**Kuntal Das**

Enrollment No.: 170103645,

Cource : BCA

Final Project

Online Courier Management Service

**September 18, 2019**

# Overview

Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diam nonummy nibh euismod tincidunt ut laoreet dolore magna aliquam erat volutpat. Ut wisi enim ad minim veniam, quis nostrud exerci tation ullamcorper suscipit lobortis nisl ut aliquip ex ea commodo consequat.

Duis autem vel eum iriure dolor in hendrerit in vulputate velit esse molestie consequat, vel illum dolore eu feugiat nulla facilisis at vero eros et accumsan.

# Goals

1. **Lorem ipsum dolor sit amet:** Duis autem vel eum iriure dolor in hendrerit in vulputate velit esse molestie consequat, vel illum dolore eu feugiat nulla facilisis at vero eros et accumsan.
2. **Sed diam nonummy nibh euismod:** Nam liber tempor cum soluta nobis eleifend option congue nihil imperdiet doming id quod mazim placerat facer possim assum.

# Specifications

Nam liber tempor cum soluta nobis eleifend option congue nihil imperdiet doming id quod mazim placerat facer possim assum. Typi non habent claritatem insitam; est usus legentis in iis qui facit eorum claritatem. Investigationes demonstraverunt lectores legere me lius quod ii legunt saepius.

# Milestones

## Lorem ipsum

Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diam nonummy nibh euismod tincidunt ut laoreet dolore magna aliquam erat volutpat.

## Dolor sit amet

Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diam nonummy nibh euismod tincidunt ut laoreet dolore magna aliquam erat volutpat.

## Consectetur adipiscing elit

Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diam nonummy nibh euismod tincidunt ut laoreet dolore magna aliquam erat volutpat.

**Project Codes**

1. **Stored Procedures(SQL Server)**

## Insert Customer

1. CREATE PROCEDURE [dbo].[InsertCustomer]
2. @NAME NVARCHAR(300), @ADDRESS NVARCHAR(300), @GENDER NVARCHAR(20),
3. @DOB DATE, @PHNO NUMERIC(15,0), @EMAIL NVARCHAR(100),
4. @BID NUMERIC(10,0), @ACCTYPE NVARCHAR(30), @STATUS NVARCHAR(30),
5. @PASS NVARCHAR(MAX)
6. AS
7. BEGIN
8. BEGIN TRANSACTION
9. INSERT INTO
10. Customer(Name,Address,Gender,DOB,Phno,Email,BID,AccType,Status)
11. VALUES(@NAME,@ADDRESS,@GENDER,@DOB,@PHNO,@EMAIL,@BID,@ACCTYPE,@STATUS);
12. DECLARE @CID NVARCHAR(20)=(SELECT CID FROM Customer WHERE Email=@EMAIL);
13. INSERT INTO Cpass
14. VALUES(@CID,@PASS);
15. COMMIT;
16. SELECT @CID AS CID;
17. END

## Update Customer

1. CREATE PROCEDURE [dbo].[UpdateCustomer]
2. @OLD\_CID NVARCHAR(15), @OLD\_NAME NVARCHAR(300), @OLD\_ADDR NVARCHAR(300),
3. @OLD\_GENDER NVARCHAR(20), @OLD\_DOB DATE, @OLD\_PHNO NUMERIC(15,0),
4. @OLD\_EMAIL NVARCHAR(100), @OLD\_BID NUMERIC(10,0),
5. @OLD\_ACCTYPE NVARCHAR(30), @OLD\_STATUS NVARCHAR(30),
6. @NAME NVARCHAR(300), @ADDR NVARCHAR(300),
7. @GENDER NVARCHAR(20), @DOB DATE, @PHNO NUMERIC(15,0),
8. @EMAIL NVARCHAR(100), @BID NUMERIC(10,0),
9. @ACCTYPE NVARCHAR(30), @STATUS NVARCHAR(30)
10. AS
11. BEGIN
12. UPDATE Customer
13. SET Name=@NAME, Address=@ADDR, Gender=@GENDER, DOB=@DOB,
14. Phno=@PHNO, Email=@EMAIL, BID=@BID, AccType=@ACCTYPE,
15. Status=@STATUS
16. WHERE
17. CID=@OLD\_CID AND Name=@OLD\_NAME AND
18. Address=@OLD\_ADDR AND Gender=@OLD\_GENDER AND
19. DOB=@OLD\_DOB AND Phno=@OLD\_PHNO AND
20. Email=@OLD\_EMAIL AND BID=@OLD\_BID AND
21. AccType=@OLD\_ACCTYPE AND Status=@OLD\_STATUS;
23. END

## Get Customer Information

1. CREATE PROCEDURE [dbo].[GetCusInfo]
2. @CID NVARCHAR(20)
3. AS
4. BEGIN
5. SET NOCOUNT ON;
6. IF((SELECT COUNT(\*) FROM Cimage WHERE CID=@CID)=1)
7. BEGIN
8. SELECT
9. CU.CID AS CID, CU.Name AS NAME, CU.Address AS ADDR,
10. CU.Gender AS GENDER, CU.DOB AS DOB, CU.Phno AS PHNO,
11. CU.Email AS EMAIL, CU.BID AS BID, CU.AccType AS ACCTYPE,
12. CU.Status AS STATUS, CI.ImageData AS IMGDATA
13. FROM Customer CU JOIN Cimage CI ON CU.CID=CI.CID
14. WHERE CU.CID=@CID;
15. END
16. ELSE
17. BEGIN
18. SELECT
19. CID AS CID, Name AS NAME, Address AS ADDR,
20. Gender AS GENDER, DOB AS DOB, Phno AS PHNO,
21. Email AS EMAIL, BID AS BID, AccType AS ACCTYPE,
22. Status AS STATUS, NULL AS IMGDATA
23. FROM Customer
24. WHERE CID=@CID;
25. END
26. END

## Get Customer ID by Password

1. CREATE PROCEDURE [dbo].[GetCIDbyPass]
2. @PHNO NUMERIC(15,0) = NULL,
3. @EMAIL NVARCHAR(100) = NULL,
4. @PASS NVARCHAR(MAX)
5. AS
6. BEGIN
7. SET NOCOUNT ON;
8. IF(@PHNO IS NOT NULL AND @EMAIL IS NOT NULL)
9. BEGIN
10. SELECT CU.CID
11. FROM Customer CU JOIN Cpass CP ON CU.CID=CP.CID
12. WHERE Phno=@PHNO AND Email=@EMAIL AND CP.password=@PASS;
13. END
14. ELSE IF(@PHNO IS NOT NULL)
15. BEGIN
16. SELECT CU.CID
17. FROM Customer CU JOIN Cpass CP ON CU.CID=CP.CID
18. WHERE Phno=@PHNO AND CP.password=@PASS;
19. END
20. ELSE IF(@EMAIL IS NOT NULL)
21. BEGIN
22. SELECT CU.CID
23. FROM Customer CU JOIN Cpass CP ON CU.CID=CP.CID
24. WHERE Email=@EMAIL AND CP.password=@PASS;
25. END
26. END

## Update Customer Password

1. CREATE PROCEDURE [dbo].[UpdateCPass]
2. @OLD\_CID NVARCHAR(20),
3. @OLD\_PASS NVARCHAR(MAX),
4. @PASS NVARCHAR(MAX)
5. AS
6. BEGIN
7. UPDATE Cpass
8. SET password=@PASS
9. WHERE CID=@OLD\_CID AND password=@OLD\_PASS;
10. END

## Insert Update Customer Image

1. CREATE PROCEDURE [dbo].[InsertUpdateCImage]
2. @CID NVARCHAR(20),
3. @NAME NVARCHAR(255),
4. @SIZE INT,
5. @IMGDATA VARBINARY(MAX)
6. AS
7. BEGIN
8. IF((SELECT COUNT(\*) FROM Cimage WHERE CID=@CID)=0)
9. BEGIN
10. INSERT INTO Cimage VALUES(@CID, @NAME, @SIZE, @IMGDATA);
11. END
12. ELSE
13. BEGIN
14. UPDATE Cimage SET Name=@NAME,Size=@SIZE,ImageData=@IMGDATA WHERE CID=@CID;
15. END
16. END

## Request Reset Customer Password

1. CREATE proc [dbo].[CRequestResetPassword]
2. @Email nvarchar(100)
3. as
4. Begin
5. Declare @CID NVARCHAR(20);
6. Declare @USER nvarchar(100);
7. Select @CID = CID , @USER = Name
8. from Customer
9. where Email = @Email;
10. if(@CID IS NOT NULL)
11. Begin
12. --If username exists
13. Declare @GUID UniqueIdentifier;
14. Set @GUID = NEWID();
15. Insert into CResetPasswordRequests(Id, CID, ResetRequestDateTime)
16. Values(@GUID, @CID, GETDATE())
17. Select 1 as ReturnCode, @GUID as UniqueId, @USER as Name
18. End
19. Else
20. Begin
21. --If username does not exist
22. SELECT 0 as ReturnCode, NULL as UniqueId, NULL as Name
23. End
24. End

## Is Customer Rest Password Link Valid

1. CREATE Proc [dbo].[CIsPasswordResetLinkValid]
2. @GUID uniqueidentifier
3. as
4. Begin
5. SELECT CID from CResetPasswordRequests where Id=@GUID;
6. End

## Reset Customer Password

1. CREATE Proc [dbo].[CResetPassword]
2. @GUID uniqueidentifier,
3. @Password nvarchar(100)
4. as
5. Begin
6. Declare @CID NVARCHAR(20)
7. Select @CID = CID
8. from CResetPasswordRequests
9. where Id= @GUID;
10. if(@CID is null)
11. Begin

--If UserId does not exist

1. Select 0 as IsPasswordChanged
2. End
3. Else
4. Begin

--If UserId exists, Update with new password

1. Update Cpass set Password=@Password where CID=@CID;

--Delete the password reset request row

1. Delete from CResetPasswordRequests where CID = @CID;
2. Select 1 as IsPasswordChanged
3. End
4. End

## Insert Employee

1. CREATE PROCEDURE [dbo].[InsertInfoByEid]
2. @BID NUMERIC(10,0) = NULL, @NAME NVARCHAR(50) = NULL,
3. @DOB DATE = NULL, @GENDER NVARCHAR(10) = NULL,
4. @ADDRESS NVARCHAR(300) = NULL, @EMAIL NVARCHAR(50) = NULL,
5. @PHNO NUMERIC(15,0) = NULL, @POST INT = NULL,
6. @PASS NVARCHAR(MAX) = NULL, @ACTIVE INT = 1,
7. @EID NUMERIC(10,0) OUTPUT
8. AS
9. BEGIN
10. SET NOCOUNT ON;
11. BEGIN TRANSACTION
12. INSERT INTO employee(BID,Ename,Edob,Egender,Eaddress,Eemail,Ephno,Epost)
13. VALUES(@BID,@NAME,@DOB,@GENDER,@ADDRESS,@EMAIL,@PHNO,@POST);
14. SET @EID = SCOPE\_IDENTITY();
15. IF(@PASS IS NOT NULL)
16. BEGIN
17. INSERT INTO epass VALUES(@EID,@PASS);
18. END
19. IF(@ACTIVE IS NOT NULL)
20. BEGIN
21. INSERT INTO activeE VALUES(@EID,@ACTIVE);
22. END
23. COMMIT;
24. END

## Insert Employee Image

1. CREATE PROCEDURE [dbo].[InsertImagebyEid]
2. @EID NUMERIC(10,0),
3. @NAME NVARCHAR(255),
4. @SIZE INT,
5. @IMGDATA VARBINARY(MAX)
6. AS
7. BEGIN
8. INSERT INTO Eimages VALUES(@EID, @NAME, @SIZE, @IMGDATA);
9. END

## Update Employee Image

1. CREATE PROCEDURE [dbo].[UpdateImagebyEid]
2. @EID NUMERIC(10,0),
3. @NAME NVARCHAR(255),
4. @SIZE INT,
5. @IMGDATA VARBINARY(MAX)
6. AS
7. BEGIN
8. UPDATE Eimages
9. SET Name = @NAME, Size = @SIZE, ImageData = @IMGDATA
10. WHERE EID = @EID;
11. END

## Update Employee

1. CREATE PROCEDURE [dbo].[UpdateInfoByEid]
2. @EID NUMERIC(10,0) = NULL, @BID NUMERIC(10,0) = NULL,
3. @NAME NVARCHAR(50) = NULL, @DOB DATE = NULL,
4. @GENDER NVARCHAR(10) = NULL, @ADDRESS NVARCHAR(300) = NULL,
5. @EMAIL NVARCHAR(50) = NULL, @PHNO NUMERIC(15,0) = NULL,
6. @POST INT = NULL, @PASS NVARCHAR(MAX) = NULL, @ACTIVE INT = 1
7. AS
8. BEGIN
9. BEGIN TRANSACTION
10. UPDATE employee
11. SET BID = @BID, Ename = @NAME,
12. Edob=@DOB, Egender=@GENDER,
13. Eaddress=@ADDRESS, Eemail=@EMAIL,
14. Ephno=@PHNO
15. WHERE EID=@EID;
16. IF(@ACTIVE IS NOT NULL)
17. BEGIN
18. UPDATE activeE SET Active=@ACTIVE WHERE EID=@EID;
19. END
20. IF(@PASS IS NOT NULL)
21. BEGIN
22. UPDATE epass SET Epassword = @PASS WHERE EID = @EID;
23. END
24. COMMIT
25. END

## Get Employee Info

1. CREATE PROCEDURE [dbo].[GetEmpByEid]
2. @EID NUMERIC(10,0)
3. AS
4. BEGIN
5. SELECT
6. EM.EID AS EID,EM.Ename AS NAME,EM.Edob AS DOB,
7. EM.Egender AS GENDER,BR.BID AS BID,BR.RID AS RID,
8. DE.DeptID AS POSTINDEX,DE.DeptName AS POST,
9. AE.Active AS STATUS
10. FROM employee EM
11. JOIN Dept DE ON Epost=DeptID JOIN Branch BR ON EM.BID=BR.BID
12. JOIN activeE AE ON AE.EID=EM.EID
13. WHERE EM.EID=@EID;
14. END

## Get Employee Image

1. CREATE PROCEDURE [dbo].[GetimageByEID]
2. @EID NUMERIC(10,0)
3. AS
4. BEGIN
5. IF((SELECT COUNT(EID) FROM Eimages WHERE EID=@EID)=1)
6. BEGIN
7. SELECT ImageData from Eimages where EID=@EID;
8. END
9. ELSE
10. BEGIN
11. SELECT NULL AS ImageData;
12. END
13. END

## Get Employee ID by Password

1. CREATE PROCEDURE [dbo].[GetEidByPass]
2. @EMAIL NVARCHAR(50) = NULL,
3. @PHNO NUMERIC(10,0) = NULL,
4. @PASS NVARCHAR(50)
5. AS
6. BEGIN
7. IF (@PHNO IS NULL)
8. BEGIN
9. SELECT E.EID FROM employee E JOIN epass P on E.EID= P.EID
10. WHERE P.Epassword=@PASS AND E.Eemail=@EMAIL;
11. END
12. ELSE IF(@EMAIL IS NULL)
13. BEGIN
14. SELECT E.EID FROM employee E JOIN epass P on E.EID= P.EID
15. WHERE P.Epassword=@PASS AND E.Ephno=@PHNO;
16. END
17. ELSE
18. BEGIN
19. SELECT E.EID FROM employee E JOIN epass P on E.EID= P.EID
20. WHERE P.Epassword=@PASS AND E.Ephno=@PHNO AND E.Eemail=@EMAIL;
21. END
22. END

## Request Reset Employee Password

1. CREATE proc [dbo].[ERequestResetPassword]
2. @Email nvarchar(100)
3. as
4. Begin
5. Declare @EID NVARCHAR(20);
6. Declare @USER nvarchar(100);
7. Select @EID = EID , @USER = Ename from employee where Eemail = @Email;
8. if(@EID IS NOT NULL)
9. Begin
10. --If username exists
11. Declare @GUID UniqueIdentifier;
12. Set @GUID = NEWID();
13. Insert into EResetPasswordRequests(Id, EID, ResetRequestDateTime)
14. Values(@GUID, @EID, GETDATE())
15. Select 1 as ReturnCode, @GUID as UniqueId, @USER as Name;
16. End
17. Else
18. Begin
19. --If username does not exist
20. SELECT 0 as ReturnCode, NULL as UniqueId, NULL as Name;
21. End
22. End

## Is Employee Rest Password Link Valid14

1. CREATE Proc [dbo].[EIsPasswordResetLinkValid]
2. @GUID uniqueidentifier
3. as
4. Begin
5. SELECT EID from EResetPasswordRequests where Id=@GUID;
6. End

## Employee Password Reset15

1. CREATE Proc [dbo].[EResetPassword]
2. @GUID uniqueidentifier, @Password nvarchar(100)
3. as
4. Begin
5. Declare @EID NUMERIC(10,0)
6. Select @EID = EID from EResetPasswordRequests where Id= @GUID;
7. if(@EID is null)
8. Begin
9. Select 0 as IsPasswordChanged;
10. End
11. Else
12. Begin

Update epass set Epassword=@Password where EID=@EID;

Delete from EResetPasswordRequests where EID = @EID;

1. Select 1 as IsPasswordChanged;
2. End
3. End

## Update Employee Info By Admin

1. CREATE PROCEDURE [dbo].[UpdateEmpByAdmin]
2. @OLD\_EID NUMERIC(10,0), @OLD\_BID NUMERIC(10,0),
3. @OLD\_POSTINDEX INT, @OLD\_STATUS NVARCHAR(50),
4. @BID NUMERIC(10,0), @POSTINDEX INT,
5. @STATUS NVARCHAR(50)
6. AS
7. BEGIN
8. DECLARE @err\_message nvarchar(255);
9. DECLARE @C INT;
10. SET @C = (SELECT COUNT(\*) FROM employee E JOIN activeE A ON E.EID=A.EID
11. WHERE E.EID=@OLD\_EID AND [E.BID=@OLD\_BID](mailto:E.BID=@OLD_BID)
12. AND [E.Epost=@OLD\_POSTINDEX](mailto:E.Epost=@OLD_POSTINDEX) AND A.Active=@OLD\_STATUS);
13. IF(@C<>1)
14. BEGIN
15. RETURN -1;
16. END
17. ELSE
18. BEGIN
19. BEGIN TRANSACTION
20. UPDATE employee SET BID=@BId, Epost=@POSTINDEX WHERE EID=@OLD\_EID;
21. UPDATE activeE SET Active=@STATUS WHERE EID=@OLD\_EID;
22. COMMIT;
23. END
24. END

## Insert Regional Branch

1. CREATE PROCEDURE [dbo].[InsertRegB]
2. @REGID INT, @REGNAME NVARCHAR(300),@REGADDR NVARCHAR(300),
3. @LAT NUMERIC(15,10), @LONG NUMERIC(15,10), @BID NUMERIC(10,0) OUTPUT
4. AS
5. BEGIN
6. BEGIN TRANSACTION
7. INSERT INTO Branch(Bname,Baddress,Latitude,Longitude)
8. VALUES(@REGNAME,@REGADDR,@LAT,@LONG);
9. SET @BID=SCOPE\_IDENTITY();
10. UPDATE Branch SET RID=@BID WHERE BID=@BID;
11. UPDATE Regions SET RBranchID=@BID WHERE RegionID=@REGID;
12. COMMIT
13. END

## Update Regional Branch

1. CREATE PROCEDURE [dbo].[UpdateRegB]
2. @OLD\_BID NUMERIC(10,0),@OLD\_BNAME NVARCHAR(300),@OLD\_BADDR NVARCHAR(300),
3. @OLD\_LAT NUMERIC(15,10),@OLD\_LON NUMERIC(15,10),@OLD\_STATUS NVARCHAR(50),
4. @BNAME NVARCHAR(300),@BADDR NVARCHAR(300),@LAT NUMERIC(15,10),
5. @LON NUMERIC(15,10), @STATUS NVARCHAR(50)
6. AS
7. BEGIN
8. UPDATE Branch SET Bname=@BNAME, Baddress=@BADDR,Latitude=@LAT,
9. Longitude=@LON,Status=@STATUS
10. WHERE
11. BID=RID AND BID=@OLD\_BID AND Bname=@OLD\_BNAME
12. AND Baddress=@OLD\_BADDR AND Latitude=@OLD\_LAT
13. AND Longitude=@OLD\_LON AND Status=@OLD\_STATUS;
14. END

## Get Regional Branch List

1. CREATE PROCEDURE [dbo].[GetRegBranchlist]
2. AS
3. BEGIN
4. SELECT BID,Bname FROM Branch WHERE RID=BID AND Status='Active';
5. END

## Insert Sub-Branch

1. CREATE PROCEDURE [dbo].[InserSubB]
2. @RID NUMERIC(18,0),
3. @NAME NVARCHAR(300),
4. @ADDR NVARCHAR(300),
5. @LAT NUMERIC(15,10),
6. @LONG NUMERIC(15,10),
7. @BID NUMERIC(10,0) OUTPUT
8. AS
9. BEGIN
10. INSERT INTO Branch(RID,Bname,Baddress,Latitude,Longitude)
11. VALUES(@RID,@NAME,@ADDR,@LAT,@LONG);
12. SET @BID=SCOPE\_IDENTITY();
13. END

## Update Sub-Branch

1. CREATE PROCEDURE [dbo].[UpdateSubB]
2. @OLD\_BID NUMERIC(10,0), @OLD\_BNAME NVARCHAR(300),
3. @OLD\_BADDR NVARCHAR(300), @OLD\_LAT NUMERIC(15,10),
4. @OLD\_LON NUMERIC(15,10), @OLD\_STATUS NVARCHAR(50),
5. @BNAME NVARCHAR(300), @BADDR NVARCHAR(300),
6. @LAT NUMERIC(15,10), @LON NUMERIC(15,10),
7. @STATUS NVARCHAR(50)
8. AS
9. BEGIN
10. UPDATE Branch SET Bname=@BNAME,Baddress=@BADDR,Latitude=@LAT,
11. Longitude=@LON,Status=@STATUS
12. WHERE
13. BID<>RID AND BID=@OLD\_BID
14. AND Bname=@OLD\_BNAME AND Baddress=@OLD\_BADDR
15. AND Latitude=@OLD\_LAT AND Longitude=@OLD\_LON
16. AND Status=@OLD\_STATUS;
17. END

## Get Sub-Branch List By Region Branch ID

1. CREATE PROCEDURE [dbo].[GetSubBranchByRID]
2. @RID NUMERIC(10,0)
3. AS
4. BEGIN
5. SELECT BID,RID,Bname,Baddress,Latitude,Longitude,Status
6. FROM Branch
7. WHERE BID<>RID AND RID=@RID;
8. END

## Insert Parcel

1. ALTER PROCEDURE [dbo].[InsertParcel]
2. @PTYPE NVARCHAR(50),
3. @WEIGHT NUMERIC(6,3),
4. @PAYSTATUS NVARCHAR(50),
5. @PAYMETHOD NVARCHAR(50),
6. @sCID NVARCHAR(20),
7. @sNAME NVARCHAR(300),
8. @sADDR NVARCHAR(300),
9. @sBID NUMERIC(10,0),
10. @sRID NUMERIC(10,0),
11. @sEMAIL NVARCHAR(100),
13. @rCID NVARCHAR(20),
14. @rNAME NVARCHAR(300),
15. @rADDR NVARCHAR(300),
16. @rBID NUMERIC(10,0),
17. @rRID NUMERIC(10,0),
18. @rEMAIL NVARCHAR(100),
19. @PByCus int,
20. @Container int
21. AS
22. BEGIN
23. DECLARE @PLID UNIQUEIDENTIFIER = NEWID();
24. BEGIN TRANSACTION
25. INSERT INTO
26. PARCEL(PLID,ParcelType,Distance,Weight,PByCus,
27. InvoicePrice,PaymentStatus,PaymentMethod,PaymentDate,
28. sCID,sName,sAddr,sBID,sEMAIL,rCID,rName,rAddr,rBID,
29. rEMAIL,Container)
30. VALUES(@PLID,@PTYPE,0,@WEIGHT,@PByCus,0,@PAYSTATUS,@PAYMETHOD,NULL,
31. @sCID,@sNAME,@sADDR,@sBID,@sEMAIL,@rCID,@rNAME,@rADDR,@rBID,@rEMAIL,
32. @Container);
33. INSERT INTO PBOUNDPATH(PLID,CP1,CP2,CP3,CP4)
34. VALUES(@PLID,@sBID,@sRID,@rRID,@rBID);
35. INSERT INTO
36. PSTATUS(PLID,Cstatus,CBIDstat,placedReq,reqProcessedBy,
37. OutfrPickUpBy,PickedUpBy,ApprovedBy,CP2recBy,CP2senBy,
38. CP3recBy,CP3senBy,CP4recBy,OutForDelBy,DeliveredBy)
39. VALUES(@PLID,'Processing Request',0,GETDATE(),
40. -1,-1,-1,-1,-1,-1,-1,-1,-1,-1,-1);
41. COMMIT
42. SELECT @PLID AS PARCELID;
43. END

## 

## Get Parcel Info

1. CREATE PROCEDURE [dbo].[GetParcelInfo]
2. @PLID UNIQUEIDENTIFIER
3. AS
4. BEGIN
5. SELECT
6. PR.PLID AS PLID,PR.ParcelType AS PTYPE,
7. PR.Distance AS DISTANCE,PR.Weight AS WEIGHT,
8. PR.PByCus AS PACKAGEBYCUS,PR.InvoicePrice AS INVOICE,
9. PR.PaymentStatus AS PAYSTATUS,PR.PaymentMethod AS PAYMETHOD,
10. PR.PaymentDate AS PAYDATE,PR.Container AS CONTAINER,
11. PR.sCID AS SCID,PR.sName AS SNAME,
12. PR.sAddr AS SADDR,PR.sBID AS SBID,
13. PR.sEMAIL AS SEMAIL,
14. PR.rCID AS RCID,PR.rName AS RNAME,
15. PR.rAddr AS RADDR,PR.rBID AS RBID,
16. PR.rEMAIL AS REMAIL,
17. PS.Cstatus AS CSTATUS,PS.CBIDstat AS CBIDSTAT,
18. PS.placedReq AS REQDATE,
20. PS.reqProcessedBy AS PROCESSED\_BY,
21. PS.reqProcessing AS PROCESSED\_DATE,
23. PS.OutfrPickUpBy AS OUTFRPICKUP\_BY,
24. PS.OutfrPickUp AS OUTFRPICKUP\_DATE,
25. PS.PickedUpBy AS PICKEDUP\_BY,
26. PS.PickedUp AS PICKEDUP\_DATE,
27. PS.ApprovedBy AS APPROVED\_BY,
28. PS.Approved AS APPROVED\_DATE,
29. PS.CP2recBy AS CP2REC\_BY,PS.CP2rec AS CP2REC\_DATE,
30. PS.CP2senBy AS CP2SENT\_BY,PS.CP2sen AS CP2SENT\_DATE,
31. PS.CP3recBy AS CP3REC\_BY,PS.CP3rec AS CP3REC\_DATE,
32. PS.CP3senBy AS CP3SENT\_BY,PS.CP3sen AS CP3SENT\_DATE,
33. PS.CP4recBy AS CP4REC\_BY,PS.CP4rec AS CP4REC\_DATE,
34. PS.OutForDelBy AS OUTFORDEL\_BY,PS.OutForDel AS OUTFORDEL\_DATE,
35. PS.DeliveredBy AS DELIVERED\_BY,PS.Delivered AS DELIVERED\_DATE,
36. PS.Returned AS RETURNED\_DATE,PS.Canceled AS CANCELED\_DATE,
37. PS.Rejected AS REJECTED\_DATE,PS.Disapproved AS DISAPPROVED\_DATE,
38. PB.CP1,PB.CP2,PB.CP3,PB.CP4
39. FROM
40. PARCEL PR JOIN
41. PSTATUS PS ON PS.PLID=PR.PLID JOIN
42. PBOUNDPATH PB ON PB.PLID=PR.PLID
43. WHERE PR.PLID=@PLID
44. ORDER BY PS.placedReq DESC;
45. END

## Get Parcel info By Parcel ID

1. CREATE PROCEDURE [dbo].[SrcParcelByPLID]
2. @PLID UNIQUEIDENTIFIER
3. AS
4. BEGIN
5. SELECT
6. PR.PLID AS PARCELid,
7. PR.sName AS SENDER,
8. PR.rName AS RECEIVER,
9. PR.ParcelType AS PARCELTYPE,
10. PS.Cstatus AS CURRENTSTATUS,
11. PR.PaymentStatus AS PAYMENTSTATUS,
12. PS.placedReq AS DATE
13. FROM
14. PARCEL PR JOIN PSTATUS PS ON PS.PLID=PR.PLID
15. WHERE PS.PLID=@PLID;
16. END

## Get Parcel(Sent/Received) Info By Customer ID

1. CREATE PROCEDURE [dbo].[SentParcelByCID]
2. @CID NVARCHAR(20),
3. @TYPE int
4. AS
5. BEGIN
6. IF(@TYPE=0)
7. BEGIN
8. SELECT
9. PR.PLID AS PARCELid,
10. PR.sName AS SENDER,
11. PR.rName AS RECEIVER,
12. PR.ParcelType AS PARCELTYPE,
13. PS.Cstatus AS CURRENTSTATUS,
14. PR.PaymentStatus AS PAYMENTSTATUS,
15. PS.placedReq AS DATE
16. FROM PARCEL PR JOIN PSTATUS PS ON PS.PLID=PR.PLID
17. WHERE pr.sCID=@CID
18. ORDER BY PS.placedReq DESC;
19. END
20. ELSE IF(@TYPE=1)
21. BEGIN
22. SELECT
23. PR.PLID AS PARCELid,
24. PR.sName AS SENDER,
25. PR.rName AS RECEIVER,
26. PR.ParcelType AS PARCELTYPE,
27. PS.Cstatus AS CURRENTSTATUS,
28. PR.PaymentStatus AS PAYMENTSTATUS,
29. PS.placedReq AS DATE
30. FROM PARCEL PR JOIN PSTATUS PS ON PS.PLID=PR.PLID
31. WHERE pr.rCID=@CID
32. ORDER BY PS.placedReq DESC;
33. END
34. END

## Insert Rates

1. CREATE PROCEDURE [dbo].[InsertRates]
2. @WEIGHT FLOAT,
3. @BASIC FLOAT,
4. @BUSINESS FLOAT,
5. @DRYFOOD FLOAT,
6. @COOKEDFOOD FLOAT,
7. @GIFT FLOAT,
8. @ELEC FLOAT,
9. @RATEID INT OUTPUT
10. AS
11. BEGIN
12. INSERT INTO
13. ParcelRates(WEIGHT,Basic,Business,DryFood,CookedFood,Gift,Electronics)
14. VALUES(@WEIGHT,@BASIC,@BUSINESS,@DRYFOOD,@COOKEDFOOD,@GIFT,@ELEC);
16. SET @RATEID=SCOPE\_IDENTITY();
17. END

## Update Rates

1. CREATE PROCEDURE [dbo].[UpdateRate]
2. @RATEID INT,@O\_WEIGHT INT,
3. @O\_BASIC FLOAT,@O\_BUSINESS FLOAT,
4. @O\_DF FLOAT,@O\_CF FLOAT,
5. @O\_GIFT FLOAT, @O\_ELEC FLOAT,
6. @WEIGHT INT,@BASIC FLOAT,
7. @BUSINESS FLOAT,@DF FLOAT,
8. @CF FLOAT,@GIFT FLOAT,
9. @ELEC FLOAT
10. AS
11. BEGIN
12. UPDATE ParcelRates
13. SET WEIGHT=@WEIGHT, Basic=@BASIC,
14. Business=@BUSINESS, DryFood=@DF,
15. CookedFood=@CF,Gift=@GIFT,
16. Electronics=@ELEC
17. WHERE
18. RateID=@RATEID AND WEIGHT=@O\_WEIGHT
19. AND Basic=@O\_BASIC AND Business=@O\_BUSINESS
20. AND DryFood=@O\_DF AND CookedFood=@O\_CF
21. AND Gift=@O\_GIFT AND Electronics=@O\_ELEC;
22. END

## Get Rates

1. CREATE PROCEDURE [dbo].[GetRates]
2. AS
3. BEGIN
4. SELECT \* FROM ParcelRates;
5. END
6. **Class Files(\*.CS)**
7. Customer(Customer.cs):

using System;

using System.Data;

using System.Data.SqlClient;

using System.IO;

using System.Web;

namespace OnlineCourierService.Class

{

public class Customer : Password

{

private static readonly string connectionstr = "Data Source=DESKTOP-74RBQ7M\\KUNTALSQLS;Initial Catalog=CourierService;Persist Security Info=True;User ID=sa;Password=!@Kd\_Hell45";

public string cid { private get; set; }

public string name { get; set; }

public string address { get; set; }

public string gender { get; set; }

public DateTime dob { get; set; }

public long phno { get; set; }

public string email { get; set; }

public long BID { get; set; }

public string accType { get; set; }

public string status { get; set; }

public string pass { private get; set; }

public string imgurl { get; set; }

public static Customer GetinfoByEmail(string Email)

{

Customer cus = null;

using (SqlConnection con = new SqlConnection(connectionstr))

{

try

{

con.Open();

string q = "Select CID from Customer WHERE Email=@EMAIL";

SqlCommand cmd = new SqlCommand(q, con);

cmd.Parameters.AddWithValue("@EMAIL", Email);

cus = new Customer(Convert.ToString(cmd.ExecuteScalar()), true);

}

catch (SqlException){return cus;}

catch (Exception){return cus;}

finally{con.Close();}

}

cus.FillData();

return cus;

}

public Customer(string CID, bool isActive)

{

this.cid = CID;

this.status = (isActive) ? "Active" : "Inactive";

}

public Customer(string email)

{

this.phno = -1;

this.email = email;

this.status = "Active";

}

public Customer(long phno)

{

this.email = null;

this.phno = phno;

this.status = "Active";

}

public Customer(string name, string address, string gender, DateTime dob, long phno, string

email, long BID, string accType, string pass)

{

this.name = name;

this.address = address;

this.gender = gender;

this.dob = dob;

this.phno = phno;

this.email = email;

this.BID = BID;

this.accType = accType;

this.pass = pass;

this.status = "Active";

}

public static string isPasswordResetLinkValid(string Uid)

{

string cid = null;

using (SqlConnection con = new SqlConnection(connectionstr))

{

try

{

con.Open();

SqlCommand cmd = new SqlCommand("CIsPasswordResetLinkValid", con);

cmd.CommandType = CommandType.StoredProcedure;

cmd.Parameters.AddWithValue("@GUID", Uid);

cid = cmd.ExecuteScalar().ToString();

}

catch (SqlException) { return null; }

catch (NullReferenceException) { return null; }

catch (Exception) { return null; }

finally { con.Close(); }

}

return cid;

}

public static int ResetPassword(string guid, string pass)

{

int c = -1;

using (SqlConnection con = new SqlConnection(connectionstr)){

try{

con.Open();

SqlCommand cmd = new SqlCommand("CResetPassword", con);

cmd.CommandType = CommandType.StoredProcedure;

cmd.Parameters.AddWithValue("@GUID", guid);

cmd.Parameters.AddWithValue("@Password", Password.HashPassword(pass));

c = Convert.ToInt32(cmd.ExecuteScalar());

}

catch (SqlException) { return c; }

catch (Exception) { return c; }

finally { con.Close(); }

}

return c;

}

public static int GenerateResetPasswordLink(string Email, string link)

{

int success = -1;

using (SqlConnection con = new SqlConnection(connectionstr))

{

con.Open();

SqlCommand cmd = new SqlCommand("CRequestResetPassword", con);

cmd.CommandType = CommandType.StoredProcedure;

cmd.Parameters.AddWithValue("@Email", Email);

SqlDataReader rdr = cmd.ExecuteReader();

while (rdr.Read())

{

if (Convert.ToBoolean(rdr["ReturnCode"]))

{

if (Password.SendPasswordResetEmail(Email, rdr["Name"].ToString(),

rdr["UniqueId"].ToString(), link)){ success = 1; }

}

else{success = -1; }

}

con.Close();

return success;

}

}

public string GetCIDbyPass(string pass)

{

this.cid = null;

pass = Password.HashPassword(pass);

using (SqlConnection con = new SqlConnection(connectionstr))

{

try

{

con.Open();

SqlCommand cmd = new SqlCommand("GetCIDbyPass", con);

cmd.CommandType = CommandType.StoredProcedure;

if (this.email != null)

{

cmd.Parameters.AddWithValue("@PHNO", null);

cmd.Parameters.AddWithValue("@EMAIL", this.email);

cmd.Parameters.AddWithValue("@PASS", pass);

}

else if (this.phno != -1)

{

cmd.Parameters.AddWithValue("@PHNO", this.phno);

cmd.Parameters.AddWithValue("@EMAIL", null);

cmd.Parameters.AddWithValue("@PASS", pass);

}

else

{

cmd.Parameters.AddWithValue("@PHNO", this.phno);

cmd.Parameters.AddWithValue("@EMAIL", this.email);

cmd.Parameters.AddWithValue("@PASS", pass);

}

this.cid = cmd.ExecuteScalar().ToString();

return this.cid;

}

catch (SqlException){return this.cid;}

catch (Exception){return this.cid;}

finally{con.Close();}

}

}

public int UpdatePassword(string newpass, string oldpass)

{

using (SqlConnection con = new SqlConnection(connectionstr))

{

try

{

con.Open();

SqlCommand cmd = new SqlCommand("UpdateCPass", con);

cmd.CommandType = CommandType.StoredProcedure;

cmd.Parameters.AddWithValue("@OLD\_CID", this.cid);

cmd.Parameters.AddWithValue("@OLD\_PASS", Password.HashPassword(oldpass));

cmd.Parameters.AddWithValue("@PASS", Password.HashPassword(newpass));

return cmd.ExecuteNonQuery();

}

catch (SqlException){

return -1;

}finally{con.Close();}

}

}

public static int CountCustomer(long phno)

{

int count = -1;

using (SqlConnection con = new SqlConnection(connectionstr))

{

try

{

con.Open();

SqlCommand cmd = new SqlCommand("CountCbyPhno", con);

cmd.CommandType = CommandType.StoredProcedure;

cmd.Parameters.AddWithValue("@PHNO", phno);

count = Convert.ToInt32(cmd.ExecuteScalar());

}

catch (SqlException){return count;}

catch (Exception){return count;}

finally{con.Close();}

}

return count;

}

public static int CountCustomer(string email)

{

int count = -1;

using (SqlConnection con = new SqlConnection(connectionstr))

{

try

{

con.Open();

SqlCommand cmd = new SqlCommand("CountCbyEmail", con);

cmd.CommandType = CommandType.StoredProcedure;

cmd.Parameters.AddWithValue("@EMAIL", email);

count = Convert.ToInt32(cmd.ExecuteScalar());

}

catch (SqlException){return count;}

catch (Exception){return count;}

finally{con.Close();}

}

return count;

}

public int FillData()

{

using (SqlConnection con = new SqlConnection(connectionstr))

{

try

{

con.Open();

SqlDataAdapter da = new SqlDataAdapter("GetCusInfo", con);

da.SelectCommand.CommandType = CommandType.StoredProcedure;

DataTable data = new DataTable("CustomerInfo");

da.SelectCommand.Parameters.AddWithValue("@CID", this.cid);

da.Fill(data);

if (data.Equals(null)){return -1;}

foreach (DataRow row in data.Rows){

this.cid = row.Field<string>("CID");

this.name = row.Field<string>("NAME");

this.address = row.Field<string>("ADDR");

this.gender = row.Field<string>("GENDER");

this.dob = row.Field<DateTime>("DOB");

this.phno = Convert.ToInt64(row.Field<decimal>("PHNO"));

this.email = row.Field<string>("EMAIL");

this.BID = Convert.ToInt64(row.Field<decimal>("BID"));

this.accType = row.Field<string>("ACCTYPE");

this.status = row.Field<string>("STATUS");

if (row.Field<byte[]>("IMGDATA") != null){

this.imgurl = Convert.ToBase64String(row.Field<byte[]>("IMGDATA"));

this.imgurl = ("data:Image/png;base64," + imgurl);

}

}

return 1;

}

catch (SqlException){return -1;}

catch (Exception){return -1;}

finally{con.Close();}

}

}

public string InsertData()

{

using (SqlConnection con = new SqlConnection(connectionstr))

{

try

{

con.Open();

SqlCommand cmd = new SqlCommand("InsertCustomer", con);

cmd.CommandType = CommandType.StoredProcedure;

cmd.Parameters.AddWithValue("@NAME", name);

cmd.Parameters.AddWithValue("@ADDRESS", address);

cmd.Parameters.AddWithValue("@GENDER", gender);

cmd.Parameters.AddWithValue("@DOB", dob);

cmd.Parameters.AddWithValue("@PHNO", phno);

cmd.Parameters.AddWithValue("@EMAIL", email);

cmd.Parameters.AddWithValue("@BID", BID);

cmd.Parameters.AddWithValue("@ACCTYPE", accType);

cmd.Parameters.AddWithValue("@STATUS", status);

cmd.Parameters.AddWithValue("@PASS", Password.HashPassword(pass));

this.cid = Convert.ToString(cmd.ExecuteScalar());

return this.cid;

}

catch (SqlException){return null;}

catch (Exception){return null;}

finally{con.Close();}

}

}

public int InsertUpdateImage(HttpPostedFile img)

{

string fileName = Path.GetFileName(img.FileName);

string fileExtennsion = Path.GetExtension(fileName).ToLower();

int fileSize = img.ContentLength;//length or size of the file

if (fileExtennsion == ".jpg" || fileExtennsion == ".jpeg"

|| fileExtennsion == ".png")

{

Stream imgstream = img.InputStream;

BinaryReader imgbin = new BinaryReader(imgstream);

byte[] imgBytes = imgbin.ReadBytes((int)imgstream.Length);

using (SqlConnection con = new SqlConnection(connectionstr)){

try

{

con.Open();

SqlCommand cmd = new SqlCommand("InsertUpdateCImage", con);

cmd.CommandType = CommandType.StoredProcedure;

cmd.Parameters.AddWithValue("@CID", this.cid);

cmd.Parameters.AddWithValue("@NAME", fileName);

cmd.Parameters.AddWithValue("@SIZE", fileSize);

cmd.Parameters.AddWithValue("@IMGDATA", imgBytes);

cmd.ExecuteNonQuery();

return 1;

}

catch (Exception){return -1;}

finally{con.Close();}

}

}

else{return -1;}

}

public int UpdateData(Customer newCus)

{

using (SqlConnection con = new SqlConnection(connectionstr))

{

try

{

con.Open();

SqlCommand cmd = new SqlCommand("UpdateCustomer", con);

cmd.CommandType = CommandType.StoredProcedure;

cmd.Parameters.AddWithValue("@OLD\_CID", this.cid);

cmd.Parameters.AddWithValue("@OLD\_NAME", this.name);

cmd.Parameters.AddWithValue("@NAME", newCus.name);

cmd.Parameters.AddWithValue("@OLD\_ADDR", this.address);

cmd.Parameters.AddWithValue("@ADDR", newCus.address);

cmd.Parameters.AddWithValue("@OLD\_GENDER", this.gender);

cmd.Parameters.AddWithValue("@GENDER", newCus.gender);

cmd.Parameters.AddWithValue("@OLD\_DOB", this.dob);

cmd.Parameters.AddWithValue("@DOB", newCus.dob);

cmd.Parameters.AddWithValue("@OLD\_PHNO", this.phno);

cmd.Parameters.AddWithValue("@PHNO", newCus.phno);

cmd.Parameters.AddWithValue("@OLD\_EMAIL", this.email);

cmd.Parameters.AddWithValue("@EMAIL", newCus.email);

cmd.Parameters.AddWithValue("@OLD\_BID", this.BID);

cmd.Parameters.AddWithValue("@BID", newCus.BID);

cmd.Parameters.AddWithValue("@OLD\_ACCTYPE", this.accType);

cmd.Parameters.AddWithValue("@ACCTYPE", newCus.accType);

cmd.Parameters.AddWithValue("@OLD\_STATUS", this.status);

cmd.Parameters.AddWithValue("@STATUS", newCus.status);

return cmd.ExecuteNonQuery();

}

catch (SqlException){return -1;}

catch (Exception){return -1;}

finally{con.Close();}

}

}

}

}

1. Package(Package.cs)

using OnlineCourierService.employee.Classes;

using System;

using System.Collections.Generic;

using System.Data;

using System.Data.SqlClient;

namespace OnlineCourierService.Class

{

public class Status

{

public string remark { get; private set; }

public int eid { get; private set; }

public Object date { get; private set; }

public Status(string remark, int eid, Object date)

{

this.eid = eid;

this.date = date;

this.remark = remark;

}

}

public class SenderReceiver

{

public string CID { get; set; }

public string Name { get; set; }

public string Address { get; set; }

public string Email { get; set; }

public SenderReceiver(string CID, string Name, string Email, string Address)

{

this.CID = CID;

this.Name = Name;

this.Email = Email;

this.Address = Address;

}

public SenderReceiver() { }

}

public class Package

{

private static readonly string connectionstr = "Data Source=DESKTOP-74RBQ7M\\KUNTALSQLS;Initial Catalog=CourierService;Persist Security Info=True;User ID=sa;Password=!@Kd\_Hell45";

public static readonly DateTime nullDate = new DateTime(1800, 12, 31);

public string ParcelID { private get; set; }

public string ParcelType { get; private set; }

public double Weight { get; private set; }

public long SourceBID { get; private set; }

public long SourceRID { get; private set; }

public long DestBID { get; private set; }

public long DestRID { get; private set; }

public double invoicePrice { get; private set; }

public string PaymentMethod { get; private set; }

public string PaymentStatus { get; private set; }

public DateTime PayDate { get; private set; }

public int PackagingByCustomer { get; private set; }

public int Container { get; private set; }

public double Distance { get; private set; }

public DateTime ReqDate { get; private set; }

public long bidStatus { get; private set;

public string currentStatus { get; private set; }

public SenderReceiver Sender { get; private set; }

public SenderReceiver Receiver { get; private set; }

public List<Status> statuslist { get; private set; }

public Object REJECTED\_DATE { get; set; }

public Object DISAPPROVED\_DATE { get; set; }

public Object RETURNED\_DATE { get; set; }

public Object CANCELED\_DATE { get; set; }

public Package(string PacelType, double Weight, SenderReceiver Sender,

SenderReceiver Receiver, long SourceBID, long DestBID,

long SourceRID, long DestRID, long bidStatus,

double invoicePrice, string PaymentMethod, string PaymentStatus,

int PackagingByCustomer, int Container)

{

this.ParcelType = PacelType;

this.Weight = Weight;

this.Sender = Sender;

this.Receiver = Receiver;

this.SourceBID = SourceBID;

this.SourceRID = SourceRID;

this.DestBID = DestBID;

this.DestRID = DestRID;

this.bidStatus = bidStatus;

this.invoicePrice = invoicePrice;

this.PaymentMethod = PaymentMethod;

this.PaymentStatus = PaymentStatus;

this.PackagingByCustomer = PackagingByCustomer;

this.Container = Container;

this.PayDate = nullDate;

this.Distance = 0;

this.statuslist = null;

}

public Package(string PLID){

this.ParcelID = PLID;

this.ParcelType = null;

this.Weight = -1;

this.Sender = null;

this.Receiver = null;

this.SourceBID = -1;

this.SourceRID = -1;

this.DestBID = -1;

this.DestRID = -1;

this.bidStatus = -1;

this.currentStatus = null;

this.invoicePrice = -1;

this.PaymentMethod = null;

this.PaymentStatus = null;

this.PayDate = nullDate;

this.PackagingByCustomer = -1;

this.Container = -1;

this.Distance = -1;

this.statuslist = null;

}

public string insertPackage(){

using (SqlConnection con = new SqlConnection(connectionstr))

{

try

{

con.Open();

if (Customer.CountCustomer(Sender.Email) == 1 && Sender.CID == "-1")

{

SqlCommand cmd = new SqlCommand("Select CID from Customer WHERE

Email=@EMAIL", con);

cmd.Parameters.AddWithValue("@EMAIL", Sender.Email);

Sender.CID = cmd.ExecuteScalar().ToString();

}

if(Customer.CountCustomer(Receiver.Email) == 1 && Receiver.CID == "-1")

{

SqlCommand cmd = new SqlCommand("Select CID from Customer WHERE

Email=@EMAIL", con);

cmd.Parameters.AddWithValue("@EMAIL", Receiver.Email);

Receiver.CID = cmd.ExecuteScalar().ToString();

}

SqlCommand cmd1 = new SqlCommand("InsertParcel", con);

cmd1.CommandType = CommandType.StoredProcedure;

cmd1.Parameters.AddWithValue("@PTYPE", this.ParcelType);

cmd1.Parameters.AddWithValue("@WEIGHT", this.Weight);

cmd1.Parameters.AddWithValue("@PAYSTATUS", this.PaymentStatus);

cmd1.Parameters.AddWithValue("@PAYMETHOD", this.PaymentMethod);

cmd1.Parameters.AddWithValue("@sCID", this.Sender.CID);

cmd1.Parameters.AddWithValue("@sNAME", this.Sender.Name);

cmd1.Parameters.AddWithValue("@sADDR", this.Sender.Address);

cmd1.Parameters.AddWithValue("@sEMAIL", this.Sender.Email);

cmd1.Parameters.AddWithValue("@rCID", this.Receiver.CID);

cmd1.Parameters.AddWithValue("@rNAME", this.Receiver.Name);

cmd1.Parameters.AddWithValue("@rADDR", this.Receiver.Address);

cmd1.Parameters.AddWithValue("@rEMAIL", this.Receiver.Email);

cmd1.Parameters.AddWithValue("@PByCus", this.PackagingByCustomer);

cmd1.Parameters.AddWithValue("@Container", this.Container);

cmd1.Parameters.AddWithValue("@sBID", this.SourceBID);

if (SourceBID == SourceRID || DestRID == SourceRID || DestBID == SourceRID)

{

cmd1.Parameters.AddWithValue("@sRID", -1);

}

else

{

cmd1.Parameters.AddWithValue("@sRID", this.SourceRID);

}

cmd1.Parameters.AddWithValue("@rBID", this.DestBID);

if (DestBID == DestRID || SourceRID == DestRID || SourceBID == DestRID)

{

cmd1.Parameters.AddWithValue("@rRID", -1);

}

else

{

cmd1.Parameters.AddWithValue("@rRID", this.DestRID);

}

return Convert.ToString(cmd1.ExecuteScalar());

}

catch (SqlException) { return null; }

catch (Exception) { return null; }

finally { con.Close(); }

}

}

public static DataTable GetParcelsByPLID(string PLID)

{

DataTable data = new DataTable("Parcel by PLID");

using (SqlConnection con = new SqlConnection(connectionstr))

{

try

{

con.Open();

SqlDataAdapter da = new SqlDataAdapter("SrcParcelByPLID", con);

da.SelectCommand.CommandType = CommandType.StoredProcedure;

da.SelectCommand.Parameters.AddWithValue("@PLID", PLID);

da.Fill(data);

return data;

}

catch (Exception) { return null; }

finally { con.Close(); }

}

}

public static DataTable GetParcelsByCID(string CID, int type)

**{**

**DataTable data = new DataTable("**Parcel by PLID");

using (SqlConnection con = new SqlConnection(connectionstr))

{

try

{

con.Open();

SqlDataAdapter da = new SqlDataAdapter("SentParcelByCID", con);

da.SelectCommand.CommandType = CommandType.StoredProcedure;

da.SelectCommand.Parameters.AddWithValue("@CID", CID);

da.SelectCommand.Parameters.AddWithValue("@type", type);

da.Fill(data);

return data;

}

catch (Exception) { return null; }

finally { con.Close(); }

}

}

public int FillData()

{

using (SqlConnection con = new SqlConnection(connectionstr))

{

try

{

con.Open();

SqlDataAdapter da = new SqlDataAdapter("GetParcelInfo", con);

da.SelectCommand.CommandType = CommandType.StoredProcedure;

da.SelectCommand.Parameters.AddWithValue("@PLID", this.ParcelID);

DataTable data = new DataTable("Parcel Details");

da.Fill(data);

if (data.Rows.Count != 1) { return -1; }

foreach (DataRow dataRow in data.Rows){

this.ParcelType = dataRow.Field<string>("PTYPE");

this.Distance = Convert.ToDouble(dataRow.Field<decimal>("DISTANCE"));

this.Weight = Convert.ToDouble(dataRow.Field<decimal>("WEIGHT"));

this.PackagingByCustomer = dataRow.Field<int>("PACKAGEBYCUS");

this.invoicePrice = Convert.ToDouble(dataRow.Field<decimal>("INVOICE"));

this.PaymentStatus = dataRow.Field<string>("PAYSTATUS");

this.PaymentMethod = dataRow.Field<string>("PAYMETHOD");

var date = dataRow.Field<Object>("PAYDATE");

this.PayDate = (date == null) ? nullDate : (DateTime)date;

this.Container = dataRow.Field<int>("CONTAINER");

this.Sender = new SenderReceiver();

this.Sender.CID = dataRow.Field<string>("SCID");

this.Sender.Name = dataRow.Field<string>("SNAME");

this.Sender.Address = dataRow.Field<string>("SADDR");

this.Sender.Email = dataRow.Field<string>("SEMAIL");

this.Receiver = new SenderReceiver();

this.Receiver.CID = dataRow.Field<string>("RCID");

this.Receiver.Name = dataRow.Field<string>("RNAME");

this.Receiver.Address = dataRow.Field<string>("RADDR");

this.Receiver.Email = dataRow.Field<string>("REMAIL");

this.SourceBID = Convert.ToInt64(dataRow.Field<decimal>("SBID"));

this.SourceRID = Convert.ToInt64(dataRow.Field<decimal>("CP2"));

this.DestBID = Convert.ToInt64(dataRow.Field<decimal>("RBID"));

this.DestRID = Convert.ToInt64(dataRow.Field<decimal>("CP3"));

this.currentStatus = dataRow.Field<string>("CSTATUS");

this.bidStatus = Convert.ToInt64(dataRow.Field<decimal>("CBIDSTAT"));

this.ReqDate = dataRow.Field<DateTime>("REQDATE");

string cp1, cp2 = "", cp3 = "", cp4;

cp1 = (SourceBID != -1) ? Branch.GetBranchNameByBID(SourceBID) + "(" +

SourceBID + ")" : "";

cp4 = (DestBID != -1) ? Branch.GetBranchNameByBID(DestBID) + "(" + DestBID

+ ")" : "";

if (SourceRID == -1 && DestRID == -1)

{

cp2 = cp3 = Branch.GetBranchNameByBID(DestBID) + "(" + DestBID + ")";

}

else if (SourceRID == -1)

{

cp2 = Branch.GetBranchNameByBID(DestRID) + "(" + DestRID + ")";

cp3 = Branch.GetBranchNameByBID(DestRID) + "(" + DestRID + ")";

}

else if (DestRID == -1)

{

cp2 = Branch.GetBranchNameByBID(SourceRID) + "(" + SourceRID + ")";

cp3 = Branch.GetBranchNameByBID(DestBID) + "(" + DestBID + ")";

}

else

{

cp2 = Branch.GetBranchNameByBID(SourceRID) + "(" + SourceRID + ")";

cp3 = Branch.GetBranchNameByBID(DestRID) + "(" + DestRID + ")";

}

this.statuslist = new List<Status>();

int eid = Convert.ToInt32(dataRow.Field<decimal>("PROCESSED\_BY"));

this.statuslist.Add(new Status("Request processed at " + cp1 + " By ",

eid, dataRow.Field<Object>("PROCESSED\_DATE")));

eid = Convert.ToInt32(dataRow.Field<decimal>("OUTFRPICKUP\_BY"));

this.statuslist.Add(new Status("Out for Pick Up By ", eid,

dataRow.Field<Object>("OUTFRPICKUP\_DATE")));

eid = Convert.ToInt32(dataRow.Field<decimal>("PICKEDUP\_BY"));

this.statuslist.Add(new Status("Parcel Picked Up By ", eid,

dataRow.Field<Object>("PICKEDUP\_DATE")));

eid = Convert.ToInt32(dataRow.Field<decimal>("APPROVED\_BY"));

if (SourceBID != DestBID)

{

this.statuslist.Add(new Status("Parcel Approved and Sent to " + cp2 +

" By ", eid, dataRow.Field<Object>("APPROVED\_DATE")));

}

else

{

this.statuslist.Add(new Status("Parcel Approved By ", eid,

dataRow.Field<Object>("APPROVED\_DATE")));

}

eid = Convert.ToInt32(dataRow.Field<decimal>("CP2REC\_BY"));

this.statuslist.Add(new Status("Parcel Received at " + cp2 + " By ", eid,

dataRow.Field<Object>("CP2REC\_DATE")));

eid = Convert.ToInt32(dataRow.Field<decimal>("CP2SENT\_BY"));

this.statuslist.Add(new Status("Parcel Sent to " + cp3 + " By ", eid,

dataRow.Field<Object>("CP2SENT\_DATE")));

eid = Convert.ToInt32(dataRow.Field<decimal>("CP3REC\_BY"));

this.statuslist.Add(new Status("Parcel Received at " + cp3 + " By ", eid,

dataRow.Field<Object>("CP3REC\_DATE")));

eid = Convert.ToInt32(dataRow.Field<decimal>("CP3SENT\_BY"));

this.statuslist.Add(new Status("Parcel Sent to " + cp4 + " By ", eid,

dataRow.Field<Object>("CP3SENT\_DATE")));

eid = Convert.ToInt32(dataRow.Field<decimal>("CP4REC\_BY"));

this.statuslist.Add(new Status("Parcel Received at " + cp4 + " By ", eid,

dataRow.Field<Object>("CP4REC\_DATE")));

eid = Convert.ToInt32(dataRow.Field<decimal>("OUTFORDEL\_BY"));

this.statuslist.Add(new Status("Out for Deivery by ", eid,

dataRow.Field<Object>("OUTFORDEL\_DATE")));

eid = Convert.ToInt32(dataRow.Field<decimal>("DELIVERED\_BY"));

this.statuslist.Add(new Status("Parcel Deivered by ", eid,

dataRow.Field<Object>("DELIVERED\_DATE")));

RETURNED\_DATE = dataRow.Field<Object>("RETURNED\_DATE");

CANCELED\_DATE = dataRow.Field<Object>("CANCELED\_DATE");

REJECTED\_DATE = dataRow.Field<Object>("REJECTED\_DATE");

DISAPPROVED\_DATE = dataRow.Field<Object>("DISAPPROVED\_DATE");

}

}catch (Exception) { return -1; }

finally { con.Close(); }

}

return 0;

}

}

}

1. Password(Password.cs)

using System;

using System.Security.Cryptography;

using System.Text;

using System.Net.Mail;

namespace OnlineCourierService.Class

{

public class Password

{

protected static string HashPassword(string pass)

{

if (pass == null)

{

return null;

}

byte[] barray = null;

using (SHA256 sha256Hash = SHA256.Create())

{

barray = sha256Hash.ComputeHash(Encoding.UTF8.GetBytes(pass));

}

return Convert.ToBase64String(barray);

}

protected static bool SendPasswordResetEmail(string ToEmail, string UserName, string UniqueId, string link){

MailMessage mailMessage = new MailMessage("noreply.lightning.corp599@gmail.com", ToEmail);

StringBuilder sbEmailBody = new StringBuilder();

sbEmailBody.Append("Dear " + UserName + ",<br/><br/>");

sbEmailBody.Append("Please click on the following link to reset your password");

sbEmailBody.Append("<br/>"); sbEmailBody.Append("<a href=\"http://" + link + "?uid=" + UniqueId + "\">Reset Password</a>");

sbEmailBody.Append("<br/> Ignore It If you Have not Made any Request Like This <br/>");

sbEmailBody.Append("<br/><br/>");

sbEmailBody.Append("<b>Lightning Corp.</b><br/><br/>");

sbEmailBody.Append("<tt>This is a Computer Generated Email Do not Reply to it.</tt>");

mailMessage.IsBodyHtml = true;

mailMessage.Body = sbEmailBody.ToString();

mailMessage.Subject = "Reset Your Password";

try

{

SmtpClient smtpClient = new SmtpClient("smtp.gmail.com", 587);

smtpClient.Credentials = new System.Net.NetworkCredential()

{

UserName = "noreply.lightning.corp599@gmail.com",

Password = "\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"

};

smtpClient.EnableSsl = true;

smtpClient.Send(mailMessage);

}

catch (Exception) { return false; }

return true;

}

}

}

1. Branch(Branch.cs)

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Data.SqlClient;

using System.Data;

namespace OnlineCourierService.employee.Classes

{

public class Branch

{

private static readonly string connectionstr = "Data Source=DESKTOP-74RBQ7M\\KUNTALSQLS;Initial Catalog=CourierService;Persist Security Info=True;User ID=sa;Password=!@Kd\_Hell45";

private static SqlConnection con;

public long Bid { get; private set; }

public long Rid { get; private set; }

public string name { get; set; }

public string address { get; set; }

public double lat { get; set; }

public string status { get; set; }

public double lon { get; set; }

public static long GetRidbyBid(long BID)

{

using (SqlConnection con = new SqlConnection(connectionstr))

{

try

{

con.Open();

SqlCommand cmd = new SqlCommand("select RID from Branch where BID=@bid",

con);

cmd.Parameters.AddWithValue("@bid", BID);

return Convert.ToInt64(cmd.ExecuteScalar());

}

catch (SqlException){return -1;}

catch (Exception){return -1;}

finally{con.Close();}

}

}

public static int GetRBidByRegID(int Regid)

{

int RBid = -1;

using (con = new SqlConnection(connectionstr))

{

try

{

con.Open();

SqlCommand cmd = new SqlCommand("GetRBidBy", con);

cmd.CommandType = CommandType.StoredProcedure;

cmd.Parameters.AddWithValue("@RegID", Regid);

string rbid = cmd.ExecuteScalar().ToString();

if (!rbid.Equals(null)){RBid = Convert.ToInt32(rbid);}

}

catch (Exception){return -1;}

finally{con.Close();}

}

return RBid;

}

public static int CreateRegBranch(int regid, string Bname, string Baddr,

double latitude, double longitude)

{

int Bid = -1;

using (con = new SqlConnection(connectionstr))

{

try

{

con.Open();

SqlCommand cmd = new SqlCommand("InsertRegB", con);

cmd.CommandType = CommandType.StoredProcedure;

cmd.Parameters.AddWithValue("@REGID", regid);

cmd.Parameters.AddWithValue("@REGNAME", Bname);

cmd.Parameters.AddWithValue("@REGADDR", Baddr);

cmd.Parameters.AddWithValue("@LAT", latitude);

cmd.Parameters.AddWithValue("@LONG", longitude);

SqlParameter bid = new SqlParameter()

{

ParameterName = "@BID",

Value = -1,

Direction = ParameterDirection.Output

};

cmd.Parameters.Add(bid);

cmd.ExecuteNonQuery();

String rbid = cmd.Parameters["@BID"].Value.ToString();

if (!rbid.Equals(null)){Bid = Convert.ToInt32(rbid);}

}

catch (Exception){return -1;}

finally{con.Close();}

}

return Bid;

}

public static int CountB(string Name, string Address, double latitude, double longitude)

{

int count = -1;

using (con = new SqlConnection(connectionstr))

{

try

{

con.Open();

SqlCommand cmd = new SqlCommand("CountB", con);

cmd.CommandType = CommandType.StoredProcedure;

cmd.Parameters.AddWithValue("@NAME", Name);

cmd.Parameters.AddWithValue("@ADDR", Address);

cmd.Parameters.AddWithValue("@LAT", latitude);

cmd.Parameters.AddWithValue("@LONG", longitude);

count = Convert.ToInt32(cmd.ExecuteScalar());

}

catch (Exception){return -1;}

finally{con.Close();}

}

return count;

}

public static int CreateSubBranch(long rid, string Name, string Address, double latitude, double longitude)

{

int Bid = -1;

using (con = new SqlConnection(connectionstr))

{

try

{

con.Open();

SqlCommand cmd = new SqlCommand("InserSubB", con);

cmd.CommandType = CommandType.StoredProcedure;

cmd.Parameters.AddWithValue("@RID", rid);

cmd.Parameters.AddWithValue("@NAME", Name);

cmd.Parameters.AddWithValue("@ADDR", Address);

cmd.Parameters.AddWithValue("@LAT", latitude);

cmd.Parameters.AddWithValue("@LONG", longitude);

SqlParameter bid = new SqlParameter()

{

ParameterName = "@BID",

Value = -1,

Direction = ParameterDirection.Output

};

cmd.Parameters.Add(bid);

cmd.ExecuteNonQuery();

String rbid = cmd.Parameters["@BID"].Value.ToString();

if (!rbid.Equals(null)){Bid = Convert.ToInt32(rbid);}

}

catch (Exception){return -1;}

finally{con.Close();}

}

return Bid;

}

public static int CountRegB()

{

using (SqlConnection con = new SqlConnection(connectionstr))

{

con.Open();

SqlCommand cmd = new SqlCommand("SELECT COUNT(\*) FROM Branch WHERE RID=BID",

con);

return Convert.ToInt32(cmd.ExecuteScalar());

}

}

public static List<Branch> GetRegBranches(int startIndex, int pageSize)

{

List<Branch> Regbs = new List<Branch>();

using (SqlConnection con = new SqlConnection(connectionstr))

{

con.Open();

SqlDataAdapter da = new SqlDataAdapter("GetRegBranchesByPage", con);

da.SelectCommand.CommandType = CommandType.StoredProcedure;

da.SelectCommand.Parameters.AddWithValue("@STARTI", startIndex);

da.SelectCommand.Parameters.AddWithValue("@PSIZE", pageSize);

DataTable data = new DataTable("RegBranchs");

da.Fill(data);

foreach (DataRow row in data.Rows)

{

Branch b1 = new Branch();

b1.Bid = b1.Rid = Convert.ToInt64(row.Field<decimal>("RID"));

b1.name = row.Field<string>("Bname");

b1.address = row.Field<string>("Baddress");

b1.lat = Convert.ToDouble(row.Field<decimal>("Latitude"));

b1.lon = Convert.ToDouble(row.Field<decimal>("Longitude"));

b1.status = row.Field<string>("Status");

Regbs.Add(b1);

}

}

return Regbs;

}

public static int UpdateRegBranches(long OLD\_Bid, string OLD\_name,

string OLD\_address, double OLD\_lat, double OLD\_lon, string OLD\_status,

string name, string address, double lat, double lon, string status)

{

using (con = new SqlConnection(connectionstr))

{

try

{

con.Open();

SqlCommand cmd = new SqlCommand("UpdateRegB", con);

cmd.CommandType = CommandType.StoredProcedure;

cmd.Parameters.AddWithValue("@OLD\_BID", OLD\_Bid);

cmd.Parameters.AddWithValue("@OLD\_BNAME", OLD\_name);

cmd.Parameters.AddWithValue("@OLD\_BADDR", OLD\_address);

cmd.Parameters.AddWithValue("@OLD\_LAT", OLD\_lat);

cmd.Parameters.AddWithValue("@OLD\_LON", OLD\_lon);

cmd.Parameters.AddWithValue("@OLD\_STATUS", OLD\_status);

cmd.Parameters.AddWithValue("@BNAME", name);

cmd.Parameters.AddWithValue("@BADDR", address);

cmd.Parameters.AddWithValue("@LAT", lat);

cmd.Parameters.AddWithValue("@LON", lon);

cmd.Parameters.AddWithValue("@STATUS", status);

return cmd.ExecuteNonQuery();

}

catch (Exception){return -1;}

finally{con.Close();}

}

}

public static List<Branch> GetSubBranch(int rid)

{

List<Branch> SubBs = new List<Branch>();

using (SqlConnection con = new SqlConnection(connectionstr))

{

con.Open();

SqlDataAdapter da = new SqlDataAdapter("GetSubBranchByRID", con);

da.SelectCommand.CommandType = CommandType.StoredProcedure;

da.SelectCommand.Parameters.AddWithValue("@RID", rid);

DataTable data = new DataTable("RegBranchs");

da.Fill(data);

foreach (DataRow row in data.Rows)

{

Branch b1 = new Branch();

b1.Bid = Convert.ToInt64(row.Field<decimal>("BID"));

b1.Rid = Convert.ToInt64(row.Field<decimal>("RID"));

b1.name = row.Field<string>("Bname");

b1.address = row.Field<string>("Baddress");

b1.lat = Convert.ToDouble(row.Field<decimal>("Latitude"));

b1.lon = Convert.ToDouble(row.Field<decimal>("Longitude"));

b1.status = row.Field<string>("Status");

SubBs.Add(b1);

}

}

return SubBs;

}

public static int UpdateSubBranches(long OLD\_Bid,string OLD\_name,

string OLD\_address, double OLD\_lat, double OLD\_lon, string OLD\_status,

string name, string address, double lat, double lon, string status)

{

using (con = new SqlConnection(connectionstr))

{

try

{

con.Open();

SqlCommand cmd = new SqlCommand("UpdateSubB", con);

cmd.CommandType = CommandType.StoredProcedure;

cmd.Parameters.AddWithValue("@OLD\_BID", OLD\_Bid);

cmd.Parameters.AddWithValue("@OLD\_BNAME", OLD\_name);

cmd.Parameters.AddWithValue("@OLD\_BADDR", OLD\_address);

cmd.Parameters.AddWithValue("@OLD\_LAT", OLD\_lat);

cmd.Parameters.AddWithValue("@OLD\_LON", OLD\_lon);

cmd.Parameters.AddWithValue("@OLD\_STATUS", OLD\_status);

cmd.Parameters.AddWithValue("@BNAME", name);

cmd.Parameters.AddWithValue("@BADDR", address);

cmd.Parameters.AddWithValue("@LAT", lat);

cmd.Parameters.AddWithValue("@LON", lon);

cmd.Parameters.AddWithValue("@STATUS", status);

return cmd.ExecuteNonQuery();

}

catch (Exception){return -1;}

finally{con.Close();}

}

}

public static string GetBranchNameByBID(long BID)

{

string name = null;

using (con = new SqlConnection(connectionstr))

{

try

{

con.Open();

SqlCommand cmd = new SqlCommand("select Bname from Branch where BID=@BID",

con);

cmd.Parameters.AddWithValue("@BID", BID);

name = Convert.ToString(cmd.ExecuteScalar());

}

catch (Exception){return null;}

}

return name;

}

}

}

1. Employee

using System;

using System.IO;

using System.Web;

using System.Data;

using System.Data.SqlClient;

using OnlineCourierService.Class;

namespace OnlineCourierService.employee.Classes

{

public class Employee : Password

{

private static readonly string connectionString = "Data Source=DESKTOP-74RBQ7M\\KUNTALSQLS;Initial Catalog=CourierService;Persist Security Info=True;User ID=sa;Password=!@Kd\_Hell45";

private SqlConnection con;

private SqlCommand cmd;

private SqlDataAdapter da;

public int Eid { private get; set; }

public long Bid { get; set; }

public string Bname { get; set; }

public long phno { get; set; }

public DateTime dob { get; set; }

public DateTime createdOn { get; protected set; }

public string name { get; set; }

public String address { get; set; }

public string gender { get; set; }

public string Email { get; set; }

public string post { get; set; }

public int postindex { get; set; }

public string password { protected get; set; }

public bool isAdmin { get; protected set; }

public bool isActive { get; protected set; }

public string imgurl { get; set; }

public Employee(int Eid)

{

this.Eid = Eid;

this.Bid = this.phno = -1;

this.name = this.address = this.Email = this.post = this.password = null;

this.isAdmin = false;

this.isActive = true;

con = new SqlConnection(connectionString);

}

public Employee(string Email)

{

this.Email = Email;

this.Eid = -1;

this.Bid = this.phno = -1;

this.name = this.address = this.post = this.password = null;

this.isAdmin = false;

this.isActive = true;

con = new SqlConnection(connectionString);

}

public Employee(long phno)

{

this.phno = phno;

this.Eid = -1;

this.Bid = -1;

this.name = this.address = this.Email = this.post = this.password = null;

this.isAdmin = false;

this.isActive = true;

con = new SqlConnection(connectionString);

}

public Employee(string Email, long phno)

{

this.phno = phno;

this.Email = Email;

this.Eid = -1;

this.Bid = -1;

this.name = this.address = this.post = this.password = null;

this.isAdmin = false;

this.isActive = true;

con = new SqlConnection(connectionString);

}

public Employee(string name, string address, string gender, string email, DateTime dob, long Bid, long phno, string password, int postindex, Boolean isActive)

{

this.Eid = -1;

this.post = null;

this.name = name;

this.address = address;

this.Email = email;

this.gender = gender;

this.password = password;

this.Bid = Bid;

this.phno = phno;

this.dob = dob;

this.postindex = postindex;

this.isActive = isActive;

this.isAdmin = isAdmin;

this.con = new SqlConnection(connectionString);

}

public static int GenerateResetPasswordLink(string Email, string link)

{

int success = -1;

using (SqlConnection con = new SqlConnection(connectionString))

{

try

{

con.Open();

SqlCommand cmd = new SqlCommand("ERequestResetPassword", con);

cmd.CommandType = CommandType.StoredProcedure;

cmd.Parameters.AddWithValue("@Email", Email);

SqlDataReader rdr = cmd.ExecuteReader();

while (rdr.Read())

{

if (Convert.ToBoolean(rdr["ReturnCode"]))

{

if (Password.SendPasswordResetEmail(Email, rdr["Name"].ToString(),

rdr["UniqueId"].ToString(), link))

{

success = 1;

}

}

else{success = -1;}

}

}

catch (SqlException) { return success; }

catch (Exception) { return success; }

finally { con.Close(); }

return success;

}

}

public static long isPasswordResetLinkValid(string Uid)

{

long Eid = -1;

using (SqlConnection con = new SqlConnection(connectionString))

{

try

{

con.Open();

SqlCommand cmd = new SqlCommand("EIsPasswordResetLinkValid", con);

cmd.CommandType = CommandType.StoredProcedure;

cmd.Parameters.AddWithValue("@GUID", Uid);

Eid = Convert.ToInt64(cmd.ExecuteScalar());

}

catch (SqlException) { return -1; }

catch (NullReferenceException) { return -1; }

catch (Exception) { return -1; }

finally { con.Close(); }

}

return Eid;

}

public static int ResetPassword(string guid, string pass)

{

int c = -1;

using (SqlConnection con = new SqlConnection(connectionString))

{

try

{

con.Open();

SqlCommand cmd = new SqlCommand("EResetPassword", con);

cmd.CommandType = CommandType.StoredProcedure;

cmd.Parameters.AddWithValue("@GUID", guid);

cmd.Parameters.AddWithValue("@Password", Password.HashPassword(pass));

c = Convert.ToInt32(cmd.ExecuteScalar());

}

catch (SqlException) { return c; }

catch (Exception) { return c; }

finally { con.Close(); }

}

return c;

}

private void OpenConnection()

{

if (con.State == ConnectionState.Closed)

{con.Open();}

}

private void CloseConnection()

{

if (con.State == ConnectionState.Open)

{con.Close();}

}

public static int CountByEmail(string email)

{

using (SqlConnection con1 = new SqlConnection(connectionString))

{

SqlCommand cmd1 = new SqlCommand("CountEbyEmPh", con1);

cmd1.CommandType = CommandType.StoredProcedure;

cmd1.Parameters.AddWithValue("@EMAIL", email);

cmd1.Parameters.AddWithValue("@PHNO", null);

try

{

con1.Open();

return Convert.ToInt32(cmd1.ExecuteScalar());

}

catch (Exception){return -1;}

finally{con1.Close();}

}

}

public static int CountByPhone(long phoneNo)

{

using (SqlConnection con1 = new SqlConnection(connectionString))

{

SqlCommand cmd1 = new SqlCommand("CountEbyEmPh", con1);

cmd1.CommandType = CommandType.StoredProcedure;

cmd1.Parameters.AddWithValue("@EMAIL", null);

cmd1.Parameters.AddWithValue("@PHNO", phoneNo);

try

{

con1.Open();

return Convert.ToInt32(cmd1.ExecuteScalar());

}

catch (Exception){return -1;}

finally{con1.Close();}

}

}

public int CheckPassword(string password)

{

if (password.Equals(null))

{return -1;}

this.cmd = new SqlCommand("GetEidByPass", con);

cmd.CommandType = CommandType.StoredProcedure;

if (this.Email != null)

{

cmd.Parameters.AddWithValue("@EMAIL", this.Email);

cmd.Parameters.AddWithValue("@PHNO", null);

}

else if (this.phno != -1)

{

cmd.Parameters.AddWithValue("@EMAIL", null);

cmd.Parameters.AddWithValue("@PHNO", this.phno);

}

cmd.Parameters.AddWithValue("@PASS", Password.HashPassword(password));

OpenConnection();

try

{

Eid = Convert.ToInt32(cmd.ExecuteScalar());

}

catch (IndexOutOfRangeException){return -1;}

finally{CloseConnection();}

return Eid;

}

public int CountbyEmailorphno()

{

int count = -1;

this.da = new SqlDataAdapter();

this.da.SelectCommand = new SqlCommand("CountEbyEmPh", con);

this.da.SelectCommand.CommandType = CommandType.StoredProcedure;

if (this.Email != null)

{

this.da.SelectCommand.Parameters.AddWithValue("@EMAIL", this.Email);

this.da.SelectCommand.Parameters.AddWithValue("@PHNO", null);

}

else if (this.phno != -1)

{

this.da.SelectCommand.Parameters.AddWithValue("@EMAIL", null);

this.da.SelectCommand.Parameters.AddWithValue("@PHNO", this.phno);

}

DataTable dt = new DataTable();

this.da.Fill(dt);

OpenConnection();

count = dt.Rows[0].Field<int>("count");

CloseConnection();

return count;

}

public Boolean IsAdmin()

{

if (this.Eid < 1000) {return false;}

this.cmd = new SqlCommand("isAdmin", con);

this.cmd.CommandType = CommandType.StoredProcedure;

this.cmd.Parameters.AddWithValue("@EID", this.Eid);

OpenConnection();

int x = Convert.ToInt32(cmd.ExecuteScalar());

this.isAdmin = (x == 1 || x == 2);

CloseConnection();

return isAdmin;

}

public Boolean IsActive(int Eid)

{

if (Eid < 1000) {return false;}

this.cmd = new SqlCommand("isActiveByEid", con);

this.cmd.CommandType = CommandType.StoredProcedure;

this.cmd.Parameters.AddWithValue("@EID", Eid);

OpenConnection();

isAdmin = (Convert.ToInt32(cmd.ExecuteScalar()).ToString().Equals("Active"));

CloseConnection();

return isActive;

}

public DataTable GetNotifications()

{

if (this.Eid < 1000) {return null;}

this.da = new SqlDataAdapter();

this.da.SelectCommand = new SqlCommand("GetNotificationByEid", con);

this.da.SelectCommand.CommandType = CommandType.StoredProcedure;

this.da.SelectCommand.Parameters.AddWithValue("@EID", this.Eid);

DataTable dttbl = new DataTable();

OpenConnection();

this.da.Fill(dttbl);

CloseConnection();

return dttbl;

}

public int GetImgCount()

{

using (con)

{

this.cmd = new SqlCommand("CountImageByEid", con);

cmd.CommandType = CommandType.StoredProcedure;

cmd.Parameters.AddWithValue("@EID", this.Eid);

try

{

OpenConnection();

return Convert.ToInt32(cmd.ExecuteScalar());

}

catch (Exception){return -1;}

finally{CloseConnection();}

}

}

public int InsertImage(HttpPostedFile img)

{

string fileName = Path.GetFileName(img.FileName);

string fileExtennsion = Path.GetExtension(fileName).ToLower();

int fileSize = img.ContentLength;

if (fileExtennsion == ".jpg" || fileExtennsion == ".jpeg" || fileExtennsion == ".png")

{

Stream imgstream = img.InputStream;

BinaryReader imgbin = new BinaryReader(imgstream);

byte[] imgBytes = imgbin.ReadBytes((int)imgstream.Length);

using (con)

{

cmd = new SqlCommand("InsertImagebyEid", con);

cmd.CommandType = CommandType.StoredProcedure;

cmd.Parameters.AddWithValue("@EID", this.Eid);

cmd.Parameters.AddWithValue("@NAME", fileName);

cmd.Parameters.AddWithValue("@SIZE", fileSize);

cmd.Parameters.AddWithValue("@IMGDATA", imgBytes);

try

{

OpenConnection();

cmd.ExecuteNonQuery();

return 1;

}

catch (Exception){return -1;}

finally{CloseConnection();}

}

}

else{return -1;}

}

public int UpdateImage(HttpPostedFile img)

{

string fileName = Path.GetFileName(img.FileName);

string fileExtennsion = Path.GetExtension(fileName).ToLower();

int fileSize = img.ContentLength;

if (fileExtennsion == ".jpg" || fileExtennsion == ".jpeg" || fileExtennsion == ".png")

{

Stream imgstream = img.InputStream;

BinaryReader imgbin = new BinaryReader(imgstream);

byte[] imgBytes = imgbin.ReadBytes((int)imgstream.Length);

using (con = new SqlConnection(connectionString)){

cmd = new SqlCommand("UpdateImagebyEid", con);

cmd.CommandType = CommandType.StoredProcedure;

cmd.Parameters.AddWithValue("@EID", this.Eid);

cmd.Parameters.AddWithValue("@NAME", fileName);

cmd.Parameters.AddWithValue("@SIZE", fileSize);

cmd.Parameters.AddWithValue("@IMGDATA", imgBytes);

try

{

OpenConnection();

cmd.ExecuteNonQuery();

return 1;

}

catch (Exception){return -1;}

finally{CloseConnection();}

}

}

else{return -1;}

}

public string LoadImage()

{

try

{

con = new SqlConnection(connectionString);

this.cmd = new SqlCommand("GetimageByEID", con);

cmd.CommandType = CommandType.StoredProcedure;

cmd.Parameters.AddWithValue("@EID", this.Eid);

OpenConnection();

byte[] imgbytes = (byte[])cmd.ExecuteScalar();

string imgstrbase64 = Convert.ToBase64String(imgbytes);

if (imgstrbase64 != null)

{

imgurl = ("data:Image/png;base64," + imgstrbase64);

return imgurl;

}

return null;

}

catch (NullReferenceException){return null;}

catch (Exception){return null;}

finally{CloseConnection();}

}

public int UpDateAll()

{

using (con = new SqlConnection(connectionString))

{

try

{

OpenConnection();

cmd = new SqlCommand("UpdateInfoByEid", con);

cmd.CommandType = CommandType.StoredProcedure;

cmd.Parameters.AddWithValue("@EID", Eid);

cmd.Parameters.AddWithValue("@BID", Bid);

cmd.Parameters.AddWithValue("@NAME", name);

cmd.Parameters.AddWithValue("@DOB", dob);

cmd.Parameters.AddWithValue("@GENDER", gender);

cmd.Parameters.AddWithValue("@ADDRESS", address);

cmd.Parameters.AddWithValue("@EMAIL", Email);

cmd.Parameters.AddWithValue("@PHNO", phno);

cmd.Parameters.AddWithValue("@POST", postindex);

cmd.Parameters.AddWithValue("@PASS", Password.HashPassword(password));

cmd.Parameters.AddWithValue("@ACTIVE", (isActive) ? 1 : -1);

cmd.ExecuteNonQuery();

return 1;

}

catch (Exception){return -1;}

finally{CloseConnection();}

}

}

public int InsertInfo()

{

using (con = new SqlConnection(connectionString))

{

try

{

OpenConnection();

cmd = new SqlCommand("InsertInfoByEid", con);

cmd.CommandType = CommandType.StoredProcedure;

cmd.Parameters.AddWithValue("@BID", Bid);

cmd.Parameters.AddWithValue("@NAME", name);

cmd.Parameters.AddWithValue("@DOB", dob);

cmd.Parameters.AddWithValue("@GENDER", gender);

cmd.Parameters.AddWithValue("@ADDRESS", address);

cmd.Parameters.AddWithValue("@EMAIL", Email);

cmd.Parameters.AddWithValue("@PHNO", phno);

cmd.Parameters.AddWithValue("@POST", postindex);

cmd.Parameters.AddWithValue("@PASS", Password.HashPassword(password));

cmd.Parameters.AddWithValue("@ACTIVE", (isActive) ? "Active" : "Inactive");

SqlParameter EID = new SqlParameter()

{

ParameterName = "@EID",

Value = -1,

Direction = ParameterDirection.Output

};

cmd.Parameters.Add(EID);

cmd.ExecuteNonQuery();

this.Eid = Convert.ToInt32(cmd.Parameters["@EID"].Value);

return this.Eid;

}

catch (Exception){return -1;}

finally{CloseConnection();}

}

}

private DataTable DownSync()

{

try

{

this.da = new SqlDataAdapter();

this.da.SelectCommand = new SqlCommand("DownloadInfoByEID", con);

this.da.SelectCommand.CommandType = CommandType.StoredProcedure;

if (this.Eid >= 1000)

{

this.da.SelectCommand.Parameters.AddWithValue("@EID", this.Eid);

this.da.SelectCommand.Parameters.AddWithValue("@EMAIL", null);

this.da.SelectCommand.Parameters.AddWithValue("@PHNO", null);

}

else if (this.Email != null)

{

this.da.SelectCommand.Parameters.AddWithValue("@EID", null);

this.da.SelectCommand.Parameters.AddWithValue("@EMAIL", this.Email);

this.da.SelectCommand.Parameters.AddWithValue("@PHNO", null);

}

else if ((this.phno != -1) && (this.phno.ToString().Length >= 10))

{

this.da.SelectCommand.Parameters.AddWithValue("@EID", null);

this.da.SelectCommand.Parameters.AddWithValue("@EMAIL", null);

this.da.SelectCommand.Parameters.AddWithValue("@PHNO", this.phno);

}

DataTable dttbl = new DataTable();

OpenConnection();

this.da.Fill(dttbl);

return dttbl;

}

catch (SqlException){return null;}

finally{CloseConnection();}

}

public int Filldata()

{

DataTable data = this.DownSync();

if (data.Equals(null)){return -1;}

try

{

foreach (DataRow row in data.Rows)

{

this.Eid = Convert.ToInt32(row.Field<decimal>("EID"));

this.Bid = Convert.ToInt64(row.Field<decimal>("BID"));

this.name = row.Field<string>("NAME");

this.address = row.Field<string>("ADDRESS");

this.gender = row.Field<string>("GENDER");

this.Email = row.Field<string>("EMAIL");

this.postindex = row.Field<int>("PINDEX");

this.post = row.Field<string>("POST");

this.phno = Convert.ToInt64(row.Field<decimal>("PHNO"));

this.dob = row.Field<DateTime>("DOB");

this.createdOn = row.Field<DateTime>("CTS");

this.isAdmin = (row.Field<int>("PINDEX") == 1 || row.Field<int>("ADMIN") == 2);

this.isActive = (row.Field<string>("ACTIVE").Equals("Active"));

}

}

catch (Exception){return -1;}

return 1;

}

public static int UpdateEmpbyAdmin(long OLD\_Eid,

long OLD\_Bid, int OLD\_postindex, string OLD\_status,

long Bid, int postindex, string status)

{

using (SqlConnection con = new SqlConnection(connectionString))

{

try

{

con.Open();

SqlCommand cmd = new SqlCommand("UpdateEmpByAdmin", con);

cmd.CommandType = CommandType.StoredProcedure;

cmd.Parameters.AddWithValue("@OLD\_EID", OLD\_Eid);

cmd.Parameters.AddWithValue("@OLD\_BID", OLD\_Bid);

cmd.Parameters.AddWithValue("@BID", Bid);

cmd.Parameters.AddWithValue("@OLD\_POSTINDEX", OLD\_postindex);

cmd.Parameters.AddWithValue("@POSTINDEX", postindex);

cmd.Parameters.AddWithValue("@OLD\_STATUS", OLD\_status);

cmd.Parameters.AddWithValue("@STATUS", status);

return cmd.ExecuteNonQuery();

}

catch (SqlException){return -1;}

finally{con.Close();}

}

}

public static DataTable GetEmpinfoByAdmin(long Eid)

{

using (SqlConnection con = new SqlConnection(connectionString))

{

try

{

con.Open();

SqlDataAdapter Da = new SqlDataAdapter("GetEmpByEid", con);

Da.SelectCommand.CommandType = CommandType.StoredProcedure;

Da.SelectCommand.Parameters.AddWithValue("@EID", Eid);

DataTable data = new DataTable("Employee");

Da.Fill(data);

return data;

}

catch (SqlException){return null;}

finally{con.Close();}

}

}

public static DataTable GetEmpinfoByAdmin2(long bid, int postindex)

{

using (SqlConnection con = new SqlConnection(connectionString))

{

try

{

con.Open();

SqlDataAdapter Da = new SqlDataAdapter("GetEmpByDept\_Branch", con);

Da.SelectCommand.CommandType = CommandType.StoredProcedure;

Da.SelectCommand.Parameters.AddWithValue("@BID", bid);

Da.SelectCommand.Parameters.AddWithValue("@POSTINDEX", postindex);

DataTable data = new DataTable("Employee");

Da.Fill(data);

return data;

}

catch (SqlException){return null;}

finally{con.Close();}

}

}

}

}

1. DataAccessLayer

using System;

using System.Collections.Generic;

using System.Data;

using System.Data.SqlClient;

namespace OnlineCourierService.employee.Classes

{

public class Rates

{

public int RateID { get; set; }

public double Weight { get; set; }

public double Basic { get; set; }

public double Business { get; set; }

public double DryFood { get; set; }

public double CookedFood { get; set; }

public double Gift { get; set; }

public double Electronics { get; set; }

public Rates() { }

public Rates(int ID, double weight, double basic, double business, double dryf, double cookedf, double gift, double elec){

this.RateID = ID;

this.Weight = weight;

this.Basic = basic;

this.Business = business;

this.DryFood = dryf;

this.CookedFood = cookedf;

this.Gift = gift;

this.Electronics = elec;

}

}

public class Region

{

public int ID { get; set; }

public string Name { get; set; }

public Region(int id, string name){

this.ID = id;

this.Name = name;

}

}

public class Branch1

{

public int Bid { get; set; }

public string BName { get; set; }

public Branch1(int bid, string name){

this.Bid = bid;

this.BName = (bid != -1) ? (name + " (" + bid + ")") : name;

}

}

public class Dept

{

public int DeptId { set; get; }

public string DeptName { set; get; }

public Dept(int DID, string Dname){

this.DeptId = DID;

this.DeptName = Dname;

}

}

public class Employee1

{

public int Eid { set; get; }

public string name { set; get; }

public string gender { set; get; }

public DateTime dob { set; get; }

public long Bid { set; get; }

public long Rid { set; get; }

public string Bname { set; get; }

public int postindex { set; get; }

public string post { set; get; }

public string status { set; get; }

}

public class DataAccessLayer

{

private static readonly string connectionstr = "Data Source=DESKTOP-74RBQ7M\\KUNTALSQLS;Initial Catalog=CourierService;Persist Security Info=True;User ID=sa;Password=!@Kd\_Hell45";

public static List<Region> GetRegion()

{

List<Region> r = new List<Region>();

r.Add(new Region(-1, "Select Region"));

DataTable data = new DataTable("Regions");

using (SqlConnection con = new SqlConnection(connectionstr)){

try{

con.Open();

SqlDataAdapter Da = new SqlDataAdapter("GetRegions", con);

Da.SelectCommand.CommandType = CommandType.StoredProcedure;

Da.Fill(data);

foreach (DataRow row in data.Rows){

r.Add(new Region(row.Field<int>("RegionID"), row.Field<string>("Region")));

}

}catch (SqlException){return null;}

finally{con.Close();}

}

return r;

}

public static List<Branch1> GetBranchByRid(int Rid)

{

List<Branch1> b1 = new List<Branch1>();

b1.Add(new Branch1(-1, "Select Branch"));

DataTable data = new DataTable("Branchs");

using (SqlConnection con = new SqlConnection(connectionstr))

{

try

{

con.Open();

SqlDataAdapter Da = new SqlDataAdapter("GetBranchlist", con);

Da.SelectCommand.CommandType = CommandType.StoredProcedure;

Da.SelectCommand.Parameters.AddWithValue("@RID", Rid);

Da.Fill(data);

foreach (DataRow row in data.Rows){

b1.Add(new Branch1(Convert.ToInt32(row.Field<decimal>("BID")), row.Field<string>("Bname")));

}

}

catch (SqlException){return null;}

finally{con.Close();}

}

return b1;

}

public static List<Branch1> GetRegBranch()

{

List<Branch1> b1 = new List<Branch1>();

b1.Add(new Branch1(-1, "Select Regional Branch"));

DataTable data = new DataTable("RegBranch");

using (SqlConnection con = new SqlConnection(connectionstr))

{

try

{

con.Open();

SqlDataAdapter Da = new SqlDataAdapter("GetRegBranchlist", con);

Da.SelectCommand.CommandType = CommandType.StoredProcedure;

Da.Fill(data);

foreach (DataRow row in data.Rows)

{

b1.Add(new Branch1(Convert.ToInt32(row.Field<decimal>("BID")), row.Field<string>("Bname")));

}

}

catch (SqlException){return null;}

finally{con.Close();}

}

return b1;

}

public static List<Dept> GetDepartments()

{

List<Dept> depts = new List<Dept>();

depts.Add(new Dept(-1, "Select Post"));

DataTable data = new DataTable("Departments");

using (SqlConnection con = new SqlConnection(connectionstr))

{

try

{

con.Open();

SqlDataAdapter Da = new SqlDataAdapter("GetDept", con);

Da.SelectCommand.CommandType = CommandType.StoredProcedure;

Da.Fill(data);

foreach (DataRow row in data.Rows)

{

depts.Add(new Dept(row.Field<int>("DeptID"), row.Field<string>("DeptName")));

}

}

catch (SqlException){return null;}

finally{con.Close();}

}

return depts;

}

public static List<Employee1> GetEmpinfoByBID\_Dept(int bid, int did)

{

List<Employee1> emps = new List<Employee1>();

using (SqlConnection con = new SqlConnection(connectionstr))

{

try

{

con.Open();

SqlDataAdapter Da = new SqlDataAdapter("GetEmpByDept\_Branch", con);

Da.SelectCommand.CommandType = CommandType.StoredProcedure;

Da.SelectCommand.Parameters.AddWithValue("@BID", bid);

Da.SelectCommand.Parameters.AddWithValue("@DEPTID", did);

DataTable data = new DataTable("Employees");

Da.Fill(data);

foreach (DataRow row in data.Rows)

{

Employee1 emp = new Employee1();

emp.Eid = Convert.ToInt32(row.Field<decimal>("EID"));

emp.name = row.Field<string>("NAME");

emp.dob = row.Field<DateTime>("DOB");

emp.Bid = Convert.ToInt64(row.Field<decimal>("BRANCH"));

emp.Bname = row.Field<string>("BNAME");

emp.post = row.Field<string>("POST");

emp.postindex = row.Field<int>("POSTINDEX");

emp.gender = row.Field<string>("GENDER");

emp.status = (row.Field<int>("STATUS") == 1) ? "Active" : "Inactive";

emps.Add(emp);

}

}

catch (SqlException){return null;}

finally{con.Close();}

}

return emps;

}

public static List<Rates> GetRates()

{

List<Rates> rates = new List<Rates>();

using (SqlConnection con = new SqlConnection(connectionstr))

{

try{

con.Open();

SqlDataAdapter Da = new SqlDataAdapter("GetRates", con);

Da.SelectCommand.CommandType = CommandType.StoredProcedure;

DataTable data = new DataTable("Rates");

Da.Fill(data);

foreach (DataRow row in data.Rows)

{

Rates r = new Rates();

r.RateID = row.Field<int>("RateID");

r.Weight = row.Field<double>("WEIGHT");

r.Basic = row.Field<double>("Basic");

r.Business = row.Field<double>("Business");

r.DryFood = row.Field<double>("DryFood");

r.CookedFood = row.Field<double>("CookedFood");

r.Gift = row.Field<double>("Gift");

r.Electronics = row.Field<double>("Electronics");

rates.Add(r);

}

}

catch (SqlException){return null;}

finally{con.Close();}

}

return rates;

}

public static int UpdateRates(int OLD\_RateID,

int OLD\_Weight, double OLD\_Basic, double OLD\_Business, double OLD\_DryFood,

double OLD\_CookedFood, double OLD\_Gift, double OLD\_Electronics,

int Weight, double Basic, double Business, double DryFood, double CookedFood,

double Gift, double Electronics)

{

if (CountRateWeight(Weight) == 0)

{

using (SqlConnection con = new SqlConnection(connectionstr))

{

try

{

con.Open();

SqlCommand cmd = new SqlCommand("UpdateRate", con);

cmd.CommandType = CommandType.StoredProcedure;

cmd.Parameters.AddWithValue("@RATEID", OLD\_RateID);

cmd.Parameters.AddWithValue("@O\_WEIGHT", OLD\_Weight);

cmd.Parameters.AddWithValue("@O\_BASIC", OLD\_Basic);

cmd.Parameters.AddWithValue("@O\_BUSINESS", OLD\_Business);

cmd.Parameters.AddWithValue("@O\_DF", OLD\_DryFood);

cmd.Parameters.AddWithValue("@O\_CF", OLD\_CookedFood);

cmd.Parameters.AddWithValue("@O\_GIFT", OLD\_Gift);

cmd.Parameters.AddWithValue("@O\_ELEC", OLD\_Electronics);

cmd.Parameters.AddWithValue("@WEIGHT", Weight);

cmd.Parameters.AddWithValue("@BASIC", Basic);

cmd.Parameters.AddWithValue("@BUSINESS", Business);

cmd.Parameters.AddWithValue("@DF", DryFood);

cmd.Parameters.AddWithValue("@CF", CookedFood);

cmd.Parameters.AddWithValue("@GIFT", Gift);

cmd.Parameters.AddWithValue("@ELEC", Electronics);

return cmd.ExecuteNonQuery();

}

catch (SqlException){return -1;}

catch (Exception){return -1;}

finally{con.Close();}

}

}

else{return -1;}

}

public static int CountRateWeight(int weight)

{

int count = -1;

using (SqlConnection con = new SqlConnection(connectionstr))

{

try

{

con.Open();

SqlCommand cmd = new SqlCommand("CountWeight", con);

cmd.CommandType = CommandType.StoredProcedure;

cmd.Parameters.AddWithValue("@weight", weight);

count = Convert.ToInt32(cmd.ExecuteScalar());

}

catch (SqlException){return count;}

catch (Exception){return count;}

finally{con.Close();}

return count;

}

}

public static int InsertRates(int Weight, double Basic, double Business, double DryFood, double CookedFood, double Gift, double Electronics)

{

if (CountRateWeight(Weight) == 0)

{

using (SqlConnection con = new SqlConnection(connectionstr))

{

try

{

con.Open();

SqlCommand cmd = new SqlCommand("InsertRates", con);

cmd.CommandType = CommandType.StoredProcedure;

cmd.Parameters.AddWithValue("@WEIGHT", Weight);

cmd.Parameters.AddWithValue("@BASIC", Basic);

cmd.Parameters.AddWithValue("@BUSINESS", Business);

cmd.Parameters.AddWithValue("@DRYFOOD", DryFood);

cmd.Parameters.AddWithValue("@COOKEDFOOD", CookedFood);

cmd.Parameters.AddWithValue("@GIFT", Gift);

cmd.Parameters.AddWithValue("@ELEC", Electronics);

SqlParameter rateID = new SqlParameter()

{

ParameterName = "@RATEID",

Value = -1,

Direction = ParameterDirection.Output

};

cmd.Parameters.Add(rateID);

cmd.ExecuteNonQuery();

return Convert.ToInt32(cmd.Parameters["@RATEID"].Value);

}

catch (SqlException){return -1;}

catch (Exception){return -1;}

finally{con.Close();}

}

}

else{return -1;}

}

public static List<Region> GetRegionWithRegBranchID()

{

List<Region> r = new List<Region>();

r.Add(new Region(-1, "Select Region"));

DataTable data = new DataTable("Regions");

using (SqlConnection con = new SqlConnection(connectionstr))

{

try

{

con.Open();

SqlDataAdapter Da = new SqlDataAdapter("GetRegionsWithRID", con);

Da.SelectCommand.CommandType = CommandType.StoredProcedure;

Da.Fill(data);

foreach (DataRow row in data.Rows)

{

r.Add(new Region(Convert.ToInt32(row.Field<decimal>("RBranchID")), row.Field<string>("Region")));

}

}

catch (SqlException){return null;}

finally{con.Close();}

}

return r;

}

}

}

1. **Dynamic Web Pages**
2. Master Page(index.Master)

<%@ Master Language="C#" AutoEventWireup="true" CodeBehind="index.master.cs" Inherits="OnlineCourierService.index1" %>

<!DOCTYPE html>

<html>

<head runat="server">

<title>Lightning Corp.</title>

<meta name="viewport" content="width=device-width, initial-scale=0.86, maximum-scale=3.0, minimum-scale=0.86" />

<link rel="shortcut icon" href="lightning.ico">

<link href="css/index-master.css" rel="stylesheet" />

<script src="js/index-master.js"></script>

<asp:ContentPlaceHolder ID="head" runat="server">

</asp:ContentPlaceHolder>

</head>

<body>

<form id="form1" runat="server">

<section class="page">

<div id="loaderImage" class="preloader">

<img id="loader" src="svg/index.scratching-reveal-loader.svg" alt="LOGO">

</div>

<asp:ContentPlaceHolder ID="CPHerrMsg" runat="server"></asp:ContentPlaceHolder>

<nav class="navi">

<asp:HyperLink ID="HyperLink21" CssClass="logo proj-logo" NavigateUrl="~/index.aspx" runat="server"></asp:HyperLink>

<ul class="menu">

<li class="level1nav"><asp:HyperLink ID="HyperLink24" CssClass="level1nav\_\_link t1" runat="server">Shipping&blacktriangledown;</asp:HyperLink>

<ul class="dropdown UL1">

<li class="level2nav"><asp:HyperLink ID="HyperLink1" CssClass="level2nav\_\_link" NavigateUrl="~/Parcel/CreateParcel.aspx" runat="server">Create a Shipment</asp:HyperLink></li>

<li class="level2nav">asp:HyperLink ID="HyperLink3" CssClass="level2nav\_\_link" NavigateUrl="~/Parcel/RateofParcel.aspx" runat="server">Shipping Rates</asp:HyperLink></li>

<li class="level2nav"><asp:HyperLink ID="HyperLink6" CssClass="level2nav\_\_link" NavigateUrl="#" runat="server">Returns</asp:HyperLink></li>

<li class="level2nav"><asp:HyperLink ID="HyperLink8" CssClass="level2nav\_\_link" NavigateUrl="#" runat="server">Learn More about Online Shipping</asp:HyperLink></li>

<li class="level2nav"><asp:HyperLink ID="HyperLink9" CssClass="level2nav\_\_link" NavigateUrl="#" runat="server">All Shiping Services</asp:HyperLink></li>

</ul>

</li>

<li class="level1nav"><asp:HyperLink ID="HyperLink23" CssClass="level1nav\_\_link t2" runat="server">Tracking&blacktriangledown;</asp:HyperLink>

<ul class="dropdown UL2">

<li class="level2nav">asp:HyperLink ID="HyperLink10" CssClass="level2nav\_\_link" NavigateUrl="~/Parcel/ParcelStat.aspx" runat="server">Advance Shipment Tracking</asp:HyperLink>

</li>

<li class="level2nav"><asp:HyperLink ID="HyperLink12" CssClass="level2nav\_\_link" NavigateUrl="#" runat="server">Track by Email</asp:HyperLink></li>

<li class="level2nav"><asp:HyperLink ID="HyperLink2" runat="server" CssClass="level2nav\_\_link" NavigateUrl="~/Parcel/SrcParcel.aspx">All Tracking Services</asp:HyperLink></li>

</ul>

</li>

<li class="level1nav"><asp:HyperLink ID="HyperLink22" CssClass="level1nav\_\_link t3" runat="server">Support&blacktriangledown;</asp:HyperLink>

<ul class="dropdown UL3">

<li class="level2nav"><asp:HyperLink ID="HyperLink16" CssClass="level2nav\_\_link" NavigateUrl="#" runat="server">File a Claim</asp:HyperLink></li>

<li class="level2nav"><asp:HyperLink ID="HyperLink18" CssClass="level2nav\_\_link" NavigateUrl="#" runat="server">Billing/Invoice Forms</asp:HyperLink></li>

<li class="level2nav"><asp:HyperLink ID="HyperLink4" runat="server" CssClass="level2nav\_\_link" NavigateUrl="#">Find Our Store Locations</asp:HyperLink></li>

</ul>

</li>

</ul>

<asp:ContentPlaceHolder ID="CPHuser" runat="server"></asp:ContentPlaceHolder>

</nav>

<header class="head1">

<h1 class="head1\_\_heading">

<asp:ContentPlaceHolder ID="CPHheader" runat="server"></asp:ContentPlaceHolder>

</h1>

</header>

<div class="container">

<asp:ContentPlaceHolder ID="CPHcontent" runat="server"></asp:ContentPlaceHolder>

</div>

<footer class="foot">

<div class="foot\_\_div about">

<h3>About</h3>

<ul>

<li><asp:HyperLink ID="HyperLink32" CssClass="foot\_\_link" NavigateUrl="#" runat="server">About Us </asp:HyperLink></li>

<li><asp:HyperLink ID="HyperLink33" CssClass="foot\_\_link" NavigateUrl="#" runat="server">Contact Us</asp:HyperLink></li>

<li><asp:HyperLink ID="HyperLink34" CssClass="foot\_\_link" NavigateUrl="#" runat="server">Our Store Locations</asp:HyperLink></li>

</ul>

</div>

<div class="foot\_\_div help">

<h3>Help</h3>

<ul>

<li><asp:HyperLink ID="HyperLink35" CssClass="foot\_\_link" NavigateUrl="#" runat="server">Payments</asp:HyperLink></li>

<li><asp:HyperLink ID="HyperLink36" CssClass="foot\_\_link" NavigateUrl="#" runat="server">Shipping</asp:HyperLink></li>

<li><asp:HyperLink ID="HyperLink37" CssClass="foot\_\_link" NavigateUrl="#" runat="server">International Holiday Schedule</asp:HyperLink></li>

<li><asp:HyperLink ID="HyperLink38" CssClass="foot\_\_link" NavigateUrl="#" runat="server">Cancellation</asp:HyperLink></li>

<li><asp:HyperLink ID="HyperLink39" CssClass="foot\_\_link" NavigateUrl="#" runat="server">FAQ</asp:HyperLink></li>

<li><asp:HyperLink ID="HyperLink40" CssClass="foot\_\_link" NavigateUrl="#" runat="server">Report Infringement</asp:HyperLink></li>

</ul>

</div>

<div class="foot\_\_div policy">

<h3>Policy</h3>

<ul>

<li><asp:HyperLink ID="HyperLink41" CssClass="foot\_\_link" NavigateUrl="#" runat="server">Cancellation Policy</asp:HyperLink></li>

<li><asp:HyperLink ID="HyperLink42" CssClass="foot\_\_link" NavigateUrl="#" runat="server">Terms of Service</asp:HyperLink></li>

<li><asp:HyperLink ID="HyperLink43" CssClass="foot\_\_link" NavigateUrl="#" runat="server">Security</asp:HyperLink></li>

<li><asp:HyperLink ID="HyperLink44" CssClass="foot\_\_link" NavigateUrl="#" runat="server">Privacy</asp:HyperLink></li>

<li><asp:HyperLink ID="HyperLink45" CssClass="foot\_\_link" NavigateUrl="#" runat="server">Conditions Of Carriage</asp:HyperLink></li>

</ul>

</div>

<div class="foot\_\_div region">

<h3>Region: </h3>

<select><option value="India">India</option></select>

</div>

</footer>

<div class="footer\_\_bottom" id="bottom">

<div class="social">

<span class="social\_\_follow">Follow Us on : </span>

<a href="#" class="social\_\_link facebook-icon">

<svg role="presentation"

class="icon facebook--icon">

<use xlink:href="#facebookicon" />

</svg></a>

<a href="#" class="social\_\_link twitter-icon">

<svg role="presentation"

class="icon twitter--icon">

<use xlink:href="#twittericon" />

</svg></a>

<a href="#" class="social\_\_link linkedin-icon">

<svg role="presentation"

class="icon linkedin--icon">

<use xlink:href="#linkedinicon" />

</svg></a>

</div>

<div class="bottom">

<a href="#" class="bottom\_\_link">Sitemap</a> <span>&copy;2019 - 2021 Lightning Corp.</span>

</div>

</div>

</section>

<svg display="none">

<symbol id="facebookicon" role="presentation" xmlns="http://www.w3.org/2000/svg" viewBox="0 0 320 512">

<path fill="currentColor"

d="M279.14 288l14.22-92.66h-88.91v-60.13c0-25.35 12.42-50.06 52.24-50.06h40.42V6.26S260.43 0 225.36 0c-73.22 0-121.08 44.38-121.08 124.72v70.62H22.89V288h81.39v224h100.17V288z" />

</symbol>

<symbol id="twittericon" role="presentation" xmlns="http://www.w3.org/2000/svg" viewBox="0 0 512 512">

<path fill="currentColor" d="M459.37 151.716c.325 4.548.325 9.097.325 13.645 0 138.72-105.583 298.558-298.558 298.558-59.452 0-114.68-17.219-161.137-47.106 8.447.974 16.568 1.299 25.34 1.299 49.055 0 94.213-16.568 130.274-44.832-46.132-.975-84.792-31.188-98.112-72.772 6.498.974 12.995 1.624 19.818 1.624 9.421 0 18.843-1.3 27.614-3.573-48.081-9.747-84.143-51.98-84.143-102.985v-1.299c13.969 7.797 30.214 12.67 47.431 13.319-28.264-18.843-46.781-51.005-46.781-87.391 0-19.492 5.197-37.36 14.294-52.954 51.655 63.675 129.3 105.258 216.365 109.807-1.624-7.797-2.599-15.918-2.599-24.04 0-57.828 46.782-104.934 104.934-104.934 30.213 0 57.502 12.67 76.67 33.137 23.715-4.548 46.456-13.32 66.599-25.34-7.798 24.366-24.366 44.833-46.132 57.827 21.117-2.273 41.584-8.122 60.426-16.243-14.292 20.791-32.161 39.308-52.628 54.253z" />

</symbol>

<symbol id="linkedinicon" role="presentation" xmlns="http://www.w3.org/2000/svg" viewBox="0 0 448 512">

<path fill="currentColor" d="M416 32H31.9C14.3 32 0 46.5 0 64.3v383.4C0 465.5 14.3 480 31.9 480H416c17.6 0 32-14.5 32-32.3V64.3c0-17.8-14.4-32.3-32-32.3zM135.4 416H69V202.2h66.5V416zm-33.2-243c-21.3 0-38.5-17.3-38.5-38.5S80.9 96 102.2 96c21.2 0 38.5 17.3 38.5 38.5 0 21.3-17.2 38.5-38.5 38.5zm282.1 243h-66.4V312c0-24.8-.5-56.7-34.5-56.7-34.6 0-39.9 27-39.9 54.9V416h-66.4V202.2h63.7v29.2h.9c8.9-16.8 30.6-34.5 62.9-34.5 67.2 0 79.7 44.3 79.7 101.9V416z" />

</symbol>

</svg>

<script src="js/libraries/TweenMax.min.js"></script>

<script src="js/libraries/jquery-3.4.1.min.js"></script>

<script>

$('a[href\*="#"]').not('[href="#"]').not('[href="#0"]').click(function (event) {

if (location.pathname.replace(/^\//, '') == this.pathname.replace(/^\//, '')

&& location.hostname == this.hostname) {

var target = $(this.hash);

target = target.length ? target : $('[name=' + this.hash.slice(1) + ']');

if (target.length) {

event.preventDefault();

$('html, body').animate({scrollTop: target.offset().top}, 1000, function () {

var $target = $(target);

$target.focus();

if ($target.is(":focus")) {return false;}

else {

$target.attr('tabindex', '-1');

$target.focus();

};

});

}

}

});

</script>

</form>

</body>

</html>

**Customer Side**

1. Home Page(index.aspx)

<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="index.aspx.cs" Inherits="OnlineCourierService.index" %>

<!DOCTYPE html>

<html xmlns="http://www.w3.org/1999/xhtml">

<head runat="server">

<meta name="viewport" content="width=device-width, initial-scale=0.86, maximum-scale=3.0, minimum-scale=0.86" />

<link rel="shortcut icon" href="lightning.ico" />

<title>Lightning Corp.</title>

<link href="css/index.css" rel="stylesheet" />

</head>

<body>

<form id="form1" runat="server">

<section class="page">

<div id="loaderImage" class="preloader">

<img id="loader" src="svg/index.scratching-reveal-loader.svg" alt="LOGO">

</div>

<asp:Panel ID="PError" runat="server" CssClass="PError">

<div class="cross"></div>

<asp:Label ID="Lerr" runat="server" Text="" CssClass="Lerr"></asp:Label>

<script>

document.querySelector(".cross").addEventListener("click", \_ => {

document.querySelector(".PError").style.opacity = 0;

document.querySelector(".PError").style.pointerEvents = "none";

});

</script>

</asp:Panel>

<nav class="navi">

<a href="index.aspx" class="logo proj-logo"></a>

<ul class="menu">

<li class="level1nav">

<asp:HyperLink ID="HyperLink24" CssClass="level1nav\_\_link t1" runat="server">Shipping&blacktriangledown;</asp:HyperLink>

<ul class="dropdown UL1">

<li class="level2nav"><asp:HyperLink ID="HyperLink1" CssClass="level2nav\_\_link" NavigateUrl="~/Parcel/CreateParcel.aspx" runat="server">Create a Shipment</asp:HyperLink></li>

<li class="level2nav"><asp:HyperLink ID="HyperLink3" CssClass="level2nav\_\_link" NavigateUrl="~/Parcel/RateofParcel.aspx" runat="server">Shipping Rates</asp:HyperLink></li>

<li class="level2nav"><asp:HyperLink ID="HyperLink6" CssClass="level2nav\_\_link" NavigateUrl="#" runat="server">Returns</asp:HyperLink></li>

<li class="level2nav"><asp:HyperLink ID="HyperLink8" CssClass="level2nav\_\_link" NavigateUrl="#" runat="server">Learn More about Online Shipping</asp:HyperLink></li>

<li class="level2nav"><asp:HyperLink ID="HyperLink9" CssClass="level2nav\_\_link" NavigateUrl="#" runat="server">All Shiping Services</asp:HyperLink></li>

</ul>

</li>

<li class="level1nav"><asp:HyperLink ID="HyperLink23" CssClass="level1nav\_\_link t2" runat="server">Tracking&blacktriangledown;</asp:HyperLink>

<ul class="dropdown UL2">

<li class="level2nav"><asp:HyperLink ID="HyperLink10" CssClass="level2nav\_\_link" NavigateUrl="~/Parcel/ParcelStat.aspx" runat="server">Advance Shipment Tracking</asp:HyperLink>

</li>

<li class="level2nav"><asp:HyperLink ID="HyperLink12" CssClass="level2nav\_\_link" NavigateUrl="#" runat="server">Track by Email</asp:HyperLink></li>

<li class="level2nav"><asp:HyperLink ID="HyperLink2" runat="server" CssClass="level2nav\_\_link" NavigateUrl="~/Parcel/SrcParcel.aspx">All Tracking Services</asp:HyperLink></li>

</ul>

</li>

<li class="level1nav"><asp:HyperLink ID="HyperLink22" CssClass="level1nav\_\_link t3" runat="server">Support&blacktriangledown;</asp:HyperLink>

<ul class="dropdown UL3">

<li class="level2nav"><asp:HyperLink ID="HyperLink16" CssClass="level2nav\_\_link" NavigateUrl="#" runat="server">File a Claim</asp:HyperLink></li>

<li class="level2nav"><asp:HyperLink ID="HyperLink18" CssClass="level2nav\_\_link" NavigateUrl="#" runat="server">Billing/Invoice Forms</asp:HyperLink></li>

<li class="level2nav"><asp:HyperLink ID="HyperLink4" runat="server" CssClass="level2nav\_\_link" NavigateUrl="#">Find Our Store Locations</asp:HyperLink></li>

</ul>

</li>

</ul>

<asp:HyperLink ID="LoginSignIn" CssClass="login" runat="server">

<span></span><span></span><span></span><span></span>Sign Up or Login

<svg role="presentation" class="icon user--icon"><use xlink:href="#usericon" /></svg>

</asp:HyperLink>

<asp:Panel ID="Plogin" CssClass="login-container" runat="server">

<div class="close"></div>

<div class="logintab">

<div class="inputbox">

<asp:TextBox ID="TBunm" CssClass="input" runat="server" autocomplete="off" required=""></asp:TextBox>

<span class="floating-placeholder">Email / Phone No.</span>

<asp:RequiredFieldValidator ID="RFVunm" runat="server" ErrorMessage="Username is Required" ControlToValidate="TBunm" Display="Dynamic" ForeColor="Red" SetFocusOnError="True">\*</asp:RequiredFieldValidator>

</div>

<div class="inputbox">

<asp:TextBox ID="TBpas" CssClass="input" runat="server" TextMode="Password" required=""></asp:TextBox>

<span class="floating-placeholder">Password</span>

<asp:RequiredFieldValidator ID="RFVpass" runat="server" ErrorMessage="Password is Required" ControlToValidate="TBpas" Display="Dynamic" ForeColor="Red" SetFocusOnError="True">\*</asp:RequiredFieldValidator>

</div>

<asp:ValidationSummary ID="VSlogin" ForeColor="Red" DisplayMode="List" runat="server" />

<asp:Button ID="Blogin" CssClass="sub" runat="server" Text="Log In" OnClick="Blogin\_Click" />

<div class="link">

<asp:HyperLink ID="HyperLink46" CssClass="link\_a" NavigateUrl="~/register.aspx" runat="server">Create Account</asp:HyperLink>

<asp:HyperLink ID="HyperLink47" CssClass="link\_a" NavigateUrl="~/forgotpassword.aspx" runat="server">Forgot Password?</asp:HyperLink>

</div>

</div>

</asp:Panel>

<asp:Panel ID="Puser" CssClass="account" runat="server" Visible="false" OnClick="toogleUserontrol()">

<asp:Panel ID="ProundedDP" CssClass="userDP" runat="server"></asp:Panel>

<asp:Label ID="Aname" CssClass="user" runat="server" Text="Employee"></asp:Label>

<ul class="usercontrol hidden">

<li class="l2\_\_nav"><asp:HyperLink CssClass="l2\_\_link" ID="profile" runat="server" NavigateUrl="~/Profile.aspx">My Profile</asp:HyperLink></li>

<li class="l2\_\_nav"><asp:LinkButton ID="signout" CssClass="l2\_\_link" runat="server" OnClick="Logout">Log Out</asp:LinkButton></li>

</ul>

</asp:Panel>

</nav>

<div class="container">

<a href="#mid" class="scroll-down">

<span class="arrow"></span><span class="arrow"></span>

<span class="arrow"></span>

</a>

<header class="head1">

<h1 class="head1\_\_heading">Welcome to Lightning Corp</h1>

</header>

<div id="mid" class="HQLinks">

<asp:HyperLink ID="HyperLink25" CssClass="HQLinks\_\_link" NavigateUrl="~/Parcel/RateofParcel.aspx" runat="server">

<svg role="presentation" class="icon rate--icon"><use xlink:href="#rate" /></svg>

<p class="HQLinks\_\_p">Rate of transport</p></asp:HyperLink>

<asp:HyperLink ID="HyperLink27" CssClass="HQLinks\_\_link" NavigateUrl="~/Parcel/SrcParcel.aspx" runat="server">

<svg role="presentation" class="icon track--icon"><use xlink:href="#trackParcel" /></svg>

<p class="HQLinks\_\_p">Track Shipmnet</p></asp:HyperLink>

<asp:HyperLink ID="HyperLink26" CssClass="HQLinks\_\_link" NavigateUrl="~/Parcel/CreateParcel.aspx" runat="server">

<svg role="presentation" class="icon create--icon"><use xlink:href="#createParcel" /></svg>

<p class="HQLinks\_\_p">New Shipment</p></asp:HyperLink>

</div>

<div class="content">

<div class="3rd imgDiv" data-src="/svg/background/gift.svg" data-pos="0">

<div class="desc">

<h2 class="desc\_\_h2">Send Gifts</h2>

<p class="desc\_\_p">

Do you want to send gifts to your dearest who lives far from you<br />

on their Birthday or in other Occation? We got it Covered,<br />

We take care of wrapping and packaging while You prepare the gift<br />

We will include a Special Gift from us too

</p>

<asp:HyperLink ID="HyperLink30" CssClass="desc\_\_link" NavigateUrl="~/Parcel/CreateParcel.aspx?parcel=Gift" runat="server">Send Gifts</asp:HyperLink>

</div>

</div>

<div class="4th imgDiv" data-src="/svg/background/food.svg" data-pos="1">

<div class="desc">

<h2 class="desc\_\_h2">Send Food</h2>

<p class="desc\_\_p">Do you want your closest one to Taste the food made by you, but they live Far from you ?<br />We have Food Delivery Service when you need<br />You just make the food, We Take care of everything else<br /></p>

<asp:HyperLink ID="HyperLink31" CssClass="desc\_\_link" NavigateUrl="~/Parcel/CreateParcel.aspx?parcel=CookedFood" runat="server">Send Cooked Food / Dry Food</asp:HyperLink>

</div>

</div>

<asp:Panel ID="Ppresonal" CssClass="1st imgDiv" runat="server" data-src="/svg/background/open.svg" data-pos="2">

<div class="desc">

<h2 class="desc\_\_h2">Open New Personal Account</h2>

<p class="desc\_\_p">Are you new here open a personal Account so that you can keep track of everything you do here</p>

<asp:HyperLink ID="HyperLink28" CssClass="desc\_\_link" NavigateUrl="~/register.aspx?q=personal" runat="server">Create Personal A/c</asp:HyperLink>

</div>

</asp:Panel>

<asp:Panel ID="Pbusiness" CssClass="2nd imgDiv" runat="server" data-src="/svg/background/business.svg" data-pos="3">

<div class="desc">

<h2 class="desc\_\_h2">Open New Business Account</h2>

<p class="desc\_\_p">Are you a business person or a corporative looking for our service Create a Business Account, we take extra care for our business clients</p>

<asp:HyperLink ID="HyperLink29" CssClass="desc\_\_link" NavigateUrl="~/register.aspx?q=business" runat="server">Create Business A/c</asp:HyperLink>

</div>

</asp:Panel>

</div>

<footer class="foot">

<div class="foot\_\_div about">

<h3>About</h3>

<ul>

<li><asp:HyperLink ID="HyperLink32" CssClass="foot\_\_link" NavigateUrl="#" runat="server">About Us </asp:HyperLink></li>

<li><asp:HyperLink ID="HyperLink33" CssClass="foot\_\_link" NavigateUrl="#" runat="server">Contact Us</asp:HyperLink></li>

<li><asp:HyperLink ID="HyperLink34" CssClass="foot\_\_link" NavigateUrl="#" runat="server">Our Store Locatons</asp:HyperLink></li>

</ul>

</div>

<div class="foot\_\_div help">

<h3>Help</h3>

<ul>

<li><asp:HyperLink ID="HyperLink35" CssClass="foot\_\_link" NavigateUrl="#" runat="server">Payments</asp:HyperLink></li>

<li><asp:HyperLink ID="HyperLink36" CssClass="foot\_\_link" NavigateUrl="#" runat="server">Shipping</asp:HyperLink></li>

<li><asp:HyperLink ID="HyperLink37" CssClass="foot\_\_link" NavigateUrl="#" runat="server">International Holiday Schedule</asp:HyperLink></li>

<li><asp:HyperLink ID="HyperLink38" CssClass="foot\_\_link" NavigateUrl="#" runat="server">Cancellation</asp:HyperLink></li>

<li><asp:HyperLink ID="HyperLink39" CssClass="foot\_\_link" NavigateUrl="#" runat="server">FAQ</asp:HyperLink></li>

<li><asp:HyperLink ID="HyperLink40" CssClass="foot\_\_link" NavigateUrl="#" runat="server">Report Infringement</asp:HyperLink></li>

</ul>

</div>

<div class="foot\_\_div policy">

<h3>Policy</h3>

<ul>

<li><asp:HyperLink ID="HyperLink41" CssClass="foot\_\_link" NavigateUrl="#" runat="server">Cancellation Policy</asp:HyperLink></li>

<li><asp:HyperLink ID="HyperLink42" CssClass="foot\_\_link" NavigateUrl="#" runat="server">Terms of Service</asp:HyperLink></li>

<li><asp:HyperLink ID="HyperLink43" CssClass="foot\_\_link" NavigateUrl="#" runat="server">Security</asp:HyperLink></li>

<li><asp:HyperLink ID="HyperLink44" CssClass="foot\_\_link" NavigateUrl="#" runat="server">Privacy</asp:HyperLink></li>

<li><asp:HyperLink ID="HyperLink45" CssClass="foot\_\_link" NavigateUrl="#" runat="server">Conditions Of Carriage</asp:HyperLink></li>

</ul>

</div>

<div class="foot\_\_div region">

<h3>Region: </h3>

<select><option value="India">India</option></select>

</div>

</footer>

<div class="footer\_\_bottom" id="bottom">

<div class="social">

<span class="social\_\_follow">Follow Us on : </span>

<a href="#" class="social\_\_link facebook-icon">

<svg role="presentation" class="icon facebook--icon">

<use xlink:href="#facebookicon" />

</svg>

</a>

<a href="#" class="social\_\_link twitter-icon">

<svg role="presentation" class="icon twitter--icon">

<use xlink:href="#twittericon" />

</svg>

</a>

<a href="#" class="social\_\_link linkedin-icon">

<svg role="presentation" class="icon linkedin--icon">

<use xlink:href="#linkedinicon" />

</svg>

</a>

</div>

<div class="bottom">

<a href="#" class="bottom\_\_link">Sitemap</a> <span>&copy;2019 - 2021 Lightning Corp.</span>

</div>

</div>

</div>

</section>

<svg display="none">

<symbol id="usericon" role="presentation" data-name="Layer 1" xmlns="http://www.w3.org/2000/svg"

viewBox="0 0 194 194">

<path class="border"

d="M97 13a84 84 0 1 1-84 84 84.09513 84.09513 0 0 1 84-84m0-13a97 97 0 1 0 97 97A97 97 0 0 0 97 0z" />

<circle class="head" cx="97" cy="65" r="33" />

<path class="body"

d="M43 142c-8.44147-9.10907-8.31122-14.73462-8-17 .83569-6.08286 6.0473-10.29041 12.5-15.5a70.56271 70.56271 0 0 1 12.00311-7.8181 6.82759 6.82759 0 0 1 6.52424.17143 60.95361 60.95361 0 0 0 60.67427.158 6.83155 6.83155 0 0 1 6.92674.11175A131.12246 131.12246 0 0 1 144.5 109.5c8.96344 6.76105 13.46625 10.24872 14.5 15.5 1.34631 6.83887-3.77936 13.1456-7 17-10.00458 11.9733-28 24-50 24-28.77063 0-49.63916-13.89883-59-24z" />

</symbol>

<symbol id="facebookicon" role="presentation" xmlns="http://www.w3.org/2000/svg" viewBox="0 0 320 512">

<path fill="currentColor"

d="M279.14 288l14.22-92.66h-88.91v-60.13c0-25.35 12.42-50.06 52.24-50.06h40.42V6.26S260.43 0 225.36 0c-73.22 0-121.08 44.38-121.08 124.72v70.62H22.89V288h81.39v224h100.17V288z" />

</symbol>

<symbol id="twittericon" role="presentation" xmlns="http://www.w3.org/2000/svg" viewBox="0 0 512 512">

<path fill="currentColor"

d="M459.37 151.716c.325 4.548.325 9.097.325 13.645 0 138.72-105.583 298.558-298.558 298.558-59.452 0-114.68-17.219-161.137-47.106 8.447.974 16.568 1.299 25.34 1.299 49.055 0 94.213-16.568 130.274-44.832-46.132-.975-84.792-31.188-98.112-72.772 6.498.974 12.995 1.624 19.818 1.624 9.421 0 18.843-1.3 27.614-3.573-48.081-9.747-84.143-51.98-84.143-102.985v-1.299c13.969 7.797 30.214 12.67 47.431 13.319-28.264-18.843-46.781-51.005-46.781-87.391 0-19.492 5.197-37.36 14.294-52.954 51.655 63.675 129.3 105.258 216.365 109.807-1.624-7.797-2.599-15.918-2.599-24.04 0-57.828 46.782-104.934 104.934-104.934 30.213 0 57.502 12.67 76.67 33.137 23.715-4.548 46.456-13.32 66.599-25.34-7.798 24.366-24.366 44.833-46.132 57.827 21.117-2.273 41.584-8.122 60.426-16.243-14.292 20.791-32.161 39.308-52.628 54.253z" />

</symbol>

<symbol id="linkedinicon" role="presentation" xmlns="http://www.w3.org/2000/svg" viewBox="0 0 448 512">

<path fill="currentColor"

d="M416 32H31.9C14.3 32 0 46.5 0 64.3v383.4C0 465.5 14.3 480 31.9 480H416c17.6 0 32-14.5 32-32.3V64.3c0-17.8-14.4-32.3-32-32.3zM135.4 416H69V202.2h66.5V416zm-33.2-243c-21.3 0-38.5-17.3-38.5-38.5S80.9 96 102.2 96c21.2 0 38.5 17.3 38.5 38.5 0 21.3-17.2 38.5-38.5 38.5zm282.1 243h-66.4V312c0-24.8-.5-56.7-34.5-56.7-34.6 0-39.9 27-39.9 54.9V416h-66.4V202.2h63.7v29.2h.9c8.9-16.8 30.6-34.5 62.9-34.5 67.2 0 79.7 44.3 79.7 101.9V416z" />

</symbol>

<symbol id="rate" role="presentation" data-name="rate" xmlns="http://www.w3.org/2000/svg"

viewBox="0 0 112.81 112.88">

<g class="calculator" fill="none" stroke="currentColor" stroke-linecap="round" stroke-linejoin="round"

stroke-width="2">

<path

d="M2.36 43.38h45.28a.86.86 0 0 1 .86.86v66.28a.86.86 0 0 1-.86.86H2.36a.86.86 0 0 1-.86-.86V44.24a.86.86 0 0 1 .86-.86z" />

<path d="M7.51 49.4h34.98v18H7.51zM7.5 77.38h5M17.5 77.38h5M27.5 77.38h5M37.5 77.38h5M7.5 87.38h5M17.5 87.38h5M27.5 87.38h5M7.5 97.38h5M17.5 97.38h5M27.5 97.38h5M40.5 87.38v10" /></g>

<path class="parcel" fill="none" stroke="currentColor" stroke-linecap="round" stroke-linejoin="round" stroke-width="2" d="M87.69 11.4L43.21 32.41M111.31 22.5L66.78 43.51M66.78 43.51v42.02M66.78 43.51L22.16 22.5M22.16 38.94V22.5l44.62-21 44.53 21v42.02L66.78 85.53l-14.22-6.99" />

</symbol>

<symbol id="trackParcel" role="presentation" data-name="track" xmlns="http://www.w3.org/2000/svg"

viewBox="0 0 111.99562 99.88413">

<path class="parcel" fill="none" stroke="currentColor" stroke-width="2"

d="M45.43 43.76967L1 22.43967M45.43 43.76967v41.56M89.69 22.43967l-44.26 21.33M66.19 11.15967l-44.26 21.33M89.69 41.18V22.44L45.39 1.11 1 22.44V65.1l44.39 21.33 12.51-6.02" />

<path class="srch" fill="none" stroke="currentColor" stroke-width="2"

d="M90.83 71.8l19.78 20.61-5.78 6.03-19.79-20.61M87.24 47.69967a19.74 19.74 0 0 1 0 27.12 17.9 17.9 0 0 1-25.30488.69512q-.357-.338-.69512-.69512a19.74 19.74 0 0 1 0-27.12 17.9 17.9 0 0 1 25.30488-.69512q.35705.33796.69512.69512z" />

</symbol>

<symbol id="createParcel" role="presentation" xmlns="http://www.w3.org/2000/svg"

viewBox="0 0 105.69275 85.44504">

<g id="parcel" fill="none" stroke="currentColor" stroke-width="2" stroke-miterlimit="10">

<path d="M55.37508 51.83L10.99947 30.52611a.038.038 0 0 0-.05439.03422v41.37084a.038.038 0 0 0 .02147.03419L55.37508 93.39" transform="translate(-10.44508 -8.43783)" />

<path d="M74.77092 29.01108L44.93 43.39217M65.69 10.78216l-44.26 21.33" />

<path d="M55.37508 51.83v41.54043a.01243.01243 0 0 0 .01772.01125L99.63509 72.61V46.50275M10.94508 30.5L53.04292 9.17774a2.22341 2.22341 0 0 1 1.94051-.03334l22.52165 10.45081 7.71092 3.66544" transform="translate(-10.44508 -8.43783)" /></g>

<g id="plus" fill="none" stroke="currentColor" stroke-width="2" stroke-miterlimit="10">

<path d="M99.205 22.06217h-20.03M89.19 12.08217v20.02999" />

<circle cx="89.19" cy="22.06217" r="16.00275" />

</g>

</symbol>

<symbol id="arrow" role="presentation" data-name="Layer 1" xmlns="http://www.w3.org/2000/svg"

viewBox="0 0 220.6233 111.09344">

<rect class="rect" y="45.94671" width="186.04651" height="19.31202" fill="currentColor" />

<polygon class="trangle" points="160.775 12.967 160.775 98.238 211.318 55.89 160.775 12.967" fill="#fff"

stroke="currentColor" stroke-miterlimit="10" stroke-width="12" />

</symbol>

</svg>

</form>

<script src="js/index.js"></script>

<script src="js/libraries/TweenMax.min.js"></script>

<script src="js/libraries/jquery-3.4.1.min.js"></script>

<script src="js/index.js"></script>

<script>

$(document).ready(function () {

$(".scroll-down").click(function (event) {

if (this.hash !== "") {

event.preventDefault();

var hash = this.hash;

$('html, body').animate({

scrollTop: 80

})

}

})

})

</script>

</body>

</html>

1. Home Page(index.aspx.cs)

using OnlineCourierService.Class;

using System;

using System.Web.UI;

namespace OnlineCourierService

{

public partial class index : System.Web.UI.Page

{

protected void Page\_Load(object sender, EventArgs e)

{

Customer cus;

if (!IsPostBack)

{

if (Session["LightCus"] != null)

{

string CID = Session["LightCus"].ToString();

cus = new Customer(CID, true);

LoginSignIn.Visible = false;

Plogin.Visible = Pbusiness.Visible = Ppresonal.Visible = false;

Puser.Visible = true;

cus.FillData();

Aname.Text = cus.name;

if (cus.imgurl != null)

{

ProundedDP.Style.Add("background-image", cus.imgurl);

}

}

}

}

private void Notifyuser(string msg, bool issuccessful)

{

if (issuccessful)

{

Lerr.Text = msg;

Lerr.Style.Add("color", "Navy");

PError.Style.Add("border", "2px solid Navy");

}

else

{

Lerr.Text = msg;

Lerr.Style.Add("color", "Red");

PError.Style.Add("border", "2px solid lightred");

}

PError.Style.Add("display", "block");

}

protected void Blogin\_Click(object sender, EventArgs e)

{

if (Page.IsValid)

{

string CID = null;

Customer Cus;

try

{

Cus = new Customer(Convert.ToInt64(TBunm.Text.Trim()));

}

catch (FormatException)

{

Cus = new Customer(TBunm.Text.Trim());

}

CID = Cus.GetCIDbyPass(TBpas.Text.Trim());

if (CID == null)

{

Notifyuser("Email/Phone Number or Password Do not Match", false);

return;

}

Session["LightCus"] = CID;

Response.Redirect("~/index.aspx");

}

else

{

Notifyuser("Validation Error, Data Not Saved", false);

}

}

protected void Logout(object sender, EventArgs e){

Session.Remove("LightCus");

Response.Redirect("~/index.aspx");

}

}

}

1. New Customer Registration Page(register.aspx.cs)

using OnlineCourierService.Class;

using System;

using System.IO;

using System.Web;

using System.Web.UI;

namespace OnlineCourierService

{

public partial class register : System.Web.UI.Page

{

protected void Page\_Load(object sender, EventArgs e)

{

string q = Request.QueryString["q"];

if (q != null)

{

if (q.Equals("personal"))

{

DDLAccType.SelectedIndex = 1;

}

else if (q.Equals("business"))

{

DDLAccType.SelectedIndex = 2;

}

DDLAccType.Focus();

DDLAccType.BorderColor = System.Drawing.Color.FromArgb(21, 238, 255);///#15eeff

}

}

protected void Cancel(object sender, EventArgs e)

{

Response.Redirect("~/index.aspx");

}

private void Notifyuser(string msg, bool isSuccessful)

{

if (isSuccessful)

{

Lerr.Text = msg;

Lerr.Style.Add("color", "Navy");

PError.Style.Add("border", "2px solid Navy");

}

else

{

Lerr.Text = msg;

Lerr.Style.Add("color", "Red");

PError.Style.Add("border", "2px solid lightred");

}

PError.Style.Add("display", "block");

}

protected void Create(object sender, EventArgs e)

{

if (Page.IsValid)

{

HttpPostedFile img = FUpic.PostedFile;

if (!img.FileName.Equals(""))

{

string fileExtennsion = Path.GetExtension(img.FileName).ToLower();

if (fileExtennsion != ".jpg" && fileExtennsion != ".jpeg" && fileExtennsion != ".png")

{

Notifyuser("Only images(.jpg, .png, .jpeg) can be uploaded", false);

return;

}

}

if (Customer.CountCustomer(Convert.ToInt64(TBphone.Text.Trim())) != 0)

{

Notifyuser("Same Phone Number is already Registered try to Login with it or try with Another one", false);

return;

}

if (Customer.CountCustomer(TBemail.Text.Trim()) != 0)

{

Notifyuser("Same Email Address is already Registered try to Login with it or try with Another one", false);

return;

}

string name = TBunm.Text.Trim();

string addr = TBaddr.Text.Trim();

string gender = RBLgenger.SelectedValue;

DateTime dob = DateTime.Parse(TBdob.Text.Trim());

long phno = Convert.ToInt64(TBphone.Text.Trim());

string email = TBemail.Text.Trim();

long bid = Convert.ToInt64(DDLbranch.SelectedValue);

string pass = TBcpass.Text.Trim();

string accType = DDLAccType.SelectedValue;

Customer cus = new Customer(name, addr, gender, dob, phno, email, bid, accType, pass);

string cid = cus.InsertData();

if (cid.Equals(null))

{

Notifyuser("Account Creation Failed", false);

return;

}

if (!img.FileName.Equals(""))

{

cus.InsertUpdateImage(img);

}

Notifyuser("Customer Account with ID: " + cid + " has been created", true);

}

else

{

Notifyuser("Validation Error, Data Not Saved", false);

}

}

}

}

1. Forgot Password Page(forgotpassword.aspx.cs)

using OnlineCourierService.Class;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Web.UI;

using System.Web.UI.WebControls;

namespace OnlineCourierService

{

public partial class forgotpassword : System.Web.UI.Page

{

protected void Page\_Load(object sender, EventArgs e){}

private void Notifyuser(string msg, bool issuccessful)

{

if (issuccessful)

{

Lerr.Text = msg;

Lerr.Style.Add("color", "Navy");

PError.Style.Add("border", "2px solid Navy");

}

else

{

Lerr.Text = msg;

Lerr.Style.Add("color", "Red");

PError.Style.Add("border", "2px solid lightred");

}

PError.Style.Add("display", "block");

}

protected void Bpreset\_Click(object sender, EventArgs e)

{

if (Page.IsValid)

{

string email = TBunm.Text.Trim();

if (Customer.CountCustomer(email) != 1)

{

Notifyuser("Email is not Found to be Registered", false);

return;

}

string link = HttpContext.Current.Request.Url.Authority;

link += "/resetPassword.aspx";

if (Customer.GenerateResetPasswordLink(email, link) == 1)

{

Notifyuser("An email with instructions to reset your password is sent to your registered email", true);

}

else

{

Notifyuser("We Are Facing Some Internal Problem Try Again Later", false);

return;

}

}

else

{

Notifyuser("Validation Error", false);

return;

}

}

}

}

1. Reset Password(resetPassword.aspx.cs)

using OnlineCourierService.Class;

using System;

using System.Web.UI;

namespace OnlineCourierService

{

public partial class resetPassword : System.Web.UI.Page

{

protected void Page\_Load(object sender, EventArgs e)

{

string guid = Request.QueryString["uid"];

if (guid == null)

{

Notifyuser("Password Reset link has expired or is invalid", false);

TBCnewPass.Enabled = TBnewPass.Enabled = BcngP.Enabled = false;

return;

}

if (!IsPostBack)

{

string cid = Customer.isPasswordResetLinkValid(guid);

if (cid == null)

{

Notifyuser("Password Reset link has expired or is invalid", false);

TBCnewPass.Enabled = TBnewPass.Enabled = BcngP.Enabled = false;

return;

}

Customer cus = new Customer(cid, true);

cus.FillData();

Luser.Text += " for " + cus.email;

Session.Add("ValidGUID", true);

}

}

private void Notifyuser(string msg, bool issuccessful)

{

if (issuccessful)

{

Lerr.Text = msg;

Lerr.Style.Add("color", "Navy");

PError.Style.Add("border", "2px solid Navy");

}

else

{

Lerr.Text = msg;

Lerr.Style.Add("color", "Red");

PError.Style.Add("border", "2px solid lightred");

}

PError.Style.Add("display", "block");

}

protected void BcngP\_Click(object sender, EventArgs e)

{

if (!Convert.ToBoolean(Session["ValidGUID"]))

{

Notifyuser("Update Password Failed !", false);

return;

}

if (Page.IsValid)

{

if (Customer.ResetPassword(Request.QueryString["uid"], TBCnewPass.Text.Trim()) != 1)

{

Notifyuser("Update Password Failed !", false);

return;

}

Notifyuser("Password Changed Successfully!", true);

}

else

{

Notifyuser("Validation Error, Try Again Later", false);

return;

}

}

}

}

1. Profile Page(Profile.aspx.cs)

using OnlineCourierService.Class;

using OnlineCourierService.employee.Classes;

using System;

using System.Web.Caching;

using System.Web;

using System.Web.UI;

using System.IO;

namespace OnlineCourierService

{

public partial class Profile : System.Web.UI.Page

{

protected void Page\_Load(object sender, EventArgs e)

{

if (Session["LightCus"] == null)

{

Response.Redirect("~/index.aspx");

}

Customer cus;

if (!IsPostBack && Session["LightCus"] != null)

{

Lcid.Text = Session["LightCus"].ToString();

cus = new Customer(Session["LightCus"].ToString(), true);

LoadData(cus);

}

}

private void LoadData(Customer cus)

{

cus.FillData();

Cache.Add("CustomerData", cus, null, Cache.NoAbsoluteExpiration, TimeSpan.FromMinutes(5), CacheItemPriority.AboveNormal, null);

Aname.Text = cus.name;

if (cus.imgurl != null)

{

ProundedDP.Style.Add("background-image", cus.imgurl);

}

TBunm.Text = cus.name;

TBaddr.Text = cus.address;

TBdob.Text = cus.dob.ToString("yyyy-MM-dd");

RBLgenger.SelectedValue = cus.gender;

TBphone.Text = cus.phno.ToString();

TBemail.Text = cus.email;

DDLAccType.SelectedValue = cus.accType;

if (cus.imgurl != null)

{

ImgProfilePic.ImageUrl = cus.imgurl;

}

long rid = Branch.GetRidbyBid(cus.BID);

DDLregwithRID.SelectedValue = rid.ToString();

DDLbranch.DataSource = DataAccessLayer.GetBranchByRid((int)rid); ;

DDLbranch.DataValueField = "Bid";

DDLbranch.DataTextField = "Bname";

DDLbranch.DataBind();

DDLbranch.SelectedValue = cus.BID.ToString();

}

private void Notifyuser(string msg, bool isSuccessful)

{

if (isSuccessful)

{

Lerr.Text = msg;

Lerr.Style.Add("color", "Navy");

PError.Style.Add("border", "2px solid Navy");

}

else

{

Lerr.Text = msg;

Lerr.Style.Add("color", "Red");

PError.Style.Add("border", "2px solid lightred");

}

PError.Style.Add("display", "block");

}

protected void Logout(object sender, EventArgs e)

{

Session.Remove("LightCus");

Response.Redirect("~/index.aspx");

}

protected void Update(object sender, EventArgs e)

{

if (Page.IsValid)

{

Customer cus = (Customer)Cache.Get("CustomerData");

if (cus == null)

{

cus = new Customer(Session["LightCus"].ToString(), true);

cus.FillData();

}

HttpPostedFile img = FUpic.PostedFile;

if (!img.FileName.Equals(""))

{

string fileExtennsion = Path.GetExtension(img.FileName).ToLower();

if (fileExtennsion != ".jpg" && fileExtennsion != ".jpeg" && fileExtennsion != ".png")

{

Notifyuser("Only images(.jpg, .png, .jpeg) can be uploaded", false);

return;

}

}

long phno = Convert.ToInt64(TBphone.Text.Trim());

if (Customer.CountCustomer(phno) != 0 && cus.phno != phno)

{

Notifyuser("Phnoe Number is Already Registed with another account", false);

return;

}

string email = TBemail.Text.Trim();

if (Customer.CountCustomer(email) != 0 && !cus.email.Equals(email))

{

Notifyuser("Email is Already Registed with another account", false);

return;

}

string name = TBunm.Text.Trim();

string addr = TBaddr.Text.Trim();

DateTime dob = DateTime.Parse(TBdob.Text.Trim());

string gender = RBLgenger.SelectedValue;

string acctype = DDLAccType.SelectedValue;

long bid = Convert.ToInt64(DDLbranch.SelectedValue);

if (cus.UpdateData(new Customer(name, addr, gender, dob, phno, email, bid, acctype, null)) > 0)

{

cus.InsertUpdateImage(img);

Notifyuser("Information Updated Successfully", true);

}

else

{

Notifyuser("Data not Saved Due to Data Conflict", false);

}

LoadData(cus);

}

else

{

Notifyuser("Validation Error Data Not Saved", false);

return;

}

}

protected void Cancel(object sender, EventArgs e)

{

Response.Redirect("~/index.aspx");

}

protected void DDLregwithRID\_SelectedIndexChanged(object sender, EventArgs e)

{

int rid = Convert.ToInt32(DDLregwithRID.SelectedValue);

DDLbranch.DataSource = DataAccessLayer.GetBranchByRid(rid);

DDLbranch.DataValueField = "Bid";

DDLbranch.DataTextField = "Bname";

DDLbranch.DataBind();

}

}

}

1. Change Password(changePassword.aspx.cs)

using OnlineCourierService.Class;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web.Caching;

using System.Web;

using System.Web.UI;

using System.Web.UI.WebControls;

namespace OnlineCourierService

{

public partial class changePassword : System.Web.UI.Page

{

protected void Page\_Load(object sender, EventArgs e)

{

if (Session["LightCus"] == null)

{

Response.Redirect("~/index.aspx");

}

Customer cus;

if (!IsPostBack && Session["LightCus"] != null)

{

cus = new Customer(Session["LightCus"].ToString(), true);

cus.FillData();

Cache.Add("CustomerData", cus, null, Cache.NoAbsoluteExpiration, TimeSpan.FromMinutes(5), CacheItemPriority.AboveNormal, null);

Aname.Text = cus.name;

if (cus.imgurl != null)

{

ProundedDP.Style.Add("background-image", cus.imgurl);

}

}

}

private void Notifyuser(string msg, bool isSuccessful)

{

if (isSuccessful)

{

Lerr.Text = msg;

Lerr.Style.Add("color", "Navy");

PError.Style.Add("border", "2px solid Navy");

}

else

{

Lerr.Text = msg;

Lerr.Style.Add("color", "Red");

PError.Style.Add("border", "2px solid lightred");

}

PError.Style.Add("display", "block");

}

protected void Logout(object sender, EventArgs e)

{

Session.Remove("LightCus");

Response.Redirect("~/index.aspx");

}

protected void Cancel(object sender, EventArgs e)

{

Response.Redirect("~/Profile.aspx");

}

protected void CngPass(object sender, EventArgs e)

{

if (Page.IsValid)

{

Customer cus = (Customer)Cache.Get("CustomerData");

if (cus == null)

{

cus = new Customer(Session["LightCus"].ToString(), true);

cus.FillData();

}

string oldpass = TBOLDpass.Text.Trim();

string cid = Session["LightCus"].ToString();

string vcid = cus.GetCIDbyPass(oldpass);

if (!vcid.Equals(cid))

{

Notifyuser("Old Password do not Match", false);

return;

}

string newpass = TBcpass.Text.Trim();

if (cus.UpdatePassword(newpass, oldpass) > 0)

{

Notifyuser("Password Changed Successfully", true);

}

else

{

Notifyuser("Password not Changed due to Data conflict", false);

}

}

else

{

Notifyuser("Validation Error Data Not Saved", false);

return;

}

}

}

}

1. Rate Of Parcel(RateofParcel.aspx.cs)

using OnlineCourierService.Class;

using System;

using System.Web;

namespace OnlineCourierService.Parcel

{

public partial class RateofParcel : System.Web.UI.Page

{

private Customer cus;

private string link;

protected void Page\_Load(object sender, EventArgs e)

{

link = HttpContext.Current.Request.Url.AbsolutePath;

if (!IsPostBack)

{

if (Session["LightCus"] != null)

{

string CID = Session["LightCus"].ToString();

cus = new Customer(CID, true);

LoginSignIn.Visible = false;

Puser.Visible = true;

cus.FillData();

Aname.Text = cus.name;

if (cus.imgurl != null)

{

ProundedDP.Style.Add("background-image", cus.imgurl);

}

}

}

}

protected void Logout(object sender, EventArgs e)

{

Session.Remove("LightCus");

Response.Redirect("~/index.aspx");

}

protected void LoginSignIn\_Click(object sender, EventArgs e)

{

HttpCookie cookie = new HttpCookie("redirectLink");

cookie["link"] = "~" + link;

cookie.Expires = DateTime.Now.AddMinutes(5);

Response.Cookies.Add(cookie);

Server.Transfer("~/LoginPopup.aspx");

}

}

}

1. Create/Ship a New Parcel(CreateParcel.aspx.cs)

using OnlineCourierService.Class;

using OnlineCourierService.employee.Classes;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Web.UI;

using System.Web.UI.WebControls;

namespace OnlineCourierService.Parcel

{

public partial class CreateParcel : System.Web.UI.Page

{

string link, package = null;

protected void Page\_Load(object sender, EventArgs e)

{

bool redirect = true;

if (Request.QueryString["Redirect"] != null)

{

try { redirect = Convert.ToBoolean(Request.QueryString["Redirect"]); }

catch (Exception) { Response.Redirect("~/Parcel/CreateParcel.aspx?redirect=false"); }

}

link = HttpContext.Current.Request.Url.AbsolutePath;

Customer cus;

if (!IsPostBack)

{

if (Request.QueryString["parcel"] != null)

{

package = Request.QueryString["parcel"];

try

{

DDLPtype.SelectedValue = package;

DDLPtype.BorderColor = System.Drawing.Color.Cyan;

TogglePFood();

}

catch (Exception)

{

Notifyuser("Somthing Happened Wrong! We are looking into it", false);

}

}

if (Session["LightCus"] != null)

{

string CID = Session["LightCus"].ToString();

cus = new Customer(CID, true);

LoginSignIn.Visible = false;

Puser.Visible = true;

cus.FillData();

Aname.Text = cus.name;

if (cus.imgurl != null)

{

ProundedDP.Style.Add("background-image", cus.imgurl);

}

TBPweight.Enabled = TBsName.Enabled = TBsEmail.Enabled =

TBsAddr.Enabled = TBrName.Enabled = TBrEmail.Enabled =

TBrAddr.Enabled = TBfood.Enabled = false;

}

else if (redirect)

{

HttpCookie cookie = new HttpCookie("redirectLink");

cookie["link"] = "~" + link;

if (package != null)

{

cookie["qstr"] = "?parcel=" + package;

}

cookie.Expires = DateTime.Now.AddMinutes(5);

Response.Cookies.Add(cookie);

Server.Transfer("~/LoginPopup.aspx");

}

else

{

Psender.Visible = false;

}

}

}

private void Notifyuser(string msg, bool isSuccessful)

{

if (isSuccessful)

{

Lerr.Text = msg;

Lerr.Style.Add("color", "Navy");

PError.Style.Add("border", "2px solid Navy");

}

else

{

Lerr.Text = msg;

Lerr.Style.Add("color", "Red");

PError.Style.Add("border", "2px solid lightred");

}

PError.Style.Add("display", "block");

}

protected void Logout(object sender, EventArgs e)

{

Session.Remove("LightCus");

Response.Redirect("~/index.aspx");

}

protected void LoginSignIn\_Click(object sender, EventArgs e)

{

if (Request.QueryString["parcel"] != null)

{

package = Request.QueryString["parcel"];

}

HttpCookie cookie = new HttpCookie("redirectLink");

cookie["link"] = "~" + link;

if (package != null)

{

cookie["qstr"] = "?parcel=" + package;

}

cookie.Expires = DateTime.Now.AddMinutes(5);

Response.Cookies.Add(cookie);

Server.Transfer("~/LoginPopup.aspx");

}

protected void DDLsReg\_SelectedIndexChanged(object sender, EventArgs e)

{

LoadsBranch();

DDLsbranch.Focus();

}

private void LoadsBranch()

{

long rid = Convert.ToInt64(DDLsReg.SelectedValue);

if (rid != -1)

{

DDLsbranch.DataSource = DataAccessLayer.GetBranchByRid((int)rid);

DDLsbranch.DataTextField = "BName";

DDLsbranch.DataValueField = "Bid";

DDLsbranch.DataBind();

}

}

protected void DDLrReg\_SelectedIndexChanged(object sender, EventArgs e)

{

LoadrBranch();

DDLrbranch.Focus();

}

private void LoadrBranch()

{

long rid = Convert.ToInt64(DDLrReg.SelectedValue);

if (rid != -1)

{

DDLrbranch.DataSource = DataAccessLayer.GetBranchByRid((int)rid);

DDLrbranch.DataTextField = "BName";

DDLrbranch.DataValueField = "Bid";

DDLrbranch.DataBind();

}

}

protected void DDLPtype\_SelectedIndexChanged(object sender, EventArgs e)

{

TogglePFood();

DDLPtype.Focus();

}

protected void DDLpackage\_SelectedIndexChanged(object sender, EventArgs e)

{

TogglePFood();

DDLpackage.Focus();

}

private void TogglePFood()

{

if (DDLPtype.SelectedValue == "CookedFood" || DDLPtype.SelectedValue == "DryFood")

{

if (DDLpackage.SelectedIndex == 0)//&& !PFood.Visible

{

PFood.Visible = true;

TBfood.Enabled = true;

}

else if (DDLpackage.SelectedIndex == 1)//&& PFood.Visible

{

PFood.Visible = false;

}

}

else if (DDLPtype.SelectedValue != "CookedFood" && DDLPtype.SelectedValue != "DryFood")

{

if (PFood.Visible)

{

PFood.Visible = false;

}

}

}

protected void BRec\_Click(object sender, EventArgs e)

{

if (TBORemail.Text != null)

{

string Recemail = TBORemail.Text.Trim();

Customer cus = null;

if (Session["LightCus"] != null)

{

cus = new Customer(Session["LightCus"].ToString(), true);

cus.FillData();

}

else

{

Server.Transfer("~/Parcel/CreateParcel.aspx");

}

if (cus.email.Equals(Recemail))

{

Notifyuser("Do not Enter Your Email Address !", false);

return;

}

if (Customer.CountCustomer(Recemail) != 1)

{

Notifyuser("Entered Email Address is not Registered in Our Website!", false);

return;

}

HFSender.Value = TBORemail.Text.Trim();

}

Psender.Visible = false;

TBPweight.Enabled = TBsName.Enabled = TBsEmail.Enabled =

TBsAddr.Enabled = TBrName.Enabled = TBrEmail.Enabled =

TBrAddr.Enabled = TBfood.Enabled = true;

LoadSender(new Customer(Session["LightCus"].ToString(), true));

LoadReceiver(Customer.GetinfoByEmail(TBORemail.Text.Trim()));

}

protected void LBRecC\_Click(object sender, EventArgs e)

{

Psender.Visible = false;

TBPweight.Enabled = TBsName.Enabled = TBsEmail.Enabled =

TBsAddr.Enabled = TBrName.Enabled = TBrEmail.Enabled =

TBrAddr.Enabled = TBfood.Visible = true;

LoadSender(new Customer(Session["LightCus"].ToString(), true));

}

private void LoadSender(Customer sen)

{

sen.FillData();

TBsName.Text = sen.name;

TBsEmail.Text = sen.email;

TBsAddr.Text = sen.address;

DDLsReg.SelectedValue = Branch.GetRidbyBid(sen.BID).ToString();

LoadsBranch();

DDLsbranch.SelectedValue = sen.BID.ToString();

}

private void LoadReceiver(Customer rec)

{

rec.FillData();

TBrName.Text = rec.name;

TBrEmail.Text = rec.email;

TBrAddr.Text = rec.address;

DDLrReg.SelectedValue = Branch.GetRidbyBid(rec.BID).ToString();

LoadrBranch();

DDLrbranch.SelectedValue = rec.BID.ToString();

}

protected void LBcancel\_Click(object sender, EventArgs e)

{

Response.Redirect("~/index.aspx");

}

protected void Bsubmit\_Click(object sender, EventArgs e)

{

if (Page.IsValid)

{

string parcelType = DDLPtype.SelectedValue;

string paymentMethod = DDLPaymentM.SelectedValue;

double weight = Convert.ToDouble(TBPweight.Text.Trim());

SenderReceiver Sender, Receiver;

if (Session["LightCus"] != null)

{

Sender = new SenderReceiver(Session["LightCus"].ToString(), TBsName.Text.Trim(), TBsEmail.Text.Trim(), TBsAddr.Text.Trim());

}

else

{

Sender = new SenderReceiver("-1", TBsName.Text.Trim(), TBsEmail.Text.Trim(), TBsAddr.Text.Trim());

}

Receiver = new SenderReceiver("-1", TBrName.Text.Trim(), TBrEmail.Text.Trim(), TBrAddr.Text.Trim());

long sBID = Convert.ToInt64(DDLsbranch.SelectedValue);

long sRID = Convert.ToInt64(DDLsReg.SelectedValue);

long rBID = Convert.ToInt64(DDLrbranch.SelectedValue);

long rRID = Convert.ToInt64(DDLrReg.SelectedValue);

int PackagingByCustomer = Convert.ToInt32(DDLpackage.SelectedValue);

int container;

if (PFood.Visible) { container = Convert.ToInt32(TBfood.Text.Trim()); }

else { container = -1; }

Package p = new Package(parcelType, weight, Sender, Receiver, sBID, rBID, sRID, rRID, 0, 0, paymentMethod, "Pending", PackagingByCustomer, container);

string plid = p.insertPackage();

if (plid == null)

{

Notifyuser("Parcel Request Creation Failed! Try Again Later", false);

return;

}

Notifyuser("Parcel Request Placed Successfully with ID: \"" + plid + "\" Note it Down if Your are Not a Registered User", true);

}

else

{

Notifyuser("Validation Error, Data Not Saved", false);

return;

}

}

}

}

1. Search Parcel(SrcParcel.aspx.cs)

using OnlineCourierService.Class;

using System;

using System.Web;

using System.Web.UI;

namespace OnlineCourierService.Parcel

{

public partial class SrcParcel : System.Web.UI.Page

{

private Customer cus;

private string link;

protected void Page\_Load(object sender, EventArgs e)

{

link = HttpContext.Current.Request.Url.AbsolutePath;

if (!IsPostBack)

{

if (Session["LightCus"] != null)

{

string CID = Session["LightCus"].ToString();

cus = new Customer(CID, true);

PYParcel.Visible = true;

LoginSignIn.Visible = false;

Puser.Visible = true;

LoadParcelByCID(0);

cus.FillData();

Aname.Text = cus.name;

if (cus.imgurl != null)

{

ProundedDP.Style.Add("background-image", cus.imgurl);

}

}

}

}

private void Notifyuser(string msg, bool isSuccessful)

{

if (isSuccessful)

{

Lerr.Text = msg;

Lerr.Style.Add("color", "Navy");

PError.Style.Add("border", "2px solid Navy");

}

else

{

Lerr.Text = msg;

Lerr.Style.Add("color", "Red");

PError.Style.Add("border", "2px solid lightred");

}

PError.Style.Add("display", "block");

}

private void LoadSrcResult()

{

GVsrcParcel.DataSource = Package.GetParcelsByPLID(TBsrcPLID.Text.Trim());

GVsrcParcel.DataBind();

}

protected void Logout(object sender, EventArgs e)

{

Session.Remove("LightCus");

Response.Redirect("~/index.aspx");

}

protected void LoginSignIn\_Click(object sender, EventArgs e)

{

HttpCookie cookie = new HttpCookie("redirectLink");

cookie["link"] = "~" + link;

cookie.Expires = DateTime.Now.AddMinutes(5);

Response.Cookies.Add(cookie);

Server.Transfer("~/LoginPopup.aspx");

}

protected void Bsrc\_Click(object sender, EventArgs e)

{

if (Page.IsValid) { LoadSrcResult(); }

else

{

Notifyuser("Validation Error Search Not Completed", false);

return;

}

}

protected void RBLParcels\_SelectedIndexChanged(object sender, EventArgs e)

{

LoadParcelByCID(RBLParcels.SelectedIndex);

}

private void LoadParcelByCID(int type)

{

if (Session["LightCus"] != null)

{

string CID = Session["LightCus"].ToString();

GVbyCID.DataSource = Package.GetParcelsByCID(CID, type);

GVbyCID.DataBind();

}

}

}

}

1. Advance Shipment/Parcel Tracking(ParcelStat.aspx.cs)

using OnlineCourierService.Class;

using OnlineCourierService.employee.Classes;

using System;

using System.Web;

using System.Web.UI.HtmlControls;

using System.Web.UI.WebControls;

namespace OnlineCourierService.Parcel

{

public partial class ParcelStat : System.Web.UI.Page

{

private Customer cus;

private string link;

protected void Page\_Load(object sender, EventArgs e)

{

link = HttpContext.Current.Request.Url.AbsolutePath;

if (!IsPostBack){

if (Session["LightCus"] != null)

{

string CID = Session["LightCus"].ToString();

cus = new Customer(CID, true);

LoginSignIn.Visible = false;

Puser.Visible = true;

cus.FillData();

Aname.Text = cus.name;

if (cus.imgurl != null)

{

ProundedDP.Style.Add("background-image", cus.imgurl);

}

}

if (Request.QueryString["PLID"] != null)

{

LoadData(Request.QueryString["PLID"]);

}

}

}

private void Notifyuser(string msg, bool isSuccessful)

{

if (isSuccessful)

{

Lerr.Text = msg;

Lerr.Style.Add("color", "Navy");

PError.Style.Add("border", "2px solid Navy");

}

else

{

Lerr.Text = msg;

Lerr.Style.Add("color", "Red");

PError.Style.Add("border", "2px solid lightred");

}

PError.Style.Add("display", "block");

}

protected void Logout(object sender, EventArgs e)

{

Session.Remove("LightCus");

Response.Redirect("~/index.aspx");

}

protected void LoginSignIn\_Click(object sender, EventArgs e)

{

HttpCookie cookie = new HttpCookie("redirectLink");

cookie["link"] = "~" + link;

cookie.Expires = DateTime.Now.AddMinutes(5);

Response.Cookies.Add(cookie);

Server.Transfer("~/LoginPopup.aspx");

}

private void LoadData(string PLID)

{

Package parcel = new Package(PLID);

parcel.FillData();

LPLID.Text += PLID;

LPtype.Text += parcel.ParcelType;

LPweight.Text += parcel.Weight + "KG";

LPRdate.Text += parcel.ReqDate;

LPpackaging.Text += (parcel.PackagingByCustomer == 0) ? "Customer" : "Lightning Corp";

if (parcel.Container != -1)

{

LPcontainer.Visible = true;

LPcontainer.Text += parcel.Container;

}

if (parcel.Distance == 0)

{

LPDistance.Text += "Yet to Be calculated";

LPinvoicePrice.Text += "Yet to Be calculated";

LPpaymentDate.Text += "Unknown";

}

else

{

LPDistance.Text += "About " + parcel.Distance + "KM";

LPinvoicePrice.Text += "Rs." + parcel.invoicePrice;

LPpaymentDate.Text += parcel.PayDate;

}

LPpaymentStat.Text += parcel.PaymentStatus;

LPpaymentMethod.Text += parcel.PaymentMethod;

LSname.Text += parcel.Sender.Name;

LSaddress.Text += parcel.Sender.Address;

LSemail.Text += parcel.Sender.Email;

if (parcel.Sender.CID == "-1") { LSisRegistered.Text += "No"; }

else

{

LSisRegistered.Text += "Yes";

Customer rec = Customer.GetinfoByEmail(parcel.Sender.Email);

LSphno.Visible = true;

LSphno.Text += rec.phno;

}

LRname.Text += parcel.Receiver.Name;

LRaddress.Text += parcel.Receiver.Address;

LRemail.Text += parcel.Receiver.Email;

if (parcel.Receiver.CID == "-1") { LRisRegistered.Text += "no"; }

else

{

LRisRegistered.Text += "Yes";

Customer rec = Customer.GetinfoByEmail(parcel.Receiver.Email);

LRphno.Visible = true;

LRphno.Text += rec.phno;

}

if (parcel.statuslist.Count != 0)

{

Pstatus.Controls.Add(GenerateStaatusDiv("stat\_\_div done", "Request Placed", parcel.ReqDate.ToString()));

double c1 = 1, c2 = 0, c = 0;

Employee emp;

parcel.statuslist.ForEach(s =>{

if ((parcel.SourceRID != -1 || (c != 4 && c != 5)) && (parcel.DestRID != -1 || (c != 6 && c != 7)) && (parcel.SourceBID != parcel.DestBID || c != 8))

{

if (s.date != null)

{

c1++;

emp = new Employee(s.eid);

emp.Filldata();

Pstatus.Controls.Add(GenerateStaatusDiv("stat\_\_div done", s.remark, s.date.ToString()));

}

else if (s.date == null)

{

c2++;

Pstatus.Controls.Add(GenerateStaatusDiv("stat\_\_div", s.remark, ""));

}

}

c++;

});

double percent = 100 - ((c1 \* 100d) / (c1 + c2));

string p = percent + "% " + percent + "%";

PprogressBar.Style.Add("background-position", p);

}

}

private HtmlGenericControl GenerateStaatusDiv(string divClassName, string l1Text, string l2Text)

{

Label l1 = new Label();

l1.Text = l1Text;

Label l2 = new Label();

l2.Text = l2Text;

HtmlGenericControl span = new HtmlGenericControl("span");

span.Attributes.Add("class", "round");

HtmlGenericControl div = new HtmlGenericControl("div");

div.Attributes.Add("class", divClassName);

div.Controls.AddAt(0, span);

div.Controls.AddAt(1, l1);

div.Controls.AddAt(2, l2);

return div;

}

}

}

**Employee/Admin Side**

1. Login Page(login.aspx.cs)

using OnlineCourierService.employee.Classes;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Web.UI;

using System.Web.UI.WebControls;

namespace OnlineCourierService.employee

{

public partial class login : System.Web.UI.Page

{

private Employee emp;

public string name

{

get

{

return Aname.Text.Trim();

}

}

protected void Page\_Load(object sender, EventArgs e){}

protected void getUser(object sender, EventArgs e)

{

String user = Aname.Text.Trim();

try

{

emp = new Employee(Convert.ToInt64(user));

}

catch (Exception)

{

emp = new Employee(user);

}

finally

{

UsernameError.Text = "\"" + user + "\" not found to be registered";

UsernameError.Visible = true;

}

emp.Filldata();

int count = emp.CountbyEmailorphno();

if (count == 1)

{

Response.Redirect("./password.aspx?user=" + user);

}

else

{

UsernameError.Text = "\"" + user + "\" not found to be registered";

UsernameError.Visible = true;

}

}

}

}

1. Check Password(password.aspx.cs)

[this Page desides wherather to login as normal Employee Or as An Administrator]

using OnlineCourierService.employee.Classes;

using System;

namespace OnlineCourierService.employee

{

public partial class password : System.Web.UI.Page

{

private string euser;

private string epass;

private int id;

private Employee emp;

protected void Page\_Load(object sender, EventArgs e)

{

euser = Request.QueryString["user"];

if (!IsPostBack){

login prevPage = this.PreviousPage;

if (prevPage != null) {

euser = prevPage.name;

}

try

{

if (euser.Equals(null)){

Response.Redirect("./login.aspx");

}

adminName.Text = "Give Password for : " + euser;

adminName.Enabled = adminName.Visible = true;

}

catch (NullReferenceException){ Response.Redirect("~/employee/login.aspx");}

catch (Exception){ Response.Redirect("~/employee/login.aspx");}

}

}

protected void logIn(object sender, EventArgs e)

{

try{emp = new Employee(Convert.ToInt64(euser));}

catch (FormatException){emp = new Employee(euser);}

epass = Apass.Text.Trim();

emp.Filldata();

this.id = emp.CheckPassword(epass);

if (id >= 1000)

{

if (emp.IsAdmin())

{

Session["LightAdm"] = id.ToString();

Response.Redirect("~/employee/admin.aspx");

}

else

{

Session["LightEmp"] = id.ToString();

Response.Redirect("~/employee/emp.aspx");

}

}

else

{

status.Text = "Wrong Password Entered";

status.Visible = true;

forgotPass.Visible = true;

}

}

}

}

1. Employee/Admin Profile(empProfile.aspx.cs)

using System;

using System.IO;

using System.Web;

using OnlineCourierService.employee.Classes;

namespace OnlineCourierService.employee

{

public partial class empProfile : System.Web.UI.Page

{

private Employee emp;

private string id;

protected void Page\_Load(object sender, EventArgs e)

{

try

{

id = Session["LightAdm"].ToString();

}

catch (NullReferenceException)

{

try

{

id = Session["LightEmp"].ToString();

signout.Click -= Logout;

signout.Click += new EventHandler(Logout1);

}

catch (Exception)

{

Response.Redirect("./login.aspx");

}

}

catch (Exception)

{

Response.Redirect("./login.aspx");

}

if (id.Equals(null))

{

Response.Redirect("./login.aspx");

}

emp = new Employee(Convert.ToInt32(id));

emp.Filldata();

if (emp.IsAdmin())

{

logo.NavigateUrl = HLhome.NavigateUrl = "~/employee/admin.aspx";

}

else

{

logo.NavigateUrl = HLhome.NavigateUrl = "~/employee/emp.aspx";

HL2.Text = "Create Parcel";

HL2.NavigateUrl = "#";

HL3.Text = "Update Parcel Status";

HL3.NavigateUrl = "#";

HL4.Text = "Track Parcel";

HL4.NavigateUrl = "#";

HL5.Visible = false;

}

if (!IsPostBack)

{

string imgurl = emp.LoadImage();

if (imgurl != null)

{

ProundedDP.Style.Add("background-image", "url(" + imgurl + ")");

ImgProfilePic.ImageUrl = imgurl;

}

Luser.Text = emp.post;

LID.Text += id;

Aname.Text = TBname.Text = emp.name;

TBAddr.Text = emp.address;

if (emp.gender.Equals("Male"))

{

RBLgenger.Items[0].Selected = true;

}

else if (emp.gender.Equals("Female"))

{

RBLgenger.Items[1].Selected = true;

}

else

{

RBLgenger.Items[2].Selected = true;

}

TBdob.Text = emp.dob.ToString("yyyy-MM-dd");

TBemail.Text = emp.Email;

TBphno.Text = emp.phno.ToString();

}

}

protected void SaveChanges(object sender, EventArgs e)

{

HttpPostedFile postedfile = FUprofileImg.PostedFile;

if (!TBoldpass.Text.Trim().Equals("") || !TBnewpass.Text.Trim().Equals("") || !TBconfirmpass.Text.Trim().Equals("") || !postedfile.FileName.Equals("") || !TBname.Text.Trim().Equals(emp.name) || !TBAddr.Text.Trim().Equals(emp.address) || !RBLgenger.SelectedValue.Equals(emp.gender) || !DateTime.Parse(TBdob.Text.Trim()).Equals(emp.dob) || !TBemail.Text.Trim().Equals(emp.Email) || !Convert.ToInt64(TBphno.Text.Trim()).Equals(emp.phno))

{

string Opass = TBoldpass.Text.Trim();

if (!Opass.Equals(""))

{

if (Convert.ToInt32(id) == emp.CheckPassword(Opass))

{

if (TBnewpass.Text.Trim().Equals(TBconfirmpass.Text.Trim()))

{

emp.password = TBconfirmpass.Text.Trim();

}

else

{

emp.password = null;

LimgiploadErr.Text = "New Passwords Do not Match";

PError.Visible = true;

PError.Style.Add("border", "2px solid red");

return;

}

}

else

{

LimgiploadErr.Text = "Old password Do not Match";

PError.Visible = true;

PError.Style.Add("border", "2px solid red");

return;

}

}

else if (!TBnewpass.Text.Trim().Equals("") || !TBconfirmpass.Text.Trim().Equals(""))

{

LimgiploadErr.Text = "Old password not Provided, Password is Unchanged";

PError.Visible = true;

PError.Style.Add("border", "2px solid white");

return;

}

if (!postedfile.FileName.Equals(""))

{

string fileExtennsion = Path.GetExtension(postedfile.FileName).ToLower();

if (fileExtennsion == ".jpg" || fileExtennsion == ".jpeg" || fileExtennsion == ".png")

{

if (emp.GetImgCount() == 1)

{

if (emp.UpdateImage(postedfile) < 0)

{

LimgiploadErr.Text = "Image Update Failed";

PError.Visible = true;

PError.Style.Add("border", "2px solid red");

return;

}

}

else if (emp.GetImgCount() == 0)

{

if (emp.InsertImage(postedfile) < 0)

{

LimgiploadErr.Text = "Image Update Failed";

PError.Visible = true;

PError.Style.Add("border", "2px solid red");

return;

}

}

}

else

{

LimgiploadErr.Text = "Only images(.jpg, .png, .jpeg) can be uploaded";

PError.Visible = true;

PError.Style.Add("border", "2px solid red");

return;

}

}

emp.name = TBname.Text.Trim();

emp.address = TBAddr.Text.Trim();

emp.gender = RBLgenger.SelectedValue;

emp.dob = DateTime.Parse(TBdob.Text.Trim());

if (!emp.Email.Equals(TBemail.Text.Trim()))

{

if (Employee.CountByEmail(TBemail.Text.Trim()) != 0)

{

LimgiploadErr.Text = "Email Already Registered, Try With Another One";

PError.Visible = true;

PError.Style.Add("border", "2px solid red");

return;

}

emp.Email = TBemail.Text.Trim();

}

else

{

emp.Email = TBemail.Text.Trim();

}

if (emp.phno != Convert.ToInt64(TBphno.Text.Trim()))

{

if (Employee.CountByPhone(Convert.ToInt64(TBphno.Text.Trim())) != 0)

{

LimgiploadErr.Text = "Phone No. Already Registered, Try With Another One";

PError.Visible = true;

PError.Style.Add("border", "2px solid red");

return;

}

emp.phno = Convert.ToInt64(TBphno.Text.Trim());

}

else

{

emp.phno = Convert.ToInt64(TBphno.Text.Trim());

}

emp.UpDateAll();

LimgiploadErr.Text = "Changes Saved Successfully";

PError.Style.Add("border", "2px solid green");

PError.Visible = true;

string imgurl = emp.LoadImage();

ImgProfilePic.ImageUrl = imgurl;

ProundedDP.Style.Add("background-image", "url(" + imgurl + ")");

}

else

{

LimgiploadErr.Text = "No Changes Detected";

PError.Style.Add("border", "2px solid white");

PError.Visible = true;

}

}

protected void Logout(object sender, EventArgs e)

{

Session.Remove("LightAdm");

if (Session["LightEmp"] != null)

{

Session.Remove("LightEmp");

}

Response.Redirect("~/employee/login.aspx");

}

protected void Logout1(object sender, EventArgs e)

{

Session.Remove("LightEmp");

Response.Redirect("./login.aspx");

}

}

}

1. Employee/Admin Forgot Password

(EforgotPassword.aspx.cs)

using OnlineCourierService.employee.Classes;

using System;

using System.Web;

using System.Web.UI;

namespace OnlineCourierService.employee

{

public partial class EforgotPassword : System.Web.UI.Page

{

protected void Page\_Load(object sender, EventArgs e) { }

private void Notifyuser(string msg, bool issuccessful)

{

if (issuccessful)

{

Lerr.Text = msg;

Lerr.Style.Add("color", "Navy");

PError.Style.Add("border", "2px solid Navy");

}

else

{

Lerr.Text = msg;

Lerr.Style.Add("color", "Red");

PError.Style.Add("border", "2px solid lightred");

}

PError.Style.Add("display", "block");

}

protected void Bpreset\_Click(object sender, EventArgs e)

{

if (Page.IsValid)

{

string email = TBunm.Text.Trim();

if (Employee.CountByEmail(email) != 1)

{

Notifyuser("Email is not Found to be Registered", false);

return;

}

string link = HttpContext.Current.Request.Url.Authority;

link += "/employee/EresetPassword.aspx";

if (Employee.GenerateResetPasswordLink(email, link) == 1)

{

Notifyuser("An email with instructions to reset your password is sent to your registered email", true);

}

else

{

Notifyuser("We Are Facing Some Internal Problem Try Again Later", false);

return;

}

}

else

{

Notifyuser("Validation Error", false);

return;

}

}

}

}

1. Employee/Admin Reset Password

(EresetPassword.aspx.cs)

using OnlineCourierService.employee.Classes;

using System;

using System.Web.UI;

namespace OnlineCourierService.employee

{

public partial class EresetPassword : System.Web.UI.Page

{

protected void Page\_Load(object sender, EventArgs e)

{

string guid = Request.QueryString["uid"];

if (guid == null)

{

Notifyuser("Password Reset link has expired or is invalid", false);

TBCnewPass.Enabled = TBnewPass.Enabled = BcngP.Enabled = false;

return;

}

if (!IsPostBack)

{

long Eid = Employee.isPasswordResetLinkValid(guid);

if (Eid < 1000)

{

Notifyuser("Password Reset link has expired or is invalid", false);

TBCnewPass.Enabled = TBnewPass.Enabled = BcngP.Enabled = false;

return;

}

Employee Emp = new Employee((int)Eid);

Emp.Filldata();

Luser.Text += " for " + Emp.Email;

Session.Add("EValidGUID", true);

}

}

private void Notifyuser(string msg, bool issuccessful)

{

if (issuccessful)

{

Lerr.Text = msg;

Lerr.Style.Add("color", "Navy");

PError.Style.Add("border", "2px solid Navy");

}

else

{

Lerr.Text = msg;

Lerr.Style.Add("color", "Red");

PError.Style.Add("border", "2px solid lightred");

}

PError.Style.Add("display", "block");

}

protected void BcngP\_Click(object sender, EventArgs e)

{

if (Page.IsValid)

{

if (Employee.ResetPassword(Request.QueryString["uid"], TBCnewPass.Text.Trim()) != 1)

{

Notifyuser("Update Password Failed !", false);

return;

}

Notifyuser("Password Changed Successfully!", true);

}

else

{

Notifyuser("Validation Error, Try Again Later", false);

return;

}

}

}

}

**Employee Side**

1. Employee Home(emp.aspx.cs)

using OnlineCourierService.employee.Classes;

using System;

namespace OnlineCourierService.employee

{

public partial class emp : System.Web.UI.Page

{

private Employee emp1;

private String id;

protected void Page\_Load(object sender, EventArgs e)

{

try

{

id = Session["LightEmp"].ToString();

emp1 = new Employee(Convert.ToInt32(id));

emp1.Filldata();

}

catch (Exception)

{

Response.Redirect("~/employee/login.aspx");

}

if (id.Equals(null))

{

Response.Redirect("~/employee/login.aspx");

}

if (emp1.IsAdmin())

{

Ppage.Visible = true;

}

if (!IsPostBack)

{

Aname.Text = emp1.name;

string imgurl = emp1.LoadImage();

if (imgurl != null)

{

ProundedDP.Style.Add("background-image", "url('" + imgurl + "')");

}

}

}

protected void Logout(object sender, EventArgs e)

{

Session.Remove("LightEmp");

if (Session["LightAdm"] != null)

{

Session.Remove("LightAdm");

}

Response.Redirect("~/employee/login.aspx");

}

protected void DDLpage\_SelectedIndexChanged(object sender, EventArgs e)

{

if (DDLpage.SelectedValue.Equals("Admin"))

{

if (Session["LightAdm"] == null)

{

Session["LightAdm"] = Session["LightEmp"];

}

Response.Redirect("~/employee/admin.aspx");

}

else if (DDLpage.SelectedValue.Equals("Employee"))

{

if (Session["LightEmp"] == null)

{

Session["LightEmp"] = Session["LightAdm"];

}

Response.Redirect("~/employee/emp.aspx");

}

}

}

}

1. Create Parcel(EcreateParcel.aspx.cs)

using OnlineCourierService.Class;

using OnlineCourierService.employee.Classes;

using System;

using System.Web.UI;

namespace OnlineCourierService.employee.EmploeeLinks

{

public partial class ECreateParcel : System.Web.UI.Page

{

protected void Page\_Load(object sender, EventArgs e){ }

private void Notifyuser(string msg, bool isSuccessful)

{

if (isSuccessful)

{

Lerrmsg.Text = msg;

Lerrmsg.Style.Add("color", "Navy");

Perrormsg.Style.Add("border", "2px solid Navy");

}

else

{

Lerrmsg.Text = msg;

Lerrmsg.Style.Add("color", "Red");

Perrormsg.Style.Add("border", "2px solid lightred");

}

Perrormsg.Style.Add("display", "block");

}

protected void DDLsReg\_SelectedIndexChanged(object sender, EventArgs e)

{

LoadsBranch();

DDLsbranch.Focus();

}

private void LoadsBranch()

{

long rid = Convert.ToInt64(DDLsReg.SelectedValue);

if (rid != -1)

{

DDLsbranch.DataSource = DataAccessLayer.GetBranchByRid((int)rid);

DDLsbranch.DataTextField = "BName";

DDLsbranch.DataValueField = "Bid";

DDLsbranch.DataBind();

}

}

protected void DDLrReg\_SelectedIndexChanged(object sender, EventArgs e)

{

LoadrBranch();

DDLrbranch.Focus();

}

private void LoadrBranch()

{

long rid = Convert.ToInt64(DDLrReg.SelectedValue);

if (rid != -1)

{

DDLrbranch.DataSource = DataAccessLayer.GetBranchByRid((int)rid);

DDLrbranch.DataTextField = "BName";

DDLrbranch.DataValueField = "Bid";

DDLrbranch.DataBind();

}

}

protected void DDLPtype\_SelectedIndexChanged(object sender, EventArgs e)

{

TogglePFood();

DDLPtype.Focus();

}

protected void DDLpackage\_SelectedIndexChanged(object sender, EventArgs e)

{

TogglePFood();

DDLpackage.Focus();

}

private void TogglePFood()

{

if (DDLPtype.SelectedValue == "CookedFood" || DDLPtype.SelectedValue == "DryFood")

{

if (DDLpackage.SelectedIndex == 0)//&& !PFood.Visible

{

PFood.Visible = true;

}

else if (DDLpackage.SelectedIndex == 1)//&& PFood.Visible

{

PFood.Visible = false;

}

}

else if (DDLPtype.SelectedValue != "CookedFood" && DDLPtype.SelectedValue != "DryFood")

{

if (PFood.Visible)

{

PFood.Visible = false;

}

}

}

protected void Bsubmit\_Click(object sender, EventArgs e)

{

if (Page.IsValid)

{

string parcelType = DDLPtype.SelectedValue;

string paymentMethod = DDLPaymentM.SelectedValue;

double weight = Convert.ToDouble(TBPweight.Text.Trim());

SenderReceiver Sender, Receiver;

Sender = new SenderReceiver("-1", TBsName.Text.Trim(), TBsEmail.Text.Trim(), TBsAddr.Text.Trim());

Receiver = new SenderReceiver("-1", TBrName.Text.Trim(), TBrEmail.Text.Trim(), TBrAddr.Text.Trim());

long sBID = Convert.ToInt64(DDLsbranch.SelectedValue);

long sRID = Convert.ToInt64(DDLsReg.SelectedValue);

long rBID = Convert.ToInt64(DDLrbranch.SelectedValue);

long rRID = Convert.ToInt64(DDLrReg.SelectedValue);

int PackagingByCustomer = Convert.ToInt32(DDLpackage.SelectedValue);

int container;

if (PFood.Visible) { container = Convert.ToInt32(TBfood.Text.Trim()); }

else { container = -1; }

Package p = new Package(parcelType, weight, Sender, Receiver, sBID, rBID, sRID, rRID, 0, 0, paymentMethod, "Pending", PackagingByCustomer, container);

string plid = p.insertPackage();

if (plid == null)

{

Notifyuser("Parcel Request Creation Failed! Try Again Later", false);

return;

}

Notifyuser("Parcel Request Placed Successfully with ID: \"" + plid + "\" Note it Down if Your are Not a Registered User", true);

}

else

{

Notifyuser("Validation Error, Data Not Saved", false);

return;

}

}

protected void LBcancel\_Click(object sender, EventArgs e)

{

Response.Redirect("~/index.aspx");

}

}

}

1. Track Parcel(EtrackParcel.aspx.cs)

using OnlineCourierService.Class;

using System;

using System.Web.UI;

namespace OnlineCourierService.employee.EmploeeLinks

{

public partial class EtrackParcel : System.Web.UI.Page

{

protected void Page\_Load(object sender, EventArgs e){ }

private void Notifyuser(string msg, bool isSuccessful)

{

if (isSuccessful)

{

Lerrmsg.Text = msg;

Lerrmsg.Style.Add("color", "Navy");

Perrormsg.Style.Add("border", "2px solid Navy");

}

else

{

Lerrmsg.Text = msg;

Lerrmsg.Style.Add("color", "Red");

Perrormsg.Style.Add("border", "2px solid lightred");

}

Perrormsg.Style.Add("display", "block");

}

private void LoadSrcResult()

{

GVsrcParcel.DataSource = Package.GetParcelsByPLID(TBsrcPLID.Text.Trim());

GVsrcParcel.DataBind();

}

protected void Bsrc\_Click(object sender, EventArgs e)

{

if (Page.IsValid) { LoadSrcResult(); }

else

{

Notifyuser("Validation Error Search Not Completed", false);

return;

}

}

}

}

1. Advance Parcel Tracking(EparcelDetails.aspx.cs)

using OnlineCourierService.Class;

using OnlineCourierService.employee.Classes;

using System;

using System.Web.UI.HtmlControls;

using System.Web.UI.WebControls;

namespace OnlineCourierService.employee.EmploeeLinks

{

public partial class EParcelDetails : System.Web.UI.Page

{

protected void Page\_Load(object sender, EventArgs e)

{

if (!IsPostBack)

{

if (Request.QueryString["PLID"] != null)

{

LoadData(Request.QueryString["PLID"]);

}

}

}

private void Notifyuser(string msg, bool isSuccessful)

{

if (isSuccessful)

{

Lerrmsg.Text = msg;

Lerrmsg.Style.Add("color", "Navy");

Perrormsg.Style.Add("border", "2px solid Navy");

}

else

{

Lerrmsg.Text = msg;

Lerrmsg.Style.Add("color", "Red");

Perrormsg.Style.Add("border", "2px solid lightred");

}

Perrormsg.Style.Add("display", "block");

}

private void LoadData(string PLID)

{

Package parcel = new Package(PLID);

parcel.FillData();

LPLID.Text += PLID;

LPtype.Text += parcel.ParcelType;

LPweight.Text += parcel.Weight + "KG";

LPRdate.Text += parcel.ReqDate;

LPpackaging.Text += (parcel.PackagingByCustomer == 0) ? "Customer" : "Lightning Corp";

if (parcel.Container != -1)

{

LPcontainer.Visible = true;

LPcontainer.Text += parcel.Container;

}

if (parcel.Distance == 0)

{

LPDistance.Text += "Yet to Be calculated";

LPinvoicePrice.Text += "Yet to Be calculated";

LPpaymentDate.Text += "Unknown";

}

else

{

LPDistance.Text += "About " + parcel.Distance + "KM";

LPinvoicePrice.Text += "Rs." + parcel.invoicePrice;

LPpaymentDate.Text += parcel.PayDate;

}

LPpaymentStat.Text += parcel.PaymentStatus;

LPpaymentMethod.Text += parcel.PaymentMethod;

LSname.Text += parcel.Sender.Name;

LSaddress.Text += parcel.Sender.Address;

LSemail.Text += parcel.Sender.Email;

if (parcel.Sender.CID == "-1") { LSisRegistered.Text += "No"; }

else

{

LSisRegistered.Text += "Yes";

Customer rec = Customer.GetinfoByEmail(parcel.Sender.Email);

LSphno.Visible = true;

LSphno.Text += rec.phno;

}

LRname.Text += parcel.Receiver.Name;

LRaddress.Text += parcel.Receiver.Address;

LRemail.Text += parcel.Receiver.Email;

if (parcel.Receiver.CID == "-1") { LRisRegistered.Text += "no"; }

else

{

LRisRegistered.Text += "Yes";

Customer rec = Customer.GetinfoByEmail(parcel.Receiver.Email);

LRphno.Visible = true;

LRphno.Text += rec.phno;

}

if (parcel.statuslist.Count != 0)

{

Pstatus.Controls.Add(GenerateStaatusDiv("stat\_\_div done", "Request Placed", parcel.ReqDate.ToString()));

double c1 = 1, c2 = 0, c = 0;

Employee emp;

parcel.statuslist.ForEach(s =>

{

if ((parcel.SourceRID != -1 || (c != 4 && c != 5)) && (parcel.DestRID != -1 || (c != 6 && c != 7)) && (parcel.SourceBID != parcel.DestBID || c != 8))

{

if (s.date != null)

{

c1++;

emp = new Employee(s.eid);

emp.Filldata();

Pstatus.Controls.Add(GenerateStaatusDiv("stat\_\_div done", s.remark, s.date.ToString()));

}

else if (s.date == null)

{

c2++;

Pstatus.Controls.Add(GenerateStaatusDiv("stat\_\_div", s.remark, ""));

}

}

c++;

});

double percent = 100 - ((c1 \* 100d) / (c1 + c2));

string p = percent + "% " + percent + "%";

PprogressBar.Style.Add("background-position", p);

}

}

private HtmlGenericControl GenerateStaatusDiv(string divClassName, string l1Text, string l2Text)

{

Label l1 = new Label();

l1.Text = l1Text;

Label l2 = new Label();

l2.Text = l2Text;

HtmlGenericControl span = new HtmlGenericControl("span");

span.Attributes.Add("class", "round");

HtmlGenericControl div = new HtmlGenericControl("div");

div.Attributes.Add("class", divClassName);

div.Controls.AddAt(0, span);

div.Controls.AddAt(1, l1);

div.Controls.AddAt(2, l2);

return div;

}

}

}

**Admin Side**

1. Admin Home(admin.aspx.cs)

using OnlineCourierService.employee.Classes;

using System;

namespace OnlineCourierService.employee

{

public partial class admin : System.Web.UI.Page

{

private Employee emp;

private String id;

protected void Page\_Load(object sender, EventArgs e)

{

try

{

id = Session["LightAdm"].ToString();

emp = new Employee(Convert.ToInt32(id));

emp.Filldata();

Aname.Text = emp.name;

}

catch (Exception)

{

Response.Redirect("~/employee/login.aspx");

}

if (id.Equals(null) || !emp.IsAdmin()){

Response.Redirect("~/employee/login.aspx");

}

if (!IsPostBack)

{

string imgurl = emp.LoadImage();

if (imgurl != null)

{

ProundedDP.Style.Add("background-image", "url('" + imgurl + "')");

}

}

}

protected void Logout(object sender, EventArgs e)

{

Session.Remove("LightAdm");

if (Session["LightEmp"] != null)

{

Session.Remove("LightEmp");

}

Response.Redirect("~/employee/login.aspx");

}

protected void DDLpage\_SelectedIndexChanged(object sender, EventArgs e)

{

if (DDLpage.SelectedValue.Equals("Admin"))

{

if (Session["LightAdm"] == null)

{

Session["LightAdm"] = Session["LightEmp"];

}

Response.Redirect("~/employee/admin.aspx");

}

else if (DDLpage.SelectedValue.Equals("Employee"))

{

if (Session["LightEmp"] == null)

{

Session["LightEmp"] = Session["LightAdm"];

}

Response.Redirect("~/employee/emp.aspx");

}

}

}

}

1. Employee Manager(empManager.aspx.cs)

using OnlineCourierService.employee.Classes;

using System;

using System.Data;

using System.IO;

using System.Web;

using System.Web.UI;

using System.Web.UI.WebControls;

namespace OnlineCourierService.employee.AdminLinks

{

public partial class empManager : System.Web.UI.Page

{

private string eid;

private Employee emp;

protected void Page\_Load(object sender, EventArgs e)

{

try

{

eid = Session["LightAdm"].ToString();

emp = new Employee(Convert.ToInt32(eid));

emp.Filldata();

Aname.Text = emp.name;

}

catch (Exception)

{

Response.Redirect("~/employee/login.aspx");

}

if (eid.Equals(null) || !emp.IsAdmin())

{

Response.Redirect("~/employee/login.aspx");

}

if (!IsPostBack)

{

string imgurl = emp.LoadImage();

if (imgurl != null)

{

ProundedDP.Style.Add("background-image", "url(" + imgurl + ")");

}

}

}

protected void Logout(object sender, EventArgs e)

{

Session.Remove("LightAdm");

if (Session["LightEmp"] != null)

{

Session.Remove("LightEmp");

}

Response.Redirect("~/employee/login.aspx");

}

protected void CreateEmp(object sender, EventArgs e)

{

LBCreateEmp.CssClass = "tab active";

LBSrcEmp.CssClass = "tab";

PcreateEmp.Visible = true;

PsrcEmp.Visible = false;

}

protected void SrcEmp(object sender, EventArgs e)

{

LBCreateEmp.CssClass = "tab";

LBSrcEmp.CssClass = "tab active";

PsrcEmp.Visible = true;

PcreateEmp.Visible = false;

}

protected void SaveChanges(object sender, EventArgs e)

{

if (Page.IsValid)

{

string name = TBname.Text.Trim();

string addr = TBAddr.Text.Trim();

string gender = RBLgenger.SelectedValue;

DateTime dob = DateTime.Parse(TBdob.Text);

string email = TBemail.Text.Trim();

long phno = Convert.ToInt64(TBphno.Text.Trim());

long bid = Convert.ToInt64(DDLbranch.SelectedValue);

int postindex = Convert.ToInt32(DDLpost.SelectedValue);

HttpPostedFile image = FUprofileImg.PostedFile;

Employee newemp = new Employee(name, addr, gender, email, dob, bid, phno, "N0password", postindex, true);

if (Employee.CountByEmail(email) != 0)

{

LErr.Text = "Same Email is Already Registered";

PError.Visible = true;

PError.Style.Add("border", "2px solid red");

return;

}

if (Employee.CountByPhone(phno) != 0)

{

LErr.Text = "Same Phone No is Already Registered";

PError.Visible = true;

PError.Style.Add("border", "2px solid red");

return;

}

if (!image.FileName.Equals(""))

{

string fileExtennsion = Path.GetExtension(image.FileName).ToLower();

if (fileExtennsion != ".jpg" && fileExtennsion != ".jpeg" && fileExtennsion != ".png")

{

LErr.Text = "Only images(.jpg, .png, .jpeg) can be uploaded";

PError.Visible = true;

PError.Style.Add("border", "2px solid red");

return;

}

goto INSERT;

}

INSERT:

string id = Convert.ToString(newemp.InsertInfo());

newemp.InsertImage(image);

LErr.Text = "Successful! Account Created With Employee ID :" + id + " and Default password";

PError.Style.Add("border", "2px solid lightgreen");

PError.Visible = true;

}

else

{

LErr.Text = "Validation Failed! Data Not Saved";

PError.Style.Add("border", "2px solid red");

PError.Visible = true;

return;

}

}

protected void HidePErr(object sender, EventArgs e)

{

PError.Visible = false;

}

protected void ChangeSrchbehavior(object sender, EventArgs e)

{

if (DDlsrchbby.SelectedValue.ToString().Equals("EID"))

{

RFVpostSrc.Visible = DDLSrcPost.Visible = false;

TBsrc.Text = String.Empty;

}

else if (DDlsrchbby.SelectedValue.ToString().Equals("BID"))

{

RFVpostSrc.Visible = DDLSrcPost.Visible = true;

TBsrc.Text = String.Empty;

}

}

protected void SearchByEid(object sender, EventArgs e)

{

if (Page.IsValid)

{

LoadGVDataByEid();

}

else

{

LErr.Text = "Validation Error Search Not Performed";

LErr.ForeColor = System.Drawing.Color.Red;

PError.Style.Add("border", "2px solid lightRed");

PError.Visible = true;

}

}

private void LoadGVDataByEid()

{

DataTable data = null;

if (DDlsrchbby.SelectedValue.Equals("EID"))

{

data = Employee.GetEmpinfoByAdmin(Convert.ToInt64(TBsrc.Text.Trim()));

}

else if (DDlsrchbby.SelectedValue.Equals("BID"))

{

data = Employee.GetEmpinfoByAdmin2(Convert.ToInt64(TBsrc.Text.Trim()), Convert.ToInt32(DDLSrcPost.SelectedValue));

}

if (data == null || data.Rows.Count <= 0)

{

data.Rows.Add(data.NewRow());

GVEmp.DataSource = data;

GVEmp.DataBind();

GVEmp.Rows[0].Cells.Clear();

GVEmp.Rows[0].Cells.Add(new TableCell());

GVEmp.Rows[0].Cells[0].ColumnSpan = data.Columns.Count;

GVEmp.Rows[0].Cells[0].Text = "No Data Found";

GVEmp.Rows[0].Cells[0].HorizontalAlign = HorizontalAlign.Center;

}

else

{

GVEmp.DataSource = data;

GVEmp.DataBind();

}

}

protected void GVEmp\_RowDataBound(object sender, GridViewRowEventArgs e)

{

if (e.Row.RowType == DataControlRowType.Header)

{

e.Row.Cells[6].Style.Add("Display", "none");

e.Row.Cells[4].Style.Add("Display", "none");

}

else if (e.Row.RowType is DataControlRowType.DataRow)

{

if (e.Row.Cells[0].Text.Equals(eid))

{

e.Row.Cells[9].Controls.Clear();

e.Row.Cells[9].Text = "Not Editable";

}

e.Row.Cells[6].Style.Add("Display", "none");

e.Row.Cells[4].Style.Add("Display", "none");

}

}

protected void DDLRID\_SelectedIndexChanged(object sender, EventArgs e)

{

((DropDownList)(GVEmp.Rows[GVEmp.EditIndex].Cells[7].FindControl("DDLBID"))).SelectedValue = "-1";

}

protected void GVEmp\_RowCommand(object sender, GridViewCommandEventArgs e)

{

if (e.CommandName.Equals("EditROW"))

{

int rowindex = ((GridViewRow)((Button)e.CommandSource).NamingContainer).RowIndex;

LoadGVDataByEid();

string selectedBID = ((Label)(GVEmp.Rows[rowindex].Cells[7].FindControl("Label3"))).Text;

GVEmp.EditIndex = rowindex;

LoadGVDataByEid();

((DropDownList)(GVEmp.Rows[rowindex].Cells[7].FindControl("DDLBID"))).SelectedValue = selectedBID;

HttpCookie oldcollection = new HttpCookie("OLD");

oldcollection["OLD\_EID"] = e.CommandArgument.ToString();

oldcollection["OLD\_BID"] = ((DropDownList)GVEmp.Rows[rowindex].FindControl("DDLBID")).SelectedValue.ToString();

oldcollection["OLD\_POSTINDEX"] = ((DropDownList)GVEmp.Rows[rowindex].FindControl("DDLpost")).SelectedValue.ToString();

oldcollection["OLD\_status"] = ((DropDownList)GVEmp.Rows[rowindex].FindControl("DDLstatus")).SelectedValue.ToString();

Response.Cookies.Add(oldcollection);

}

else if (e.CommandName.Equals("CancelEDIT"))

{

GVEmp.EditIndex = -1;

LoadGVDataByEid();

}

else if (e.CommandName.Equals("UpdateROW"))

{

if (Page.IsValid)

{

int rowindex = ((GridViewRow)((Button)e.CommandSource).NamingContainer).RowIndex;

long BID = Convert.ToInt64(((DropDownList)GVEmp.Rows[rowindex].FindControl("DDLBID")).SelectedValue);

int POSTINDEX = Convert.ToInt32(((DropDownList)GVEmp.Rows[rowindex].FindControl("DDLpost")).SelectedValue);

string status = ((DropDownList)GVEmp.Rows[rowindex].FindControl("DDLstatus")).SelectedValue;

HttpCookie oldcollection = Request.Cookies["OLD"];

long OLD\_EID = Convert.ToInt64(oldcollection["OLD\_EID"].ToString());

long OLD\_BID = Convert.ToInt64(oldcollection["OLD\_BID"].ToString());

int OLD\_POSTINDEX = Convert.ToInt32(oldcollection["OLD\_POSTINDEX"].ToString());

string OLD\_status = oldcollection["OLD\_status"].ToString();

if (Employee.UpdateEmpbyAdmin(OLD\_EID, OLD\_BID, OLD\_POSTINDEX, OLD\_status, BID, POSTINDEX, status) > 0)

{

GVEmp.EditIndex = -1;

LoadGVDataByEid();

LErr.Text = "Row with BID :" + e.CommandArgument + " is Updated Successflly";

LErr.ForeColor = System.Drawing.Color.SteelBlue;

PError.Style.Add("border", "2px solid Navy");

PError.Visible = true;

}

else

{

GVEmp.EditIndex = -1;

LoadGVDataByEid();

LErr.Text = "Row with EID : " + e.CommandArgument + " not Saved Due to Data Conflict";

LErr.ForeColor = System.Drawing.Color.Red;

PError.Style.Add("border", "2px solid lightRed");

PError.Visible = true;

}

}

else

{

LErr.Text = "Validation Error Data not Saved";

LErr.ForeColor = System.Drawing.Color.Red;

PError.Style.Add("border", "2px solid lightRed");

PError.Visible = true;

}

}

}

}

}

1. Regional Branch Manager(regBManager.aspx.cs)

using OnlineCourierService.employee.Classes;

using System;

using System.Web.UI;

using System.Web.UI.WebControls;

namespace OnlineCourierService.employee.AdminLinks

{

public partial class regBManager : System.Web.UI.Page

{

private string id;

private Employee emp;

protected void Page\_Load(object sender, EventArgs e)

{

try

{

id = Session["LightAdm"].ToString();

emp = new Employee(Convert.ToInt32(id));

emp.Filldata();

Aname.Text = emp.name;

}

catch (Exception) {Response.Redirect("~/employee/login.aspx");}

if (id.Equals(null) || !emp.IsAdmin()){

Response.Redirect("~/employee/login.aspx");

}

if (!IsPostBack){

string imgurl = emp.LoadImage();

if (imgurl != null){

ProundedDP.Style.Add("background-image", "url(" + imgurl + ")");

}

}

}

protected void Logout(object sender, EventArgs e)

{

Session.Remove("LightAdm");

if (Session["LightEmp"] != null){

Session.Remove("LightEmp");

}

Response.Redirect("~/employee/login.aspx");

}

protected void CreateRb(object sender, EventArgs e)

{

LBCreateREGb.CssClass = "tab active";

LBSrcREGb.CssClass = "tab";

PcreateRb.Visible = true;

PSrcRb.Visible = false;

}

protected void SrcRb(object sender, EventArgs e)

{

LBCreateREGb.CssClass = "tab";

LBSrcREGb.CssClass = "tab active";

PSrcRb.Visible = true;

PcreateRb.Visible = false;

}

protected void IsAlreadyRegistered(object sender, EventArgs e)

{

if (Branch.GetRBidByRegID(Convert.ToInt32(DDLReg.SelectedValue)) != -1){

LDdlError.Text = "This Region Already have a Regional Branch";

LDdlError.Visible = true;

HFalreadyExists.Value = "true";

}else{

LDdlError.Visible = false;

HFalreadyExists.Value = "false";

}

}

protected void Submit(object sender, EventArgs e)

{

if (Page.IsValid && !Convert.ToBoolean(HFalreadyExists.Value)){

int regid = Convert.ToInt32(DDLReg.SelectedValue);

string Bname = TBRegBname.Text.Trim();

string Baddr = TBRegBaddr.Text.Trim();

double lat = Convert.ToDouble(TBLat.Text.Trim());

double lon = Convert.ToDouble(TBLong.Text.Trim());

if (Branch.CountB(Bname, Baddr, lat, lon) == 0){

int rbid = Branch.CreateRegBranch(regid, Bname, Baddr, lat, lon);

if (rbid != -1){

LErr.Text = "Regional Branch Created with Regional Branch ID : " + rbid;

LErr.ForeColor = System.Drawing.Color.Green;

PError.Style.Add("border", "2px solid lightgreen");

PError.Visible = true;

}else{

LErr.Text = "Regional Branch Creation FAILED, please contact system administratior !";

LErr.ForeColor = System.Drawing.Color.Red;

PError.Style.Add("border", "2px solid lightred");

PError.Visible = true;

}

}else{

LErr.Text = "Branch Name/Branch Address/Latetude/Longitude Already Present ! Contact Administrator or try with another one";

LErr.ForeColor = System.Drawing.Color.Red;

PError.Style.Add("border", "2px solid lightred");

PError.Visible = true;

}

}else{

LErr.Text = "Validation Error! Data Not Saved";

LErr.ForeColor = System.Drawing.Color.Red;

PError.Style.Add("border", "2px solid lightred");

PError.Visible = true;

return;

}

}

protected void HidePErr(object sender, EventArgs e)

{

PError.Visible = false;

}

protected void OBSRegB\_Updated(object sender, ObjectDataSourceStatusEventArgs e)

{

if (e.ReturnValue is int && (int)e.ReturnValue > 0){

e.AffectedRows = (int)e.ReturnValue;

}

}

protected void GVRegB\_RowUpdated(object sender, GridViewUpdatedEventArgs e)

{

if (e.AffectedRows < 1){

e.KeepInEditMode = true;

LErr.Text = "Row with Regonal Branch ID :" + e.Keys[0].ToString() + " is not Updated Due to Data Conflict ";

LErr.ForeColor = System.Drawing.Color.Red;

PError.Style.Add("border", "2px solid lightRed");

PError.Visible = true;

}else{

LErr.Text = "Row with Regonal Branch ID :" + e.Keys[0].ToString() + " is Updated Successflly";

LErr.ForeColor = System.Drawing.Color.SteelBlue;

PError.Style.Add("border", "2px solid Navy");

PError.Visible = true;

}

}

}

}

1. Sub-Branch Manager(subBManager.aspx.cs)

using OnlineCourierService.employee.Classes;

using System;

using System.Web.UI;

using System.Web.UI.WebControls;

namespace OnlineCourierService.employee.AdminLinks

{

public partial class subBManager : System.Web.UI.Page

{

private string id;

private Employee emp;

protected void Page\_Load(object sender, EventArgs e)

{

try

{

id = Session["LightAdm"].ToString();

emp = new Employee(Convert.ToInt32(id));

emp.Filldata();

Aname.Text = emp.name;

}

catch (Exception)

{

Response.Redirect("~/employee/login.aspx");

}

if (id.Equals(null) || !emp.IsAdmin())

{

Response.Redirect("~/employee/login.aspx");

}

if (!IsPostBack)

{

string imgurl = emp.LoadImage();

if (imgurl != null)

{

ProundedDP.Style.Add("background-image", "url(" + imgurl + ")");

}

}

}

protected void Logout(object sender, EventArgs e)

{

Session.Remove("LightAdm");

if (Session["LightEmp"] != null)

{

Session.Remove("LightEmp");

}

Response.Redirect("~/employee/login.aspx");

}

protected void HidePErr(object sender, EventArgs e)

{

PError.Visible = false;

}

protected void Createb(object sender, EventArgs e)

{

LBCreateB.CssClass = "tab active";

LBSrcB.CssClass = "tab";

PCreateB.Visible = true;

PSrcB.Visible = false;

}

protected void Srcb(object sender, EventArgs e)

{

LBSrcB.CssClass = "tab active";

LBCreateB.CssClass = "tab";

PSrcB.Visible = true;

PCreateB.Visible = false;

}

protected void SaveChanges(object sender, EventArgs e)

{

if (Page.IsValid)

{

long rid = Convert.ToInt64(DDLRegB.SelectedValue);

string Bname = TBSubBname.Text.Trim();

string Baddr = TBSubBaddr.Text.Trim();

double lat = Convert.ToDouble(TBLat.Text.Trim());

double lon = Convert.ToDouble(TBLong.Text.Trim());

if (Branch.CountB(Bname, Baddr, lat, lon) == 0)

{

int rbid = Branch.CreateSubBranch(rid, Bname, Baddr, lat, lon);

if (rbid != -1)

{

LErr.Text = "Regional Branch Created with Regional Branch ID : " + rbid;

LErr.ForeColor = System.Drawing.Color.Green;

PError.Style.Add("border", "2px solid lightgreen");

PError.Visible = true;

}

else

{

LErr.Text = "Regional Branch Creation FAILED, please contact system administratior !";

LErr.ForeColor = System.Drawing.Color.Red;

PError.Style.Add("border", "2px solid lightred");

PError.Visible = true;

}

}

else

{

LErr.Text = "Branch Name/Branch Address/Latetude/Longitude Already Present ! Contact Administrator or try with another one";

LErr.ForeColor = System.Drawing.Color.Red;

PError.Style.Add("border", "2px solid lightred");

PError.Visible = true;

}

}

else

{

LErr.Text = "Validation Error! Data Not Saved";

LErr.ForeColor = System.Drawing.Color.Red;

PError.Visible = true;

return;

}

}

protected void OBSSubB\_Updated(object sender, ObjectDataSourceStatusEventArgs e)

{

if (e.ReturnValue is int && (int)e.ReturnValue > 0)

{

e.AffectedRows = (int)e.ReturnValue;

}

}

protected void GVSubB\_RowUpdated(object sender, GridViewUpdatedEventArgs e)

{

if (e.AffectedRows < 1)

{

e.KeepInEditMode = true;

LErr.Text = "Row with Sub-Branch ID :" + e.Keys[0].ToString() + " is not Updated Due to Data Conflict ";

LErr.ForeColor = System.Drawing.Color.Red;

PError.Style.Add("border", "2px solid lightRed");

PError.Visible = true;

}

else

{

LErr.Text = "Row with Sub-Branch ID :" + e.Keys[0].ToString() + " is Updated Successflly";

LErr.ForeColor = System.Drawing.Color.SteelBlue;

PError.Style.Add("border", "2px solid Navy");

PError.Visible = true;

}

}

}

}

1. Parcel-Rates Manager(ratesManager.aspx.cs)

using OnlineCourierService.employee.Classes;

using System;

using System.Web.UI.WebControls;

namespace OnlineCourierService.employee.AdminLinks

{

public partial class ratesManager : System.Web.UI.Page

{

private string id;

private Employee emp;

protected void Page\_Load(object sender, EventArgs e)

{

try

{

id = Session["LightAdm"].ToString();

emp = new Employee(Convert.ToInt32(id));

emp.Filldata();

Aname.Text = emp.name;

}

catch (Exception)

{

Response.Redirect("~/employee/login.aspx");

}

if (id.Equals(null) || !emp.IsAdmin())

{

Response.Redirect("~/employee/login.aspx");

}

if (!IsPostBack)

{

string imgurl = emp.LoadImage();

if (imgurl != null)

{

ProundedDP.Style.Add("background-image", "url(" + imgurl + ")");

}

}

}

protected void Logout(object sender, EventArgs e)

{

Session.Remove("LightAdm");

if (Session["LightEmp"] != null)

{

Session.Remove("LightEmp");

}

Response.Redirect("~/employee/login.aspx");

}

protected void InsertNewRate(object sender, EventArgs e)

{

if (DataAccessLayer.CountRateWeight(Convert.ToInt32(((TextBox)GVParcelRates.FooterRow.FindControl("TBweight")).Text.Trim())) != 0)

{

LErr.Text = "Row is not Inserted as Similar Weight value already present";

LErr.ForeColor = System.Drawing.Color.Red;

PError.Style.Add("border", "2px solid lightRed");

PError.Visible = true;

return;

}

OBSRates.InsertParameters["Weight"].DefaultValue =

((TextBox)GVParcelRates.FooterRow.FindControl("TBweight")).Text.Trim();

OBSRates.InsertParameters["Basic"].DefaultValue =

((TextBox)GVParcelRates.FooterRow.FindControl("TBBasic")).Text.Trim();

OBSRates.InsertParameters["Business"].DefaultValue =

((TextBox)GVParcelRates.FooterRow.FindControl("TBBusiness")).Text.Trim();

OBSRates.InsertParameters["DryFood"].DefaultValue =

((TextBox)GVParcelRates.FooterRow.FindControl("TBDryFood")).Text.Trim();

OBSRates.InsertParameters["CookedFood"].DefaultValue =

((TextBox)GVParcelRates.FooterRow.FindControl("TBCookedFood")).Text.Trim();

OBSRates.InsertParameters["Gift"].DefaultValue =

((TextBox)GVParcelRates.FooterRow.FindControl("TBGift")).Text.Trim();

OBSRates.InsertParameters["Electronics"].DefaultValue =

((TextBox)GVParcelRates.FooterRow.FindControl("TBElectronics")).Text.Trim();

OBSRates.Insert();

LErr.Text = "Row is Inserted Successflly";

LErr.ForeColor = System.Drawing.Color.SteelBlue;

PError.Style.Add("border", "2px solid Navy");

PError.Visible = true;

}

protected void GVParcelRates\_RowUpdated(object sender, GridViewUpdatedEventArgs e)

{

if (e.AffectedRows < 1)

{

e.KeepInEditMode = true;

LErr.Text = "Row with Rate ID :" + e.Keys[0].ToString() + " is not Updated Due to Data Conflict / Similar Weight value already present";

LErr.ForeColor = System.Drawing.Color.Red;

PError.Style.Add("border", "2px solid lightRed");

PError.Visible = true;

}

else

{

LErr.Text = "Row with Rate ID :" + e.Keys[0].ToString() + " is Updated Successflly";

LErr.ForeColor = System.Drawing.Color.SteelBlue;

PError.Style.Add("border", "2px solid Navy");

PError.Visible = true;

}

}

protected void HidePErr(object sender, EventArgs e)

{

PError.Visible = false;

}

protected void OBSRates\_Updated(object sender, ObjectDataSourceStatusEventArgs e)

{

if (e.ReturnValue is int && (int)e.ReturnValue > 0)

{

e.AffectedRows = (int)e.ReturnValue;

}

}

}

}

1. **Code Efficiency**

**Code efficiency** is a broad term used to depict the reliability, speed and programming methodology used in developing codes for an application. Code efficiency is directly linked with algorithmic efficiency and the speed of runtime execution for software. It is the key element in ensuring high performance. The goal of code efficiency is to reduce resource consumption and completion time as much as possible with minimum risk to the business or operating environment. The software product quality can be accessed and evaluated with the help of the efficiency of the code used.

# Steps Taken for Efficient Coding in this Project

### Load CSS first : The CSS content of the site is placed in the head section of the page.

### To understand the reasoning here, we need to understand a little bit about how browsers achieve their incredible speed. When downloading a page, the browser will attempt to start rendering the application as soon as it has any content. Often what it renders is something of a guessing game because the browser doesn’t know what content on the page may invalidate the guesses it has made. When the browser realizes that it has made an incorrect guess about how the page should be rendered, then all the work that was done needs to be thrown out and started over again. One of the things which causes one of these reflows is the addition of a new stylesheet. Load style sheets first to avoid having a style that alters an already-rendered element.

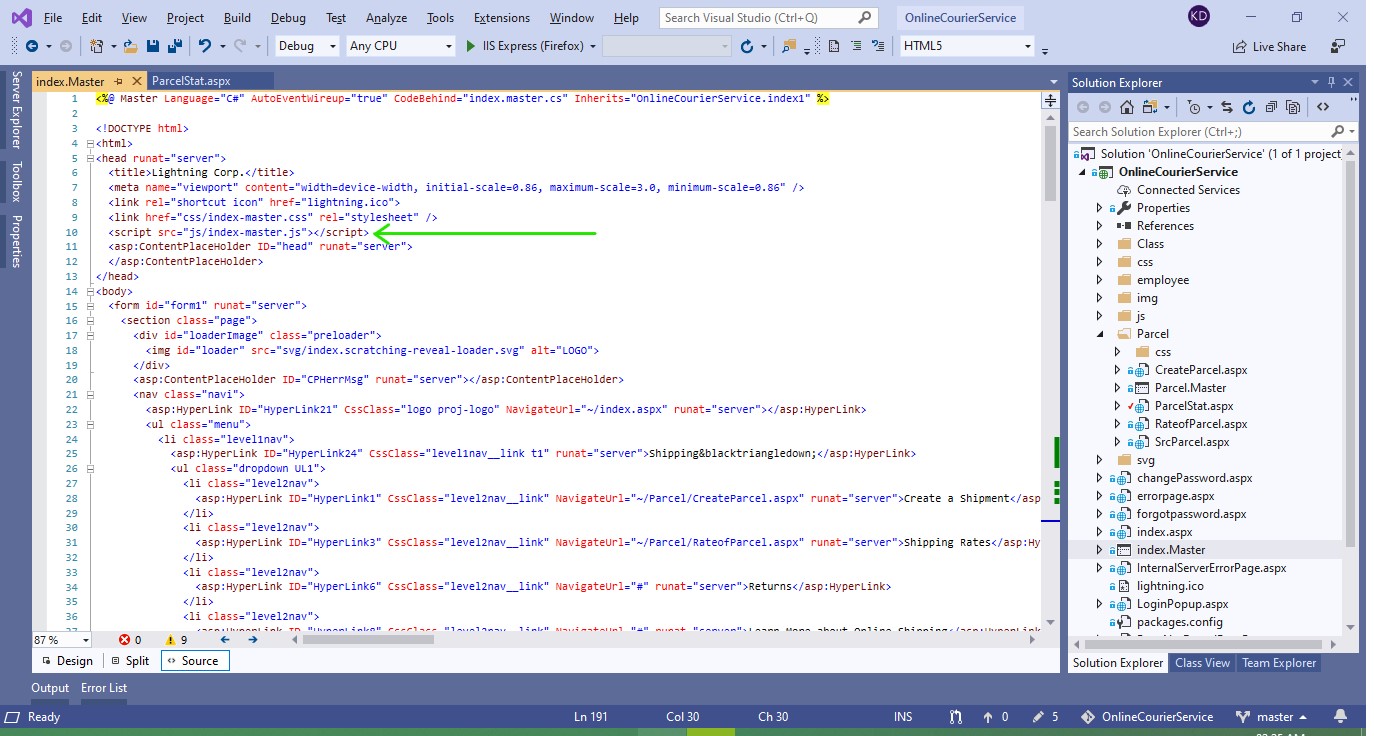


Figure : Stylesheets (\*.css files) is linked in the head tag of the Master Page

### Load JavaScript last : JavaScript is a complete about-face from CSS, and should be loaded last. This is because we want the page to render as quickly as possible, and JavaScript is not typically necessary for the initial render. Users will typically take a moment to read the page and decide what to do next. This window is be used to load scripts in the background and light up the page’s interaction.

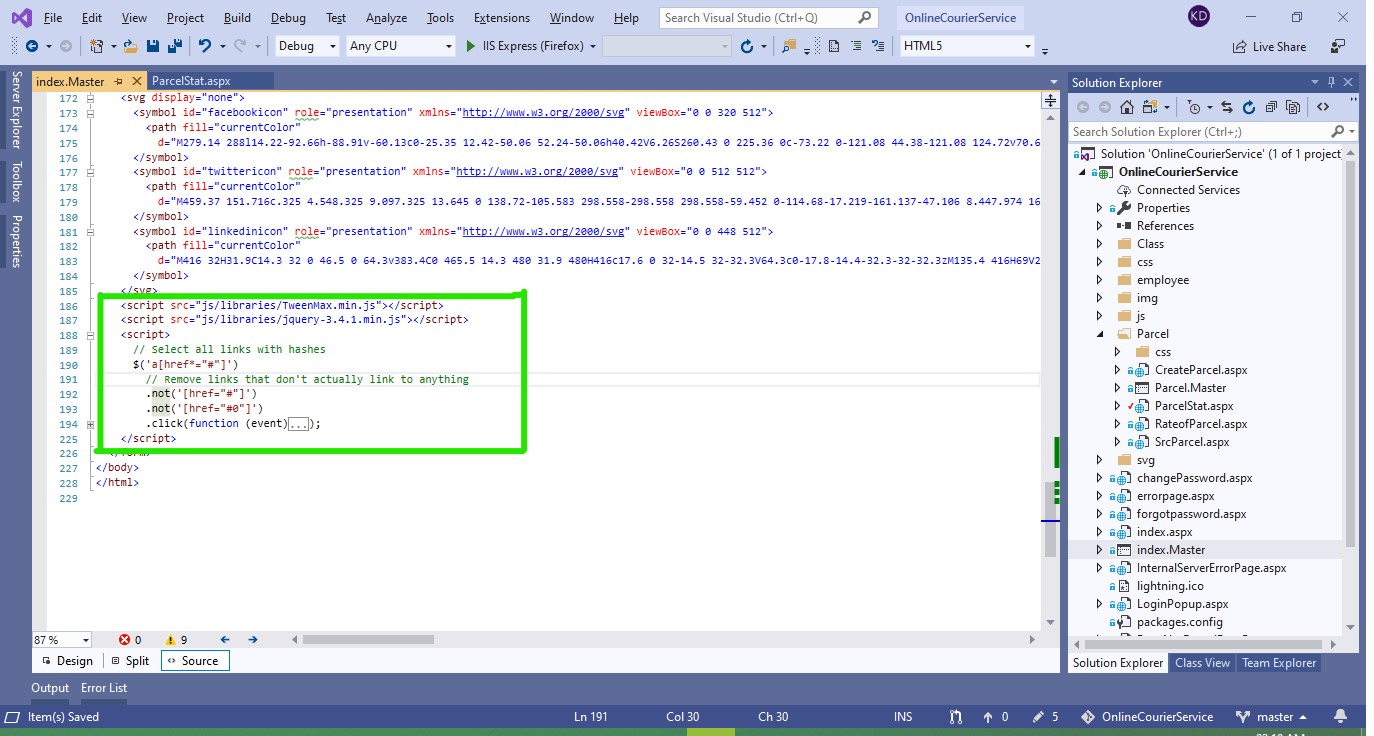


Figure : JavaScript script tag specified at the bottom of the master page

### Shrink images : In an ideal world, site would contain no images at all. It is typically a lot more efficient to use inline-SVG or CSS tricks to create vector art for pages because they are far smaller than raster images. Figuring out the right encoding settings can be difficult, but there are some really impressive services to do it for you.

I have used svg when ever possible like in Logos, Icons and Preloaders.

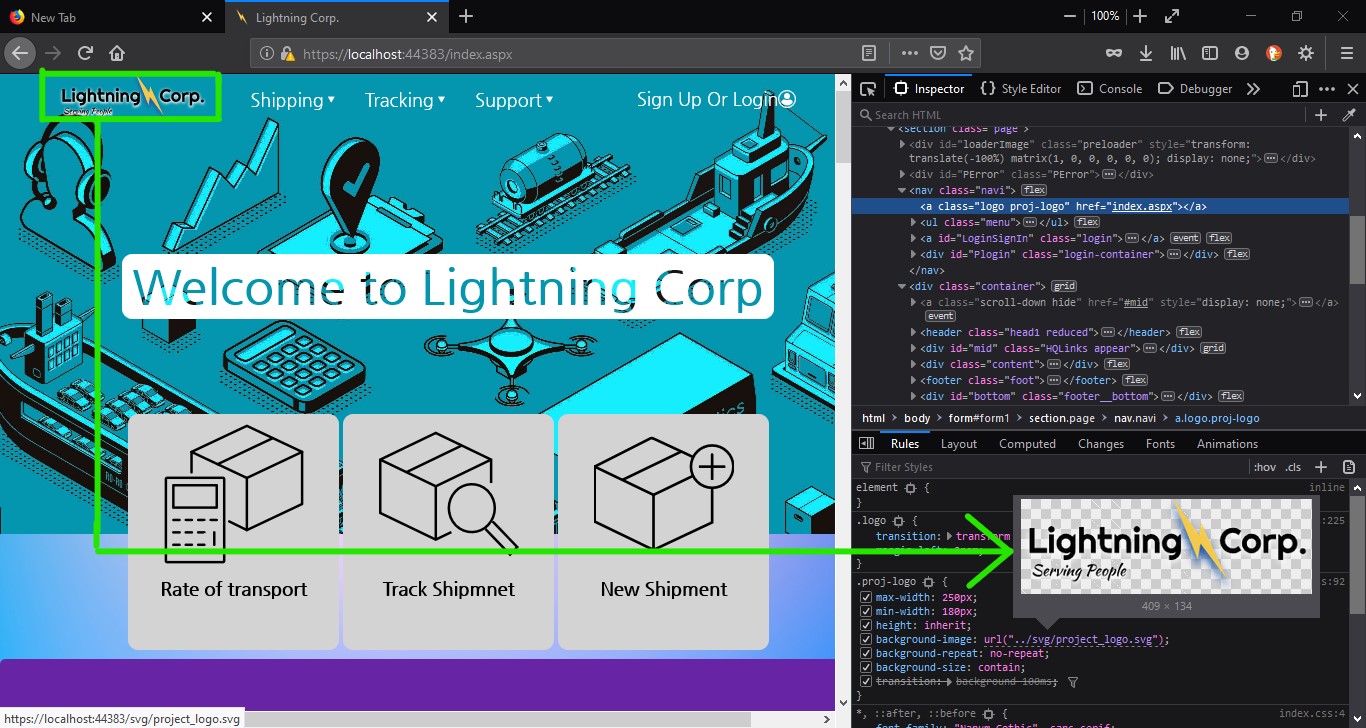


Figure : svg is used as Logo in the Home page of the project.

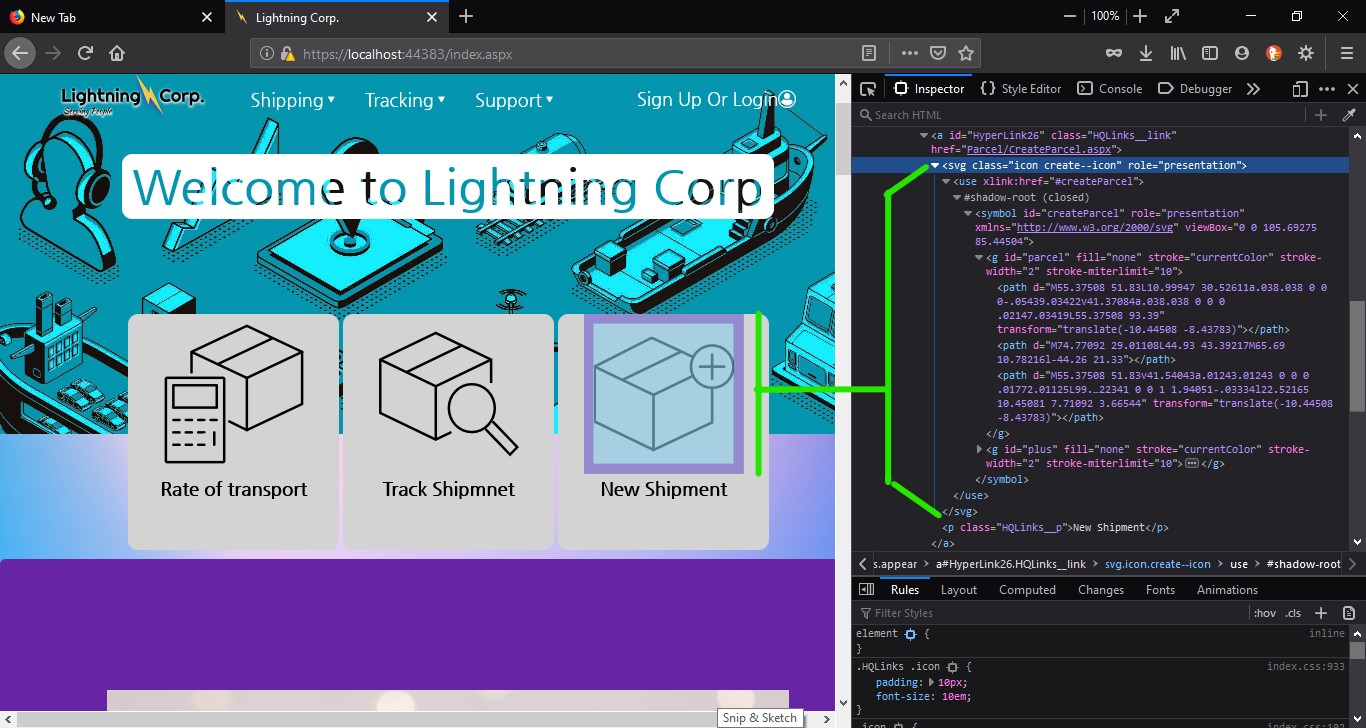


Figure : Inline svg is used as an icon in the Home page of the project

I have also used [JPNG.svg](https://codepen.io/shshaw/full/LVKEdv) (<https://codepen.io/shshaw/full/LVKEdv>) to shrink down raster images used in my project and used

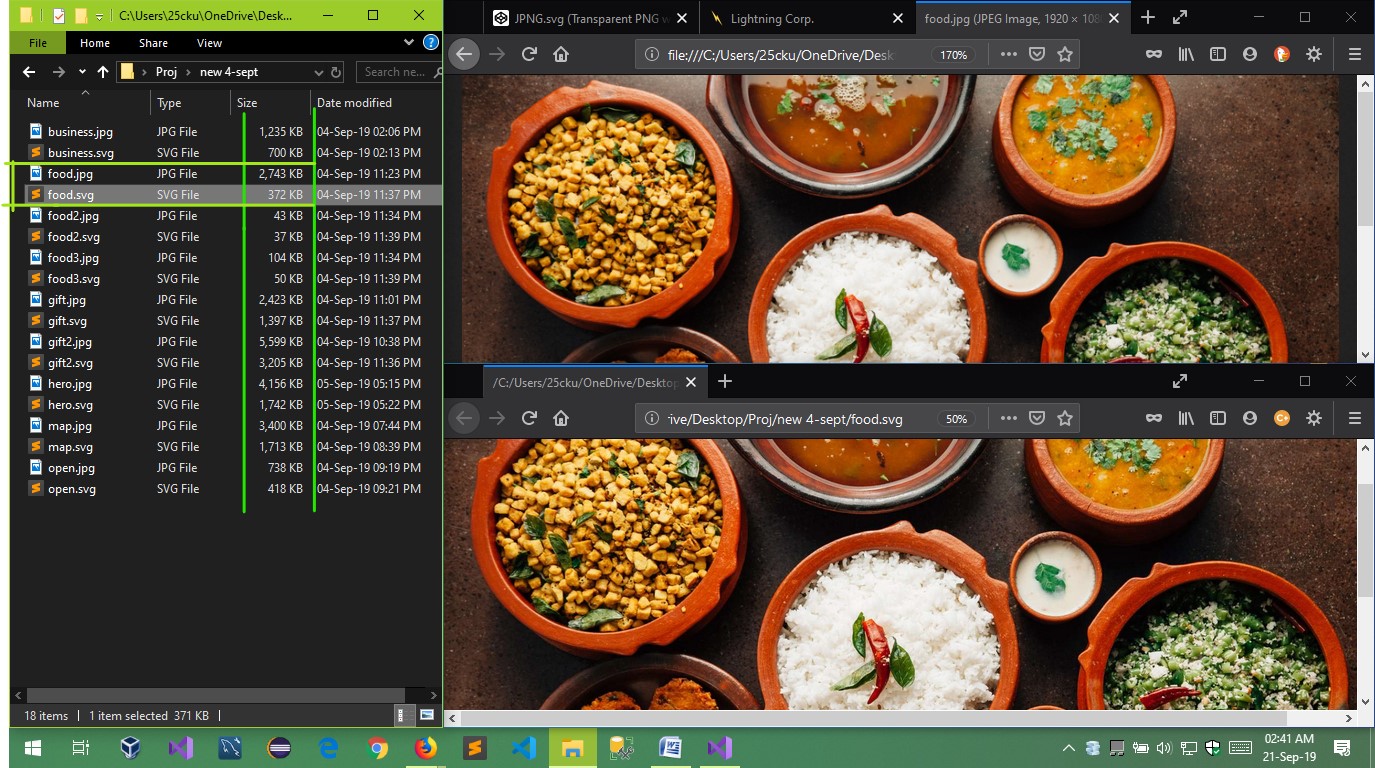


Figure : Comparison between svg and raster(JPG) imge size

### Cache your pages : Very frequently, the data on the pages changes at a slow pace. As an example, the ‘Advance Parcel Tracking’ Page fetches high bandwidth of date from Our DataBase, So If the page is updated in real time, but the data changes are not significant enough to bother re-querying the database. Instead of taking a hit going to the database and re-rendering a complex looking page, we can shove the page into a cache and serve subsequent requests using that data.

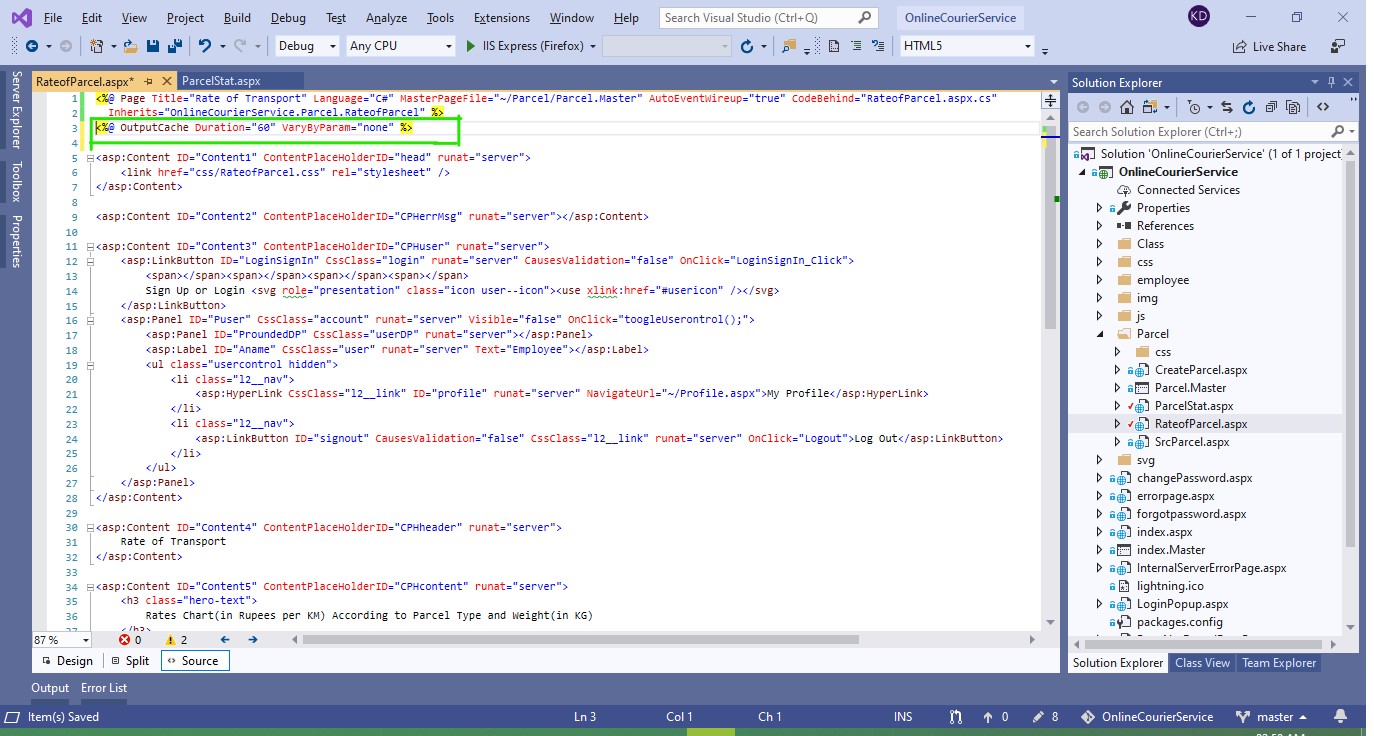


Figure : Parcel Rates web Page is Cached for 60 seconds

### Cache parts of your pages : You may want to cache only part of your page; this is colloquially known as donut hole caching. It is a useful approach when you have user-specific data mixed with general data on the same page. The user data varies by the user, while the rest of the page is the same for all users.

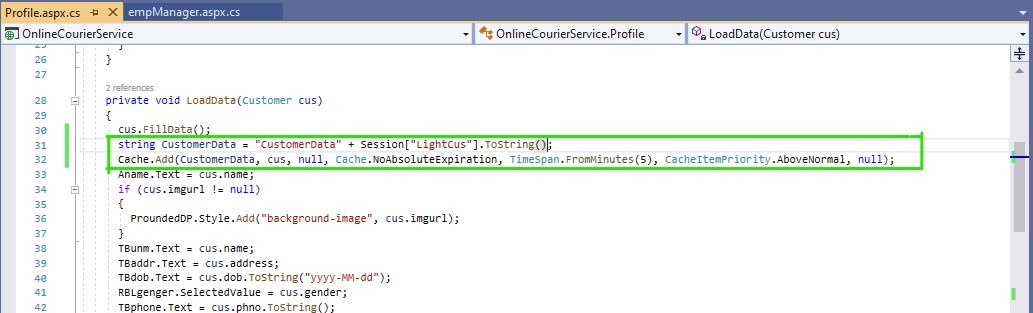


Figure : In Customer profile page only customer object with old Data is cached

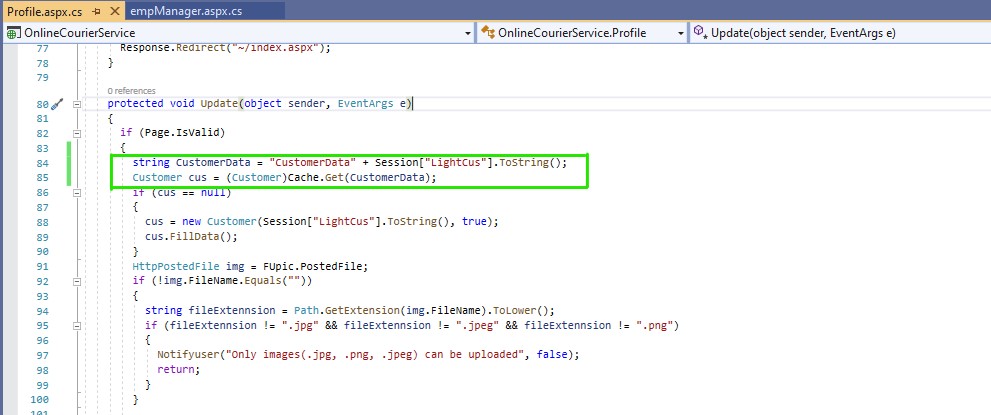


Figure : In Customer profile old data is fetched from cache when updating info. to avoid data concurrency in DataBase