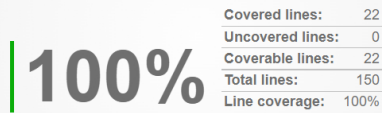


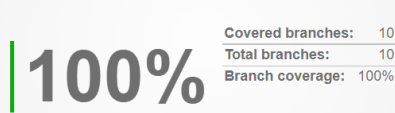
Information

Class: FuelQuoteApp_p1.Controllers.AccountController
Assembly: FuelQuoteApp_p1
File(s): C:\Users\Dell\OneDrive\Desktop\FuelQuoteApp_p1\FuelQuoteApp_p1\Controllers\AccountController.cs

Line coverage



Branch coverage



Method coverage

Method coverage is only available for sponsors.

[Upgrade to PRO version](#)

- .ctor(Microsoft.AspNetCore.Identity.User...
- RegisterDataValidation(FuelQuoteApp_p...

Metrics

Method	Branch coverage	Cyclomatic complexity	Line coverage
.ctor(...)	100%	1	100%
RegisterDataValidation(...)	100%	10	100%

File(s)

C:\Users\Dell\OneDrive\Desktop\FuelQuoteApp_p1\FuelQuoteApp_p1\Controllers\AccountController.cs

#	Line	Line coverage
	1	using FuelQuoteApp_p1.EntModels.Models;
	2	using FuelQuoteApp_p1.Models.Account;
	3	using Microsoft.AspNetCore.Authorization;
	4	using Microsoft.AspNetCore.Http;
	5	using Microsoft.AspNetCore.Identity;
	6	using Microsoft.AspNetCore.Mvc;
	7	using Newtonsoft.Json;
	8	using System;
	9	using System.Diagnostics.CodeAnalysis;
	10	using System.Text.RegularExpressions;
	11	using System.Threading.Tasks;
	12	
	13	namespace FuelQuoteApp_p1.Controllers
	14	{
	15	public class AccountController : Controller
	16	{
	17	private readonly UserManager<IdentityUser> userManager;
	18	private readonly SignInManager<IdentityUser> signInManager;
	19	
1	20	public AccountController(UserManager<IdentityUser> userManager, SignInManager<IdentityUser> signInManager)
1	21	{
1	22	this.userManager = userManager;
1	23	this.signInManager = signInManager;
1	24	}
	25	
	26	[HttpGet]
	27	[ExcludeFromCodeCoverage]
	28	public IActionResult Display()
	29	{
	30	return View();
	31	}
	32	
	33	[HttpGet]
	34	[AllowAnonymous]
	35	[ExcludeFromCodeCoverage]
	36	public IActionResult Login()
	37	{
	38	return View();
	39	}
	40	
	41	[HttpPost]
	42	[AllowAnonymous]
	43	[ExcludeFromCodeCoverage]
	44	public async Task<ActionResult> Login(Login model)
	45	{
	46	if (ModelState.IsValid)
	47	{
	48	
	49	var result = await signInManager.PasswordSignInAsync(model.Email, model.Password, model.RememberMe, fals
	50	
	51	if (result.Succeeded)
	52	{
	53	int userID = 1; //Hard Coding
	54	User userinfo = new User
	55	{
	56	Id = userID,
	57	Email = model.Email
--		}

```

58         };
59
60         HttpContext.Session.SetString("SessionUser", JsonConvert.SerializeObject(userinfo));
61
62         bool clientinfo = true; //Hard Coding
63         if (clientinfo)
64         {
65             return RedirectToAction("ClientDashBoard", "Client");
66         }
67         else
68         {
69             TempData["ClientProfileInfo"] = "Fill the profile details before requesting for a quote";
70             return RedirectToAction("ClientProfile", "Client");
71         }
72     }
73 }
74
75     ModelState.AddModelError(string.Empty, "Invalid Credentials!");
76 }
77     return View(model);
78 }
79
80     [HttpGet]
81     [AllowAnonymous]
82     [ExcludeFromCodeCoverage]
83     public IActionResult Register()
84     {
85         return View();
86     }
87
88     [HttpPost]
89     [AllowAnonymous]
90     [ExcludeFromCodeCoverage]
91     public async Task<IActionResult> Register(Register registerInfo)
92     {
93         if (ModelState.IsValid)
94         {
95             var user = new IdentityUser { UserName = registerInfo.Email, Email = registerInfo.Email };
96             var result = await userManager.CreateAsync(user, registerInfo.Password);
97
98             if (result.Succeeded)
99             {
100                 await signInManager.SignInAsync(user, isPersistent: false);
101                 User userinfo = new User
102                 {
103                     UserName = registerInfo.UserName,
104                     Email = registerInfo.Email
105                 };
106
107                 TempData["RegistrationSuccessful"] = "You're registered succesfully!";
108                 return RedirectToAction("Login", "Account");
109             }
110
111             foreach (var error in result.Errors)
112             {
113                 ModelState.AddModelError("", error.Description);
114             }
115         }
116         return View();
117     }
118 }
119
120     [HttpPost]
121     [ExcludeFromCodeCoverage]
122     public IActionResult Logout()
123     {
124         signInManager.SignOutAsync();
125         return RedirectToAction("Index", "Home");
126     }
127 }
128
129     public bool RegisterDataValidation(Register registerinfo)
130     {
131         bool flag = false;
132         if ((registerinfo.UserName.Length <= 50) && (registerinfo.UserName != String.Empty))
133         {
134             if ((Regex.IsMatch(registerinfo.Email, @"^[a-zA-Z0-9_!#$%&'()*+,-/=:?^`{|}~]+(\.[a-zA-Z0-9_!#$%&'()*+,-/=:?^`{|}~]+)*") &
135             {
136                 if (registerinfo.Password == registerinfo.ConfirmPassword)
137                 {
138                     flag = true;
139                 }
140             }
141         }
142         else
143         {
144             flag = false;
145         }
146
147         return flag;
148     }
149 }
150 }

```