EU General Data Protection Regulation (GDPR) support in ASP.NET Core

07/11/2019 • 7 minutes to read • +8

In this article

By Rick Anderson

ASP.NET Core provides APIs and templates to help meet some of the EU General Data Protection Regulation (GDPR) requirements:

- The project templates include extension points and stubbed markup that you can replace with your privacy and cookie use policy.
- A cookie consent feature allows you to ask for (and track) consent from your users for storing personal information. If a user hasn't consented to data collection and the app has CheckConsentNeeded set to `true`, non-essential cookies aren't sent to the browser.
- Cookies can be marked as essential. Essential cookies are sent to the browser even when the user hasn't consented and tracking is disabled.
- TempData and Session cookies aren't functional when tracking is disabled.
- The Identity manage page provides a link to download and delete user data.

The sample app allows you test most of the GDPR extension points and APIs added to the ASP.NET Core 2.1 templates. See the ReadMe file for testing instructions.

View or download sample code (how to download)

ASP.NET Core GDPR support in template-generated code

Razor Pages and MVC projects created with the project templates include the following GDPR support:

- CookiePolicyOptions and UseCookiePolicy are set in the `Startup` class.
- The *_CookieConsentPartial.cshtml* partial view. An **Accept** button is included in this file. When the user clicks the **Accept** button, consent to store cookies is provided.
- The *Pages/Privacy.cshtml* page or *Views/Home/Privacy.cshtml* view provides a page to detail your site's privacy policy. The *_CookieConsentPartial.cshtml* file generates a link to the Privacy page.
- For apps created with individual user accounts, the Manage page provides links to download and delete personal user data.

CookiePolicyOptions and UseCookiePolicy

CookiePolicyOptions are initialized in `Startup.ConfigureServices`:

C#

Copy

```
public class Startup
{
    public Startup(IConfiguration configuration)
    {
        Configuration = configuration;
    }

    public IConfiguration Configuration { get; }

    // This method gets called by the runtime. Use this method to add services
    // to the container.
    public void ConfigureServices(IServiceCollection services)
    {
        services.Configure<CookiePolicyOptions>(options =>
        {
            // This lambda determines whether user consent for non-essential cookies
            // is needed for a given request.
            options.CheckConsentNeeded = context => true;
            options.MinimumSameSitePolicy = SameSiteMode.None;
        });

        services.AddDbContext<ApplicationDbContext>(options =>
            options.UseSqlServer(
Configuration.GetConnectionString("DefaultConnection")));
        services.AddDefaultIdentity<IdentityUser>()
            .AddEntityFrameworkStores<ApplicationDbContext>();

        // If the app uses session state, call AddSession.
        // services.AddSession();
services.AddMvc().SetCompatibilityVersion(CompatibilityVersion.Version_
2_1);
    }

    // This method gets called by the runtime. Use this method to con-
figure the
    // HTTP request pipeline.
    public void Configure(IApplicationBuilder app, IHostingEnvironment
env)
    {
        if (env.IsDevelopment())
        {
            app.UseDeveloperExceptionPage();
            app.UseDatabaseErrorPage();
        }
        else
        {
            app.UseExceptionHandler("/Error");
            app.UseHsts();
        }

        app.UseHttpsRedirection();
        app.UseStaticFiles();
        app.UseCookiePolicy();

        app.UseAuthentication();

        // If the app uses session state, call Session Middleware after
Cookie
        // Policy Middleware and before MVC Middleware.
        // app.UseSession();

        app.UseMvc();
    }
}
```

UseCookiePolicy is called in `Startup.Configure`:

C# Copy

```
public class Startup
{
    public Startup(IConfiguration configuration)
    {
        Configuration = configuration;
    }

    public IConfiguration Configuration { get; }
    // This method gets called by the runtime. Use this method to add
services
    // to the container.
    public void ConfigureServices(IServiceCollection services)
    {
        services.Configure<CookiePolicyOptions>(options =>
        {
            // This lambda determines whether user consent for non-es-
sential cookies
            // is needed for a given request.
            options.CheckConsentNeeded = context => true;
            options.MinimumSameSitePolicy = SameSiteMode.None;
        });

        services.AddDbContext<ApplicationDbContext>(options =>
            options.UseSqlServer(

Configuration.GetConnectionString("DefaultConnection")));
        services.AddDefaultIdentity<IdentityUser>()
            .AddEntityFrameworkStores<ApplicationDbContext>();

        // If the app uses session state, call AddSession.
        // services.AddSession();


services.AddMvc().SetCompatibilityVersion(CompatibilityVersion.Version_
2_1);
    }

    // This method gets called by the runtime. Use this method to con-
figure the
    // HTTP request pipeline.
    public void Configure(IApplicationBuilder app, IHostingEnvironment
env)
    {
        if (env.IsDevelopment())
        {
            app.UseDeveloperExceptionPage();
            app.UseDatabaseErrorPage();
        }
        else
        {
            app.UseExceptionHandler("/Error");
            app.UseHsts();
        }

        app.UseHttpsRedirection();
        app.UseStaticFiles();
        app.UseCookiePolicy();
        app.UseAuthentication();

        // If the app uses session state, call Session Middleware after Cookie
        // Policy Middleware and before MVC Middleware.
        // app.UseSession();

        app.UseMvc();
    }
}
```

_CookieConsentPartial.cshtml partial view

The *_CookieConsentPartial.cshtml* partial view:

CSHTML

Copy

```
@using Microsoft.AspNetCore.Http.Features

@{
    var consentFeature = Context.Features.Get<ITrackingConsentFeature>();
    var showBanner = !consentFeature?.CanTrack ?? false;
    var cookieString = consentFeature?.CreateConsentCookie();
}

@if (showBanner)
{
    <nav id="cookieConsent" class="navbar navbar-default navbar-fixed-top" role="alert">
        <div class="container">
            <div class="navbar-header">
                <button type="button" class="navbar-toggle" data-toggle="collapse" data-target="#cookieConsent .navbar-collapse">
                    <span class="sr-only">Toggle cookie consent banner</span>
                    <span class="icon-bar"></span>
                    <span class="icon-bar"></span>
                    <span class="icon-bar"></span>
                </button>
                <span class="navbar-brand"><span class="glyphicon glyphicon-info-sign" aria-hidden="true"></span></span>
            </div>
            <div class="collapse navbar-collapse">
                <p class="navbar-text">
                    Use this space to summarize your privacy and cookie use policy.
                </p>
                <div class="navbar-right">
                    <a asp-page="/Privacy" class="btn btn-info navbar-
btn">Learn More</a>
                    <button type="button" class="btn btn-default
navbar-btn" data-cookie-string="@cookieString">Accept</button>
                </div>
            </div>
        </div>
    </nav>
    <script>
        (function () {
            document.querySelector("#cookieConsent button[data-cookie-
string]").addEventListener("click", function (el) {
                document.cookie = el.target.dataset.cookieString;

document.querySelector("#cookieConsent").classList.add("hidden");
            }, false);
        })();
    </script>
}
```

This partial:

- Obtains the state of tracking for the user. If the app is configured to require consent, the user must consent before cookies can be tracked. If consent is required, the cookie consent panel is fixed at top of the navigation bar created by the *_Layout.cshtml* file.
- Provides an HTML `<p>` element to summarize your privacy and cookie use policy.
- Provides a link to Privacy page or view where you can detail your site's privacy policy.

Essential cookies

If consent to store cookies hasn't been provided, only cookies marked essential are sent to the browser. The following code makes a cookie essential:

C#

Copy

```
public IActionResult OnPostCreateEssentialAsync()
{
    HttpContext.Response.Cookies.Append(Constants.EssentialSec,
        DateTime.Now.Second.ToString(),
        new CookieOptions() { IsEssential = true });

    ResponseCookies =
Response.Headers[HeaderNames.SetCookie].ToString();
```

```
    return RedirectToPage("./Index");
}
```

TempData provider and session state cookies aren't essential

The TempData provider cookie isn't essential. If tracking is disabled, the TempData provider isn't functional. To enable the TempData provider when tracking is disabled, mark the TempData cookie as essential in `Startup.ConfigureServices`:

C# Copy

```
// The TempData provider cookie is not essential. Make it essential
// so TempData is functional when tracking is disabled.
services.Configure<CookieTempDataProviderOptions>(options => {
    options.Cookie.IsEssential = true;
});
```

Session state cookies are not essential. Session state isn't functional when tracking is disabled. The following code makes session cookies essential:

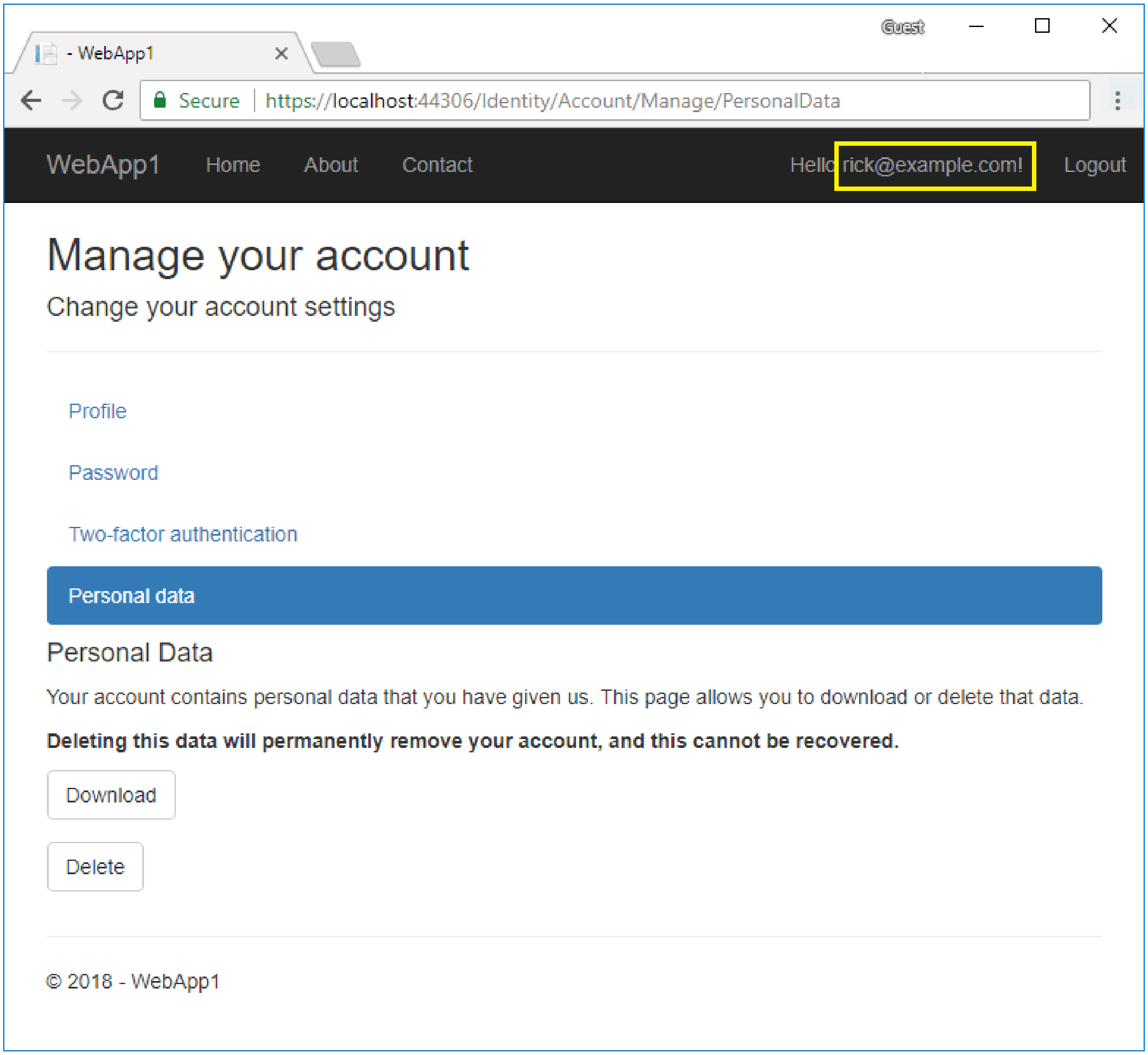
C# Copy

```
services.AddSession(options =>
{
    options.Cookie.IsEssential = true;
});
```

Personal data

ASP.NET Core apps created with individual user accounts include code to download and delete personal data.

Select the user name and then select **Personal data**:



Manage your account

Change your account settings

Personal Data

Your account contains personal data that you have given us. This page allows you to download or delete that data.

Deleting this data will permanently remove your account, and this cannot be recovered.

© 2018 - WebApp1

Notes:

- To generate the `Account/Manage` code, see Scaffold Identity.
- The **Delete** and **Download** links only act on the default identity data. Apps that create custom user data must be extended to delete/download the custom user data. For more information, see Add, download, and delete custom user data to Identity.
- Saved tokens for the user that are stored in the Identity database table `AspNetUserTokens` are deleted when the user is deleted via the cascading delete behavior due to the foreign key.
- External provider authentication, such as Facebook and Google, isn't available before the cookie policy is accepted.

Encryption at rest

Some databases and storage mechanisms allow for encryption at rest. Encryption at rest:

- Encrypts stored data automatically.
- Encrypts without configuration, programming, or other work for the software that accesses the data.
- Is the easiest and safest option.
- Allows the database to manage keys and encryption.

For example:

- Microsoft SQL and Azure SQL provide Transparent Data Encryption (TDE).
- SQL Azure encrypts the database by default
- Azure Blobs, Files, Table, and Queue Storage are encrypted by default.

For databases that don't provide built-in encryption at rest, you may be able to use disk encryption to provide the same protection. For example:

- BitLocker for Windows Server
- Linux:
  - eCryptfs
  - EncFS.

Additional resources

- Microsoft.com/GDPR
- GDPR - Adding a Revoke Consent Button in ASP.NET Core

Is this page helpful?

👍 Yes 👎 No