

## Проект Phones. Авторизация, аутентификация

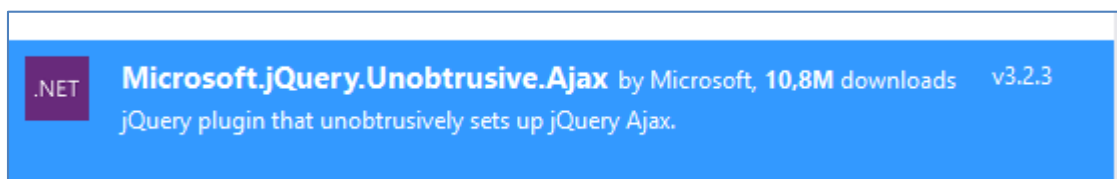
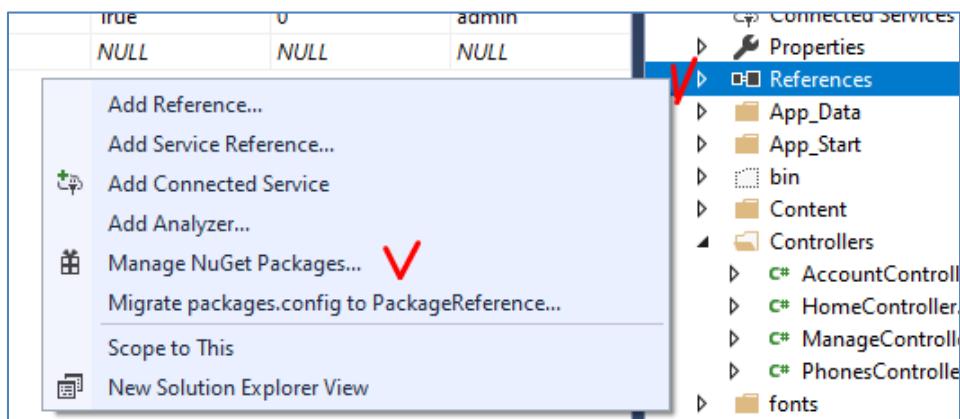
### Идея проекта:

Имеются данные о фирмах и телефонах. О фирмах известно: **название, страна**. О телефонах известно **название, цена, id фирмы**.

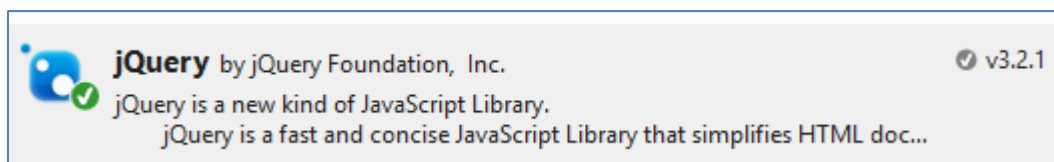
Данные хранятся в базе данных - информация о фирмах, телефонах.

## Часть 4. Блок Регистрация

Шаг 1. Добавим скрипт **ajax**



Update



Update

# Register, Login

В начальном исполнении блоки выглядят так (см. картинки)

The screenshot shows the 'Register' page of the 'PhonesPortal' application. The page has a dark header with 'PhonesPortal', 'Phones', and 'PhoneActions' links, and 'Register' and 'Log in' buttons. The main content area is titled 'Register.' and 'Create a new account.' It contains three input fields: 'Email', 'Password', and 'Confirm password'. A 'Register' button is at the bottom. The footer shows '© 2020 - My ASP.NET Application'.

The screenshot shows the 'Log in.' page of the 'PhonesPortal' application. The page has a dark header with 'PhonesPortal', 'Phones', and 'PhoneActions' links. The main content area is titled 'Log in.' and 'Use a local account to log in.' It contains two input fields: 'Email' and 'Password'. There is a 'Remember me?' checkbox and a 'Log in' button. A link 'Register as a new user' is at the bottom.

**Изменим код** и добавим при регистрации для входа блок **UserName**. Имя в дальнейшем будем использовать при добавлении комментариев-отзывов.

Изменим формы регистрации и login на сайте

Файл **Models/AccountViewModels.cs** - методы **RegisterViewModel**

```
roller.cs Login.cshtml AccountViewModels.cs Register.cshtml
o2020 PhonesKool2020.Models.LoginViewModel

4 references
public string Password { get; set; }

[Display(Name = "Remember me?")]
4 references
public bool RememberMe { get; set; }
}

3 references
public class RegisterViewModel
{
    [Required]
    [Display(Name = "UserName")]
    3 references
    public string UserName { get; set; }

    [Required]
    [EmailAddress]
    [Display(Name = "Email")]
    3 references
    public string Email { get; set; }

    [Required]
    [StringLength(100, ErrorMessage = "The {0} must be at least {2} characters long.", MinimumLength = 6)]
    [DataType(DataType.Password)]
    [Display(Name = "Password")]
    3 references
    public string Password { get; set; }

    [DataType(DataType.Password)]
    [Display(Name = "Confirm password")]
    [Compare("Password", ErrorMessage = "The password and confirmation password do not match.")]
    2 references
    public string ConfirmPassword { get; set; }
}
```

```

3 references
public class LoginViewModel
{
    [Required]
    [Display(Name = "UserName")]
    public string UserName { get; set; }

    [Required]
    [Display(Name = "Email")]
    [EmailAddress]
    public string Email { get; set; }

    [Required]
    [DataType(DataType.Password)]
    [Display(Name = "Password")]
    public string Password { get; set; }

    [Display(Name = "Remember me?")]
    public bool RememberMe { get; set; }
}
3 references

```

Файл **Controller/AccountController.cs**

Метод **Register**

```

// GET: /Account/Register
[AllowAnonymous]
public ActionResult Register()
{
    return View();
}

// POST: /Account/Register
[HttpPost]
[AllowAnonymous]
[ValidateAntiForgeryToken]
public async Task<ActionResult> Register(RegisterViewModel model)
{
    if (ModelState.IsValid)
    {
        var user = new ApplicationUser { UserName = model.UserName, Email = model.Email };
        var result = await UserManager.CreateAsync(user, model.Password);
        if (result.Succeeded)
        {
            await SignInManager.SignInAsync(user, isPersistent:false, rememberBrowser:false);

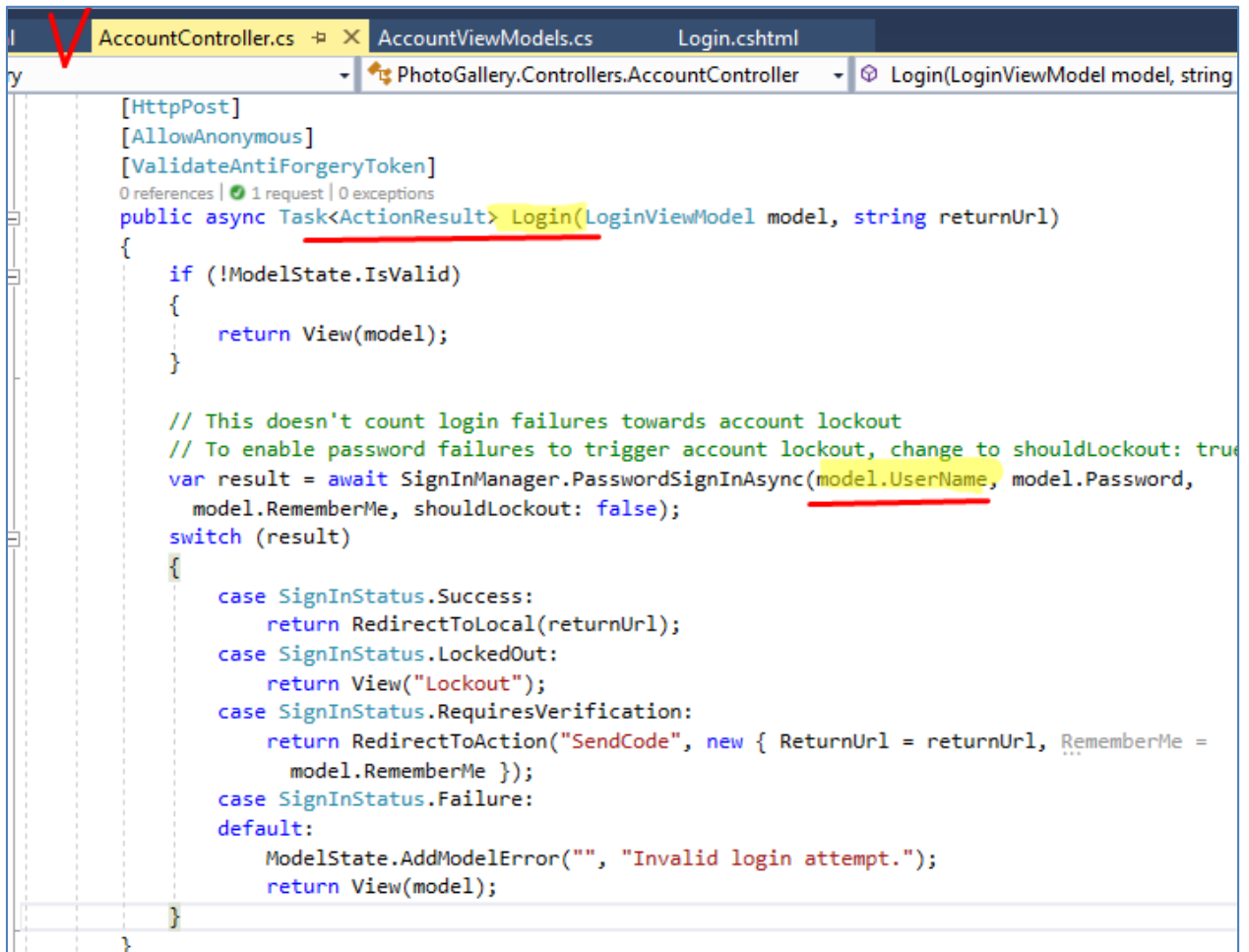
            // For more information on how to enable account confirmation and password reset please visit https://go.microsoft.com/fwlink/?LinkID=320755
            // Send an email with this link
            // string code = await UserManager.GenerateEmailConfirmationTokenAsync(user.Id);
            // var callbackUrl = Url.Action("ConfirmEmail", "Account", new { userId = user.Id, code = code }, protocol: Request.Url.Scheme);
            // await UserManager.SendEmailAsync(user.Id, "Confirm your account", "Please confirm your account by clicking < a href='{callbackUrl}'>here</a>");

            return RedirectToAction("Index", "Phones");
        }
        AddErrors(result);
    }

    // If we got this far, something failed, redisplay form
    return View(model);
}

```

## Метод Login



```
[HttpPost]
[AllowAnonymous]
[ValidateAntiForgeryToken]
0 references | 1 request | 0 exceptions
public async Task<ActionResult> Login(LoginViewModel model, string returnUrl)
{
    if (!ModelState.IsValid)
    {
        return View(model);
    }

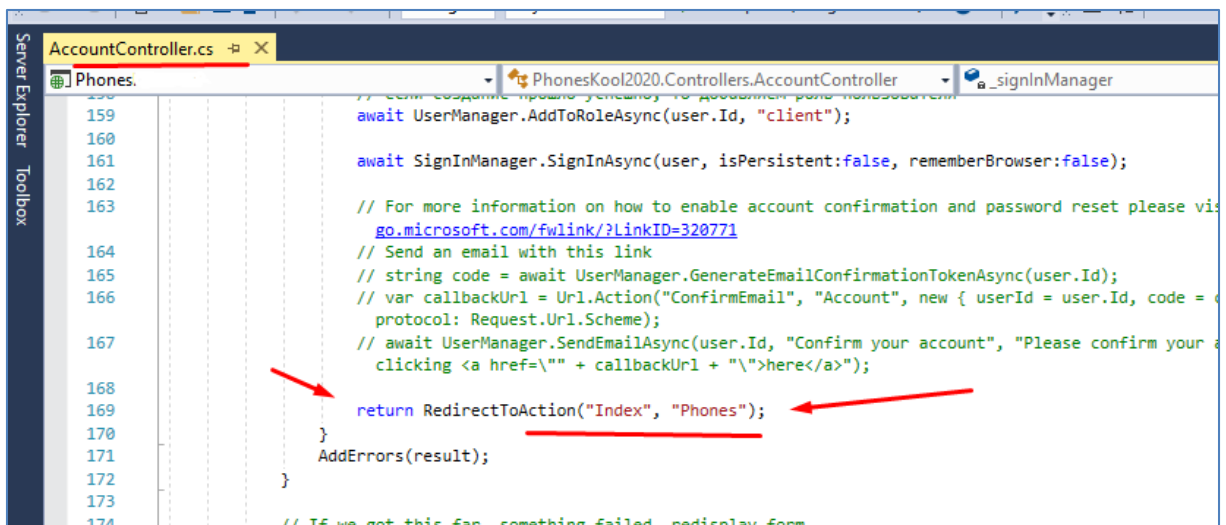
    // This doesn't count login failures towards account lockout
    // To enable password failures to trigger account lockout, change to shouldLockout: true
    var result = await SignInManager.PasswordSignInAsync(model.UserName, model.Password,
        model.RememberMe, shouldLockout: false);
    switch (result)
    {
        case SignInStatus.Success:
            return RedirectToLocal(returnUrl);
        case SignInStatus.LockedOut:
            return View("Lockout");
        case SignInStatus.RequiresVerification:
            return RedirectToAction("SendCode", new { ReturnUrl = returnUrl, RememberMe =
                model.RememberMe });
        case SignInStatus.Failure:
        default:
            ModelState.AddModelError("", "Invalid login attempt.");
            return View(model);
    }
}
```

Исправьте в коде переход после регистрации и входа с **Home** на **Phones**.

Просмотрите **весь код** и строки, где инструкция

`return RedirectToAction("Index", "Home");` **ИСПРАВЬТЕ**

`return RedirectToAction("Index", "Phones");`



```
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
// If we got this far, something failed, redisplay form

return RedirectToAction("Index", "Phones");
AddErrors(result);
}
```

## Представления Views

Файл **Views/Account/Register.cshtml**

Добавим блок для **UserName**

```
1 @model PhonesKool2020.Models.RegisterViewModel
2 @{
3     ViewBag.Title = "Register";
4 }
5
6 <h2>@ViewBag.Title.</h2>
7
8 @using (Html.BeginForm("Register", "Account", FormMethod.Post, new { @class = "form-horiz
9 {
10     @Html.AntiForgeryToken()
11     <h4>Create a new account.</h4>
12     <hr />
13     @Html.ValidationSummary("", new { @class = "text-danger" })
14
15     <div class="form-group">
16         @Html.LabelFor(m => m.UserName, new { @class = "col-md-2 control-label" })
17         <div class="col-md-10">
18             @Html.TextBoxFor(m => m.UserName, new { @class = "form-control" })
19         </div>
20     </div>
21     <div class="form-group">
22         @Html.LabelFor(m => m.Email, new { @class = "col-md-2 control-label" })
23         <div class="col-md-10">
24             @Html.TextBoxFor(m => m.Email, new { @class = "form-control" })
25         </div>
26     </div>
```

Файл **Views/Account/Login.cshtml**

```
1 <section id="loginForm">
2     @using (Html.BeginForm("Login", "Account", new { returnUrl = ViewBag.ReturnUrl }, FormMethod.Post,
3     { @class = "form-horizontal", role = "form" }))
4     {
5         @Html.AntiForgeryToken()
6         <h4>Use a local account to log in.</h4>
7         <hr />
8         @Html.ValidationSummary(true, "", new { @class = "text-danger" })
9
10        <div class="form-group">
11            @Html.LabelFor(m => m.UserName, new { @class = "col-md-2 control-label" })
12            <div class="col-md-10">
13                @Html.TextBoxFor(m => m.UserName, new { @class = "form-control" })
14                @Html.ValidationMessageFor(m => m.UserName, "", new { @class = "text-danger" })
15            </div>
16        </div>
17
18        <div class="form-group">
19            @Html.LabelFor(m => m.Email, new { @class = "col-md-2 control-label" })
20            <div class="col-md-10">
21                @Html.TextBoxFor(m => m.Email, new { @class = "form-control" })
22                @Html.ValidationMessageFor(m => m.Email, "", new { @class = "text-danger" })
23            </div>
24        </div>
25    }
26 </section>
```

Результат:

PhonesPortal Phones PhoneActions Register

## Register.

Create a new account.

UserName

Email

Password

Confirm password

Register

© 2020 - My ASP.NET Application

PhonesPortal Phones PhoneActions

## Register.

Create a new account.

UserName admin

Email admin@phones.ee

Password .....

Confirm password .....

Register

© 2020 - My ASP.NET Application

PhonesPortal Phones PhoneActions

## Log in.

Use a local account to log in.

UserName admin

Email admin@phones.ee

Password .....

☐ Remember me?

Log in

[Register as a new user](#)

© 2020 - My ASP.NET Application

**Зарегистрируйте 2х пользователей!**

**UserName: admin**

Email: [admin@phones.ee](mailto:admin@phones.ee)

Password: Admin123\*

**UserName: quest**

Email: [quest@phones.ee](mailto:quest@phones.ee)

Password: Quest123\*

**Остановите проект!**

## Роли в проекте

Добавим **роли для нашего проекта**: admin, client

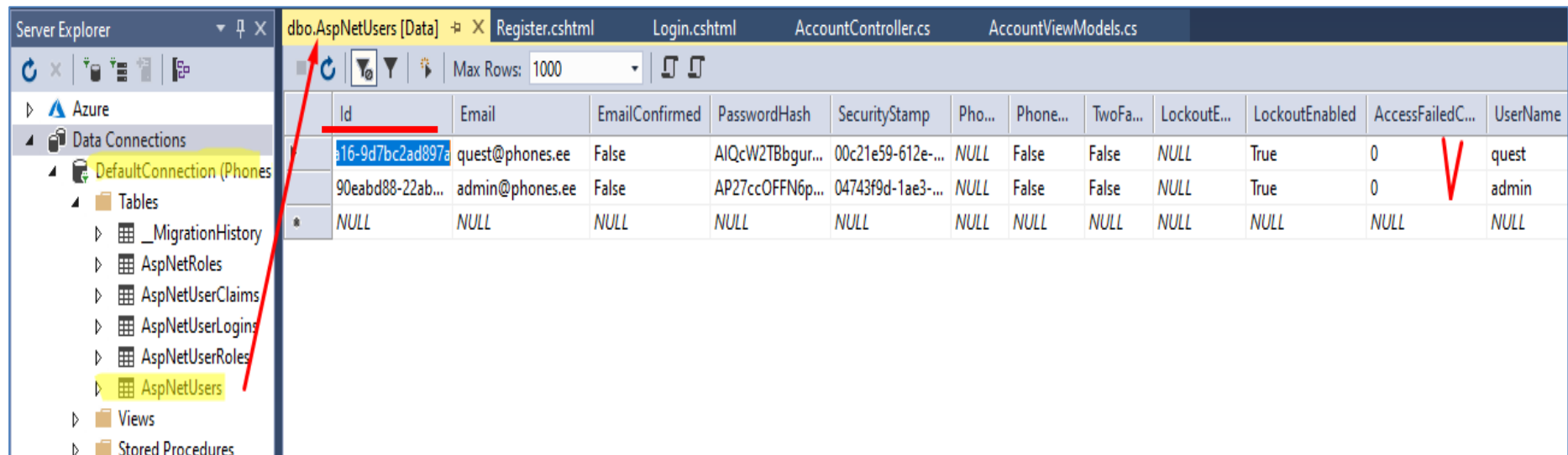
Настроим проект так, что все пользователи будут регистрироваться с ролью **client**

**Роль admin**: управление данными продуктов: create, edit, delete phones

**Роль client**: добавление комментариев

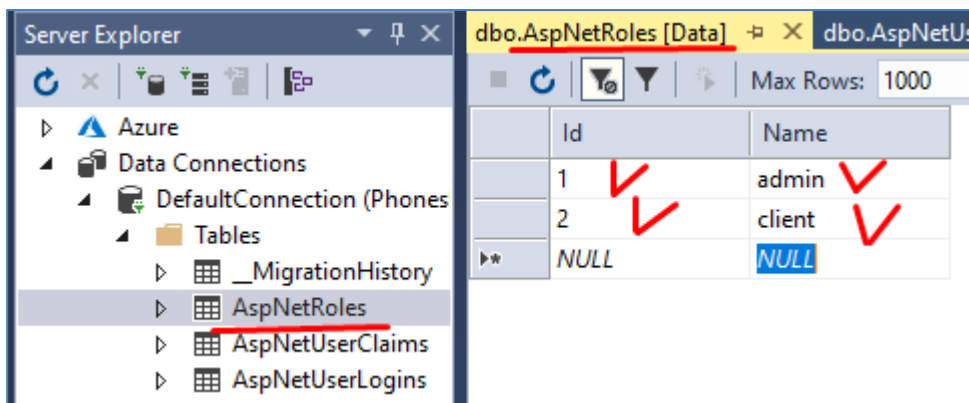
**Гость на сайте (незарегистрированный пользователь)**: просмотр информации продуктов(телефонов), чтение комментариев

Мы зарегистрировали **2х пользователей**: Name - **admin** – роль **admin**, Name - **quest** – роль **client**



Id	Email	EmailConfirmed	PasswordHash	SecurityStamp	Pho...	Phone...	TwoFa...	LockoutE...	LockoutEnabled	AccessFailedC...	UserName
16-9d7bc2ad897a	quest@phones.ee	False	AIQcW2TBgur...	00c21e59-612e-...	NULL	False	False	NULL	True	0	quest
90eabd88-22ab...	admin@phones.ee	False	AP27ccOFFN6p...	04743f9d-1ae3-...	NULL	False	False	NULL	True	0	admin
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

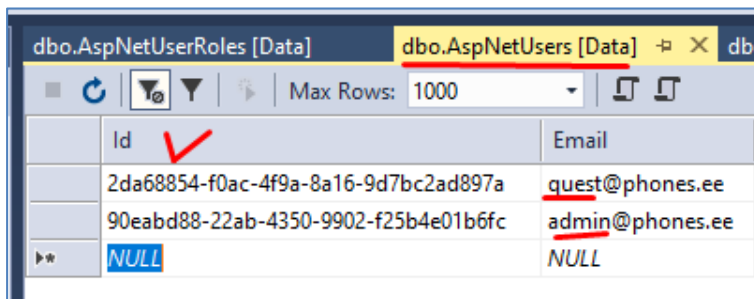
Откроем таблицу **AspNetRoles** и добавим роли для пользователей:



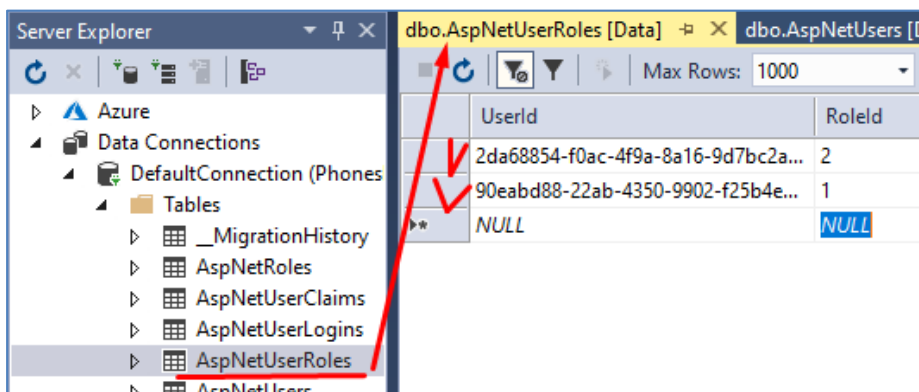
Id	Name
1	admin
2	client
NULL	NULL

Таблица **AspNetUserRoles** – добавим связь между пользователями и ролями

Из таблицы **AspNetUsers** скопируйте **Id** - длинный цифровой- буквенный код и вставьте в таблицу **AspNetUserRoles** – и поставьте код роли в проекте - соответственно **admin** код 1, **quest** – код 2



Id	Email
2da68854-f0ac-4f9a-8a16-9d7bc2ad897a	quest@phones.ee
90eabd88-22ab-4350-9902-f25b4e01b6fc	admin@phones.ee
NULL	NULL



UserId	RoleId
2da68854-f0ac-4f9a-8a16-9d7bc2a...	2
90eabd88-22ab-4350-9902-f25b4e...	1
NULL	NULL

Обновите все в БД **Server Explorer** -> **Refresh**.

**Закройте таблицы, закройте соединение**



Добавим код

Все зарегистрированные пользователи будут теперь иметь роль **client**

Файл **Controller/AccountController.cs**

Метод **Register** – добавьте блок кода – **ТОЛЬКО 2 СТРОКИ**

```
146 // POST: /Account/Register
147 [HttpPost]
148 [AllowAnonymous]
149 [ValidateAntiForgeryToken]
150 public async Task<ActionResult> Register(RegisterViewModel model)
151 {
152     if (ModelState.IsValid)
153     {
154         var user = new ApplicationUser { UserName = model.UserName, Email = model.Email };
155         var result = await UserManager.CreateAsync(user, model.Password);
156         if (result.Succeeded)
157         {
158             // если создание прошло успешно, то добавляем роль пользователя
159             await UserManager.AddToRoleAsync(user.Id, "client");
160
161             await SignInManager.SignInAsync(user, isPersistent:false, rememberBrowser:false);
162
163             // For more information on how to enable account confirmation and password reset please
164             // visit https://go.microsoft.com/fwlink/?LinkID=320771
165             // Send an email with this link
166             string code = await UserManager.GenerateEmailConfirmationTokenAsync(user.Id);
167             var callbackUrl = Url.Action("ConfirmEmail", "Account", new { userId = user.Id, code = code }, protocol: Request.Url.Scheme);
168             await UserManager.SendEmailAsync(user.Id, "Confirm your account", "Please confirm your account by clicking <a href=\"" + callbackUrl + "\">here</a>");
169
170             return RedirectToAction("Index", "Phones");
171         }
172         AddErrors(result);
173     }
174
175     // If we got this far, something failed, redisplay form
176     return View(model);
177 }
```

**//КОПИРОВАТЬ и ДОБАВИТЬ ТОЛЬКО 2 СТРОЧКИ ниже!!**

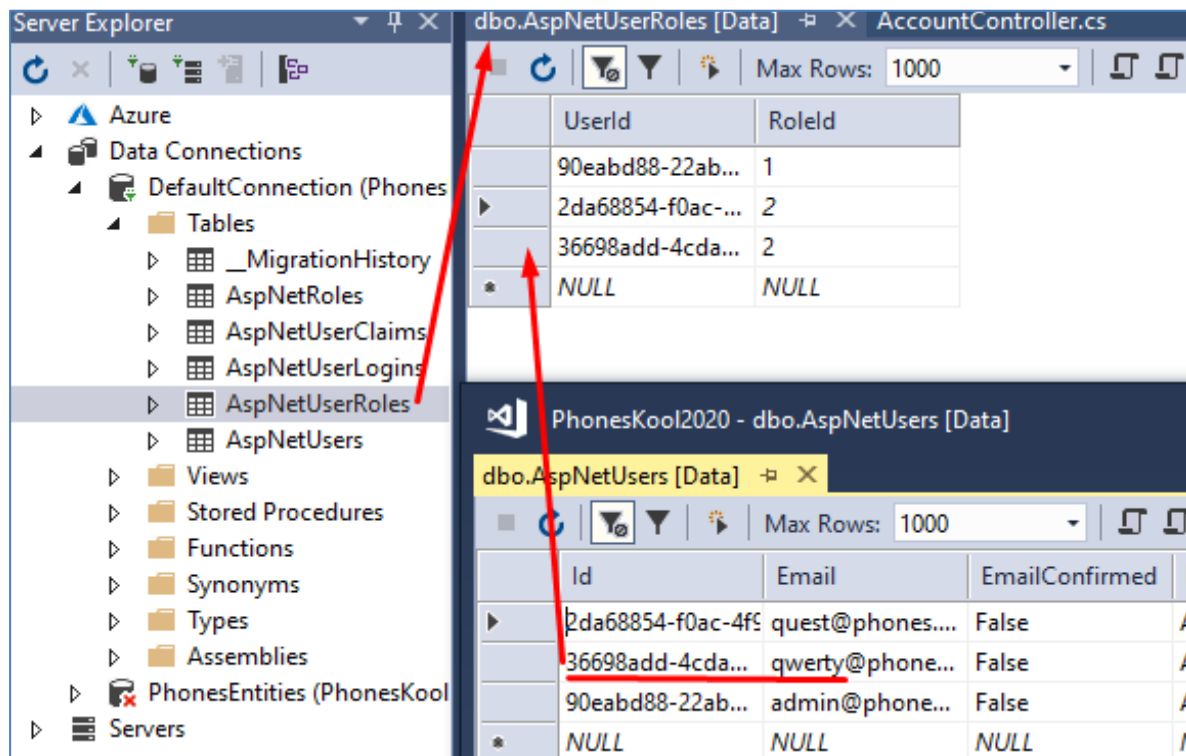
Если регистрация прошла успешно

```
// если создание прошло успешно, то добавляем роль пользователя
await UserManager.AddToRoleAsync(user.Id, "client");
```

Запустите проект, зарегистрируйте еще одного пользователя

UserName: **qwerty**  
Email: [qwerty@phones.ee](mailto:qwerty@phones.ee)  
Password: Qwerty123\*

Результат: **новый пользователь** получил роль **client**



UserId	RoleId
90eabd88-22ab...	1
2da68854-f0ac...	2
36698add-4cda...	2
NULL	NULL

Id	Email	EmailConfirmed
2da68854-f0ac-4f9...	quest@phones....	False
36698add-4cda...	qwerty@phone...	False
90eabd88-22ab...	admin@phone...	False
NULL	NULL	NULL

## Часть 5. Действия пользователей по ролям

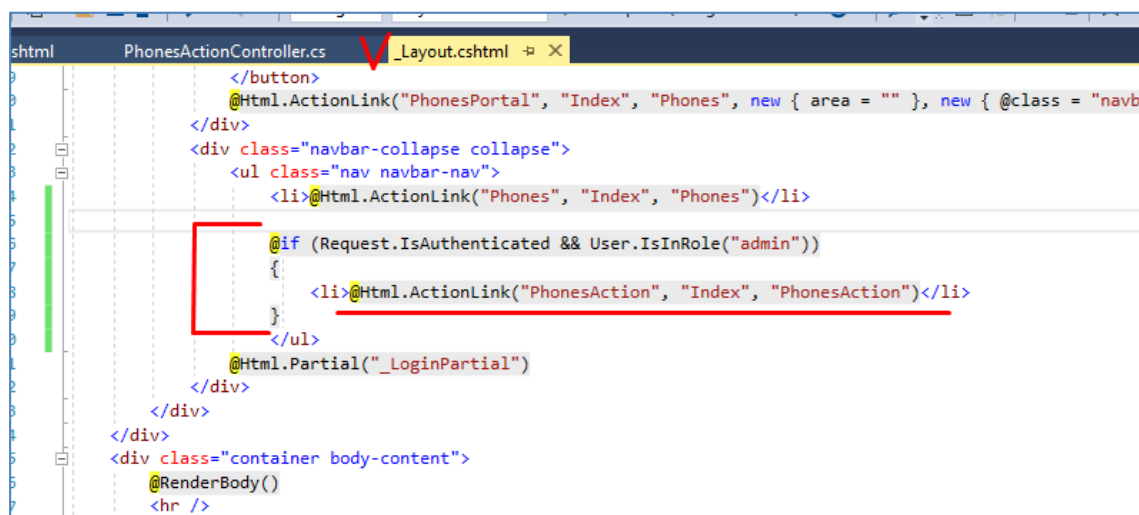
роль **admin** имеет право добавлять, редактировать, удалять записи

роль **client** – для него определим задачи позже! – **Добавлять комментарии**

**Не зарегистрированный пользователь** или **ГОСТЬ** на сайте видит только список продуктов-телефонов и детальные записи, читает комментарии

Файл **Views/Shared/\_Layout.cshtml**

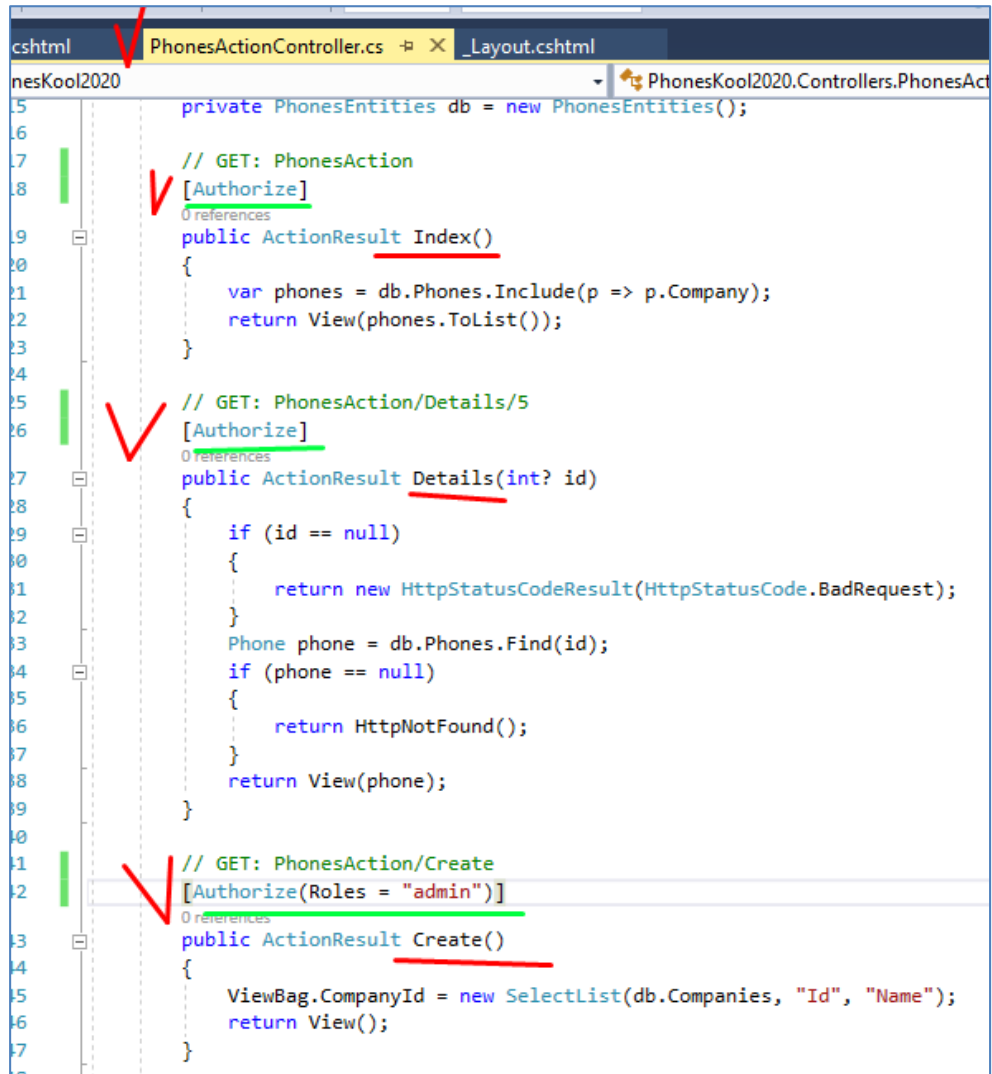
Откроем пункт меню только для зарегистрированного пользователя и роль admin - **admin**



```
1 </button>
2 @Html.ActionLink("PhonesPortal", "Index", "Phones", new { area = "" }, new { @class = "navb
3 </div>
4 <div class="navbar-collapse collapse">
5 <ul class="nav navbar-nav">
6 <li>@Html.ActionLink("Phones", "Index", "Phones")</li>
7
8 <li>@if (Request.IsAuthenticated && User.IsInRole("admin"))
9 {
10 <li>@Html.ActionLink("PhonesAction", "Index", "PhonesAction")</li>
11 }
12 </ul>
13 @Html.Partial("_LoginPartial")
14 </div>
15 </div>
16 <div class="container body-content">
17 @RenderBody()
18 <hr />
```

Файл **Controller/PhonesActionController.cs**

Разрешим методы **ИЛИ** для **зарегистрированного пользователя** - **ИЛИ** для роли **admin**

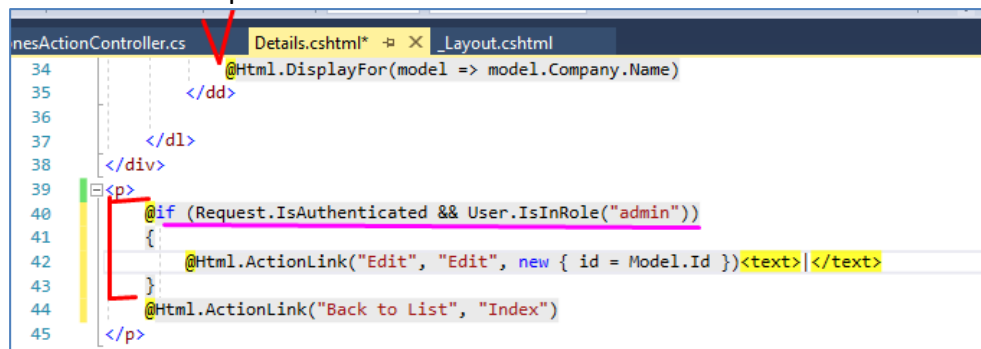


```
cshtml PhonesActionController.cs X _Layout.cshtml
nesKool2020
5 private PhonesEntities db = new PhonesEntities();
6
7 // GET: PhonesAction
8 [Authorize]
9 public ActionResult Index()
10 {
11     var phones = db.Phones.Include(p => p.Company);
12     return View(phones.ToList());
13 }
14
15 // GET: PhonesAction/Details/5
16 [Authorize]
17 public ActionResult Details(int? id)
18 {
19     if (id == null)
20     {
21         return new HttpStatusCodeResult(HttpStatusCode.BadRequest);
22     }
23     Phone phone = db.Phones.Find(id);
24     if (phone == null)
25     {
26         return HttpNotFound();
27     }
28     return View(phone);
29 }
30
31 // GET: PhonesAction/Create
32 [Authorize(Roles = "admin")]
33 public ActionResult Create()
34 {
35     ViewBag.CompanyId = new SelectList(db.Companies, "Id", "Name");
36     return View();
37 }
38
39 }
```

Если нужно закрыть какой-то текст в html файле из папки View, то ставим условие для вывода на экран. Для условия можно использовать && - И, вариант || Или

Например:

Файл **Views/PhonesAction/Details.cshtml** – исправьте код – для **авторизованного** пользователя **И** роль **admin**



```
nesActionController.cs Details.cshtml X _Layout.cshtml
34 @Html.DisplayFor(model => model.Company.Name)
35 </dd>
36
37 </dl>
38 </div>
39 <p>
40     @if (Request.IsAuthenticated && User.IsInRole("admin"))
41     {
42         @Html.ActionLink("Edit", "Edit", new { id = Model.Id })<text>|</text>
43     }
44     @Html.ActionLink("Back to List", "Index")
45 </p>
```

**Задание:** Протестируйте приложение для разных ролей