Relatório docker 6

Criando, configurando o docker-compose:

docker-compose.yml

```yaml
version: '3'

services:
  db:
    image: postgres
    container_name: postgres_ex6
    volumes:
      - eduMountPostgres:/var/lib/postgres
    environment:
      - POSTGRES_DB=postgres_ex6
      - POSTGRES_USER=postgres_user
      - POSTGRES_PASSWORD=811944
    ports:
      - 5432:5432
volumes:
  eduMountPostgres:

```

Rodando o comando **docker compose up -d** para criar o container:

```
PS C:\Users\Eduardo\Desktop\Docker\ex6> docker compose up -d
[+] Running 3/3
 ✓ Network ex6_default           Created                                    0.2s
 ✓ Volume "ex6_eduMountPostgres" Created                                    0.0s
 ✓ Container postgres_ex6        Started                                    6.6s
PS C:\Users\Eduardo\Desktop\Docker\ex6>
```

Verificando se o container está rodando:

```
PS C:\Users\Eduardo\Desktop\Docker\ex6> docker ps
CONTAINER ID   IMAGE      COMMAND                  CREATED         STATUS         PORTS                    NAMES
9a821066985e   postgres   "docker-entrypoint.s…"   2 minutes ago   Up 2 minutes   0.0.0.0:5432->5432/tcp   postgres_ex6
```

conectando ao postgres que se encontra no container que acabei de criar:

```
PS C:\Users\Eduardo\Desktop\Docker\ex6> docker compose exec db psql -U postgres_user -d postgres_ex6
psql (15.3 (Debian 15.3-1.pgdg110+1))
Type "help" for help.

postgres_ex6=#
```

criando a tabela necessária para a API:

```
postgres_ex6=# create table accounts(id serial primary key not null, email varchar(255), name varchar(255), passsword varchar(60));
CREATE TABLE
postgres_ex6=#
```

derrubando o container com o comando **docker compose down:**

```
PS C:\Users\Eduardo\Desktop\Docker\ex6> docker compose down
[+] Running 2/2
 ✓ Container postgres_ex6  Removed                                          0.6s
 ✓ Network ex6_default     Removed                                          0.3s
PS C:\Users\Eduardo\Desktop\Docker\ex6> docker ps
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES
PS C:\Users\Eduardo\Desktop\Docker\ex6>
```

Acrescentando o container **backend-web:**

```yaml
  backend-web:
    image: ex5
    container_name: backend_edu
    environment:
      - PORT=8080
      - HOST=localhost
      - PG_HOST=db
      - PG_USER=postgres_user
      - PG_DATABASE=postgres_ex6
      - PG_PASSWORD=811944
      - PG_PORT=5432
      - JWTSECRET=TypescriptComCafe
    ports:
    - 8080:8080

volumes:
  eduMountPostgres:
```

Rodando o compose:

```
PS C:\Users\Eduardo\Desktop\Docker\ex6> docker compose up -d
validating C:\Users\Eduardo\Desktop\Docker\ex6\docker-compose.yml: networks must be a mapping
[+] Running 3/3
 ✓ Network ex6_default      Created
 ✓ Container postgres_ex6   Started
 ✓ Container backend_edu    Started
PS C:\Users\Eduardo\Desktop\Docker\ex6>
```

Verificando se os 2 containers estão rodando:

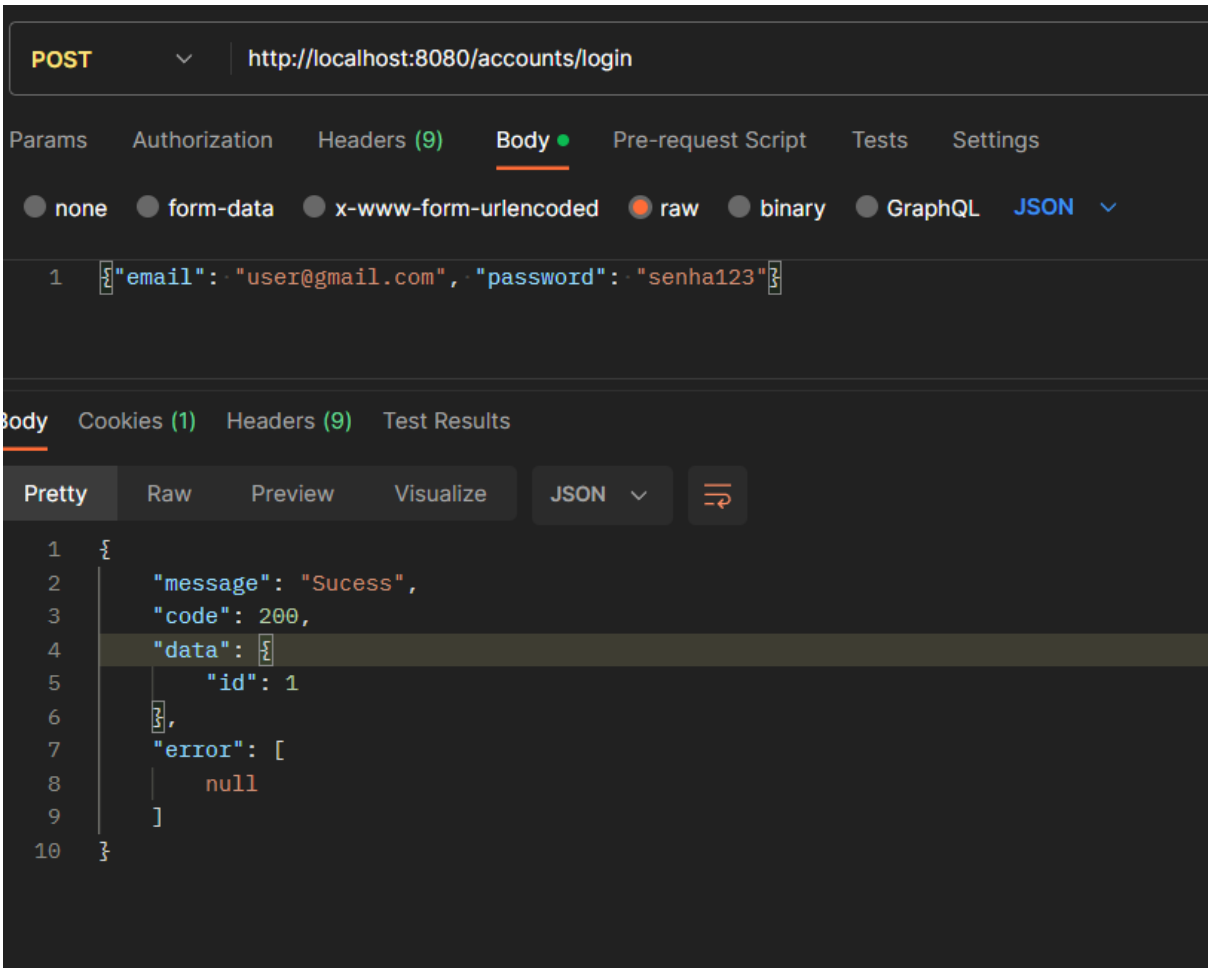
```
PS C:\Users\Eduardo\Desktop\Docker\ex6> docker ps
CONTAINER ID   IMAGE      COMMAND                  CREATED         STATUS         PORTS                              NAMES
e5b85a875bd7   ex5        "docker-entrypoint.s…"   9 minutes ago   Up 9 minutes   3000/tcp, 0.0.0.0:8080->8080/tcp   backend_edu
8c8731ffe2af   postgres   "docker-entrypoint.s…"   9 minutes ago   Up 9 minutes   0.0.0.0:5432->5432/tcp             postgres_ex6
PS C:\Users\Eduardo\Desktop\Docker\ex6>
```

Depois de recriar a tabela **accounts** inseri um usuário chamado ‘user’:

```
postgres_ex6=# INSERT INTO accounts (email, name, password) VALUES ('user@gmail.com', 'user', 'senha123');
INSERT 0 1
postgres_ex6=# SELECT * FROM accounts;
 id |     email      | name | password
----+----------------+------+----------
  1 | user@gmail.com | user | senha123
(1 row)

postgres_ex6=#
```

Acessando ao container que está rodando a API pelo postman:



VII Reposta:

Sem a definição de uma rede no arquivo docker-compose.yml, o Docker Compose cria automaticamente uma rede padrão para os serviços especificados, essa rede permite que os containers se comuniquem entre si usando os nomes dos serviços como endereços de host.

Por isso que o serviço **‘backen-web’** consegue se comunicar com o serviço **‘db’**, pois os dois serviços estão em uma rede padrão que o docker compose cria.

A falta da declaração da porta no **docker-compose.yml** não impede a comunicação entre os dois containers, mas impede o acesso ao container que está rodando o serviço ‘db’ por qualquer outra ferramenta de gerenciamento de banco, como o pgAdmin, que está fora do ambiente Docker.