< NLP Chatbot Development using Dialogflow>

**Software Requirements Specification**

Version 1.0



**Group Id: < F24PROJECTF99DD >**

**Supervisor Name :<** Abdullah Qamar  **>**

**Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| **Date (dd/mm/yyyy)** | **Version** | **Description** | **Author** |
| 3/12/2024 | 1.0 | In this project, we will develop an AI-powered chatbot using Google Dialogflow to automate customer interactions and enhance user experience. The chatbot will address specific business needs in one of three industries: Training Company, Pharmacy Store, or Restaurant. By leveraging natural language processing, the chatbot will effectively understand and respond to user inquiries, perform key tasks such as reservations or order processing, and improve operational efficiency. This project aims to demonstrate the practical application of NLP in creating interactive and user-friendly web applications. | Bc230211251 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**Table of Contents**

1. [Scope (of the project)](#scope)
2. [Functional Requirements Non Functional requirements](#FRNFR)
3. [Use Case Diagram](#UCD)
4. [Usage Scenarios](#UCS)
5. [Adopted Methodology](#Adopted)
6. [Work Plan (Use MS Project to create Schedule/Work Plan)](#Gantt)

**SRS Document**

Scope of Project:

Purpose:

The primary purpose of this project is to develop an AI-powered chatbot using Google Dialogflow to automate customer interactions and enhance user experience in a restaurant setting. The chatbot will leverage natural language processing to understand and respond to customer inquiries effectively, thereby improving operational efficiency and customer satisfaction.

Intended Functionalities:

1.

User Interaction: The chatbot will engage customers in a conversational manner, providing relevant responses to their queries about the restaurant.

2.

Reservation Management: It will handle table reservations, allowing customers to book tables seamlessly.

3.

Order Processing: The chatbot will assist in taking orders, either for dine-in or takeout, ensuring accuracy and efficiency.

4.

Menu Navigation: It will help customers navigate the menu, providing details about dishes, ingredients, and specials.

5.

Customer Support: The chatbot will address frequently asked questions (FAQs) related to the restaurant's services, hours of operation, and policies.

Specific Tasks:

• Table Reservations: Allow customers to check availability and book tables.

• Order Taking: Facilitate the process of placing orders, including customization options.

• Menu Assistance: Provide detailed information about menu items, including dietary options and specials.

• Customer Inquiries: Answer common questions about the restaurant, such as location, hours, and services offered.

Scope Boundaries:

• Included: The project will focus on developing a functional chatbot that can handle the specified tasks within the restaurant domain. It will include the design, development, and integration of the chatbot with the restaurant's backend system.

• Excluded: The project will not cover advanced AI features such as sentiment analysis, multi-language support, or complex decision-making processes. It will also not include extensive user interface design beyond basic interaction capabilities.

Project Overview:

This project aims to demonstrate the practical application of NLP in creating an interactive and user-friendly chatbot for a restaurant. By automating routine tasks such as reservations, order processing, and customer inquiries, the chatbot will enhance operational efficiency and improve the overall customer experience. The scope boundaries ensure a clear focus on essential functionalities while excluding advanced features that are beyond the project's initial scope.

Functional and non Functional Requirements:

Functional Requirements

User Interaction**:**

• The chatbot will engage users in a conversational manner, providing relevant responses to their queries about the restaurant.

• It will guide users through various options and help them find the information they need.

Reservation Management:

• The chatbot will handle table reservations, allowing customers to check availability and book tables.

• It will confirm reservations and send reminders to customers.

Order Processing:

•The chatbot will assist in taking orders for dine-in or takeout, ensuring accuracy and efficiency.

• It will allow customers to customize their orders and provide real-time updates on order status.

Menu Navigation:

• The chatbot will help customers navigate the menu, providing details about dishes, ingredients, and specials.

• It will suggest popular items and accommodate dietary preferences.

Customer Support:

• The chatbot will address frequently asked questions (FAQs) related to the restaurant's services, hours of operation, and policies.

• It will handle customer complaints and feedback, escalating issues to human staff when necessary.

Non -Functional Requirements

Performance:

• The chatbot should respond to user queries within 2 seconds to ensure a smooth user experience.

• It should handle multiple concurrent users without performance degradation.

Reliability:

• The chatbot must be available 99.9% of the time, ensuring it is accessible to users whenever needed.

• It should have mechanisms to recover from failures and maintain data integrity.

Usability:

• The chatbot interface should be intuitive and easy to use, requiring minimal user training.

• It should provide clear and concise responses, avoiding technical jargon.

Scalability:

• The system should be able to scale to accommodate increasing numbers of users and interactions.

• It should support the addition of new features and functionalities without significant rework.

Security:

• The chatbot must ensure the privacy and security of user data, complying with relevant data protection regulations.

• It should implement authentication and authorization mechanisms to prevent unauthorized access.

Activities to Fulfill Non-Functional Requirements

Performance Testing:

• Conduct load testing to ensure the chatbot can handle high volumes of traffic.

• Optimize the backend infrastructure to reduce response times.

Reliability Measures:

• Implement redundancy and failover mechanisms to ensure high availability.

• Regularly back up data and perform disaster recovery drills.

Usability Enhancements:

• Conduct user testing to gather feedback and improve the chatbot interface.

• Provide user guides and tutorials to help users navigate the chatbot.

Scalability Planning:

• Design the system architecture to support horizontal scaling.

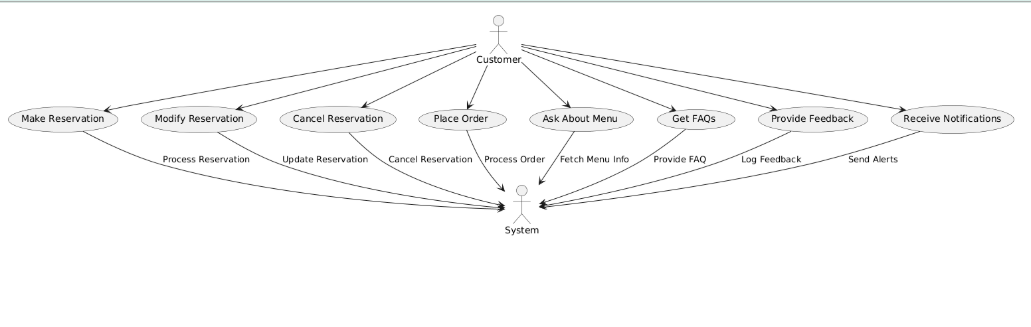
• Use cloud services to dynamically allocate resources based on demand.

Security Protocols:

• Implement encryption for data in transit and at rest.

• Regularly update and patch the system to protect against vulnerabilities.

Use Case Diagram(s):



Usage Scenarios

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Use case title | Use case ID | Actions | Description | Alternatives path | Pre-condition | Post-condition | Author | Exceptions |
| Make reservation | UC-001 | 1.initiate  2.input  Date/time  3.confirm | Customer books a table for a specific time. | Suggests alternatives if unavailable. | Customer knows reservation policy. | Reservation confirmed | Team project | Network issues may hinder confirmation. |
| Modify reservation | UC-002 | 1.Request modify  2.input new date/time | Customer changes an existing reservation. | suggests alternatives if new time is full | Existing reservation must exist. | Reservation updated |  | Invalid data entered during modification. |
| Cancel reservation | UC-003 | 1. Request cancel 2. Confirm cancellation | Customer cancels an existing reservation. | Inform if reservation not found. | Existing reservation must exist. | Reservation successfully canceled. |  | Network issues may hinder cancellation. |
| Place order | UC-004 | 1. Start order 2. Select items 3. Confirm | Customer places an order from the menu | Suggests  Alternatives if an item is out. | Menu should e available. | Order successfully placed. |  | Invalid item selection or unavailability. |
| Ask about menu | UC-005 | 1. Ask about menu 2. Retrieve details | Customer inquires about menu items. | Inform about specials or unavailability. | Menu must be up-to-date. | Accurate menu info provided. |  | Database retrieval error. |
| Get FAQs | UC-006 | 1. Ask question 2. Retrieve answer | Customer asks general questions. | Refer to support if not found. | FAQs must be defined | Satisfactory answer given. |  | FAQ database retrieval issue. |
| Provide feedback | UC-007 | 1. Start feedback 2. Input feedback | Customer submits feedback about their dining. | Prompt for missing info if incomplete. | Recent dining experience required. | Feedback logged successfully. |  | Inappropriate content or format issues. |
| Receive notifications | UC-008 | 1. Subscribe to alerts 2. Send notification | Customer receives updates related to reservations. | Opt-out available. | Permission granted for notifications. | User informed about updates. |  | Settings may fail to save. |

Adopted Methodology

1. Planning and Requirement Analysis:

• Agile Methodology: Use Agile practices for iterative planning and requirement gathering. Agile allows for flexibility and continuous feedback, ensuring that the project adapts to changing requirements and stakeholder needs.

2. Design:

• Unified Modeling Language (UML): Utilize UML for creating detailed design diagrams. UML helps in visualizing the system architecture and design, making it easier to understand and communicate complex structures.

3. Implementation:

• Waterfall Model: For the implementation phase, follow a structured approach similar to the Waterfall model. This ensures that each phase is completed before moving on to the next, providing a clear and organized development process.

4. Testing:

• V-Model: Integrate the V-Model for testing, which emphasizes verification and validation. This model ensures that each development stage has a corresponding testing phase, improving the overall quality and reliability of the system.

5. Deployment:

• DevOps Practices: Adopt DevOps practices for continuous integration and continuous deployment (CI/CD). This approach facilitates automated deployment and monitoring, ensuring that the system is always in a deployable state.

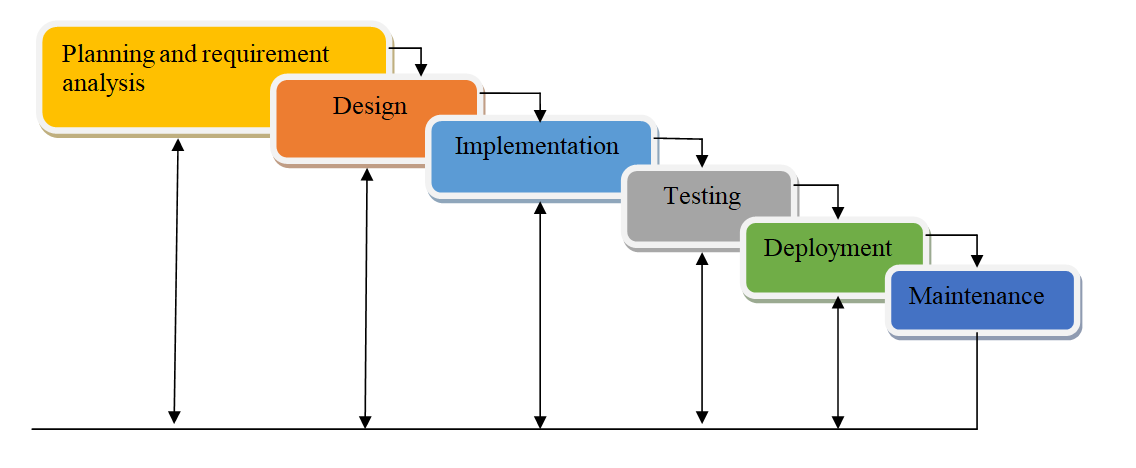
6. Maintenance:

• Iterative Model: Use the Iterative model for maintenance, allowing for regular updates and improvements based on user feedback and performance metrics.

Combining Methodologies:

Combining these methodologies involves careful planning and coordination. Here are some steps to ensure a smooth integration:

1. Define Objectives: Clearly outline the goals and benefits of combining different models.
2. Select Suitable Models: Choose models that complement each other and address the specific needs of your project.
3. Establish Guidelines: Create guidelines for how and when to apply each model during the project lifecycle.
4. Train Team Members: Ensure that all team members are familiar with the chosen methodologies and understand how to implement them effectively.
5. Monitor and Adjust: Continuously monitor the project’s progress and make adjustments as needed to optimize the combination of methodologies.



Work Plan (Use MS Project to create Schedule/Work Plan)

