**HandyEat**

Group Project Report

E.C.W

SEIS 610 - 02 Software Engineering

University of St. Thomas

**Content**

Background 3

Design and Functionality 4

Plan & Execution Progress

Team Member’s tasks & work log

Evaluation of Tools & API

Conclusion

**Background**

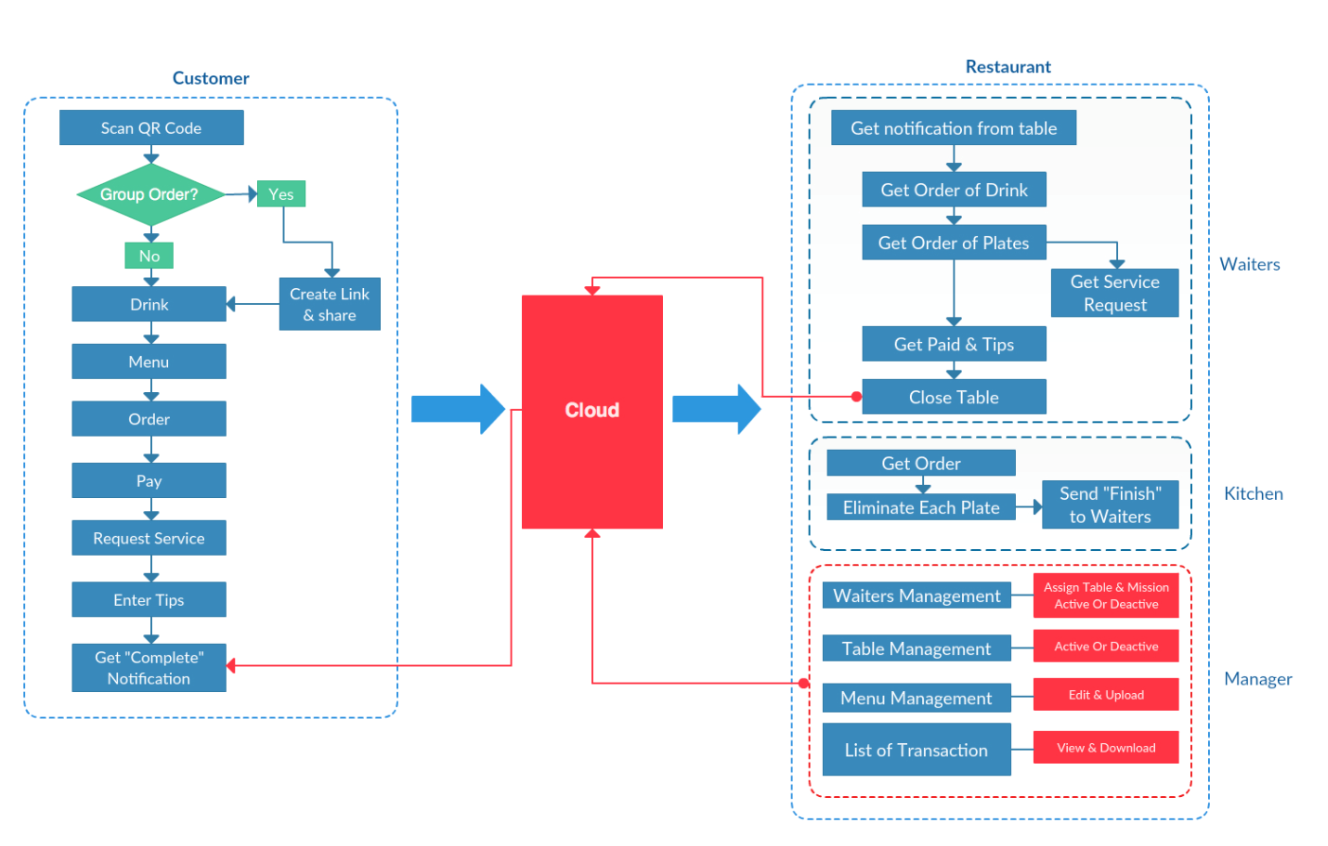
**ProcessOn**

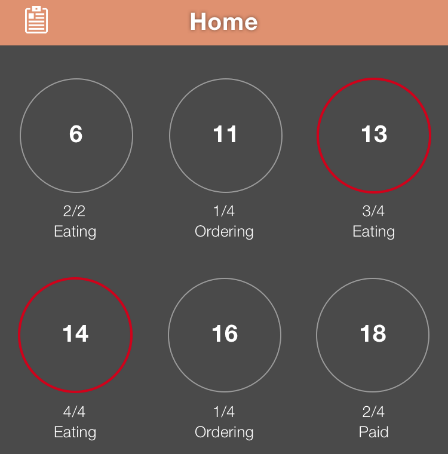
When making software products, we often need to make visuals to go along with other materials we present. A simple flowchart can help people explain to team members or customers the software’s structure. We planned to use ProcessOn as our flowchart maker, not only because the final chart is beautiful and clean, but also it is very easy to use and learn.

ProcessOn comes with tons of shapes and templates. They are sample versions of the visuals we are likely going to create. They feature drag-and-drop functionality that makes it easy to rearrange the shapes and connectors on the canvas. And because it's online, there's no need to worry about whether all the collaborators are running macOS or Windows. ProcessOn simply runs in the browser.

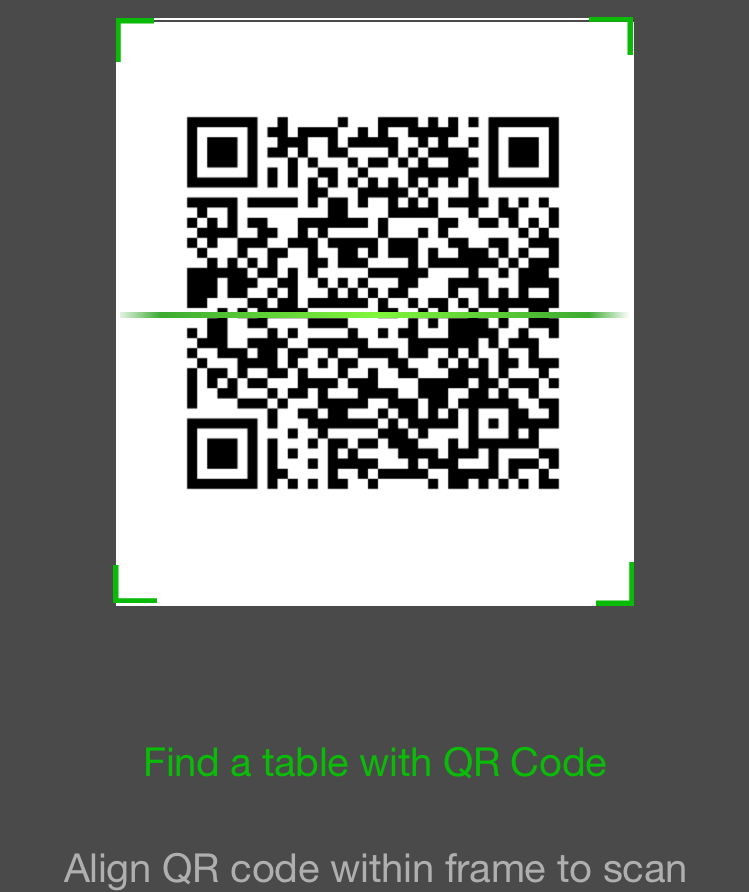
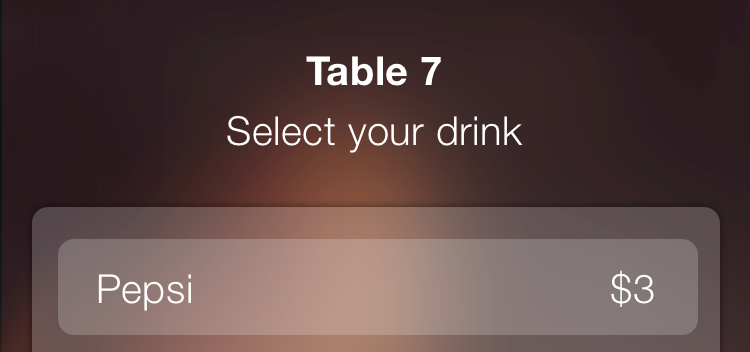
**Design & Functionality – Yu, Will**

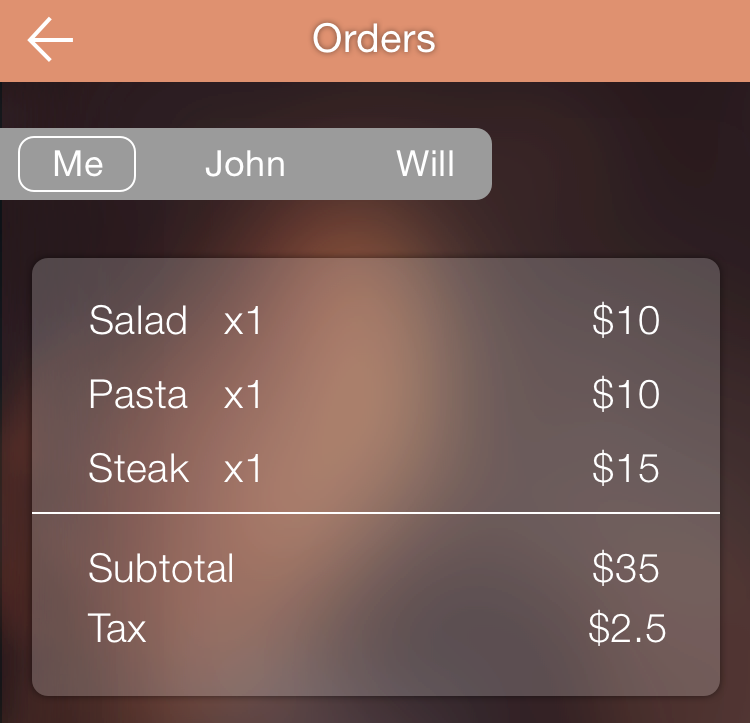
Flow Chart of HandyEat App consists of three major parts: customers-end, cloud (firebase) and restaurants-end. All data will be sent to and saved in cloud and notification will be sent to either customers-end or restaurants-end for actions needed. Below are details about how HandyEat App works from flowchart perspective.



All restaurants registered in HandyEat App are assigned QR codes for all tables. Customers are able to book available tables in a specific restaurant to request dine-in service via scanning QR code. Once QR code has been scanned, data will be sent to cloud and restaurants will get updated.

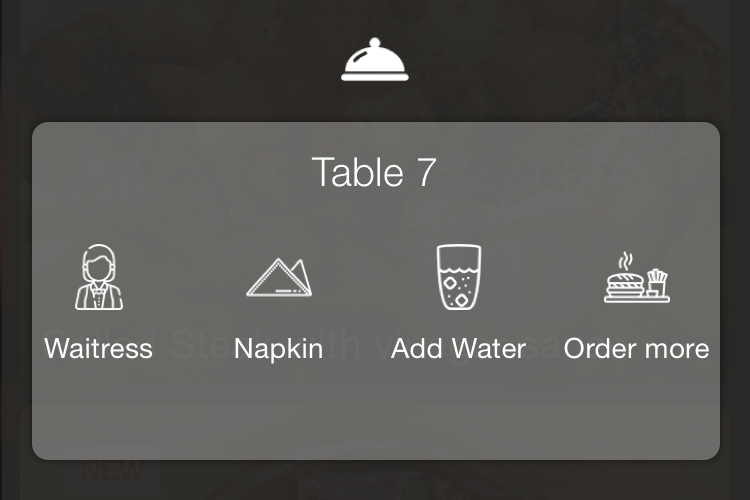
Managers will activate tables and assign waiters correspondingly. There is also an option to create order link and share with others for group orders. Once everything is set, customers are able to place order.



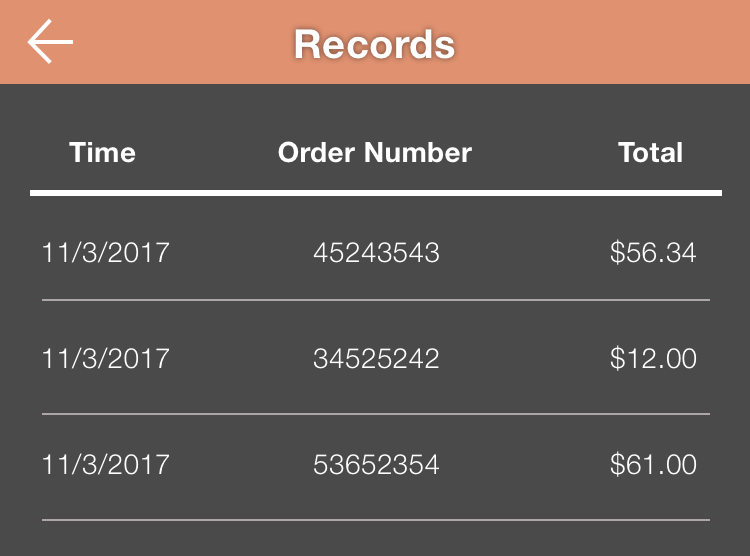
Customers order drinks first and data will be sent to cloud. Waiters will receive drink orders notification from cloud and then bring drinks and sets of tableware to customers. While waiters are preparing drinks and bringing to customers, dishes menu is displaying for customers to place food order. Customers who sitting on the same table will also allow to check friends’ dishes from our app directly.

Data will be sent to cloud once dishes have been placed and payments have been paid successfully. Waiters will get notification of payments successfully received and kitchen will receive food order information.

Once all dishes have been completed, Kitchen will send notification to waiters through cloud. Waiters will bring dishes to customers. During the dine-in time, customers can always request services via HandyEat App and corresponding data will be sent to cloud, and then to waiters.



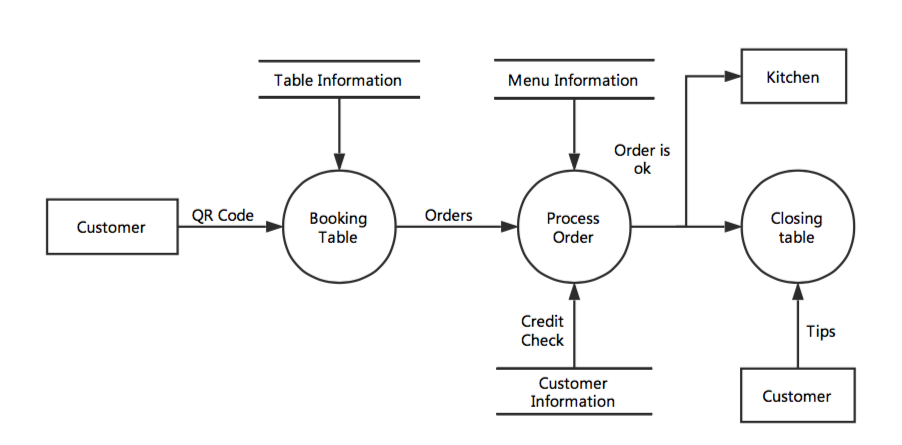
Once customers finish eating, customers can grant tips and tips information will be sent to waiters via cloud. Customers will confirm if the dine-in service has been completed on their end. Once the “completion” notification has been generated to send to cloud, waiters will be notified to clean table. Waiters will send closing table notification to cloud once the clean-up is completed. Managers will then change table status from deactivated to active for further dine-in service.

In addition to waiters and tables management, managers are able to revise menu and view all transactions.

**Evaluation of Tools & API**

**ProcessOn**

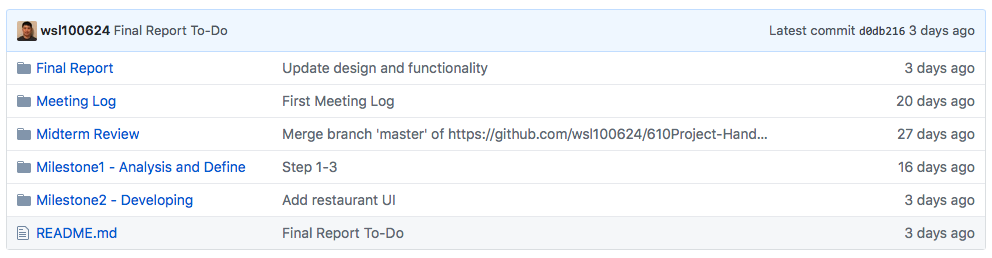
When making software products, we often need to make visuals to go along with other materials we present. A simple flowchart can help people explain to team members or customers the software’s structure. We planed to use ProcessOn as our flowchart maker, not only because the final chart is beautiful and clean, but also it is very easy to use and learn.

ProcessOn comes with tons of shapes and templates. They are sample versions of the visuals we are likely going to create. They feature drag-and-drop functionality that makes it easy to rearrange the shapes and connectors on the canvas. And because it's online, there's no need to worry about whether all the collaborators are running macOS or Windows. ProcessOn simply runs in the browser.

**GitHub**

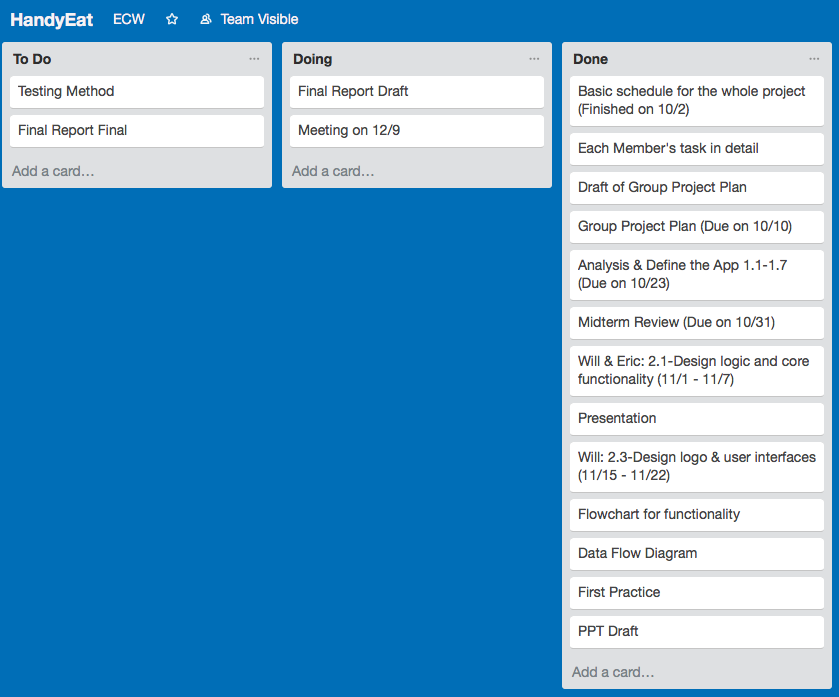
We have been using GitHub all the time during this project for documentation and communication, because of these three features – fork, pull request and merge.

GitHub is a Git repository hosting service, but it adds many of its own features. While Git is a command line tool, GitHub provides a Web-based graphical interface. It also provides access control and several collaboration features, such as a wikis and basic task management tools for every project.

The flagship functionality of GitHub is “forking” – copying our project repository from one team leader’s account to another. This enables our team members to take a project that they don’t have write access to and modify it under your own account. If anyone makes changes and like to share, they can send a notification called a “pull request” to the team leader. Team leader can then, with a click of a button, merge the changes found in that team member’s repository with the original repository.

Trello

Trello is our main project management tool. It is a collaboration software that helps members to discuss a project in real-time with very good drag-and-drop capabilities.. It is an easy-to-use web application that were designed to facilitate project delivery and make it more efficient.

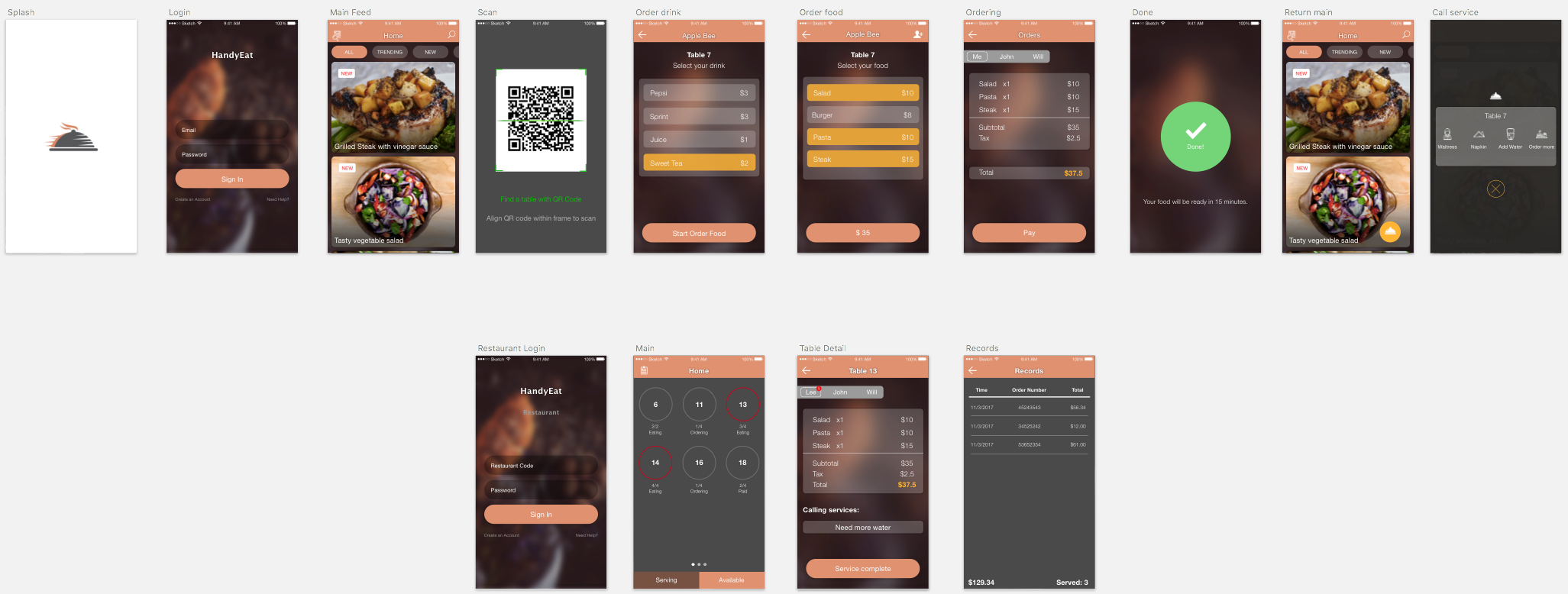
While using Trello, we can set more realistic milestones and deadlines, tie up and automate interdependent assignments, and keep all of other team mates focused on each milestone of this project. It's a reasonably lightweight, flexible, and focused on collaboration tools.

Every card on a Trello board has to be in a list, so before making any cards, we set up the workflow for our board as To Do, Doing and Done. The best thing about cards is that they allow us to talk about the task at hand. We can mention any teammates in a comment by typing “@” and their username. Then all pertinent communication stays with the task, and anybody can pick up the work from there.

**Sketch**

Sketch is a design tool entirely vector-based and focused on user interface design. It’s easy to pick up and costs a fraction of the price of Photoshop. And because it comes with 7 days free trail, I can finish our app’s UI and logo design without any cost.

Because of its simplicity, anyone with little to no training can learn Sketch. On top of that, it’s perfect for today’s need for designing for multiple devices. As a result, we found Sketch far more approachable because it doesn’t get in the way of producing simple designs.

Sketch comes preloaded with a number of user interface templates for iOS. The iOS and Material Design templates are particularly comprehensive. They’ll be a great starting point for any designer, beginners and experts alike.

**Firebase**

We planed to use Firebase as our backend server, which also act as our datastore. The biggest features on Firebase is Realtime Database. It’s a database that lets you store and sync data between your users in realtime.

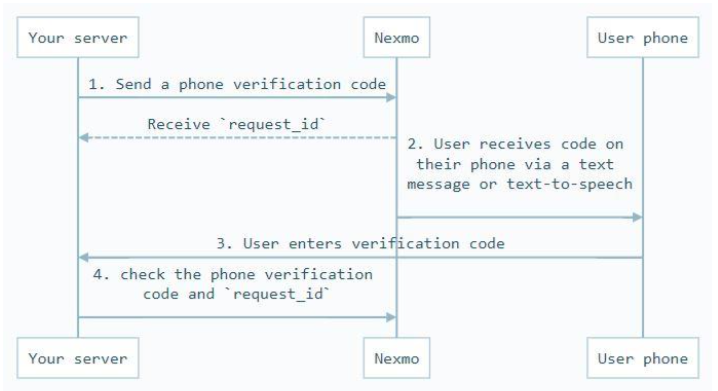
We found this feature is very useful for our app is because restaurant-end may change food’s picture or price frequently. Based on the size of these image, we really need a database that can change and store data instantly. The connection between app and Firebase is WebSocket, instead of normal HTTP calls. According to research, WebSockets are much faster than HTTP. Developer doesn’t have to make individual WebSocket calls, all of our data syncs automagically through a single WebSocket as fast as our network can carry it.

**Stripe**

Stripe is a credit card processing system which enables you to receive payments from your customers, without the need to set up a Merchant account with your bank.

We decided to integrated Stripe in our app is because it’s user-friendly API. The documentation is clear and concise. If we sign up an account, the documentation is customized to our profile. Because different account come with different tokens. So we can literally copy and paste any code in directly to IDE and then see the result instantly. Additionally, Stripe supports several programming languages. These features will increase the efficiency of our app’s development process.

**Nexmo**

We plan to use Nexmo on a daily basis to send Promo and Transactional SMS to our Client's customers. When user create an account, or sending invitation to friends, we will also need Nexmo to generate verification code and send text messages. The real time APIs enable a seamless connectivity between our systems and Nexmo.