Phase-5: CREATE A CHATBOT USING PYTHON

Project Documentation and Submission

# Chatbot definition

A Chatbot is a computer program that is designed to simulate conversation with human users through text or voice interfaces. Chatbots can be used for a variety of purposes, such as customer support, information retrieval, and entertainment.

# Problem Definition

The problem is to build an AI powered diabetes prediction system that uses machine learning algorithms to analyse medical data and predict the likelihood of an individual developing diabetes. The system aims to provide early risk assessment and personalized preventive measures, allowing individuals to take proactive actions to manage their health.

# Functionality

1.Medication Reminders

2.Blood Sugar Tracking and Emergency Information

3.Dietary Assistance

4.Support and Motivation

5.Privacy and Data security

# User Interface

1. user onboarding and Simple and clean design

2.User-friendly navigation

3.Conversational interface

4.Voice and text input

5.Data visualization

## Responses

1. Plan a range of responses:

2. Accurate answers to medical questions

3. Personalized recommendations for prevention

4. Guidance on health monitoring

5. Empathetic and clear communication

### Integration

Decide integration details:

1. Seamless integration with the chosen platform

2. Minimal disruption to the user experience

3. Accessibility features for all users

## Testing and Improvement

Continuous testing and improvement plan:

1. User feedback collection

2. Regular updates to knowledge base and algorithms

3.Adherence to privacy and security standards.

# Conduct User Interface:

Begin by conducting user research to understand your target audience's needs, preferences, and pain points.

• Gather insights through,

Surveys: Create surveys to collect quantitative data about user preference and demographics.

Interviews: Conduct one to one or group interviews to gather qualitative insight from users.

Observation: Observe user behavior and interaction in relevant code

• Key Questions to answer,

What are the user's expectations for a chatbot?

What kind of guidance and assistance do users need?

What are the most common questions that users ask?

# Chatbot Objectives:

Chatterbot is a library in python which generates responses to user input.

It uses a number of machine learning algorithms to produce a variety of responses.

Chatbot are designed to give people an automated way to communicate with your company.

They may answer basic question, make product recommendations and provide customer support.

# Conversational design:

Once I have a good understanding of the user's needs, I will design the chatbot conversation flow.

Empathize: I will put myself in the shoes of the users and try to understand their needs and pain points.

Define: I will define the goals of the chatbot and users desired outcomes.

Ideate: I will generate a variety of ideas for how to design the chatbots conversation flow.

Prototype: I will create a prototype of the chatbots conversation flow and test it with users.

Test: I will test the chatbots conversation flow with users and make improvements based on their feedback.

# Natural Language Processing (NLP):

Natural language processing (NLP) is a branch of artificial intelligence(AI) that enables computers to comprehend, generate, and manipulate human language. Natural language processing has the ability to interrogate the data with natural language text or voice.

# Feedback Mechanism:

A **feedback mechanism** is a physiological regulation system in a living body that works to return the body to its normal internal state, or commonly known as [homeostasis](https://www.biologyonline.com/dictionary/homeostasis). In nature, feedback mechanisms can be found in a variety of environments and animal types. In a living system, the feedback mechanism takes the shape of a loop, which aids in maintaining homeostasis.

**Positive feedback mechanism examples**:

➛blood clotting

➛menstrual cycle

➛fruit ripening

**Negative feedback mechanism example:**

➛regulating blood glucose level

➛regulating temperature

# Testing and deployment:

Once the chatbot is implemented I will test the chatbot. During testing, the chatbot undergoes rigorous evaluation, including unit testing to validate individual components, functional testing to assess its performance in various user scenarios, and integration testing to confirm seamless communication with external systems.

# DATA PREPROCESSING:

Data preprocessing is required tasks for cleaning the data and making it suitable for a machine learning model which also increases the accuracy and efficiency of a machine learning model.

x= data\_set.iloc[:,:-1].values

y= data\_set.iloc[:,3].values

imputer= Imputer(missing\_values ='NaN', strategy='mean', axis = 0)

imputerimputer= imputer.fit(x[:, 1:3])

x[:, 1:3]= imputer.transform(x[:, 1:3])

label\_encoder\_x= LabelEncoder()

x[:, 0]= label\_encoder\_x.fit\_transform(x[:, 0]) = label\_encoder\_x.fit\_transform(x[:, 0])

labelencoder\_y= LabelEncoder()

y= labelencoder\_y.fit\_transform(y)

train\_test\_split

x\_train, x\_test, y\_train, y\_test= train\_test\_split(x, y, test\_size= 0.2, random\_state=0)

# Challenges:

Datasets often have missing values, which can adversely affect the performance of

machine learning models.

Machine learning algorithms typically work with numerical data, so categorical variables need to be transformed into a numerical format.

# Dataset used:

Dataset Link: <https://www.kaggle.com/datasets/grafstor/simple-dialogs-for-chatbot>

# Generating Chatbot Responses Using CSV File:

To generate output from a customer service chatbot using Flask and a CSV file for a website, I implemented the logic for the chatbot to respond to user messages. Here's how I generated chatbot responses based on user input and the CSV file:

from flask import Flask, render\_template, request, jsonify

import csv

import random

app = Flask(\_\_name\_\_)

# Load CSV data

with open('chatbot\_responses.csv', 'r') as csvfile:

reader = csv.reader(csvfile)

responses = [row[0] for row in reader]

# Define a function to get a random response from the CSV

def get\_response():

return random.choice(responses)

@app.route('/')

def index():

return render\_template('index.html')

@app.route('/chat', methods=['POST'])

def chat():

user\_message = request.form['user\_message']

chatbot\_response = get\_response()

return jsonify({'response': chatbot\_response})

if \_\_name\_\_ == '\_\_main\_\_':

app.run(debug=True)

In this the `/chat` route receives user messages via a POST request, and it generates chatbot responses using the `get\_response()` function, which selects a random response from the CSV file.

My HTML template for the chat interface (`index.html`) should include a form for users to input their messages, and JavaScript code to send the user's message to the `/chat` endpoint and display the chatbot's response.

Here's a code of an HTML template:

<!DOCTYPE html>

<html>

<head>

<title>Customer Service Chatbot</title>

</head>

<body>

<div id="chat-container">

<div id="chat-log"></div>

<input type="text" id="user-message" placeholder="Type a message...">

<button id="send-button">Send</button>

</div>

<script>

const chatLog = document.getElementById('chat-log');

const userMessage = document.getElementById('user-message');

const sendButton = document.getElementById('send-button');

sendButton.addEventListener('click', () => {

const message = userMessage.value;

userMessage.value = '';

chatLog.innerHTML += `<p><strong>You:</strong> ${message}</p>`;

fetch('/chat', {

method: 'POST',

body: new URLSearchParams({ user\_message: message }),

headers: { 'Content-Type': 'application/x-www-form-urlencoded' }

})

.then(response => response.json())

.then(data => {

chatLog.innerHTML += `<p><strong>Chatbot:</strong> ${data.response}</p>`;

});

});

</script>

</body>

</html>

With this setup, when a user enters a message and clicks "Send," the user's message is sent to the `/chat` endpoint, and the chatbot's response is displayed on the web page. The chatbot responses are randomly selected from the CSV file.

# Output for a Customer Service Chatbot:

Below is a chatbot conversation, assuming that the chatbot responses arerandomly selected from a CSV file, as mentioned in the previous code:

User: Hello, I need help with my account.

Chatbot: Sure, I can help you with that.

User: My password isn't working. Can you reset it?

Chatbot: I'm sorry to hear that. Let me assist you with resetting your password.

User: Thank you. How long does it usually take to reset the password?

Chatbot: Password resets usually take about 10-15 minutes to process.

User: Great, thanks for the information.

Chatbot: You're welcome! If you have any more questions, feel free to ask. User: What are your customer support hours?

Chatbot: Our customer support is available 24/7, so you can reach out to us anytime.

User: That's perfect! Thanks for your help.

Chatbot: No problem. If you have any other questions, just let me know.

This is a conversation between a user and a customer service chatbot. Since I have loadedand preprocessed my data and I have also implemented specific functions to train andmodel my chatbot in the previous phases, my chatbot is ready.

# CONCLUSION:

In conclusion, creating a chatbot is a dynamic and iterative process that involves a combination of technology, design, user experience, and data management. Awell implemented chatbot can provide significant benefits to both your organizationandyour users. It can enhance customer service, streamline interactions, and provide a convenient and efficient way for users to get the information they need.