Integrantes del Grupo (min 2, máx 4)
Adrián Aguado Fernández
Ignacio Barquilla Fernández
Miguel Portero Ruiz
Alberto Villar Díaz

Proyecto final de curso 1

Realización de un proyecto que ponga en contexto los diferentes conocimientos adquiridos en este primer curso de los módulos:

- Programación
- Bases de Datos
- Entornos de Desarrollo

¿Qué hay que hacer?

Diseñaremos e implementaremos una aplicación **sencilla** que haga uso de una base de datos para la persistencia y almacenamiento de información. Estructuraremos la realización del proyecto en 4 fases y por tanto 4 entregas, que contarán para la nota de las asignaturas de la siguiente forma:

Entrega 1:

Descripción de la aplicación que se quiere realizar + Diseño con UML (diagrama casos de uso y de secuencia o estado según aplique) + Plan de pruebas

Fecha de entrega: 18/05/2018

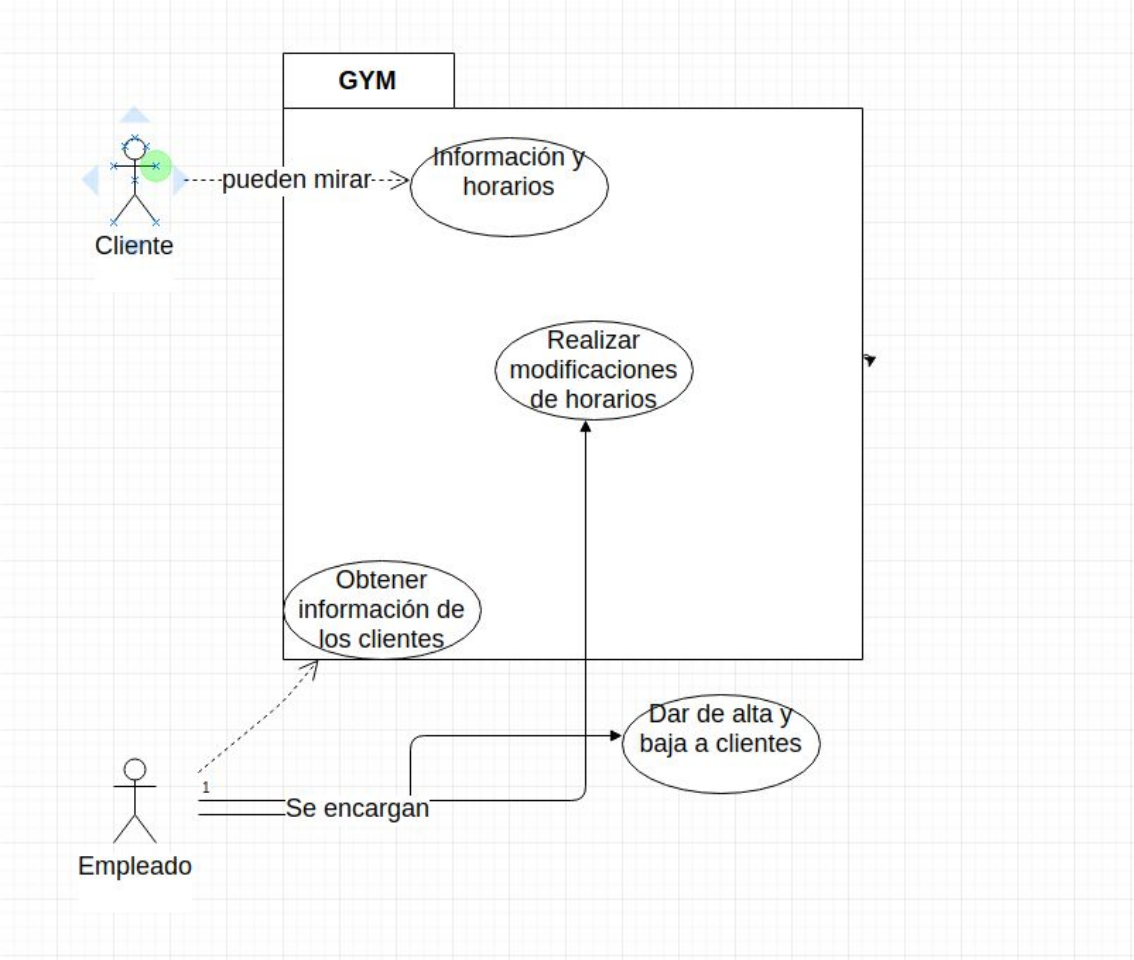
Cuenta para la nota de Entornos de Desarrollo*

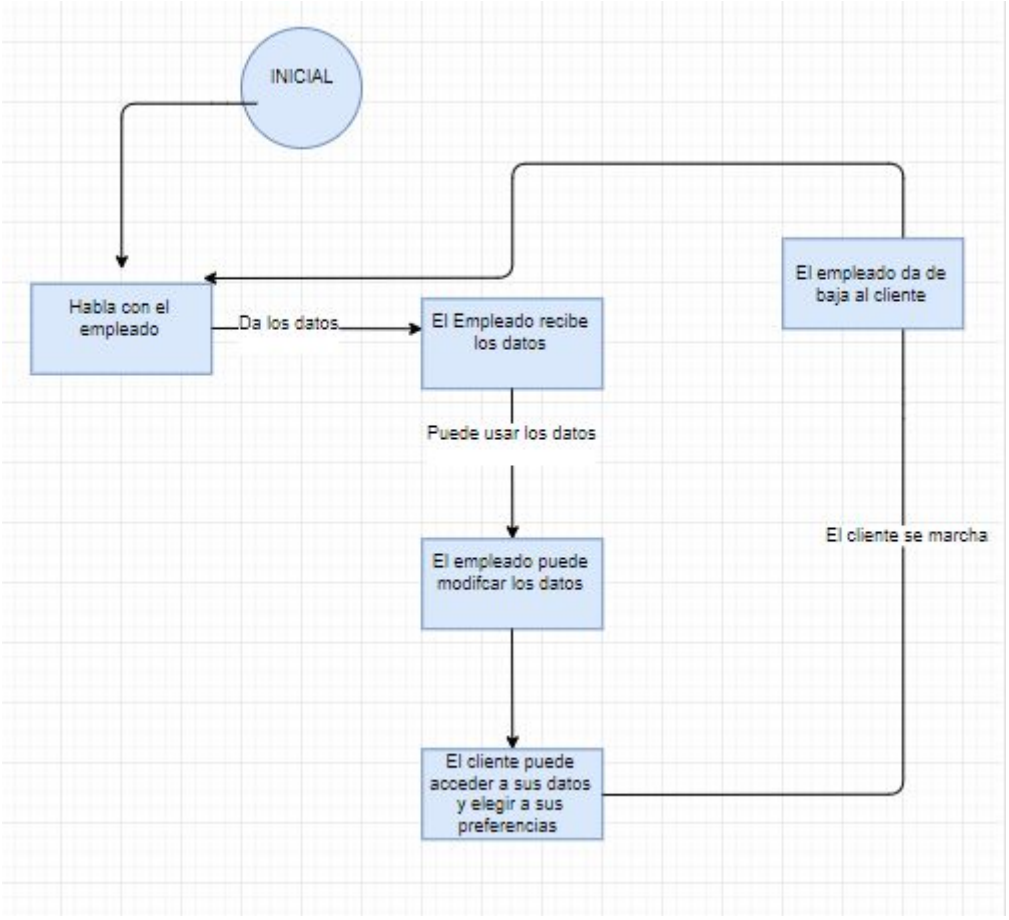
Aplicación

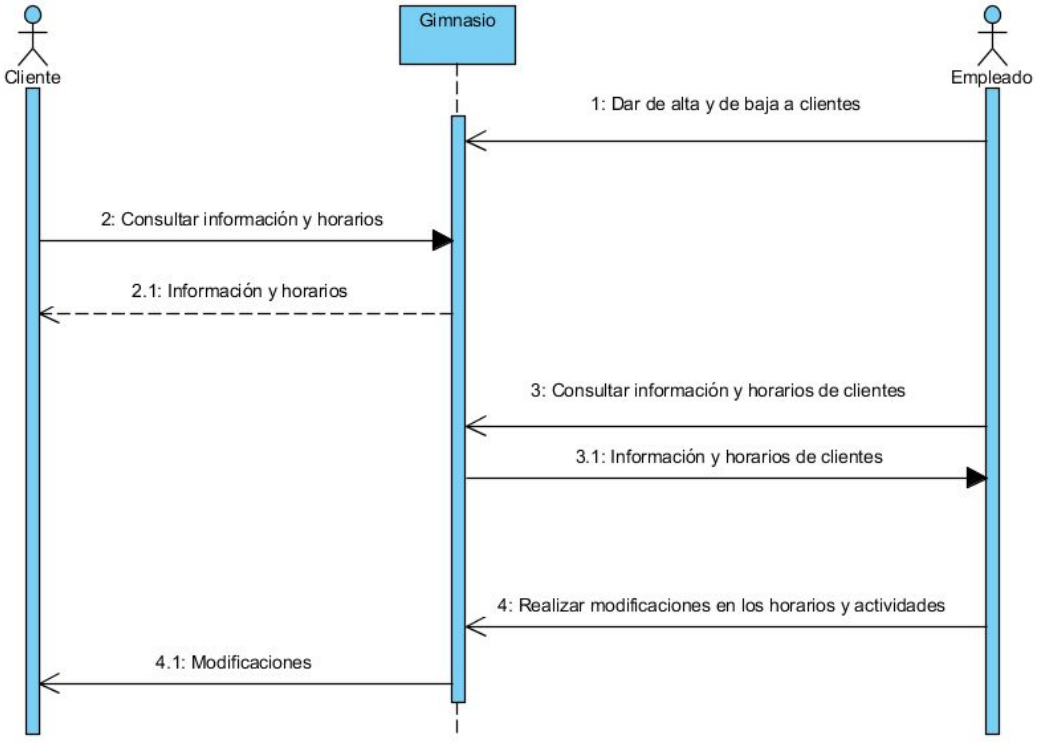
Realizaremos una aplicación para la total gestión de clientes de un gimnasio. En nuestra aplicación podremos dar de alta a los clientes, de baja, la cuota que tendrán y el modelo de cuota(mensual, trimestral, anual).

Así mismo, en los datos con referencia a los clientes tendremos: nombre, apellido, nif, fecha de nacimiento, miembro activo o de baja, el tipo de cuota que tienen, IBAN, email y dirección.

Los trabajadores del gimnasio tendrán acceso a la información para poder mirarla y modificarla. Así mismo, los clientes podrán visualizar el horario de clases y seleccionar a la que asistirá. Se creará un informe de las clases más visitadas y horario más escogido.

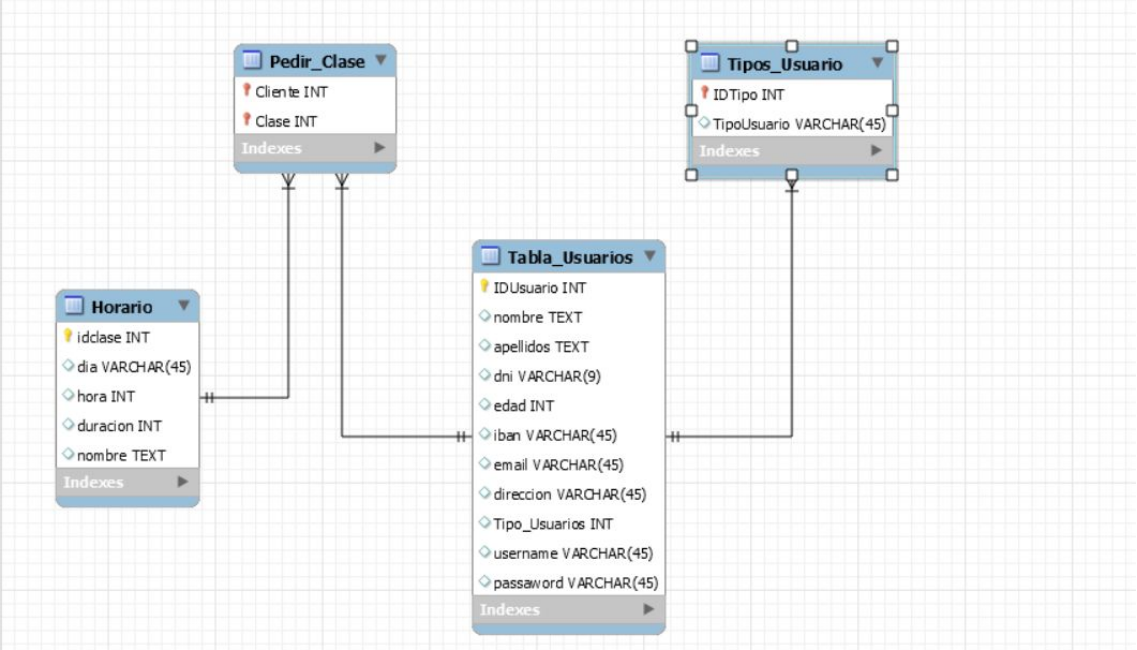

GYM
Cliente
pueden mirar
Información y horarios
Realizar modificaciones de horarios
Obtener información de los clientes
Dar de alta y baja a clientes
1
Se encargan
Empleado


INICIAL
Habla con el empleado
Da los datos
El Empleado recibe los datos
Puede usar los datos
El empleado puede modifcar los datos
El cliente puede acceder a sus datos y elegir a sus preferencias
El empleado da de baja al cliente
El cliente se marcha


Cliente
Gimnasio
Empleado
1: Dar de alta y de baja a clientes
2: Consultar información y horarios
2.1: Información y horarios
3: Consultar información y horarios de clientes
3.1: Información y horarios de clientes
4: Realizar modificaciones en los horarios y actividades
4.1: Modificaciones

Entrega 2:

Diagrama E/R y diagrama del modelo de BBDD + script capa de datos (todos los objetos de la base de datos, tablas, procedimientos, triggers, etc...).



Procedimiento 1: Editar datos de la tabla Tabla_Usuarios

```
Use Gimnasio

GO

/***** Se crea el procedimiento almacenado *****/

CREATE PROCEDURE EDITARUSUARIOS

/***** Se declaran los parámetros a usar *****/

@IDUsuario int,

@Nombre varchar,

@Apellidos varchar,

@DNI varchar(9),

@Edad int,

@IBAN varchar,

@EMail varchar,

@Direccion varchar,
```

```
@Tipos_Usuario int,
@Username varchar,
@Password varchar

AS

/***** Se hace una actualización / Modificación de la tabla y envían los parámetros *****/

UPDATE Tabla_Usuarios SET
nombre=@nombre, apellidos=@apellidos, DNI=@DNI, edad=@edad, IBAN=@IBAN,
EMail=@EMail, direccion=@direccion, Tipo_Usuarios=@Tipos_Usuario,
username=@username, passaword=@password
WHERE IDUsuario=@IDUsuario
GO
```

Procedimiento 2: Mostrar datos de la tabla Tabla_Usuarios

```
/*****El nombre de la base de datos *****/
Use Gimnasio
GO
/***** Se crea el procedimiento almacenado *****/
CREATE PROCEDURE MOSTRARUSUARIO
AS
/*****SE hace un select a la tabla y se envían los parámetros *****/
SELECT * FROM Tabla_Usuario
GO
```

Procedimiento 3: Insertar cliente

```
/***** El nombre de la base de datos *****/

USE Gimnasio

GO

/***** Se crea el procedimiento almacenado *****/

CREATE PROCEDURE INSERTARUSUARIO

/***** Se declaran los parámetros que usaremos *****/

@IDUsuario int,

@Nombre text,

@Apellidos text,

@DNI varchar(9),

@Edad int,

@IBAN varchar,

@EMail varchar,

@Direccion varchar,

@Tipos_Usuario int,

@Username varchar,

@Passaword varchar

AS

BEGIN

/***** Se hace un insert a la tabla y envían los parámetros *****/

INSERT INTO dbo.Tabla_Usuarios (idusuario, nombre, apellidos, dni, edad, iban, email,
direccion, Tipo_usuarios, username, passaword)

VALUES(@IDUsuario, @Nombre, @Apellidos, @DNI, @Edad,

@IBAN, @EMail, @Direccion, @Tipos_Usuario,

@Username, @Passaword)

END;
```

Procedimiento 4: Resumen

```
Create procedure Resumen
AS
BEGIN
SELECT nombre,dia, COUNT (*) FROM
Pedir_Clase
INNER JOIN Horario ON
Clase=IDclase
GROUP BY nombre, dia
END
```

Trigger1: Crear contraseña por defecto si se deja vacía

```
CREATE TRIGGER password_default
        ON Tabla_Usuarios AFTER INSERT AS
BEGIN
        declare @id int
        declare @passaword varchar(45)
        select @id=IDUsuario, @passaword = passaword from INSERTED
        if @passaword is null
                update Tabla_usuarios set passaword = '1234' where IDUsuario = @id
END;
```

Cursor1:

```
DECLARE @nombre varchar(45);
```

```
DECLARE cur1 CURSOR FOR SELECT email FROM Tabla_Usuarios;

OPEN cur1

FETCH NEXT FROM cur1 INTO @nombre

WHILE @@FETCH_STATUS = 0

BEGIN

print 'El e-mail es: '+@nombre

FETCH NEXT FROM cur1 INTO @nombre

END

CLOSE cur1

DEALLOCATE cur1
```

SCRIPT:

```
CREATE database Gimnasio

-- -----------------------------------------------------

-- Table Tipos_Usuario

-- -----------------------------------------------------

CREATE TABLE Tipos_Usuario (

  IDTipo INT NOT NULL DEFAULT 1,

  TipoUsuario VARCHAR(45) NULL DEFAULT 2,

  PRIMARY KEY (IDTipo))
```

```
-- -----------------------------------------------------
-- Table Tabla_Usuarios
-- -----------------------------------------------------
CREATE TABLE Tabla_Usuarios (
  IDUsuario INT NOT NULL,
  nombre TEXT NULL,
  apellidos TEXT NULL,
  dni VARCHAR(9) NULL,
  edad INT NULL,
  iban VARCHAR(45) NULL,
  email VARCHAR(45) NULL,
  direccion VARCHAR(45) NULL,
  Tipo_Usuarios INT NULL,
  username VARCHAR(45) NULL,
  passaword VARCHAR(45) NULL,
  PRIMARY KEY (IDUsuario),
   CONSTRAINT FKTipoUsuario
    FOREIGN KEY (Tipo_Usuarios)
    REFERENCES Tipos_Usuario (IDTipo))


-- -----------------------------------------------------
-- Table Horario
-- -----------------------------------------------------
CREATE TABLE Horario (
  idclase INT NOT NULL,
```

```
  dia VARCHAR(45) NULL,
  hora INT NULL,
  duracion INT NULL,
  nombre TEXT NULL,
  PRIMARY KEY (idclase))


-- -----------------------------------------------------
-- Table Pedir_Clase
-- -----------------------------------------------------
CREATE TABLE Pedir_Clase (
  Cliente INT NOT NULL,
  Clase INT NOT NULL,
  PRIMARY KEY (Cliente, Clase),
  INDEX FKClases_idx (Clase ASC),
  CONSTRAINT FKUsuarios
    FOREIGN KEY (Cliente)
    REFERENCES Tabla_Usuarios (IDUsuario),
  CONSTRAINT FKClases
    FOREIGN KEY (Clase)
    REFERENCES Horario (idclase))
```

Fecha de entrega: 25/05/2018

Cuenta para la nota de Bases de Datos*

Entrega 3:

Código aplicación (con comentarios en el código para documentación) + tests unitarios (métodos esenciales)

Fecha de entrega: 01/06/2018

Cuenta para la nota de Programación*

PANEL PRINCIPAL:

```
using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;


namespace Gestor_Gym

{

   public partial class Form3 : Form

   {

      public Form3()

      {

         InitializeComponent();

      }
```

```
        private void label1_Click(object sender, EventArgs e)
        {

        }

        private void Validar_Click(object sender, EventArgs e)
        {
            if (TXusuario.Text == "Alberto" && double.Parse(TXcontraseña.Text) == 1234) ;
            {
                MessageBox.Show("Bienvenidos a TheForceGym");
                Form2 llamar = new Form2();
                llamar.Show();
            }
        }
    }
}
```

PANEL QUE MUESTRA LOS USUARIOS:

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
```

```
using System.Windows.Forms;
using System.Configuration;
using System.Data.SqlClient;

namespace Gestor_Gym
{
   public partial class Form2 : Form
   {
      public Form2()
      {
         InitializeComponent();
         CargarUsuario();
      }

      private void button1_Click(object sender, EventArgs e)
      {
         CargarUsuario();
      }

      private void CargarUsuario()
      {
         //Leer archivo configuracion
         ConnectionStringSettings setting =
ConfigurationManager.ConnectionStrings["GimnasioConnectionString"];// Leer la coneection
                                                                    //Conectarme con
bbdd
         SqlConnection conn = new SqlConnection(setting.ConnectionString);//Lee
pasamos lo que hemos preparado
```

```
                                        //Seleccionar los campos de operacion
            string query = "SELECT IDUsuario, nombre, apellidos, edad, email FROM
Gimnasio.dbo.Tabla_Usuarios";
            SqlCommand cmdVerCO = new SqlCommand(query, conn); //Ver el campo de
operacion
                                        //Mostrarlos en el grid
            try
            {
               //Abrir la conexión
               conn.Open();
               //Ejecutar comando
               SqlDataReader rdr = cmdVerCO.ExecuteReader();

               DataTable dt = new DataTable();
               dt.Load(rdr); //Cargando una tabla donde le pasamos el DataReader

               this.dataGridView1.DataSource = dt;

               //Cierro el datareader, cerrara también la conexion
               rdr.Close();
            }
            catch (Exception ex)
            {
               MessageBox.Show("No se ha podido conectar" + ex.Message);
            }
         }
```

```
        private void button2_Click(object sender, EventArgs e)
        {
            Form frm = new Form1();
            frm.Show();
        }

        private void dataGridView1_CellContentClick(object sender,
DataGridViewCellEventArgs e)
        {

        }
    }
}
```

PANEL DE REGISTRO:

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using System.Data.SqlClient;
using System.Configuration;
```

```
namespace Gestor_Gym
{
    public partial class Form1 : Form
    {
        public Form1()
        {
            InitializeComponent();
        }

        private void button1_Click(object sender, EventArgs e)
        {
            //Lerr archivo configuracion
            ConnectionStringSettings setting =
ConfigurationManager.ConnectionStrings["GimnasioConnectionString"];
            //Conectarme con bbdd
            SqlConnection conn = new SqlConnection(setting.ConnectionString);
            SqlCommand cmdNuevoUsuariosp = new
SqlCommand("Gimnasio.dbo.INSERTARUSUARIO", conn);
            cmdNuevoUsuariosp.CommandType = CommandType.StoredProcedure;

            cmdNuevoUsuariosp.Parameters.Add(new SqlParameter("@IDUsuario",
SqlDbType.Int));
            cmdNuevoUsuariosp.Parameters["@IDUsuario"].Value =
Convert.ToInt32(id.Text);

            cmdNuevoUsuariosp.Parameters.Add(new SqlParameter("@Nombre",
SqlDbType.Text));
            cmdNuevoUsuariosp.Parameters["@Nombre"].Value = nombre.Text;
```

```
cmdNuevoUsuariosp.Parameters.Add(new SqlParameter("@Apellidos",
SqlDbType.Text));

cmdNuevoUsuariosp.Parameters["@Apellidos"].Value = apellidos.Text;


cmdNuevoUsuariosp.Parameters.Add(new SqlParameter("@DNI",
SqlDbType.VarChar,9));

cmdNuevoUsuariosp.Parameters["@DNI"].Value = dni.Text;


cmdNuevoUsuariosp.Parameters.Add(new SqlParameter("@Edad",
SqlDbType.Int));

cmdNuevoUsuariosp.Parameters["@Edad"].Value = Convert.ToInt32
(edad.Text);


cmdNuevoUsuariosp.Parameters.Add(new SqlParameter("@IBAN",
SqlDbType.VarChar,45));

cmdNuevoUsuariosp.Parameters["@IBAN"].Value = iban.Text;


cmdNuevoUsuariosp.Parameters.Add(new SqlParameter("@Email",
SqlDbType.VarChar, 45));

cmdNuevoUsuariosp.Parameters["@Email"].Value = email.Text;


cmdNuevoUsuariosp.Parameters.Add(new SqlParameter("@Direccion",
SqlDbType.VarChar, 45));

cmdNuevoUsuariosp.Parameters["@Direccion"].Value = direccion.Text;


cmdNuevoUsuariosp.Parameters.Add(new SqlParameter("@Tipos_Usuario",
SqlDbType.Int));

cmdNuevoUsuariosp.Parameters["@Tipos_Usuario"].Value =
Convert.ToInt32(tipo_usuario.Text);
```

```
            cmdNuevoUsuariosp.Parameters.Add(new SqlParameter("@Username",
SqlDbType.VarChar, 45));

            cmdNuevoUsuariosp.Parameters["@Username"].Value = username.Text;


            cmdNuevoUsuariosp.Parameters.Add(new SqlParameter("@Passaword",
SqlDbType.VarChar, 45));

            cmdNuevoUsuariosp.Parameters["@Passaword"].Value = password.Text;


            try

            {

               //abrir la conexion

               conn.Open();


               cmdNuevoUsuariosp.ExecuteNonQuery();

            }

            catch (Exception ex)

            {

               MessageBox.Show("no se ha podido insertar nuevo CO" + ex.Message);

            }

            finally

            {

               conn.Close();

            }

         }

         private void button2_Click(object sender, EventArgs e)

         {
```

```
            this.Close();
        }
    }
}
```

Entrega final:

Manual de usuario de la aplicación y entrega de las versiones definitivas de las entregas anteriores:

- Código de la aplicación y de base de datos terminado
- Documentación y diagramas corregidos si procede
- Plan de pruebas hecho (que se vea que hemos probado la aplicación)

Fecha de entrega: 8/06/2018

*Esta entrega es imprescindible para que la nota de las otras entregas cuente en la asignatura correspondiente.

Este día de entrega final, todos los grupos harán una pequeña **Demo para los compañeros.** Haremos una evaluación del proyecto.

En la elaboración del proyecto son requisitos indispensables:

- Que la aplicación **conecte con la base de datos**
  - ✓ puede ser cualquier lenguaje, java, C#, puede ser cualquiera de los dos SGBD vistos, mySQL, SQL Server, cualquier forma de conexión, la "fácil", directa, con entity framework, ADO, ...)
- Uso de **control de versiones** (Git, GitHub)
- Manejo de **excepciones** en la aplicación
- Uso de **procedimientos almacenados y funciones** en bbdd

Se valorará:

1. Que incluya aspectos técnicos y herramientas vistas en clase:
   a. Interfaz de usuario clara, organizada
   b. Lectura/escritura de ficheros
   c. Uso de colecciones/ficheros xml
   d. Clases heredadas, abstractas, interfaces
   e. Triggers, cursores, realización de consultas multitabla y agrupadas
2. La idea y el objetivo de la aplicación, que sea original, que genere tejido social (es decir, que sea útil desde un punto de vista social. CUIDADO, no escojáis algo muy difícil ni complejo que sea muy chulo pero que no os de tiempo de terminar, mejor algo simple pero que funcione.

3. La calidad de la documentación
4. El grado de finalización de la aplicación