Overview:

The FreeWater Chatbot REST API is a web application that utilizes Python and Flask to provide responses to userqueries about the FreeWater company. It is designed to be easily accessigble and user-freindly. In addition, the Rest API includes a Discord bot that allows users to interact with the chatbot in a conversational manner.

The REST API is built on top of the Flask framework, which makes it easy to develop and deploy web applicaitons. The endpoints of the REST API are defined in main.py, which include the home page, an API endpoint for user input, and two API endpoints to run the Discord bot.

The Discord bot is implemented in bot.py, which includes the functions responsible for running and responding to user messages. The responses to user queries are generated using the GPTSimpleVectorIndex from the llama_index package, and the connection to the OpenAL API is established through the connect_AI function in responses.py

Overall, the FreeWater Chatbot REST API is versatile and easy-to-use web application that provides users with a simple way to get answers to their questions about the FreeWater company. The documentation should provide clear instauctions for setting up and running the REST API, as well as any necessary environment variables or dependencies.

Files:

Main.py:
This file is the main entry point for the REST API. It contains three API endpointsthat canbe accessed using HTTP GET request:

- /: The home page that displays a welcome message
- /api: An API endpoint that accepts user input and returns a response in JSON format.
- /demobot: API endpoints used to run a Discord bot with the default token
- /createbot: API endpoints used to run a Discord bot with a custom token thus allowing users to create their own bot.

Bot.py:
This file contains the functions responsible for running the Discord bot. It imports the necessary modules and packages, including discord and responses.

The run_discord_bot function is responsible for running the Discord bot. It takes an optional parameter, p_token, which is the bot token. If no token is provided, it retrieves the token from the DISCORD_TOKEN environment variable. The function connects to the OpenAI API using the connect_AI function in the responses module.

The send_message function is responsible for sending a response to the user's input message. It takes three parameters: message, user_message, and is_private. The message parameter represents the Discord message object that triggered the function call, user_message represents the user's input message, and is_private is a boolean value indicating whether the message should be private or public.

responses.py:
This file contains the get_response and connect_AI functions. The get_response function generates a response to the user's message using the GPTSimpleVectorIndex from the llama_index package. It takes a string message as input and returns a string response.

The connect_AI function connects to the OpenAI API by setting the OPENAI_API_KEY environment variable. It enters a while loop that keeps trying to connect to the API until successful. If the API key is not found, the function prompts the user to enter a valid key.

Overall, these files work together to provide a REST API and Discord bot that respond to user queries about the FreeWater company. The documentation should provide instructions for setting up and running the application, as well as any necessary environment variables or dependencies.

interview.txt:
The training data used to train the FreeWater chatbot is stored in the "context_data/data" directory. This data contains sample questions and responses that the chatbot can use to generate appropriate answers to user queries. Specifically, the data includes questions and responses related to FreeWater's business model.

The data has been organized and formatted in a way that can be easily read and processed by the GPTSimpleVecto

End Points Implementation:

The FreeWater Chatbot REST API provides three endpoints that can be accessed using HTTP GET requests. The public URL for this API is https://freewaterchatbotapi.onrender.com/.

Home page (/)

The home page endpoint displays a welcome message when accessed. It can be accessed by sending an HTTP GET request to the following URL:

https://freewaterchatbotapi.onrender.com/

Example usage:

```
import requests

response = requests.get('https://freewaterchatbotapi.onrender.com/')
print(response.text)
```

Output:

```
    <h1>Welcome to FreeWater chatbot.</h1>
```

API endpoint (/api)

The API endpoint accepts a user's input message and returns a response in JSON format. It can be accessed by sending an HTTP GET request to the following URL:
https://freewaterchatbotapi.onrender.com/api?input={user_input}

where {user_input} is the message input by the user.

Example usage:

```
    import requests

    response =
requests.get('https://freewaterchatbotapi.onrender.com/api?input=What is
FreeWater?')
    print(response.json())
```

Output:

```
    {
        "input": "What is FreeWater?",
        "response": "FreeWater is a company that provides clean and
accessible water to people in need around the world."
    }
```

Run Discord bot endpoint (/demobot and /createbot)

The /demobot endpoint runs the default Discord bot with the token stored in the DISCORD_TOKEN environment variable, while the /createbot endpoint runs a custom Discord bot with a specified token. They can be accessed by sending an HTTP GET request to the following URLs:

https://freewaterchatbotapi.onrender.com/demobot

https://freewaterchatbotapi.onrender.com/createbot?token={bot_token}

where {bot_token} is the token for the custom Discord bot.

Example usage:

```
    import requests

    # Run default bot
    response =
requests.get('https://freewaterchatbotapi.onrender.com/demobot')

    # Run custom bot
    response =
requests.get('https://freewaterchatbotapi.onrender.com/createbot?token={bot_tok
en}')
```

Note: The /demobot and /createbot endpoints are used to run the Discord bot and do not return any data.

Dependencies:

The following dependencies are required to run the FreeWater Chatbot REST API:

- Python 3.8.3 or higher
- Flask
- python-dotenv
- Discord.py
- llama_index
- langchain

Setup:

To set up the FreeWater Chatbot RESTAPI, follow these steps:

1. Clone the repository to your local machine.
2. Install the required dependencies using pip.
3. Set up environment variables for your OpenAI API key and Discord bot token. You can do this by creating a .env file in the root directory of the project and adding the following lines:
4. OPENAI_API_KEY=<your OpenAI API key>
5. DISCORD_TOKEN=<your Discord bot token>
6. Run main.py using the command "python main.py".
7. Access the API endpoints using HTTP GET requests.

Running the Discord Bot:

To run the Discord bot, use the /demobot API endpoint to run the bot with the default token, or use the /createbot API endpoint to run the bot with a custom token. Note that you must have a Discord bot token to run the bot.

To run the Discord bot, you will need a Discord bot token. To get your own token, follow these steps:

1. Go to the Discord Developer Portal website at https://discord.com/developers/applications.

2. Click on the "New Application" button in the top-right corner.
3. Enter a name for your application, then click "Create".
4. Click on the "Bot" tab in the left menu, then click "Add Bot".
5. Confirm that you want to add a bot to your application by clicking "Yes, do it!".
6. Under the "Token" section, click "Copy" to copy your bot's token to your clipboard.
7. Paste the token into a secure location, such as a password manager.

Once you have your token, you can run the bot using the /createbot API endpoint by passing the token as a query parameter. For example, if your token is "abcdefg", you can run the bot by visiting the URL https://freewaterchatbotapi.onrender.com/createbot?token=abcdefg. Alternatively, you can run the bot with the default token using the /demobot API endpoint by visiting the URL https://freewaterchatbotapi.onrender.com/demobot.

Conclusion:

In conclusion, the FreeWater Chatbot REST API is a versatile and powerful tool for users seeking information about the FreeWater company. The application provides a variety of API endpoints that allow users to easily access information through simple HTTP GET requests. In addition, the integration of a Discord bot allows users to interact with the application in a more engaging and interactive way. By combining these features, the FreeWater Chatbot REST API is able to reach a wider audience and provide a more convenient and accessible experience for users. With its ease of use, flexibility, and powerful functionality, the FreeWater Chatbot REST API is an excellent example of the power of modern web technologies and the potential they offer for improving communication and engagement between businesses and their customers.