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: fabio2035

MyFinance

Description

MyFinance is a personal budget manager app which makes managing personal savings a fun and involving activity. The app acts as a financial advisor alerting the user when certain budgets are at a critical point as well as keeps track of all spendings and purchases done by the user.

Intended User

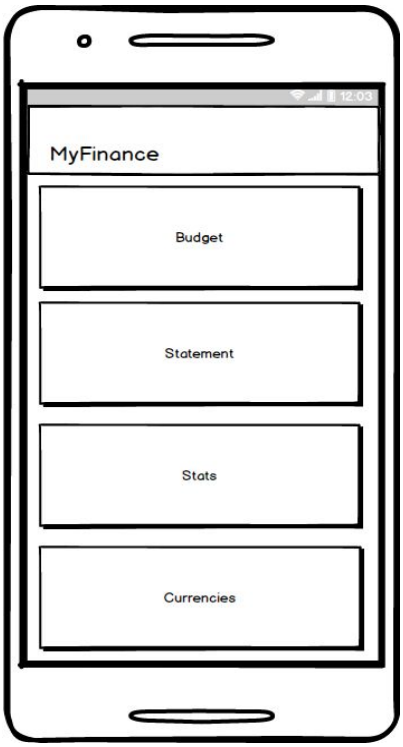
This app is intended for all users that are economically active.

Features

- Saves transaction data
- Shows budget graphs
- Enables bank statement personalization
- Sends notifications of purchases
- Sends alerts of budget status

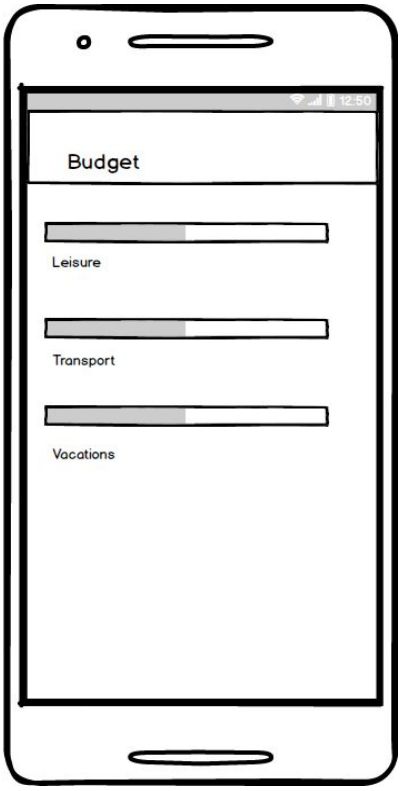
User Interface Mocks

Main Menu



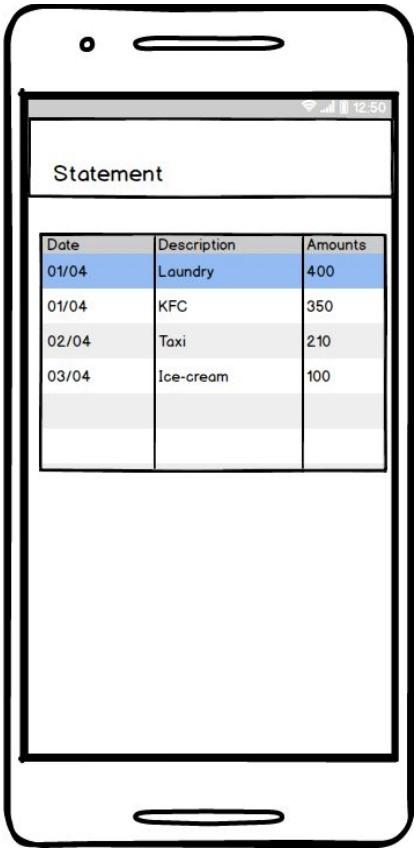
The main menu of the application will provide a common port for all the options accessible to the user, including: Budget, Statement, Stats and currencies analysis.

Budget Screen



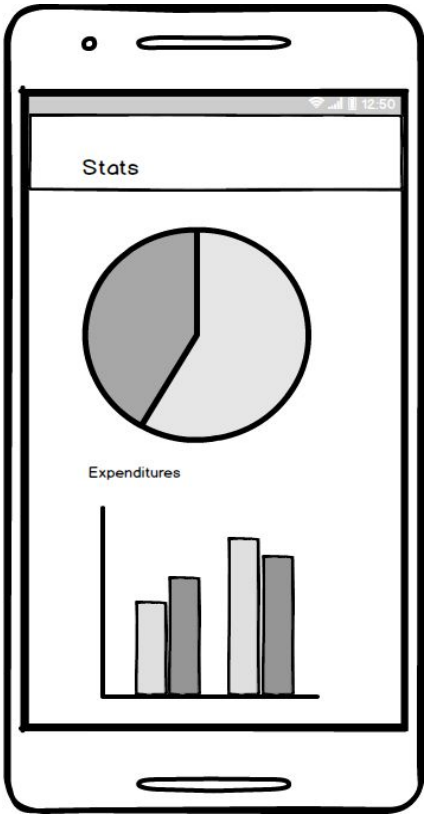
The budget screen shows the monthly/yearly progress as budgets are consumed in each representative category

Statement Screen



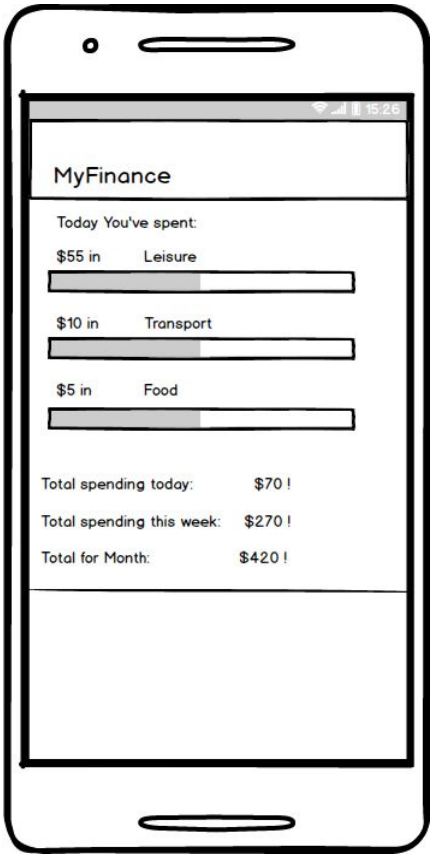
The statement screen displays an editable bank statement

Stats Screen



The stats screen displays a generalized panoramic view of what are the main expenditures build-up throughout a given period.

App Widget



The widget has a resumed information on total spendings of the current day, week and month so the user keeps updated with the day's progress.

Key Considerations

How will your app handle data persistence?

The app has a SQLite database that stores up to 3 months worth of statement data. All other data is stored in central server and is downloaded through a content provider when user wants to see historical data.

Data will be periodically updated using a SyncAdapter service

The views will be filled using a loader

The app will provide a widget that will allow the user to verify basic finance information

Describe any corner cases in the UX.

The app is designed to have a simple interface, and navigation will consist of choosing option from main menu and accessing contents from a fragment. Navigating back works through the usage of device's back button or the back button on the toolbar.

App includes support for accessibility, including content description.

App includes RTL layout switching on all layouts

App also includes appbar and associated toolbars

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

Libraries used:

MPAndroidChart - For displaying stat graphs, including pie charts and bar charts

Picasso - For loading predefined icons that represent different budget categories

Describe how you will implement Google Play Services.

AdMob - For displaying advertisements in free version of app
Google Cloud - for storing user personalized statements

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

1.Database design and normalization - As this is a data-centered application, the database is one of the key elements which has to be robust and allow for optimum data loading, updating and inserting;

2. Setup mock database on google cloud to represent user's bank data server

Task 2: Implement UI for Each Activity and Fragment

1.Setup the main layouts for all the screens, including wide screens and smaller screens, with basic code to enable simulation

Task 3: Check Connectivity to Data Server

1. Setup connection with main data server, and make sure that data is retrieved and that all key names are recognized and can be referenced for database insertion/updates;

2. Make sure that update services retrieve data and data is updated accordingly and in timely manner

3. Make sure that all network errors are recognized and appropriately handled with Toast notifications for user

Task 4: Finalize layouts

1. Finalize layouts with retrieved data, and make sure that statements and graphs and filled accordingly and change dynamically according to stored data

Task 5: Check all functionalities

1.Check that all functionalities work appropriately

2.Make sure that all data input by user is accounted for with appropriate validations

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**"