

## 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”**

---

### [Description](#)

### [Intended User](#)

### [Features](#)

### [User Interface Mocks](#)

[Screen 1 - Login](#)

[Screen 2 - Profile](#)

[Screen 3 - Following](#)

[Screen 4 - Favorites](#)

[Screen 5 - Details](#)

[Screen 6 - Search](#)

[Screen 6 - Gallery](#)

[Screen 7 - Project](#)

[Screen 8 - Inspiration](#)

[Screen 9 - Widget](#)

### [Key Considerations](#)

[How will your app handle data persistence?](#)

[Describe any corner cases in the UX.](#)

[Describe any libraries you’ll be using and share your reasoning for including them.](#)

[Describe how you will implement Google Play Services.](#)

### [Next Steps: Required Tasks](#)

[Task 1: Project Setup](#)

[Task 2: Determine NoSQL data model for Firebase Database use](#)

[Task 3: Implement Skeleton UI for Each Activity, Fragment and Dialog](#)

[Task 4: Enable Firebase Authentication](#)

[Task 5: Provide Firebase Storage connectivity](#)

[Task 5: Implement the UI for the Following fragment](#)

[Task 6: Implement the UI for the Favorites fragment](#)

[Task 7: Implement the UI for the Search fragment](#)

[Task 8: Implement the UI for the Gallery fragment](#)

[Task 9: Implement the UI for the Details activity](#)

[Task 10: Implement the UI for the Project activity](#)

[Task 11: Implement the UI for the Inspiration activity](#)

[Task 12: Implement the Widget displaying Gallery summary information](#)

**GitHub Username:** `concaveNP`

## Artistry Muse

### Description

Artistry Muse is about sharing your art projects and what you used for the inspiration behind them. Receive feedback from the community about your work with ratings, viewings and how many people have favorited your art. Discover and follow other artists in order to see what they used as the inspiration in their work.

### Intended User

The intended audience is artists who would like to see what inspired the works of other artists. While also showing off their work and what inspired them.

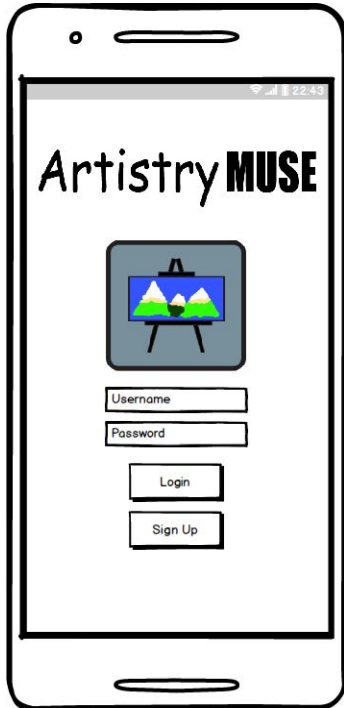
### Features

- Create art projects with documentation explaining them and works of art that were used as inspiration for them
- Takes pictures
- Share art projects
- See ratings others have given your art work
- See the number of times your art has been viewed
- See the number of times your art has been favorited
- Search for other artists
- Follow an artist to see their any new work they create
- Mark artwork as a favorite of yours
- Art projects, favorites, and followers data is saved within the cloud

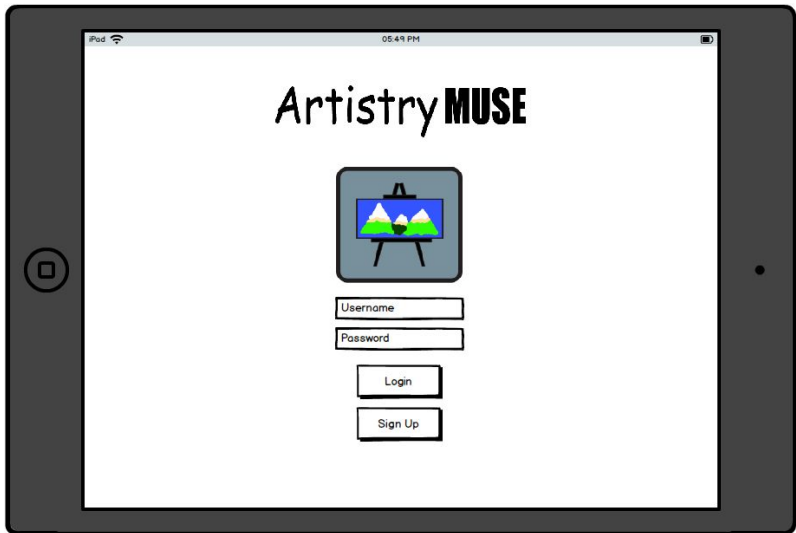
### User Interface Mocks

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 - Login



The mobile app login screen features the 'Artistry MUSE' logo at the top. Below the logo is a square icon depicting a landscape with mountains and a sun on an easel. Underneath the icon are two input fields labeled 'Username' and 'Password', followed by 'Login' and 'Sign Up' buttons.

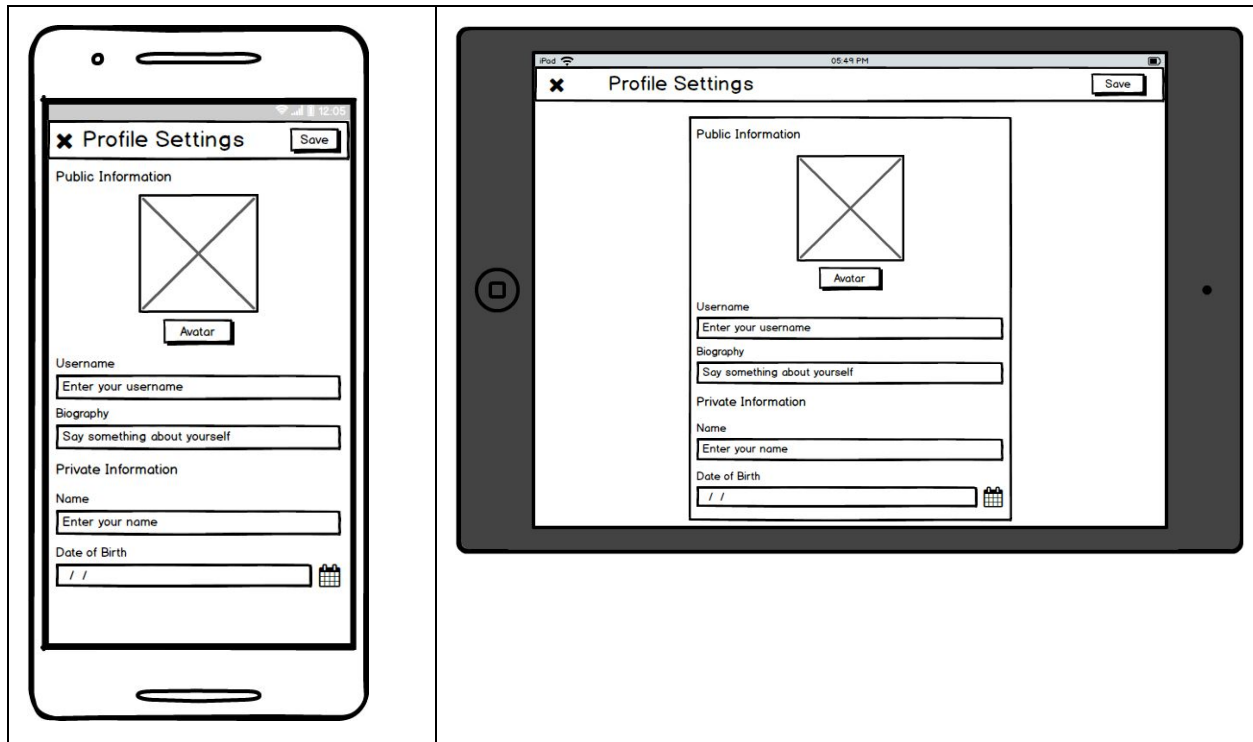


The tablet app login screen displays the 'Artistry MUSE' logo at the top. Below the logo is a square icon depicting a landscape with mountains and a sun on an easel. Underneath the icon are two input fields labeled 'Username' and 'Password', followed by 'Login' and 'Sign Up' buttons. The status bar at the top shows 'iPad', signal strength, and the time '05:49 PM'.

The **Login** screen will allow the user to either login or create a new account via the Google Firebase Authentication services using the FirebaseUI as a complete drop-in auth solution.

(Ref: <https://firebase.google.com/docs/auth/>)

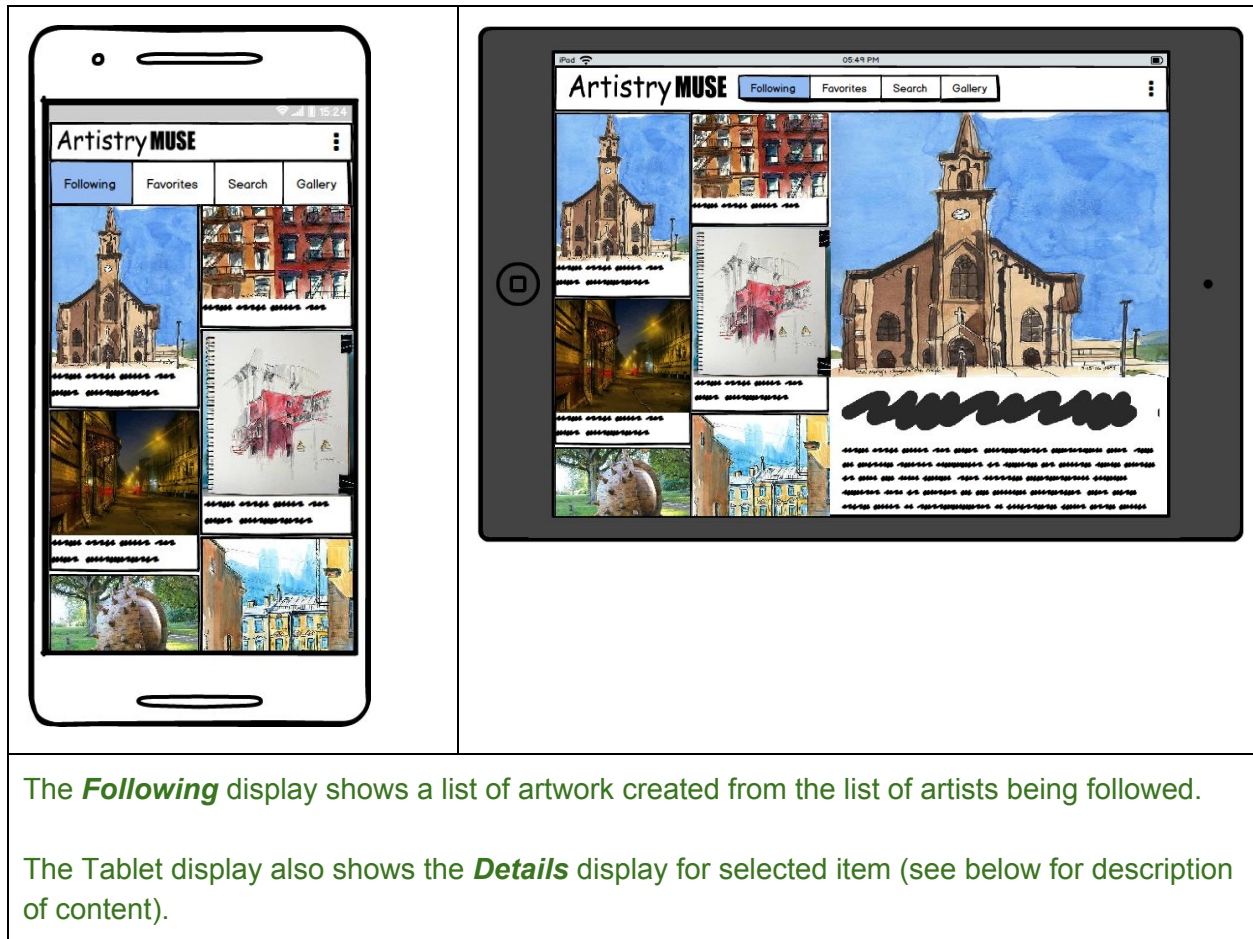
## Screen 2 - Profile



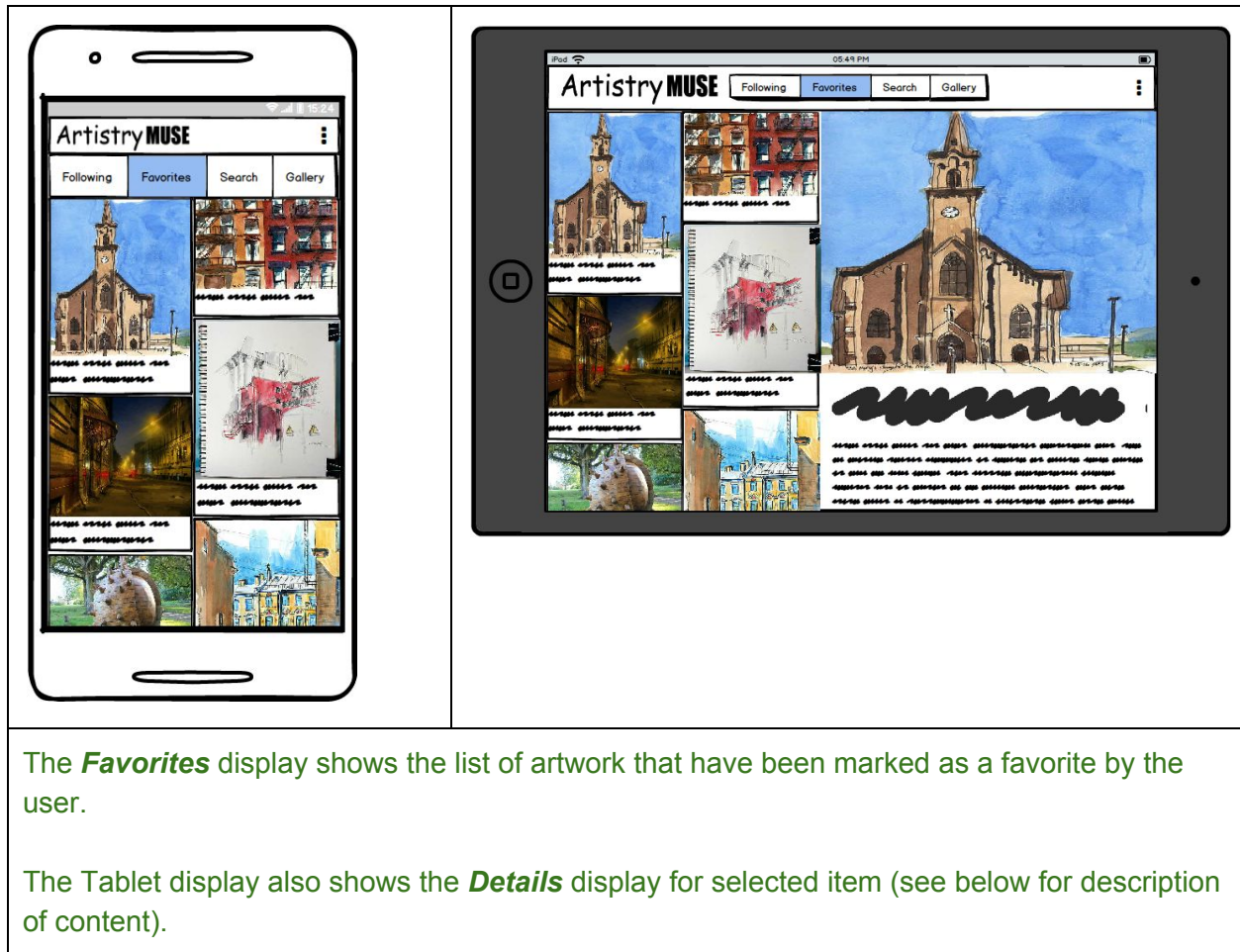
The **Profile** is where the user can manipulate information about themselves that will be both publically viewable and personally private. This display would be brought up in the creation of a new account or from within the **Settings->Profile** menu.

The **Profile** will appear during the creation of a new account. The user must fill out the required information in order for the account to be created. I am planning on using the Firebase Authentication to achieve the account creation and I'm currently unsure about the specific points of data it will require. Password and email should also be apart of account creation.

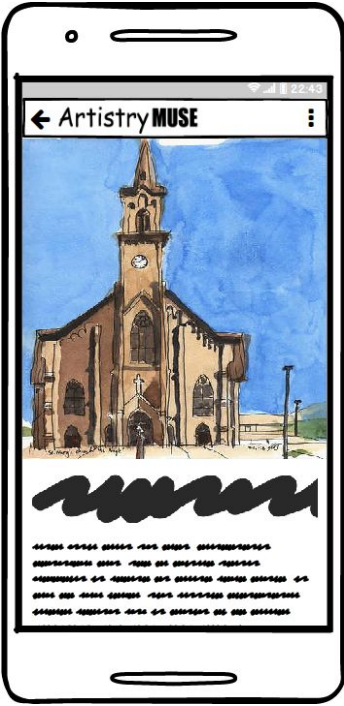
## Screen 3 - Following



## Screen 4 - Favorites

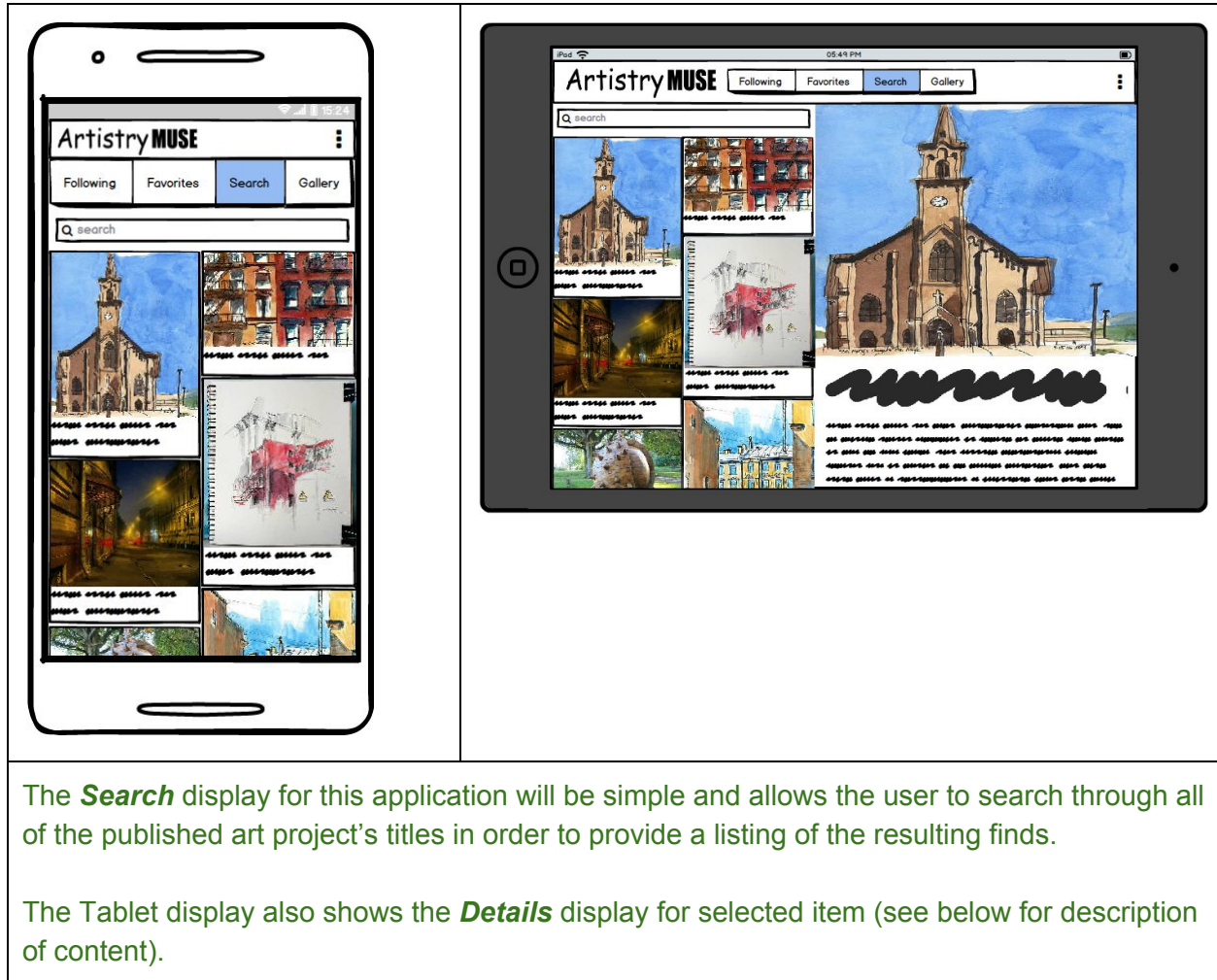


## Screen 5 - Details

	<p>The <b>Details</b> display shows a selected artwork and all of the details associated with it. This includes the inspirations behind it.</p> <p>It also is where the user would have the option of Following the originator, rating it, or marking it as a Favorite.</p> <p>(Not an individual Tablet display)</p>
--	---

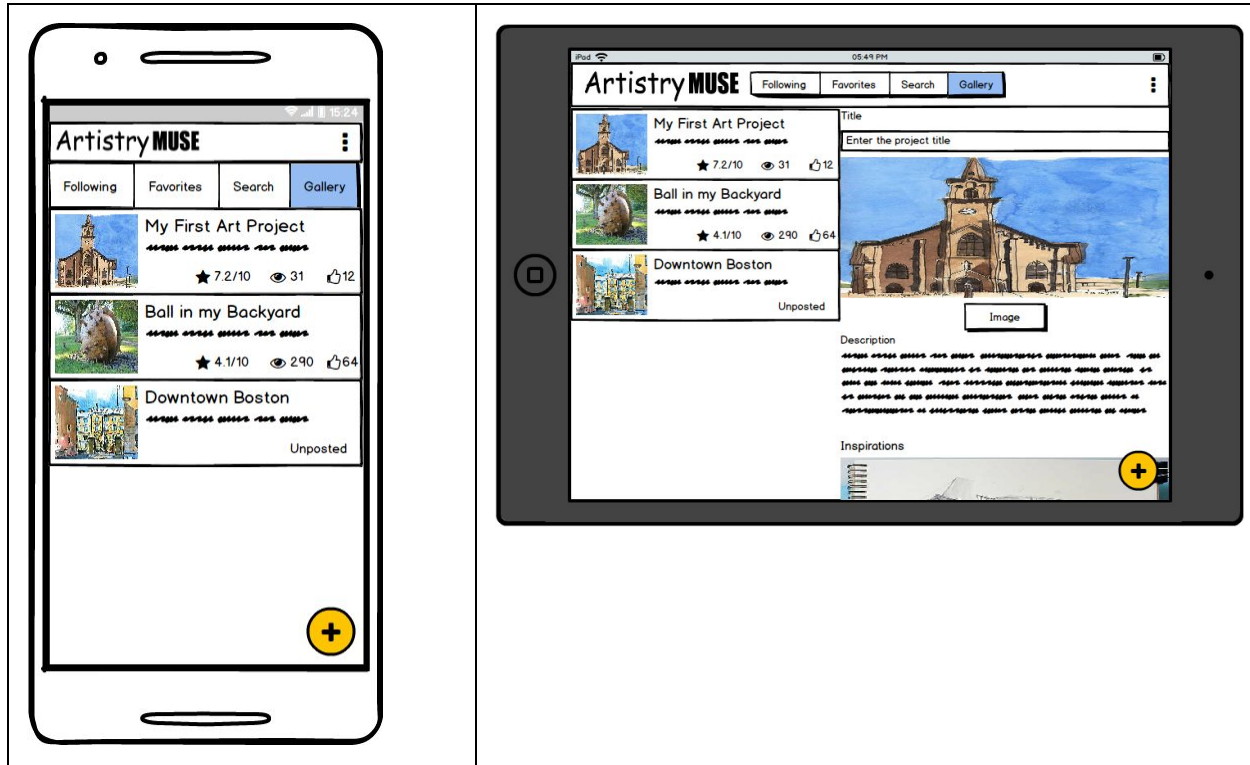


## Screen 6 - Search





## Screen 6 - Gallery



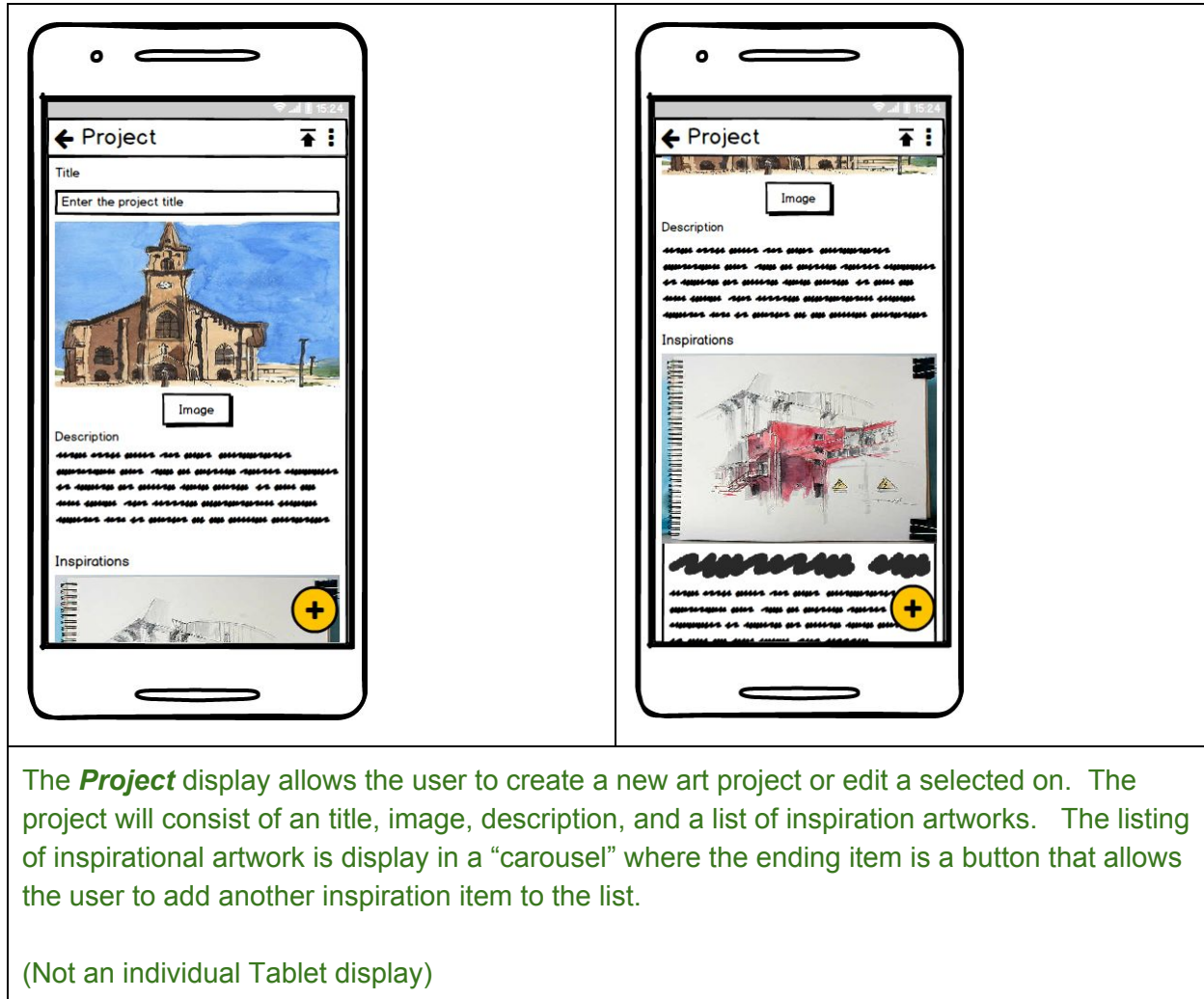
The **Gallery** display shows the list of artwork projects that the user has created. Additionally, it will show the ratings the public has given the artwork, the number of views (**Details** viewing), and the number “Thumbs-Up” (aka Favorited) it has received.

Clicking the FAB brings up the **Project** display in order to create a new art project.

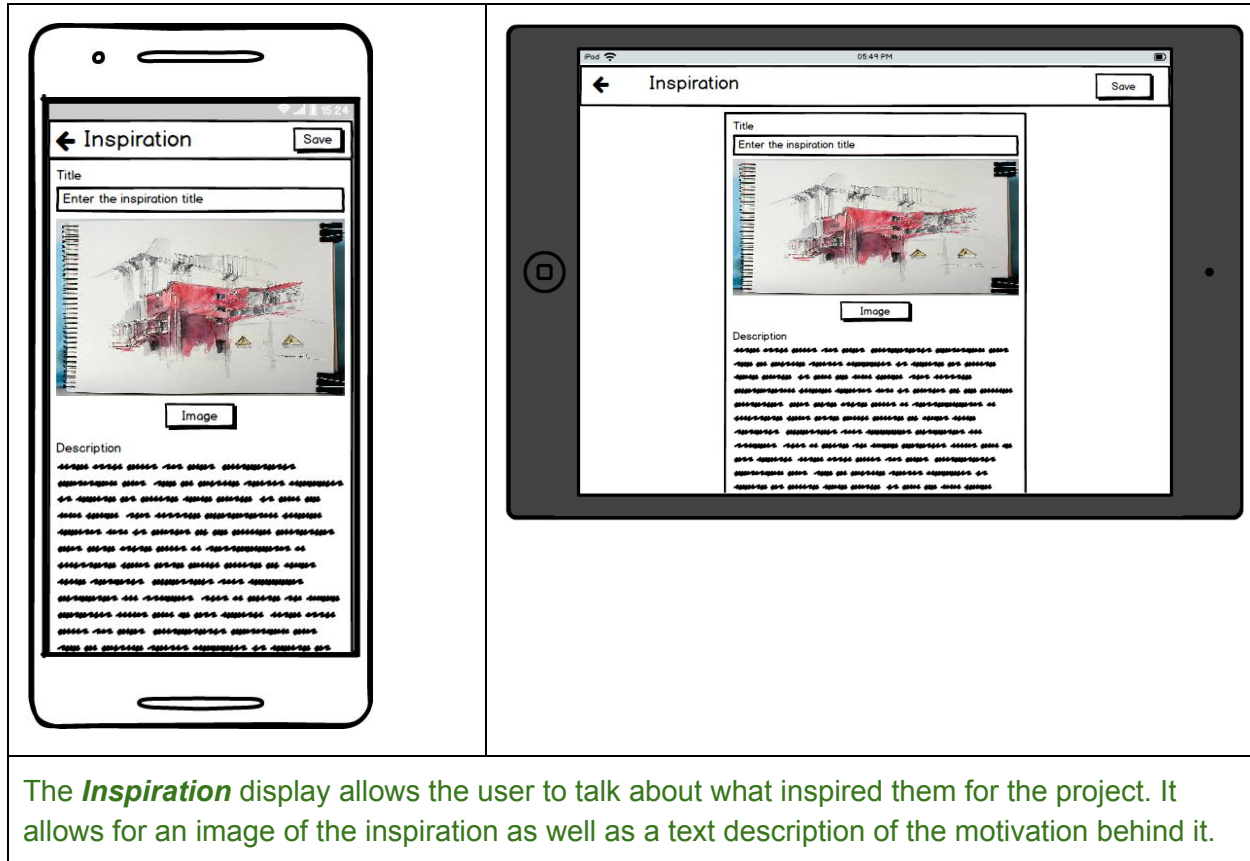
Clicking a project item will also bring up the **Project** display in order to edit the art project content.

The Tablet display also shows the **Project** display as well (see below for description of content).

## Screen 7 - Project

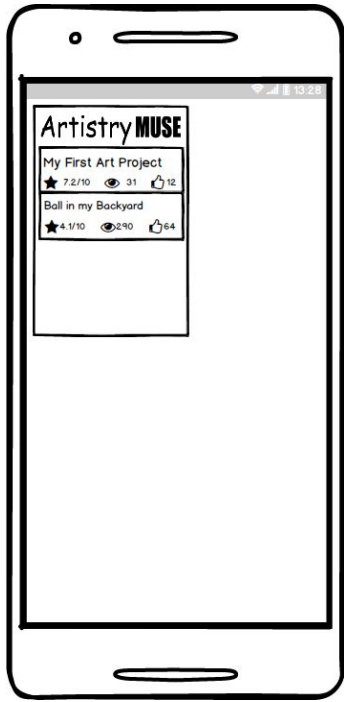


## Screen 8 - Inspiration



The *Inspiration* display allows the user to talk about what inspired them for the project. It allows for an image of the inspiration as well as a text description of the motivation behind it.

## Screen 9 - Widget

	<p>The <b>Widget</b> display show how the user can place a widget on the home screen that summarizes their Gallery.</p> <p>(Same for Tablet)</p>
--	--

## Key Considerations

### How will your app handle data persistence?

Data will be saved using the Firebase Storage and Database services. Heavy leverage will be on the “Offline” capability to sync up data when connectivity becomes available. Both DB (NoSQL) and image data is persisted to local storage and then moved to cloud storage automatically by the library.

### Describe any corner cases in the UX.

- When the user is creating an account and does not complete filling out the profile data, clicking the X, the app will revert back to the Login activity. Clicking the save button will attempt to create the account. Problems in creating the account will be shown to the user in the form of a dialog explaining the problem.
- When the user is creating a new art project and subsequently creates a new Inspiration item the user might decide not to save the documented inspiration at all. This requires clicking the X button and acknowledging the dialog.

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

The Picasso library will be used for image loading and caching. There is also the libraries that Firebase services will need as well.

### Describe how you will implement Google Play Services.

The design will include the use of three of the Google Firebase services: Authentication, Realtime Database and Storage. The Firebase Authentication will be used to establish an individual account for the user based on their preferred method of credentials verification (for time constraints I think this will be limited to just Google accounts). The Firebase Realtime Database will be used to hold the information regarding to the user and their documented art projects. Finally, the Firebase Storage will be used to hold the user’s photos.

## 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 project will start off by first setting up a base application by generating code from the IDE and getting the necessary libraries in place. Next, will be the Firebase account setup and association to the application.

- Create the base generated application as a starting place
- Setup gradle for using the Picasso library
- Setup gradle for using the Firebase libraries
- Create the keystore and key for signing the application
- Create the Firebase account and in the Firebase Console and set the application key used for signing the application. From the console, enable the following services:
  - Enable Authentication
  - Enable Database
  - Enable Storage

## Task 2: Determine NoSQL data model for Firebase Database use

- Determine the data model to be used
  - User Model
  - Art Project Model
- Builds the Java classes that will encapsulate the data (POJOs)
- Place in starter data in order to test app with (several artists with several art projects each)

## Task 3: Implement Skeleton UI for Each Activity, Fragment and Dialog

Create the skeleton UI framework by stubbing out all of the displays:

- Build UI for the Login activity
- Build UI for the Profile full-screen dialog
- Build UI for the Main activity
  - Build UI for the Following fragment
  - Build UI for the Favorites fragment
  - Build UI for the Gallery fragment
- Build UI for the Details activity
- Build UI for the Project activity
- Build UI for the Inspiration activity
- Build UI for the Summary widget

## Task 4: Enable Firebase Authentication

Authentication needs to be enabled in order for the Login/Splash activity and the Profile dialog to be filled out and working correctly. This design will make use the Firebase Authentication library's provided UIs (FirebaseUI).

- Modify the current placeholder activity to extend the provided FirebaseUI activity for login purposes
- Modify the provided layout to achieve app's look & feel for the app (i.e. icons, background coloring, images, etc.)
- Create layout views within the Profile dialog to support the Firebase Authentication needs of Username, email, password, etc.
- Link the data within the views to the Firebase API calls and then link with the service
- Test account for verification
- Test for the closing of the Profile dialog when either logging in or creating a new account in order to assure that no login occurs

## Task 5: Provide Firebase Storage connectivity

Interaction with the Firebase Storage service from the different activities, fragments and dialogs will require some classes to be place in order to send/receive data. I'm using the FirebaseQuickstarts examples as a reference to the needed classes.

(<https://github.com/firebase/quickstart-android>)

- Storage interaction will be done through the use of a services in order to manage the multiple download/upload requests:
  - Implement the base service
  - Implement the download service
  - Implement the upload service

## Task 5: Implement the UI for the Following fragment

The Followers fragment shows a listing of all the artists the user has chosen to follow.

- Create the fragment layout as specified within the mockup
- Make use of a FirebaseRecyclerAdapter given a query that makes use of the user's chosen followers
- Images will create download requests of the Storage service and display them via picasso
- Link presses of displayed items will start a corresponding Details activity



## Task 6: Implement the UI for the Favorites fragment

The Favorites fragment shows a listing of all the art projects the user has marked as a favorite.

- Create the fragment layout as specified within the mockup
- Make use of a `FirestoreRecyclerAdapter` given a query that makes use of the user's chosen favorites
- Images will create download requests of the Storage service and display them via Picasso
- Link presses of displayed items will start a corresponding Details activity

## Task 7: Implement the UI for the Search fragment

The Search fragment for this application will be kept simple and allows the user to search through all of the published art projects titles.

- Create the fragment layout as specified within the mockup
- Make use of a `FirestoreRecyclerAdapter` given a query that will take the search string entered by the user and use it for locating published art projects from their title
- Images will create download requests of the Storage service and display them via Picasso
- Link presses of displayed items will start a corresponding Details activity

## Task 8: Implement the UI for the Gallery fragment

This fragment will show off the user's art projects in a listing of project summary items. It gives quick and simple feedback as to whether their projects are being viewed, how they are rated, and how many people favorited their work. Selecting an item allows the user to change the information associated with the art project.

- Create the fragment layout as specified within the mockup
- Make use of a `FirestoreRecyclerAdapter` given a query that makes use of the user's gallery of art projects
- Images will create download requests of the Storage service and display them via Picasso
- Art projects that are not yet published will show an "unpublished" string versus the summary details that would be shown if the work was published
- Link presses of displayed items to a creation of a corresponding Project activity that allows the user to change information about the art project

## Task 9: Implement the UI for the Details activity

The details activity allows the user to interact with other people's art projects. This can be done by rating, favoriting, or following the artist.

- Create the activity layout as specified within the mockup
- Button presses of the **follow** icon will add the art project's creator to the user's list of followers
- Button presses of the **rating** icon will bring up a simple dialog allowing the user to rate the work of art
- Button presses of the **favorites** ("thumbs up") icon will add this art project to the user's list of favorited items
- Increment the view count associated with the art project

## Task 10: Implement the UI for the Project activity

The Project activity allows the creator of an art project to put in information about the work. This includes information such as an image, a description of the work, and a listing of other things that the creator used as an inspiration to their work.

- Create the activity layout as specified within the mockup
- Enable the publish button when the user trips a dirty flag
- Enable the FAB button to start an Inspiration activity for a new item
- Clicking of an existing inspiration will start an Inspiration activity for that inspiration's data
- A return from an Inspiration activity will, if the data model is not null, replace the data model for an item if it existed already and create a new inspiration item if it did not already exist
- When the user clicks the publish button the current Art Project model data will be made available to other users searches of art projects
- When the user clicks the Up or Back button and the dirty flag has been set the data model for the art project will saved

## Task 11: Implement the UI for the Inspiration activity

- Create the activity layout as specified within the mockup
- Enable the Save button if the user trips a dirty flag
- When the user clicks the save button create a new Inspiration Model of data will be created and populated (POJO defined earlier) and returned back to the activity's creator

- When the user clicks the Up or Back button and the dirty flag has been set and yet the Inspiration has not been saved, a dialog will appear asking to user to verify the loss of data.

## Task 12: Implement the Widget displaying Gallery summary information

- Create the widget layout as specified within the mockup
- This will be based upon how the SuperDuo project implemented the following widget supportive classes
  - GalleryAppWidgetConfigureActivity
  - GalleryAppWidgetProvider
  - GalleryAppWidgetService

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**"