Description
Intended User
Features
User Interface Mocks
Main Screen
Detail Screen
Repository and Contributor Item
Widget
Key Considerations
How will your app handle data persistence?
Describe any corner cases in the UX.
Describe any libraries you'll be using and share your reasoning for including them.
Describe how you will implement Google Play Services.
Next Steps: Required Tasks

GitHub Username: davidllorca

Trending Repos

Description

The main purpose of Trending Repos is browsing the most famous source of repositories platform. You can search the most popular projects by name or programing language and add projects to favorites list also.

Intended User

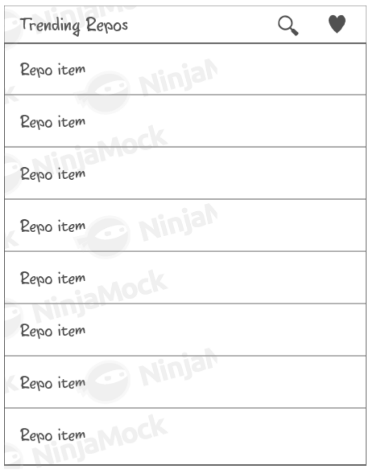
Who wants a quick access to Github and its popular repositories.

Features

- View trending repos
- Search repos by name or programming language.
- Bookmark repos.

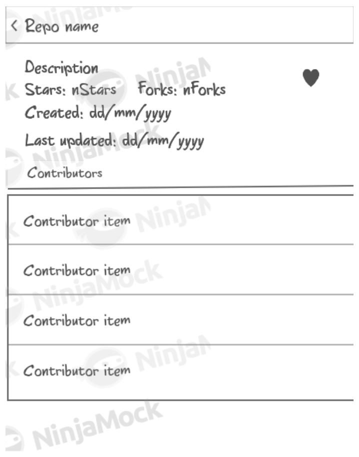
User Interface Mocks

Main Screen



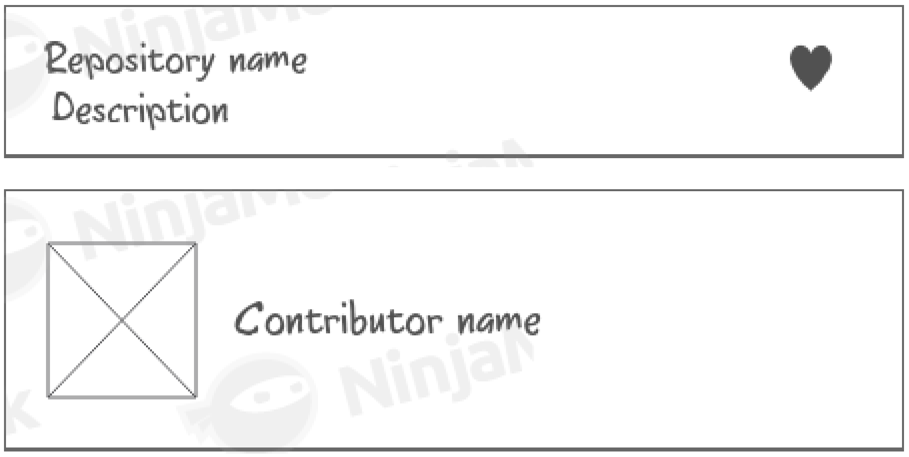
- List trending repositories.
- Search by specific term.
- List favourites repositories.

Detail Screen

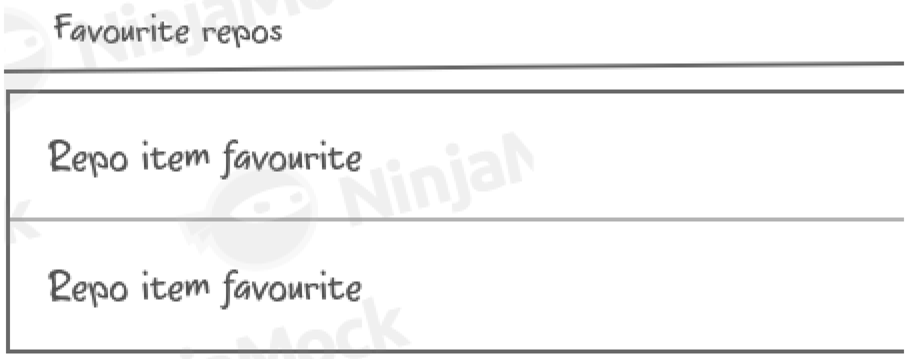


- Show details of target repository.
- Add to favourite list.

Repository and Contributor Item



Widget



Widget shows the favourite list of user.

Key Considerations

Common Project requirements

Application will be written solely in the Java Programming Language.

How will your app handle data persistence?

App use Room library to handle data persistence. There are two use cases that app persists some data:

- Saving favorites repositories.
- Saving the initial main query results to avoid blank screen when user launches the app.

Describe any edge or corner cases in the UX.

- If there is no internet connection available user will be notified with a message.
- To avoid large call's responses there will be a cache that saves the searches along the user session.

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

Picasso to handle the loading and caching of images.
Architecture Components Room to persist local data.
Architecture Components ViewModel to implement a MVVM architecture.
Retrofit 2 to make remote calls.
RxJava 2 to handle threading.
Android Data Binding to avoid boilerplate declaring the UI.
Dagger 2 to provide the dependencies across the app.

Release versions of all libraries, Gradle and Android Studio

Libraries:

Picasso: "com.squareup.picasso:picasso:2.71828"
Architecture Components Room:
"android.arch.persistence.room:runtime:1.1.1"
"android.arch.persistence.room:compiler:1.1.1"
"android.arch.persistence.room:rxjava2:1.1.1"
Architecture Components ViewModel: "android.arch.lifecycle:viewmodel:1.1.1"
Retrofit 2: "'com.squareup.retrofit2:retrofit:2.4.0"
RxJava 2:
"io.reactivex.rxjava2:rxandroid:2.0.2"
"io.reactivex.rxjava2:rxjava:2.0.2"
Dagger 2:
"com.google.dagger:dagger-android:2.x"

Gradle Plugin version 3.0.1

Android Studio version 3.0.1

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

Firebase Analytics to track which are the most searched terms.
Crashlytics to report app's crashes.

Accessibility

App will follow up all possible recommendations of Android guide about accessibility https://developer.android.com/guide/topics/ui/accessibility/apps. Like UI view labeled properly, making touch targets as large as possible, choosing a adequate color contrast,...

Theming and resources

- App's them will extend from AppCompat theme.
- All resources will be stored in /res folders properly.
- All strings needed will be retrieved from 'string.xml' located in /res folders.

Next Steps: Required Tasks

Task 1: Project Setup

- Configure all libraries used.

Task 2: Implement data layer

- Create DAO to manage local persistence.
- Create model for Repository an Contributor

Task 3: Create remote model response

- Create model for Github call's responses.

Task 4: Implement local API (Repository pattern)

- Create accessors to provide data.

Task 5: Implement remote calls to Github API

- Implement local API to connect with remote server.

Task 6: Implement UI screens

- Implement List layout.
- Implement Detail layout.

Task 7: Implement TrendingReposViewModel

- Create ViewModel with logic to list the repos and other search options.

Task 8: Implement DetailRepoViewModel

- Create ViewModel with logic to display detail information of target repository.

Task 9: Implement Save Repo feature locally

- Implement logic to save a favorite repo.

Task 10: Implement SearchRepoViewModel

- Create ViewModel with logic to list the repos by search options.

Task 11: Create proper Activities

- Implement MainListActvity and bind its viewmodels properly.
- Implement DetailActivity and bind its viewmodels properly.