Enhancing ChatGPT Accessibility: Integrating with WhatsApp

By: Goodluck Caiser Malata.

gitHub Repo:

Introduction:

I recently encountered a minor challenge while attempting to upload an image prompt on ChatGPT. Despite having the image on my phone, I was accessing ChatGPT through my computer. Consequently, I had to transfer the image to my computer before uploading it to the ChatGPT interface. Similar inconveniences can arise when dealing with voice prompts. In such cases, having alternatives to eliminate these extra steps would be highly advantageous. What if ChatGPT could become more accessible through WhatsApp? The user-friendly interface of WhatsApp provides a clean platform for accessing various items. Integrating ChatGPT with WhatsApp could streamline the process and significantly enhance accessibility.

Challenges with Media Uploads to ChatGPT:

Uploading images or voice prompts can become cumbersome, especially when users access the model from one location while the media resides elsewhere. Suddenly, what seems like a straightforward task turns into a challenge as users realize they need to transfer the image or voice prompt from their phone to their computer before uploading it to ChatGPT.

In light of these challenges, there is a clear need for a more accessible solution that streamlines the process of interacting with ChatGPT, particularly when it comes to media uploads. Users should be able to seamlessly integrate their preferred media formats, whether images or voice prompts, into their conversations with ChatGPT without being encumbered by the limitations of device compatibility or file transfer procedures.

Integrating ChatGPT with user-friendly platforms like WhatsApp presents an opportunity to address these accessibility issues effectively. By leveraging the intuitive interface and widespread adoption of WhatsApp, users can interact with ChatGPT seamlessly, regardless of the device they are using or the media format they wish to employ as a prompt.

Nevertheless, If accessibility issues primarily arise from accessing ChatGPT from different platforms while media remains on another device, users may indeed question the necessity of integrating with WhatsApp instead of accessing ChatGPT directly from their phones.

To address this concern and emphasize the benefits of integrating ChatGPT with WhatsApp, it's essential to highlight specific advantages that the integration offers beyond accessing GPT directly from a phone. These advantages could include:

1. Unified Communication Platform: Integrating ChatGPT with WhatsApp provides users with a centralized platform for communication and prompt generation. Instead of switching between multiple apps or devices, users can seamlessly interact with ChatGPT within the familiar WhatsApp environment.

2. Enhanced Media Integration: WhatsApp offers robust media integration capabilities, allowing users to upload images, voice messages, and text prompts effortlessly. By leveraging WhatsApp's features, users can initiate diverse conversations with ChatGPT using various media formats, enhancing the richness and depth of interaction.

3. Real-time Collaboration: WhatsApp facilitates real-time collaboration and communication among users. By integrating ChatGPT with WhatsApp, users can easily share generated responses and collaborate on prompt creation in group chats or private conversations, fostering a collaborative and engaging environment.

4. Cross-Platform Compatibility: Integrating ChatGPT with WhatsApp ensures cross-platform compatibility, enabling users to access the AI model seamlessly from different devices and operating systems. Whether users prefer accessing WhatsApp from their smartphones, tablets, or computers, they can engage with ChatGPT consistently across all platforms.

In summary, integrating ChatGPT with WhatsApp offers a promising solution to streamline media uploads and enhance accessibility. By leveraging WhatsApp's user-friendly platform, users can enjoy a unified communication experience, seamless media integration, real-time collaboration, and cross-platform compatibility. This integration not only simplifies the interaction with ChatGPT but also opens up new possibilities for diverse and engaging conversations, marking a significant step towards a more efficient and user-centric AI experience.

Technical Implementation:

Integrating ChatGPT with WhatsApp involves several technical steps to enable seamless communication between the AI model and the messaging platform. Here's an overview of the key technical aspects and potential APIs or platforms that can facilitate the integration:

1. Selecting a Messaging Platform:Before integrating ChatGPT with WhatsApp, it's essential to choose a messaging platform or provider that offers robust APIs and tools for building chatbots and conversational interfaces. Platforms like Gupshup, Clickatell, Twilio, and WhatsApp Business API provide comprehensive solutions for integrating chatbots with WhatsApp.

2. WhatsApp Business API Setup:If using the WhatsApp Business API, the first step is to set up a WhatsApp Business account and apply for access to the API. Once approved, developers can obtain API credentials and configure webhook endpoints to receive incoming messages and events from WhatsApp users(But in our case we will use Gupshup API).

3. Creating Webhooks and Endpoints:Developers need to create webhook endpoints on their server infrastructure to handle incoming messages and events from WhatsApp. These endpoints act as bridges between WhatsApp and the ChatGPT backend, facilitating message processing and response generation.

4. Implementing Message Processing Logic:Upon receiving messages from WhatsApp users, developers need to implement message processing logic to extract user queries, prompts, or commands. This logic involves parsing incoming messages, extracting relevant information, and formulating appropriate requests to the ChatGPT backend for response generation.

5. Integrating with ChatGPT: Once user messages are processed, developers can integrate with the ChatGPT to generate responses based on user inputs. This integration typically involves making API calls to the ChatGPT service, passing user queries as inputs, and retrieving generated responses or predictions.

6. Testing and Deployment:Before deploying the integration to production, thorough testing is necessary to validate functionality, performance, and reliability. Developers should conduct unit tests, integration tests, and end-to-end testing to identify and address any potential issues or bugs. Once testing is complete, the integration can be deployed to production environments for live usage by WhatsApp users.

By following these technical steps and leveraging robust APIs or platforms like Gupshup or Clickatell, developers can seamlessly integrate ChatGPT with WhatsApp, enabling users to engage in meaningful conversations and interactions with the AI model directly from the messaging platform.

Technical Implementation:

Prerequisites:

Note: This blog assumes the use of Ubuntu.

1. Node JS

Ensure that Node.js is installed on your system. If not, you can install it directly from the Ubuntu terminal. To begin, update and upgrade the package index if you haven't done so recently:

- *sudo apt update*
- *sudo apt upgrade*

Then, proceed to install Node.js:

- *sudo apt install nodejs*

It's also beneficial to install npm separately:

- *sudo apt install npm*

Once installed, you can verify the installation with:

- *node -v*
- *npm -v*

If Node.js is installed, you can proceed with the installation of the code editor or move to the next section.

2. Code Editor:
   Text editors facilitate writing and executing code. While there are various options available, this guide will focus on installing Microsoft Visual Studio Code on Ubuntu 20. To begin, ensure that you've updated and upgraded the package index as mentioned during the installation of Node.js.

   Next, install necessary packages before installing Visual Studio Code:

   - *sudo apt install software-properties-common apt-transport-https wget -y*

   Furthermore, import Microsoft's GPG key to your system using the following command:

   - *wget -O- https://packages.microsoft.com/keys/microsoft.asc | sudo gpg --dearmor | sudo tee /usr/share/keyrings/vscode.gpg*

   Add the VS Code repository to your Ubuntu system:
   - *sudo add-apt-repository "deb [arch=amd64] https://packages.microsoft.com/repos/vscode stable main"*

   Then, update the package index and install Visual Studio Code:

   - *sudo apt update*
   - *sudo apt install code*

   Finally, launch Microsoft Visual Studio Code from the terminal with the following command, or find its icon in the application menu:

   - *code .*

For those using window or mac os visit the official node documentation or microsoft visual studio code to install these packages

Steps to Integration:

1. Setting up a Node server for handling WhatsApp interactions.
   To set up node server cd into the project directory and then initialize node
2. Configuring callback URLs and endpoints for communication between ChatGPT and WhatsApp.
3. Hosting considerations and the use of tools like NGROK for local development.
4. Integrating ChatGPT APIs with the Node server to enable seamless communication.

Setting Up A Node Server For Handling Whatsapp Interactions:

1. Create a new project
2. Initialize npm on the new project:
3. Install express and spin up a server on port of choice
    - *mkdir projectName*
    - *cd projectName*
    - *npm init*

    You would have created a new project and initialized Node. Now to spin up a server we need to install express on the project and spin up the server from localhost.
    - *npm install express*

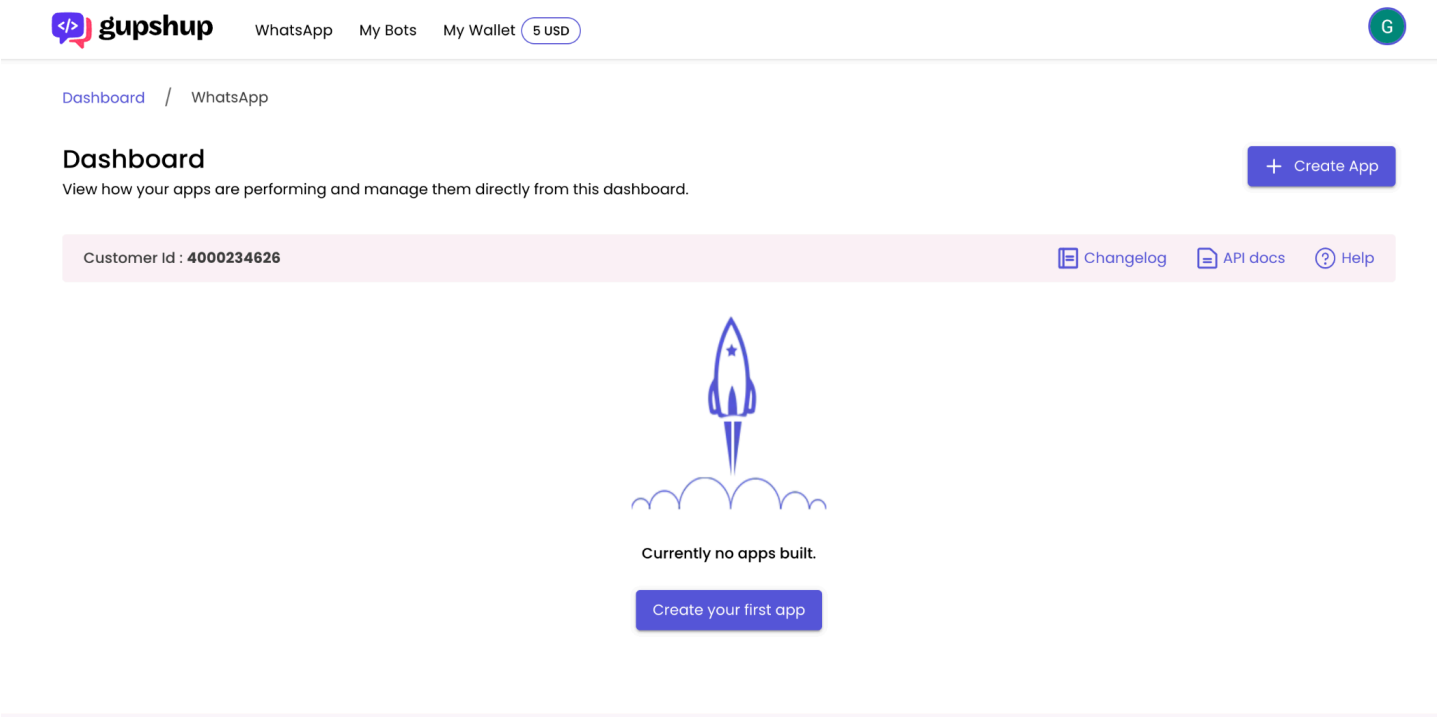
    Please find the screenshot for implementation of the server.

With this we have a server running and we just need to integrate whatApp and GPT through their APIs.

Integrating Node server with WhatsApp using Gupshup API:

1. Create an app on Gupshup:
    - *Sign in to your account or create one if you're not registered.*
    - *Create a new app and make note of the app name.*

Image showing the landing page for creating new apps with gupshup



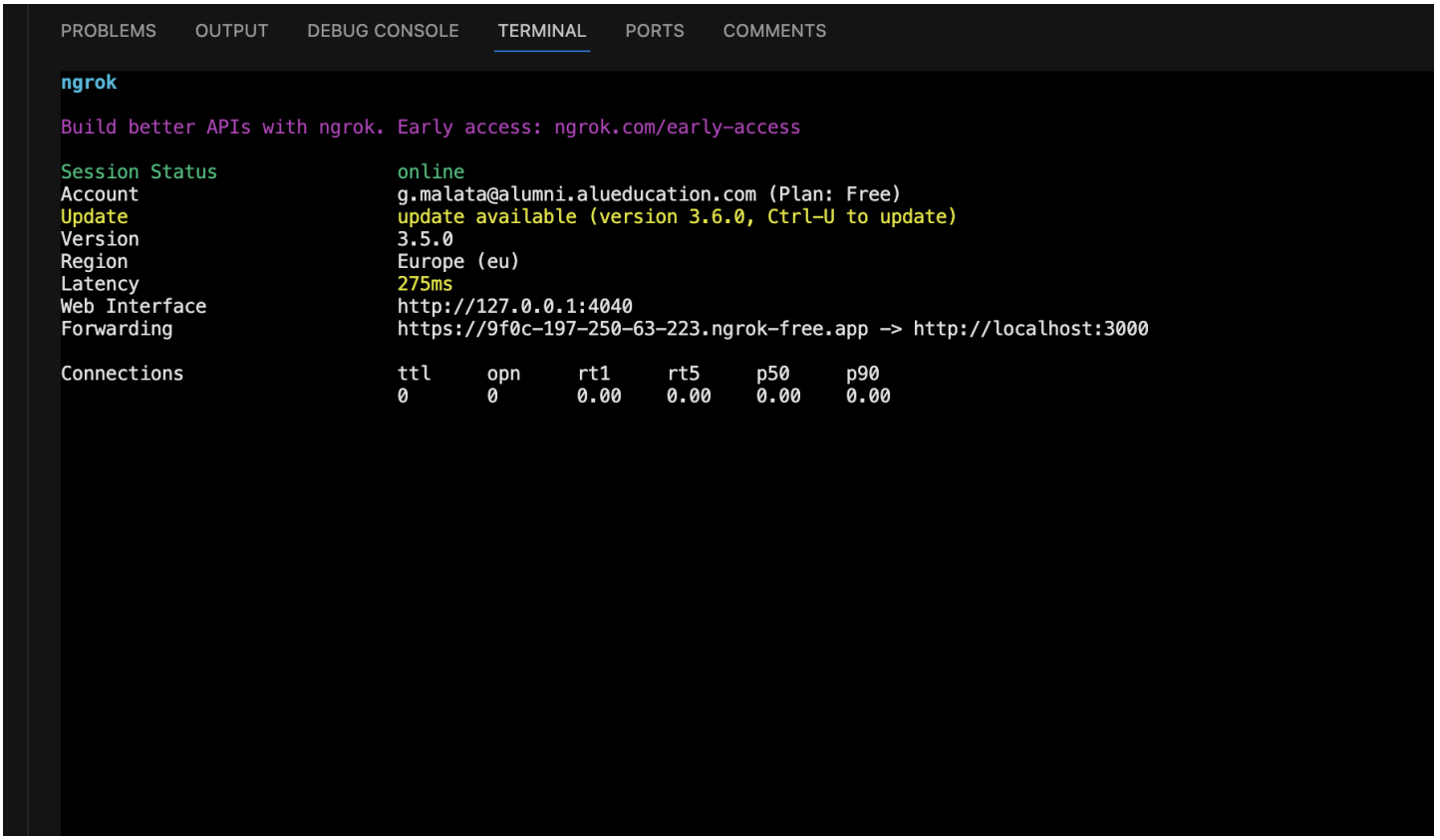
2. Configure a callback URL:
    - Create an endpoint that will connect to Gupshup. Endpoints are paths within a web application that lead to specific operations. In our case, our endpoint will send and receive data from Gupshup, which refers to the chat messages. Since we act as middlemen, we'll receive and forward chats.

Image below shows the implementation of the server together with the newly added endpoint.

```
JS index.js > ...
1   const express = require('express');
2
3   const app = express();
4
5
6   app.use(express.json());
7
8   app.post('/gupshup', async (req, res) => {
9       try {
10          // catching whatsApp user's input whether image, voice or text
11          const prompt = req.body.messageText;
12
13      } catch (error) {
14          console.error("Error processing request:", error);
15          res.status(500).json({ error: "Internal server error" });
16      }
17  });
18
19  app.listen(3000, () => {
20      console.log('App listening on port 3000');
21  });
22
```

3. Hosting considerations:
    - As your application is hosted locally, HTTP requests cannot be forwarded to applications running locally. Instead, you can use NGROK. NGROK allows you to make your application, running on any port, accessible via the internet. Learn more about NGROK [here].
    - To host with NGROK access terminal window and type ngrok and the port in which your application is running

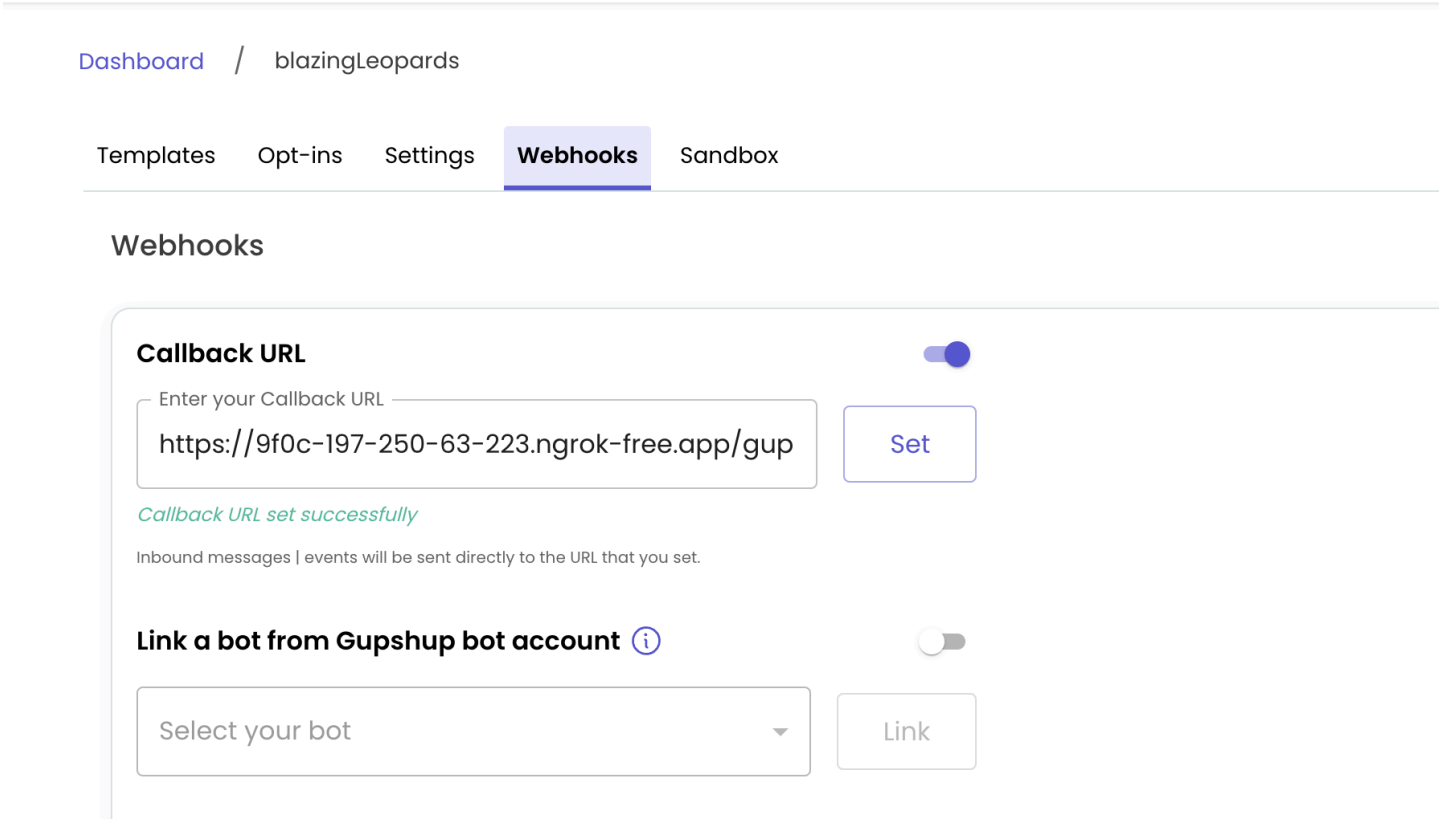
    - *ngrok http 3000*



4. Register the endpoint on the Gupshup dashboard.

   Now that our Node server is accessible on the internet, capable of both receiving and sending data, it's time to register this endpoint with Gupshup. This involves informing Gupshup that we want all traffic to be directed to this specific address.

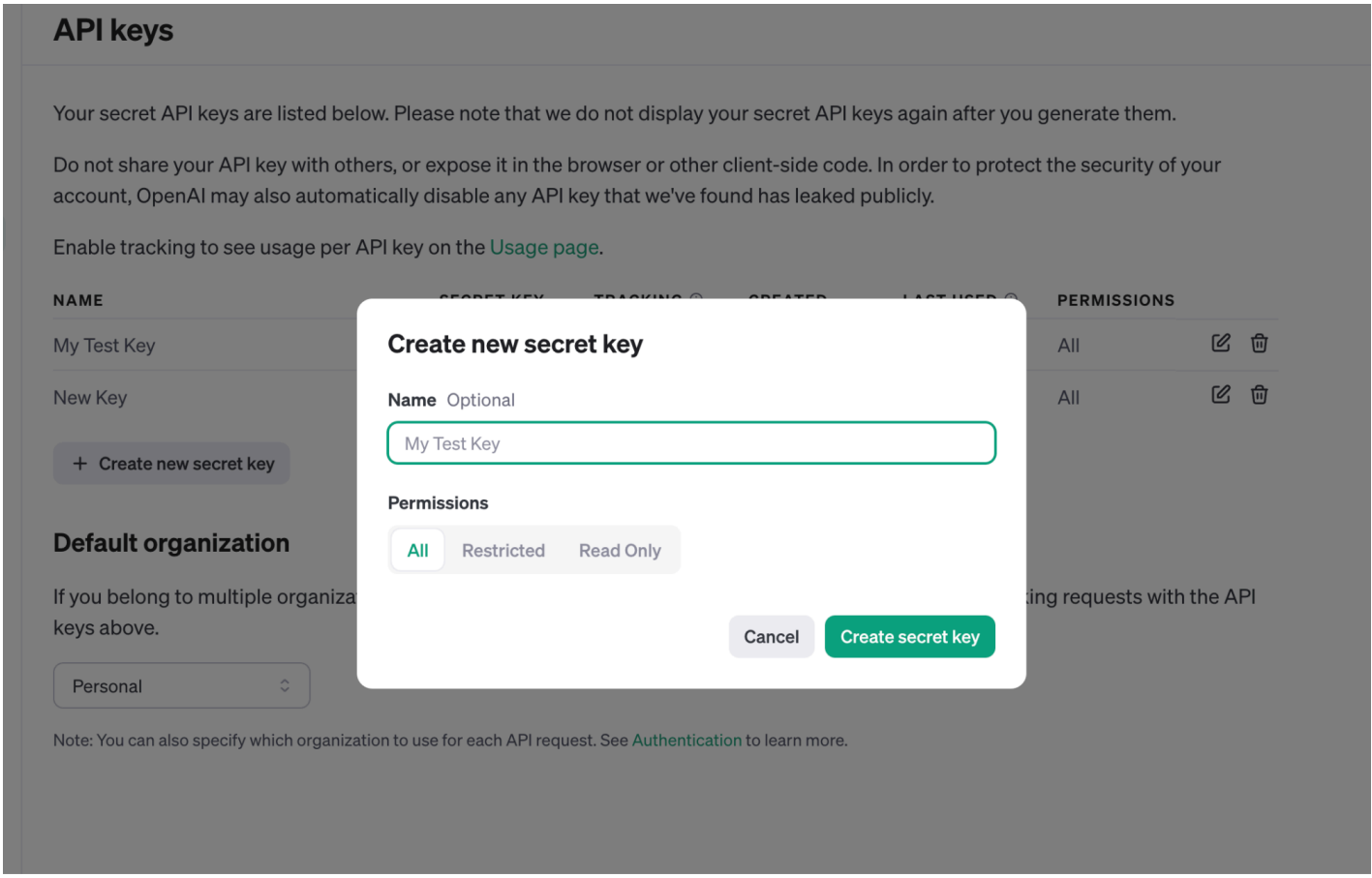
   - To set up a call back on the gupshup portal Navigate to the webhook section then enter the call back URL (for our case NGROK URL)



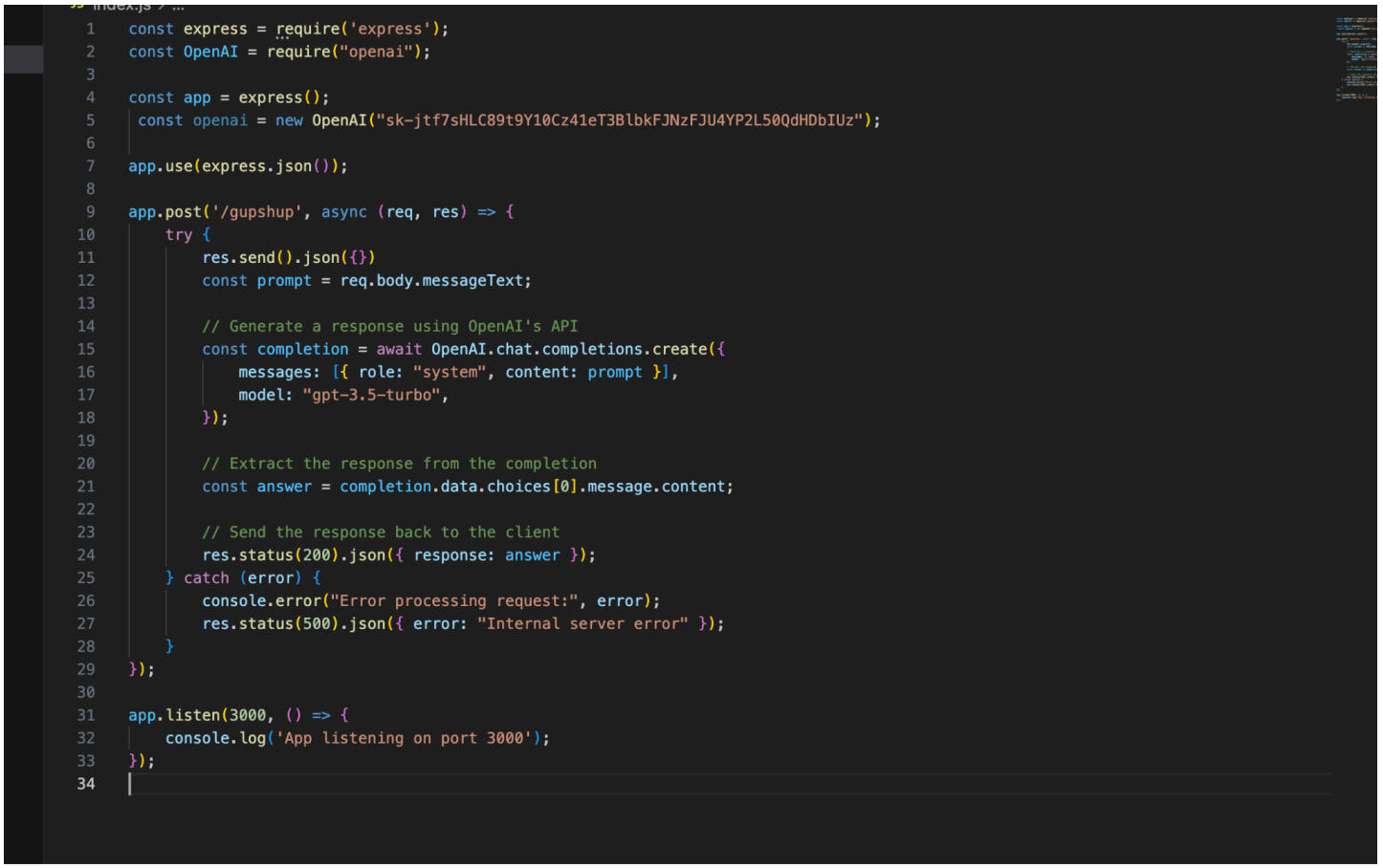
5. Go Live With WhatsApp Number.
   For now we skip this to later stages.

Integrating With GPT Api:

1. Install the open AI node js library
2. Setting up API key



3. Implementing logic to forward prompts from Gupshup to GPT and receiving the responses.



```
const express = require('express');
const OpenAI = require("openai");

const app = express();
 const openai = new OpenAI("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");

app.use(express.json());

app.post('/gupshup', async (req, res) => {
    try {
        res.send().json({})
        const prompt = req.body.messageText;

        // Generate a response using OpenAI's API
        const completion = await OpenAI.chat.completions.create({
            messages: [{ role: "system", content: prompt }],
            model: "gpt-3.5-turbo",
        });

        // Extract the response from the completion
        const answer = completion.data.choices[0].message.content;

        // Send the response back to the client
        res.status(200).json({ response: answer });
    } catch (error) {
        console.error("Error processing request:", error);
        res.status(500).json({ error: "Internal server error" });
    }
});

app.listen(3000, () => {
    console.log('App listening on port 3000');
});
```

Conclusion:

In this blog, we have explored the significance of accessibility in AI interactions and discussed the benefits of integrating ChatGPT with WhatsApp to address accessibility challenges effectively.

Accessibility is a crucial aspect of AI interactions, ensuring that users can engage with AI models seamlessly and effortlessly across various platforms and devices. The ability to access AI services without encountering barriers or limitations enhances user experience and promotes inclusivity in the digital landscape.

Integrating ChatGPT with WhatsApp offers numerous benefits that streamline the interaction process and enhance accessibility for users. By leveraging the user-friendly interface and widespread adoption of WhatsApp, users can seamlessly engage with ChatGPT from their preferred messaging platform, regardless of the device they are using or the media format they wish to employ as prompts.

The integration of ChatGPT with WhatsApp not only simplifies the process of interacting with AI models but also fosters collaboration, real-time communication, and multimedia integration, enriching the overall user experience.

As we move towards a more accessible and inclusive digital environment, it is essential for developers and organizations to explore the possibilities of integrating AI technologies with user-friendly platforms like WhatsApp. By doing so, we can empower users to engage with AI models more effectively, promote accessibility, and enhance the overall user experience in the digital realm.

I encourage readers to explore the potential of integrating AI technologies with platforms like WhatsApp and embrace innovative solutions that prioritize accessibility and inclusivity. Together, we can create a more accessible and inclusive digital landscape where everyone can participate and benefit from the transformative power of AI technologies.

Additional Resources:

For readers interested in implementing ChatGPT integration with WhatsApp, here are some relevant documentation, APIs, and tools:

1. WhatsApp Business API Documentation:
    - Access the official documentation for the WhatsApp Business API to learn about integration requirements, API endpoints, and best practices.
    - WhatsApp Business API Documentation

2. Gupshup API Platform:
    - Explore the Gupshup API platform, which offers comprehensive solutions for integrating chatbots and conversational interfaces with WhatsApp and other messaging platforms.
    - Gupshup API Documentation

3. Twilio API for WhatsApp:
    - Discover the Twilio API for WhatsApp, which provides tools and resources for building custom WhatsApp integrations, including chatbots and messaging workflows.
    - Twilio API for WhatsApp Documentation

4. ChatGPT API Documentation:
    - Access the official documentation for the ChatGPT API to learn about API endpoints, request parameters, and response formats for interacting with the AI model.
    - ChatGPT API Documentation

5. NGROK for Local Development:
    - Explore NGROK, a tool that enables secure tunneling of local server environments, allowing developers to expose their localhost to the internet for testing and development purposes.
    - NGROK Documentation

I invite readers to share their experiences and insights on enhancing accessibility in AI interactions. Whether you have implemented ChatGPT integration with WhatsApp or explored other innovative solutions, your perspectives and contributions are valuable in driving forward the conversation on accessibility and inclusivity in AI technologies. Feel free to share your experiences, challenges, and success stories in the comments section or reach out to us directly. Together, we can create a more accessible and inclusive digital ecosystem where AI technologies empower and enrich the lives of all users.