Minor Project Report

On

REAL TIME SUPPORT SYSTEM

In partial fulfilments requirements of the degree

Of

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE & ENGINEERING

Submitted by:

AMIT JAIN (18100BTCSE02661)

PRAKHAR SAKI (18100BTCSE02713)

PRAVEEN SHARMA(18100BTCSE02716)

Under guidance of

Prof. Rupali Bhartiya

SHRI VAISHNAV VIDYAPEETH VISHWAVIDYALAYA, INDORE

SHRI VAISHNAV INSTITUTE OF INFORMATION TECHNOLOGY

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Jan-June 2021

SHRI VAISHNAV VIDHYAPEETH VISHWAVIDHYALA, INDORE

SHRI VAISHNAV INSTITUTE OF INFORMATION TECHNOLOGY

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

DECLARATION

We here declare that work which is being presented in the project entitled "**REAL TIME SUPPORT SYSTEM**" in partial fulfilment of degree of **Bachelor of Technology in Computer Science & Engineering** is an authentic record of our work carried out under the supervision and guidance Mrs. Rupali Bhartiya Asst. professor of Computer Science & Engineering. The matter embodied in this project has not been submitted for award of any other degree.

Date:

Amit Jain

Prakhar Saki

Praveen Sharma

SHRI VAISHNAV VIDYAPEETH VISHWAVIDYALAYA, INDORE

SHRI VAISHNAV INSTITUTE OF INFORMATION TECHNOLOGY

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

PROJECT APPROVAL SHEET

Following team has done the appropriate work related to the "**Real Time Support System**" in partial fulfilment for the award of **Bachelor of Technology in Computer Science & Engineering** of "SHRI VAISHNAV INSTITUTE OF INFORMATION TECHNOLOGY" and is being submitted to SHRI VAISHNAV VIDYAPEETH VISHWAVIDYALAYA, INDORE.

Team

1. **Amit Jain (18100BTCSE02661)**
2. **Prakhar Saki (18100BTCSE02713)**
3. **Praveen Sharma (18100BTCSE02716)**

Internal Examiner **External Examiner**

Date:

SHRI VAISHNAV VIDYAPEETH VISHWAVIDYALAYA, INDORE

SHRI VAISHNAV INSTITUTE OF INFORMATION TECHNOLOGY

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

CERTIFICATE

This is to clarify that **Mr Amit Jain, Mr. Prakhar Saki** and **Mr. Praveen Sharma** working in a team have satisfactorily completed the project entitled **"REAL TIME SUPPORT SYSTEM"** under the guidance of Mrs. Rupali Bhartiya in partial of the fulfilment of the degree of **Bachelor of Technology in Computer Science & Engineering** awarded by SHRI VAISHNAV INSTITUTE OF INFORMATION TEFCHNOLOGY affiliated to SHRI VAISHNAV VIDHYAPEETH VISHWAVIDHYALAY, INDORE during the academic year **Jan 2021- Jun 2021.**

Mrs. Rupali Bhartiya
Project Guide

Dr. Anand Rajawat
Director & Head of
Department of Computer
Science & Engineering

ACKNOWLEDGEMENT

We are grateful to a number of persons for their advice and support during the time of complete our project work. First and foremost our thanks goes to **Dr. Anand Rajawat** Head of the Department of Computer Science & Engineering and **Ms. Rupali Bhartiya** the mentor of our project for providing us valuable support and necessary help whenever required and also helping us explore new technologies by the help of their technical expertise. His direction, supervision and constructive criticism were indeed the source of inspiration for us.

We would also like to express our sincere gratitude towards our Director **Dr. Anand Rajawat** for providing us valuable support.

We are really indebted to **Prof. Rupali Bhartiya,** project coordinator for helping us in each aspect of our academics activities. We also owe our sincere thanks to all the **faculty members** of Computer Science & Engineering Department who have always been helpful.

We forward our sincere thanks to all **teaching and non-teaching staff** of Computer Science & Engineering department, SVVV Indore for providing necessary information and there kind co-operation.

Amit Jain (18100BTCSE02661)

Prakhar Saki (18100BTCSE02713)

Praveen Sharma (18100BTCSE02716)

Index

Content **Page no.**

1. INTRODUCTION

1.1 Problem Statement

As today all the business are going live so there is high requirement of people who can interact with people and help them with their problem but due to high salary of employees and low profit of a company they can't afford many people as costumer services executive.

Also there is not possible for any human to give his hundred percentage at all the time so if a client comes and there is no one available then it makes a bad impact on clients and will also affect in degrading the profit percentage of company.

1.2 Objective

The objective of "Real Time Response System" is to provide 24/7* support to users that visit your website at any time. As we are heading toward global market and the first thing that comes in our mind while starting a business is customer satisfaction. That is if the user is facing any problem or issue then we should be able to solve that issue as soon as possible. But it is not possible for a small business person to keep a team of employee to handle error and issue all the time so here is this situation the business man can apply our "Real Time Support System" in his website so that if any person visit at time he will get the solution for his query .Also it is Graphical User Interface so the user will easily understand it and will use as he wants.

After the visitor is gone from your website then the chat will be saved in your system so that owner will know what was the query of that user and if the response was not correct then the owner can easily edit it in database.

1.3 Scope

The scope of this project is to provide information to the users in so far as possible, making then easier to use our product. Also it is a prototype which is only a desktop based application and it will give you all the information regarding python modules and libraries but in future whenever the owner of any website will implement this in their website they can give the access database of Frequently used questions in so that whenever any user ask any questions it will be answered from predefined database.

1.4 Platform Specification

1.4.1 Hardware:

Processor	64 Bits
Hard Disk Space	20GB
Ram	4GB

1.4.2 Software

Operating System	Windows 7 and above
Development:	
Technology	Python 3.7
IDE	PyCharm2020.2.2
Front End	Tkinter

1.4.3 Implementing language

Front End:

We have used python platform which include in tkinter.

Python:

Our project is completely coded in PYTHON language. It is one of the most powerful programming. Python is the fully object oriented language developed by Guido van Rossum. In Object-Oriented Programming(OOP), a computer program is considered to be a group of objects that interact with each other.

Graphical User Interface(GUI):

Brevity

Python programs using Tkinter can be very brief, partly because of the power of Python, but also due to Tk. In particular, reasonable default values are defined for many options used in creating a widget, and packing it (i.e., placing and displaying).

Cross platform

Tk provides widgets on Windows, Macs, and most Unix implementations with very little platform-specific dependence. Some newer GUI frameworks are achieving a degree of platform independence, but it will be some time before they match Tk's in this respect.

Maturity

First released in 1990, the core is well developed and stable.

Extensibility

Many extensions of Tk exist, and more are being frequently distributed on the Web. Any extension is immediately accessible from Tkinter, if not through an extension to Tkinter, than at least through Tkinter's access to the Tcl language.

Speed

There is some concern with the speed of Tkinter. Most calls to Tkinter are formatted as a Tcl command (a string) and interpreted by Tcl from where the actual Tk calls are made. This theoretical slowdown caused by the successive execution of two interpreted languages is rarely seen in practice and most real-world ..

2. SYSTEM ANALYSIS

2.1 Identification of Need

At present in the existing system, No one is available of 24/7 rather a normal human can't handle a user all the time. If we are hiring a person for that job but it is too costly. one most common issue is that if a normal person come difficulties to understand the website, while person contact using call. Sometime employee can't pickup the call or it is busy that time. Another issue that in existing system we can't store data while communication between user and employee. After a user ask that when my query to be resolve. employee said that I forgot your query So it is very difficult to handle such type issue .

2.2 Preliminary Investigation

1. Firstly we gathered data from online surveys that were conducted by big organizations to know the reasons for the failures of small business on online platform
2. From there we get to know that lack of interaction between user and owner is one of the main cause for this and mostly online users don't like to interact directly to person in charge

3.FEASIBILITY STUDY

Feasibility Study is a study to evaluate feasibility of proposed project or system. As name suggests feasibility study is the feasibility analysis or it is a measure of the software product in terms of how much beneficial product development will be for the organization in a practical point of view. Feasibility study is carried out based on many purposes to analyze whether software product will be right in terms of development, implantation, contribution of project to the organization etc.

3.1 Technical Feasibility :

In Technical Feasibility current resources both hardware software along with required technology are analyzed/assessed to develop project. This technical feasibility study gives report whether there exists correct required resources and technologies which will be used for project development. Along with this, feasibility study also analyzes technical skills and capabilities of technical team, existing technology can be used or not, maintenance and up-gradation is easy or not for chosen technology etc.

3.2 Economic Feasibility :

In Economic Feasibility study cost and benefit of the project is analyzed. Means under this feasibility study a detail analysis is carried out what will be cost of the project for development which includes all required cost for final development like hardware and software resource required, design and development cost and operational cost and so on.

3.3 Behavioural Feasibility:

It evaluates and estimates the user attitude or behaviour towards the development of new system. It helps in determining if the system requires special effort to educate, retrain, transfer, and changes in employee's job status on new ways of conducting business.

4. LITERATURE SURVEY

I believe it is feasible, and at Quick Responses I have several early users who put their money in it. Survey chatbots, if done right, address two of the fundamental problems of the Market Research industry today:
1: The younger crowd (millennia's, you name it) don’t email - they text. Hence, they are out of reach for traditional sampling, which mostly relies on email-based consumer panels. If you engage with respondents on messengers, you can at least get their attention - and then it’s up to you to use it right.
2: The sets chatbots apart is their capability to catch the users’ attention. Most subjects tend to lose interest in regular surveys owing to the long registration process. However, when a chatbot is involved, the conversation picks off from the beginning with better engagement and concrete results. Rather than an elaborate question and answer session, the chatbot engages in a conversation to conduct the survey Implementing language

5. TECHNICAL PART

5.1 Modules Used:

1.time:- The Python time module provides many ways of representing time in code, such as objects, numbers, and strings. It also provides functionality other than representing time, like waiting during code execution and measuring the efficiency of your code.

a. time.strftime(format[, t])-Convert a tuple or struct_time representing a time as returned by gmtime() or local time() to a string as specified by the format argument. If t is not provided, the current time as returned by local time() is used. format must be a string. ValueError is raised if any field in t is outside of the allowed range.

- %B- Locale's full month name.
- %p- Locale's equivalent of either AM or PM.
- %d- Day of the month as a decimal number [01,31].
- %I- Hour (12-hour clock) as a decimal number [01,12].
- %M- Minute as a decimal number [00,59].
- %Y- Year with century as a decimal number.

2. random:- The random module is a built-in module to generate the pseudo-random variables. It can be used perform some action randomly such as to get a random number, selecting a random elements from a list, shuffle elements randomly, etc.

a. choice():- The random.choice() method returns a randomly selected element from a non-empty sequence. An empty sequence as argument raises an Index Error.

3. pyttsx3:-**pyttsx3** is a text-to-speech conversion library in Python. Unlike alternative libraries, it works offline and is compatible with both Python 2 and 3. An application invokes the pyttsx3.init() factory function to get a reference to a pyttsx3. Engine instance. it is a very easy to use tool which converts the entered text into speech.

a. init():-The Instance of the initialized pyttsx3 package is stored in the engine variable. We are calling the variable engine as it works as the engine and converts Text-To-Speech whenever execute the functions from the package.

b. say():-There is a built-in say() function in the pyttsx3 package that takes a string value and speaks it out.

c. runAndWait():-This function keeps track when the engine starts converting text to speech and waits for that much time, and do not allow the engine to close. If we don't write this code,

it may happen that the engine might not work properly as the processes will not be synchronized.

d. getProperty():-The getProperty() function of the pyttsx3 package takes a string as a parameter and returns an object matching the string

4. threading:-Threading in python is used to run multiple threads (tasks, function calls) at the same time Threading allows python to execute other code while waiting; this is easily simulated with the sleep function.

a. Sleep():-The sleep() function suspends execution of the current thread for a given number of seconds.

5. Warning:-Warning messages are typically issued in situations where it is useful to alert the user of some condition in a program, where that condition (normally) doesn't warrant raising an exception and terminating the program.

a. filterwarnings(): This function adds an entry into the specifications of the warnings filter.

- "error"-turn matching warnings into exceptions
- "ignore"- never print matching warnings

6. Wikipedia:-Wikipedia is a multilingual online encyclopedia created and maintained as an open collaboration project by a community of volunteer editors using a wiki-based editing system.

a. summary():-The **summary()** method, which returns the article's summary or topic. This method takes the two arguments - title and sentences and returns the summary in the string format.

7. String:- string module contains a single utility function – capwords(s, sep=None). Then it capitalizes each word using str.

a. F-string:- It is a new string formatting mechanism introduced by the PEP 498. It is also known as Literal String Interpolation or more commonly as F-strings **(f character preceding the string literal).** The primary focus of this mechanism is to make the interpolation easier.

b. strip():-Remove spaces at the beginning and at the end of the string.

c. split():-This function split the specified string into words using str.

8. Nltk:- Natural Language ToolKit, also known as NLTK, is a library written in Python. NLTK library is generally used for symbolic and statistical Natural Language Processing and works well with textual data. Natural Language ToolKit (NLTK) is a Third-party Library that can be installed using the following syntax in a command shell or terminal:

a. Word Tokenize:- The word_tokenize() method is used to split a string into tokens or say words.

b. Sentence Tokenize:- The sent_tokenize() method is used to split a string or paragraph into sentences.

c. lemmatization:-Lemmatization is the process of grouping together the different inflected forms of a word so they can be analysed as a single item. Lemmatization is similar to stemming but it brings context to the words. So it links words with similar meaning to one word.

9.Tkinter:-Python provides the standard library Tkinter for creating the graphical user interface for desktop based applications. Developing desktop based applications with python Tkinter is not a complex task.

a. Button:-The Button is used to add various kinds of buttons to the python application.

b. Entry:-The entry widget is used to display the single-line text field to the user. It is commonly used to accept user values.

c. Frame:-It can be defined as a container to which, another widget can be added and organized.

d. Label:-A label is a text used to display some message or information about the other widgets.

e. Menubutton:-The Menu button is used to display the menu items to the user.

f. Menu:-It is used to add menu items to the user.

g. Message:-The Message widget is used to display the message-box to the user.

h. Scrollbar:-It provides the scrollbar to the user so that the user can scroll the window up and down.

i. Text:-It is different from Entry because it provides a multi-line text field to the user so that the user can write the text and edit the text inside it.

j. LabelFrame:-A LabelFrame is a container widget that acts as the container

k. MessageBox:-This module is used to display the message-box in the desktop based applications.

l. mainloop():-mainloop tells Python to run the Tkinter event loop. This method listens for events, such as button clicks or keypresses, and blocks any code that comes after it from running until the window it's called on is closed.

5.2 Application Code:

5.2.1 gui.py(main file):

```
from tkinter import *
import time
import tkinter.messagebox

import random
from basics import chat,Introduce_Ans
import pyttsx3
import threading

# engine = pyttsx3.init()
# voices=engine.getProperty('voices')
# engine.setProperty('voice', voices[1].id)


window_size = "400x400"
engine = pyttsx3.init()
voices = engine.getProperty('voices')
engine.setProperty('voice', voices[1].id)
engine.setProperty("rate", 150)



class ChatInterface(Frame):

    def __init__(self, master=None):
        Frame.__init__(self, master)
        self.master = master

        # sets default bg for top level windows
        self.tl_bg = "#EEEEEE"
        self.tl_bg2 = "#EEEEEE"
        self.tl_fg = "#000000"
        self.font = "Verdana 10"

        menu = Menu(self.master)
        self.master.config(menu=menu, bd=5)
        # Menu bar

        # File
        file = Menu(menu, tearoff=0)
        menu.add_cascade(label="File", menu=file)
        # file.add_command(label="Save Chat Log", command=self.save_chat)
        file.add_command(label="Clear Chat", command=self.clear_chat)
```

```
#file.add_command(label="Save", command=self.savechat)
file.add_separator()

file.add_command(label="Exit", command=self.chatexit)

# Options
options = Menu(menu, tearoff=0)
menu.add_cascade(label="Options", menu=options)

# username

# font
font = Menu(options, tearoff=0)
options.add_cascade(label="Font", menu=font)
font.add_command(label="Default", command=self.font_change_default)
font.add_command(label="Times", command=self.font_change_times)
font.add_command(label="System", command=self.font_change_system)
font.add_command(label="Helvetica", command=self.font_change_helvetica)
font.add_command(label="Fixedsys", command=self.font_change_fixedsys)

# color theme
color_theme = Menu(options, tearoff=0)
options.add_cascade(label="Color Theme", menu=color_theme)
color_theme.add_command(label="Default", command=self.color_theme_default)
# color_theme.add_command(label="Night",command=self.)
color_theme.add_command(label="Grey", command=self.color_theme_grey)
color_theme.add_command(label="Blue", command=self.color_theme_dark_blue)

color_theme.add_command(label="Torque", command=self.color_theme_turquoise)
color_theme.add_command(label="Hacker", command=self.color_theme_hacker)
# color_theme.add_command(label='Mkbhd',command=self.MKBHD)

help_option = Menu(menu, tearoff=0)
menu.add_cascade(label="Help", menu=help_option)
# help_option.add_command(label="Features", command=self.features_msg)
help_option.add_command(label="About PyBot", command=self.msg)
help_option.add_command(label="Develpoers", command=self.about)

self.text_frame = Frame(self.master, bd=6)
self.text_frame.pack(expand=True, fill=BOTH)

# scrollbar for text box
self.text_box_scrollbar = Scrollbar(self.text_frame, bd=0)
self.text_box_scrollbar.pack(fill=Y, side=RIGHT)

# contains messages
self.text_box = Text(self.text_frame, yscrollcommand=self.text_box_scrollbar.set, state=DISABLED,
                     bd=1, padx=6, pady=6, spacing3=8, wrap=WORD, bg=None, font="Verdana 10",
```

```
relief=GROOVE,
                      width=10, height=1)
        self.text_box.pack(expand=True, fill=BOTH)
        self.text_box_scrollbar.config(command=self.text_box.yview)

        # frame containing user entry field
        self.entry_frame = Frame(self.master, bd=1)
        self.entry_frame.pack(side=LEFT, fill=BOTH, expand=True)

        # entry field
        self.entry_field = Entry(self.entry_frame, bd=1, justify=LEFT)
        self.entry_field.pack(fill=X, padx=6, pady=6, ipady=3)
        # self.users_message = self.entry_field.get()

        # frame containing send button and emoji button
        self.send_button_frame = Frame(self.master, bd=0)
        self.send_button_frame.pack(fill=BOTH)

        # send button
        self.send_button = Button(self.send_button_frame, text="Send", width=5, relief=RAISED,
bg='white',
                          bd=1, command=lambda: self.send_message_insert(None),
activebackground="#FFFFFF",
                          activeforeground="#000000")
        self.send_button.pack(side=LEFT, ipady=8)
        self.master.bind("<Return>", self.send_message_insert)

        self.last_sent_label(date="No messages sent.")

        self.text_box.configure(state=NORMAL)
        x="About me -: "+ random.choice(Introduce_Ans)+"\n"
        self.text_box.insert(END, x)
        self.text_box.configure(state=DISABLED)
        self.text_box.see(END)

        # t2 = threading.Thread(target=self.send_message_insert(, name='t1')
        # t2.start()

    def playResponce(self, responce):

        engine.say(responce)
        engine.runAndWait()

    def last_sent_label(self, date):

        try:
            self.sent_label.destroy()
        except AttributeError:
```

```
        pass

    self.sent_label = Label(self.entry_frame, font="Verdana 7", text=date, bg=self.tl_bg2, fg=self.tl_fg)
    self.sent_label.pack(side=LEFT, fill=X, padx=3)

  def clear_chat(self):
    self.text_box.config(state=NORMAL)
    self.last_sent_label(date="No messages sent.")
    self.text_box.delete(1.0, END)
    self.text_box.delete(1.0, END)
    self.text_box.config(state=DISABLED)


  def chatexit(self):
    exit()

  def msg(self):
    tkinter.messagebox.showinfo("PyBOT v1.0",
                        'PyBOT is a chatbot for answering python queries\nIt is based on retrival-based NLP
using pythons NLTK tool-kit module\nGUI is based on Tkinter\nIt can answer questions regarding python
language for new learners')


  def about(self):
    tkinter.messagebox.showinfo("PyBOT Developers", " (18100BTCSE02661)Amit jain\n
(18100BTCSE02713)Prakhar Saki\n (18100BTCSE02716)Praveen Sharma")

  def send_message_insert(self, message):
    # print("X",x)



    user_input = self.entry_field.get()
    pr1 = "Human : " + user_input + "\n"
    f.write(pr1)
    self.text_box.configure(state=NORMAL)
    self.text_box.insert(END, pr1)
    self.text_box.configure(state=DISABLED)
    self.text_box.see(END)
    # t1 = threading.Thread(target=self.playResponce, args=(user_input,))
    # t1.start()
    # time.sleep(1)
    result = chat(user_input)
    pr = "PyBot : " + result + "\n"
    f.write(pr+"\n")
    self.text_box.configure(state=NORMAL)
    self.text_box.insert(END,pr)
    self.text_box.configure(state=DISABLED)
```

```
        self.text_box.see(END)
        self.last_sent_label(str(time.strftime("Last message sent: " + '%B %d, %Y' + ' at ' + '%I:%M %p')))
        self.entry_field.delete(0, END)
        # print()

        time.sleep(0)
        t2 = threading.Thread(target=self.playResponce, args=(result,))
        t2.start()
        # return ob

    def font_change_default(self):
        self.text_box.config(font="Verdana 10")
        self.entry_field.config(font="Verdana 10")
        self.font = "Verdana 10"

    def font_change_times(self):
        self.text_box.config(font="Times")
        self.entry_field.config(font="Times")
        self.font = "Times"

    def font_change_system(self):
        self.text_box.config(font="System")
        self.entry_field.config(font="System")
        self.font = "System"

    def font_change_helvetica(self):
        self.text_box.config(font="helvetica 10")
        self.entry_field.config(font="helvetica 10")
        self.font = "helvetica 10"

    def font_change_fixedsys(self):
        self.text_box.config(font="fixedsys")
        self.entry_field.config(font="fixedsys")
        self.font = "fixedsys"

    def color_theme_default(self):
        self.master.config(bg="#EEEEEE")
        self.text_frame.config(bg="#EEEEEE")
        self.entry_frame.config(bg="#EEEEEE")
        self.text_box.config(bg="#FFFFFF", fg="#000000")
        self.entry_field.config(bg="#FFFFFF", fg="#000000", insertbackground="#000000")
        self.send_button_frame.config(bg="#EEEEEE")
        self.send_button.config(bg="#FFFFFF", fg="#000000", activebackground="#FFFFFF",
activeforeground="#000000")
        # self.emoji_button.config(bg="#FFFFFF", fg="#000000", activebackground="#FFFFFF",
activeforeground="#000000")
        self.sent_label.config(bg="#EEEEEE", fg="#000000")
```

```
        self.tl_bg = "#FFFFFF"
        self.tl_bg2 = "#EEEEEE"
        self.tl_fg = "#000000"

    # Dark
    def color_theme_dark(self):
        self.master.config(bg="#2a2b2d")
        self.text_frame.config(bg="#2a2b2d")
        self.text_box.config(bg="#212121", fg="#FFFFFF")
        self.entry_frame.config(bg="#2a2b2d")
        self.entry_field.config(bg="#212121", fg="#FFFFFF", insertbackground="#FFFFFF")
        self.send_button_frame.config(bg="#2a2b2d")
        self.send_button.config(bg="#212121", fg="#FFFFFF", activebackground="#212121",
activeforeground="#FFFFFF")
        # self.emoji_button.config(bg="#212121", fg="#FFFFFF", activebackground="#212121",
activeforeground="#FFFFFF")
        self.sent_label.config(bg="#2a2b2d", fg="#FFFFFF")

        self.tl_bg = "#212121"
        self.tl_bg2 = "#2a2b2d"
        self.tl_fg = "#FFFFFF"

    # Grey
    def color_theme_grey(self):
        self.master.config(bg="#444444")
        self.text_frame.config(bg="#444444")
        self.text_box.config(bg="#4f4f4f", fg="#ffffff")
        self.entry_frame.config(bg="#444444")
        self.entry_field.config(bg="#4f4f4f", fg="#ffffff", insertbackground="#ffffff")
        self.send_button_frame.config(bg="#444444")
        self.send_button.config(bg="#4f4f4f", fg="#ffffff", activebackground="#4f4f4f",
activeforeground="#ffffff")
        # self.emoji_button.config(bg="#4f4f4f", fg="#ffffff", activebackground="#4f4f4f",
activeforeground="#ffffff")
        self.sent_label.config(bg="#444444", fg="#ffffff")

        self.tl_bg = "#4f4f4f"
        self.tl_bg2 = "#444444"
        self.tl_fg = "#ffffff"

    def color_theme_turquoise(self):
        self.master.config(bg="#003333")
        self.text_frame.config(bg="#003333")
        self.text_box.config(bg="#669999", fg="#FFFFFF")
        self.entry_frame.config(bg="#003333")
        self.entry_field.config(bg="#669999", fg="#FFFFFF", insertbackground="#FFFFFF")
        self.send_button_frame.config(bg="#003333")
        self.send_button.config(bg="#669999", fg="#FFFFFF", activebackground="#669999",
```

```
activeforeground="#FFFFFF")
        # self.emoji_button.config(bg="#669999", fg="#FFFFFF", activebackground="#669999",
activeforeground="#FFFFFF")
        self.sent_label.config(bg="#003333", fg="#FFFFFF")

        self.tl_bg = "#669999"
        self.tl_bg2 = "#003333"
        self.tl_fg = "#FFFFFF"

        # Blue

    def color_theme_dark_blue(self):
        self.master.config(bg="#263b54")
        self.text_frame.config(bg="#263b54")
        self.text_box.config(bg="#1c2e44", fg="#FFFFFF")
        self.entry_frame.config(bg="#263b54")
        self.entry_field.config(bg="#1c2e44", fg="#FFFFFF", insertbackground="#FFFFFF")
        self.send_button_frame.config(bg="#263b54")
        self.send_button.config(bg="#1c2e44", fg="#FFFFFF", activebackground="#1c2e44",
activeforeground="#FFFFFF")
        # self.emoji_button.config(bg="#1c2e44", fg="#FFFFFF", activebackground="#1c2e44",
activeforeground="#FFFFFF")
        self.sent_label.config(bg="#263b54", fg="#FFFFFF")

        self.tl_bg = "#1c2e44"
        self.tl_bg2 = "#263b54"
        self.tl_fg = "#FFFFFF"

    # Torque
    def color_theme_turquoise(self):
        self.master.config(bg="#003333")
        self.text_frame.config(bg="#003333")
        self.text_box.config(bg="#669999", fg="#FFFFFF")
        self.entry_frame.config(bg="#003333")
        self.entry_field.config(bg="#669999", fg="#FFFFFF", insertbackground="#FFFFFF")
        self.send_button_frame.config(bg="#003333")
        self.send_button.config(bg="#669999", fg="#FFFFFF", activebackground="#669999",
activeforeground="#FFFFFF")
        # self.emoji_button.config(bg="#669999", fg="#FFFFFF", activebackground="#669999",
activeforeground="#FFFFFF")
        self.sent_label.config(bg="#003333", fg="#FFFFFF")

        self.tl_bg = "#669999"
        self.tl_bg2 = "#003333"
        self.tl_fg = "#FFFFFF"

    # Hacker
    def color_theme_hacker(self):
```

```
        self.master.config(bg="#0F0F0F")
        self.text_frame.config(bg="#0F0F0F")
        self.entry_frame.config(bg="#0F0F0F")
        self.text_box.config(bg="#0F0F0F", fg="#33FF33")
        self.entry_field.config(bg="#0F0F0F", fg="#33FF33", insertbackground="#33FF33")
        self.send_button_frame.config(bg="#0F0F0F")
        self.send_button.config(bg="#0F0F0F", fg="#FFFFFF", activebackground="#0F0F0F",
activeforeground="#FFFFFF")
        # self.emoji_button.config(bg="#0F0F0F", fg="#FFFFFF", activebackground="#0F0F0F",
activeforeground="#FFFFFF")
        self.sent_label.config(bg="#0F0F0F", fg="#33FF33")

        self.tl_bg = "#0F0F0F"
        self.tl_bg2 = "#0F0F0F"
        self.tl_fg = "#33FF33"

    # Default font and color theme
    def default_format(self):
        self.font_change_default()
        self.color_theme_default()


root = Tk()

a = ChatInterface(root)
root.geometry(window_size)
root.title("PyBot")
root.iconbitmap('i.ico')
x=str(time.strftime('%I-%M-%S %p'))
f = open(f"save{x}.txt", "a+",encoding='UTF8')
# print(random.choice(Introduce_Ans))
# text_box.insert(END, random.choice(Introduce_Ans))

root.mainloop()
f.close()
```

5.2.2 basics.py:

```
# Meet Pybot: your friend
import webbrowser
import nltk
import warnings
import wikipedia
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.metrics.pairwise import cosine_similarity
warnings.filterwarnings("ignore")

import random
import string

f = open('modules.txt', 'r', errors='ignore')
m = open('modules.txt', 'r', errors='ignore')
# checkpoint = "./chatbot_weights.ckpt"
# session = tf.InteractiveSession()
# session.run(tf.global_variables_initializer())
# saver = tf.train.Saver()
# saver.restore(session, checkpoint)

raw = f.read()
raw = raw
nltk.download('punkt')  # first-time use only
nltk.download('wordnet')  # first-time use only
sent_tokens = nltk.sent_tokenize(raw)  # converts to list of sentences
word_tokens = nltk.word_tokenize(raw)  # converts to list of words

lemmer = nltk.stem.WordNetLemmatizer()


def LemTokens(tokens):
    return [lemmer.lemmatize(token) for token in tokens]


remove_punct_dict = dict((ord(punct), None) for punct in string.punctuation)


def LemNormalize(text):
    return LemTokens(nltk.word_tokenize(text.lower().translate(remove_punct_dict)))


Introduce_Ans = ["My name is PyBot.", "My name is PyBot you can called me pi.", "Im PyBot :) ",
                 "My name is PyBot. and my nickname is pi and i am happy to solve your queries :) "]
GREETING_INPUTS = ("hello", "hi", "hiii", "hii", "hiiii", "hiiii", "greetings", "sup", "what's up", "hey",)
GREETING_RESPONSES = ["hi", "hey", "hii there", "hi there", "hello", "I am glad! You are talking to
me"]
Basic_Q = ("what is python ?", "what is python", "what is python?", "what is python.")
Basic_Ans = "Python is a high-level, interpreted, interactive and object-oriented scripting programming
```

```
language python is designed to be highly readable It uses English keywords frequently where as other languages use punctuation, and it has fewer syntactical constructions than other languages."
Basic_Om = ("what is module", "what is module.", "what is module ", "what is module ?", "what is module?",
          "what is module in python", "what is module in python.", "what is module in python?",
          "what is module in python ?")
Basic_AnsM = ["Consider a module to be the same as a code library.",
           "A file containing a set of functions you want to include in your application.",
           "A module can define functions, classes and variables. A module can also include runnable code. Grouping related code into a module makes the code easier to understand and use."]

# Checking for greetings
def greeting(sentence):
    """If user's input is a greeting, return a greeting response"""
    for word in sentence.split():
        if word.lower() in GREETING_INPUTS:
            return random.choice(GREETING_RESPONSES)


# Checking for Basic_Q
def basic(sentence):
    for word in Basic_Q:
        if sentence.lower() == word:
            return Basic_Ans


# Checking for Basic_QM
def basicM(sentence):
    """If user's input is a greeting, return a greeting response"""
    for word in Basic_Om:
        if sentence.lower() == word:
            return random.choice(Basic_AnsM)

#introduction
def IntroduceMe(sentence):
    return random.choice(Introduce_Ans)

# Generating response
def response(user_response):
    found=0
    with open("modules.txt","r",encoding="utf8") as f:
        for i in f:
            m=i[0:len(user_response)].lower().strip()
            if m==user_response:
                result=i
                # engine.say(i[len(user_querry)+3: ])
                # engine.runAndWait()

                found=1
                break
        if found==0:
```

```
        try:
            result = wikipedia.summary(f"%{user_response}", sentences=2)
        except:
            result=f"{user_response} is not found in data base"
    return result

#chat function
def chat(user_response):
    user_response=user_response.lower()
    if (user_response != 'bye'):
        if (user_response == 'thanks' or user_response == 'thank you'):
            return "You are welcome.."
        elif (basicM(user_response) != None):
            return basicM(user_response)
        else:
            if (greeting(user_response) != None):
                return greeting(user_response)
            elif (user_response.find("your name") != -1 or user_response.find(
                    "who are you") != -1 or user_response.find("your name ") != -1 or user_response.find(
                    " your name ") != -1):
                return IntroduceMe(user_response)
            elif (basic(user_response) != None):
                return basic(user_response)
            else:
                return response(user_response)

    else:
        return "Bye! take care.."
```

5.2.3 modules.txt:

Python decorator is a specific change that we make in Python syntax to alter functions.

String store characters and have many built-in convenience methods that let you modify their content strings are immutable, meaning they cannot be changed in place.

Variables are assigned values using the = operator, which is not to be confused with the == sign used for testing equality, a variable can hold almost any type of value such as lists, dictionaries, functions.

Pickle module accepts any Python object and converts it into a string representation and dumps it into a file by using dump function this process is called pickling While the process of retrieving original Python objects from the stored string representation is called unpickling.

Map function executes the function given as the first argument on all the elements of the iterable given as the second argument If the function given takes in more than 1 arguments then many iterables are given.

TkInter is Python library It is a toolkit for GUI development It provides support for various GUI tools or widgets (such as buttons, labels, text boxes, radio buttons, etc) that are used in GUI applications The common attributes of them include Dimensions, Colors, Fonts, Cursors, etc.

A Module is a Python script that generally contains import statements, functions, classes and variable definitions, and Python runnable code and it “lives” file with a ‘.py’ extension zip files and DLL files can also be modules.Inside the module, you can refer to the module name as a string that is stored in the global variable name.

Lambda is a single expression anonymous function often used as inline function in python.

strip() in-built function of Python is used to remove all the leading and trailing spaces from a string.

In Python every name introduced has a place where it lives and can be hooked for This is known as namespace It is like a box where a variable name is mapped to the object placed Whenever the variable is searched out, this box will be searched, to get corresponding object.

Pass means, no-operation Python statement, or in other words it is a place holder in compound statement, where there should be a blank left and nothing has to be written there.

A tuple is another sequence data type that is similar to the list A tuple consists of a number of values separated by commas Unlike lists, however, tuples are enclosed within parentheses.

Dictionaries are kind of hash table type They work like associative arrays or hashes found in Perl and consist of key-value pairs a dictionary key can be almost any Python type, but are usually numbers or strings Values, on the other hand, can be any arbitrary python object.

Pickle module accepts any Python object and converts it into a string representation and dumps it into a file by using dump function, this process is called pickling.

Unpickling process of retrieving original Python objects from the stored string representation is called unpickling.

Iterators are used for iterating a group of elements, containers like list.

Slicing is a mechanism that helps you in selecting a range of items from sequence types such as tuple, string, list, etc.

Docstring is another name for a Python documentation string It is way to document Python modules, classes and functions.

Unittest is the unit testing framework in Python It aids in automation testing, sharing of setups, aggregation of tests into collections, shutdown code tests, etc.

You can delete a file with the help of command (os.remove(filename)) or (os.unlink (filename)).

Split function in Python helps in breaking a string into shorter strings by using the defined separator It provides a list of the words contained in the string.

Features python is an interpreted language, dynamically typed, well suited to object orientated programming, code is quick,etc.

Python memory is managed by Python private heap space All Python objects and data structures are located in a private heap.

Flask is a web micro framework for Python based on “Werkzeug, Jinja2 and good intentions” BSD license Werkzeug and Jinja2 are two of its dependencies This means it will have little to no dependencies on external libraries.

Help function is used to display the documentation string and also facilitates you to see the help related to modules, keywords, attributes, etc.

The dir() function is used to display the defined symbols.

Monkey patch only refers to dynamic modifications of a class or module at run-time.

We use *args when we aren’t sure how many arguments are going to be passed to a function, or if we want to pass a stored list or tuple of arguments to a function.

**kwargs is used when we don’t know how many keyword arguments will be passed to a function, or it can be used to pass the values of a dictionary as keyword arguments.

Django provides session that lets you store and retrieve data on a per-site-visitor basis Django abstracts the process of sending and receiving cookies, by placing a session ID cookie on the client side, and storing all the related data on the server side.

Operator evaluates to true if the variables on either side of the operator point to the same object and false otherwise x is y, here is results in 1 if id(x) equals id(y).

Break terminates the loop statement and transfers execution to the statement immediately following the loop.

Continue auses the loop to skip the remainder of its body and immediately retest its condition prior to reiterating.

Sqlite3 is the standard library for python for database.

In python we can use import socket a standard library for socket programming in python for client server interaction.

Yes Python support MongoDB.

import threading can be used for threaded application in python.

Opencv is the computer vision library in python.

Yes python support creating (.exe) file using module cx_freez.

Pycharm,idle,Spyder are various ide for python programming.

Yes python can be used for mobile app development using module kivy.

Shallow copy is used when a new instance type gets created and it keeps the values that are copied in the new instance and Deep copy is used to store the values that are already copied Deep copy doesn't copy the reference pointers to the objects.

Built-in datatypes in Python is called dictionary It defines one-to-one relationship between keys and values Dictionaries contain pair of keys and their corresponding values. Dictionaries are indexed by keys.

Random module is the standard module that is used to generate the random number.

Flask is a "microframework" primarily build for a small application with simpler requirements In flask, you have to use external libraries Flask is ready to use Pyramid is built for larger applications It provides flexibility and lets the developer use the right tools for their project The developer can choose the database, URL structure, templating style and more Pyramid is heavy configurable Django can also used for larger applications just like Pyramid It includes an ORM.

Matplotlib provides basic 3D plotting in the mplot3d.

Numbers,Strings,List,Tuples,Dictionaries are the types of data type in python.

(list.reverse()) Reverses objects of list in place.

In python generally "with" statement is used to open a file, process the data present in the file, and also to close the file without calling a close() method "with" statement makes the exception handling simpler by providing cleanup activities.

Yes!! Python is Object Oriented Programming language.

Python does not provide interfaces like in Java Abstract Base Class (ABC) and its feature are provided by the Python's "abc" module.

Both append() and extend() methods are the methods of list These methods a regex used to add the elements at the end of the list.

We have three logical operators and , or , not.

Tuple packing is a way to pack a set of values into a tuple.

In python (.py) files are Python source files (.pyc) files are the compiled bvtecode files that is generated by the Python compiler.

In python try, except and finally blocks are used in Python error handling.

Modules are brought in python via the import statement.

Functions in python are defined using the def statement An example might be def foo(bar):.

Class are created using the class statement An example might be class abc(parameter):.

In python functions return values using the return statement.

The lambda operator is used to create anonymous functions It is mostly used in cases where one wishes to pass functions as parameters or assign them to variable names.

Local namespaces are created within a function when that function is called Global name spaces are created when the program starts.

In python control statements are for and While Loop.

The break statement stops execution of the current loop and transfers control to the next block The continue statement ends the current block's execution and jumps to the next iteration of the loop.

Pyqt5,tkinter,pyside2 are the gui modules in python.

Numpy,scipy,tensorflow,keras are the scientific libraries in python.

Pygame is the game development module in python.

Pandas is the data filtering module in python used for data preprocessing.

Pygame is a module through which we can make/create games in python.

Pythonpath has a role similar to PATH PYTHONPATH variable tells the Python interpreter where to

locate the module files imported into a program PYTHONPATH should include the Python source library directory.

Pythonstartup contains the path of an initialization file containing python source code.

Pythoncaseok is used in windows to instruct python to find the first case-insensitive match in an import statement.

Pythonhome is an alternative module search path It is usually embedded in the PYTHONSTARTUP or PYTHONPATH directories to make switching module libraries easy.

Inheritance is a process of using details from a new class without modifying existing class.

Encapsulation is hiding the private details of a class from other objects.

Polymorphism is a concept of using common operation in different ways for different data input.

File is a named location on disk to store related information, it is used to permanently store data in a non-volatile memory example hard disk.

Directory are a large number of files to handle in your program, you can arrange your code within different directories to make things more manageable.

AssertionError is raised when assert statement fails.

AttributeError is raised when attribute assignment or reference fails.

EOFError is raised when the input() functions hits end-of-file condition.

FloatingPointError is raised when a floating point operation fails.

GeneratorExit is raise when a generator's close() method is called.

ImportError is raised when the imported module is not found.

IndexError is raised when index of a sequence is out of range.

KeyError is raised when a key is not found in a dictionary.

KeyboardInterrupt is raised when the user hits interrupt key (Ctrl+c or delete).

MemoryError is raised when an operation runs out of memory.

NameError is raised when a variable is not found in local or global scope.

NotImplementedError is raised by abstract methods.

OSError is raised when system operation causes system related error.

OverflowError is raised when result of an arithmetic operation is too large to be represented.

ReferenceError is raised when a weak reference proxy is used to access a garbage collected referent.

RuntimeError is raised when an error does not fall under any other category.

StopIteration is raised by next() function to indicate that there is no further item to be returned by iterator.

SyntaxError is raised by parser when syntax error is encountered.

IndentationError is raised when there is incorrect indentation.

TabError is raised when indentation consists of inconsistent tabs and spaces.

SystemError is raised when interpreter detects internal error.

SystemExit is raised by [sys.exit()] function.

TypeError is raised when a function or operation is applied to an object of incorrect type.

UnboundLocalError is raised when a reference is made to a local variable in a function or method, but no value has been bound to that variable.

UnicodeError is raised when a Unicode-related encoding or decoding error occurs.

UnicodeEncodeError is raised when a Unicode-related error occurs during encoding.

UnicodeDecodeError is raised when a Unicode-related error occurs during decoding.

UnicodeTranslateError is raised when a Unicode-related error occurs during translating.

ValueError is raised when a function gets argument of correct type but improper value.

ZeroDivisionError is raised when second operand of division or modulo operation is zero.

Exceptions which forces your program to output an error when something in it goes wrong.

List Comprehensions is convenient ways to generate or extract information from lists.

Lists is data type that holds an ordered collection of values, which can be of any type and they are ordered mutable data type unlike tuples, lists can be modified in-place.

While loop permits code to execute repeatedly until a certain condition is met.

print() is a function to display the output of a program, using the parenthesized version is arguably more consistent.

range() function returns a list of integers, the sequence of which is defined by the arguments passed to it.

Sets are collections of unique but unordered items, it is possible to convert certain iterables to a set.

Abs return the absolute value of a number.

Argument is an extra information which the computer uses to perform commands.

Argparse is a parser for command-line options, arguments and subcommands.

Assert is used during debugging to check for conditions that ought to apply

Break is used to exit a for loop or a while loop.

Class is a template for creating user-defined objects.

Compiler translates a program written in a high-level language into a low-level language.

Continue used to skip the current block, and return to the "for" or "while" statement

Conditional statement contains an "if" or "if/else".

Debugging is the process of finding and removing programming errors.

def Defines a function or method

distutils package included in the Python Standard Library for installing, building and distributing python code.

Docstring is a string literal that occurs as the first statement in a module,function, class, or method definition.

Easy Install is a python module (easy_install) bundled with setuptools that lets you automatically download, build, install, and manage Python packages.

Evaluation order python evaluates expressions from left to right

Exceptions means of breaking out of the normal flow of control of a code block in order to handle errors or other exceptional conditions

Expression is a python code that produces a value.

filter (function, sequence) returns a sequence consisting of those items from the sequence for which function(item) is true

float is an immutable floating point number.

for iterates over an iterable object, capturing each element to a local variable for use by the attached block

function is a parameterized sequence of statements.

function call is an invocation of the function with arguments.

garbage collection i the process of freeing memory when it is not used anymore.

generators is a function which returns an iterator.

high level language is designed to be easy for humans to read and write.

IDLE is abbreviation for Integrated development environment

immutable values cannot be changed after its created.

import is used to import modules whose functions or variables can be used in the current program.

indentation python uses white-space indentation, rather than curly braces or keywords,to delimit blocks.

int is an immutable integer of unlimited magnitude.

interactive mode is when commands are read from a tty, the interpreter is said to be in interactive mode.

interpret execute a program by translating it one line at a time.

IPython is an Interactive shell for interactive computing.

iterable is an object capable of returning its members one at a time.

Literals are notations for constant values of some built-in types.

map (function, iterable, ...) apply function to every item of iterable and return a list of the results.

method is like a function, but it runs "on" an object.

module is the basic unit of code reusability in python,a block of code imported by some other code.

object is any data with state (attributes or value) and defined behavior (methods).

object-oriented allows users to manipulate data structures called objects in order to build and execute programs.

pass is needed to create an empty code block

PEP 8 is a set of recommendations how to write Python code.

Python Package Index is official repository of third-party software for Python.

Pythonic is an idea or piece of code which closely follows the most common idioms of the python language, rather than implementing code using concepts common to other languages.

set an Unordered set, contains no duplicates

setuptools is a collection of enhancements to the Python distutils that allow you to more easily build and distribute Python packages

slice is a sub parts of sequences

strings can include numbers, letters, and various symbols and be enclosed by either double or single quotes, although single quotes are more commonly used.

statement is a statement is part of a suite (a "block" of code).

try allows exceptions raised in its attached code block to be caught and handled by except clauses.

with encloses a code block within a context manager.

yield returns a value from a generator function.

Zen of Python is When you type "import this", Python's philosophy is printed out.

annotation is a label associated with a variable, a class attribute or a function parameter or return value, used by convention as a type hint.

asynchronous context manager is an object which controls the environment seen in an async with statement by defining __aenter__() and __aexit__() methods introduced by PEP 492.

asynchronous generator is a function which returns an asynchronous generator iterator it looks like a coroutine function defined with async def except that it contains yield expressions for producing a series of values usable in an async for loop.

asynchronous generator iterator is an object created by a asynchronous generator function.

asynchronous iterable is an object, that can be used in an async for statement Must return an asynchronous iterator from its __aiter__() method introduced by PEP 492.

waitable is an object that can be used in an await expression Can be a coroutine or an object with an __await__() method

BDFL Benevolent Dictator For Life, [a.k.a.] Guido van Rossum, Python's creator.

binary file is a file object able to read and write bytes-like objects Examples of binary files are files opened in binary mode ('rb', 'wb' or 'rb+'), [sys.stdin.buffer], [sys.stdout.buffer], and instances of [io.BytesIO] and [gzip.GzipFile].

docstring is a string literal which appears as the first expression in a class, function or module.

duck-typing is a programming style which does not look at an object's type to determine if it has the right interface; instead, the method or attribute is simply called or used By emphasizing interfaces rather than specific types, well-designed code improves its flexibility by allowing polymorphic substitution.

EAFP Easier to ask for forgiveness than permission. This common Python coding style assumes the existence of valid keys or attributes and catches exceptions if the assumption proves false.

extension module is a module written in C or C++, using Python's C API to interact with the core and with user code.

GIL global interpreter lock is the mechanism used by the CPython interpreter to assure that only one thread executes Python bytecode at a time.

hash-based pyc is a bytecode cache file that uses the hash rather than the last-modified time of the corresponding source file to determine its validity.

Hashability makes an object usable as a dictionary key and a set member, because these data structures use the hash value internally.

key function or collation function is a callable that returns a value used for sorting or ordering For example, [locale.strxfrm()] is used to produce a sort key that is aware of locale specific sort conventions.

LBYL Look before you leap This coding style explicitly tests for pre-conditions before making calls or lookups This style contrasts with the EAFP approach and is characterized by the presence of many if statements.

loader is an object that loads a module It must define a method named load_module() A loader is typically returned by a finder.

mapping is a container object that supports arbitrary key lookups and implements the methods specified in the Mapping or MutableMapping abstract base classes.

meta path finder is a finder returned by a search of [sys.meta_path] Meta path finder's are related to, but different from path entry finders.

metaclass is the class of a class, Class definitions create a class name, a class dictionary, and a list of base classes.

Method Resolution Order is the order in which base classes are searched for a member during lookup.

namespace is the place where a variable is stored, Namespaces are implemented as dictionaries, There

are the local, global and built-in namespaces as well as nested namespaces in objects (in methods).

nested scopeis the ability to refer to a variable in an enclosing definition, For instance, a function defined inside another function can refer to variables in the outer function.

new-style class is old name for the flavor of classes now used for all class objects.

path entry is a single location on the import path which the path based finder consults to find modules for importing.

path entry finder a finder returned by a callable on sys.path_hooks which knows how to locate modules given a path entry.

path entry hook is a callable on the [sys.path_hook] list which returns a path entry finder if it knows how to find modules on a specific path entry.

path based finder is one of the default meta path finders which searches an import path for modules.

path-like object is an object representing a file system path.

Python Enhancement Proposal PEP is a design document providing information to the Python community, or describing a new feature for Python or its processes or environment, PEPs should provide a concise technical specification and a rationale for proposed features.

portion is a set of files in a single directory that contribute to a namespace package, as defined in PEP 420.

Provisional API is one which has been deliberately excluded from the standard library's backwards compatibility guarantees.

single dispatch is a form of generic function dispatch where the implementation is chosen based on the type of a single argument.

slice is an object usually containing a portion of a sequence.

special method is a method that is called implicitly by Python to execute a certain operation on a type, such as addition.

struct sequence is a tuple with named elements, Struct sequences expose an interface similar to named tuple in that elements can be accessed either by index or as an attribute.

type alias is a synonym for a type, created by assigning the type to an identifier, Type aliases are useful for simplifying type hints.

type hint is an annotation that specifies the expected type for a variable, a class attribute, or a function parameter or return value.

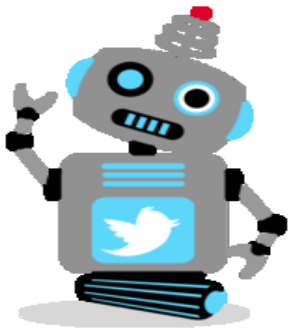
virtual environment is a cooperatively isolated runtime environment that allows Python users and

applications to install and upgrade Python distribution packages without interfering with the behaviour of other Python applications running on the same system.

virtual machine is a computer defined entirely in software Python's virtual machine executes the bytecode emitted by the bytecode compiler.

Zen of Python is Listing of Python design principles and philosophies that are helpful in understanding and using the language, The listing can be found by typing "import this" at the interactive prompt.

5.2.4 i.ico (Image)

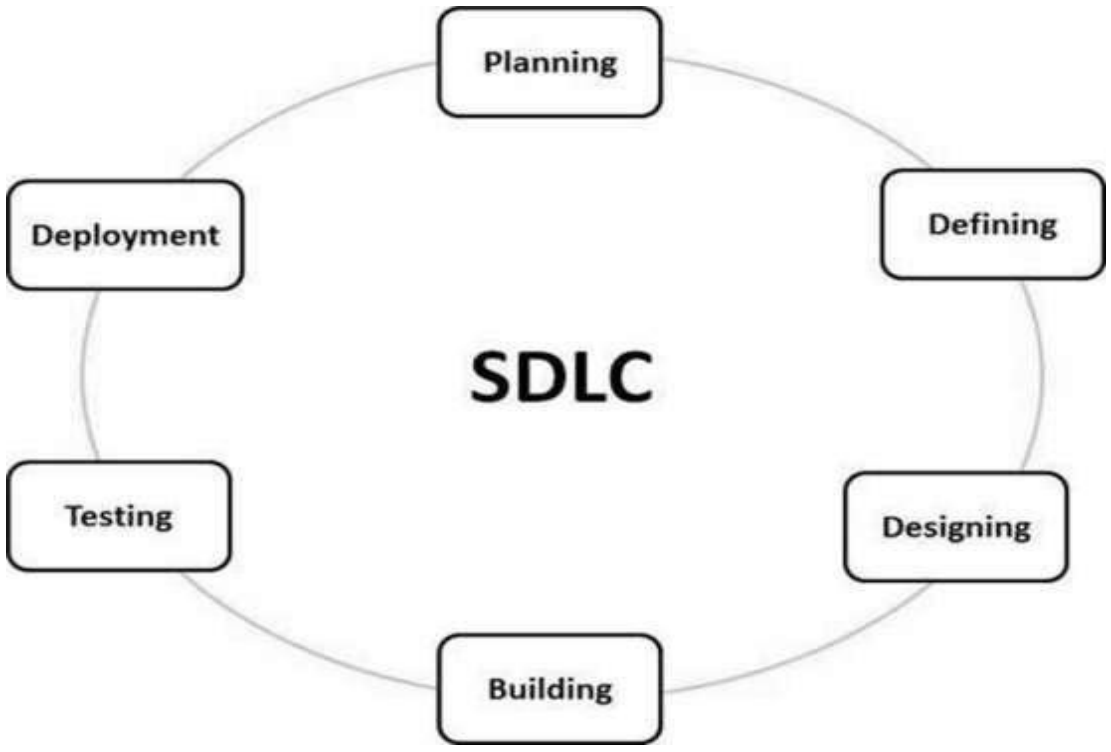


6. SOFTWARE ENGINEERING APPROACH

6.1 Software Development Life Cycle (SDLC)

SDLC is a process followed for a software project, within a software organization. It consists of a detailed plan describing how to develop, maintain, replace and alter or enhance specific software. The life cycle defines a methodology for improving the quality of software and the overall development process.

The following figure is a graphical representation of the various stages of a typical SDLC.



6.2Incremental Model

Incremental Model is a process of software development where requirements divided into multiple standalone modules of the software development cycle. In this model, each module goes through the requirements, design, implementation and testing phases. Every subsequent release of the module adds function to the previous release. The process continues until the complete system achieved.

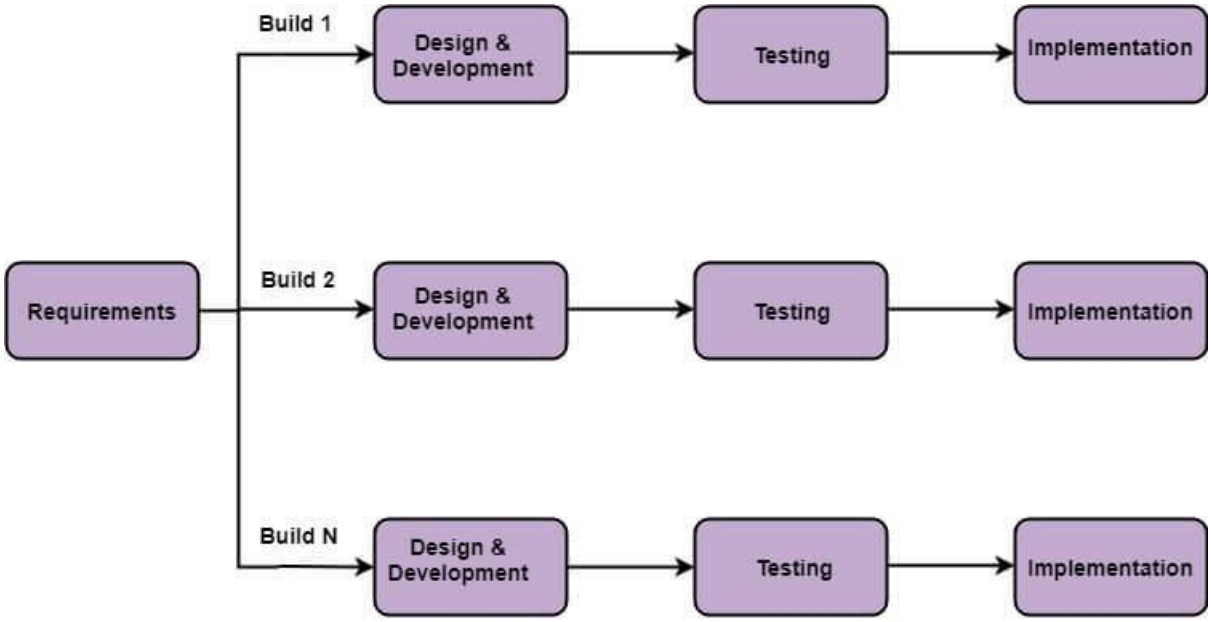
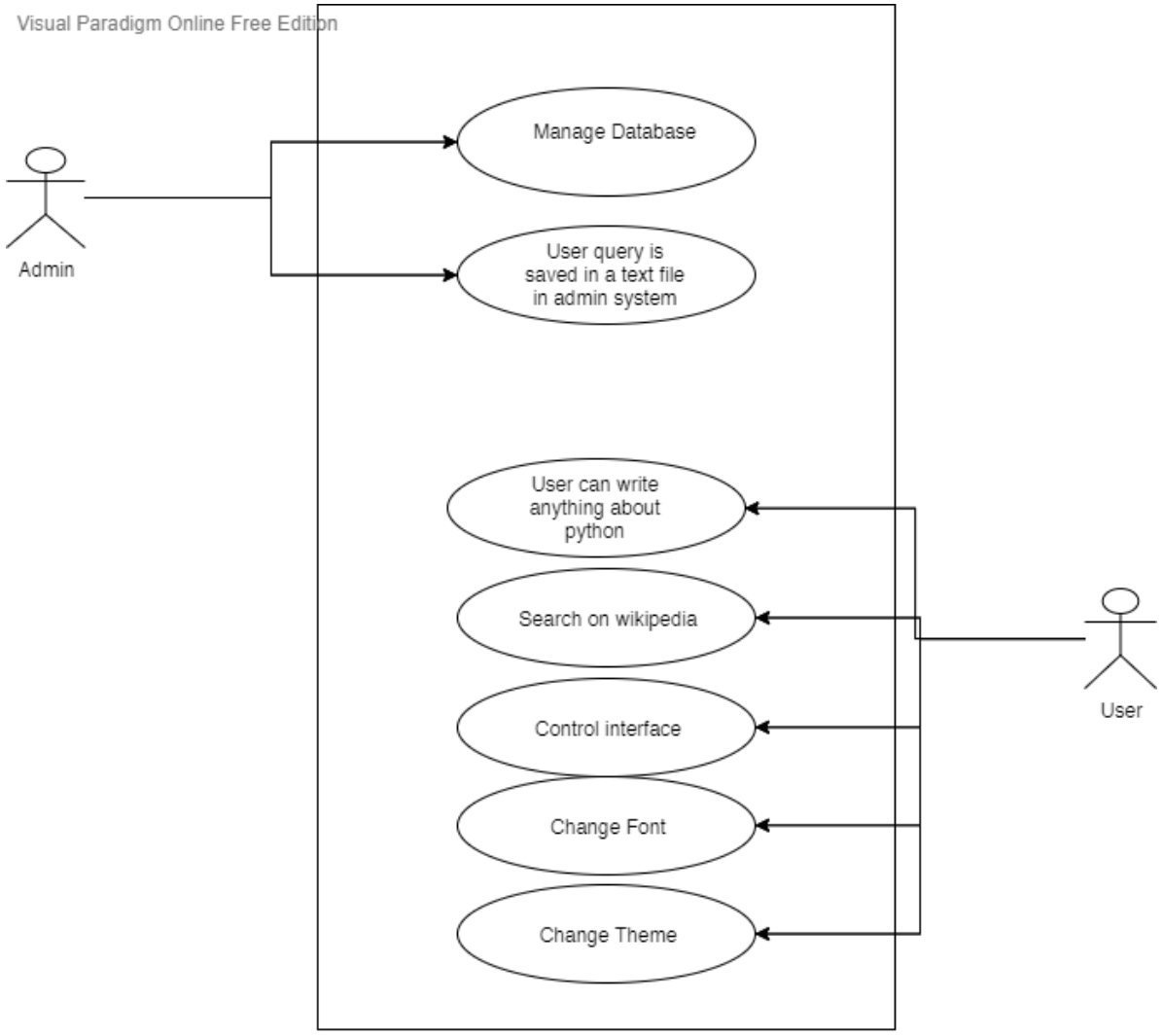


Fig: Incremental Model

6.3 Use Case Diagram of Real Time Support System



Use Case Diagram: Real Time Support System

7. CONCLUSION & DISCUSSION

7.1 Limitation of Project:

1. This is a prototype of a working chatbot which is used in online websites, hence it can only fetch data from either a text file named modules.txt or from a module of Wikipedia.

2. It can only work in English language.

3. It does not implement unsupervised learning.

7.2 Difficulties Encountered:

1.While saving a file of what a user have asked for in chat system it was replacing the previous file due to same name. We overcome this problem by naming each file name with the exact time when a user have stated his first statement and for this we use time module.

2.In starting while fetching data we were only getting data from a text file that is module.txt, which we thought was a little bit less data for this application. So to overcome this issue we surfed the internet and at last we got the solution in the form of Wikipedia module as it has numerous number of information in mostly every field related to our day to day life.

7.3 Future Enhancement & Suggestions

1.We will try to connect its database with Google search so that it will have a wide number of solutions for any particular problem.

2.We will also try to put a voice command for users so that they can easily tell their issue in voice note instead of writing in text.

3.In future it can be implemented websites so that users can get appropriate solution for their problems immediately. Also it is secure so no other person will be able to see what one person is searching on this system.

8.BIBLIOGRAPHY AND REFERENCES

8.1.1.References Links

- **https://www.google.com/**
- **https://www.wikipedia.org/**
- **https://stackoverflow.com/**
- **https://www.javatpoint.com/python-tkinter**
- **https://www.youtube.com/watch?v=VMP1oQOxfM0**
- **https://realpython.com/**
- **https://docs.python.org/3/**

8.1.2 References Books:

- **Python Projects for Beginners:-** Connor P. Milliken
- **Python and Tkinter Programming:-** John Grayson
- **Natural Language Processing With Python:-** Frank Millstein