**Documentation for Scraper API using Llama 3**

#### 1. Choice of LLM and Rationale:

I chose the LLaMA 3 model from Groq for this project. The rationale behind this choice includes:

- Advanced Language Understanding: LLaMA 3 excels in comprehending complex language structures, which is vital for accurately extracting information from HTML content.

- High Accuracy: It provides precise data extraction, reducing errors in the scraped information in comparison to BERT.

- Scalability: The model can handle large-scale data processing efficiently, making it suitable for extensive web scraping tasks.

#### 2. API Design and Implementation:

**Architecture**:

- Frontend: A simple HTML form where users can input the Flipkart URL they want to scrape.

- Backend: A Flask application that handles the scraping, processing, and conversion of data.

**Components**:

- HTML Form: index.html

- Main Application: app.py

- Groq Client: groq\_client.py

- Helper Functions: helper.py

- Scraper Logic: scraper.py

- Dependencies: requirements.txt

**Flow**:

1. User Input: The user inputs the Flipkart URL in the frontend form.

2. Scraping: The backend fetches the HTML content of the provided URL.

3. Processing: The HTML content is cleaned and split into chunks.

4. LLM Interaction: Each chunk is processed by the LLaMA 3 model to extract relevant information.

5. Compilation: The extracted data is combined and converted into JSON format.

6. Response: The JSON data is saved, and a success message is sent to the user.

#### 3. Setup and Testing Instructions:

**1. Clone the Repository:**  
git clone <repository-url>  
cd <repository-name>

**2. Create a Virtual Environment:**  
python -m venv venv  
source venv/bin/activate # On Windows use `venv\Scripts\activate`

**3. Install Dependencies:**  
pip install -r requirements.txt

**4. Set Environment Variables:**

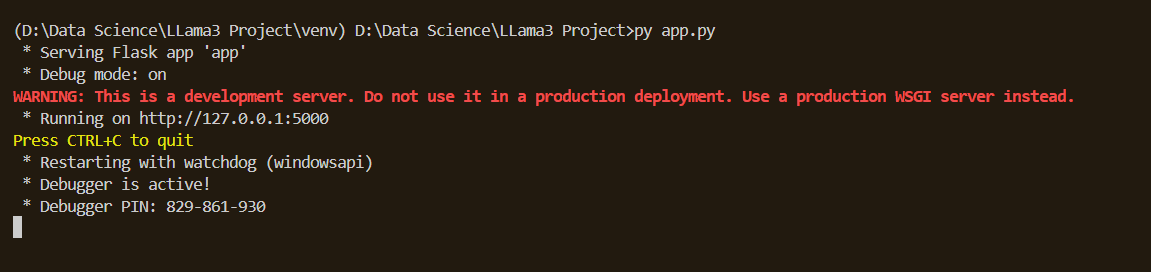
Create a `.env` file in the root directory and add your Groq API key:  
GROQ\_API\_KEY= “Enter gorq api key” (link https://console.groq.com/keys)

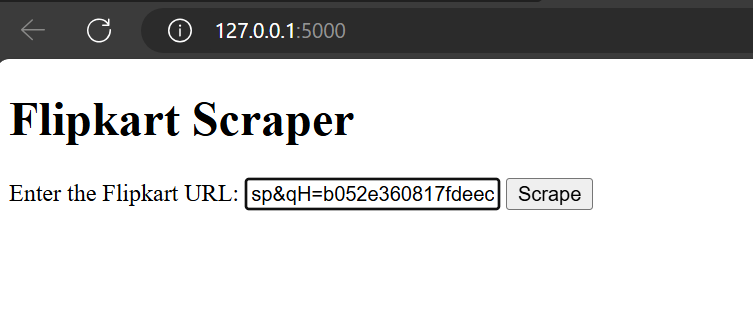
**5. Run the Application:**  
python app.py

**6. Access the Application:**

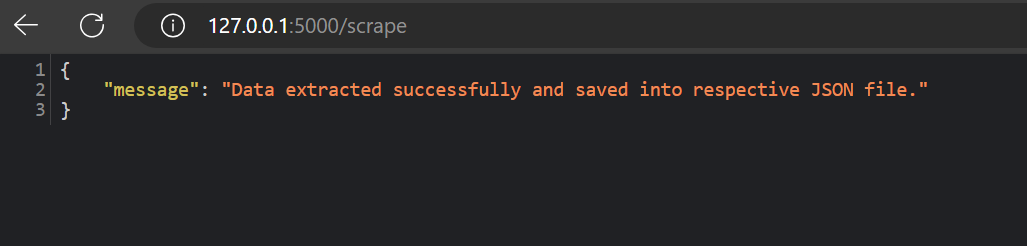
Open your web browser and navigate to `http://127.0.0.1:5000`.

#### 4. Example Inputs and Outputs:

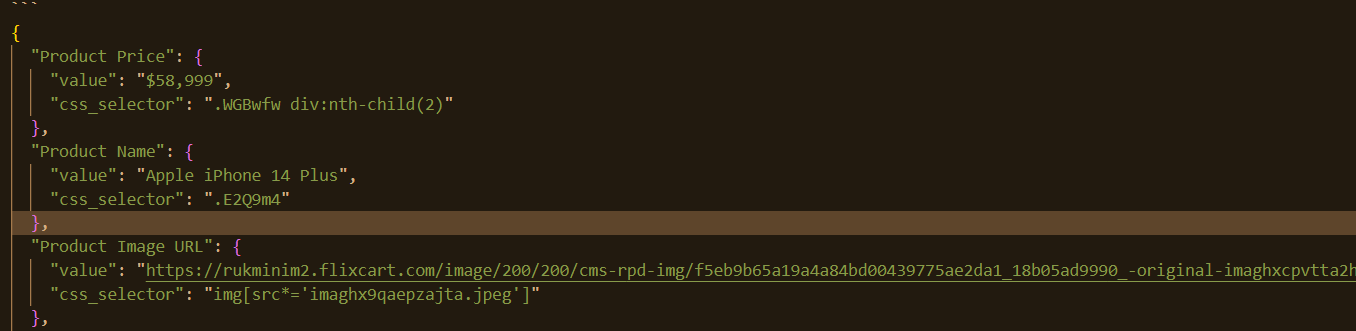




After entering url we will get



And the data will be updated in json file as



Here are the sample links to try out:

<https://www.flipkart.com/boat-airdopes-161-163-asap-charge-40-hrs-playback-bluetooth-headset/p/itmaf6bcbf816bbe?pid=ACCG6DS7Z2Z8SEAF&lid=LSTACCG6DS7Z2Z8SEAFJY1AMZ&marketplace=FLIPKART&q=headphone&store=0pm%2Ffcn&srno=s_1_15&otracker=search&otracker1=search&fm=Search&iid=8863ada8-8b6c-4535-a45c-d3445e23a76d.ACCG6DS7Z2Z8SEAF.SEARCH&ppt=sp&ppn=sp&ssid=e04e5109qo0000001716633801911&qH=b052e360817fdeec>

<https://www.flipkart.com/apple-iphone-14-plus-blue-128-gb/p/itmac8385391b02b?pid=MOBGHWFHUYWGB5F2&lid=LSTMOBGHWFHUYWGB5F2XIJVA7&marketplace=FLIPKART&q=iphone+14&store=tyy%2F4io&srno=s_1_1&otracker=AS_QueryStore_OrganicAutoSuggest_1_2_na_na_na&otracker1=AS_QueryStore_OrganicAutoSuggest_1_2_na_na_na&fm=Search&iid=709ac056-4065-4556-933b-ae8a2f10d56a.MOBGHWFHUYWGB5F2.SEARCH&ppt=sp&ppn=sp&ssid=x843uj7iog0000001716637352171&qH=860f3715b8db08cd>

<https://www.flipkart.com/r-rgk-3-fold-auto-open-close-umbrella/p/itm3b62641a46848?pid=UMBGRCSG6WYYYX2R&lid=LSTUMBGRCSG6WYYYX2RNVG9AI&marketplace=FLIPKART&q=chari&store=h1m%2Fiee%2Fkjp&srno=s_1_11&otracker=search&otracker1=search&fm=Search&iid=en_yWyOtiOgnFdpaAXwNBYh6A4Y3QG8LKCqZ10PxQ9w0Vfd6N91YFb69JCtx2S1xNgLnbhlfkMfIDknL4qOTAu74Q%3D%3D&ppt=sp&ppn=sp&ssid=ya8n0e6gv40000001716637728968&qH=8fb1c71c7e8fed20>

### Workflow:

Here’s a simple step by step workflow to illustrate the API’s process:

1. User Inputs URL: The user inputs the Flipkart URL on the frontend form (index.html).
2. POST Request: The form submits the URL to the /scrape endpoint via a POST request.
3. Scraping: The scraper.py fetches the HTML content from the provided URL.
4. HTML Processing: The HTML content is processed to remove unwanted tags and elements.
5. Chunk Processing: The cleaned HTML content is split into manageable chunks using helper.py.
6. Groq API Interaction: Each chunk is processed by the LLaMA 3 model using groq\_client.py.
7. JSON Compilation: The results from Groq are combined and converted into JSON format.
8. Saving JSON: The final JSON data is saved into flipkart\_data.json.
9. Response to User: The API responds to the user with a success message and the data is displayed on the frontend.

***Note: This Api is suitable for Flipkart URL but it can be designed for other ecommerce site or any other site as per the requirement.***