GitHub Username: BFadairo

Marvelpedia

Description

Welcome to Marvelpedia! This app provides a convenient location for obtaining data of heroes, stories, events, and comics within the Marvel Universe. Users are able to create their own superhero teams with characters from the Marvel Universe!

Intended User

Lovers of the Marvel Universe!

Features

1. App will allow users to search for their favorite Marvel Super Heroes including
    a. Character’s biography

   b. Major events the character is featured in
   c. # of Comics and the Comics character has appeared in
   d. Story Volumes
2. App will allow users to create their own superhero “team” and share it with their friends
   a. Users will be able to view which one of their team members appeared in the same comics, stories, & events

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.

Link to NinjaMock

Key Considerations

How will your app handle data persistence?

App will utilize Firebase database in order to allow users to save their hero teams

Describe any edge or corner cases in the UX.

User will return to the Main Screen on the press of the back button if that was the most recent screen, otherwise they will be redirected to the previous item they viewed

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

GSON for Json Parsing
Retrofit for Network Requests
Firebase Database to allow user to save heroes to their team
CircleImageView for Images next to labels on the main activity
Firebase Realtime Database for team saving
Google Play AdMob to display ads

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

App will periodically display ads using Google Play Services: AdMob throughout activities

App will use Firebase Realtime Database to save user hero teams locally and online

Next Steps: Required Tasks

Task 1: Project Setup

Tasks

- Set up Libraries (UI,Utility, Espresso)
    - Confirm libraries are probably loaded
- Verify API Key is valid (Obtain API key from same link)
    - Simple as testing calls from Marvel portal https://developer.marvel.com/docs

Task 2: Implement UI for Each Activity and Fragment

Tasks

- Main Activity
    - Set-up UI for Main Activity
        - Container for Fragment
    - Set-up Marvel API calls
        - Will be located within Utilities folder
    - Main Activity will hold one fragment
        - Fragment
            - Fragment will utilize API calls from Api Utilities
            - Will hold Horizontal RecyclerViews
- Detail Activity
    - Detail Activity will hold 4 fragments
        - Character
        - Comics
        - Stories
        - Events
            - Fragments will be replaced depending on what Item is being viewed (Comic, Story, Event, or Character)
- Settings Activity
    - Settings Activity will hold user search preferences
        - Options will be from any of the above entries (Character, Comics, Stories, & Events
- Database (Need Firebase Dependency for this)
    - Confirm depency is in the project
    - Implement the the Database to be used to store hero information for users
    - Database CRUD

        - Database will be created on launch of the application
        - Database will be queried when a user has heroes on their team and is within their Team Screen (Accessed via Navigation drawer)
        - Database will be updated when a user changes their team loadout.
        - Database deletion will commence when user deletes their team
- Navigation Drawer
    - Navigation Drawer will give user access to their team screen
- General Important Tasks
    - Provide attribution to Marvel everywhere a data call to their API is active

Task 3: Testing

UI Testing

- Espresso
    - Testing UI elements click listeners to confirm proper activity/action is launched/taken
    - Checking if proper data is passed through intents
- Database
    - Database must query and update entries within the database correctly to be displayed to the user

---

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**"