Create Chatbot using Python

Phase 1: Problem Definition and Design Thinking

Problem Definition: The problem is to build an AI-powered diabetes prediction system that uses machine learning algorithms to analyze 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.

Design Thinking:

1. Functionality: Define the scope of the chatbot's abilities, including answering common questions, providing guidance, and directing users to appropriate resources.
2. User Interface: Determine where the chatbot will be integrated (website, app) and design a user-friendly interface for interactions.
3. Natural Language Processing (NLP): Implement NLP techniques to understand and process user input in a conversational manner.
4. Responses: Plan responses that the chatbot will offer, such as accurate answers, suggestions, and assistance.

5. Integration: Decide how the chatbot will be integrated with the website or app.
6. Testing and Improvement: Continuously test and refine the chatbot's performance based on user interactions.

Steps Involved:

1. Design: Develop a chatbot for website/apps to provide instant customer service.
2. Applicability: Enhance user experience by offering immediate assistance.
3. Technology: NLP and AI algorithms for understanding and responding to queries.
4. Coding: Python, NLP libraries, and possibly Tensorflow for advanced chatbots.
5. Architecture: An AI-Powered concierge guilding users through interactions.
6. Transformation: Transforms customer service into an always-available virtual assistant.
7. Real-World Analogy: A knowledgeable hotel concierge offerings guidance at any time.

INSTRUCTIONS:

Step 1: Download and Install Git

1. Visit the official Git website: https://git-scm.com/

2. Download the appropriate version of Git for your operating system (Windows, macOS, or Linux).

3. Run the installer and follow the on-screen instructions to complete the installation.

4. Open a terminal or command prompt and verify the installation by typing git––version.

Step 2: Download and Install Visual Studio Code

1. Go to the Visual Studio Code website: https://code.visualstudio.com/

2. Download the installer for your operating system (Windows, macOS, or Linux).

3. Run the installer and follow the installation prompts.

4. Launch Visual Studio Code.

Step 3: Create a GitHub Account

1. Open a web browser and go to https://github.com/

2. Click on the "Sign up" button.

3. Follow the registration process, providing your username, email address, and password.

4. Complete the verification process if prompted.

Step 4: Create a GitHub Repository

1. Log in to your GitHub account.

2. Click on your profile icon in the upper right corner and select "Your repositories" from the dropdown menu.

3. On the "Repositories" page, click the green "New" button.

4. Fill in the required information for your new repository, including the repository name, description, visibility, and other settings.

5. Optionally, you can choose to initialize the repository with a README file or add a .gitignore file for your specific project.

6. Click the green "Create repository" button to create your GitHub repository.

Step 5: Create a Local Folder

1. Minimize any open windows on your computer to see your desktop.

2. Right-click on an empty area of your desktop.

3. Hover over "New" in the context menu.

4. Click on "Folder" to create a new folder.

5. Give your folder a meaningful name, like "MyProject."

Step 6: Open the Folder in Visual Studio Code

1. Launch Visual Studio Code.

2. Click on "File" in the top-left corner.

3. Select "Open Folder" from the dropdown menu.

4. Browse to your desktop and select the folder you created in Step 5 (e.g., "MyProject").

5. Click the "Open" button to open the folder in Visual Studio Code.

Step 7: Clone Your GitHub Repository

1. In Visual Studio Code, open the integrated terminal by clicking on "View" in the top menu and selecting "Terminal" or using the keyboard shortcut (**Ctrl+** on Windows/Linux or **Cmd+** on macOS).

2. Use the **git clone** command to clone your GitHub repository by pasting the HTTPS URL of your repository.
Replace **repository_url** with the actual URL.

```
git clone <repository_url>
```

3. Navigate to the newly created repository folder using the cd command:

```
cd <repository_name>
```

Step 8: Check Git Status

1. To check the status of your local repository, enter the following command:

```
git status
```

Step 9: Modify the README File

1. Open the README file in your repository folder using Visual Studio Code.

2. Make the desired modifications to the README file.

Step 10: Check Git Status Again

1. Return to the terminal in Visual Studio Code.

2. Use the gitstatus command again to see the changes you made:

```
   git status
```

Step 11: Add Modifications to Staging Area

1. To stage your changes for a commit, use the gitadd command:

```
   git add README.md
```

Step 12: Commit Your Changes

1. Commit your staged changes with a descriptive message:

```
   git commit -m "Updated README file"
```

Step 13: Push Changes to GitHub

1. Push your committed changes to your GitHub repository:

```
   git push
```

Step 14: Create a New Branch

1. To create a new branch, use the gitbranch command followed by the desired branch name:

```
   git branch branch_name
```

Step 15: Switch to the New Branch

1. To switch to the newly created branch, use the gitcheckout command:

```
   git checkout branch_name
```

Step 16: Check Your Current Branch

1. To confirm the branch you're currently working on, use the gitbranch command:

```
  git branch
```

Report:

All the above instructions are installed and executed successfully.

Project by:

Name: P.muralidharan

Dept: CSE III YEAR

Reg No: 621421104037

College code: 6214

Group: IBM-Group 5