

Aula05 – Forms Authentication

Disciplina: Programação Web

Prof. Allbert Velleniche de Aquino Almeida

E-mail: allbert.almeida@fatec.sp.gov.br

Site: <http://www.allbert.com.br>



/allbert.almeida

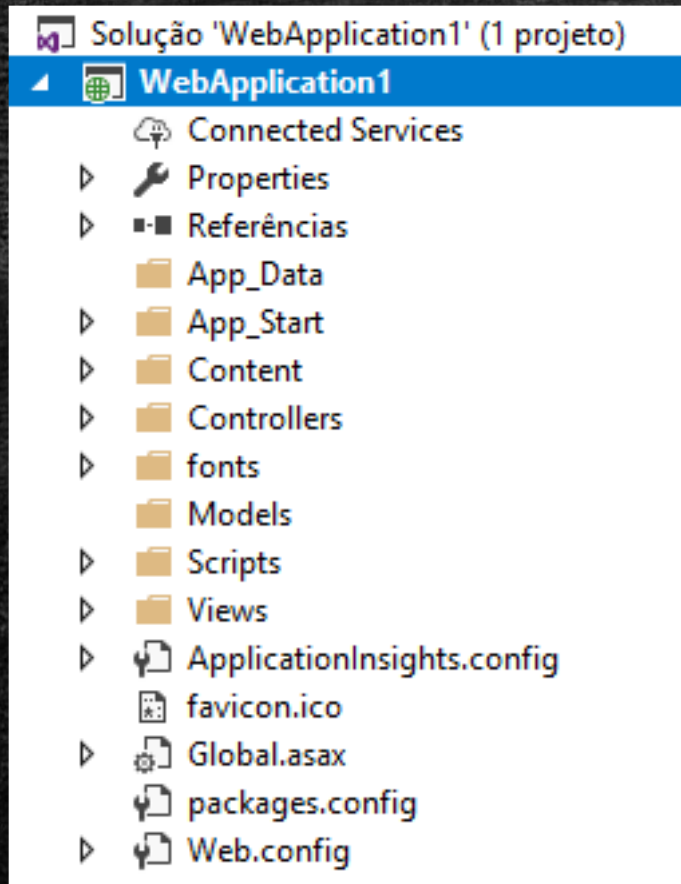
Objetivo

- O objetivo dessa aula é construir uma aplicação autenticando usuários com diferentes níveis de acesso usando Forms Authentication;

Criando uma nova aplicação

- Vamos abrir o Visual Studio;
- File >> New >> Project >> Aplicativo Web;
- Tipo de aplicação MVC;
- Change Authentication >> No Authentication

Exemplo de uma aplicação



Modelo

Usuario
Id
Nome
Email
Senha

UsuarioPerfil
Id
UsuarioId
PerfilId

Perfil
Id
Descricao



Criar uma classe Usuario

```
public class Usuario {
    [Key]
    public int Id { get; set; }
    [MaxLength(255)]
    public string Nome { get; set; }
    [MaxLength(100)]
    [EmailAddress]
    public string Email { get; set; }
    [Required]
    [DataType(DataType.Password)]
    public string Senha { get; set; }
    [Required]
    [DataType(DataType.Password)]
    [Compare("Senha")]
    public string ConfirmaSenha { get; set; }
}
```

Criar uma classe Perfil

```
public class Perfil
{
    public int Id { get; set; }
    [Required]
    [DisplayName("Descrição")]
    [MaxLength(255)]
    public string Descricao { get; set; }
}
```

Criar uma classe UsuarioPerfil

```
public class UsuarioPerfil {  
    public int Id { get; set; }  
    [DisplayName("Usuário")]  
    public int UsuarioId { get; set; }  
    [DisplayName("Perfil")]  
    public int PerfilId { get; set; }  
    public virtual Usuario Usuario { get; set; }  
    public virtual Perfil Perfil { get; set; }  
}
```


Criar uma classe Contexto

```
public class Contexto:DbContext
{
    public Contexto() : base(nameOrConnectionString:
"StringConexao") { }

    public DbSet<Usuario> Usuario { get; set; }
    public DbSet<Perfil> Perfil { get; set; }
    public DbSet<UsuarioPerfil> UsuarioPerfil { get; set; }
}
```

Adicionar 2 pacotes para solução

- EntityFramework
- MySql Data Entity 6.9.11

EntityFramework por Microsoft

v6.2.0



Entity Framework is Microsoft's recommended data access technology for new applications.



MySql.Data.Entity por Oracle

v6.10.6



Entity Framework 6.0 supported

Adicionar a String de Conexão no WebConfig

```
<connectionStrings>
```

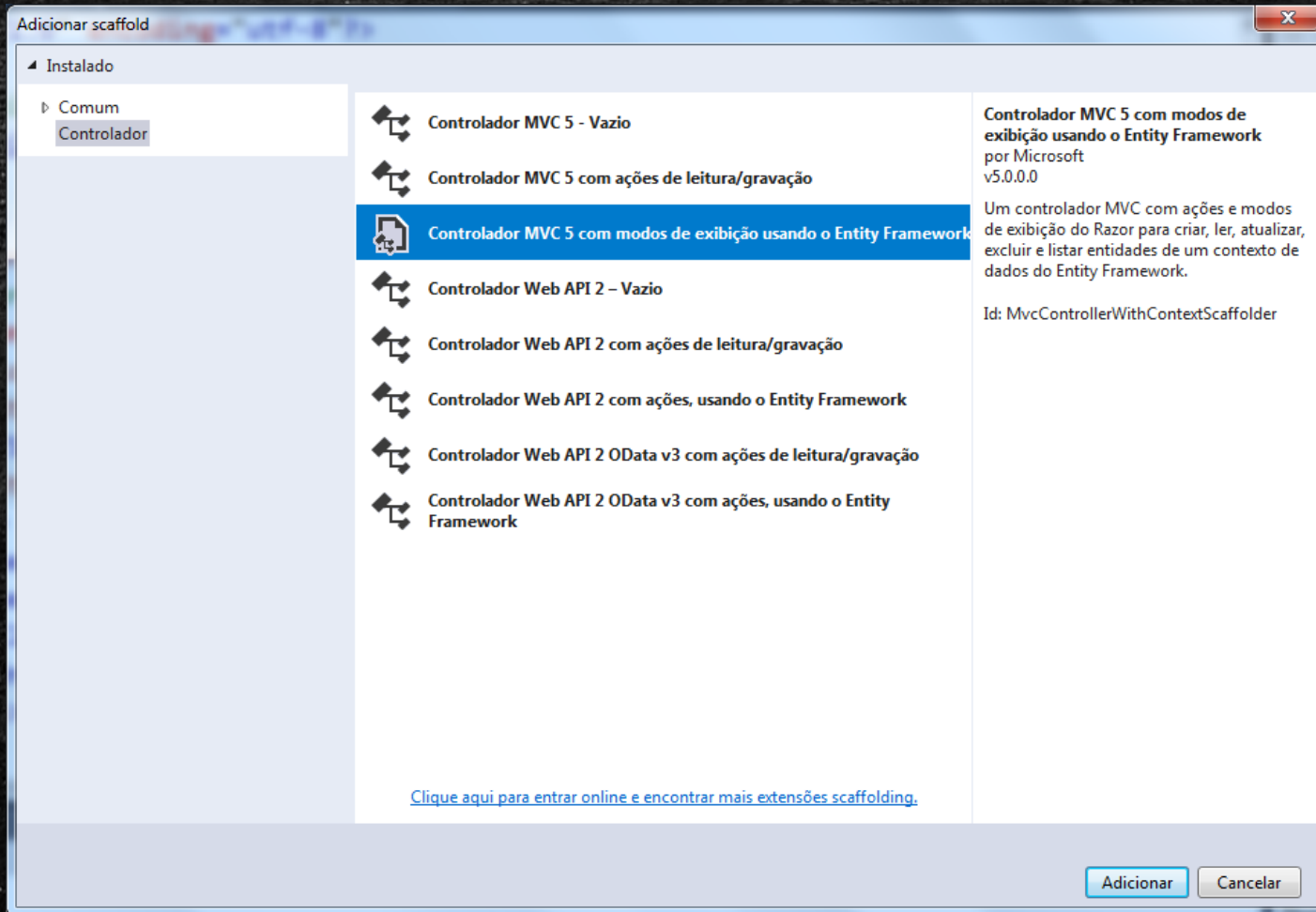
```
  <add name="StringConexao"  
    providerName="MySql.Data.MySqlClient"  
    connectionString="server=localhost;dat  
    abase=teste;uid=root;pwd='root'" />
```

```
</connectionStrings>
```

Compilar a solução

- Compilar a solução para que os modelos sejam validados e seja possível criar o “CRUD” da classe Usuario; Perfil; UsuarioPerfil

Adicionar Controller



Adicionar Controller

Adicionar controlador

Classe do modelo: Usuario (WebApplication1.Models)

Classe de contexto de dados: Contexto (WebApplication1.Models) +

Usar ações assíncronas do controlador

Modos de exibição:

Gerar modos de exibição

Bibliotecas de scripts de referência

Usar uma página de layout:

...

(deixe em branco se ele estiver definido em um arquivo Razor _viewstart)

Nome do controlador: UsuariosController

Adicionar Cancelar

Executar a solução

- Na URL entre com o caminho da aplicação +
/Perfils/
- Na URL entre com o caminho da aplicação +
/Usuarios/
- Cadastrar perfil e usuário;

Forms Authentication

- Adicionar no system.web do web.config:

```
<authentication mode="Forms">  
<forms loginUrl="~/Home/Login"  
timeout="2880" />  
</authentication>
```


HomeController

- Vamos adicionar na HomeController a action Login;

```
[HttpPost]
public ActionResult Login(string email, string senha, string returnUrl)
{
    Usuario usu = db.Usuario.Where(t => t.Email == email && t.Senha == senha).ToList().FirstOrDefault();
    if (usu != null) {
        string permissoes="";
        foreach(UsuarioPerfil p in usu.UsuarioPerfil)
            permissoes += p.Perfil.Descricao + ",";
        permissoes = permissoes.Substring(0, permissoes.Length - 1);
        FormsAuthentication.SetAuthCookie(usu.Nome, false);
        FormsAuthenticationTicket ticket = new FormsAuthenticationTicket(1, usu.Email, DateTime.Now, DateTime.Now.AddMinutes(30), false, permissoes);
        string hash = FormsAuthentication.Encrypt(ticket);
        HttpCookie cookie = new HttpCookie(FormsAuthentication.FormsCookieName, hash);
        if (ticket.IsPersistent)
            cookie.Expires = ticket.Expiration;
        Response.Cookies.Add(cookie);
        if (String.IsNullOrEmpty(returnUrl))
            return RedirectToAction("Index", "Perfis");
        else {
            var decodedUrl = Server.UrlDecode(returnUrl);
            if (Url.IsLocalUrl(decodedUrl))
                return Redirect(decodedUrl);
            else
                return RedirectToAction("Index", "Perfis");
        }
    }
    else {
        ModelState.AddModelError("", "Usuário/Senha inválidos");
        return View();
    }
}
```

View Login.cshtml

- Vamos adicionar a view login;

```
@using (Html.BeginForm()) {
    @Html.AntiForgeryToken()
    <div class="form-group rx-form-group">
        <div class="col-md-12 col-sm-12 form-group">
            <i class="fa fa-envelope select"></i>
            <input type="email" placeholder="E-mail" required="required" class="form-
control" id="email" value="" name="email"><br />
        </div>
        <div class="col-md-12 col-sm-12 form-group">
            <i class="fa fa-unlock-alt"></i>
            <input type="password" placeholder="Senha" required="required"
class="form-control" id="senha" value="" name="senha"><br />
        </div>
    </div>
    <div class="col-md-12 col-xs-12 form-group">
        <button class="btn btn-default" type="submit">Acessar</button>
    </div>
    @Html.ValidationSummary(true, "", new { @class = "text-danger" })
}
```

Adicionar autorização

- Na PerfilController vamos adicionar autorização. Somente usuários autorizados terão acesso a esse controller;
- Importante a autorização pode ser definida no controller inteiro ou somente nas actions;

[Authorize]

Compilar a solução novamente

- Compilar a solução novamente e verificar se conseguem acessar perfil;

Validar o Perfil

- Já vimos que somente usuários autorizados podem entrar na página, agora vamos ver como controlar o perfil;

Global.asax

- Vamos adicionar o método;

```
protected void Application_PostAuthenticateRequest(Object sender, EventArgs e)
{
    var authCookie =
    HttpContext.Current.Request.Cookies[FormsAuthentication.FormsCookieName];
    if (authCookie != null) {
        FormsAuthenticationTicket authTicket =
        FormsAuthentication.Decrypt(authCookie.Value);
        if (authTicket != null && !authTicket.Expired) {
            var roles = authTicket.UserData.Split(',');
            HttpContext.Current.User = new
            System.Security.Principal.GenericPrincipal(new FormsIdentity(authTicket),
            roles);
        }
    }
}
```

Adicionar perfil na autorização

- Na PerfilController vamos adicionar autorização. Somente usuários autorizados com roles definido terão acesso a esse controller;
- Importante a autorização pode ser definida no controller inteiro ou somente nas actions;

```
[Authorize(Roles = "Admin")]
```

Compilar a solução novamente

- Compilar a solução novamente e verificar se conseguem acessar perfil;