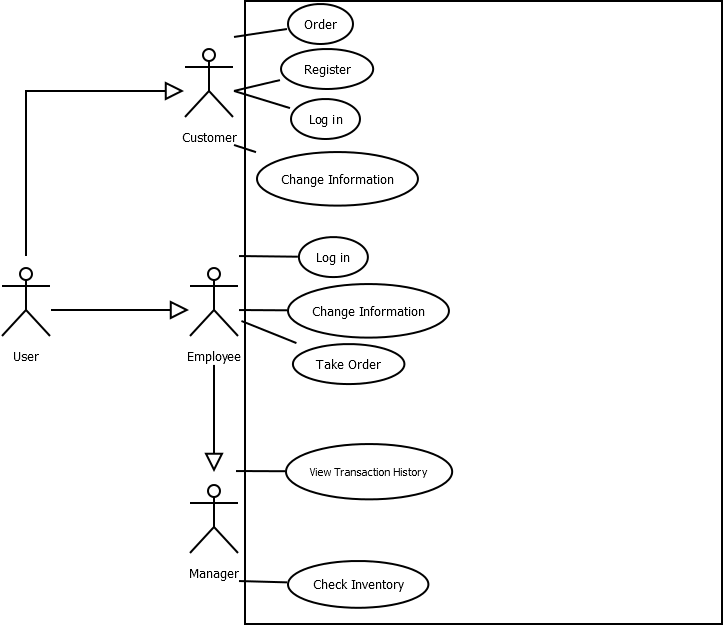
Requirements definition:

| Req. # | Input | Process | Output |
| --- | --- | --- | --- |
| #1: User Authentication | - Username  - Password | - Verify credentials  - Grant access based on role (Customer, Employee, Manager) | - Redirect to the appropriate dashboard upon successful login |
| #2: Menu Management | - Manager access to edit menu items  - Menu item details: name, description, price, customizable options | - Updates menu attributes | - Menu reflects changes for customers and employees |
| #3: Order Placement | - Menu selection  - Customization options | - Confirmation of order with timestamp and order number | - Order confirmation with details provided |
| #4: Order Processing | - Mark orders as processed | - Updates order status in real-time | - Order Status |
| #5: Inventory Management | - Manager access to track ingredient inventory  - Add to inventory | - System updates inventory with order processing | - Low stock notifications |
| #6: Customer Management | - Manager access to customer information  - Customer account creation option | - Creates Customer’s accounts for order history tracking | - Customer account creation |
| #7: Reporting and Analytics | - Managers access sales reports  - orders request to print receipts | - Gather necessary data | - Prints report or order receipt |
| #8: Payment Integration | - Card  - Cash | - Secure processing of payments | - prints out a payment receipt |
| #9: Security | - sensitive information | - Secure handling of sensitive information | - Passwords encrypted  - Protection of user data and transactions |

Requirements specification:

Use case diagram:



Use Case Flow of Events:

**1.0: Flow of events for the order use case**

**1.1 Preconditions**

The user must have already done the register or log-in use case

**1.2 Main Flow**

This use case begins after the user logs in to the restaurant system and enters his/her Information.

The user will then be able to view the menu and select items to order.

The user will be able to choose specifics for some items (extra cheese, no tomato, etc)

The user will then be able to view their receipt and confirm their order.

**1.3 Alternative Flows**

E-1: Use case for registering for an account

E-2: Use case to sign in to the website

E-3: Use case for customers to change personal information

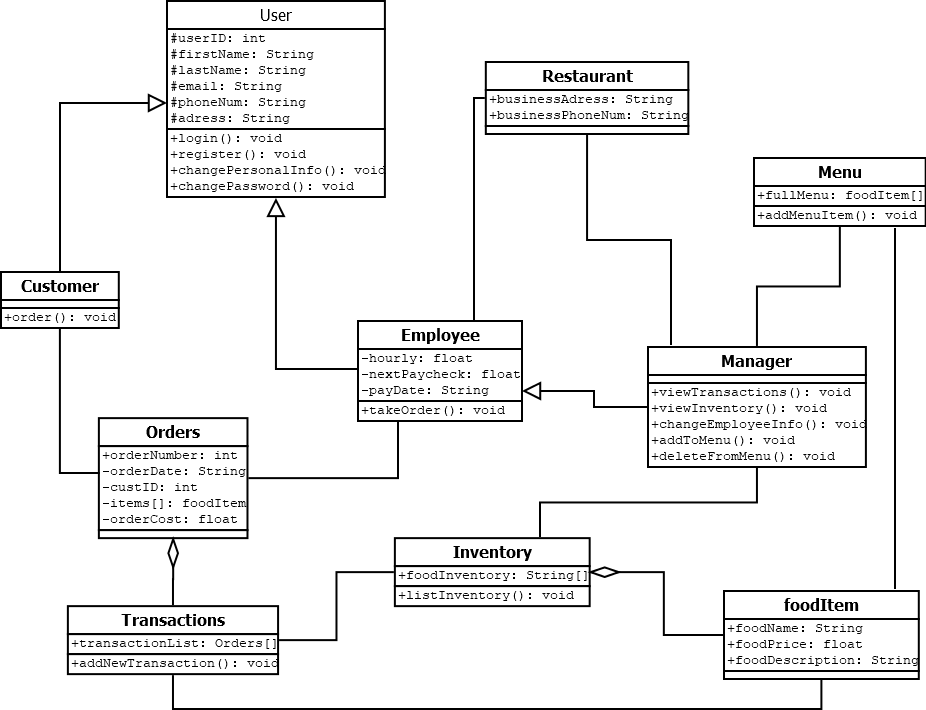
E-4: Use case for employees to change personal information

E-5: Use case to check inventory (Manager only)

E-6: Use case to view transaction history (Manager only)

E-7: Forgot password use case

Class diagrams:



Class documentation:

The **User class** contains the data for each user, both customers for the client and the client. It also has methods for logging in, registering, recovering a password, and changing the user’s information.

The **Customer class** is a child of the User class and contains a method for ordering food from the restaurant.

The **Employee class** is a child of the User class that contains additional values for an employee's payment information, tracking whether or not they are a full or part-time employee.

The **Manager class** is a child of the employee class since all managers are also employees. Managers have the ability to view past transactions, view the restaurant's inventory and add to it manually, and to change other employee’s information. The manager is also able to edit the full menu selection.

The **Menu class** saves each unique menu item and allows for addition and deletion of menu items.

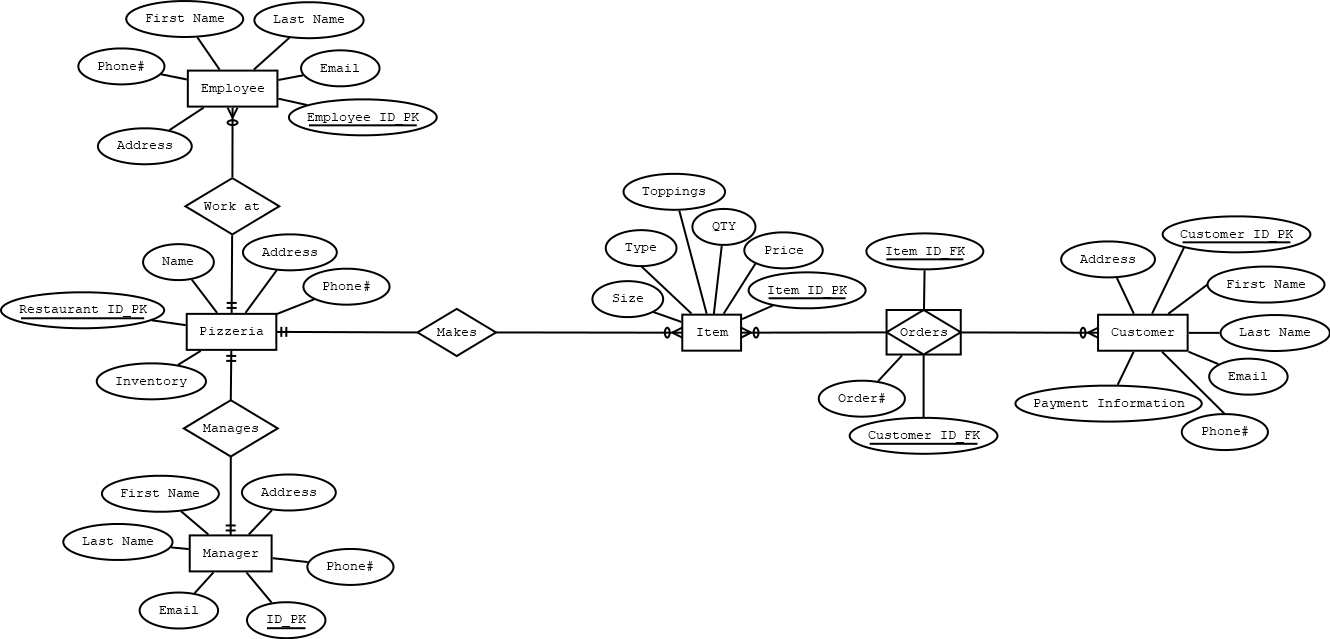
The **Restaurant class** is used to store the restaurant’s permanent information.

The **Orders class** saves the details of every order as an order object, which is saved into the **Transactions class**.

The **Inventory class** stores every individual piece of inventory in a list, which is called on in the **FoodItem class**.

The transaction class will send the menu items ordered to the FoodItem class, which will take each food item’s required ingredients out of the Inventory class.

Entity relationship diagram:



Decision table:

