Project Title:

World's Top 50 Hospitals Data Collection and Private GPT Model Training

Table of Contents

1. Project Overview

This project focuses on collecting data from the "World's Top 50 Hospitals 2023" ranking and training a Private GPT model for healthcare-related tasks. It includes data collection, cleaning, model training, and documentation of the process.

2. Data Collection

2.1 Data Sources

Data was obtained from the "World's Top 50 Hospitals 2023" ranking published by Newsweek https://www.newsweek.com/rankings/worlds-best-hospitals-2023.

2.2 Web Scraping Methodology

Python libraries, including BeautifulSoup and requests, were employed for web scraping.
Hospital names and website URLs were extracted from the Newsweek ranking.
Data from each hospital's website, including name, location, services, and awards, was collected.

3. Data Cleaning

3.1 Cleaning Methodology

Data cleaning steps included:

- Handling missing data through imputation or marking.
- Standardizing data to ensure consistent formats.
- Removing duplicate entries within the dataset.

4. Model Training

4.1 Model Repository

Model training utilized the Private GPT repository hosted on GitHub: Private GPT Repository https://github.com/imartinez/privateGPT.

4.2 Model Training Process

The cleaned data was preprocessed for model training, including text encoding.
The Private GPT model was fine-tu ned for healthcare-related tasks, such as generating descriptions or answering questions about hospitals.

5. Documentation of Challenges

Several challenges were encountered during the project:

- Rate Limiting: Website requests had to be managed carefully as frequent pinging was not allowed.
- Access Issues: Some websites were not accessible, resulting in permission denied errors.
- Data Structure: Data from various hospital websites often appeared cluttered.

6. Model Performance

Model performance was assessed through interactions with the Private GPT model. Sample interactions, questions, and responses were documented to gauge the model's effectiveness.

6.1 Model Interaction Snippets

snippets or examples of interactions with the model, demonstrating its capabilities and limitations.

7. Replication Guide

To replicate this project, follow these steps:

7.1 Clone the Private GPT model repository from [GitHub](https://github.com/imartinez/privateGPT).

7.2 Data Scraping

Code:

```
# Function to scrape, clean, and display hospital data
def scrape_clean_and_display_hospital_data():
    # Send an HTTP GET request to the Newsweek URL
    newsweek_response = requests.get(newsweek_url, headers=headers)

    # Check if the request to Newsweek was successful
    if newsweek_response.status_code == 200:
        # Parse the HTML content of the Newsweek page using BeautifulSoup
        newsweek_soup = BeautifulSoup(newsweek_response.text, 'html.parser')

        # Find the table element with the specified class
        table = newsweek_soup.find('table', class_='ranking-table')

        # Extract rows from the table
        rows = table.find_all('tr')

        # Initialize counters to skip the 9th and 27th links
        link_counter = 0

        # List to store scraped data
        scraped_data = []

        # Initialize variables to store the current hospital data
        current_hospital_data = {
            "p_tags": [],
```

```
    "h1_tags": [],
    "h2_tags": [],
    "h3_tags": [],
    "div_tags": [],
}
current_hospital_name = ""
current_hospital_rank = ""

# Loop through the rows and scrape content from the top 50 hospitals with links
for row in rows[1:51]:  # Skip the header row and limit to the top 50 hospitals
    columns = row.find_all('td')
    rank = columns[0].get_text(strip=True)
    publication_name = columns[1].find('a').get_text(strip=True)
    country = columns[2].get_text(strip=True)
    city = columns[3].get_text(strip=True)
    state = columns[4].get_text(strip=True)
    link = columns[1].find('a')['href']

    # Increment the link counter
    link_counter += 1

    # Skip the 9th and 27th links and continue to the next link
    if link_counter == 9 or link_counter == 27:
        continue

    # Introduce a delay before sending the request to the hospital's link
    time.sleep(2)  # Wait for 2 seconds (adjust the delay as needed)

    # Send an HTTP GET request to the hospital's link
    hospital_response = requests.get(link, headers=headers)

    # Check if the request to the hospital's link was successful
    if hospital_response.status_code == 200:
        # Parse the HTML content of the hospital's page using BeautifulSoup
```

```
            hospital_soup = BeautifulSoup(hospital_response.text, 'html.parser')

            # Extract and clean content from p, h1, h2, h3, and div tags
            p_tags = [clean_html_tags(tag.get_text()) for tag in hospital_soup.find_all('p')]
            h1_tags = [clean_html_tags(tag.get_text()) for tag in hospital_soup.find_all('h1')]
            h2_tags = [clean_html_tags(tag.get_text()) for tag in hospital_soup.find_all('h2')]
            h3_tags = [clean_html_tags(tag.get_text()) for tag in hospital_soup.find_all('h3')]
            div_tags = [clean_html_tags(tag.get_text()) for tag in hospital_soup.find_all('div')]

            # Store the cleaned data in the current hospital data dictionary
            current_hospital_data["p_tags"] = p_tags
            current_hospital_data["h1_tags"] = h1_tags
            current_hospital_data["h2_tags"] = h2_tags
            current_hospital_data["h3_tags"] = h3_tags
            current_hospital_data["div_tags"] = div_tags

            # Store the current hospital name and rank
            current_hospital_name = publication_name
            current_hospital_rank = rank

            # Append the current hospital data to the list
            scraped_data.append({
                "Hospital Rank": current_hospital_rank,
                "Hospital Name": current_hospital_name,
                **current_hospital_data
            })

    return scraped_data
```

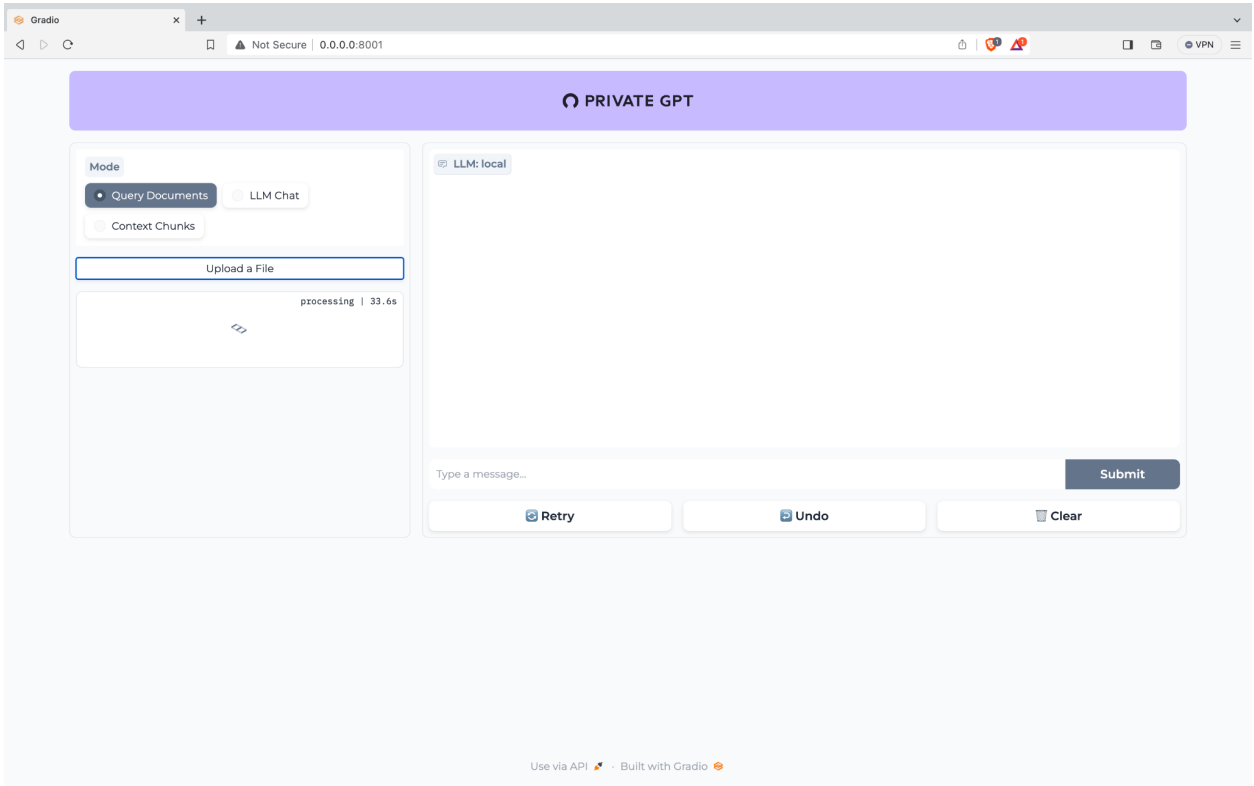
7.3 Data Cleaning

```
# Function to scrape, clean, and display hospital data
def scrape_clean_and_display_hospital_data():
    # ... (previous code)

    # Loop through the rows and scrape content from the top 50 hospitals with links
    for row in rows[1:51]:  # Skip the header row and limit to the top 50 hospitals
        columns = row.find_all('td')
        rank = columns[0].get_text(strip=True)
        publication_name = columns[1].find('a').get_text(strip=True)
        country = columns[2].get_text(strip=True)
        city = columns[3].get_text(strip=True)
        state = columns[4].get_text(strip=True)
        link = columns[1].find('a')['href']

        # ... (previous code)

        # Send an HTTP GET request to the hospital's link
        hospital_response = requests.get(link, headers=headers)

        # Check if the request to the hospital's link was successful
        if hospital_response.status_code == 200:
            # Parse the HTML content of the hospital's page using BeautifulSoup
            hospital_soup = BeautifulSoup(hospital_response.text, 'html.parser')

            # Extract and clean content from p, h1, h2, h3, and div tags
            p_tags = [clean_html_tags(tag.get_text()) for tag in hospital_soup.find_all('p')]
            h1_tags = [clean_html_tags(tag.get_text()) for tag in hospital_soup.find_all('h1')]
            h2_tags = [clean_html_tags(tag.get_text()) for tag in hospital_soup.find_all('h2')]
            h3_tags = [clean_html_tags(tag.get_text()) for tag in hospital_soup.find_all('h3')]
            div_tags = [clean_html_tags(tag.get_text()) for tag in hospital_soup.find_all('div')]

            # ... (previous code)
```

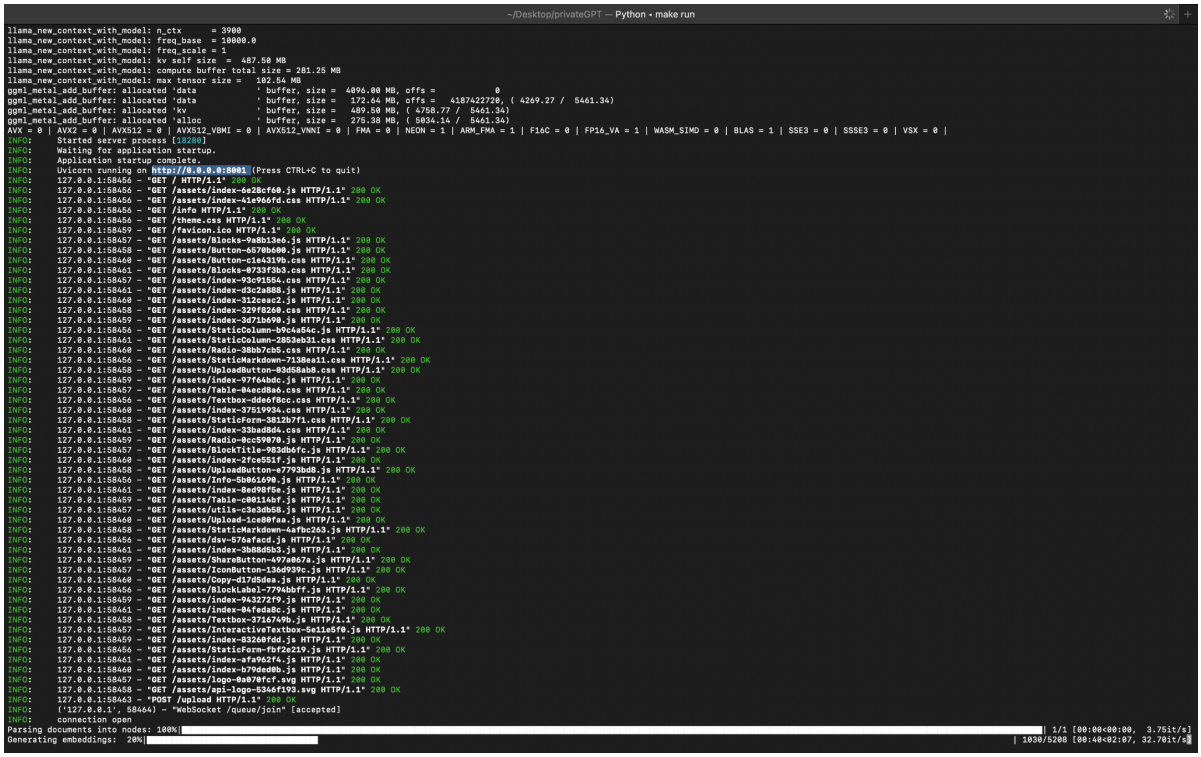
```
        # Store the cleaned data in the current hospital data dictionary

        # Append the current hospital data to the list
        scraped_data.append({
            "Hospital Rank": current_hospital_rank,
            "Hospital Name": current_hospital_name,
            **current_hospital_data
        })

    return scraped_data
```

7.4 Model Training

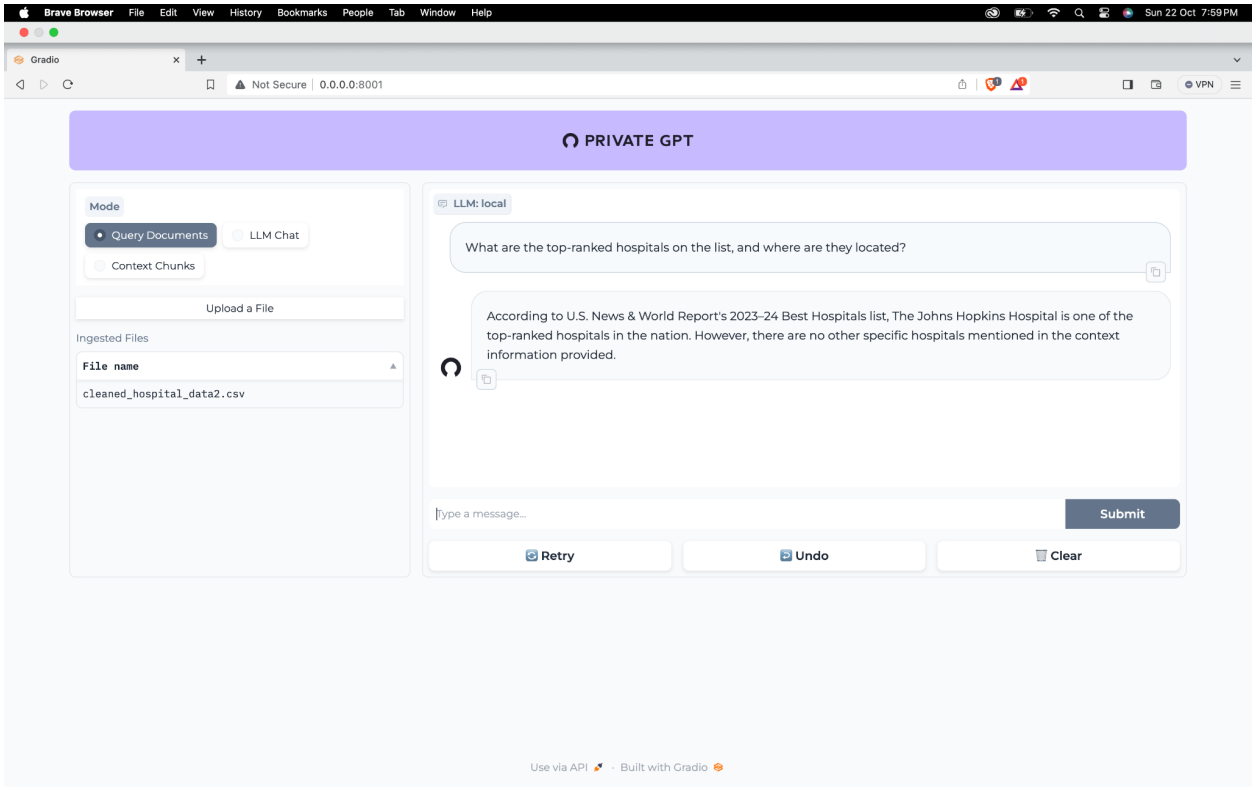
Img1:Uploding A Csv File.



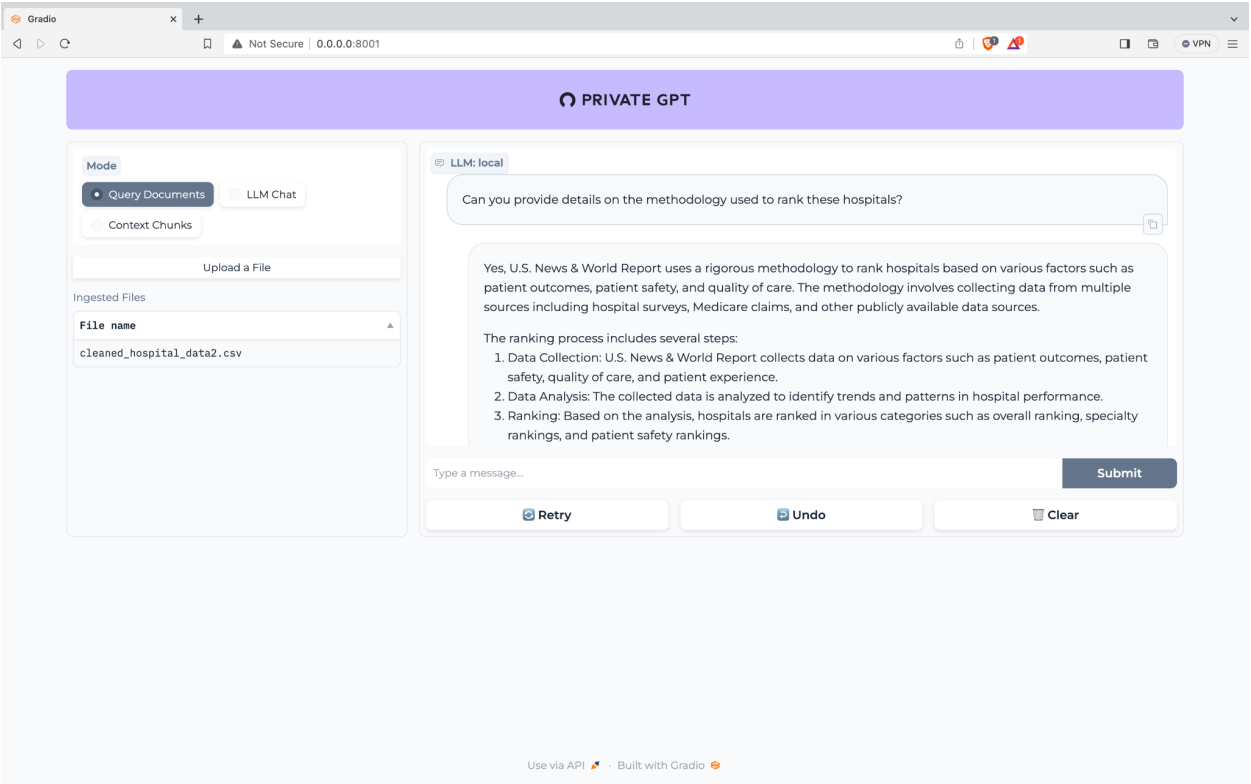
Img2: Executing a Csv File in Background.



Img3: Query:-What are the top Ranked Hospital



Img4:Query Can you provide details on the methodology used to rank these hospitals



Query You can explore more here is a List:-

1. What are the top-ranked hospitals on the list, and where are they located?
2. Can you provide details on the methodology used to rank these hospitals?
3. Are there any notable trends or changes in the rankings compared to previous years?
4. What criteria were used to assess the quality of these hospitals?
5. How many hospitals were considered in this ranking?
6. Can you provide information about the best hospitals in a specific country or region?
7. Are there any specialized or top-performing departments or services within these hospitals?
8. Do the rankings take into account the hospital's patient outcomes or other performance indicators?
9. Are there any specific medical breakthroughs or innovations associated with the top-ranked hospitals?
10. Can you provide information on the services and specialties offered by these hospitals?
11. Is there a way to access more detailed profiles or information about each hospital on the list?
12. Are there any specific areas of medical expertise or research where these hospitals excel?
13. What is the geographical distribution of the top hospitals? Are there notable clusters in certain regions?

14. Are there any partnerships or collaborations these hospitals have with other healthcare institutions?
15. Is there information about the accessibility and affordability of healthcare services in these top-ranked hospitals?

8. Conclusion

In conclusion, the "World's Top 50 Hospitals Data Collection and Private GPT Model Training" project successfully collected data from leading hospitals, cleaned and processed the data, and trained a Private GPT model for healthcare-related tasks. Challenges were documented, and model performance was evaluated. The provided replication guide allows others to reproduce the project's results.