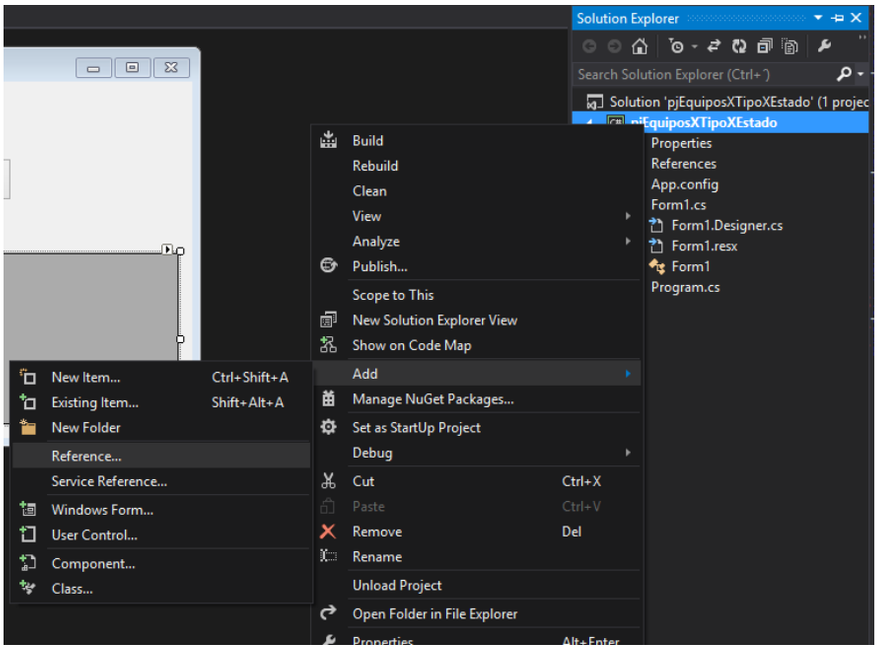
Ejercicio: Implementar un aplicativo que permita listar los equipos por estado y tipo.

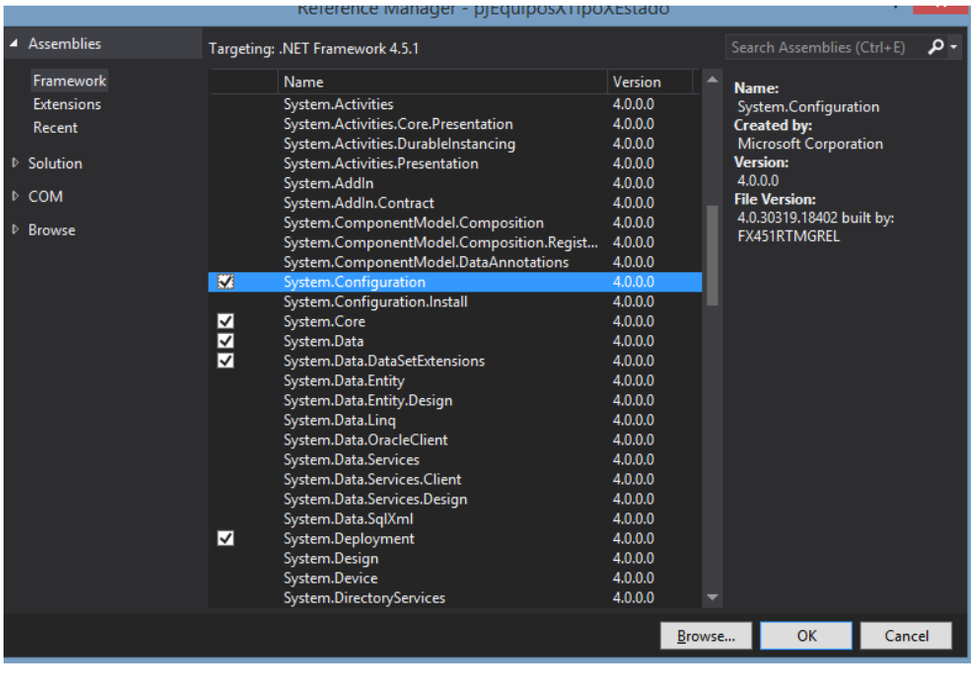
- Crear un procedimiento almacenado para listarlos
- Configurar la cadena de conexión en el archivo app.config
- Crear la clase Conexión y un método para asociarse a la cadena de conexión configurada en app.config
- Crear la clase “LogicAaNegocio”

Procedimientos almacenados

```
-- SP para listar los tipo de equipos
IF OBJECT_ID('SP_TIPOEQUIPO') IS NOT NULL
	DROP PROC SP_TIPOEQUIPO
GO

CREATE PROC SP_TIPOEQUIPO
AS
	SELECT T.COD_TIP_EQU AS CODIGO,
		T.DES_TIP AS TIPO
	FROM TIPO_EQUIPO T
GO

-- SP para listar estado de equipo
IF OBJECT_ID('SP_ESTADOEQUIPO') IS NOT NULL
	DROP PROC SP_ESTADOEQUIPO
GO

CREATE PROC SP_ESTADOEQUIPO
AS
	SELECT E.COD_EST AS CODIGO,
		E.DES_EST ESTADO
	FROM ESTADO_EQUIPO E
GO

-- SP para listar los equipos por estado y tipo
IF OBJECT_ID('SP_EQUIPOXESTADOXTIPO') IS NOT NULL
	DROP PROC SP_EQUIPOXESTADOXTIPO
GO

CREATE PROC SP_EQUIPOXESTADOXTIPO(@EST CHAR(6), @TIP CHAR(6))
AS
	SELECT E.IDE_EQU AS CODIGO,
		T.DES_TIP AS TIPO_EQUIPO,
		E.DESC_EQU AS DESCRIPCION,
		E.PREC_EQU AS PRECIO,
		ES.DES_EST AS ESTADO
	FROM EQUIPO E
	JOIN ESTADO_EQUIPO ES ON E.COD_EST = ES.COD_EST
	JOIN TIPO_EQUIPO T ON E.COD_TIP_EQU = T.COD_TIP_EQU
	WHERE E.COD_EST = @EST AND E.COD_TIP_EQU = @TIP
GO
```

Agregar la referencia





Agregar la cadena de conexión

```
<configuration>
    <startup>
        <supportedRuntime version="v4.0" sku=".NETFramework,Version=v4.5.1" />
    </startup>

    <!-- Cadena de conexión-->
    <connectionStrings>
      <add name="cn" connectionString="server=.;
                                       database=CONTRATO;
                                       integrated security=SSPI"/>
    </connectionStrings>
</configuration>
```

Crear la clase “Conexion”

```
namespace pjEquiposXTipoXEstado
{
    2 references
    class Conexion
    {
        // Método para la conexión
        3 references
        public SqlConnection getConexion()
        {
            SqlConnection sqlConnection = new SqlConnection(
                        ConfigurationManager.ConnectionStrings["cn"].ConnectionString);
            return sqlConnection;
        }
    }
}
```

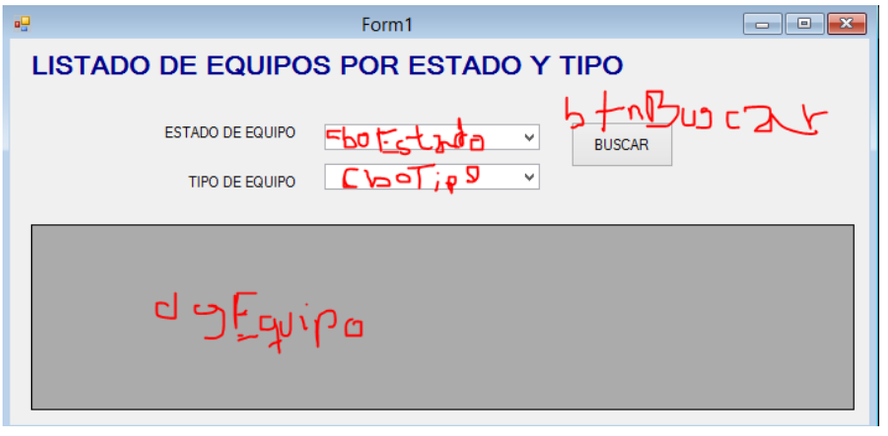
Crear la clase “LogicaNegocio”

```
namespace pjEquiposXTipoXEstado
{
    2 references
    class LogicaNegocio
    {
        // Variables globales
        Conexion conexion = new Conexion();
        SqlConnection sqlConnecion;

        // Métood para listar los tipo de equipos (para el combo)
        1 reference
        public  DataTable tipoEquiposCbo()
        {
            sqlConnecion = conexion.getConexion(); // Conexión
            // Trabajando con sql y no con el SP
            string sql = "SELECT T.COD_TIP_EQU AS CODIGO," +
                         "T.DES_TIP AS TIPO FROM TIPO_EQUIPO T";
            SqlDataAdapter sqlDataAdapter = new SqlDataAdapter(sql, sqlConnecion);
            // Objeto dataTable para llenarlo
            DataTable dataTable = new DataTable();
            sqlDataAdapter.Fill(dataTable);
            return dataTable;
        }
```

```
// Método para listar los tipos de equipos (para el combo de estados)
1 reference
public DataTable estadoEquipoCbo()
{
    sqlConnecion = conexion.getConexion(); // Conexión
    SqlDataAdapter sqlDataAdapter = new SqlDataAdapter("SP_ESTADOEQUIPO", sqlConnecion);
    DataTable dataTable = new DataTable();
    sqlDataAdapter.Fill(dataTable);
    return dataTable;
}
```

```
// Método para listar los equipos por estado y tipo
1 reference
public DataTable EquiposXEstadoXTipoQry(string estado, string tipo)
{
    sqlConnecion = conexion.getConexion(); // Conexión
    sqlConnecion.Open(); // Abrir la conexión
    SqlDataAdapter sqlDataAdapter = new SqlDataAdapter("SP_EQUIPOXESTADOXTIPO", sqlConnecion);
    // Para agregar parámetros (SP lo requiere), se necesita especificar el tipo
    sqlDataAdapter.SelectCommand.CommandType = CommandType.StoredProcedure;
    // Agregando el parámetro estado
    sqlDataAdapter.SelectCommand.Parameters.Add("EST", SqlDbType.Char).Value = estado;
    // Agregando el parámetro tipo
    sqlDataAdapter.SelectCommand.Parameters.Add("TIP", SqlDbType.Char).Value = tipo;

    DataTable dataTable = new DataTable();
    sqlDataAdapter.Fill(dataTable);
    return dataTable;
}
```

Formulario



```
public partial class Form1 : Form
{
    // Objeto Logica de Negocio
    private LogicaNegocio logicaNegocio = new LogicaNegocio();
    1 reference
    public Form1()
    {
        InitializeComponent();
    }

    // Método para llenar los tipo en el combo
    1 reference
    private void tipoEquipoCbo()
    {
        cboTipo.DataSource= logicaNegocio.tipoEquiposCbo();
        cboTipo.DisplayMember = "TIPO";
        cboTipo.ValueMember = "CODIGO";
    }
```

```
// Método para llenar los estados en el combo
1 reference
private void estadoEquipoCbo()
{
    cboEstado.DataSource = logicaNegocio.estadoEquipoCbo();
    cboEstado.DisplayMember = "ESTADO";
    cboEstado.ValueMember = "CODIGO";
}

1 reference
private void Form1_Load(object sender, EventArgs e)
{
    tipoEquipoCbo();
    estadoEquipoCbo();
}
```

```
// Botón buscar
1 reference
private void btnBuscar_Click(object sender, EventArgs e)
{
    string estado = cboEstado.SelectedValue.ToString();
    string tipo = cboTipo.SelectedValue.ToString();

    // Agregar al dataGridView
    dgEquipo.DataSource = logicaNegocio.EquiposXEstadoXTipoQry(estado, tipo);
}
```