Reports Format

**Problem Statement**: This Pizza Ordering System is designed to make an easy to use yet highly customizable website wherein customers can order pizzas to their specification from the comfort of their homes. Oftentimes, pizza restaurants are weighed down by having to deal with both in house customers and remote ones. A cashier can’t answer the phone to accept an order if they’re taking the order of a dine-in customer, for instance. In order to solve this problem, our restaurant only offers takeout and delivery options, further streamlining an already efficient process and solving many problems along the way. In order to further speed up the process, repeat customers will have their details saved for quicker service.

**Functional Requirements**:

-Records will be created for each customer, tied to their phone number.

-Records must include name, address, phone number, type of charge account, and any information needed to find their address.

-Must accept both cash, check and Visa/Mastercard.

-A record must be kept of the type of payment and the amount of it.

-A business class user must be able to access the database that will show a list of all customer transactions, as well as their address and necessary delivery information.

-For each credit card transaction, a receipt must be printed out which the customer can then sign.

-For both credit card and non-credit card orders, the receipt must contain basic customer information, a list of the items ordered, if it is for delivery or pickup, and the amount due.

-The system must have a GUI based menu for order-taking purposes. This menu must include various options commonly found on pizzas, including size, toppings and beverage options to go with the meal.

Non-Functional Requirements

-Website must work in all major browsers. This includes Internet Explorer, Firefox and Chrome.

-Website should be fast, and not require a high-end computer to run at full speed.

-Ordering must be simple enough to quickly understand and use.

**Stakeholders:**

-SWE Team 6: Group behind the project, creating the website and its accompanying documentation.

-Prof. Mona Chavoshi: Client the project is being made for.

**Use Cases:**

Use Case Name: Customer Record Creation (u1.1)

Actors: Customer, Database.

Precondition: Begins when user clicks the ‘create account’ button OR user purchases item.

1.1.1 User clicks the ‘create account’ button, taking them to new page.

1.1.2 Subflow: Alternatively, user successfully orders item(s).

1.2 User enters in their information. This includes name, phone number, address, type of charge, and any necessary information for locating address.

1.3.1 User clicks confirms their account creation.

1.3.2 Subflow: Alternatively, User finishes order by clicking the ‘check out’ button.

1.4 All information is sent to the database, where it is saved and tied to the given phone number for later use.

Use Case Name: Card Payment Processing (u1.2)

Actors: Database, Credit Card Companies, User.

Precondition: Begins when user attempts to make a payment. User has items in cart.

1.1.1 Given payment info and amount to charge is sent to whichever credit card company the customer has indicated, either Visa or Mastercard, through their public API.

1.1.2 Exceptional Flow: Alternatively, if customer chooses to pay with either cash or check, a record of this is recorded, and the order proceeds as normal for payment in person.

1.2 Payment info is checked by the credit card company, as is amount charged. This is then either confirmed or denied by the company.

1.3.1 If payment is confirmed, a database record is made containing date, payment type, and amount. Use case u1.3 is triggered, printing out a receipt.

1.3.2 Subflow: Alternatively, if payment is rejected by the credit card company, the order does not go through, and the reason is stated on screen as to why it was rejected.

Use Case Name: Receipt Generation (u1.3)

Actors: Database

Precondition: Begins when successful credit card payment is made, OR when a manually receipt is printed from the database.

1.1.1 Receipt generation is automatically queried after a successful credit card payment.

1.1.2 Subflow: Alternatively, a receipt generation is started via manually calling the database.

1.2 The database is queried for the user’s information.

1.3 This information is used to create a receipt. Information on the receipt includes all relevant information for the order, including address, date, amount and customer name. It must also have a spot for the customer to sign.

1.4 This receipt is then printed out at the physical location.

1.5 Customer signs this receipt upon order pickup or delivery.

Use Case Name: Customer Database Access (u1.4)

Actors: Database, Business User

Precondition: Begins when an authorized user queries the database for a list of customers.

1.1 User clicks ‘show customers’ button to initiate the operation, wherein their authorization status is checked.

1.2.1 Exceptional Flow: If the user is not authorized to view that information, they should be given the appropriate error, and not be allowed to access the information.

1.2.2 Upon successful authorization, a spreadsheet is printed onto the screen detailing various information about each customer in the database. This information includes telephone numbers, addresses, and past orders.

Use Case Name: Menu Display (u1.5)

Actors: User, Database.

Precondition: User has clicked on the menu button.

1.1 Menu button is clicked on, taking user to new web page.

1.2 The database is queried for all items currently on the menu.

1.3 These items are then displayed for the user, with pictures and details of each item shown.

User Case Name: Adding items to cart (u1.6)

Actors: User, Database.

Precondition: User has begun adding items to their shopping cart

1.1 User clicks the add to cart button for an item in the menu GUI.

1.2 Cart is updated in the database

1.3 User clicks on cart, is then taken to a page displaying all items in it.

Use Case Name: Removing Items from cart (u1.7)

Actors: User, Database.

Precondition: User has at least one item in their shopping cart

1.1 User navigates to their shopping cart page.

1.2 User clicks the ‘remove item’ button for whichever item they wish to remove.

1.3 Web page communicates with the database to remove item from their cart.

1.4 Page refreshes to show the new, updated cart without that item.

Use Case Name: Ordering (u1.8)

Actors: User, Database, Business.

Preconditions: User has at least one item in their shopping cart. User has navigated to their shopping cart page OR clicked the ‘check out’ button on the main page.

1.1 User navigates to the checkout page.

1.2 User clicks the checkout button.

1.3 User to a new page to confirm their order. Information here includes items ordered, subtotal, tax, and total price.

1.4.1 User selects pay by credit card, and wherein use case u1.2 begins.

1.4.2 Subflow: User instead chooses either pay by cash or pay by check.

1.5 A record is created of the order in the database.

1.6 Order is then sent to kitchen for preparation.