МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ЧЕРНІВЕЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

ІМЕНІ ЮРІЯ ФЕДЬКОВИЧА

Навчально-науковий інститут фізико-технічних та комп’ютерних наук

Кафедра комп’ютерних наук

**ЗВІТ**  
**про виконання лабораторної роботи № 8**

**з дисципліни**

**«Сучасні технології програмування»**

**на тему: «CRUD операції в ASP.NET WEB API»**

Виконав

студент 5-го курсу

544м групи

Вікнянський Руслан

Чернівці, 2025

**Мета:** Реалізація базових CRUD операцій (створення, читання, оновлення, видалення) для об'єктів в базі даних.

Посилання на Github - <https://github.com/RuslanCHNU/STP_Labs/tree/main/Lab8>

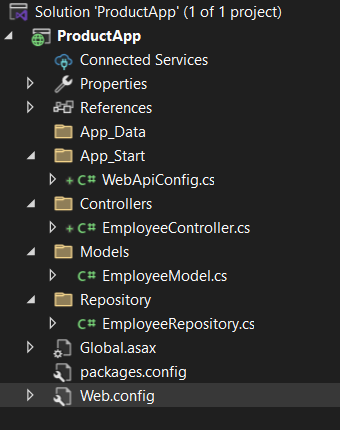
**Порядок виконання роботи**

1. **Налаштування бази даних:**

*CREATE DATABASE Lab8DB;*  
*USE Lab8DB;*  
*CREATE TABLE Employee(...);*  
*-- Збережені процедури: AddNewEmpDetails, GetEmployees, UpdateEmpDetails, DeleteEmpById*

1. **Налагодження connectionString** у Web.config:

*<add name="GetConn" connectionString="Data Source=localhost\SQLEXPRESS,1433;Initial Catalog=Lab8DB;Integrated Security=True" providerName="System.Data.SqlClient"/>*

1. **Структура проекту:**
   1. Models/EmployeeModel.cs
   2. Repository/EmployeeRepository.cs
   3. Controllers/EmployeeController.cs
2. **Реалізація моделі, репозиторію та контролера** з методами Add, Get, Edit, Delete.  
   
3. **Налагодження протоколів TCP/IP**, запуск SQL Server, перевірка брандмауера для порту 1433.
4. **Запуск Web API** та тестування запитів у Postman:
   1. GET: <http://localhost:51986/api/Employee/GetAllEmpDetails>
   2. POST: <http://localhost:51986/api/Employee/AddEmployee>
   3. PUT: <http://localhost:51986/api/Employee/EditEmpDetails/1>
   4. DELETE: <http://localhost:51986/api/Employee/DeleteEmp/1>



**4. Використані команди**

* **SQL (SSMS):**

CREATE DATABASE Lab8DB;  
CREATE TABLE Employee(...);  
EXEC AddNewEmpDetails 'Іван','Львів','вул. Шевченка,1';

**5. Посилання на створений Web API**

* Доступ до API: [http://localhost:51986](http://localhost:51986/)

**6. Код EmployeeRepository.cs**

*using System;*

*using System.Collections.Generic;*

*using System.Linq;*

*using System.Web;*

*using System.Configuration;*

*using System.Data;*

*using System.Data.SqlClient;*

*using ProductApp.Models;*

*namespace ProductApp.Repository*

*{*

*public class EmployeeRepository*

*{*

*private SqlConnection con;*

*private void Connection()*

*{*

*var constr = ConfigurationManager.ConnectionStrings["GetConn"].ToString();*

*con = new SqlConnection(constr);*

*}*

*public bool AddEmployee(EmployeeModel obj)*

*{*

*Connection();*

*SqlCommand com = new SqlCommand("AddNewEmpDetails", con)*

*{*

*CommandType = CommandType.StoredProcedure*

*};*

*com.Parameters.AddWithValue("@Name", obj.Name);*

*com.Parameters.AddWithValue("@City", obj.City);*

*com.Parameters.AddWithValue("@Address", obj.Address);*

*con.Open();*

*int i = com.ExecuteNonQuery();*

*con.Close();*

*return i > 0;*

*}*

*public List<EmployeeModel> GetAllEmployees()*

*{*

*Connection();*

*List<EmployeeModel> empList = new List<EmployeeModel>();*

*SqlCommand com = new SqlCommand("GetEmployees", con)*

*{*

*CommandType = CommandType.StoredProcedure*

*};*

*SqlDataAdapter da = new SqlDataAdapter(com);*

*DataTable dt = new DataTable();*

*con.Open();*

*da.Fill(dt);*

*con.Close();*

*foreach (DataRow dr in dt.Rows)*

*{*

*empList.Add(new EmployeeModel*

*{*

*Empid = Convert.ToInt32(dr["Id"]),*

*Name = Convert.ToString(dr["Name"]),*

*City = Convert.ToString(dr["City"]),*

*Address = Convert.ToString(dr["Address"])*

*});*

*}*

*return empList;*

*}*

*public bool UpdateEmployee(EmployeeModel obj)*

*{*

*Connection();*

*SqlCommand com = new SqlCommand("UpdateEmpDetails", con)*

*{*

*CommandType = CommandType.StoredProcedure*

*};*

*com.Parameters.AddWithValue("@EmpId", obj.Empid);*

*com.Parameters.AddWithValue("@Name", obj.Name);*

*com.Parameters.AddWithValue("@City", obj.City);*

*com.Parameters.AddWithValue("@Address", obj.Address);*

*con.Open();*

*int i = com.ExecuteNonQuery();*

*con.Close();*

*return i > 0;*

*}*

*public bool DeleteEmployee(int id)*

*{*

*Connection();*

*SqlCommand com = new SqlCommand("DeleteEmpById", con)*

*{*

*CommandType = CommandType.StoredProcedure*

*};*

*com.Parameters.AddWithValue("@EmpId", id);*

*con.Open();*

*int i = com.ExecuteNonQuery();*

*con.Close();*

*return i > 0;*

*}*

*}*

*}*

*ADO.NET.*

1. **Код EmployeeController.cs**

using System;

using System.Web.Http;

using ProductApp.Models;

using ProductApp.Repository;

namespace ProductApp.Controllers

{

[RoutePrefix("api/Employee")]

public class EmployeeController : ApiController

{

private readonly EmployeeRepository \_empRepo = new EmployeeRepository();

// GET: api/Employee/GetAllEmpDetails

[HttpGet]

[Route("GetAllEmpDetails")]

public IHttpActionResult GetAllEmpDetails()

{

try

{

var employees = \_empRepo.GetAllEmployees();

return Ok(employees);

}

catch (Exception ex)

{

return InternalServerError(ex);

}

}

// POST: api/Employee/AddEmployee

[HttpPost]

[Route("AddEmployee")]

public IHttpActionResult AddEmployee([FromBody] EmployeeModel emp)

{

if (!ModelState.IsValid)

{

return BadRequest(ModelState);

}

bool added = \_empRepo.AddEmployee(emp);

if (added)

{

return Ok("Employee added successfully");

}

else

{

return InternalServerError(new Exception("Failed to add employee"));

}

}

// PUT: api/Employee/EditEmpDetails/5

[HttpPut]

[Route("EditEmpDetails/{id}")]

public IHttpActionResult EditEmpDetails(int id, [FromBody] EmployeeModel emp)

{

if (!ModelState.IsValid)

{

return BadRequest(ModelState);

}

emp.Empid = id;

bool updated = \_empRepo.UpdateEmployee(emp);

if (updated)

{

return Ok("Employee updated successfully");

}

else

{

return NotFound();

}

}

// DELETE: api/Employee/DeleteEmp/5

[HttpDelete]

[Route("DeleteEmp/{id}")]

public IHttpActionResult DeleteEmp(int id)

{

bool deleted = \_empRepo.DeleteEmployee(id);

if (deleted)

{

return Ok("Employee deleted successfully");

}

else

{

return NotFound();

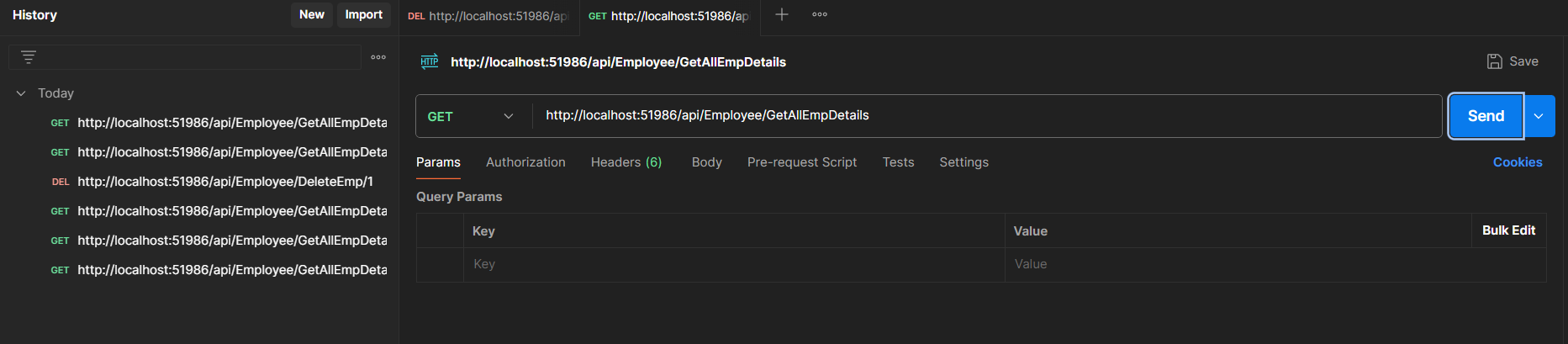
}

}

}

}

1. **Тестування GET запиту в Postman**



**7. Контрольні питання та відповіді**

**1. Що таке ADO.NET і для чого він використовується в розробці програмного забезпечення?**

ADO.NET — це набір бібліотек .NET Framework для роботи з даними та базами даних. Він надає класи для встановлення з'єднання, виконання SQL-запитів і процедур, а також роботи з результатами у вигляді DataSet, DataReader тощо.

**2. Які є основні об'єкти в ADO.NET і як вони пов'язані між собою?**

* **SqlConnection:** керує з'єднанням з базою даних.
* **SqlCommand:** виконує SQL-запит або збережену процедуру.
* **SqlDataReader:** зчитує результати запиту у режимі тільки для читання та вперед.
* **SqlDataAdapter:** дозволяє наповнювати DataSet/DataTable та синхронізувати зміни з базою.
* **DataSet/DataTable:** in-memory представлення даних для маніпуляцій і зв'язків.

Взаємодія: Command використовує Connection для виконання, DataReader/DataAdapter отримує дані.

**3. Як підготувати та виконати SQL-запит до бази даних за допомогою ADO.NET?**

1. Створити і відкрити SqlConnection з рядком підключення.
2. Створити SqlCommand, вказати текст запиту або ім'я процедури (CommandType.StoredProcedure).
3. Додати необхідні параметри через SqlParameter.
4. Викликати ExecuteReader() для отримання SqlDataReader або ExecuteNonQuery() для змін.
5. Обробити результати та закрити з'єднання.