**.Net Mvc With Out Entity FrameWork.**

* .Net Mvc With Out The Entity Frame Work are directly connected to data base.
* Data will Get And Store by the sql Query.

Sql DataAdapter:-

* Sql DataAdapter are use for the conjunction with sql connection.
* Sql DataAdapter are used in the get data of sql server.

SqlCommand:-

* Sql Command are used in the edit and post method.
* And this are increase the performance when connecting to a Sql Server Database.

DataTable:-

* DataTable Are used for the store table.
* That actually work like the DataBase.

DataSet:-

* Fcgvb

StoreProcedure:-

* The store procedure are used to store query.
* They have access the in data base and when it create the store procedure are build. So that they Use are increase the performance.

**Code For Get Method:-**

List<Inquiry> objInquiryList = new List<Inquiry>(); // list mate

**DataTable dtbl = new DataTable(); // without ef mate**

**using (SqlConnection sqlConnection = new SqlConnection(\_configuration.GetConnectionString("defaultconnection")))**

**{**

**sqlConnection.Open();**

string str = "select C.\*,A.[CountryName] As CountryName,B.[StateName] As StateName,C.[CityName] As CityName,D.[InquirySourceName] As InquirySourceName,E.[InquiryStatusName] As InquiryStatusName,F.[Name] As Name from Inquiries G INNER JOIN Countries A ON G.CountryId = A.Id INNER JOIN States B ON G.StateId = B.Id INNER JOIN Cities C ON G.CityId = C.Id INNER JOIN InquirySources D ON G.InquirySourceId = D.Id INNER JOIN InquiryStatuses E ON G.InquiryStatusId = E.Id INNER JOIN Products F ON G.ProductId = F.Id";

**SqlDataAdapter sqlDa = new SqlDataAdapter(str, sqlConnection);**

**sqlDa.SelectCommand.CommandType = CommandType.Text;**

**sqlDa.Fill(dtbl);**

for (int i = 0; i < dtbl.Rows.Count; i++)

{

Inquiry inquiry = new Inquiry();

inquiry.Id = Convert.ToInt32(dtbl.Rows[i]["Id"]);

inquiry.Organization = Convert.ToString(dtbl.Rows[i]["Organization"];

inquiry.CountryId = Convert.ToInt32(dtbl.Rows[i]["CountryId"]);

inquiry.Country = new Country

{

Id = Convert.ToInt32(dtbl.Rows[i]["CountryId"]),

CountryName = Convert.ToString(dtbl.Rows[i]["CountryName"])

};

objInquiryList.Add(inquiry);

}

}

* First you need to add data table for the store table.
* After that you have to instablize sql connection.
* And After that the you need to open the sql connection.
* Sql Adddpter you must have to describe name of string.=> **(str, sqlConnection)**
* If you have to use store procedure so that use name of store procedure.
* **sqlDa.SelectCommand.CommandType = CommandType.Text; =>** In this line you must describe the type of command .
* If you use sting in controller you must add text and if you have use the store procedure you must be add Store Procedure.
* In the index method if you have the foreign key you must the define the object and get the the name.
* And convert the data in suitable data type in right hand side.

**Code For Post Method:-**

using (SqlConnection sqlConnection = new SqlConnection(\_configuration.GetConnectionString("defaultconnection")))

{

**sqlConnection.Open();**

**SqlCommand sqlCmd = new SqlCommand("Upsert", sqlConnection);**

sqlCmd.CommandType = CommandType.StoredProcedure;

**sqlCmd.Parameters.AddWithValue("Id", inquiryVM.Inquiry.Id);**

sqlCmd.Parameters.AddWithValue("Organization", inquiryVM.Inquiry.Organization);

sqlCmd.Parameters.AddWithValue("InquirySourceId", inquiryVM.Inquiry.InquirySourceId);

sqlCmd.Parameters.AddWithValue("InquiryStatusId", inquiryVM.Inquiry.InquiryStatusId);

**sqlCmd.ExecuteNonQuery();**

* In post method you must add store procedure name.
* And After that You Add the below Line For store data in database.
* => **sqlCmd.Parameters.AddWithValue("Id", inquiryVM.Inquiry.Id);**
* **sqlCmd.ExecuteNonQuery(); =>** This Line are used in the the execute the query in the database.

**Code for the Check The Duplicate Value in Handle the data base error:-.**

Controller:-

**try**

**{**

**sqlCmd.ExecuteNonQuery();**

**}**

**catch (SqlException ex)**

**{**

**if (ex.Number == 50000) // Custom error number used in the stored procedure RAISERROR statements**

**{**

**ModelState.AddModelError("", ex.Message);**

**TempData["error"] = "inquiry Name Already Exist!";**

**return View(inquiryVM);**

**}**

**else**

**{**

**throw; // Rethrow other unexpected SQL exceptions**

**}**

**}**

**}**

**DataBase:-**

**IF NOT EXISTS (SELECT \* FROM Brands WHERE BrandName = @BrandName AND Id <> @Id)**

BEGIN

UPDATE Brands

SET

BrandName = @BrandName,

BrandImage = @BrandImage

WHERE Id = @Id

END

ELSE

BEGIN

**RAISERROR('Brand with the same name already exists for a different brand.', 16, 1);**

END

* RAISEERROR :- Are used in the the throw error in controller.
* Ex.Error :- Can tack the error display which can throw by the sqlserver.
* And Above Line are used in the checck duplicate value and throw the error.

**Handle Null Value In DataBase And Model:-**

**Get Method( Data Come From The Data Base):-**

* **This Method Are Used In When The Data will Come from The Database.**

**Method:- 1**

if (dtbl.Rows[i]["UpdatedDate"] != DBNull.Value)

{

product.UpdatedDate = Convert.ToDateTime(dtbl.Rows[i]["UpdatedDate"]);

}

else

{

product.UpdatedDate = null;

}

**Method:- 2 :- (This Method are worked When The Above Not worked)**

if (dtbl.Rows[i]["UpdatedDate"] != DBNull.Value)

{

if (DateTime.TryParse(dtbl.Rows[i]["UpdatedDate"].ToString(), out DateTime updatedDate))

{

product.UpdatedDate = updatedDate;

}

else

{

// Handle the case when the value in the DataTable is not a valid DateTime format.

// You might choose to set a default value or take appropriate action based on your requirements.

}

}

else

{

Handle the case when dtbl.Rows[i]["UpdatedDate"] is DBNull.Value (null).

You might choose to set product.UpdatedDate to null or take appropriate action based on your requirements.

product.UpdatedDate = null; // Assuming the UpdatedDate property is nullable (DateTime?).

}

**Post Method(Data Push In the Controller.):-**

* **This Method Are Used In When The Data will put in the database.**

if (productVM.Product.UpdatedDate != null)

{

sqlCmd.Parameters.AddWithValue("UpdatedDate", productVM.Product.UpdatedDate);

}

else

{

sqlCmd.Parameters.AddWithValue("UpdatedDate", DBNull.Value);

}

**StoreProcedure Of Index:-**

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

ALTER PROCEDURE [dbo].[BrandIndex]

AS

BEGIN

-- SET NOCOUNT ON added to prevent extra result sets from

-- interfering with SELECT statements.

SET NOCOUNT ON;

-- Insert statements for procedure here

select \* from Brands

END

**StoreProcedure Of Get Item By Id:-**

USE [ProductManagment\_WOEF]

GO

/\*\*\*\*\*\* Object: StoredProcedure [dbo].[BrandById] Script Date: 7/22/2023 12:17:35 PM \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE PROCEDURE [dbo].[BrandById]

@Id int

AS

BEGIN

SET NOCOUNT ON;

SELECT \*

FROM Brands

WHERE Id=@Id

END

GO

**StoreProcedure Of Delete:-**

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE PROCEDURE [dbo].[BrandDeleteById]

@Id int

AS

BEGIN

SET NOCOUNT ON;

DELETE Brands

WHERE Id= @Id

END

GO

**StoreProcedure Of Upsert:-**

USE [ProductManagment\_WOEF]

GO

/\*\*\*\*\*\* Object: StoredProcedure [dbo].[Upsert] Script Date: 7/22/2023 12:40:47 PM \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

ALTER PROCEDURE [dbo].[Upsert]

@Id int,

@BrandName nvarchar(50),

@BrandImage nvarchar(450)

AS

BEGIN

SET NOCOUNT ON;

IF @Id = 0

BEGIN

-- Check if the BrandName already exists

IF NOT EXISTS (SELECT \* FROM Brands WHERE BrandName = @BrandName)

BEGIN

INSERT INTO Brands(BrandName, BrandImage)

VALUES(@BrandName, @BrandImage)

END

ELSE

BEGIN

-- Handle the case when the BrandName already exists

RAISERROR('Brand with the same name already exists.', 16, 1);

END

END

ELSE

BEGIN

-- Check if the BrandName already exists for a different brand (when performing an update)

IF NOT EXISTS (SELECT \* FROM Brands WHERE BrandName = @BrandName AND Id <> @Id)

BEGIN

UPDATE Brands

SET

BrandName = @BrandName,

BrandImage = @BrandImage

WHERE Id = @Id

END

ELSE

BEGIN

-- Handle the case when the BrandName already exists for a different brand

RAISERROR('Brand with the same name already exists for a different brand.', 16, 1);

END

END

END