Introduction:

A chatbot is a software application used to conduct an online chat conversation via text or text-to-speech. [1] Chatbots are designed to interact with humans through texts as a human would. Creating an accurate chatbot system requires a lot of tuning and testing. These days, most organisations use chatbots as their customer service provider because it's faster and more efficient.

Chatbots can be of many types. 3 of the most common types are-

1. Rule-based chatbots
2. AI-powered chatbots
3. Hybrid chatbots

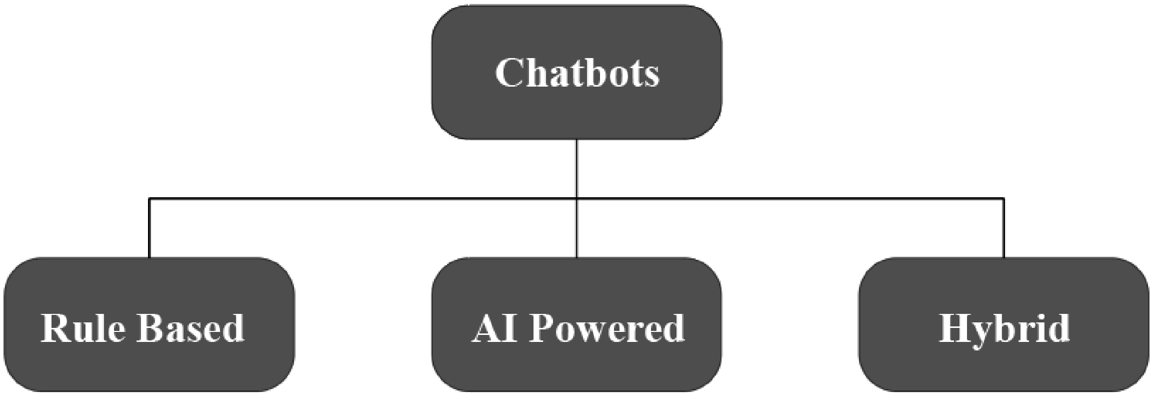


Fig: Types of Chatbots

We made a rule-based chatbot named 'Cara' for departmental use for our Software Engineering Sessional course's final project.

Features: The features of our chatbot include the following-

- Providing class and exam schedules
- Providing important recent notices
- Teachers' information and contact details

- Link to necessary resources such as class notes, books, etc.
- Information related to Robotronics and other Competitions
- Necessary information regarding the Department of MTE

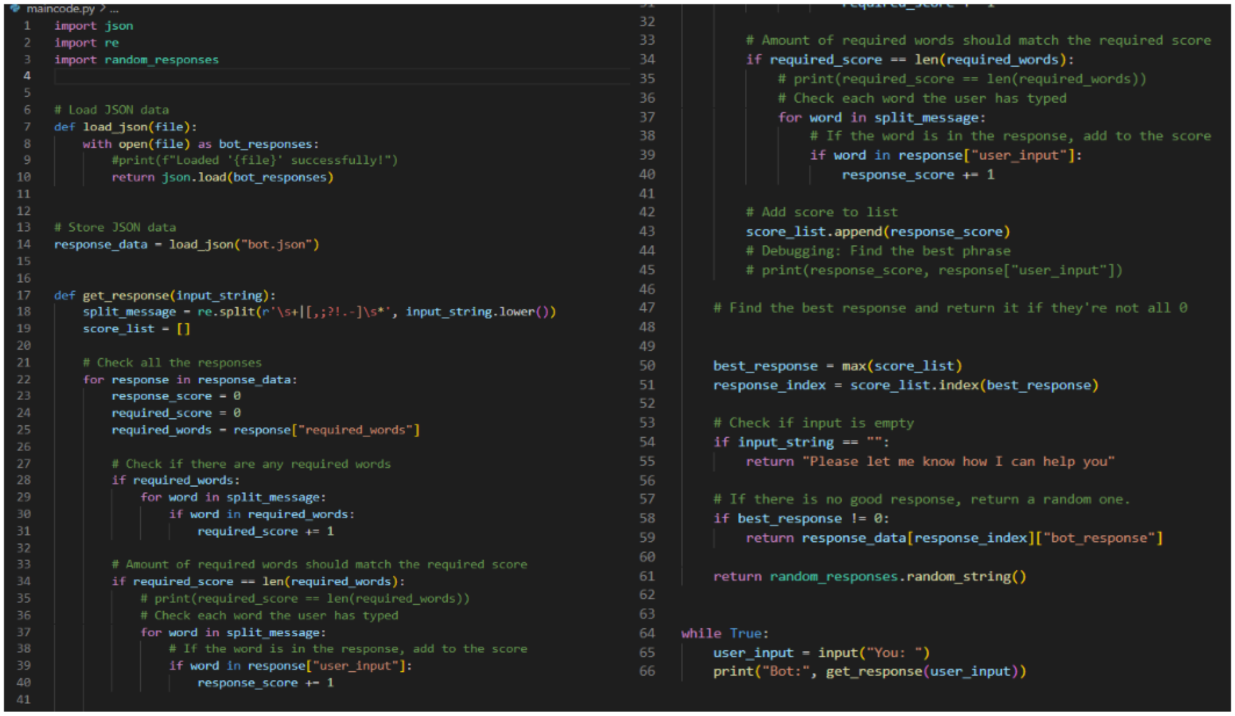
Platforms Used: Visual Studio Code (VS Code), Django, Ngrok.

Coding Language: Python 3.10

Procedure:

Part-1:

The entire process can be divided into two parts. In one part, the chatbot was coded and tested on VS Code. The entire code for the chatbot was written in Python. For clarity and efficient use of data, a separate JSON file was created which was imported to the main code. Moreover, to code an accurate chatbot a very simple algorithm was used that matched the words from the input to those given in the JSON file.



```python
import json
import re
import random_responses


# Load JSON data
def load_json(file):
    with open(file) as bot_responses:
        #print(f"Loaded '{file}' successfully!")
        return json.load(bot_responses)


# Store JSON data
response_data = load_json("bot.json")


def get_response(input_string):
    split_message = re.split(r'\s+|[,;?!.-]\s*', input_string.lower())
    score_list = []

    # Check all the responses
    for response in response_data:
        response_score = 0
        required_score = 0
        required_words = response["required_words"]

        # Check if there are any required words
        if required_words:
            for word in split_message:
                if word in required_words:
                    required_score += 1

        # Amount of required words should match the required score
        if required_score == len(required_words):
            # print(required_score == len(required_words))
            # Check each word the user has typed
            for word in split_message:
                # If the word is in the response, add to the score
                if word in response["user_input"]:
                    response_score += 1

```

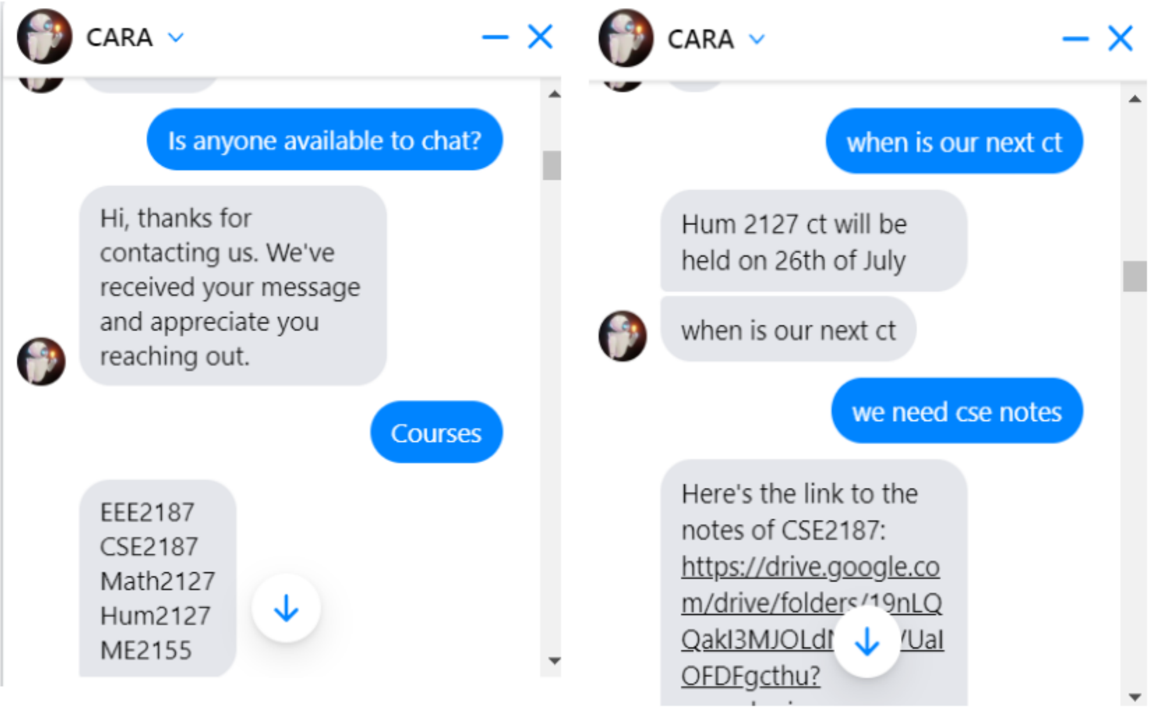
```python

        # Amount of required words should match the required score
        if required_score == len(required_words):
            # print(required_score == len(required_words))
            # Check each word the user has typed
            for word in split_message:
                # If the word is in the response, add to the score
                if word in response["user_input"]:
                    response_score += 1

```

```python
        # Add score to list
        score_list.append(response_score)
        # Debugging: Find the best phrase
        # print(response_score, response["user_input"])

    # Find the best response and return it if they're not all 0


    best_response = max(score_list)
    response_index = score_list.index(best_response)

    # Check if input is empty
    if input_string == "":
        return "Please let me know how I can help you"

    # If there is no good response, return a random one.
    if best_response != 0:
        return response_data[response_index]["bot_response"]

    return random_responses.random_string()


while True:
    user_input = input("You: ")
    print("Bot:", get_response(user_input))
```

Fig: Main chatbot code

Part-2:

This was the rather complicated part of the entire process. In this part, at first, Django was installed as was used to work as our local host. Then we created an app named CARA which is the name of our chatbot. After that, it was connected to the main folder. There is a section named views.py which contains the code where it connects to the chatbot to Facebook. After adding all the code the local host was connected to a public server using Ngrok. Then an app was created using Facebook Developer's account. A feedback URL was given by Facebook which was later pasted in the main code. The main chatbot code and the code that connects it to Facebook were later merged together.



Is anyone available to chat?

Hi, thanks for contacting us. We've received your message and appreciate you reaching out.

Courses

EEE2187
CSE2187
Math2127
Hum2127
ME2155

when is our next ct

Hum 2127 ct will be held on 26th of July

when is our next ct

we need cse notes

Here's the link to the notes of CSE2187: https://drive.google.com/drive/folders/19nLQQakI3MJOLd /UalOFDFgcthu?

Fig: The final output on Facebook

Errors:

We faced a few errors while creating the entire program. For instance, while coding the chatbot the JSON wasn't working. There were issues with the path file that was showing errors. But we took help from Stackoverflow and solved our issue. There were also multiple testing errors but we finally got accurate outputs.

However, we faced the most errors in the second part. The local host wasn't connecting to the public host which was a very big issue to deal with. We suffered for weeks solving this problem. Then again there were issues while merging the two codes.

Discussion:

Even though we faced multiple errors, the biggest one was the failure to connect it with the public server. But with the help of our course instructor, the problem was solved in a jiffy and finally, was connected to Facebook. Problems arose while choosing the right platforms and mediums. For instance, at first, the bot was coded on Google Colab for team collaboration and later moved to VS code. In the final presentation, some of the features of the bot were missing but most of them worked accurately.

Conclusion:

The main result of the project lies in our learnings. That is, we learned a lot and developed new skills. We also learned to work as a team more efficiently. There were multiple issues and even at the last moment, all of them were solved successfully. We believe this chatbot if monitored wisely, can be a big help for the department and the students.

Reference:

1. “Chatbot”. https://en.wikipedia.org/wiki/Chatbot