**Text Summarization of PubMed Articles using Gemini API**

**Introduction**

The PubMed Summarization Project aims to simplify and expedite the review process of articles from PubMed by providing a tool that summarizes lengthy texts into concise versions. This project involves loading and preparing a dataset for summarization, and developing a web application where users can input PubMed articles to receive a summarized version using a state-of-the-art API-based model like Gemini by Google.

**Data Exploration and Preparation**

This all work has been done on Jupyter Notebook. Main libraries we used for data cleaning and preprocessing were pandas and nltk.

**1. Loading the Dataset**

We use the PubMed Summarization dataset available on Hugging Face. This dataset contains pairs of PubMed articles and their summaries, which will be instrumental in training and evaluating our summarization model.

For this you must have to install the dataset library of Python by using command: pip install datasets

**2. Exploring the Dataset**

This step helps us identify key components such as article text, summary, and any metadata.

**3. Preprocessing the Dataset**

Preprocessing is crucial for effective summarization. Typical preprocessing steps include Tokenization, Removing special characters and stop words, Lowercasing the text, Lemmatization.

After this process we stored our data in a csv file for further operations.

**Web Application Development**

**Choosing the Framework**

For this project, we use Flask, a lightweight and flexible web framework for Python. Flask allows for building scalable and customizable web applications with ease.

**1. Set Up Flask:**

Create a new directory, app, for the Flask application. Inside this directory, create the main Flask script, app.py, and a folder, templates, for HTML files.

**2. Create app.py:**

Set up the basic structure of the Flask app to accept user input or file upload.

**3. Create HTML Templates:**

In the templates directory, create index.html for the home page and summary.html for displaying the results.

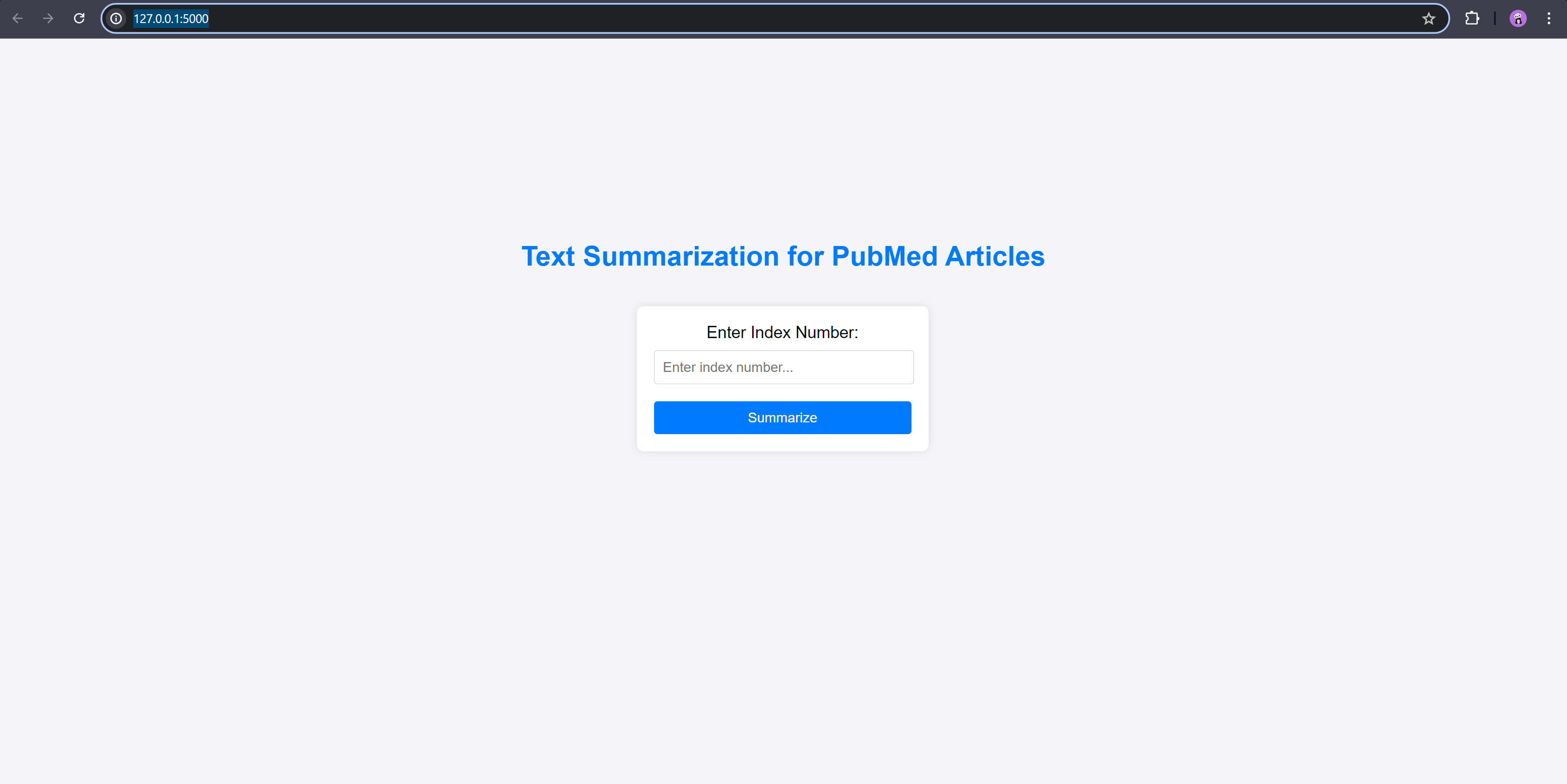
You just need to type this command on terminal to run the application: python app.py

**Integrating Text Summarization**

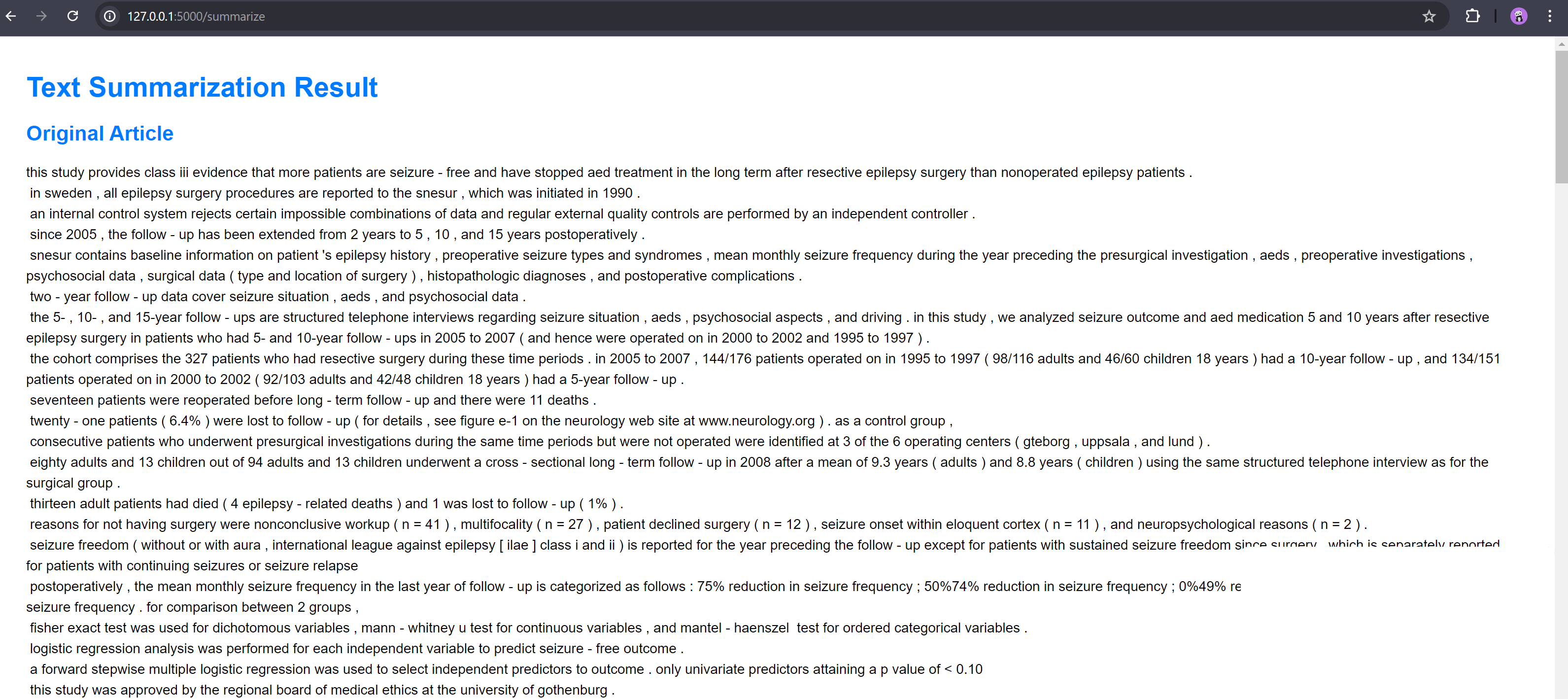
We used the API-based model (Gemini by Google) for summarization. Assuming you have API access and key. You can get it from Google AI studio.

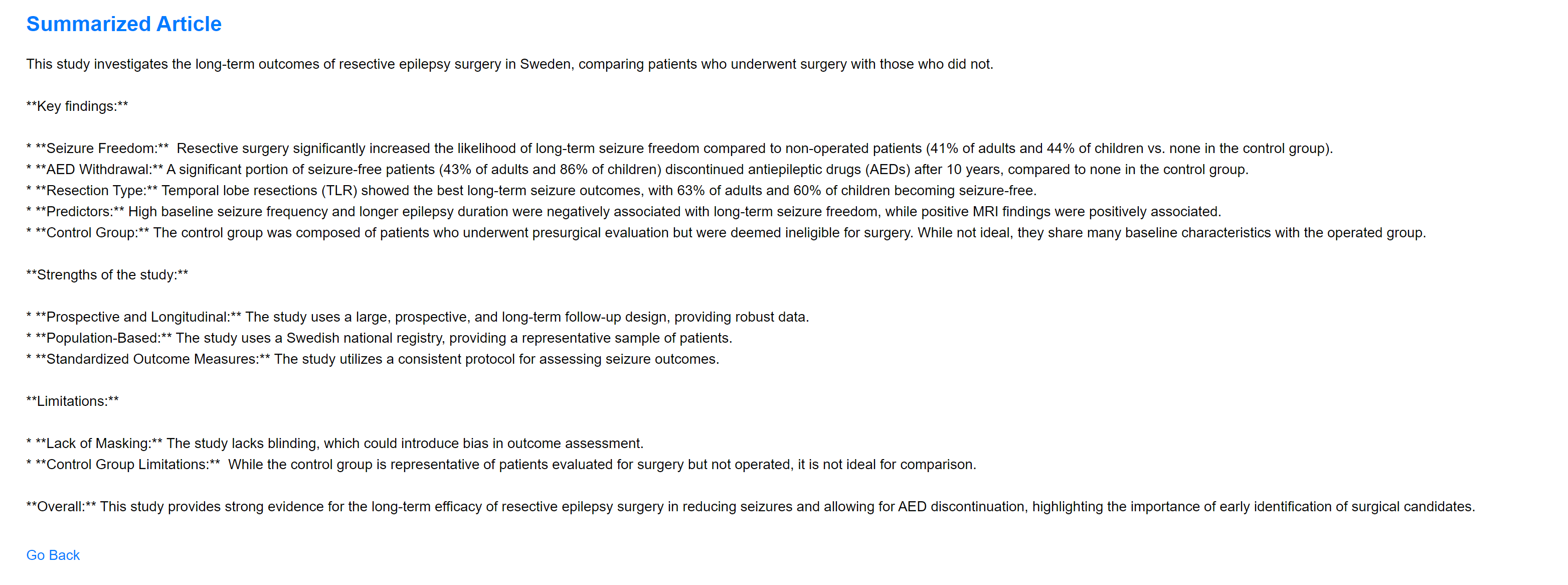
**Interacting with the Application**

Get index of the article as an input from user. Click the "Summarize" button to see the summarized version of the article.



**Result:**





**Future Improvements**

Enhanced Preprocessing: Implement more advanced text preprocessing techniques like stemming or lemmatization.

Model Fine-Tuning: Train and fine-tune the summarization model on domain-specific data for improved accuracy.

User Interface: Improve the UI for better user experience and support for more file types.

Caching and Optimization: Optimize API calls and application performance.

**Conclusion**

The PubMed Summarization Project encapsulates a comprehensive approach to simplifying the review process of scientific literature by harnessing the power of advanced text summarization technologies. Through meticulous data exploration and preparation, we ensured that the dataset was ready for effective summarization. The development of a user-friendly web application using Flask has provided an accessible platform for users to quickly and efficiently summarize lengthy PubMed articles. By integrating a cutting-edge API-based summarization model like Gemini, the project not only enhances the accessibility of scientific knowledge but also accelerates research and review workflows.

This project serves as a testament to the potential of combining data science with modern web technologies to create tools that can significantly impact research communities. While the current implementation provides a robust foundation, there is ample scope for future improvements, such as fine-tuning the summarization model for specific domains, enhancing the preprocessing pipeline, and expanding the application's capabilities to support a broader range of document formats and languages.

Overall, this project stands as a valuable resource for researchers, academics, and professionals who seek to keep pace with the ever-growing volume of scientific literature. By continuing to build on this work, we can further bridge the gap between extensive research data and its efficient, digestible presentation.