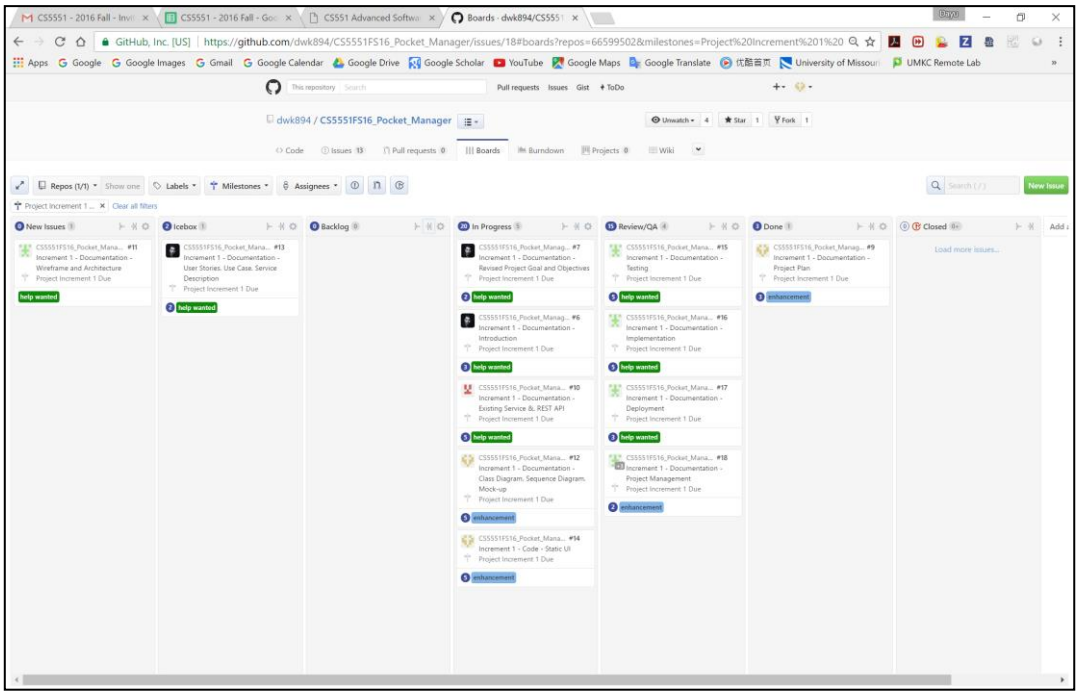
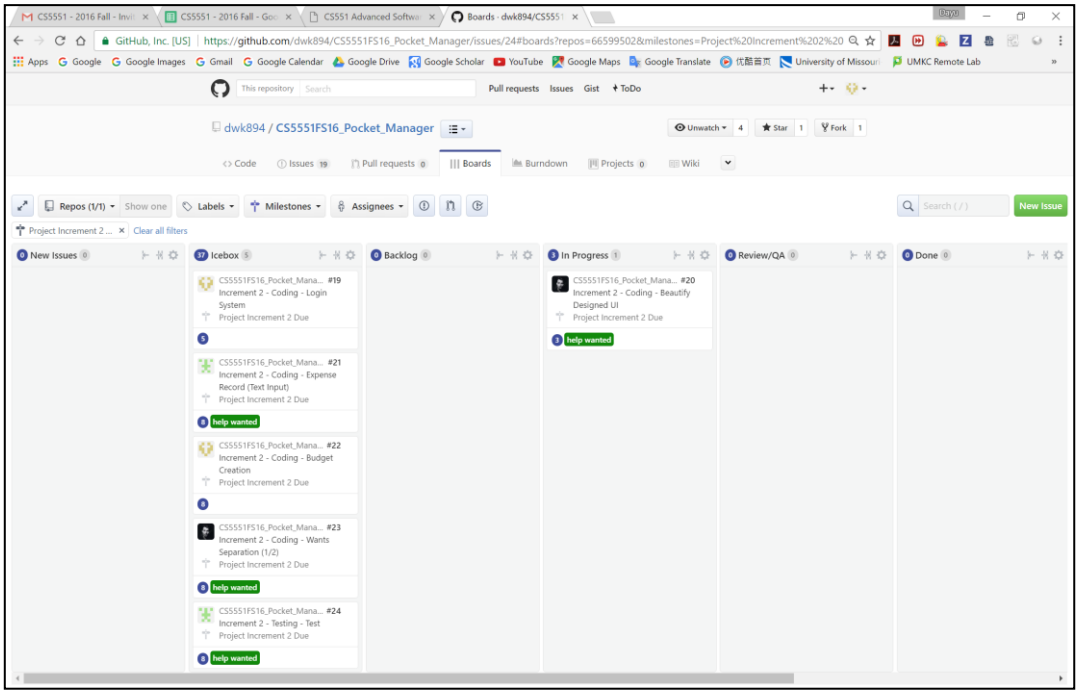
1. Schedule for the Four Different Increments

Based on the factors mentioned above, we finally created a project plan. Using the ZenHub tool, iterations of our project plan to develop the entire system (also based on the due day of increment reports) is shown in Figure 6, which is a series of screenshots. For more detailed information, you can look at our actual GitHub repository at https://github.com/dwk894/CS5551FS16_Pocket_Manager. Note that since the report guideline is only available for the first increment, our plan for the increments 2-4 is mainly based on the programming issues. Also, our plan is just good for now and may change or tremendously change in future.

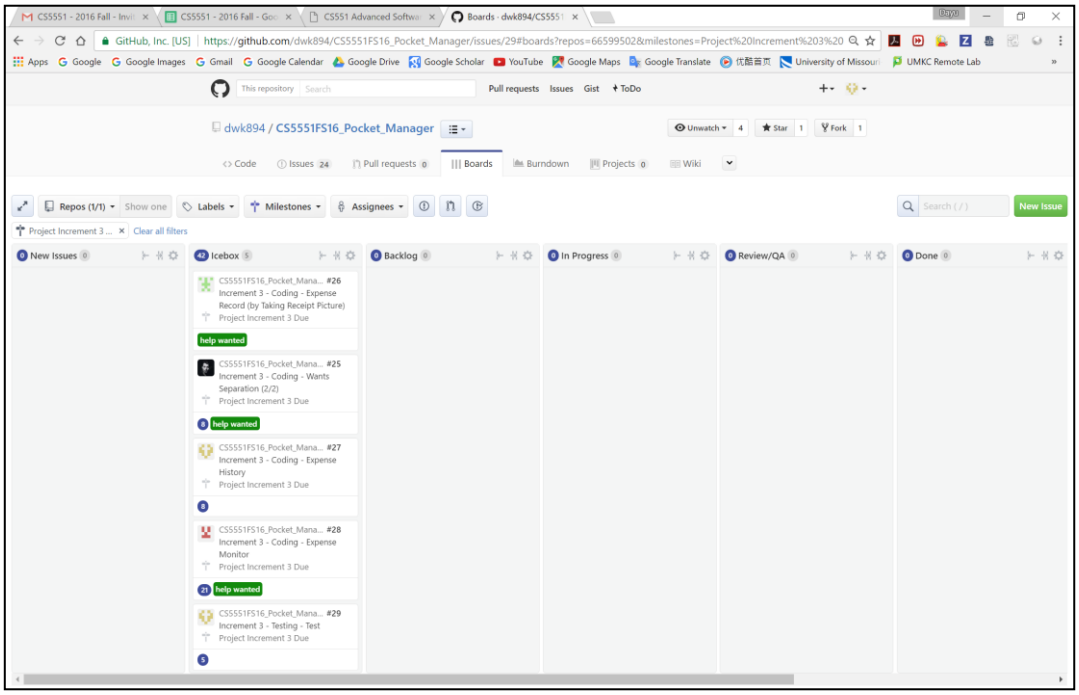
(a)



(b)



(c)



(d)

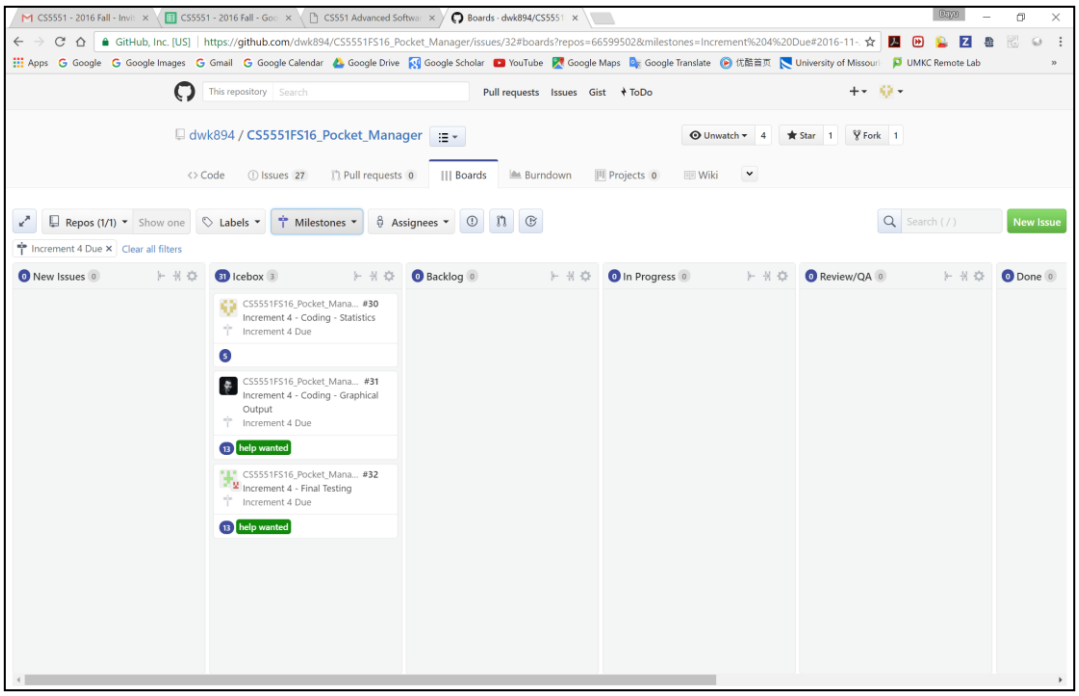


Figure 6. Screenshots from project ZenHub for the four different iterations (*a*: increment 1; *b*: increment 2; *c*: increment 3; *d*: increment 4) of the development of the *Pocket Manager* system. At this moment, most of the future plans are located in the icebox since they could not be started this early. For more detailed information, please look at the project GitHub repository at https://github.com/dwk894/CS5551FS16_Pocket_Manager.

2. Project Timelines, Members, Task Responsibility

Since we tried our best to divide the entire implementation work equally into four independent series and assign one category to each member in the group, it is not considered possible at this moment to build up a realistic timeline for the project development because each member's personal time schedule varies greatly for us. As a team leader (*Dayu Wang*), I respect every gentleman in our team to control his time schedule by himself, with only the due days and tasks were assigned in advance. Instead, we are using the four increments of the project as the time spot to draw something that is similar to a timeline, which is a high-level overview. Nevertheless, the amount of coding work for each body can be summarized based on the estimates of coding workload for each assignment appeared in the ZenHub board (Figure 6). The real timeline for our project is demonstrated in Figure 7.

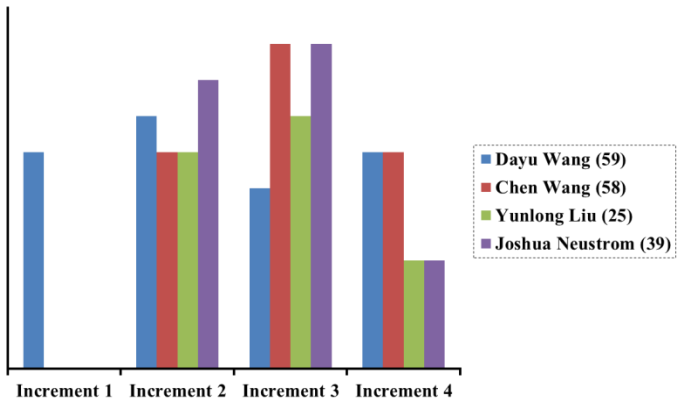


Figure 7. Coding workload for each member of our group amongst the four different project increment iterations. Note that a lower workload of coding does **not** necessarily indicate the lower overall contribution to the entire project, since the project is not just only coding. Also, this is the planned workload for each person and is subjected to change in future.

3. Burndown Chart

Figure 8 is the burndown chart obtained from the screenshot of our GitHub repository.

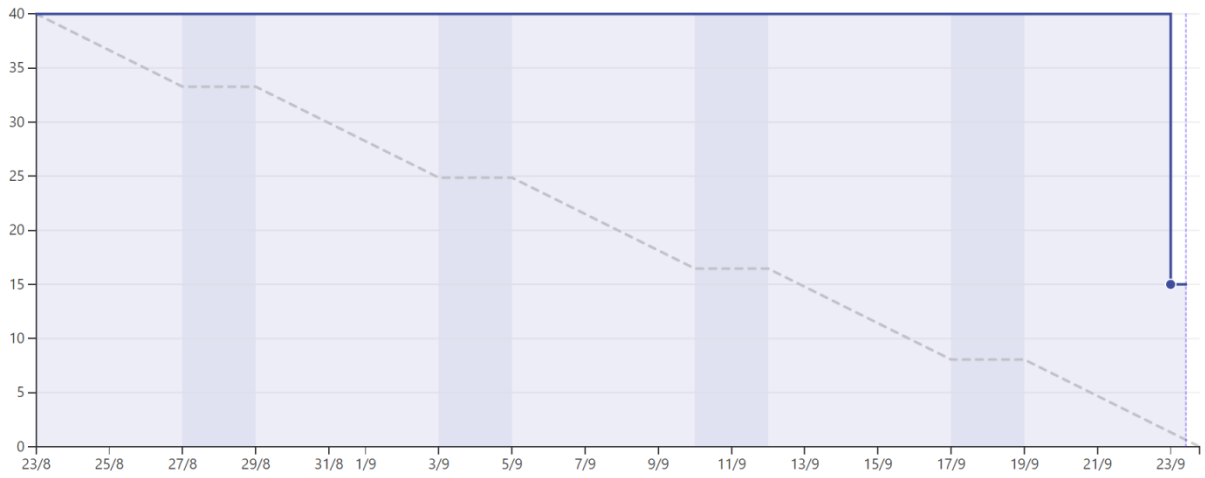


Figure 8. Burndown chart of the project of the *Pocket Manager* system.

- **Existing Services/REST API**

1. Google Cloud Vision - Google Cloud Vision is an API which used the optical character recognition (OCR) technology[3].

The OCR is the mechanical or electronic conversion of images of typed, handwritten or printed text into machine-encoded text, whether from a scanned document, a photo of a document, a scene-photo on an image. For example, it can transform a photo which with a novel in it to a text with a simple novel. OCR technology always is used for the automated recognition of documents, credit cards, recognizing and translating signs on billboards—all of this could save time for collecting and processing data. Figure 9 has showed the OCR technology visually. In general, it needs some step to implement it: first, preprocessing the images. It needs to find the object we need in the picture, such as a novel in the picture, the book on the desk should be select at first. Second is to find the text. Before the transformation, finding the characters is required. Third is the recognition. Recognition and transform the characters in images to the text are the last but most important step.

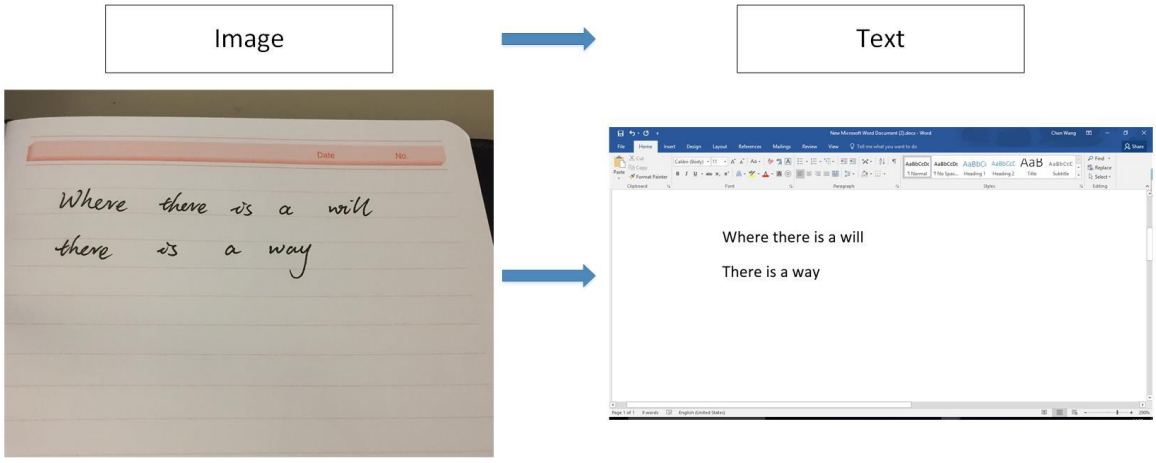


Figure 9. Visual demonstration of an OCR service. OCR is the technology that extracts words from the images and transforms those into editable text.

Google cloud vision is a very good API for the OCR technology, it can detect broad sets of objects in images, which means the cars, trees, sun, birds and so on in images can be detected and return the user as text information. It can also detect inappropriate contents, powered by Google Safe Search, easily moderate content from the crowd sourced images. Vision API enables users to detect different types of inappropriate content from adult to violent content. Google cloud vision even can do analysis the sentiment in the image. Like joy, sad, sorrow and angry can be detected and analysis. It can also detect the label, Logo, landmark, face and image attributes. A matter of a course, the Google cloud can extract the text in images, along with automatic language identification. Figure 10 has showed the functions of the Google cloud.

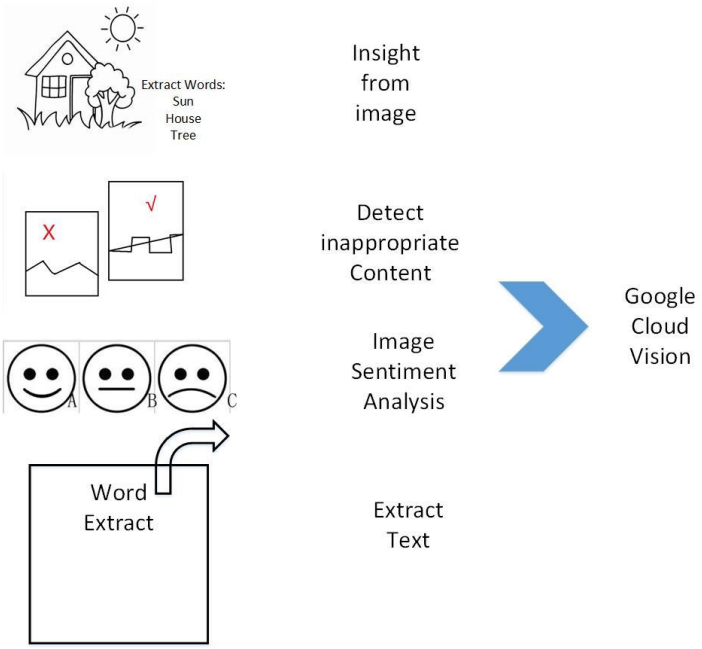


Figure 10. Functions of Google Cloud Vision. It includes insight the objects in images, detect inappropriate content, analysis the sentiment in images and extract text from images.

For our project, receipts can be scan and store automatically. So the camera and the google cloud vision API will be used for take photos for the receipts, extract the details of it and store them to the database.

2. Mongo DB[4]

MongoDB is a free and open-source cross-platform document-oriented database program. It is based on the NoSQL database program, avoids the traditional table-based relational database structure in favor of JSON-like documents with dynamic schemas, then the integration of data in certain types of applications easier and faster. Mongo DB is an open-source, document database designed for ease of development and scaling. The Manual introduces key concepts in Mongo DB, presents the query language, and provides operational and administrative considerations and procedures as well as a comprehensive reference section.

NoSQL is a database type which includes a wide variety of different database technologies. It is based on scale-out architecture using open source software, commodity servers and cloud computing instead of a whole large server and storage. With it, developer can work with applications that create a lot of volumes of new and rapidly changing datasets. It made the development cyclically then the small team can work iterating quickly and pushing code in very short time. And it can access by many devices and scaled globally on amount of users. The NoSQL database type has a lot of benefits, the most important is that it is more scalable and provides superior performance and their data model addresses issues that the relational model is not designed to address. On the large volumes of rapidly changing dataset, it worked very well. And it has quick schema iteration, frequent code push and easy to programming.

Using the database, our expenses recommend and manage system can save a lot of dataset about the users' in it. First of all, the users' account an information of the system will be stored in the database, and they can choose a reasonable plan, input their expense with receipt, and the details of their expense and

plan. All of them will be stored in the Mongo DB. Figure 11 showed the details about the data exchange and store of the users.

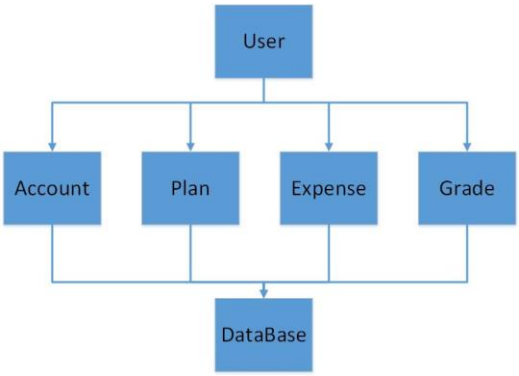


Figure 11. The usage of database. The users' account, plan, expense, and grade have been stored in the database.

3. Highcharts[5]

Highcharts is an application that we can use it to create interactive charts. It can be used for the SaaS projects, web applications, intranets, and websites. It is open source for the users, it based on native browser technologies and using the HTML5, all users can fork them on GitHub and participate in tech discussions. It can support a lot kind of charts for the user such as the basic line charts, area charts, column and bar charts, Pie charts, scatter and bubble charts, combination charts, dynamic charts, 3D charts, gauges charts, heat and tree maps. Figure 12 showed the kinds of charts that Highcharts work.

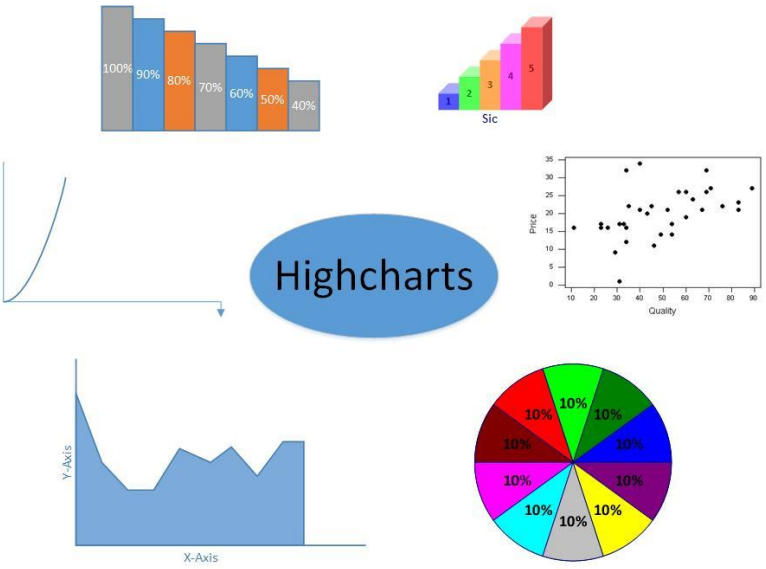


Figure 12. Functions of Highcharts. It supports the line charts, area charts, column and bar charts, pie charts, scatter and bubble charts and so on.

The most popular charts tools are the Highcharts and google charts, on the aspects of usability, setup, maintenance, product directions, google charts is better than Highcharts, but on the parts of meeting

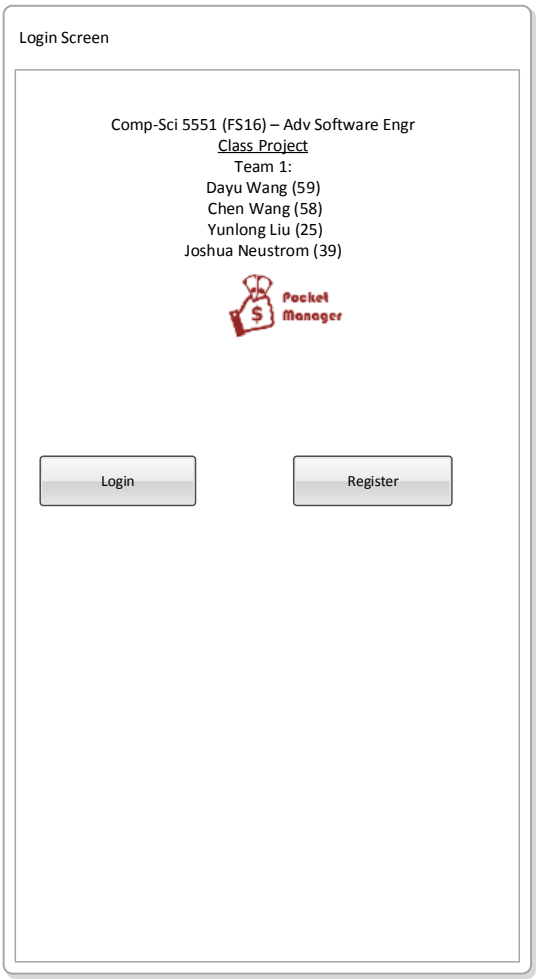
requirement, support, ease of business, the Highcharts get the higher score, so more small business and individuals choose the Highcharts for their project. A lot of people think that the Highcharts is more users friendly; it is easy to use and produces great results.

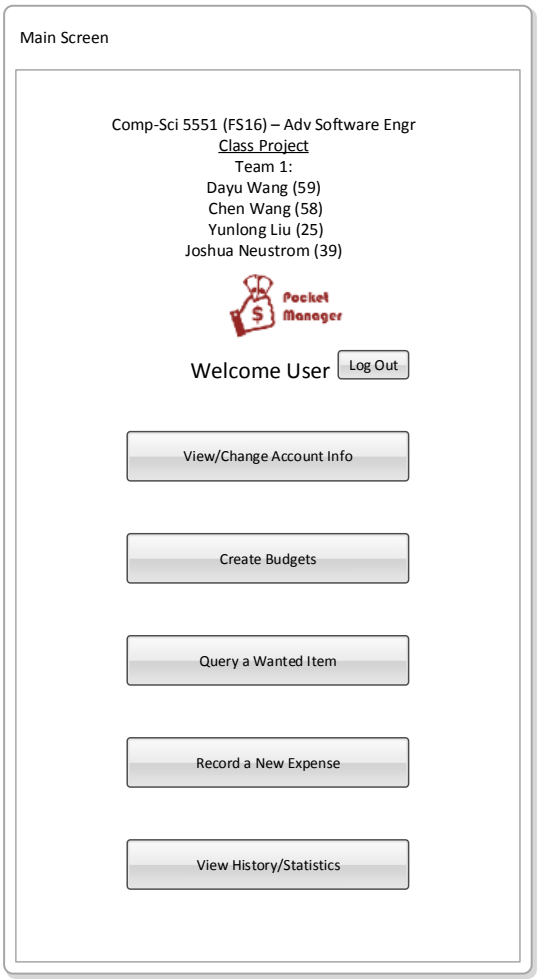
For our project, when the user chooses a plan, and when they review their expense in one mouth, the chart will be posted for the user then they can see their expenses in different parts visual. Then they can pay attention and improve their expense plan.

- **Detail Design of Features**

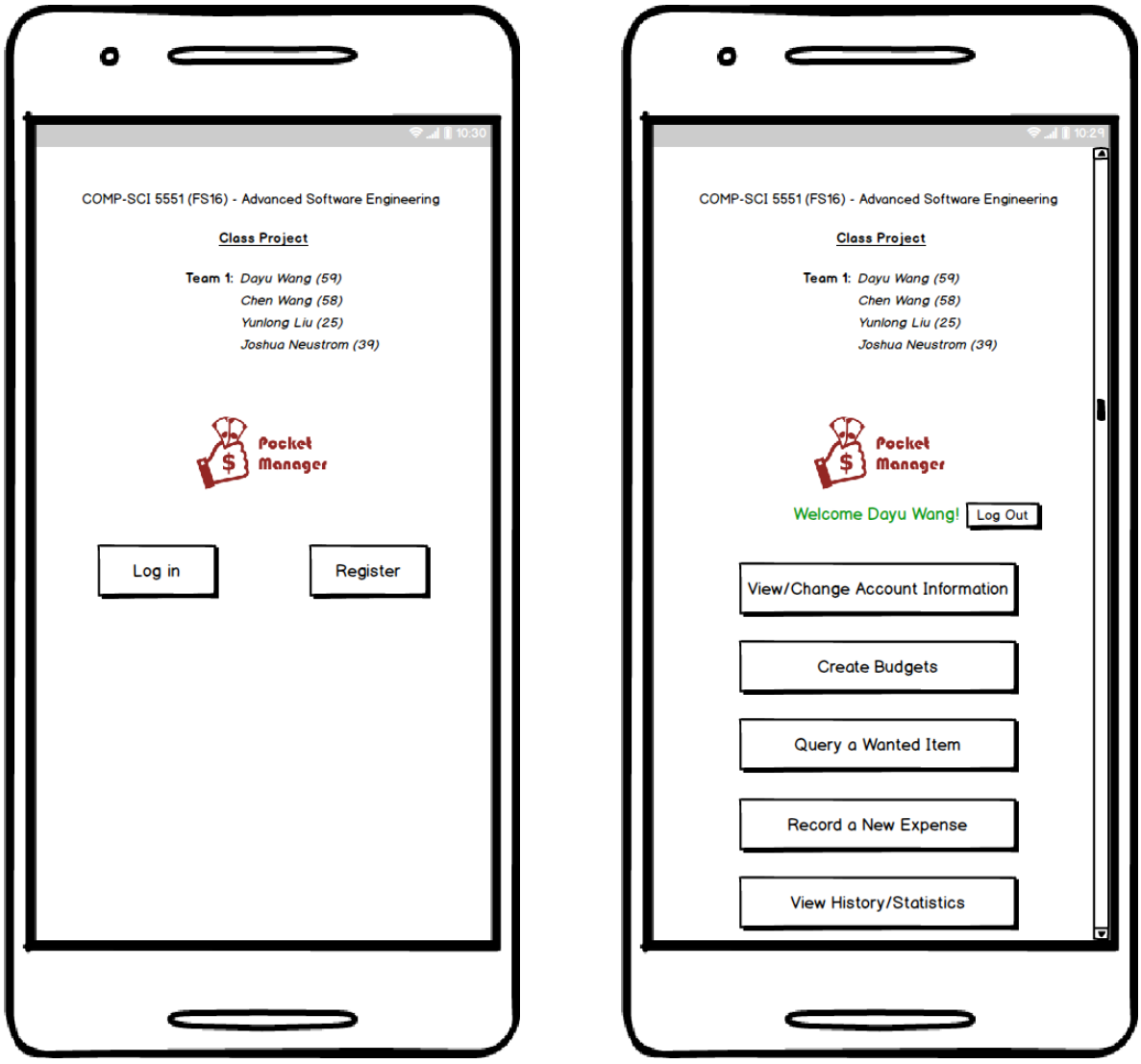
1. Wireframes and Mockups

The high-level wireframes and mockups of the *Pocket Manger* system to be developed are depicted in Figure 13, which act as the shell of our entire system that shows the main components in our system. For each smaller functional portion of the system, in future, whoever is assigned to a feature should design the UI of the feature by himself.





(a)



(b)

Figure 13. Wireframes and Mockups of the designation of user interfaces in the *Pocket Manager* system. These pictures are the shell model for our system. The further details of the UI of each feature will be designed by whoever assigned for that feature during the implementation process.

2. Architecture Diagram/Sequence Diagram/Class Diagram

Software architecture is defined by New York University (NYU) as the structure of structures for a system[6]. The diagram outlines a top level view of key systems utilized such as software frameworks and API services with basic connections on how they interact.

For the *Pocket Manager* system, the architecture covers the native application, phone hardware, API services, and remote database. Ionic was the selected Framework for the native application which uses HTML5 and AngularJS to build Hybrid Applications. Major functions include the login, budget creation, scanning receipts, recording payments, fetching statistics, graphing, and budget creation. Besides hosting the actual application, the phone provides camera, screen, and data connection. Various API provide data to the application including the Google Sign In which provides authentication and user information with