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: groverankush

Let's Wiki

Description

Let's Wiki is a FREE game available for Android smartphones.. Let's Wiki lets you get through a boring day with a series of short and quick word guessing games. It fetches a random article from wikipedia, omit 10 words from it and displays the user. Then user has to figure out the missing words based on the article's context. The game offers two modes. You can choose as per your expertise. Let's Wiki presents a fun and interactive way to enhance your knowledge base.

Intended User

Let's Wiki is an ideal companion for knowledge frenzy people. Anyone who seeks to extend their knowledge base will find using Let's Wiki enjoyable.

Features

Let's Wiki is solely written in Java programming language. It presents a signup screen at first if the user is not signed up. As soon as user logs in and for subsequent launches app presents a launch screen where user can view his score and global rank, tweak the difficulty level and start a new game. The main game screen fetches a random wikipedia article and presents it to the user with some omitted words. User can tap on omitted words and choose the word which he feels correct. On pressing the button at the bottom center or on time expiry user is navigated to the scores' screen, where the score of the game is displayed. User can then choose to play again, check for the correct answers or move to the launch screen. These are some key features

- Fun and Interactive design
- Pro mode for added complexity
- Compete against the world

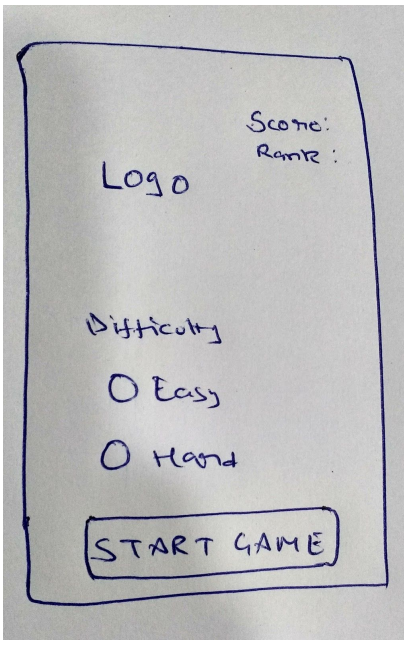
Design Specifications

1. App keeps all strings in the `strings.xml` file.
2. The app enables RTL layout switching on all layouts.
3. The app includes support for accessibility. That includes content descriptions, navigation using a D-pad, and, if applicable, non-audio versions of audio cues.
4. The app uses an AsyncTask to perform short duration on-demand requests

User Interface Mocks

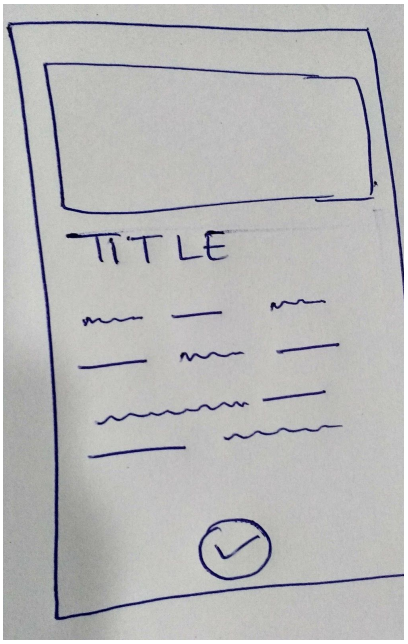
These can be created by hand (take a photo of your drawings and insert them in this flow), or using a program like Google Drawings, www.ninjamock.com, Paper by 53, Photoshop or Balsamiq.

Screen 1



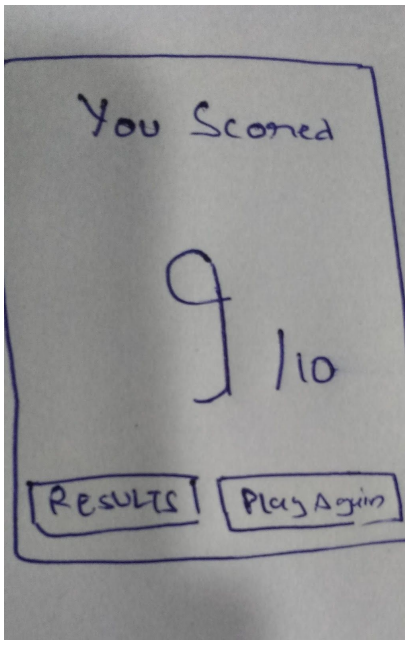
This will be the first screen for the app. The launch screen from where user can start a new game, check the global score and rank.

Screen 2



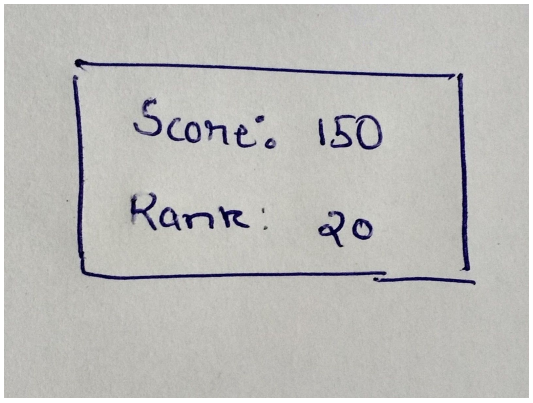
This will the game screen. In this random wikipedia article will be pulled and displayed with the missing words.

Screen 3



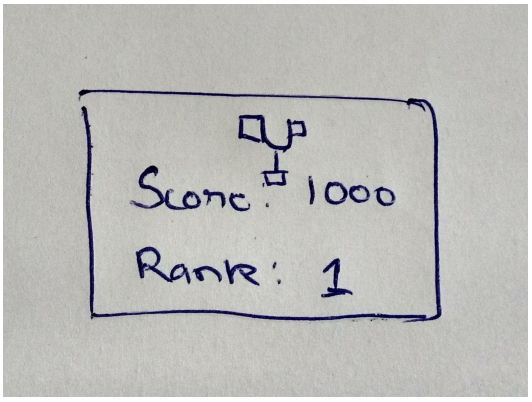
This screen shows the score of the user.

Screen 4



Widget shown to the user displaying the score and global rank.

Screen 5



A trophy of golden, silver or bronze is shown is the user's rank is first, second or third respectively.

Key Considerations

How will your app handle data persistence?

App uses local shared preferences to save and retain game's state along with firebase Realtime database to maintain the world rank.

Describe any edge or corner cases in the UX.

The hard version provides a time limit to the user to finish the game. In that time limit if the user puts the app in background, the time keeps on running and the game terminates if the time expires.

Describe the working environment.

The project uses **Android Studio:3.2.1** as IDE and gradle 4 with **gradle 4.6** and **Android plugin 3.6.1**.

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

Retrofit:2.4.0: For network calls
Leak Canary:1.6.1: For determining leaks in the codebase.
Gson:2.8.5: For interconversion of json and POJOs
ViewModel:1.1.1: For maintaining Activity Lifecycle and handle config changes.

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

Firebase Realtime Database:16.0.4: For storing user's score and maintaining world rank.
FIrebase Authentication:16.0.5: For signing up users.

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

- Complete the UI as per the decided designs.
- Implement game/business logic.
    - Manage the persistent storage to save user's score.
- Hook up Firebase Authentication to sign up user.
- Implement Firebase Realtime Database for maintaining score at the Server.
    - Replace preferences call with Firebase Realtime database calls.
- Develop a widget as per designs to show user's score and global rank.

Task 2: Implement UI for Each Activity and Fragment

- Build UI for Login Sign up screen.
- Build Home Fragment
- Build Game Fragment
- Build Score Fragment

Task 3: Implement Business/Game Logic

- Add logic to implement the game functionality
    - Fetch Data from Wikipedia
    - Remove random words from the data
    - Display the data to the user with missing blanks
    - Manage user's interaction to fill the missing words

Task 4: Implement Firebase Authentication

- Make a project on Firebase console.
- Connect the project with app
- Setup Firebase Authentication in project
- Hookup Firebase Auth with Login Fragment

Task 5: Implement Firebase Realtime Database

- Enable Realtime database on Firebase console
- Add Firebase Realtime database dependency to the android app.
- Hookup the Realtime database calls in place of share preferences calls.

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**"