GitHub Username: counteer

Marvel Database

Description

It is a small application, which connects to Marvel API. You can search heroes, villains and read their backstories. And you can see which comics they appeared last.
The app is written solely in Java programming language.

Intended User

Intended users are comic fans, who wants to know more about Marvel stories and characters.

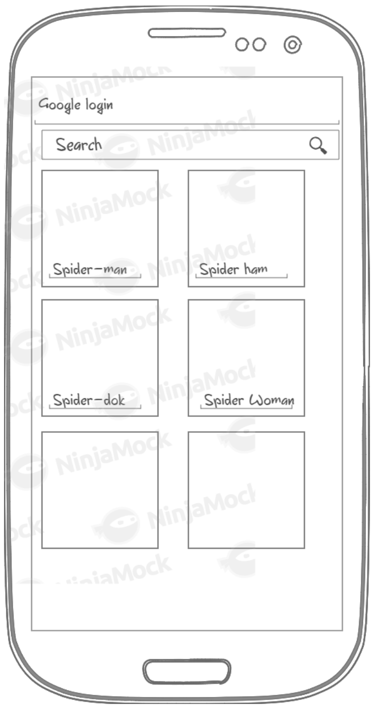
Features

- It connects to Marvel database and lists the heroes and villains
- It shows the latest comic books of characters

- It can show your favorite heroes or villains in your home screen
- Every text will be in the strings.xml for international support.
- The images are loaded mostly from Marvel Api

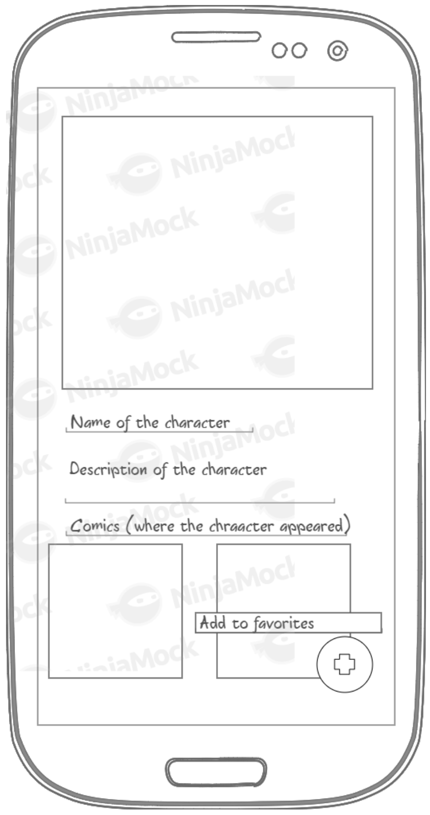
User Interface Mocks

Screen 1



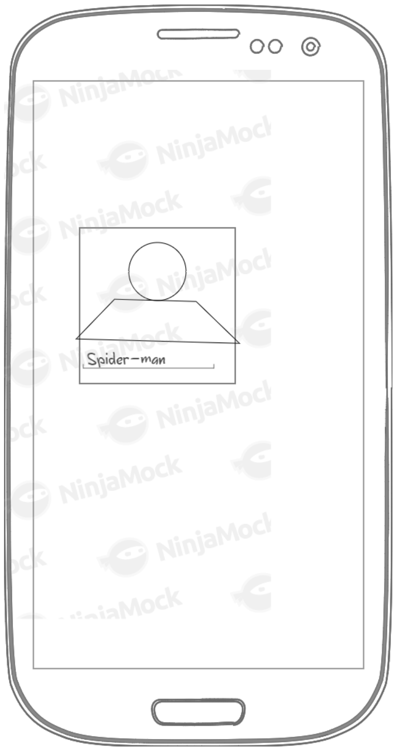
This is the first/main screen. The Google login “widget” is on the top. It shows your name if you are signed in, or it shows a login button. Under that, there is the search field, where you can write a text, and it shows the result with pictures. The pictures are clickable, and if you click them, you get to the Screen 2. The list of the characters are loaded by an AsyncTaskLoader

Screen 2


Name of the character
Description of the character
Comics (where the chraacter appeared)
Add to favorites

The second screen is the character's screen. Here is the picture of the character, under that, the name of the character and the description of the character. On the bottom of the screen, there is a list, with the latest comics, in which the character appeared.
There is a floating action button, where you can add this character to the favorites. The favorite characters are shown in the home screen widget.

Widget screen



The widget screen contains a picture of a comic book character, which was added to favorites. The widget is updated by an IntentService.

Key Considerations

How will your app handle data persistence?

It stores minimal information about characters. The application has a database, and a content provider reads and writes to it.

Describe any edge or corner cases in the UX.

The app loads data mainly from the Marvel Api. If something wrong happens (empty list, or error when downloading) then a message will appear instead of the lists.

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

Picasso to handle the loading and caching of images.
ButterKnife to handle UI elements.
Gson to convert data from the Api to objects.

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

I will use **Google Identity**. The user can login, and then it shows the user's name.
I will use **Google Analytics** to see which hero or villain is the most popular amongst 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

- Configure libraries
  - Android Studio 3.1.3
  - Gradle 4.4
  - Picasso 2.5.2
  - Butterknife 8.8.1
  - Google Play services 15.0.1
- Play with Google play services. How they work.
- Implementing RTL support

Task 2: Implement UI for Each Activity and Fragment

- Build UI for MainActivity
  - Implement Adapter for characters
  - Implement AsyncTaskLoader
- Build UI for CharacterActivity
  - Implement add to favorites button
  - Implement Adapter for comics
  - Implement AsyncTaskLoader to load latest comics
- Build UI for Home screen widget

Task 3: Your Next Task

Implement Google Play Services
- Analytics
- Identity

Build Widget
- Implement IntentService for refreshing Widget