M.KUMARASAMY
COLLEGE OF ENGINEERING
NAAC Accredited Autonomous Institution
Approved by AICTE & Affiliated to Anna University
ISO 9001:2015 & ISO 14001:2015 Certified Institution
Thalavapalayam, Karur – 639 113.

A Minor Project Report

on

CHATBOT

Submitted in partial fulfilment of requirements for the award of the degree

of

BACHELOR OF ENGINEERING

in

COMPUTER SCIENCE AND ENGINEERING

Under the guidance of
Mrs.A.Selvi, M.E,

Associate Professor / CSE

Submitted By

B.DEEPAK (19BCS4019)
S.DHANISH (19BCS4023)
B.GOPINATH (19BCS4032)
B.NAVADEEPAN (19BCS4301)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

M.KUMARASAMY COLLEGE OF ENGINEERING

(Autonomous)

KARUR – 639 113

November 2021

M.KUMARASAMY
COLLEGE OF ENGINEERING
NAAC Accredited Autonomous Institution
Approved by AICTE & Affiliated to Anna University
ISO 9001:2015 & ISO 14001:2015 Certified Institution
Thalavapalayam, Karur – 639 113.

M. KUMARASAMY COLLEGE OF ENGINEERING

(Autonomous Institution affiliated to Anna University, Chennai)

KARUR – 639113

BONAFIDE CERTIFICATE

Certified that this minor project report "**CHATBOT** " is the bonafide work of "**DEEPAK.B (19BCS4019) , DHANISH.S (19BCS4023) , GOPINATH.B (19BCS4032) , NAVADEEPAN.B (19BCS4301)**" who carried out the project work during the academic year 2021-2022 under my supervision.

Signature

Mrs.A.SELVI M.E.,

SUPERVISOR,

Department of Computer Science
and Engineering,
M. Kumarasamy College of Engineering,
Thalavapalayam, Karur -639113

Signature

Dr. S.THILAGAMANI M.E., Ph.D.,

HEAD OF THE DEPARTMENT,

Department of Computer Science
and Engineering,
M. Kumarasamy College of Engineering
Thalavapalayam, Karur -639113.

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

VISION OF THE INSTITUTION

To emerge as a leader among the top institutions in the field of technical education

MISSION OF THE INSTITUTION

- Produce smart technocrats with empirical knowledge who can surmount the global challenges
- Create a diverse, fully-engaged, learner-centric campus environment to provide quality education to the students
- Maintain mutually beneficial partnerships with our alumni, industry, and Professional associations

VISION OF THE DEPARTMENT

To achieve education and research excellence in Computer Science and Engineering

MISSION OF THE DEPARTMENT

- To excel in academic through effective teaching learning techniques
- To promote research in the area of computer science and engineering with the focus on innovation
- To transform students into technically competent professionals with societal and ethical responsibilities

PROGRAM EDUCATIONAL OBJECTIVES (PEOs)

PEO 1: Graduates will have successful career in software industries and R&D divisions through continuous learning.

PEO 2: Graduates will provide effective solutions for real world problems in the key domain of computer science and engineering and engage in lifelong learning.

PEO 3: Graduates will excel in their profession by being ethically and socially responsible.

PROGRAM OUTCOMES (Pos)

Engineering students will be able to:

1. **Engineering knowledge:** Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
2. **Problem analysis:** Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.
3. **Design/development of solutions:** Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.
4. **Conduct investigations of complex problems:** Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.
5. **Modern tool usage:** Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.
6. **The engineer and society:** Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.
7. **Environment and sustainability:** Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.
8. **Ethics:** Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.
9. **Individual and team work:** Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.
10. **Communication:** Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.

11. **Project management and finance:** Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.
12. **Life-long learning:** Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

PROGRAM SPECIFIC OUTCOMES (PSOs)

- **PSO1: Professional Skills:** Ability to apply the knowledge of computing techniques to design and develop computerized solutions for the problems.
- **PSO2: Successful career:** Ability to utilize the computing skills and ethical values in creating a successful career.

ABSTRACT

Chatbots or conversational interfaces as they are also known, present a new way for individuvals to interact with computer systems. Traditionally,to get a questions answered by a software program involved using a search engine ,or filling out a form .A chatbot allows auser to simply askquestion in the same manner then they would address a human.The most well known chatbots are currently are voice chatbots :Alexa and Siri. However, Chatbots are currently being adopted at a high rate on computer platforms.The techonology at the core of the rise of the chatbot is “Natural Language Processing”(NLP).This paper aims to develop a chatbot which resemble as siri and Alexa but it will resolve queries about our department .

ABSTRACT WITH PO AND PSO MAPPING

ABSTRACT	POs MAPPED	PSOs MAPPED
In this project, we are developing a chat bot which will provide result to the user's queries about the students of our department, hence the name " Chatbot", The user will be able to ask various questions to the chat bot about the college and students, the questions may be concerned with the infrastructure, accreditation, Training & Placements, Result etc. and also the chatbot can be used by the college faculties to ask specific information about the students, and the chat bot will be able to provide answer to the user's queries.	**PO 1(3)** **PO 2(3)** **PO 3(3)** **PO 4(2)** **PO 5(3)** **PO 6(3)** **PO 7(2)** **PO 8(3)** **PO 9(3)** **PO 10(3)** **PO 11(2)** **PO 12(1)**	**PSO 1(3)** **PSO 2(3)**

Note: 1- Low, 2-Medium, 3- High

SUPERVISOR **HEAD OF THE DEPARTMENT**

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF ABBREVIATIONS

NLP	Natural Language Processing
GUI	Graphical User Interface
MDI	Mean Driven Interface
CL	Command Line

CHAPTER 1

INTRODUCTION

1.1 INTRODUCTION

CHATBOTS are automated systems which replicate users behavior on one side of the chatting communication. They are mimic systems which imitate the conversations between two individuals. They provide a simulating platform for effective and smart communications with the user on the other end. They copy marketers, sales person, counsellors and other mediators and work to provide services that the above-mentioned people provide. There are wide ranges of chatbots catering in many domains some of them are as follows: business, market, stock, customer care, healthcare, counselling, recommendation systems, support system, entertainment, brokering, journalism, online food and accessory shopping, travel chatbots, banking chatbots, recipe guides, etc.

The most famous chatbots like Alexa or Google assistant are the best examples that can be given for smart communicating chatbots. These are general purpose chatbots that provide services for all domains and are not restricted to a specific domain. There are also domain-specific chatbots which provide functionalities to the above-mentioned domains. Some of them are as follows: Botsify is a chatbot which helps developers to create smart Facebook Messenger Chatbots and is used to collect information from Facebook users. Imperson is a chatbot which helps developers to create business chatbots and provide customer care services. NBC is a chatbot which helps the newsreaders to navigate quickly through top headlines.

Fig 1.1: CHATBOT

1.2 CHATBOTS

A chatbot is an intelligent piece of software that is capable of communicating and performing actions similar to a human. Chatbots are used a lot in customer interaction, marketing on social network sites and instantly messaging the client. There are two basic types of chatbot models based on how they are built; Retrieval based and Generative based models.

User interfaces for software applications can come in a variety of formats, ranging from command-line, graphical, web application, and even voice. While the most popular user interfaces include graphical and web-based applications, occasionally the need arises for an alternative interface. Whether due to multi-threaded complexity, concurrent connectivity, or details surrounding execution of the service, a chat bot-based interface may suit the need.

Chat bots typically provide a text-based user interface, allowing the user to type commands and receive text as well as text to speech response. Chat bots are usually a stateful services, remembering previous commands (and perhaps even conversation) in order to provide functionality. When chat bot technology is integrated with popular web services it can be utilized securely by an even larger audience.

Retrieval based Chatbots

Retrieval-based chatbot uses predefined input patterns and responses. It then uses some type of heuristic approach to select the appropriate response. It is widely used in the industry to make goal-oriented chatbots where we can customize the tone and flow of the chatbot to drive our customers with the best experience.

Generative based Chatbots

Generative models are not based on some predefined responses.

They are based on seq 2 seq neural networks. It is the same idea as machine translation. In machine translation, we translate the source code from one language to another language but here, we are going to transform input into an output. It needs a large amount of data and it is based on Deep Neural networks.

CHAPTER 2

LITERATURE REVIEW

A review on Word Segmentation Segmentation, also referred to as tokenization is the process of splitting text into smaller and meaningful units. These units could be paragraphs, sentences, clauses, phrases, words or letters. The smallest unit are the letters. Word segmentation is the splitting of sentences into individual words separated by blank spaces. The tokenized units of the sentences are called as tokens. The tokenizers split the sentences into words and punctuations marks as independent units. The most commonly used tokenizer is of space type, i.e. it splits the sentences into words at the blank spaces. It is also required that the tokenizer should consider abbreviations, acronyms, dates, numbers in decimal formats, etc., which cannot split at punctuations and blank spaces, as they will lose their meaning if done so.

Mohammed Javed et al. [1] [2015] explained a method to implement word segmentation. He proposed in his algorithm to calculate character spaces in the sentences. The character spaces should include all types of gaps between characters.

They include the gaps between letter, punctuations and the words. The algorithm functions on the basis of the amount of gap or character space between each unit in the sentence. After the calculation of character spaces, an average of the gaps is calculated to know the mean average between characters in the sentence. This average gap distance is then applied to the sentence which is to be segmented.

The places at which the character space is more than the average character space are said to be the points of tokenization. The gap between words is always more than the average gap and hence tokenization takes place at the blank spaces between words in the sentences.

Naeun Lee et al. [2] [2017] proposed the implementation of word segmentation using NLTK. Natural Language ToolKit (NLTK) is a package which caters to provide services for NLP. It has inbuilt tokenizers. Users need to import the package and use the required type of tokenizer which is present in the form of functions.

The NLTK includes a wide range of tokenizers which are as follows standard, letter, word, classic, lowercase, N-gram, pattern, keyword, path, etc. The most commonly used tokenizer is the word-punkt tokenizer which splits the sentences at the blank spaces. The accuracy, speed and efficiency of the NLTK tokenizers is commendable. Also, it does not require any algorithm implementation as the package executes them at the backend.

Tao Jaing [3] [2011] explains the usage of CRF (Conditional Random Fields) Algorithm for word

segmentation. This algorithm trains the system for spaces between the characters. Using this training, the system identifies the gap between characters in the test sentence. The system keeps a threshold value for the gap distance. If the value of gaps in the test sentence is more than the specified threshold, then the sentence splits at those points. CRF requires a lot of training to be given to the system, which makes the process time consuming. Comparing the three methods illustrated above, the NLTK proves to be more efficient in all aspects as compared to the other two. The usage of NLTK does not require the implementation of any algorithm as everything is taken care by the package itself. Also, the accuracy, speed and diversity provided by the package is better than the two algorithm.

A review on POS Tagging

POS Tagging is the process of assigning grammatical annotations to individual words in the sentences. These annotations include the Parts-Of-Speech Tags. They denote the grammatical importance of the word in the sentence based on the dependency of that word with other words in that phrase, clause, sentence, paragraph, etc. The common POS tags are noun, verb, pronoun, etc. There are number of ways which can be used to perform the POS Tagging. Some of them are explained below.

Jerome R. Bellegarda [4] [2010] proposed a method called latent analogy for POS Tagging. In this algorithm, latent semantic mapping (LSM) technique is used. It requires the training on the available corpus. The LSM maintains a feature space of the trained corpus which has been tagged. Now, newsentences are provided to the LSM for tagging and the analysis is performed so as to determine the sentences from the training data which are closest to the test sentence. This is called as sentence neighbourhood. Sentence neighbourhood holds true for two sentences if they share the same intent matter. Once the intent matching sentences are found from the trained data, the POS tags attached to those sentences are then mapped to the test sentences.

CHAPTER 3

EXISTING SYSTEM

3.1 INTRODUCTION

Now days, many software based applications has been designed with a great user interfaces. These interfaces can be in various formats such as command line based interface, web based application, graphical based application, voice based interface. The interfaces that based on Web and graphical are most widely using applications in this human environment. So the need for introduction of new interface has been aroused to act as alternative choice. Finally, an interface that based on chat rooms named chat bot has developed in order to provide connectivity concurrently, the complexity that depends on multi-threading, detailing executions of services around the environment. Google Assistant ,Alexa are the example of the Chatbot Systems.

3.2 DESCRIPTION OF THE SYSTEM

Google Assistant is Google's voice assistant AI for Android devices. It provides a virtual personal assistant experience through a natural language speech interface to perform a variety of tasks.

Examples of how Assistant can be used include the following:

Pull up a boarding pass at an airport to speed up the check-in process ,Retrieve a summary of exercise activity (i.e. miles walked or calories burned), .Receive news updates.Check current traffic conditions and follow navigation instructions via Google Maps,View current weather conditions ,See updates on sporting events.Set reminders and alarms.Find information about restaurants, concerts, movies, or other attractions. Answer questions via Google search.Integrate home automation with Google Home.

As a voice assistant, Google Assistant adds two-way conversation abilities to Google's earlier assistant service, Google Now, which is a web and text-based service. Assistant uses cognitive computing, machine learning and voice recognition technology.

Assistant's development began in 2016 and was intended for use with Google's Allo messaging app and Google Home smart speaker. Aside from typical tablets, smartphones and notebooks, Assistant was also built into Android Wear 2.0 for wearable technology, with Android TV and Android Auto integration to follow. According to Google CEO Sundar Pichai, Assistant was designed to be a conversational and interactive experience, and "an ambient experience that extends across devices."Other digital assistants on the market include Apple's Siri, Amazon's Alexa, Google Now and Microsoft's Cortana.

The features include:

- letting third-party device makers incorporate their own "Actions on Google" commands for their respective products
- incorporating text-based interactions and more languages
- allowing users to set a precise geographic location for the device to enable improved location-specific queries.

On May 2, 2018, Google announced a new program on their blog that focuses on investing in the future of the Google Assistant through early-stage startups. Their focus was to build an environment where developers could build richer experiences for their users. This includes startups that broaden Assistant's features, are building new hardware devices, or simply differentiating in different industries.[45]

CHAPTER 4

PROBLEM DESCRIPTION

Changing of requirements As an industrial project to build a product, we must follow the requirement from the user. However, because the project's goal is to be used by the business team, but it is responsible by the technical team, the requirement changed a lot in the middle after a meeting with the business team. The business team want a simple bot that can give recommendation immediately. We had to archive what we had done before and build a new one.

- Lacking of training data.

The quantity and quality of training data is critical to the performance to a machine learning model. However, because some confidential and privacy reasons, the business team cannot provide enough data for us,and we had to make up data by our own. For the machine learning model, we generate some fake data based on our daily life experience, which is really biased, although with a good accuracy on the fake data.

- Unstable API version.

Because API service we are using are still under development, and we cannot fix to a version for the API, the API may changes overtime. Moreover, there are inconsistencies between the APIs and their documents or sample codes.

- Not familiar with the PHP language and .NET framework.

None of the three of us has previous knowledge with PHP language. Programming in a new language in such a huge framework is quite challenging for us at the beginning ofthe project. However,when we comes to the later phases, we are more used to that.

CHAPTER 5

PROPOSED SYSTEM

5.1 INTRODUCTION

The chat bot system introduces the areas where the people will interact and get solutions. Having the environment that one paves a path to chat bot system users. That path may be based on querying and answering. Introduced chat bot system will provide an interface that depends on text. This allows the users to type the commands and receive the text. It also involves receiving end may be having response as text to the speech.It uses the state ful based services which could remember the commands that previously asked that may be conversation.This system is developed in order to enhance the functionality.It allows the different set of people to be organizing in a specified space in order to make their discussion. The technology named chat bot can be highly utilized by large set of people in their working environment. This mechanism has developed by using Artificial based algorithms that allows the people to input their queries. Queries can be taken by chat bot and it has been analysed and understudied. There is no specific format for inputting the queries. The answers for every query has been provided since the system had used advance technology namely Artificial Intelligence. Answers would be appropriate, in case if user found that the provided answer as invalid they can make select over invalid answer. This way allows the admin to view that invalid one and to delete it. It can be deleting by admin and the admin would add relevant answer to that particular question. Finally it alsohelpful for student in case of their inquiry, update of their information, activities or events of college etc…

.

Fig 5.1: PROPOSED SYSTEM

5.2 SYSTEM ARCHITECTURE

There are seven steps to design the Chatbot process they are scope and requirement, identifying the inputs, understanding the UI elements, craft first interaction, build conversation and finally testing. The Chatbot design process figure is shown in the below

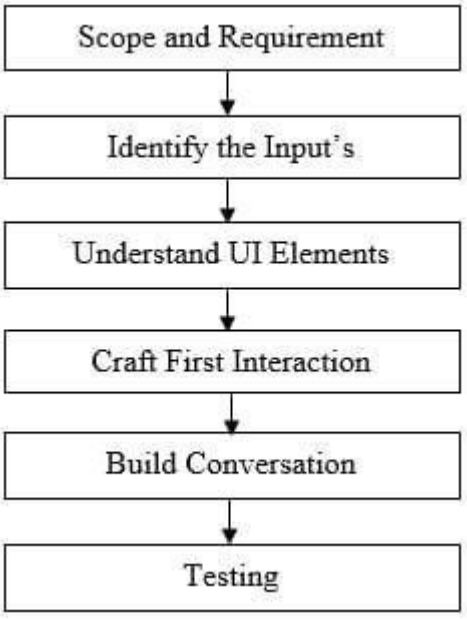

Fig 5.2: CHATBOT DESIGN PROCESS

The first step to designing the Chatbot is to know the scope and requirements like why chatbot, platform to launch chatbots and its limitations. The second step is to identify the inputs from users in the form of queries through text, voice or images, from devices, and intelligence systems. The third step is to understand the User Interface (UI) elements, that we can see in our applications. UI elements are of five types they are: Command Line(CL), Graphical User Interface(GUI), Menu-Driven Interface (MDI), Form-Based Interface (FBI) and Natural Language Interface (NLI). After understanding user interface elements, the next step is to craft the first interaction and build a conversation. The final step of the Chatbot design process is testing, which is done on mobile and websites to know how it's working.

CHAPTER 6

MODULES DESCRIPTION

6.1 INTENT

An intent in the above figure is defined as a user's intention, example the intent of the word "Good Bye" is to end the conversation similarly, the intent of the word "What are some good Chinese restaurants" the intent would be to find a restaurant.

6.2 ENTITY

An entity in the Chatbot is used to modifies an intent and there are three types of entities they are system entity, developer entity and session entity.

6.3 CANDIDATE RESPONSE GENERATOR

The candidate response generator in the Chatbot do the calculations using different algorithms to process the user request. Then the result of these calculations is the candidate's response.

6.4 RESPONSE SELECTOR

The response selector in the Chatbot used to select the word or text according to the user queries to give a response to the users which should work better.

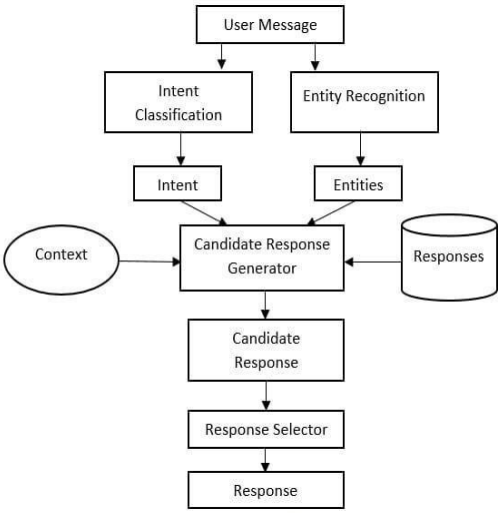

Fig 6.1 :MODULES DESCRIPTION

CHAPTER 7

SYSTEM SPECIFICATION

7.1 Hardware Requirement

- Quad core Intel Core i7 Skylake or higher (Dual core is not the best for this kind of work, but manageable)
- 32GB of RAM (16GB is okay but not for the performance you may want and or expect)
- M.2 PCIe or regular PCIe SSD with at least 256GB of storage, though 512GB is best for performance.
- The faster you can load and save your applications, the better the system will perform. (SATA III will get in the way of the system's performace)
- Broadband internet access and it should covers a wide range of technologies.
- In the admin module we have four main actions login, manage ,verify by calls. In the login the admin can login to his/her account. He can manage the CHATBOT using manage chats page.
- With the help of the verify by calls the admin verifies the orders by phone calls. And they collect the garbage with the help of employees.

7.2 Software Requirement

- **Programming language :** Python(Machine Learning)
- **Kit required to develop chatbot in :** No kit.
- **Technologies working in chatbot in :** Natural Language Processing (NLP).

CHAPTER 8

RESULT AND DISCUSSION

8.1 SCREEN SHOTS

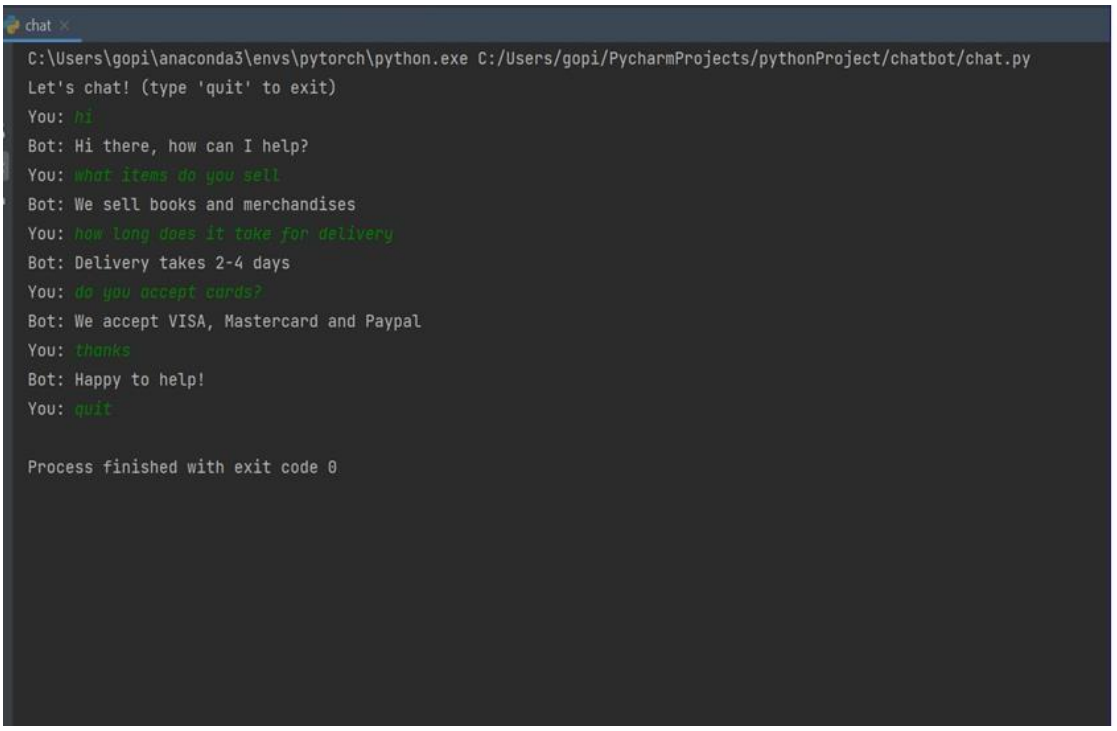

FIG 8.1: CHATBOT ANSWERING QUERIES OF USERS

This project we made a Department specific chatbot system that can be custom fitted to education domain chatbot, the addition of this chatbot system in the college website will make the webpage more user interactive as it responds to the user queries very accurately as it is a domain specific chatbot system, and furthermore we had investigated our college chatbot system design stages and a few different techniques by which the precision of the chatbot system can be made much better. To make the responses given by the chatbot system more meaningful and accurate the administrator has to train the chatbot system with more information regarding to college and increase the scope of knowledge base. Nevertheless, gathering feedback from the potential user can be helpful in developing the college Chatbot system, ultimately servicing the user queries.

CHAPTER 9

CONCLUSION AND FUTURE ENHANCEMENT

Chatbots are the new Apps! As we have discussed in the above deliverables, this project brings the power of chatbots to Yioop and enriches its usability. Chatbots in Yioop can give a human like touch to some aspects and make it an enjoying conversation. And they are focused entirely on providing information and completing tasks for the humans they interact with. The above mentioned functionality in all the deliverables is implemented and pushed into Yioop code. By implementing the above mentioned deliverables I was able to add a basic chatbot functionality in to the Yioop. I.e., configuring and creating accounts for bot users with bot settings which is mentioned in deliverable 2, activating a bot whenever a user asks for it via post in a thread which is discussed in deliverable 3 and as I discussed in deliverable4, I have implemented a simple weather chatbot that gives weather information whenever a user ask and Fig. 3 tells that I was also able to converse with the bot in Yioop.

I intend to enhance the system developed so far in CS298. Next step towards building chatbots involvehelping people to facilitate their work and interact with computers using natural language or using set of rules. Future Yioop chatbots, backed by machine-learning technology, will be able to remember past conversations and learn from them to answer new ones. The challenge would be conversing with multiple bot users and multiple user

There are limitations to what has been currently achieved with chatbots. The limitations ofdata processing and retrieval are hindering chatbots to reach their full potential. It is not thatwe lack the computational processing power to do so. However, there is a limitation on“How” we do it. One of the biggest examples is the retail customer market.

APPENDIX

SOURCE CODE:

INTENTS(JSON):

```
{
 "intents": [
   {
    "tag": "greeting",
    "patterns": [
      "Hi",
      "Hey",
      "How are you",
      "Is anyone there?",
      "Hello",
      "Good day"
    ],
    "responses": [
      "Hey :-)",
      "Hello, thanks for visiting",
      "Hi there, what can I do for you?",
      "Hi there, how can I help?"
    ]
   },
   {
    "tag": "goodbye",
    "patterns": ["Bye", "See you later", "Goodbye"],
    "responses": [
      "See you later, thanks for visiting",
      "Have a nice day",
      "Bye! Come back again soon."
    ]
   },
   {
    "tag": "thanks",
    "patterns": ["Thanks", "Thank you", "That's helpful", "Thank's a lot!"],
    "responses": ["Happy to help!", "Any time!", "My pleasure"]
   },
   {
    "tag": "items",
    "patterns": [
      "Which items do you have?",
      "What kinds of items are there?",
      "What do you sell?"
    ],
    "responses": [
      "We sell books and merchandises",
      "We have books and merchandises"
    ]
   },
```

```
  {
    "tag": "payments",
    "patterns": [
      "Do you take credit cards?",
      "Do you accept Mastercard?",
      "Can I pay with Paypal?",
      "Are you cash only?"
    ],
    "responses": [
      "We accept VISA, Mastercard and Paypal",
      "We accept most major credit cards, and Paypal"
    ]
  },
  {
    "tag": "delivery",
    "patterns": [
      "How long does delivery take?",
      "How long does shipping take?",
      "When do I get my delivery?"

    ],
    "responses": [
      "Delivery takes 2-4 days",
      "Shipping takes 2-4 days"
    ]
  },
  {
    "tag": "funny",
    "patterns": [
      "Tell me a joke!",
      "Tell me something funny!",
      "Do you know a joke?"
    ],
    "responses": [
      "Why did the hipster burn his mouth? He drank the coffee before it was cool.",
      "What did the buffalo say when his son left for college? Bison."
    ]
  },
  {
    "tag": "college",
    "patterns": [
    "recommend college",
    "suggest college",
      "am i eligible for this college?"
    ],
    "responses":[
      "mkce",
      "u r eligible for mkce"
    ]
  }
]}
```

NLTK:

```
import numpy as np
import nltk
#nltk.download('punkt')
from nltk.stem.porter import PorterStemmer
stemmer = PorterStemmer()

def tokenize(sentence):
    """
    split sentence into array of words/tokens
    a token can be a word or punctuation character, or number
    """
    return nltk.word_tokenize(sentence)


def stem(word):
    """
    stemming = find the root form of the word
    examples:
    words = ["organize", "organizes", "organizing"]
    words = [stem(w) for w in words]
    -> ["organ", "organ", "organ"]
    """
    return stemmer.stem(word.lower())


def bag_of_words(tokenized_sentence, words):
    """
    return bag of words array:
    1 for each known word that exists in the sentence, 0 otherwise
    example:
    sentence = ["hello", "how", "are", "you"]
    words = ["hi", "hello", "I", "you", "bye", "thank", "cool"]
    bog   = [  0 ,    1 ,    0 ,   1 ,    0 ,    0 ,      0]
    """
    # stem each word
    sentence_words = [stem(word) for word in tokenized_sentence]
    # initialize bag with 0 for each word
    bag = np.zeros(len(words), dtype=np.float32)
    for idx, w in enumerate(words):
        if w in sentence_words:
            bag[idx] = 1

    return bag
```

TRAINING:

```
import numpy as np
import random
import json

import torch
```

```
import torch.nn as nn
from torch.utils.data import Dataset, DataLoader

from nltk_utils import bag_of_words, tokenize, stem
from model import NeuralNet

with open('intents.json', 'r') as f:
    intents = json.load(f)

all_words = []
tags = []
xy = []
# loop through each sentence in our intents patterns
for intent in intents['intents']:
    tag = intent['tag']
    # add to tag list
    tags.append(tag)
    for pattern in intent['patterns']:
        # tokenize each word in the sentence
        w = tokenize(pattern)
        # add to our words list
        all_words.extend(w)
        # add to xy pair
        xy.append((w, tag))

# stem and lower each word
ignore_words = ['?', '.', '!']
all_words = [stem(w) for w in all_words if w not in ignore_words]
# remove duplicates and sort
all_words = sorted(set(all_words))
tags = sorted(set(tags))

print(len(xy), "patterns")
print(len(tags), "tags:", tags)
print(len(all_words), "unique stemmed words:", all_words)

# create training data
X_train = []
y_train = []
for (pattern_sentence, tag) in xy:
    # X: bag of words for each pattern_sentence
    bag = bag_of_words(pattern_sentence, all_words)
    X_train.append(bag)
    # y: PyTorch CrossEntropyLoss needs only class labels, not one-hot
    label = tags.index(tag)
    y_train.append(label)

X_train = np.array(X_train)
y_train = np.array(y_train)

# Hyper-parameters
```

```
num_epochs = 1000
batch_size = 8
learning_rate = 0.001
input_size = len(X_train[0])
hidden_size = 8
output_size = len(tags)
print(input_size, output_size)


class ChatDataset(Dataset):

    def __init__(self):
        self.n_samples = len(X_train)
        self.x_data = X_train
        self.y_data = y_train

    # support indexing such that dataset[i] can be used to get i-th sample
    def __getitem__(self, index):
        return self.x_data[index], self.y_data[index]

    # we can call len(dataset) to return the size
    def __len__(self):
        return self.n_samples


dataset = ChatDataset()
train_loader = DataLoader(dataset=dataset,
                          batch_size=batch_size,
                          shuffle=True,
                          num_workers=0)

device = torch.device('cuda' if torch.cuda.is_available() else 'cpu')

model = NeuralNet(input_size, hidden_size, output_size).to(device)

# Loss and optimizer
criterion = nn.CrossEntropyLoss()
optimizer = torch.optim.Adam(model.parameters(), lr=learning_rate)

# Train the model
for epoch in range(num_epochs):
    for (words, labels) in train_loader:
        words = words.to(device)
        labels = labels.to(dtype=torch.long).to(device)

        # Forward pass
        outputs = model(words)
        # if y would be one-hot, we must apply
        # labels = torch.max(labels, 1)[1]
        loss = criterion(outputs, labels)
```

```
        # Backward and optimize
        optimizer.zero_grad()
        loss.backward()
        optimizer.step()

    if (epoch + 1) % 100 == 0:
        print(f'Epoch [{epoch + 1}/{num_epochs}], Loss: {loss.item():.4f}')

print(f'final loss: {loss.item():.4f}')

data = {
    "model_state": model.state_dict(),
    "input_size": input_size,
    "hidden_size": hidden_size,
    "output_size": output_size,
    "all_words": all_words,
    "tags": tags
}

FILE = "data.pth"
torch.save(data, FILE)

print(f'training complete. file saved to {FILE}')
```

MODEL:

```
 import torch
import torch.nn as nn

class NeuralNet(nn.Module):
    def init (self, input_size, hidden_size, num_classes):
        super(NeuralNet, self). init_()
        self.l1 = nn.Linear(input_size, hidden_size)
        self.l2 = nn.Linear(hidden_size, hidden_size)
        self.l3 = nn.Linear(hidden_size, num_classes)
        self.relu = nn.ReLU()

    def forward(self, x):
        out = self.l1(x)
        out = self.relu(out)
        out = self.l2(out)
        out = self.relu(out)
        out = self.l3(out)
        # no activation and no softmax at the end
        return out
```

CHAT:

```
import random
import json
import torch

from model import NeuralNet
```

```
from nltk_utils import bag_of_words, tokenize

device = torch.device('cuda' if torch.cuda.is_available() else 'cpu')

with open('intents.json', 'r') as json_data:
    intents = json.load(json_data)

FILE = "data.pth"
data = torch.load(FILE)

input_size = data["input_size"]
hidden_size = data["hidden_size"]
output_size = data["output_size"]
all_words = data['all_words']
tags = data['tags']
model_state = data["model_state"]

model = NeuralNet(input_size, hidden_size, output_size).to(device)
model.load_state_dict(model_state)
model.eval()

bot_name = "BOT"
print("Let's chat! (type 'quit' to exit)")
while True:
    # sentence = "do you use credit cards?"
    sentence = input("You: ")
    if sentence == "quit":
        break

    sentence = tokenize(sentence)
    X = bag_of_words(sentence, all_words)
    X = X.reshape(1, X.shape[0])
    X = torch.from_numpy(X).to(device)

    output = model(X)
    _, predicted = torch.max(output, dim=1)

    tag = tags[predicted.item()]

    probs = torch.softmax(output, dim=1)
    prob = probs[0][predicted.item()]
    if prob.item() > 0.75:
        for intent in intents['intents']:
            if tag == intent["tag"]:
                print(f"{bot_name}: {random.choice(intent['responses'])}")
    else:
        print(f"{bot_name}: I do not understand...")
```

REFERENCES

[1] Introduction to Artificial Intelligence Markup Language, Available at https://www.tutorialspoint.com/aiml/aiml_introduction.html

[2] Prof.K.Bala, Mukesh Kumar, SayaliHulawale, SahilPandita,"Chat-Bot For College Management System Using A.I" International Research Journal of Engineering and Technology (IRJET) Volume: 04, Issue: 11, Page no: 2030-2033| Nov 2017.

[3] Porter Stemmer Algorithm, Available at http://snowball.tartarus.org/algorithms/porter/stemmer.html

[4] Guruswami Hiremath, AishwaryaHajare, PriyankaBhosale, RasikaNanaware, Dr. K. S. Wagh, "Chatbot for education system" International Journal of Advance Research, Ideas and Innovations in Technology (IJARIIT) ISSN: 2454-132X, Volume: 4, Issue: 3, Page no: 37-43|2018.

[5] Amey Tiwari, Rahul Talekar, Prof.S.M.Patil, "College Information Chat Bot System" International Journal of Engineering Research and General Science (IJERGS) Volume: 5, Issue: 2, Page no: 131-137| March-April2017.

[6] K. Jwala, G.N.V.G Sirisha, G.V. Padma Raju, "Developing a Chatbot using Machine Learning" International Journal of Recent Technology and Engineering (IJRTE) ISSN: 2277-3878, Volume: 8 Issue: 1S3, Page no: 89-92| June 2019.

[7] Basics of Natural Language ToolKit, Avalilable at https://www.nltk.org/

[8]] Naeun Lee, Kirak Kim, Taeseon Yoon, "Implementation of Robot Journalism by Programming Custombot using Tokenization and Custom Tagging" International Conference on Advanced Communications Technology (ICACT) Page no: 566-570| Feb 2017.

[9] Fundamentals of Natural Language Processing - Tokenization, Lemmatization, Stemming and Sentence Segmentation, Available at https://colab.research.google.com/github/dair-ai/notebooks/blob/master/_notebooks/2020-03-19-nlp_basics_tokenization_segmentation.ipynb#scrollTo=H7gQFbUxOQtb

[10] Jazzy spell checker Library, Available at http://jazzy.