**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ**

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ**

**ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ**

«ДОНСКОЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ» (ДГТУ)

Факультет «Информатика и вычислительная техника»

Кафедра «Кибербезопасность информационных систем»

**Лабораторная работа № 2**

на тему «установка соединения по протоколу SSL/TLS»

|  |
| --- |
| Выполнил: студент группы ВКБ43 |
| Ковалев Данил Петрович |
| (Фамилия, имя, отчество) |
| Проверил: |
|  |
| (Фамилия, имя, отчество) |

Оглавление

[Цель: 3](#_Toc177201999)

[Задание: 3](#_Toc177202000)

## **Цель:**

Научиться создавать безопасное подключение между клиентом и сервером mysql на базе стека протоколов ssl/tls.

## **Задание:**

1. Сгенерировать сертификаты.

2. Настроить mysql для доступа по сети с использованием SSL.

3. Продемонстрировать доступ по сети с правильным / неправильным сертификатом и выполнение команд на изменение данных (клиент и сервер могут быть на разных виртуальных машинах).

4. Разработать прикладное ПО, которое демонстрирует доступ к БД по сети с использованием сертификата и позволяет отображать данные БД в виде таблицы, а также редактировать их.

## **Ход работы:**

Для генерации сертификатов OpenSSL использовался скрипт, который представлен на рисунке 1. В данном случае он будет работать только на платформе Windows. Преждевременно надо установить OpenSSL на ПК, чтобы все заработало.

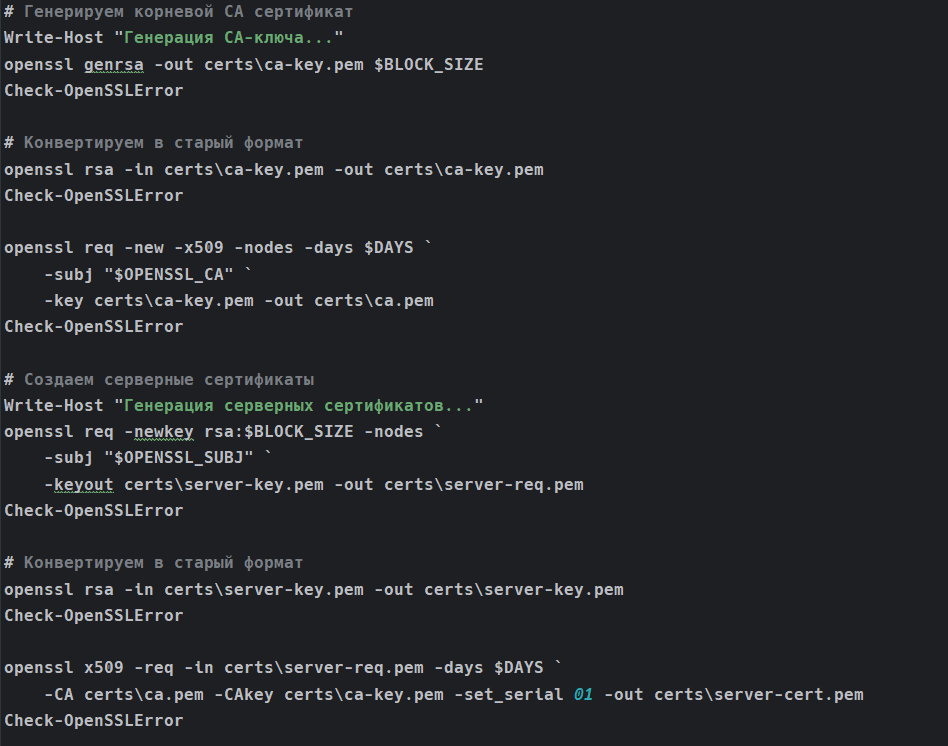


Рисунок 1 – скрипт для генерации сертификатов для MySQL

В результате выполнения скрипта получился такой вот набор файлов, которые в дальнейшем будут использоваться для настройки MySQL в Dcoker контейнере.

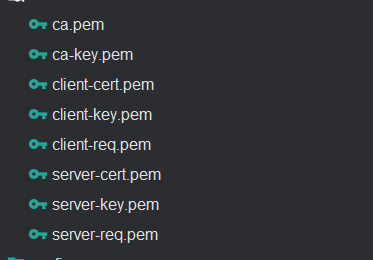


Рисунок 2 – результат выполнения скрипта

Чтобы запустить базу данных MySQL используется система контейнеризации – Docker. Была написана такая конфигурация, которая позволяет запустить MySQL, настроив SSL подключение. Результат представлен на рисунке 3.

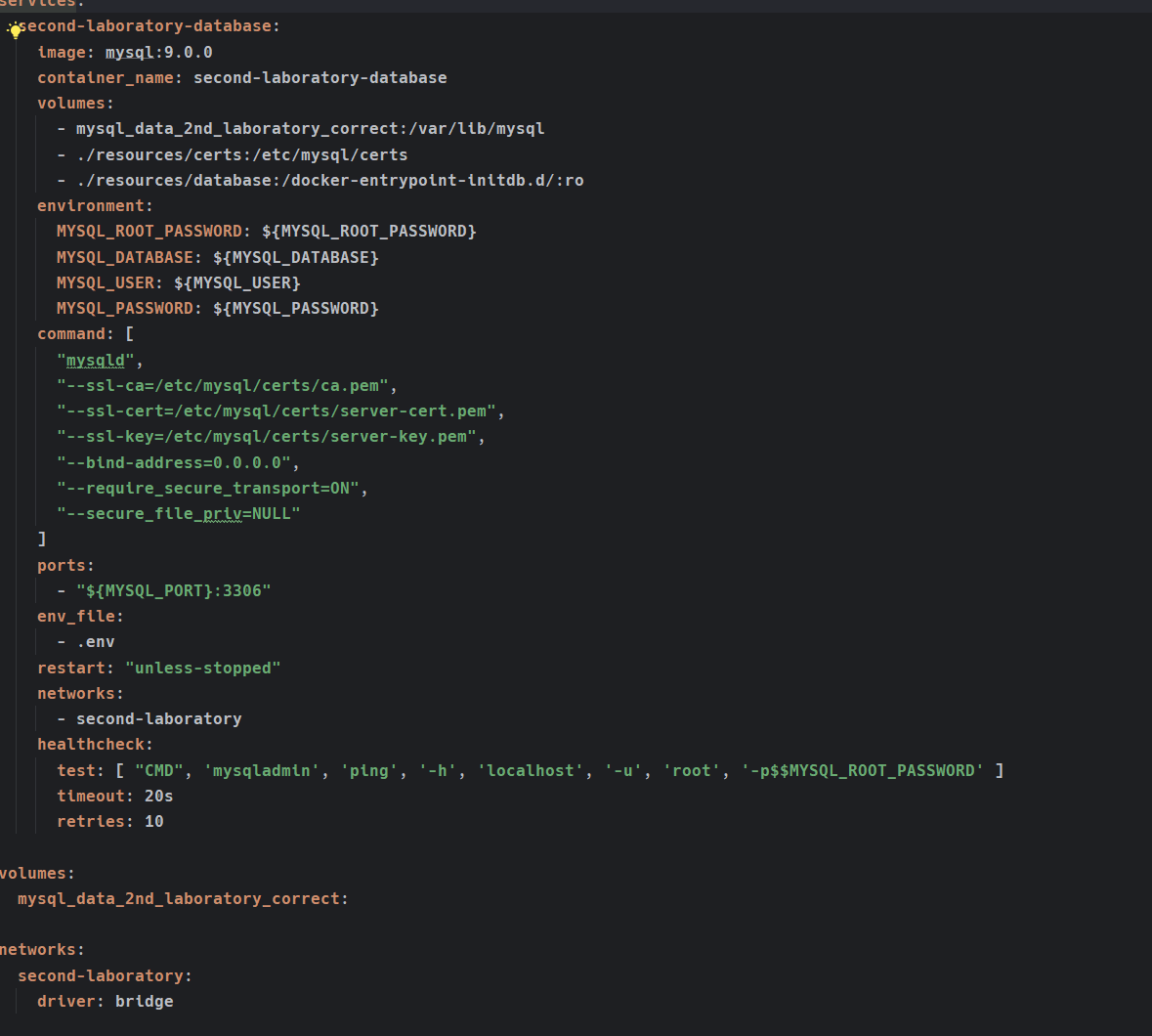


Рисунок 3 – docker compose для MySQL с SSL подключением

Попробуем теперь подключиться к MySQL, введя корректные сертификаты. Результат представлен на риснуке 4. Здесь я взял те же самые сертификаты, что находятся в контейнере, чтобы не мучиться с прокидыванием файла из хоста в команду для Docker.

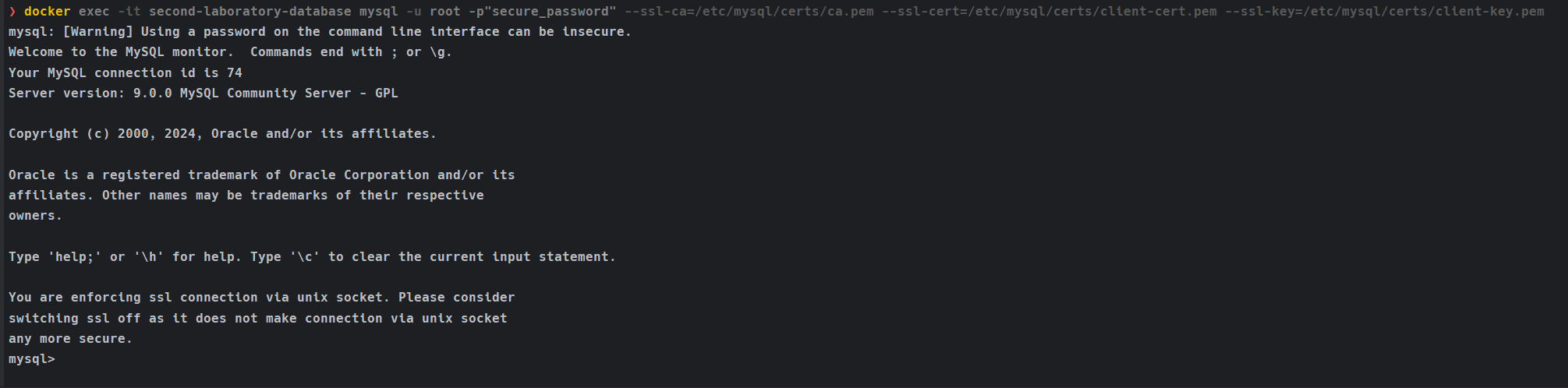


Рисунок 4 – подключение к MySQL с помощью корректных SSL сертификатов

Попробуем точно удостовериться, что у нас используется SSL, подключение, для этого я буду использовать команду “status”, которая отображает текущие характеристики сессии. Результат представлен на рисунке 5.

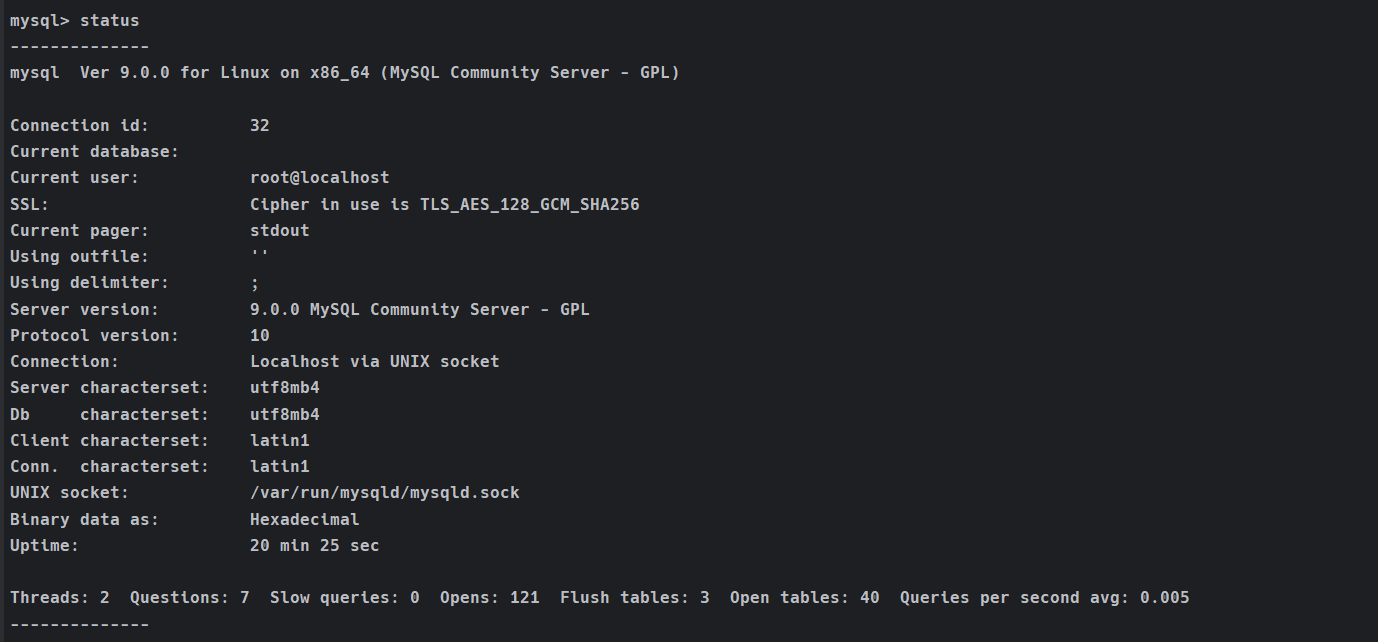


Рисунок 5 – проверка, что сессия использует SSL подключение

Теперь подключимся к конкретной базе данных – mydb и просмотрим таблицу users, которая была создана с помощью SQLAlchemy. Результат отображения таблицы представлен на рисунке 6.

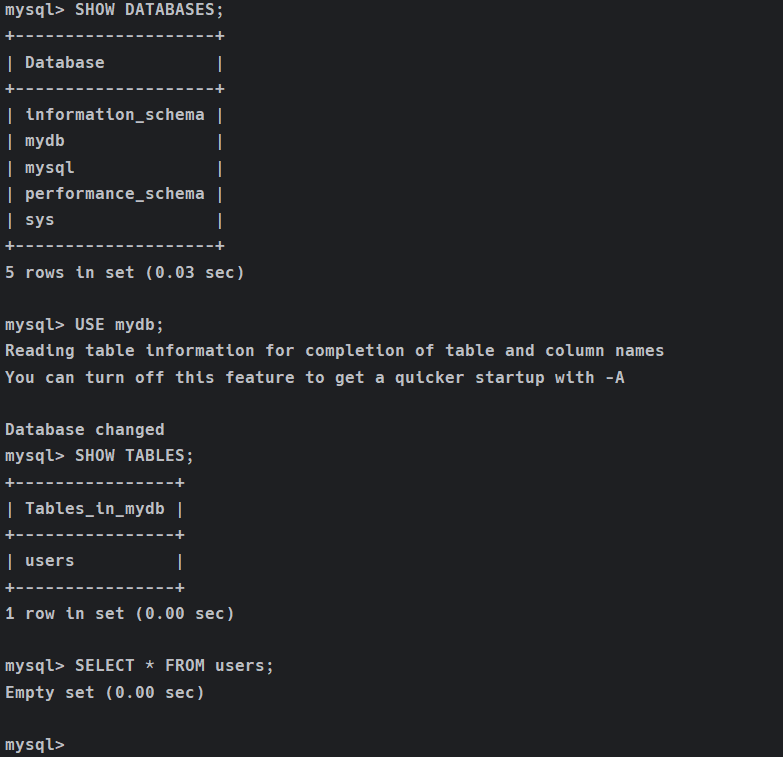


Рисунок 6 – отображение таблицы users

При выборе неправильного сертификата бросается ошибка. В моем случае я вместо клиентского сертификата отправил server-key.pem. Мне выдало ошибку, что нельзя достать приватный ключ. Это вот один из примеров, что нельзя будет.

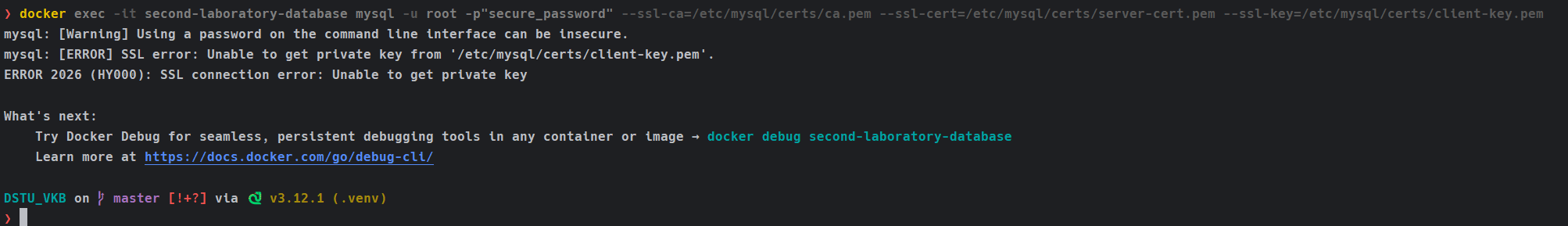


Рисунок 7 – невозможность вычисления приватного ключа при неправильных сертификатах

Теперь приступим к написанию прикладного ПО для работы с нашей базой данных – MySQL с SSL подключением. В качестве языка программирования был выбран язык Python. Проект собран при помощи сборщика – poetry. Основные библиотеки, которые используются: pymysql, dishka, sqlalchemy, fastapi, pydantic.

Была выявлена такая массовая проблема, что большинство асинхронных драйверов под язык Python имеют ряд проблем при работе SSL. Данный эффект обусловлен тем, что MySQL не пользуется спросом на рынке, так как его вытяснил PostgreSQL из-за чего большинство OpenSource разработчиков забросили поддержку драйверов под MySQL. Был найден единственный стабильный драйвер для работы, но, к сожалению, он синхронный, что портит всю асинхронность.

Файл конфигурации проекта представлен на рисунке 8. Был использован сборщик Poetry, который предоставляет удобный интерфейс для взаимодействия с установщиком Python. Его основные преимущества: контроль версий на конфликт, гибкость настроек, возможность разделения окружения.



Рисунок 8 – конфигурация проекта с помощью Poetry

Для написания проекта был использован архитектурный паттерн DDD. Структура проекта представлена на рисунке 9. Его основная концепция заключается в том, что код подстраивается под бизнес-требования предметной области, создается единый язык для взаимодействия с бизнесом. Благодаря такому подходу, код становится предметно ориентированным. Основные сущности в DDD: доменные сущности, агрегаты, доменные сервисы, репозитории, Unit Of Work, Value Objects, интеракторы.

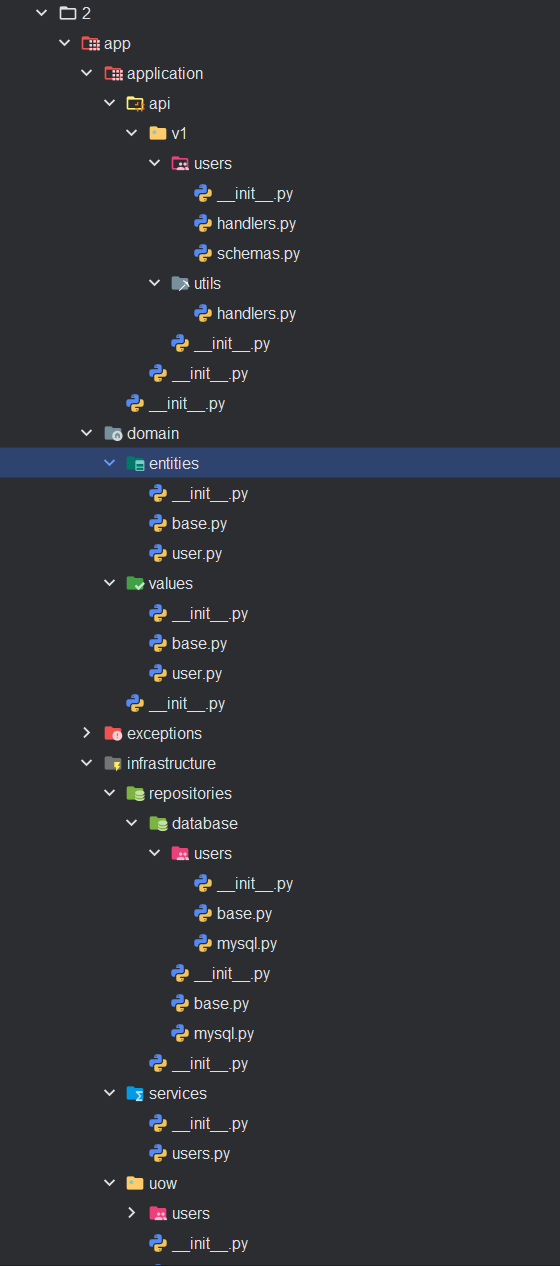


Рисунок 9 – часть структуры проекта

На рисунке 10 представлен env файл, в котором хранятся секреты для нашего проекта.

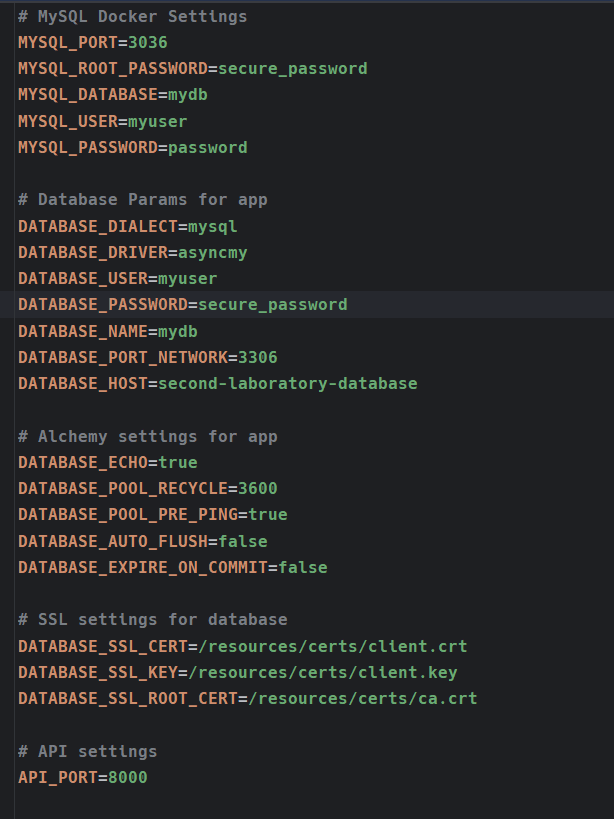


Рисунок 10 – параметры ПО для работы

Продемонстрируем процесс работы приложения. В моем случае используется интерактивная документация Swagger, которая предоставляет UI интерфейс для работы с backend приложением. Чтобы открыть документацию Swagger нужно в браузере ввести адресс: “localhost:8000/docs”. Интерфейс OpenAPI Swagger представлен на рисунке 11.

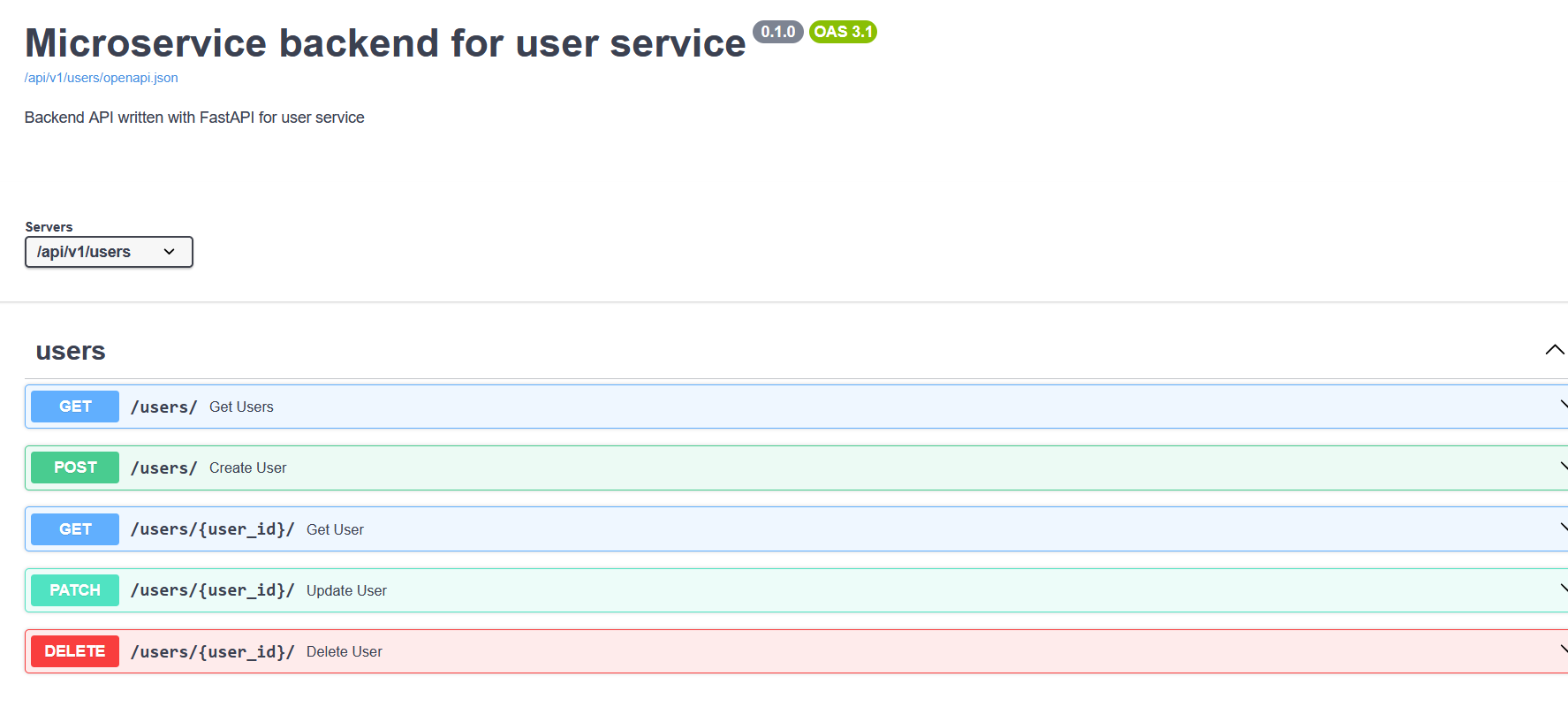


Рисунок 11 – интерфейс Swagger для работы с сервисом

Попробуем просто создать пользователя, используя http запрос: post “localhost:8000/users/”. В нашем случае получилось то, что представлено на рисунке 12 ниже.

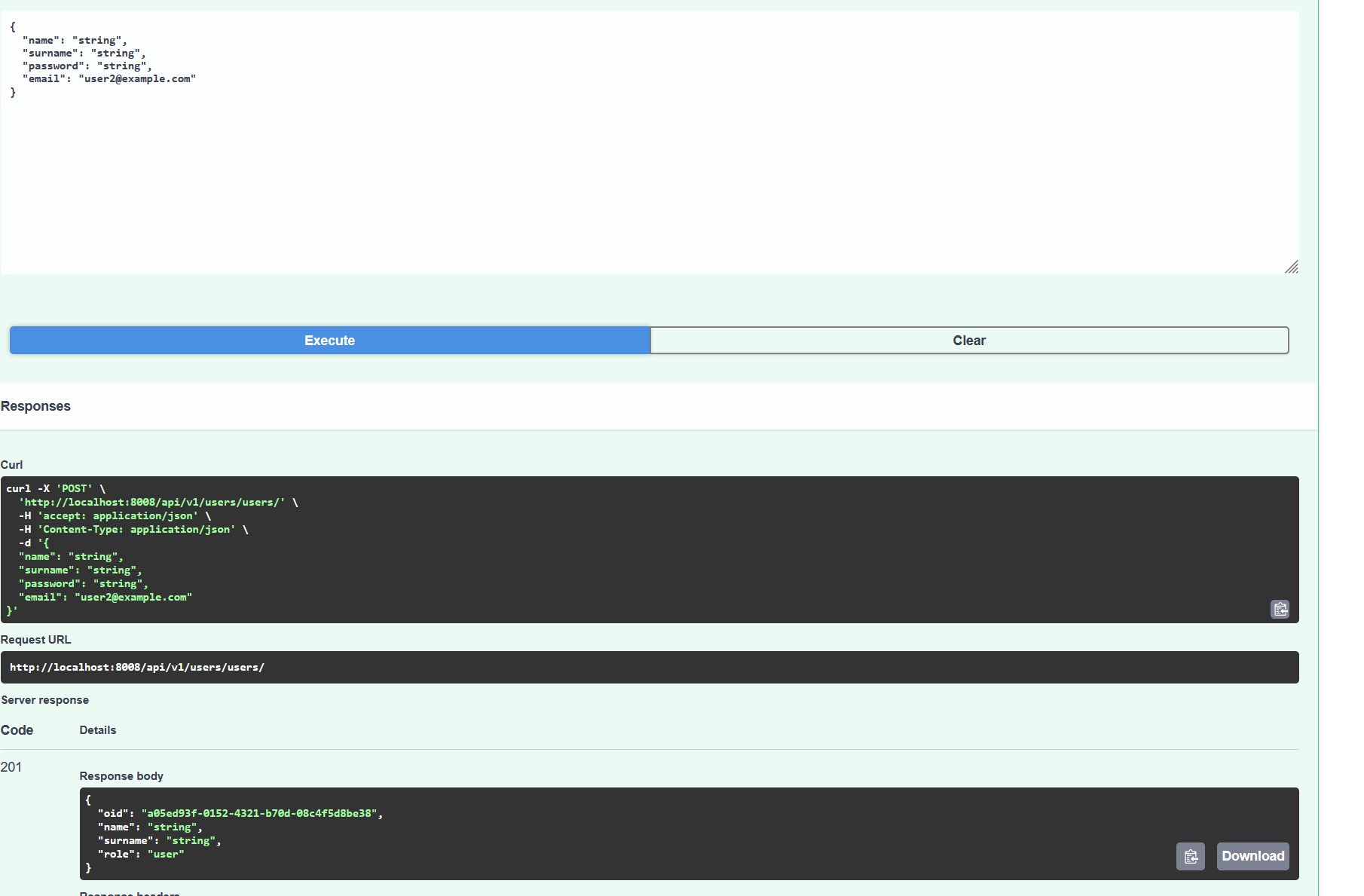


Рисунок 12 – создание пользователя

Для примера был создан ещё один пользователь до этого. Попробуем теперь просмотреть всех пользователей в системе. Результат представлен на рисунке 13.

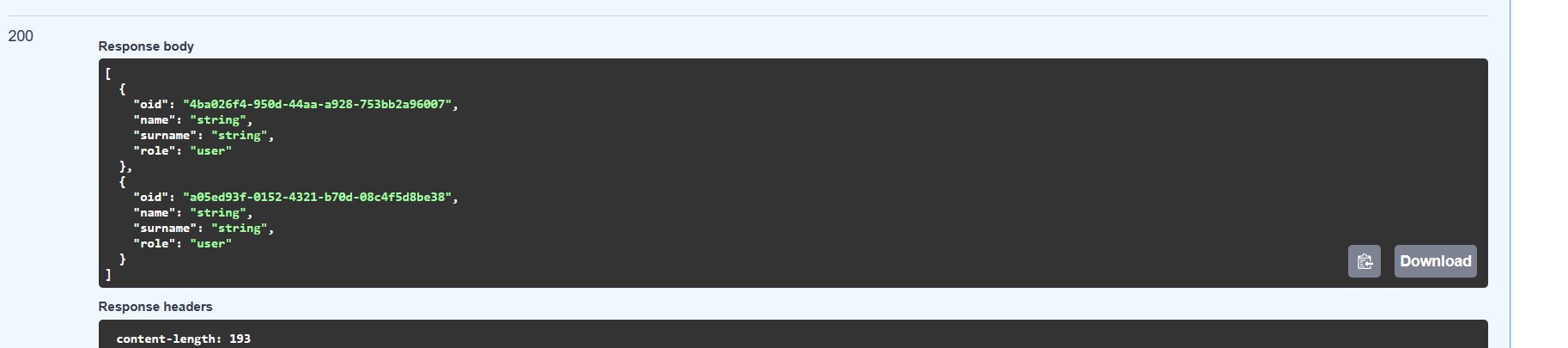


Рисунок 13 – просмотр всех пользователей в системе

Для редактирования пользователя есть соответствующий HTTP Patch метод, который позволяет обновлять любые поля у пользователя. Результат данного запроса представлен на рисунке 14.

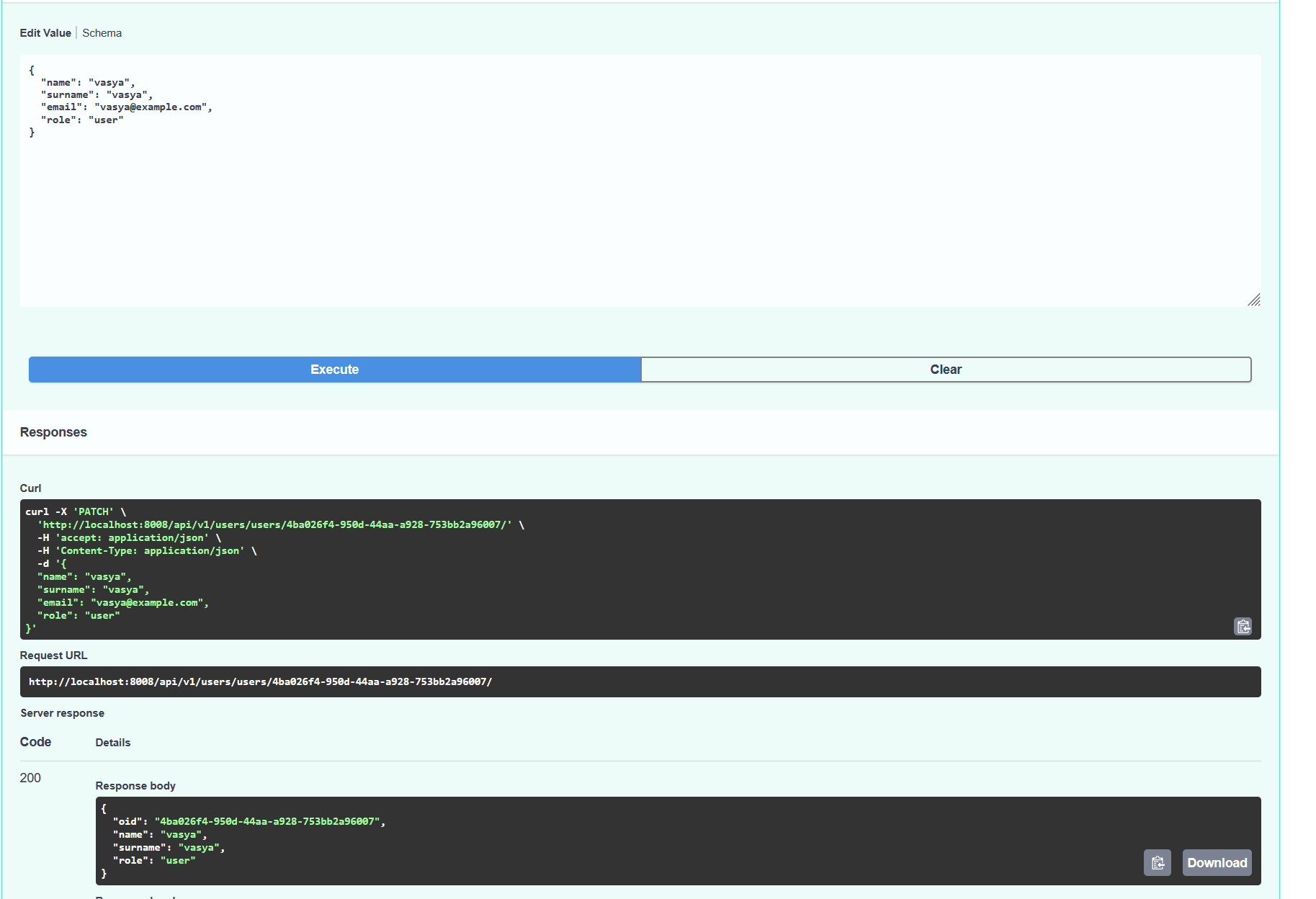


Рисунок 14 – обновление пользователя через HTTP Patch