

Customer Support Website - with ChatGPT

(NodeJS application)

Maryam Zubair
Student Id: 19709

Table of Contents

- ❖ Introduction
- ❖ Design & Implementation
- ❖ Test
- ❖ Enhancement Ideas
- ❖ Conclusion
- ❖ References



Introduction

This project is creating an “Answer Question System” utilizing GPT 3.5 Turbo model by OpenAI. The system aims to provide accurate and relevant answers to the questions asked related to the content on the OpenAI website. To achieve this, a Flask application has been established earlier to process and respond to user questions using GPT-3.5 Turbo. Flask application exposes a REST API endpoint (/ask) that accepts questions via POST requests and generates responses based on the provided inquiries. The purpose of this project is to enhance the project's capabilities and user interaction, I am integrating this Flask application with a Node.js application using rest API



Design & Implementation

The implementation involved the following steps:

1. **API Endpoint in Flask:** Flask application exposes a REST API endpoint (/ask) to receive questions via POST request
2. **Node.js Application:** Acts as a client that communicates with the Flask application by sending questions and receiving responses.
3. **REST API Calls from Node.js:** Makes HTTP POST requests to the Flask application's /ask endpoint, passing the user's question in the request payload.
4. **Handling Responses in Node.js:** Receives the responses from the Flask application and presents them to users in a seamless and efficient manner.



API Endpoint in Flask:

A virtual address where other programs (like your Node.js application) can send requests to communicate with your Flask application.



OpenAI Question Answering

Ask a question:

what is OpenAI's mission and vision?

Submit

Answer: OpenAI's mission is to build safe AI and ensure AI's benefits are as widely and evenly distributed as possible. Their vision is to develop general learning algorithms that will help them achieve their long-term goal.

Node.js Application:

1. **Installing Node.js:** Runtime environment for executing JavaScript code outside web browser.
2. **Installing Node Packet Manager (NPM):** Default package manager for Node.js. Which provide vast repository of packages and libraries to integrate into your Node.js applications.

First two steps were already installed on my computer for which checking version.

Command: `node -v , npm -v`

3. **Initialize a Node.js Project:** Initializing new node.js package with default values for the package.json file.

Command: `npm init -y`

```
maryamz@Maryams-MacBook-Pro Webapplication(Express) % node -v
v18.17.1
maryamz@Maryams-MacBook-Pro Webapplication(Express) % npm -v
9.6.7
maryamz@Maryams-MacBook-Pro Webapplication(Express) % npm init -y
Wrote to /Users/maryamz/Desktop/Semester3/Generative AI/Project/HW_03/Webapplication(Express)/package.json:

{
  "name": "webapplication(express)",
  "version": "1.0.0",
  "description": "",
  "main": "main.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "keywords": [],
  "author": "",
  "license": "ISC"
}
```

Express Application:-

Using Express, very popular and flexible web application framework for Node.js to create robust server-side applications. Additionally, I will be utilizing Axios, a tool for making HTTP requests, to interact with my REST-API for answering user queries dynamically.

```
● maryamz@Maryams-MacBook-Pro Webapplication(Express) % npm i express axios
```

```
1  {
2    "name": "webapplication",
3    "version": "1.0.0",
4    "description": "",
5    "main": "main.js",
6    "scripts": {
7      "test": "echo \"Error: no test specified\" && exit 1"
8    },
9    "keywords": [],
10   "author": "",
11   "license": "ISC",
12   "dependencies": {
13     "axios": "^1.5.1",
14     "express": "^4.18.2"
15   }
16 }
17
```

Nodemon:

Installing Nodemon dependency: to monitors changes in files in a Node.js application and automatically restarting the server when changes are made.

Installed as Dev Dependency: Necessary for development and testing purposes but are not required for the application to run in a production environment.

Code : Npm i nodemon --save-dev

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

```
maryamz@Maryams-MacBook-Pro Webapplication(Express) % npm i nodemon --save-dev
```

```
added 34 packages, and audited 100 packages in 619ms
```

```
12 packages are looking for funding  
run `npm fund` for details
```

```
},  
"devDependencies": {  
  "nodemon": "^3.0.1"  
}
```


Ejs

Installing Ejs module in Node.js applications as it facilitates the creation of dynamic, maintainable, and organized HTML templates while allowing for the integration of JavaScript logic directly within the templates.

Code : npm i ejs

```
"dependencies": {  
  "axios": "^1.5.1",  
  "ejs": "^3.1.9",  
  "express": "^4.18.2"  
},  
"devDependencies": {  
  "nodemon": "^3.0.1"
```

Testing

Testing performed to ensure that both the applications (Flask & Express) are working perfectly fine. Taking questions from user and responding answer according using GPT-3.5 Turbo.

1. Running Flask Application First
2. Running Express Application (hitting API endpoint)



Flask Application

```
b_app.py
* Serving Flask app 'web_app'
* Debug mode: on
WARNING: This is a development server. Do not use it in a product
ion deployment. Use a production WSGI server instead.
* Running on http://127.0.0.1:5000
Press CTRL+C to quit
* Restarting with stat
* Debugger is active!
* Debugger PIN: 280-261-619
Received question: whats the day today?
127.0.0.1 - - [04/Oct/2023 11:16:20] "POST /ask HTTP/1.1" 200 -
```

Express Application

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

```
found 0 vulnerabilities
maryamz@Maryams-MacBook-Pro Webapplication(Express) % npx nodemo
n
main.js
[nodemon] 3.0.1
[nodemon] to restart at any time, enter `rs`
[nodemon] watching path(s): *.*
[nodemon] watching extensions: js,mjs,cjs,json
[nodemon] starting `node main.js`
Running on port 3000
Response data: I don't know.
```

Testing on Server:

The screenshot shows the Chrome DevTools Network tab. The 'All' filter is selected, and the 'ask' request is highlighted. The 'Preview' tab is active, showing the response body. The response is a JSON object with an 'answer' property set to 'I don't know.' and a 'question' property set to 'what is the day today?'. The 'Submit' button is visible below the input field.

Elements Console Sources Network Performance >> 2 1

Filter ☐ Invert ☐ Hide data URLs

All Fetch/XHR JS CSS Img Media Font Doc WS Wasm Manifest Other ☐ Has blocked cookies

☐ Blocked Requests ☐ 3rd-party requests

5000 ms 10000 ms 15000 ms 20000 ms 25000 ms 30000 ms 35000 ms 40000 ms 45000 ms

Name X Headers Payload Preview Response Initiator Timing

localhost

ask

ask

http://localhost:3000/ask

OpenAI Answer

answer: I don't know.

Ask a question:

Submit

3 requests 3.0 kB transferred

The screenshot shows the Chrome DevTools Network tab with the 'ask' request selected. The 'Response Headers' and 'Request Headers' are visible. The response headers include 'Connection: keep-alive', 'Content-Length: 1164', 'Content-Type: text/html; charset=utf-8', 'Date: Wed, 04 Oct 2023 18:23:58 GMT', 'Etag: W/"48c-6pTXzyR67H8KwHUNvFzeBGZeJV8"', 'Keep-Alive: timeout=5', and 'X-Powered-By: Express'. The request headers include 'Accept: */*', 'Accept-Encoding: gzip, deflate, br', 'Accept-Language: en-US,en;q=0.9', 'Connection: keep-alive', 'Content-Length: 35', 'Content-Type: application/json', 'Host: localhost:3000', and 'Origin: http://localhost:3000'.

Response Headers ☐ Raw

Connection: keep-alive

Content-Length: 1164

Content-Type: text/html; charset=utf-8

Date: Wed, 04 Oct 2023 18:23:58 GMT

Etag: W/"48c-6pTXzyR67H8KwHUNvFzeBGZeJV8"

Keep-Alive: timeout=5

X-Powered-By: Express

Request Headers ☐ Raw

Accept: */*

Accept-Encoding: gzip, deflate, br

Accept-Language: en-US,en;q=0.9

Connection: keep-alive

Content-Length: 35

Content-Type: application/json

Host: localhost:3000

Origin: http://localhost:3000

The screenshot shows the OpenAI Question Answering web application. The title is 'OpenAI Question Answering'. The input field contains the question 'what is the day today?'. The 'Submit' button is visible. The output field displays the answer 'Answer: I don't know.'.

OpenAI Question Answering

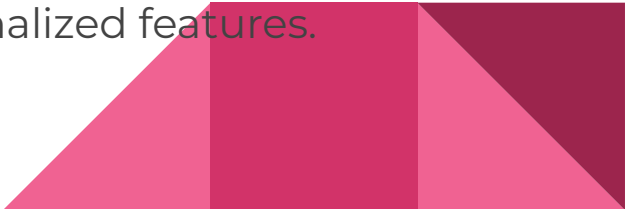
Ask a question:

what is the day today?

Submit

Answer: I don't know.

Enhancement Ideas

- 1. Deploy to a Web Server:** Hosting application on a cloud-based server (e.g., AWS, Azure, Heroku, DigitalOcean) to make it accessible to users over the internet.
 - 2. Domain Setup:** Purchasing and configuring a custom domain for my application to provide a more professional and branded user experience.
 - 3. SSL Integration:** Implement SSL (Secure Sockets Layer) to ensure secure communication between my server and users, especially if handling sensitive data.
 - 4. User Authentication:** Implement user authentication and authorization systems to allow users to create accounts, log in, and access personalized features.
- 

Conclusion

In this project, I successfully created a question-answering system using GPT-3.5 Turbo by OpenAI. I applied web scraping, data processing, embeddings, and GPT-3.5 Turbo, to craft an application that provides informative and accurate responses to user's queries. This project signifies the power of integrating AI technologies for practical and insightful applications. The purpose of this project was to enhance the project's capabilities and user interaction, to integrate Flask application with a Node.js application using rest API.



References

<https://platform.openai.com/docs/tutorials/web-qa-embeddings>

<https://www.digitalocean.com/community/tutorials/how-to-create-your-first-web-application-using-flask-and-python-3>

<https://www.geeksforgeeks.org/flask-http-methods-handle-get-post-requests/>



Github Link

[https://github.com/Maryam-Zubair/MachineLearning_Assignment/tree/main/ChatGPT/Customer%20Support%20\(NodeJS\)](https://github.com/Maryam-Zubair/MachineLearning_Assignment/tree/main/ChatGPT/Customer%20Support%20(NodeJS))

