План / Выводы

- FastAPI и UploadFile не могут автоматически обработать отправленные данные файла в виде байтов, без этой дополнительной информации. Они обычно принимают данные как часть multipart/form-data запроса.

- Данные отправленные потоком байтов без дополнительной информации нужно обрабатывать на стороне сервера самому,

- HTTP/1.1 и HTTP/2, которые обычно используются для передачи данных в веб-приложениях, не поддерживают передачу нескольких потоков байтов в одном запросе без какой-либо формы структурирования или кодирования.

- gRPC поддерживает передачу нескольких потоков байтов в одном запросе без необходимости в дополнительной сериализации или кодировании.

- На HTTP и FastAPI нужно выбрать между структурированием данных (например, с помощью multipart/form-data или JSON) и передачей данных как одного потока байтов и дальнейшей сериализацией или кодирования.

- Сериализация - это процесс преобразования структуры данных или объекта (например словарь, список и тд) в последовательность байтов.

- Кодирование - это процесс преобразования данных из одного формата в другой, например преобразование имени файла в байты с помощью os.path.basename(filepath).encode('utf-8').

- Нулевой байт, обозначаемый как b'\0' использую как разделитель

- UploadFile - класс FastAPI для работы с загружаемыми файлами, предоставляет полезные методы и атрибуты, таких как .filename для получения имени файла и .read() для чтения содержимого файла.

- Класс Form также может быть использован для получения файлов, отправленных с помощью формы multipart/form-data. Однако, в этом случае файл будет доступен как строка (str) или байты (bytes)

- В FastAPI чтение тела запроса является асинхронной операцией, и нет встроенного способа сделать это синхронно

-

ВЫВОД:

- Если передавать с помощью стандартных классов FastApi - UploadFile и File, то передача более удобная (так как передаются разные данные файла, можно передавать несколько параметров), но медленная скорость передачи.

- Если передавать чисто байты информации, то время передачи быстрое, но нельзя передать несколько потоков байтов в одном запросе, поэтому чтобы передать информацию из файла и его имя а также другие данные, нужно их соединять в один поток байтов, а потом на сервере разъединять и декодировать, что является не очень удобной практикой.

- Если использовать gRPC то время передачи примерно на 30% быстрее, а также есть возможность использовать множество потоков байтов в одном запросе.

-

ПЛАН:

- Нужно сравнить передачу данных двух видов (много маленьких файлов, один большой файл).

Передавать буду файл и его имя, так как без имени не бывает файлов.

HTTP1 и HTTP2:

1 - синхронная передача

* только байты (**request: Request)**

2 - асинхронная передача

* только байты (**request: Request)**

3 - веб сокеты

4 - gRPC

передачу можно выполнить:

* gRPC
* синхронно
* асинхронно
* в разных потоках
* в разных процессах. (не получается использовать ProcessPoolExecutor скорее всего из-за того что передаем клиента, а он является блокировщиком и возникает ошибка TypeError: cannot pickle '\_thread.RLock' object)