**Reflection on the Creation of My Mobile Application**

1. Developing this application for a phone rather than a tablet gave me a small window to fit my fragments and scroll views on each activity. I could’ve fit more fragments on the screen if I had developed this application for a tablet. Instead of having one list of tabs for a phone, I could put fragments side by side if I were developing for a tablet because of the larger screen.
2. The minimum operating system for this application is Android 8.0 (API level 26) and the target operating system for this application is Android 14.0 (API level 34).
3. Some of the challenges I faced when building this application revolved around my wanting to use Date and Enum objects for my class attributes. This created a problem when getting the extras from my Intents for an Activity. I tried calling the getParcelableExtra method which allowed me to specify the class of the object I was trying to get, but using the getParcelableExtra method and specifying the class in the arguments of this method requires the device operating system to be at API level 33 and up which means the code would not run correctly for the minimum specified API level of 26.

I also had trouble with emulators eating too much RAM on my laptop which caused my computer to crash multiple times. When trying to build an emulator my computer would freeze and then shut down. There was one time when my computer crashed and my theme XML files disappeared for no apparent reason. I realized the issue when I tried running the program again and got a bug saying that the theme specified in the manifest didn’t exist.

1. What I did to fix the first problem specified in F3 was simply converting the Date and Enum objects to String objects when putting the extras in an Intent. This allowed me to use the getStringExtra method which would run on API 26. Using the getParcelableExtra method without specifying the class is deprecated and this is why I didn’t use the method in this way.

I solved the second problem by using my device to run the application. Using my device allowed me to conserve more RAM, especially since using the Chrome browser while coding eats so much RAM in the first place. I also followed GeeksforGeeks’ advice and would “invalidate caches on Android Studio, run one project at a time, disable all useless plugins, adjust my memory settings, and delete unused and unwanted projects to improve performance.” (GeeksforGeeks, 2021)

1. The thing I would do differently if I was doing this project again would be to either talk to a course instructor or watch the bike shop webinars before I started my project. I initially had a lot of activities in my application because I had no idea how this application would flow. I wasted time trying to figure out the flow of this application and after watching the bike shop webinars I realized that there was an easier way to do things. I changed the flow of my application and deleted some activities which made coding easier.
2. The pros of emulators are they are cheaper when testing the application on different devices because there is no need to buy multiple devices to test on. You can test multiple screen sizes and operating systems on emulators without buying a physical device. The cons of emulators are the performance of the emulators. Emulators consume a lot of RAM which drastically affects the performance of the computer used for developing the application. To run emulators on a computer it would need lots of RAM and computing power.

# References

GeeksforGeeks. (2021, May 3). *How to Speed Up Android Studio*. Retrieved from Youtube: https://www.youtube.com/watch?v=fOzzJ83A2og