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: maheshgaya
https://github.com/maheshgaya

Cool Wallpapers

Description

Cool Wallpapers is a photo sharing app where you can find and share awesome wallpapers.

Intended User

Teenagers

Features

Allow users to:

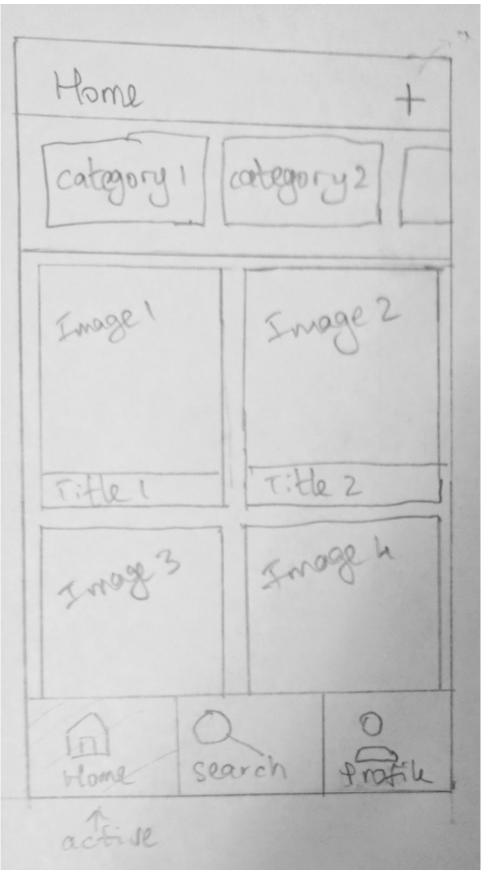
- upload photos
- set photos as wallpaper on their phone
- create profile and follow others
- like photos

Login/Sign-up

The app will start with a login/signup page. Users will be able to select a list of categories that they want wallpapers from. This will become the basis of the user's feed (Home Page).

User Interface Mocks

Home Page

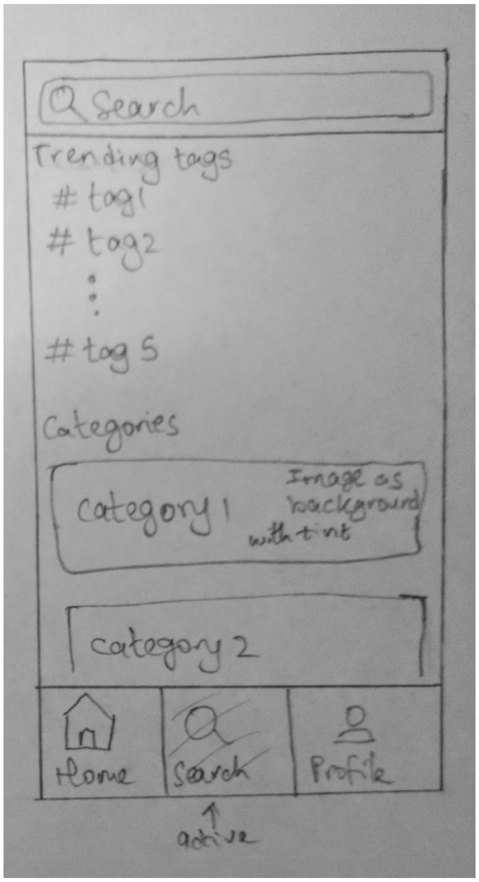


When the user opens the app, it will show the home page. This page can also be accessed by tapping on the first icon on the bottom navigation. It will show an image feed for the user based on pre-selected categories (the Gridview). It will also show all the categories (10 max) at the top, so that the user can just tap on them and explore images of that particular category.

The screen will have a bottom navigation that will allow the user to navigate to their “Home Page”, “Search Page” and “Profile Page”.

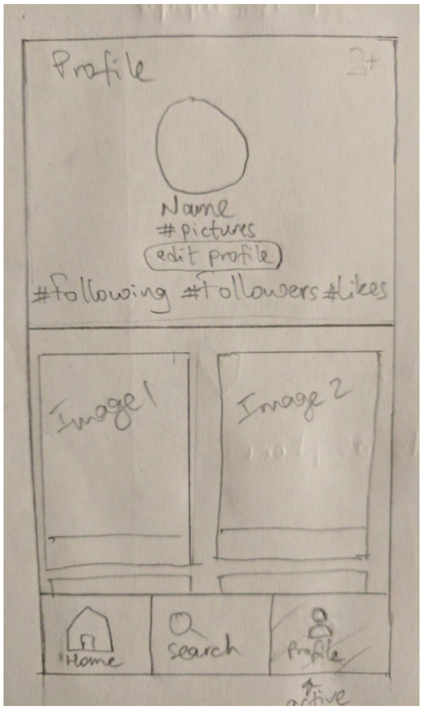
The “+” button will allow the user to upload a new image to the app.

Search Page



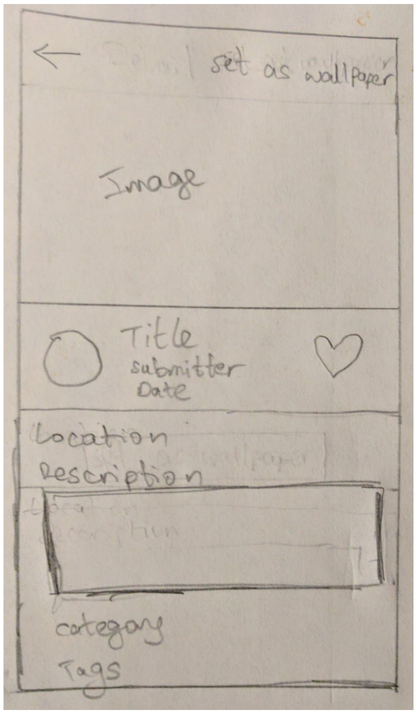
This Search Page can be accessed by tapping on the second icon on the bottom navigation. This will show the trending tags and the categories in a list format. Users will be able to search by tapping on the searching edit text. The background image of the category will be a tint and will contain an icon based on the category.

Profile Page



This will be the profile page. It will show the images that the user uploaded. It will also show the number of followers, number of people the user is following and the number of likes the user got on his uploaded images. User will also be able to edit name and the profile picture. In the case of other users that this user is browsing, the “edit profile” will turn into “Follow” or “Following”.

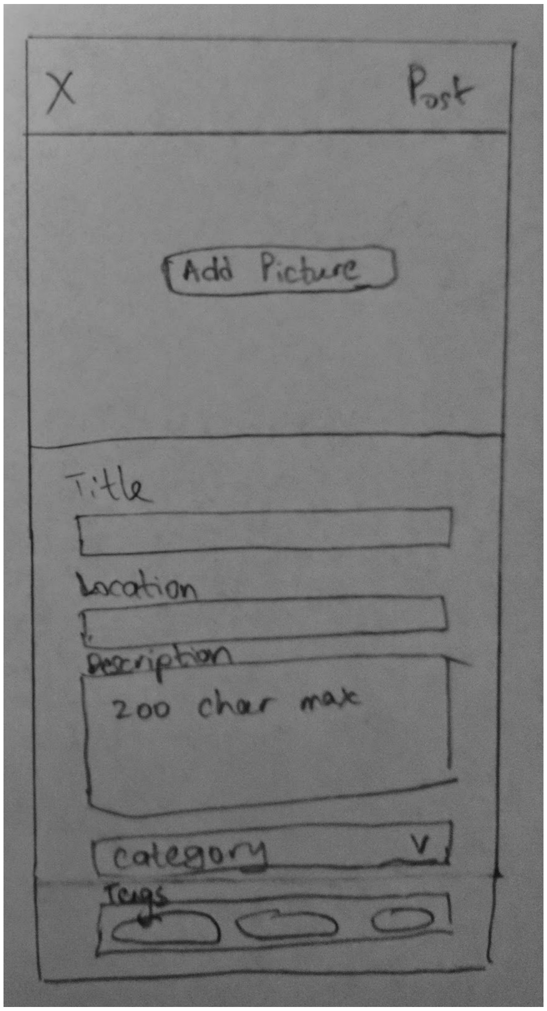
Detail Page



On tapping on an image from the home page, the image will go to Details activity. This activity will show the image, who posted the image, and other details about the image. User will be able to “like” the image or set it as wallpaper on their phone.

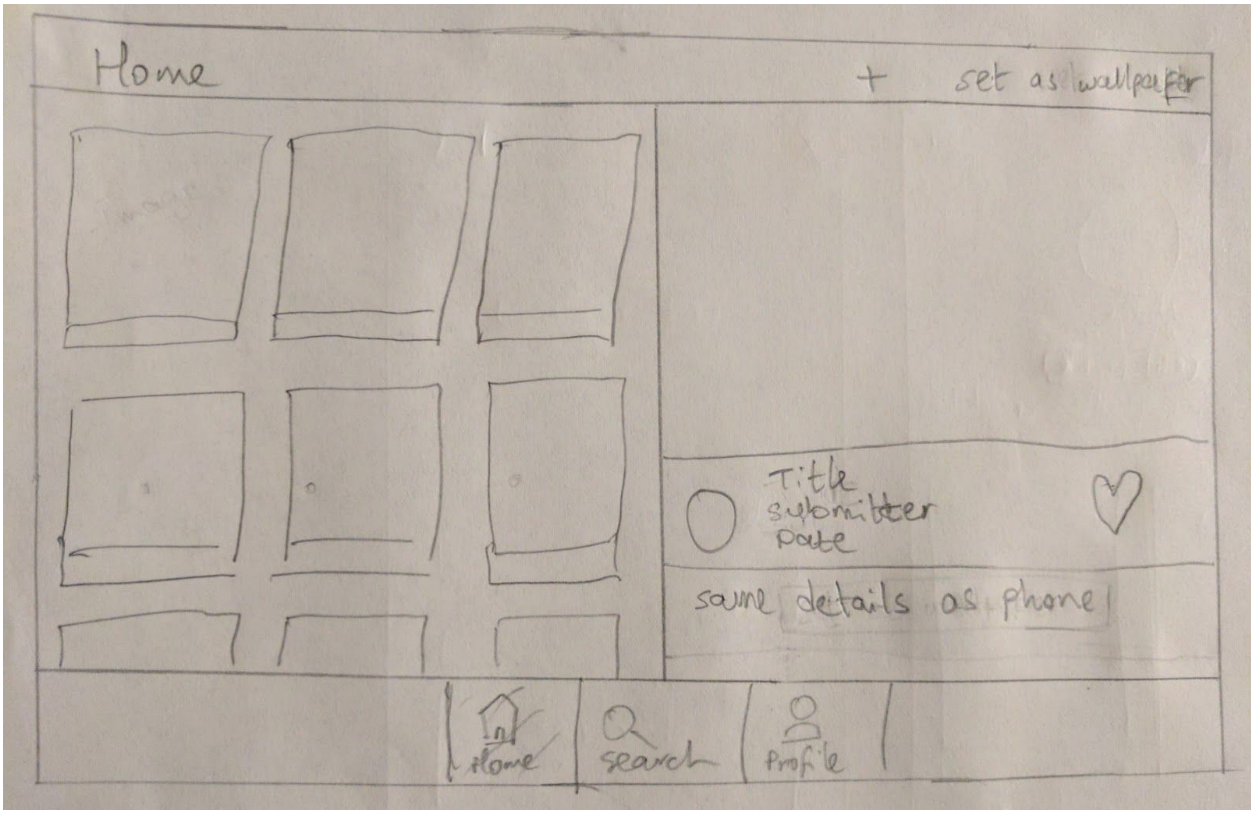
On tapping on the image from the detail page, the image will be shown on full screen. If the user taps on “set as wallpaper”, it will open up a bottomsheet and user will choose the native wallpaper app to manipulate the image.

Add a new image



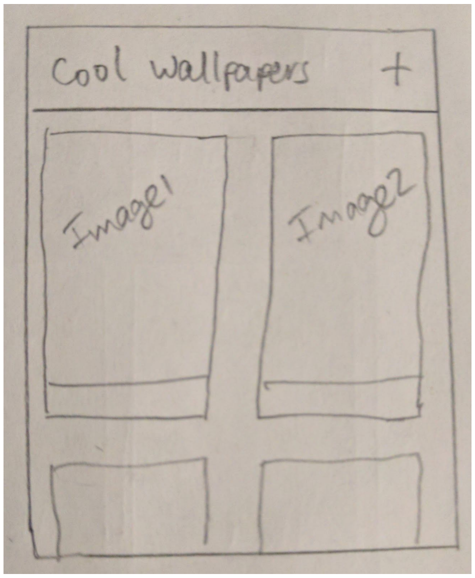
This page will allow the user to upload a new image. This page is accessed by the “+” on the home page.
Uploading a new image contains: image, title, location (optional), Description (optional), category, tags (optional)

UI for tablet



The UI for tablet will split home page, and profile page into two columns (as shown above) if it is landscape. Portait for tablet will be similar as portait for phone. When user selects an image (thumbnail) on the left, the detail fragment will show the details of that thumbnail.

App Widget



The app widget will always show the latest feed from the database. This layout show the minimum height and width of the widget. (It will at least show 2 columns with at least one row, with a bit of an overflow to show that there is more content.)

Key Considerations

How will your app handle data persistence?

I am planning to use Firebase Realtime database for online storage and offline storage.

Describe any corner cases in the UX.

A lot of the interactions have already been discussed in User Interface Mocks. Transitions will be used.

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

Glide - for caching and loading the images on the gridview
Butterknife - for dependency injection
Appcompat library and support library - for material design and backward compatibility

Describe how you will implement Google Play Services.

Google Places API - for searching location (with autocomplete ability)
Firebase Auth and UI - for sign up and login
Firebase Realtime Database - for online database
Firebase Storage - for online image storage
Google Analytics - for monitoring how the app is being used

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

The app will contain three main pages: Home, Search, Profile.

Subtask:

1. Configure login and signup

2. Profile page setup
3. Home page, upload page, search page and app widget setup
4. Tablet layout setup
5. Clean up, refactoring, error handling

Task 2: Configure Login and signup

- Use Firebase UI and Auth to configure this
- Add some data in the database
- Setup the basic UI for all of the views (in xml)

Task 3: Profile page Setup

- Once the Authentication is done, the user will provide the information to setup the profile
- Make sure that the user can edit the page
- Add some random data for the user and check if the image is being shown

Task 4: Home page, upload page, search page and app widget setup

- Setup the three pages, use Firebase Realtime Database, Places API to get location
- Test if the user can add new images and that the image is shown in his feed
- Test if the can search for an image by title, user's name, tags, and category
- Implement Firebase Job Dispatcher to update the offline database and the app widget
- Search will require IntentService for getting data from the online realtime database

Task 5: Tablet Layout setup

- Implement UI for tablet

Task 6: Clean up, refactoring and error handling

- Put everything together
- Make sure that the app does not crash when internet is not available, data is not available
- Clean up if necessary

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**"