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: victorcocuz

Trivia

Description

Trivia is a quiz app that is meant to help you improve your general knowledge on a range of subjects, including Arts & Culture, Geography, History, Movies, Music, Science, Social Sciences and Sport. Each game contains 10 questions from different categories placed randomly. The question format is multiple choice and the answer to each question must be provided within 10 seconds.

You will receive points for correct answers. You will also receive bonus points for the time it took to answer, which will ultimately help you improve your answering speed as well as increasing

your knowledge. In the end all your points will be added up and the more you answer the more you increase in level.

Intended User

This game is intended for school and university students curious to learn at their leisure more than what is provided in their curriculum, but also for everybody else willing to find out new things and test their knowledge.

Features

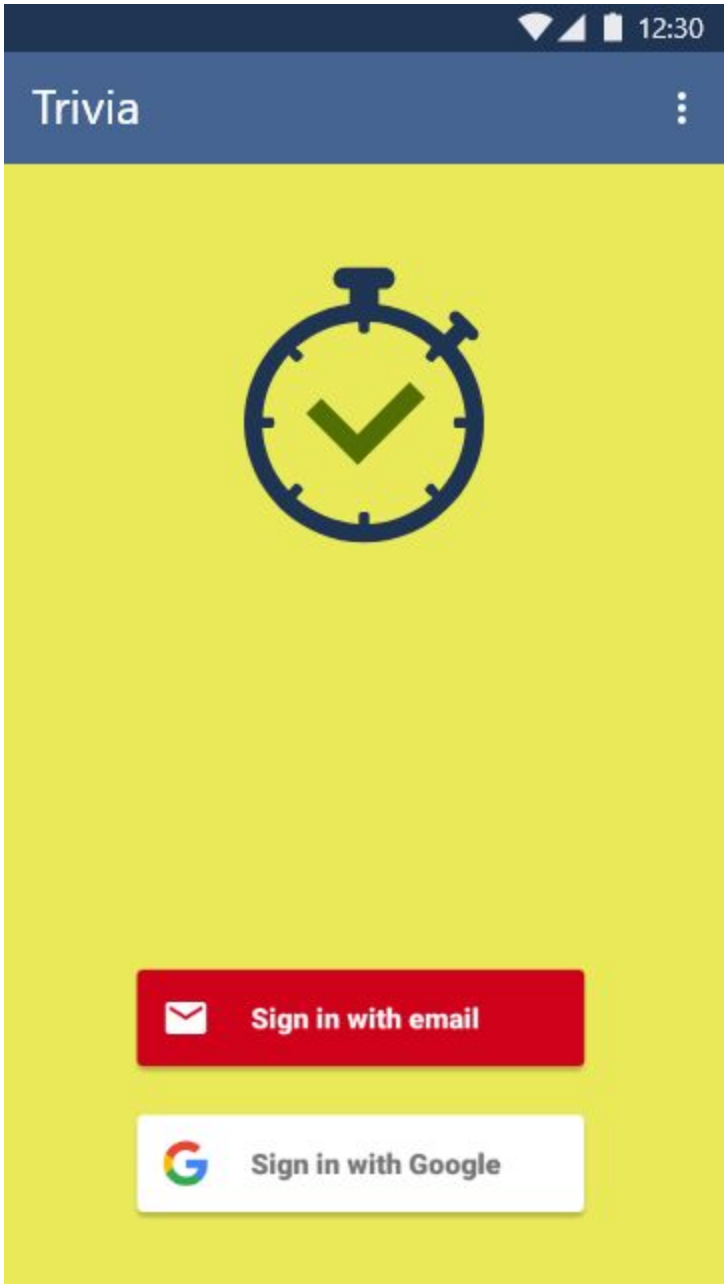
Main features of the app:

- Will be written solely in Java, by using Android Studio
- Will utilize stable release versions of all libraries, Gradle and Android Studio
- Provides a sign in method to store individual answers and calculate user score
- Offers the possibility to add questions for the quiz and stores them
- Asks questions in form of a quiz game
- Stores answers and score for each user
- Analyses response to break down user skills by category
- App will also follow accessibility guidelines. All strings will be kept in a strings.xml, will include content descriptions and navigation using a D-pad.

User Interface Mocks

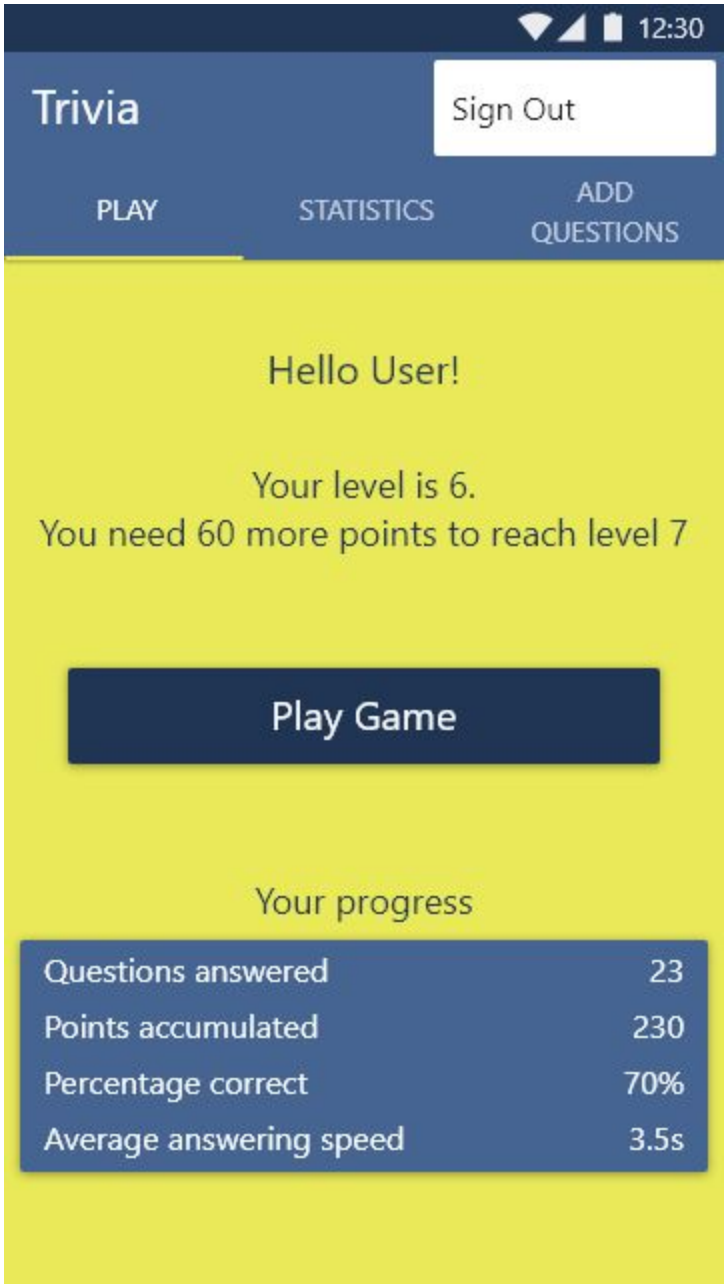
These can be created by hand (take a photo of your drawings and insert them in this flow), or using a program like Google Drawings, www.ninjamock.com, Paper by 53, Photoshop or Balsamiq.

Screen 1 - Login Screen



This is where the user has the ability to sign in. The logo will be shown here. The login will be done by using Firebase UI and Firebase Authentication.

Screen 2 - Play Fragment



This is where the user sees his progress and has the ability to start a new game. I am also showing here the possibility of logging out.

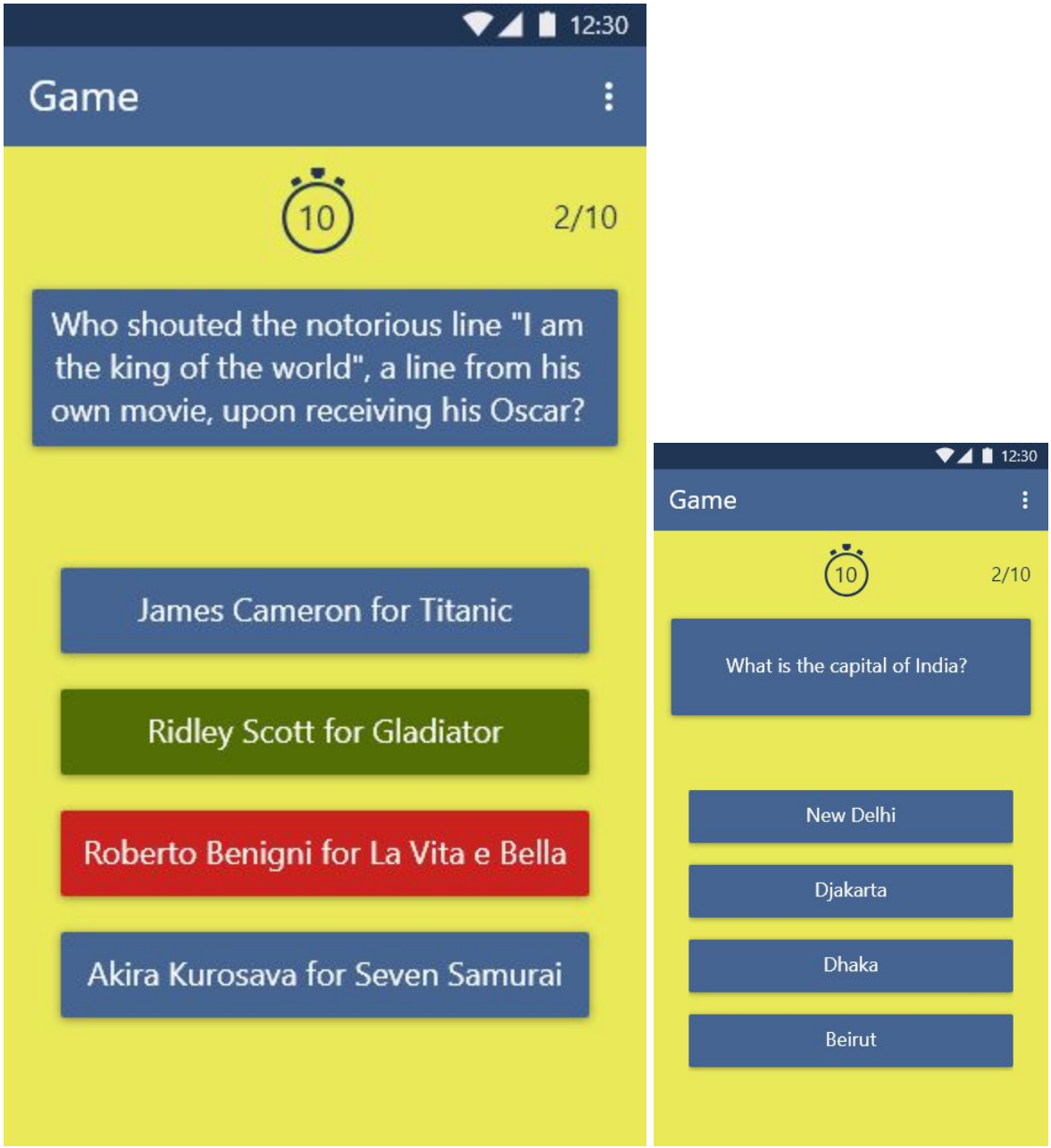
Screen 3 - Statistics Fragment

This is where the user sees the breakdown of his progress.

Screen 4 - Add questions fragment

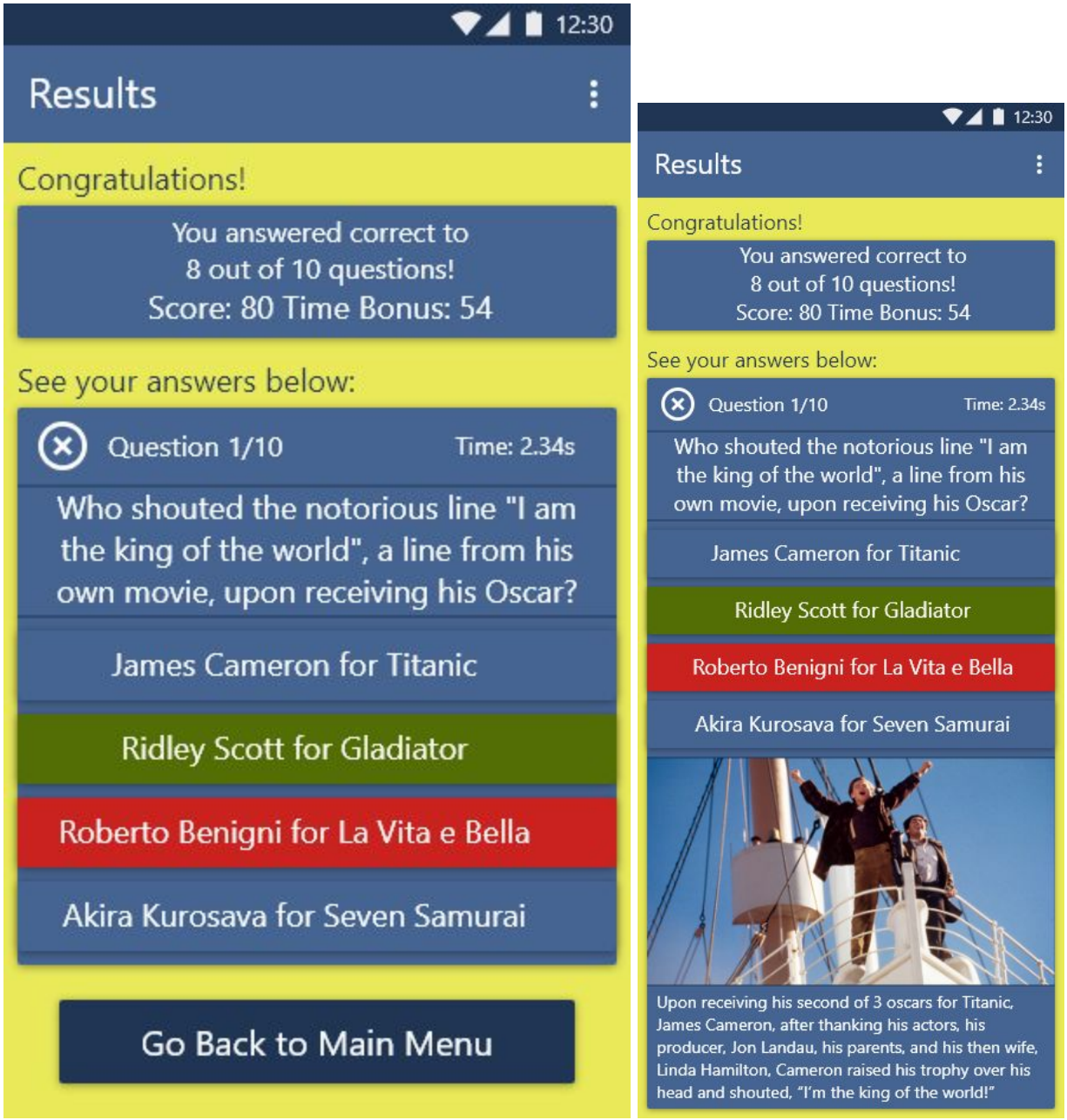
This is where the user has the ability to contribute to the list of questions stored in the database and add his own question. The second screenshot shows the full extent of this fragment.

Screen 5 - Game Activity



This is where the user actually plays the game. There will be 10 questions for each game. The user will have 10 seconds allocated for each question. The screenshot on the right shows the same activity with shorter text.

Screen 5 - Result Activity



This is where the result of the quiz is shown. This includes the score and all the questions answered. Further, it shows a description and an image on each question in a recycler view. On the right you can see the full extent of the card view showing information about the question.

Screen 6 - Widget

The widget will display curiosities taken from the quiz questions. The show me more button will show more curiosities. The ask me something button will open the app so that the user can play a new game.

Key Considerations

How will your app handle data persistence?

The app will use Firebase Realtime Database, but will also use a content provider to store some information locally upon sign in, in order to aid out in analysing the data. The local data will be added at sign in by using a content provider. It will then be queried for score analysis by using an Async Task that will return a Cursor loader. The local data will be removed on sign out.

Describe any edge or corner cases in the UX.

The Main Activity will contain three main fragments, as shown above:

- Play Fragment
- Statistics Fragment
- Add Question Fragment

The functionality between these will be very fluent, since a Pager Adapter will be used.

From the Play Fragment a new Game Activity will be created, which will create the quiz game. From this activity, when pressing the back button, it will lead back to the Main Activity and the game will be stopped.

When the game is finished, a new activity Result Activity will be created. From this activity, when pressing the back button, it will also lead back to Main Activity, to allow the user to create a new game, or to see the progress.

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

- Picasso - to handle the loading and caching of images.
- Timber - to help write the logs more efficiently
- Facebook Stetho - to read any local database

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

- Firebase authentication - will be implemented to separate answers and to be able to track progress for each user individually. This will be implemented by connecting the project to the Firebase Console, adding the Firebase authenticatication and the Firebase UI Auth implementations, getting the necessary references and adding a log in and log out listener. An action item inside the Action Bar will also be added to allow the user to log out at any point.
- Firebase database - will be implemented to store all questions, answers and current score on the Firebase server. This will be implemented by connecting the project to the Firebase Console, by adding the Firebase implementation, getting the necessary database references and adding the appropriate listeners

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

- Configure libraries
- Add the chosen color palette in the colors resources
- Start filling in resources such basic strings for names, basic dims for margins, text and containers, basic integers, arrays and styles
- Import resources for logos, launcher and icons

Task 2: Implement UI for Each Activity and Fragment

- Build UI for MainActivity
- Build UI for the three fragments in MainActivity - PlayFragment, StatisticsFragment, AddQuestionsFragment
- Build UI for GameActivity

- Build UI for ResultActivity
- Create card views for each Recycler View (for question, answers and statistics)

Task 3: Implement authentication

- Connect project to Firebase and turn on Authentication
- Add implementation for Authentication & Firebase UI Auth in gradle
- Add authentication listeners for login in MainActivity
- Add sign in and sign out functionality, including detaching and attaching other listeners

Task 3: Implement functionality for adding new question

- Get the text from all the completed fields
- Similar to above, add Firebase Realtime Database to the project
- Create a FirebaseDatabase reference and push the question to a certain path
- Clear question fields

Task 4: Implement functionality for playing game

- Download all questions by using a firebase child listener and add them to an adapter
- Connect the adapter to a recyclerView
- Add a onItemClick interface in the Adapter
- Add a time counter for each question and an integer counter for questions left
- Store all information in a ParcelableArray on each answer
- When quiz finishes, send all the answered information to the Result Activity

Task 5: Implement functionality for result activity

- Get all information from intent
- Add an adapter to show all answered questions and highlight wrong ones and connect to a recycler view
- Show the score and time bonus
- Add all the answered information to Firebase
- Also add functionality to add the points to preexisting points and show user level

Task 5: Implement functionality for statistics activity

- Create an AsyncTask to get all answers by their answer status (1 for correct) and by category from the local database
- Calculate what percentage is correct on each answer
- Order all answers by percentage and by number of answers
- Add an adapter to display statistics in a recycler view

Task 5: Final functionality and design

- Add widget
- Fix issues and make sure all values are kept in the values files and not declared manually
- Make sure all content descriptions and D-pad navigation support are implemented
- Finish design

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

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**"