Team Jack Sprint 5 Documentation

1. Project Description:
   1. Original Description:   
      Grizzwalk is an app that helps the user keep track of their class schedule. Classes could be entered manually or pulled from the course catalog by entering the CRN number. With the class information the user could get a weekly or monthly overview of what their schedule looks like. They could also set alerts for exams and homework and such, and the app would send an alert to their phone’s notification bar to remind them about the upcoming assignment.
   2. Current Status:   
      Grizzwalk is an app for Android devices. The app will help the user keep tabs on their class schedule. The user enters their class information manually and a course list is displayed using that information. The user can see their classes on a day to day basis using the Monthly View calendar. Alerts for exams can be set on a per class basis, notifying the user via the Android notification bar when they are within a certain timeframe of the exam. The user can also get directions on how to get from one building on campus to another.
   3. Project Evolution:
      1. In general, the project is like how we imagined it, with a few features added and a few removed. The app helps the user keep track of their schedule through an easy to read interface, and it lets the user set alerts so they can be notified of an upcoming exam. In that sense, it is close to how we imagined it.
      2. We reduced our specification in some areas, and increased them in a few others. We did not implement a weekly view due to difficulties in implementing third party code into our app and the feature was mostly redundant with the monthly view in place.
2. Target System:
   1. We focused on Android Devices running OS version 5.0.1 at minimum. We picked smartphones for our project, since a schedule program that would be useful for checking at a moment’s notice is perfect for a device a lot of people carry around in their pockets. Our peers and our group also felt the MySail mobile app was lacking at the time the project started, so we were hoping to make an app that would fill that void. We chose to focus on Android rather than iOS mostly out of convenience and familiarity with the platform. Most of the members in our group were in possession of Android devices that could easily test our app. All members of our group had some familiar with Java, making developing for Android easier, since Java is the language of choice for Android apps. Only one of us was familiar with Swift, the programming language of choice for iOS, which would have slowed development down due to needing to learn another programming language.
   2. We changed the minimum required version of the Android Operating System from what it was originally. The minimum requirement was originally OS version 4.0.1 due to most Android users being at or above that version (around 97% according to Android Studio’s usage statistics), and at the time we did not think we would need any features added past 4.0.1. We changed our target to version Android 5.0.1 while developing the alert system, due to changes in how the notification bar works in 5.0.1 onwards.
   3. We did not have many problems due to our system choice. Aside from the alert system, the only other problem we encountered was when Austin was updating the building navigation algorithm for the first time. The update used something new to Java 8.0 called Lambda. While the algorithm worked, features from Java 8.0 could only be used in Android OS version 7.0 and up. At that time, only 2% of Android users were using Android OS version 7.0, so the algorithm was revamped into the algorithm found in the Sprint 3 deliverables, in interest of making sure more users would be able to use our app.
3. Timeline:
   1. Timeline as it transpired:
      1. Sprint 1:
         1. Basic functionality was completed. The user could enter in classes and the app would display them on the course list activity. Pretty much every feature we were going to add to the app depended on this functionality.
         2. Building navigation feature started, creating an algorithm that could navigate the campus in the four cardinal directions (North/South/East/West). The feature was not implemented into the UI at the time, so the destination had to be plugged into the code manually and the results read through the debugger in order to test the algorithm.
      2. Sprint 2:
         1. Monthly View was implemented. User could select a date on a calendar and see what classes they had scheduled for that day. Dependency for the exam alert system.
         2. Building navigator implemented into the UI. The user could enter two buildings into the app and get directions on how to navigate from one to the other. Building search algorithm also revamped, since the previous one started breaking as more potential directions were being added to it.
      3. Sprint 3:
         1. Alert System partially functional on monthly view. UI was in place and user could send an alert out to the Android OS. Would be improved further by the end of Sprint 4.
         2. Building navigator improved. More landmarks were added to help the user navigate campus easier. New Java class was added specifically to print out directions to the user.
      4. Sprint 4:
         1. Exam alert function completed. User could select a day and class from the Monthly View. The user could then how long before the exam they would like to be reminded. At the specified time, an alert in the Android notification bar will remind them of the exam, even if Grizzwalk is not up at the time.
         2. UI improved. Add class activity improved in general. Color scheme changed for a few screens to give the app more of a visual identity.
      5. Sprint 5:
         1. App complete.
         2. App has a consistent color scheme throughout all activities.
         3. Many bugs have been ironed out, such as the app crashing whenever the user would input the same building into the location and destination on the Get Directions screen.
   2. Discrepancies:
      1. Sprint 1:   
         Building navigation was not originally planned to have been started by this point, but it was since getting basic functionality started was easier than we had thought.
      2. Sprint 2:   
         Weekly view was not implemented. We ran into trouble trying to create a Weekly View from scratch, we expected it to be much easier than it was.
      3. Sprint 3:   
         Alert system only partially implemented, due to difficulties in sending information to the Android Operating System. Notification changes introduced in Android version 5.0.1. made it easier to implement.
      4. Sprint 4:   
         Cut building interior navigation feature completely. We did not think we would have a satisfactory implementation of the system in time for the end of the semester.
      5. Sprint 5:   
         Austin ran into issues and errors as he tried to expand the building navigation feature to include more blank spaces on our “map”.
4. Tools and Technologies:
   1. Used Tools and Technologies:
      1. Android Studio. Google’s official IDE for Android app development. Used since it seemed to be the best choice for Android Development IDEs. It also had some similarities with Android IDE Eclipse, which some of us had experience with.
      2. We made our app using Java, the programming language that most Android apps run on. Used because it is the go-to language for Android Development.
      3. Photoshop, image editing software. Was be used for creating a logo, mockups for UI before we had UI implemented, and UML Diagrams. Used due to group members having prior experience.
      4. Google Drive. Cloud-based file hosting service for hosting project files. Used so we could host files and documentation all on one service.
      5. Android Studio Debugger. The debugger built into Google’s official Android Development IDE. Used because it came with Android Studio the IDE we were using, for convenience.
      6. Oakland University campus map. Used for dividing campus buildings into a grid to get the coordinates to use in our building navigator feature. Found online directly from the website of the school.
   2. Unused Tools
      1. Third Party Weekly View Library  
         Unused since we had problems implementing their code into ours.
5. User Response:
   1. Producer response was generally positive for Grizzwalk. Users thought the app was quick and easy to use thanks to a straightforward and easy to understand GUI. Despite that, they found a fair number of problems that we did not catch in time for testing, or just plain did not think about. For example, we did not realize how easy it was to accidentally tap the Clear Course button and throw away the schedule until a producer did it. We had the button bring up an additional prompt before clearing the list, asking the user if they are sure they want to clear their class list. Accidents do happen, and clearing the schedule due to a single accidental tap of the screen would be obnoxious to say the least. The virtual keyboard on the Get Directions page would also cover up the directions list unless the user pressed the back button to manually close the keyboard. We did not realize how annoying this was until the producers had told us how annoyed they were by it.
   2. We made various minor tweaks, such as making it so the Android keyboard closes automatically when the Get Directions button is pressed on the Get Directions Activity, so the user can automatically see the information they want without having to press an additional button.
   3. This experience helped us to learn that it is easy for developers to lose sight of the small problems when you are busy trying to fix bigger problems and add features. The reset button issue previously stated is an example of that, we had to deal with it every time we tested the Get Directions activity, but we did not think anything of it because we had bigger problems to worry about. We also learned that what may seem obvious to a developer who has been working night and day on a project, might not be obvious to an end user that has never seen the project before.
6. Debrief and Review:
   1. In general, we think we were successful in our project. We delivered a scheduling app that is fairly easy to use and can help new students keep track of their schedule and navigate campus easier, which was the ultimate goal.
   2. The building navigation was probably the most difficult part of the app that actually made it in to the final product, due to the number of times it needed an overhaul. The easiest part was getting basic functionality up, since it was just setting up a few classes and a singleton array to store class information.
7. Demo under “Grizzwalk Demonstration” document in deliverables, for the sake of not cluttering up the rest of the documentation.