Designing and developing a mobile application turned out to be very different from desktop application development. For this project, I developed on an Apple Macbook Air, using Android Studio as my Integrated Development Environment (IDE). The operating system used was the Android 9.0 API 28 on the Android Accelerated Oreo emulator. The application is meant to be used to help students track their school curriculum as well as provide alerts for specific approaching dates.

There were several challenges I faced during the development of this project. When I first downloaded Android Studio, my initial problem was figuring out how Android Studio itself was laid out and where to find certain things I was looking for. Furthermore, the file structure was very different from the past JavaFX desktop applications I worked on. As I started to get comfortable working in Android Studio, I ran into the issue of actually designing the application itself in the IDE. Due to this being a mobile application, compatibility with various screen sizes and layouts is of utmost importance. Constraint layouts in particular presented a new learning obstacle for me. Finally, I believe I had the most difficulty with the emulator. I was used to being able to boot up my application and manually test various functionality fairly quickly and efficiently when using JavaFX. Because I had to work through an emulator to test any functionality, which at times was rather slow and clunky, it was not as easy to quickly test the application. The process took a little bit longer than I would have preferred and made it more difficult to do things efficiently.

Throughout all of these challenges, nothing seemed too out of the ordinary or complicated to push through. Learning about how Android Studio worked as an IDE and recognizing the mobile application file structure just took time, patience, and experience before I grew comfortable working in this environment. Understanding constraint layouts was brand new to me, but it did not take long to figure out. A constraint layout helps with mobile application design specifically, so that if a device is either in portrait or landscape mode, the application display will have a similar, if not exactly the same, look and feel to the user. Unfortunately, there was nothing I could do to solve the problem with the emulator. Without access to a physical android device I had to be more deliberate with my functionality testing, so that I didn’t waste needless time booting and rebooting the emulator and the app within the emulator.

Since I am so new to software development itself, there is very little I would change from my current experience with coding and learning to code. I have come to realize that each difficulty faced in the projects I have built has given me new insight into something I can use for future projects or applications. The only thing with this specific project that I might have changed, would be access to an actual Android phone. This would have sped up a lot of the testing that I performed and made development a bit faster.

Emulators are a great tool for developers to utilize when testing for one specific reason: availability. With an emulator, you do not need access to the device you are developing an application for. However, my experience with emulators tends to skew negatively simply based on performance issues. Emulators are great because they allow for developers to create device-specific applications without access to the physical device. However, actually testing and developing in an emulator can be frustrating and slow due to the frame drops, keyboard malfunctionality, and general glitches when using the emulator. In my case, I did not have access to an Android phone, so I had to use an emulator. Overall, I think if a developer has access to a device, then it would be far easier to use that device for testing. However, if access to a device is either unavailable or expensive to acquire, then an emulator provides sufficient capability as a testing environment.