

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**
-

[Description](#)

[Intended User](#)

[Features](#)

[User Interface Mocks](#)

[Screen 1](#)

[Screen 2](#)

[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: Implement UI for Each Activity and Fragment](#)

[Task 3: Your Next Task](#)

[Task 4: Your Next Task](#)

[Task 5: Your Next Task](#)

GitHub Username: mohammedgeek

Sticky Notes

Description

Sticky Notes helps you to stay organized , focused on your goals , save your memos and important contents .

Intended User

Everyone

Features

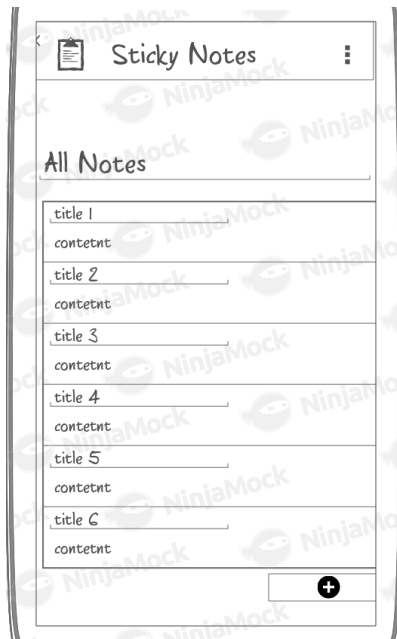
List the main features of your app. For example:

- Save notes
- Update and modify notes
- Delete Notes

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



The first screen will show all notes , the user can add notes .

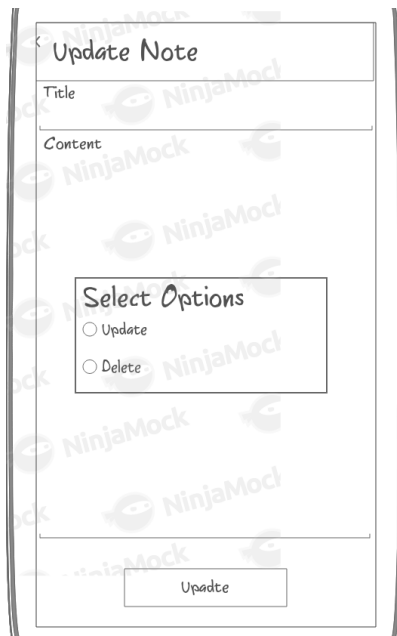
Screen 2



A mobile app screen titled "Add Note". It features a back arrow icon on the top left. Below the title, there is a "Title" label followed by a text input field. Below that is a "Text Field" label followed by a larger text input area. At the bottom of the screen is a button with a floppy disk icon and the text "save". The entire screen is overlaid with a "NinjaMock" watermark.

This screen the user can set a title for the note and type content then save it .

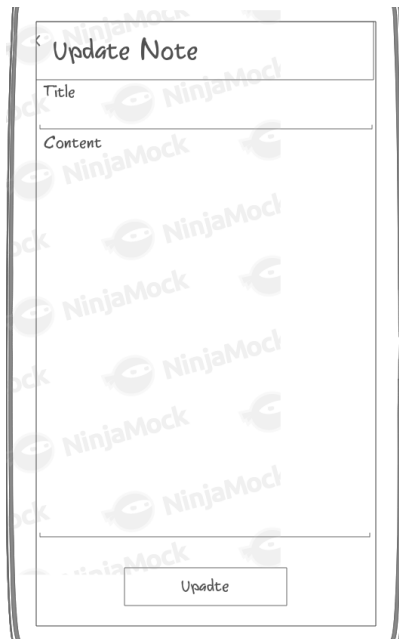
Screen 3



A mobile app screen titled "Update Note". It features a back arrow icon on the top left. Below the title, there is a "Title" label followed by a text input field. Below that is a "Content" label followed by a larger text input area. In the center of the screen is a dialog box titled "Select Options" containing two radio button options: "Update" and "Delete". At the bottom of the screen is a button with the text "Upadte" (misspelled). The entire screen is overlaid with a "NinjaMock" watermark.

In this screen the dialogue make choices for the user to update or delete the note .

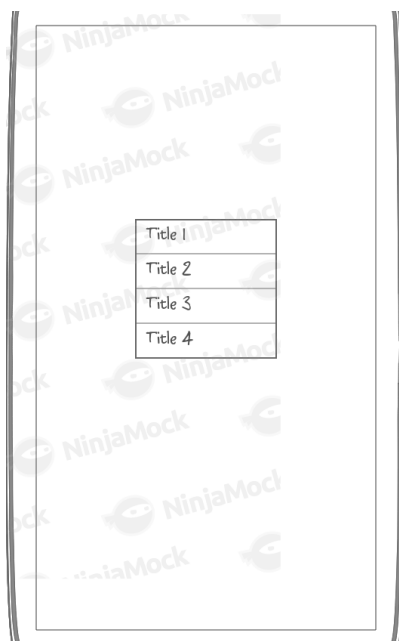
Screen 4



A mobile app screen titled "Update Note". It features a back arrow icon in the top left corner. Below the title, there are two input fields: "Title" and "Content". The "Content" field is a larger text area. At the bottom of the screen, there is a button labeled "Update".

This screen the user can modify a saved note .

Screen 5



A mobile app screen showing an app widget. The widget is a small rectangular box containing four lines of text, each representing a note title: "Title 1", "Title 2", "Title 3", and "Title 4".

App widget will view the title of each note in home screen.

Key Considerations

How will your app handle data persistence?

I will use Room to handle the database.

Describe any edge or corner cases in the UX.

User will add note locally by the add button and if he clicked on a note will choose from dialogue to update or delete the note app will include a content description for views to support accessibility .

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

Butterknife to Bind Android views and callbacks to fields and methods .

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

App will include Firebase Jobdispatcher to update widget schedule and Google Analytics and Admob will implemented in project files .

Versions of libraries , Gradle and Android Studio.

Android Studio v: 3.5.0

Gradle v: 3.5.0

Room v 1.0.0

Butterknife v 10.2.0

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

Write out the steps you will take to setup and/or configure this project. See previous implementation guides for an example.

You may want to list the subtasks. For example:

- Add Material design dependencies.
- Add butterknife and Firebase dependencies.
- Add Google Analytics and Admob dependencies.
- Project implemented with java .

If it helps, imagine you are describing these tasks to a friend who wants to follow along and build this app with you.

Task 2: Implement UI for Each Activity and Fragment

- Build UI for all screens provided .

Task 3: Implement Database

- Define the entities needed for database table .
- Create the database table and its instance.
- Implement DAO .
- Implement Excuters .

Task 4: Implement Widget

- When note is added the widget will updated in home screen .
- Update is done in background using intent service .

Task 5: Handle Firebase jobdispatcher

- Make a schedule to update widget .

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**"