ARTIFICIAL INTELLIGENT PERSONAL ASSISTANT

A Project

Submitted in partial fulfillment of the requirements for

The award of the diploma in

COMPUTER SCIENCE & TECHNOLOGY

By

SHUVOJIT CHOWDHURY

ROLL NO: D177055006-01039 AND REGISTRATION NO D161721682

RAHUL DHAR

ROLL NO: D177055006-01039 AND REGISTRATION NO D177055006

SAMIR CHANDRA PRAMANIK

ROLL NO: D177055006-01040 AND REGISTRATION NO D161721683

SOUVIK SEN

ROLL NO: D177055006-01039 AND REGISTRATION NO D177055006

DEPARTMENT OF COMPUTER SCIENCE & TECHNOLOGY

REGENT INSTITUTE OF SCIENCE & TECHNOLOGY

2019

DECLARATION CERTIFICATE

This is to certify that the work presented in the project entitled **"ARTIFICIAL INTELLIGENT PERSONAL ASSISTANT"** in partial fulfillment of the requirement for the award of diploma in **Computer Science & Technology** of **Regent Institute of Science & Technology** is an authentic work carried out under my supervision and guidance.

To the best of my knowledge the content of this thesis does not form a basis for the award of any previous diploma to anyone else.

Date:

(Guide's Name and Signature)

Dept. of Computer Science & Technology

Regent Institute of Science & Technology

(Name and Signature of TIC)

Dept. of Computer Science & Technology

Regent Institute of Science & Technology

Signature of External Examiner

CERTIFICATE OF APPROVAL

The foregoing project entitled **“ARTIFICIAL INTELLIGENT PERSONAL ASSISTANT”** is hereby approved as a creditable study and has been presented in satisfactory manner to warrant its acceptance as prerequisite to the degree/diploma for which it has been submitted.

It is understood that by this approval, the undersigned do not necessarily endorse any conclusion drawn or opinion expressed therein, but approve the thesis for the purpose for which it is submitted.

Signature of Internal Examiner **Signature of External Examiner**

Acknowledgements

We take this occasion to thank God, almighty for blessing us with his grace and taking our endeavor to a successful culmination. We extend our sincere and heartfelt thanks to our esteemed guide, Mr. **DEBAYAN BHATTACHARYA**. For providing us with the right guidance and advice at the crucial junctures and Showing me the right way. We take this opportunity to express a deep sense of gratitude to our class coordinators, Mr. **KRISHNA KANTA MAITY** FOR their Cordial support, valuable suggestions and guidance. We extend our sincere. Thanks to our respected Head of the division Mr. **DEBAYAN BHATTACHARYA** for allowing us to use the facilities available. We would like to thank the other faculty members also, at this occasion. Last but not the least, we would like to thank our friends and family for the Support and encouragement they have given us during the course of our work.

SHUVOJIT CHOWDHURY
ROLL NO: D177055006-01039 AND REGISTRATION NO D161721682
RAHUL DHAR
ROLL NO: D177055006-01039 AND REGISTRATION NO D177055006
SAMIR CHANDRA PRAMANIK
ROLL NO: D177055006-01039 AND REGISTRATION NO D161721683
SOUVIK SEN
ROLL NO: D177055006-01039 AND REGISTRATION NO D177055006

Contents

Abstract vi

Chapter 1

1.1 Introduction 1

1.2 Background of project 2

Chapter 2

2.1 Software requirement specification 3

2.2 Advantages 4

2.3 Requirements 5

Chapter 3

3.1 Design diagrams 6

Chapter 4

4.1 Screenshot 7

4.2 Code 10

Chapter 5

5.1 Unit Testing 44

5.2 Integration Testing 44

Chapter 6

6.1 Conclusions 45

6.2 Future Work 46

6.3 References 47

Notes 48

ABSTRACT

An Artificial Intelligent Personal Assistant aims to make a conversation between both human and machine. The machine has been embedded knowledge to identify the sentences and making a decision itself as response to answer a question. The response principle is matching the input sentence from user. From input sentence, it will be scored to get the similarity of sentences, the higher score obtained the more similar of reference sentences. The sentence similarity calculation in this paper using bigram which divides input sentence as two letters of input sentence. The knowledge of Artificial Intelligent Personal Assistant are stored in the database. The Artificial Intelligent Personal Assistant consists of core and interface that is accessing that core in relational database management systems (RDBMS). The database has been employed as knowledge storage and interpreter has been employed as stored programs of function and procedure sets for pattern-matching requirement. The interface is standalone which has been built using programing language of Python.

CHAPTER 1

INTRODUCTION

This chapter gives an overview about the aim, objectives, background and operation environment of the system.

1.1 PROJECT AIMS AND OBJECTIVES

The project aims and objectives that will be achieved after completion of this project are discussed in this subchapter. The aims and objectives are as follows:

- Can Schedule meetings, Broadcast newsletters, auto-sequences
- Acquire leads from Comments
- Create conversational forms and saving all the data on spreadsheets
- You train them once and they will communicate with your target audience in their language.
- Multilingual chatbots have saved you from investing much on hiring different languages resources.
- If you are a company that's functional all around the world, you get hands-on chatbot asap! Because, while you are asleep, your bot can entertain your customers anywhere in the world.
- It is mainly used to provide customer support.
- It helps in creating a huge amount of target audience at the same time 24/7

1.2 BACKGROUND OF PROJECT

An Artificial Intelligent Personal Assistant is a computer program which conducts a conversation via auditory or textual methods. Such programs are often designed to convincingly simulate how a human would behave as a conversational partner, although as of 2019, they are far short of being able to pass the Turing test. It is typically used in dialog systems for various practical purposes including customer service or information acquisition. Some chatbots use sophisticated natural language processing systems, but many simpler ones scan for keywords within the input, then pull a reply with the most matching keywords, or the most similar wording pattern, from a database.

Artificial Intelligent Personal Assistant can be classified into usage categories such as conversational commerce (e-commerce via chat), analytics, communication, customer support, design, developer tools, education, entertainment, finance, food, games, health, HR, marketing, news, personal, productivity, shopping, social, sports, travel and utilities.

CHAPTER 2

SYSTEM ANALYSIS

In this chapter, we will discuss and analyses about the developing process of Artificial Intelligent Personal Assistant including software requirement specification (SRS) and comparison between existing and proposed system. The functional and non-functional requirements are included In SRS part to provide complete description and overview of system requirement before the developing process is carried out.
Besides that, existing vs. proposed provides a view of how the proposed system will be more efficient than the existing one

2.1 SOFTWARE REQUIREMENT SPECIFICATION

PRODUCT DESCRIPTION:

An Artificial Intelligent Personal Assistant is a computer program that simulates human conversation through voice commands or text chats or both. It is an Artificial Intelligence (AI) feature that can be embedded and used through any major messaging applications.

PROBLEM STATEMENT:

The problem occurred before having computerized system includes:

- Handling capacity.

Before Artificial Intelligent Personal Assistant the most common problem was handling a customer. A human can only handle one person at a time which is a very time consuming.

- Language Barrier.

The major problem of language barrier is a human cannot speak multiple language at a time also to understand at their own style.

- Cost.

Before the Artificial Intelligent Personal Assistant we have to set one group of people who can resolve customers query and we have to spend a huge amount for them.

2.2 ADVANTAGES

ADVANTAGES OF ARTIFICIAL INTELLIGENT PERSONAL ASSISTANT:

- Reduced Costs – Artificial Intelligent Personal Assistants eliminate the requirement of any manpower during online interaction and are hence seen as a big advantage by companies receiving multiple queries at once. This also presents companies with the opportunity to save on costs while aligning Artificial Intelligent Personal Assistant s with their goals and hence presenting customers with a particular type of interaction leading to conversion.
- 24-7 availability – Unlike humans, Artificial Intelligent Personal Assistant s once installed can attend queries at any time of the day. Thus, the customer doesn't have to wait for the company executive to help them. This also lets companies keep an eye on the traffic during the non-working hours and reach out to them later. On the other hand, while hiring people, there would be no access to these potential customers and could lead to loss of business.
- Learning and Updating – AI-based Personal Assistant s are capable of learning from interactions and updating themselves on their own. This is a big benefit when it comes to investing time in educating the executives about the same. Due to machine learning and algorithms capable of updating themselves, the need for same is eliminated while using a Artificial Intelligent Personal Assistant.
- Multiple Customer Handling – Humans have a limit to the number of clients they can handle at once. However, with Artificial Intelligent Personal Assistant s, there is no such constraint and they can handle as many queries as required at once. This is a major benefit of using Artificial Intelligent Personal Assistant s as no customer stays unattended and everyone's problem is being resolved. Developers are trying to come up with new features which can work on voice assisted services and help in guided sales. However, this is still expected to take some time, but will

be a major breakthrough in the Artificial Intelligent Personal Assistant and AI industry.

2.3 REQUIREMENTS

HARDWARE REQUIREMENTS:

- Intel XEON Server-Side processor is used as a processor because it is fast than other processors and provide reliable and stable and we can run our pc for longtime. By using this processor, we can keep on developing our project without any worries,
- Ram 16 gb is used as it will provide fast reading and writing capabilities and will in turn support in processing
- Internet Connectivity is required for getting a desirable output.

SOFTWARE REQUIREMENTS:

- HTML - (Hypertext Markup Language) is a text-based approach to describing how content contained within an HTML file is structured. This markup tells a web browser how to display the text, images and other forms of multimedia on a webpage. We have used HTML for designing all the webpages.
- CSS – (Cascading style sheets) are used to format the layout of Web pages. It can be used to define text styles, table sizes, and other aspects of Web pages that previously could only be defined in a page's HTML. We have used CSS for defining all the HTML files.
- JS - (JavaScript) is a scripting languages, primarily used on the Web. It is used to enhance HTML pages and is commonly found embedded in HTML code. JavaScript is an interpreted language. Thus, it doesn't need to be compiled. JavaScript renders web pages in an interactive and dynamic fashion. This allowing the pages to react to events, exhibit special effects, accept variable text, validate data, create cookies, detect a user’s browser, etc.

- Python 3 - is an interpreted, object-oriented programming language. Python is said to be relatively easy to learn and portable. We have used Python 3 as a back end development, Server Side development and AI mechanism.
-

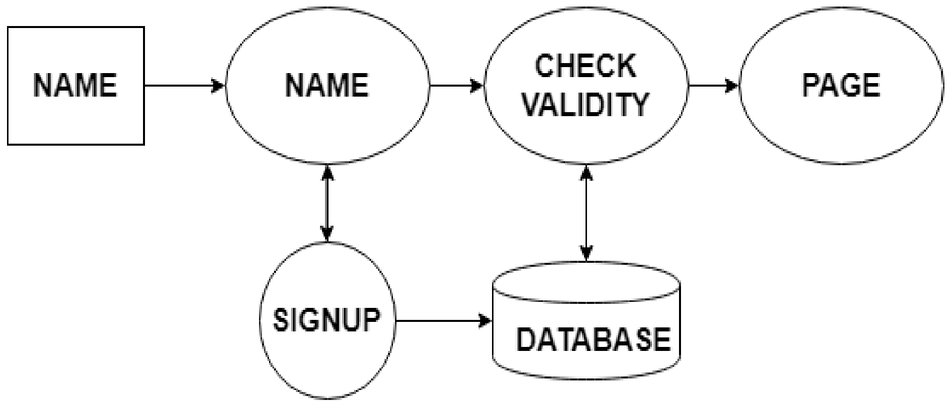
CHAPTER 3

DESIGN DIAGRAMS

TABLE DESIGN:

FIELD NAME	DATA TYPE	CONSTRAINTS	DESCRIPTION
User id	int(10)	Not Null	
Name	varchar(45)	Not Null	
Email id	varchar(10)	Not Null	
Date of Birth	varchar(45)	Not Null	
Gender	varchar(45)	Not Null	
Password	varchar(50)	Not Null	

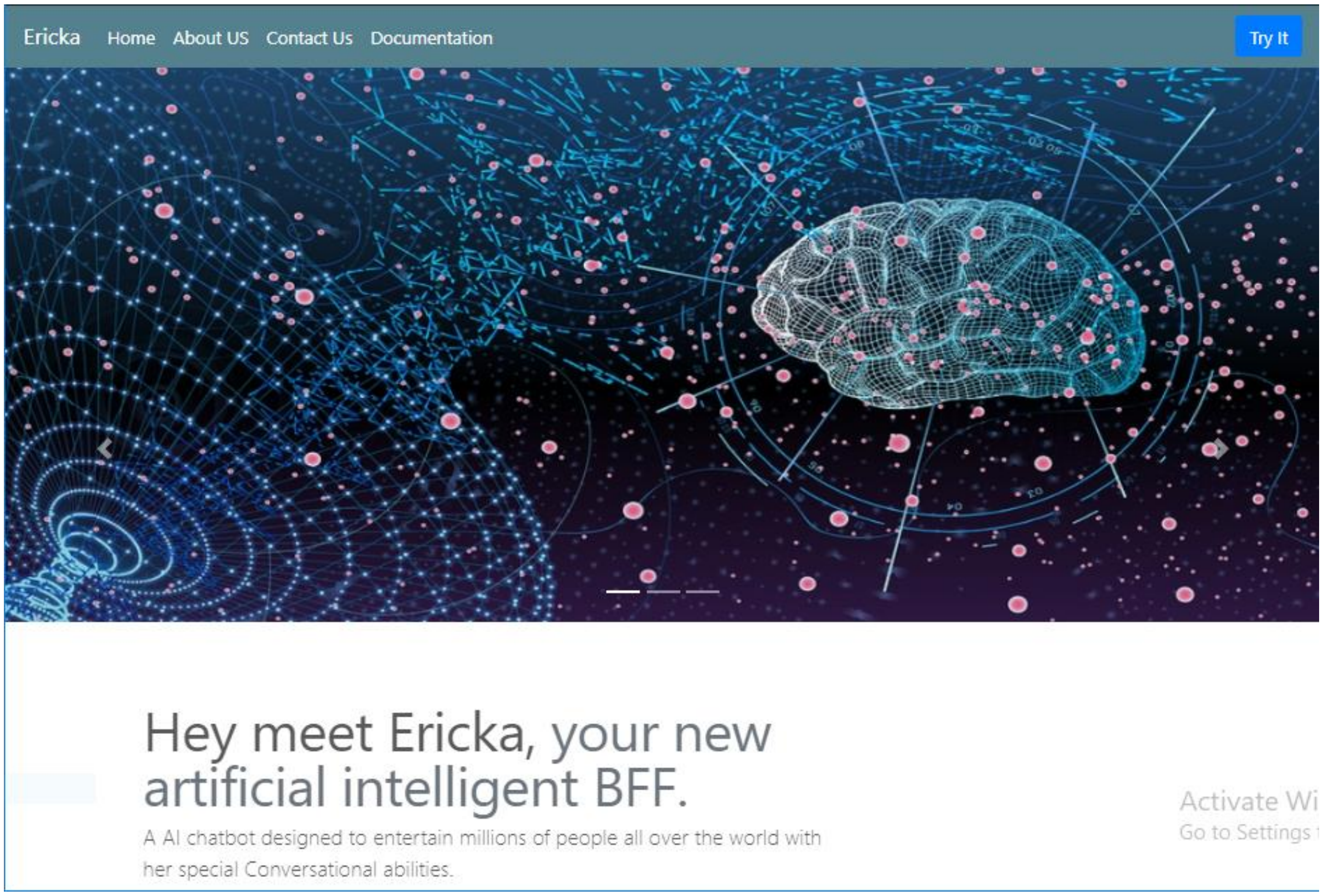
DATA FLOW DIAGRAMS :



CHAPTER 4
SYSTEM IMPLEMENTATION

4.1 SCREENSHOT

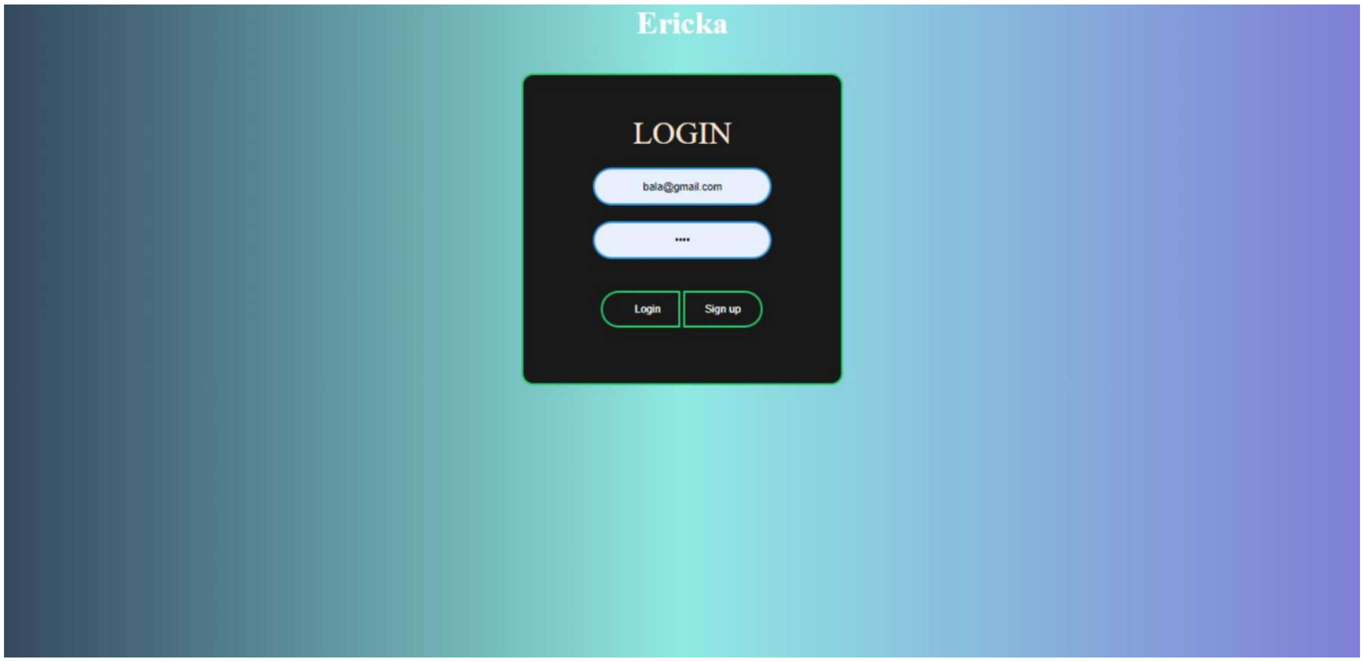
SCREENSHOT FOR HOMEPAGE:



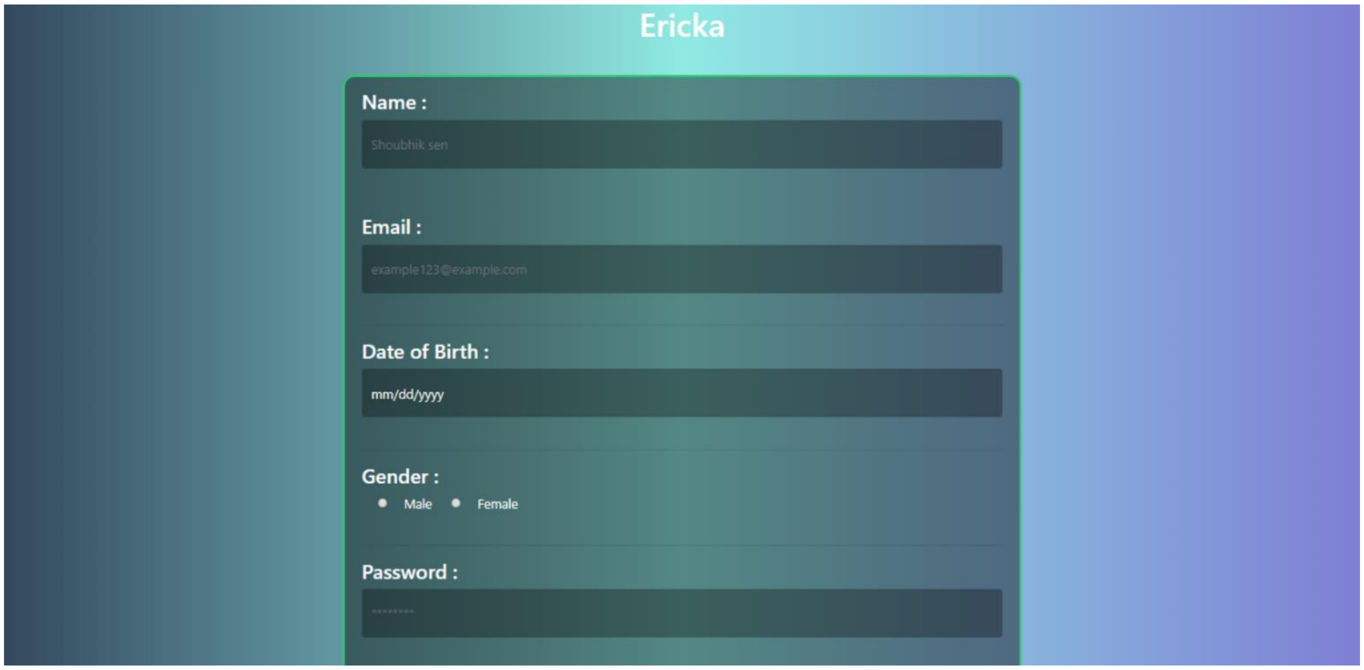
Hey meet Ericka, your new artificial intelligent BFF.

A AI chatbot designed to entertain millions of people all over the world with her special Conversational abilities.

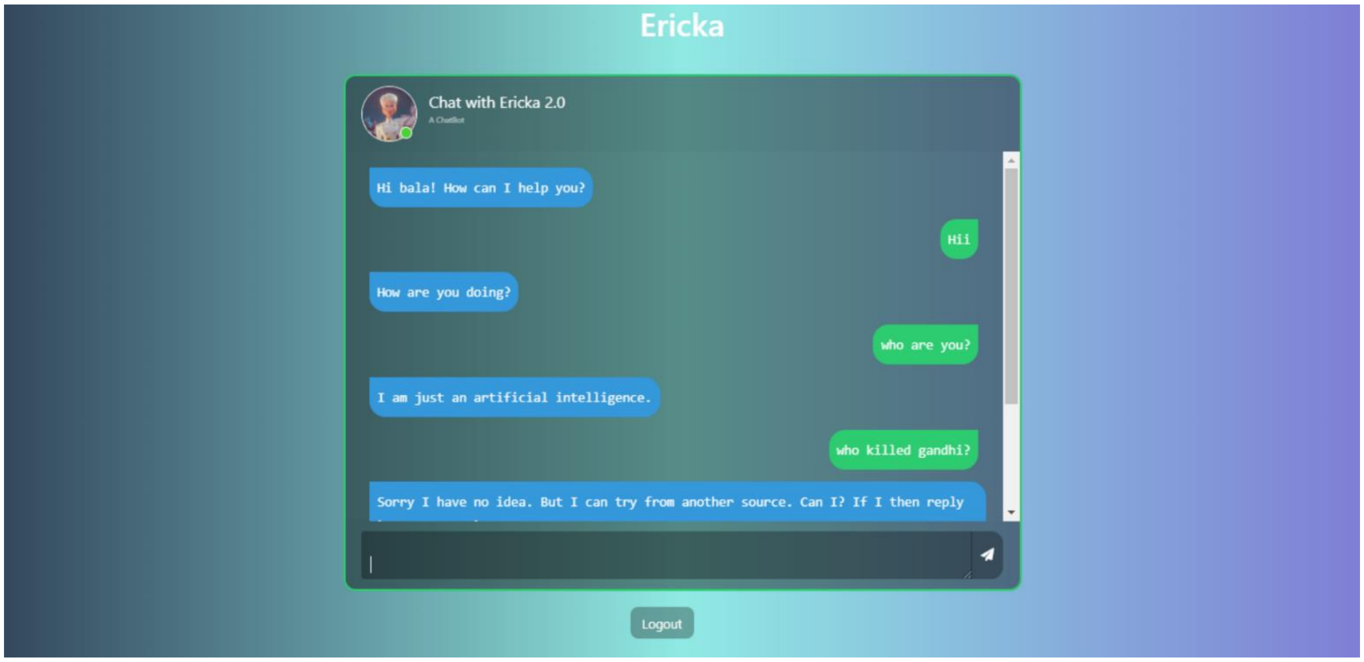
SCREENSHOT FOR LOGIN PAGE:



SCREENSHOT FOR SIGNUP PAGE:



SCREENSHOT FOR CHATTING PAGE:



Hi bala! How can I help you?

Hii

How are you doing?

who are you?

I am just an artificial intelligence.

who killed gandhi?

Sorry I have no idea. But I can try from another source. Can I? If I then reply

4.2 CODE

CODE FOR HOME PAGE:

```
<html>
        <head>
                <meta charset="utf-8">
           <meta name="viewport" content="width=device-width, initial-scale=1, shrink-
to-fit=no">
           <meta name="description" content="">
           <meta name="author" content="">
           <link rel="icon" href="./bootstrap-4.0.0/favicon.ico">

           <title>Ericka : ChatBot</title>

           <!-- Bootstrap core CSS -->
           <link href="{{ url_for('static', filename='css/bootstrap.min.css') }}"
rel="stylesheet">

           <!-- Custom styles for this template -->
           <link href="{{ url_for('static', filename='css/carousel.css') }}" rel="stylesheet">
        </head>
        <body>
        <!-- Navbar -->
   <header>
     <nav class="navbar navbar-expand-md navbar-dark fixed-top bg-dark">
      <a class="navbar-brand" href="#">Ericka</a>
```

```
    <button class="navbar-toggler" type="button" data-toggle="collapse" data-
target="#navbarCollapse" aria-controls="navbarCollapse" aria-expanded="false" aria-
label="Toggle navigation">
     <span class="navbar-toggler-icon"></span>
    </button>
    <div class="collapse navbar-collapse" id="navbarCollapse">
     <ul class="navbar-nav mr-auto">
      <li class="nav-item active">
       <a class="nav-link" href="#">Home <span class="sr-only">(current)</span></a>
      </li>
      <li class="nav-item active">
       <a class="nav-link" href="#aboutus">About US</a>
      </li>
      <li class="nav-item active">
       <a class="nav-link" href="#contactus">Contact Us</a>
      </li>
                  <li class="nav-item active">
       <a class="nav-link" href="#documentation">Documentation</a>
      </li>
     </ul>
              <form class="form-inline my-2 my-lg-0" action="{{ url_for('register') }}">
                     <button class="btn btn-primary my-2 my-sm-0" type="submit">Try
It</button>
              </form>
    </div>
   </nav>
  </header>
```

```
    <!-- Navbar Closed -->
<main role="main">
    <!-- Carousel -->
 <div id="myCarousel" class="carousel slide" data-ride="carousel">
  <ol class="carousel-indicators">
   <li data-target="#myCarousel" data-slide-to="0" class="active"></li>
   <li data-target="#myCarousel" data-slide-to="1"></li>
   <li data-target="#myCarousel" data-slide-to="2"></li>
  </ol>
  <div class="carousel-inner">
   <div class="carousel-item active">
    <img class="first-slide" src="static/img/1.jpg" alt="First slide">

   </div>

   <div class="carousel-item">
    <img class="second-slide" src="static/img/2.jpg" alt="Second slide">

   </div>

   <div class="carousel-item">
    <img class="third-slide" src="static/img/3.jpg" alt="Third slide">

    </div>
   </div>
  </div>
```

```
    <a class="carousel-control-prev" href="#myCarousel" role="button" data-
slide="prev">
      <span class="carousel-control-prev-icon" aria-hidden="true"></span>
      <span class="sr-only">Previous</span>
    </a>
    <a class="carousel-control-next" href="#myCarousel" role="button" data-
slide="next">
      <span class="carousel-control-next-icon" aria-hidden="true"></span>
      <span class="sr-only">Next</span>
    </a>
   </div>

                <!-- Carousel Closed -->
   <!-- Marketing messaging and featurettes
   ================================================== -->
   <!-- Wrap the rest of the page in another container to center all the content. -->

   <div class="container marketing">



    <!-- START THE FEATURETTES -->


    <div class="row featurette">
      <div class="col-md-8">
        <h2 class="featurette-heading" style="
  margin-top: 8px;">Hey meet Ericka, <span class="text-muted">your new artificial
intelligent BFF.</span></h2>
```

```
          <p class="lead">A AI chatbot designed to entertain millions of people all over the
world with her special Conversational abilities.<br/><br/>

                    No need to install any specific application weather you are on  Android
Ios or Windows a web based application which always there to entertain you.<br/><br/>
                    In every conversation she learn new things and gives the better
resolves.<br/><br/>

                    Had any Query, Ask Erica { show example (What is the forcast of Delhi)
}</p>
       </div>
     </div>
                <!-- Documentation  -->
     <hr id="documentation" class="featurette-divider">

                <div class="row featurette">
                        <div class="col-md-8">
                          <p class="lead">To download the documentation part click here,
                          <a href="ericka_documentation.pdf" download>Click Here</a> </p>
                        </div>
                </div>
                <!-- Documentation Closed -->

                <hr id="aboutus" class="featurette-divider">
     <!-- /END THE FEATURETTES -->

    </div><!-- /.container -->
```

```
            <!-- About Us  -->

       <div class="container marketing">



    <!-- START THE FEATURETTES -->

    <div class="row featurette" style="margin-top: 50px;" >
      <div class="col-md-7">
        <h2 class="featurette-heading">Shuvojit Chowdhury</h2>
        <p class="lead">I am the front-end developer of the project . I have designed all
the web-pages using Html, Css, BootStrap and JavaScript.</p>
      </div>
      <div class="col-md-5">
        <img class="featurette-image img-fluid mx-auto" src="static/img/shuvojit.jpg"
alt="Generic placeholder image">
      </div>
    </div>

    <hr class="featurette-divider">

    <div class="row featurette">
      <div class="col-md-7 order-md-2">
        <h2 class="featurette-heading">Rahul Dhar</h2>
```

```
          <p class="lead">I am the AI, back-end developer of the project . I have worked on
how the AI ChatBot will work efficiently. I have done all the back end development using
Flask and AI Chatbot using Python</p>
        </div>
        <div class="col-md-5 order-md-1">
          <img class="featurette-image img-fluid mx-auto" src="static/img/rahul.png"
alt="Generic placeholder image">
        </div>
      </div>

      <hr class="featurette-divider">

      <div class="row featurette">
        <div class="col-md-7">
          <h2 class="featurette-heading">Samir Chandra Pramanik</h2>
          <p class="lead">I am the content writer of the project . I have written all the
contents on the AI ChatBot's Website.</p>
        </div>
        <div class="col-md-5">
          <img class="featurette-image img-fluid mx-auto" src="static/img/samir.jpeg"
alt="Generic placeholder image">
        </div>
      </div>

        <hr class="featurette-divider">

        <div class="row featurette">
```

```
     <div class="col-md-7 order-md-2">
      <h2 class="featurette-heading">Souvik Sen</h2>
      <p class="lead">I am the conversation script writer of the project . I have trained
the Ericka ChatBot using YML format.</p>
     </div>
     <div class="col-md-5 order-md-1">
      <img class="featurette-image img-fluid mx-auto" src="static/img/souvik.png"
alt="Generic placeholder image">
     </div>
    </div>

    <hr id="contactus" class="featurette-divider">

    <!-- /END THE FEATURETTES -->

   </div><!-- /.container -->

           <!-- About Us Closed -->



           <!-- Contact Us -->
      <div  class="container marketing">

           <div class="container" style="margin-top: 40px;">
            <div class="col-md-6">
            <form action="/action_page.php">
```

```
                    <div class="form-group">
                      <label for="name">Name :-</label>
                      <input type="text" class="form-control" id="name"
placeholder="Enter name" name="Name">
                    </div>

                    <div class="form-group">
                      <label for="Address">Address:</label>
                      <input type="text" class="form-control" id="Address"
placeholder="Enter address" name="Address">
                    </div>

                    <div class="form-group">
                      <label for="phno">Phone Number :</label>
                      <input type="text" class="form-control" id="phno"
placeholder="Name phone number" name="Name">
                    </div>

                    <div class="form-group">
                      <label for="email">Email:</label>
                      <input type="email" class="form-control" id="email"
placeholder="Enter email" name="email">
                    </div>

                    <div class="form-group">
                      <label for="comment">Comment:</label>
```

```
                        <textarea class="form-control" rows="5" id="comment"
name="comment" id="comment" placeholder="Enter comments"></textarea>
                       </div>

                       <button type="submit" class="btn btn-primary">Submit</button>
                  </form>
                  </div>
                </div>
                <hr class="featurette-divider">
        </div>
                <!-- Contact Us Closed -->
      <!-- FOOTER -->



      <footer class="container">
        <p class="float-right"><a href="#">Back to top</a></p>
        <p>&copy; 2017-2018 Company, Inc. &middot; <a href="#">Privacy</a> &middot;
<a href="#">Terms</a></p>
      </footer>
    </main>

    <!-- Bootstrap core JavaScript
    ================================================== -->
    <!-- Placed at the end of the document so the pages load faster -->
    <script src="https://code.jquery.com/jquery-3.2.1.slim.min.js" integrity="sha384-
KJ3o2DKtIkvYIK3UENzmM7KCkRr/rE9/Qpg6aAZGJwFDMVNA/GpGFF93hXpG5KkN"
crossorigin="anonymous"></script>
```

```
   <script>window.jQuery || document.write('<script
src="../../../../assets/js/vendor/jquery-slim.min.js"><\/script>')</script>
   <script src="{{ url_for('static', filename='js/popper.min.js') }}"></script>
   <script src="{{ url_for('static', filename='js/bootstrap.min.js') }}"></script>
   <!-- Just to make our placeholder images work. Don't actually copy the next line! -->
   <script src="{{ url_for('static', filename='js/holder.min.js') }}"></script>


        </body>
</html>
```

CODE FOR LOGIN PAGE:

```
<!DOCTYPE html>
<html>
        <head>
                <title>Login</title>
                <link href="{{ url_for('static', filename='css/style.css') }}" rel="stylesheet">
        </head>
        <body>
                <h1>Ericka</h1>
                <h1><font color="red">{{ message }}</font></h1>
                <div class="login"  style="width: {{ wid }}">
                        <form action="/login" method="POST">
                                <h1>Login</h1>
                                <input type="email" name="email" placeholder="Email id">
                                <input  type="password" name="password"
placeholder="password">
                                <input class="li" type="submit" value="Login">
```

```
                    </form>
                    <a href="/signup"><input type="submit" class="sp" value="Sign
up"></a>
            </div>
      </body>
</html>
```

CODE FOR SIGNUP PAGE:

```
<!DOCTYPE html>
<html>
  <head>
    <title>signup</title>
    <link rel="stylesheet" type="text/css"
href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/4.7.0/css/font-
awesome.min.css">
    <link href="{{ url_for('static', filename='css/bootstrap.min.css') }}" rel="stylesheet">
    <link href="{{ url_for('static', filename='css/style.css') }}" rel="stylesheet">
  </head>
  <body>
    <h1>Ericka</h1>
    <div class="card" style="height:'auto'">
      <div class="card-footer">
        <form action="" method="POST">
          <h3>Name : </h3>
          <input type="text" name="name" class="form-control type_msg"
placeholder="Shoubhik sen"> <br><hr>
          <h3>Email : </h3>
          <input type="email" name="email" class="form-control type_msg"
placeholder="example123@example.com" value=""><br><hr>
          <h3>Date of Birth : </h3>
          <input type="date" name="dob" class="form-control type_msg"><br><hr>
          <h3>Gender : </h3>
          <div class='gender'>
            <input type="radio" name="Gender" value="Male" class="gender-
mf"><font color="white">Male</font>
            <input type="radio" name="Gender" value="Female" class="gender-
mf"><font color="white">Female</font>
          </div><br /><br/><hr>
          <h3>Password : </h3>
          <input type="password" name="password" class="form-control type_msg"
placeholder="********"><br /><hr />
          <input type="submit" value="Submit" class="submit_btn">
        </form>
      </div>
    </div>
  </body>
</html>
```

SCREENSHOT FOR CHATTING PAGE:

```
<!DOCTYPE html>
<html>
 <head>
  <title>Chat</title>
```

```
   <link rel="stylesheet" type="text/css"
href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/4.7.0/css/font-
awesome.min.css">
   <link rel="stylesheet" type="text/css" href="{{ url_for('static',
filename='css/bootstrap.min.css') }}">
   <link rel="stylesheet" type="text/css" href="{{ url_for('static', filename='css/style.css')
}}">

        <script src="{{ url_for('static', filename='js/bootstrap.min.js') }}"></script>
        <script src="{{ url_for('static', filename='js/jquery3.2.1.min.js') }}"></script>
  </head>
  <body>
   <h1>Ericka</h1>
   <div class="chat">
     <div class="card" style="height:500px">
       <div class="card-header msg_head">
                                    <div class="d-flex bd-highlight">
                                        <div class="img_cont">
                                            <img src="static/image.jpg"
class="rounded-circle user_img">
                                            <span
class="online_icon"></span>
                                        </div>
                                        <div class="user_info">
                                            <span>Chat with Ericka
2.0</span>
                                            <p>A ChatBot</p>
```

```
                                                  </div>
                                             </div>
                              </div>
      <div class="card-body msg_card_body">
        <div id="chatbox">
          <p class="botText"><span>Hi {{ session['username'] }}! How can I help
you?</span></p>
        </div>
      </div>
      <div id="userInput" class="card-footer">
          <div class="input-group">
              <textarea name="msg" class="form-control type_msg" id="textInput"
placeholder="Type your message..."></textarea>
              <div class="input-group-append">
                  <span class="input-group-text send_btn" id="buttonInput"><i class="fa fa-
send"></i></span>
              </div>
          </div>
      </div>
    </div>
  </div>
  <form action = "/logout">
    <button class="submit_btn" type="submit" style="left:45%">Logout</button>
  </form>
    <script>
      function getBotResponse() {
        var rawText = $("#textInput").val();
```

```
          var userHtml = '<p class="userText"><span>' + rawText + '</span></p>';
          $("#textInput").val("");
          $("#chatbox").append(userHtml);
          document.getElementById('userInput').scrollIntoView({block: 'end', behavior:
'smooth'});
          $.get("/get", { msg: rawText }).done(function(data) {
            var botHtml = '<p class="botText"><span>' + data + '</span></p>';
            $("#chatbox").append(botHtml);
            document.getElementById('userInput').scrollIntoView({block: 'end', behavior:
'smooth'});
          });
        }
        $("#textInput").keypress(function(e) {
            if(e.which == 13) {
                getBotResponse();
            }
        });
        $("#buttonInput").click(function() {
          getBotResponse();
        })
      </script>
    </div>
  </body>
</html>
```

CODE FOR BACK END DEVELOPMENT:

```
# import Python Library
from flask import Flask, session, escape, render_template, request,redirect, url_for
from chatterbot import ChatBot
from chatterbot.trainers import ChatterBotCorpusTrainer, ListTrainer
import mysql.connector
#import speech_recognition as sr
import requests
import re
import webbrowser
import os



#-------Flask App -------
app = Flask(__name__)

userInputs = [None]



#============= Fuctions for Various of Query =========

def u():
   return userInputs[len(userInputs)-2 ]



""" def voice(value):
   import pyttsx3
```

```
 def speak(value):
    engine = pyttsx3.init()
    voices = engine.getProperty('voices')
    engine.setProperty('voice', voices[1].id)
    rate = engine.getProperty('rate')
    engine.setProperty('rate', rate-30)
    engine.say(value)
    engine.runAndWait()

 speak(value) """

""" def myCommand():

 r = sr.Recognizer()

 with sr.Microphone() as source:
    r.pause_threshold = 1
    r.adjust_for_ambient_noise(source, duration=1)
    audio = r.listen(source)

 try:
    command = r.recognize_google(audio).lower()

 #loop back to continue to listen for commands if unrecognizable speech is received
 except sr.UnknownValueError:
    command = myCommand()
 return command """


#-------------- Google API ------------


def call(user):
 try:
    list_data = []


    query = user


    list_data.append(query)


    from googleapiclient.discovery import build


    # Google API Key
    my_api_key = "your-google-API-key"
    # Custome Search Engine Key
    my_cse_id = "your-cse-id"


    def google_search(search_term, api_key, cse_id, **kwargs):


        service = build("customsearch", "v1", developerKey=api_key)
        res = service.cse().list(q=search_term, cx=cse_id, **kwargs).execute()


        return res
```

```
      re = google_search(query, my_api_key, my_cse_id)

      ans = []

      for i in range(2):
         ans.append(f"Title : {re['items'][i]['title']}<br>Url : <a href='{re['items'][i]['link']}'
target='_blank' style='color:black'>{re['items'][i]['link']}</a><br>Description :
{re['items'][i]['snippet']}<br><br>")

      ans = ''.join(ans)

      list_data.append(ans)

      trBot = ChatBot("Erica")

      trBot.set_trainer(ListTrainer)
      trBot.train(list_data)

      return ans

   except:
      return "Nework not found!"


#------------Image Result ----------------------
def imgSearch(user):
   import requests, json
```

```
    query = user.replace(' ','+')

    response =
requests.get(f"https://www.googleapis.com/customsearch/v1?key=<google-image-
api>&cx=cse-id&q={user}")

    x = response.json()

    a = (x['items'][0]['pagemap']['cse_image'][0]['src'])

    return f"<img src={a} height='300' width='400'>"

#------------ Weather API (openweathermap.org) -------------
def weather(user):
    import requests, json
    import place

    try:
        def getWeather(city_name="Delhi"):

            response =
requests.get(f"http://api.openweathermap.org/data/2.5/weather?appid=<openweather
API>&q={city_name}")

            x = response.json()
            if x["cod"] != "404":
                return (f"City : {city_name}<br>Temp : {(x['main']['temp'])-
273.15}°C<br>Pressure : {x['main']['pressure']}<br>Humidity :
{x['main']['humidity']}%<br>weather : {x['weather'][0]['description']}")
            else:
                return (" City Not Found ")



        return getWeather((place.place(user))[0])

    except:
        return "Network not Found!!"



# ----------- Places(Hosoital, Restaurant, Temple etc.) in google map API -------------
def getPlace(user):
    import requests, json

    query = user.replace(' ','+')

    def rest(query):
        a = []

        response =
requests.get(f"https://maps.googleapis.com/maps/api/place/textsearch/json?query={qu
ery}&key=<google map api key>")
```

```
        x = response.json()

        if len(x['results']) < 5:
            for i in range(len(x['results'])):
                b = {'Name' : f"{x['results'][i]['name']}",
                'Address' : f"{x['results'][i]['formatted_address']}<br>"}
                a.append(b)
        else:
            for i in range(5):
                b = {'Name' : f"{x['results'][i]['name']}",
                'Address' : f"{x['results'][i]['formatted_address']}<br>"}
                a.append(b)

        return a

    return rest(query)



#------------- Google Map Direction Matrix API ------
def Dir(user):
    import requests, json
    import place

    dirB = f"{place.place(user)[0]}/{place.place(user)[1]}"
    r =
requests.get(f"https://maps.googleapis.com/maps/api/distancematrix/json?units=metric
&origins={place.place(user)[0]}&destinations={place.place(user)[1]}&key=AIzaSyAfQeZU
uGOH0_7YIzbmHX94aWVl4Xj363o")

    x = r.json()

    string = f"From : {''.join(x['origin_addresses'])}<br>To :
{''.join(x['destination_addresses'])}<br>Distance :
{x['rows'][0]['elements'][0]['distance']['text']}<br>Duration :
{x['rows'][0]['elements'][0]['duration']['text']}<br><a
href='https://www.google.com/maps/dir/{dirB}' target='_blank' style='color:black'>Click
to see in Google Map</a>"

    return string

# ------------ Open a Software that installed in 'C - Drive' by user command------
def openFile(user):
    import subprocess
    import os, fnmatch

    d = user.split(' ')
    c = d[len(d)-1]
    # This is to get the directory that the program
    # is currently running in.
    path = 'C:\Program Files (x86)'

    def find(pattern, path):
        result = []
        for root, dirs, files in os.walk(path):
            for name in files:
                if fnmatch.fnmatch(name, pattern):
                    result.append(os.path.join(root, name))
        return result

    try:
        subprocess.call(find(f'{c}.exe', path))
        return "<font color='green'>Your file is opend Successfully.</font>"
    except:
        return "<font color='red'>File not found!.</font>"

# ============== End of Functions =========



# ============= Database Connection =============
mydb = mysql.connector.connect(
    host = 'localhost',
    user = 'root',
    passwd = '',
    database = 'rist'
)

# -------- Object declare of MySql Database --------
cur = mydb.cursor()
```

```
# =========== ChatterBot Object Creation =========
Bot = ChatBot(
    "Ericka",
    preprocessors=[
        'chatterbot.preprocessors.clean_whitespace',
        'chatterbot.preprocessors.unescape_html',
        'chatterbot.preprocessors.convert_to_ascii'
    ],
    storage_adapter="chatterbot.storage.SQLStorageAdapter",
    logic_adapters=[
        "chatterbot.logic.MathematicalEvaluation",
        'chatterbot.logic.BestMatch',
        {
            'import_path': 'chatterbot.logic.LowConfidenceAdapter',
            'threshold': 0.65,
            'default_response': "Sorry I have no idea. But I can try from another source. Can I?
If I then reply `yes you can.`"
        }
    ]
)



# =========== Trainer =======================

"""for file in os.listdir('./data/'):
        Bot.set_trainer(ChatterBotCorpusTrainer)
        Bot.train('./data/'+file)
        print("Training completed")"""



# ============ Starting Main Web Application using Python Flask =============


# ---------- Redirect to main IP ----------
@app.route('/')
def Home(message=None):
   if message==None:
      return render_template('home.html')
   else:
      return render_template('register.html', message="signup succesfull")


@app.route('/logged_in')
def logged_in():
        if session['logged_in'] == False:
                return render_template('home.html')
        else:
                return render_template('index.html')


# ------------- Signup Page for new users ----------


@app.route('/signup')
def sign():
   return render_template('signup.html')


# -------------- Success Page ---------
```

```
@app.route('/success')
def success():
    return render_template('success.html', message = 'Signup Successful')



# ------------- Signup page with MySql ----------
# ------------- New user can register in Database -----

@app.route('/signup', methods=['POST'])
def signup():
    val = (request.form['name'], request.form['email'], request.form['dob'],
request.form['Gender'], request.form['password'])
    sql = "INSERT INTO `rist`.`register` (`Name`, `Email id`, `Date of Birth`, `Gender`,
`Password`) VALUES (%s, %s, %s, %s, %s)"

    cur.execute(sql, val)
    mydb.commit()

    return redirect(url_for('register'))




@app.route('/aboutus')
def aboutus():
    return render_template('aboutus.html')

@app.route('/contactus')
```

```
def contactus():
   return render_template('contactus.html')


@app.route('/register')
def register():
   return render_template('register.html')



# ------------ Main Bot Response Function ----------


@app.route("/get")
def get_bot_response():


        if session['logged_in'] == True:
                userText = request.args.get('msg')


                userInputs.append(userText)


                # use token values
                # Bot can give answer for GK or other questions That can't know bot
                if 'yes you can' in userText.lower() or 'yes u can' in userText.lower() or 'Yes'
in userText.lower():
                        return str(call(u()))


                # Bot Search restaurent near location via google map api
        elif 'restaurant' in userText.lower() or 'pizza hut' in userText.lower() or
'domino' in userText.lower() or 'kfc' in userText.lower() or 'McDonald' in
userText.lower():
            c = []
            a = (getPlace(userText))
            for i in a:
                for j,k in i.items():
                    c.append(f"{j} : {k}<br>")
            return str(''.join(c))

        # Bot Search Temple near location via google map api
        elif 'temple' in userText.lower() or 'mandir' in userText.lower() or 'mondir' in
userText.lower():
            c = []
            a = (getPlace(userText))
            for i in a:
                for j,k in i.items():
                    c.append(f"{j} : {k}<br>")
            return str(''.join(c))



        # Bot Search Shops near location via google map api
        elif 'shop' in userText.lower().lower() or 'shoping' in userText.lower():
            c = []
            a = (getPlace(userText))
            for i in a:
                for j,k in i.items():
                    c.append(f"{j} : {k}<br>")
            return str(''.join(c))



        # Bot Search Hospitals near location via google map api
        elif 'hospital' in userText.lower():
            c = []
            a = (getPlace(userText))
            for i in a:
                for j,k in i.items():
                    c.append(f"{j} : {k}<br>")
            return str(''.join(c))

        # Bot Search direction, Distance via google map api
        elif 'direction' in userText.lower() or 'distance' in userText.lower():
            return str(Dir(userText))



        # Weather Report API
        elif 'weather' in userText.lower() or 'temp' in userText.lower() or
'tempareture' in userText.lower():
            return str(weather(userText))

        elif 'image' in userText.lower() or 'img' in userText.lower() or 'photo' in
userText.lower() or 'picture' in userText.lower() or 'pic' in userText.lower():
            return str(imgSearch(userText))
        elif 'today' in userText.lower():
            userText = userText.replace("today", f"{x.strftime('%A')}")
            return str(Bot.get_response(userText))

        elif 'open website' in userText.lower():
            reg_ex = re.search('open website (.+)', userText.lower())
            if reg_ex:
                domain = reg_ex.group(1)
                url = 'https://www.' + domain
                webbrowser.open(url)
                return "Done!"
            else:
                pass

        else:
            s = str(Bot.get_response(userText))
            return s
    else:
        return render_template('register.html', wid = '400px')



# ---------- Render Template (register.html) or Login Page ------
@app.route('/login', methods=['POST'])
def Login():
    val = (request.form['email'], request.form['password'])
    sql = "select count(*) FROM `rist`.`register` WHERE `Email id` = %s AND `Password`
= %s"
```

```
        cur.execute(sql, val)
        res = cur.fetchone()

        if res[0] == 1:
                sql = "SELECT `Name`, `Email id`, `password` FROM `rist`.`register` WHERE
`Email id` = %s AND `Password` = %s"
                cur.execute(sql, val)
                res = cur.fetchone()

                if request.form['email'] == res[1] and request.form['password'] == res[2]:
                        session['logged_in'] = True
                        session['username'] = res[0]
                        return redirect(url_for('logged_in'))
                else:
                   return render_template('register.html', message = "Wroung user or
Password")
        else:
                return render_template('register.html', message = 'Wroung user or
Password')


@app.route('/logout')
def logout():
        session['logged_in'] = False
        return redirect(url_for('logged_in'))
```

```
app.secret_key = 'rdj'


# End of Flask
if __name__ == "__main__":
    app.run()
```

CHAPTER 5

TESTING

The aim of the system testing process was to determine all defects in our project. The program was subjected to a set of test inputs and various observations were [made and based on these observations it will be decided whether the program 'behaves as expected or not.

[Our Project went through two levels of testing]

1. Unit testing
2. Integration testing

5.1 UNIT TESTING

Unit testing is undertaken when a module has been created and successfully reviewed .In order to test a single module we need to provide a complete environment i.e. besides the module we would require.

- Tie procedures belonging to other modules that the module under test call.
- Non local data structures that module accesses
- A procedure to call the functions of the module under test with appropriate parameters

5.2 INTEGRATION TESTING

In this type of testing we test various integration of the project module by providing the input .The primary objective is to test the module interfaces in order to ensure that no errors are occurring when one module invokes the other module.

CHAPTER 6

POTENTIAL DIRECTION AND FURTHER DEVELOPMENT

6.1 CONCLUSIONS

From our perspective, Artificial Intelligent Personal Assistant with artificial intelligence are dramatically changing business. Artificial Intelligent Personal Assistant can reach out to a large audience and help of machine learning they can learn simultaneously and be more effective than humans.

It takes a lot of practice and a deeper understanding of underlying concepts to get the design right and build an Artificial Intelligent Personal Assistant that give users a great experience. The user should be able to get the job done by having a conversation with the bot without having to think too much and with a smile on their face.

6.2 FUTURE WORK

1. PLATFORM INDEPENDENT
   In future, Artificial Intelligent Personal Assistant will become platform independent will work on any Operating Systems and Devices.

2. FULLY VOICE RECOGNITION SYSTEM
   Artificial Intelligent Personal Assistant will become fully voice recognition system, means user will call Artificial Intelligent Personal Assistant by using the name and can ask his questions.

3. FULL PERSONAL ASSISTANT ECOSYSTEM
   We will further create an ecosystem, which will give information to user Q/A (Question and Answer) method.

4. HOME AUTOMATION SYTEM USING AI
   Home automation can be developed for which a user can control machines for their personal use.

6.3 REFERENCES

- GITHUB
- STACKOVERFLOW
- CHATTERBOT
- CHATTERBOT CORPUS
- GUNTHERCOX
- REDDIT
- W3SCHOOLS
- TUTORIASPOINT
- GOOGLE CLOUD SERVICES

NOTES