DotnetIdentity

Generated by Doxygen 1.9.4

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

Chapter 2

Namespace Index

2.1 Namespace List

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

Chapter 3

Hierarchical Index

3.1 Class Hierarchy

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

Chapter 4

Class Index

4.1 Class List

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

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 SettingsController

  controller class for settings views
- 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 IdentotyError 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 }
```

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 LdapSettings

  class to define ldap settings
- 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 TwoFactorLoginVieWModel

  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.SettingsService Namespace Reference

Classes

- class AppSettingsBase

  base class for Settings classes
- interface ISettingsService

  Interface class for SettingsService
- class SettingsService

  class for settings service

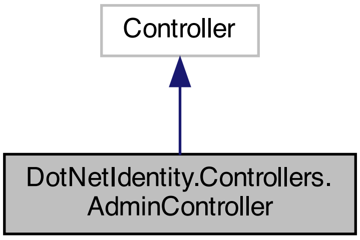
Chapter 6

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 AutditLogs
- 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)

  controler 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

db	type DotNetIdentity.Data.AppDbContext
log	type Microsoft.Extensions.Logging.ILogger
conf	type Microsoft.Extensions.Configuration.IConfiguration
userManager	type Microsoft.AspNetCore.Identity.UserManager
roleManager	type Microsoft.AspNetCore.Identity.RoleManager
signInManager	type Microsoft.AspNetCore.Identity.SignInManager

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

id	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,
       f.Description));
00400                 }
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

id	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,
       f.Description));
00377                 }
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

UserId	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

UserId	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

UserId	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

viewModel	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
```

```
                    {
                        user.FirstName = viewModel.FirstName;
                        user.LastName = viewModel.LastName;
                        user.UserName = viewModel.UserName;
                        user.Email = viewModel.Email;
                        user.PhoneNumber = viewModel.PhoneNumber;
                        user.Gender = viewModel.Gender;
                        user.Department = viewModel.Department.ToString();

                        var result = await _userManager.UpdateAsync(user);
                        if (result.Succeeded)
                        {
                            // Roles
                            foreach (var item in viewModel.Roles)
                            {
                                if (item.IsAssigned)
                                {
                                    await _userManager.AddToRoleAsync(user, item.RoleName);
                                }
                                else
                                {
                                    await _userManager.RemoveFromRoleAsync(user, item.RoleName);
                                }
                            }

                            // Claims
                            var userClaims = await _userManager.GetClaimsAsync(user);
                            var departmentClaim = userClaims.FirstOrDefault(a => a.Type ==
"Department");
                            var claimsToAdd = new Claim("Department",
Enum.GetName(viewModel.Department)!);
                            if (departmentClaim != null)
                            {
                                await _userManager.ReplaceClaimAsync(user, departmentClaim,
claimsToAdd);
                            }
                            else
                            {
                                await _userManager.AddClaimAsync(user, claimsToAdd);
                            }
                            _logger.LogInformation("AUDIT: " + User.Identity!.Name + " modified user "
+ user.UserName);
                            return RedirectToAction("Users");
                        }
                        result.Errors.ToList().ForEach(f => ModelState.AddModelError(string.Empty,
f.Description));
                    }
                }
                else
                {
                    ModelState.AddModelError(string.Empty, "User not found.");
                }
            }

            return View(viewModel);
        }
```

References DotNetIdentity.Models.ViewModels.EditUserViewModel.Department, DotNetIdentity.Models.ViewModels.EditUserViewMo 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

id	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

UserId	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

UserId	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

UserId	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()

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

Controller Action for ErrorLogs

Returns

View of type DotnetIdentity.Views.Admin.Errorogs

6.1.3.13 GetAppLogs()

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

Controller Post Method to fetch Applicationlogs

Parameters

request	type DatatablesJs.Data.DataRequest

Returns

Jason-Array of DatatablesJs.Data.DataResult

Definition at line 119 of file AdminController.cs.

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

6.1.3.14 GetAuditLogs()

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

Controller Post Method to fetch AuditLogs

Parameters

request	type DatatablesJs.Data.DataRequest

Returns

Jason-Array of DatatablesJs.Data.DataResult

Definition at line 131 of file AdminController.cs.

```
00132         {
00133             DataResult<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

request	type DatatablesJs.Data.DataRequest

Returns

Jason-Array of DatatablesJs.Data.DataResult

Definition at line 107 of file AdminController.cs.

```
00108         {
00109             DataResult<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

request	type DatatablesJs.Data.DataRequest

Returns

Jason-Array of DatatablesJs.Data.DataResult

Definition at line 143 of file AdminController.cs.

```
00143                                             {
00144             DataResult<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 
00161             return View(mod);
00162         }
```

6.1.3.19 NewUser() [2/2]

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

controler Post Method to save a new User

Parameters

viewModel	type DotnetIdentity.Models.ViewModels.NewUserModel

Returns

view of type DotnetIdentity.Views.Admin.NewUser

Definition at line 170 of file AdminController.cs.

```
00171         {
```

```
            if (ModelState.IsValid)
            {
                var user = new AppUser();
                user.IsEnabled = true;
                user.IsMfaForce = viewModel.IsMfaForce;
                user.IsLdapLogin = viewModel.IsLdapLogin;
                user.FirstName = viewModel.FirstName;
                user.LastName = viewModel.LastName;
                user.Email = viewModel.Email;
                user.EmailConfirmed = true;
                user.Gender = viewModel.Gender;
                user.Department = viewModel.Department.ToString();
                user.UserName = viewModel.UserName;

                var result = await _userManager.CreateAsync(user, viewModel.Password);
                if (result.Succeeded)
                {
                    // roles
                    foreach (var item in viewModel.Roles)
                    {
                        if (item.IsAssigned)
                        {
                            await _userManager.AddToRoleAsync(user, item.RoleName);
                        }
                        else
                        {
                            await _userManager.RemoveFromRoleAsync(user, item.RoleName);
                        }
                    }
                }

                ViewData["message"] = "User created success fully.";
                _logger.LogInformation("AUDIT: " + User.Identity!.Name + " created new user " +
       user.UserName);
            }
            return View(new NewUserModel());
        }
```

References DotNetIdentity.Models.ViewModels.NewUserModel.Department, DotNetIdentity.Models.ViewModels.NewUserModel.Ema 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

id	

Returns

view of type Dotnetidentity.Views.Admin.UpsertRole

Definition at line 214 of file AdminController.cs.

```
        {
            if (id != null)
            {
                var role = await _roleManager.FindByIdAsync(id);
                return View(role.Adapt<UpsertRoleViewModel>());
            }

            return View(new UpsertRoleViewModel());
        }
```

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

viewModel	tmodel of type Dotnetidentity.Models.ViewModels.UpsertRoleViewModel

Returns

redirect to controller action Roles

Definition at line 333 of file AdminController.cs.

```
        {
            if (ModelState.IsValid)
            {
                var isUpdate = viewModel.Id != null;

                var role = isUpdate ? await _roleManager.FindByIdAsync(viewModel.Id) : new AppRole() { Name = viewModel.Name, CreatedOn = DateTime.Now };

                if (isUpdate)
                {
                    role.Name = viewModel.Name;
                }

                var result = isUpdate ? await _roleManager.UpdateAsync(role) : await _roleManager.CreateAsync(role);
                if (result.Succeeded)
                {
                    if(isUpdate) {
                        _logger.LogInformation("AUDIT: " + User.Identity!.Name + " changed name of " + viewModel.Id + " to " + viewModel.Name);
                    } else {
                        _logger.LogInformation("AUDIT: " + User.Identity!.Name + " created role " + viewModel.Name);
                    }
                    return RedirectToAction("Roles");
                }
                result.Errors.ToList().ForEach(f => ModelState.AddModelError(string.Empty, f.Description));
                result.Errors.ToList().ForEach(f => TempData["Error"] += f.Description);
            }

            //return View(viewModel);
            //return View("Roles", await _roleManager.Roles.ToListAsync());
            return RedirectToAction("Roles");
        }
```

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

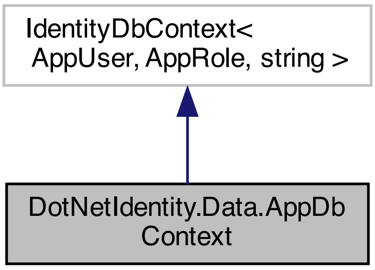
Returns

View of type DotnetIdentity.Views.Admin.Users

6.2 DotNetIdentity.Data.AppDbContext Class Reference

Application Database context class

Inheritance diagram for DotNetIdentity.Data.AppDbContext:


IdentityDbContext< AppUser, AppRole, string >
DotNetIdentity.Data.AppDb Context

Public Member Functions

- AppDbContext (DbContextOptions< AppDbContext > options)
  class constructor

Protected Member Functions

- override void OnModelCreating (ModelBuilder builder)

  Overriding OnModelCreation to seed intitial 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

options	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 intitial data

Parameters

builder	type ModelBuilder

Definition at line 41 of file AppDbContext.cs.

```
        {
            base.OnModelCreating(builder);

            builder.Entity<ApplicationSettings>()
                    .HasKey(x => new { x.Name, x.Type });

            builder.Entity<ApplicationSettings>()
                        .Property(x => x.Value);

            // seeding default settings
            builder.Entity<ApplicationSettings>().HasData(new ApplicationSettings {Name = "ApplicationName", Type = "GlobalSettings", Value = "SecPass"});
            builder.Entity<ApplicationSettings>().HasData(new ApplicationSettings {Name = "SessionTimeoutWarnAfter", Type = "GlobalSettings", Value = "50000"});
            builder.Entity<ApplicationSettings>().HasData(new ApplicationSettings {Name = "SessionTimeoutRedirAfter", Type = "GlobalSettings", Value = "70000"});
            builder.Entity<ApplicationSettings>().HasData(new ApplicationSettings {Name = "SessionCookieExpiration", Type = "GlobalSettings", Value = "7"});
            builder.Entity<ApplicationSettings>().HasData(new ApplicationSettings {Name = "Username", Type = "MailSettings", Value = "YOUR_Smtp_Username"});
            builder.Entity<ApplicationSettings>().HasData(new ApplicationSettings {Name = "Password", Type = "MailSettings", Value = "YOUR_SmtpPassword"});
            builder.Entity<ApplicationSettings>().HasData(new ApplicationSettings {Name = "SmtpServer", Type = "MailSettings", Value = "YOUR_SmtpServer"});
            builder.Entity<ApplicationSettings>().HasData(new ApplicationSettings {Name = "SmtpPort", Type = "MailSettings", Value = "587"});
            builder.Entity<ApplicationSettings>().HasData(new ApplicationSettings {Name = "SmtpUseTls", Type = "MailSettings", Value = "true"});
            builder.Entity<ApplicationSettings>().HasData(new ApplicationSettings {Name = "SmtpFromAddress", Type = "MailSettings", Value = "YOUR_From_Address"});
            builder.Entity<ApplicationSettings>().HasData(new ApplicationSettings {Name = "LdapDomainController", Type = "LdapSettings", Value = "YOUR_Domaincontroller_FQDN"});
            builder.Entity<ApplicationSettings>().HasData(new ApplicationSettings {Name = "LdapDomainName", Type = "LdapSettings", Value = "YOUR_Domainname"});
            builder.Entity<ApplicationSettings>().HasData(new ApplicationSettings {Name = "LdapBaseDn", Type = "LdapSettings", Value = "DC=YOUR,DC=Domain,DC=com"});
            builder.Entity<ApplicationSettings>().HasData(new ApplicationSettings {Name = "LdapGroup", Type = "LdapSettings", Value = "YOUR_Ldap_Group"});
            builder.Entity<ApplicationSettings>().HasData(new ApplicationSettings {Name = "LdapEnabled", Type = "LdapSettings", Value = "true"});

            //Seeding roles to AspNetRoles table
            builder.Entity<AppRole>().HasData(new AppRole { Id = "dffc6dd5-b145-41e9-a861-c87ff673e9ca", Name = "Admin", NormalizedName = "ADMIN".ToUpper() });
            builder.Entity<AppRole>().HasData(new AppRole { Id = "f8a527ac-d7f6-4d9d-aca6-46b2261b042b", Name = "User", NormalizedName = "USER".ToUpper() });
            builder.Entity<AppRole>().HasData(new AppRole { Id = "g7a527ac-d7t6-4d7z-aca6-45t2261b042b", Name = "Editor", NormalizedName = "EDITOR".ToUpper() });
            builder.Entity<AppRole>().HasData(new AppRole { Id = "p9a527ac-d77w-4d3r-aca6-35b2261b042b", Name = "Moderator", NormalizedName = "MODERATOR".ToUpper() });



            //a hasher to hash the password before seeding the user to the db
            var hasher = new PasswordHasher<AppUser>();

            //Seeding the Admin User to AspNetUsers table
            builder.Entity<AppUser>().HasData(
                new AppUser
                {
                    Id = "6fbfb682-568c-4f5b-a298-85937ca4f7f3", // primary key
                    UserName = "super.admin",
                    NormalizedUserName = "SUPER.ADMIN",
                    PasswordHash = hasher.HashPassword(null!, "Test1000!"),
                    Email = "super.admin@local.app",
                    NormalizedEmail = "SUPER.ADMIN@LOCAL.APP",
                    EmailConfirmed = true,
                    FirstName = "Super",
                    LastName = "Admin",
                    IsMfaForce = false,
                    IsLdapLogin = false,
                    IsEnabled = true
                }
            );

            //Seeding the relation between our user and role to AspNetUserRoles table
            builder.Entity<IdentityUserRole<string»().HasData(
                new IdentityUserRole<string>
                {
                    RoleId = "dffc6dd5-b145-41e9-a861-c87ff673e9ca",
                    UserId = "6fbfb682-568c-4f5b-a298-85937ca4f7f3"
                }
            );
        }
```

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.AppSe

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

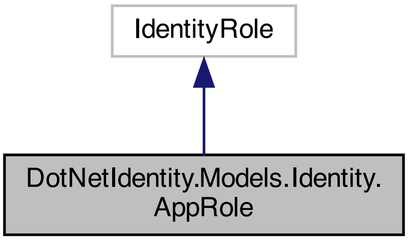
Definition at line 40 of file AppLogs.cs.

```
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

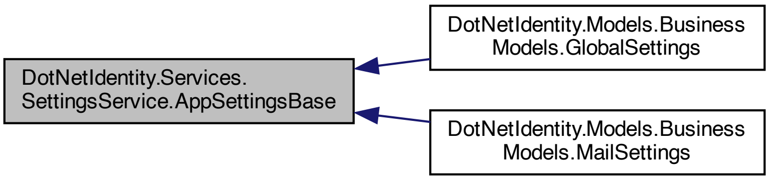
Definition at line 14 of file AppRole.cs.

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

6.6 DotNetIdentity.Services.SettingsService.AppSettingsBase Class Reference

base class for Settings classes

Inheritance diagram for DotNetIdentity.Services.SettingsService.AppSettingsBase:


DotNetIdentity.Services.
SettingsService.AppSettingsBase
DotNetIdentity.Models.Business
Models.GlobalSettings
DotNetIdentity.Models.Business
Models.MailSettings

Public Member Functions

- AppSettingsBase ()

  class constructor
- virtual void Load (AppDbContext unitOfWork)

  load method
- virtual async Task Save (AppDbContext unitOfWork)

  save method

6.6.1 Detailed Description

base class for Settings classes

Definition at line 10 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 24 of file AppSettingsBase.cs.

```
        {
            var type = this.GetType();
            _name = type.Name;
            // 2
            _properties = type.GetProperties();
        }
```

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

unitOfWork	type DbContext

Definition at line 36 of file AppSettingsBase.cs.

```
        {
            // ARGUMENT CHECKING SKIPPED FOR BREVITY
            // 3 get settings for this type name

            var settings = unitOfWork.AppSettings!.Where(w => w.Type == _name).ToList();

            foreach (var propertyInfo in _properties)
            {
                // get the setting from the settings list
                var setting = settings.SingleOrDefault(s => s.Name == propertyInfo.Name);
                if (setting != null)
                {
                    // 4 assign the setting values to the properties in the type inheriting this class
                    propertyInfo.SetValue(this, Convert.ChangeType(setting.Value,
 propertyInfo.PropertyType));
                }
            }
        }
```

References DotNetIdentity.Data.AppDbContext.AppSettings.

6.6.3.2 Save()

```
virtual async Task DotNetIdentity.Services.SettingsService.AppSettingsBase.Save (
            AppDbContext unitOfWork )  [inline], [virtual]
```

save method

Parameters

unitOfWork	type DbContext

Definition at line 59 of file AppSettingsBase.cs.

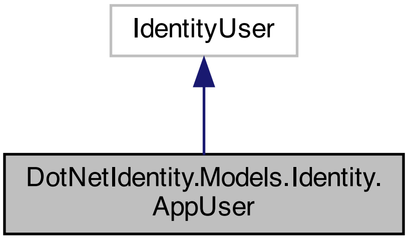
```
        {
            // 5 load existing settings for this type
            var settings = unitOfWork.AppSettings!.Where(w => w.Type == _name).ToList();

            foreach (var propertyInfo in _properties)
            {
                object propertyValue = propertyInfo.GetValue(this, null)!;
                string value = (propertyValue == null) ? null! : propertyValue.ToString()!;

                var setting = settings.SingleOrDefault(s => s.Name == propertyInfo.Name);
                if (setting != null)
                {
                    // 6 update existing value
                    setting.Value = value;
                }
                else
                {
                    // 7 create new setting
                    var newSetting = new ApplicationSettings()
                    {
                        Name = propertyInfo.Name,
                        Type = _name,
                        Value = value,
                    };
                    await unitOfWork.AppSettings!.AddAsync(newSetting);
                }
            }
        }
```

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.

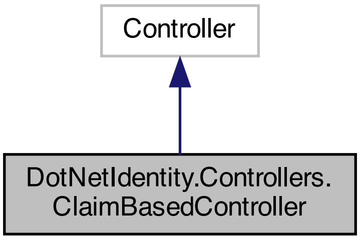
```
00014 { get; set; } = default!;
```

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()

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

Controller action for Software view

Returns

view Software

6.10.2.5 Violence()

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

Controller action for AtLeast18Policy view

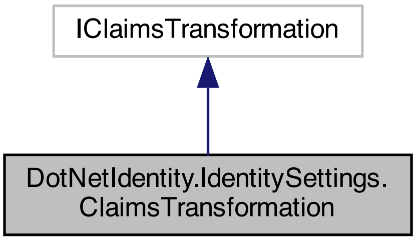
Returns

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 identoty

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

userManager	UserManager
roleManager	RoleManager

Definition at line 27 of file ClaimsTransformation.cs.

```
        {
            _userManager = userManager;
            _roleManager = roleManager;
        }
```

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

principal	ClaimsPrincipal

Returns

ClaimsPrincipal

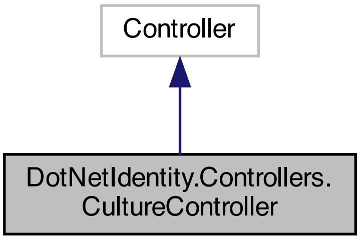
Definition at line 38 of file ClaimsTransformation.cs.

```
        {
            var identity = principal.Identity as ClaimsIdentity;
            var user = await _userManager.FindByNameAsync(identity?.Name);
            if (user != null)
            {
                if (!principal.HasClaim(c => c.Type == ClaimTypes.Gender))
                {
                    var genderClaim = new Claim(ClaimTypes.Gender, Enum.GetName(user.Gender)!);
                    identity?.AddClaim(genderClaim);
                }

                if (!principal.HasClaim(c => c.Type == ClaimTypes.DateOfBirth))
                {
                    var birthDayClaim = new Claim(ClaimTypes.DateOfBirth, user.BirthDay.ToShortDateString());
                    identity?.AddClaim(birthDayClaim);
                }

                if (!principal.HasClaim(c => c.Type == "FreeTrial"))
                {
                    var freeTrialClaim = new Claim("FreeTrial", user.CreatedOn.ToShortDateString());
                    identity?.AddClaim(freeTrialClaim);
                }

                if (!principal.HasClaim(c => c.Type == "Department") && !String.IsNullOrEmpty(user.Department))
                {
                    var departmentClaim = new Claim("Department", user.Department!);
                    identity?.AddClaim(departmentClaim);
                }

                if (!principal.HasClaim(c => c.Type == ClaimTypes.GivenName))
                {
                    var givennameClaim = new Claim(ClaimTypes.GivenName, user.FirstName!);
                    identity?.AddClaim(givennameClaim);
                }

                if (!principal.HasClaim(c => c.Type == ClaimTypes.Surname))
                {
                    var surnameClaim = new Claim(ClaimTypes.Surname, user.LastName!);
                    identity?.AddClaim(surnameClaim);
                }

                if(!principal.HasClaim(c=>c.Type=="SelectedCulture")) {
                    var culture = System.Globalization.CultureInfo.CurrentCulture.ToString().ToLower();
                    var cultureClaim = new Claim("SelectedCulture", culture);
                    identity?.AddClaim(cultureClaim);
                }

                if (!principal.HasClaim(c => c.Type == "RolesCombined"))
                {
                    var rls = await _userManager.GetRolesAsync(user);
                    if (rls != null)
                    {
                        var rlsComb = "";
                        if(rls.Count()>0 && rls.Count()<=1) {
                            rlsComb = rls[0].ToString();
                        } else {
                            for(int i=0;i<rls.Count;i++)
                            {
                                if(i+1!=rls.Count){ rlsComb += rls[i] + ","; }
                                else { rlsComb += rls[i]; }
                            }
                        }
                        if(rlsComb.Length>0)
                        {
                            var rolesCombinedClaim = new Claim("RolesCombined", rlsComb!);
                            identity?.AddClaim(rolesCombinedClaim);
                        }
                    }
                }
            }

            return principal;
        }
```

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

culture	the culture as string
returnUrl	the return url as string

Returns

a local redirect to variable returnUrl

Definition at line 26 of file CultureController.cs.

```
        {
            Response.Cookies.Append(
                CookieRequestCultureProvider.DefaultCookieName,
                CookieRequestCultureProvider.MakeCookieValue(new RequestCulture(culture)),
                new CookieOptions { Expires = DateTimeOffset.UtcNow.AddYears(1) }
            );
            return LocalRedirect(returnUrl);
        }
```

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

sett	type ISettingsService

Definition at line 23 of file EmailHelper.cs.

```
        {
            _sett = sett;
        }
```

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

model	type EmailModel

Returns

Definition at line 33 of file EmailHelper.cs.

```
        {
            var mail = new MailMessage
            {
                Subject = model.Subject,
                Body = model.Body,
                From = new MailAddress(_sett.Mail.SmtpFromAddress!),
                IsBodyHtml = true
            };

            var smtp = new SmtpClient
            {
                Host = _sett.Mail.SmtpServer!,
                Port = Convert.ToInt32(_sett.Mail.SmtpPort),
                EnableSsl = Convert.ToBoolean(_sett.Mail.SmtpUseTls),
                DeliveryMethod = SmtpDeliveryMethod.Network,
                UseDefaultCredentials = false,
                Credentials = new NetworkCredential(_sett.Mail.Username, _sett.Mail.Password)
            };

            mail.To.Add(model.To);

            await smtp.SendMailAsync(mail);
        }
```

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.

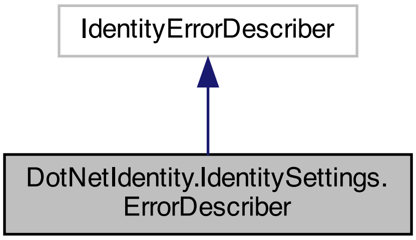
```
00022 { get; set; } = default!;
```

Referenced by DotNetIdentity.Helpers.EmailHelper.SendAsync().

6.16 DotNetIdentity.IdentitySettings.ErrorDescriber Class Reference

class to override IdentityErrorDescriber IdentotyError 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 IdentotyError 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

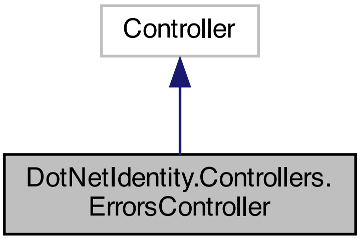
IdentityError

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

statusCode	

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 = "Acess 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                     break;
00074                 // server error
00075                 case 500:
00076                     md.ErrorMessage = "Sorry something went wrong on the server";
00077                     break;
00078                 // service unavailable
00079                 case 503:
00080                     md.ErrorMessage = "Sorry, the service is currently unavailable.";
00081                     break;
00082                 default:
00083                     md.ErrorMessage = "Sorry, there was an error.";
00084                     break;
00085             }
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<IExceptionHandlerPathFeature>();
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 rquest id
- bool ShowRequestId `[get]`

  show the requst 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 rquest 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 requst 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.

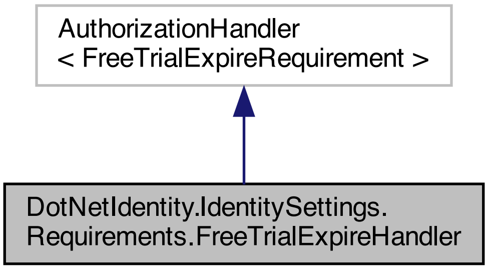
```
00012 { get; set; } = default!;
```

Referenced by DotNetIdentity.Controllers.UserController.ForgotPassword().

6.20 DotNetIdentity.IdentitySettings.Requirements.FreeTrialExpire↩ Handler Class Reference

class to define own Identity requirements type of FreeTrialExpireRequirement

Inheritance diagram for DotNetIdentity.IdentitySettings.Requirements.FreeTrialExpireHandler:


AuthorizationHandler
< FreeTrialExpireRequirement >
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

context	AuthorizationHandlerContext
requirement	FreeTrialExpireRequirement

Returns

Task

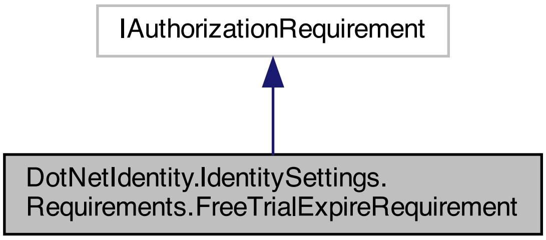
Definition at line 21 of file FreeTrialExpireRequirement.cs.

```
        {
            var freeTrialClaim = context.User.FindFirst(f => f.Type == "FreeTrial");
            if (freeTrialClaim != null)
            {
                var freeTrialExpires = Convert.ToDateTime(freeTrialClaim.Value).AddDays(30);
                if (DateTime.Now < freeTrialExpires)
                {
                    context.Succeed(requirement);
                }
                else
                {
                    context.Fail();
                }
            }

            return Task.CompletedTask;
        }
```

6.21 DotNetIdentity.IdentitySettings.Requirements.FreeTrialExpireRequirement Class Reference

class to define own Identity requirements type of IAuthorizationRequirement

Inheritance diagram for DotNetIdentity.IdentitySettings.Requirements.FreeTrialExpireRequirement:


IAuthorizationRequirement
DotNetIdentity.IdentitySettings.
Requirements.FreeTrialExpireRequirement

6.21.1 Detailed Description

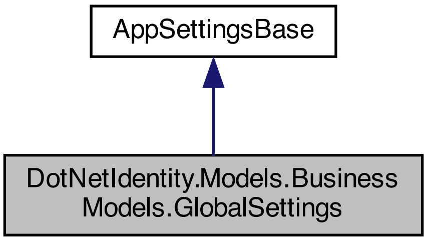
class to define own Identity requirements type of IAuthorizationRequirement

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;}
```

Referenced by DotNetIdentity.Controllers.SettingsController.SettingsMail().

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;}
```

Referenced by DotNetIdentity.Controllers.SettingsController.SettingsMail().

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;}
```

Referenced by DotNetIdentity.Controllers.SettingsController.SettingsMail().

6.22.2.4 SessionTimeoutWarnAfter

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

property SessionTimeoutWarnAfter

string

Definition at line 17 of file GlobalSettings.cs.

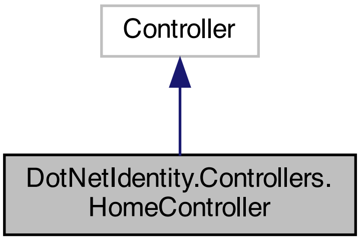
```
00017 {get; set;}
```

Referenced by DotNetIdentity.Controllers.SettingsController.SettingsMail().

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

userManager	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

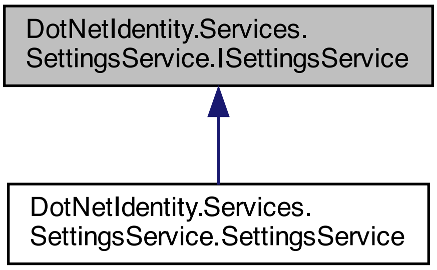
Returns

view Index

6.24 DotNetIdentity.Services.SettingsService.ISettingsService Interface Reference

Interface class for SettingsService

Inheritance diagram for DotNetIdentity.Services.SettingsService.ISettingsService:


DotNetIdentity.Services.
SettingsService.ISettingsService
DotNetIdentity.Services.
SettingsService.SettingsService

Public Member Functions

- Task Save ()

  save method

Properties

- GlobalSettings Global `[get]`

  property global settings
- MailSettings Mail `[get]`

  property mail settings
- LdapSettings Ldap `[get]`

  property mail settings

6.24.1 Detailed Description

Interface class for SettingsService

Definition at line 8 of file ISettingsService.cs.

6.24.2 Member Function Documentation

6.24.2.1 Save()

```
Task 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 13 of file ISettingsService.cs.

```
00013 { get; }
```

6.24.3.2 Ldap

```
LdapSettings DotNetIdentity.Services.SettingsService.ISettingsService.Ldap  [get]
```

property mail settings

MailSettings

Implemented in DotNetIdentity.Services.SettingsService.SettingsService.

Definition at line 23 of file ISettingsService.cs.

```
00023 { get; }
```

6.24.3.3 Mail

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

property mail settings

MailSettings

Implemented in DotNetIdentity.Services.SettingsService.SettingsService.

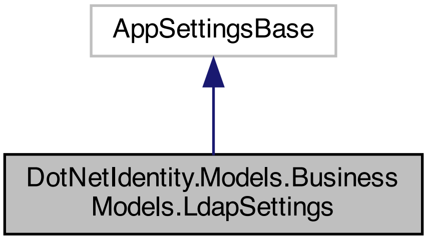
Definition at line 18 of file ISettingsService.cs.

```
00018 { get; }
```

6.25 DotNetIdentity.Models.BusinessModels.LdapSettings Class Reference

class to define ldap settings

Inheritance diagram for DotNetIdentity.Models.BusinessModels.LdapSettings:



Properties

- string? LdapDomainController `[get, set]`

  domain controller property
- string? LdapDomainName `[get, set]`

  domain name property
- string? LdapBaseDn `[get, set]`

  domain base dn property
- string? LdapGroup `[get, set]`

  domain group name property
- bool LdapEnabled `[get, set]`

  ldap enabled property

6.25.1 Detailed Description

class to define ldap settings

Definition at line 5 of file LdapSettings.cs.

6.25.2 Property Documentation

6.25.2.1 LdapBaseDn

```
string? DotNetIdentity.Models.BusinessModels.LdapSettings.LdapBaseDn [get], [set]
```

domain base dn property

string

Definition at line 20 of file LdapSettings.cs.

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

Referenced by DotNetIdentity.Controllers.SettingsController.SettingsLdap().

6.25.2.2 LdapDomainController

```
string? DotNetIdentity.Models.BusinessModels.LdapSettings.LdapDomainController [get], [set]
```

domain controller property

string

Definition at line 10 of file LdapSettings.cs.

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

6.25.2.3 LdapDomainName

```
string? DotNetIdentity.Models.BusinessModels.LdapSettings.LdapDomainName [get], [set]
```

domain name property

string

Definition at line 15 of file LdapSettings.cs.

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

Referenced by DotNetIdentity.Controllers.SettingsController.SettingsLdap().

6.25.2.4 LdapEnabled

```
bool DotNetIdentity.Models.BusinessModels.LdapSettings.LdapEnabled [get], [set]
```

ldap enabled property

bool

Definition at line 30 of file LdapSettings.cs.

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

Referenced by DotNetIdentity.Controllers.SettingsController.SettingsLdap().

6.25.2.5 LdapGroup

```
string? DotNetIdentity.Models.BusinessModels.LdapSettings.LdapGroup [get], [set]
```

domain group name property

string

Definition at line 25 of file LdapSettings.cs.

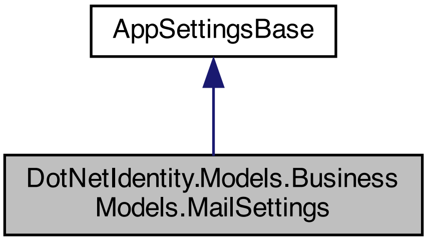
```
00025 {get;set;}
```

Referenced by DotNetIdentity.Controllers.SettingsController.SettingsLdap().

6.26 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

6.26.1 Detailed Description

class to define a ail settings object

Definition at line 7 of file MailSettings.cs.

6.26.2 Property Documentation

6.26.2.1 Password

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

property Password

string

Definition at line 17 of file MailSettings.cs.

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

Referenced by DotNetIdentity.Controllers.SettingsController.SettingsMail().

6.26.2.2 SmtpFromAddress

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

property SmtpFromAddress

string

Definition at line 32 of file MailSettings.cs.

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

Referenced by DotNetIdentity.Controllers.SettingsController.SettingsMail().

6.26.2.3 SmtpPort

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

property SmtpPort

string

Definition at line 27 of file MailSettings.cs.

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

Referenced by DotNetIdentity.Controllers.SettingsController.SettingsMail().

6.26.2.4 SmtpServer

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

property SmtpServer

string

Definition at line 22 of file MailSettings.cs.

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

Referenced by DotNetIdentity.Controllers.SettingsController.SettingsMail().

6.26.2.5 SmtpUseTls

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

property SmtpUseTls

bool

Definition at line 37 of file MailSettings.cs.

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

Referenced by DotNetIdentity.Controllers.SettingsController.SettingsMail().

6.26.2.6 Username

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

property Username

string

Definition at line 12 of file MailSettings.cs.

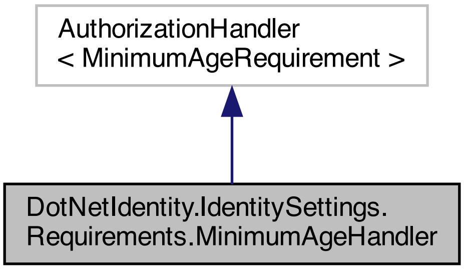
```
00012 { get; set; }
```

Referenced by DotNetIdentity.Controllers.SettingsController.SettingsMail().

6.27 DotNetIdentity.IdentitySettings.Requirements.MinimumAgeHandler Class Reference

class implementing AuthorizationHandler of type MinimumAgeRequirement

Inheritance diagram for DotNetIdentity.IdentitySettings.Requirements.MinimumAgeHandler:


AuthorizationHandler
< MinimumAgeRequirement >
DotNetIdentity.IdentitySettings.
Requirements.MinimumAgeHandler

Protected Member Functions

- override Task HandleRequirementAsync (AuthorizationHandlerContext context, MinimumAgeRequirement requirement)

  overriding HandleRequirementAsync

6.27.1 Detailed Description

class implementing AuthorizationHandler of type MinimumAgeRequirement

Definition at line 30 of file MinimumAgeRequirement.cs.

6.27.2 Member Function Documentation

6.27.2.1 HandleRequirementAsync()

```
override Task DotNetIdentity.IdentitySettings.Requirements.MinimumAgeHandler.HandleRequirementAsync (
              AuthorizationHandlerContext context,
              MinimumAgeRequirement requirement )  [inline], [protected]
```

overriding HandleRequirementAsync

Parameters

context	AuthorizationHandlerContext
requirement	MinimumAgeRequirement

Returns

Task

Definition at line 38 of file MinimumAgeRequirement.cs.

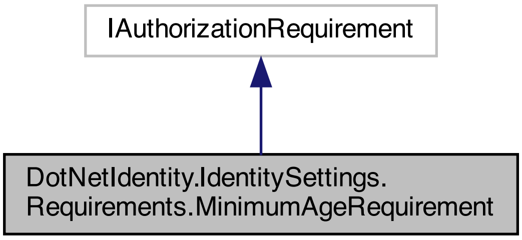
```
        {
            var birthDayClaim = context.User.FindFirst(f => f.Type == ClaimTypes.DateOfBirth);
            if (birthDayClaim != null)
            {
                var birthDay = Convert.ToDateTime(birthDayClaim.Value);
                var calculatedAge = DateTime.Now.Year - birthDay.Year;
                if (calculatedAge >= requirement.MinimumAge)
                {
                    context.Succeed(requirement);
                }
                else
                {
                    context.Fail();
                }
            }

            return Task.CompletedTask;
        }
```

References DotNetIdentity.IdentitySettings.Requirements.MinimumAgeRequirement.MinimumAge.

6.28 DotNetIdentity.IdentitySettings.Requirements.MinimumAge↩ Requirement Class Reference

class to define own Identity requirements type of IAuthorizationRequirement

Inheritance diagram for DotNetIdentity.IdentitySettings.Requirements.MinimumAgeRequirement:


IAuthorizationRequirement
DotNetIdentity.IdentitySettings.
Requirements.MinimumAgeRequirement

Public Member Functions

- MinimumAgeRequirement (int minimumAge)

    class constructor

Properties

- int MinimumAge `[get, set]`

    property MinimumAge

6.28.1 Detailed Description

class to define own Identity requirements type of IAuthorizationRequirement

Definition at line 9 of file MinimumAgeRequirement.cs.

6.28.2 Constructor & Destructor Documentation

6.28.2.1 MinimumAgeRequirement()

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

class constructor

Parameters

minimumAge	

Definition at line 15 of file MinimumAgeRequirement.cs.

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

References DotNetIdentity.IdentitySettings.Requirements.MinimumAgeRequirement.MinimumAge.

6.28.3 Property Documentation

6.28.3.1 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().

6.29 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

6.29.1 Detailed Description

model for NewUser view

Definition at line 9 of file NewUserModel.cs.

6.29.2 Property Documentation

6.29.2.1 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().

6.29.2.2 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().

6.29.2.3 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().

6.29.2.4 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().

6.29.2.5 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().

6.29.2.6 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().

6.29.2.7 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().

6.29.2.8 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().

6.29.2.9 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().

6.29.2.10 UserName

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

the username

string

Definition at line 15 of file NewUserModel.cs.

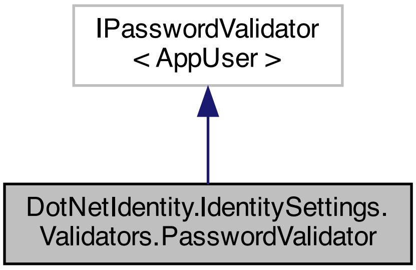
```
00015 { get; set; } = default!;
```

Referenced by DotNetIdentity.Controllers.AdminController.NewUser().

6.30 DotNetIdentity.IdentitySettings.Validators.PasswordValidator Class Reference

class to validate a password

Inheritance diagram for DotNetIdentity.IdentitySettings.Validators.PasswordValidator:


IPasswordValidator
< AppUser >
DotNetIdentity.IdentitySettings.
Validators.PasswordValidator

Public Member Functions

- Task< IdentityResult > ValidateAsync (UserManager< AppUser > manager, AppUser user, string password)

  Task to validate a password

6.30.1 Detailed Description

class to validate a password

Definition at line 9 of file PasswordValidator.cs.

6.30.2 Member Function Documentation

6.30.2.1 ValidateAsync()

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

Task to validate a password

Parameters

manager	UserManager
user	AppUser
password	string

Returns

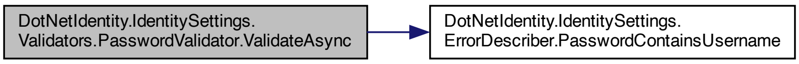
Task of type IdentityResult

Definition at line 18 of file PasswordValidator.cs.

```
        {
            var errors = new List<IdentityError>();

            if (user.UserName == password)
            {
                errors.Add(ErrorDescriber.PasswordContainsUsername());
            }

            if (errors.Any())
            {
                return Task.FromResult(IdentityResult.Failed(errors.ToArray()));
            }
            return Task.FromResult(IdentityResult.Success);
        }
```

References DotNetIdentity.IdentitySettings.ErrorDescriber.PasswordContainsUsername().

Here is the call graph for this function:


DotNetIdentity.IdentitySettings.
Validators.PasswordValidator.ValidateAsync
DotNetIdentity.IdentitySettings.
ErrorDescriber.PasswordContainsUsername

6.31 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

6.31.1 Detailed Description

model for ResetPassword view

Definition at line 6 of file ResetPasswordViewModel.cs.

6.31.2 Property Documentation

6.31.2.1 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().

6.31.2.2 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().

6.31.2.3 UserId

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

the user id

string

Definition at line 12 of file ResetPasswordViewModel.cs.

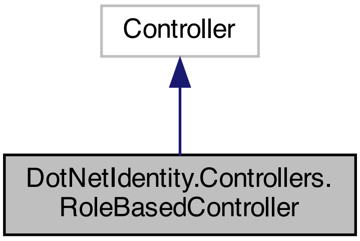
```
00012 { get; set; } = default!;
```

Referenced by DotNetIdentity.Controllers.UserController.ResetPassword().

6.32 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

6.32.1 Detailed Description

Controller to test RoleBased Authorization

Definition at line 9 of file RoleBasedController.cs.

6.32.2 Member Function Documentation

6.32.2.1 Articles()

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

Controller Action for Articles view

Returns

6.32.2.2 Dashboard()

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

Controller Action for Dashboard view

Returns

6.32.2.3 Profile()

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

Controller Action for Profile view

Returns

6.32.2.4 Users()

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

Controller Action for Users view

Returns

6.33 DotNetIdentity.Models.DataModels.SessionCache Class Reference

class to define a SessionCache object

Properties

- string? id `[get, set]`

  proerty 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

6.33.1 Detailed Description

class to define a SessionCache object

Definition at line 11 of file SessionCache.cs.

6.33.2 Property Documentation

6.33.2.1 AbsoluteExpiration

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

property AbsoluteExpiration

DateTime

Definition at line 37 of file SessionCache.cs.

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

6.33.2.2 ExpiresAtTime

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

property ExpiresAtTime

DateTime

Definition at line 27 of file SessionCache.cs.

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

6.33.2.3 id

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

proerty id

int

Definition at line 17 of file SessionCache.cs.

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

6.33.2.4 SlidingExpirationInSeconds

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

property SlidingExpirationInSeconds

int

Definition at line 32 of file SessionCache.cs.

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

6.33.2.5 Value

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

property value

byte[]

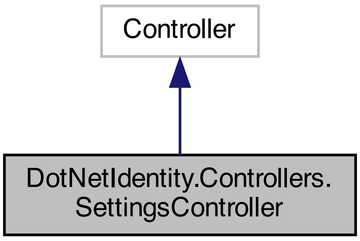
Definition at line 22 of file SessionCache.cs.

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

6.34 DotNetIdentity.Controllers.SettingsController Class Reference

controller class for settings views

Inheritance diagram for DotNetIdentity.Controllers.SettingsController:



Public Member Functions

- SettingsController (ILogger< SettingsController > log, ISettingsService sett, IStringLocalizer< SettingsController > loc, UserManager< AppUser > usr, RoleManager< AppRole > rol, IWebHostEnvironment env)

    class constructor
- ActionResult SettingsLdap ()

    Controller action for SettingsLdap view
- ActionResult SettingsMail ()

    Controller action for SettingsMail view
- ActionResult SettingsApp ()

    Controller action for SettingsApp view
- async Task< IActionResult > SettingsMail (MailSettings viewModel)

    controller POST method to save Mail settings
- async Task< IActionResult > SettingsMail (GlobalSettings viewModel)

    controller POST method to save Application settings
- async Task< IActionResult > SettingsLdap (LdapSettings viewModel)

    controller POST method to save ldap settings

6.34.1 Detailed Description

controller class for settings views

Definition at line 20 of file SettingsController.cs.

6.34.2 Constructor & Destructor Documentation

6.34.2.1 SettingsController()

```
DotNetIdentity.Controllers.SettingsController.SettingsController (
            ILogger< SettingsController > log,
            ISettingsService sett,
            IStringLocalizer< SettingsController > loc,
            UserManager< AppUser > usr,
            RoleManager< AppRole > rol,
            IWebHostEnvironment env )  [inline]
```

class constructor

Parameters

sett	ISettingsService
loc	IStringLocalizer
usr	UserManager
rol	RoleManager
log	ILogger

///

Parameters

env	IWebHostEnvironment

Definition at line 56 of file SettingsController.cs.

```
00056

                {
00057                   _hostEnvironment = env;
00058                   _userManager = usr;
00059                   _roleManager = rol;
00060                   _localizer = loc;
00061                   _settings = sett;
00062                   _logger = log;
00063           }
```

6.34.3 Member Function Documentation

6.34.3.1 SettingsApp()

`ActionResult DotNetIdentity.Controllers.SettingsController.SettingsApp ( )  [inline]`

Controller action for SettingsApp view

Returns

view SettingsApp

Definition at line 89 of file SettingsController.cs.

```
00089                                               {
00090                   var viewModel = new GlobalSettings();
00091                   viewModel = _settings.Global;
00092                   return View(viewModel);
00093           }
```

6.34.3.2 SettingsLdap() [1/2]

`ActionResult DotNetIdentity.Controllers.SettingsController.SettingsLdap ( )  [inline]`

Controller action for SettingsLdap view

Returns

view SettingsLdap

Definition at line 69 of file SettingsController.cs.

```
00069                                                {
00070                   var viewModel = new LdapSettings();
00071                   viewModel = _settings.Ldap;
00072                   return View(viewModel);
00073           }
```

6.34.3.3 SettingsLdap() [2/2]

```
async Task< IActionResult > DotNetIdentity.Controllers.SettingsController.SettingsLdap (
                LdapSettings viewModel )  [inline]
```

controller POST method to save ldap settings

Parameters

viewModel	LdapSettings

Returns

view

Definition at line 142 of file SettingsController.cs.

```
00142                                                                               {
00143
00144                   _settings.Ldap.LdapDomainName = viewModel.LdapDomainName;
00145                   _settings.Ldap.LdapDomainController = viewModel.LdapDomainName;
00146                   _settings.Ldap.LdapBaseDn = viewModel.LdapBaseDn;
00147                   _settings.Ldap.LdapEnabled = viewModel.LdapEnabled;
00148                   _settings.Ldap.LdapGroup = viewModel.LdapGroup;
00149                   await _settings.Save();
00150                   _logger.LogInformation("AUDIT: " + User.Identity!.Name + " modified Ldap settings! ");
00151                   ViewData["message"] = "Settings successfully saved!";
00152                   return View();
00153           }
```

References DotNetIdentity.Models.BusinessModels.LdapSettings.LdapBaseDn, DotNetIdentity.Models.BusinessModels.LdapSettings DotNetIdentity.Models.BusinessModels.LdapSettings.LdapEnabled, and DotNetIdentity.Models.BusinessModels.LdapSettings.LdapG

6.34.3.4 SettingsMail() [1/3]

```
ActionResult DotNetIdentity.Controllers.SettingsController.SettingsMail ( )  [inline]
```

Controller action for SettingsMail view

Returns

view SettingsMail

Definition at line 79 of file SettingsController.cs.

```
00079                                                      {
00080               var viewModel = new MailSettings();
00081               viewModel = _settings.Mail;
00082               return View(viewModel);
00083           }
```

6.34.3.5 SettingsMail() [2/3]

```
async Task< IActionResult > DotNetIdentity.Controllers.SettingsController.SettingsMail (
              GlobalSettings viewModel )  [inline]
```

controller POST method to save Application settings

Parameters

viewModel	GlobalSettings

Returns

view

Definition at line 123 of file SettingsController.cs.

```
00123                                                                                  {
00124 
00125                 _settings.Global.ApplicationName = viewModel.ApplicationName;
00126                 _settings.Global.SessionCookieExpiration = viewModel.SessionCookieExpiration;
00127                 _settings.Global.SessionTimeoutWarnAfter = viewModel.SessionTimeoutWarnAfter;
00128                 _settings.Global.SessionTimeoutRedirAfter = viewModel.SessionTimeoutRedirAfter;
00129                 await _settings.Save();
00130                 _logger.LogInformation("AUDIT: " + User.Identity!.Name + " modified Application
       settings! ");
00131                 ViewData["message"] = "Settings successfully saved!";
00132                 return View();
00133         }
```

References DotNetIdentity.Models.BusinessModels.GlobalSettings.ApplicationName, DotNetIdentity.Models.BusinessModels.GlobalS
DotNetIdentity.Models.BusinessModels.GlobalSettings.SessionTimeoutRedirAfter, and DotNetIdentity.Models.BusinessModels.Globa

6.34.3.6 SettingsMail() [3/3]

```
async Task< IActionResult > DotNetIdentity.Controllers.SettingsController.SettingsMail (
              MailSettings viewModel )  [inline]
```

controller POST method to save Mail settings

Parameters

viewModel	MailSettings

Returns

view

Definition at line 102 of file SettingsController.cs.

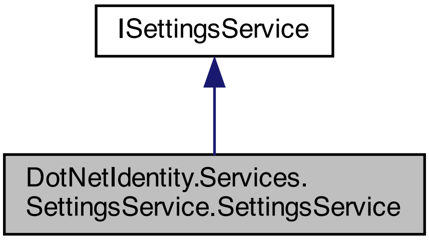
```
00102                                                                                {
00103 
00104                 _settings.Mail.SmtpServer = viewModel.SmtpServer;
00105                 _settings.Mail.SmtpFromAddress = viewModel.SmtpFromAddress;
00106                 _settings.Mail.SmtpPort = viewModel.SmtpPort;
00107                 _settings.Mail.SmtpUseTls = viewModel.SmtpUseTls;
00108                 _settings.Mail.Username = viewModel.Username;
00109                 _settings.Mail.Password = viewModel.Password;
00110                 await _settings.Save();
00111                 _logger.LogInformation("AUDIT: " + User.Identity!.Name + " modified Mail settings! ");
00112                 ViewData["message"] = "Settings successfully saved!";
00113                 return View();
00114         }
```

References DotNetIdentity.Models.BusinessModels.MailSettings.Password, DotNetIdentity.Models.BusinessModels.MailSettings.Smt
DotNetIdentity.Models.BusinessModels.MailSettings.SmtpPort, DotNetIdentity.Models.BusinessModels.MailSettings.SmtpServer,
DotNetIdentity.Models.BusinessModels.MailSettings.SmtpUseTls, and DotNetIdentity.Models.BusinessModels.MailSettings.Usernam

6.35 DotNetIdentity.Services.SettingsService.SettingsService Class Reference

class for settings service

Inheritance diagram for DotNetIdentity.Services.SettingsService.SettingsService:



Public Member Functions

- SettingsService (AppDbContext unitOfWork)

  class constructor
- async Task Save ()

  save method

Properties

- LdapSettings Ldap `[get]`

  global settings property
- GlobalSettings Global `[get]`

  global settings property
- MailSettings Mail `[get]`

  mail settzings property

6.35.1 Detailed Description

class for settings service

Definition at line 8 of file SettingsService.cs.

6.35.2 Constructor & Destructor Documentation

6.35.2.1 SettingsService()

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

class constructor

Parameters

unitOfWork	

Definition at line 45 of file SettingsService.cs.

```
{
    // ARGUMENT CHECKING SKIPPED FOR BREVITY
    _unitOfWork = unitOfWork;
    // 3
    _globalSettings = new Lazy<GlobalSettings>(CreateSettings<GlobalSettings>);
    _mailSettings = new Lazy<MailSettings>(CreateSettings<MailSettings>);
    _ldapSettings = new Lazy<LdapSettings>(CreateSettings<LdapSettings>);
}
```

6.35.3 Member Function Documentation

6.35.3.1 Save()

`async Task DotNetIdentity.Services.SettingsService.SettingsService.Save ( )  [inline]`

save method

Returns

Implements DotNetIdentity.Services.SettingsService.ISettingsService.

Definition at line 59 of file SettingsService.cs.

```
{
    // only save changes to settings that have been loaded
    if (_globalSettings.IsValueCreated)
        await _globalSettings.Value.Save(_unitOfWork);

    if (_mailSettings.IsValueCreated)
        await _mailSettings.Value.Save(_unitOfWork);

    if (_ldapSettings.IsValueCreated)
        await _ldapSettings.Value.Save(_unitOfWork);

    await _unitOfWork.SaveChangesAsync();
}
```

6.35.4 Property Documentation

6.35.4.1 Global

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

global settings property

Implements DotNetIdentity.Services.SettingsService.ISettingsService.

Definition at line 26 of file SettingsService.cs.

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

6.35.4.2 Ldap

```
LdapSettings DotNetIdentity.Services.SettingsService.SettingsService.Ldap  [get]
```

global settings property

Implements DotNetIdentity.Services.SettingsService.ISettingsService.

Definition at line 17 of file SettingsService.cs.

```
00017 { get { return _ldapSettings.Value; } }
```

6.35.4.3 Mail

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

mail settzings property

Implements DotNetIdentity.Services.SettingsService.ISettingsService.

Definition at line 35 of file SettingsService.cs.

```
00035 { get { return _mailSettings.Value; } }
```

6.36 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

6.36.1 Detailed Description

model for Login view

Definition at line 6 of file SignInViewModel.cs.

6.36.2 Property Documentation

6.36.2.1 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().

6.36.2.2 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().

6.36.2.3 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().

6.37 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

6.37.1 Detailed Description

model for Register view

Definition at line 6 of file SignUpViewModel.cs.

6.37.2 Property Documentation

6.37.2.1 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().

6.37.2.2 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().

6.37.2.3 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().

6.37.2.4 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().

6.37.2.5 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().

6.37.2.6 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().

6.37.2.7 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().

6.37.2.8 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().

6.38 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

6.38.1 Detailed Description

model for Error Statuscodes

Definition at line 5 of file StatusCodesModel.cs.

6.38.2 Property Documentation

6.38.2.1 ErrorMessage

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

the error message

string

Definition at line 15 of file StatusCodesModel.cs.

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

6.38.2.2 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;}
```

6.38.2.3 StatusCode

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

the status code

int

Definition at line 10 of file StatusCodesModel.cs.

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

6.39 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

6.39.1 Detailed Description

class providing 2fa auth services

Definition at line 8 of file TwoFactorAuthenticationService.cs.

6.39.2 Constructor & Destructor Documentation

6.39.2.1 TwoFactorAuthenticationService()

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

class constructor

Parameters

cnf	IConfiguration
urlEncoder	UrlEncoder
sett	ISettingsService

Definition at line 29 of file TwoFactorAuthenticationService.cs.

```
        {
            _urlEncoder = urlEncoder;
            _configuration = cnf;
            _sett = sett;
        }
```

6.39.3 Member Function Documentation

6.39.3.1 GenerateQrCodeUri()

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

method to generate a barcode for 2fa configuration

Parameters

email	string
authenticatorKey	string

Returns

string

Definition at line 42 of file TwoFactorAuthenticationService.cs.

```
        {
            var encodedUrl = _urlEncoder.Encode(_sett.Global.ApplicationName!);
            var encodedEmail = _urlEncoder.Encode(email);

            return
      $"otpauth://totp/{encodedUrl}:{encodedEmail}?secret={authenticatorKey}&issuer={encodedUrl}&digits=6";
        }
```

6.40 DotNetIdentity.Models.ViewModels.TwoFactorAuthenticatorViewModel 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

6.40.1 Detailed Description

model for TwoFactorAuthenticator view

Definition at line 6 of file TwoFactorAuthenticatorViewModel.cs.

6.40.2 Property Documentation

6.40.2.1 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!;
```

6.40.2.2 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!;
```

6.40.2.3 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

6.41 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

6.41.1 Detailed Description

the model for TwoFactorLogin view

Definition at line 6 of file TwoFactorLoginVieWModel.cs.

6.41.2 Property Documentation

6.41.2.1 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; }
```

6.41.2.2 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().

6.41.2.3 VerificationCode

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

the otp code

string

Definition at line 12 of file TwoFactorLoginVieWModel.cs.

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

Referenced by DotNetIdentity.Controllers.UserController.TwoFactorLogin().

6.42 DotNetIdentity.Models.ViewModels.TwoFactorTypeViewModel Class Reference

model for TwoFactorType view

Properties

- TwoFactorType TwoFactorType `[get, set]`

  the 2fa type

6.42.1 Detailed Description

model for TwoFactorType view

Definition at line 6 of file TwoFactorTypeViewModel.cs.

6.42.2 Property Documentation

6.42.2.1 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().

6.43 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

6.43.1 Detailed Description

model for UpdateProfile view

Definition at line 6 of file UpdateProfileViewModel.cs.

6.43.2 Property Documentation

6.43.2.1 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().

6.43.2.2 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().

6.43.2.3 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().

6.43.2.4 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().

6.43.2.5 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().

6.43.2.6 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().

6.44 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

6.44.1 Detailed Description

model for the UpsertRole view

Definition at line 6 of file UpsertRoleViewModel.cs.

6.44.2 Property Documentation

6.44.2.1 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().

6.44.2.2 Name

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

the role name

string

Definition at line 17 of file UpsertRoleViewModel.cs.

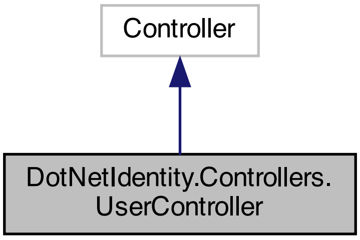
```
00017 { get; set; } = default!;
```

Referenced by DotNetIdentity.Controllers.AdminController.UpsertRole().

6.45 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 (TwoFactorLoginVieWModel vieWModel)

  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 methood 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

6.45.1 Detailed Description

Controller Class for User views

Todo add translation

Definition at line 20 of file UserController.cs.

6.45.2 Constructor & Destructor Documentation

6.45.2.1 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

localizer	type IStringLocalizer
log	type ILogger
webhostEnviroment	type IwebHostEnviroment
userManager	type UserManager
signInManager	type SIgnInManager
twoFactorAuthService	type TwoFactorAuthService
emailHelper	työe EmailHelper

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         }
```

6.45.3 Member Function Documentation

6.45.3.1 ChangePassword() [1/2]

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

controller action for ChangePassword view

Returns

view ChangePassword

6.45.3.2 ChangePassword() [2/2]

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

controller POST method for ChangePassword view

Parameters

viewModel	ChangePasswordViewModel

Returns

redirect to action Index

Definition at line 410 of file UserController.cs.

```
        {
            if (ModelState.IsValid)
            {
                var me = await _userManager.FindByNameAsync(User.Identity?.Name);

                var passwordValid = await _userManager.CheckPasswordAsync(me, viewModel.Password);
                if (passwordValid)
                {
                    var result = await _userManager.ChangePasswordAsync(me, viewModel.Password, viewModel.NewPassword);
                    if (result.Succeeded)
                    {
                        await _userManager.UpdateSecurityStampAsync(me);

                        await _signInManager.SignOutAsync();
                        await _signInManager.SignInAsync(me, true);
                        _logger.LogInformation("AUDIT: " + me.UserName + " successfully changed own password!");

                        return RedirectToAction("Index", "Admin");
                    }
                    _logger.LogWarning("AUDIT: " + me.UserName + "tried to change password. Error: " + result.Errors.FirstOrDefault()!.ToString());
                    result.Errors.ToList().ForEach(f => ModelState.AddModelError(string.Empty, f.Description));
                }
                else
                {
                    _logger.LogWarning("AUDIT: " + me.UserName + " tried to change password. But provided invalid current pasword!");
                    ModelState.AddModelError(string.Empty, _localizer["8"]);
                }
            }

            return View();
        }
```

References DotNetIdentity.Models.ViewModels.ChangePasswordViewModel.NewPassword, and DotNetIdentity.Models.ViewModels.C

6.45.3.3 ConfirmEmail()

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

controller action for ConfirmEmail view

Parameters

userId	the user id
token	the confirmation token

Returns

redirect to action Index

Definition at line 546 of file UserController.cs.

```
        {
            var user = await _userManager.FindByIdAsync(userId);
            if (user != null)
            {
                var result = await _userManager.ConfirmEmailAsync(user, token);
                if (result.Succeeded)
                {
                    _logger.LogInformation("AUDIT: " + user.UserName + " successfully confirmed email addres!");
                    return RedirectToAction("Login");
                }
            }
            return RedirectToAction("Index", "Home");
        }
```

6.45.3.4 debugUserInfo()

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

Controller action for DebugUserInfo view

Returns

view DebugUserInfo

6.45.3.5 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.

```
        {
            var user = await _userManager.FindByNameAsync(User.Identity?.Name);
            user.TwoFactorType = Models.TwoFactorType.Authenticator;
            await _userManager.UpdateAsync(user);

            var authenticatorKey = await _userManager.GetAuthenticatorKeyAsync(user);
            if (authenticatorKey == null)
            {
                await _userManager.ResetAuthenticatorKeyAsync(user);
                authenticatorKey = await _userManager.GetAuthenticatorKeyAsync(user);
            }

            return View(new TwoFactorAuthenticatorViewModel
            {
                SharedKey = authenticatorKey,
                AuthenticationUri = _twoFactorAuthService.GenerateQrCodeUri(user.Email, authenticatorKey)
            });
        }
```

6.45.3.6 EnforceTwoFactorAuthenticator() [2/2]

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

controller POST method for view EnforceTwoFactorAuthenticator

Parameters

viewModel	type TwoFactorAuthenticatorViewModel

Returns

redirect to action EnforceTwoFactorResult

Definition at line 629 of file UserController.cs.

```
        {
            var user = await _userManager.FindByNameAsync(User.Identity?.Name);

            var verificationCode = viewModel.VerificationCode.Replace(" ", string.Empty).Replace("-", string.Empty);

            var isTokenValid = await _userManager.VerifyTwoFactorTokenAsync(user, _userManager.Options.Tokens.AuthenticatorTokenProvider, verificationCode);
            if (isTokenValid)
            {
                if(user.IsMfaForce && user.TwoFactorEnabled==false) {
                    await _userManager.SetTwoFactorEnabledAsync(user, true);
                    user.IsMfaForce = false;
                    await _userManager.UpdateAsync(user);
                }

                TempData["RecoveryCodes"] = await _userManager.GenerateNewTwoFactorRecoveryCodesAsync(user, 5);
            }
            _logger.LogInformation("AUDIT: " + user.UserName + " successfully configured 2fa after enforcement. Redirecting to action: EnforceTwoFactorResult");
            return RedirectToAction("EnforceTwoFactorResult", "User");
        }
```

References DotNetIdentity.Models.ViewModels.TwoFactorAuthenticatorViewModel.VerificationCode.

6.45.3.7 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.

```
        {
            await _signInManager.SignOutAsync();
            return View();
        }
```

6.45.3.8 ForgotPassword() [1/2]

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

controller action for ForgotPassword view

Returns

view ForgotPassword

6.45.3.9 ForgotPassword() [2/2]

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

controller POST method for ForgotPassword view

Parameters

viewModel	ForgotPasswordViewModel

Returns

redirect to action Index

Definition at line 455 of file UserController.cs.

```
        {
            if (ModelState.IsValid)
            {
                var user = await _userManager.FindByEmailAsync(viewModel.Email);
                if (user != null)
                {
                    var passwordResetToken = await _userManager.GeneratePasswordResetTokenAsync(user);

                    var passwordLink = Url.Action("ResetPassword", "User", new
                    {
                        userId = user.Id,
                        token = passwordResetToken
                    }, HttpContext.Request.Scheme);

                    await _emailHelper.SendAsync(new()
                    {
                        Subject = "Reset password",
                        Body = $"Please <a href='{passwordLink}'>click</a> to reset your password.",
                        To = user.Email
                    });
                    _logger.LogWarning("AUDIT: " + user.UserName + " requested a password reset!");
                    return RedirectToAction("Index", "Home");
                }
                else
                {
                    ModelState.AddModelError(string.Empty, _localizer["9"]);
                }
            }
            return View(viewModel);
        }
```

References DotNetIdentity.Models.ViewModels.ForgotPasswordViewModel.Email.

6.45.3.10 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.

```
        {
            if (ModelState.IsValid)
            {
                var user = await _userManager.FindByNameAsync(viewModel.UserName);
                if (user != null)
                {
                    await _signInManager.SignOutAsync();

                    var result = await _signInManager.PasswordSignInAsync(user, viewModel.Password, viewModel.RememberMe, true);
                    if(result.Succeeded && user.IsEnabled==false) {
                        await _signInManager.SignOutAsync();
                        _logger.LogWarning("AUDIT: " + user.UserName + " logged in successfully. But account is disabled!");
                        ModelState.AddModelError(string.Empty, _localizer["1"]);
                    }
                    else if(result.Succeeded && user.IsMfaForce==true) {
                        await _userManager.ResetAccessFailedCountAsync(user);
                        await _userManager.SetLockoutEndDateAsync(user, null);
                        _logger.LogInformation("AUDIT: " + user.UserName + " logged in successfully. But MFA is enforced. Redirecting to action: EnforceTwoFactorAuthenticator");
                        return RedirectToAction("EnforceTwoFactorAuthenticator");
                    }
                    else if (result.Succeeded && user.IsMfaForce==false)
                    {
                        await _userManager.ResetAccessFailedCountAsync(user);
                        await _userManager.SetLockoutEndDateAsync(user, null);
                        _logger.LogInformation("AUDIT: " + user.UserName + " logged in successfully.");
                        var returnUrl = TempData["ReturnUrl"];
                        if (returnUrl != null)
                        {
                            return Redirect(returnUrl.ToString() ?? "/");
                        }
                        return RedirectToAction("Index", "Admin");
                    }
                    else if (result.RequiresTwoFactor)
                    {
                        _logger.LogInformation("AUDIT: " + user.UserName + " logged in successfully. But MFA is enabled. Redirecting to action: TwoFactorLogin");
                        return RedirectToAction("TwoFactorLogin", new { ReturnUrl = TempData["ReturnUrl"] });
                    }
                    else if (result.IsLockedOut)
                    {
                        var lockoutEndUtc = await _userManager.GetLockoutEndDateAsync(user);
                        var timeLeft = lockoutEndUtc.Value - DateTime.UtcNow;
                        _logger.LogWarning("AUDIT: " + user.UserName + " logged in successfully. But Account is locked out for at least another " + timeLeft.Minutes + "minutes!");
                        ModelState.AddModelError(string.Empty, _localizer["2", timeLeft.Minutes.ToString()]);
                    }
                    else if (result.IsNotAllowed)
                    {
                        _logger.LogWarning("AUDIT: " + user.UserName + " logged in successfully. But email is not confirmed!");
                        ModelState.AddModelError(string.Empty, _localizer["3"]);
                    }
                    else
                    {
                        _logger.LogWarning("AUDIT: " + user.UserName + " login in failed. Invalid username or password!");
                        ModelState.AddModelError(string.Empty, _localizer["4"]);
                    }
                }
                else
                {
                    _logger.LogWarning("AUDIT: " + "Login in failed. Invalid username or password!");
                    ModelState.AddModelError(string.Empty, _localizer["4"]);
                }
            }
            return View(viewModel);
        }
```

References DotNetIdentity.Models.ViewModels.SignInViewModel.Password, DotNetIdentity.Models.ViewModels.SignInViewModel.Re and DotNetIdentity.Models.ViewModels.SignInViewModel.UserName.

6.45.3.11 Login() [2/2]

```
async Task< IActionResult > DotNetIdentity.Controllers.UserController.Login (
              string? returnUrl ) [inline]
```

Controller action for Login view

Parameters

returnUrl	

Returns

view Login

Definition at line 146 of file UserController.cs.

```
        {
            await _signInManager.SignOutAsync();
            if (returnUrl != null)
            {
                TempData["ReturnUrl"] = returnUrl;
            }
            return View();
        }
```

6.45.3.12 Logout()

```
async Task DotNetIdentity.Controllers.UserController.Logout ( )
```

controller action for Logout

Returns

6.45.3.13 ping()

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

Controller action for Ping view

Returns

view Ping

6.45.3.14 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.

```
        {
            var me = await _userManager.FindByNameAsync(User.Identity?.Name);
            if (me == null)
            {
                await _signInManager.SignOutAsync();
                return RedirectToAction("Index", "Home");
            }
            var mod = new UpdateProfileViewModel();
            mod.Email = me.Email;
            mod.BirthDay = me.BirthDay;
            mod.UserName = me.UserName;
            mod.PhoneNumber = me.PhoneNumber;
            mod.Gender = me.Gender;
            //return View(me.Adapt<UpdateProfileViewModel>());
            return View(mod);
        }
```

6.45.3.15 Profile() [2/2]

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

controller POST methood for Profile view

Parameters

viewModel	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 =>
       a.PhoneNumber == viewModel.PhoneNumber))
00338                     {
00339                         ModelState.AddModelError(string.Empty, _localizer["6"]);
00340                     }
00341                     else
00342                     {
00343                         if(viewModel.ProfilePicture!=null) {
00344                             using (var memoryStream = new MemoryStream())
00345                             {
00346                                 await viewModel.ProfilePicture.CopyToAsync(memoryStream);
00347 
00348                                 // Upload the file if less than 2 MB
00349                                 if (memoryStream.Length < 2097152)
00350                                 {
00351                                     var format = "image/png";
00352                                     var Content = memoryStream.ToArray();
00353                                     var imageData =
       $"data:{format};base64,{Convert.ToBase64String(Content)}";
00354                                     me.ProfilePicture = imageData;
00355 
00356                                 }
00357                                 else
00358                                 {
00359                                     ModelState.AddModelError("File", _localizer["7"]);
00360                                 }
00361                             }
00362                         } else {
00363                             me.ProfilePicture = null;
00364                         }
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
       user profile");
00381 
00382                             return RedirectToAction("Index", "Home");
00383                         }
00384                         result.Errors.ToList().ForEach(f => ModelState.AddModelError(string.Empty,
       f.Description));
00385                     }
00386                 }
00387                 else
00388                 {
00389                     await _signInManager.SignOutAsync();
00390                     return RedirectToAction("Index", "Home");
00391                 }
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.UpdateProfileView

6.45.3.16 Register() [1/2]

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

Controller action for Register view

Returns

view Register

6.45.3.17 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.

```
        {
            if (ModelState.IsValid)
            {
                var user = new AppUser
                {
                    FirstName = viewModel.FirstName,
                    LastName = viewModel.LastName,
                    UserName = viewModel.UserName,
                    Email = viewModel.Email,
                    Gender = viewModel.Gender,
                    BirthDay = viewModel.BirthDay,
                    Department = viewModel.Department.ToString(),
                    TwoFactorType = Models.TwoFactorType.None,
                    CreatedOn = DateTime.UtcNow,
                    IsEnabled = true
                };

                var result = await _userManager.CreateAsync(user, viewModel.Password);
                if (result.Succeeded)
                {
                    await _userManager.AddToRoleAsync(user, "User");
                    var confirmationToken = await _userManager.GenerateEmailConfirmationTokenAsync(user);
                    var confirmationLink = Url.Action("ConfirmEmail", "User", new
                    {
                        userId = user.Id,
                        token = confirmationToken
                    }, HttpContext.Request.Scheme);

                    await _emailHelper.SendAsync(new()
                    {
                        Subject = "Confirm e-mail",
                        Body = $"Please <a href='{confirmationLink}'>click</a> to confirm your e-mail address.",
                        To = user.Email
                    });
                    _logger.LogInformation("AUDIT: " + "registered successfully new user: " + user.UserName);
                    return RedirectToAction("Login");
                }
                result.Errors.ToList().ForEach(f => ModelState.AddModelError(string.Empty, f.Description));
            }
            return View(viewModel);
        }
```

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.UserNar

6.45.3.18 ResetPassword() [1/2]

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

controller POST method for ResetPassword view

Parameters

viewModel	type ResetPasswordViewModel

Returns

redirect to action Login

Definition at line 513 of file UserController.cs.

```
        {
            if (ModelState.IsValid)
            {
                var user = await _userManager.FindByIdAsync(viewModel.UserId);
                if (user != null)
                {
                    var result = await _userManager.ResetPasswordAsync(user, viewModel.Token, viewModel.Password);
                    if (result.Succeeded)
                    {
                        await _userManager.UpdateSecurityStampAsync(user);
                        _logger.LogInformation("AUDIT: " + user.UserName + "successfully resetted own password.");
                        return RedirectToAction("Login", "User");
                    }
                    else
                    {
                        result.Errors.ToList().ForEach(f => ModelState.AddModelError(string.Empty, f.Description));
                    }
                }
                else
                {
                    ModelState.AddModelError(string.Empty, "User not found.");
                }
            }
            return View(viewModel);
        }
```

References DotNetIdentity.Models.ViewModels.ResetPasswordViewModel.Password, DotNetIdentity.Models.ViewModels.ResetPassw and DotNetIdentity.Models.ViewModels.ResetPasswordViewModel.UserId.

6.45.3.19 ResetPassword() [2/2]

```
IActionResult DotNetIdentity.Controllers.UserController.ResetPassword (
            string userId,
            string token )  [inline]
```

controller action for ResetPassword view

Parameters

userId	the user id
token	the reset token

Returns

view ResetPassword

Definition at line 493 of file UserController.cs.

```
        {
            if (userId == null || token == null)
            {
                return RedirectToAction("Login", "User");
            }

            return View(new ResetPasswordViewModel
            {
                UserId = userId,
                Token = token
            });
        }
```

6.45.3.20 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.

```
        {
            var user = await _userManager.FindByNameAsync(User.Identity?.Name);
            if (user.TwoFactorEnabled && user.TwoFactorType == Models.TwoFactorType.Authenticator)
            {
                var authenticatorKey = await _userManager.GetAuthenticatorKeyAsync(user);
                if (authenticatorKey == null)
                {
                    await _userManager.ResetAuthenticatorKeyAsync(user);
                    authenticatorKey = await _userManager.GetAuthenticatorKeyAsync(user);
                }

                return View(new TwoFactorAuthenticatorViewModel
                {
                    SharedKey = authenticatorKey,
                    AuthenticationUri = _twoFactorAuthService.GenerateQrCodeUri(user.Email, authenticatorKey)
                });
            }
            return RedirectToAction("Index", "Home");
        }
```

6.45.3.21 TwoFactorAuthenticator() [2/2]

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

controller POST method for view TwoFActorAuthenticator

Parameters

viewModel	type TwoFactorAuthenticatorViewModel

Returns

redirect to action TwoFactorType

Definition at line 690 of file UserController.cs.

```
        {
            var user = await _userManager.FindByNameAsync(User.Identity?.Name);

            var verificationCode = viewModel.VerificationCode.Replace(" ", string.Empty).Replace("-", string.Empty);

            var isTokenValid = await _userManager.VerifyTwoFactorTokenAsync(user, _userManager.Options.Tokens.AuthenticatorTokenProvider, verificationCode);
            if (isTokenValid)
            {
                if(user.IsMfaForce && user.TwoFactorEnabled==false) {
                    await _userManager.SetTwoFactorEnabledAsync(user, true);
                    user.IsMfaForce = false;
                    await _userManager.UpdateAsync(user);
                }

                TempData["RecoveryCodes"] = await _userManager.GenerateNewTwoFactorRecoveryCodesAsync(user, 5);
            }
            _logger.LogInformation("AUDIT: " + user.UserName + " successfully configured 2fa!");
            return RedirectToAction("TwoFactorType", "User");
        }
```

References DotNetIdentity.Models.ViewModels.TwoFactorAuthenticatorViewModel.VerificationCode.

6.45.3.22 TwoFactorLogin() [1/2]

```
async Task< IActionResult > DotNetIdentity.Controllers.UserController.TwoFactorLogin (
              string? returnUrl ) [inline]
```

Controller action for view TwoFactorLogin

Parameters

returnUrl	optional return url as string

Returns

view TwoFactorLogin

Definition at line 232 of file UserController.cs.

```
        {
            if (returnUrl != null)
            {
                TempData["ReturnUrl"] = returnUrl;
            }

            var user = await _signInManager.GetTwoFactorAuthenticationUserAsync();

            if(user.TwoFactorType == DotNetIdentity.Models.TwoFactorType.Email) {
                var token = await _userManager.GenerateTwoFactorTokenAsync(user, "Email");
                await _emailHelper.SendAsync(new()
                    {
                        Subject = "Your 2fa code",
                        Body = $"Please use this " + token + " 2fa code to verify your login.",
                        To = user.Email
                    });
                    _logger.LogInformation("AUDIT: send otp code to " + user.Email);
            }

            return View(new TwoFactorLoginVieWModel
            {
                TwoFactorType = user.TwoFactorType,
            });
        }
```

References DotNetIdentity.Models.ViewModels.TwoFactorLoginVieWModel.TwoFactorType.

6.45.3.23 TwoFactorLogin() [2/2]

```
async Task< IActionResult > DotNetIdentity.Controllers.UserController.TwoFactorLogin (
              TwoFactorLoginVieWModel vieWModel )  [inline]
```

controller POST method for TwoFactorLogin view

Parameters

vieWModel	tyle TwoFactorLoginVieWModel

Returns

redirect to return url

Definition at line 264 of file UserController.cs.

```
        {
            var user = await _signInManager.GetTwoFactorAuthenticationUserAsync();

            if (user.TwoFactorType == Models.TwoFactorType.Authenticator)
            {
                var result = vieWModel.IsRecoveryCode ? await _signInManager.TwoFactorRecoveryCodeSignInAsync(vieWModel.VerificationCode) : await _signInManager.TwoFactorAuthenticatorSignInAsync(vieWModel.VerificationCode, true, false);
                if (result.Succeeded)
                {
                    _logger.LogInformation("AUDIT: " + user.UserName + "Authenticator 2fa verified successfully");
                    return Redirect(TempData["ReturnUrl"]?.ToString() ?? "/");
                }
                _logger.LogWarning("AUDIT: " + user.UserName + "Authenticator 2fa verification failed. Invalid otp code");
                ModelState.AddModelError(string.Empty, _localizer["5"]);
            }
            else if (user.TwoFactorType == Models.TwoFactorType.Email)
            {
                var result = vieWModel.IsRecoveryCode ? await _signInManager.TwoFactorRecoveryCodeSignInAsync(vieWModel.VerificationCode) : await _signInManager.TwoFactorSignInAsync("Email", vieWModel.VerificationCode, true, false);
                if (result.Succeeded)
                {
                    _logger.LogInformation("AUDIT: " + user.UserName + "Email 2fa verified successfully");
                    return Redirect(TempData["ReturnUrl"]?.ToString() ?? "/");
                }
                _logger.LogWarning("AUDIT: " + user.UserName + "Email 2fa verification failed. Invalid otp code");
                ModelState.AddModelError(string.Empty, _localizer["5"]);
            }

            return View(vieWModel);
        }
```

References DotNetIdentity.Models.ViewModels.TwoFactorLoginVieWModel.VerificationCode.

6.45.3.24 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.

```
        {
            var user = await _userManager.FindByNameAsync(User.Identity?.Name);

            return View(new TwoFactorTypeViewModel
            {
                TwoFactorType = user.TwoFactorType
            });
        }
```

6.45.3.25 TwoFactorType() [2/2]

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

controller POST method for view TwoFactorType

Parameters

viewModel	TwoFactorTypeViewModel

Returns

rediorect to action TwoFactorType

Definition at line 583 of file UserController.cs.

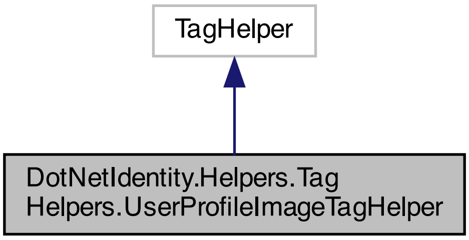
```
        {
            var user = await _userManager.FindByNameAsync(User.Identity?.Name);
            user.TwoFactorType = viewModel.TwoFactorType;
            await _userManager.UpdateAsync(user);
            await _userManager.SetTwoFactorEnabledAsync(user, user.TwoFactorType != Models.TwoFactorType.None);

            if (viewModel.TwoFactorType == Models.TwoFactorType.Authenticator)
            {
                _logger.LogInformation("AUDIT: " + user.UserName + " set authenticator as 2fa method. Redirecting to action: TwoFactorAuthenticator");
                return RedirectToAction("TwoFactorAuthenticator", "User");
            }
            _logger.LogInformation("AUDIT: " + user.UserName + " set " + user.TwoFactorType.ToString() + " as 2fa method.");
            return RedirectToAction("TwoFactorType", "User");
        }
```

References DotNetIdentity.Models.ViewModels.TwoFactorTypeViewModel.TwoFactorType.

6.46 DotNetIdentity.Helpers.TagHelpers.UserProfileImageTagHelper Class Reference

html taghelper class

Inheritance diagram for DotNetIdentity.Helpers.TagHelpers.UserProfileImageTagHelper:



Public Member Functions

- UserProfileImageTagHelper (UserManager< AppUser > userManager)

  class contsructor
- override async Task ProcessAsync (TagHelperContext context, TagHelperOutput output)

  override ProcessAsync

Properties

- string UserId = default! `[get, set]`

  html attribute definition

6.46.1 Detailed Description

html taghelper class

Definition at line 12 of file UserProfileImageTagHelper.cs.

6.46.2 Constructor & Destructor Documentation

6.46.2.1 UserProfileImageTagHelper()

```
DotNetIdentity.Helpers.TagHelpers.UserProfileImageTagHelper.UserProfileImageTagHelper (
            UserManager< AppUser > userManager )  [inline]
```

class contsructor

Parameters

userManager	type UserManager

Definition at line 23 of file UserProfileImageTagHelper.cs.

```
        {
            _userManager = userManager;
        }
```

6.46.3 Member Function Documentation

6.46.3.1 ProcessAsync()

```
override async Task DotNetIdentity.Helpers.TagHelpers.UserProfileImageTagHelper.ProcessAsync (
            TagHelperContext context,
            TagHelperOutput output )  [inline]
```

override ProcessAsync

Parameters

context	type TagHelperContext
output	type TagHelperOutput

Returns

TagHelperOutput

Definition at line 41 of file UserProfileImageTagHelper.cs.

```
        {
            var user = await _userManager.FindByIdAsync(UserId);

            var sb = new StringBuilder();
            if(user.ProfilePicture==null) {
                sb.Append($"<img src=’/assets/images/faces/user.png’ height=’32’/>");
            } else {
                sb.Append($"<img src=’" + user.ProfilePicture + "’ height=’32’/>");
            }

            output.Content.SetHtmlContent(sb.ToString());
        }
```

References DotNetIdentity.Helpers.TagHelpers.UserProfileImageTagHelper.UserId.

6.46.4 Property Documentation

6.46.4.1 UserId

```
string DotNetIdentity.Helpers.TagHelpers.UserProfileImageTagHelper.UserId = default!  [get],
[set]
```

html attribute definition

Definition at line 33 of file UserProfileImageTagHelper.cs.

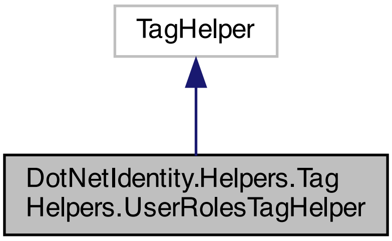
```
00033 { get; set; } = default!;
```

Referenced by DotNetIdentity.Helpers.TagHelpers.UserProfileImageTagHelper.ProcessAsync().

6.47 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

6.47.1 Detailed Description

html tag helper class

Definition at line 12 of file UserRolesTagHelper.cs.

6.47.2 Constructor & Destructor Documentation

6.47.2.1 UserRolesTagHelper()

```
DotNetIdentity.Helpers.TagHelpers.UserRolesTagHelper.UserRolesTagHelper (
            UserManager< AppUser > userManager ) [inline]
```

class constructor

Parameters

userManager	

Definition at line 23 of file UserRolesTagHelper.cs.

```
        {
            _userManager = userManager;
        }
```

6.47.3 Member Function Documentation

6.47.3.1 ProcessAsync()

```
override async Task DotNetIdentity.Helpers.TagHelpers.UserRolesTagHelper.ProcessAsync (
            TagHelperContext context,
            TagHelperOutput output ) [inline]
```

method to override ProcessAsync

Parameters

context	type TagHelperContext
output	type TagHelperOutput

Returns

TagHelperOutput

Definition at line 41 of file UserRolesTagHelper.cs.

```
        {
            var user = await _userManager.FindByIdAsync(UserId);
            var roles = await _userManager.GetRolesAsync(user);

            var sb = new StringBuilder();

            if (roles.Count() > 0)
            {
                foreach (var role in roles)
                {
                    sb.Append($"<span class='badge rounded-pill bg-danger'>{role}</span>");
                }
            }
            else
            {
                sb.Append("No roles found");
            }

            output.Content.SetHtmlContent(sb.ToString());
        }
```

References DotNetIdentity.Helpers.TagHelpers.UserRolesTagHelper.UserId.

6.47.4 Property Documentation

6.47.4.1 UserId

```
string DotNetIdentity.Helpers.TagHelpers.UserRolesTagHelper.UserId = default!  [get], [set]
```

attribute definition

Definition at line 33 of file UserRolesTagHelper.cs.

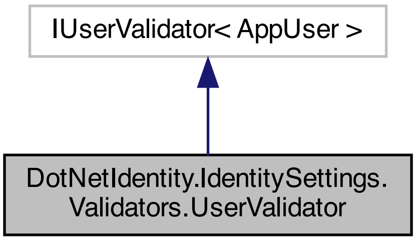
```
00033 { get; set; } = default!;
```

Referenced by DotNetIdentity.Helpers.TagHelpers.UserRolesTagHelper.ProcessAsync().

6.48 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

6.48.1 Detailed Description

class to validate IdentityUser

Definition at line 9 of file UserValidator.cs.

6.48.2 Member Function Documentation

6.48.2.1 ValidateAsync()

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

Task to validate IdentityUser

Parameters

manager	UserManager
user	AppUser

Returns

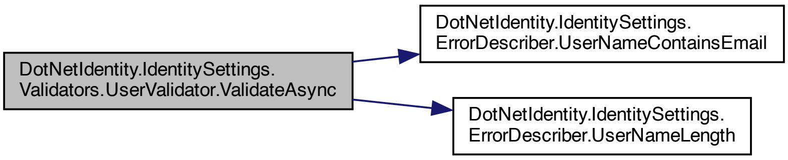
Task of type IdentiyResult

Definition at line 17 of file UserValidator.cs.

```
        {
            var errors = new List<IdentityError>();

            if (user.UserName.Length < 6)
            {
                errors.Add(ErrorDescriber.UserNameLength());
            }
            if (user.Email.Substring(0, user.Email.IndexOf("@")) == user.UserName)
            {
                errors.Add(ErrorDescriber.UserNameContainsEmail());
            }

            if (errors.Any())
            {
                return Task.FromResult(IdentityResult.Failed(errors.ToArray()));
            }
            return Task.FromResult(IdentityResult.Success);
        }
```

References DotNetIdentity.IdentitySettings.ErrorDescriber.UserNameContainsEmail(), and DotNetIdentity.IdentitySettings.ErrorDescri

Here is the call graph for this function:


DotNetIdentity.IdentitySettings.
Validators.UserValidator.ValidateAsync
DotNetIdentity.IdentitySettings.
ErrorDescriber.UserNameContainsEmail
DotNetIdentity.IdentitySettings.
ErrorDescriber.UserNameLength

Index