Saaransh: Chrome Extension for Text Summarization and Major Points Extraction

Introduction:

The "Saaransh" Chrome Extension is designed to enhance users' browsing experience by providing them with the capability to quickly obtain summarized content and major points from web pages. The extension offers a user-friendly interface that interacts with a backend server to generate concise summaries and key points based on the content of a given webpage.

Project Components:

The project consists of three main components: the Flask backend server, the Chrome extension's HTML, CSS, and JavaScript, and the OpenAI GPT-3.5 Turbo API to generate summaries and major points.

Flask Backend Server:

- The Flask backend serves as the bridge between the Chrome extension and the OpenAI GPT-3.5 Turbo API.
- It receives requests from the extension, extracts text content from the specified webpage using web scraping techniques, and interacts with the OpenAI API to generate summaries and major points.
- The backend server also includes CORS headers to enable cross-origin requests.

Chrome Extension:

1. **Manifest:**
   - The manifest file defines the extension's metadata, permissions, and default behaviours.
   - It also Reference's popup HTML, CSS, and JavaScript files.
2. **Popup HTML:**
   - The HTML file defines the layout and structure of the popup window. It includes placeholders for displaying the summary, major points, loading indicator, and buttons for interaction.
3. **Popup CSS:**
   - The CSS file styles the popup's visual appearance, including buttons, loading indicator, and content display.
4. **Popup JavaScript:**
   - The JavaScript file adds interactivity to the popup. It fetches the current URL of the active tab using the Chrome API and sends requests to the Flask backend to generate summaries and major points.
   - The script updates the popup's content dynamically and provides the "Copy to Clipboard" functionality.

OpenAI GPT-3.5 Turbo API:

- The OpenAI API is used to generate summaries and major points based on the extracted content of web pages.
- API requests are made by the Flask backend, which acts as an intermediary between the extension and the OpenAI service.

Detailed Instructions for installing and using the extension:

Installation:
- Download the extension package from the provided source, the package will likely be in a compressed format, such as a ZIP file.
- Unzip the package to a directory of your choice.

Adding the Extension to Chrome:
- Open Google Chrome.
- In the address bar, type "chrome://extensions/" and press Enter.
- Enable "Developer mode" toggle switch at the top right corner.
- Click "Load unpacked" button.
- Browse to the directory where you unzipped the extension package and select the folder "Frontend" that contains extension files.
- The "Saaransh" extension will appear in the list of installed extensions.

Starting the Backend Server:

1. **Install Required Packages:**
   - Make sure you have Python installed on your system.
   - Open a terminal or command prompt.
   - Navigate to the directory where the Flask backend code is located.
   - Run the following command to install the required packages:

   pip install flask flask-cors requests beautifulsoup4

2. **Get OpenAI API Key:**
   - To use the OpenAI GPT-3.5 Turbo API, you need an API key.
   - Sign up for an account on the OpenAI platform and obtain your API key.

3. **Set API Key:**
   - Open the Flask backend code (Python file).
   - Find the line where the "openai_api_key" variable is assigned.
   - Replace "YOUR_OPENAI_API_KEY" with your actual API key.

4. **Start the Backend Server:**
   - In the terminal or command prompt, navigate to the directory containing the Flask backend code.
   - Run the following command to start the backend server:

   py main.py

5. **Server Running:**
   - The backend server should now be running and accessible at "http://127.0.0.1:3000/".

Using the Extension:
- Click on the "Saaransh" extension icon in the Chrome toolbar to open the popup.
- The popup will display two buttons: "Provide Summary" and "Provide Major Points."
- Click "Provide Summary" to get a summary of the content from the active tab's webpage.
- Click "Provide Major Points" to generate key points from the same content.
- The extension will communicate with the backend server and OpenAI API to generate content. Wait for the process to complete.

- The generated summary or major points will be displayed in the popup.
- Use the "Copy to Clipboard" button to copy the content for your use.
- You can generate content from different web pages and interact with the extension as needed.

Workflow:

- The user interacts with the Chrome extension by clicking on the "Saaransh" extension icon in the browser toolbar.
- The extension popup opens, displaying buttons for "Provide Summary" and "Provide Major Points."
- the user clicks the "Provide Summary" or "Provide Major Points" button, the JavaScript code fetches the active tab's URL and sends a request to the Flask backend.
- The Flask backend extracts text content from the specified webpage using web scraping and sends a request to the OpenAI GPT-3.5 Turbo API for text generation.
- The OpenAI API processes the request and generates summary or major points based on the input text.
- The Flask backend receives the API response, extracts the generated content, and sends it back to the extension.
- The extension's JavaScript code updates the popup's content with the generated summary or major points.
- The user can copy the content to the clipboard using the "Copy to Clipboard" button.

Conclusion:

The "Saaransh" Chrome Extension provides a convenient tool for users to quickly access summarized content and major points from web pages they visit. By integrating web scraping, the OpenAI GPT-3.5 Turbo API, and user-friendly interface components, the extension enhances the browsing experience and facilitates the extraction of essential information from lengthy content.

References:

- Flask Documentation: https://flask.palletsprojects.com/
- OpenAI GPT-3.5 Turbo API Documentation: https://beta.openai.com/docs/api-reference/introduction
- Chrome Extension Development Guide: https://developer.chrome.com/docs/extensions/mv3/getstarted/