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: ramkrishna757575

KhanaKhoj

Description

This app helps users to search for restaurants nearby. It considers the current location of the user and shows restaurants within a certain radius. It shows the pricing, rating, cuisines, etc. of the restaurants, so the users can find the best restaurant at reasonable prices.
It will be very helpful for people who are travelling and need to find some place to eat.

Intended User

Anyone who is hungry and needs to find a place to eat ;)

Features

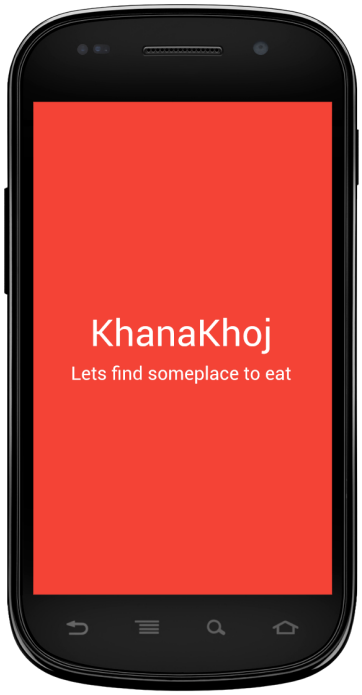
Here are the important features of this app:

- Based on location, shows restaurants nearby.
- Restaurants can be searched by name, cuisines, etc.
- A restaurant can be bookmarked, and will be stored in the app
- Shows details of restaurant like ratings, reviews, prices, etc.

User Interface Mocks

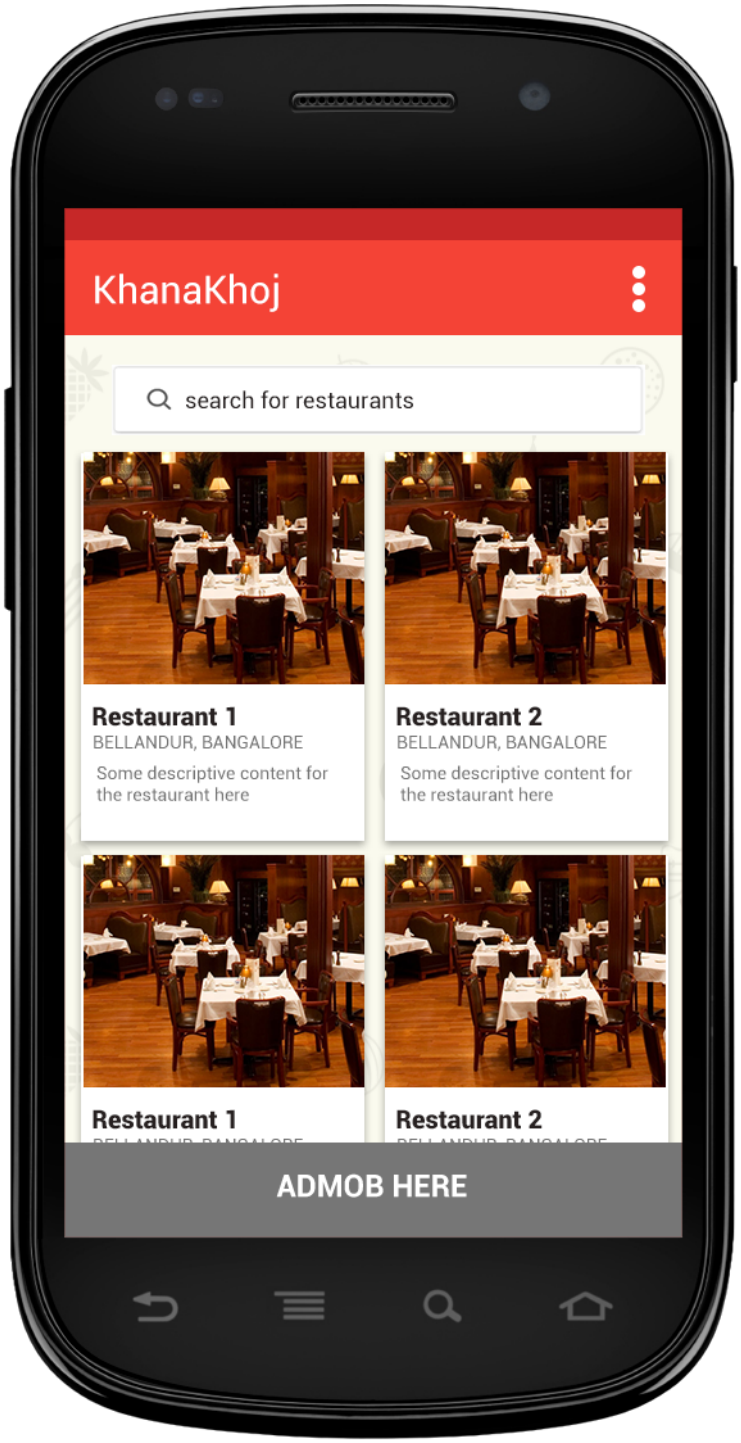
These can be created by hand (take a photo of your drawings and insert them in this flow), or using a program like Photoshop or Balsamiq.

Screen 1



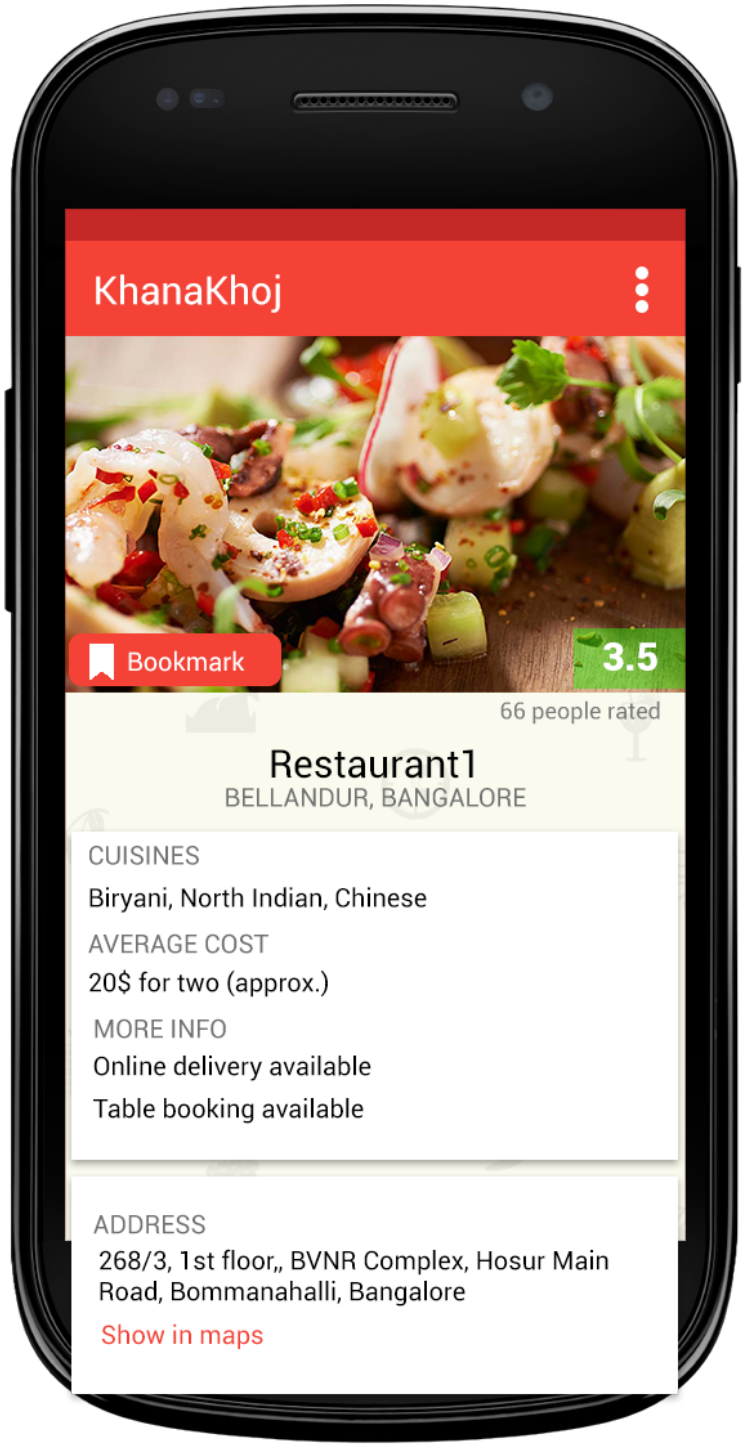
This is the splash screen of the app. At this screen, the location of the device will be detected using google location service. If the location is not determined successfully, a default location will be used.

Screen 2



At this screen a list of restaurants will be displayed based on the location found previously. The search bar can be used to search for restaurants in the nearby locations.
Each item will contain certain details of the restaurant like name, rating, location, image, etc.

Screen 3



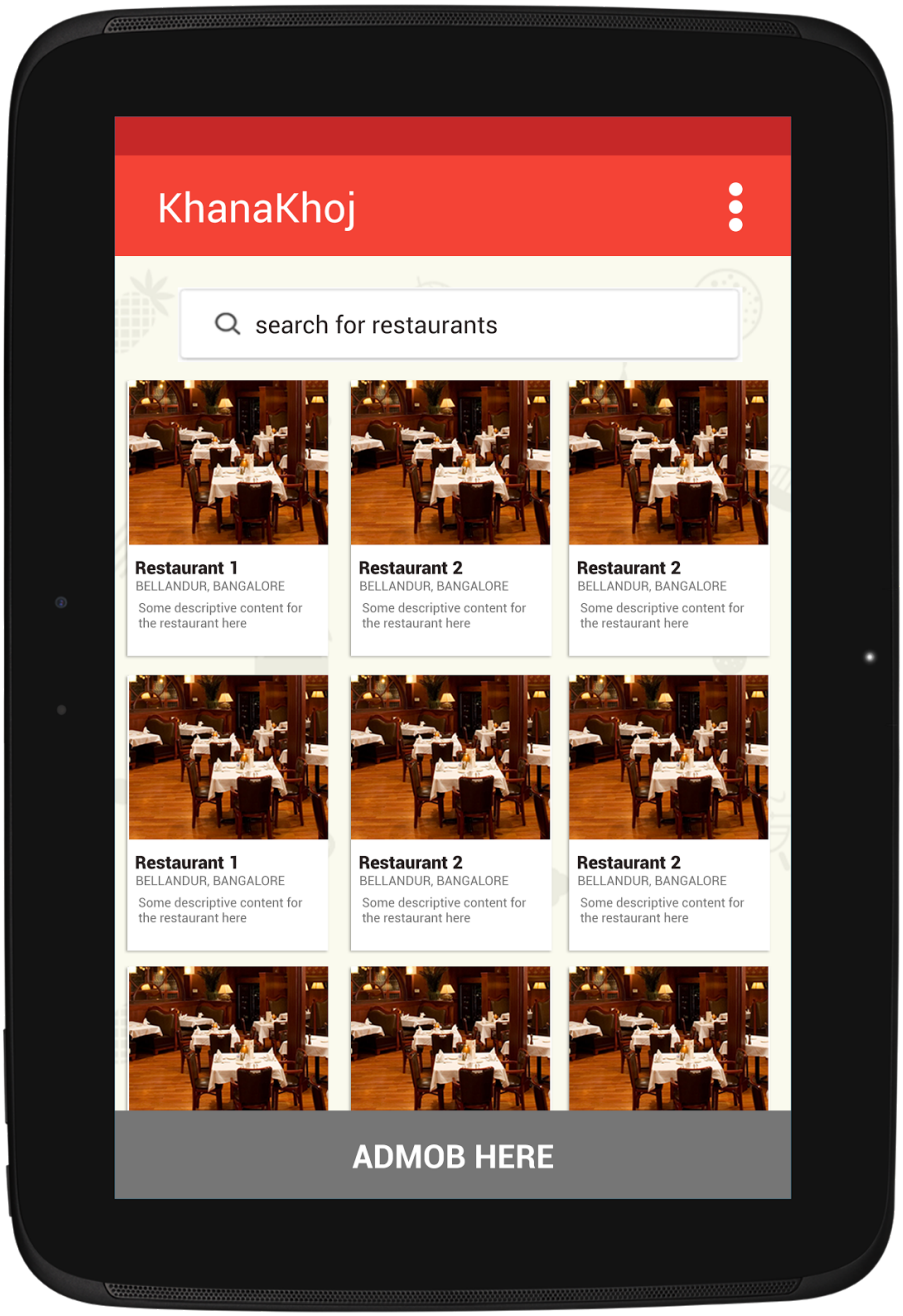
At this screen, the details of the restaurant will be displayed. There is an option to bookmark the restaurant, and it will be saved locally in the app using content provider. In the ‘ADDRESS’

section of the screen, when the ‘Show in maps’ is clicked, the address of the restaurant will be shown in an external app that supports Maps.

In the settings option in the title bar, there will be an option to show the list of bookmarked restaurants. Clicking on it will open the Screen 2 above and show the list of bookmarked restaurants. Clicking on any restaurant on that screen will open the Screen 3 above showing the details of that restaurant.

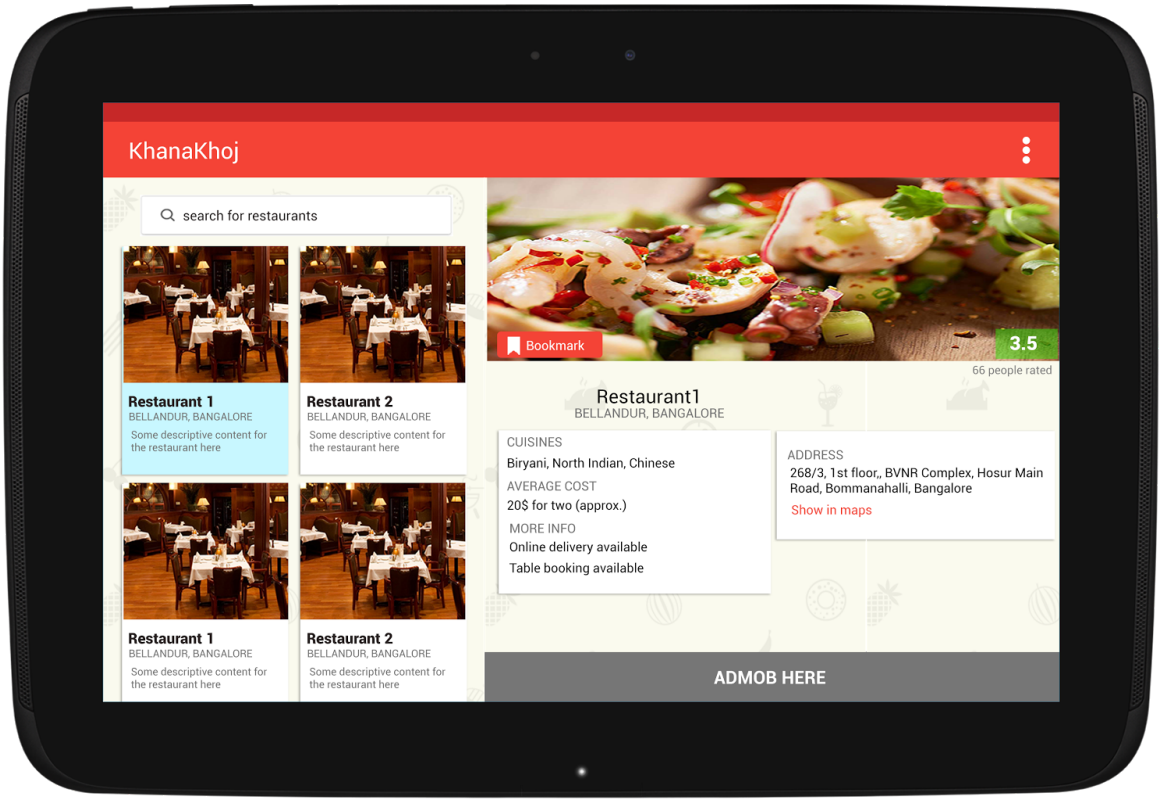
When in the Screen 3, if seeing bookmarked restaurants, the ‘Bookmark’ button will change to ‘Bookmarked’. Clicking on it will remove that restaurant from the database and from the list of bookmarked restaurant.

Screen 4



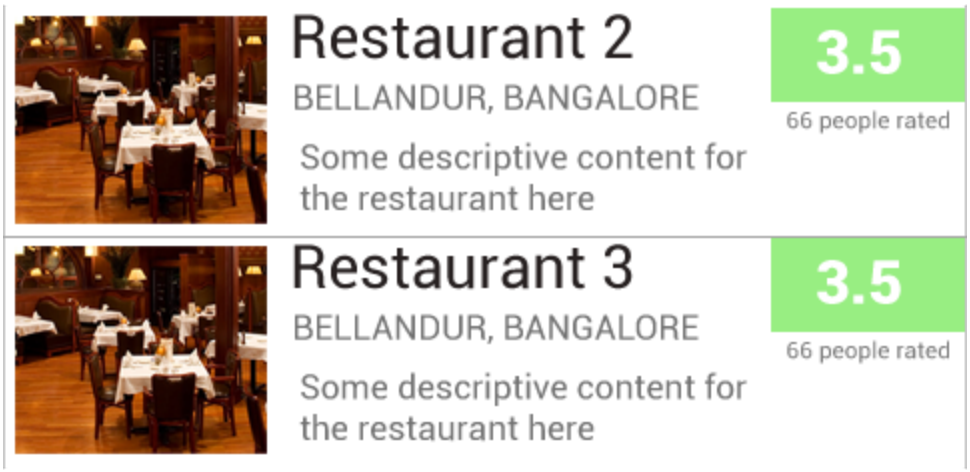
Screen for Tablet Layout in portrait mode to show list of restaurants

Screen 5

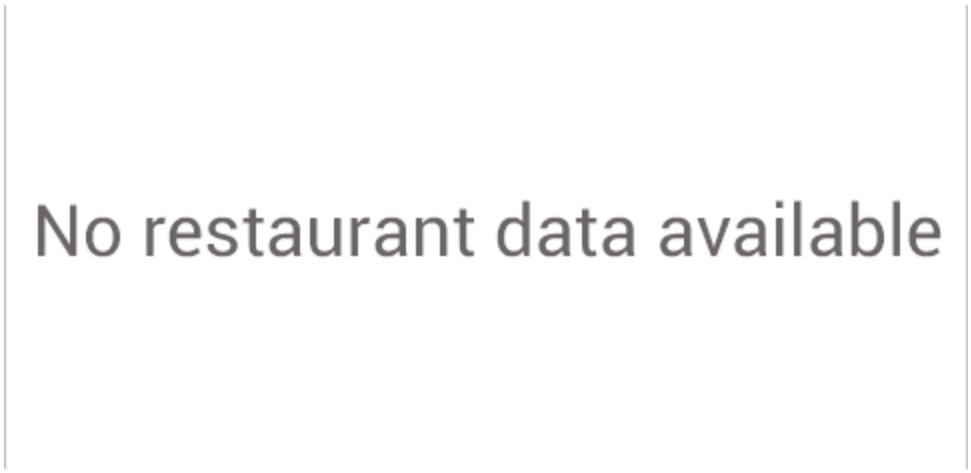


Screen for Tablet Layout in landscape mode to show list of restaurants and the detail of the selected restaurant.

UI design for Widget



List of bookmarked restaurants shown in widget.



Design of widget when, no restaurants bookmarked(i.e. Data in local DB is empty).

Key Considerations

How will your app handle data persistence?

The app will store bookmarked restaurants in the local database using content provider. When un-bookmarking the bookmarked restaurant, the records related to that restaurant will be removed from the database.

Describe any corner cases in the UX.

One corner case is when the app is unable to determine the location at the first screen. At this point, there will be a retry button to do it again. If no internet is available, a dialog box will appear informing the user to connect to internet and retry.

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

- Picasso to handle the loading and caching of images. This is because it has nice features like Handling of image caching, downloading, download cancellation, image transformations, etc. And it is very easy to use. Also an error image can also be set if the actual image fails to load.
- Retrofit - to handle API requests and also parse the returned data from the APIs. It is also very easy to use and saves a lot of time by avoiding to do a lot of redundant work, like writing code to parse JSON data.
  Most importantly, it can be used to make request Asynchronously on a separate thread from UI thread (equivalent to AsyncTask).

Describe how you will implement Google Play Services.

- Google Play Services: Location - At the first screen, this will be used to get the current location of the device. Base on this location, the list of restaurants within a certain radius of this location will be displayed in the list. Also, when the user is searching for restaurants using the search bar, the search results are again based on the location(i.e within certain radius of the current location).
- Google Play Services: AdMob - At the second screen(i.e the list of restaurants screen), the AdMob service will be used to display ad at the bottom of the screen.

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

Following are the steps to configure the project.

1. Clone the project from the repository using the following command at your git terminal: git clone https://github.com/ramkrishna757575/Capstone-Project.git
2. Go to the following link in your web-browser: https://developers.zomato.com/api and click on the 'Generate API key' button.
3. Follow the steps required to generate the API key.
4. Once you obtain the API key, go to the root directory of the project and open gradle.properties file.
5. Add the following at the end of the file and save ApiKey="API Key you obtained above including the double quotes"
6. Open the project in Android Studio and build the project.
7. Run the project in emulator or a device.

Task 2: Implement UI for Each Activity and Fragment

- Build UI for Splash Screen Activity
- Build UI for Restaurant List Fragment
- Build UI for Restaurant Detail Fragment

- Build UI for tablet layouts
- Build UI for Widget

Task 3: Implement Google Play Services

- Implement Google Play Services: Location to get the location of the user and save in SharedPreferences.
  The implementation will be done as per the tutorials provided here:
  https://developer.android.com/training/building-location.html
- Implement Google Play Services: AdMobs to get ads in the Restaurant List Fragment.
  The implementation will be done as per the guide here:
  https://firebase.google.com/docs/admob/android/quick-start

Task 4: Implement Activities and Fragments

- Implement Splash Screen Activity
- Implement Restaurant List Activity
- Implement Restaurant Detail Activity
- Implement Restaurant List Fragment
- Implement Restaurant Detail Fragment
- Implement Bookmarked Restaurant List Fragment
- Implement Bookmarked Restaurant Detail Fragment
- Handle the implementation for Tablet Layout

1. Once location is obtained from the Google Locations API(on splash screen), then the Restaurant List Activity will be called.
2. In this activity, the zomato API will be used to get the list of nearby locations. This API call will be made using Retrofit library. Once this list is obtained, GridView will be used to show the list of restaurants in the Restaurant List Fragment. A corresponding Adapter(extended from BaseAdapter)will be used to populate the GridView with the data.
3. When a restaurant is clicked, the Restaurant Detail Fragment will open, make a call to the Zomato API to get the restaurant details and display it in the UI.
4. In the above screen, if a restaurant is bookmarked, using Content Provider, the details of this restaurant will be saved in local database, and this restaurant will be removed from this list (It will show up in the Bookmarked Restaurant List Fragment as mentioned below).
5. From the Settings menu, if Bookmarked option is selected, then Bookmarked Restaurant List Fragment will show up. The data in this fragment will be acquired from the Content Provider. Loader will be used to move this data to the Views (No API call required).
6. Clicking on any restaurant will take to Bookmarked Restaurant Detail Fragment which will get the restaurant data using the Content Provider.

7. Here, the user has option to remove a restaurant from bookmark. By doing so, the data of this restaurant will be removed from the database via Content Provider, and this restaurant will be removed from the list.

Note - All the API calls will be made using Retrofit library. So the use of AsyncTask for the same is not required. Retrofit can be used to make Asynchronous calls on separate thread from UI thread. The same will be used in the App.

Task 5: Implement Widget for App

- Implement Widget for App

1. Implementation of this widget will be done by using the RemoteViewsService, to get the data from content provider and show up in the widget in the form of list.
   The following guide will be used as a reference to implement the widget:
   https://developer.android.com/guide/topics/appwidgets/index.html

Task 6: Test App, find bugs and crashes and handle them

- Test App with all possible scenarios
- Find bugs and fix them
- Find corner cases and handle them

Task 7: Make App ready for release

- Generate Signing keys for the App
- Make it ready for release in play store

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**"