

COMP 4

Tommy Tham

January 16, 2015

Contents

1	Analysis	5
1.1	Introduction	5
1.1.1	Client Identification	5
1.1.2	Define the current system	5
1.1.3	Describe the problems	6
1.1.4	Section appendix	6
1.2	Investigation	7
1.2.1	The current system	7
1.2.2	The proposed system	16
1.3	Objectives	20
1.3.1	General Objectives	20
1.3.2	Specific Objectives	21
1.3.3	Core Objectives	22
1.3.4	Other Objectives	22
1.4	ER Diagrams and Descriptions	23
1.4.1	ER Diagram	23
1.4.2	Entity Descriptions	24
1.5	Object Analysis	24
1.5.1	Object Listing	24
1.5.2	Relationship diagrams	25
1.5.3	Class definitions	26
1.6	Other Abstractions and Graphs	27
1.7	Constraints	27
1.7.1	Hardware	27
1.7.2	Software	27
1.7.3	Time	27
1.7.4	User Knowledge	27
1.7.5	Access restrictions	28
1.8	Limitations	28
1.8.1	Areas which will not be included in computerisation	28
1.8.2	Areas considered for future computerisation	28
1.9	Solutions	29
1.9.1	Alternative solutions	29

1.9.2	Justification of chosen solution	29
2	Design	30
2.1	Overall System Design	30
2.1.1	Short description of the main parts of the system	30
2.1.2	System flowcharts showing an overview of the complete system	34
2.2	User Interface Designs	41
2.3	Hardware Specification	51
2.4	Program Structure	53
2.4.1	Top-down design structure charts	53
2.4.2	Algorithms in pseudo-code for each data transformation process	56
2.4.3	Object Diagrams	59
2.4.4	Class Definitions	61
2.5	Prototyping	62
2.6	Definition of Data Requirements	62
2.6.1	Identification of all data input items	62
2.6.2	Identification of all data output items	63
2.6.3	Explanation of how data output items are generated	65
2.6.4	Data Dictionary	67
2.6.5	Identification of appropriate storage media	68
2.7	Database Design	70
2.7.1	Normalisation	70
2.7.2	SQL Queries	73
2.8	Security and Integrity of the System and Data	74
2.8.1	Security and Integrity of Data	74
2.8.2	System Security	75
2.9	Validation	76
2.10	Testing	76
2.10.1	Outline Plan	77
2.10.2	Detailed Plan	77
3	Testing	87
3.1	Test Plan	87
3.1.1	Original Outline Plan	88
3.1.2	Changes to Outline Plan	88
3.1.3	Original Detailed Plan	88
3.1.4	Changes to Detailed Plan	88
3.2	Test Data	89
3.2.1	Original Test Data	89
3.2.2	Changes to Test Data	89
3.3	Annotated Samples	89
3.3.1	Actual Results	89
3.3.2	Evidence	89
3.4	Evaluation	90

3.4.1	Approach to Testing	90
3.4.2	Problems Encountered	90
3.4.3	Strengths of Testing	90
3.4.4	Weaknesses of Testing	90
3.4.5	Reliability of Application	90
3.4.6	Robustness of Application	90
4	System Maintenance	91
4.1	Environment	92
4.1.1	Software	92
4.1.2	Usage Explanation	92
4.1.3	Features Used	92
4.2	System Overview	92
4.2.1	System Component	92
4.3	Code Structure	92
4.3.1	Particular Code Section	92
4.4	Variable Listing	92
4.5	System Evidence	92
4.5.1	User Interface	92
4.5.2	ER Diagram	92
4.5.3	Database Table Views	92
4.5.4	Database SQL	92
4.5.5	SQL Queries	92
4.6	Testing	92
4.6.1	Summary of Results	92
4.6.2	Known Issues	92
4.7	Code Explanations	92
4.7.1	Difficult Sections	92
4.7.2	Self-created Algorithms	92
4.8	Settings	92
4.9	Acknowledgements	92
4.10	Code Listing	92
4.10.1	Module 1	93
5	User Manual	94
5.1	Introduction	95
5.2	Installation	95
5.2.1	Prerequisite Installation	95
5.2.2	System Installation	95
5.2.3	Running the System	95
5.3	Tutorial	95
5.3.1	Introduction	95
5.3.2	Assumptions	95
5.3.3	Tutorial Questions	95
5.3.4	Saving	95
5.3.5	Limitations	95

5.4	Error Recovery	95
5.4.1	Error 1	95
5.4.2	Error 2	95
5.5	System Recovery	95
5.5.1	Backing-up Data	95
5.5.2	Restoring Data	95
6	Evaluation	96
6.1	Customer Requirements	97
6.1.1	Objective Evaluation	97
6.2	Effectiveness	97
6.2.1	Objective Evaluation	97
6.3	Learnability	97
6.4	Usability	97
6.5	Maintainability	97
6.6	Suggestions for Improvement	97
6.7	End User Evidence	97
6.7.1	Questionnaires	97
6.7.2	Graphs	97
6.7.3	Written Statements	97

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 separate 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 separate 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 isn't 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  $\leftarrow$  false
2:
3: WHILE notOrderTaken
4:   IF Customer ready to order THEN
5:     Order  $\leftarrow$  USERINPUT
6:     OrderTaken  $\leftarrow$  true
7:   ELSE
8:     Wait
9:   ENDIF
10: ENDWHILE
```

Algorithm 2 Generating invoice

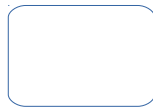
```
1: InvoiceGenerated  $\leftarrow$  false
2:
3: WHILE notInvoiceGenerated
4:   IF Customer has finished ordering THEN
5:     Copy order details from order pad onto invoice pad
6:     Get prices of each dish and drink ordered from menu
7:     Copy prices onto invoice pad
8:     Calculate total price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

Algorithm 3 Payment

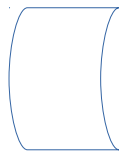
```
1: Payment  $\leftarrow$  false
2:
3: WHILE notPayment
4:   IF Customer ask for bill THEN
5:     Give invoice
6:     Payment  $\leftarrow$  USERINPUT
7:     Payment  $\leftarrow$  true
8:   ELSE
9:     Wait
10:  ENDIF
11: ENDWHILE
```

Data flow diagramKey

Data source/destination



Process



Data store

Figure 1.1: Data flow key

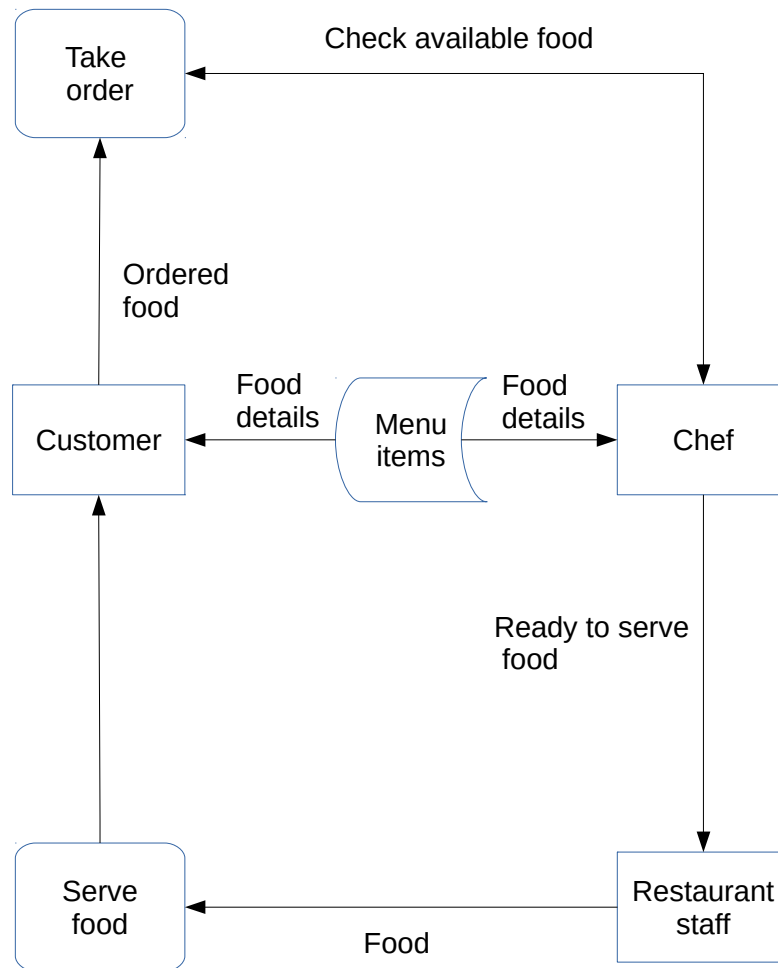


Figure 1.2: Data flow diagram of placing an order

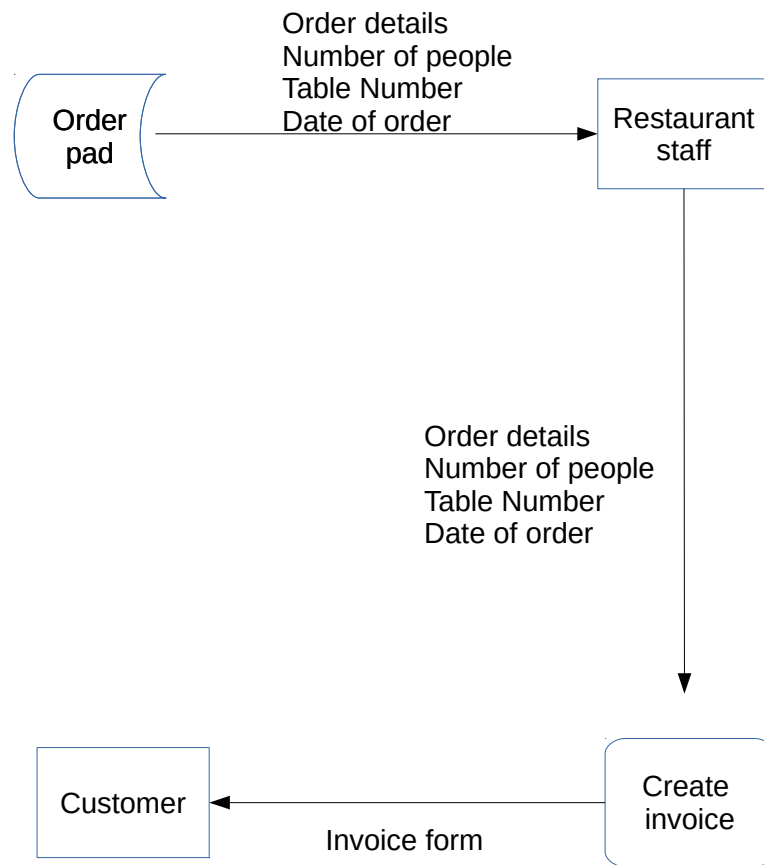


Figure 1.3: Data flow diagram of generating an invoice

Input Forms, Output Forms, Report Formats

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

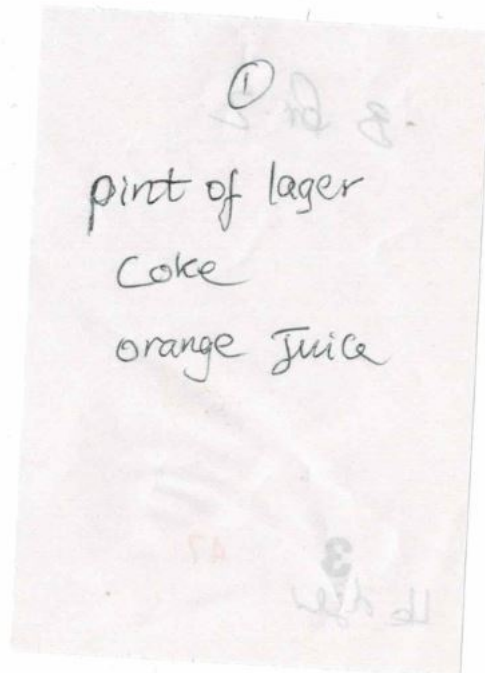


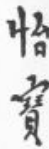
Figure 1.4: Writing down drinks ordered on the drink pad

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.

TABLE No. ①	COVERS 4
spare Ribs	
Butterfly prawn	
Sesame prawn	
$\frac{1}{4}$ Duck	
kung po chicken	
chicken chowmen	
Egg Ric	
chilli beef	
Sweet sour Pork	
DATE 30/9/14	INITIALS
2	27

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.



Links
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
	1/2 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. ①

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.

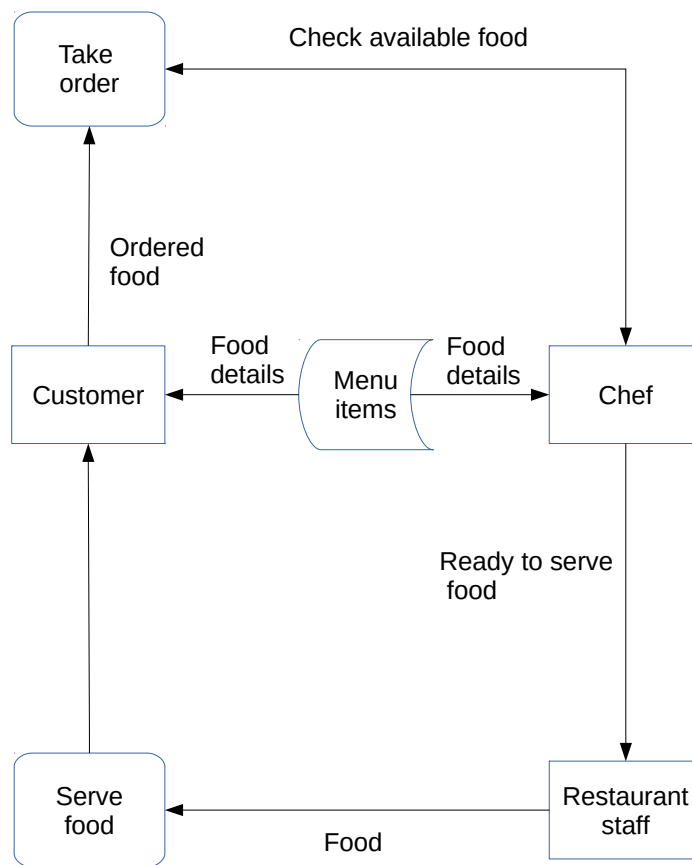


Figure 1.7: A data flow diagram of the proposed system - placing and processing the order

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 inputted data saved in a database once the customer has finished with their meal. Also, invoices will be created through this application.

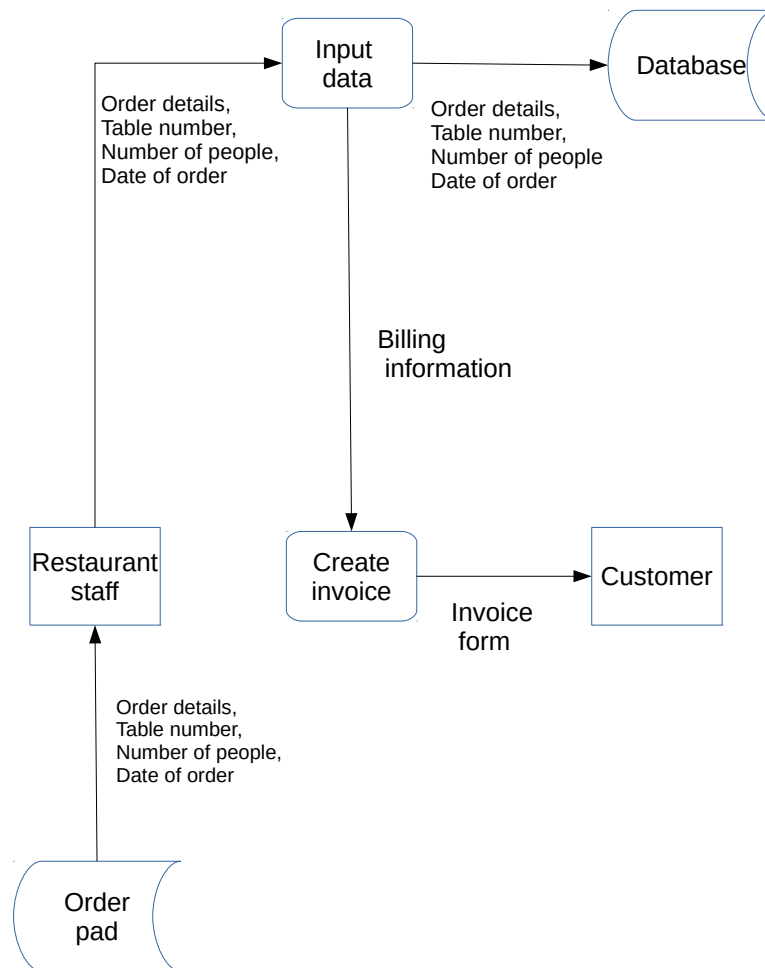


Figure 1.8: Data flow diagram proposed system

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 $1050 \times 6 = 6300$ 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

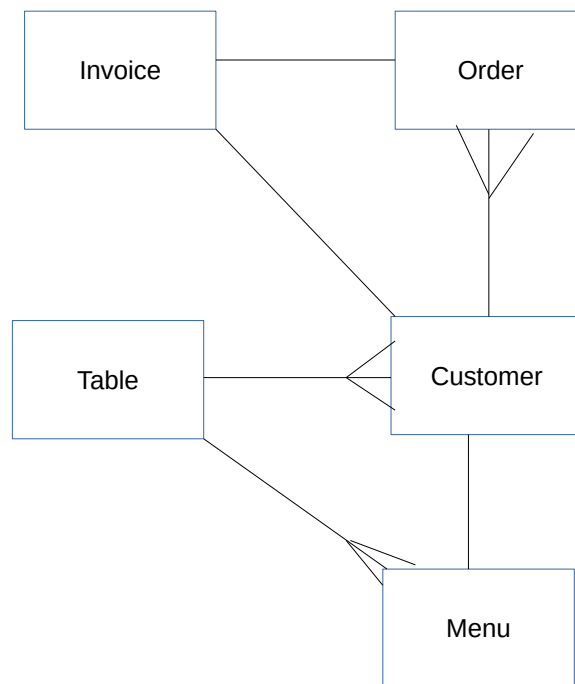


Figure 1.9: E-R Diagram

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

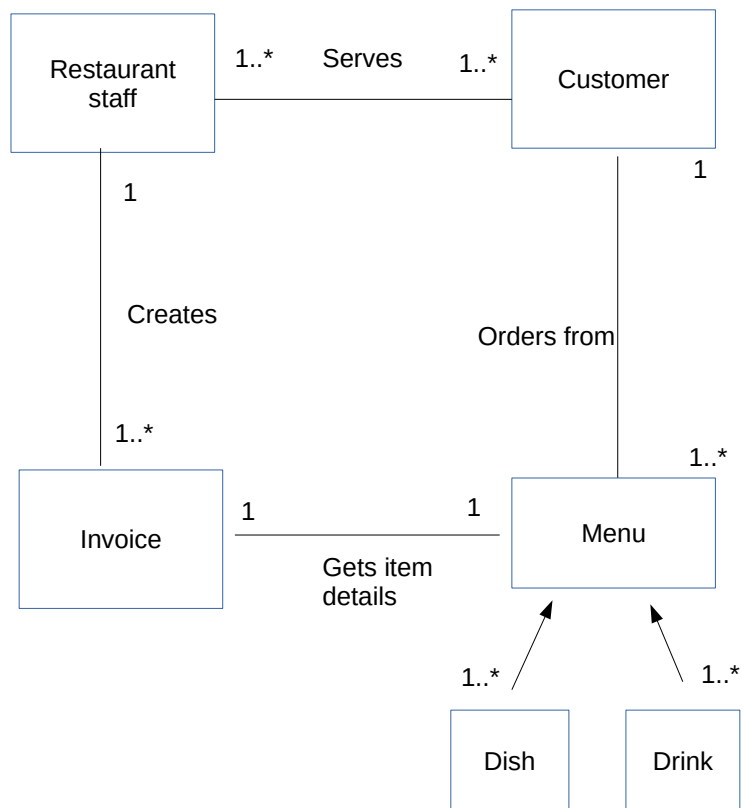


Figure 1.10: Relationship diagram

1.5.3 Class definitions

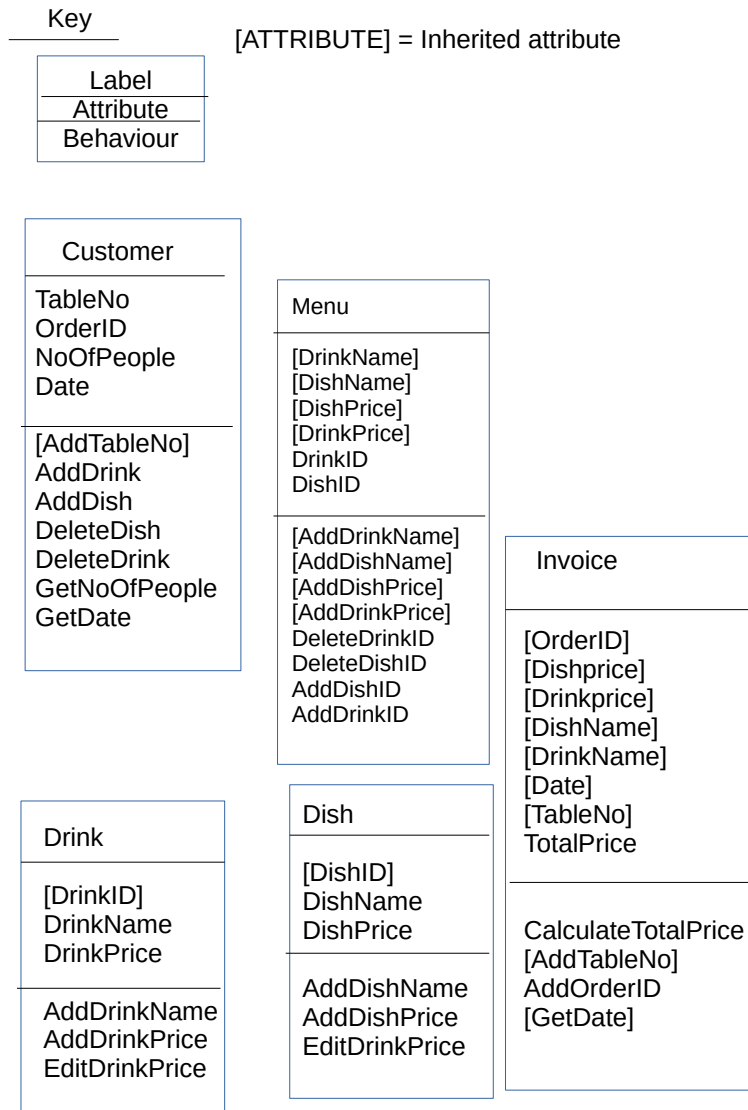


Figure 1.11: Class diagram

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 constraint. 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 irrelevant 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.
- Clicking the 'Delete' button again will cancel this feature and the red boxes will disappear.

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

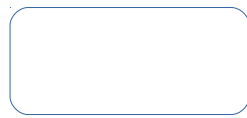
- A button labelled "Bookings" will be at the main screen. A box will appear that will be used to manage bookings.
- Adding and deleting bookings will be available through this box that is used to manage bookings.

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 requires 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

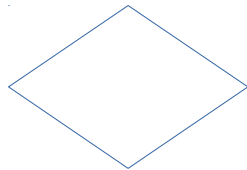
Key



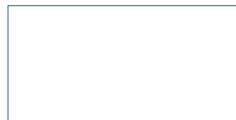
Start/Stop



Input/Output



Decision



Process



Storage

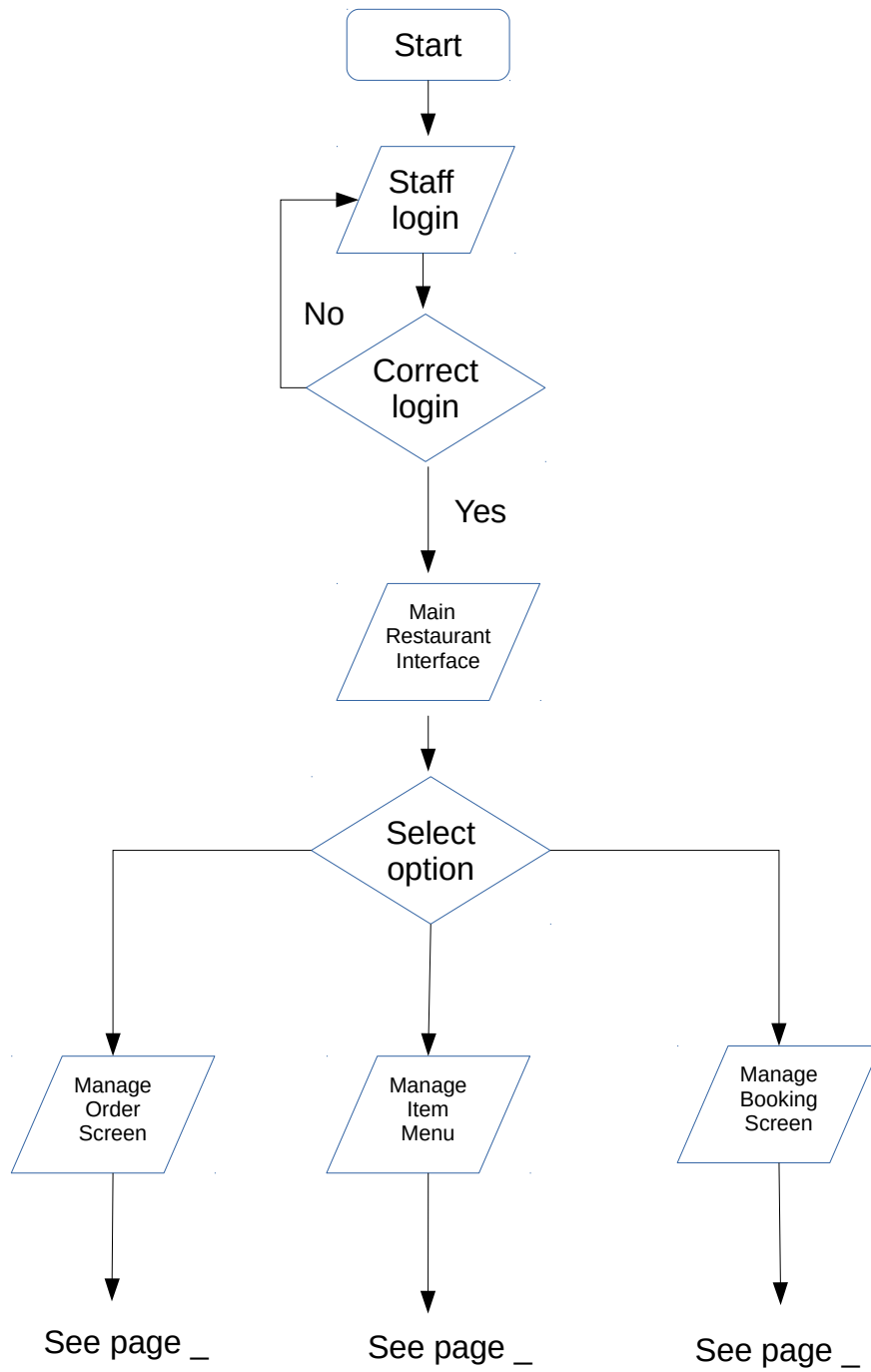


Figure 2.2: Flow chart of system

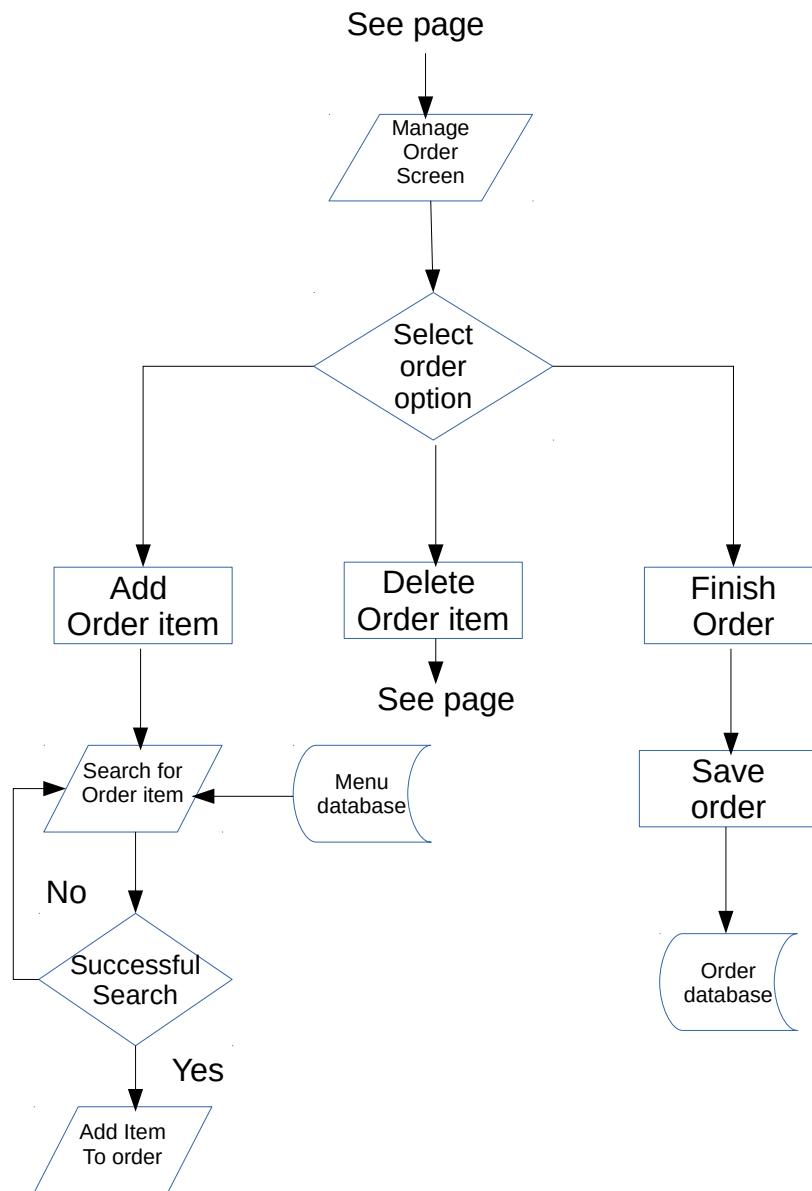


Figure 2.3: Flow chart of order

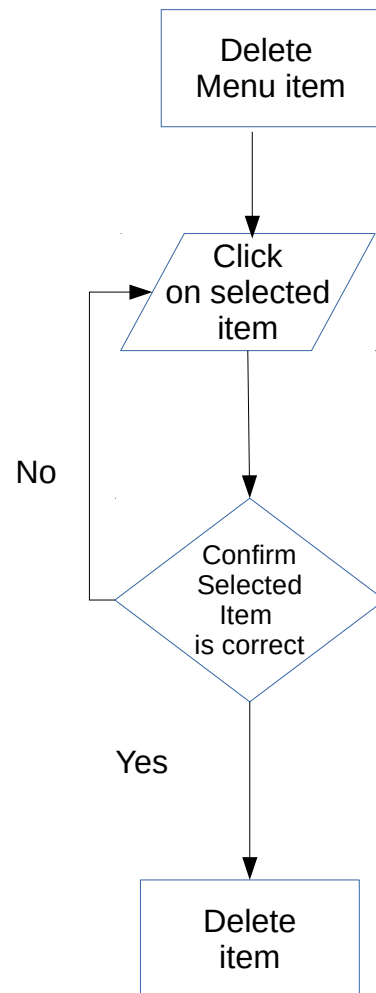


Figure 2.4: Flow chart of deleting an item of an order

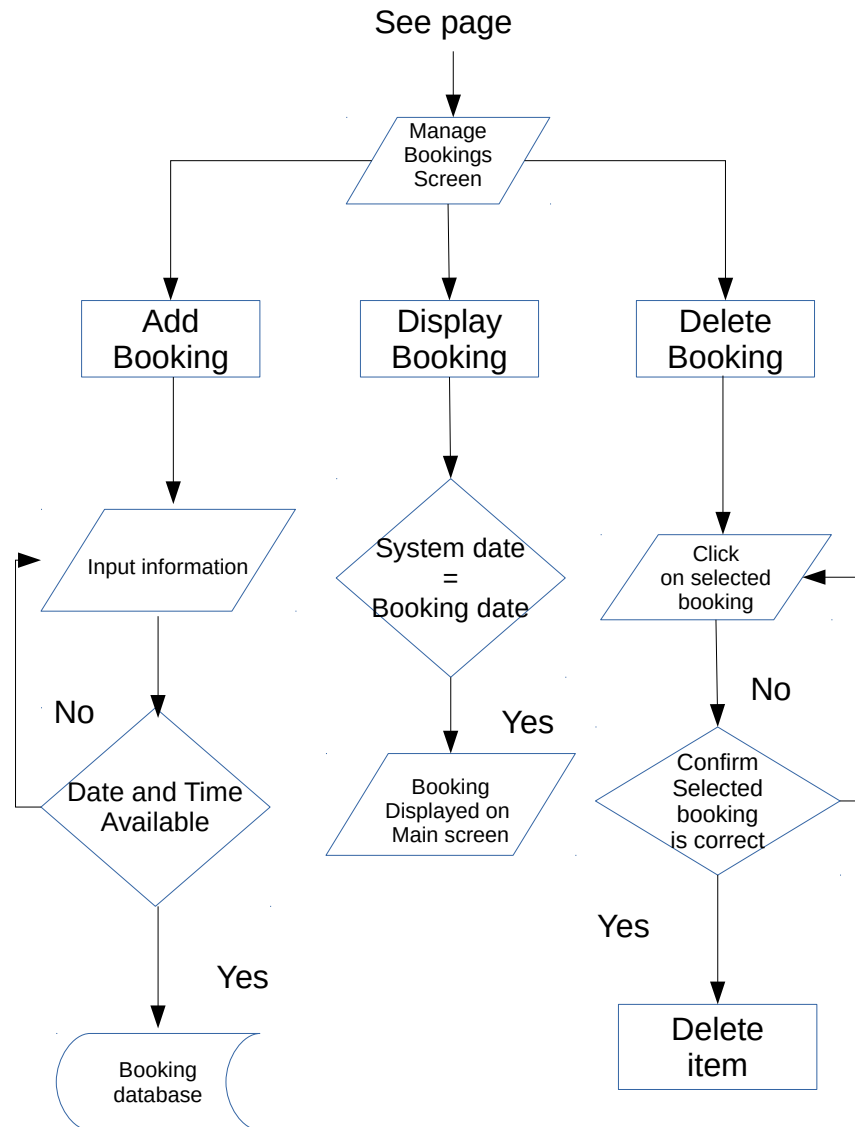


Figure 2.5: Flow chart of bookings

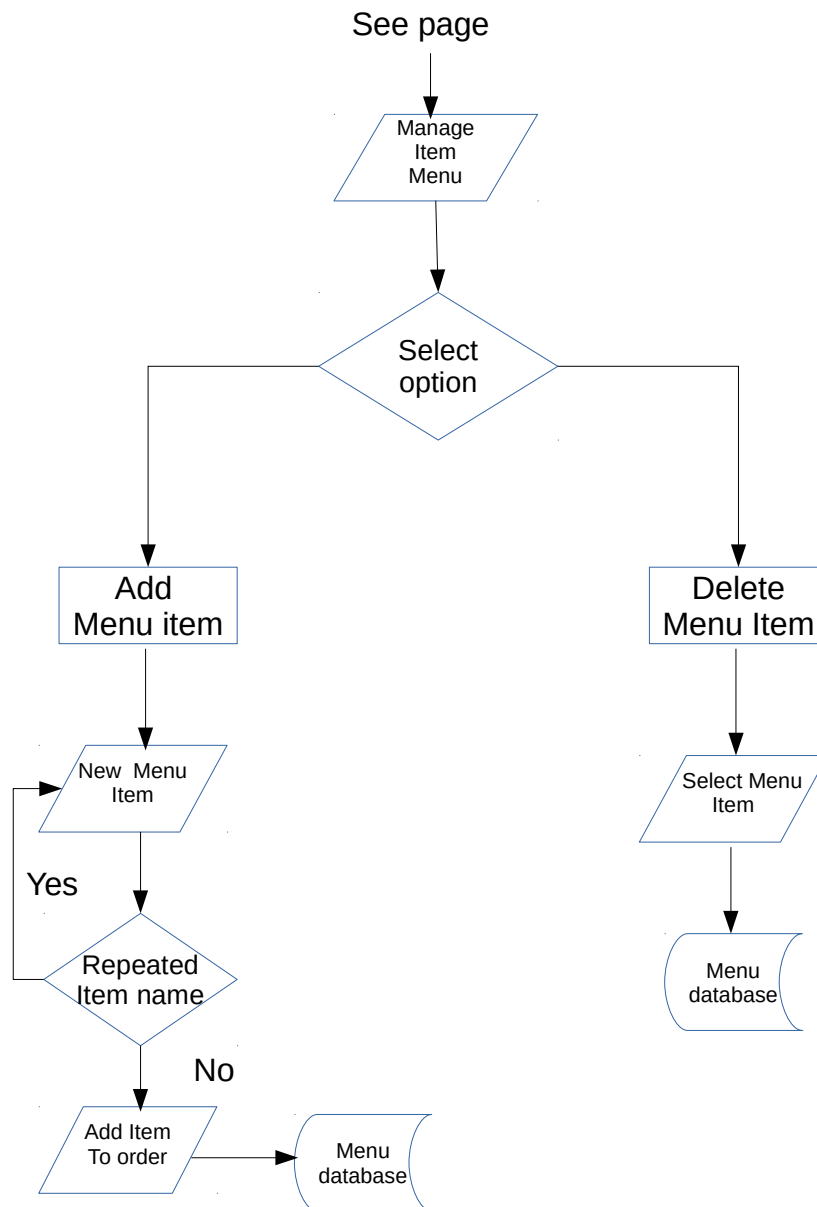


Figure 2.6: Flow chart of adding an item to the menu

2.2 User Interface Designs



Restaurant Simulator

Please enter the password

This box asking for the password will pop up once the program is opened. The purpose of this is to only allow staff to access this program.

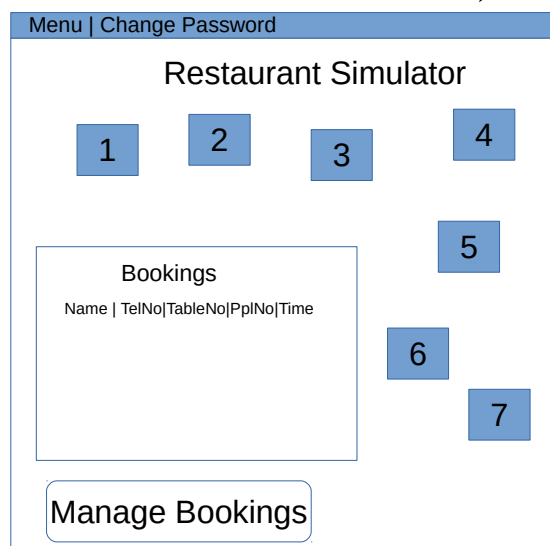


Restaurant Simulator

Please enter the password

If the user enters the wrong password then it will inform the user at the bottom left of the box

The password prompt box will disappear once the correct password has been entered. The main program will continue to run.



Menu | Change Password

Restaurant Simulator

1 2 3 4

5

6

7

Bookings

Name | TelNo|TableNo|PplNo|Time

Manage Bookings

Figure 2.7: Password Prompt

The diagram shows a horizontal menu bar with two items: 'Menu' and 'Change Password'. Below the 'Menu' item, a dropdown menu is displayed, containing two options: 'Add Item' and 'Delete Item'. An arrow points from the 'Add Item' option to the 'Add item to menu' dialog box.

This is the menu bar that will be on the window.

Clicking on Menu will bring down a drop down box containing 'Add Item' and 'Delete Item'.

This is the box that will pop up once clicked on 'Add Item'.

The user will input information and once completed, the item will be added to the database.

Clicking Cancel will close the box.

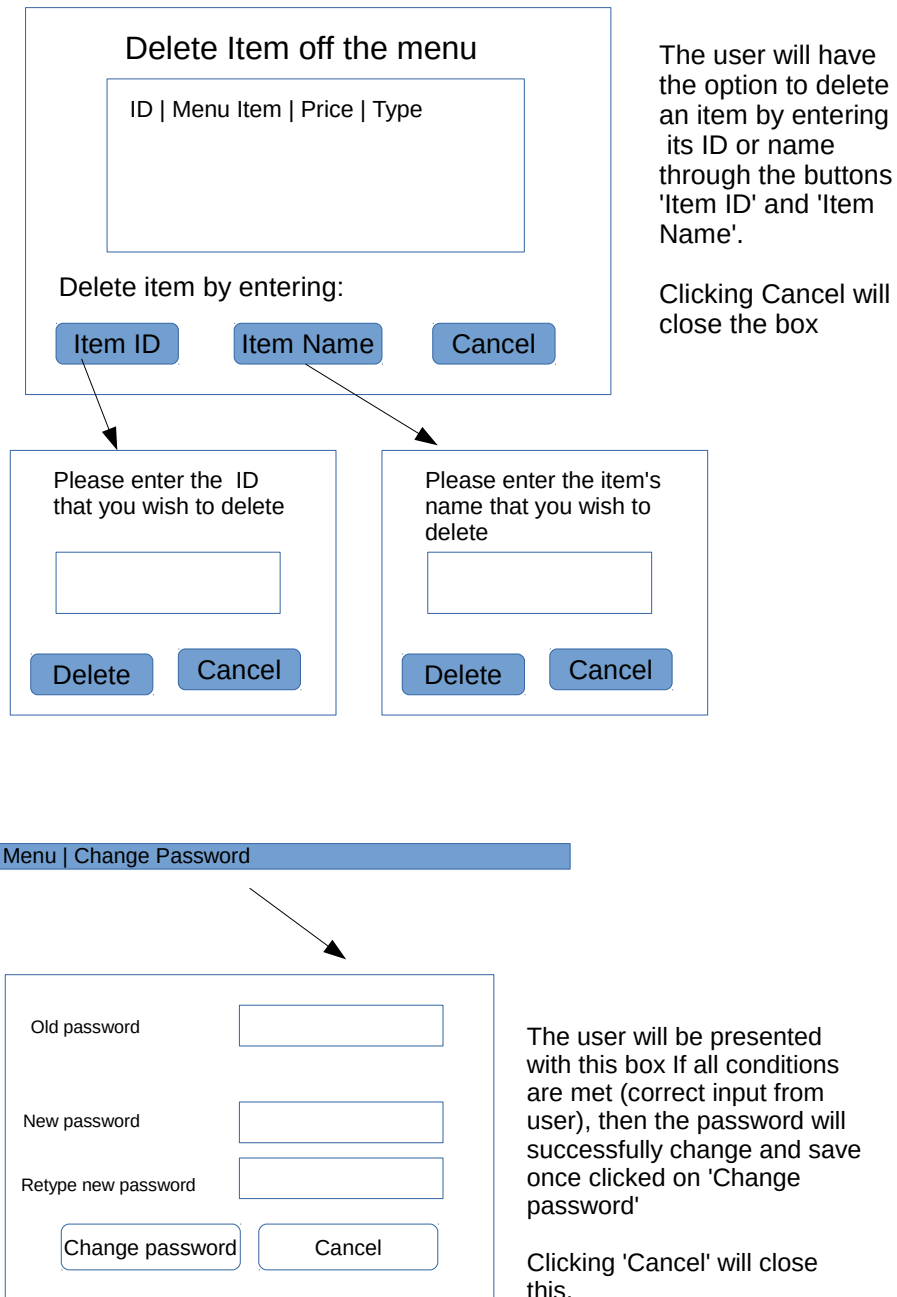
The dialog box is titled 'Add item to menu'. It contains three input fields: 'Item Name', 'Item Price', and 'Item Type'. At the bottom, there are two buttons: 'Add Item' and 'Cancel'.

The diagram shows a horizontal menu bar with two items: 'Menu' and 'Change Password'. Below the 'Menu' item, a dropdown menu is displayed, containing two options: 'Add Item' and 'Delete Item'. An arrow points from the 'Delete Item' option to the 'Delete Item off the menu' dialog box.

Clicking on 'Delete Item' will bring up this window. The menu database will be displayed for the user to see which item(s) they would like to delete.

The dialog box is titled 'Delete Item off the menu'. It contains a table with the following header: 'ID | Menu Item | Price | Type'. Below the table, it says 'Delete item by entering:'. At the bottom, there are three buttons: 'Item ID', 'Item Name', and 'Cancel'.

Carried on over the next page



Restaurant Simulator

Menu | Change Password

1 2 3 4

5

6 7

Bookings

Name | TelNo | TableNo | PplNo | Time

Manage Bookings

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

Table 6

Number of people: 2

Time of arrival: 19:32

Date of arrival: 04/06/13

Create Cancel

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.

Menu | Change Password

← **Table 6**

Number of people: 2 Date: 04/06/13
Time of Arrival: 19:32

Qty	Dish	Price	Qty	Drink	Price
Total Dish Price			Total Drink Price		
				Total Price	

Add Delete Finish

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

Menu | Change Password

Table 7

Number of people: 6

Date: 17/05/12

Time of Arrival: 17:05

Qty	Dish	Price	Qty	Drink	Price
Total Dish Price			Total Drink Price		
				Total Price	

Add

Delete

Finish

Clicking on Add will display a window which shows the menu database so the user can see what items are on the menu.

Add Item to order

ID	Menu Item	Price	Type

Add item by entering:

The user will have the option to add an item by entering its ID or name through the buttons 'Item ID' and 'Item Name'.

Clicking Cancel will
close the box

Please enter the ID
that you wish to add

Add

Cancel

Please enter the item's name that you wish to add

Add

Cancel

Figure 2.12: Add Item

Add Item to order

ID | Menu Item | Price | Type

Add item by entering:

Item ID

Item Name

Cancel

Please enter the ID that you wish to add

14

Add

Cancel

Please enter the item's name that you wish to add

Coke

Add

Cancel

Menu | Change Password

Table 7

Number of people: 6 Date: 17/05/12
Time of Arrival: 17:05

Qty	Dish	Price	Qty	Drink	Price
1	Spare ribs	4.20	1	Coke	0.70
Total Dish Price		4.20	Total Drink Price		0.70
				Total Price	

Add

Delete

Finish

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

Menu | Change Password

Table 7

Number of people: 6 Date: 17/05/12
Time of Arrival: 17:05

Qty	Dish	Price	Qty	Drink	Price
1	Spare ribs	4.20			
Total Dish Price		4.20	Total Drink Price		
				Total Price	

Add Delete Finish

Menu | Change Password

Table 7

Number of people: 6 Date: 17/05/12
Time of Arrival: 17:05

Qty	Dish	Price	Qty	Drink	Price
1	Spare ribs	4.20			
Total Dish Price		4.20	Total Drink Price		
				Total Price	

Add Delete Finish

Are you sure you want to delete Spare ribs off the order?

Yes No

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 accidentally 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

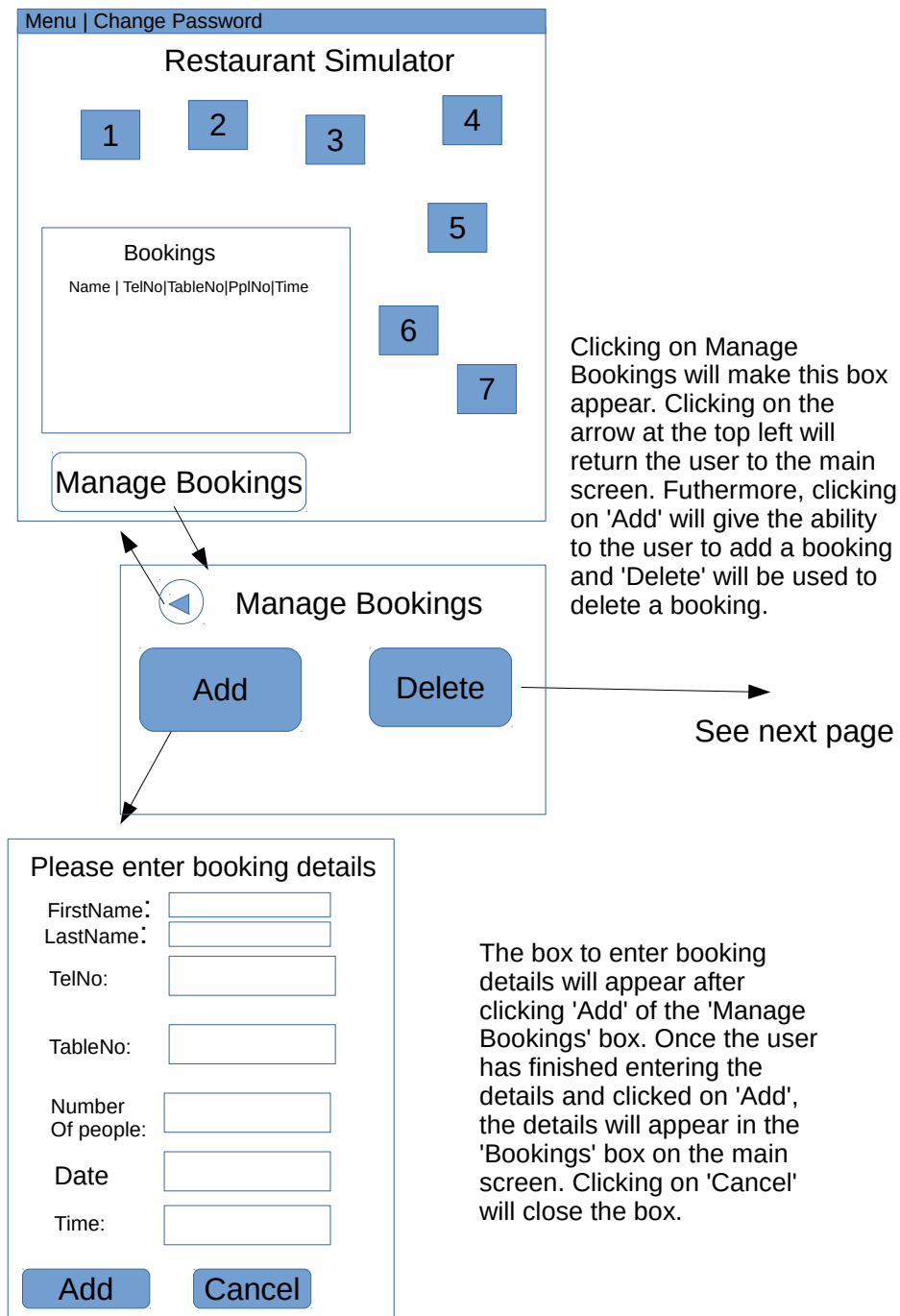


Figure 2.15: Add Booking



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.

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

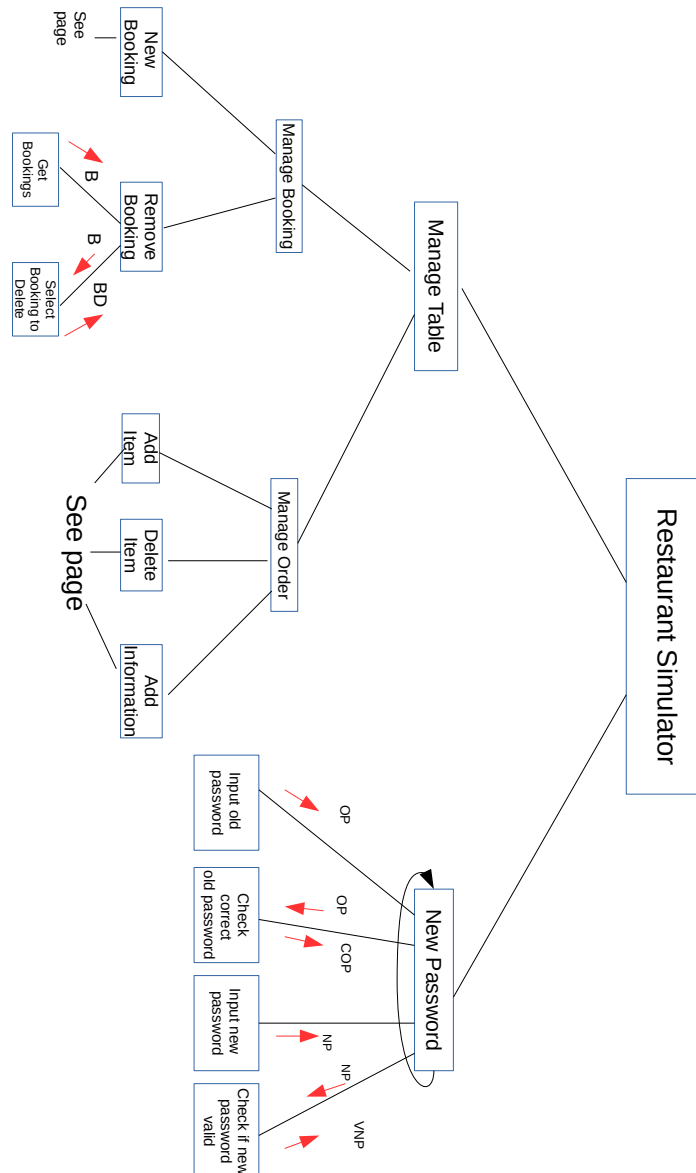


Figure 2.17: Main structure

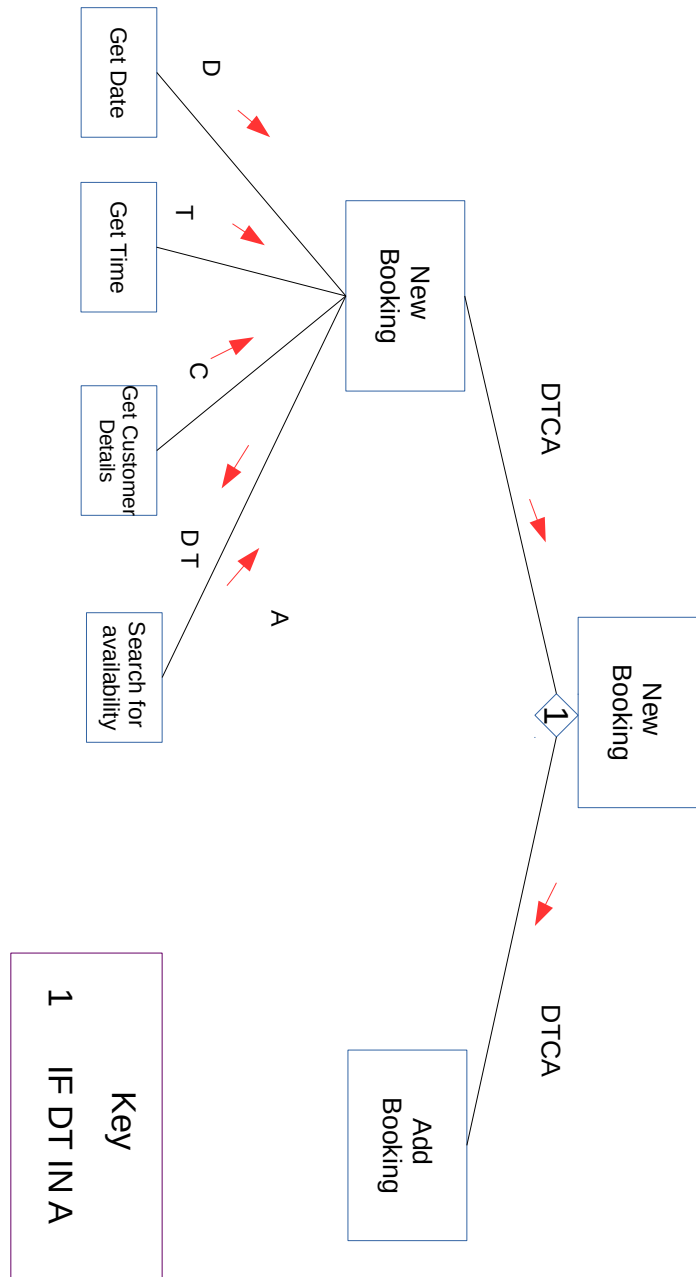
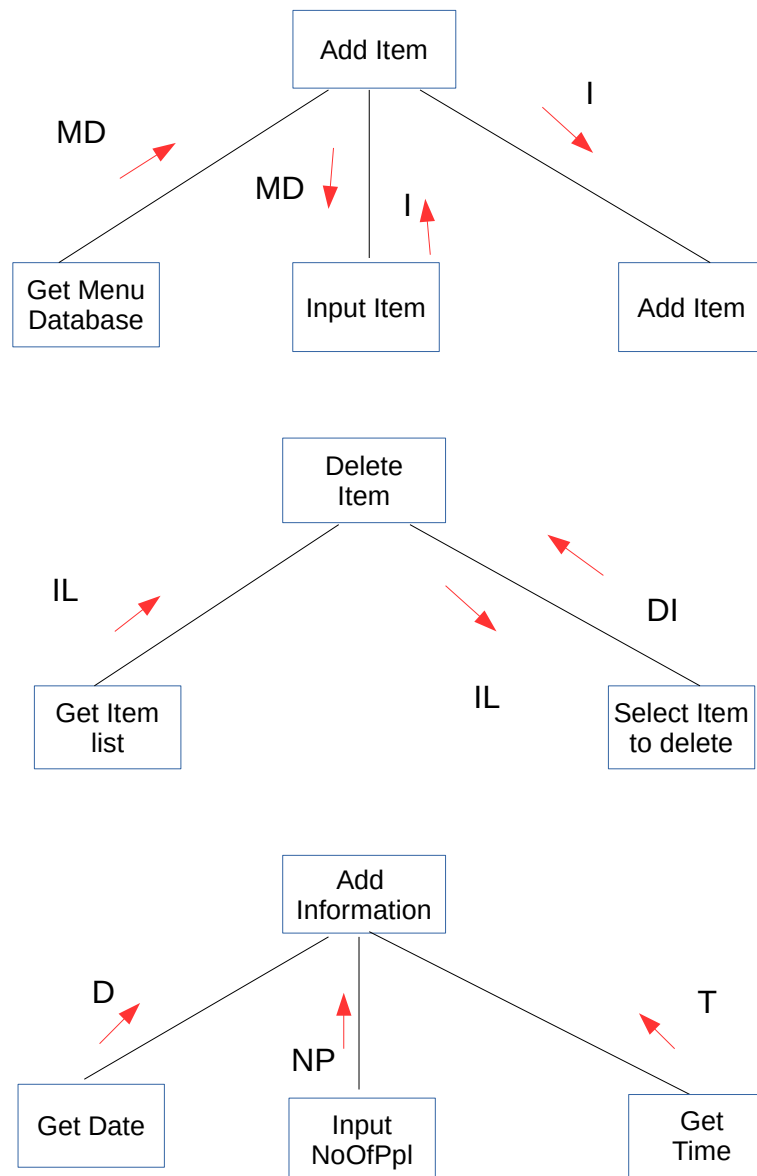


Figure 2.18: Add Booking Structure



2.4.2 Algorithms in pseudo-code for each data transformation process

Algorithm 4 Password change

```

1: OldPassword  $\leftarrow$  CurrentPassword
2: ValidNewPassword  $\leftarrow$  False
3:
4: OUTPUT "Please enter the old password"
5: UserCurrentPassword  $\leftarrow$  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  $\leftarrow$  USERINPUT
11:    OUTPUT "Please re – enter the new password"
12:    ReEnteredNewPassword  $\leftarrow$  USERINPUT
13:    IF len(NewPassword) > 4 AND NewPassword =
        ReEnteredNewPassword THEN
14:      CurrentPassword  $\leftarrow$  NewPassword
15:      ValidNewPassword  $\leftarrow$  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  $\leftarrow$  USERINPUT
4: IF GetMenuID in MenuID Database THEN
5:   ItemAdded  $\leftarrow$  (MenuIDDatabase , MenuItems
        OrderList.insert(ItemAdded)
6: ELSE
7:   OUTPUT You have entered an invalid menuID
8: ENDIF

```

Algorithm 6 Calculating prices

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

2.4.3 Object Diagrams

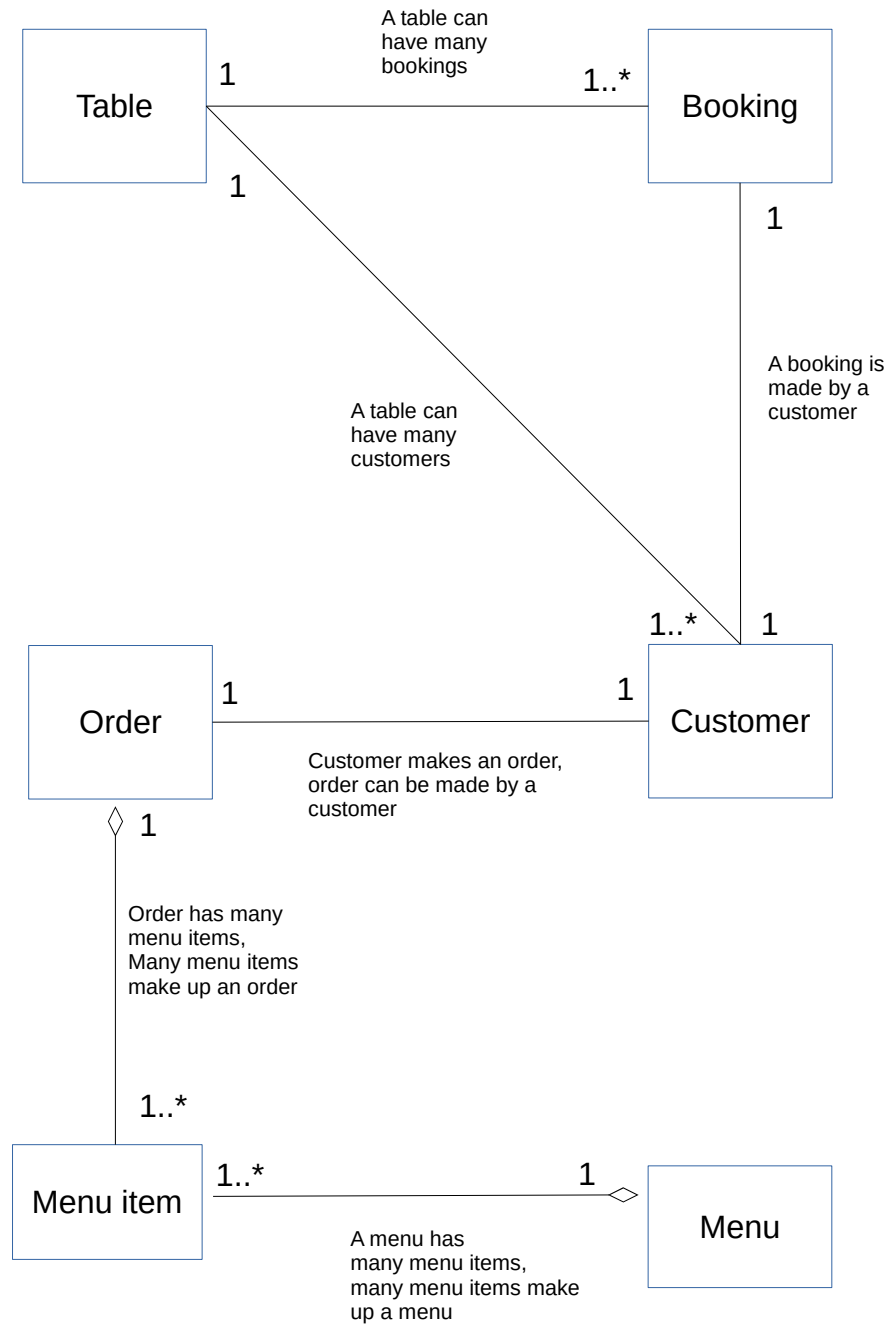


Figure 2.20: Object Diagram

2.4.4 Class Definitions

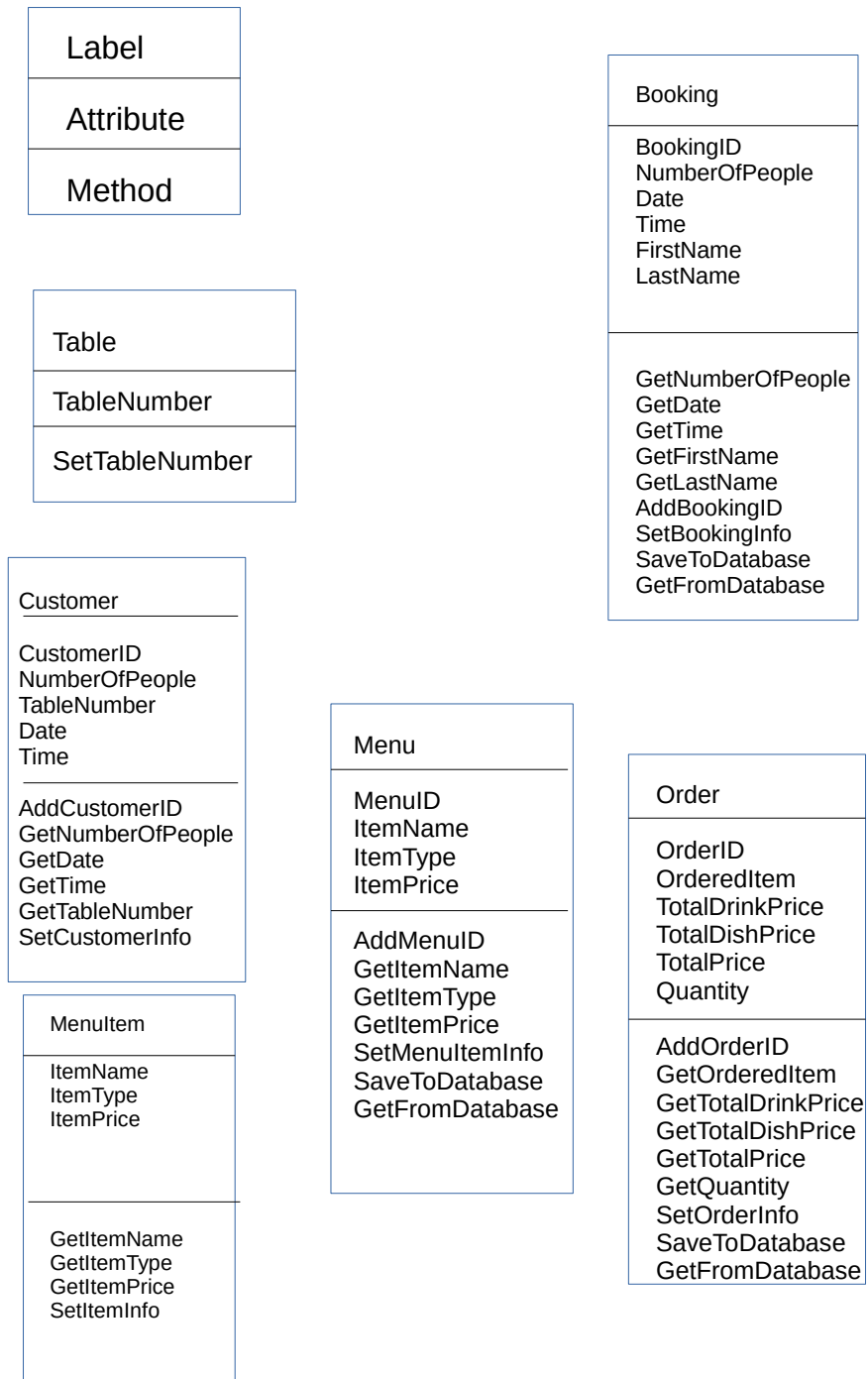


Figure 2.21: Class Definitions

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 prototype:

- 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 Dish-Price	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

2.6.5 Identification of appropriate storage media

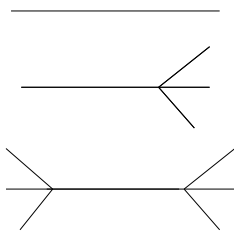
A hard drive would be preferable to store information due to the large capacity size for the database and the speed to transfer data. The only data that will be stored will be stored in the database which will hold old data for 2 years as data older than 2 years will be deleted. The application itself 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

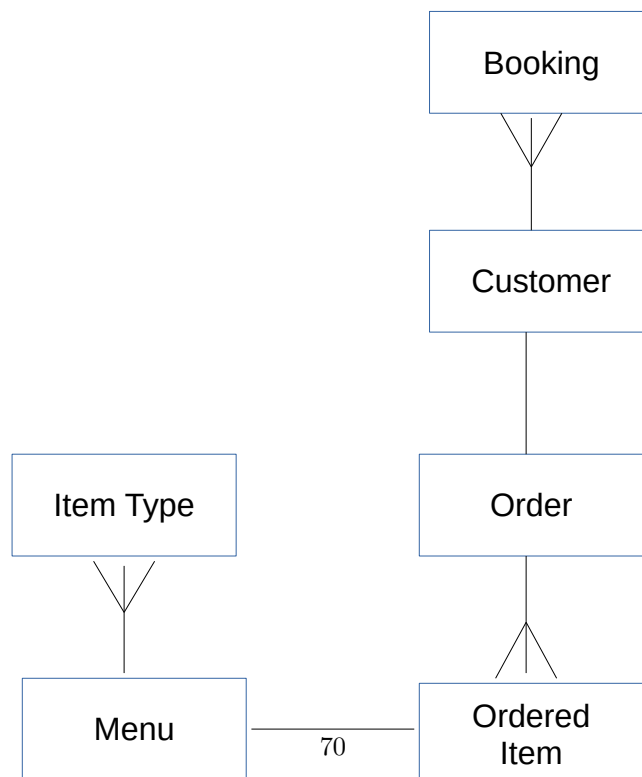
Key



One to one relationship

One to many relationship

Many to many relationship



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 formatted using Python.

```

1  create table Menu(
2      MenuID integer,
3      MenuItem text,
4      ItemPrice real,
5      ItemTypeID integer,
6      primary key(MenuID)
7      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*

```

1  insert into OrderItem
2  where OrderID = ?, MenuID = ? and Quantity = ?

```

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

```

1  select *
2  from Booking
3  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 *Today'sDate* holds the system date at that current time.

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

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

```
1 select *
2 from OrderedItems
3 where OrderID = ?
```

This will return all ordered items from an order.

```
1 update ItemPrice
2 from Menu
3 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 implement 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 telephone 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. The 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 unauthorised 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 fulfils 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 interface 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 indicate 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), 01523859372, 014829, 0158925 8295289	Erroneous, Normal, Erroneous, Erroneous	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), 06/05/18, 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
Example	Example	Example	Example

3.1.2 Changes to Outline Plan

Test Series	Purpose of Test Series	Testing Strategy	Strategy Rationale
Example	Example	Example	Example

3.1.3 Original Detailed Plan

8

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

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
Example	Example	Example	Example	Example	Example	Example	Example

3.2 Test Data

3.2.1 Original Test Data

3.2.2 Changes to Test Data

3.3 Annotated Samples

3.3.1 Actual Results

3.3.2 Evidence

3.4 Evaluation

3.4.1 Approach to Testing

3.4.2 Problems Encountered

3.4.3 Strengths of Testing

3.4.4 Weaknesses of Testing

3.4.5 Reliability of Application

3.4.6 Robustness of Application

Chapter 4

System Maintenance

4.1 Environment

4.1.1 Software

4.1.2 Usage Explanation

4.1.3 Features Used

4.2 System Overview

4.2.1 System Component

4.3 Code Structure

4.3.1 Particular Code Section

4.4 Variable Listing

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

92

4.5.5 SQL Queries

4.6 Testing

4.6.1 Screenshots of Results

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

5.4.1 Error 1

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