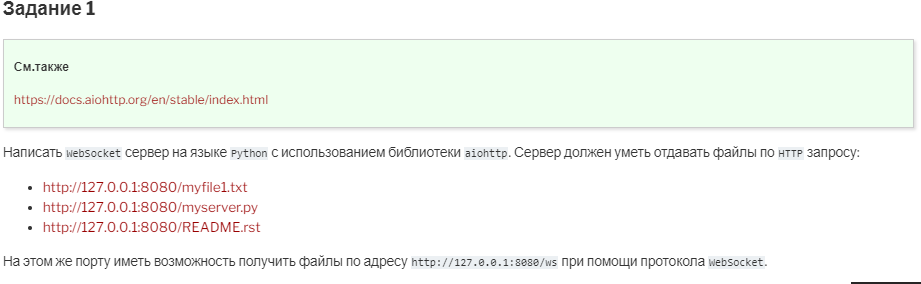
Лабораторная работа 3

Тюнькин Александр Евгеньевич РИ-380014

Задание 1

Постановка задачи:



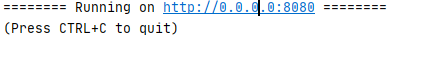
Для начала нам необходимо написать WebSocket сервер, основываясь на информации из лекции и иных открытых источников.



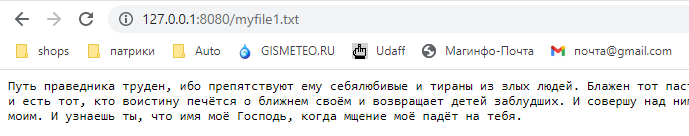


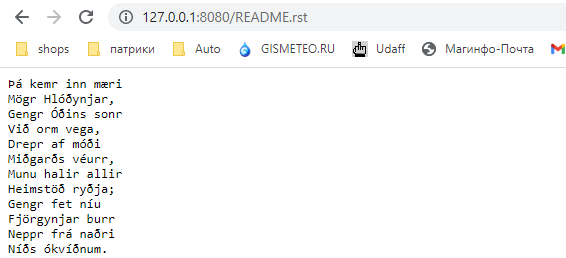
Так как msg.data содержит в себе строку, пересекающуюся с названием файла, находящегося с файлом сервера в одной директории, будет разумно вместо прописания для каждого из подобных файлов условия через if msg.data == ‘filename’ указать нечто более универсальное, как, например, префикс ‘./’ к названию запрашиваемого файла для нахождения оного в текущей директории, что упростит процесс добавления файлов.

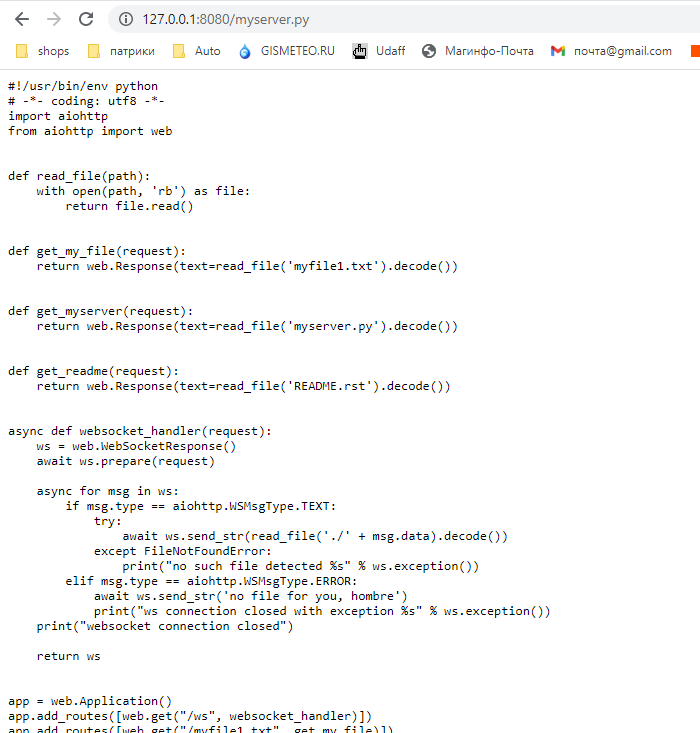
Далее — запускаем сервер.



Открыв localhost в браузере, начинаю поочередно запрашивать файлы.







Далее нам необходим клиент для того, чтобы было возможным получение файлов через websocket. Код клиента на языке python указан ниже.

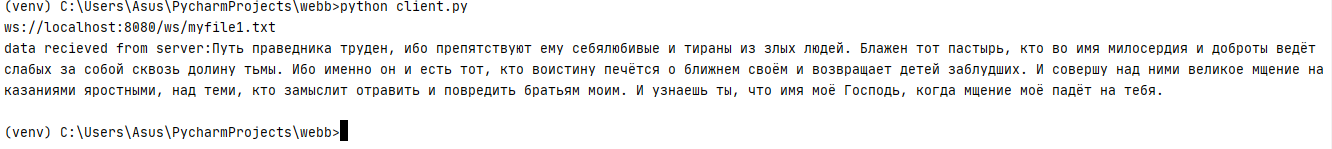


Запуск клиентского приложения и запрос файла README.rst.



Ответ получен. Websocket соединение было закрыто.

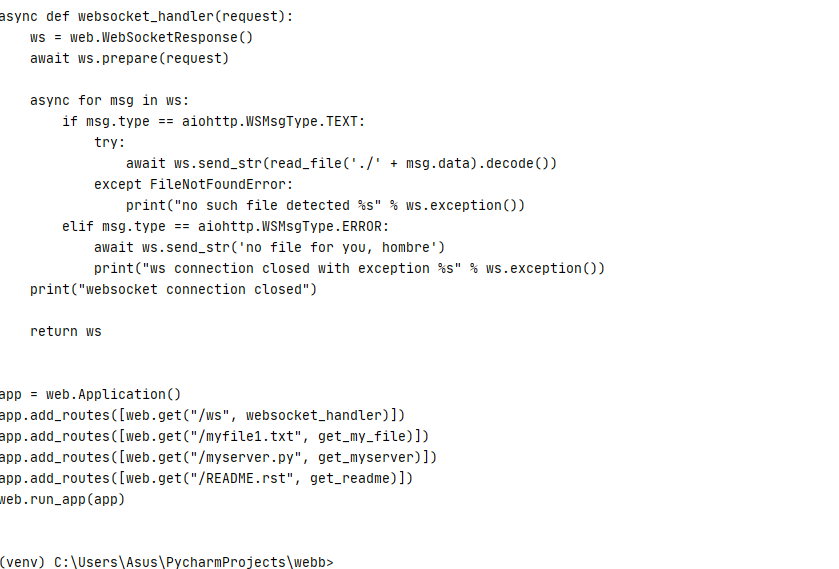
Аналогичным образом запрашиваю файл myfile1.txt.



Ответ получен. Websocket соединение было закрыто.

Аналогичным образом запрашиваю файл myserver.py.

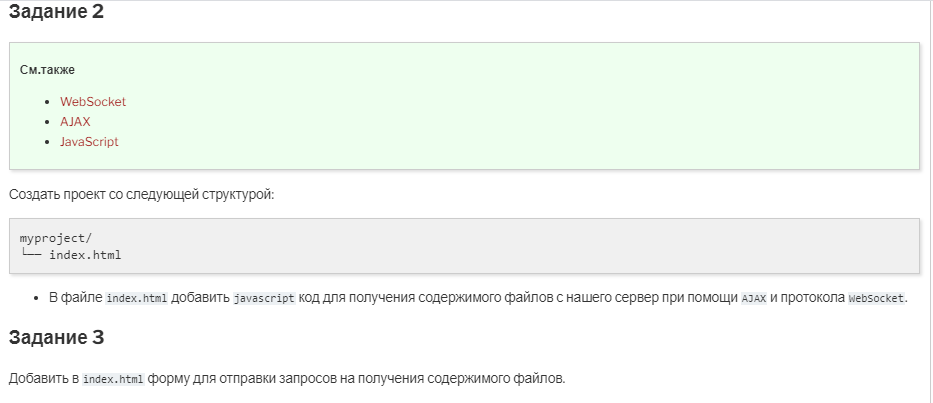




Ответ получен. Websocket соединение было закрыто.

Задания 2, 3

Постановка задачи:



Создаем файл index.html с javascript, в котором выполняется отправка запроса на сервер, ожидание ответа и установка слушателя для кнопки. Это не является стандартной формой html, так как данному скрипту присуща событийность: при нажатии на кнопку срабатывает скрипт и происходит подключение по вебсокет к созданному нами ранее серверу.



При запущенном сервере, открываем в браузере файл index.html, содержащий форму для получения файла через написанный ранее скрипт.



Далее вводим названия файлов в поле и получаем информацию с сервера.

