PRACTICAL-8

Website development

Que.1: Write down steps for establishing database connection from a web page on an asp.net website.

Establishing a database connection from a web page in an ASP.NET website typically involves several steps. Here's a general outline of the process:

1. **Choose a Database Management System (DBMS):**
   Before you start, decide which DBMS you want to use, such as SQL Server, MySQL, PostgreSQL, or others. Make sure you have it installed and running.

2. **Create a Database:**
   Create a database within your chosen DBMS. Define tables, columns, and relationships as needed for your application.

3. **Install Required Packages:**
   If you're using Entity Framework or another ORM (Object-Relational Mapping) tool, install the necessary NuGet packages for your project.

4. **Configuration:**
   Configure the connection string. This is typically done in the `web.config` or `appsettings.json` file. Here's an example for SQL Server:

   In `web.config` (for older ASP.NET projects):

   ```xml
   <connectionStrings>
       <add name="DefaultConnection"
            connectionString="Server=YourServer;Database=YourDatabase;User
Id=YourUser;Password=YourPassword;"
            providerName="System.Data.SqlClient" />
   </connectionStrings>
   ```

   In `appsettings.json` (for ASP.NET Core projects):

   ```json
   {
     "ConnectionStrings": {
       "DefaultConnection": "Server=YourServer;Database=YourDatabase;User
Id=YourUser;Password=YourPassword;"
     }
   }
   ```

5. **Import Namespaces:**
   In your code-behind file (C# or VB.NET), import the necessary namespaces. For SQL Server, you might need `System.Data.SqlClient` or for Entity Framework, `System.Data.Entity`.

6. **Create a Database Connection Object:**
   Instantiate a database connection object. For SQL Server, it would look like this:

```csharp
using System.Data.SqlClient;

// ...

SqlConnection connection = new
SqlConnection(ConfigurationManager.ConnectionStrings["DefaultConnection"].ConnectionString);
```

For Entity Framework, you'd create a DbContext instance.

7. **Open the Connection:**
   Open the database connection using the `Open()` method.

```csharp
connection.Open();
```

8. **Perform Database Operations:**
   You can now execute SQL queries or use ORM methods to interact with the database. For example:

```csharp
SqlCommand command = new SqlCommand("SELECT * FROM YourTable", connection);
SqlDataReader reader = command.ExecuteReader();

// Process the results or update the database as needed

// Close the reader and the connection when done
reader.Close();
connection.Close();
```

9. **Close the Connection:**
   It's essential to close the connection when you're done to release resources.

10. **Exception Handling:**
    Always use try-catch blocks to handle exceptions that may occur during database operations. This ensures graceful error handling and prevents resource leaks.

11. **Dispose of Resources:**
    In an ASP.NET application, it's good practice to use `using` statements or explicitly call the `Dispose` method for database-related objects to release resources when they are no longer needed.

12. **Testing and Debugging:**
    Thoroughly test your database connection and operations to ensure they work as expected.

PRACTICAL-8

Que.2: Create a website with necessary validation control, Database and following Web-pages.

- Register User (containing fields name, surname, username, password, email, gender, city)
- Login Page (Put a sign-up link. If login fails, it will display an error message)
- Success (If user logins successfully, he/she will be redirected to page.
- Display a welcome message along with the name of the user)

Register.aspx:

```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="Que2.aspx.cs" Inherits="LAB8.Que2" %>

<!DOCTYPE html>

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
  <title>Registration Form</title>
</head>
<body>
  <form id="form1" runat="server">
    <div>
      <h2>Registration Form</h2>
      <div>
        <asp:Label ID="lblName" runat="server" Text="Name:"></asp:Label>
        <asp:TextBox ID="txtName" runat="server"></asp:TextBox>
        <asp:RequiredFieldValidator ID="RequiredFieldValidator1" runat="server" ControlToValidate="txtName"
ErrorMessage="*"></asp:RequiredFieldValidator>
        <asp:RegularExpressionValidator ID="RegularExpressionValidator1" runat="server" ControlToValidate="txtName"
ErrorMessage="Only Alpha Allowed...!!!" ValidationExpression="[A-Za-z]"></asp:RegularExpressionValidator>
      </div>
      <div>
        <asp:Label ID="lblSurname" runat="server" Text="Surname:"></asp:Label>
        <asp:TextBox ID="txtSurname" runat="server"></asp:TextBox>
        <asp:RequiredFieldValidator ID="RequiredFieldValidator2" runat="server" ControlToValidate="txtSurname"
ErrorMessage="*"></asp:RequiredFieldValidator>
        <asp:RegularExpressionValidator ID="RegularExpressionValidator4" runat="server" ControlToValidate="txtSurname"
ErrorMessage="Only Alpha Allowed...!!!" ValidationExpression="[A-Za-z]"></asp:RegularExpressionValidator>
      </div>
      <div>
        <asp:Label ID="lblUsername" runat="server" Text="Username:"></asp:Label>
        <asp:TextBox ID="txtUsername" runat="server"></asp:TextBox>
        <asp:RequiredFieldValidator ID="RequiredFieldValidator3" runat="server" ControlToValidate="txtUsername"
ErrorMessage="*"></asp:RequiredFieldValidator>
        <asp:RegularExpressionValidator ID="RegularExpressionValidator3" runat="server" ControlToValidate="txtUsername"
ErrorMessage="Digits, Letters(At least on uppercase) and underscore" ValidationExpression="^(?=.*[A-Z])[\dA-Za-
z_]+$"></asp:RegularExpressionValidator>
      </div>
      <div>
        <asp:Label ID="lblPassword" runat="server" Text="Password:"></asp:Label>
        <asp:TextBox ID="txtPassword" runat="server" TextMode="Password"></asp:TextBox>
        <asp:RequiredFieldValidator ID="RequiredFieldValidator4" runat="server" ControlToValidate="txtPassword"
ErrorMessage="*"></asp:RequiredFieldValidator>
        <asp:RegularExpressionValidator ID="RegularExpressionValidator5" runat="server" ControlToValidate="txtPassword"
ErrorMessage="Minimum length 8 characters, Including One letter- one digit-one" ValidationExpression="^(?=.[A-Za-
z])(?=.\d)(?=.*[^A-Za-z\d]).{8,}$"></asp:RegularExpressionValidator>
```

```
        </div>
        <div>
          <asp:Label ID="lblEmail" runat="server" Text="Email:"></asp:Label>
          <asp:TextBox ID="txtEmail" runat="server"></asp:TextBox>
          <asp:RequiredFieldValidator ID="RequiredFieldValidator5" runat="server" ControlToValidate="txtEmail"
ErrorMessage="*"></asp:RequiredFieldValidator>
          <asp:RegularExpressionValidator ID="RegularExpressionValidator2" runat="server" ErrorMessage="Invalid Email Address...."
ValidationExpression="\w+([-+.']\w+)@\w+([-.]\w+)\.\w+([-.]\w+)*"></asp:RegularExpressionValidator>
        </div>
        <div>
          <asp:Label ID="lblGender" runat="server" Text="Gender:"></asp:Label>
          <asp:DropDownList ID="ddlGender" runat="server">
            <asp:ListItem Text="Male" Value="Male"></asp:ListItem>
            <asp:ListItem Text="Female" Value="Female"></asp:ListItem>
            <asp:ListItem Text="Other" Value="Other"></asp:ListItem>
          </asp:DropDownList>
        </div>
        <div>
          <asp:Label ID="lblCity" runat="server" Text="City:"></asp:Label>
          <asp:TextBox ID="txtCity" runat="server"></asp:TextBox>
          <asp:RequiredFieldValidator ID="RequiredFieldValidator6" runat="server" ControlToValidate="txtCity"
ErrorMessage="*"></asp:RequiredFieldValidator>
          <asp:RegularExpressionValidator ID="RegularExpressionValidator6" runat="server" ControlToValidate="txtCity"
ErrorMessage="Only Alpha Allowed...!!!" ValidationExpression="[A-Za-z]"></asp:RegularExpressionValidator>
        </div>
        <div>
          <asp:Button ID="btnRegister" runat="server" Text="Register" OnClick="btnRegister_Click" />
        </div>
      </div>
    </form>
</body>
</html>
```

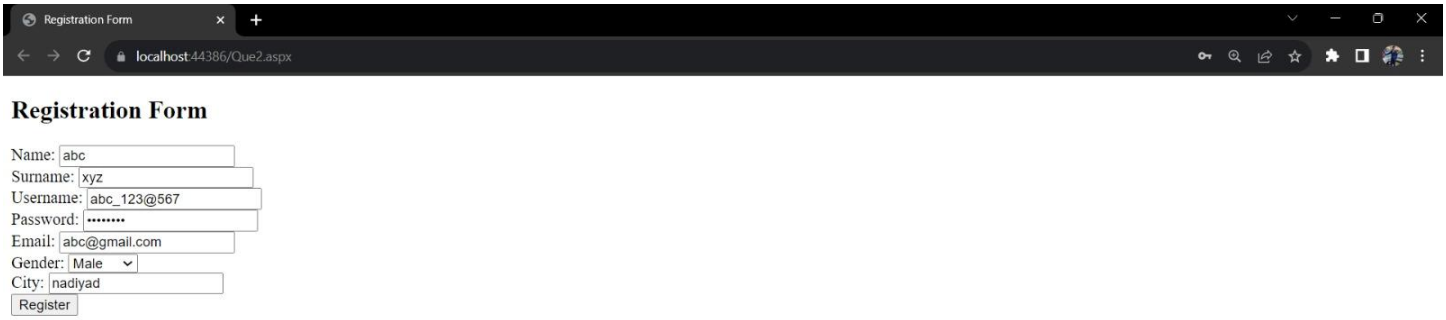
Register.aspx.cs:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Data.SqlClient;
namespace LAB8
{
    public partial class Que2 : System.Web.UI.Page
    {
        SqlConnection con;
        //SqlCommand cmd;
        protected void Page_Load(object sender, EventArgs e)
        {
            con = new SqlConnection(@"Data
Source=(LocalDB)\MSSQLLocalDB;AttachDbFilename=C:\Users\iamja\OneDrive\Documents\LAB8.mdf;Integrated Security=True;");
}
```

```
    protected void btnRegister_Click(object sender, EventArgs e)
    {
        // Retrieve values from the form controls
        string name = txtName.Text;
        string surname = txtSurname.Text;
        string username = txtUsername.Text;
        string password = txtPassword.Text;
        string email = txtEmail.Text;
        string gender = ddlGender.SelectedValue;
        string city = txtCity.Text;

        // Define your SQL con string

        // Create an SQL con and SQL command
        using (con)
        {
            con.Open();

            // Define the SQL insert query
            string insertQuery = "INSERT INTO [User] (Name, Surname, Username, Password, Email, Gender, City) " +
                                 "VALUES (@Name, @Surname, @Username, @Password, @Email, @Gender, @City)";

            using (SqlCommand cmd = new SqlCommand(insertQuery, con))
            {
                // Add parameters to the query to prevent SQL injection
                cmd.Parameters.AddWithValue("@Name", name);
                cmd.Parameters.AddWithValue("@Surname", surname);
                cmd.Parameters.AddWithValue("@Username", username);
                cmd.Parameters.AddWithValue("@Password", password);
                cmd.Parameters.AddWithValue("@Email", email);
                cmd.Parameters.AddWithValue("@Gender", gender);
                cmd.Parameters.AddWithValue("@City", city);

                // Execute the SQL command to insert the data
                if(cmd.ExecuteNonQuery()>0)
                {
                    Response.Write("<script>alert('Record inserted...')</script>");
                }
            }
        }

    }
}
```

Outputs

PRACTICAL-8



Login.aspx:

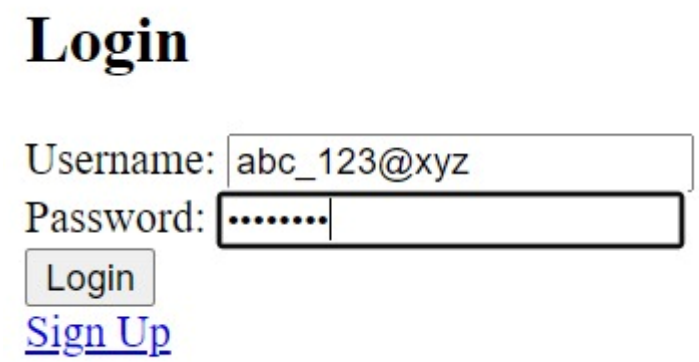
```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="login.aspx.cs" Inherits="LAB8.login" %>

<!DOCTYPE html>

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
   <title>Login Page</title>
</head>
<body>
   <form id="form1" runat="server">
      <div>
         <h2>Login</h2>
         <asp:Label ID="lblMessage" runat="server" ForeColor="Black" Visible="False"></asp:Label>
         <div>
            <asp:Label ID="lblUsername" runat="server" Text="Username:"></asp:Label>
            <asp:TextBox ID="txtUsername" runat="server"></asp:TextBox>
         </div>
         <div>
            <asp:Label ID="lblPassword" runat="server" Text="Password:"></asp:Label>
            <asp:TextBox ID="txtPassword" runat="server" TextMode="Password"></asp:TextBox>
         </div>
         <div>
            <asp:Button ID="btnLogin" runat="server" Text="Login" OnClick="btnLogin_Click" />
         </div>
         <div>
            <asp:HyperLink ID="lnkSignUp" runat="server" NavigateUrl="signup.aspx" Text="Sign Up" />
         </div>
      </div>
```

```
   </form>
</body>
</html>
```

Login.aspx.cs

```
using System;
using System.Collections.Generic;
using System.Data.SqlClient;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;

namespace LAB8
{
   public partial class login : System.Web.UI.Page
   {
      protected void Page_Load(object sender, EventArgs e)
      {

      }
      private bool IsValidCredentials(string username, string password)
      {
         // Set your database connection string
         string connectionString = @"Data
Source=(LocalDB)\MSSQLLocalDB;AttachDbFilename=C:\Users\iamja\OneDrive\Documents\LAB8.mdf;Integrated Security=True;";

         // Define a SQL query to check if the provided username and password match
         string query = "SELECT COUNT(*) FROM [User] WHERE Username = @Username AND Password = @Password";

         using (SqlConnection connection = new SqlConnection(connectionString))
         using (SqlCommand cmd = new SqlCommand(query, connection))
         {
            // Add parameters to the query
            cmd.Parameters.AddWithValue("@Username", username);
            cmd.Parameters.AddWithValue("@Password", password);

            connection.Open();
            int result = (int)cmd.ExecuteScalar();

            // If a matching user is found, result will be 1 (valid), otherwise 0 (invalid)
            return result == 1;
         }
      }
      protected void btnLogin_Click(object sender, EventArgs e)
      {
```

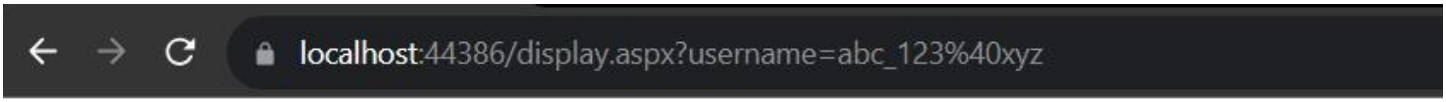
```
        string username = txtUsername.Text;
        string password = txtPassword.Text;

        // Perform authentication logic (e.g., check credentials against a database)
        if (IsValidCredentials(username, password))
        {
            // Authentication successful
            // You can redirect the user to a different page or perform other actions here.
            lblMessage.Visible = true;
            lblMessage.Text = "Login successful!";
            Response.Redirect($"display.aspx?username={HttpUtility.UrlEncode(username)}");
        }
        else
        {
            // Authentication failed
            lblMessage.Visible = true;
            lblMessage.Text = "Invalid username or password.";
        }
    }
  }

}
```



Display.aspx

```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="display.aspx.cs" Inherits="LAB8.display" %>

<!DOCTYPE html>

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title></title>
</head>
<body>
    <form id="form1" runat="server">
        <div>
```

```
        <h2>Welcome, <%: Request.QueryString["username"] %></h2>
    </div>
  </form>
</body>
</html>
```



Welcome, abc_123@xyz