# Step by step for the Login and Registration:

1. **Make the UserAccount model**

public class UserAccount

{

[Key]

public int UserId { get; set; }

[Required(ErrorMessage = "First name is required.")]

public string FirstName { get; set; }

[Required(ErrorMessage = "Last name is required.")]

public string LastName { get; set; }

[Required(ErrorMessage = "Email is required.")]

[EmailAddress(ErrorMessage = "Invalid Email Address")]

public string Email { get; set; }

[Required(ErrorMessage = "Username is required.")]

public string Username { get; set; }

[Required(ErrorMessage = "Password is required.")]

[DataType(DataType.Password)]

public string Password { get; set; }

[Compare("Password", ErrorMessage = "Please confirm your password.")]

[DataType(DataType.Password)]

public string ConfirmPassword { get; set; }

}

1. **Add the UserAccount model to the DbContext object for code first db approach**

public DbSet<UserAccount> UserAccounts { get; set; }

1. **Create the AccountController**

public class AccountController : Controller

{

}

1. **Add the REGISTER ActionMethod (HTTPGET) that returns the register view.**

public ActionResult Register()

{

return View();

}

1. **Add the REGISTER ActionMethod (HTTPPOST) that submits the data after successfully filling out a registration form.**
   1. **The UserAccount parameter is saved to the database**

[HttpPost]

public ActionResult Register(UserAccount account)

{

if (ModelState.IsValid)

{

using (DatabaseContext ctx = new DatabaseContext())

{

ctx.UserAccounts.Add(account);

ctx.SaveChanges(); // Saved to the database

}

MigrateShoppingCart(account.Username);

Session["UserId"] = account.UserId.ToString();

Session["Username"] = account.Username.ToString();

return RedirectToAction("LoggedIn");

}

return View();

}

1. **Add the LOGIN ActionMethod (HTTPGET) that returns the login view.**

// Login

public ActionResult Login()

{

return View();

}

1. **Add the LOGIN ActionMethod (HTTPPOST) that validates a user’s credentials and makes a session for them.**
   1. **After the session is made, the user is redirected to the store view.**

[HttpPost]

public ActionResult Login(UserAccount user)

{

using (DatabaseContext ctx = new DatabaseContext())

{

var usr = ctx.UserAccounts.Where(u => u.Username == user.Username &&

u.Password == user.Password).FirstOrDefault();

if (usr != null)

{

MigrateShoppingCart(usr.Username);

Session["UserId"] = usr.UserId.ToString();

Session["Username"] = usr.Username.ToString();

return RedirectToAction("LoggedIn");

}

else

{

ModelState.AddModelError("", "Username or Password is incorrect.");

}

}

return View();

}

1. **Add the LOGOUT ActionMethod that ends the user’s session and redirects them to the LOGIN ActionMethod.**

public ActionResult Logout()

{

Session.Abandon();

return RedirectToAction("Login");

}

1. **Create the view for registration**
   1. **Redirect user back to home page after redirect is complete to test for successful registration**

@model MusicStore1.Models.UserAccount

@{

ViewBag.Title = "Register";

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

}

<h2>Register</h2>

@using (Html.BeginForm())

{

@Html.AntiForgeryToken()

<div class="form-horizontal">

<hr />

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

@if (ViewBag.Message != null)

{

<div class="form-group">

<div class="col-md-10">@ViewBag.Message</div>

</div>

}

<div class="form-group">

@Html.LabelFor(model => model.FirstName, htmlAttributes: new { @class =

"control-label col-md-2" })

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

@Html.EditorFor(model => model.FirstName, new { htmlAttributes = new {

@class = "form-control" } })

@Html.ValidationMessageFor(model => model.FirstName, "", new { @class =

"text-danger" })

</div>

</div>

<!—Remaining Code left out for brevity purposes. It follows the same pattern as the last div element--!>

1. **Create the view for login**
   1. **The strongly-typed model information for login credentials is sent back to the controller when the user submits valid credentials.**

@model MusicStore1.Models.UserAccount

@{

ViewBag.Title = "Login";

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

}

<h2>Login</h2>

@using (Html.BeginForm())

{

@Html.AntiForgeryToken()

<div class="form-horizontal">

<hr />

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

<div class="form-group">

@Html.LabelFor(model => model.Username, htmlAttributes: new { @class =

"control-label col-md-2" })

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

@Html.EditorFor(model => model.Username, new { htmlAttributes = new {

@class = "form-control" } })

@Html.ValidationMessageFor(model => model.Username, "", new { @class =

"text-danger" })

</div>

</div>

<!—Remaining Code left out for brevity purposes. It follows the same pattern as the last div element--!>

1. **Add the Navbar links for login/logout and registration in the shared layout view.**

<li>@Html.ActionLink("Register", "Register", "Account")</li>

@if (@HttpContext.Current.Session["UserName"] != null)

{

<li>@Html.ActionLink("Logout", "Logout", "Account")</li>

<li>@Html.ActionLink("Admin", "Index", "StoreManager")</li>

}

else

{

<li>@Html.ActionLink("Login", "Login", "Account")</li>

}