How to Use this Template

1. Make a copy [File → Make a copy...]
2. Rename this file: "**Capstone_Stage1**"
3. Replace the text in green

Submission Instructions

1. After you've completed all the sections, download this document as a PDF [File → Download as PDF]
2. Create a new GitHub repo for the capstone. Name it "**Capstone Project**"
3. Add this document to your repo. Make sure it's named "**Capstone_Stage1.pdf**"

GitHub Username: bruferrari

Crawler

Description

You ever feel tired looking for that news, post, or discussion thread that you want in Google? Nowadays Google brings to us an enormous amount of information, but sometimes that's not enough, Google itself isn't a reader and haven't to be. Crawler comes to solve this problem, organizing our search in most relevant articles, filtering by location and so on.

Crawler is a multi-category reader, it will search on the entire web about the topic or subject that you want to read, get these information and display in a beautiful Android Material Design interface.

You also can pin your favorite news, blog posts to check offline or in another most appropriate time, whenever you want.

Share the content with the others. Crawler can easily share your reading with your friends by sms and other supported apps.

Crawler get your location and brings to you the content in your native language if it's possible.

Visit the original version of content that you're reading with one tap on the source information. Your browser must be opened and the original site displayed for you.

Intended User

Everyone who wants to search and read with comfort, save the most important topics, etc...

Features

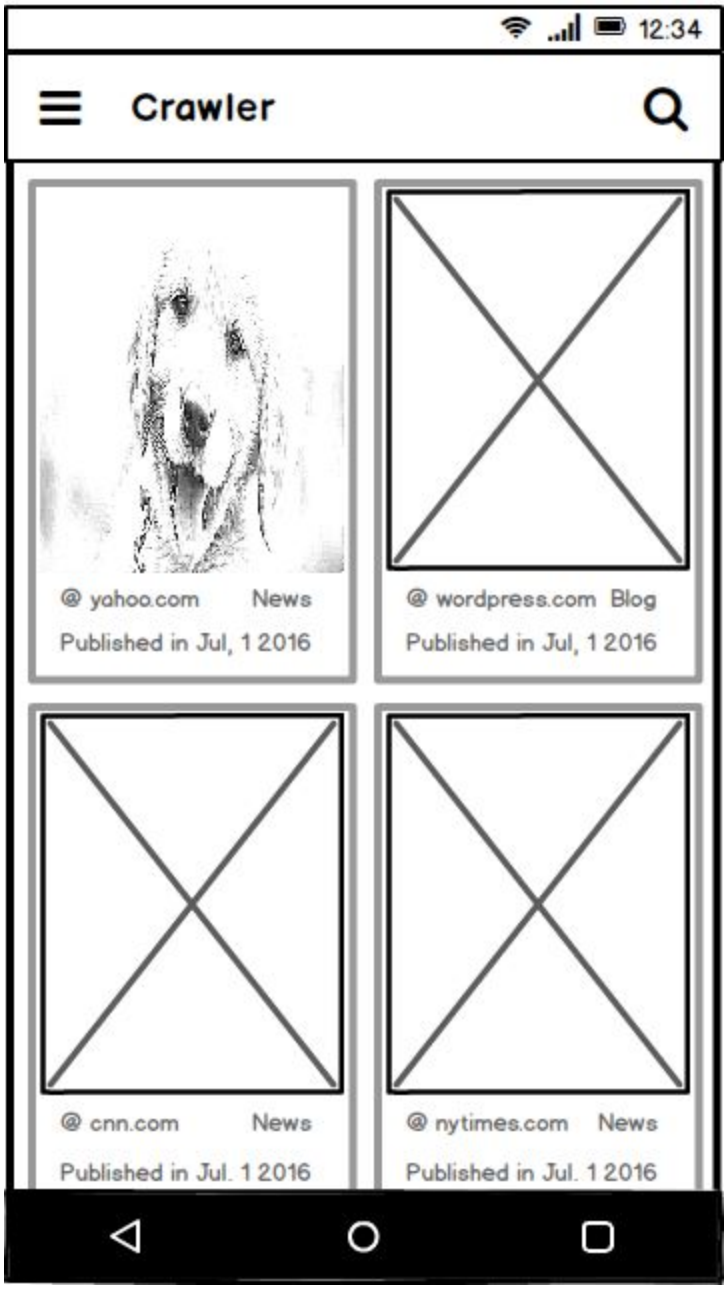
- Saves your most important reading (Pin)
- Structured search
- Beautiful design
- Share the reading with the others
- Rank the most relevant content to the search matter
- Provide a link to the original content

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 Photoshop or Balsamiq.

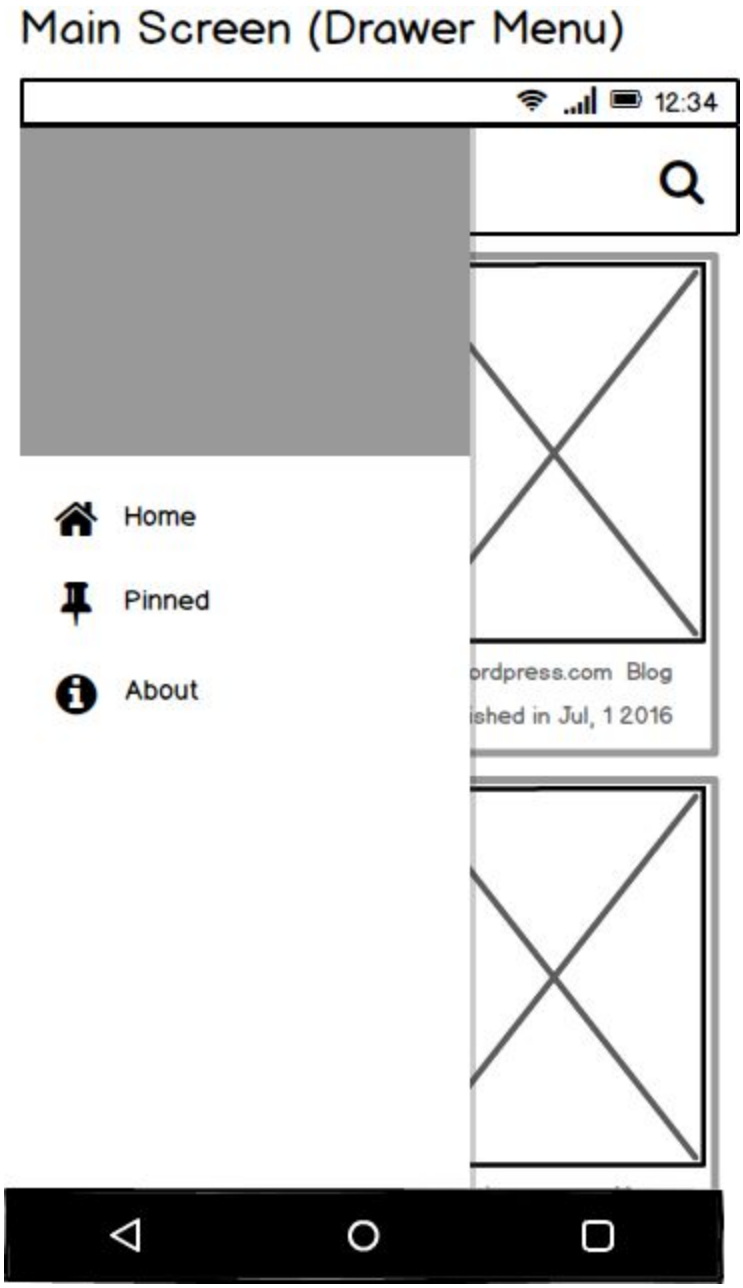
Screen 1

Main Screen



The screen above show the cards containing an image related to the content. This image must be provided by the api and resized to better fit on the placeholder.
Right below the image, the cards must show some important information about that content, like source, date and category.

Screen 2



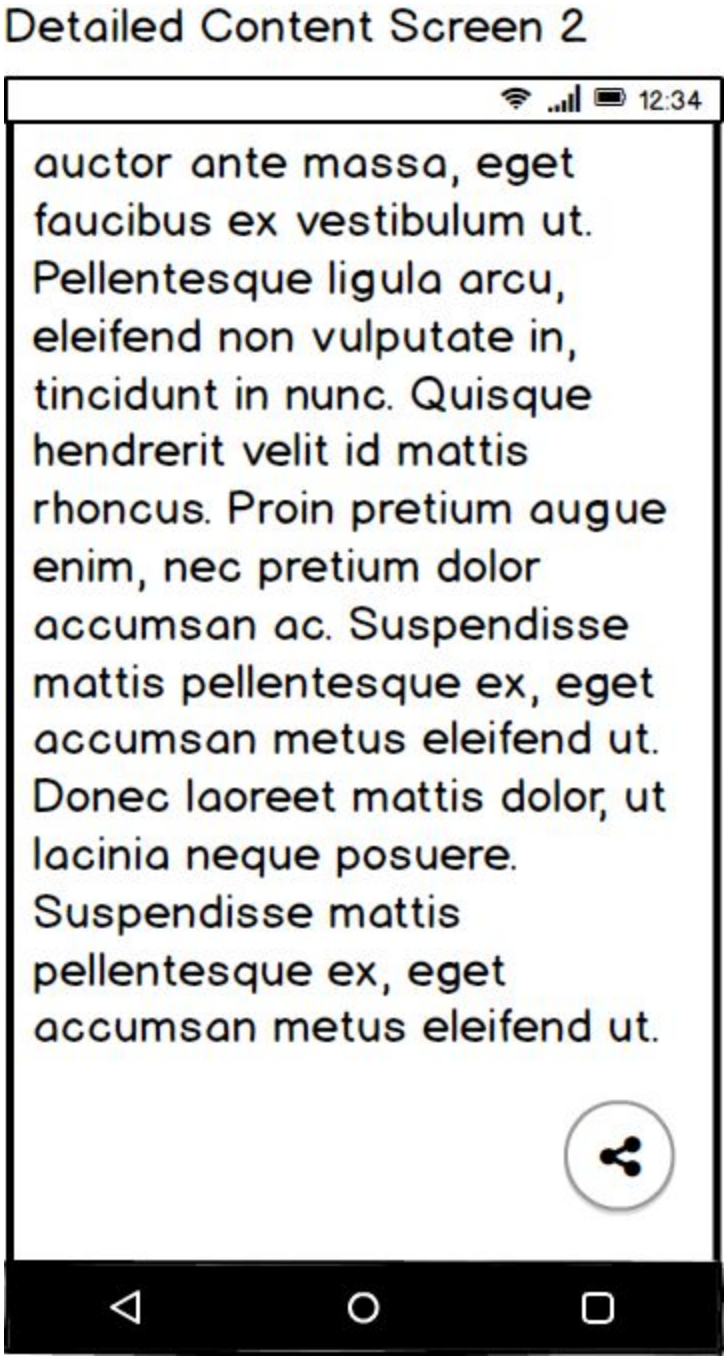
The image above refers to Drawer menu located on the Main Screen of application. It shows the most access menu and make the things easier to the user.

Screen 3

Detailed Content Screen

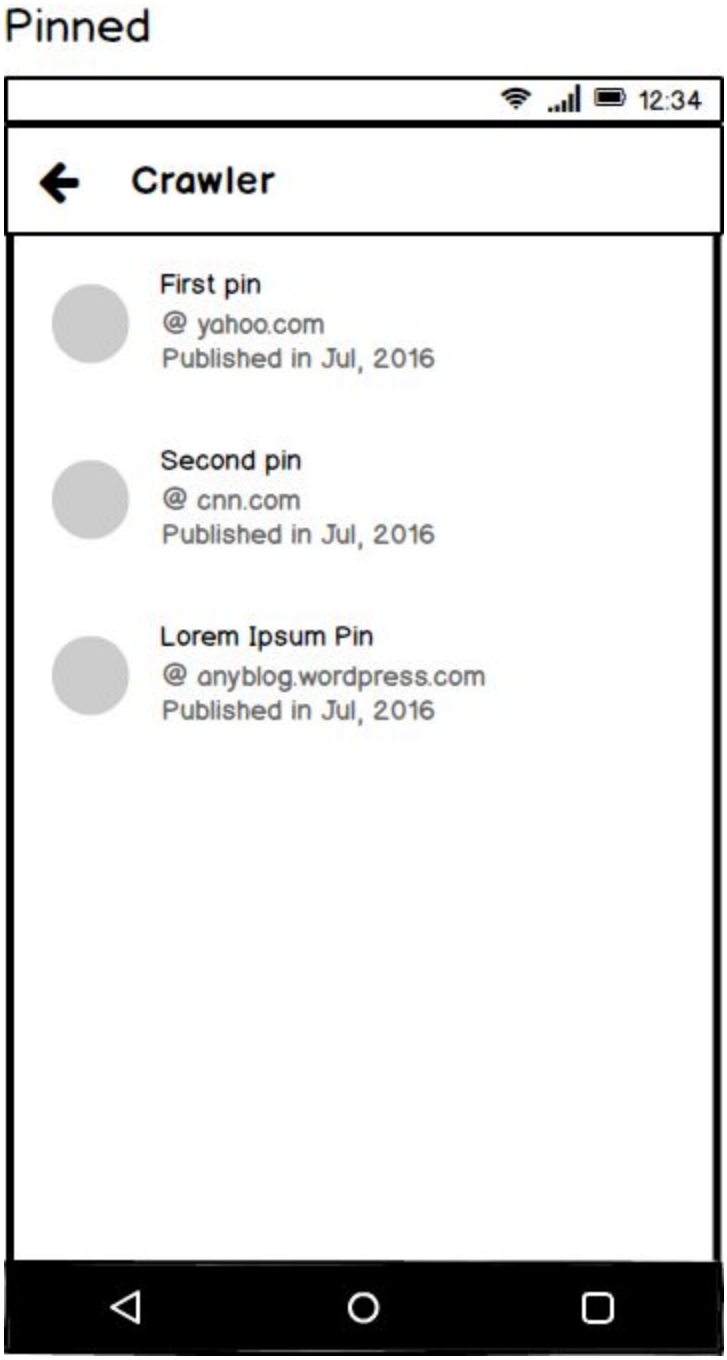
Here we have the detailed screen when you tap on a card that content will be loaded and displayed on this screen. This screen has a FAB to pin if the user doesn't want to read it right now, it also displays the information about published date and source. The source must point to the original link that content was taken from the web.

Screen 4



On the end of Detailed Content Screen we can see a FAB, actually, this FAB will follow the screen scroll until the end of content and disappear when the PIN FAB becomes visible again.

Screen 5



Here we have the Pinned Screen. It displays to the users the content that they have pinned before. All these pins will be available in offline mode to read even when you don't have an available internet connection.
On the left, we can see a round image that displays the image related to the pinned content and on the body of the row are displayed information like, title, source and published date.

Key Considerations

How will your app handle data persistence?

This app will consume an API from webhose.io to process all the data related to submittted query. The pinned posts will be saved on the device using SQLite in order to be available for reading without any internet connection. Pinned posts also can be deleted by the user.

Describe any corner cases in the UX.

The users will have always an option to return to the previous screen by pressing the native android bottom back button or the back button in the app toolbar. They have the same functionality.

The users can visualize better a content by tapping on the card or list, that content will be loaded and displayed in another specific screen with suitable space for the content.

Once the user is on the main screen of the app there is no back button in the toolbar, it will be replaced by a "hamburger" button that calls the drawer menu.

In the main screen, the toolbar will have a search button what allow the users to search whatever content related with the words placed there.

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

- Picasso: This library will manage all the workload with the loaded images, resizing, cropping and so on.
- Retrofit: It will handle the requests to the webhose's API, abstracting the complexity of code.
- Gson: The library made by Google to parse Java Objects into JSON and from JSON to Java Objects. It will abstract the complexity of build my own parser.
- Google Play Services: Crawler get the location of user and brings the relevant content in his native language. The location API's of Google Play Services will be very useful here in order to get the location of the users.
- AppCompat: The AppCompat library will be used to backward compatibility with older Android API versions.
- Design Library: Will be used to help on the implementation of Material Design on the app UI.

Next Steps: Required Tasks

This is the section where you can take the main features of your app (declared above) and decompose them into tangible technical tasks that you can complete incrementally until you have a finished app.

Task 1: Project Setup

- Set the development environment
- Configure libraries on Gradle
- Create a new Git repository
- Create a .gitignore file
- Link the project files with created Git repository
- Check if everything is working as expected

Task 2: Implement UI for Each Activity and Fragments

- Build UI for MainActivity
- Build UI for Detailed Activity
- Build UI for PinnedAcitivty

Task 3: Polish the UI

- Implement UI behaviors when the device is rotated
- Make UI compatible with other screen sizes.
- Implement animations on transitions
- Test the performance of UI mocking elements on that

Task 4: Implementing the API consumer

- Create account on Webhose.io to have access to the API
- Implement the authentication with Webhose.io API
- Implement the consumer using Retrofit and Gson
- Test the performance of API

Task 5: Put retrieved data on the screen

- Remove data mocks from UI
- Put the retrieved data instead of mocks
- Test the performance

Task 6: Finishing

- Make refactoring on entire code
- Check if everything is working as expected
- Test everything

Add as many tasks as you need to complete your app.

Submission Instructions

1. After you've completed all the sections, download this document as a PDF [File → Download as PDF]
2. Create a new GitHub repo for the capstone. Name it "**Capstone Project**"
3. Add this document to your repo. Make sure it's named "**Capstone_Stage1.pdf**"