

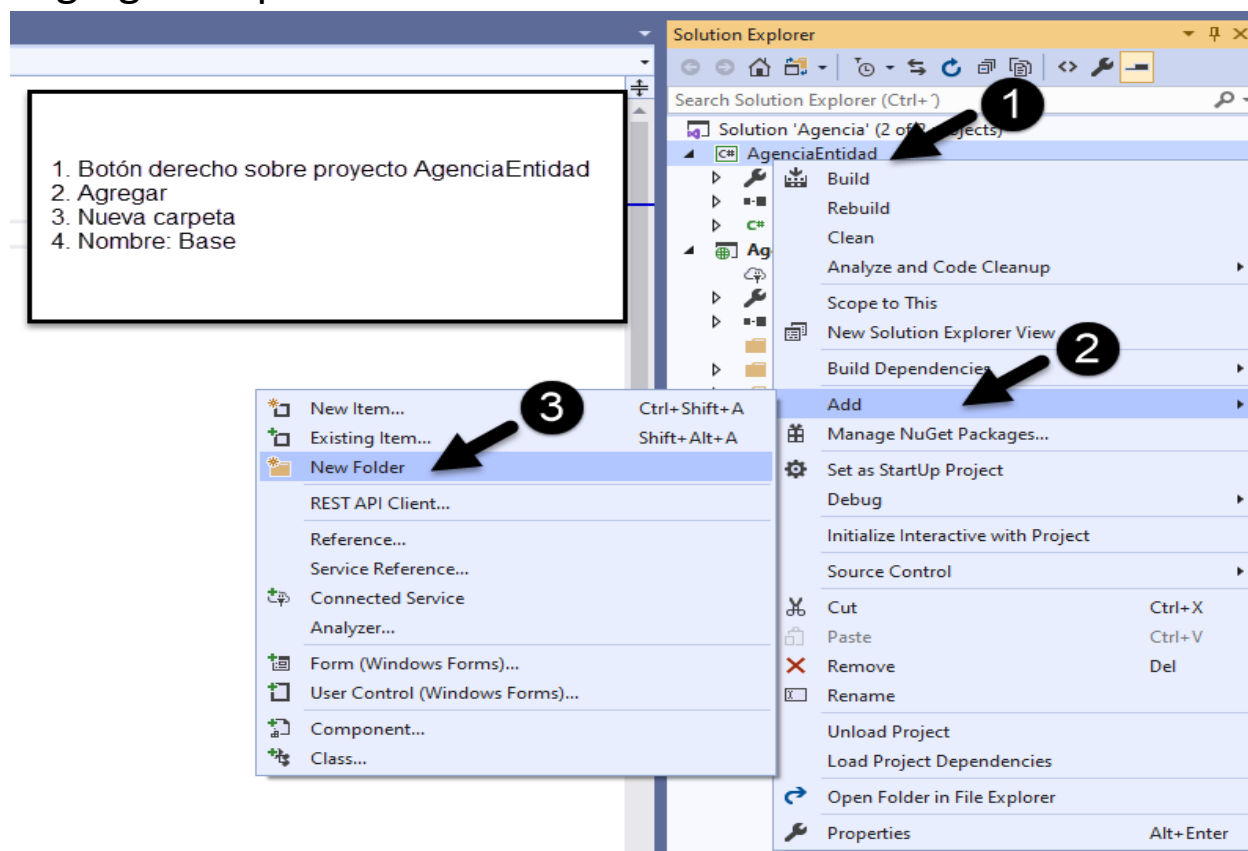
# Asp .Net MVC

Creación de Proyecto

Parte 2

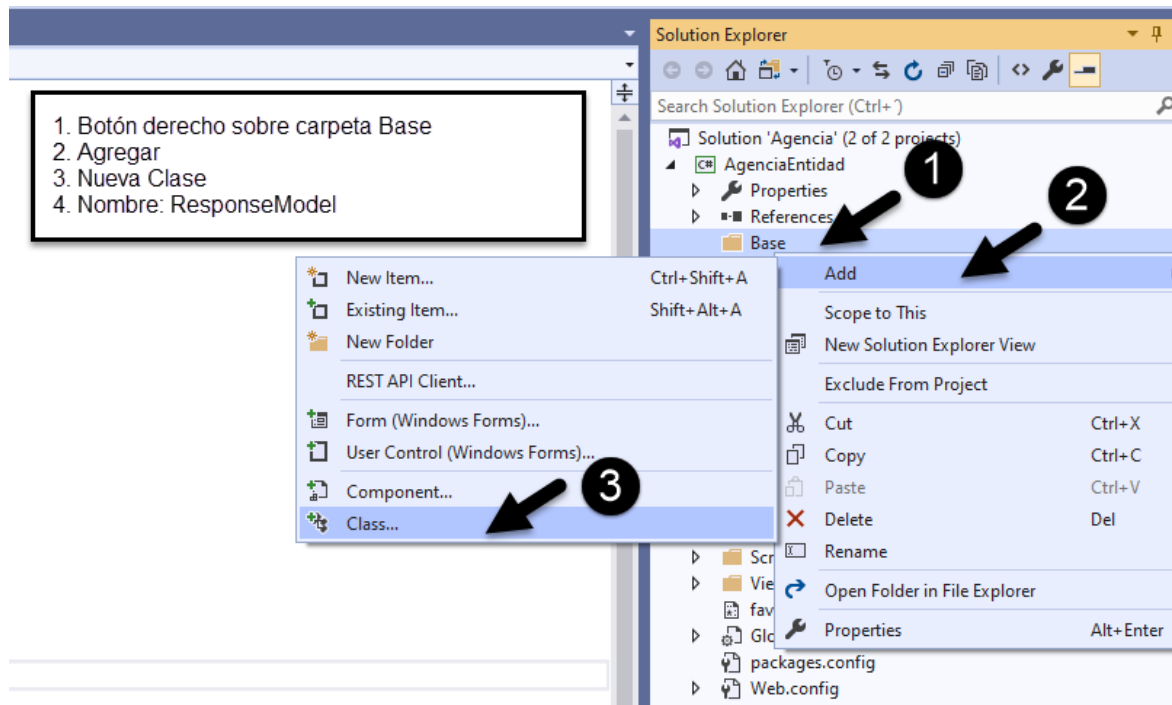
# Proyecto

- Agregar Carpeta Base



# Proyecto

- Agregar Clase ResponseModel a Carpeta Base



# Proyecto

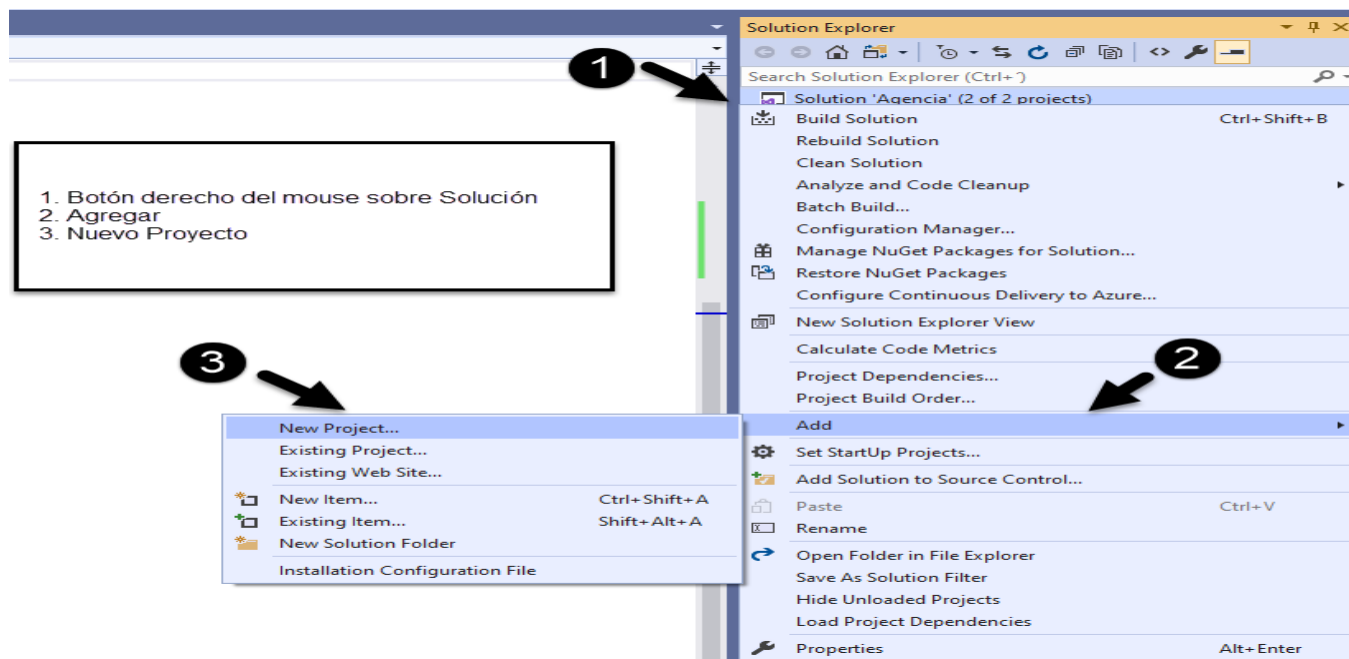
- Clase ResponseModel

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace AgenciaEntidad.Base
{
    public class ResponseModel
    {
        public dynamic Data { get; set; }
        public bool Success { get; set; }
        public string Message { get; set; }
    }
}
```

# Proyecto

- Creación de Proyecto AgenciaDatos
- A continuación crearemos el proyecto donde manejaremos las peticiones que se envían a través de la capa de lógica o negocio.



# Proyecto

- Creación de Proyecto AgenciaDatos

Configure your new project

Class Library (.NET Framework) C# Windows Library

Project name  
AgenciaDatos

Location  
C:\Users\user\Documents\Visual Studio 2019\Projects\Agencia

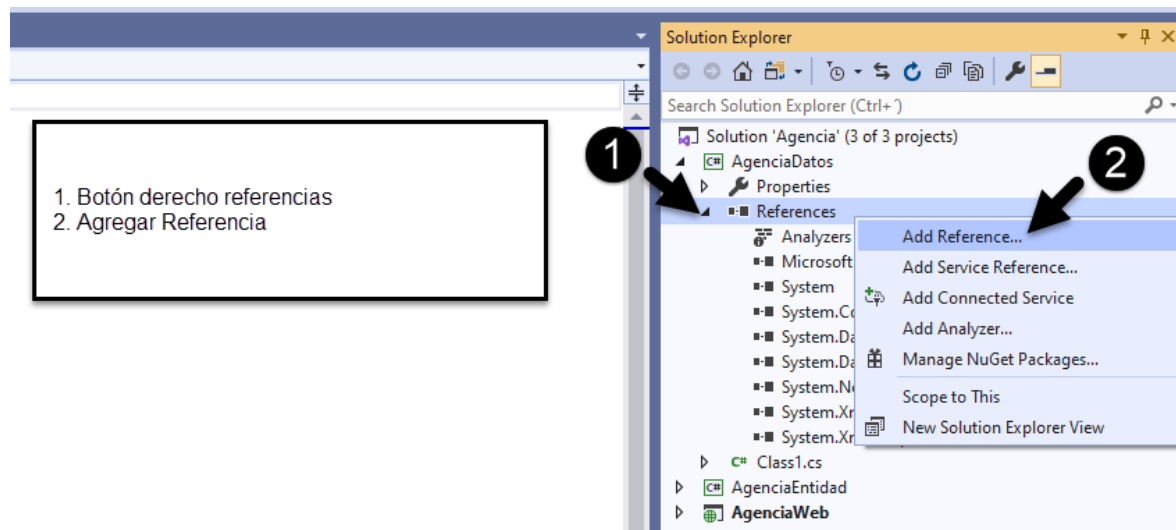
Framework  
.NET Framework 4.7.2

1. Indicar nombre de proyecto  
2. Seleccionar ubicacion  
3. Indicar versión de .NET Framework  
4. Crear

Back Create

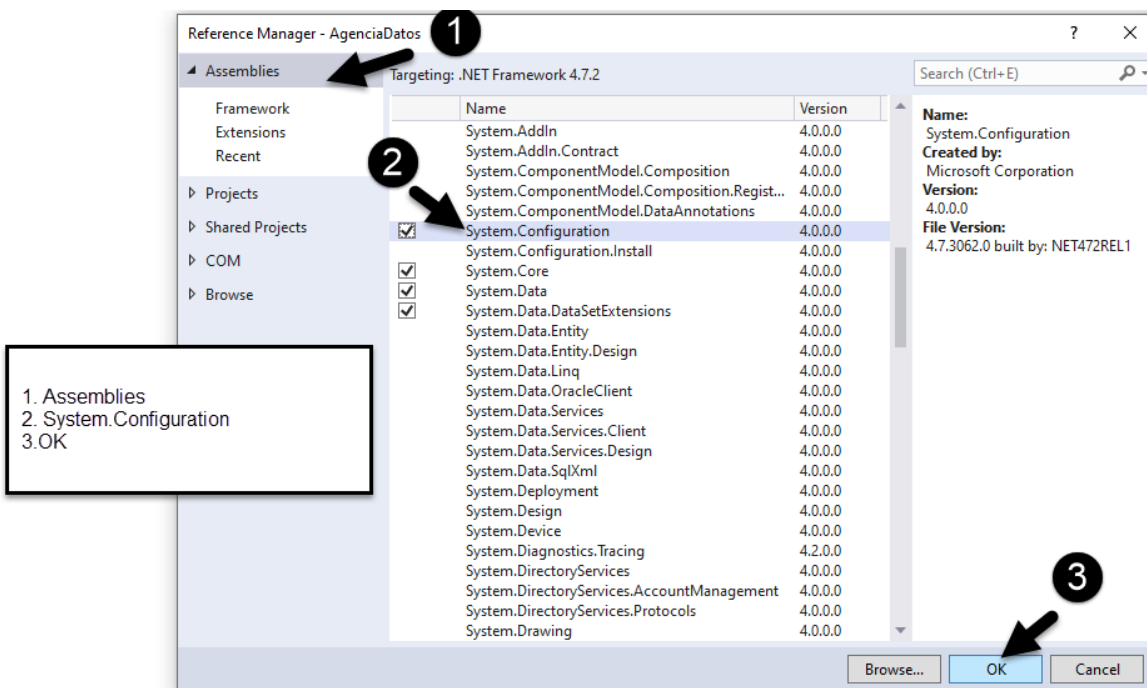
# Proyecto

- Creación de Proyecto AgenciaDatos



# Proyecto

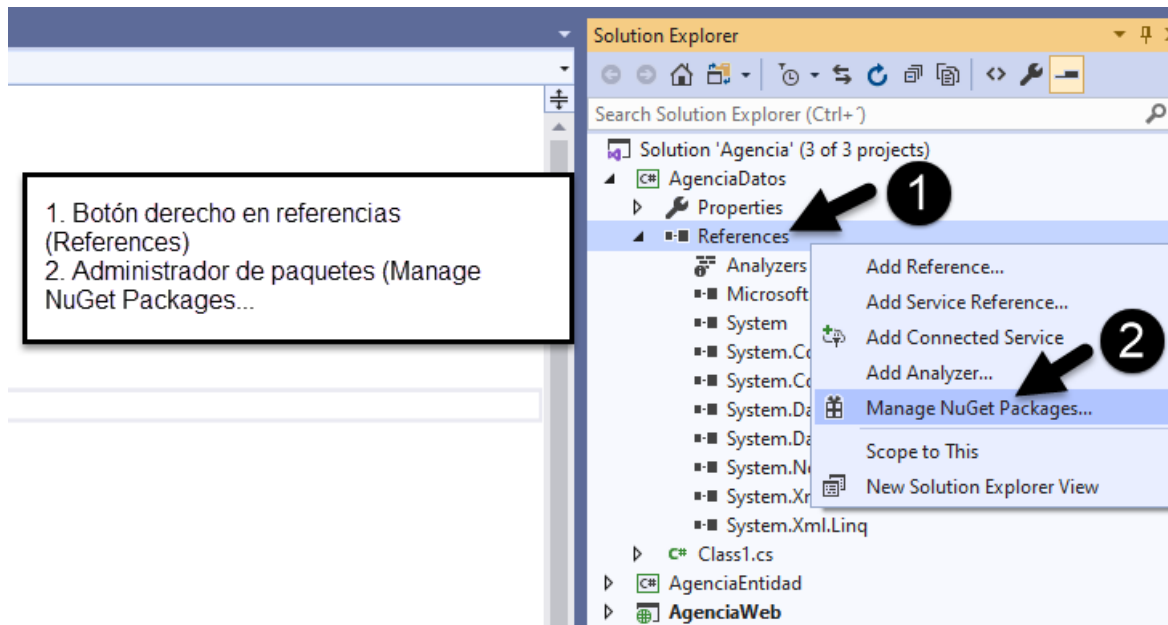
- Creación de Proyecto AgenciaDatos





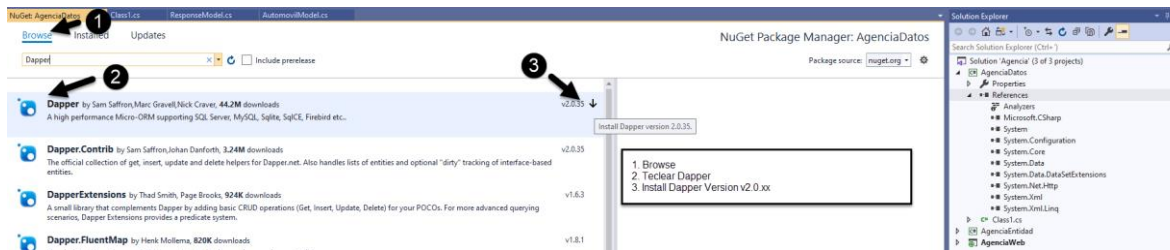
# Proyecto

- Creación de Proyecto AgenciaDatos- Agregar Package



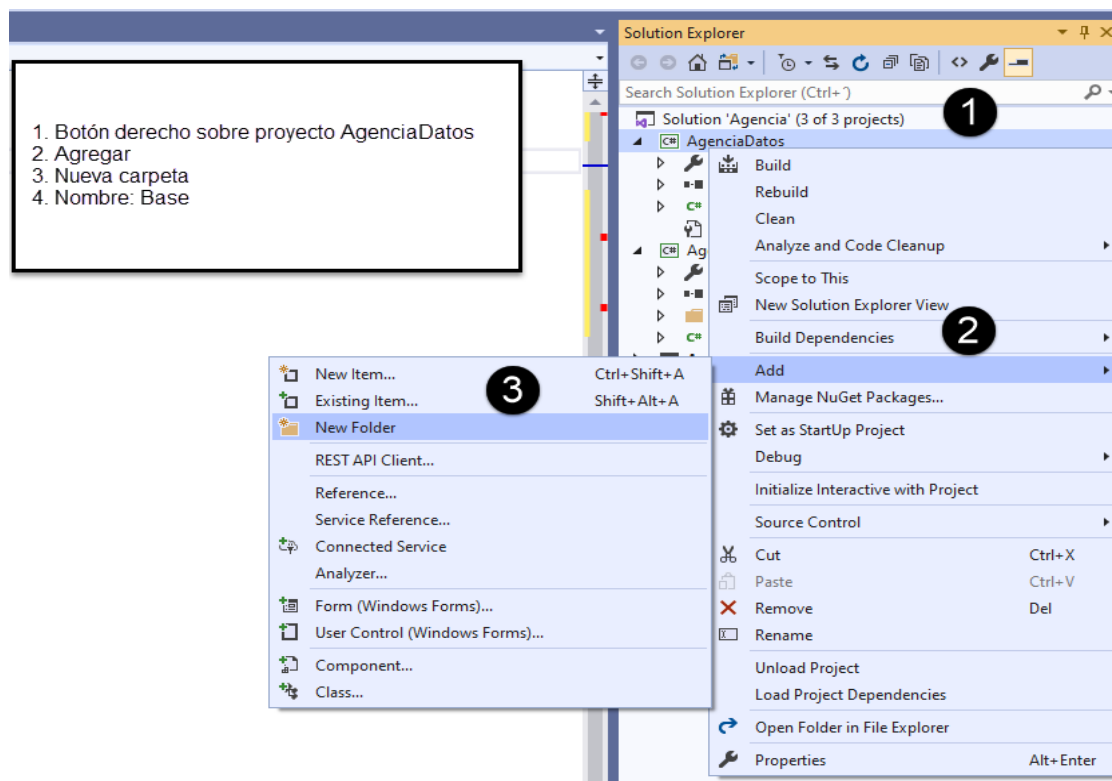
# Proyecto

- Creación de Proyecto AgenciaDatos- Agregar Instalación de Dapper



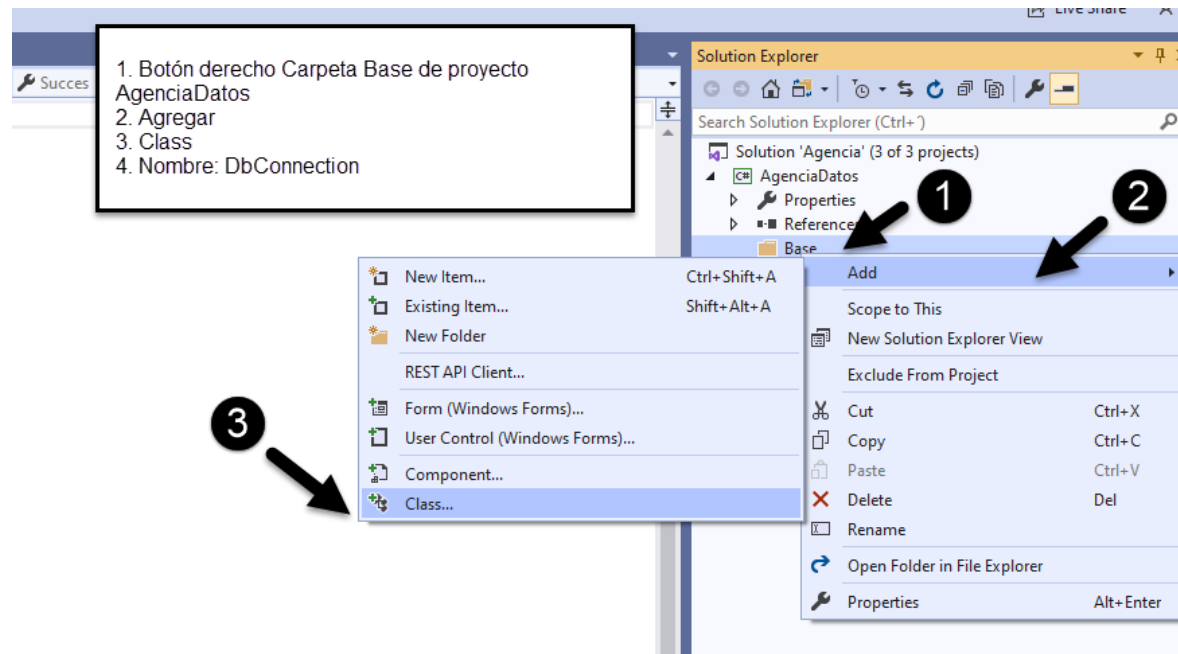
# Proyecto

- Creación de Proyecto AgenciaDatos- Agregar Carpeta Base



# Proyecto

- Creación de Proyecto AgenciaDatos- Agregar DbConnection



# Proyecto


## List<T> Class

Namespace: [System.Collections.Generic](#)

Assembly: [System.Collections.dll](#)

Represents a strongly typed list of objects that can be accessed by index. Provides methods to search, sort, and manipulate lists.

C#

 Copy

```
public class List<T> : System.Collections.Generic.ICollection<T>,  
System.Collections.Generic.IEnumerable<T>, System.Collections.Generic.IList<T>,  
System.Collections.Generic.IReadOnlyCollection<T>,  
System.Collections.Generic.IReadOnlyList<T>, System.Collections.IList
```

# Proyecto


## IDbConnection Interface

Namespace: [System.Data](#)

Assembly: [System.Data.Common.dll](#)

Represents an open connection to a data source, and is implemented by .NET Framework data providers that access relational databases.

C#

 Copy

```
public interface IDbConnection : IDisposable
```

Derived [System.Data.Common.DbConnection](#)  
[System.Data.OleDb.OleDbConnection](#)

Implements [IDisposable](#)

# Proyecto

- Creación de Proyecto AgenciaDatos- Clase DbConnection

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using Dapper;
using System.Data;
using System.Data.SqlClient;
using System.Configuration;

namespace AgenciaDatos.Base
{
    public class DbConnection
    {
        private string conexion = "";
        public DbConnection(string _conexion)
        {
            conexion = ConfigurationManager.ConnectionStrings[_conexion].ToString();
        }
    }
}
```

# Proyecto

- Creación de Proyecto AgenciaDatos- Clase DbConnection

```
public List<T> Query<T>(Dictionary<string, dynamic> Parametros, string SP)
{
    List<T> res;
    try
    {
        using (IDbConnection conn = new SqlConnection(conexion))
        {
            var parametros = new DynamicParameters();
            foreach (KeyValuePair<string, dynamic> item in Parametros)
            {
                parametros.Add(item.Key, item.Value);
            }
            res = conn.Query<T>(SP, param: parametros, commandType: CommandType.StoredProcedure).ToList();
            conn.Close();
        }
    }
    catch (Exception ex)
    {
        throw new Exception("No se pudo completar el proceso", ex);
    }
    return res;
}
```



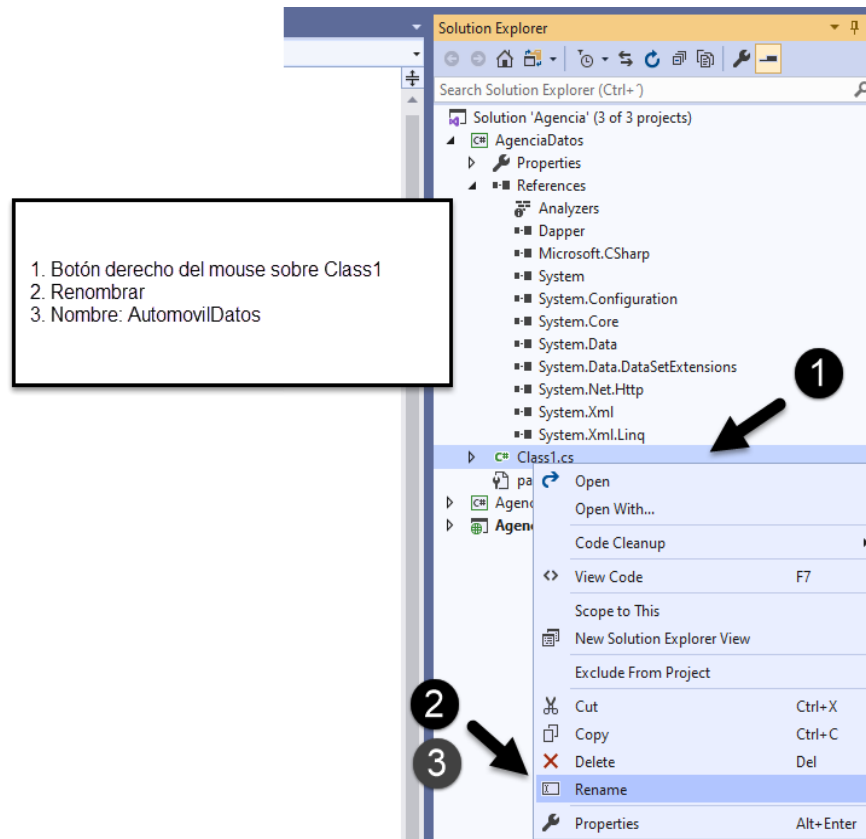
# Proyecto

- Creación de Proyecto AgenciaDatos- Clase DbConnection

```
public T QuerySingle<T>(Dictionary<string, dynamic> Parametros, string SP)
{
    T res;
    try
    {
        using (IDbConnection conn = new SqlConnection(conexion))
        {
            var parametros = new DynamicParameters();
            foreach (KeyValuePair<string, dynamic> item in Parametros)
            {
                parametros.Add(item.Key, item.Value);
            }
            res = conn.QueryFirst<T>(SP, param: parametros, commandType: CommandType.StoredProcedure);
            conn.Close();
        }
    }
    catch (Exception ex)
    {
        throw new Exception("No se pudo completar el proceso", ex);
    }
    return res;
}
}
```

# Proyecto

- Creación de Proyecto AgenciaDatos- Clase AutomovilDatos



# Proyecto

- Creación de Proyecto AgenciaDatos- Clase AutomovilDatos

```
using System;
using AgenciaDatos.Base;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace AgenciaDatos
{
    public class AutomovilDatos
    {
        private DbConnection db;
        public AutomovilDatos(string conexion)
        {
            db = new DbConnection(conexion);
        }
        // Método para consultar a la tabla de Automovil
        public List<T> consultarAutomovil<T>(Dictionary<string, dynamic> Parametros)
        {
            return db.Query<T>(Parametros, "[dbo].[AutomovilConsulta]");
        }
        //Método para buscar Automovil X Id
        public T consultarAutomovilXId<T>(Dictionary<string, dynamic> Parametros)
        {
            return db.QuerySingle<T>(Parametros, "[dbo].[AutomovilConsultaXId]");
        }
    }
}
```

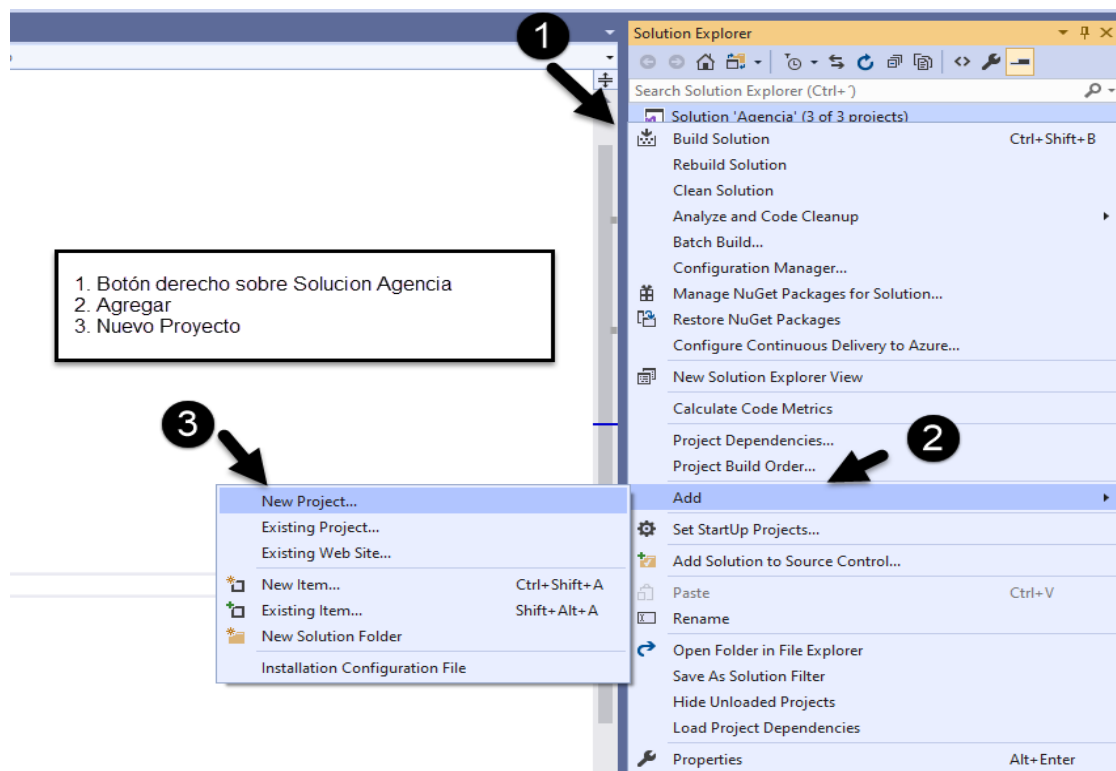
# Proyecto

- Creación de Proyecto AgenciaDatos- Clase AutomovilDatos

```
//Método para almacenar un automovil en la tabla de Automovil
public T insertarAutomovil<T>(Dictionary<string, dynamic> Parametros)
{
    return db.QuerySingle<T>(Parametros, "[dbo].[AutomovilInserta]");
}
//Método para actualizar Automovil
public T actualizarAutomovil<T>(Dictionary<string, dynamic> Parametros)
{
    return db.QuerySingle<T>(Parametros, "[dbo].[AutomovilActualiza]");
}
}
```

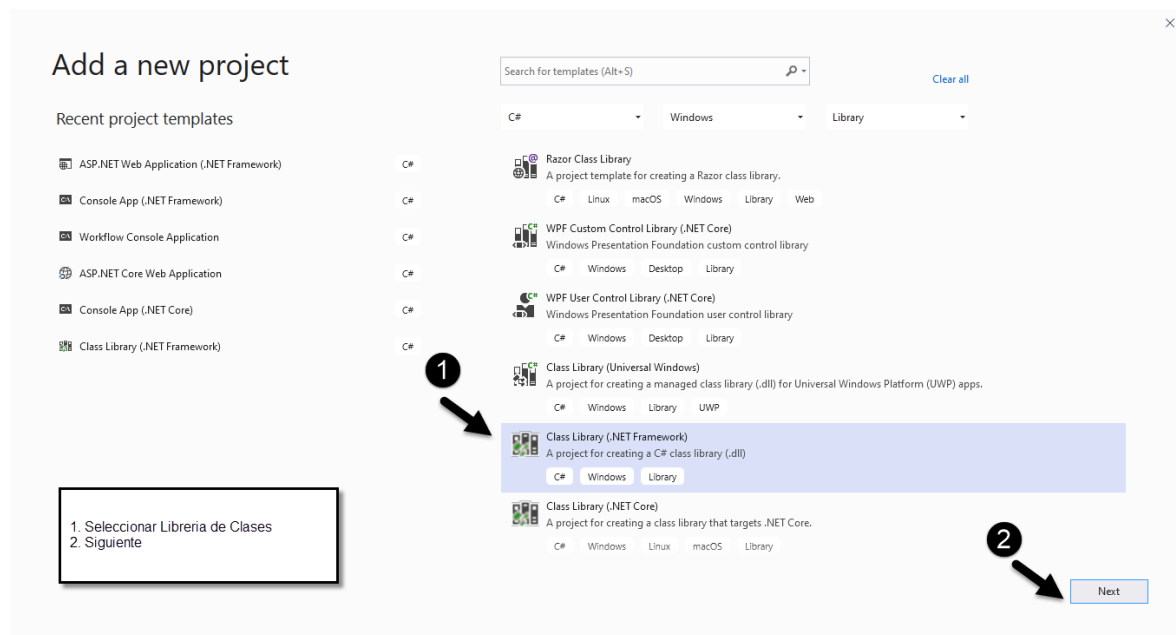
# Proyecto

- Creación de Proyecto AgenciaLogica



# Proyecto

- Creación de Proyecto AgenciaLogica



# Proyecto

- Creación de Proyecto AgenciaLogica

Configure your new project

Class Library (.NET Framework) C# Windows Library

Project name  
AgenciaLogica

Location  
C:\Users\user\Documents\Visual Studio 2019\Projects\Agencia

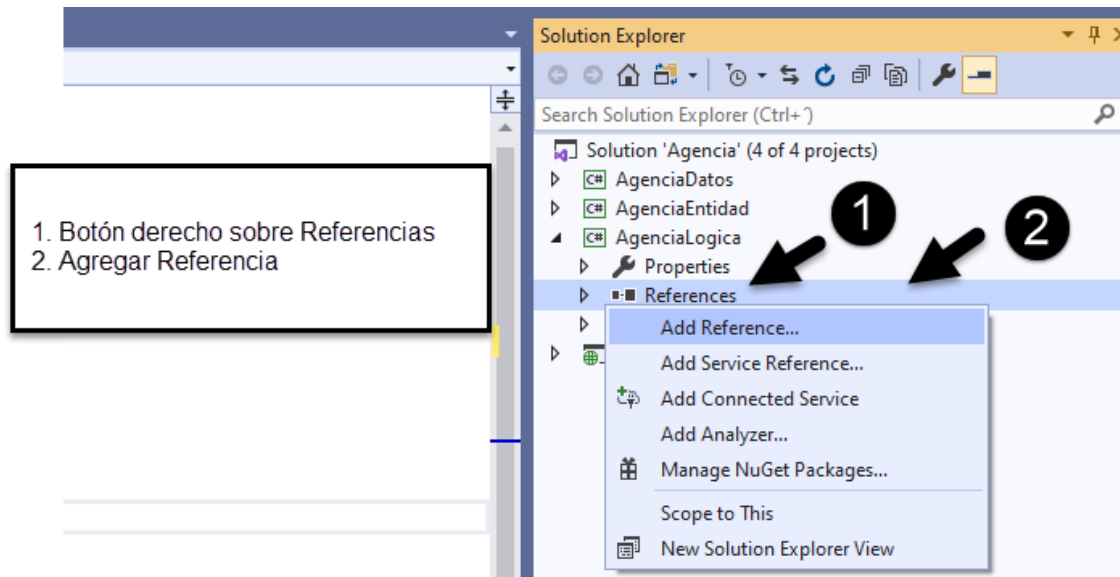
Framework  
.NET Framework 4.7.2

1. Indicar nombre de proyecto  
2. Seleccionar ubicacion  
3. Indicar versión de .NET Framework  
4. Crear

Back Create

# Proyecto

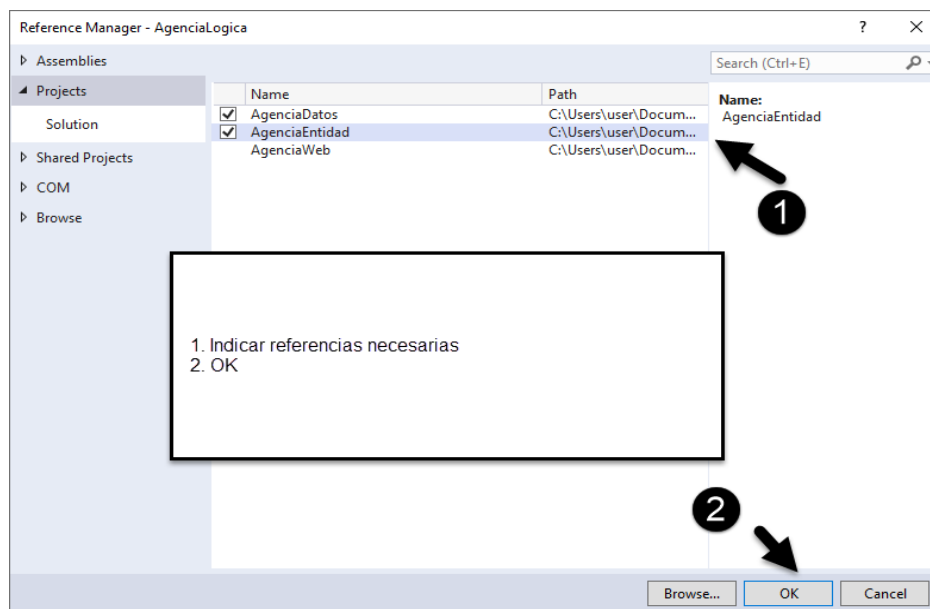
- Agregar Referencias AgenciaEntidad, AgenciaDatos





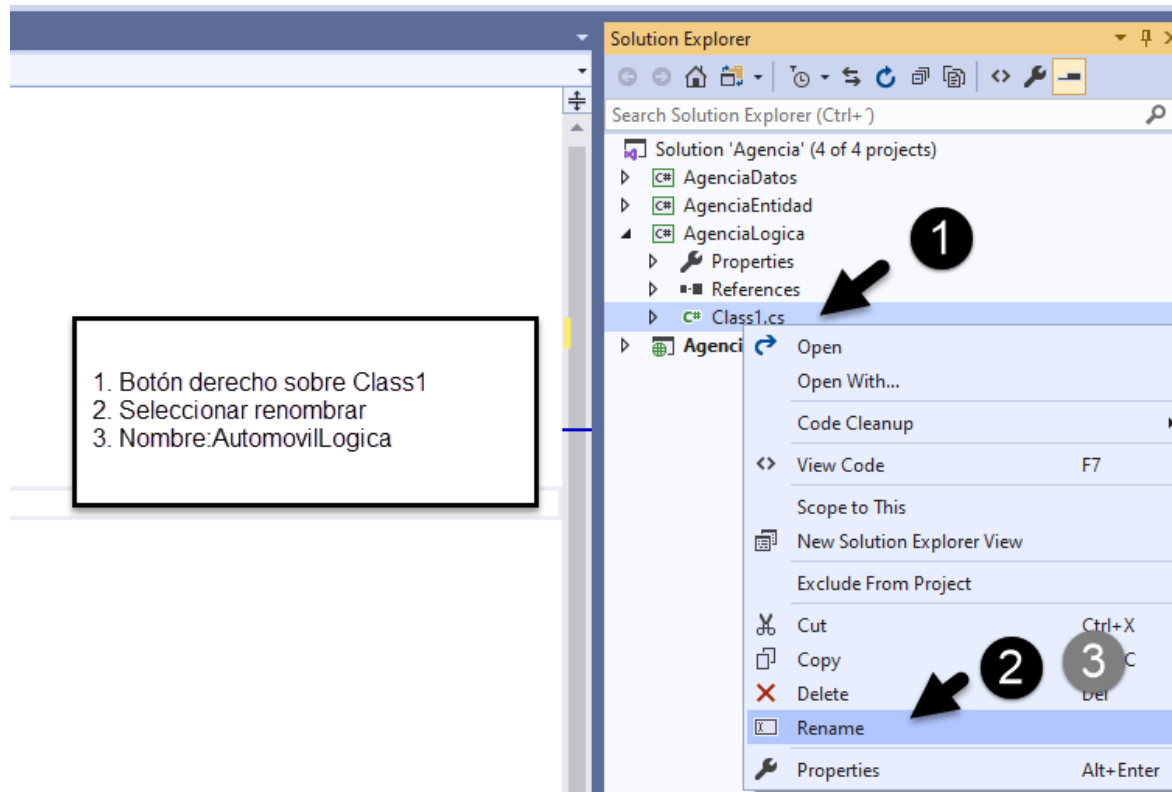
# Proyecto

- Selección de referencias



# Proyecto

- Renombrar Class1.cs por AutomovilLogica.cs



# Proyecto

- Clase AutomovilLogica

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using AgenciaDatos;
using AgenciaEntidad;
using AgenciaEntidad.Base;

namespace AgenciaLogica
{
    public class AutomovilLogica
    {
        private readonly AutomovilDatos automovilDatos;
        public AutomovilLogica()
        {
            automovilDatos = new AutomovilDatos("conexion");
        }
    }
}
```

# Proyecto

- Clase AutomovilLogica

```
//Método para consultar a la tabla de Automovil
public ResponseModel consultaAutomovil(Dictionary<string, dynamic> parametros)
{
    ResponseModel m = new ResponseModel();
    try
    {
        var res = automovilDatos.consultarAutomovil<AutomovilModel>(parametros);
        m.Data = res;
        m.Success = true;
        m.Message = "";
    }
    catch (Exception ex)
    {
        m.Data = null;
        m.Success = false;
        m.Message = ex.Message + ". " + ex.InnerException;
    }
    return m;
}
```

# Proyecto

- Clase AutomovilLogica

```
//Método para agregar un automovil
public RespuestaModel insertarAutomovil(Dictionary<string, dynamic> parametros)
{
    RespuestaModel m = new RespuestaModel();
    try
    {
        m = automovilDatos.insertarAutomovil<RespuestaModel>(parametros);
    }
    catch (Exception ex)
    {
        m.ErrorId = -2;
        m.Id = 0;
        m.MensajeRespuesta = ex.Message + ". " + ex.InnerException;
    }
    return m;
}
```

# Proyecto

- Clase AutomovilLogica

```
//Método para actualizar un Automovil
public RespuestaModel actualizarAutomovil(Dictionary<string, dynamic> parametros)
{
    RespuestaModel m = new RespuestaModel();
    try
    {
        m = automovilDatos.actualizarAutomovil<RespuestaModel>(parametros);
    }
    catch (Exception ex)
    {
        m.ErrorId = -2;
        m.Id = 0;
        m.MensajeRespuesta = ex.Message + ". " + ex.InnerException;
    }
    return m;
}
```

# Proyecto

- Clase AutomovilLogica

```
//Método para consultar AutomovilXId
public ResponseModel consultaAutomovilXId(Dictionary<string, dynamic> parametros)
{
    ResponseModel m = new ResponseModel();
    try
    {
        var res = automovilDatos.consultarAutomovilXId<AutomovilModel>(parametros);
        m.Data = res;
        m.Success = true;
        m.Message = "";
    }
    catch (Exception ex)
    {
        m.Data = null;
        m.Success = false;
        m.Message = ex.Message + ". " + ex.InnerException;
    }
    return m;
}
}
```