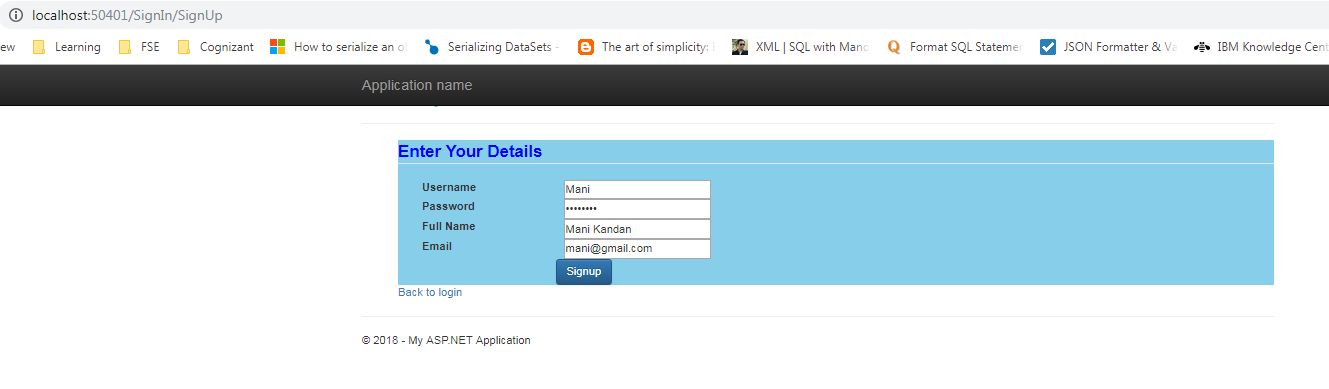
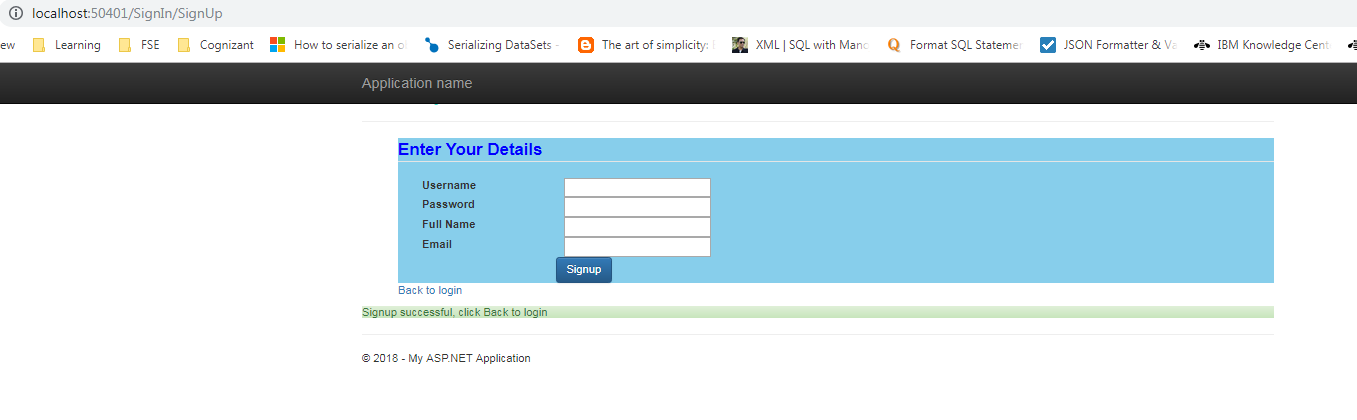
Assignment 25

Twitter with Additional Profile Validation

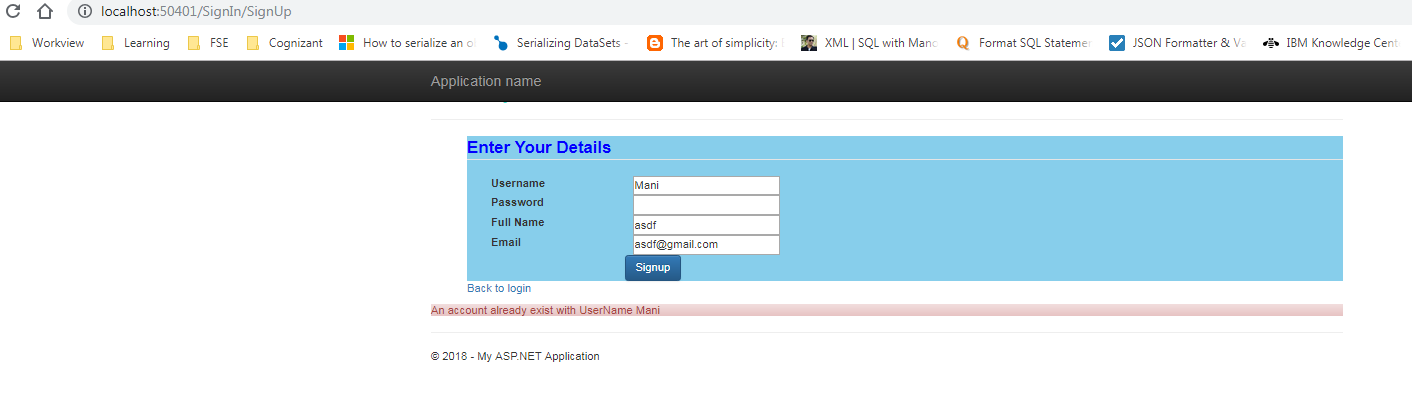
Sign up



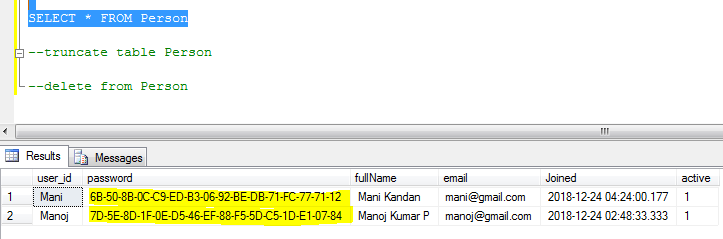
Successful Insertion:



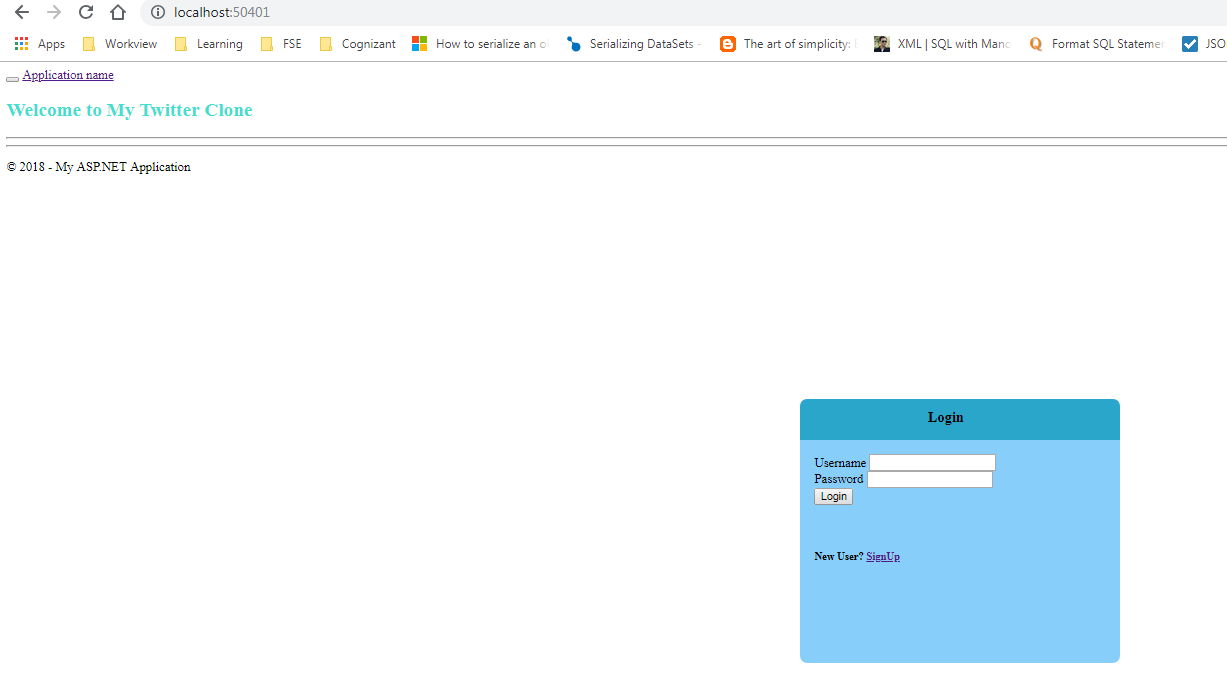
Existing User Insertion in SignUp



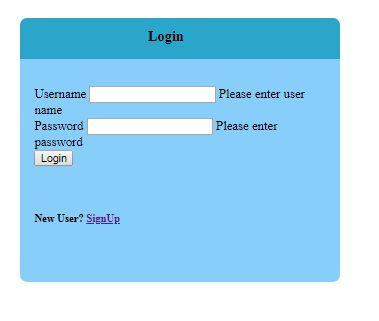
Password Encryption



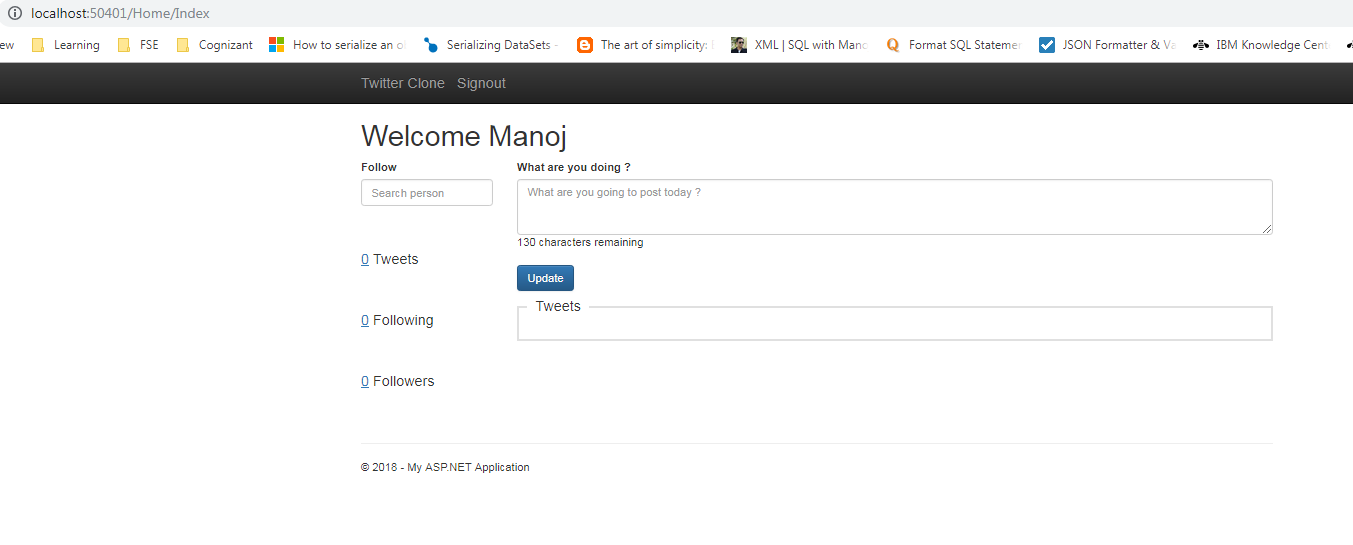
SignIN



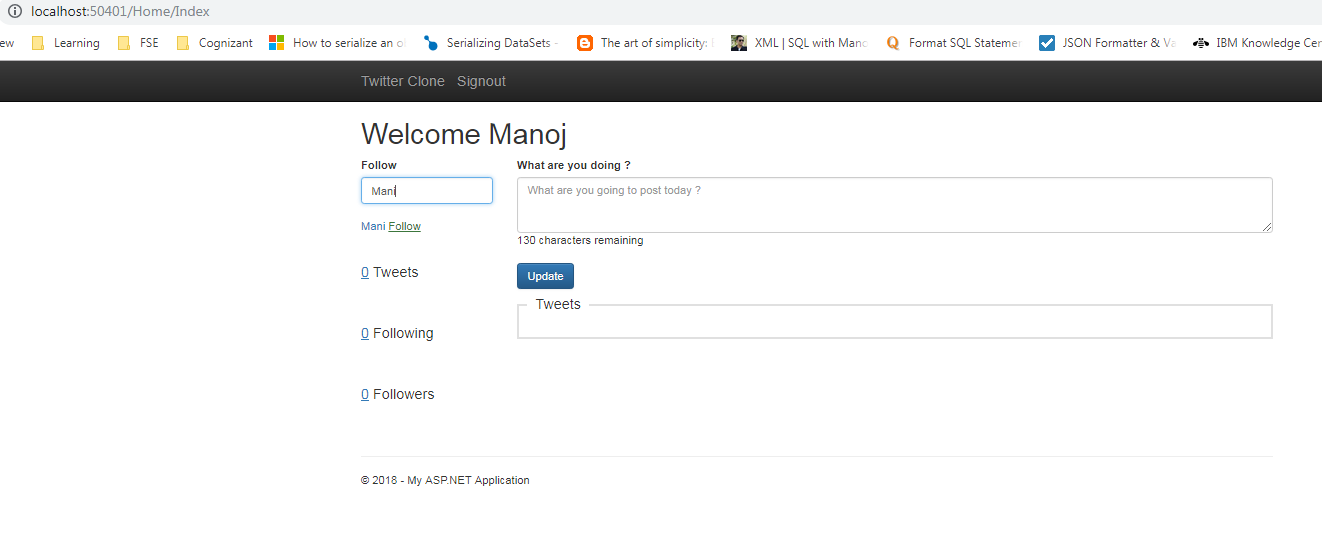
Empty User Validation



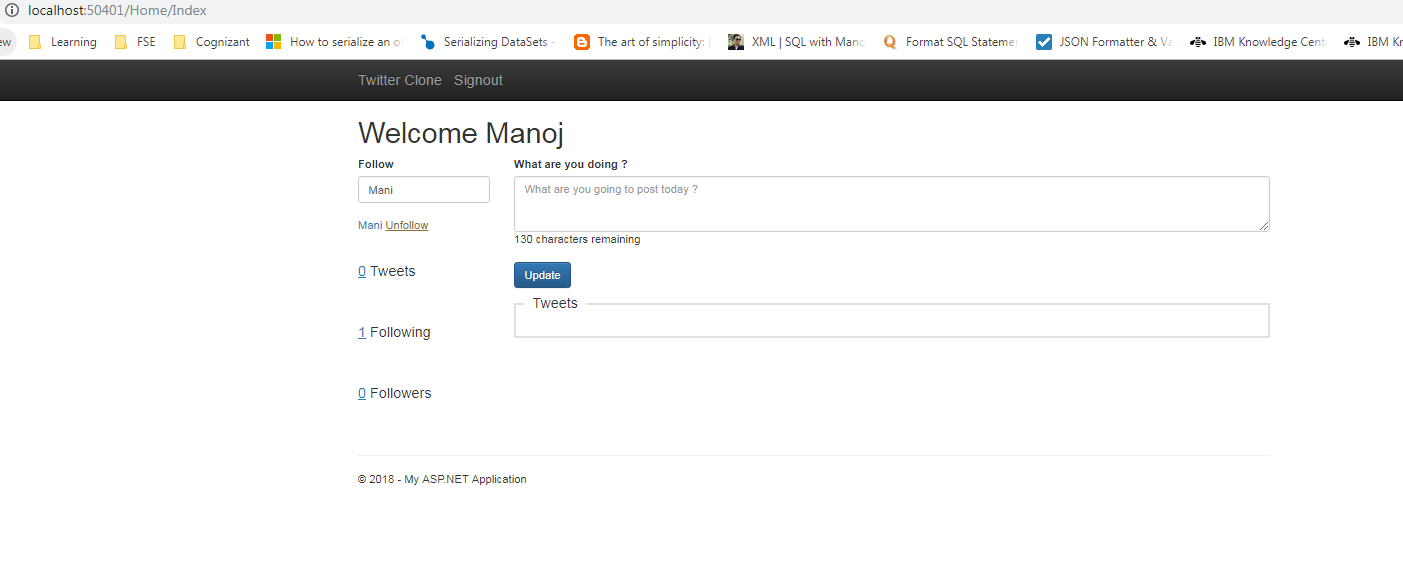
After Sign-in



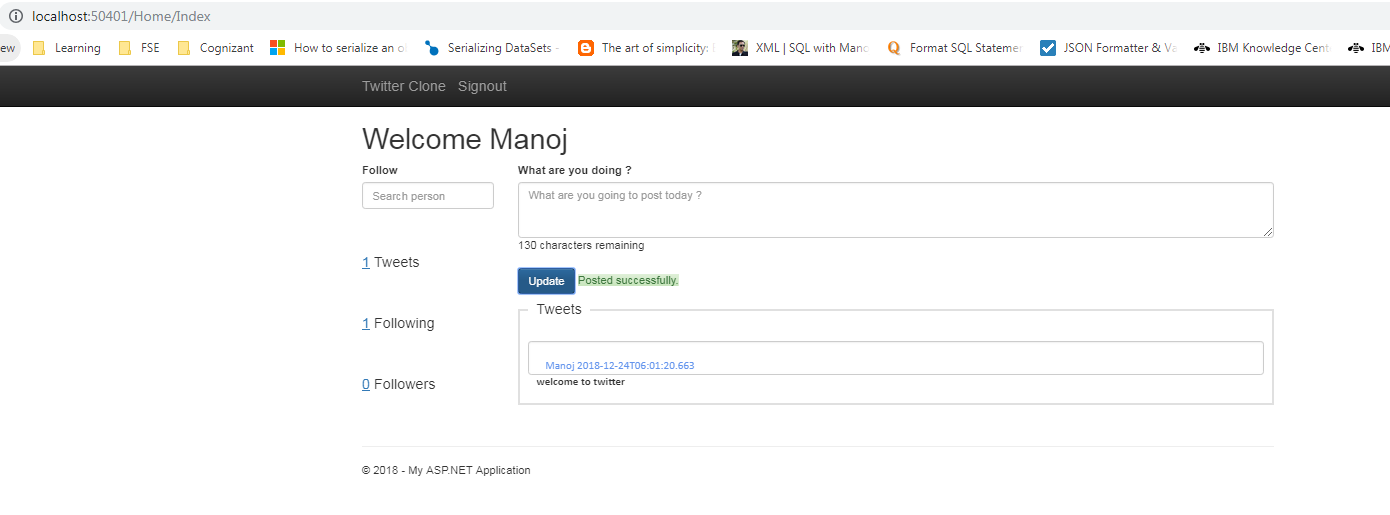
After Follow Search



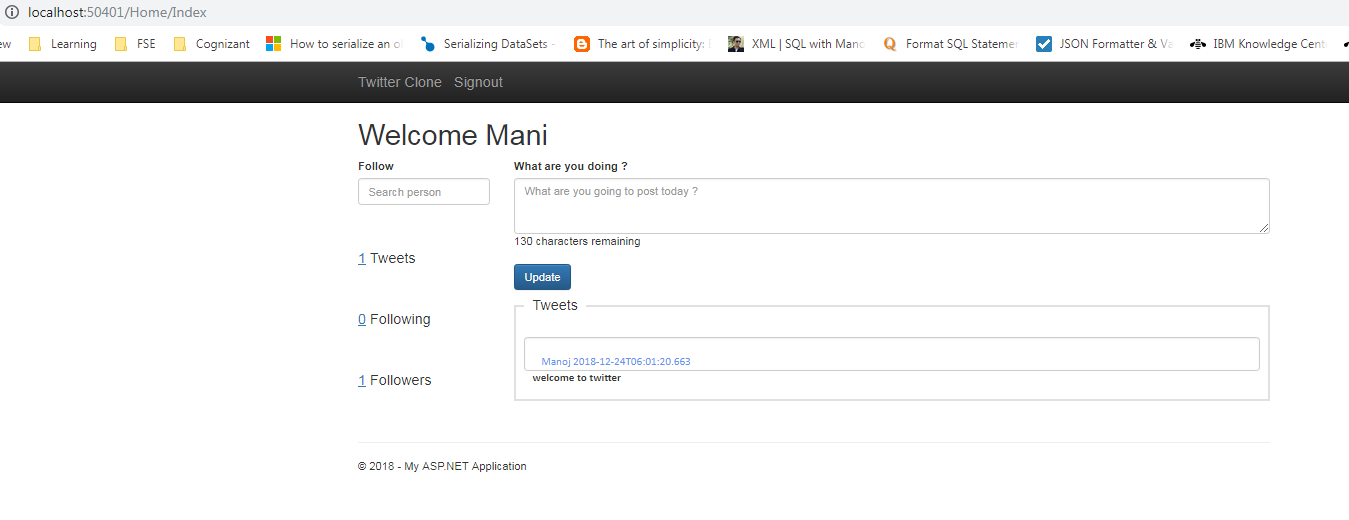
After Hitting Follow

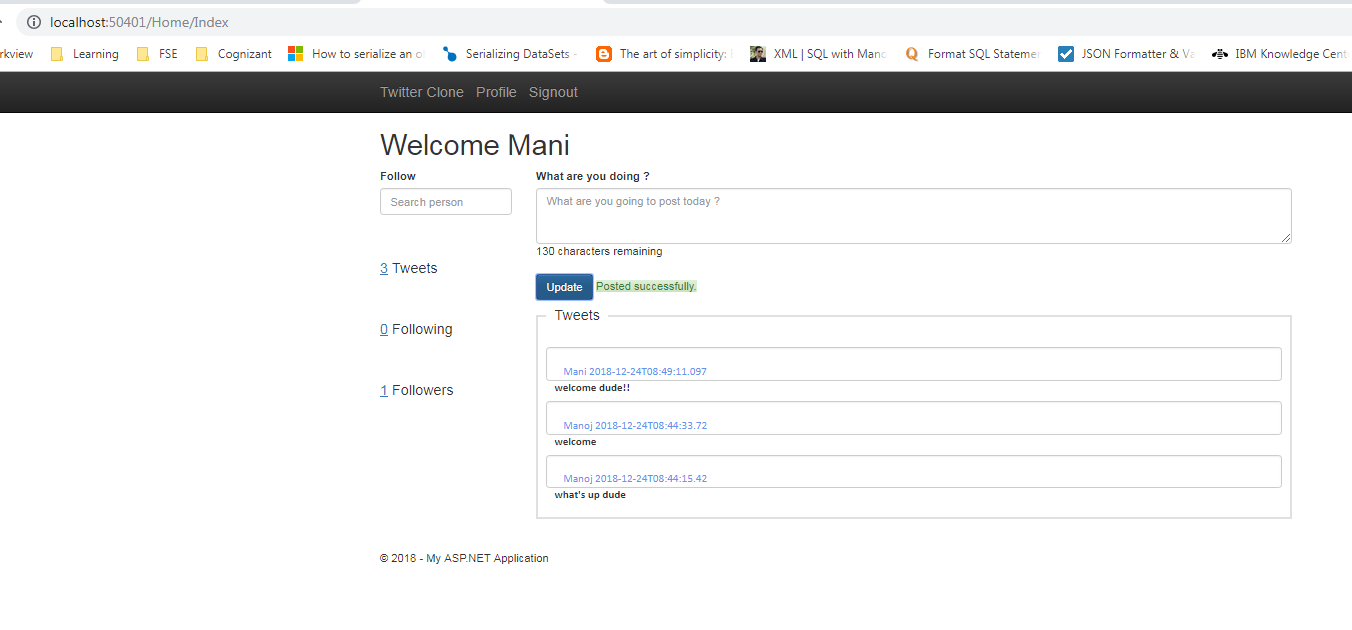


After first tweet

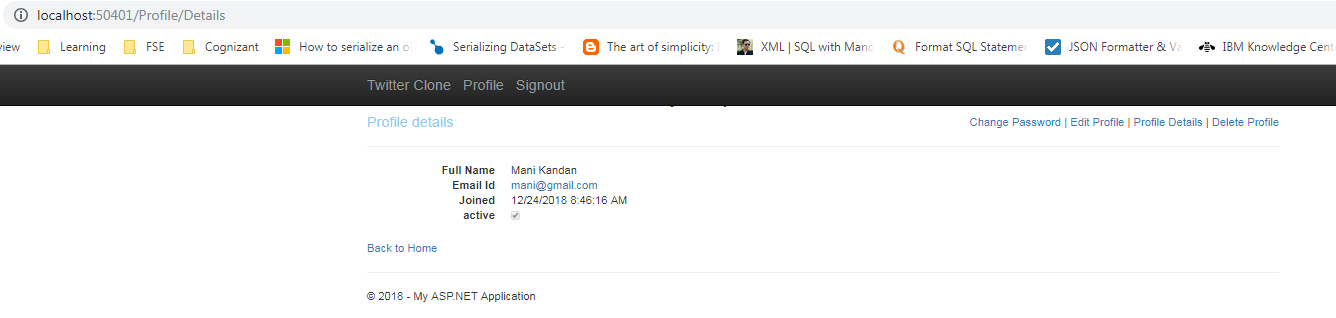


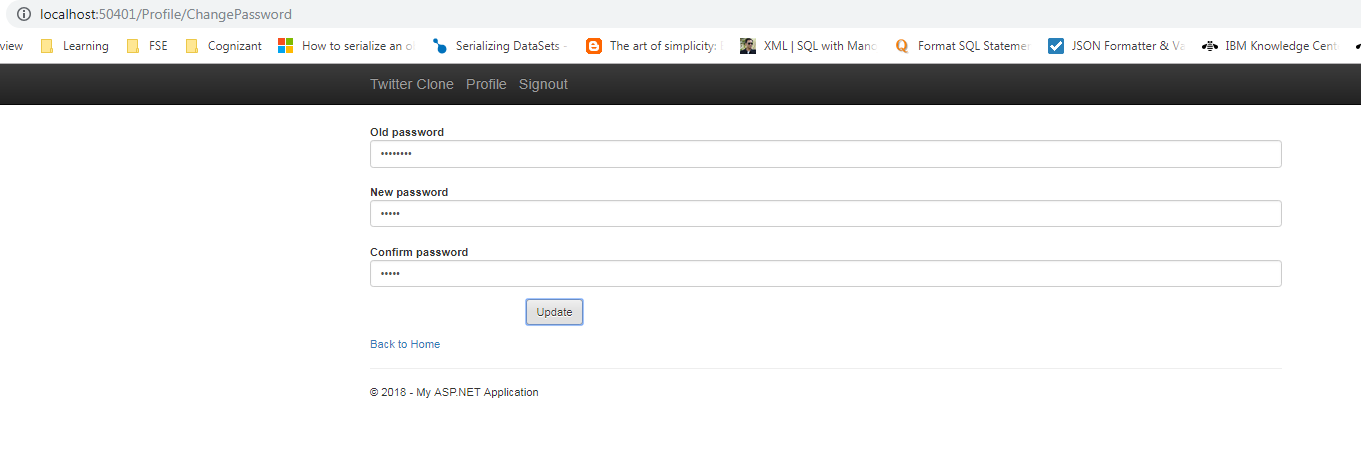
Details from other Login

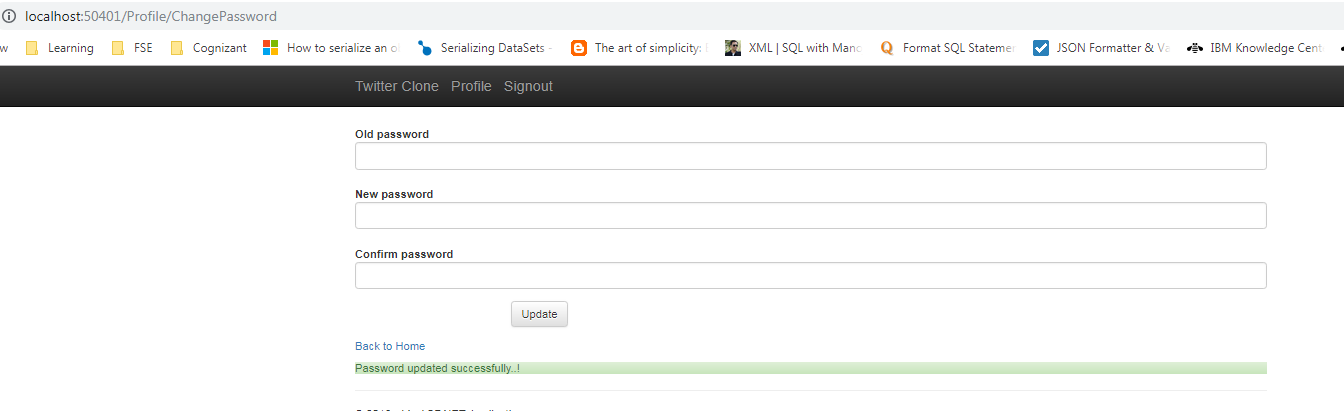


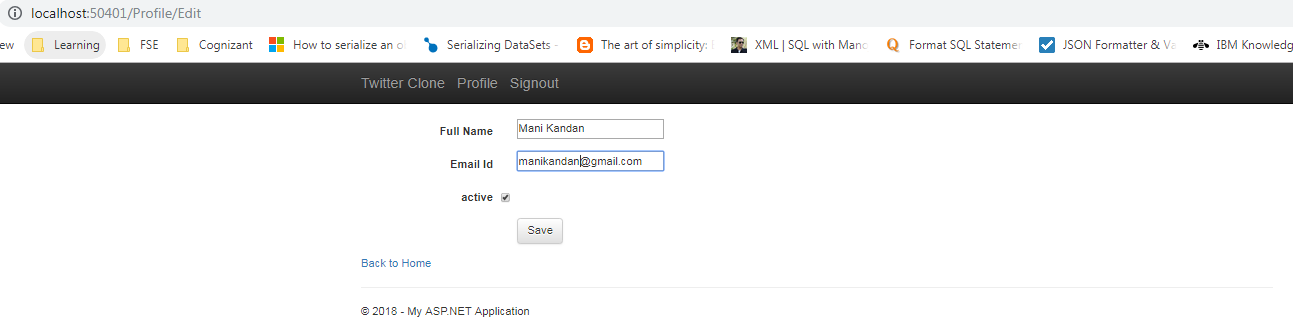


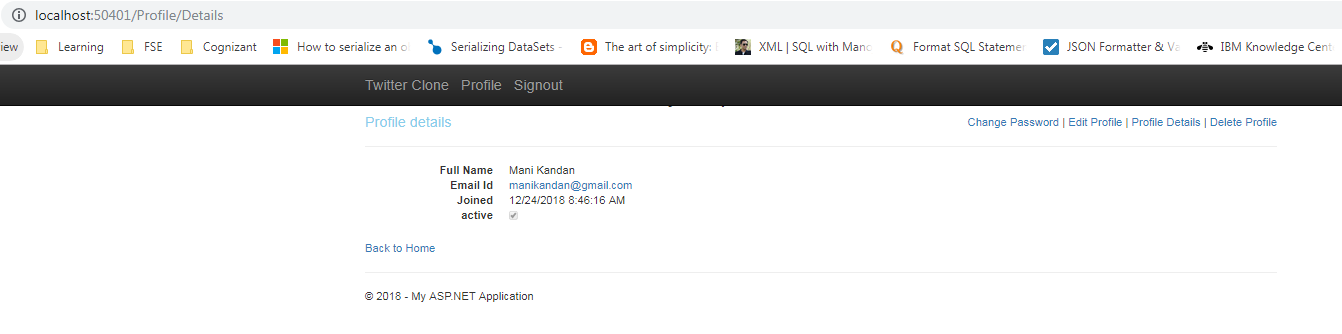
Edit Profile and dependent details.

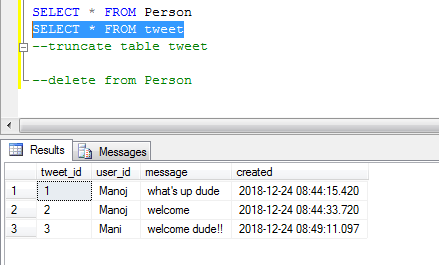












Code Snippet

SignInController.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Web.Mvc;

using System.Web.Security;

using Tweeter.Models;

namespace Tweeter.Controllers

{

public class SignInController : Controller

{

private TweeterEntities1 db = new TweeterEntities1();

// GET: SignIn

public ActionResult login()

{

return View();

}

// POST: SignIn/login

[HttpPost]

public ActionResult login(Account accDetails)

{

try

{

// TODO: Add login logic here

if (ModelState.IsValid)

{

Person person = db.People.Find(accDetails.userName);

if (person != null)

{

if (accDetails.userName == person.user\_id && Helper.EncodePasswordMd5(accDetails.passWord) == person.password)

{

FormsAuthentication.SetAuthCookie(person.user\_id, false);

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

}

else

{

ModelState.AddModelError(string.Empty, "Inalid password");

ViewBag.InValidlogin = "Incorrect credential..!";

return View();

}

}

else

{

ModelState.AddModelError(string.Empty, "Invalid user name");

ViewBag.InValidlogin = "Invalid user please signup today";

return View();

}

}

return View();

}

catch (Exception ex)

{

ModelState.AddModelError(string.Empty, ex.Message.ToString());

return View();

}

}

public ActionResult SignUp()

{

return View();

}

// POST: SignIn/SignUp

[HttpPost]

[ValidateAntiForgeryToken]

public ActionResult SignUp([Bind(Include = "user\_id,password,fullName,email,joined,active")]Person personDetails)

{

try

{

// TODO: Add insert logic here

if (ModelState.IsValid)

{

try

{

Person person = db.People.Find(personDetails.user\_id);

if (person == null)

{

personDetails.Joined = DateTime.Now;

personDetails.active = true;

personDetails.password = Helper.EncodePasswordMd5(personDetails.password);

db.People.Add(personDetails);

db.SaveChanges();

ViewBag.SignupSuccess = "True";

ModelState.Clear();

return View();

}

else

{

ViewBag.SuccessMessage = "An account already exist with UserName " + personDetails.user\_id;

return View();

}

}

catch (Exception exec)

{

ViewBag.errorMessage = "Oops there is an error:- " + exec.Message;

return View();

}

}

return View();

}

catch

{

return View("login");

}

}

[OutputCache(Duration = 0, NoStore = true)]

public ActionResult signout()

{

FormsAuthentication.SignOut();

return View("login");

}

}

}

ProfileController.cs

using System;

using System.Collections.Generic;

using System.Data.Entity;

using System.Linq;

using System.Net;

using System.Web;

using System.Web.Mvc;

using Tweeter.Filters;

using Tweeter.Models;

namespace Tweeter.Controllers

{

[CustomExceptionHelperFilter]

public class ProfileController : Controller

{

private TweeterEntities1 db = new TweeterEntities1();

// GET: People/Details/5

[Authorize]

[OutputCache(Duration = 0, NoStore = true)]

public ActionResult Details()

{

if (User.Identity.Name == null)

{

return new HttpStatusCodeResult(HttpStatusCode.BadRequest);

}

Person person = db.People.Find(User.Identity.Name);

if (person == null)

{

return HttpNotFound();

}

return View(person);

}

// GET: People/Edit/5

[Authorize]

[OutputCache(Duration = 0, NoStore = true)]

public ActionResult Edit()

{

if (User.Identity.Name == null)

{

return new HttpStatusCodeResult(HttpStatusCode.BadRequest);

}

Person person = db.People.Find(User.Identity.Name);

if (person == null)

{

return HttpNotFound();

}

EditProfile editprof = new EditProfile();

editprof.fullName = person.fullName;

editprof.emailID = person.email;

editprof.active = person.active;

return View(editprof);

}

[HttpPost]

[ValidateAntiForgeryToken]

[Authorize]

[OutputCache(Duration = 0, NoStore = true)]

public ActionResult Edit(EditProfile editProf)

{

if (ModelState.IsValid)

{

Person persondetails = db.People.Find(User.Identity.Name);

if (persondetails != null)

{

persondetails.fullName = editProf.fullName;

persondetails.email = editProf.emailID;

persondetails.active = editProf.active;

db.Entry(persondetails).State = EntityState.Modified;

db.SaveChanges();

return RedirectToAction("Details");

}

}

return View(editProf);

}

[Authorize]

[OutputCache(Duration = 0, NoStore = true)]

public ActionResult UpdatePassword()

{

if (User.Identity.Name == null)

{

return new HttpStatusCodeResult(HttpStatusCode.BadRequest);

}

Person person = db.People.Find(User.Identity.Name);

if (person == null)

{

return HttpNotFound();

}

return View();

}

[HttpPost]

[ValidateAntiForgeryToken]

[Authorize]

[OutputCache(Duration = 0, NoStore = true)]

public ActionResult UpdatePassword(UpdatePassword updpass)

{

if (ModelState.IsValid)

{

Person person = db.People.Find(User.Identity.Name);

if (Helper.EncodePasswordMd5(updpass.oldpassWord) == person.password)

{

if (Helper.EncodePasswordMd5(updpass.NewpassWord) == Helper.EncodePasswordMd5(updpass.ConfirmpassWord))

{

person.password = Helper.EncodePasswordMd5(updpass.ConfirmpassWord);

db.Entry(person).State = EntityState.Modified;

db.SaveChanges();

ViewBag.PasswordUpdateSuccess = true;

}

else

{

ModelState.AddModelError(string.Empty, "New password not matching with confirm password..!");

return View();

}

}

else

{

ModelState.AddModelError(string.Empty, "Invalid old password..!");

return View();

}

}

return View();

}

// GET: People/Delete/5

[Authorize]

[OutputCache(Duration = 0, NoStore = true)]

public ActionResult Delete()

{

if (User.Identity.Name == null)

{

return new HttpStatusCodeResult(HttpStatusCode.BadRequest);

}

Person person = db.People.Find(User.Identity.Name);

if (person == null)

{

return HttpNotFound();

}

return View(person);

}

// POST: People/Delete/5

[HttpPost, ActionName("Delete")]

[ValidateAntiForgeryToken]

[Authorize]

[OutputCache(Duration = 0, NoStore = true)]

public ActionResult DeleteConfirmed()

{

var twt = db.tweets.Where(x => x.user\_id == User.Identity.Name);

db.tweets.RemoveRange(twt);

db.SaveChanges();

Person person = db.People.Find(User.Identity.Name);

db.People.Remove(person);

db.SaveChanges();

System.Web.Security.FormsAuthentication.SignOut();

return RedirectToAction("login", "SignIn");

}

protected override void Dispose(bool disposing)

{

if (disposing)

{

db.Dispose();

}

base.Dispose(disposing);

}

}

}

HomeController.cs

using Newtonsoft.Json;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Web.Mvc;

using Tweeter.Filters;

using Tweeter.Models;

namespace Tweeter.Controllers

{

[CustomExceptionHelperFilter]

public class HomeController : Controller

{

private TweeterEntities1 db = new TweeterEntities1();

// GET: Home

[Authorize]

[OutputCache(Duration = 0, NoStore = true)]

public ActionResult Index()

{

Person person = db.People.Find(User.Identity.Name);

return View(person);

}

[Authorize]

[OutputCache(Duration = 0, NoStore = true)]

public JsonResult SearchPerson(string PersionUserId)

{

string person\_Id = string.Empty;

var Srhperson = db.People.Find(PersionUserId);

if (Srhperson != null)

person\_Id = Srhperson.user\_id;

var FollowingChk = db.followings.Where(x => x.user\_id == User.Identity.Name && x.following\_id == PersionUserId).Select(x => x.user\_id).Count();

Dictionary<string, string> dt = new Dictionary<string, string>();

dt.Add("PersionId", person\_Id);

dt.Add("FollowCheck", FollowingChk.ToString());

var jsonobject = JsonConvert.SerializeObject(dt);

return Json(jsonobject, JsonRequestBehavior.AllowGet);

}

[HttpPost]

[Authorize]

[OutputCache(Duration = 0, NoStore = true)]

public ActionResult PostTwitter(string messagePost)

{

tweet twts = new tweet();

twts.message = messagePost;

twts.created = DateTime.Now;

twts.user\_id = User.Identity.Name;

db.tweets.Add(twts);

db.SaveChanges();

string message = "SUCCESS";

return Json(new { Message = message, JsonRequestBehavior.AllowGet });

}

[HttpPost]

[Authorize]

[OutputCache(Duration = 0, NoStore = true)]

public ActionResult FollowPerson(string PersionUserId)

{

following flw = new following();

flw.user\_id = User.Identity.Name;

flw.following\_id = PersionUserId;

object message = "";

var FollowingChk = db.followings.Where(x => x.user\_id == User.Identity.Name && x.following\_id == PersionUserId).Select(x => x.user\_id).Count();

if (FollowingChk == 0)

{

db.followings.Add(flw);

message = "SUCCESS";

}

else

{

following FOLLOWINGperson = db.followings.Where(x => x.user\_id == flw.user\_id && x.following\_id == flw.following\_id).FirstOrDefault();

db.followings.Remove(FOLLOWINGperson);

message = "DeleteSUCCESS";

}

db.SaveChanges();

var jsonobject = JsonConvert.SerializeObject(message);

return Json(jsonobject, JsonRequestBehavior.AllowGet);

}

[Authorize]

[OutputCache(Duration = 0, NoStore = true)]

public JsonResult GetTwitter()

{

var twt = db.tweets.OrderByDescending(cr => cr.created).Select(x => new { user\_id = x.user\_id, messaage = x.message, created = x.created }).ToList();

var jsonobject = JsonConvert.SerializeObject(twt);

return Json(jsonobject, JsonRequestBehavior.AllowGet);

}

[Authorize]

[OutputCache(Duration = 0, NoStore = true)]

public JsonResult GetTwittecount()

{

var twt = db.tweets.Select(x => x.tweet\_id).Count();

var jsonobject = JsonConvert.SerializeObject(twt);

return Json(jsonobject, JsonRequestBehavior.AllowGet);

}

[Authorize]

[OutputCache(Duration = 0, NoStore = true)]

public JsonResult GetFollowingFollowercount()

{

var FollowingCnt = db.followings.Where(x => x.user\_id == User.Identity.Name).Select(x => x.user\_id).Count();

var FollowerCnt = db.followings.Where(x => x.following\_id == User.Identity.Name).Select(x => x.following\_id).Count();

Dictionary<string, string> cnt = new Dictionary<string, string>();

cnt.Add("FollowingCnt", FollowingCnt.ToString());

cnt.Add("FollowerCnt", FollowerCnt.ToString());

var jsonobject = JsonConvert.SerializeObject(cnt);

return Json(jsonobject, JsonRequestBehavior.AllowGet);

}

}

}

Models

Account.cs

using System;

using System.Collections.Generic;

using System.ComponentModel.DataAnnotations;

using System.Linq;

using System.Web;

namespace Tweeter.Models

{

public class Account

{

[Display(Name = "Username")]

[Required(ErrorMessage = "Please enter user name")]

[MaxLength(length: 25, ErrorMessage = "Maximum length 25 exceeded")]

public string userName { get; set; }

[Display(Name = "Password")]

[Required(ErrorMessage = "Please enter password")]

[MaxLength(length: 100, ErrorMessage = "Maximum length 100 exceeded")]

public string passWord { get; set; }

}

}

EditProfile.cs

using System;

using System.Collections.Generic;

using System.ComponentModel.DataAnnotations;

using System.Linq;

using System.Web;

namespace Tweeter.Models

{

public class EditProfile

{

[Display(Name = "Full Name")]

[Required(ErrorMessage = "Please enter full name")]

[MaxLength(length: 30, ErrorMessage = "Maximum length 30 exceeded")]

public string fullName { get; set; }

[Display(Name = "Email Id")]

[Required(ErrorMessage = "Please enter email Id")]

[EmailAddress]

[MaxLength(length: 50, ErrorMessage = "Maximum length 50 exceeded")]

public string emailID { get; set; }

public bool active { get; set; }

}

}

Helper.cs

using System;

using System.Collections.Generic;

using System.Linq;

using System.Security.Cryptography;

using System.Text;

using System.Web;

namespace Tweeter.Models

{

public static class Helper

{

public static string ToAbsoluteUrl(this string relativeUrl) //Use absolute URL instead of adding phycal path for CSS, JS and Images

{

if (string.IsNullOrEmpty(relativeUrl)) return relativeUrl;

if (HttpContext.Current == null) return relativeUrl;

if (relativeUrl.StartsWith("/")) relativeUrl = relativeUrl.Insert(0, "~");

if (!relativeUrl.StartsWith("~/")) relativeUrl = relativeUrl.Insert(0, "~/");

var url = HttpContext.Current.Request.Url;

var port = url.Port != 80 ? (":" + url.Port) : String.Empty;

return String.Format("{0}://{1}{2}{3}", url.Scheme, url.Host, port, VirtualPathUtility.ToAbsolute(relativeUrl));

}

public static string GeneratePassword(int length) //length of salt

{

const string allowedChars = "abcdefghijkmnopqrstuvwxyzABCDEFGHJKLMNOPQRSTUVWXYZ0123456789";

var randNum = new Random();

var chars = new char[length];

var allowedCharCount = allowedChars.Length;

for (var i = 0; i <= length - 1; i++)

{

chars[i] = allowedChars[Convert.ToInt32((allowedChars.Length) \* randNum.NextDouble())];

}

return new string(chars);

}

public static string EncodePassword(string pass, string salt) //encrypt password

{

byte[] bytes = Encoding.Unicode.GetBytes(pass);

byte[] src = Encoding.Unicode.GetBytes(salt);

byte[] dst = new byte[src.Length + bytes.Length];

System.Buffer.BlockCopy(src, 0, dst, 0, src.Length);

System.Buffer.BlockCopy(bytes, 0, dst, src.Length, bytes.Length);

HashAlgorithm algorithm = HashAlgorithm.Create("SHA1");

byte[] inArray = algorithm.ComputeHash(dst);

//return Convert.ToBase64String(inArray);

return EncodePasswordMd5(Convert.ToBase64String(inArray));

}

public static string EncodePasswordMd5(string pass) //Encrypt using MD5

{

Byte[] originalBytes;

Byte[] encodedBytes;

MD5 md5;

//Instantiate MD5CryptoServiceProvider, get bytes for original password and compute hash (encoded password)

md5 = new MD5CryptoServiceProvider();

originalBytes = ASCIIEncoding.Default.GetBytes(pass);

encodedBytes = md5.ComputeHash(originalBytes);

//Convert encoded bytes back to a 'readable' string

return BitConverter.ToString(encodedBytes);

}

public static string base64Encode(string sData) // Encode

{

try

{

byte[] encData\_byte = new byte[sData.Length];

encData\_byte = System.Text.Encoding.UTF8.GetBytes(sData);

string encodedData = Convert.ToBase64String(encData\_byte);

return encodedData;

}

catch (Exception ex)

{

throw new Exception("Error in base64Encode" + ex.Message);

}

}

public static string base64Decode(string sData) //Decode

{

try

{

var encoder = new System.Text.UTF8Encoding();

System.Text.Decoder utf8Decode = encoder.GetDecoder();

byte[] todecodeByte = Convert.FromBase64String(sData);

int charCount = utf8Decode.GetCharCount(todecodeByte, 0, todecodeByte.Length);

char[] decodedChar = new char[charCount];

utf8Decode.GetChars(todecodeByte, 0, todecodeByte.Length, decodedChar, 0);

string result = new String(decodedChar);

return result;

}

catch (Exception ex)

{

throw new Exception("Error in base64Decode" + ex.Message);

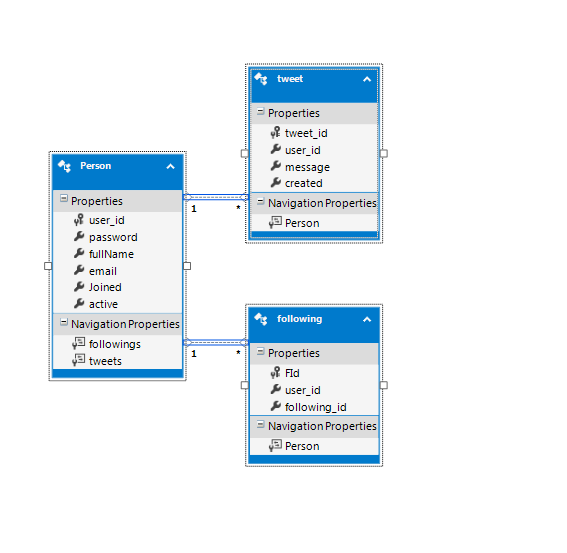
}

}

}

}

TwitterEntityModel



//------------------------------------------------------------------------------

// <auto-generated>

// This code was generated from a template.

//

// Manual changes to this file may cause unexpected behavior in your application.

// Manual changes to this file will be overwritten if the code is regenerated.

// </auto-generated>

//------------------------------------------------------------------------------

namespace Tweeter.Models

{

using System;

using System.Data.Entity;

using System.Data.Entity.Infrastructure;

public partial class TweeterEntities1 : DbContext

{

public TweeterEntities1()

: base("name=TweeterEntities1")

{

}

protected override void OnModelCreating(DbModelBuilder modelBuilder)

{

throw new UnintentionalCodeFirstException();

}

public virtual DbSet<following> followings { get; set; }

public virtual DbSet<Person> People { get; set; }

public virtual DbSet<tweet> tweets { get; set; }

}

}

Following.cs

//------------------------------------------------------------------------------

// <auto-generated>

// This code was generated from a template.

//

// Manual changes to this file may cause unexpected behavior in your application.

// Manual changes to this file will be overwritten if the code is regenerated.

// </auto-generated>

//------------------------------------------------------------------------------

namespace Tweeter.Models

{

using System;

using System.Collections.Generic;

public partial class following

{

public int FId { get; set; }

public string user\_id { get; set; }

public string following\_id { get; set; }

public virtual Person Person { get; set; }

}

}

Person.cs

//------------------------------------------------------------------------------

// <auto-generated>

// This code was generated from a template.

//

// Manual changes to this file may cause unexpected behavior in your application.

// Manual changes to this file will be overwritten if the code is regenerated.

// </auto-generated>

//------------------------------------------------------------------------------

namespace Tweeter.Models

{

using System;

using System.Collections.Generic;

using System.ComponentModel.DataAnnotations;

public partial class Person

{

public Person()

{

this.followings = new HashSet<following>();

this.tweets = new HashSet<tweet>();

}

[Display(Name = "Username")]

[Required(ErrorMessage = "Please enter user name")]

[MaxLength(length: 25, ErrorMessage = "Maximum length 25 exceeded")]

public string user\_id { get; set; }

[Display(Name = "Password")]

[Required(ErrorMessage = "Please enter password")]

[MaxLength(length: 100, ErrorMessage = "Maximum length 100 exceeded")]

public string password { get; set; }

[Display(Name = "Full Name")]

[Required(ErrorMessage = "Please enter full name")]

[MaxLength(length: 30, ErrorMessage = "Maximum length 30 exceeded")]

public string fullName { get; set; }

[Display(Name = "Email")]

[Required(ErrorMessage = "Please enter email id")]

[MaxLength(length: 50, ErrorMessage = "Maximum length 50 exceeded")]

[EmailAddress]

public string email { get; set; }

public System.DateTime Joined { get; set; }

public bool active { get; set; }

public virtual ICollection<following> followings { get; set; }

public virtual ICollection<tweet> tweets { get; set; }

}

}

Tweet

//------------------------------------------------------------------------------

// <auto-generated>

// This code was generated from a template.

//

// Manual changes to this file may cause unexpected behavior in your application.

// Manual changes to this file will be overwritten if the code is regenerated.

// </auto-generated>

//------------------------------------------------------------------------------

namespace Tweeter.Models

{

using System;

using System.Collections.Generic;

public partial class tweet

{

public int tweet\_id { get; set; }

public string user\_id { get; set; }

public string message { get; set; }

public System.DateTime created { get; set; }

public virtual Person Person { get; set; }

}

}

UpdatePassword

using System;

using System.Collections.Generic;

using System.ComponentModel.DataAnnotations;

using System.Linq;

using System.Web;

namespace Tweeter.Models

{

public class UpdatePassword

{

[Display(Name = "Old password")]

[Required(ErrorMessage = "Please enter old password")]

[MaxLength(length: 100, ErrorMessage = "Maximum length 100 exceeded")]

public string oldpassWord { get; set; }

[Display(Name = "New password")]

[Required(ErrorMessage = "Please enter new password")]

[MaxLength(length: 100, ErrorMessage = "Maximum length 100 exceeded")]

public string NewpassWord { get; set; }

[Display(Name = "Confirm password")]

[Required(ErrorMessage = "Please confirm password")]

[MaxLength(length: 100, ErrorMessage = "Maximum length 100 exceeded")]

public string ConfirmpassWord { get; set; }

}

}

Home Index.html

@model Tweeter.Models.Person

@{

if (ViewContext.IsChildAction != true && !Request.IsAjaxRequest())

{

Layout = "~/Views/Shared/\_Layout.cshtml";

}

else

{

Layout = null;

}

}

@{

ViewBag.Title = "Index";

}

<head>

<title>ViewBag.Title</title>

<link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css" integrity="sha384-BVYiiSIFeK1dGmJRAkycuHAHRg32OmUcww7on3RYdg4Va+PmSTsz/K68vbdEjh4u" crossorigin="anonymous">

<!-- Optional theme -->

<link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap-theme.min.css" integrity="sha384-rHyoN1iRsVXV4nD0JutlnGaslCJuC7uwjduW9SVrLvRYooPp2bWYgmgJQIXwl/Sp" crossorigin="anonymous">

<!-- Latest compiled and minified JavaScript -->

<script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/js/bootstrap.min.js" integrity="sha384-Tc5IQib027qvyjSMfHjOMaLkfuWVxZxUPnCJA7l2mCWNIpG9mGCD8wGNIcPD7Txa" crossorigin="anonymous"></script>

<script src="~/Scripts/jquery-1.10.2.min.js"></script>

<script type="text/javascript" language="javascript">

var urlContent = "@Url.Content("~/")";

</script>

<style>

legend {

border-style: none;

border-width: 0;

font-size: 14px;

line-height: 20px;

margin-bottom: 0;

width: auto;

padding: 0 10px;

}

fieldset {

border: 3px solid #e0e0e0;

padding: 10px;

}

.spanUserID {

padding: 20px;

font-family: Calibri;

color: cornflowerblue;

}

.spanMessage {

font-family: Calibri;

margin: 0;

padding-left: 10px;

word-wrap: break-word;

}

input, select, textarea {

max-width: none;

}

.ScrollStyle {

max-height: 330px;

overflow-y: scroll;

}

.dipBlock {

display: block !important;

}

.dipNone {

display: none !important;

}

</style>

</head>

<div class="row">

<div class="col-md-12">

<h2>Twitter Clone</h2>

<h1>Welcome @Model.user\_id</h1>

</div>

</div>

<div class="row">

<div class="col-md-2">

@Html.Label("Follow", new { @class = "control-label " })

@Html.TextBox("txtSearchPerson", null, new { placeholder = "Search person", @maxlength = 25, @class = "form-control" })

<br />

<div id="followdiv" style="display:none;">

<span id="spnSrhUser" class="text-primary"></span> <u><a id="btnfollowUnfollow" value="Follow" /></u>

</div>

<h6 id="shrMsg"></h6>

<br />

<h4><a id="twtCnt" href="#"><u></u></a> Tweets</h4>

<br /><br />

<h4><a id="FollwingCnt" href="#"><u></u></a> Following</h4>

<br /><br />

<h4><a id="FollwertCnt" href="#"><u></u></a> Followers</h4>

<br /><br />

</div>

<div class="col-md-10">

@Html.Label("What are you doing ?", new { @class = "control-label " })

@Html.TextArea("MessageUpd", null, new { @class = "form-control", @rows = 3, maxlength = 130, placeholder = "What are you going to post today ?" })

<div id="textarea\_feedback"></div>

<br />

<input type="button" id="btnUpdateMsg" value="Update" class="btn btn-primary" /> <span id="spnSuccess"></span>

<br />

<fieldset>

<legend>

<h4>Tweets</h4>

</legend>

<div id="tweetsDetails">

@\*class="ScrollStyle"\*@

</div>

</fieldset>

</div>

</div>

<script>

$(document).ready(function () {

LoadData();

LoadFollwingAndFollwerCount();

var text\_max = 130;

$('#textarea\_feedback').html(text\_max + ' characters remaining');

$('#MessageUpd').keyup(function () {

var text\_length = $('#MessageUpd').val().length;

var text\_remaining = text\_max - text\_length;

$('#textarea\_feedback').html(text\_remaining + ' characters remaining');

});

$('#txtSearchPerson').keyup(function () {

var SrhPersontxt = $("#txtSearchPerson").val();

$.ajax({

url: "/Home/SearchPerson",

method: "POST",

data: '{ "PersionUserId": "' + SrhPersontxt + '"}',

dataType: 'json',

contentType: "application/json; charset=utf-8",

success: function (datas) {

datas = JSON.parse(datas);

if (datas.PersionId != "") {

$('#shrMsg').empty();

$('#followdiv').show();//.addClass("dipBlock");

$("#spnSrhUser").text(datas.PersionId);

if (datas.FollowCheck == "0") {

$("#followdiv").find('#btnfollowUnfollow').addClass("text-success");

$("#followdiv").find('#btnfollowUnfollow').text("Follow").css('cursor', 'pointer');

}

else {

$("#followdiv").find('#btnfollowUnfollow').addClass("text-warning");

$("#followdiv").find('#btnfollowUnfollow').text("Unfollow").css('cursor', 'pointer');;

}

}

else {

$('#shrMsg').removeClass("dipNone");

$('#shrMsg').addClass("dipBlock alert-danger");

$("#shrMsg").text("search not found");

}

},

error: function (ex) {

var r = jQuery.parseJSON(response.responseText);

alert("Message: " + r.Message);

alert("StackTrace: " + r.StackTrace);

alert("ExceptionType: " + r.ExceptionType);

}

});

return false;

});

$("#btnfollowUnfollow").click(function () {

var FollowPerson = $("#followdiv").find('#spnSrhUser').html();

$.ajax({

url: "/Home/FollowPerson",

method: "POST",

data: '{ "PersionUserId": "' + FollowPerson + '"}',

dataType: "json",

contentType: "application/json; charset=utf-8",

success: function (Message) {

Message = JSON.parse(Message);

if (Message == "SUCCESS") {

$("#followdiv").find('#btnfollowUnfollow').addClass("text-warning");

$("#followdiv").find("#btnfollowUnfollow").text("Unfollow");

}

else {

$("#followdiv").find('#btnfollowUnfollow').addClass("text-success");

$("#followdiv").find("#btnfollowUnfollow").text("Follow");

}

LoadFollwingAndFollwerCount();

},

error: function (error) {

}

});

return false;

});

$("#btnUpdateMsg").click(function () {

if (validateMessage()) {

var Messagetxt = $("#MessageUpd").val();

$.ajax({

url: "/Home/PostTwitter",

method: "POST",

data: '{ "messagePost": "' + Messagetxt + '"}',

dataType: "json",

contentType: "application/json; charset=utf-8",

success: function () {

$("#MessageUpd").val('');

$("#spnSuccess").removeClass("alert-danger");

$("#spnSuccess").addClass("alert-success");

$("#spnSuccess").text(" Posted successfully.");

LoadData();

},

error: function (error) {

$("#spnSuccess").addClass("alert-danger");

$("#spnSuccess").text(" Error while posting");

console.log(error);

}

});

return false;

}

});

});

function LoadData() {

$('#tweetsDetails').empty();

$.ajax({

url: "/Home/GetTwitter",

method: "GET",

dataType: 'json',

success: function (data) {

data = JSON.parse(data);

$.each(data, function (i, item) {

var rows = "<h5>"

+ "<span class='spanUserID form-control'>" + item.user\_id + " " + item.created + "</span>"

+ "<label class='spanMessage '>" + item.messaage + "</label>"

+ "</h5>";

$('#tweetsDetails').append(rows);

});

LoadTweetCount();

},

error: function (ex) {

var r = jQuery.parseJSON(response.responseText);

alert("Message: " + r.Message);

alert("StackTrace: " + r.StackTrace);

alert("ExceptionType: " + r.ExceptionType);

}

});

return false;

}

function LoadTweetCount() {

$.ajax({

url: "/Home/GetTwittecount",

method: "GET",

dataType: 'json',

success: function (data) {

data = JSON.parse(data);

$("#twtCnt u").text(data);

},

error: function (ex) {

var r = jQuery.parseJSON(response.responseText);

alert("Message: " + r.Message);

alert("StackTrace: " + r.StackTrace);

alert("ExceptionType: " + r.ExceptionType);

}

});

return false;

}

function LoadFollwingAndFollwerCount() {

$.ajax({

url: "/Home/GetFollowingFollowercount",

method: "GET",

dataType: 'json',

success: function (data) {

data = JSON.parse(data);

$("#FollwingCnt u").text(data.FollowingCnt);

$("#FollwertCnt u").text(data.FollowerCnt);

},

error: function (ex) {

var r = jQuery.parseJSON(response.responseText);

alert("Message: " + r.Message);

alert("StackTrace: " + r.StackTrace);

alert("ExceptionType: " + r.ExceptionType);

}

});

return false;

}

function validateMessage() {

if ($("#MessageUpd").val() == '') {

$("#spnSuccess").removeClass("alert-success");

$("#spnSuccess").addClass("alert-danger");

$("#spnSuccess").text(" Please enter your message to post");

return false;

}

else {

var text\_max = 130;

$('#textarea\_feedback').html(text\_max + ' characters remaining');

return true;

}

}

</script>

Profile views

@model Tweeter.Models.EditProfile

@{

if (ViewContext.IsChildAction != true && !Request.IsAjaxRequest())

{

Layout = "~/Views/Shared/\_Layout.cshtml";

}

else

{

Layout = null;

}

}

@{

ViewBag.Title = "Edit";

}

<script type="text/javascript" language="javascript">

var urlContent = "@Url.Content("~/")";

</script>

@using (Html.BeginForm())

{

@Html.AntiForgeryToken()

<div class="form-horizontal">

<h4 style="color:cornflowerblue">Edit profile</h4>

<hr />

@Html.ValidationSummary(true, "", new { @class = "text-danger" })

<div class="form-group">

@Html.LabelFor(model => model.fullName, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.fullName, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.fullName, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.emailID, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.emailID, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.emailID, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.active, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

<div class="checkbox">

@Html.EditorFor(model => model.active)

@Html.ValidationMessageFor(model => model.active, "", new { @class = "text-danger" })

</div>

</div>

</div>

<div class="form-group">

<div class="col-md-offset-2 col-md-10">

<input type="submit" value="Save" class="btn btn-default" />

</div>

</div>

</div>

}

<div>

@Html.ActionLink("Back to Home", "Index", "Home")

</div>

Details.cs.html

@model Tweeter.Models.EditProfile

@{

if (ViewContext.IsChildAction != true && !Request.IsAjaxRequest())

{

Layout = "~/Views/Shared/\_Layout.cshtml";

}

else

{

Layout = null;

}

}

@{

ViewBag.Title = "Details";

}

<script type="text/javascript" language="javascript">

var urlContent = "@Url.Content("~/")";

</script>

<p style="float:right">

@Html.ActionLink("Change Password", "UpdatePassword") |

@Html.ActionLink("Edit Profile", "Edit") |

@Html.ActionLink("Profile Details", "Details") |

@Html.ActionLink("Delete Profile", "Delete")

</p>

<div>

<h4 style="color:cornflowerblue">Profile details</h4>

<hr />

<dl class="dl-horizontal">

<dt>

@Html.DisplayNameFor(model => model.fullName)

</dt>

<dd>

@Html.DisplayFor(model => model.fullName)

</dd>

<dt>

@Html.DisplayNameFor(model => model.emailID)

</dt>

<dd>

@Html.DisplayFor(model => model.emailID)

</dd>

<dt>

@Html.DisplayNameFor(model => model.joined)

</dt>

<dd>

@Html.DisplayFor(model => model.joined)

</dd>

<dt>

@Html.DisplayNameFor(model => model.active)

</dt>

<dd>

@Html.DisplayFor(model => model.active)

</dd>

</dl>

</div>

<div>

@Html.ActionLink("Back to Home", "Index", "Home")

</div>

Delete.cs.html

@model Tweeter.Models.Person

@{

if (ViewContext.IsChildAction != true && !Request.IsAjaxRequest())

{

Layout = "~/Views/Shared/\_Layout.cshtml";

}

else

{

Layout = null;

}

}

@{

ViewBag.Title = "Delete";

}

<script type="text/javascript" language="javascript">

var urlContent = "@Url.Content("~/")";

</script>

<h3>Are you sure you want to delete this?</h3>

<div>

<h4 style="color:cornflowerblue">Delete profile</h4>

<hr />

<dl class="dl-horizontal">

<dt>

@Html.DisplayNameFor(model => model.fullName)

</dt>

<dd>

@Html.DisplayFor(model => model.fullName)

</dd>

<dt>

@Html.DisplayNameFor(model => model.email)

</dt>

<dd>

@Html.DisplayFor(model => model.email)

</dd>

<dt>

@Html.DisplayNameFor(model => model.joined)

</dt>

<dd>

@Html.DisplayFor(model => model.joined)

</dd>

<dt>

@Html.DisplayNameFor(model => model.active)

</dt>

<dd>

@Html.DisplayFor(model => model.active)

</dd>

</dl>

@using (Html.BeginForm())

{

@Html.AntiForgeryToken()

<div class="form-actions no-color">

<input type="submit" value="Delete" class="btn btn-default" /> |

@Html.ActionLink("Back to Home", "Index", "Home")

</div>

}

</div>

ChangePassword.cs.html

@model Tweeter.Models.UpdatePassword

@{

if (ViewContext.IsChildAction != true && !Request.IsAjaxRequest())

{

Layout = "~/Views/Shared/\_Layout.cshtml";

}

else

{

Layout = null;

}

}

@{

ViewBag.Title = "UpdatePassword";

}

<script type="text/javascript" language="javascript">

var urlContent = "@Url.Content("~/")";

</script>

@using (Html.BeginForm())

{

@Html.AntiForgeryToken()

<h4 style="color:cornflowerblue">Update password</h4>

<hr />

<div class="form-horizontal">

@Html.ValidationSummary(true, "", new { @class = "text-danger" })

<div class="form-group">

<div class="col-md-12">

@Html.LabelFor(model => model.oldpassWord, htmlAttributes: new { @class = "control-label " })

@Html.PasswordFor(model => model.oldpassWord, new { @class = "form-control" })

@Html.ValidationMessageFor(model => model.oldpassWord, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

<div class="col-md-12">

@Html.LabelFor(model => model.NewpassWord, htmlAttributes: new { @class = "control-label " })

@Html.PasswordFor(model => model.NewpassWord, new { @class = "form-control" })

@Html.ValidationMessageFor(model => model.NewpassWord, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

<div class="col-md-12">

@Html.LabelFor(model => model.ConfirmpassWord, htmlAttributes: new { @class = "control-label " })

@Html.PasswordFor(model => model.ConfirmpassWord, new { @class = "form-control" })

@Html.ValidationMessageFor(model => model.ConfirmpassWord, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

<div class="col-md-offset-2 col-md-10">

<input type="submit" value="Update" class="btn btn-default" />

</div>

</div>

</div>

}

<div>

@Html.ActionLink("Back to Home", "Index", "Home")

</div>

@if (ViewBag.PasswordUpdateSuccess == true)

{

<h5 class="alert-success">Password updated successfully..!</h5>

}

Shared

DefaultLayoutPage.cs.html

<!DOCTYPE html>

<html>

<head>

<meta charset="utf-8" />

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>@ViewBag.Title - My ASP.NET Application</title>

<link href="~/Content/Site.css" rel="stylesheet" type="text/css" />

<link href="~/Content/bootstrap.min.css" rel="stylesheet" type="text/css" />

<script src="~/Scripts/modernizr-2.6.2.js"></script>

</head>

<body>

<div class="navbar navbar-inverse navbar-fixed-top">

<div class="container">

<div class="navbar-header">

<button type="button" class="navbar-toggle" data-toggle="collapse" data-target=".navbar-collapse">

<span class="icon-bar"></span>

<span class="icon-bar"></span>

<span class="icon-bar"></span>

</button>

@Html.ActionLink("Application name", "Index", "Home", null, new { @class = "navbar-brand" })

</div>

<div class="navbar-collapse collapse">

<ul class="nav navbar-nav"></ul>

</div>

</div>

</div>

<div class="container body-content">

@RenderBody()

<hr />

<footer>

<p>&copy; @DateTime.Now.Year - My ASP.NET Application</p>

</footer>

</div>

<script src="~/Scripts/jquery-1.10.2.min.js"></script>

<script src="~/Scripts/bootstrap.min.js"></script>

</body>

</html>

Layout.cs.html

<!DOCTYPE html>

<html>

<head>

<meta charset="utf-8" />

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>@ViewBag.Title - My ASP.NET Application</title>

<link href="~/Content/Site.css" rel="stylesheet" type="text/css" />

<link href="~/Content/bootstrap.min.css" rel="stylesheet" type="text/css" />

<script src="~/Scripts/modernizr-2.6.2.js"></script>

</head>

<body>

<div class="navbar navbar-inverse navbar-fixed-top">

<div class="container">

<div class="navbar-header">

<button type="button" class="navbar-toggle" data-toggle="collapse" data-target=".navbar-collapse">

<span class="icon-bar"></span>

<span class="icon-bar"></span>

<span class="icon-bar"></span>

</button>

@Html.ActionLink("Twitter Clone", "Index", "Home", null, new { @class = "navbar-brand" })

@Html.ActionLink("Signout", "login", "SignIn", null, new { @class ="navbar-brand"})

</div>

<div class="navbar-collapse collapse">

<ul class="nav navbar-nav">

</ul>

</div>

</div>

</div>

<div class="container body-content">

@RenderBody()

<hr />

<footer>

<p>&copy; @DateTime.Now.Year - My ASP.NET Application</p>

</footer>

</div>

<script src="~/Scripts/jquery-1.10.2.min.js"></script>

<script src="~/Scripts/bootstrap.min.js"></script>

</body>

</html>

SignIn

Login.cs.html

@model Tweeter.Models.Account

@{

if (ViewContext.IsChildAction != true && !Request.IsAjaxRequest())

{

Layout = "~/Views/Shared/\_DefaultLayoutPage.cshtml";

}

else

{

Layout = null;

}

}

@{

ViewBag.Title = "login";

}

<script type="text/javascript" language="javascript">

var urlContent = "@Url.Content("~/")";

window.onload = function window\_OnLoad() {

nobackhfive();

}

</script>

<style>

.wrapper {

margin: 0 auto;

}

.contentSignin {

background-color: lightskyblue;

width: 400px;

height: 330px;

position: absolute;

top: 0;

bottom: 0;

left: 0;

right: 0;

margin: auto;

border-radius: 9px;

}

.innercontent {

padding: 18px;

}

.HeaderSignin {

background-color: #2ba6cb;

border-top-left-radius: 9px;

border-top-right-radius: 9px;

height: 51px;

align-items: center;

}

</style>

<h2 style="color:turquoise">Welcome to My Twitter Clone </h2>

<hr />

<div class="contentSignin">

@using (Html.BeginForm("login", "SignIn", FormMethod.Post))

{

@Html.AntiForgeryToken()

<div class="HeaderSignin">

<h5 style="padding-left:40%;padding-top:12px; font-size: 18px;margin-top: 0px">Login</h5>

</div>

<div class="innercontent">

@Html.ValidationSummary(true, "", new { @class = "text-danger" })

<div class="form-group">

<div class="col-md-12">

@Html.LabelFor(model => model.userName, htmlAttributes: new { @class = "control-label " })

@Html.EditorFor(model => model.userName, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.userName, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

<div class="col-md-12">

@Html.LabelFor(model => model.passWord, htmlAttributes: new { @class = "control-label " })

@Html.PasswordFor(model => model.passWord, new { @class = "form-control" })

@Html.ValidationMessageFor(model => model.passWord, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

<div class="col-md-offset-7 col-md-5">

<input type="submit" value="Login" class="btn btn-primary" />

</div>

</div>

</div>

}

<div class="innercontent">

<h5>New User? @Html.ActionLink("SignUp", "SignUp")</h5>

@if (ViewBag.Inalidlogin != null && ViewBag.Inalidlogin != string.Empty)

{

<h5 class="alert-danger">@ViewBag.Inalidlogin</h5>

}

</div>

</div>

Signup.cs.html

@model Tweeter.Models.Person

@{

if (ViewContext.IsChildAction != true && !Request.IsAjaxRequest())

{

Layout = "~/Views/Shared/\_DefaultLayoutPage.cshtml";

}

else

{

Layout = null;

}

}

@{

ViewBag.Title = "SignUp";

}

<head>

<title>

@ViewBag.Title

</title>

<script src="~/Scripts/jquery-1.10.2.js"></script>

<!-- Latest compiled and minified CSS -->

<link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css" integrity="sha384-BVYiiSIFeK1dGmJRAkycuHAHRg32OmUcww7on3RYdg4Va+PmSTsz/K68vbdEjh4u" crossorigin="anonymous">

<!-- Optional theme -->

<link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap-theme.min.css" integrity="sha384-rHyoN1iRsVXV4nD0JutlnGaslCJuC7uwjduW9SVrLvRYooPp2bWYgmgJQIXwl/Sp" crossorigin="anonymous">

<!-- Latest compiled and minified JavaScript -->

<script src="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/js/bootstrap.min.js" integrity="sha384-Tc5IQib027qvyjSMfHjOMaLkfuWVxZxUPnCJA7l2mCWNIpG9mGCD8wGNIcPD7Txa" crossorigin="anonymous"></script>

</head>

<script type="text/javascript" language="javascript">

var urlContent = "@Url.Content("~/")";

window.onload = function window\_OnLoad() {

nobackhfive();

}

</script>

<style>

.contentSignup {

height: auto;

top: 0;

bottom: 0;

left: 0;

right: 0;

margin-left: 45px;

background-color : skyblue;

border-spacing:inherit;

}

</style>

<h2 style="color:turquoise">Join My Twitter Clone</h2>

<hr />

@using (Html.BeginForm("SignUp", "SignIn", FormMethod.Post))

{

@Html.AntiForgeryToken()

<div class="contentSignup">

<fieldset>

<legend style="color:blue;font-weight:bold;">Enter Your Details</legend>

@Html.ValidationSummary(true, "", new { @class = "text-danger" })

<div class="form-group">

<div class="col-md-12">

<div class="col-md-2">@Html.LabelFor(model => model.user\_id, htmlAttributes: new { @class = "control-label" })</div>

<div class="col-md-3">@Html.EditorFor(model => model.user\_id, new { htmlAttributes = new { @class = "form-control" } })</div>

<div class="col-md-3">@Html.ValidationMessageFor(model => model.user\_id, "", new { @class = "text-danger" })</div>

</div>

</div>

<div class="form-group">

<div class="col-md-12">

<div class="col-md-2 ">@Html.LabelFor(model => model.password, htmlAttributes: new { @class = "control-label" })</div>

<div class="col-md-3">@Html.PasswordFor(model => model.password, new { htmlAttributes = new { @class = "form-control" } })</div>

<div class="col-md-3">@Html.ValidationMessageFor(model => model.password, "", new { @class = "text-danger" })</div>

</div>

</div>

<div class="form-group">

<div class="col-md-12">

<div class="col-md-2">@Html.LabelFor(model => model.fullName, htmlAttributes: new { @class = "control-label" })</div>

<div class="col-md-3">@Html.EditorFor(model => model.fullName, new { htmlAttributes = new { @class = "form-control" } })</div>

<div class="col-md-3">@Html.ValidationMessageFor(model => model.fullName, "", new { @class = "text-danger" })</div>

</div>

</div>

<div class="form-group">

<div class="col-md-12">

<div class="col-md-2">@Html.LabelFor(model => model.email, htmlAttributes: new { @class = "control-label" })</div>

<div class="col-md-3">@Html.EditorFor(model => model.email, new { htmlAttributes = new { @class = "form-control" } })</div>

<div class="col-md-3">@Html.ValidationMessageFor(model => model.email, "", new { @class = "text-danger" })</div>

</div>

</div>

<div class="form-group">

<div class="col-md-offset-2 col-md-10">

<input type="submit" value="Signup" class="btn btn-primary" />

</div>

</div>

</fieldset>

</div>

}

<div style="margin-left: 45px">

@Html.ActionLink("Back to login", "login")

</div>

@if (ViewBag.SuccessMessage != null && ViewBag.SuccessMessage != string.Empty)

{

<h5 class="alert-danger">@ViewBag.SuccessMessage</h5>

}

@if (ViewBag.SignupSuccess == "True")

{

<h5 class="alert-success">Signup successful, click Back to login </h5>

}

@if (ViewBag.errorMessage != null && ViewBag.errorMessage != string.Empty)

{

<h5 class="alert-danger">@ViewBag.errorMessage</h5>

}

ViewStart.cs.thml.

@{

Layout = "~/Views/Shared/\_Layout.cshtml";

}

Web.config file

<?xml version="1.0"?>

<!--

For more information on how to configure your ASP.NET application, please visit

http://go.microsoft.com/fwlink/?LinkId=301880

-->

<configuration>

<configSections>

<!-- For more information on Entity Framework configuration, visit http://go.microsoft.com/fwlink/?LinkID=237468 -->

<section name="entityFramework" type="System.Data.Entity.Internal.ConfigFile.EntityFrameworkSection, EntityFramework, Version=6.0.0.0, Culture=neutral, PublicKeyToken=b77a5c561934e089" requirePermission="false"/>

</configSections>

<appSettings>

<add key="webpages:Version" value="3.0.0.0"/>

<add key="webpages:Enabled" value="false"/>

<add key="ClientValidationEnabled" value="true"/>

<add key="UnobtrusiveJavaScriptEnabled" value="true"/>

</appSettings>

<!--

For a description of web.config changes see http://go.microsoft.com/fwlink/?LinkId=235367.

The following attributes can be set on the <httpRuntime> tag.

<system.Web>

<httpRuntime targetFramework="4.5.2" />

</system.Web>

-->

<system.web>

<authentication mode="Forms">

<forms loginUrl="~/SignIn/Login" timeout="1" />

</authentication>

<compilation debug="true" targetFramework="4.5.2"/>

<httpRuntime targetFramework="4.5.2"/>

</system.web>

<connectionStrings>

<add name="TweeterEntities" connectionString="metadata=res://\*/Models.TwitterEntityModel.csdl|res://\*/Models.TwitterEntityModel.ssdl|res://\*/Models.TwitterEntityModel.msl;provider=System.Data.SqlClient;provider connection string=&quot;data source=.;initial catalog=Tweeter;integrated security=True;MultipleActiveResultSets=True;App=EntityFramework&quot;" providerName="System.Data.EntityClient"/>

<add name="TweeterEntities1" connectionString="metadata=res://\*/Models.TwitterEntityModel.csdl|res://\*/Models.TwitterEntityModel.ssdl|res://\*/Models.TwitterEntityModel.msl;provider=System.Data.SqlClient;provider connection string=&quot;data source=.;initial catalog=Tweeter;integrated security=True;MultipleActiveResultSets=True;App=EntityFramework&quot;" providerName="System.Data.EntityClient"/>

</connectionStrings>

<entityFramework>

<defaultConnectionFactory type="System.Data.Entity.Infrastructure.LocalDbConnectionFactory, EntityFramework">

<parameters>

<parameter value="v11.0"/>

</parameters>

</defaultConnectionFactory>

<providers>

<provider invariantName="System.Data.SqlClient" type="System.Data.Entity.SqlServer.SqlProviderServices, EntityFramework.SqlServer"/>

</providers>

</entityFramework>

</configuration>

SQL Details.

CREATE TABLE Person(user\_id VARCHAR(25) PRIMARY KEY,

password VARCHAR(50) NOT NULL,

fullName VARCHAR(30) NOT NULL,

email VARCHAR(50) NOT NULL,

Joined DateTime NOT NULL,

active bit NOT NULL)

Create table tweet (tweet\_id int IDENTITY(1,1) PRIMARY KEY,

user\_id varchar(25) NOT NULL REFERENCES Person(user\_id),

message varchar(140) NOT NULL,

created datetime NOT NULL)

--drop table following

create table following (FId INT IDENTITY(1,1) PRIMARY KEY,user\_id varchar(25) NOT NULL REFERENCES Person(user\_id),

following\_id varchar(25) NOT NULL);