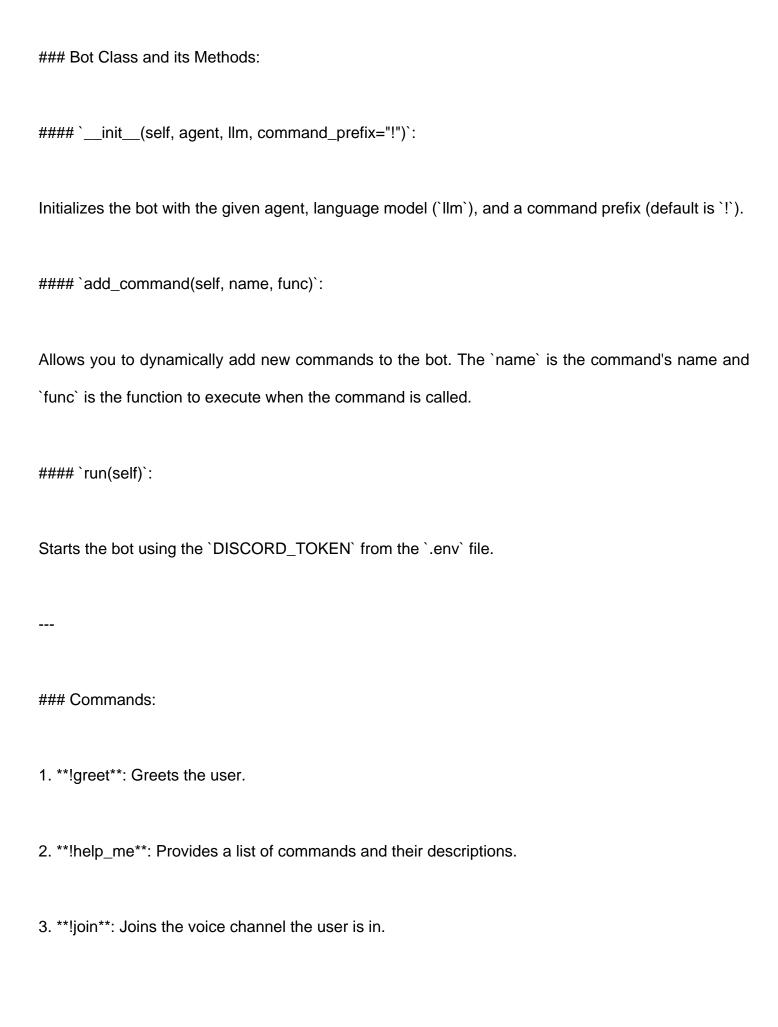
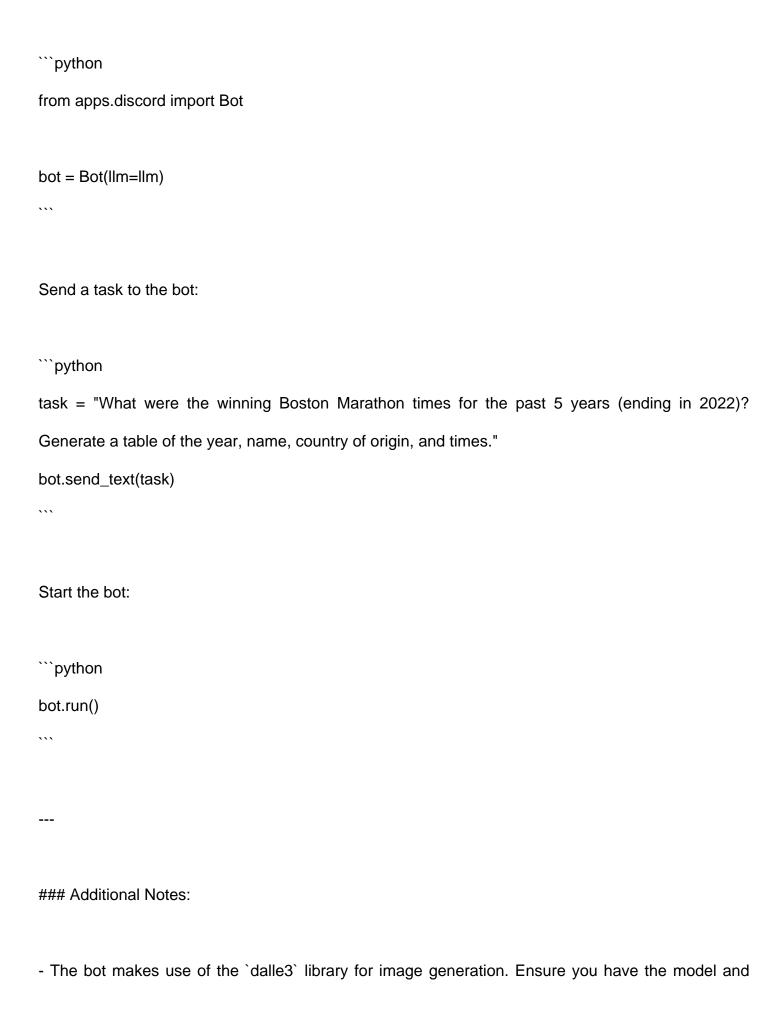
| ## Usage Documentation: Discord Bot with Advanced Features |
|---|
| |
| |
| |
| |
| ### Overview: |
| |
| This code provides a structure for a Discord bot with advanced features such as voice channel |
| interactions, image generation, and text-based interactions using OpenAI models. |
| |
| |
| |
| |
| ### Setup: |
| |
| 1. Ensure that the necessary libraries are installed: |
| ```bash |
| pip install discord.py python-dotenv dalle3 invoke openai |
| with the state of |
| |
| |
| 2. Create a `.env` file in the same directory as your bot script and add the following: |
| |
| DISCORD_TOKEN=your_discord_bot_token |
| STORAGE_SERVICE=your_storage_service_endpoint |
| SAVE_DIRECTORY=path_to_save_generated_images |
| |
| |
| |
| |



| 4. **!leave**: Leaves the voice channel the bot is currently in. |
|--|
| 5. **!listen**: Starts listening to voice in the current voice channel and records the audio. |
| 6. **!generate_image [prompt]**: Generates images based on the provided prompt using the DALL-E3 model. |
| 7. **!send_text [text] [use_agent=True]**: Sends the provided text to the worker (either the agent or the LLM) and returns the response. |
| |
| ### Usage: |
| Initialize the `Ilm` (Language Learning Model) with your OpenAl API key: |
| ```python from swarm_models import OpenAlChat |
| Ilm = OpenAlChat(|
| openai_api_key="Your_OpenAI_API_Key", temperature=0.5, |
|) |
| Initialize the bot with the `llm`: |



necessary setup for it.

- For the storage service, you might want to integrate with a cloud service like Google Cloud Storage or AWS S3 to store and retrieve generated images. The given code assumes a method `.upload()` for the storage service to upload files.
- Ensure that you've granted the bot necessary permissions on Discord, especially if you want to use voice channel features.
- Handle API keys and tokens securely. Avoid hardcoding them directly into your code. Use environment variables or secure secret management tools.