# Milknote - speak your notes

**GitHub Username**: willwallis

# Milknote

## Description

Milknote brings note taking into the 21st Century. Want to remind yourself to pick up the milk on the way home or to put in your order for a Telsa 3, use Milknote. With a tap of a button Milknote transcribes your spoken notes that can be retrieved at any time along with the time and place you took the note to help jog your memory.

## Intended User

The intended user is busy people who have regular ideas that need to be captured for later actioning. Examples include busy professionals as they commute, parents with their hands full with the children or students having a moment of genius between beers.

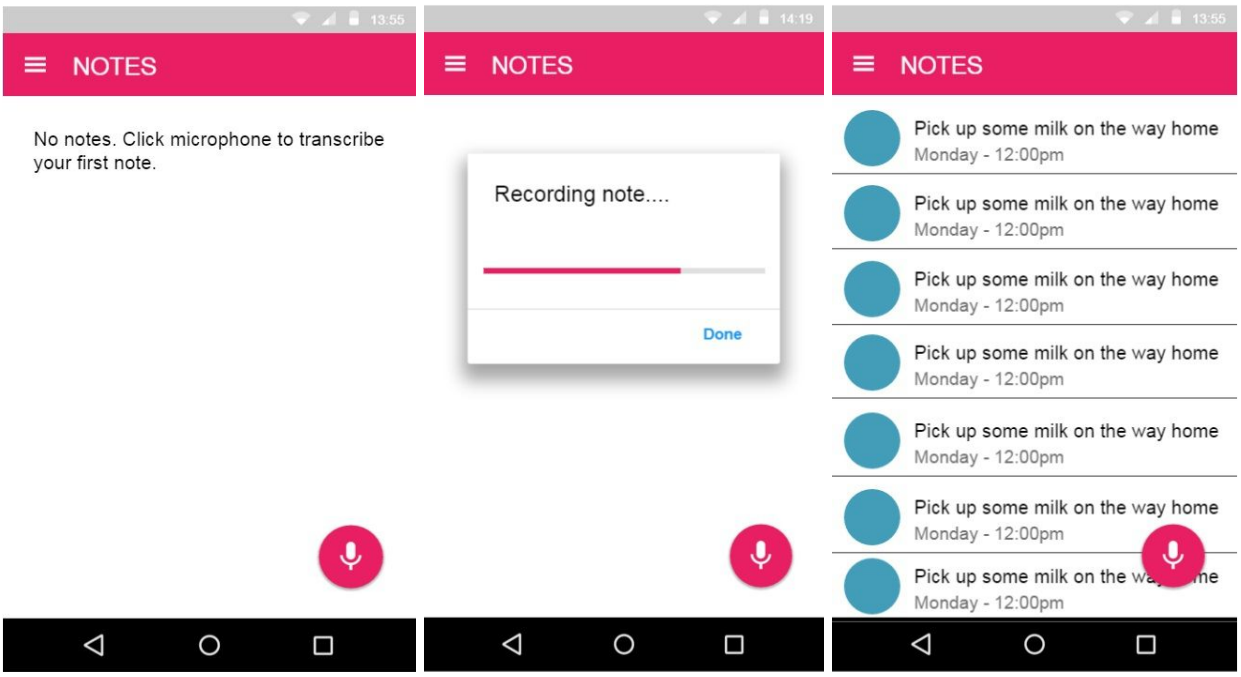
## Features

Milknote features the ability to:

- Transcribe spoken notes into written notes.
- Retrieve notes at a later stage along with the time and location they were made.
- The ability to delete notes or share them with friends.
- The ability to edit notes.
- The ability to trigger transcription from a notification or the lock screen.

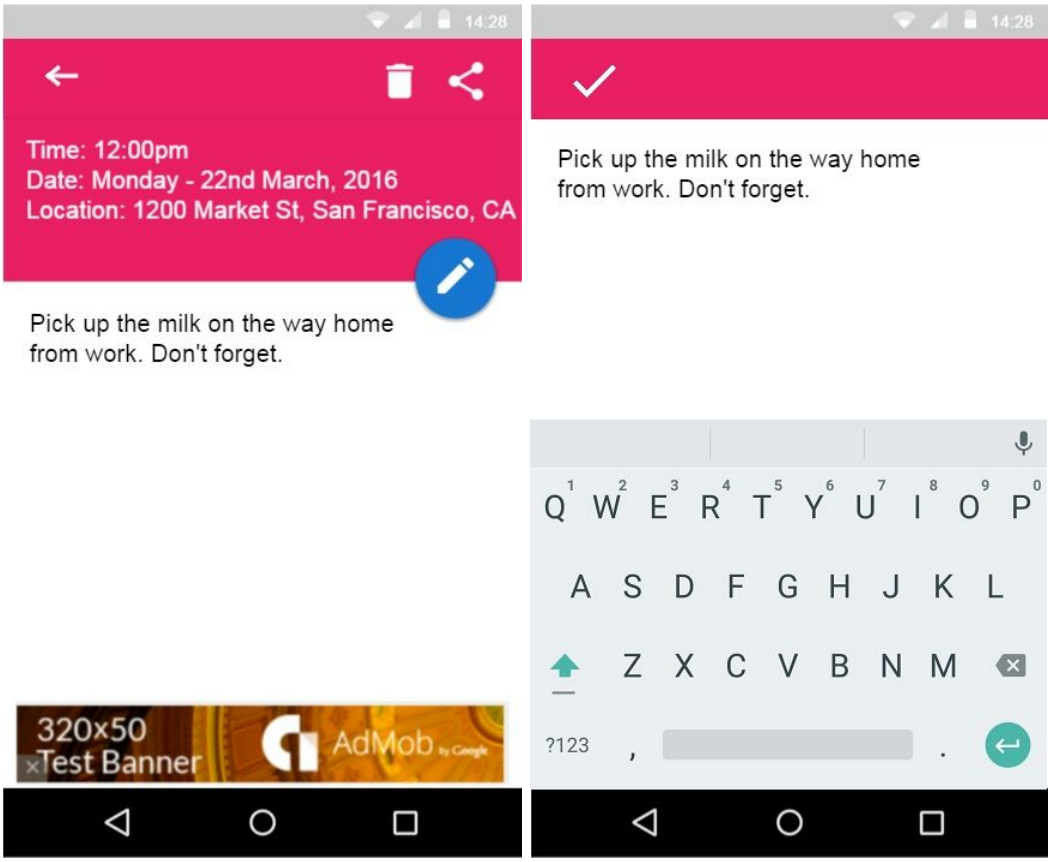
# User Interface Mocks

## Screen 1 - List View



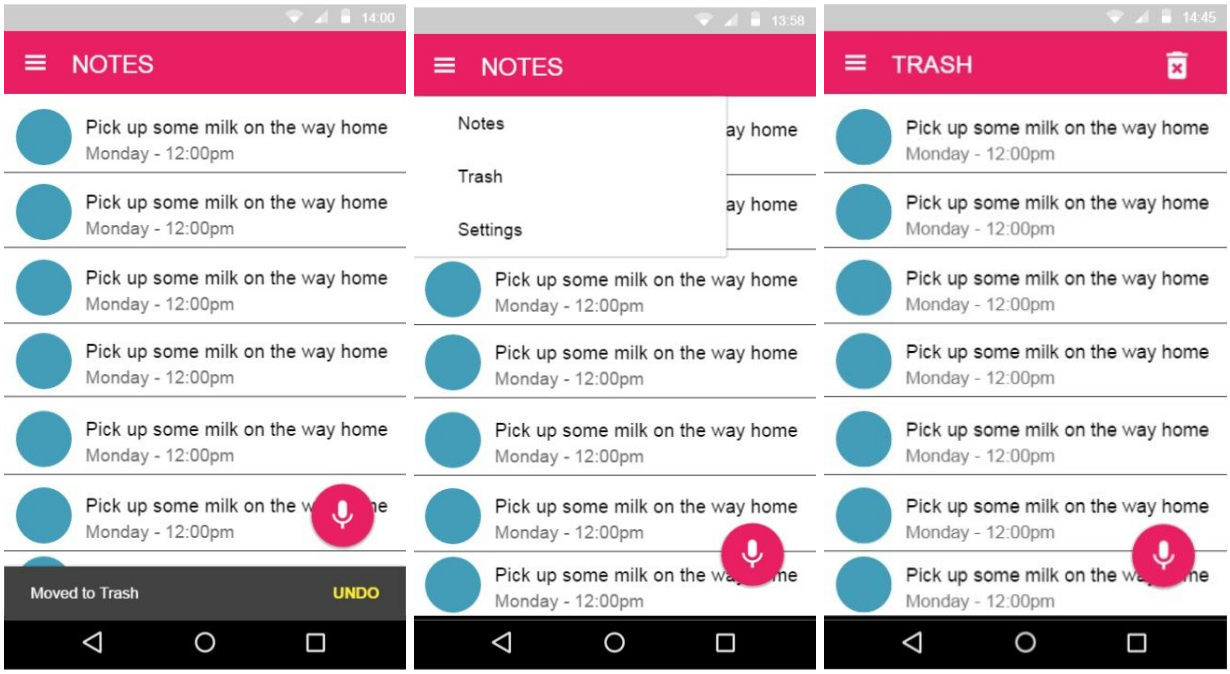

- Empty screen displays prompt to record first note.
- Note recording is a dialog - progress bar indicates maximum transcription length. Will navigate user back to list view after recording.
- List view shows all recorded notes

## Screen 2 - Detail View



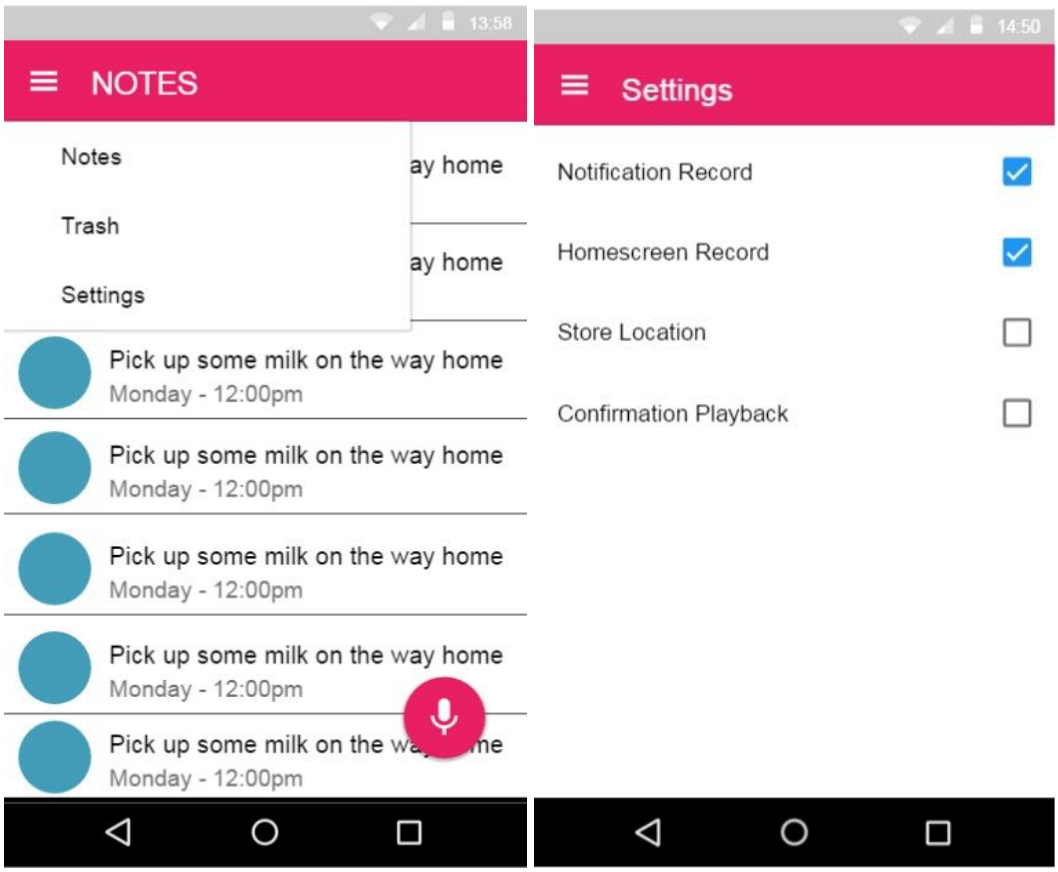

- Detail view allows viewing and editing of notes
- Displays time, date, and location of message which can’t be edited.
- Can share using ShareProvider which sends text of message
- Has small ad at bottom of message
- Clicking trash icon deletes message.

## Screen 3 - Delete Functionality



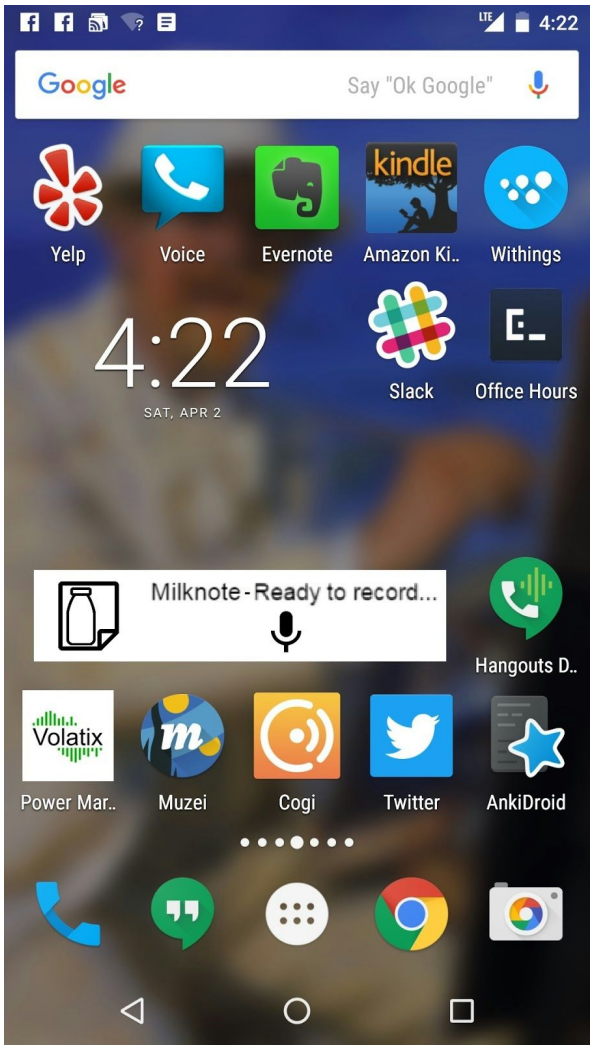

- After deleting a record in the Detail view the user is navigated to the List View where a snackbar notifies the user the message is moved to trash and allows them to undo.
- To view the trash folder the user clicks the hamburger icon to open the nav drawer and selects trash.
- In the trash list the “Delete Permanently” icon allows the user to delete messages permanently.
- When viewing a deleted message In the Detail view (not shown) the trash can is replaced by a restore icon. Edit and Share are not available.

## Screen 4 - Settings



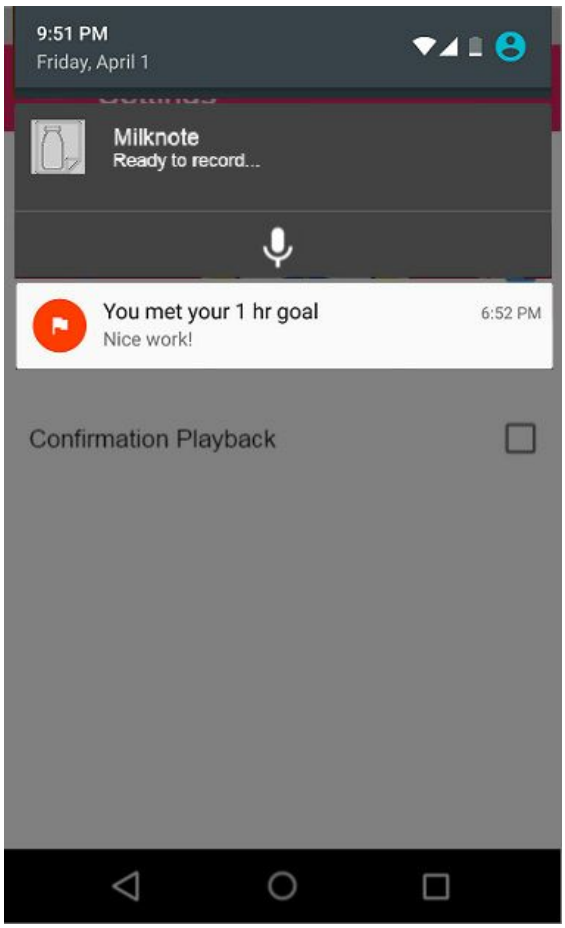

- Settings can be reached through the nav drawer in the list view.
- It will allow the users to turn on and off notification and homescreen recording of notes.
- Other features like store location and using text to speech to playback transcribed speech (potential feature) can be turned on and off here.

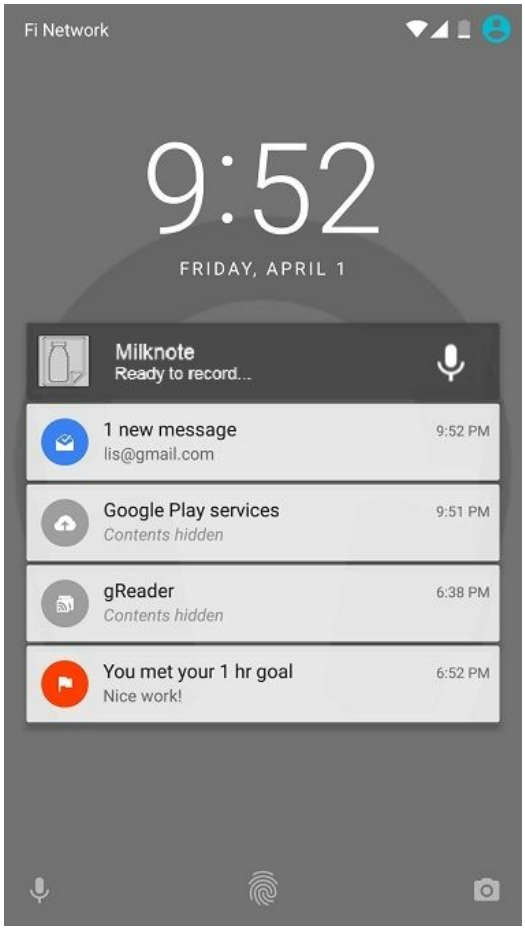
## Screen 5 - Widget




- Widget allows recording without opening app
- May use text to speech or toast to provide confirmation of transcribed text (will need to see how quickly response comes back).

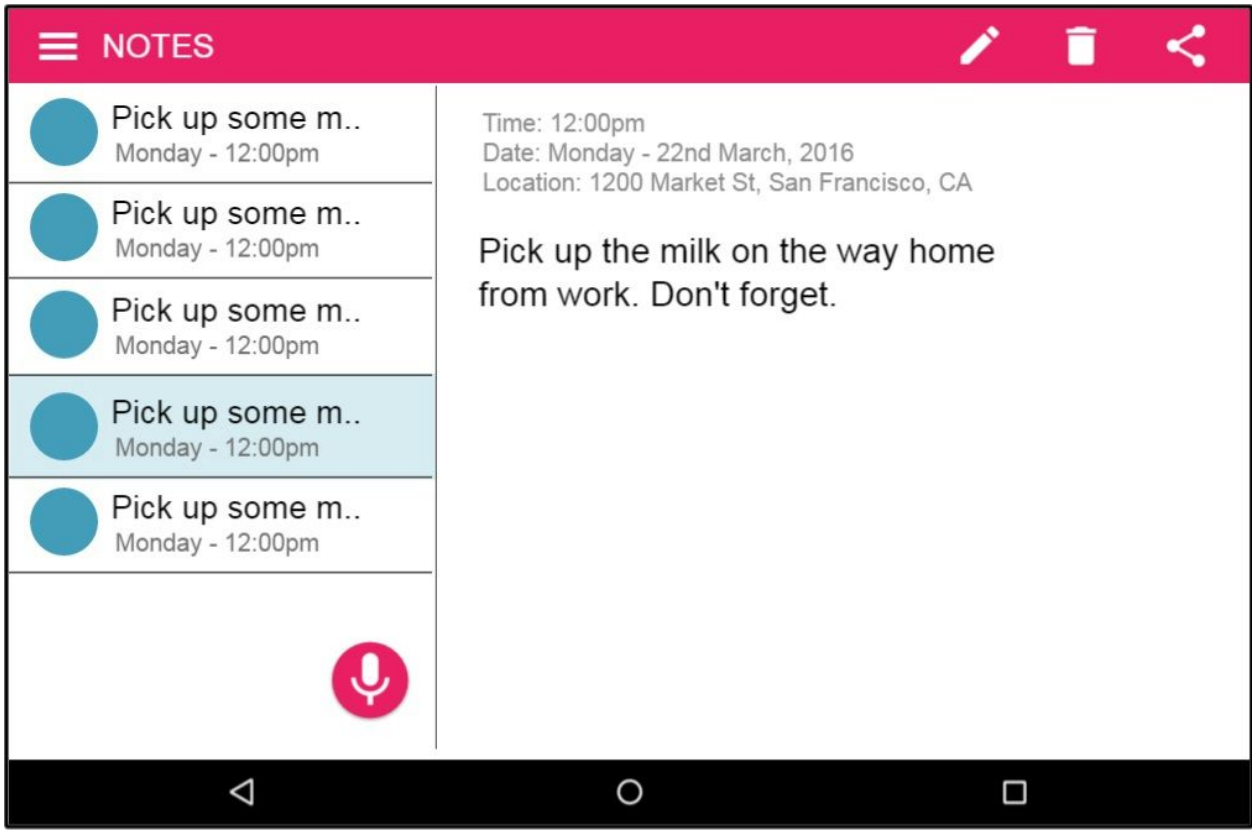
## Screen 6 - Notifications




- Add controls similar to media controls to notification that is launched when the app is launched or reopened.
- Allows user to record note from lock screen or notification without opening the app
- May use text to speech or toast to provide confirmation of transcribed text (will need to see how quickly response comes back).

## Screen 7 - Tablet Layout




- Tablet layout combines the list and detail view into a side by side layout.

# Key Considerations

### How will your app handle data persistence?

The app will have it's own Content Provider storing the transcribed note content, current note folder(trash or main), and metadata including time, date, and location. Could also store support data like whether note was recorded through widget, app, or notification.

### Describe any corner cases in the UX.

Any navigation away from the recording dialog will end the recording.
Key concern is that current API requires network access to transcribe so will have to check for network access and provide error handling and messaging.

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

I'll be using Nuance Speechkit for the speech to text transcription. I downloaded sample apps from AT&T, Nuance, and IBM Watson found the Naunce transcription accuracy was much higher. I applied for the Google Speech API Limited Preview but have not heard back.

# Next Steps: Required Tasks

My preference is to verify the technical functionality then add the visual polish.

## Task 1: Create project & verify Nuance functionality

- Create Android Project
- Import and configure Nuance Speechkit libraries (using Nuance sample app as guide)
- Build simple view with button and textbox to test Nuance. Clicking button should trigger recording, transcribing, and writing transcribed text to text box (build on AppCompat theme).
- Add feature to store transcribed value in shared preference. Change text box to read from shared preference (preparation for widget and notification).
- Create simple widget with record button. Button should operate the same as app button. Verify that transcribed text is stored in shared preference by opening app.
- Build simple notification and test like widget.
- Build simple lock screen notification and test like widget.
- Add toast to display transcribed text when recorded from widget.

If all these steps pass the Nuance library will support the desired application functionality.

## Task 2: Store transcribed values and metadata

- Define data schema
- Build SQL Lite Content Provider
- Modify simple view from Task 1 to use RecyclerView to display list from content provider.
- Modify app to store transcribed text using content provider rather than shared preference.
- Verify values transcribed in app, widget, notification, and lock screen are stored in content provider and can be retrieved in the list view.
- Build functionality to store date, day, time, and location of each transcription including associated permissions.

Completion of this task will verify that multiple records can be transcribed and retrieved.

## Task 3: Implement List View

- Modify simple view from Task 1 to be List View
- Add empty state message
- Make button a FAB - Add transition from microphone to stop and back.

## Task 4: Implement Detail View

- Build view including metadata in header & transcribed text in body.
- Build edit functionality to edit and store modified text.
- Ad admob ad to bottom of detail view

## Task 5: Trash Functionality

- Add trash can to Detail view that deletes record (labels record trash) and navigates user back to List view.
- Add Snackbar to List view with Undo functionality that is displayed to user after being navigated back to List View after Delete.
- Add navigation drawer to List view with Notes and Trash folders
- Modify List View to determine whether it is in Notes or Trash mode. In trash mode display “Trash” heading, retrieve records in trash, and display Permanent Delete icon
- Build Permanent delete that deletes records and navigates back to notes view. Add confirmation “Are you sure you want to permanently delete records?”. On confirmation navigate user to notes view and display snackbar  with confirmation message.
- Modify Detail View to determine whether it is displaying a note from Trash. In trash mode do not allow delete, share or edit. Only have restore button. Click restore button moves note back to notes and enables/displays normal options.

## Task 6: Settings

- Add Settings to List View Navigation drawer
- Build Settings View with required settings.
- Modify widget, notification, and any other code that needs to check settings before proceeding.

## Task 7: Visual Polish, Error Handling & Other Features

- Build dialog to display maximum transcribe length and current progress.
- Create Shared Element transition from List to Detail
- If not already implemented add material color palette.
- Clean up Widget layout
- Clean up Notification layout
- Add App Launch icon
- Build error handling for no network
- Build error handling for permission denied.

- Verify accessibility support (content descriptions & D-pad navigation)
- Verify all strings in strings.xml and translatable flag set
- Verify RTL layout switching on all layouts.

## Task 8: Tablet Layout

- Build List Detail Tablet View
- Verify Setting View in tablet mode and modify if required.
- Verify trash functionality in tablet layout.

## Task 9: Build & Package

- Ensure installRelease Gradle task works
- Equip with signing configuration, including keystore and passwords in the repository (refer to keystore by a relative path).

---

**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**"