FIRST CITY UNIVERSITY COLLEGE
WAY AHEAD

Faculty of Engineering and Computing

Bachelor of Computer Science (Intelligent Systems) (Hons)
Bachelor of Software Engineering (Hons)

CAPC3008 Natural Language Processing

Assignment Specification

STUDENT NAME	STUDENT ID	CLASS CODE
SEE JIA JUN	B1708	
CHONG CHOON NAM	B1672	
JAMESON CHIN JIAN HAN	B1716	
YEE JUN JIE	B1669	

This coursework requires you to write a report and program.

SUBMISSION DATE: Friday, 24 April 2024, 5 pm.

PART 1 – The Implementation (30 marks)

Design and Development of a Chatbot

You are required to enhance the design and develop of a basic pattern-and-response chatbot so that the program would carry on a more convincing conversation. You should apply NLP techniques to achieve specific goals which you can choose. You must choose **at least two** goals.

Examples of goals:

- **Anaphoric resolution**
- **Prepositional Phrase-attachment resolution**
- Query the user for their name and remember their name
- Resolve word ambiguities
- Update knowledge of chatbot
- Keep history of chat

PART 2 – The Report (70 marks)

Section 1 - Introduction should (i) explain what chatbots/conversational agents are, including a brief history including the original Eliza, and (ii) provide several (at least three, *not* including Eliza) examples of some real chatbots or conversational agents, with brief analyses of their capabilities. The analysis of their capabilities should **focus on their NLP abilities**.

(3 – 6 pages)
10 marks

Section 2 – Description of a Basic Chatbot's Algorithm
Describe and explain the way a basic pattern-and-response chatbot works. You may use the chatbot program made available to you in the description. This means explaining the overall processing involved – i.e. what happens when you start up the program, when it is running and accepting inputs from the user, and when it ends. Remember to describe the purpose and structure of any files that the program reads from or writes to. A good answer will explain the pattern matching mechanism in detail, including any variable-matching.

(3 - 6 pages)
15 marks

Section 3 – Proposed NLP Design Enhancements – The Goals

Describe your chosen two (or more) goals and their design in detail. It must describe the NLP techniques you propose to use. Explain the algorithm (series of steps in the processing) required, together with descriptions of any data required (in files) and data structures required (in the program).

(Unlimited pages)
30 marks

Section 4 – Summary

Summarise your results and achievements. A good answer will include the comparison of your modified chatbot with an existing chatbot.

(1 to 3 pages)
10 marks

Overall Report Format and Presentation: correct IEEE referencing, formatting, presentation, grammar, spelling, lengths of sections etc.

5 marks

Submission requirements

Two formats are required for submission – printed hard copy, and soft copy on CD.

Follow standard reporting requirements unless stated otherwise and should be appropriately word processed. Text must be 12 point Arial or Verdana, flush left, single spaced. Include your name and student registration number in a header.

You **must** include citations within your text and a References section at the end of your report. You may also include a Bibliography, and Appendices for other useful information.

You do not need to include title page and table of contents.

You are also required to submit the report to Turnitin for plagiarism check.

About the ELIZA Source Code Provided

You have been provided with an ELIZA-type program. This is a small ELIZA shell created by Paul McDonald, written in C++. There are three versions, of varying capability. They can be compiled and run using the standard Microsoft Visual development environment. You are free to edit your copies of them to experiment with your designs.

There are also many Eliza-like programs available over the Internet, often written in JAVA, and you could look at these to help develop your ideas. Some examples and useful sources are:

http://en.wikipedia.org/wiki/ELIZA

http://chayden.net/eliza/Eliza.html

http://jerz.setonhill.edu/if/canon/eliza.htm

http://www.ai-programming.com/

http://www.cs.cmu.edu/afs/cs/project/ai-repository/ai/areas/classics/eliza/0.html

References:

Joseph Weizenbaum, 1966, ELIZA – a computer program for the study of natural language communication between man and machine, Comms ACM, 9, 36-45 (PDF supplied)

FACULTY OF ENGINEERING AND COMPUTING

Bachelor of Computer Science (Intelligent Systems) (Hons)
Bachelor of Software Engineering (Hons)

CAPC3008 Natural Language Processing

<u>Assignment Mark Sheet</u>

Name of Student/ID: ____

No	Items	Poor	Satisfactory	Excellent	Marks
1	The Program [30 %]	0 5 10 (Minor changes – e.g. update knowledge or better structure)	15 20 (Two key goals)	25 30 (At least 3 non-trivial goals implemented)	30
2	Introduction [10 %]	0 2 4 (Insufficient)	5 6 7 8 (Reasonably written)	9 10 (Very clear, detail and concise)	10
3	Description of Chatbot [15 %]	0 2 4 (Insufficient)	6 8 9 10 12 (Reasonably written)	14 15 (Very clear, detail and concise)	15
4	The Goals [30 %]	0 5 10 (Insufficient)	15 20 (Reasonably written)	25 30 (Very clear, detail and concise)	30
5	Summary and Insights [10 %]	0 2 4 (Insufficient)	5 6 7 8 (Reasonably written)	9 10 (Very clear, detail and concise)	10
6	Report Format & Presentation [5 %]	0 1 2 (Improper, rush work)	3 4 (Minor problems with formatting / referencing)	5 (Everything presented neatly)	5
Comments: ***Lecturer's Signature:***				**Total Report Mark (100%)**	

Table of Contents

List of Figures

1 Introduction

1.1 Definition of Chatbots/ Conversational Agents

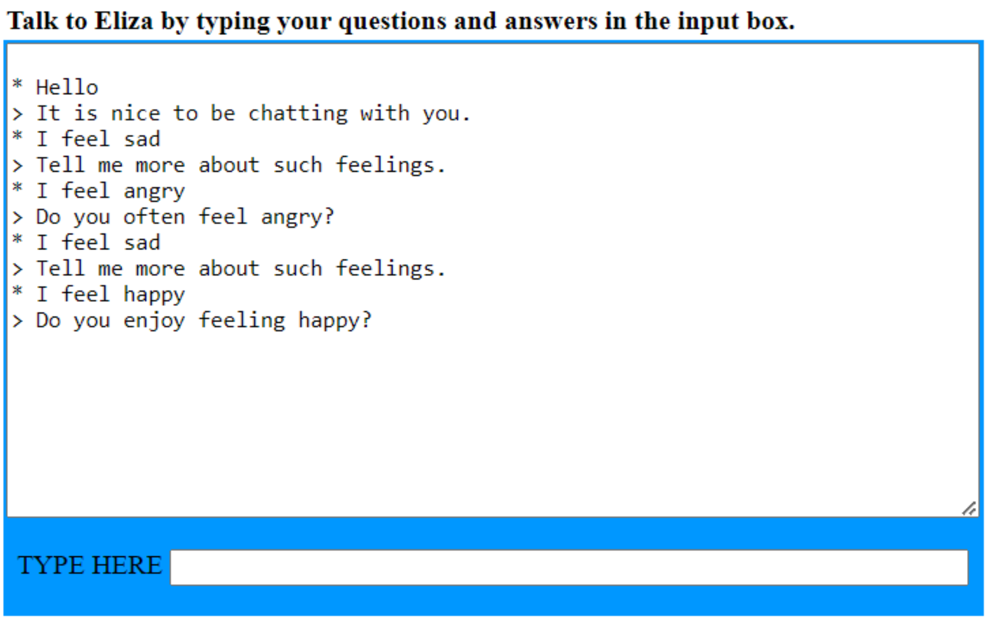
A chatbot is like having a digital buddy on your device, designed to talk with you through messages or even voice commands.[1] Some are very simple like giving quick responses to simple question, while others are like a personal digital assistant, learning information and getting better at helping you over time. You have probably chatted with one without even realizing it, especially if you've reached out to customer service online. They're great at handling common questions and giving people a hand when they need it. And in online shopping, they're there to track your orders, offer suggestions, and clear up any confusion. More businesses are seeing the benefits of these chatbots, using them to streamline tasks and make interactions smoother, all to give customers a better experience.

When you see a system that chats with you using human-like language, that's what we call a conversational agent. These agents, chatbots or virtual assistants, are like the real-world application of computer language skills. They're handy tools for having natural conversations, just like you would with another person. These agents can talk to you through text, voice, or chat on all sorts of devices like phones and computers, making it easy for you to do things like ordering food or handling tasks.[2] They are armed with a bunch of cool tech-like speech recognition, text-to-speech magic, and some serious machine learning skills to understand and respond to what you're saying.

1.2 Difference between Chatbots and Conversation Agents

Aspect	**Chatbot**	**Conversational Agent**
Definition	Software programmed to simulate human conversation.	Program using NLP and NLU to converse with humans.
Capabilities	Can provide information, answer questions, perform tasks, and make purchases.	Can understand human emotions, answer basic questions, respond to commands, and engage in natural language conversations.
Interaction Complexity	May use rule-based responses or decision trees.	Utilizes advanced NLP and NLU technologies for more complex and natural interactions.
Focus	Often focus on specific tasks or objectives.	More adaptable and capable of handling a broader range of queries and commands.
Personalization	Limited ability to provide personalized interactions.	Capable of adapting responses based on context and providing more personalized interactions.

1.3 History of Eliza


Talk to Eliza by typing your questions and answers in the input box.

```
* Hello
> It is nice to be chatting with you.
* I feel sad
> Tell me more about such feelings.
* I feel angry
> Do you often feel angry?
* I feel sad
> Tell me more about such feelings.
* I feel happy
> Do you enjoy feeling happy?
```

Figure 1: Eliza Chatbot

In the 1960s, MIT professor Joseph Weizenbaum, a computer scientist at the Massachusetts Institute of Technology (MIT) created the first ever chatbot, Eliza. Eliza is not just an old computer program; it was a groundbreaking experiment in human-computer chitchat. Eliza could understand what you were saying and come up with clever responses, even though she didn't really understand it like a human does. She did this by spotting keywords in a message and giving replies that sounded like she was really listening, using some pretty basic matching and swapping techniques.[3]

Eliza's software, originally written in a language called MAD-SLIP, had different scripts that gave her the capability to carry on a conversation. One of her most famous roles was playing doctor.

When Eliza was initially released people were chatting away with her, thinking they'd struck up a conversation with a fellow human. Weizenbaum was blown away by how folks connected with Eliza, even though she was just a bunch of code with no real understanding.

But Eliza's impact went way beyond just fooling around with people. She kickstarted a whole new era of interest in stuff like chatbots, language processing, and how we interact with computers. Showing the world just what artificial intelligence could do in mimicking human communication. Even now, her legacy lives on in the chatbots and virtual assistants we chat with every day.

1.4 Example of Chatbots

1.4.1 Cortana

Figure 2: Cortana Logo

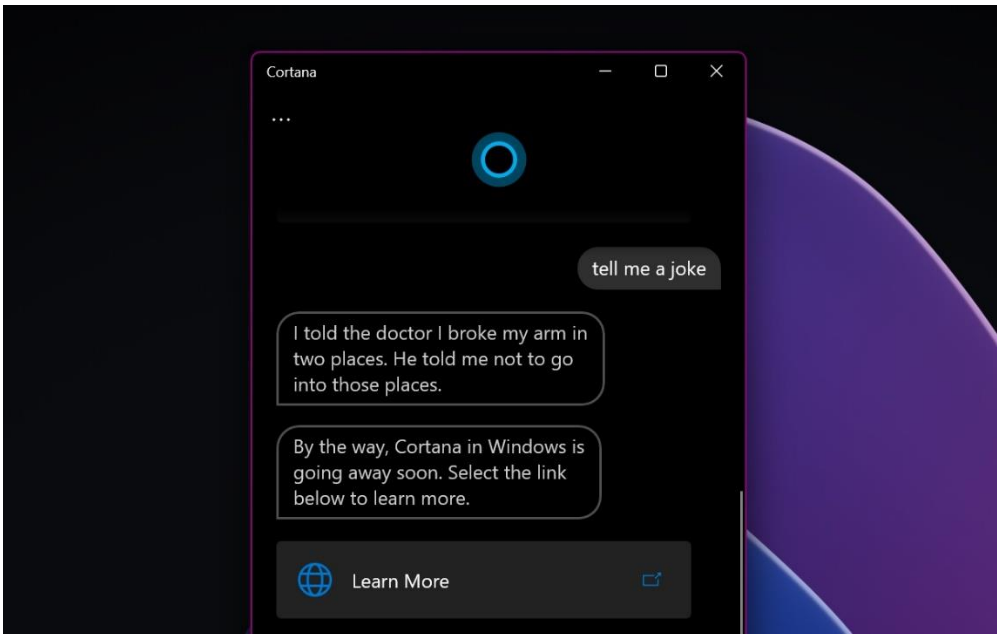
1.4.1.1 Introduction of Cortana

Back in 2014, Microsoft launched Cortana, a virtual assistant that is all about lending a helping hand, whether you're talking to it or typing away. From answering questions and firing off emails to keeping your calendar in check and even giving you the information on the weather.[5]

For most users, Cortana is like a walking encyclopaedia and dictionary. Whenever you need quick info without diving into the depths of the internet, Cortana's got your back. Whether you're asking a quick question or firing off a query, it's all about getting speedy answers without breaking a sweat.

Cortana isn't just retrieving info from a pre-loaded database, it's all about AI and machine learning. Over time, Cortana gets to know you better than your friends. It picks up on your habits, your likes, and dislikes. So instead of just referring to a stored set of questions and answers, Cortana's all about having real conversations with you, like a true friend.

1.4.1.2 Capabilities of Cortana



tell me a joke

I told the doctor I broke my arm in two places. He told me not to go into those places.

By the way, Cortana in Windows is going away soon. Select the link below to learn more.

Figure 3: Cortana GUI

Cortana can understand and analyze what you throw its way. When you ask Cortana to "book a flight to Malaysia", it's breaking down your request into specific details. That means picking out the important stuff, like ("destination: Malaysia"), so it can spring into action and get things done.

It's also pretty good at reading between the lines. That means when you ask a question, Cortana dives deep into the context, whether it's your personal preferences or what's been discussed before. It remembers what you've talked about before, so over time, it gets better at giving you exactly what you need.

Furthermore, Cortana is always leveling up its language skills with some serious machine learning algorithms by crunching through tons of user data. With every interaction and of feedback, Cortana gets a little bit smarter.[6] It's like having a personal tutor, but for understanding what you say and tailoring its responses just for you. So, the more you chat with Cortana, the better it gets at predicting what you need and serving up the perfect solution.

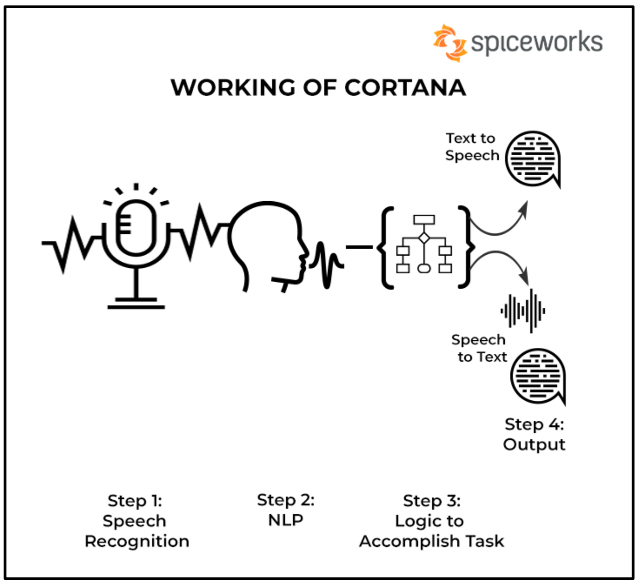


Figure 4: Architecturen Design of Cortana

1.4.2 Siri

Figure 5: Siri Logo

1.4.2.1 Introduction of Siri

Siri is Apple's virtual assistant for all your iOS, macOS, tvOS, and watchOS devices. Powered by the magic of artificial intelligence, will listen when you speak and dish out helpful information. Whether it's pulling info from Safari or your calendar, reading and composing emails and texts, or even tackling other tasks you throw their way.[7]

It all started at SRI International, who were working on this project called CALO. Fast forward to 2007, these same folks decided to turn their creation into a separate business, Siri, Inc. Leading the charge were Tom Gruber as the CTO, with Dag Kittlaus and Adam Cheyer as CEO and Vice President of Engineering.

Then, in 2010, Apple came and bought Siri, Inc. in a big acquisition move. Suddenly, Siri was part of the Apple family, finding its way onto iOS devices as a native feature. In June 2012, Apple announced that Siri was expanding its horizons, making its way onto the third-gen iPad with iOS 6 and even teaming up with third-party apps.

To trigger Siri on Apple’s devices, it is as simple as tapping, pressing, or using the wake command. Saying "Hey Siri" to wake Siri from almost any Apple device lets user ask questions or issue instructions. In the latest version of iOS, users may now speak "Siri" before a command with iOS 17 and the HomePod Software Version 17. "Hey" has been eliminated.

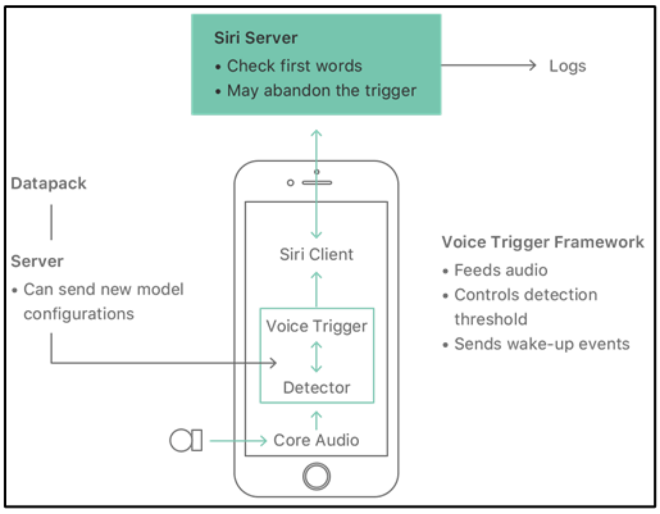
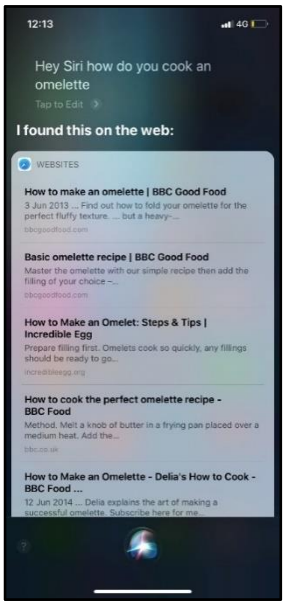


Figure 6: Siri's Architecture Design

1.4.2.2 Capabilities of Siri

When you ask a question, Siri goes to work, using the natural language processing (NLP) playbook.



12:13 4G

Hey Siri how do you cook an omelette

Tap to Edit

I found this on the web:

WEBSITES

How to make an omelette | BBC Good Food
3 Jun 2013 ... Find out how to fold your omelette for the perfect fluffy texture. ... but a heavy-...
bbcgoodfood.com

Basic omelette recipe | BBC Good Food
Master the omelette with our simple recipe then add the filling of your choice –...
bbcgoodfood.com

How to Make an Omelet: Steps & Tips | Incredible Egg
Prepare filling first. Omelets cook so quickly, any fillings should be ready to go...
incredibleegg.org

How to cook the perfect omelette recipe - BBC Food
Method. Melt a knob of butter in a frying pan placed over a medium heat. Add the...
bbc.co.uk

How to Make an Omelette - Delia's How to Cook - BBC Food ...
12 Jun 2014 ... Delia explains the art of making a successful omelette. Subscribe here for me...

Figure 7: Siri in Action

First, Siri listens with **automated speech recognition (ASR)**, turning spoken words into text it can understand. From there, it's all about making sense of what is being said. Siri breaks down the text, analyzing its structure with techniques like **dependency parsing**,

syntactic analysis, **part-of-speech (POS) tagging**, and **noun phrase chunking**. It's like putting together a puzzle, figuring out how all the pieces fit together to create a coherent sentence. It’s not only understanding the words being said but also picking up on the important stuff, like what is being asked for and any clues dropped along the way.

Siri is constantly improving its language skills using natural language processing (NLP) techniques. First, there's **statistical modeling**, which helps Siri learn language patterns and sort out any tricky parts. Then there's **machine learning**, where Siri's constantly learning from your interactions to improve its understanding and responses. Siri is also diving into the world of **semantic searching**, which helps it dig up relevant info from all sorts of sources by understanding the meaning behind your queries.[8]

Auto-categorization and parsing are also used in Siri. These tools help Siri break down sentences and get a handle on the grammar and structure of what you're saying.

Siri also has advanced capabilities like **sentiment analysis**. This tool evaluates the tone and emotion behind your questions, so Siri can tailor its responses to match your mood. Whether you're feeling happy, sad, or somewhere in between, Siri's got your back with personalized replies. Siri also uses machine translation, where Siri can seamlessly translate text between languages, breaking down barriers and making communication easier no matter where you are in the world.

Thanks to integrated services like Ask.com and Wolfram Alpha, Siri can dig up all sorts of insightful info to answer your inquiries. Siri's got you covered with smart and helpful replies on every questions.

1.4.3 DialogFlow

Figure 8: DialogFlow Logo

1.4.3.1 Introduction of DialogFlow

In September 2016, Google purchased Speaktoit and rebranded it as API.AI. However, in December 2016, the assistant app has been discontinued. After that, the platform’s name was changed to DialogFlow in October 2017 to better reflect its emphasis on enabling dialogue-based interactions. Dialogflow then merged with Google Cloud Platform in November 2017, strengthening its place in Google's developer tools and services portfolio.

DialogFlow agent is a virtual agent that manages several discussions with the end-user. It consists of natural language understanding module which have the ability of comprehending natural language and the text of complexity found in human speech. Dialogflow converts voice or text input from the user into structured data that can be understood by the applications and services. User can design and build their own DialogFlow agent to manage the different types of conversation based on the user’s system.

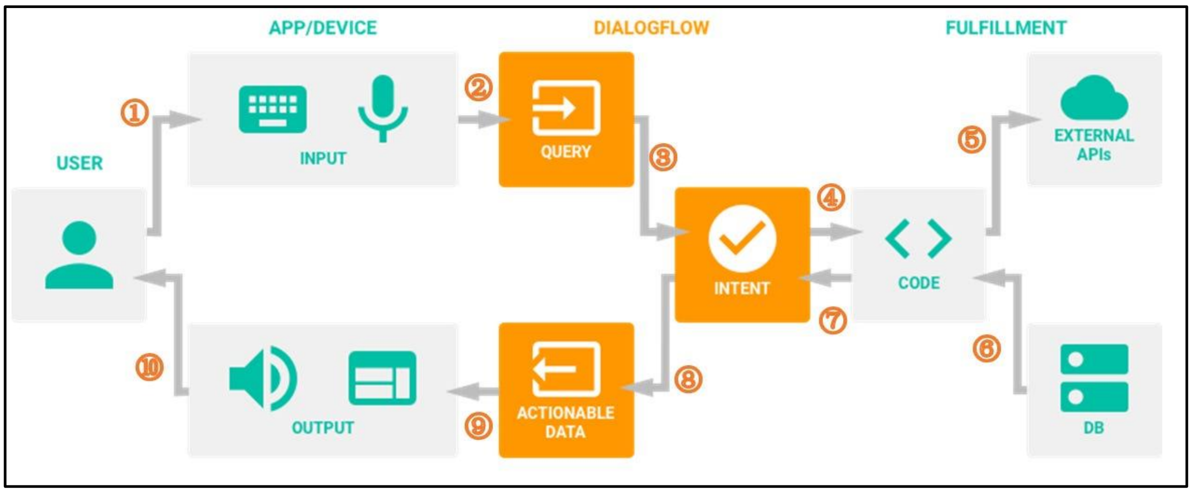


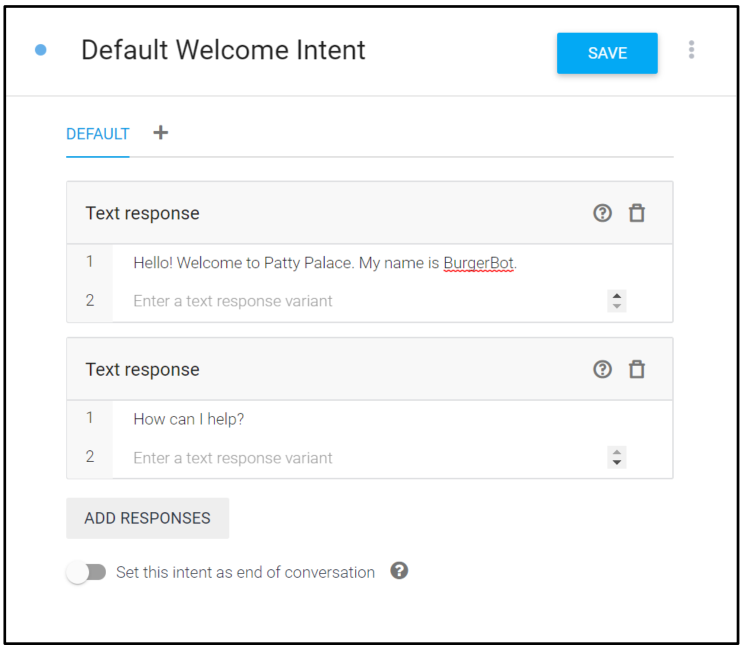
Figure 9: Architecture of DialogFlow

1.4.3.2 Brief Analyses of Capabilities

There are few Natural Language Processing capabilities applied in DialogFlow. The first capability is **intent recognition**. This feature allows DialogFlow to interpret a user's communication and determine the purpose and intention of the user. For example, DialogFlow can identify the intention of user to get the weather information and adjust the response about the weather to the user. Unlike rule-based chatbot, DialogFlow applies the process of identifying and extracting specific information relevant to the intent from user's message. It involves extract information from user input such names, dates, places or others to improve the chatbot capability to deliver the more accurate and relevant answer for the user input.

Besides, the DialogFlow also has the capabilities of **advanced context management strategies** which allow it to keep the context across several conservational rounds. The feature allows DialogFlow to recall past exchanges and leverage that context to interpret messages more effectively which can lead to more cohesive and organic conversations between DialogFlow and users. Furthermore, Dialogflow also uses **machine learning** methods to continually learn and get better over time. It improves the entire conversational experience by honing its grasp of linguistic subtleties and user preferences through the analysis of user interactions and feedback.[9]

Additionally, Dialogflow provides **smooth connection to external databases and knowledge bases**, enabling developers to add information from several sources to the chatbot's answers. The connection to database helps the chatbot in the developer's application having more functionality by giving it access to a wide range of relevant information from user input includes product specifications, FAQs, and support materials. DialogFlow's strong natural language processing (NLP) feature allows developers to design different perspective and smart conversational experiences in the application that easy to satisfy the demands of their target user in a variety of contexts and sectors.


Default Welcome Intent

SAVE

DEFAULT +

Text response

1 Hello! Welcome to Patty Palace. My name is BurgerBot.

2 Enter a text response variant

Text response

1 How can I help?

2 Enter a text response variant

ADD RESPONSES

Set this intent as end of conversation

Figure 10: DialogFlow Back-End

2 Description of Basic Chatbot’s Algorithm

2.1 Basic pattern-and-response chatbot

Based on our study, a basic pattern-and-response chatbot operates on a straightforward principle. It will receive user input in text format. This input can be any questions, commands or statements. After the chatbot has receive the input from user, the chatbot will automatically compare it against a set of predefined patterns or rules. Using predefined patterns or rules, the chatbot can easily capture the key words from user input and make response. This type of chatbot is known as rule-based chatbot. Unlike more advanced chatbots that use machine learning and natural language processing to learn from user interactions, basic pattern-and-response chatbots are designed to be simpler and rely on a fixed set of rules.[4]

The structure of pattern-and-response chatbot include two main components, namely pattern matcher and response generator. In order to find patterns and keywords that satisfy predetermined rules, the pattern matcher looks through user input. These rules might be as simple as matching a keyword or as complicated as regular expressions. The response generator selects a predefined response associated with the closest matching pattern. These responses are stored in a database or some other data structure. With the response selected, the chatbot generates an appropriate output to display back to the user. This output may include short or long text messages. In cases where the input doesn't align with any predefined patterns, the chatbot will apply a fallback mechanism. This could involve offering a generic response like "I didn't understand, can you please rephrase?" or presenting a menu of options for the user to choose from.

2.2 Overall Processing

When using basic pattern-and-response chatbot, the initialization process will be started. The predefined rules and patterns with the corresponding responses will load into memory as well. This can help the pattern matcher easily to recognize and identify the key words from user. After recognizing the key words from user input, the response generator will response to the user in few seconds. The pattern and rules are carefully created to collect frequently used phrases or keywords that users may use to optimize the chatbot's capacity to have relevant dialogues. The programme starts a continuous loop and get ready to handle messages or questions, as soon as it launches.

The chatbot will constantly take user inputs and use pattern matching to find the key patterns. To achieve this, the pattern matcher will compare the incoming user input and present patterns kept in the memory. It chooses the predetermined response that corresponds to a pattern after a matching pattern has been found. The interaction cycle of match the pattern is finished when the key words is match to predefined rules and response the message to the user. The chatbot may have dynamic conversations and respond to user messages or inquiries in a timely and appropriate manner.

When the basic pattern-and-response chatbot is ended, all ongoing chat or interactions are ended politely to provide users the proper closure. This could involve notifying users via a final message that the chat session is ended such as ‘Bye’, ‘Thank You’ or others. To maintain system efficiency and free up system resources, any resources such as memory or file handles is allocated during the program's runtime are ended. Lastly, any analytics or logging data that is gathered throughout the chatbot's operation, if appropriate, may be analysed or stored for additional study. The chatbot eventually reaches a state of completion, which indicates the conclusion of its active processing and gives the user or system back control. This systematic shutdown procedure guarantees a fault user experience and a smooth and effective end to the chatbot's work.

2.3 Purpose and Structure of Files

The purpose and structure of files in basic pattern and response chatbot depends on the implementation and design concept in development stages. However, there are common purpose and structure of files for basic chatbot. Three common files include intent files, pattern files, and response files.

Pattern file is used to store a list of patterns or templates that the chatbot uses for pattern matching. The patterns are typically organized by intent and may include placeholders to capture variable information. Intent file is used to store predefine intent that represents the goal or purpose of user inputs. Each intent may contain multiple patterns associated with it, along with corresponding responses. Response file stores the list of responses corresponding to patterns or intents. Responses in response file may include placeholder markers that are replaced with values extracted from the user input during response generation.

To explain the flow of how these files been used to process the input, we follow step by step from the input. When a user input, the input will undergo pre-processing process such as tokenization, remove stopwords, remove punctuation, lemmatization and others. After the pre-processing process, the input will compare the pattern with pattern store in pattern files. Each pattern in the file represents a specific conversational structure or intent. The chatbot iterates through the patterns and attempts to find the closest match for the user input. Once a match is found, the corresponding response associated with that pattern is retrieved from the response file.

2.4 Pattern Matching Mechanism

To identify the user intents accurately, understand the pattern matching mechanism is important in basic chatbot. First of all, chatbot will loads pattern from predefined pattern files. After that, when the chatbot receive a user input, it will undergo pre-processing process to simplify the text. This includes tokenization, remove stop words, remove punctuation, lemmatization and others.

After that, chatbot will iterates through the loaded patterns and attempts to find a match for the pre-processed user input. The chatbot will compare and check whether the input contains any of the keywords or phrases defined in the pattern. The chatbot will selects the closest matching pattern based on predefined criteria, such as the number of matched keywords or the presence of variables. If multiple patterns match equally well, the chatbot may prioritize based on predefined rules or choose randomly.

Once a match is found, the chatbot retrieves the corresponding response associated with that pattern from the predefined response file. If no pattern match is found, the chatbot may respond with a default message indicating that it didn't understand the user's input and the chatbot mat prompt user to enter again their input.

3 Propose NLP Design Enhancements

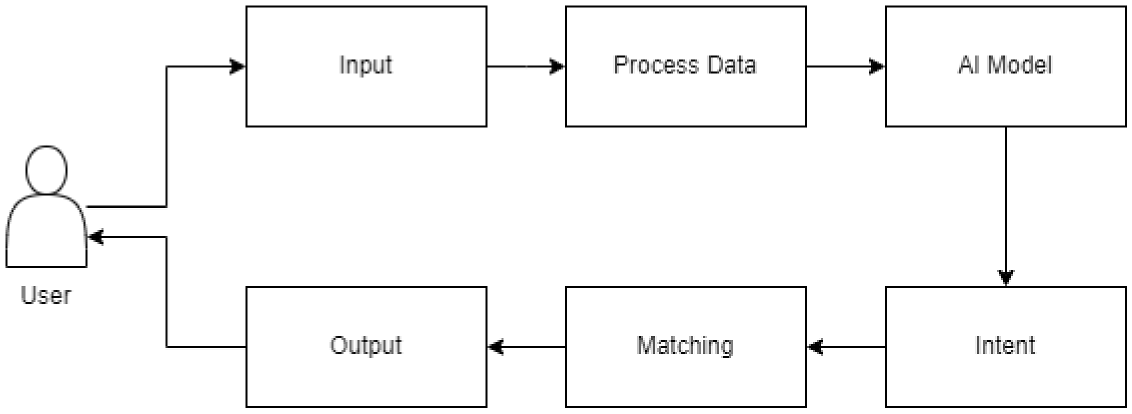


Figure 11: Architecture Diagram of our System

First, the user will input something. Then the sentence will be processed to lemmatization, pos tagging, ner tag, filtered word, and stop word. After processing the input, we will use AI model to look for the intent of the user input. After searching for the intent, the chatbot will match the tag and intent of the user and provide an output based on the tag.

The model that we are using for this chatbot is Neural Network because it has the highest accuracy among the others that we have tested as shown in figure 12.

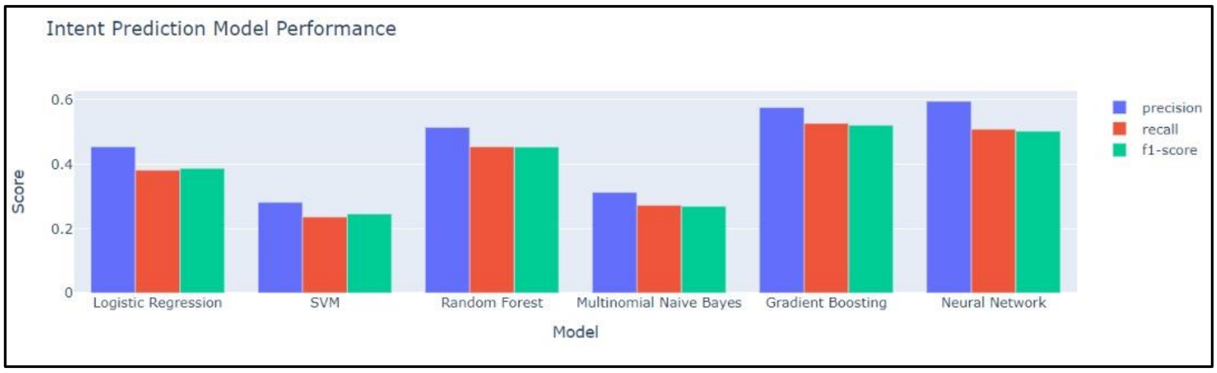


Figure 12: Intent Prediction of other Models

Figure 13 shows the accuracy after using Neural Network for training and testing.

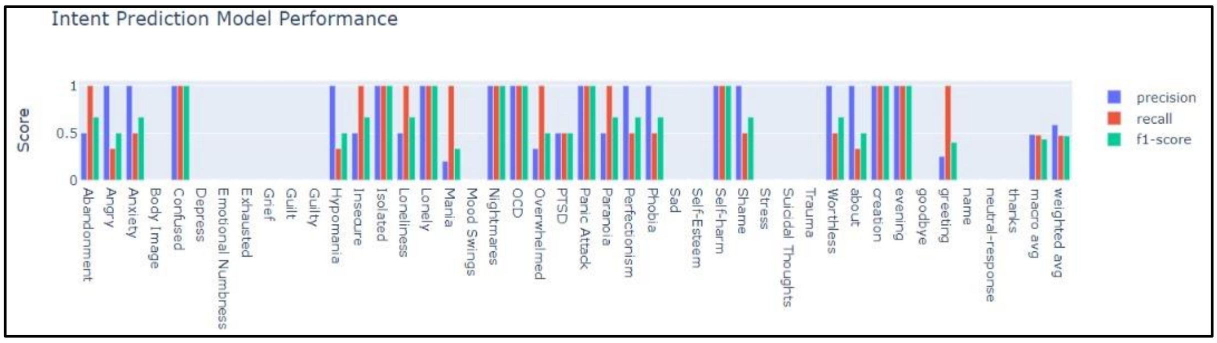


Figure 13: Training and Testing Output

Figure 14 shows the amount of common word used in a pattern.

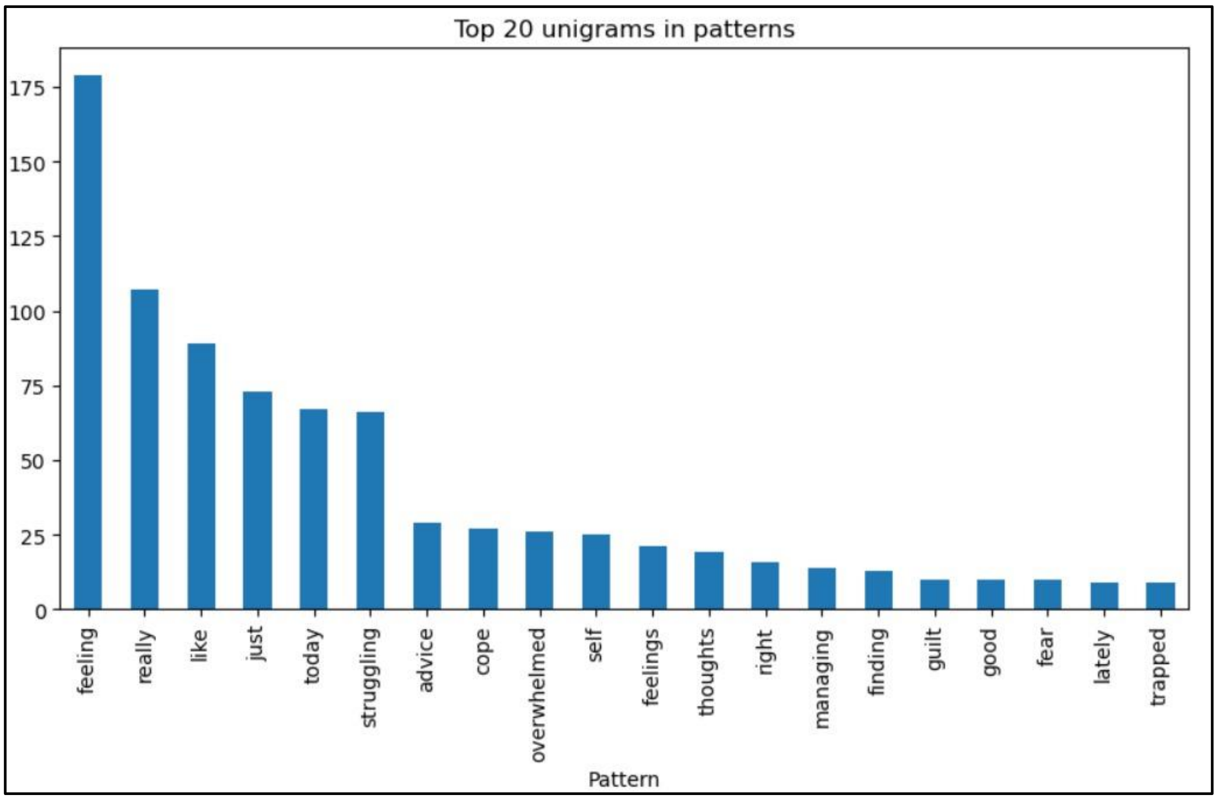
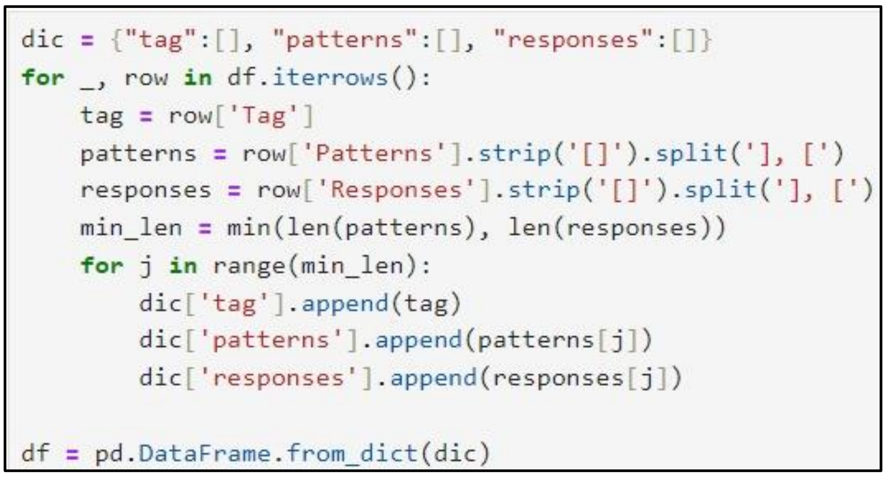


Figure 14: Unigram for Pattern

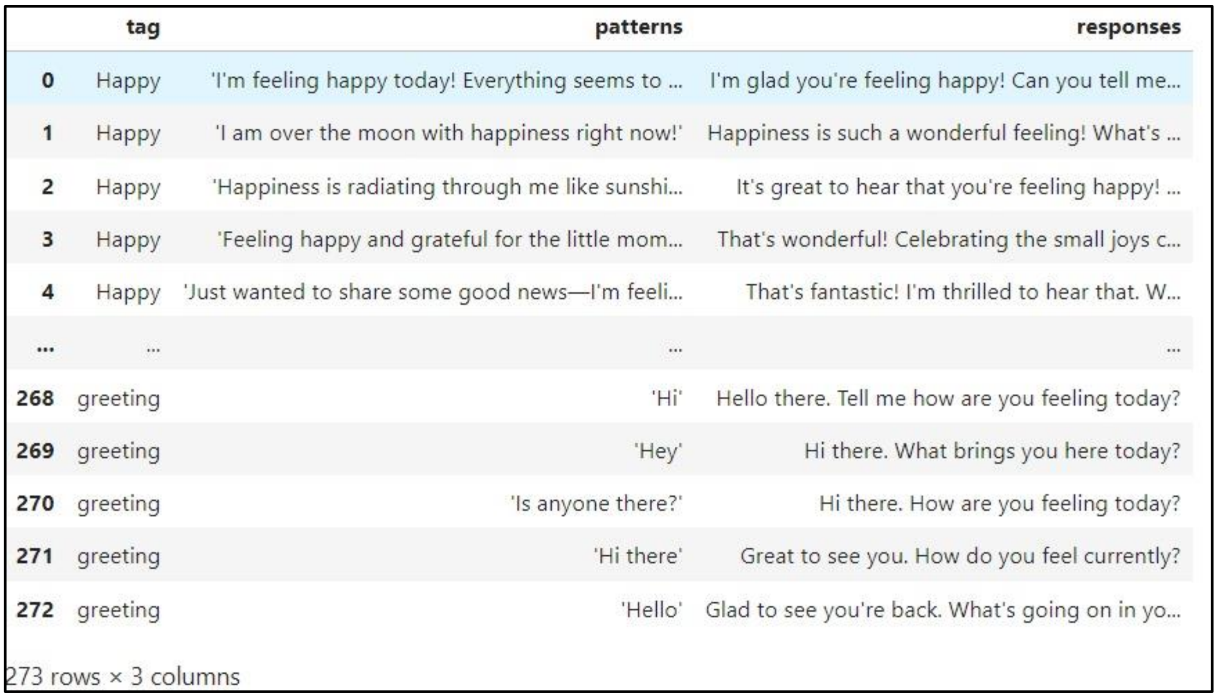
3.1 Prepositional Phrase-attachment resolution



```python
dic = {"tag":[], "patterns":[], "responses":[]}
for _, row in df.iterrows():
    tag = row['Tag']
    patterns = row['Patterns'].strip('[]').split('], [')
    responses = row['Responses'].strip('[]').split('], [')
    min_len = min(len(patterns), len(responses))
    for j in range(min_len):
        dic['tag'].append(tag)
        dic['patterns'].append(patterns[j])
        dic['responses'].append(responses[j])

df = pd.DataFrame.from_dict(dic)
```

Figure 15: Splitting the columns in dataset .xsl file

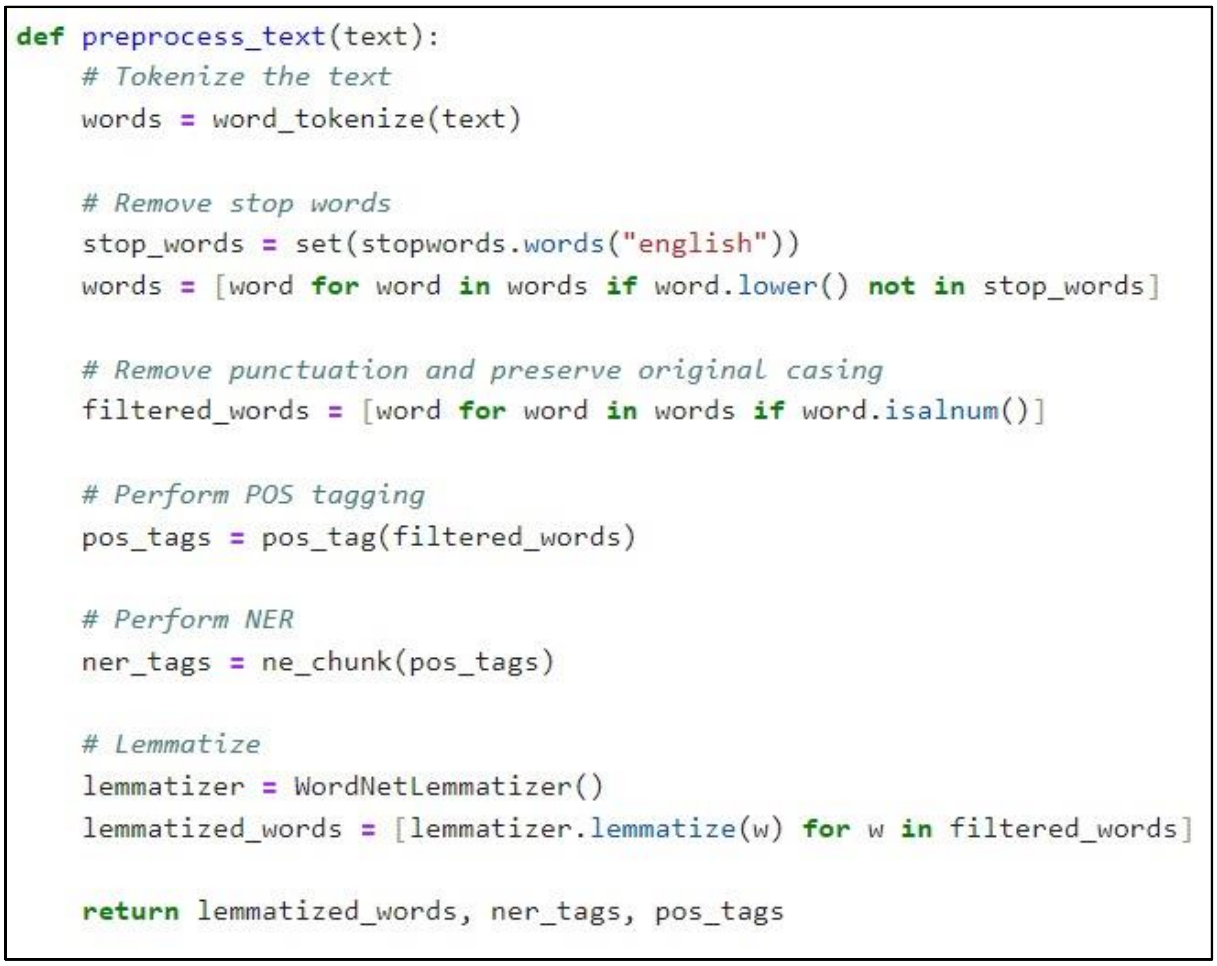
The data in .xsl file is first split into 3 sections: tag, patterns and responses. In each column in pattern and responses, we have more than one data splitting it with [], [], []. Hence, we will split the column into rows based on the data we have as shown in figure 16.



	tag	patterns	responses
0	Happy	'I'm feeling happy today! Everything seems to ...	I'm glad you're feeling happy! Can you tell me...
1	Happy	'I am over the moon with happiness right now!'	Happiness is such a wonderful feeling! What's ...
2	Happy	'Happiness is radiating through me like sunshi...	It's great to hear that you're feeling happy! ...
3	Happy	'Feeling happy and grateful for the little mom...	That's wonderful! Celebrating the small joys c...
4	Happy	'Just wanted to share some good news—I'm feeli...	That's fantastic! I'm thrilled to hear that. W...
...	...	...	...
268	greeting	'Hi'	Hello there. Tell me how are you feeling today?
269	greeting	'Hey'	Hi there. What brings you here today?
270	greeting	'Is anyone there?'	Hi there. How are you feeling today?
271	greeting	'Hi there'	Great to see you. How do you feel currently?
272	greeting	'Hello'	Glad to see you're back. What's going on in yo...

273 rows × 3 columns

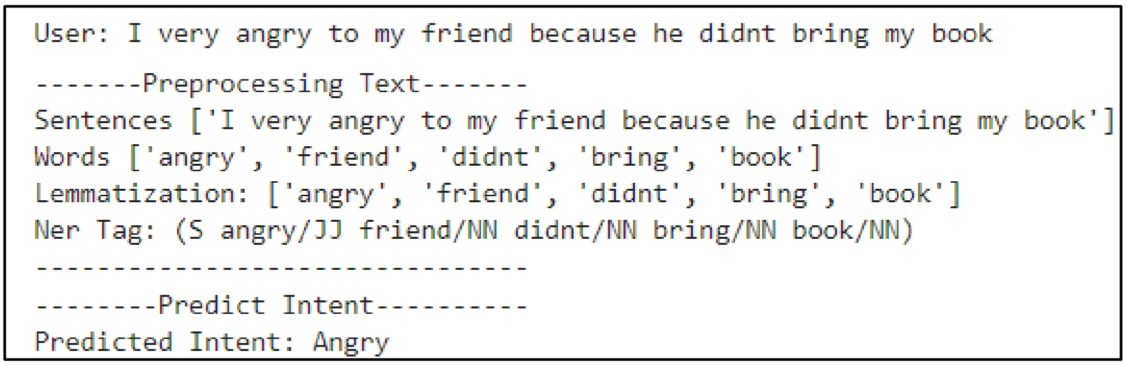
Figure 16: Data after splitting pattern and responses column



```python
def preprocess_text(text):
    # Tokenize the text
    words = word_tokenize(text)

    # Remove stop words
    stop_words = set(stopwords.words("english"))
    words = [word for word in words if word.lower() not in stop_words]

    # Remove punctuation and preserve original casing
    filtered_words = [word for word in words if word.isalnum()]

    # Perform POS tagging
    pos_tags = pos_tag(filtered_words)

    # Perform NER
    ner_tags = ne_chunk(pos_tags)

    # Lemmatize
    lemmatizer = WordNetLemmatizer()
    lemmatized_words = [lemmatizer.lemmatize(w) for w in filtered_words]

    return lemmatized_words, ner_tags, pos_tags
```

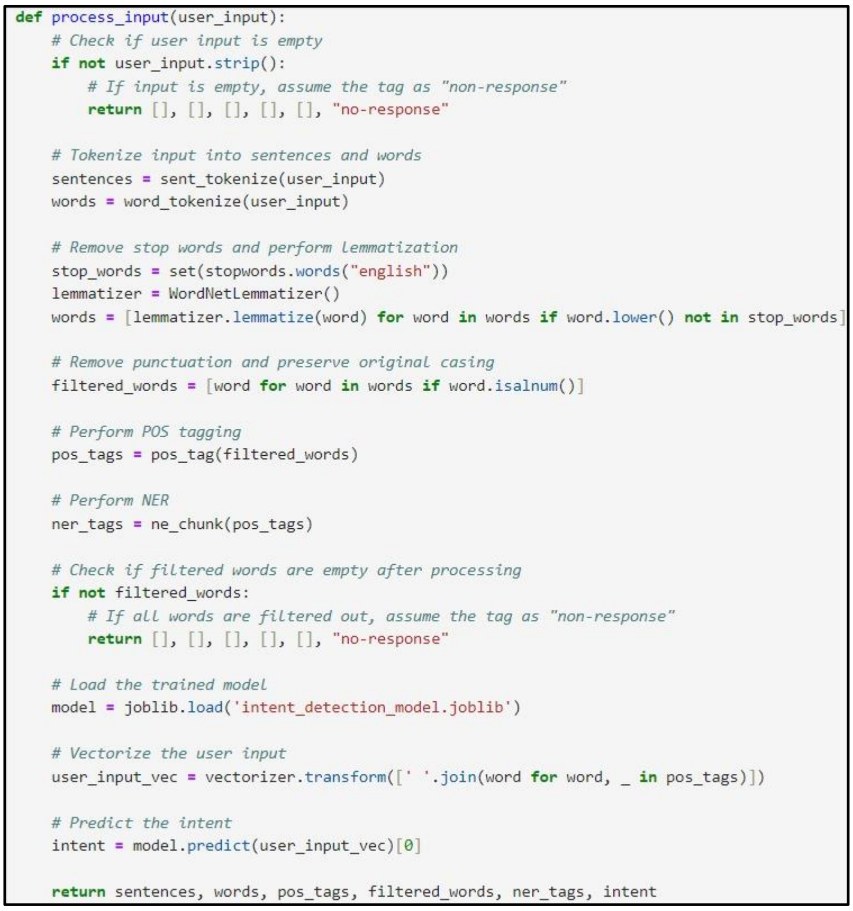
Figure 17: Preprocessing Dataset

As shown in figure 17, after splitting the data, we will preprocess the data with stop_words, filtered_words, pos_tags, ner_tags, lemmatize.



```
User: I very angry to my friend because he didnt bring my book
-------Preprocessing Text-------
Sentences ['I very angry to my friend because he didnt bring my book']
Words ['angry', 'friend', 'didnt', 'bring', 'book']
Lemmatization: ['angry', 'friend', 'didnt', 'bring', 'book']
Ner Tag: (S angry/JJ friend/NN didnt/NN bring/NN book/NN)
--------------------------------
--------Predict Intent----------
Predicted Intent: Angry
```

Figure 18: Prepositional Phrase-Attachment Resolution



```python
def process_input(user_input):
    # Check if user input is empty
    if not user_input.strip():
        # If input is empty, assume the tag as "non-response"
        return [], [], [], [], [], "no-response"

    # Tokenize input into sentences and words
    sentences = sent_tokenize(user_input)
    words = word_tokenize(user_input)

    # Remove stop words and perform lemmatization
    stop_words = set(stopwords.words("english"))
    lemmatizer = WordNetLemmatizer()
    words = [lemmatizer.lemmatize(word) for word in words if word.lower() not in stop_words]

    # Remove punctuation and preserve original casing
    filtered_words = [word for word in words if word.isalnum()]

    # Perform POS tagging
    pos_tags = pos_tag(filtered_words)

    # Perform NER
    ner_tags = ne_chunk(pos_tags)

    # Check if filtered words are empty after processing
    if not filtered_words:
        # If all words are filtered out, assume the tag as "non-response"
        return [], [], [], [], [], "no-response"

    # Load the trained model
    model = joblib.load('intent_detection_model.joblib')

    # Vectorize the user input
    user_input_vec = vectorizer.transform([' '.join(word for word, _ in pos_tags)])

    # Predict the intent
    intent = model.predict(user_input_vec)[0]

    return sentences, words, pos_tags, filtered_words, ner_tags, intent
```

Figure 19: Processing User Input

3.1.1 Algorithm

- User input: First we will get the user input by prompting
- Tokenization: The user input is then tokenized from sentences into words. This step breaks down the input text into individual words.
- Stopword Removal: Removed all the "the", "is", "and" from the tokenized words.
- Lemmatization: The remaining words are lemmatized, which is changes the word back into their root form.
- Part-of-Speech (POS) Tagging: POS tagging is performed on the lemmatized words. These grammatical categories noun, verb, adjective to each word in the input text.
- Named Entity Recognition (NER): NER is applied to the POS-tagged words. This step is used to identifies named entities such as person names, organization names, locations.
- Intent Prediction: Finally, the preprocessed input is passed through an AI model to predict the intent of the sentence.

3.1.2 Data Structure

- In Figure 17, show that all the preprocessed input Tokenization, Stopword Removal, Lemmatization, Part-of-Speech (POS) Tagging and Named Entity Recognition (NER) is performed and displayed before getting it to the ai model which show the intent angry.

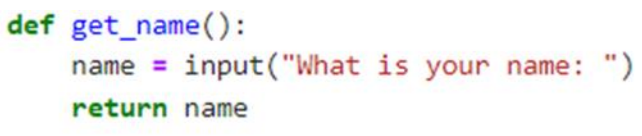
3.1.3 Data Required

- Tokenization, stopword list, and lemmatization, POS tagging and NER. The AI model for intent prediction is trained on tag/intent containing pattern with their responses.

3.1.4 Explanation

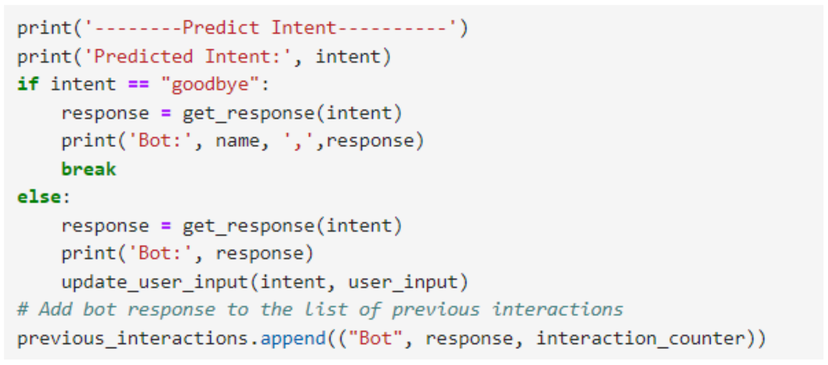
- We have a function call process_input which is used to preprocess the user input before running through the ai model to predict the intent but before that we will tokenize the input then remove stop words and finally lemmatize then we will do pos tag. After pos tagging we will do NER. After that we run it through the ai model to get the intent of the sentence.

3.2 Query the user for their name and remember their name



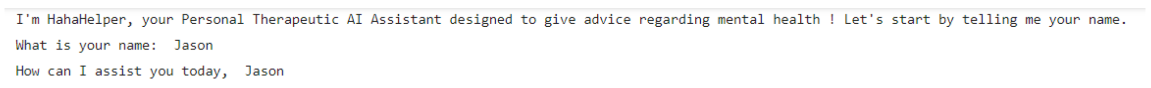
```
def get_name():
    name = input("What is your name: ")
    return name
```

Figure 20: Print function to get user name



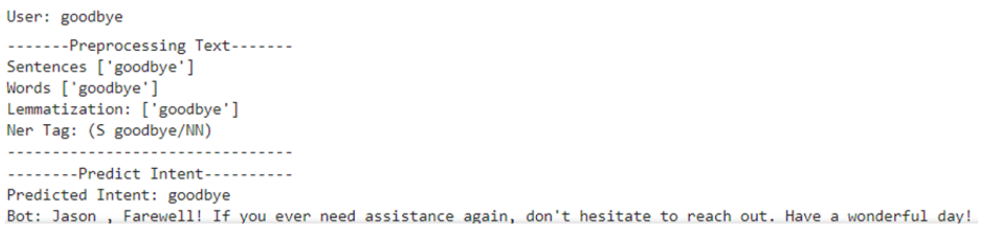
```
print('--------Predict Intent----------')
print('Predicted Intent:', intent)
if intent == "goodbye":
    response = get_response(intent)
    print('Bot:', name, ',',response)
    break
else:
    response = get_response(intent)
    print('Bot:', response)
    update_user_input(intent, user_input)
# Add bot response to the list of previous interactions
previous_interactions.append(("Bot", response, interaction_counter))
```

Figure 21: Exit chat if user intent is goodbye, else continue to loop chat bot



```
I'm HahaHelper, your Personal Therapeutic AI Assistant designed to give advice regarding mental health ! Let's start by telling me your name.
What is your name:  Jason
How can I assist you today,  Jason
```

Figure 22: First prompt when using chatbot



```
User: goodbye
-------Preprocessing Text-------
Sentences ['goodbye']
Words ['goodbye']
Lemmatization: ['goodbye']
Ner Tag: (S goodbye/NN)
--------------------------------
--------Predict Intent----------
Predicted Intent: goodbye
Bot: Jason , Farewell! If you ever need assistance again, don't hesitate to reach out. Have a wonderful day!
```

Figure 23: Prompt when user exit chat

3.2.1 Algorithm

- Prompting the user for their name: Display a message prompt user to input their name.
- Storing the name: Store the user's name using a variable to hold the name.
- Displaying name at the beginning: Display a welcome message to the user, to show that their name has been stored.
- Displaying name at the end: Display an exit message to the user.

3.2.2 Data structured

- In Figure 20, show using a function call get_name to ask user to input their and store in the variable call name. This is to ensure the variable name can use in future.

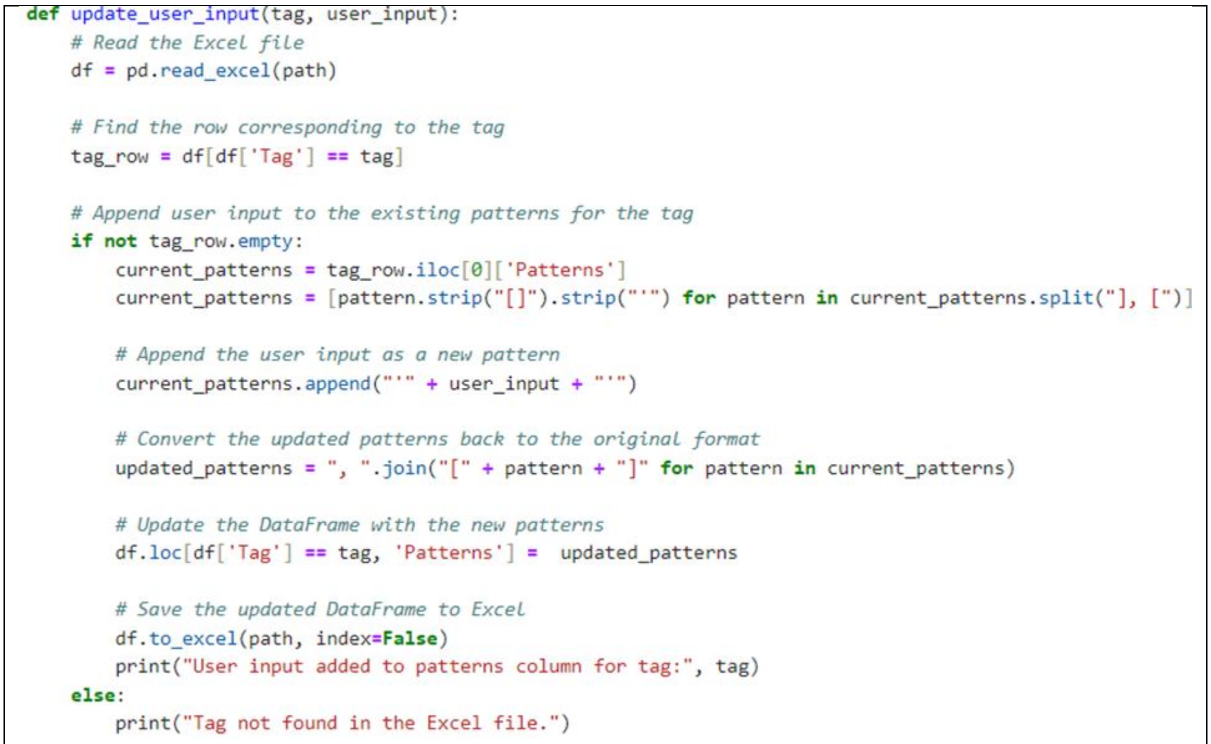
3.2.3 Data required

- Input Data: The only data required is the user's input, in this case which will be their name. This name is input during program execution and doesn't need to be predefined.

3.2.4 Explanation

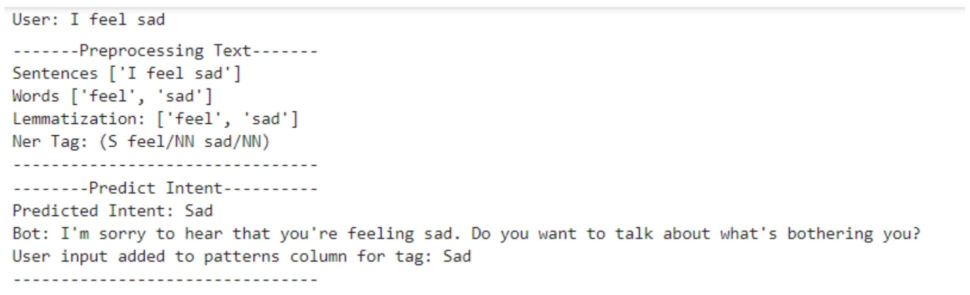
- At the start of the chatbot we will ask the user for their name which we will save inside a variable. After getting the name we will use it to greet the user. When the user finish talking with us and said goodbye we will once again mention the username to greet farewell.

3.3 Update knowledge of chatbot



```python
def update_user_input(tag, user_input):
    # Read the Excel file
    df = pd.read_excel(path)

    # Find the row corresponding to the tag
    tag_row = df[df['Tag'] == tag]

    # Append user input to the existing patterns for the tag
    if not tag_row.empty:
        current_patterns = tag_row.iloc[0]['Patterns']
        current_patterns = [pattern.strip("[]").strip("'") for pattern in current_patterns.split("], [")]

        # Append the user input as a new pattern
        current_patterns.append("'" + user_input + "'")

        # Convert the updated patterns back to the original format
        updated_patterns = ", ".join("[" + pattern + "]" for pattern in current_patterns)

        # Update the DataFrame with the new patterns
        df.loc[df['Tag'] == tag, 'Patterns'] =  updated_patterns

        # Save the updated DataFrame to Excel
        df.to_excel(path, index=False)
        print("User input added to patterns column for tag:", tag)
    else:
        print("Tag not found in the Excel file.")
```

Figure 24: Updating knowledge of Chatbot



```
User: I feel sad
-------Preprocessing Text-------
Sentences ['I feel sad']
Words ['feel', 'sad']
Lemmatization: ['feel', 'sad']
Ner Tag: (S feel/NN sad/NN)
-------------------------------
--------Predict Intent----------
Predicted Intent: Sad
Bot: I'm sorry to hear that you're feeling sad. Do you want to talk about what's bothering you?
User input added to patterns column for tag: Sad
-------------------------------
```

Figure 25: Use Case scenario for updating knowledge

3.3.1 Algorithm

- Predict Intent: After receiving user input and preprocess it then we put it through the AI model to predict the intent of the input text.
- Save Input into Respective Intent Category pattern: Determine the predicted intent category for the user input then the user input is saved into that intent pattern.
- Data Storage: We use an Excel file as the data storage to store user input, intent and response.

3.3.2 Data Structures

- The user input is saved as pattern base on the intent can been seen inside the Excel file.

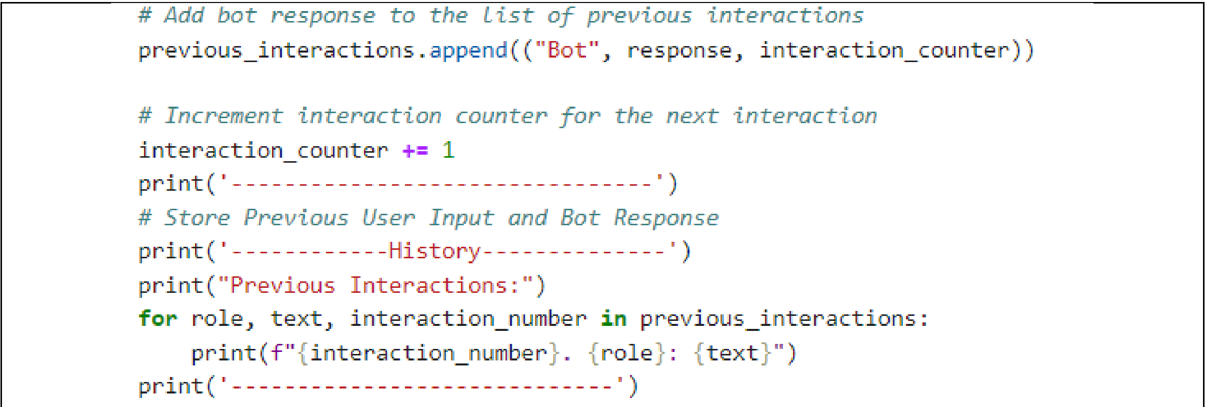
3.3.3 Data Required

- Excel file it contains columns for intent/tag categories and their associated input patterns and response.

3.3.4 Explanation

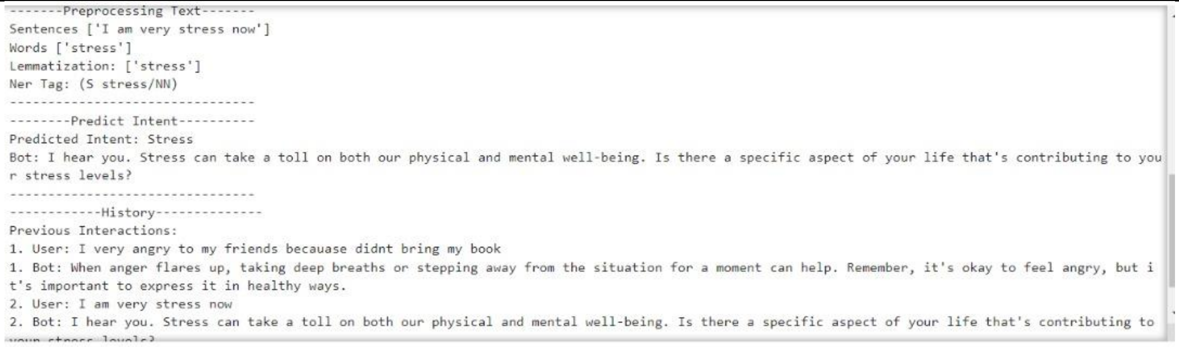
- Each time user input we will save the input into excel file which is our data storage. Each save input are saved into their respective predicted intent for above example when the user enters, I feel sad which the predicted intent is grief, so this user input is then saved into the tag grief pattern.

3.4 Keep history of chat



```
# Add bot response to the list of previous interactions
previous_interactions.append(("Bot", response, interaction_counter))

# Increment interaction counter for the next interaction
interaction_counter += 1
print('--------------------------------')
# Store Previous User Input and Bot Response
print('------------History-------------')
print("Previous Interactions:")
for role, text, interaction_number in previous_interactions:
    print(f"{interaction_number}. {role}: {text}")
print('-----------------------------')
```

Figure 26: Keep History of Chats



```
-------Preprocessing Text-------
Sentences ['I am very stress now']
Words ['stress']
Lemmatization: ['stress']
Ner Tag: (S stress/NN)
--------------------------------
--------Predict Intent----------
Predicted Intent: Stress
Bot: I hear you. Stress can take a toll on both our physical and mental well-being. Is there a specific aspect of your life that's contributing to you
r stress levels?
--------------------------------
------------History-------------
Previous Interactions:
1. User: I very angry to my friends becauase didnt bring my book
1. Bot: When anger flares up, taking deep breaths or stepping away from the situation for a moment can help. Remember, it's okay to feel angry, but i
t's important to express it in healthy ways.
2. User: I am very stress now
2. Bot: I hear you. Stress can take a toll on both our physical and mental well-being. Is there a specific aspect of your life that's contributing to
```

Figure 27: Use case of keeping history of chat

3.4.1 Algorithm

- Initialization: Initialize an empty list to store the chat history.
- Chat Loop: Enter a chat interaction loop. Prompt the user to enter their input. Append the user input to the chat history list. The user input is then used to be preprocess then predict intent which give back respond the bot's response is then append to the chat history list.
- Display History: After the chat loop ends display the chat history to the user.

3.4.2 Data Structures

- We use a list to store the chat history for both user and bot. Which we display at the chat.

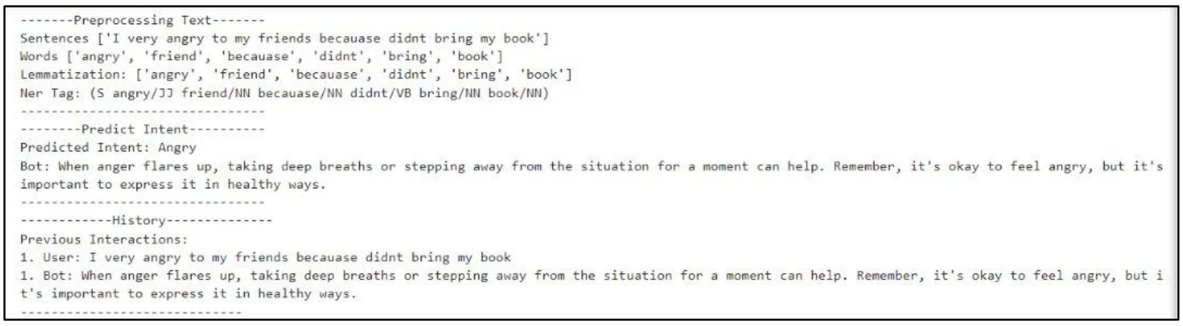
3.4.3 Explanation

- Each time the user enters inside the chat after entering their name all text enter is saved inside a list. The response of the bot for that input is also stored inside the list. The list is then be displayed for the user to see history.

4 Summary

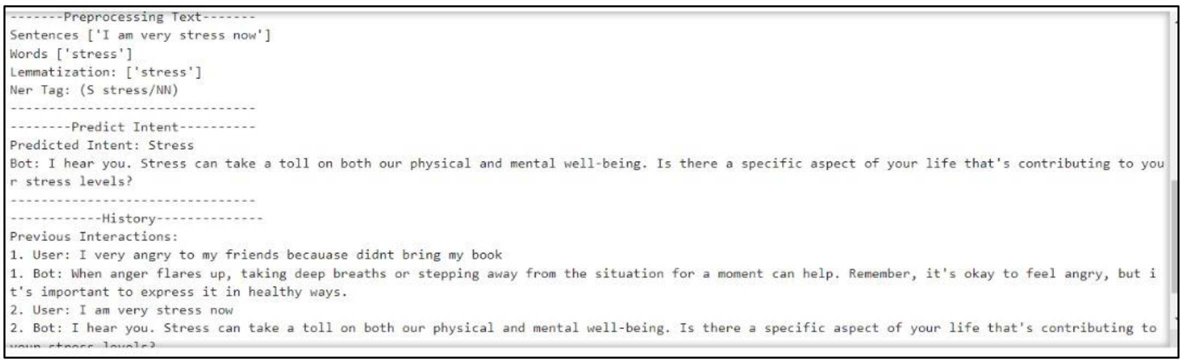
4.1 Results and Achievements

By completing the project, we have successfully developed a Natural Language Processing Chatbot Therapy Assistant modal. The chatbot can be used to communicate with user and give them assist or advice base on their cases. Below shows some examples of the chatbots operations.



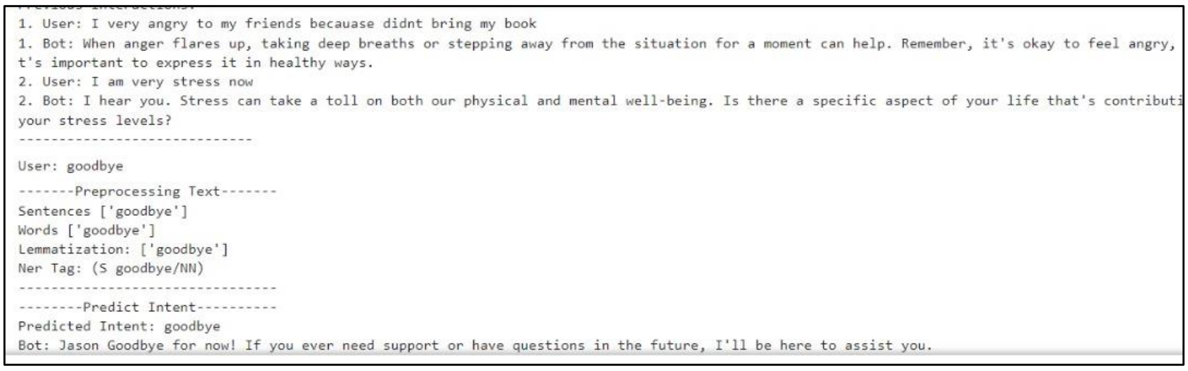
```
-------Preprocessing Text-------
Sentences ['I very angry to my friends becauase didnt bring my book']
Words ['angry', 'friend', 'becauase', 'didnt', 'bring', 'book']
Lemmatization: ['angry', 'friend', 'becauase', 'didnt', 'bring', 'book']
Ner Tag: (S angry/JJ friend/NN becauase/NN didnt/VB bring/NN book/NN)
--------------------------------
--------Predict Intent----------
Predicted Intent: Angry
Bot: When anger flares up, taking deep breaths or stepping away from the situation for a moment can help. Remember, it's okay to feel angry, but it's
important to express it in healthy ways.
--------------------------------
------------History--------------
Previous Interactions:
1. User: I very angry to my friends becauase didnt bring my book
1. Bot: When anger flares up, taking deep breaths or stepping away from the situation for a moment can help. Remember, it's okay to feel angry, but i
t's important to express it in healthy ways.
-----------------------------
```

Figure 28:Example of test case



```
-------Preprocessing Text-------
Sentences ['I am very stress now']
Words ['stress']
Lemmatization: ['stress']
Ner Tag: (S stress/NN)
--------------------------------
--------Predict Intent----------
Predicted Intent: Stress
Bot: I hear you. Stress can take a toll on both our physical and mental well-being. Is there a specific aspect of your life that's contributing to you
r stress levels?
--------------------------------
------------History--------------
Previous Interactions:
1. User: I very angry to my friends becauase didnt bring my book
1. Bot: When anger flares up, taking deep breaths or stepping away from the situation for a moment can help. Remember, it's okay to feel angry, but i
t's important to express it in healthy ways.
2. User: I am very stress now
2. Bot: I hear you. Stress can take a toll on both our physical and mental well-being. Is there a specific aspect of your life that's contributing to
```

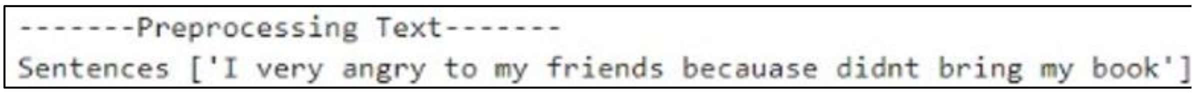
Figure 29: Example of test case



```
1. User: I very angry to my friends becauase didnt bring my book
1. Bot: When anger flares up, taking deep breaths or stepping away from the situation for a moment can help. Remember, it's okay to feel angry,
t's important to express it in healthy ways.
2. User: I am very stress now
2. Bot: I hear you. Stress can take a toll on both our physical and mental well-being. Is there a specific aspect of your life that's contributi
your stress levels?
-----------------------------

User: goodbye
-------Preprocessing Text-------
Sentences ['goodbye']
Words ['goodbye']
Lemmatization: ['goodbye']
Ner Tag: (S goodbye/NN)
--------------------------------
--------Predict Intent----------
Predicted Intent: goodbye
Bot: Jason Goodbye for now! If you ever need support or have questions in the future, I'll be here to assist you.
```

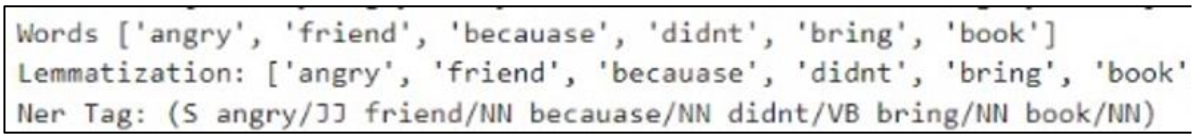
Figure 30: Example of test case

The figure above shows the example of how user input and how the chatbot will response to it. To futhuremore explain, we split the examples and shows on below.



```
-------Preprocessing Text-------
Sentences ['I very angry to my friends becauase didnt bring my book']
```

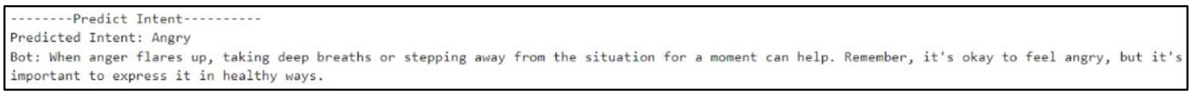
Figure 31: sentences input by user



```
Words ['angry', 'friend', 'becauase', 'didnt', 'bring', 'book']
Lemmatization: ['angry', 'friend', 'becauase', 'didnt', 'bring', 'book'
Ner Tag: (S angry/JJ friend/NN becauase/NN didnt/VB bring/NN book/NN)
```

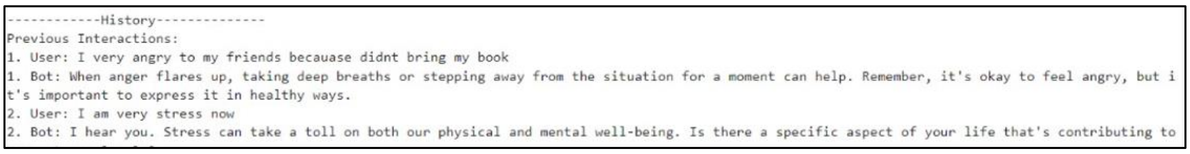
Figure 32: preprocessing text

As you can see from the figure above, when user input a sentence, the senteces will be preprocessing as the first steps. It goes with some steps include tokenization, remove stopwards, remove punctuation (filter words), lemmatization, and Name Entity Recognition (Ner tags). The tokenization, remove stopwards and remove punctuation is to get the keypoint or keywords in the sentences. Lemmatization is the process to breaks down a word into its dictionary form, or lemma, and identifies its part of speech and dictionary headword. For the Net tag shows in the figure representing the Name Entity Recognition process which is to identify and classify named entities within text into predefined categories.



```
--------Predict Intent----------
Predicted Intent: Angry
Bot: When anger flares up, taking deep breaths or stepping away from the situation for a moment can help. Remember, it's okay to feel angry, but it's
important to express it in healthy ways.
```

Figure 33: predict intent



```
------------History--------------
Previous Interactions:
1. User: I very angry to my friends becauase didnt bring my book
1. Bot: When anger flares up, taking deep breaths or stepping away from the situation for a moment can help. Remember, it's okay to feel angry, but i
t's important to express it in healthy ways.
2. User: I am very stress now
2. Bot: I hear you. Stress can take a toll on both our physical and mental well-being. Is there a specific aspect of your life that's contributing to
```

Figure 34:History

After the preprocessing part done, the chatbot will predict the intent base on preprocessing data and response to the user by refering to the response files. Also, to allow user able to communicate continuosly with chatbot and save chat history for futhur use, we do include history features for the chatbot to save the chat.

4.2 Comparison of Modified Chatbot with an Existing Chatbot

To compare the modified chatbot with existing chatbot, we would like to compare our chatbot with the meaningful chatbot in the worlds – Eliza. Eliza is an early natural language processing computer program developed to explore communication between humans and machines. It using keyword matching techniques to identify patterns within user inputs. It scanned the user's input and identify the specific keywords or phrases that matched predefined pattern or intent rules. Eliza are not process to understand the whole sentence but just identify the keywords or phrases to understand the user's intent or emotional state.

On this point, our modified chatbot was processing the whole sentence for understanding the meaning rather than just identify the keywords or phrases in the sentence. Our modified chatbot using advanced natural language processing (NLP) techniques to analyze the entire sentence for meaning and context. The steps of analysing the entire sentence includes Tokenization, remove stopword, remove punctuation, pos tagging, ner tagging and lemmatization.

The advantage of our chatbot using sentence processing is it can enhance deeper understanding of user inputs so that it can generate more accurate and relevant responses. Also, chatbot can understand the overall meaning of the system rather than just understand the keywords in the sentence. Other than that, by analysing the sentence, chatbot can still identify the user preferences and emotion to generate more personalized responses to user. Unfortunately, it bring some challenges as well for the sentence processing. It may encounter issues with complex or ambiguos input such as sarcasm, slang, or grammatical errors.

5 References

1] X. Huang, “CHATBOT: DESIGN, ARCHITECUTRE, AND APPLICATIONS,” University of Pennsylvania, Pennsylvania, 2021.

2] “What Is a Chatbot,” Oracle, [Online]. Available: https://www.oracle.com/chatbots/what-is-a-chatbot/. [Accessed 6 March 2024].

3] K. Brush, “What is a chatbot?,” TechTarget, November 2021. [Online]. Available: https://www.techtarget.com/searchcustomerexperience/definition/chatbot#:~:text=A%20chatbot%20is%20a%20software,assistants%20to%20handle%20simple%20tasks.. [Accessed 6 March 2024].

4] “AI History : Eliza,” AIGeneration.BOT, 5 March 2023. [Online]. Available: https://aigeneration.blog/2023/03/05/ai-history-eliza/. [Accessed 7 March 2024].

5] “Named Entity Recognition: The Mechanism, Methods, Use Cases, and Implementation Tips,” altexsoft, 1 November 2023. [Online]. Available: https://www.altexsoft.com/blog/named-entity-recognition/. [Accessed 7 March 2024].

6] “What Is Cortana? Definition, Working, Features, and Challenges,” spiceworks, 11 July 2022. [Online]. Available: https://www.spiceworks.com/tech/artificial-intelligence/articles/what-is-cortana/. [Accessed 8 March 2024].

7] “CORTANA-INTELLIGENT PERSONAL DIGITAL ASSISTANT: A REVIEW,” ResearchGate, August 2017. [Online]. Available: https://www.researchgate.net/publication/319617167_CORTANA-

INTELLIGENT_PERSONAL_DIGITAL_ASSISTANT_A_REVIEW. [Accessed 9 March 2024].

8] “Siri,” TechTarget, February 2023. [Online]. Available: https://www.techtarget.com/searchmobilecomputing/definition/Siri. [Accessed 8 March 2024].

9] “'Hey, Siri:' Inside Apple's speech AI and the technology behind it,” Business Insider, 15 January 2024. [Online]. Available: https://www.businessinsider.com/siri-apple. [Accessed 2024 March 9].

10] “The Natural Language Processing In Siri,” UKEssays.com, 18 April 2017. [Online]. Available: https://www.ukessays.com/essays/english-language/the-natural-language-processing-in-siri-english-language-essay.php. [Accessed 10 March 2024].