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: wssholmes

Headlines

Description

The brief idea here is that my app uses the “newsapi”(https://newsapi.org) to provide user the headlines and a brief description of news articles from various news sources based on news categories like business, entertainment, sports,etc., or even a list of headlines based on preference like popular, recent and top.

The app will also provide a detail screen for each article and a link which will take the user to the detailed article source page. Users will also have the functionality to share the article link via various sharing apps in their devices. Also there will be additional functionality to for users to mark articles as favourites, which they will be able to view any time they want, to follow up on later when it's convenient for them.

The main attraction here is to keep people up to date about various events, based on their preference, convenience, interest, etc. without consuming much of their time by providing snippets of information which they can read quickly and can follow up on either at that very moment or sometime later when they are more relaxed. Also they will be getting their information from not one but various sources, most of them with a specialized domain, thus providing our users with the latest and greatest happening all over the world be it science, technology, business, sports or politics.

Intended User

This app in general targets people with a busy lifestyle, who despite having a hectic schedule, like to be aware of what's happening in the world around them. It can be a student in college or in school who is busy with his studies and submission deadlines, but would also like to be aware about new discoveries in science and technologies or how did their favourite sports team performed last night or the latest rumours about the upcoming iPhone or Nexus.

This app is also for an average everyday working Joe or a daily commuters who likes to gets all his updates about various events while they are having their lunch or travelling in a train/bus/etc., and like the news to be short and concise and to his preference, so that they can catch up on most without losing much of their precious time and also allows them to mark their favourite articles on which they can follow up during a more comfortable time or even on the weekends.

This app is also for users with limited valuable data plans, who don't want to waste their time browsing various news websites and loading articles, when what they really want is a gist. And in the process wasting their precious data plan as well as time, which they could have put to much better use.

Features

List the main features of your app. For example:

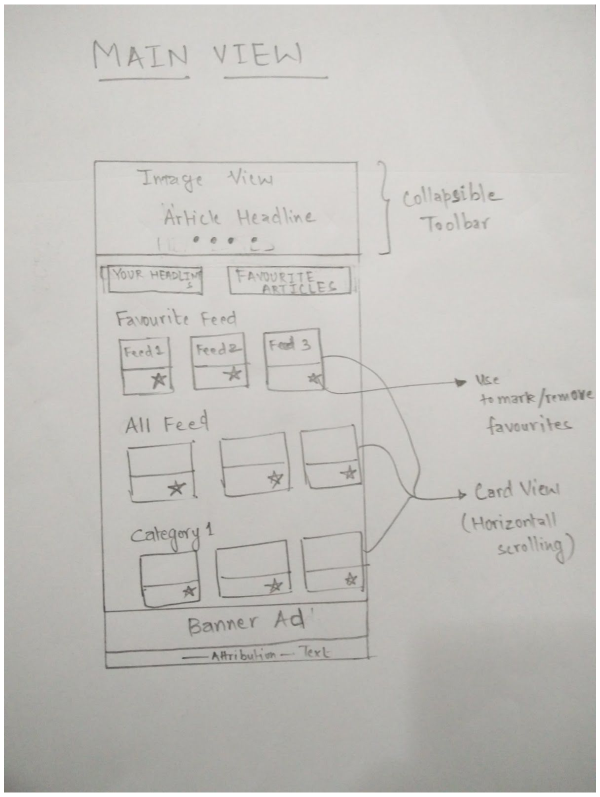
- Users can save their favourite articles.
- Users can share the links of the news articles via any sharing app on their phone.

- Directs user to the webpage containing the detailed article.

User Interface Mocks

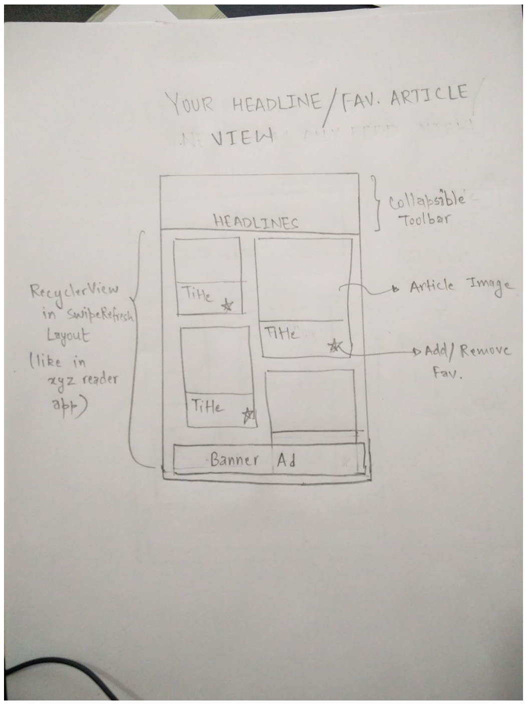
These can be created by hand (take a photo of your drawings and insert them in this flow), or using a program like Photoshop or Balsamiq.

Screen 1

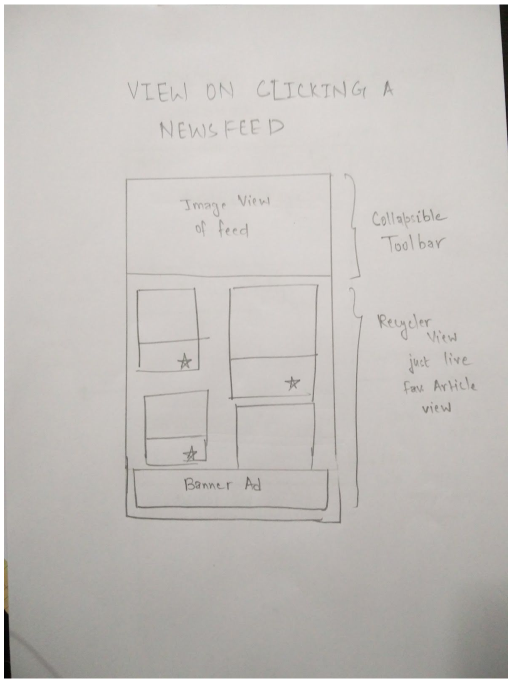


This is the screen shown when the app is launched. The description for various components is provided in the image.
NOTE: This activity will contain the attribution text for the api I am using for the app.

Screen 2

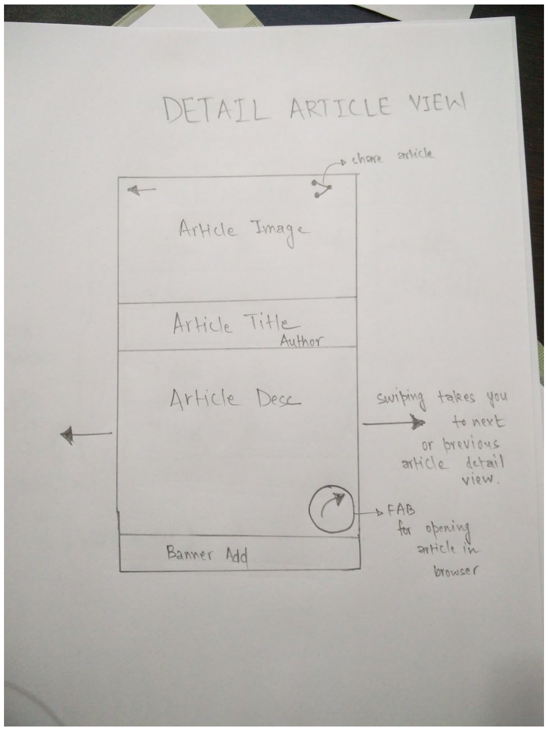

YOUR HEADLINE/FAV. ARTICLE VIEW
HEADLINES
Collapsible Toolbar
RecyclerView in SwipeRefresh Layout (like in xyz reader app)
Title
Article Image
Add/Remove Fav.
Banner Ad

This screen is launched when the user clicks on the “Your Headline” or “Favourite Articles” button in the main view. The content will differ according to the explanation provided in the Content Provider section below.

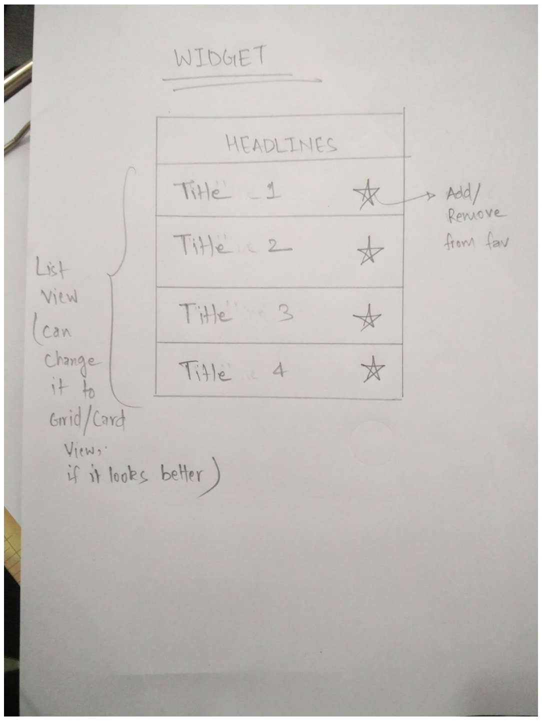

VIEW ON CLICKING A NEWSFEED
Image View of feed
Collapsible Toolbar
Recycler View just like Fav Article view
Banner Ad

This view will launch when user clicks on any of the newsfeed on the main screen.

NOTE: There will be at most 3 buttons at the top(depending upon api capabilities) like that in main view allowing user to see the news in at most three preference : top, recent and popular.


DETAIL ARTICLE VIEW
share article
Article Image
Article Title
Author
Article Desc
swiping takes you to next or previous article detail view.
FAB for opening article in browser
Banner Add

This view is launched when any article is selected from any of the view or the widget.


WIDGET
HEADLINES
Title 1
Title 2
Title 3
Title 4
Add/ Remove from fav
List View (can change it to Grid/Card View, if it looks better)

This will be the widget for the app.
NOTE: I might change the list view to grid or card depending upon how they look and perform.

I have not provided mocks for the UI of tablet as it will be simple modifications of the views already shown above.

Add as many screens as you need to portray your app's UI flow.

Key Considerations

How will your app handle data persistence?

The app will have a Content Provider and 3 SQL Database table. The first table will contain all the available newsfeed from the api, this will allow the app to always display the main home screen. This table will be updated once a week or so as the api website doesn't explicitly say about expanding the newsfeeds, neither it says anything about their number being a constant. This table will also store information about whether the user has marked this feed as favourite or not (via the star button available on the card view). The newsfeeds marked as favourite will be used to populate the second table.

The second table will contain the information about headlines from the newsfeeds marked as favourite by the user. The content of this table will be used to populate the recyclerview that opens on clicking the "Your Headlines" button on the homescreen as well as the list view widget that will be provided with the app. This table will be updated using the sync adapter frequently (currently I was thinking every hour, considering news change fast).

The final table will be used to store the article marked as favourite by the user. This will be used to show populate the view when the user presses the "Favourite Articles" button on the main screen. The reason I am implementing a separate table, instead of storing the information in the second table itself, is that I want to empty the second table every time I update it using sync adapter, so that the "Your Headlines" view and the widget contains always the new news and not the news they have saved for later reading.

This will give me a better control over the data as each type of data is stored in separate tables making the app less bug riddled, the code clean and reduces the chance of any accidental insertion (I will provide checks to ensure same article is not added more than once) or deletion.

Describe any corner cases in the UX.

The corner case in the UX will be when I empty the second table, and reload and it contains some of the articles that the user has already marked as favourite. Now, if the user tries to mark some of these articles as favourite again, then that will result in duplication in the 3rd table.
To resolve this at first check the presence of an article in the 3rd table before adding it. I will provide a feedback to the user via snackbar that the article was already in the favourites.

Also, if does not impact the performance much, I will during the sync adapter update mark the the newly added article as whether they are already in the favourites or not. And will show the same using the appropriate star image (empty if not and filled if yes) button in the respective views. In this way the user will always be informed that whether the article is already present in their favourite list.

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

I will be implementing Picasso libraries for loading and caching of all images I am going to use in the app.

Describe how you will implement Google Play Services.

I will implement the Google AdMob and Google Analytics in my app. The former will be used to monetize the app, as I am planning to launch this app on the Play Store (and there is really no sense in making a Paid News App).

The second will be used get reviews about how well/bad my app is doing, where is it being used, what features people are using the most/least, etc. I will use all this information to keep updating and improving my app and understand how user's respond to various features and UI/UX implementation. This will not only help me in improving this app but also any other apps I develop in the near future.

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

I will start out by first implementing the data persistence part of the app,i.e., the content provider. I will also add the sync adapter during this phase as these will be the core of my backend implementations for the app.

Subtasks:

- Configure libraries
- Determine target and min Sdk
- Implement Contracts
- Implement Sync Adapters
- Adding Junit test to check the database implementation
- Adding checks to pinpoint the cause of problems in polling data from the server.

Task 2: Implement UI for Each Activity and Fragment

After implementing the backend for the app, then I will start building the UI for the mobile. This will involve building all the activities and fragments like for the main activity, the favourite articles, the detailed article, the headlines, etc. I will also add the Picasso library here while building this UI.

Subtasks:

- Build UI for all activities for mobile.
- Implementing Asynctask for fetching news from newsfeeds other than favourites.
- Implementing cursor loaders for loading data in the activity.
- Implementing image transition in relevant places.
- Making the app conform to the material design.
- Making the app conform to the accessibility checklist.
- Implementing system to provide user's feedback if anything goes wrong, e.g, if user is not connected to internet, if server is down, etc.
- Implementing empty views for the recyclerview to provide feedback to users.
- Making the app conform to the localization checklist.
- Implement and apply Picasso Library.
- Implementing the sharing feature in the detailed article view.

Task 3: Adding UI for the tablet

After implementing the UI for the mobile view, I will next make the necessary additions for the app UI to be tablet optimized. This will also involve making changes in the way the activities and their fragments are called and inflated on the screen.

Subtasks:

- Create additional optimized layouts for the tablet.
- Implement two pane layout for Headlines View.

Task 4: Adding the Google Play Services

After completing the UI for both phones and tablets, the next step will be to integrate the above mentioned google play services to the app.
Describe the next task. List the subtasks. For example:

- Adding the libraries to the gradle file.
- Making changes to the layout for implementing advertisement banner.
- Adding the Google Analytics

Task 5: Adding the widget for the app

After all the services and the UI, UX has been implemented the final step will be to add the list widget showing the headlines from the favourite newsfeeds.
Describe the next task. List the subtasks. For example:

- Implementing the widget provider.
- Implementing the widget remote service.
- Adding a preview for the widget.
- Signing the app.

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**"