

МІНІСТЕРСТВО ОСВІТИ ТА НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ «КПІ»



Кафедра інформаційних систем та технологій

Курсова робота

з дисципліни

«Бази даних»

Варіант № 20

Перевірив:

Попенко В. Д.

Виконала: Павлова Софія

Студентка гр. ІС-12 , ФІОТ

2 курс,

залікова книжка № ІС-1224

Київ 2023

ЗМІСТ

ВСТУП.....	3
1 ПОБУДОВА ER-МОДЕЛІ ПРЕДМЕТНОЇ ОБЛАСТІ	5
2 СТВОРЕННЯ БАЗИ ДАНИХ У POSTGRESQL	7
2.1 Установка Postgre	7
2.2 Створення серверу.....	13
2.3 Створення таблиць	14
2.4 Заповнення таблиць	17
2.5 Редагування таблиць	20
2.6 Створення зв'язків між таблицями та зміна структури	21
2.7 Створення ERD діаграми.....	21
3 СТВОРЕННЯ ЗАПИТІВ У POSTGRESQL.....	23
3.1 Запити від двох або більше таблиць.....	23
3.2 Групуючі запити	26
3.3 Аналітичні запити	30
3.4 Запити на зміну.....	32
4 СТВОРЕННЯ ЗАСТОСУВАННЯ НА ASP.NET.....	41
4.1 Створення додатку в робочому середовищі ASP.NET.....	41
4.2 Огляд застосунку	47
5 ЕСКІЗ ЗВІТУ.....	54
ВИСНОВКИ.....	55
ПЕРЕЛІК ВИКОРИСУНОКТАНИХ ДЖЕРЕЛ.....	56

ВСТУП

Метою виконання курсової роботи є поглиблення і закріплення студентами теоретичних знань з дисципліни «Організація баз даних та знань», набуття вмінь аналізувати опрацьований матеріал, робити відповідні узагальнення та висновки. Написання курсової роботи дає студентові можливість навчитися самостійно і творчо виконувати наукові джерела та узагальнювати теоретичні положення.

Основні етапи підготовки та написання курсової роботи:

- Вибір теми та осмислення завдання дослідження.
- Добір та вивчення літератури, збирання та обробка інформації.
- Складання плану роботи.
- Написання та оформлення курсової роботи.

На основі опису предметної області необхідно виконати наступне:

- Розробити ER-діаграму предметної області.
- Створити базу даних і наповнити її тестовими даними (схему бази даних відобразити в курсовій роботі).
- До цієї бази розробити по 2 запити наступних типів:
 - запити від двох або більше таблиць [1],
 - групуючі запити [10] та аналітичні запити,
 - запити на вставку, оновлення, видалення значень [1].
- Розробити необхідні форми введення інформації (1-3) у вигляді застосування на одному з технологічних стеків (.Net, NodeJs, Java або інші подібні).
- Розробити ескізи форм необхідних звітів (1-2). Ескіз форми звіту - текст Word або таблиця Excel, що містить структуру звіту: заголовок, капелюх, колонки, підзаголовки, структуру рядків, підсумки. Виконання кожного з перелічених кроків необхідно відобразити в основній частині тексту курсової роботи.

Під час роботи виконується повний цикл інфологічного та даталогічного проектування з урахуванням вимог до предметної області. На основі цього будується фізична реалізація бази даних за допомогою СУБД, що обирається спираючись на

можливості операційної системи та інші фактори, що залежать від обчислювальних можливостей. Результатом роботи є програмний продукт, що готовий до застосування.

У даній роботі розглянемо основні етапи проектування та створення бази даних для будь якої предметної області, такі як постановку задачі бази даних та вимог до неї, інфологічне та даталогічне проектування, а також фізичне проектування. Бази даних є вкрай необхідними для будь-якого проєкту, будь-якого програмного застосунку. Правильно спроектована база даних гарантує її безпомилкове, ефективне та зручне виконання. При цьому треба враховувати вимоги клієнта до середовища та операційної обстановки. Ціль роботи - побудувати зручну базу для підтримки приймальної народного депутата, що отримує та опрацьовує звернення адресатів.

1. ПОБУДОВА ER-МОДЕЛІ ПРЕДМЕТНОЇ ОБЛАСТІ

Варіант 20:

Відповідно до варіанту необхідно створити ER-модель згідно з предметною областю приймальні депутата.

До народного депутата від мажоритарного округу надходять звернення громадян. Як правило, вони містять скарги на роботу державних органів і потребують звернення депутата до цих або інших органів влади.

На практиці існують такі варіанти реагування на звернення громадян:

- на звернення громадянина надається відповідь депутата громадянину в формі листа;

- на звернення громадянина відправляються один або кілька листів депутата в державні органи. Ці органи мають надати відповіді у встановлений термін, після чого депутат має надіслати листа з відповіддю громадянину.

Помічники народного депутата постійно аналізують:

- по яких зверненнях і по яких листах органи влади порушили терміни відповіді;

- які звернення громадян наразі без відповіді і який стан справ по кожному зверненню, тобто які листи розіслані, чи є на них відповіді і т.п.

Відправником і отримувачем листів є депутат, а не помічник. Звернення і листи з одного приводу об'єднуються спільною темою, як-от: *Стосовно ремонту гуртожитку по адресі пер.Ковальський, 15.*

ER діаграму, наведено на Рисунок 1.1.

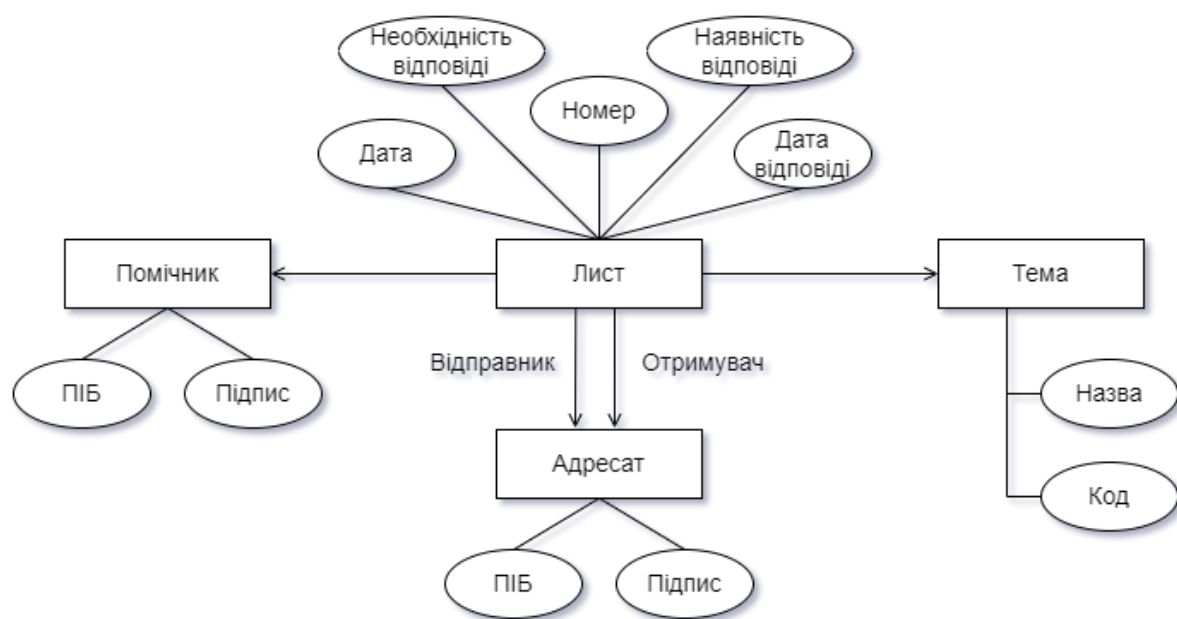


Рисунок 1.1 ER-діаграма предметної області Приймальні народного депутата

2. СТВОРЕННЯ БАЗИ ДАНИХ У POSTGRESQL

Відповідно до варіанту необхідно створити БД згідно з предметною областю приймальні депутата.

2.1. Установка Postgre

Installer було встановлено з сайту <https://www.enterprisedb.com/downloads/postgres-postgresql-downloads> . Ключові етапи установки наведені на скріншотах нижче.

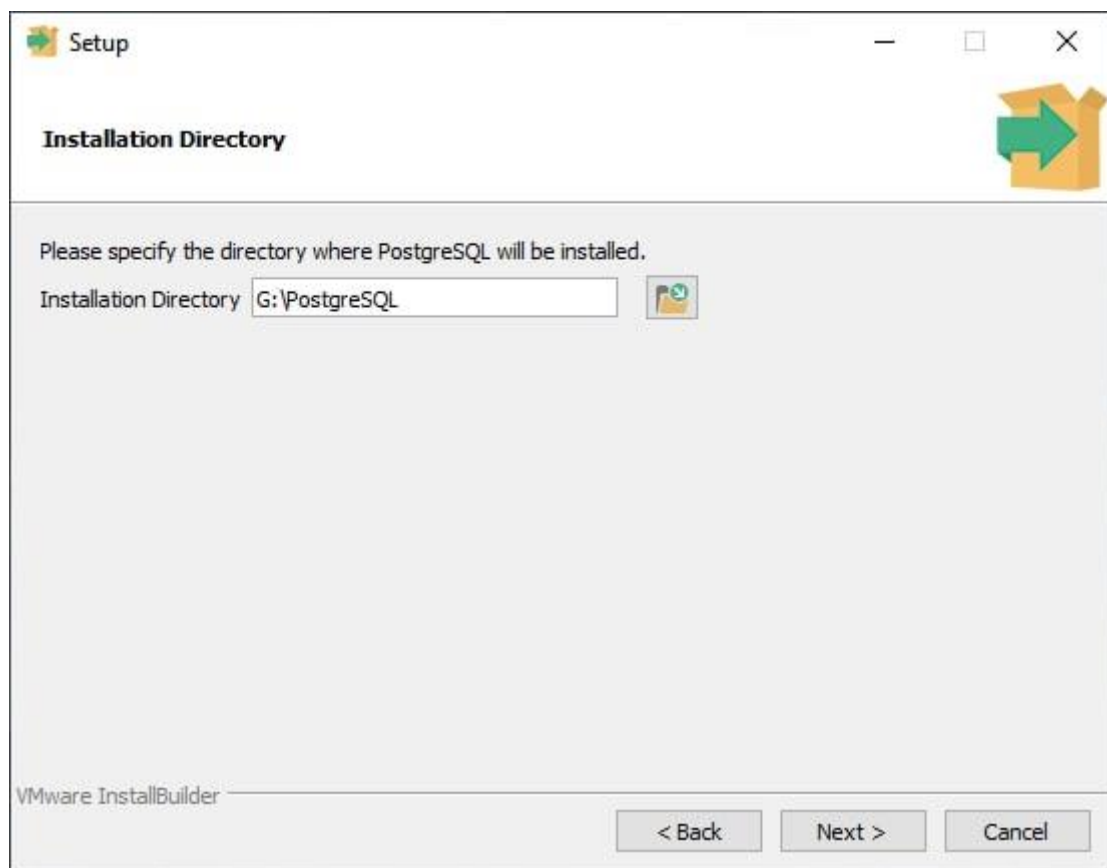


Рисунок 2.1 Вибір директорії для установки

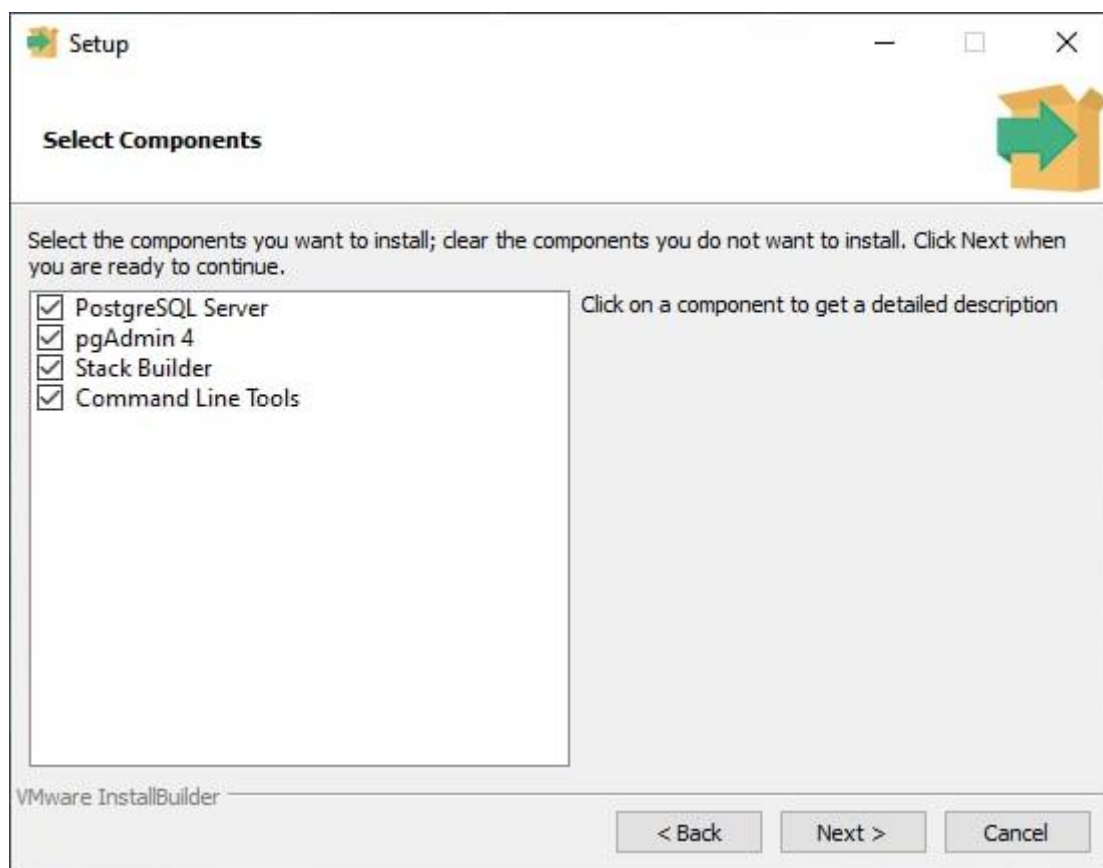


Рисунок 2.2 Вибір компонентів для установки

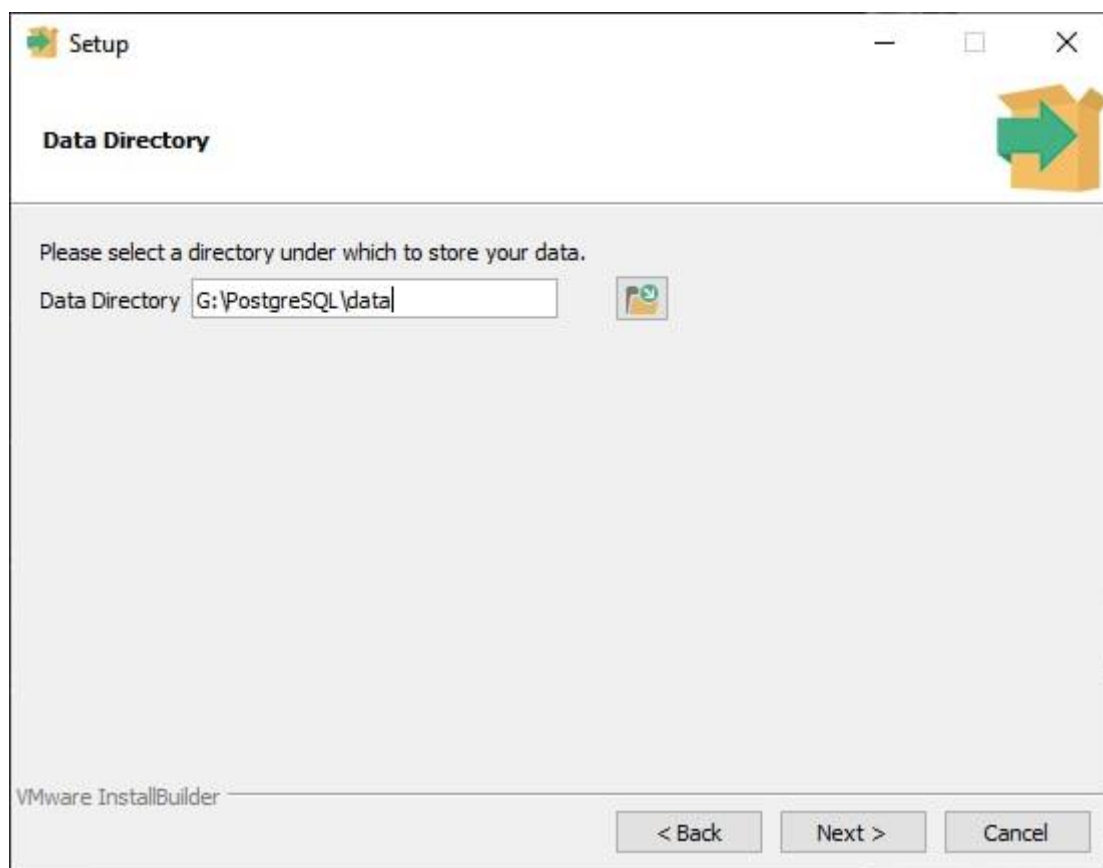


Рисунок 2.3 Вибір директорії для зберігання даних

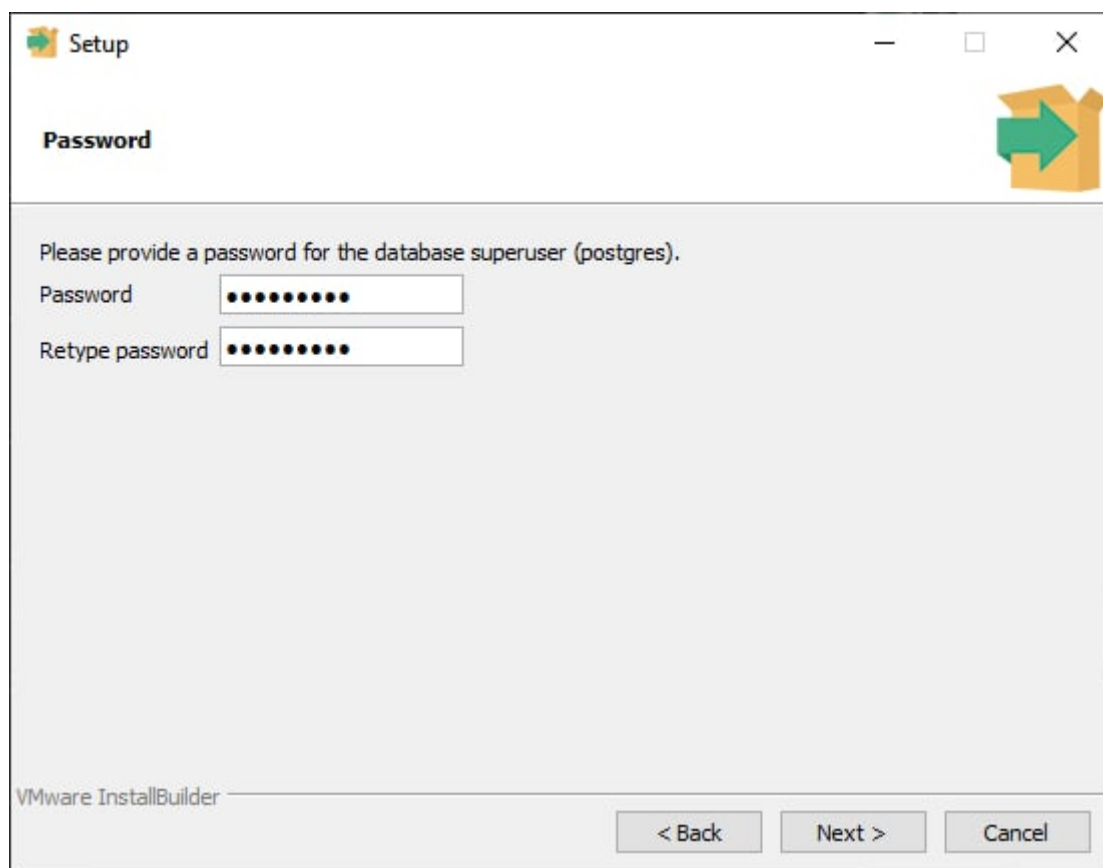


Рисунок 2.4 Установка пароля для администратора

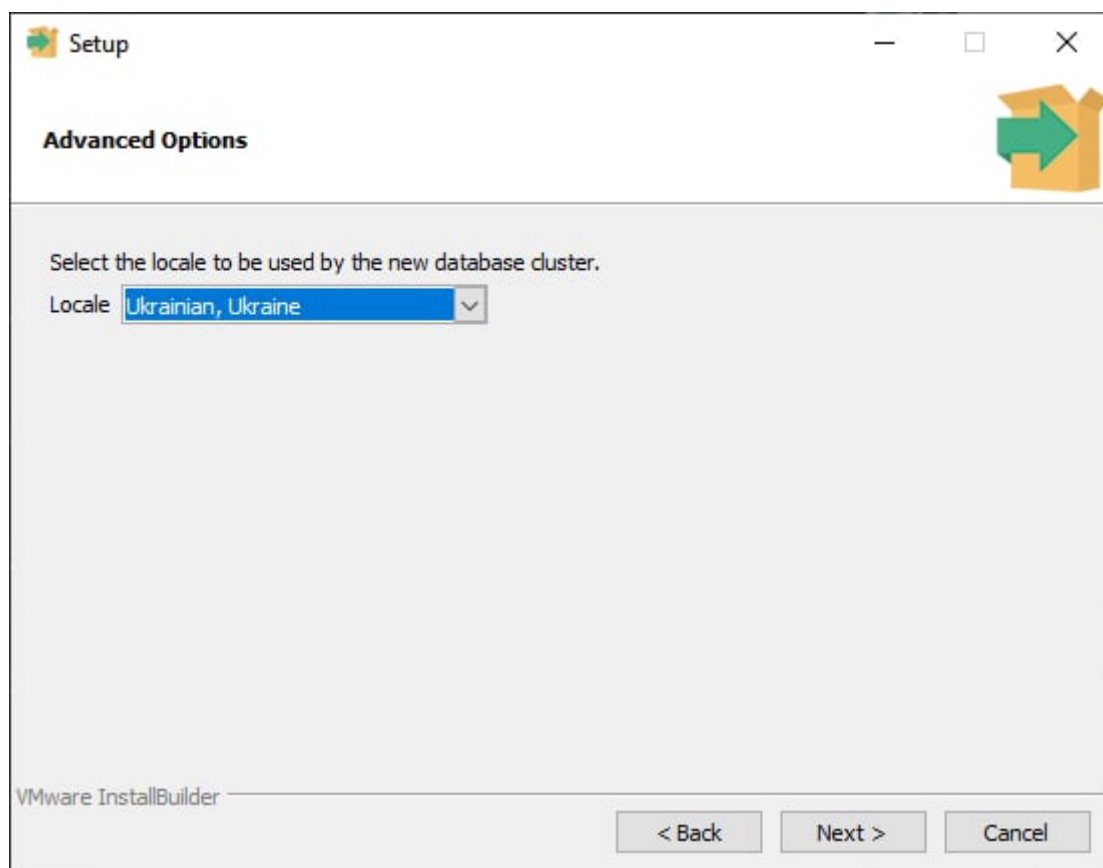


Рисунок 2.5 Вибір мови



Рисунок 2.6 Завершення установки

Після завершення установки, потрібно відкрити файл pgAdmin4.exe і можна починати роботу з БД (рисунок 2.7).

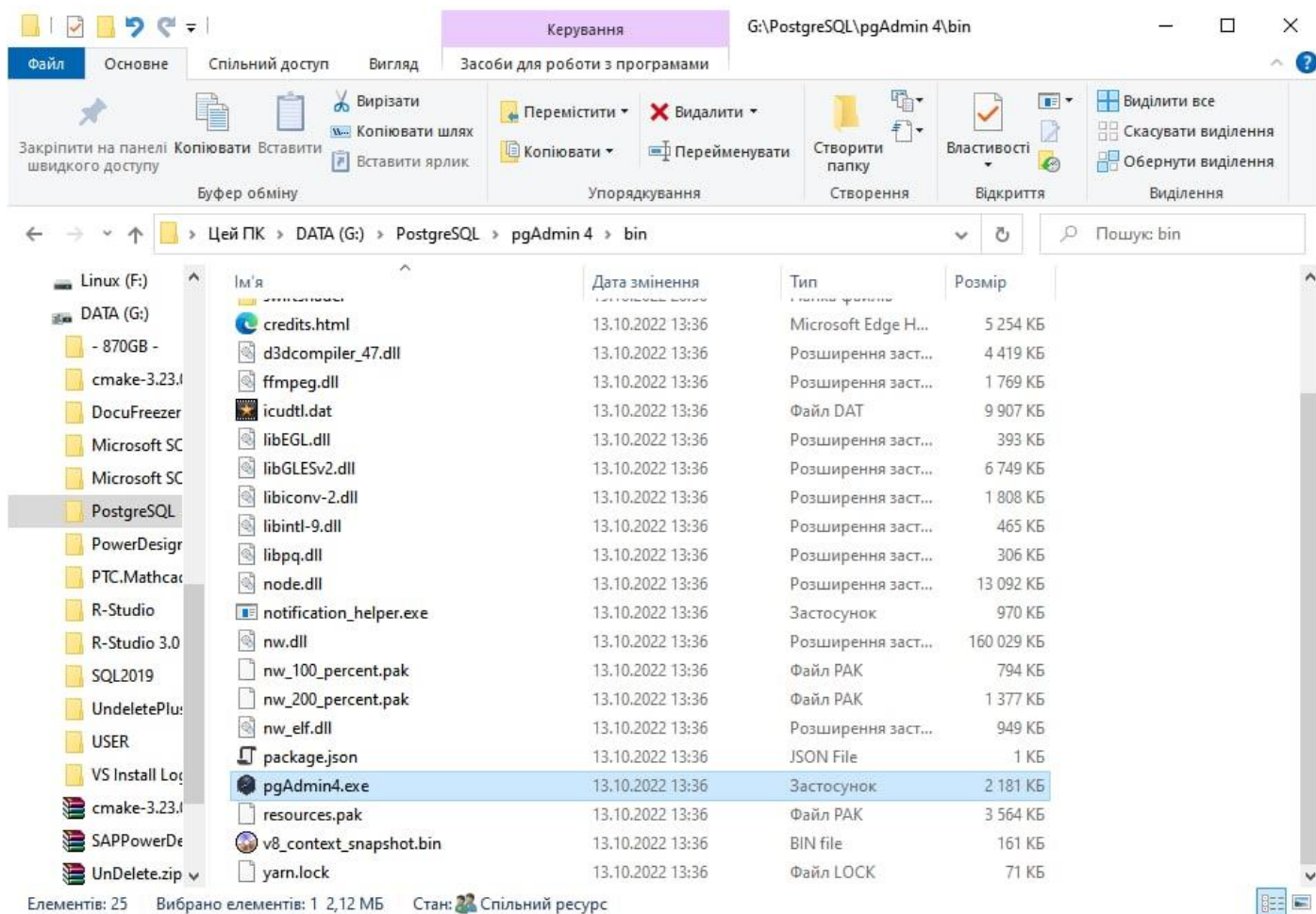


Рисунок 2.7 Установлений застосунок

По заходженню в програму, вона просить ввести пароль, що був вказаний при інсталяції (рисунок 2.8).

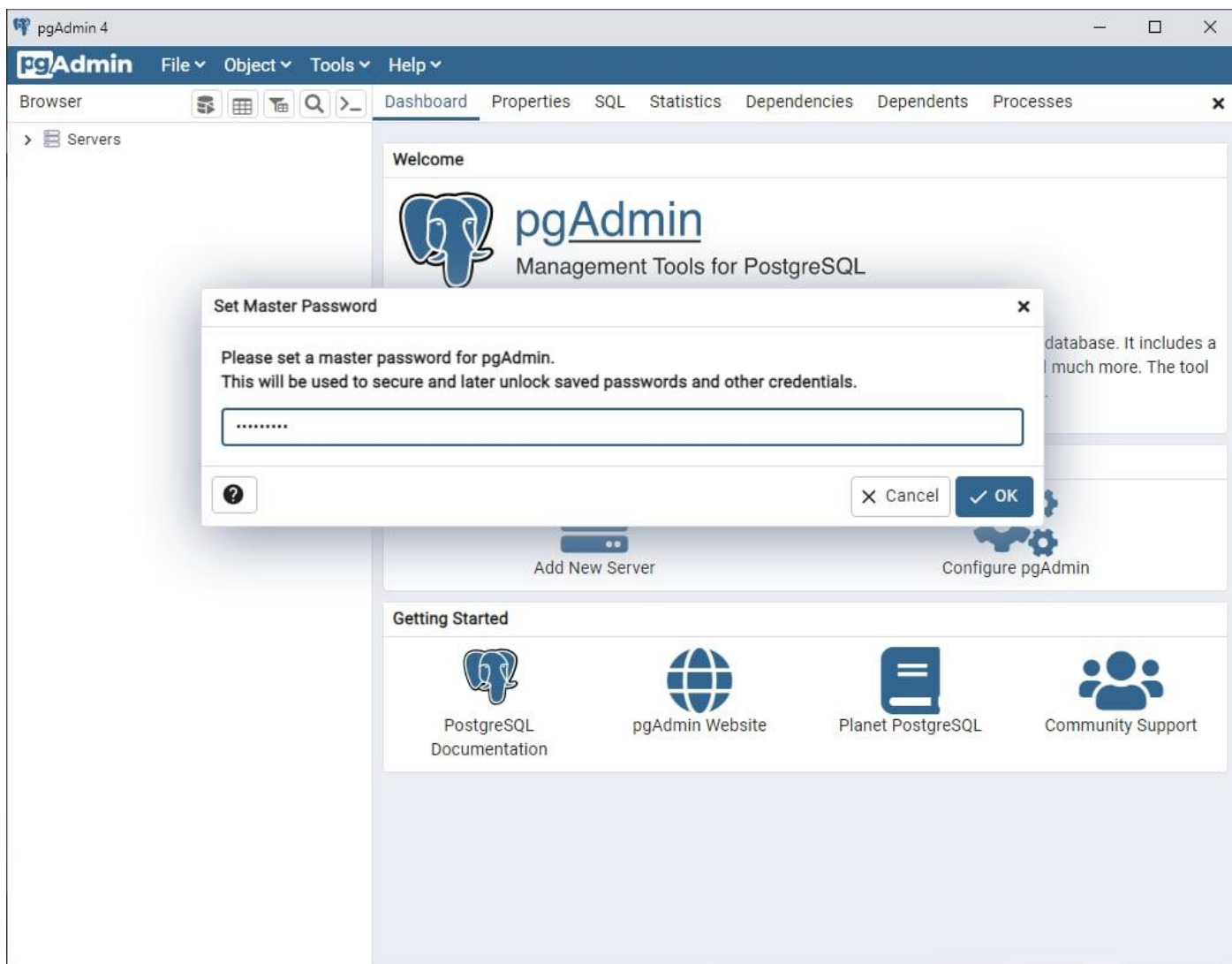


Рисунок 2.8 Установка пароля для пользователя

Опісля завершення приготувань, маємо наступний результат інсталяції (рисунок 2.9). Програма готова до роботи з БД.

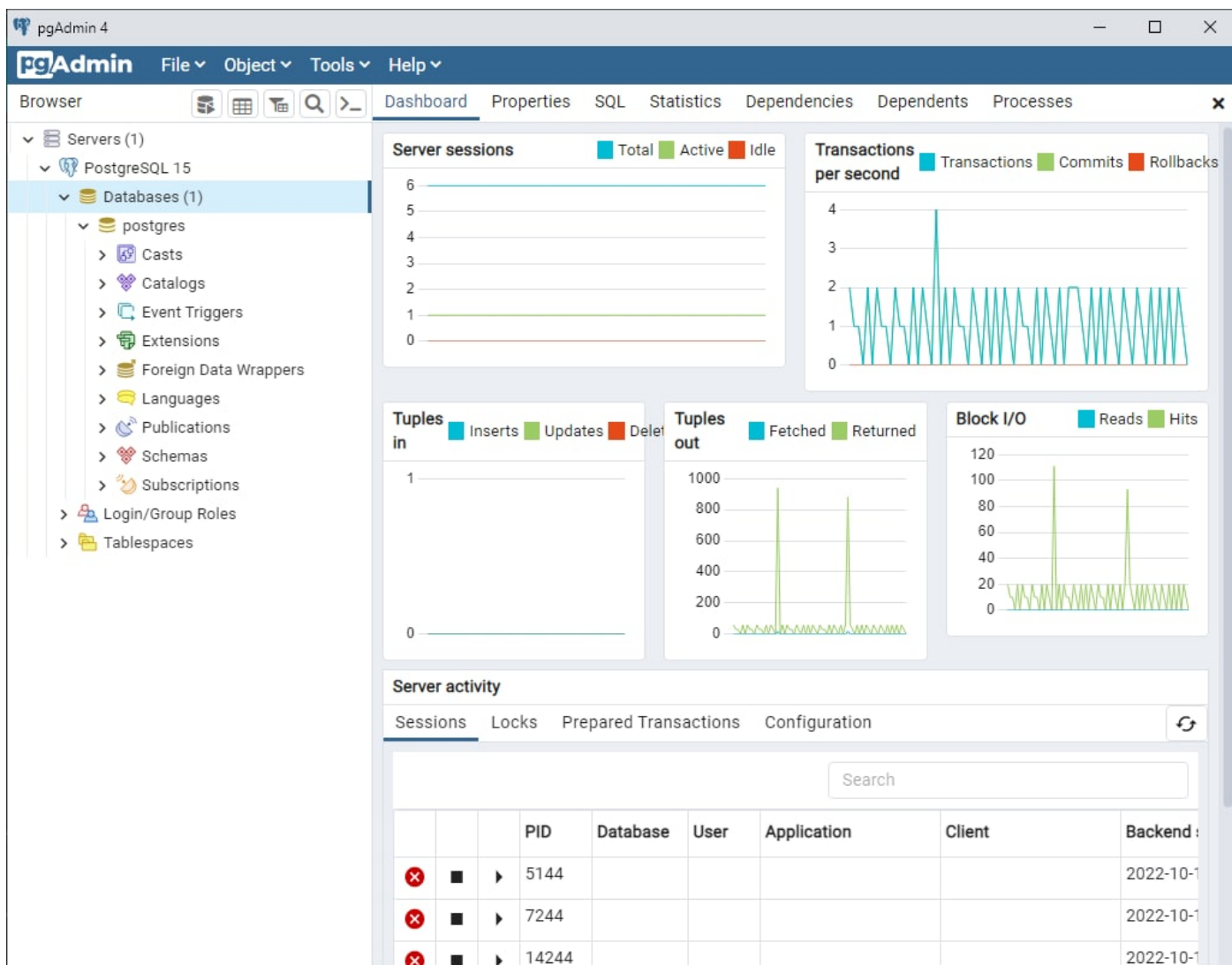


Рисунок 2.9 Результат інсталяції Postgre

2.2. Створення серверу

Для того, щоб почати працювати з БД, потрібно спочатку створити сервер.

Етапи створення серверу наведені на скріншотах нижче.

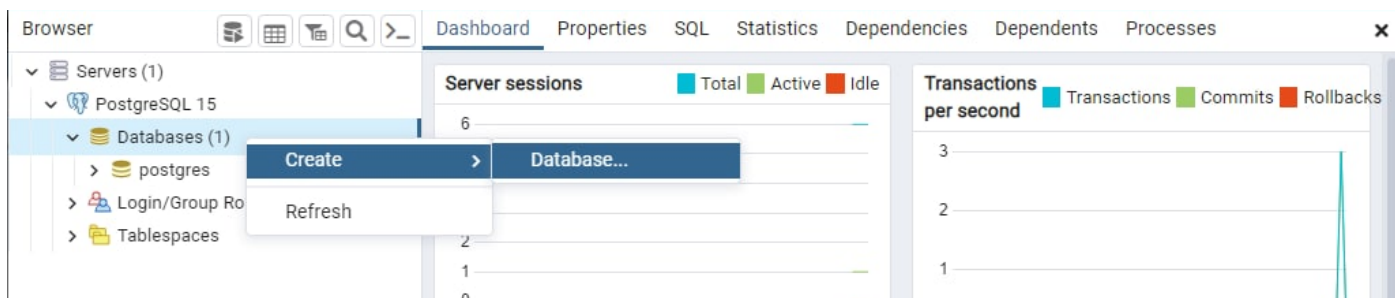


Рисунок 2.10 Створення БД

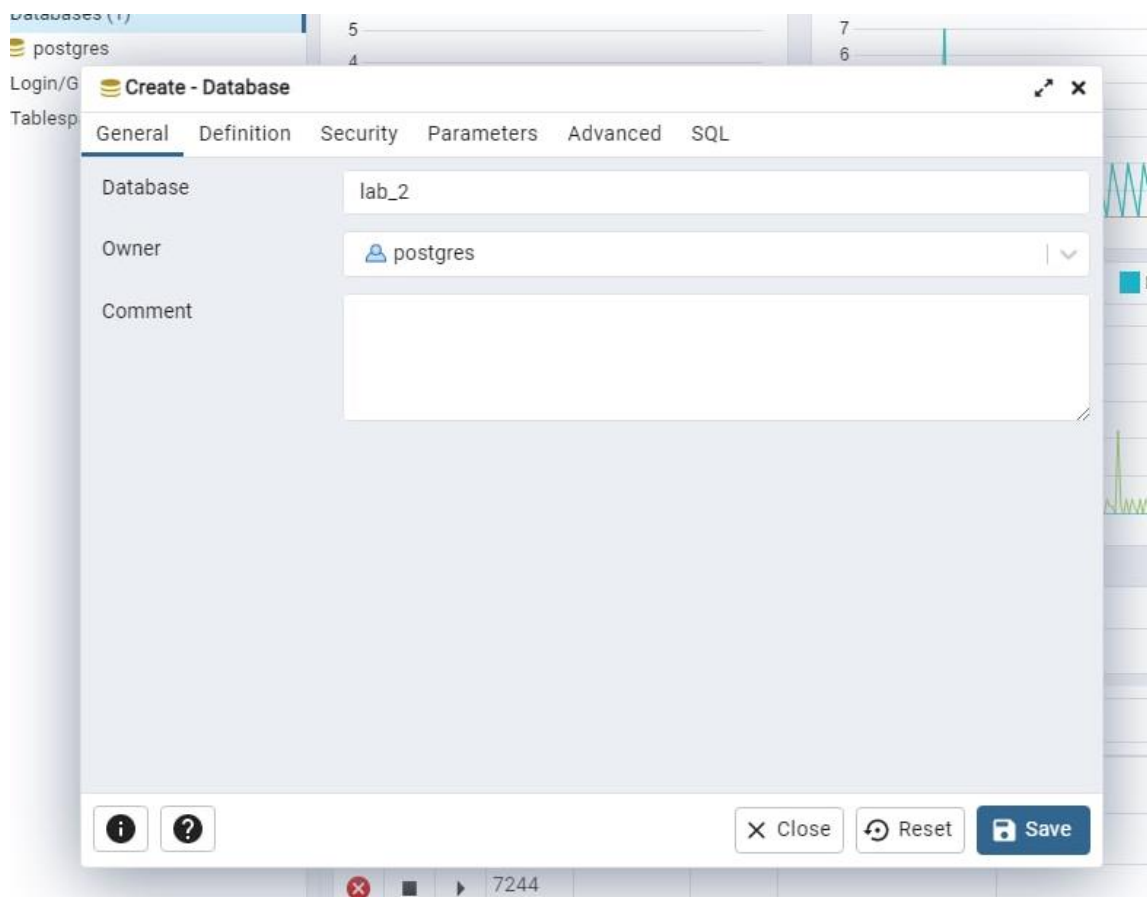


Рисунок 2.11 Назва БД

Опісля створення серверу, його можна буде побачити в дереві проекту у лівій частині екрану.

2.3. Створення таблиць

У Postgre існує 2 способи створення таблиць: мовою SQL та вручну.

Використовуємо другий спосіб, так, як він легший. Послідовні скріни створення таблиць наведено нижче.

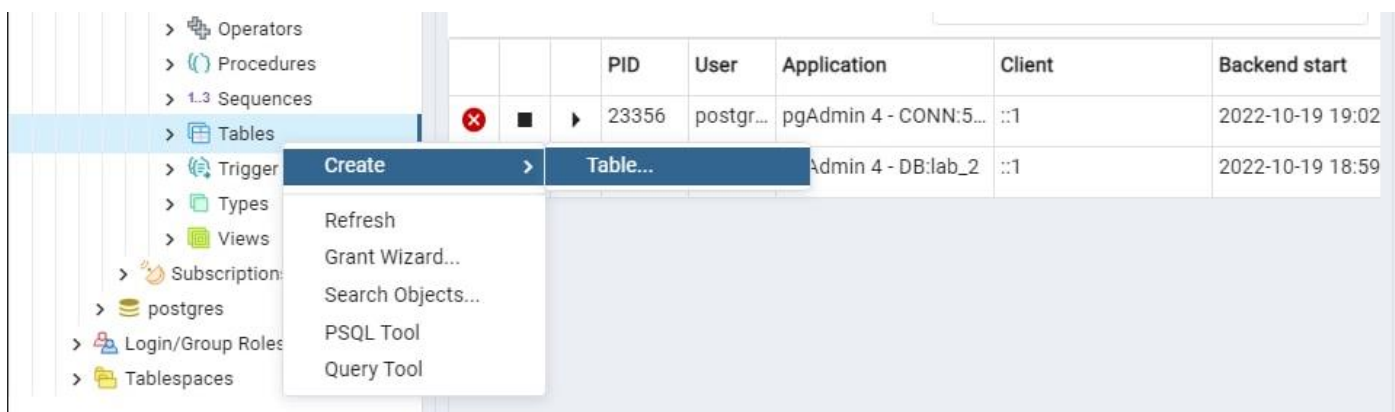


Рисунок 2.12 Створення таблиці

Створимо таблицю helper відповідно до умов варіанту. Решта таблиць створена аналогічно.

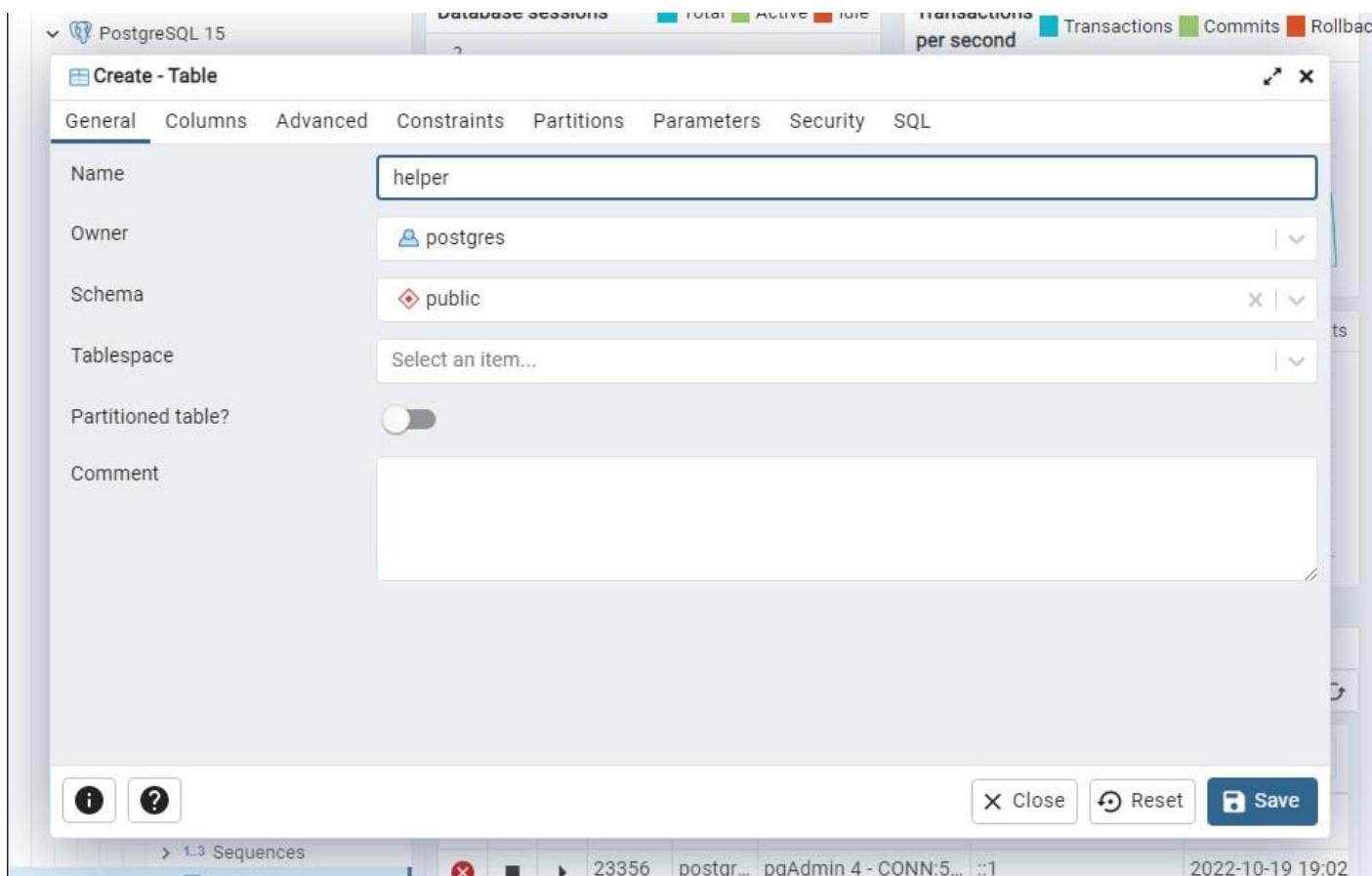


Рисунок 2.13 Назва таблиці

У секції Columns впишемо поля таблиці, які є атрибутами сутності Помічник і їх тип даних (рисунок 2.14).

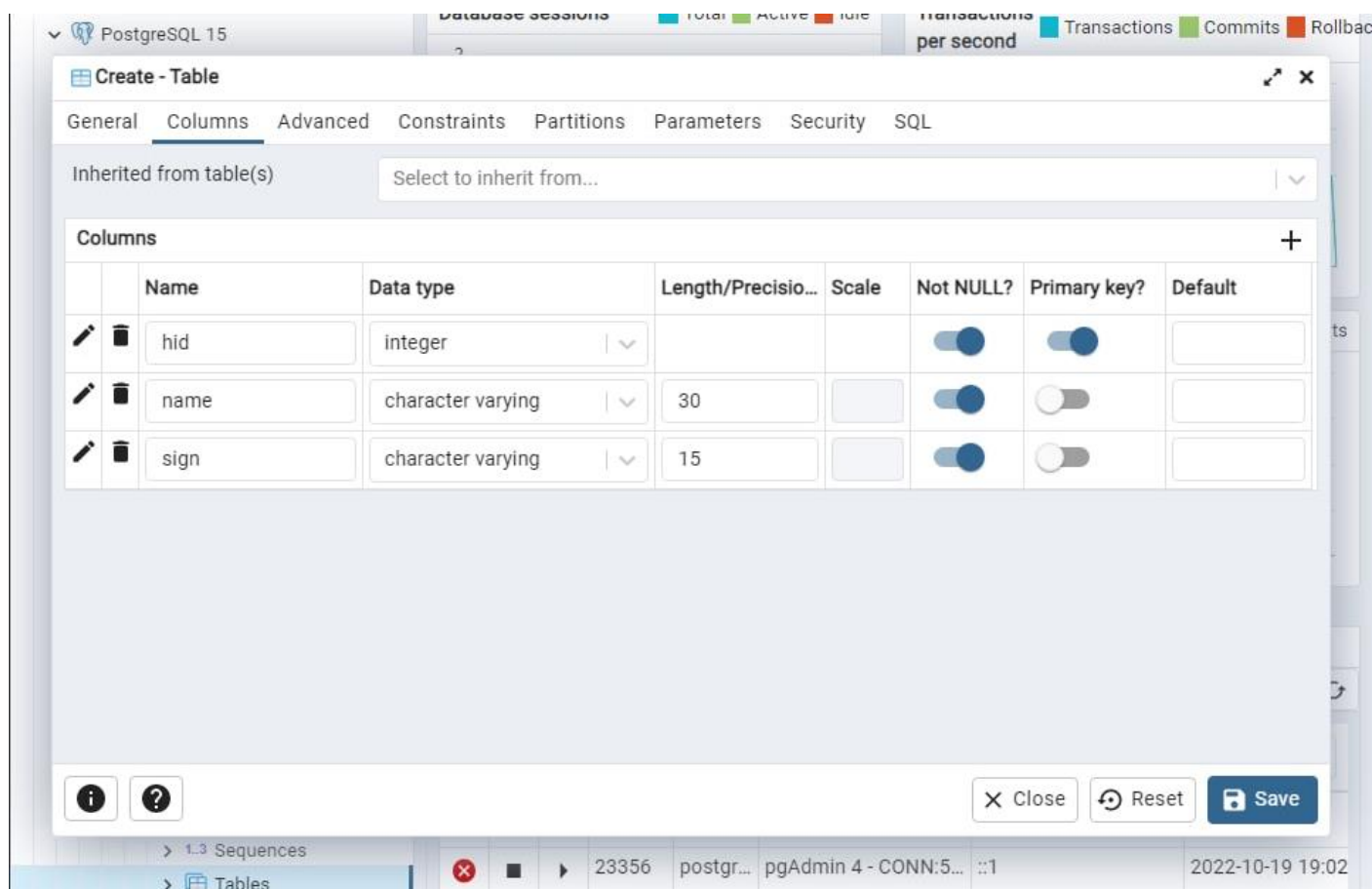


Рисунок 2.14 Налаштування стовпців таблиці

При використанні другого методу створення таблиць, скрипт SQL генерується автоматично. Якщо ж користуватися першим методом, для створення таблиці необхідно ввести наступний скрипт у Query.

Скрипт створення таблиці *helper*:

```
CREATE TABLE IF NOT EXISTS public.helper
(
    hid integer NOT NULL,
    name character varying(30) COLLATE pg_catalog."default" NOT NULL,
    sign character varying(15) COLLATE pg_catalog."default" NOT NULL,
);
```

Скрипт створення таблиці *adresser*:

```
CREATE TABLE IF NOT EXISTS public.adresser
(
    adid integer NOT NULL,
    name character varying(30) COLLATE pg_catalog."default" NOT NULL,
    sign character varying(15) COLLATE pg_catalog."default" NOT NULL,
    CONSTRAINT adresser_pkey PRIMARY KEY (adid)
);
```


Скрипт створення таблиці *letter*:

```
CREATE TABLE IF NOT EXISTS public.letter
(
    lid integer NOT NULL,
    date character(10) COLLATE pg_catalog."default" NOT NULL,
    is_ans boolean NOT NULL,
    date_ans character(10) COLLATE pg_catalog."default" NOT NULL,
    thema_id bigint,
    sender_id bigint,
    receiver_id bigint,
    helper_id bigint,
    CONSTRAINT letter_pkey PRIMARY KEY (lid),
    CONSTRAINT letter_thema_id_key UNIQUE (thema_id)
);
```

Скрипт створення таблиці *thema*:

```
CREATE TABLE IF NOT EXISTS public.thema
(
    tid integer NOT NULL,
    name character varying(50) COLLATE pg_catalog."default" NOT NULL,
    CONSTRAINT thema_pkey PRIMARY KEY (tid)
);
```

2.4. Заповнення таблиць

Для заповнення таблиць як і для їх створення існує два аналогічні методи, але використовуємо ми простіший з них. Для цього виконаємо наступні дії, наведені на скріншотах нижче.

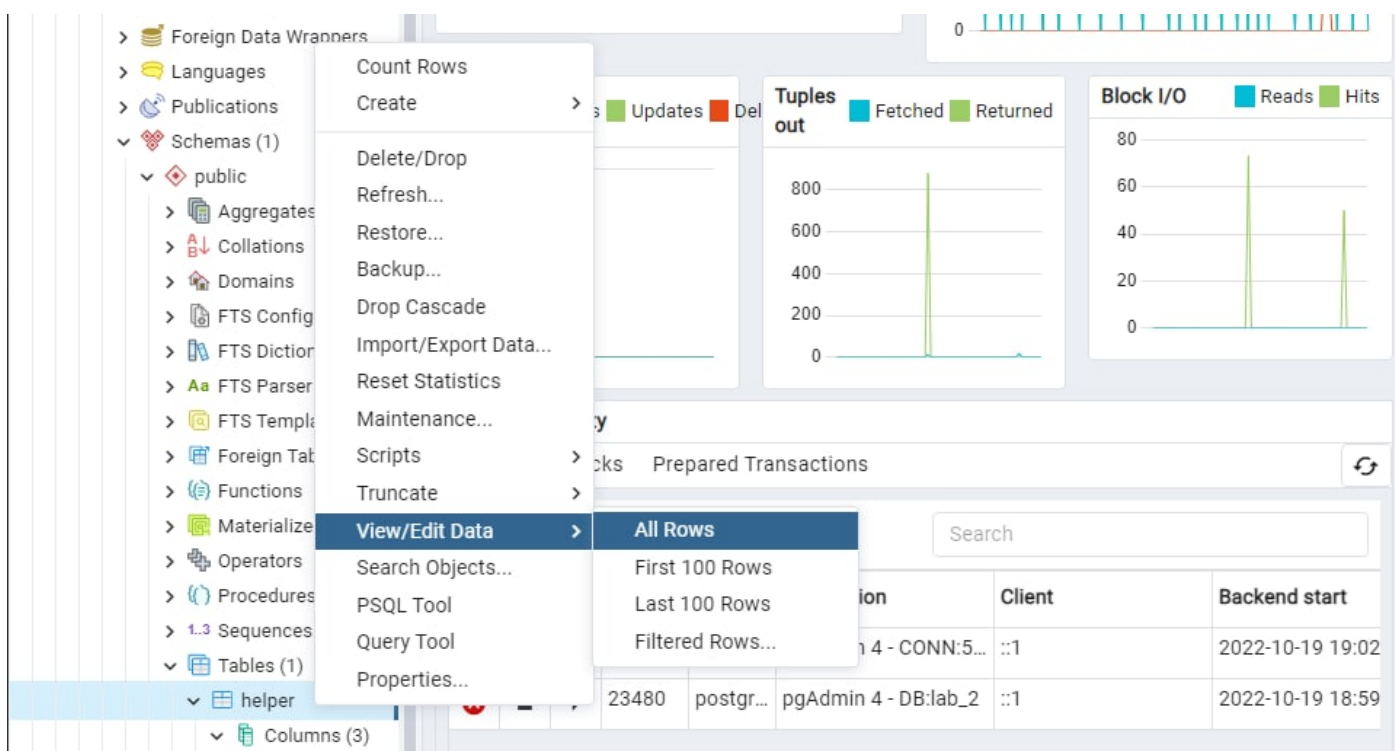


Рисунок 2.15 Перегляд таблиці

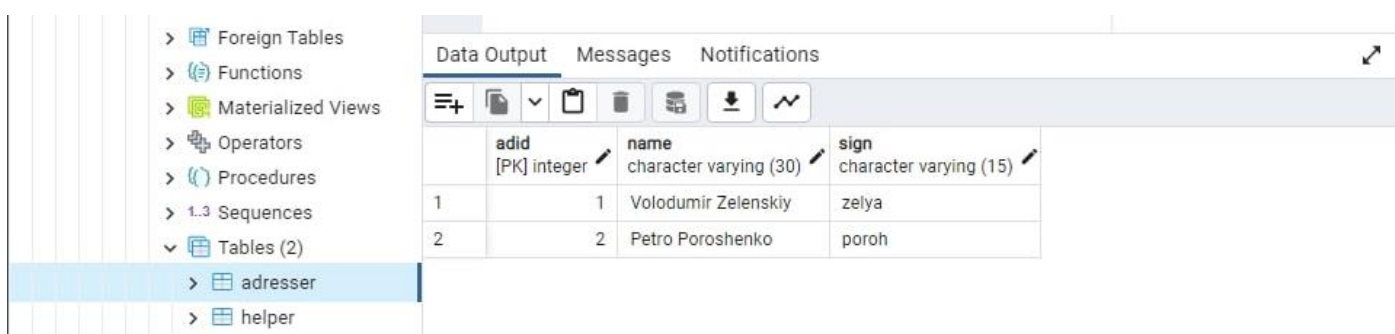


Рисунок 2.16 Вигляд створеної таблиці

Перед виходом з цього режиму необхідно обов'язково зберігати зміни.

Усі зміни, виконані другим методом відображаються в Query history, де можна переглянути згенерований для них скрипт. Утім, якщо заповнювати таблицю першим методом, необхідно ввести наступний скрипт.

Скрипт заповнення таблиці *helper*:

```
INSERT INTO public.helper (
hid, name, sign, boss_id) VALUES (
'7'::integer, 'Apircot Agresovich'::character varying, 'huk'::character varying, '1'::bigint)
returning hid;
```

Такий скрипт потрібен під кожен рядок таблиці.

Результати заповнення таблиць можна побачити на рисунках нижче.

- **Таблиця Adressor**

Data Output Messages Notifications			
<div> ≡+ 📄 ▼ 📋 🗑️ 🗄️ ⬇️ 📈 </div>			
	adid [PK] integer	name character varying (30)	sign character varying (15)
1	1	Volodumir Zelenskiy	zelya
2	2	Petro Poroshenko	poroh
3	3	Vova Putler	killer
4	4	miss Fidanyan	wOmen
5	5	Kolyasik	chel
6	6	Marichka	maha

Рисунок 2.17 Вигляд створеної таблиці Adressor

- **Таблиця Helper**

Data Output Messages Notifications				
<div> ≡+ 📄 ▼ 📋 🗑️ 🗄️ ⬇️ 📈 </div>				
	hid [PK] integer	name character varying (30)	sign character varying (15)	boss_id bigint
1	1	Sofiia Pavlova	zerorchik	1
2	2	Sofiia Hohol	Stresika	2
3	3	Did Mazaylo	dead	1
4	4	Apostol Petro	svyatuy	2
5	5	Chupakabra Bolotna	4upick	1
6	6	Bobik Zdoch	rip	2
7	7	Apircot Agresovich	huk	1

Рисунок 2.18 Вигляд створеної таблиці Helper

- Таблиця Letter

Data Output Messages Notifications

	lid [PK] integer	date character (10)	is_ans boolean	date_ans character (10)	thema_id bigint	sender_id bigint	receiver_id bigint	helper_id bigint
1	1	20/10/2022	false	0	1	1	2	1
2	2	24/02/2022	false	0	2	2	1	2
3	3	01/08/2021	true	15/01/2022	3	3	5	4
4	4	12/12/2021	true	29/12/2021	5	6	4	3
5	5	13/01/2022	false	0	6	4	3	6

Рисунок 2.19 Вигляд створеної таблиці Letter

- Таблиця Thema

Data Output Messages Notifications

	tid [PK] integer	name character varying (50)
1	1	Kiilling Pootin
2	2	Burning Kreml
3	3	Kiilling rosnya
4	4	Bombing Krimea bridge
5	5	Fixing bomb shelter
6	6	Opening Chreschatuk ...
7	7	Changing siren signal

Рисунок 2.20 Вигляд створеної таблиці Thema

Усі створені таким чином таблиці можна завантажити на пРисунотрій у форматі .csv таблиць.

2.5. Редагування таблиць

Для редагування введених даних у комірці таблиці потрібно викоРисунотати наступний скрипт.

Скрипт перезапису рядка у таблиці **helper**:

```
UPDATE public.letter SET
receiver_id = '3'::bigint, sender_id = '4'::bigint WHERE
lid = 5;
```

2.6. Створення зв'язків між таблицями та зміна структури

Для створення зв'язків між двома таблицями необхідно відповідно до типу зв'язку додати до цих таблиць рядок Foreign key.

Між таблицями letter та thema зв'язок 1:1, тому достатньо до таблиці letter додати атрибут thema_id, к це показано нижче.

Скрипт додавання зв'язуючого рядка до таблиці **letter**:

```
ALTER TABLE IF EXISTS public.letter
ADD CONSTRAINT thema_id FOREIGN KEY (thema_id)
REFERENCES public.thema (tid) MATCH SIMPLE
ON UPDATE NO ACTION
ON DELETE NO ACTION
NOT VALID;
```

2.7. Створення ERD діаграми



Для цього після створення таблиць необхідно натиснути у верхній панелі інструментів Tools > ERD Tool і перетягнути готові таблиці в зону редагування.

Наступним кроком потрібно відобразити зв'язки між таблицями. Для цього необхідно подвійним кліком по відображенню таблиці перейти у верхньому вискакуючому меню в Constraints > Foreign Key і додати посилання на відповідні таблиці (рисунок 2.21).

Table: helper (public) X

General Columns Advanced Constraints

Primary Key Foreign Key Unique

	Name	Columns	Referenced Table
 	boss_id	(boss_id) -> (adid)	(public) adresser

Close
Reset
Save

Рисунок 2.21 Додавання зв'язків

Виконаємо цю операцію для всіх відображень таблиць і отримаємо наступну діаграму (рисунок 2.22).

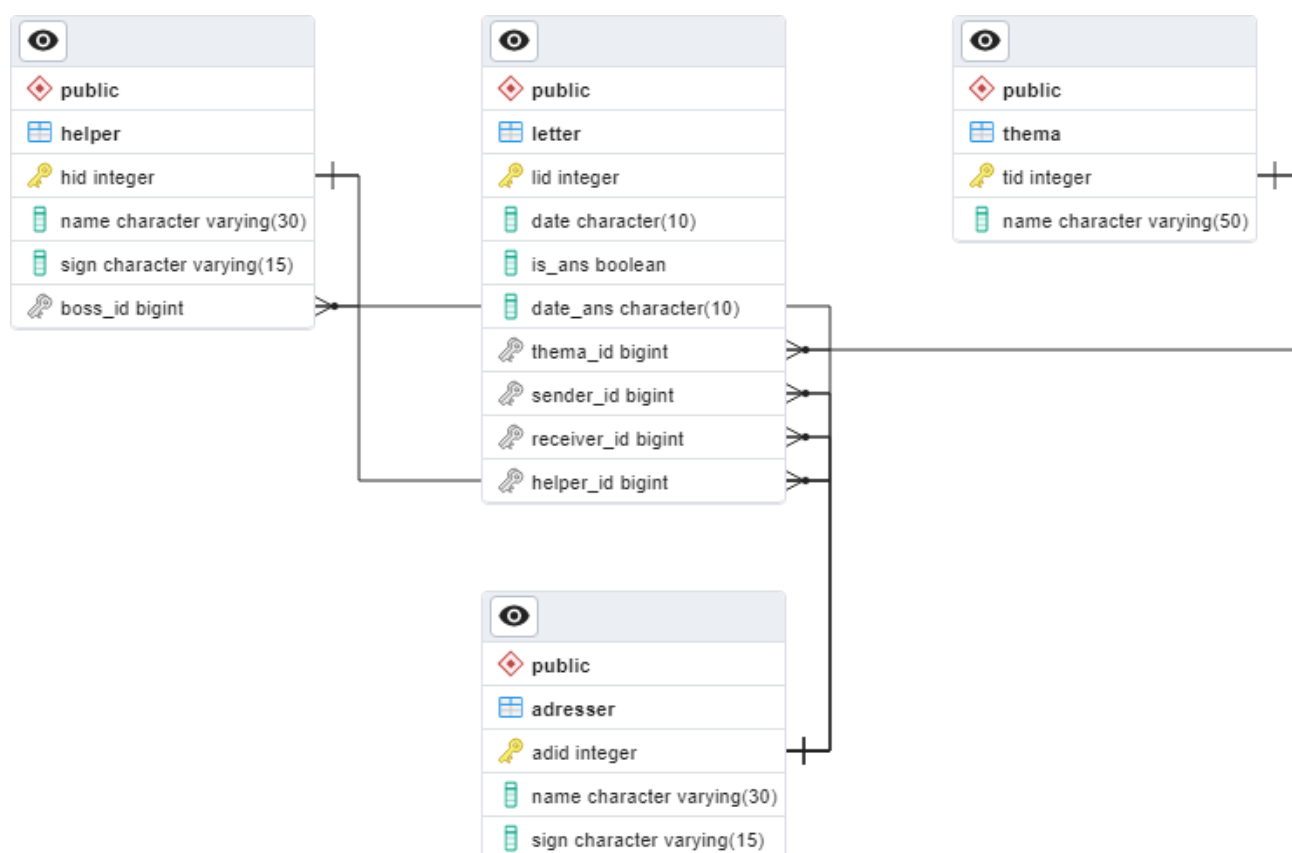


Рисунок 2.22 ERD діаграма Приймальні депутата

3. СТВОРЕННЯ ЗАПИТІВ У POSTGRESQL

3.1. Запити від двох або більше таблиць

Запит 1: Вибірка помічників, що допомагають Зеленському.

Таблиця **helper** до:

Data Output Messages Notifications				
	hid [PK] integer	name character varying (30)	sign character varying (15)	boss_id bigint
1	1	Sofiia Pavlova	zerorchik	1
2	2	Sofiia Hohol	Stresika	2
3	3	Did Mazaylo	dead	1
4	4	Apostol Petro	svyatuy	2
5	5	Chupakabra Bolotna	4upick	1
6	6	Bobik Zdoch	rip	2
7	7	Apircot Agresovich	huk	1

Рисунок 3.1 Вигляд таблиці helper

Таблиця **addresser** до:

Data Output Messages Notifications			
	adid [PK] integer	name character varying (30)	sign character varying (15)
1	1	Volodimir Zelenskiy	zelya
2	2	Petro Poroshenko	poroh
3	3	Vova Putler	killer
4	4	miss Fidanyan	wOmen
5	5	Kolyasik	chel
6	6	Marichka	maha

Рисунок 3.2 Вигляд таблиці addresser

SQL-запит:

```
// вибірка помічників, що допомагають Зеленському
SELECT
    name,
    sign
FROM
    helper
WHERE
    boss_id =
    (
        SELECT
            adid
        FROM
            adresser
        WHERE
            name = 'Volodumir Zelenskiy'
    )
```

Результат:

Query	Query History	
1	SELECT	
2	name,	
3	sign	
4	FROM	
5	helper	
6	WHERE	
7	boss_id =	
8	(
9	SELECT	
10	adid	
11	FROM	
12	adresser	
13	WHERE	
14	name = 'Volodumir Zelenskiy'	
15)	
16		

Data Output	Messages	Notifications
<div> <div>≡+</div> <div>📄</div> <div>▼</div> <div>📋</div> <div>🗑️</div> <div>🗄️</div> <div>⬇️</div> <div>📈</div> </div>		
	name character varying (30) 🔒	sign character varying (15) 🔒
1	Sofia Pavlova	zerorchik
2	Did Mazaylo	dead
3	Chupakabra Bolotna	4upick
4	Apircot Agresovich	huk

Рисунок 3.3 Результат виконання запиту

Запит 2: Вибірка тем, на які є хоча б один лист.

Таблиця **thema** до:

Data Output			Messages	Notifications
	tid [PK] integer	name character varying (50)		
1	1	Killing Pootin		
2	2	Burning Kreml		
3	3	Kiilling rosnya		
4	4	Bombing Krimea bridge		
5	5	Fixing bomb shelter		
6	6	Opening Chreschatuk ...		
7	7	Changing siren signal		

Рисунок 3.4 Вигляд таблиці thema

Таблиця **letter** до:

Data Output

Messages

Notifications

	lid [PK] integer	date date	is_ans boolean	date_ans date	thema_id bigint	sender_id bigint	receiver_id bigint	helper_id bigint
1	1	2022-10-20	false	[null]	1	1	2	1
2	2	2022-02-24	false	[null]	2	2	1	2
3	3	2021-08-01	true	2022-01-15	3	3	5	4
4	4	2021-12-12	true	2021-12-29	5	6	4	3
5	5	2022-01-13	false	[null]	6	4	3	6

Рисунок 3.5 Вигляд таблиці letter

SQL-запит:

```
// вибірка тем, на які є хоча б один лист
SELECT
  tid,
  name
FROM
  thema
WHERE
  EXISTS
  (
    SELECT
      thema_id
    FROM
      letter
    WHERE
      letter.thema_id=thema.tid
  )
```

Результат:

The screenshot shows a database query editor with two tabs: 'Query' and 'Query History'. The 'Query' tab is active, displaying a SQL query. Below the query, there are tabs for 'Data Output', 'Messages', and 'Notifications'. The 'Data Output' tab is active, showing a table of results. The table has two columns: 'tid' (integer, primary key) and 'name' (character varying (50)). The results are as follows:

	tid [PK] integer	name character varying (50)
1	1	Killing Pootin
2	2	Burning Kreml
3	3	Kiilling rosnya
4	5	Fixing bomb shelter
5	6	Opening Chreschatuk ...

Рисунок 3.6 Результат виконання запиту

3.2. Групуючі запити

Запит 1: Визначимо сумарний час очікування відповіді на листи по адресату.

Вигляд таблиці **letter** можна побачити на Рисунок 3.5.

Спочатку обрахуємо час очікування відповіді на кожен лист.

SQL-запит:

```

/ обрахунок часу очікування відповіді на лист
SELECT
    lid,
    date_ans::DATE - date::DATE AS ans_termin
FROM
    letter
WHERE
    is_ans IS true

```

Результат:

Query		Query History	
1	SELECT		
2	lid,		
3	date_ans::DATE - date::DATE AS ans_termin		
4	FROM		
5	letter		
6	WHERE		
7	is_ans IS true		

Data Output		Messages		Notifications	
	lid		ans_termin		
	[PK] integer		integer		
1	3		167		
2	4		17		
3	1		3		
4	2		28		

Рисунок 3.7 Результат виконання запиту

Знайдемо суму часу очікування по кожному адресату (незалежно від того, чи є він відправником, чи отримувачем).

SQL-запит:

```

// знаходження сумарного часу очікування відповіді на листи по адресату
SELECT
    addresser.name,
    SUM (date_ans::DATE - date::DATE) ans_termin
FROM
    letter
INNER JOIN
    addresser
    ON (letter.sender_id = addresser.adid OR letter.receiver_id = addresser.adid)
WHERE
    is_ans IS true
GROUP BY
    adid;

```

Результат:

Query









Query History

```
1 SELECT
2     adresser.name,
3     SUM (date_ans::DATE - date::DATE) ans_termin
4 FROM
5     letter
6 INNER JOIN
7     adresser
8     ON (letter.sender_id = adresser.adid OR letter.receiver_id = adresser.adid)
9 WHERE
10     is_ans IS true
11 GROUP BY
12     adid;
```

Data Output

Messages

Notifications





	name character varying (30) 	ans_termin bigint 
1	Volodumir Zelenskiy	31
2	Petro Poroshenko	31
3	Vova Putler	167
4	miss Fidanyan	17
5	Kolyasik	167
6	Marichka	17

Рисунок 3.8 Результат виконання запиту

Запит 2: Знайдемо депутатів, які мають більше 1 помічника.

Спочатку створимо запит, що підрахує скільки помічників є у кожного депутата (якщо вони є).

SQL-запит:

```

// кількість помічників у депутатів
SELECT
    adresser.name,
    COUNT (adid) AS helpers
FROM
    adresser
LEFT JOIN helper
    ON helper.boss_id = adresser.adid
GROUP BY
    adid
ORDER BY adid;

```

Результат:

Query	Query History
1	SELECT
2	addresser.name,
3	COUNT (adid) AS helpers
4	FROM
5	addresser
6	LEFT JOIN helper
7	ON helper.boss_id = addresser.adid
8	GROUP BY
9	adid
10	ORDER BY adid;

Data Output	Messages	Notifications
<div> <div>≡+</div> <div>📄</div> <div>▼</div> <div>📋</div> <div>🗑️</div> <div>🗑️</div> <div>📦</div> <div>⬇️</div> <div>📈</div> </div>		
	name	helpers
	character varying (30) 🔒	bigint 🔒
1	Volodumir Zelenskiy	4
2	Petro Poroshenko	3
3	Vova Putler	1
4	miss Fidanyan	1
5	Kolyasik	1
6	Marichka	1

Рисунок 3.9 Результат виконання запиту

Тепер виберемо ті записи, де кількість помічників буде більше 1.

SQL-запит:

```
// депутати, у яких більше 1 помічника
SELECT
    addresser.name,
    COUNT (adid) AS helpers
FROM
    addresser
LEFT JOIN helper
    ON helper.boss_id = addresser.adid
GROUP BY
    adid
HAVING
    COUNT (adid) > 1;
```

Результат:

Query	Query History
1	SELECT
2	addresser.name,
3	COUNT (adid) AS helpers
4	FROM
5	addresser
6	LEFT JOIN helper
7	ON helper.boss_id = addresser.adid
8	GROUP BY
9	adid
10	HAVING
11	COUNT (adid) > 1;

Data Output	Messages	Notifications
<div> <div>≡</div> <div>📄</div> <div>▼</div> <div>📋</div> <div>🗑️</div> <div>🗄️</div> <div>⬇️</div> <div>📈</div> </div>		
	name character varying (30) 🔒	helpers bigint 🔒
1	Petro Poroshenko	3
2	Volodumir Zelenskiy	4

Рисунок 3.10 Результат виконання запиту

3.3. Аналітичні запити

Запит 1: Подання, яке виводить всі елементи таблиці letter (листи) у порядку дат їх написання.

Вигляд таблиці **letter** можна побачити на Рисунок 3.5.

SQL-запит:

```
// сортування листів по даті
SELECT
    lid,
    thema_id,
    sender_id,
    receiver_id,
    helper_id
FROM
    letter
ORDER BY
    date;
```

Результат:

Query

Query History

1

2

3

4

5

6

7

8

9

10

SELECT

lid,

thema_id,

sender_id,

receiver_id,

helper_id

FROM

letter

ORDER BY

date;

Data Output

Messages

Notifications

≡+

▼

	lid [PK] integer	thema_id bigint	sender_id bigint	receiver_id bigint	helper_id bigint
1	3	3	3	5	4
2	4	5	6	4	3
3	5	6	4	3	6
4	1	1	1	2	1
5	2	2	2	1	2

Рисунок 3.11 Результат виконання запиту

Запит 2: Вибірка листів, відповіді на які не було отримано.

Вигляд таблиці **letter** можна побачити на Рисунок 3.5.

SQL-запит:

```
// вибірка листів, відповіді на які не було отримано
SELECT
    lid,
    date,
    thema_id,
    sender_id,
    receiver_id,
    helper_id
FROM
    letter
WHERE
    date_ans IS NULL
```

Результат:

Query

Query History

1

SELECT

2

lid,

3

date,

4

thema_id,

5

sender_id,

6

receiver_id,

7

helper_id

8

FROM

9

letter

10

WHERE

11

date_ans IS NULL

Data Output

Messages

Notifications

Рисунок 3.12 Результат виконання запиту

3.4. Запити на зміну

Запити на вставку

Запит 1: Продублюємо записи про депутатів, що мають більше 3 помічників.

Таблиця **deputy_helpers** до:

Data Output	Messages	Notifications																					
<table> <thead> <tr> <th></th><th>name character varying (30)</th><th>helpers bigint</th></tr> </thead> <tbody> <tr> <td>1</td><td>Volodumir Zelenskiy</td><td>4</td></tr> <tr> <td>2</td><td>Petro Poroshenko</td><td>3</td></tr> <tr> <td>3</td><td>Vova Putler</td><td>1</td></tr> <tr> <td>4</td><td>miss Fidanyan</td><td>1</td></tr> <tr> <td>5</td><td>Kolyasik</td><td>1</td></tr> <tr> <td>6</td><td>Marichka</td><td>1</td></tr> </tbody> </table>		name character varying (30)	helpers bigint	1	Volodumir Zelenskiy	4	2	Petro Poroshenko	3	3	Vova Putler	1	4	miss Fidanyan	1	5	Kolyasik	1	6	Marichka	1		
	name character varying (30)	helpers bigint																					
1	Volodumir Zelenskiy	4																					
2	Petro Poroshenko	3																					
3	Vova Putler	1																					
4	miss Fidanyan	1																					
5	Kolyasik	1																					
6	Marichka	1																					

Рисунок 3.13 Вигляд таблиці deputy_helpers

SQL-запит:

```
// продублювати записи про депутатів, що мають більше 3 помічників
INSERT INTO
    deputy_helpers (name, helpers)
SELECT
    name,
    helpers
FROM
    deputy_helpers
WHERE
    helpers > 3
ORDER BY helpers DESC;
```

Результат:

Query Query History

```

1  INSERT INTO
2      deputy_helpers (name, helpers)
3  SELECT
4      name,
5      helpers
6  FROM
7      deputy_helpers
8  WHERE
9      helpers > 3;
```

Data Output Messages Notifications

INSERT 0 1

Query returned successfully in 60 msec.

Рисунок 3.14 Запит

	name	helpers
	character varying (30)	bigint
1	Volodumir Zelenskiy	4
2	Volodumir Zelenskiy	4
3	Petro Poroshenko	3
4	Marichka	1
5	miss Fidanyan	1
6	Vova Putler	1
7	Kolyasik	1

Рисунок 3.15 Результат виконання запиту

Запити на оновлення

Запит 1: Замінімо нульові рядки іншими значеннями.

SQL-запит:

```
// заміна нульових рядків
UPDATE
  letter
SET
  date_ans = '2222-12-22'::date
WHERE
  date_ans IS NULL;
```

Результат:

Query

Query History

1

UPDATE

2

letter

3

SET

4

date_ans = '2222-12-22'::date

5

WHERE

6

date_ans IS NULL;

Data Output

Messages

Notifications

UPDATE 3

Query returned successfully in 347 msec.

	lid	date	is_ans	date_ans	thema_id	sender_id	receiver_id	helper_id
	[PK] integer	date	boolean	date	bigint	bigint	bigint	bigint
1	1	2022-10-20	false	2222-12-22	1	1	2	1
2	2	2022-02-24	false	2222-12-22	2	2	1	2
3	3	2021-08-01	true	2022-01-15	3	3	5	4
4	4	2021-12-12	true	2021-12-29	5	6	4	3
5	5	2022-01-13	false	2222-12-22	6	4	3	6

Рисунок 3.16 Результат виконання запиту

Запит 2: Оновимо отримувачів листів шляхом додавання до айді отримувача число.

Таблиця **letter** до:

Data Output

Messages

Notifications

	lid [PK] integer	date date	is_ans boolean	date_ans date	thema_id bigint	sender_id bigint	receiver_id bigint	helper_id bigint
1	1	2022-10-20	true	2022-10-23	1	1	2	1
2	2	2022-02-24	true	2022-03-24	2	2	1	2
3	3	2021-08-01	true	2022-01-15	3	3	5	4
4	4	2021-12-12	true	2021-12-29	5	6	4	3
5	5	2022-01-13	false	[null]	6	4	3	6

Рисунок 3.17 Вигляд таблиці letter

SQL-запит:

```
// оновити отримувачів листів, чиї листи було відправлено в 2021 році шляхом додавання до айді отримувача
число
UPDATE
  letter
SET
  receiver_id = receiver_id + 1
WHERE
  EXTRACT(YEAR FROM date) IS 2021;
```

Результат:

Query	Query History
1	UPDATE
2	letter
3	SET
4	receiver_id = receiver_id + 1
5	WHERE
6	EXTRACT(YEAR FROM date) = 2021;

Data Output	Messages	Notifications
UPDATE 2		
Query returned successfully in 229 msec.		

Рисунок 3.18 Запит

Data Output

Messages

Notifications

	lid [PK] integer	date date	is_ans boolean	date_ans date	thema_id bigint	sender_id bigint	receiver_id bigint	helper_id bigint
1	1	2022-10-20	true	2022-10-23	1	1	2	1
2	2	2022-02-24	true	2022-03-24	2	2	1	2
3	3	2021-08-01	true	2022-01-15	3	3	6	4
4	4	2021-12-12	true	2021-12-29	5	6	5	3
5	5	2022-01-13	false	[null]	6	4	3	6

Рисунок 3.19 Результат виконання запиту

Запити на видалення

Запит 1: Видаємо записи про помічників, які допомагають не Зеленському та не Порошенку.

Таблиця **helper** до:

Data Output

Messages

Notifications

≡+

▼

	<div>hid</div> <div>[PK] integer</div> <div></div>	<div>name</div> <div>character varying (30)</div> <div></div>	<div>sign</div> <div>character varying (15)</div> <div></div>	<div>boss_id</div> <div>bigint</div> <div></div>
1	1	Sofiia Pavlova	zerorchik	1
2	2	Sofiia Hohol	Stresika	2
3	3	Did Mazaylo	dead	1
4	4	Apostol Petro	svyatuy	2
5	5	Chupakabra Bolotna	4upick	1
6	6	Bobik Zdoch	rip	2
7	7	Apircot Agresovich	huk	1
8	8	htos	avos	3

Рисунок 3.20 Вигляд таблиці helper

SQL-запит:

```
// видалити записи про помічників, які допомагають не Зеленському та не Порошенку
DELETE
FROM
    helper
WHERE
    boss_id != 1 AND boss_id != 2;
```

Результат:

The screenshot shows a database query interface with two tabs: 'Query' and 'Query History'. The 'Query' tab is active, displaying a SQL statement:

```

1 DELETE
2 FROM
3     helper
4 WHERE
5     boss_id != 1 AND boss_id != 2;

```

Below the query, there are three tabs: 'Data Output', 'Messages', and 'Notifications'. The 'Messages' tab is active, showing the message 'DELETE 1' and 'Query returned successfully in 80 msec.'

Рисунок 3.21 Запит

The screenshot shows a database query interface with three tabs: 'Data Output', 'Messages', and 'Notifications'. The 'Data Output' tab is active, displaying a table with 7 rows and 5 columns. The columns are: 'hid' (integer, PK), 'name' (character varying (30)), 'sign' (character varying (15)), and 'boss_id' (bigint). The table contains the following data:

	hid [PK] integer	name character varying (30)	sign character varying (15)	boss_id bigint
1	1	Sofiia Pavlova	zerorchik	1
2	2	Sofiia Hohol	Stresika	2
3	3	Did Mazaylo	dead	1
4	4	Apostol Petro	svyatuy	2
5	5	Chupakabra Bolotna	4upick	1
6	6	Bobik Zdoch	rip	2
7	7	Apircot Agresovich	huk	1

Рисунок 3.22 Результат виконання запиту

Тригер з функціями INSERT та DELETE

Створимо також тригер з функціями INSERT та DELETE.

Задача: Якщо вже є лист на таку тему і з таким адресатом, то дата отримання такого листа повинна бути більшою за дату отримання його попередника.

SQL-запит (Тригерна функція):

Query	Query History
1	CREATE OR REPLACE FUNCTION letter_function()
2	RETURNS TRIGGER
3	LANGUAGE PLPGSQL
4	AS
5	\$\$
6 ▼	BEGIN
7 ▼	IF EXISTS
8	(
9	SELECT FROM
10	letter
11	WHERE
12	letter.thema_id = NEW.thema_id AND
13	letter.sender_id = NEW.sender_id AND
14	letter.date > NEW.date
15)
16	THEN
17	DELETE
18	FROM
19	letter
20	WHERE
21	lid = NEW.lid;
22	END IF;
23	RETURN NULL;
24	END;
25	\$\$

Data Output	Messages	Notifications
CREATE FUNCTION		
Query returned successfully in 39 msec.		

Рисунок 3.23 Тригерна функція

SQL-запит (Тригер):

The screenshot shows a database query editor with two tabs: 'Query' and 'Query History'. The 'Query' tab is active, displaying the following SQL code:

```
1 CREATE OR REPLACE TRIGGER
2   letter_insert
3   AFTER INSERT
4   ON letter
5   REFERENCING NEW TABLE AS NEW
6   FOR EACH ROW
7   EXECUTE FUNCTION letter_function();
```

Below the query editor, there are three tabs: 'Data Output', 'Messages', and 'Notifications'. The 'Messages' tab is active, showing the message:

```
CREATE TRIGGER

Query returned successfully in 41 msec.
```

Рисунок 3.24 Тригер

Зробимо 2 пробні запити на перевірку:

- якщо дата отримання нового листа менша за дату існуючого (помилка);
- якщо дата нового листа більша за попередника (успіх).

Створимо лист аналогічний до першого запису:

	lid [PK] integer	date date	is_ans boolean	date_ans date	thema_id bigint	sender_id bigint	receiver_id bigint	helper_id bigint
1	1	2022-10-20	true	2022-10-23	1	1	2	1

, зменшивши день отримання листа на 1 (19).

Перевірка (помилка):

The screenshot shows a database query editor with two tabs: 'Query' and 'Query History'. The 'Query' tab is active, displaying the following SQL code:

```
1 INSERT INTO
2   letter
3   (lid, date, is_ans, date_ans, thema_id, sender_id, receiver_id, helper_id)
4   VALUES
5   (6, '2022-10-19'::date, false, null, 1, 1, 2, 1);
```

Below the query editor, there are three tabs: 'Data Output', 'Messages', and 'Notifications'. The 'Messages' tab is active, showing the message:

```
INSERT 0 1

Query returned successfully in 39 msec.
```

Data Output Messages Notifications

	lid [PK] integer	date date	is_ans boolean	date_ans date	thema_id bigint	sender_id bigint	receiver_id bigint	helper_id bigint
1	1	2022-10-20	true	2022-10-23	1	1	2	1
2	2	2022-02-24	true	2022-03-24	2	2	1	2
3	3	2021-08-01	true	2022-01-15	3	3	5	4
4	4	2021-12-12	true	2021-12-29	5	6	4	3
5	5	2022-01-13	false	[null]	6	4	3	6

Рисунок 3.25 Результат помилкового сценарію для трєгра

Бачимо, що запис не додався.

Збільшимо день отримання листа на 1 (21).

Перевірка (успіх):

Query Query History

```

1 INSERT INTO
2   letter
3   (lid, date, is_ans, date_ans, thema_id, sender_id, receiver_id, helper_id)
4 VALUES
5   (6, '2022-10-21'::date, false, null, 1, 1, 2, 1);

```

Data Output Messages Notifications

INSERT 0 1

Query returned successfully in 39 msec.

Data Output Messages Notifications

	lid [PK] integer	date date	is_ans boolean	date_ans date	thema_id bigint	sender_id bigint	receiver_id bigint	helper_id bigint
1	1	2022-10-20	true	2022-10-23	1	1	2	1
2	2	2022-02-24	true	2022-03-24	2	2	1	2
3	3	2021-08-01	true	2022-01-15	3	3	5	4
4	4	2021-12-12	true	2021-12-29	5	6	4	3
5	5	2022-01-13	false	[null]	6	4	3	6
6	6	2022-10-21	false	[null]	1	1	2	1

Рисунок 3.26 Результат успішного сценарію для тригера

Бачимо, що запис додався.

4. СТВОРЕННЯ ЗАСТОСУВАННЯ НА ASP.NET

4.1. Створення додатку в робочому середовищі ASP.NET

Перед початком створення проекту потрібно встановити у середовищі Visual Studio 2019 пак з всіма налаштуваннями для роботи з ASP.NET Core Web App.

Коли потрібні паки встановлені, можемо переходити до створення проекту.

При створенні проекту оберемо тип «ASP.NET Core Web App».

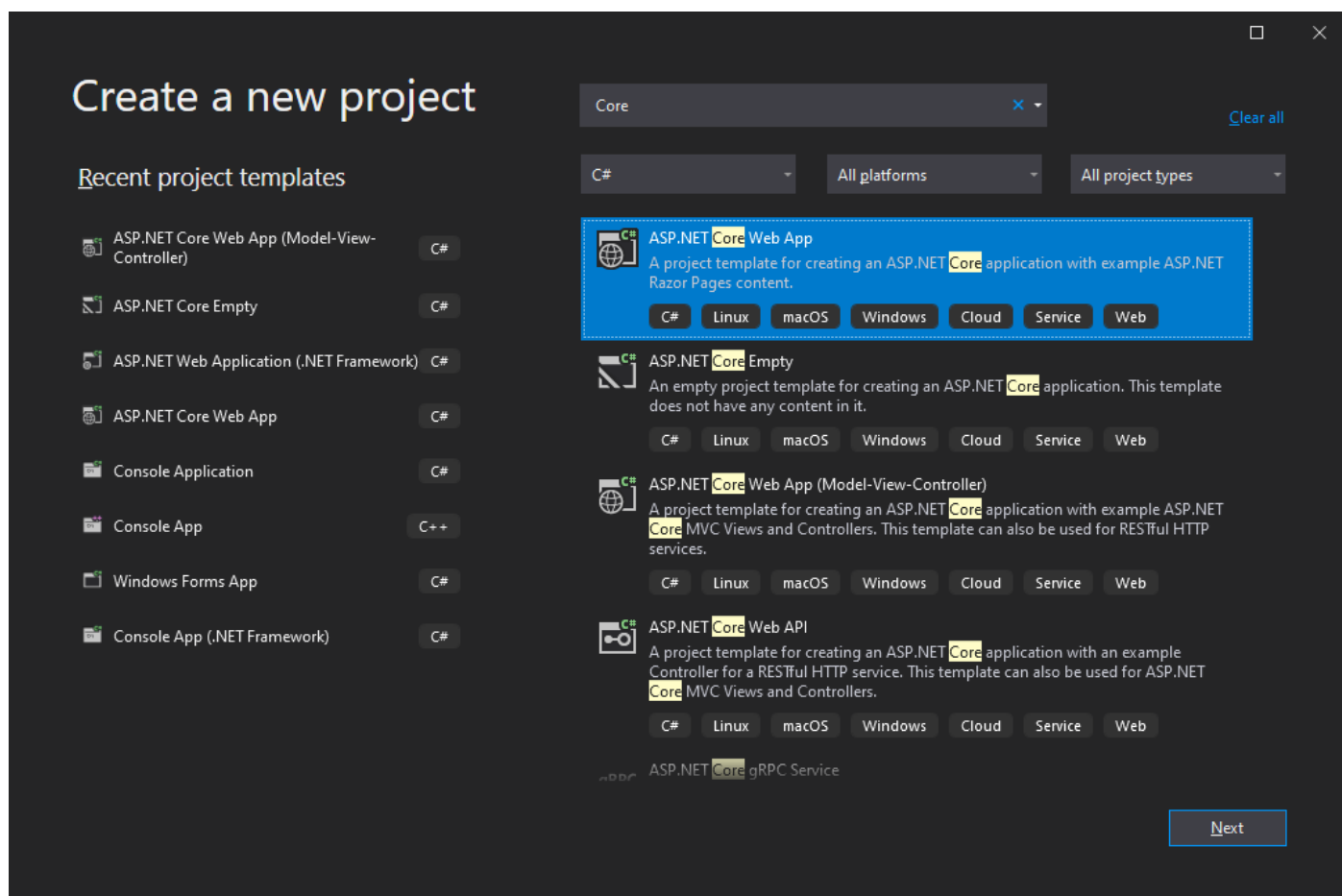
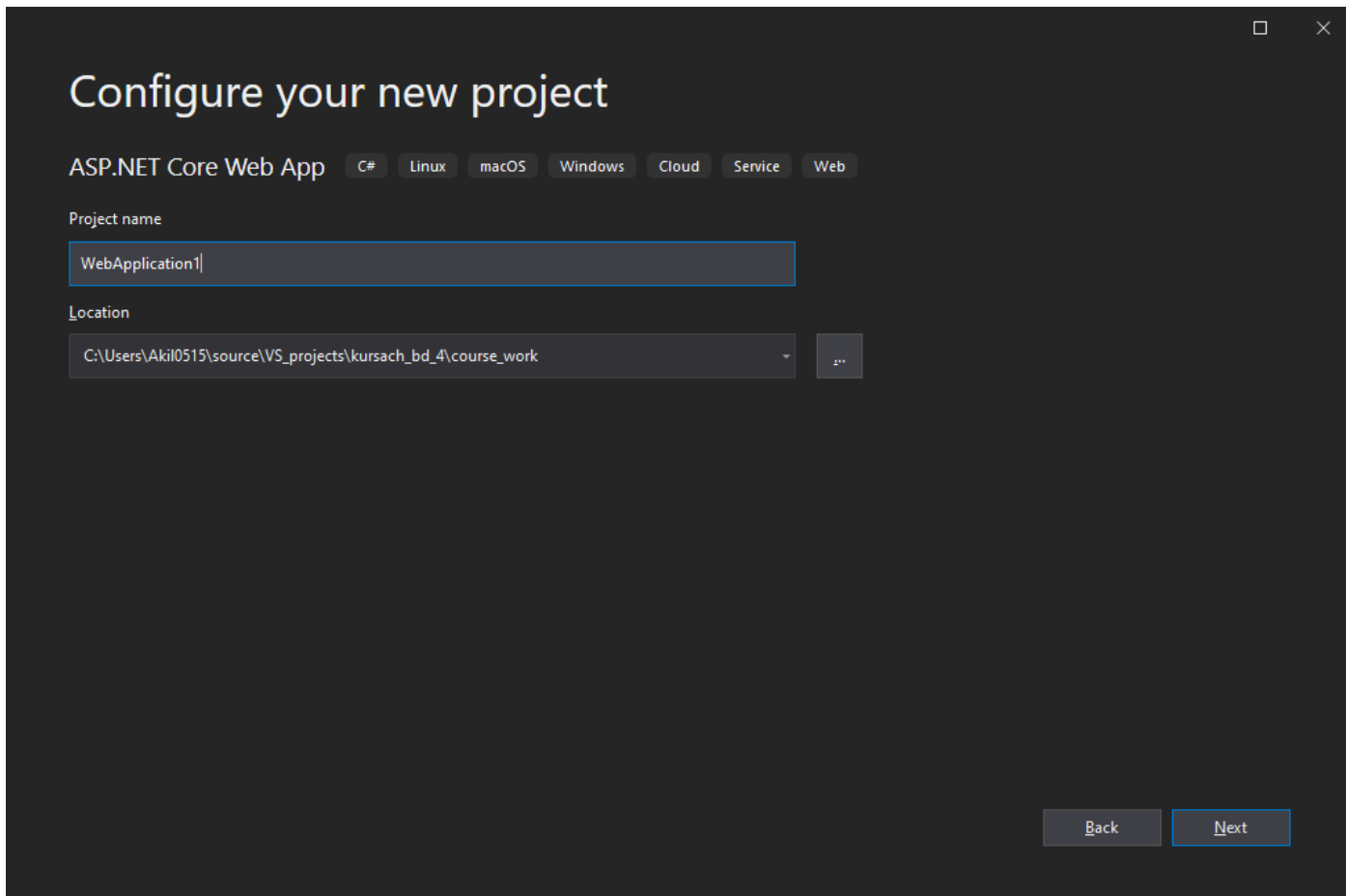


Рисунок 4.1 Вибір типу проекту



Configure your new project

ASP.NET Core Web App C# Linux macOS Windows Cloud Service Web

Project name

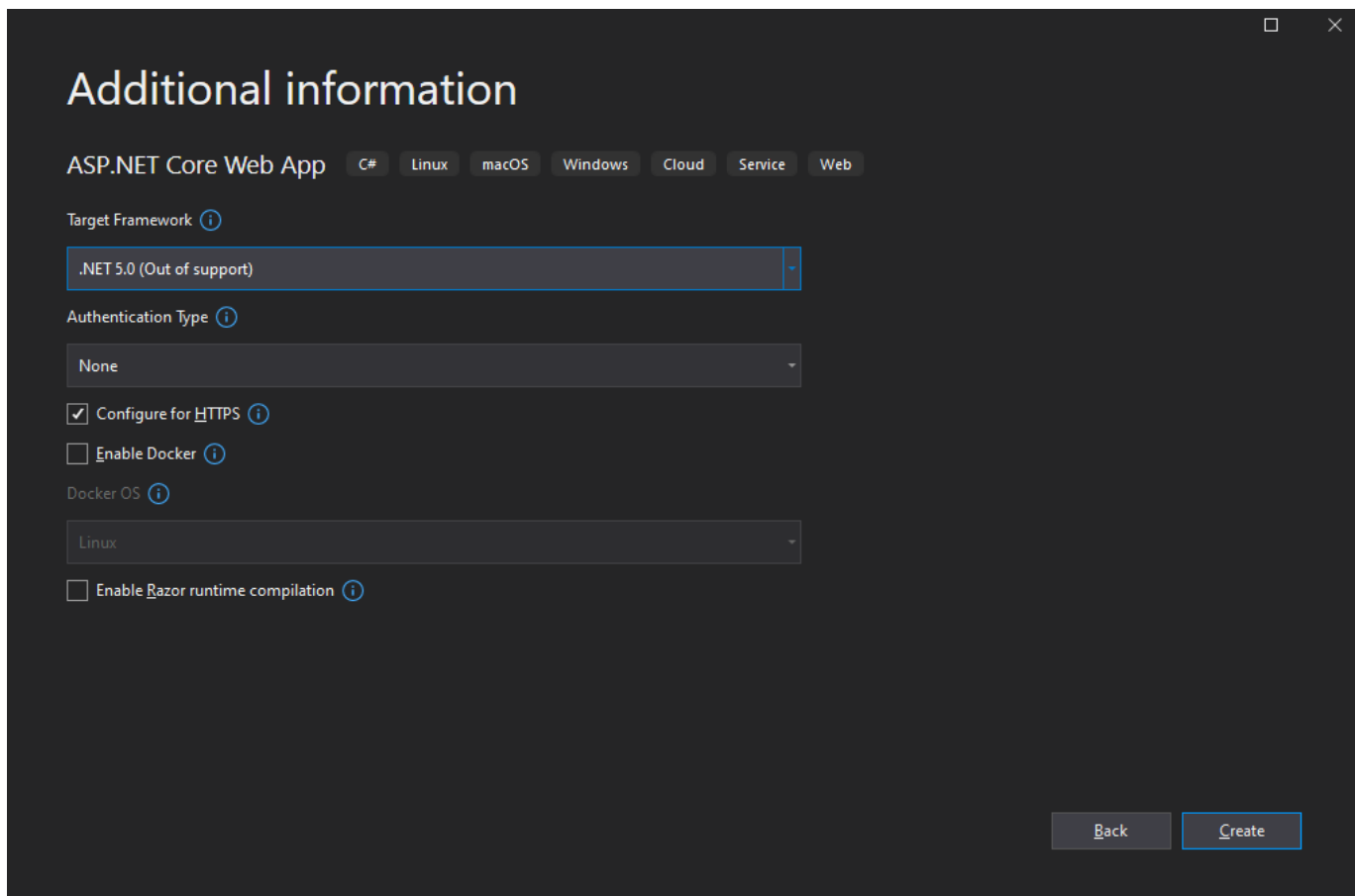
WebApplication1

Location

C:\Users\Akil0515\source\VS_projects\kursach_bd_4\course_work

Back Next

Рисунок 4.2 Назва проекту



Additional information

ASP.NET Core Web App C# Linux macOS Windows Cloud Service Web

Target Framework ⓘ

.NET 5.0 (Out of support)

Authentication Type ⓘ

None

☒ Configure for HTTPS ⓘ

☐ Enable Docker ⓘ

Docker OS ⓘ

Linux

☐ Enable Razor runtime compilation ⓘ

Back Create

Рисунок 4.3 Вибір версії фреймворку

При запуску та компіляції проекту можна відразу побачити головну сторінку на яку ми якраз і під'єднаємо нашу баз даних.

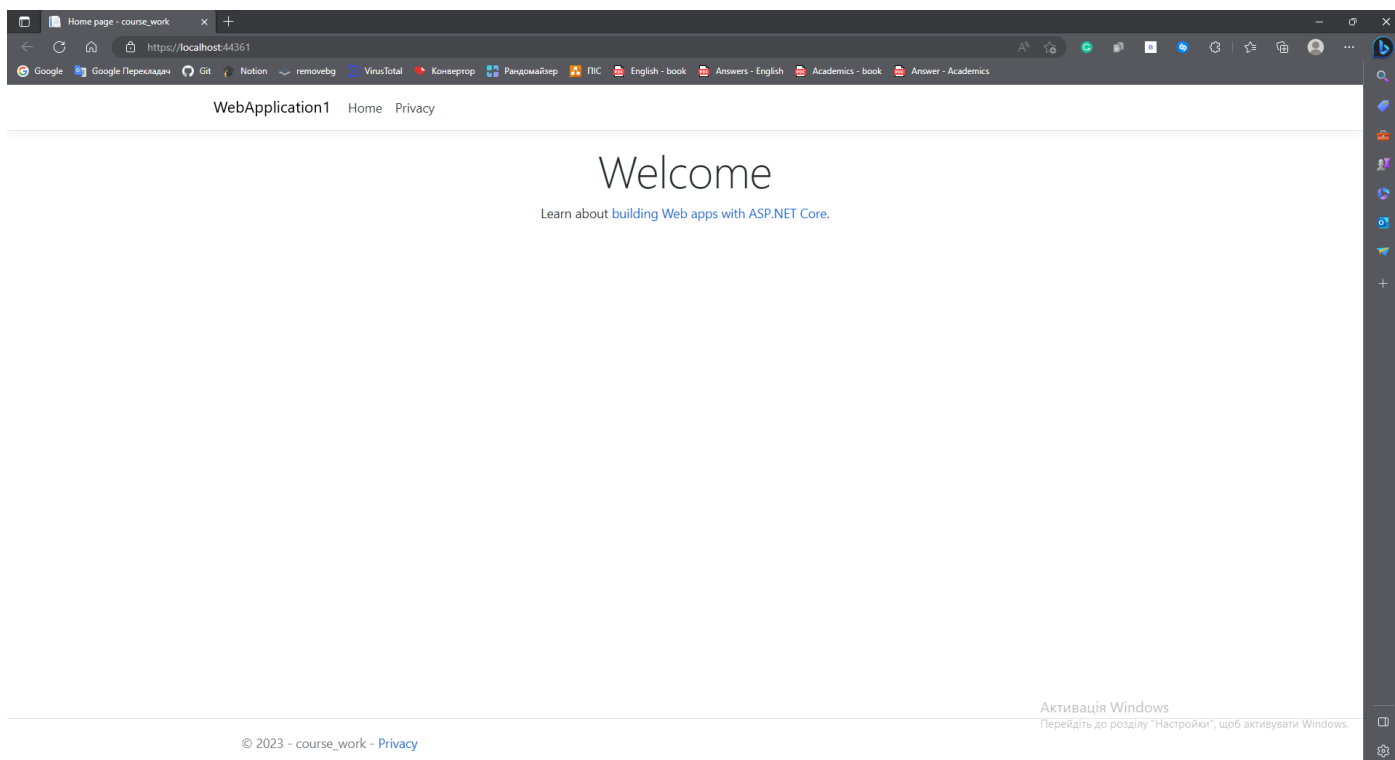


Рисунок 4.4 Початкова сторінка серверу

Для роботи з нашою базою даних в ASP.NET завантажимо додаткові бібліотеки через NuGet Package Manager -> Manage NuGet Packages.

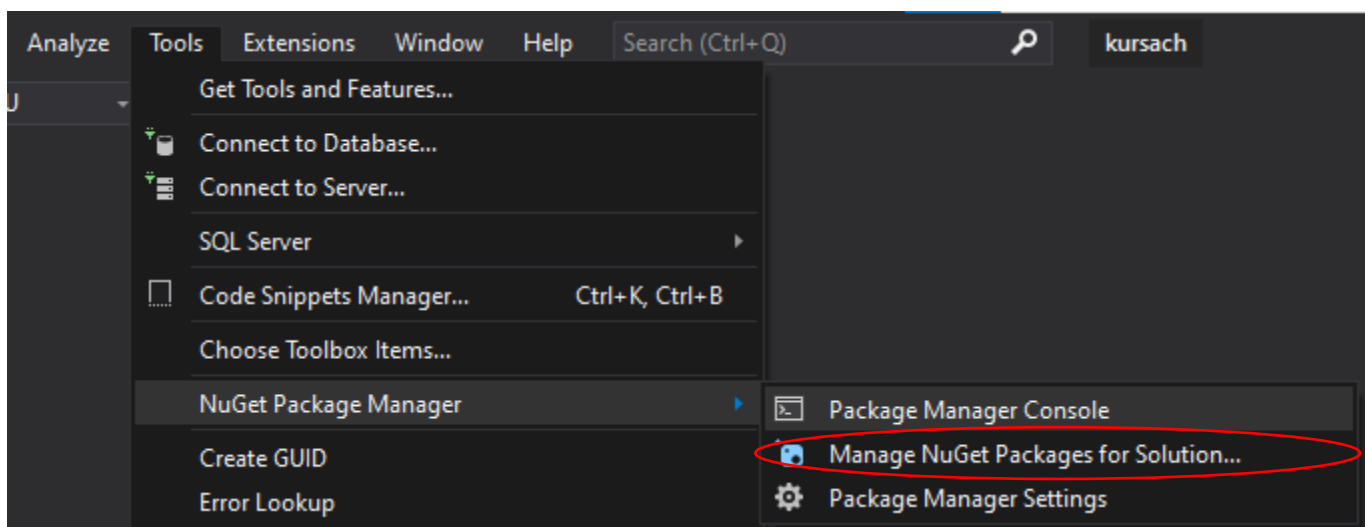


Рисунок 4.5 Використання Manage NuGet Packages

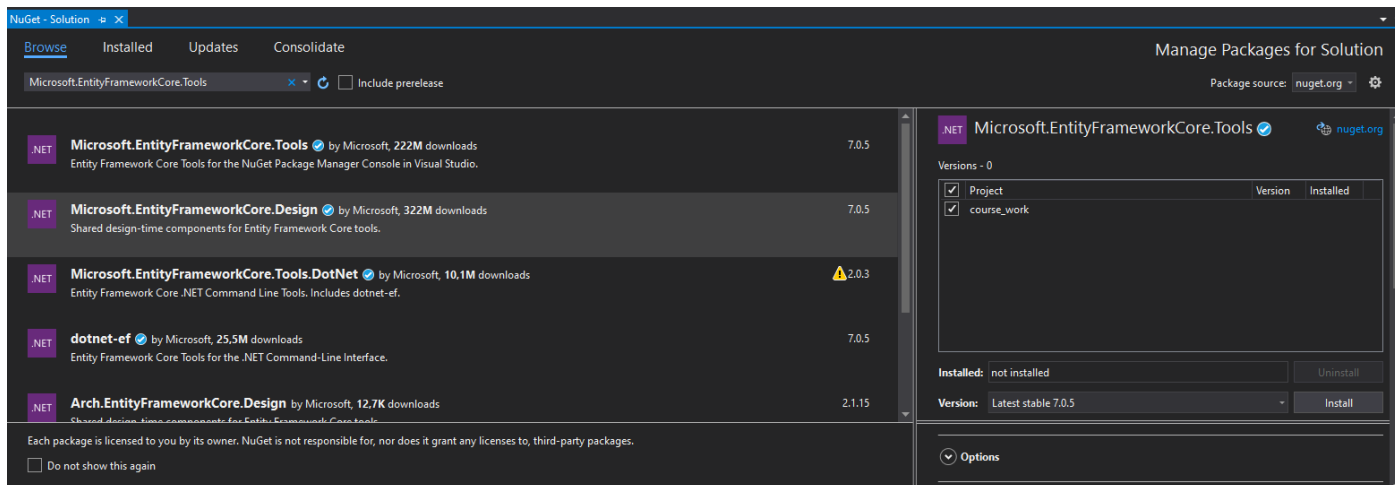


Рисунок 4.6 Завантаження необхідних для роботи пакетів

Тепер за допомогою нижченаведеної команди підв'яжемо наші таблиці зі всіма даними.

Лістинг:

```
Scaffold-DbContext "Host=localhost;Port=5432;Database=lab_2;Username=postgres;Password=Sonya2004;"
Npgsql.EntityFrameworkCore.PostgreSQL -OutputDir Models
```

Результат:

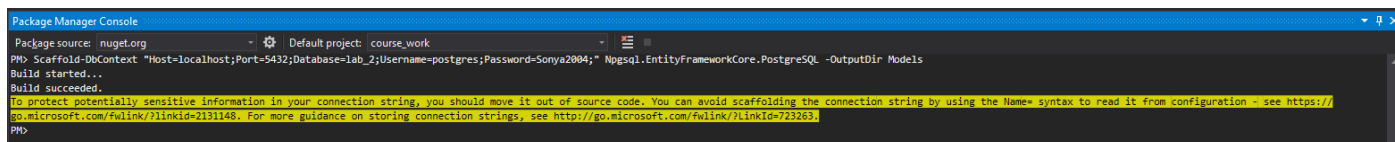


Рисунок 4.7 Успішне підключення до БД

Як результат можемо бачити, що створились наші таблиці з заповненими типами даних, як у PostgreSQL.

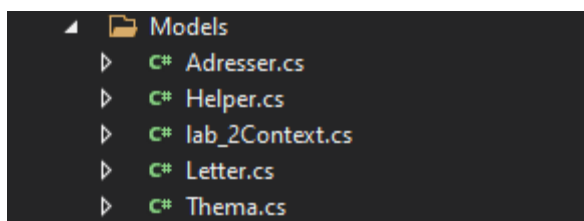


Рисунок 4.8 Успішне створення таблиць з заповненими типами даних

Adresser:

```
using System;
using System.Collections.Generic;

#nullable disable

namespace course_work.Models
{
    public partial class Adresser
    {
        public Adresser()
        {
            Helpers = new HashSet<Helper>();
        }

        public long Adid { get; set; }
        public string Name { get; set; }
        public string Sign { get; set; }

        public virtual ICollection<Helper> Helpers { get; set; }
    }
}
```

Helper:

```
using System;
using System.Collections.Generic;

#nullable disable

namespace course_work.Models
{
    public partial class Helper
    {
        public int Hid { get; set; }
        public string Name { get; set; }
        public string Sign { get; set; }
        public long? BossId { get; set; }

        public virtual Adresser Boss { get; set; }
    }
}
```

Letter:

```
using System;
using System.Collections.Generic;

#nullable disable

namespace course_work.Models
{
    public partial class Letter
    {
        public int Lid { get; set; }
        public DateTime Date { get; set; }
        public bool IsAns { get; set; }
        public DateTime? DateAns { get; set; }
        public long ThemaId { get; set; }
        public long SenderId { get; set; }
        public long ReceiverId { get; set; }
        public long HelperId { get; set; }
    }
}
```

Thema:

```
using System;
using System.Collections.Generic;

#nullable disable

namespace course_work.Models
{
    public partial class Thema
    {
        public int Tid { get; set; }
        public string Name { get; set; }
    }
}
```

Щоб зв'язати програму, контекст даних та БД, додамо у файл appsettings.json рядок підключення до цієї бази даних наступним чином.

Лістинг:

```
"ConnectionStrings": {
  "DefaultConnection": "Server=localhost; Port=5432; Database=lab_2; User Id=postgres;
Password=Sonya2004;"
}
```

Результат:

```
9    "AllowedHosts": "*",
10
11    "ConnectionStrings": {
12      "DefaultConnection": "Server=localhost; Port=5432; Database=lab_2; User Id=postgres; Password=Sonya2004;"
13    }
14  }
```

Рисунок 4.9 Вставлений код у файл appsettings.json

І додамо в файл Startup.cs виклик цього методу наступним чином.

Лістинг:

```
services.AddControllersWithViews();
    services.AddDbContext<lab_2Context>(options =>
options.UseNpgsql(Configuration.GetConnectionString("DefaultConnection")));
```

Результат:

```
25 // This method gets called by the runtime. Use this method to add services to the container.
26 0 references
27 public void ConfigureServices(IServiceCollection services)
28 {
29     services.AddRazorPages();
30     services.AddControllersWithViews();
31     services.AddDbContext<lab_2Context>(options => options.UseNpgsql(Configuration.GetConnectionString("DefaultConnection")));
}
```

Рисунок 4.10 Вставлений код у файл Startup.cs

Застосунок готовий.

4.2. Огляд застосунку

Вигляд застосунку

Після запуску серверу бачимо основну сторінку Номе з описом проекту.

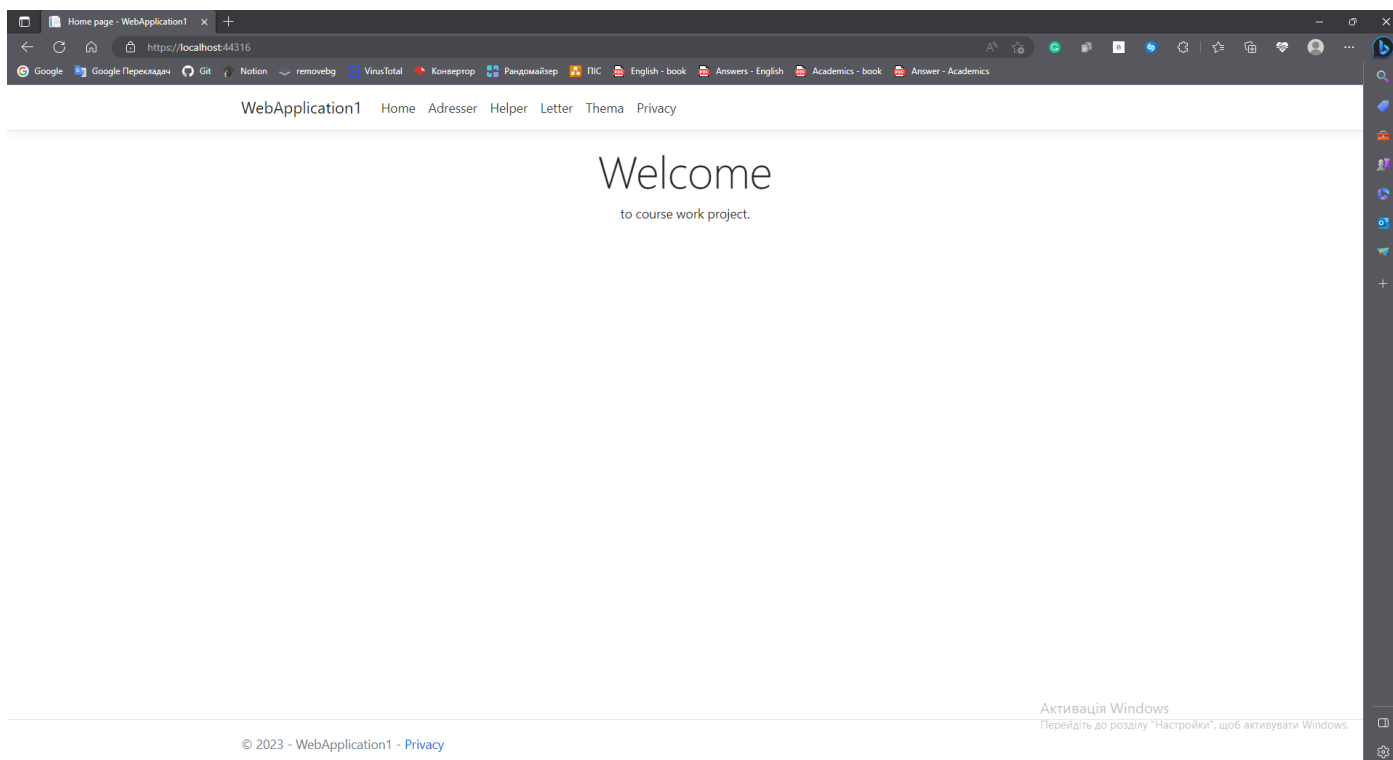


Рисунок 4.11 Вигляд сторінки Номе серверу

А також сторінки з вмістом таблиць PostgreSQL. Виведені таблиці можна редагувати, дивитись деталі введених даних, видаляти та додавати нові дані.

Adresser

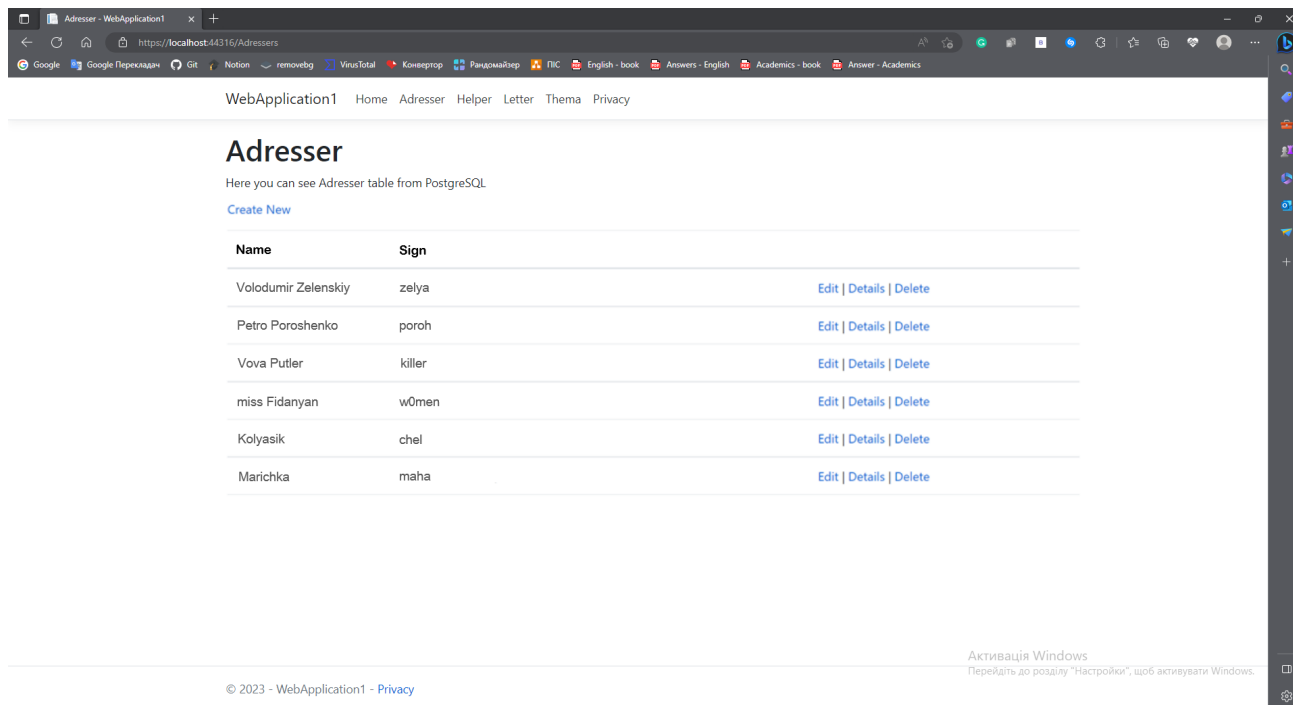


Рисунок 4.12 Вигляд сторінки Adresser серверу

Helper

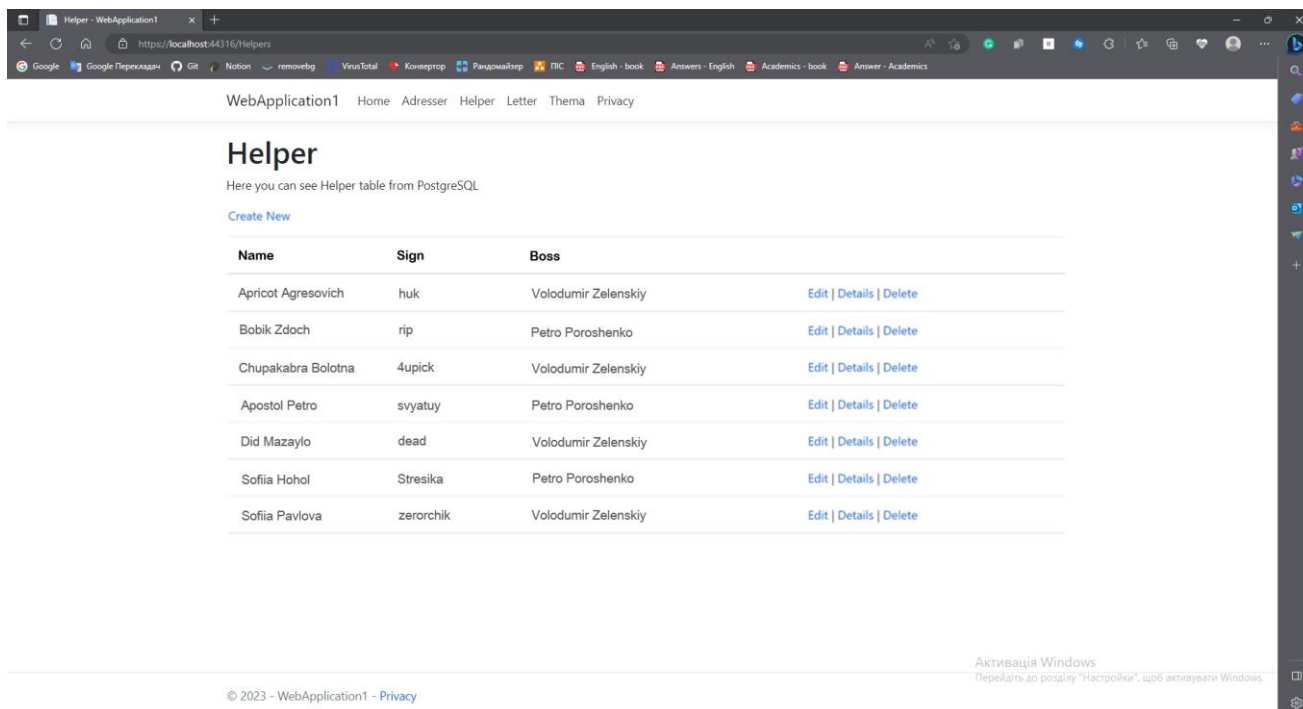


Рисунок 4.13 Вигляд сторінки Helper серверу

Letter

WebApplication1 Home Adresser Helper Letter Thema Privacy

Letter

Here you can see Letter table from PostgreSQL

[Create New](#)

Date	IsAns	DateAns	Themald	SenderId	ReceiverId	HelperId	
13.01.2022 0:00:00	False		6	4	3	6	Edit Details Delete
20.10.2022 0:00:00	True	23.10.2022 0:00:00	1	1	2	1	Edit Details Delete
24.02.2022 0:00:00	True	24.03.2022 0:00:00	2	2	1	2	Edit Details Delete
01.08.2021 0:00:00	True	15.01.2022 0:00:00	3	3	5	4	Edit Details Delete
12.12.2021 0:00:00	True	29.12.2021 0:00:00	5	6	4	3	Edit Details Delete

Активация Windows
Перейдите до розділу "Настройки", щоб активувати Windows.

© 2023 - WebApplication1 - [Privacy](#)

Рисунок 4.14 Вигляд сторінки Letter серверу

Thema

WebApplication1 Home Adresser Helper Letter Thema Privacy

Thema

Here you can see Thema table from PostgreSQL

[Create New](#)

Name	
killling pootin	Edit Details Delete
burning kreml	Edit Details Delete
killling rosnya	Edit Details Delete
bombing krimea bridge	Edit Details Delete
fixing bomb shelter	Edit Details Delete
opening chreshatuk station	Edit Details Delete
changing siren signal	Edit Details Delete

Активация Windows
Перейдите до розділу "Настройки", щоб активувати Windows.

© 2023 - WebApplication1 - [Privacy](#)

Рисунок 4.15 Вигляд сторінки Thema серверу

Privacy

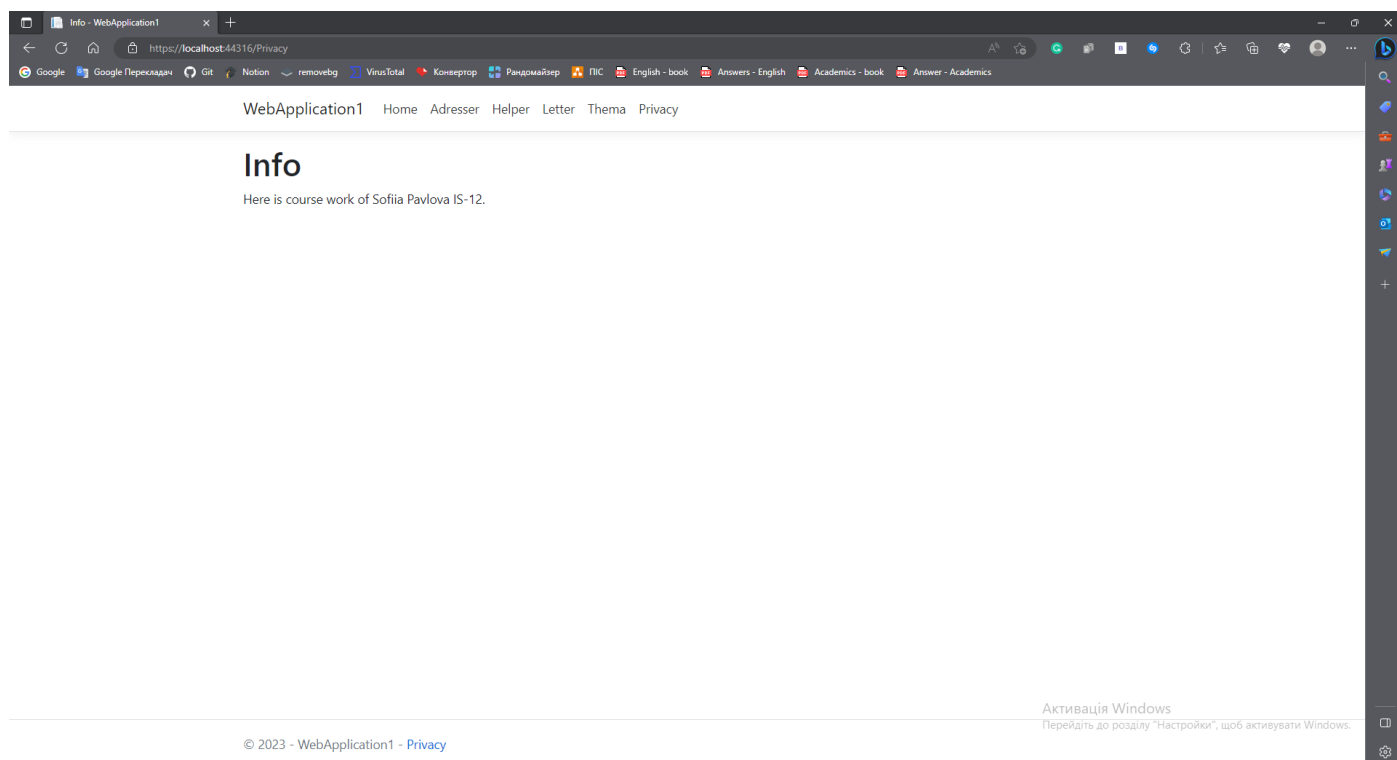


Рисунок 4.16 Вигляд сторінки Privacy серверу

Функціонал застосунку

Create New

Викличемо функцію створення на прикладі таблиці Adresser.

Створимо нового адресата ририри.

The screenshot shows a web form titled 'Create Adresser'. It has two input fields: 'Name' and 'Sign'. Both fields contain the text 'ририри'. Below the input fields is a blue 'Create' button and a blue link 'Back to List'.

Рисунок 4.17 Використання функції Create

Бачимо, що новий адресат успішно створився і відобразився в таблиці.

Результат:

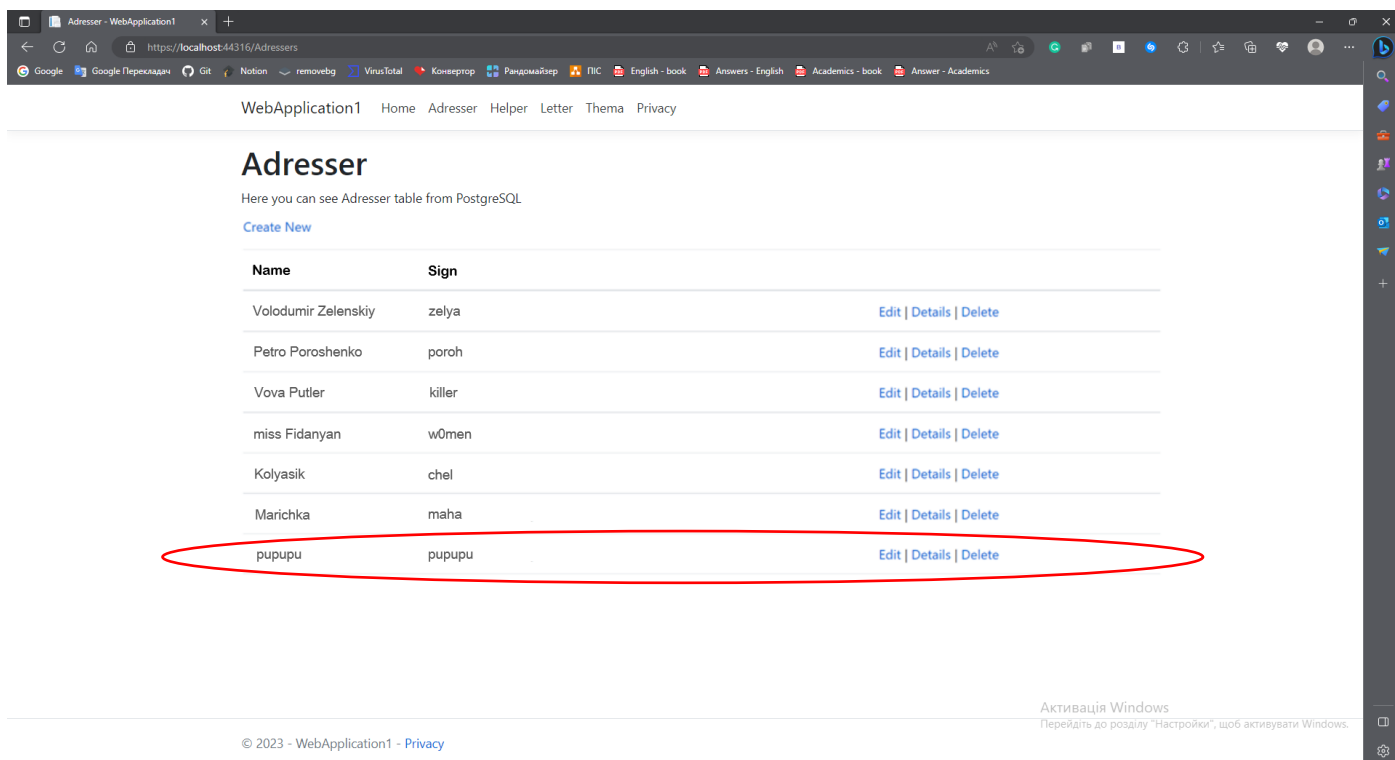


Рисунок 4.18 Успішне створення нового адресата

Details

Викличемо функцію перегляду деталей на прикладі новоствореного адресата.

Результат:

Details

Adresser

Name pupupu
Sign pupupu

[Edit](#) | [Back to List](#)

Рисунок 4.19 Використання функції Details

Застосунок вивів на екран деталі запису про адресата.

Edit

Викличемо функцію редагування на тому ж прикладі.

Змінимо підпис адресата рипури на поooooooooo.

Edit

Adresser

Name

рипури

Sign

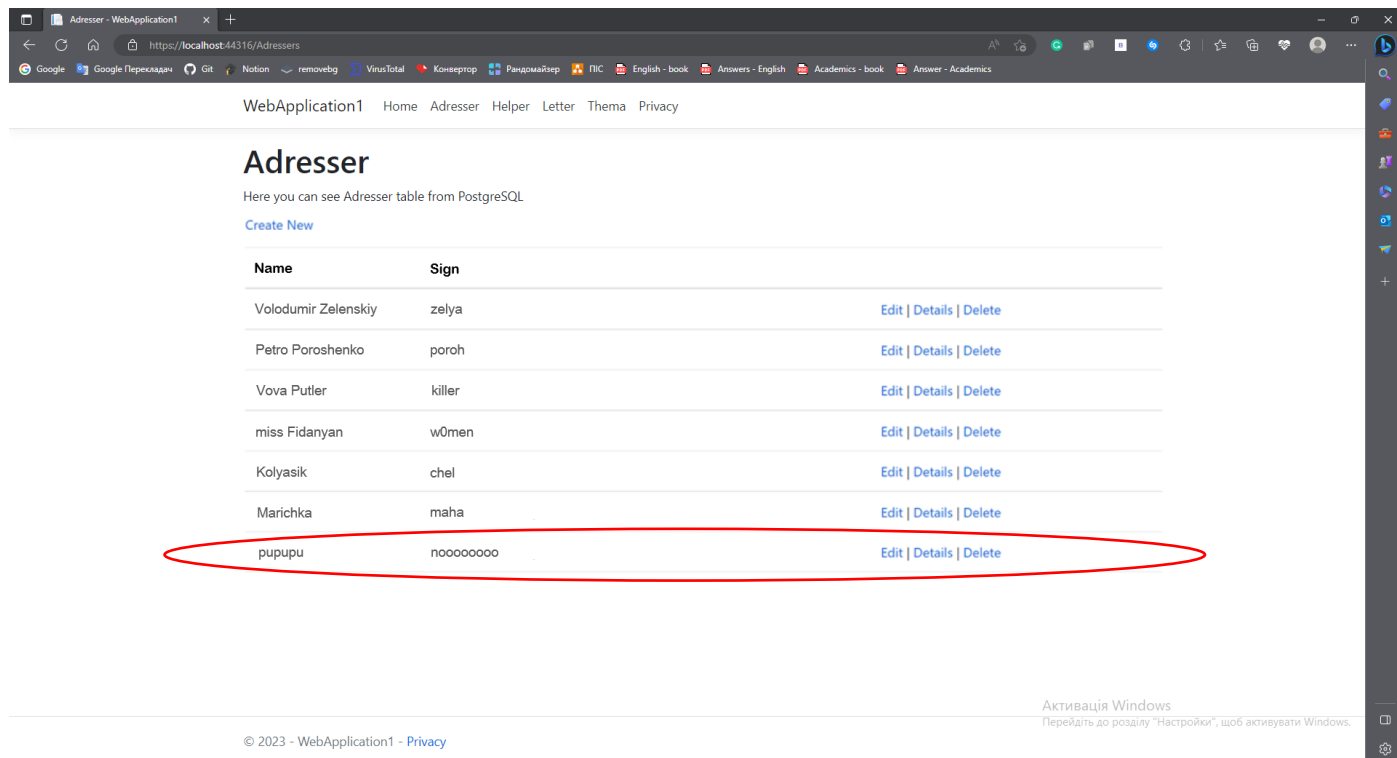
поooooooooo

Save

Рисунок 4.20 Використання функції Edit

Бачимо, що зміни успішно збережені.

Результат:



Name	Sign	
Volodimir Zelenskiy	zelya	Edit Details Delete
Petro Poroshenko	poroh	Edit Details Delete
Vova Putler	killer	Edit Details Delete
miss Fidanyan	w0men	Edit Details Delete
Kolyasik	chel	Edit Details Delete
Marichka	maha	Edit Details Delete
рипури	поooooooooo	Edit Details Delete

Рисунок 4.21 Успішне редагування підпису адресата

Змінимо підпис назад на рурири.

Delete

Викличемо функцію видалення на прикладі того ж самого адресата.

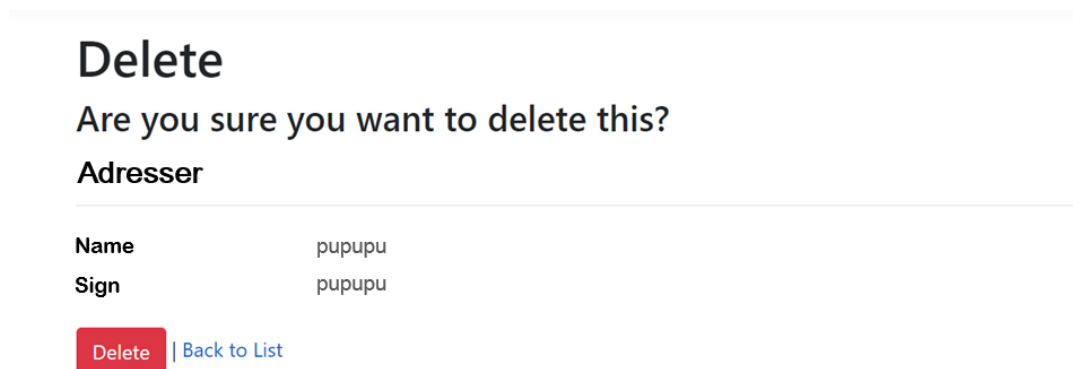


Рисунок 4.22 Використання функції Delete

Бачимо, що запис видалився.

Результат:

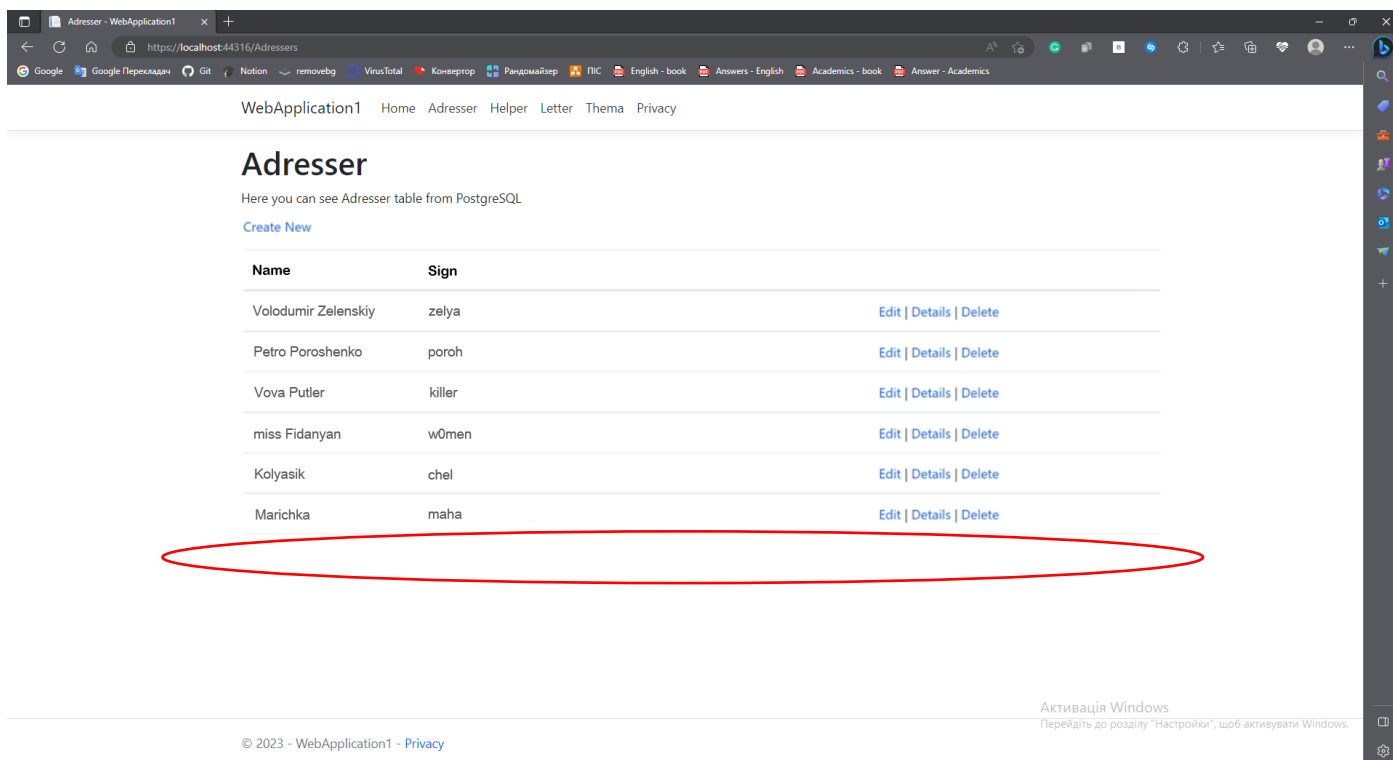


Рисунок 4.23 Успішне видалення адресата

Повний лістинг проекту

Повний лістинг проекту можна знайти за посиланням:

https://github.com/zerorchik/Kursach_DB

5. ЕСКІЗ ЗВІТУ

Для створення Report було використано веб-інтерфейс draw.io.

ПНД Зеленського В. О.

Система "Приймальня народного депутата"

Відомість наявності прийняття звернень за 25.10.2022

Id	Дата прийняття	Вирішення	Дата вирішення	Тема	Відправник	Отримувач	Помічник
1	20.10.2022	розглядається		Вбивство Путіна	Зеленський В. О.	Порошенко П. О.	Павлова С. О.
2	24.02.2022	розглядається		Спалення Кремля	Порошенко П. О.	Зеленський В. О.	Гоголь С. В.
3	13.08.2021	вирішено	08.10.2022	Підбив Кримського моста	Зеленський В. О.	Путін В. В.	Апостол П. С.
4	06.01.2023	вирішено	08.01.2023	Відкриття станції Хрещатик	Подольна М. К.	Зеленський В. О.	Дід В. П.
Групування за вирішенням "розглядається"							
1	20.10.2022	розглядається		Вбивство Путіна	Зеленський В. О.	Порошенко П. О.	Павлова С. О.
2	24.02.2022	розглядається		Спалення Кремля	Порошенко П. О.	Зеленський В. О.	Гоголь С. В.
Сортування за датою прийняття, спочатку новіші							
4	06.01.2023	вирішено	08.01.2023	Відкриття станції Хрещатик	Подольна М. К.	Зеленський В. О.	Дід В. П.
1	20.10.2022	розглядається		Вбивство Путіна	Зеленський В. О.	Порошенко П. О.	Павлова С. О.
2	24.02.2022	розглядається		Спалення Кремля	Порошенко П. О.	Зеленський В. О.	Гоголь С. В.
3	13.08.2021	вирішено	08.10.2022	Підбив Кримського моста	Зеленський В. О.	Путін В. В.	Апостол П. С.

Рисунок 5.1 Відомість наявності звернень у приймальні народного депутата

ВИСНОВКИ

У ході виконання другої частини курсової роботи я проаналізувала предметну область, виокремила сутності та зв'язки між ними та створила ER-модель приймальні народного депутата.

У ході виконання другої частини курсової роботи я навчилась працювати з базою даних Postgre, створювати в ній сервера, таблиці, заповнювати та редагувати їх та створювати відповідні ERD діаграми.

У ході виконання третьої частини курсової роботи я навчилась працювати з базою даних Postgre, створювати запити від двох або більше таблиць, підзапити, аналітичні та групуючі запити, сортувати елементи таблиць, виборки елементів таблиць за певними правилами, використовувати групуючі запити для отримання агрегованої інформації та запити на зміну змісту БД та використовувати тригери та їх функції.

У ході виконання четвертої частини курсової роботи я отримала практичний досвід роботи з Razor Pages у проекті ASP.NET Core, ознайомила з структурою проекту та файлом Startup.cs, який відповідає за конфігурацію додатку, створила сторінки Index та Privacy за допомогою Razor Pages та зрозуміла, як вони взаємодіють зі зв'язаними моделями та базою даних PostgreSQL. Крім того, я створила робочий веб-застосунок на ASP.NET через Visual Studio і підключила до нього раніше створену БД PostgreSQL. Створений застосунок дозволяє переглядати деталі записів у таблицях, створювати, редагувати та видаляти їх.

У ході виконання п'ятої частини курсової роботи я навчилась створювати ескізи звітів предметних областей БД. Створила ескіз звіту для приймальні народного депутата у draw.io.

ПЕРЕЛІК ВИКОРИСУНОКТАНИХ ДЖЕРЕЛ

- [1] Проектирование и реализация баз данных Microsoft SQL Server 2000. Учебный курс MCAD/MCSE, MCDBA/Пер. с англ. — 2-е изд., испр. — М.: Издательско-торговый дом «Русская Редакция», 2003. — 512 стр.: ил.
- [2] Морган С. Проектирование и оптимизация доступа к базам данных Microsoft SQL Server 2005. Учебный курс Microsoft : Пер.с англ. / С.Морган, Т.Тернстрем. — М.: Издательство «Русская редакция», 2008. — 480 стр.: ил.
- [3] Лилишенко О.В. Теорія бухгалтерського обліку: Підручник — Київ: Вид-во „Центр навчальної літератури”, 2008-219 с.
- [4] Гарсія-Моліна Г. Системы баз данных Полный курс/ Г. Гарсія Моліна, Дж.Ульман, Дж. Уидом М.: Изд. дом “Вильямс”, 2003. — 1088 с.
- [5] Ульман Дж. Основы систем баз даних / Дж. Ульман. М.: “Финансы и статистика”, Гарвард, 1983. — 325 с.
- [6] Гаврилов Д.А. Управление производством на базе стандарта MRP II. 2-е изд.-СПб: Питер 2008. — 416 с.: ил.
- [7] Г.А.Гайна. Основы проектування баз даних: Навчальний посібник. - К.; Кондор, 2008. - 200 с.
- [8] Джексон Г. Проектирование реляционных баз данных. — М.: Мир, 1991. - 252 с.
- [9] Дейт К. Введение в систему баз данных. — М.: Мир, 1998. - 846 с.
- [10] Д. Крёнке. Теория и практика построения баз данных. / Учебное пособие. - СПб.: — Питер. 2003. — 800с.: — ил.