How to Use this Template

1. Create a new document, and copy and paste the text from this template into your new document [Select All → Copy → Paste into new document]
2. Name your document file: "**Capstone_Stage1**"
3. Replace the text in green

GitHub Username: buildbro

ClipAir

Description

ClipAir helps you extend the clipboard on your device. By default when you copy text to the clipboard on your device, the text is overwritten the next time you perform another copy operation. With ClipAir, you can view an extended history of your clipboard and share copied text across devices.

Intended User

This is an app for any mobile user than perform text copying operation often on their phones. It can be very helpful to students, bloggers and tech enthusiasts.

Features

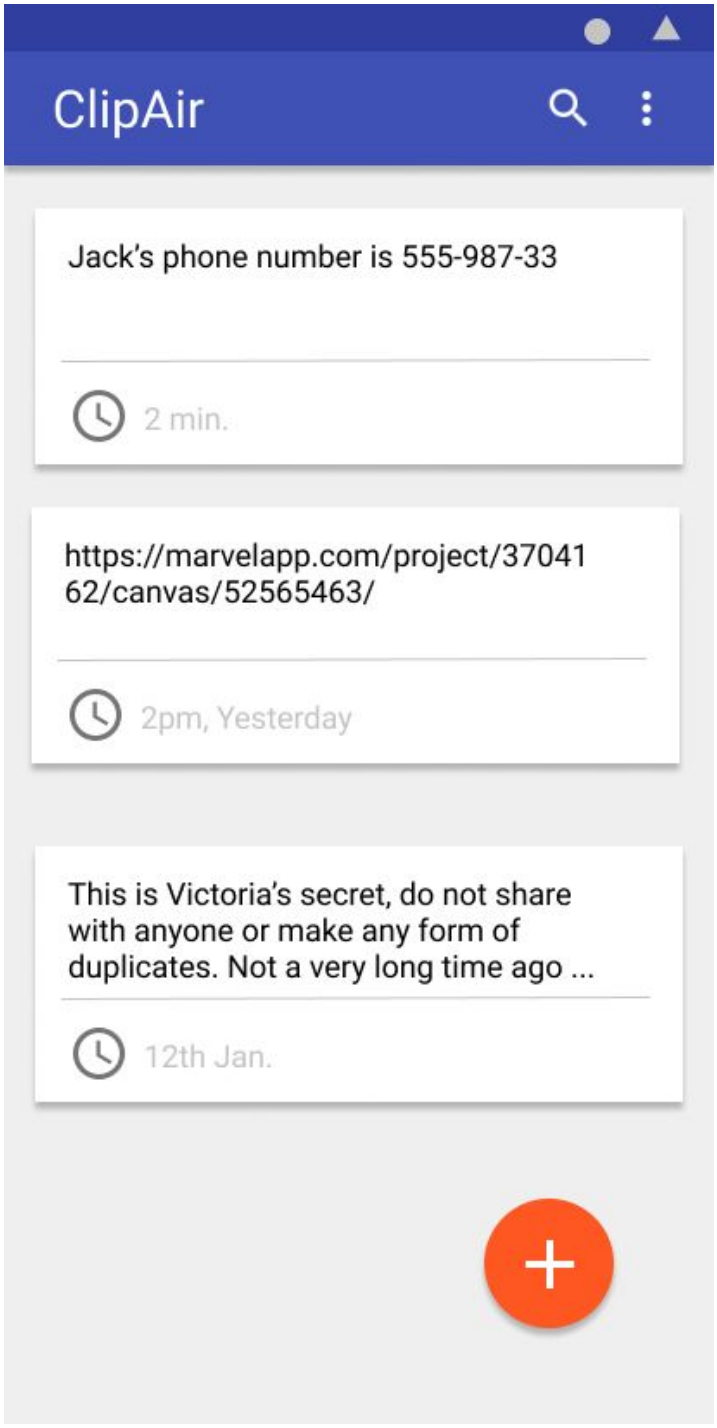
List the main features of your app. For example:

- Saves copied (clipboard) texts locally
- Saves copied texts on the cloud so that you can access them even when you change device
- Lets you share your clipboard with other devices and users

User Interface Mocks

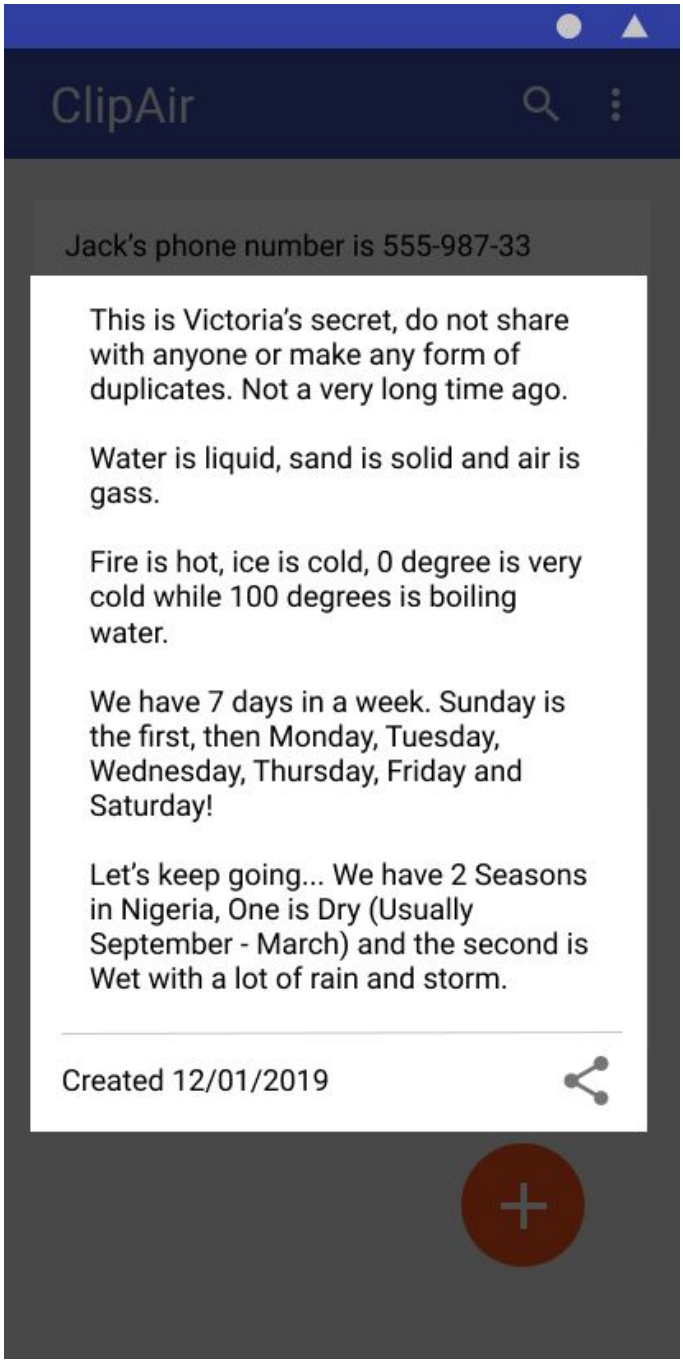
These can be created by hand (take a photo of your drawings and insert them in this flow), or using a program like Google Drawings, www.ninjamock.com, Paper by 53, Photoshop or Balsamiq.

Screen 1 (Main Activity)



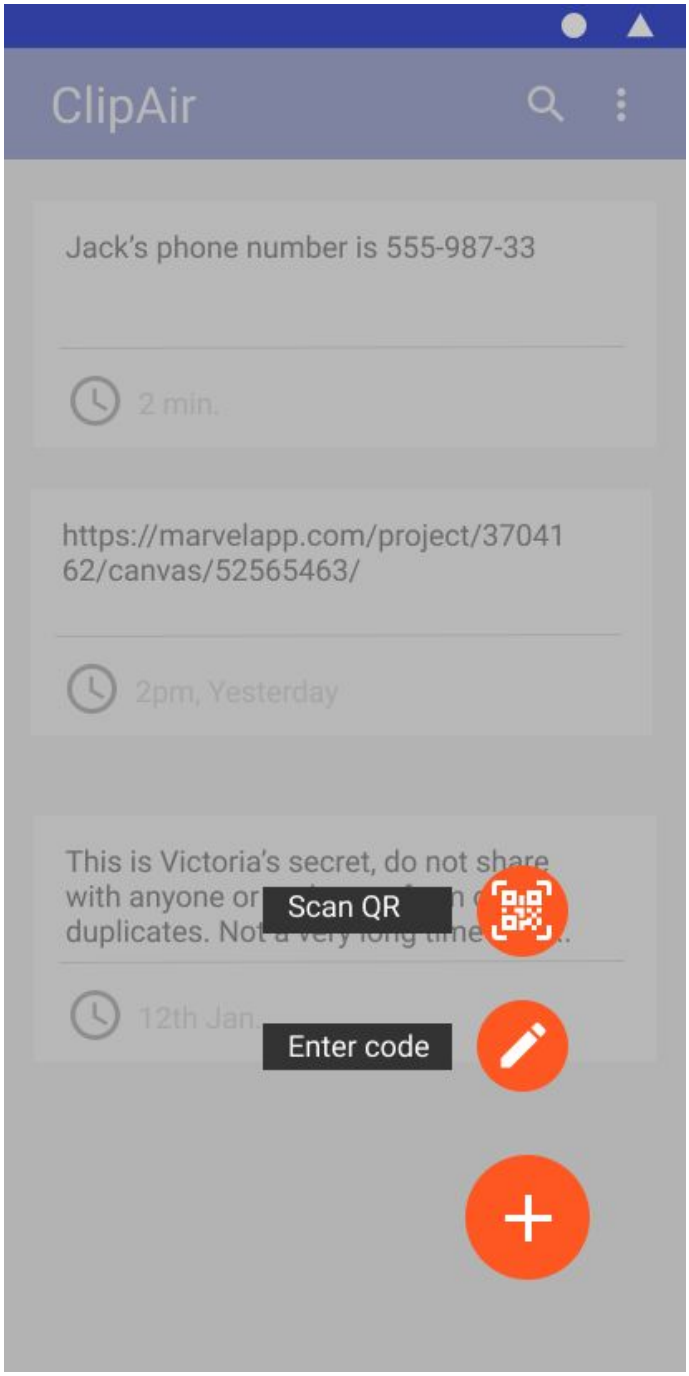
Replace the above image with your own mock [click on the above image, then navigate to Insert → Image…]
This is the first screen a user will see on starting the app. It displays clipboard history in a list showing a preview of the text content and "time created". The screen also shows a FAB, a search bar and overflow icon.

Screen 2 (Details Dialog)



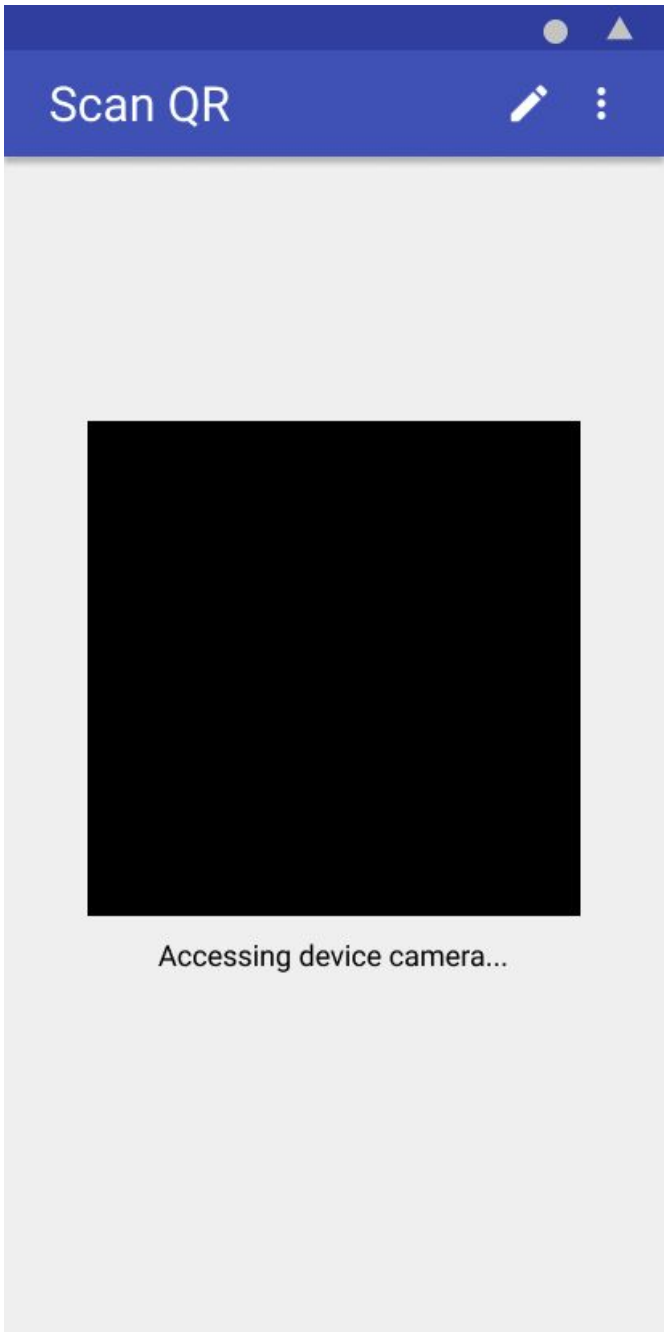
Replace the above image with your own mock [click on the above image, then navigate to Insert → Image…]
Tapping a list item from the previous (Main Activity list) will start this dialog. The dialog shows full details for the selected history item. It also contains a share button that lets you share the text content.

Screen 3 (FAB expanded)



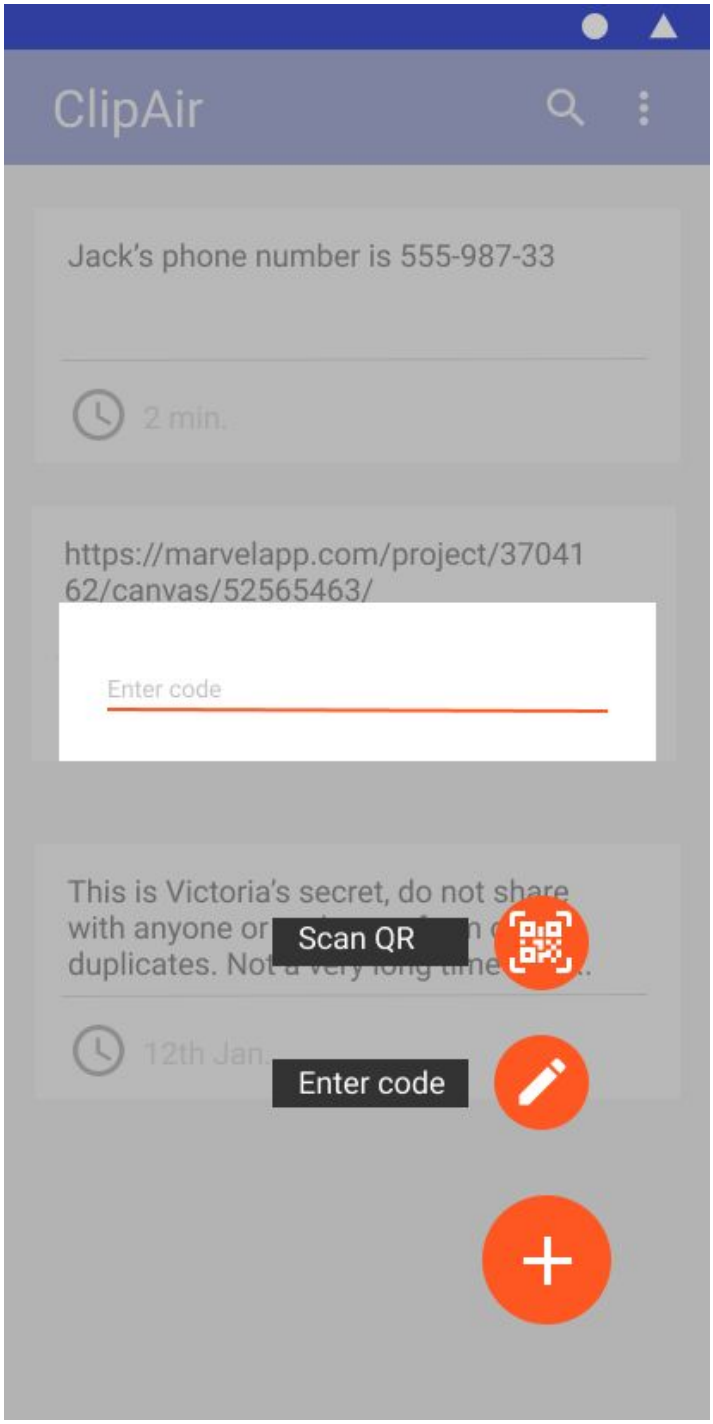
Replace the above image with your own mock [click on the above image, then navigate to Insert → Image…]
Tapping the FAB icon + will expand the fab and show 2 extra icons. The first one lets you scan QR code to receive new clipboard item from another device, while the second allow you to enter a special pairing code to receive items from another device.

Screen 4 (Scan Code Activity)



Replace the above image with your own mock [click on the above image, then navigate to Insert → Image…]
Tapping on the Scan QR FAB will start this activity. It requires access to camera to complete its job of scanning to receive a new clipboard item from another device. On the toolbar there a button to switch to “enter code mode” instead.

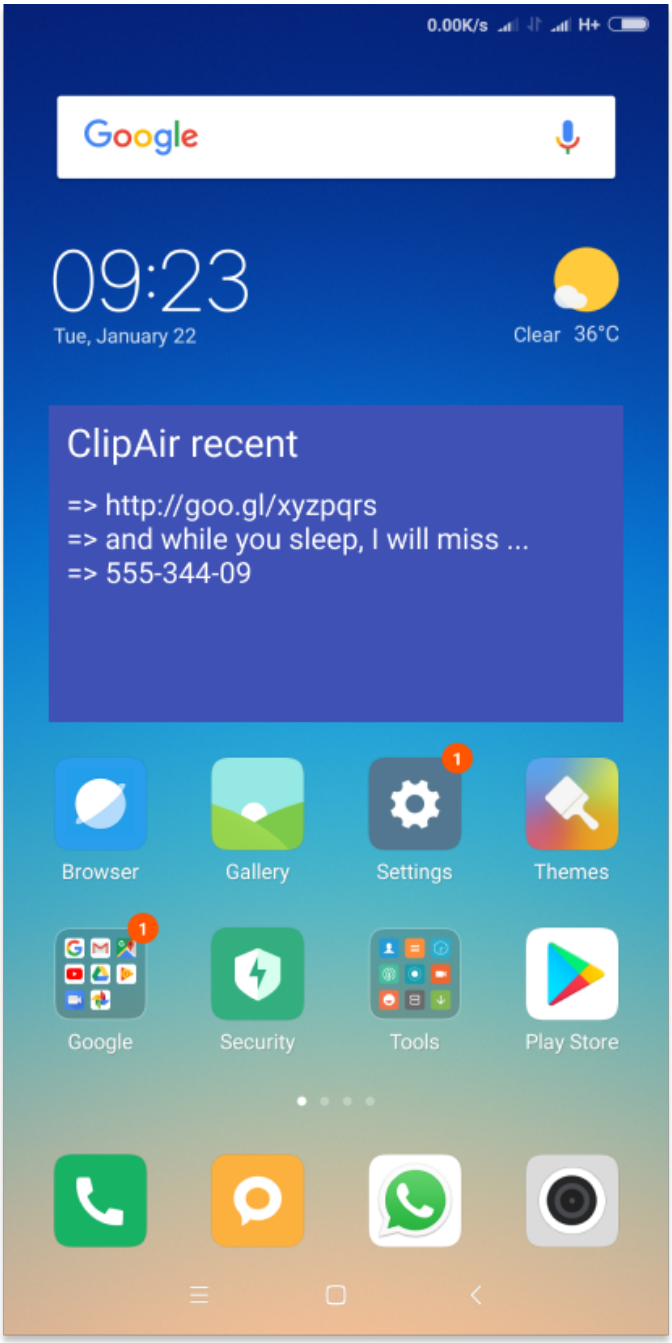
Screen 5 (Enter Code Dialog)



Replace the above image with your own mock [click on the above image, then navigate to Insert → Image…]

This Dialog provides an alternative way to pair with another device and receive new clipboard item with using the QR scanner feature.

Screen 5 (Home Screen widget)



Replace the above image with your own mock [click on the above image, then navigate to Insert → Image…]
ClipAir Recent widget displays some of the most recent items from the clipboard history on the device homescreen.

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 use firebase realtime database to store clipboard items.

Describe any edge or corner cases in the UX.

For example, how does the user return to a Now Playing screen in a media player if they hit the back button?

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

- Dm77 BarcodeScanner (**me.dm7.barcodescanner:zxing:1.9.8)**, for scanning QR code
- Google play service, to enable google signing and other google services.
- Firebase-database, to allow access to firebase database.
- Com.getbase:floatingactionbutton:1.10.1, provides an expandable FAB with built-in animation and background blur effect.

Describe how you will implement Google Play Services or other external services.

Users will need to sign-in with google in order to access clipboard item sharing feature of the app. So that the app can easily sign their clipboard history matching it to their google account.

Next Steps: Required Tasks

This is the section where you can take the main features of your app (declared above) and break them down into tangible technical tasks that you can complete one at a time until you have a finished app.

Task 1: Project Setup

- Create new project on android studio
- Include all required libraries in app level build.gradle file

Task 2: Implement UI for Each Activity and Fragment

- Build UI for MainActivity
- Build UI for Details Dialog
- Build UI for QR scanner Activity

Task 3: Setup Firebase

- Create new firebase project in firebase console
- Include the firebase dependencies in project
- Create firebase database
- Setup FCM notification

Task 4: User Sign-in

- Enable Google Sign-in in firebase console
- Connect app to firebase
- Integrate Google Sign-in into app

Task 5: Database Setup

- Design database structure for an entity (table) with id, main_content, date_added, created_by, sharing_status properties.
- Setup a firebase database with the entity and properties stated earlier

Task 6: Setup Firebase database

- Configure firebase database rules
- Setup read and write access to database
- Implement code to read and write database

Task 7: Push Notification

- Setup FCM notification

Task 8: Build App Widget

- Implement a widget to display 3-5 recent items from clipboard history on homescreen

Task 9: Testing

- Create tests for app features to make sure every feature works as expected

Add as many tasks as you need to complete your app.

Submission Instructions

- After you've completed all the sections, download this document as a PDF [File → Download as PDF]
    - Make sure the PDF is named "**Capstone_Stage1.pdf**"
- Submit the PDF as a zip or in a GitHub project repo using the project submission portal

If using GitHub:

- Create a new GitHub repo for the capstone. Name it "**Capstone Project**"
- Add this document to your repo. Make sure it's named "**Capstone_Stage1.pdf**"