

# ASP.NET Core

Authentication & Authorization

- Authentication
- Roles
- Authorization

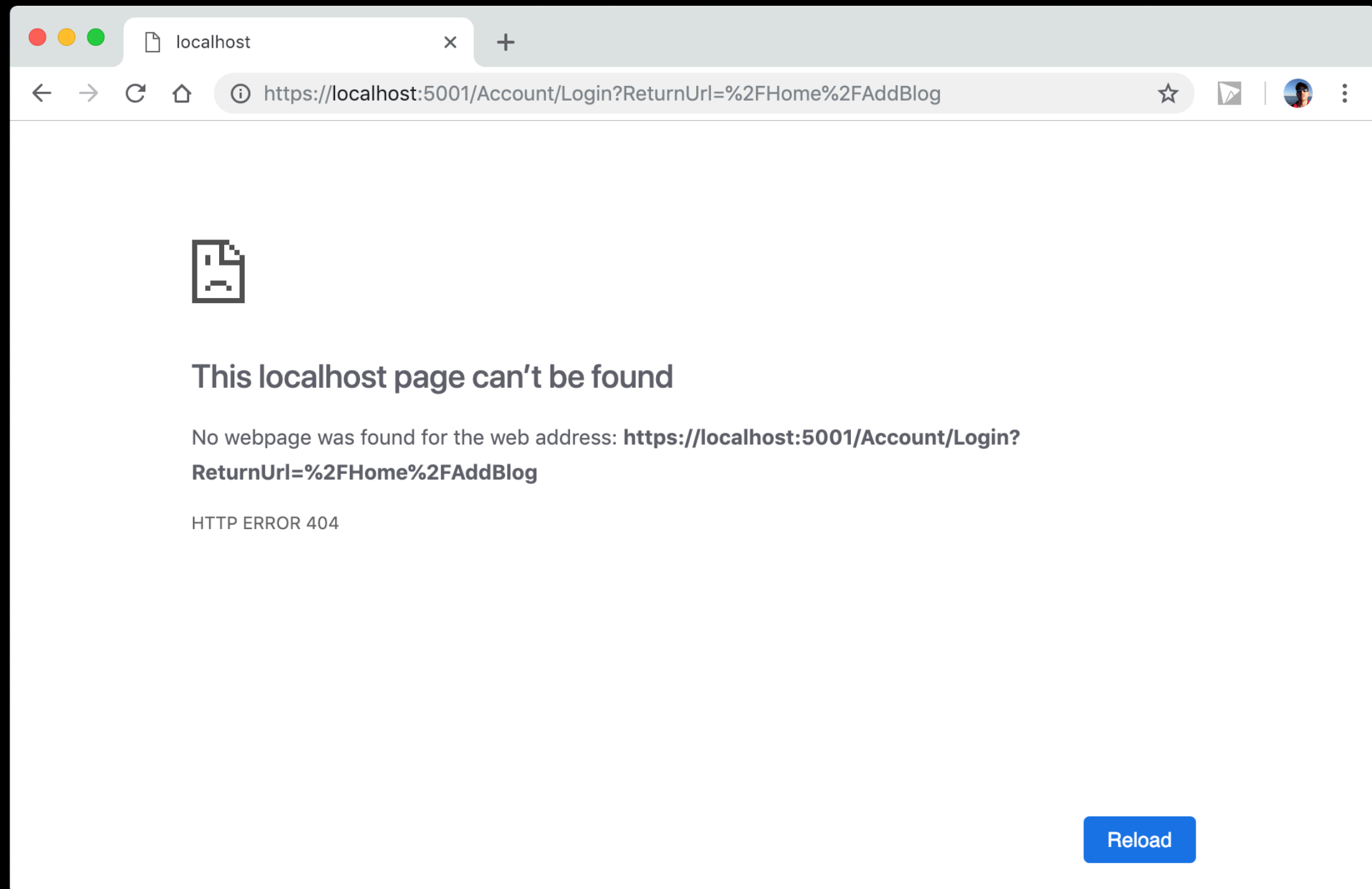
# Authentication

- The primary purpose for ASP.NET Core Identity is to authenticate users
- The main tool we use to restrict access to web services (like controller methods) is the Authorize attribute

# Authentication - Demo

- Add the Authorize attribute to the Home/AddBlog controller method
- The Authorize attribute is found in the `Microsoft.AspNetCore.Authorization` namespace
- Test in browser, try adding a new Blog

# Authentication - Demo



# Authentication

- The unauthenticated request is redirected to Account/Login
- The default login URL can be changed (p. 922), however, since this is a logical URL, we will utilize the default

# Authentication - Demo

- Modify the User View Model that was created in the last lesson - it currently defines the class needed to Create a new user
- Define the model class to support user login
- Utilize the UIHint attribute to ensure that the tag helper will render the appropriate form field (email, password)

# Authentication

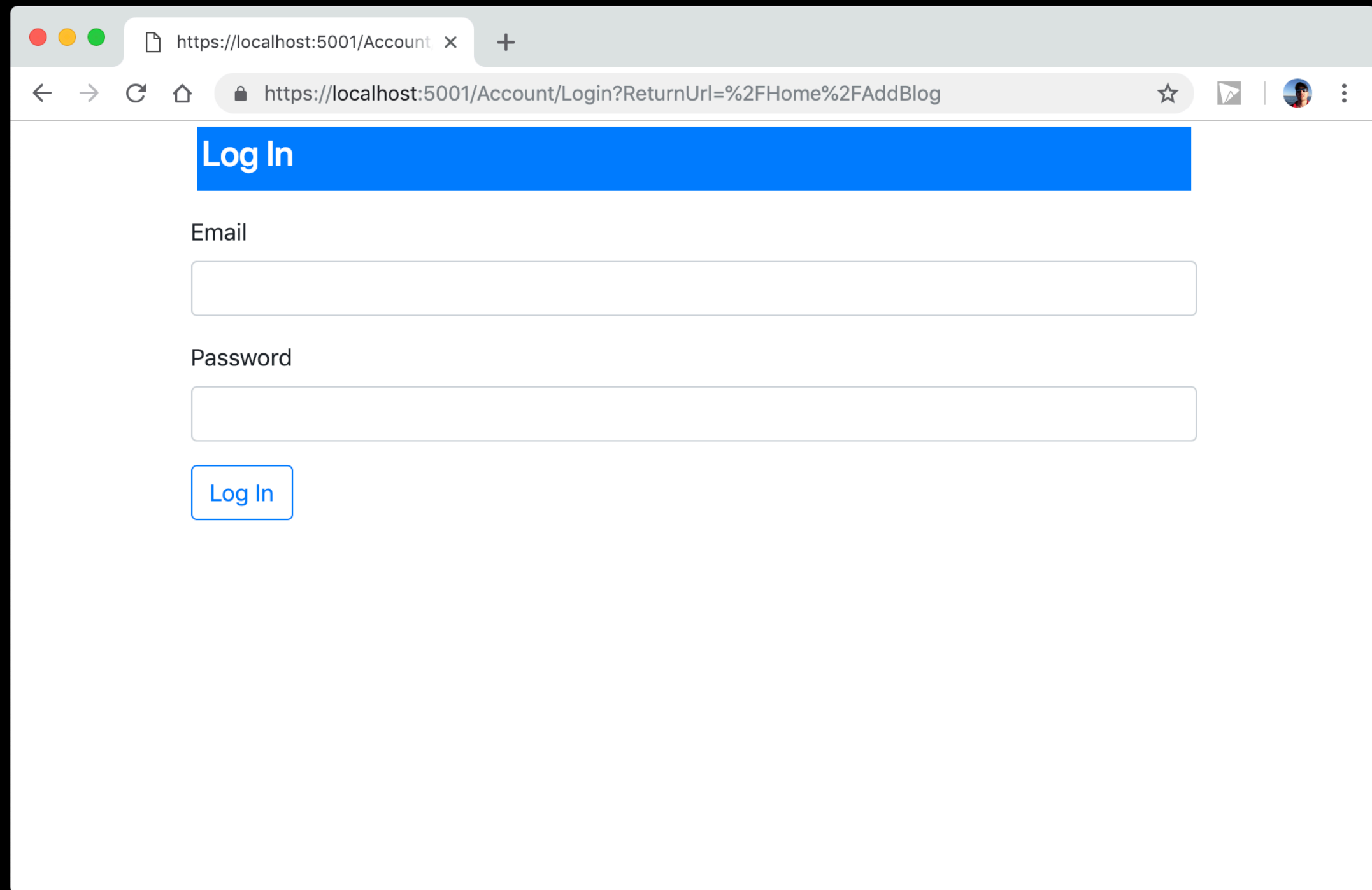
- The Authorize attribute can be used to restrict access to a single controller method or an entire class
- This can be a more secure way of ensuring authentication for related services
- Individual controller methods can override the Authorize attribute using theAllowAnonymous attribute



# Authentication - Demo

- Create the Account Controller
- Add the Login controller method that responds to http get requests (it will display a view)
- Create the related view

# Authentication - Demo



https://localhost:5001/Account x +

← → ↻ 🏠 🔒 https://localhost:5001/Account/Login?ReturnUrl=%2FHome%2FAddBlog ☆ 📧 👤 ⋮

## Log In

Email

Password

Log In

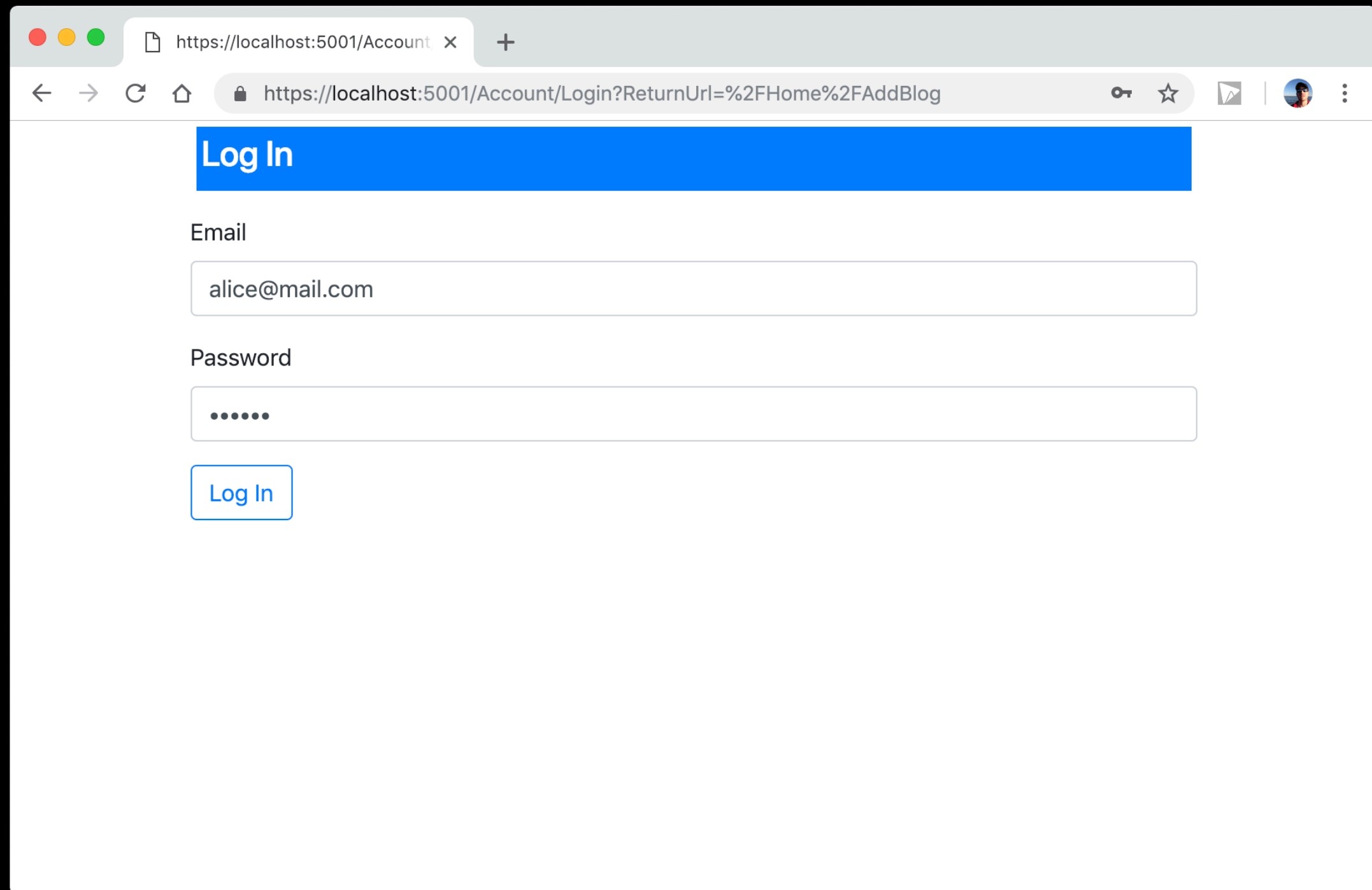
# Authentication - Demo

```
10 <body>
11   <div class="container">
12
13   <div class="bg-primary m-1 p-1 text-white"><h4>Log In</h4></div>
14   <div class="text-danger validation-summary-valid" data-valmsg-summary="true"><ul><li style="display:none"></li>
15   </ul></div>
16   <form method="post" action="/Account/Create">
17     <input type="hidden" name="returnUrl" value="/Home/AddBlog" />
18     <div class="form-group">
19       <label for="Email">Email</label>
20       <input class="form-control" type="text" data-val="true" data-val-required="The Email field is required."
21       id="Email" name="Email" value="" />
22     </div>
23     <div class="form-group">
24       <label for="Password">Password</label>
25       <input class="form-control" type="password" data-val="true" data-val-required="The Password field is
26       required." id="Password" name="Password" />
27     </div>
28     <button type="submit" class="btn btn-outline-primary">Log In</button>
29     <input name="__RequestVerificationToken" type="hidden"
30     value="CfDJ8NeItmijddxCujVp5GrVtBZAvL8nennKnjyAJA636V6xy9J8ZA0lvCZvrvoapvvDdQ6k-
31     ayRuJjTyVT5QKhDWpX6JJqZdF3EwyNDpfSfdo5pJAX3UUJE65JKwV-5wz2Su-sNUIfgfSOntnMCpOROFxU" /></form>
32
33   </div>
34   <!-- jQuery -->
35   <script src="https://code.jquery.com/jquery-3.3.1.min.js"
36   integrity="sha256-FgpCb/KJQlLNfOu91ta32o/NMZxltwRo8QtmkMRdAu8="
37   crossorigin="anonymous"></script>
38   <!-- Bootstrap -->
39   <script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.6/umd/popper.min.js" integrity="sha384-
40   wHAIffRlMFy6i5SRaxvfOCifBUQy1xHdJ/yoi7FRNXMRBu5WHdZYU1hA6ZOblgut" crossorigin="anonymous"></script>
41   <script src="https://stackpath.bootstrapcdn.com/bootstrap/4.2.1/js/bootstrap.min.js" integrity="sha384-
42   B0UglyR+jN6CkvvICOB2joaf5I413gm9GU6Hclog6Ls7i6U/mkkaduKaBhlAXv9k" crossorigin="anonymous"></script>
43 </body>
44 </html>
```

# Authentication - Demo

- Add the controller method to handle http post requests for Account/Login

# Authentication - Demo



Log In

Email

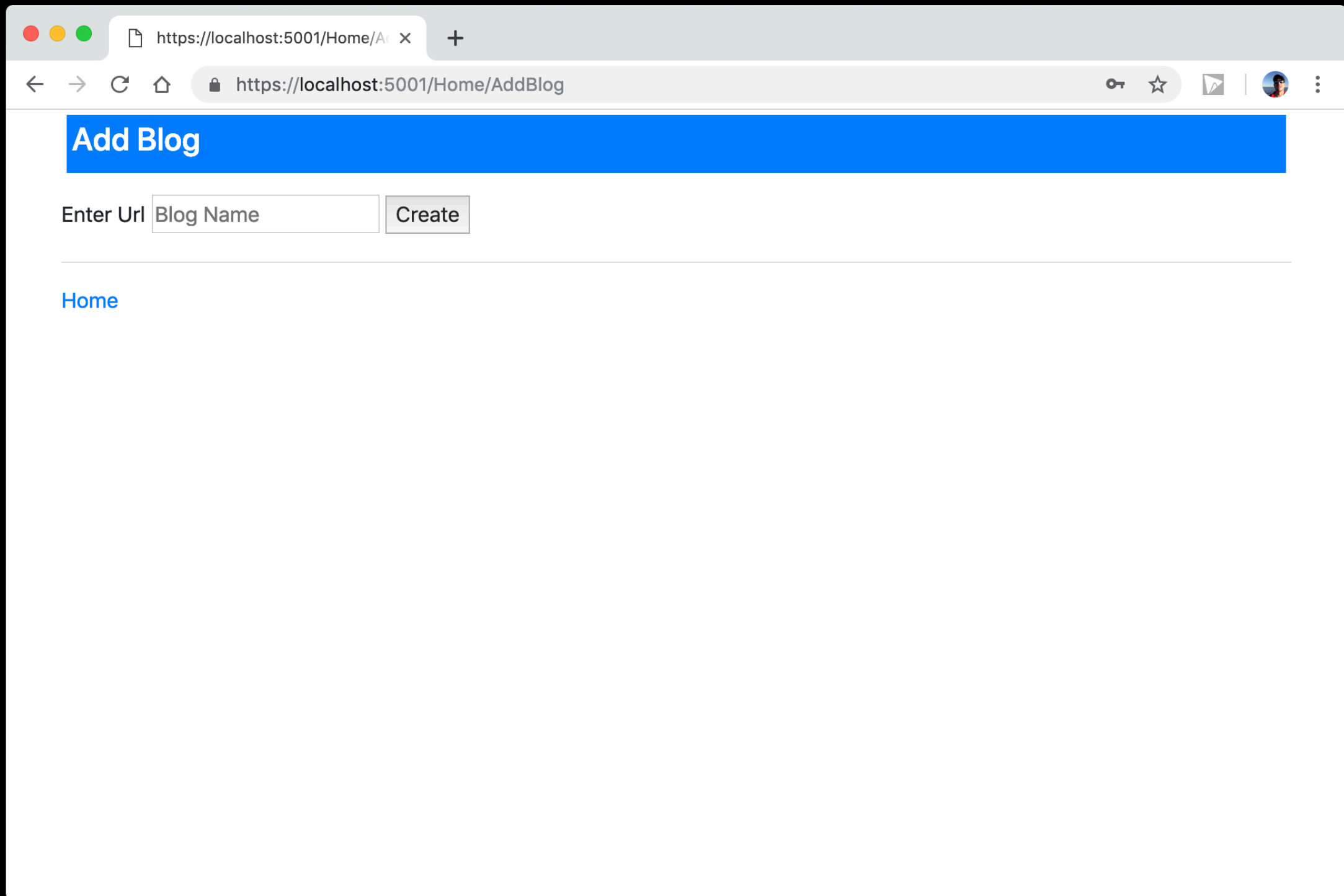
alice@mail.com

Password

.....

Log In

# Authentication - Demo



A screenshot of a web browser window. The address bar shows the URL `https://localhost:5001/Home/AddBlog`. The page has a blue header bar with the text "Add Blog". Below the header, there is a form with the label "Enter Url" followed by a text input field containing "Blog Name" and a "Create" button. A horizontal line separates the form from the rest of the page. Below the line, there is a blue link labeled "Home".

https://localhost:5001/Home/Add x +

← → ↻ 🏠 🔒 https://localhost:5001/Home/AddBlog 🔑 ☆ 📄 👤 ⋮

## Add Blog

Enter Url

---

[Home](#)

# Authentication - Demo

- In addition to requiring authentication for the Home/AddBlog controller method that responds to http get requests, we must also require the same for the Home/AddBlog controller method that responds to http post requests
- Add the Authorize attribute to the Home/AddBlog controller method that responds to http post requests

# Authentication

- There are other controller methods that should require authentication, we will address those near the end of this demo



# Roles

- According to the book (p. 928), a role is a label that you define to represent permission to perform a set of activities within main application

# Roles

In our Blogging application:

- Any user will be able to view blogs & posts
- Any authenticated user will be able to add posts to a blog

# Roles

In our Blogging application:

- We will define 2 roles with elevated abilities:
  1. Moderate - can edit and delete blogs and posts
  2. Admin - can create, edit, delete users and assign / remove users to / from roles

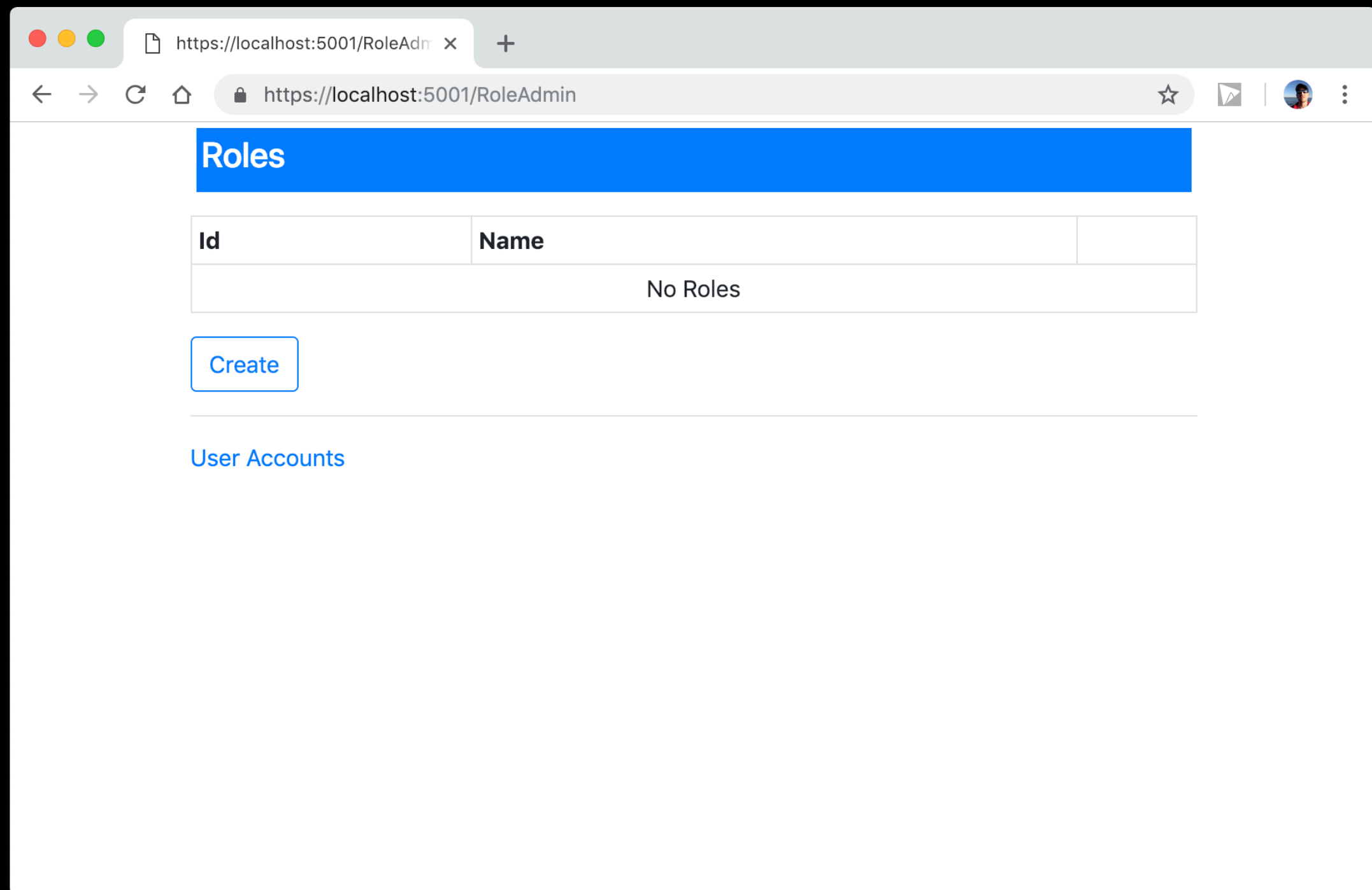
# Roles

- The first step is to update our Admin application to provide the ability to view / update roles and users

# Roles - Demo

- Add RoleAdmin controller
- Add RoleAdmin/Index controller method
- Add related view (display list of roles)
- Create links to/from Admin/Index and RoleAdmin/Index

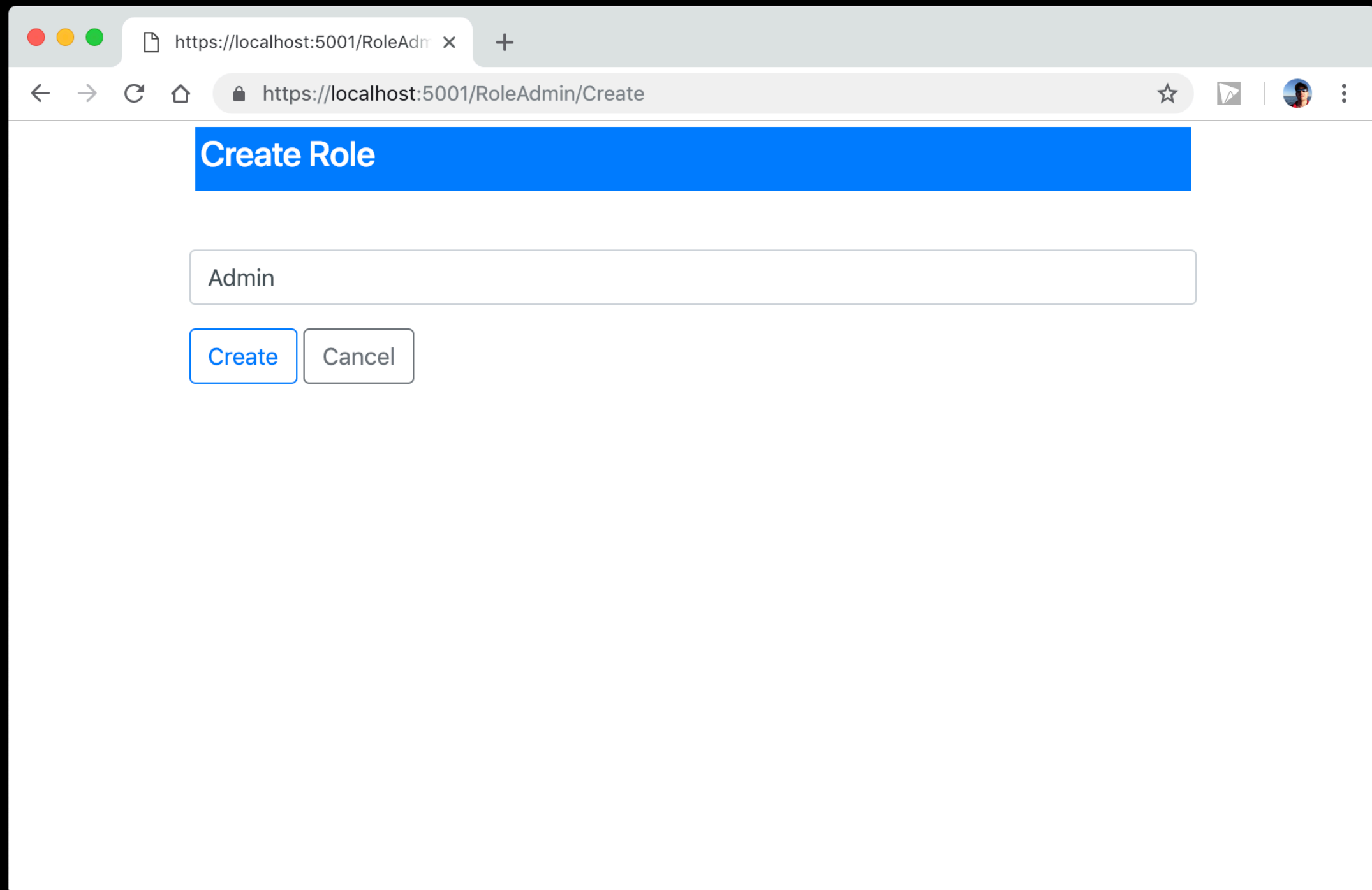
# Roles - Demo



# Roles - Demo

- Add RoleAdmin/Create controller method to handle http get requests
- Add RoleAdmin/Create view - all that is needed to create the role is a name
- Add RoleAdmin/Create controller method to handle http post requests

# Roles - Demo



A screenshot of a web browser window displaying a 'Create Role' form. The browser's address bar shows the URL `https://localhost:5001/RoleAdmin/Create`. The form has a blue header bar with the text 'Create Role'. Below this, there is a text input field containing the word 'Admin'. At the bottom of the form, there are two buttons: 'Create' and 'Cancel'.

https://localhost:5001/RoleAdmin/Create

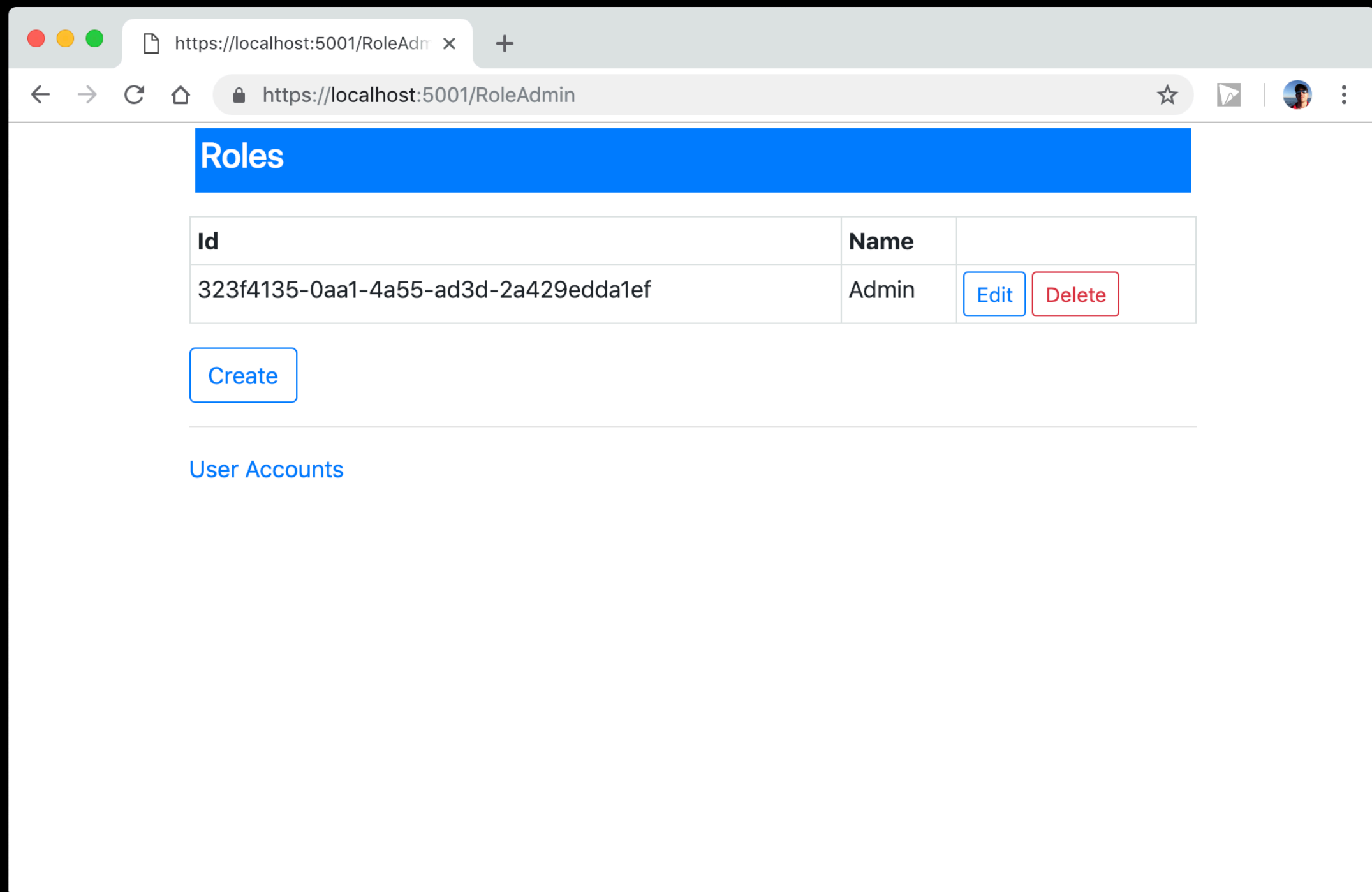
## Create Role

Admin

Create Cancel



# Roles - Demo



# Roles - Demo

- Add RoleAdmin/Delete controller method to respond to http post requests
- This will handle the deleting of roles
- Test in browser

# Roles

- Editing a role will involve listing the users that are members of the role and listing the user that are not members of the role
- Users that are members of the role can be removed from the role
- Users that are not members of the role can be added to the role

# Roles - Demo

- Update User View Model - add class to support the role edit view
- Create RoleAdmin/Edit controller method to handle http get requests
- Create RoleAdmin/Edit view

# Roles - Demo

https://localhost:5001/RoleAdmin x +

← → ↻ 🏠 🔒 https://localhost:5001/RoleAdmin/Edit/4ae4aa6f-5eae-4a6d-beb4-35bc683daeb7 ☆ 📄 👤 ⋮

Edit Role

Add To Admin Role

☐ joe

☐ alice

Remove From Admin Role

No Users Are Members

Save Done

# Roles - Demo

- Update User View Model
- Create RoleAdmin/Edit controller method to handle http post requests

# Roles - Demo

https://localhost:5001/RoleAdmin x +

← → ↻ 🏠 🔒 https://localhost:5001/RoleAdmin/Edit/4ae4aa6f-5eae-4a6d-beb4-35bc683daeb7 ☆ 📄 👤 ⋮

Edit Role

Add To Admin Role

☒ joe

☐ alice

Remove From Admin Role

No Users Are Members

Save Done

# Roles - Demo

The screenshot shows a web browser window with the following elements:

- Browser Tab:** `https://localhost:5001/RoleAdmin`
- Address Bar:** `https://localhost:5001/RoleAdmin/Edit/4ae4aa6f-5eae-4a6d-beb4-35bc683daeb7`
- Page Header:** Edit Role
- Add To Admin Role Section:**
  - Checkbox ☐ alice
- Remove From Admin Role Section:**
  - Checkbox ☐ joe
- Buttons:** Save, Done



# Roles - Demo

- The book example lists all users within of a role in the RoleAdmin/Index view
- The book also talks about seeding the database so you can avoid deleting the admin role / superuser account
- These are important features to look at

# Authorization

- Before we continue, I have 3 users: joe, alice, and bob
- The actual password for each user is “secret”
- alice is a member of the “Admin” role
- joe is a member of the “Moderate” role
- Bob is not a member of any role

# Authorization - Demo

- Update Account Controller - add controller method for logging out
- Update Account Controller - add controller method for access denied (user is authenticated, but unauthorized)
- Add view for access denied
- Update Admin Controller - only members of the Admin role can access
- Update RoleAdmin Controller - only members of the Admin role can access
- Update \_Layout view - add link to logout

# Authorization - Demo

- Update Home Controller - only allow members of Moderate role to access AddBlog and DeleteBlog controller methods
- Update Index view - show links to add, delete blogs to moderators only
- Update \_Layout view - display Login link to unauthenticated users, display Logout link to authenticated users

# Assignment

There are a few features that are remaining to implement

1. Only moderators should be allowed to delete a post
2. Only authenticated users should be allowed to create a post