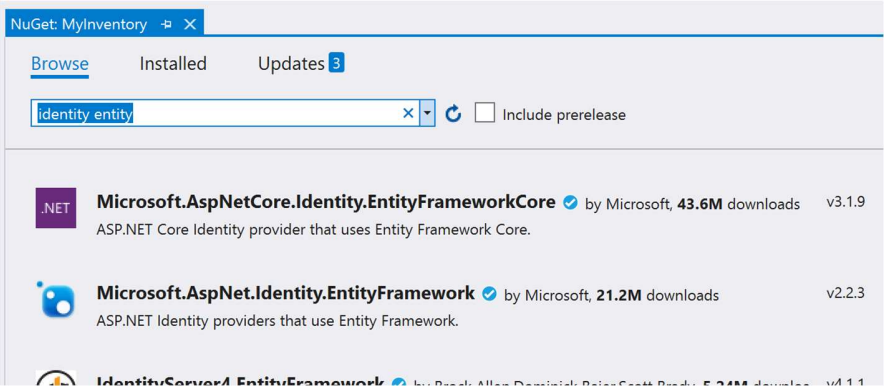
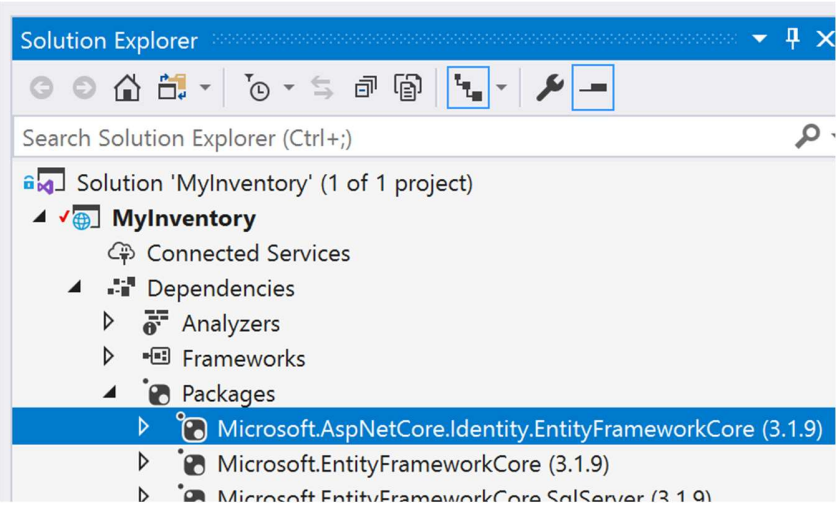
## Creating a Basic Registration and Login Pages using ASP.NET Core Identity

1. Open the existing **MyInventory** project using **Microsoft Visual Studio**.
2. From the **Solution Explorer** section, right-click on the project, then select **Manage NuGet Packages…**
3. From the **NuGet Package Manager** section, input the keywords **identity entity**.

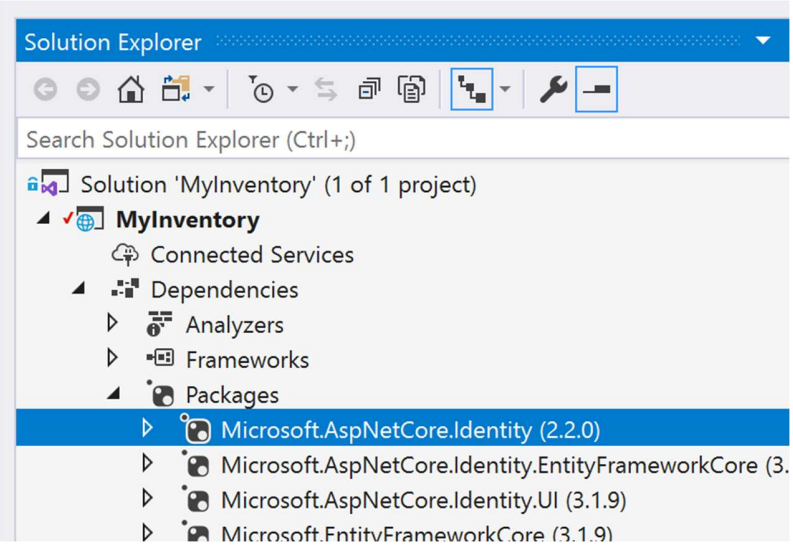



4. Select **Microsoft.AspNetCore.Identity.EntityFrameworkCore** package. Click the **Install** button to proceed. (Ensure that all existing packages are updated.)
5. From the **License Acceptance** window, other packages required will be installed. Click the **I Accept** button to proceed.
6. The package has been installed successfully.

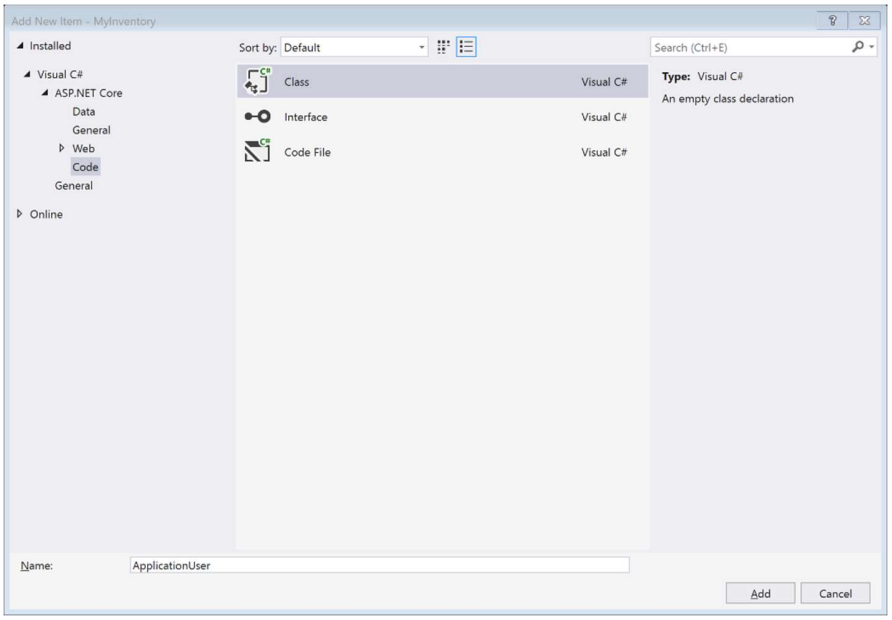



7. From the **NuGet Package Manager** section, input the keywords **identity core**.
8. Select **Microsoft.AspNetCore.Identity** package. Click the **Install** button to proceed. (Ensure that all existing packages are updated.)
9. From the **License Acceptance** window, other packages required will be installed. Click the **I Accept** button to proceed.

10. The package has been installed successfully.




11. From the **Solution Explorer** section, right-click **Models** folder, then select **Add > Class…**
12. From the **Add New Item** window, rename Class.cs (or equivalent) to **ApplicationUser.cs**.



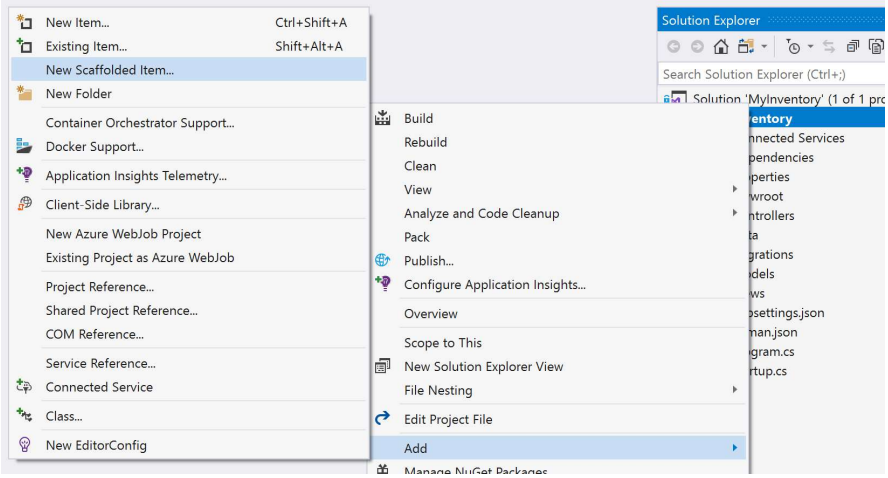

13. Click the **Add** button to proceed.
14. From **ApplicationUser.cs**, inherit the class from **IdentityUser** class and include your [custom user data](). (Ensure that the namespace **Microsoft.AspNetCore.Identity** has been added.)

```
ApplicationUser.cs
MyInventory                                   MyInventory.Models.ApplicationUser
1   using Microsoft.AspNetCore.Identity;
2   using System;
3   using System.Collections.Generic;
4   using System.Linq;
5   using System.Threading.Tasks;
6
7   namespace MyInventory.Models
8   {
9       public class ApplicationUser : IdentityUser
10      {
11          public string FirstName { get; set; }
12
13          public string LastName { get; set; }
14      }
```

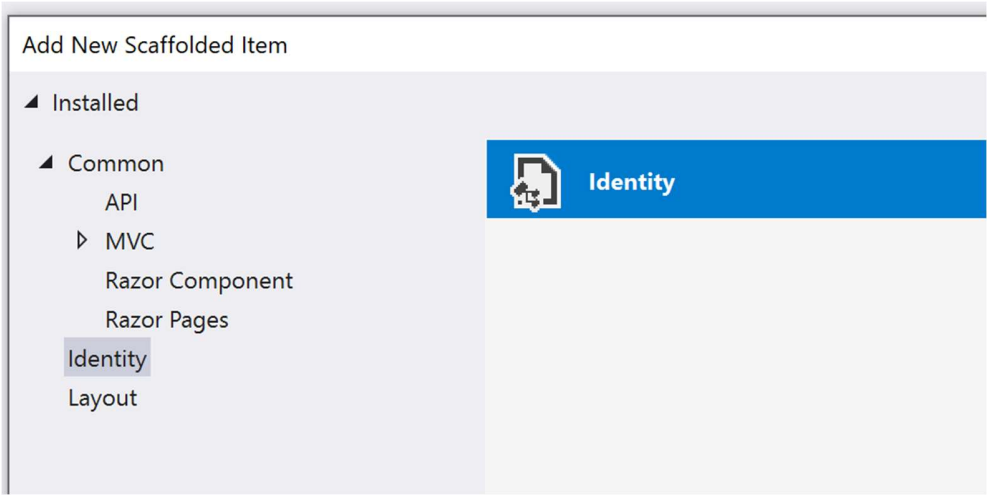
15. From the **Solution** Explorer section, open **Data > ApplicationDbContext.cs.**
16. Replace **DbContext** class to **IdentityDbContext**. Ensure that the namespace **Microsoft.AspNetCore.Identity.EntityFrameworkCore** has been added. Use the **ApplicationUser** type as a generic argument for the context.

```
ApplicationDbContext.cs
MyInventory                    MyInventory.Data.ApplicationDbContext    ApplicationDbContex
1   using System;
2   using System.Collections.Generic;
3   using System.Linq;
4   using System.Threading.Tasks;
5   using Microsoft.AspNetCore.Identity.EntityFrameworkCore;
6   using Microsoft.EntityFrameworkCore;
7   using MyInventory.Models;
8
9   namespace MyInventory.Data
10  {
11      public class ApplicationDbContext : IdentityDbContext<ApplicationUser>
12      {
13          public ApplicationDbContext(DbContextOptions<ApplicationDbContext> options)
14              : base(options)
15          {
16          }
```

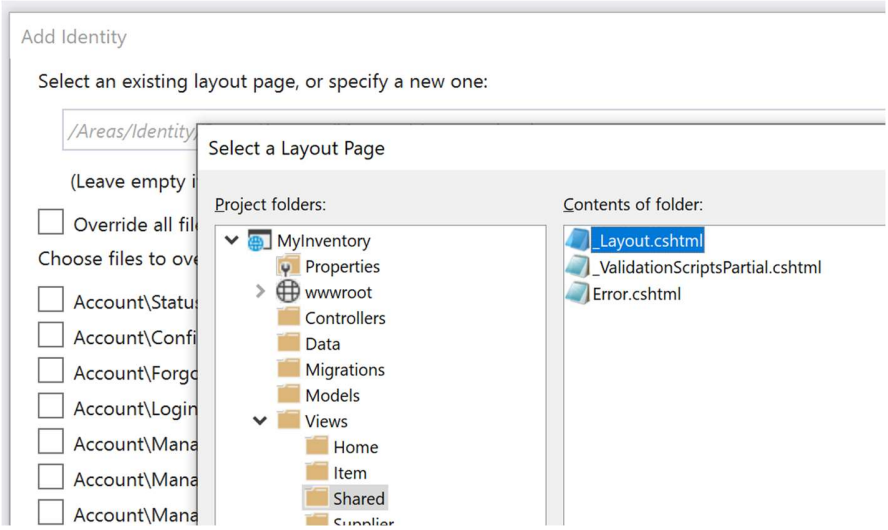
17. From the **Solution Explorer** section, right-click on the project, then select **Add > New Scaffold Item…**



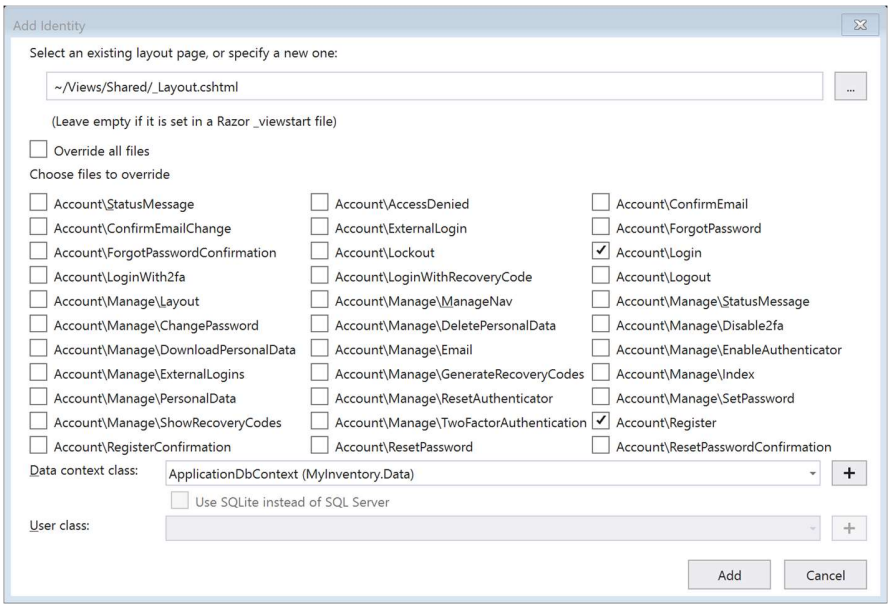

18. From the **Add New Scaffold** Item window, choose the **Identity** page, then select **Identity.** Click the **Add** button to proceed.



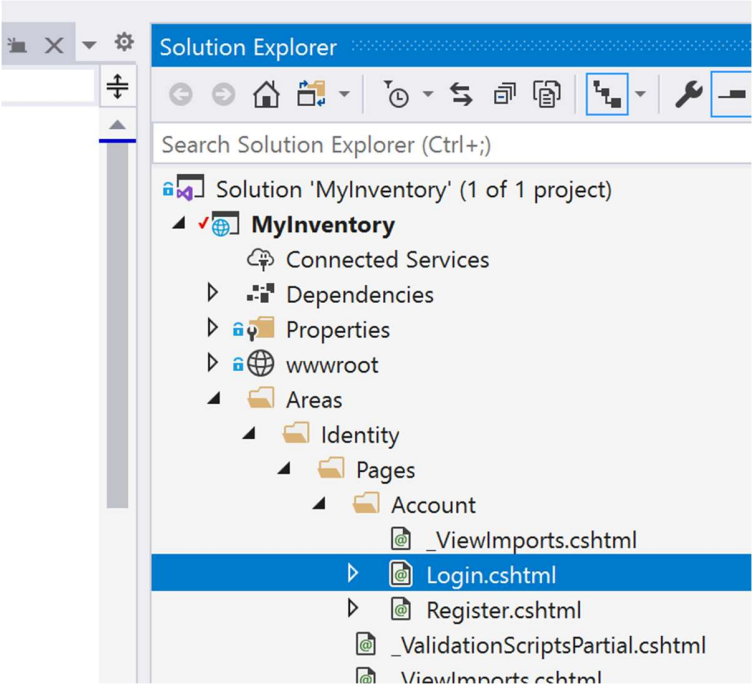

19. Wait for the scaffolding process to load.
20. From the **Add Identity** window, select the existing layout page by clicking the **…** button, then choose **Views > Shared > _Layout.cshtml**. Click the **OK** button to proceed.




21. Check **Account\Login** and **Account\Register**. Choose **ApplicationDbContext** as the data context class.




22. Click the **Add** button to proceed.
23. The Login and Register page are now automatically created during the scaffolding process.




24. From the **Solution Explorer** section, open **Startup.cs**.
25. Inside the scope of the **ConfigureServices** method, invoke the **AddDefaultIdentity** and **AddRazorPages** methods. Ensure that the namespaces **MyInventory.Models** and **Microsoft.AspNetCore.Identity** are added.

```
// This method gets called by the runtime. Use this method to add services to the container.
public void ConfigureServices(IServiceCollection services)
{
    services.AddDbContext<ApplicationDbContext>(option =>
        option.UseSqlServer(Configuration.GetConnectionString("MyConnection")));

    services.AddDefaultIdentity<IdentityUser>().AddEntityFrameworkStores<ApplicationDbContext>();

    services.AddControllersWithViews();
    services.AddRazorPages();
}
```

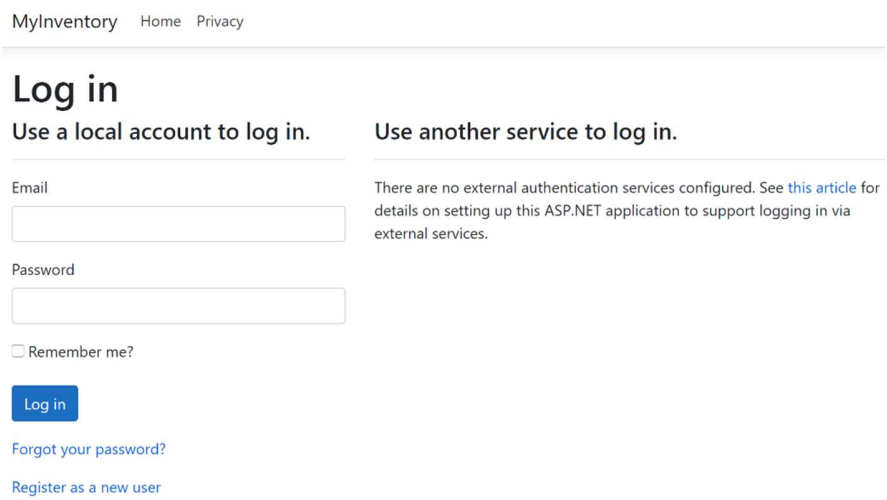
26. Inside the scope of the **Configure** method, invoke the **UseAuthentication** method.

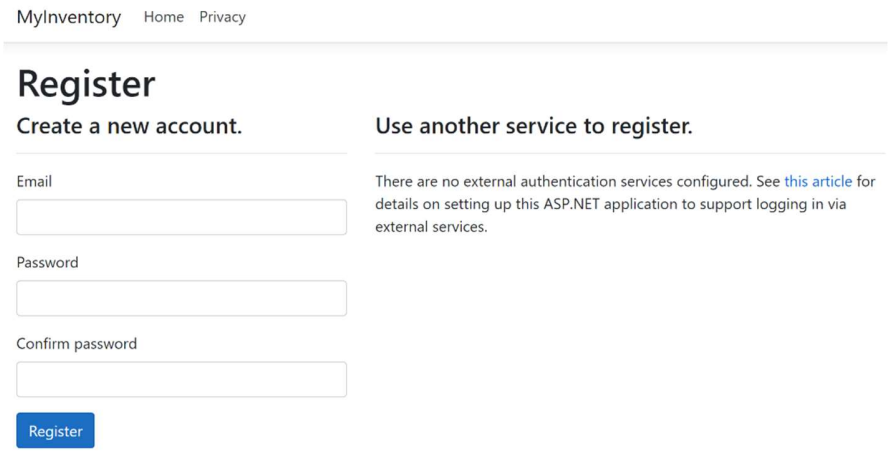
```
// This method gets called by the runtime. Use this me
public void Configure(IApplicationBuilder app, IWebHos
{
    if (env.IsDevelopment())
    {
        app.UseDeveloperExceptionPage();
    }
    else
    {
        app.UseExceptionHandler("/Home/Error");
        // The default HSTS value is 30 days. You may
        app.UseHsts();
    }
    app.UseHttpsRedirection();
    app.UseStaticFiles();

    app.UseRouting();

    app.UseAuthorization();
```

27. Include **MapRazorPages** method from the site's endpoints.

```
// This method gets called by the runtime. Use this method to confi
public void Configure(IApplicationBuilder app, IWebHostEnvironment
{
    if (env.IsDevelopment())
    {
        app.UseDeveloperExceptionPage();
    }
    else
    {
        app.UseExceptionHandler("/Home/Error");
        // The default HSTS value is 30 days. You may want to chang
        app.UseHsts();
    }
    app.UseHttpsRedirection();
    app.UseStaticFiles();

    app.UseRouting();

    app.UseAuthorization();

    app.UseEndpoints(endpoints =>
    {
        endpoints.MapControllerRoute(
            name: "default",
            pattern: "{controller=Home}/{action=Index}/{id?}");
        endpoints.MapRazorPages();
    });
}
```

28. Save all the files. Build the solution.
29. Access the **Login** page using **https://localhost:<port number>/Identity/Account/Login** and the **Register** page using **https://localhost:<port number>/Identity/Account/Register**

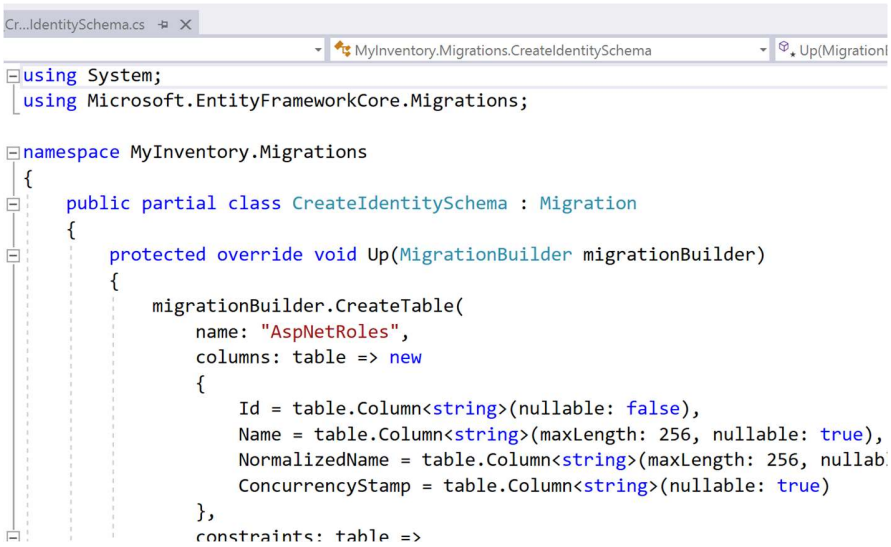




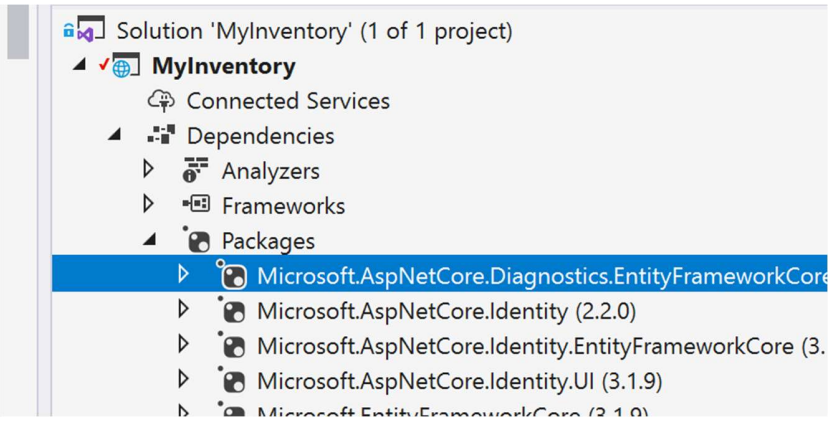

30. From **Microsoft Visual Studio**, open the **Package Manager Console** (Tools > NuGet Package Manager > Package Manager Console)
31. Run the following commands:

    Install-Package Microsoft.AspNetCore.Diagnostics.EntityFrameworkCore
    Add-Migration CreateIdentitySchema
    Update-Database

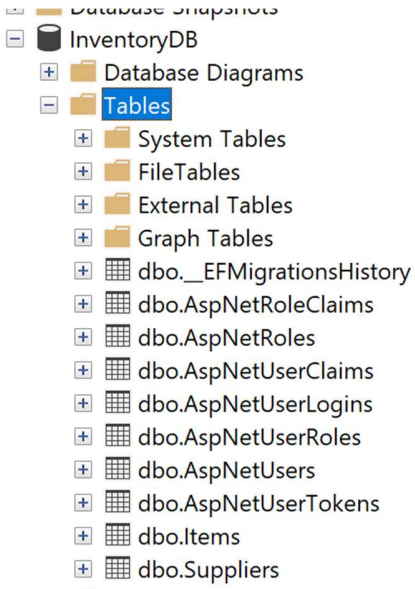
32. A migration file will be generated after the commands have been completed.




33. The package **Microsoft.AspNetCore.Diagnostics.EntityFrameworkCore** will also be installed from the project.




34. From the existing **InventoryDB** database, newly-created database tables can be seen:




35. You can now register and login accounts.