Tipo : Guía de laboratorio
Capítulo : Creando aplicaciones ASP.NET MVC
Duración : 180 minutos

I. OBJETIVO

Implementar el patrón repositorio y unidad de trabajo (Repository Pattern y Unit of Work Pattern).

II. REQUISITOS

Los siguientes elementos de software son necesarios para la realización del laboratorio:

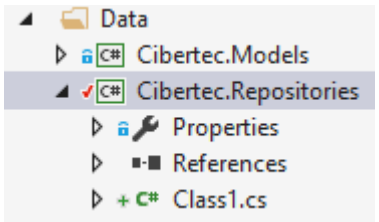
- Windows 10 (como mínimo Windows 8)
- Visual Studio 2017 (como mínimo Visual Studio 2015)

III. EJECUCIÓN DEL LABORATORIO

- Ejercicio Nº 4.2: Implementa Patrón Repositorio y Unit Of Work con Dapper.

Crear un proyecto simple en MVC

1. En la carpeta de solución Data, crear un nuevo proyecto del tipo librería de clases con el nombre "**Cibertec.Repositories**"

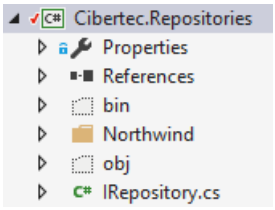


2. Eliminar la clase "**Class1.cs**".

3. Agregar la interfaz "**IRepository**" con el siguiente código:

```
using System.Collections.Generic;

namespace Cibertec.Repositories
{
    public interface IRepository<T> where T : class
    {
        bool Delete(T entity);
        bool Update(T entity);
        int Insert(T entity);
        IEnumerable<T> GetList();
        T GetById(int id);
    }
}
```

4. Crea la carpeta “**Northwind**” al proyecto de repositorios.



5. En la carpeta “**Northwind**” agrega la interfaz “ICustomerRepository” con el siguiente código

```
using Cibertec.Models;

namespace Cibertec.Repositories.Northwind
{
    public interface ICustomerRepository: IRepository<Customer>
    {
    }
}
```

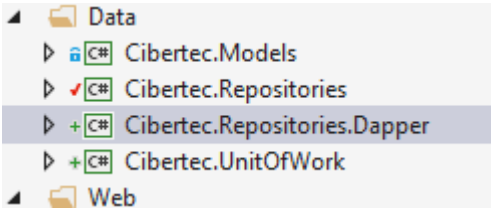
6. Crea el proyecto “**Cibertec.UnitOfWork**" de tipo librería de clases.

7. Al proyecto creado agrega la referencia a “**Cibertec.Repositories**”.

8. Ahora agrega la interfaz “**IUnitOfWork**” con el siguiente código:

```
using Cibertec.Repositories.Northwind;

namespace Cibertec.UnitOfWork
{
    public interface IUnitOfWork
    {
        ICustomerRepository Customers { get; }
    }
}
```

9. Procedemos a crear un nuevo proyecto para poder implementar las interfaces creadas, el nombre de esta librearía de clases es: “**Cibertec.Repositories.Dapper**”:



10. Agrega las referencias a: **Cibertec.Models, Cibertec.Repositories y Cibertec.UnitOfWork**.

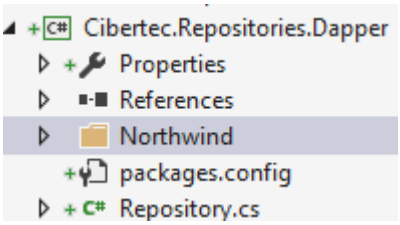
11. Instala los siguientes paquetes:
    - Install-Package Dapper
    - Install-Package Dapper.Contrib

12. Crea la clase “**Repository**” con el siguiente código:

```
using Dapper.Contrib.Extensions;
using System;
using System.Collections.Generic;
using System.Data.SqlClient;
using System.Text;
```

```
namespace Cibertec.Repositories.Dapper
{
    public class Repository<T> : IRepository<T> where T : class
    {
        protected string _connectionString;
        public Repository(string connectionString)
        {
            SqlMapperExtensions.TableNameMapper = (type) => { return
$"{ type.Name }"; };
            _connectionString = connectionString;
        }

        public bool Delete(T entity)
        {
            using (var connection = new
SqlConnection(_connectionString))
            {
                return connection.Delete(entity);
            }
        }

        public T GetById(int id)
        {
            using (var connection = new
SqlConnection(_connectionString))
            {
                return connection.Get<T>(id);
            }
        }

        public IEnumerable<T> GetList()
        {
            using (var connection = new
SqlConnection(_connectionString))
            {
                return connection.GetAll<T>();
            }
        }

        public int Insert(T entity)
        {
            using (var connection = new
SqlConnection(_connectionString))
            {
                return (int)connection.Insert(entity);
            }
        }

        public bool Update(T entity)
        {
            using (var connection = new
SqlConnection(_connectionString))
            {
                return connection.Update(entity);
            }
        }
    }
}
```

13. Agrega la carpeta “**Northwind**”



14. En la carpeta Northwind agrega la clase “**CustomerRepository**” con el siguiente código:

```
using Cibertec.Models;
using Cibertec.Repositories.Northwind;

namespace Cibertec.Repositories.Dapper.Northwind
{
    public class CustomerRepository : Repository<Customer>,
ICustomerRepository
    {
        public CustomerRepository(string connectionString) :
base(connectionString)
        {
        }
    }
}
```

15. En la misma carpeta creamos la clase “**NorthwindUnitOfWork**” con el siguiente código:

```
using Cibertec.UnitOfWork;
using Cibertec.Repositories.Northwind;

namespace Cibertec.Repositories.Dapper.Northwind
{
    public class NorthwindUnitOfWork : IUnitOfWork
    {
        public NorthwindUnitOfWork(string connectionString)
        {
            Customers = new CustomerRepository(connectionString);
        }

        public ICustomerRepository Customers { get; private set; }

    }
}
```

16. Compilamos toda la solución para validar que no tengamos ningún error.

IV. EVALUACIÓN

1. ¿Cuáles son las carpetas principales en una aplicación MVC?

   Las carpetas principales son: Controllers, Views, Models y wwwroot

2. ¿Qué pasa si al nombre del controlador no se le coloca el prefijo Controller?

   Por convención todos los archivos que van a ser controladores deben tener el prefijo Controller al final del nombre para que puedan ser reconocidos por Asp.NET como tales.

3. ¿Cuál es el tipo de datos de los métodos del controller?

   El tipo base de todos los controladores es IActionResult.

4. ¿Qué es una vista Layout?

   Es la vista maestra, en ella se establece la estructura base (html) de todas las vistas que tendrá la aplicación web. Algunas vistas pueden no utilizar la vista base.

5. Los modelos, ¿pueden ser librería de clases?

   Sí, los modelos son un concepto que involucra entidades, acceso a datos, reglas de negocio y pueden ser implementadas en librería de clases (dll).

6. ¿Qué son las vistas parciales?

   Al igual que las vistas normales, estas permiten ser reutilizadas en ciertas partes de la aplicación.