



# **WEBESCUELA FINAL**

**Proyecto final de webEscuela  
(2)**

# BASE DE DATOS (1)

**Agregamos al script de base de datos el siguiente código**

```
--TABLA CARRERAS
create table Carreras(
    ID int identity (1,1) primary key,
    Sigla varchar(10)unique not null,
    Nombre varchar(40)unique not null,
    Duracion int not null,
    Titulo varchar(40)not null,
    Estado varchar(10) not null check(Estado='ACTIVO' or Estado='INACTIVO')
)
go
```

# BASE DE DATOS(2)

## Procedimiento Carreras:Insert

```
--Store procedures de Carreras
create procedure Carreras_Insert(
    @Sigla varchar(10),@Nombre varchar(40),@Duracion int,@Titulo varchar(40),@Estado varchar(10)
)
as begin
insert Carreras
values(@Sigla,@Nombre,@Duracion,@Titulo,@Estado)
Select @@identity
end
go
```



# BASE DE DATOS(3)

## Procedimiento Carreras:Update

```
create procedure Carreras_Update(  
@ID int,@Sigla varchar(10),@Nombre varchar(40),@Duracion int,@Titulo varchar(40),@Estado varchar(10)  
)  
as  
update Carreras set  
    Sigla=@Sigla,Nombre=@Nombre,Duracion=@Duracion,Titulo=@Titulo,Estado=@Estado  
where  
    ID=@ID  
go
```

# BASE DE DATOS(4)

## Procedimientos Carreras:Delete y Carreras\_Find

```
create procedure Carreras_Delete(@ID int)
as
Delete Carreras where ID=@ID
go
```

```
create procedure Carreras_Find(@ID int)
as
select * from Carreras where ID=@ID
go
```

# BASE DE DATOS(5)

## Procedimientos Carreras:List y Carreras\_FindBySigla

```
create procedure Carreras_List(@Estado varchar(10))  
as  
    select * from Carreras where Estado=@Estado  
go
```

```
create procedure Carreras_FindBySigla(@Sigla varchar(10))  
as  
Select * from Carreras where Sigla=@Sigla  
go
```

# BASE DE DATOS(6)

## Procedimientos Carreras:SiglaExists y Carreras NameExists

```
create procedure Carreras_SiglaExists(@ID int,@Sigla varchar(10))
as
    select Count(*) from Carreras where ID!=@ID and @sigla=Sigla
go
```

```
create procedure Carreras_NameExists(@ID int,@Nombre varchar(40))
as
    select Count(*) from Carreras where ID!=@ID and @Nombre=Nombre
go
```



A woman with short blonde hair, wearing a black and white striped long-sleeved shirt and light-colored pants, is sitting in a meditative lotus position on a dark grey sofa. Her eyes are closed, and she has a calm expression. Behind her is a large, white, multi-tiered bookshelf filled with numerous books of various colors. To the left and right of the bookshelf are two floor lamps with white conical shades. The scene is softly lit, creating a peaceful atmosphere.

# CAPA DE NEGOCIOS BUSSINESSCHOOL

[Ver video de base de](#)



# INTERFACE ICARRERA

**Hereda de IABMFIMG (por poseer imagen)**

**Sobre el proyecto BUSSINESSCHOOL agregamos un nuevo elemento de tipo Interface Denominado**

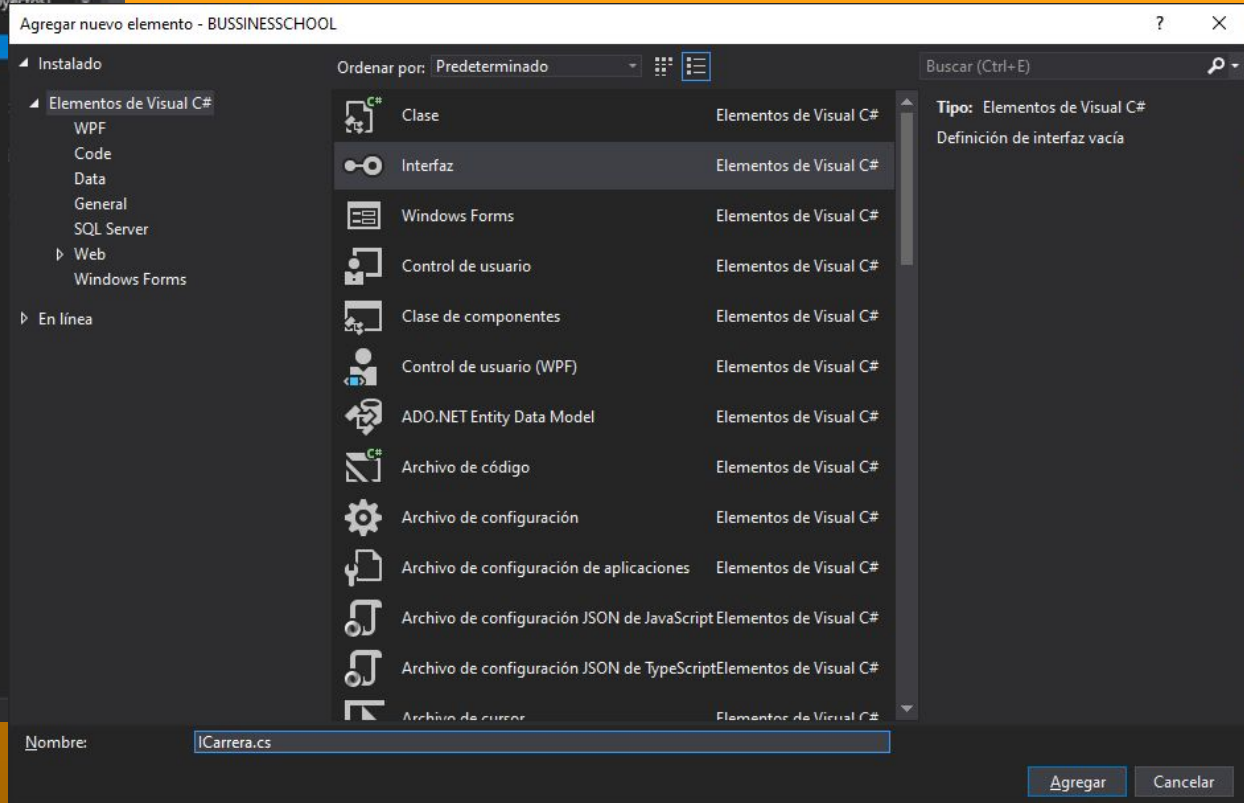
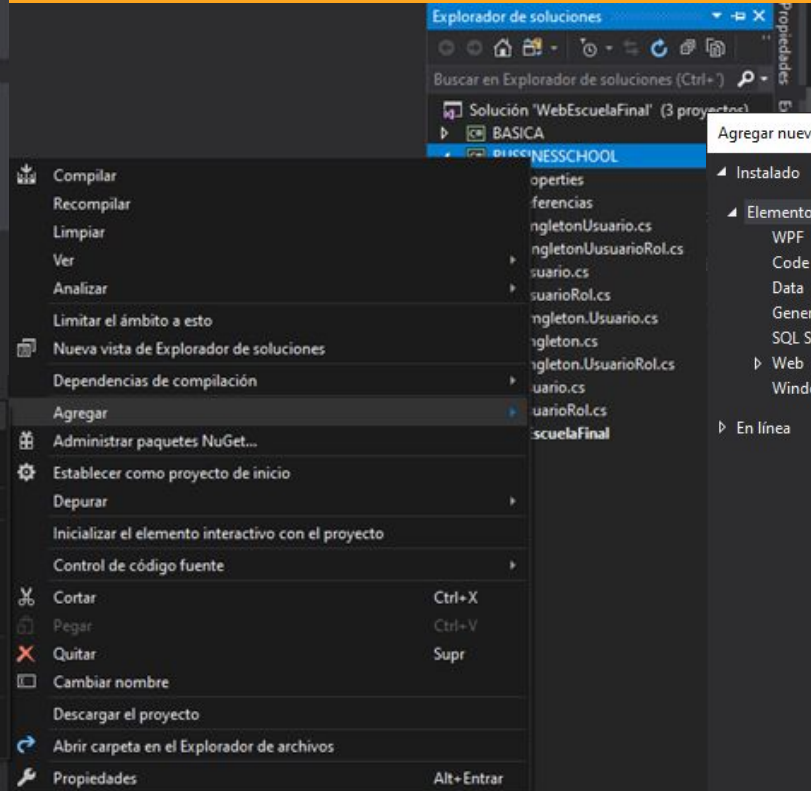
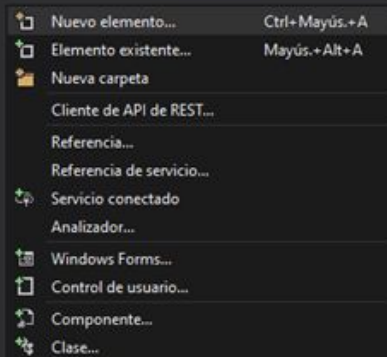
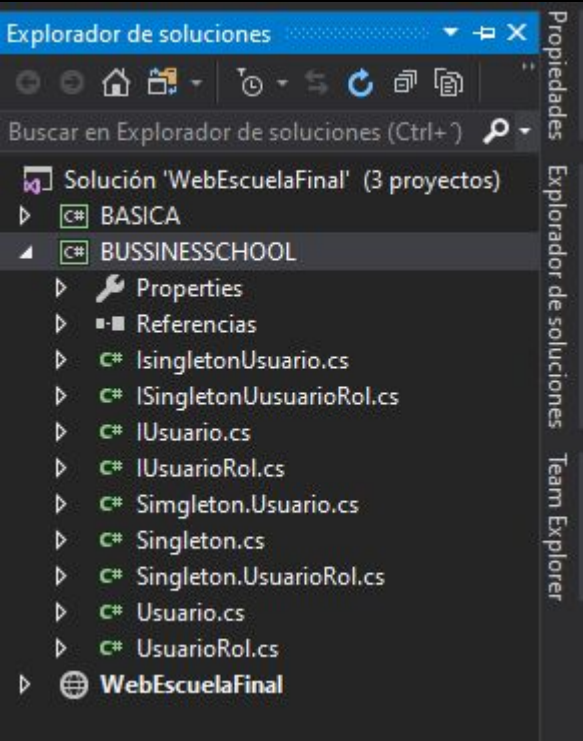
**Debe ser de acceso público**

**Debe contener un HttpPostedFile FUPdf**

**Debe Cumplir con el dataContract de la tabla Carreras**

**Debe Cumplir con el MethodContract establecidos por los procedimientos de la tabla Carreras**

# AGREGAR ICARRRE



```
namespace BUSSINESSCHOOL
{
    public interface ICarrera:BASICA.IABMFIMG
    {
        #region DataContract
        string Sigla { get; set; }
        string Nombre { get; set; }
        string Titulo { get; set; }
        int Duracion { get; set; }
        string Estado { get; set; }
        #endregion
        System.Web.HttpPostedFile FUPdf { get; set; }
        #region MethodContract
        bool NameExists();
        bool SiglaExists();
        string List();
        string FindBySigla();
        #endregion
    }
}
```

# CÓDIGO DE ICARRERA



# CÓDIGO CLASS CARRERA

```
using System.Web;
namespace BUSSINESSCHOOL
{
    public class Carrera : ICarrera
    {
        internal Singleton S => Singleton.GetInstance;
       DataContract
        MethodContract
    }
}
```

Referencia a Singleton

```
#region DataContact
public string Sigla { get ; set;}
public string Nombre { get; set;}
public string Titulo { get; set;}
public int Duracion { get; set;}
public string Estado { get; set;}
public HttpPostedFile FUPdf { get; set; }
public string DirBase =>"IMAGENES";
public string DirectorY => "CARRERAS";
public string Prefix => "CARRERA";
public HttpPostedFile Fu { get; set; }
public int ID { get ; set;}
public string MakeData =>ID+";" +DirBase + ";" + DirectorY + ";" + Prefix + ".jpg";
public string MakedataPDF => ID + ";" + DirBase + ";" + DirectorY + ";" + Prefix + ".pdf";
#endregion
```

```
#region DataContact
public string Sigla { get ; set;}
public string Nombre { get; set;}
public string Titulo { get; set;}
public int Duracion { get; set;}
public string Estado { get; set;}
public HttpPostedFile FUPdf { get; set; }
public string DirBase =>"IMAGENES";
public string DirectorY => "CARRERAS";
public string Prefix => "CARRERA";
public HttpPostedFile Fu { get; set; }
public int ID { get ; set;}
#endregion
```

# INTERFACE ISINGLETONCARRERA

**Agrego a BUSSINESSCHOOL una interface IsingletonCarrera**

**Es de acceso internal**

**Hereda de IGenericSingleton<Carrera>**

**Implementa el MethodContract de ICarrera pasando como  
parámetros  
Carrera Data**

# CÓDIGO DE ISINGLETONCARRERA

```
namespace BUSSINESSCHOOL
{
    internal interface ISingletonCarrera: BASICA.IGenericSingleton<Carrera>
    {
        bool NameExists(Carrera Data);
        bool SiglaExists(Carrera Data);
        string List(Carrera Data);
        string FindBySigla(Carrera Data);
    }
}
```



# IMPLEMENTACIÓN EN SIGLETON

**Agrego a BUSSINESSCHOOL un nuevo elemento tipo Clase  
Singleton.Carrera,cs**

**Es una internal partial class Singleton**

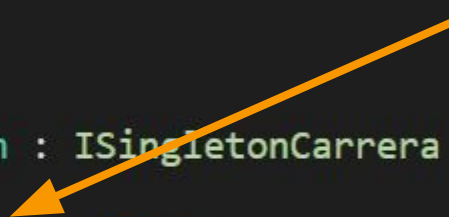
**Implementa la interface ISingletonCarrera en forma Explícita**

**Agrego una Property pública de tipo ISingletonCarrera de  
nombre**

**ISC que apunta a Singleton**

# PARTIAL CLASS SINGLETON (1)

```
using System.Data;
using BASICA;
namespace BUSSINESSCHOOL
{
    internal partial class Singleton : ISingletonCarrera
    {
        public ISingletonCarrera ISC => this;
        ISingletonCarrera
    }
}
```



AGREGO LA  
REFERENCIA PÚBLICA  
ISC

# PARTIAL CLASS SINGLETON (2)

```
public ISingletonCarrera ISC => this;
#region ISingletonCarrera
void IGenericSingleton<Carrera>.Add(Carrera Data)
{
    if (Data.SiglaExists()) throw new System.Exception("Error ya existe otra carrera con la misma sigla");
    if (Data.NameExists()) throw new System.Exception("Error ya existe otra carrera con el mismo Nombre");
    IConnection.CreateCommand("Carreras_Insert", "Carrera");
    //agrego paramentros en el mismo orden del Store Procedure
    IConnection.AddVarchar("Sigla", 10, Data.Sigla);
    IConnection.AddVarchar("Nombre", 40, Data.Nombre);
    IConnection.AddInt("Duracion", Data.Duracion);
    IConnection.AddVarchar("Titulo", 40, Data.Titulo);
    IConnection.AddVarchar("Estado", 10, Data.Estado);
    Data.ID = IConnection.Insert();
    IImage.Add(Data.Fu, Data.MakeData);
    IImage.Add(Data.FUPdf, Data.MakedataPDF);
}
```

Se guarda la imagen si  
FU la ha cargado

Se guarda el pdf si  
FUPdf lo ha cargado



# PARTIAL CLASS SINGLETON (3)

```
void IGenericSingleton<Carrera>.Modify(Carrera Data)
{
    if (Data.SiglaExists()) throw new System.Exception("Error ya existe otra carrera con la misma sigla");
    if (Data.NameExists()) throw new System.Exception("Error ya existe otra carrera con el mismo Nombre");
    IConnection.CreateCommand("Carreras_Update", "Carrera");
    //agrego paramentros en el mismo orden del Store Procedure
    IConnection.AddInt("ID", Data.ID);
    IConnection.AddVarchar("Sigla", 10, Data.Sigla);
    IConnection.AddVarchar("Nombre", 40, Data.Nombre);
    IConnection.AddInt("Duracion", Data.Duracion);
    IConnection.AddVarchar("Titulo", 40, Data.Titulo);
    IConnection.AddVarchar("Estado", 10, Data.Estado);
    IConnection.Update();
    IImage.Add(Data.Fu, Data.MakeData);
    IImage.Add(Data.FUPdf, Data.MakedataPDF);
}
```

Se guarda la imagen si  
FU la ha cargado

Se guarda el pdf si  
FUPdf lo ha cargado

# PARTIAL CLASS SINGLETON (4)

```
void IGenericSingleton<Carrera>.Erase(Carrera Data)
{
    IConnection.CreateCommand("Carreras_Delete", "Carrera");
    IConnection.AddInt("ID", Data.ID);
    IConnection.Delete();
    IImage.Erase(Data.MakeData);
    IImage.Erase(Data.MakedataPDF);
}
```

SI TIENE  
UNA IMAGEN LA ELIMINA

SI TIENE  
UN PDF LO ELIMINA

```
string IGenericSingleton<Carrera>.Find(Carrera Data)
{
    IConnection.CreateCommand("Carreras_Find", "Carrera");
    IConnection.AddInt("ID", Data.ID);
    DataRow Dr = IConnection.Find();
    return ISC.MakeJson(Dr, Data);
}
```

INVOCA A MAKEJSON PARA  
DEVOLVER UN JSON Y  
CARGAR DATA DESDE Dr

# PARTIAL CLASS SINGLETON (5)

```
string ISingletonCarrera.FindBySigla(Carrera Data)
{
    IConnection.CreateCommand("Carreras_FindBySigla", "Carrera");
    IConnection.AddVarchar("Sigla", 10, Data.Sigla);
    DataRow Dr = IConnection.Find();
    return ISC.MakeJson(Dr, Data);
}
```

INVOCA A MAKEJSON PARA  
DEVOLVER UN JSON Y CARGAR  
DATA DESDE Dr



# PARTIAL CLASS SINGLETON (6)

```
string ISingletonCarrera.List(Carrera Data)
{
    IConnection.CreateCommand("Carreras_List", "Carreras");
    IConnection.AddVarchar("Estado", 10, Data.Estado);
    DataTable Dt = IConnection.List();
    return IListToTable.TableToJson<Carrera>(Dt, ISC);
}
```



**IListToTable** MANEJA LOS LISTADOS;  
APUNTA A UNA CLASE QUE MANEJA FUNCIONES GENERICAS  
LA INTERFACE ISC FUNCIONA POR SER HEREDERA DE  
**IGENERICSINGLETON<Carrera>**  
**Carrera** CUMPLE POR TENER UN CONSTRUCTOR DEFAULT

# PARTIAL CLASS SINGLETON (7)

```
bool ISingletonCarrera.NameExists(Carrera Data)
{
    IConnection.CreateCommand("Carreras_NameExists", "Carrera");
    IConnection.AddInt("ID", Data.ID);
    IConnection.AddVarchar("Nombre", 40, Data.Nombre);
    return IConnection.Exists();
}

bool ISingletonCarrera.SiglaExists(Carrera Data)
{
    IConnection.CreateCommand("Carreras_Update", "Carrera");
    IConnection.AddInt("ID", Data.ID);
    IConnection.AddVarchar("Sigla", 10, Data.Sigla);
    return IConnection.Exists();
}
```

# PARTIAL CLASS SINGLETON (7)

```
string IGenericSingleton<Carrera>.MakeJson(DataRow Dr, Carrera Data)
{
    string json = "{}";
    string s = "ID";
    Data.ID = int.Parse(Dr[s].ToString());
    json = IJson.addObject(json, s, Dr[s].ToString());
    s = "Nombre";
    Data.Nombre = Dr[s].ToString();
    json = IJson.addObject(json, s, Dr[s].ToString());
    s = "Sigla";
    Data.Sigla = Dr[s].ToString();
    json = IJson.addObject(json, s, Dr[s].ToString());
    s = "Titulo";
    Data.Titulo = Dr[s].ToString();
    json = IJson.addObject(json, s, Dr[s].ToString());
    s = "Duracion";
    Data.Duracion = int.Parse(Dr[s].ToString());
    json = IJson.addObject(json, s, Dr[s].ToString());
    string URL = IImage.URL(Data.MakeData);
    json = IJson.addObject(json, "Url", URL);
    string URLPdf= IImage.URL(Data.MakedataPDF);
    json = IJson.addObject(json, "UrlPDF", URLPdf);
    return json;
}
```

CARGA EL DATO DE Dr A Data Y  
DEVUELVE EL Json DEL OBJETO

CAPTURA LA URL DE LA  
IMAGEN DE LA CARRERA

CAPTURA EL PDF DE LA  
CARRERA

# CLASS CARRERA IMPLEMENTACIÓN DE METHOD CONTRACT

```
using System.Web;
namespace BUSSINESSCHOOL
{
    public class Carrera : ICarrera
    {
        internal Singleton S => Singleton.GetInstance;
        DataContact
        #region MethodContract
        public void Add()          {S.ISC.Add(this);}
        public void Erase()       { S.ISC.Erase(this); }
        public string Find()      {return S.ISC.Find(this);}
        public string FindBySigla(){ return S.ISC.FindBySigla(this);}
        public string List()      {return S.ISC.List(this);}
        public void Modify()      {S.ISC.Modify(this);}
        public bool NameExists()  {return S.ISC.NameExists(this);}
        public bool SiglaExists() {return S.ISC.SiglaExists(this); }
        #endregion
    }
}
```



A group of four students are gathered around a table in a library, smiling and looking at a laptop. The background is filled with bookshelves. A black rectangular box is positioned at the top left of the image.

**BUSSINESSCHOOL**

# CAPA DE PRESENTACIÓN WEBESCUELA FINAL

# CAMBIO EN TABLAS.CSS

```
section iframe.iframepdf {  
    width: 100%;  
    height: 30rem;  
    margin-top: 5rem;  
}
```

# CAMBIO EN FORM.JS

## \$\$CAREERSFORM()

### ADDCAREER()

Formato de addCareer  
(Se suprime Requerimientos)

### NUEVA CARRERA

NOMBRE

SIGLA

DURACIÓN

TÍTULO

IMAGEN

Examinar...

PLAN DE ESTUDIOS

Examinar...

ESTADO

ACTIVO

AGREGAR CARRERA

CANCELAR



# CAMBIO EN FORM.JS

## \$\$CAREERSFORM() (1)

```
this.addCareer = function () {  
  const Flood = function (...);  
  let f = $df.form("agregar carrera", "form w60 l28", "Nueva carrera");  
  f.action = "default.aspx"; f.method = "POST"; f.enctype = "multipart/form-data"; f.target = "ifr";  
  //f.onsubmit = Submit;  
  let ifr = $db.ce("iframe");ifr.name = "ifr";ifr.id = "ifr";$db.ac(Section, ifr);ifr.className = "hidden";ifr.onload = Flood;  
  let nombre = $dlc.labelAndText("nombre", "Nombre");nombre.required = true;  
  let sigla = $dlc.labelAndText("Sigla", "Sigla");sigla.required = true;  
  let duracion = $dlc.labelAndNumber("duración", "Duracion");duracion.required = true;  
  let titulo = $dlc.labelAndText("título", "Titulo");titulo.required = true;  
  //let requisitos = $dlc.labelAndText("Requisitos", "Requisitos");requisitos.required = true;  
  $dlc.labelAndFileImg("Imagen", "Img");  
  $dlc.labelAndFilePdf("Plan de estudios", "Pdf");  
  $dc.inputHidden("accion", "ADDCARRERA");  
  let select = $dlc.labelAndSelect("Estado", "Estado", ["ACTIVO", "INACTIVO"], ["ACTIVO", "INACTIVO"]);  
  $tc.makeTableCareers();  
  return f;  
};
```

```
const Flood = function ()  
{  
  let content = this.contentWindow.document.childNodes[0].innerText;  
  if (content === null || content === undefined || content === "") return;  
  if (content === "OK") $cf.addCareer();  
  else alert(content);  
};
```



# DEFAULT.ASPX.CS

```
private void ModifyCarrera()  
{  
    Carrera c = new Carrera();  
    c.Nombre = Request["Nombre"];  
    c.Duracion = int.Parse(Request["Duracion"]);  
    c.Titulo = Request["Titulo"];  
    c.Estado = Request["Estado"];  
    c.Sigla = Request["Sigla"];  
    c.ID = int.Parse(Request["ID"]);  
    if (Request.Files.Count > 0)  
    {  
        for (int i = 0; i < Request.Files.Count; i++)  
            if (Request.Files[i].FileName.EndsWith("pdf")) c.FUPdf = Request.Files[i];  
            else if (Request.Files[i].FileName != "") c.Fu = Request.Files[i];  
    }  
    try  
    {  
        c.Modify();  
        Response.Write("OK");  
    }  
    catch (Exception er)  
    {  
        Response.Write(er.Message);  
    }  
}
```

```
private void AddCarrera()  
{  
    Carrera Car = new Carrera();  
    Car.Nombre = Request["Nombre"];  
    Car.Duracion = int.Parse(Request["Duracion"]);  
    Car.Titulo = Request["Titulo"];  
    Car.Sigla = Request["Sigla"];  
    Car.Estado = Request["Estado"];  
    if (Request.Files.Count > 0)  
    {  
        for (int i = 0; i < Request.Files.Count; i++)  
            if (Request.Files[i].FileName.EndsWith("pdf")) Car.FUPdf = Request.Files[i];  
            else if (Request.Files[i].FileName != "") Car.Fu = Request.Files[i];  
    }  
    try  
    {  
        Car.Add();  
        Response.Write("OK");  
    } catch (Exception er)  
    {  
        Response.Write(er.Message);  
    }  
}
```

```
private void ListCarreras()  
{  
    try  
    {  
        Carrera c = new Carrera();  
        c.Estado = "ACTIVO";  
        Response.Write(c.List());  
    }  
    catch (Exception er)  
    {  
        Response.Write(er.Message);  
    }  
}
```

Page\_Load

```
case "RECOVERYPASSWORD": RecoveryPassword(); break;  
case "ADDCARRERA": AddCarrera(); break;  
case "LISTCARRERAS": ListCarreras(); break;  
case "MODIFICARCARRERA": ModifyCarrera(); break;  
}
```

# CAMBIO EN TABLES.JS MAKE TABLECAREERS()

```
const Td = function (parent)
{
    let td = $db.ce("td");
    $db.ac(parent, td);
    return td;
}; // obtengo un objeto td
```

```
this.makeTableCareers = function ()
{
    const FRow = function (tr, career)
    {
        Td(tr).innerText = career.Sigla.toUpperCase();
        Td(tr).innerText = career.Nombre.toUpperCase();
        Td(tr).innerHTML = "TÍTULO " +
            career.Titulo.toUpperCase() +
            "<br>DURACIÓN " + career.Duracion + " AÑOS";

        $dc.img(Td(tr), "", career.Url, "icon");
        MakeTd(tr, "#000", "yellow", "Plan<br>carrera " + career.Sigla, function ()
        {
            MakePlan(career);
        });
        MakeTd(tr, "#FFF", "#000", "Modificar<br>carrera " + career.Sigla, function () { $cf.modifyCareer(career); });
    };
    let lista = Post("accion=LISTCARRERAS");
    lista = JSON.parse(lista);
    $dt.makeTable(["sigla", "Nombre", "Datos", "imagen", "plan", "modificar"], 1, lista, FRow);
}; // lista laas carreras en el formulario alta carreras
```

```
const MakeTd = function (tr, color, backgroundColor, innerhtml, onclick)
{
    let td = Td(tr);
    td.style.backgroundColor = backgroundColor;
    td.style.color = color;
    td.onmouseover = function ()
    { this.style.opacity = 0.6; };
    td.onmouseleave = function ()
    { this.style.opacity = 1; };
    td.innerHTML = innerhtml.toUpperCase();
    td.onclick = onclick;
};
```



# CAMBIO EN TABLES.JS

```
const $$TablesCareers = function ()
{
  this.makeTableCareers = function (...); //lista laas carreras en el formulario alta carreras
  this.listCarreras = function ()
  {
    const arrayheader = ["sigla", "nombre", "descripcion", "Imagen", "plan de estudios"];
    const frow = function (tr, career)
    {
      Td(tr).innerText = career.Sigla.toUpperCase();
      Td(tr).innerText = career.Nombre.toUpperCase();
      Td(tr).innerHTML = "TÍTULO:" + career.Titulo.toUpperCase() +
        "<br/>DURACIÓN:" + career.Duracion + " AÑOS";
      //AGREGO IMAGEN
      $dc.img(Td(tr), "", career.Url, "maxicon");
      MakeTd(tr, "#000", "yellow", "Plan<br/>carrera " +
        career.Sigla, function () { MakePlan(career); });
    };
    //INICIO LA FUNCIÓN
    let list = JSON.parse(Post("accion=LISTCARRERAS"));
    if (list.length === 0) return;
    Section.innerHTML = "";
    $dt.makeTable(arrayheader, 5, list, frow);
  }; //lista las carreras en la section
};

$tc = new $$TablesCareers(); //INSTANCIA
```

```
const MakePlan = function (career)
{
  Section.innerHTML = "";
  //ContenedorTabla.style.display = "none";
  let ifr = $db.ce("iframe");
  $db.ac(Section, ifr);
  ifr.className = "iframepdf";
  ifr.id = "Iframe";
  ifr.src = career.UrlPDF;
};
```

[INICIO](#)[CARRERAS](#)[NOTICIAS](#)[INGRESAR AL SISTEMA](#)

SIGLA	NOMBRE	DESCRIPCION	IMAGEN	PLAN DE ESTUDIOS
TECAS	TECAS	TÍTULO:TECNICO DURACIÓN:3 AÑOS		PLAN CARRERA TECAS



**INSTITUTO  
EDUCATIVO  
BALLESTER**

**EXCELENCIA EN LA  
EDUCACIÓN**

AVDA MITRE 541  
v.BALLESTER BS.AS. TEL  
25433234 MAIL  
cballester@mail.com.ar





ESTE ES UN  
EJEMPLO DE PDF



**INSTITUTO  
EDUCATIVO  
BALLESTER**

**EXCELENCIA EN LA  
EDUCACIÓN**

AVDA MITRE 541  
v.BALLESTER BS.AS. TEL  
25433234 MAIL  
[cballester@mail.com.ar](mailto:cballester@mail.com.ar)

[INICIO](#)[PERFIL](#)[FUNCIONES](#)[ADMIN](#)

## NUEVA CARRERA



NOMBRE

SIGLA

DURACIÓN

TÍTULO

IMAGEN

[Examinar...](#)

PLAN DE ESTUDIOS

[Examinar...](#)

ESTADO

[AGREGAR CARRERA](#)[CANCELAR](#)

SIGLA	NOMBRE	DATOS	IMAGEN	PLAN	MODIFICAR
TECAS	TECAS	TÍTULO TECNICO DURACIÓN 3 AÑOS		PLAN CARRERA TECAS	MODIFICAR CARRERA TECAS



**INSTITUTO  
EDUCATIVO  
BALLESTER**

**EXCELENCIA EN LA  
EDUCACIÓN**

AVDA MITRE 541  
v.BALLESTER BS.AS. TEL  
25433234 MAIL  
cballester@mail.com.ar

**Ver**  
**Video**

FIN DEL CURSO  
FELICITACIONES