fetch Favorites and displayed to Recycler View
- Handle errors
- Implement offlinehandling

Task 9: Leftovers

- Implement Widget that displayed the user's favorites.
- Implement Google Play services and Ads

Submission Instructions

- After you've completed all the sections, download this document as a PDF [File → Download as PDF]
    - Make sure the PDF is named "**Capstone_Stage1.pdf**"
- Submit the PDF as a zip or in a GitHub project repo using the project submission portal

If using GitHub:

- Create a new GitHub repo for the capstone. Name it "**Capstone Project**"
- Add this document to your repo. Make sure it's named "**Capstone_Stage1.pdf**"