

Цель работы:

Освоить фундаментальные концепции и базовые операции Docker: создание образов, запуск контейнеров, управление ими, работа с сетями и томами. На практике закрепить навыки, запустив изолированную базу данных PostgreSQL и подключившись к ней извне.

Задачи:

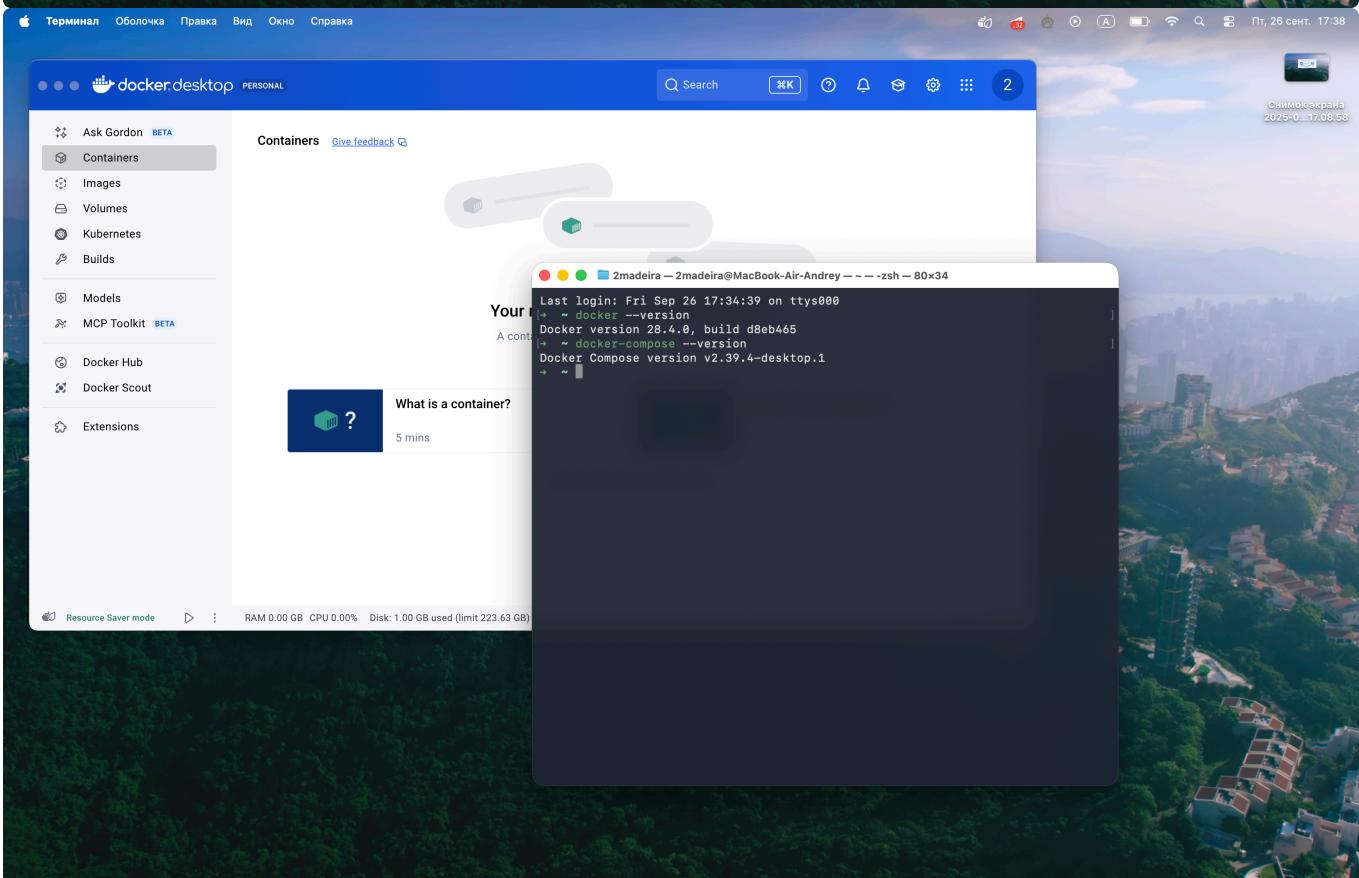
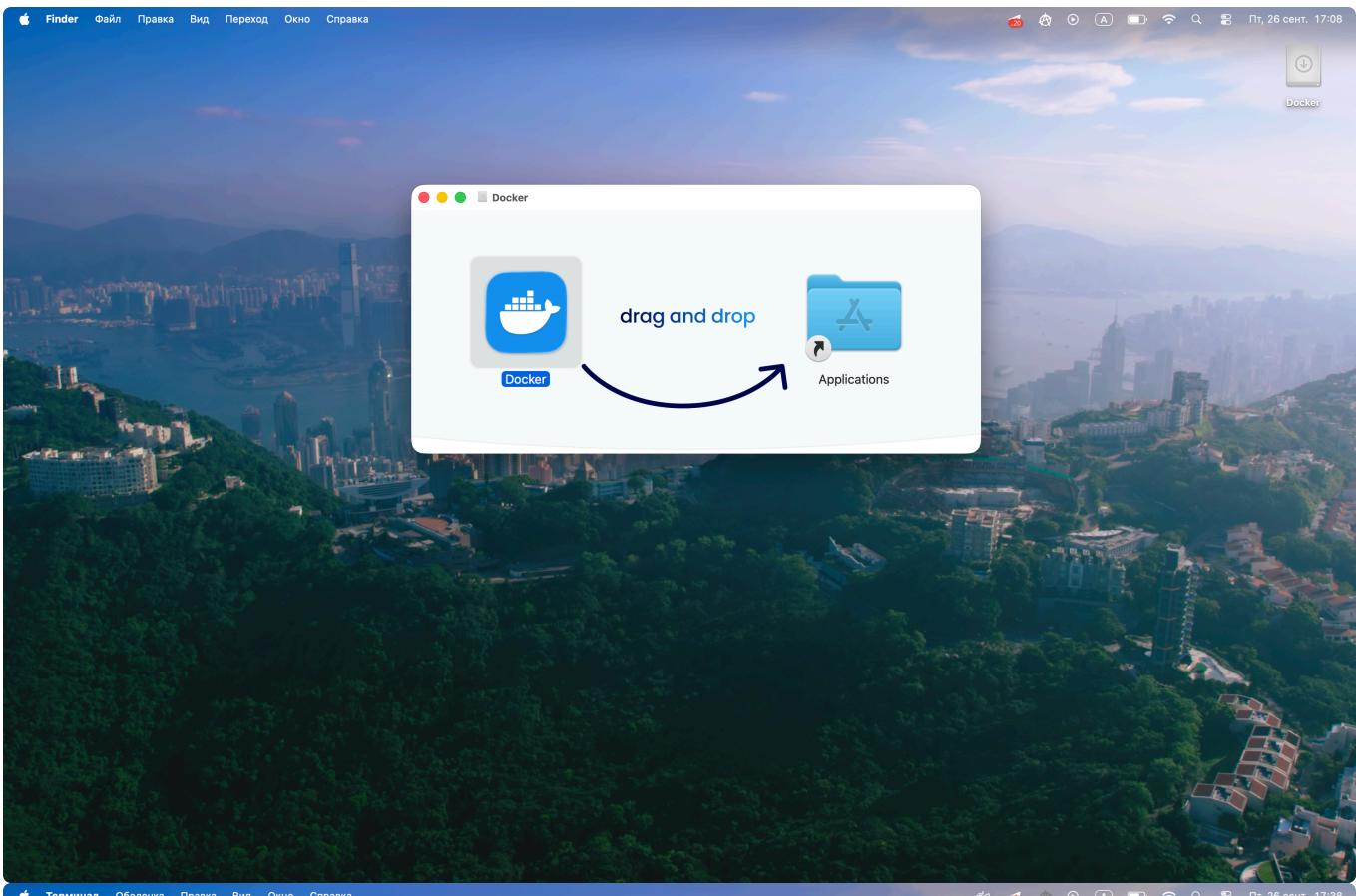
1. Установить и проверить работу Docker.
2. Изучить базовые команды Docker.
3. Запустить контейнер с PostgreSQL в изолированном режиме.
4. Запустить контейнер с pgAdmin и подключить его к контейнеру с БД через сеть Docker.
5. Подключиться к БД из pgAdmin, создать схему и выполнить запросы.
6. Обеспечить сохранность данных БД с помощью томов Docker.

Контрольный срок сдачи: 30 сентября

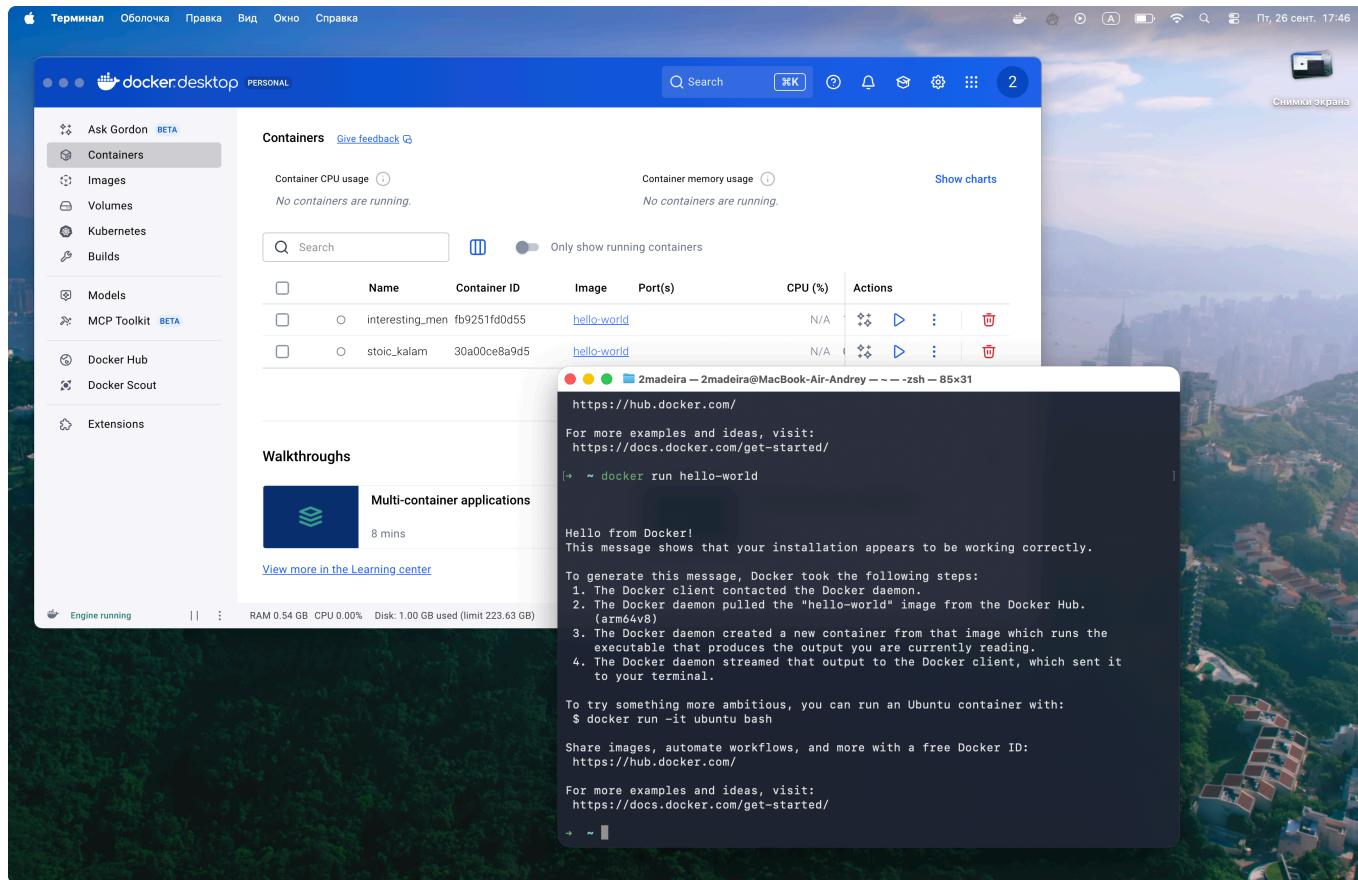
Ссылку на github присыпаем в личные сообщения

Отчет:

1. Докер устанавливал через образ Docker desktop, вроде можно через homebrew, но решил, что мне пока проще будет так.



Проверил успешную установку запуском тестового контейнера



2. Список базовых команд Docker

Просто указал какие то базовые команды, далее буду использовать некоторые. Выше, при запуске тестового контейнера мне понадобилось тянуть контейнер через docker

pull. Логи пока смотрел только через дашборд docker desktop.

Команда	Описание	Пример использования
<code>docker run [образ]</code>	Запуск нового контейнера из образа	<code>docker run hello-world</code>
<code>docker ps</code>	Просмотр запущенных контейнеров	<code>docker ps</code>
<code>docker ps -a</code>	Просмотр всех контейнеров (в том числе остановленных)	<code>docker ps -a</code>
<code>'docker stop [id]</code> имя`		Остановка работающего контейнера
<code>'docker start [id]</code> имя`		Запуск остановленного контейнера
<code>'docker rm [id]</code> имя`		Удаление контейнера
<code>docker images</code>	Список загруженных образов	<code>docker images</code>
<code>docker rmi [образ]</code>	Удаление образа	<code>docker rmi hello-world</code>
<code>docker pull [образ]</code>	Скачивание образа из Docker Hub	<code>docker pull postgres</code>
<code>'docker exec -it [id]</code> имя] bash`		Вход внутрь работающего контейнера (терминал)
<code>'docker logs [id]</code> имя`		Просмотр логов контейнера

3. Запустить контейнер с PostgreSQL в изолированном режиме.

Изолированный режим это базовый режим работы контейнера, поэтому выполняю простые команды и запускаю. Выбрал версию postgres 17.6, вроде как, считается стабильной, но не уверен здесь до конца, не было опыта работы с ней, основываюсь только на отзывах в интернете - далее я также буду выбирать конкретную версию образа для того, чтобы была возможность легче бороться с проблемами несовместимости версий (такой совет подчерпнул из лекции).

The screenshot shows the VS Code interface with a dark theme. On the left, there's a file tree with a folder named 'APP_DEV' containing '.env', 'docker-compose.yml', and 'Dockerfile'. The 'Dockerfile' tab is active, displaying the following code:

```
# Подгружаем образ postgres
FROM postgres:17.6-bookworm
#
# Устанавливаем рабочую директорию
WORKDIR /app
#
# Устанавливаем переменные окружения - плохая практика хранения паролей
ENV POSTGRES_USER=admin
ENV POSTGRES_PASSWORD=admin
ENV POSTGRES_DB=postgres
#
# Копируем скрипт инициализации базы данных
COPY . .
#
# Открываем порт 5432
EXPOSE 5432
#
# Запускаем postgres
CMD ["postgres"]
```

Below the code editor is a terminal window titled 'Terminal' with the following command history:

- APP_DEV docker ps
- CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
- 09e321e5b89 first_lab_postgres "docker-entrypoint.s..." 31 seconds ago Up 31 seconds 5432/tcp practical_shtern
- APP_DEV docker ps -a
- CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
- 09e321e5b89 first_lab_postgres "docker-entrypoint.s..." 54 seconds ago Up 54 seconds 5432/tcp practical_shtern
- * APP_DEV docker stop 09e321e5b89
- APP_DEV docker ps
- CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
- APP_DEV []

The status bar at the bottom indicates 'Cursor Tab' and 'Ln 7, Col 72'.

Указал переменные в .env файле – значения стандартные:

POSTGRES_USER=admin
POSTGRES_PASSWORD=admin
POSTGRES_DB=postgres

The screenshot shows the VS Code interface with a dark theme. On the left, there's a file tree with a folder named 'APP_DEV' containing '.env', 'docker-compose.yml', and 'Dockerfile'. The 'docker-compose.yml' tab is active, displaying the following configuration:

```
version: '3.8'
services:
  db:
    image: postgres:17.6-bookworm
    environment:
      - POSTGRES_USER=${POSTGRES_USER}
      - POSTGRES_PASSWORD=${POSTGRES_PASSWORD}
      - POSTGRES_DB=${POSTGRES_DB}
    ports:
      - "5432:5432"
    volumes:
      - db_data:/var/lib/postgresql/data
```

Below the code editor is a terminal window titled 'Terminal' with the same command history as the previous screenshot.

The status bar at the bottom indicates 'Cursor Tab' and 'Ln 16, Col 11'.

Поднял контейнер, вывел список контейнеров в консоль

```
APP_DEV docker ps
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
0a9e321e5b89 first_lab_postgres "docker-entrypoint.s..." 31 seconds ago Up 31 seconds 5432/tcp practical_shtern
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
0a9e321e5b89 first_lab_postgres "docker-entrypoint.s..." 54 seconds ago Up 54 seconds 5432/tcp practical_shtern
APP_DEV docker stop 0a9e321e5b89
Error response from daemon: No such container: first_lab_postgres
APP_DEV docker rm 0a9e321e5b89
APP_DEV docker ps
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
zsh: no matches found: first_lab_postgres
APP_DEV docker ps -a
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
0a9e321e5b89 first_lab_postgres "docker-entrypoint.s..." 9 minutes ago Exited (0) 8 minutes ago
APP_DEV docker rm 0a9e321e5b89
APP_DEV docker ps -a
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
APP_DEV

```



```
APP_DEV docker ps -a
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
0a9e321e5b89 first_lab_postgres "docker-entrypoint.s..." 31 seconds ago Up 31 seconds 5432/tcp practical_shtern
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
0a9e321e5b89 first_lab_postgres "docker-entrypoint.s..." 54 seconds ago Up 54 seconds 5432/tcp practical_shtern
APP_DEV docker stop 0a9e321e5b89
APP_DEV docker rm 0a9e321e5b89
APP_DEV docker ps
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
zsh: no matches found: first_lab_postgres
APP_DEV docker ps -a
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
APP_DEV

```

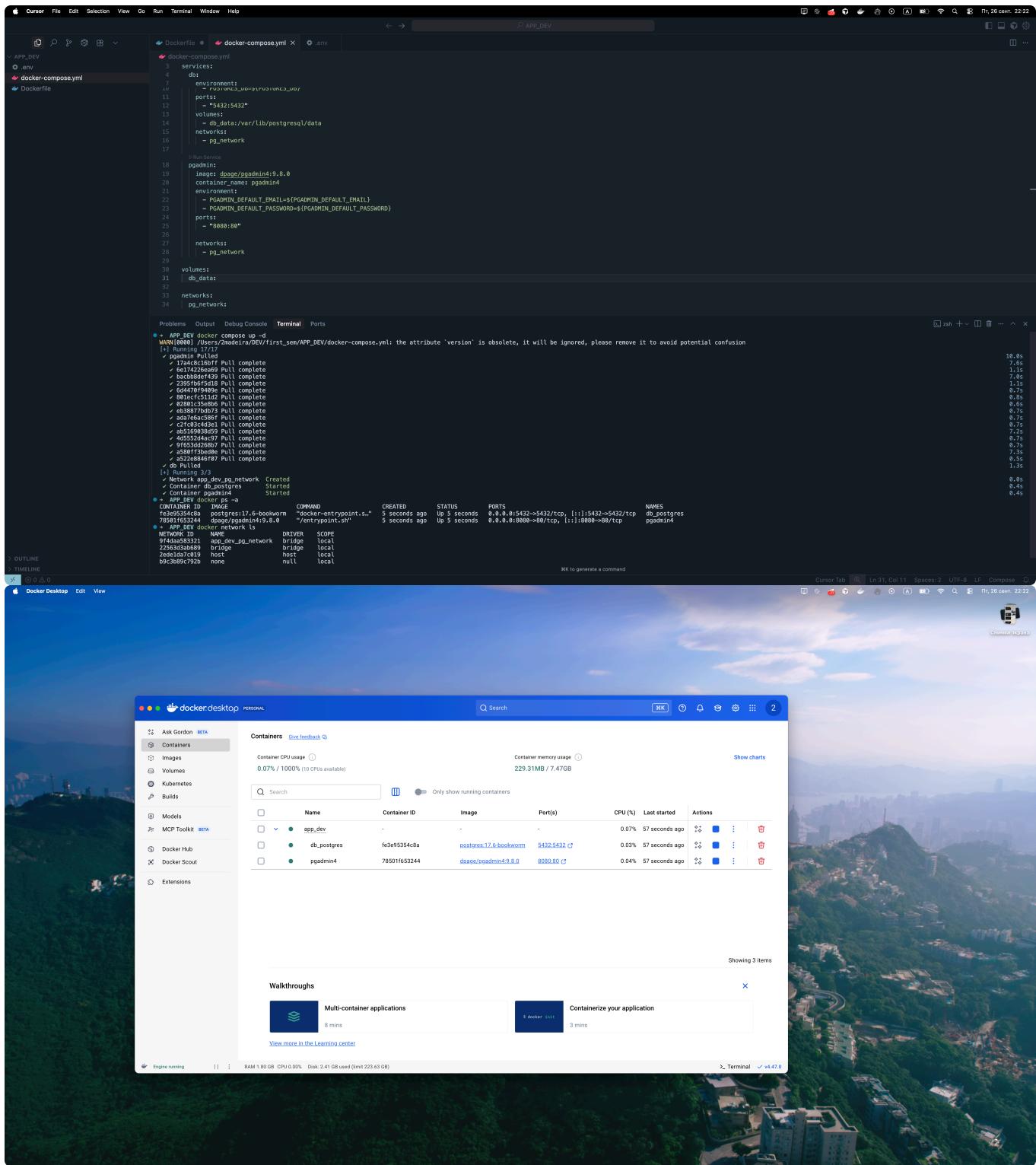


```
APP_DEV docker ps -a
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
0a9e321e5b89 first_lab_postgres "docker-entrypoint.s..." 31 seconds ago Up 31 seconds 5432/tcp practical_shtern
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
0a9e321e5b89 first_lab_postgres "docker-entrypoint.s..." 54 seconds ago Up 54 seconds 5432/tcp practical_shtern
APP_DEV docker stop 0a9e321e5b89
APP_DEV docker rm 0a9e321e5b89
APP_DEV docker ps
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
zsh: no matches found: first_lab_postgres
APP_DEV docker ps -a
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
APP_DEV

```

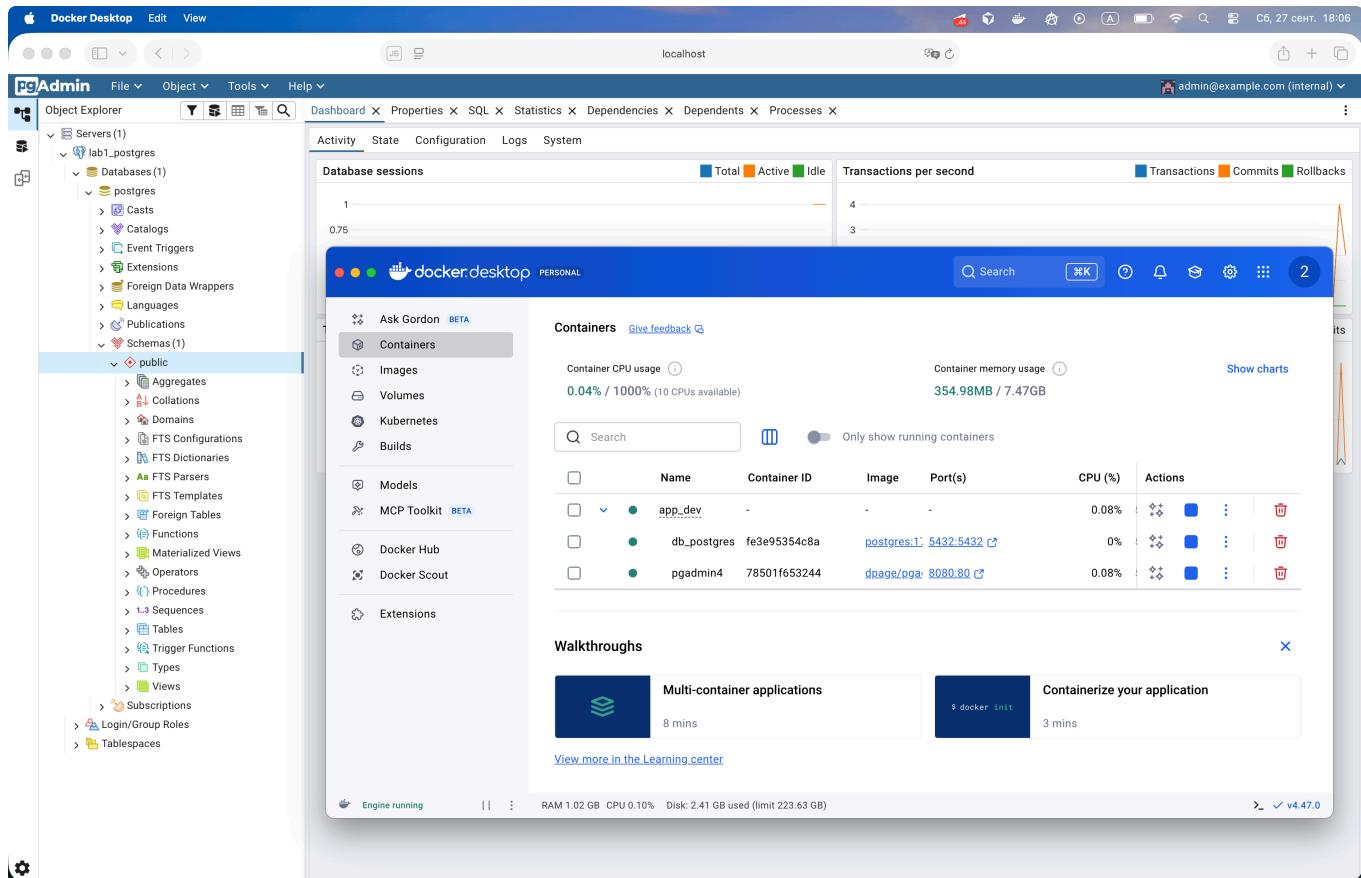
4. Запустить контейнер с pgAdmin и подключить его к контейнеру с БД через сеть Docker.

Далее, я буду использовать оркестрацию с помощью docker compose. Добавил pgadmin, также конкретный образ был выбран, указал переменные для доступа в .env.



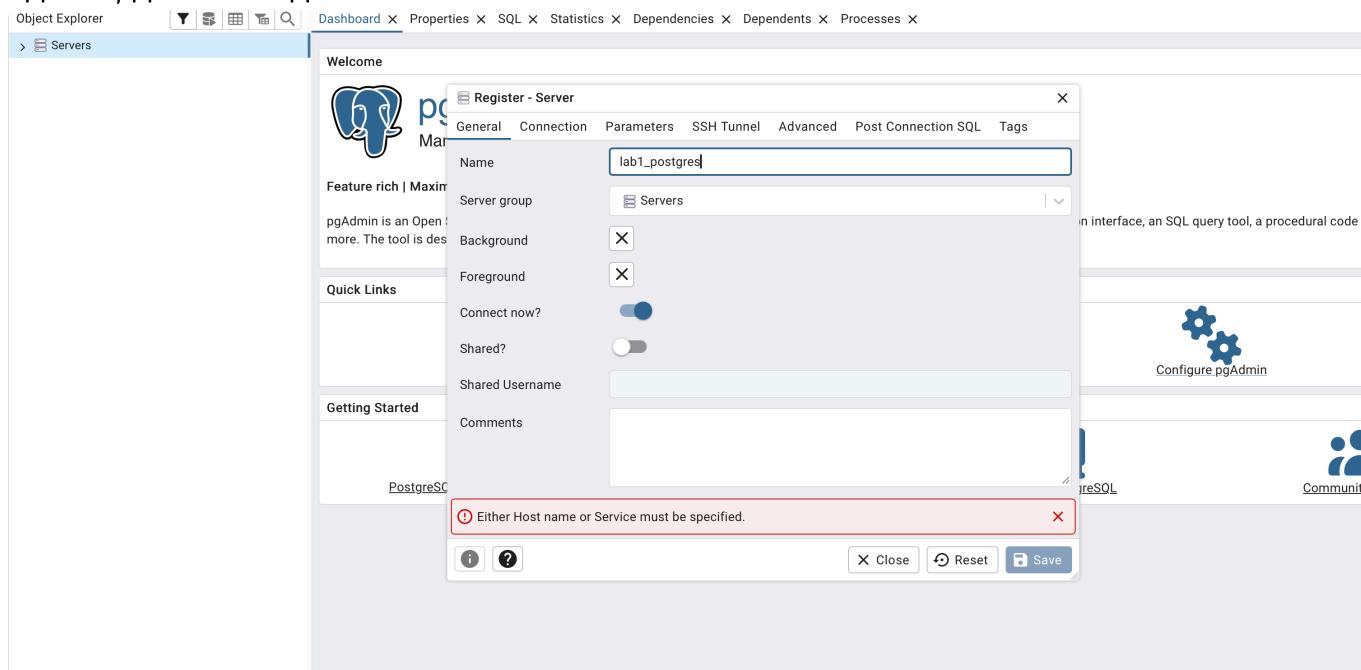
Успешно развернул контейнеры, авторизовался в pgadmin и добавил сервер

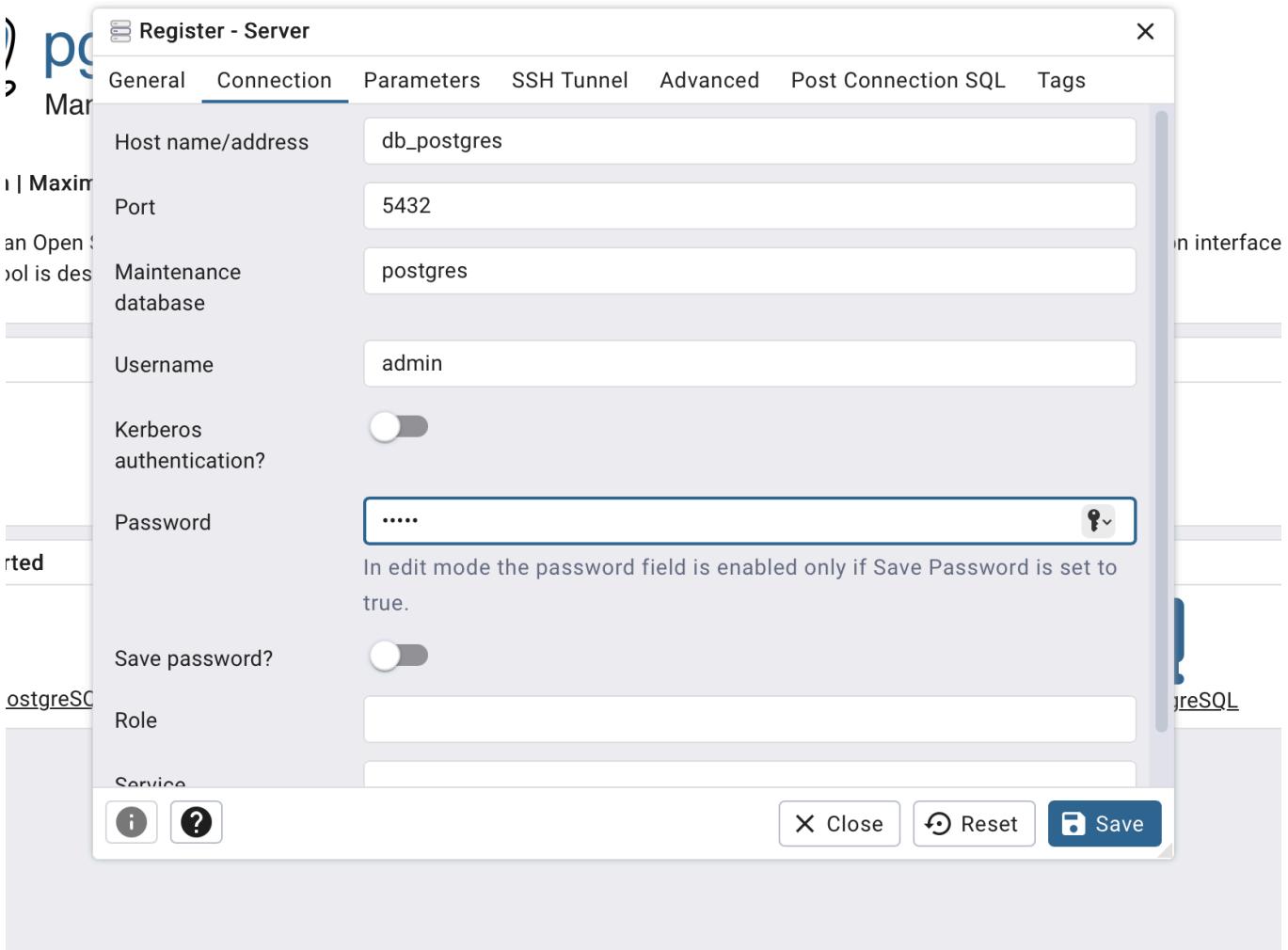
lab1_postgres с моей БД.



5. Подключиться к БД из pgAdmin, создать схему и выполнить запросы.

Ранее, я уже подключился к БД через pgAdmin, но забыл указать скриншоты, как я это сделал, добавлю здесь





Теперь пришла очередь запросов.

В этом пункте, я продемонстрирую то, как создавала запросы через Querry Tool внутри pgAdmin. Запрос состоит из следующих частей:

- создание новой схемы для работы внутри нее
- создание типа данных для моей таблицы (нужен был enum)
- непосредственно создание самой таблицы с нужными полями – ничего лучше таблицы с примерами блюд и сложности их готовки я сейчас не придумал

- запрос на добавление данных в таблицу

```

CREATE SCHEMA IF NOT EXISTS lab1;
CREATE TYPE complexity AS ENUM ('easy', 'normal', 'hard');
CREATE TABLE lab1.dishes (
    id SERIAL PRIMARY KEY,
    dishname VARCHAR(64) NOT NULL,
    cook_complexity complexity NOT NULL
);
INSERT INTO lab1.dishes(dishname, cook_complexity) VALUES
('солянка', 'normal'),
('копченые ребра', 'hard'),
('пельмени', 'normal'),
('чизбургер', 'easy'),
('лазанья', 'hard');

```

Total rows: 5 Query complete 00:00:00.040

Как видно, запрос успешно был выполнен, таблица создана и наполнена.

```

SELECT * FROM lab1.dishes

```

	id [PK] integer	dishname character varying (64)	cook_complexity
1	1	солянка	normal
2	2	копченые ребра	hard
3	3	пельмени	normal
4	4	чизбургер	easy
5	5	лазанья	hard

Total rows: 5 Query complete 00:00:00.076

Пример просто селекта.

The screenshot shows the PgAdmin 4 interface. In the Object Explorer, under the 'Servers' node, there is a 'lab1_postgres' server with a 'postgres' database selected. Inside 'postgres', there is a 'dishes' table. A query is being run in the main pane:

```
1  SELECT dishname
2  FROM lab1.dishes
3  WHERE cook_complexity = 'normal';
```

The results of the query are displayed in the Data Output tab:

dishname
солянка
пельмени

Total rows: 2 Query complete 00:00:00.059

6. Обеспечить сохранность данных БД с помощью томов Docker.

В моем docker-compose.yml я уже указал монтирование отдельного тома для хранения данных из БД. Демонстрирую в терминале, что том существует.

The screenshot shows a terminal window with several tabs at the top: Dockerfile, docker-compose.yml (which is the active tab), init.sql, and .env. Below the tabs, the terminal content is as follows:

```
3   services:
18     pgadmin:
19       container_name: pgadmin4
20       environment:
21         - PGADMIN_DEFAULT_EMAIL=${PGADMIN_DEFAULT_EMAIL}
22         - PGADMIN_DEFAULT_PASSWORD=${PGADMIN_DEFAULT_PASSWORD}
23       ports:
24         - "8080:80"
25
26     networks:
27       - pg_network
28
29
30   volumes:
31     db_data:
32
33   networks:
34     pg_network:
```

Below the code, the terminal interface includes tabs for Problems, Output, Debug Console, Terminal (which is selected), and Ports.

```
● → APP_DEV touch init.sql
\% 
● → APP_DEV docker volume ls
DRIVER      VOLUME NAME
local      82a06c4b0a00e81a4a6824322cff6610781aa5d974fb3010c81d1c006a6d7d4
local      app_dev_db_data
local      b19ad5fb0f7955d0998da3e99c663acc457a37e3be618ccc0b6fe002e4045d6c
● → APP_DEV docker compose stop
WARN[0000] /Users/2madeira/DEV/first_sem/APP_DEV/docker-compose.yml: the attribute 'volumes' is deprecated. Please use 'volumes_from' instead. See https://docs.docker.com/compose/compose-file/#volumes for more information.
WARN[0000] /Users/2madeira/DEV/first_sem/APP_DEV/docker-compose.yml: the attribute 'networks' is deprecated. Please use 'networks' instead. See https://docs.docker.com/compose/compose-file/#networks for more information.
[+] Stopping 2/2
  ✓ Container db_postgres  Stopped
  ✓ Container pgadmin4    Stopped
● → APP_DEV docker compose up -d
WARN[0000] /Users/2madeira/DEV/first_sem/APP_DEV/docker-compose.yml: the attribute 'volumes' is deprecated. Please use 'volumes_from' instead. See https://docs.docker.com/compose/compose-file/#volumes for more information.
WARN[0000] /Users/2madeira/DEV/first_sem/APP_DEV/docker-compose.yml: the attribute 'networks' is deprecated. Please use 'networks' instead. See https://docs.docker.com/compose/compose-file/#networks for more information.
[+] Running 2/2
  ✓ Container db_postgres  Started
  ✓ Container pgadmin4    Started
○ → APP_DEV
```

И при остановке и запуске БД данные сохраняются.

The screenshot shows the PgAdmin 4 interface. In the Object Explorer, under the 'Tables (1)' section, the 'dishes' table is selected. The 'Query' tab in the main window contains the SQL command:

```
1 SELECT *
2 FROM lab1.dishes
```

The 'Data Output' tab displays the results of the query:

	id	dishname	cook_complexity
1	1	солянка	normal
2	2	кочечные ребра	hard
3	3	пельмени	normal
4	4	чизбургер	easy
5	5	лазанья	hard

Total rows: 5 Query complete 00:00:00.079

Также, данные сохраняются и при удалении через docker-compose down. Однако, можно полностью удалить и тома тоже, если добавить флаг -v.

Итоговый вариант работы представляет собой следующее:

Я добавил запуск скрипта с созданием рабочей схемы и таблицы в ней в момент создания контейнера, для удобства.

```
version: '3.7'
services:
  db:
    image: postgres:17.0-bookworm
    container_name: db_postgres
    environment:
      - POSTGRES_USER=${POSTGRES_USER}
      - POSTGRES_PASSWORD=${POSTGRES_PASSWORD}
      - POSTGRES_DB=${POSTGRES_DB}
    ports:
      - "5432:5432"
    volumes:
      - db_data:/var/lib/postgresql/data
      - ./db_init:/docker-entrypoint-initdb.d
    networks:
      - pg_network
  pgadmin:
    image: dpage/pgadmin4:9.8.0
    container_name: pgadmin4
    environment:
      - PGADMIN_DEFAULT_EMAIL=${PGADMIN_DEFAULT_EMAIL}
      - PGADMIN_DEFAULT_PASSWORD=${PGADMIN_DEFAULT_PASSWORD}
    ports:
      - "8080:80"
    networks:
      - pg_network
  volumes:
    db_data:
  networks:
    pg_network
```

Problems Output Debug Console Terminal Ports

```
→ APP_DEV docker compose down -v
WARN[0000] /Users/2madeira/DEV/first_sem/APP_DEV/docker-compose.yml: the attribute `version` is obsolete, it will be ignored, please remove it to avoid potential confusion
[+] Running 4/4
  ✓ Container pgadmin4     Removed
  ✓ Container db_postgres   Removed
  ✓ Volume app_dev_db_data Removed
  ✓ Network app_dev_pg_network Removed
○ → APP_DEV
```

В терминале можно видеть, что я удалил и образы и тома для того, чтобы правильно протестировать итоговый вариант. Еще добавил поля с датой создания и обновления записи, чтобы таблица была более полезной.

```
CREATE SCHEMA IF NOT EXISTS lab1;
CREATE TYPE complexity AS ENUM ('easy', 'normal', 'hard');
CREATE TABLE lab1.dishes (
    id SERIAL PRIMARY KEY,
    dishname VARCHAR(64) NOT NULL,
    cook_complexity complexity NOT NULL,
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    updated_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);
INSERT INTO lab1.dishes(dishname, cook_complexity) VALUES
('солянка', 'normal'),
('копченые ребра', 'hard'),
('пельмени', 'normal'),
('чизбургер', 'easy'),
('лазанья', 'hard');
```

Problems Output Debug Console Terminal Ports

```
→ APP_DEV docker compose up -d
WARN[0000] /Users/2madeira/DEV/first_sem/APP_DEV/docker-compose.yml: the attribute `version` is obsolete, it will be ignored, please remove it to avoid potential confusion
[+] Running 4/4
  ✓ Network app_dev_pg_network Created
  ✓ Volume app_dev_db_data Created
  ✓ Container pgadmin4 Started
  ✓ Container db_postgres Started
○ → APP_DEV
```

Как видно, все работает, таблица и схема создаются.

The screenshot shows the pgAdmin 4 interface running in a Safari browser window. The left sidebar displays the Object Explorer with a tree structure of servers, databases, and schema. In the main pane, a query editor window is open with the following SQL code:

```
1 SELECT *
2 FROM lab1.dishes
```

The results of the query are displayed in a Data Output grid:

	id [PK] integer	dishname character varying (64)	cook_complexity complexity	created_at timestamp without time zone	updated_at timestamp without time zone
1	1	солянка	normal	2025-09-27 14:10:25.716002	2025-09-27 14:10:25.716002
2	2	котченые ребра	hard	2025-09-27 14:10:25.716002	2025-09-27 14:10:25.716002
3	3	пельмени	normal	2025-09-27 14:10:25.716002	2025-09-27 14:10:25.716002
4	4	чизбургер	easy	2025-09-27 14:10:25.716002	2025-09-27 14:10:25.716002
5	5	лазанья	hard	2025-09-27 14:10:25.716002	2025-09-27 14:10:25.716002

At the bottom of the pgAdmin window, status information indicates "Total rows: 5" and "Query complete 00:00:00.081".