COMP 4

Tommy Tham

March 5, 2015

Contents

Chapter 1

Analysis

1.1 Introduction

1.1.1 Client Identification

My client is Linh Tham, the owner of Linh's Restaurant. Linh's Restaurant is a family run Chinese restaurant that is situated in small village called Fordham in Cambridgeshire. Linh works 'outside' usually on her own where she carries out many roles such as taking phone calls for orders/bookings, serving customers and calculating the bills. Outside is referred as the place where the the serving takes place. When times are busy, relatives come and help out at Linh's Restaurant.

Linh would like a more computerized system to be more efficient as manually transferring order details to the invoice book can be time consuming. In addition, when it is busy, using time more efficiently is definitely going to give a better service to the customers and would make it less stressful for Linh. Linh has basic knowledge on how to use a computer such as surfing the internet, checking emails and streaming videos.

1.1.2 Define the current system

Customers come in and get seated according to the number of people. Menus are given and the customers are asked what they would like to drink and what dishes they would like if they are ready. The dishes and drinks are recorded on seperate papers. The top copy of the ordering pad, where the dish order is recorded, is given to the chefs where they cook the dishes and one is kept for outside. Once the dishes are served, the customers are checked upon to see if there are any problems occasionally, and once the customers are satisfied and

finish with their meal, they ask for the bill. The recorded order is then copied on to an invoice where the price is calculated for their meal. One invoice is kept and one is given to the customers. The meal is then paid and the customers leave.

The current system is paper based. A diary is used for bookings in which customers can book a table over the phone. A name, number of people on the table, a date and time are recorded in the diary. Orders are taken upfront which is recorded on an ordering pad where one copy is handed over to the kitchen and one is kept to refer to and to transfer order details onto an invoice form.

1.1.3 Describe the problems

When times are busy there could be confusion between on what has been ordered by what table. Also, having to rewrite the order into the invoice book takes time, this is a problem. Furthermore, any inexperienced workers will have to keep referring to the menu when taking orders or calculating the total bill to check if the dish is on the menu and the prices for each dish. This can also lead to a problem where the total price calculated is wrong. Additionally, recorded orders can go missing but that would only happen if the recorded order drops on the floor and no one realises it, this is something that is very unlikely to happen.

1.1.4 Section appendix

Interview with Linh Tham

What is the current system?

LT : I ask the customers what they would like to eat and drink and record it on an ordering pad, I then take top copy to the kitchen and keep the second copy for myself so I can refer to who ordered what. Once the customers has finished eating and ready to pay, I transfer all the details from the ordering pad on to the invoice form such as the drinks and dishes with the prices of each. I give a copy of the invoice to the customer and keep one for myself.

How are the second copies of the order and invoice created?

LT : Because of how thin the paper is on the ordering pad , writing things down marks down what I write on the second copy. However, the second copies are hard to read because the ink from the pen isn't exactly transferred.

What are the problems with the current way of doing things?

LT : Doing it manually is very time consuming as it takes one person just to rewrite everything on the invoice book. If that one person would be able to finish quicker, that person could help out which would benefit us.

What data or information is recorded in the current system?

LT : Food items, drinks,total price and the date of an order.

What are the benefits of the current system?

LT : As the system is paper based, any power cuts or weather issues, will not affect how we run the restaurant.

What should the new system be able to do?

LT : Having a way to look at what tables have ordered what, like a simulator, this will help the restaurant staff to keep track of tables and will reduce confusion. Storing sit down orders would be helpful also.

Having a way to look at what tables have ordered what, like a simulator, this will help the restaurant staff to keep track of tables and will reduce confusion. Storing sit down orders would be helpful also.

Would you like to store phone call orders?

LT: No, I would only like to store sit down orders.

How long would you like to store the information?

LT: I would like to store the information for 3 months.

1.2 Investigation

1.2.1 The current system

Data sources and destinations

There are two main data sources in the current system, the menu and the customer. The menu contains foods and drinks the customer can choose from, the restaurant staff takes the order and then writes down the drinks and dishes onto seperate ordering pads without prices. The details of the order is copied onto the invoice including prices and the date once the customer is finished. Each dish ordered is recorded on the invoice however, each drink ordered isnt and so the total price of drinks ordered is recorded instead and referred as 'Drinks' on the invoice form. Additionally, the number of people on the table isn't recorded on the invoice.

Source	**Data**	**Example Data**	**Destination**
Menu	Drink and dishes with prices	Orange Juice £0.70 Special fried rice £3.70	Customer
Customer	Drink	Bottled water	Restaurant staff
Customer	Dish	Wonton soup	Restaurant staff
Restaurant staff	Drink ordered by customer, table number	Bottled water Sprite Table No. 3	Ordering pad 1
Restaurant staff	Dish ordered by customer, date of order, number of people, table number	Wonton soup Special fried rice 30/9/14 Covers 2 Table No. 3	Ordering pad 2
Ordering pad 1	Total price of drinks - each drink is not specified on invoice, table number	(£0.60+£0.70) Total £1.30 Table No. 3	Invoice pad
Ordering pad 2	Dishes ordered by customer including price of each dish, table number	Wonton soup £1.80 Special fried rice £3.70 Table No. 3	Invoice pad
Restaurant staff	Total price of order	Total price £6.8	Invoice pad
Invoice pad	Copy of invoice	Wonton soup £1.70 Special fried rice £3.70 Drinks £1.30 Total price £6.8 Date 30/9/14 Table No. 3	Customer

Algorithms

Algorithm 1 Taking an order

```
1: OrderTaken ← false
2:
3: WHILE notOrderTaken
4:     IF Customer ready to order THEN
5:         Order ← USERINPUT
6:         OrderTaken ← true
7:     ELSE
8: Wait
9:     ENDIF
10: ENDWHILE
```

Algorithm 2 Generating invoice

```
1: InvoiceGenerated ← false
2:
3: WHILE notInvoiceGenerated
4:     IF Customer has finished ordering THEN
5: Copy order details from order pad onto invoice pad
6: Get prices of each dish and drink ordered from menu
7: Copy prices onto invoice pad
8: Calculate total price
9: Add date
10:         InvoiceGenerated ← true
11:     ELSE
12: Wait for customer to ask for the bill
13:     ENDIF
14: ENDWHILE
```

Algorithm 3 Payment

```
1: Payment ← false
2:
3: WHILE notPayment
4:     IF Customer ask for bill THEN
5: Give invoice
6:         Payment ← USERINPUT
7:         Payment ← true
8:     ELSE
9: Wait
10:     ENDIF
11: ENDWHILE
```

Data flow diagram

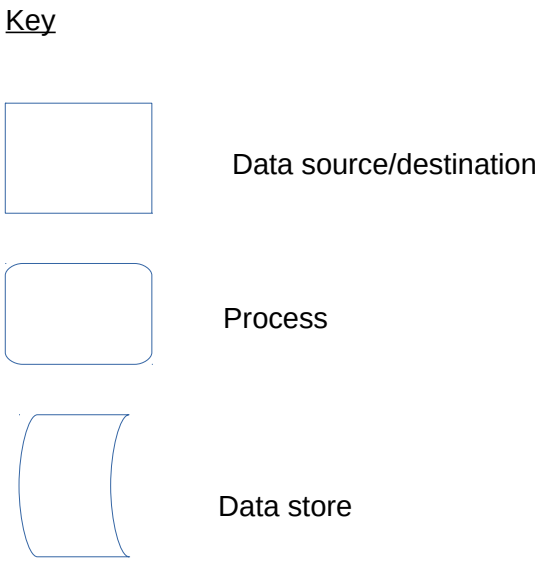


Figure 1.1: Data flow key

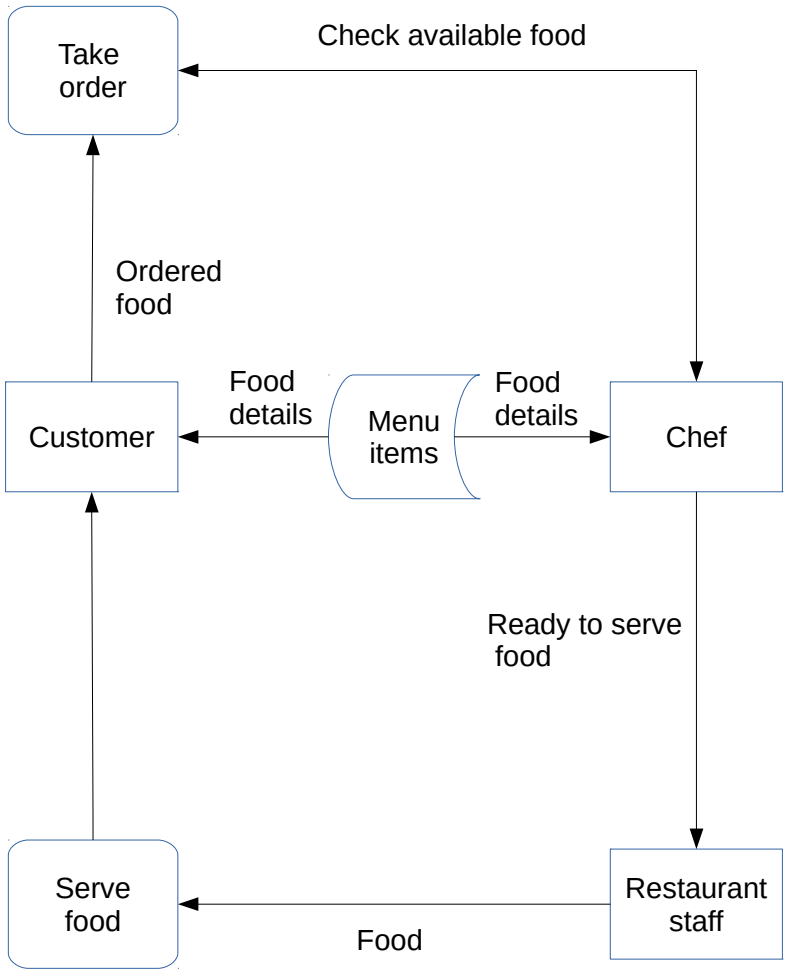


Figure 1.2: Data flow diagram of placing an order

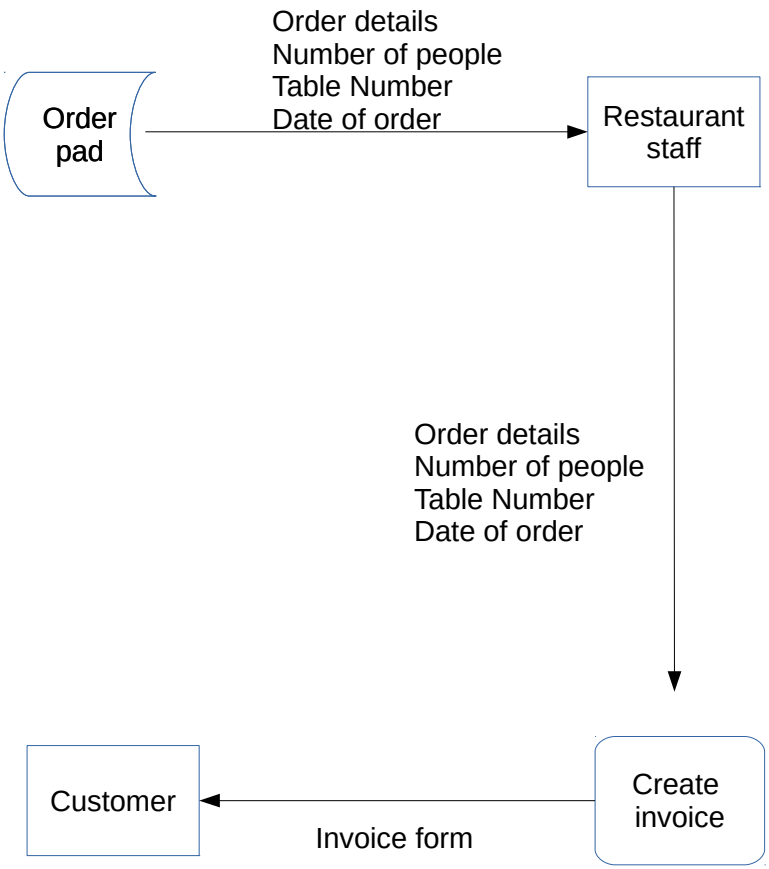


Figure 1.3: Data flow diagram of generating an invoice

Input Forms, Output Forms, Report Formats

Drinks are recorded seperately from dishes as shown below. The number at the top represents what table number this order is from.

Figure 1.4: Writing down drinks ordered on the drink pad

Below is an example of what the ordering pad looks like when a customer's order has been taken. It provides information about the order such as the table number, how many people is seated, dishes ordered and the date the order has taken place. Two copies of this is made, one is taken to the chefs and one is kept for the waitors. This is an input form.

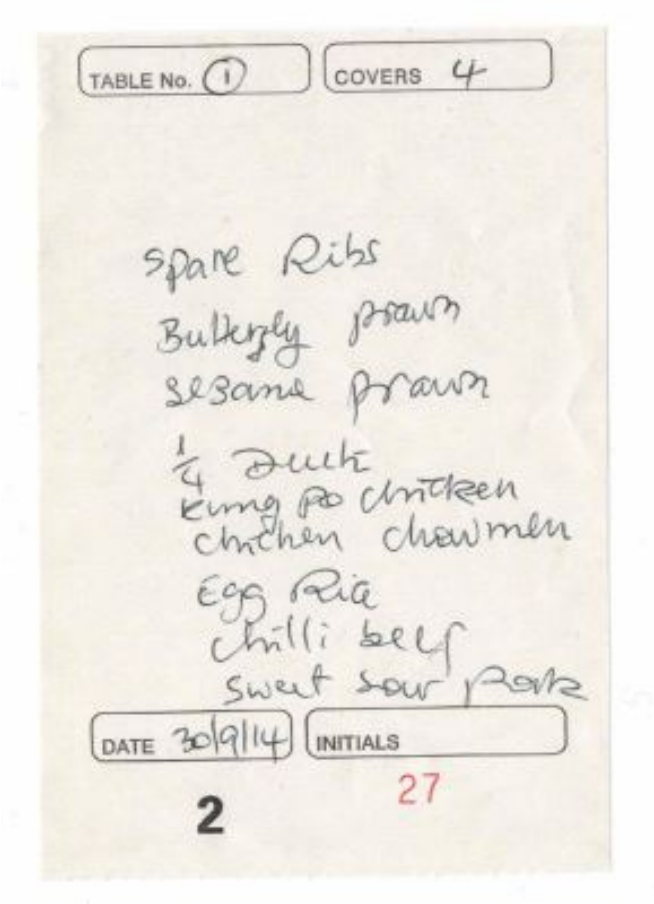
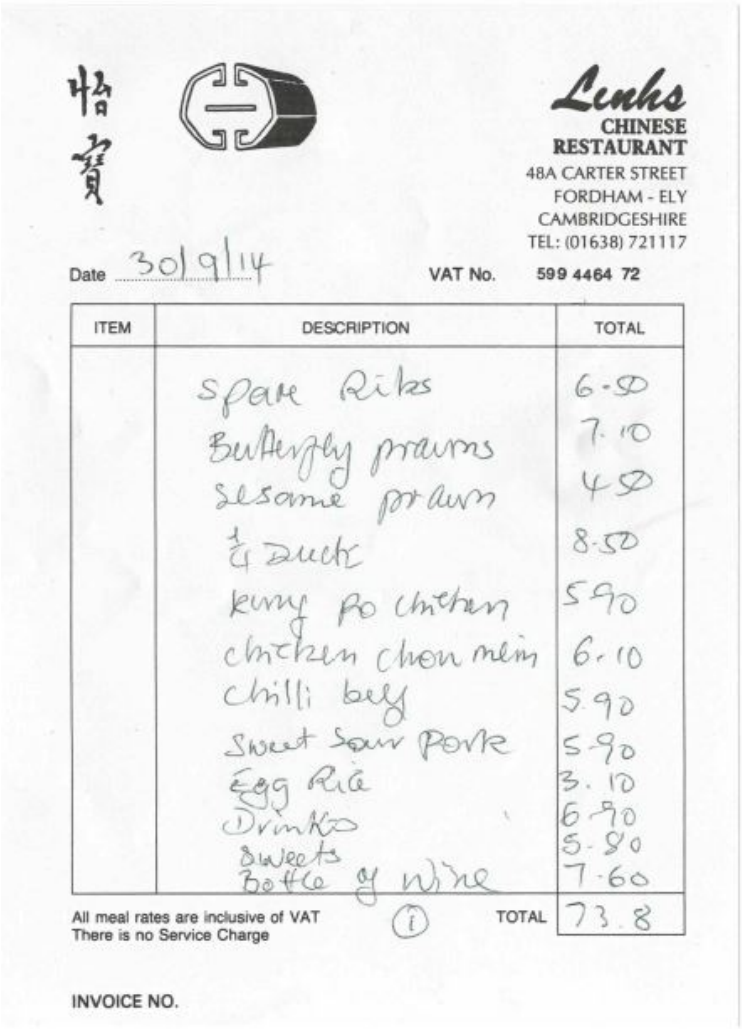


Figure 1.5: Getting an order from a customer

A picture of an invoice is shown below, the information has been transferred from the ordering pad, as shown above, to the invoice pad. An invoice is created once a table has finished eating and ready to pay. Only the date, description, prices and total price is put on the invoice. This is an output which is given to the customers and another copy of the invoice is kept.



Lunhs
CHINESE RESTAURANT
48A CARTER STREET
FORDHAM - ELY
CAMBRIDGESHIRE
TEL: (01638) 721117

Date 30/9/14 VAT No. 599 4464 72

ITEM	DESCRIPTION	TOTAL
	Spare Ribs	6.50
	Butterfly prawns	7.10
	Sesame prawn	4.50
	½ Duck	8.50
	kung po chicken	5.90
	chicken chow mein	6.10
	chilli beef	5.90
	Sweet sour pork	5.90
	Egg Rice	3.10
	Drinks	6.90
	Sweets	5.80
	Bottle of wine	7.60
	TOTAL	73.8

All meal rates are inclusive of VAT
There is no Service Charge

INVOICE NO.

Figure 1.6: Creating invoice

1.2.2 The proposed system

Data sources and destinations

In the proposed system, getting an order from the customer is still the same via using restaurant staff and an order pad. The only change in the propose system is transferring the order details onto an invoice.

Source	**Data**	**Example Data**	**Destination**
Menu	Dish and drink	Spare ribs, orange juice	Customer
Customer	Drink ordered	Orange juice	Restaurant staff
Customer	Dish ordered	Wonton soup	Restaurant staff
Restaurant staff	Drink ordered by customer, table number	Orange juice Table No. 1	Ordering pad
Restaurant staff	Dish ordered by customer, table number, number of people, date of order	Wonton soup, Table No. 1, Covers 1 04/09/14	Ordering pad
Proposed system software	Invoice form	04/09/14 Wontop soup £ 1.80 Drinks £0.7 Total price £2.50	Customer

The new part of the system's data sources and destinations is shown below. Entering the food item onto the software should automatically retrieve its price from the menu database. After a customer has finished with their meal, the simulator saves the Table status (drinks, dishes, table number and date) to the order history database and creates an invoice form.

Source	**Data**	**Data type**	**Destination**
Restaurant staff	Dish	String	Computer - Table status
Restaurant staff	Drink	String	Computer - Table status
Restaurant staff	TableNumber	Integer	Computer - Table status
Restaurant staff	NumberOfPeople	Integer	Computer - Table status
Restaurant staff	DateOfOrder	Date	Computer - Table status
Computer - Table status	OrderID	Integer	Database - Order records
Computer - Table status	Dish	String	Database - Order records
Computer - Table status	Drink	String	Database - Order records
Computer - Table status	TableNumber	Integer	Database - Order records
Computer - Table status	DateOfOrder	Date	Database - Order records
Computer - Table status	TotalDrinkPrice	Float	Database - Order records
Computer - Table status	TotalPrice (TotalDrinkPrice + each dish)	Float	Database - Order records
Computer - Table status	InvoiceForm	string	InvoiceFolder

Data flow diagram

The data flow diagram of placing an order will be the same due to no changes to the way of placing and processing the order.

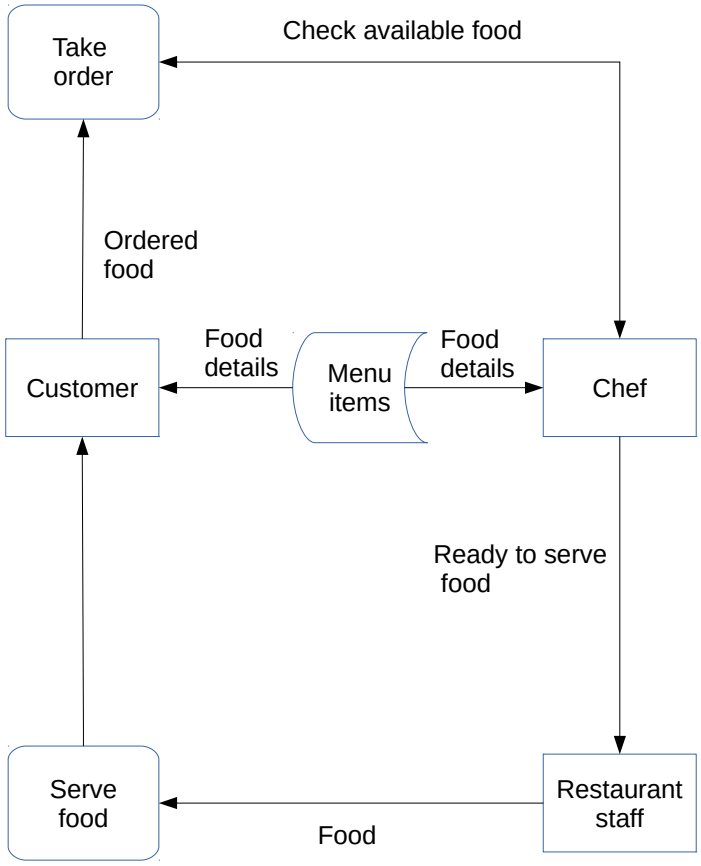


Figure 1.7: A data flow diagram of the proposed system - placing and processing the order

The proposed system will make the restaurant staff input data into the system which will be shown on the application if the user checks what table has ordered what. In addition the inputed data saved in a database once the customer has finished with their meal. Also, invoices will be created though this application.

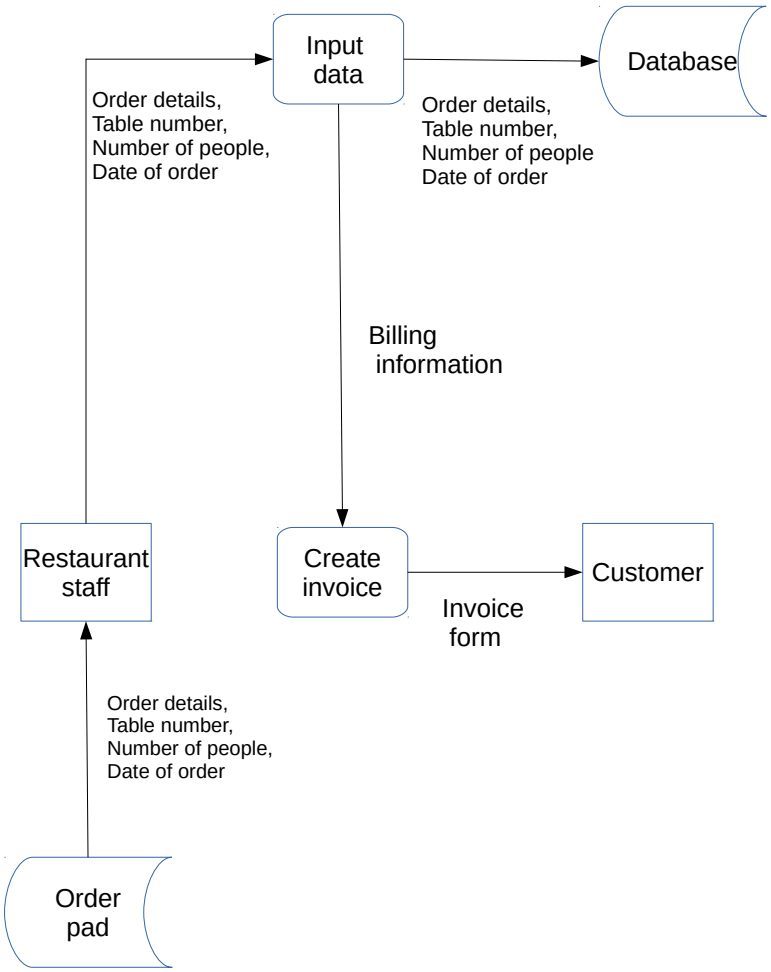


Figure 1.8: Data flow diagram proposed system

Data dictionary

Name	Data Type	Length	Validation	Example Data
TableNumber	Integer	1 - 16	Range	13
NumberOfPeople	Integer	1 - 20	Range	4
MenuItem	String	1 - 20 Characters	Length	Spare ribs
ItemQuantity	Integer	1 - 10	Range	4
ItemPrice	Float	0 - 20	Range	3.2
TotalPrice	Float	0 - 500	Range	54.4
DateOfOrder	Date	4 - 6	Format	16/11/14
InvoiceCreated	Boolean		Presence Check	

Volumetrics

As an ascii character is 1 byte, there will be 35 bytes for one sitdown order. 35*30(approximately the max sit down orders per day) = 1050 bytes is stored per day. Linh's restaurant is open 6 days a week so 1050*6 = 6300 bytes and Linh has stated that she would like to store the information for 3 months so 6300*13.2(weeks) = 83160 bytes will need to be stored.

81360 bytes is equivalent to 79.45 kilobytes (81360/1024). 79.45 kb would be needed to store 3 months of information. The software it self will contain pictures which will increase the size by roughly 2MB. Therefore the total space required would be 6MB if the application itself took 4MB without any images (2MB + 4MB).

1.3 Objectives

1.3.1 General Objectives

- Create a restaurant simulator to track orders
- Simple and clear GUI for user-friendly experience.
- Having the ability to easily modify orders.
- Create a digital invoice after table has finished their meal.
- Storing orders.

1.3.2 Specific Objectives

Simple and clear GUI

- Having a very simple birds eye view image of the restaurant which is made out of shapes to ease the understanding of where each table is.
- Label table with their corresponding number.
- Table shapes will be big so it won't be hard to click on them but not so big that 16 tables can fit on the GUI.
- Clicking on table will bring up a window which shows the status such as the date and food items ordered with noticeable order modification options.

Order alterations

- Have clear Add, Delete and Create invoice buttons.
- When user chooses the add option, have an input box appear where user can type in an ID for a dish/drink or the actual name of the dish/drink.
- Make the input search function not case sensitive.
- When user wants to delete a food item off the list, have clear red X boxes appear next to the name. When red X boxes are clicked on and with confirmation, the item gets deleted.
- Have an up arrow or bottom arrow button just in case a customer orders another food item which is already on the list. The up arrow would increase the quantity of the item by 1 and the down arrow would decrease the item by 1 .
- Clicking on create invoice button will clear the information on the table status and save the digital invoice in a folder.

Track orders

- Drinks and dishes will be seperated by columns.
- Clicking on a table will bring up a small window with the list of food items that the table has ordered, formatted like the invoice form shown on page 15. This also includes the date and table number.

Invoice creation

- Automatically creating a digital invoice when a customer has finished.
- Calculate total price
- The digital invoice will look very similar to the invoice on page 15.
- Invoice will contain the items ordered, prices of each and total price.

- Have the option to print out invoice.

Storing orders

- When using the clear information button, the information is stored in the database.
- Filtering database for user if searching specific information.
- Have an option to view database.

1.3.3 Core Objectives

- Have a working simulator that will have the restaurant layout
- Having clickable tables that will bring up a window showing a digital invoice
- The digital invoice will show the current status such as items ordered, date of order and number of people on the table.
- Application must be able to modify orders
- Application must be able to generate an invoice after table has finished with their meal

1.3.4 Other Objectives

- Print invoice function
- Store order data in a database
- Database search functions such as sort and filtering.

1.4 ER Diagrams and Descriptions

1.4.1 ER Diagram

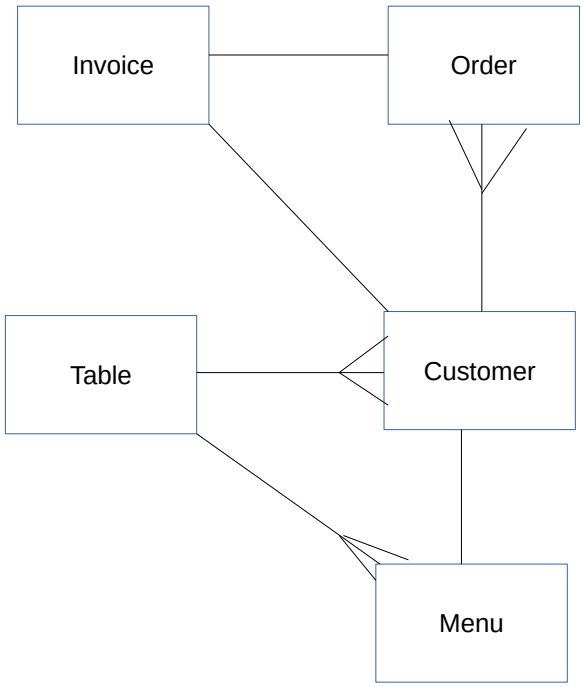


Figure 1.9: E-R Diagram

1.4.2 Entity Descriptions

Customer(CustomerID, *TableID*, *OrderID*, NumberOfPeople, Invoice, Date)

Order(OrderID, *CustomerID*, *TableID*, *MenuID*, DishOrdered, DrinkOrdered, Quantity)

Table(TableID, *OrderID*, *CustomerID*, TableNumber)

Menu(MenuID, Dishes, Drinks, DishPrice, DrinkPrice)

Invoice(InvoiceID, *CustomerID*, *OrderID*, TotalDrinkPrice, TotalPrice)

1.5 Object Analysis

1.5.1 Object Listing

- Customer
- RestaurantStaff
- Dish
- Drink
- Invoice
- Menu

1.5.2 Relationship diagrams

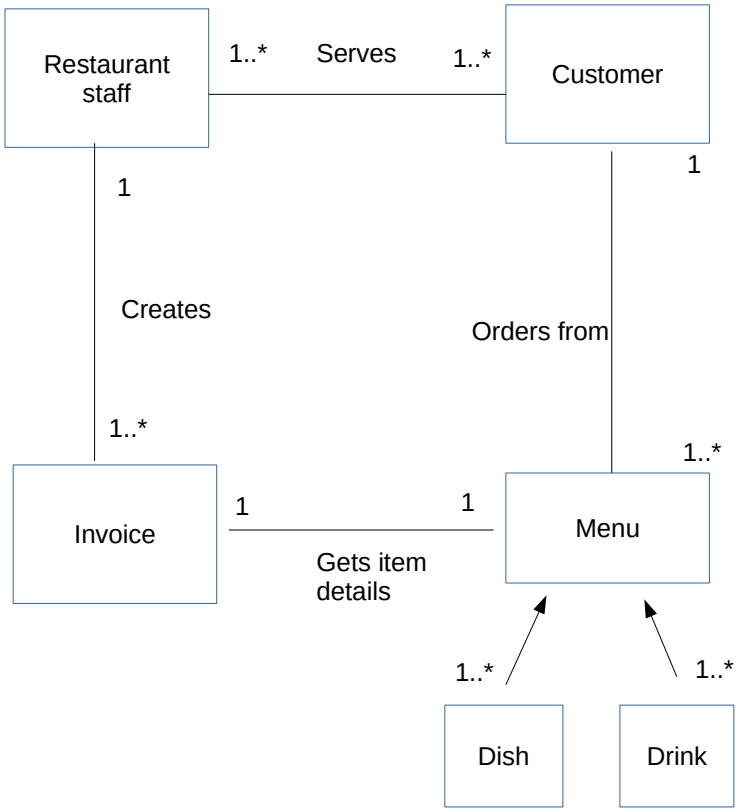


Figure 1.10: Relationship diagram

1.5.3 Class definitions

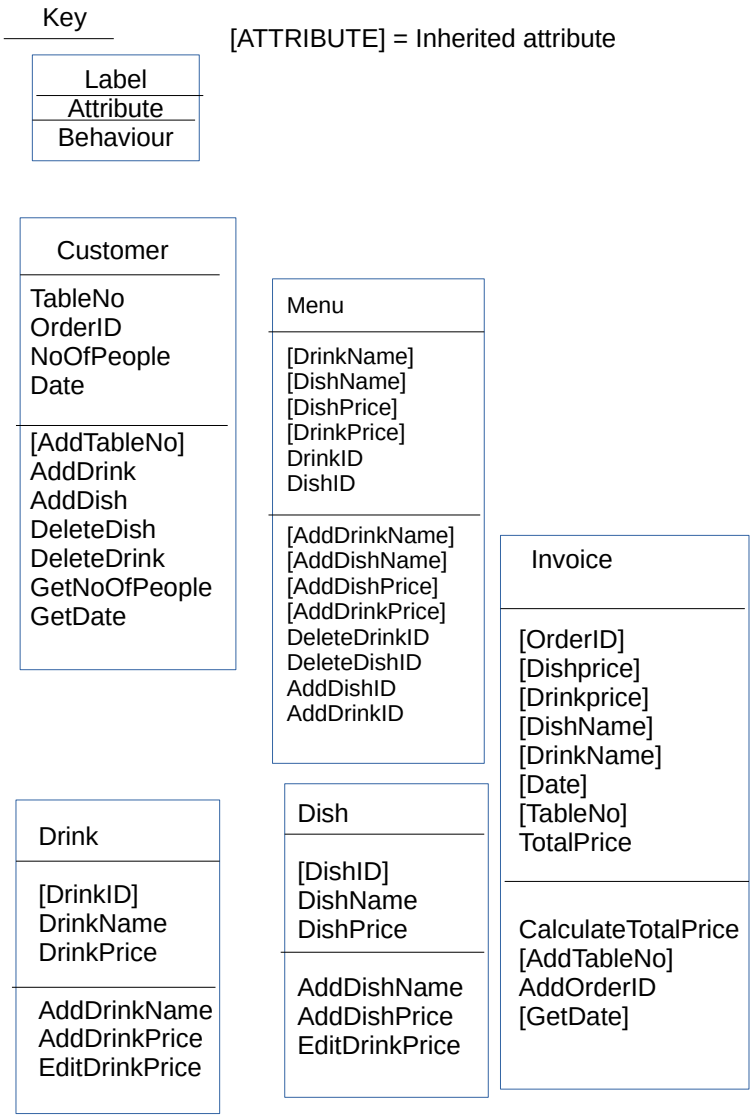


Figure 1.11: Class diagram

1.6 Other Abstractions and Graphs

1.7 Constraints

1.7.1 Hardware

The current computer specifications is as follows:

- 19" Display
- AMD FX(fm) - 6300 six-core CPU 3.50Hz
- 8GB RAM
- NViDiA GeForce 9600 GT 1GB
- Windows 8.1 64 bit

There shouldn't be any constraints apart from the fact that the new system will have to be designed to fit the 19" screen. Also the position of where the computer will be placed in the restaurant is a limitation.

1.7.2 Software

The current computer uses Windows 8.1 and Linh would prefer it to stay that way as she is familiar with the operating system. This is not a problem as the proposed system will run fine on Windows 8.1. Apart from that, Linh has not stated what software can or cannot be used.

1.7.3 Time

Linh has not set me a deadline for the new system and is in no rush for it to get done. Therefore the deadline will be Friday 27th March 2015 which is the coursework deadline set by my teacher.

1.7.4 User Knowledge

Linh has basic knowledge on how to use a computer such as being able to check emails and simple web surfing. Basic knowledge will not constrain the project as one of the objectives is for the software to be simple and clear.

1.7.5 Access restrictions

All working staff should be able to use this software due to the nature on how the business is run. All waiting staff should be able to carry out the same roles such taking an order, serving and creating an invoice form. However, customers should not be able to access this application at all which could be considered as a constrait. A simple enter password-to-access mechanic could be used as a solution to this.

1.8 Limitations

1.8.1 Areas which will not be included in computerisation

The method of taking orders will not be computerised as it more convenient to just take orders by pad. Using an ordering pad is useful as it is small, light and easy to carry around. Also, the payment system (receiving money and giving back change) will not be computerised as there are no problems with the current payment system. More problems will likely be created if it was to be computerised such as giving back the correct amount of change and registering the amount of money received.

1.8.2 Areas considered for future computerisation

Tracking bookings for tables can be a feature for later as it could be helpful if the book of table bookings goes missing or if theres no more space to write down bookings. In addition, Linh has not stated that she wanted take aways to be computerised. This could be an additional feature in the future to the program where it creates invoices for take aways.

1.9 Solutions

1.9.1 Alternative solutions

Solution	Advantages	Disadvantages
Python Desktop Application with a GUI	The design can be changed according to client needs. Not complicated to use. Very low cost. User-friendly and problems with current system will be fixed. Extra features can be implemented.	Application will take up noticeable computer storage. Will take a long time to create GUI application. If theres a power cut then system will not be useable
Touch sceen self-order system	Customer has more freedom. Less work for restaurant staff. Problems with current system will be gone.	More hardware and software needed - can be very costly. Technical issues will be hard to fix.
Getting someone to do invoices only	Will solve the main problem with current system. No need for a computer.	Will be hard to find someone who will only do invoices. If business isn't busy then invoice person will be almost useless. Could be more costly in long run.
Redesign current manual system	No cost or very low cost as no computer/software will be needed. Current manual system is simple.	May not be able to fix problems. Will take some time to figure out how to fix problem.

1.9.2 Justification of chosen solution

I have chosen Python Desktop Application with a GUI as the solution because of many reasons. One reason is that the touch screen solution will be very costly and customers would have to queue up to use the machine if it gets busy. This will affect how the business will run as many customers do not like to wait. Also, hiring out someone to do invoices will not be efficient as money will be wasted if business is not busy. Furthermore redesigning the current system will take time as Linh would need to figure how fix the problems in the system and also this will most likely not fix most problems with the current system. Therefore I choice Python Desktop Application because it would not need any further hardware, this will not negatively affect customers experience at the restaurant in any way and due to Python being very flexible, the program can always be changed to Linh's wants.

Chapter 2

Design

2.1 Overall System Design

2.1.1 Short description of the main parts of the system

- Restaurant Simulator
  - Core Elements of System
  - General User Interface
  - Adding Item
  - Deleting Item
  - Saving Order Information
  - Managing Bookings
  - Managing Item Menu

Core Elements of System
The system will be designed to make it easier to track information about the restaurant for the restaurant staff, information will be displayed on the application. Information tracked down includes order information such as what has been ordered by each table and the information about that table like the number of people, date and time arrived. In addition, booking times will be displayed at the main screen. As well as displaying key information, the system will have features to add/delete/edit information. For example, adding items to an order, deleting irrelevent bookings and editing booking times. The core elements of the system will be based on managing orders and bookings.

General User Interface

- Only staff will be able to access this application, so a box will be the first thing that prompts up when the applicaiton is opened. This box will require staff members to enter a password which they have created.
- After entering the correct password, the application will display the layout of restaurant in a birds eye view way. The layout will contain shapes which represent each table, each shape will have the number of table on it.
- Clicking on table will bring up a box with a layout like an invoice such as the one on page 15.This screen will contain the table's status such as what they ordered, date,time, table number, number of people and total price. The main box in the middle will be split in half where the left half will contain the dishes ordered and the right will contain the drinks ordered. At the bottom will be contain the editing features where there will be an Add, Delete and Finish buttons. In addition, there will be a back arrow at the top and once this is clicked, it will return to the main interface with the restaurant layout and save the order information.

Adding Item

- The managing order box that pops up when clicked on a table, will have an 'Add' button at the bottom. This button will have the feature to add a menu item to the order.
- When the 'Add' button is clicked, a box will pop up where the user enters the name or item ID and if name or ID is entered correctly, the item will appear on the table status. The menu will be displayed to aid the user.

Deleting Item

- The managing order box will have a 'Delete' button located at the bottom. This button will have the feature to delete a menu item off the order
- When the 'Delete' button is clicked, red boxes with an X will appear next to each item ordered. If the red button is clicked, the item will disappear off the order.

Saving Order Information

- A 'Finish' button will be located along with the 'Add' and 'Delete' buttons.
- The 'Finish' button will be used once a table has finished eating/ordering. It will save all of the current information about this particular order.
- Information will be the ordered menu items, table number, date, time, number of people and the total price.

Managing Bookings

- Any table bookings will be displayed on the main screen
- A button labelled "Bookings" will be at the main screen. A box will appear that will be used to manage bookings.

- Adding and deleting bookings will be available through this box that is used to manage bookings.

Managing Item Menu

- There will be an option to add an item to the menu or delete an item off the menu. This will be accessed at the menu bar.
- The menu bar will have a drop down box containing "Add Item" "Delete Item"
- Adding an item requres the user to input the information required.
- Deleting an item will be done by the user entering either the name of the item or the ID. The menu will be displayed to aid the user.

2.1.2 System flowcharts showing an overview of the complete system

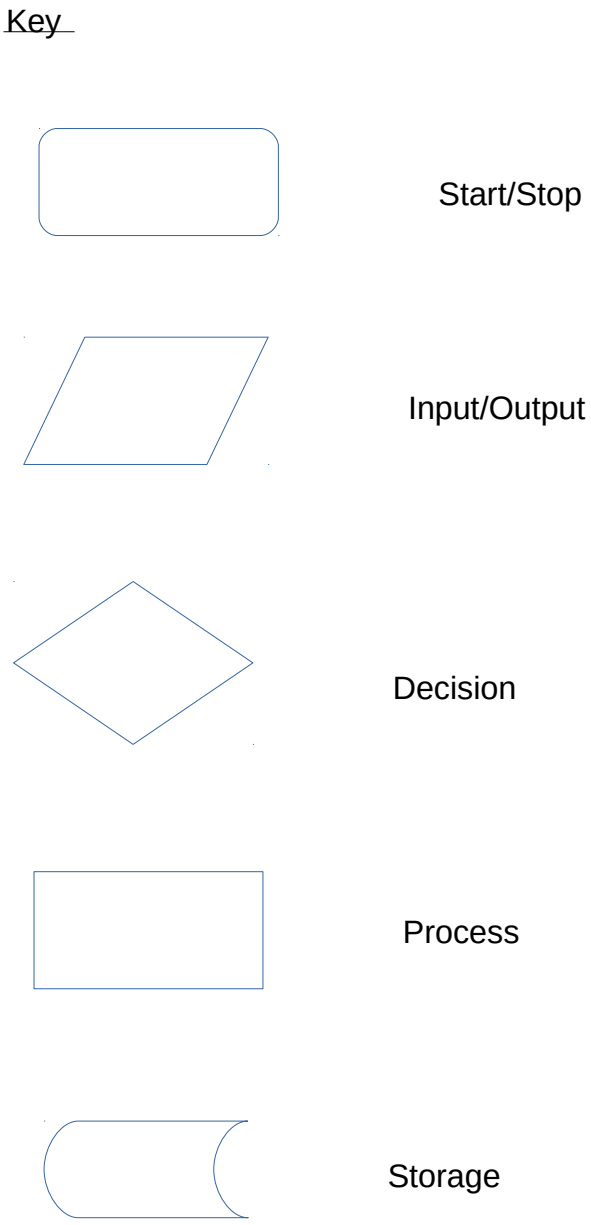


Figure 2.1: Key for flowchart

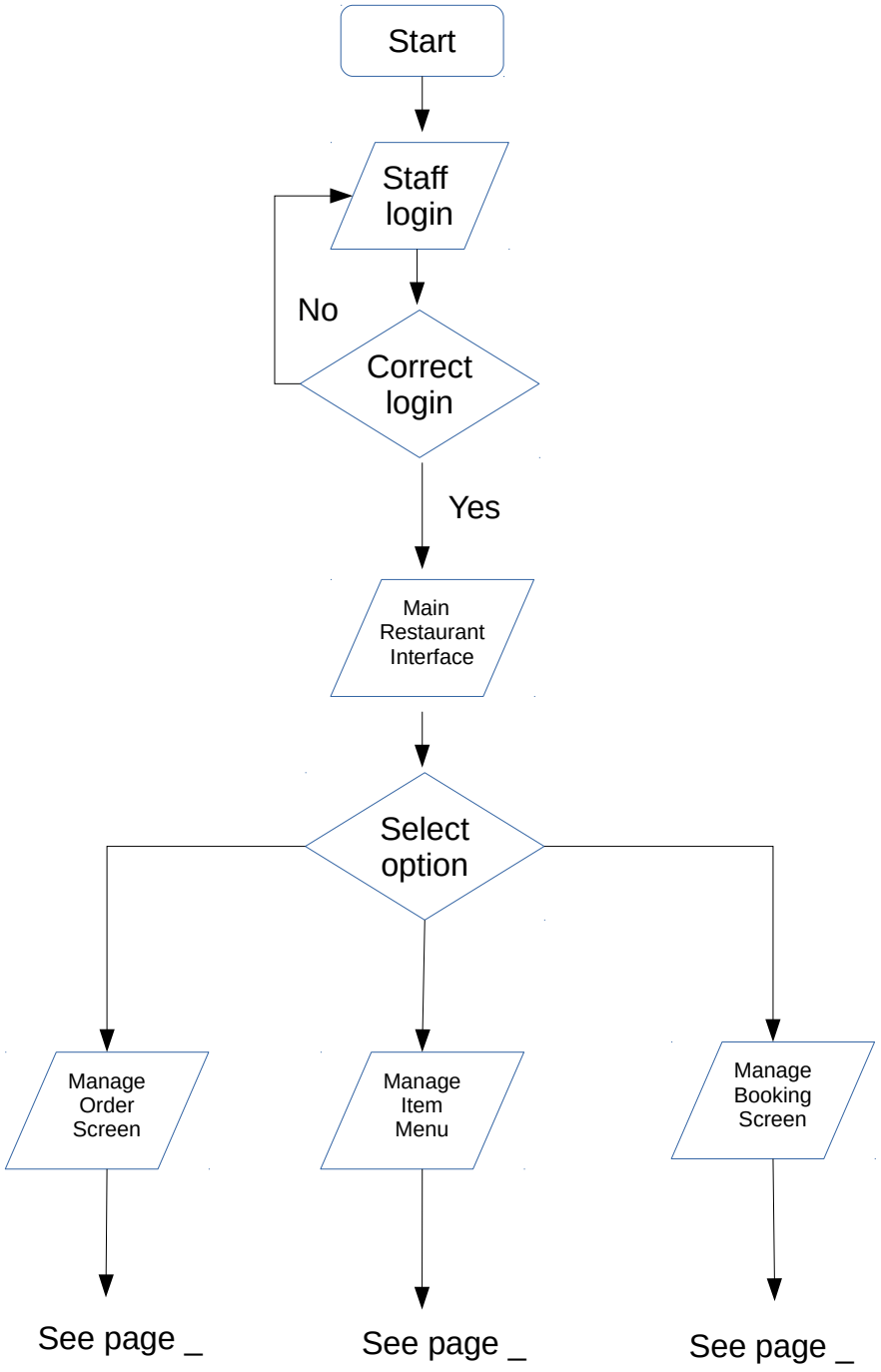


Figure 2.2: Flow chart of system

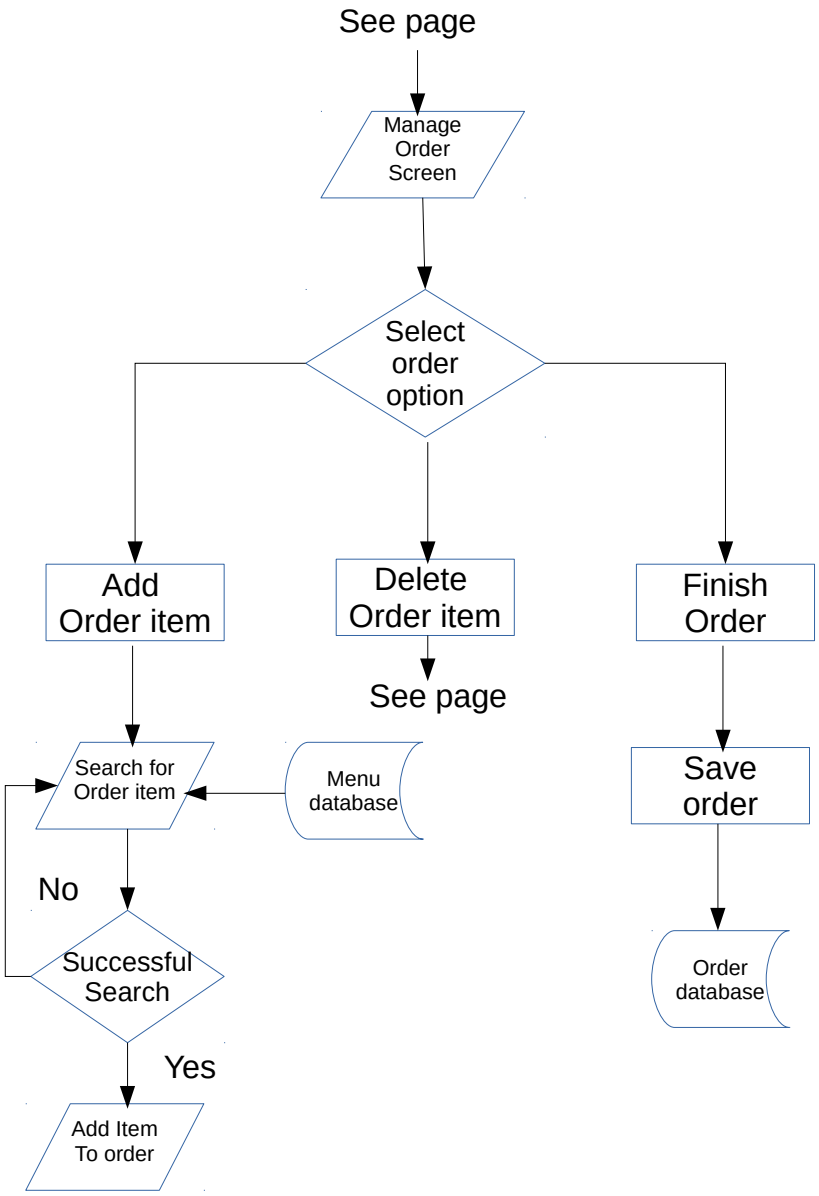


Figure 2.3: Flow chart of order

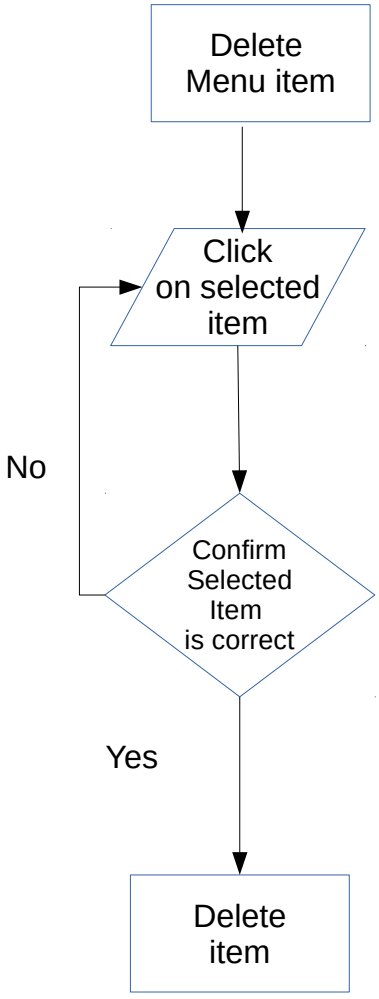


Figure 2.4: Flow chart of deleting an item of an order

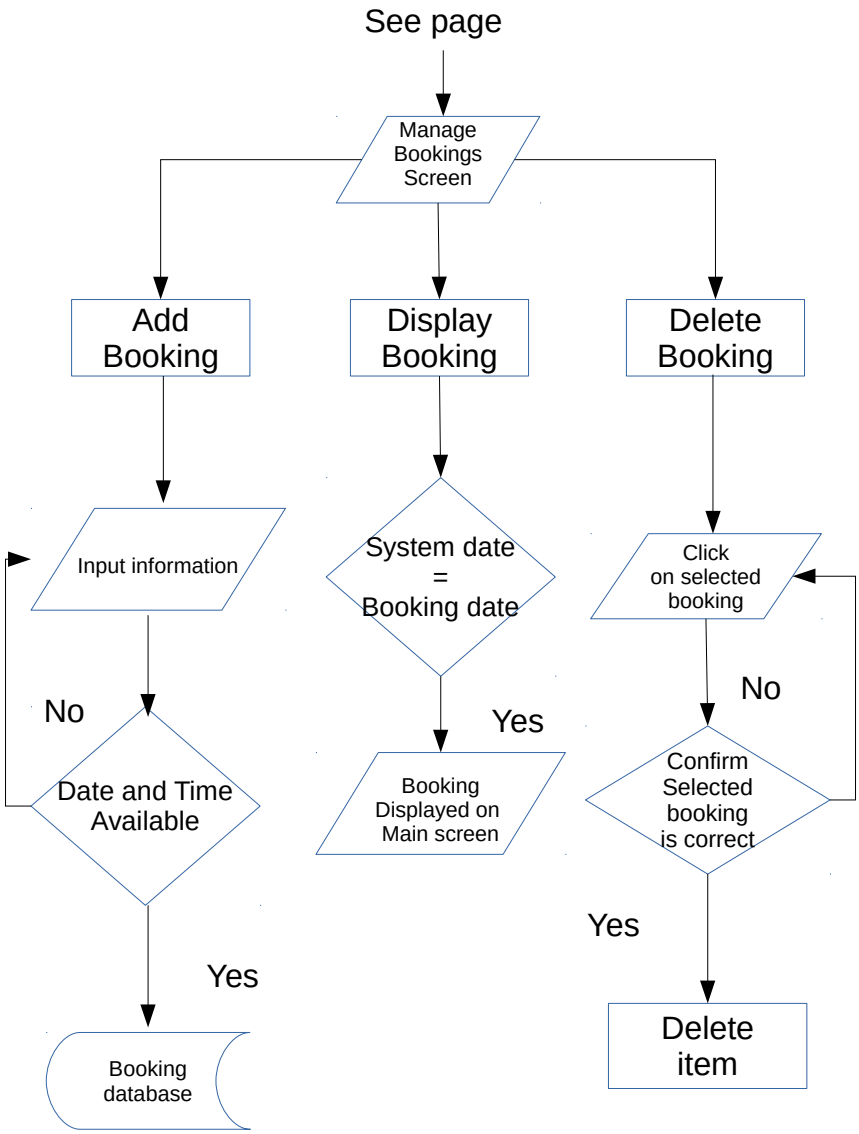


Figure 2.5: Flow chart of bookings

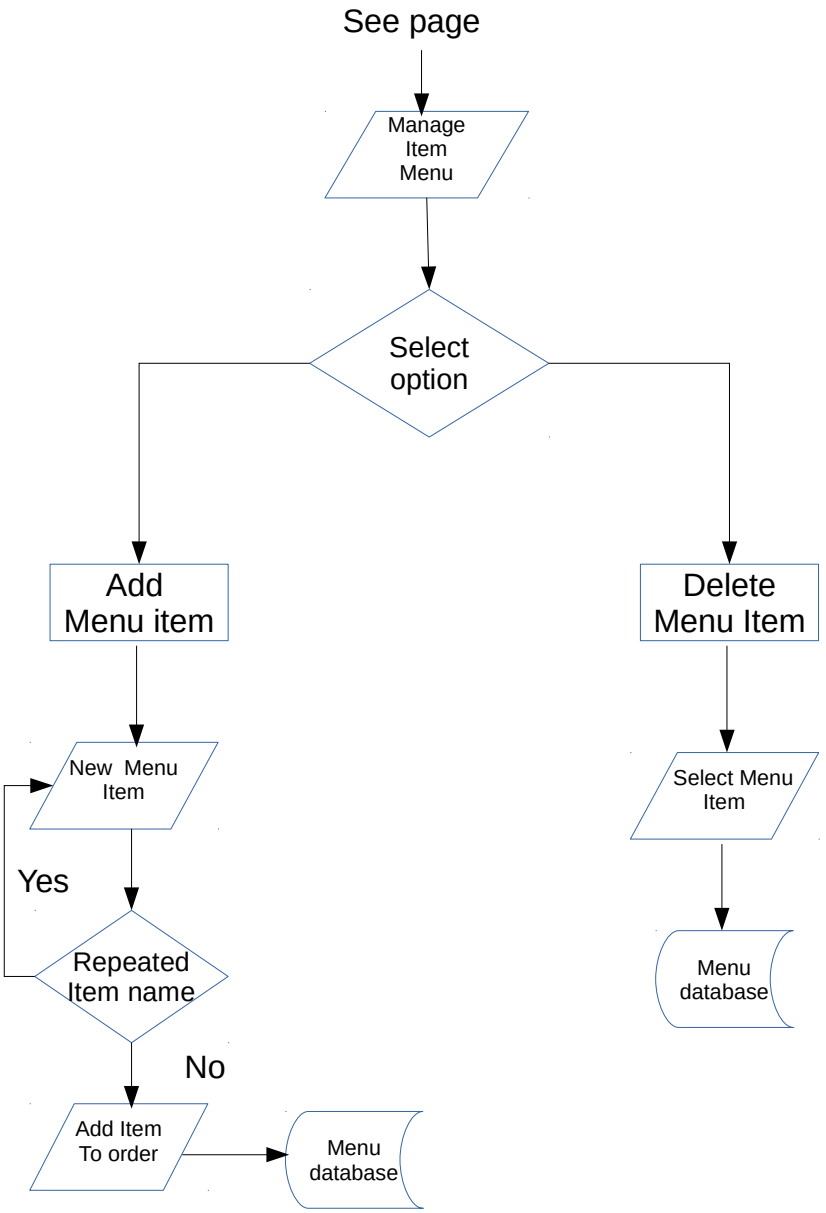


Figure 2.6: Flow chart of adding an item to the menu

2.2 User Interface Designs

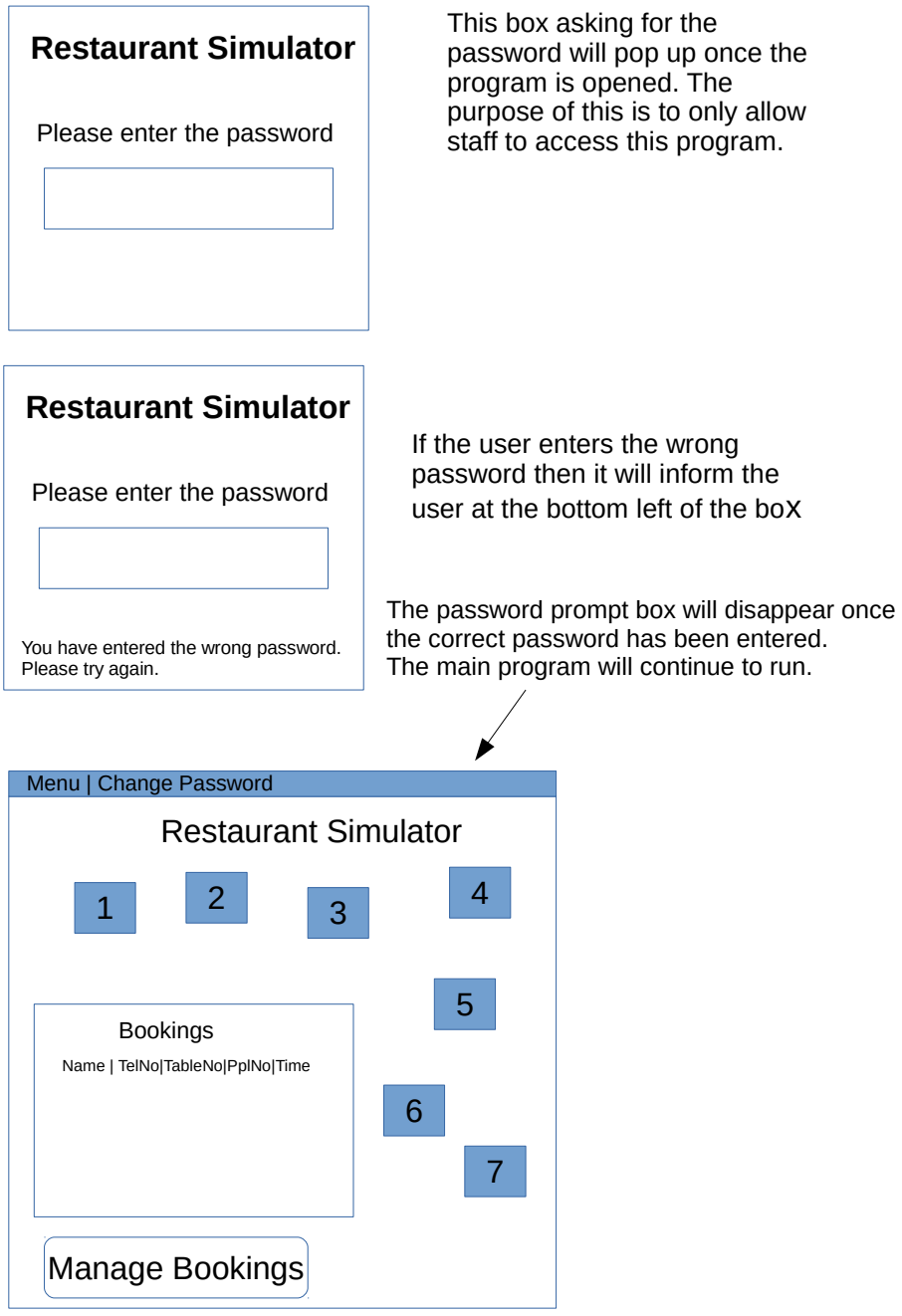
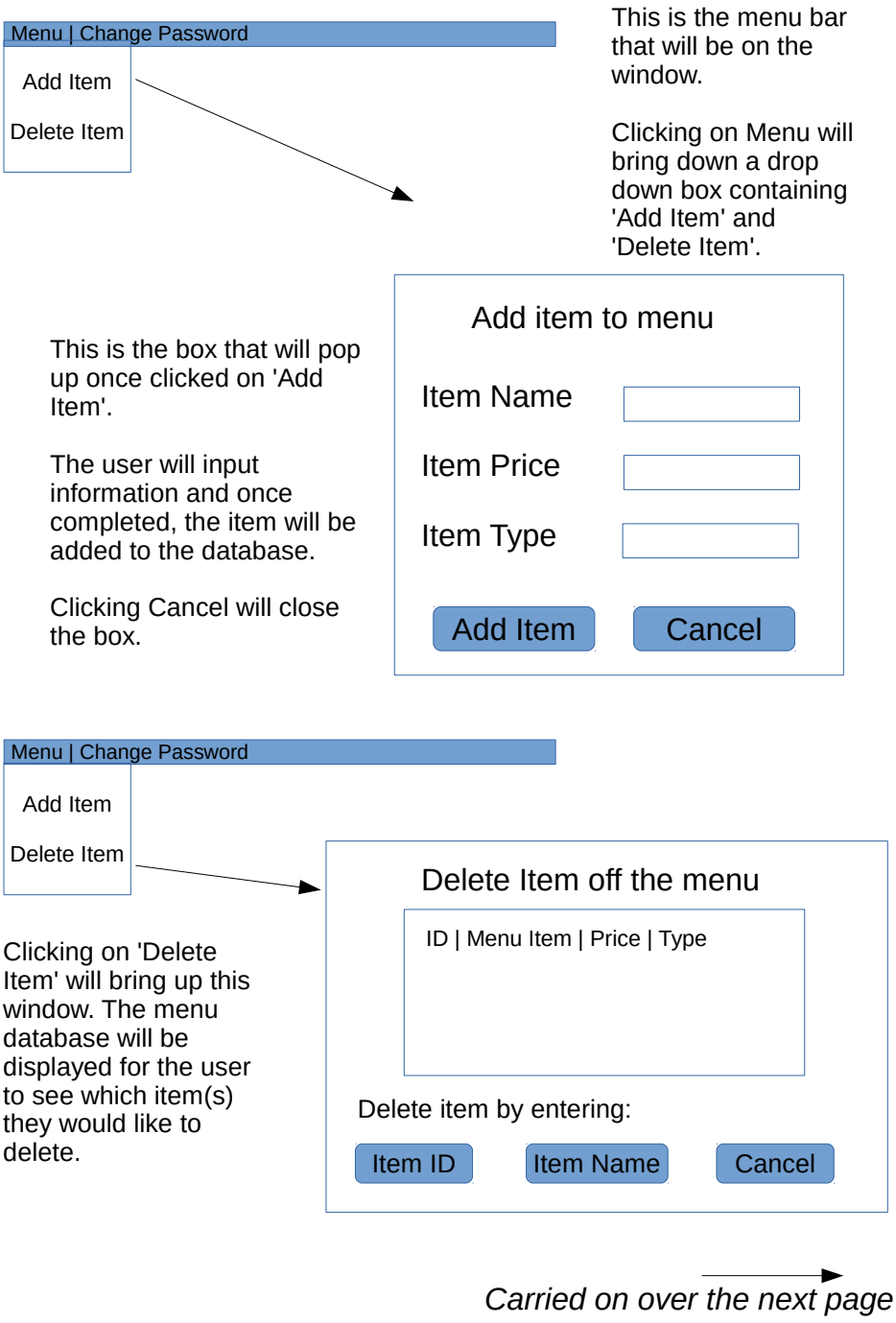


Figure 2.7: Password Prompt



Carried on over the next page

Figure 2.8: Explaining Menu Bar

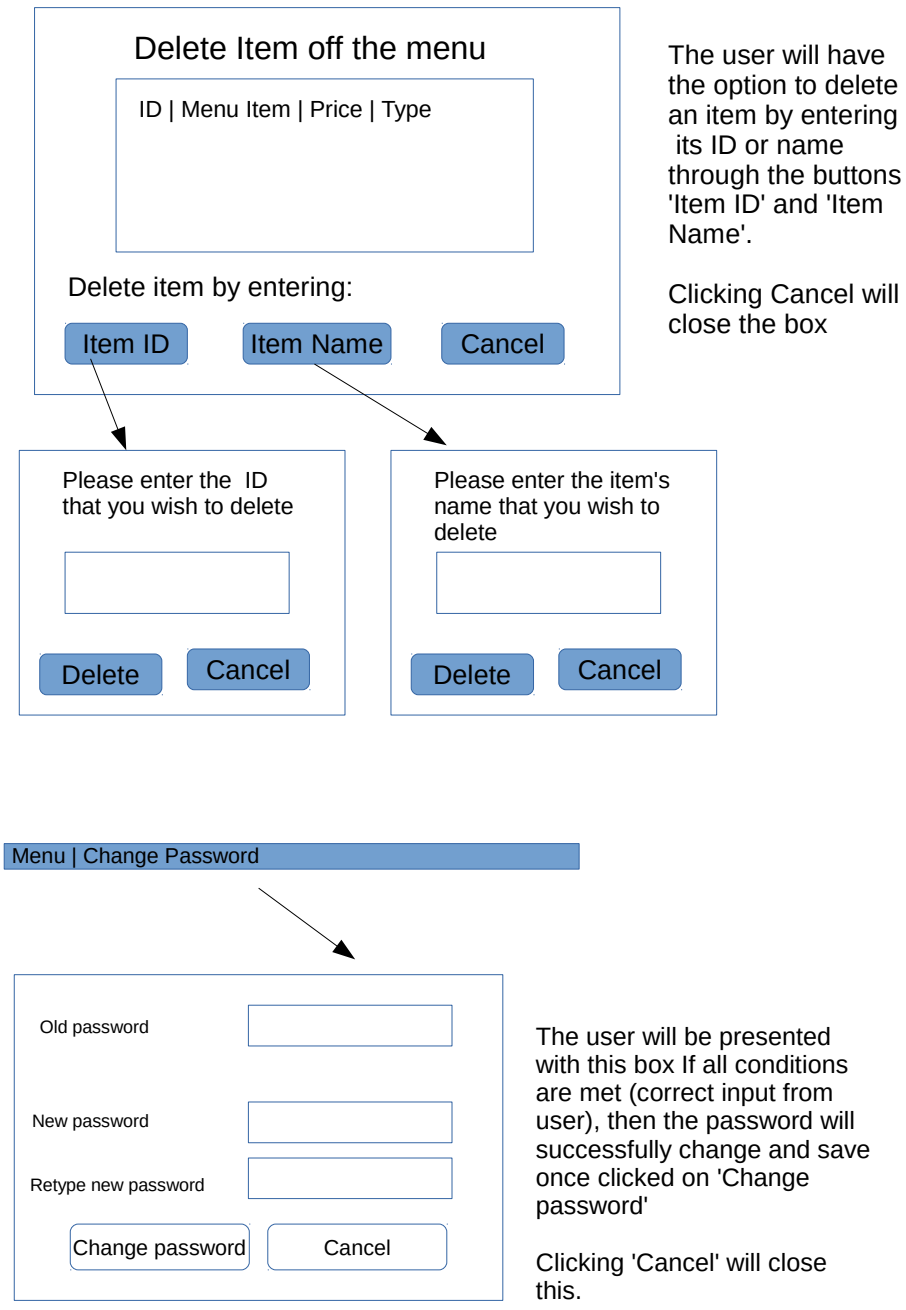
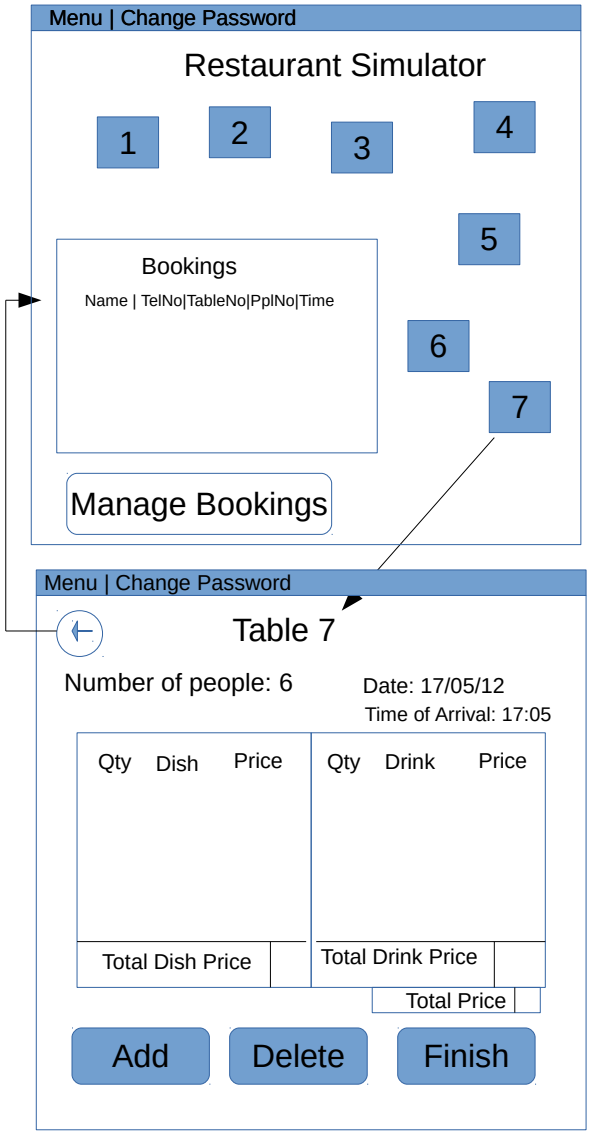


Figure 2.9: Explaining Menu Bar



This is the main screen. Squares with numbers will represent the tables in the restaurant,the number represents the table number. There will be 16 tables and the squares will be large so it will be easy to click on.

Bookings will be shown inside the left box and the button 'Manage Bookings' will be used to add/delete bookings.

An order can be checked by clicking on the respective table.

This is the order screen that is displayed once clicked on a table. The order information such as the items ordered, number of people and prices will be displayed according to the table number.

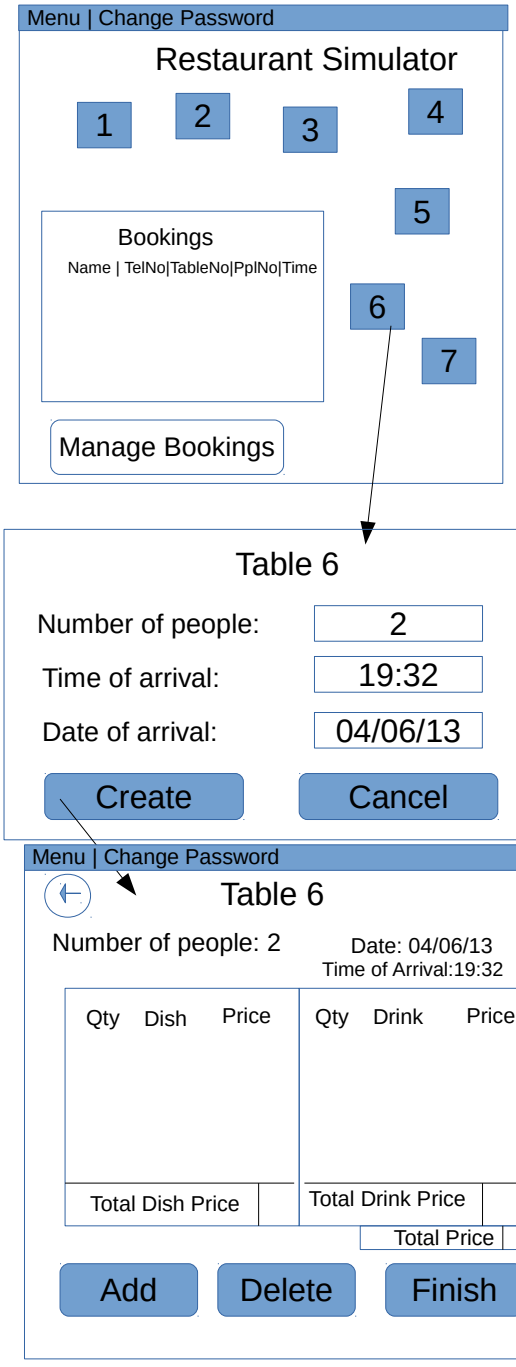
The 'Add' button will be used to add menu items to the order, the item added will be displayed in the appropriate box.

The 'Delete' button will be used to delete any items of the order.

Selecting 'Finish' will result in the information to be saved into the database and clear any information on the order screen apart from the table number.

The back arrow at the top left will save the current information about the order and return the user to the main screen

Figure 2.10: Main Screen



However, if a table hasn't been occupied then the 'Assign customers to table' box will appear once clicked on the unoccupied table.

User inputs the number of people that will be on the table

The user will not input the time of arrival or date of arrival as these will be from the system time and date.

Selecting 'Create' will transfer all the details from the 'Assign customer' box such as number of people, date and time of arrival to the order screen.

Selecting 'Cancel' will close the 'Assign customer' box.

Figure 2.11: Unoccupied table

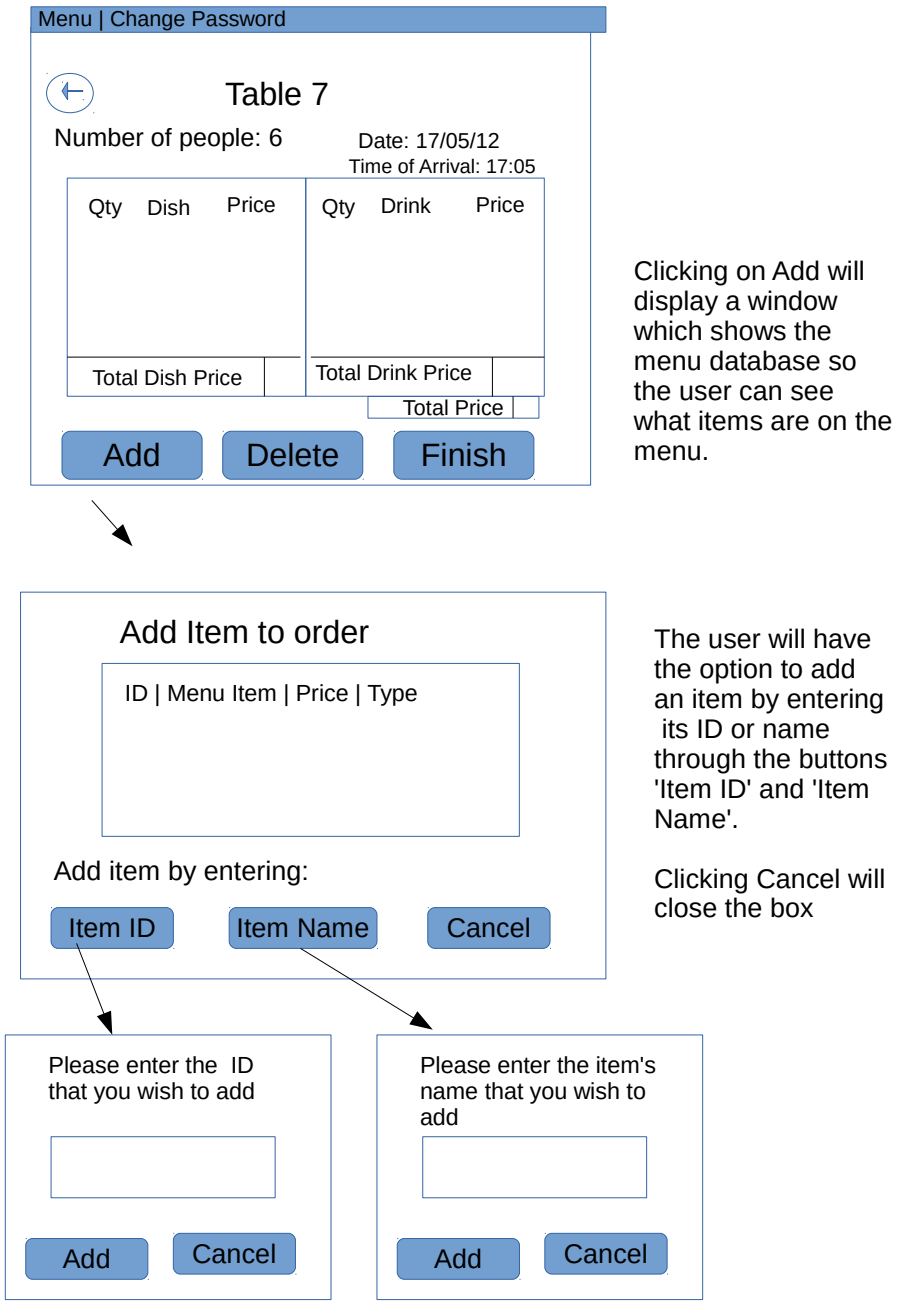
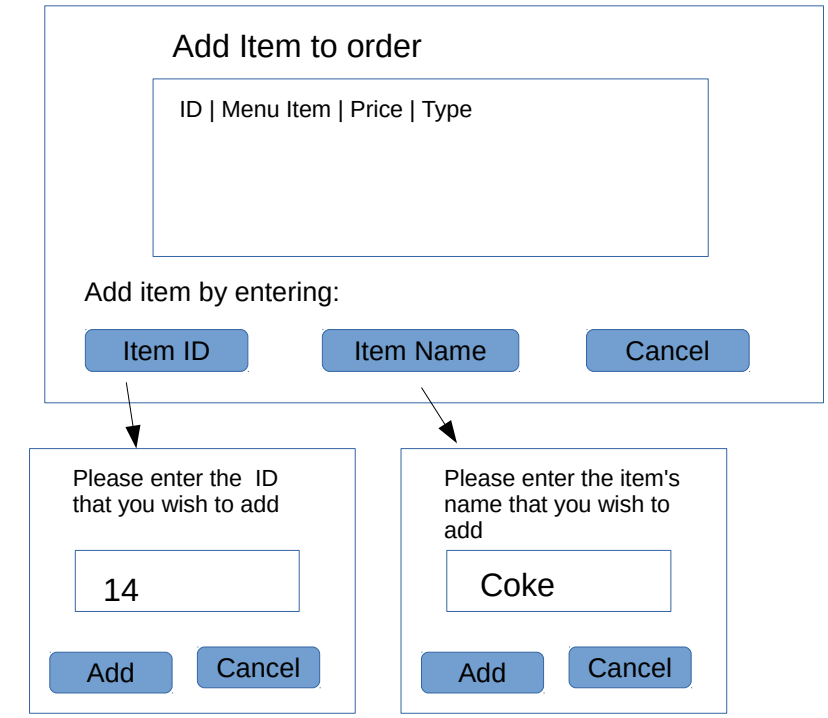
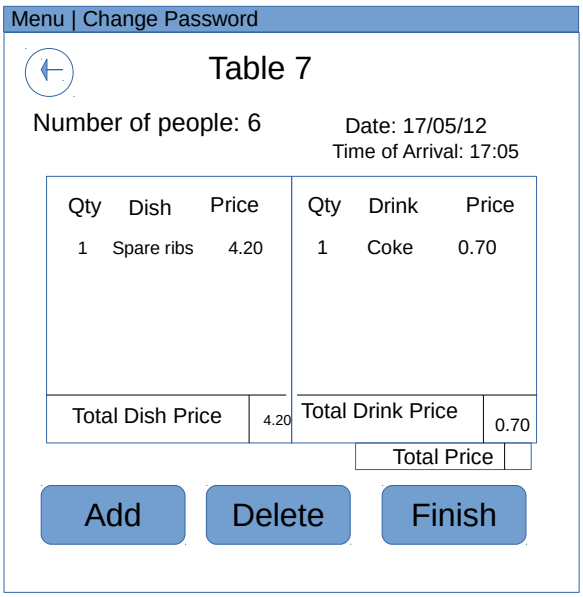


Figure 2.12: Add Item



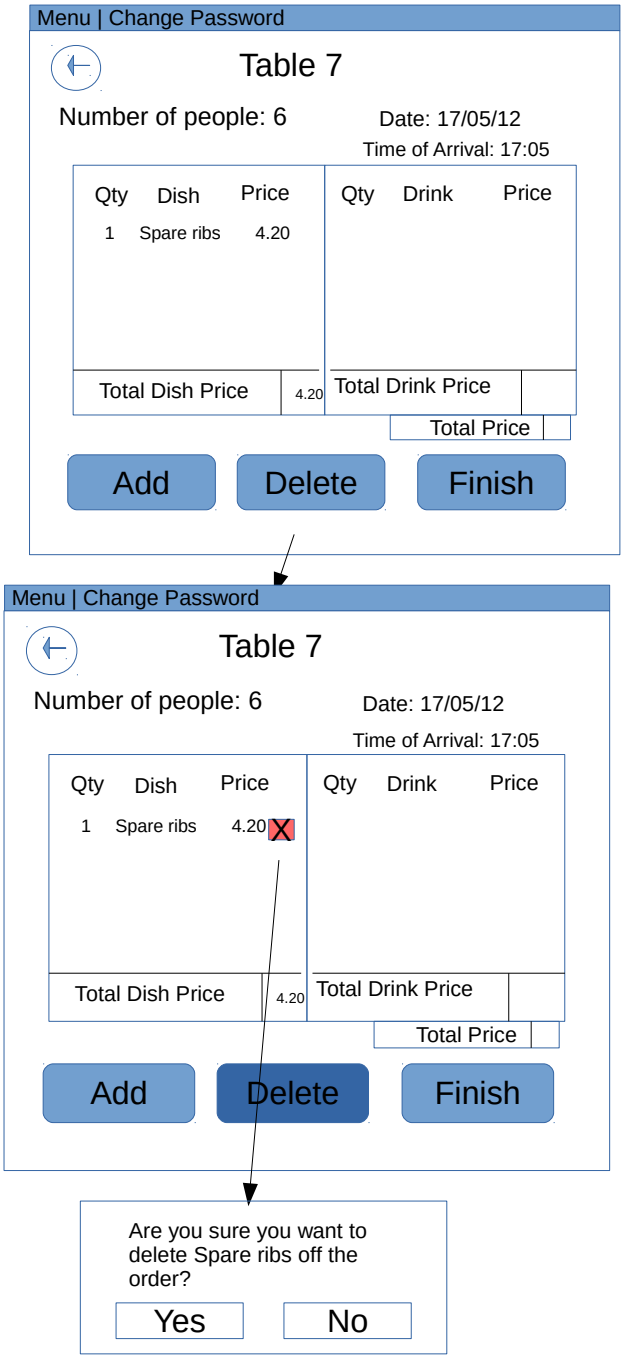


ID 14 has been added to the order which is spare ribs.

Using the Item name 'Coke', the item has been added to the order.

Adding items to the order obtains the prices of these items and displays it. It also calculates the totals for dishes and drinks thus far.

Figure 2.13: Add Item



Selecting 'Delete' will change the colour of the button to make the user aware that it is in the 'Delete' mode.

In 'Delete' mode, red boxes with an 'x' will appear next to each order item. Clicking on it will bring up a confirmation box. If the quantity is more than 1 then it will reduce the quantity by 1.

After selecting 'Yes', on the confirmation box, the item chosen will be removed off the order. Clicking 'Yes' or 'No' on the confirmation box will make the user go out of the Delete mode.

So if the user wanted to delete another item, the user would have to click the 'Delete' button again.

On the other hand, if the user accidently clicked on the 'Delete' button, the user could just click on the 'Delete' button again to get out of the Delete mode.

Figure 2.14: Delete Item

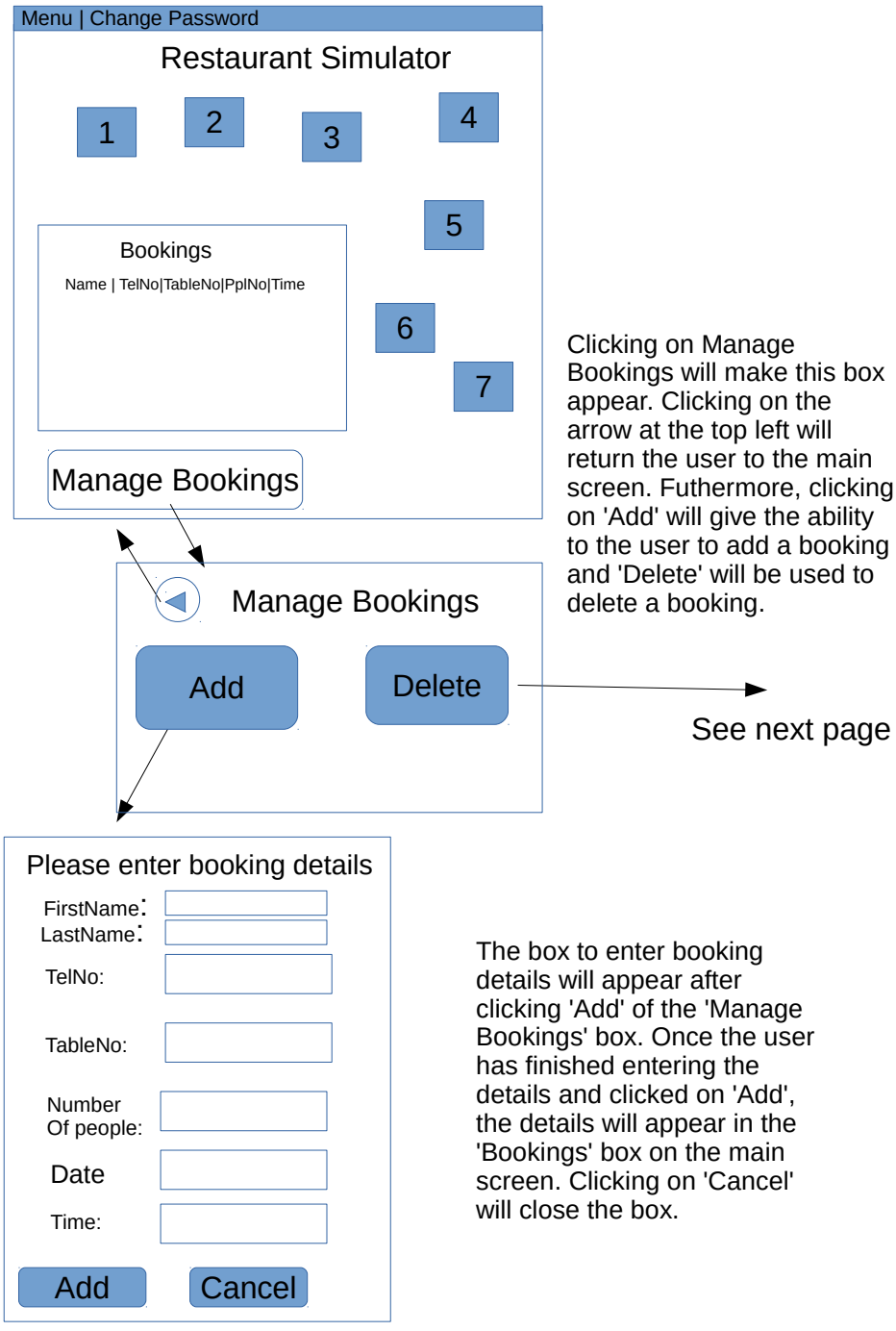
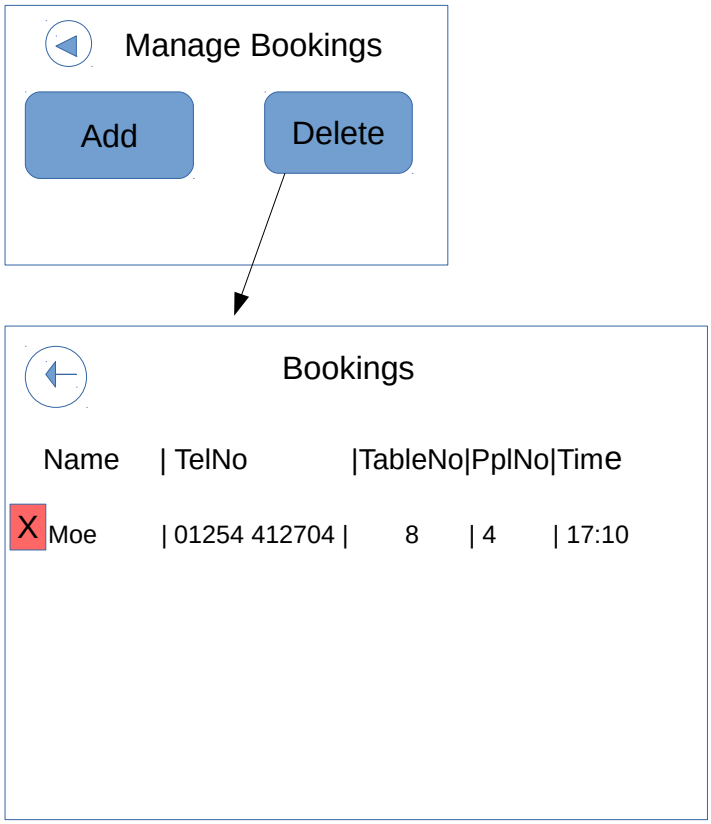


Figure 2.15: Add Booking



Selecting 'Delete' will put the user in delete mode where red boxes will appear. The Bookings box from the main screen will appear but in a larger view. Clicking the arrow at the top left will return the user back to the main screen.

Just like deleting an item from an order, clicking on Delete will make red boxes appear for each booking. Clicking on the red boxes will delete the booking of the list.

Figure 2.16: Delete Booking

2.3 Hardware Specification

Keyboard and mouse are essential as the keyboard will be used to input information and the mouse will be used to navigate. The program would need to fit a 19" screen, this is important because one of my client's main requirements is to be able to track information and so having a large window fitting the screen will make it easier to look at. A processer with 1GHz will be perfectly suitable for this program to run smoothly and since the user has AMD FX(fm) - 6300 six-core CPU 3.50GHz, the program shouldnt run without any problems. In addition, not much RAM would be needed to run this program, 1GB would be more than enough and since the user has 8GB RAM the program shouldn't experience any further hardware based problems.

2.4 Program Structure

2.4.1 Top-down design structure charts

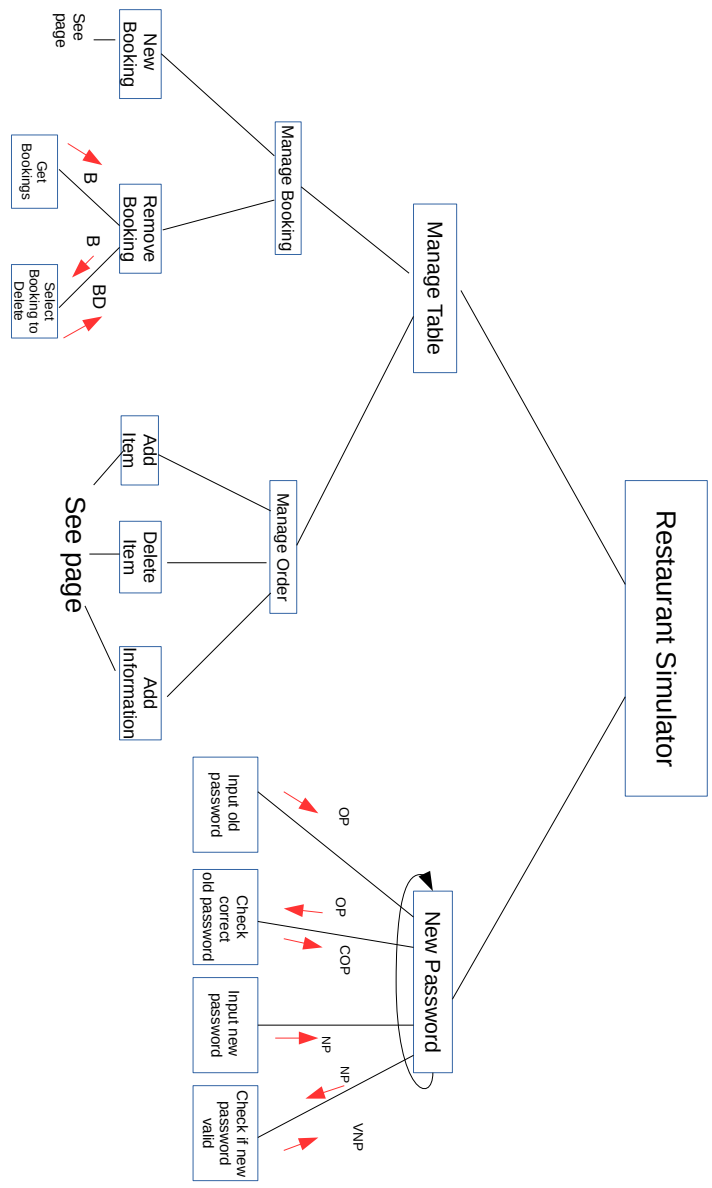


Figure 2.17: Main structure

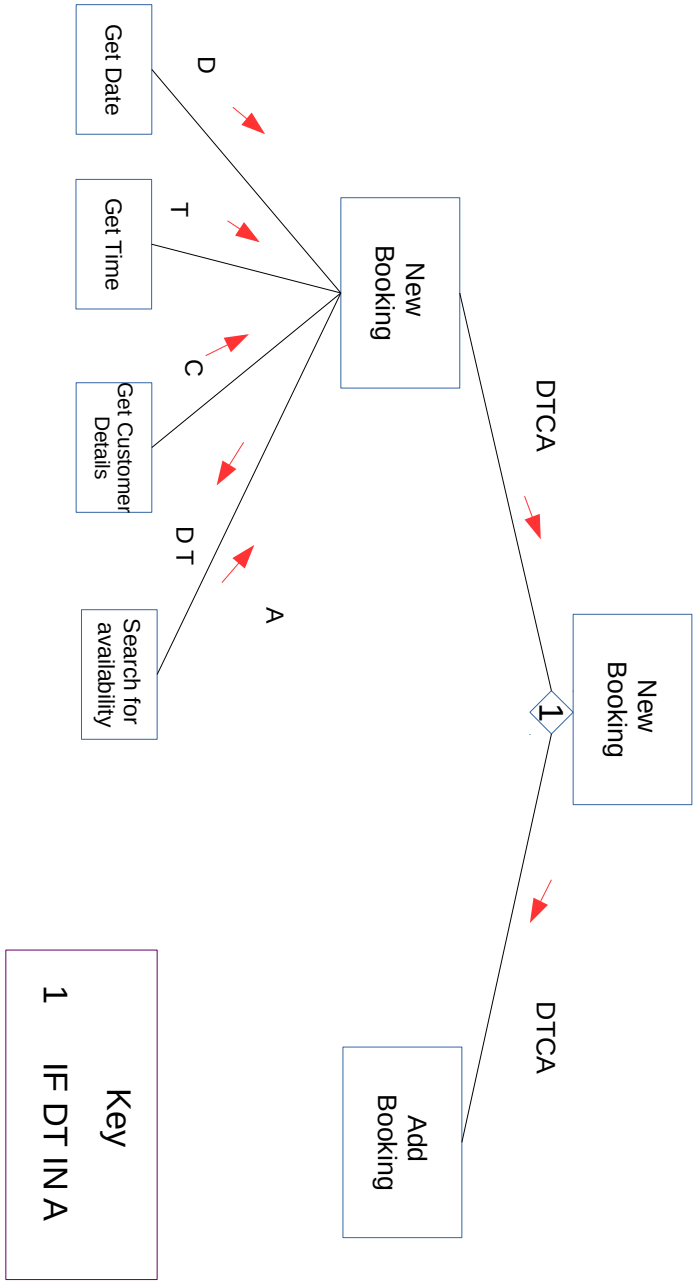


Figure 2.18: Add Booking Structure

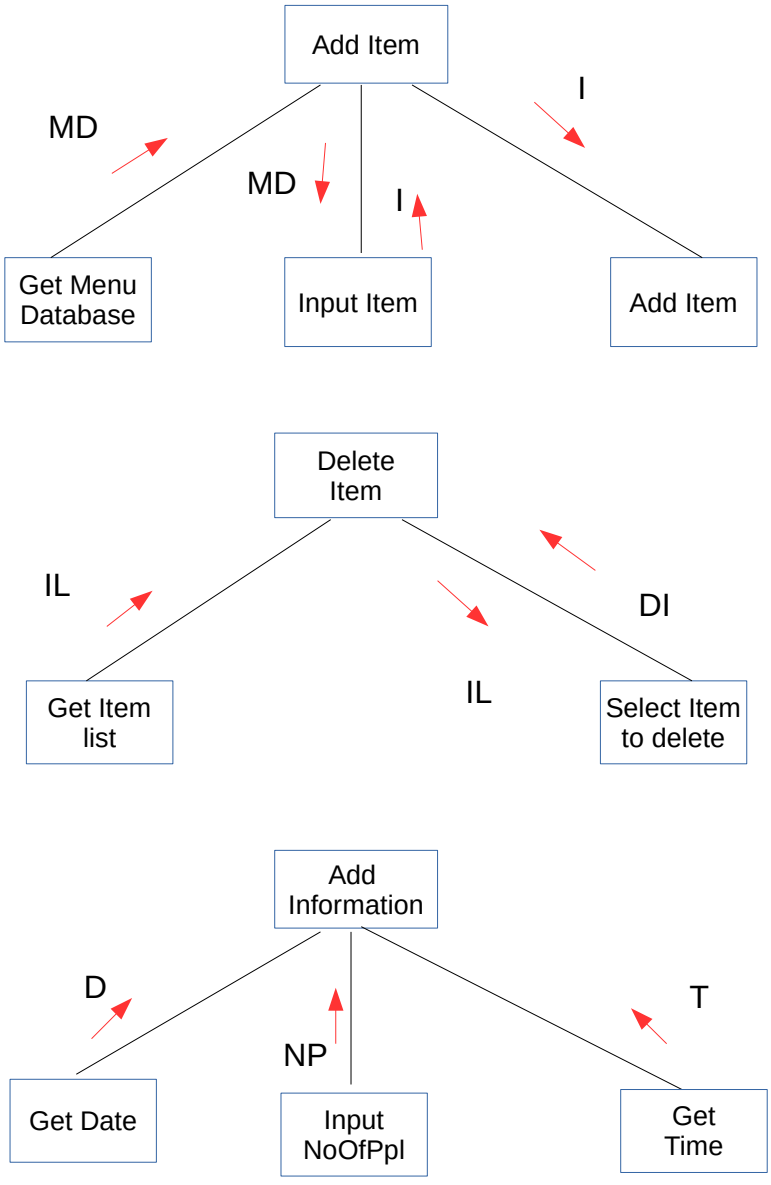


Figure 2.19: Editing Order

2.4.2 Algorithms in pseudo-code for each data transformation process

Algorithm 4 Password change

```
1: OldPassword ← CurrentPassword
2: ValidNewPassword ← False
3:
4: OUTPUT "Please enter the old password"
5: UserCurrentPassword" ← USERINPUT
6:
7: IF UserCurrentPassword = OldPassword THEN
8:     WHILE notValidNewPassword
9:         OUTPUT "Please enter a new password(Must be longer than 4 characters)"
10:        NewPassword ← USERINPUT
11:        OUTPUT "Please re − enter the new password"
12:        ReEnteredNewPassword ← USERINPUT
13:        IF len(NewPassword) > 4 AND NewPassword =
   ReEnteredNewPassword THEN
14:            CurrentPassword ← NewPassword
15:            ValidNewPassword ← True
16:        ELSE
17:            OUTPUT Please try again.
18:        ENDIF
19:    ENDWHILE
20:
21: ELSE
22:    OUTPUT You have entered the wrong password.
23: ENDIF
```

Algorithm 5 Adding an item to an order(MenuID database will need to be retrieved)

```
1:
2: OUTPUT "Please enter a menuID"
3: GetMenuID ← USERINPUT
4: IF GetMenuID in MenuID Database THEN
5:     ItemAdded ← (MenuIDDatabase , MenuItems
       OrderList.insert(ItemAdded)
6: ELSE
7:     OUTPUT You have entered an invalid menuID
8: ENDIF
```

Algorithm 6 Calculating prices

```
1: TotalPrice ← 0
2: OrderLength ← Length(OrderedItems)
3:
4: FOR OrderedItems.Price ← 1 TO OrderLength
5:     TotalPrice ← TotalPrice + OrderedItems.Price
6: ENDFOR
```

2.4.3 Object Diagrams

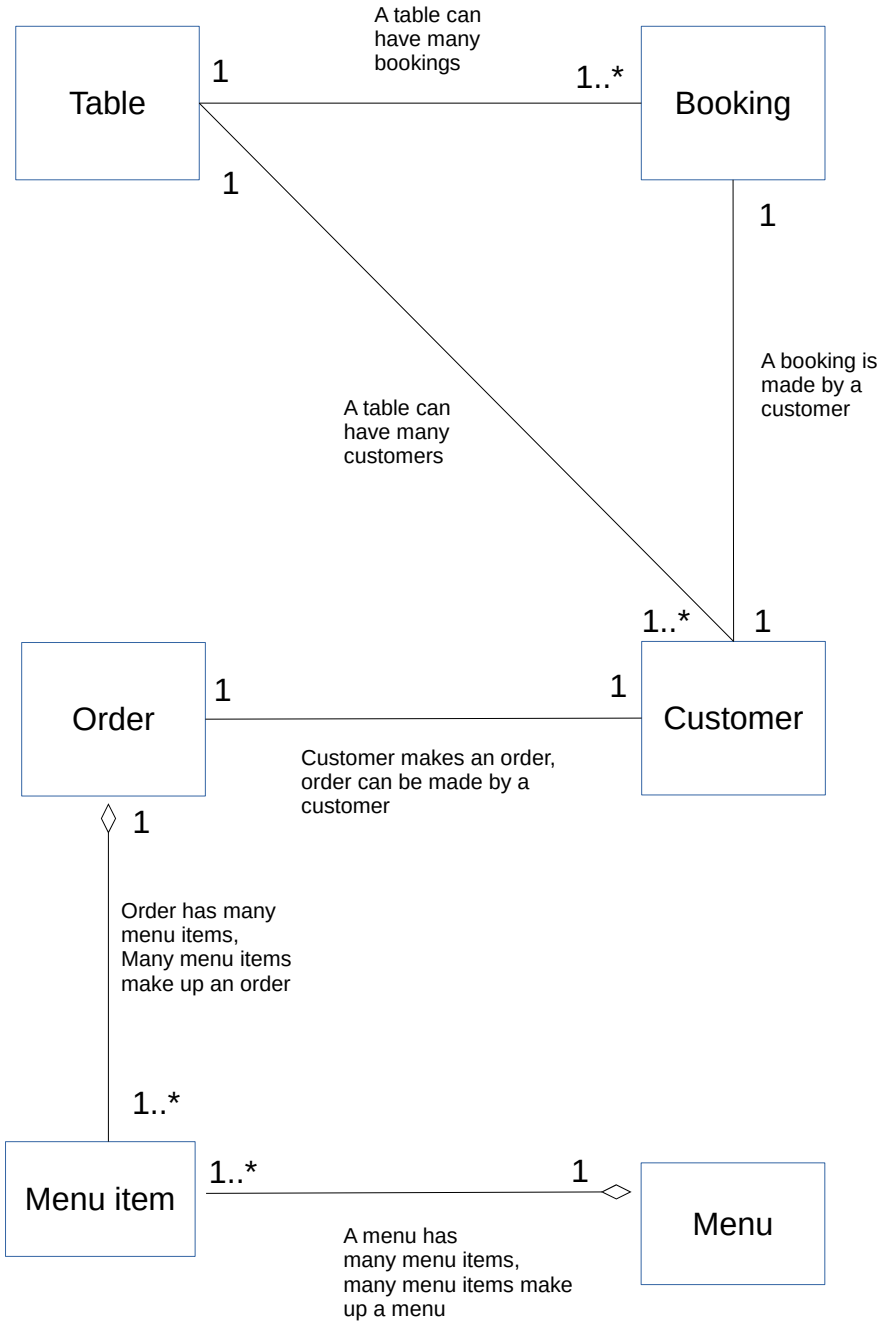


Figure 2.20: Object Diagram

2.4.4 Class Definitions

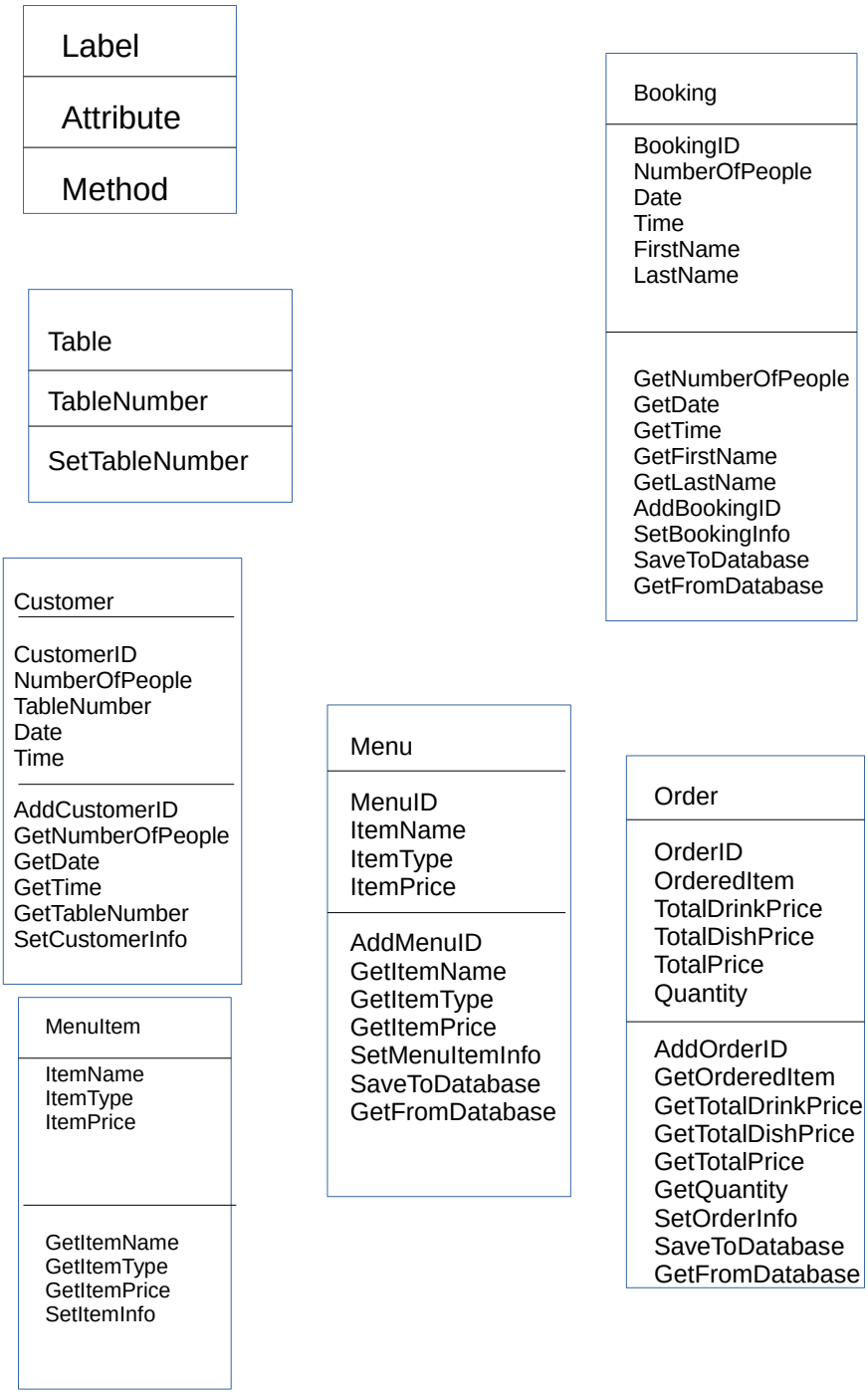


Figure 2.21: Class Definitions

2.5 Prototyping

There are many parts of the system that I would like to prototype due to my limited knowledge of them or its complexity.

I will try to prototye:

- The graphical user interface as this would probably one of the most difficult parts of the system I have to create due to not having a lot of experience in the area.
- Linking tables to the correct current customer order through GUI. By linking, I want it to display all the correct information such as what they ordered.
- The order screen where I would have to function the ability to add items to an order using the database as the source for the items. In addition, displaying the items in a simple and clear layout such as the one on page 46. Also, functioning both Delete and Finish would be parts of the program that I am going to prototype.
- The linking to the database and have the ability to manipulate different records through the GUI. I am not sure how to display tables from the database either and so I will attempt this. I want to display tables because it would help the user to track information such as displaying bookings where the booking date matches the system date.

2.6 Definition of Data Requirements

2.6.1 Identification of all data input items

- Password - used to access program
- Booking name
- Booking telephone number
- Booking time
- Booking date
- Booking table number
- Number of people
- Order menu item - menu item ID from database
- Menu item - adding item to menu
- Menu item type

- Menu item price

2.6.2 Identification of all data output items

Output to order screen

- Dish price
- Drink price
- Total dish price
- Total drink price
- Total price
- Ordered items
- Date of order
- Time of order
- Number of people
- Table number
- Quantity of ordered item

Output to booking screen

- Booking name
- Booking telephone number
- Booking time
- Booking date
- Booking table number
- Booking number of people

Output to database

- Total dish price
- Total drink price
- Total price
- Ordered items
- Quantity of ordered item

- Date of order
- Time of order
- Number of people
- Table number
- Booking name
- Booking telephone number
- Booking time
- Booking date
- Booking table number
- Booking number of people
- Quantity of ordered item
- Menu item
- Menu item price

2.6.3 Explanation of how data output items are generated

Output	How the output is generated
Dish price	Retrieved from the menu database
Drink price	Retrieved from menu database
Total dish price	Calculated by adding up the dish prices
Total drink price	Calculated by adding up the drink prices
Total price	Calculated by adding together total dish price and total drink price
Ordered items	A member of staff inputs information
Quantity of ordered item	A member of staff inputs information
Date of order	Taken from system time
Time of order	Taken from system time
Number of people	A member of staff inputs information
Table number	Predefined by program
Booking name	A member of staff inputs information
Booking telephone number	A member of staff inputs information
Booking time	A member of staff inputs information
Booking date	A member of staff inputs information
Booking table number	A member of staff inputs information
Number of people	A member of staff inputs information
Menu item	A member of staff inputs information when adding a new item to the menu
Menu item price	A member of staff inputs information when adding a new item to the menu
Menu item type	A member of staff inputs information when adding a new item to the menu

2.6.4 Data Dictionary

Data dictionary

Name	Data Type	Length	Validation	Example Data	Comment
TableNumber	Integer	2 Characters	Range	13	Max range will be 16
Number Of People	Integer	2 Characters	Not empty and must be a number	4	Number of people sitting on a table

MenuID	Integer	3 Characters	Range(Not out of range of number of menuIDs)	52	Unique ID to identify an item from the menu
MenuItem	String	1 - 20 Characters	Not empty	Spare ribs	Item description
ItemType	Boolean		Presence Check		If false then type is drink, true is dish
ItemQuantity	Integer	2 Characters	Not empty and must be a number	4	
ItemPrice	Float	4 Characters	Not empty and must be a number	11.20	
Total DrinkPrice	Float	5 Characters	Must be a number	42.35	Added from price of drinks ordered
Total DishPrice	Float	5 Characters	Must be a number	75.63	Added from price of dishes ordered
TotalPrice	Float	5 Characters	Must be a number	154.43	Total price of an order calculated by adding total dishprice and total drinkprice
DateOfOrder	String	4 - 6	Format	16/11/14	
TimeOfOrder	String	4 Characters	Format	07:32	
CustomerID	Integer	2bytes	Number, not used before	0412	Unique ID for someone who sits down and makes an order
OrderID	Integer	2 bytes	Number, not used before	0315	Unique ID for an order
OrderedItem	String	0-20 Characters	Item from menu	Egg fried rice	Item ordered by customer
FirstName	String	2-20 Characters	Not empty or contain numbers	Moe	Used for booking

LastName	String	2-20 Characters	Not empty or contain numbers	Moon	Used for booking
Telephone Number	String	11 Characters	Not empty and 11 numbers	03125 425634	Used for booking
BookingID	Integer	2bytes	Number, not used before	1232	Unique ID when someone makes booking

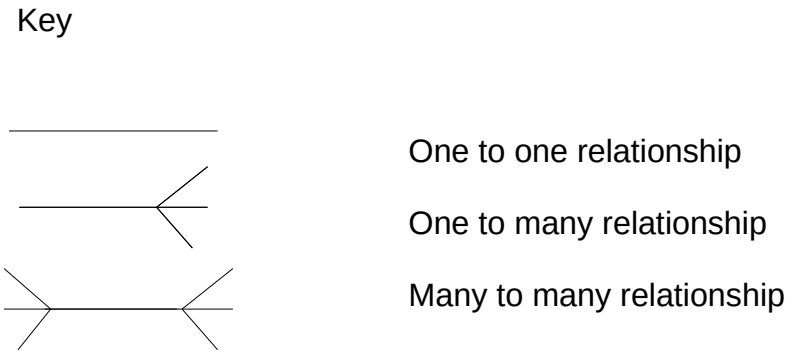
2.6.5 Identification of appropriate storage media

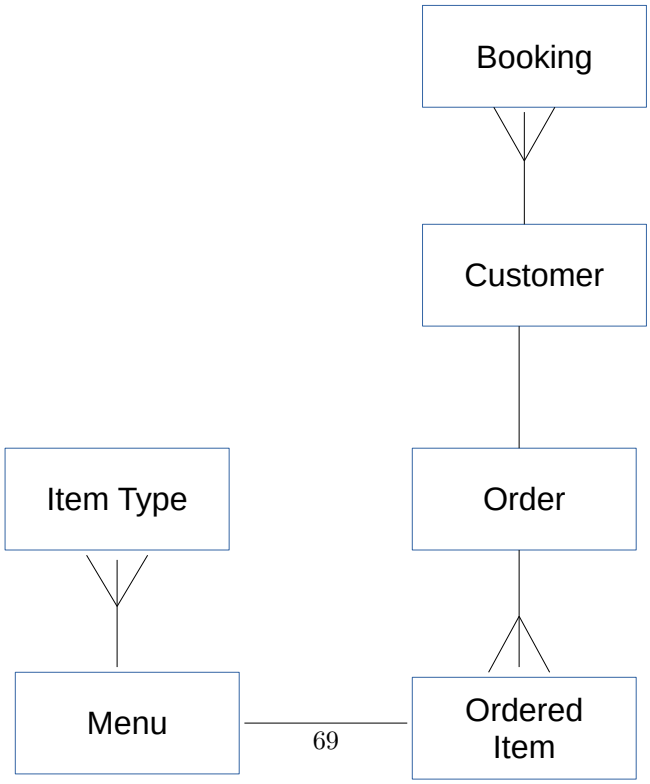
A hard drive would be preferable to store information due to the large capacity size for the database and the speed to transfer data. The only data that will be stored will be stored in the database which will hold old data for 2 years as data older than 2 years will be deleted. The application itself shouldn't be more than 20mb and the database shouldn't take the majority of a hard drive as a lot of hard drives will be more than 100gb, the database shouldn't be 1gb. In addition, a USB flash drive would be a much preferred option to back-up data. A USB flash drive is portable and the capacity size is large enough to store the data from the database. Also, they are immune to mechanical shock, magnetic fields, scratches and dust which makes them suitable for backing-up data - data will not corrupt easily. Almost all computers supports USB in this current time and may still be for many more years as USBs keep getting developed and improved.

2.7 Database Design

2.7.1 Normalisation

ER Diagrams





Entity Descriptions

Customer(CustomerID,*BookingID*, *OrderID*,Date,Time,NoOfPpl,TableNumber)
Booking(BookingID,FirstName,LastName,TelephoneNo,BookingDate,BookingTime)
Menu(MenuID,*Type*, MenuItem, ItemPrice)
ItemType(Type,TypeDescription)
Order(OrderID,TotalDrinkPrice,TotalDishPrice,TotalPrice)
OrderedItem(OrderItemID,*OrderID*,*MenuID*,Quantity)

Key
* Primary Key
- Foreign Key

UNF
CustomerID
Date
Time
NoOfPpl
TableNumber
MenuID
MenuItem
Type
TypeDescription
ItemPrice
OrderID
TotalDrinkPrice
TotalDishPrice
TotalPrice
OrderItemID
Quantity
BookingID
FirstName
LastName
TelephoneNo
BookingDate
BookingTime

1NF

Repeating	Non-Repeating
*OrderID	*CustomerID
*CustomerID	Date
MenuID	Time
MenuItem	NoOfPpl
Type	TableNumber
TypeDescription	BookingID
ItemPrice	FirstName
OrderItemID	LastName
Quantity	TelephoneNo
TotalDrinkPrice	BookingDate
TotalDishPrice	BookingTime
TotalPrice	

2NF

Repeating	Non-Repeating
*OrderID	*CustomerID
*CustomerID	Date
	Time
*OrderID	NoOfPpl
TotalDrinkPrice	TableNumber
TotalDishPrice	
TotalPrice	BookingID
OrderItemID	FirstName
Quantity	LastName
	TelephoneNo
MenuID	BookingDate
MenuItem	BookingTime
Type	
TypeDescription	
ItemPrice	

3NF

*CustomerID
-BookingID
-OrderID
Date
Time
NoOfPpl
TableNumber

*BookingID
FirstName
LastName
TelephoneNo
BookingDate
BookingTime

*MenuID
-Type
MenuItem
ItemPrice

*Type
TypeDescription

*OrderID
TotalDrinkPrice
TotalDishPrice
TotalPrice

*OrderItemID
-OrderID
-MenuID
Quantity

2.7.2 SQL Queries

The following SQL Queries will be formated using Python.

```
   create table  Menu(
        MenuID integer,
        MenuItem text,
        ItemPrice real,
        ItemTypeID integer,
        primary key(MenuID)
        foreign key(ItemTypeID) references
           ItemType(ItemTypeID))
```

This creates a table called Menu with the attributes *MenuItem*, *ItemPrice*. The primary key is *MenuID* and the foreign key is *ItemTypeID*

```
insert into OrderItem
where OrderID = ?, MenuID = ? and Quantity = ?
```

This inserts a new OrderItem record with the attributes *OrderID*, *MenuID* and *Quantity*

```
select *
from Booking
where BookingDate = TodaysDate
```

This will return all of the records from the *Booking* table that has the booking date matched with the present system date. The parameter TodaysDate holds the system date at that current time.

```
delete from Booking
where BookingID = ?
```

This will delete a booking from the *Booking* table with the ID of *BookingID*

```
select *
from OrderedItems
where OrderID = ?
```

This will return all ordered items from an order.

```
update ItemPrice
from Menu
where MenuItem = ?
```

This will update the price of an item from the menu with the item name of what the user chooses.

2.8 Security and Integrity of the System and Data

2.8.1 Security and Integrity of Data

To ensure that certain data is accurate such as prices of items, I will implememnt referential integrity to various tables in my database. Adding referential integrity would mean, if i perform a certain action to a record in a table which is also used in different table, the records in both tables will be both affected by this action. So if I updated a price of an item from the Menu table, this would also update the price of the item in a previous order.

This program will store personal information about customers such as the customer's name and telehphone number and so according to the data protection act, the information must not be kept longer than necessary. Information that is 2 years old will be deleted automatically, this will be done through the start up of the application. T he application will compare the records of the customers booking dates and the system dates, if there is a difference of 2 years,

then the application will delete the records off the database. The information entered must also be accurate and so there will be many validations to make sure information is as accurate as possible.

2.8.2 System Security

I will implement a simple yet effective security feature where a password would need to be inputted by the user to access the program. The user would have to enter the correct password when accessing any data on the system, this will prevent unauthoried access to data. Unauthorised access is also supported by the Computer Misuse Act 1990 which covers:

- unauthorised access to computer material
- unauthorised access to computer material with criminal intent
- unauthorised modification of computer material

2.9 Validation

Item	**Example**	**Validation / Verification applied**	**Comments**
OrderedItem	Wonton soup	Presence check Lookup check	To check that this item exists in menu database
Telephone Number	01325 419603	Presence check Length check Number check	To make sure that a number has been entered which is 11 characters long
FirstName	Rudolph	Presence check	To make sure that a name has been entered
LastName	Moln	Presence check	To make sure that a name has been entered
TableNumber	4	Look up check	Make sure that a non-existing number is not created
MenuID	63	Lookup check	Make sure that a non-existing menuid is not created
MenuItem	Crispy duck	Presence check Lookup check	Check that there aren't repeating menu items
TotalPrice	42.1	Float check	Must be calculated from TotalDrinkPrice and TotalDishPrice
Total Drink Price	1.6	Float check Look up check	Must be calculated from the correct order and drink category
Total Dish Price	40.5	Float check Look up check	Must be calculated from the correct order and dish category
Number Of People	9	Range check	Must be a number but not an unrealistic number like 100 or 0

2.10 Testing

2.10.1 Outline Plan

Test Series	Purpose of Test Series	Testing Strategy	Strategy Rationale
1	Test the flow of control between the user interfaces	Top-down testing	
2	Test validation of data input is detected	Bottom-up testing	Each component will be tested once it is developed
3	Test information input is stored in the correct place	Black box testing	Each component will be tested once it is developed
4	Test algorithms to make sure that the output is correct	White box testing	Each component will be tested once it is developed
5	Test that the system fufils the specification	Acceptance testing	Each component will be tested once it is developed
6	Test database has referential integrity	Integration testing	Each component will be tested once it is developed

2.10.2 Detailed Plan

Test Series	Purpose of Test	Test Description	Test Data	Test Data Type (Normal/ Erroneous/ Boundary)	Expected Result	Actual Result	Evidence

1.01	Test 'Change password' button functions correctly	Should direct user to change password interface	Click Change password button	Normal	Change password interface should be displayed		
1.02	Test Cancel button functions correctly on change password interface	Should redirect user to login screen	Click Cancel button on change password interface	Normal	Change password interfact should close		
1.03	Test interactive table functions correctly	Should direct user to the order details from the table selected	Click on occupied table	Normal	Table information screen should be displayed		
1.04	Test unoccupied table functions correctly	Should direct user to 'add details to table' interface	Click on unoccupied table	Normal	'Add details to table' interface should be displayed		
1.05	Test Table information screen, add button functions correctly	Should direct user to add item interface	Click Add on table information screen	Normal	Add item interface should be displayed		

1.06	Test table information screen, delete function correctly	Should change colour of delete button and red box will appear to indiciate deletion for items	Click Delete button	Normal	Delete button should change colour and red boxes should appear next to each order item		
1.07	Test 'Change password' button functions correctly	Should direct user to change password interface	Click Change password button	Normal	Change password interface should be displayed		
1.08	Test back arrow button functions correctly on table information screen	Should direct user to main screen	Click back arrow button	Normal	User redirected back to main screen should be displayed		
1.09	Test 'Manage Bookings' button functions correctly on main screen	Should direct user to Manage Bookings interface	Click Manage Bookings	Normal	Manage Bookings interface should be displayed		

1.10	Test Add button functions correctly on Manage Bookings interface	Should direct user to create booking interface	Click Add button	Normal	Create booking interface should be displayed		
1.11	Test Cancel button functions correctly on create booking interface	Should redirect user to Manage Bookings interface	Click Cancel button	Normal	User should be redirected to Manage Bookings interface		
1.12	Test back arrow on manage bookings interface functions correctly	Should redirect user to main screen	Click Change back arrow button	Normal	Main screen should be displayed		
1.13	Test Delete button on Manage Bookings screen	Should direct user to bookings display interface	Click Dlete button	Normal	Bookings display should be displayed		
1.14	Test back arrow button functions correctly on bookings display screen	Should redirect user to Manage Bookings interface	Click back arrow button	Normal	User should be redirected to Manage Bookings interface		
2.01	Verify password entered	The field cannot be left blank	(Nothing), Treem	Erroneous, Normal	Error, Accepted		

2.02	Verify new password entered at change password screen	The field cannot be left blank	(Nothing), PineTree	Erroneous, Normal	Error, Accepted		
2.03	Verify retype new password entered at change password screen	The field cannot be left blank	(Nothing), PineTree	Erroneous, Normal	Error, Accepted		
2.04	Verify old password entered at change password screen	The field cannot be left blank	(Nothing) ,Treem	Erroneous, Normal	Error, Accepted		
2.05	Verify Number of people entered at 'add details table'	The field cannot be left blank	(Nothing),3 , pigs	Erroneous, Normal, Erroneous	Error, Accepted, Error		
2.06	Verify MenuID entered at 'add item to order' interface	The field cannot be left blank	(Nothing),3, 9552	Erroneous, Normal, Erroneous	Error, Accepted, Error		
2.07	Verify First Name entered at 'enter booking details' interface	The field cannot be left blank	(Nothing), Milly, 63	Erroneous, Normal, Erroneous	Error, Accepted, Error		

2.08	Verify Last Name entered at 'enter booking details' interface	The field cannot be left blank	(Nothing), Milk, 2	Erroneous, Normal, Erroneous	Error, Accepted, Error		
2.09	Verify Telephone Number entered at 'enter booking details" interface	The field cannot be left blank	(Nothing),01523 859372, 014829, 0158925 8295289	Erroneous, Normal, Erroneous, Errorneous	Error, Accepted, Error, Error		
2.10	Verify Table Number entered at 'enter booking details' interface	The field cannot be left blank	(Nothing),7, Hey	Erroneous, Normal, Erroneous	Error, Accepted, Error		
2.11	Verify Number Of People entered at 'enter booking details' interface	The field cannot be left blank	(Nothing),3, Lisa	Erroneous, Normal, Erroneous	Error, Accepted, Error		
2.12	Verify Date entered at 'enter booking details' interface	The field cannot be left blank	(Nothing),06/05/13, Homer, 032/63/153	Erroneous, Normal, Erroneous, Erroneous	Error, Accepted, Error, Error		

2.13	Verify Time entered at 'enter booking details' interface	The field cannot be left blank	(Nothing),18:12, Bart, 53:62	Erroneous, Normal, Erroneous, Erroneous	Error, Accepted, Error, Error		
3.01	Verify all table details entered are added to relevant database tables	Information should be added to the correct fields in customer, order and orderitem tables. If necessary reservation table	customer information, order information, orderitem information, if necessary reservation table	Normal	Added to customer, order and orderitem table. If necessary reservation table		
3.02	Verify that all details entered at 'enter booking details' interface are added to the reservation database	All of the information should be added to the correct field in the reservations table	Reservation informationl	Normal	Added to the reservation table		

4.01	Verify password changed	Password should not successfully change if length is not bigger than 4 and old password does not match input old password	-Try changing password with incorrect input and length of 2 new password, -Try changing password with new password having length of 2, - Try changing password with correct input and correct length	Error, Error, Accepted			
4.02	Verify add item function works correctly	Entering MenuID will return information based on that ID	Enter ID	Normal	Return all information based on the ID		

4.03	Verify Total price calculation functions correctly	Adds up all items prices together to get a total	Enter items to order	Normal	Calculates the total price based on items entered		
4.04	Check bookings displayed on correct day	Should display all bookings that match with system date	Create a range of bookings that have different dates	Normal	Displays correct bookings		
5.01	Verify program fulfills the specification	Run through the program, testing all aspects to make sure the meet the objectives in the specification	Enter information in all places required input	Normal	Program fulfils specification		
6.01	Verify menu item name updates in case an item is mistakenly spelt	Check the item name is updated in all records that it appears in	Update name of a menu item (Wate to Water)	Normal	Wate should change to Water		

6.02	Verify menu item price updates in case an item is mistakenly priced	Check the price of the item is updated in all records the item appears in	Update price of a menu item (0.060) to (0.60)	Normal	Price should change to 0.60		

Chapter 3

Testing

3.1 Test Plan

3.1.1 Original Outline Plan

Test Series	Purpose of Test Series	Testing Strategy	Strategy Rationale
1	Test the flow of control between the user interfaces	Top-down testing	
2	Test validation of data input is detected	Bottom-up testing	Each component will be tested once it is developed
3	Test information input is stored in the correct place	Black box testing	Each component will be tested once it is developed
4	Test algorithms to make sure that the output is correct	White box testing	Each component will be tested once it is developed
5	Test that the system fufils the specification	Acceptance testing	Each component will be tested once it is developed
6	Test database has referential integrity	Integration testing	Each component will be tested once it is developed

3.1.2 Changes to Outline Plan

Test Series	Purpose of Test Series	Testing Strategy	Strategy Rationale
4	Test algorithms **and SQL statements** to make sure that the output is correct	White box testing	Each component will be tested once it is developed

3.1.3 Original Detailed Plan

The original details plan below looks different than the one in the Design section as I have formatted the plan below so that each test data has its own row.

I have not tested the rows that are in grey due to changes in my program.

Test Series	**Purpose of Test**	**Test Description**	**Test Data**	**Test Data Type (Normal/ Erroneous/ Boundary)**	**Expected Result**	**Actual Result**	**Evidence**
1.01	Test 'Change password' button functions correctly	Should direct user to change password interface	Click Change password button	Normal	Change password interface should be displayed		
*1.02	Test Cancel button functions correctly on change password interface	Should redirect user to login screen	Click Cancel button on change password interface	Normal	Change password interfact should close		
*1.03	Test interactive table functions correctly	Should direct user to the order details from the table selected	Click on occupied table	Normal	Table information screen should be displayed		
*1.04	Test unoccupied table functions correctly	Should direct user to 'add details to table' interface	Click on unoccupied table	Normal	'Add details to table' interface should be displayed	Add details to table displayed - expected	

*1.05	Test Table information screen, add button functions correctly	Should direct user to add item interface	Click Add on table information screen	Normal	Add item interface should be displayed	Add item interface displayed - expected	
*1.06	Test table information screen, delete function correctly	Should change colour of delete button and red box will appear to indiciate deletion for items	Click Delete button	Normal	Delete button should change colour and red boxes should appear next to each order item		
*1.07	Test 'Change password' button functions correctly	Should direct user to change password interface	Click Change password button	Normal	Change password interface should be displayed		
*1.08	Test back arrow button functions correctly on table information screen	Should direct user to main screen	Click back arrow button	Normal	User redirected back to main screen should be displayed		

1.09	Test 'Manage Bookings' button functions correctly on main screen	Should direct user to Manage Bookings interface	Click Manage Bookings	Normal	Manage Bookings interface should be displayed	Manage Bookings interface displayed - expected	3.8 on page 125
1.10	Test Add Booking button functions correctly on Manage Bookings interface	Should direct user to create booking interface	Click Add Booking button	Normal	Create booking interface should be displayed	Create booking interface displayed - expected	3.2 on page 118
*1.11	Test Cancel button functions correctly on create booking interface	Should redirect user to Manage Bookings interface	Click Cancel button	Normal	User should be redirected to Manage Bookings interface		
*1.12	Test back arrow on manage bookings interface functions correctly	Should redirect user to main screen	Click Change back arrow button	Normal	Main screen should be displayed		
1.13	Test Delete Booking button on Manage Bookings screen	Should direct user to delete bookings display interface	Click Delete button	Normal	Delete bookings display should be displayed	Delete bookings layout displayed - expected	3.3 on page 119

*1.14	Test back arrow button functions correctly on bookings display screen	Should redirect user to Manage Bookings interface	Click back arrow button	Normal	User should be redirected to Manage Bookings interface		
* 2.01	Verify password entered	The field cannot be left blank	(Nothing), Treem	Erroneous, Normal	Error, Accepted		
*2.01.01	Verify new password entered at change password screen	The field cannot be left blank	(Nothing), PineTree	Erroneous, Normal	Error, Accepted		
*2.02	Verify retype new password entered at change password screen	The field cannot be left blank	(Nothing), PineTree	Erroneous, Normal	Error, Accepted		
*2.03	Verify old password entered at change password screen	The field cannot be left blank	(Nothing) ,Treem	Erroneous, Normal	Error, Accepted		
2.04	Verify Number of people entered at 'assign customer to table'	User inputs nothing	(Nothing)	Erroneous	Error	Nothing - expected	3.4 on page 121

2.04.01	Verify Number of people entered at 'assign customer to table'	User inputs value	3	Normal	Accepted	Accept input - expected	
2.04.02	Verify Number of people entered at 'assign customer to table'	User inputs value	pigs	Erroneous	Error	Can only enter numbers - expected due to regular expression	
2.05	Verify ItemID entered at 'add item to order' interface	User inputs vnothing	(Nothing)	Erroneous	Error/ nothing	No changes - expected	
2.05.01	Verify ItemID entered at 'add item to order' interface	User inputs value	3	Normal	Accepted	Accepted	
2.05.02	Verify ItemID entered at 'add item to order' interface	User inputs value	9552	Erroneous	Error	Only allowed to input 3 digits	

2.06	Verify First Name entered at 'enter booking details' interface	User inputs nothing	(Nothing)	Erroneous	Error	Add booking did not proceed after clicking add booking - expected	
2.06.01	Verify First Name entered at 'enter booking details' interface	User inputs name	Milly	Normal	Accepted	Milly was accepted and booking proceeded - expected	
2.06.02	Verify First Name entered at 'enter booking details' interface	User inputs name	63	Erroneous	Error	Could not enter numbers - expected due to regular expression	
2.07	Verify Last Name entered at 'enter booking details' interface	User inputs name	(Nothing)	Erroneous	Error	Add booking did not proceeed after clicking add booking	
2.07.01	Verify Last Name entered at 'enter booking details' interface	User inputs name	Milk	Normal	Accepted	Milk was accepted and booking proceeded - expected	

2.07.02	Verify Last Name entered at 'enter booking details' interface	User inputs name	2	Erroneous	Error	Could not enter numbers - expected due to regular expression	
2.08	Verify Telephone Number entered at 'enter booking details" interface	User inputs nothing	(Nothing)	Errorneous	Error	Add booking did not proceed - expected	
2.08.01	Verify Telephone Number entered at 'enter booking details" interface	User inputs number	01523 859372	Normal	Accepted	Add booking did proceed - expected	
2.08.02	Verify Telephone Number entered at 'enter booking details" interface	User inputs number	014829	Boundary	Error	Add booking did not proceed - expected	
*2.09	Verify Table Number entered at 'enter booking details' interface	User inputs number	(Nothing)	Erroneous	Error		

*2.09.01	Verify Table Number entered at 'enter booking details' interface	User inputs number	7	Normal	Accepted		
*2.09.02	Verify Table Number entered at 'enter booking details' interface	User inputs number	Hey	Erroneous	Error		
2.10	Verify Number Of People entered at 'enter booking details' interface	User inputs nothing	(Nothing)	Erroneous	Error	Add booking did not proceed - expected	
2.10.01	Verify Number Of People entered at 'enter booking details' interface	User inputs number	3	Normal	Accepted	Add booking proceeded - expected	
2.10.02	Verify Number Of People entered at 'enter booking details' interface	User inputs number	Lisa	Erroneous	Error	Could not enter letters - expected due to regular expression	

*2.11	Verify Date entered at 'enter booking details' interface	User inputs date	(Nothing	Erroneous	Error		
*2.11.01	Verify Date entered at 'enter booking details' interface	User inputs date	06/05/13	Normal	Accepted		
*2.11.02	Verify Date entered at 'enter booking details' interface	User inputs date	Homer	Erroneous	Error		
*2.11.03	Verify Date entered at 'enter booking details' interface	User inputs date	032/63/153	Erroneous	Error		
*2.12	Verify Time entered at 'enter booking details' interface	User inputs time	(Nothing)	Erroneous	Error		
2.12.01	Verify Time entered at 'enter booking details' interface	User inputs time	18:12	Normal	Accepted	Add booking proceeded - expected	

*2.12.02	Verify Time entered at 'enter booking details' interface	User inputs time	Bart	Erroneous	Error		
*2.12.03	Verify Time entered at 'enter booking details' interface	User inputs time	53:62	Erroneous	Error		
*3.01	Verify all table details entered are added to relevant database tables	Information should be added to the correct fields in customer, order and orderitem tables. If necessary reservation table	customer information, order information, orderitem information, if necessary reservation table	Normal	Added to customer, order and orderitem table. If necessary reservation table		
3.02	Verify that all details entered at 'enter booking details' interface are added to the booking database	All of the information should be added to the correct field in the booking table	Booking information	Normal	Relevent details added to booking table	All details have been added to relevent database tables - expected	

*4.01	Verify password changed	Password should not successfully change if length is not bigger than 4 and old password does not match input old password	Try changing password with incorrect input and length of 2 new password,	Error			
*4.01.01	Verify password changed	Password should not successfully change if length is not bigger than 4 and old password does not match input old password	Try changing password with new password having length of 2	Error			
*4.01.02	Verify password changed	Password should not successfully change if length is not bigger than 4 and old password does not match input old password	Try changing password with correct input and correct length	Accepted			

*4.02	Verify add item function works correctly	Entering MenuID will return information based on that ID	Enter ID	Normal	Return all information based on the ID		
4.03	Verify Total price calculation functions correctly	Adds up all items prices together to get a total	Enter items to order	Normal	Calculates the total price based on items entered	Total doesnt update - unexpected	
4.04	Check bookings displayed on correct day	Should display all bookings that match with system date	Create a range of bookings that have different dates	Normal	Displays correct bookings		

5.01	Verify program fulfills the specification	Run through the program, testing all aspects to make sure the meet the objectives in the specification	Enter information in all places required input	Normal	Program fulfils specification	Can run through program without any problems, some minor objectives were not met such as having clickable tables (I have radio buttons instead).	
*6.01	Verify menu item name updates in case an item is mistakenly spelt	Check the item name is updated in all records that it appears in	Update name of a menu item (Wate to Water)	Normal	Wate should change to Water		
*6.02	Verify menu item price updates in case an item is mistakenly priced	Check the price of the item is updated in all records the item appears in	Update price of a menu item (0.060) to (0.60)	Normal	Price should change to 0.60		

I have removed some tests under 2.09, 2.11 and 2.12 due to changes in my program which made it impossible to have the wrong input in terms of erroneous and boundary inputs. For example, test 2.09 was to verify the table number inputted was valid, I have made my program so now the user can only select tables which exist through a combo box. As for tests 2.11 and 2.12, the user is forced into using the correct times/dates format. I set the minimum date for QDateEdit to be the system date which would mean the user will not be able to input boundary data (make a booking for yesterday).

3.1.4 Changes to Detailed Plan

Test Series	**Purpose of Test**	**Test Description**	**Test Data**	**Test Data Type (Normal/ Erroneous/ Boundary)**	**Expected Result**	**Actual Result**	**Evidence**
1.15	Test Add Item on 'Item Menu' menu bar	Check if Add Item layout is displayed after clicking on Add Item	Click on Add Item	Normal	Add Item layout displayed	Add Item layout displayed - expected	
1.16	Test Delete Item on 'Item Menu' menu bar	Check if Delete Item layout is displayed after clicking on Delete Item	Click on Delete Item	Normal	Delete Item layout displayed	Delete Item layout is displayed	Example

1.17	Test Update Item Price on 'Item Menu' menu bar	Check if Update Item Price layout is displayed after clicking on Update Item Price	Click on Update Item Price	Normal	Update Item Price layout displayed	Update Item Price layout displayed - expected	
1.18	Test Add Booking on 'Bookings' menu bar	Check if Add Booking layout is displayed after clicking on Add Booking	Click on Add Booking	Normal	Add Booking layout displayed	Add Booking layout displayed - expected	
1.19	Test Delete Booking on 'Bookings' menu bar	Check if Delete Booking layout is displayed after clicking on Delete Booking	Click on Delete Booking	Normal	Delete Booking layout displayed	Delete Booking layout displayed - expected	
1.20	Test Update Booking on 'Bookings' menu bar	Check if Update Booking layout is displayed after clicking on Update Booking	Click on Update Booking	Normal	Update Booking layout displayed	Update Booking layout displayed - expected	
1.21	Test 'Search Order' on tool bar	Check if Search Order layout is displayed after clicking on Search Order	Click on Search Order	Normal	Search Order layout displayed	Search Order layout displayed - expected	

1.22	Test 'View Bookings' on tool bar	Check if View Bookings layout is displayed after clicking on View Bookings	Click on View Bookings	Normal	View Bookings layout displayed	View Bookings layout displayed - expected	
1.23	Test 'View Customers' on tool bar	Check if View Customers layout is displayed after clicking on View Customers	Click on View Customers	Normal	View Customers layout displayed	View Customers layout displayed - expected	
1.24	Test 'View Dishes' on tool bar	Check View Dishes layout is displayed after clicking on View Dishes	Click on View Dishes	Normal	View Dishes layout displayed	View Dishes layout displayed - expected	
1.25	Test 'View Drinks' on tool bar	Check if View Drinks layout is displayed after clicking on View Drinks	Click on View Drinks	Normal	View Drinks layout displayed	View Drinks layout displayed - expected	
1.26	Test 'Main Screen' on tool bar	Check if Main Screen is displayed after clicking on 'Main Screen'	Click on Main Screen from Search Order layout	Normal	Main Screen layout displayed	Main Screen layout displayed - expected	

1.27	Test table radio buttons on main screen	Check if dialog box shows after clicking on Select Table	Choose an unoccupied table and click Select Table	Normal	Assign customer dialog box shows	Assign customer dialog box is shown - expected	
1.28	Test Assign customer layout	Check if manage order box shows after clicking on Create (after filling in required field)	Click on Create	Normal	Relevent manage order dialog box shows	Relevent manage order dialog box is shown - expected	
1.28.01	Test Assign customer layout	Check if manage order box shows after clicking on Select	Click on Select	Normal	Relevent manage order dialog box shows	Relevent manage order dialog box is shown	
1.29	Test Add button on manage order box	Check if Add Item To Order box shows after clicking on Add	Click on Add	Normal	Add Item To Order dialog box shows	Add Item To Order dialog box is shown - expected	
1.30	Test Delete button on manage order box	Check Delete Item Off Order box shows after clicking on Delete	Click on Delete	Normal	Delete Item Off Order dialog box shows	Delete Item Off Order dialog box is shown	

1.31	Test Finish button on manage order box	Check if Manage Order box closes after clicking on Finish	Click on Finish	Normal	Manage Order box closes	Manage Order box closes - expected	
1.32	Test Invoice Preview button on manage order box	Check if the preview of the invoice box shows after clicking on Invoice Preview	Click on Invoice Preview	Normal	Invoice preview shows	Invoice shown - expected	3.7 on page 124
1.33	Test Print Invoice button on manage order box	Check if print option appear after clicking Print Invoice	Click on Print Invoice	Normal	Print options appear	Print options appeared - expected	
2.04.03	Verify Number of people entered at 'assign customer to table'	Check if user input is valid after clicking Create	0	Boundary	Input not accepted	Input was not accepted - expected	
2.13	Verify Item Name input at Add Item to Menu	Check if user input is valid after clicking Add Item assuming all other fields are filled with normal data	Rice	Normal	Accepted	Item was successfully added	

2.13.01	Verify Item Name input at Add Item to Menu	Check if user input is valid after clicking Add Item assuming all other fields are filled with normal data	(Nothing)	Erroneus	Error	Item add unsuccessful - expected	
2.14	Verify ItemID input at Update Item Price	Check if user input is valid after clicking Update Item assuming all other fields are filled with normal data	7	Normal	Accepted	Price was successfully updated	
2.14.01	Verify ItemID input at Update Item Price	Check if user input is valid after clicking Update Item assuming all other fields are filled with normal data	0	Boundary	Error	Nothing happened - expected	
2.15	Verify Number Of People at Update Booking	Check if user input is valid after clicking Update Number Of People with a booking that exists	5	Normal	Accepted	Booking updated - expected	

2.15.01	Verify Number Of People at Update Booking	Check if user input is valid after clicking Update Number Of People with a booking that exists	0	Boundary	No changes will be made	Booking did not update - expected	
2.16	Verify Item Name at Delete Item Off Menu	Check if user input is valid after clicking Delete Item for Item Name	Nothing	Erroneous	Error(no chages will be made)	The process of deleting an item did not happen - expected	
2.16.01	Verify Item Name at Delete Item Off Menu	Check if user input is valid after clicking Delete Item for Item Name	Steak	Normal	Steak will be deleted	Steak was successfully deleted (removed from the displayed table widget) - expected	
2.17	Verify Item ID at Delete Item Off Menu	Check is user input is valid after clicking Delete Item for Item Name	10(Steak which ive added again)	Normal	Record with Item ID 10 deleted	Item ID 10 was deleted (removed from the displayed table) - expected	

2.17.01	Verify Item ID at Delete Item Off Menu	Check is user input is valid after clicking Delete Item for Item Name	(Nothing)	Erroneous	Error(no changes will be made)	The process of deleting an item did not happen - expected	
2.17.02	Verify Item ID at Delete Item Off Menu	Check is user input is valid after clicking Delete Item for Item Name	645(There was not an item with itemID 645)	Boundary	Error(no changes will be made)	The process of deleting an item did not happen - expected	
2.18	Verify Booking ID at Delete Booking	Check is user input is valid	(Nothing)	Erroneous	Error(no changes will be made)	The process of deleting a booking did not happen - expected	
2.18.01	Verify Booking ID at Delete Booking	Check is user input is valid	888	Boundary	Error(no changes will be made)	There was not an error but the process of deleting a booking did not happen	

2.18.02	Verify Booking ID at Delete Booking	Check is user input is valid	18(Booking which ive added for testing purposes)	Normal	Booking deleted	Booking removed(Booking was removed from the displayed boking table) - expected	
3.03	Check all details from the creation of a booking at Assign Customer to table is transferred	Check all correct details are transferred over to the Manage Order box	Click on Create after filling in fields correctly	Normal	All relevent details displayed	All relevent details transferred	3.5 on page 122
3.04	Check all details from the selecting of a booking at Assign Customer to table is transferred	Check all correct details are transferred over to the Manage Order box	Click on Select after selecting a booking from combo box	Normal	All relevent details displayed	All relevent details transferred 3.8 on page 125	
3.05	Check all details from an order is transferred to invoice	Compare the order details to invoice details to make sure they are the same	Click on Invoice Preview	Normal	All relevent details displayed	All relevent details transferred	3.7 on page 124

3.06	Booking added is stored in correct table	After successfully adding a booking, the booking should be displayed in table above	Add Booking	Normal	All relevent details displayed	All relevent details displayed in table above	
3.07	Customer added is stored in correct table	After successfully adding a booking, a customer record should be added to Customer table	Add Booking	Normal	All relevent details displayed	All relevant details added to table	
3.08	Item added to menu is stored in correct table	After successfully adding an item, an item record should be appear on Items table	Add Booking	Normal	All relevent details displayed	New item appeared	
4.03.01	Verify Total price calculation algorithm is correct	Check that an algorithm adds up all items prices together to get a total	Check invoice preview to see total price	Normal	Calculates the total price based on items ordered	Correct total price - expected	

4.05	Check if the 'Increasing the quantity of an ordered item' algorithm works	A customer can order x more of an item - the quantity should therefore increase by x amount	Add an item initially then add 10 more if it	Normal	Should expect quantity to be 11	Quantity increased to 11 - expected	3.6 on page 123
4.06	Check if only drinks are displayed on View Drinks tool bar	After clicking View Drinks, a table should appear with only drinks in it	Click on View Drinks	Normal	Only items with ItemTypeID 2 appear	Drinks were only displayed - expected	
4.07	Check if only dishes are displayed on View Dishes tool bar	After clicking View Dishes, a table should appear with only dishes in it	Click on View Dishes	Normal	Only items with ItemTypeID 1 appear	Dishes were only displayed	
4.08	Check search order function	Leave the booking field blank and click Search Order	(Nothing)	Erroneous	Error /empty table appear	Empty table appeared	
4.08.01	Check search order function	Enter a booking ID that doesn't exist and click Search Order	93	Boundary	Error / empty table appear	Empty table appeared	

4.08.02	Check search order function	Enter a booking ID that exists and click Search Order	11	Normal	A populated table appear	Correct table appeared	3.9 on page 126
6.03	Ensure order gets deleted when booking gets deleted	After deleting a booking, the order should be deleted with it.	Delete a booking which has booking items	Normal	Order should be deleted	Order(Booking items) has been deleted	3.10 on page 127
6.04	Check if an item that does not exist is added to an order	Adding an item that does not exist from the manage order box	Add Item ID 933	Boundary	Nothing should happen as there isnt an item with an ID of 993	No items were added	

3.2 Test Data

3.2.1 Original Test Data

Test Number	**Test Data**	**Justification for choice of test data**
2.04.01	3	Program must allow the correct input
2.04.01	pigs	Program must not allow the wrong input, regular expression shouldn't allow letters to be inputted
2.05	(Nothing)	User could accidently try to proceed without entering anything which shouldn't be accepted by validation
2.05.01	3	Program must allow the correct input
2.05.02	9552	Program must not allow an erroneous input
2.06	(Nothing)	User could accidently try to proceed without entering anything which shouldn't be accepted by validation
2.06.01	3	Program must allow the correct input
2.06.02	63	Program must not allow the wrong input, regular expression shouldn't allow numbers to be inputted
2.07	(Nothing)	User could accidently try to proceed without entering anything which shouldn't be accepted by validation
2.07.01	Milk	Program must allow the correct input
2.07.02	2	Program must not allow the wrong input, regular expression shouldn't allow numbers to be inputted
2.08	(Nothing)	User could accidently try to proceed without entering anything which shouldn't be accepted by validation
2.08.01	0152385972	Program must allow the correct input (11 numbers)
2.08.02	014829	Program must not allow an invalid number (not 11 numbers)
2.10	(Nothing)	User could accidently try to proceed without entering anything which shouldn't be accepted by validation
2.10.01	3	Program must allow the correct input
2.10.02	2	Program must not allow the wrong input, regular expression shouldn't allow letters to be inputted

3.2.2 Changes to Test Data

Some of the new test's test data will be included in the table below.

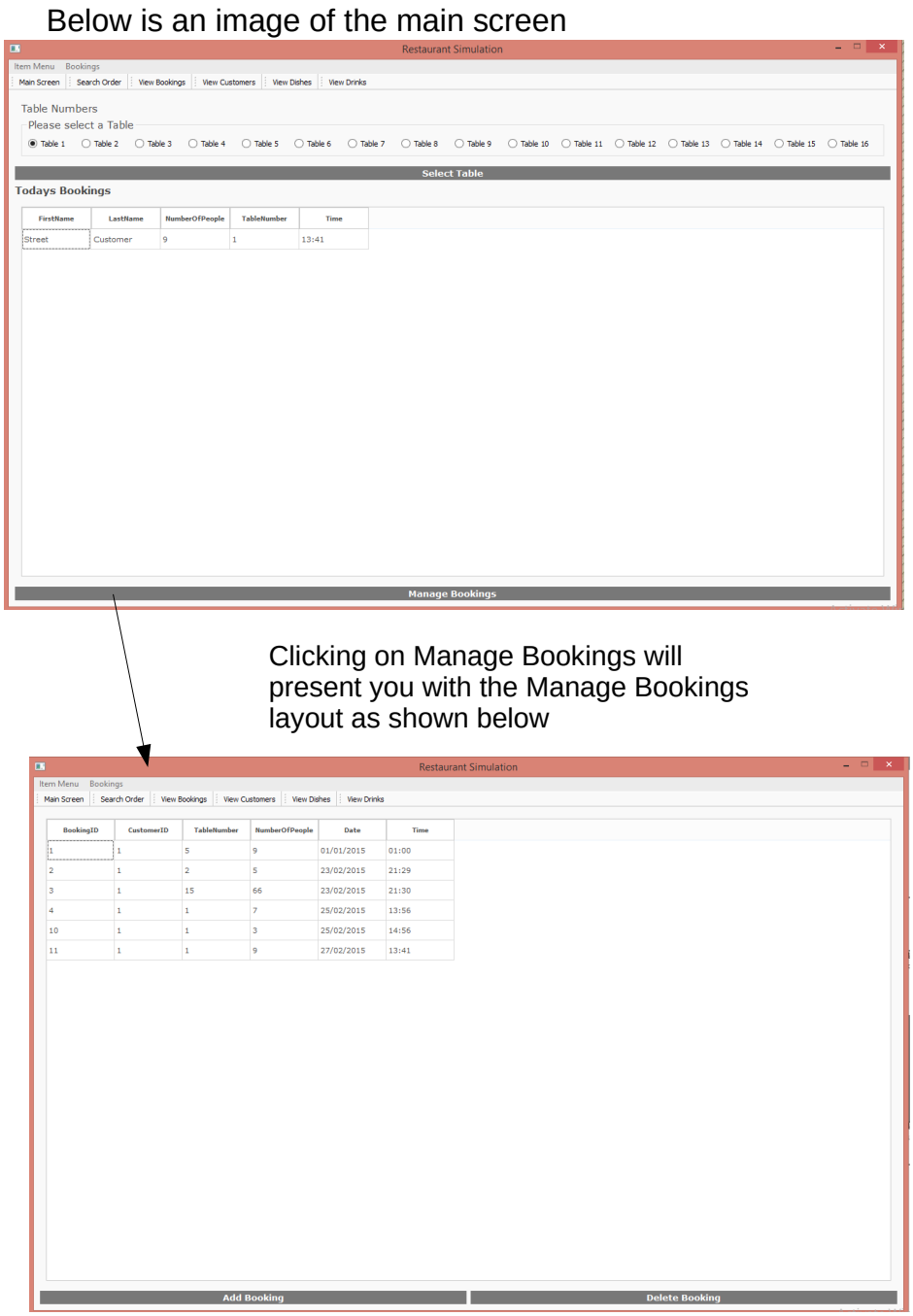
Test Number	**Test Data**	**Justification for choice of test data**
2.04.03	0	Program should not allow 0 as it would not make sense if there was a booking for 0 people
2.13	Rice	Program must allow the correct input
2.13.01	(Nothing)	User could accidently try to proceed without entering anything which shouldn't be accepted by validation
2.14	7	Program must allow the correct input
2.14.01	0	Program must not allow 0 because it will not exist in the database
2.15	5	Program must allow the correct input
2.15.01	0	Program should not allow 0 as it would not make sense if there was a booking for 0 people
2.16	(Nothing)	User could accidently try to proceed without entering anything which shouldn't be accepted by validation
2.16.01	Steak	Program must allow the correct input
2.17	10(Steak)	Program must allow the correct input
2.17.01	(Nothing)	User could accidently try to proceed without entering anything which shouldn't be accepted by validation
2.17.02	645	Should allow the correct input but not do anything/error stating input is incorrect
2.18	(Nothing)	User could accidently try to proceed without entering anything which shouldn't be accepted by validation
2.18.01	888	Should allow 888 as it is a valid input but not do anything as there is not a booking with an id of 888 in the current database used for testing
2.18.02	18	Should allow the correct input but not do anything/error stating input is incorrect
4.05	Have an item with a quantity of 1 then add 10 more of it	I have chosen 1 as the initial quantity and the additional 10 because it will be very clear if the adding quantity algorithm works.

3.3 Annotated Samples

3.3.1 Actual Results

The actual results for the tests can be found on the detailed plans, there is a seperate column for it named 'Actual Results'.

3.3.2 Evidence



1.pdf

Figure 3.1: Manage Order layout

Clicking on Add Booking will present the Add Booking layout as shown below.

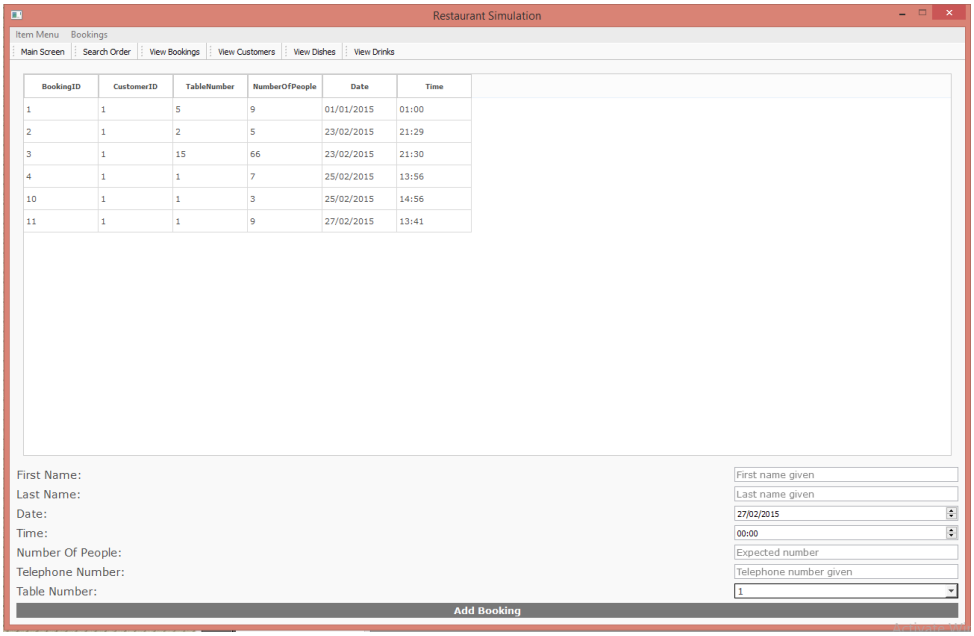


Figure 3.2: Add Booking

Clicking on Delete Booking will present the Delete Booking layout as shown below.

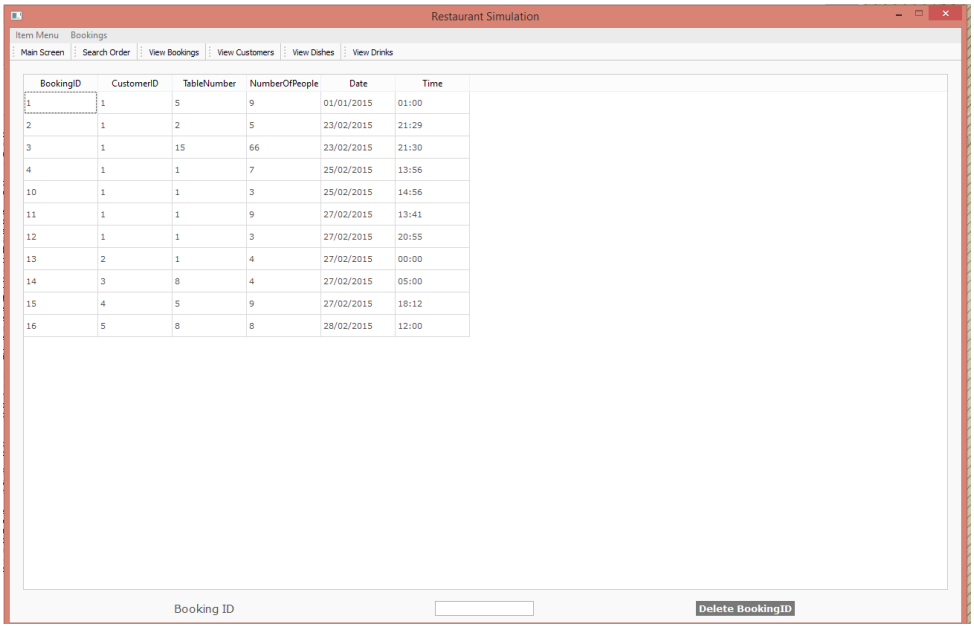
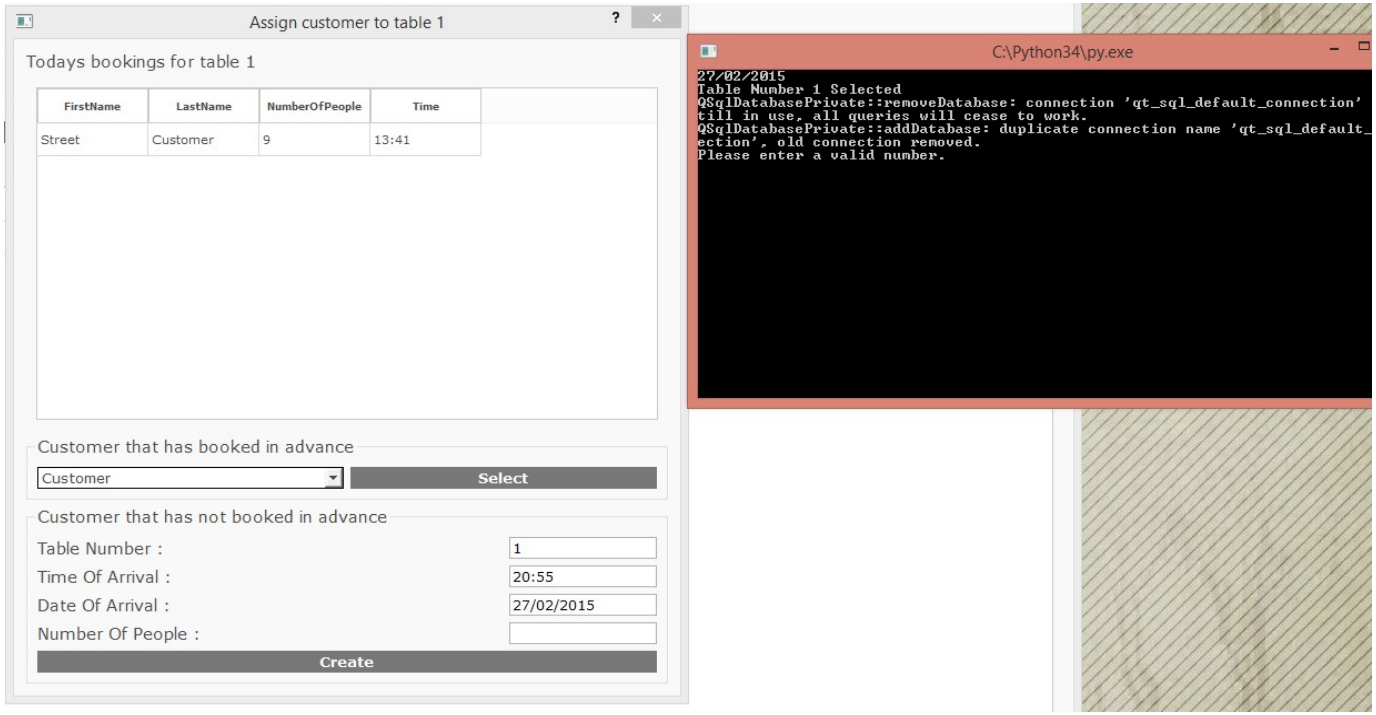
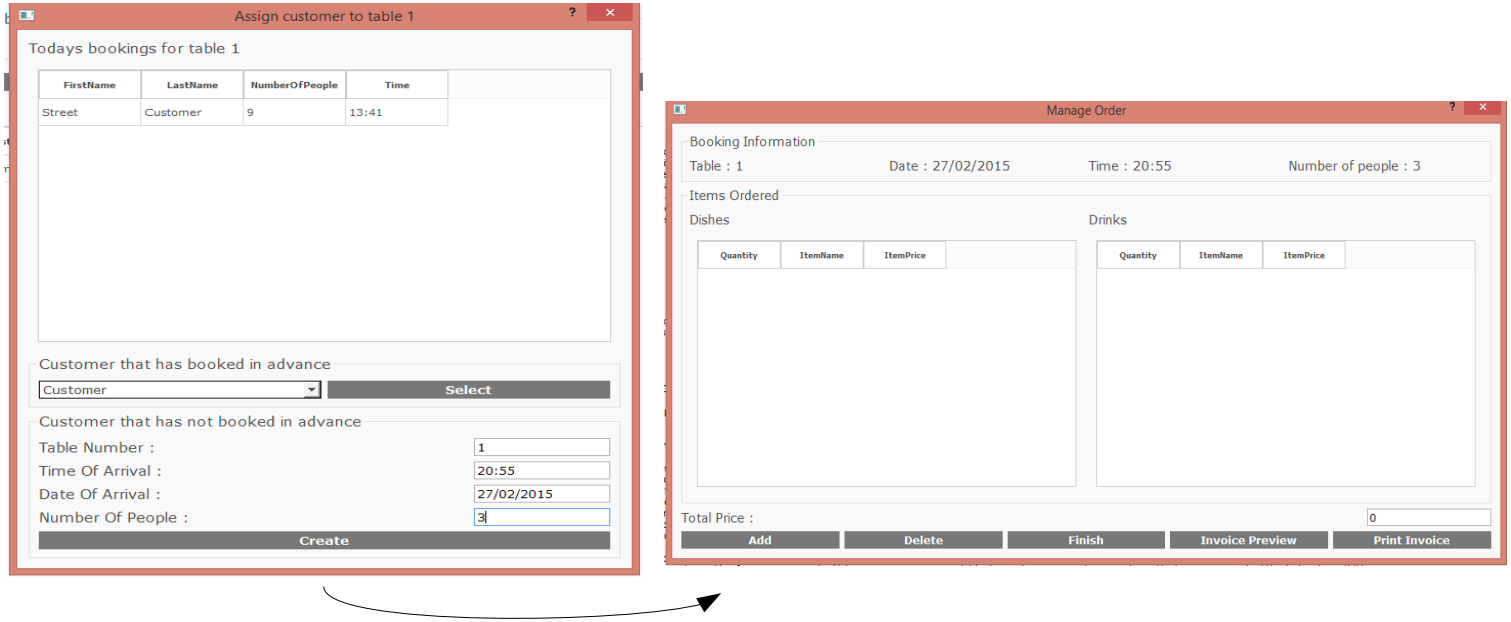


Figure 3.3: Delete Booking layout



Leaving the number of people field empty and clicking on create gave a print statement Of 'Please enter a valid number'. The booking was not created.

Figure 3.4: Assign customer validation



All relevant details such as table number, time of arrival, date of arrival and the number of people transferred after clicking Create.

Figure 3.5: Check all details transferred

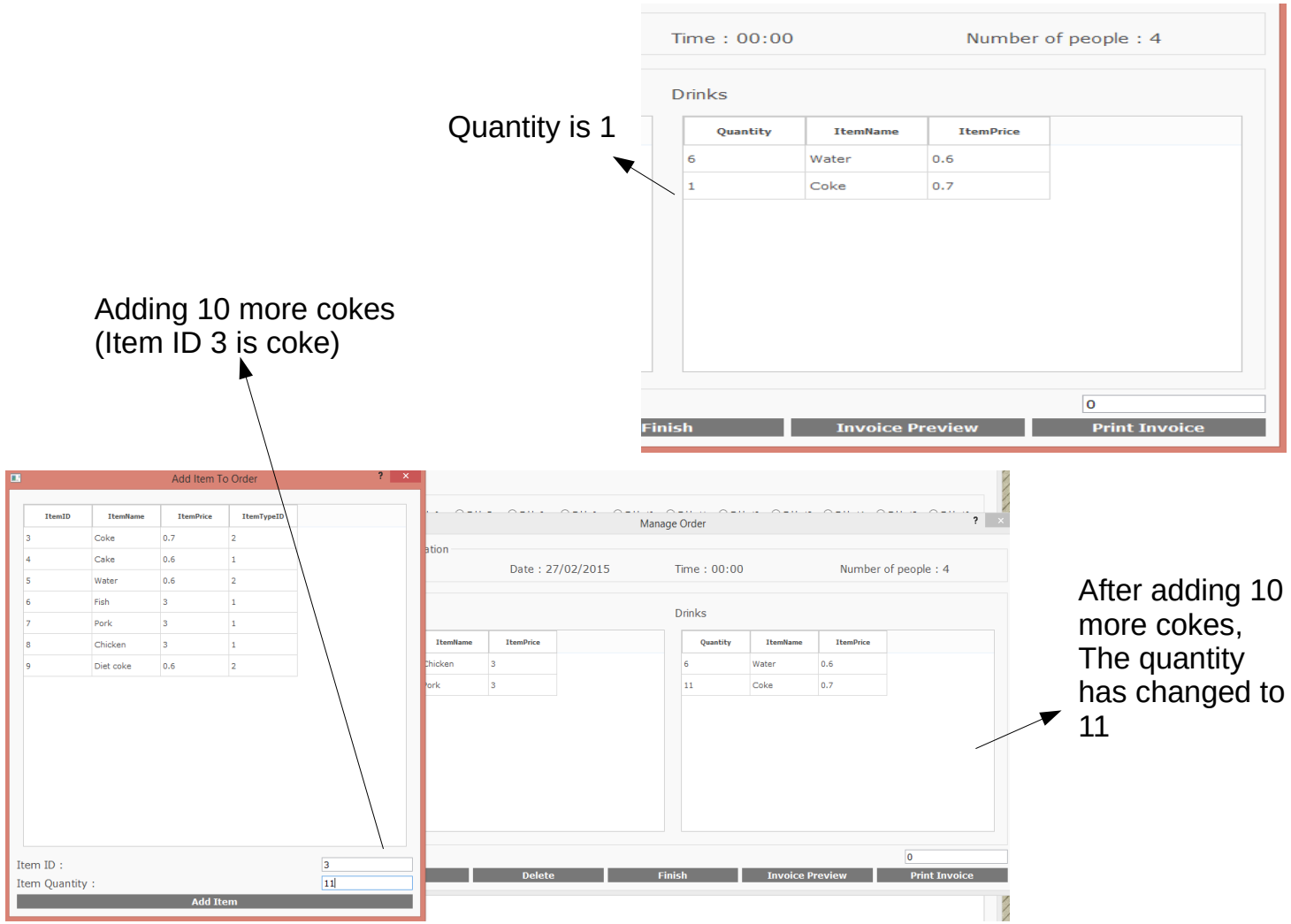
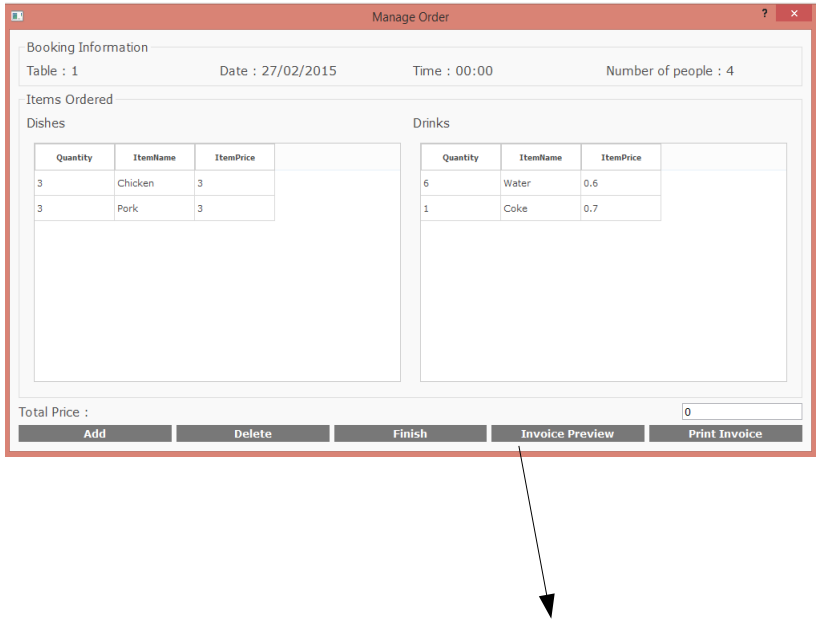
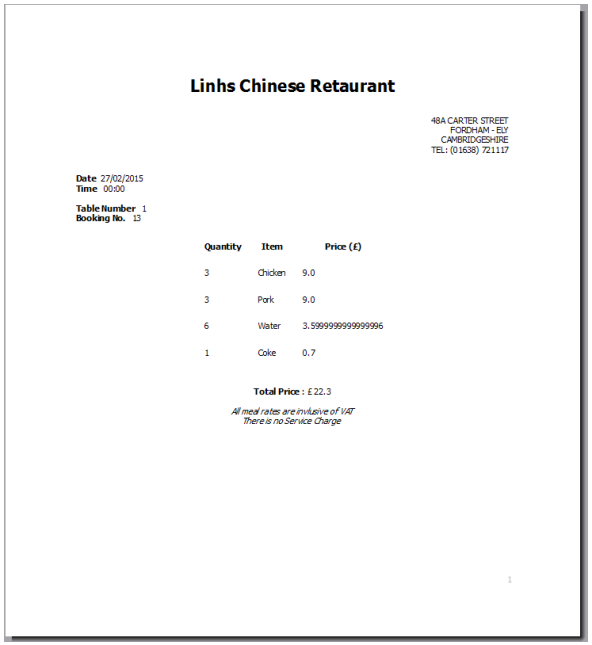


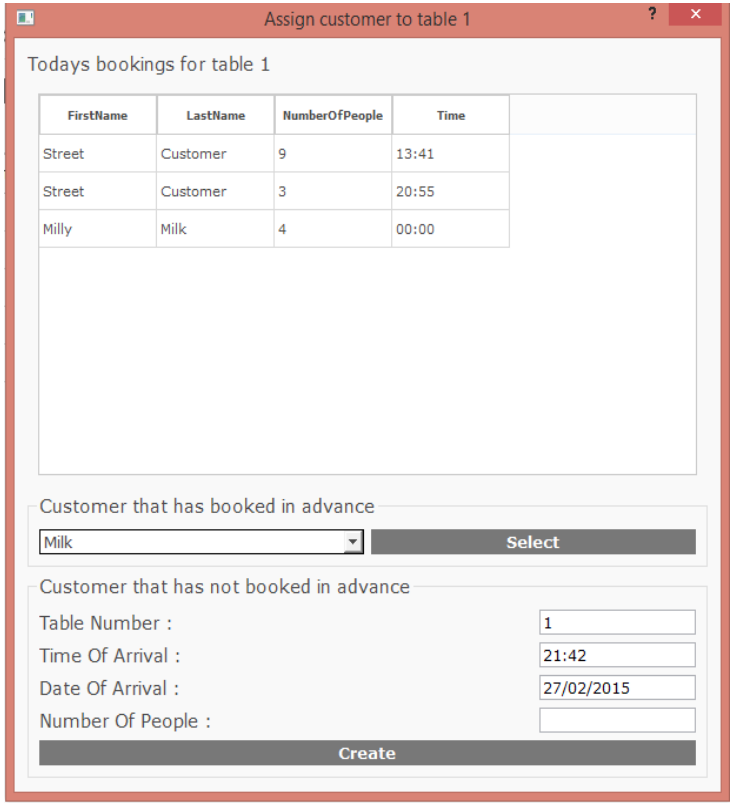
Figure 3.6: Quantity check

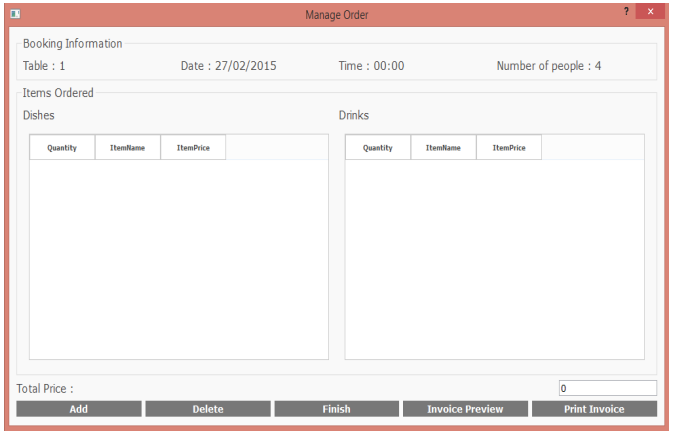




Clicking on Invoice Preview has made the preview pop up.
The details have been copied from the manage order to the invoice as shown

Figure 3.7: Invoice check





Selecting the customer Milly Milk and pressing select has passed the details of the booking to the manage order box

Figure 3.8: Check the select function works (passes the correct details of the selected customer

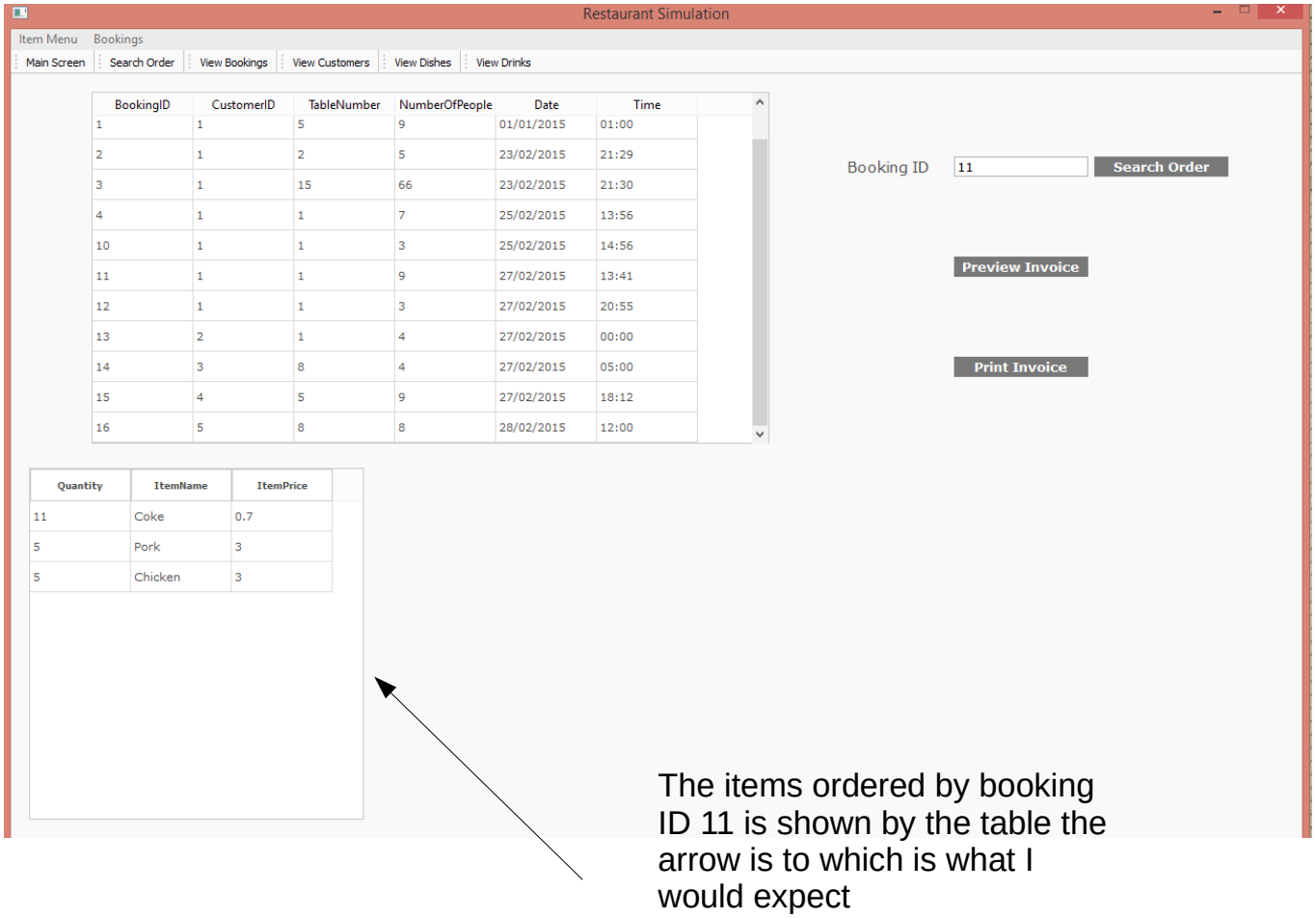


Figure 3.9: Checking the search function

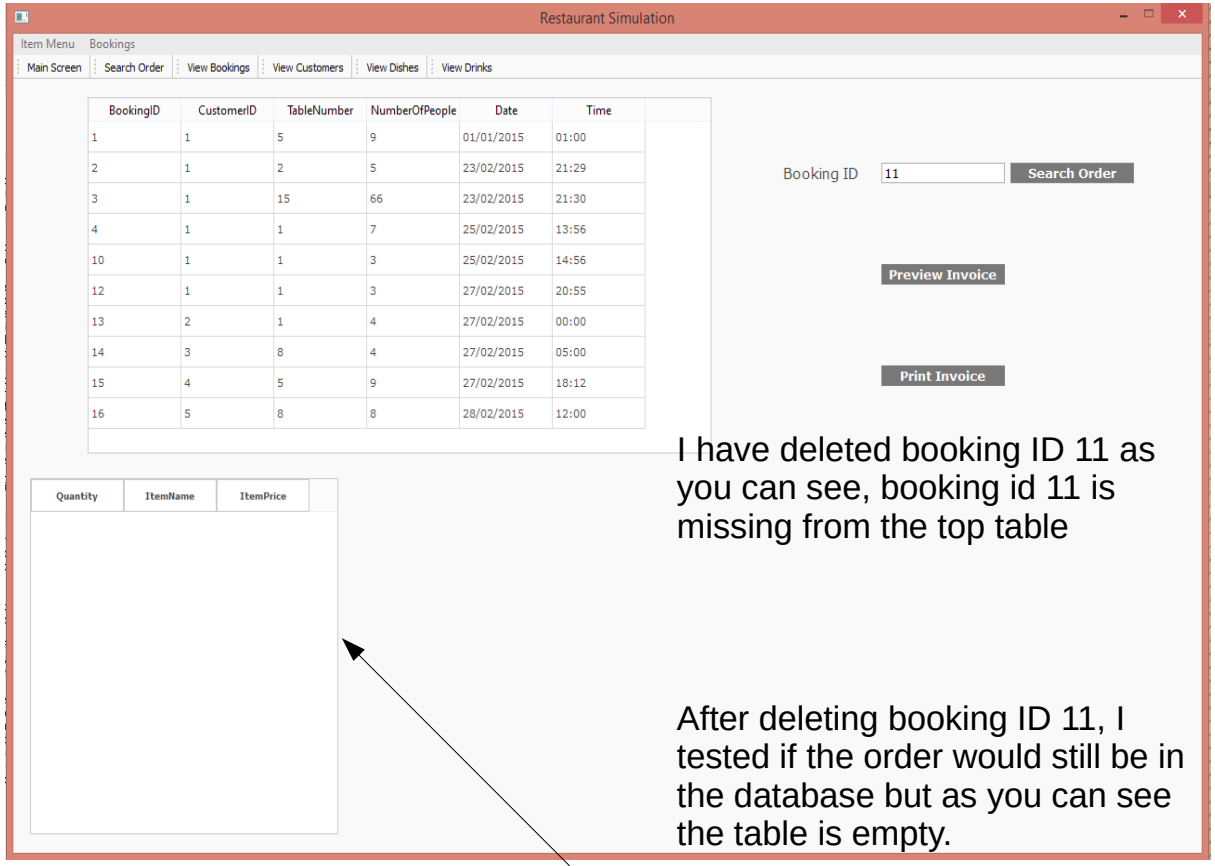


Figure 3.10: Checking if the order is deleted after deleting booking ID 11

3.4 Evaluation

3.4.1 Approach to Testing

I made sure I tested my program thoroughly by going through different types of testing strategies. Going through different testing strategies made me cover most, if not all areas of my program such as testing the flow of control, validation, algorithms and the outputs. I chose this approach to ensure my whether my program was usable or not.

3.4.2 Problems Encountered

I encountered a problem on test 4.03. The total price did not refresh after adding an item, it would only refresh after closing the manage order box and selecting the table again.However, the algorithm to calculate the total price was correct

3.4.3 Strengths of Testing

The strengths of the approach I took were that I was able to find out whether my program was usable or not. Users of my system will undoubtly make mistakes when inputting information and so I have tested different test data to ensure the system was still usable. In addition, checking the flow of control proved the navigation of the system to be effective.

3.4.4 Weaknesses of Testing

When I tested the sql statements, I relied on the tables displayed on my application to tell me whether the sql statements worked without problems. For example, I deleted a booking on test 6.03 and checked to see if there was an order attached to the booking. The order did not display after searching for the booking but it may have still been in the database - which is what I didn't check. Also, the applcation has 16 radio buttons that represent the tables in the restaurant - I did not test all of them which would of meant that there could have been a fault with the application within the radio buttons that I did not test.

3.4.5 Reliability of Application

I believe that the application is reliable. I have tested the input validation which worked as the system did not proceed if there was an invalid input which would mean there couldn't be any faulty data in the database. However, it would be

up to the User to input the correct data such as correctly spelling an item when adding an item to the menu or inputting the right ID when deleting a record.

3.4.6 Robustness of Application

After testing my system without closing it, I believe that application is robust as I did not experience any crahses or any problems that made me not be able use the application normally. Testing the validation did not affect the application in anyway, same goes for the execution of the sql queries.

Chapter 4

System Maintenance

4.1 Environment

4.1.1 Software

- Python 3
- IDLE
- PyQt
- SQLite 3
- SQLite Database Browser

4.1.2 Usage Explanation

The table below includes all the listed software from the previous section with the explanation as to why I used them. The listed software can be downloaded for free which made the creation of my system an easier approach. Also, this would mean that my client wouldn't need to purchase anything.

Software	**Usage Explanation**
Python 3	I used Python because it was the only programming language I was familiar with as I learnt how to use it during my time at sixth form.
IDLE	I used IDLE to write the Python scripts and since I am the most familiar with this software , it made implementation of the client application easier.
PyQt	PyQt included everything needed to create the graphical user interfance for my system and there was a lot of information on PyQt accessible which helped me create the graphical interface more effectively.
SQLite3	Although this came with python, I have used this to create the database and manipulate the database as it was easy for me to understand on how to do so and was very effective in doing so.
SQLite Database Browser	This software helped me make the system's code relating the database a much more easier process because it allowed me to observe whether the database queries excutely correctly or not.

4.1.3 Features Used

Software	Features Used
Python 3	I took advantage of the ability to import modules to structure my code clearer.
IDLE	There are countless number of features that IDLE has to offer which helped me create the application but I will mention a few of them. The syntax highlighter made it easier to understand the code that I was writing which is important when a system is complex(helps track what you are doing). Also it prevented me from making more errors. For example, I was able to spot out straight away if i mispelt a keywork such as print or while, etc. The 'Go to File/Line' feature when an error occurs, helped me tremendously as it was the main factor of helping me debug my program. Being able to run the system allowed me to test the system
PyQt	PyQt has many features that allowed me to create graphical user interface (GUI) for my system. The core components that I used to create the GUI were main windows, dialog boxes and widgets.
SQLite3	I was able to create the database for the system through SQLite3. I used most of core features that was available to me through SQLite3 such as being able to ADD, DELETE,UPDATE to/from the database. Enforcing referential integrity was useful as it helped the database to become consistent.
SQLite Database Browser	I mainly used 'Browse Data' to check whether I have added, deleted or updated a record successfully. I also used 'Execute SQL' for the SELECT statements in the apllication as it allowed me to see whether the SQL query was correct or not.

4.2 System Overview

4.2.1 System Component

Graphical User Interface (GUI)

Having a graphical user interface for the system makes it a lot more userable, giving the user a much more user friendly experience. Including a GUI makes it easier for the user to navigate around the system.

Manage Item Menu

The item menu can be managed at any time through the menu bar 'Item Menu'. The user can add/delete/update an item.

To add an item to the menu, the user must select 'Add Item'. By selecting 'Add Item', the user will be presented with a layout that consists of a table widget displaying all the records of the menu and the fields which will be used to input information for the new item.

Deleting an item off the menu can also be found under the 'Item Menu' menu by selecting 'Delete Item'. The user will be presented with a layout that contains the same table widget that displays all records of the menu and has either the choose to delete an item by inputting the item name or the item ID.

The user also has the option to update an item's price. To do this, the user must select 'Update Item Price' where the user will be presented with a layout that contains the same the item menu table widget as the add and delete item layout. The user would have to input the ID of the item and the new price then click on the 'Update Item' button to update an item's price.

Manage Bookings

The user will be able to add/delete and update bookings. To do this, the user could either selection these options through "Bookings" on the menu bar or click on the 'Manage Bookings' button at the bottom of the main screen.

Clicking on the button will switch the central widget to the manage bookings widget where the user will be presented with the Bookings table widget where all of the booking records will be displayed and below the widget are the buttons "Add Booking" and "Delete Booking". Clicking on Add Booking will then present the user with the same table widget and the required fields which the user would have to successfully fill to add a booking. As for the "Delete Booking" button, the user will be presented with the same Bookings table widget and a input field for the user to delete a booking by inputting a booking id and pressing "Delete BookingID".

The "Bookings" menu bar also has 3 options; "Add Bookings", "Delete Booking" and "Update Booking". The add/delete booking options are the same as described in the paragraph above. As for the "Update Booking" option, the user will be presented with the usual Bookings table widget and the input fields to update the booking.

Manage Sit-In Orders

To manage an order, the user must select the table and if not already, assign a customer to the table (A dialog box will pop up telling the user to assign a customer to the selected table). After assigning a customer to the table, the table will be known as 'occupied' which would allow the user to select that table without assigning a customer to that table everytime. So now that the table is occupied, there will be a manage order box where the booking details will be displayed on a row at the top of the box. The dishes and drinks ordered will be split into two table widget, the dishes ordered will be displayed on the left and the drinks on the right.

The user has all the neccessary options on the manage order box such as "Add", "Delete", "Finish", "Invoice Preview" and "Print Invoice". The "Add" button is for adding items to the order, the "Delete" button will be used to delete items off the order, the "Invoice Preview" will show the user what the invoice would look like for the order, the "Print Invoice" will print the invoice and the "Finish" button will set the status of the table as unoccupied, clearing the booking details for that table and so the user would have to assign a customer to that table when selecting the table from the main screen.

4.3 Code Structure

4.3.1 Particular Code Section

4.4 Variable Listing

Variable Name	**Purpose**	**Line number in code**	**Section**
regExp	Holds the regular expression to only allow the user to enter letters for name inputs	31	
regexp	Holds the regular expression to only allow the user to input digits	42	
regexp2	Holds the regular expression to only allow the user to input digits	48	
each	Stepper variable used in the for loop	57, 58	
self.maximumdate	Used to set maximum date for QDateEdit	63	

self.minimumdate	Used to set minimum date for QDateEdit	64	
FirstName	Used to store user input	106, 119 ,121	
LastName	Used to store user input	107, 119, 121	
TeleNumber	Used to store user input	108, 119, 121	
NumberOfPeople	Used to store user input	110, 119, 135	
TableNumber	Holds the index of the table number combo box	113, 129, 135	
BookingDate	Used to store user input	114, 135	
BookingTime	Used to store user input	115, 135	
customer	Holds the variables FirstName,LastName and TeleNumber to create a record for Customers	121	
db	Stores the path of the database	123,129,137	
booking	Holds the variables customerid, TableNumber, NumberOfPeople, BookingDate and BookingTime to create a record for Bookings	135	
regexpp	Holds the regular expression to only allow a maximum of 20 letters	35	
ItemName	Used to store user input	72	
ItemPrice	Used to store user input	73	
ItemType	Holds the selected index of the item type combo box	73,74,76,78,79	
MenuItem	Holds the variables ItemName, ItemPrice and ItemType to create a record for Items	79	
self.bookingDetails	Holds booking details such as bookingid, customerid, booking date, booking time, table number and number of people	16	
bookingID	Holds booking id	57	
self.ItemID	Used to store user input	58	
Quantity	Used to store user input	59	
addedAlready	Holds a boolean value to indicate whether an item has been added to the order already (Used to increase quantity)	61, 66, 83, 97, 128, 130	

newQuantity	Used to calculate the new quantity after adding an item that's already been added	72	
updateOrder	Holds the new quantity and item id to update the record	73	
itemsOrdered	Holds an array of ordered items for a particular booking	98, 111	

4.5 System Evidence

4.5.1 User Interface

4.5.2 ER Diagram

4.5.3 Database Table Views

4.5.4 Database SQL

4.5.5 SQL Queries

4.6 Testing

4.6.1 Summary of Results

4.6.2 Known Issues

4.7 Code Explanations

4.7.1 Difficult Sections

4.7.2 Self-created Algorithms

4.8 Settings

4.9 Acknowledgements

4.10 Code Listing

4.10.1 Module 1

Chapter 5

User Manual

5.1 Introduction

5.2 Installation

5.2.1 Prerequisite Installation

Installing Python

Installing PyQt

Etc.

5.2.2 System Installation

5.2.3 Running the System

5.3 Tutorial

5.3.1 Introduction

5.3.2 Assumptions

5.3.3 Tutorial Questions

Question 1

Question 2

5.3.4 Saving

5.3.5 Limitations

5.4 Error Recovery

Chapter 6

Evaluation

6.1 Customer Requirements

6.1.1 Objective Evaluation

6.2 Effectiveness

6.2.1 Objective Evaluation

6.3 Learnability

6.4 Usability

6.5 Maintainability

6.6 Suggestions for Improvement

6.7 End User Evidence

6.7.1 Questionnaires

6.7.2 Graphs

6.7.3 Written Statements