Mobile Application Development

Project Summary

Application Name: Academic Organizer

Author: Nelson C. Araujo

Date: July 22nd, 2022

Table of Contents

Executive summary

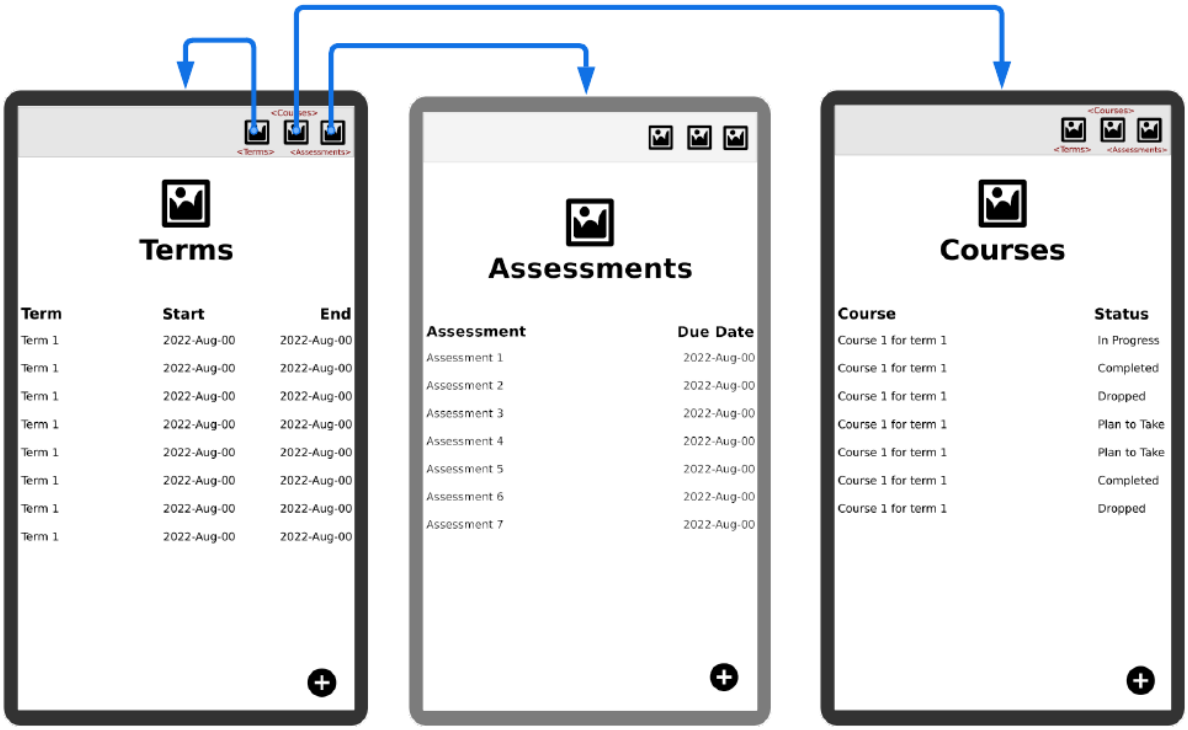
Academic Organizer is an application designed from a student's perspective that allows the student to organize each school year's term, courses per term, and track assessments per course. It provides the students the ability to organize and plan their academic program from start to graduation. Academic Organizer provides a high-level view of when terms and courses start and end as well as when assessments should be started and their due date.

Storyboard

The following storyboards outline the workflow of each user action in the application.

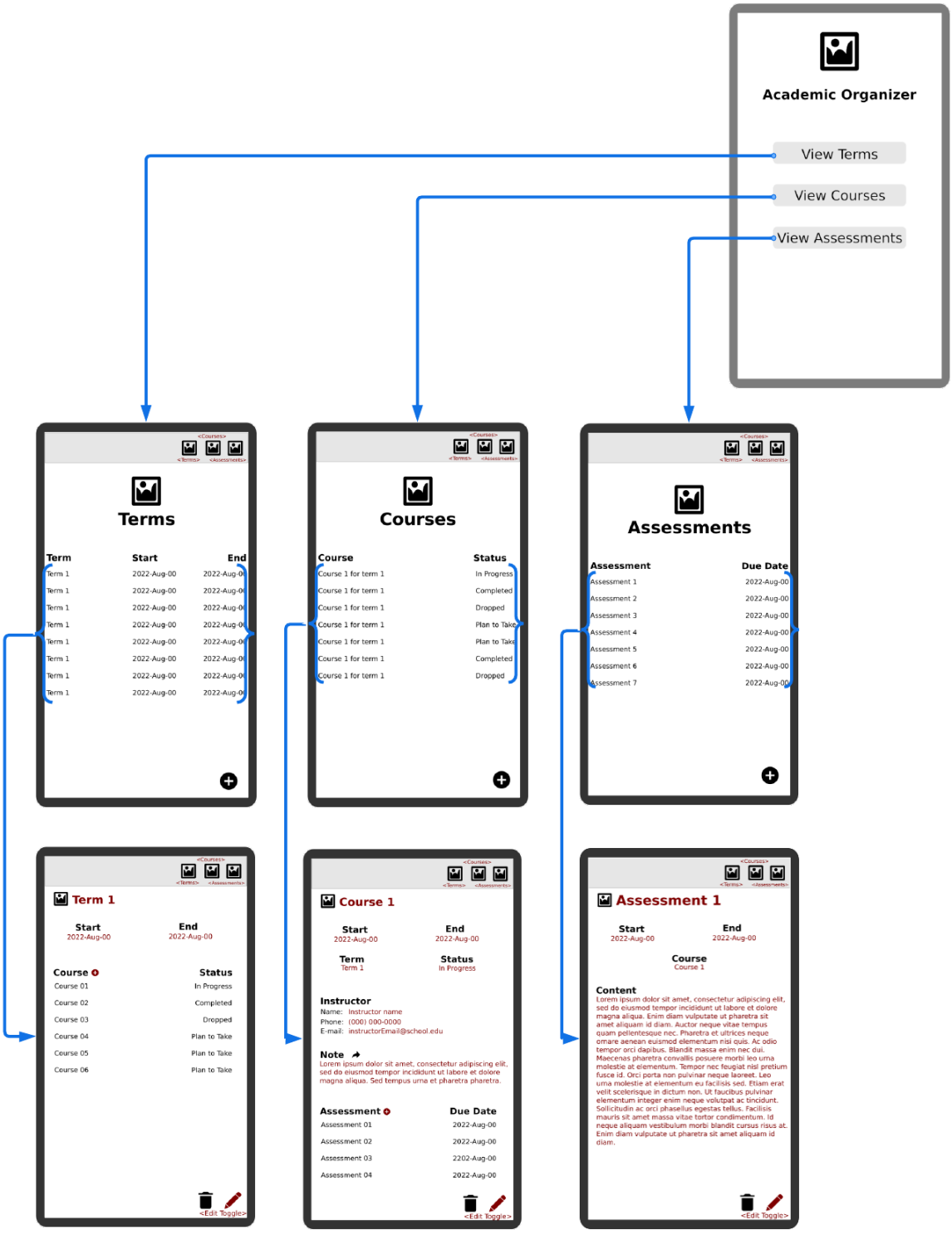
Menu

The following storyboard outlines the flow of each menu option.



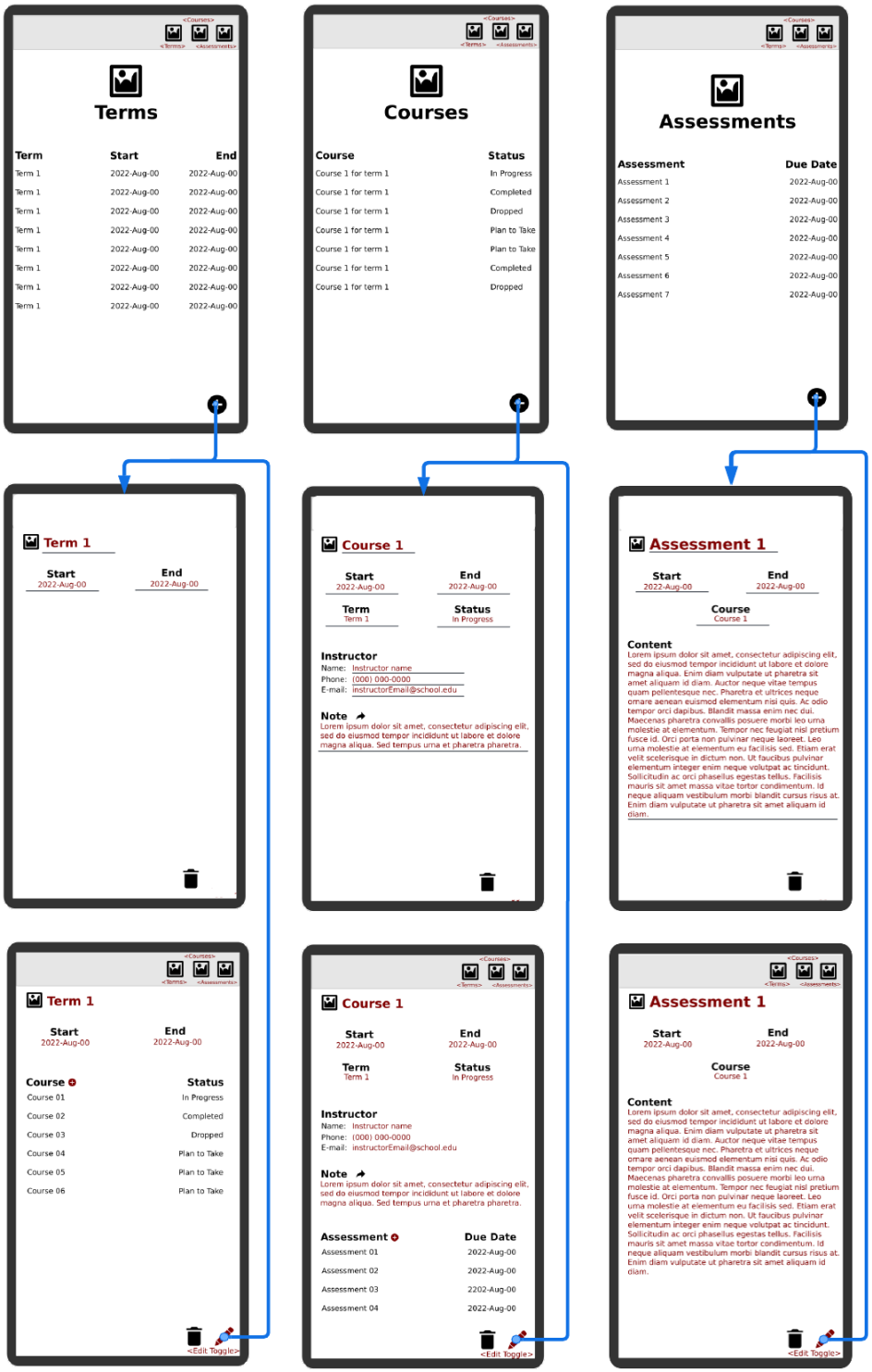
Main view layouts

The following storyboard outlines the user flow of each view layout.



Add and edit

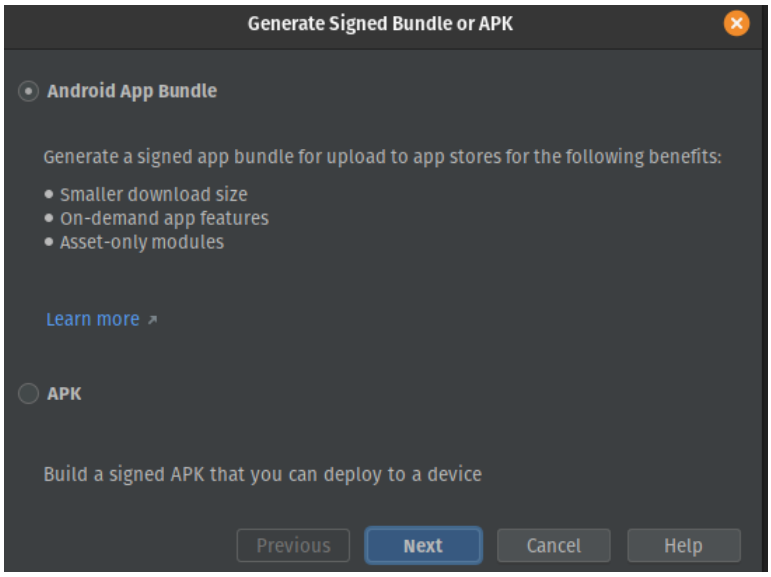
The following outlines the user flow for adding and editing terms, courses, and assessments.



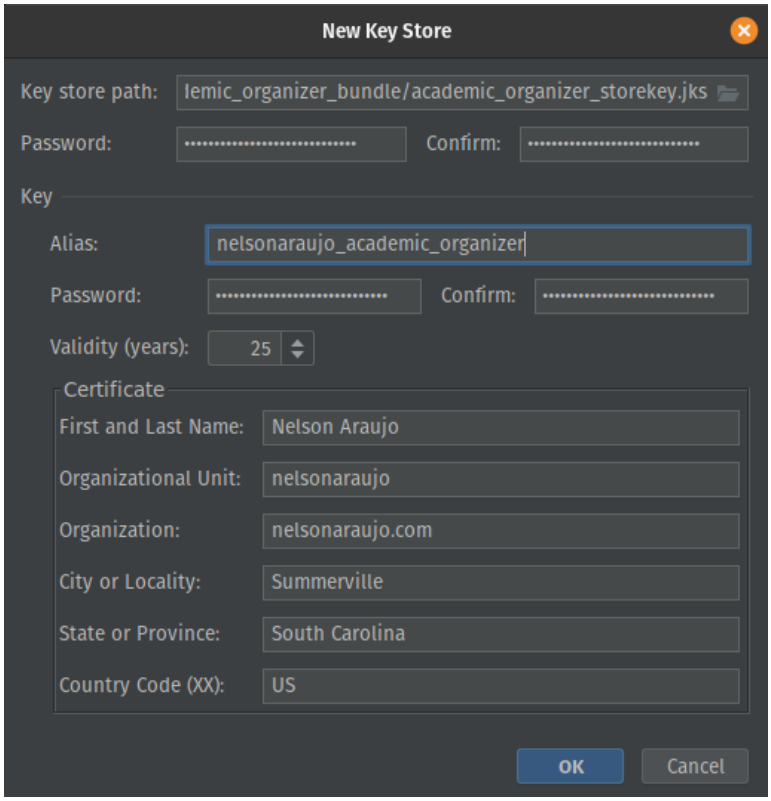
Signed Bundle / APK

The following screenshots follow the process of generating the signed bundle/APK from Android Studio.

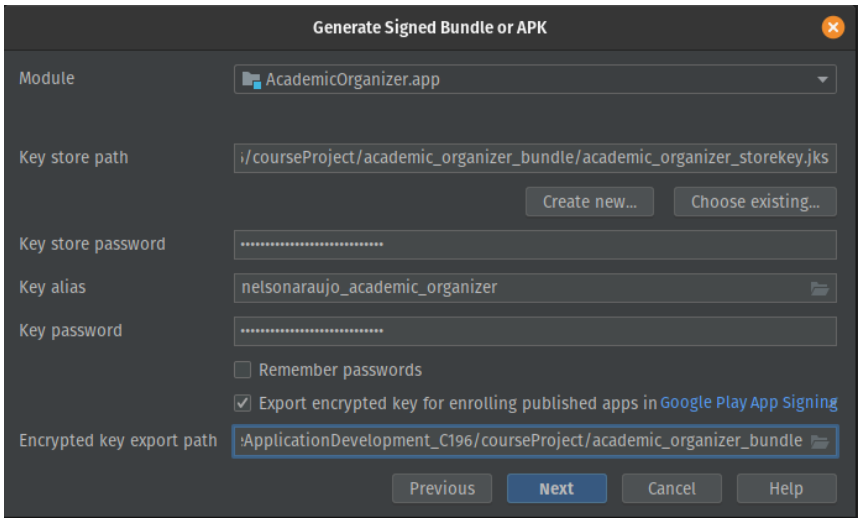
1. Select Android App Bundle option.



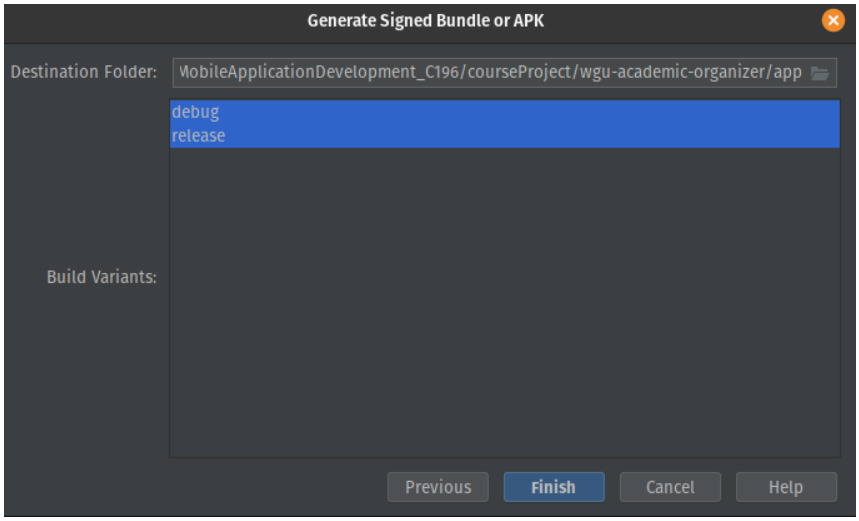
2. Generate a new key store.



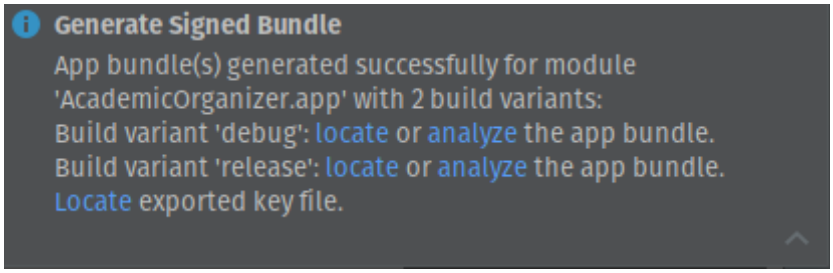
3. Select the signed Bungle or APK options.

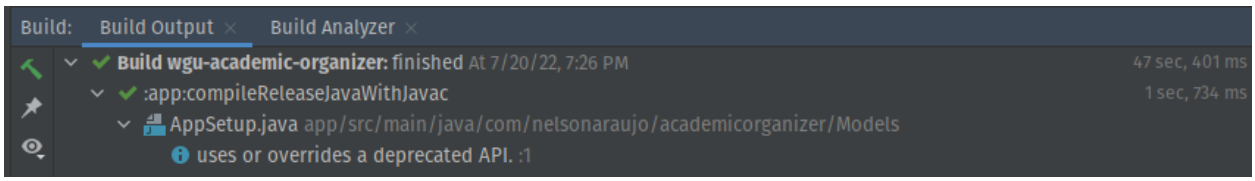


4. Select the build variants.



5. Successful generation of the signed bundle.





Reflections

Thought patterns of developing for a phone versus a table.

Developing an application for a phone versus a tablet has a couple of significant thought pattern shifts. The first and perhaps most important is that the phone has limited layout space compared to the tablet. The second is the difference in how users operate each device. Phones are usually operated using a single hand while tablets are more commonly used with two hands.

To enhance the user experience of this application for a tablet the use of fragments would be required. A fragment allows Android to make layout decisions based on the screen space available to the individual device the application is running on. Fragments alone can enhance the user experience significantly.

Fragments enhance the user experience but are not the only change that would be required when developing for a tablet versus a phone. The use of alternative layouts for a tablet would be required as well. The fact that the user has two hands on the device provides the opportunity for a tablet layout to give the user quick and easy access to many more functions of the application without having to move their hands on the device.

Target operating system and compatibility.

This application targets Android 12L, API level 32, but is compatible with Android 8, API level 26, and newer. Android 8 introduced notification channels, auto-sizing text views, and the content provider refresh functionality that is used in this application.

The challenges faced during the development process.

While developing this application there were multiple challenges faced. The top challenges that caused delays in the development process included the layouts initially designed being too cluttered for the phone screen resolution, Android function deprecation, schedule alarms, and schedule alarm delays.

During the planning and design phase, screen wireframes and layouts were created but unfortunately during the implementation phase, it was found that the layouts were too cluttered and didn’t fit on the phone’s screen resolution. Many of the fields that required the user to enter or edit text didn't offer the user enough space to properly add or edit the existing text. Many of the layouts were redesigned and recreated in Android Studio.

While executing the design some of the functions introduced by the learning material were deprecated in Android 12L. An example of this is the `getLoaderManager()` function, this particular function was deprecated in Android P, API 28. Android is a fast-evolving platform that Google is

constantly making improvements to. Although this is good for the platform in general it can cause delays while the developer researches the best path forward when a specific function has been deprecated.

Lastly scheduled alarms were the cause of many weeks of delay during the implementation phase. The concept of properly utilizing scheduled alarms to set up a user notification at specific dates was a challenge to understand and visualize. Once the concept was understood an additional challenge was presented, the notifications were inconsistent when presented to the user. After changing the date on the device to the specific date it could take 45 seconds to 5 minutes for the notification to appear.

How the challenges were overcome.

The development of this application provided multiple challenges and opportunities to learn and improve my knowledge of the platform. The key challenges faced included the layouts initially designed being too cluttered for the phone screen resolution, Android function deprecation, schedule alarms, and schedule alarm delays. Many of these challenges caused delays and frustration but they improved my knowledge of the platform directly related to the specific challenge and other areas as I researched the solution.

The first challenge came during the start of the implementation phase when it was found that the layouts were too cluttered and didn't fit on the phone's screen resolution. To resolve these issues and provide adequate field space for the user to add and edit the fields most of the layouts were redesigned. During the redesign, I ensured to provide ample space for the user. This redesign delayed the development for a week but it provided a better user interaction.

As with all new technologies Android is constantly being improved for a better user and developer experience but sometimes these improvements can be the cause of some challenges as well. During my development time, I found functions that were deprecated in the current Android version that was used by the training material. When these were encountered it caused delays while an alternative was researched. For example the `getLoaderManager()` function was replaced by `LoaderManager.getInstance(this)`. During my research, I became more knowledgeable on how these particular functions worked and why the replacement was an improvement to the platform.

Scheduled alarms were my biggest challenge in this project. I struggled to understand how to properly use them to set up user notifications for specific dates. To overcome this challenge I reviewed Android's documentation, reviewed alternative ways to notify the user, created prototypes to evaluate possible solutions, reviewed lifecycle diagrams to understand the lifecycle of a scheduled alarm, and discussed it with the course instructor. After the research, the prototypes, and the discussion with the course instructor I understood scheduled alarms and implemented them in the application.

During the testing of scheduled alarms, it was discovered that the user notifications were inconsistent. After changing the date on the device to the specific date it could take 45 seconds to 5 minutes for the notification to appear. Initially, it was thought to be an error in the code logic but after some research, it was found that it's the intended behavior of Android. To reduce the number of distractions to the user, Android batches overdue notifications and notifies the user at one time which can cause delays on the overdue notifications. To resolve this issue the `setExact()` function must be used and `<uses-permission android:name="android.permission.SCHEDULE_EXACT_ALARM" />` added to the Android manifest.

The lessons learned.

This project provided many lessons to be used in future Android development projects including the use of fragments, improved screen layouts for user interactions, and contacting an instructor or senior developer when concepts aren't clear. The use of fragments and improved screen layouts for user interaction in future projects will improve the user experience. Reaching out to an instructor or a senior developer when concepts aren't clear is a key life lesson.

The use of emulators versus a physical device.

Android Studio relies on emulators to display and test applications being developed. This has great advantages such as the ability to test the compatibility of an application on multiple devices, quickly build an application and install it on the virtual device, and test an application on different device screen sizes. The advantages of a physical device and disadvantages of emulators are that physical devices have more power and can run an application better. Physical devices also have the advantage of running the application on real hardware that the tester can physically hold and test the application as the end-user will use it.

Sources

Learn Programming Academy [LearnProgramming.academy]. (2021, August 1). Android Java Masterclass - Become an App Developer [Video]. Udemy. https://www.udemy.com/course/master-android-7-nougat-java-app-development-step-by-step/

Masri, A. M. [Mastering Android]. (2022, July 1). The Complete Android 12 Developer Course - Mastering Android [Video]. Udemy. https://www.udemy.com/course/the-complete-android-10-developer-course-mastering-android/

Android Mobile App Developer Tools. (n.d.). Android Developers. https://developer.android.com

Stack Overflow - Where Developers Learn, Share, & Build Careers. (n.d.). Stack Overflow. https://stackoverflow.com/