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: WojciechCz

Juwenalia PWr 2016

Description

Application dedicated for Students Carnival 2016 in Wroclaw. This application will deliver to user up to date information about event. It will contains timetable, sponsors, partners, organizers information screen.

Intended User

Every student participating in Carnival.

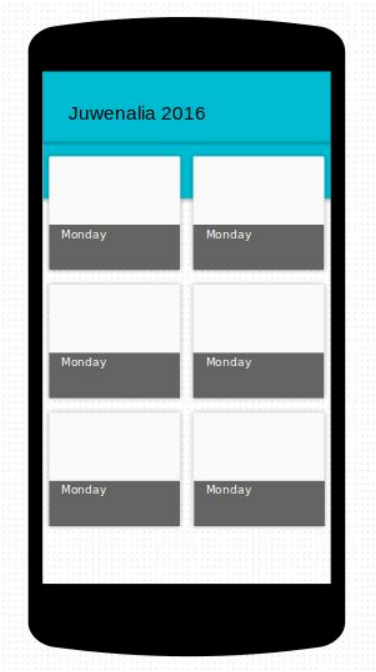
Features

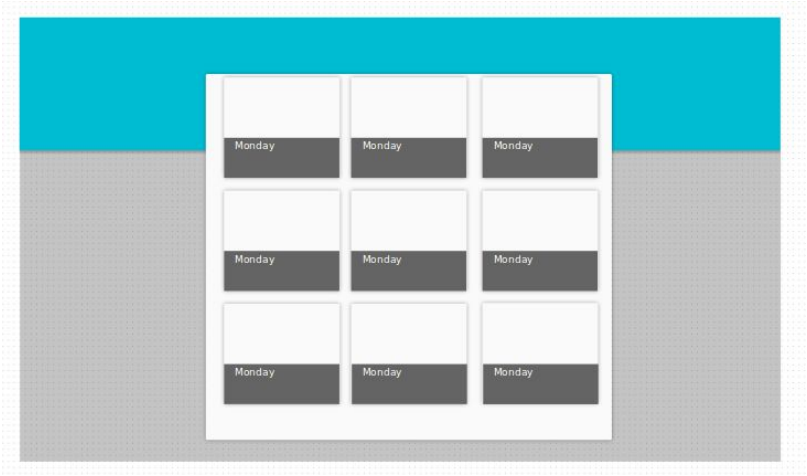
List the main features of your app. For example:

- Shows timetable
- Shows sponsor info
- Shows organizers contact info
- Shows app author info
- Static map - displays event image custom map
- Creating event remainder

User Interface Mocks

Start screen - event days list



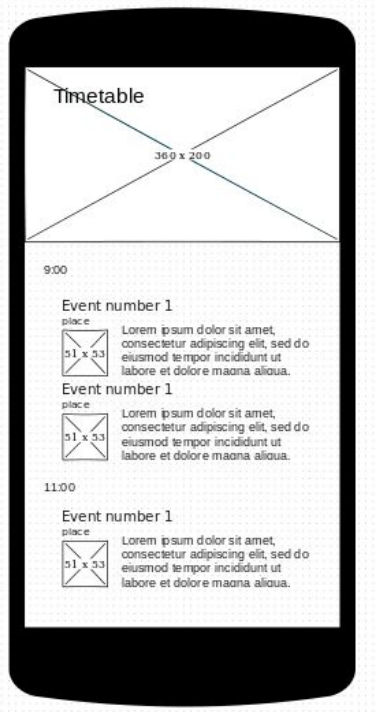


Smartphone

Tablet

User on application start can see list of event days. Each day is represented by image and name of the week.

Screen - event in selected day list

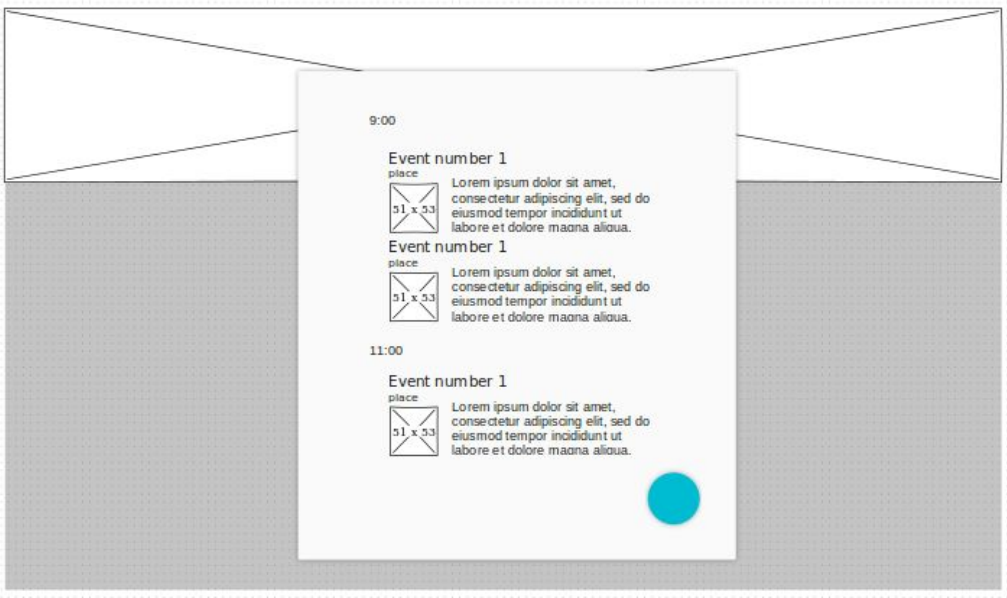


After selecting event day user can see list of events in this day. Events are group in hours. Toolbar displays event day main image.
Every event item holds:

- event name
- event image
- event short desc
- event place

Every hour item holds start hour

Smartphone



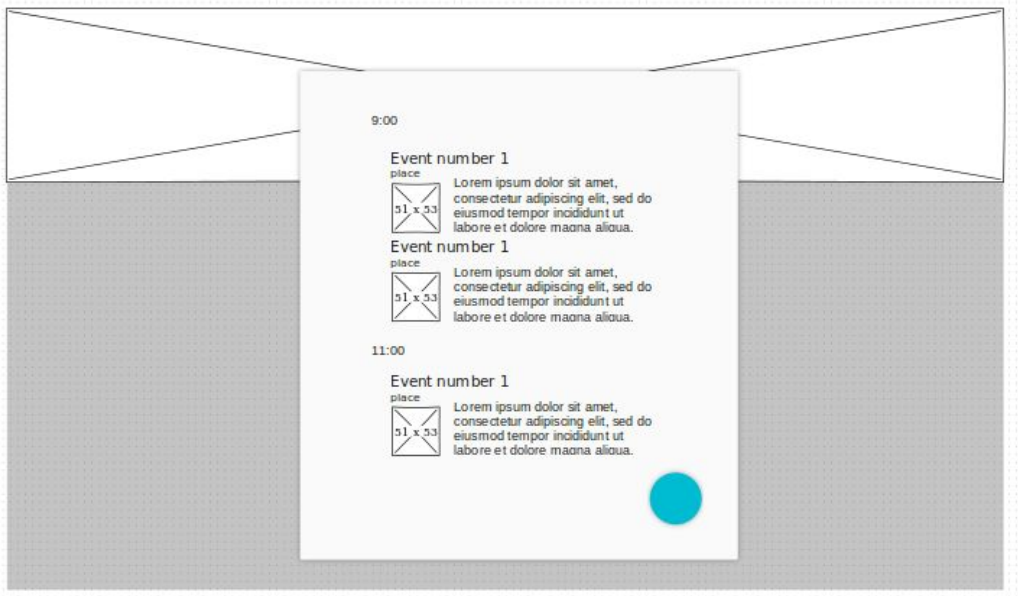
Tablet

Screen - event details

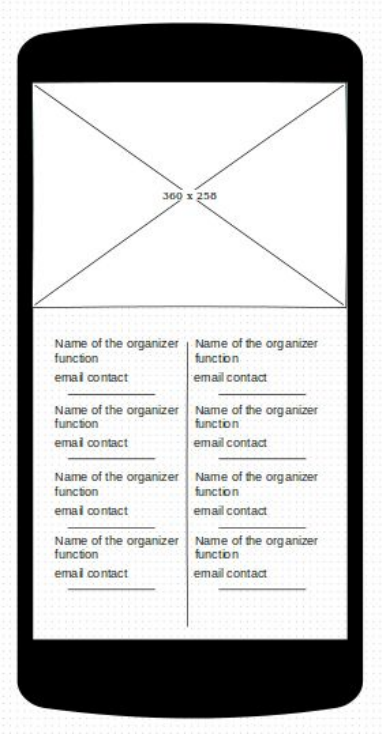
This screen shows event details. User can check full event description and add remainder for event. Toolbar displays event image.

Smartphone

Tablet



Screen - organizers screen



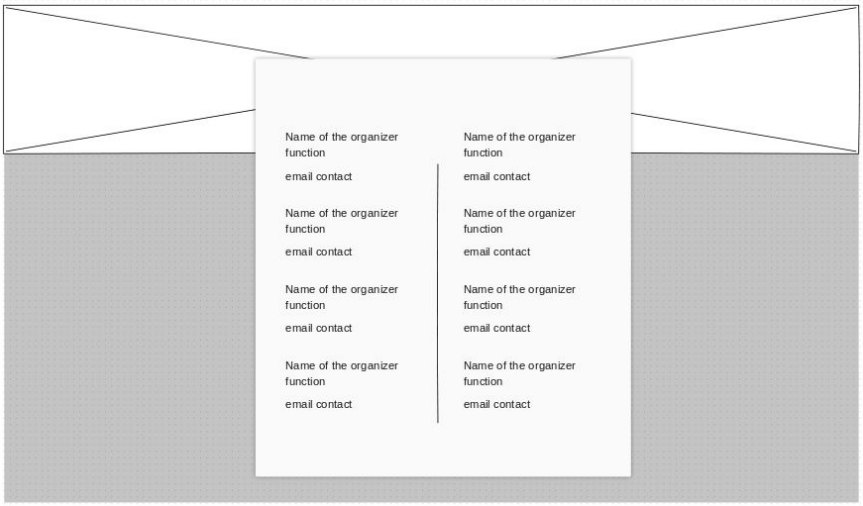
This screen shows Juwenalia 2016 organizers. Organizers are shown in grid. Every grid item contains:

- organizer name
- organizer function
- organizer email

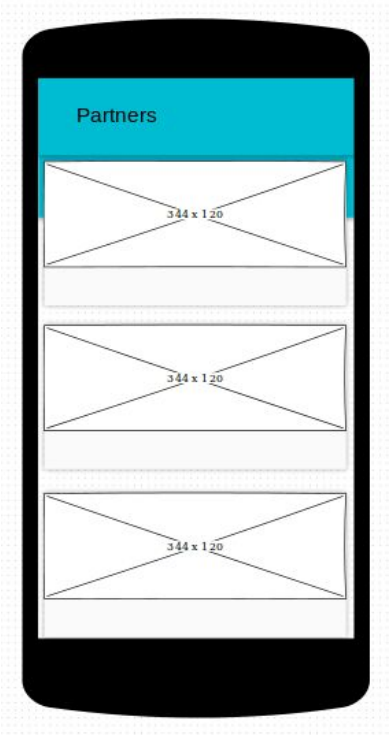
Toolbar contains photo of all organizers or Juwenalia 2016 logo (it depends on what content will be available).

Smartphone

Tablet



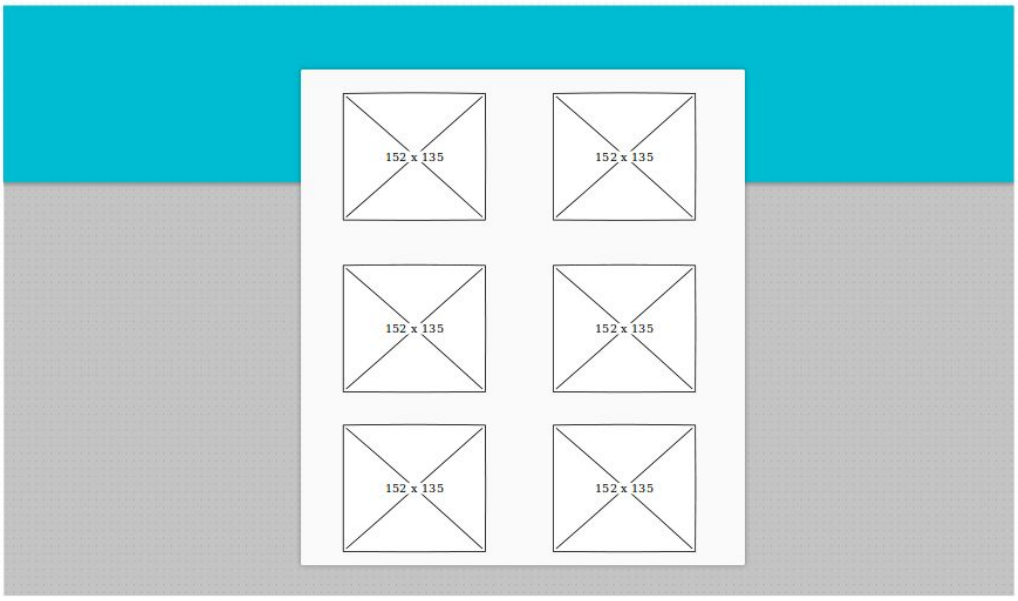
Screen - partners screen



This screen shows user all partners which agreed to cooperate with organizers on creating whole event. Partners are shown in list. Every list item contains partner logotype. After click on partner logotype application will open web browser with partner official web page.

Smartphone

Tablet



Key Considerations

How will your app handle data persistence?

I will use newest tool, which replace SQLite in android application: ***realm.***

Describe any corner cases in the UX.

From main screen with event days user can navigate to event day timetable. When user is on particular timetable screen he can choose event to see description screen.

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

- Glide for handling image download, caching and loading.
- Retrofit 2.0 & GSON - for connect to REST API and parsing JSON's..
- RxJava & RxAndroid - to make background work(eg. network downloading).
- Retrolambda & ButterKnife - to make code simpler.
- Support Annotations - to make code simpler.
- Support Design - for 5.0 features.

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

Steps:

- Install JDK 1.8.
- Install Android Studio - from stable channel.
- Download latest Android SDK.
- Create new android project.
- Check if gradle plugin is in latest version in build.gradle file in main project folder.
- Add required libraries do build.gradle file in “app” module.

Task 2: Implement UI for Each Activity and Fragment

Steps:

- Create ActivityMain
- Create FragmentDays
- Create FragmentEvents
- Create FragmentPartners
- Create FragmentOrganizers
- Build FragmentDays xml layout and bind views.
- Build FragmentEvents xml layout and bind views.
- Build FragmentPartners xml layout and bind views.
- Build FragmentOrganizers xml layout and bind views.

Task 3: Build REST API interface

Steps:

- Create Util class which will handle rest api requests.
- Create interface which will mock REST API
- Add to interface all requests
- Build methods for downloading date and passing response

Task 4: Create fragment’s flow in Activity

Steps:

- Add IntDef annotation for fragments id’s.
- Make method for opening fragment’s.
- Build method for managing fragments callbacks.

Task 5: Use REST API interface

Steps:

- Create method for make http request.
- Handle http response.
- Display downloaded data.
- Display errors if occurs.