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: oluwadara-abijo

Moffin Options

Description

Pre order healthy muffins from Moffin Options.

Moffin Options allows you choose your muffins, variants and toppings, as well as make payment so they can be delivered to you.

Intended User

Individuals, Families

Features

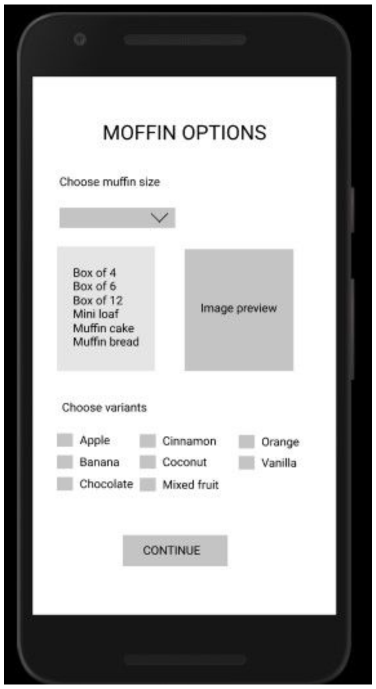
List the main features of your app. For example:

- Choose muffin sizes
- Choose muffin variants
- Choose muffin toppings
- Pre order your muffins
- Choose delivery address
- Pay using secure online payment methods
- Receive notifications about new variants

User Interface Mocks

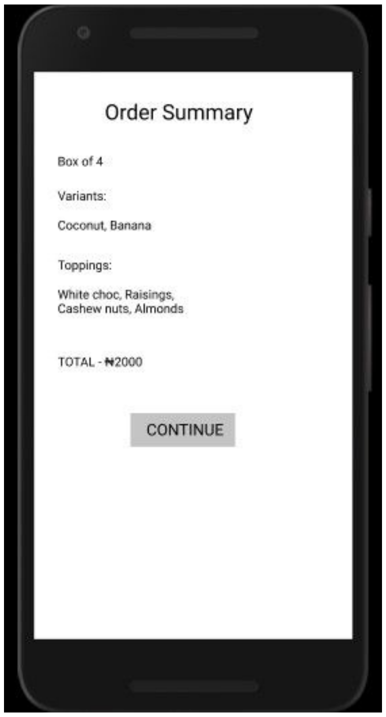
These can be created by hand (take a photo of your drawings and insert them in this flow), or using a program like Google Drawings, www.ninjamock.com, Paper by 53, Photoshop or Balsamiq.

Screen 1



Users select the muffin size, variants and toppings.

Screen 2



Order Summary

Box of 4

Variants:

Coconut, Banana

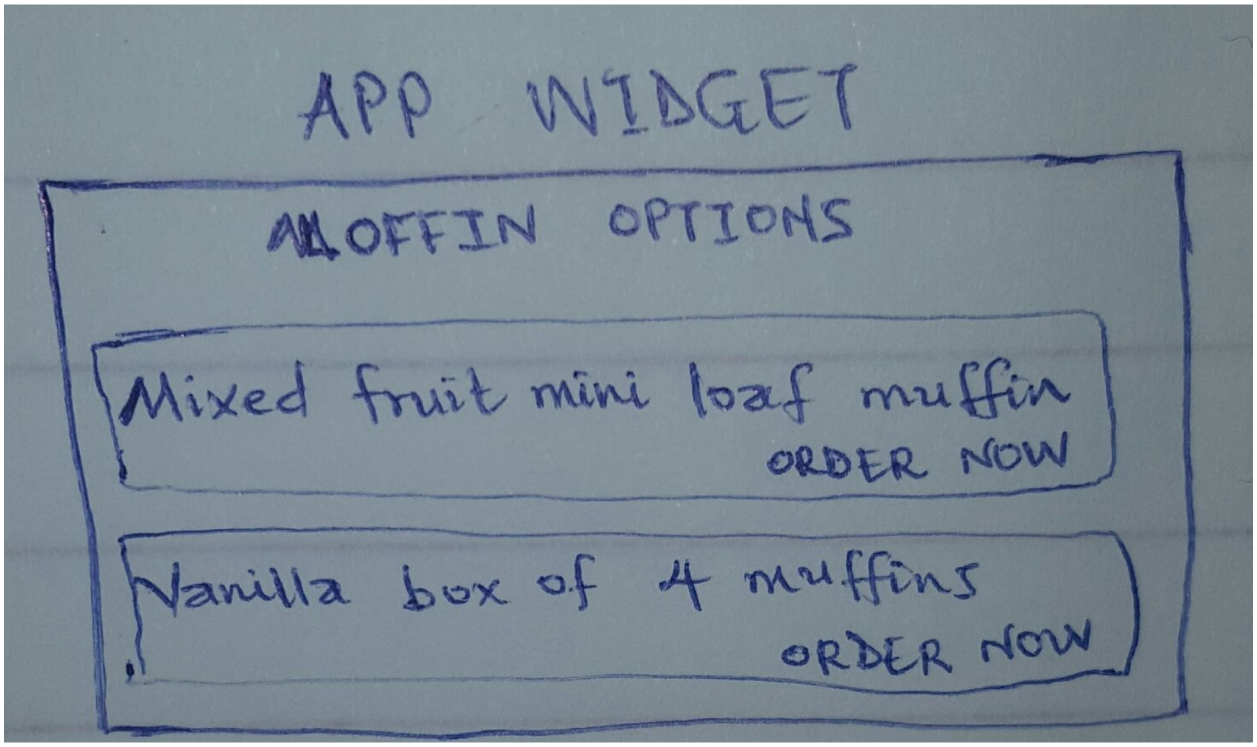
Toppings:

White choc, Raisings,
Cashew nuts, Almonds

TOTAL - ₦2000

Users get the cost of their order and proceed to payment.

App Widget



The widget displays top picks, and users can place their orders from the widget.

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

Key Considerations

- The app will be written solely in the Java programming language
- App will utilize stable release versions of all libraries, Gradle, and Android Studio.
- App will provide a widget to provide relevant information to the user on the home screen.
- App will include support for accessibility by using content descriptions.
- All strings will be stored in strings.xml
- App will support RTL layout switching in all layouts.
- App will use an Asynctask to make on-demand network requests.
- App will use an IntentService to update the app widget.

How will your app handle data persistence?

Data persistence will be handled using Room. Livedata and ViewModel will be used to avoid making unnecessary calls to the database.

Describe any edge or corner cases in the UX.

In case of no internet connection, a message will be displayed through a Snackbar, and an empty state where applicable.

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

Picasso will handle the loading and caching of images.
Retrofit will handle network calls.
Payment will be integrated using Paystack library.

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

App will implement Firebase Auth for Google sign in.
App will implement Google Analytics to access reports and usage statistics.

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

- Configure Retrofit library
- Configure Paystack library
- Configure Firebase Authentication
- Configure Google Analytics

Task 2: Implement UI for Each Activity and Fragment

- Build UI for order fragment
- Build UI for order summary fragment
- Build UI for payment fragment
- Build UI for login fragment
- Build UI for home screen widget

Task 3: Implement Google Services

- Implement Firebase Authentication for sign in
- Implement Google Analytics
-

Task 4: Define Business Logic

- Write business logic
- Define methods for calculating prices

Task 5: Integrate Payment

- Integrate payment
- Handle error cases

Task 6: Home screen widget

- Implement home screen widget

Task 7: Write Tests

- Write Espresso tests

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**”