HomeController.cs

```
using Assignment3.DB;
using Assignment3.Models;
using System.Collections.Generic;
using System.Linq;
using System.Web.Mvc;

namespace Assignment3.Controllers
{
    public class HomeController : Controller
    {
        private AppDbContext ctx;

        public HomeController()
        {
            ctx = new AppDbContext();
        }

        public ActionResult Index()
        {
            if (Session["Id"] == null)
            {
                return RedirectToAction("Login"); // Redirect to the login page if the user is not logged in
            }

            var model = ctx.Products.ToList();
            int userId = (int)Session["Id"];
            string username = (string)Session["Username"];

            return View(model);
        }

        [HttpGet]
        public ActionResult Login()
        {
            return View();
        }

        [HttpPost]
        public ActionResult Login(string username, string password)
        {
            var user = ctx.Users.FirstOrDefault(u => u.Username == username && u.Password == password);

            if (user != null)
            {
                // Set session variables
                Session["Id"] = user.Id;
                Session["Username"] = user.Username;

                return RedirectToAction("Index"); // Redirect to the home page after successful login
            }

            ViewBag.ErrorMessage = "Invalid username or password";
            return View();
        }

        public ActionResult Logout()
        {
            Session.Abandon(); // Delete the session

            return RedirectToAction("Login"); // Redirect to the login page after logout
        }


        //
        // GET: /Home/Details/5
        public ActionResult Details(int id)
        {
            return View();
        }

        //
        // GET: /Home/Create
        public ActionResult Create()
        {
            return View();
        }


        //
        // POST: /Home/Create
        [HttpPost]
        public ActionResult Create(Products0025 cnt)
        {
            try
            {
                ctx.Products.Add(cnt);
                ctx.SaveChanges();
                return RedirectToAction("Index");
            }
            catch
            {
                return View();
            }
        }

        //
        // GET: /Home/Edit/5
        public ActionResult Edit(int id)
        {
            return View();
        }

        //
        // POST: /Home/Edit/5
        [HttpPost]
        public ActionResult Edit(int id, Products0025 cnt)
        {
            try
            {
                // TODO: Add update logic here
                ctx.Entry(cnt).State = System.Data.Entity.EntityState.Modified;
                ctx.SaveChanges();
                return RedirectToAction("Index");
            }
            catch
            {
                return View();
            }
        }

        //
        // GET: /Home/Delete/5
        public ActionResult Delete(int id)
        {
            var keyid = ctx.Products.Find(id);
            return View(keyid);
        }

        // POST: /Home/Delete/5
        [HttpPost]
        [ActionName("Delete")]
        public ActionResult DeleteConfirmed(int id)
        {
            var product = ctx.Products.Find(id);

            try
            {
                ctx.Products.Remove(product);
                ctx.SaveChanges();

                return RedirectToAction("Index");
            }
            catch
            {
                return View();
            }
        }


        public ActionResult AddToCart(int id)
        {
            var product = ctx.Products.Find(id);

            if (product != null)
            {
                var cart = GetCart();
                var cartItem = cart.FirstOrDefault(item => item.ProductId == id);

                if (cartItem != null)
                {
                    // Increment the quantity if the product already exists in the cart
                    cartItem.Quantity++;
                }
                else
                {
                    // Add the product to the cart with a quantity of 1
                    cart.Add(new CartItem
                    {
                        ProductId = product.Id,
                        ProductName = product.Name,
                        Price = product.Price,
                        Quantity = 1
                    });
                }

                SaveCart(cart);
            }

            return RedirectToAction("Index");
        }

        public ActionResult RemoveFromCart(int id)
        {
            var cart = GetCart();
            var cartItem = cart.FirstOrDefault(item => item.ProductId == id);

            if (cartItem != null)
            {
                cart.Remove(cartItem);
                SaveCart(cart);
            }

            return RedirectToAction("Cart");
        }

        public ActionResult Cart()
        {
            var cart = GetCart();
            return View(cart);
        }

        private List<CartItem> GetCart()
        {
            var cart = Session["Cart"] as List<CartItem>;

            if (cart == null)
            {
                cart = new List<CartItem>();
                Session["Cart"] = cart;
            }

            return cart;
        }

        private void SaveCart(List<CartItem> cart)
        {
            Session["Cart"] = cart;
        }
    }
}
```

Index.cshtml View

```
@model IEnumerable<Assignment3.Models.Products0025>

@{
    ViewBag.Title = "Index";
    bool isLoggedIn = (Session["Id"] != null);
}

<!DOCTYPE html>
<html>
<head>
    <title>@ViewBag.Title</title>
    <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.5.2/css/bootstrap.min.css" integrity="sha384-
pzjw8b+Ue5p1wqTN8TlXW9Fq8C9us21stO9SQiqyTGvRtS9l7+a+jrDI+Q7bq5T9" crossorigin="anonymous">
    <style>
        table {
            width: 100%;
            max-width: 100%;
        }

        th {
            font-weight: bold;
        }

        td, th {
            padding: 8px;
            text-align: left;
            border-bottom: 1px solid #ddd;
        }

        .btn-group {
            white-space: nowrap;
        }
    </style>
</head>
<body>

    @if (isLoggedIn)
    {
        <form action="@Url.Action("Logout", "Home")" method="post">
            <input type="submit" value="Logout" />
        </form>
    }

    <h1 style="text-align: center">Index</h1>

    <p>
        <button onclick="location.href='@Url.Action("Cart")'" class="btn btn-primary">MY CART</button>

    </p>
    <p>
        <button onclick="location.href='@Url.Action("Create")'" class="btn btn-primary">Create New</button>
    </p>

    <table class="table">
        <thead>
            <tr>
                <th>@Html.DisplayNameFor(model => model.Id)</th>
                <th>@Html.DisplayNameFor(model => model.Code)</th>
                <th>@Html.DisplayNameFor(model => model.Name)</th>
                <th>@Html.DisplayNameFor(model => model.Price)</th>
                <th>@Html.DisplayNameFor(model => model.Color)</th>
                <th>@Html.DisplayNameFor(model => model.Dimension)</th>
                <th>@Html.DisplayNameFor(model => model.Description)</th>
                <th>@Html.DisplayNameFor(model => model.Brand)</th>
                <th></th>
            </tr>
        </thead>
        <tbody>
            @foreach (var item in Model)
            {
                <tr>
                    <td>@Html.DisplayFor(modelItem => item.Id)</td>
                    <td>@Html.DisplayFor(modelItem => item.Code)</td>
                    <td>@Html.DisplayFor(modelItem => item.Name)</td>
                    <td>@Html.DisplayFor(modelItem => item.Price)</td>
                    <td>@Html.DisplayFor(modelItem => item.Color)</td>
                    <td>@Html.DisplayFor(modelItem => item.Dimension)</td>
                    <td>@Html.DisplayFor(modelItem => item.Description)</td>
                    <td>@Html.DisplayFor(modelItem => item.Brand)</td>
                    <td>
                        <div class="btn-group" role="group">
                            <button onclick="location.href='@Url.Action("Edit", new { id = item.Id })'" class="btn btn-primary">Edit</button>
                            <button onclick="location.href='@Url.Action("Delete", new { id = item.Id })'" class="btn btn-primary">Delete</button>
                            <button onclick="location.href='@Url.Action("AddToCart", new { id = item.Id })'" class="btn btn-success">Add to Cart</button>
                        </div>
                    </td>

                </tr>
            }
        </tbody>
    </table>

    <script src="https://stackpath.bootstrapcdn.com/bootstrap/4.5.2/js/bootstrap.min.js" integrity="sha384-
pzjw8b+Ue5p1wqTN8TlXW9Fq8C9us21stO9SQiqyTGvRtS9l7+a+jrDI+Q7bq5T9" crossorigin="anonymous"></script>
</body>
</html>
```

CartItem.cs Model

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;

namespace Assignment3.Models
```

```
{
    public class CartItem
    {
        public int ProductId { get; set; }
        public string ProductName { get; set; }
        public int Price { get; set; }
        public int Quantity { get; set; }
    }

}
```

Products0025.cs Model

```
using System;
using System.Collections.Generic;
using System.ComponentModel.DataAnnotations;
using System.Linq;
using System.Web;

namespace Assignment3.Models
{
    public class Products0025

    {
        [Key]
        public int Id { get; set; }
        public int Code { get; set; }
        public string Name { get; set; }
        public int Price { get; set; }
        public string Color { get; set; }
        public string Dimension { get; set; }
        public string Description { get; set; }
        public string Brand { get; set; }
    }
}
```

Users0025.cs Model

```
using System;
using System.Collections.Generic;
using System.ComponentModel.DataAnnotations;
using System.Linq;
using System.Web;

namespace Assignment3.Models
{
    public class Users0025
    {
        public int Id { get; set; }
        public string Username { get; set; }
        public string Password { get; set; }
    }

}
```

Cart.cshtml

```
@model List<Assignment3.Models.CartItem>


  <table class="table">
        <tr>
            <th>
                Name
            </th>
            <th>
                Price
            </th>
            <th>
                Quantity
            </th>
            <th></th>
        </tr>

        @foreach (var item in Model)
        {
            <tr>
                <td>
                    @Html.DisplayFor(modelItem => item.ProductName)
                </td>
                <td>
                    @Html.DisplayFor(modelItem => item.Price)
                </td>
                <td>
                    @Html.DisplayFor(modelItem => item.Quantity)
                </td>
                <td>
                    <button onclick="location.href='@Url.Action("RemoveFromCart", new { id = item.ProductId })'" class="btn btn-primary">Remove</button>

                    <button type="button" onclick="location.href='@Url.Action("Index")'" class="btn btn-default">Back to List</button>
                </td>
            </tr>
        }

</table>
```

Login.cshtml

```
@using (Html.BeginForm())
            {
                <div class="form-group">
                    <label for="username">Username:</label>
                    <input type="text" name="username" id="username" required class="form-control" />
                </div>
                <div class="form-group">
                    <label for="password">Password:</label>
                    <input type="password" name="password" id="password" required class="form-control" />
                </div>
                <input type="submit" value="Login" class="btn btn-primary" />
            }
```

DB AppDbContext.cs

```
using System;
using System.Collections.Generic;
using System.Data.Entity;
using System.Linq;
using System.Web;
using System.Data.SqlClient;
using Assignment3.Models;

namespace Assignment3.DB
{
    public class AppDbContext : DbContext
    {
        public AppDbContext() : base("name=conn")
        {



        }
        public DbSet<Products0025> Products { get; set; }
        public DbSet<Users0025> Users { get; set; }


    }
}
```

Migrations Configuration.cs

```
namespace Assignment3.Migrations
{
    using System;
    using System.Data.Entity;
    using System.Data.Entity.Migrations;
    using System.Linq;

    internal sealed class Configuration : DbMigrationsConfiguration<Assignment3.DB.AppDbContext>
    {
        public Configuration()
        {
            AutomaticMigrationsEnabled = false;
        }

        protected override void Seed(Assignment3.DB.AppDbContext context)
        {
            //  This method will be called after migrating to the latest version.

            //  You can use the DbSet<T>.AddOrUpdate() helper extension method
            //  to avoid creating duplicate seed data.
        }
    }
}
```

Migrations 202306181610160_InitialCreate2.cs

```
namespace Assignment3.Migrations
{
    using System;
    using System.Data.Entity.Migrations;

    public partial class InitialCreate2 : DbMigration
    {
        public override void Up()
        {
            RenameTable(name: "dbo.Products", newName: "Products0025");
            CreateTable(
                "dbo.Users0025",
                c => new
                    {
                        Id = c.Int(nullable: false, identity: true),
                        Username = c.String(),
                        Password = c.String(),
                    })
                .PrimaryKey(t => t.Id);

        }

        public override void Down()
        {
            DropTable("dbo.Users0025");
            RenameTable(name: "dbo.Products0025", newName: "Products");
        }
    }
}
```

File Upload WebController

```
using System;
using System.Collections.Generic;
using System.IO;
using System.Linq;
using System.Web;
using System.Web.Mvc;

namespace WebApplication26.Controllers
{
    public class WebController : Controller
    {
        //
        // GET: /Web/
        public ActionResult Index()
        {
            return View();
        }

        [HttpPost]
        public ActionResult UploadTextFile(HttpPostedFileBase file)
        {
            if (file != null && file.ContentLength > 0)
            {
                string fileContent = "";
                using (var reader = new StreamReader(file.InputStream))
                {
                    fileContent = reader.ReadToEnd();
                }

                string updatedData = fileContent.ToUpper();
                string newFileName = "M_TextFile.txt";
                string path = Server.MapPath("~/UploadedFiles/") + newFileName;

                System.IO.File.WriteAllText(path, updatedData);
                ViewBag.Content = updatedData;
            }
            else
            {
                // Handle file upload error
            }

            return View("Index");
        }

        [HttpPost]
        public ActionResult UploadCSVFile(HttpPostedFileBase file)
        {
            if (file != null && file.ContentLength > 0)
            {
                var csvData = new List<string[]>();
                using (var reader = new StreamReader(file.InputStream))
                {
                    while (!reader.EndOfStream)
                    {
                        string line = reader.ReadLine();
                        string[] row = line.Split(',');

                        csvData.Add(row);
                    }
                }

                ViewBag.Content = csvData;
            }
            else
            {
                // Handle file upload error
            }

            return View("Index");
        }

        [HttpPost]
        public ActionResult DeleteFile(string fileName)
        {
            if (!string.IsNullOrEmpty(fileName))
            {
                string path = Server.MapPath("~/UploadedFiles/") + fileName;

                if (System.IO.File.Exists(path))
                {
                    System.IO.File.Delete(path);
                    ViewBag.Message = "File deleted successfully.";
                }
                else
                {
                    ViewBag.Message = "File not found.";
                }
            }
            else
            {
                // Handle missing fileName parameter
            }

            return View("Index");
        }

        [HttpPost]
        public ActionResult UpdateFile(HttpPostedFileBase file)
        {
            if (file != null && file.ContentLength > 0)
            {
                string fileContent = "";
                using (var reader = new StreamReader(file.InputStream))
                {
                    fileContent = reader.ReadToEnd();
                }

                string updatedData = fileContent.ToUpper();
                string newFileName = "M_TextFile.txt";
                string path = Server.MapPath("~/UploadedFiles/") + newFileName;

                if (System.IO.File.Exists(path))
                {
                    System.IO.File.WriteAllText(path, updatedData);
                    ViewBag.Content = updatedData;
                }
                else
                {
                    // File does not exist, handle accordingly
                }
            }
            else
            {
                // Handle file upload error
            }

            return View("Index");
        }
    }
}
```

File Upload Model

```
using System.ComponentModel.DataAnnotations;

using System.Web;



namespace WebApplication1.Models

{

    public class FileUploadModel

    {

        [Required(ErrorMessage = "Please select a file.")]

        [Display(Name = "File")]

        public HttpPostedFileBase File { get; set; }

    }

}
```

File Upload Index

```
@{
    ViewBag.Title = "File Operations";
}

<h2>Upload Text File</h2>

<form action="@Url.Action("UploadTextFile", "Web")" method="post" new { enctype="multipart/form-data" }>
    <input type="file" name="file" />
    <input type="submit" value="Upload" />
</form>

<h2>Upload CSV File</h2>
@using (Html.BeginForm("UploadCSVFile", "Web", FormMethod.Post, new { enctype = "multipart/form-data" }))
{
    <input type="file" name="file" />
    <input type="submit" value="Upload" />
}

<h2>Delete File</h2>
@using (Html.BeginForm("DeleteFile", "Web", FormMethod.Post))
{
    <input type="text" name="fileName" placeholder="File name" />
    <input type="submit" value="Delete" />
}

<h2>Update File</h2>
@using (Html.BeginForm("UpdateFile", "Web", FormMethod.Post, new { enctype = "multipart/form-data" }))
{
    <input type="file" name="file" />
    <input type="submit" value="Update" />
}

@if (!string.IsNullOrEmpty(ViewBag.Message))
{
    <p>@ViewBag.Message</p>
}
```

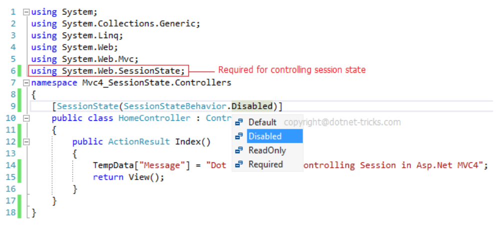
Session

Session Less Controller

If some of the controllers of your Asp.Net MVC application are not using session state features, you can disable session for those controllers and can gain slight performance improvement of your application. You can simplify session state for your application by using available options for session state.

In Asp.Net MVC, SessionState attribute provides you more control over the behavior of session-state by specifying the value of SessionStateBehavior enumeration as shown below:

Value	Description
Default	The default Asp.Net behavior is used to determine the session state behavior.
Disabled	Session state is disabled entirely.
ReadOnly	Read-only session state behavior is enabled.
Required	Full read-write session state behavior is enabled.



```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.Mvc;
using System.Web.SessionState;
namespace Mvc4_SessionState.Controllers
{
    [SessionState(SessionStateBehavior.Disabled)]
    public class HomeController : Contr
    {
        public ActionResult Index()
        {
            TempData["Message"] = "Dot                ontrolling Session in Asp.Net MVC4";
            return View();
        }
    }
}
```

When a user visits a website, a unique session is created for that user. The session is identified by a session ID, which is typically stored in a cookie or appended to the URL. The session ID is used to associate subsequent requests from the same user with their specific session data.

Sesion create

```
public class HomeController : Controller
{
    public ActionResult Index()
    {
        // Storing a value in session
        Session["Username"] = "JohnDoe";

        // Retrieving a value from session
        string username = (string)Session["Username"];

        // Checking if a session variable exists
        if (Session["Username"] != null)
        {
            // Session variable exists
        }
        else
        {
            // Session variable does not exist
        }

        return View();
    }
}

Or

public ActionResult Index()
{
    // Storing a value in session
    Session["Username"] = "JohnDoe";

    // Retrieving a value from session
    string username = (string)Session["Username"];

    return View();
}
```

//List

```
public ActionResult MyAction()
{
    // Create a new list
    List<string> myList = new List<string>();

    // Add items to the list
    myList.Add("Item 1");
    myList.Add("Item 2");
    myList.Add("Item 3");

    // Store the list in the session
    Session["MyList"] = myList;

    return View();
}


public ActionResult AnotherAction()
{
    // Retrieve the list from the session
    List<string> myList = Session["MyList"] as List<string>;
         //List<string> myList = (List<string>)Session["MyList"];

    // Do something with the list
```

```
        if (myList != null)
        {
            // Access the items in the list
            foreach (string item in myList)
            {
                // Process each item
                // ...
            }
        }

        return View();
    }
```

//Update

```
    // Retrieve the existing value from the session
    string oldValue = (string)Session["MyVariable"];

    // Update the value
    string newValue = "New Value";
    Session["MyVariable"] = newValue;
```

//Delete

```
    public ActionResult Logout()
    {
        // Abandon the session
        Session.Abandon();

        // Perform any additional logout logic

        return RedirectToAction("Index", "Home");
    }

    if (Session != null)
    {
        // Session is available
        // Perform session-related operations
    }
    else
    {
        // Session is not available
        // Handle the absence of session
    }
```

// Is Session Exists

```
    if (Session["MyVariable"] != null)
    {
        // Session variable is not null

        // Check if the session variable has a specific value
        if ((string)Session["MyVariable"] == "desiredValue")
        {
            // Session variable has the desired value
            // Perform actions accordingly
        }
        else
        {
            // Session variable exists but has a different value
            // Perform other actions
        }
    }
    else
    {
        // Session variable is null or doesn't exist
        // Handle the absence of session variable
    }
```

DOM / jQuery events

```
let table = new DataTable('#example');

table.on('click', 'tbody tr', function () {

  let data = table.row(this).data();

  alert('You clicked on ' + data[0] + "'s row");

});
```

DataTables events

```
function eventFired(type) {

  let n = document.querySelector('#demo_info');

  n.innerHTML +=

    '<div>' + type + ' event - ' + new Date().getTime() + '</div>';
  n.scrollTop = n.scrollHeight;
}
new DataTable('#example')
  .on('order.dt', () => eventFired('Order'))
  .on('search.dt', () => eventFired('Search'))
  .on('page.dt', () => eventFired('Page'));
```

Column rendering

```
new DataTable('#example', {
    columnDefs: [
        {
            // The `data` parameter refers to the data for the cell (defined by the
            // `data` option, which defaults to the column being worked with, in
            // this case `data: 0`.
            render: (data, type, row) => data + ' (' + row[3] + ')',
            targets: 0
        },
        { visible: false, targets: [3] }
    ]
});
```

Enter Key to Search

```
new DataTable('#example', {
    search: {
        return: true
    }
});
```

Page length options

```
new DataTable('#example', {
    lengthMenu: [
        [10, 25, 50, -1],
        [10, 25, 50, 'All']
    ]
});
```

Multiple table control elements

```
new DataTable('#example', {
    dom: '<"top"iflp<"clear">>rt<"bottom"iflp<"clear">>'
});
```

Complex headers with column visibility

```
new DataTable('#example', {
    columnDefs: [
        {
            targets: -1,
            visible: false
        }
```

```
   ]
})
```

Read HTML to data objects

```
new DataTable('#example', {
    columns: [
        { data: 'name' },
        { data: 'position' },
        { data: 'office' },
        { data: 'age' },
        { data: 'start_date' },
        { data: 'salary' }
    ]
});
```

HTML5 data-* attributes - cell data/HTML5 data-* attributes - table options

```
new DataTable('#example');
```

Setting defaults

```
Object.assign(DataTable.defaults, {
    searching: false,
    ordering: false
});
new DataTable('#example');
```

Row created callback

```
new DataTable('#example', {
    createdRow: (row, data, index) => {
        if (data[5].replace(/[\$,]/g, '') * 1 > 150000) {
            row.querySelector(':nth-child(6)').classList.add('highlight');
        }
    }
});
```

Custom toolbar elements

```
new DataTable('#example', {
    dom: '<"toolbar">frtip'
});
document.querySelector('div.toolbar').innerHTML = '<b>Custom tool bar! Text/images etc.</b>';
```

Order direction sequence control

```
new DataTable('#example', {
    columns: [
        null,
        null,
        { orderSequence: ['asc'] },
        { orderSequence: ['desc', 'asc', 'asc'] },
        { orderSequence: ['desc'] },
        null
    ]
});
```

```
@model IEnumerable<MyEcommerceAdmin.Models.Order>

@{
    ViewBag.Title = "Order";
    Layout = "~/Views/Shared/_Layout.cshtml";
}

<link rel="stylesheet" href="https://cdn.datatables.net/1.13.5/css/jquery.dataTables.min.css" />

<div class="row">
    <div class="col-md-12">
        <table id="example" class="table table-bordered table-hover table-striped">
            <thead>
                <tr>
                    <th>Order ID</th>
                    <th>Customer ID</th>
                    <th>Payment ID</th>
                    <th>Shipping ID</th>
                    <th>Discount</th>
                    <th>Taxes</th>
                    <th>Total Amount</th>
                    <th>Order Date</th>
                    <th>Action</th>
                </tr>
            </thead>
            <tbody>
                @foreach (var item in Model)
                {
                    <tr>
                        <td>@Html.DisplayFor(modelItem => item.OrderID)</td>
                        <td>@Html.DisplayFor(modelItem => item.Customer.CustomerID)</td>
                        <td>@Html.DisplayFor(modelItem => item.Payment.PaymentID)</td>
                        <td>@Html.DisplayFor(modelItem => item.ShippingDetail.ShippingID)</td>
                        <td>@Html.DisplayFor(modelItem => item.Discount)</td>
                        <td>@Html.DisplayFor(modelItem => item.Taxes)</td>
                        <td>@Html.DisplayFor(modelItem => item.TotalAmount)</td>
                        <td>@Html.DisplayFor(modelItem => item.OrderDate)</td>
                        <td>
                            @Html.ActionLink("View Details", "Details", new { id = item.OrderID }, new { @class = "btn btn-info" })
                        </td>
                    </tr>
                }
            </tbody>
        </table>
    </div>
</div>

@section Scripts {
    <script src="https://code.jquery.com/jquery-3.6.0.min.js"></script>
    <script src="https://cdn.datatables.net/1.13.5/js/jquery.dataTables.min.js"></script>
    <script>
        $(document).ready(function () {
            $('#example').DataTable();
        });
    </script>
}
```

Others

Connection

```
<connectionStrings>
    <add name="conn" connectionString="Data Source=AdvWebClass.mssql.somee.com;Initial Catalog=AdvWebClass;User
ID=hammadali562002_SQLLogin_1;Password=7ittcy6x85" providerName="System.Data.SqlClient" />
      </connectionStrings>
```

Session Time

```
    <system.web>
      <!-- Other configuration settings -->

      <sessionState mode="InProc" timeout="20" />
    </system.web>

    Enable-Migrations
    Add-Migration InitialCreate
    Update-Database

    INSERT INTO table_name (column1, column2, ...)
    VALUES (value1, value2, ...);

    DELETE FROM table_name
    WHERE condition;

    UPDATE table_name
    SET column1 = value1, column2 = value2, ...
    WHERE condition;

    DROP TABLE table_name;

    SELECT column1, column2, ...
    FROM table_name
    WHERE condition;

    <table>
      <tr>
        <td rowspan="2">Header 1</td>
        <td colspan="2">Header 2 and 3</td>
      </tr>
      <tr>
        <td>Data 1</td>
      </tr>
    </table>
```