APL64 Cross-platform Component as an Azure Function Application

Contents

Overview

An APL64 Cross-platform component (CPC) can be the basis of a 'server-less' web application. APL64 programmer-defined functions support the algorithmic, calculation and other selected portions of the application, such as data persistence, output generation, etc.

The application is 'server-less' because the application will run only when there is a user request for it. The application does not require a '24-7' dedicated web server running even when there are no users. The cost to expose a 'server-less' application is generally based on the actual resources used to run the application by an end user.

Such an application is automatically scalable from a no, a few, or many end users, without excess costs for 'idle' web servers.

The technology to create a 'server-less' application is available in Microsoft Azure Function, Google Cloud and AWS Lambda. This example illustrates an APL64 CPC exposed as a Microsoft Azure function. In these environments, the same APL64 CPC can be transparently used.

Microsoft Azure is a cloud-based application platform. An Azure Function is server-less application with minimal infrastructure and production costs dependent on actual client usage. In this example a Azure Function project is created with calculation functionality provided by an APL64 cross-platform component (CPC).

The event which causes the Azure function to run is a 'trigger'. An Azure function can be triggered in many ways, including timer, http web request, mobile application, another web server, a request from a different Azure Function, a desktop application, etc.

In this example the application is triggered by a client's web request. The web request is initiated by browsing to a url for the Azure function application. The web request triggers the first Azure function in this example to present an html page for application-specific user input. When the user submits the input, another Azure function calculates the desired results, and prepares the browser response for the user.

In this example the APL64 cross-platform component (CPC) exposes two APL64 programmer-developed, public functions which provide the application's calculation functionality.

Developing an Azure Function application involves APL64 and Microsoft .Net technology. In this example Microsoft Visual Studio 2022 is used to develop the application. Microsoft Visual Studio Community version and Visual Studio Code are available at no cost.

Download the Example Application

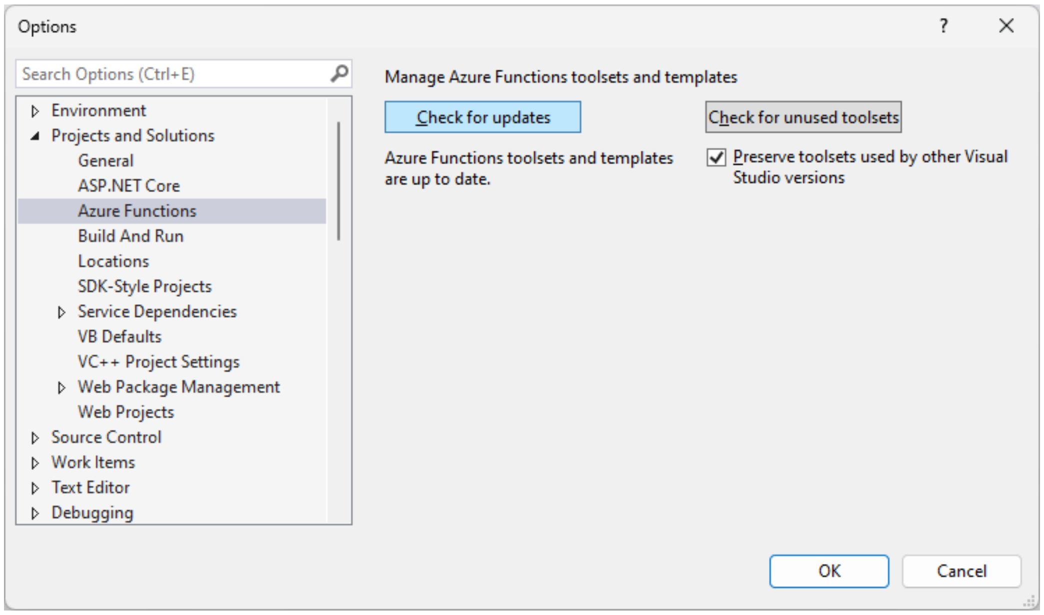
Download the example source code here:

http://apl2000.com/APL64/UserDocumentation/APL64CpcAzureFunction.zip

Expand the zip-format file to c:\, so that it is easy to follow this document. The compiled portions of the application are not included in the zip-format file. To be able to run the application follow the directions in each step of this document. Some steps have already been applied to the downloaded application. The document instructions provide the steps to a complete sample application without downloading the zip-format file.

Create the Azure Function Application in Visual Studio

Open Visual Studio, without creating a project, and check that the version is at least 'Version 17.12.4' and that the Visual Studio project templates are up to date:


Options

Search Options (Ctrl+E)

- Environment
- Projects and Solutions
  - General
  - ASP.NET Core
  - Azure Functions
  - Build And Run
  - Locations
  - SDK-Style Projects
  - Service Dependencies
  - VB Defaults
  - VC++ Project Settings
  - Web Package Management
  - Web Projects
- Source Control
- Work Items
- Text Editor
- Debugging

Manage Azure Functions toolsets and templates

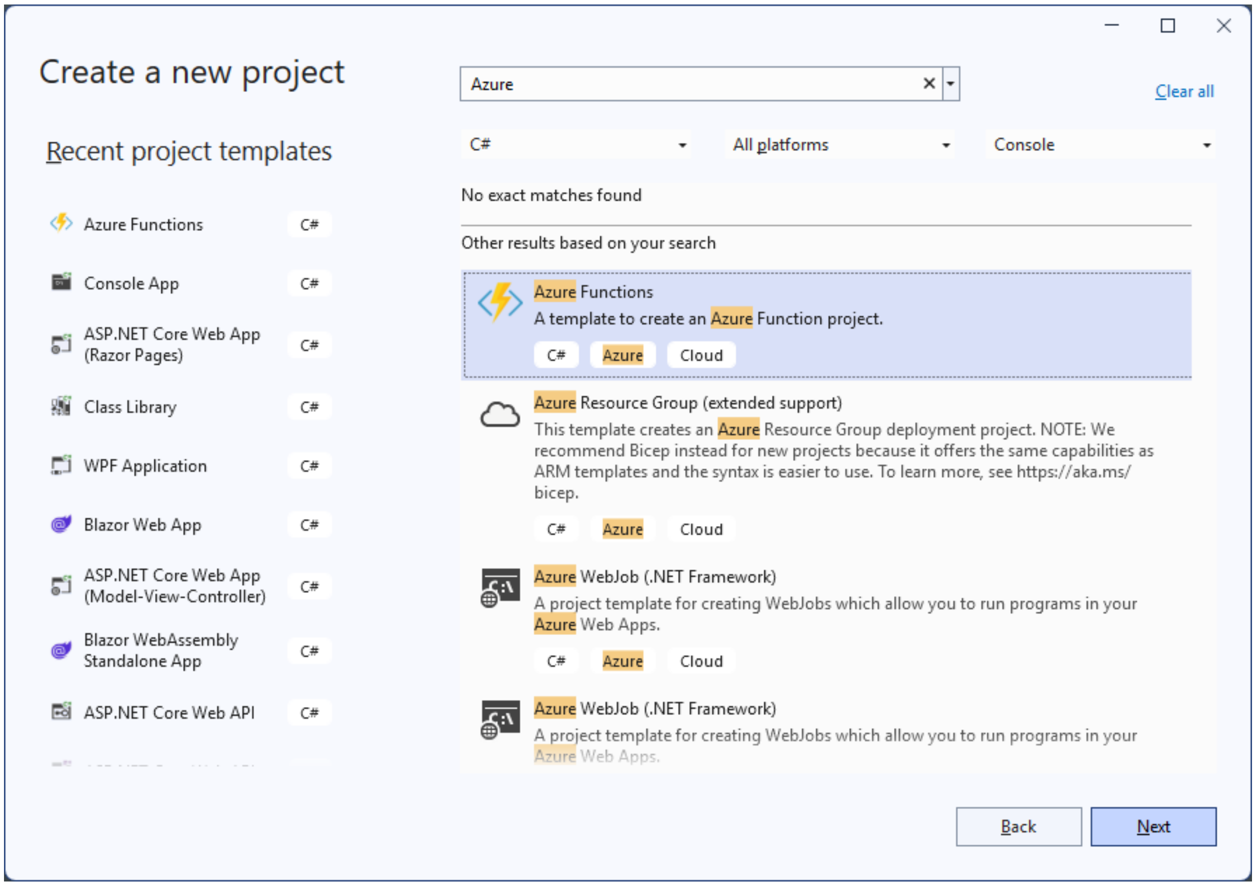
Check for updates | Check for unused toolsets

Azure Functions toolsets and templates are up to date.

[✓] Preserve toolsets used by other Visual Studio versions

OK | Cancel

In Visual Studio 2022, create a new project using the Azure Functions template:


Create a new project

Recent project templates

Azure Functions	C#
Console App	C#
ASP.NET Core Web App (Razor Pages)	C#
Class Library	C#
WPF Application	C#
Blazor Web App	C#
ASP.NET Core Web App (Model-View-Controller)	C#
Blazor WebAssembly Standalone App	C#
ASP.NET Core Web API	C#

Azure | Clear all

C# | All platforms | Console

No exact matches found

Other results based on your search

Azure Functions
A template to create an Azure Function project.
C# Azure Cloud

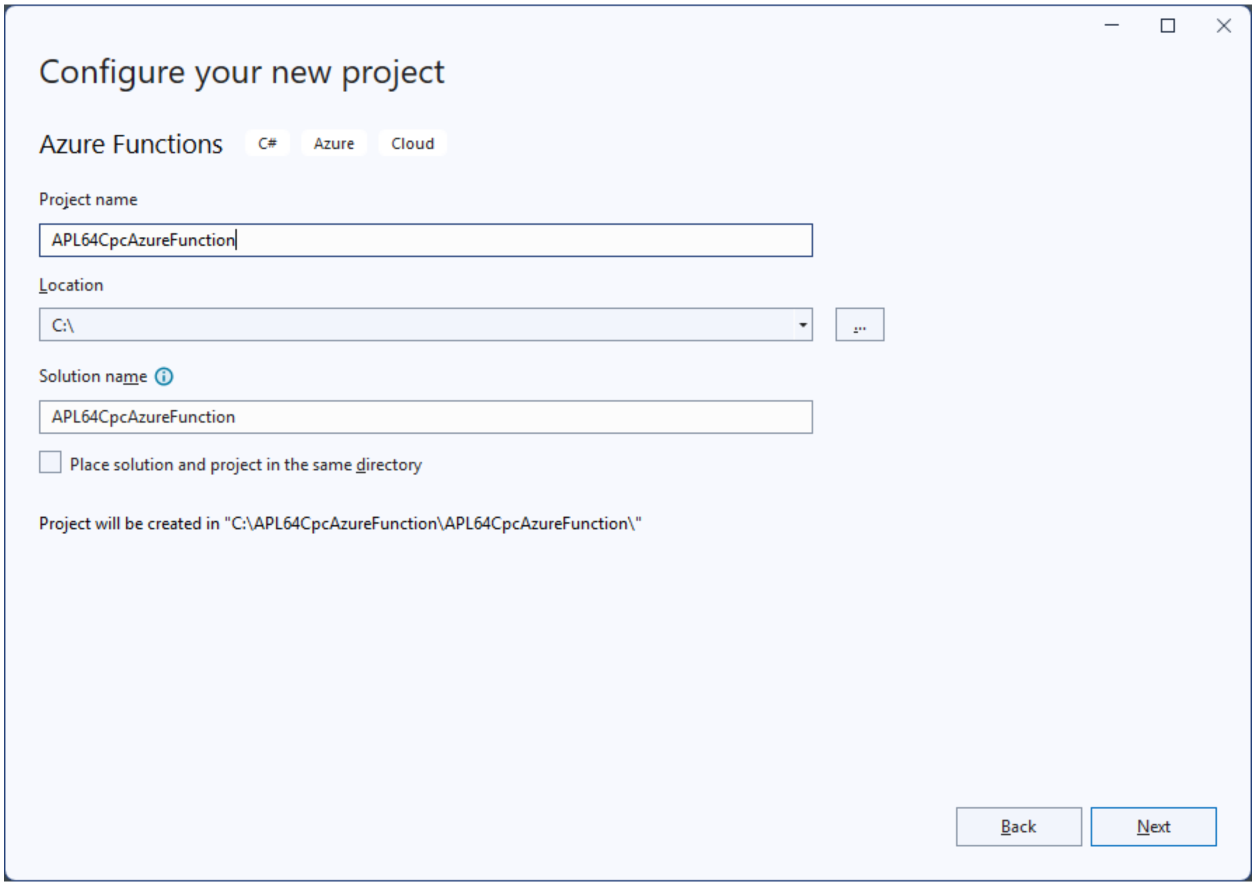
Azure Resource Group (extended support)
This template creates an Azure Resource Group deployment project. NOTE: We recommend Bicep instead for new projects because it offers the same capabilities as ARM templates and the syntax is easier to use. To learn more, see https://aka.ms/bicep.
C# Azure Cloud

Azure WebJob (.NET Framework)
A project template for creating WebJobs which allow you to run programs in your Azure Web Apps.
C# Azure Cloud

Azure WebJob (.NET Framework)
A project template for creating WebJobs which allow you to run programs in your Azure Web Apps.

Back | Next

Complete the configuration dialog:


Configure your new project

Azure Functions C# Azure Cloud

Project name: APL64CpcAzureFunction

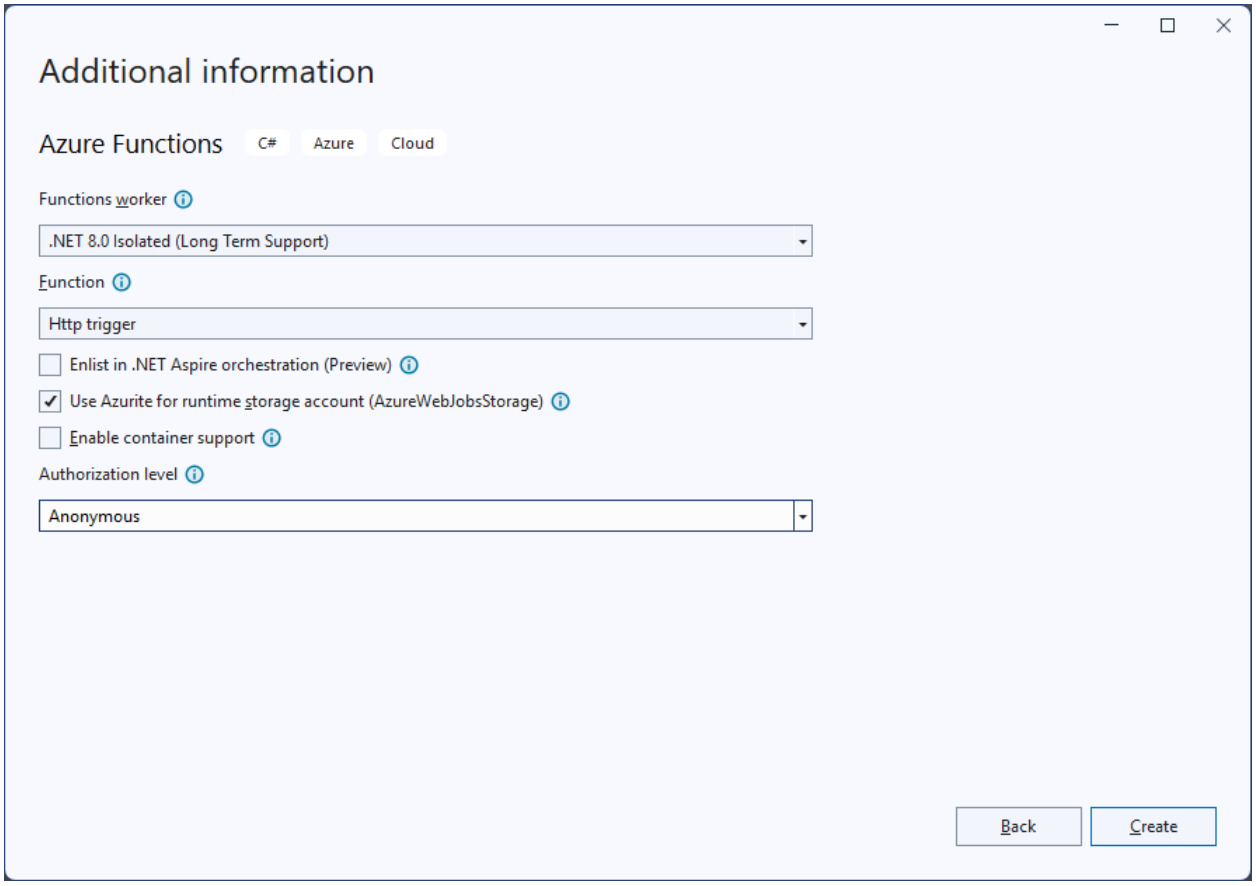
Location: C:\

Solution name: APL64CpcAzureFunction

☐ Place solution and project in the same directory

Project will be created in "C:\APL64CpcAzureFunction\APL64CpcAzureFunction\"

Complete the additional information dialog. Iin this example an HTTP Trigger is illustrated. The 'Anonymous' authorization level is selected for design and debugging purposes. In a production environment a more secure authorization level is recommended.


Additional information

Azure Functions C# Azure Cloud

Functions worker: .NET 8.0 Isolated (Long Term Support)

Function: Http trigger

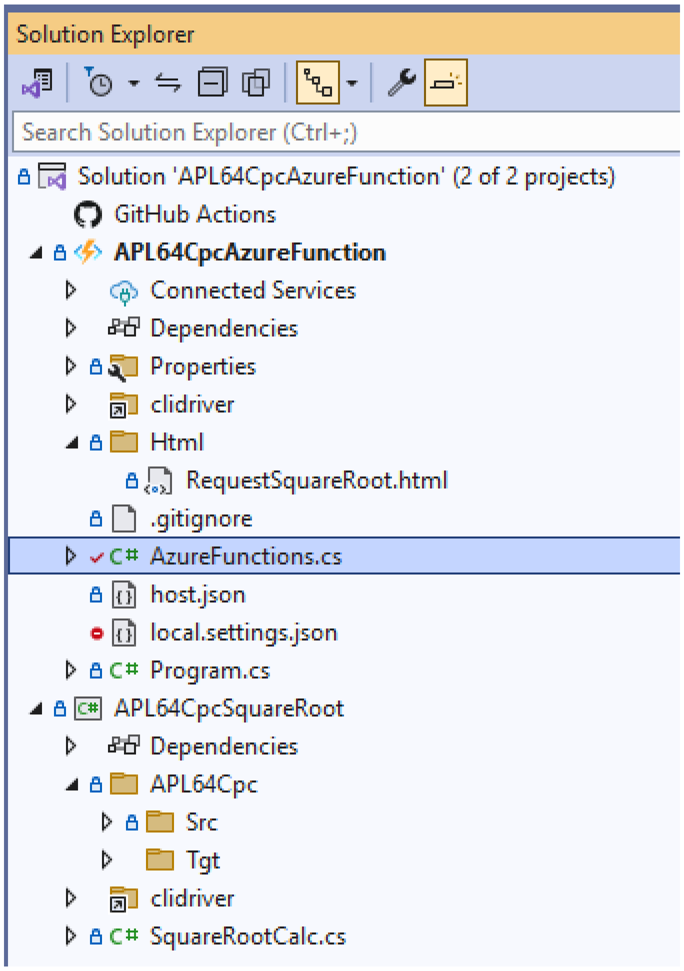
[ ] Enlist in .NET Aspire orchestration (Preview)

[✓] Use Azurite for runtime storage account (AzureWebJobsStorage)

[ ] Enable container support

Authorization level: Anonymous

Rename the ‘Function1.cs’ and included class source code in the APL64CpcAzureFunction project to ‘AzureFunctions.cs’:

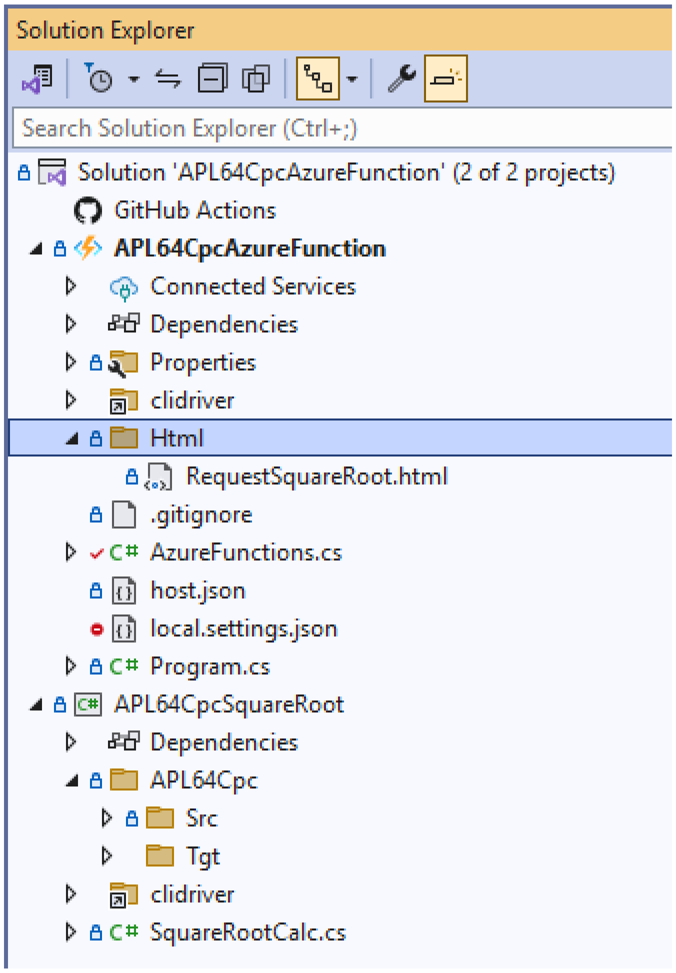


Solution Explorer

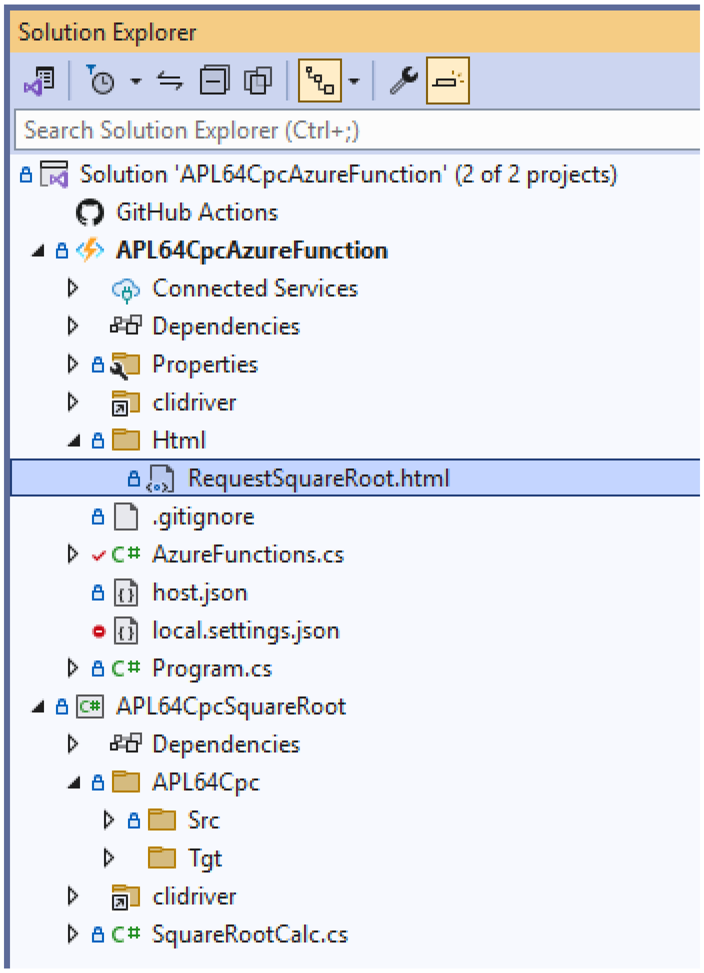
Search Solution Explorer (Ctrl+;)

```
Solution 'APL64CpcAzureFunction' (2 of 2 projects)
    GitHub Actions
  APL64CpcAzureFunction
    Connected Services
    Dependencies
    Properties
    clidriver
    Html
      RequestSquareRoot.html
    .gitignore
    AzureFunctions.cs
    host.json
    local.settings.json
    Program.cs
  APL64CpcSquareRoot
    Dependencies
    APL64Cpc
      Src
      Tgt
    clidriver
    SquareRootCalc.cs
```

Add the Html folder to the APL64CpcAzureFunction project:



Add the ‘RequestSquareRoot.html’ web page into the Html folder of the APL64CpcAzureFunction project:


Solution Explorer

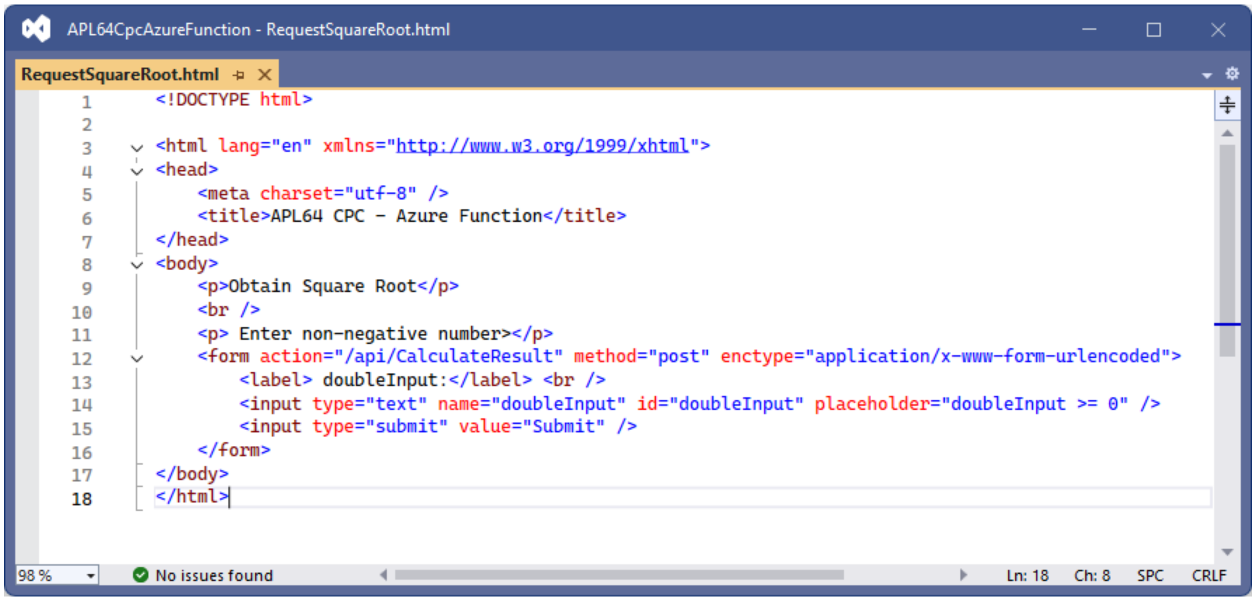
Search Solution Explorer (Ctrl+;)

```
Solution 'APL64CpcAzureFunction' (2 of 2 projects)
    GitHub Actions
  APL64CpcAzureFunction
      Connected Services
      Dependencies
      Properties
      clidriver
      Html
          RequestSquareRoot.html
      .gitignore
      AzureFunctions.cs
      host.json
      local.settings.json
      Program.cs
  APL64CpcSquareRoot
      Dependencies
      APL64Cpc
          Src
          Tgt
      clidriver
      SquareRootCalc.cs
```

Replace the source code of the 'RequestSquareRoot.html' web page into the Html folder of the APL64CpcAzureFunction project with:

```
<!DOCTYPE html>

<html lang="en" xmlns="http://www.w3.org/1999/xhtml">
<head>
  <meta charset="utf-8" />
  <title>APL64 CPC - Azure Function</title>
</head>
<body>
  <p>Obtain Square Root</p>
  <br />
  <p> Enter non-negative number></p>
  <form  action="/api/CalculateResult"  method="post"  enctype="application/x-www-form-
urlencoded">
    <label> doubleInput:</label> <br />
    <input type="text" name="doubleInput" id="doubleInput" placeholder="doubleInput >= 0" />
    <input type="submit" value="Submit" />
  </form>
```

```
</body>
</html>
```


APL64CpcAzureFunction - RequestSquareRoot.html

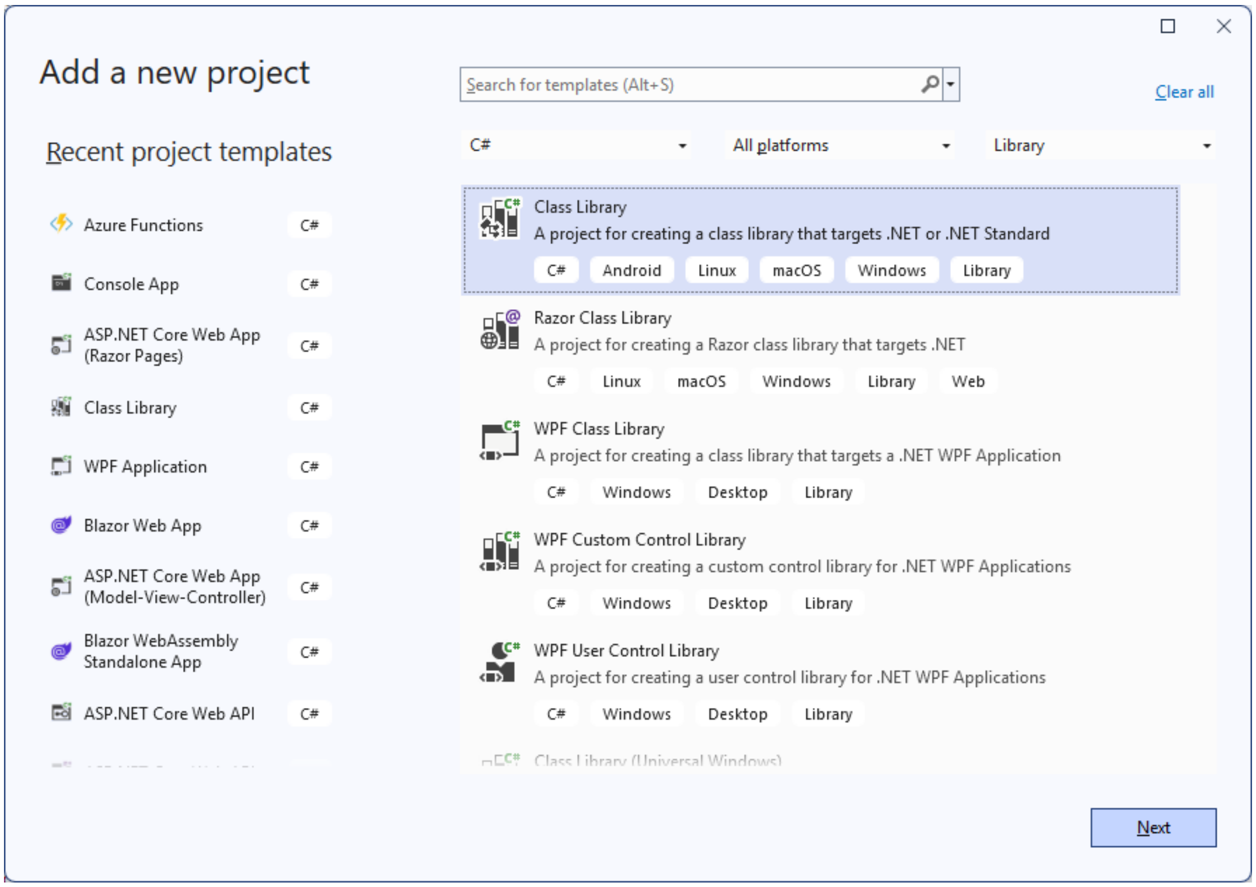
RequestSquareRoot.html

```
<!DOCTYPE html>

<html lang="en" xmlns="http://www.w3.org/1999/xhtml">
<head>
    <meta charset="utf-8" />
    <title>APL64 CPC - Azure Function</title>
</head>
<body>
    <p>Obtain Square Root</p>
    <br />
    <p> Enter non-negative number></p>
    <form action="/api/CalculateResult" method="post" enctype="application/x-www-form-urlencoded">
        <label> doubleInput:</label> <br />
        <input type="text" name="doubleInput" id="doubleInput" placeholder="doubleInput >= 0" />
        <input type="submit" value="Submit" />
    </form>
</body>
</html>
```

98 %   No issues found   Ln: 18   Ch: 8   SPC   CRLF

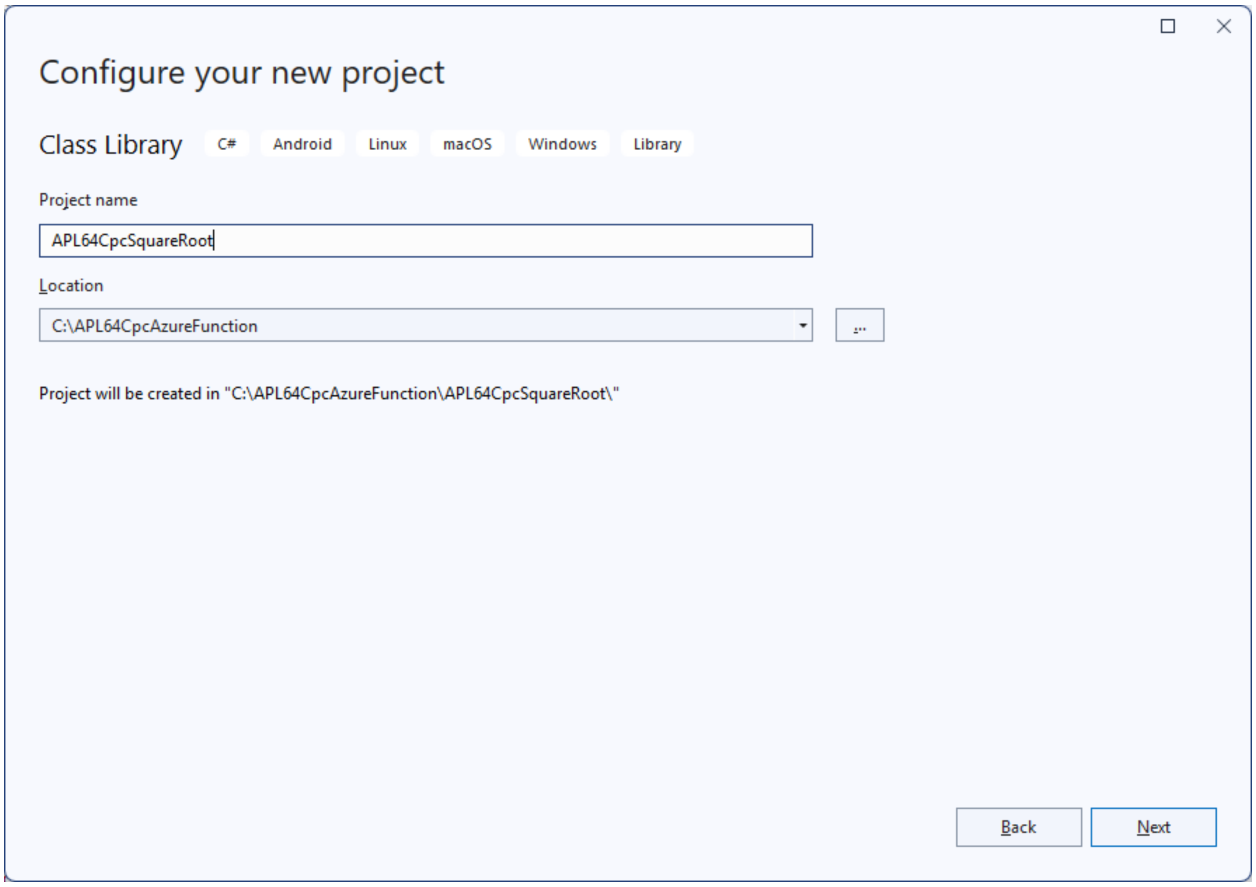
Set the Properties | Build Action of the 'RequestSquareRoot.html' file to 'Content' and its 'Copy to Output Directory' setting to 'Copy always'. These settings assure that when the Azure function is run, the 'RequestSquareRoot.html' page text can be served to the user as a 'WebResponse'.

Add a 'class library' project to the solution, to contain the APL64 CPC:



Add a new project
Search for templates (Alt+S)
Clear all
C#
All platforms
Library
Recent project templates
Azure Functions C#
Console App C#
ASP.NET Core Web App (Razor Pages) C#
Class Library C#
WPF Application C#
Blazor Web App C#
ASP.NET Core Web App (Model-View-Controller) C#
Blazor WebAssembly Standalone App C#
ASP.NET Core Web API C#
Class Library
A project for creating a class library that targets .NET or .NET Standard
C# Android Linux macOS Windows Library
Razor Class Library
A project for creating a Razor class library that targets .NET
C# Linux macOS Windows Library Web
WPF Class Library
A project for creating a class library that targets a .NET WPF Application
C# Windows Desktop Library
WPF Custom Control Library
A project for creating a custom control library for .NET WPF Applications
C# Windows Desktop Library
WPF User Control Library
A project for creating a user control library for .NET WPF Applications
C# Windows Desktop Library
Class Library (Universal Windows)
Next

Configure the class library project:


Configure your new project

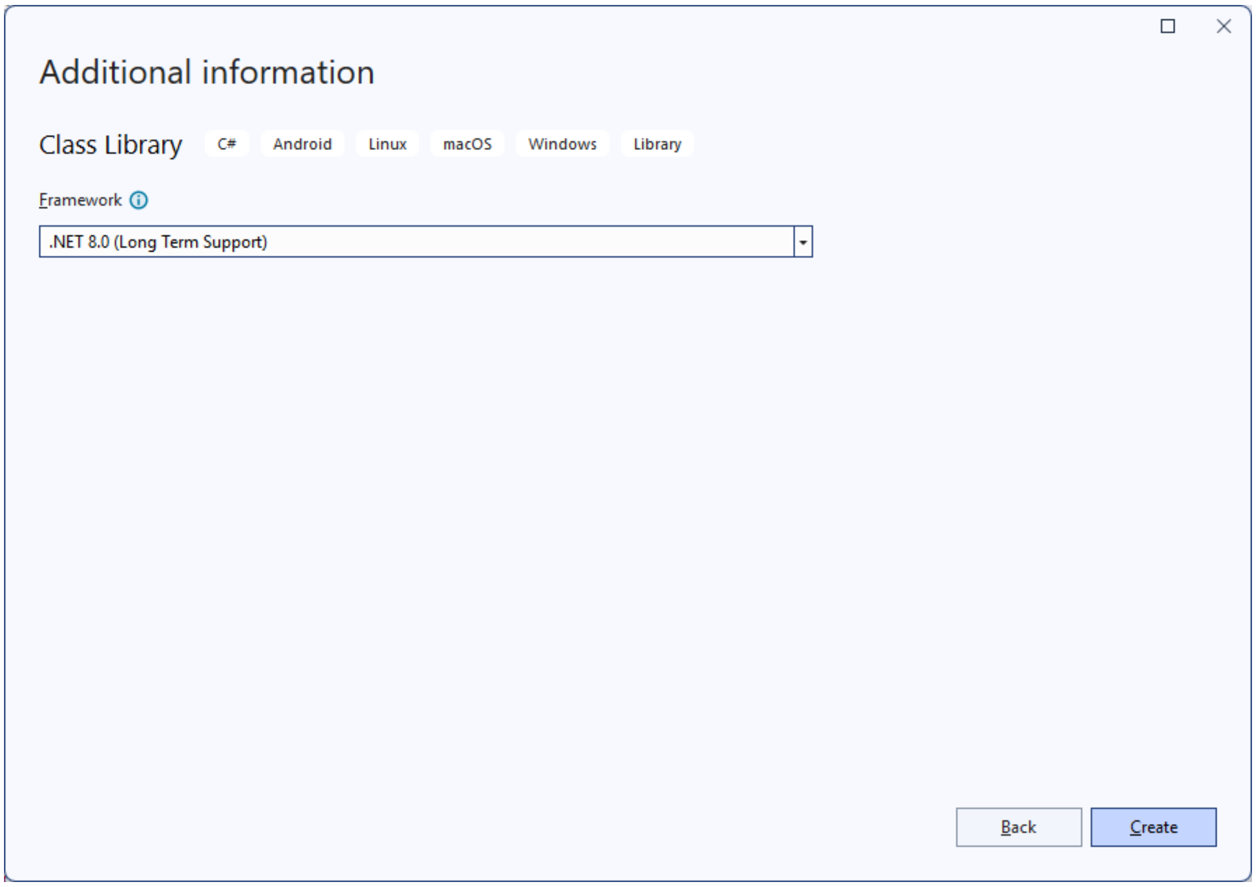
Class Library

Project name: APL64CpcSquareRoot

Location: C:\APL64CpcAzureFunction

Project will be created in "C:\APL64CpcAzureFunction\APL64CpcSquareRoot\"

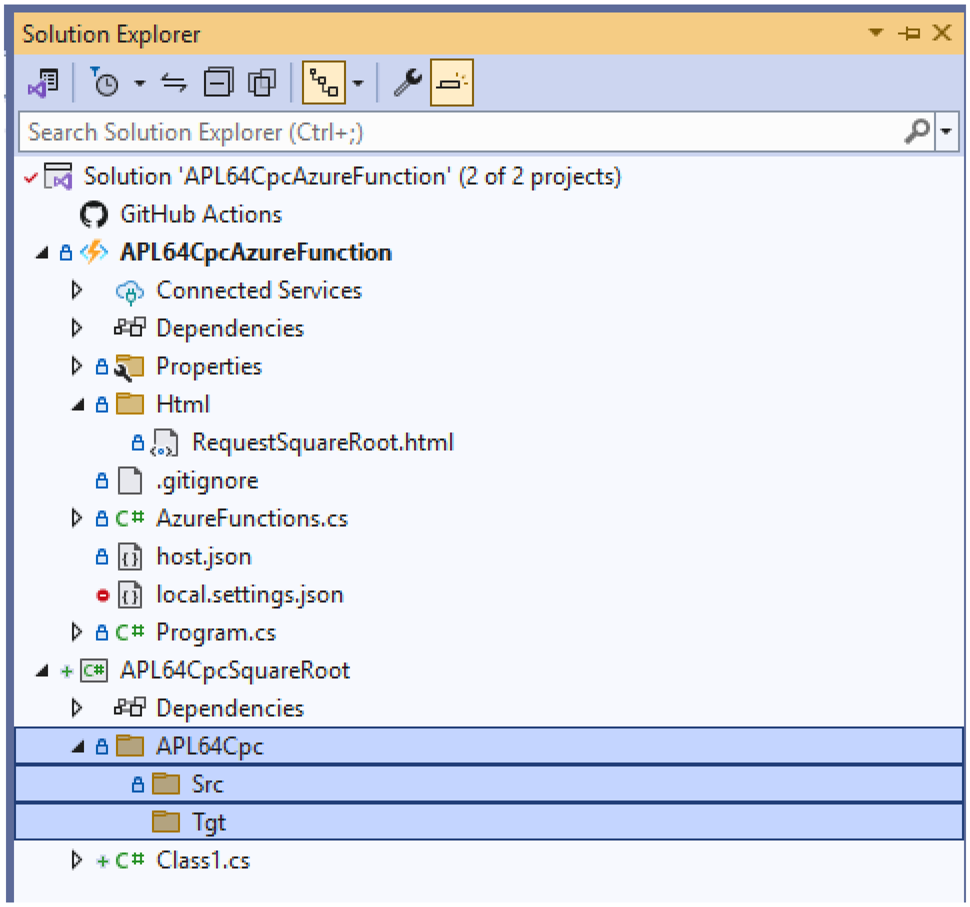
Provide the additional information for the class library project:


Additional information

Class Library

Framework: .NET 8.0 (Long Term Support)

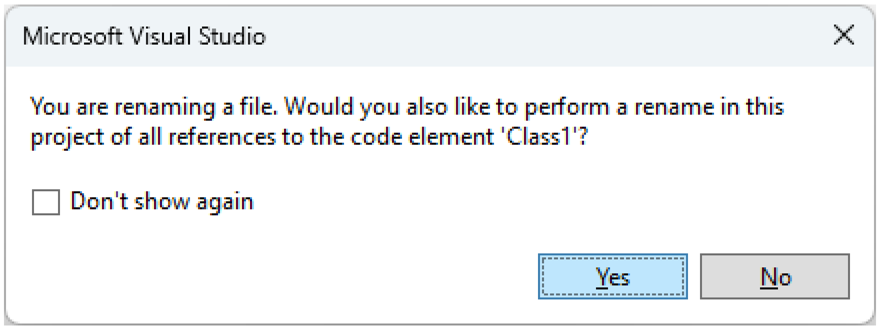
Add the APL64Cpc folder and Src and Tgt folders to the APL64CpcSquareRoot project:


Solution Explorer

Search Solution Explorer (Ctrl+;)

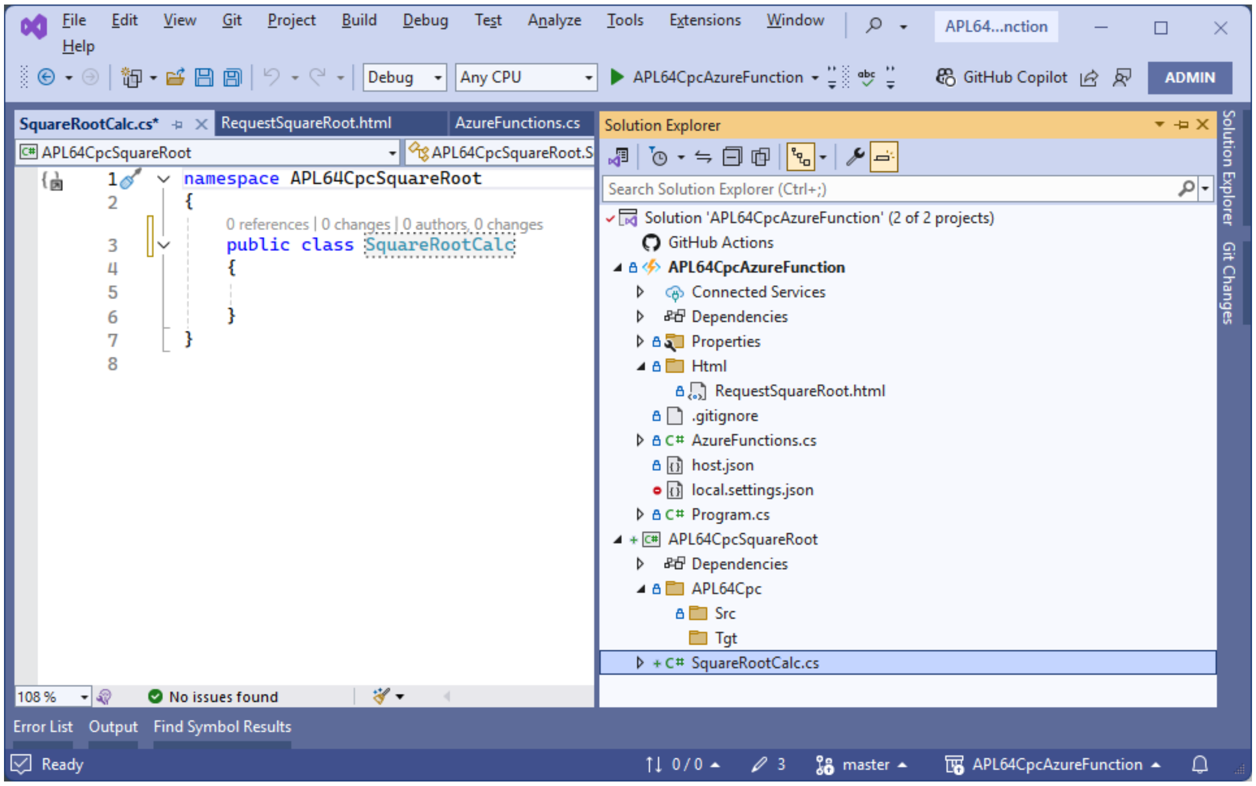
```
Solution 'APL64CpcAzureFunction' (2 of 2 projects)
    GitHub Actions
  APL64CpcAzureFunction
      Connected Services
      Dependencies
      Properties
      Html
        RequestSquareRoot.html
      .gitignore
      AzureFunctions.cs
      host.json
      local.settings.json
      Program.cs
  APL64CpcSquareRoot
      Dependencies
      APL64Cpc
        Src
        Tgt
      Class1.cs
```

In the APL64CpcSquareRoot project, rename the Class1.cs code file and the contained class source code to SquareRootCalc.cs


Microsoft Visual Studio

You are renaming a file. Would you also like to perform a rename in this project of all references to the code element 'Class1'?

Don't show again



```
namespace APL64CpcSquareRoot
{
    0 references | 0 changes | 0 authors, 0 changes
    public class SquareRootCalc
    {

    }
}
```

Create the APL64 CPC

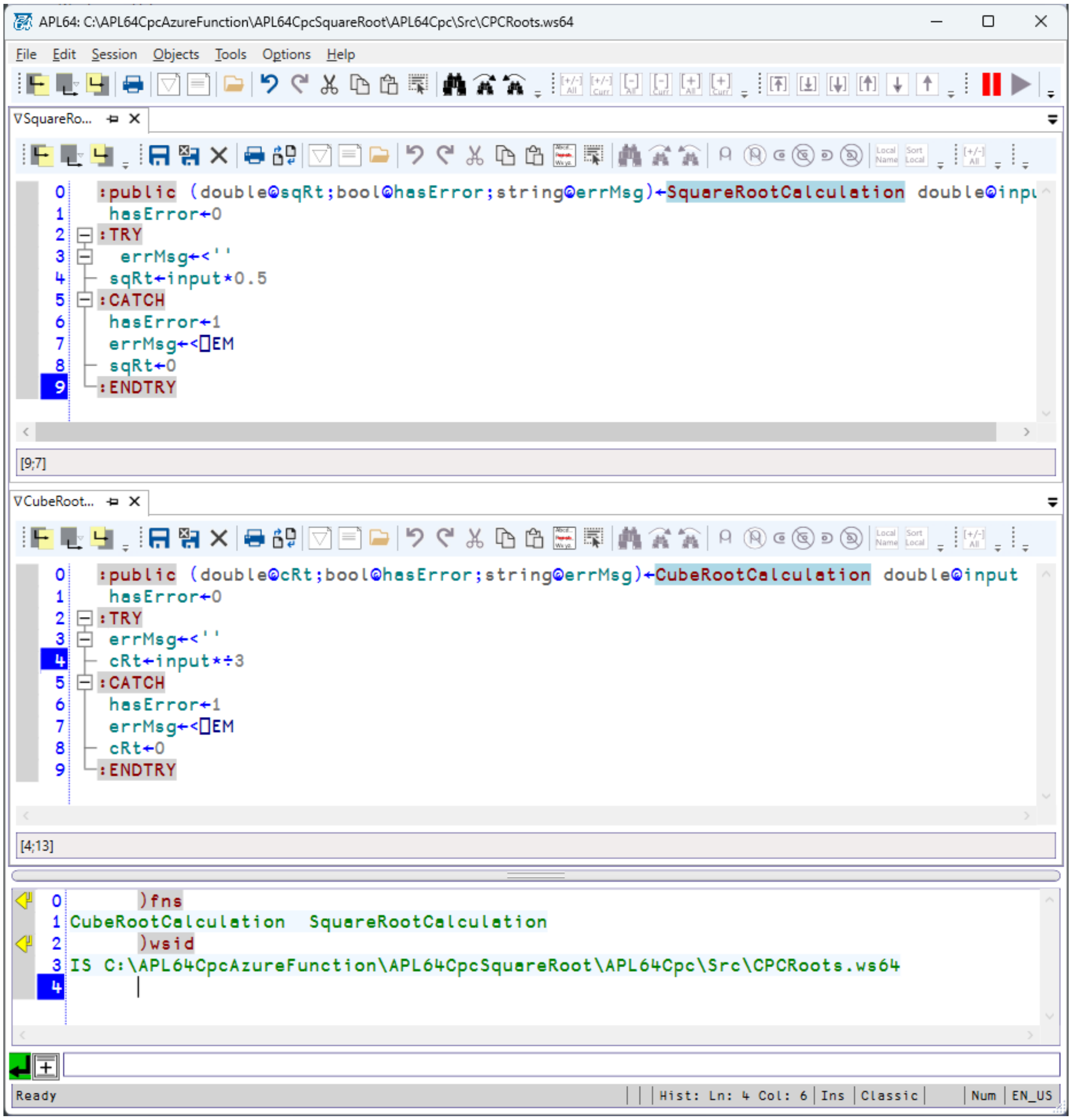
In an APL64 Developer version, create the APL64 public functions SquareRootCalculation and CubeRootCalculation functions.

.Net requires explicit data types for method arguments and results. In this example the Azure function provides and expects string data type values, so the APL64 programmer-defined, public function must use the enstring/destring (</>) functions appropriately.

```
:public (double@sqRt;bool@hasError;string@errMsg)←SquareRootCalculation double@input
hasError←0
:TRY
  errMsg←<''
 sqRt←input*0.5
:CATCH
 hasError←1
 errMsg←<⎕EM
 sqRt←0
:ENDTRY
```

```
:public (double@cRt;bool@hasError;string@errMsg)←CubeRootCalculation double@input
 hasError←0
```

```
:TRY
 errMsg←<''
 cRt←input*÷3
:CATCH
 hasError←1
 errMsg←<⎕EM
 cRt←0
:ENDTRY
```

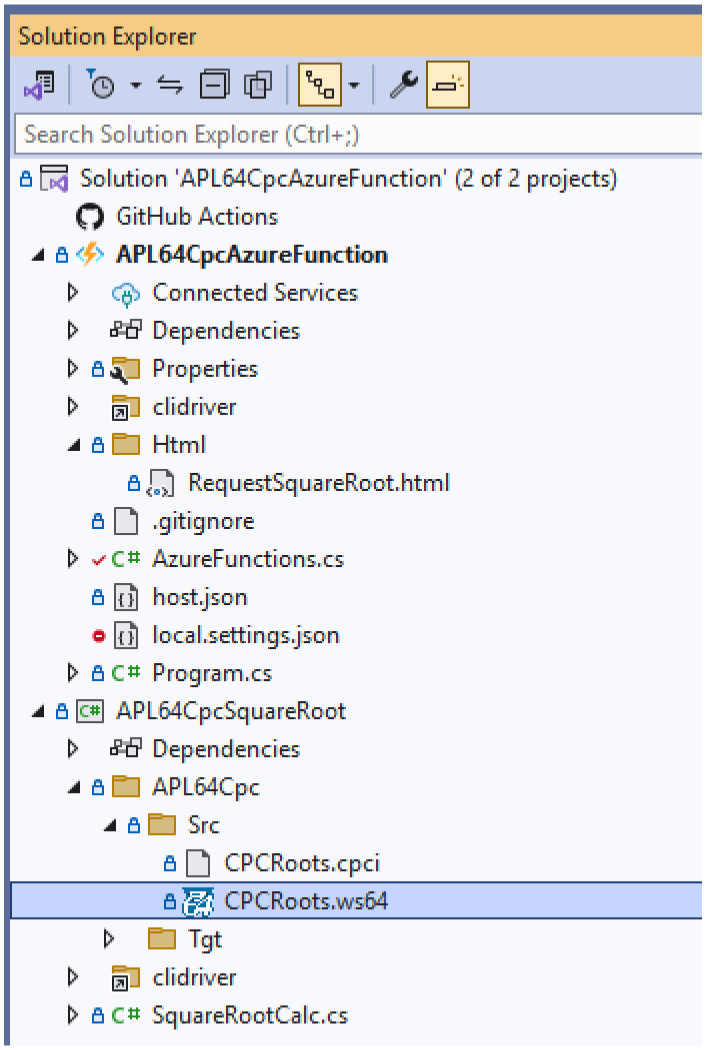

APL64: C:\APL64CpcAzureFunction\APL64CpcSquareRoot\APL64Cpc\Src\CPCRoots.ws64
File Edit Session Objects Tools Options Help
∇SquareRo...
∇CubeRoot...
[9;7]
[4;13]
Ready
Hist: Ln: 4 Col: 6 Ins Classic Num EN_US

```
:public (double@sqRt;bool@hasError;string@errMsg)←SquareRootCalculation double@inpu
 hasError←0
:TRY
  errMsg←<''
 sqRt←input*0.5
:CATCH
 hasError←1
 errMsg←<⎕EM
 sqRt←0
:ENDTRY
```

```
:public (double@cRt;bool@hasError;string@errMsg)←CubeRootCalculation double@input
 hasError←0
:TRY
 errMsg←<''
 cRt←input*÷3
:CATCH
 hasError←1
 errMsg←<⎕EM
 cRt←0
:ENDTRY
```

```
      )fns
CubeRootCalculation   SquareRootCalculation
      )wsid
IS C:\APL64CpcAzureFunction\APL64CpcSquareRoot\APL64Cpc\Src\CPCRoots.ws64
```

Save the workspace to the Src folder of the APL64CpcSquareRoot project:



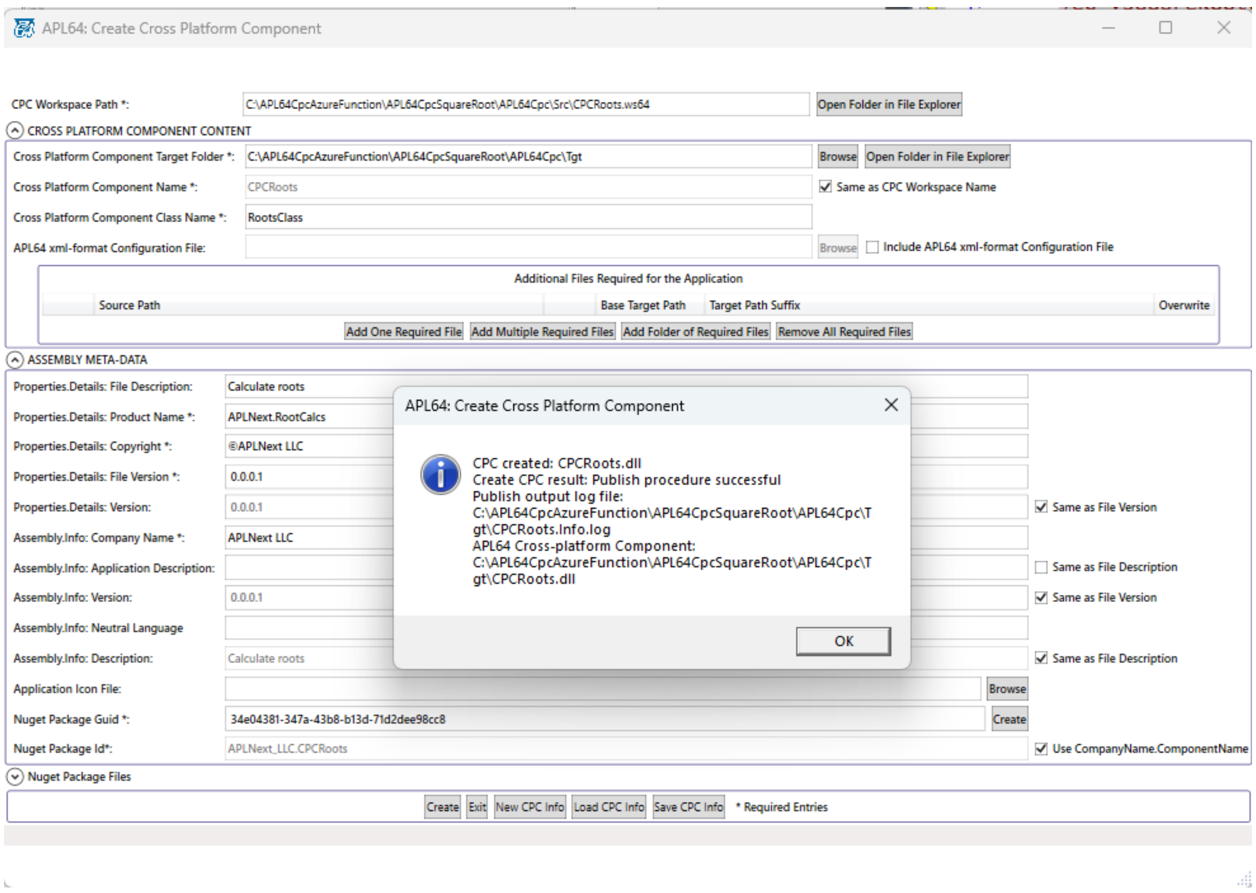
```
Solution 'APL64CpcAzureFunction' (2 of 2 projects)
    GitHub Actions
  APL64CpcAzureFunction
    Connected Services
    Dependencies
    Properties
    clidriver
    Html
      RequestSquareRoot.html
    .gitignore
    AzureFunctions.cs
    host.json
    local.settings.json
    Program.cs
  APL64CpcSquareRoot
    Dependencies
    APL64Cpc
      Src
        CPCRoots.cpci
        CPCRoots.ws64
      Tgt
    clidriver
    SquareRootCalc.cs
```

Use the Options | Create Runtime Assembly | Create Cross-platform Component menu item to present the CPC creation dialog.

Complete the required entries of the CPC creation dialog.

Click the Create button to create the CPC. Some of the entries are case-sensitive since they are used in .Net.

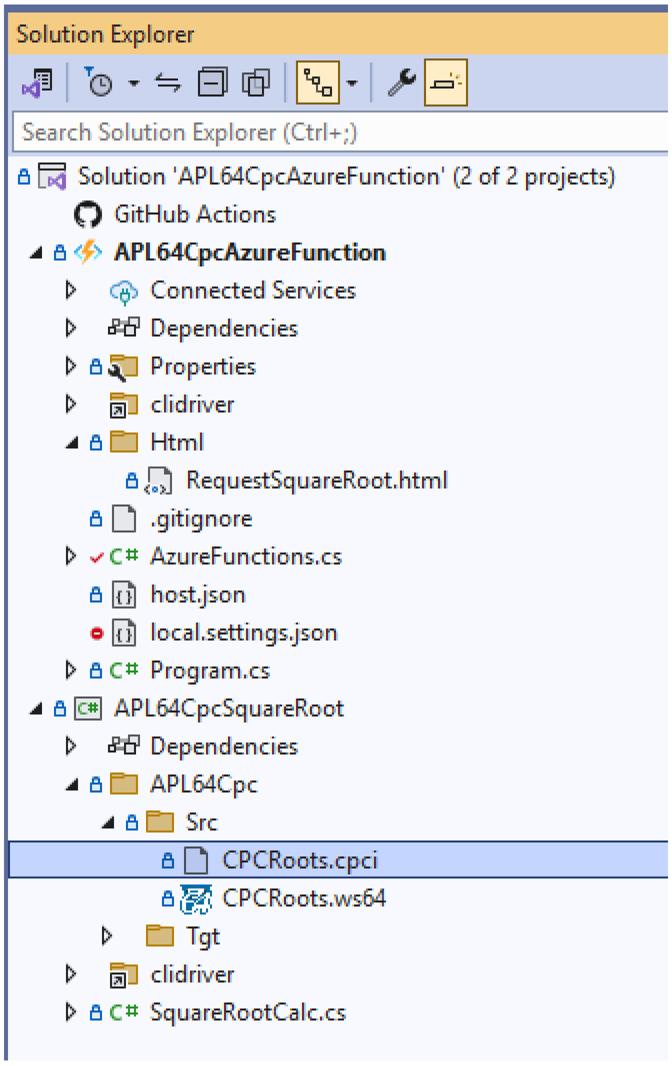
Save the completed CPC information to the Src folder of the APL64CpcSquareRoot project


APL64: Create Cross Platform Component
Open Folder in File Explorer
CROSS PLATFORM COMPONENT CONTENT
Browse
Open Folder in File Explorer
Additional Files Required for the Application
Source Path
Base Target Path
Target Path Suffix
Overwrite
Add One Required File
Add Multiple Required Files
Add Folder of Required Files
Remove All Required Files
ASSEMBLY META-DATA
Browse
Create
Nuget Package Files
Create
Exit
New CPC Info
Load CPC Info
Save CPC Info
* Required Entries
OK

CPC Workspace Path *:	C:\APL64CpcAzureFunction\APL64CpcSquareRoot\APL64Cpc\Src\CPCRoots.ws64
Cross Platform Component Target Folder *:	C:\APL64CpcAzureFunction\APL64CpcSquareRoot\APL64Cpc\Tgt
Cross Platform Component Name *:	CPCRoots ☑ Same as CPC Workspace Name
Cross Platform Component Class Name *:	RootsClass
APL64 xml-format Configuration File:	☐ Include APL64 xml-format Configuration File
Properties.Details: File Description:	Calculate roots
Properties.Details: Product Name *:	APLNext.RootCalcs
Properties.Details: Copyright *:	©APLNext LLC
Properties.Details: File Version *:	0.0.0.1
Properties.Details: Version:	0.0.0.1 ☑ Same as File Version
Assembly.Info: Company Name *:	APLNext LLC
Assembly.Info: Application Description:	☐ Same as File Description
Assembly.Info: Version:	0.0.0.1 ☑ Same as File Version
Assembly.Info: Neutral Language	
Assembly.Info: Description:	Calculate roots ☑ Same as File Description
Application Icon File:	
Nuget Package Guid *:	34e04381-347a-43b8-b13d-71d2dee98cc8
Nuget Package Id*:	APLNext_LLC.CPCRoots ☑ Use CompanyName.ComponentName

APL64: Create Cross Platform Component

```
CPC created: CPCRoots.dll
Create CPC result: Publish procedure successful
Publish output log file:
C:\APL64CpcAzureFunction\APL64CpcSquareRoot\APL64Cpc\T
gt\CPCRoots.Info.log
APL64 Cross-platform Component:
C:\APL64CpcAzureFunction\APL64CpcSquareRoot\APL64Cpc\T
gt\CPCRoots.dll
```


Solution Explorer

Search Solution Explorer (Ctrl+;)

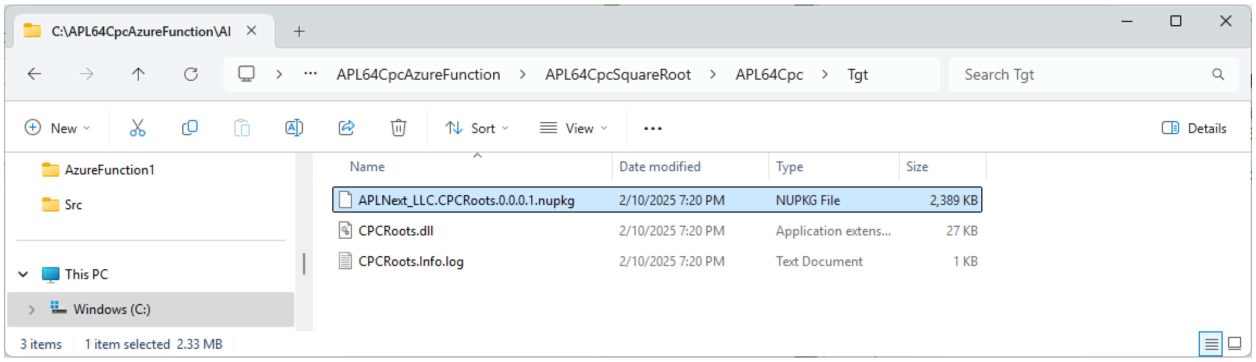
```
Solution 'APL64CpcAzureFunction' (2 of 2 projects)
  GitHub Actions
  APL64CpcAzureFunction
    Connected Services
    Dependencies
    Properties
    clidriver
    Html
      RequestSquareRoot.html
    .gitignore
    AzureFunctions.cs
    host.json
    local.settings.json
    Program.cs
  APL64CpcSquareRoot
    Dependencies
    APL64Cpc
      Src
        CPCRoots.cpci
        CPCRoots.ws64
      Tgt
    clidriver
    SquareRootCalc.cs
```

Publish the APL64 CPC Nuget package

In a production environment the Nuget package may be published to a public or private Nuget repository. For testing purposes, the Nuget package must be copied to a location on the current workstation which is not in the Visual Studio solution folder to avoid including unnecessary components in the project.

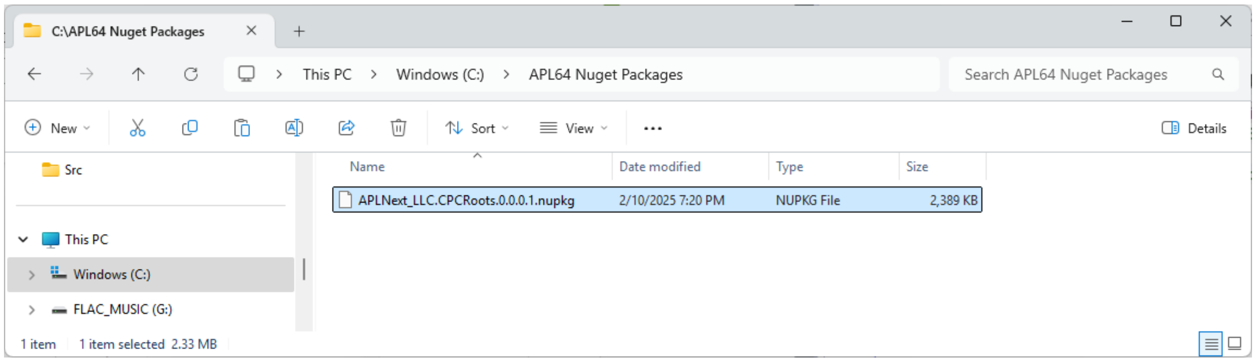
Copy the APL64 CPC Nuget package from the Tgt folder of the APL64CpcSquareRoot project to the APL64 Nuget Package publishing folder:

From:


C:\APL64CpcAzureFunction\Al
APL64CpcAzureFunction > APL64CpcSquareRoot > APL64Cpc > Tgt
Search Tgt
New
Sort
View
Details
AzureFunction1
Src
This PC
Windows (C:)
3 items 1 item selected 2.33 MB

Name	Date modified	Type	Size
APLNext_LLC.CPCRoots.0.0.0.1.nupkg	2/10/2025 7:20 PM	NUPKG File	2,389 KB
CPCRoots.dll	2/10/2025 7:20 PM	Application extens...	27 KB
CPCRoots.Info.log	2/10/2025 7:20 PM	Text Document	1 KB

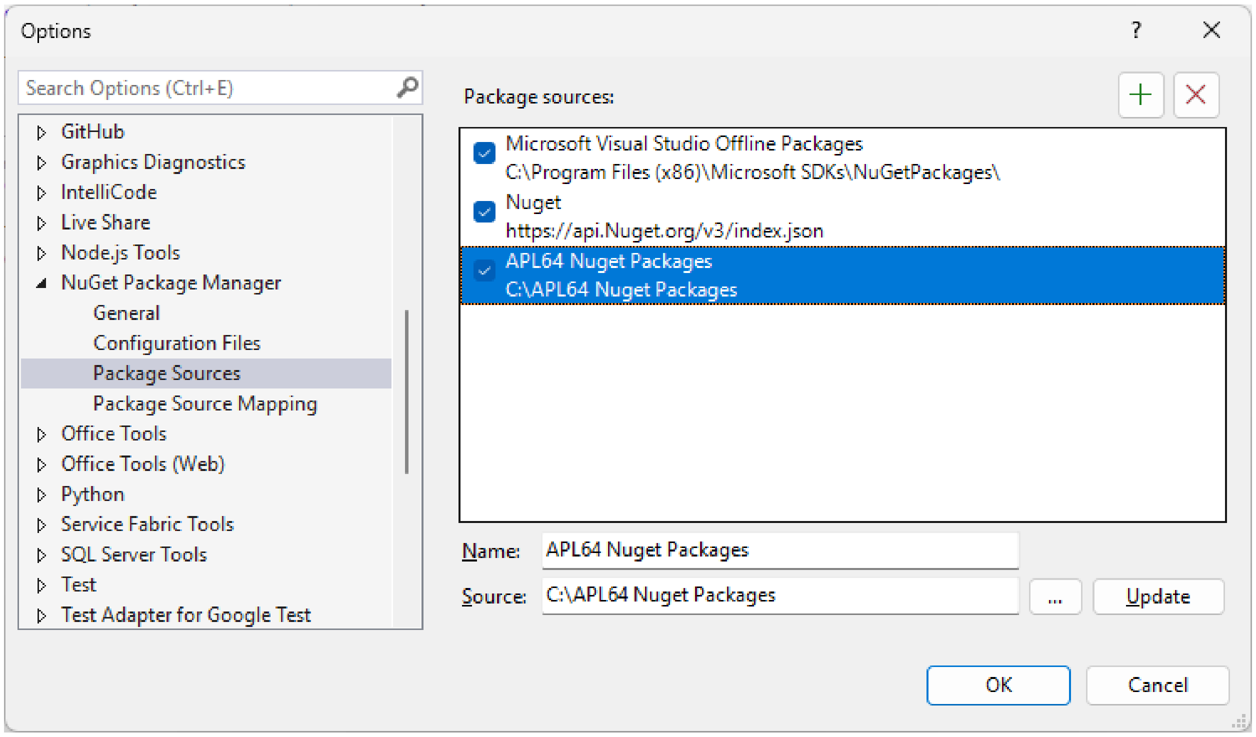
To:


C:\APL64 Nuget Packages
This PC > Windows (C:) > APL64 Nuget Packages
Search APL64 Nuget Packages
New
Sort
View
Details
Src
This PC
Windows (C:)
FLAC_MUSIC (G:)
1 item 1 item selected 2.33 MB

Name	Date modified	Type	Size
APLNext_LLC.CPCRoots.0.0.0.1.nupkg	2/10/2025 7:20 PM	NUPKG File	2,389 KB

Set-up the Visual Studio Package Sources

The publish folder containing the APL64 Nuget package must be included in the Visual Studio ‘package sources’:



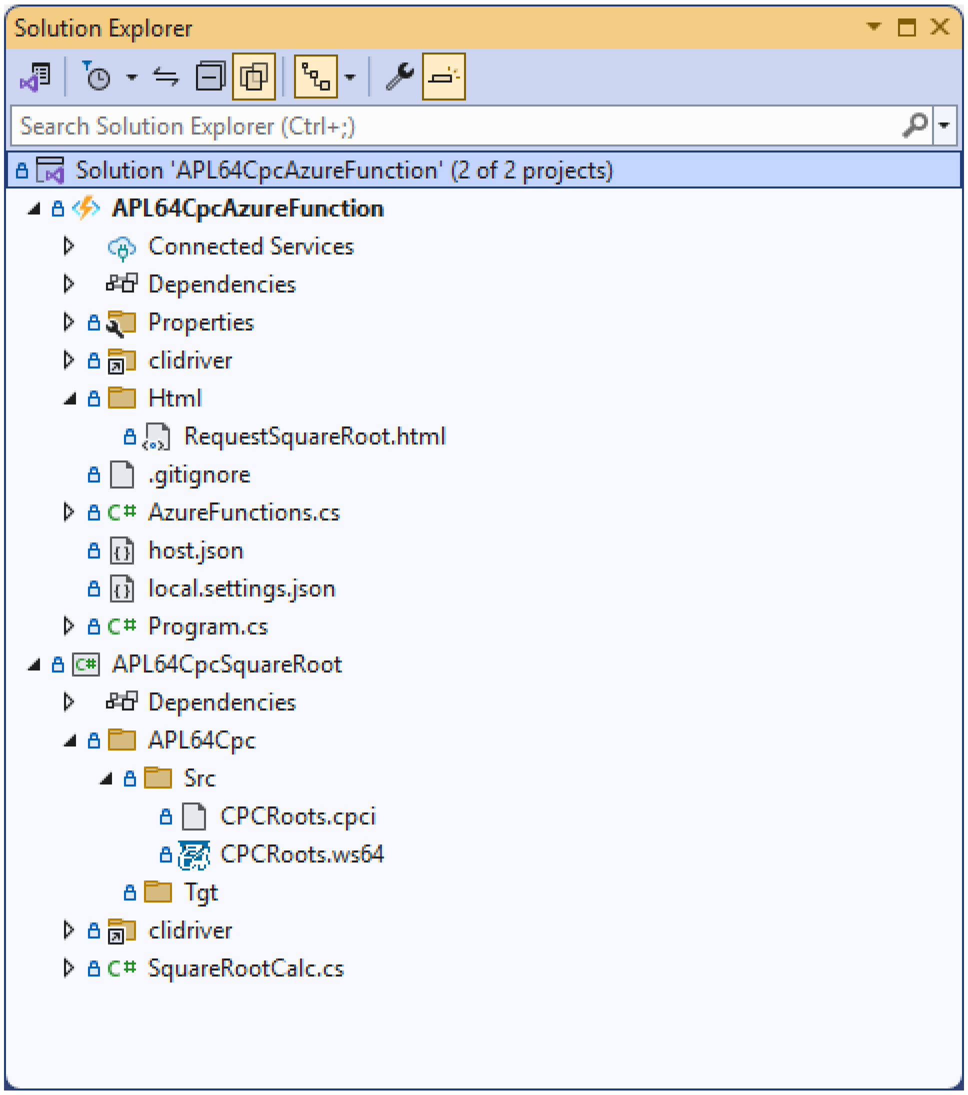
Package sources:

Microsoft Visual Studio Offline Packages
C:\Program Files (x86)\Microsoft SDKs\NuGetPackages\
Nuget
https://api.Nuget.org/v3/index.json
APL64 Nuget Packages
C:\APL64 Nuget Packages

Name: APL64 Nuget Packages
Source: C:\APL64 Nuget Packages

Install the APL64 CPC Nuget package into the Project

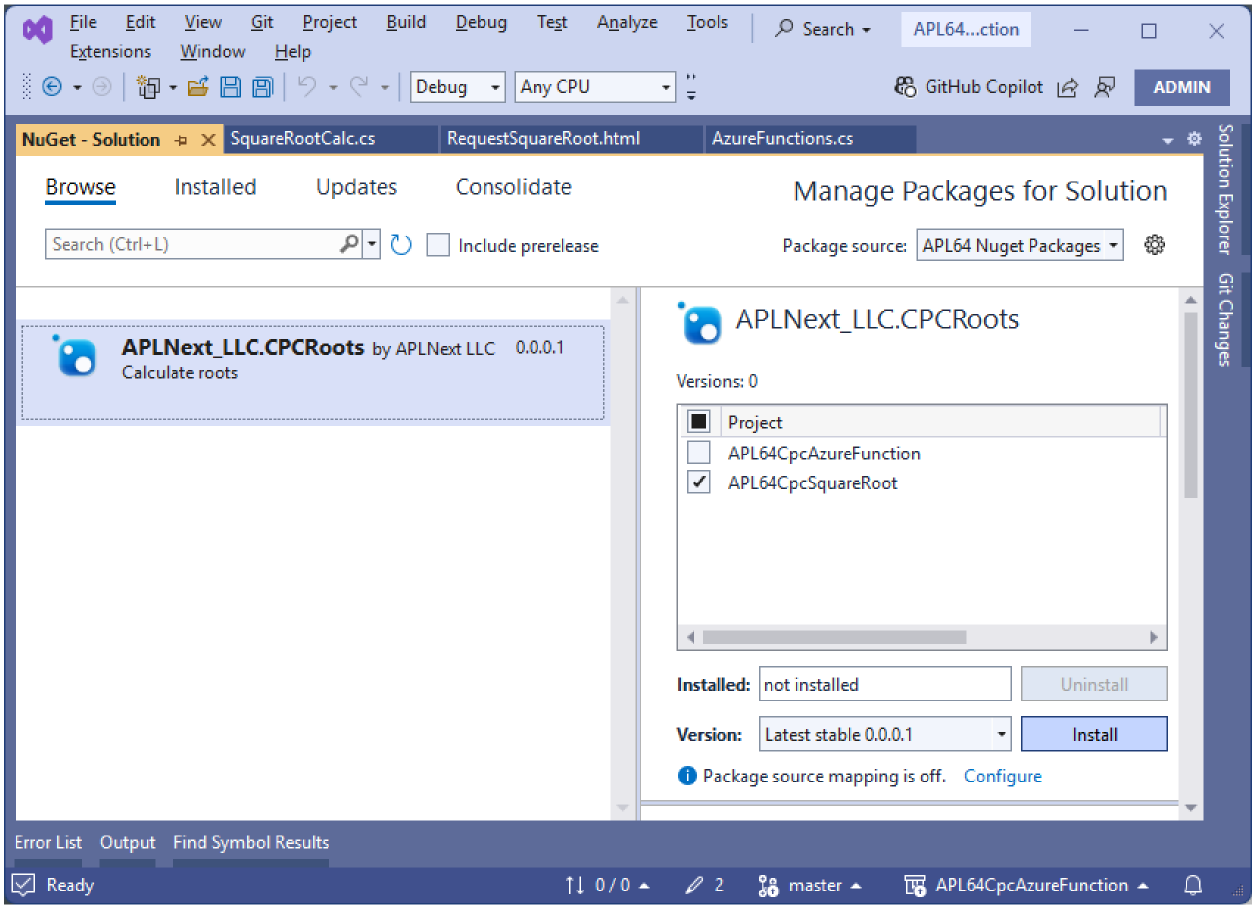
Select the Solution node of the Visual Studio Solution Explorer:



```
Solution 'APL64CpcAzureFunction' (2 of 2 projects)
  APL64CpcAzureFunction
    Connected Services
    Dependencies
    Properties
    clidriver
    Html
      RequestSquareRoot.html
    .gitignore
    AzureFunctions.cs
    host.json
    local.settings.json
    Program.cs
  APL64CpcSquareRoot
    Dependencies
    APL64Cpc
      Src
        CPCRoots.cpci
        CPCRoots.ws64
      Tgt
    clidriver
    SquareRootCalc.cs
```

Use the **Tools | Nuget Package Manager | Manage Nuget Packages for Solution** menu item in Visual Studio to present the Manage Packages for Solution dialog. Select the APL64 CPC Nuget package, and install it into the APL64CpcSquareRoot project of the solution.

The APL64 CPC is used only in the APL64CpcSquareRootProject, so the APL64 CPC Nuget package needs to be installed only in that project.



Package source: APL64 Nuget Packages

APLNext_LLC.CPCRoots by APLNext LLC 0.0.0.1
Calculate roots

APLNext_LLC.CPCRoots

Versions: 0

	Project
☐	APL64CpcAzureFunction
✓	APL64CpcSquareRoot

Installed: not installed — Uninstall

Version: Latest stable 0.0.0.1 — Install

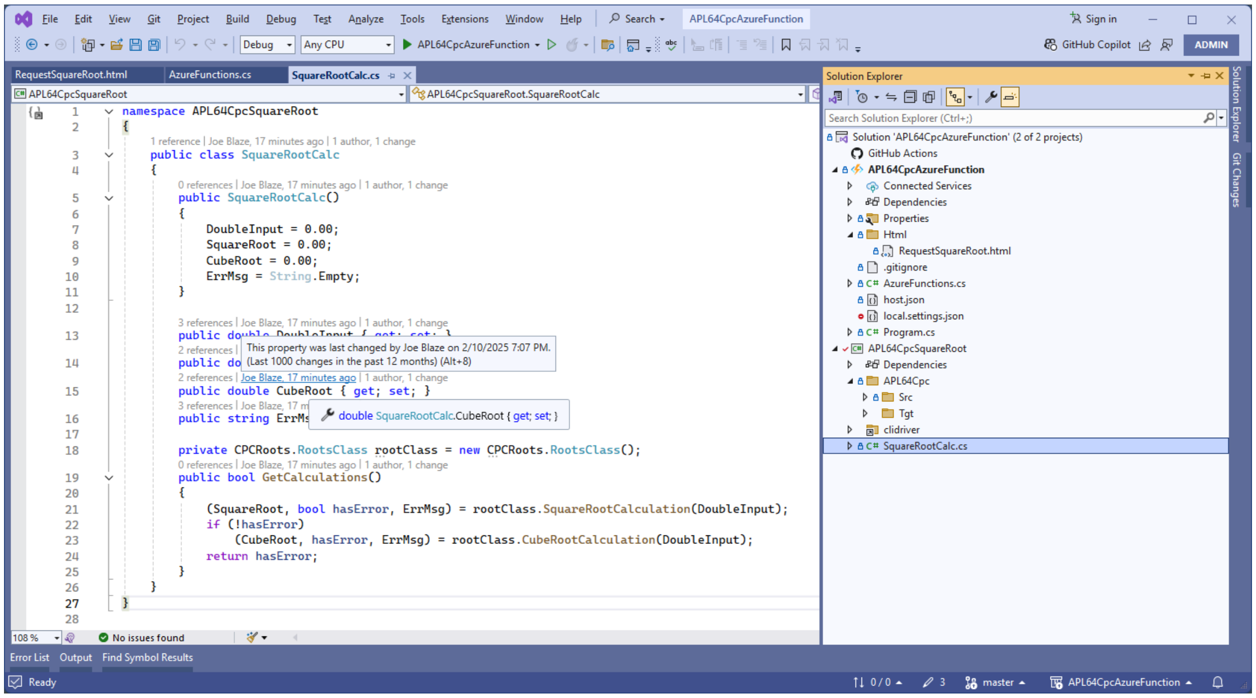
Package source mapping is off. Configure

Update the SquareRootCalc.cs Source Code

Replace the source code in the SquareRootCalc.cs code file in the APL64CpcSquareRoot project with:

```
namespace APL64CpcSquareRoot
{
  public class SquareRootCalc
  {
    public SquareRootCalc()
    {
      DoubleInput = 0.00;
      SquareRoot = 0.00;
      CubeRoot = 0.00;
      ErrMsg = String.Empty;
    }

    public double DoubleInput { get; set; }
    public double SquareRoot { get; set; }
    public double CubeRoot { get; set; }
    public string ErrMsg { get; set; }
```

```
        private CPCRoots.RootsClass rootClass = new CPCRoots.RootsClass();
        public bool GetCalculations()
        {
            (SquareRoot, bool hasError, ErrMsg) = rootClass.SquareRootCalculation(DoubleInput);
            if (!hasError)
                (CubeRoot, hasError, ErrMsg) = rootClass.CubeRootCalculation(DoubleInput);
            return hasError;
        }
    }
}
```

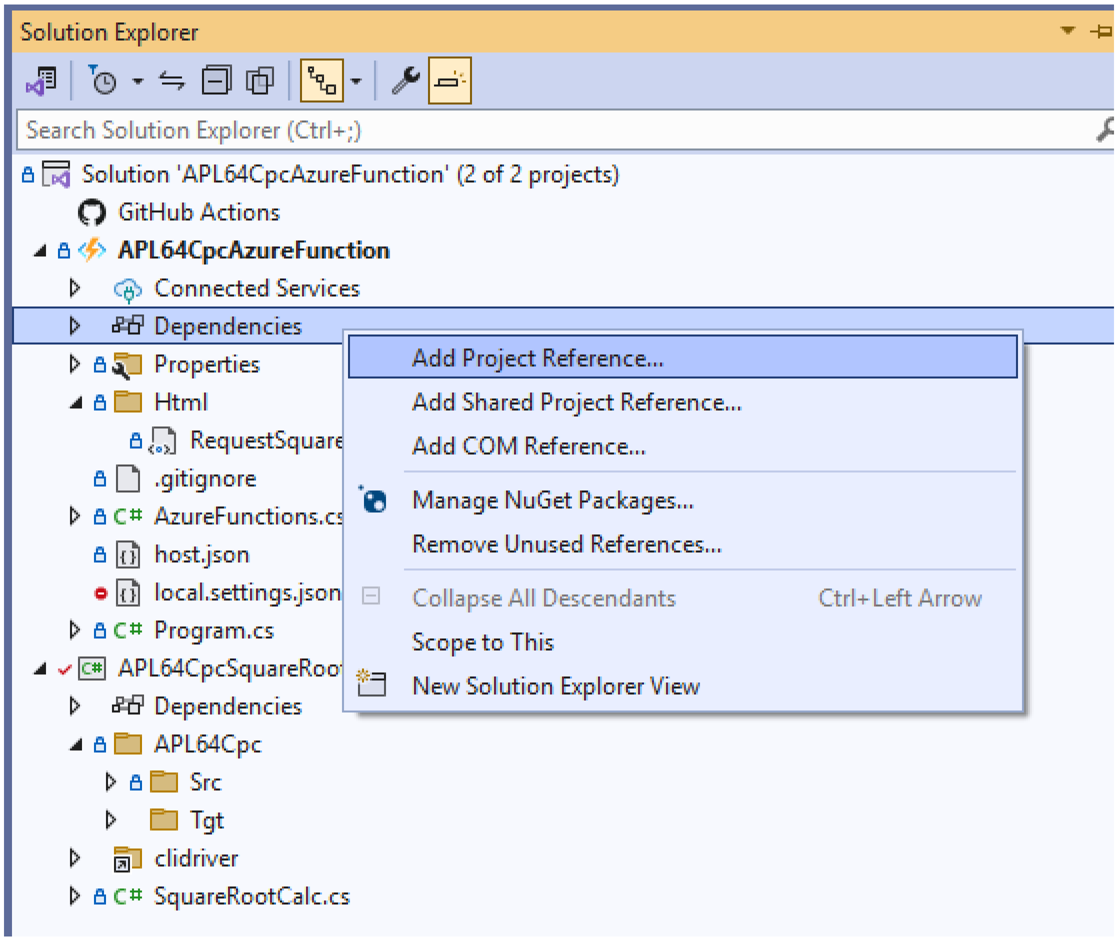


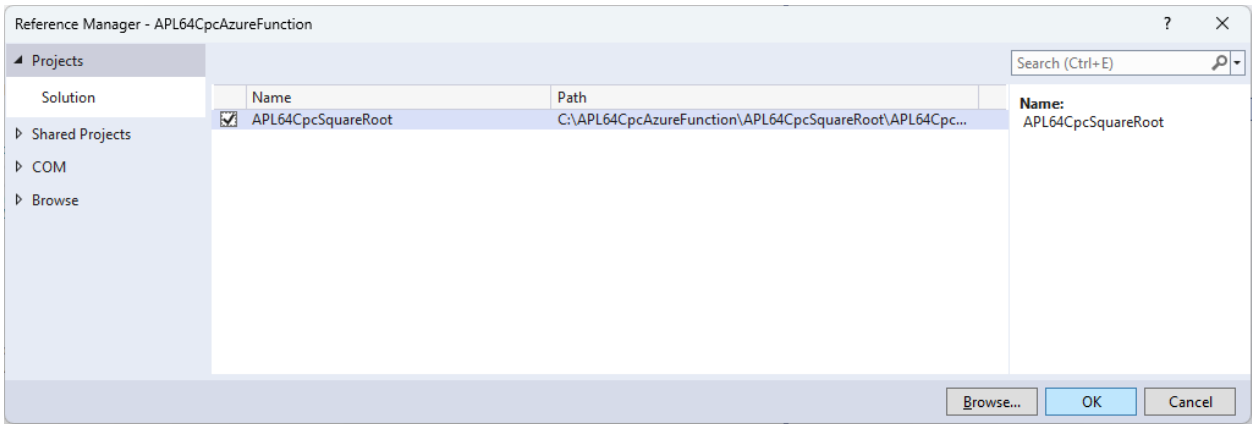
```
namespace APL64CpcSquareRoot
{
    1 reference | Joe Blaze, 17 minutes ago | 1 author, 1 change
    public class SquareRootCalc
    {
        0 references | Joe Blaze, 17 minutes ago | 1 author, 1 change
        public SquareRootCalc()
        {
            DoubleInput = 0.00;
            SquareRoot = 0.00;
            CubeRoot = 0.00;
            ErrMsg = String.Empty;
        }

        3 references | Joe Blaze, 17 minutes ago | 1 author, 1 change
        public double DoubleInput { get; set; }
        2 references | This property was last changed by Joe Blaze on 2/10/2025 7:07 PM.
        public do      (Last 1000 changes in the past 12 months) (Alt+8)
        2 references | Joe Blaze, 17 minutes ago | 1 author, 1 change
        public double CubeRoot { get; set; }
        3 references | Joe Blaze, 17 m
        public string ErrMs   double SquareRootCalc.CubeRoot { get; set; }

        private CPCRoots.RootsClass rootClass = new CPCRoots.RootsClass();
        0 references | Joe Blaze, 17 minutes ago | 1 author, 1 change
        public bool GetCalculations()
        {
            (SquareRoot, bool hasError, ErrMsg) = rootClass.SquareRootCalculation(DoubleInput);
            if (!hasError)
                (CubeRoot, hasError, ErrMsg) = rootClass.CubeRootCalculation(DoubleInput);
            return hasError;
        }
    }
}
```

The 'rootClass' is an instance of the APL64 CPC CPCRoots.RootsClassm which exposes the APL64 SquareRootCalculation and CubeRootCalculation public functions as .Net methods. When the GetCalculations() method is run, the user input (DoubleInput) is passed to the APL64 public functions which set the SquareRoot class SquareRoot, CubeRoot and ErrMsg property values.

Add the APL64CpcSquareRoot Project Dependency

Add the APL64CpcSquareRoot project as a dependency of the APL64CpcAzureFunction project:





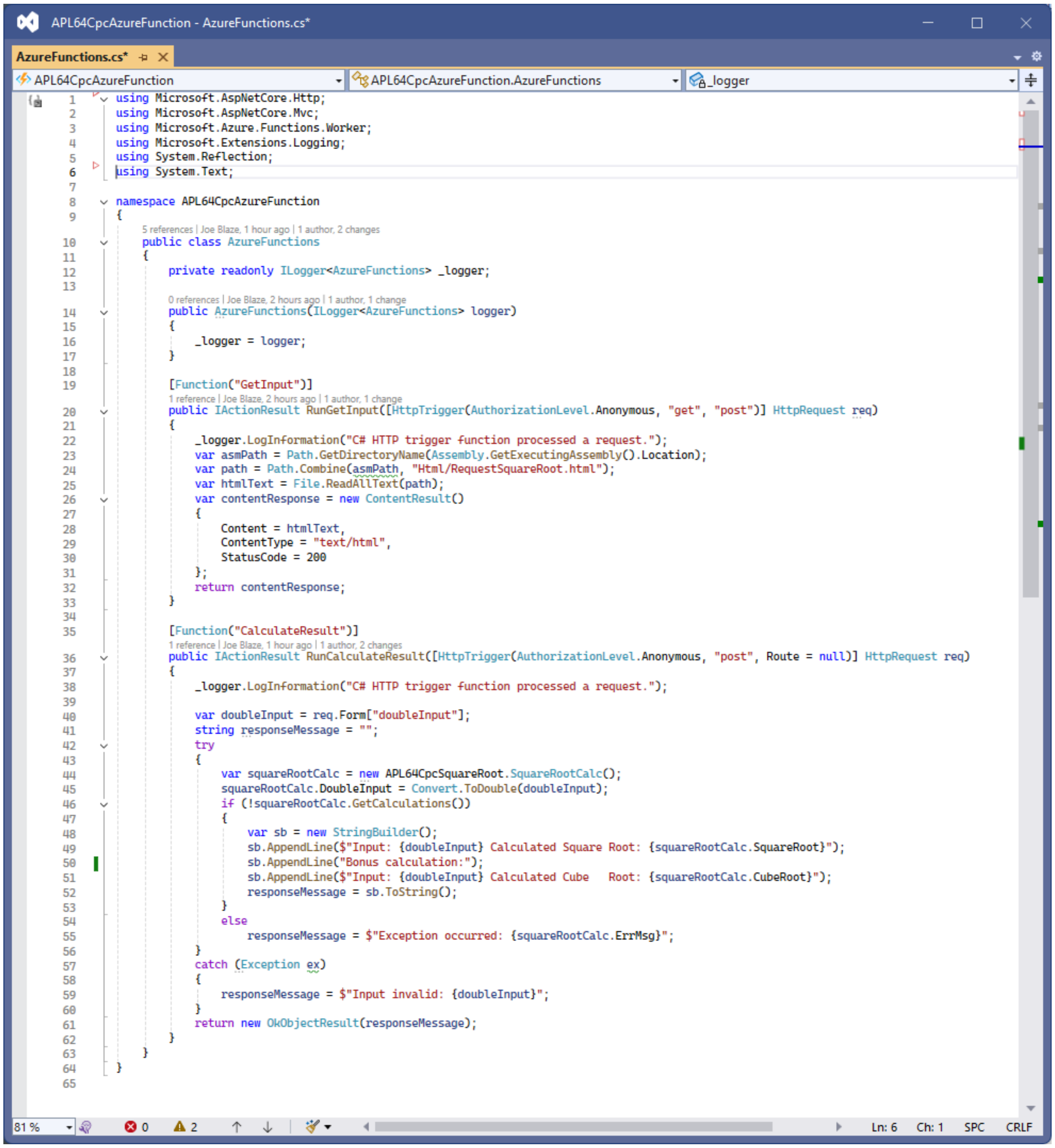
	Name	Path
✓	APL64CpcSquareRoot	C:\APL64CpcAzureFunction\APL64CpcSquareRoot\APL64Cpc...

Name: APL64CpcSquareRoot

Update the AzureFunctions.cs Code File

Update (replace) the code in the AzureFunctions.cs code file with:

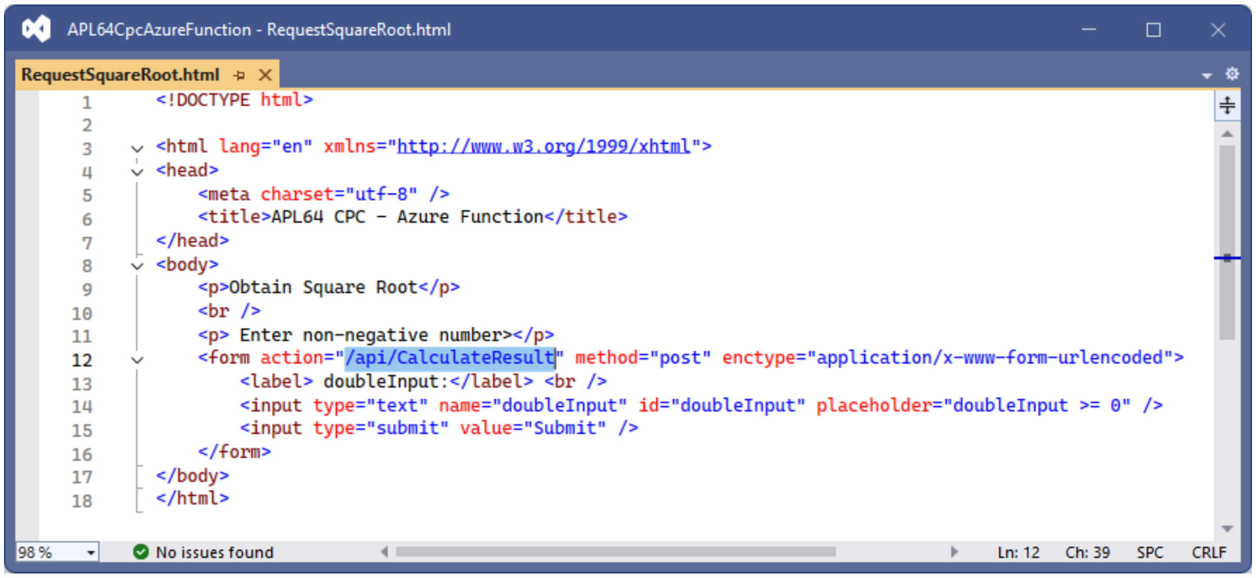
```
using Microsoft.AspNetCore.Http;
using Microsoft.AspNetCore.Mvc;
using Microsoft.Azure.Functions.Worker;
using Microsoft.Extensions.Logging;
using System.Reflection;
using System.Text;

namespace APL64CpcAzureFunction
{
  public class AzureFunctions
  {
    private readonly ILogger<AzureFunctions> _logger;

    public AzureFunctions(ILogger<AzureFunctions> logger)
    {
      _logger = logger;
    }

    [Function("GetInput")]
    public IActionResult RunGetInput([HttpTrigger(AuthorizationLevel.Anonymous, "get", "post")]
HttpRequest req)
    {
      _logger.LogInformation("C# HTTP trigger function processed a request.");
      var asmPath = Path.GetDirectoryName(Assembly.GetExecutingAssembly().Location);
      var path = Path.Combine(asmPath, "Html/RequestSquareRoot.html");
      var htmlText = File.ReadAllText(path);
      var contentResponse = new ContentResult()
      {
        Content = htmlText,
        ContentType = "text/html",
        StatusCode = 200
      };
      return contentResponse;
    }

    [Function("CalculateResult")]
    public IActionResult RunCalculateResult([HttpTrigger(AuthorizationLevel.Anonymous, "post",
Route = null)] HttpRequest req)
    {
      _logger.LogInformation("C# HTTP trigger function processed a request.");

      var doubleInput = req.Form["doubleInput"];
      string responseMessage = "";
      try
      {
        var squareRootCalc = new APL64CpcSquareRoot.SquareRootCalc();
        squareRootCalc.DoubleInput = Convert.ToDouble(doubleInput);
        if (!squareRootCalc.GetCalculations())
        {
          var sb = new StringBuilder();
          sb.AppendLine($"Input: {doubleInput} Calculated Square Root: {squareRootCalc.SquareRoot}");
          sb.AppendLine("Bonus calculation:");
          sb.AppendLine($"Input: {doubleInput} Calculated Cube Root: {squareRootCalc.CubeRoot}");
          responseMessage = sb.ToString();
        }
        else
          responseMessage = $"Exception occurred: {squareRootCalc.ErrMsg}";
      }
      catch (Exception ex)
      {
        responseMessage = $"Input invalid: {doubleInput}";
      }
      return new OkObjectResult(responseMessage);
    }
  }
}
```


APL64CpcAzureFunction - AzureFunctions.cs*
AzureFunctions.cs*
APL64CpcAzureFunction
APL64CpcAzureFunction.AzureFunctions
_logger
81 %
0
2
Ln: 6
Ch: 1
SPC
CRLF

```csharp
using Microsoft.AspNetCore.Http;
using Microsoft.AspNetCore.Mvc;
using Microsoft.Azure.Functions.Worker;
using Microsoft.Extensions.Logging;
using System.Reflection;
using System.Text;

namespace APL64CpcAzureFunction
{
    5 references | Joe Blaze, 1 hour ago | 1 author, 2 changes
    public class AzureFunctions
    {
        private readonly ILogger<AzureFunctions> _logger;

        0 references | Joe Blaze, 2 hours ago | 1 author, 1 change
        public AzureFunctions(ILogger<AzureFunctions> logger)
        {
            _logger = logger;
        }

        [Function("GetInput")]
        1 reference | Joe Blaze, 2 hours ago | 1 author, 1 change
        public IActionResult RunGetInput([HttpTrigger(AuthorizationLevel.Anonymous, "get", "post")] HttpRequest req)
        {
            _logger.LogInformation("C# HTTP trigger function processed a request.");
            var asmPath = Path.GetDirectoryName(Assembly.GetExecutingAssembly().Location);
            var path = Path.Combine(asmPath, "Html/RequestSquareRoot.html");
            var htmlText = File.ReadAllText(path);
            var contentResponse = new ContentResult()
            {
                Content = htmlText,
                ContentType = "text/html",
                StatusCode = 200
            };
            return contentResponse;
        }

        [Function("CalculateResult")]
        1 reference | Joe Blaze, 1 hour ago | 1 author, 2 changes
        public IActionResult RunCalculateResult([HttpTrigger(AuthorizationLevel.Anonymous, "post", Route = null)] HttpRequest req)
        {
            _logger.LogInformation("C# HTTP trigger function processed a request.");

            var doubleInput = req.Form["doubleInput"];
            string responseMessage = "";
            try
            {
                var squareRootCalc = new APL64CpcSquareRoot.SquareRootCalc();
                squareRootCalc.DoubleInput = Convert.ToDouble(doubleInput);
                if (!squareRootCalc.GetCalculations())
                {
                    var sb = new StringBuilder();
                    sb.AppendLine($"Input: {doubleInput} Calculated Square Root: {squareRootCalc.SquareRoot}");
                    sb.AppendLine("Bonus calculation:");
                    sb.AppendLine($"Input: {doubleInput} Calculated Cube   Root: {squareRootCalc.CubeRoot}");
                    responseMessage = sb.ToString();
                }
                else
                    responseMessage = $"Exception occurred: {squareRootCalc.ErrMsg}";
            }
            catch (Exception ex)
            {
                responseMessage = $"Input invalid: {doubleInput}";
            }
            return new OkObjectResult(responseMessage);
        }
    }
}
```

The ‘GetInput’ Azure function reads the text of the ‘RequestSquareRoot.html’ file in the project and uses that text as the content response of the end user’s request for a square root calculation. The ‘RequestSquareRoot.html’ file text is presented as a web page to the end user.

When the submit button is clicked, the ‘RequestSqareRoot.html’ page directs the end user’s request for a square root calculation to the ‘/api/CalculateResult’ url which is supported by the ‘CalculateResult’ Azure function:


APL64CpcAzureFunction - RequestSquareRoot.html

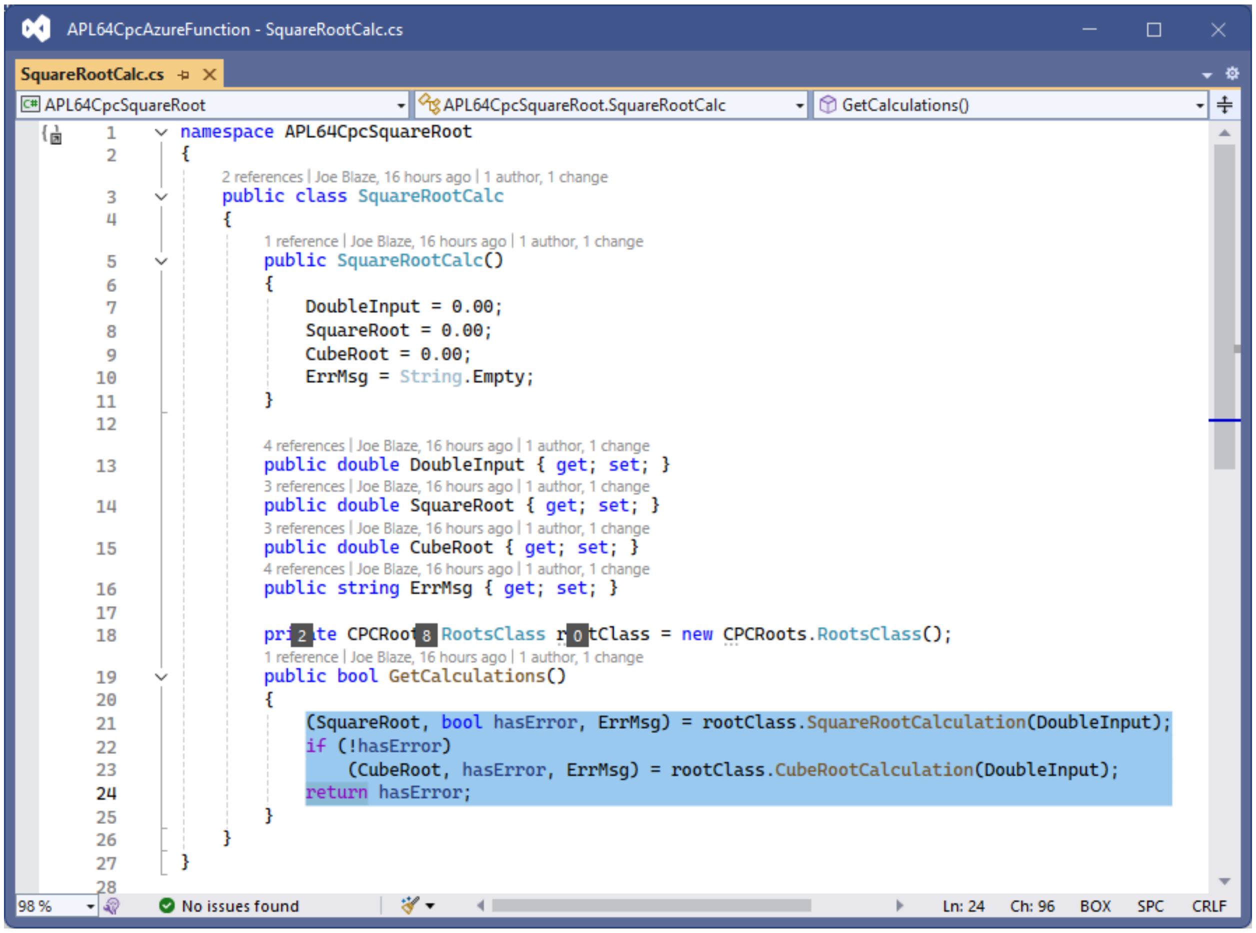
RequestSquareRoot.html

```
<!DOCTYPE html>

<html lang="en" xmlns="http://www.w3.org/1999/xhtml">
<head>
    <meta charset="utf-8" />
    <title>APL64 CPC - Azure Function</title>
</head>
<body>
    <p>Obtain Square Root</p>
    <br />
    <p> Enter non-negative number></p>
    <form action="/api/CalculateResult" method="post" enctype="application/x-www-form-urlencoded">
        <label> doubleInput:</label> <br />
        <input type="text" name="doubleInput" id="doubleInput" placeholder="doubleInput >= 0" />
        <input type="submit" value="Submit" />
    </form>
</body>
</html>
```

98 % No issues found Ln: 12 Ch: 39 SPC CRLF

When the end user provides the input and clicks the submit button, the input is provided to the 'CalculateResult' Azure function. After the calculations are completed, the 'CalculateResult' Azure function prepares the response and sends it to the end user's browser.

The 'CalculateResult' Azure function uses the APL64 public functions, exposed as .Net methods in the APL64 CPC Nuget package to obtain the calculated results. The 'CalculateResult' Azure function returns the result as an 'OkObjectResult' message to the client's browser:



APL64CpcAzureFunction - SquareRootCalc.cs

SquareRootCalc.cs

APL64CpcSquareRoot | APL64CpcSquareRoot.SquareRootCalc | GetCalculations()

```
namespace APL64CpcSquareRoot
{
    2 references | Joe Blaze, 16 hours ago | 1 author, 1 change
    public class SquareRootCalc
    {
        1 reference | Joe Blaze, 16 hours ago | 1 author, 1 change
        public SquareRootCalc()
        {
            DoubleInput = 0.00;
            SquareRoot = 0.00;
            CubeRoot = 0.00;
            ErrMsg = String.Empty;
        }

        4 references | Joe Blaze, 16 hours ago | 1 author, 1 change
        public double DoubleInput { get; set; }
        3 references | Joe Blaze, 16 hours ago | 1 author, 1 change
        public double SquareRoot { get; set; }
        3 references | Joe Blaze, 16 hours ago | 1 author, 1 change
        public double CubeRoot { get; set; }
        4 references | Joe Blaze, 16 hours ago | 1 author, 1 change
        public string ErrMsg { get; set; }

        pri2ate CPCRoot8RootsClass r0tClass = new CPCRoots.RootsClass();
        1 reference | Joe Blaze, 16 hours ago | 1 author, 1 change
        public bool GetCalculations()
        {
            (SquareRoot, bool hasError, ErrMsg) = rootClass.SquareRootCalculation(DoubleInput);
            if (!hasError)
                (CubeRoot, hasError, ErrMsg) = rootClass.CubeRootCalculation(DoubleInput);
            return hasError;
        }
    }
}
```

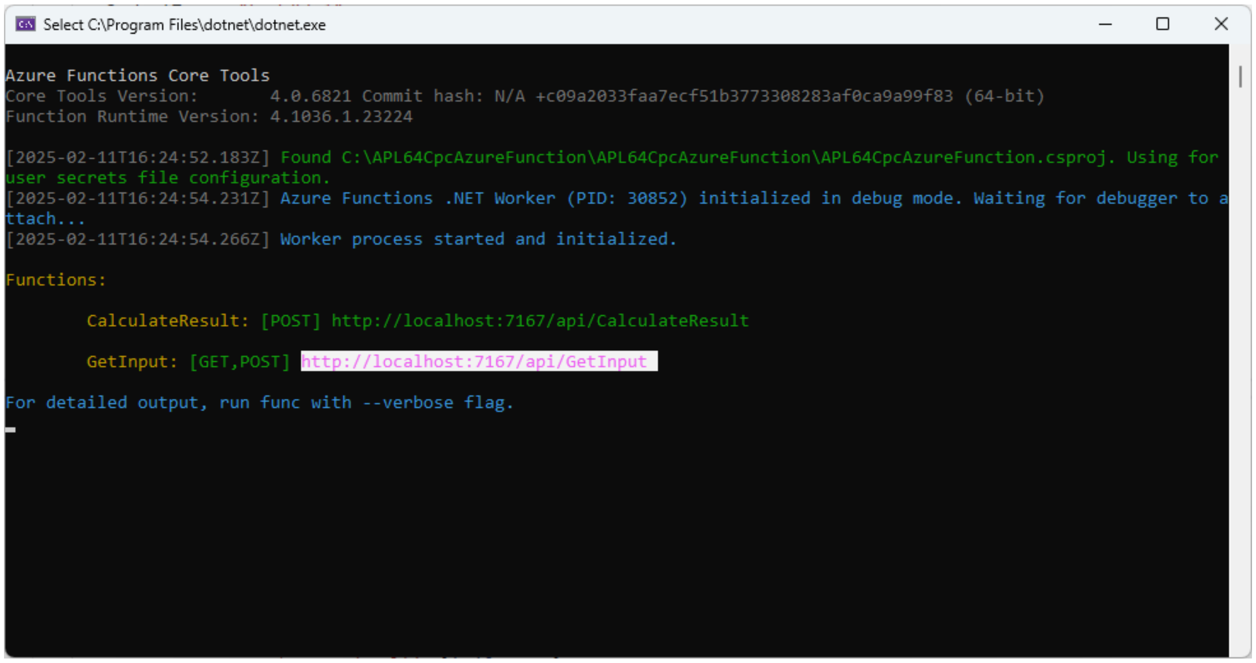
98 % | No issues found | Ln: 24 | Ch: 96 | BOX | SPC | CRLF

Run the Application

Click F5 in Visual Studio to run the application.

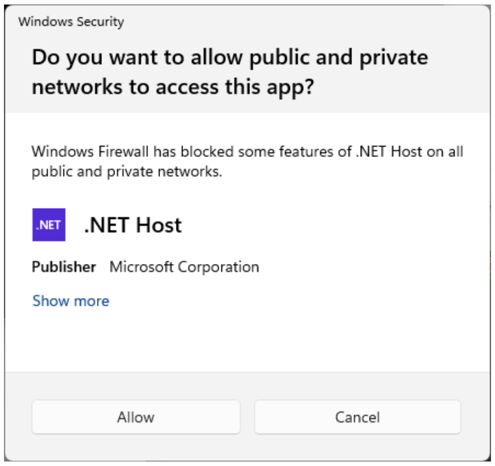
The Azurite Azure emulator creates the urls for the 'GetInput' and 'CalculateResults' Azure functions. In a production application the urls are published by Azure for (authorized) use by the clients of the application.

In Visual Studio, click F5 to run the Azurite simulator:



```
Azure Functions Core Tools
Core Tools Version:       4.0.6821 Commit hash: N/A +c09a2033faa7ecf51b3773308283af0ca9a99f83 (64-bit)
Function Runtime Version: 4.1036.1.23224

[2025-02-11T16:24:52.183Z] Found C:\APL64CpcAzureFunction\APL64CpcAzureFunction\APL64CpcAzureFunction.csproj. Using for
user secrets file configuration.
[2025-02-11T16:24:54.231Z] Azure Functions .NET Worker (PID: 30852) initialized in debug mode. Waiting for debugger to a
ttach...
[2025-02-11T16:24:54.266Z] Worker process started and initialized.

Functions:

        CalculateResult: [POST] http://localhost:7167/api/CalculateResult

        GetInput: [GET,POST] http://localhost:7167/api/GetInput

For detailed output, run func with --verbose flag.
```

Before the application starts, a Windows warning may be presented. Click the ‘Allow’ button to continue:

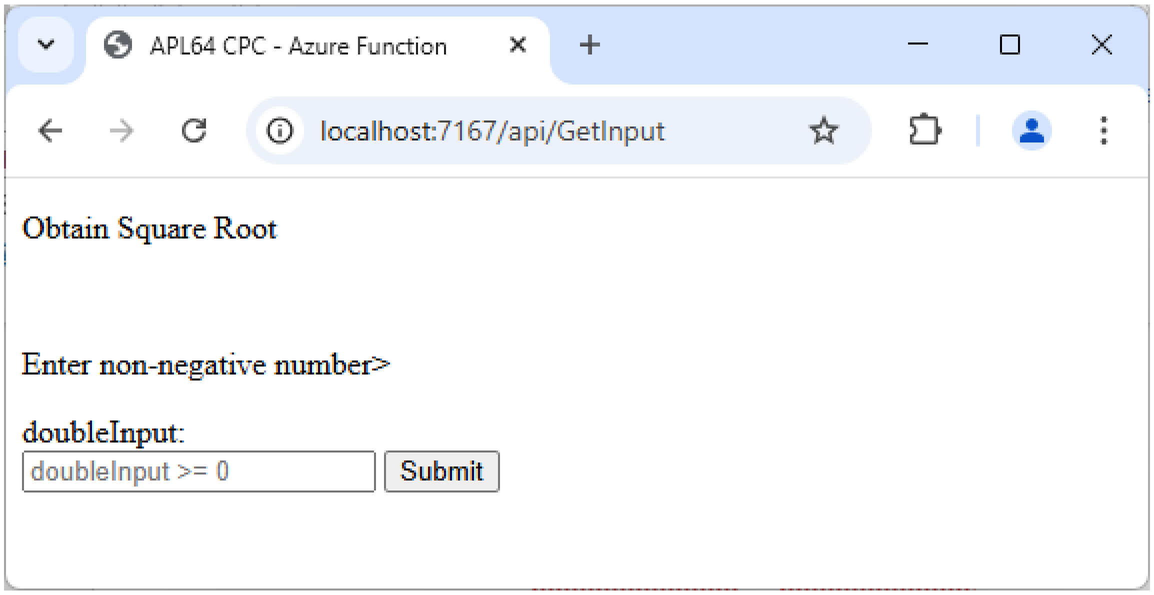


Do you want to allow public and private networks to access this app?

Windows Firewall has blocked some features of .NET Host on all public and private networks.

The developer of the application will provide the end user with the url of the application. To simulate the end user, copy the copy the ‘GetInput’ link: http://localhost:7167/api/GetInput, and paste it into a web browser to run the application:

Paste it into a Web browser and browse to the link, to display the input html-format page:



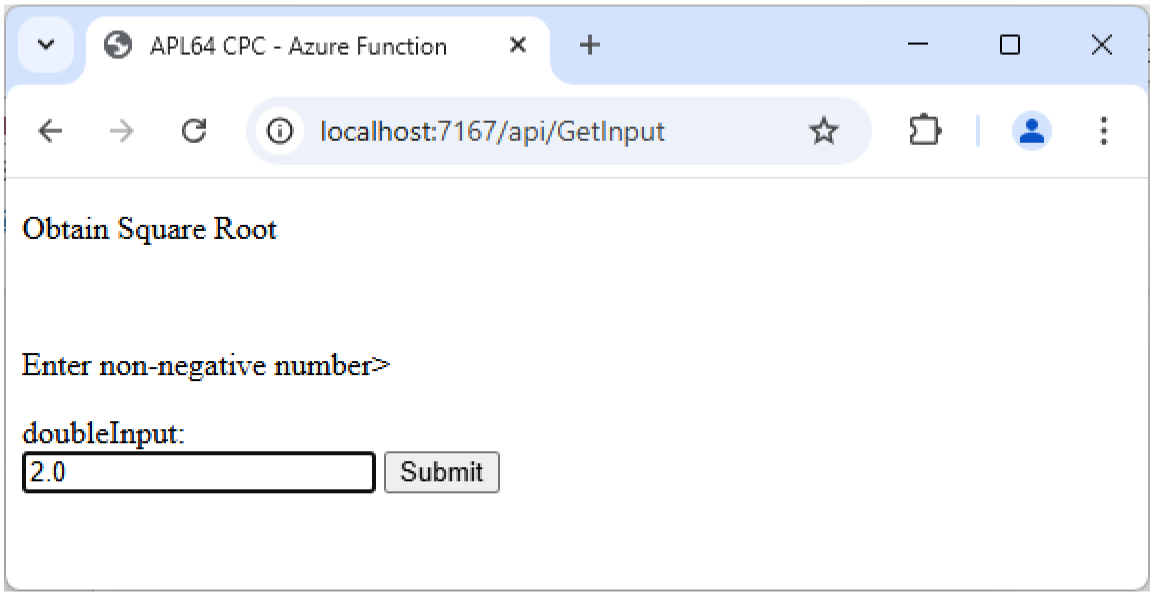
Obtain Square Root

Enter non-negative number>

doubleInput:

doubleInput >= 0 Submit

Enter a non-negative number into the Html input GUI control, and click the Html Submit button GUI control to initiate the request to calculate the square root of the input value:

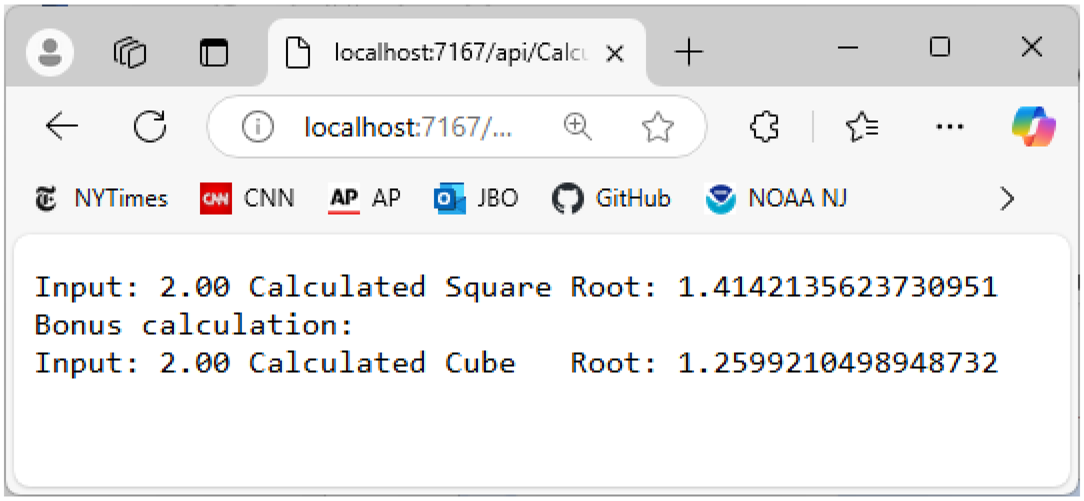


Obtain Square Root

Enter non-negative number>

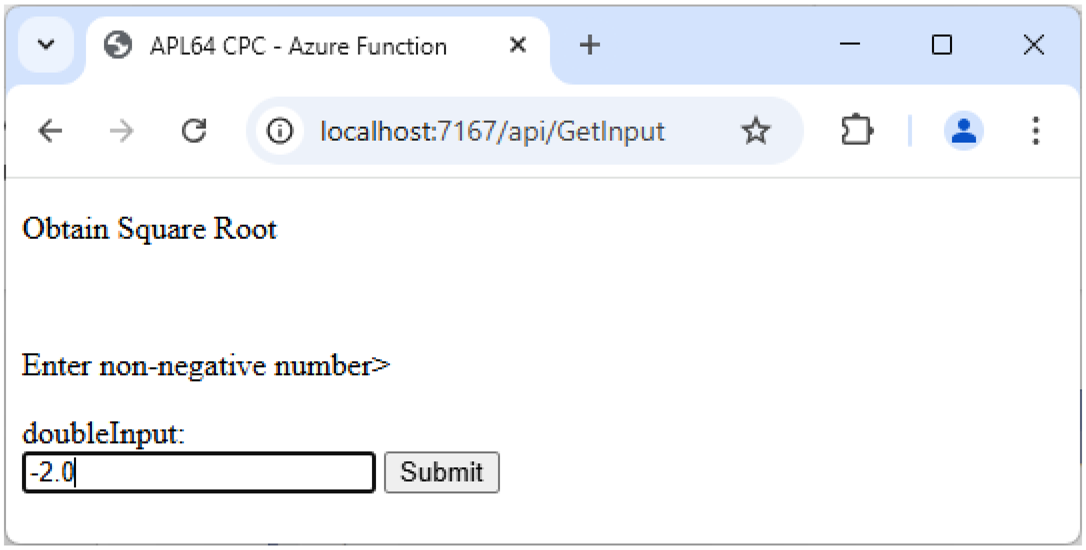
doubleInput:

2.0 Submit



```
Input: 2.00 Calculated Square Root: 1.4142135623730951
Bonus calculation:
Input: 2.00 Calculated Cube   Root: 1.2599210498948732
```

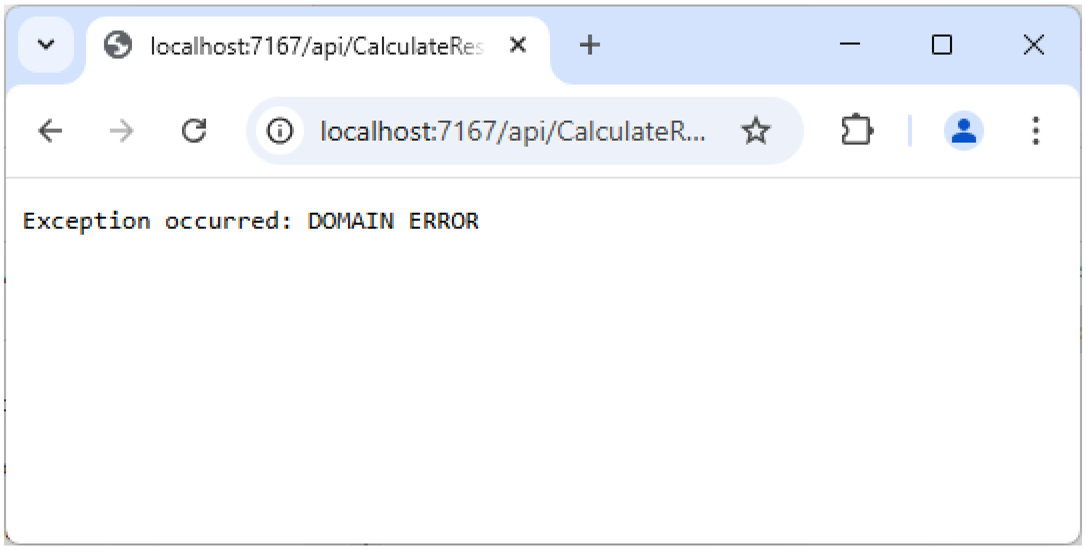
Browse to the application's data input url, enter a negative number, and click the 'Submit' button:



Obtain Square Root

Enter non-negative number>

doubleInput:
-2.0


localhost:7167/api/CalculateRes
localhost:7167/api/CalculateR...

```
Exception occurred: DOMAIN ERROR
```

Azure Function Deployment

This example is a demonstration project. Significant cosmetic enhancements, security and validation features need to be added for a production-ready project.

A no-cost Azure subscription is available to test an Azure Function application, and eventually make it available as a production application to users.

Publishing an Azure Function application is beyond the scope of this document. A basic workflow for the process of publishing an Azure Function application is available starting with the 'Publish the project to Azure' paragraph in the linked article.

Learn More

Contact support@apl2000.com for assistance using APL64 to create applications like an Azure Function application.