**NLP Chatbot Development Using Dialogflow**

**Design Document**

**Version 1.0**



**Group Id: F24PROJECT88F2E (BC210414987)**

**Supervisor Name: Abdullah Qamar**

**Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| **Date (dd/mm/yyyy)** | **Version** | **Description** | **Author** |
| 27/2/2025 | 1.0 | I made this project to develop an **AI-powered chatbot** using **Dialogflow** for the restaurant industry. The chatbot is designed to enhance customer interactions by handling table reservations, taking orders, providing menu details, answering frequently asked questions, and offering customer support. With **Natural Language Processing (NLP)**, the chatbot engages users in human-like conversations, ensuring smooth, efficient, and personalized interactions.  By automating routine tasks, the chatbot reduces manual workload, improves service speed, and boosts customer satisfaction. **Chatbots** are transforming customer service by providing instant responses, 24/7 availability, and reducing human effort. In the restaurant industry, they play a crucial role in enhancing customer experience by offering quick, accurate responses, automating orders and reservations, and allowing staff to focus on more critical tasks.  Furthermore, chatbots improve efficiency by minimizing wait times, handling multiple customers simultaneously, and driving sales through personalized recommendations. By integrating this chatbot, restaurants can streamline operations, enhance customer engagement, and remain competitive in the digital era, making service more efficient, responsive, and convenient for customers. | BC210414987 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**Table of Contents**

1. [Introduction of Design Document](#One)
2. [Entity Relationship Diagram (ERD)](#ERD)
3. [Sequence Diagrams](#Six)
4. [Architecture Design Diagram](#Seven)
5. [Class Diagram](#class)
6. [Database Design](#databasedesign)
7. [Interface Design](#interfacedesign)
8. [Test Cases](#testcases)

**1. Introduction of Design Document**

In this design document, I have included several key components to outline the system’s design and architecture, providing a comprehensive blueprint for the development of the AI-powered chatbot.

**Entity Relationship Diagram (ERD)**

The Entity Relationship Diagram (ERD) is one of the primary components, representing the database structure and showing how entities such as users, orders, and reservations are related to each other. This diagram helps ensure that the data is structured in a way that promotes smooth data flow and maintains integrity throughout the system. By clearly defining the relationships between these entities, it assists in organizing the database to handle queries and operations efficiently.

**Sequence Diagrams**

The Sequence Diagrams serve another crucial purpose by illustrating the interaction between users and the system over time. These diagrams highlight the sequence of events during various processes like placing an order or making a reservation. They are essential for understanding how the system behaves in real-time and how different components of the chatbot interact with one another. This is critical for visualizing the flow of user requests and system responses, providing insights into how data is processed step by step.

**Architecture Design Diagram**

The Architecture Design Diagram provides a high-level view of the system, illustrating the components, their relationships, and the technologies used across the backend, frontend, and database layers. This diagram helps visualize how the different parts of the chatbot system will communicate with each other and interact within the overall architecture. It acts as a roadmap to guide the development process, ensuring that all system components fit together smoothly and efficiently.

**Class Diagram**

The Class Diagram focuses on breaking down the system into manageable and reusable components by outlining the various classes, their attributes, and methods. It helps in understanding how each class will function and how they will work together within the overall system. By identifying the relationships and behaviors of the system's objects, this diagram makes the development process more modular and scalable.

**Database Design**

The Database Design section goes into detail about the structure of the database, defining tables, columns, and relationships. This ensures that data is stored and retrieved efficiently, with the relationships between different pieces of information clearly defined. A well-organized database design is essential for ensuring that the chatbot can process queries and handle user interactions effectively.

**Interface Design**

The Interface Design section focuses on the user interface components, ensuring that the chatbot is intuitive and user-friendly. This section outlines how the user will interact with the system and what their experience will be like. It considers user needs and ensures that the chatbot’s interface is simple, responsive, and accessible, ultimately enhancing the customer experience.

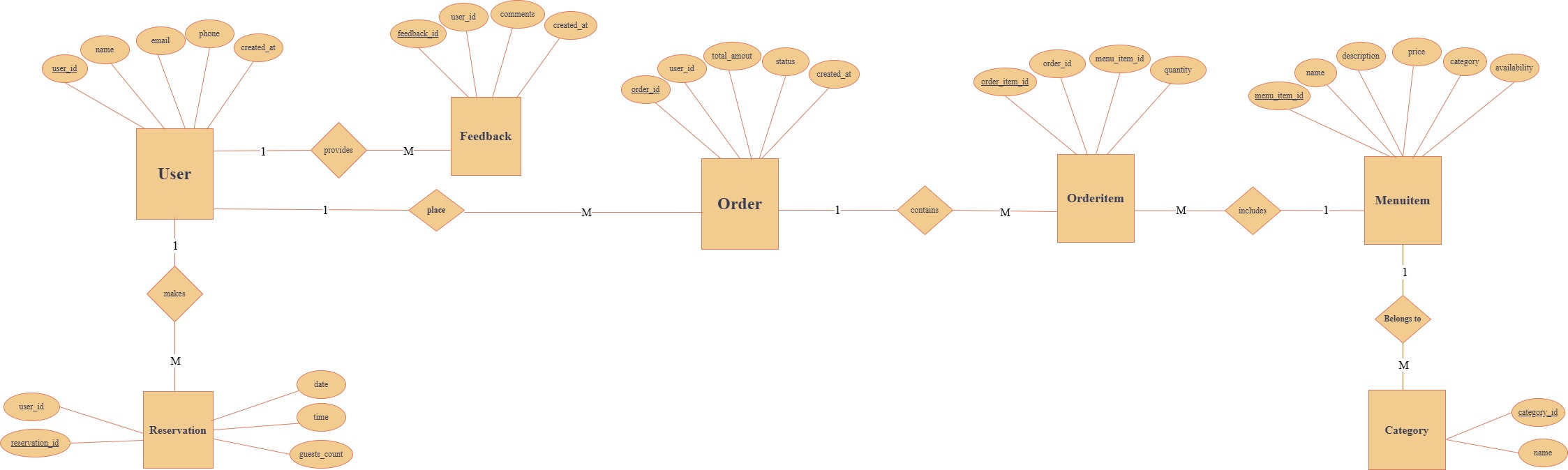
**Test Cases**

Finally, the Test Cases section provides a detailed testing strategy that outlines different scenarios to ensure that the system works as expected. By testing each component and functionality, the system can be validated to meet the defined requirements and ensure that it performs as intended. This section ensures that potential issues are identified early in the development process, reducing the risk of errors in the final product.

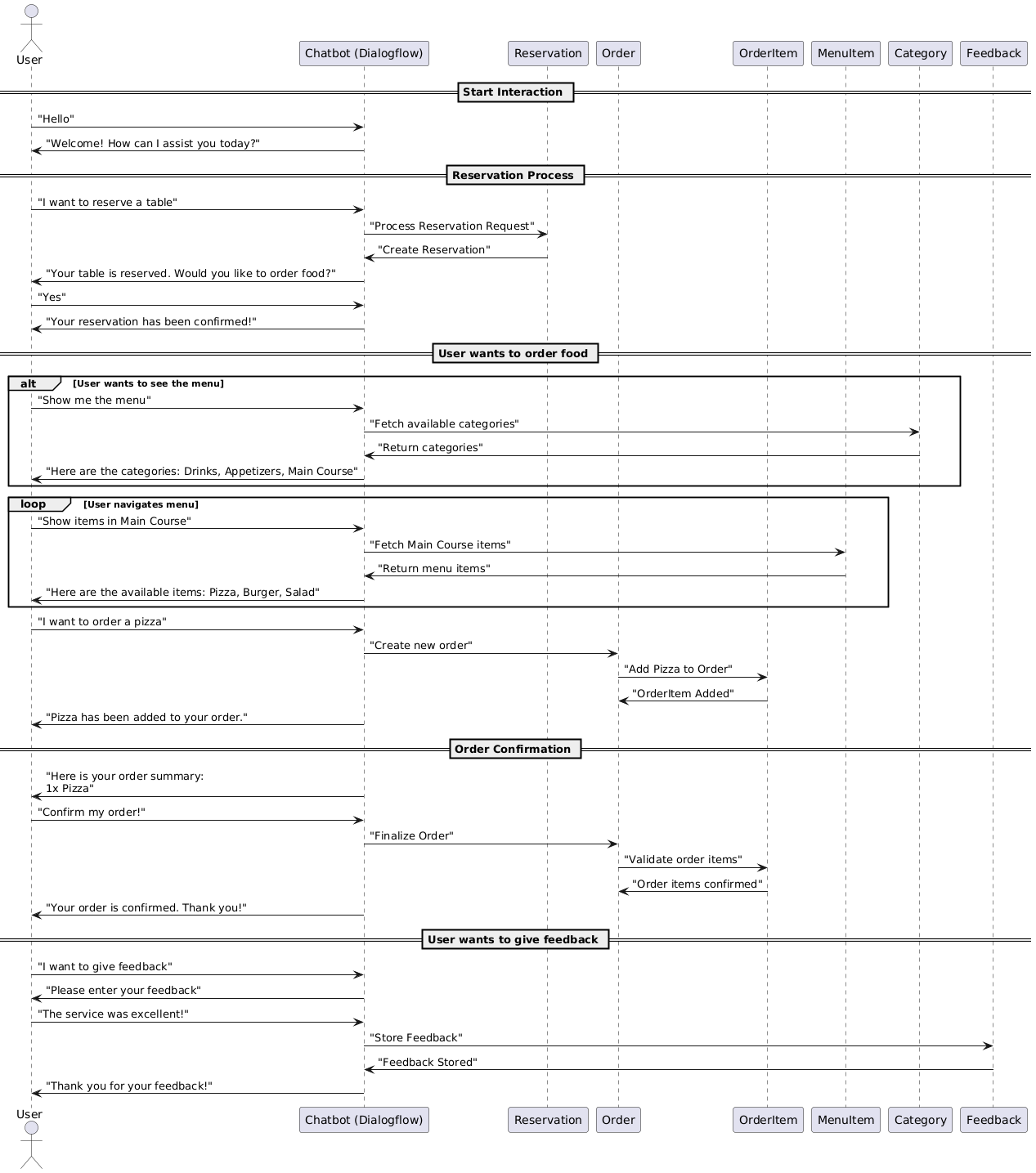
**Purpose of the Design Phase**

The purpose of this design phase is to provide a detailed plan that guides the development of the system. It ensures that the architecture is well-organized, the database is structured efficiently, and the user interface is intuitive. This phase is vital for laying a strong foundation for the project by clarifying the system’s components and how they will interact. By defining these elements early, the design document minimizes risks, enhances clarity for the development team, and streamlines the overall development process. It also provides a clear roadmap for the entire software development lifecycle, ensuring the system is built according to the specified requirements and is executed efficiently.

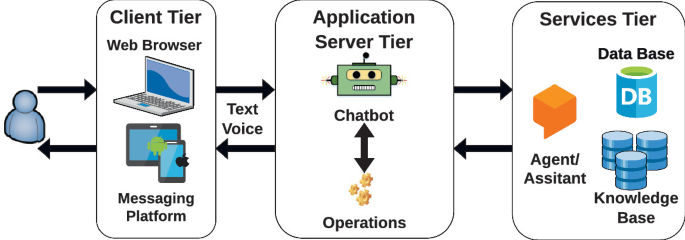
1. Entity Relationship Diagram (ERD) (To be developed using Microsoft Visio or any other drawing software of your choice)



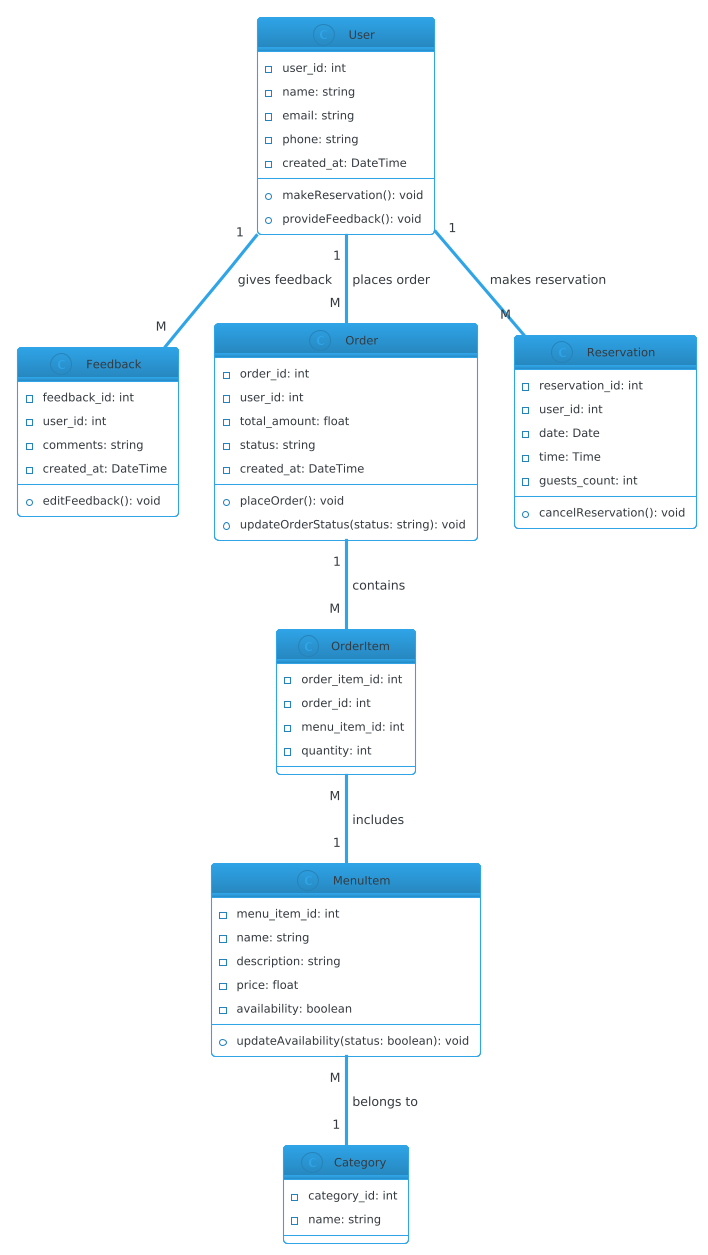
1. Sequence Diagrams (To be developed using Rational Rose or any other drawing software of your choice)



1. Architecture Design Diagram



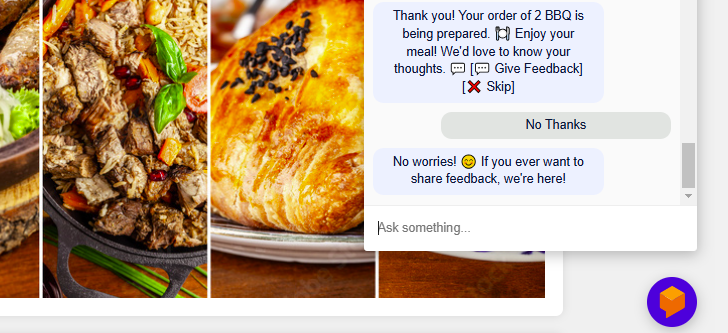
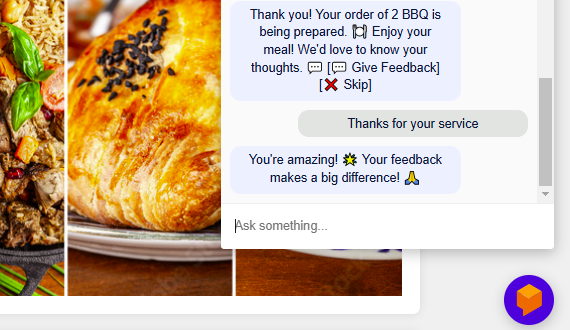
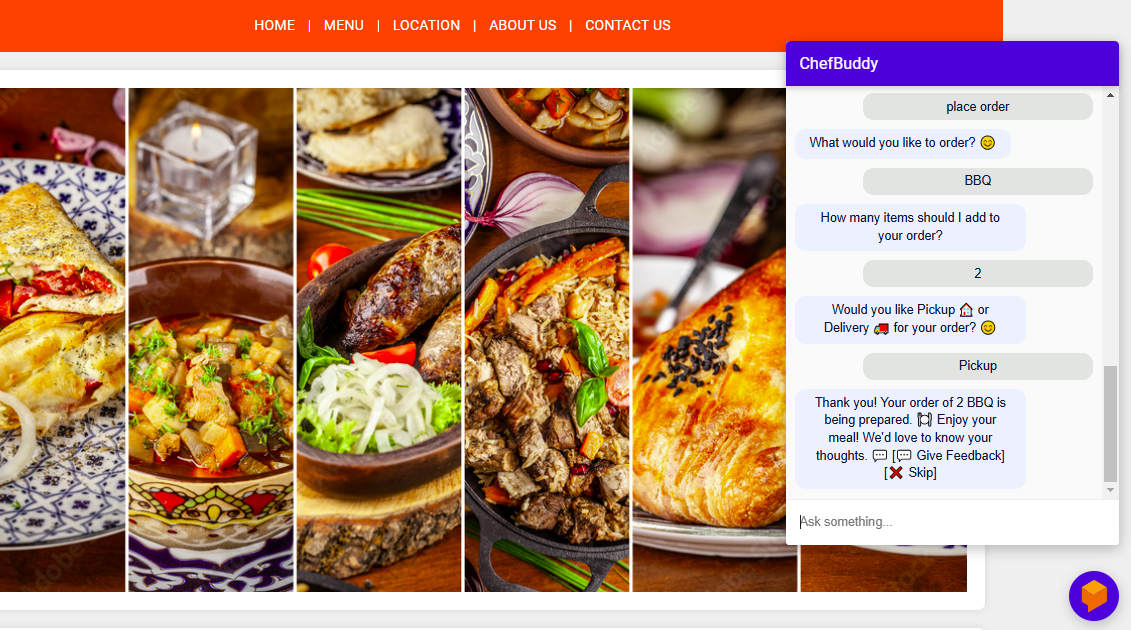
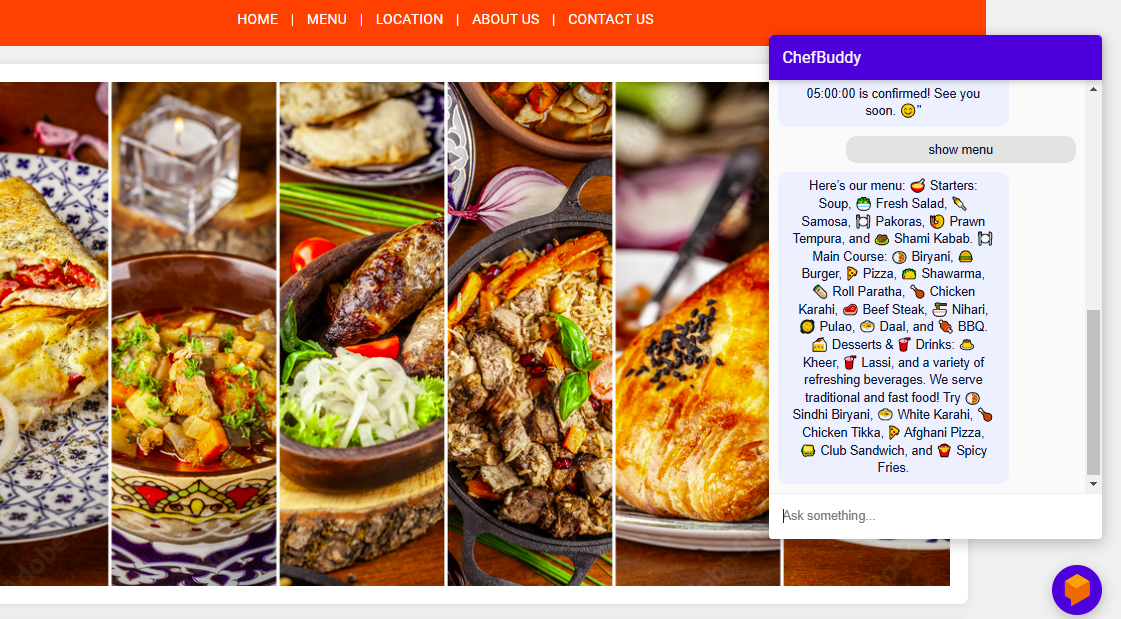
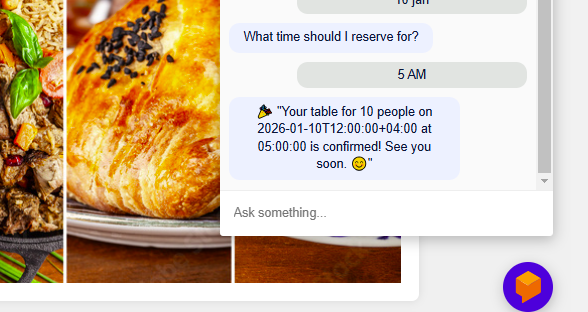
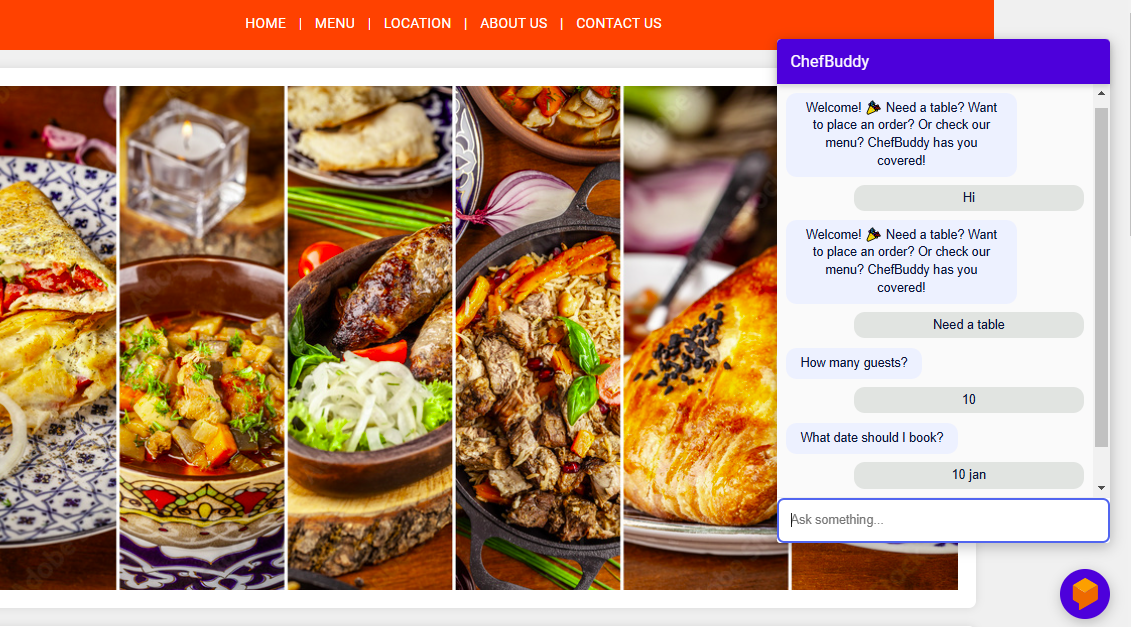
1. Class Diagram



1. Database Design



1. Interface Design



1. Test Cases

**TC001 - User Registration (Valid Data)**

| **Field** | **Description** |
| --- | --- |
| **Test Case ID** | TC001 |
| **Description** | Verify that a user can register successfully with valid details. |
| **Input** | Name, Email, Phone, Password |
| **Expected Output** | User registered successfully. |
| **Pass/Fail** | Pass |
| **Tested By** | BC21041987 |

**TC002 - User Login (Valid Credentials)**

| **Field** | **Description** |
| --- | --- |
| **Test Case ID** | TC002 |
| **Description** | Ensure users can log in with valid credentials. |
| **Input** | Email, Password |
| **Expected Output** | Login successful. |
| **Pass/Fail** | Fail |
| **Tested By** | BC21041987 |

**TC003 - Make a Reservation (Valid Data)**

| **Field** | **Description** |
| --- | --- |
| **Test Case ID** | TC003 |
| **Description** | Verify that a user can successfully make a reservation. |
| **Input** | Date, Time, Guests Count, User ID |
| **Expected Output** | Reservation confirmed. |
| **Pass/Fail** | Pass |
| **Tested By** | BC21041987 |

**TC004 - Cancel Reservation**

| **Field** | **Description** |
| --- | --- |
| **Test Case ID** | TC004 |
| **Description** | Ensure a user can cancel their reservation. |
| **Input** | Reservation ID |
| **Expected Output** | Reservation canceled. |
| **Pass/Fail** | Fail |
| **Tested By** | BC21041987 |

**TC005 - Place an Order (Valid Data)**

| **Field** | **Description** |
| --- | --- |
| **Test Case ID** | TC005 |
| **Description** | Verify that a user can place an order successfully. |
| **Input** | User ID, Items, Quantity, Payment Details |
| **Expected Output** | Order placed successfully. |
| **Pass/Fail** | Pass |
| **Tested By** | BC21041987 |

**TC006 - Cancel Order**

| **Field** | **Description** |
| --- | --- |
| **Test Case ID** | TC006 |
| **Description** | Ensure users can cancel their order before processing starts. |
| **Input** | Order ID |
| **Expected Output** | Order canceled. |
| **Pass/Fail** | Fail |
| **Tested By** | BC21041987 |

**TC007 - Add Item to Order**

| **Field** | **Description** |
| --- | --- |
| **Test Case ID** | TC007 |
| **Description** | Verify that users can add an item to their order. |
| **Input** | Order ID, Item ID, Quantity |
| **Expected Output** | Item added to order. |
| **Pass/Fail** | Pass |
| **Tested By** | BC21041987 |

**TC008 - Remove Item from Order**

| **Field** | **Description** |
| --- | --- |
| **Test Case ID** | TC008 |
| **Description** | Ensure users can remove an item from their order. |
| **Input** | Order ID, Item ID |
| **Expected Output** | Item removed from order. |
| **Pass/Fail** | Pass |
| **Tested By** | BC21041987 |

**TC009 - Submit Feedback**

| **Field** | **Description** |
| --- | --- |
| **Test Case ID** | TC009 |
| **Description** | Verify that a user can submit feedback for an order. |
| **Input** | Order ID, Comments |
| **Expected Output** | Feedback submitted successfully. |
| **Pass/Fail** | Pass |
| **Tested By** | BC21041987 |

**TC010 - Add New Category**

| **Field** | **Description** |
| --- | --- |
| **Test Case ID** | TC010 |
| **Description** | Verify that an admin can add a new category. |
| **Input** | Category Name, Description |
| **Expected Output** | Category added successfully. |
| **Pass/Fail** | Pass |
| **Tested By** | BC21041987 |

**TC011 - Delete Category (No Items)**

| **Field** | **Description** |
| --- | --- |
| **Test Case ID** | TC011 |
| **Description** | Verify that an admin can delete a category if it has no items. |
| **Input** | Category ID (No items) |
| **Expected Output** | Category deleted successfully. |
| **Pass/Fail** | Fail |
| **Tested By** | BC21041987 |

**TC012 - Add New Menu Item**

| **Field** | **Description** |
| --- | --- |
| **Test Case ID** | TC012 |
| **Description** | Verify that an admin can add a new menu item under a category. |
| **Input** | Name, Description, Price, Availability, Category ID |
| **Expected Output** | Menu item added successfully. |
| **Pass/Fail** | Pass |
| **Tested By** | BC21041987 |

**TC013 - Update Menu Item Details**

| **Field** | **Description** |
| --- | --- |
| **Test Case ID** | TC013 |
| **Description** | Ensure that an admin can update menu item details. |
| **Input** | Menu Item ID, Updated Details |
| **Expected Output** | Menu item updated successfully. |
| **Pass/Fail** | Pass |
| **Tested By** | BC21041987 |

**TC014 - Delete Menu Item**

| **Field** | **Description** |
| --- | --- |
| **Test Case ID** | TC014 |
| **Description** | Ensure that a menu item can be deleted only if it is not linked to any order. |
| **Input** | Menu Item ID (Not in Orders) |
| **Expected Output** | Menu item deleted successfully. |
| **Pass/Fail** | Fail |
| **Tested By** | BC21041987 |