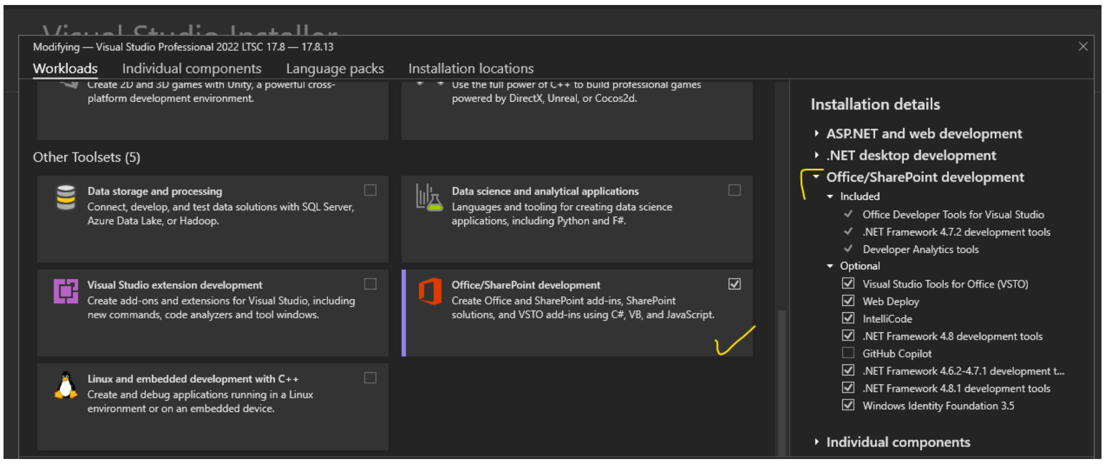
Create BlazorWebAssembly Excel Addin project with TR-Saffron Styling

Pre-requisites

- Visual Studio 2022 Latest version is installed.
- .Net Framework v4.8.1 is installed.
- Following Office/Sharepoint development toolset is installed.



Visual Studio Installer

Modifying — Visual Studio Professional 2022 LTSC 17.8 — 17.8.13

Create 2D and 3D games with Unity, a powerful cross-platform development environment.

Use the full power of C++ to build professional games powered by DirectX, Unreal, or Cocos2d.

Other Toolsets (5)

☐ **Data storage and processing**
Connect, develop, and test data solutions with SQL Server, Azure Data Lake, or Hadoop.

☐ **Data science and analytical applications**
Languages and tooling for creating data science applications, including Python and F#.

☐ **Visual Studio extension development**
Create add-ons and extensions for Visual Studio, including new commands, code analyzers and tool windows.

☑ **Office/SharePoint development**
Create Office and SharePoint add-ins, SharePoint solutions, and VSTO add-ins using C#, VB, and JavaScript.

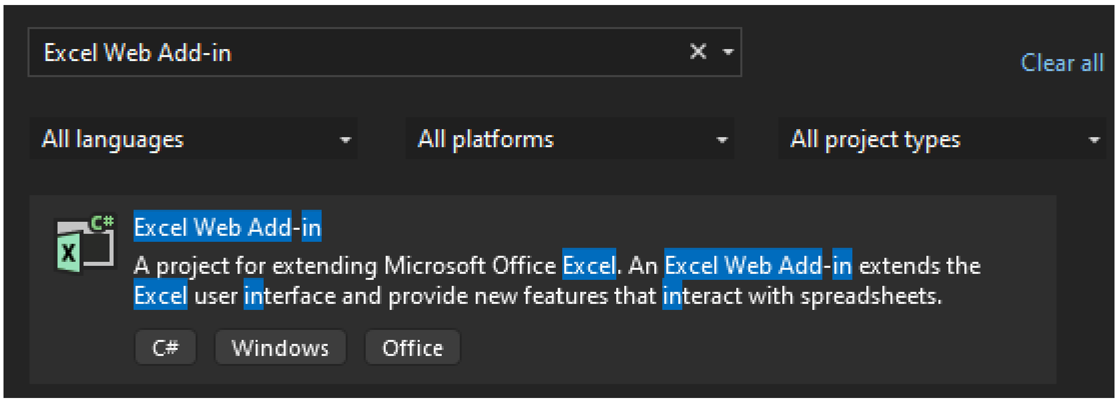
☐ **Linux and embedded development with C++**
Create and debug applications running in a Linux environment or on an embedded device.

Installation details

- ASP.NET and web development
- .NET desktop development
- Office/SharePoint development
  - Included
    - ✓ Office Developer Tools for Visual Studio
    - ✓ .NET Framework 4.7.2 development tools
    - ✓ Developer Analytics tools
  - Optional
    - ☑ Visual Studio Tools for Office (VSTO)
    - ☑ Web Deploy
    - ☑ IntelliCode
    - ☑ .NET Framework 4.8 development tools
    - ☐ GitHub Copilot
    - ☑ .NET Framework 4.6.2-4.7.1 development t...
    - ☑ .NET Framework 4.8.1 development tools
    - ☑ Windows Identity Foundation 3.5
- Individual components

Creating Excel Add-In project

- In VS 2022, create a new project.
- In search box, enter Excel Web Add-in and select that project and click on Next.



Excel Web Add-in

Clear all

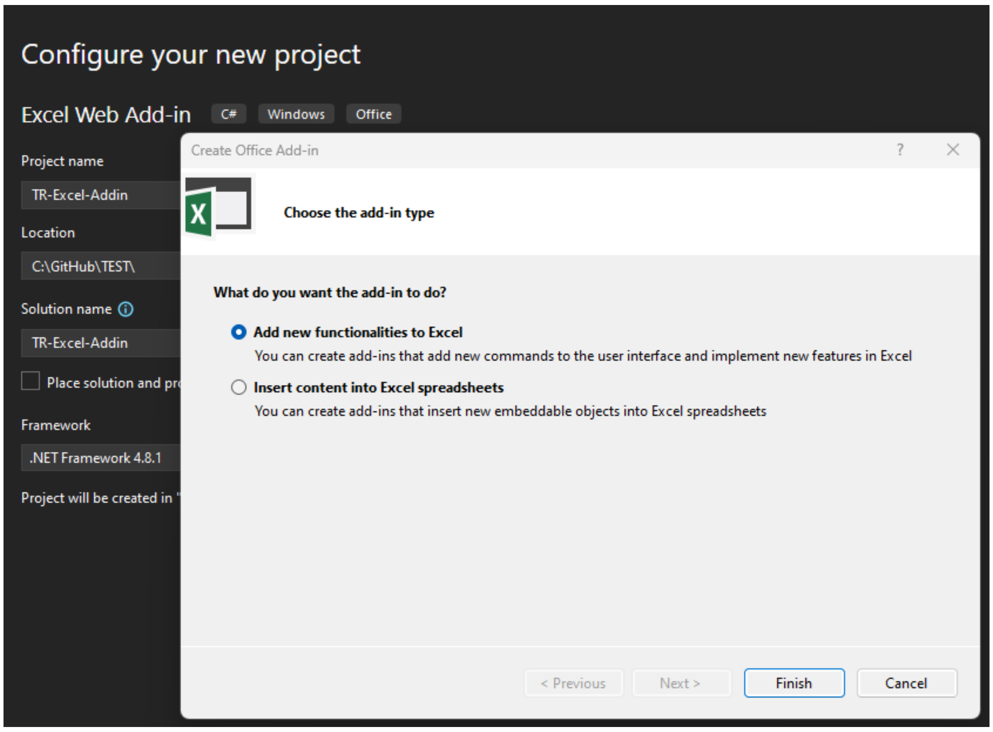
All languages | All platforms | All project types

Excel Web Add-in

A project for extending Microsoft Office Excel. An Excel Web Add-in extends the Excel user interface and provide new features that interact with spreadsheets.

C# Windows Office

- Give any project name, ex: "**TR-Excel-Addin**" and select .Net Framework 4.8.1 under Framework and click on Create.
- In the popup window, choose the add-in type as "Add new functionalities to Excel" and click on Finish



Configure your new project

Excel Web Add-in C# Windows Office

Project name: TR-Excel-Addin

Location: C:\GitHub\TEST\

Solution name: TR-Excel-Addin

Place solution and pr

Framework: .NET Framework 4.8.1

Project will be created in "

Create Office Add-in

Choose the add-in type

What do you want the add-in to do?

● Add new functionalities to Excel
You can create add-ins that add new commands to the user interface and implement new features in Excel

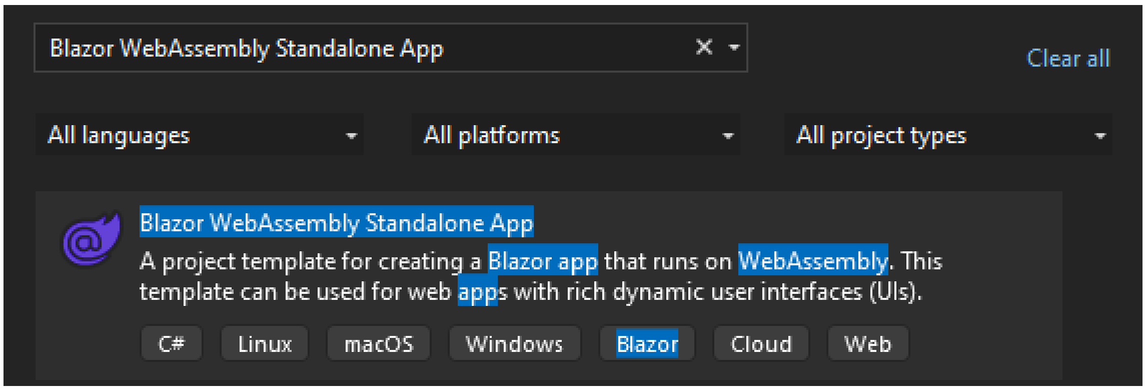
○ Insert content into Excel spreadsheets
You can create add-ins that insert new embeddable objects into Excel spreadsheets

< Previous | Next > | Finish | Cancel

- Visual Studio creates a solution, and its two projects appear in Solution Explorer.
- The Home.html file opens in Visual Studio.

Creating Blazor Web Assembly project

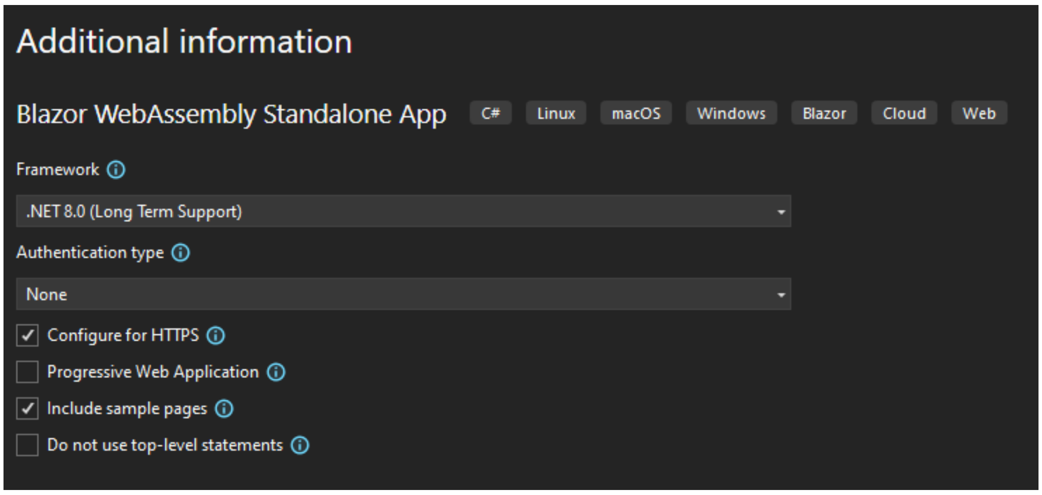
- In VS 2022, create a new project
- In search box type: Blazor WebAssembly Standalone App and select that project and click on Next



Blazor WebAssembly Standalone App

Blazor WebAssembly Standalone App
A project template for creating a Blazor app that runs on WebAssembly. This template can be used for web apps with rich dynamic user interfaces (UIs).

- Give any project name, for ex: "**TR-Blazor-WebUI**" and click on Next
- Use .Net 8.0 framework and other options as follows and click on Create

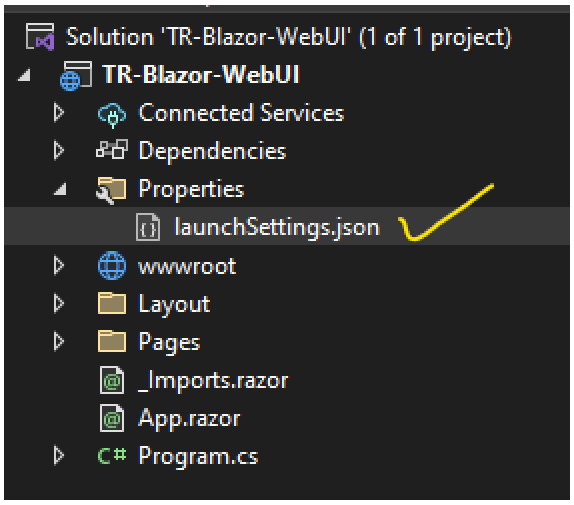


Additional information

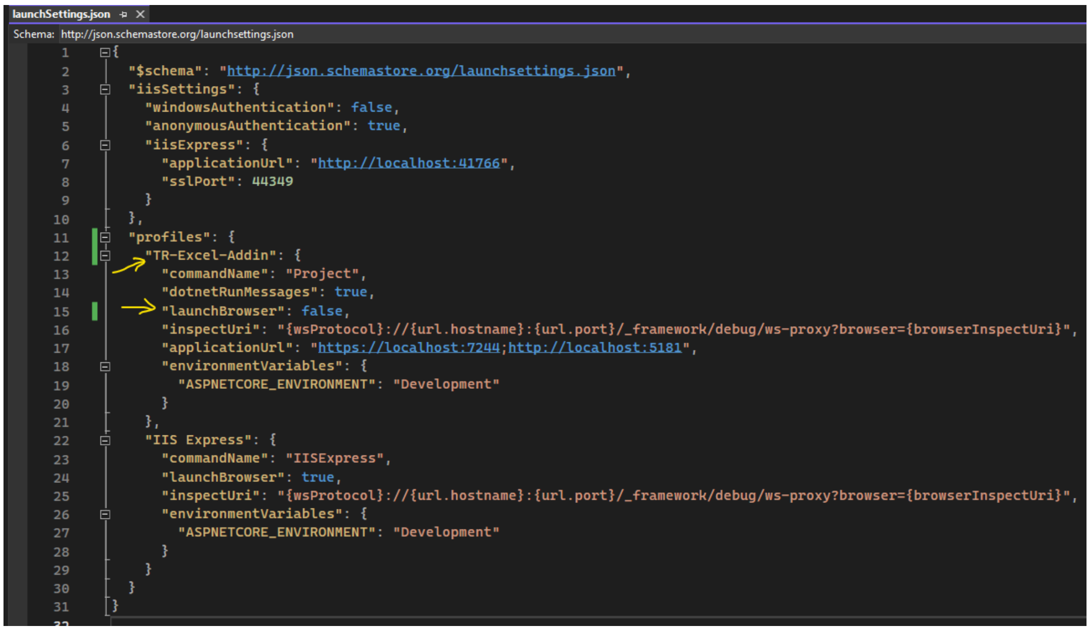
Blazor WebAssembly Standalone App

Framework
.NET 8.0 (Long Term Support)
Authentication type
None
☑ Configure for HTTPS
☐ Progressive Web Application
☑ Include sample pages
☐ Do not use top-level statements

- This project creates several razor pages out of the box which are ready to run in Web.
- Inside this TR-Blazor-WebUI project, open launchSettings.json file from Properties folder



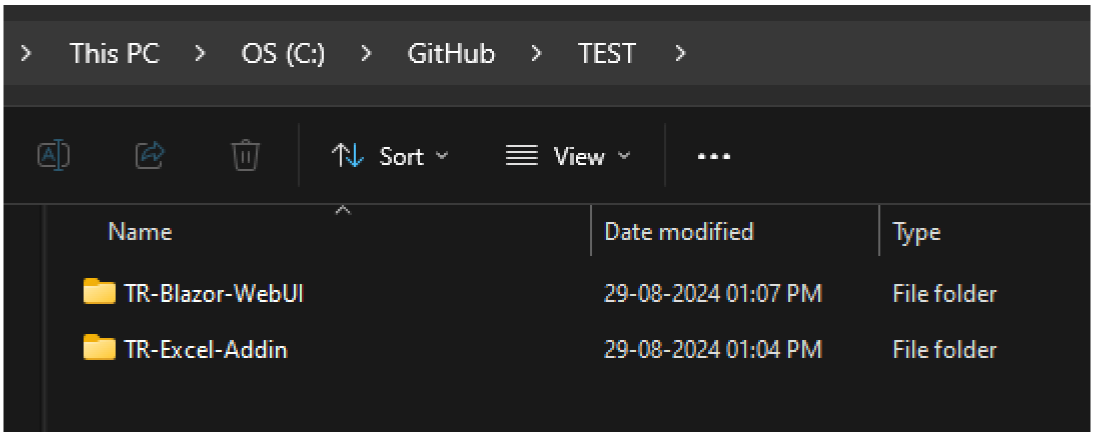
- In the profiles section, remove http profile and rename “**https**” profile name to “**TR-Excel-Addin**” and make launchBrowser to **false**



```json
{
  "$schema": "http://json.schemastore.org/launchsettings.json",
  "iisSettings": {
    "windowsAuthentication": false,
    "anonymousAuthentication": true,
    "iisExpress": {
      "applicationUrl": "http://localhost:41766",
      "sslPort": 44349
    }
  },
  "profiles": {
    "TR-Excel-Addin": {
      "commandName": "Project",
      "dotnetRunMessages": true,
      "launchBrowser": false,
      "inspectUri": "{wsProtocol}://{url.hostname}:{url.port}/_framework/debug/ws-proxy?browser={browserInspectUri}",
      "applicationUrl": "https://localhost:7244;http://localhost:5181",
      "environmentVariables": {
        "ASPNETCORE_ENVIRONMENT": "Development"
      }
    },
    "IIS Express": {
      "commandName": "IISExpress",
      "launchBrowser": true,
      "inspectUri": "{wsProtocol}://{url.hostname}:{url.port}/_framework/debug/ws-proxy?browser={browserInspectUri}",
      "environmentVariables": {
        "ASPNETCORE_ENVIRONMENT": "Development"
      }
    }
  }
}
```

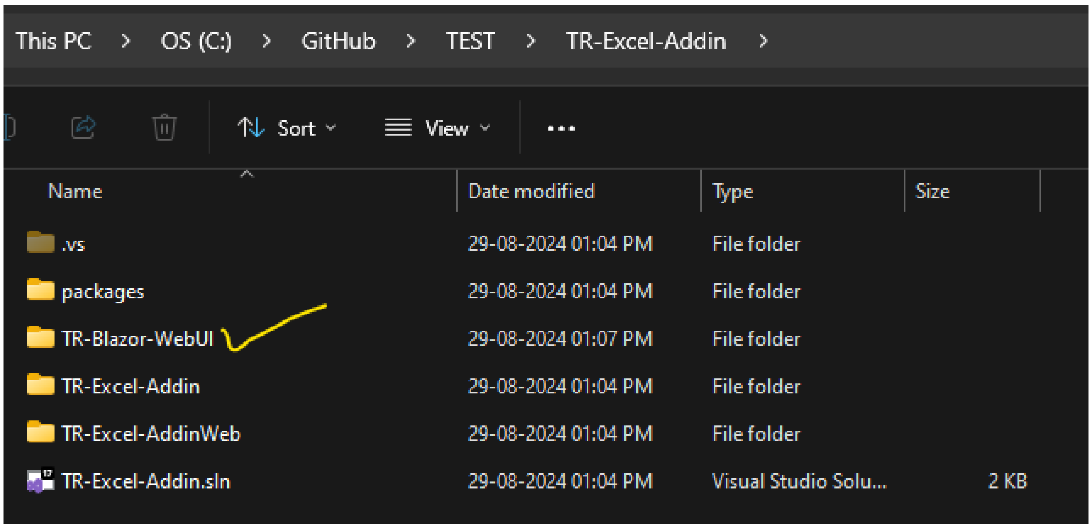
Integrating TR-Excel-Addin project with TR-Blazor-WebUI project

- Close both the Visual Studio projects (TR-Excel-Addin and TR-Blazor-WebUI)
- Go to the path where both the projects are created in File explorer



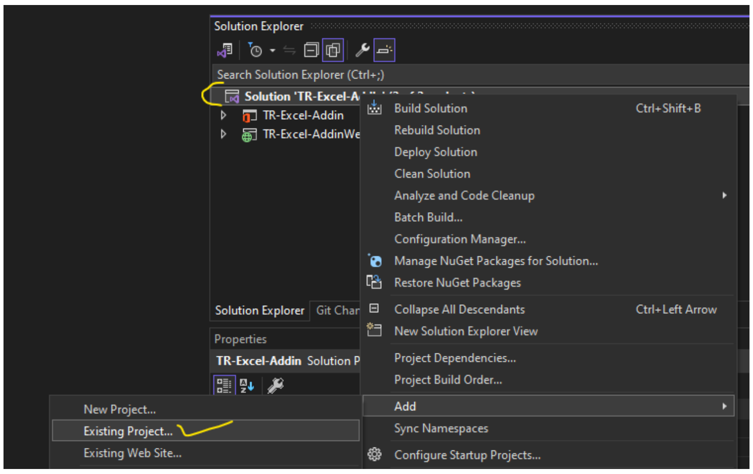
Name	Date modified	Type
TR-Blazor-WebUI	29-08-2024 01:07 PM	File folder
TR-Excel-Addin	29-08-2024 01:04 PM	File folder

- From this path, move the "TR-Blazor-WebUI" project folder inside the "TR-Excel-Addin" project folder as follows:

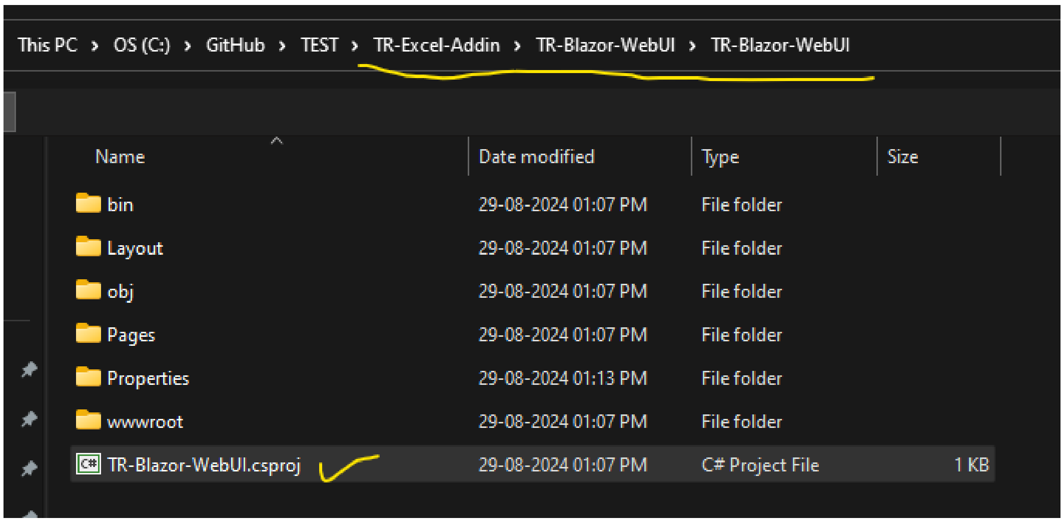


Name	Date modified	Type	Size
.vs	29-08-2024 01:04 PM	File folder	
packages	29-08-2024 01:04 PM	File folder	
TR-Blazor-WebUI	29-08-2024 01:07 PM	File folder	
TR-Excel-Addin	29-08-2024 01:04 PM	File folder	
TR-Excel-AddinWeb	29-08-2024 01:04 PM	File folder	
TR-Excel-Addin.sln	29-08-2024 01:04 PM	Visual Studio Solu...	2 KB

- Now open "**TR-Excel-Addin.sln**" file in VS 2022
- Now right click the main solution and select Add >> Existing Project



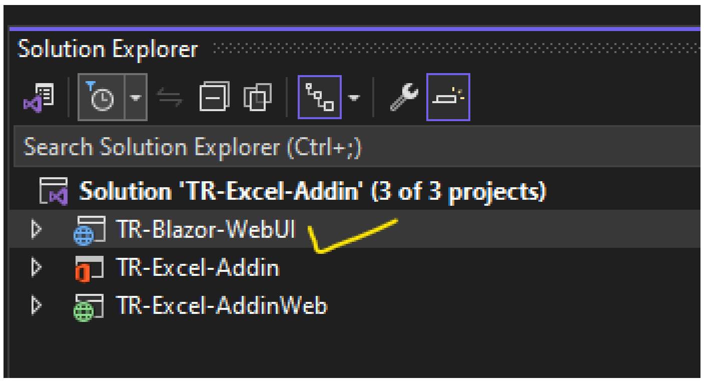
- In the File explorer, go to the path where we moved our “TR-Blazor-WebUI” project folder and select the **csproj** file of this project



This PC › OS (C:) › GitHub › TEST › TR-Excel-Addin › TR-Blazor-WebUI › TR-Blazor-WebUI

Name	Date modified	Type	Size
bin	29-08-2024 01:07 PM	File folder	
Layout	29-08-2024 01:07 PM	File folder	
obj	29-08-2024 01:07 PM	File folder	
Pages	29-08-2024 01:07 PM	File folder	
Properties	29-08-2024 01:13 PM	File folder	
wwwroot	29-08-2024 01:07 PM	File folder	
TR-Blazor-WebUI.csproj	29-08-2024 01:07 PM	C# Project File	1 KB

- The TR-Excel-Addin solution will now add this TR-Blazor-WebUI project

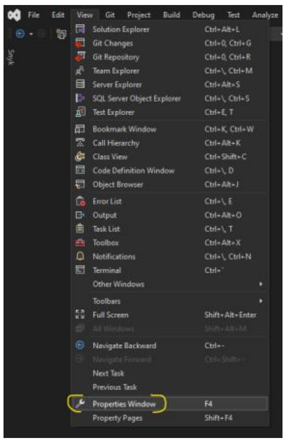


Solution Explorer

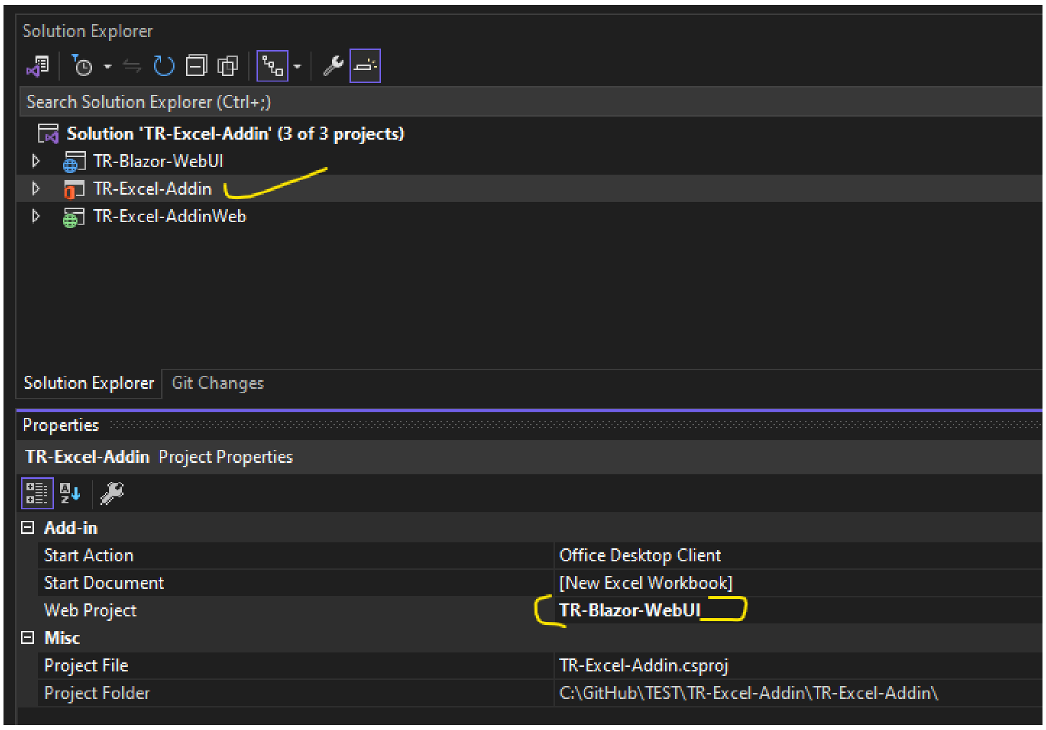
Search Solution Explorer (Ctrl+;)

Solution 'TR-Excel-Addin' (3 of 3 projects)
- TR-Blazor-WebUI
- TR-Excel-Addin
- TR-Excel-AddinWeb

- Open View >> **Properties Window** or press F4


File Edit View Git Project Build Debug Test Analyze
Snyk
Solution Explorer Ctrl+Alt+L
Git Changes Ctrl+0, Ctrl+G
Git Repository Ctrl+0, Ctrl+R
Team Explorer Ctrl+\, Ctrl+M
Server Explorer Ctrl+Alt+S
SQL Server Object Explorer Ctrl+\, Ctrl+S
Test Explorer Ctrl+E, T
Bookmark Window Ctrl+K, Ctrl+W
Call Hierarchy Ctrl+Alt+K
Class View Ctrl+Shift+C
Code Definition Window Ctrl+\, D
Object Browser Ctrl+Alt+J
Error List Ctrl+\, E
Output Ctrl+Alt+O
Task List Ctrl+\, T
Toolbox Ctrl+Alt+X
Notifications Ctrl+\, Ctrl+N
Terminal Ctrl+`
Other Windows
Toolbars
Full Screen Shift+Alt+Enter
All Windows Shift+Alt+M
Navigate Backward Ctrl+-
Navigate Forward Ctrl+Shift+-
Next Task
Previous Task
Properties Window F4
Property Pages Shift+F4

- Now select **TR-Excel-Addin** project from the solution.
- In the Properties window, under **Add-in >> Web Project**, change the project from **“TR-Excel-AddinWeb”** to **“TR-Blazor-WebUI”**
- On the confirmation popup, select Yes.


Solution Explorer

Search Solution Explorer (Ctrl+;)

Solution 'TR-Excel-Addin' (3 of 3 projects)
- TR-Blazor-WebUI
- TR-Excel-Addin
- TR-Excel-AddinWeb

Solution Explorer | Git Changes

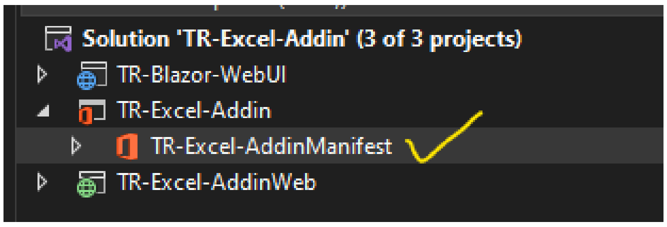
Properties

TR-Excel-Addin Project Properties

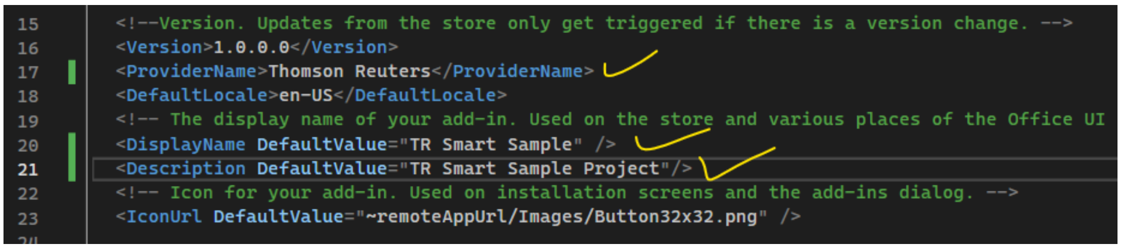
Add-in	
Start Action	Office Desktop Client
Start Document	[New Excel Workbook]
Web Project	**TR-Blazor-WebUI**
Misc	
Project File	TR-Excel-Addin.csproj
Project Folder	C:\GitHub\TEST\TR-Excel-Addin\TR-Excel-Addin\

Configuring TR-Excel-Addin project’s Manifest

- Double click on **TR-Excel-AddinManifest** file

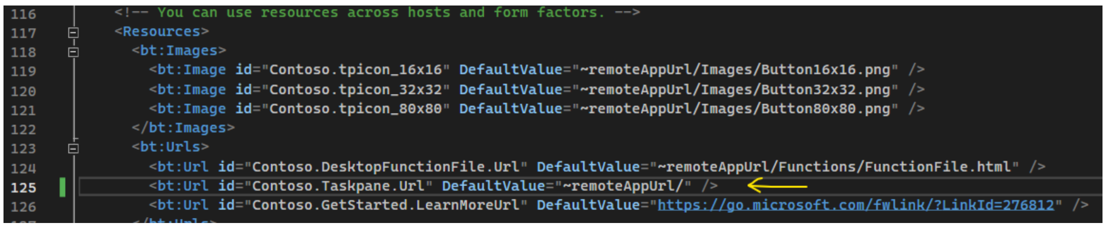


- The manifest xml file will be opened in the VS where we will configure some changes.
- Rename some default values as follows



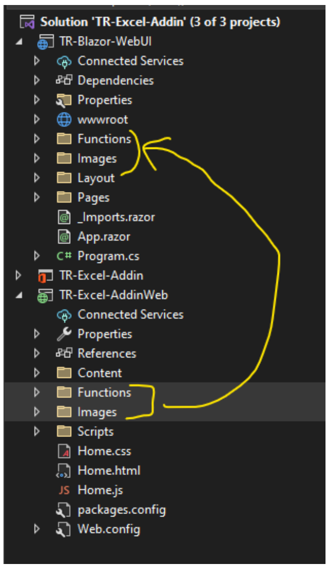
```xml
<!--Version. Updates from the store only get triggered if there is a version change. -->
<Version>1.0.0.0</Version>
<ProviderName>Thomson Reuters</ProviderName>
<DefaultLocale>en-US</DefaultLocale>
<!-- The display name of your add-in. Used on the store and various places of the Office UI
<DisplayName DefaultValue="TR Smart Sample" />
<Description DefaultValue="TR Smart Sample Project"/>
<!-- Icon for your add-in. Used on installation screens and the add-ins dialog. -->
<IconUrl DefaultValue="~remoteAppUrl/Images/Button32x32.png" />
```

- Scroll down to the bottom of this xml file and remove **Home.html** from **DefaultValue="~remoteAppUrl/Home.html"** so that the Excel addin pane will point to the default page for “**TR-Blazor-WebUI**” project
- Save the file and close it.

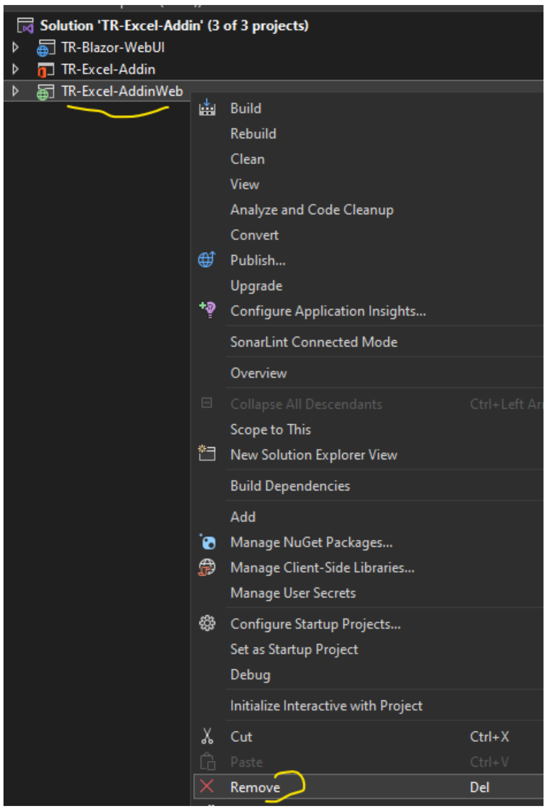


```xml
<!-- You can use resources across hosts and form factors. -->
<Resources>
  <bt:Images>
    <bt:Image id="Contoso.tpicon_16x16" DefaultValue="~remoteAppUrl/Images/Button16x16.png" />
    <bt:Image id="Contoso.tpicon_32x32" DefaultValue="~remoteAppUrl/Images/Button32x32.png" />
    <bt:Image id="Contoso.tpicon_80x80" DefaultValue="~remoteAppUrl/Images/Button80x80.png" />
  </bt:Images>
  <bt:Urls>
    <bt:Url id="Contoso.DesktopFunctionFile.Url" DefaultValue="~remoteAppUrl/Functions/FunctionFile.html" />
    <bt:Url id="Contoso.Taskpane.Url" DefaultValue="~remoteAppUrl/" />
    <bt:Url id="Contoso.GetStarted.LearnMoreUrl" DefaultValue="https://go.microsoft.com/fwlink/?LinkId=276812" />
```

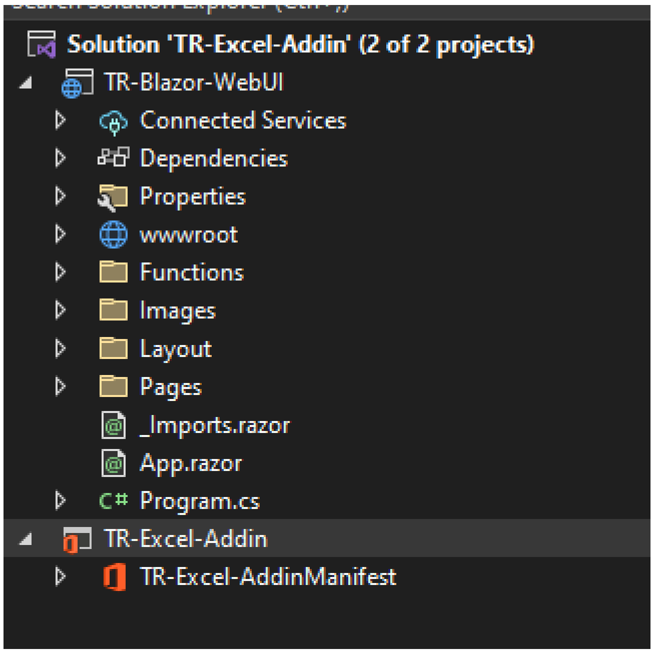
- We can also copy “**Images**” and “**Functions**” folders from **TR-Excel-AddinWeb** project to **TR-Blazor-WebUI** project, as those are referenced in the Manifest xml file.



- At this point, it is now safe to completely **DELETE (Remove)** the "***TR-Excel-AddinWeb***" project from the solution, as **TR-Excel-Addin** is now associated with **TR-Blazor-WebUI** project.

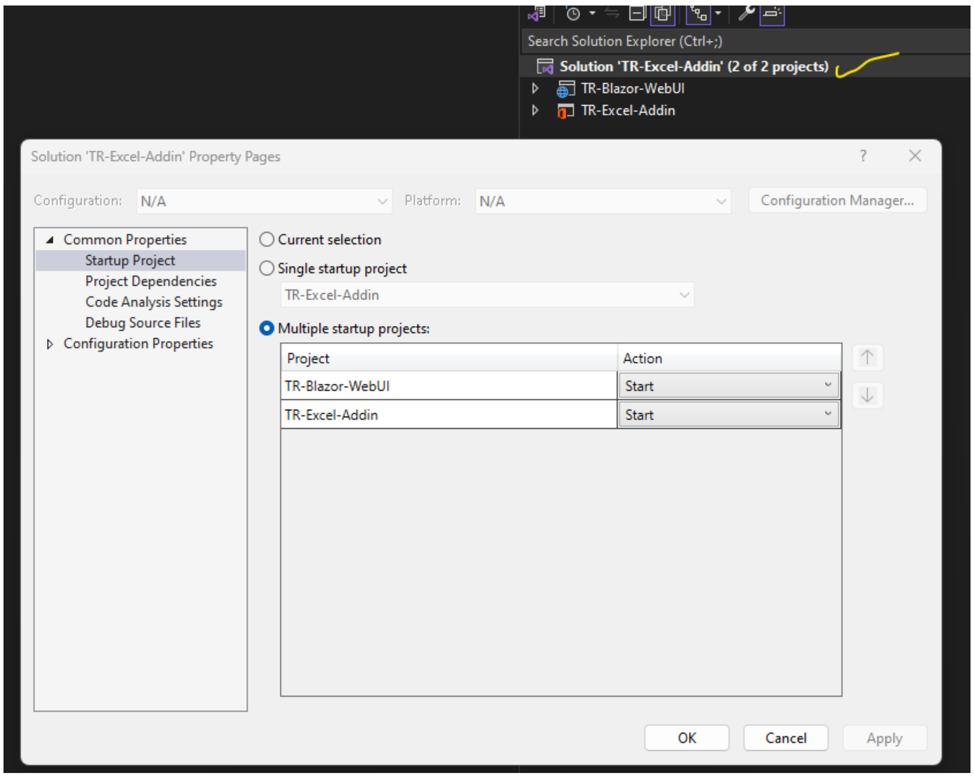


- So, the solution should now look as follows, after the ***TR-Excel-AddinWeb*** deletion



- Also, DELETE the TR-Excel-AddinWeb folder from the file explorer as well
- Right click the main solution and click Properties.

- Inside the Startup Project, make sure: "**Multiple startup projects**" radio button is selected in the following way:



Search Solution Explorer (Ctrl+;)

Solution 'TR-Excel-Addin' (2 of 2 projects)

TR-Blazor-WebUI

TR-Excel-Addin

Solution 'TR-Excel-Addin' Property Pages

Configuration: N/A

Platform: N/A

Configuration Manager...

Common Properties

Startup Project

Project Dependencies

Code Analysis Settings

Debug Source Files

Configuration Properties

( ) Current selection

( ) Single startup project

TR-Excel-Addin

(●) Multiple startup projects:

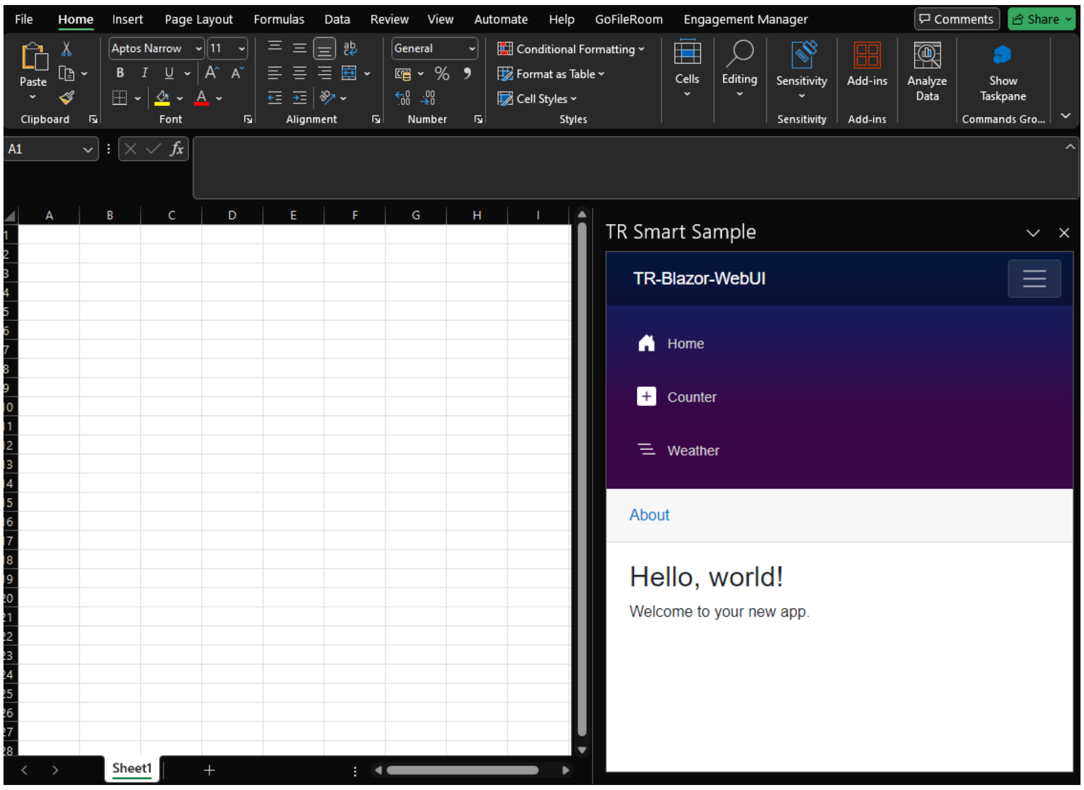
Project	Action
TR-Blazor-WebUI	Start
TR-Excel-Addin	Start

OK Cancel Apply

- Click Apply and Ok.

Running the project

- Run the solution and you'll notice the Excel will open with a **Taskpane** in the right side.
- The Taskpane will have the default page "**Home.razor**", which you can change from the DDMenu from the Navbar



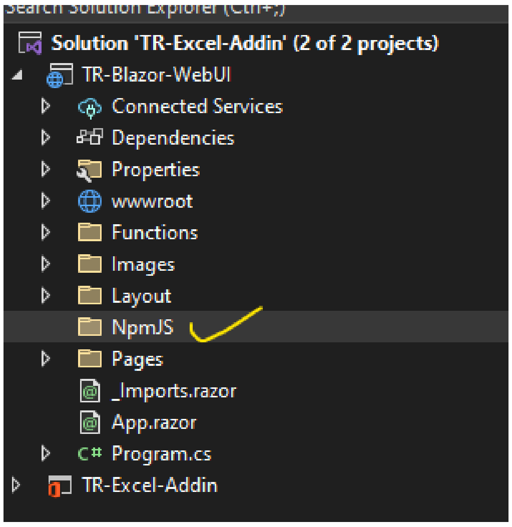
Hello, world!

Welcome to your new app.

- We are ready with our Blazor WebAssembly Project associated with Excel Add-in.

Integrating TR Saffron styling to the Blazor project

- Create **NpmJS** folder under main root of the **TR-Blazor-WebUI** project

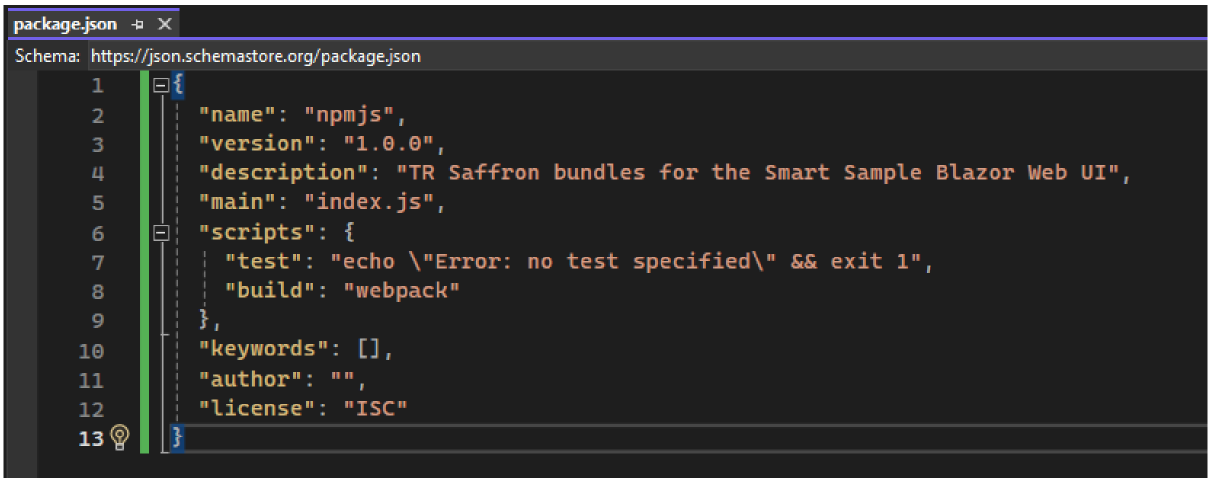


- **Make sure you have the login username and auth-token of tr-jfrog artifactory.**
- Create a new "**.npmrc**" file inside NpmJS folder and add the following snippet.
- Replace the **email** and **authToken** values with your respective Jfrog creds.

```
registry=https://tr1.jfrog.io/tr1/api/npm/npm/
always-auth=true
email=myemail@thomsonreuters.com
//tr1.jfrog.io/tr1/api/npm/npm/:_authToken=abcde
```

- Save and close the file.
- Add **package.json** file with following contents inside NpmJS folder.

```
{
 "name": "npmjs",
 "version": "1.0.0",
 "description": "TR Saffron bundles for the Smart Sample Blazor Web UI",
 "main": "index.js",
 "scripts": {
  "test": "echo \"Error: no test specified\" && exit 1",
  "build": "webpack"
 },
 "keywords": [],
 "author": "",
 "license": "ISC"
}
```



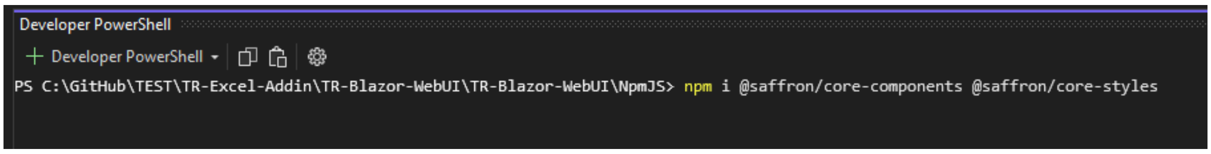
package.json

Schema: https://json.schemastore.org/package.json

```json
{
  "name": "npmjs",
  "version": "1.0.0",
  "description": "TR Saffron bundles for the Smart Sample Blazor Web UI",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1",
    "build": "webpack"
  },
  "keywords": [],
  "author": "",
  "license": "ISC"
}
```

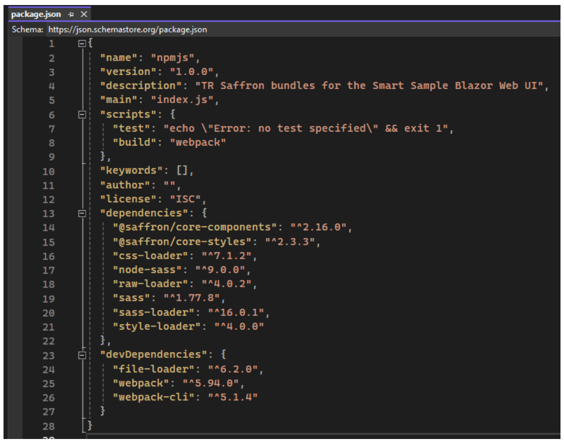
- Click on View >> Terminal from VS menu bar.
- Navigate to the **NpmJS path** and run following 3 commands separately.

```
npm i @saffron/core-components @saffron/core-styles
npm i css-loader node-sass raw-loader sass sass-loader style-loader
npm i file-loader webpack webpack-cli --save-dev
```



```
PS C:\GitHub\TEST\TR-Excel-Addin\TR-Blazor-WebUI\TR-Blazor-WebUI\NpmJS> npm i @saffron/core-components @saffron/core-styles
```

- Finally, your package.json should look like this



package.json

Schema: https://json.schemastore.org/package.json

```json
{
  "name": "npmjs",
  "version": "1.0.0",
  "description": "TR Saffron bundles for the Smart Sample Blazor Web UI",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1",
    "build": "webpack"
  },
  "keywords": [],
  "author": "",
  "license": "ISC",
  "dependencies": {
    "@saffron/core-components": "^2.16.0",
    "@saffron/core-styles": "^2.3.3",
    "css-loader": "^7.1.2",
    "node-sass": "^9.0.0",
    "raw-loader": "^4.0.2",
    "sass": "^1.77.8",
    "sass-loader": "^16.0.1",
    "style-loader": "^4.0.0"
  },
  "devDependencies": {
    "file-loader": "^6.2.0",
    "webpack": "^5.94.0",
    "webpack-cli": "^5.1.4"
  }
}
```

- Add **webpack.config.js** file under NpmJS folder with following content

```js
const path = require("path");
module.exports = {
  module: {
    rules: [
      {
        test: /\.js$/,
      },
            {
                test: /\.css$/,
                use: ["style-loader", "css-loader"]
            },
            {
                test: /\.s[ac]ss$/i,
                use: ["style-loader", "css-loader", "sass-loader",],
            }
        ]
    },
    output: {
        path: path.resolve(__dirname, '../wwwroot/js'),
        filename: "index.bundle.js"
    }
};
```

- Save and close the webpack file.
- Create a folder “**src**” under NpmJS and inside this src folder add 2 files: “**index.js**” and “**styles.scss**” with following contents

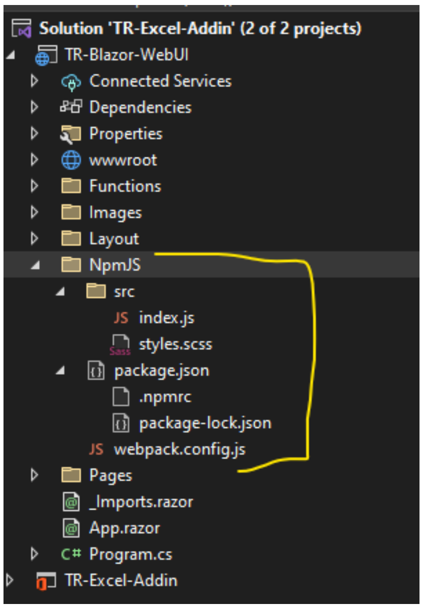
index.js:

```
import { SafButton, SafIcon } from '@saffron/core-components';
import './styles.scss';
SafButton();
SafIcon();
```

styles.scss:

```
@import "../node_modules/@saffron/core-styles/dist/index.css";
@import '../node_modules/@saffron/core-styles/dist/fonts.css';
@import '../node_modules/@saffron/core-styles/dist/font-awesome.css';
```

- Finally, your NpmJS folder should have following files and folder



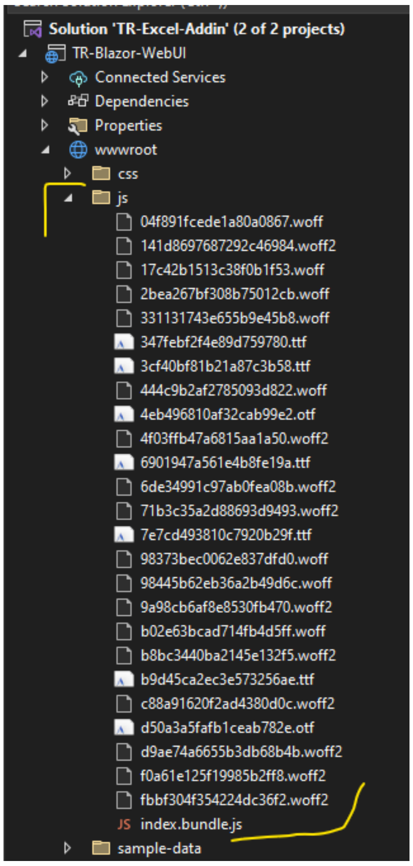
```
Solution 'TR-Excel-Addin' (2 of 2 projects)
  TR-Blazor-WebUI
    Connected Services
    Dependencies
    Properties
    wwwroot
    Functions
    Images
    Layout
    NpmJS
      src
        index.js
        styles.scss
      package.json
        .npmrc
        package-lock.json
      webpack.config.js
    Pages
    _Imports.razor
    App.razor
    Program.cs
  TR-Excel-Addin
```

- Now go to the Terminal window again and run following 2 commands separately at NpmJS path

npm install

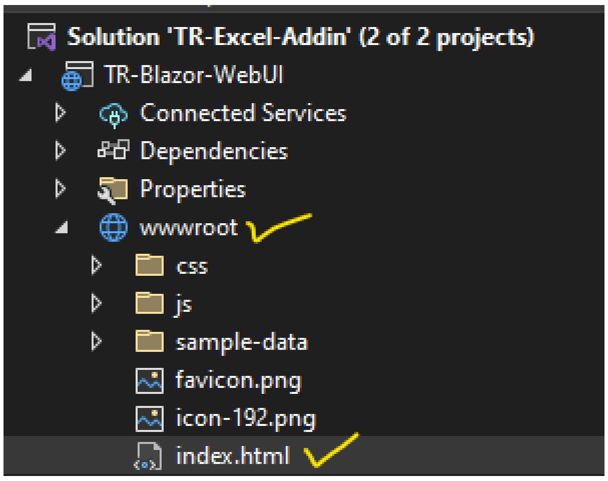
npm run build

- After running **npm run build**, you should see the new folder "**js**" will be created under **wwwroot,** with all the javascript bundles along with **index.bundle.js** file as follows



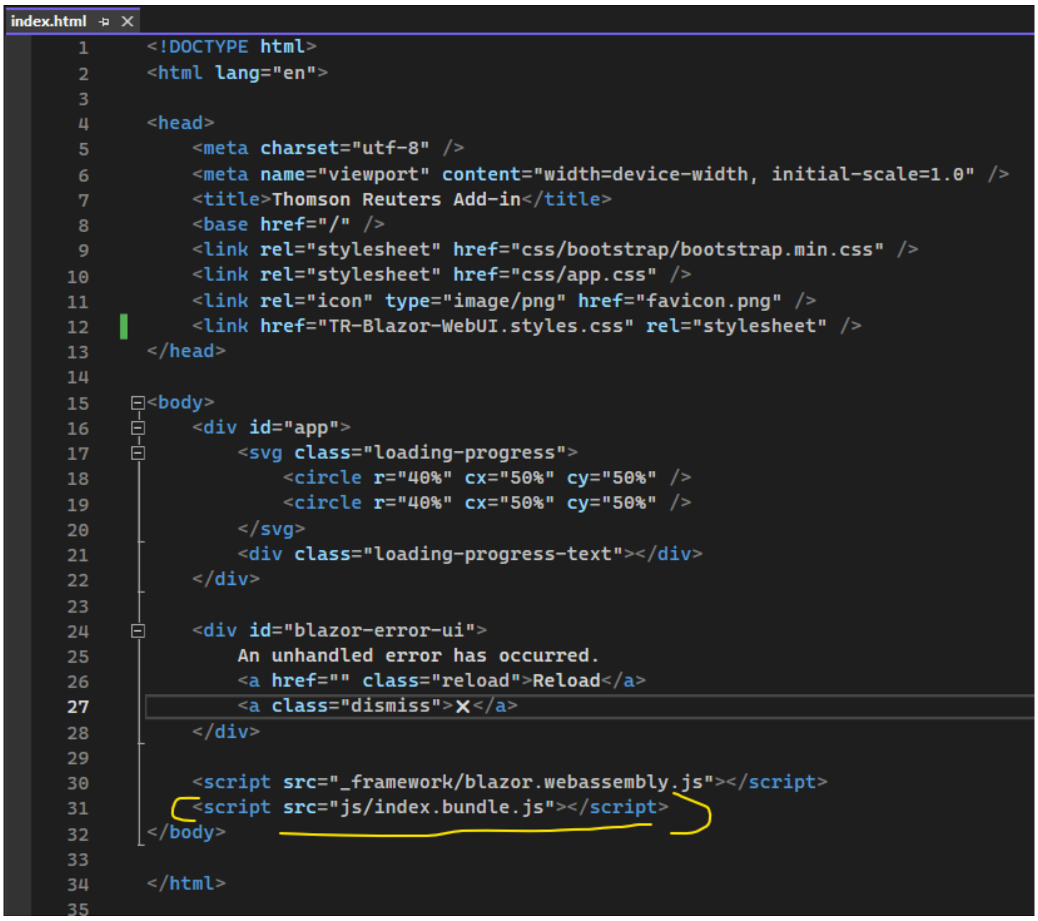
```
Solution 'TR-Excel-Addin' (2 of 2 projects)
  TR-Blazor-WebUI
    Connected Services
    Dependencies
    Properties
    wwwroot
      css
      js
        04f891fcede1a80a0867.woff
        141d8697687292c46984.woff2
        17c42b1513c38f0b1f53.woff
        2bea267bf308b75012cb.woff
        331131743e655b9e45b8.woff
        347febf2f4e89d759780.ttf
        3cf40bf81b21a87c3b58.ttf
        444c9b2af2785093d822.woff
        4eb496810af32cab99e2.otf
        4f03ffb47a6815aa1a50.woff2
        6901947a561e4b8fe19a.ttf
        6de34991c97ab0fea08b.woff2
        71b3c35a2d88693d9493.woff2
        7e7cd493810c7920b29f.ttf
        98373bec0062e837dfd0.woff
        98445b62eb36a2b49d6c.woff
        9a98cb6af8e8530fb470.woff2
        b02e63bcad714fb4d5ff.woff
        b8bc3440ba2145e132f5.woff2
        b9d45ca2ec3e573256ae.ttf
        c88a91620f2ad4380d0c.woff2
        d50a3a5fafb1ceab782e.otf
        d9ae74a6655b3db68b4b.woff2
        f0a61e125f19985b2ff8.woff2
        fbbf304f354224dc36f2.woff2
        index.bundle.js
      sample-data
```

- Now open **index.html** file under wwwroot folder



- Add following script at **body** section of the **index.html** file according to the screenshot below

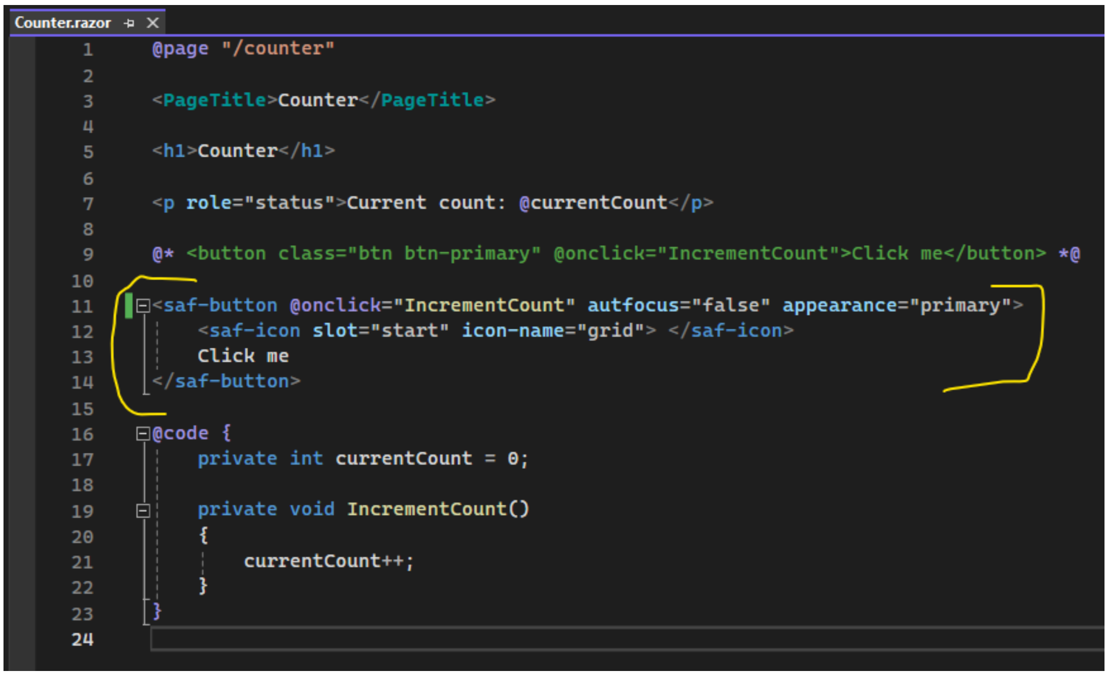
<script src="js/index.bundle.js"></script>



```
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="utf-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <title>Thomson Reuters Add-in</title>
    <base href="/" />
    <link rel="stylesheet" href="css/bootstrap/bootstrap.min.css" />
    <link rel="stylesheet" href="css/app.css" />
    <link rel="icon" type="image/png" href="favicon.png" />
    <link href="TR-Blazor-WebUI.styles.css" rel="stylesheet" />
</head>

<body>
    <div id="app">
        <svg class="loading-progress">
            <circle r="40%" cx="50%" cy="50%" />
            <circle r="40%" cx="50%" cy="50%" />
        </svg>
        <div class="loading-progress-text"></div>
    </div>

    <div id="blazor-error-ui">
        An unhandled error has occurred.
        <a href="" class="reload">Reload</a>
        <a class="dismiss">X</a>
    </div>

    <script src="_framework/blazor.webassembly.js"></script>
    <script src="js/index.bundle.js"></script>
</body>

</html>
```

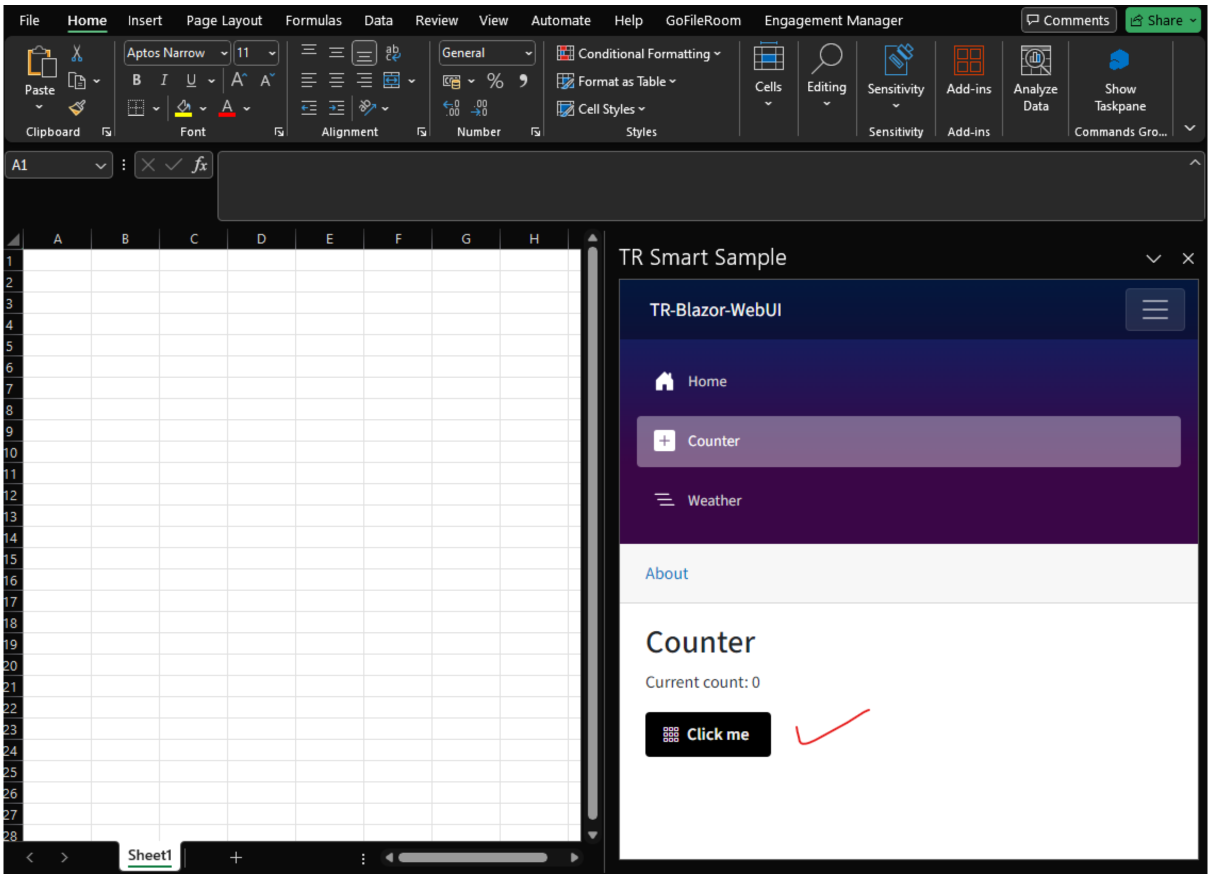
- Our Blazor project is ready to consume the Saffron styling.
- Hence, now go to the **Counter.razor** page under the Pages folder and replace the default bootstrap button with the saffron button as follows:

<saf-button @onclick="IncrementCount" autfocus="false" appearance="primary">
<saf-icon slot="start" icon-name="grid"> </saf-icon>
Click me
</saf-button>



```
@page "/counter"

<PageTitle>Counter</PageTitle>

<h1>Counter</h1>

<p role="status">Current count: @currentCount</p>

@* <button class="btn btn-primary" @onclick="IncrementCount">Click me</button> *@

<saf-button @onclick="IncrementCount" autfocus="false" appearance="primary">
    <saf-icon slot="start" icon-name="grid"> </saf-icon>
    Click me
</saf-button>

@code {
    private int currentCount = 0;

    private void IncrementCount()
    {
        currentCount++;
    }
}
```

- Build the solution and run.
- When the excel opens with Taskpane, navigate to the Counter page from Navbar DDMenu as follows, and you will see that the **"Click me"** button is now replaced with the TR-Saffron styling button with icon.



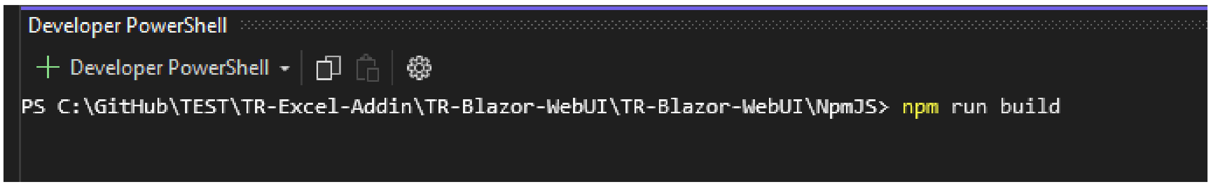
Counter

Current count: 0

Click me

- **<u>Note:</u>**

  Anytime, you add new Saffron components inside **NpmJS >> src >> index.js** file, you need to run the "**npm run build**" command inside NpmJS directory path, to consume that saf-component into the Blazor WebUI



```
PS C:\GitHub\TEST\TR-Excel-Addin\TR-Blazor-WebUI\TR-Blazor-WebUI\NpmJS> npm run build
```