# Setting Up Identity

1. Select Authentication Type when creating project

[Add ScreenShot]

1. Update ConnectionString in app.Setting

[Add ScreenShot]

1. Add New Scaffolded item (only if required to edit formatting or code)

[Add ScreenShot]

# Adding PhoneNumber to Registration Page

1. Note: PhoneNumber field is already existing in the AspNetUsers table
2. In Register.cshtml.cs add a field to the class named **InputModel**

[Display(Name = "PhoneNumber / MobileNumber")]

[DataType(DataType.PhoneNumber)]

public string PhoneNumber { get; set; }

1. Add an input element in Register.cshtml

<div class="form-floating">

<**input** **asp-for**="Input.PhoneNumber" class="form-control"

autocomplete="phonenumber" aria-required="true" />

<**label** **asp-for**="Input.PhoneNumber"></**label**>

<**span** **asp-validation-for**="Input.PhoneNumber" class="text-danger"></**span**>

</div>

1. In Register.cshtml.cs, in the line before \_userManager.CreateAsync, add the following code.

user.PhoneNumber = Input.PhoneNumber;

# Adding custom fields to User Table

1. Let us add 1 additional fields, Age
2. Create a new class ApplicationIsdentityUser inheriting from IdentityUser and add the Age field.s

public class ApplicationIdentityUser : IdentityUser

{

public int Age { get; set; }

}

1. In code files replace all mentions of IdentityUser to ApplicationIdentityUser
2. In ApplicationDbContext file, change the base class signature from IdentityDbContext to

IdentityDbContext<ApplicationIdentityUser, IdentityRole, string>

1. Do Add-Migration and Update-Database
2. To add this field in the Register page, follow steps in Topic B above

# Adding Roles

1. In Program.cs add the following

builder.Services

.AddDefaultIdentity<ApplicationIdentityUser>(options =>

options.SignIn.RequireConfirmedAccount = false)

.AddRoles<IdentityRole>()

.AddEntityFrameworkStores<ApplicationDbContext>();

1. Add Role Field in InputModel class (in Register.cshtml.cs)

[Display(Name = "Role")]

public string Role { get; set; }

1. Add radio buttons for any roles you need e.g.

Role: ( \* ) User ( ) Admin ( ) SuperAdmin

1. In Register.cshtml.cs, after user is created, add the following code

Note: This will add records to the *AspNetRoles* and *AspNetUserRoles* tables in database

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// Pre checking if this role is available in the database

var role = await \_roleManager.FindByNameAsync(Input.Role);

// if not available insert it

if (role == null) await \_roleManager.CreateAsync(new IdentityRole(Input.Role));

// Associate this role with current user

await \_userManager.AddToRoleAsync(user, Input.Role);

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

1. Add the role wise authorization in controllers as follows

[Authorize(Roles = "SuperAdmin")] // Allow only SuperAdmin

[Authorize(Roles = "Admin,SuperAdmin")] // Allow admin and superadmin

[Authorize(Roles = "User,Admin")] // All User and Admin

[Authorize()] // Allow all logged in users

1. The [Authorize] Attribute can be used with individual actions or at controller level. If used at controller level, all actions in that controller will be authorized.
2. You can also use the [AllowAnonymous] attribute. This could be used, if the Controller has [Authorize] and if you need any of the actions to be allowed for non-logged in users also.