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. Create your own team with heroes throughout the Marvel Universe! (5 Heroes per Team)

Intended User

Lovers of the Marvel Universe!

Anyone interested in gaining more information on content within the Marvel Universe

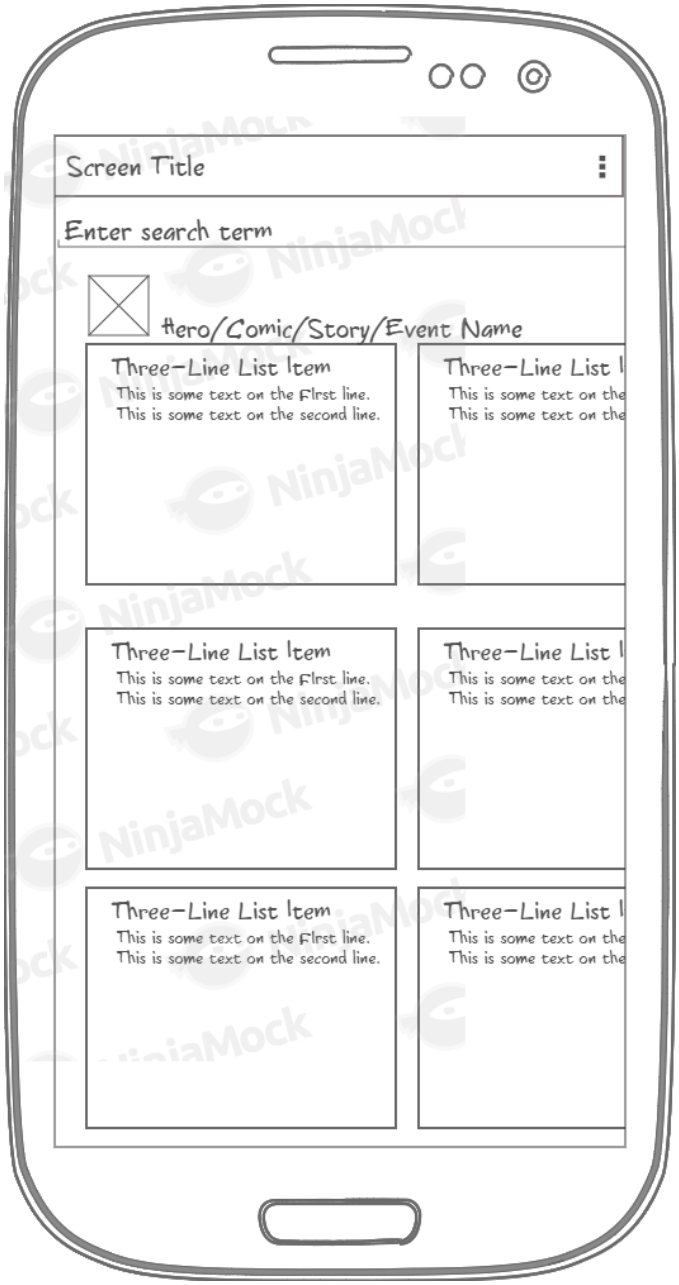
Features

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

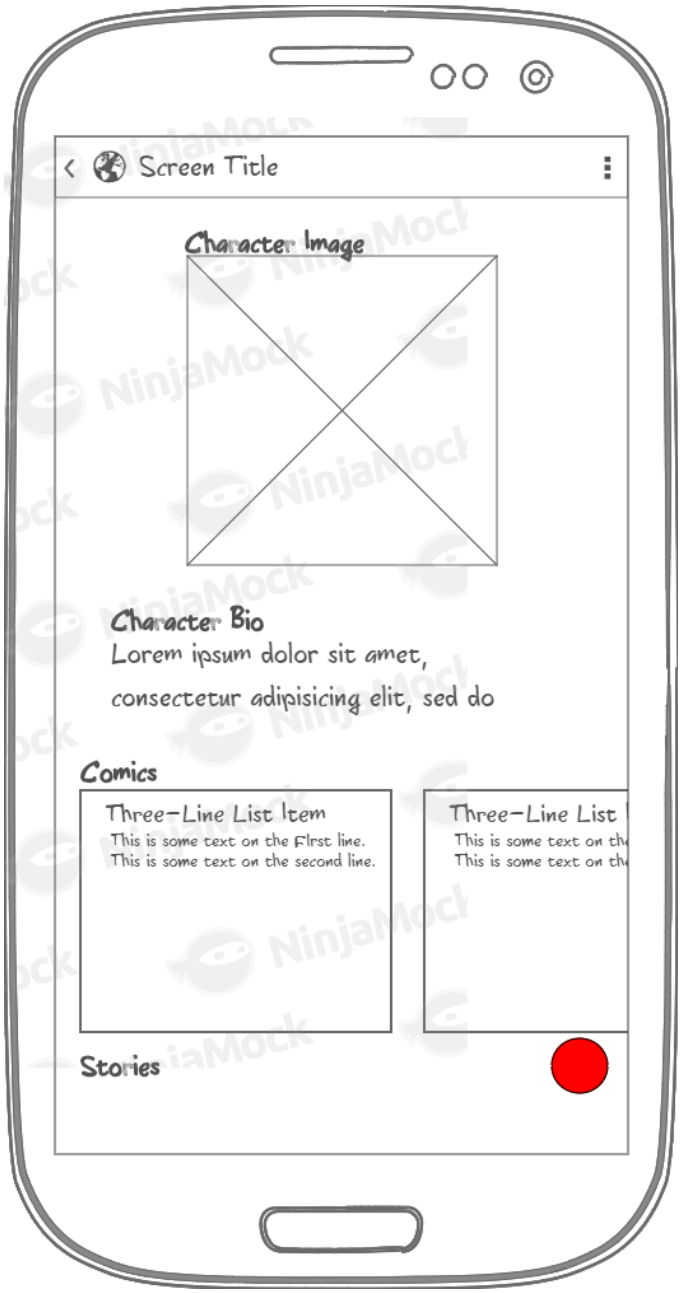
Screen 1:



Screen 1 Description:

The MainActivity screen will be used to display initial search results (from search term) to the user depending on the category of item they have requested to search for. Images for heroes (if available) will be displayed next to the name of the hero. Activity will use Horizontal Recycler views.

Screen 2:

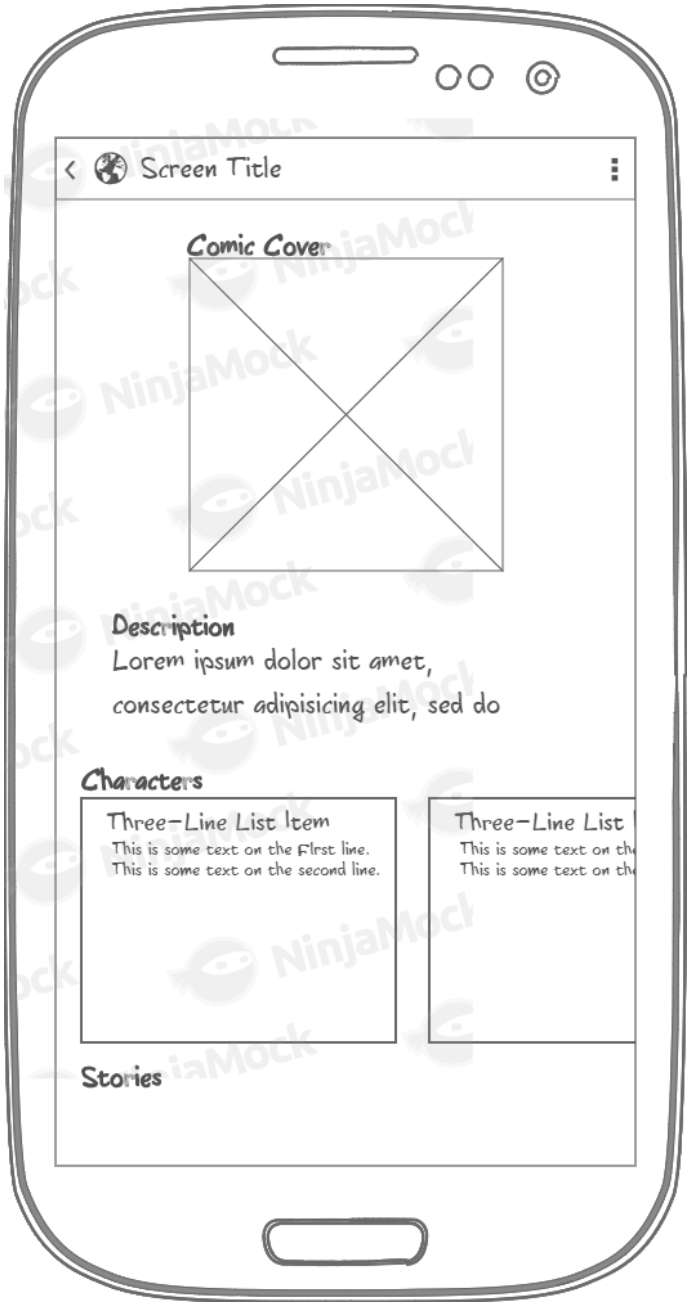


Screen 2 Description:

Character Detail will display a bigger image of the character (if available) as well as a short bio of the character. Below the description, different comics, stories, & events the character has appeared in will be displayed. Floating Action Button will be located at the bottom right to allow user to add character to their team. (FAB will not be plain red in actual app)

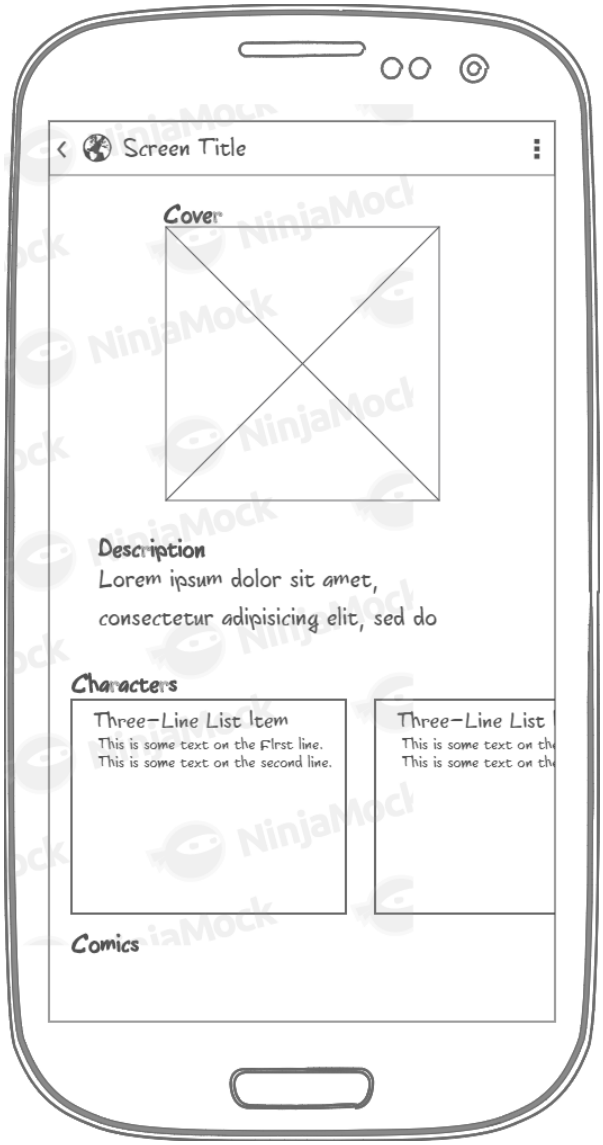
Screen 2 Variants:

Comic Detail:



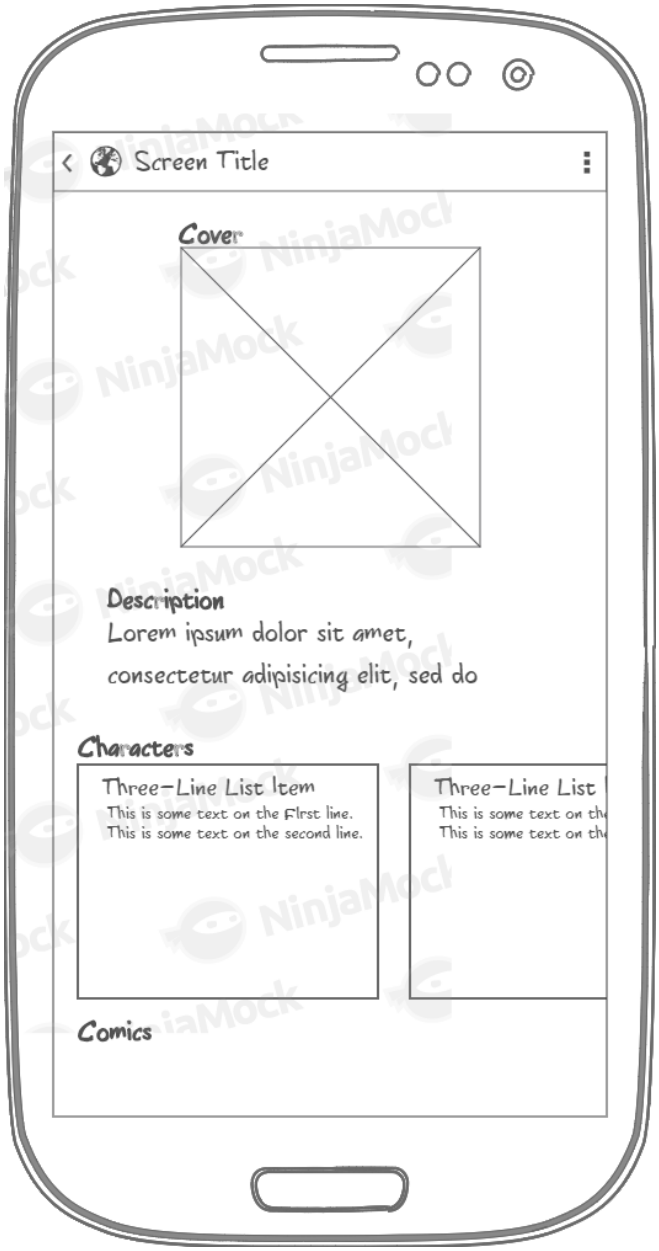
Comic Detail will also display characters that appear in the comics. Stories & Events associated with the comic will be displayed below

Event Detail:



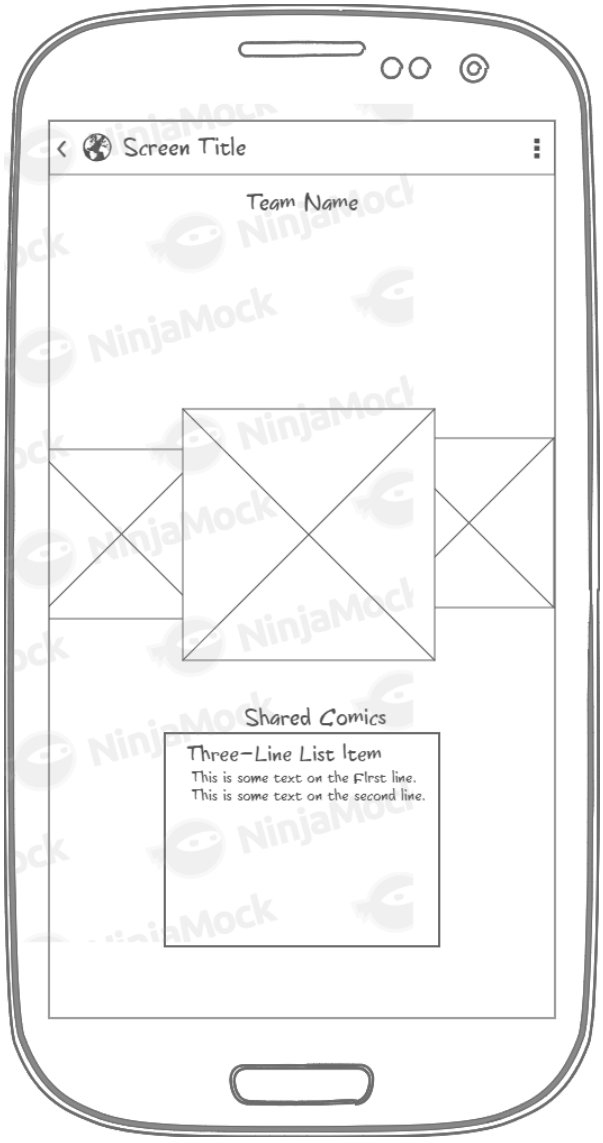
Event detail will display a description of the event, an image (if available), heroes that took part in the event, comics containing the event, as well as stories the event is within.

Story Detail:



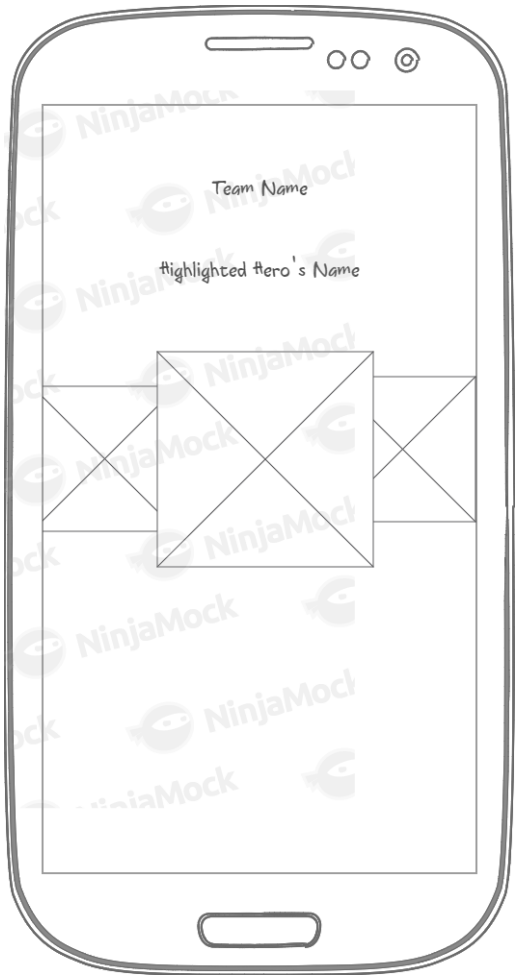
Will display relevant information about a particular Marvel Story such as heroes involved, comics contained in the story, and events that take place within it

Team Screen:



This is where heroes that have been added to the user's team will be displayed. Users will be able to edit the name of their teams at anytime as well as add and delete members from their teams. Users will also be able to search for what comics their team members appear in as a collective (Will probably add checkboxes to deselect/select heroes for the search)

Widget:



Widget will display information from the TeamScreen of the app. Current Team Members will be displayed with labels displaying the team name and the Hero's name.

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

Tapping a character on the widget will bring users to the detail screen for that character

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

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

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
        - API Calls
            - There will be fragments for each horizontal recyclerView within the Detail Activities.
            - 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 first addition to hero team.
        - 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

Extra Details:

- Application will implement Application Compatibility Libraries to ensure compatibility across API levels
- Application will be written using the Java programming language
- String values will be kept in their associated String.xml file (English only currently)
- Application will implement a IntentService for API calls
    - Initially users search for comic, hero, story, or event
        - Items will only start loading once a search term is entered and confirmed
        - Detail Activity
            - When item is clicked from MainActivity, IntentService will begin loading items for Detail
- Libraries used will be the most updated stable release version
- Accessibility
    - Content Descriptions will be provided