Capstone_Stage1

How to use this template

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

GitHub Username : Your GitHub username

Radio

Description

Radio player with the ability to add to your favorite radio station. You can enjoy your favorite programs and songs.
If the connection is poor, the application will automatically resume the broadcast after the signal is lost.

Assigned user

Music lover.

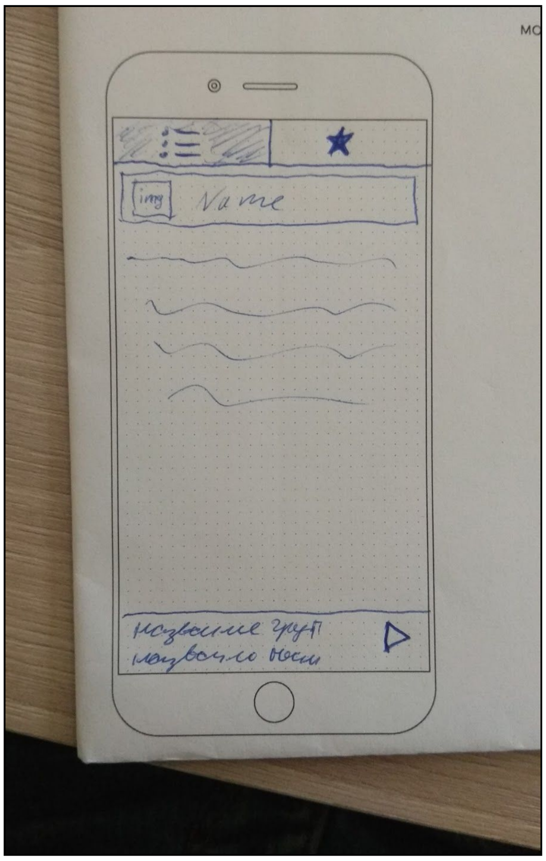
Features

- Play Radio
- Save to Favorites
- Restore communication

Mocks user interface

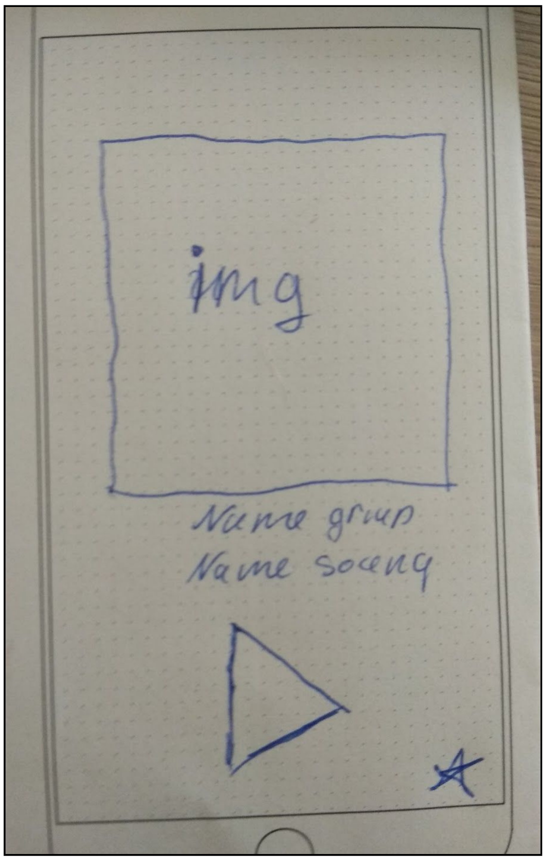
They can be created manually (take a snapshot of your drawings and paste them into this thread) or using a program such as Google Drawings, www.ninjamock.com , Paper by 53, Photoshop or Balsamiq.

Screen 1



The main screen with a list of radio stations and a mini player showing which radio station is currently playing.

Screen 2



Player in which the name of the song and group is displayed. Button with the ability to add to favorites.

Key Considerations

How will your application maintain consistency of data?

In Firebase there will be a database with a list of radio stations and the version of the database. At the first download, the entire list will be unloaded. When you enter the application, the versions of the database will be checked and if the base becomes obsolete on the phone, a newer version will be downloaded. Search through radio stations.

Describe any extreme or angular occurrences in UX.

The player will be executed in service with notification.

Describe all the libraries that you will be using, and share your arguments to include them.

Glide - Upload images.
RxJava2 - Multithreading.
Dagger2 - Get rid of direct dependencies.

Moxy - A helper for implementing the viewstate pattern.
ButterKnife is a replacement for findViewById.
Firebase - connect to Firebase.
ExoPlayer - the main player.
Room - To manage the local database.

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

I will not implement the services.

The following steps: the required tasks

This is a section where you can use the main functions of your application (announced above) and split them into tangible technical tasks, which you can perform one at a time, until you have a ready-made application.

Task 1: Setting up the project

- Import the required libraries in the Gradle.
- Create a DB in Firebase.
- Tune Firebase in the application.

Task 2: Implement the user interface for each operation and fragment

- Create an interface for the list of radio stations.
- Interface for the player.
- Interface for notifications.
- Adapter for items RV.

Task 3: Display the information on the screen

- List the radio stations from the common database.
- List the radio stations from Favorites.
- Ability to add to your favorites.
- Search.

Task 4: Playing Radio

- Create a service for streaming audio playback.
- Create notification management of the player.

- Interrupt playback when the headphones are disconnected.
- Interrupt playback when an incoming call is received.

Task 5: Prepare an attachment for publication.

- Create page in Google Play
- Make screenshots.
- Add description.

Add as many tasks as needed to complete your application.

Feed instructions

- After all sections have been completed, download this document in PDF format [File → Download as PDF]
  - Make sure that the PDF file is named " **Capstone_Stage1.pdf** "
- Send PDF as a zip or to the GitHub project repository, using the project's presentation portal

If you are using GitHub:

- Create a new GitHub repository for the main stone. Call it the " **Capstone Project** "
- Add this document to your repo. Make sure it's called " **Capstone_Stage1.pdf** "