Rule-Based Chatbot Project Report

# NAME:K.MANIKANTA

REG NO : 22BCE9181

# 1. Abstract

This project aims to develop a simple rule-based chatbot capable of responding to a limited set of predefined questions and commands. Unlike AI-powered chatbots, a rule-based chatbot relies on pattern matching and decision trees to provide fixed responses to specific user inputs.

# 2. Problem Statement

In many customer service or FAQ settings, users often ask common questions that can be easily anticipated. A rule-based chatbot serves as a lightweight, efficient solution to automate answering such queries without the need for complex machine learning models.

# 3. Tools and Technologies Used

- Python 3.x  
- Regular Expressions (re library)  
- Random (for response selection)  
- Jupyter Notebook or any Python IDE

# 4. Algorithm

1. Define a dictionary of intents, each with patterns and possible responses.  
  
2. Convert user input to lowercase.  
  
3. Use regex to check if user input matches any predefined patterns.  
  
4. If a match is found, return a random response from the corresponding intent.  
  
5. If no match is found, return a fallback response.  
  
6. Loop until the user types a termination command like 'bye' or 'exit'.

# 5. Source Code (Main Parts)

import random  
import re  
  
intents = {  
 "greeting": {  
 "patterns": ["hi", "hello", "hey"],  
 "responses": ["Hello!", "Hi there!"]  
 },  
 "farewell": {  
 "patterns": ["bye", "goodbye"],  
 "responses": ["Goodbye!", "See you soon!"]  
 }  
}  
  
def get\_response(user\_input):  
 user\_input = user\_input.lower()  
 for intent, data in intents.items():  
 for pattern in data["patterns"]:  
 if re.search(pattern, user\_input):  
 return random.choice(data["responses"])  
 return "I don't understand that."  
  
while True:  
 user\_input = input("You: ")  
 if user\_input.lower() in ['bye', 'exit']:  
 print("Bot: Goodbye!")  
 break  
 print("Bot:", get\_response(user\_input))

# 6. Output Example

You: hi  
Bot: Hello!  
You: bye  
Bot: Goodbye!

# 7. Conclusion

This project demonstrates how a simple chatbot can be developed using rule-based logic. Although limited in flexibility, this approach is ideal for handling FAQs or repetitive queries in a controlled environment.