GitHub Username: blackbird973

Jokesters Kingdom

Description

The aim of my app is to create a community between funny people and jokesters.
Each people will have the ability to share their own jokes to the community which i called the “Kingdom” and all these jokes will be displayed together in the “Jokes Feed” of the app which will be common to all the users.

Each user will have to register to be a member of the “Kingdom”, they could choose a username (and maybe a profile picture)

My app don’t solve a particular problem, it just allow users to make their day a little happier while reading funny jokes.

App will be written solely in the Java Programming Language
App will keeps all strings in the strings.xml file and enables RTL layout switching on all layouts.
The app will include contents description, and the navigation could be done using a D-pad.
The app will use an IntentService to fetch data for the widget.

The libraries i will use are:
firebase-core : 16.0.0
firebase-auth : 16.0.2
firebase-database: 16.0.1

Intended User

This is an app for jokesters and funny people who like spreading some love around themselves

Features

Main features of the app:

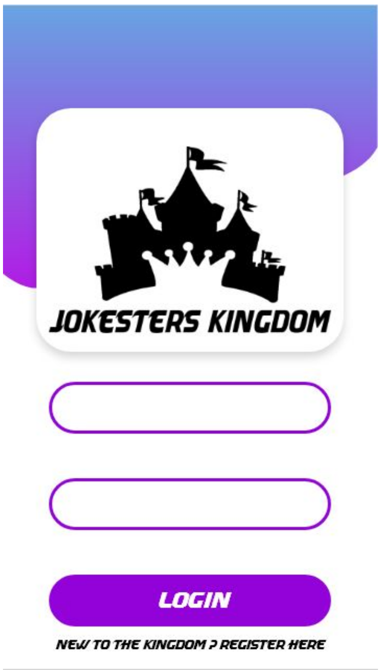
- Register/login User
- Store jokes data and retrieve them in firebase
- Store user data and display it in the User profile Activity
- Give a "like" on jokes you like

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.

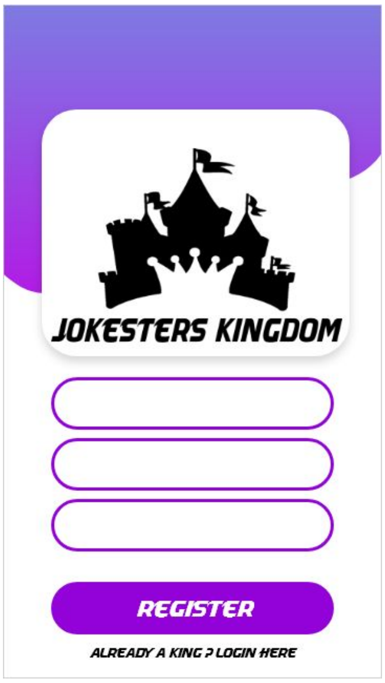
I made these mockups using AdobeXD

Screen 1



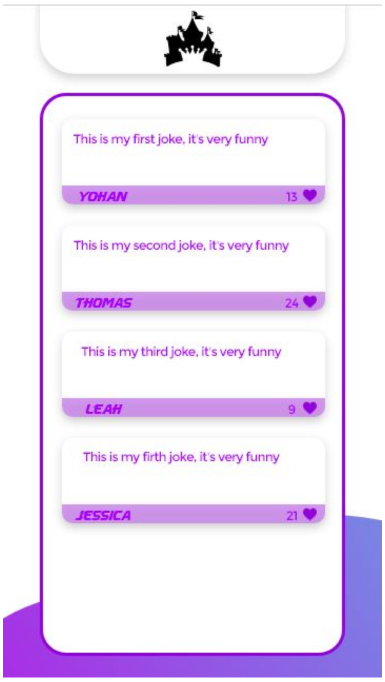
The first screen is the login screen, if the user is not registered yet, he can go to the second screen, the register screen, by touching the textview below the login button.

Screen 2



The second screen is the activity to register the user, in the first TextInput he enter his name, then his e-mail, then his password and once he click the register button he will be bring to the main screen of the app, the “jokes feed”

Screen 3



This is my first joke, it's very funny

YOHAN 13 ♥

This is my second joke, it's very funny

THOMAS 24 ♥

This is my third joke, it's very funny

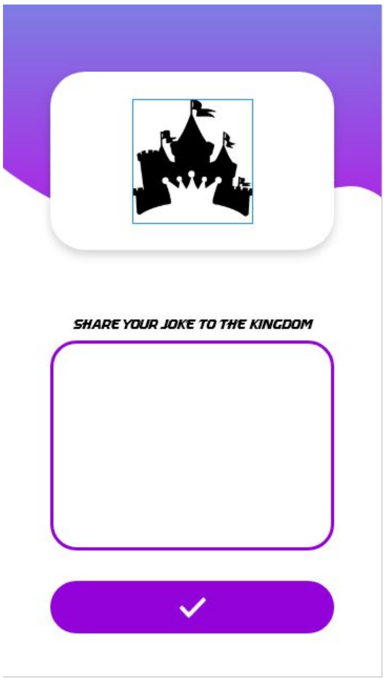
LEAH 9 ♥

This is my firth joke, it's very funny

JESSICA 21 ♥

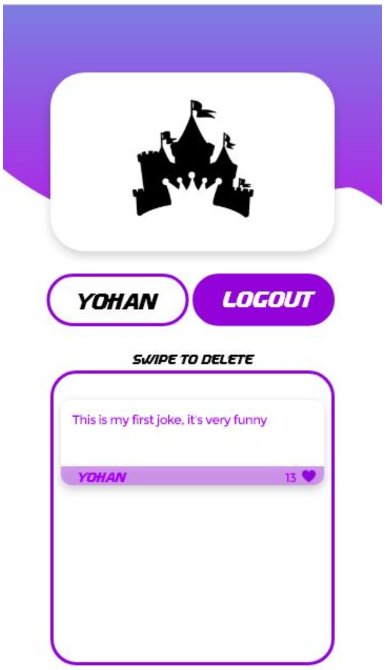
The third screen is the jokes feed, here will be displayed all the jokes that each user will upload to the database, on each joke, the name of the user who uploaded this joke will be displayed, and also the number of likes the joke has, if i managed to code this feature, i would like that each user could like each other jokes by double tapping on it

Screen 4



The fourth screen is the activity to upload a joke to the “Kingdom”, the user will write his joke in the InputText Layout and when he will click on the button, the joke will be uploaded to the database and shown in the “jokes feed”

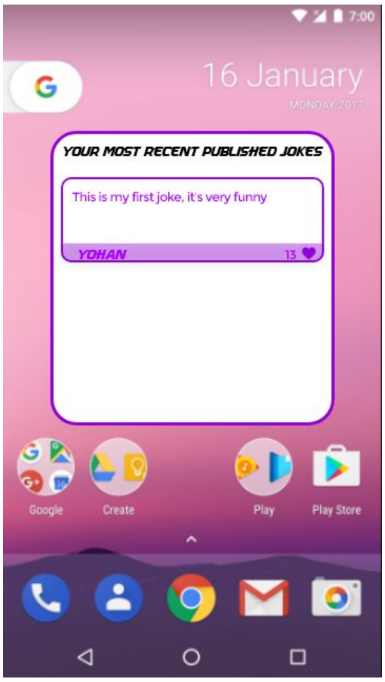
Screen 5



This is my first joke, it's very funny

The fifth screen is the “Profile” activity, here the user will see his name, and all the jokes that he uploaded to the database, if i manage to implement this feature, i would like that the user could delete the joke he want by swiping it. And also, the user will have the possibility to logout of his account by touching the “logout” button.

Screen 6



This is my first joke, it's very funny

The sixth screen is the widget, the widget will allow the user to see the most recent jokes he uploaded to the community.

.

Key Considerations

How will your app handle data persistence?

I will use Firebase Realtime Database

Describe any edge or corner cases in the UX.

Users will be notified by a toast, or a snackbar if their login information are wrong.
Users will be notified by a toast or a snackbar if the registration was sucessfull or not.
Users will navigate through the 3 screens, (Jokes Feed, Profile and Upload) with a bottom navigation bar.

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

Firebase to authenticate the user and save his data in the cloud

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

I will use two services from Firebase : Firebase Auth to Login/Register the user, and, Firebase Realtime Database to store and retrieve the user's data from it.

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

Write out the steps you will take to setup and/or configure this project. See previous implementation guides for an example.

- Configure libraries and dependencies
- Sync your Android app with your firebase project

Task 2: Implement UI for Each Activity and Fragment

- Build UI for the Login Activity
- Build UI for the Register Activity
- Build UI for the “Profile” Fragment
- Build UI for the “Upload Joke” Fragment
- Build UI for the “Jokes Feed” Fragment

Task 3: Implement Firebase Auth

- Implement the firebase auth so that we can register and login a user
- Write the method to log out the user instance when we click the “logout” button of the “profile” fragment

Task 4: Display Toast for each case

- Display toast when the user is logged in or when the login information are wrong

Task 5: Assign Fragments

- Assign each fragment with each icons of the bottom navigation bar

Task 6: Upload data and retrieve it

- Upload name data to firebase
- Upload jokes to firebase when the user upload a joke
- Retrieve the user’s name from firebase to the “profile” Fragment
- Retrieve all the jokes from firebase to the “Jokes Feed” Fragment in a recyclerView
- Delete the corresponding joke from firebase when the user swipe it
- Add the “like” feature

Task 7: The Widget

- Create the Widget layout
- Create the Widget Activity
- Retrieve the recent user’s jokes from the database and display them in the widget

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**”