By: Alexander Wu, Yi Chen,Joseph Fodera,
Shashank Garg, Shane Gan

Procrastaint

A ToDo list for Android & iOS with a NLP

Stack

Kotlin Multiplatform

We used Kotlin Multiplatform to share backend, such as database and major logic with each other without having to rewrite everything in both Kotlin and Swift.This works relatively well until Android and iOS starts to diverge due to not having the same UI elements.

Shared Kotlin

The shared portion was built using Kotlin Multiplatform. This contained all the algorithms for the NLP and database for both iOS and Android.

KWhen was the library built for the NLP. This detects times in sentences and returns the found data. It was also built using Kotlin Multiplatform to support iOS, Android, and more. It is hosted on Maven Central where updates can be released with Github Actions.

Room was the database framework used which let us interact with the database in a safe way with minimal SQL.

Initially had some issues creating our development environments, as the SwiftUI portion of the team needed to compile the Kotlin Multiplatform in order to create a working build. Eventually got everything running and included documentation for newcomers.

Organization

KWhen (NLP) **Procrastaint**

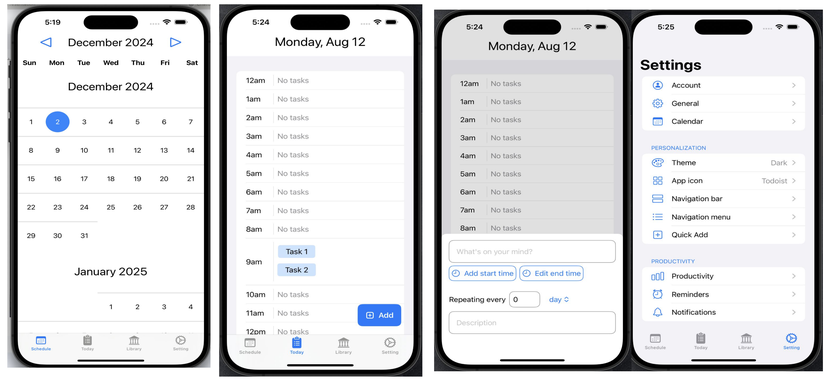
All code is hosted on the Githubs above.
KWhen is auto hosted onto maven with Github Actions
Kotlin code smell and formatting is managed by Detekt and KtLint

About Us

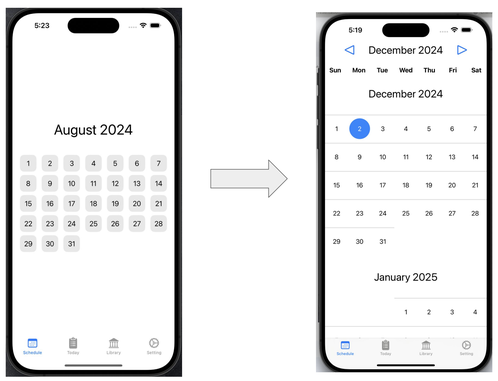
This is a ToDo list app aimed at having you click less to navigate. Your time should be spent doing tasks, not making them. Creating a new task is as easy as typing. You write a sentence and the NLP will take the time you want and assigns the task to that time. This would handle various formats of times including range based tasks and repeating tasks.

Results

Full IOS app GUI build. It is currently not attached to the backend but will be in the future. Full IOS app, non-functional UI-UX driven build. It is currently not attached to the backend but will be in the future. Schedule, Today, Library, and Settings tabs were created as the main form of navigation throughout the application.

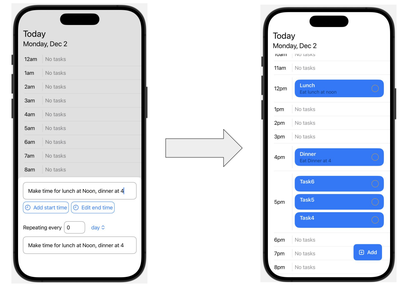


A calendar view overhaul was done. Addition of a month tabber with working right/left arrow buttons to create a more seamless, yet defined transition through months and an overall better UX.



Main Objectives

-Allow for the user to submit a a non-formatted entry
-have the NLP process that entry
-Have the NLP update the users calendar correctly based on the entry
-have the GUI update accordingly based off of changes made



Android

The Android app was built using MVVM architecture along with Jetpack Compose and UIKit.. This was combined together using adrielcafe/voyager to create screens and viewmodels in an easier way with the combination of Koin. Koin was used to fetch the database through dependency injection to be able to add new tasks and retrieve old ones.

Conclusions

-Most of the group has never worked with SwiftUI or Kotlin so the entire project was a big learning experience
-Learning how the Kotlin backend and Swift Frontend had to connect was pretty difficult, but once we got through that, it was smooth sailing
- Having the ability to view edits made in real time using the Xcode preview feature was extremely helpful while developing with a new language

Future Goals

We added future tasks that we did not get to this semester to our Github issues, making it easier for new contributors to jump in:

- Connect to Google/Apple Calendar
- Make a calendar to see all tasks
- Add widgets to quickly add tasks
- Connect with Google Assistant/Siri