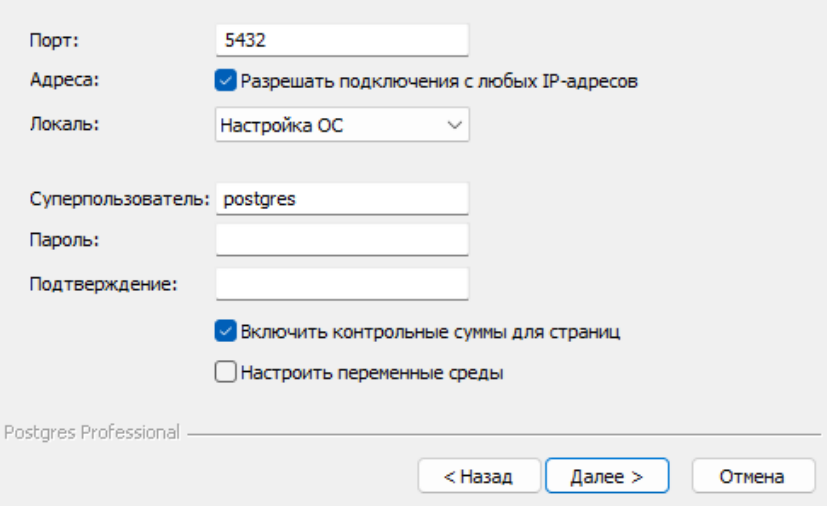
Настройка PostgreSQL и клонирование репозитория

Настройка PostgreSQL

1. Установить PostgreSQL (использовал пока PostgreSQL 16.10)

https://postgrespro.ru/windows



- **Порт**: можно оставить таким же.
- **Настройка ОС**: выбрать ru-RU.
- **Суперпользователь**: можно оставить **postgres**.
- **Пароль**: свой на своем компьютере.

2. Проверить PostgreSQL

- Проверка версии PostgreSQL

  ```
  postgres --version
  ```

  или

  ```
  psql --version
  ```

  или с полным путем

  ```
  "C:\Program Files\PostgreSQL\16\bin\psql.exe" –version
  ```

- Проверка службы

  ```
  sc query postgresql-x64-16
  ```

  Если не запущена, запустить

  ```
  net start postgresql-x64-16
  ```

3. Создать базу данных

- Создание базы данных, например, **edulocal,** на своем компьютере

  ```
  psql -U postgres -c "CREATE DATABASE edulocal;"
  ```

  Эта команда не зависит от текущей директории.

  В дальнейшем схема **edudb** с таблицами и данными будет восстановлена в базу данных **edulocal** из дампа, который в Git-репозитории.

Клонирование репозитория

1. Проверка Git на ПК

- Проверить, Git установлен или нет:

```
git --version
```

- Настроить Git (если еще не сделано)

```
git config --global user.name "Ваше Имя"
git config --global user.email "ваш_email@server.com"
```

Примечание:

На чужом ПК настройку **user.name и user.email** можно не делать.
Эти команды не обязательны для клонирования, просмотра кода и работы с кодом.
Но они нужны, если потом планируется делать коммиты и пушить изменения в Git-репозиторий.

2. Клонировать репозиторий и восстановить дамп

- Перейти в папку, где будет сохранен (создан) проект

```
cd C:\Projects
```

- Клонировать репозиторий в текущую папку

```
git clone https://github.com/gucodegit/checking-code.git
```

Если ошибка типа:

```
Cloning into 'checking-code'... fatal: unable to access
'https://github.com/gucodegit/checking-code.git/': SSL certificate problem:
self signed certificate in certificate chain
```

То команды:

```
# Временно отключить SSL проверку
git config --global http.sslVerify false
# Клонировать проект
git clone https://github.com/gucodegit/checking-code.git
# Вернуть настройки безопасности
git config --global http.sslVerify true
```

Будет создана папка **checking-code** и все вложенные папки с файлами.

- Перейти в папку проекта

```
cd checking-code
```

- Развернуть (восстановить) дамп схемы **edudb** из подпапки **edu-db**

```
psql -U postgres -d edulocal -f edu-db\edudb_backup251025.sql
```

Можно проверить данные из таблицы **edudb.groups**

```
psql -U postgres -d edulocal -c "SELECT group_name FROM edudb.groups;"
```

3. Настроить проект для работы с базой данных вуза (в терминале VS Code)

- Открыть проект в VS Code из текущей папки **checking-code**

  ```
  code .
  ```

- В терминале VS Code перейти в папку **edubackend**

  ```
  cd edubackend
  ```

- Создать виртуальное окружение

  ```
  python -m venv venv
  ```

- Активировать виртуальное окружение

  ```
  .\venv\Scripts\activate
  ```

  Если возникнет ошибка, то пробовать команду:

  ```
  Set-ExecutionPolicy -ExecutionPolicy Bypass -Scope Process
  ```

  И потом команда активации.

- Установить зависимости (требуемые пакеты уже указаны в файле requirements.txt)

  ```
  pip install -r requirements.txt
  ```

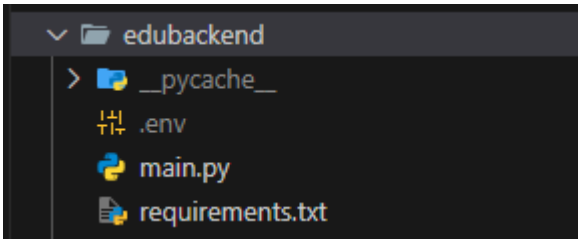
Если проблемы на этом этапе, то можно попробовать:

1. Закрыть и открыть заново терминал.
2. Или выбрать вручную интерпретатор:
   - Ctrl+Shift+P
   - Python: Select Interpreter
   - Выбрать: ./edubackend /venv/Scripts/python.exe

- Настроить переменные окружения

  В папке **edubackend** создать файл **.env:**

  ```
  DB_HOST=localhost
  DB_NAME=edulocal
  DB_USER=postgres
  DB_PASSWORD=ваш_пароль
  DB_PORT=5432
  ```



4. Запуск приложения

- **Запустить бэкенд**

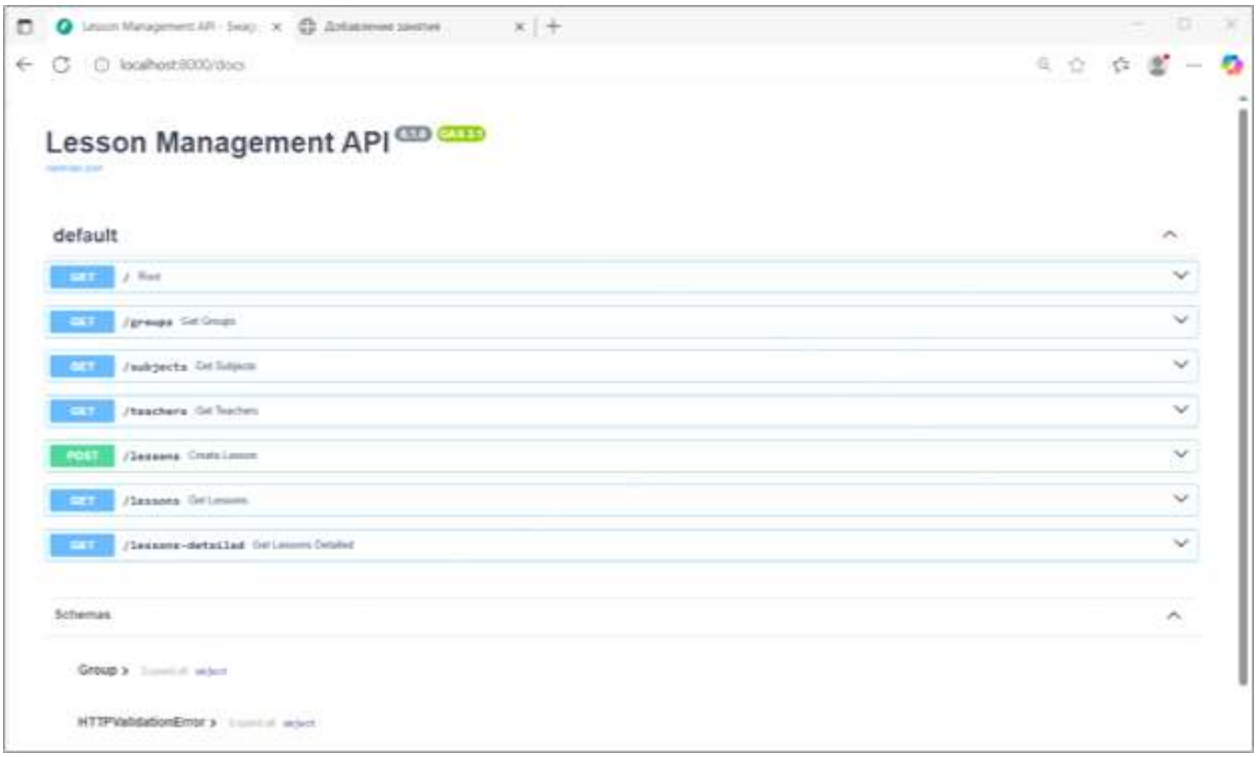
```
cd edubackend
uvicorn main:app --reload --host 0.0.0.0 --port 8000
```

```
(venv) PS C:\WorkDocs\checking-code> cd .\edubackend\
(venv) PS C:\WorkDocs\checking-code\edubackend> uvicorn main:app --reload --host 0.0.0.0 --port 8000
INFO:     Will watch for changes in these directories: ['C:\\WorkDocs\\checking-code\\edubackend']
INFO:     Uvicorn running on http://0.0.0.0:8000 (Press CTRL+C to quit)
INFO:     Started reloader process [4140] using WatchFiles
INFO:     Started server process [2384]
INFO:     Waiting for application startup.
INFO:     Application startup complete.
```

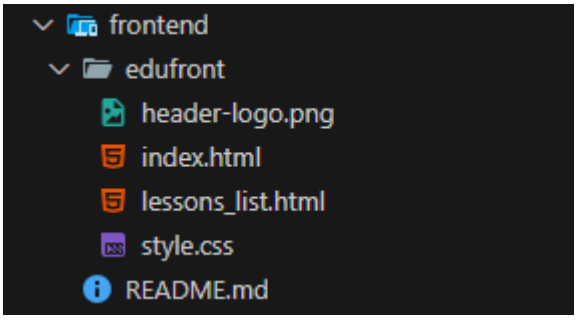
Бэкенд должен быть доступен по адресу:

http://localhost:8000

http://localhost:8000/docs

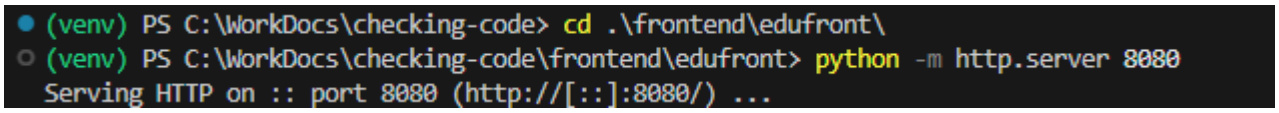


- **Запустить фронтенд**



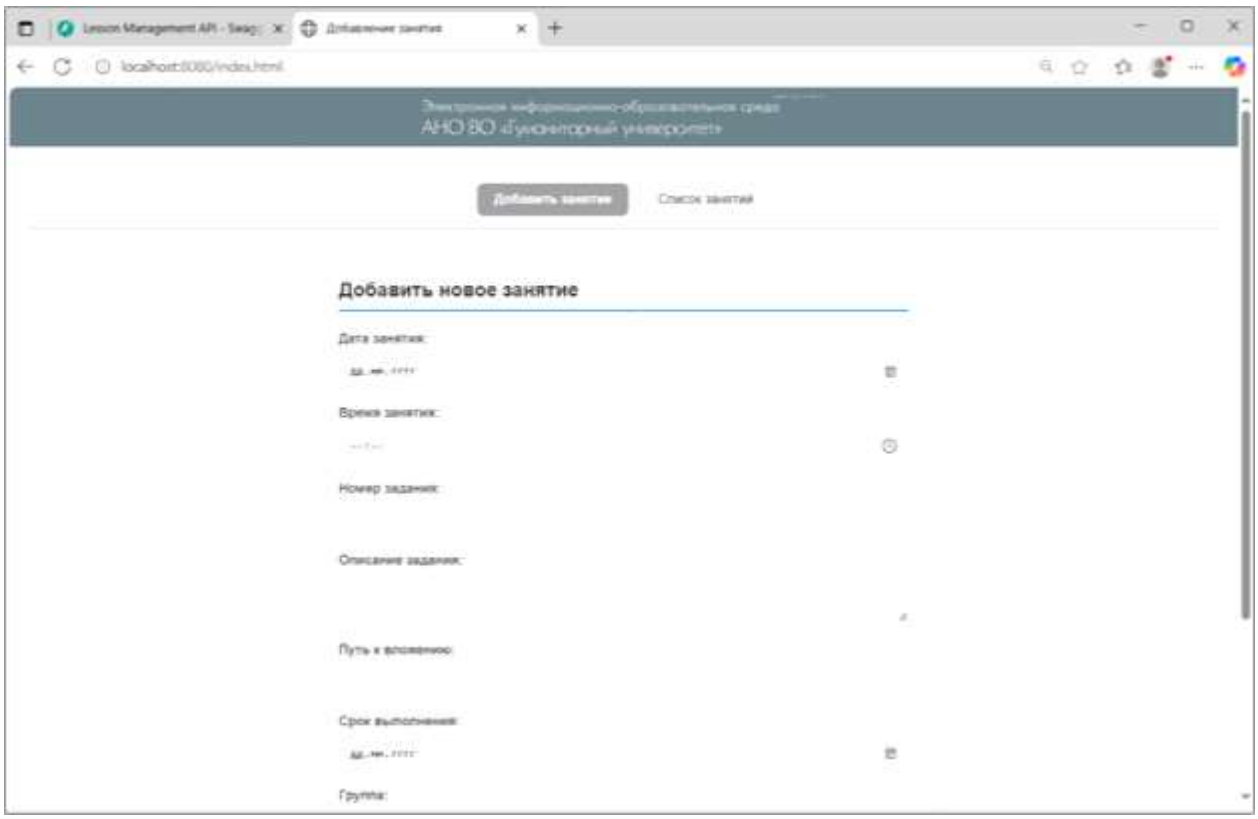
Сначала создать второй терминал, потом команды

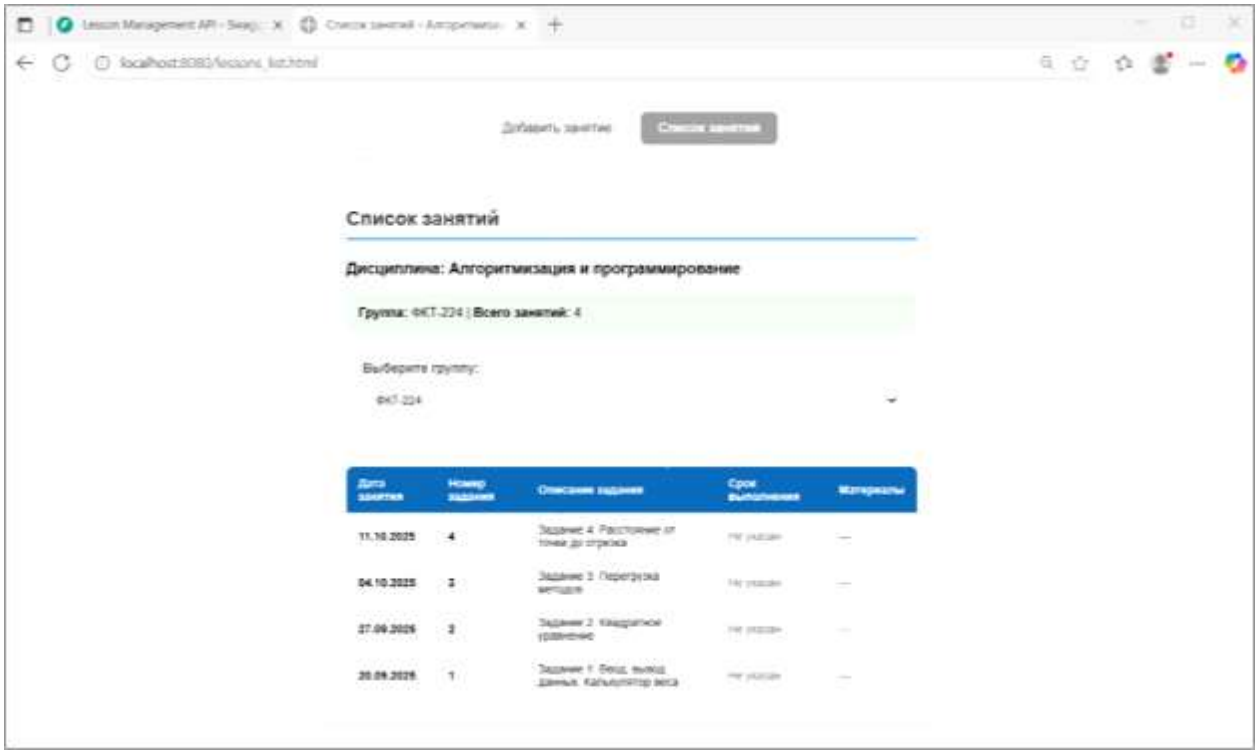
```
cd frontend\edufront
python -m http.server 8080
```



Должен быть доступен по адресу:

http://localhost:8080




Добавить занятие
Список занятий
Список занятий
Дисциплина: Алгоритмизация и программирование
Группа: ФКТ-224 | Всего занятий: 4
Выберите группу:
ФКТ-224
Дата занятия
Номер задания
Описание задания
Срок выполнения
Материалы
11.10.2025
4
Задание 4: Расстояние от точки до отрезка
Не указан
—
04.10.2025
3
Задание 3: Перегрузка методов
Не указан
—
27.09.2025
2
Задание 2: Квадратное уравнение
Не указан
—
20.09.2025
1
Задание 1: Ввод, вывод данных. Калькулятор веса
Не указан
—

5. Предварительная настройка Git для локальной работы

- Проверить подключение к удаленному репозиторию

  ```
  git remote -v
  ```

  Должно быть:

  ```
  origin  https://github.com/gucodegit/checking-code.git (fetch)
  origin  https://github.com/gucodegit/checking-code.git (push)
  ```

Примечание:

При клонировании **git clone** (команда была выполнена ранее) автоматически настраивается **remote** с именем **origin.**

- Создать свою ветку для первой задачи, например:

  ```
  git checkout -b feature/checking/petrov/algorithm-optimization
  ```

  Эта команда создает новую ветку и переключает на нее.

  Команду git можно выполнять из любой папки внутри репозитория, но обычно выполняют в корне проекта, потому что там находится папка .git.

- Проверить ветки и узнать активную (отмечена звездочкой *)

  ```
  git branch
  ```

Рекомендуемые типы веток:

- feature/ – новая функциональность.
- bugfix/ – исправление ошибок.
- hotfix/ – срочные исправления.
- refactor/ – рефакторинг кода.
- docs/ – работа с документацией.

Рекомендуемое именование веток:

- тип/модуль/студент/краткое-описание