
National University
of computer and emerging sciences

# Test Report

# Tourister

## Team # 8

| Member Name | Member Roll # |
|---|---|
| Wisha Riaz | 15L-4005 |
| Aqsa Noor | 15L-4049 |
| Ifrah Nadeem | 15L-4224 |
| Alina Sajid | 14L-4379 |
| Bilal Iqbal | 14L-4143 |
| Dawood Umer | 14L-4161 |

## Table of Contents

# 1. Membership

## 1.1 Code Snippets

Update Membership

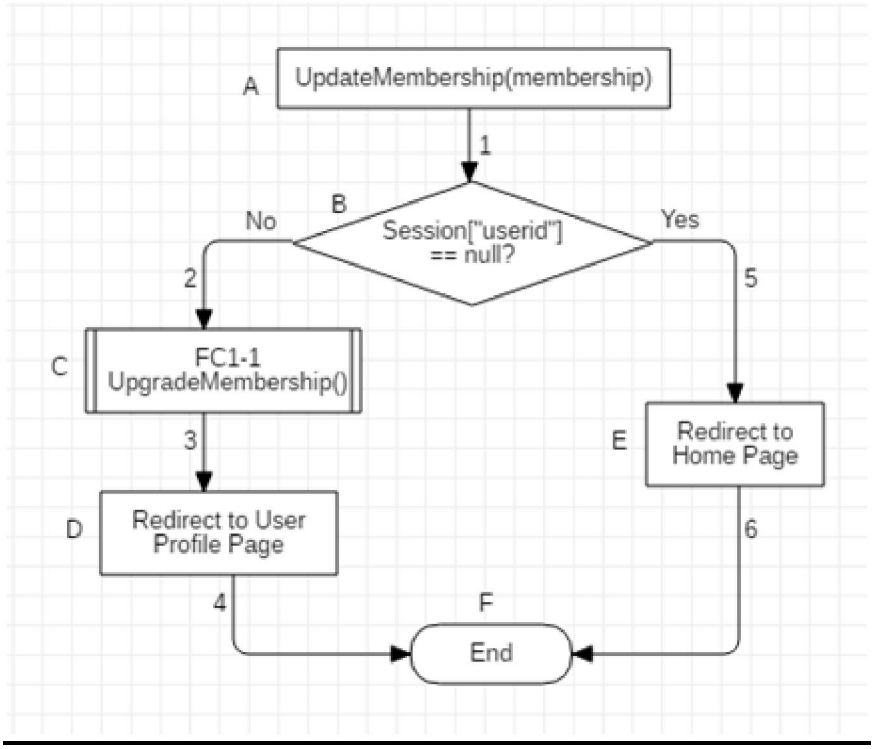
```
public ActionResult UpdateMembership(int Membership)
{
    if (Session["userid"] == null)
    {
        return RedirectToAction("Index", "Home");
    }
    int userid = int.Parse(Session["userid"].ToString());
    DAL.UpdateMembership(Membership, userid);
    return RedirectToAction("UserProfile", "UserMain");

}
```

Update Membership DAL

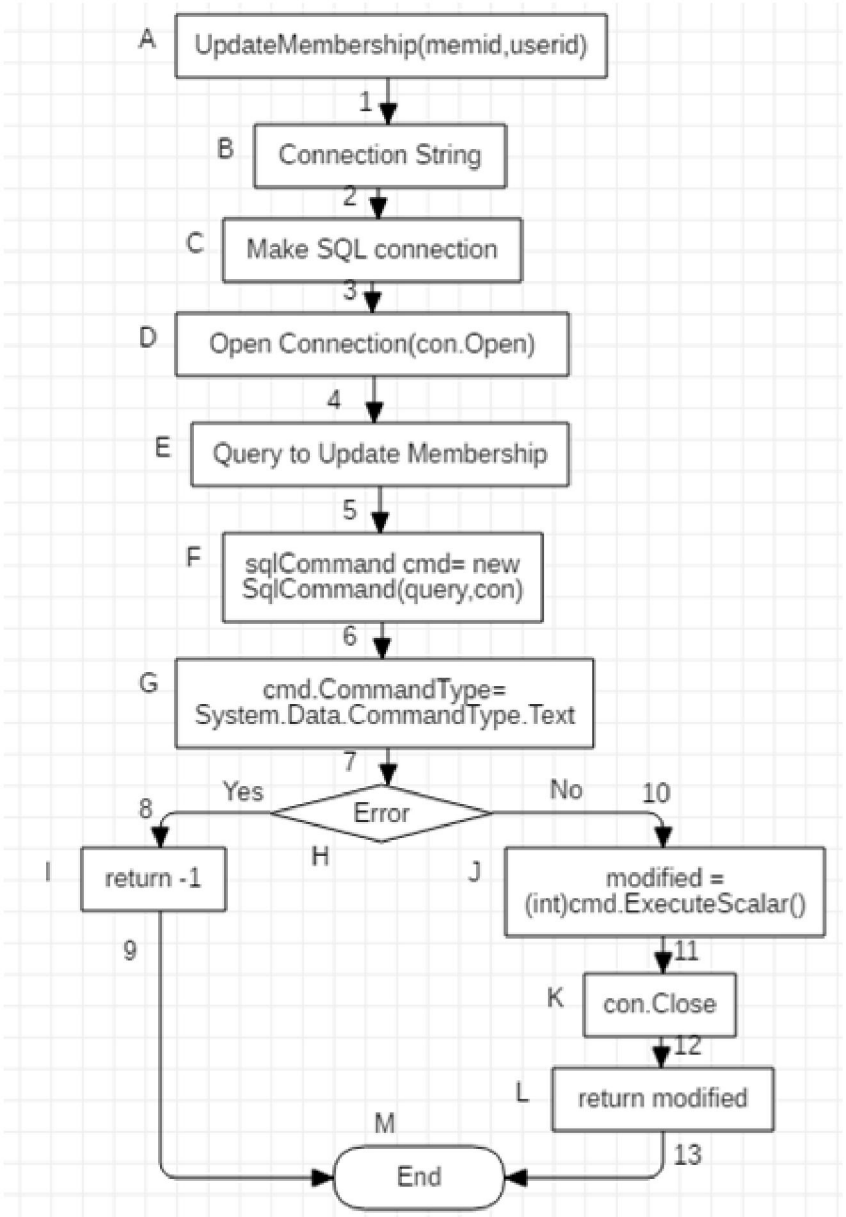
```
public static int UpdateMembership(int Membershipid, int userid)
{
    string connectionString = ConfigurationSettings.AppSettings["ConnectionString"].ToString();
    SqlConnection con = new SqlConnection(connectionString);
    con.Open();
    string query = "UPDATE [User] SET MembershipId = " + Membershipid + " WHERE Id = " + userid;
    SqlCommand cmd = new SqlCommand(query, con);
    cmd.CommandType = System.Data.CommandType.Text;
    int modified = -1;
    try
    {
        modified = (int)cmd.ExecuteScalar();

    }
    catch (Exception ex)
    {
        return modified;
    }
    con.Close();
    return modified;
}
```

## 1.2 Flow Charts

FC1 Update Membership




FC1-1 Update Membership DAL




## 1.3 Test Cases and Coverage

**Statement Coverage**

FC1

1. A1 – B5 – E6
2. A1 – B2 – C3 –FC1-1 – D4

FC1-1

3. A1 – B2 – C3 – D4 – E5 – F6 – G7 – H8 –I9
4. A1 – B2 – C3 – D4 – E5 – F6 – G7 – H10 – J11 – K12 –L13

Statement Coverage = 17 / 17 = 100 %

**Branch Coverage**

FC1

1. A1 – B5 – E6
2. A1 – B2 – C3 –FC1-1 – D4

FC1-1

3. A1 – B2 – C3 – D4 – E5 – F6 – G7 – H8 –I9
4. A1 – B2 – C3 – D4 – E5 – F6 – G7 – H10 – J11 – K12 –L13

Branch Coverage = 4 / 4 = 100 %

**Path Coverage**

FC1

1. A1 – B5 – E6
2. A1 – B2 – C3 –FC1-1 – D4

FC1-1

3. A1 – B2 – C3 – D4 – E5 – F6 – G7 – H8 –I9
4. A1 – B2 – C3 – D4 – E5 – F6 – G7 – H10 – J11 – K12 –L13

Path Coverage = 100 %

**Test Cases**

| | |
|---|---|
| Identifier | TC1-1 |
| Priority | High |
| Short description | This test case checks if the session is Null |
| Pre-condition(s) | NA |
| Post-conditions(s) | NA |
| Input data | Session[“UserId”]=Null |
| Detailed Steps | 1. Click on Membership tab without logging in. |
| Expected result(s) | Redirect to Home Page |

| | |
|---|---|
| Identifier | TC1-2 |

| | |
|---|---|
| Priority | High |
| Short description | This test case checks if the session is Null |
| Pre-condition(s) | User is logged in |
| Post-conditions(s) | NA |
| Input data | Session[“UserId”]=1 |
| Detailed Steps | 1. Log in the account<br>2. Click on Membership tab. |
| Expected result(s) | Redirect to Home Page |

| | |
|---|---|
| Identifier | TC1-3 |
| Priority | High |
| Short description | This test case checks if the query to update membership runs successfully. |
| Pre-condition(s) | User is logged in |
| Post-conditions(s) | NA |
| Input data | Membership ID=3 (doesn’t exist in database) |
| Detailed Steps | 1. Log in the account<br>2. Click on Membership tab.<br>3. Enter membership ID.<br>4. Click update button. |
| Expected result(s) | Redirect to User Profile Page + Membership updated |

**Loop Boundary Testing**

As there is no loop in the code, so loop boundary testing will not be performed.

# 2. WishList

## 2.1 Code Snippets

Wishlist

```
public ActionResult WishList(string submitButton)
{
    if (Session["userid"] == null)
    {
        return RedirectToAction("Index", "Home");
    }
    if (submitButton != null)
    {
        int userid = int.Parse(Session["userid"].ToString());
        DAL.AddToWishList(userid, int.Parse(submitButton));

        return RedirectToAction("Booking", "UserMain");
    }
    else
    {
        int userid = int.Parse(Session["userid"].ToString());

        List<WishlistBO> wishes = DAL.GetWishlists(userid);
        return View(wishes);
    }
```

Add To Wishlist DAL

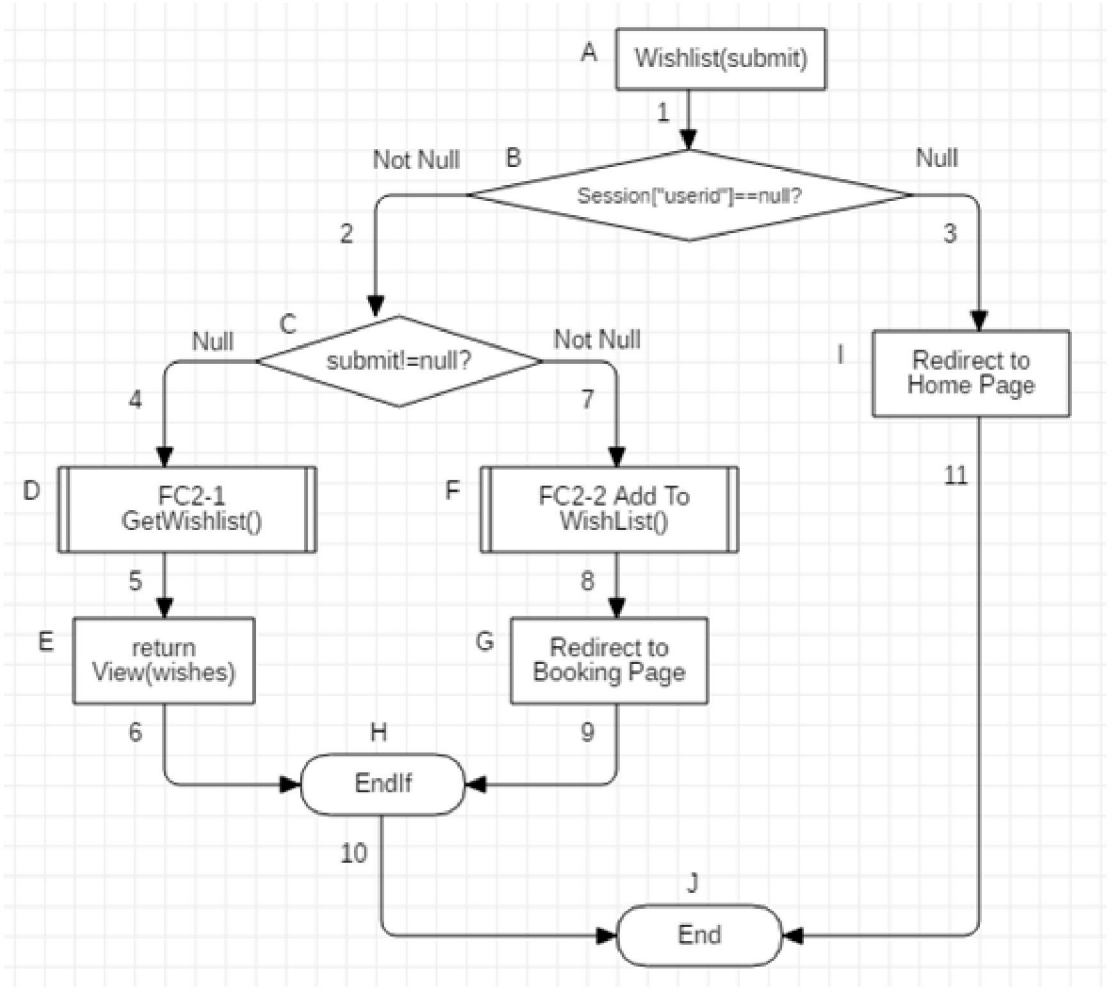
```
public static void AddToWishList(int userid, int tourid)
{
    string connectionString = ConfigurationSettings.AppSettings["ConnectionString"].ToString();
    SqlConnection con = new SqlConnection(connectionString);
    con.Open();
    string query = "INSERT INTO Wishlist VALUES(" + userid + "," +  tourid   +   ")";
    SqlCommand cmd = new SqlCommand(query, con);
    cmd.CommandType = System.Data.CommandType.Text;
    cmd.ExecuteNonQuery();

    con.Close();

}
```

Get Wishlist DAL

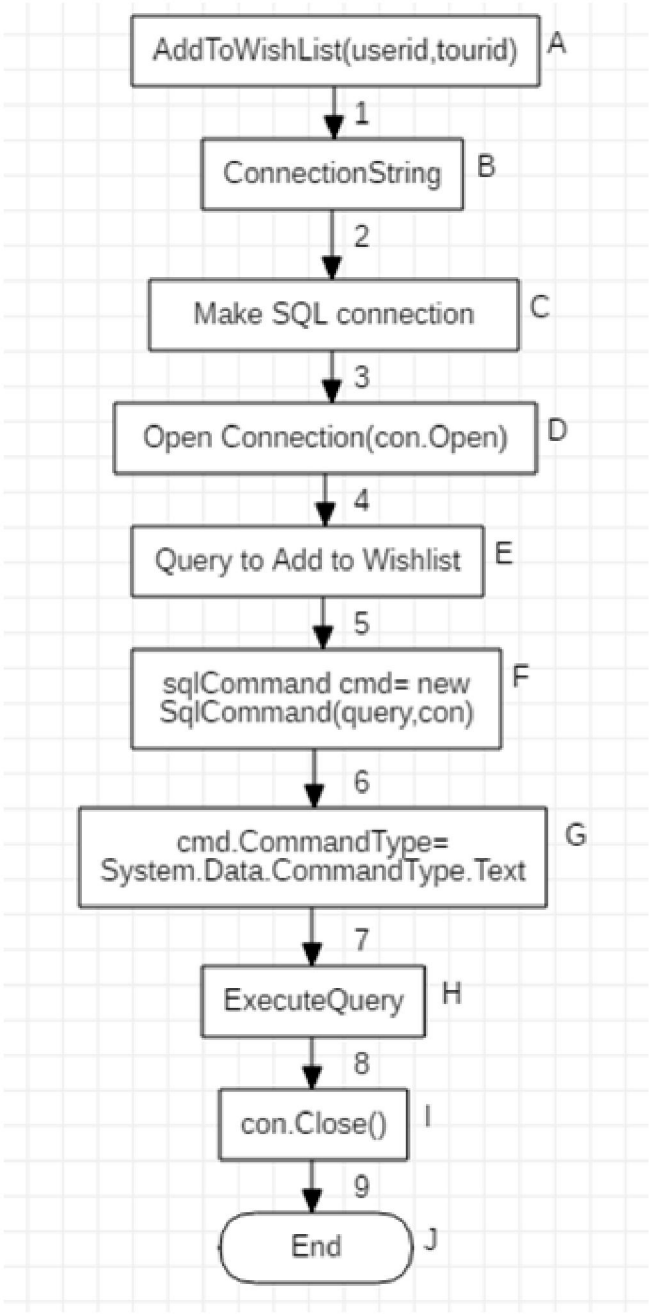
```
public static List<WishlistBO> GetWishlists(int userid)
{
    List<WishlistBO> Wislists = new List<WishlistBO>();

    string connectionString = ConfigurationSettings.AppSettings["ConnectionString"].ToString();
    SqlConnection con = new SqlConnection(connectionString);
    con.Open();
    string query = "SELECT distinct * From Wishlist WHERE userid = " + userid;
    SqlCommand cmd = new SqlCommand(query, con);
    try
    {
        SqlDataReader reader = cmd.ExecuteReader();
        while (reader.Read())
        {
            WishlistBO Wishlist = new WishlistBO();
            Wishlist.tourid = reader.GetInt32(2);

            Wislists.Add(Wishlist);
        }
    }
    catch (Exception ex)
    {
        return Wislists;
    }
    con.Close();
    return Wislists;
}
```

## 2.2 Flow Charts

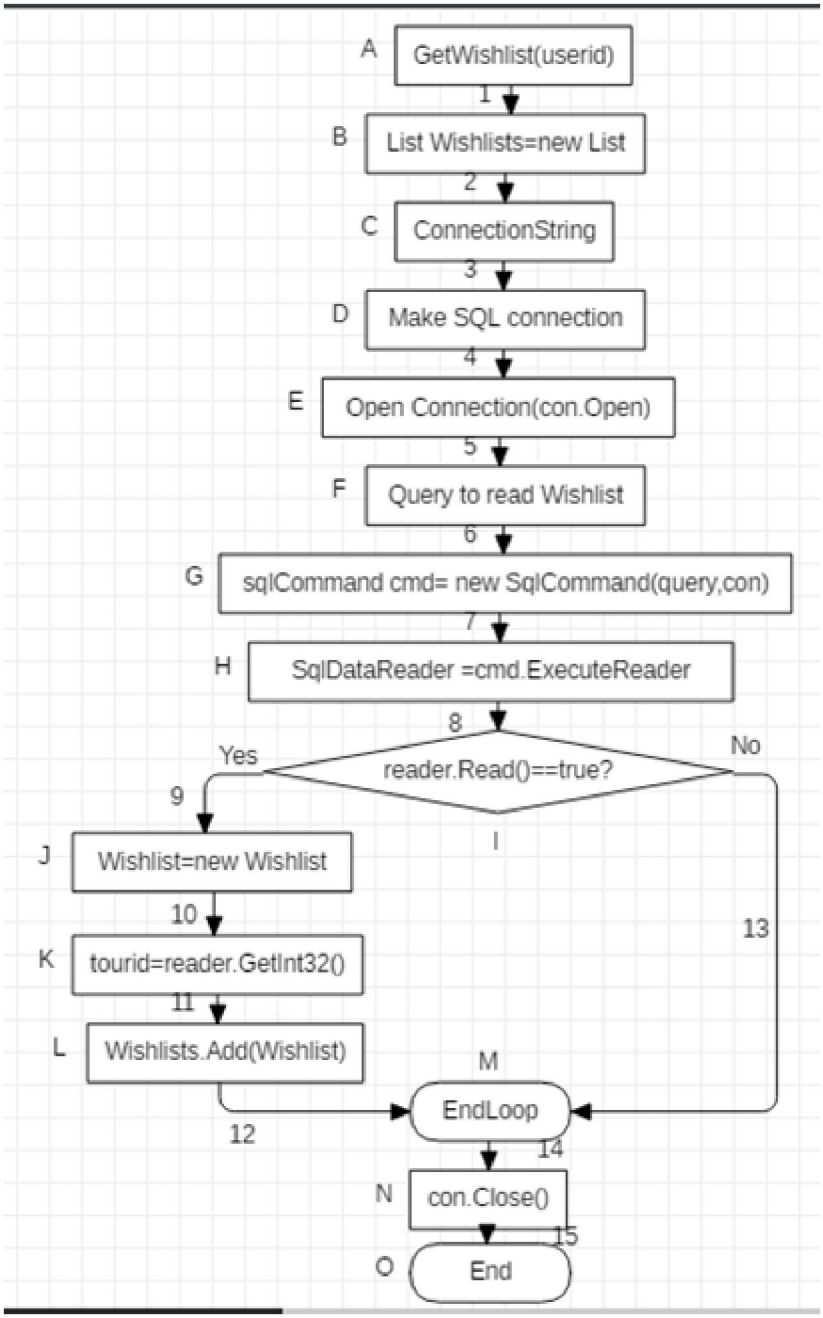
FC2 WishList




FC2-1 AddToWishlist DAL




FC2-2 Get Wishlist DAL




## 2.3 Test Cases and Coverage

**Statement Coverage**

FC2

1. A1 – B3 – I11
2. A1 – B2 – C4 – D5 – FC2-1 – E6 – H10
3. A1 – B2 –C7 –F8 – FC2-2 – G9 – H10

FC1-1

1. A1 – B2 – C3 – D4 – E5 – F6 – G7 – H8 –I9

FC1-2

1. A1 – B2 – C3 – D4 – E5 – F6 – G7 – H8 –I9 –J10 –K11 – L12 – I13 – M14 –N15

Statement Coverage = 32 / 32 = 100 %

**Branch Coverage**

FC2

1. A1 – B3 – I11
2. A1 – B2 – C4 – D5 – FC2-1 – E6 – H10
3. A1 – B2 –C7 –F8 – FC2-2 – G9 – H10

FC1-1

1. A1 – B2 – C3 – D4 – E5 – F6 – G7 – H8 –I9

FC1-2

1. A1 – B2 – C3 – D4 – E5 – F6 – G7 – H8 –I9 –J10 –K11 – L12 – I13 – M14 –N15

Branch Coverage = 6 / 6 = 100 %

**Path Coverage**

FC2

1. A1 – B3 – I11
2. A1 – B2 – C4 – D5 – FC2-1 – E6 – H10
3. A1 – B2 –C7 –F8 – FC2-2 – G9 – H10

FC1-1

1. A1 – B2 – C3 – D4 – E5 – F6 – G7 – H8 –I9

FC1-2

1. A1 – B2 – C3 – D4 – E5 – F6 – G7 – H8 –I9 –J10 –K11 – L12 – I13 – M14 –N15

Path Coverage= 100%

**Test Cases**

| | |
|---|---|
| Identifier | TC2-1 |
| Priority | High |
| Short description | This test case checks if the session is Null |
| Pre-condition(s) | NA |
| Post-conditions(s) | NA |
| Input data | Session[“UserId”]=Null |
| Detailed Steps | 1. Click on Wishlist tab without logging in. |
| Expected result(s) | Redirect to Home Page |

| | |
|---|---|
| Identifier | TC2-2 |
| Priority | High |
| Short description | This test case checks if the session is Null |
| Pre-condition(s) | User is logged in |
| Post-conditions(s) | NA |
| Input data | Session[“UserId”]=1 |
| Detailed Steps | 1. Log in the account<br>2. Click on Wishlist tab. |
| Expected result(s) | Allows user to view wishlist |

| | |
|---|---|
| Identifier | TC2-3 |
| Priority | High |
| Short description | This test case checks if submit =true (item added to wishlist). |
| Pre-condition(s) | User is logged in |
| Post-conditions(s) | NA |
| Input data | Submit=false |
| Detailed Steps | 1. Log in the account<br>2. Click on wishlist tab. |
| Expected result(s) | Display wishlist of the user. |

| | |
|---|---|
| Identifier | TC2-4 |
| Priority | High |

| | |
|---|---|
| Short description | This test case checks if submit =true or false(item added to wishlist). |
| Pre-condition(s) | User is logged in |
| Post-conditions(s) | NA |
| Input data | Submit=true |
| Detailed Steps | 1. Log in the account<br>2. Click on booking tab.<br>3. Click “Add to Wishlist” button next to the desired tour. |
| Expected result(s) | Tour added to wishlist. |

**Loop Boundary Testing(Retrieve Wishlist of User from database)**

| | |
|---|---|
| Identifier | TC2-5 |
| Priority | High |
| Short description | This test case is used to test the loop that retrieves the wishlist of the user from database. |
| Pre-condition(s) | User is logged in. |
| Post-condition(s) | Wishlist will be shown. |
| Input Data | Select tab Wishlist. |
| Detailed Steps | 1.Click on Tab “Wishlist”.<br>2. Wishlist will be shown. |
| Expected Results | Wishlist will be shown. |

# 3. Manage Tours

## 3.1 Code Snippets

### Tour Main Function

```
public ActionResult Trips(string DepartureDate, string ArrivalDate, string Source, string Destination, string Trip)
{
    if (Session["userid"] == null)
    {
        return RedirectToAction("Index", "Home");
    }
    else
    {
        if (DepartureDate != null && ArrivalDate != null && Source != null && Destination != null)
        {
            TourBO Tour = new TourBO();
            Tour.DepartureDate = DepartureDate;
            Tour.ArrivalDate = ArrivalDate;
            Tour.Source = Source;
            Tour.Destination = Destination;

            DAL.AddTour(Tour);
        }
        if (Trip != null)
        {
            DAL.DeleteTour(int.Parse(Trip));
        }

        int id = int.Parse(Session["userid"].ToString());
        List<TourBO> Tours = DAL.GetAllTours();
        return View(Tours);

    }
}
```

### Add Tour DAL

```
public static int AddTour(TourBO Tour)
{
    string connectionString = ConfigurationSettings.AppSettings["ConnectionString"].ToString();
    SqlConnection con = new SqlConnection(connectionString);
    con.Open();
    string query = "INSERT INTO Tour VALUES('" + Tour.DepartureDate + "','" + Tour.ArrivalDate + "','" + Tour.Source + "','" + Tour.Destination + "');SELECT CAST(scope_identity() AS int)";

    SqlCommand cmd = new SqlCommand(query, con);
    cmd.CommandType = System.Data.CommandType.Text;
    int modified = -1;
    try
    {
        modified = (int)cmd.ExecuteScalar();

    }
    catch (Exception ex)
    {
        return modified;
    }
    con.Close();
    return modified;
}
```

### Retrieve all Tours

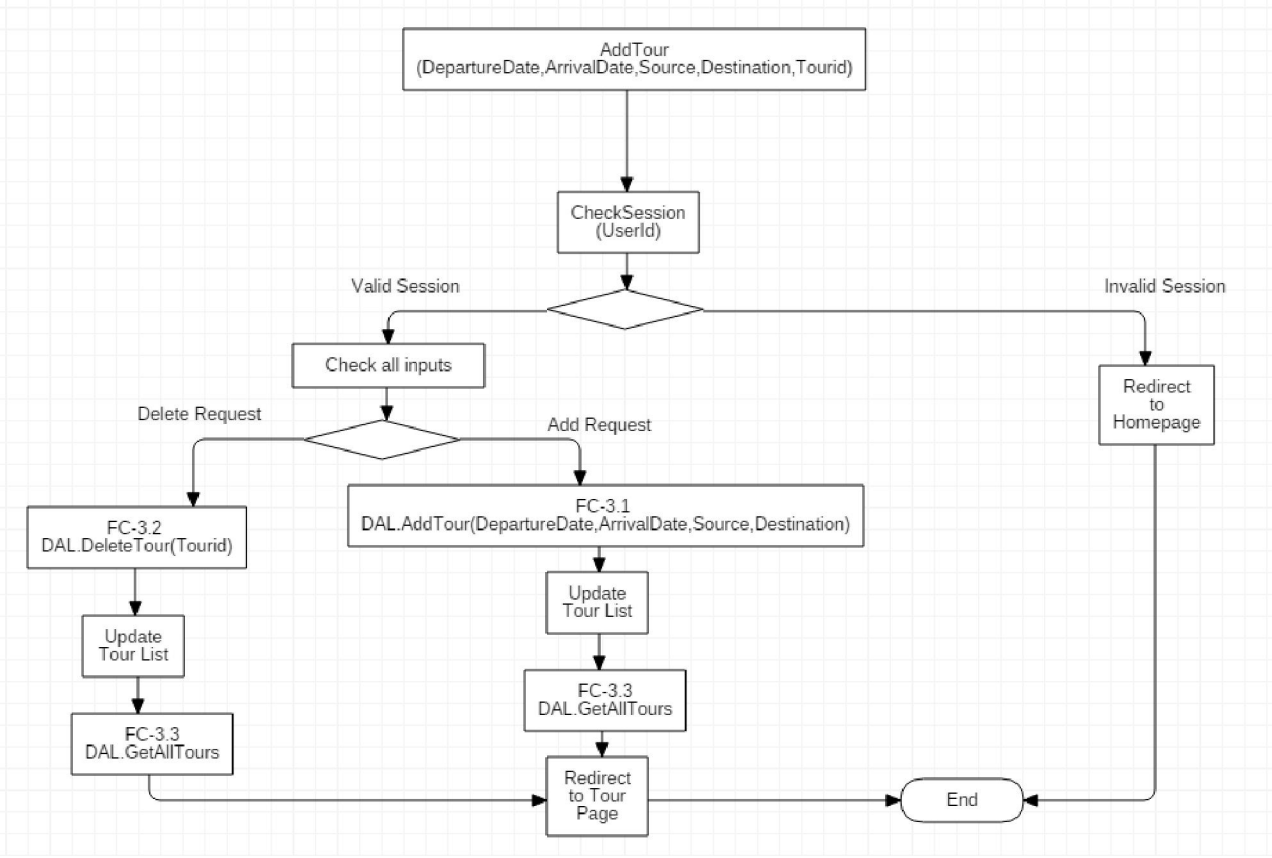
```
public static List<TourBO> GetAllTours()
{
    List<TourBO> Tours = new List<TourBO>();

    string connectionString = ConfigurationSettings.AppSettings["ConnectionString"].ToString();
    SqlConnection con = new SqlConnection(connectionString);
    con.Open();
    string query = "SELECT * From Tour";
    SqlCommand cmd = new SqlCommand(query, con);
    try
    {
        SqlDataReader reader = cmd.ExecuteReader();
        while (reader.Read())
        {
            TourBO Tour = new TourBO();
            Tour.Id = reader.GetInt32(0);
            Tour.DepartureDate = reader.GetDateTime(1).ToString();
            Tour.ArrivalDate = reader.GetDateTime(2).ToString();
            Tour.Source = reader.GetString(3);
            Tour.Destination = reader.GetString(4);

            Tours.Add(Tour);
        }
    }
    catch (Exception ex)
    {
        return Tours;
    }
    con.Close();
    return Tours;
}
```

### Delete Tour

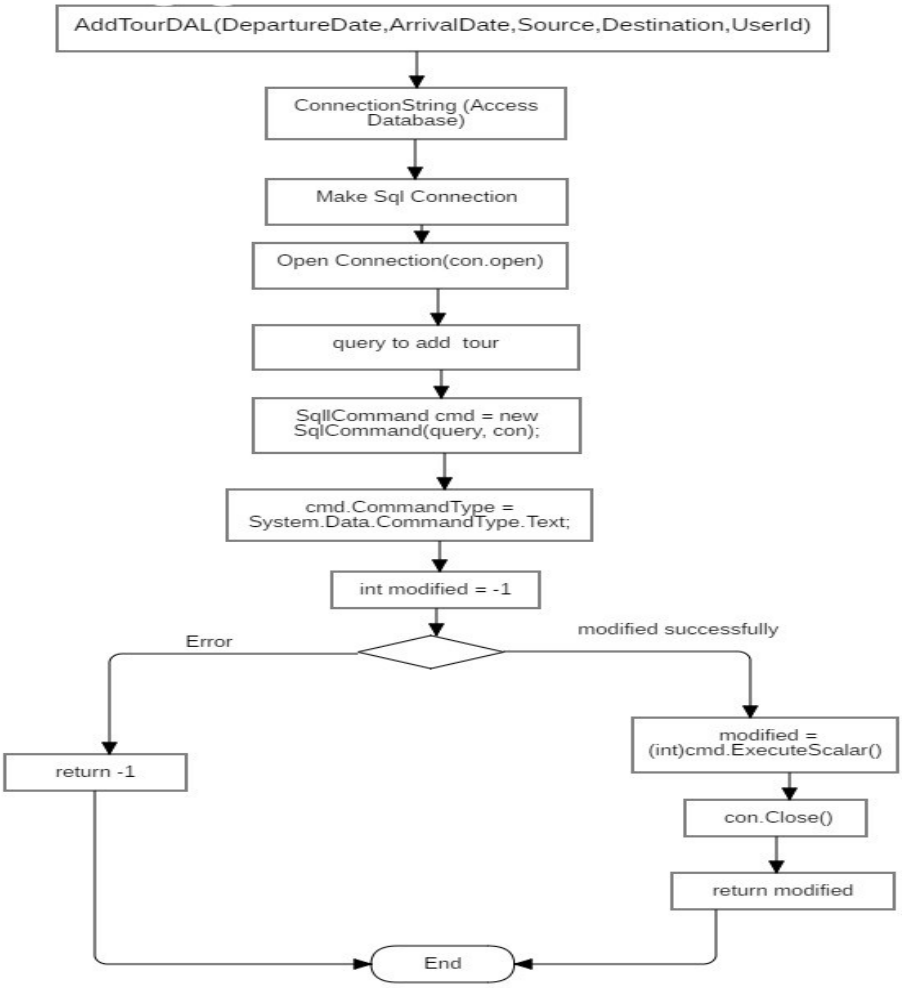
```
public static void DeleteTour(int tourid)
{
    string connectionString = ConfigurationSettings.AppSettings["ConnectionString"].ToString();
    SqlConnection con = new SqlConnection(connectionString);
    con.Open();
    string query = "DELETE FROM Tour WHERE Id = " + tourid;

    SqlCommand cmd = new SqlCommand(query, con);
    cmd.CommandType = System.Data.CommandType.Text;
    cmd.ExecuteNonQuery();

    con.Close();

}
```

## 3.2 Flow Charts

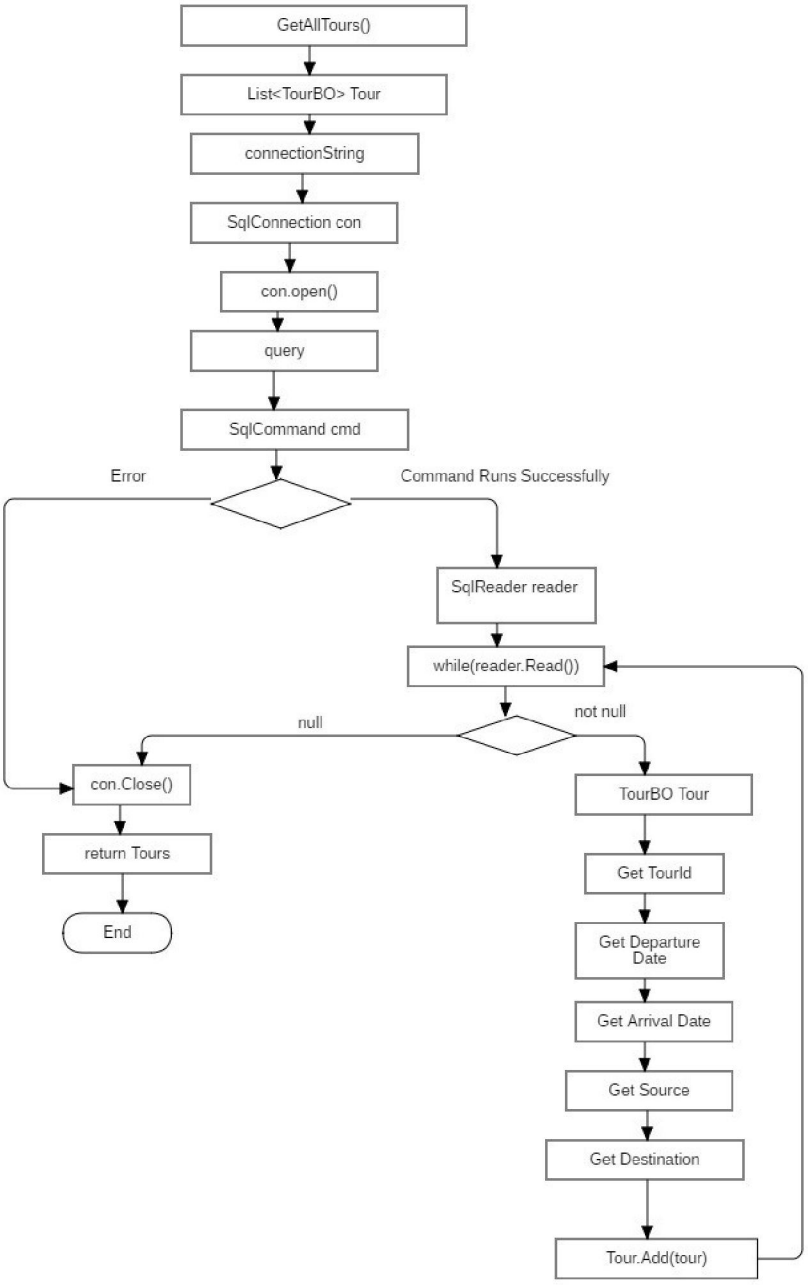
### FC-3 Add Tour Flow Diagram




**FC- 3.1 Add Tour DAL Function Flow Diagram**

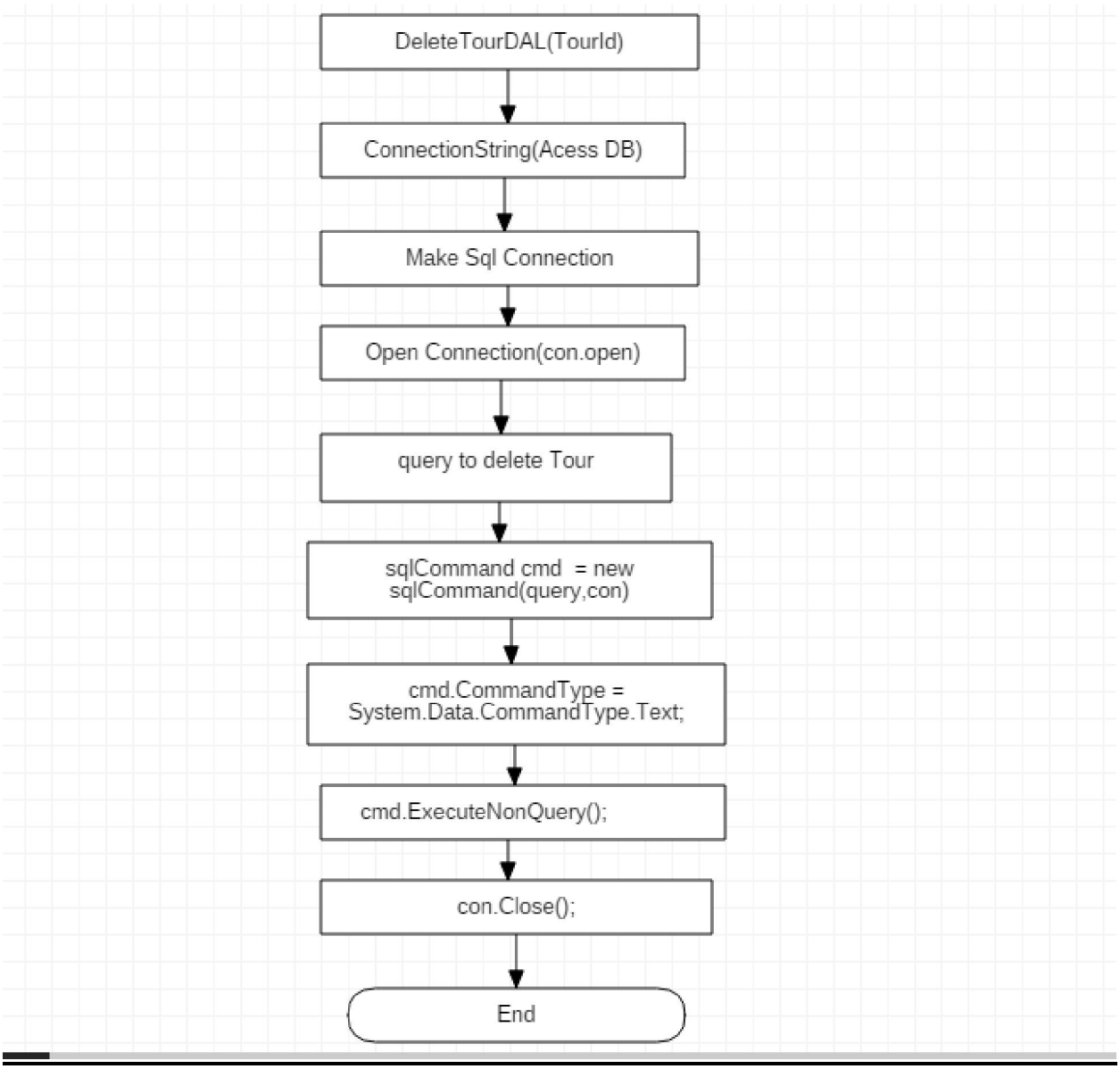

AddTourDAL(DepartureDate,ArrivalDate,Source,Destination,UserId)
ConnectionString (Access Database)
Make Sql Connection
Open Connection(con.open)
query to add tour
SqlCommand cmd = new SqlCommand(query, con);
cmd.CommandType = System.Data.CommandType.Text;
int modified = -1
Error
modified successfully
return -1
modified = (int)cmd.ExecuteScalar()
con.Close()
return modified
End

## FC- 3.2 Retrieve All Tours Flow Diagram




---

## FC-3.3 Delete Tour Flow Diagram


DeleteTourDAL(TourId)
ConnectionString(Acess DB)
Make Sql Connection
Open Connection(con.open)
query to delete Tour
sqlCommand cmd = new sqlCommand(query,con)
cmd.CommandType = System.Data.CommandType.Text;
cmd.ExecuteNonQuery();
con.Close();
End

## 3.3 Test Cases and Coverage

### Test Cases for Manage Tours (Add Tour)

All possible paths are:

1. 1A-2A-3A-4A-12A
2. 1A-2A-3A-6A-7A-1B-2B-3B-4B-7B-8B-9B-10B-9A-10A-11A-1D-2D-3D-4D-5D-6D-7D-8D-9D10D-11D-12D-13D-12A
3. 1A-2A-3A-6A-7A-1B-2B-3B-4B-7B-8B-9B-10B-9A-10A-11A-1D-2D-3D-4D-14D-15D-16D-12A
4. 1A-2A-3A-6A-7A-1B-2B-3B-4B-7B-8B-9B-10B-9A-10A-11A-1D-2D-3D-4D-5D-6D-14D-15D-16D-12A
5. 1A-2A-3A-6A-7A-1B-2B-3B-4B-5B-6B
6. 1A-2A-3A-6A-8A-1C-2C-3C-4C-5C-6C-7C-8C-9C-10C-11A-1D-2D-3D-4D-14D-15D-16D-12A
7. 1A-2A-3A-6A-8A-1C-2C-3C-4C-5C-6C-7C-8C-9C-10C-11A-1D-2D-3D-4D-5D-6D-7D-8D-9D10D-11D-12D-13D-12A
8. 1A-2A-3A-6A-8A-1C-2C-3C-4C-5C-6C-7C-8C-9C-10C-11A--1D-2D-3D-4D-5D-6D-14D-15D-16D-12A

**Statement Testing**

Total No. of statements=58
Statements covered =54
Coverage = Statements covered / Total No. of statements
Coverage=54/58 = 93 %

| | |
|---|---|
| **Identifier** | TC3-1 |
| **Priority** | High |
| **Short description** | This test case is used to test the Tours added by Admin covering all statements. |
| **Pre-condition(s)** | Admin should be logged-in. |
| **Post-condition(s)** | Tour will be added. |
| **Input Data** | TripId=1, UserId=5,Destination=”Kashmir” ,<br>Source =”Lahore”, DepartureDate =”1-01-2019” , ArrivalDate = “20-01-2019”. |

| | |
|---|---|
| **Detailed Steps** | 1. Admin will log-in.<br>2. Click on Tab “Tours”.<br>3. Enter valid TripId.<br>4. Enter valid UserId.<br>5. Enter valid Destination.<br>6. Enter valid Source.<br>7. Enter valid DepartureDate.<br>8. Enter valid ArrivalDate.<br>9. Click on Add Button. |
| **Expected Results** | Tour will be added and displayed on the page. |

**Branch Testing (valid inputs)**
Total No. of branches =10
Branches covered =10

Coverage = branches covered / Total No. of branches
Coverage=10/10 = 100 %

| | |
|---|---|
| **Identifier** | TC3-2 |
| **Priority** | High |
| **Short description** | This test case is used to test the Tour added by the admin covering all branches. |
| **Pre-condition(s)** | Admin should be logged-in. |
| **Post-condition(s)** | Tour will be added. |
| **Input Data** | TripId=1, UserId=5,Destination=”Kashmir” ,Source =”Lahore”, DepartureDate =”1-01-2019”, ArrivalDate = “20-01-2019”. |

| | |
|---|---|
| **Detailed Steps** | 1. Admin will log-in.<br>2. Click on Tab “Tours”.<br>3. Enter valid TripId.<br>4. Enter valid UserId.<br>5. Enter valid Destination.<br>6. Enter valid Source.<br>7. Enter valid DepartureDate.<br>8. Enter valid ArrivalDate.<br>9. Click on Add Button. |
| **Expected Results** | Tour will be added and displayed on the page. |

**Branch Testing (in-valid inputs)**

| | |
|---|---|
| **Identifier** | TC3-3 |
| **Priority** | High |
| **Short description** | This test case is used to test the tour added by the admin covering all branches. |
| **Pre-condition(s)** | Admin is  logged-in. |
| **Post-condition(s)** | Admin Home Page will be displayed. |
| **Input Data** | TripId=-1, UserId=10000000000,Destination=”Naran” ,<br>Source =”Naran”, DepartureDate =”1-02-2019”, ArrivalDate = “10-02-2018”. |
| **Detailed Steps** | 1. Admin will log-in.<br>2. Click on Tab “Tours”.<br>3. Enter invalid TripId.<br>4. Enter invalid UserId.<br>5. Enter invalid Destination.<br>6. Enter invalid Source.<br>7. Enter invalid DepartureDate.<br>8. Enter invalid ArrivalDate.<br>9. Click on Add Button. |

| | |
|---|---|
| **Expected Results** | Error Message will be displayed and Admin will redirect to Admin Home Page. |

**Path Testing (Add Tour – Valid Login)**

All possible paths are:

1. 1A-2A-3A-4A-12A
2. 1A-2A-3A-6A-7A-1B-2B-3B-4B-7B-8B-9B-10B-9A-10A-11A-1D-2D-3D-4D-5D-6D-7D-8D-9D10D-11D-12D-13D-12A
3. 1A-2A-3A-6A-7A-1B-2B-3B-4B-7B-8B-9B-10B-9A-10A-11A-1D-2D-3D-4D-14D-15D-16D-12A
4. 1A-2A-3A-6A-7A-1B-2B-3B-4B-7B-8B-9B-10B-9A-10A-11A-1D-2D-3D-4D-5D-6D-14D-15D-16D-12A
5. 1A-2A-3A-6A-7A-1B-2B-3B-4B-5B-6B
6. 1A-2A-3A-6A-8A-1C-2C-3C-4C-5C-6C-7C-8C-9C-10C-11A-1D-2D-3D-4D-14D-15D-16D-12A
7. 1A-2A-3A-6A-8A-1C-2C-3C-4C-5C-6C-7C-8C-9C-10C-11A-1D-2D-3D-4D-5D-6D-7D-8D-9D10D-11D-12D-13D-12A
8. 1A-2A-3A-6A-8A-1C-2C-3C-4C-5C-6C-7C-8C-9C-10C-11A--1D-2D-3D-4D-5D-6D-14D-15D-16D-12A

| | |
|---|---|
| **Identifier** | TC3-4 |
| **Priority** | High |
| **Short description** | This test case is used to test the tour added by the admin covering all branches. |
| **Pre-condition(s)** | Admin should be logged-in. |
| **Post-condition(s)** | Admin will be added. |
| **Input Data** | TripId=5, UserId=6,Destination=”Naran” ,Source =”Lahore”, DepartureDate =”1-02-2019”, ArrivalDate = “10-02-2019”. |

| | |
|---|---|
| **Detailed Steps** | 1. Admin will log-in.<br>2. Click on Tab “Tours”.<br>3. Enter valid Trip Id.<br>4. Enter valid UserId.<br>5. Enter valid Destination.<br>6. Enter valid Source.<br>7. Enter valid DepartureDate.<br>8. Enter valid Arrival Date.<br>9. Click on Add Button. |
| **Expected Results** | Tour will be added and displayed on the page. |

**Path Testing (Add Tour – Invalid Login (Session Not Implemented))**

| | |
|---|---|
| **Identifier** | TC3-5 |
| **Priority** | High |
| **Short description** | This test case is used to test the tour added by the Admin covering all branches. |
| **Pre-condition(s)** | Admin is not logged-in (Session not implemented) . |
| **Post-condition(s)** | Admin Home Page will be displayed. |
| **Input Data** | Trip Id=5, UserId=6,Destination=”Naran” ,<br>Source =”Lahore”, DepartureDate =”1-02-2019”, Arrival Date = “10-02-2019”. |
| **Detailed Steps** | 1. Click on Tab “Tours”.<br>2. Enter valid Trip Id.<br>3. Enter valid UserId.<br>4. Enter valid Destination.<br>5. Enter valid Source.<br>6. Enter valid DepartureDate.<br>7. Enter valid Arrival Date.<br>8. Click on Add Button. |
| **Expected Results** | Redirect to Admin Home Page. |

**Loop Boundary Testing**
As there is no loop in the code, so loop boundary testing will not be performed.

## Test Cases for Manage Tours (Delete Tour)

All possible paths are:

1. 1A-2A-3A-4A-12A
2. 1A-2A-3A-6A-7A-1B-2B-3B-4B-7B-8B-9B-10B-9A-10A-11A-1D-2D-3D-4D-5D-6D-7D-8D-9D10D-11D-12D-13D-12A
3. 1A-2A-3A-6A-7A-1B-2B-3B-4B-7B-8B-9B-10B-9A-10A-11A-1D-2D-3D-4D-14D-15D-16D-12A
4. 1A-2A-3A-6A-7A-1B-2B-3B-4B-7B-8B-9B-10B-9A-10A-11A-1D-2D-3D-4D-5D-6D-14D-15D-16D-12A
5. 1A-2A-3A-6A-7A-1B-2B-3B-4B-5B-6B
6. 1A-2A-3A-6A-8A-1C-2C-3C-4C-5C-6C-7C-8C-9C-10C-11A-1D-2D-3D-4D-14D-15D-16D-12A
7. 1A-2A-3A-6A-8A-1C-2C-3C-4C-5C-6C-7C-8C-9C-10C-11A-1D-2D-3D-4D-5D-6D-7D-8D-9D10D-11D-12D-13D-12A
8. 1A-2A-3A-6A-8A-1C-2C-3C-4C-5C-6C-7C-8C-9C-10C-11A--1D-2D-3D-4D-5D-6D-14D-15D-16D-12A

**Statement Testing**
Total No. of statements=58
Statements covered =54
Coverage = Statements covered / Total No. of statements
Coverage=54/58 = 93 %

| | |
|---|---|
| **Identifier** | TC3-1 |
| **Priority** | High |
| **Short description** | This test case is used to test the Delete Tour Function covering all statements. |
| **Pre-condition(s)** | Admin should be logged-in. |
| **Post-condition(s)** | Tour will be Deleted. |
| **Input Data** | TripId=1 |

| | |
|---|---|
| **Detailed Steps** | 1. Admin will log-in.<br>2. Click on Tab “Tours”.<br>3. Enter valid TripId.<br>4. Click on Delete Button. |
| **Expected Results** | Tour will be Delete and updated tour list will be displayed on the page. |

**Branch Testing (valid input)**
Total No. of branches =10
Branches covered =10
Coverage = branches covered / Total No. of branches
Coverage=10/10 = 100 %

| | |
|---|---|
| **Identifier** | TC3-2 |
| **Priority** | High |
| **Short description** | This test case is used to test the Delete Tour Function covering all statements. |
| **Pre-condition(s)** | Admin should be logged-in. |
| **Post-condition(s)** | Tour will be Deleted. |
| **Input Data** | TripId=1 |
| **Detailed Steps** | 1. Admin will log-in.<br>2. Click on Tab “Tours”.<br>3. Enter valid TripId.<br>4. Click on Delete Button. |
| **Expected Results** | Tour will be deleted and updated tour list will be displayed on the page. |

**Branch Testing (in-valid inputs)**

| | |
|---|---|
| **Identifier** | TC3-3 |
| **Priority** | High |

| | |
|---|---|
| **Short description** | This test case is used to test the Delete Tour Function covering all statements. |
| **Pre-condition(s)** | Admin is  logged-in. |
| **Post-condition(s)** | Admin Home Page will be displayed. |
| **Input Data** | TripId=-1 |
| **Detailed Steps** | 1. Admin will log-in.<br>2. Click on Tab “Tours”.<br>3. Enter invalid TripId.<br>4. Click on Delete Button. |
| **Expected Results** | Error message will be displayed and Admin will be redirected to Admin Home Page. |

**Path Testing (Add Tour – Valid Login)**

All possible paths are:

1. 1A-2A-3A-4A-12A
2. 1A-2A-3A-6A-7A-1B-2B-3B-4B-7B-8B-9B-10B-9A-10A-11A-1D-2D-3D-4D-5D-6D-7D-8D-9D10D-11D-12D-13D-12A
3. 1A-2A-3A-6A-7A-1B-2B-3B-4B-7B-8B-9B-10B-9A-10A-11A-1D-2D-3D-4D-14D-15D-16D-12A
4. 1A-2A-3A-6A-7A-1B-2B-3B-4B-7B-8B-9B-10B-9A-10A-11A-1D-2D-3D-4D-5D-6D-14D-15D-16D-12A
5. 1A-2A-3A-6A-7A-1B-2B-3B-4B-5B-6B
6. 1A-2A-3A-6A-8A-1C-2C-3C-4C-5C-6C-7C-8C-9C-10C-11A-1D-2D-3D-4D-14D-15D-16D-12A
7. 1A-2A-3A-6A-8A-1C-2C-3C-4C-5C-6C-7C-8C-9C-10C-11A-1D-2D-3D-4D-5D-6D-7D-8D-9D10D-11D-12D-13D-12A
8. 1A-2A-3A-6A-8A-1C-2C-3C-4C-5C-6C-7C-8C-9C-10C-11A--1D-2D-3D-4D-5D-6D-14D-15D-16D-12A

| | |
|---|---|
| **Identifier** | TC3-4 |
| **Priority** | High |

| | |
|---|---|
| **Short description** | This test case is used to test the Delete Tour Function covering all statements. |
| **Pre-condition(s)** | Admin should be logged-in. |
| **Post-condition(s)** | Tour will be Deleted. |
| **Input Data** | TripId=5 |
| **Detailed Steps** | 1. Admin will log-in.<br>2. Click on Tab “Tours”.<br>3. Enter valid TripId.<br>4. Click on Delete Button. |
| **Expected Results** | Tour will be deleted and updated tour list will be displayed on the page. |

**Path Testing (Add Tour – Invalid Login (Session Not Implemented))**

| | |
|---|---|
| **Identifier** | TC3-5 |
| **Priority** | High |
| **Short description** | This test case is used to test the Delete Tour Function covering all statements. |
| **Pre-condition(s)** | Admin is not logged-in (Session not implemented). |
| **Post-condition(s)** | Admin Home Page will be displayed. |
| **Input Data** | TripId=5 |
| **Detailed Steps** | 1. Click on Tab “Tours”.<br>2. Enter valid TripId.<br>3. Click on Delete Button. |
| **Expected Results** | Redirect to Admin Home Page. |

**Loop Boundary Testing**

As there is no loop in the code, so loop boundary testing will not be performed.

# 4. Booking

## 4.1 Code Snippets

### Booking Main Code

```
public ActionResult Booking()
{
    if (Session["userid"] == null)
    {
        return RedirectToAction("Index", "Home");
    }
    List<TourPackage> Package = DAL.GetAllTourPackages();
    return View(Package);
}
public ActionResult SelectPackage(int PackageID)
{
    if (Session["userid"] == null)
    {
        return RedirectToAction("Index", "Home");
    }
    int userid = int.Parse(Session["userid"].ToString());
    int bookingId = DAL.BookTourPackage(userid, PackageID);
    ViewBag.bookingid = bookingId;
    return View();
}
public ActionResult AddFacility(int bookingid, int FacilityId)
{
    DAL.AddFacilityToBooking(bookingid, FacilityId);
    return RedirectToAction("UserProfile", "UserMain");

}
```

### Book Package DAL

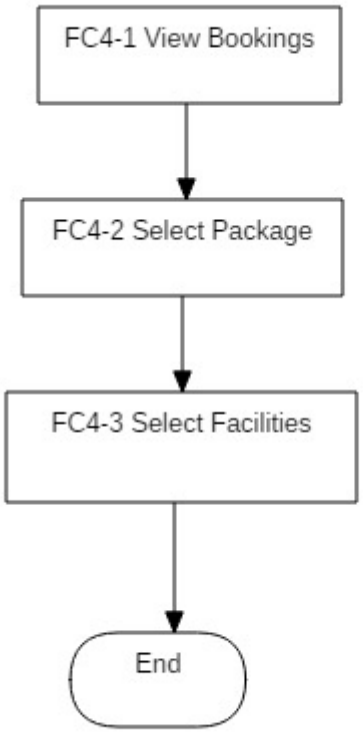
```
public static int BookTourPackage(int userid, int tourPackageId)
{
    string connectionString = ConfigurationSettings.AppSettings["ConnectionString"].ToString();
    SqlConnection con = new SqlConnection(connectionString);
    con.Open();
    string query = "INSERT INTO BOOKING VALUES(" + userid + "," + tourPackageId +
        ", GETDATE(), 0, 'Done');SELECT CAST(scope_identity() AS int)";
    SqlCommand cmd = new SqlCommand(query, con);
    int modified = -1;
    try
    {
        modified = (int)cmd.ExecuteScalar();

    }
    catch (Exception ex)
    {
        return modified;
    }
    con.Close();
    return modified;
}
```

### Book Facility DAL

```
public static int AddFacilityToBooking(int bookingid, int facilityid)
{
    string connectionString = ConfigurationSettings.AppSettings["ConnectionString"].ToString();
    SqlConnection con = new SqlConnection(connectionString);
    con.Open();
    string query = "INSERT INTO UserTourFacility VALUES(" + bookingid + "," + facilityid + ");" +
        "SELECT CAST(scope_identity() AS int)";
    SqlCommand cmd = new SqlCommand(query, con);
    cmd.CommandType = System.Data.CommandType.Text;
    int modified = -1;
    try
    {
        modified = (int)cmd.ExecuteScalar();

    }
    catch (Exception ex)
    {
        return modified;
    }
    con.Close();
    return modified;
}
```

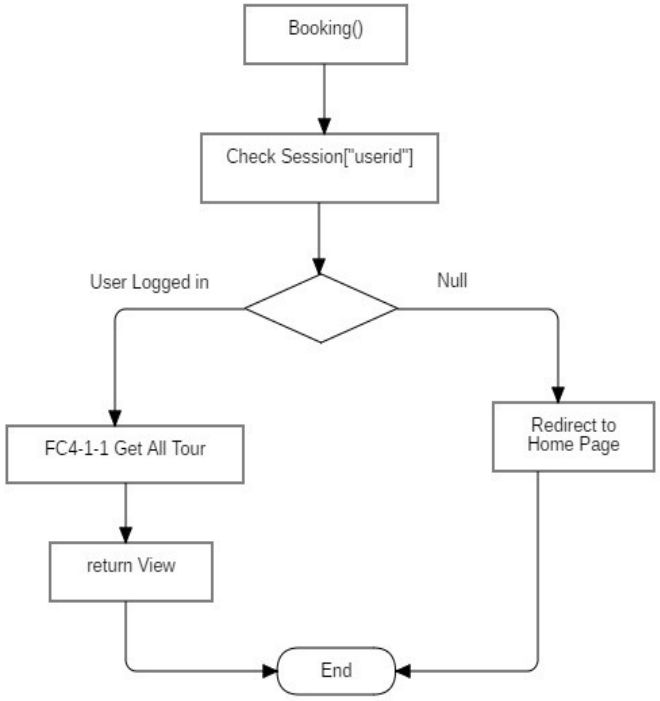
## 4.2 Flow Charts

### FC-4 Book Tours

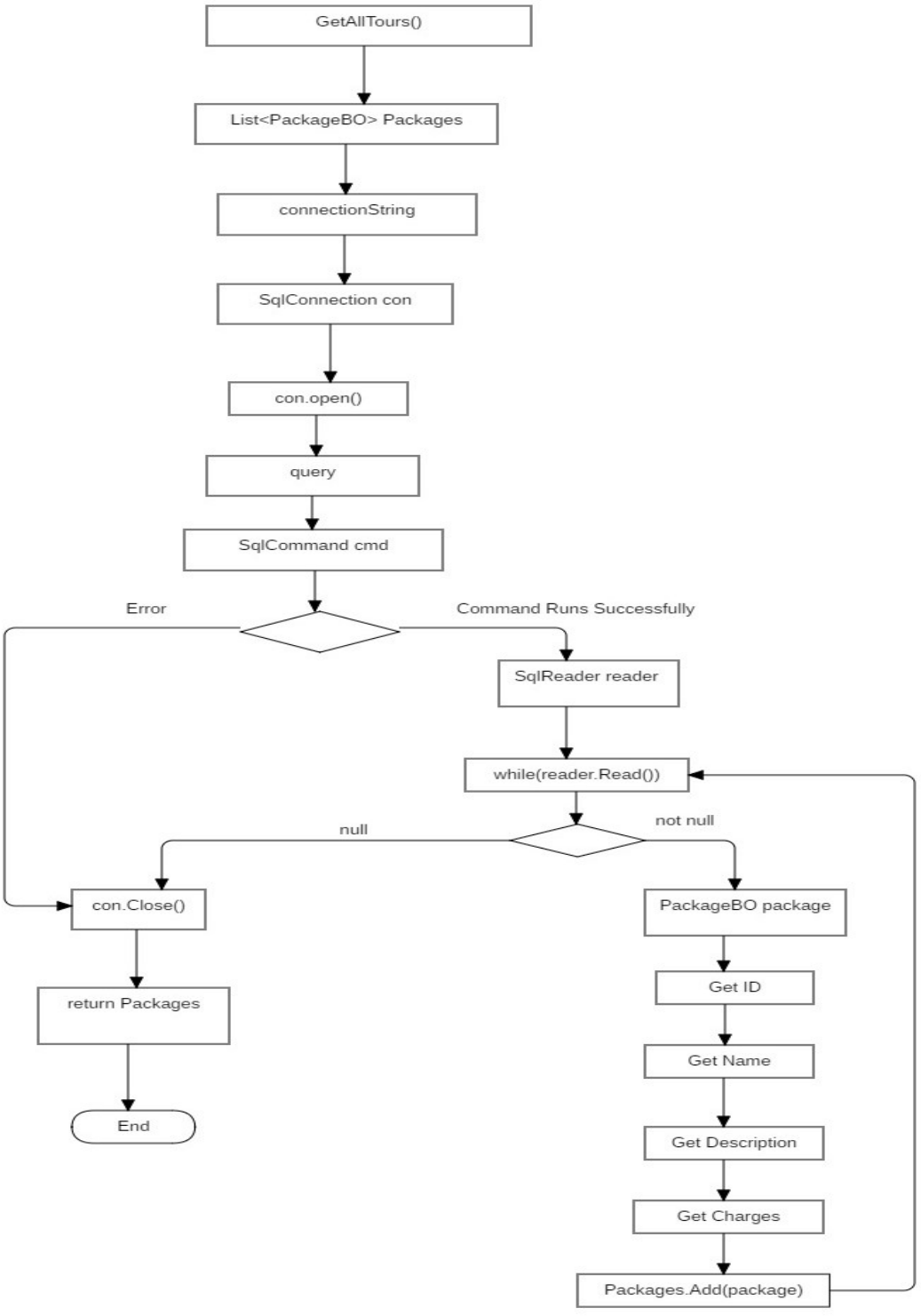



---

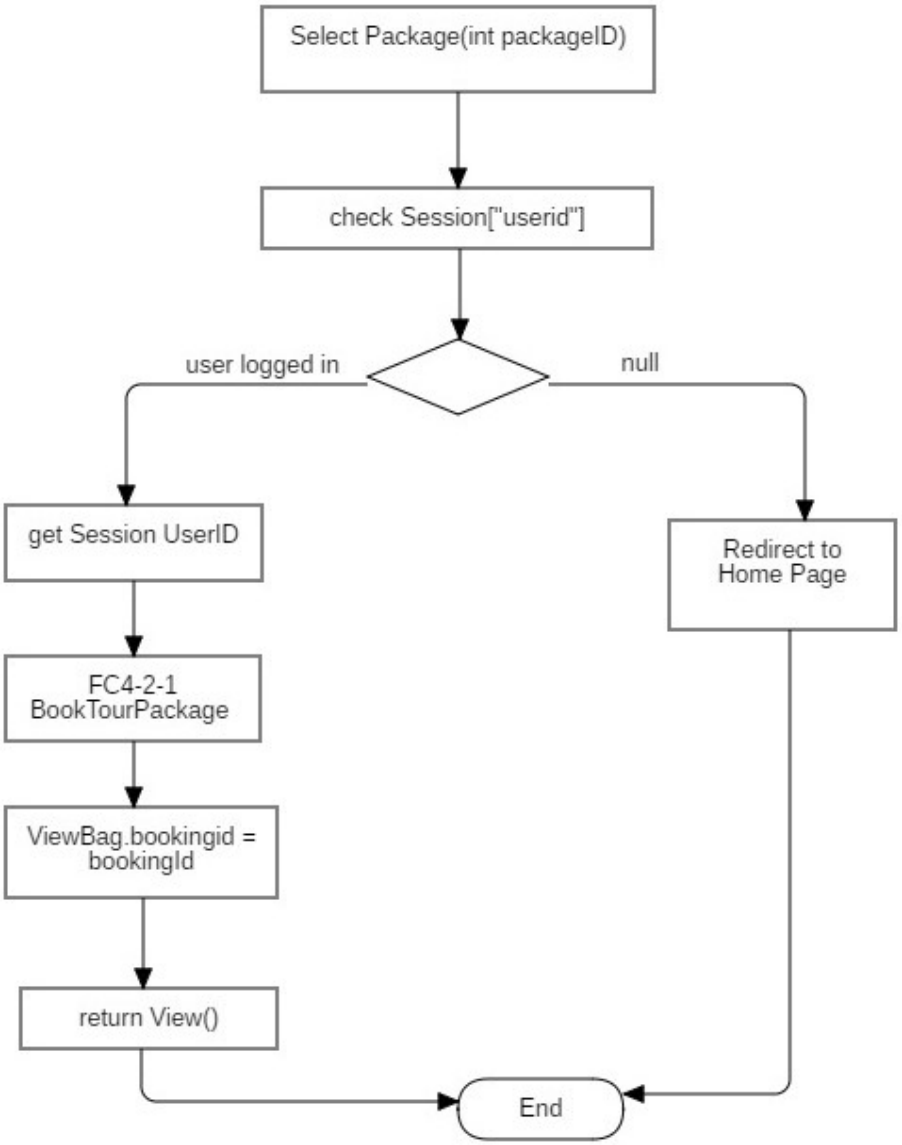
**FC4-1 View Bookings**


Booking()
Check Session["userid"]
User Logged in
Null
FC4-1-1 Get All Tour
Redirect to Home Page
return View
End

## FC4-1-1 GetAllTour

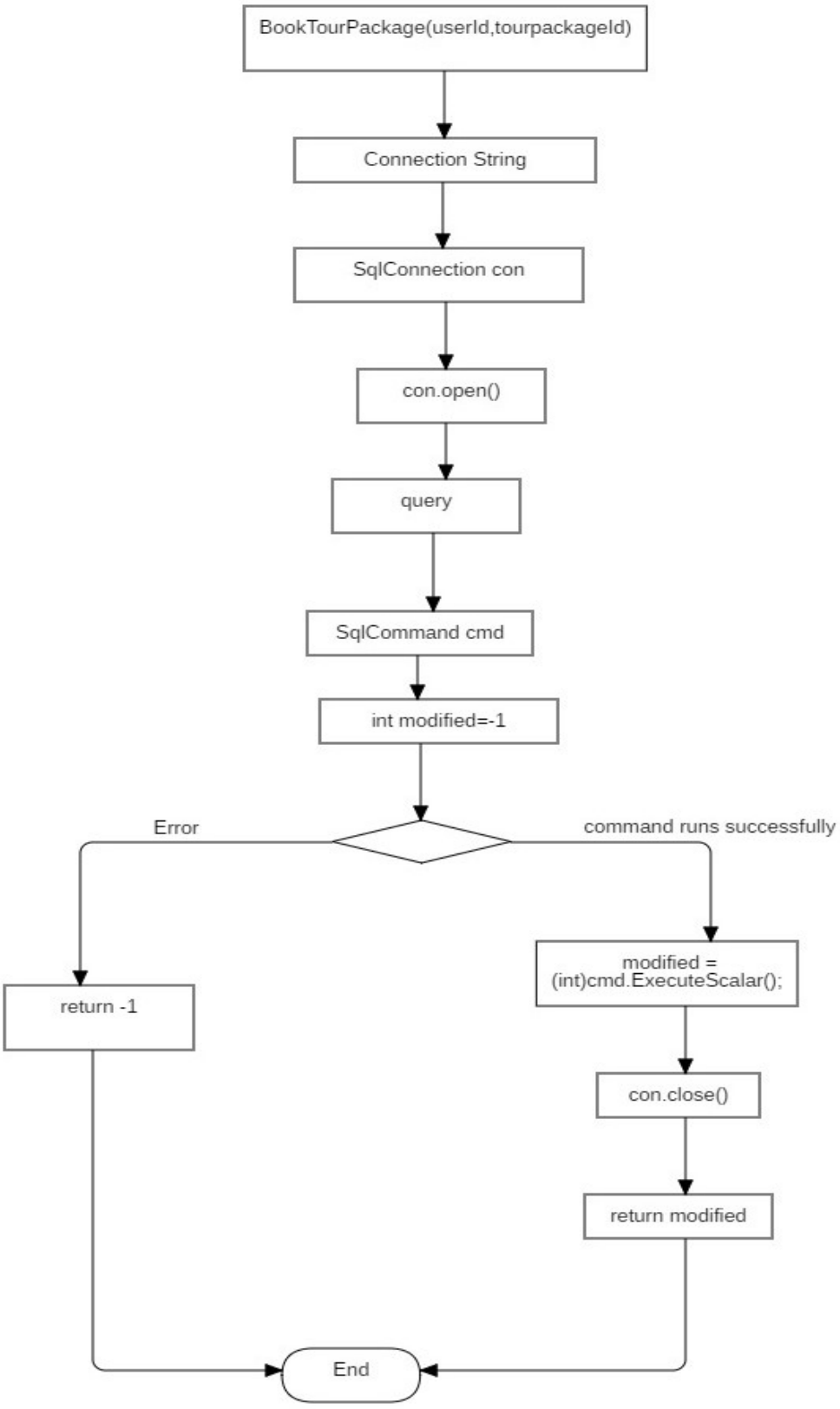


## FC4-2 Select Packages

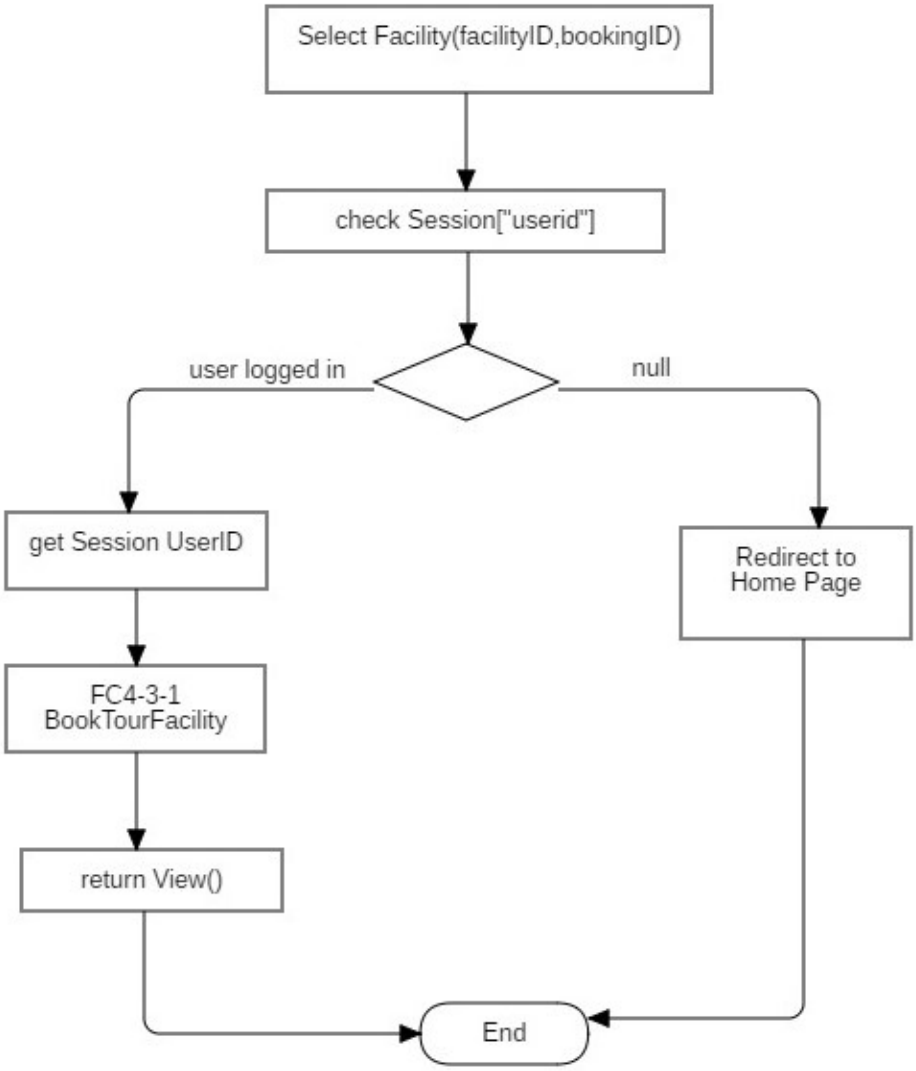



---

## FC4-2-1 BookTourPackages

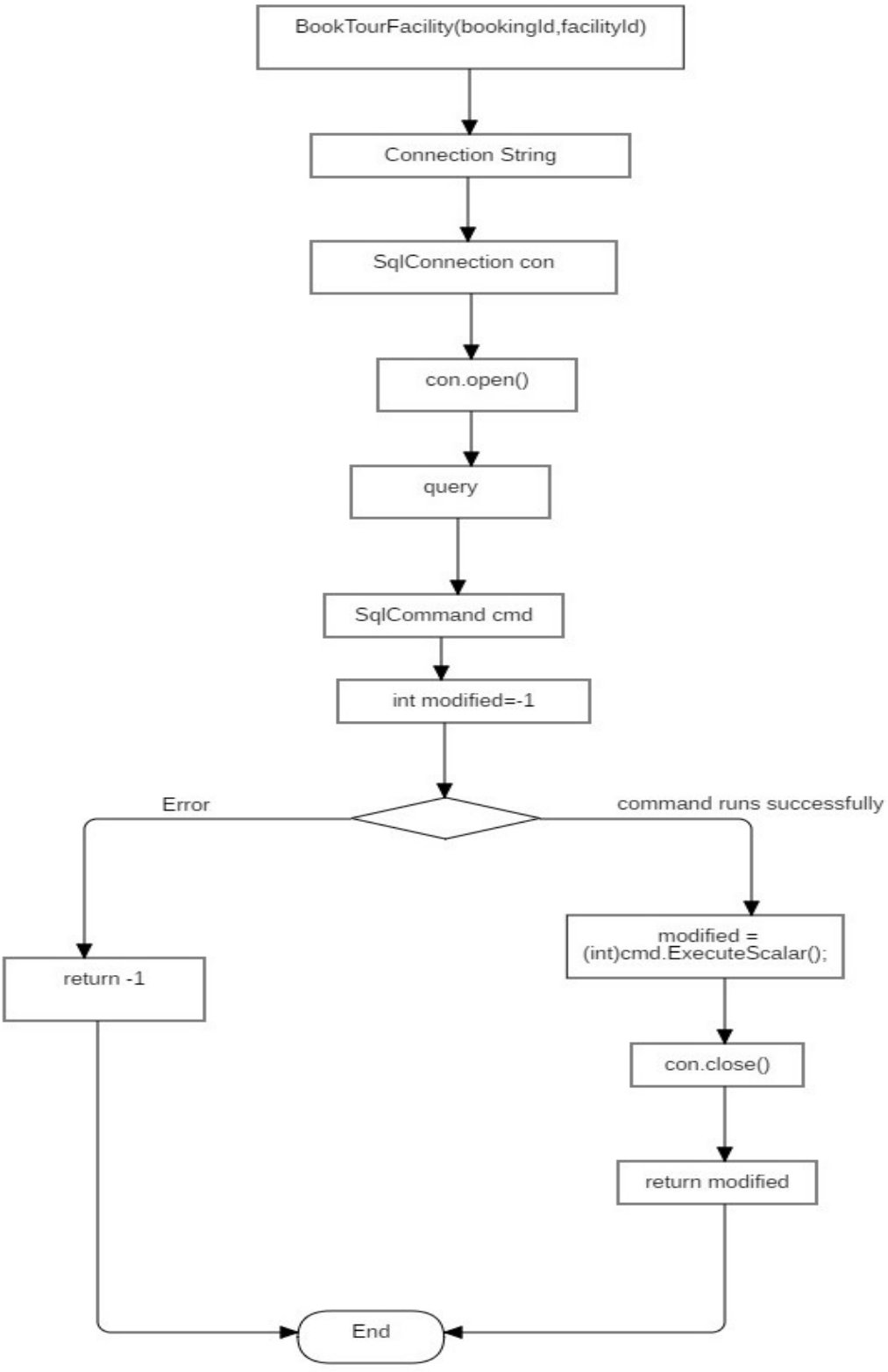

BookTourPackage(userId,tourpackageId)
Connection String
SqlConnection con
con.open()
query
SqlCommand cmd
int modified=-1
Error
command runs successfully
return -1
modified =
(int)cmd.ExecuteScalar();
con.close()
return modified
End

## FC4-3 Select Facilities




---

## FC4-3-1 BookTourFacility


BookTourFacility(bookingId,facilityId)
Connection String
SqlConnection con
con.open()
query
SqlCommand cmd
int modified=-1
Error
command runs successfully
return -1
modified =
(int)cmd.ExecuteScalar();
con.close()
return modified
End

## 4.3 Test Cases and Coverage

**Test Cases**

All possible paths are:

1. 1A-1B-2B-3B-1C-2C-3C-4C-5C-6C-7C-8C-9C-10C-11C12C-13C-14C-15C-4B-2A-1D-2D-3D-4D-1E-2E-3E-4E-5E-6E-7E-8E-9E-10E-11E-5D-3A-1F-2F-3F-4F-1G-2G-3G-4G-5G-6G-7G-8G-9G-10G-5F.
2. 1A-1B-2B-5B.
3. 1A-1B-2B-3B-1C-2C-3C-4C-5C-6C-7C-8C-9C-16C-17C-4B.
4. 1A-2B-3B-1C-2C-3C-4C-5C-6C-7C-8C-9C-10C-11C-12C-13C-14C-15C-4B-2A-1D-2D-5D.
5. 1A-1B-2B-3B-1C-2C-3C-4C-5C-6C-7C-8C-9C-10C-11C12C-13C-14C-15C-4B-1E-2E-3E-4E-5E-6E-7E-12E.
6. 1A-1B-2B-3B-1C-2C-3C-4C-5C-6C-7C-8C-9C-10C-11C12C-13C-14C-15C-4B-2A-1D-2D-3D-4D-1E-2E-3E-4E-5E-6E-7E-8E-9E-10E-11E-5D-3A-1F-2F-3F-5F.
7. 1A-1B-2B-1C-2C-3C-4C-5C-6C-7C-8C-9C-10C-11C12C-13C-14C-15C-4B-2A-1D-2D-3D-4D-1E-2E-3E-4E-5E-6E-7E-8E-9E-10E-11E-5D-3A-1F-2F-3F-4F-1G-2G-3G-4G-5G-6G-7G-11G-12G-5F.

**Statement Testing**

Total No. of statements=55
Statements covered =53

Coverage = Statements covered / Total No. of statements
Coverage=53/55 = **96 %**

| | |
|---|---|
| **Identifier** | TC4-1 |
| **Priority** | High |
| **Short description** | This test case is used to test bookings of tour done by traveller covering all statements. |
| **Pre-condition(s)** | Traveller should be logged-in. |
| **Post-condition(s)** | Selected Tour will be booked |
| **Input Data** | packageID=1, facilityID=1 |
| **Detailed Steps** | 1.Traveller will logged in.<br>2.Click on Tab “Booking”.<br>3.Select Package.<br>4. Click on Add Facility. |

| | |
|---|---|
| | 5. Select Facilities.<br>6. Click on Submit Button. |
| **Expected Results** | Tour will be booked. |

**Branch Testing**
Total No. of branches =7
Branches covered =7

Coverage = branches covered / Total No. of branches
Coverage=7/7 = **100 %**

| | |
|---|---|
| **Identifier** | TC4-2 |
| **Priority** | High |
| **Short description** | This test case is used to test bookings of tour done by traveller covering all branches. |
| **Pre-condition(s)** | Traveller should be logged-in. |
| **Post-condition(s)** | Selected Tour will be booked |
| **Input Data** | packageID=1, facilityID=1 |
| **Detailed Steps** | 1.Traveller will logged in.<br>2.Click on Tab “Booking”.<br>3.Select Package.<br>4. Click on Add Facility.<br>5. Select Facilities.<br>6. Click on Submit Button. |
| **Expected Results** | Tour will be booked. |

| | |
|---|---|
| **Identifier** | TC4-3 |
| **Priority** | High |
| **Short description** | This test case is used to test bookings of tour done by traveller covering all branches. |
| **Pre-condition(s)** | Traveller is not logged in. |
| **Post-condition(s)** | Traveller will be at Home Page. |

| | |
|---|---|
| **Input Data** | packageID=1, facilityID=1 |
| **Detailed Steps** | 1.Click on Tab “Booking”.<br>2.Select Package.<br>3. Click on Add Facility.<br>4. Select Facilities.<br>5. Click on Submit Button. |
| **Expected Results** | Redirects to Home Page. |

**Path Testing**

All possible paths are:

1. 1A-1B-2B-3B-1C-2C-3C-4C-5C-6C-7C-8C-9C-10C-11C12C-13C-14C-15C-4B-2A-1D-2D-3D-4D-1E-2E-3E-4E-5E-6E-7E-8E-9E-10E-11E-5D-3A-1F-2F-3F-4F-1G-2G-3G-4G-5G-6G-7G-8G-9G-10G-5F.
2. 1A-1B-2B-5B.
3. 1A-1B-2B-3B-1C-2C-3C-4C-5C-6C-7C-8C-9C-16C-17C-4B.
4. 1A-2B-3B-1C-2C-3C-4C-5C-6C-7C-8C-9C-10C-11C-12C-13C-14C-15C-4B-2A-1D-2D-5D.
5. 1A-1B-2B-3B-1C-2C-3C-4C-5C-6C-7C-8C-9C-10C-11C12C-13C-14C-15C-4B-1E-2E-3E-4E-5E-6E-7E-12E.
6. 1A-1B-2B-3B-1C-2C-3C-4C-5C-6C-7C-8C-9C-10C-11C12C-13C-14C-15C-4B-2A-1D-2D-3D-4D-1E-2E-3E-4E-5E-6E-7E-8E-9E-10E-11E-5D-3A-1F-2F-3F-5F.
7. 1A-1B-2B-1C-2C-3C-4C-5C-6C-7C-8C-9C-10C-11C12C-13C-14C-15C-4B-2A-1D-2D-3D-4D-1E-2E-3E-4E-5E-6E-7E-8E-9E-10E-11E-5D-3A-1F-2F-3F-4F-1G-2G-3G-4G-5G-6G-7G-11G-12G-5F.

| | |
|---|---|
| **Identifier** | TC4-4 |
| **Priority** | High |
| **Short description** | This test case is used to test bookings of tour done by traveller covering all paths. |
| **Pre-condition(s)** | Traveller should be logged-in. |
| **Post-condition(s)** | Selected Tour will be booked |
| **Input Data** | packageID=1, facilityID=1 |
| **Detailed Steps** | 1.Traveller will logged in.<br>2.Click on Tab “Booking”.<br>3.Select Package. |

| | |
|---|---|
| | 4. Click on Add Facility.<br>5. Select Facilities.<br>6. Click on Submit Button. |
| **Expected Results** | Tour will be booked. |

| | |
|---|---|
| **Identifier** | TC4-5 |
| **Priority** | High |
| **Short description** | This test case is used to test bookings of tour done by traveller covering all paths. |
| **Pre-condition(s)** | Traveller is not logged in. |
| **Post-condition(s)** | Traveller will be at Home Page. |
| **Input Data** | packageID=1, facilityID=1 |
| **Detailed Steps** | 1.Click on Tab “Booking”.<br>2.Select Package.<br>3. Click on Add Facility.<br>4. Select Facilities.<br>5. Click on Submit Button. |
| **Expected Results** | Redirects to Home Page. |

**Loop Boundary Testing(Retrieve All Tour Packages from database)**

| | |
|---|---|
| **Identifier** | TC4-6 |
| **Priority** | High |
| **Short description** | This test case is used to test the loop that retrieve all tour bookings from database. |
| **Pre-condition(s)** | Traveller is logged in. |
| **Post-condition(s)** | All Tour Bookings will be shown. |
| **Input Data** | Select Option Booking. |
| **Detailed Steps** | 1.Click on Tab “Booking”.<br>2. All Bookings will be shown. |
| **Expected Results** | All Bookings will be shown. |

# 5. Feedback

## 5.1 Code Snippets

### Main Code

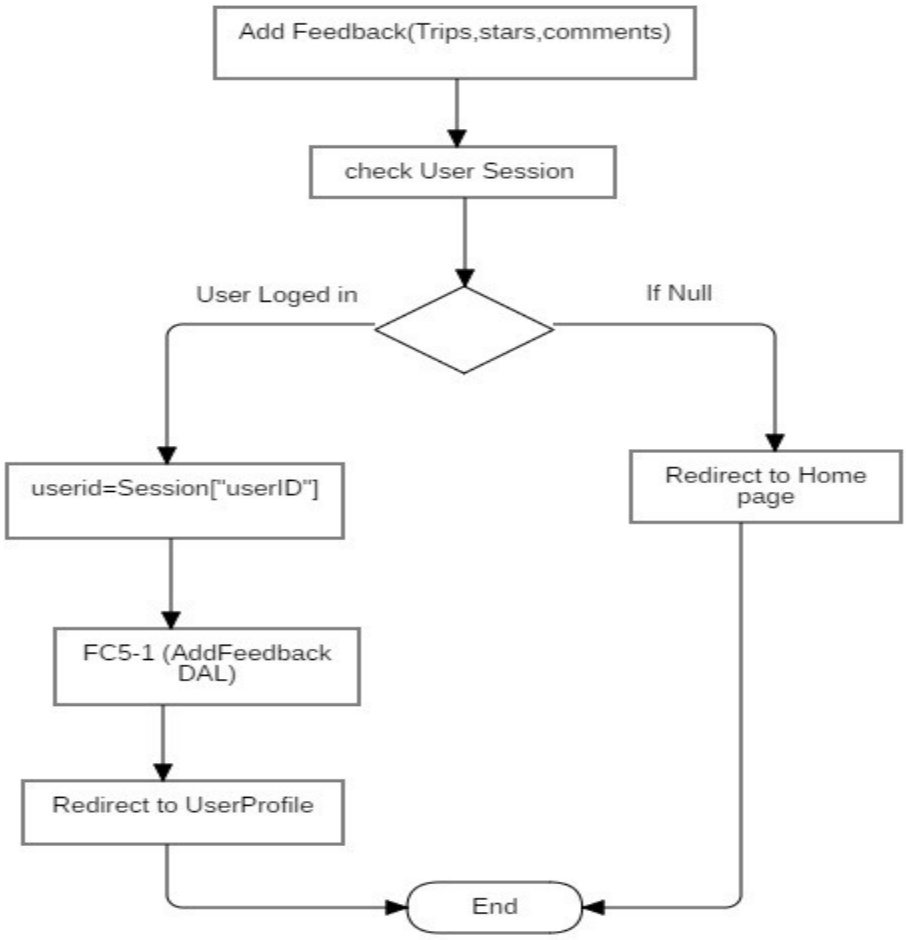
```
}
public ActionResult AddFeedback(int Trip, int stars, string comment)
{
    if (Session["userid"] == null)
    {
        return RedirectToAction("Index", "Home");
    }
    int userid = int.Parse(Session["userid"].ToString());
    DAL.AddFeedback(userid, Trip, stars, comment);
    return RedirectToAction("UserProfile", "UserMain");
}
```

### Add Feedback DAL

```
public static int AddFeedback(int userid, int tripid, int stars, string comment)
{
    string connectionString = ConfigurationSettings.AppSettings["ConnectionString"].ToString();
    SqlConnection con = new SqlConnection(connectionString);
    con.Open();
    string query = "INSERT INTO Reviews VALUES('" + comment + "', " + stars + "," + tripid + ", " + userid + ")";
    SqlCommand cmd = new SqlCommand(query, con);
    cmd.CommandType = System.Data.CommandType.Text;
    int modified = -1;
    try
    {
        modified = (int)cmd.ExecuteScalar();

    }
    catch (Exception ex)
    {
        return modified;
    }
    con.Close();
    return modified;
}
```

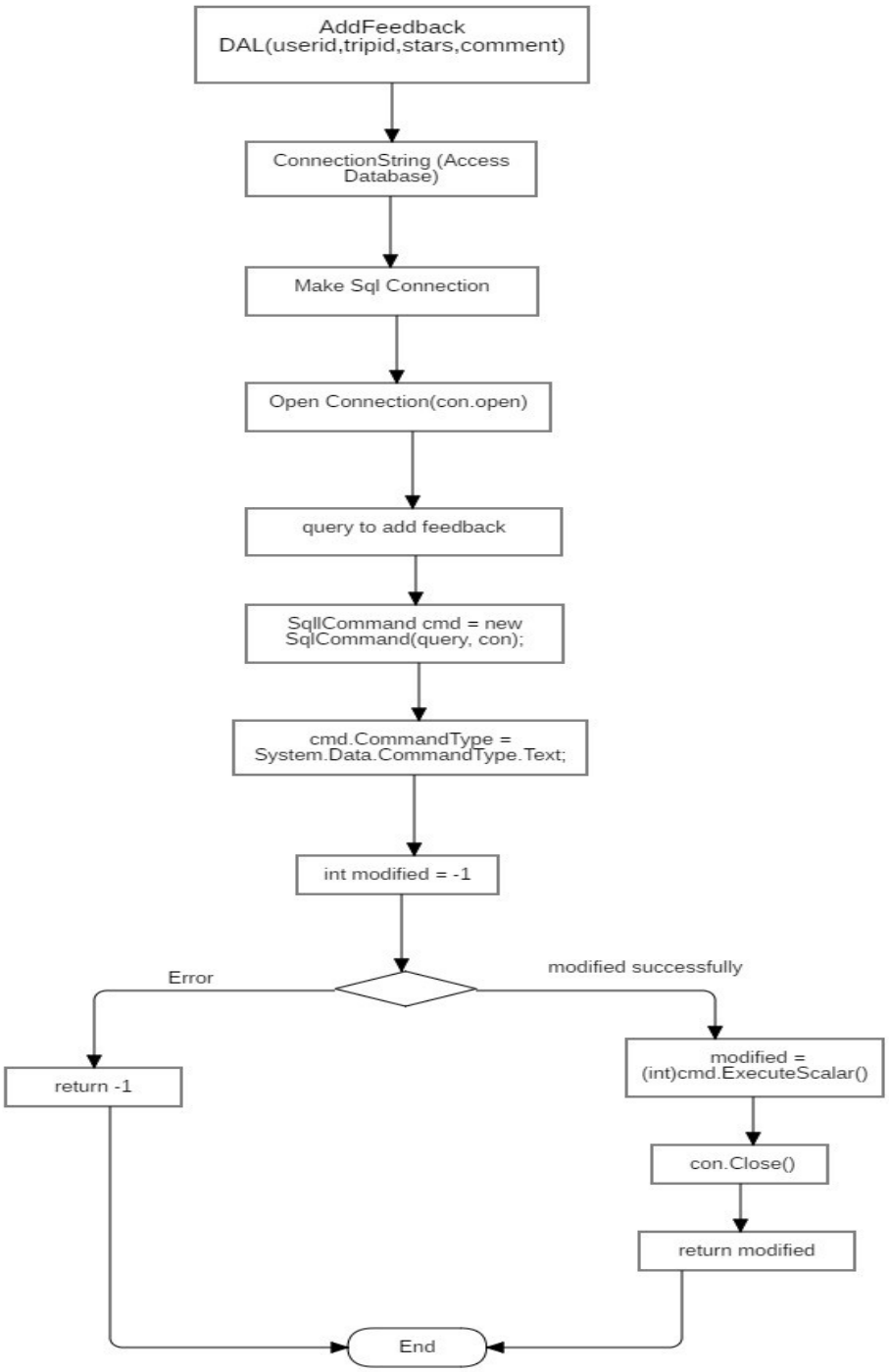
## 5.2 Flow Charts

### FC5 Add Feedback




---

**FC5-1 Add Feedback DAL(Database)**


AddFeedback
DAL(userid,tripid,stars,comment)
ConnectionString (Access Database)
Make Sql Connection
Open Connection(con.open)
query to add feedback
SqlCommand cmd = new SqlCommand(query, con);
cmd.CommandType = System.Data.CommandType.Text;
int modified = -1
Error
modified successfully
return -1
modified = (int)cmd.ExecuteScalar()
con.Close()
return modified
End

## 5.3 Test Cases and Coverage

**Test Cases**

All possible paths are:

1) 1A-2A-3A-4A-1B-2B-3B-4B-5B-6B-7B-8B-9B-10B-11B-4A-5A.
2) 1A-2A-3A-6A.
3) 1A-2A-3A-4A-1B-2B-3B-4B-5B-6B-7B-8B-12B-4A-5A

**Statement Testing**

Total No. of statements=17
Statements covered =17

Coverage = Statements covered / Total No. of statements
Coverage=53/55 = **100 %**

| | |
|---|---|
| **Identifier** | TC5-1 |
| **Priority** | High |
| **Short description** | This test case is used to test the feedback added by the traveller covering all statements. |
| **Pre-condition(s)** | Traveller should be logged-in. |
| **Post-condition(s)** | Feedback will be added. |
| **Input Data** | TripID=1,Rating=4,Comment=Good |
| **Detailed Steps** | 1.Traveller will logged in.<br>2. Click on Tab “Feedback”.<br>3. Select Trip.<br>4. Enter Rating.<br>5. Enter Comment.<br>6. Click on Submit Button. |
| **Expected Results** | Feedback will be added and displayed on the page. |

**Branch Testing**

Total No. of branches =3
Branches covered =3

Coverage = branches covered / Total No. of branches
Coverage=3/3 = **100 %**

| | |
|---|---|
| **Identifier** | TC5-2 |
| **Priority** | High |
| **Short description** | This test case is used to test the feedback added by the traveller covering all branches. |
| **Pre-condition(s)** | Traveller should be logged-in. |
| **Post-condition(s)** | Feedback will be added. |
| **Input Data** | TripID=1,Rating=4,Comment=Good |
| **Detailed Steps** | 1.Traveller will logged in.<br>2.Click on Tab “Feedback”.<br>3.Select Trip.<br>4. Enter Rating.<br>5. Enter Comment.<br>6. Click on Submit Button. |
| **Expected Results** | Feedback will be added and displayed on the page. |

| | |
|---|---|
| **Identifier** | TC5-3 |
| **Priority** | High |
| **Short description** | This test case is used to test the feedback added by the traveller covering all branches. |
| **Pre-condition(s)** | Traveller is not logged-in. |
| **Post-condition(s)** | Home Page will be displayed. |
| **Input Data** | TripID=1,Rating=4,Comment=Good |
| **Detailed Steps** | 1.Click on Tab “Feedback”.<br>2.Select Trip.<br>3. Enter Rating.<br>4. Enter Comment.<br>4. Click on Submit Button. |
| **Expected Results** | Redirect to Home Page. |

**Path Testing**

All possible paths are:

1) 1A-2A-3A-4A-1B-2B-3B-4B-5B-6B-7B-8B-9B-10B-11B-4A-5A.
2) 1A-2A-3A-6A.
3) 1A-2A-3A-4A-1B-2B-3B-4B-5B-6B-7B-8B-12B-4A-5A

| | |
|---|---|
| **Identifier** | TC5-4 |
| **Priority** | High |
| **Short description** | This test case is used to test the feedback added by the traveller covering all branches. |
| **Pre-condition(s)** | Traveller should be logged-in. |
| **Post-condition(s)** | Feedback will be added. |
| **Input Data** | TripID=1,Rating=4,Comment=Good |
| **Detailed Steps** | 1.Traveller will logged in.<br>2.Click on Tab “Feedback”.<br>3.Select Trip.<br>4. Enter Rating.<br>5. Enter Comment.<br>6. Click on Submit Button. |
| **Expected Results** | Feedback will be added and displayed on the page. |

| | |
|---|---|
| **Identifier** | TC5-5 |
| **Priority** | High |
| **Short description** | This test case is used to test the feedback added by the traveller covering all branches. |
| **Pre-condition(s)** | Traveller is not logged-in. |
| **Post-condition(s)** | Home Page will be displayed. |
| **Input Data** | TripID=1,Rating=4,Comment=Good |
| **Detailed Steps** | 1.Click on Tab “Feedback”.<br>2.Select Trip.<br>3. Enter Rating.<br>4. Enter Comment.<br>4. Click on Submit Button. |
| **Expected Results** | Redirect to Home Page. |

### Loop Boundary Testing

As there is no loop in the code, so loop boundary testing will not be performed.

# 6. Manage Travellers

## 6.1 Code Snippets

### Main function Of Travellers

```
public ActionResult Travellers(string UserName, string FirstName, string LastName, string Email, string ContactNo, string UserType, string Password, string MembershipId, string User )
{
    if (Session["userid"] == null)
    {
        return RedirectToAction("Index", "Home");
    }
    else
    {
        if (UserName != null && FirstName != null && LastName != null && Email != null && ContactNo != null && Password != null)
        {
            UserBO user = new UserBO();
            user.Username = UserName;
            user.Fname = FirstName;
            user.Lname = LastName;
            user.email = Email;
            user.ContactNo = ContactNo;
            user.type = "User";
            user.setPassword(Password);
            user.MembershipId = 3;

            UserDAL dal = new UserDAL();
            dal.RegisterUser(user);


        }
        if (User != null)
        {
            DAL.DeleteUser(int.Parse(User));
        }

        int id = int.Parse(Session["userid"].ToString());
        List<UserBO> Users = DAL.GetAllUsers();
        return View(Users);

    }
}
}
```

---

### Add Traveller DAL

```
public bool RegisterUser(UserBO user)
{
    SqlConnection con = new SqlConnection(connectionString);
    SqlCommand command = new SqlCommand("insert into [User]([UserName],[FirstName],[LastName],[UserType],[Password],[Email],[ContactNo],[MembershipId]) values('" + user.Username + "','" + user.Fname
        + "','" + user.Lname + "','" + user.type + "','"
        + user.getPassword() + "','"
        +user.email+"','"+user.ContactNo+"',1)", con);
    con.Open();
    try
    {
        int i = command.ExecuteNonQuery();
        if (i > 0)
        {
            Console.Write("User registered..\n");

            return true;
        }
        else
        {
            Console.Write("User not  registered...\n");

            return false;

        }
    }


    catch (SqlException ex)
    {
        Console.Write("User not  registered...\n");
        return false;
    }
    finally
    {
        con.Close();
    }
}
```

## Retrieve Users List

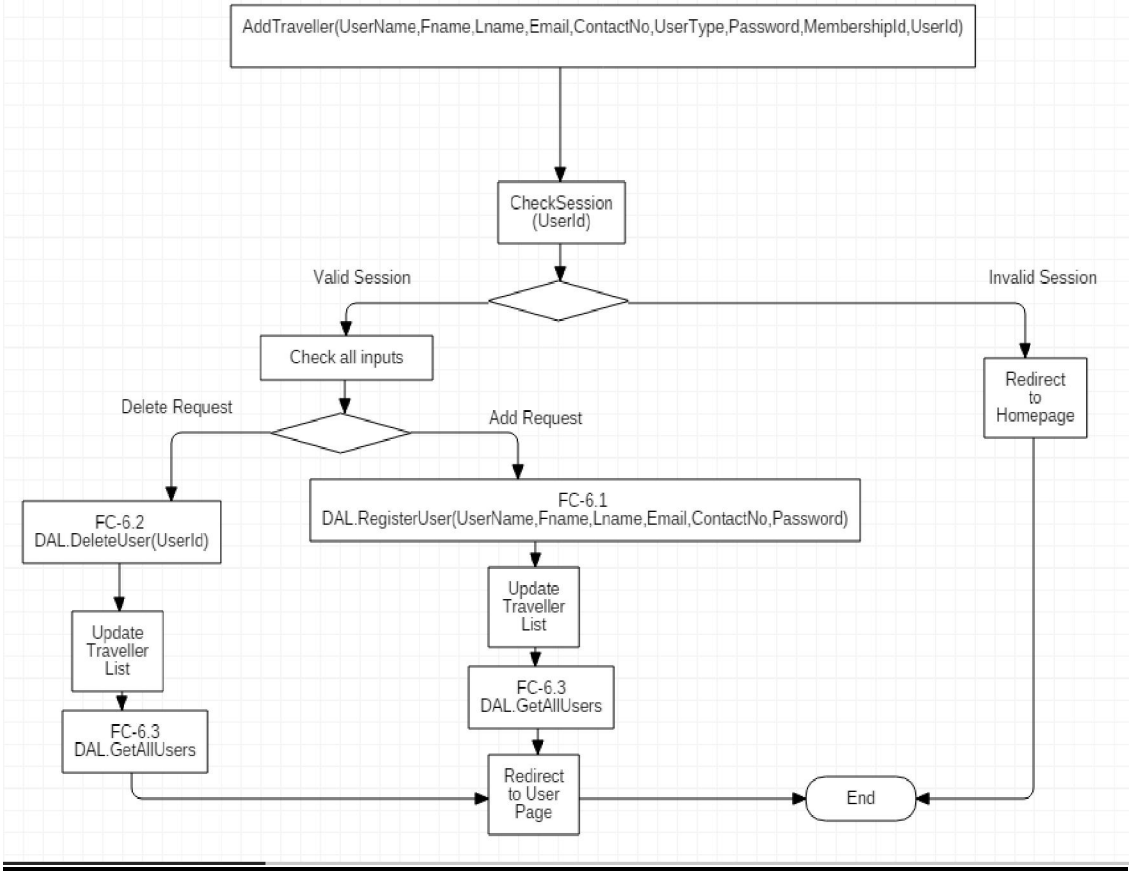
```
public static List<UserBO> GetAllUsers()
{
    List<UserBO> Users = new List<UserBO>();

    string connectionString = ConfigurationSettings.AppSettings["ConnectionString"].ToString();
    SqlConnection con = new SqlConnection(connectionString);
    con.Open();
    string query = "SELECT * From [User] WHERE UserType = 'User'";
    SqlCommand cmd = new SqlCommand(query, con);
    try
    {
        SqlDataReader reader = cmd.ExecuteReader();
        while (reader.Read())
        {
            UserBO User = new UserBO();
            User.Uid = reader.GetInt32(0);
            User.Username = reader.GetString(1);
            User.Fname = reader.GetString(2);
            User.Lname = reader.GetString(3);
            User.email = reader.GetString(4);
            User.ContactNo = reader.GetString(5);
            User.type = reader.GetString(6);
            User.setPassword(reader.GetString(7));
            User.MembershipId = reader.GetInt32(8);

            Users.Add(User);



        }
    }
    catch (Exception ex)
    {
        return Users;
    }
    con.Close();
    return Users;
}
```

## Delete User

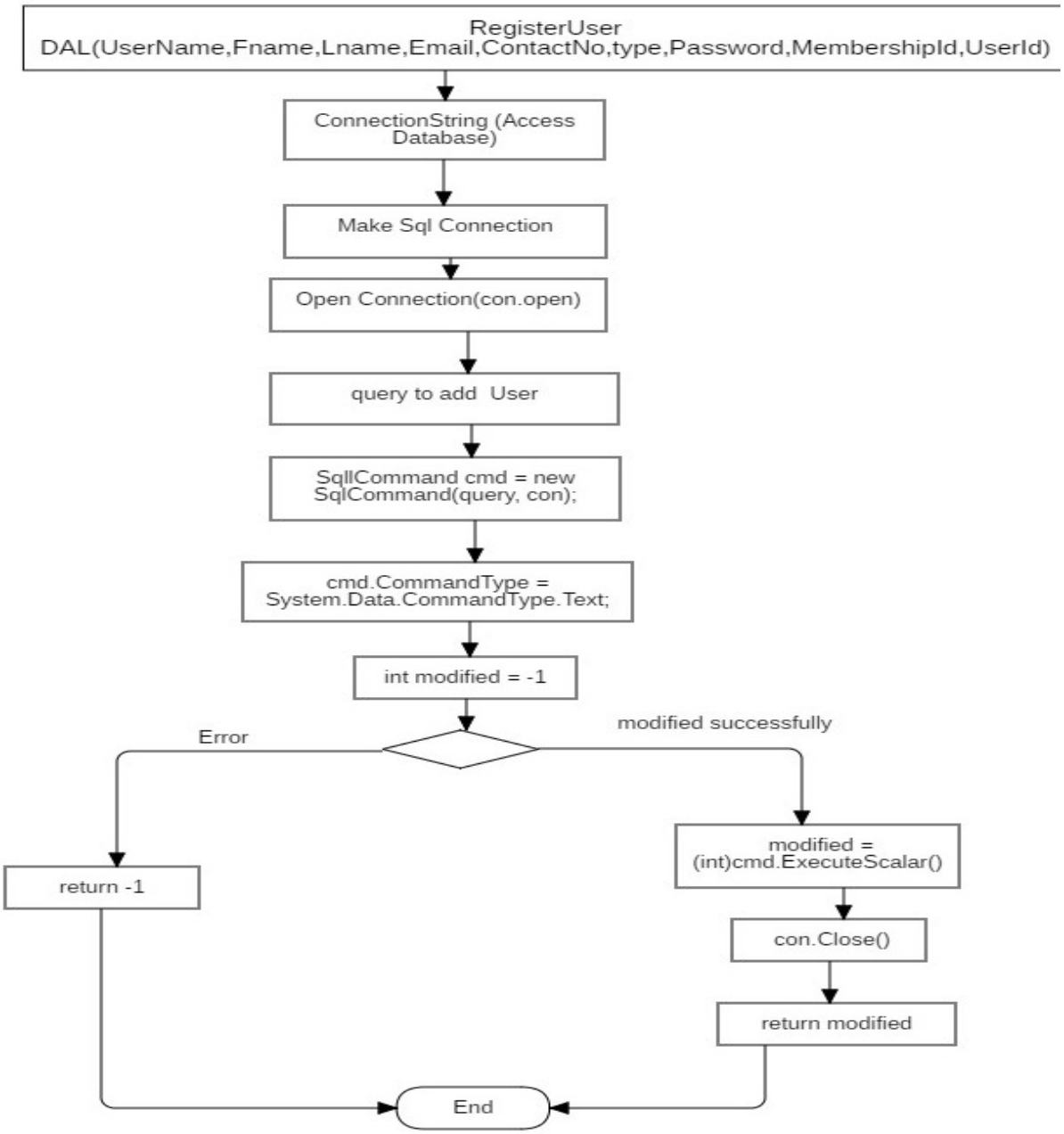
```
public static void DeleteUser(int userid)
{
    string connectionString = ConfigurationSettings.AppSettings["ConnectionString"].ToString();
    SqlConnection con = new SqlConnection(connectionString);
    con.Open();
    string query = "DELETE FROM [User] WHERE Id = " + userid;
    SqlCommand cmd = new SqlCommand(query, con);
    cmd.CommandType = System.Data.CommandType.Text;
    cmd.ExecuteNonQuery();
    con.Close();

}
public static void AddToWishList(int userid, int tourid)
```

# 6.2 Flow Charts

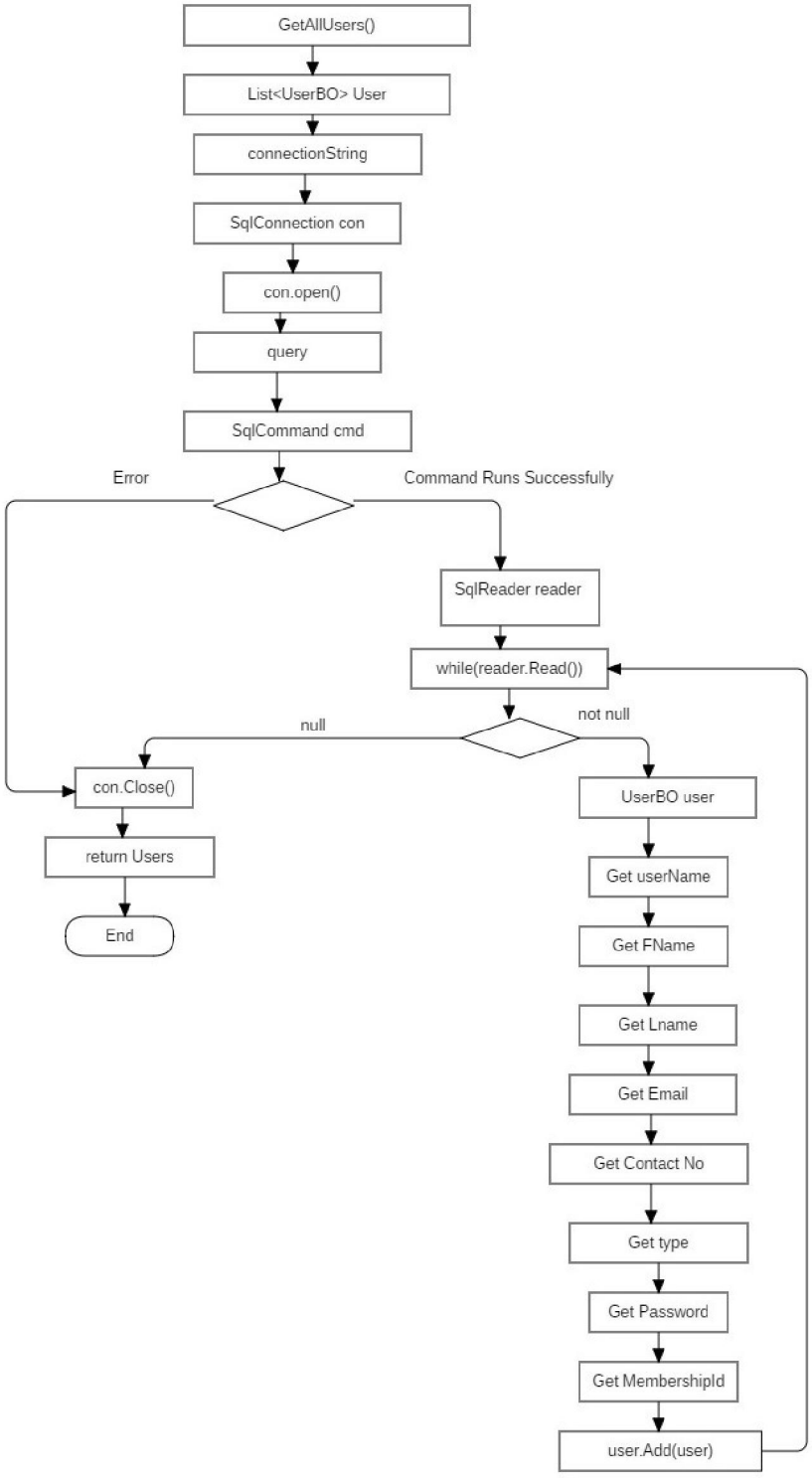
## FC - 6 Traveller Flow Diagram




## FC- 6.1 Register User Flow Diagram

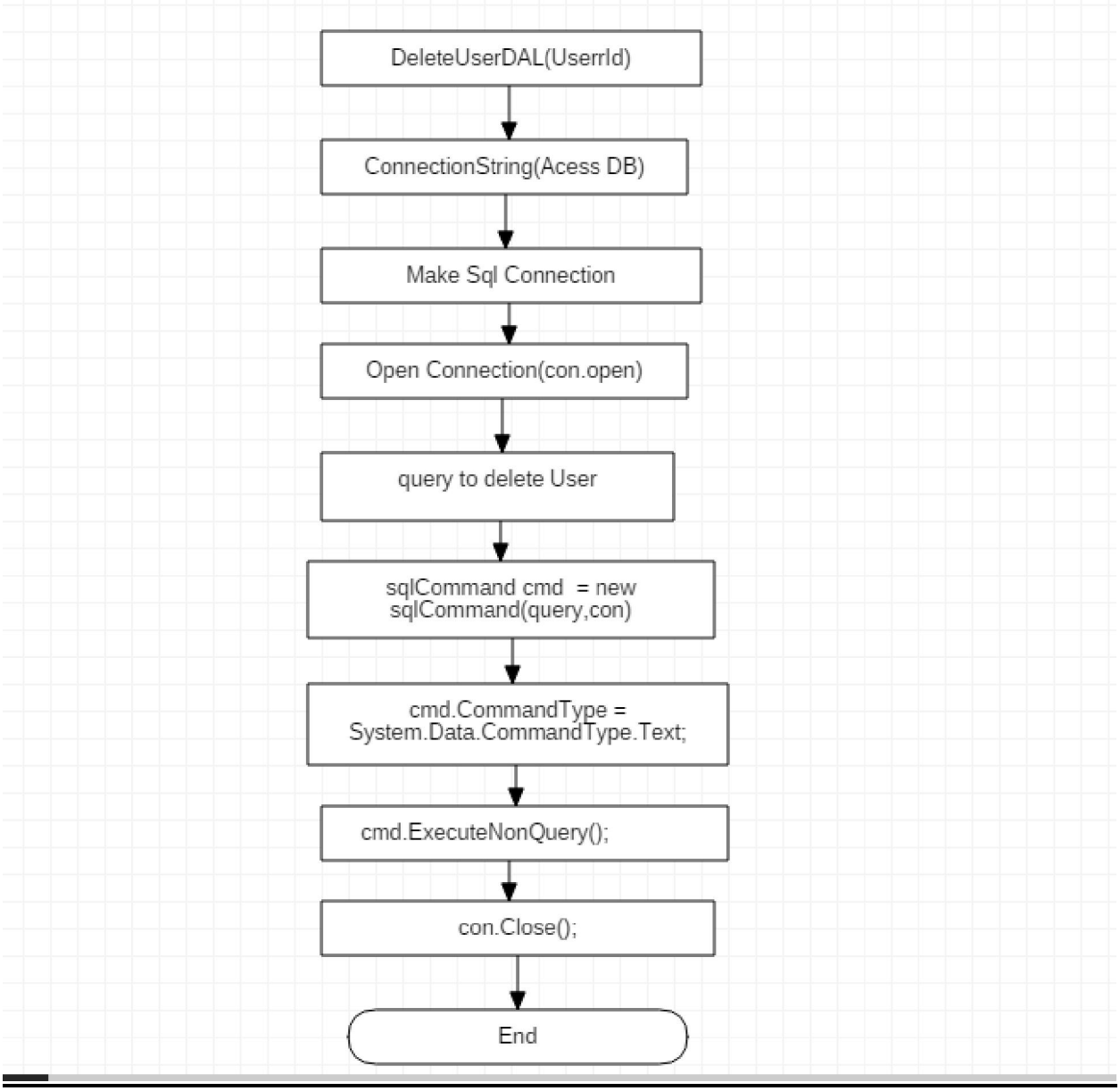



**FC- 6.2 Retrieve Users List Flow Diagram**




---

## FC - 6.3 Delete User Flow Diagram




## 6.3 Test Cases and Coverage

**Test Cases for Manage Traveler (Register Traveler)**

All possible paths are:

1. 1A-2A-3A-6A-7A-1B-2B-3B-4B-5B-6B
2. 1A-2A-3A-4A-12A
3. 1A-2A-3A-6A-7A-1B-2B-3B-4B-7B-8B-9B-10B-16B-17B-25C-26C-27C-9A-10A-11A-1D-2D-3D-4D-5D-6D-14D-15D-16D-12A
4. 1A-2A-3A-6A-7A-1B-2B-3B-4B-7B-8B-9B-10B-9A-10A-11A-1D-2D-3D-4D-5D-6D-7D-8D-9D10D-11D-12D-13D-12A

5. 1A-2A-3A-6A-7A-1B-2B-3B-4B-7B-8B-9B-10B-9A-10A-11A-1D-2D-3D-4D-14D-15D-16D-12A
6. 1A-2A-3A-6A-8A-1C-2C-3C-4C-5C-6C-7C-8C-9C-10C-11A-1D-2D-3D-4D-14D-15D-16D-12A
7. 1A-2A-3A-6A-8A-1C-2C-3C-4C-5C-6C-7C-8C-9C-10C-11A-1D-2D-3D-4D-5D-6D-7D-8D-9D10D-11D-12D-13D-12A
8. 1A-2A-3A-6A-8A-1C-2C-3C-4C-5C-6C-7C-8C-9C-10C-11A--1D-2D-3D-4D-5D-6D-14D-15D-16D-12A

**Statement Testing**

Total No. of statements=62
Statements covered =57

Coverage = Statements covered / Total No. of statements
Coverage=57/62 = 92 %

| | |
|---|---|
| **Identifier** | TC6-1 |
| **Priority** | High |
| **Short description** | This test case is used to test the Traveler added by Admin covering all statements. |
| **Pre-condition(s)** | Admin should be logged-in. |
| **Post-condition(s)** | Traveler will be added. |
| **Input Data** | Traveler Id=10, User Name=”Ayesha12” ,<br>FirstName =”Ayesha” ,<br>LastName=”Aziz”, Type =”user” ,<br>Password = “1234”,Membership Id =2. |
| **Detailed Steps** | 1. Admin will log-in.<br>2. Click on Tab “Traveler”.<br>3. Enter valid Traveler Id.<br>4. Enter valid User Name.<br>5. Enter valid First Name.<br>6. Enter valid Last Name.<br>7. Enter valid Type.<br>8. Enter valid Password.<br>9. Enter valid Membership Id.<br>10. Click on Add Button. |

| | |
|---|---|
| **Expected Results** | Traveler will be added and displayed on the page. |

**Branch Testing (valid inputs)**

Total No. of branches =10
Branches covered =10
Coverage = branches covered / Total No. of branches
Coverage=10/10 = 100 %

| | |
|---|---|
| **Identifier** | TC6-2 |
| **Priority** | High |
| **Short description** | This test case is used to test the Traveler added by the admin covering all branches. |
| **Pre-condition(s)** | Admin should be logged-in. |
| **Post-condition(s)** | Traveler will be added. |
| **Input Data** | Traveler Id=10, User Name=”Ayesha12” ,<br>FirstName =”Ayesha” ,<br>LastName=”Aziz”, Type =”user” ,<br>Password = “1234”,Membership Id =2. |
| **Detailed Steps** | 1. Admin will log-in.<br>2. Click on Tab “Traveler”.<br>3. Enter valid Traveler Id.<br>4. Enter valid User Name.<br>5. Enter valid First Name.<br>6. Enter valid Last Name.<br>7. Enter valid Type.<br>8. Enter valid Password.<br>9. Enter valid Membership Id.<br>10. Click on Add Button. |
| **Expected Results** | Traveler will be added and displayed on the page. |

**Branch Testing (in-valid inputs)**

| | |
|---|---|
| **Identifier** | TC6-3 |

| | |
|---|---|
| **Priority** | High |
| **Short description** | This test case is used to test the traveler added by the admin covering all branches. |
| **Pre-condition(s)** | Admin is  logged-in. |
| **Post-condition(s)** | Admin Home Page will be displayed. |
| **Input Data** | Traveler Id=-4, User Name=”12” ,<br>FirstName =023 ,<br>LastName= -123, Type =”xvw” ,<br>Password = “1234”,Membership Id =0. |
| **Detailed Steps** | 1. Admin will log-in.<br>2. Click on Tab “Traveler”.<br>3. Enter invalid Traveler Id.<br>4. Enter invalid User Name.<br>5. Enter invalid First Name.<br>6. Enter invalid Last Name.<br>7. Enter invalid Type.<br>8. Enter invalid Password.<br>9. Enter invalid Membership Id.<br>10. Click on Add Button. |
| **Expected Results** | Error Message will be displayed and Admin will redirect to Admin Home Page. |

**Path Testing (Register Traveler – Valid Login)**

All possible paths are:

1. 1A-2A-3A-6A-7A-1B-2B-3B-4B-5B-6B
2. 1A-2A-3A-4A-12A
3. 1A-2A-3A-6A-7A-1B-2B-3B-4B-7B-8B-9B-10B-16B-17B-25C-26C-27C-9A-10A-11A-1D-2D-3D-4D-5D-6D-14D-15D-16D-12A
4. 1A-2A-3A-6A-7A-1B-2B-3B-4B-7B-8B-9B-10B-9A-10A-11A-1D-2D-3D-4D-5D-6D-7D-8D-9D10D-11D-12D-13D-12A
5. 1A-2A-3A-6A-7A-1B-2B-3B-4B-7B-8B-9B-10B-9A-10A-11A-1D-2D-3D-4D-14D-15D-16D-12A
6. 1A-2A-3A-6A-8A-1C-2C-3C-4C-5C-6C-7C-8C-9C-10C-11A-1D-2D-3D-4D-14D-15D-16D-12A
7. 1A-2A-3A-6A-8A-1C-2C-3C-4C-5C-6C-7C-8C-9C-10C-11A-1D-2D-3D-4D-5D-6D-7D-8D-9D10D-11D-12D-13D-12A

8. 1A-2A-3A-6A-8A-1C-2C-3C-4C-5C-6C-7C-8C-9C-10C-11A--1D-2D-3D-4D-5D-6D-14D-15D-16D-12A

| | |
|---|---|
| **Identifier** | TC6-4 |
| **Priority** | High |
| **Short description** | This test case is used to test the traveler added by the admin covering all branches. |
| **Pre-condition(s)** | Admin should be logged-in. |
| **Post-condition(s)** | Traveler will be added. |
| **Input Data** | Traveler Id=10, User Name=”Ayesha12” ,<br>FirstName =”Ayesha” ,<br>LastName=”Aziz”, Type =”user” ,<br>Password = “1234”, Membership Id =2. |
| **Detailed Steps** | 1. Admin will log-in.<br>2. Click on Tab “Traveler”.<br>3. Enter valid Traveler Id.<br>4. Enter valid User Name.<br>5. Enter valid First Name.<br>6. Enter valid Last Name.<br>7. Enter valid Type.<br>8. Enter valid Password.<br>9. Enter valid Membership Id.<br>10. Click on Add Button. |
| **Expected Results** | Traveler will be added and displayed on the page. |

**Path Testing (Register Traveler – Invalid Login (Session Not Implemented))**

| | |
|---|---|
| **Identifier** | TC6-5 |
| **Priority** | High |

| | |
|---|---|
| **Short description** | This test case is used to test the traveler added by the Admin covering all branches. |
| **Pre-condition(s)** | Admin is not logged-in (Session not implemented). |
| **Post-condition(s)** | Admin Home Page will be displayed. |
| **Input Data** | Traveler Id=10, User Name=”Ayesha12” ,<br>FirstName =”Ayesha” ,<br>LastName=”Aziz”, Type =”user” ,<br>Password = “1234”,Membership Id =2. |
| **Detailed Steps** | 1. Click on Tab “Traveler”.<br>2. Enter valid Traveler Id.<br>3. Enter valid User Name.<br>4. Enter valid First Name.<br>5. Enter valid Last Name.<br>6. Enter valid Type.<br>7. Enter valid Password.<br>8. Enter valid Membership Id.<br>9. Click on Add Button. |
| **Expected Results** | Redirect to Admin Home Page. |

**Loop Boundary Testing**

As there is no loop in the code, so loop boundary testing will not be performed.

**Test Cases for Manage Travelers (Delete Traveler**

All possible paths are:

1. 1A-2A-3A-6A-7A-1B-2B-3B-4B-5B-6B
2. 1A-2A-3A-4A-12A
3. 1A-2A-3A-6A-7A-1B-2B-3B-4B-7B-8B-9B-10B-16B-17B-25C-26C-27C-9A-10A-11A-1D-2D-3D-4D-5D-6D-14D-15D-16D-12A
4. 1A-2A-3A-6A-7A-1B-2B-3B-4B-7B-8B-9B-10B-9A-10A-11A-1D-2D-3D-4D-5D-6D-7D-8D-9D10D-11D-12D-13D-12A
5. 1A-2A-3A-6A-7A-1B-2B-3B-4B-7B-8B-9B-10B-9A-10A-11A-1D-2D-3D-4D-14D-15D-16D-12A
6. 1A-2A-3A-6A-8A-1C-2C-3C-4C-5C-6C-7C-8C-9C-10C-11A-1D-2D-3D-4D-14D-15D-16D-12A
7. 1A-2A-3A-6A-8A-1C-2C-3C-4C-5C-6C-7C-8C-9C-10C-11A-1D-2D-3D-4D-5D-6D-7D-8D-9D10D-11D-12D-13D-12A

8. 1A-2A-3A-6A-8A-1C-2C-3C-4C-5C-6C-7C-8C-9C-10C-11A--1D-2D-3D-4D-5D-6D-14D-15D-16D-12A

**Statement Testing**
Total No. of statements=62
Statements covered =57
Coverage = Statements covered / Total No. of statements
Coverage=57/62 = 92 %

| | |
|---|---|
| **Identifier** | TC6-1 |
| **Priority** | High |
| **Short description** | This test case is used to test the Delete Traveler Function covering all statements. |
| **Pre-condition(s)** | Admin should be logged-in. |
| **Post-condition(s)** | Traveler will be Deleted. |
| **Input Data** | Traveler Id=1 |
| **Detailed Steps** | 1. Admin will log-in.<br>2. Click on Tab “Traveler”.<br>3. Enter valid Traveler Id.<br>4. Click on Delete Button. |
| **Expected Results** | Traveler will be Delete and updated Traveler list will be displayed on the page. |

**Branch Testing (valid input)**
Total No. of branches =10
Branches covered =10
Coverage = branches covered / Total No. of branches
Coverage=10/10 = 100 %

| | |
|---|---|
| **Identifier** | TC6-2 |
| **Priority** | High |

| | |
|---|---|
| **Short description** | This test case is used to test the Delete Traveler Function covering all statements. |
| **Pre-condition(s)** | Admin should be logged-in. |
| **Post-condition(s)** | Traveler will be Deleted. |
| **Input Data** | Traveler Id=1 |
| **Detailed Steps** | 1. Admin will log-in.<br>2. Click on Tab “Traveler”.<br>3. Enter valid Traveler Id.<br>4. Click on Delete Button. |
| **Expected Results** | Traveler will be Delete and updated Traveler list will be displayed on the page. |

**Branch Testing (in-valid inputs)**

| | |
|---|---|
| **Identifier** | TC6-3 |
| **Priority** | High |
| **Short description** | This test case is used to test the Delete Traveler Function covering all statements. |
| **Pre-condition(s)** | Admin is  logged-in. |
| **Post-condition(s)** | Admin Home Page will be displayed. |
| **Input Data** | Traveler Id=-1 |
| **Detailed Steps** | 1. Admin will log-in.<br>2. Click on Tab “Traveler”.<br>3. Enter invalid Traveler Id.<br>4. Click on Delete Button. |
| **Expected Results** | Error Message will be displayed and Admin will redirect to Admin Home Page. |

**Path Testing (Delete Traveler – Valid Login)**
All possible paths are:

1. 1A-2A-3A-6A-7A-1B-2B-3B-4B-5B-6B
2. 1A-2A-3A-4A-12A
3. 1A-2A-3A-6A-7A-1B-2B-3B-4B-7B-8B-9B-10B-16B-17B-25C-26C-27C-9A-10A-11A-1D-2D-3D-4D-5D-6D-14D-15D-16D-12A
4. 1A-2A-3A-6A-7A-1B-2B-3B-4B-7B-8B-9B-10B-9A-10A-11A-1D-2D-3D-4D-5D-6D-7D-8D-9D10D-11D-12D-13D-12A
5. 1A-2A-3A-6A-7A-1B-2B-3B-4B-7B-8B-9B-10B-9A-10A-11A-1D-2D-3D-4D-14D-15D-16D-12A
6. 1A-2A-3A-6A-8A-1C-2C-3C-4C-5C-6C-7C-8C-9C-10C-11A-1D-2D-3D-4D-14D-15D-16D-12A
7. 1A-2A-3A-6A-8A-1C-2C-3C-4C-5C-6C-7C-8C-9C-10C-11A-1D-2D-3D-4D-5D-6D-7D-8D-9D10D-11D-12D-13D-12A
8. 1A-2A-3A-6A-8A-1C-2C-3C-4C-5C-6C-7C-8C-9C-10C-11A--1D-2D-3D-4D-5D-6D-14D-15D-16D-12A

| | |
|---|---|
| **Identifier** | TC6-4 |
| **Priority** | High |
| **Short description** | This test case is used to test the Delete Traveler Function covering all statements. |
| **Pre-condition(s)** | Admin should be logged-in. |
| **Post-condition(s)** | Traveler will be Deleted. |
| **Input Data** | Traveler Id=5 |
| **Detailed Steps** | 1. Admin will log-in.<br>2. Click on Tab “Traveler”.<br>3. Enter valid Traveler Id.<br>4. Click on Delete Button. |
| **Expected Results** | Traveler will be deleted and updated tour list will be displayed on the page. |

**Path Testing (Delete Traveler – Invalid Login (Session Not Implemented))**

| | |
|---|---|
| **Identifier** | TC6-5 |
| **Priority** | High |

| | |
|---|---|
| **Short description** | This test case is used to test the Delete Traveler Function covering all statements. |
| **Pre-condition(s)** | Admin is not logged-in (Session not implemented). |
| **Post-condition(s)** | Admin Home Page will be displayed. |
| **Input Data** | Traveler Id=5 |
| **Detailed Steps** | 1. Click on Tab “Traveler”.<br>2. Enter valid Traveler Id.<br>3. Click on Delete Button. |
| **Expected Results** | Redirect to Admin Home Page. |

**Loop Boundary Testing**

As there is no loop in the code, so loop boundary testing will not be performed.