H446-03 PROJECT
MOBILE POS SYSTEM

Samuel Jacob Herodotou
Candidate Number: 1210

Centre: Sandringham School
Centre Number: 17535

Contents

Analysis

1.1 Problem Description

Most restaurants today use some form of electronic POS (point-of-service) system, which is used to keep track of orders and tables, manage payments and allow orders to be sent to the kitchen. Although useful, many POS systems can be quite inefficient to use in practice.

My project will be an online-based, paperless restaurant POS system, which will have the ability to be accessed and interacted with via any Android device, anywhere and anytime provided an internet connection is available. It will aim to improve the efficiency of table service in restaurants by allowing orders to be taken and sent at the table, rather than via traditional methods which involves writing the order down and then inputting it into a terminal located somewhere else in the restaurant. This not only saves time but reduces the chance for human error to occur when placing orders, improving accuracy and consequently customer satisfaction.

Once an order is placed, it will be available to see instantly by the relevant staff (such as the chef/barista), allowing them to prepare the order as quickly as possible. Once completed, the waiting staff can be notified, making communication between staff significantly easier.

The system will also support an administration mode, allowing the owners to view statistics and configure the system. This will be supported by a login system where each user has individual access rights and a respective username/password.

1.2 Stakeholders

The primary stakeholders to this project will be the restaurant staff- specifically waiting staff, chefs, baristas and owners. The system provides an interface in which these staff can complete their jobs with ease and assists with the communication between them, allowing the process of taking orders, preparing orders and serving them to the customer to be completed as efficiently as possible. For example, the chefs and baristas will be able to view orders as soon as they are sent and mark them as complete. Serving staff will be able to place orders as the customer is making it, and does not need to worry about missing details or running out of ink or paper. On top of this, they do not need to keep checking the kitchen to see if orders are ready since they can be remotely notified via the app. For the owners, the system provides a platform for them to both manage their restaurant and view vital statistics (such as what menu items are most popular) which will help them to maximise profits and improve their business as a whole.

The customer is also a key stakeholder since the accuracy of their order and how quickly they are served is vital to their satisfaction. The system attempts to maximise these aspects of their dining experience by allowing the staff to work as effectively as possible.

For this project, I will be working with the owners and staff from a local Italian restaurant ‘Ballo’ to assist with providing the requirements and feedback of the system.

1.3 Why the problem can be solved by computational methods

My proposed solution to the problem requires computer systems in order to function since it requires the handling and processing of digital data (rather than physical), and communication via the internet. The benefits of using a digital system as opposed to a manual ‘pen-and-paper’ approach have been discussed already. To enable my software to work with computer systems, computational methods such as thinking ahead, abstraction, procedural, logical and concurrent thinking must be applied during the design and implementation phases of the development lifecycle.

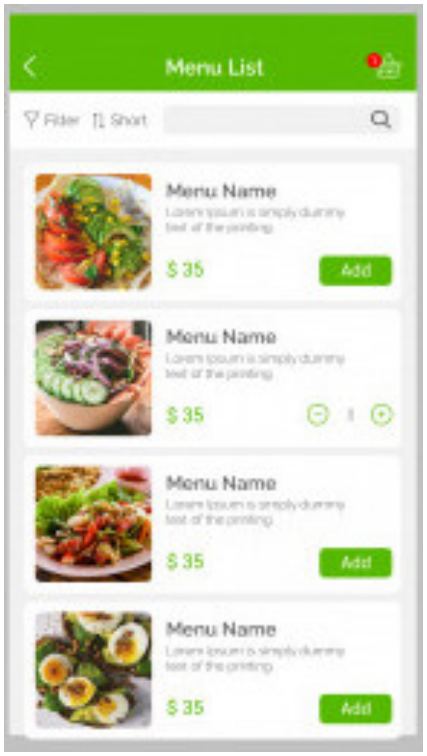
Abstraction

Abstraction is the process of removing unnecessary detail from an object or concept. For this project, I will be using abstraction to simplify the development process and to make the application more intuitive for the user.

For example, when displaying menu items for the waiter to input as the customer is making their order, only necessary information such as the name and price will be shown to make the process less complicated. Hiding extra information such as the item description prevents the waiter from being overloaded with information which would slow them down.

Here is an example of abstraction being used in this scenario from a POS system currently available:

As you can see, only the item name and price is displayed. The simplicity of this makes inputting an order much easier compared to an interface which is crammed with unnecessary information such as the following:



Abstraction will also be used when handling information. Useless information (with respect to the purpose of the software) such as the customer's middle name will not be stored or processed at any time. This not only saves both computational and storage resources but makes the development process simpler as less code has to be written to handle this information.

Thinking Ahead

Since the software which I will be developing is very data orientated, it is vital that I plan out the structure of the database which will be storing all the information. This planning will involve identifying the information which is necessary (and not) to store, the data types that they will use and the structure and relationship between the tables which will be within the database.

For example, when developing the user system, there will be a table 'Users', which would store at a minimum their respective ID (integer), username (string), password (hashed string) and display name (string).

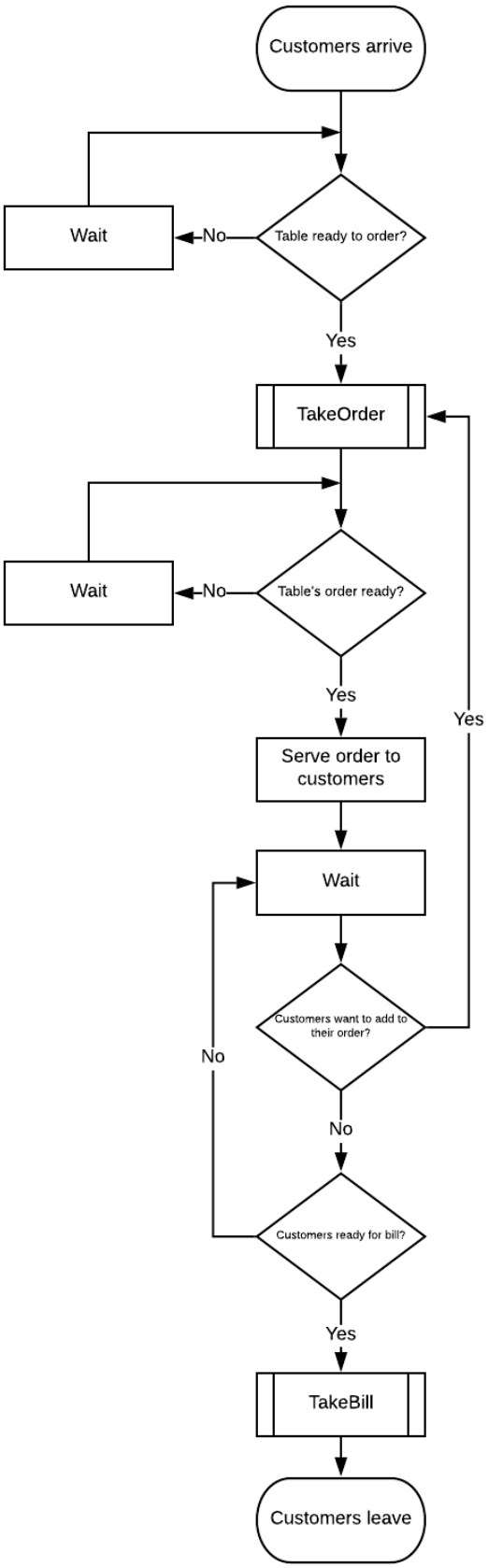
I will also need to plan in advance the design of the UI and the algorithms which will be working in the background in order to make the system as intuitive as possible whilst maximising computational efficiency and robustness.

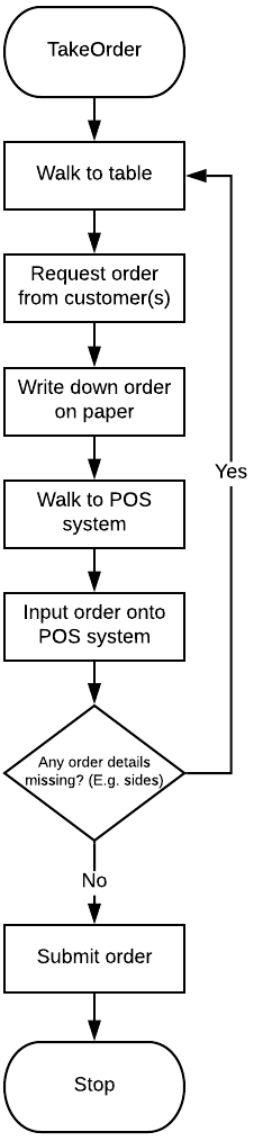
Planning how to handle all possible inputs from the user is also necessary to prevent problems occurring in the future which would cause the software to behave inappropriately or crash. All inputs will be ‘pointer’ input from the touchscreen of the mobile device and textual and numerical input through the use of a software keyboard.

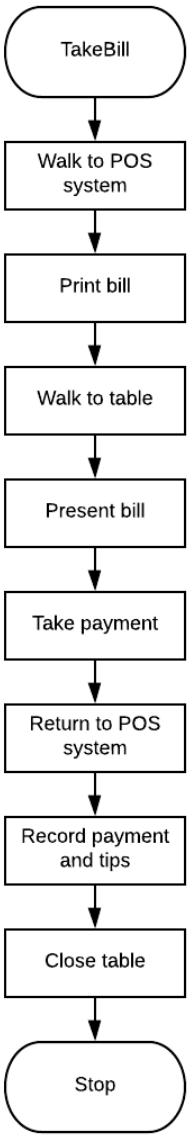
Thinking Procedurally and Decomposition

In order for the software to assist with the tasks that the restaurant staff undertake on their shifts, their workflows need to be studied. Thinking procedurally involves developing an ordered sequence of instructions which give achieve a certain goal. Decomposition involves breaking down a larger problem into smaller, more manageable sub-problems. I have used procedural thinking and decomposition alongside with my knowledge of the roles of chefs and waiters to produce the following flow charts for their respective duties:

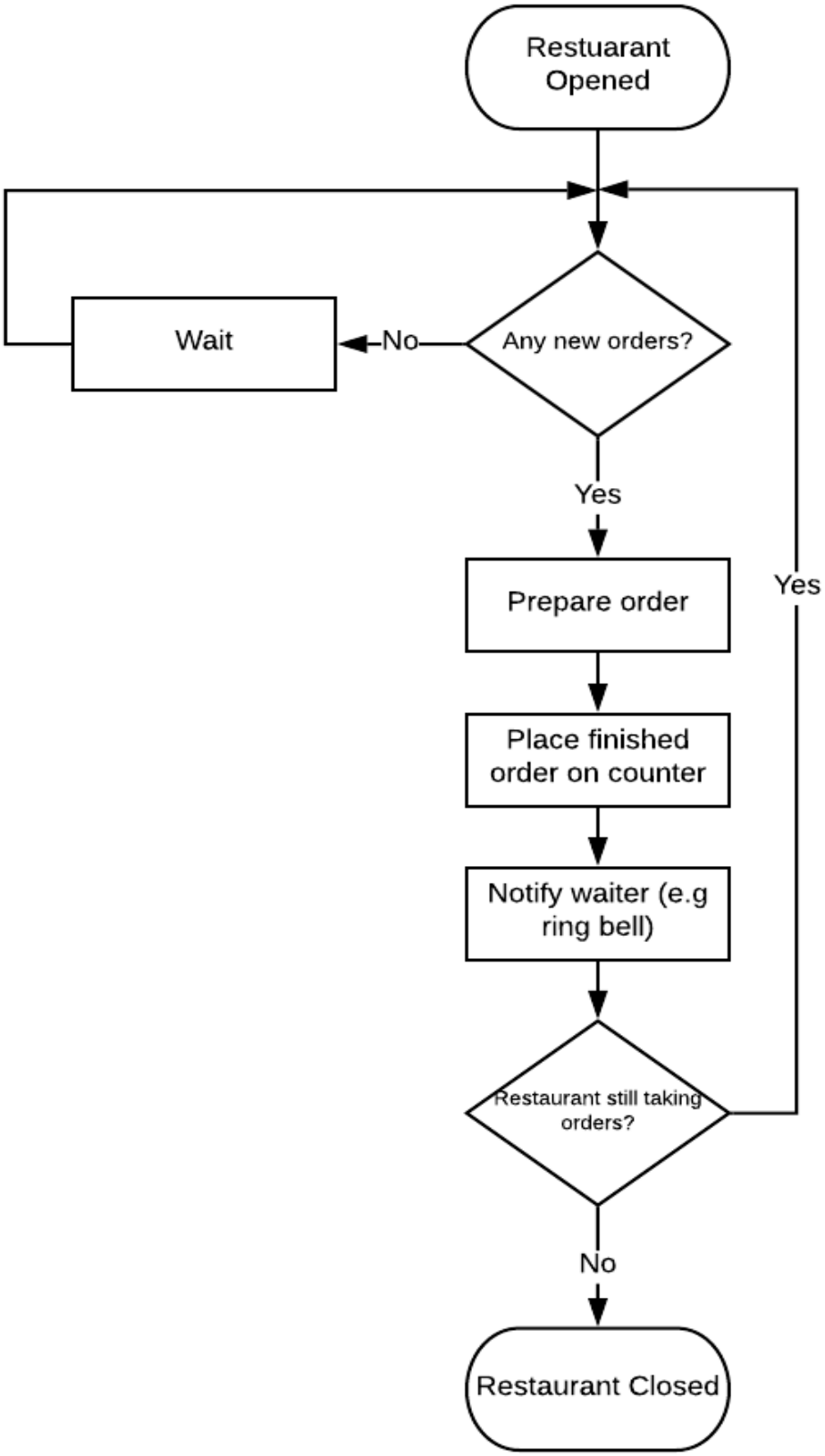
Waiter:


Customers arrive
Table ready to order?
No
Wait
Yes
TakeOrder
Table's order ready?
No
Wait
Yes
Serve order to customers
Wait
Customers want to add to their order?
Yes
No
Customers ready for bill?
No
Yes
TakeBill
Customers leave


TakeOrder
Walk to table
Request order from customer(s)
Write down order on paper
Walk to POS system
Input order onto POS system
Any order details missing? (E.g. sides)
Yes
No
Submit order
Stop


TakeBill
Walk to POS system
Print bill
Walk to table
Present bill
Take payment
Return to POS system
Record payment and tips
Close table
Stop

Chef:


Restuarant Opened
Any new orders?
No
Wait
Yes
Prepare order
Place finished order on counter
Notify waiter (e.g ring bell)
Restaurant still taking orders?
Yes
No
Restaurant Closed

The use of subprocedures in the waiter's flow chart demonstrates how decomposition has been used to break down the problems of 'Taking an Order' and 'Taking the Bill' into their own respective sub-problems. The sequences of instructions within them demonstrate how procedural thinking was used to devise the steps and the order in which they are carried out in order to solve the sub-problems. The importance of the order of instructions can also be seen here, for example you cannot present the bill to the customer before walking to their table.

Thinking Logically

The software must behave differently for different inputs given to it. Therefore logical thinking must be applied in order to define the conditions and pathways which different inputs will cause the program to follow. Some logical thinking has already been demonstrated through the flowcharts seen previously with the use of 'IF' statements. Using the waiter as an example, one decision is based on 'Is the table ready to order?'. If the answer is 'Yes', then the waiter will start to take the order. If not, the waiter will wait and then return back to the original IF statement. This is also an example of a condition-controlled loop.

Thinking Concurrently

Concurrent thinking involves considering processes which occur simultaneously. This is a necessity for the application since parallel processing maximises efficiency and saves time. For example, requests made to the server must be made in a separate thread to the UI thread, which handles user input and displaying information to the user. If this was not the case, the application would be unresponsive and freeze up since the processor would be focused only on communicating with the server. This problem will be amplified in situations where the requests take longer to finish, such as slow internet or heavy load on the server. As a result, the application will be unusable which is a severe issue in practice since it would waste time, defeating the whole purpose of developing a more efficient system.

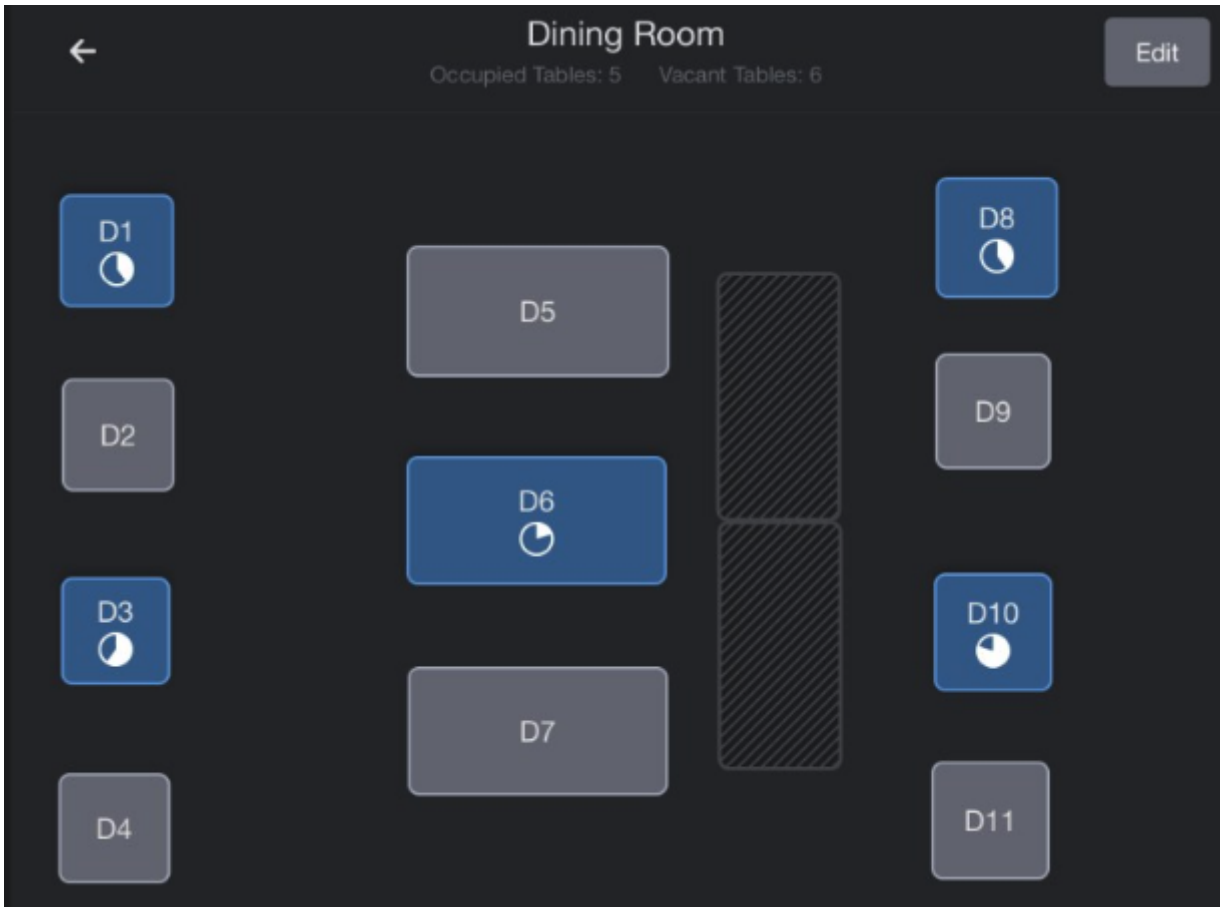
Thinking concurrently is also a necessity when considering multi-user systems. Since the system will be used by several users simultaneously, the database and backend must be able to fulfill requests without corrupting data or behaving unexpectedly. An example of a solution involving concurrent thinking in this situation is the use of database transactions. This ensures that data is not corrupted when the same field in a database is accessed or modified simultaneously.

1.4 Research into the current system

The restaurant I am working with currently uses the traditional 'pen and paper' approach to taking orders. They have a single POS terminal located near the kitchen, and after writing down each order, it then has to be inputted into the POS system as well. The flowchart for the waiter (above) is based on this system.

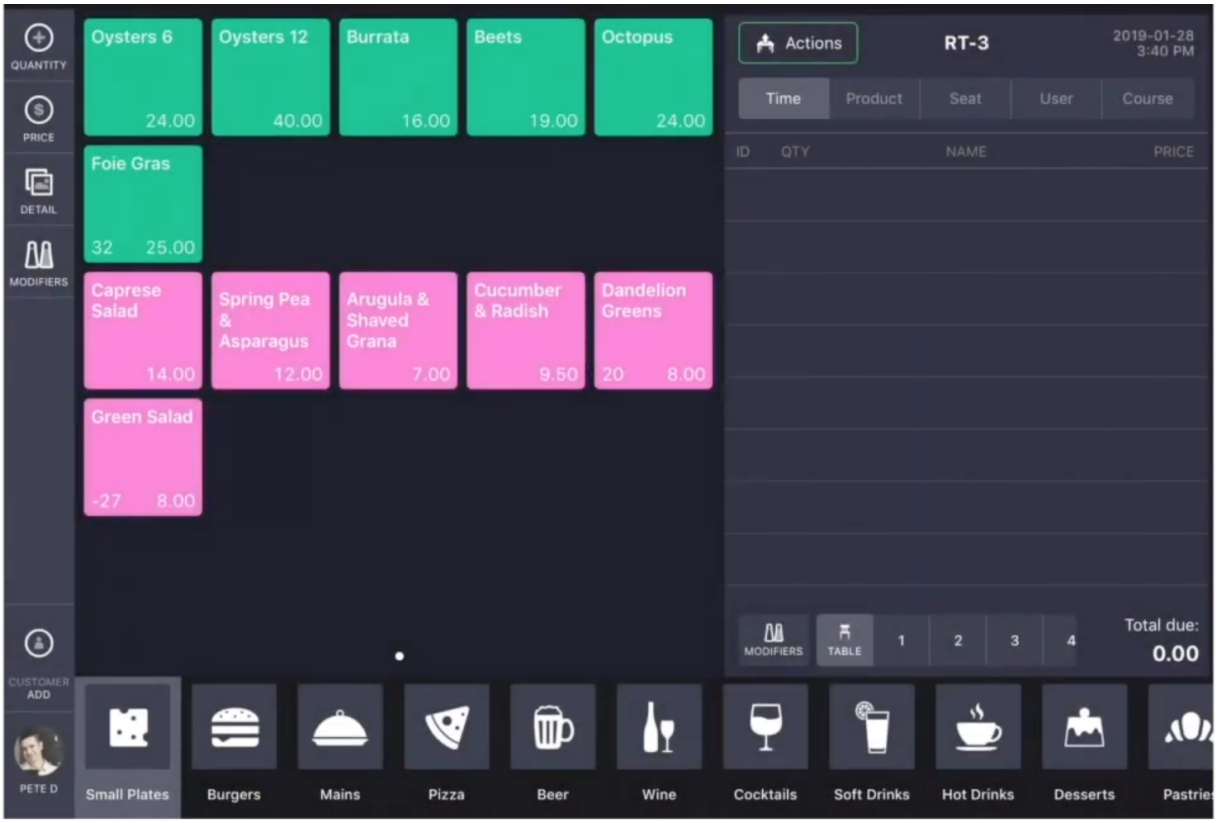
The POS terminal features a touchscreen and an NFC reader. It is connected to multiple printers which are located in the kitchen and bar (for sending orders) and one at the till which is used for printing the bill. The NFC reader is used as a security measure to prevent unauthorised access to the system. Staff members have their own token which they place on top of the reader which then allows them to use the system.

Once authenticated, a screen with a virtual table layout of the restaurant is presented.



An example of a virtual table layout view

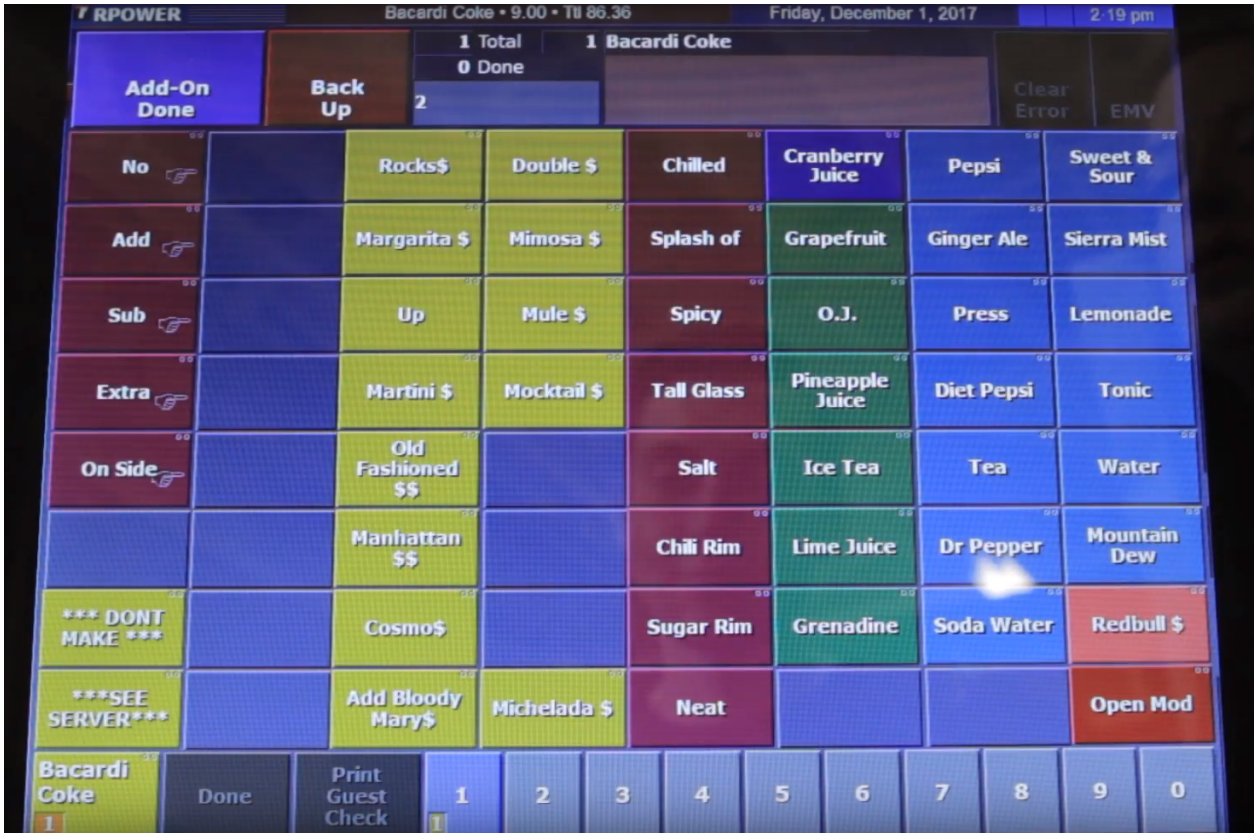
On selecting the desired table, the waiter/waitress can input the order that they have been given.



Example order screen layout

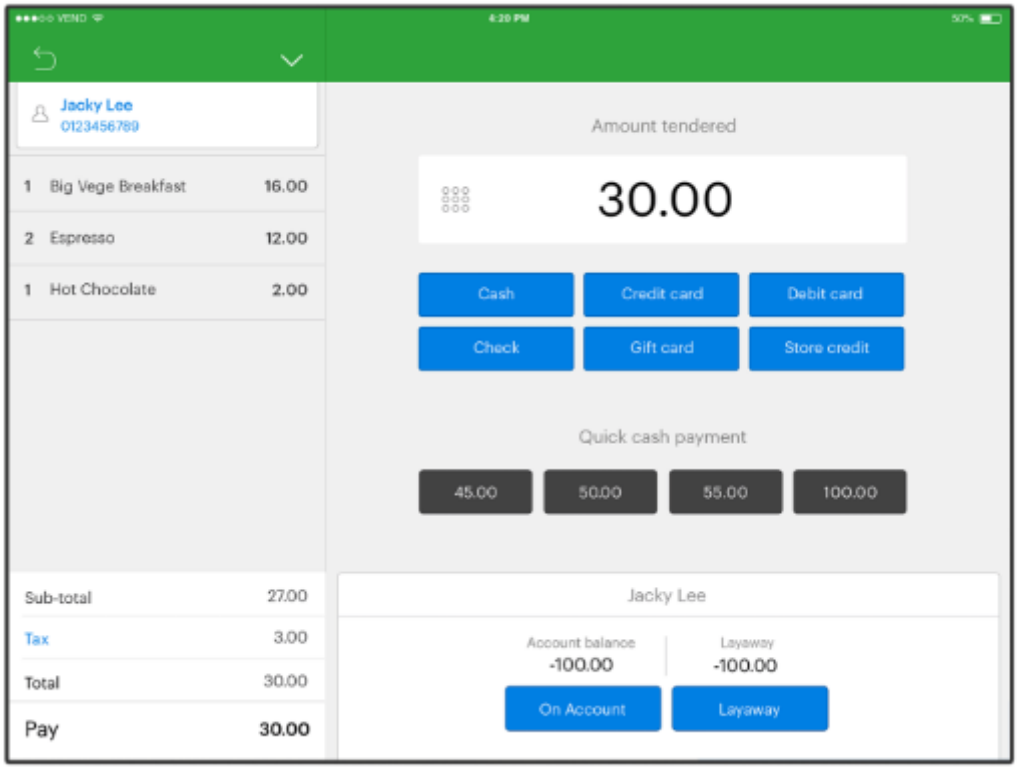
The key features of the order screen can be seen in the example above. The left-hand side displays the menu options, whilst the right-hand side shows the order summary. The menu is organised into categories, which can be seen at the bottom. Once the order has been inputted, the waiter/waitress will be able to send it to the kitchen staff (via the printer). Orders can also be cancelled or modified in case the customer changes their mind.

An important feature of the menu display is having item addons. This is the ability to add additional information to a product within an order, such as choosing a side or adding extras (such as ‘extra ice’ for a drink). Each addon will have its own associated cost since not all extras may be given out for free. Each item along with its ‘extras’ (where applicable), will have a quantity attribute also.



Addon screen from an existing POS system

Since the POS system is designed for table-service restaurants, it does not require payment at the time of order. The restaurant staff can switch between open tables and create new orders or add to existing ones whenever necessary. A table can only be closed once it has been paid for fully. When taking payments, the POS system accounts for different payment types. The waiter/waitress must input how much money has been paid via cash and card, and it will calculate if there is any money left that is owed. Additional payments such as tips can also be recorded. All orders are stored in a database and can be accessed by the owners for analytical/accounting purposes.



An example of a payment screen. It features 'quick cash payment' buttons for convenience and supports multiple payment methods. A summary of the order is also available on the left, which is helpful if any mistakes were made when taking the order (e.g overcharged for something).

The system also features an administration panel. It has the ability to view reports and configure the system. The report displays total income and sales figures for menu items. Configuration options include modifying the menu, managing users and editing the virtual table layout.

Interview

I conducted an interview with one of the waitresses at 'Ballo', Eva Byers, to help with some of the requirements since the system will be primarily used by waiting staff. The following is a transcript of our conversation:

If you were introduced to a new POS system, would you like it to look and feel similar to the one you work with currently?

Yes, the system I use now is very easy to work with. I would want a new POS system to work similarly so that I could get the hang of it quickly.

With your current POS system, what features do you find the most helpful?

The organisation of the menu into categories makes inputting the orders fast and simple. Also when taking payments, all calculations such as working out change are done automatically, which is very helpful.

With your current POS system, what do you find the most inconvenient?

When taking an order, it first has to be written down and then inputted into the POS system which is located somewhere else in the restaurant. It feels a bit unnecessary since every order has to be recorded twice.

Are there any features that you feel are missing from your current system, or would be helpful to have?

There is no easy way of being notified when orders are ready. We have to check the kitchen frequently to see if any tables are ready to be served, so having some form of communication between the kitchen and the waiting staff would be helpful.

Have you encountered any problems when working with your current POS system, and if so what are they?

When I first started working here and wasn't familiar with the menu, I missed out a lot of order details when taking them from the customer. I only noticed this after I had walked away from the table to the POS system, where it then asked me for information which I didn't know (such as a side they wanted which came with the meal). This was quite embarrassing since I had to walk back to the table and ask the customer again for more information.

I also asked the restaurant owner, Nino Bianchi some questions:

As a restaurant owner, what statistics would you find most useful to help grow your business?

Most importantly, I would want to see sales figures for each menu item and be able to order them from most popular to least, etc. Viewing income for different time periods would also be helpful.

With your current POS system, what administrative tools do you find most useful?

The ability to view reports is very helpful. Tracking incoming and outgoing cash and card payments also helps us with tax calculations and prevents any cash from being removed from the till without being noticed.

Are there any features that you feel are missing from your current system, or would be helpful to have?

I would like to be able to view reports remotely rather than having to view them directly at the terminal since it is always too busy to view them during working hours.

Have you encountered any problems when working with your current POS system, and if so what are they?

We usually have a lot of problems with the printers. They can be unreliable at times and we have to frequently check that enough paper is remaining in them so we do not run into problems during service.

Conclusion

Based on the responses to the interview questions, I will try to keep the system that I develop as similar to use as the current system. The key difference will be that instead of having a fixed terminal acting as a hub, each staff member will be able to access the system using a mobile device from any point within the restaurant. This also removes the need for printers to aid with communication. To support a 'paperless' environment, the application will email receipts to customers rather than print them. To enable the kitchen staff to view orders, the application will have an order list screen, where individual orders and their items will be displayed in detail and will have the ability to be marked as complete once they have been finished.

Additional features, such as being able to mark orders as complete and the ability to view statistics anytime and anywhere will also be added since these were desired.

I will be developing the application for Android devices. The primary reason for choosing Android is the fact that the operating system is available on a much greater range of devices compared to other mobile operating systems (since it is open-source). On top of this, there are many cheap Android devices available (unlike iOS devices), making the solution much more affordable for restaurant owners, especially since the system requires multiple devices in order to function.

1.5 Features of the proposed solution

Based on my research and project proposal, the following features will be included in the system:

Feature	Explanation	Justification
Virtual table view	A virtual display of the restaurant's table layout. Each table is numbered, and on selection will allow the user to manage the table's order.	This feature makes selecting a table easier since it mimics the physical arrangement of tables in the restaurant. Allows waiting staff to manage order information for any table when required.
Virtual menu view	A simplified view of the restaurant's menu. Items are grouped into colour-coded categories.	This is used by waiting staff when taking orders. It allows them to quickly find the desired item and add it to the table's order.
Order list view	Displays all open orders which have not been completed. Each individual item can be viewed in detail (such as viewing sides/extras/special requirements). The list is updated automatically so new orders appear without the need to refresh.	Used by kitchen staff to allow them to view and prepare orders.
Ability to mark order items as 'completed'	Part of the order list view, it allows kitchen staff to mark items within an order as 'completed'.	This allows waiting staff to be notified that the item can be collected and served to the customer. This feature was requested by Eva during her interview.
Order summary	Displays a list of items which have been added to the order along with the total price.	Allows waiting staff to confirm the order with the customer and helps to prevent mistakes when taking the order.
Payment handler	Ensures that the bill has been	Prevents customers from

	paid fully and is responsible for storing and handling transactions (e.g. calculating change). Supports additional charges, tips and discounts.	being able to leave the restaurant without paying unnoticed. Simplifies the process of calculating change and ensures all transactions are recorded in the database.
Online database storage	All data that is shared between devices during service is stored remotely on a database. The database stores users, orders, payments, statistics and any other information required by the service. It can be accessed anywhere in the world as long as an internet connection is available.	Allows data to be accessed anytime and anywhere whilst storing it permanently. Used both during restaurant service (e.g. accessing open orders) and outside of service (e.g. viewing reports).
Login system	Allows restaurant staff to login using a username and password.	Prevents unauthorised access to the system. Allows for the separation of roles (e.g a waiter can manage/create orders whereas a chef can view orders and mark them as complete).
User permissions	Allows individual users to have their own access rights. (E.g the owner can configure the system)	Prevents all users from accessing the entire system.
Menu editor	Configuration tool which allows for the modification and creation of individual order items, item categories and item extras (such as sides).	Allows for the menu to be modified easily. Without this feature, the database would have to be modified manually which is slow and requires technical knowledge.
Virtual table editor	Configuration tool which allows the virtual table layout to be modified.	The user can change the layout of the restaurant without having to worry about the system's layout being out of date.
Statistics screen	Displays reports to the user. Includes total income and sales statistics for menu	Allows the owner to keep track of their business. Helps with managing stock and

	items.	accounting. This feature is also available in the restaurant's current system and was considered helpful by the owner in his interview.
Menu item extras	Allows additional items and requirements to be associated with each individual order item (such as sides). Also includes the quantity of the item and any extras associated with it.	Supports more complicated orders and gives the customer more freedom.

Limitations

I will not be including some features which are available in the current system due to restrictions with both time and resources.

Feature	**Reason for not including**
Cancelling orders	To save time, this feature will not be included. It is very rare for this feature to be needed during service, so it is not necessary to include.
Modifying individual order items	It would be hard to track changes to orders and would involve making the system significantly more complicated, so is not included due to time restrictions.
Printing the bill to customers	I do not have access to a printer system, so will not be adding support to print the bill (which some customers may want instead of being emailed it).

1.6 Software and hardware requirements

As a whole, all devices working within the system will require access to an internet connection.

Runtime Requirements for the Application

The requirements below are for the Android device running the application.

Hardware Requirement	Justification
1.2 GHz CPU or faster	In order for the Android OS and the application to run smooth enough to be usable, the CPU must have a clock speed of at least 1.2GHz.
6GB or more internal storage capacity	The Android operating system requires around 5GB of storage. The application will be smaller than 1GB, so 6GB is sufficient for both.
4GB of RAM	4GB of RAM will provide enough memory for the runtime components of both the operating system and the application (such as instructions and data).
Touchscreen	The touchscreen allows the user to interact with the application by detecting touches and is also used as a visual output device (used to display the UI of the application).
Speakers	Notification sounds require a speaker in order to be heard by the user.
Network Interface Card	A NIC is required to allow the device to connect to the internet. The application requires an internet connection in order to work.

Software Requirement	Justification
Android 4.4 (KitKat) or higher	With this version of Android, the developer SDK supports many features which will make creating the application easier. Being an older version, it is widely available, allowing lower-end devices to use the application.

Requirements for the Backend Server

These requirements are for the server running the database and API.

Hardware Requirement	**Justification**
4 x 1.6 GHz CPUs	The server software and OS which will be running on the system requires 4 cores to allow for requests to be handled efficiently and to allow for parallel processing (e.g handling web requests and executing database queries simultaneously). A clock speed of 1.6GHz will allow instructions to be executed at a sufficient rate.
7GB RAM	The server software and OS which will be running on the system requires a minimum of 7GB of RAM to run.

Software Requirement	**Justification**
MySQL Server	The MySQL server is responsible for handling requests to the database and storing data.
PHP 5+	The PHP runtime is required to interpret PHP code which is executed during web requests to the server. Version 5 of PHP is widely supported and has many useful libraries available.
Apache Web Server	The web server is responsible for handling web requests which is required for the API. Key for communication between application and server.
Visual C Runtime	Required for the PHP server to run on the system.
Linux OS	All the software above is able to run on a Linux system. The hosting service which will be used uses Linux-based servers.

Requirements for the Development Environment

I will be using Android Studio to develop the mobile application and Visual Studio Code to develop server-side code. My operating system is MacOS.

Hardware Requirement	**Justification**
6GB RAM or higher	Android Studio requires 4GB RAM at a minimum to be able to run. The additional 2GB is to enable the emulator to run smoothly.
6GB disk space minimum	The IDE, SDK and emulator OS images require a total of 6GB in storage space.
1280 x 800 minimum screen resolution	In order for all the tools/icons within Android Studio to be accessible and visible to the user, the resolution of the monitor must be at a minimum of 1280 x 800.

Software Requirement	**Justification**
MacOS 10.10 or higher	Both Android Studio and Visual Studio Code require MacOS Yosemite (10.10) or higher to run.
Android SDK (API level 19+)	The SDK provides development tools, build tools, documentation and emulator software which is vital for developing and testing the application.
Java SE JDK	The JDK (Java Development Kit) provides build and runtime tools (such as a compiler) which are required to develop Android applications since they are coded in Java.

1.7 Success criteria

The requirements of the system can be seen below. Their importance will be from 1-3, where 1 is necessary, 2 is important and 3 is desirable.

Number	Feature	Justification	Importance
1	The system is able to	If order information	1

	take orders and store them correctly in a database	is not stored properly, it will not be able to be accessed by other users of the system.	
2	Kitchen staff can view open orders in real-time	Customers' orders will not be able to be fulfilled if the kitchen staff are not aware of them.	1
3	The system will be able to be accessed 24/7 by anyone with an internet connection	Although necessary during service, it is useful to be able to access the system outside of working hours (e.g for the owner viewing reports).	2
4	All requests to the server should be fulfilled within 1 second	The application and backend must be efficient enough to feel responsive and not waste time.	2
5	The system should not be able to be accessed without authentication	Unauthorised access to the system could lead to theft or a breach of confidential information.	1
6	The application and backend should be robust and not crash	If the application or backend crashes or encounters errors, there will most likely be a significant disruption of service.	1
7	All user inputs should be validated and sanitised sufficiently	By validating and sanitising user inputs, it is much less likely for there to be security breaches and errors.	1
8	The user interface	An easy to use UI	2

	should be simple and intuitive	makes learning how to use and working with the system much easier, saving time and effort.	
9	Waiting staff will be notified when orders have been completed	Waiters no longer need to check the kitchen to see if orders are ready to be served, saving time.	3
10	A table cannot be closed until it has been paid for fully, and all payments should be recorded.	By tracking owed money and payments, it makes it much harder for theft/disputes to occur and makes accounting easier.	1
11	Sales figures and statistics are recorded and can be viewed within the application	By recording sales figures and other statistics, it helps the owners to manage stock and improve their business.	2
12	Multiple users can access the system simultaneously	The system requires the use of multiple devices which communicate with each other via a central server. Therefore the individual staff members need to access the system at the same time.	1
13	The application will activate the correct features based on the role of the user. For example, waiting staff will have the ability to create orders, kitchen staff	If the user was given access to the wrong part of the system then they would not be able to fulfil their duties properly.	1

	can view orders and owners can manage the system		
14	The application and server will keep track of login sessions by storing authentication tokens	This prevents the need for the user to keep logging in to the system (and authenticating for every request) which would waste significant time.	1