
- Description
- Intended User
- Features
- User Interface Mocks
  - Screen 1
  - Screen 2
- 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
  - Task 1: Project Setup
  - Task 2: Implement UI for Each Activity and Fragment
  - Task 3: Your Next Task
  - Task 4: Your Next Task
  - Task 5: Your Next Task

GitHub Username: okaxel

SemiColon 

Description

SemiColon gives response for two major problems of software developers. It searches after fresh releases of libraries and software furthermore it provides inspiring and motivating quotes too.

Intended User

SemiColon is for software developers who uses third-party libraries and need some motivation.

Features

List of the main features SemiColon:

- **Widget**:
  - App has a home-screen widget to provide quotes and inform about freshness of favorite libraries and software

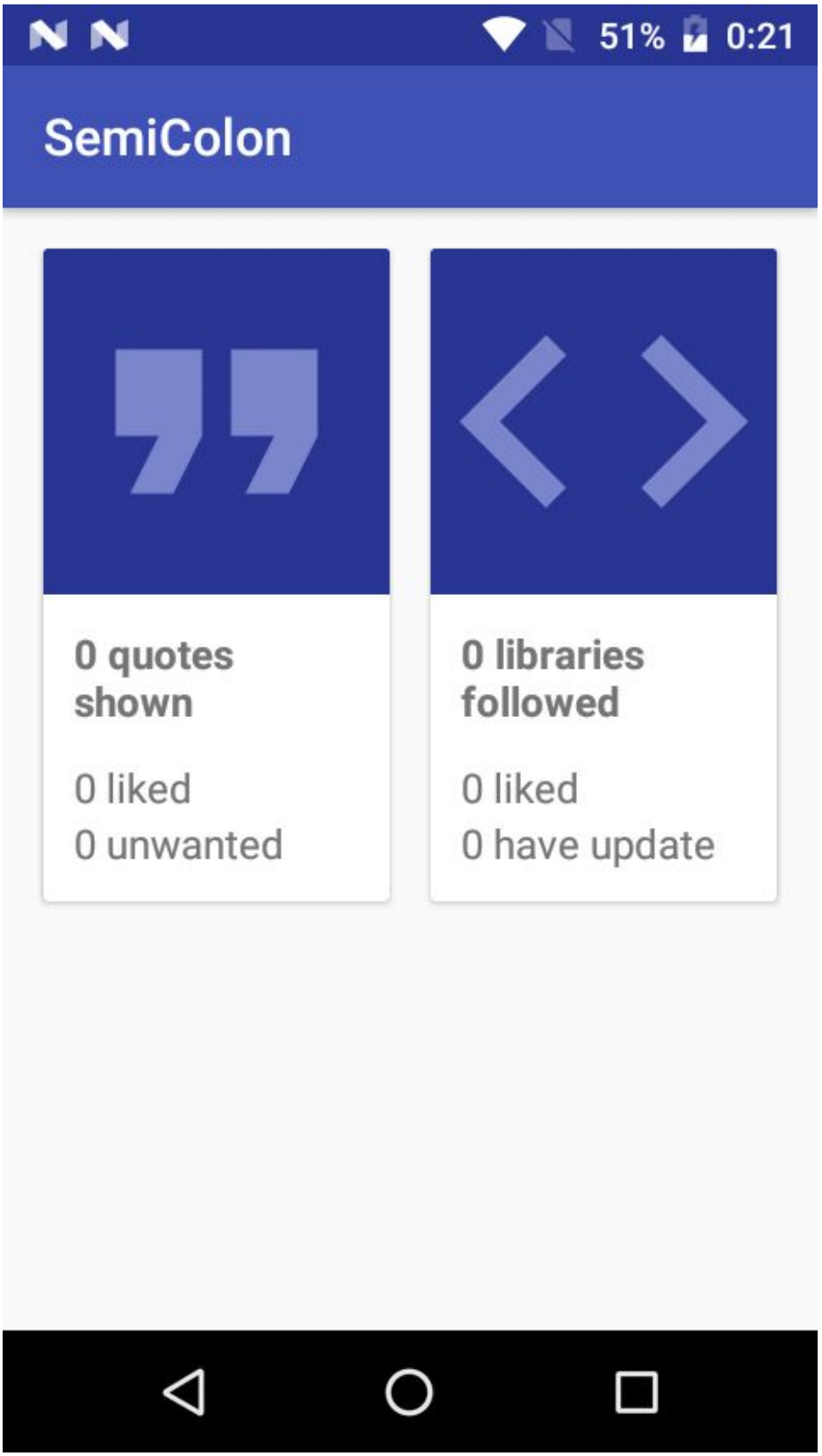
    - If user clicks on the quote additional functions are available via applications activity
    - Information about freshness of libraries and software is provided on the top side of the widget. If the user clicks this area additional functions are available via applications activity (libraries and software main screen)
- **Home-screen:**
    - App has a home-screen with two cards
    - One card shows details about provided, shared, hidden quotes
    - Other card shows details about libraries and software available to follow, number of favorites or shared libraries and software
- **Libraries and software:**
    - App has a screen for libraries and software list
    - From the list user can view library or software details (tap, click) or view separate favorites page (swipe left)
    - User can mark and unmark library or software as favorite (on details screen with a star)
    - The freshness of favorite libraries and software is checked by the app regularly
    - The used API has a default project list but user can add any github repositories to the list (fab on list screen)
    - User can share their favorite list (fab on favorites screen)
    - User can share the details of a library and software (fab on details screen)
    - On the details screen user can navigate to the homepage of the software, library or github repo (with tap, click) or copy the link (long tap)
- **Quotes:**
    - Quotes main screen contains the quote that appears on the widget
    - User can mark/unmark quote as favorite or unwanted
    - User can browse favorite quotes (swipe left) and unwanted quotes (swipe right) as well
    - User can share the actual quote or the list of favorite and unwanted quotes (with fab on the concerning tab of this activity)
- **Data:**
    - SemiColon uses ContentProvider and to store data locally and Loader to show it in views
    - Name of github repositories is stored which are added by the user
    - Favorite libraries, software and repositories are stored
    - Last version of favorite libraries, software and repositories are stored as well
    - Number of shown quotes is stored
    - Favorite and unwanted quotes are stored too
- **API handling:**
    - SemiColon uses Verse API to check freshness of libraries, software and github repositories (https://verse.pawelad.xyz/)
    - SemiColon uses Forismatic API to get quotes (http://forismatic.com/en/api/)
    - Both APIs are pulled regularly, Verse API daily, Forismatic API even more often, the processes are managed with JobDispatcher

    - App validates all input from servers and users. If data does not exist or is in the wrong format, the app logs this fact and does not crash
- **Development criteria:**
    - SemiColon conforms to common standards found in the Android Nanodegree General Project Guidelines
    - SemiColon is written solely in the Java Programming Language
    - SemiColon utilizes stable release versions of all libraries, Gradle, and Android Studio.
    - SemiColon integrates third-party libraries:
        - ButterKnife http://jakewharton.github.io/butterknife/
        - Schematic https://github.com/SimonVT/schematic
        - gson https://github.com/google/gson
        - Timber: https://github.com/JakeWharton/timber
    - SemiColon includes support for accessibility. That includes content descriptions and navigation.
    - SemiColon keeps all strings in a strings.xml file and enables RTL layout switching on all layouts
- **Integrated Google services**
    - Admob on MainActivity screens
    - Firebase Analytics
- **Design:**
    - SemiColon's theme extends AppCompat
    - SemiColon uses app bar and associated toolbars
    - SemiColon uses standard and simple transitions between activities
- **Legal stuff:**
    - SemiColon pay special respect to all legal needs of Google Services, APIs and third party libraries

User Interface Mocks

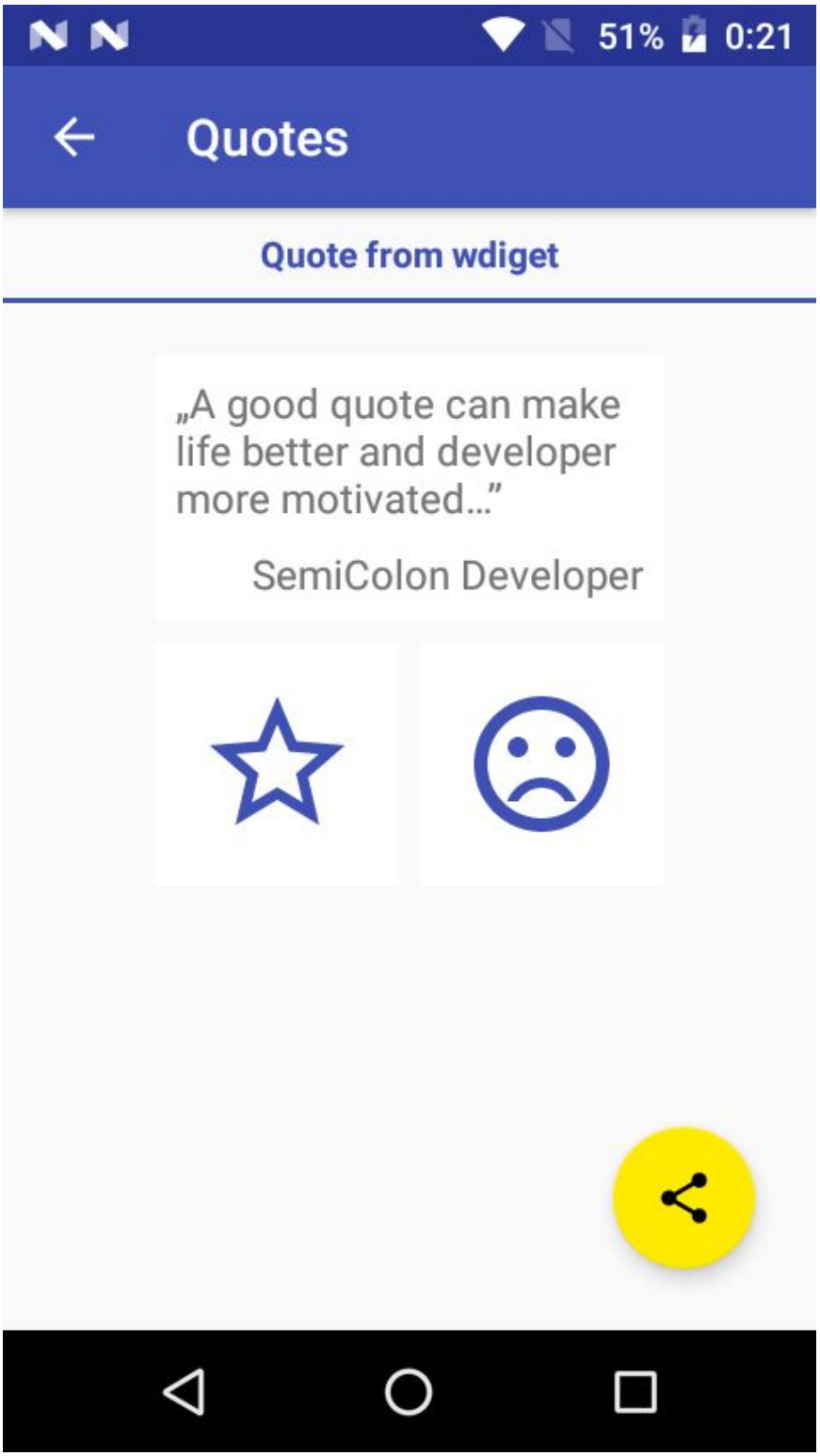
These can be created by hand (take a photo of your drawings and insert them in this flow), or using a program like Google Drawings, www.ninjamock.com, Paper by 53, Photoshop or Balsamiq.

MainActivity



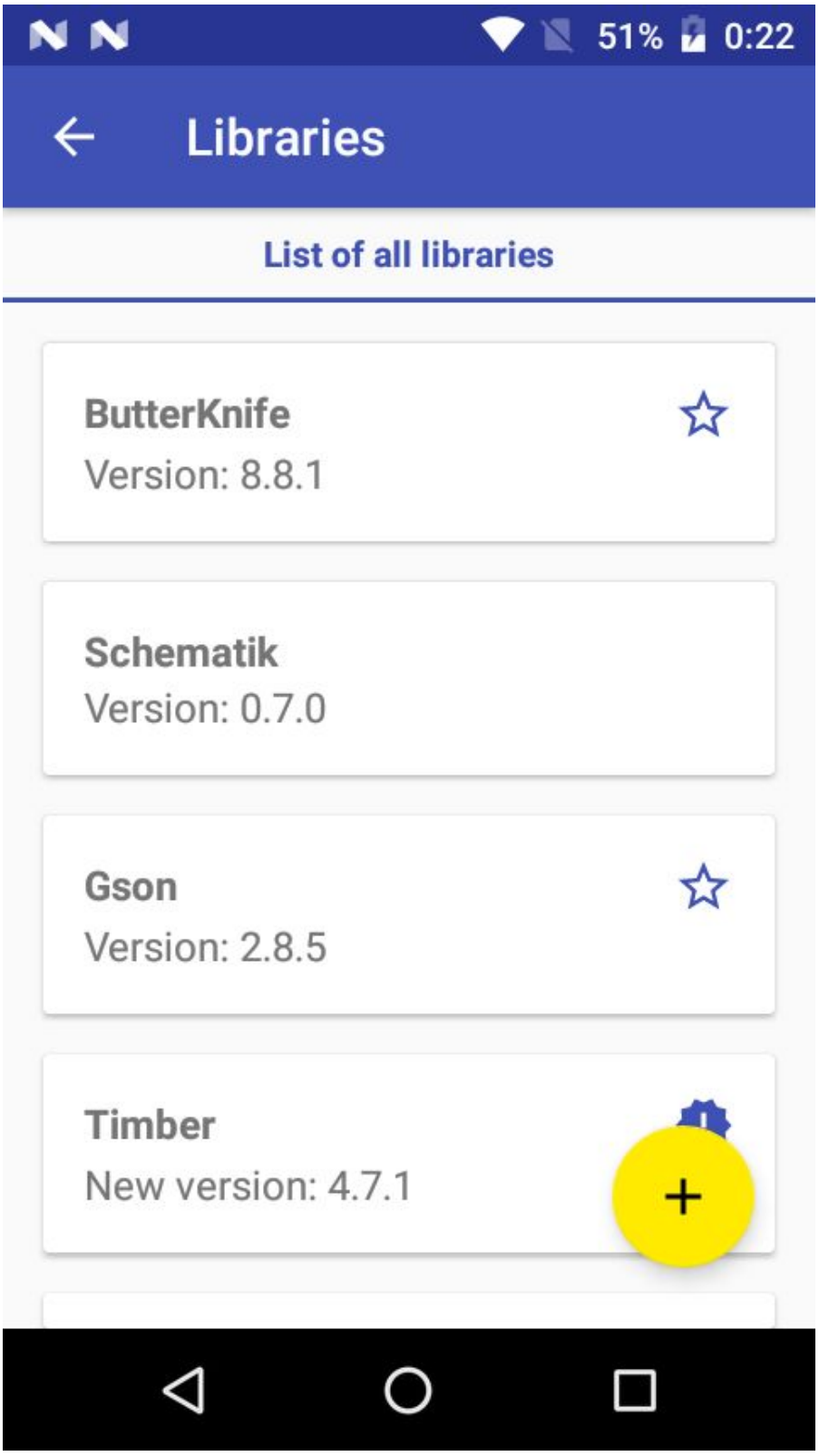
App has a home-screen with two cards. One card shows details about provided, shared, hidden quotes other card shows details about libraries and software available to follow, number of favorites or shared libraries and software.

QuotesActivity



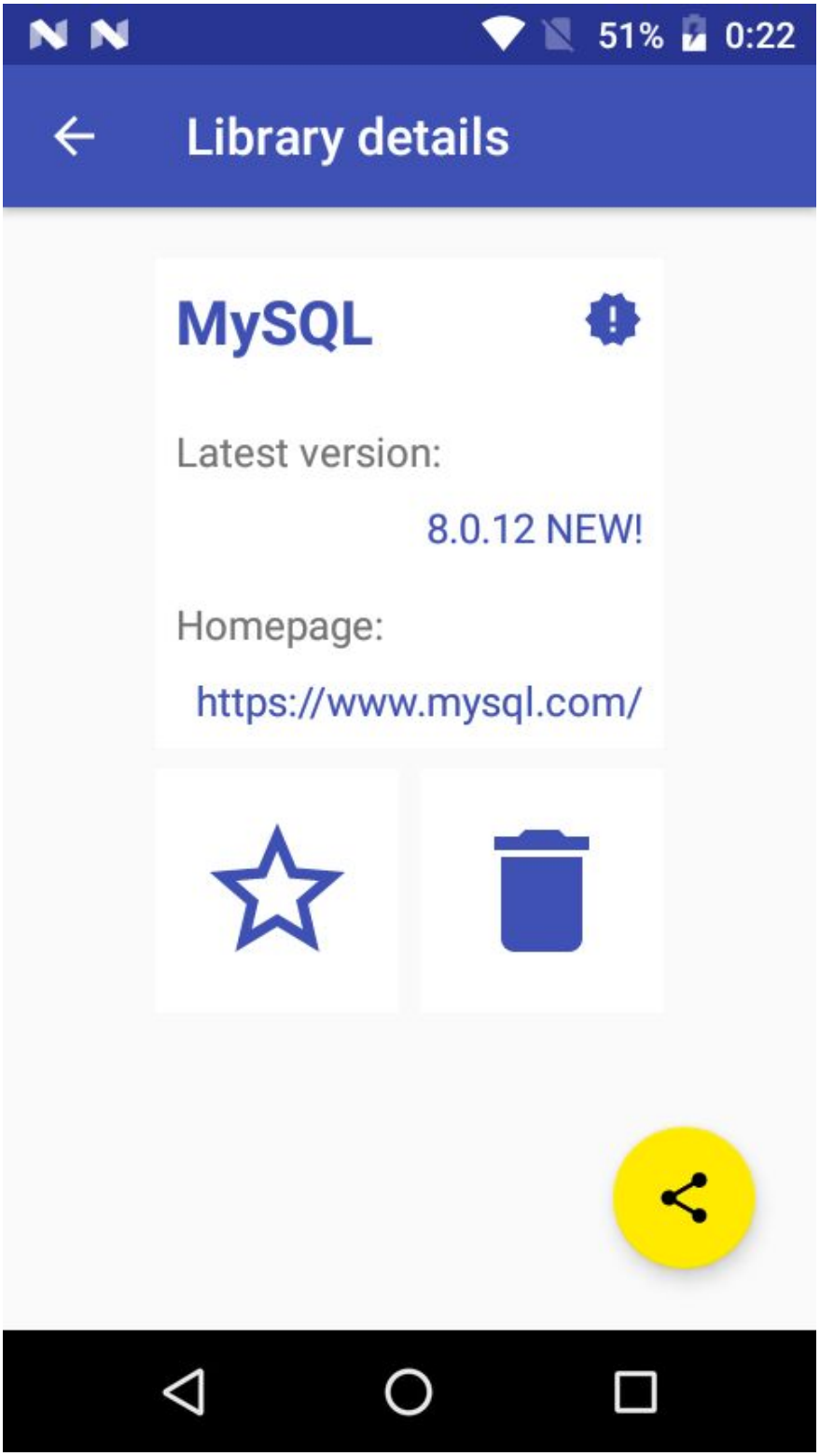
Quotes main screen contains the quote that appears on the widget. User can mark/unmark quote as favorite or unwanted. User can browse favorite quotes (swipe left) and unwanted quotes (swipe right) as well. User can share the actual quote or the list of favorite and unwanted quotes (with fab on the concerning tab of this activity).

CodeListActivity



App has a screen for libraries and software list. From the list user can view library or software details (tap, click) or view separate favorites page (swipe left). The used API has a default project list but user can add any github repositories to the list (fab on list screen). User can share their favorite list (fab on favorites screen).

CodeDetailsActivity



User can mark and unmark library or software as favorite. User can share the details of a library and software (fab). User can navigate to the homepage of the software, library or github repo (with tap, click) or copy the link (long tap)

Widget

App has a home-screen widget to provide quotes and inform about freshness of favorite libraries and software. If user clicks on the quote additional functions are available via applications activity. Information about freshness of libraries and software is provided on the top side of the widget. If the user clicks this area additional functions are available via applications activity (libraries and software main screen).

Key Considerations

How will your app handle data persistence?

SemiColon uses ContentProvider and to store data locally and Loader to show it in views. Database tables are: Name of github repositories added by the user; Favorite libraries, software and repositories; Last version of favorite libraries, software and repositories; Number of shown quotes; Favorite and unwanted quotes.

Describe any edge or corner cases in the UX.

User can open the application using its icon, this case user sees the main screen first. There the user can click two cards, that open software or quotes main screen. Software main screen consists of two tabs (“All”, “Favorites”). User can swype between tabs. Both tabs contains Floating Action Button. On “All” tab it handles “Add new github repository” event, on “Favorites” tab it handles “Share list of favorites” event. Clicking on software’s card on each tab invokes Software details activity, where detailed information appear. This activity contains three buttons (“Delete from list”, “Mark as favorite”, “Share”). “Delete from list” is only available if the repo is added by the user. Because of the three almost equally important tasks, this activity doesn’t contain FAB. Quotes main screen contains the quote that appears on the widget. User can mark/umark quote as favorite or unwanted by clicking buttons. User can browse favorite quotes (swipe left) and unwanted quotes (swipe right) as well. User can share the actual quote or the list of favorite and unwanted quotes (with fab on the concerning tab of this activity). App has a widget. It consists of two parts. A small line on the top shows information about the freshness of favorite software and the large part below contains a quote which changes on a hourly basis. If the user clicks on the quote in the widget, it launches quotes main screen, if the user clicks on the top of the widget it launches software main screen.

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

ButterKnife is used to bind elements easily and Schematic is used to create Content Provider with the needed 5 datasets in a couple of minutes. Gson is used to parse data the simplest way and Timber is used to make logs quickly.

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

Each activity of SemiColon will contain Admob view and the application will use one instance to handle Firebase 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

Tasks:

- Configure Firebase
- Configure Admob
- Configure Libraries
- Add every needed Support and Java stuff to gradle and make a successful sync

Task 2: Implement UI for Each Activity and Fragment

Tasks:

- Import SemiColon Icon
- Set colors and style basics
- Add unique font for the widget
- Build UI for MainActivity
- Build UI for CodeListActivity
- Build UI for CodeDetailsActivity
- Build UI for QuotesActivity
- Build UI for SemiColonWidget
- Code animations between views
- Sort the content of the values xml files to make them organized by views

Task 3: Local data

Tasks:

- Create data models
- Create databases
- Create Content Provider
- Create Loader
- Create JobDispatcher
- Wire those things together and test with fake data using LogCat
- Create and polish JavaDoc notes in new files

Task 4: API handling

Tasks:

- Create quotes API handler (its easier)
- Create software API handler
- Test network and API hazards
- Wire it with local data section together
- Test the whole data flow using LogCat
- Create JavaDoc notes in the handlers
- Review JavaDoc in local data handling files

Task 5: Wire together

Tasks:

- Wire MainActivity to the data-flow
- Wire CodeListActivity to the data-flow
- Wire CodeDetailsActivity to the data-flow
- Wire QuotesActivity to the data-flow
- Wire SemiColonWidget to the data-flow
- Test each event and its effect to data-flow
- Create JavaDoc for UI classes
- Review any JavaDoc where something changed

Task 6: Setup additional services

Tasks:

- Setup Firebase Analytics
- Setup Admob
- Test Firebase Analytics
- Test Admob
- Create and review JavaDoc where needed

Task 7: Polish the UI

Tasks:

- Fix elevations, animations, colors, texts, etc.
- Test UI on different screens
- Create JavaDoc for UI classes
- Create and review JavaDoc where needed

Task 8: Polish the code and files

Tasks:

- Add legal information everywhere
- Add missing xml headers to resources
- Remove temporary notes, TODOs, etc.
- Remove temporary Log calls
- Review gradle files
- Create and review JavaDoc where needed

Task 9: Submit the project

Tasks:

- Clean the project
- Commit and Push
- Submit