How to save new record with hashed password in my custom table instead of aspnet user?

Asked 8 years, 5 months ago Modified 7 years, 4 months ago Viewed 2k times

12

I am using asp.net identity to create new user but getting error:

> Cannot insert the value NULL into column 'Id', table 'Mydb.dbo.AspNetUsers'; column does not allow nulls. INSERT fails.\r\nThe statement has been terminated

But here I don't have any such table like **AspNetUsers** but instead I have my own table that is **Users**.

Code: Web.config: 2 conection strings

```
 <add name="myEntities" connectionString="metadata=res://*/DataModel.csdl|
res://*/DataModel.ssdl|res://*/
DataModel.msl;provider=System.Data.SqlClient;provider connection
string=&quot;data source=;initial catalog=mydb;user
id=sa;password=sdfsdfsdf;MultipleActiveResultSets=True;App=EntityFramework&quot;"
providerName="System.Data.EntityClient" />
 <add name="MyConnString" connectionString="data source=;initial
catalog=Mydb;user id=sa;password=sdfsdfsdf;"
providerName="System.Data.SqlClient" />
```

IdentityModel.cs:

```
public class ApplicationUser : IdentityUser
    {
        public async Task<ClaimsIdentity>
GenerateUserIdentityAsync(UserManager<ApplicationUser> manager)
        {
            // Note the authenticationType must match the one defined in
CookieAuthenticationOptions.AuthenticationType
            this.SecurityStamp = Guid.NewGuid().ToString();
            var userIdentity = await manager.CreateIdentityAsync(this,
DefaultAuthenticationTypes.ApplicationCookie);
            // Add custom user claims here
            return userIdentity;
        }

        public string Id { get; set; }
        public string FirstName { get; set; }
        public string LastName { get; set; }
        public virtual string Email { get; set; }
        public string Password { get; set; }
        public string Role { get; set; }
        public Nullable<bool> IsActive { get; set; }
        public Nullable<int> CreatedBy { get; set; }
        public Nullable<System.DateTime> CreatedDate { get; set; }
        public Nullable<System.DateTime> LastLogin { get; set; }

        public ApplicationUser()
        {

        }

        public ApplicationUser(string email, string firstName, string lastName,
string designation, bool isActive)
```

```
        {
            Email = email;
            FirstName = firstName;
            LastName = lastName;
            Designation = designation;
            IsActive = isActive;
        }
    }
  public class ApplicationDbContext : IdentityDbContext<ApplicationUser>
    {
        public ApplicationDbContext()
            : base("MyConnString", throwIfV1Schema: false)
        {
        }

        public static ApplicationDbContext Create()
        {
            return new ApplicationDbContext();
        }
    }
```

UserStore1.cs:

```
public class UserStore1 : IUserStore<ApplicationUser>,
IUserPasswordStore<ApplicationUser>
    {
        private readonly HttpContext _httpContext;
        UserStore<IdentityUser> userStore = new UserStore<IdentityUser>(new
ApplicationDbContext());

        public System.Threading.Tasks.Task CreateAsync(ApplicationUser user)
        {
            HttpContext.Current = _httpContext ?? HttpContext.Current;
            var context = userStore.Context as ApplicationDbContext;
           context.Users.Add(user);
           context.Configuration.ValidateOnSaveEnabled = false;
           context.SaveChanges();
            return Task.FromResult(true);
        }
     }
```

Controller:

```
     [Authorize]
    public class AccountController : Controller
    {
        public AccountController()
            : this(new UserManager<ApplicationUser>(new UserStore1()))
        {
        }

        public AccountController(UserManager<ApplicationUser> userManager)
        {
            UserManager = userManager;
        }
        public UserManager<ApplicationUser> UserManager { get; private set; }
[HttpPost]
```

```
        [AllowAnonymous]
        [ValidateAntiForgeryToken]
        public async Task<ActionResult> Login(string email, string password,
bool rememberMe = false, string returnUrl = null)
        {
            if (ModelState.IsValid)
            {
                var user = new ApplicationUser
                {
                    FirstName= "Abc",
                    LastName= "Pqr",
                    UserName="Abc@yahoo.com",
                    SecurityStamp = Guid.NewGuid().ToString()
                };

                var result= await UserManager.CreateAsync(user,"123456");
            }
            return View();
        }
    }
```

Note: I have autogenerated Id in my database table field and that Id is Int.

Update: I am using database first(edmx) and the table that I am using are custom tables for inserting new records(for eg:**Users**).

At first I have implemented microsoft asp.net identity as shown in below question but 1 user pointed out that I am not using **ApplicationUser class** which is responsible for handling **sign in,cookies** etc so I am now trying to use **ApplicationUser** class:

How to give custom implementation of UpdateAsync method of asp.net identity?

I am really now regretting over my decision to choose Microsoft Identity Framework for Authentication purpose as because I am really finding it complex but now as I have move forward I have to go with it.

c#   asp.net   asp.net-mvc   asp.net-identity   owin

Share Improve this question Follow

edited Sep 26, 2017 at 21:38
halfer
20.4k 19 108 201

asked Sep 3, 2016 at 7:35
I Love Stackoverflow
6,868 22 112 238

It seems to me that you're missing something. You haven't defined your model in `OnModelCreating` for your `ApplicationDbContext` . Check this answer – LeftyX Sep 4, 2016 at 16:06

@LeftyX sorry but I am not using code first.i am using databse first(.edmx) and I have already seen some of your answers and infact tried it but unfortunately you have used code first and I am using edmx – I Love Stackoverflow Sep 4, 2016 at 17:55

1 Answer

Sorted by: Highest score (default)

7

+25

The inconsistency i found, in `ApplicationUser` class you are declaring property `Id` and `Email` which is wrong because the `IdentityUser` class already have those properties. This may arise the issue. But you can **override** them if necessary. Also the constructor you are using isn't necessary. The `ApplicationUser` class should be:

```
public class ApplicationUser : IdentityUser
{
    public async Task<ClaimsIdentity>
GenerateUserIdentityAsync(UserManager<ApplicationUser> manager)
    {
        // Note the authenticationType must match the one defined in
CookieAuthenticationOptions.AuthenticationType
        this.SecurityStamp = Guid.NewGuid().ToString();
        var userIdentity = await manager.CreateIdentityAsync(this,
DefaultAuthenticationTypes.ApplicationCookie);
        // Add custom user claims here
        return userIdentity;
    }

    public string FirstName { get; set; }
    public string LastName { get; set; }
    public string Password { get; set; }
    public string Role { get; set; }
    public bool? IsActive { get; set; }
    public int? CreatedBy { get; set; }
    public DateTime? CreatedDate { get; set; }
    public DateTime? LastLogin { get; set; }

}
```

Second thing, you are creating the **user** inside `Login` action which is also not valid. you should do it inside `Register` action. Following in an example:

```
 public async Task<ActionResult> Register(RegisterViewModel model)
    {
        if (ModelState.IsValid)
        {
          var user = new ApplicationUser
            {
                FirstName = "Abc",
                LastName = "Pqr",
                UserName = "Abc@yahoo.com",
                Email= model.Email,
                Password= model.Password,
                PasswordHash =
UserManager.PasswordHasher.HashPassword(model.Password),
                SecurityStamp = Guid.NewGuid().ToString()
            };

            var result = await UserManager.CreateAsync(user);
        if (result.Succeeded)
            {
                await SignInManager.SignInAsync(user, isPersistent:false,
rememberBrowser:false);
                return RedirectToAction("Index", "Home");
            }
            AddErrors(result);
        }

        return View(model);
    }
```

Hope this will help :)

Share Improve this answer Follow

edited Sep 8, 2016 at 14:14

answered Sep 8, 2016 at 14:05

Mahbubur Rahman Manik
5,161 3 41 47

This is not the problem that i am creating user in login or in register method.this was just for testing that custom implementation of CreateAsync method is working fine or not.If you see my UserStore1 class you will notice custom implementation of CreateAsync method – I Love Stackoverflow Sep 8, 2016 at 15:37

you are not using the custom implementation of `CreateAsync` . look at the signature of method. you are using the default `CreateAsync` of `UserManager` . – Mahbubur Rahman Manik Sep 8, 2016 at 18:04

When I put a debugger on my custom createasync method which is on userstore1 then debugger hit my method – I Love Stackoverflow Sep 8, 2016 at 18:19

I actually focused on `ApplicationUser` class. Did the change resolve the issue or any other error message ? – Mahbubur Rahman Manik Sep 8, 2016 at 18:24

Nope still getting same error and I guess that is not a problem because what I think is that problem is related to entity framework and applicationdbcontext – I Love Stackoverflow Sep 8, 2016 at 18:28