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: kwiecien

World Cup 2018

Description

Brace yourself, FIFA World Cup 2018 is coming!

If you want to stay updated with:

- teams,
- fixtures,
- and scores,

just download this app and enjoy the biggest sport event this year!

Intended User

Football fan

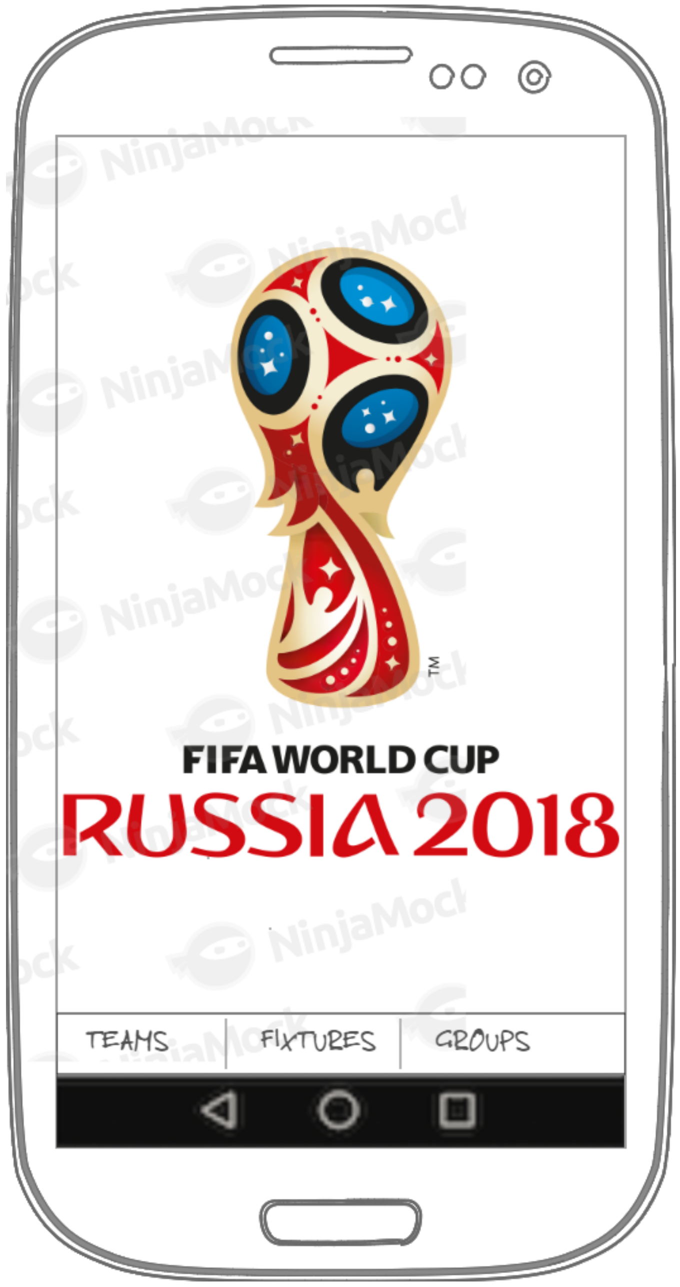
Features

- Fetch data from http://api.football-data.org/v1
- Save information to a local database
- Present data in a beautiful way
- Choose favorite team and get notifications
- Share scores with friends
- The application will support accessibility by: providing content description for graphical elements; creating easy-to-follow navigation; making touch targets large; making font readable and customizable (sp); providing adequate color contrast; using cues other than color
- The resources (colors, strings, themes etc.) will be stored in theirs xml files
- The application will implement JobDispatcher for regularly fetching scores in the background
- The application will implement AsyncTaskLoader for fetching the up to date data on demand when entering new screens
- The application will be written solely in the Java Programming Language
- The application will be developed in Android Studio 3.1.3 and built with Gradle 4.4

User Interface Mocks

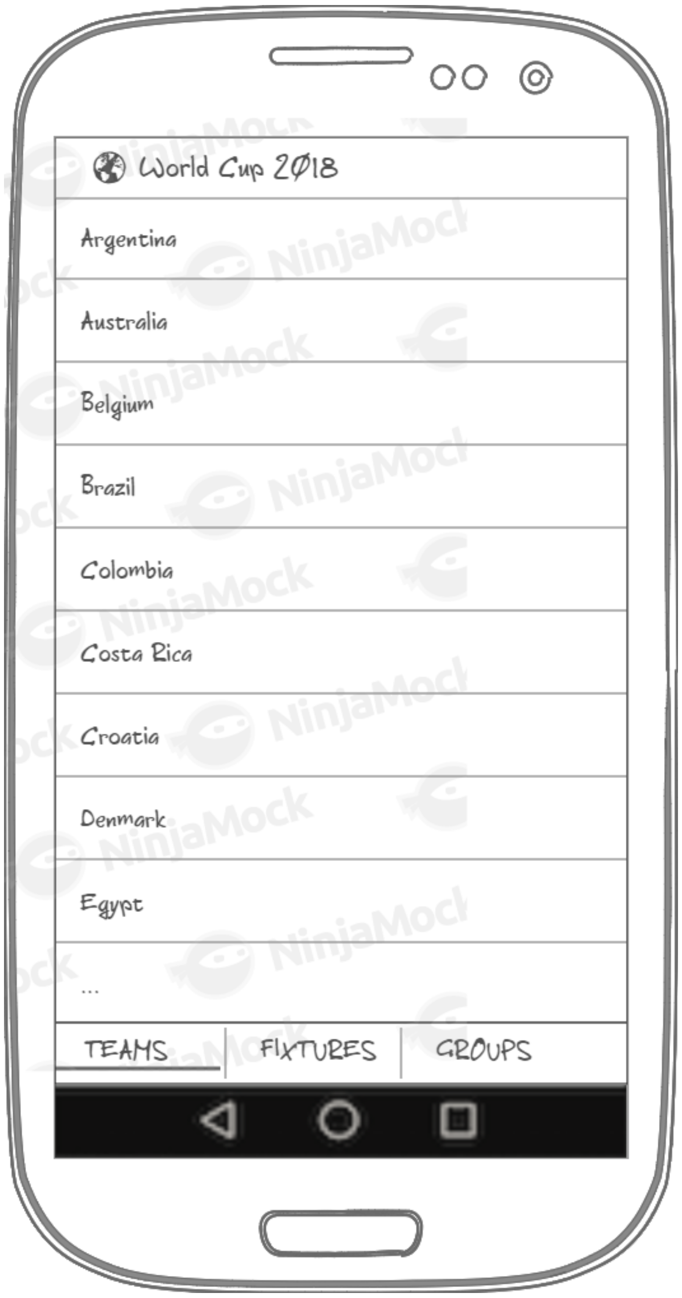
These can be created by hand (take a photo of your drawings and insert them in this flow), or using a program like Google Drawings, www.ninjamock.com, Paper by 53, Photoshop or Balsamiq.

Screen 1



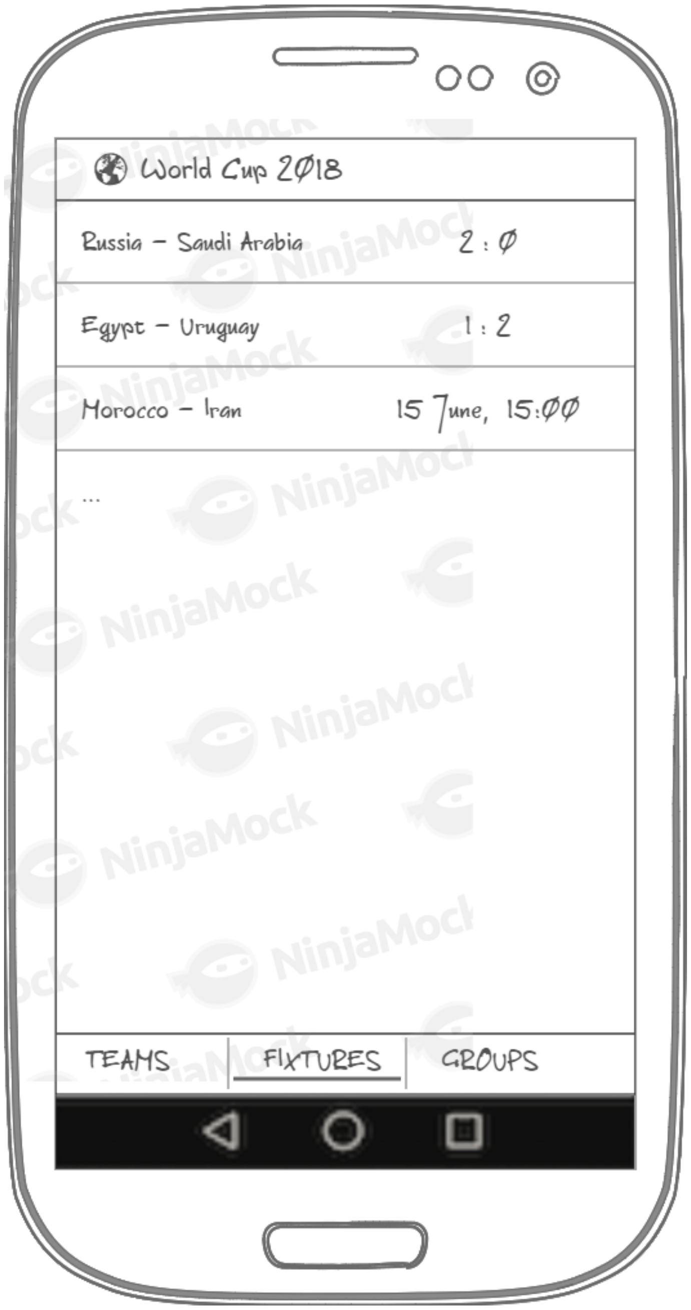
Welcome screen of the application. It displays the World Cup 2018 logo and a bottom navigation view.

Screen 2



Teams screen. It displays all the teams participating in the WC2018. Clicking on each team opens the team details view.

Screen 3

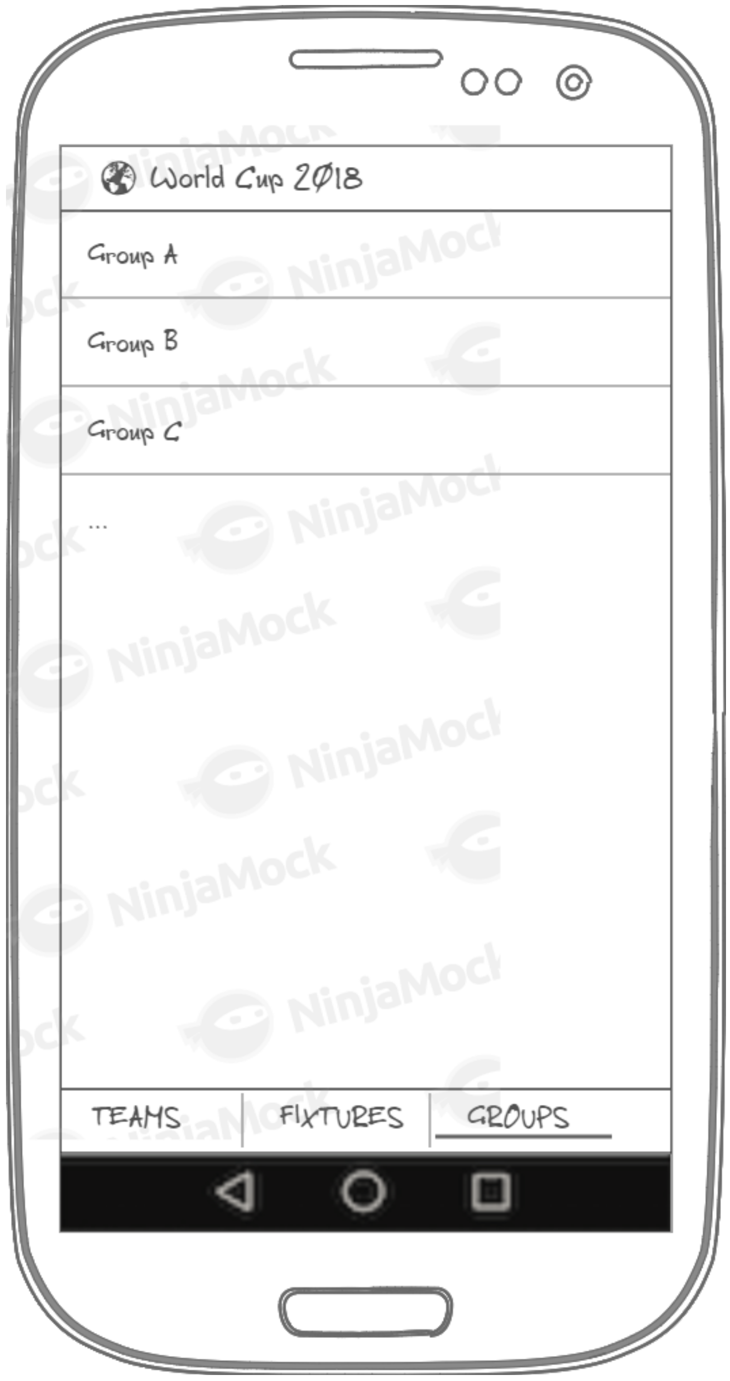

World Cup 2018

Russia – Saudi Arabia	2 : 0
Egypt – Uruguay	1 : 2
Morocco – Iran	15 June, 15:00

...

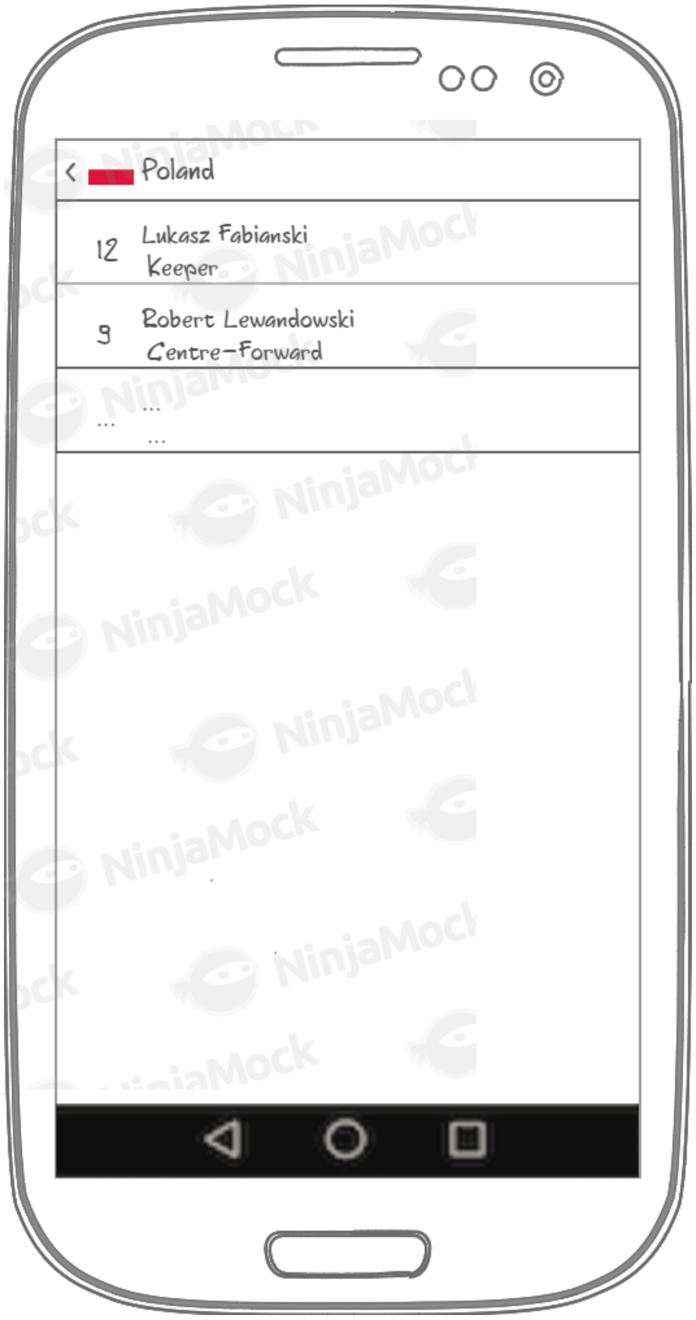
Teams screen. It displays all the teams participating in the WC2018. Clicking on each fixture opens the fixture details view.

Screen 4



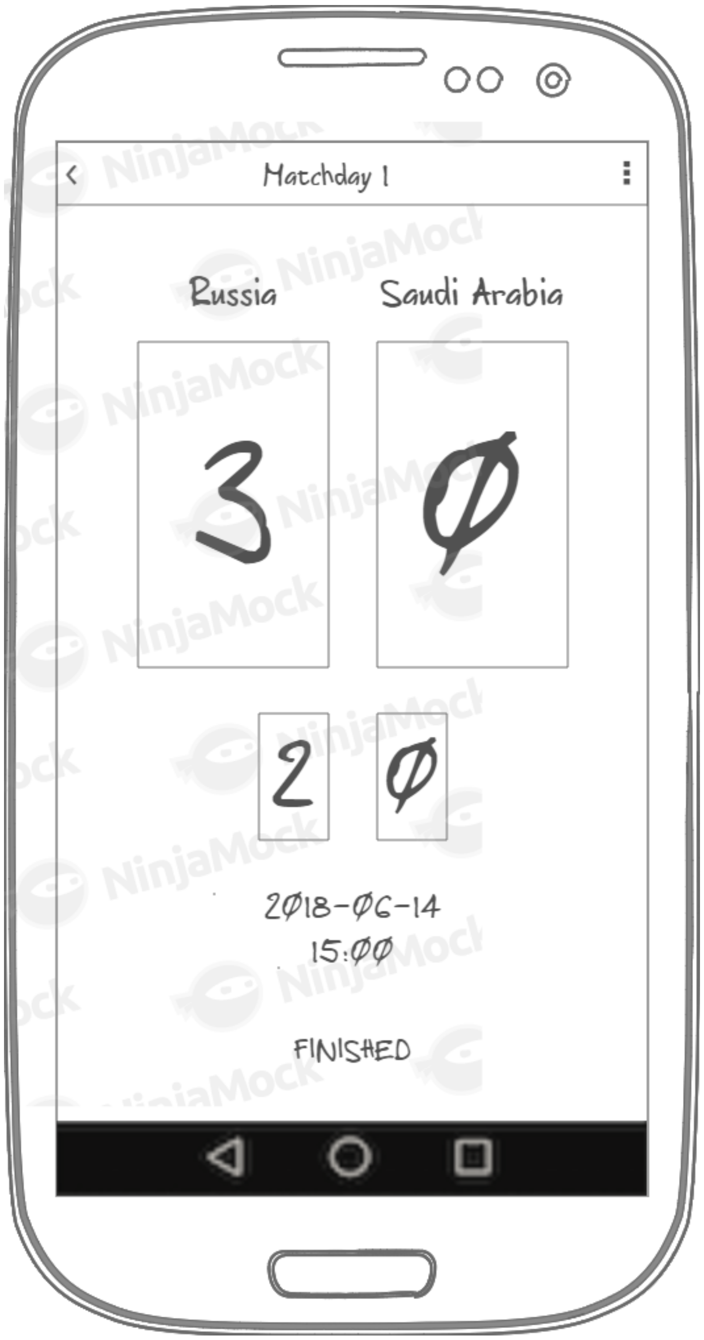
Groups screen. It displays all the groups in the WC2018. Clicking on each group opens the group details view.

Screen 5



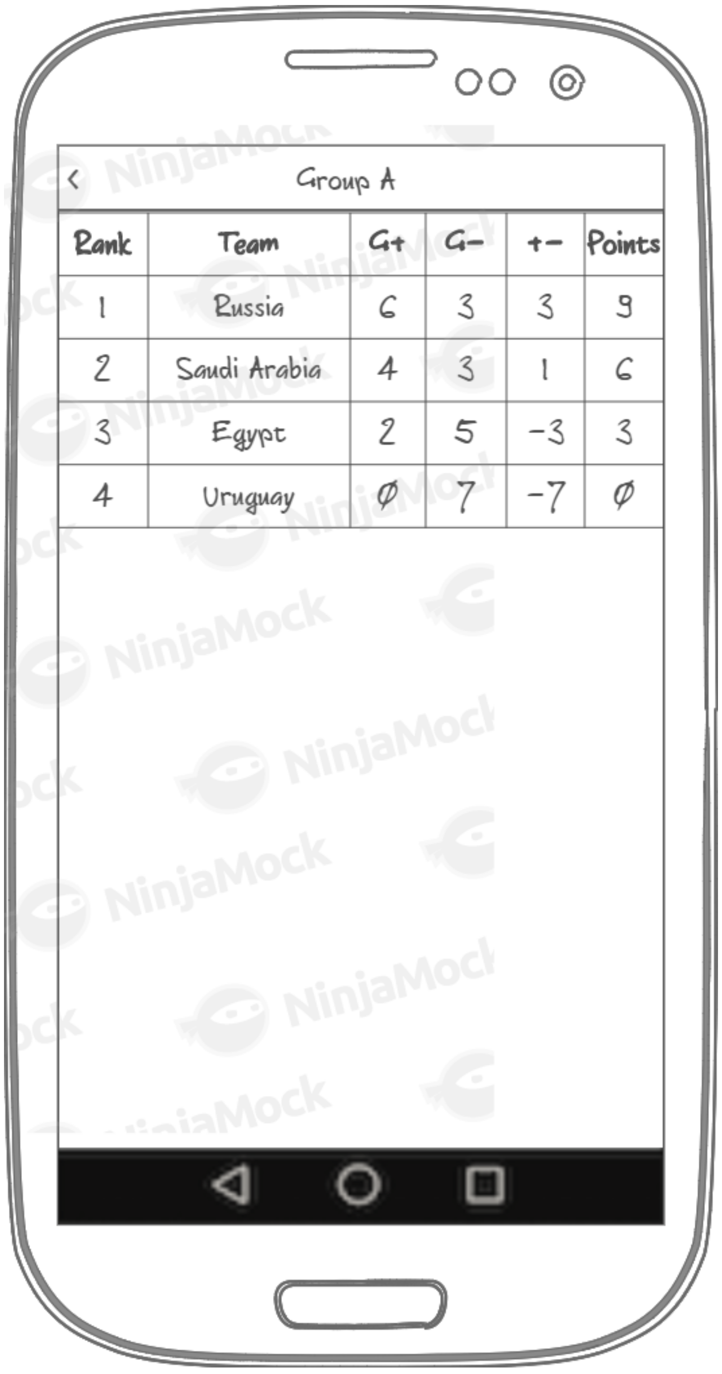
Team screen. It displays a list of the team’s players with their position and jersey number.

Screen 6



Fixture screen. It displays the playing team's names and score of the game. There is also information about when the game takes place and the status of the game. User can share the fixture's result.

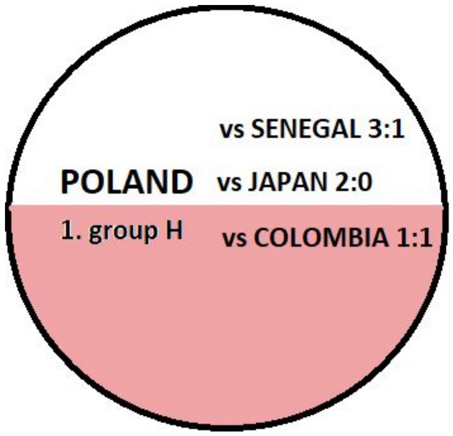
Screen 7



Rank	Team	G+	G-	+-	Points
1	Russia	6	3	3	9
2	Saudi Arabia	4	3	1	6
3	Egypt	2	5	-3	3
4	Uruguay	Ø	7	-7	Ø

Group screen. It displays the current group table. The table contains teams and their rank, goals and points.

Widget



Widget. It displays the favorite team information. The country name is always is displayed, the fixtures only if the widget’s size is big enough.

Key Considerations

How will your app handle data persistence?

The app will fetch data from http://api.football-data.org/v1. Jsons will be parsed to objects using Gson library. Objects will be stored in a database using ORM Room.

Describe any edge or corner cases in the UX.

In the fixtures screen, if the game has started, the score will be displayed; otherwise the match date and time will be displayed.
Widget will only display additional data if it is big enough.

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

Glide (4.7.1) to handle the loading and caching of images.
Parceler (1.1.10) to pass objects between activities and fragments easily and efficiently.
Room (1.1.0) to persist data.
Espresso (3.0.2), Mockito (1.10.19), Hamcrest (1.3) to test application.
Dagger (2.11) to have a dependency injection framework.
Retrofit (2.4.0) to handle endpoint calls.
Timber (4.7.0) and Hugo (1.2.1) to log.

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

Google Analytics as a measurement solution that will provide insight on app usage and user engagement.
Firebase Cloud Messaging (FCM) as a cross-platform messaging solution that will let me reliably deliver messages at no cost.

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

- Create git repository
- Create Android Studio project
- Add necessary dependencies

Task 2: Implement UI for Each Activity and Fragment

- Build UI for MainActivity
- Build UI for TeamsFragment
- Build UI for FixturesFragment
- Build UI for GroupsFragment
- Build UI for TeamActivity/Fragment
- Build UI for FixtureActivity/Fragment
- Build UI for GroupActivity/Fragment
- Build UI for FavoriteTeamWidget

Task 3: Fetch and persist data

- Create data classes
- Fetch data from endpoint using Retrofit
- Map data to classes using Gson
- Persist data using Room

Task 4: Implement Google Services

- Create accounts
- Implement Analytics according to the instructions
- Implement FCM according to the instructions

Task 5: Make app material

- Implement shared and content transitions
- Polish app visually

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**”