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: kareem-adel

The verge

Description

The verge is a news application that delivers the latest news with ease, it enables users on tight schedules to be able to keep in touch with the latest technologies and scientific discoveries on the run.

Intended User

Technology enthusiasts.

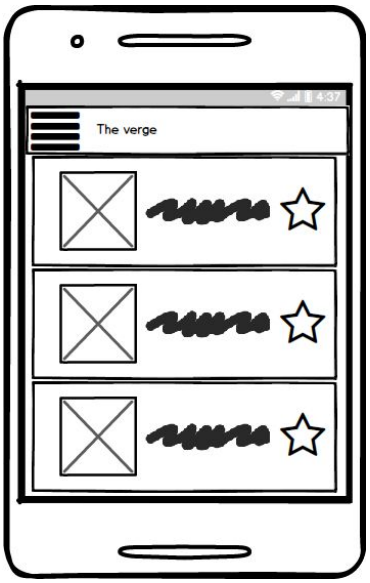
Features

- Getting information about the latest technologies.
- Offline mode for later reading.
- Navigate top stories.
- Stay updated from your home screen using a widget.

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.

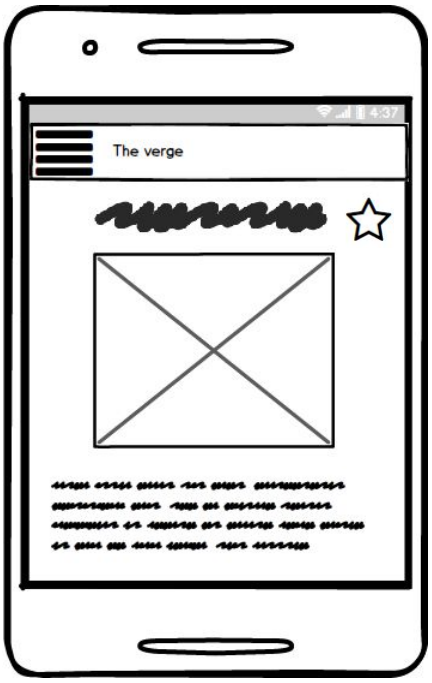
Screen 1



This is the main screen that contains the chosen type of news (top, latest or favourite).
The user sees the news image and the headline next to it.
The user can define a news to be a favourite by pressing on the star.

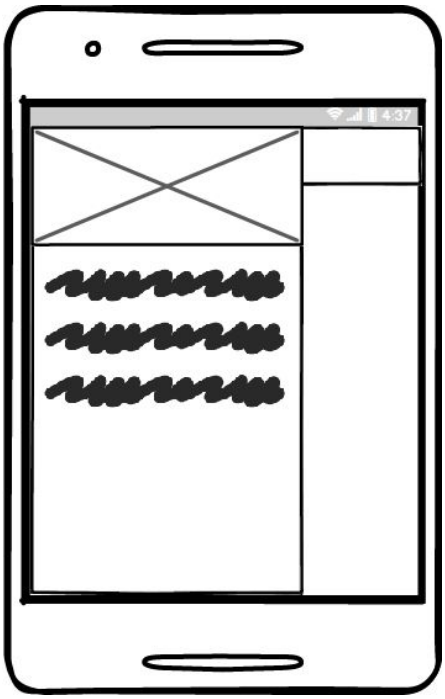
The user goes to screen 2 by clicking on an item.

Screen 2



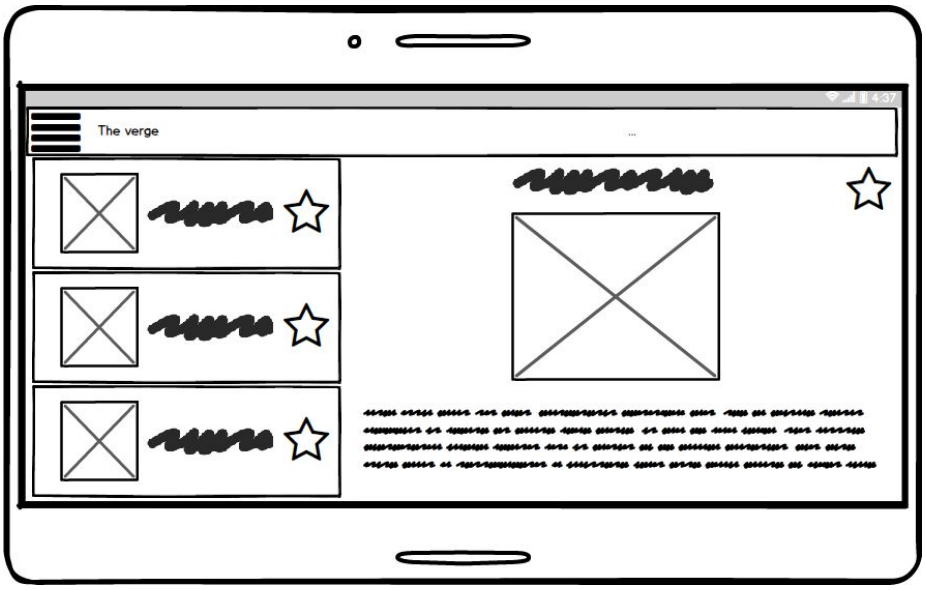
This is the detail screen that includes more information about the news, it includes the title, description, date, the source, the image, and a favourite star.

Screen 3



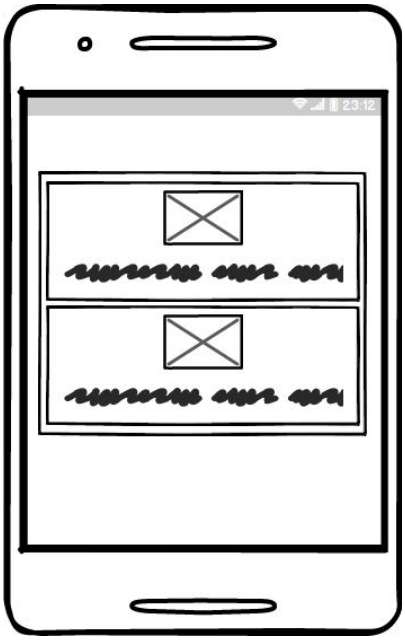
This is the navigation drawer screen that will contain the navigation items such as top, latest and favourites.

Screen 4



This is the tablet screen that includes both screen 1 and screen 2 with the same functionality and description.

Screen 5



This is the home screen showing the widget that contains a list of the latest news.

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

Key Considerations

How will your app handle data persistence?

The application will contain a content provider that handles data flow in the application, it will contain information such as news articles, and favourite news.

Describe any corner cases in the UX.

When the user marks an article as favourite in the detail view, the effect is reflected in the main list view.
When the user reads a news and returns to the main activity, the list view shows the last item clicked.
When the user rotates the screen, the list view shows the last item visible to the user.

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

Glide would be used for image loading.
Iconics-core for loading icons dynamically in the application
Butterknife for binding views with annotations and minimal code.
Materialdrawer for managing the navigation drawer items.
Retrofit for handling asynchronous requests to the api server.
Cardview for showing items as cards.
Recyclerview for showing items efficiently.

Describe how you will implement Google Play Services.

The application will use admob to display ads, and firebase job dispatcher to update the contents of the content provider each period.

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

- Configure the mentioned libraries and gradle configurations.
- Initialize the project with a main activity and two fragments, a listview and a detail fragment.

Task 2: Implement UI for Each Activity and Fragment

- Building the layouts for the activity and fragments as mentioned in the mockups.

Task 3: Building a library for news fetching

- Configure retrofit to retrieve news from the news api https://newsapi.org/the-verge-api

Task 4: Building the content provider

- Implement the content provider that will provide articles data.
- Use the output of the news fetching library to update the content provider periodically.

Task 5: Binding views with the content provider using a Loader

- Synchronize the views with data from content providers using loaders.

Task 6: Building the widget

- Build a widget that contains an updated list of latest news from the verge.

Task 7: Building the JobScheduler

- Build the JobScheduler to make sure that the content provider has updated articles every 15 minutes.

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**"