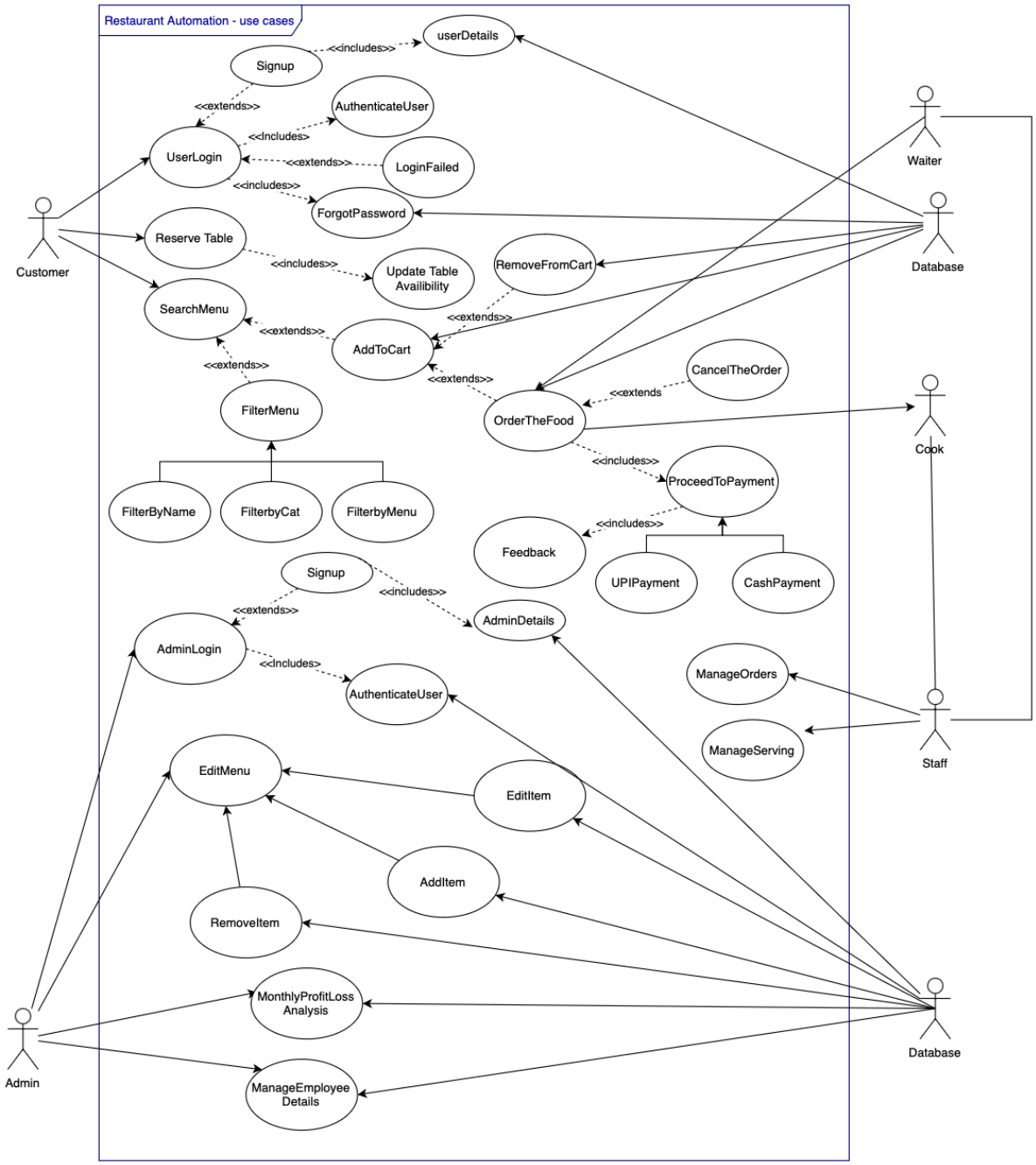
Restaurant Automation

Group-18

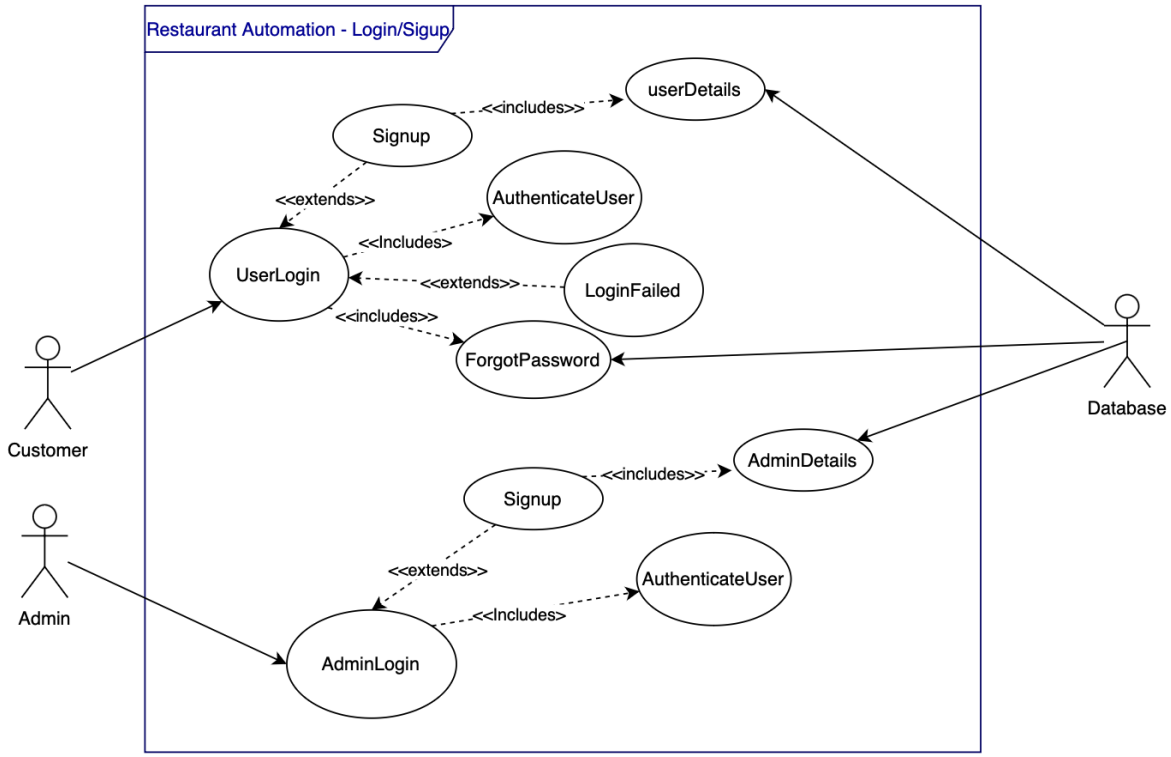
[IT314 Software Engineering]

**[Use case Diagram]**


Restaurant Automation - use cases
userDetails
<<includes>>
Signup
<<extends>>
AuthenticateUser
<<Includes>
UserLogin
<<extends>>
LoginFailed
<<includes>>
ForgotPassword
Customer
Reserve Table
<<includes>>
Update Table Availibility
SearchMenu
<<extends>>
AddToCart
<<extends>>
FilterMenu
FilterByName
FilterbyCat
FilterbyMenu
RemoveFromCart
<<extends>>
CancelTheOrder
<<extends
OrderTheFood
<<includes>>
ProceedToPayment
<<includes>>
Feedback
UPIPayment
CashPayment
Waiter
Database
Cook
Staff
ManageOrders
ManageServing
Signup
<<includes>>
<<extends>>
AdminDetails
AdminLogin
<<Includes>
AuthenticateUser
EditMenu
EditItem
AddItem
RemoveItem
MonthlyProfitLoss Analysis
ManageEmployee Details
Admin
Database

# The Use Case Description for each use case

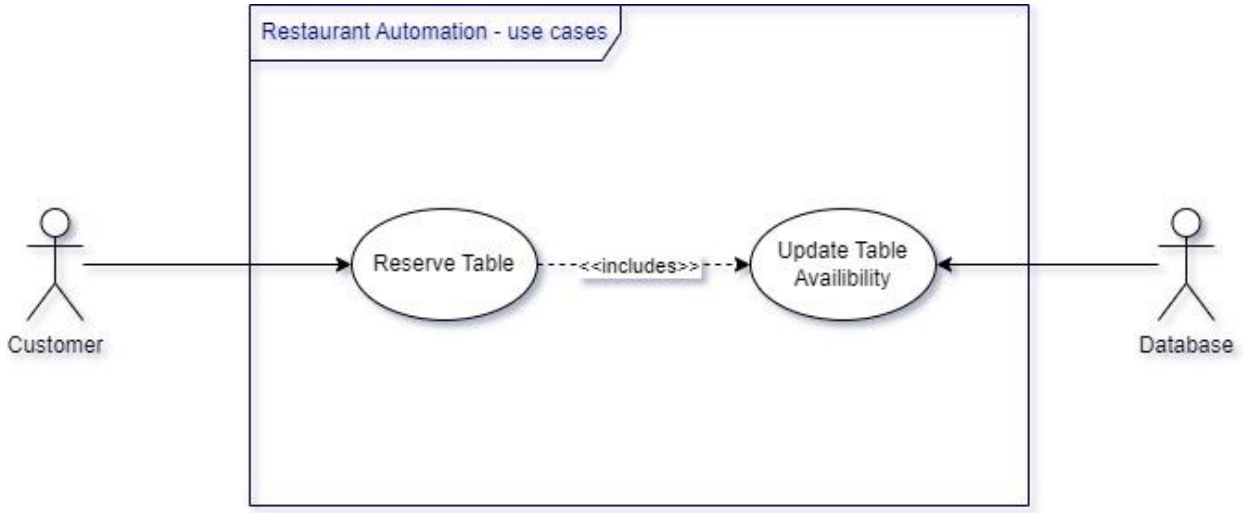
## Use Case 1: Login




| | |
|---|---|
| **Name** | Admin Login, User Login |
| **Actors** | Admin, User |
| **Goal** | To successfully get into the system and use the functionalities effectively. |
| **Trigger** | When a user wants to reserve a table. |
| **Precondition** | Admin/User account should get  identified by system. |
| **Description** | After entering the valid credentials staff and admin gets logged in to the system. |
| **Basic Flow** | 1. Actor enters the login page.<br>2. Actor gets into the login form where he enters his/her credentials.<br>3. After getting authenticated, the user gets into the website. |

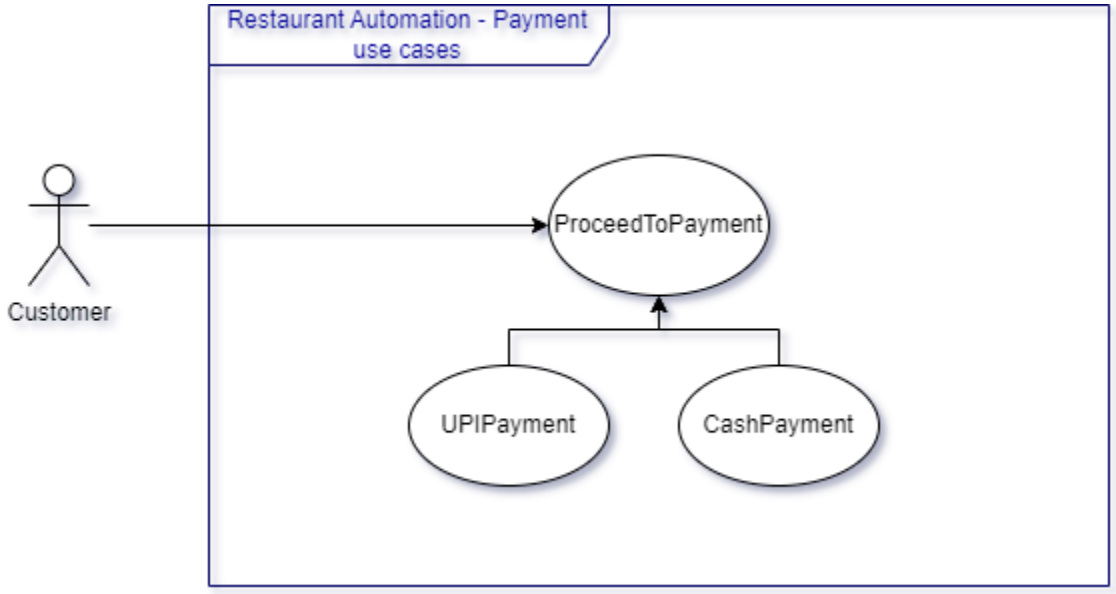
| | |
|---|---|
| **Alternative Flow** | 1.a If the user's mail Has not been verified then the system should be able to show a message and also provide features to verify the email as sign up activity.<br>2.a If the user enters the used username then the system should be able to show an error message. |
| **PostCondition** | User is logged in and is able to access all the available features. |
| **Domain Flow** | 1. If User forgets the password then the system should be able to provide the feature for resetting the password for particular user. |

## Use Case 2: Table Management




| | |
|---|---|
| **Name** | Table Management |
| **Actors** | User |
| **Goal** | To get all the table booking management transparent and neat. |
| **Trigger** | When a user wants to reserve a table. |
| **Precondition** | Users have to login to reserve a table. |
| **Description** | According to the number of members, the user will select a table with available seats. |
| **Basic Flow** | 1. Actors can enter the number of people.<br>2. Actor gets to see the available table.<br>3. System confirms and updates the table status. |
| **Alternative Flow** | 2.a The user should be able to select multiple tables according to their occupancy. |
| **PostCondition** | 1. System allocates a valid table to the user or asks the user to wait.<br>2. System updates the table status accordingly. |

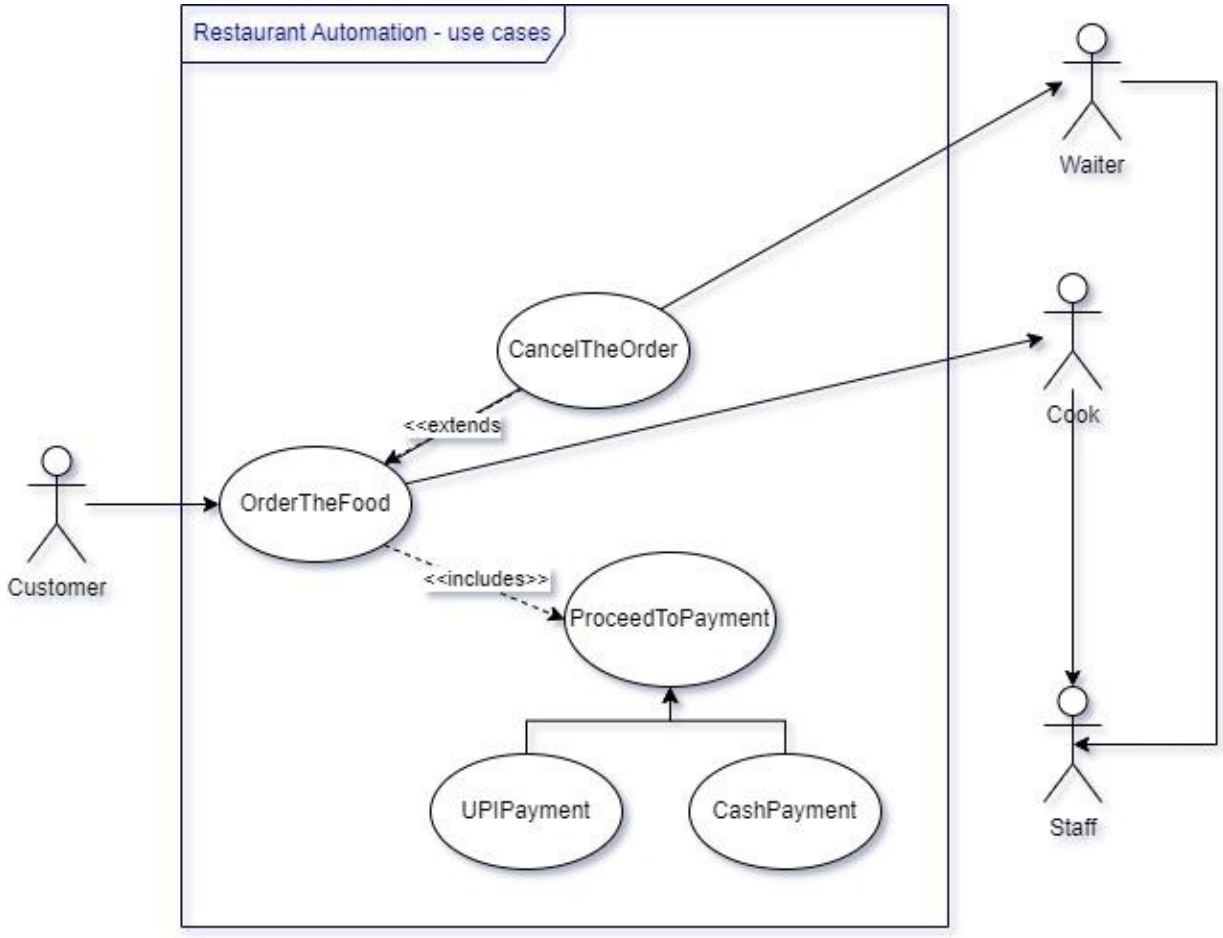
## Use case 3: Payment




| | |
|---|---|
| **Name** | Payment |
| **Actors** | Admin, User |
| **Goal** | **To successfully receive payment from customers.** |
| **Trigger** | When users hit the checkout button. |
| **Precondition** | 1. Users should select and order available food items and don't want to order anything else. |
| **Description** | After finishing the meal, User will pay the total bill via his desired payment method. |
| **Basic Flow** | 1. Users should be able to select the appropriate payment method.<br>2. Customer pays the required amount via the desired payment method.<br>3. System/Admin confirms the payment. |
| **Alternative Flow** | **1.a The system receives the wrong amount then it should show an error message.**<br>**1.b The system should be able to cancel orders.**<br>**2.b The system is not able to deliver the message to admin then users should show paid money and admin confirms manually.** |

| | |
|---|---|
| **PostCondition** | 1. Users should receive the ordered food item after successful payment. |

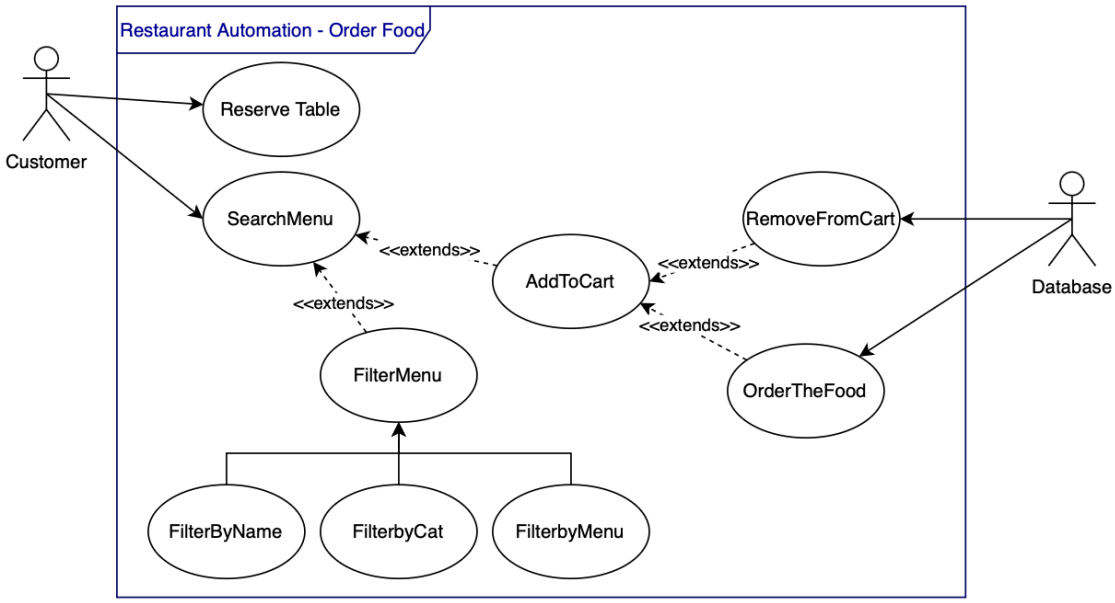
## Use case 4: Order management




| | |
|---|---|
| **Name** | Order Management |
| **Actors** | Chef, Servant, User |
| **Goal** | To efficiently process customer orders and ensure timely delivery of food ordered along with tracking the order. |
| **Trigger** | When a user clicks on order the food and pays for the food successfully. |
| **Precondition** | 1. The menu items should be correctly selected.<br>2. User should have paid for the food. |
| **Description** | Delivering food from chef to user. |

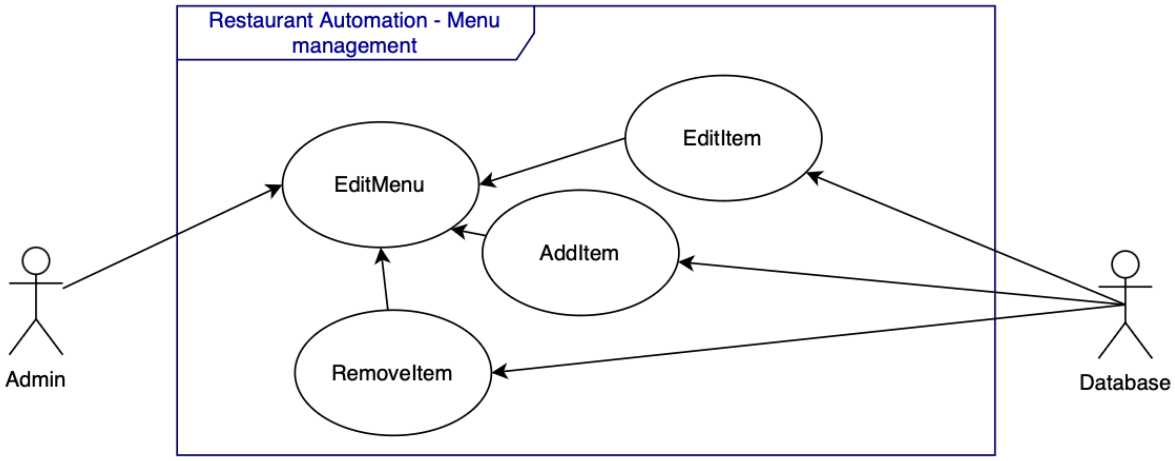
| | |
|---|---|
| **Basic Flow** | 1. After receiving notification from the user, the chef starts to prepare the order.<br>2. After the food gets prepared, the servant brings the food to the appropriate table number. |
| **Alternative Flow** | **2.a If the chef did not receive the order/See the mail then the user should contact the admin with the bill.** |
| **PostCondition** | 1. The customer receives the correct food on time. |

## Use case 5: Order Food



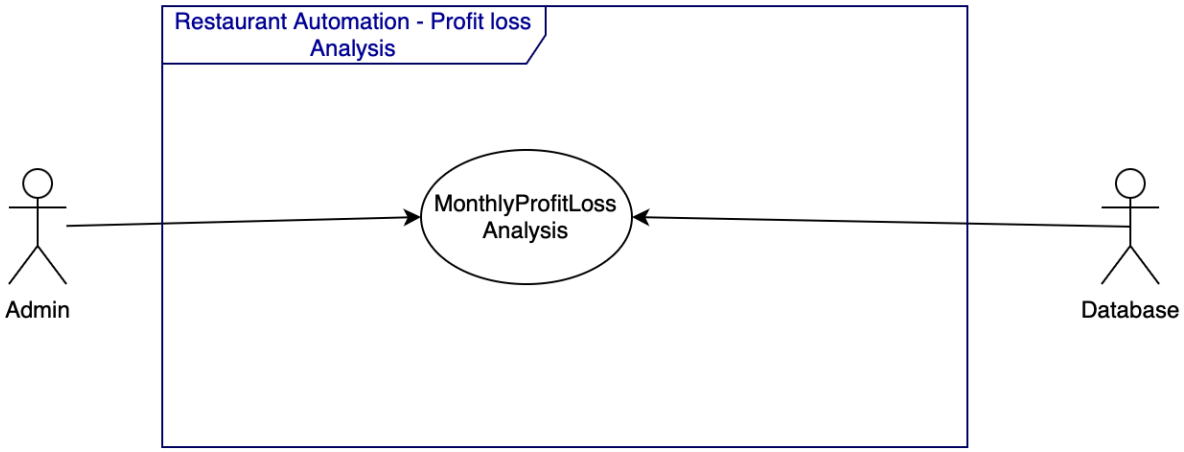

| | |
|---|---|
| **Name** | Order Food |
| **Actors** | User |
| **Goal** | To provide the desired food to users. |
| **Trigger** | When the user selects the food items to order from the menu. |
| **Precondition** | 1. Users should have logged in to the website.<br>2. Users should have selected the available items. |
| **Description** | User will add the desired amount of food to the cart. |
| **Basic Flow** | 1. User sees through the menu with filtering options.<br>2. User adds the desired food items with desired quantity into the cart. |
| **Alternative Flow** | 2.a The system shows items which are not available then it should show an error message. |
| **PostCondition** | 1. The customer receives the correct food on time. |

## Use case 6: Menu management




| | |
|---|---|
| **Name** | Menu Management |
| **Actors** | Admin |
| **Goal** | To successfully add a new item or remove the existing item or to change the item price or description in the existing menu. |
| **Trigger** | When the user selects the food items to order from the menu. |
| **Precondition** | 1. Admin should have logged into the system as an Admin. |
| **Description** | Admin will be able to review menu items and can add or remove food items according to the analysis. |
| **Basic Flow** | 1. User sees through the menu with filtering options.<br>2. User adds the desired food items with desired quantity into the cart. |
| **Alternative Flow** | **2.a If the system fails to edit inventory then it should show an error message.** |
| **PostCondition** | 1. Menu should be updated as needed. |

## Use Case 7 : Profit loss analysis




| | |
|---|---|
| **Name** | Profit loss analysis |
| **Actors** | Admin |
| **Goal** | To generate the report on sales and to optimize the management and services accordingly. |
| **Description** | Admin will analyze sales reports and user reviews to provide better experience. |

## Non functional requirements:

Non-functional requirements are attributes of a system that describe how the system should behave, rather than what it should do. Here are some non-functional requirements for a restaurant automation system:

- **Security:** The system should be secure and protect sensitive information from unauthorized access and data breaches. Only authorized administrators should have access to menu management, accounting, inventory, etc. No third-party access should be allowed.

- **Reliability:** The system should be highly reliable and consistently available, with minimal downtime and technical issues. The feedback mechanism should promptly report any errors to the admin for immediate resolution.

- **User-Friendliness:** The system should feature a user-friendly interface that requires minimal training for restaurant owners, admins, and staff to navigate and use effectively.

- **Responsiveness:** The system should be responsive across various platforms, ensuring that it functions well regardless of the device or operating system it is running on. It should be platform-independent.

- **Scalability:** The system should be scalable and flexible to accommodate the growing needs of the restaurant over time. It should efficiently handle concurrent food orders and dynamically adjust to varying workloads.

- **Performance:** The system should deliver fast and efficient performance, with quick load times and minimal lag. It should effectively manage workload and handle concurrent orders without compromising speed and efficiency.

- **Support:** The system should have a dedicated support team that is responsive and knowledgeable, offering technical assistance and updates as required. The feedback system should be directly connected to the support team and admin to prioritize and resolve issues promptly.

## Functional requirements:

- **Order Taking:** The system should allow customers to take orders from the customers.

- **Table Booking:** Customers should be able to book tables according to the requirements and number of members.

- **Menu Management:** The system should allow admin to manage and update their menu, including the ability to add, edit, and remove menu items.

- **Payment Processing:** The system should allow customers to make payments through various methods, such as cash, or mobile payment, and provide the restaurant with real-time payment tracking and reporting.

- **Online Accessibility:** The system should be accessible online allowing customers to place orders, make payments, and receive updates and notifications.

- **Reporting and Analytics:** The system should provide restaurants with reporting and analytics capabilities, including sales reports, and performance metrics.