## **PostgreSQL CRUD Test with Playwright**

This Playwright test script validates the **Create, Read, Update, and Delete (CRUD)** operations on a PostgreSQL database using a custom db module.

Connection String:

1. Neon connection string for PostgreSQL
   1. postgresql://neondb\_owner:npg\_6GnMoJmZr9LN@ep-gentle-boat-a5tn1mjj-pooler.us-east-2.aws.neon.tech/neondb?sslmode=require

### **File Location**

* **Test file**: Located in a Playwright test suite.
* **Database module**: ../pages/postgre.ts exposes db with CRUD methods.

**/pages/**[**postgre.ts**](http://postgre.ts)**:**

### **Functionality Overview**

* Create User  
   Adds a new student record to the database using a name and email address. Returns the newly created user.
* Get All Users  
   Retrieves all student records currently stored in the database. Useful for validation and listing operations.
* Update User  
   Updates the name of an existing student identified by their unique ID. Returns the updated user details.
* Delete User  
   Removes a student record from the database based on their ID. Returns the deleted record to confirm the operation.
* Close Connection  
   Closes the database connection pool. This is particularly important in automated test environments to ensure all connections are properly released.

### Best Practices

* Always close the connection pool after tests or scripts are done to avoid resource leaks.
* Ensure that the student table schema matches the expected structure (with id, name, and email fields).
* Integrate these methods into service layers or test scripts for modular and maintainable database interactions.

### Use Cases

* Database seeding and cleanup during end-to-end testing.
* Backend service methods for user management.
* Prototyping or quick API scaffolding with real database interaction.

**/tests/**[**postgre.spec.ts**](http://postgre.spec.ts)**:**

### 📌 Purpose

The test is designed to ensure that the db utility functions interact correctly with the PostgreSQL student table. It validates each step in the CRUD cycle using assertions and logs for visibility.

### 🧭 Test Flow Overview

1. Create Operation  
    A new student record is inserted into the database with a sample name and email. The test confirms that a valid ID is returned, indicating successful creation.
2. Read Operation  
    The test retrieves all student records to verify that the newly created record exists. It checks that the list of users is not empty.
3. Update Operation  
    The student’s name is updated using their ID. The test verifies that the updated name reflects the intended change.
4. Delete Operation  
    The student record is deleted using the same ID. The test confirms that the returned result corresponds to the deleted user.
5. Cleanup  
    After all operations, the database connection is gracefully closed to ensure resource cleanup and avoid open handles.