

DotnetIdentity

Generated by Doxygen 1.9.4



<b>1 Todo List</b>	<b>1</b>
<b>2 Namespace Index</b>	<b>3</b>
2.1 Namespace List . . . . .	3
<b>3 Hierarchical Index</b>	<b>5</b>
3.1 Class Hierarchy . . . . .	5
<b>4 Class Index</b>	<b>7</b>
4.1 Class List . . . . .	7
<b>5 Namespace Documentation</b>	<b>9</b>
5.1 DotNetIdentity Namespace Reference . . . . .	9
5.2 DotNetIdentity.Controllers Namespace Reference . . . . .	9
5.3 DotNetIdentity.Data Namespace Reference . . . . .	9
5.4 DotNetIdentity.Helpers Namespace Reference . . . . .	10
5.5 DotNetIdentity.Helpers.TagHelpers Namespace Reference . . . . .	10
5.6 DotNetIdentity.IdentitySettings Namespace Reference . . . . .	10
5.7 DotNetIdentity.IdentitySettings.Requirements Namespace Reference . . . . .	10
5.8 DotNetIdentity.IdentitySettings.Validators Namespace Reference . . . . .	11
5.9 DotNetIdentity.Models Namespace Reference . . . . .	11
5.9.1 Enumeration Type Documentation . . . . .	11
5.9.1.1 Department . . . . .	11
5.9.1.2 Gender . . . . .	11
5.9.1.3 TwoFactorType . . . . .	13
5.10 DotNetIdentity.Models.BusinessModels Namespace Reference . . . . .	13
5.11 DotNetIdentity.Models.DataModels Namespace Reference . . . . .	13
5.12 DotNetIdentity.Models.Identity Namespace Reference . . . . .	14
5.13 DotNetIdentity.Models.ViewModels Namespace Reference . . . . .	14
5.14 DotNetIdentity.Services Namespace Reference . . . . .	15
5.15 DotNetIdentity.Services.SettingsService Namespace Reference . . . . .	15
<b>6 Class Documentation</b>	<b>17</b>
6.1 DotNetIdentity.Controllers.AdminController Class Reference . . . . .	17
6.1.1 Detailed Description . . . . .	18
6.1.2 Constructor & Destructor Documentation . . . . .	19
6.1.2.1 AdminController() . . . . .	19
6.1.3 Member Function Documentation . . . . .	19
6.1.3.1 AuditLogs() . . . . .	19
6.1.3.2 DeleteRole() . . . . .	19
6.1.3.3 DeleteUser() . . . . .	20
6.1.3.4 DisableLdapLogin() . . . . .	21
6.1.3.5 DisableMfaForce() . . . . .	21
6.1.3.6 DisableUser() . . . . .	22

6.1.3.7 EditUser() [1/2]	22
6.1.3.8 EditUser() [2/2]	23
6.1.3.9 EnableLdapLogin()	24
6.1.3.10 EnableMfaForce()	25
6.1.3.11 EnableUser()	25
6.1.3.12 ErrorLogs()	26
6.1.3.13 GetAppLogs()	26
6.1.3.14 GetAuditLogs()	26
6.1.3.15 GetErrorLogs()	27
6.1.3.16 GetUsers()	27
6.1.3.17 Index()	28
6.1.3.18 NewUser() [1/2]	28
6.1.3.19 NewUser() [2/2]	28
6.1.3.20 Roles()	29
6.1.3.21 SystemLogs()	29
6.1.3.22 UpsertRole() [1/2]	30
6.1.3.23 UpsertRole() [2/2]	30
6.1.3.24 Users()	31
6.2 DotNetIdentity.Data.AppDbContext Class Reference	31
6.2.1 Detailed Description	32
6.2.2 Constructor & Destructor Documentation	32
6.2.2.1 AppDbContext()	32
6.2.3 Member Function Documentation	32
6.2.3.1 OnModelCreating()	33
6.2.4 Property Documentation	34
6.2.4.1 AppLogs	34
6.2.4.2 AppSessionCache	34
6.2.4.3 AppSettings	34
6.3 DotNetIdentity.Models.DataModels.ApplicationSettings Class Reference	34
6.3.1 Detailed Description	35
6.3.2 Property Documentation	35
6.3.2.1 Name	35
6.3.2.2 Type	35
6.3.2.3 Value	36
6.4 DotNetIdentity.Models.DataModels.AppLogs Class Reference	36
6.4.1 Detailed Description	36
6.4.2 Property Documentation	36
6.4.2.1 Exception	37
6.4.2.2 id	37
6.4.2.3 Level	37
6.4.2.4 Message	37
6.4.2.5 MessageTemplate	38

6.4.2.6 Properties . . . . .	38
6.4.2.7 Timestamp . . . . .	38
6.5 DotNetIdentity.Models.Identity.AppRole Class Reference . . . . .	38
6.5.1 Detailed Description . . . . .	39
6.5.2 Property Documentation . . . . .	39
6.5.2.1 CreatedOn . . . . .	39
6.6 DotNetIdentity.Services.SettingsService.AppSettingsBase Class Reference . . . . .	39
6.6.1 Detailed Description . . . . .	40
6.6.2 Constructor & Destructor Documentation . . . . .	40
6.6.2.1 AppSettingsBase() . . . . .	40
6.6.3 Member Function Documentation . . . . .	40
6.6.3.1 Load() . . . . .	40
6.6.3.2 Save() . . . . .	41
6.7 DotNetIdentity.Models.Identity.AppUser Class Reference . . . . .	41
6.7.1 Detailed Description . . . . .	42
6.7.2 Property Documentation . . . . .	42
6.7.2.1 BirthDay . . . . .	43
6.7.2.2 CreatedOn . . . . .	43
6.7.2.3 Department . . . . .	43
6.7.2.4 FirstName . . . . .	43
6.7.2.5 Gender . . . . .	44
6.7.2.6 IsEnabled . . . . .	44
6.7.2.7 IsLdapLogin . . . . .	44
6.7.2.8 IsMfaForce . . . . .	44
6.7.2.9 LastName . . . . .	45
6.7.2.10 ProfilePicture . . . . .	45
6.7.2.11 TwoFactorType . . . . .	45
6.8 DotNetIdentity.Models.ViewModels.AssignRoleViewModel Class Reference . . . . .	45
6.8.1 Detailed Description . . . . .	46
6.8.2 Property Documentation . . . . .	46
6.8.2.1 IsAssigned . . . . .	46
6.8.2.2 RoleId . . . . .	46
6.8.2.3 RoleName . . . . .	46
6.9 DotNetIdentity.Models.ViewModels.ChangePasswordViewModel Class Reference . . . . .	47
6.9.1 Detailed Description . . . . .	47
6.9.2 Property Documentation . . . . .	47
6.9.2.1 NewPassword . . . . .	47
6.9.2.2 NewPasswordConfirm . . . . .	47
6.9.2.3 Password . . . . .	48
6.10 DotNetIdentity.Controllers.ClaimBasedController Class Reference . . . . .	48
6.10.1 Detailed Description . . . . .	49
6.10.2 Member Function Documentation . . . . .	49

6.10.2.1 Employee()	49
6.10.2.2 FreeTrial()	49
6.10.2.3 HR()	49
6.10.2.4 Software()	50
6.10.2.5 Violence()	50
6.11 DotNetIdentity.IdentitySettings.ClaimsTransformation Class Reference	50
6.11.1 Detailed Description	51
6.11.2 Constructor & Destructor Documentation	51
6.11.2.1 ClaimsTransformation()	51
6.11.3 Member Function Documentation	51
6.11.3.1 TransformAsync()	51
6.12 DotNetIdentity.Controllers.CultureController Class Reference	52
6.12.1 Detailed Description	53
6.12.2 Member Function Documentation	53
6.12.2.1 SetCulture()	53
6.13 DotNetIdentity.Models.ViewModels.EditUserViewModel Class Reference	54
6.13.1 Detailed Description	54
6.13.2 Property Documentation	54
6.13.2.1 Department	55
6.13.2.2 Email	55
6.13.2.3 FirstName	55
6.13.2.4 Gender	55
6.13.2.5 Id	56
6.13.2.6 LastName	56
6.13.2.7 PhoneNumber	56
6.13.2.8 Roles	56
6.13.2.9 UserName	57
6.14 DotNetIdentity.Helpers.EmailHelper Class Reference	57
6.14.1 Detailed Description	57
6.14.2 Constructor & Destructor Documentation	57
6.14.2.1 EmailHelper()	57
6.14.3 Member Function Documentation	58
6.14.3.1 SendAsync()	58
6.15 DotNetIdentity.Models.BusinessModels.EmailModel Class Reference	58
6.15.1 Detailed Description	59
6.15.2 Property Documentation	59
6.15.2.1 Body	59
6.15.2.2 Subject	59
6.15.2.3 To	60
6.16 DotNetIdentity.IdentitySettings.ErrorDescriber Class Reference	60
6.16.1 Detailed Description	60
6.16.2 Member Function Documentation	61

6.16.2.1 PasswordContainsUsername()	61
6.16.2.2 UserNameContainsEmail()	61
6.16.2.3 UserNameLength()	61
6.17 DotNetIdentity.Controllers.ErrorsController Class Reference	62
6.17.1 Detailed Description	62
6.17.2 Member Function Documentation	62
6.17.2.1 ErrorCd()	62
6.17.2.2 ErrorEx()	63
6.18 DotNetIdentity.Models.ViewModels.ErrorViewModel Class Reference	64
6.18.1 Detailed Description	64
6.18.2 Property Documentation	64
6.18.2.1 RequestId	64
6.18.2.2 ShowRequestId	65
6.19 DotNetIdentity.Models.ViewModels.ForgotPasswordViewModel Class Reference	65
6.19.1 Detailed Description	65
6.19.2 Property Documentation	65
6.19.2.1 Email	65
6.20 DotNetIdentity.IdentitySettings.Requirements.FreeTrialExpireHandler Class Reference	66
6.20.1 Detailed Description	66
6.20.2 Member Function Documentation	66
6.20.2.1 HandleRequirementAsync()	66
6.21 DotNetIdentity.IdentitySettings.Requirements.FreeTrialExpireRequirement Class Reference	67
6.21.1 Detailed Description	67
6.22 DotNetIdentity.Models.BusinessModels.GlobalSettings Class Reference	68
6.22.1 Detailed Description	68
6.22.2 Property Documentation	68
6.22.2.1 ApplicationName	69
6.22.2.2 SessionCookieExpiration	69
6.22.2.3 SessionTimeoutRedirAfter	69
6.22.2.4 SessionTimeoutWarnAfter	69
6.23 DotNetIdentity.Controllers.HomeController Class Reference	70
6.23.1 Detailed Description	70
6.23.2 Constructor & Destructor Documentation	70
6.23.2.1 HomeController()	70
6.23.3 Member Function Documentation	71
6.23.3.1 AccessDenied()	71
6.23.3.2 Index()	71
6.24 DotNetIdentity.Services.SettingsService.ISettingsService Interface Reference	71
6.24.1 Detailed Description	72
6.24.2 Member Function Documentation	72
6.24.2.1 Save()	72
6.24.3 Property Documentation	72

6.24.3.1 Global	72
6.24.3.2 Mail	73
6.25 DotNetIdentity.Models.BusinessModels.MailSettings Class Reference	73
6.25.1 Detailed Description	74
6.25.2 Property Documentation	74
6.25.2.1 Password	74
6.25.2.2 SmtptFromAddress	74
6.25.2.3 SmtptPort	74
6.25.2.4 SmtptServer	75
6.25.2.5 SmtptUseTls	75
6.25.2.6 Username	75
6.26 DotNetIdentity.IdentitySettings.Requirements.MinimumAgeHandler Class Reference	75
6.26.1 Detailed Description	76
6.26.2 Member Function Documentation	76
6.26.2.1 HandleRequirementAsync()	76
6.27 DotNetIdentity.IdentitySettings.Requirements.MinimumAgeRequirement Class Reference	77
6.27.1 Detailed Description	77
6.27.2 Constructor & Destructor Documentation	77
6.27.2.1 MinimumAgeRequirement()	77
6.27.3 Property Documentation	78
6.27.3.1 MinimumAge	78
6.28 DotNetIdentity.Models.ViewModels.NewUserModel Class Reference	78
6.28.1 Detailed Description	79
6.28.2 Property Documentation	79
6.28.2.1 Department	79
6.28.2.2 Email	79
6.28.2.3 FirstName	79
6.28.2.4 Gender	80
6.28.2.5 IsLdapLogin	80
6.28.2.6 IsMfaForce	80
6.28.2.7 LastName	80
6.28.2.8 Password	81
6.28.2.9 Roles	81
6.28.2.10 UserName	81
6.29 DotNetIdentity.IdentitySettings.Validators.PasswordValidator Class Reference	82
6.29.1 Detailed Description	82
6.29.2 Member Function Documentation	82
6.29.2.1 ValidateAsync()	82
6.30 DotNetIdentity.Models.ViewModels.ResetPasswordViewModel Class Reference	83
6.30.1 Detailed Description	84
6.30.2 Property Documentation	84
6.30.2.1 Password	84



6.30.2.2 Token . . . . .	84
6.30.2.3 UserId . . . . .	84
6.31 DotNetIdentity.Controllers.RoleBasedController Class Reference . . . . .	85
6.31.1 Detailed Description . . . . .	85
6.31.2 Member Function Documentation . . . . .	85
6.31.2.1 Articles() . . . . .	85
6.31.2.2 Dashboard() . . . . .	86
6.31.2.3 Profile() . . . . .	86
6.31.2.4 Users() . . . . .	86
6.32 DotNetIdentity.Models.DataModels.SessionCache Class Reference . . . . .	86
6.32.1 Detailed Description . . . . .	87
6.32.2 Property Documentation . . . . .	87
6.32.2.1 AbsoluteExpiration . . . . .	87
6.32.2.2 ExpiresAtTime . . . . .	87
6.32.2.3 id . . . . .	87
6.32.2.4 SlidingExpirationInSeconds . . . . .	88
6.32.2.5 Value . . . . .	88
6.33 DotNetIdentity.Services.SettingsService.SettingsService Class Reference . . . . .	88
6.33.1 Detailed Description . . . . .	89
6.33.2 Constructor & Destructor Documentation . . . . .	89
6.33.2.1 SettingsService() . . . . .	89
6.33.3 Member Function Documentation . . . . .	89
6.33.3.1 Save() . . . . .	89
6.33.4 Property Documentation . . . . .	90
6.33.4.1 Global . . . . .	90
6.33.4.2 Mail . . . . .	90
6.34 DotNetIdentity.Models.ViewModels.SignInViewModel Class Reference . . . . .	90
6.34.1 Detailed Description . . . . .	91
6.34.2 Property Documentation . . . . .	91
6.34.2.1 Password . . . . .	91
6.34.2.2 RememberMe . . . . .	91
6.34.2.3 UserName . . . . .	92
6.35 DotNetIdentity.Models.ViewModels.SignUpViewModel Class Reference . . . . .	92
6.35.1 Detailed Description . . . . .	92
6.35.2 Property Documentation . . . . .	92
6.35.2.1 BirthDay . . . . .	93
6.35.2.2 Department . . . . .	93
6.35.2.3 Email . . . . .	93
6.35.2.4 FirstName . . . . .	93
6.35.2.5 Gender . . . . .	94
6.35.2.6 LastName . . . . .	94
6.35.2.7 Password . . . . .	94

6.35.2.8 UserName	94
6.36 DotNetIdentity.Models.ViewModels.StatusCodesModel Class Reference	95
6.36.1 Detailed Description	95
6.36.2 Property Documentation	95
6.36.2.1 ErrorMessage	95
6.36.2.2 RouteOfException	95
6.36.2.3 StatusCode	96
6.37 DotNetIdentity.IdentitySettings.TwoFactorAuthenticationService Class Reference	96
6.37.1 Detailed Description	96
6.37.2 Constructor & Destructor Documentation	96
6.37.2.1 TwoFactorAuthenticationService()	96
6.37.3 Member Function Documentation	97
6.37.3.1 GenerateQrCodeUri()	97
6.38 DotNetIdentity.Models.ViewModels.TwoFactorAuthenticatorViewModel Class Reference	97
6.38.1 Detailed Description	98
6.38.2 Property Documentation	98
6.38.2.1 AuthenticationUri	98
6.38.2.2 SharedKey	98
6.38.2.3 VerificationCode	98
6.39 DotNetIdentity.Models.ViewModels.TwoFactorLoginVieWModel Class Reference	99
6.39.1 Detailed Description	99
6.39.2 Property Documentation	99
6.39.2.1 IsRecoveryCode	99
6.39.2.2 TwoFactorType	99
6.39.2.3 VerificationCode	100
6.40 DotNetIdentity.Models.ViewModels.TwoFactorTypeViewModel Class Reference	100
6.40.1 Detailed Description	100
6.40.2 Property Documentation	100
6.40.2.1 TwoFactorType	100
6.41 DotNetIdentity.Models.ViewModels.UpdateProfileViewModel Class Reference	101
6.41.1 Detailed Description	101
6.41.2 Property Documentation	101
6.41.2.1 BirthDay	101
6.41.2.2 Email	102
6.41.2.3 Gender	102
6.41.2.4 PhoneNumber	102
6.41.2.5 ProfilePicture	102
6.41.2.6 UserName	103
6.42 DotNetIdentity.Models.ViewModels.UpsertRoleViewModel Class Reference	103
6.42.1 Detailed Description	103
6.42.2 Property Documentation	103
6.42.2.1 Id	103

6.42.2.2 Name	104
6.43 DotNetIdentity.Controllers.UserController Class Reference	104
6.43.1 Detailed Description	105
6.43.2 Constructor & Destructor Documentation	106
6.43.2.1 UserController()	106
6.43.3 Member Function Documentation	106
6.43.3.1 ChangePassword() [1/2]	106
6.43.3.2 ChangePassword() [2/2]	107
6.43.3.3 ConfirmEmail()	107
6.43.3.4 debugUserInfo()	108
6.43.3.5 EnforceTwoFactorAuthenticator() [1/2]	108
6.43.3.6 EnforceTwoFactorAuthenticator() [2/2]	109
6.43.3.7 EnforceTwoFactorResult()	109
6.43.3.8 ForgotPassword() [1/2]	110
6.43.3.9 ForgotPassword() [2/2]	110
6.43.3.10 Login() [1/2]	111
6.43.3.11 Login() [2/2]	112
6.43.3.12 Logout()	112
6.43.3.13 ping()	113
6.43.3.14 Profile() [1/2]	113
6.43.3.15 Profile() [2/2]	113
6.43.3.16 Register() [1/2]	114
6.43.3.17 Register() [2/2]	115
6.43.3.18 ResetPassword() [1/2]	116
6.43.3.19 ResetPassword() [2/2]	116
6.43.3.20 TwoFactorAuthenticator() [1/2]	117
6.43.3.21 TwoFactorAuthenticator() [2/2]	117
6.43.3.22 TwoFactorLogin() [1/2]	118
6.43.3.23 TwoFactorLogin() [2/2]	119
6.43.3.24 TwoFactorType() [1/2]	120
6.43.3.25 TwoFactorType() [2/2]	120
6.44 DotNetIdentity.Helpers.TagHelpers.UserProfileImageTagHelper Class Reference	121
6.44.1 Detailed Description	121
6.44.2 Constructor & Destructor Documentation	121
6.44.2.1 UserProfileImageTagHelper()	121
6.44.3 Member Function Documentation	122
6.44.3.1 ProcessAsync()	122
6.44.4 Property Documentation	122
6.44.4.1 UserId	123
6.45 DotNetIdentity.Helpers.TagHelpers.UserRolesTagHelper Class Reference	123
6.45.1 Detailed Description	123
6.45.2 Constructor & Destructor Documentation	124

6.45.2.1 UserRolesTagHelper() . . . . .	124
6.45.3 Member Function Documentation . . . . .	124
6.45.3.1 ProcessAsync() . . . . .	124
6.45.4 Property Documentation . . . . .	125
6.45.4.1 UserId . . . . .	125
6.46 DotNetIdentity.IdentitySettings.Validators.UserValidator Class Reference . . . . .	125
6.46.1 Detailed Description . . . . .	126
6.46.2 Member Function Documentation . . . . .	126
6.46.2.1 ValidateAsync() . . . . .	126
<b>Index</b>	<b>129</b>

# Chapter 1

## Todo List

**Class** [DotNetIdentity.Controllers.AdminController](#)

add translation

**Class** [DotNetIdentity.Controllers.ErrorsController](#)

add translation

**Class** [DotNetIdentity.Controllers.UserController](#)

add translation

**Member** [DotNetIdentity.Controllers.UserController.Login](#) ([SignInViewModel](#) viewModel)

Implement Ldap Login

**Class** [DotNetIdentity.Services.SettingsService.SettingsService](#)

add ldap settings class



## Chapter 2

# Namespace Index

### 2.1 Namespace List

Here is a list of all documented namespaces with brief descriptions:

<a href="#">DotNetIdentity</a>	9
<a href="#">DotNetIdentity.Controllers</a>	9
<a href="#">DotNetIdentity.Data</a>	9
<a href="#">DotNetIdentity.Helpers</a>	10
<a href="#">DotNetIdentity.Helpers.TagHelpers</a>	10
<a href="#">DotNetIdentity.IdentitySettings</a>	10
<a href="#">DotNetIdentity.IdentitySettings.Requirements</a>	10
<a href="#">DotNetIdentity.IdentitySettings.Validators</a>	11
<a href="#">DotNetIdentity.Models</a>	11
<a href="#">DotNetIdentity.Models.BusinessModels</a>	13
<a href="#">DotNetIdentity.Models.DataModels</a>	13
<a href="#">DotNetIdentity.Models.Identity</a>	14
<a href="#">DotNetIdentity.Models.ViewModels</a>	14
<a href="#">DotNetIdentity.Services</a>	15
<a href="#">DotNetIdentity.Services.SettingsService</a>	15





## Chapter 3

# Hierarchical Index

### 3.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

DotNetIdentity.Models.DataModels.ApplicationSettings . . . . .	34
DotNetIdentity.Models.DataModels.AppLogs . . . . .	36
DotNetIdentity.Services.SettingsService.AppSettingsBase . . . . .	39
DotNetIdentity.Models.BusinessModels.GlobalSettings . . . . .	68
DotNetIdentity.Models.BusinessModels.MailSettings . . . . .	73
DotNetIdentity.Models.ViewModels.AssignRoleViewModel . . . . .	45
AuthorizationHandler	
DotNetIdentity.IdentitySettings.Requirements.FreeTrialExpireHandler . . . . .	66
DotNetIdentity.IdentitySettings.Requirements.MinimumAgeHandler . . . . .	75
DotNetIdentity.Models.ViewModels.ChangePasswordViewModel . . . . .	47
Controller	
DotNetIdentity.Controllers.AdminController . . . . .	17
DotNetIdentity.Controllers.ClaimBasedController . . . . .	48
DotNetIdentity.Controllers.CultureController . . . . .	52
DotNetIdentity.Controllers.ErrorsController . . . . .	62
DotNetIdentity.Controllers.HomeController . . . . .	70
DotNetIdentity.Controllers.RoleBasedController . . . . .	85
DotNetIdentity.Controllers.UserController . . . . .	104
DotNetIdentity.Models.ViewModels.EditUserViewModel . . . . .	54
DotNetIdentity.Helpers.EmailHelper . . . . .	57
DotNetIdentity.Models.BusinessModels.EmailModel . . . . .	58
DotNetIdentity.Models.ViewModels.ErrorViewModel . . . . .	64
DotNetIdentity.Models.ViewModels.ForgotPasswordViewModel . . . . .	65
IAuthorizationRequirement	
DotNetIdentity.IdentitySettings.Requirements.FreeTrialExpireRequirement . . . . .	67
DotNetIdentity.IdentitySettings.Requirements.MinimumAgeRequirement . . . . .	77
IClaimsTransformation	
DotNetIdentity.IdentitySettings.ClaimsTransformation . . . . .	50
IdentityDbContext	
DotNetIdentity.Data.AppDbContext . . . . .	31
IdentityErrorDescriber	
DotNetIdentity.IdentitySettings.ErrorDescriber . . . . .	60
IdentityRole	
DotNetIdentity.Models.Identity.AppRole . . . . .	38
IdentityUser	

DotNetIdentity.Models.Identity.AppUser . . . . .	41
IPasswordValidator	
DotNetIdentity.IdentitySettings.Validators.PasswordValidator . . . . .	82
DotNetIdentity.Services.SettingsService.ISettingsService . . . . .	71
DotNetIdentity.Services.SettingsService.SettingsService . . . . .	88
IValidator	
DotNetIdentity.IdentitySettings.Validators.UserValidator . . . . .	125
DotNetIdentity.Models.ViewModels.NewUserModel . . . . .	78
DotNetIdentity.Models.ViewModels.ResetPasswordViewModel . . . . .	83
DotNetIdentity.Models.DataModels.SessionCache . . . . .	86
DotNetIdentity.Models.ViewModels.SignInViewModel . . . . .	90
DotNetIdentity.Models.ViewModels.SignUpViewModel . . . . .	92
DotNetIdentity.Models.ViewModels.StatusCodesModel . . . . .	95
TagHelper	
DotNetIdentity.Helpers.TagHelpers.UserProfileImageTagHelper . . . . .	121
DotNetIdentity.Helpers.TagHelpers.UserRolesTagHelper . . . . .	123
DotNetIdentity.IdentitySettings.TwoFactorAuthenticationService . . . . .	96
DotNetIdentity.Models.ViewModels.TwoFactorAuthenticatorViewModel . . . . .	97
DotNetIdentity.Models.ViewModels.TwoFactorLoginViewWModel . . . . .	99
DotNetIdentity.Models.ViewModels.TwoFactorTypeViewModel . . . . .	100
DotNetIdentity.Models.ViewModels.UpdateProfileViewModel . . . . .	101
DotNetIdentity.Models.ViewModels.UpsertRoleViewModel . . . . .	103

## Chapter 4

# Class Index

### 4.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

<a href="#">DotNetIdentity.Controllers.AdminController</a>	
Controller Class for Admin views . . . . .	17
<a href="#">DotNetIdentity.Data.AppDbContext</a>	
Application Database context class . . . . .	31
<a href="#">DotNetIdentity.Models.DataModels.ApplicationSettings</a>	
class define an Settings object . . . . .	34
<a href="#">DotNetIdentity.Models.DataModels.AppLogs</a>	
class to define ann Log object . . . . .	36
<a href="#">DotNetIdentity.Models.Identity.AppRole</a>	
class to define a role object . . . . .	38
<a href="#">DotNetIdentity.Services.SettingsService.AppSettingsBase</a>	
base class for Settings classes . . . . .	39
<a href="#">DotNetIdentity.Models.Identity.AppUser</a>	
class to define a user object . . . . .	41
<a href="#">DotNetIdentity.Models.ViewModels.AssignRoleViewModel</a>	
model for UserEditView . . . . .	45
<a href="#">DotNetIdentity.Models.ViewModels.ChangePasswordViewModel</a>	
model for ChangePassword view . . . . .	47
<a href="#">DotNetIdentity.Controllers.ClaimBasedController</a>	
Controller to test ClaimBased Authorization . . . . .	48
<a href="#">DotNetIdentity.IdentitySettings.ClaimsTransformation</a>	
class to define own claims . . . . .	50
<a href="#">DotNetIdentity.Controllers.CultureController</a>	
controller for culture settings . . . . .	52
<a href="#">DotNetIdentity.Models.ViewModels.EditUserViewModel</a>	
model for EditUser view . . . . .	54
<a href="#">DotNetIdentity.Helpers.EmailHelper</a>	
class for email methods . . . . .	57
<a href="#">DotNetIdentity.Models.BusinessModels.EmailModel</a>	
class to define a email . . . . .	58
<a href="#">DotNetIdentity.IdentitySettings.ErrorDescriber</a>	
class to override IdentityErrorDescriber IdentotyError messages . . . . .	60
<a href="#">DotNetIdentity.Controllers.ErrorsController</a>	
Controller for error views . . . . .	62
<a href="#">DotNetIdentity.Models.ViewModels.ErrorViewModel</a>	
model for displaying errors . . . . .	64

<a href="#">DotNetIdentity.Models.ViewModels.ForgotPasswordViewModel</a>	
model for ForgotPassword view	65
<a href="#">DotNetIdentity.IdentitySettings.Requirements.FreeTrialExpireHandler</a>	
class to define own Identity requirements type of <a href="#">FreeTrialExpireRequirement</a>	66
<a href="#">DotNetIdentity.IdentitySettings.Requirements.FreeTrialExpireRequirement</a>	
class to define own Identity requirements type of <a href="#">IAuthorizationRequirement</a>	67
<a href="#">DotNetIdentity.Models.BusinessModels.GlobalSettings</a>	
class to define a global settings object	68
<a href="#">DotNetIdentity.Controllers.HomeController</a>	
Controller for Home views	70
<a href="#">DotNetIdentity.Services.SettingsService.ISettingsService</a>	
Interface class for <a href="#">SettingsService</a>	71
<a href="#">DotNetIdentity.Models.BusinessModels.MailSettings</a>	
class to define a mail settings object	73
<a href="#">DotNetIdentity.IdentitySettings.Requirements.MinimumAgeHandler</a>	
class implementing <a href="#">AuthorizationHandler</a> of type <a href="#">MinimumAgeRequirement</a>	75
<a href="#">DotNetIdentity.IdentitySettings.Requirements.MinimumAgeRequirement</a>	
class to define own Identity requirements type of <a href="#">IAuthorizationRequirement</a>	77
<a href="#">DotNetIdentity.Models.ViewModels.NewUserModel</a>	
model for NewUser view	78
<a href="#">DotNetIdentity.IdentitySettings.Validators.PasswordValidator</a>	
class to validate a password	82
<a href="#">DotNetIdentity.Models.ViewModels.ResetPasswordViewModel</a>	
model for ResetPassword view	83
<a href="#">DotNetIdentity.Controllers.RoleBasedController</a>	
Controller to test RoleBased Authorization	85
<a href="#">DotNetIdentity.Models.DataModels.SessionCache</a>	
class to define a <a href="#">SessionCache</a> object	86
<a href="#">DotNetIdentity.Services.SettingsService.SettingsService</a>	
class for settings service	88
<a href="#">DotNetIdentity.Models.ViewModels.SignInViewModel</a>	
model for Login view	90
<a href="#">DotNetIdentity.Models.ViewModels.SignUpViewModel</a>	
model for Register view	92
<a href="#">DotNetIdentity.Models.ViewModels.StatusCodesModel</a>	
model for Error Statuscodes	95
<a href="#">DotNetIdentity.IdentitySettings.TwoFactorAuthenticationService</a>	
class providing 2fa auth services	96
<a href="#">DotNetIdentity.Models.ViewModels.TwoFactorAuthenticatorViewModel</a>	
model for TwoFactorAuthenticator view	97
<a href="#">DotNetIdentity.Models.ViewModels.TwoFactorLoginViewWModel</a>	
the model for TwoFactorLogin view	99
<a href="#">DotNetIdentity.Models.ViewModels.TwoFactorTypeViewModel</a>	
model for TwoFactorType view	100
<a href="#">DotNetIdentity.Models.ViewModels.UpdateProfileViewModel</a>	
model for UpdateProfile view	101
<a href="#">DotNetIdentity.Models.ViewModels.UpsertRoleViewModel</a>	
model for the UpsertRole view	103
<a href="#">DotNetIdentity.Controllers.UserController</a>	
Controller Class for User views	104
<a href="#">DotNetIdentity.Helpers.TagHelpers.UserProfileImageTagHelper</a>	
html taghelper class	121
<a href="#">DotNetIdentity.Helpers.TagHelpers.UserRolesTagHelper</a>	
html tag helper class	123
<a href="#">DotNetIdentity.IdentitySettings.Validators.UserValidator</a>	
class to validate IdentityUser	125

## Chapter 5

# Namespace Documentation

### 5.1 DotNetIdentity Namespace Reference

### 5.2 DotNetIdentity.Controllers Namespace Reference

#### Classes

- class [AdminController](#)  
*Controller Class for Admin views*
- class [ClaimBasedController](#)  
*Controller to test ClaimBased Authorization*
- class [CultureController](#)  
*controller for culture settings*
- class [ErrorsController](#)  
*Controller for error views*
- class [HomeController](#)  
*Controller for Home views*
- class [RoleBasedController](#)  
*Controller to test RoleBased Authorization*
- class [UserController](#)  
*Controller Class for User views*

### 5.3 DotNetIdentity.Data Namespace Reference

#### Classes

- class [AppDbContext](#)  
*Application Database context class*

## 5.4 DotNetIdentity.Helpers Namespace Reference

### Classes

- class [EmailHelper](#)  
*class for email methods*

## 5.5 DotNetIdentity.Helpers.TagHelpers Namespace Reference

### Classes

- class [UserProfileImageTagHelper](#)  
*html taghelper class*
- class [UserRolesTagHelper](#)  
*html tag helper class*

## 5.6 DotNetIdentity.IdentitySettings Namespace Reference

### Classes

- class [ClaimsTransformation](#)  
*class to define own claims*
- class [ErrorDescriber](#)  
*class to override IdentityErrorDescriber IdentityError messages*
- class [TwoFactorAuthenticationService](#)  
*class providing 2fa auth services*

## 5.7 DotNetIdentity.IdentitySettings.Requirements Namespace Reference

### Classes

- class [FreeTrialExpireHandler](#)  
*class to define own Identity requirements type of [FreeTrialExpireRequirement](#)*
- class [FreeTrialExpireRequirement](#)  
*class to define own Identity requirements type of [IAuthorizationRequirement](#)*
- class [MinimumAgeHandler](#)  
*class implementing AuthorizationHandler of type [MinimumAgeRequirement](#)*
- class [MinimumAgeRequirement](#)  
*class to define own Identity requirements type of [IAuthorizationRequirement](#)*

## 5.8 DotNetIdentity.IdentitySettings.Validators Namespace Reference

### Classes

- class [PasswordValidator](#)  
*class to validate a password*
- class [UserValidator](#)  
*class to validate IdentityUser*

## 5.9 DotNetIdentity.Models Namespace Reference

### Enumerations

- enum [Gender](#) { [Unknown](#) , [Male](#) , **Female** }  
*Gender enum*
- enum [Department](#) { [Software](#) , [HR](#) , [Accounting](#) , **Management** }  
*Department enum*
- enum [TwoFactorType](#) { [None](#) , [Email](#) , **Authenticator** }  
*TwoFactorType enum*

### 5.9.1 Enumeration Type Documentation

#### 5.9.1.1 Department

```
enum DotNetIdentity.Models.Department
```

Department enum

##### Enumerator

Software	summary>Department Softwaresummary>Department HR
HR	summary>Department Accounting
Accounting	summary>Department Management

Definition at line 16 of file [Enums.cs](#).

```
00016      {  
00018          Software,  
00020          HR,  
00022          Accounting,  
00024          Management }  
00024      }
```

#### 5.9.1.2 Gender

```
enum DotNetIdentity.Models.Gender
```

Gender enum



## Enumerator

Unknown	summary>gender unknownsummary>gender male
Male	summary>gender female

Definition at line 6 of file [Enums.cs](#).

```
00006      {
00008          Unknown,
00010          Male,
00012          Female }
```

## 5.9.1.3 TwoFactorType

```
enum DotNetIdentity.Models.TwoFactorType
```

TwoFactorType enum

## Enumerator

None	summary>2fa type nonesummary>2fa type email
Email	summary>2fa type Authenticator app

Definition at line 28 of file [Enums.cs](#).

```
00028      {
00030          None,
00032          Email,
00034          Authenticator
00035      }
```

## 5.10 DotNetIdentity.Models.BusinessModels Namespace Reference

## Classes

- class [EmailModel](#)  
*class to define a email*
- class [GlobalSettings](#)  
*class to define a global settings object*
- class [MailSettings](#)  
*class to define a ail settings object*

## 5.11 DotNetIdentity.Models.DataModels Namespace Reference

## Classes

- class [ApplicationSettings](#)  
*class define an Settings object*
- class [AppLogs](#)  
*class to define ann Log object*
- class [SessionCache](#)  
*class to define a [SessionCache](#) object*

## 5.12 DotNetIdentity.Models.Identity Namespace Reference

### Classes

- class [AppRole](#)  
*class to define a role object*
- class [AppUser](#)  
*class to define a user object*

## 5.13 DotNetIdentity.Models.ViewModels Namespace Reference

### Classes

- class [AssignRoleViewModel](#)  
*model for UserEditView*
- class [ChangePasswordViewModel](#)  
*model for ChangePassword view*
- class [EditUserViewModel](#)  
*model for EditUser view*
- class [ErrorViewModel](#)  
*model for displaying errors*
- class [ForgotPasswordViewModel](#)  
*model for ForgotPassword view*
- class [NewUserModel](#)  
*model for NewUser view*
- class [ResetPasswordViewModel](#)  
*model for ResetPassword view*
- class [SignInViewModel](#)  
*model for Login view*
- class [SignUpViewModel](#)  
*model for Register view*
- class [StatusCodesModel](#)  
*model for Error Statuscodes*
- class [TwoFactorAuthenticatorViewModel](#)  
*model for TwoFactorAuthenticator view*
- class [TwoFactorLoginViewWMModel](#)  
*the model for TwoFactorLogin view*
- class [TwoFactorTypeViewModel](#)  
*model for TwoFactorType view*
- class [UpdateProfileViewModel](#)  
*model for UpdateProfile view*
- class [UpsertRoleViewModel](#)  
*model for the UpsertRole view*

## 5.14 DotNetIdentity.Services Namespace Reference

## 5.15 DotNetIdentity.Services.SettingService Namespace Reference

### Classes

- class [AppSettingsBase](#)  
*base class for Settings classes*
- interface [ISettingService](#)  
*Interface class for [SettingService](#)*
- class [SettingService](#)  
*class for settings service*



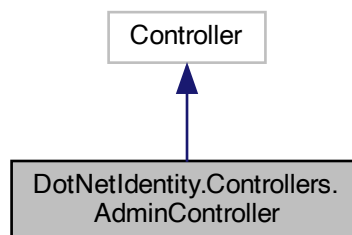
## Chapter 6

# Class Documentation

### 6.1 DotNetIdentity.Controllers.AdminController Class Reference

Controller Class for Admin views

Inheritance diagram for DotNetIdentity.Controllers.AdminController:



#### Public Member Functions

- [AdminController](#) ([AppDbContext](#) db, ILogger< [AdminController](#) > log, IConfiguration conf, UserManager< [AppUser](#) > userManager, RoleManager< [AppRole](#) > roleManager, SignInManager< [AppUser](#) > signInManager)  
*Controller Class constructor*
- IActionResult [Index](#) ()  
*Controller Action for Index*
- IActionResult [SystemLogs](#) ()  
*Controller Action for SystemLogs*
- IActionResult [AuditLogs](#) ()  
*Controller Action for AuditLogs*
- IActionResult [ErrorLogs](#) ()  
*Controller Action for ErrorLogs*

- `async Task< IActionResult > Users ()`  
*Controller Action for Users*
- `async Task< IActionResult > Roles ()`  
*Controller Action for Roles*
- `IActionResult GetErrorLogs (DataRequest request)`  
*Controller Post Method to fetch ErrorLogs*
- `IActionResult GetAppLogs (DataRequest request)`  
*Controller Post Method to fetch Applicationlogs*
- `IActionResult GetAuditLogs (DataRequest request)`  
*Controller Post Method to fetch AuditLogs*
- `JsonResult GetUsers (DataRequest request)`  
*Controller Post Method to fetch Users*
- `async Task< IActionResult > NewUser ()`  
*Controller Action for NewUser view*
- `async Task< ActionResult > NewUser (NewUserModel viewModel)`  
*controller Post Method to save a new User*
- `async Task< IActionResult > UpsertRole (string? id)`  
*Controller Action for view UpsertRole*
- `async Task< ActionResult > DisableUser (string UserId)`  
*Controller GET Method to disable a user account*
- `async Task< ActionResult > EnableUser (string UserId)`  
*Controller GET Method to enable a user account*
- `async Task< ActionResult > EnableLdapLogin (string UserId)`  
*Controller GET Method to enable ldap login for a user account*
- `async Task< ActionResult > DisableLdapLogin (string UserId)`  
*Controller GET Method to disable ldap login for a user account*
- `async Task< ActionResult > EnableMfaForce (string UserId)`  
*Controller GET Method to enable mfa enforcement for a user account*
- `async Task< ActionResult > DisableMfaForce (string UserId)`  
*Controller GET Method to disable mfa enforcement for a user account*
- `async Task< IActionResult > UpsertRole (UpsertRoleViewModel viewModel)`  
*Controller POST Method to enable a user account*
- `async Task< IActionResult > DeleteUser (string id)`  
*Controller POST Method to delete a user account*
- `async Task< IActionResult > DeleteRole (string id)`  
*Controller POST Method to enable a role*
- `async Task< IActionResult > EditUser (string id)`  
*Controller action to serve the EditUserView*
- `async Task< IActionResult > EditUser (EditUserViewModel viewModel)`  
*Controller POST Method to save user changes*

### 6.1.1 Detailed Description

Controller Class for Admin views

**Todo** add translation

Definition at line 24 of file [AdminController.cs](#).

## 6.1.2 Constructor & Destructor Documentation

### 6.1.2.1 AdminController()

```
DotNetIdentity.Controllers.AdminController.AdminController (
    AppDbContext db,
    ILogger< AdminController > log,
    IConfiguration conf,
    UserManager< AppUser > userManager,
    RoleManager< AppRole > roleManager,
    SignInManager< AppUser > signInManager ) [inline]
```

Controller Class constructor

#### Parameters

<i>db</i>	type <a href="#">DotNetIdentity.Data.AppDbContext</a>
<i>log</i>	type <a href="#">Microsoft.Extensions.Logging.ILogger</a>
<i>conf</i>	type <a href="#">Microsoft.Extensions.Configuration.IConfiguration</a>
<i>userManager</i>	type <a href="#">Microsoft.AspNetCore.Identity.UserManager</a>
<i>roleManager</i>	type <a href="#">Microsoft.AspNetCore.Identity.RoleManager</a>
<i>signInManager</i>	type <a href="#">Microsoft.AspNetCore.Identity.SignInManager</a>

Definition at line 60 of file [AdminController.cs](#).

```
00061     {
00062         _userManager = userManager;
00063         _roleManager = roleManager;
00064         _signInManager = signInManager;
00065         _configuration = conf;
00066         _logger = log;
00067         _context = db;
00068     }
```

## 6.1.3 Member Function Documentation

### 6.1.3.1 AuditLogs()

```
IActionResult DotNetIdentity.Controllers.AdminController.AuditLogs ( )
```

Controller Action for AutditLogs

#### Returns

View of type [DotnetIdentity.Views.Admin.AuditLogs](#)

### 6.1.3.2 DeleteRole()

```
async Task< IActionResult > DotNetIdentity.Controllers.AdminController.DeleteRole (
    string id ) [inline]
```

Controller POST Method to enable a role

**Parameters**

<i>id</i>	id of the role
-----------	----------------

**Returns**

redirect to controller action Roles

Definition at line 391 of file [AdminController.cs](#).

```

00392     {
00393         var role = await _roleManager.FindByIdAsync(id);
00394         if (role != null)
00395         {
00396             var result = await _roleManager.DeleteAsync(role);
00397             if (!result.Succeeded)
00398             {
00399                 result.Errors.ToList().ForEach(f => ModelState.AddModelError(string.Empty,
00400 f.Description));
00401             }
00402             else
00403             {
00404                 ModelState.AddModelError(string.Empty, "Role not found.");
00405             }
00406             _logger.LogInformation("AUDIT: " + User.Identity!.Name + " deleted role " + id);
00407             return RedirectToAction("Roles");
00408         }

```

**6.1.3.3 DeleteUser()**

```

async Task< IActionResult > DotNetIdentity.Controllers.AdminController.DeleteUser (
    string id ) [inline]

```

Controller POST Method to delete a user account

**Parameters**

<i>id</i>	the id of the user
-----------	--------------------

**Returns**

redirect to controller action Users

Definition at line 371 of file [AdminController.cs](#).

```

00371     {
00372         var user = await _userManager.FindByIdAsync(id);
00373         if (user != null) {
00374             var result = await _userManager.DeleteAsync(user);
00375             if (!result.Succeeded) {
00376                 result.Errors.ToList().ForEach(f => ModelState.AddModelError(string.Empty,
00377 f.Description));
00378             } else {
00379                 ModelState.AddModelError(string.Empty, "User not found.");
00380             }
00381             _logger.LogInformation("AUDIT: " + User.Identity!.Name + " deleted user " + id);
00382             return RedirectToAction("Users");
00383         }

```



### 6.1.3.4 DisableLdapLogin()

```
async Task< ActionResult > DotNetIdentity.Controllers.AdminController.DisableLdapLogin (
    string UserId ) [inline]
```

Controller GET Method to disable ldap login for a user account

#### Parameters

<i>UserId</i>	the id of the user
---------------	--------------------

#### Returns

json object (true / false)

Definition at line 282 of file [AdminController.cs](#).

```
00282
00283         var user = await _userManager.FindByIdAsync(UserId);
00284         if(user==null) {
00285             return Json(new { success = false });
00286         }
00287         user.IsLdapLogin = false;
00288         await _userManager.UpdateAsync(user);
00289         _logger.LogInformation("AUDIT: " + User.Identity!.Name + " disabled ldap login for user "
+ user.UserName);
00290         return Json(new {success = true});
00291     }
```

### 6.1.3.5 DisableMfaForce()

```
async Task< ActionResult > DotNetIdentity.Controllers.AdminController.DisableMfaForce (
    string UserId ) [inline]
```

Controller GET Method to disable mfa enforcement for a user account

#### Parameters

<i>UserId</i>	the id of the user
---------------	--------------------

#### Returns

json object (true / false)

Definition at line 316 of file [AdminController.cs](#).

```
00316
00317         var user = await _userManager.FindByIdAsync(UserId);
00318         if(user==null) {
00319             return Json(new { success = false });
00320         }
00321         user.IsMfaForce = false;
00322         await _userManager.UpdateAsync(user);
00323         _logger.LogInformation("AUDIT: " + User.Identity!.Name + " disabled mfa enforce for user "
+ user.UserName);
00324         return Json(new {success = true});
00325     }
```

### 6.1.3.6 DisableUser()

```
async Task< ActionResult > DotNetIdentity.Controllers.AdminController.DisableUser (
    string UserId ) [inline]
```

Controller GET Method to disable a user account

#### Parameters

<i>UserId</i>	the id of the user
---------------	--------------------

#### Returns

json object (true / false)

Definition at line 231 of file [AdminController.cs](#).

```
00231     {
00232         var user = await _userManager.FindByIdAsync(UserId);
00233         if(user==null) {
00234             return Json(new { success = false });
00235         }
00236         user.IsEnabled = false;
00237         await _userManager.UpdateAsync(user);
00238         _logger.LogInformation("AUDIT: " + User.Identity!.Name + " disabled user " +
user.UserName);
00239         return Json(new {success = true});
00240     }
```

### 6.1.3.7 EditUser() [1/2]

```
async Task< IActionResult > DotNetIdentity.Controllers.AdminController.EditUser (
    EditUserViewModel viewModel ) [inline]
```

Controller POST Method to save user changes

#### Parameters

<i>viewModel</i>	a model of type Dotnetidentity.Models.ViewModels.EditUserViewModel
------------------	--------------------------------------------------------------------

#### Returns

view of type Dotnetidentity.Views.Admin.EditUser

Definition at line 448 of file [AdminController.cs](#).

```
00449     {
00450         if (ModelState.IsValid)
00451         {
00452             var user = await _userManager.FindByIdAsync(viewModel.Id);
00453             if (user != null)
00454             {
00455                 if (user.PhoneNumber != viewModel.PhoneNumber && _userManager.Users.Any(a =>
a.PhoneNumber == viewModel.PhoneNumber))
00456                 {
00457                     ModelState.AddModelError(string.Empty, "Phone number already in use.");
00458                 }
00459                 else
```

```

00460         {
00461             user.FirstName = viewModel.FirstName;
00462             user.LastName = viewModel.LastName;
00463             user.UserName = viewModel.UserName;
00464             user.Email = viewModel.Email;
00465             user.PhoneNumber = viewModel.PhoneNumber;
00466             user.Gender = viewModel.Gender;
00467             user.Department = viewModel.Department.ToString();
00468
00469             var result = await _userManager.UpdateAsync(user);
00470             if (result.Succeeded)
00471             {
00472                 // Roles
00473                 foreach (var item in viewModel.Roles)
00474                 {
00475                     if (item.IsAssigned)
00476                     {
00477                         await _userManager.AddToRoleAsync(user, item.RoleName);
00478                     }
00479                     else
00480                     {
00481                         await _userManager.RemoveFromRoleAsync(user, item.RoleName);
00482                     }
00483                 }
00484
00485                 // Claims
00486                 var userClaims = await _userManager.GetClaimsAsync(user);
00487                 var departmentClaim = userClaims.FirstOrDefault(a => a.Type ==
00488                     "Department");
00489                 var claimsToAdd = new Claim("Department",
00490                     Enum.GetName(viewModel.Department));
00491                 if (departmentClaim != null)
00492                 {
00493                     await _userManager.ReplaceClaimAsync(user, departmentClaim,
00494                         claimsToAdd);
00495                 }
00496                 else
00497                 {
00498                     await _userManager.AddClaimAsync(user, claimsToAdd);
00499                 }
00500                 _logger.LogInformation("AUDIT: " + User.Identity!.Name + " modified user "
00501                     + user.UserName);
00502                 return RedirectToAction("Users");
00503             }
00504             result.Errors.ToList().ForEach(f => ModelState.AddModelError(string.Empty,
00505                 f.Description));
00506         }
00507     }
00508     else
00509     {
00510         ModelState.AddModelError(string.Empty, "User not found.");
00511     }
00512     return View(viewModel);
00513 }

```

References [DotNetIdentity.Models.ViewModels.EditUserViewModel.Department](#), [DotNetIdentity.Models.ViewModels.EditUserViewModel.FirstName](#), [DotNetIdentity.Models.ViewModels.EditUserViewModel.Gender](#), [DotNetIdentity.Models.ViewModels.EditUserViewModel.Id](#), [DotNetIdentity.Models.ViewModels.EditUserViewModel.LastName](#), [DotNetIdentity.Models.ViewModels.EditUserViewModel.PhoneNumber](#), [DotNetIdentity.Models.ViewModels.EditUserViewModel.Roles](#) and [DotNetIdentity.Models.ViewModels.EditUserViewModel.UserName](#).

### 6.1.3.8 EditUser() [2/2]

```

async Task< IActionResult > DotNetIdentity.Controllers.AdminController.EditUser (
    string id ) [inline]

```

Controller action to serve the EditUserView

#### Parameters

<i>id</i>	the id of the user
-----------	--------------------

**Returns**

view of type [DotNetIdentity.Models.ViewModels.EditUserViewModel](#)

Definition at line 415 of file [AdminController.cs](#).

```

00416     {
00417         var user = await _userManager.FindByIdAsync(id);
00418         if (user == null)
00419         {
00420             return RedirectToAction("Users");
00421         }
00422         var viewModel = user.Adapt<EditUserViewModel>();
00423
00424         var userRoles = await _userManager.GetRolesAsync(user);
00425         viewModel.Roles = await _roleManager.Roles.Select(s => new AssignRoleViewModel
00426         {
00427             RoleId = s.Id,
00428             RoleName = s.Name,
00429             IsAssigned = userRoles.Any(a => a == s.Name)
00430         }).ToListAsync();
00431
00432         var userClaims = await _userManager.GetClaimsAsync(user);
00433         var departmentClaim = userClaims.FirstOrDefault(f => f.Type == "Department");
00434         if (departmentClaim != null)
00435         {
00436             viewModel.Department = Enum.Parse<Department>(departmentClaim.Value);
00437         }
00438
00439         return View(viewModel);
00440     }

```

References [DotNetIdentity.Models.ViewModels.EditUserViewModel.Roles](#).

**6.1.3.9 EnableLdapLogin()**

```

async Task< ActionResult > DotNetIdentity.Controllers.AdminController.EnableLdapLogin (
    string UserId ) [inline]

```

Controller GET Method to enable ldap login for a user account

**Parameters**

<i>User↔ Id</i>	the id of the user
---------------------	--------------------

**Returns**

json object (true / false)

Definition at line 265 of file [AdminController.cs](#).

```

00265     {
00266         var user = await _userManager.FindByIdAsync(UserId);
00267         if(user==null) {
00268             return Json(new { success = false });
00269         }
00270         user.IsLdapLogin = true;
00271         await _userManager.UpdateAsync(user);
00272         _logger.LogInformation("AUDIT: " + User.Identity!.Name + " enabled LDAP login for user " +
user.UserName);
00273         return Json(new {success = true});
00274     }

```

### 6.1.3.10 EnableMfaForce()

```
async Task< ActionResult > DotNetIdentity.Controllers.AdminController.EnableMfaForce (
    string UserId ) [inline]
```

Controller GET Method to enable mfa enforcement for a user account

#### Parameters

<i>UserId</i>	the id of the user
---------------	--------------------

#### Returns

json object (true / false)

Definition at line 299 of file [AdminController.cs](#).

```
00299 {
00300     var user = await _userManager.FindByIdAsync(UserId);
00301     if(user==null) {
00302         return Json(new { success = false });
00303     }
00304     user.IsMfaForce = true;
00305     await _userManager.UpdateAsync(user);
00306     _logger.LogInformation("AUDIT: " + User.Identity!.Name + " enabled mfa enforce for user "
+ user.UserName);
00307     return Json(new {success = true});
00308 }
```

### 6.1.3.11 EnableUser()

```
async Task< ActionResult > DotNetIdentity.Controllers.AdminController.EnableUser (
    string UserId ) [inline]
```

Controller GET Method to enable a user account

#### Parameters

<i>UserId</i>	the id of the user
---------------	--------------------

#### Returns

json object (true / false)

Definition at line 248 of file [AdminController.cs](#).

```
00248 {
00249     var user = await _userManager.FindByIdAsync(UserId);
00250     if(user==null) {
00251         return Json(new { success = false });
00252     }
00253     user.IsEnabled = true;
00254     await _userManager.UpdateAsync(user);
00255     _logger.LogInformation("AUDIT: " + User.Identity!.Name + " enabled user " +
user.UserName);
00256     return Json(new {success = true});
00257 }
```

#### 6.1.3.12 ErrorLogs()

```
ActionResult DotNetIdentity.Controllers.AdminController.ErrorLogs ( )
```

Controller Action for ErrorLogs

##### Returns

View of type DotnetIdentity.Views.Admin.Errorogs

#### 6.1.3.13 GetAppLogs()

```
ActionResult DotNetIdentity.Controllers.AdminController.GetAppLogs (
    DataRequest request ) [inline]
```

Controller Post Method to fetch Applicationlogs

##### Parameters

<i>request</i>	type DatatablesJs.Data.DataRequest
----------------	------------------------------------

##### Returns

Jason-Array of DatatablesJs.Data.DataResult

Definition at line 119 of file [AdminController.cs](#).

```
00120     {
00121         DataResult<AppLogs> result =
            _context.AppLogs!.Where(l=>l.Message.ToLower().StartsWith("audit")==false && Convert.ToInt32(l.Level)
            < 4).ToDataResult(request);
00122         return Json(result);
00123     }
```

#### 6.1.3.14 GetAuditLogs()

```
ActionResult DotNetIdentity.Controllers.AdminController.GetAuditLogs (
    DataRequest request ) [inline]
```

Controller Post Method to fetch AuditLogs

##### Parameters

<i>request</i>	type DatatablesJs.Data.DataRequest
----------------	------------------------------------

##### Returns

Jason-Array of DatatablesJs.Data.DataResult

Definition at line 131 of file [AdminController.cs](#).

```
00132     {
00133         IActionResult<AppLogs> result =
            _context.AppLogs!.Where(l=>l.Message.ToLower().StartsWith("audit")==true).ToDataResult(request);
00134         return Json(result);
00135     }
```

### 6.1.3.15 GetErrorLogs()

```
IActionResult DotNetIdentity.Controllers.AdminController.GetErrorLogs (
    DataRequest request ) [inline]
```

Controller Post Method to fetch ErrorLogs

#### Parameters

<i>request</i>	type DatatablesJs.Data.DataRequest
----------------	------------------------------------

#### Returns

Jason-Array of DatatablesJs.Data.DataResult

Definition at line 107 of file [AdminController.cs](#).

```
00108     {
00109         IActionResult<AppLogs> result =
            _context.AppLogs!.Where(l=>l.Message.ToLower().StartsWith("audit")==false && Convert.ToInt32(l.Level)
            == 4).ToDataResult(request);
00110         return Json(result);
00111     }
```

### 6.1.3.16 GetUsers()

```
JsonResult DotNetIdentity.Controllers.AdminController.GetUsers (
    DataRequest request ) [inline]
```

Controller Post Method to fetch Users

#### Parameters

<i>request</i>	type DatatablesJs.Data.DataRequest
----------------	------------------------------------

#### Returns

Jason-Array of DatatablesJs.Data.DataResult

Definition at line 143 of file [AdminController.cs](#).

```
00143     {
00144         IActionResult<AppUser> result = _userManager.Users.ToDataResult(request);
00145         return Json(result);
00146     }
```

**6.1.3.17 Index()**

```
IActionResult DotNetIdentity.Controllers.AdminController.Index ( )
```

Controller Action for Index

**Returns**

View of type DotnetIdentity.Views.Admin.Index

**6.1.3.18 NewUser() [1/2]**

```
async Task< IActionResult > DotNetIdentity.Controllers.AdminController.NewUser ( ) [inline]
```

Controller Action for NewUser view

**Returns**

view of type DotnetIdentity.Views.Admin.NewUser

Definition at line 152 of file [AdminController.cs](#).

```
00152                                     {
00153         var mod = new NewUserModel();
00154         mod.Roles = await _roleManager.Roles.Select(s => new AssignRoleViewModel
00155         {
00156             RoleId = s.Id,
00157             RoleName = s.Name,
00158             IsAssigned = false
00159         }).ToListAsync();
00160         return View(mod);
00161     }
00162 }
```

**6.1.3.19 NewUser() [2/2]**

```
async Task< ActionResult > DotNetIdentity.Controllers.AdminController.NewUser (
    NewUserModel viewModel ) [inline]
```

controller Post Method to save a new User

**Parameters**

<i>viewModel</i>	type DotnetIdentity.Models.ViewModels.NewUserModel
------------------	----------------------------------------------------

**Returns**

view of type DotnetIdentity.Views.Admin.NewUser

Definition at line 170 of file [AdminController.cs](#).

```
00171     {
```



```

00172         if (ModelState.IsValid)
00173         {
00174             var user = new AppUser();
00175             user.IsEnabled = true;
00176             user.IsMfaForce = viewModel.IsMfaForce;
00177             user.IsLdapLogin = viewModel.IsLdapLogin;
00178             user.FirstName = viewModel.FirstName;
00179             user.LastName = viewModel.LastName;
00180             user.Email = viewModel.Email;
00181             user.EmailConfirmed = true;
00182             user.Gender = viewModel.Gender;
00183             user.Department = viewModel.Department.ToString();
00184             user.UserName = viewModel.UserName;
00185
00186             var result = await _userManager.CreateAsync(user, viewModel.Password);
00187             if (result.Succeeded)
00188             {
00189                 // roles
00190                 foreach (var item in viewModel.Roles)
00191                 {
00192                     if (item.IsAssigned)
00193                     {
00194                         await _userManager.AddToRoleAsync(user, item.RoleName);
00195                     }
00196                     else
00197                     {
00198                         await _userManager.RemoveFromRoleAsync(user, item.RoleName);
00199                     }
00200                 }
00201             }
00202
00203             ViewData["message"] = "User created success fully.";
00204             _logger.LogInformation("AUDIT: " + User.Identity!.Name + " created new user " +
user.UserName);
00205         }
00206         return View(new NewUserModel());
00207     }

```

References [DotNetIdentity.Models.ViewModels.NewUserModel.Department](#), [DotNetIdentity.Models.ViewModels.NewUserModel.Email](#), [DotNetIdentity.Models.ViewModels.NewUserModel.FirstName](#), [DotNetIdentity.Models.ViewModels.NewUserModel.Gender](#), [DotNetIdentity.Models.ViewModels.NewUserModel.IsLdapLogin](#), [DotNetIdentity.Models.ViewModels.NewUserModel.IsMfaForce](#), [DotNetIdentity.Models.ViewModels.NewUserModel.LastName](#), [DotNetIdentity.Models.ViewModels.NewUserModel.Password](#), [DotNetIdentity.Models.ViewModels.NewUserModel.Roles](#), and [DotNetIdentity.Models.ViewModels.NewUserModel.UserName](#).

### 6.1.3.20 Roles()

```
async Task< IActionResult > DotNetIdentity.Controllers.AdminController.Roles ( )
```

Controller Action for Roles

#### Returns

View of type [DotnetIdentity.Views.Admin.Users](#)

### 6.1.3.21 SystemLogs()

```
IActionResult DotNetIdentity.Controllers.AdminController.SystemLogs ( )
```

Controller Action for SystemLogs

#### Returns

View of type [DotnetIdentity.Views.Admin.SystemLogs](#)

**6.1.3.22 UpsertRole()** [1/2]

```
async Task< IActionResult > DotNetIdentity.Controllers.AdminController.UpsertRole (
    string? id ) [inline]
```

Controller Action for view UpsertRole

**Parameters**

<i>id</i>	
-----------	--

**Returns**

view of type Dotnetidentity.Views.Admin.UpsertRole

Definition at line 214 of file [AdminController.cs](#).

```
00215     {
00216         if (id != null)
00217         {
00218             var role = await _roleManager.FindByIdAsync(id);
00219             return View(role.Adapt<UpsertRoleViewModel>());
00220         }
00221
00222         return View(new UpsertRoleViewModel());
00223     }
```

**6.1.3.23 UpsertRole()** [2/2]

```
async Task< IActionResult > DotNetIdentity.Controllers.AdminController.UpsertRole (
    UpsertRoleViewModel viewModel ) [inline]
```

Controller POST Method to enable a user account

**Parameters**

<i>viewModel</i>	tmmodel of type Dotnetidentity.Models.ViewModels.UpsertRoleViewModel
------------------	----------------------------------------------------------------------

**Returns**

redirect to controller action Roles

Definition at line 333 of file [AdminController.cs](#).

```
00334     {
00335         if (ModelState.IsValid)
00336         {
00337             var isUpdate = viewModel.Id != null;
00338
00339             var role = isUpdate ? await _roleManager.FindByIdAsync(viewModel.Id) : new AppRole() {
Name = viewModel.Name, CreatedOn = DateTime.Now };
00340
00341             if (isUpdate)
00342             {
00343                 role.Name = viewModel.Name;
00344             }
00345
00346             var result = isUpdate ? await _roleManager.UpdateAsync(role) : await
_roleManager.CreateAsync(role);
00347             if (result.Succeeded)
```

```

00348         {
00349             if (isUpdate) {
00350                 _logger.LogInformation("AUDIT: " + User.Identity!.Name + " changed name of " +
viewModel.Id + " to " + viewModel.Name);
00351             } else {
00352                 _logger.LogInformation("AUDIT: " + User.Identity!.Name + " created role " +
viewModel.Name);
00353             }
00354             return RedirectToAction("Roles");
00355         }
00356         result.Errors.ToList().ForEach(f => ModelState.AddModelError(string.Empty,
f.Description));
00357         result.Errors.ToList().ForEach(f => TempData["Error"] += f.Description);
00358     }
00359
00360     //return View(viewModel);
00361     //return View("Roles", await _roleManager.Roles.ToListAsync());
00362     return RedirectToAction("Roles");
00363 }

```

References [DotNetIdentity.Models.ViewModels.UpsertRoleViewModel.Id](#), and [DotNetIdentity.Models.ViewModels.UpsertRoleViewMo](#)

### 6.1.3.24 Users()

```
async Task< IActionResult > DotNetIdentity.Controllers.AdminController.Users ( )
```

Controller Action for Users

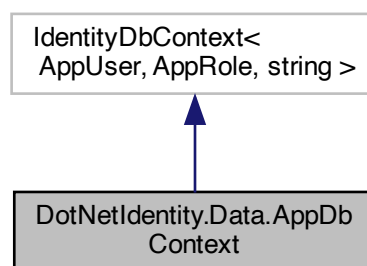
Returns

View of type `DotnetIdentity.Views.Admin.Users`

## 6.2 DotNetIdentity.Data.AppDbContext Class Reference

Application Database context class

Inheritance diagram for `DotNetIdentity.Data.AppDbContext`:



### Public Member Functions

- [AppDbContext](#) (`DbContextOptions< AppDbContext > options`)  
*class constructor*

## Protected Member Functions

- override void [OnModelCreating](#) (ModelBuilder builder)  
*Overriding OnModelCreation to seed initial data*

## Properties

- DbSet< [AppLogs](#) >? [AppLogs](#) [get, set]  
*Property AppLogs*
- DbSet< [ApplicationSettings](#) >? [AppSettings](#) [get, set]  
*property AppSettings*
- DbSet< [SessionCache](#) >? [AppSessionCache](#) [get, set]  
*property AppSessionCache*

### 6.2.1 Detailed Description

Application Database context class

Definition at line 12 of file [AppDbContext.cs](#).

### 6.2.2 Constructor & Destructor Documentation

#### 6.2.2.1 AppDbContext()

```
DotNetIdentity.Data.AppDbContext.AppDbContext (
    DbContextOptions< AppDbContext > options ) [inline]
```

class constructor

#### Parameters

<i>options</i>	type DbContextOptions
----------------	-----------------------

#### Returns

Definition at line 19 of file [AppDbContext.cs](#).

```
00019                                     : base(options) {
00020                                     }
```

### 6.2.3 Member Function Documentation

## 6.2.3.1 OnModelCreating()

```
override void DotNetIdentity.Data.AppDbContext.OnModelCreating (
    ModelBuilder builder ) [inline], [protected]
```

Overriding OnModelCreation to seed initial data

## Parameters

<i>builder</i>	type ModelBuilder
----------------	-------------------

Definition at line 41 of file [AppDbContext.cs](#).

```
00042     {
00043         base.OnModelCreating(builder);
00044
00045         builder.Entity<ApplicationSettings>()
00046             .HasKey(x => new { x.Name, x.Type });
00047
00048         builder.Entity<ApplicationSettings>()
00049             .Property(x => x.Value);
00050
00051         // seeding default settings
00052         builder.Entity<ApplicationSettings>().HasData(new ApplicationSettings { Name =
00053 "ApplicationName", Type = "GlobalSettings", Value = "SecPass"});
00054         builder.Entity<ApplicationSettings>().HasData(new ApplicationSettings { Name =
00055 "SessionTimeoutWarnAfter", Type = "GlobalSettings", Value = "50000"});
00056         builder.Entity<ApplicationSettings>().HasData(new ApplicationSettings { Name =
00057 "SessionTimeoutRedirAfter", Type = "GlobalSettings", Value = "70000"});
00058         builder.Entity<ApplicationSettings>().HasData(new ApplicationSettings { Name =
00059 "SessionCookieExpiration", Type = "GlobalSettings", Value = "7"});
00060         builder.Entity<ApplicationSettings>().HasData(new ApplicationSettings { Name = "Username",
00061 Type = "MailSettings", Value = "YOUR_Smtp_Username"});
00062         builder.Entity<ApplicationSettings>().HasData(new ApplicationSettings { Name = "Password",
00063 Type = "MailSettings", Value = "YOUR_SmtpPassword"});
00064         builder.Entity<ApplicationSettings>().HasData(new ApplicationSettings { Name =
00065 "SmtpServer", Type = "MailSettings", Value = "YOUR_SmtpServer"});
00066         builder.Entity<ApplicationSettings>().HasData(new ApplicationSettings { Name = "SmtpPort",
00067 Type = "MailSettings", Value = "587"});
00068         builder.Entity<ApplicationSettings>().HasData(new ApplicationSettings { Name =
00069 "SmtpUseTls", Type = "MailSettings", Value = "true"});
00070         builder.Entity<ApplicationSettings>().HasData(new ApplicationSettings { Name =
00071 "SmtpFromAddress", Type = "MailSettings", Value = "YOUR_From_Address"});
00072
00073         //Seeding roles toAspNetRoles table
00074         builder.Entity<AppRole>().HasData(new AppRole { Id =
00075 "dffc6dd5-b145-41e9-a861-c87ff673e9ca", Name = "Admin", NormalizedName = "ADMIN.ToUpper() });
00076         builder.Entity<AppRole>().HasData(new AppRole { Id =
00077 "f8a527ac-d7f6-4d9d-aca6-46b2261b042b", Name = "User", NormalizedName = "USER.ToUpper() });
00078         builder.Entity<AppRole>().HasData(new AppRole { Id =
00079 "g7a527ac-d7t6-4d7z-aca6-45t2261b042b", Name = "Editor", NormalizedName = "EDITOR.ToUpper() });
00080         builder.Entity<AppRole>().HasData(new AppRole { Id =
00081 "p9a527ac-d77w-4d3r-aca6-35b2261b042b", Name = "Moderator", NormalizedName = "MODERATOR.ToUpper()
00082 });
00083
00084         //a hasher to hash the password before seeding the user to the db
00085         var hasher = new PasswordHasher<AppUser>();
00086
00087         //Seeding the Admin User to AspNetUsers table
00088         builder.Entity<AppUser>().HasData(
00089             new AppUser
00090             {
00091                 Id = "6fbfb682-568c-4f5b-a298-85937ca4f7f3", // primary key
00092                 UserName = "super.admin",
00093                 NormalizedUserName = "SUPER.ADMIN",
00094                 PasswordHash = hasher.HashPassword(null!, "Test1000!"),
00095                 Email = "super.admin@local.app",
00096                 NormalizedEmail = "SUPER.ADMIN@LOCAL.APP",
00097                 EmailConfirmed = true,
00098                 FirstName = "Super",
00099                 LastName = "Admin",
00100                 IsMfaForce = false,
00101                 IsLdapLogin = false,
00102                 IsEnabled = true
00103             }
00104         );
00105
00106         //Seeding the relation between our user and role to AspNetUserRoles table
```

```

00094         builder.Entity<IdentityUserRole<string>>().HasData(
00095             new IdentityUserRole<string>
00096             {
00097                 RoleId = "dffc6dd5-b145-41e9-a861-c87ff673e9ca",
00098                 UserId = "6fbfb682-568c-4f5b-a298-85937ca4f7f3"
00099             }
00100         );
00101     }

```

## 6.2.4 Property Documentation

### 6.2.4.1 AppLogs

`DbSet<AppLogs>?` `DotNetIdentity.Data.AppDbContext.AppLogs` `[get]`, `[set]`

Property AppLogs

Definition at line 25 of file [AppDbContext.cs](#).

```
00025 {get;set;}
```

### 6.2.4.2 AppSessionCache

`DbSet<SessionCache>?` `DotNetIdentity.Data.AppDbContext.AppSessionCache` `[get]`, `[set]`

property AppSessionCache

Definition at line 35 of file [AppDbContext.cs](#).

```
00035 {get;set;}
```

### 6.2.4.3 AppSettings

`DbSet<ApplicationSettings>?` `DotNetIdentity.Data.AppDbContext.AppSettings` `[get]`, `[set]`

property AppSettings

Definition at line 30 of file [AppDbContext.cs](#).

```
00030 {get;set;}
```

Referenced by [DotNetIdentity.Services.SettingsService.AppSettingsBase.Load\(\)](#), and [DotNetIdentity.Services.SettingsService.AppSettingsBase.Save\(\)](#)

## 6.3 DotNetIdentity.Models.DataModels.ApplicationSettings Class Reference

class define an Settings object

## Properties

- string? **Name** [get, set]  
*property name*
- string? **Type** [get, set]  
*property Type*
- string? **Value** [get, set]  
*property value*

### 6.3.1 Detailed Description

class define an Settings object

Definition at line 7 of file [ApplicationSettings.cs](#).

### 6.3.2 Property Documentation

#### 6.3.2.1 Name

```
string? DotNetIdentity.Models.DataModels.ApplicationSettings.Name [get], [set]
```

property name

string

Definition at line 12 of file [ApplicationSettings.cs](#).

```
00012 {get;set;}
```

Referenced by [DotNetIdentity.Services.SettingsService.AppSettingsBase.Save\(\)](#).

#### 6.3.2.2 Type

```
string? DotNetIdentity.Models.DataModels.ApplicationSettings.Type [get], [set]
```

property Type

string

Definition at line 17 of file [ApplicationSettings.cs](#).

```
00017 {get;set;}
```

### 6.3.2.3 Value

```
string? DotNetIdentity.Models.DataModels.ApplicationSettings.Value [get], [set]
```

property value

string

Definition at line 22 of file [ApplicationSettings.cs](#).

```
00022 {get;set;}
```

## 6.4 DotNetIdentity.Models.DataModels.AppLogs Class Reference

class to define ann Log object

### Properties

- int [id](#) [get, set]  
*property id*
- string? [Exception](#) [get, set]  
*property Exception*
- string [Level](#) = string.Empty [get, set]  
*property Level*
- string [Message](#) = string.Empty [get, set]  
*property Message*
- string [MessageTemplate](#) = string.Empty [get, set]  
*property MessageTemplate*
- string [Properties](#) = string.Empty [get, set]  
*property Properties*
- string [Timestamp](#) = string.Empty [get, set]  
*property Timestamp*

### 6.4.1 Detailed Description

class to define ann Log object

Definition at line 5 of file [AppLogs.cs](#).

### 6.4.2 Property Documentation



### 6.4.2.1 Exception

```
string? DotNetIdentity.Models.DataModels.AppLogs.Exception [get], [set]
```

property Exception

string

Definition at line 15 of file [AppLogs.cs](#).

```
00015 {get;set;}
```

### 6.4.2.2 id

```
int DotNetIdentity.Models.DataModels.AppLogs.id [get], [set]
```

property id

int

Definition at line 10 of file [AppLogs.cs](#).

```
00010 {get;set;}
```

### 6.4.2.3 Level

```
string DotNetIdentity.Models.DataModels.AppLogs.Level = string.Empty [get], [set]
```

property Level

string

Definition at line 20 of file [AppLogs.cs](#).

```
00020 {get;set;} = string.Empty;
```

### 6.4.2.4 Message

```
string DotNetIdentity.Models.DataModels.AppLogs.Message = string.Empty [get], [set]
```

property Message

string

Definition at line 25 of file [AppLogs.cs](#).

```
00025 {get;set;} = string.Empty;
```

#### 6.4.2.5 MessageTemplate

```
string DotNetIdentity.Models.DataModels.AppLogs.MessageTemplate = string.Empty [get], [set]
```

property MessageTemplate

string

Definition at line 30 of file [AppLogs.cs](#).

```
00030 {get;set;} = string.Empty;
```

#### 6.4.2.6 Properties

```
string DotNetIdentity.Models.DataModels.AppLogs.Properties = string.Empty [get], [set]
```

property Properties

string

Definition at line 35 of file [AppLogs.cs](#).

```
00035 {get;set;} = string.Empty;
```

#### 6.4.2.7 Timestamp

```
string DotNetIdentity.Models.DataModels.AppLogs.Timestamp = string.Empty [get], [set]
```

property Timestamp

string

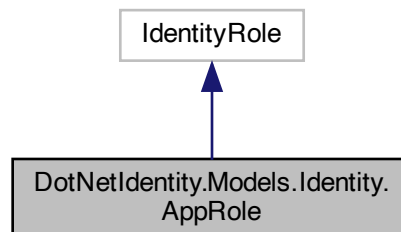
Definition at line 40 of file [AppLogs.cs](#).

```
00040 {get;set;} = string.Empty;
```

### 6.5 DotNetIdentity.Models.Identity.AppRole Class Reference

class to define a role object

Inheritance diagram for DotNetIdentity.Models.Identity.AppRole:



## Properties

- DateTime [CreatedOn](#) [get, set]  
*property created on*

### 6.5.1 Detailed Description

class to define a role object

Definition at line 8 of file [AppRole.cs](#).

### 6.5.2 Property Documentation

#### 6.5.2.1 CreatedOn

```
DateTime DotNetIdentity.Models.Identity.AppRole.CreatedOn [get], [set]
```

property created on

DateTime

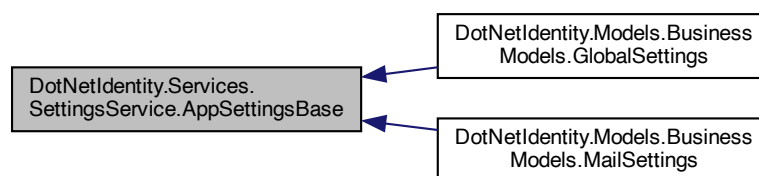
Definition at line 14 of file [AppRole.cs](#).

```
00014 { get; set; }
```

## 6.6 DotNetIdentity.Services.SettingsService.AppSettingsBase Class Reference

base class for Settings classes

Inheritance diagram for DotNetIdentity.Services.SettingsService.AppSettingsBase:



## Public Member Functions

- [AppSettingsBase](#) ()  
*class constructor*
- virtual void [Load](#) ([AppDbContext](#) unitOfWork)  
*load method*
- virtual void [Save](#) ([AppDbContext](#) unitOfWork)  
*save method*

### 6.6.1 Detailed Description

base class for Settings classes

Definition at line 9 of file [AppSettingsBase.cs](#).

### 6.6.2 Constructor & Destructor Documentation

#### 6.6.2.1 AppSettingsBase()

```
DotNetIdentity.Services.SettingsService.AppSettingsBase.AppSettingsBase ( ) [inline]
```

class constructor

Definition at line 23 of file [AppSettingsBase.cs](#).

```
00024     {
00025         var type = this.GetType();
00026         _name = type.Name;
00027         // 2
00028         _properties = type.GetProperties();
00029     }
```

### 6.6.3 Member Function Documentation

#### 6.6.3.1 Load()

```
virtual void DotNetIdentity.Services.SettingsService.AppSettingsBase.Load (
    AppDbContext unitOfWork ) [inline], [virtual]
```

load method

Parameters

<i>unitOfWork</i>	type DbContext
-------------------	----------------

Definition at line 35 of file [AppSettingsBase.cs](#).

```

00036     {
00037         // ARGUMENT CHECKING SKIPPED FOR BREVITY
00038         // 3 get settings for this type name
00039         var settings = unitOfWork.AppSettings!.Where(w => w.Type == _name).ToList();
00040
00041         foreach (var propertyInfo in _properties)
00042         {
00043             // get the setting from the settings list
00044             var setting = settings.SingleOrDefault(s => s.Name == propertyInfo.Name);
00045             if (setting != null)
00046             {
00047                 // 4 assign the setting values to the properties in the type inheriting this class
00048                 propertyInfo.SetValue(this, Convert.ChangeType(setting.Value,
00049                     propertyInfo.PropertyType));
00049             }
00050         }
00051     }

```

References [DotNetIdentity.Data.AppDbContext.AppSettings](#).

### 6.6.3.2 Save()

```

virtual void DotNetIdentity.Services.SettingsService.AppSettingsBase.Save (
    AppDbContext unitOfWork ) [inline], [virtual]

```

save method

Parameters

<i>unitOfWork</i>	type <a href="#">DbContext</a>
-------------------	--------------------------------

Definition at line 57 of file [AppSettingsBase.cs](#).

```

00058     {
00059         // 5 load existing settings for this type
00060         var settings = unitOfWork.AppSettings!.Where(w => w.Type == _name).ToList();
00061
00062         foreach (var propertyInfo in _properties)
00063         {
00064             object propertyValue = propertyInfo.GetValue(this, null!);
00065             string value = (propertyValue == null) ? null! : propertyValue.ToString();
00066
00067             var setting = settings.SingleOrDefault(s => s.Name == propertyInfo.Name);
00068             if (setting != null)
00069             {
00070                 // 6 update existing value
00071                 setting.Value = value;
00072             }
00073             else
00074             {
00075                 // 7 create new setting
00076                 var newSetting = new ApplicationSettings()
00077                 {
00078                     Name = propertyInfo.Name,
00079                     Type = _name,
00080                     Value = value,
00081                 };
00082                 unitOfWork.AppSettings!.Add(newSetting);
00083             }
00084         }
00085     }

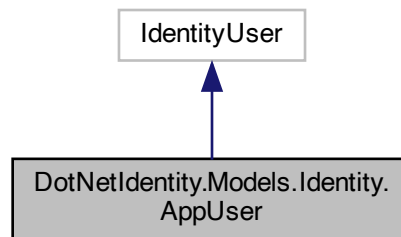
```

References [DotNetIdentity.Data.AppDbContext.AppSettings](#), and [DotNetIdentity.Models.DataModels.ApplicationSettings.Name](#).

## 6.7 DotNetIdentity.Models.Identity.AppUser Class Reference

class to define a user object

Inheritance diagram for DotNetIdentity.Models.Identity.AppUser:



## Properties

- [TwoFactorType](#) [TwoFactorType](#) [get, set]  
*2fa type property*
- [DateTime](#) [BirthDay](#) [get, set]  
*birthday property*
- [Gender](#) [Gender](#) [get, set]  
*gender property*
- [DateTime](#) [CreatedOn](#) [get, set]  
*birthday CreatedOn*
- [String?](#) [FirstName](#) = string.Empty [get, set]  
*firstname property*
- [String?](#) [LastName](#) = string.Empty [get, set]  
*lastname property*
- [bool](#) [IsMfaForce](#) [get, set]  
*2fa force property*
- [bool](#) [IsLdapLogin](#) [get, set]  
*ldap login property*
- [string?](#) [Department](#) = string.Empty [get, set]  
*department property*
- [string?](#) [ProfilePicture](#) [get, set]  
*profile picture proerty*
- [bool](#) [IsEnabled](#) [get, set]  
*account enbaled property*

### 6.7.1 Detailed Description

class to define a user object

Definition at line 8 of file [AppUser.cs](#).

### 6.7.2 Property Documentation

### 6.7.2.1 Birthday

`DateTime DotNetIdentity.Models.Identity.AppUser.Birthday [get], [set]`

birthday property

DateTime

Definition at line 19 of file [AppUser.cs](#).

```
00019 { get; set; }
```

### 6.7.2.2 CreatedOn

`DateTime DotNetIdentity.Models.Identity.AppUser.CreatedOn [get], [set]`

birthday CreatedOn

DateTime

Definition at line 29 of file [AppUser.cs](#).

```
00029 { get; set; }
```

### 6.7.2.3 Department

`string? DotNetIdentity.Models.Identity.AppUser.Department = string.Empty [get], [set]`

department property

Department

Definition at line 54 of file [AppUser.cs](#).

```
00054 { get; set; } = string.Empty;
```

### 6.7.2.4 FirstName

`String? DotNetIdentity.Models.Identity.AppUser.FirstName = string.Empty [get], [set]`

firstname property

string

Definition at line 34 of file [AppUser.cs](#).

```
00034 {get; set;} = string.Empty;
```

#### 6.7.2.5 Gender

`Gender` `DotNetIdentity.Models.Identity.AppUser.Gender` [get], [set]

gender property

Gender

Definition at line 24 of file [AppUser.cs](#).

```
00024 { get; set; }
```

#### 6.7.2.6 IsEnabled

`bool` `DotNetIdentity.Models.Identity.AppUser.IsEnabled` [get], [set]

account enbaled property

bool

Definition at line 64 of file [AppUser.cs](#).

```
00064 {get; set;}
```

#### 6.7.2.7 IsLdapLogin

`bool` `DotNetIdentity.Models.Identity.AppUser.IsLdapLogin` [get], [set]

ldap login property

bool

Definition at line 49 of file [AppUser.cs](#).

```
00049 {get; set;}
```

#### 6.7.2.8 IsMfaForce

`bool` `DotNetIdentity.Models.Identity.AppUser.IsMfaForce` [get], [set]

2fa force property

bool

Definition at line 44 of file [AppUser.cs](#).

```
00044 {get; set;}
```



### 6.7.2.9 LastName

```
String? DotNetIdentity.Models.Identity.AppUser.LastName = string.Empty [get], [set]
```

lastname property

string

Definition at line 39 of file [AppUser.cs](#).

```
00039 {get; set;} = string.Empty;
```

### 6.7.2.10 ProfilePicture

```
string? DotNetIdentity.Models.Identity.AppUser.ProfilePicture [get], [set]
```

profile picture proerty

string

Definition at line 59 of file [AppUser.cs](#).

```
00059 { get; set; }
```

### 6.7.2.11 TwoFactorType

```
TwoFactorType DotNetIdentity.Models.Identity.AppUser.TwoFactorType [get], [set]
```

2fa type property

TwoFactorType

Definition at line 14 of file [AppUser.cs](#).

```
00014 { get; set; }
```

## 6.8 DotNetIdentity.Models.ViewModels.AssignRoleViewModel Class Reference

model for UserEditView

### Properties

- string [RoleId](#) = default! [get, set]  
*the role id*
- string [RoleName](#) = default! [get, set]  
*the role neame*
- bool [IsAssigned](#) [get, set]  
*if role is assigned*

### 6.8.1 Detailed Description

model for UserEditView

Definition at line 6 of file [AssignRoleViewModel.cs](#).

### 6.8.2 Property Documentation

#### 6.8.2.1 IsAssigned

```
bool DotNetIdentity.Models.ViewModels.AssignRoleViewModel.IsAssigned [get], [set]
```

if role is assigned

bool

Definition at line 22 of file [AssignRoleViewModel.cs](#).

```
00022 { get; set; }
```

#### 6.8.2.2 RoleId

```
string DotNetIdentity.Models.ViewModels.AssignRoleViewModel.RoleId = default! [get], [set]
```

the role id

string

Definition at line 12 of file [AssignRoleViewModel.cs](#).

```
00012 { get; set; } = default!;
```

#### 6.8.2.3 RoleName

```
string DotNetIdentity.Models.ViewModels.AssignRoleViewModel.RoleName = default! [get], [set]
```

the role neame

string

Definition at line 17 of file [AssignRoleViewModel.cs](#).

```
00017 { get; set; } = default!;
```

## 6.9 DotNetIdentity.Models.ViewModels.ChangePasswordViewModel Class Reference

model for ChangePassword view

### Properties

- string `Password` = default! [get, set]  
*the password*
- string `NewPassword` = default! [get, set]  
*the new password*
- string `NewPasswordConfirm` = default! [get, set]  
*the new password confirmed*

### 6.9.1 Detailed Description

model for ChangePassword view

Definition at line 8 of file [ChangePasswordViewModel.cs](#).

### 6.9.2 Property Documentation

#### 6.9.2.1 NewPassword

```
string DotNetIdentity.Models.ViewModels.ChangePasswordViewModel.NewPassword = default! [get],  
[set]
```

the new password

string

Definition at line 19 of file [ChangePasswordViewModel.cs](#).

```
00019 { get; set; } = default!;
```

Referenced by [DotNetIdentity.Controllers.UserController.ChangePassword\(\)](#).

#### 6.9.2.2 NewPasswordConfirm

```
string DotNetIdentity.Models.ViewModels.ChangePasswordViewModel.NewPasswordConfirm = default!  
[get], [set]
```

the new password confirmed

string

Definition at line 25 of file [ChangePasswordViewModel.cs](#).

```
00025 { get; set; } = default!;
```

### 6.9.2.3 Password

```
string DotNetIdentity.Models.ViewModels.ChangePasswordViewModel.Password = default! [get],  
[set]
```

the password

string

Definition at line 14 of file [ChangePasswordViewModel.cs](#).

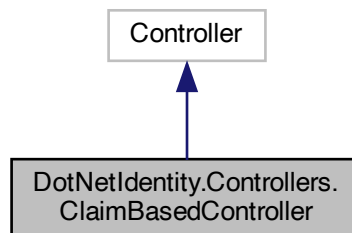
```
00014 { get; set; } = default!;
```

Referenced by [DotNetIdentity.Controllers.UserController.ChangePassword\(\)](#).

## 6.10 DotNetIdentity.Controllers.ClaimBasedController Class Reference

Controller to test ClaimBased Authorization

Inheritance diagram for DotNetIdentity.Controllers.ClaimBasedController:



### Public Member Functions

- IActionResult [HR](#) ()  
*Controller action for HR view*
- IActionResult [Software](#) ()  
*Controller action for Software view*
- IActionResult [Employee](#) ()  
*Controller action for Employee view*
- IActionResult [FreeTrial](#) ()  
*Controller action for FreeTrial view*
- IActionResult [Violence](#) ()  
*Controller action for AtLeast18Policy view*

### 6.10.1 Detailed Description

Controller to test ClaimBased Authorization

Definition at line 9 of file [ClaimBasedController.cs](#).

### 6.10.2 Member Function Documentation

#### 6.10.2.1 Employee()

```
IActionResult DotNetIdentity.Controllers.ClaimBasedController.Employee ( )
```

Controller action for Employee view

##### Returns

view Employee

#### 6.10.2.2 FreeTrial()

```
IActionResult DotNetIdentity.Controllers.ClaimBasedController.FreeTrial ( )
```

Controller action for FreeTrial view

##### Returns

view FreeTrial

#### 6.10.2.3 HR()

```
IActionResult DotNetIdentity.Controllers.ClaimBasedController.HR ( )
```

Controller action for HR view

##### Returns

view HR

#### 6.10.2.4 Software()

```
ActionResult DotNetIdentity.Controllers.ClaimBasedController.Software ( )
```

Controller action for Software view

##### Returns

view Software

#### 6.10.2.5 Violence()

```
ActionResult DotNetIdentity.Controllers.ClaimBasedController.Violence ( )
```

Controller action for AtLeast18Policy view

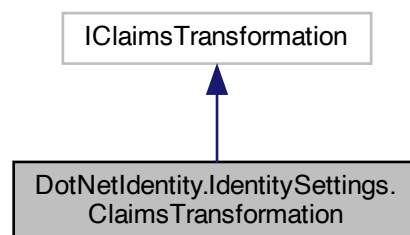
##### Returns

view AtLeast18Policy

## 6.11 DotNetIdentity.IdentitySettings.ClaimsTransformation Class Reference

class to define own claims

Inheritance diagram for DotNetIdentity.IdentitySettings.ClaimsTransformation:



### Public Member Functions

- `ClaimsTransformation` (UserManager< [AppUser](#) > userManager, RoleManager< [AppRole](#) > roleManager)  
*class constructor*
- `async Task< ClaimsPrincipal > TransformAsync` (ClaimsPrincipal principal)  
*Task to add claims to identity*

### 6.11.1 Detailed Description

class to define own claims

Definition at line 11 of file [ClaimsTransformation.cs](#).

### 6.11.2 Constructor & Destructor Documentation

#### 6.11.2.1 ClaimsTransformation()

```
DotNetIdentity.IdentitySettings.ClaimsTransformation.ClaimsTransformation (
    UserManager< AppUser > userManager,
    RoleManager< AppRole > roleManager ) [inline]
```

class constructor

##### Parameters

<i>userManager</i>	UserManager
<i>roleManager</i>	RoleManager

Definition at line 27 of file [ClaimsTransformation.cs](#).

```
00028     {
00029         _userManager = userManager;
00030         _roleManager = roleManager;
00031     }
```

### 6.11.3 Member Function Documentation

#### 6.11.3.1 TransformAsync()

```
async Task< ClaimsPrincipal > DotNetIdentity.IdentitySettings.ClaimsTransformation.Transform↵
Async (
    ClaimsPrincipal principal ) [inline]
```

Task to add claims to identoty

##### Parameters

<i>principal</i>	ClaimsPrincipal
------------------	-----------------

##### Returns

ClaimsPrincipal

Definition at line 38 of file [ClaimsTransformation.cs](#).

```

00039     {
00040         var identity = principal.Identity as ClaimsIdentity;
00041         var user = await _userManager.FindByNameAsync(identity?.Name);
00042         if (user != null)
00043         {
00044             if (!principal.HasClaim(c => c.Type == ClaimTypes.Gender))
00045             {
00046                 var genderClaim = new Claim(ClaimTypes.Gender, Enum.GetName(user.Gender!));
00047                 identity?.AddClaim(genderClaim);
00048             }
00049
00050             if (!principal.HasClaim(c => c.Type == ClaimTypes.DateOfBirth))
00051             {
00052                 var birthDayClaim = new Claim(ClaimTypes.DateOfBirth,
00053                     user.BirthDay.ToShortDateString());
00054                 identity?.AddClaim(birthDayClaim);
00055             }
00056
00057             if (!principal.HasClaim(c => c.Type == "FreeTrial"))
00058             {
00059                 var freeTrialClaim = new Claim("FreeTrial", user.CreatedOn.ToShortDateString());
00060                 identity?.AddClaim(freeTrialClaim);
00061             }
00062
00063             if (!principal.HasClaim(c => c.Type == "Department") &&
00064                 !String.IsNullOrEmpty(user.Department))
00065             {
00066                 var departmentClaim = new Claim("Department", user.Department!);
00067                 identity?.AddClaim(departmentClaim);
00068             }
00069
00070             if (!principal.HasClaim(c => c.Type == ClaimTypes.GivenName))
00071             {
00072                 var givennameClaim = new Claim(ClaimTypes.GivenName, user.FirstName!);
00073                 identity?.AddClaim(givennameClaim);
00074             }
00075
00076             if (!principal.HasClaim(c => c.Type == ClaimTypes.Surname))
00077             {
00078                 var surnameClaim = new Claim(ClaimTypes.Surname, user.LastName!);
00079                 identity?.AddClaim(surnameClaim);
00080             }
00081
00082             if(!principal.HasClaim(c=>c.Type=="SelectedCulture")) {
00083                 var culture =
00084                     System.Globalization.CultureInfo.CurrentCulture.ToString().ToLower();
00085                 var cultureClaim = new Claim("SelectedCulture", culture);
00086                 identity?.AddClaim(cultureClaim);
00087             }
00088
00089             if (!principal.HasClaim(c => c.Type == "RolesCombined"))
00090             {
00091                 var rls = await _userManager.GetRolesAsync(user);
00092                 if (rls != null)
00093                 {
00094                     var rlsComb = "";
00095                     if(rls.Count()>0 && rls.Count()<=1) {
00096                         rlsComb = rls[0].ToString();
00097                     } else {
00098                         for(int i=0;i<rls.Count;i++)
00099                         {
00100                             if(i+1!=rls.Count){ rlsComb += rls[i] + ","; }
00101                             else { rlsComb += rls[i]; }
00102                         }
00103                     }
00104                     if(rlsComb.Length>0)
00105                     {
00106                         var rolesCombinedClaim = new Claim("RolesCombined", rlsComb!);
00107                         identity?.AddClaim(rolesCombinedClaim);
00108                     }
00109                 }
00110             }
00111         }
00112     }
00113 }

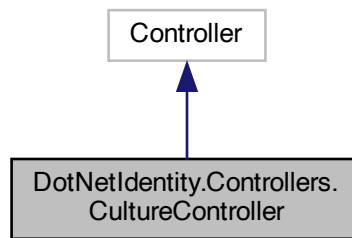
```

## 6.12 DotNetIdentity.Controllers.CultureController Class Reference

controller for culture settings



Inheritance diagram for DotNetIdentity.Controllers.CultureController:



## Public Member Functions

- IActionResult [SetCulture](#) (string culture, string returnUrl)  
*controller POST method to set current culture*

### 6.12.1 Detailed Description

controller for culture settings

Definition at line 17 of file [CultureController.cs](#).

### 6.12.2 Member Function Documentation

#### 6.12.2.1 SetCulture()

```
IActionResult DotNetIdentity.Controllers.CultureController.SetCulture (  
    string culture,  
    string returnUrl ) [inline]
```

controller POST method to set current culture

#### Parameters

<i>culture</i>	the culture as string
<i>returnUrl</i>	the return url as string

#### Returns

a local redirect to variable returnUrl

Definition at line 26 of file [CultureController.cs](#).

```
00027     {
00028         Response.Cookies.Append(
00029             CookieRequestCultureProvider.DefaultCookieName,
00030             CookieRequestCultureProvider.MakeCookieValue(new RequestCulture(culture)),
00031             new CookieOptions { Expires = DateTimeOffset.UtcNow.AddYears(1) }
00032         );
00033         return LocalRedirect(returnUrl);
00034     }
```

## 6.13 DotNetIdentity.Models.ViewModels.EditUserViewModel Class Reference

model for EditUser view

### Properties

- string **Id** = default! [get, set]  
*the user id*
- string **UserName** = default! [get, set]  
*the username*
- string **Email** = default! [get, set]  
*the email*
- string **PhoneNumber** = default! [get, set]  
*the phone number*
- **Gender** **Gender** = **Gender.Unknown** [get, set]  
*the gener*
- **Department** **Department** = **Department.Software** [get, set]  
*the department*
- List< **AssignRoleViewModel** > **Roles** = new() [get, set]  
*the roles assigned*
- string **FirstName** = string.Empty [get, set]  
*the firstname*
- string **LastName** = string.Empty [get, set]  
*the lastname*

### 6.13.1 Detailed Description

model for EditUser view

Definition at line 6 of file [EditUserViewModel.cs](#).

### 6.13.2 Property Documentation

### 6.13.2.1 Department

`Department` DotNetIdentity.Models.ViewModels.EditUserViewModel.Department = `Department.Software` [get], [set]

the department

Department

Definition at line 37 of file [EditUserViewModel.cs](#).

```
00037 { get; set; } = Department.Software;
```

Referenced by [DotNetIdentity.Controllers.AdminController.EditUser\(\)](#).

### 6.13.2.2 Email

`string` DotNetIdentity.Models.ViewModels.EditUserViewModel.Email = `default!` [get], [set]

the email

string

Definition at line 22 of file [EditUserViewModel.cs](#).

```
00022 { get; set; } = default!;
```

Referenced by [DotNetIdentity.Controllers.AdminController.EditUser\(\)](#).

### 6.13.2.3 FirstName

`string` DotNetIdentity.Models.ViewModels.EditUserViewModel.FirstName = `string.Empty` [get], [set]

the firstname

string

Definition at line 47 of file [EditUserViewModel.cs](#).

```
00047 { get; set; } = string.Empty;
```

Referenced by [DotNetIdentity.Controllers.AdminController.EditUser\(\)](#).

### 6.13.2.4 Gender

`Gender` DotNetIdentity.Models.ViewModels.EditUserViewModel.Gender = `Gender.Unknown` [get], [set]

the gener

Gender

Definition at line 32 of file [EditUserViewModel.cs](#).

```
00032 { get; set; } = Gender.Unknown;
```

Referenced by [DotNetIdentity.Controllers.AdminController.EditUser\(\)](#).

#### 6.13.2.5 Id

```
string DotNetIdentity.Models.ViewModels.EditUserViewModel.Id = default! [get], [set]
```

the user id

string

Definition at line 12 of file [EditUserViewModel.cs](#).

```
00012 { get; set; } = default!;
```

Referenced by [DotNetIdentity.Controllers.AdminController.EditUser\(\)](#).

#### 6.13.2.6 LastName

```
string DotNetIdentity.Models.ViewModels.EditUserViewModel.LastName = string.Empty [get], [set]
```

the lastname

string

Definition at line 52 of file [EditUserViewModel.cs](#).

```
00052 {get; set;} = string.Empty;
```

Referenced by [DotNetIdentity.Controllers.AdminController.EditUser\(\)](#).

#### 6.13.2.7 PhoneNumber

```
string DotNetIdentity.Models.ViewModels.EditUserViewModel.PhoneNumber = default! [get], [set]
```

the phone number

string

Definition at line 27 of file [EditUserViewModel.cs](#).

```
00027 { get; set; } = default!;
```

Referenced by [DotNetIdentity.Controllers.AdminController.EditUser\(\)](#).

#### 6.13.2.8 Roles

```
List<AssignRoleViewModel> DotNetIdentity.Models.ViewModels.EditUserViewModel.Roles = new()  
[get], [set]
```

the roles assigned

##### Returns

List of type AppRole

Definition at line 42 of file [EditUserViewModel.cs](#).

```
00042 { get; set; } = new();
```

Referenced by [DotNetIdentity.Controllers.AdminController.EditUser\(\)](#).

### 6.13.2.9 UserName

```
string DotNetIdentity.Models.ViewModels.EditUserViewModel.UserName = default! [get], [set]
```

the username

string

Definition at line 17 of file [EditUserViewModel.cs](#).

```
00017 { get; set; } = default!;
```

Referenced by [DotNetIdentity.Controllers.AdminController.EditUser\(\)](#).

## 6.14 DotNetIdentity.Helpers.EmailHelper Class Reference

class for email methods

### Public Member Functions

- [EmailHelper](#) (ISettingsService sett)  
*class constructor*
- async Task [SendAsync](#) ([EmailModel](#) model)  
*method to send a mail*

### 6.14.1 Detailed Description

class for email methods

Definition at line 12 of file [EmailHelper.cs](#).

### 6.14.2 Constructor & Destructor Documentation

#### 6.14.2.1 EmailHelper()

```
DotNetIdentity.Helpers.EmailHelper.EmailHelper (  
    ISettingsService sett ) [inline]
```

class constructor

Parameters

<i>sett</i>	type ISettingsService
-------------	-----------------------

Definition at line 23 of file [EmailHelper.cs](#).

```

00024         {
00025             _sett = sett;
00026         }

```

## 6.14.3 Member Function Documentation

### 6.14.3.1 SendAsync()

```

async Task DotNetIdentity.Helpers.EmailHelper.SendAsync (
    EmailModel model ) [inline]

```

method to send a mail

#### Parameters

<i>model</i>	type EmailModel
--------------	-----------------

#### Returns

Definition at line 33 of file [EmailHelper.cs](#).

```

00034     {
00035         var mail = new MailMessage
00036         {
00037             Subject = model.Subject,
00038             Body = model.Body,
00039             From = new MailAddress(_sett.Mail.SmtpFromAddress!),
00040             IsBodyHtml = true
00041         };
00042
00043         var smtp = new SmtpClient
00044         {
00045             Host = _sett.Mail.SmtpServer!,
00046             Port = Convert.ToInt32(_sett.Mail.SmtpPort),
00047             EnableSsl = Convert.ToBoolean(_sett.Mail.SmtpUseTls),
00048             DeliveryMethod = SmtpDeliveryMethod.Network,
00049             UseDefaultCredentials = false,
00050             Credentials = new NetworkCredential(_sett.Mail.Username, _sett.Mail.Password)
00051         };
00052
00053         mail.To.Add(model.To);
00054
00055         await smtp.SendMailAsync(mail);
00056     }

```

References [DotNetIdentity.Models.BusinessModels.EmailModel.Body](#), [DotNetIdentity.Models.BusinessModels.EmailModel.Subject](#), and [DotNetIdentity.Models.BusinessModels.EmailModel.To](#).

## 6.15 DotNetIdentity.Models.BusinessModels.EmailModel Class Reference

class to define a email

## Properties

- string [Subject](#) = default! [get, set]  
*property subject*
- string [Body](#) = default! [get, set]  
*property body*
- string [To](#) = default! [get, set]  
*property to*

### 6.15.1 Detailed Description

class to define a email

Definition at line 6 of file [EmailModel.cs](#).

### 6.15.2 Property Documentation

#### 6.15.2.1 Body

```
string DotNetIdentity.Models.BusinessModels.EmailModel.Body = default! [get], [set]
```

property body

string

Definition at line 17 of file [EmailModel.cs](#).

```
00017 { get; set; } = default!;
```

Referenced by [DotNetIdentity.Helpers.EmailHelper.SendAsync\(\)](#).

#### 6.15.2.2 Subject

```
string DotNetIdentity.Models.BusinessModels.EmailModel.Subject = default! [get], [set]
```

property subject

string

Definition at line 12 of file [EmailModel.cs](#).

```
00012 { get; set; } = default!;
```

Referenced by [DotNetIdentity.Helpers.EmailHelper.SendAsync\(\)](#).

### 6.15.2.3 To

```
string DotNetIdentity.Models.BusinessModels.EmailModel.To = default! [get], [set]
```

property to

string

Definition at line 22 of file [EmailModel.cs](#).

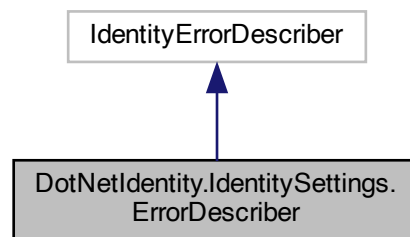
```
00022 { get; set; } = default!;
```

Referenced by [DotNetIdentity.Helpers.EmailHelper.SendAsync\(\)](#).

## 6.16 DotNetIdentity.IdentitySettings.ErrorDescriber Class Reference

class to override IdentityErrorDescriber IdentityError messages

Inheritance diagram for DotNetIdentity.IdentitySettings.ErrorDescriber:



### Static Public Member Functions

- static IdentityError [PasswordContainsUsername](#) ()  
*custom IdentityError if password contains username*
- static IdentityError [UserNameLength](#) ()  
*custom IdentityError if username length is too short*
- static IdentityError [UserNameContainsEmail](#) ()  
*custom IdentityError if username contains user email*

### 6.16.1 Detailed Description

class to override IdentityErrorDescriber IdentityError messages

Definition at line 8 of file [ErrorDescriber.cs](#).



## 6.16.2 Member Function Documentation

### 6.16.2.1 PasswordContainsUsername()

```
static IdentityError DotNetIdentity.IdentitySettings.ErrorDescriber.PasswordContainsUsername (
) [static]
```

custom IdentityError if password contains username

#### Returns

IdentityError

Referenced by [DotNetIdentity.IdentitySettings.Validators.PasswordValidator.ValidateAsync\(\)](#).

### 6.16.2.2 UserNameContainsEmail()

```
static IdentityError DotNetIdentity.IdentitySettings.ErrorDescriber.UserNameContainsEmail ( )
[static]
```

custom IdentityError if username contains user email

#### Returns

IdentityError

Referenced by [DotNetIdentity.IdentitySettings.Validators.UserValidator.ValidateAsync\(\)](#).

### 6.16.2.3 UserNameLength()

```
static IdentityError DotNetIdentity.IdentitySettings.ErrorDescriber.UserNameLength ( ) [static]
```

custom IdentityError if username length is too short

#### Returns

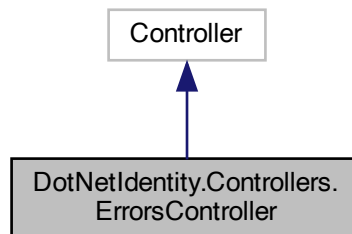
IdentityError

Referenced by [DotNetIdentity.IdentitySettings.Validators.UserValidator.ValidateAsync\(\)](#).

## 6.17 DotNetIdentity.Controllers.ErrorsController Class Reference

Controller for error views

Inheritance diagram for DotNetIdentity.Controllers.ErrorsController:



### Public Member Functions

- IActionResult [ErrorEx](#) ()  
*controller action for ErrorEx view*
- IActionResult [ErrorCd](#) (int statusCode)  
*Controller action for ErrorCd view*

### 6.17.1 Detailed Description

Controller for error views

**Todo** add translation

Definition at line 17 of file [ErrorsController.cs](#).

### 6.17.2 Member Function Documentation

#### 6.17.2.1 ErrorCd()

```
IActionResult DotNetIdentity.Controllers.ErrorsController.ErrorCd (  
    int statusCode ) [inline]
```

Controller action for ErrorCd view

## Parameters

<i>statusCode</i>	
-------------------	--

## Returns

view ErrorCd

Definition at line 45 of file [ErrorsController.cs](#).

```

00046     {
00047         var statusCodeData = HttpContext.Features.Get<IStatusCodeReExecuteFeature>();
00048         StatusCodesModel md = new StatusCodesModel();
00049         md.StatusCode = statusCode;
00050         md.RouteOfException = statusCodeData!.OriginalPath;
00051
00052         switch (statusCode)
00053         {
00054             // request timeout
00055             case 408:
00056                 md.ErrorMessage = "Timeout. The request timed out..";
00057                 break;
00058             // badrequest
00059             case 400:
00060                 md.ErrorMessage = "Badrequest. The request meets not the controllers expetation.";
00061                 break;
00062             // forbidden
00063             case 403:
00064                 md.ErrorMessage = "Access denied!";
00065                 break;
00066             // unauthorized
00067             case 401:
00068                 md.ErrorMessage = "You're not authorized to access this resource!";
00069                 break;
00070             //page not found
00071             case 404:
00072                 md.ErrorMessage = "Sorry the page you requested could not be found";
00073
00074                 break;
00075             // server error
00076             case 500:
00077                 md.ErrorMessage = "Sorry something went wrong on the server";
00078                 break;
00079             // service unavailable
00080             case 503:
00081                 md.ErrorMessage = "Sorry, the service is currently unavailable.";
00082                 break;
00083             default:
00084                 md.ErrorMessage = "Sorry, there was an error.";
00085                 break;
00086         }
00087         return View("ErrorCd", md);
00088     }

```

## 6.17.2.2 ErrorEx()

`IActionResult DotNetIdentity.Controllers.ErrorsController.ErrorEx ( ) [inline]`

controller action for ErrorEx view

## Returns

view ErrorEx

Definition at line 23 of file [ErrorsController.cs](#).

```
00024     {
00025         var exceptionFeature = HttpContext.Features.Get<ExceptionHandlerPathFeature>();
00026
00027         if (exceptionFeature != null)
00028         {
00029             if(exceptionFeature.Error.StackTrace!=null) {
00030                 ViewBag.StackTrace = exceptionFeature.Error.StackTrace;
00031             }
00032             ViewBag.ErrorMessage = exceptionFeature.Error.Message;
00033             ViewBag.RouteOfException = exceptionFeature.Path;
00034         }
00035
00036         return View();
00037     }
```

## 6.18 DotNetIdentity.Models.ViewModels.ErrorViewModel Class Reference

model for displaying errors

### Properties

- string? [RequestId](#) [get, set]  
*the request id*
- bool [ShowRequestId](#) [get]  
*show the request id*

### 6.18.1 Detailed Description

model for displaying errors

Definition at line 5 of file [ErrorViewModel.cs](#).

### 6.18.2 Property Documentation

#### 6.18.2.1 RequestId

string? DotNetIdentity.Models.ViewModels.ErrorViewModel.RequestId [get], [set]

the request id

string

Definition at line 11 of file [ErrorViewModel.cs](#).

```
00011 { get; set; }
```

### 6.18.2.2 ShowRequestId

`bool DotNetIdentity.Models.ViewModels.ErrorViewModel.ShowRequestId [get]`

show the request id

#### Returns

`bool`

Definition at line 16 of file [ErrorViewModel.cs](#).

## 6.19 DotNetIdentity.Models.ViewModels.ForgotPasswordViewModel Class Reference

model for ForgotPassword view

### Properties

- `string Email = default! [get, set]`  
*the user email*

### 6.19.1 Detailed Description

model for ForgotPassword view

Definition at line 6 of file [ForgotPasswordViewModel.cs](#).

### 6.19.2 Property Documentation

#### 6.19.2.1 Email

`string DotNetIdentity.Models.ViewModels.ForgotPasswordViewModel.Email = default! [get], [set]`

the user email

`string`

Definition at line 12 of file [ForgotPasswordViewModel.cs](#).

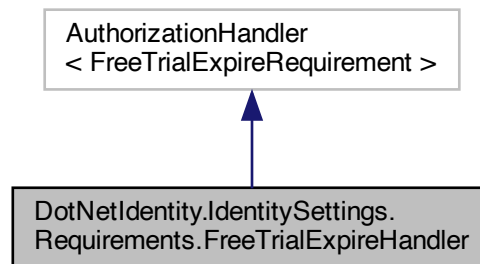
```
00012 { get; set; } = default!;
```

Referenced by [DotNetIdentity.Controllers.UserController.ForgotPassword\(\)](#).

## 6.20 DotNetIdentity.IdentitySettings.Requirements.FreeTrialExpireHandler Class Reference

class to define own Identity requirements type of [FreeTrialExpireRequirement](#)

Inheritance diagram for DotNetIdentity.IdentitySettings.Requirements.FreeTrialExpireHandler:



### Protected Member Functions

- override Task [HandleRequirementAsync](#) (AuthorizationHandlerContext context, [FreeTrialExpireRequirement](#) requirement)  
*overriding HandleRequirementAsync*

#### 6.20.1 Detailed Description

class to define own Identity requirements type of [FreeTrialExpireRequirement](#)

Definition at line 13 of file [FreeTrialExpireRequirement.cs](#).

#### 6.20.2 Member Function Documentation

##### 6.20.2.1 HandleRequirementAsync()

```

override Task DotNetIdentity.IdentitySettings.Requirements.FreeTrialExpireHandler.Handle↵
RequirementAsync (
    AuthorizationHandlerContext context,
    FreeTrialExpireRequirement requirement ) [inline], [protected]
  
```

overriding HandleRequirementAsync

## Parameters

<i>context</i>	AuthorizationHandlerContext
<i>requirement</i>	<a href="#">FreeTrialExpireRequirement</a>

## Returns

Task

Definition at line 21 of file [FreeTrialExpireRequirement.cs](#).

```

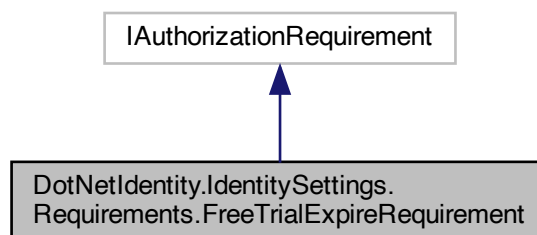
00022     {
00023         var freeTrialClaim = context.User.FindFirst(f => f.Type == "FreeTrial");
00024         if (freeTrialClaim != null)
00025         {
00026             var freeTrialExpires = Convert.ToDateTime(freeTrialClaim.Value).AddDays(30);
00027             if (DateTime.Now < freeTrialExpires)
00028             {
00029                 context.Succeed(requirement);
00030             }
00031             else
00032             {
00033                 context.Fail();
00034             }
00035         }
00036         return Task.CompletedTask;
00037     }
00038 }

```

## 6.21 DotNetIdentity.IdentitySettings.Requirements.FreeTrialExpireRequirement Class Reference

class to define own Identity requirements type of `IAuthorizationRequirement`

Inheritance diagram for `DotNetIdentity.IdentitySettings.Requirements.FreeTrialExpireRequirement`:



### 6.21.1 Detailed Description

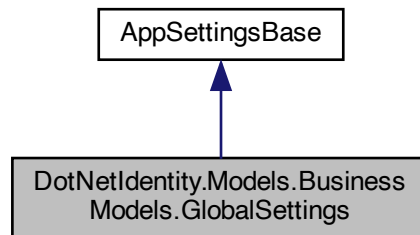
class to define own Identity requirements type of `IAuthorizationRequirement`

Definition at line 8 of file [FreeTrialExpireRequirement.cs](#).

## 6.22 DotNetIdentity.Models.BusinessModels.GlobalSettings Class Reference

class to define a global settings object

Inheritance diagram for DotNetIdentity.Models.BusinessModels.GlobalSettings:



### Properties

- string? [ApplicationName](#) [get, set]  
*property ApplicationName*
- string? [SessionTimeoutWarnAfter](#) [get, set]  
*property SessionTimeoutWarnAfter*
- string? [SessionTimeoutRedirAfter](#) [get, set]  
*property SessionTimeoutRedirAfter*
- string? [SessionCookieExpiration](#) [get, set]  
*property SessionCookieExpiration*

### Additional Inherited Members

#### 6.22.1 Detailed Description

class to define a global settings object

Definition at line 7 of file [GlobalSettings.cs](#).

#### 6.22.2 Property Documentation



### 6.22.2.1 ApplicationName

```
string? DotNetIdentity.Models.BusinessModels.GlobalSettings.ApplicationName [get], [set]
```

property ApplicationName

string

Definition at line 12 of file [GlobalSettings.cs](#).

```
00012 {get; set;}
```

### 6.22.2.2 SessionCookieExpiration

```
string? DotNetIdentity.Models.BusinessModels.GlobalSettings.SessionCookieExpiration [get],  
[set]
```

property SessionCookieExpiration

string

Definition at line 27 of file [GlobalSettings.cs](#).

```
00027 {get; set;}
```

### 6.22.2.3 SessionTimeoutRedirAfter

```
string? DotNetIdentity.Models.BusinessModels.GlobalSettings.SessionTimeoutRedirAfter [get],  
[set]
```

property SessionTimeoutRedirAfter

string

Definition at line 22 of file [GlobalSettings.cs](#).

```
00022 {get; set;}
```

### 6.22.2.4 SessionTimeoutWarnAfter

```
string? DotNetIdentity.Models.BusinessModels.GlobalSettings.SessionTimeoutWarnAfter [get],  
[set]
```

property SessionTimeoutWarnAfter

string

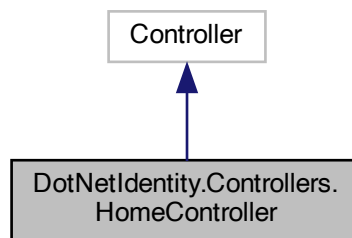
Definition at line 17 of file [GlobalSettings.cs](#).

```
00017 {get; set;}
```

## 6.23 DotNetIdentity.Controllers.HomeController Class Reference

Controller for Home views

Inheritance diagram for DotNetIdentity.Controllers.HomeController:



### Public Member Functions

- [HomeController](#) (UserManager< [AppUser](#) > userManager)  
*Controller constructor*
- IActionResult [Index](#) ()  
*Controller action for view Index*
- IActionResult [AccessDenied](#) ()  
*controller action for view AccessDenied*

### 6.23.1 Detailed Description

Controller for Home views

Definition at line 11 of file [HomeController.cs](#).

### 6.23.2 Constructor & Destructor Documentation

#### 6.23.2.1 HomeController()

```
DotNetIdentity.Controllers.HomeController.HomeController (
    UserManager< AppUser > userManager ) [inline]
```

Controller constructor

**Parameters**

<i>userManager</i>	Property UserManager
--------------------	----------------------

Definition at line 22 of file [HomeController.cs](#).

```
00023     {  
00024         _userManager = userManager;  
00025     }
```

### 6.23.3 Member Function Documentation

#### 6.23.3.1 AccessDenied()

```
IActionResult DotNetIdentity.Controllers.HomeController.AccessDenied ( )
```

controller action for view AccessDenied

**Returns**

view AccessDenied

#### 6.23.3.2 Index()

```
IActionResult DotNetIdentity.Controllers.HomeController.Index ( )
```

Controller action for view Index

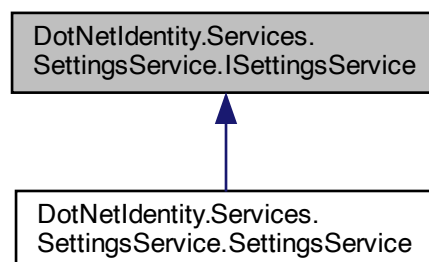
**Returns**

view Index

## 6.24 DotNetIdentity.Services.SettingsService.ISettingsService Interface Reference

Interface class for [SettingsService](#)

Inheritance diagram for DotNetIdentity.Services.SettingsService.ISettingsService:



## Public Member Functions

- void [Save](#) ()  
*save method*

## Properties

- [GlobalSettings](#) [Global](#) [get]  
*property global settings*
- [MailSettings](#) [Mail](#) [get]  
*property mail settings*

### 6.24.1 Detailed Description

Interface class for [SettingsService](#)

Definition at line 7 of file [ISettingsService.cs](#).

### 6.24.2 Member Function Documentation

#### 6.24.2.1 Save()

```
void DotNetIdentity.Services.SettingsService.ISettingsService.Save ( )
```

save method

Implemented in [DotNetIdentity.Services.SettingsService.SettingsService](#).

### 6.24.3 Property Documentation

#### 6.24.3.1 Global

```
GlobalSettings DotNetIdentity.Services.SettingsService.ISettingsService.Global [get]
```

property global settings

GlobalSettings

Implemented in [DotNetIdentity.Services.SettingsService.SettingsService](#).

Definition at line 12 of file [ISettingsService.cs](#).

```
00012 { get; }
```

### 6.24.3.2 Mail

`MailSettings` `DotNetIdentity.Services.SettingsService.ISettingsService.Mail` [get]

property mail settings

MailSettings

Implemented in `DotNetIdentity.Services.SettingsService.SettingsService`.

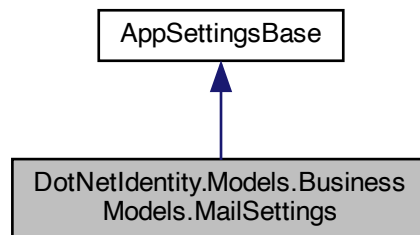
Definition at line 17 of file `ISettingsService.cs`.

```
00017 { get; }
```

## 6.25 DotNetIdentity.Models.BusinessModels.MailSettings Class Reference

class to define a ail settings object

Inheritance diagram for `DotNetIdentity.Models.BusinessModels.MailSettings`:



### Properties

- string? `Username` [get, set]  
*property Username*
- string? `Password` [get, set]  
*property Password*
- string? `SmtpServer` [get, set]  
*property SmtpServer*
- string? `SmtpPort` [get, set]  
*property SmtpPort*
- string? `SmtpFromAddress` [get, set]  
*property SmtpFromAddress*
- bool `SmtpUseTls` [get, set]  
*property SmtpUseTls*

## Additional Inherited Members

### 6.25.1 Detailed Description

class to define a ail settings object

Definition at line 7 of file [MailSettings.cs](#).

### 6.25.2 Property Documentation

#### 6.25.2.1 Password

```
string? DotNetIdentity.Models.BusinessModels.MailSettings.Password [get], [set]
```

property Password

string

Definition at line 17 of file [MailSettings.cs](#).

```
00017 { get; set; }
```

#### 6.25.2.2 SmtptFromAddress

```
string? DotNetIdentity.Models.BusinessModels.MailSettings.SmtptFromAddress [get], [set]
```

property SmtptFromAddress

string

Definition at line 32 of file [MailSettings.cs](#).

```
00032 {get; set;}
```

#### 6.25.2.3 SmtptPort

```
string? DotNetIdentity.Models.BusinessModels.MailSettings.SmtptPort [get], [set]
```

property SmtptPort

string

Definition at line 27 of file [MailSettings.cs](#).

```
00027 { get; set; }
```

#### 6.25.2.4 SmtpServer

string? DotNetIdentity.Models.BusinessModels.MailSettings.SmtpServer [get], [set]

property SmtpServer

string

Definition at line 22 of file [MailSettings.cs](#).

```
00022 { get; set; }
```

#### 6.25.2.5 SmtpUseTls

bool DotNetIdentity.Models.BusinessModels.MailSettings.SmtpUseTls [get], [set]

property SmtpUseTls

bool

Definition at line 37 of file [MailSettings.cs](#).

```
00037 { get; set; }
```

#### 6.25.2.6 Username

string? DotNetIdentity.Models.BusinessModels.MailSettings.Username [get], [set]

property Username

string

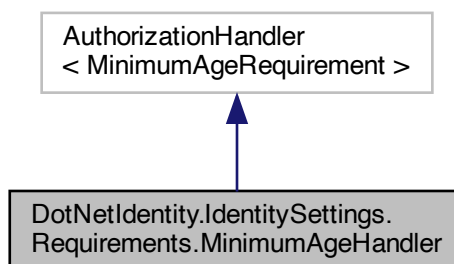
Definition at line 12 of file [MailSettings.cs](#).

```
00012 { get; set; }
```

## 6.26 DotNetIdentity.IdentitySettings.Requirements.MinimumAgeHandler Class Reference

class implementing AuthorizationHandler of type [MinimumAgeRequirement](#)

Inheritance diagram for DotNetIdentity.IdentitySettings.Requirements.MinimumAgeHandler:



## Protected Member Functions

- override Task [HandleRequirementAsync](#) (AuthorizationHandlerContext context, [MinimumAgeRequirement](#) requirement)  
*overriding HandleRequirementAsync*

### 6.26.1 Detailed Description

class implementing AuthorizationHandler of type [MinimumAgeRequirement](#)

Definition at line 30 of file [MinimumAgeRequirement.cs](#).

### 6.26.2 Member Function Documentation

#### 6.26.2.1 HandleRequirementAsync()

```
override Task DotNetIdentity.IdentitySettings.Requirements.MinimumAgeHandler.HandleRequirement↵
Async (
```

```
    AuthorizationHandlerContext context,
    MinimumAgeRequirement requirement ) [inline], [protected]
```

overriding HandleRequirementAsync

#### Parameters

<i>context</i>	AuthorizationHandlerContext
<i>requirement</i>	<a href="#">MinimumAgeRequirement</a>

#### Returns

Task

Definition at line 38 of file [MinimumAgeRequirement.cs](#).

```
00039     {
00040         var birthDayClaim = context.User.FindFirst(f => f.Type == ClaimTypes.DateOfBirth);
00041         if (birthDayClaim != null)
00042         {
00043             var birthDay = Convert.ToDateTime(birthDayClaim.Value);
00044             var calculatedAge = DateTime.Now.Year - birthDay.Year;
00045             if (calculatedAge >= requirement.MinimumAge)
00046             {
00047                 context.Succeed(requirement);
00048             }
00049             else
00050             {
00051                 context.Fail();
00052             }
00053         }
00054         return Task.CompletedTask;
00055     }
```

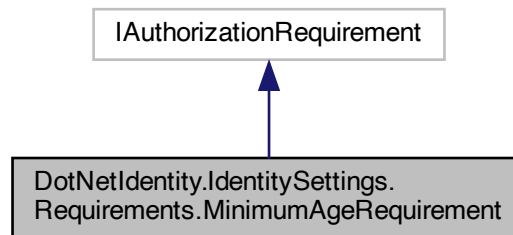
References [DotNetIdentity.IdentitySettings.Requirements.MinimumAgeRequirement.MinimumAge](#).



## 6.27 DotNetIdentity.IdentitySettings.Requirements.MinimumAgeRequirement Class Reference

class to define own Identity requirements type of IAuthorizationRequirement

Inheritance diagram for DotNetIdentity.IdentitySettings.Requirements.MinimumAgeRequirement:



### Public Member Functions

- [MinimumAgeRequirement](#) (int minimumAge)  
*class constructor*

### Properties

- int [MinimumAge](#) [get, set]  
*property MinimumAge*

#### 6.27.1 Detailed Description

class to define own Identity requirements type of IAuthorizationRequirement

Definition at line 9 of file [MinimumAgeRequirement.cs](#).

#### 6.27.2 Constructor & Destructor Documentation

##### 6.27.2.1 MinimumAgeRequirement()

```
DotNetIdentity.IdentitySettings.Requirements.MinimumAgeRequirement.MinimumAgeRequirement (
    int minimumAge ) [inline]
```

class constructor

**Parameters**

<i>minimumAge</i>	
-------------------	--

Definition at line 15 of file [MinimumAgeRequirement.cs](#).

```
00016     {
00017         MinimumAge = minimumAge;
00018     }
```

References [DotNetIdentity.IdentitySettings.Requirements.MinimumAgeRequirement.MinimumAge](#).

**6.27.3 Property Documentation****6.27.3.1 MinimumAge**

```
int DotNetIdentity.IdentitySettings.Requirements.MinimumAgeRequirement.MinimumAge [get], [set]
```

property MinimumAge

Definition at line 24 of file [MinimumAgeRequirement.cs](#).

```
00024 { get; set; }
```

Referenced by [DotNetIdentity.IdentitySettings.Requirements.MinimumAgeHandler.HandleRequirementAsync\(\)](#), and [DotNetIdentity.IdentitySettings.Requirements.MinimumAgeRequirement.MinimumAgeRequirement\(\)](#).

**6.28 DotNetIdentity.Models.ViewModels.NewUserModel Class Reference**

model for NewUser view

**Properties**

- string [UserName](#) = default! [get, set]  
*the username*
- string [Email](#) = default! [get, set]  
*the users email address*
- [Gender](#) [Gender](#) = [Gender.Unknown](#) [get, set]  
*the gender*
- [Department](#) [Department](#) = [Department.Software](#) [get, set]  
*the department*
- List< [AssignRoleViewModel](#) > [Roles](#) = new() [get, set]  
*the roles assigned*
- string [FirstName](#) = string.Empty [get, set]  
*the firstname*
- string [LastName](#) = string.Empty [get, set]  
*the lastname*
- bool [IsLdapLogin](#) [get, set]  
*if user is ldap login*
- bool [IsMfaForce](#) [get, set]  
*is 2fa enforced*
- string [Password](#) = string.Empty [get, set]  
*the password*

## 6.28.1 Detailed Description

model for NewUser view

Definition at line 9 of file [NewUserModel.cs](#).

## 6.28.2 Property Documentation

### 6.28.2.1 Department

```
Department DotNetIdentity.Models.ViewModels.NewUserModel.Department = Department.Software  
[get], [set]
```

the department

Department

Definition at line 30 of file [NewUserModel.cs](#).

```
00030 { get; set; } = Department.Software;
```

Referenced by [DotNetIdentity.Controllers.AdminController.NewUser\(\)](#).

### 6.28.2.2 Email

```
string DotNetIdentity.Models.ViewModels.NewUserModel.Email = default! [get], [set]
```

the users email address

string

Definition at line 20 of file [NewUserModel.cs](#).

```
00020 { get; set; } = default!;
```

Referenced by [DotNetIdentity.Controllers.AdminController.NewUser\(\)](#).

### 6.28.2.3 FirstName

```
string DotNetIdentity.Models.ViewModels.NewUserModel.FirstName = string.Empty [get], [set]
```

the firstname

string

Definition at line 40 of file [NewUserModel.cs](#).

```
00040 { get; set; } = string.Empty;
```

Referenced by [DotNetIdentity.Controllers.AdminController.NewUser\(\)](#).

#### 6.28.2.4 Gender

`Gender` `DotNetIdentity.Models.ViewModels.NewUserModel.Gender` = `Gender.Unknown` [get], [set]

the gender

Gender

Definition at line 25 of file [NewUserModel.cs](#).

```
00025 { get; set; } = Gender.Unknown;
```

Referenced by [DotNetIdentity.Controllers.AdminController.NewUser\(\)](#).

#### 6.28.2.5 IsLdapLogin

`bool` `DotNetIdentity.Models.ViewModels.NewUserModel.IsLdapLogin` [get], [set]

if user is ldap login

bool

Definition at line 50 of file [NewUserModel.cs](#).

```
00050 { get; set; }
```

Referenced by [DotNetIdentity.Controllers.AdminController.NewUser\(\)](#).

#### 6.28.2.6 IsMfaForce

`bool` `DotNetIdentity.Models.ViewModels.NewUserModel.IsMfaForce` [get], [set]

is 2fa enforced

bool

Definition at line 55 of file [NewUserModel.cs](#).

```
00055 { get; set; }
```

Referenced by [DotNetIdentity.Controllers.AdminController.NewUser\(\)](#).

#### 6.28.2.7 LastName

`string` `DotNetIdentity.Models.ViewModels.NewUserModel.LastName` = `string.Empty` [get], [set]

the lastname

string

Definition at line 45 of file [NewUserModel.cs](#).

```
00045 { get; set; } = string.Empty;
```

Referenced by [DotNetIdentity.Controllers.AdminController.NewUser\(\)](#).

### 6.28.2.8 Password

```
string DotNetIdentity.Models.ViewModels.NewUserModel.Password = string.Empty [get], [set]
```

the password

string

Definition at line 60 of file [NewUserModel.cs](#).

```
00060 { get; set; } = string.Empty;
```

Referenced by [DotNetIdentity.Controllers.AdminController.NewUser\(\)](#).

### 6.28.2.9 Roles

```
List<AssignRoleViewModel> DotNetIdentity.Models.ViewModels.NewUserModel.Roles = new() [get], [set]
```

the roles assigned

Returns

List of type AppRole

Definition at line 35 of file [NewUserModel.cs](#).

```
00035 { get; set; } = new();
```

Referenced by [DotNetIdentity.Controllers.AdminController.NewUser\(\)](#).

### 6.28.2.10 UserName

```
string DotNetIdentity.Models.ViewModels.NewUserModel.UserName = default! [get], [set]
```

the username

string

Definition at line 15 of file [NewUserModel.cs](#).

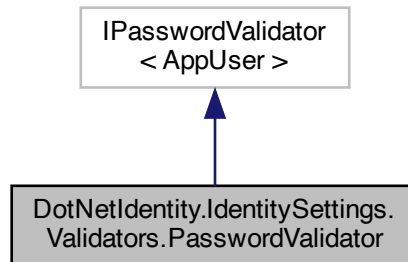
```
00015 { get; set; } = default!;
```

Referenced by [DotNetIdentity.Controllers.AdminController.NewUser\(\)](#).

## 6.29 DotNetIdentity.IdentitySettings.Validators.PasswordValidator Class Reference

class to validate a password

Inheritance diagram for DotNetIdentity.IdentitySettings.Validators.PasswordValidator:



### Public Member Functions

- Task< IdentityResult > [ValidateAsync](#) (userManager< [AppUser](#) > manager, [AppUser](#) user, string password)  
*Task to validate a password*

### 6.29.1 Detailed Description

class to validate a password

Definition at line 9 of file [PasswordValidator.cs](#).

### 6.29.2 Member Function Documentation

#### 6.29.2.1 ValidateAsync()

```

Task< IdentityResult > DotNetIdentity.IdentitySettings.Validators.PasswordValidator.ValidateAsync (
    UserManager< AppUser > manager,
    AppUser user,
    string password ) [inline]
  
```

Task to validate a password

## Parameters

<i>manager</i>	UserManager
<i>user</i>	AppUser
<i>password</i>	string

## Returns

Task of type IdentityResult

Definition at line 18 of file [PasswordValidator.cs](#).

```

00019     {
00020         var errors = new List<IdentityError>();
00021
00022         if (user.UserName == password)
00023         {
00024             errors.Add(ErrorDescriber.PasswordContainsUsername());
00025         }
00026
00027         if (errors.Any())
00028         {
00029             return Task.FromResult(IdentityResult.Failed(errors.ToArray()));
00030         }
00031         return Task.FromResult(IdentityResult.Success);
00032     }

```

References [DotNetIdentity.IdentitySettings.ErrorDescriber.PasswordContainsUsername\(\)](#).

Here is the call graph for this function:



## 6.30 DotNetIdentity.Models.ViewModels.ResetPasswordViewModel Class Reference

model for ResetPassword view

### Properties

- string [UserId](#) = default! [get, set]  
the user id
- string [Token](#) = default! [get, set]  
the reset token
- string [Password](#) = default! [get, set]  
the password

### 6.30.1 Detailed Description

model for ResetPassword view

Definition at line 6 of file [ResetPasswordViewModel.cs](#).

### 6.30.2 Property Documentation

#### 6.30.2.1 Password

```
string DotNetIdentity.Models.ViewModels.ResetPasswordViewModel.Password = default! [get],  
[set]
```

the password

string

Definition at line 22 of file [ResetPasswordViewModel.cs](#).

```
00022 { get; set; } = default!;
```

Referenced by [DotNetIdentity.Controllers.UserController.ResetPassword\(\)](#).

#### 6.30.2.2 Token

```
string DotNetIdentity.Models.ViewModels.ResetPasswordViewModel.Token = default! [get], [set]
```

the reset token

string

Definition at line 17 of file [ResetPasswordViewModel.cs](#).

```
00017 { get; set; } = default!;
```

Referenced by [DotNetIdentity.Controllers.UserController.ResetPassword\(\)](#).

#### 6.30.2.3 UserId

```
string DotNetIdentity.Models.ViewModels.ResetPasswordViewModel.UserId = default! [get], [set]
```

the user id

string

Definition at line 12 of file [ResetPasswordViewModel.cs](#).

```
00012 { get; set; } = default!;
```

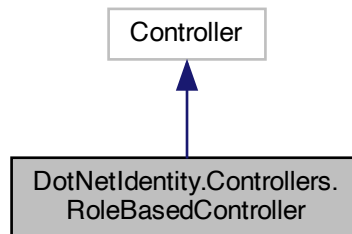
Referenced by [DotNetIdentity.Controllers.UserController.ResetPassword\(\)](#).



## 6.31 DotNetIdentity.Controllers.RoleBasedController Class Reference

Controller to test RoleBased Authorization

Inheritance diagram for DotNetIdentity.Controllers.RoleBasedController:



### Public Member Functions

- IActionResult [Dashboard](#) ()  
*Controller Action for Dashboard view*
- IActionResult [Users](#) ()  
*Controller Action for Users view*
- IActionResult [Articles](#) ()  
*Controller Action for Articles view*
- IActionResult [Profile](#) ()  
*Controller Action for Profile view*

### 6.31.1 Detailed Description

Controller to test RoleBased Authorization

Definition at line 9 of file [RoleBasedController.cs](#).

### 6.31.2 Member Function Documentation

#### 6.31.2.1 Articles()

```
IActionResult DotNetIdentity.Controllers.RoleBasedController.Articles ( )
```

Controller Action for Articles view

Returns

### 6.31.2.2 Dashboard()

```
ActionResult DotNetIdentity.Controllers.RoleBasedController.Dashboard ( )
```

Controller Action for Dashboard view

Returns

### 6.31.2.3 Profile()

```
ActionResult DotNetIdentity.Controllers.RoleBasedController.Profile ( )
```

Controller Action for Profile view

Returns

### 6.31.2.4 Users()

```
ActionResult DotNetIdentity.Controllers.RoleBasedController.Users ( )
```

Controller Action for Users view

Returns

## 6.32 DotNetIdentity.Models.DataModels.SessionCache Class Reference

class to define a [SessionCache](#) object

### Properties

- string? [id](#) [get, set]  
*property id*
- byte?[] [Value](#) [get, set]  
*property value*
- DateTime [ExpiresAtTime](#) [get, set]  
*property ExpiresAtTime*
- int [SlidingExpirationInSeconds](#) [get, set]  
*property SlidingExpirationInSeconds*
- DateTime? [AbsoluteExpiration](#) [get, set]  
*property AbsoluteExpiration*

### 6.32.1 Detailed Description

class to define a [SessionCache](#) object

Definition at line 11 of file [SessionCache.cs](#).

### 6.32.2 Property Documentation

#### 6.32.2.1 AbsoluteExpiration

DateTime? DotNetIdentity.Models.DataModels.SessionCache.AbsoluteExpiration [get], [set]

property AbsoluteExpiration

DateTime

Definition at line 37 of file [SessionCache.cs](#).

```
00037 {get; set;}
```

#### 6.32.2.2 ExpiresAtTime

DateTime DotNetIdentity.Models.DataModels.SessionCache.ExpiresAtTime [get], [set]

property ExpiresAtTime

DateTime

Definition at line 27 of file [SessionCache.cs](#).

```
00027 {get; set;}
```

#### 6.32.2.3 id

string? DotNetIdentity.Models.DataModels.SessionCache.id [get], [set]

property id

int

Definition at line 17 of file [SessionCache.cs](#).

```
00017 {get; set;}
```

#### 6.32.2.4 SlidingExpirationInSeconds

```
int DotNetIdentity.Models.DataModels.SessionCache.SlidingExpirationInSeconds [get], [set]
```

property SlidingExpirationInSeconds

int

Definition at line 32 of file [SessionCache.cs](#).

```
00032 {get; set;}
```

#### 6.32.2.5 Value

```
byte? [] DotNetIdentity.Models.DataModels.SessionCache.Value [get], [set]
```

property value

byte[]

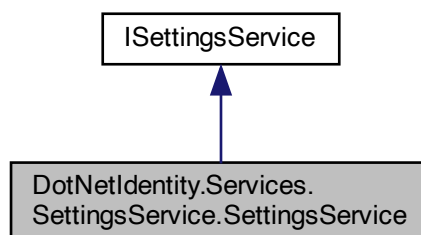
Definition at line 22 of file [SessionCache.cs](#).

```
00022 {get;set;}
```

### 6.33 DotNetIdentity.Services.SettingService.SettingService Class Reference

class for settings service

Inheritance diagram for DotNetIdentity.Services.SettingService.SettingService:



#### Public Member Functions

- [SettingService](#) ([AppDbContext](#) unitOfWork)  
*class constructor*
- void [Save](#) ()  
*sve method*

## Properties

- [GlobalSettings Global](#) [get]  
*global settings property*
- [MailSettings Mail](#) [get]  
*mail settings property*

### 6.33.1 Detailed Description

class for settings service

**Todo** add ldap settings class

Definition at line 9 of file [SettingsService.cs](#).

### 6.33.2 Constructor & Destructor Documentation

#### 6.33.2.1 SettingsService()

```
DotNetIdentity.Services.SettingsService.SettingsService.SettingsService (
    ApplicationDbContext unitOfWork ) [inline]
```

class constructor

Parameters

<i>unitOfWork</i>	
-------------------	--

Definition at line 37 of file [SettingsService.cs](#).

```
00038 {
00039     // ARGUMENT CHECKING SKIPPED FOR BREVITY
00040     _unitOfWork = unitOfWork;
00041     // 3
00042     _globalSettings = new Lazy<GlobalSettings>(CreateSettings<GlobalSettings>);
00043     _mailSettings = new Lazy<MailSettings>(CreateSettings<MailSettings>);
00044 }
```

### 6.33.3 Member Function Documentation

#### 6.33.3.1 Save()

```
void DotNetIdentity.Services.SettingsService.SettingsService.Save ( ) [inline]
```

sve method

Implements [DotNetIdentity.Services.SettingsService.ISettingsService](#).

Definition at line 49 of file [SettingsService.cs](#).

```
00050     {
00051         // only save changes to settings that have been loaded
00052         if (_globalSettings.IsValueCreated)
00053             _globalSettings.Value.Save(_unitOfWork);
00054
00055         if (_mailSettings.IsValueCreated)
00056             _mailSettings.Value.Save(_unitOfWork);
00057
00058         _unitOfWork.SaveChanges();
00059     }
```

## 6.33.4 Property Documentation

### 6.33.4.1 Global

[GlobalSettings](#) [DotNetIdentity.Services.SettingsService.SettingsService.Global](#) [get]

global settings property

Implements [DotNetIdentity.Services.SettingsService.ISettingsService](#).

Definition at line 18 of file [SettingsService.cs](#).

```
00018 { get { return _globalSettings.Value; } }
```

### 6.33.4.2 Mail

[MailSettings](#) [DotNetIdentity.Services.SettingsService.SettingsService.Mail](#) [get]

mail settings property

Implements [DotNetIdentity.Services.SettingsService.ISettingsService](#).

Definition at line 27 of file [SettingsService.cs](#).

```
00027 { get { return _mailSettings.Value; } }
```

## 6.34 DotNetIdentity.Models.ViewModels.SignInViewModel Class Reference

model for Login view

## Properties

- string [UserName](#) = string.Empty [get, set]  
*the username*
- string [Password](#) = default! [get, set]  
*the password*
- bool [RememberMe](#) = true [get, set]  
*the set cookie persisten option*

### 6.34.1 Detailed Description

model for Login view

Definition at line 6 of file [SignInViewModel.cs](#).

### 6.34.2 Property Documentation

#### 6.34.2.1 Password

```
string DotNetIdentity.Models.ViewModels.SignInViewModel.Password = default! [get], [set]
```

the password

string

Definition at line 17 of file [SignInViewModel.cs](#).

```
00017 { get; set; } = default!;
```

Referenced by [DotNetIdentity.Controllers.UserController.Login\(\)](#).

#### 6.34.2.2 RememberMe

```
bool DotNetIdentity.Models.ViewModels.SignInViewModel.RememberMe = true [get], [set]
```

the set cookie persisten option

bool

Definition at line 22 of file [SignInViewModel.cs](#).

```
00022 { get; set; } = true;
```

Referenced by [DotNetIdentity.Controllers.UserController.Login\(\)](#).

### 6.34.2.3 UserName

```
string DotNetIdentity.Models.ViewModels.SignInViewModel.UserName = string.Empty [get], [set]
```

the username

string

Definition at line 12 of file [SignInViewModel.cs](#).

```
00012 {get; set;} = string.Empty;
```

Referenced by [DotNetIdentity.Controllers.UserController.Login\(\)](#).

## 6.35 DotNetIdentity.Models.ViewModels.SignUpViewModel Class Reference

model for Register view

### Properties

- string [UserName](#) = default! [get, set]  
*the username*
- string [Email](#) = default! [get, set]  
*the email*
- [Gender](#) [Gender](#) = [Gender.Unknown](#) [get, set]  
*the gender*
- DateTime [BirthDay](#) [get, set]  
*the birthday*
- [Department](#) [Department](#) = [Department.Software](#) [get, set]  
*the department*
- string [Password](#) = default! [get, set]  
*the password*
- string [FirstName](#) = string.Empty [get, set]  
*the firstname*
- string [LastName](#) = string.Empty [get, set]  
*the lastname*

### 6.35.1 Detailed Description

model for Register view

Definition at line 6 of file [SignUpViewModel.cs](#).

### 6.35.2 Property Documentation



### 6.35.2.1 BirthDay

`DateTime DotNetIdentity.Models.ViewModels.SignUpViewModel.BirthDay [get], [set]`

the birthday

DateTime

Definition at line 27 of file [SignUpViewModel.cs](#).

```
00027 { get; set; }
```

Referenced by [DotNetIdentity.Controllers.UserController.Register\(\)](#).

### 6.35.2.2 Department

`Department DotNetIdentity.Models.ViewModels.SignUpViewModel.Department = Department.Software [get], [set]`

the department

Department

Definition at line 32 of file [SignUpViewModel.cs](#).

```
00032 { get; set; } = Department.Software;
```

Referenced by [DotNetIdentity.Controllers.UserController.Register\(\)](#).

### 6.35.2.3 Email

`string DotNetIdentity.Models.ViewModels.SignUpViewModel.Email = default! [get], [set]`

the email

string

Definition at line 17 of file [SignUpViewModel.cs](#).

```
00017 { get; set; } = default!;
```

Referenced by [DotNetIdentity.Controllers.UserController.Register\(\)](#).

### 6.35.2.4 FirstName

`string DotNetIdentity.Models.ViewModels.SignUpViewModel.FirstName = string.Empty [get], [set]`

the firstname

string

Definition at line 42 of file [SignUpViewModel.cs](#).

```
00042 {get; set;} = string.Empty;
```

Referenced by [DotNetIdentity.Controllers.UserController.Register\(\)](#).

### 6.35.2.5 Gender

`Gender` `DotNetIdentity.Models.ViewModels.SignUpViewModel.Gender` = `Gender.Unknown` [get], [set]

the gender

Gender

Definition at line 22 of file [SignUpViewModel.cs](#).

```
00022 { get; set; } = Gender.Unknown;
```

Referenced by [DotNetIdentity.Controllers.UserController.Register\(\)](#).

### 6.35.2.6 LastName

`string` `DotNetIdentity.Models.ViewModels.SignUpViewModel.LastName` = `string.Empty` [get], [set]

the lastname

string

Definition at line 47 of file [SignUpViewModel.cs](#).

```
00047 { get; set; } = string.Empty;
```

Referenced by [DotNetIdentity.Controllers.UserController.Register\(\)](#).

### 6.35.2.7 Password

`string` `DotNetIdentity.Models.ViewModels.SignUpViewModel.Password` = `default!` [get], [set]

the password

string

Definition at line 37 of file [SignUpViewModel.cs](#).

```
00037 { get; set; } = default!;
```

Referenced by [DotNetIdentity.Controllers.UserController.Register\(\)](#).

### 6.35.2.8 UserName

`string` `DotNetIdentity.Models.ViewModels.SignUpViewModel.UserName` = `default!` [get], [set]

the username

string

Definition at line 12 of file [SignUpViewModel.cs](#).

```
00012 { get; set; } = default!;
```

Referenced by [DotNetIdentity.Controllers.UserController.Register\(\)](#).

## 6.36 DotNetIdentity.Models.ViewModels.StatusCodesModel Class Reference

model for Error Statuscodes

### Properties

- int [StatusCode](#) [get, set]  
*the status code*
- string? [ErrorMessage](#) [get, set]  
*the error message*
- string? [RouteOfException](#) [get, set]  
*the route that thowed the exception*

### 6.36.1 Detailed Description

model for Error Statuscodes

Definition at line 5 of file [StatusCodesModel.cs](#).

### 6.36.2 Property Documentation

#### 6.36.2.1 ErrorMessage

```
string? DotNetIdentity.Models.ViewModels.StatusCodesModel.ErrorMessage [get], [set]
```

the error message

string

Definition at line 15 of file [StatusCodesModel.cs](#).

```
00015 {get; set;}
```

#### 6.36.2.2 RouteOfException

```
string? DotNetIdentity.Models.ViewModels.StatusCodesModel.RouteOfException [get], [set]
```

the route that thowed the exception

string

Definition at line 20 of file [StatusCodesModel.cs](#).

```
00020 {get; set;}
```

### 6.36.2.3 StatusCode

```
int DotNetIdentity.Models.ViewModels.StatusCodesModel.StatusCode [get], [set]
```

the status code

int

Definition at line 10 of file [StatusCodesModel.cs](#).

```
00010 {get; set;}
```

## 6.37 DotNetIdentity.IdentitySettings.TwoFactorAuthenticationService Class Reference

class providing 2fa auth services

### Public Member Functions

- [TwoFactorAuthenticationService](#) (IConfiguration cnf, UrlEncoder urlEncoder, ISettingsService sett)  
*class constructor*
- string [GenerateQrCodeUri](#) (string email, string authenticatorKey)  
*method to generate a barcode for 2fa configuration*

### 6.37.1 Detailed Description

class providing 2fa auth services

Definition at line 8 of file [TwoFactorAuthenticationService.cs](#).

### 6.37.2 Constructor & Destructor Documentation

#### 6.37.2.1 TwoFactorAuthenticationService()

```
DotNetIdentity.IdentitySettings.TwoFactorAuthenticationService.TwoFactorAuthenticationService
(
    IConfiguration cnf,
    UrlEncoder urlEncoder,
    ISettingsService sett ) [inline]
```

class constructor

Parameters

<i>cnf</i>	IConfiguration
<i>urlEncoder</i>	UrlEncoder
<i>sett</i>	ISettingsService

Definition at line 29 of file [TwoFactorAuthenticationService.cs](#).

```
00030     {
00031         _urlEncoder = urlEncoder;
00032         _configuration = cnf;
00033         _sett = sett;
00034     }
```

## 6.37.3 Member Function Documentation

### 6.37.3.1 GenerateQrCodeUri()

```
string DotNetIdentity.IdentitySettings.TwoFactorAuthenticationService.GenerateQrCodeUri (
    string email,
    string authenticatorKey ) [inline]
```

method to generate a barcode for 2fa configuration

#### Parameters

<i>email</i>	string
<i>authenticatorKey</i>	string

#### Returns

string

Definition at line 42 of file [TwoFactorAuthenticationService.cs](#).

```
00043     {
00044         var encodedUrl = _urlEncoder.Encode(_sett.Global.ApplicationName!);
00045         var encodedEmail = _urlEncoder.Encode(email);
00046
00047         return
00048         $"otpauth://totp/{encodedUrl}:{encodedEmail}?secret={authenticatorKey}&issuer={encodedUrl}&digits=6";
00048     }
```

## 6.38 DotNetIdentity.Models.ViewModels.TwoFactorAuthenticatorView Model Class Reference ↩

model for TwoFactorAuthenticator view

### Properties

- string [SharedKey](#) = default! [get, set]  
*the shared key*
- string [AuthenticationUri](#) = default! [get, set]  
*the auth uri*
- string [VerificationCode](#) = default! [get, set]  
*the otp code*

### 6.38.1 Detailed Description

model for TwoFactorAuthenticator view

Definition at line 6 of file [TwoFactorAuthenticatorViewModel.cs](#).

### 6.38.2 Property Documentation

#### 6.38.2.1 AuthenticationUri

```
string DotNetIdentity.Models.ViewModels.TwoFactorAuthenticatorViewModel.AuthenticationUri =  
default! [get], [set]
```

the auth uri

string

Definition at line 17 of file [TwoFactorAuthenticatorViewModel.cs](#).

```
00017 { get; set; } = default!;
```

#### 6.38.2.2 SharedKey

```
string DotNetIdentity.Models.ViewModels.TwoFactorAuthenticatorViewModel.SharedKey = default!  
[get], [set]
```

the shared key

string

Definition at line 12 of file [TwoFactorAuthenticatorViewModel.cs](#).

```
00012 { get; set; } = default!;
```

#### 6.38.2.3 VerificationCode

```
string DotNetIdentity.Models.ViewModels.TwoFactorAuthenticatorViewModel.VerificationCode =  
default! [get], [set]
```

the otp code

string

Definition at line 22 of file [TwoFactorAuthenticatorViewModel.cs](#).

```
00022 { get; set; } = default!;
```

Referenced by [DotNetIdentity.Controllers.UserController.EnforceTwoFactorAuthenticator\(\)](#), and [DotNetIdentity.Controllers.UserContro](#)

## 6.39 DotNetIdentity.Models.ViewModels.TwoFactorLoginVieWModel Class Reference

the model for TwoFactorLogin view

### Properties

- string [VerificationCode](#) = default! [get, set]  
*the otp code*
- bool [IsRecoveryCode](#) [get, set]  
*bool if otp code is a recovery code*
- [TwoFactorType](#) [TwoFactorType](#) [get, set]  
*the type of 2fa*

### 6.39.1 Detailed Description

the model for TwoFactorLogin view

Definition at line 6 of file [TwoFactorLoginVieWModel.cs](#).

### 6.39.2 Property Documentation

#### 6.39.2.1 IsRecoveryCode

```
bool DotNetIdentity.Models.ViewModels.TwoFactorLoginVieWModel.IsRecoveryCode [get], [set]
```

bool if otp code is a recovery code

bool

Definition at line 17 of file [TwoFactorLoginVieWModel.cs](#).

```
00017 { get; set; }
```

#### 6.39.2.2 TwoFactorType

```
TwoFactorType DotNetIdentity.Models.ViewModels.TwoFactorLoginVieWModel.TwoFactorType [get],  
[set]
```

the type of 2fa

TwoFactorType

Definition at line 22 of file [TwoFactorLoginVieWModel.cs](#).

```
00022 { get; set; }
```

Referenced by [DotNetIdentity.Controllers.UserController.TwoFactorLogin\(\)](#).

### 6.39.2.3 VerificationCode

```
string DotNetIdentity.Models.ViewModels.TwoFactorLoginViewWModel.VerificationCode = default!  
[get], [set]
```

the otp code

string

Definition at line 12 of file [TwoFactorLoginViewWModel.cs](#).

```
00012 { get; set; } = default!;
```

Referenced by [DotNetIdentity.Controllers.UserController.TwoFactorLogin\(\)](#).

## 6.40 DotNetIdentity.Models.ViewModels.TwoFactorTypeViewModel Class Reference

model for TwoFactorType view

### Properties

- [TwoFactorType](#) [TwoFactorType](#) [get, set]  
*the 2fa type*

### 6.40.1 Detailed Description

model for TwoFactorType view

Definition at line 6 of file [TwoFactorTypeViewModel.cs](#).

### 6.40.2 Property Documentation

#### 6.40.2.1 TwoFactorType

```
TwoFactorType DotNetIdentity.Models.ViewModels.TwoFactorTypeViewModel.TwoFactorType [get],  
[set]
```

the 2fa type

TwoFactorType

Definition at line 12 of file [TwoFactorTypeViewModel.cs](#).

```
00012 { get; set; }
```

Referenced by [DotNetIdentity.Controllers.UserController.TwoFactorType\(\)](#).



## 6.41 DotNetIdentity.Models.ViewModels.UpdateProfileViewModel Class Reference

model for UpdateProfile view

### Properties

- string [UserName](#) = default! [get, set]  
*the username*
- string [Email](#) = default! [get, set]  
*the email*
- string [PhoneNumber](#) = default! [get, set]  
*the phone number*
- [Gender](#) [Gender](#) [get, set]  
*the gender*
- DateTime [BirthDay](#) [get, set]  
*the birthday*
- IFormFile? [ProfilePicture](#) [get, set]  
*the profile picture*

### 6.41.1 Detailed Description

model for UpdateProfile view

Definition at line 6 of file [UpdateProfileViewModel.cs](#).

### 6.41.2 Property Documentation

#### 6.41.2.1 BirthDay

```
DateTime DotNetIdentity.Models.ViewModels.UpdateProfileViewModel.BirthDay [get], [set]
```

the birthday

DateTime

Definition at line 32 of file [UpdateProfileViewModel.cs](#).

```
00032 { get; set; }
```

Referenced by [DotNetIdentity.Controllers.UserController.Profile\(\)](#).

### 6.41.2.2 Email

```
string DotNetIdentity.Models.ViewModels.UpdateProfileViewModel.Email = default! [get], [set]
```

the email

string

Definition at line 17 of file [UpdateProfileViewModel.cs](#).

```
00017 { get; set; } = default!;
```

Referenced by [DotNetIdentity.Controllers.UserController.Profile\(\)](#).

### 6.41.2.3 Gender

```
Gender DotNetIdentity.Models.ViewModels.UpdateProfileViewModel.Gender [get], [set]
```

the gender

Gender

Definition at line 27 of file [UpdateProfileViewModel.cs](#).

```
00027 { get; set; }
```

Referenced by [DotNetIdentity.Controllers.UserController.Profile\(\)](#).

### 6.41.2.4 PhoneNumber

```
string DotNetIdentity.Models.ViewModels.UpdateProfileViewModel.PhoneNumber = default! [get],  
[set]
```

the phone number

string

Definition at line 22 of file [UpdateProfileViewModel.cs](#).

```
00022 { get; set; } = default!;
```

Referenced by [DotNetIdentity.Controllers.UserController.Profile\(\)](#).

### 6.41.2.5 ProfilePicture

```
IFormFile? DotNetIdentity.Models.ViewModels.UpdateProfileViewModel.ProfilePicture [get],  
[set]
```

the profile picture

IFormFile

Definition at line 37 of file [UpdateProfileViewModel.cs](#).

```
00037 { get; set; }
```

Referenced by [DotNetIdentity.Controllers.UserController.Profile\(\)](#).

### 6.41.2.6 UserName

```
string DotNetIdentity.Models.ViewModels.UpdateProfileViewModel.UserName = default! [get],  
[set]
```

the username

string

Definition at line 12 of file [UpdateProfileViewModel.cs](#).

```
00012 { get; set; } = default!;
```

Referenced by [DotNetIdentity.Controllers.UserController.Profile\(\)](#).

## 6.42 DotNetIdentity.Models.ViewModels.UpsertRoleViewModel Class Reference

model for the UpsertRole view

### Properties

- string? [Id](#) [get, set]  
*the role id*
- string [Name](#) = default! [get, set]  
*the role name*

### 6.42.1 Detailed Description

model for the UpsertRole view

Definition at line 6 of file [UpsertRoleViewModel.cs](#).

### 6.42.2 Property Documentation

#### 6.42.2.1 Id

```
string? DotNetIdentity.Models.ViewModels.UpsertRoleViewModel.Id [get], [set]
```

the role id

string

Definition at line 12 of file [UpsertRoleViewModel.cs](#).

```
00012 { get; set; }
```

Referenced by [DotNetIdentity.Controllers.AdminController.UpsertRole\(\)](#).

### 6.42.2.2 Name

```
string DotNetIdentity.Models.ViewModels.UpsertRoleViewModel.Name = default! [get], [set]
```

the role name

string

Definition at line 17 of file [UpsertRoleViewModel.cs](#).

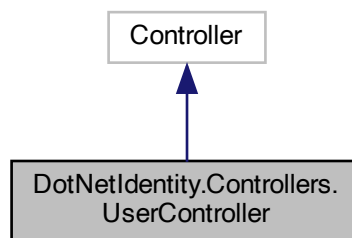
```
00017 { get; set; } = default!;
```

Referenced by [DotNetIdentity.Controllers.AdminController.UpsertRole\(\)](#).

## 6.43 DotNetIdentity.Controllers.UserController Class Reference

Controller Class for User views

Inheritance diagram for DotNetIdentity.Controllers.UserController:



### Public Member Functions

- [UserController](#) (IStringLocalizer< [UserController](#) > localizer, ILogger< [UserController](#) > log, IWebHostEnvironment webhostEnviroment, UserManager< [AppUser](#) > userManager, SignInManager< [AppUser](#) > signInManager, [TwoFactorAuthenticationService](#) twoFactorAuthService, [EmailHelper](#) emailHelper)  
*Controller constructor*
- IActionResult [Register](#) ()  
*Controller action for Register view*
- IActionResult [debugUserInfo](#) ()  
*Controller action for DebugUserInfo view*
- IActionResult [ping](#) ()  
*Controller action for Ping view*
- async Task< IActionResult > [Register](#) ([SignUpViewModel](#) viewModel)  
*Controller POST method for Register view*
- async Task< IActionResult > [Login](#) (string? returnUrl)  
*Controller action for Login view*
- async Task< IActionResult > [Login](#) ([SignInViewModel](#) viewModel)

- Controller POST method for Login view*
- async Task< IActionResult > [TwoFactorLogin](#) (string? returnUrl)
- Controller action for view TwoFactorLogin*
- async Task< IActionResult > [TwoFactorLogin](#) ([TwoFactorLoginViewWModel](#) viewWModel)
- controller POST method for TwoFactorLogin view*
- async Task [Logout](#) ()
- controller action for Logout*
- async Task< IActionResult > [Profile](#) ()
- controller action for Profile view*
- async Task< IActionResult > [Profile](#) ([UpdateProfileViewModel](#) viewModel)
- controller POST method for Profile view*
- IActionResult [ChangePassword](#) ()
- controller action for ChangePassword view*
- async Task< IActionResult > [ChangePassword](#) ([ChangePasswordViewModel](#) viewModel)
- controller POST method for ChangePassword view*
- IActionResult [ForgotPassword](#) ()
- controller action for ForgotPassword view*
- async Task< IActionResult > [ForgotPassword](#) ([ForgotPasswordViewModel](#) viewModel)
- controller POST method for ForgotPassword view*
- IActionResult [ResetPassword](#) (string userId, string token)
- controller action for ResetPassword view*
- async Task< IActionResult > [ResetPassword](#) ([ResetPasswordViewModel](#) viewModel)
- controller POST method for ResetPassword view*
- async Task< IActionResult > [ConfirmEmail](#) (string userId, string token)
- controller action for ConfirmEmail view*
- async Task< IActionResult > [TwoFactorType](#) ()
- controller action for view TwiFactorType*
- async Task< IActionResult > [TwoFactorType](#) ([TwoFactorTypeViewModel](#) viewModel)
- controller POST method for view TwoFactorType*
- async Task< IActionResult > [EnforceTwoFactorAuthenticator](#) ()
- controller action for view EnforceTwoFactorAuthenticator*
- async Task< IActionResult > [EnforceTwoFactorAuthenticator](#) ([TwoFactorAuthenticatorViewModel](#) view↔Model)
- controller POST method for view EnforceTwoFactorAuthenticator*
- async Task< IActionResult > [EnforceTwoFactorResult](#) ()
- controller action for view EnforceTwoFactorResult*
- async Task< IActionResult > [TwoFactorAuthenticator](#) ()
- controller action for view TwoFactorAuthenticator*
- async Task< IActionResult > [TwoFactorAuthenticator](#) ([TwoFactorAuthenticatorViewModel](#) viewModel)
- controller POST method for view TwoFactorAuthenticator*

### 6.43.1 Detailed Description

Controller Class for User views

**Todo** add translation

Definition at line 20 of file [UserController.cs](#).

## 6.43.2 Constructor & Destructor Documentation

### 6.43.2.1 UserController()

```

DotNetIdentity.Controllers.UserController.UserController (
    IStringLocalizer< UserController > localizer,
    ILogger< UserController > log,
    IWebHostEnvironment webhostEnviroment,
    UserManager< AppUser > userManager,
    SignInManager< AppUser > signInManager,
    TwoFactorAuthenticationService twoFactorAuthService,
    EmailHelper emailHelper ) [inline]

```

Controller constructor

#### Parameters

<i>localizer</i>	type <a href="#">IStringLocalizer</a>
<i>log</i>	type <a href="#">ILogger</a>
<i>webhostEnviroment</i>	type <a href="#">IWebHostEnviroment</a>
<i>userManager</i>	type <a href="#">UserManager</a>
<i>signInManager</i>	type <a href="#">SignInManager</a>
<i>twoFactorAuthService</i>	type <a href="#">TwoFactorAuthService</a>
<i>emailHelper</i>	type <a href="#">EmailHelper</a>

Definition at line 61 of file [UserController.cs](#).

```

00062     {
00063         _userManager = userManager;
00064         _signInManager = signInManager;
00065         _emailHelper = emailHelper;
00066         _twoFactorAuthService = twoFactorAuthService;
00067         _hostEnvironment = webhostEnviroment;
00068         _logger = log;
00069         _localizer = localizer;
00070     }

```

## 6.43.3 Member Function Documentation

### 6.43.3.1 ChangePassword() [1/2]

```

ActionResult DotNetIdentity.Controllers.UserController.ChangePassword ( )

```

controller action for ChangePassword view

#### Returns

view ChangePassword

### 6.43.3.2 ChangePassword() [2/2]

```
async Task< IActionResult > DotNetIdentity.Controllers.UserController.ChangePassword (
    ChangePasswordViewModel viewModel ) [inline]
```

controller POST method for ChangePassword view

#### Parameters

<i>viewModel</i>	ChangePasswordViewModel
------------------	-------------------------

#### Returns

redirect to action Index

Definition at line 410 of file [UserController.cs](#).

```
00411     {
00412         if (ModelState.IsValid)
00413         {
00414             var me = await _userManager.FindByNameAsync(User.Identity?.Name);
00415
00416             var passwordValid = await _userManager.CheckPasswordAsync(me, viewModel.Password);
00417             if (passwordValid)
00418             {
00419                 var result = await _userManager.ChangePasswordAsync(me, viewModel.Password,
00420                     viewModel.NewPassword);
00421                 if (result.Succeeded)
00422                 {
00423                     await _userManager.UpdateSecurityStampAsync(me);
00424
00425                     await _signInManager.SignOutAsync();
00426                     await _signInManager.SignInAsync(me, true);
00427                     _logger.LogInformation("AUDIT: " + me.UserName + " successfully changed own
00428                     password!");
00429                     return RedirectToAction("Index", "Admin");
00430                 }
00431                 _logger.LogWarning("AUDIT: " + me.UserName + " tried to change password. Error: " +
00432                     result.Errors.FirstOrDefault()?.ToString());
00433                 result.Errors.ToList().ForEach(f => ModelState.AddModelError(string.Empty,
00434                     f.Description));
00435             }
00436             else
00437             {
00438                 _logger.LogWarning("AUDIT: " + me.UserName + " tried to change password. But
00439                     provided invalid current pasword!");
00440                 ModelState.AddModelError(string.Empty, _localizer["8"]);
00441             }
00442         }
00443     }
00444     return View();
00445 }
```

References [DotNetIdentity.Models.ViewModels.ChangePasswordViewModel.NewPassword](#), and [DotNetIdentity.Models.ViewModels.C](#)

### 6.43.3.3 ConfirmEmail()

```
async Task< IActionResult > DotNetIdentity.Controllers.UserController.ConfirmEmail (
    string userId,
    string token ) [inline]
```

controller action for ConfirmEmail view

**Parameters**

<i>userId</i>	the user id
<i>token</i>	the confirmation token

**Returns**

redirect to action Index

Definition at line 546 of file [UserController.cs](#).

```

00547     {
00548         var user = await _userManager.FindByIdAsync(userId);
00549         if (user != null)
00550         {
00551             var result = await _userManager.ConfirmEmailAsync(user, token);
00552             if (result.Succeeded)
00553             {
00554                 _logger.LogInformation("AUDIT: " + user.UserName + " successfully confirmed email
00555                 address!");
00556                 return RedirectToAction("Login");
00557             }
00558             return RedirectToAction("Index", "Home");
00559         }

```

**6.43.3.4 debugUserInfo()**

`IActionResult DotNetIdentity.Controllers.UserController.debugUserInfo ( )`

Controller action for DebugUserInfo view

**Returns**

view DebugUserInfo

**6.43.3.5 EnforceTwoFactorAuthenticator() [1/2]**

`async Task< IActionResult > DotNetIdentity.Controllers.UserController.EnforceTwoFactorAuthenticator ( ) [inline]`

controller action for view EnforceTwoFactorAuthenticator

**Returns**

view EnforceTwoFactorAuthenticator

Definition at line 603 of file [UserController.cs](#).

```

00604     {
00605         var user = await _userManager.FindByNameAsync(User.Identity?.Name);
00606         user.TwoFactorType = Models.TwoFactorType.Authenticator;
00607         await _userManager.UpdateAsync(user);
00608
00609         var authenticatorKey = await _userManager.GetAuthenticatorKeyAsync(user);
00610         if (authenticatorKey == null)
00611         {
00612             await _userManager.ResetAuthenticatorKeyAsync(user);
00613             authenticatorKey = await _userManager.GetAuthenticatorKeyAsync(user);
00614         }
00615
00616         return View(new TwoFactorAuthenticatorViewModel
00617         {
00618             SharedKey = authenticatorKey,
00619             AuthenticationUri = _twoFactorAuthService.GenerateQrCodeUri(user.Email,
00620             authenticatorKey)
00621         });
00621     }

```



**6.43.3.6 EnforceTwoFactorAuthenticator()** [2/2]

```
async Task< IActionResult > DotNetIdentity.Controllers.UserController.EnforceTwoFactorAuthenticator
(
    TwoFactorAuthenticatorViewModel viewModel ) [inline]
```

controller POST method for view EnforceTwoFactorAuthenticator

**Parameters**

<i>viewModel</i>	type TwoFactorAuthenticatorViewModel
------------------	--------------------------------------

**Returns**

redirect to action EnforceTwoFactorResult

Definition at line 629 of file [UserController.cs](#).

```
00630     {
00631         var user = await _userManager.FindByNameAsync(User.Identity?.Name);
00632
00633         var verificationCode = viewModel.VerificationCode.Replace(" ", string.Empty).Replace("-",
string.Empty);
00634
00635         var isValidToken = await _userManager.VerifyTwoFactorTokenAsync(user,
_userManager.Options.Tokens.AuthenticatorTokenProvider, verificationCode);
00636         if (isValidToken)
00637         {
00638             if (user.IsMfaForce && user.TwoFactorEnabled==false) {
00639                 await _userManager.SetTwoFactorEnabledAsync(user, true);
00640                 user.IsMfaForce = false;
00641                 await _userManager.UpdateAsync(user);
00642             }
00643
00644             TempData["RecoveryCodes"] = await
_userManager.GenerateNewTwoFactorRecoveryCodesAsync(user, 5);
00645         }
00646         _logger.LogInformation("AUDIT: " + user.UserName + " successfully configured 2fa after
enforcement. Redirecting to action: EnforceTwoFactorResult");
00647         return RedirectToAction("EnforceTwoFactorResult", "User");
00648     }
```

References [DotNetIdentity.Models.ViewModels.TwoFactorAuthenticatorViewModel.VerificationCode](#).

**6.43.3.7 EnforceTwoFactorResult()**

```
async Task< IActionResult > DotNetIdentity.Controllers.UserController.EnforceTwoFactorResult (
) [inline]
```

controller action for view EnforceTwoFactorResult

**Returns**

view EnforceTwoFactorResult

Definition at line 654 of file [UserController.cs](#).

```
00654     {
00655         await _signInManager.SignOutAsync();
00656         return View();
00657     }
```

**6.43.3.8 ForgotPassword()** [1/2]

`ActionResult DotNetIdentity.Controllers.UserController.ForgotPassword ( )`

controller action for ForgotPassword view

**Returns**

view ForgotPassword

**6.43.3.9 ForgotPassword()** [2/2]

`async Task< ActionResult > DotNetIdentity.Controllers.UserController.ForgotPassword (   
ForgotPasswordViewModel viewModel ) [inline]`

controller POST method for ForgotPassword view

**Parameters**

<i>viewModel</i>	ForgotPasswordViewModel
------------------	-------------------------

**Returns**

redirect to action Index

Definition at line 455 of file [UserController.cs](#).

```

00456     {
00457         if (ModelState.IsValid)
00458         {
00459             var user = await _userManager.FindByEmailAsync(viewModel.Email);
00460             if (user != null)
00461             {
00462                 var passwordResetToken = await _userManager.GeneratePasswordResetTokenAsync(user);
00463
00464                 var passwordLink = Url.Action("ResetPassword", "User", new
00465                 {
00466                     userId = user.Id,
00467                     token = passwordResetToken
00468                 }, HttpContext.Request.Scheme);
00469
00470                 await _emailHelper.SendAsync(new()
00471                 {
00472                     Subject = "Reset password",
00473                     Body = $"Please <a href='{passwordLink}'>click</a> to reset your password.",
00474                     To = user.Email
00475                 });
00476                 _logger.LogWarning("AUDIT: " + user.UserName + " requested a password reset!");
00477                 return RedirectToAction("Index", "Home");
00478             }
00479             else
00480             {
00481                 ModelState.AddModelError(string.Empty, _localizer["9"]);
00482             }
00483         }
00484         return View(viewModel);
00485     }

```

References [DotNetIdentity.Models.ViewModels.ForgotPasswordViewModel.Email](#).

## 6.43.3.10 Login() [1/2]

```
async Task< IActionResult > DotNetIdentity.Controllers.UserController.Login (
    SignInViewModel viewModel ) [inline]
```

Controller POST method for Login view

**Todo** Implement Ldap Login

## Returns

Redirect to action

Definition at line 162 of file [UserController.cs](#).

```
00163     {
00164         if (ModelState.IsValid)
00165         {
00166             var user = await _userManager.FindByNameAsync(viewModel.UserName);
00167             if (user != null)
00168             {
00169                 await _signInManager.SignOutAsync();
00170
00171                 var result = await _signInManager.PasswordSignInAsync(user, viewModel.Password,
viewModel.RememberMe, true);
00172                 if (result.Succeeded && user.IsEnabled==false) {
00173                     await _signInManager.SignOutAsync();
00174                     _logger.LogWarning("AUDIT: " + user.UserName + " logged in successfully. But
account is disabled!");
00175                     ModelState.AddModelError(string.Empty, _localizer["1"]);
00176                 }
00177                 else if (result.Succeeded && user.IsMfaForce==true) {
00178                     await _userManager.ResetAccessFailedCountAsync(user);
00179                     await _userManager.SetLockoutEndDateAsync(user, null);
00180                     _logger.LogInformation("AUDIT: " + user.UserName + " logged in successfully.
But MFA is enforced. Redirecting to action: EnforceTwoFactorAuthenticator");
00181                     return RedirectToAction("EnforceTwoFactorAuthenticator");
00182                 }
00183                 else if (result.Succeeded && user.IsMfaForce==false)
00184                 {
00185                     await _userManager.ResetAccessFailedCountAsync(user);
00186                     await _userManager.SetLockoutEndDateAsync(user, null);
00187                     _logger.LogInformation("AUDIT: " + user.UserName + " logged in
successfully.");
00188                     var returnUrl = TempData["ReturnUrl"];
00189                     if (returnUrl != null)
00190                     {
00191                         return Redirect(returnUrl.ToString() ?? "/");
00192                     }
00193                     return RedirectToAction("Index", "Admin");
00194                 }
00195                 else if (result.RequiresTwoFactor)
00196                 {
00197                     _logger.LogInformation("AUDIT: " + user.UserName + " logged in successfully.
But MFA is enabled. Redirecting to action: TwoFactorLogin");
00198                     return RedirectToAction("TwoFactorLogin", new { ReturnUrl =
TempData["ReturnUrl"] });
00199                 }
00200                 else if (result.IsLockedOut)
00201                 {
00202                     var lockoutEndUtc = await _userManager.GetLockoutEndDateAsync(user);
00203                     var timeLeft = lockoutEndUtc.Value - DateTime.UtcNow;
00204                     _logger.LogWarning("AUDIT: " + user.UserName + " logged in successfully. But
Account is locked out for at least another " + timeLeft.Minutes + "minutes!");
00205                     ModelState.AddModelError(string.Empty, _localizer["2",
timeLeft.Minutes.ToString()]);
00206                 }
00207                 else if (result.IsNotAllowed)
00208                 {
00209                     _logger.LogWarning("AUDIT: " + user.UserName + " logged in successfully. But
email is not confirmed!");
00210                     ModelState.AddModelError(string.Empty, _localizer["3"]);
00211                 }
00212                 else
00213                 {
00214                     _logger.LogWarning("AUDIT: " + user.UserName + " login in failed. Invalid
username or password!");
00215                     ModelState.AddModelError(string.Empty, _localizer["4"]);
```

```

00216         }
00217     }
00218     else
00219     {
00220         _logger.LogWarning("AUDIT: " + "Login in failed. Invalid username or password!");
00221         ModelState.AddModelError(string.Empty, _localizer["4"]);
00222     }
00223 }
00224 return View(viewModel);
00225 }

```

References [DotNetIdentity.Models.ViewModels.SignInViewModel.Password](#), [DotNetIdentity.Models.ViewModels.SignInViewModel.Re](#) and [DotNetIdentity.Models.ViewModels.SignInViewModel.UserName](#).

### 6.43.3.11 Login() [2/2]

```

async Task< IActionResult > DotNetIdentity.Controllers.UserController.Login (
    string? returnUrl ) [inline]

```

Controller action for Login view

#### Parameters

<i>returnUrl</i>	
------------------	--

#### Returns

view Login

Definition at line 146 of file [UserController.cs](#).

```

00147     {
00148         await _signInManager.SignOutAsync();
00149         if (returnUrl != null)
00150         {
00151             TempData["ReturnUrl"] = returnUrl;
00152         }
00153         return View();
00154     }

```

### 6.43.3.12 Logout()

```

async Task DotNetIdentity.Controllers.UserController.Logout ( )

```

controller action for Logout

#### Returns

### 6.43.3.13 ping()

```
IActionResult DotNetIdentity.Controllers.UserController.ping ( )
```

Controller action for Ping view

#### Returns

view Ping

### 6.43.3.14 Profile() [1/2]

```
async Task< IActionResult > DotNetIdentity.Controllers.UserController.Profile ( ) [inline]
```

controller action for Profile view

#### Returns

view Profile

Definition at line 305 of file [UserController.cs](#).

```
00306     {
00307         var me = await _userManager.FindByNameAsync(User.Identity?.Name);
00308         if (me == null)
00309         {
00310             await _signInManager.SignOutAsync();
00311             return RedirectToAction("Index", "Home");
00312         }
00313         var mod = new UpdateProfileViewModel();
00314         mod.Email = me.Email;
00315         mod.Birthday = me.Birthday;
00316         mod.UserName = me.UserName;
00317         mod.PhoneNumber = me.PhoneNumber;
00318         mod.Gender = me.Gender;
00319         //return View(me.Adapt<UpdateProfileViewModel>());
00320         return View(mod);
00321     }
```

### 6.43.3.15 Profile() [2/2]

```
async Task< IActionResult > DotNetIdentity.Controllers.UserController.Profile (
    UpdateProfileViewModel viewModel ) [inline]
```

controller POST method for Profile view

#### Parameters

<i>viewModel</i>	type UpdateProfileViewModel
------------------	-----------------------------

#### Returns

redirect to action Index

Definition at line 330 of file [UserController.cs](#).

```

00331     {
00332         if (ModelState.IsValid)
00333         {
00334             var me = await _userManager.FindByNameAsync(User.Identity?.Name);
00335             if (me != null)
00336             {
00337                 if (me.PhoneNumber != viewModel.PhoneNumber && _userManager.Users.Any(a =>
00338 a.PhoneNumber == viewModel.PhoneNumber))
00339             {
00340                 ModelState.AddModelError(string.Empty, _localizer["6"]);
00341             }
00342             else
00343             {
00344                 if (viewModel.ProfilePicture != null) {
00345                     using (var memoryStream = new MemoryStream())
00346                     {
00347                         await viewModel.ProfilePicture.CopyToAsync(memoryStream);
00348
00349                         // Upload the file if less than 2 MB
00350                         if (memoryStream.Length < 2097152)
00351                         {
00352                             var format = "image/png";
00353                             var Content = memoryStream.ToArray();
00354                             var imageData =
00355                                 $"data:{format};base64,{Convert.ToBase64String(Content)}";
00356                             me.ProfilePicture = imageData;
00357                         }
00358                     }
00359                     else
00360                     {
00361                         ModelState.AddModelError("File", _localizer["7"]);
00362                     }
00363                 } else {
00364                     me.ProfilePicture = null;
00365                 }
00366                 me.UserName = viewModel.UserName;
00367                 me.Email = viewModel.Email;
00368                 me.PhoneNumber = viewModel.PhoneNumber;
00369                 me.Gender = viewModel.Gender;
00370                 me.BirthDay = viewModel.BirthDay;
00371                 //me.Department = viewModel.Department.ToString();
00372
00373                 var result = await _userManager.UpdateAsync(me);
00374                 if (result.Succeeded)
00375                 {
00376                     await _userManager.UpdateSecurityStampAsync(me);
00377                     await _signInManager.SignOutAsync();
00378                     await _signInManager.SignInAsync(me, true);
00379
00380                     _logger.LogInformation("AUDIT: " + me.UserName + "successfully updated own
00381 user profile");
00382
00383                     return RedirectToAction("Index", "Home");
00384                 }
00385                 result.Errors.ToList().ForEach(f => ModelState.AddModelError(string.Empty,
00386 f.Description));
00387             }
00388             else
00389             {
00390                 await _signInManager.SignOutAsync();
00391                 return RedirectToAction("Index", "Home");
00392             }
00393             return View(viewModel);
00394         }

```

References [DotNetIdentity.Models.ViewModels.UpdateProfileViewModel.BirthDay](#), [DotNetIdentity.Models.ViewModels.UpdateProfileV](#), [DotNetIdentity.Models.ViewModels.UpdateProfileViewModel.Gender](#), [DotNetIdentity.Models.ViewModels.UpdateProfileViewModel.PH](#), [DotNetIdentity.Models.ViewModels.UpdateProfileViewModel.ProfilePicture](#), and [DotNetIdentity.Models.ViewModels.UpdateProfileV](#)

### 6.43.3.16 Register() [1/2]

`IActionResult DotNetIdentity.Controllers.UserController.Register ( )`

Controller action for Register view

## Returns

view Register

## 6.43.3.17 Register() [2/2]

```
async Task< IActionResult > DotNetIdentity.Controllers.UserController.Register (
    SignUpViewModel viewModel ) [inline]
```

Controller POST method for Register view

## Parameters

viewModel	
-----------	--

## Returns

Redirect to action Login

Definition at line 98 of file [UserController.cs](#).

```
00099     {
00100         if (ModelState.IsValid)
00101         {
00102             var user = new AppUser
00103             {
00104                 FirstName = viewModel.FirstName,
00105                 LastName = viewModel.LastName,
00106                 UserName = viewModel.UserName,
00107                 Email = viewModel.Email,
00108                 Gender = viewModel.Gender,
00109                 BirthDay = viewModel.BirthDay,
00110                 Department = viewModel.Department.ToString(),
00111                 TwoFactorType = Models.TwoFactorType.None,
00112                 CreatedOn = DateTime.UtcNow,
00113                 IsEnabled = true
00114             };
00115
00116             var result = await _userManager.CreateAsync(user, viewModel.Password);
00117             if (result.Succeeded)
00118             {
00119                 await _userManager.AddToRoleAsync(user, "User");
00120                 var confirmationToken = await
00121                 _userManager.GenerateEmailConfirmationTokenAsync(user);
00122                 var confirmationLink = Url.Action("ConfirmEmail", "User", new
00123                 {
00124                     userId = user.Id,
00125                     token = confirmationToken
00126                 }, HttpContext.Request.Scheme);
00127                 await _emailHelper.SendAsync(new()
00128                 {
00129                     Subject = "Confirm e-mail",
00130                     Body = $"Please <a href='{confirmationLink}'>click</a> to confirm your e-mail
00131 address.",
00132                     To = user.Email
00133                 });
00134                 _logger.LogInformation("AUDIT: " + "registered successfully new user: " +
00135                 user.UserName);
00136                 return RedirectToAction("Login");
00137             }
00138             result.Errors.ToList().ForEach(f => ModelState.AddModelError(string.Empty,
00139             f.Description));
00140             return View(viewModel);
00141         }
```

References [DotNetIdentity.Models.ViewModels.SignUpViewModel.BirthDay](#), [DotNetIdentity.Models.ViewModels.SignUpViewModel.D](#), [DotNetIdentity.Models.ViewModels.SignUpViewModel.Email](#), [DotNetIdentity.Models.ViewModels.SignUpViewModel.FirstName](#),

[DotNetIdentity.Models.ViewModels.SignUpViewModel.Gender](#), [DotNetIdentity.Models.ViewModels.SignUpViewModel.LastName](#), [DotNetIdentity.Models.ViewModels.SignUpViewModel.Password](#), and [DotNetIdentity.Models.ViewModels.SignUpViewModel.UserName](#)

### 6.43.3.18 ResetPassword() [1/2]

```
async Task< IActionResult > DotNetIdentity.Controllers.UserController.ResetPassword (
    ResetPasswordViewModel viewModel ) [inline]
```

controller POST method for ResetPassword view

#### Parameters

<i>viewModel</i>	type ResetPasswordViewModel
------------------	-----------------------------

#### Returns

redirect to action Login

Definition at line 513 of file [UserController.cs](#).

```
00514     {
00515         if (ModelState.IsValid)
00516         {
00517             var user = await _userManager.FindByIdAsync(viewModel.UserId);
00518             if (user != null)
00519             {
00520                 var result = await _userManager.ResetPasswordAsync(user, viewModel.Token,
viewModel.Password);
00521                 if (result.Succeeded)
00522                 {
00523                     await _userManager.UpdateSecurityStampAsync(user);
00524                     _logger.LogInformation("AUDIT: " + user.UserName + "successfully reseted own
password.");
00525                     return RedirectToAction("Login", "User");
00526                 }
00527                 else
00528                 {
00529                     result.Errors.ToList().ForEach(f => ModelState.AddModelError(string.Empty,
f.Description));
00530                 }
00531             }
00532             else
00533             {
00534                 ModelState.AddModelError(string.Empty, "User not found.");
00535             }
00536         }
00537         return View(viewModel);
00538     }
```

References [DotNetIdentity.Models.ViewModels.ResetPasswordViewModel.Password](#), [DotNetIdentity.Models.ViewModels.ResetPasswordViewModel.Token](#), and [DotNetIdentity.Models.ViewModels.ResetPasswordViewModel.UserId](#).

### 6.43.3.19 ResetPassword() [2/2]

```
IActionResult DotNetIdentity.Controllers.UserController.ResetPassword (
    string userId,
    string token ) [inline]
```

controller action for ResetPassword view



## Parameters

<i>userId</i>	the user id
<i>token</i>	the reset token

## Returns

view ResetPassword

Definition at line 493 of file [UserController.cs](#).

```

00494     {
00495         if (userId == null || token == null)
00496         {
00497             return RedirectToAction("Login", "User");
00498         }
00499
00500         return View(new ResetPasswordViewModel
00501         {
00502             UserId = userId,
00503             Token = token
00504         });
00505     }

```

## 6.43.3.20 TwoFactorAuthenticator() [1/2]

```

async Task< IActionResult > DotNetIdentity.Controllers.UserController.TwoFactorAuthenticator (
) [inline]

```

controller action for view TwoFactorAuthenticator

## Returns

view TwoFactorAuthenticator

Definition at line 663 of file [UserController.cs](#).

```

00664     {
00665         var user = await _userManager.FindByNameAsync(User.Identity?.Name);
00666         if (user.TwoFactorEnabled && user.TwoFactorType == Models.TwoFactorType.Authenticator)
00667         {
00668             var authenticatorKey = await _userManager.GetAuthenticatorKeyAsync(user);
00669             if (authenticatorKey == null)
00670             {
00671                 await _userManager.ResetAuthenticatorKeyAsync(user);
00672                 authenticatorKey = await _userManager.GetAuthenticatorKeyAsync(user);
00673             }
00674
00675             return View(new TwoFactorAuthenticatorViewModel
00676             {
00677                 SharedKey = authenticatorKey,
00678                 AuthenticationUri = _twoFactorAuthService.GenerateQrCodeUri(user.Email,
authenticatorKey)
00679             });
00680         }
00681         return RedirectToAction("Index", "Home");
00682     }

```

## 6.43.3.21 TwoFactorAuthenticator() [2/2]

```

async Task< IActionResult > DotNetIdentity.Controllers.UserController.TwoFactorAuthenticator (
    TwoFactorAuthenticatorViewModel viewModel ) [inline]

```

controller POST method for view TwoFactorAuthenticator

## Parameters

<i>viewModel</i>	type <a href="#">TwoFactorAuthenticatorViewModel</a>
------------------	------------------------------------------------------

## Returns

redirect to action [TwoFactorType](#)

Definition at line 690 of file [UserController.cs](#).

```

00691     {
00692         var user = await _userManager.FindByNameAsync(User.Identity?.Name);
00693
00694         var verificationCode = viewModel.VerificationCode.Replace(" ", string.Empty).Replace("-",
string.Empty);
00695
00696         var isValidToken = await _userManager.VerifyTwoFactorTokenAsync(user,
_userManager.Options.Tokens.AuthenticatorTokenProvider, verificationCode);
00697         if (isValidToken)
00698         {
00699             if (user.IsMfaForce && user.TwoFactorEnabled==false) {
00700                 await _userManager.SetTwoFactorEnabledAsync(user, true);
00701                 user.IsMfaForce = false;
00702                 await _userManager.UpdateAsync(user);
00703             }
00704
00705             TempData["RecoveryCodes"] = await
_userManager.GenerateNewTwoFactorRecoveryCodesAsync(user, 5);
00706         }
00707         _logger.LogInformation("AUDIT: " + user.UserName + " successfully configured 2fa!");
00708         return RedirectToAction("TwoFactorType", "User");
00709     }

```

References [DotNetIdentity.Models.ViewModels.TwoFactorAuthenticatorViewModel.VerificationCode](#).

### 6.43.3.22 TwoFactorLogin() [1/2]

```

async Task< IActionResult > DotNetIdentity.Controllers.UserController.TwoFactorLogin (
    string? returnUrl ) [inline]

```

Controller action for view [TwoFactorLogin](#)

## Parameters

<i>returnUrl</i>	optional return url as string
------------------	-------------------------------

## Returns

view [TwoFactorLogin](#)

Definition at line 232 of file [UserController.cs](#).

```

00233     {
00234         if (returnUrl != null)
00235         {
00236             TempData["ReturnUrl"] = returnUrl;
00237         }
00238
00239         var user = await _signInManager.GetTwoFactorAuthenticationUserAsync();
00240
00241         if (user.TwoFactorType == DotNetIdentity.Models.TwoFactorType.Email) {
00242             var token = await _userManager.GenerateTwoFactorTokenAsync(user, "Email");
00243             await _emailHelper.SendAsync(new()

```

```

00244         {
00245             Subject = "Your 2fa code",
00246             Body = $"Please use this " + token + " 2fa code to verify your login.",
00247             To = user.Email
00248         });
00249         _logger.LogInformation("AUDIT: send otp code to " + user.Email);
00250     }
00251
00252     return View(new TwoFactorLoginVieWModel
00253     {
00254         TwoFactorType = user.TwoFactorType,
00255     });
00256 }

```

References [DotNetIdentity.Models.ViewModels.TwoFactorLoginVieWModel.TwoFactorType](#).

### 6.43.3.23 TwoFactorLogin() [2/2]

```

async Task< IActionResult > DotNetIdentity.Controllers.UserController.TwoFactorLogin (
    TwoFactorLoginVieWModel viewModel ) [inline]

```

controller POST method for TwoFactorLogin view

#### Parameters

<i>vieWModel</i>	type TwoFactorLoginVieWModel
------------------	------------------------------

#### Returns

rediret to return url

Definition at line 264 of file [UserController.cs](#).

```

00265     {
00266         var user = await _signInManager.GetTwoFactorAuthenticationUserAsync();
00267
00268         if (user.TwoFactorType == Models.TwoFactorType.Authenticator)
00269         {
00270             var result = viewModel.IsRecoveryCode ? await
00271                 _signInManager.TwoFactorRecoveryCodeSignInAsync(viewModel.VerificationCode) : await
00272                 _signInManager.TwoFactorAuthenticatorSignInAsync(viewModel.VerificationCode, true, false);
00273             if (result.Succeeded)
00274             {
00275                 _logger.LogInformation("AUDIT: " + user.UserName + "Authenticator 2fa verified
00276                 successfully");
00277                 return Redirect(TempData["ReturnUrl"]?.ToString() ?? "/");
00278             }
00279             _logger.LogWarning("AUDIT: " + user.UserName + "Authenticator 2fa verification failed.
00280             Invalid otp code");
00281             ModelState.AddModelError(string.Empty, _localizer["5"]);
00282         }
00283         else if (user.TwoFactorType == Models.TwoFactorType.Email)
00284         {
00285             var result = viewModel.IsRecoveryCode ? await
00286                 _signInManager.TwoFactorRecoveryCodeSignInAsync(viewModel.VerificationCode) : await
00287                 _signInManager.TwoFactorSignInAsync("Email", viewModel.VerificationCode, true, false);
00288             if (result.Succeeded)
00289             {
00290                 _logger.LogInformation("AUDIT: " + user.UserName + "Email 2fa verified
00291                 successfully");
00292                 return Redirect(TempData["ReturnUrl"]?.ToString() ?? "/");
00293             }
00294             _logger.LogWarning("AUDIT: " + user.UserName + "Email 2fa verification failed. Invalid
00295             otp code");
00296             ModelState.AddModelError(string.Empty, _localizer["5"]);
00297         }
00298     }
00299     return View(viewModel);
00300 }

```

References [DotNetIdentity.Models.ViewModels.TwoFactorLoginVieWModel.VerificationCode](#).

**6.43.3.24 TwoFactorType() [1/2]**

`async Task< IActionResult > DotNetIdentity.Controllers.UserController.TwoFactorType ( ) [inline]`

controller action for view TwiFactorType

**Returns**

view TwoFactorTaype

Definition at line 566 of file [UserController.cs](#).

```
00567     {
00568         var user = await _userManager.FindByNameAsync(User.Identity?.Name);
00569
00570         return View(new TwoFactorTypeViewModel
00571         {
00572             TwoFactorType = user.TwoFactorType
00573         });
00574     }
```

**6.43.3.25 TwoFactorType() [2/2]**

`async Task< IActionResult > DotNetIdentity.Controllers.UserController.TwoFactorType ( TwoFactorTypeViewModel viewModel ) [inline]`

controller POST method for view TwoFactorType

**Parameters**

<i>viewModel</i>	TwoFactorTypeViewModel
------------------	------------------------

**Returns**

redireoct to action TwoFactorType

Definition at line 583 of file [UserController.cs](#).

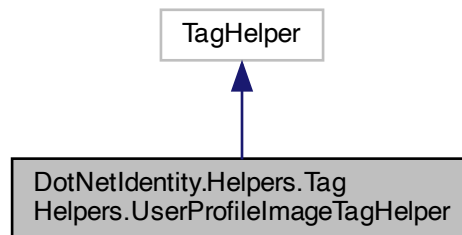
```
00584     {
00585         var user = await _userManager.FindByNameAsync(User.Identity?.Name);
00586         user.TwoFactorType = viewModel.TwoFactorType;
00587         await _userManager.UpdateAsync(user);
00588         await _userManager.SetTwoFactorEnabledAsync(user, user.TwoFactorType !=
Models.TwoFactorType.None);
00589
00590         if (viewModel.TwoFactorType == Models.TwoFactorType.Authenticator)
00591         {
00592             _logger.LogInformation("AUDIT: " + user.UserName + " set authenticator as 2fa method.
Redirecting to action: TwoFactorAuthenticator");
00593             return RedirectToAction("TwoFactorAuthenticator", "User");
00594         }
00595         _logger.LogInformation("AUDIT: " + user.UserName + " set " + user.TwoFactorType.ToString()
+ " as 2fa method.");
00596         return RedirectToAction("TwoFactorType", "User");
00597     }
```

References [DotNetIdentity.Models.ViewModels.TwoFactorTypeViewModel.TwoFactorType](#).

## 6.44 DotNetIdentity.Helpers.TagHelpers.UserProfileImageTagHelper Class Reference

html taghelper class

Inheritance diagram for DotNetIdentity.Helpers.TagHelpers.UserProfileImageTagHelper:



### Public Member Functions

- [UserProfileImageTagHelper](#) (UserManager< [AppUser](#) > userManager)  
*class constructor*
- override async Task [ProcessAsync](#) (TagHelperContext context, TagHelperOutput output)  
*override ProcessAsync*

### Properties

- string [UserId](#) = default! [get, set]  
*html attribute definition*

#### 6.44.1 Detailed Description

html taghelper class

Definition at line 12 of file [UserProfileImageTagHelper.cs](#).

#### 6.44.2 Constructor & Destructor Documentation

##### 6.44.2.1 UserProfileImageTagHelper()

```
DotNetIdentity.Helpers.TagHelpers.UserProfileImageTagHelper.UserProfileImageTagHelper (
    UserManager< AppUser > userManager ) [inline]
```

class constructor

## Parameters

<i>userManager</i>	type UserManager
--------------------	------------------

Definition at line 23 of file [UserProfileImageTagHelper.cs](#).

```
00024     {
00025         _userManager = userManager;
00026     }
```

## 6.44.3 Member Function Documentation

### 6.44.3.1 ProcessAsync()

```
override async Task DotNetIdentity.Helpers.TagHelpers.UserProfileImageTagHelper.ProcessAsync (
    TagHelperContext context,
    TagHelperOutput output ) [inline]
```

override ProcessAsync

## Parameters

<i>context</i>	type TagHelperContext
<i>output</i>	type TagHelperOutput

## Returns

TagHelperOutput

Definition at line 41 of file [UserProfileImageTagHelper.cs](#).

```
00042     {
00043         var user = await _userManager.FindByIdAsync(UserId);
00044
00045         var sb = new StringBuilder();
00046         if(user.ProfilePicture==null) {
00047             sb.Append($"<img src='/assets/images/faces/user.png' height='32' />");
00048         } else {
00049             sb.Append($"<img src='" + user.ProfilePicture + "' height='32' />");
00050         }
00051
00052         output.Content.SetHtmlContent(sb.ToString());
00053     }
```

References [DotNetIdentity.Helpers.TagHelpers.UserProfileImageTagHelper.UserId](#).

## 6.44.4 Property Documentation

#### 6.44.4.1 UserId

```
string DotNetIdentity.Helpers.TagHelpers.UserProfileImageTagHelper.UserId = default! [get],  
[set]
```

html attribute definition

Definition at line 33 of file [UserProfileImageTagHelper.cs](#).

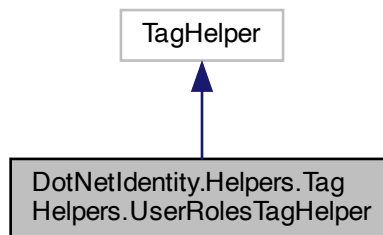
```
00033 { get; set; } = default!;
```

Referenced by [DotNetIdentity.Helpers.TagHelpers.UserProfileImageTagHelper.ProcessAsync\(\)](#).

## 6.45 DotNetIdentity.Helpers.TagHelpers.UserRolesTagHelper Class Reference

html tag helper class

Inheritance diagram for DotNetIdentity.Helpers.TagHelpers.UserRolesTagHelper:



### Public Member Functions

- [UserRolesTagHelper](#) (userManager < [AppUser](#) > userManager)  
*class constructor*
- override async Task [ProcessAsync](#) (TagHelperContext context, TagHelperOutput output)  
*method to override ProcessAsync*

### Properties

- string [UserId](#) = default! [get, set]  
*attribute definition*

#### 6.45.1 Detailed Description

html tag helper class

Definition at line 12 of file [UserRolesTagHelper.cs](#).

## 6.45.2 Constructor & Destructor Documentation

### 6.45.2.1 UserRolesTagHelper()

```
DotNetIdentity.Helpers.TagHelpers.UserRolesTagHelper.UserRolesTagHelper (
    UserManager< AppUser > userManager ) [inline]
```

class constructor

#### Parameters

<i>userManager</i>	
--------------------	--

Definition at line 23 of file [UserRolesTagHelper.cs](#).

```
00024     {
00025         _userManager = userManager;
00026     }
```

## 6.45.3 Member Function Documentation

### 6.45.3.1 ProcessAsync()

```
override async Task DotNetIdentity.Helpers.TagHelpers.UserRolesTagHelper.ProcessAsync (
    TagHelperContext context,
    TagHelperOutput output ) [inline]
```

method to override ProcessAsync

#### Parameters

<i>context</i>	type TagHelperContext
<i>output</i>	type TagHelperOutput

#### Returns

TagHelperOutput

Definition at line 41 of file [UserRolesTagHelper.cs](#).

```
00042     {
00043         var user = await _userManager.FindByIdAsync(UserId);
00044         var roles = await _userManager.GetRolesAsync(user);
00045
00046         var sb = new StringBuilder();
00047
00048         if (roles.Count() > 0)
00049         {
00050             foreach (var role in roles)
00051             {
```



```

00052         sb.Append($"<span class='badge rounded-pill bg-danger'>{role}</span>");
00053     }
00054 }
00055 else
00056 {
00057     sb.Append("No roles found");
00058 }
00059
00060 output.Content.SetHtmlContent(sb.ToString());
00061 }

```

References [DotNetIdentity.Helpers.TagHelpers.UserRolesTagHelper.UserId](#).

## 6.45.4 Property Documentation

### 6.45.4.1 UserId

```
string DotNetIdentity.Helpers.TagHelpers.UserRolesTagHelper.UserId = default! [get], [set]
```

attribute definition

Definition at line 33 of file [UserRolesTagHelper.cs](#).

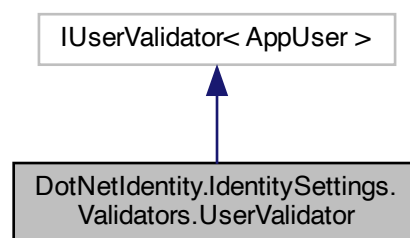
```
00033 { get; set; } = default!;
```

Referenced by [DotNetIdentity.Helpers.TagHelpers.UserRolesTagHelper.ProcessAsync\(\)](#).

## 6.46 DotNetIdentity.IdentitySettings.Validators.UserValidator Class Reference

class to validate IdentityUser

Inheritance diagram for DotNetIdentity.IdentitySettings.Validators.UserValidator:



## Public Member Functions

- Task< IdentityResult > [ValidateAsync](#) (userManager< [AppUser](#) > manager, [AppUser](#) user)  
*Task to validate IdentityUser*

### 6.46.1 Detailed Description

class to validate IdentityUser

Definition at line 9 of file [UserValidator.cs](#).

### 6.46.2 Member Function Documentation

#### 6.46.2.1 ValidateAsync()

```
Task< IdentityResult > DotNetIdentity.IdentitySettings.Validators.UserValidator.ValidateAsync
(
    UserManager< AppUser > manager,
    AppUser user ) [inline]
```

Task to validate IdentityUser

##### Parameters

<i>manager</i>	UserManager
<i>user</i>	AppUser

##### Returns

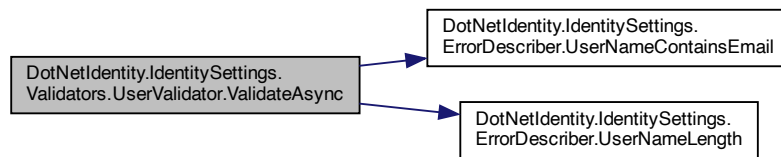
Task of type IdentityResult

Definition at line 17 of file [UserValidator.cs](#).

```
00018     {
00019         var errors = new List<IdentityError>();
00020
00021         if (user.UserName.Length < 6)
00022         {
00023             errors.Add(ErrorDescriber.UserNameLength());
00024         }
00025         if (user.Email.Substring(0, user.Email.IndexOf("@")) == user.UserName)
00026         {
00027             errors.Add(ErrorDescriber.UserNameContainsEmail());
00028         }
00029
00030         if (errors.Any())
00031         {
00032             return Task.FromResult(IdentityResult.Failed(errors.ToArray()));
00033         }
00034         return Task.FromResult(IdentityResult.Success);
00035     }
```

References [DotNetIdentity.IdentitySettings.ErrorDescriber.UserNameContainsEmail\(\)](#), and [DotNetIdentity.IdentitySettings.ErrorDescriber.UserNameLength\(\)](#).

Here is the call graph for this function:





# Index

AbsoluteExpiration  
    DotNetIdentity.Models.DataModels.SessionCache, [87](#)

AccessDenied  
    DotNetIdentity.Controllers.HomeController, [71](#)

Accounting  
    DotNetIdentity.Models, [11](#)

AdminController  
    DotNetIdentity.Controllers.AdminController, [19](#)

AppDbContext  
    DotNetIdentity.Data.AppDbContext, [32](#)

ApplicationName  
    DotNetIdentity.Models.BusinessModels.GlobalSettings, [68](#)

AppLogs  
    DotNetIdentity.Data.AppDbContext, [34](#)

AppSessionCache  
    DotNetIdentity.Data.AppDbContext, [34](#)

AppSettings  
    DotNetIdentity.Data.AppDbContext, [34](#)

AppSettingsBase  
    DotNetIdentity.Services.SettingsService.AppSettingsBase, [40](#)

Articles  
    DotNetIdentity.Controllers.RoleBasedController, [85](#)

AuditLogs  
    DotNetIdentity.Controllers.AdminController, [19](#)

AuthenticationUri  
    DotNetIdentity.Models.ViewModels.TwoFactorAuthenticationViewModel, [98](#)

BirthDay  
    DotNetIdentity.Models.Identity.AppUser, [42](#)  
    DotNetIdentity.Models.ViewModels.SignUpViewModel, [92](#)  
    DotNetIdentity.Models.ViewModels.UpdateProfileViewModel, [101](#)

Body  
    DotNetIdentity.Models.BusinessModels.EmailModel, [59](#)

ChangePassword  
    DotNetIdentity.Controllers.UserController, [106](#)

ClaimsTransformation  
    DotNetIdentity.IdentitySettings.ClaimsTransformation, [51](#)

ConfirmEmail  
    DotNetIdentity.Controllers.UserController, [107](#)

CreatedOn  
    DotNetIdentity.Models.Identity.AppRole, [39](#)  
    DotNetIdentity.Models.Identity.AppUser, [43](#)

Dashboard  
    DotNetIdentity.Controllers.RoleBasedController, [85](#)

debugUserInfo  
    DotNetIdentity.Controllers.UserController, [108](#)

DeleteRole  
    DotNetIdentity.Controllers.AdminController, [19](#)

DeleteUser  
    DotNetIdentity.Controllers.AdminController, [20](#)

Department  
    DotNetIdentity.Models, [11](#)  
    DotNetIdentity.Models.Identity.AppUser, [43](#)  
    DotNetIdentity.Models.ViewModels.EditUserViewModel, [54](#)  
    DotNetIdentity.Models.ViewModels.NewUserModel, [79](#)  
    DotNetIdentity.Models.ViewModels.SignUpViewModel, [93](#)

DisableLdapLogin  
    DotNetIdentity.Controllers.AdminController, [20](#)

DisableMfaForce  
    DotNetIdentity.Controllers.AdminController, [21](#)

DisableUser  
    DotNetIdentity.Controllers.AdminController, [21](#)

DotNetIdentity, [9](#)

DotNetIdentity.Controllers, [9](#)

DotNetIdentity.Controllers.AdminController, [17](#)

AdminController, [19](#)

AuditLogs, [19](#)

DeleteRole, [19](#)

DeleteUser, [20](#)

DisableLdapLogin, [20](#)

DisableMfaForce, [21](#)

DisableUser, [21](#)

EditUser, [22](#), [23](#)

EnableLdapLogin, [24](#)

EnableMfaForce, [24](#)

EnableUser, [25](#)

ErrorLogs, [25](#)

GetAppLogs, [26](#)

GetAuditLogs, [26](#)

GetErrorLogs, [27](#)

GetUsers, [27](#)

Index, [27](#)

NewUser, [28](#)

Roles, [29](#)

SystemLogs, [29](#)

UpsertRole, [29](#), [30](#)

Users, [31](#)

- DotNetIdentity.Controllers.ClaimBasedController, 48
  - Employee, 49
  - FreeTrial, 49
  - HR, 49
  - Software, 49
  - Violence, 50
- DotNetIdentity.Controllers.CultureController, 52
  - SetCulture, 53
- DotNetIdentity.Controllers.ErrorsController, 62
  - ErrorCd, 62
  - ErrorEx, 63
- DotNetIdentity.Controllers.HomeController, 70
  - AccessDenied, 71
  - HomeController, 70
  - Index, 71
- DotNetIdentity.Controllers.RoleBasedController, 85
  - Articles, 85
  - Dashboard, 85
  - Profile, 86
  - Users, 86
- DotNetIdentity.Controllers.UserController, 104
  - ChangePassword, 106
  - ConfirmEmail, 107
  - debugUserInfo, 108
  - EnforceTwoFactorAuthenticator, 108
  - EnforceTwoFactorResult, 109
  - ForgotPassword, 109, 110
  - Login, 110, 112
  - Logout, 112
  - ping, 112
  - Profile, 113
  - Register, 114, 115
  - ResetPassword, 116
  - TwoFactorAuthenticator, 117
  - TwoFactorLogin, 118, 119
  - TwoFactorType, 119, 120
  - UserController, 106
- DotNetIdentity.Data, 9
- DotNetIdentity.Data.AppDbContext, 31
  - AppDbContext, 32
  - AppLogs, 34
  - AppSessionCache, 34
  - AppSettings, 34
  - OnModelCreating, 32
- DotNetIdentity.Helpers, 10
- DotNetIdentity.Helpers.EmailHelper, 57
  - EmailHelper, 57
  - SendAsync, 58
- DotNetIdentity.Helpers.TagHelpers, 10
- DotNetIdentity.Helpers.TagHelpers.UserProfileImageTagHelper, To, 59
  - 121
  - ProcessAsync, 122
  - UserId, 122
  - UserProfileImageTagHelper, 121
- DotNetIdentity.Helpers.TagHelpers.UserRolesTagHelper,
  - 123
  - ProcessAsync, 124
  - UserId, 125
  - UserRolesTagHelper, 124
- DotNetIdentity.IdentitySettings, 10
- DotNetIdentity.IdentitySettings.ClaimsTransformation,
  - 50
  - ClaimsTransformation, 51
  - TransformAsync, 51
- DotNetIdentity.IdentitySettings.ErrorDescriber, 60
  - PasswordContainsUsername, 61
  - UserNameContainsEmail, 61
  - UserNameLength, 61
- DotNetIdentity.IdentitySettings.Requirements, 10
- DotNetIdentity.IdentitySettings.Requirements.FreeTrialExpireHandler,
  - 66
  - HandleRequirementAsync, 66
- DotNetIdentity.IdentitySettings.Requirements.FreeTrialExpireRequirement,
  - 67
- DotNetIdentity.IdentitySettings.Requirements.MinimumAgeHandler,
  - 75
  - HandleRequirementAsync, 76
- DotNetIdentity.IdentitySettings.Requirements.MinimumAgeRequirement,
  - 77
  - MinimumAge, 78
  - MinimumAgeRequirement, 77
- DotNetIdentity.IdentitySettings.TwoFactorAuthenticationService,
  - 96
  - GenerateQrCodeUri, 97
  - TwoFactorAuthenticationService, 96
- DotNetIdentity.IdentitySettings.Validators, 11
- DotNetIdentity.IdentitySettings.Validators.PasswordValidator,
  - 82
  - ValidateAsync, 82
- DotNetIdentity.IdentitySettings.Validators.UserValidator,
  - 125
  - ValidateAsync, 126
- DotNetIdentity.Models, 11
  - Accounting, 11
  - Department, 11
  - Email, 13
  - Gender, 11
  - HR, 11
  - Male, 13
  - None, 13
  - Software, 11
  - TwoFactorType, 13
  - Unknown, 13
- DotNetIdentity.Models.BusinessModels, 13
- DotNetIdentity.Models.BusinessModels.EmailModel, 58
  - Body, 59
  - Subject, 59
- DotNetIdentity.Models.BusinessModels.GlobalSettings,
  - 68
  - ApplicationName, 68
  - SessionCookieExpiration, 69
  - SessionTimeoutRedirAfter, 69
  - SessionTimeoutWarnAfter, 69
- DotNetIdentity.Models.BusinessModels.MailSettings, 73
  - Password, 74

- SmtpFromAddress, 74
  - SmtpPort, 74
  - SmtpServer, 74
  - SmtpUseTls, 75
  - Username, 75
- DotNetIdentity.Models.DataModels, 13
- DotNetIdentity.Models.DataModels.ApplicationSettings, 34
  - Name, 35
  - Type, 35
  - Value, 35
- DotNetIdentity.Models.DataModels.AppLogs, 36
  - Exception, 36
  - id, 37
  - Level, 37
  - Message, 37
  - MessageTemplate, 37
  - Properties, 38
  - Timestamp, 38
- DotNetIdentity.Models.DataModels.SessionCache, 86
  - AbsoluteExpiration, 87
  - ExpiresAtTime, 87
  - id, 87
  - SlidingExpirationInSeconds, 87
  - Value, 88
- DotNetIdentity.Models.Identity, 14
- DotNetIdentity.Models.Identity.AppRole, 38
  - CreatedOn, 39
- DotNetIdentity.Models.Identity.AppUser, 41
  - BirthDay, 42
  - CreatedOn, 43
  - Department, 43
  - FirstName, 43
  - Gender, 43
  - IsEnabled, 44
  - IsLdapLogin, 44
  - IsMfaForce, 44
  - LastName, 44
  - ProfilePicture, 45
  - TwoFactorType, 45
- DotNetIdentity.Models.ViewModels, 14
- DotNetIdentity.Models.ViewModels.AssignRoleViewModel, 45
  - IsAssigned, 46
  - RoleId, 46
  - RoleName, 46
- DotNetIdentity.Models.ViewModels.ChangePasswordViewModel, 47
  - NewPassword, 47
  - NewPasswordConfirm, 47
  - Password, 47
- DotNetIdentity.Models.ViewModels.EditUserViewModel, 54
  - Department, 54
  - Email, 55
  - FirstName, 55
  - Gender, 55
  - Id, 55
  - LastName, 56
  - PhoneNumber, 56
  - Roles, 56
  - UserName, 56
- DotNetIdentity.Models.ViewModels.ErrorViewModel, 64
  - RequestId, 64
  - ShowRequestId, 64
- DotNetIdentity.Models.ViewModels.ForgotPasswordViewModel, 65
  - Email, 65
- DotNetIdentity.Models.ViewModels.NewUserModel, 78
  - Department, 79
  - Email, 79
  - FirstName, 79
  - Gender, 79
  - IsLdapLogin, 80
  - IsMfaForce, 80
  - LastName, 80
  - Password, 80
  - Roles, 81
  - UserName, 81
- DotNetIdentity.Models.ViewModels.ResetPasswordViewModel, 83
  - Password, 84
  - Token, 84
  - UserId, 84
- DotNetIdentity.Models.ViewModels.SignInViewModel, 90
  - Password, 91
  - RememberMe, 91
  - UserName, 91
- DotNetIdentity.Models.ViewModels.SignUpViewModel, 92
  - BirthDay, 92
  - Department, 93
  - Email, 93
  - FirstName, 93
  - Gender, 93
  - LastName, 94
  - Password, 94
  - UserName, 94
- DotNetIdentity.Models.ViewModels.StatusCodesModel, 95
  - ErrorMessage, 95
  - RouteOfException, 95
  - StatusCode, 95
- DotNetIdentity.Models.ViewModels.TwoFactorAuthenticatorViewModel, 97
  - AuthenticationUri, 98
  - SharedKey, 98
  - VerificationCode, 98
- DotNetIdentity.Models.ViewModels.TwoFactorLoginViewWModel, 99
  - IsRecoveryCode, 99
  - TwoFactorType, 99
  - VerificationCode, 99
- DotNetIdentity.Models.ViewModels.TwoFactorTypeViewModel, 100

- TwoFactorType, [100](#)
- DotNetIdentity.Models.ViewModels.UpdateProfileViewModel, [101](#)
- BirthDay, [101](#)
- Email, [101](#)
- Gender, [102](#)
- PhoneNumber, [102](#)
- ProfilePicture, [102](#)
- UserName, [102](#)
- DotNetIdentity.Models.ViewModels.UpsertRoleViewModel, [103](#)
- Id, [103](#)
- Name, [103](#)
- DotNetIdentity.Services, [15](#)
- DotNetIdentity.Services.SettingsService, [15](#)
- DotNetIdentity.Services.SettingsService.AppSettingsBase, [39](#)
- AppSettingsBase, [40](#)
- Load, [40](#)
- Save, [41](#)
- DotNetIdentity.Services.SettingsService.ISettingsService, [71](#)
- Global, [72](#)
- Mail, [72](#)
- Save, [72](#)
- DotNetIdentity.Services.SettingsService.SettingsService, [88](#)
- Global, [90](#)
- Mail, [90](#)
- Save, [89](#)
- SettingsService, [89](#)
- EditUser
  - DotNetIdentity.Controllers.AdminController, [22](#), [23](#)
- Email
  - DotNetIdentity.Models, [13](#)
  - DotNetIdentity.Models.ViewModels.EditUserViewModel, [55](#)
  - DotNetIdentity.Models.ViewModels.ForgotPasswordViewModel, [65](#)
  - DotNetIdentity.Models.ViewModels.NewUserModel, [79](#)
  - DotNetIdentity.Models.ViewModels.SignUpViewModel, [93](#)
  - DotNetIdentity.Models.ViewModels.UpdateProfileViewModel, [101](#)
- EmailHelper
  - DotNetIdentity.Helpers.EmailHelper, [57](#)
- Employee
  - DotNetIdentity.Controllers.ClaimBasedController, [49](#)
- EnableLdapLogin
  - DotNetIdentity.Controllers.AdminController, [24](#)
- EnableMfaForce
  - DotNetIdentity.Controllers.AdminController, [24](#)
- EnableUser
  - DotNetIdentity.Controllers.AdminController, [25](#)
- EnforceTwoFactorAuthenticator
  - DotNetIdentity.Controllers.UserController, [108](#)
- EnforceTwoFactorResult
  - DotNetIdentity.Controllers.UserController, [109](#)
- ErrorCd
  - DotNetIdentity.Controllers.ErrorsController, [62](#)
- ErrorEx
  - DotNetIdentity.Controllers.ErrorsController, [63](#)
- ErrorLogs
  - DotNetIdentity.Controllers.AdminController, [25](#)
- ErrorMessage
  - DotNetIdentity.Models.ViewModels.StatusCodesModel, [95](#)
- Exception
  - DotNetIdentity.Models.DataModels.AppLogs, [36](#)
- ExpiresAtTime
  - DotNetIdentity.Models.DataModels.SessionCache, [87](#)
- FirstName
  - DotNetIdentity.Models.Identity.AppUser, [43](#)
  - DotNetIdentity.Models.ViewModels.EditUserViewModel, [55](#)
  - DotNetIdentity.Models.ViewModels.NewUserModel, [79](#)
  - DotNetIdentity.Models.ViewModels.SignUpViewModel, [93](#)
- ForgotPassword
  - DotNetIdentity.Controllers.UserController, [109](#), [110](#)
- FreeTrial
  - DotNetIdentity.Controllers.ClaimBasedController, [49](#)
- Gender
  - DotNetIdentity.Models, [11](#)
  - DotNetIdentity.Models.Identity.AppUser, [43](#)
  - DotNetIdentity.Models.ViewModels.EditUserViewModel, [55](#)
  - DotNetIdentity.Models.ViewModels.NewUserModel, [79](#)
  - DotNetIdentity.Models.ViewModels.SignUpViewModel, [93](#)
  - DotNetIdentity.Models.ViewModels.UpdateProfileViewModel, [102](#)
- GenerateQrCodeUri
  - DotNetIdentity.IdentitySettings.TwoFactorAuthenticationService, [97](#)
- GetAppLogs
  - DotNetIdentity.Controllers.AdminController, [26](#)
- GetAuditLogs
  - DotNetIdentity.Controllers.AdminController, [26](#)
- GetErrorLogs
  - DotNetIdentity.Controllers.AdminController, [27](#)
- GetUsers
  - DotNetIdentity.Controllers.AdminController, [27](#)
- Global
  - DotNetIdentity.Services.SettingsService.ISettingsService, [72](#)
  - DotNetIdentity.Services.SettingsService.SettingsService, [90](#)



- HandleRequirementAsync
  - DotNetIdentity.IdentitySettings.Requirements.FreeTrialExpiration, 66
  - DotNetIdentity.IdentitySettings.Requirements.MinimumAgeRequirement, 76
- HomeController
  - DotNetIdentity.Controllers.HomeController, 70
- HR
  - DotNetIdentity.Controllers.ClaimBasedController, 49
  - DotNetIdentity.Models, 11
- Id
  - DotNetIdentity.Models.ViewModels.EditUserViewModel, 55
  - DotNetIdentity.Models.ViewModels.UpsertRoleViewModel, 103
- id
  - DotNetIdentity.Models.DataModels.AppLogs, 37
  - DotNetIdentity.Models.DataModels.SessionCache, 87
- Index
  - DotNetIdentity.Controllers.AdminController, 27
  - DotNetIdentity.Controllers.HomeController, 71
- IsAssigned
  - DotNetIdentity.Models.ViewModels.AssignRoleViewModel, 46
- IsEnabled
  - DotNetIdentity.Models.Identity.AppUser, 44
- IsLdapLogin
  - DotNetIdentity.Models.Identity.AppUser, 44
  - DotNetIdentity.Models.ViewModels.NewUserModel, 80
- IsMfaForce
  - DotNetIdentity.Models.Identity.AppUser, 44
  - DotNetIdentity.Models.ViewModels.NewUserModel, 80
- IsRecoveryCode
  - DotNetIdentity.Models.ViewModels.TwoFactorLoginViewModel, 99
- LastName
  - DotNetIdentity.Models.Identity.AppUser, 44
  - DotNetIdentity.Models.ViewModels.EditUserViewModel, 56
  - DotNetIdentity.Models.ViewModels.NewUserModel, 80
  - DotNetIdentity.Models.ViewModels.SignUpViewModel, 94
- Level
  - DotNetIdentity.Models.DataModels.AppLogs, 37
- Load
  - DotNetIdentity.Services.SettingsService.AppSettingsBase, 40
- Login
  - DotNetIdentity.Controllers.UserController, 110, 112
- Logout
  - DotNetIdentity.Controllers.UserController, 112
- Mail
  - DotNetIdentity.Services.SettingsService.ISettingsService, 72
  - DotNetIdentity.Services.SettingsService.SettingsService, 90
- Male
  - DotNetIdentity.Models, 13
- Message
  - DotNetIdentity.Models.DataModels.AppLogs, 37
- MessageTemplate
  - DotNetIdentity.Models.DataModels.AppLogs, 37
- MinimumAge
  - DotNetIdentity.IdentitySettings.Requirements.MinimumAgeRequirement, 78
- MinimumAgeRequirement
  - DotNetIdentity.IdentitySettings.Requirements.MinimumAgeRequirement, 77
- Name
  - DotNetIdentity.Models.DataModels.ApplicationSettings, 35
  - DotNetIdentity.Models.ViewModels.UpsertRoleViewModel, 103
- NewPassword
  - DotNetIdentity.Models.ViewModels.ChangePasswordViewModel, 47
- NewPasswordConfirm
  - DotNetIdentity.Models.ViewModels.ChangePasswordViewModel, 47
- NewUser
  - DotNetIdentity.Controllers.AdminController, 28
- None
  - DotNetIdentity.Models, 13
- OnModelCreating
  - DotNetIdentity.Data.AppDbContext, 32
- Password
  - DotNetIdentity.Models.BusinessModels.MailSettings, 74
  - DotNetIdentity.Models.ViewModels.ChangePasswordViewModel, 47
  - DotNetIdentity.Models.ViewModels.NewUserModel, 80
  - DotNetIdentity.Models.ViewModels.ResetPasswordViewModel, 84
  - DotNetIdentity.Models.ViewModels.SignInViewModel, 91
  - DotNetIdentity.Models.ViewModels.SignUpViewModel, 94
- PasswordContainsUsername
  - DotNetIdentity.IdentitySettings.ErrorDescriber, 61
- PhoneNumber
  - DotNetIdentity.Models.ViewModels.EditUserViewModel, 56
  - DotNetIdentity.Models.ViewModels.UpdateProfileViewModel, 102
- ping
  - DotNetIdentity.Controllers.UserController, 112

- ProcessAsync
  - DotNetIdentity.Helpers.TagHelpers.UserProfileImageTagHelper, 122
  - DotNetIdentity.Helpers.TagHelpers.UserRolesTagHelper, 124
- Profile
  - DotNetIdentity.Controllers.RoleBasedController, 86
  - DotNetIdentity.Controllers.UserController, 113
- ProfilePicture
  - DotNetIdentity.Models.Identity.AppUser, 45
  - DotNetIdentity.Models.ViewModels.UpdateProfileViewModel, 102
- Properties
  - DotNetIdentity.Models.DataModels.AppLogs, 38
- Register
  - DotNetIdentity.Controllers.UserController, 114, 115
- RememberMe
  - DotNetIdentity.Models.ViewModels.SignInViewModel, 91
- RequestId
  - DotNetIdentity.Models.ViewModels.ErrorViewModel, 64
- ResetPassword
  - DotNetIdentity.Controllers.UserController, 116
- RoleId
  - DotNetIdentity.Models.ViewModels.AssignRoleViewModel, 46
- RoleName
  - DotNetIdentity.Models.ViewModels.AssignRoleViewModel, 46
- Roles
  - DotNetIdentity.Controllers.AdminController, 29
  - DotNetIdentity.Models.ViewModels.EditUserViewModel, 56
  - DotNetIdentity.Models.ViewModels.NewUserModel, 81
- RouteOfException
  - DotNetIdentity.Models.ViewModels.StatusCodesModel, 95
- Save
  - DotNetIdentity.Services.SettingsService.AppSettingsBase, 41
  - DotNetIdentity.Services.SettingsService.ISettingsService, 72
  - DotNetIdentity.Services.SettingsService.SettingsService, 89
- SendAsync
  - DotNetIdentity.Helpers.EmailHelper, 58
- SessionCookieExpiration
  - DotNetIdentity.Models.BusinessModels.GlobalSettings, 69
- SessionTimeoutRedirAfter
  - DotNetIdentity.Models.BusinessModels.GlobalSettings, 69
- SessionTimeoutWarnAfter
  - DotNetIdentity.Models.BusinessModels.GlobalSettings, 69
- SetCulture
  - DotNetIdentity.Controllers.CultureController, 53
- SettingsService
  - DotNetIdentity.Services.SettingsService.SettingsService, 89
- SharedKey
  - DotNetIdentity.Models.ViewModels.TwoFactorAuthenticatorViewModel, 98
- ShowRequestId
  - DotNetIdentity.Models.ViewModels.ErrorViewModel, 64
- SlidingExpirationInSeconds
  - DotNetIdentity.Models.DataModels.SessionCache, 87
- SmtpFromAddress
  - DotNetIdentity.Models.BusinessModels.MailSettings, 74
- SmtpPort
  - DotNetIdentity.Models.BusinessModels.MailSettings, 74
- SmtpServer
  - DotNetIdentity.Models.BusinessModels.MailSettings, 74
- SmtpUseTls
  - DotNetIdentity.Models.BusinessModels.MailSettings, 75
- Software
  - DotNetIdentity.Controllers.ClaimBasedController, 49
- StatusCodes
  - DotNetIdentity.Models.ViewModels.StatusCodesModel, 95
- Subject
  - DotNetIdentity.Models.BusinessModels.EmailModel, 59
- SystemLogs
  - DotNetIdentity.Controllers.AdminController, 29
- Timestamp
  - DotNetIdentity.Models.DataModels.AppLogs, 38
- To
  - DotNetIdentity.Models.BusinessModels.EmailModel, 59
- Token
  - DotNetIdentity.Models.ViewModels.ResetPasswordViewModel, 84
- TransformAsync
  - DotNetIdentity.IdentitySettings.ClaimsTransformation, 51
- TwoFactorAuthenticationService
  - DotNetIdentity.IdentitySettings.TwoFactorAuthenticationService, 96
- TwoFactorAuthenticator
  - DotNetIdentity.Controllers.UserController, 117
- TwoFactorLogin
  - DotNetIdentity.Controllers.UserController, 118, 119
- TwoFactorType
  - DotNetIdentity.Controllers.UserController, 119, 120

- DotNetIdentity.Models, [13](#)
- DotNetIdentity.Models.Identity.AppUser, [45](#)
- DotNetIdentity.Models.ViewModels.TwoFactorLoginViewModel, [99](#)
- DotNetIdentity.Models.ViewModels.TwoFactorTypeViewModel, [98](#)
- DotNetIdentity.Models.ViewModels.TwoFactorLoginViewWModel, [99](#)
- Type
  - DotNetIdentity.Models.DataModels.ApplicationSettings, [35](#)
  - DotNetIdentity.Models.DataModels.SessionCache, [88](#)
  - DotNetIdentity.Models.ViewModels.TwoFactorAuthenticatorViewModel, [99](#)
  - DotNetIdentity.Controllers.ClaimBasedController, [50](#)
- Unknown
  - DotNetIdentity.Models, [13](#)
- UpsertRole
  - DotNetIdentity.Controllers.AdminController, [29](#), [30](#)
- UserController
  - DotNetIdentity.Controllers.UserController, [106](#)
- UserId
  - DotNetIdentity.Helpers.TagHelpers.UserProfileImageTagHelper, [122](#)
  - DotNetIdentity.Helpers.TagHelpers.UserRolesTagHelper, [125](#)
  - DotNetIdentity.Models.ViewModels.ResetPasswordViewModel, [84](#)
- UserName
  - DotNetIdentity.Models.ViewModels.EditUserViewModel, [56](#)
  - DotNetIdentity.Models.ViewModels.NewUserModel, [81](#)
  - DotNetIdentity.Models.ViewModels.SignInViewModel, [91](#)
  - DotNetIdentity.Models.ViewModels.SignUpViewModel, [94](#)
  - DotNetIdentity.Models.ViewModels.UpdateProfileViewModel, [102](#)
- Username
  - DotNetIdentity.Models.BusinessModels.MailSettings, [75](#)
- UserNameContainsEmail
  - DotNetIdentity.IdentitySettings.ErrorDescriber, [61](#)
- UserNameLength
  - DotNetIdentity.IdentitySettings.ErrorDescriber, [61](#)
- UserProfileImageTagHelper
  - DotNetIdentity.Helpers.TagHelpers.UserProfileImageTagHelper, [121](#)
- UserRolesTagHelper
  - DotNetIdentity.Helpers.TagHelpers.UserRolesTagHelper, [124](#)
- Users
  - DotNetIdentity.Controllers.AdminController, [31](#)
  - DotNetIdentity.Controllers.RoleBasedController, [86](#)
- ValidateAsync
  - DotNetIdentity.IdentitySettings.Validators.PasswordValidator, [82](#)
  - DotNetIdentity.IdentitySettings.Validators.UserValidator, [126](#)
- Value
  - DotNetIdentity.Models.DataModels.ApplicationSettings, [35](#)