

Computing Unit 3: Coursework: Design Chapter

Oliver Looney

Breaking down of the problem into subproblems

Explanation/description of each form/interface and an overview of its functionality and the problem(s) it will solve:

I will need to log in and register customers and staff

- Whenever someone logs in their log in details must be checked against the access table
- They will need to get a different view depending on their role
- They must be able to register from this screen
- They must be able to login using a barcode

Customer:

I will need a section where the customer can browse the menu

- The customer should be able to add these items to a cart
- The customer should be able to switch between multiple categories of the menu
- The customer should be able to click on each item and view more details about that item

I will need a section where the customer can view & interact with the cart

- They will need to be able to view details of what is in their cart
- They will need to be able to change the quantity of each item
- They will need to be able to remove items
- They will need to see a subtotal & total
- The customer should have the option to save this order

I will need a section for the customer to check out

- This will need to have access to the customer's cart
- This will need to be able to save information in the customer's cart in the database in the ORDER & ORDERPART tables
- This will need user input for the order type (collection, in store or delivery)
- The customer should be able to chose if they are paying with a credit card or cash
- The customer should be able to pay with card through this section

I will need a section for the customer to track their order

- This should only show their current order
- It will have to retrieve data from the ORDER & ORDERPART tables

I will need a section for the customer to view their details & saved orders

- These should output data retrieved from the CUSTOMER table
- The customer should be able to see a history of receipts form all of their previous orders
- The customer should be able to view all saved orders and quickly re-order any of them straight away

Staff:

When they are creating their account they will need an admin to activate their account.

They should receive an SMS message or email reminding them about their shift times the day before

They will need a section to order for customers

- They should be able to create, edit and delete customer's accounts
- They should be able to view the current orders' statuses

A section for routine procedures

- They must be able to view the status of each on going order
- They must be notified when each order is completed
- They should be able to see how long it takes for each item & the target time
- Any orders that they take, should be associated with their account
- They should be able to print the receipt
- They should be able to view their shift times
- They should receive a message about their shift times beforehand

Kitchen Staff:

When they are creating their account they will need an admin to activate their account.

They should receive an SMS message or email reminding them about their shift times the day before.

A section for monitoring ingredients

- They will need to see how much of each ingredient is left
- They should be able to change the quantity of each ingredient
- The system will automatically account for the quantity of each ingredient being used for each item

A section for each item

- They will need to view a list of items
- Each item should have a recipe and ingredient list, estimated time it should take and a target time

A section for customer orders

- They must be able to update the status for each order part they have completed
- They should be able to print out stickers for the customer's food boxes
- When an order is finished the system will notify the customer/ counter staff

A section for routine procedure

- They should be able to view their shift times
- The system should analyse all of the previous orders and the time and day, and from that be able to roughly predict to the kitchen staff how much of each item they will have to make. This allows the system to be more accurate when tracking how much of each ingredient the restaurant needs for the next day or week

Driver:

When they are creating their account they will need an admin to activate their account.

They should receive an SMS message or email reminding them about their shift times the day before.

They should be able to view their shift times.

The manager and the customer should be able to track the driver

A section to view upcoming orders that need to be delivered

- They should be able to view the order parts of each order
- They should be able to view the customer details

A section to help deliver

- This should include a map with the customer's locations overlaid

Manager:

A section to monitor staff accounts

- The manager must be able to enable & disable accounts
 - Whenever a member of staff is creating an account they must have their account enabled by the manager before they can do anything, while the customer's account should be automatically enabled

A section for editing the menu

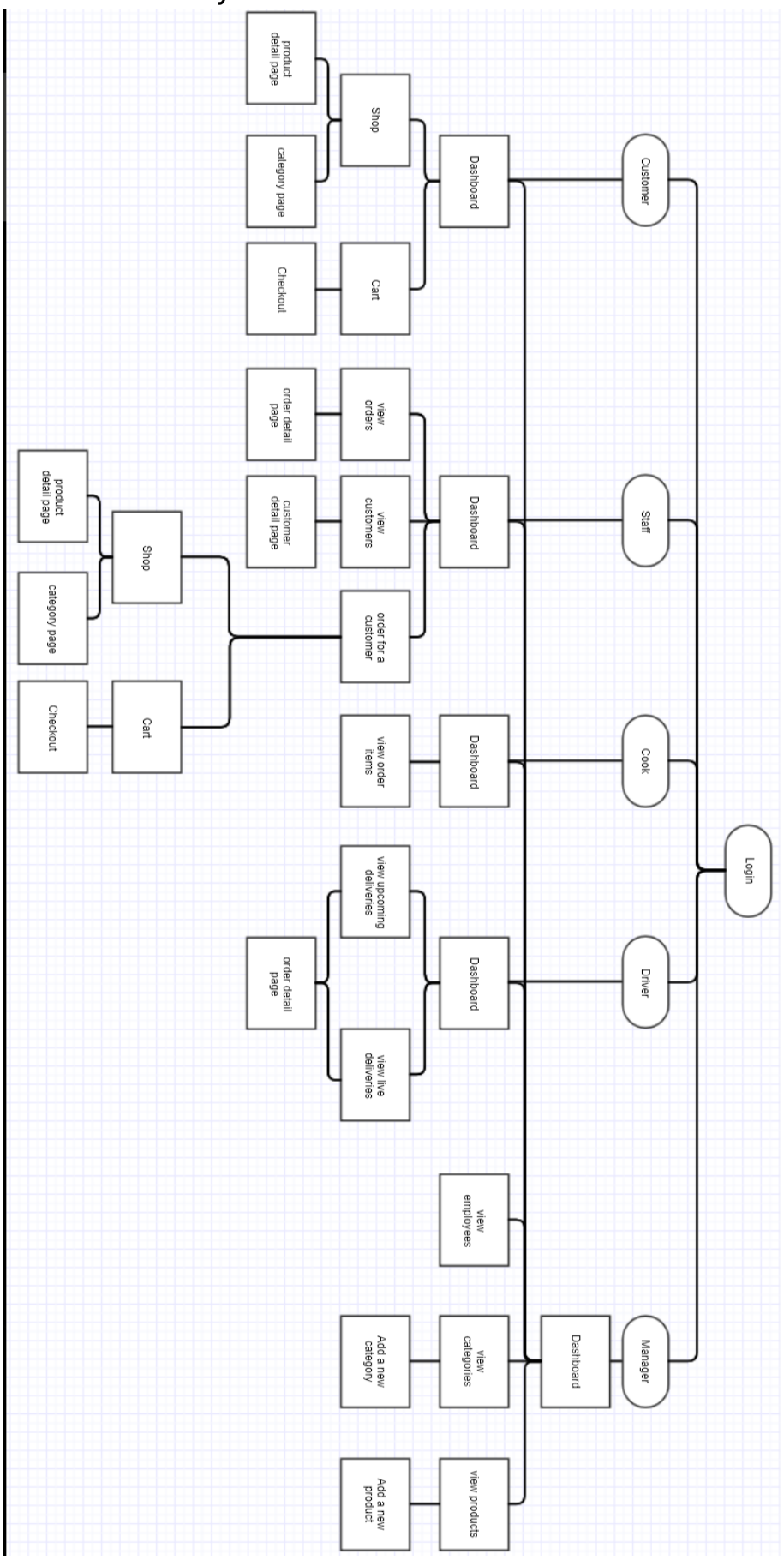
- The manager must be able to add an item to the menu, with a recipe and ingredient list
- The manager must be able to edit the opening, delivery, pick up and closing times for each day
- The manager should be able to hide items on the menu from the customer if they do not want that item to be ordered but do not want the hassle of deleting the record temporarily
- The manager should have the option of hiding how long it takes to prepare items or how long it takes to deliver, from the customer. However, the staff should still see it so that they can improve to the target.

A section for viewing graphed data about the restaurant

- Heatmap
- Be able to pull any customer variable against other variables
- Time graphs
- Most/least popular

A section for daily & weekly reports

diagrammatic overview of the system:



Decomposition

Decomposition is one of the four cornerstones of Computer Science. It involves breaking down a complex problem or system into smaller parts that are more manageable and easier to understand. The smaller parts can then be examined and solved, or designed individually, as they are simpler to work with. In Django I will have separate app, and in each have separate files/modules, which will mean my code will use decomposition. This will apply for example by having a customer app which will be responsible for all of the customer's user requirements, etc.

Required files and / or data structures

Normalisation

Database Normalization is a technique of organizing the data in the database. Normalization is a systematic approach of decomposing tables to eliminate data redundancy and undesirable characteristics like Insertion, Update and Deletion Anomalies.

Method of Access

For most of my access I shall be using random/direct access to my database tables. This will be done for viewing customer information for example, using the customer id. However whenever I want to view the customer's order information for example I will have to use the customer's id, so that will be an indexed sequential access. I may also use normal text files with serial access to store information on the restaurant such as opening & closing times.

Random-access file is a term used to describe a file or set of files that are accessed directly instead of requiring that other files be read first. Computer hard drives access files directly, where tape drives commonly access files sequentially.

Serial file organisation is the simplest file organisation method. In serial files, records are entered in the order of their creation. As such, the file is unordered, and is at best in chronological order.

Sequential access devices are usually a form of magnetic storage or optical storage. While sequential access memory is read in sequence, arbitrary locations can still be accessed by "seeking" to the requested location.

ISAM (Indexed Sequential Access Method) is a file management system developed at IBM that allows records to be accessed either sequentially (in the order they were entered) or randomly (with an index). Each index defines a different ordering of the records. An employee database may have several indexes, based on the information being sought. For example, a name index may order employees alphabetically by last name, while a department index may order employees by their department. A key is specified in each index. For an alphabetical index of employee names, the last name field would be the key.

O Normal Form:

Order:

Order ID, created, updated, paid, completed, order slug, saved, order status, order type **Order Item ID**, , quantity, order item status, **Product ID**, name, slug, image, description, price, available, created, updated, **Category ID**, name, slug, **MyUser ID**, gender, password, username, is_superuser, is_staff, is_cook, is_driver, is_manager, is_active, date_joined, first_name, last_name, email, address, postal_code, city, last_login, customer_slug, is_customer

1st Normal Form:

Order:

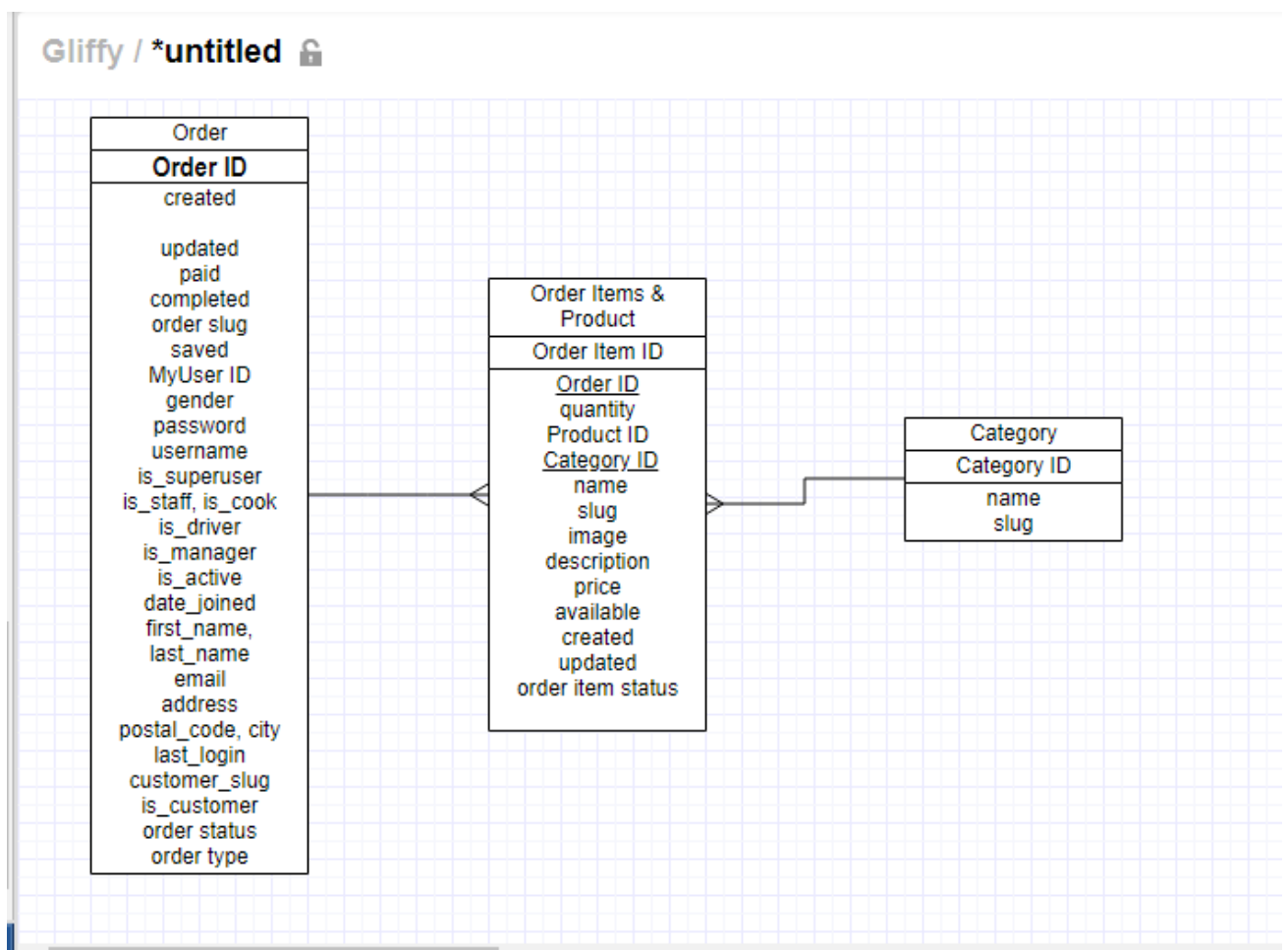
Order ID, created, updated, paid, completed, order slug, saved, order type, order status, **MyUser ID**, gender, password, username, is_superuser, is_staff, is_cook, is_driver, is_manager, is_active, date_joined, first_name, last_name, email, address, postal_code, city, last_login, customer_slug, is_customer

Order Items & Product:

Order Item ID, Order ID, quantity, **Product ID**, Category ID, name, slug, image, description, price, available, created, updated, order item status

Category:

Category ID, name, slug



2nd Normal Form:

Order:

Order ID, created, updated, paid, completed , order slug, saved, order status, order type MyUser ID

MyUser:

MyUser ID, gender, password, username, is_superuser, is_staff, is_cook, is_driver, is_manager, is_active, date_joined, first_name, last_name, email, address, postal_code, city, last_login, customer_slug, is_customer

Order Items:

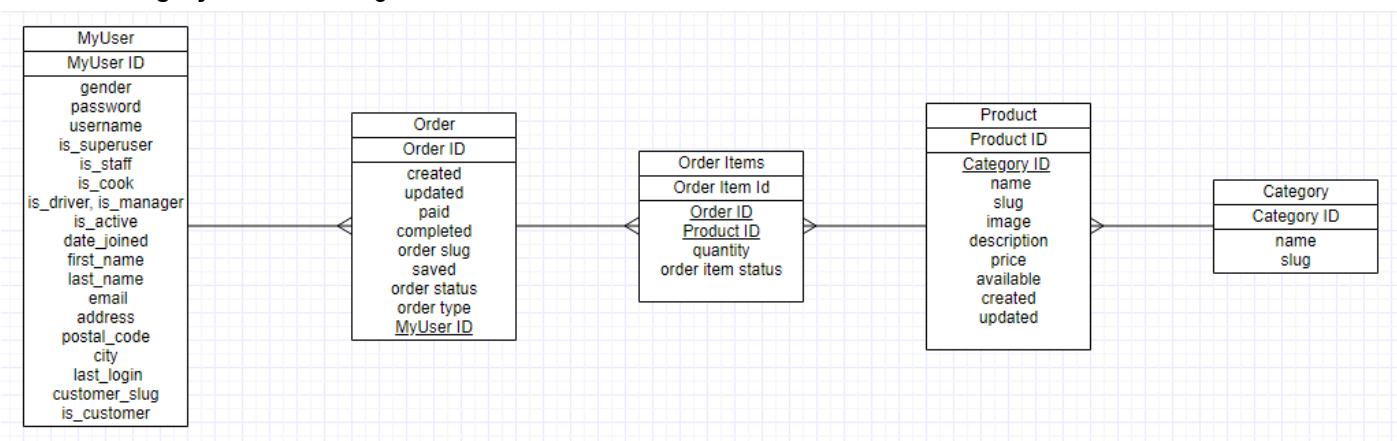
Order Item ID, Order ID, Product ID, quantity, order item status

Product:

Product ID, Category ID, name, slug, image, description, price, available, created, updated

Category:

Category ID, name, slug



3rd Normal Form:

Order:

Order ID, created, updated, paid, completed , order slug, saved, order status, order type MyUser ID

MyUser:

MyUser ID, gender, password, username, is_superuser, is_staff, is_cook, is_driver, is_manager, is_active, date_joined, first_name, last_name, email, address, postal_code, city, last_login, customer_slug, is_customer

Order Items:

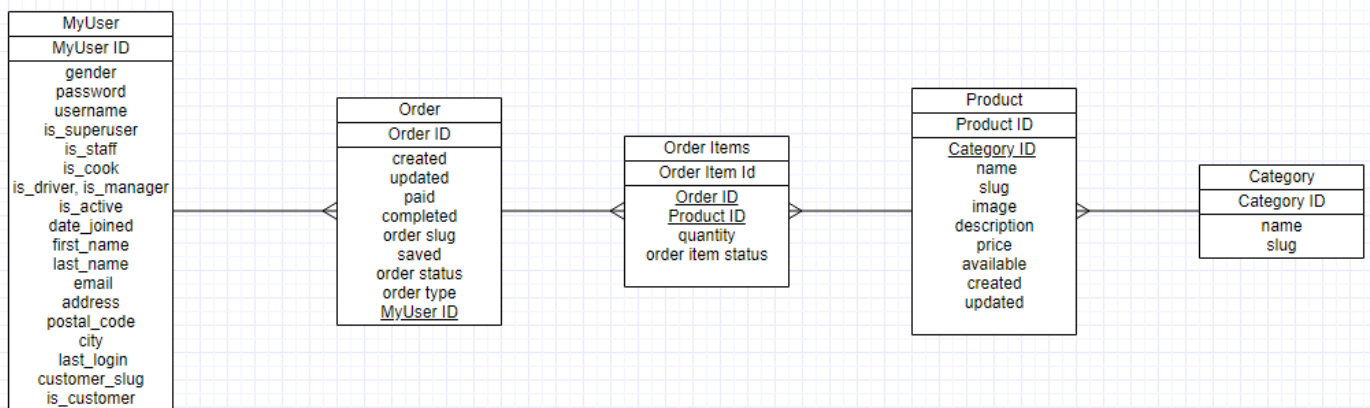
Order Item ID, Order ID, Product ID, quantity, order item status

Product:

Product ID, Category ID, name, slug, image, description, price, available, created, updated

Category:

Category ID, name, slug



Data Descriptions (Data Dictionary)

Note: R = Required, I = Indexed

User:

This table is required because, as stated in the investigation, the system must allow the employees to enter the relevant and necessary information on customers. This information must be kept secure and must only be used for the reasons stated during collection of data.

CUSTOMER									
Related to Table:					Order				
Foreign Keys:					None				
General table description:									
Field Name	R	I	Data Type	Length	Input Validation	Default	Description	Typical Data	
MyUser_id	Y	Y	Integer	2 bytes	It is auto incremented, therefore no validation is required	None	Primary Key, uniquely identifies each customer	3	
firstName	Y	N	String	50 bytes	Length Check Presence Check	None	Customer's first name	Oliver	
lastName	Y	N	String	50 bytes	Length Check Presence Check	None	Customer's last name	Looney	
dateJoined	Y	N	Date	10 bytes	Format Check Presence Check	None	Customer's date of joining	2018-16-11	
gender	Y	N	boolean	1 byte	Type Check Presence Check	None	Customer's gender	Male Female	
address	Y	N	string	100 bytes	Length Check Presence Check	None	Customer's address	261 Main Street	
postcode	Y	N	string	10 bytes	Look Up Check Length Check Presence Check	None	Customer's postcode	BT9 6RB	

email	Y	N	string	50	Format Check Length Check Presence Check	None	Customer's email address	email@ email.com
password	Y	N	String	50	Length Check Presence Check	None	Customer's password	Ebfygb328h
username	Y	N	String	50	Length Check Presence Check	None	Customer's username	Olooney62
is_superuser	Y	N	boolean	1	Type Check Presence Check	None	Boolean flag to determine if super user	0 or 1
is_staff	Y	N	Boolean	1	Type Check Presence Check	None	Boolean flag to determine if staff member	0 or 1
is_cook	Y	N	boolean	1	Type Check Presence Check	None	Boolean flag to determine if cook	0 or 1
is_driver	Y	N	Boolean	1	Type Check Presence Check	None	Boolean flag to determine if driver	0 or 1
is_manager	Y	N	Boolean	1	Type Check Presence Check	None	Boolean flag to determine if manager	0 or 1
is_active	Y	N	boolean	1	Type Check Presence Check	None	Boolean flag to determine if account is active	0 or 1
customer_slug	Y	N	string	200	Presence Check Length Check	MyUser ID	Used in the url of customer detail page	3
is_customer	Y	N	boolean	boolean	Type Check Presence Check	None	Boolean flag to determine if customer	0 or 1

Order:

This table is required because, as stated in the investigation, the system must allow the employees and customers to enter the relevant and necessary information on orders. This information must be kept secure and must only be used for the reasons stated during collection of data.

ORDER								
Related to Table:					Customer, OrderPart			
Foreign Keys:					Customer_id			
General table description:								
Field Name	R	I	Data Type	Length	Input Validation	Default	Description	Typical Data
Order_id	Y	Y	integer	2 bytes	It is auto incremented, therefore no validation is required	none	Primary key, uniquely identifies each order	3
Customer_id	Y	N	integer	2 bytes	Presence check Type check Check if the relationship can exist	none	Foreign key, identifies the order's customer	3
orderType	Y	N	string	1 byte	Look Up Check Presence Check	's' In store	A single character code that saves whether the order is in store, delivery or collection	('s', 'd', 'c')
orderStatus	Y	N	String	1 byte	Look Up Check	'p' preparing	A single character code that stores what stage of the process the order is on	('p', 'b', 'r')
created	Y	N	Date time	10 bytes	Format Check Presence Check Range Check	Current date time	Date and time of order creation	01/05/2019 01:24
updated	Y	N	Date time	10 bytes	Format Check Presence Check Range Check	Current date time	Date of time of when the order was last updated	01/05/2019 01:24
completed	Y	N	Boolean	1 byte	Presence Check	0	Boolean flag to determine if	0

							it has been completed	
order slug	Y	N	String	200 bytes	Presence Check	Order ID	Used in the url of order detail page	3
saved	Y	N	boolean	1 byte	Presence Check	0	Boolean flag to determine if it has been saved	0

Order Items:

This table is required because, as stated in the investigation, the system must allow the employees and customers to enter the relevant and necessary information on orders, which includes order items. This information must be kept secure and must only be used for the reasons stated during collection of data.

ORDERPART								
Related to Table:					Order, Menu, Cook			
Foreign Keys:					Order_id, Product_id			
General table description:								
Field Name	R	I	Data Type	Length	Input Validation	Default	Description	Typical Data
OrderPart_id	Y	Y	Integer	2 bytes	It is auto incremented, therefore no validation is required	none	Primary key, uniquely identifies each orderPart	3
Order_id	Y	N	Integer	2 bytes	Presence check Type check Check if the relationship can exist	None	Foreign key, identifies the order item's order	3
Product_id	Y	N	Integer	1 byte	Presence check Type check Check if the relationship can exist	None	Foreign key, identifies the order item's product	3
Quantity	Y	N	Integer	1 byte	Range Check Presence Check Type Check	1	How much of each item	2
status	Y	N	String	1 byte	Look Up Check	'p'	status	b

Product:

This table is required because, as stated in the investigation, the system must allow the employees to enter the relevant and necessary information on products, which includes order items. This information must be kept secure and must only be used for the reasons stated during collection of data.

PRODUCTS								
Related to Table:					Order, Menu, Cook			
Foreign Keys:					Category_id			
General table description:								
Field Name	R	I	Data Type	Length	Input Validation	Default	Description	Typical Data
Product_id	Y	Y	Integer	2 bytes	It is auto incremented , therefore no validation is required	none	Primary key, uniquely identifies each Product	3
Category_id	Y	N	Integer	2 bytes	Presence check Type check Check if the relationship can exist	None	Foreign key, identifies the product's category	3
Name	Y	N	String	200	Length Check Presence Check	None	Name of product	Fanta
Slug	Y	N	String	200	Length Check Presence Check	Name	Used in making url of product detail	Fanta
Image	N	N	String	No limit	Format Check	No_image.png	Used to show customer what the product looks like	Image
Description	N	N	String	No limit	None	None	Description of product	Fshfgf uwe fefef
Price	Y	N	Decimal	10 digits	Range Check Format Check	None	Price of product	5.00

Available	Y	N	Boolean	1 byte	Presence Check	0	Boolean flag determining if a product is available	1
Created	Y	N	Datetime	1 byte	Format Check	Current time	Date and time of product creation	01/05/2019 01:24
updated	Y	N	DateTime	1 byte	Format Check	Current time	Date of time of when the order was last updated	01/05/2019 01:24

Category:

This table is required because, as stated in the investigation, the system must allow the employees to enter the relevant and necessary information on categories, which includes order items. This information must be kept secure and must only be used for the reasons stated during collection of data.

PRODUCTS								
Related to Table:					Products			
Foreign Keys:								
General table description:								
Field Name	R	I	Data Type	Length	Input Validation	Default	Description	Typical Data
Category_id	Y	Y	Integer	2 bytes	It is auto incremented, therefore no validation is required	none	Primary key, uniquely identifies each Category	3
Name	Y	N	String	200	Length Check Presence Check	None	Name of category	Pizza
Slug	Y	N	String	200	Length Check Presence Check	Name	Used in making url of category detail	pizza

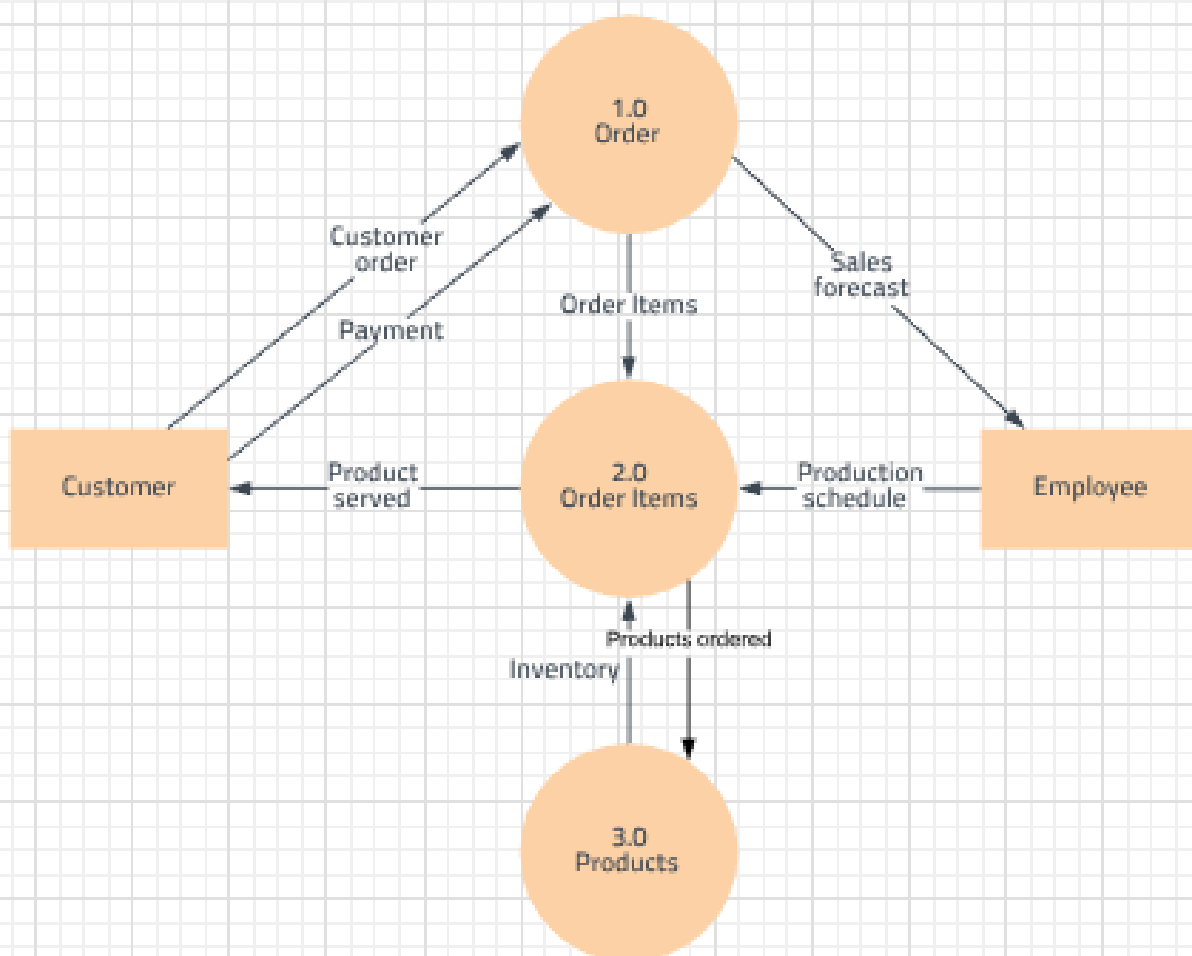
Fully identify and describe the data to be input to and output from the system

While people learn to write code, programs that they produce are often written on the fly. Errors are resolved as they are encountered, and good features are ultimately replaced with better features. As programmes become more complex, and larger teams are used to produce them, design work becomes more important.

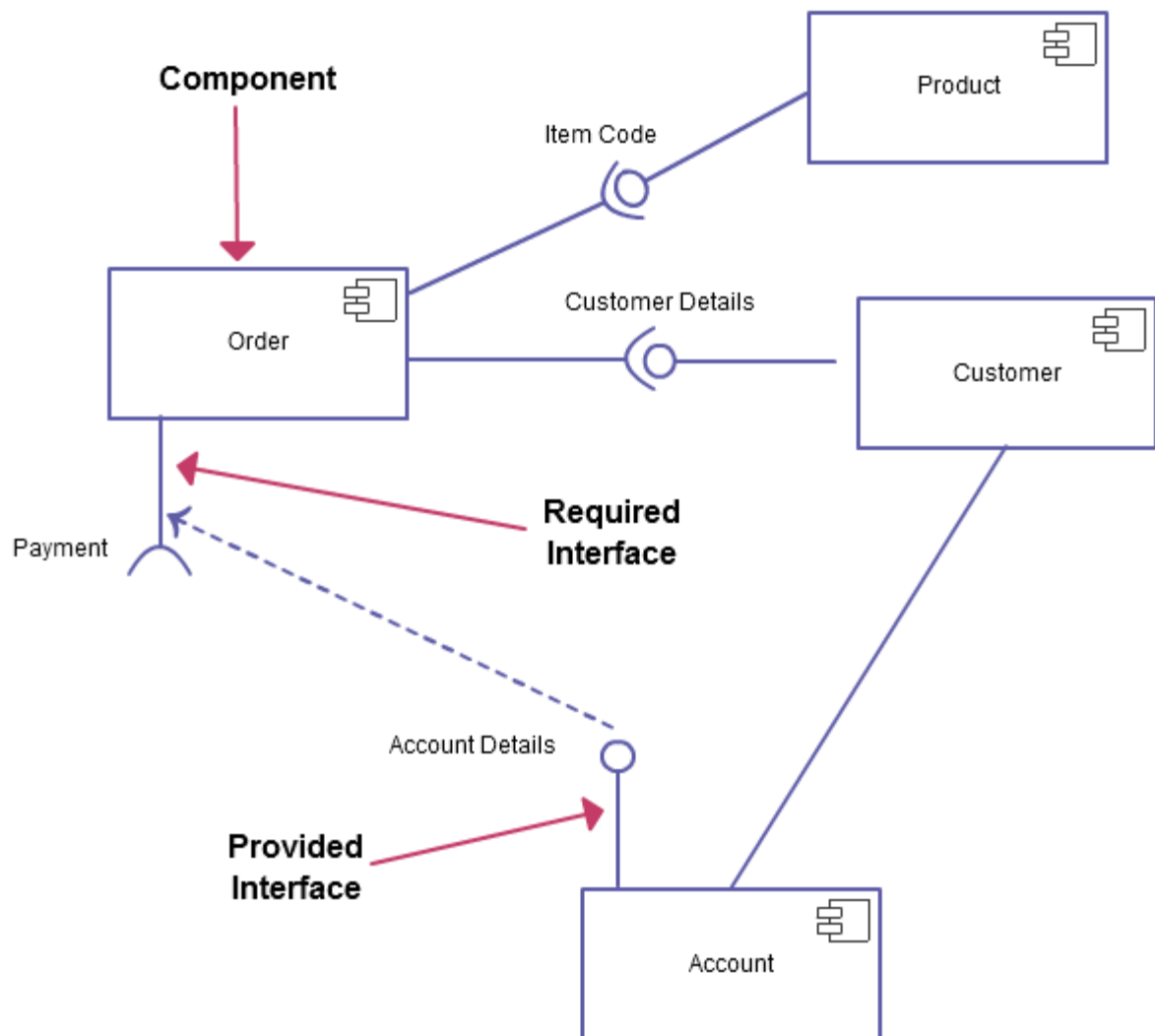
The following are common components of design:

- Data Flow Diagrams, DFD
- Data Dictionaries
- Entity Relationship Diagrams, ERD
- Unified Modelling Language, UML
- Algorithm Designs

Level 0 Data Flow Diagram:

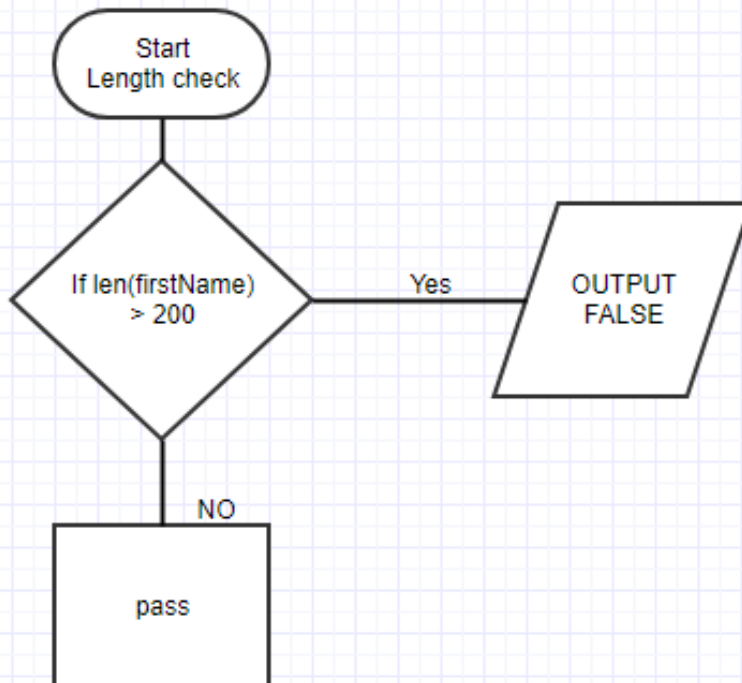


UML:



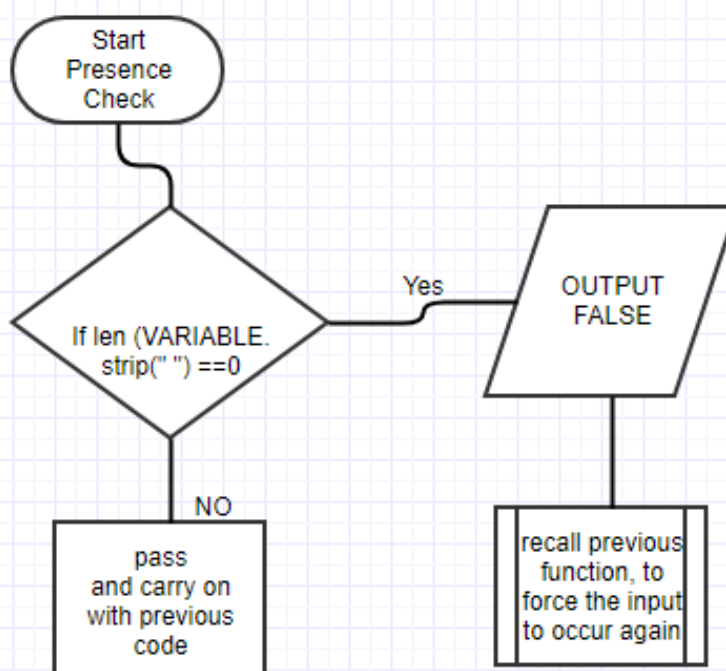
Validation Rules

Length Check:

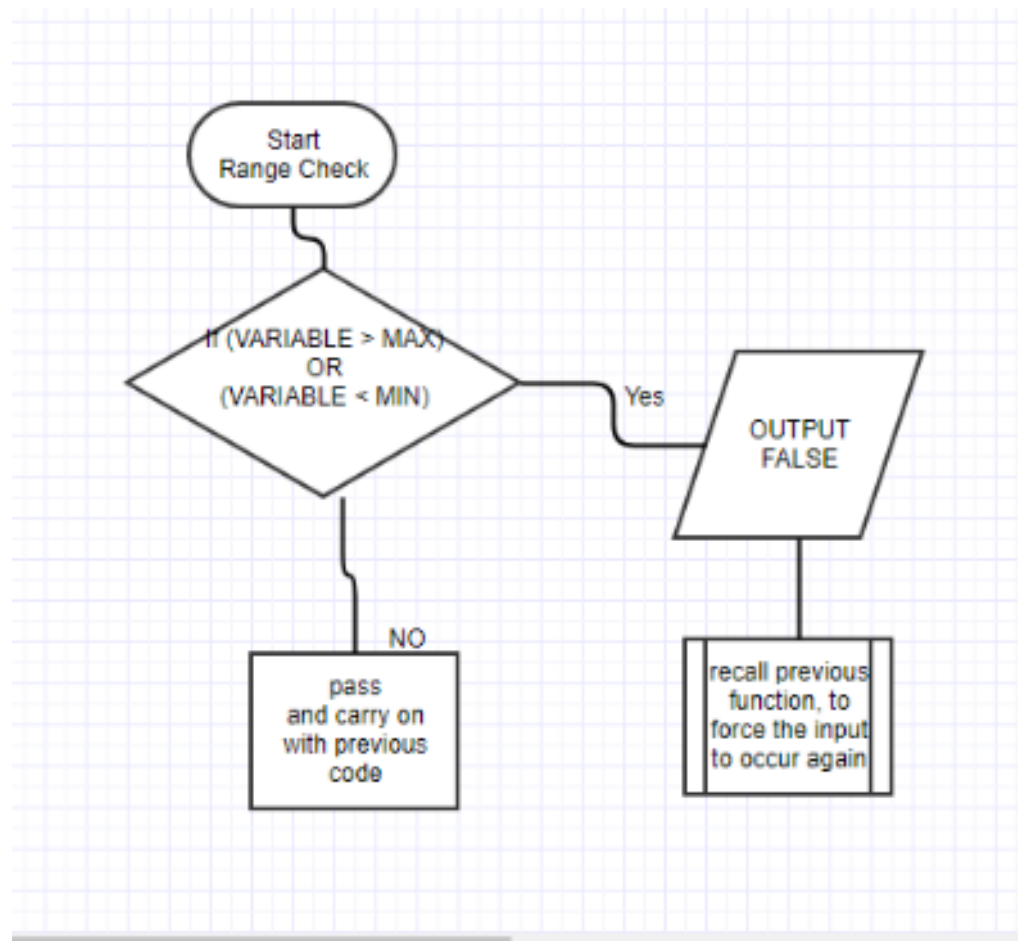


This will be used for the majority of my string inputs, e.g: last & first names, names, descriptions

Presence Check:



Range Check:



This will be used for my quantity variable for example.

Type Check:

```
Def integerTypeChecl(userInput):
```

```
    While True:
```

```
        Try:
```

```
            userFinalInput = int(input(userInput))
```

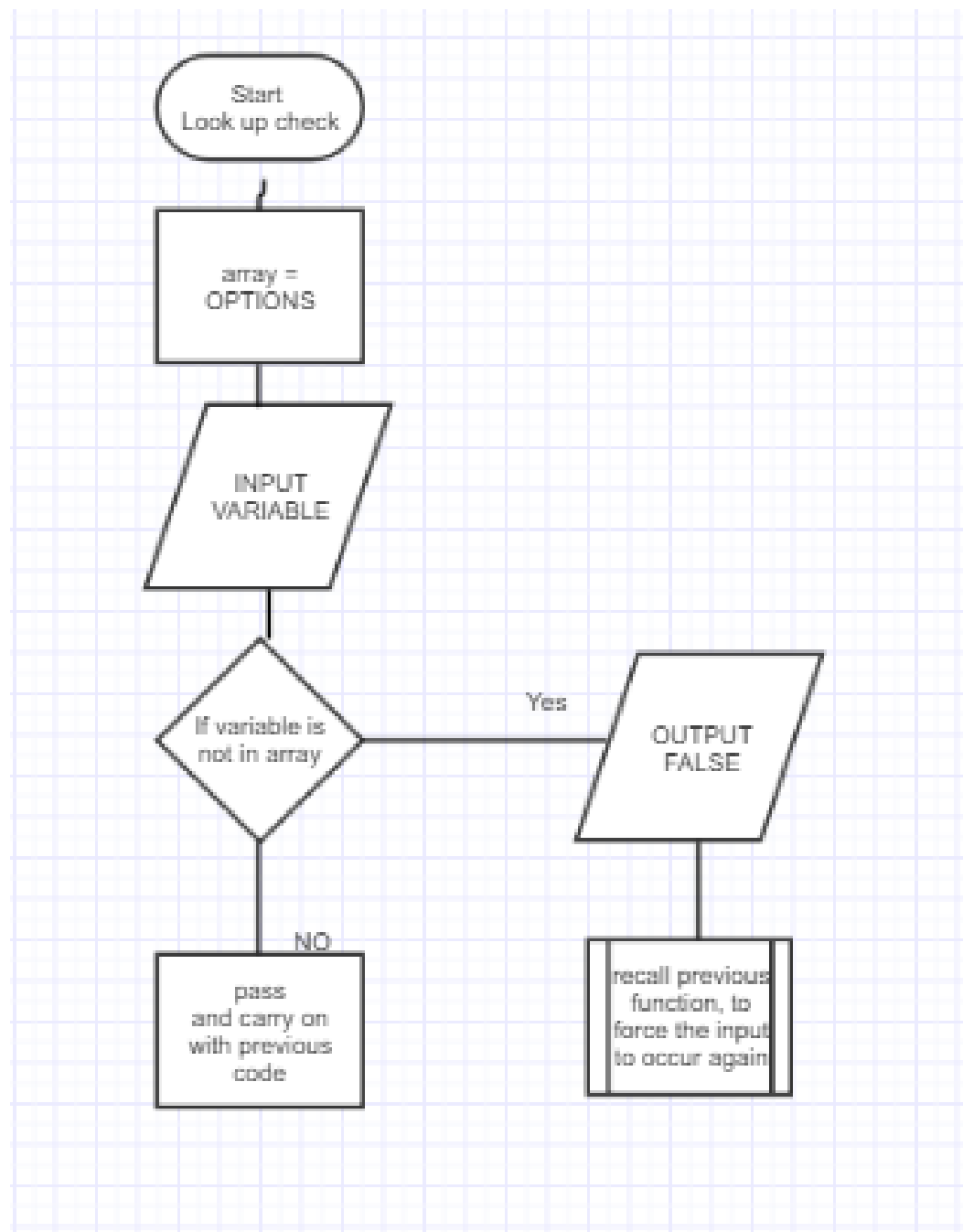
```
        except ValueError:
```

```
            print("ERROR: Not an integer, please try again")
```

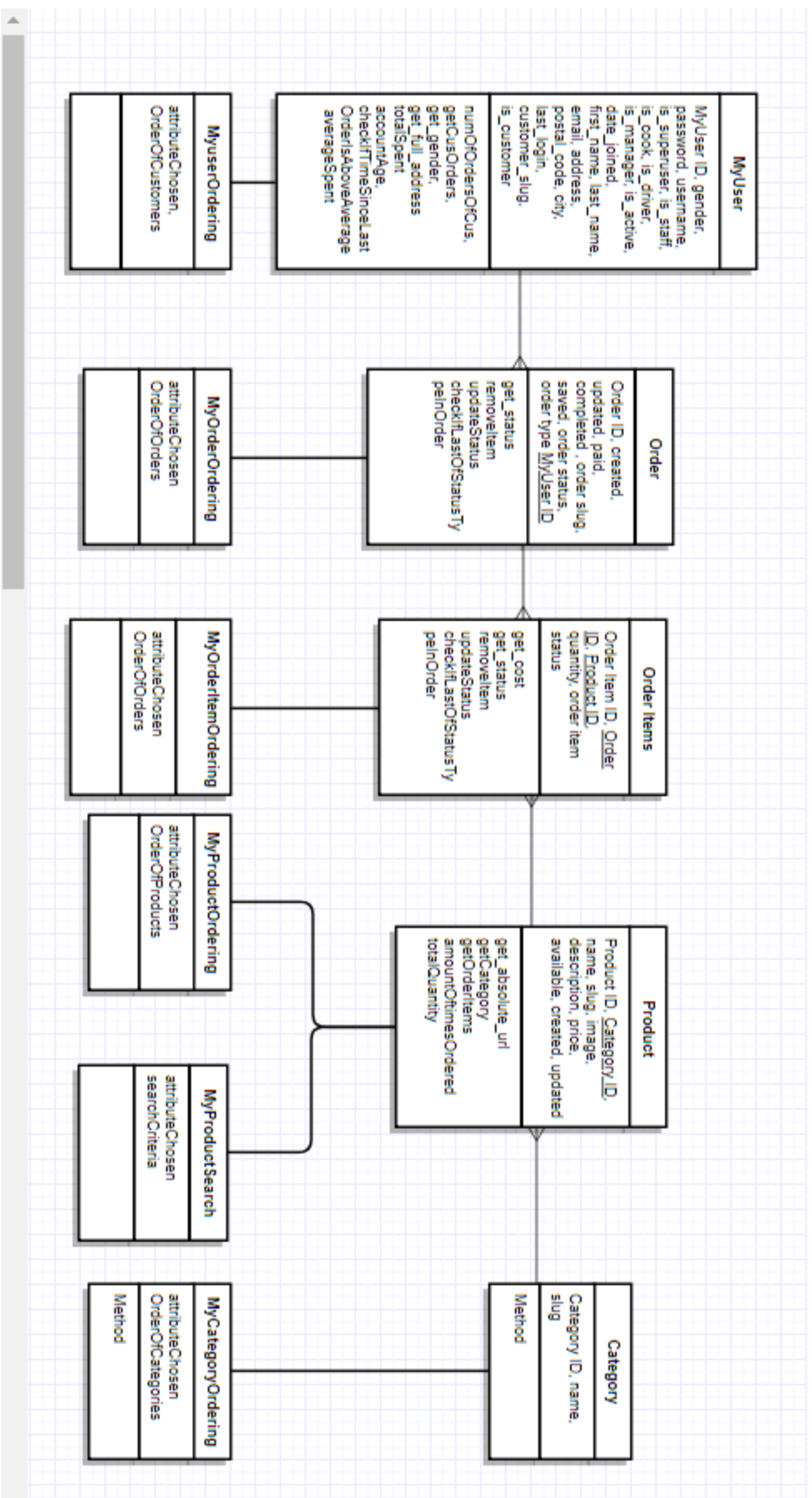
```
        else:
```

```
            return userFinalInput
```

Look up check:



Class Diagram



One Customer has many Orders. One Order has many Order Items. One Product has many Order Items. One Category has many Products. The different types of staff are stored in the MyUser table with the customers. The classes that inherit from the classes previously explained, are used to create user input forms for their respective classes.

Customers can **order** multiple items from a set **menu**, which is created and edited by the **managers**, or the **customer** can create their own pizza. This will require multiple validation algorithms as some items will not be able to be put together on a custom pizza. This will also require a more complex algorithm to calculate the cost as it will be based on input by the **customer**. The **customer** can order over the phone(data is then entered into the system by a member of the **counter-staff**), through a kiosk(data is entered into the system by the **customer** onsite) or online through this database driven system. If the **customer** orders online or on the phone, the system will automatically e-mail the customer a receipt but if the customer orders in the restaurant, their receipt will be printed out as well as receiving an e-mail.

For deliveries a separate **order** is printed out for the driver with other useful information about the **customer**, e.g.: address, phone number. All of the deliveries will have an added bill that will be at a set rate per mile from the customer to the restaurant plus a flat fee. This will be calculated before the driver leaves, using google maps API. The **drivers** would also need to see all of their delivery information online through this system. I will also design an algorithm that displays the customers' addresses on a map for the **drivers**.

The **managers** will need to be able to see the total sales and transactions for that day/week/month. Along with this should be profit and the system should also display what else money was spent on, e.g: how much of their revenue was spent on supplies, wages. This should be graphed by the system. As the system is not going to calculate wages the **managers** will manually input it.

The pizzeria would like to implement a discount scheme where loyal **customers** are rewarded with vouchers/discount. **Customers** spending over a certain amount will receive discount vouchers via email. Different **staff** must have different views of the system, including counter-staff, drivers and managers.

Every receipt will have a barcode that when scanned will bring up information on the **customer** who ordered it, and another barcode that will bring up information on the **order**.

Output Design & Forms

Mock ups

The mockup shows a web browser window with a single tab labeled 'Page 1'. The address bar contains the URL 'www.pedro'spizzeria.com'. The main content area displays a 'Sign In' form. The form has a title 'Sign In' followed by a horizontal line. Below this, there are two input fields: 'User Name:' with the text 'johndoe' and 'Password:' with masked characters '*****'. A blue 'SIGN IN' button is positioned at the bottom of the form.

This is the login screen for all users.

IF sign in button clicked:

```
UserDetails = SELECT * FROM ACCESS
```

```
For user in UserDetails:
```

```
    IF user[1] == currentDetails[0]:
```

```
        IF user[2] == currentDetails[1]:
```

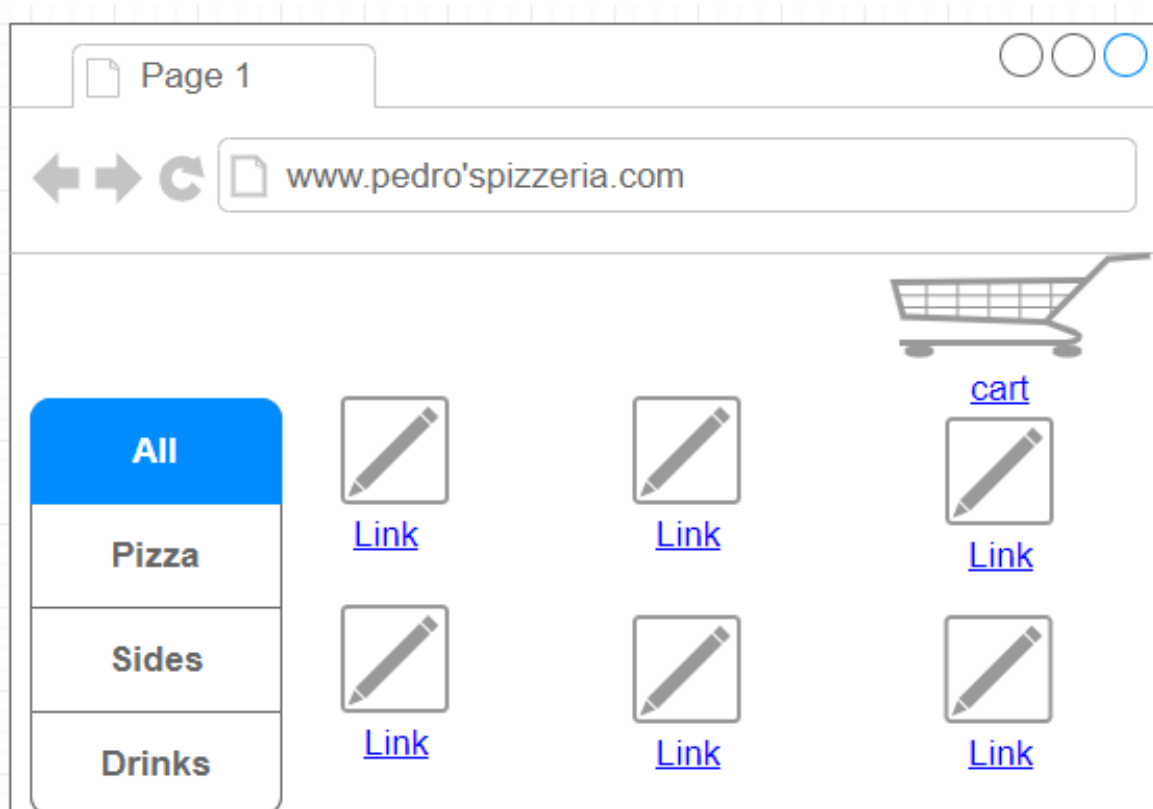
```
            SIGN USER IN
```

```
        ELSE:
```

```
            RETURN("Password Error")
```

```
    ELSE:
```

```
        RETURN("No account found")
```



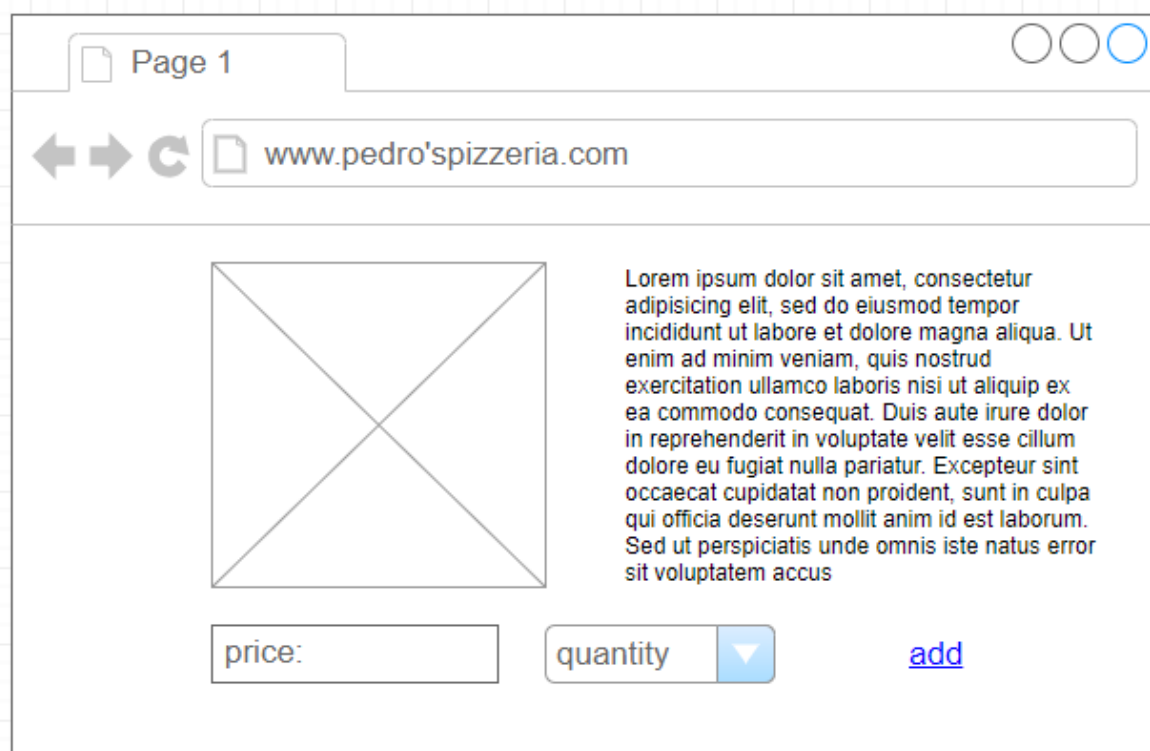
This is the home page for the customer. This allows them to view the products on the menu.

When you click on a product it will bring you to the product's own page, which you can use to view details and add that product to the cart.

Pressing the cart link will bring you to the cart page. Pressing one of the categories (e.g: sides), will only display products that have that category as their category attribute.

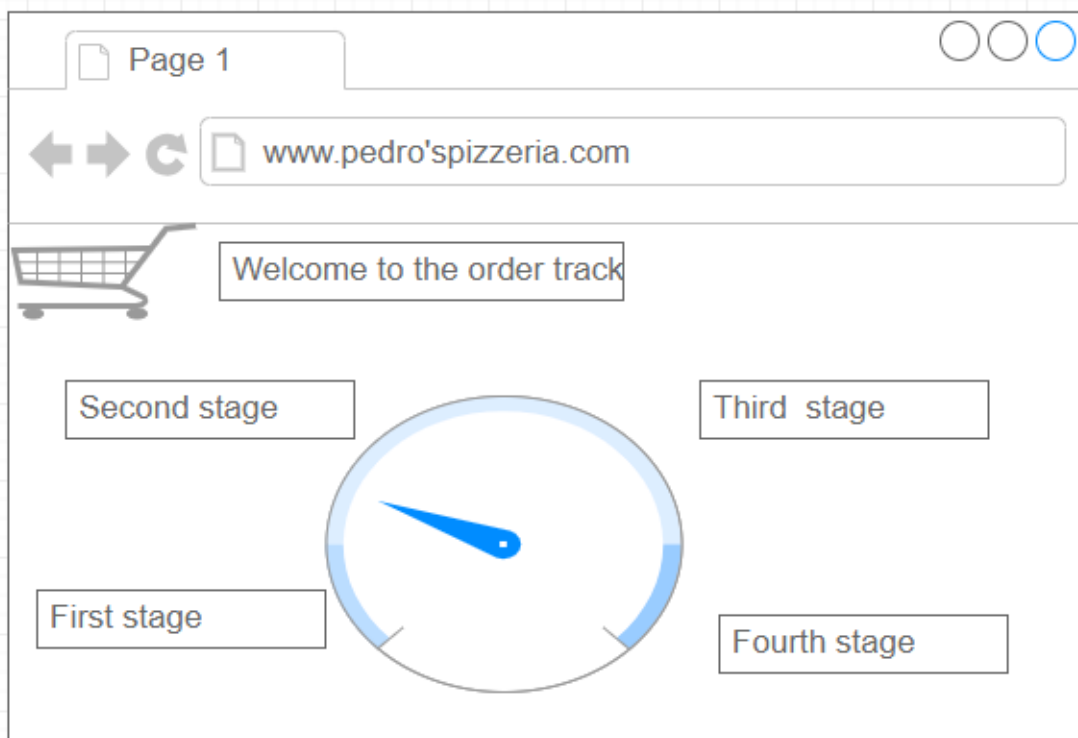
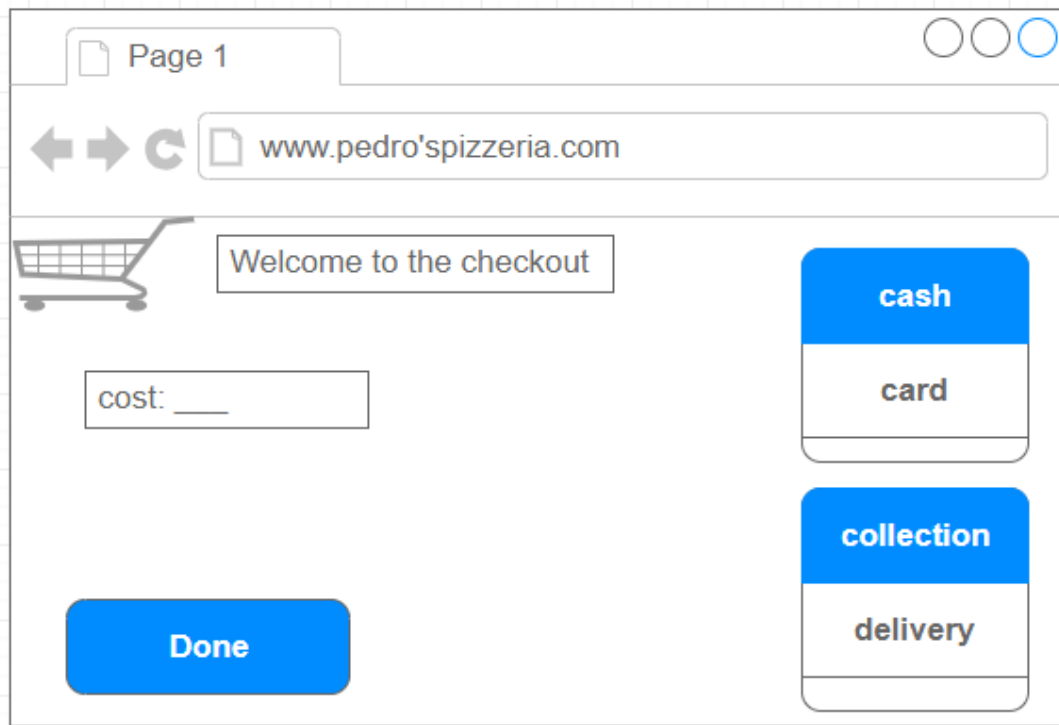


This is the customer's cart. The user will be able to edit the quantity from a drop down menu (look up table) when they click on quantity. The price will be calculated by retrieving the price attribute from that product and multiplying it by the quantity every time the cart is refreshed. When the user clicks checkout they will be brought to the checkout.



Above is the template for the product page. From here the user will be able to view details about the product, set the quantity and then add the product & quantity to the cart.

Below is the checkout page. The user will select the checkout options and when the done button is clicked the order is saved to the database.



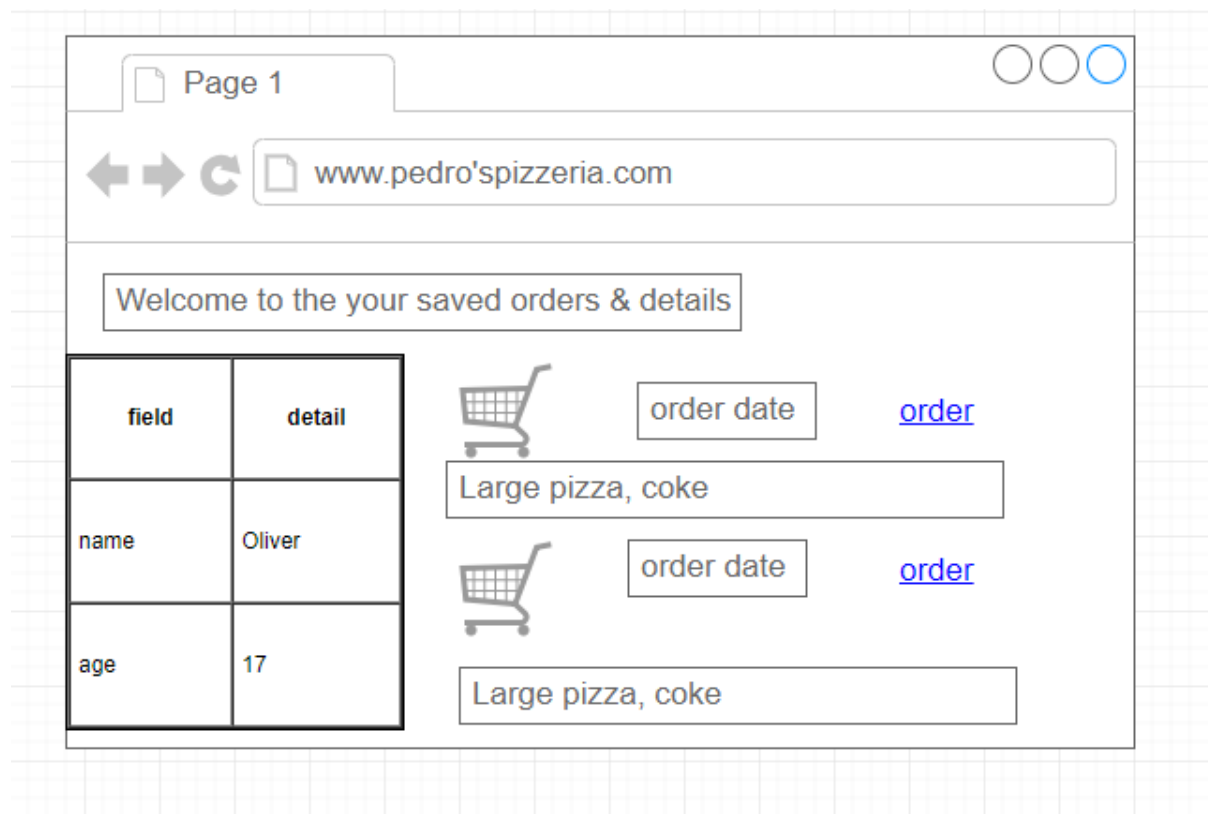
Above is the order track page, this will update the customer to the stage of the order. This page is just output.

cusID = customerID

orders = SELECT * FROM ORDERS WHERE customerID = cusID, finished = False ORDER BY date

currentOrder = orders[-1]

stage = currentOrder[orderStage]



This page will show the user their details and saved orders, and allow them to immediately order one of their saved orders.

Details:

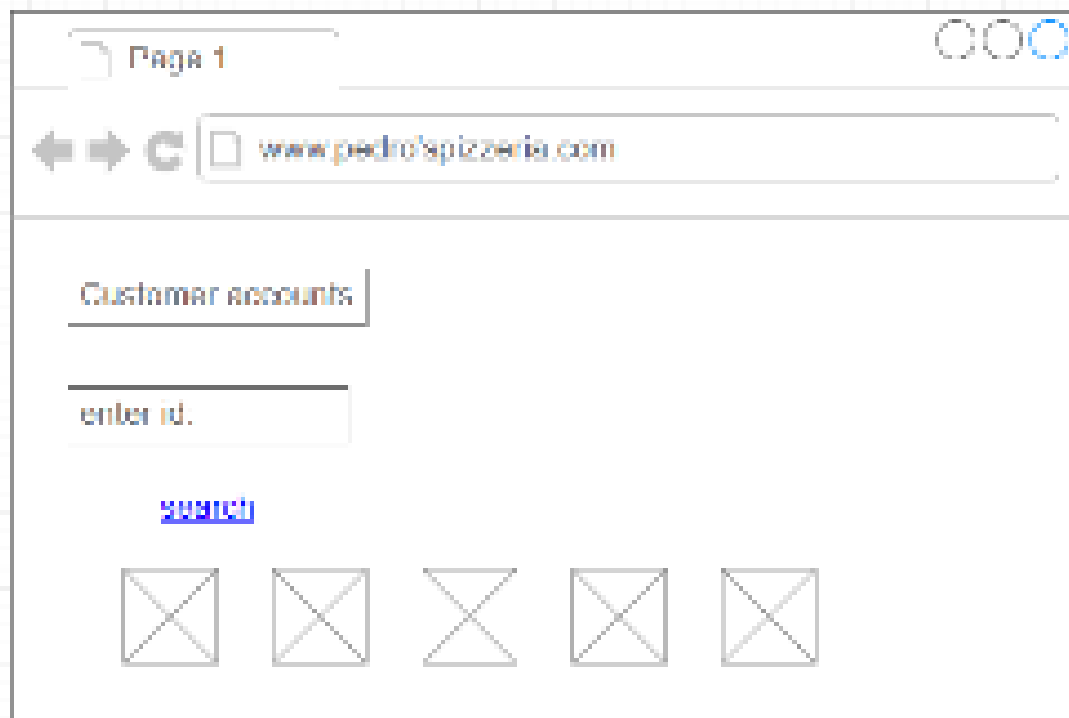
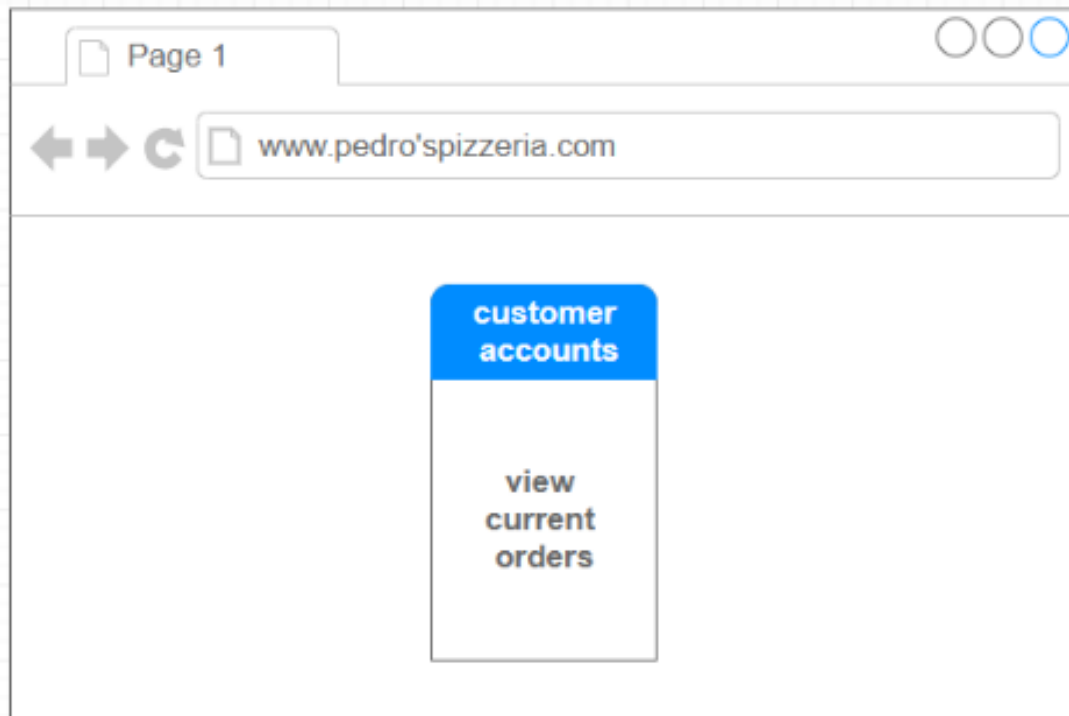
Details = SELECT * FROM CUSTOMERS WHERE customerID = cusID

Orders:

savedOrders = SELECT * FROM ORDERS WHERE customerID = cusID, saved = True

This is the counter staff's view. When the staff logs in they will be presented with the option of viewing information on the customer's accounts or view the current orders going on in the restaurant.

Above is how the counter staff search through the customers to view details.



SELECT * FROM CUSTOMERS WHERE customerID = searchCriteria

Below shows the counter staff's view of the current orders. currentOrders = SELECT * ORDERS WHERE finished = False ORDER BY TIME

A web browser window titled "Page 1" with the address bar showing "www.pedro'spizzeria.com". The page contains a form with three columns of input fields. Each column has three fields: "order #", "status:", and "time ordered:". The form is displayed on a grid background.

order #	order #	order #
status:	status:	status:
time ordered:	time ordered:	time ordered:

A web browser window titled "Page 1" with the address bar showing "www.pedro'spizzeria.com". The page contains a form with three columns of input fields. Each column has three fields: "order #", "item", and "finish". The "item" and "finish" fields are highlighted in blue. The form is displayed on a grid background.

order #	order #	order #
item	item	item
finish	finish	finish

Above is the cook's view of current items to be prepared. When they click on the item, they will be brought to a page to view more details on that item. When they click on the finish button that will update the orderpart table and check if the order is now complete.

```
UPDATE ORDERPART SET finished = True WHERE orderPartId = currentOrderPartID
```

```
orderPartList = SELECT * FROM ORDERPART WHERE orderID = currentOrderID
```

```
finished = True
```

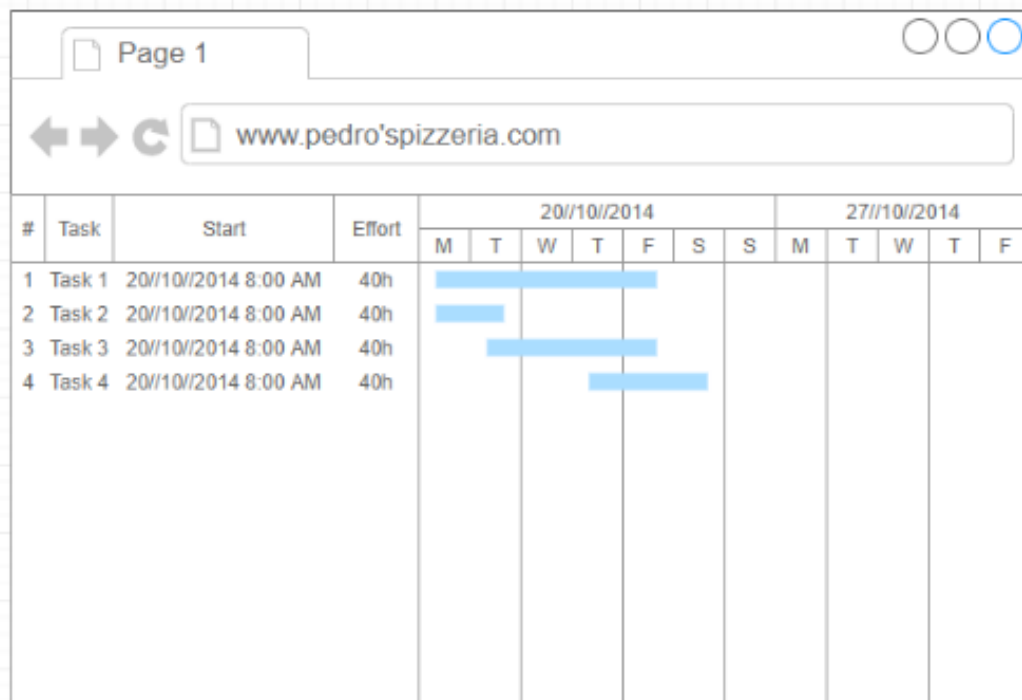
```
for (orderPartItem in OrderPartList) or finished:
```

```
    if orderPartItem[3] == False: #if the order part is not finished
```

```
        finished == False
```

```
IF finished == True:
```

```
    UPDATE ORDER SET finished = True WHERE orderID = currentOrderID
```



This is the view where the cook's will see the expected workload. The system will display average quantities of each item here, so that the staff can plan appropriately.

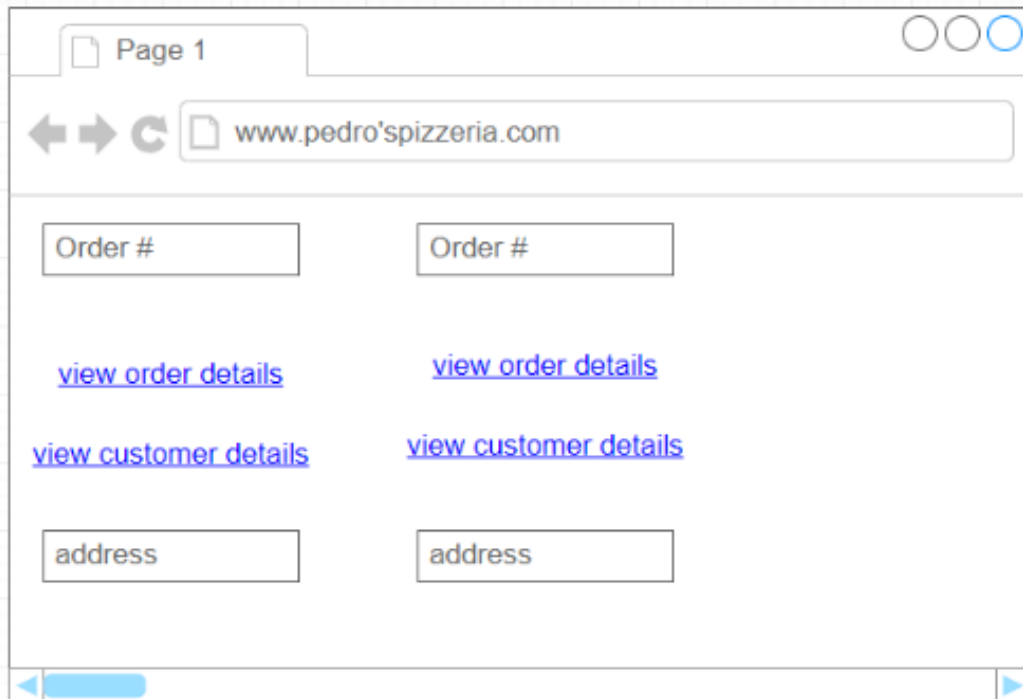
Page 1

www.pedro'spizzeria.com

search... [search](#)

Item name	Item name	Item name
		
view recipe	view recipe	view recipe

Here the cook's can search for items by the productid to view the recipe and ingredient of each product.



This is the view for the delivery drivers. This displays the current deliveries.

```
currentDeliveries = SELECT * FROM ORDERS WHERE finished = True, delivery = True
```

The driver can view the customer details & the order details by clicking on the links and the address is already displayed.

```
currentCusID = SELECT customerID FROM ORDERS WHERE orderID = currentOrderID
```

```
customerAddress = SELECT address FROM CUSTOMER WHERE cusID = currentCusID
```

Below is the page where the driver can view customer details.

Page 1

← → ↻

www.pedro'spizzeria.com

Customer #

name:

address:

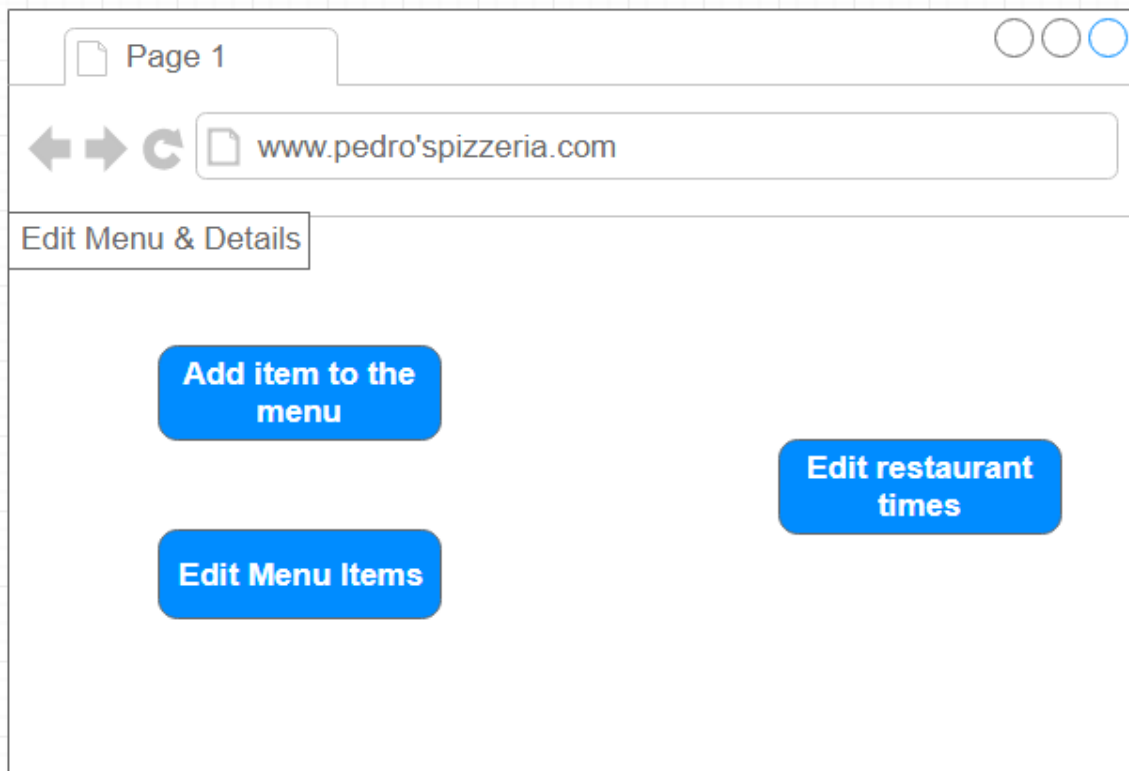
email:

phone number:

Below is the driver's view of the map with the customer's addresses overlaid on top. I shall be using the python map library folium to do this part.



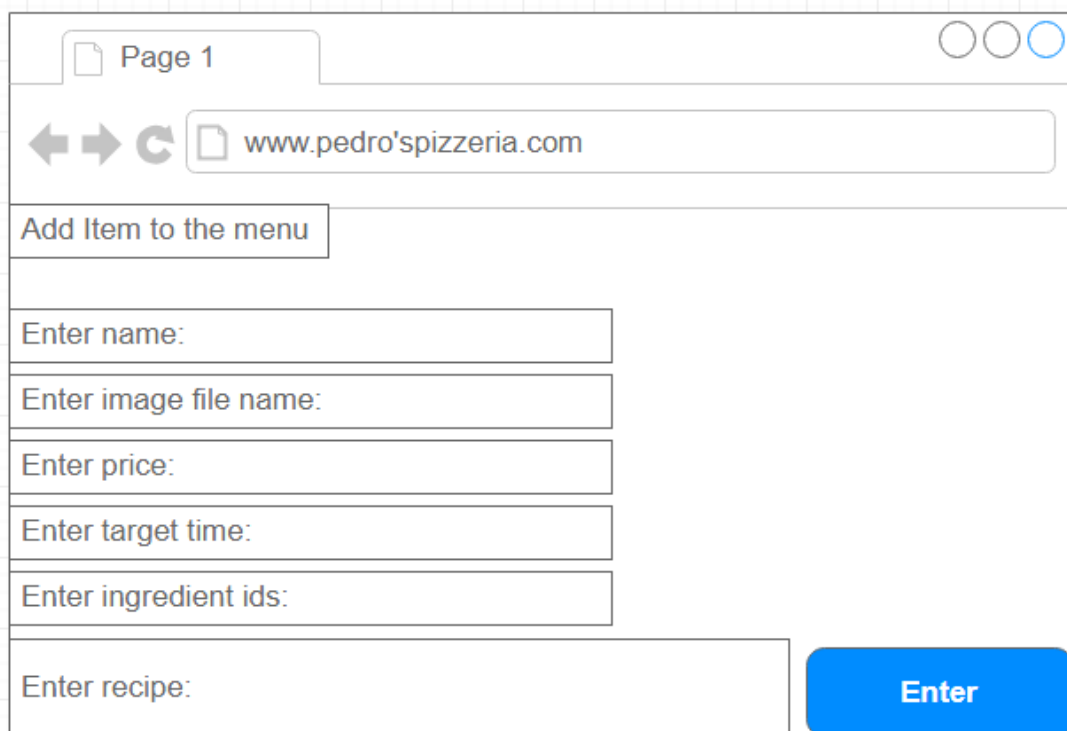
The manager will get a notification will accept or decline option whenever a new staff account is created. This is the managers opening view.



The image shows a web browser window with a single tab labeled "Page 1". The address bar contains the URL "www.pedro'spizzeria.com". Below the address bar is a header bar with the text "Edit Menu & Details". The main content area contains three blue buttons with white text: "Add item to the menu" on the left, "Edit Menu Items" below it, and "Edit restaurant times" on the right.

If the manager clicks add item they will be brought to the screen below.

The manager inputs the data for the new product and presses enter to save the it to the database.

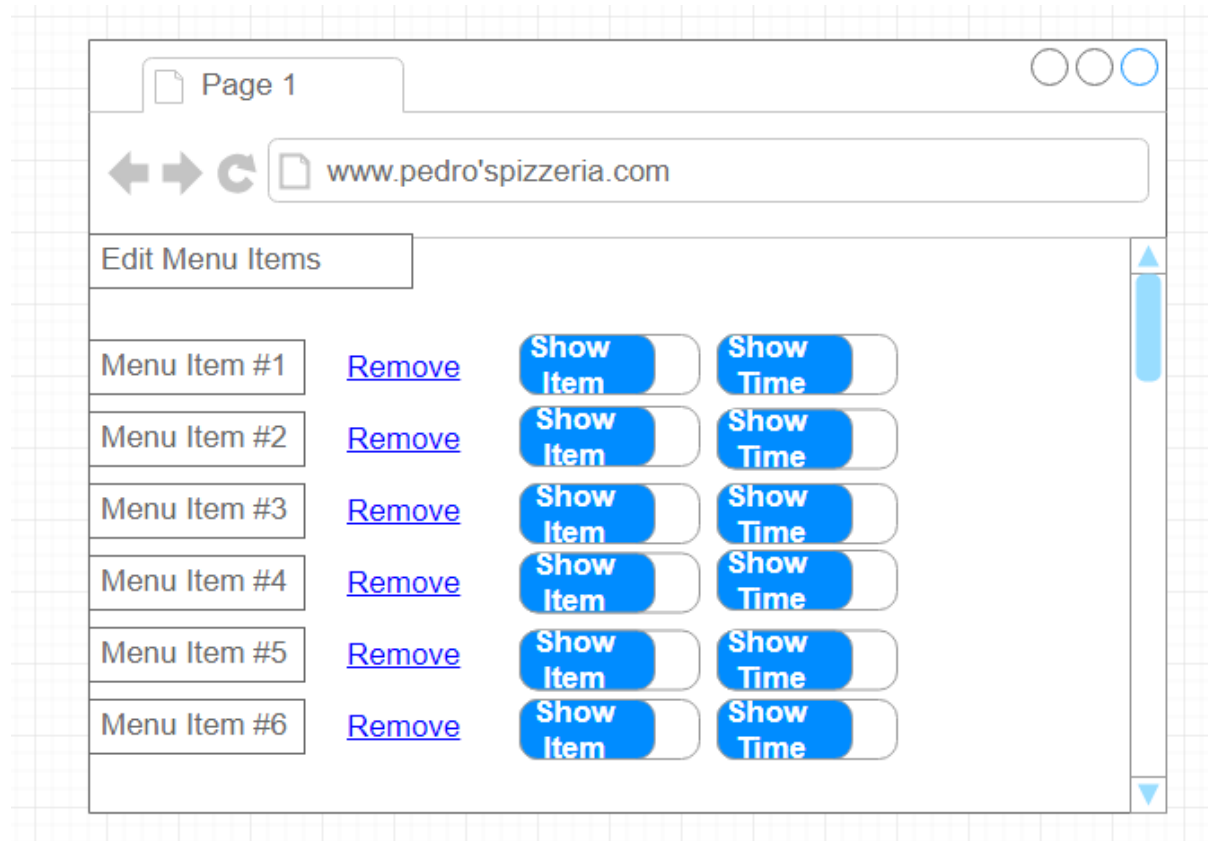


The image shows a web browser window with a single tab labeled "Page 1". The address bar contains the URL "www.pedro'spizzeria.com". Below the address bar is a header bar with the text "Add Item to the menu". The main content area contains several input fields for data entry: "Enter name:", "Enter image file name:", "Enter price:", "Enter target time:", "Enter ingredient ids:", and "Enter recipe:". A blue button with the text "Enter" is located at the bottom right of the form.

INSERT INTO MENU VALUES(name, image file name, price, target time, recipe, ...)

FOR ingredientid in ingredientids:

INSERT INTO MENUINGREDIENT VALUES (menuID, ingredientID)



This is where the manager can edit the menu items. The manger can remove the items:

When remove clicked, the menu item id is passed through.

DELETE FROM MENU WHERE menuID = currentMenuID

DELETE FROM MENUINGREDIENT WHERE menuID = currentMenuID

The manager can also hide items from the user by toggling the option:

UPDATE MENU SET hide = True WHERE menuId = currentMenuID

or

UPDATE MENU SET hide = False WHERE menuId = currentMenuID

The manager can hide the times from the user by toggling the option:

UPDATE MENU SET showTime = True WHERE menuId = currentMenuID

or

UPDATE MENU SET showTime = False WHERE menuId = currentMenuID

Page 1

www.pedro'spizzeria.com

Edit Restaurant Times

Option ▼

New Start Time:

New Finish Time:

[Save](#)

Here the manager can edit the opening & closing times, delivery times and more from the drop down menu (look up table). This data is simply stored in a text file



This is where the manager will view the reports. This will be the main screen, which will consist of a heatmap, a bar graph showing the most popular items and other necessary graphs. The manager will also be able to view other custom graphs.