**Звіт БД**

**Tests**

**Cтруктура Бази Данних:**

**1. User (Користувач)**

* **Атрибути**:
  + customer\_id: Унікальний ідентифікатор користувача.
  + customer\_name: Ім'я користувача.
  + role: Роль користувача (наприклад, адміністратор, студент тощо).
* **Призначення**: Зберігає основну інформацію про користувачів, які взаємодіють із системою.

**2. Test (Тест)**

* **Атрибути**:
  + test\_id: Унікальний ідентифікатор тесту.
  + title: Назва тесту.
  + status: Статус тесту (наприклад, активний чи завершений).
* **Зв'язки**:
  + Пов'язаний із сутністю User через зв'язок "один до багатьох" (один користувач може створювати декілька тестів).
* **Призначення**: Містить основну інформацію про тести.

**3. Questions (Питання)**

* **Атрибути**:
  + question\_id: Унікальний ідентифікатор питання.
  + text: Текст питання.
* **Зв'язки**:
  + Пов'язаний із сутністю Test через зв'язок "один до багатьох" (кожен тест може мати кілька питань).
* **Призначення**: Зберігає текст питань, що входять до тестів.

**4. TestStatuses (Статуси тестів)**

* **Атрибути**:
  + teststatus\_id: Унікальний ідентифікатор запису статусу тесту.
  + status: Статус (наприклад, "пройдено", "не розпочато").
  + date: Час і дата зміни статусу.
* **Зв'язки**:
  + Пов'язаний із Test та User, щоб відслідковувати статуси тестів для конкретних користувачів.
* **Призначення**: Дозволяє зберігати історію змін статусів тесту.

**5. Attempts (Спроби)**

* **Атрибути**:
  + attempt\_id: Унікальний ідентифікатор спроби.
  + attempt\_num: Номер спроби.
  + start\_timestamp, end\_timestamp: Час початку та завершення спроби.
  + score: Оцінка, отримана за спробу.
* **Зв'язки**:
  + Зв'язок із Test та User.
* **Призначення**: Зберігає інформацію про спроби проходження тестів користувачами.

**6. Mistakes (Помилки)**

* **Атрибути**:
  + mistakes\_id: Унікальний ідентифікатор помилки.
* **Зв'язки**:
  + Пов'язаний із сутностями Attempts і Questions, що дозволяє фіксувати помилки, зроблені під час спроб.
* **Призначення**: Відстежує неправильні відповіді на питання.

**7. Results (Результати)**

* **Атрибути**:
  + result\_id: Унікальний ідентифікатор результату.
  + date\_passed: Час завершення тесту.
* **Зв'язки**:
  + Пов'язаний із сутністю Attempts, щоб відображати результати спроб.
* **Призначення**: Зберігає фінальні результати після завершення тесту.

**8. Answers (Відповіді)**

* **Атрибути**:
  + answer\_id: Унікальний ідентифікатор відповіді.
  + text: Текст відповіді.
  + is\_correct: Флаг, що визначає правильність відповіді.
* **Зв'язки**:
  + Пов'язаний із Questions та Attempts.
* **Призначення**: Зберігає варіанти відповідей на питання.

**Основні Зв'язки**

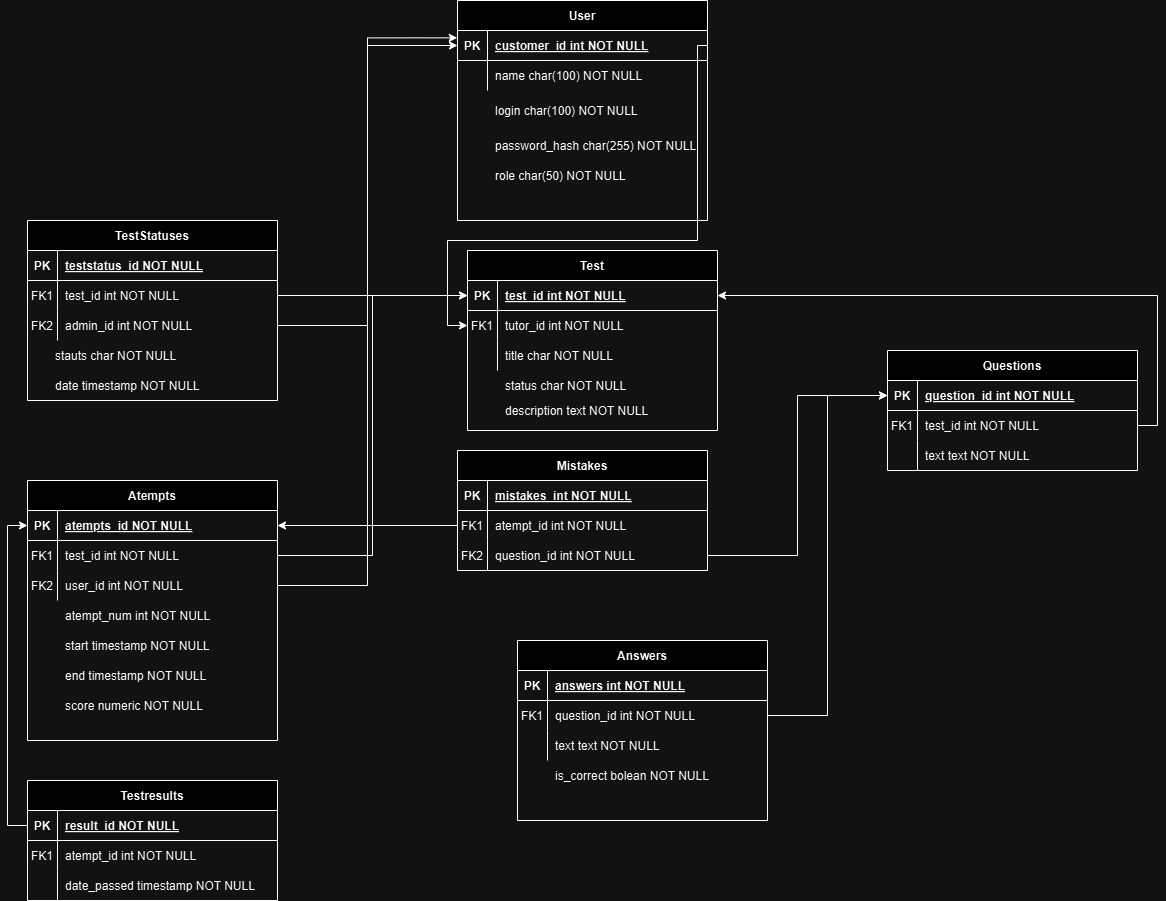
* **Користувач -> Тести**: Користувач створює тести.
* **Тести -> Питання**: Кожен тест складається з кількох питань.
* **Спроби -> Помилки**: Фіксуються помилки у відповідях під час спроби.
* **Результати -> Спроби**: Результати тесту базуються на спробах.

**Використання**

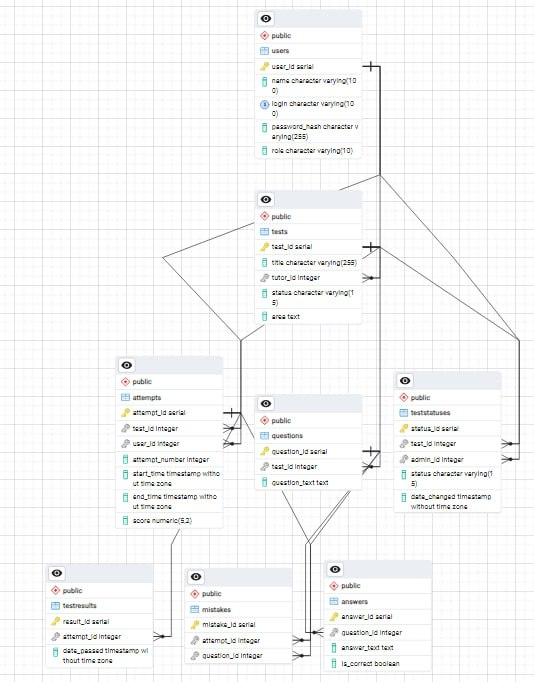
Ця база дозволяє:

1. Створювати, проходити й аналізувати тести.
2. Відстежувати історію спроб і статусів тестів.
3. Фіксувати результати, помилки та правильність відповідей.

**UML Діаграма:**

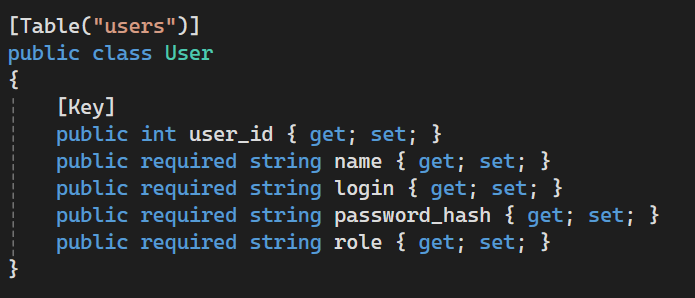
****

**ER діаграма:**

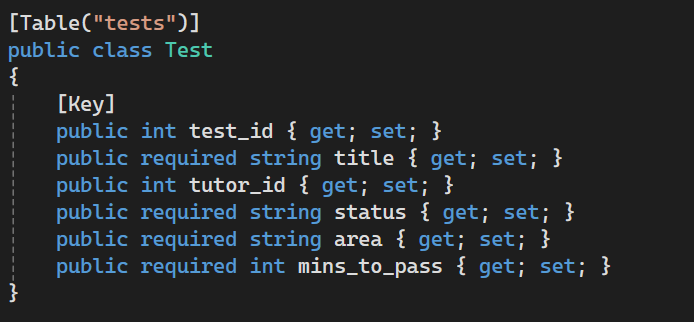


1. **Створення бази данних через ADO.net:**

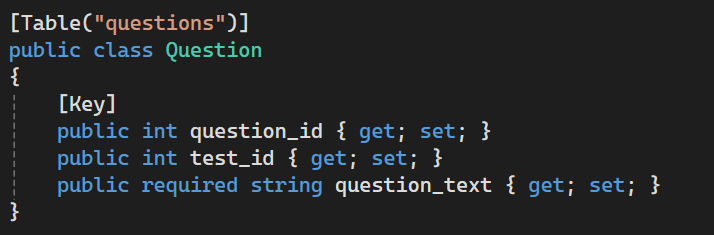
Таблиця users:



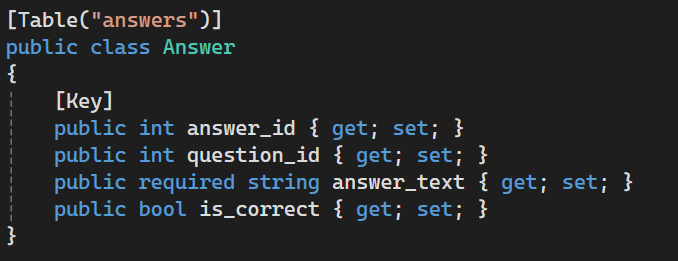
Таблиця tests:



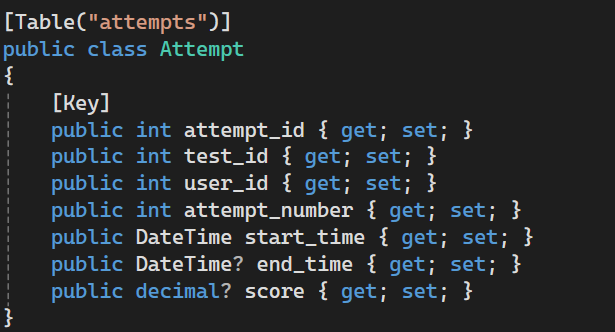
Таблиця questions:



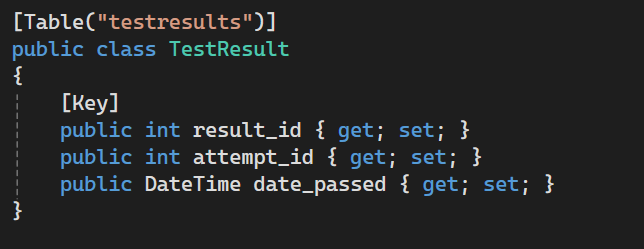
Таблиця answers:



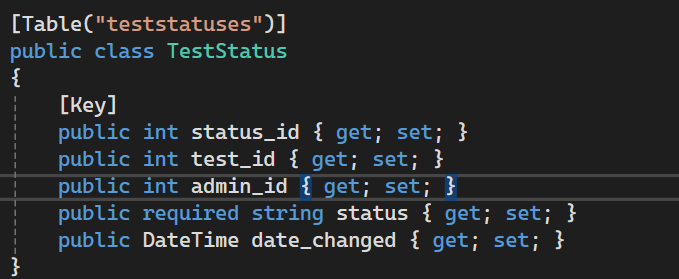
Таблиця attempts:



Таблиця mistakes:



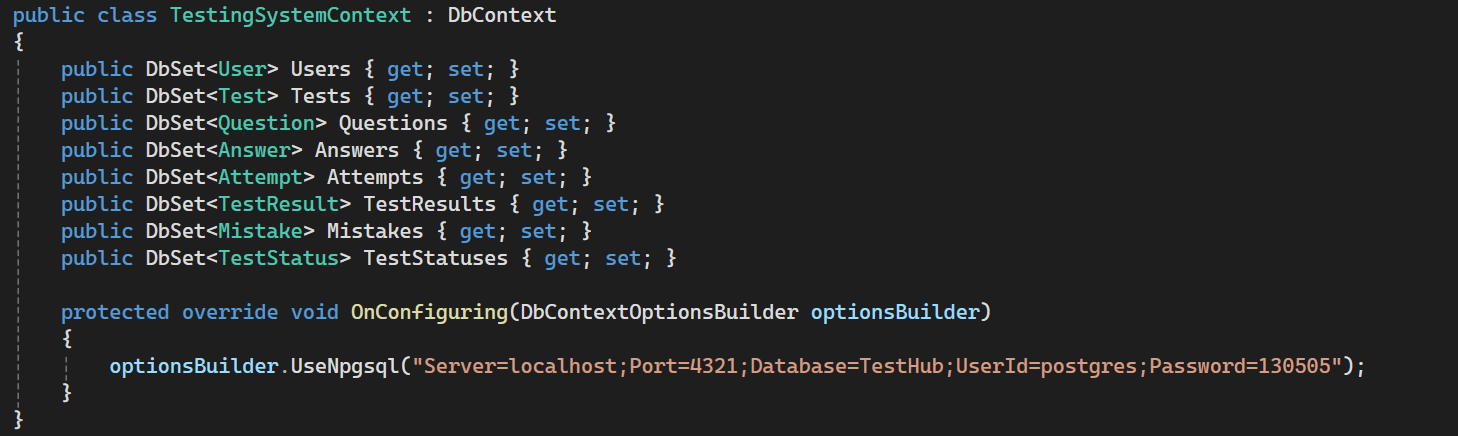
Таблиця teststatuses:



### ****2. Контекст бази даних****

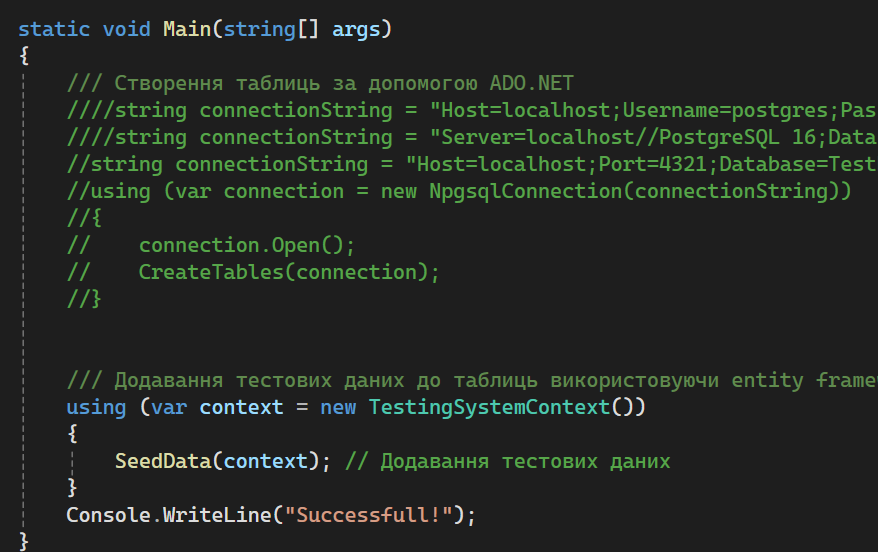
Клас TestingSystemContext реалізує контекст бази даних, що відповідає за:

* Зв’язок моделей із базою даних.
* Налаштування підключення через UseNpgsql() із вказаним рядком з’єднання.



У Main виконується:

1. **Створення таблиць:** За допомогою ADO.NET і SQL-запитів (CreateTables()).
2. **Додавання тестових даних:** Метод SeedData() додає випадкові дані до таблиць, включаючи користувачів, тести, питання, відповіді, спроби, результати та помилки.



Функція створення таблиць CreateTables:

static void CreateTables(NpgsqlConnection connection)

{

using (var command = new NpgsqlCommand())

{

command.Connection = connection;

// 1. Створення таблиці Users

command.CommandText = @"

CREATE TABLE IF NOT EXISTS Users (

user\_id SERIAL PRIMARY KEY,

name VARCHAR(100) NOT NULL,

login VARCHAR(100) UNIQUE NOT NULL,

password\_hash VARCHAR(255) NOT NULL,

role VARCHAR(10) CHECK(role IN ('student', 'tutor', 'admin')) NOT NULL

);";

command.ExecuteNonQuery();

// 2. Створення таблиці Tests

command.CommandText = @"

CREATE TABLE IF NOT EXISTS Tests (

test\_id SERIAL PRIMARY KEY,

title VARCHAR(255) NOT NULL,

tutor\_id INT REFERENCES Users(user\_id),

status VARCHAR(15) CHECK(status IN ('pending', 'on\_review', 'approved', 'rejected')) NOT NULL,

area TEXT NOT NULL,

mins\_to\_pass INT

);";

command.ExecuteNonQuery();

// 3. Створення таблиці Questions

command.CommandText = @"

CREATE TABLE IF NOT EXISTS Questions (

question\_id SERIAL PRIMARY KEY,

test\_id INT REFERENCES Tests(test\_id),

question\_text TEXT NOT NULL

);";

command.ExecuteNonQuery();

// 4. Створення таблиці Answers

command.CommandText = @"

CREATE TABLE IF NOT EXISTS Answers (

answer\_id SERIAL PRIMARY KEY,

question\_id INT REFERENCES Questions(question\_id),

answer\_text TEXT NOT NULL,

is\_correct BOOLEAN NOT NULL

);";

command.ExecuteNonQuery();

// 5. Створення таблиці Attempts

command.CommandText = @"

CREATE TABLE IF NOT EXISTS Attempts (

attempt\_id SERIAL PRIMARY KEY,

test\_id INT REFERENCES Tests(test\_id),

user\_id INT REFERENCES Users(user\_id),

attempt\_number INT NOT NULL,

start\_time TIMESTAMP WITHOUT TIME ZONE,

end\_time TIMESTAMP WITHOUT TIME ZONE,

score DECIMAL(5, 2)

);";

command.ExecuteNonQuery();

// 7. Створення таблиці Mistakes

command.CommandText = @"

CREATE TABLE IF NOT EXISTS Mistakes (

mistake\_id SERIAL PRIMARY KEY,

attempt\_id INT REFERENCES Attempts(attempt\_id),

question\_id INT REFERENCES Questions(question\_id),

user\_id INT REFERENCES Users(user\_id)

);";

command.ExecuteNonQuery();

// 8. Створення таблиці TestStatuses

command.CommandText = @"

CREATE TABLE IF NOT EXISTS TestStatuses (

status\_id SERIAL PRIMARY KEY,

test\_id INT REFERENCES Tests(test\_id),

admin\_id INT REFERENCES Users(user\_id),

status VARCHAR(15) CHECK(status IN ('on\_review', 'approved', 'rejected')) NOT NULL,

date\_changed TIMESTAMP DEFAULT CURRENT\_TIMESTAMP

);";

command.ExecuteNonQuery();

Console.WriteLine("All tables created successfully.");

}

}