IIDT(INTERNATIONAL INSTITUTE OF DIGITAL TECHNOLOGIES)

CHATBOT FOR SIMPLE QUESTIONS USING DEEP LEARNING

In partial fulfilment for the award of the degree of

BACHELOR OF TECHNOLOGY

IN

COMPUTER AND SCIENCE ENGINEERING

(ARTIFICIAL INTELLIGENCE AND MACHINE LEARING)

CHEBROLU ENGINEERING COLLEGE

Submitted by

Angusamy Sanjay

(21HU1A4207)

TABLE OF CONTENTS

1. ABSTRACT

A chatbot is a computer program that simulates and processes human conversation allowing humans to interact with digital devices as if they were communicating with a real person. Chatbots can be as simple as rudimentary programs that answer a simple query with a single line response, or as sophisticated as digital assistants that learn and evolve to deliver incresing levels of personalization as they gather and process information.

Chatbots helps in customer support, generating information, user navigation, social media marketing, instant messaging etc., Organizations looking to increase sales or service productivity may adopt chatbots for time saving and efficiency, as artificial intelligece (AI) chatbots can converse with users and answer recurring questions. As consumers move away from traditional forms of communication, many experts expect chat-based communication methods to rise. Organizations increasingly use chat-based virtual assistants to handle simple tasks, allowing human agents to focus on other responsibilities.

In this project we will use python to build a simple chatbot that can interact with users. Here we make use of AI and Natural Language Processing in Python to create and train a chatbot to respond appropriately to an user requests.

Natural Language Processing (NLP) makes it possible for computers to understand the human language. Behind the scenes, NLP analyzes the grammatical structure of sentences and the individual meaning of words, then uses algorthims to extract meaning and deliver outputs. In other words, it makes sense of human language so that it can automatically perform different tasks. A well-known application of NLP is chatbots. They help support team solve issues by understanding common language requests and responding automatically.

2. INTRODUCTION

2.1 OVERVIEW OF THE PROJECT:-

The chatbot is designed to handle frequently asked questions (FAQs) and provide relevant answers in real-time. It focuses on understanding user intent, retrieving appropriate information, and delivering concise responses. The project encompasses the development of a user-friendly interface, robust backend support, and effective NLP models to ensure the chatbot's efficiency. The primary objective is to reduce the workload on human support staff by automating responses to common queries.

The proposed chatbot will be capable of understanding and responding to user queries on a range of topics, including but not limited to history, science, technology, health, and entertainment. The chatbot will utilize natural language processing (NLP) and machine learning algorithms to analyze user input and generate relevant responses. The system will be designed to provide accurate and concise answers, making it an ideal tool for users seeking quick information on various topics.

2.2 Objective:-

The main aim of the project is to provide the users with a platform to get their questions andqueries answered in a easy and simple way. This helps users avoid having to wait on a call or wait for an E-mail to get their queries resolved. This chatbot is a way for users to get their queries answered without having to go through a customer service call or having to talk to anautomated recording to get their concerns cleared. Here we train the chatbot to

recognizemultiple types of words and phrases related to a particular question in order to provide as proper answers to the users question as possible. Chatbots allow businesses to connect with customers in a personal way without the expense of human representatives. For example, many of the questions or issues customers have are common and easily answered. That's why companies create FAQs and troubleshooting guides.

2.3 Scope:-

The goal of the chatbot is to respond as precisely as possible to client questions. Theconsumer should be able to receive the clearest possible responses to their inquiries. The Chatbot Scope of Work Template can help you: Define the objectives and goals of your chatbot project. Outline the specific features and functionalities you want your chatbot to have. Set clear expectations and timelines for the development and implementation process.

2.4 Importance:-

- **Reduce customer waiting time** – 21% of consumers **see chatbots as the easiest way to contact a business**. Bots are a smarter way to ensure that customers receive the immediate response that they are looking for without making them wait in a queue.
- **24×7 availability** – A B2B or B2B Chatbot can be always available to engage customers with immediate answers to the common questions asked by them. The top potential benefit of using chatbots is 24-hour customer service.
- **Better customer engagement** – Conversational bots can engage customers round the clock by starting proactive conservation and offering personalized recommendations that boost customer experience.
- **Save customer service costs** – Chatbots will help businesses save more than billion per year. Bots can be easily scaled which saves customer support costs of hiring more resources, infrastructure costs, etc.
- **Automate lead qualification & sales** – You can automate your sales funnel with chatbots to prequalify leads and direct them to the right team for further nurturing. Being able to engage customers instantly increases the number of leads and conversion rates.

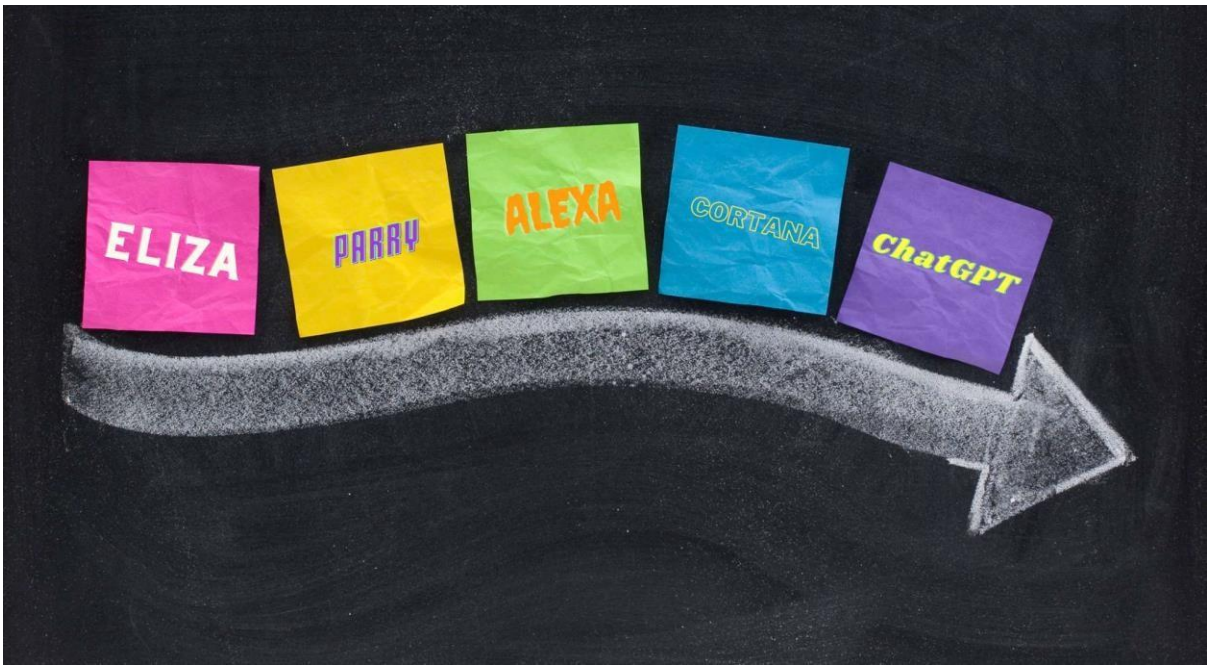
2.5 HISTORY OF CHATBOTS:-

The first chatbot ever was developed by MIT professor Joseph Weizenbaum in the 1960s. It was called ELIZA. You'll read more about ELIZA and other popular chatbots that were developed in the second half of the 20th century later on.

In the year 2009, a company called WeChat in China created a more advanced Chatbot. Since its launch, WeChat has conquered the hearts of many users who demonstrate an unwavering loyalty to it. It is a highly thriving social media platform.

Though it has implications and is less performant than today's messaging apps such as Facebook Messenger, Slack, and Telegram, it doesn't mean that you cannot construct a very smart bot on WeChat. Chumen Wenwen Company, founded in 2012 by a former Google employee, has built a very sophisticated bot running on WeChat.

Early in 2016, we saw the intro of the first wave of artificial data technology in the design of chatbots. Social media platforms like Facebook enabled developers to build a chatbot for their trademark or service so that customers could carry out some of their daily actions from inside their messaging platform.



3. LITERATURE SURVEY

3.1 INTRODUCTION:-

A literature survey or a literature review in a project report is that section which shows various analysis and research made in the field of your interest and the results already published, taking into account the various parameters of the project and the extent of project. Once the programmers start building the tool programmers need a lot of external support. This support can be obtained from senior programmers, books or from the websites. It is the most important part of your report as it gives you a direction in the area of your research. It helps you set a goal for your analysis - thus giving you your problem of statement. Literature survey is the most important sector in the software development process. Before developing the tools and the associated designing the software it is necessary to determine the survey the time factor, resource requirement etc., The consumer needs regarding online customer service differs from person to person. The needs are also based off each persons personal needs. We need to identify and anticipate these needs in order to completely and accurately meet them.

3.2 LITERATURE SURVEY:-

MEDICAL ASSISTANT CHATBOTS

In this paper, Prakhar Srivastava etc. This Paper explains automated medical chatbots are conversationally developed with technology in mind, with the potential to minimise efforts to increase access to medical services and knowledge and reduce efforts to healthcare expenditures. We developed a diagnosis bot that converses with patients about their medical issues and questions in order to deliver a personalised diagnosis based on their identified manifestation and profile. Our chatbot system is qualified to identify symptoms from user inputs with a standard precision of 65%. Using these extracted diagnosed symptoms correct symptoms were identified with a recall of 65% and a precision of 71%. Finally, the chatbot returned the expected diagnosis for further more operations. The fact that a medical chatbot can diagnose patients with a fair degree of accuracy using straightforward symptom analysis and a conversational approach indicates that a successful spoken language medical bot might be feasible.Moreover, the relative effectiveness of this bot indicates that more proceeds automated medical products may flourish to serve a bigger role in healthcare.

CHATBOTS IN VARIOUS FIELDS

In this paper, Adamopoulou etc. The author explains the use of chatbots evolved rapidly in numerous fields in recent years,including Marketing, Supporting Systems, Education, Health Care, Cultural Her-itage, and Entertainment. In this paper, we first give a historical summary of how interest in chatbots has changed throughout the world. Next, we go over the reasons why people use chatbots and explain how valuable they are in a variety of contexts. Additionally, we emphasise the influence of societal stereotypes on the design of chatbots. We continue on to a classification of chatbots based on several criteria, such as the area of knowledge they refer to, the need they serve, and others, after delineating key technological aspects.Furthermore, we present the generalarchitecture of modern chatbots while also mentioning the main platforms for their creation. Our engagement with the subject so far, reassures us of the prospects of chatbots and encourages us to study them in greater extent and depth.

AI BASED CHATBOTS

In this paper,Parth Thosani etc. This paper demonstrates An interactive AI tool called a chatbot attempts to mimic human behaviour by interpreting input and responding in either text or audio format. These days, chatbots are employed to conduct digital communication effectively.Our concept introduces a new field of research in chatbot communication. This essay demonstrates An interactive AI tool called a chatbot attempts to mimic human behaviour by interpreting input and responding in either text or audio format. These days, chatbots are employed to conduct digital communication effectively. The majority of chatbots nowadays successfully carry out the needed duty, however there is something to note. Conversations between a person and a chatbot frequently repeat themselves. Because technology has always reduced human effort, a multi-agent system with a chatbot serving as a middleman between the user and the outside world is proposed. It is comparable to having a personal assistant who is aware of the user's needs. Now that it has this understanding of the user, it is able to make decisions on the user's behalf.This chatbot has something unique, i.e. it understands the user and their requirements. It is comparable to having a personal assistant who is aware of the user's needs. Now that it has this understanding of the user, it is able to make decisions on the user's behalf.Our concept of considering one task that is done with the help of a chatbot is demonstrated. The proposed system adapts and acts accordingly on

behalf of the user.

ALGORITHM

Description & Steps:-

Step 1: Text Preprocessing

- Tokenization: split the user's input into individual words or tokens.
- Stopword removal: remove common words like "the", "and", etc. that don't add much value to the conversation.
- Stemming or Lemmatization: reduce words to their base form (e.g., "running" becomes "run").

Step 2: Intent Identification

- Intent classification: use a machine learning model to classify the user's input into a specific intent category (e.g., booking a flight, asking for weather, etc.).
- Named Entity Recognition (NER): extract specific entities like names, locations, and dates from the user's input.

Step 3: Dialogue Management

- State machine: use a state machine to manage the conversation flow based on the user's input and the chatbot's responses.
- Response generation: use a template-based approach or a generative model to generate a response to the user's input.

Step 4: Response Post-processing

- Spell checking: check the response for spelling errors.
- Grammar checking: check the response for grammatical errors.
- Tone and style adjustment: adjust the tone and style of the response to match the chatbot's personality.

4.2 Flow Chart:-

[Start]

|

[Initialize Components]

|

[Set Up GUI]

|

[Event Handling: User Sends Message]

|

[send() Function]

|

[Retrieve User Input]

|

[Check if Message is Not Empty]

|

[Call chatbot_response(msg)]

|

[chatbot_response(msg) Function]

|

[predict_class(msg, model)]

|

[Preprocess Message]

|

[Predict Intent]

|

[Filter Predictions]

|

[Return Intent Predictions]

|

[getResponse(ints, intents_json)]

|

[Retrieve Response]

|

[Return Response]

|

[Display Bot Response in GUI]

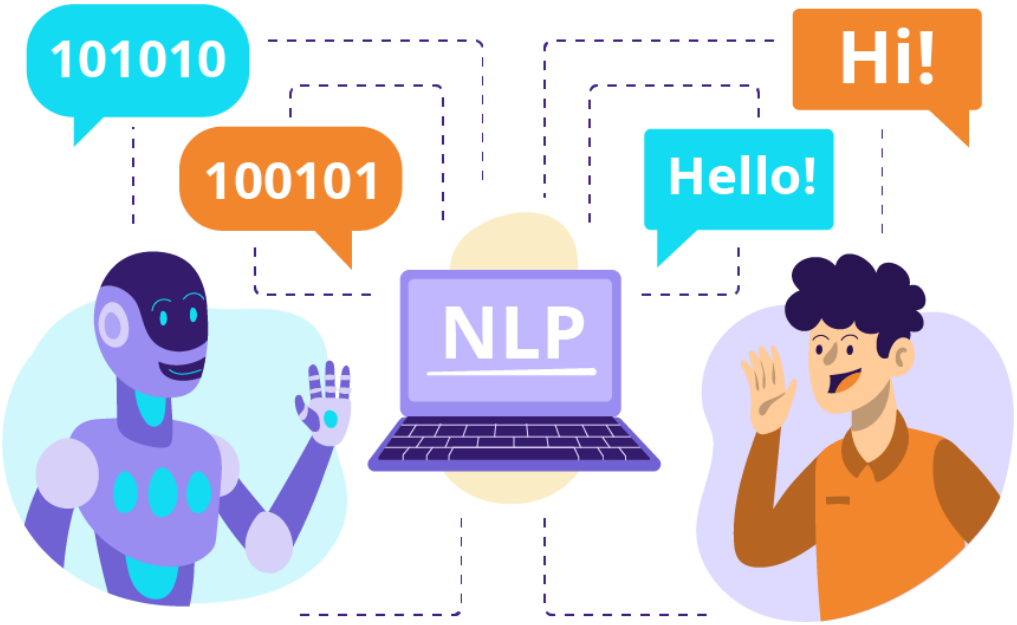
|

[End]

5. SYSTEM DESIGN

5.1 Natural Language Processing (NLP):-

The meaning of NLP is Natural Language Processing (NLP) which is a fascinating and rapidly evolving field that intersects computer science, artificial intelligence, and linguistics. NLP focuses on the interaction between computers and human language, enabling machines to understand, interpret, and generate human language in a way that is both meaningful and useful. With the increasing volume of text data generated every day, from social media posts to research articles, NLP has become an essential tool for extracting valuable insights and automating various tasks.



Natural language processing (NLP) is a field of computer science and a subfield of artificial intelligence that aims to make computers understand human language. NLP uses computational linguistics, which is the study of how language works, and various models based on statistics, machine learning, and deep learning. These technologies allow computers to analyze and process text or voice data, and to grasp their full meaning, including the speaker's or writer's intentions and emotions.

Natural Language ToolKit(NLTK):-

he Natural Language Toolkit (NLTK) is a Python programming environment for creating applications for statistical natural language processing (NLP). It includes language processing libraries for tokenization, parsing, classification, stemming, labeling, and semantic reasoning. It also comes with a curriculum and even a book describing the usually presented language processing jobs NLTK offers, together with visual demos, including experimental data repositories.

NLTK supports a wide range of languages, not just English. It provides tokenization, stemming, and morphological analysis tools for languages such as Arabic, Chinese, Dutch, French, German, Hindi, Italian, Japanese, Portuguese, Russian, Spanish, and more. While NLTK is a powerful toolkit in its own right, it can also be used in conjunction with other machine learning libraries such as sci-kit-learn and TensorFlow. This allows for even more sophisticated NLP applications, such as deep learning-based language modeling.

Advantages of NLP:-

- Analyse larger data sets.
- Provide a more objective analysis.
- Streamline daily processes.
- Improve customer experience.
- Extract actionable insights.

Disadvantages of NLP:-

- NLP may not show context.
- NLP is unpredictable.
- NLP may require more keystrokes.
- NLP is unable to adapt to the new domain, and it has a limited function that's why NLP is built for a single and specific task only.

5.2 SYSTEM ARCHITECTURE:-

User Input: User types or speaks a query

Tokenization: Break down input into individual words or tokens

POS Tagging: Identify parts of speech (nouns, verbs, adjectives, etc.) for each token

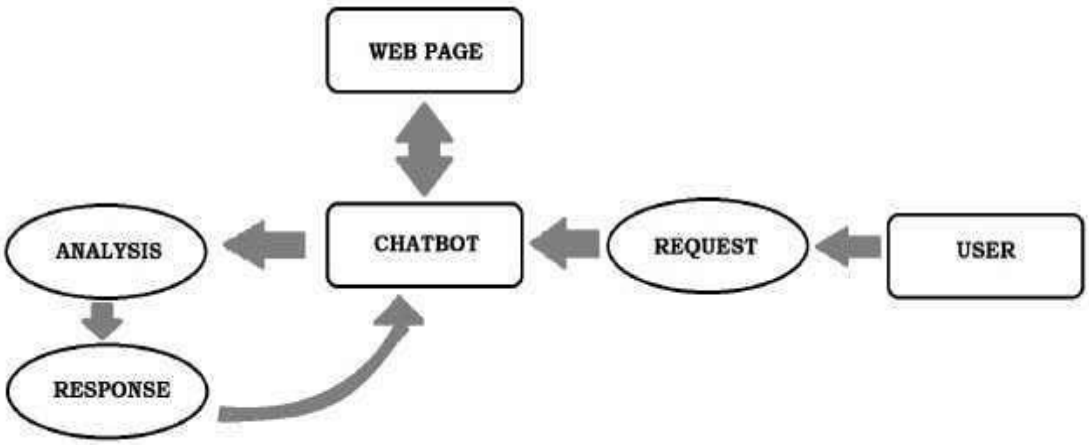
NER: Identify named entities (people, places, organizations, etc.) in the input

Intent Identification: Determine the user's intent behind the query (e.g., booking a flight, asking for directions)

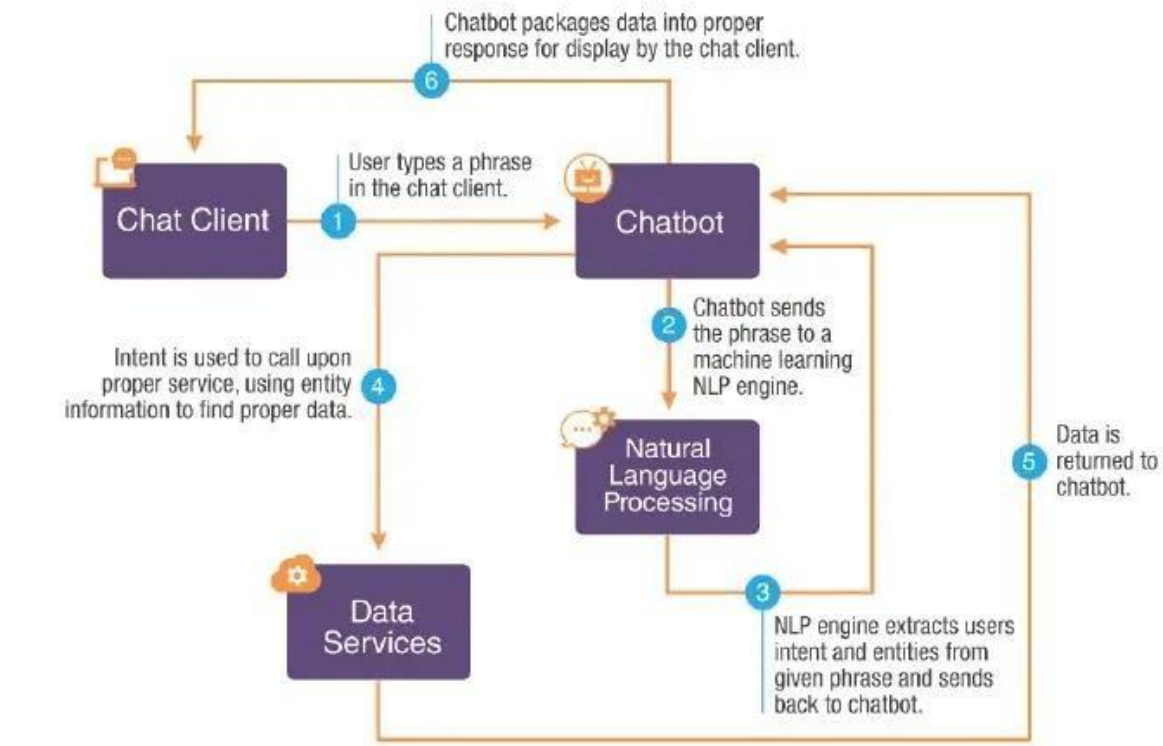
Response Generation: Use deep learning models to generate a response based on the intent and context

Response Output: Provide the response to the user

Feedback Loop: Continuously update and refine the chatbot's understanding and response generation using user feedback



USECASE DIAGRAM



A use case is a methodology used in system analysis to identify, clarify, and organize systemrequirements. In this context, the term "system" refers to something being developed or operated,such as a mail-order product sales and service Web site. Use case diagrams are employedin UML (Unified Modeling Language), a standard notation for the modeling of real-worldobjects and systems.

5.3 TOOLS:-

Hardware Requirements:-

- Modern Opearting System
  - Windows 7 or higher, 64-bit
  - Mac OS X 11 or higher, 64-bit
  - Linux:RHEL 6/7, 64-bit
- 8 GB RAM
- 11 or higher Gen Intel(R)

Software Requirements:-

- Programming Languages: Python for backend development and scripting.
- NLP Libraries: NLTK and spaCy for natural language understanding and processing.
- Machine Learning Frameworks: TensorFlow and PyTorch for training and deploying machine learning models.
- Databases: MySQL for storing user interactions and response data.

6. IMPLEMENTATION AND ANALYSIS

6.1 Python library:-

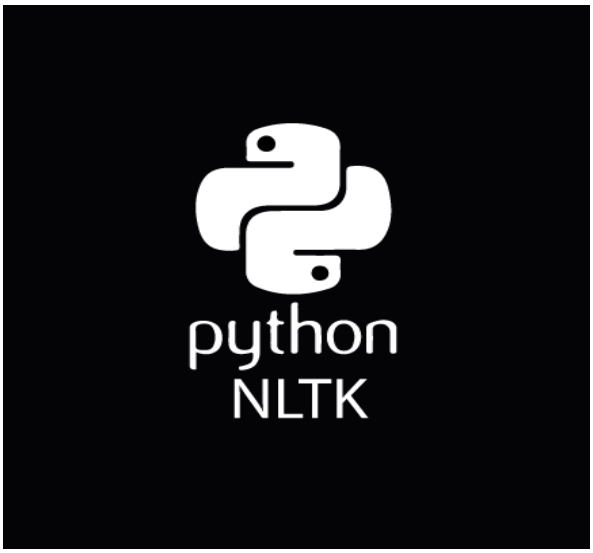
A Python library is a collection of related modules. It contains bundles of code that can be used repeatedly in different programs. It makes Python Programming simpler and convenient for the programmer. As we don't need to write the same code again and again for different programs. Python libraries play a very vital role in fields of Machine Learning, Data Science, Data Visualization, etc.

Python libraries that are used in the project are:

- NLTK
- TensorFlow
- Numpy
- Random
- Pickle

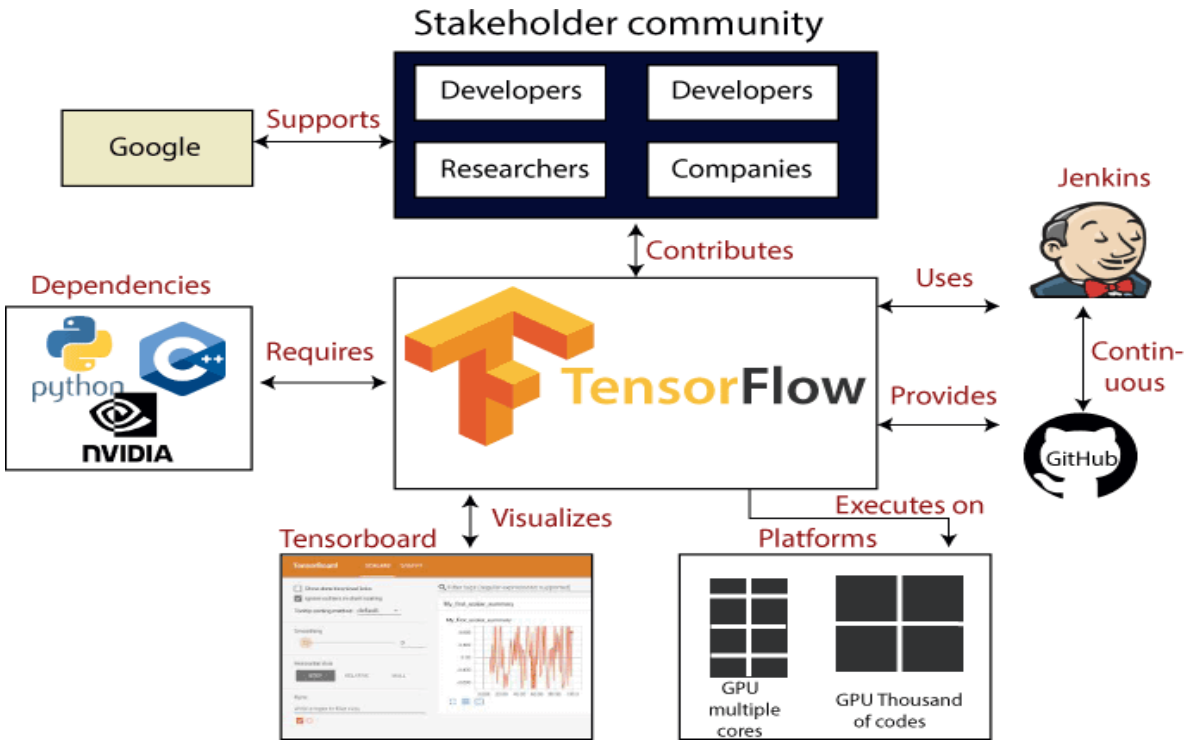
NLTK:-

The Natural Language Toolkit, or more commonly NLTK, is a suite of libraries and programs for symbolic and statistical NLP for English written in the python programming language. It supports classification, tokenization, stemming, tagging, parsing, and semantic reasoning functionalities.



TensorFlow:-

TensorFlow can train and run the deep neural networks for image recognition, handwritten digit classification, recurrent neural network, word embedding, natural language processing, video detection, and many more. TensorFlow is run on multiple CPUs or GPUs and also mobile operating systems.

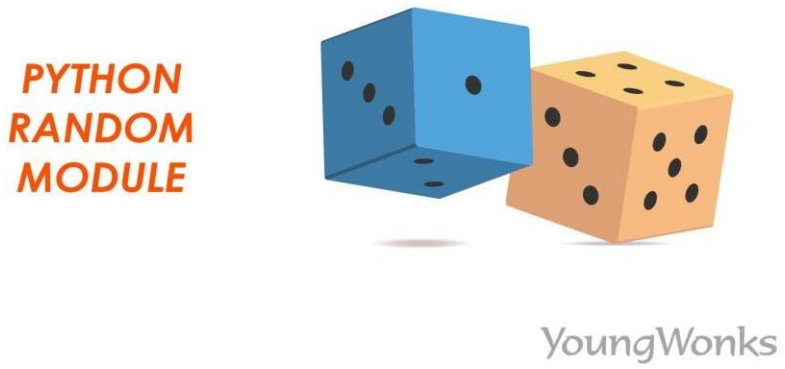


Numpy:-

NumPy is a general-purpose array-processing package.It provides a high-performance multidimensional array object and tools for working with these arrays.It is the fundamental package for scientific computing with Python. It is open-source software.

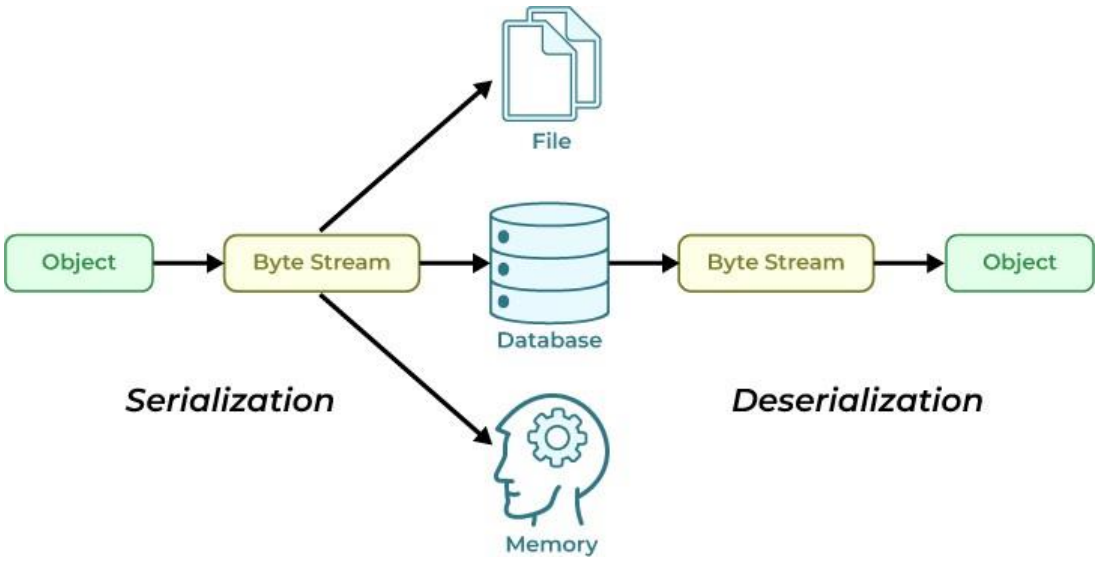
Random:-

Python Random module generates random numbers in Python. These are pseudo-random numbers means they are not truly random. This module can be used to perform random actions such as generating random numbers, printing random a value for a list or string, etc. It is an in-built function in Python.



Pickle:-

Pickle in Python is primarily used in serializing and deserializing a Python object structure. In other words, it's the process of converting a Python object into a byte stream to store it in a file/database, maintain program state across sessions, or transport data over the network.



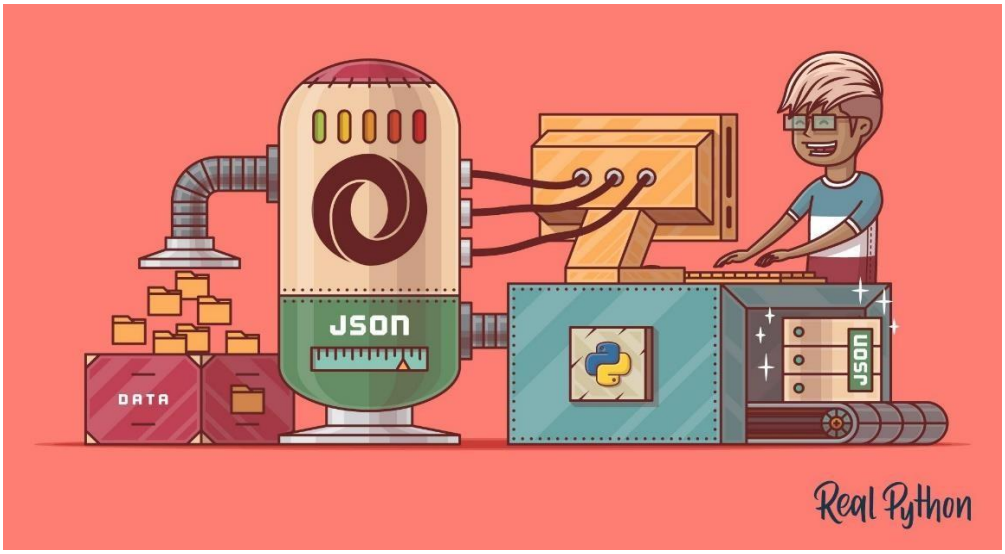
6.2 OTHER LIBRARIES:-

Tkinter:-

Tkinter is a Python library that can be used to construct basic graphical user interface (GUI) applications. In Python, it is the most widely used module for GUI applications.

Json:-

The json library can parse JSON from strings or files. The library parses JSON into a Python dictionary or list. It can also convert Python dictionaries or lists into JSON strings. The JSON Library provides a representation of the JavaScript Object Notation (JSON) with support for parsing, printing, and manipulating JSON values as trees. There is also support for processing JSON data as streams using a "SAX-style" API.



6.3 DATA:-

The crucial element in artificial intelligence tasks is the data. The results will be highlyinfluenced by the data that are given, how are they formatted, their consistency, their re levance to the subject at hand and so on. At this step, many questions should be answeredin order to guarantee that the results will be accurate and relevant. The data that is usedshould be clearly stated, in this case, with proper patterns and responses.

6.4 SOFTWARE DESCRIPTION:-

Python:-

Python is an interpreted, object-oriented, high-level programming language with dynamic semantics. Its high-level built in data structures, combined with dynamic typing and dynamic binding, make it very attractive for Rapid Application Development, as well as for use as a scripting or glue language to connect existing components together. Python's simple, easy to learn syntax emphasizes readability and therefore reduces the cost of program maintenance. Python supports modules and packages, which encourages program modularity and code reuse. The Python interpreter and the extensive standard library are available in source or binary form without charge for all major platforms, and can be freely distributed.

Often, programmers fall in love with Python because of the increased productivity it provides. Since there is no compilation step, the edit-test-debug cycle is incredibly fast. Debugging Python programs is easy: a bug or bad input will never cause a segmentation fault. Instead, when the interpreter discovers an error, it raises an exception. When the program doesn't catch the exception, the interpreter prints a stack trace. A source level debugger allows inspection of local and global variables, evaluation of arbitrary expressions, setting breakpoints, stepping through the code a line at a time, and so on. The debugger is written in Python itself, testifying to Python's introspective power. On the other hand, often the quickest way to debug a program is to add a few print statements to the source: the fast edit-test-debug cycle makes this simple approach very effective.

6.5 Coding:-

Train the machine learning model

Training.py

```
import nltk
from nltk.stem import WordNetLemmatizer
lemmatizer = WordNetLemmatizer()
import json
import pickle
import numpy as np
from tensorflow import keras
from keras.models import Sequential
from keras.layers import Dense, Activation, Dropout
from tensorflow.keras.optimizers.legacy import SGD
import random
words = []
classes = []
documents = []
ignore_words = ['?', '!']
data_file = open('intents.json').read()
intents = json.loads(data_file)
for intent in intents['intents']:
    for pattern in intent['patterns']:
        w = nltk.word_tokenize(pattern)
        words.extend(w)
```

```
        documents.append((w, intent['tag']))
        if intent['tag'] not in classes:
            classes.append(intent['tag'])
words = [lemmatizer.lemmatize(w.lower()) for w in words if w not in ignore_words]
words = sorted(list(set(words)))
classes = sorted(list(set(classes)))
print(len(documents), "documents")
print(len(words), "unique lemmatized words", words)
pickle.dump(words, open('words.pkl', 'wb'))
pickle.dump(classes, open('classes.pkl', 'wb'))
training = []
output_empty = [0] * len(classes)
for doc in documents:
    bag = []
    pattern_words = doc[0]
    pattern_words = [lemmatizer.lemmatize(word.lower()) for word in pattern_words]
    for w in words:
        bag.append(1) if w in pattern_words else bag.append(0)
    output_row = list(output_empty)
    output_row[classes.index(doc[1])] = 1
training.append([bag, output_row])
random.shuffle(training)
# Separate the training data into train_x and train_y
train_x = np.array([element[0] for element in training])
```

```
train_y = np.array([element[1] for element in training])
print("Training data created")
model = Sequential()
model.add(Dense(128, input_shape=(len(train_x[0]),), activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(64, activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(len(train_y[0]), activation='softmax'))
sgd = SGD(learning_rate=0.01, momentum=0.9, nesterov=True)
model.compile(loss='categorical_crossentropy', optimizer=sgd, metrics=['accuracy'])
hist = model.fit(train_x, train_y, epochs=200, batch_size=5, verbose=1)
model.save('chatbot_model.h5', hist)
print("model created")
```

Build the chatbot

Chatbot.py

```
import nltk
from nltk.stem import WordNetLemmatizer
import pickle
import numpy as np
from keras.models import load_model
import json
import random
import tkinter
from tkinter import *
```

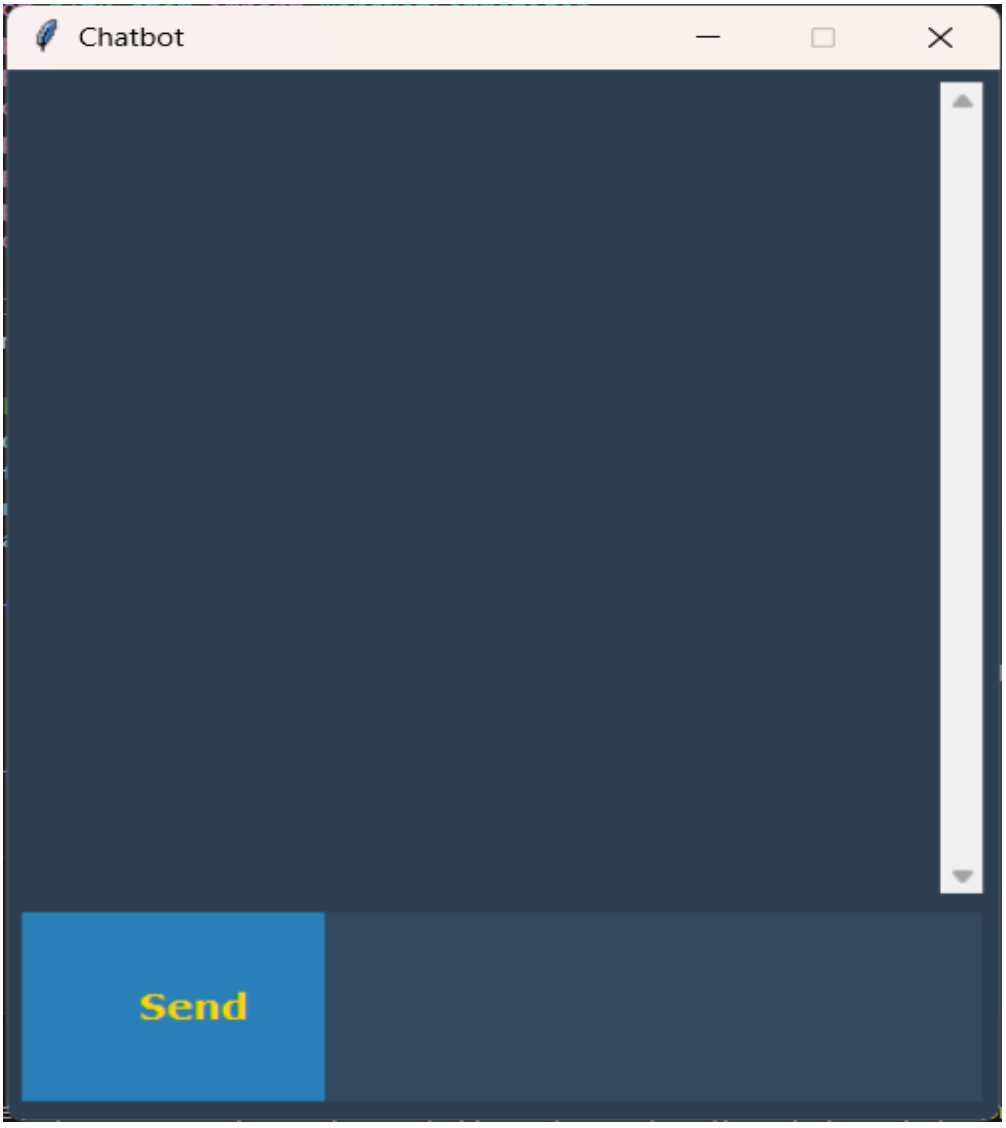
```
# Initialize the lemmatizer
lemmatizer = WordNetLemmatizer()
# Load the model, intents, words, and classes
model = load_model('chatbot_model.h5')
intents = json.loads(open('intents.json').read())
words = pickle.load(open('words.pkl', 'rb'))
classes = pickle.load(open('classes.pkl', 'rb'))
def clean_up_sentence(sentence):
    sentence_words = nltk.word_tokenize(sentence)
    sentence_words = [lemmatizer.lemmatize(word.lower()) for word in sentence_words]
    return sentence_words
def bow(sentence, words, show_details=True):
    sentence_words = clean_up_sentence(sentence)
    bag = [0] * len(words)
    for s in sentence_words:
        for i, w in enumerate(words):
            if w == s:
                bag[i] = 1
                if show_details:
                    print("found in bag: %s" % w)
    return np.array(bag)
def predict_class(sentence, model):
    p = bow(sentence, words, show_details=False)
```

```
    res = model.predict(np.array([p]))[0]
    ERROR_THRESHOLD = 0.25
    results = [[i, r] for i, r in enumerate(res) if r > ERROR_THRESHOLD]
    return_list = []
    for r in results:
        return_list.append({"intent": classes[r[0]], "probability": str(r[1])})
    return return_list
def getResponse(ints, intents_json):
    tag = ints[0]['intent']
    list_of_intents = intents_json['intents']
    for i in list_of_intents:
        if i['tag'] == tag:
            result = random.choice(i['responses'])
            break
    return result
def chatbot_response(msg):
    ints = predict_class(msg, model)
    res = getResponse(ints, intents)
    return res
# GUI setup
def send():
    msg = EntryBox.get("1.0", 'end-1c').strip()
    EntryBox.delete("0.0", END)
```

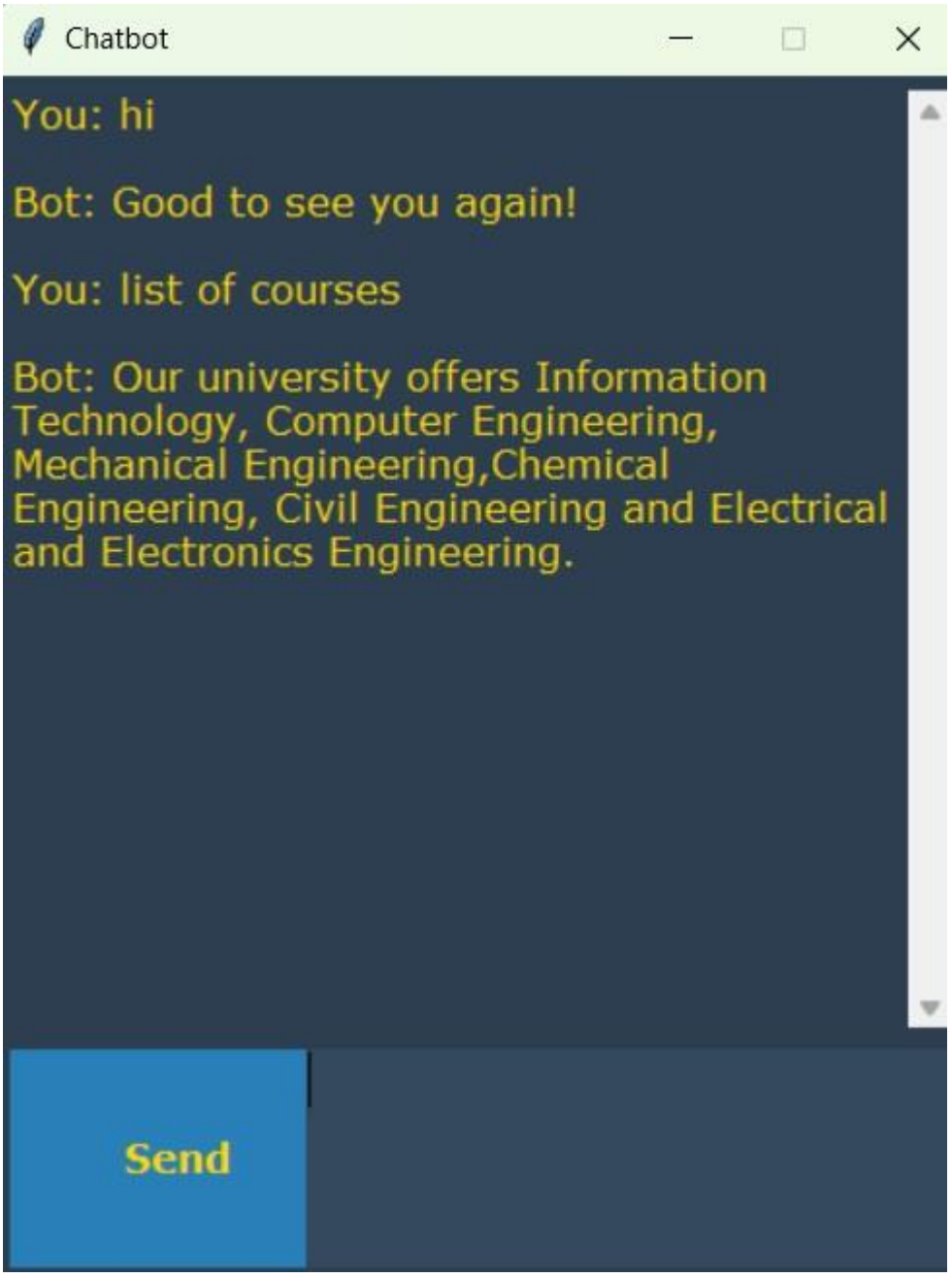
```
    if msg != '':
        ChatLog.config(state=NORMAL)
        ChatLog.insert(END, "You: " + msg + '\n\n')
        ChatLog.config(foreground="#FFD700", font=("Verdana", 12)) # Brighter color for user
messages

        res = chatbot_response(msg)
        ChatLog.insert(END, "Bot: " + res + '\n\n')
        ChatLog.config(state=DISABLED)
        ChatLog.yview(END)
base = Tk()
base.title("Chatbot")
base.geometry("400x500")
base.resizable(width=FALSE, height=FALSE)
# Colors
bg_color = "#2c3e50"
text_color = "#FFD700"  # Brighter yellow color for bot messages
button_color = "#2980b9"
entry_bg_color = "#34495e"
entry_text_color = "#ecf0f1"
button_active_color = "#3498db"
# Create chat window
ChatLog = Text(base, bd=0, bg=bg_color, fg=text_color, height="8", width="50",
font="Arial", wrap=WORD)
ChatLog.config(state=DISABLED)
```

```
# Bind scrollbar to chat window

scrollbar = Scrollbar(base, command=ChatLog.yview, cursor="heart")

ChatLog['yscrollcommand'] = scrollbar.set

# Create button to send message

SendButton = Button(base, font=("Verdana", 12, 'bold'), text="Send", width="12", height=5,
bd=0, bg=button_color, activebackground=button_active_color, fg=text_color,
command=send)
# Create the box to enter message

EntryBox = Text(base, bd=0, bg=entry_bg_color, fg=entry_text_color, width="29",

height="5", font="Arial", wrap=WORD)

# Place all components on the screen

scrollbar.place(x=376, y=6, height=386)

ChatLog.place(x=6, y=6, height=386, width=370)

EntryBox.place(x=128, y=401, height=90, width=265)

SendButton.place(x=6, y=401, height=90)

base.config(bg=bg_color)

scrollbar.config(bg=bg_color, troughcolor=bg_color, activebackground=button_active_color)

base.mainloop()
```

6.6 Output:-



Sample User Request and Response



7. CONCLUSION

The chatbot for simple questions aims to revolutionize customer support by automating the process of answering frequently asked questions. By leveraging cutting-edge NLP techniques and robust backend infrastructure, the chatbot can provide quick, accurate, and contextually relevant responses. This project not only enhances user experience but also allows human support staff to focus on more complex and value-added tasks. The proposed system represents a significant improvement over traditional methods, offering a scalable and efficient solution to handle simple queries.

The proposed chatbot has the potential to revolutionize the way users access information on various topics. By providing accurate and concise responses to simple questions, the chatbot can save users time and effort, making it an ideal tool for individuals seeking quick information on the go. The project's outcome is expected to contribute significantly to the development of more advanced chatbot systems that can provide high-quality responses to user queries.

Through the proposed project and research, we draw a conclusion that the new proposed system will help the college by providing a functional and valuable tool in the field of artificial intelligence. The project demonstrated how the theories and concepts of AI and NLP come together to create a chatbot that can offer high-quality support and conflict resolution to customers. The chatbot has the potential to satisfy the modern customer's demand for personalized experiences and can be easily integrated into an organization to provide 24/7 customer service. With 90% of consumers expecting an online portal for customer service, the need for AI-powered chatbots will only continue to rise, making it essential for businesses to deploy a chatbot solution to stay ahead.