Task instructions

Docker:

1) Ensure Docker and Docker Compose are installed

First of all, make sure you have Docker installed on your machine. If not, you may download it from official Docker's official website.

2) Build and Run Containers

Open folder **"Project"**, then open a **"docker-compose.yml"**.

After that you need to open terminal and write a command **"docker-compose build"** to build the Docker images. Then write **"docker-compose up"** to start the containers.

This will build your Docker images and start your services defined in **"docker-compose.yml"**. You can access Jupyter Notebook (password token: at http://localhost:8081 and Adminer with PostgreSQL at http://localhost:8080.

3) Stopping containers

When you will finish your work you need to stop containers. To do this use command **"docker-compose down"**.

SQL Queries:

1) Create Table Query:

Query | Query History

```
CREATE TABLE customer(
    user_id SERIAL PRIMARY KEY,
    name VARCHAR(150) NOT NULL,
    email VARCHAR(250) UNIQUE NOT NULL,
    signup_date DATE NOT NULL,
    domain VARCHAR(50) NOT NULL
)
```

2) Insert data query:

Copy data from my new CSV file: "New_Customer.csv":

Open query tool and write this command:

```
COPY Public."customer" FROM 'C:\Inforce Task\Project\New_Customer.csv' DELIMITER ',' CSV HEADER
```

After that your data will be completely insert into your database.

3) SQL Queries

Open folder **"Project\SQL queries"**. After that you can paste each of the, into Query tool.

1.

```
SELECT signup_date, COUNT(*) FROM customer
GROUP BY signup_date
```

	signup_date date	count bigint
1	2022-12-25	3
2	2023-02-15	2
3	2022-11-14	1
4	2023-06-08	2
5	2024-02-16	2
6	2024-05-03	2
7	2024-06-19	1
8	2024-07-04	1
9	2023-06-20	2

2.

```
SELECT DISTINCT(domain) FROM customer
```

	domain character varying (100)
1	example.com
2	yahoo.com
3	gmail.com

3.

Query Query History

```
1 SELECT * FROM customer
2 WHERE signup_date >= CURRENT_DATE - INTERVAL '7 DAYS'
```

Data Output Messages Notifications

	user_id [PK] integer	name character varying (150)	email character varying (250)	signup_date date	domain character varying (100)
1	135	Karen Hernandez	thompsondavid@gmail.com	2024-08-19	gmail.com
2	169	Shirley Scott	robertjohnson@example.com	2024-08-18	example.com
3	221	John Shepherd	warrenjared@gmail.com	2024-08-19	gmail.com
4	382	Omar Scott	jeremyhill@yahoo.com	2024-08-19	yahoo.com
5	590	Sara Brown	brandirubio@yahoo.com	2024-08-17	yahoo.com
6	728	Douglas Mathews	smithvictoria@yahoo.com	2024-08-15	yahoo.com
7	734	Kaitlyn Cole	ortiztasha@yahoo.com	2024-08-20	yahoo.com
8	744	Duane James	emily63@gmail.com	2024-08-16	gmail.com
9	866	Jonathan Martinez	brittney20@example.com	2024-08-15	example.com

4.

Query Query History

```
1 WITH most_common_domain AS(
2     SELECT domain, COUNT(domain) FROM customer
3     GROUP BY domain
4     ORDER BY COUNT(domain) DESC
5     LIMIT 1
6 )
7
8 SELECT * FROM customer
9 WHERE domain = (SELECT domain FROM most_common_domain)
```

Data Output Messages Notifications

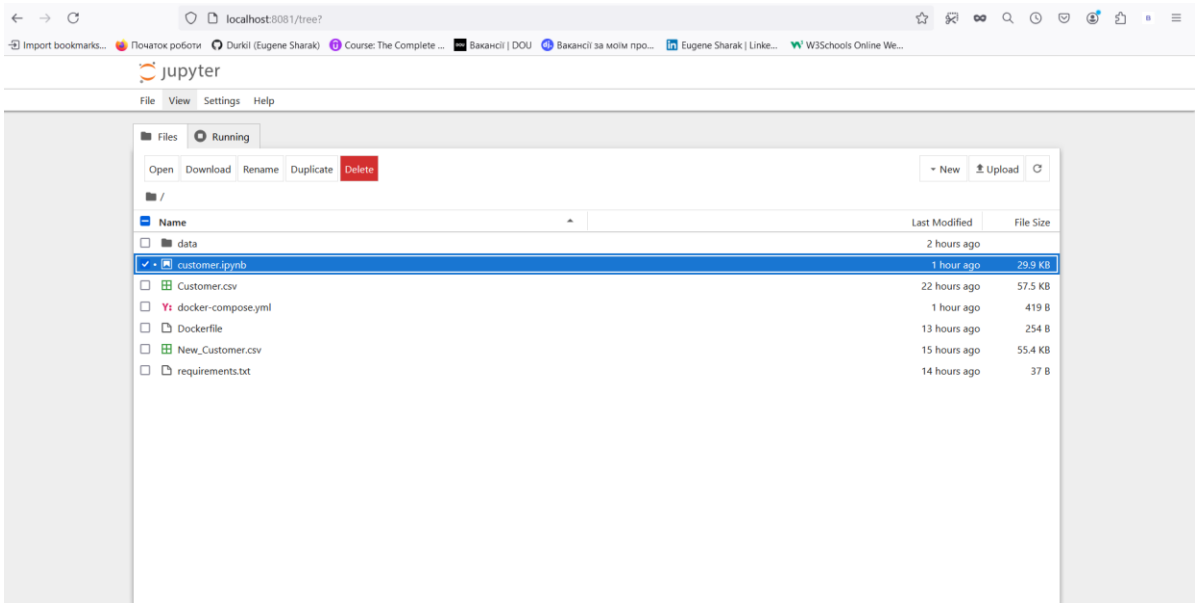
	user_id [PK] integer	name character varying (150)	email character varying (250)	signup_date date	domain character varying (100)
1	1	David Oneill	jenniferedwards@gmail.com	2024-04-25	gmail.com
2	2	Shelly Martinez	suzanne38@gmail.com	2023-10-26	gmail.com
3	4	Megan Romero	tinakrause@gmail.com	2023-09-01	gmail.com
4	6	William Miller	brian13@gmail.com	2024-03-11	gmail.com
5	24	Linda Romero	gstanley@gmail.com	2024-04-12	gmail.com
6	26	Courtney Green	imarquez@gmail.com	2024-08-01	gmail.com
7	27	Nicole Rodriguez	uyork@gmail.com	2023-08-23	gmail.com
8	34	Andrew Thomas	debra96@gmail.com	2024-06-16	gmail.com

5.

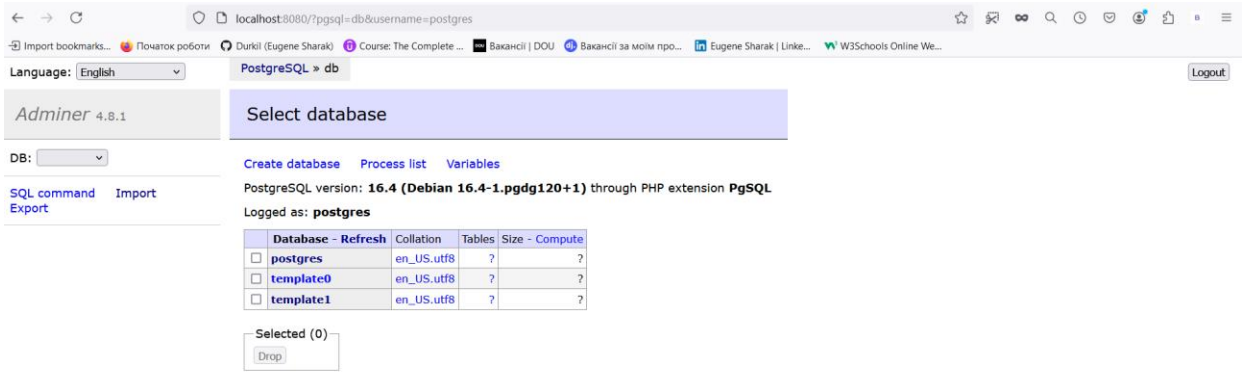
```
Query  Query History
1 DELETE FROM customer
2 WHERE domain NOT IN('gmail.com', 'example.com', 'yahoo.com')
```

Results:

http://localhost:8081



http://localhost:8080



P.S. Unfortunately, I did not have time to fully implement PostgreSQL in Docker Compose (localhost is working, but it does not have my database). It was my first experience with Docker and while I have not yet fully integrated database into the Docker Compose, I am confident in my ability to quickly learn and resolve this issue.