Task Definition – Oorja Dorkar

Use Case: Customer Support Chatbot

Objective:
The goal of this project is to design and implement a customer support chatbot using Azure's prompt flow tools. The chatbot will assist users with common queries, such as password resets, account information, and troubleshooting issues. The expected outcomes include improved customer satisfaction, reduced response times, and streamlined support operations.

Prompt Flow Design

Components of the Prompt Flow:

1. **Input Nodes:**
    - Capture user queries (e.g., "How do I reset my password?").
    - Preprocess inputs to remove irrelevant text and standardize formats.
2. **Model Nodes:**
    - Use a pre-trained LLM (e.g., GPT-4) to generate responses based on structured prompts.
    - Example prompt: "You are a customer support assistant. Provide a step-by-step solution to reset a password."
3. **Output Nodes:**
    - Deliver structured responses to users (e.g., "To reset your password, go to the login page and click 'Forgot Password.' Follow the instructions sent to your email.").

Integrations:

- **Azure Cognitive Services:** For natural language understanding and processing.
- **Database Integration:** To fetch user account details and validate information.
- **API for Email Services:** To send password reset instructions.

Flow Structure Diagram:

[User Query] → [Input Node: Preprocess Query] → [Model Node: Generate Response] → [Output Node: Display Response]

Prototype Summary

Implementation Steps:

1. **Environment Setup:** Installed required libraries and verified GPU availability using Azure Machine Learning.
2. **Data Preparation:** Cleaned and labeled a dataset of customer queries for training and testing.
3. **Prompt Flow Design:** Used Azure's visual editor to define input, model, and output nodes.
4. **Testing and Optimization:** Tested the chatbot with sample queries and refined prompts to improve response accuracy.

Challenges Faced:

1. **Inconsistent Responses:** The LLM occasionally generated irrelevant or incomplete answers.
    - **Solution:** Refined prompts and added context to guide the model.
2. **Latency Issues:** High response times during peak usage.
    - **Solution: Optimized the model and used Azure's auto-scaling features to handle traffic.

Monitoring Insights

Metrics Tracked:

1. **Latency:** Average response time of 2.5 seconds.
2. **Error Rates:** Less than 5% of queries resulted in errors.
3. **Usage Patterns:** Peak usage during business hours, with most queries related to account management.

User Feedback:

- Positive feedback on response accuracy and speed.
- Suggestions for adding multilingual support and handling complex queries better.

Improvements Made:

- Added fallback responses for unrecognized queries.
- Integrated a feedback mechanism to collect user ratings and comments.

Future Improvements

1. **Multilingual Support:** Expand the chatbot's capabilities to handle queries in multiple languages.
2. **Advanced Query Handling:** Train the model to handle more complex and domain-specific queries.
3. **Enhanced Monitoring:** Implement real-time analytics and alerts to proactively address performance issues.
4. **User Personalization:** Use user data to provide personalized responses and recommendations.

Conclusion

This project successfully demonstrated the design, implementation, and maintenance of a customer support chatbot using Azure's prompt flow tools. By leveraging Azure's capabilities, the chatbot achieved high accuracy, low latency, and positive user feedback. Future improvements will focus on expanding functionality and enhancing user experience, ensuring the chatbot remains a valuable tool for customer support.