How to Use this Template

1. Make a copy [File → Make a copy...]
2. Rename this file: "**Capstone_Stage1**"
3. Replace the text in green

Submission Instructions

1. After you've completed all the sections, download this document as a PDF [File → Download as PDF]
2. Create a new GitHub repo for the capstone. Name it "**Capstone Project**"
3. Add this document to your repo. Make sure it's named "**Capstone_Stage1.pdf**"

GitHub Username: dschmucki

Catch

Description

Want to know when the next public transport towards your home or work leaves? 'Catch' will help you. Just tell 'Catch' a station on your way and it will always show you how many minutes you have left until your ride departures.
Of course a handy map view of the stations next to you is also included.

Intended User

Everybody who is regularly using public transport in Switzerland.

Features

- Shows next departure of public transport towards home/work
- Shows nearby stations on a map
- Allows to configure a set of geofences, for which a destination can be defined.
- A widget which shows the next departure at the station nearest to you.

User Interface Mocks

These can be created by hand (take a photo of your drawings and insert them in this flow), or using a program like Photoshop or Balsamiq.

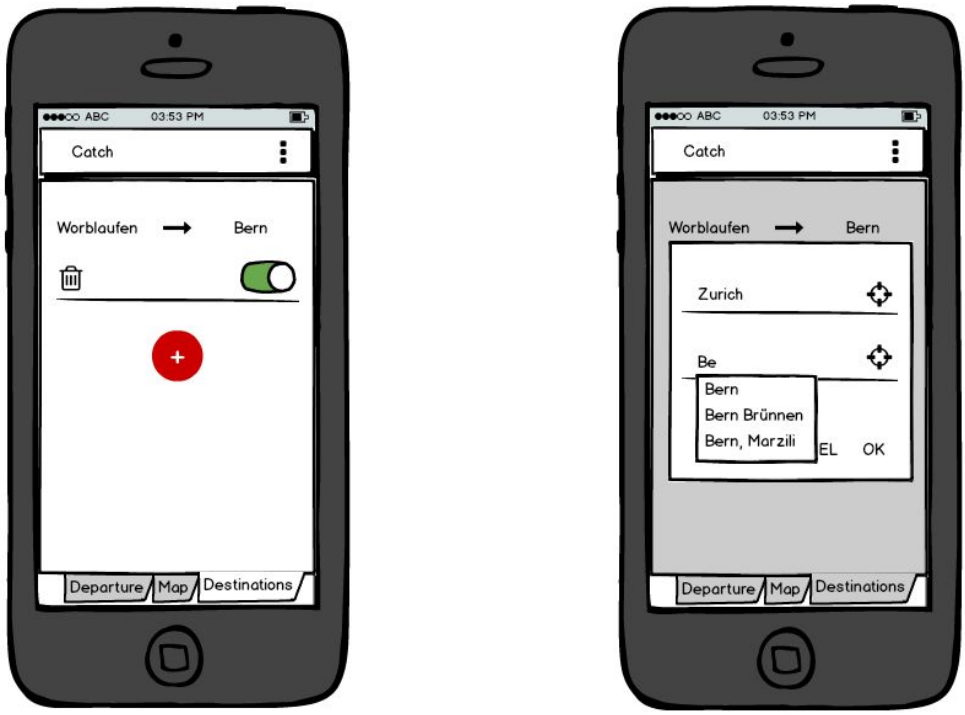
Screen 1 - Departures

The main screen. Shows a list of nearby stations, distance and a time in minutes, when the next transport is leaving towards a predefined destination. Also displayed are the minutes from now for the following transports.

Screen 2 - Map

The map screen shows the area around you, the stations in this area and an indicator, when the next transport is leaving.

Screen 3 - Destinations



The destination screen is intended to help you manage your geofences. You add a starting point (the coordinate of this place is then used to search nearby stations) and a destination where you want to travel. You can add several entries, so the app will always give you the right information

Screen 4 - Tablet

When in tablet mode, the map will become part of the departure view

Key Considerations

How will your app handle data persistence?

The timetable data will be requested from http://transport.opendata.ch/ As there is no realtime information available at the moment, timetable information will be cached on the device (for one day). A new Content Provider will be used for providing the relevant data (nearby stations, timetable). The data the Content Provider uses will be updated at least daily or as soon as the user enters a new location.

Describe any corner cases in the UX.

The user navigates between the three tabs by selecting them at the bottom of the screen. Rotation on a tablet leads to a different view, containing departures and map on one screen. Adding destinations is handled by a dialog, which provides autocompletion for station names and a map view for choosing a place.

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

Retrofit is used as REST client, to simplify the communications part.
Butterknife for view injection, to remove some boilerplate code.
Google Play Services for map and location.

Next Steps: Required Tasks

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

Task 1: Project Setup

- Create project structure / folders
- Set up Gradle files
    - Configure Android Plugin
    - Configure Dependencies

Task 2: Implement UI for Each Activity and Fragment

- Build UI for departures
- Build UI for map
- Build UI for destinations
    - Create dialog for adding destination
- Build UI for tablet (combination of departures and map)

Task 3: Retrieve data from transport.opendata.ch

- Read API documentation
- Build simple request to retrieve data
- Design data model for handling data on device
- Build test cases for retrieving data

Task 4: Persist data on device

- Use knowledge learned from task 3 to create a Content Provider
- Build test cases for Content Provider and implement functionality

    - Retrieve nearby stations
    - Search for transports from station to given destination
- Add capability to sync Content Provider with transport.opendata.ch
    - Check if data for today and station/destination is available
    - Retrieve data from service and persist it
- Persist user configured destinations in SharedPreferences

Task 5: Update UI

- Use Content Provider to automatically update the UI

Task 6: Location and Map

- Use location services to get position of device for searching nearby stations
- Implement displaying nearby stations on map
    - Add departure of next transport to station marker on map

Task 7: Widget

- Create a widget showing the next departure from the nearest station

Task 8: Pack and ship

- Make app ready for submission
    - Cleanup
    - Check Lint
    - Signing

Add as many tasks as you need to complete your app.

Submission Instructions

1. After you've completed all the sections, download this document as a PDF [File → Download as PDF]
2. Create a new GitHub repo for the capstone. Name it "**Capstone Project**"
3. Add this document to your repo. Make sure it's named "**Capstone_Stage1.pdf**"