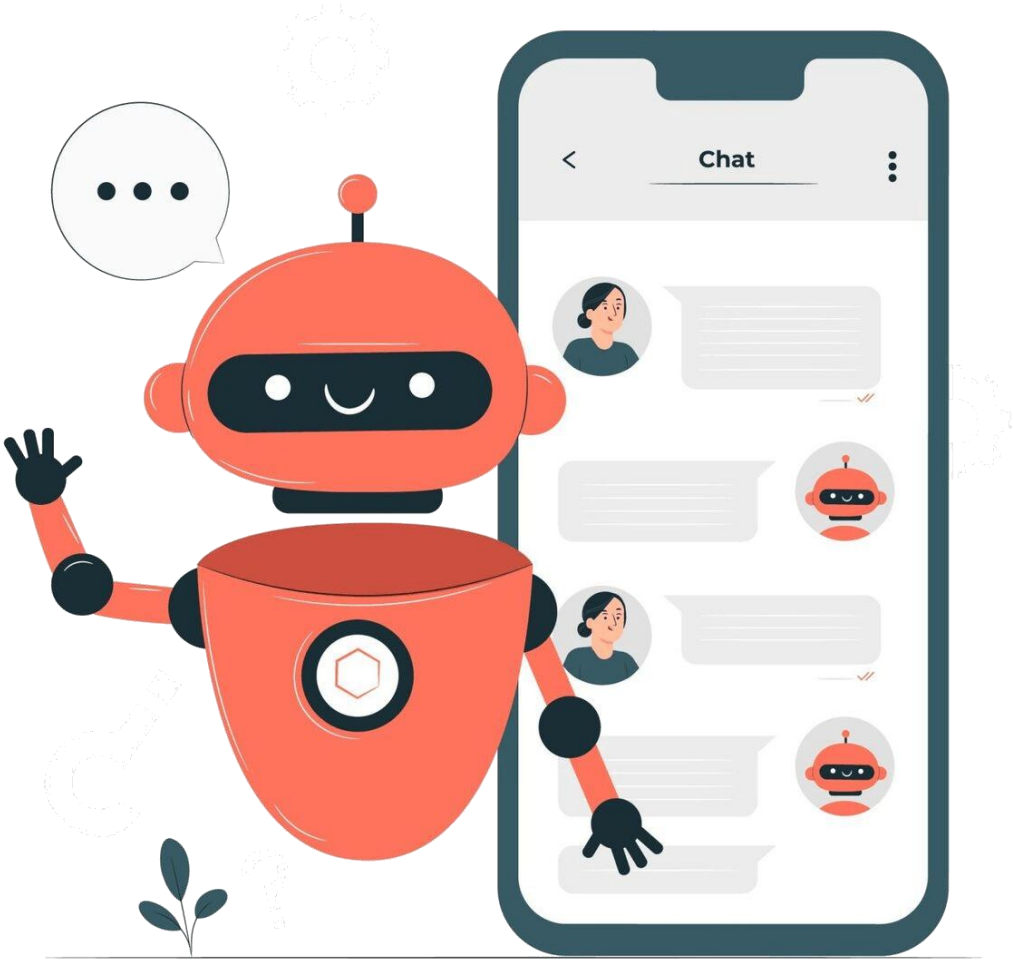
Principles of Artificial Intelligence Project

Title: A.I. Chatbot using Python



Name : Shourya
Roll No. : 00314811621
Branch : AI&ML
Sem : 3rd Semester

Synopsis

Objective

To make a speech to text chatbot that has multiple functionalities like similar A.I.s like Google Assistant, Siri and Alexa.

Requirements

- Python
- Code Editors (VS Code / Sublime Text)

Introduction

A chatbot is a software or computer program that simulates human conversation or "chatter" through text or voice interactions.
Users in both business-to-consumer (B2C) and business-to-business (B2B) environments increasingly use chatbot virtual assistants to handle simple tasks.
Adding chatbot assistants reduces overhead costs, uses support staff time better and enables organizations to provide customer service during hours when live agents aren't available.

How do chatbots work?

Chatbots have varying levels of complexity, being either stateless or stateful.
Stateless chatbots approach each conversation as if interacting with a new user. In contrast, stateful chatbots can review past interactions and frame new responses in context.
Adding a chatbot to a service or sales department requires low or no coding. Many chatbot service providers allow developers to build conversational user interfaces for third-party business applications.
A critical aspect of chatbot implementation is selecting the right natural language processing (NLP) engine. If the user interacts with the bot through voice, for example, then the chatbot requires a speech recognition engine.

Types of chatbots

As chatbots are still a relatively new business technology, debate surrounds how many different types of chatbots exist and what the industry should call them.

Some common types of chatbots include the following:
Scripted or quick reply chatbots. As the most basic chatbots, they act as a hierarchical decision tree. These bots interact with users through predefined questions that progress until the chatbot answers the user's question.

Similar to this bot is the **menu-based chatbot** that requires users to make selections from a predefined list, or menu, to provide the bot with a deeper understanding of what the customer needs.
Keyword recognition-based chatbots. These chatbots are a bit more complex; they attempt to listen to what the user types and respond accordingly using keywords from customer responses. This bot combines customizable keywords and AI to respond appropriately. Unfortunately, these chatbots struggle with repetitive keyword use or redundant questions.

Hybrid chatbots. These chatbots combine elements of menu-based and keyword recognition-based bots. Users can choose to have their questions answered directly or use the chatbot's menu to make selections if keyword recognition is ineffective.

Contextual chatbots. These chatbots are more complex than others and require a data-centric focus. They use AI and ML to remember user conversations and interactions, and use these memories to grow and improve over time. Instead of relying on keywords, these bots use what customers ask and how they ask it to provide answers and self-improve.

Voice-enabled chatbots. This type of chatbot is the future of this technology. Voice-enabled chatbots use spoken dialogue from users as input that prompts responses or creative tasks. Developers can create these chatbots using text-to-speech and voice recognition APIs. Examples include Amazon Alexa and Apple's Siri.

Methodology

In this project, I have used python to implement some functionality of “Soft” A.I. like opening applications, greeting the user, taking notes etc. I have used python libraries like pyttsx3 and built-in libraries like web browser. I used the following libraries in making the program:

Dependencies

- pyttsx3 - A python library that helps to convert text to speech. In short, it is a text to speech library. It uses the Microsoft Speech API for text to speech conversion.
- speechRecognition - A library with speech recognition functionality from different engines.
- wikipedia - A library which helps in searching Wikipedia entries of specific topics.

Built - In Libraries

- webbrowser - A built in module to access searching functionality.
- os - A built in module to access local files and directories.
- datetime - A built in module to get current date and time.

Functions

- speak(audio) - Allows the chatbot to speak. It takes audio as an argument and then speaks it out.
- pyttsx3.init('sapi5') - Initializes the Microsoft Speech API for speech recognition.
- getProperty('voices') - gets the male and female text to speech voice properties.
- setProperty('voice', voice[1].id) - sets the chatbot to female voice.
- wishme() - Wishes the user based on the time of the day.
- webbrowser.open('url') - Opens the url using a web browser.
- takeCommand() - Uses the speechRecogniser module to convert user audio input into text using the microphone as source.

Future Scope

The chatbot has immense potential for additional functionality. For example - Natural Language Processing can be used to make audio input more consistent. Mailing service can also be added at the expense of privacy settings, which is why I have refrained from using it in this project. Weather functionality can also be added using Weather APIs.