GitHub Username: ijzepeda

Audio To-Do

Description

With just one button (widget as well). Save a voice note and save it to a to-do list. It includes Speech to text. And/or Text to Speech. It will detect times and days, and setup an alarm. When the alarm goes off, it will speak the Voice Note. Some of this features I haven't use them, therefore I am not sure if they are feasible (such as automatic alarm)

Intended User

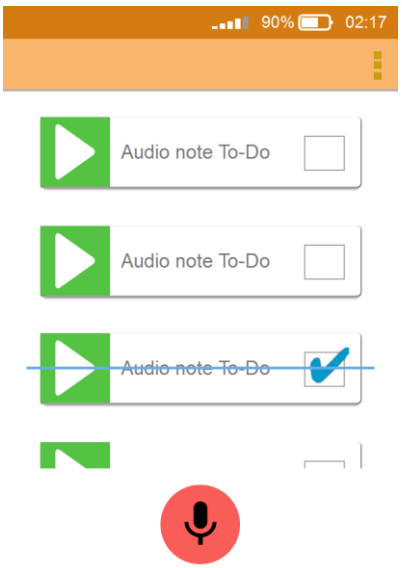
Productivity. Anyone with tasks on their life.

Features

- Record voice note
- Speech to text
- Alarm
- To-do list

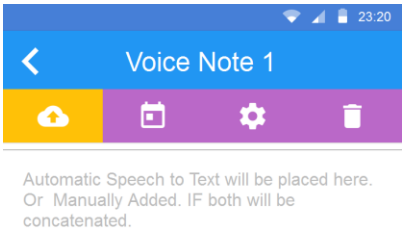
User Interface Mocks

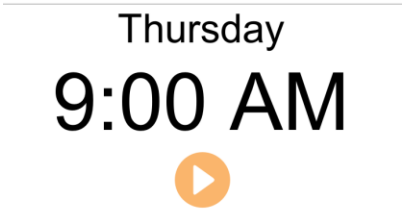
Main Screen



Pressing the red button will start to record the audio. When the button is released, the new voice note will appear on the list. When the app connects to WiFi it will Acces the Speech to Text API from google. And the text will be added to details of the Voice Note Card.

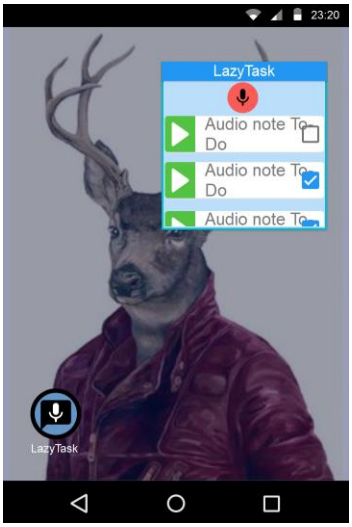
Card Details Screen





There will be 4 buttons: Upload Voice Note, Schedule, Edit/Settings, Delete.
Upload: will prompt an Alert Dialog to notice the user that it will consume Data. After receiving the text, it will be attached to the description.
Schedule: will allow the user add an alarm on a date, day or time.
Edit: will enable edition on description and title.
Delete: will delete the voice note with its audio file.

Widget



The widget will display a recylcerview of the voice notes and a button to easily record a new voice note. [I want to create an ViewOverlay REC Button.]

Key Considerations

How will your app handle data persistence?

Content Provider

Describe any corner cases in the UX.

The App will have two screens only [in tablets or landscape for large resolution devices a Master Detail Screen]. Where the first and Main Activity will have a recyclerview that directs to the following activity.
Details activity will have a back button on the action bar (Navigate up).

If App is Stopped it will return to the main activity. Unless it is just paused, return to details (no edition enabled if it was active)

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

None.

Describe how you will implement Google Play Services.

Speech To Text API to fetch text from an audio file to add a description.
AdMob on free flavor.

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

- Main Screen: create an Activity with a RecyclerView, Adapter and a Card. Detail Screen: an Activity that will have an Actionbar, a set of buttons
- Implement the content provider that will save the Voice Notes that are composed by: Title, Descrition, date, day, time, URI, finished Boolean.
- BroadcastReceiver to detect Wifi Connections.
- Server to fetch text from the audio file to Google SpeechToText API.
- Add proper permissions to the manifest.

Task 2: Implement UI for Each Activity and Fragment

- Build UI for MainActivity
- Build UI for VoiceNoteDetailsActivity
- Build UI for Widget

Task 3: Widget

- Having the MainActivity RecyclerView working. Develop the widget with the form of MainActivity.
- Checkbox will interact directly on the widget
- Rec Button will record a new voicenote without opening the app.
- Tapping on the card will open its details on the app

Task 4: Add Resources, LTR, RTL, and Accessibility option.

- Place all Strings in Res folder.
- For all views enable the LTR option and any other accessibility option.

Task 5: Gradle

- Setup all dependencies, sdk and paths.
- Create two flavor versions, a paid and a free version
- Add the API KEY from google console to work with AdMob and SpeechToText.

Link to demo:
https://pr.to/65XNHX/