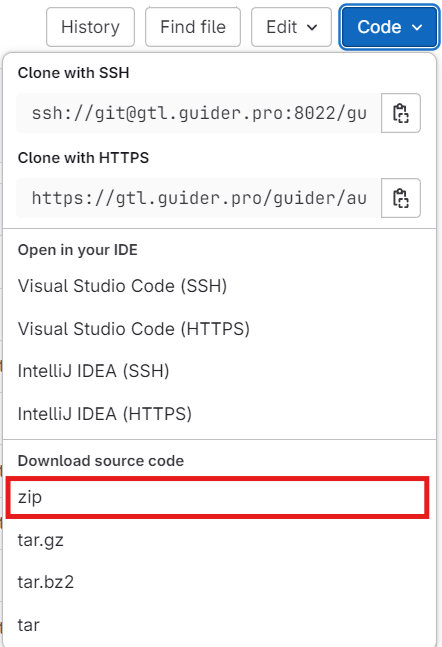
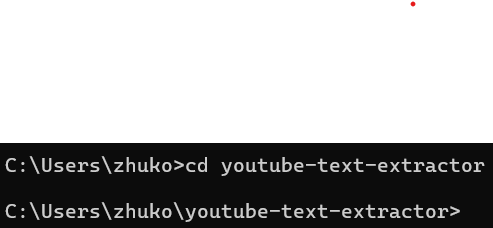
Запуск веб-приложения youtube\_text\_extractor

1. Необходимо скопировать папку с проектом себе на ПК. Сделать это можно, нажав на кнопку «Code» и выбрав «Download source code» и выбрав вариант zip (проект будет скачан на ПК в архивированном виде).

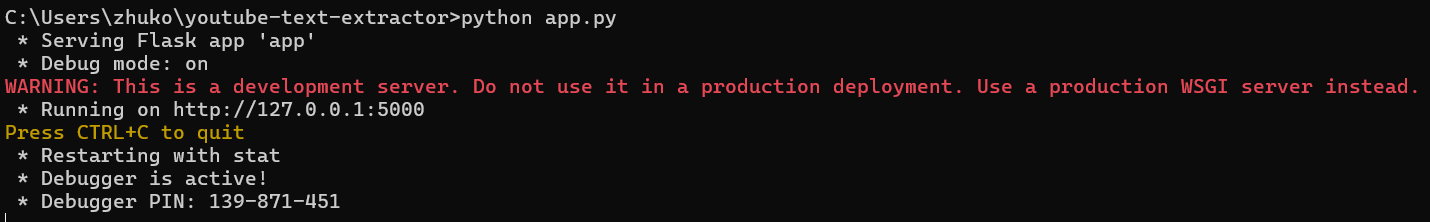


1. Необходимо распаковать архив в необходимую директорию.
2. Открыть командную строку Windows (для этого в поиске можно ввести .cmd или найти ее вручную). По умолчанию при запуске вы находитесь в C:\Users\user\_name>, где user\_name - ваше имя пользователя в системе. Далее, необходимо перейти в директорию проекта (указать путь к папке, в которую вы распаковали архив). Например, у меня это C:\Users\zhuko\youtube-text-extractor. Итак, чтобы перейти в директорию проекта, нужно выполнить команду cd и далее указать путь к файлу. Например, у меня этот запрос будет выглядеть так:

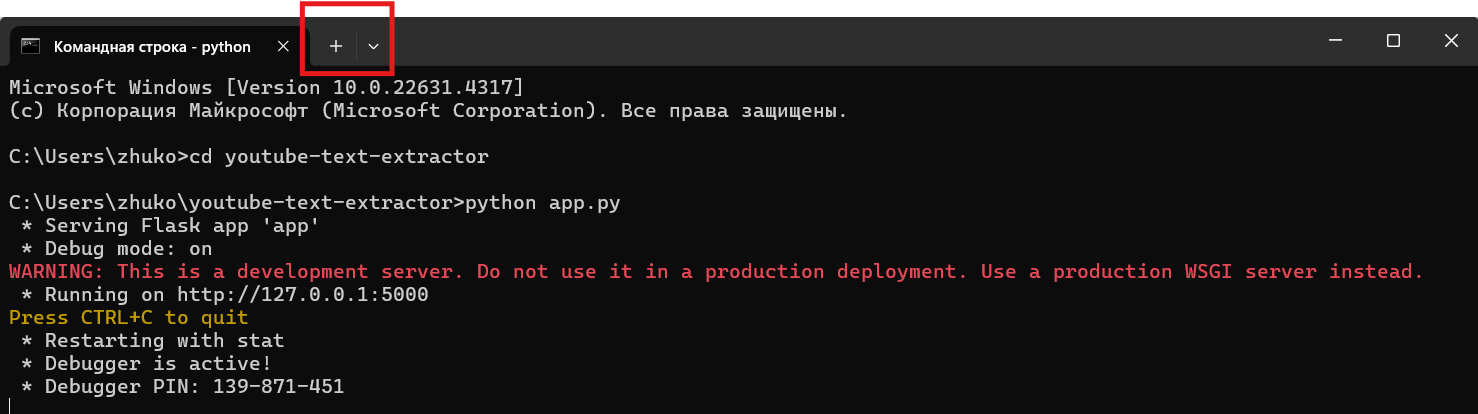


Как можно видеть, команда выполнилась, и мы перешли в директорию проекта.

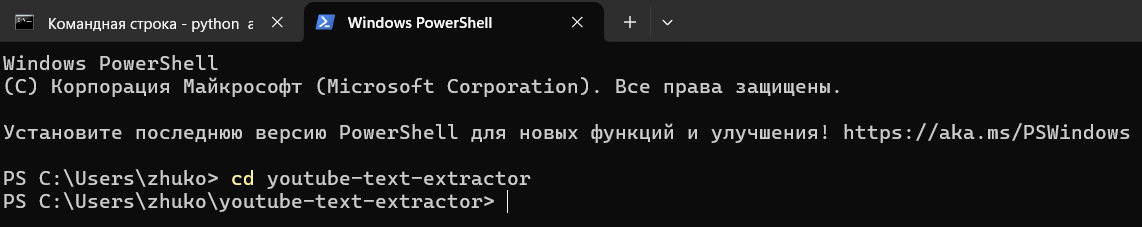
1. Далее необходимо установить все необходимые зависимости, используя команду pip install -r requirements.txt в терминале.
2. Затем необходимо запустить бэкенд часть приложения используя команду python app.py, вы увидите следующие сообщения, свидетельствующие о том, что сервер Flask запущен:



1. Необходимо открыть еще одну вкладку терминала:

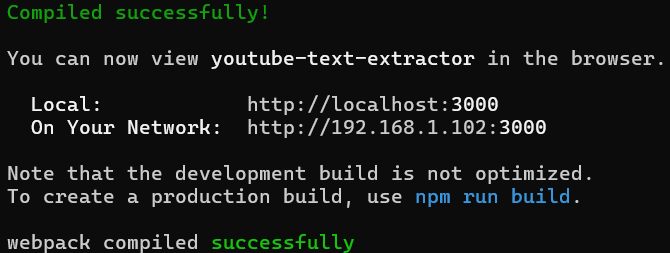


И также перейти в директорию проекта (можно скопировать запрос из соседней вкладки):



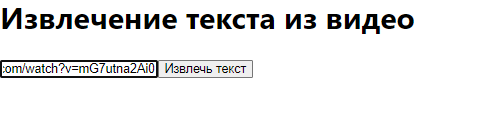
И наконец выполнить команду npm start для запуска приложения на React.

Вы увидите такое сообщение в терминале:



При этом у вас автоматически откроется вкладка в браузеру с адресом <http://localhost:3000/>. Если этого не произошло, откройте браузер и вручную введите этот адрес.

1. На вкладке вы увидите простой интерфейс для вставки url видео из Youtube. Вставляете ссылку на видео в поле и нажимаете «Извлечь текст».



Необходимо подождать несколько секунд, после чего вы увидите весь извлеченный текст из видео.



1. Далее этот текст можно «скормить» нейросети со следующим промтом:

«Нужно суммаризировать текст: выделить основные темы заголовками, внутри заголовков - суть рассматриваемых проблем и пути их решения, предложенные в исходном тексте, максимально подробно. В конце сделать общий вывод по тексту».

Модель выдает такой результат:

Конечно, вот суммаризированная версия текста с выделением основных тем и путей их решения:

### Фильтры и теги

\*\*Основные темы:\*\*

- Структура фильтров и тегов.

- Взаимосвязь между фильтрами и тегами.

\*\*Проблемы и решения:\*\*

- Дата-менеджеры должны задавать индексы сортировки для тегов.

- В поиске будет использоваться индекс сортировки и частота использования тегов.

- Фронтенд должен корректно отображать результаты поиска.

### Поиск и сортировка

\*\*Основные темы:\*\*

- Использование индекса сортировки и частоты использования тегов.

- Корректное отображение результатов поиска.

\*\*Проблемы и решения:\*\*

- Дата-менеджеры задают индексы сортировки для тегов.

- Фронтенд должен корректно отображать результаты поиска.

### Клавиатура и формы

\*\*Основные темы:\*\*

- Различные типы ввода (поиск, формы).

- Использование разных кнопок (Search, Next, Done) в зависимости от контекста.

\*\*Проблемы и решения:\*\*

- Для поиска и форм будут использоваться разные кнопки.

- Важно учитывать контекст при выборе кнопок.

### Фотографии

\*\*Основные темы:\*\*

- Ограничения на количество и размер фотографий.

- Поддержка популярных форматов изображений.

\*\*Проблемы и решения:\*\*

- Ограничения на количество и размер фотографий будут установлены с учетом безопасности и удобства пользователей.

- Важно поддерживать популярные форматы изображений.

### Безопасность

\*\*Основные темы:\*\*

- Вопросы безопасности.

- Передача вопросов специалистам по безопасности.

\*\*Проблемы и решения:\*\*

- Вопросы безопасности будут переданы специалистам по безопасности для дальнейшего решения.

### Общий вывод

В ходе совещания были определены структура фильтров и тегов, а также их взаимосвязь. Дата-менеджеры должны задавать индексы сортировки для тегов. В поиске будет использоваться индекс сортировки и частота использования тегов. Фронтенд должен корректно отображать результаты поиска. Для различных типов ввода (поиск, формы) будут использоваться разные кнопки (Search, Next, Done) в зависимости от контекста. Ограничения на количество и размер фотографий будут установлены с учетом безопасности и удобства пользователей. Важно поддерживать популярные форматы изображений. Вопросы безопасности будут переданы специалистам по безопасности для дальнейшего решения.

Как можно видеть результат впечатляет, особенно если посмотреть на исходный текст. В данном примере я использовал Mistral (<https://chat.mistral.ai/chat>), одно из преимуществ которой – большое количество входящих токенов, а также работа без необходимости регистрации и использования vpn. Но вы можете использовать свой вариант, а также экспериментировать с промтом!

.