National College of Ireland
BSc in Computing
2016/2017

Declan Casey
13564737
Declan.casey@student.ncirl.ie

WaiterAider

Technical Report

National College of Ireland

Table of Contents

Executive Summary

The purpose of this document is to detail the capabilities and functionalities of my software system.

My project idea is to create a piece of software that would improve the productivity and reduce costs in a restaurant setting.

Desired Outcome:

The desired outcome of this software would be to replace the job of the waiter taking the customer's order and bringing that order to the kitchen.

By taking this role away from the waiter and having them being responsible only for delivering food to the customer's tables, there wouldn't need to be as many waiters in a restaurant which would reduce costs.

Productivity would also be increased as orders are sent almost instantly to the kitchen once placed through the mobile application.

Stock management is also one of the main features of the application. As customers make orders, the stock required to complete these orders would be deducted from the total count which would give real time information to the staff. The staff also have the ability to manually update the amount of stock by adding/deducting their chosen amount from a particular item's stock count.

1 Introduction

The project is a software system which could be used in any restaurant as a replacement for paper menus and waiters taking orders. I propose that this could be accomplished by having the whole process done through the Android application.

The customer would create and send the order directly to the staff in the kitchen themselves through the application.

1.1 Background

When completed and properly marketed this piece of software could have a large impact on the restaurant industry. The application could create the process of making an order more efficient by handing over the responsibility of making an order and sending that order to the kitchen over to the customer.

After working in a catering setting myself, I have been given an insight as to how the process of ordering food and keeping track of the stock count works which I believe has given me an advantage in approaching the development process.

I have worked in a retail kiosk in the Aviva Stadium as a cashier. Through working there I got a feel for how orders are made to chefs and food preparers (usually by continuously cooking items before they are ordered and by shouting the order towards a nearby chef which is inefficient as customer's often had to wait several minutes for their order).

I also gained an insight into how the stock count is kept by manually updating each item in the stock at the end of each shift into a computer system. In my opinion, this was not a good system as the counting of stock could be inaccurate due to human error.

1.2 Aims

The aims of this software are to create an android application that can be used by customers and staff to manage order creation and completion for a restaurant.

Below is a list of each individual aim:

- Allowing customers to be able to navigate through the different dishes and beverages in the menu and add items to their order before confirming their order at the end.
- The order to be sent to the staff Android devices in the kitchen once it is confirmed.
- Allowing the staff to update the status of the order and set it as complete once it has been paid for.
- Allow the orders to be viewed on a digital display which is connected to a Raspberry Pi. The web application can be seen on any device with a web browser but for the purposes of this system the Raspberry Pi is ideal as it is lightweight and portable, and the web application does not require a lot of hardware resources.
- The staff will also be able to edit the menu such as uploading new images for different meals etc.
- From creation of order to delivery of food to take less than 23 minutes, the average waiting time. [1]

1.3 Technologies

The technologies that will be used in this project are Android devices, the Raspberry Pi, Android Studio, Java/XML, HTML & JavaScript and the Firebase Server.

Android Devices – These are devices with Google's Android operating system that allow for Android applications to be installed and run on. This (my phone and tablet) will be used to run and test my Android application

WHY? – As of 2015, the Android operating system is used by 80.7% of smart phones in the world. I have also begun learning how to code for Android devices in college this year. [2]

Raspberry Pi – This is a miniature computer that requires minimum setup and wiring which means that it can be set up almost anywhere. It will be used to display the customer's orders on a digital screen.

WHY? – It is more portable than a PC as it is smaller. It can fit somewhere small in a kitchen and not be in the way as much as a PC would be. Alternatives to this could be an Intel Edison or even a PC if there is room.

Android Studio – This is an IDE (Integrated Development Environment) which my Android application will be developed with.

WHY? – This is the IDE used in class to develop Android applications.

Java/XML – These are languages used to create android applications. I will be using XML to code layouts in my application and Java classes will be used to add functionality to these layouts.

HTML & JavaScript – These languages will be used to build the single-page web application.

WHY? – I have used these occasionally throughout my time in college. Sample code for extracting information from the Firebase server is written in these languages.

Firebase Server [3] - This is the server that the software system will use to store and send information. It will also be used or hosting the web application and authentication in the Android application.

WHY? – This server/database can be easily integrated into an Android application as there is an easy setup process built into Android Studio in order to attach it to your Android project.

1.4 Structure

2: System - This section will briefly detail both the functional and non-functional requirements for the software project. The section will also detail the design and architecture of the system, the testing methods that have been used throughout development, and the implementation of certain functionality (how they were

coded); this includes classes, data structures and algorithms used etc. GUI screenshots will be included to further detail the system.

3: Conclusions – This section will detail the advantages, opportunities and disadvantages of the software system.

4: Further Development or Research - This section details what could be done with the software system with further development on it after the proposed functionality has been completed.

5: References – This section contains all reference material used to construct this document.

6: Appendix – This section contains the project proposal, project requirements, monthly report documents and a user manual which add further information to this document.

2 System

All components of the system that will interact with one another include one or more Android applications, one or more Raspberry Pis, the Firebase server and the web application.

The sections below will go into detail on how these components will interact and the functionalities available in the system.

2.1 Requirements

This section will give a brief explanation of the following requirements.

2.1.1 Functional requirements

1. View Order - This requirement details the process of viewing a created order. This is the most important requirement in the system as without the possibility to view orders there would be no information to work with to complete many of the other requirements. The customer will be able to view their own orders and the staff will be able to view all orders.
2. Place/Edit Order - This requirement describes creating an order by the customer that will be sent to the kitchen. It also details the process of editing a previously created order by either removing items from their order or adding new items.
3. Complete Order – Orders are set as complete by the staff. They will be able to view the customer's orders and set them as complete once the food/drink has been sent out to the customer and paid for.
4. Check Stock - The process of checking the stock is carried out by the kitchen staff. This functionality offers an easier solution to managing stock than to manually count it as the stock would need to be constantly counted whereas the app will keep track of the stock numbers as they are used.
5. Update Stock - This requirement details the process of updating the stock levels by the manager/admin. As new stock arrives and food expires, the stock numbers will need to be updated manually which this functionality will allow.

6. Sign Up/Log In/Sign Out – This functionality allows the staff user to sign up/log into the application and sign out by the staff. Each staff member will have their own log in details. The staff will log in the customer devices at the start of each day and assign a device and table number to them. They will also be able to create a new account through a sign-up process.
7. Update Order Status –The staff users will have the ability to update the status of the customers' orders to keep the customer updated on the stage of their order. These updates include information such as [Preparation Stage], [Cooking Stage], [On the Way] etc.
8. Edit Menu - This requirement allows the staff user to upload new images for different meals etc.

2.1.2 Data requirements

Database [4] – A real-time database will be setup with live data to store all required information so that it can be retrieved. This information includes login details, orders, menu information and stock count. Firebase database will be used for storing orders/stock and Firebase Authorisation will be used to hold user login information. A non-final version of the database layout can be seen in *Fig 2.1.2.1*

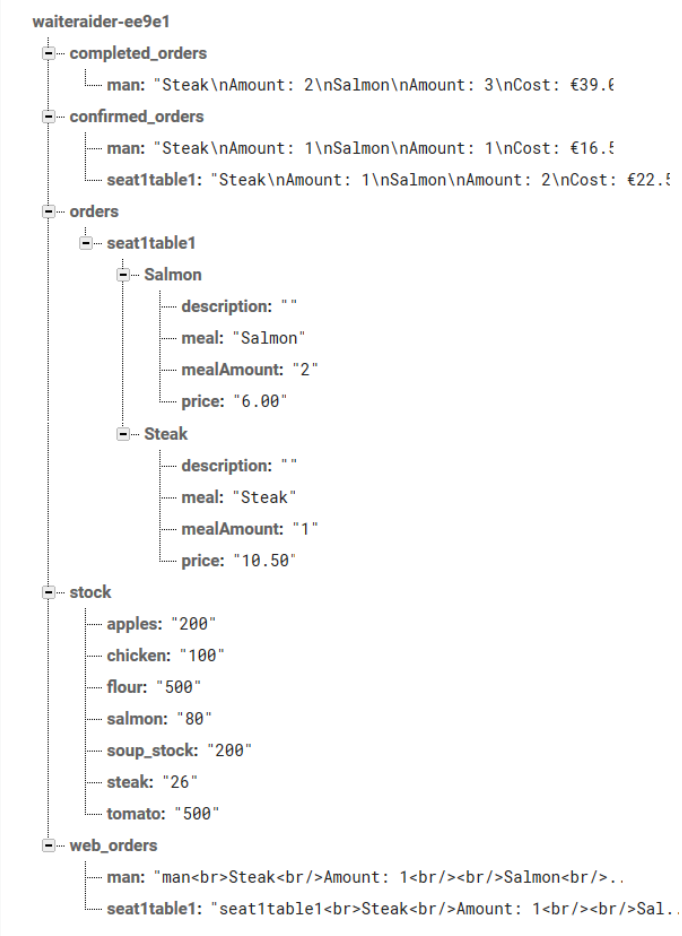


Fig 2.1.2.1 Structure of the Database

Backup and Recovery – The project's development will use version control by committing and pushing the project code to GitHub every time a change is made or a requirement added. If the project files are lost, corrupted or contain a new error that cannot be reversed, I will be able to download the backed-up files from GitHub.

When working on a new functionality I will checkout to a new branch so that I will be able to test it without breaking the whole project and needing to revert the changes. I will then be able to merge with the master branch once the functionality in the branch is complete and error-free. I will repeat this process until the project is complete.

2.1.3 Accessibility requirements

Accessibility – The app should be easy to use, easy to read, have intuitive UI and be able to be used by people who are not tech savvy.

2.1.4 Environmental requirements

Dry Environment – The use of this software is not suitable for an environment that is prone to getting wet as the devices could malfunction and render the app unusable. The recommendation is for the devices supplied by the restaurant to be waterproof as spillages often occur.

Constant Power Supply and Internet connection - In order for the software system to work correctly there will need to be a constant Wi-Fi connection within the restaurant so that the devices can communicate with the server. If there is no Internet connection, orders will not be able to be made. There will also need to be a constant power supply for the RaspberryPi.

2.1.5 Hardware requirements

For this system to work as intended, the restaurant should provide customers with a tablet with which they will make an order, provide staff with either a tablet or smart phone (or let them use their own), have one or more Raspberry Pis along with a keyboard and mouse in order to operate them.

2.1.6 Usability requirements

The following usability requirements from usabilityfirst.com [5] are good examples of what I have aimed for in my project:

- Efficiency of use: goals are easy to accomplish quickly and with few or no user errors. I believe this is true for my application as adding items to an order and confirming that order can be done in less than 10 clicks.
- Intuitiveness: the interface is easy to learn and navigate; buttons, headings, and help/error messages are simple to understand. All parts of my Android application's GUI will have a help button that the user can click to receive

guidance information for the part of the application that they are currently navigating.

- Low perceived workload: the interface appears easy to use, rather than intimidating, demanding and frustrating. To address this issue I tried to make the application easy to understand by having the functionality controlled mostly through button presses and designing the layout so that it could be used without any prior instructions.

2.2 Design and Architecture

The system architecture for this system is detailed as a many-to-one-to-many relationship from one android device to the server and back to another android device. The Raspberry Pi has a many-to-one relationship with the Firebase server as more than one Raspberry Pis could be used.

The server will act as both a real-time database and a service for sending information between devices. The connection and data transmission will be made between these devices through the Internet.

A diagram of the system's architecture can be seen in *Fig 2.2.1.*

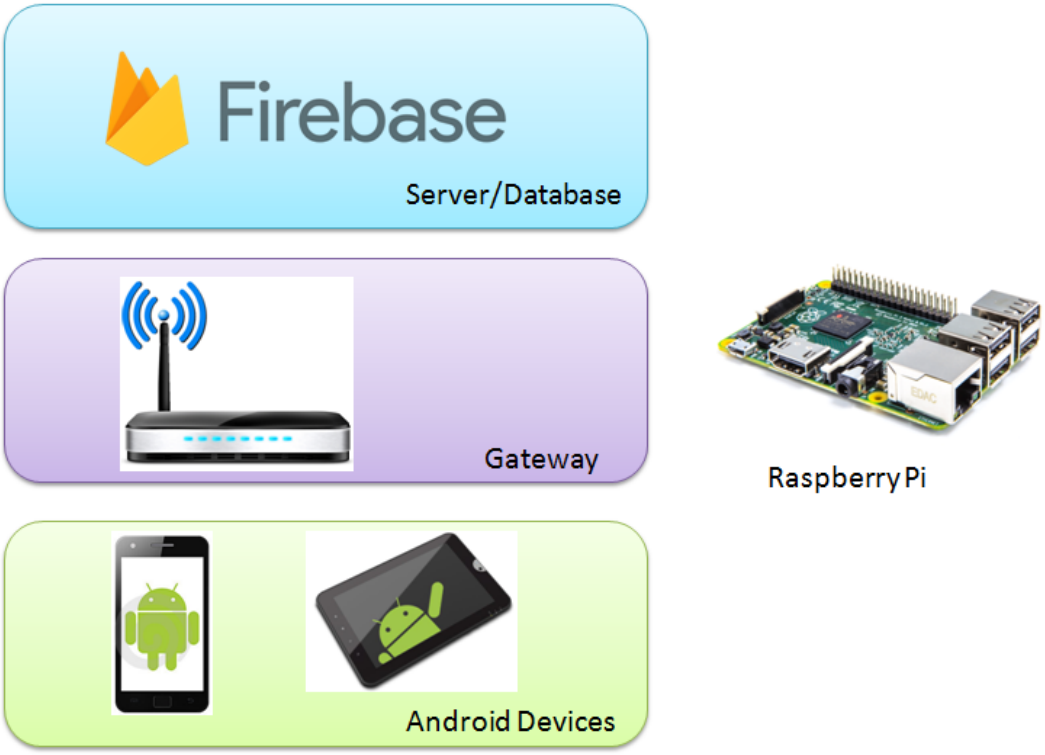


Fig 2.2.1 System Architecture Diagram

2.3 Implementation

Looking at the system at a high level; confirmed orders and orders that are in the process of being confirmed are stored in the database and given a unique identifier (the current user's username which is the seat and table number). This ensures that more than one user can create orders as they would be overwritten otherwise.

Once they have been confirmed, they are available for viewing and updating by the staff users. If the order is set as complete by staff it will be removed from the application and sent to a separate "completed orders" section of the database. If the staff user decides to update the order, it will stay in the application and the update will be sent to the customer's application.

Looking at the system at a low level I will explain how certain functionalities/features are implemented:

Menu – This feature lists the courses and meals/drinks available to the Customer. This is implemented by using an ExpandableListView [6]. An ExpandableListView is a View with a list of items with one or more sub-items. Each item in the list can be expanded or collapsed. Each sub-item in the list represents a meal and when clicked on the application navigates to the layout of that particular meal. This functionality can be seen in the code snippet in *Fig 2.3.1*

```
// for clicking on children of the list view
ExpList.setOnChildClickListener(new ExpandableListView.OnChildClickListener() {
    @Override
    public boolean onChildClick(ExpandableListView parent, View v, int groupPosition, int childPosition, long id) {
        TextView mealTV = (TextView) findViewById(R.id.mealTV);

        if(childPosition == 0 && groupPosition == 2){
            Context context = v.getContext();
            Intent myIntent = new Intent(context, SteakActivity.class);
            context.startActivity(myIntent);
        }
```

Fig 2.3.1 ExpandableListView sub-item click functionality

Deducting Stock on Order Creation – When a meal is added to the order of a customer its associated ingredients are deducted from stock count and if the customer decides to delete that item from their order, they are added back to the count. In the below code snippet in *Fig 2.3.2* you can see how the stock reference of the Firebase stock table is updated when adding a meal to the order.

```
public void onDataChange(DataSnapshot dataSnapshot) {
    String stockSteak = dataSnapshot.child("steak").getValue(String.class);
    int amountInt = Integer.parseInt(steakNum);
    int stockInt = Integer.parseInt(stockSteak);
    //deducts the number of steaks added to the order from overall stock
    int newStock = stockInt - amountInt;
    String newStockS = Integer.toString(newStock);
    //updates stock number
    stockRef.child("steak").setValue(newStockS);
}
```

Fig 2.3.2 Deducting meal's ingredients from the stock count

2.4 Complexity and Difficulties

Complexity:

The main complexity in this project was understanding how the Firebase system worked and how to structure the database. As there is plenty of documentation

available on how to integrate Firebase into an Android application, the main complexity wasn't integrating it but more how and what to use to meet the needs of my system. This included how to reference my database; its tables, its parent and children keys and values. Another important factor was to have unique values for orders to enable multiple users to use the same system at once without overwriting another's order.

Extra complexity in the project included having two different user sides to system (customer and staff). A user would be given access to one side or the order depending on their authentication. If a user signed in with a username with "staff" in it, they would be redirected to the staff side of the application otherwise they would be redirected to the customer side. As all user accounts are created by the staff there are no problems with customers being able to access the staff side as they are already logged into the application when they arrive and are not able to sign out.

Difficulties & Research:

When I started coding the prototype I had no idea how I would design the application. I started by experimenting with the master/detail flow [7] to present the layout of the restaurant menu. I could not understand how to fully implement this into my application so I decided to scrap the prototype and start from scratch.

At the start of development I did not understand the capabilities of Firebase. Before I had implemented it into my project I was trying to implement the android put extra [8] [9] method to store order information locally on the device which I had a lot of problems with.

The difficulties that I have experienced throughout the development of this software system include developing an Android application which can communicate with both a server and a RaspberryPi. As this was the first academic year that I have worked on android, I had to learn a lot about how to develop them proficiently as well as how to integrate other frameworks into them, such as the Firebase server.

Other difficulties include allowing the user to edit the contents of the restaurant's menu and creating a single-page web application with Firebase so that the customer orders can be seen on the Raspberry Pi via a web browser. I had originally planned to create an application for the Raspberry Pi itself through Python but felt it unnecessary as the web application was less complex to setup with Firebase's Hosting tool. [10]

Important inspiration for the end result of my menu came from the following sources [11] [12].

2.5 Graphical User Interface (GUI) Layout

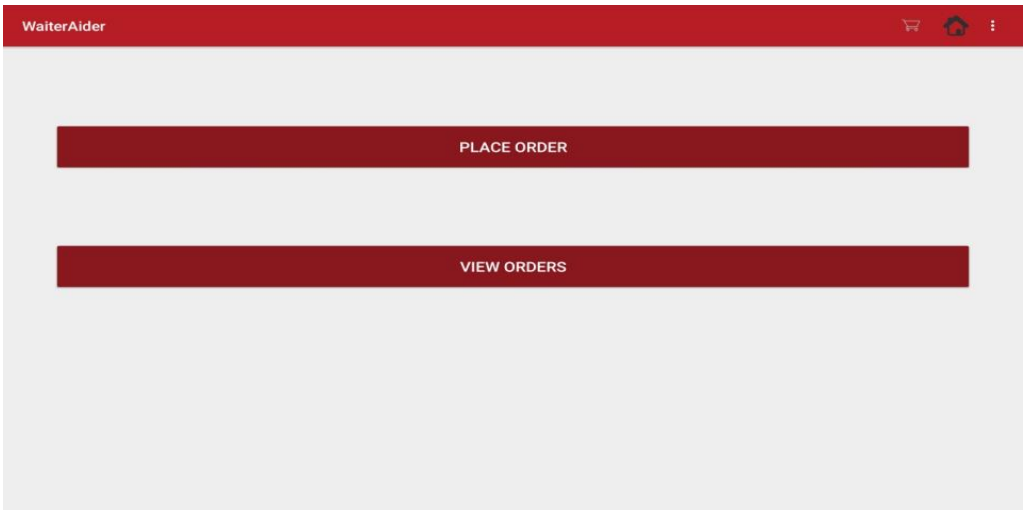


Fig 2.4.1 Customer Main Menu

The screenshot in *Fig 2.4.1* represents the main menu for the customer. It contains 2 buttons, the place order button will bring the user to a list of available meals, and view orders will bring the user to a list of their created orders.

Fig 2.4.2 Collapsed Menu

The screenshot in *Fig 2.4.2* represents the list of courses that the customer can choose from. By clicking on one of the courses, a list of meals will be expanded.

Fig 2.4.3 Expanded Menu

By clicking on the name of a meal they will be brought to a screen where they can view an image of the meal, its details, cost and they will be able to add up to 5 servings of that meal to their order. They can also remove the item from their order. See *Fig 2.4.4.*

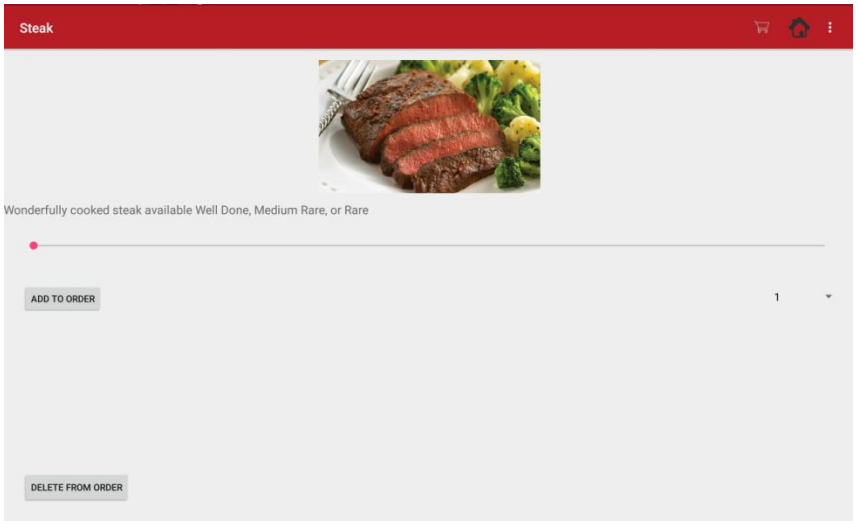


Fig 2.4.4 Meal Layout

Once the customer has selected all of their meals for order, they would click the cart button on the Toolbar which would bring them to the layout seen in *Fig 2.4.5.* There they can see all of the meals in their order, the total cost and they would be able to confirm their order as long as they don't currently have an order made.

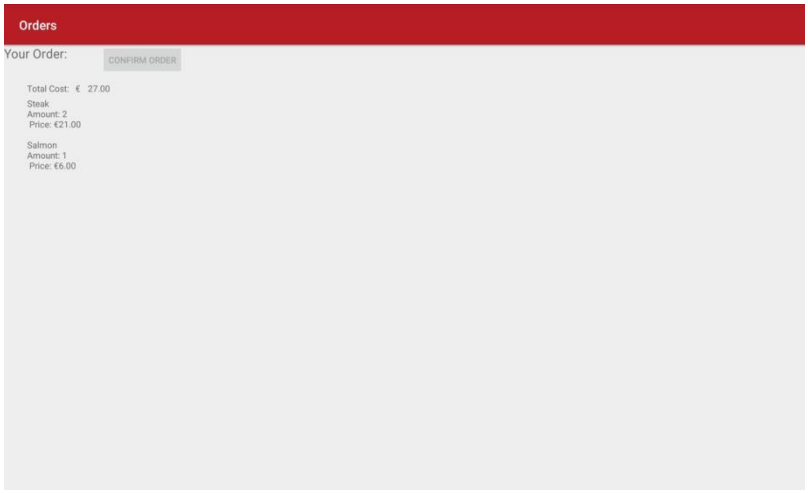


Fig 2.4.5 Order confirmation layout

The customer can also view the orders they have made as seen in *Fig 2.4.6.*

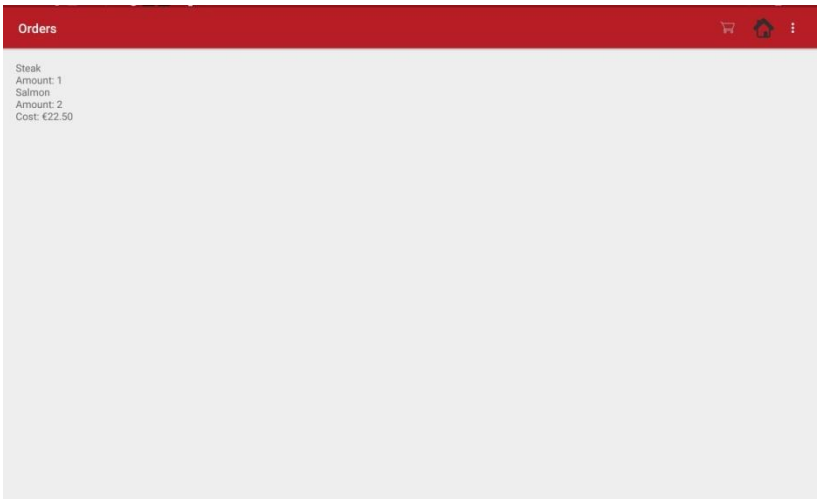


Fig 2.4.6 Orders made layout

The login section of the application can be seen in *Fig 2.4.7.* The layout for the sign up section is almost identical. To sign out of the application the user would click the Sign Out button in the collapsed section of the toolbar (the 3 dots). Only a staff member will have access to signing out.

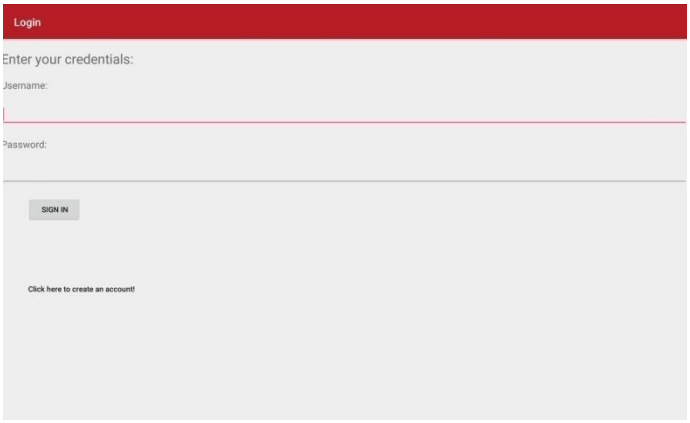


Fig 2.4.7 Login layout

The main menu for the Staff user can be seen in *Fig 2.4.8* and the update stock layout can be seen in *Fig 2.4.9.*To update the stock the user would select an item from the dropdown list, enter an amount and add/deduct their desired amount.

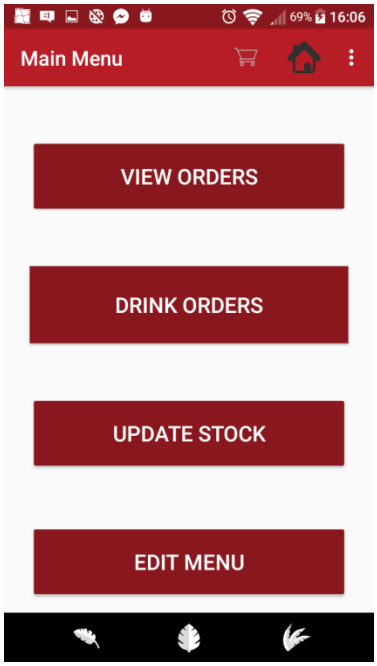


Fig 2.4.8 Staff Main Menu

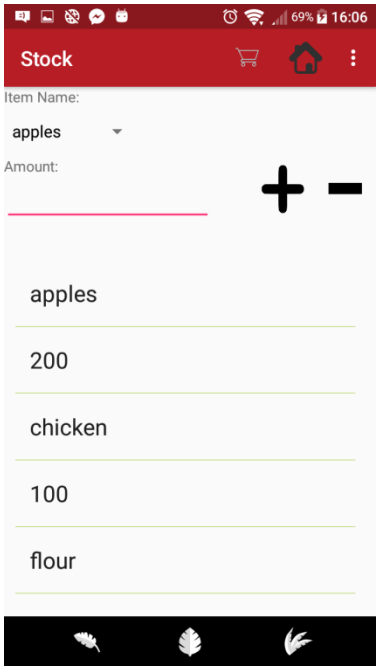


Fig 2.4.9 Update Stock layout

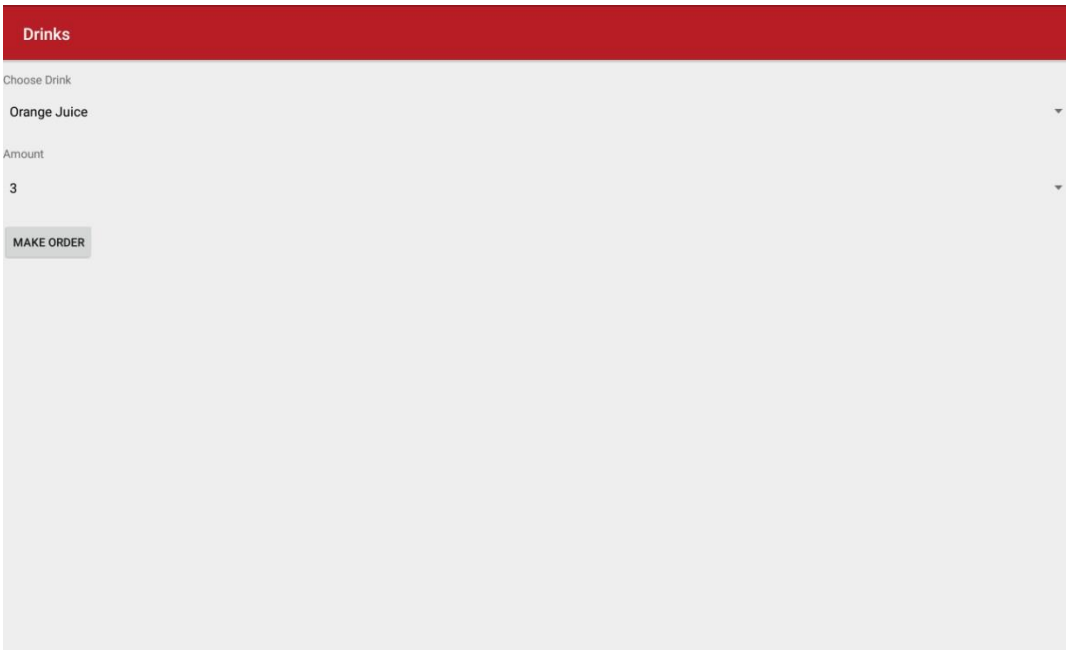


Fig 2.4.10 Ordering Drinks layout

2.6 Testing

2.6.1 Testing Different Devices

I have used a range of methods for testing compatibility of the Android application on different Android devices with different API levels. [13]

Code name	Version	API level
Nougat	7.1	API level 25
Nougat	7.0	API level 24
Marshmallow	6.0	API level 23
Lollipop	5.1	API level 22
Lollipop	5.0	API level 21
KitKat	4.4 - 4.4.4	API level 19
Jelly Bean	4.3.x	API level 18
Jelly Bean	4.2.x	API level 17
Jelly Bean	4.1.x	API level 16
Ice Cream Sandwich	4.0.3 - 4.0.4	API level 15, NDK 8
Ice Cream Sandwich	4.0.1 - 4.0.2	API level 14, NDK 7
Honeycomb	3.2.x	API level 13
Honeycomb	3.1	API level 12, NDK 6
Honeycomb	3.0	API level 11
Gingerbread	2.3.3 - 2.3.7	API level 10

Fig 2.6.1.1 Android API versions

The device that I have done most of my testing on is my personal smart phone which has API level 23. I have also used a smart tablet with API level 22.

To test other devices with different API levels I have used the Firebase Robo Tests that perform a series of automatic tests to find problems/errors with the Android application. This test is further explained in the following quote:

> *"Robo test is a test tool that is integrated with Firebase Test Lab for Android. Robo test analyzes the structure of your app's UI and then explores it methodically, automatically simulating user activities."* [14]

The vast majority of Android devices are within these two API levels (63.2%) [15].

I have tested API level 24 with the Firebase Robo Test and passed. This can be seen in *Fig 2.5.1.1*.

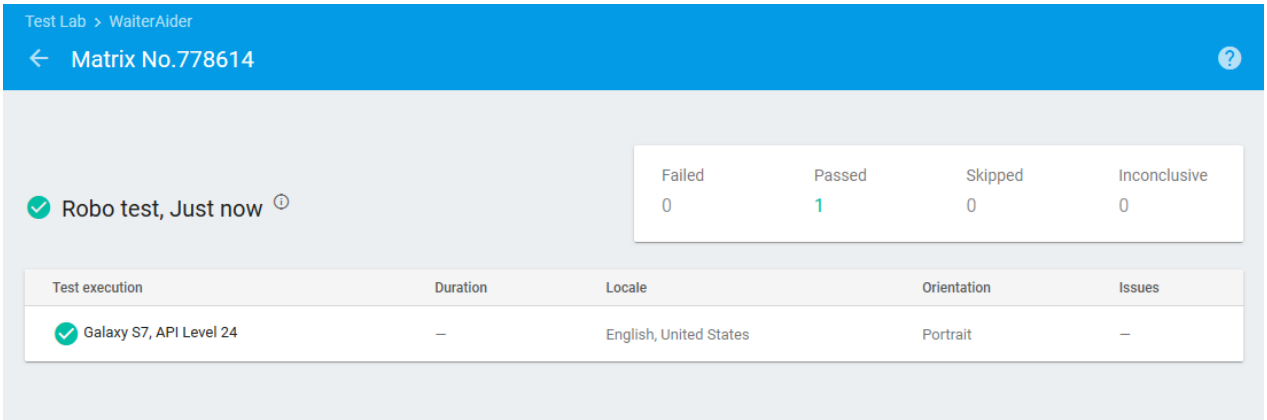


Fig 2.5.1.1 Result of the Robo Test on the Android application for API level 24

2.5.2 Manual Testing

For testing my application and database I did so manually using and manipulating different sections of the system. After a significant change was made in the application I would run the application on my phone. I would then test each section of the application where I had made changes to test for crashing and errors etc. To test the database, I would use a section of the application which had functionality to update the contents of table in the database and check to see if the desired changes were being reflected in the database by visiting the Firebase

console website. I also ran the application on my family member's tablet to make sure everything worked correctly on that.

2.5.3 Web-App testing

To manually test the web application I would make changes in the JavaScript or HTML code, run the *firebase deploy* command in the Windows command prompt on my computer which would deploy the latest code to the Firebase hosting service and then check the web application to see if the desired changes were reflected on the page.

I tested the web application by visiting its link on Microsoft Edge, Firefox, Chrome and Chromium (this is the browser that will be ran on the Raspberry Pi).

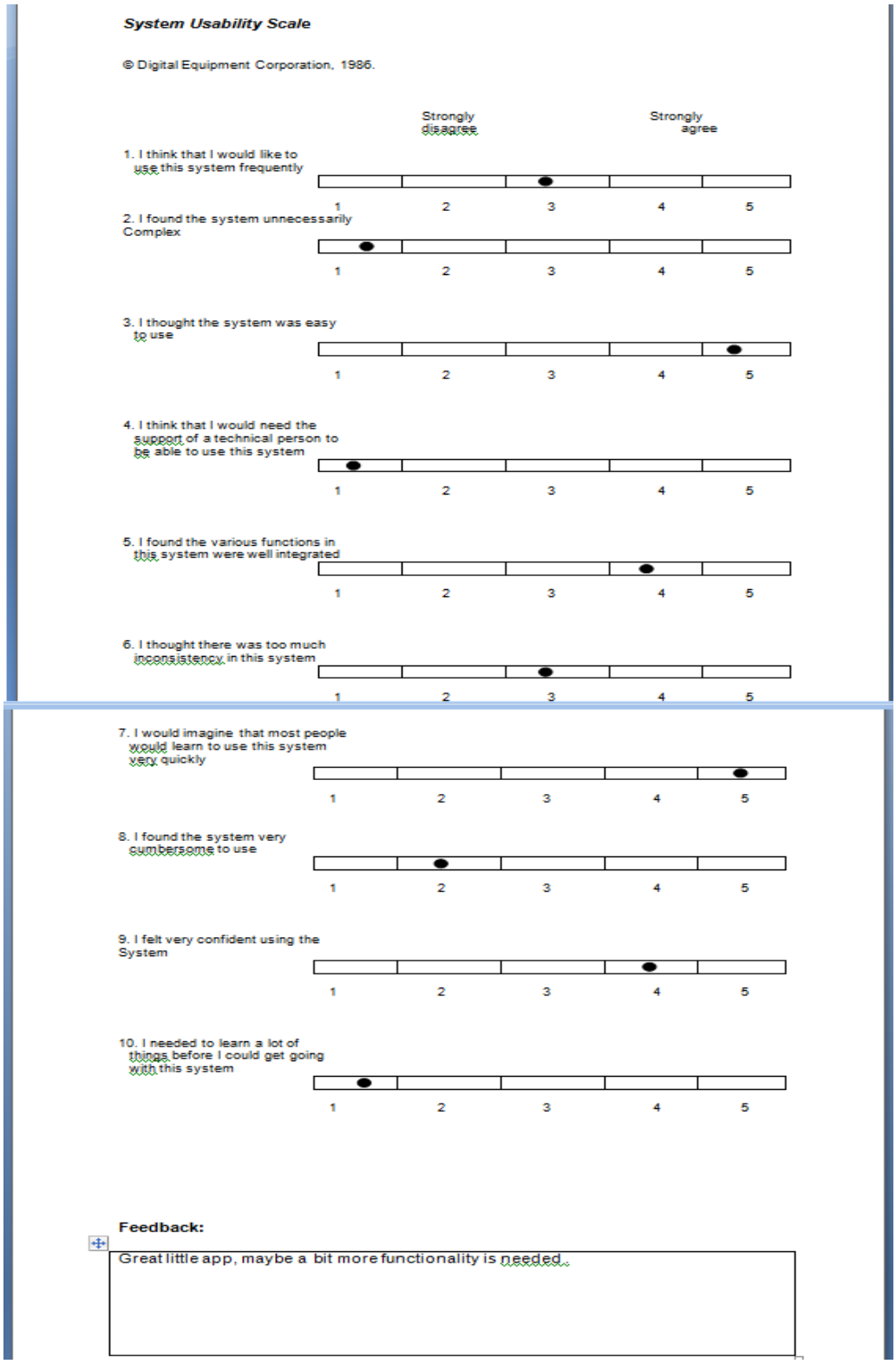
2.5.4 Unit & Implementation Testing

I was too far into development before I thought about implementing these automated testing techniques to gain many benefits as I should have been using them since day one.

2.7 Customer testing

The approach that I took for customer testing was to provide my participants with an SUS (System Usability Scale) survey [16], along with a section for allowing them to provide feedback on what they liked/disliked in regards to the application.

Below are the documents that they filled out. They tested the customer side of the application as the staff side was not in a complete-enough state at the time of the tests to gain much benefit from customer testing.



System Usability Scale

© Digital Equipment Corporation, 1986.

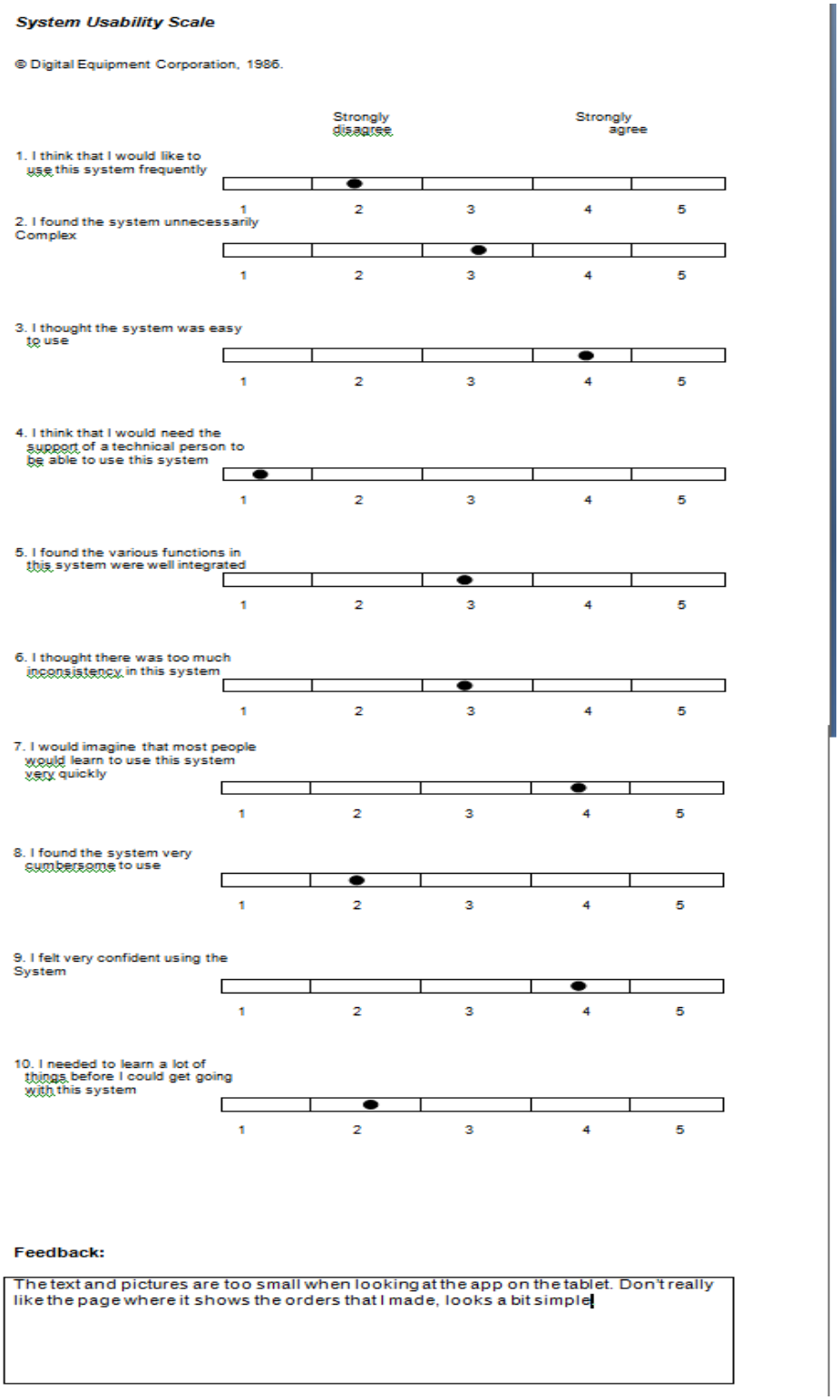
Strongly disagree — Strongly agree

1. I think that I would like to use this system frequently — 1 2 3 4 5
2. I found the system unnecessarily Complex — 1 2 3 4 5
3. I thought the system was easy to use — 1 2 3 4 5
4. I think that I would need the support of a technical person to be able to use this system — 1 2 3 4 5
5. I found the various functions in this system were well integrated — 1 2 3 4 5
6. I thought there was too much inconsistency in this system — 1 2 3 4 5
7. I would imagine that most people would learn to use this system very quickly — 1 2 3 4 5
8. I found the system very cumbersome to use — 1 2 3 4 5
9. I felt very confident using the System — 1 2 3 4 5
10. I needed to learn a lot of things before I could get going with this system — 1 2 3 4 5

Feedback:

Great little app, maybe a bit more functionality is needed.

Fig 2.6.1 Tester 1 SUS survey results



System Usability Scale

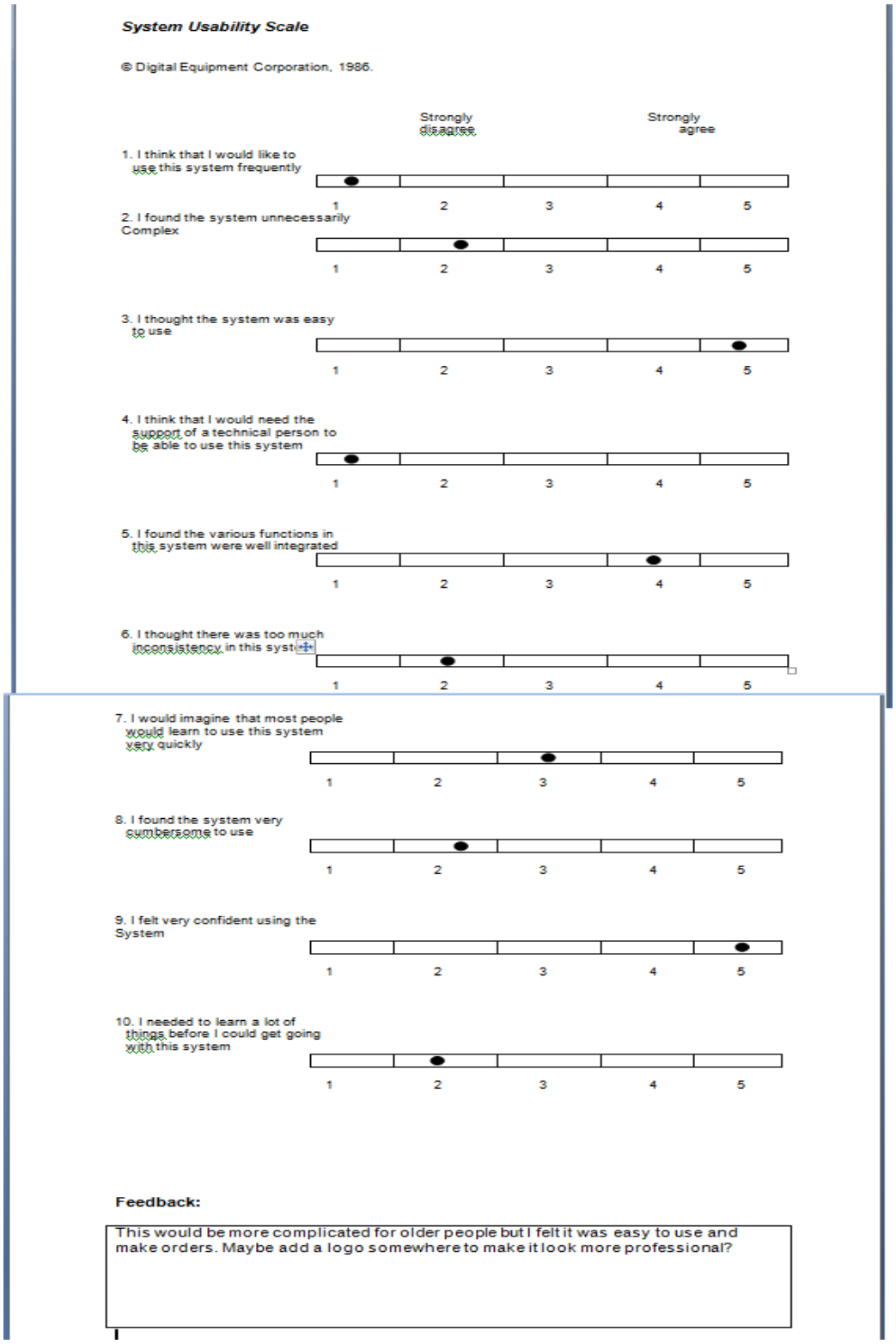
© Digital Equipment Corporation, 1986.

	Strongly disagree				Strongly agree
1. I think that I would like to use this system frequently	1	2	3	4	5
2. I found the system unnecessarily Complex	1	2	3	4	5
3. I thought the system was easy to use	1	2	3	4	5
4. I think that I would need the support of a technical person to be able to use this system	1	2	3	4	5
5. I found the various functions in this system were well integrated	1	2	3	4	5
6. I thought there was too much inconsistency in this system	1	2	3	4	5
7. I would imagine that most people would learn to use this system very quickly	1	2	3	4	5
8. I found the system very cumbersome to use	1	2	3	4	5
9. I felt very confident using the System	1	2	3	4	5
10. I needed to learn a lot of things before I could get going with this system	1	2	3	4	5

Feedback:

The text and pictures are too small when looking at the app on the tablet. Don't really like the page where it shows the orders that I made, looks a bit simple

Fig 2.6.2 Tester 2 SUS survey results



System Usability Scale

© Digital Equipment Corporation, 1986.

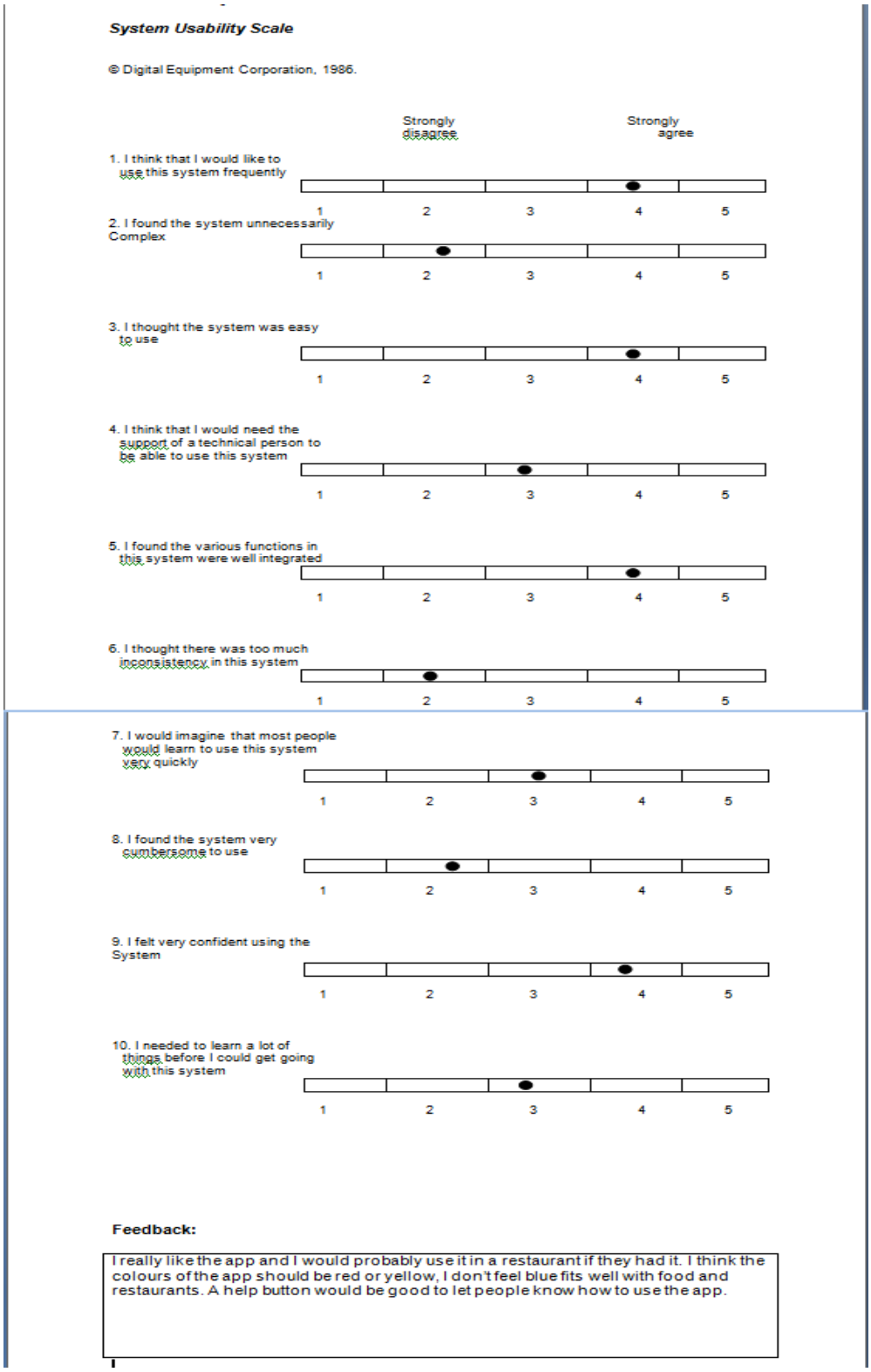
Strongly disagree — Strongly agree

1. I think that I would like to use this system frequently — 1 2 3 4 5
2. I found the system unnecessarily Complex — 1 2 3 4 5
3. I thought the system was easy to use — 1 2 3 4 5
4. I think that I would need the support of a technical person to be able to use this system — 1 2 3 4 5
5. I found the various functions in this system were well integrated — 1 2 3 4 5
6. I thought there was too much inconsistency in this syst — 1 2 3 4 5
7. I would imagine that most people would learn to use this system very quickly — 1 2 3 4 5
8. I found the system very cumbersome to use — 1 2 3 4 5
9. I felt very confident using the System — 1 2 3 4 5
10. I needed to learn a lot of things before I could get going with this system — 1 2 3 4 5

Feedback:

This would be more complicated for older people but I felt it was easy to use and make orders. Maybe add a logo somewhere to make it look more professional?

Fig 2.6.3 Tester 3 SUS survey results



System Usability Scale

© Digital Equipment Corporation, 1986.

Strongly disagree — Strongly agree

1. I think that I would like to use this system frequently — 1 2 3 4 5
2. I found the system unnecessarily Complex — 1 2 3 4 5
3. I thought the system was easy to use — 1 2 3 4 5
4. I think that I would need the support of a technical person to be able to use this system — 1 2 3 4 5
5. I found the various functions in this system were well integrated — 1 2 3 4 5
6. I thought there was too much inconsistency in this system — 1 2 3 4 5
7. I would imagine that most people would learn to use this system very quickly — 1 2 3 4 5
8. I found the system very cumbersome to use — 1 2 3 4 5
9. I felt very confident using the System — 1 2 3 4 5
10. I needed to learn a lot of things before I could get going with this system — 1 2 3 4 5

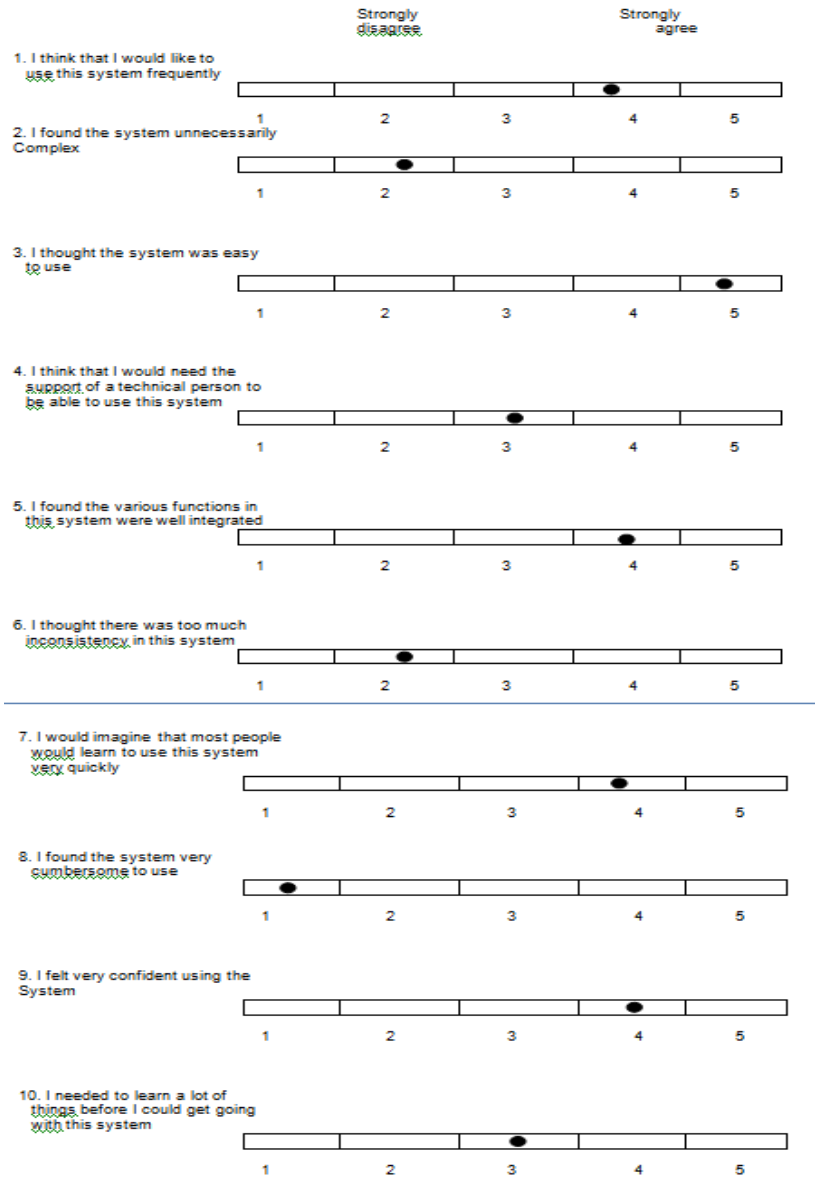
Feedback:

I really like the app and I would probably use it in a restaurant if they had it. I think the colours of the app should be red or yellow, I don't feel blue fits well with food and restaurants. A help button would be good to let people know how to use the app.

Fig 2.6.4 Tester 4 SUS survey results

System Usability Scale

© Digital Equipment Corporation, 1986.



Feedback:

Very well laid out. Easy to order. Straight forward. Quick and easy to use.

Fig 2.6.5 Tester 5 SUS survey results

Scoring of SUS results:

Tester Number	Score (Avg. is 68)	Rating
1	72.5	Above Average
2	70	Above Average
3	67.5	Below Average
4	77.5	Above Average
5	80	Above Average

The actions that I took to improve my application based on this feedback were:

I changed the theme colours of the Android app from blue to a maroon-y red colour which has more a restaurant feeling to it.

I allowed for the application to adjust its text size / picture size to match the current device.

I gained more of an insight into how to implement order progress into the application. I decided to add a progress bar that will be updated by the staff.

I understood that most things need to be labelled otherwise some people may not know what the intended use is for something.

Things to note:

4/5 surveys were answered by people in my own age-group (early 20s) who are comfortable with modern technologies such as using mobile applications. One survey was answered by a person who is older and less familiar with these technologies.

2.8 System Limitations

- Currently if more meals are to be added to the menu a new xml layout file and Java file need to be manually coded instead of them ideally being automatically generated/having one for all meals.

- The staff cannot edit every part of the design of the application and cannot add new meals through the application.

3 Conclusions

Advantages:

- The completed system can improve productivity by instantly sending orders to the kitchen in comparison to a longer delay by the waiter bringing that order to the kitchen. With customers being served more quickly, the restaurant could, given demand, have an increased number of customers.
- Costs can be decreased as the application allows the restaurant to not need as many waiters as the task of taking orders will not need to be done by them. They would only be needed to deliver the orders to the customers' tables.
- Communication can be increased by implementing this software system as it would reduce human error and mistaken orders as the responsibility for making sure the correct order is sent to the kitchen would be handed over from the waiter to the customer and the application. It can also be increased between staff and the customer as when a staff member updates an order, the customer will be notified and will have more information on when to expect their food.

Disadvantages:

- Requires internet connection.
- Devices are battery powered meaning they will need to be charged every so often / need a constant power supply (having a power supply on the tables themselves).

Opportunities:

- Could make the restaurant industry more profitable by decreasing labour costs.

4 Further development or research

The equipment that is used could be designed specifically for purpose. Currently, regular android devices are used for this system whereas with more funding, devices could be designed for the tasks of this system. For example, the supplied devices having increased battery length, waterproofing, and shock resistance etc.

With further development, this piece of software could expand its market interest to bars and other retail outlets i.e. customer picks their drink with an app which could speed up the pouring of drinks. This would work by each user having their own account, ordering a drink and being given a notification once it has been poured.

Payment transactions would be another possibility for the system using NFC technologies, given more development. NFC is a technology with a set of communication protocols that allow two electronic devices to communicate with one another. This technology is embedded into some credit/debit cards and smartphones. Given further development, a customer could swipe their card or smart phone against the supplied tablet device to make a payment. [17]

5 References

[1] Food Newsfeed, *Study Released on Average Restaurant Wait Times*. [online] Available at: https://www.foodnewsfeed.com/new-concepts/study-released-average-restaurant-wait-times [Accessed 7 May 2017].

[2] Terry, J. *Apple vs Android: Just the Facts - Top Mobile Trends*. [online] Available at: http://topmobiletrends.com/apple-vs-android/ [Accessed 3 May 2017].

[3] Firebase [online] Available at: https://firebase.google.com/ [Accessed 7 May 2017].

[4] Bowman, David. "Data Requirements". Available at: information-management-architect.com. Web. 9 Dec. 2016.

[5] *Usability First - About Usability - Requirements Specification | Usability First*. [online] Available at: http://www.usabilityfirst.com/about-usability/requirements-specification/ [Accessed 2 May 2017].

[6] *ExpandableListView* [online] Available at: https://developer.android.com/reference/android/widget/ExpandableListView.html [Accessed 7 May 2017].

[7] Techotopia.com. *An Android Master/Detail Flow Tutorial - Techotopia*. [online] Available at: http://www.techotopia.com/index.php/An_Android_Master/Detail_Flow_Tutorial [Accessed 6 May 2017].

[8] Developer.android.com. *Intent | Android Developers*. [online] Available at: https://developer.android.com/reference/android/content/Intent.html [Accessed 7 May 2017].

[9] Stackoverflow.com. *How to use putExtra() and getExtra() for string data*. [online] Available at: http://stackoverflow.com/questions/5265913/how-to-use-putextra-and-getextra-for-string-data [Accessed 7 May 2017].

[10] Firebase. *Firebase Hosting | Firebase*. [online] Available at: https://firebase.google.com/docs/hosting/ [Accessed 7 May 2017].

[11] Tutorialsface.com. *Android Fully Functional Ecommerce App Tutorial from Scratch – Part 1 | Tutorials Face*. [online] Available at: http://www.tutorialsface.com/2015/10/android-fully-functional-ecommerce-app-sample-example-tutorial-from-scratch-part-1/ [Accessed 7 May 2017].

[12] Androiddom.com. *Android Shopping Cart Tutorial*. [online] Available at: http://www.androiddom.com/2011/02/android-shopping-cart-tutorial.html [Accessed 7 May 2017].

[13] Android Open Source Project. *Codenames, Tags, and Build Numbers | Android Open Source Project*. [online] Available at: https://source.android.com/source/build-numbers [Accessed 7 May 2017].

[14] Firebase.*Firebase Test Lab for Android Robo Test | Firebase*. [online] Available at: https://firebase.google.com/docs/test-lab/robo-ux-test [Accessed 7 May 2017].

[15] Developer.android.com. *Dashboards | Android Developers*. [online] Available at: https://developer.android.com/about/dashboards/index.html [Accessed 7 May 2017].

[16] Usability.gov. *System Usability Scale (SUS) | Usability.gov*. [online] Available at: https://www.usability.gov/how-to-and-tools/methods/system-usability-scale.html [Accessed 7 May 2017].

[17] Faulkner, C. *What is NFC? Everything you need to know*. [online] TechRadar. Available at: http://www.techradar.com/news/phone-and-communications/what-is-nfc-and-why-is-it-in-your-phone-948410 [Accessed 7 May 2017].

6 Appendix

6.1 Project Proposal

Project Proposal

WaiterAider

Declan Casey, 13564737, x13564737@student.ncirl.ie

BSc (Hons) in Computing

Internet of Things

14/10/2016

Objectives

Project Pitch (Due Date 5th October)

For this objective I was tasked to think of an idea for my project and to present it in front of several lecturers in a 'Dragon's Den' style scenario. I was provided with feedback on my idea and was given advice on how I could alter it to make it more suitable for a final year project.

Project Proposal (Due Date 21st October)

The project proposal is a document that details the early stages/the start of a project's idea and goals. It is a vital piece of information that can be looked at throughout the development to keep track of upcoming deadlines and objectives of the project. This document will include some background information, research and a plan for how the project will be completed.

Requirements Specification (Due Date 11th November)

The requirements specification document details several important details including the project scope, use case diagrams for each requirement and the proposed GUI and system architecture for the software project. The document also details all functional and non-functional requirements that the project has.

This document will be beneficial when I start developing the project to keep track of all required functionality and how my proposed finish product will end up like.

Prototype (Due Date 2nd December)

The project prototype is the first example of the project's working code that holds some but not all of the planned functionality for the piece of software.

It is not determined yet whether this prototype is to be uploaded to moodle or shown to the lecturer through other means.

Mid-Point Presentation (Due Date 16th/17th December)

This presentation's purpose is to present the progress that has been made so far to ensure that the student is going in the right direction and so that they know what they have done so far is up to an acceptable standard.

Testing (Throughout Development)

I will continuously test the application throughout development to ensure all sections of the system are working as they should be, and to eliminate all bugs and crashes.

Customer Testing (Throughout April – May)

This will include allowing people to use my application and making them take surveys and give feedback to improve the final version of my application.

Final Project Completion (Due Date 14h May)

This date is when every part of the project should be completed. This includes the software, hard copies of all project materials and the final presentation of the fully functional software. All of these will be due on the same date.

Attend Supervisor Meetings (Attend continuously)

In order to keep my project on track with its goals I will have to arrange meetings with my academic supervisor and attend them every so often throughout the two semesters.

Background

I first started my research on the subject of my specialisation, the Internet of Things, during my work placement earlier in the year. Once I had seen the specialisations that were on offer and I had chosen mine I started searching for the different technologies that were used in my specialisation as well as projects created with the Internet of Things (IoT) on tech websites and YouTube.

Examples of these would be drones, IR technology and apps that can control all appliances in a home. I found the large amount of possibilities open to this new

age technology very interesting and realised the advantages open to me by choosing this specialisation.

The first version of this idea came to me while in the first class of the semester. After hearing from my lecturer that everyone should have at least some idea for what their final year project will be I stuck the thinking cap on and by the end of the class I had the first version of my idea.

I knew the idea had to be associated with my specialisation. I knew I would probably be using technologies such as Raspberry Pis as well as creating apps with android studio. So taking that in mind, what I came up with was the idea for an app which could connect to a visual display wirelessly in order to display information generated by the app. My original use for this idea was to be able to change the text on a storefront sign using the app to display special offers and other information.

After discussion with my IoT lecturer, he advised me that this idea was suitable for my final year project but that I had to expand on it in some way. In the following weeks I continued to think on my final idea by adding more complexity to it and ensuring that it has a clear target market.

The idea that I ended up proposing at my Project Pitch was similar to my original idea with a slight difference/ added complexity. Instead of or as well as the app being used for customising a storefront sign I proposed new functionality that would allow the app to customise a behind-the-counter menu similar to what you would see in a café or fast food service. The menu's design as well as its contents would be customisable and would have different design schemes available.

After completing my Project Pitch I was advised to alter my project in a small way. I was told that for a final year project a bit more would be expected and the 'dragons' suggested that I use my app for restaurants. Their idea was to have the app as a replacement to restaurant menus. This would work by handing the customer a smart tablet with the app on it and they would be able to order their meal with it and that order would be sent to a screen in the kitchen. This sending

of information and replacement of the restaurant menu shows the IoT aspect to the project and how it could impact a business positively.

I feel that this idea is an improvement to my proposed idea so I feel that the pitch scenario was very useful to me looking forward.

Technical Approach

The approach that I will take to see this project to completion will be one taken over a long period of time. I have already started this approach by researching what IoT is and how it is implemented into our everyday lives.

I've done some research on different projects created within the area of IoT and reviewed several online articles on new technologies and possibilities that it opens up.

The next steps that I will take will be to complete my proposal and to start on my requirements document which will go into more detail about my proposed project and how I will go about completing it.

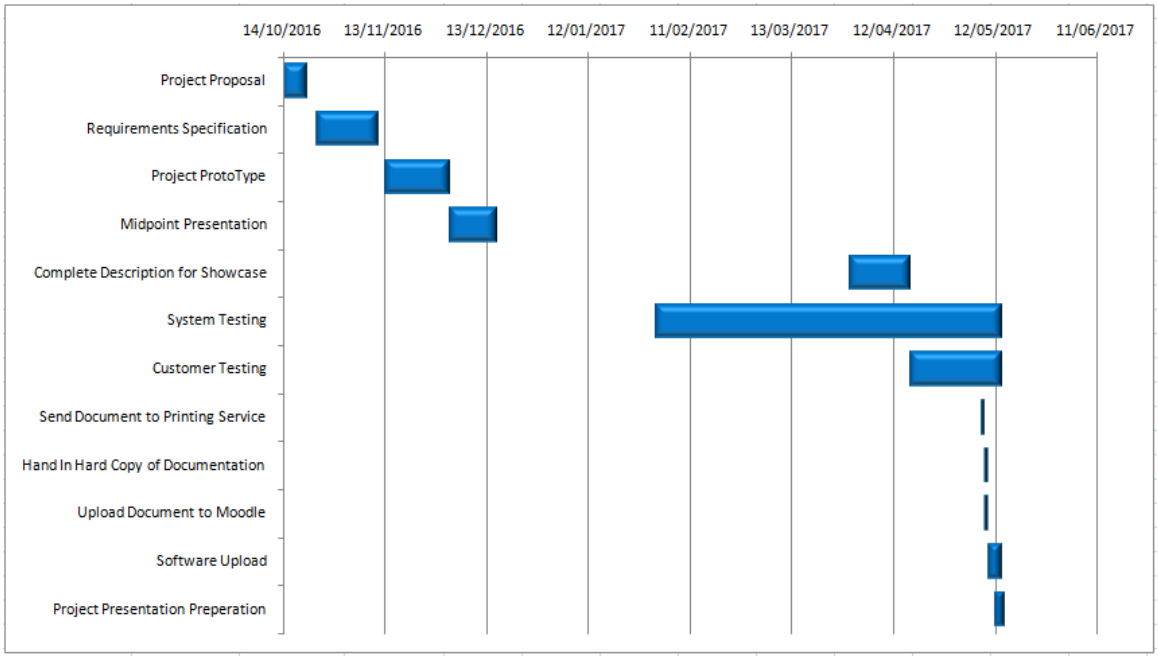
I will ensure that I keep on schedule with my project deadlines as well as my own personal deadlines and other modules so that I do not lag behind in where my project should be at any given point in the year. This will be done by keeping track of my progress by writing the monthly journal and by writing out plans for upcoming weeks/months.

I will also learn the basics of python as it is a common programming language used with raspberry pi and continue to attend the multimedia and mobile app development module to increase my skills in android studio.

Special resources required

The special resources required for my project will be a Raspberry Pi, smart tablet and a digital display like a monitor as well as any accessories that go with them.

Project Plan



Technical Details

What I do know is that I will probably be using a majority of Java and XML to create the app through Android studio as well as HTML and JavaScript for the web application that will run on the Raspberry Pi. The code for android studio is java which has specific predefined classes in it so that it will work on an android device. To code in python I will be using the python command line interface.

Evaluation

I will evaluate my application with constant testing including personal testing and user testing. I will repeatedly compare it to the requirements specification document to ensure that all requirements are completed.

To test the system I will run the application every time a new feature is added to ensure that feature works correctly and so that no other features have been broken by adding it. This will require checking each feature individually throughout the process.

For integration testing I will test my app on multiple devices, including a virtual machine on my PC, on my android phone and an android tablet.

To evaluate the system with an end user I will be using my family, friends, classmates and if possible my supervisor. This testing will cover those who are and aren't tech savvy which will provide good feedback to me on how user friendly the app's UI is and how easy it is to use.

Declan Casey_19/10/2016_____

Signature of student and date

6.2 Project Requirements Document

National College of Ireland

Project Requirements

Declan Casey
01/11/2016

Requirements Specification (RS)

Document Control

Revision History

Date	Version	Scope of Activity	Prepared	Reviewed	Approved
01/11/2016	1	Create	AB		
01/05/2017	2	Update	CD		

Distribution List

Name	Title	Version
Eamon Nolan	Lecturer	
Dominic Carr	Academic Supervisor	

Related Documents

Title	Comments
Title of Use Case Model	
Title of Use Case Description	

Table of Contents

Introduction

My project is an android application which could be used in any restaurant as a replacement for paper menus and waiters taking orders by having the whole process done through an Android application.

Purpose

The purpose of this document is to set out the requirements for the development of my final year software project. It will detail all current requirements as well as providing use cases for each requirement. The document will also be updated throughout the development of the project with any changes to the requirements.

The intended customers for the finished software are business owners; those who are in the restaurant industry in particular. The users of the application will be both the customers and staff. Each set of users will have their own separate account in the application.

Project Scope

The scope of the project is to develop an Android application that will be a replacement for restaurant menus and the taking of orders by waiters. The system shall have the functionality to take orders from customers and to send that information wirelessly to the kitchen where the chef or someone similar will view the order and complete it.

Stock will also be able to be managed through the application by updating the stock numbers and as people order meals the stock number will go down.

I was involved in discussion with my academic supervisor to elicit the following requirements. I also used my own personal experience of working in a kitchen type environment.

Definitions, Acronyms, and Abbreviations

IoT – Internet of Things

Android Studio – Software for creating android applications.

App – Android application

Admin – Administrator

- Number

GUI – Graphical User Interface

Clicks – Meaning either using the mouse's left click or applying pressure to a button on an app.

User Requirements Definition

This section defines what the objectives and requirements for the system are from the user's perspective for both Staff and Customer users.

The main user non-functional requirements for this piece of software are:

- Responsiveness – the app's UI should be quick to navigate and pages should be able to be changed in less than three seconds. No loading screen should last longer than eight seconds.

- Convenience – the app should provide an improvement in its purpose to what is currently the standard. In this case, it should be more convenient than getting a waiter to take the customer's order and for them to bring that order to the kitchen to alert the kitchen staff.

- Cost Efficiency – the app should not increase on the user's current costs. For example, the cost of the equipment and software for my project should not be higher than the savings they can create.

- Increased Communication – the app should provide an increased level of communication between staff that will rule out or decrease errors in orders. By handing the responsibility of creating the order to the customer through the app and that information being displayed to the staff in the kitchen it can cut out mistakes in orders.

Requirements Specification

For kitchen staff, they should be given sufficient training in using the app to a point where they should be capable of using all functionality open to them. An expected one-to-two hours of training for these users should be more than sufficient to

become proficient in using the app and have the ability to assist customers who are not familiar with mobile technologies (elderly etc.).

The customers will require no training and the desired outcome is that they will be able to operate the app and complete their order without help from staff.

Customers will operate the app through an Android tablet provided by the restaurant. Staff will operate the app either through an android phone or tablet and will be able to view current orders through a digital display connected to a Raspberry Pi.

Functional requirements

The following functional requirements will be listed in a ranked order. These requirements describe the goals that the software system must accomplish to be a complete application.

Use Case Diagram

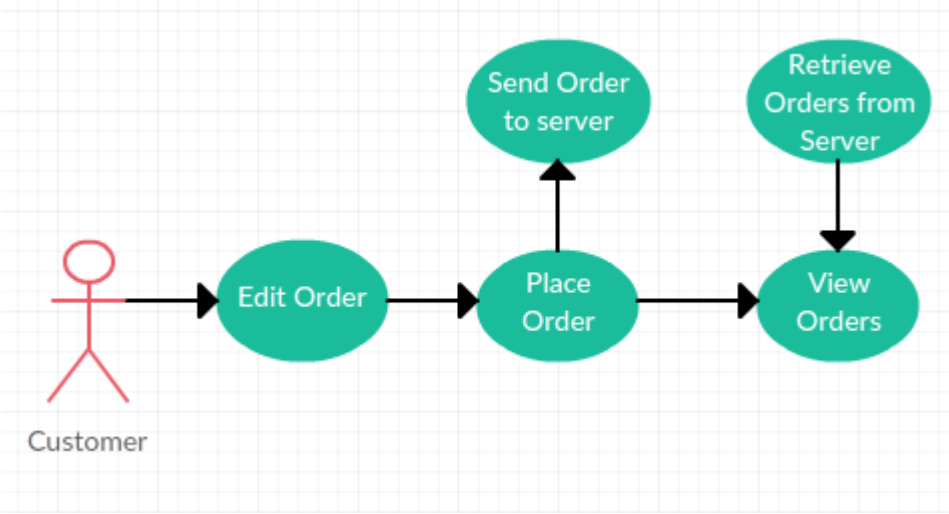


Fig 3.1.1.1 Every use case for the Customer user

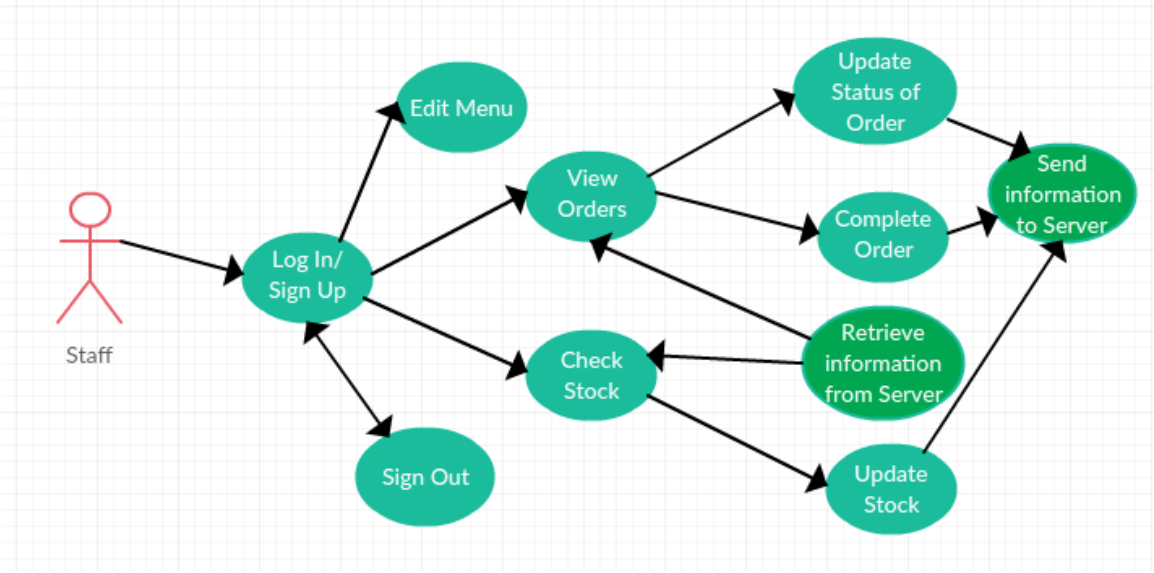


Fig 3.1.1.2 every use case for the Staff user

Requirement 1: View Orders

Description & Priority

This requirement details the process of viewing a created order/s. This is the most important requirement in the system as without the possibility to view orders there would be no information to work with to complete many of the other requirements.

Use Case

Actors

Customer, Staff

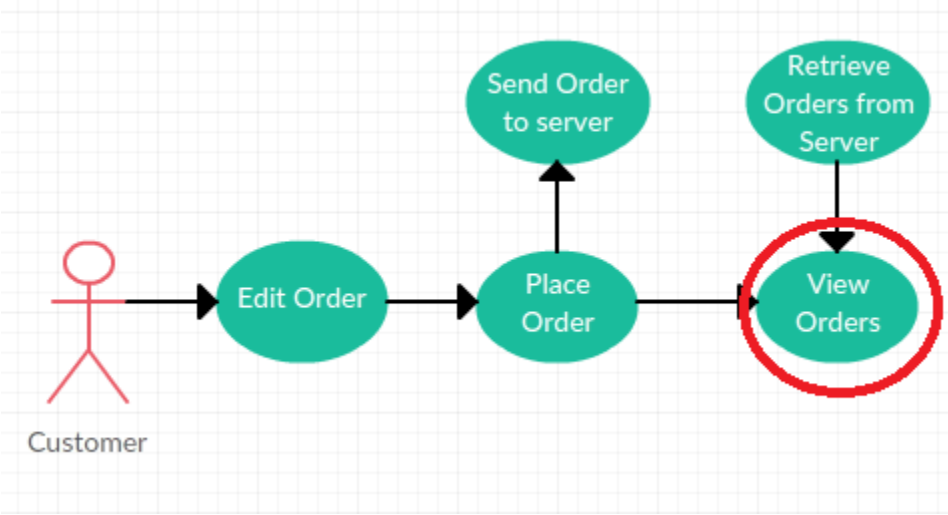
Scope

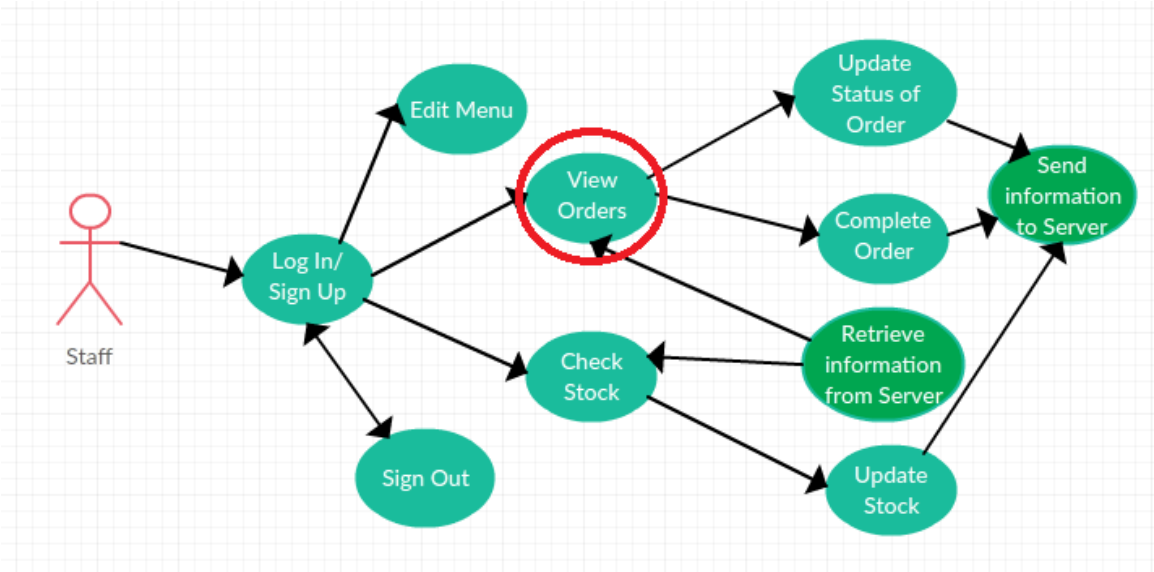
The scope of this use case is to view created orders on the software system so that the customer can review their order and so that the staff can complete/edit the order. The orders are available for viewing through a web application on the Raspberry Pi or through the app.

Description

This use case describes a step-by-step process to view created orders on the system.

Use Case Diagram





Flow Description

Precondition

The system is waiting on an input from the user.

Activation

This use case starts when the user opens the main menu of the application.

Main flow

1. The Customer/Staff navigates to the view orders section of the application by clicking on the View Orders button on the main menu.

2. The system displays all current orders to Staff with cost, table and seat number attached to them. It displays all orders associated with a customer's seat number to that customer. This information is retrieved from the server. (See A1, A2, E1)

Alternate flow

A1: No Orders

1. If there are no orders, the system will display a message to the user notifying them that no orders have been placed.
2. The use case continues at position 1 of the main flow.

Exceptional flow

E1: No Orders (fault)

1. The system notifies the user that there are no current orders when it is not the case.
2. The system displays an unable to sync with server error message.
3. The system attempts to reconnect to the server. (See E2, E3)

E2: Connection re-established

1. The system re-establishes connection with the server.
2. The use case continues at position 2 of the main flow.

E3: Connection Failed

1. The system fails to re-connect with the server.
2. The system goes into a wait state until it reconnects with the user.

E4: Power Failure
If there is a power failure due to a loss of power or a power-surge then the user will be logged out from the system and will no longer have access to the application.

Termination

The user views the orders.

Post condition

The system goes into a wait state

Requirement 2: Place/Edit Order

Description & Priority

This requirement details the process of creating an order by the Customer that will be sent to the kitchen. It also details the process of editing a created order. Its priority is high as no orders would be sent without the use case, essentially making the software system useless.

Use Case

Actors

Customer

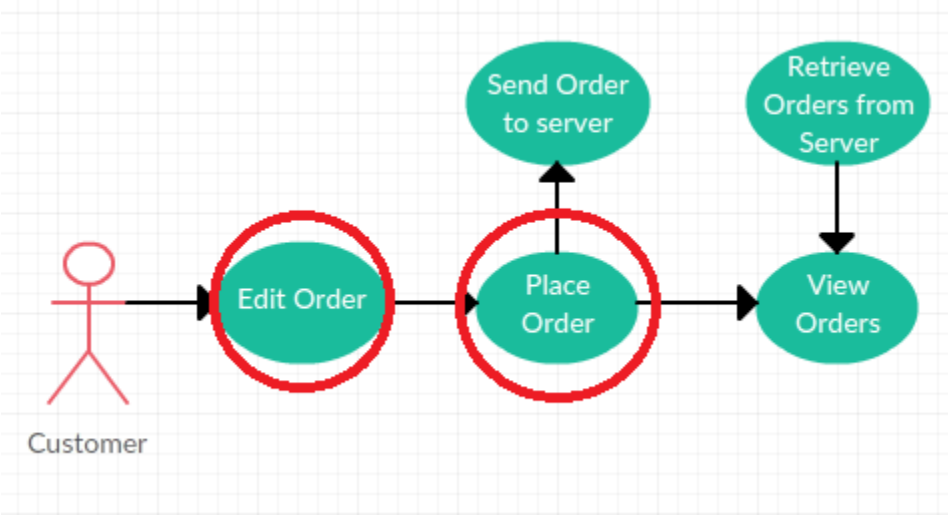
Scope

The scope of this use case is to create or edit orders so that the staff in the kitchen can view that information and complete the order. The customer will be able to navigate through the restaurant's menu and pick their meals and/or drinks by using the app.

Description

This use case describes the steps involved in placing an order and editing a placed order.

Use Case Diagram



Flow Description

Precondition

The system is waiting on an input from the user.

Activation

This use case starts when a customer clicks on the [Place Order] button.

Main flow

1. The system navigates to the Place Order section of the application.
2. The Customer clicks on either the [Starter], [Main Course], [Desert], [Deals] or [Drinks] button which expands a list of meals/drinks associated.
3. The Customer selects items for order. (See A1)
4. The system adds all selected items to an order cart.(See E1)
5. The customer navigates to the cart layout.
6. The Customer clicks the [Confirm Order] button. (See A3)
7. The order is sent and stored in the server. It is accessible for viewing on the My Orders section. (See A2)

Alternate flow

A1: Select Additional Item

1. The Customer backs out to the meal list and selects a new item from the list.
2. The use case continues at position 2 of the main flow.

A2: Edit Order

1. The Customer navigates to the item that they have added to their order.
2. They can then delete that item from their order and re-order a different amount of that order if they wish.
3. The use case continues at position 4 of the main flow.

A3: Drink Order

1. The customer clicks on the [Drinks] button.
2. The customer selects beverages for order.
3. The system adds the selected items to the cart.
4. The use case continues at position 5 of the main flow.

Exceptional flow

E1: Power Failure

Identical to E4 of the View Orders use case.

Termination

The system sets the order as placed and waits for the order to be completed.

Post condition

The system goes into a wait state.

Requirement 3: Complete Order

Description & Priority

This requirement details the process of completing an order by the staff that is sent to the kitchen. Its priority is high as it would not be a good idea for a staff member to be able to view every single order that has ever been created as it would cause a cluttering of orders in the application. Setting the order as complete removes the order from the application.

Use Case

Actors

Staff

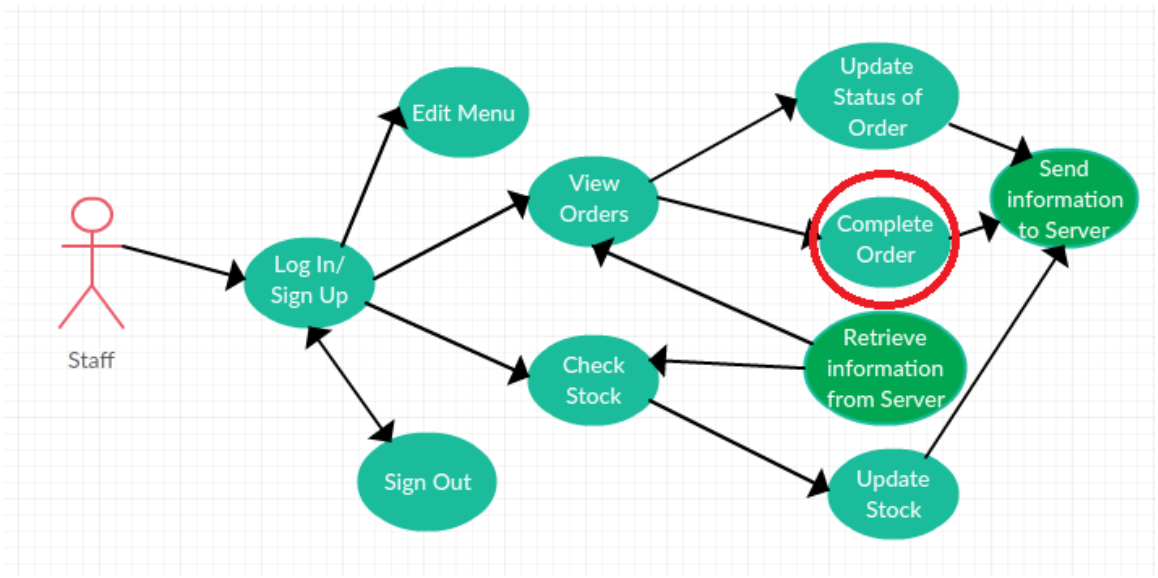
Scope

The scope of this use case is to complete orders so that the staff in the kitchen can make up the order and deliver it to the customer. The staff will be able to navigate to the View Orders section of the app and from there set the order as complete.

Description

This use case describes the steps involved in completing a placed order.

Use Case Diagram



Flow Description

Precondition

The system is waiting on an input from the user.

Activation

This use case starts when a staff member clicks on the [View Orders] button.

Main flow

1. The system navigates to the View Orders section of the application.
2. The staff member navigates to the order that they wish to complete. (see E1)
3. The staff member types the username belonging to the order they wish to complete and clicks the [Complete Order] button. (See E1)
4. The system sends and stores the orders in the Completed Orders of the database.

Exceptional flow

E1: No Orders (fault)

Identical to E2 of the View Orders use case.

E2: Power Failure
Identical to E4 of the View Orders use case.
Termination

The system sets the order as completed and removes it from the confirmed order list.

Post condition

The system goes into a wait state.

Requirement 4: Check/Update Stock

Description & Priority

This requirement details the process of checking and updating the stock by the kitchen staff. Its priority is medium as it is not a vital requirement for the kitchen as it can be done manually but it also offers an easier alternative solution for managing stock.

Use Case

Actors

Staff

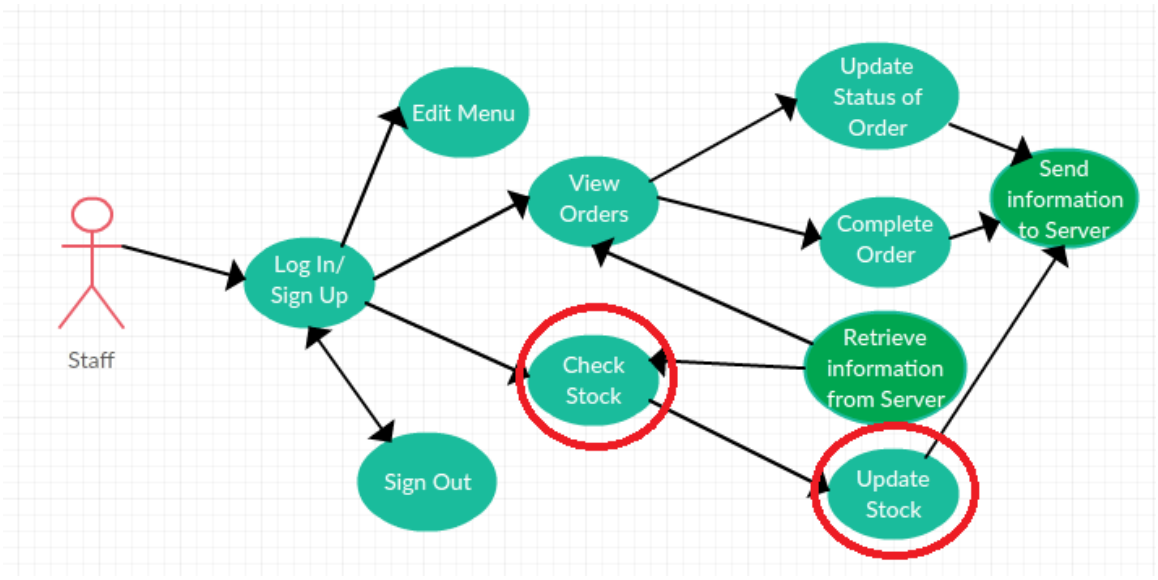
Scope

The scope of this use case is to view and update the restaurant's stock so that no orders can be made if there is not enough stock to complete them.

Description

This use case describes the steps involved in checking and updating the stock.

Use Case Diagram



Flow Description

Precondition

The system is waiting on an input from the user.

Activation

This use case starts when the staff clicks on the [Update Stock] button.

Main flow

1. The system navigates to the Stock section of the GUI.
2. The system outputs the list of stock. Items that are low in stock will be marked in red. (see A1, E1)
3. The user navigates through the stock list.
4. The user selects the item they wish to update and enters an amount that they wish to add/subtract from that item.
5. The user clicks the add or subtract button to update the stock value for the selected item.
6. The system reflects the changes made in the layout.

Alternate flow

A1: No Stock

1. If there is no stock information entered into the system no stock is shown in the stock list.
2. The use case ends.

Exceptional flow

E1: No Stock (fault)

The system will attempt to reconnect with the server to retrieve correct stock information.

E2: Power Failure
Identical to E4 of the View Orders use case.

Termination

The user has checked the stock.

Post condition

The system goes into a wait state.

Requirement 5: Log In/Sign Up/Sign Out

Description & Priority

This requirement details the process of logging in and signing out of the application, and signing up for a new account to access the application by the staff. Its priority is medium as it is an important feature that allows access to other areas of the application but it is only needed to be done once a day, usually.

Use Case

Actors

Staff

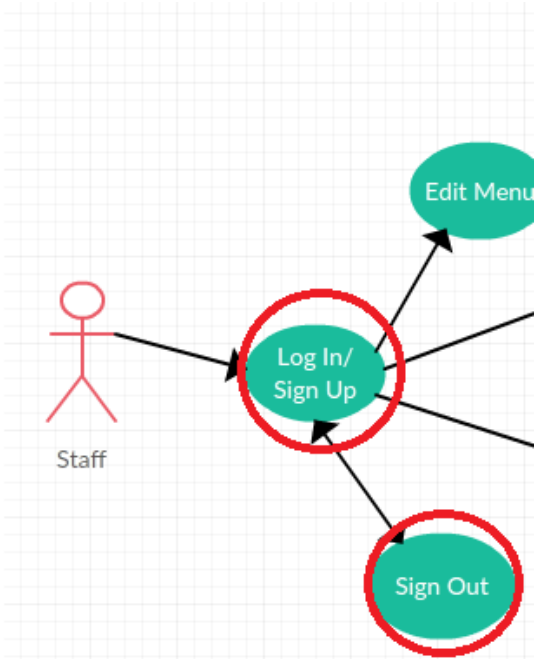
Scope

The scope of this use case is to log in/sign out/sign up to the application so that the user can access other sections of the application.

Description

This use case describes the steps involved in completing the logging in, signing out and signing up operation.

Use Case Diagram



Flow Description

Precondition

The system is waiting on an input from the user.

Activation

This use case starts when the staff/admin launches the application.

Main flow

1. The system prompts the user to log in. (See A1)
2. The user enters their username and password into the input boxes presented by using the android's keyboard. (See A2)
3. The user clicks the Log In button which sends the entered information to the server to be validated.
4. The system completes the logging in operation and the user will be logged in. (See A3, A4)

Alternate flow

A1: Sign-up

1. The user clicks on the button to navigate to the sign-up page.
2. The user enters a valid username and a password with at least 6 characters.
3. The user clicks the sign-up button.
4. The use case continues at position 4 of the main flow.

A2: Log In for Customer Device

1. When the staff member is logging in they enter the seat and table number that the device has been assigned to as a username and its password.
2. The use case continues at position 3 of the main flow.

A3: Invalid Log In

1. The system notifies the user that their entered log-in details are incorrect.
2. The system returns to the log in prompt.
3. The use case continues at position 2 of the main flow.

A4: Sign Out

1. The user clicks the sign out button.
2. The system signs the user out and redirects them to the log in section of the application.

Exceptional flow

E1: Power Failure
Identical to E4 of the View Orders use case.

Termination

The user is logged in and has access to other areas of the application.

Post condition

The system goes into a wait state.

Requirement 6: Edit Menu

Description & Priority

This requirement details the process of editing the image and details of a meal in the application's menu. Its priority is low as it is not vital in the overall purpose of the application; it just holds some extra functionality for staff to freshen up the layout of the meals. It will be done through the Android application.

Use Case

Actors

Staff

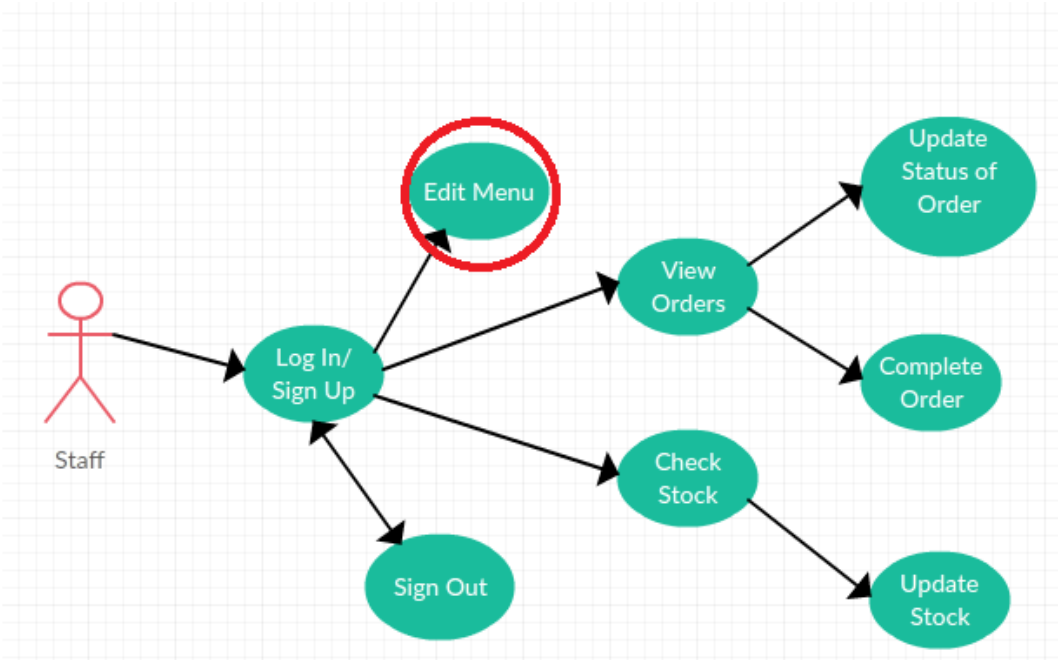
Scope

The scope of this use case is to edit the meals in the application's menu. Once the changes have been made the application should reflect those changes.

Description

This use case describes the steps involved in editing the restaurant's menu.

Use Case Diagram



Flow Description

Precondition

The system is waiting on an input from the user.

Activation

This use case starts when the staff clicks on the [Edit] button of a particular meal layout.

Main flow

1. The system shows a selection of new buttons to the user.
2. The staff user navigates selects one of the buttons corresponding to what they wish to edit.
3. The system allows the user to edit that item.
4. The staff clicks on the [Save] button. (See A1)
5. The system changes the layout to reflect the changes made. (See A2)

Alternate flow

A1: Additional Change

1. The user repeats step 2, 3 and 4 of the main flow.
2. The use case continues at position 5 of the main flow.

A2: No Save

1. The user clicks the [Cancel Edit] button.
2. The system cancels the changes made to the menu.
3. The use case continues terminates.

Exceptional flow

E1: Power Failure
Identical to E4 of the View Orders use case.

Termination

The system amends the current menu to reflect the changes made.

Post condition

The system goes into a wait state.

Requirement 7: Update progress of Orders

Description & Priority

This requirement details the process of updating the progress of Customer orders by the Staff users to reflect the progress made on their order. It will be done through the Android application.

Use Case

Actors

Staff

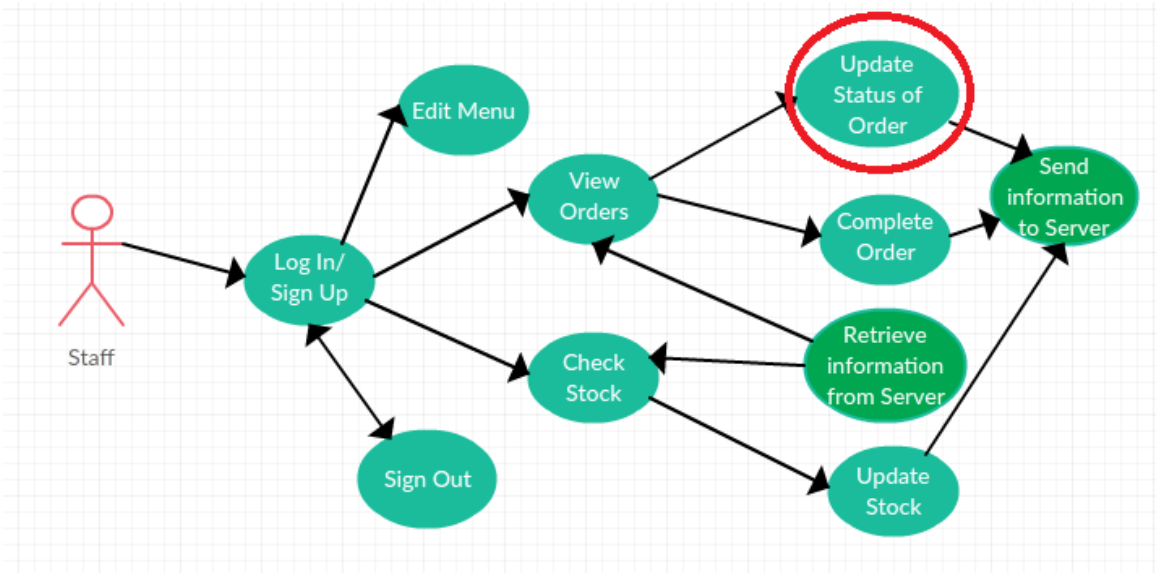
Scope

The scope of this use case is to edit update the progress of a particular customer's order. Once the update has been made the customer should be able to view this update.

Description

This use case describes the steps involved in updating progress on a customer's order.

Use Case Diagram



Flow Description

Precondition

The system is waiting on an input from the user.

Activation

This use case starts when the staff clicks on the [View Orders] button.

Main flow

1. The system pulls the orders from the server. (See A1)
2. The staff user selects the update type that they wish set on a particular order.
3. The staff types the name of the seat and table number of and clicks the Confirm button.
4. The system sends the update to the server.
5. The update is sent to the View Orders section attached to the customer user that made the order.

Alternate flow

A1: No Orders

1. There are no orders in the server.
2. A message is sent to the user alerting them that there are no orders.
3. The use case ends.

Exceptional flow

E1: Power Failure
Identical to E4 of the View Orders use case.

Termination

The system updates the status of the order.

Post condition

The system goes into a wait state.

Non-Functional Requirements

Performance/Response time requirement

The application should be quick and responsive and should not take more than 3 seconds to navigate between pages and should take no more than 10 seconds when exchanging information with the server.

Availability requirement

This application should be available for use on all android phone and tablet devices.

Recover requirement

Version control will be kept of each version of the application so that previous versions can be recovered.

Robustness requirement

There should be no more than 2 bugs experienced per day within the application.

Security requirement

Access to sensitive information should not be able to be accessed without securely logging in to the application. Account log-ins should have password encryption to ensure strong security. Customer users should not be able to gain the functionality of a staff user.

Reliability requirement

The application should not crash more than twice a day.

Maintainability requirement

Regular bug fixing should be carried out to maintain working order of the system.

Portability requirement

The application should be able to be used in the whole restaurant. Every area of the operational space of the restaurant should have a WIFI signal.

Interface requirements

GUI

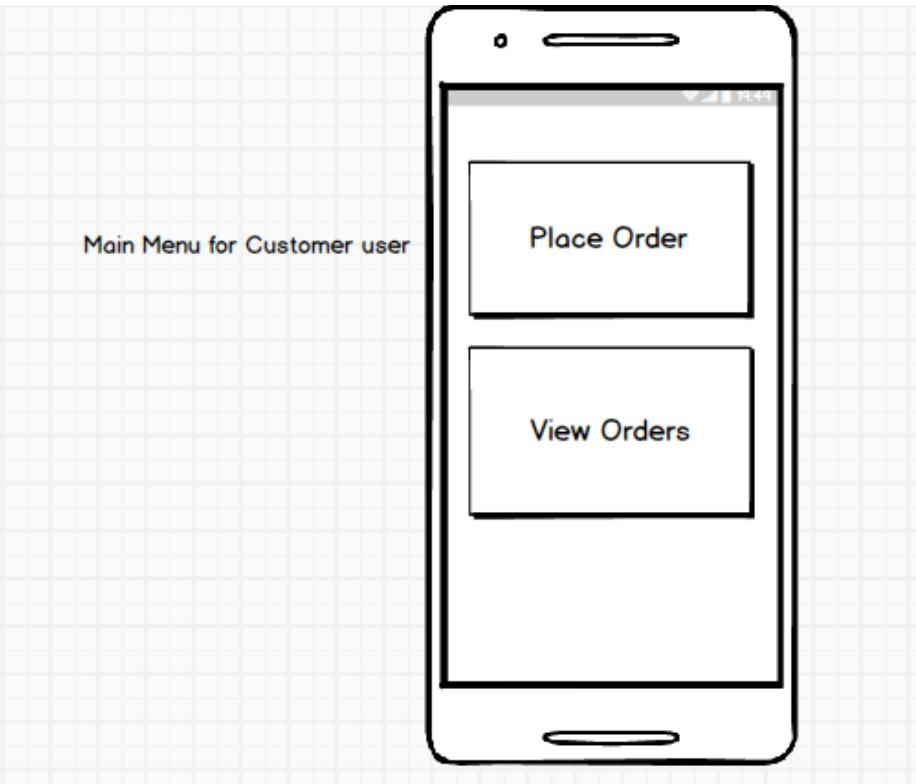


Fig 1: Customer Main Menu

This will be displayed when the customer clicks the place order button

Fig 2: Place Order Menu

This will be displayed when the customer clicks the meals/drinks/deals button

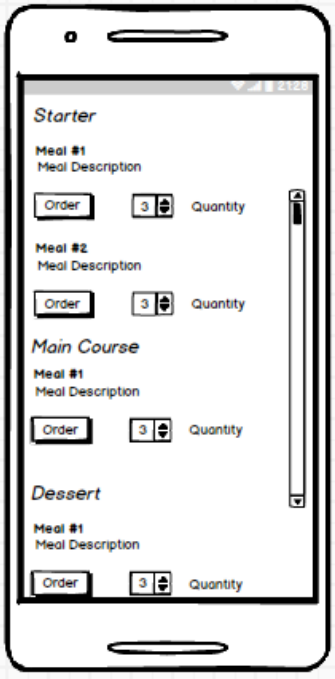


Fig 3: Order Selection Menu

This will be displayed when the customer clicks the view orders button

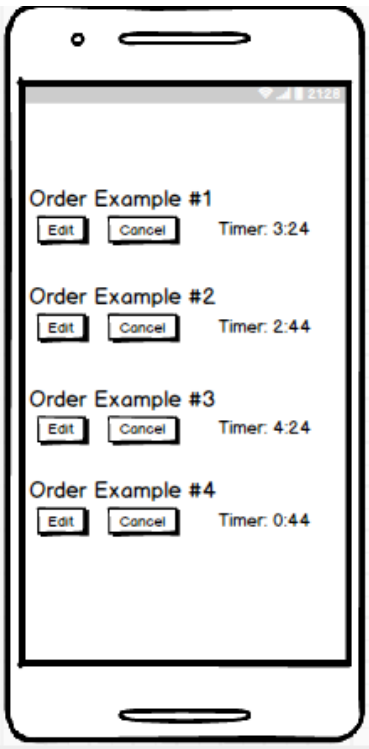


Fig 4: Customer Order List

This is displayed when the staff user click the view orders button

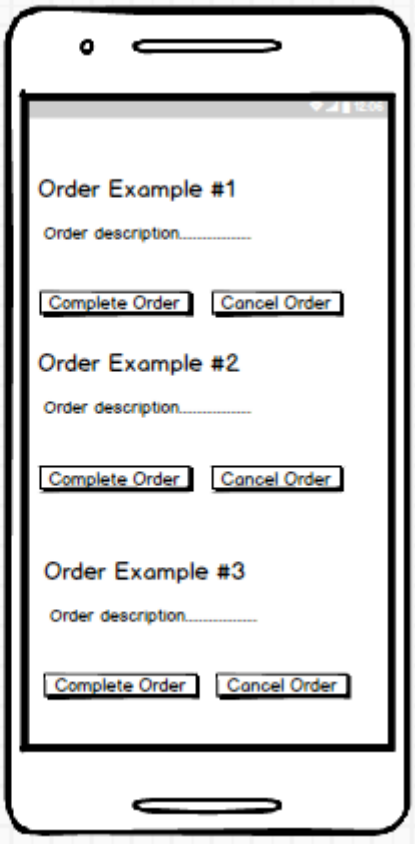


Fig 5: Staff Order List

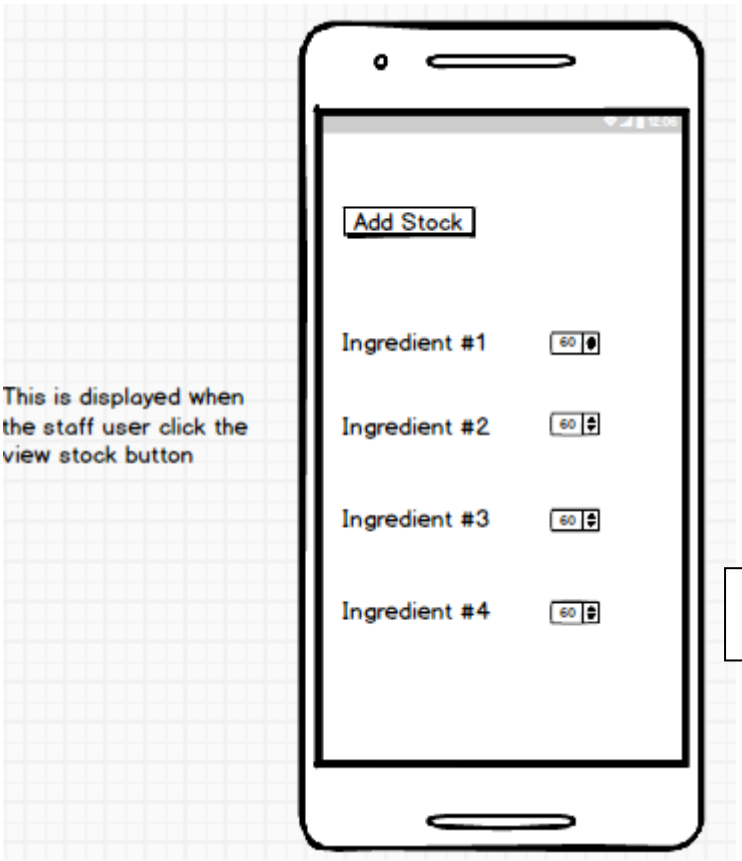


Fig 6: Stock List

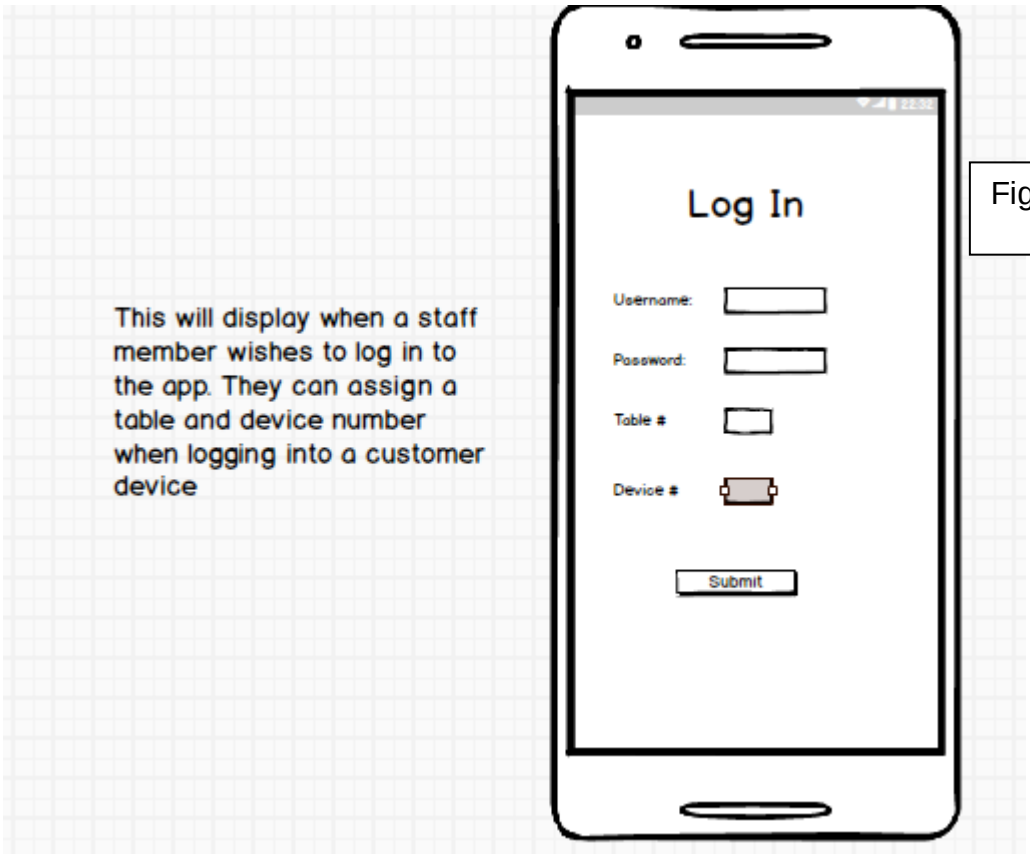


Fig 7: Log-In Menu

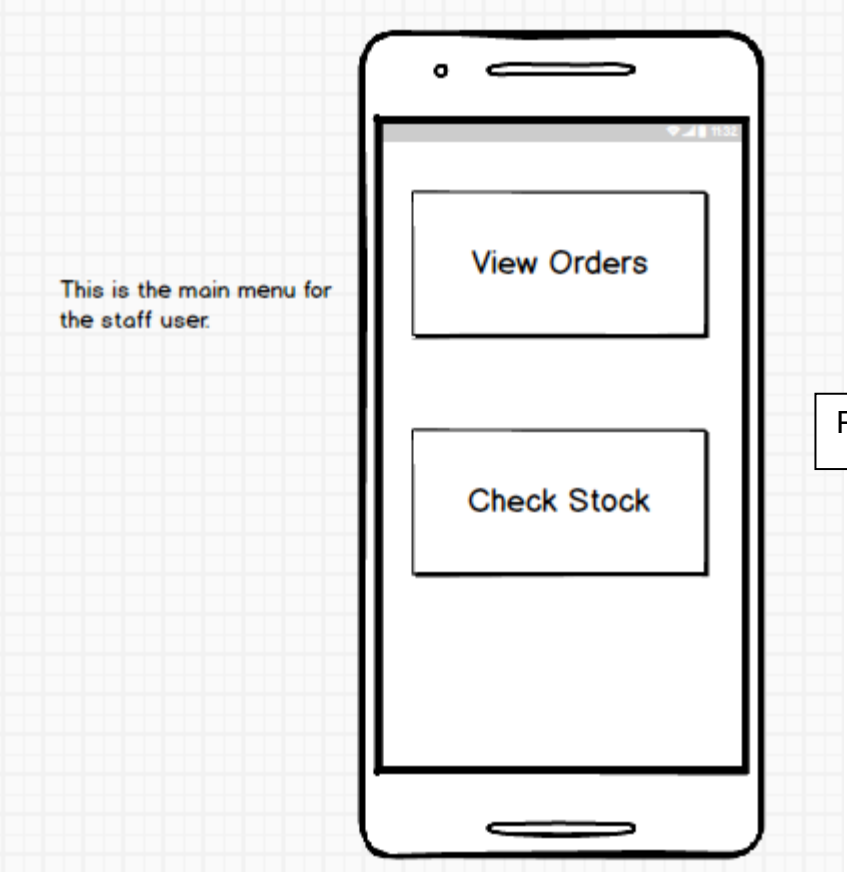


Fig 7: Staff Main Menu

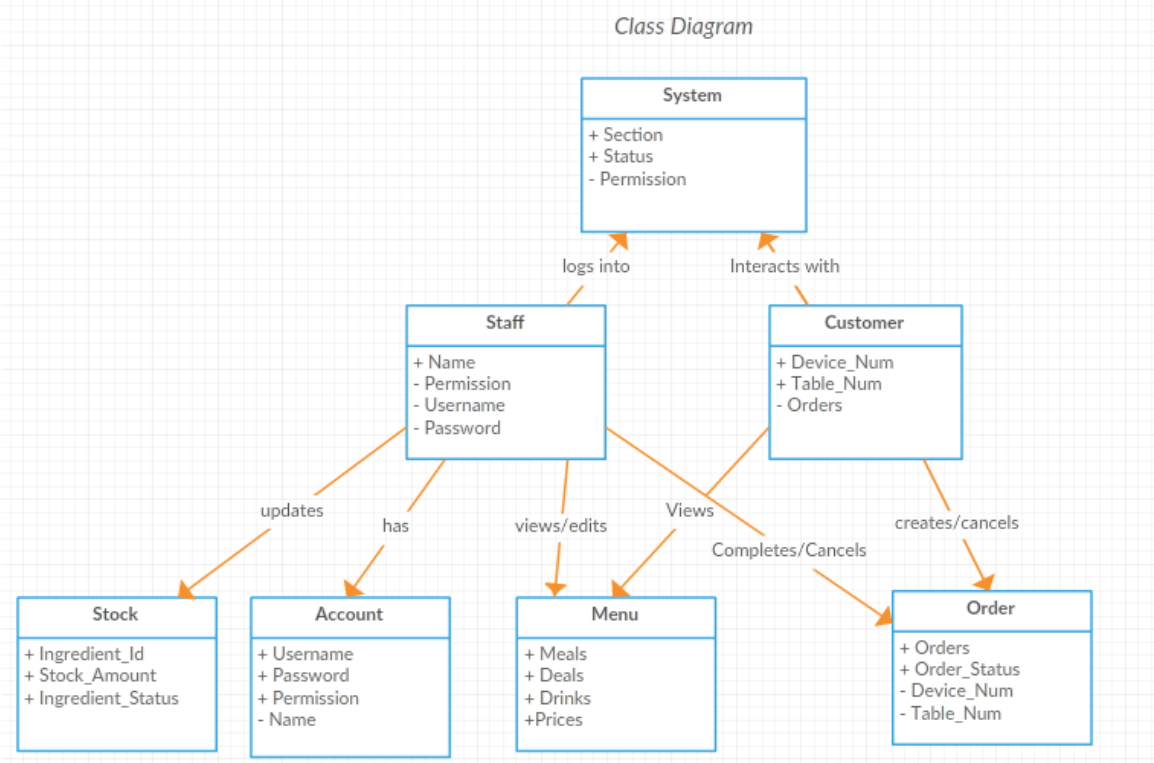
Application Programming Interfaces (API)

Firebase API for the system's server/real-time database.

System Architecture

I have chosen this structure as it displays all classes in a hierarchal state which describes the order in which the classes are called upon.

The classes also contain attributes which will be used in the system's server/database to store needed information.



System Evolution

This system could evolve from being strictly focused on restaurants to being applicable to a large variety of commercial settings.

To take the business Argos as an example, this could replace their method of the customer going through the catalogue and writing down their order number on a slip of paper. The customer would then hand it in at the till so that the staff in the background that are in charge of the stock can go and retrieve it. This system could be modified to allow the customer to view the catalogue on a tablet in the store and their order being sent directly to the background staff which would save time and resources.

6.3 Monthly Journals

Reflective Journal

Student name: Declan Casey

Programme: BSc in Computing

Month: September-October

My Achievements

This month, I thought of an idea to use for my software project that fitted in with the Internet of Things specialisation that I was a part of.

Since it is the first month back in college I had not done a considerable amount of work on the shaping of my project. My contributions to the project included coming up with the idea, considering and researching the technologies that would be needed in order to create it, and attending the project pitch session.

My Reflection

On the first day back to college after the short summer break, I attended the software project class. After a few minutes of the lecturer discussing what sort of things were to come he mentioned that everyone should at least have some sort of idea of what their project will be based on.

After hearing this, I became slightly worried as I had not much of an idea of what I would do. I then started to focus my mind on what sort of projects could be accomplished in my specialisation. By the end of the class I had figured out the foundation for my idea and from there on it started to grow into a more solid one.

I was a bit apprehensive before going in to the present my pitch as I wasn't sure whether the idea was complex or good enough for a final year project. The "dragons" seemed to like the idea but noted that more functionality should be added to the app that I proposed to gain the best mark.

Overall, I'm happy with how these few weeks have turned out as I didn't think I would have a full idea by this stage.

Intended Changes

Next month, I will try to research the ideas that the dragons gave me to improve on my idea.

I realised that I need to add a bit more complexity to my idea as the functionality was slightly limited in what it offered the user.

Supervisor Meetings

No supervisor has been assigned to me yet.

Date of Meeting: N/A

Items discussed: N/A

Action Items: N/A

Reflective Journal

Student name: Declan Casey

Programme: BSc in Computing

Month: October – November

My Achievements

This month, I discovered that my project was approved and was assigned an academic supervisor.

Through attending the IoT lectures and labs I feel as though I have gained valuable knowledge for use in my project especially learning the basics of using the raspberry pi. My contributions to the project included setting up my raspberry pi as well as using the sensors in the GrovePi kit, and working on the requirements specification document for my project that is due next week.

This month was tough as it had several assignments due and programming tests. This meant I wasn't able to do as much research and project preparation as last month.

Through writing the requirements specification I discovered more about the features that my software would have and the steps it would take to complete the tasks that those features require. By doing the use case diagram I realised how the different requirements would work with each other to have a complete software system.

Overall, I'm looking forward to the upcoming weeks where I will be starting the early stages of development on my software project as well as discussing my project with my academic supervisor to see if there is any way to improve upon my idea.

Intended Changes

Next month, I will try to attend several meetings with my academic supervisor to get a solid plan of action for my project.

At this stage in development I have no intended changes to make to my project but I predict that I will have several in the coming weeks.

Supervisor Meetings

I was not able to arrange a meeting with my supervisor as I was sick with the flu the week prior to writing this report.

I have arranged with my supervisor by email to meet next Tuesday.

Date of Meeting: N/A

Items discussed: N/A

Action Items: N/A

Reflective Journal

Student name: Declan Casey

Programme: BSc in Computing

Month: November – December

My Achievements

This month, I started developing my project prototype that will be displayed for the midpoint presentation.

By attending lectures with my academic supervisor I have been given some good information on how to approach both the prototype and the technical report. He

gave me details on what portion of my overall idea would be expected to be completed for this prototype.

This month was very challenging as I had to work simultaneously on several projects as well as study for CAs. This meant my time for developing the prototype was limited.

Through writing the technical report I discovered more about the features that my software would have and the steps it would take to complete the tasks and goals it discussed. By starting development on the prototype I gained a better idea of how the android application will work and what the UI will look like.

I look forward to presenting my project in the coming weeks as well as having a couple weeks off for the Christmas.

Intended Changes

Next month, I will try to continue working on my prototype and adding more functionality to it.

I have no big changes to be made so far in development.

Supervisor Meetings

I was not able to arrange a meeting with my supervisor as I was sick with the flu the week prior to writing this report.

I have had 3 meetings with my academic supervisor since writing the last monthly report.

I have learnt quite a bit about how to approach my project by attending these meetings. I also have learned what is expected for the midpoint presentation, the prototype for the presentation and what to write for the executive summary in the technical report.

Date of Meeting: 15/11/2016, 08/11/2016, 06/12/2016

Items discussed: Prototype, Technical Report, Implementation, Requirements Specifications Document, and Midpoint Presentation.

Action Items: Requirements Specifications Document, Prototype, Technical Report

Reflective Journal

Student name: Declan Casey

Programme: BSc in Computing

Month: December - January

My Achievements

This month, I completed development of my project prototype and attended my midpoint presentation. I also started studying for upcoming exams.

This was a very stressful month as I had to complete several projects for continuous assessments, I had to complete my prototype as well as prepare for exams.

During development of my prototype I encountered several problems and had to make a second, simpler version of the prototype so that it would be functional in time for the presentation.

Not much progress has been made in regards to the project this month as it was packed with a lot of other work.

Intended Changes

Next month, I will start testing on my software as it is being developed and start adding more functionality to it.

I was advised in the midpoint presentation to add more unique selling points to my project.

Supervisor Meetings

N/A

Reflective Journal

Student name: Declan Casey

Programme: BSc in Computing

Month: January- February

My Achievements

I started back in college this month for the second semester following a week and a bit off. I used the week off to rest my mind after a tough few weeks of studying for the exams.

I attended my new modules this month which seem both interesting and difficult as I have to learn two programming languages (Python and Ruby/Ruby on Rails) which I've done little of before.

I was a bit lost as to where to continue on my software project after a long break from the development process. I started re-designing the UI of the android application to make it more user friendly by making the different courses easier to view/select.

I also uploaded the code base onto GitHub and setup GitHub integration into Android Studio. I also setup the FireBase Server in Android Studio in preparation for when databases will be setup.

I have started testing on my software and plan to continue it at regular intervals.

Intended Changes

Next month, I plan to meet with my academic supervisor to discuss strategies for how to approach certain problems and deadlines. I also plan to add functionality that wasn't initially proposed such as adding a time limit which will reduce the cost of the meal depending on how overdue it is.

Supervisor Meetings

I plan to meet with my academic supervisor for the first time of this semester next week.

Reflective Journal

Student name: Declan Casey

Programme: BSc in Computing

Month: February-March

My Achievements

I've made quite a bit of progress on my project this month.

After working on my first CA for IoT Application Development I've gained increased knowledge of working with the Raspberry Pi which I think will help my project.

I have redesigned the UI on my Android application to make it more user-friendly and easier to use and have functionality for order to be sent and retrieved to/from the Firebase server.

I setup several new branches in GitHub to isolate my changes from the master branch and proceeded to merge them into the master branch once they were fully working and bug-free.

I have been continually testing my android code throughout development.

Intended Changes

Next month, I intend to add log-in functionality, start layouts for the staff users and start on code for the Raspberry Pi.

Supervisor Meetings

Date of Meeting: 15/02/2017

Items discussed: Testing, GitHub integration, How to achieve the best results (marking scheme).

Action Items: Testing, GitHub integration.

Reflective Journal

Student name: Declan Casey

Programme: BSc in Computing

Month: March – April

My Achievements

I've made further progress on my project as well as several other projects this month.

I've implemented login authentication to my project as well as cart functionality. I have also started work on the staff side of the application and its associated functionality; such as retrieving customer orders from the database and displaying them in a listview.

After working on my second CA for IoT Application Development I've gained the knowledge of how the AWS (Amazon Web Services) system works by sending/receiving JSON data. I believe this information will be helpful when I'm developing my Raspberry Pi section of the project.

I have continuously uploaded my project work to github with new branches for new functionalities added etc. I have also started User testing by giving surveys/evaluations to friends so that they can give feedback on the application and I can improve it based on their comments.

Intended Changes

Over the course of the next few weeks, I intend to complete my project, upload it to Moodle and present it in the final presentation. I can't believe I'm nearly at the finish line!

Supervisor Meetings

Date of Meeting: 29/03/2017

Items discussed: User Testing, Unit Testing, Final report.

Action Items: User Testing, GitHub integration.

6.4 User Manual

To create a staff account in the system, sign up with an account containing the word staff in the email.

Sample Sign In

Staff account:

Username/email: staff1@waiteraider.com

Password: password

Customer account:

Username/email: seat1table1@waiteraider.com

Password: foobar

Website address for the web application:

https://waiteraider-ee9e1.firebaseapp.com/