The Basics

GitHub Username: https://github.com/fullmers

App name: NASA View

Description

NASA View allows you to explore the treasure trove of public NASA images. Admire photos from the Apollo mission to galaxy images from the Hubble telescope and everything in between! Get inspired by the amazing collection of images that NASA has made publically available.

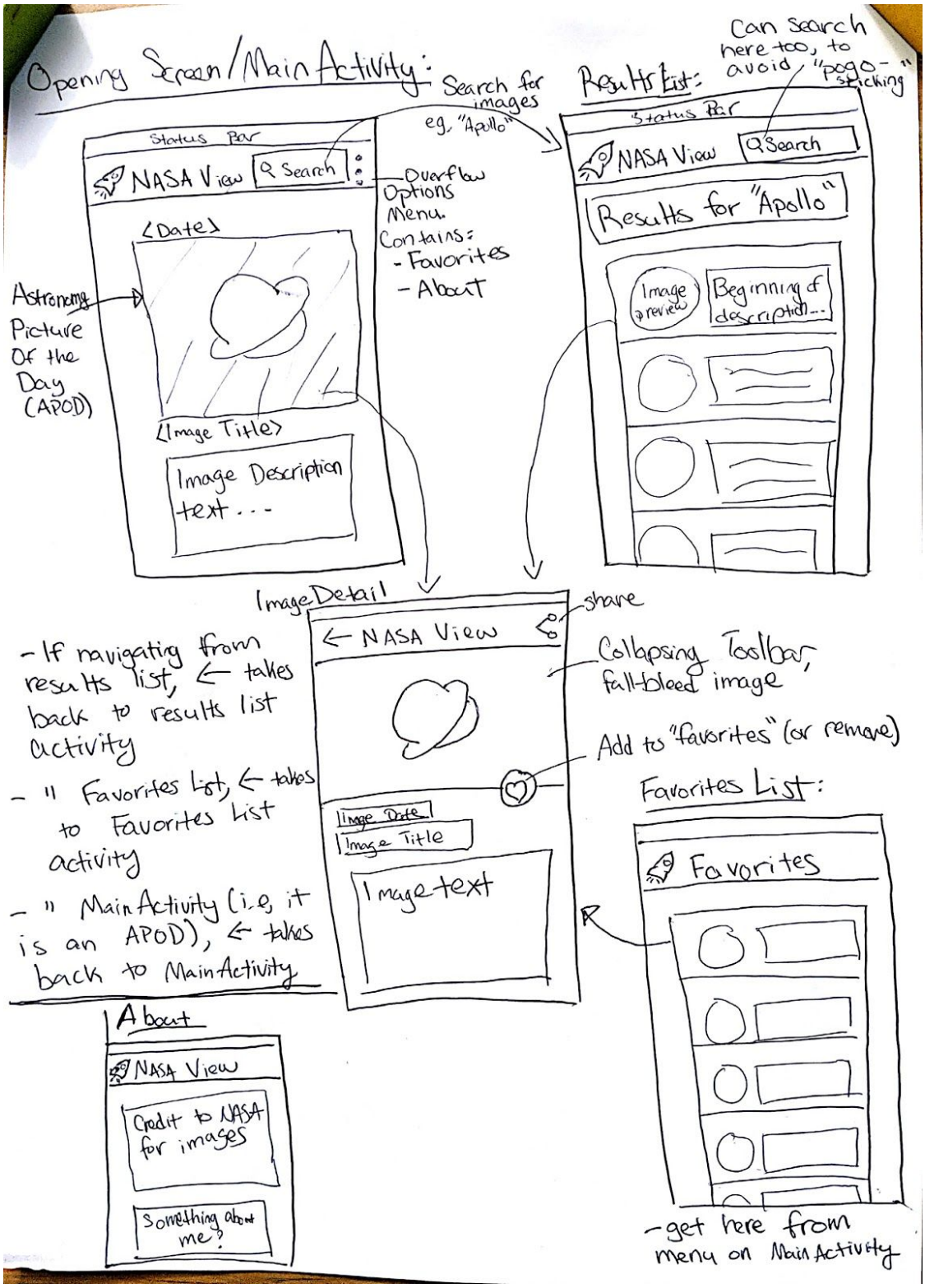
Intended User

NASA View is for anyone who is interested in NASA or space exploration.

Features

- Search the NASA image archive by keyword and learn some details about each image
- Be greeted by the Astronomy Picture of the Day when opening the app
- Share images

User Interface Mocks



I tried NinjaMock, but found it very limited. So, please see image above for wireframes for each screen, along with relevant descriptive information.

Key Considerations

How will your app handle data persistence?

User favorites data will be stored using the Room persistence library.
https://developer.android.com/topic/libraries/architecture/room

Describe any edge or corner cases in the UX.

-If there are no results for given search term, a “no results, try again” screen will be shown.
-If internet connection is lost, snackbar will appear anytime the user tries to do something that requires internet.
-navigation edge cases covered in wireframe
-list of favorites stored locally, but images not. So if the user tries to see their favorites list with no internet, they will still not be able to.

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

Picasso - for image handling and caching.
Butterknife - nicer data binding
Mockito - for testing

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

To search for images, the user will be accessing the NASA images api (https://api.nasa.gov/api.html#Images) I will only enable image results (not video or audio)

The MainActivity will display the Astronomy Picture Of the Day image (https://api.nasa.gov/api.html#apod)

If I must also use a Google Play Services (do I really? It seems like pointless make-work, since they are not really needed), then I will try out Analytics.

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

- Create an Android Studio project.
    - Start with an “Empty Activity”
    - Min API 21 (sorry, not sorry)
- Git init, special Android .gitignore
- To gradle, add dependencies for
    - Butterknife
    - Mockito
    - Support design
    - Support RecyclerView
    - Support AppCompat

Task 2: Implement UI and navigation for each Activity/Fragment

- Create Activity and layout for MainActivity
    - Show APOD image with meta data
    - Options menu with link to FavoritesListActivity and AboutActivity
- Create ResultsListActivity and layout
    - Toolbar has search action
    - Make ListItem xml
    - Make an Adapter and ViewHolder to show search results
- Create FavoritesListActivity and layout
    - Toolbar does not have search action
    - Reuse Adapter, ViewHolder and ListItem xml from above
- Create ImageDetailActivity and layout
    - Set up-button logic so it navigates back to correct activity:
        - MainActivity (for APOD)
        - FavoritesListActivity
        - ResultsListActivity
- Create ImageDetailFragment and layout
    - Uses ViewPager, so user can side swipe between results or favorites.
    - CoordinatorLayout with CollapsingToolbar, FAB
- Create AboutActivity and layout
- While creating above activities and layouts, create instrumentation tests to make sure navigation works correctly.

Task 3: Connect to NASA APIs

- Put API Key in file that is not committed to git
- Define data model
- Connect to APOD api
    - Validate data

    - Display in MainActivity
    - Use IntentService or AsyncTask
- Connect to NASA Images API for search results
    - Validate data
    - Display in Search Results Activity
    - Use IntentService or AsyncTask
- Create appropriate tests

Task 4: Implement Data Caching

- Set up Room to store favorites
- Connect Create/Delete favorite logic to button
- Display favorites in FavoritesListActivity
- Create appropriate tests

Task 5: Implement Google Play Services

- Learn about Google Analytics
- Implement Analytics
    - Only create one instance, per rubric and best practices
- Learn about Google TestLab
- Implement Google TestLab
    - Use Google TestLab to run tests

Task 6: Handle Error cases

- Handle no internet case
- Handle no search results found case
- Make appropriate tests

Task 7: Make it shine

- Shared element transitions between list and detail views
- Animation behavior on favorite button
- Fade image when swiping between search results
- Let user share image on social media
- Find or make rocket icon, put in toolbar and as launch icon

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**"