CCP Report: Simple Python Chatbot using Regex and Learning Capabilities

Name: Shahmeer Khan

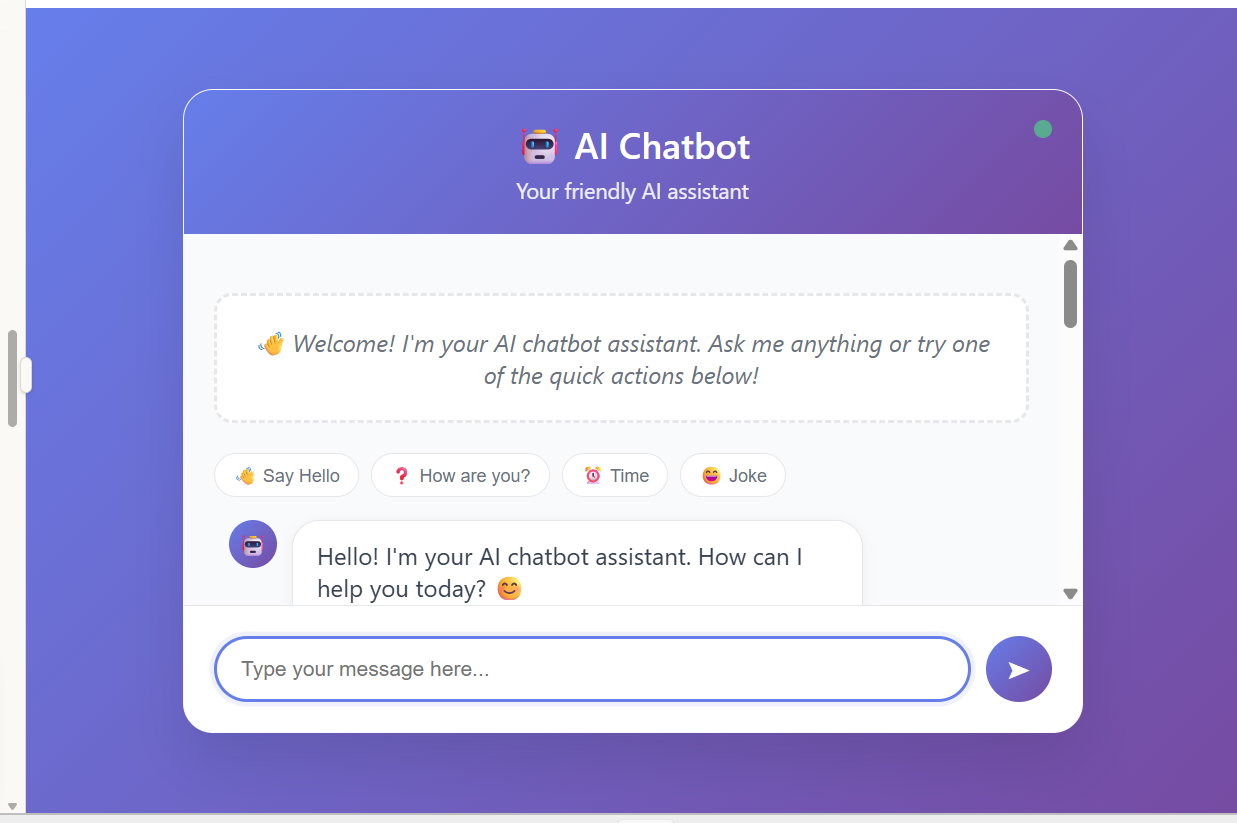
Reg. No: 60099

Course: Parallel and Distributed Computing

Instructor: Sir Ali Akbar

# Table of Contents

1. Abstract
2. 1. Introduction
3. 2. Objectives
4. 3. Tools and Technologies Used
5. 4. System Architecture
6. 5. Methodology
7. 6. Features
8. 7. Code Overview
9. 8. Output Screenshot (Example)



Below is a screenshot of the frontend chatbot interface built with HTML/CSS/JS:

1. 9. Limitations and Future Work
2. 10. Conclusion
3. 11. References

# Abstract

This report presents the implementation of a simple Python-based chatbot using pattern matching via regular expressions and random response selection. Additionally, an enhanced version is developed with learning capabilities, allowing the bot to adapt based on user-taught phrases. The bot maintains conversation history, responds contextually, and supports basic natural language interaction.

# 1. Introduction

The goal of this project is to build a lightweight chatbot in Python without relying on large language models or APIs. It demonstrates core chatbot functionalities such as pattern recognition, response generation, learning new inputs, and session history handling.

# 2. Objectives

• Design a basic chatbot using regex and random responses.  
• Add an enhanced learning capability to store and recall custom responses.  
• Maintain conversation logs and enable saving to JSON.  
• Demonstrate chatbot functionalities through a menu-driven interface.

# 3. Tools and Technologies Used

• Programming Language: Python  
• Libraries: re, random, datetime, json  
• IDE: Visual Studio Code / Jupyter Notebook / Google Colab  
• Platform: Console (Command Line Interface)

# 4. System Architecture

The system consists of the following components:  
• Input Handler: Accepts and preprocesses user queries.  
• Pattern Matcher: Matches user input against known regex patterns.  
• Response Generator: Returns a random or learned response.  
• History Tracker: Stores conversation logs with timestamps.  
• Learner: Stores new response mappings based on user instructions.

# 5. Methodology

The chatbot follows a modular approach:  
• Regular expressions identify intents (greetings, time, name, etc.).  
• A dictionary maps intents to predefined responses.  
• A subclass extends functionality for dynamic learning from users.  
• Optional: Conversation saved as JSON file for review.

# 6. Features

• Basic chatbot with intent recognition  
• Enhanced learning chatbot with custom response storage  
• Session history tracking and optional saving  
• Menu-based demo selector for quick interaction  
• Graceful handling of exit and errors

# 7. Code Overview

The code contains two classes: `SimpleChatbot` and `LearningChatbot`. `SimpleChatbot` handles basic logic including preprocessing, regex matching, and response generation. `LearningChatbot` inherits from it and adds user-teachable triggers. Demo functions allow console testing.

# 8. Output Screenshot (Example)

Screenshots of example chatbot conversation will be added here later.

Note: Please capture screenshots of a session from both chatbot versions and paste here.

# 9. Limitations and Future Work

Current Limitations:  
• Works only via command line  
• Cannot fetch real-time data (e.g., weather, news)  
• No GUI  
  
Planned Enhancements:  
• Integration with GUI using Tkinter or Streamlit  
• Add persistent memory using file/database  
• Add voice input/output support

# 10. Conclusion

This project shows that with basic Python libraries, a simple but effective chatbot can be implemented. With added learning capabilities, it simulates personalized interactions. This foundational chatbot can be a stepping stone for further developments such as GUI-based or web-deployed conversational systems.

# 11. References

[1] Python Documentation – https://docs.python.org/3/  
[2] Regular Expressions HOWTO – https://docs.python.org/3/howto/regex.html  
[3] JSON module – https://docs.python.org/3/library/json.html  
[4] Streamlit – https://streamlit.io (for potential GUI implementation)