Prioritize – Progress Report 5

Joel Wilhelm, Cody Jones, Shahrukh Rehman, Wajahat Iqbal

1. Current Progress
   1. Completed

We have completed the JSON code for this project. This code creates a JSON object that gathers values from the UI, temporarily stores them as an object, and creates a String in the proper format for a JSON text. This text will then be stored in the SQlite database. The JSON code will also be responsible for parsing through the string, and converting those values back into a JSON object, which will then be usable by the code. Testing will be talked about in the testing section.

The UI is functionally complete as well. We may still make tweaks to the UI, depending on what happens with the rest of the project, and implementing features that have not been completed as of yet. The UI is functional as of now.

The Alarm and Alerts are almost complete. A few minor bugs and glitches still occur, but are being worked out.

* 1. In Progress

The Priority Algorithm is almost completed. We have decided to split up the algorithm into sections, depending on how much time remains from the creation date to the due date. The first section is completed, the section that deals with a due time within a week of creation. We are still working with what happens when the due time is more than a week out.

Integration of all features. We have not fully integrated all our moving parts together yet. This will happen, hopefully, by the end of this week. We are still working on testing each component individually before they are implemented together. Then, testing can be done on progressively bigger parts of the project until everything is complete.

The Device Synching is not quite complete yet. Initial login and login persistence is completed. Files currently save to appropriate folders. Files are accessible across devices. Listener service is implemented, and downloading reminders in background works. It has not yet been integrated with the JSON implementation, or the SQlite insertions. In addition, we are having difficulties with server calls, we can’t force sync repeatedly.

The SQlite database is still being developed as well. Reminders are currently being stored in the database, and the database is functional. However, queries to display entries on the home screen are not yet complete.

1. Plan and Timeline

***We are behind our goal and timeline as of this point.*** We wanted to be nearly done with coding, working on testing and debugging to have a functional application at this point in time. There is still work to be done, finishing up each part, then integrating them together.

We want to finish coding by Friday, April 7th. In addition, the User Manual and Reporting phase should be worked on until that point as well. If we have time, we would like to develop additional features as well, such as Geofencing, and adding a stopwatch and countdown timer as well.

1. Coding / Coding Responsibilities

Joel Wilhelm was responsible for coding the JSON object parser and creator, as well as helping Cody with the Priority Algorithm. He also was responsible for making sure all the documentation was finished, and was of good quality. He also is responsible for helping test various code.

The JSON file was written in Java, and tested for correctness in Java. The Priority algorithm was written in Java as well.

Cody was responsible for working on the SQlite Database, the Priority Algorithm with help from Joel, and synching the data with the Google Drive API. (INSERT MORE INFO HERE)

Shahrukh Rehman was responsible for the Front-End of the application. He is also integrating all of the different sections of the code, and tying everything together. (INSERT INFO HERE)

Wajahat Iqbal was responsible for making the Notification/Alarm object. He is retrieving reminder information from the SQLite database for Notification/Alarm services. Both of these object were written in Java.

1. Testing

Joel Wilhelm wrote a unit test code for the JSON object creation and parser to make sure it worked properly before sending it to be integrated with the rest of the project code. In order to test this code, a new JSON object was created, and field values were set. The code should then take each field, and convert it to a JSON string. When the JSON code needs to unwrap and parse through a string to get the fields, it takes a string. So, the string that was created was passed to a new JSON object, and the fields were set to be what the values in the string were. If the two objects contained the same values for each of their fields, both the string creation and the string unpacking were successful.

This code was run a few times, and each time, the code functioned properly, the Unit Test was completed successfully.

UI Testing – ADD INFO HERE

Alarms / Alert testing – ADD INFO HERE

Priority Algorithm testing – ADD INFO HERE

Google Drive Testing – ADD INFO HERE

SQlite Database Testing – ADD INFO HERE