# Créate Database and SP

CREATE TABLES

|  |
| --- |
|  |

## CAR TABLE STORE PROCEDURE

|  |
| --- |
|  |

|  |
| --- |
| create proc InsertCar  @plate varchar(10),  @model varchar(4),  @branch varchar (50),  @owner\_id varchar(15)  as  begin  if not exists (select plate from Car where plate = @plate)  begin  insert into Car (plate, model, branch, owner\_id) values (@plate,@model,@branch,@owner\_id)  end  end |

Testing insertcar sp

|  |
| --- |
| DECLARE  @plate varchar(10),  @model varchar(4),  @branch varchar (50),  @owner\_id varchar(15)  EXEC InsertCar @plate = 'CF71E', @model = '2016', @branch = 'TVS' , @owner\_id = '88222558'  select \* from Car |

SP UpdateCar

|  |
| --- |
| create proc UpdateCar  @plate varchar(10),  @model varchar(4),  @branch varchar (50),  @owner\_id varchar(15)  as  begin  update car set model = @model, branch = @branch, owner\_id = @owner\_id  where plate = @plate  end |

Testing UpdateCar

|  |
| --- |
| exec UpdateCar @plate = 'CF71E'  ,@model = '2015'  ,@branch = 'TVS'  ,@owner\_id = '88222558'  select \* from Car |

SP DisplayCarbyPlate

|  |
| --- |
| CREATE PROC DisplayCarbyPlate  @plate varchar (10)  as  begin  select plate, model, branch, owner\_id from car  where plate = @plate  end |

Testing DisplayCarbyPlate

|  |
| --- |
| exec DisplayCarbyPlate @plate = 'CF71E' |

SP DisplayAllCars

|  |
| --- |
| CREATE PROC DisplayAllCars  as  begin  select plate, model, branch, owner\_id from car  end  --testing unmark  -- exec DisplayAllCars |

## OWNER\_CAR STORE PROCEDURE

|  |
| --- |
|  |

|  |
| --- |
| CREATE PROC InsertOwner  @id varchar (15),  @name varchar (50),  @lastname varchar (50)  as  begin  if not exists (select id from Owner\_Car where id = @id)  begin  insert into Owner\_Car (id, name, lastname) values (@id, @name, @lastname)  end  end |

TESTING

|  |
| --- |
| EXEC InsertOwner @id='88222558',@name='NELSON',@lastname='ORTIZ'  SELECT \* FROM Owner\_Car |

SP UpdateOwner

|  |
| --- |
| CREATE PROC UpdateOwner  @id varchar (15),  @name varchar (50),  @lastname varchar (50)  as  begin  if exists (select id from Owner\_Car where id = @id)  begin  Update Owner\_Car set name = @name, lastname = @lastname  where id = @id  end  end |

TESTING

|  |
| --- |
| EXEC UpdateOwner @id='88222557',@name='BOLIVAR',@lastname='FLOREZ'  SELECT \* FROM Owner\_Car |

SP DisplayOwnerByID

|  |
| --- |
| CREATE PROC DisplayOwnerByID  @id varchar (15)  as  begin  select Id,name, lastname from Owner\_Car  where id = @id  end |

TESTING

|  |
| --- |
| exec DisplayOwnerByID @id='88222558' |

SP DisplayAllOwners

|  |
| --- |
| CREATE PROC DisplayAllOwners  as  begin  select Id,name, lastname from Owner\_Car  end  --testing  --exec DisplayAllOwners |

SP DeleteOwner

|  |
| --- |
| CREATE PROC DeleteOwner  @id varchar (15)  as  begin  delete from Owner\_Car where id = @id  end |

TESTING DeleteOwner

|  |
| --- |
| exec DeleteOwner @id='88222556'  select \* from Owner\_Car |

## TAXES CAR STORE PROCEDURE

|  |
| --- |
|  |

SP InsertTaxes

|  |
| --- |
| CREATE PROC InsertTaxes  @plate varchar (10),  @year varchar (10),  @tax\_paid float,  @owner\_id varchar (15)  as  begin  if not exists (select plate, year, tax\_paid, owner\_id from Taxes where plate =@plate and year = @year)  begin  insert into Taxes (plate, year, tax\_paid, owner\_id) values (@plate,@year,@tax\_paid,@owner\_id)  end  end |

TESTING InsertTaxes

|  |
| --- |
| CREATE PROC InsertTaxes  @plate varchar (10),  @year varchar (10),  @tax\_paid float,  @owner\_id varchar (15)  as  begin  if not exists (select plate, year, tax\_paid, owner\_id from Taxes where plate =@plate and year = @year)  begin  insert into Taxes (plate, year, tax\_paid, owner\_id) values (@plate,@year,@tax\_paid,@owner\_id)  end  end |

TESTING InsertTaxes

|  |
| --- |
| EXEC InsertTaxes @plate='CF71E' ,@year='2021',@tax\_paid=58000.52,@owner\_id='88222558'  select \* from Taxes |

SP UpdateTaxes

|  |
| --- |
| ALTER PROC UpdateTaxes  @plate varchar (10),  @year varchar (10),  @tax\_paid float,  @owner\_id varchar (15)  as  begin  Update Taxes set tax\_paid=@tax\_paid, owner\_id=@owner\_id where plate=@plate and year =@year  end |

Testing UpdateTaxes

|  |
| --- |
| EXEC UpdateTaxes @plate='CF71E' ,@year='2020',@tax\_paid=30000,@owner\_id='88222558'  select \* from Taxes |

SP DisplayTaxesByPaid

|  |
| --- |
| CREATE PROC DisplayTaxesByPaid  @plate varchar (10),  @year varchar (4)  as  begin  select plate, year, tax\_paid, owner\_id from Taxes  where plate = @plate and year = @year  end |

TESTING SP DisplayTaxesByPaid

|  |
| --- |
| exec DisplayTaxesByPaid @plate='CF71E',@year='2018' |

SP DisplayAllTaxes

|  |
| --- |
| CREATE PROC DisplayAllTaxes  as  begin  select plate, year, tax\_paid, owner\_id from Taxes  end |

TESTING DisplayAllTaxes

|  |
| --- |
| exec DisplayAllTaxes |

CREATE A WCF PROJECT

|  |
| --- |
|  |

CREATE CLASS CAR

|  |
| --- |
|  |

|  |
| --- |
| using System.Runtime.Serialization;  using System.Web;  namespace WcfCars  {  [DataContract]  public class Car  {  [DataMember]  public string plate { get; set; }  [DataMember]  public string model { get; set; }  [DataMember]  public string branch { get; set; }  [DataMember]  public string owner\_id { get; set; }  }  } |

ADD A WCF CarWcf: two files are created

|  |
| --- |
|  |

Define the Interface and its methods like service

ServiceContract: Interface

OperationContract: for Methods

|  |
| --- |
|  |

## interface ICarWcf

|  |
| --- |
| using System;  using System.Collections.Generic;  using System.Linq;  using System.Runtime.Serialization;  using System.ServiceModel;  using System.Text;  namespace WcfCars  {  [ServiceContract]  public interface ICarWcf  {  [OperationContract]  List<Car> DisplayAllCars();  [OperationContract]  Car DisplayCarbyPlate(string plate);  [OperationContract]  String InsertCar(Car car);  [OperationContract]  String UpdateCar(Car car);  [OperationContract]  String DeleteCar(string plate);  }  } |

Implementacion de la Interfaz: class CarWcf : ICarWcf

|  |
| --- |
|  |

public string DeleteCar(string plate)

|  |
| --- |
|  |

|  |
| --- |
| public string DeleteCar(string plate)  {  String msg = string.Empty;  //2.  SqlCommand com = new SqlCommand("[DeleteCar]", con);  //.3  com.CommandType = CommandType.StoredProcedure;  try  {  //4.  com.Parameters.AddWithValue("@plate",plate);  con.Open();  int n = com.ExecuteNonQuery();  if (n > 0)  {  msg = "success";  }  else  {  msg = "Plate not exists";  }  con.Close();  }//  catch (Exception ex)  {  msg = ex.Message;  }//catch  return msg;  }//End DeleteCar |

|  |
| --- |
| public List<Car> DisplayAllCars()  {  List<Car> lstCars = new List<Car>();  //2.  SqlCommand com = new SqlCommand("[DisplayAllCars]",con);  //3  com.CommandType = CommandType.StoredProcedure;  //4.  DataTable dt = new DataTable();  //5.  SqlDataAdapter da = new SqlDataAdapter(com);  try  {  da.Fill(dt);  if (dt.Rows.Count > 0)  {  foreach (DataRow eachrow in dt.Rows)  {  Car car = new Car();  car.plate = eachrow["plate"].ToString();  car.model = eachrow["model"].ToString();  car.branch = eachrow["branch"].ToString();  car.owner\_id = eachrow["owner\_id"].ToString();  lstCars.Add(car);  }//foreach  }//endif  }  catch (Exception ex)  {  String err = ex.Message;  }//catch  return lstCars;    }//End DisplayAllCars |

|  |
| --- |
| public Car DisplayCarbyPlate(string plate)  {  Car car = new Car();  //2.  SqlCommand com = new SqlCommand("DisplayCarbyPlate",con);  //3  com.CommandType = CommandType.StoredProcedure;  //4. Datatable  DataTable dt = new DataTable();  //5 SqlAdapter  SqlDataAdapter da = new SqlDataAdapter(com);  try  {  //.6 Parameters  com.Parameters.AddWithValue("@plate",plate);  da.Fill(dt);  if (dt.Rows.Count > 0)  {  foreach (DataRow eachrow in dt.Rows)  {  car.plate = dt.Rows[0]["plate"].ToString();  car.model = dt.Rows[0]["model"].ToString();  car.branch = dt.Rows[0]["branch"].ToString();  car.owner\_id = dt.Rows[0]["owner\_id"].ToString();  }//end foreach  }//end if  }  catch (Exception ex)  {  String err = ex.Message;  }  return car;  }//end DisplayCarbyPlate |

|  |
| --- |
| public string InsertCar(Car car)  {  String msg = string.Empty;  //2. SqlCommand  SqlCommand com = new SqlCommand("InsertCar",con);  //3. CommandType  com.CommandType = CommandType.StoredProcedure;  try  {  //4 Parameters  com.Parameters.AddWithValue("@plate",car.plate);  com.Parameters.AddWithValue("@model",car.model);  com.Parameters.AddWithValue("@branch",car.branch);  com.Parameters.AddWithValue("owner\_id",car.owner\_id);  con.Open();  int n = com.ExecuteNonQuery();  if (n > 0)  {  msg = "success";  string msgstatusemail = string.Empty;  string fromemail = "nelsonortiz2712@gmail.com";  string toemail = "nelsonortiz1227@gmail.com";  string subject = "Car saved";  string msgbody = "Registration plate successfull: " + car.plate;  msgstatusemail = SendEmail(fromemail, toemail, subject, msgbody);  }  }  catch (Exception ex)  {  msg = ex.Message;  }//try  return msg;  }//InsertCar |

|  |
| --- |
| public string UpdateCar(Car car)  {  String msg = string.Empty;  //2.  SqlCommand com = new SqlCommand("UpdateCar",con);  //3.  com.CommandType = CommandType.StoredProcedure;  try  {  //4  com.Parameters.AddWithValue("@plate",car.plate);  com.Parameters.AddWithValue("@model",car.model);  com.Parameters.AddWithValue("@branch",car.branch);  com.Parameters.AddWithValue("@owner\_id",car.owner\_id);  con.Open();  int n = com.ExecuteNonQuery();  if (n > 0)  {  msg = "success";  }  else  {  msg = "Record not updated, check the plate number";  }  }  catch (Exception ex)  {  msg = ex.Message;  }  return msg;  }//UpdateCar |

|  |
| --- |
| public string SendEmail(string from, string to, string subject, string msgBody)  {  string msg = "Nothing";  try  {  MailMessage message = new MailMessage(from, to);  SmtpClient smtp = new SmtpClient();  message.Subject = subject;  message.IsBodyHtml = false;  message.Body = msgBody;  smtp.Port = 587; // Free of cost google service port number (max 25 email per 1 Gmail ID they allow)  smtp.Host = "smtp.gmail.com"; //Free of cost google service for gmail host  smtp.EnableSsl = true;  smtp.UseDefaultCredentials = false;  smtp.Credentials = new NetworkCredential("nelsonortiz2712@gmail.com", "xxxxx...");  smtp.DeliveryMethod = SmtpDeliveryMethod.Network;  smtp.Send(message);  msg = "Password Sent to Registered Email";  }  catch (Exception ex)  {  msgBody = ex.Message;  }  return msg;  }  //send Email |

WEBCONFIG

|  |
| --- |
| <appSettings>  <add key="aspnet:UseTaskFriendlySynchronizationContext" value="true"/>  <add key="constring" value="server=COCBRANW10P;database=sample;integrated security=true"/>  </appSettings> |

## DEBUG THE SERVICE

|  |
| --- |
|  |

# CREATE CONSUMER WCF

Select MVC

|  |
| --- |
|  |

|  |
| --- |
|  |

Firs debug the WCF

|  |
| --- |
|  |

Now, Add reference to the WCF

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

|  |
| --- |
|  |

Add a new controller

|  |
| --- |
|  |

Controller: DisplayAllCars

|  |
| --- |
|  |

|  |
| --- |
| using System;  using System.Collections.Generic;  using System.Linq;  using System.Web;  using System.Web.Mvc;  namespace WcfCar\_Consumer.Controllers  {  public class WcfCarController : Controller  {  // GET: WcfCar  public ActionResult DisplayAllCars()  {  WcfCar.CarWcfClient obj = new WcfCar.CarWcfClient();  List<WcfCar.Car> lst = obj.DisplayAllCars().ToList();  return View(lst);  }  }  } |

Add a View

|  |
| --- |
|  |

View DisplayAlCars code

|  |
| --- |
|  |

|  |
| --- |
| @model IEnumerable<WcfCar\_Consumer.WcfCar.Car>  @{  Layout = null;  }  <!DOCTYPE html>  <html>  <head>  <meta name="viewport" content="width=device-width" />  <title>Index</title>  </head>  <body>  <div>  <table>  <tr>  <th>Plate</th>  <th>Model</th>  <th>Branch</th>  <th>Owner</th>  </tr>  @foreach (var item in Model)  {  <tr>  <td>@Html.DisplayFor(model => item.plate)</td>  <td>@Html.DisplayFor(model => item.model)</td>  <td>@Html.DisplayFor(model => item.branch)</td>  <td>@Html.DisplayFor(model => item.owner\_id)</td>  </tr>  }  </table>  </div>  </body>  </html> |

Run the View

|  |
| --- |
|  |

Controller: DisplayCarbyPlate

|  |
| --- |
|  |

|  |
| --- |
| public ActionResult DisplayCarbyPlate(string plate)  {  WcfCar.Car car = new WcfCar.Car();  // WcfCar.CarWcfClient obj = new WcfCar.CarWcfClient();  car = obj.DisplayCarbyPlate(plate);  return View(car);  } |

View DisplayCarbyPlate

|  |
| --- |
| @model WcfCar\_Consumer.WcfCar.Car  @{  Layout = null;  }  <!DOCTYPE html>  <html>  <head>  <meta name="viewport" content="width=device-width" />  <title>DisplayCarbyPlate</title>  </head>  <body>  @using (Html.BeginForm())  {  <div class="form-horizontal">  <div class="form-group">  Plate:  <br />  <input type="text" id="plate" name="plate">  </div>  <div class="form-group">    <br />  <input type="submit" value="DisplayCarbyPlate">  </div>  </div>  <br />  <h3>LICENSE PLATE</h3>  <table>  <tr>  <th>Plate</th>  <th>Model</th>  <th>Branch</th>  <th>Owner</th>  </tr>  <tr>  <td>@Html.DisplayFor(model => model.plate)</td>  <td>@Html.DisplayFor(model => model.model)</td>  <td>@Html.DisplayFor(model => model.branch)</td>  <td>@Html.DisplayFor(model => model.owner\_id)</td>  </tr>  </table>  }  </body>  </html> |

|  |
| --- |
|  |