



Blazor Podcast Al











Contents

| Setup & Start |
|-----------------------------|
| Install .NET SDK |
| Create Project |
| Install Visual Studio Code7 |
| Launch Visual Studio Code9 |
| Update Project |
| Create GitHub Account |
| Get Personal Access Token |
| Launch Project in Browser |
| Implement |
| Provider Class |
| Item Component |
| Send Component |
| Home Component |
| Generate |
| Podcast Assistant |
| Custom Assistant |







Setup & Start

Install .NET SDK

tutorialr.com



Information - **Blazor** is the front-end framework from **Microsoft** based on **HTML**, **CSS** and **C#** using **.NET** to build web or hybrid mobile and desktop applications. To find out more about **Blazor** visit <u>blazor.net</u>.

First you will need to **Download** the latest .NET SDK, do so use a Browser and visit the Website at dot.net.



Information - .NET is the is the free, open-source framework from **Microsoft** to build modern applications for web with **Blazor** and **ASP.NET Core**, for mobile with **.NET MAUI**, for cloud with **.NET Aspire** and more.





Then choose **Download** which should display the **.NET SDK** for your platform of **Windows** or **Mac**.

| 🕼 🔲 🔤 Download .NET (Linux, macOS, an 🗙 🕂 | | | | - | o × |
|--|--|--|--|--------------------------|-------|
| ← C ∴ https://dotnet.microsoft.com/en-us/downloa | ad | | | ☆ C C 🎓 🕈 | 🥠 |
| Microsoft .NET why | y.NET 🗸 Features 🗸 Learn Docs 🗸 Downloads Com | Infunity LIVE TV | | All Microsoft ~ Search 🔎 | Î |
| Home / Download | | | | | |
| | | | | | |
| | Free. (| Cross-platform. Open source. | | | |
| | Dow | nload .N | ET | | |
| | | | | | |
| | | For windows | | | |
| | | | | | |
| | .NET 9.0 | | VET 8.0 | | |
| | Standard Term Support Recommended | | Long Term Support | | |
| | | | | | |
| | Download .NET SDK x64 | Dow | nload .NET SDK x64 | | |
| | Version 9.0.5, released May 13, 2025 | Version 8 | 1.0.16, released May 22, 2025 | | |
| | All_NET 9.0 downloads _NET 9.0 runtime All_NET | Tversions All.NET 8.0 | downloads .NET 8.0 runtime | | |
| | | | | | |
| | | | | | × |
| | | | | | edbac |
| | Other downloads | .NET Framework | Docker | | Ē |
| | | Looking for previous .NET Framework downloads for | Find official images for .NET and ASP.NET Core on the Microsoft | | |
| | | Windows? | Artifact Registry. | | |

Next select **Download** for the .**NET SDK** for **.NET 9.0** which varies based on your exact platform of **Mac** or **Windows** for example **Download .NET SDK x64** although your exact **Version** may be different or newer.



Information - .NET 9.0 SDK here is v9.0.300 for Windows x64 which was the one used for this Workshop.







Then the **Installer** for **.NET SDK** will begin **Downloading** and once it has been **Downloaded** it will show in **Downloads** for your **Browser** where you can **Open** it to launch the **Installer** for **.NET SDK** as follows:

| 🔀 Microsoft .NET SDK 9.0. | 300 (x64) Installer – 🗆 🗙 |
|---------------------------|--|
| | Microsoft .NET SDK 9.0.300 |
| | .NET SDK |
| | The .NET SDK is used to build, run, and test .NET applications. You can choose from multiple languages, editors, and developer tools, and take advantage of a large ecosystem of libraries to build apps for web, mobile, desktop, gaming, and IoT. We hope you enjoy it! |
| | If you plan to use .NET 9.0 with Visual Studio, Visual Studio 2022 17.12 or newer is required. <u>Learn more</u> . |
| .NET | By clicking Install, you agree to the following terms: Privacy Statement |
| | Telemetry collection and opt-out Licensing Information for .NET |
| | |

After **Opening** the **Installer** for the **.NET SDK** select **Install** to begin the installation process for **.NET SDK**.

| 闄 Microsoft .NET SDK 9. | 0.300 (x64) Installer — |] | × |
|-------------------------|---|-----|---|
| | Microsoft .NET SDK 9.0.300 | | |
| .NET | The installation was successful. The following products were installed: NET SDK 9.0.300 NET Runtime 9.0.5 .NET Runtime 9.0.5 .NET Windows Desktop Runtime 9.0.5 This product collects usage data .More information and opt-out <u>https://aka.ms/dotnet-cli-telemer</u> Resources NET Documentation <u>https://aka.ms/dotnet-docs</u> .SDK Documentation <u>https://aka.ms/dotnet-sdk-docs</u> .Release Notes <u>https://aka.ms/dotnet9-release-notes</u> .Tutorials <u>https://aka.ms/dotnet9-release-notes</u> | try | |
| | Close | | |

Finally, when installation has completed for the **.NET SDK**, you can select **Close** in the **Installer** as this completes the process of **Downloading** and **Installing** the **.NET SDK** for **Windows** or **Mac**.







Create Project

Once **.NET SDK** has been **Installed** you will need to create a **Project** using **Blazor**, to do this, if using **Mac** you need to go to **Finder**, search for **Terminal** and then select it to **Open** it, or if using **Windows** you need to go to **Start**, search for **Command Prompt** and then select it to **Open** it as follows:



Information – You can choose a different location for the **Project** in **Terminal** on **Mac** or **Command Prompt** on **Windows** if needed, but the default location should be fine for the **Workshop**.

Then you will need to create the **Project** using the **.NET SDK**, to do this from the **Terminal** on **Mac** or **Command Prompt** on **Windows** you need to *Copy* and *Paste* the following **Command** then press **Enter**:

dotnet new blazorwasm -o Blazor.Podcast.AI

Information – This will create a **Project** using **Blazor WebAssembly** called **Blazor.Podcast.Al** which is a special kind of application using **Blazor** that runs in your **Browser**.

Once the **Project** for **Blazor.Podcast.Al** which uses the template **Blazor WebAssembly Standalone App** has been created successfully, then within the **Terminal** on **Mac** or **Command Prompt** on **Windows** you need to switch to the **Folder** for the **Project** of **Blazor.Podcast.Al**. To do this *Copy* and *Paste* the following **Command** and then press **Enter**:

cd Blazor.Podcast.AI

Information – The **Command** of **cd** means change directory which is common to both the **Terminal** on **Mac** and **Command Prompt** on **Windows** to switch to a specified **Folder**.







Once done you should have switched to the Folder for the Project of Blazor.Podcast.AI as follows:



Then to add the **Package** for **Microsoft.Extensions.AI.OpenAI** you need to *Copy* and *Paste* the following **Command** and then press **Enter**:

dotnet add package Microsoft.Extensions.AI.OpenAI --prerelease

Information – This will add the **Package** for **Microsoft.Extensions.AI.OpenAI** allowing it to be used within the **Project** for the **Blazor** application. This **Package** integrates **AI Models** from **OpenAI** and is part of the broader Microsoft.Extensions.AI ecosystem that provides a unified way of working with AI services in .**NET**. To find out more about the **Package** you can visit <u>nuget.org/packages/Microsoft.Extensions.AI.OpenAI</u>.

Don't **Close** the **Terminal** on **Mac** or **Command Prompt** on **Windows** as you'll need it throughout the **Workshop** and to know what **Folder** to **Open** later in the **Workshop**, for example in this case it would be *C*:*Workshop**Blazor*.*Podcast*.*AI*.

Once the **Command** has completed in **Terminal** on **Mac** or **Command Prompt** on **Windows** this completes the process of creating the **Project** of **Blazor.Podcast.Al** and adding the **Package**.







Install Visual Studio Code

You will need to **Download** the latest **Visual Studio Code** for **Windows** or **Mac**, to do this use a **Browser** and visit the **Website** at <u>code.visualstudio.com</u> where you'll also find out more about **Visual Studio Code**.



Information – **Visual Studio Code** is the free and open-source code editor with support for every major programming language including **C#** which is used with **.NET** and with optional support for AI features.

Next select the **Download for Windows** option in this case for **Windows** or the option to **Download** for **Mac** although your exact **Version** of **Visual Studio Code** may be different or newer.

| Documentation for Visual Studio x + | | | | | - | ٥ | × |
|---|---|---|---|-----------------|------------|---|----------|
| ← C | | | | 습) 않 쇼 | ⊥ 🎓 | | 0 |
| 🗙 Visual Studio Code | Docs Updates Blog API Extension | s FAQ GitHub Copilot | D Search Docs | Downloads C Q 5 | \$ | | Î |
| Oveniew SETUP GET STARTED CONFIGURE | Than Do | Iks for downloading VS | Code! Llink. | See more | | | |
| EDIT CODE BUILD, DEBUG, TEST SOURCE CONTROL TERMINAL GITHIR COPILOT | Visual Studio Cod Get familiar with Visual Studio Code and lear Getting started | e documentation | | | | | |
| LANGUAGES NODEJES / JAVASCRIPT TYPESCRIPT | Set un Visual Studio Code | | | | | | I |
| РУТНОМ ЈАVА С++ С# | Follow the setup guide to install and configure VS Code for your OS. | Discover the key features of VS Code with the step-by-step tutorial. | Get started with GitHub Copil Al coding assistant. | ilot. your | | | |
| CONTAINER TOOLS DATA SCIENCE | Code with rich features | | | | | | |
| INTELLIGENT APPS AZURE DEMAYE | | 2º | - ② | | | | |
| TAMOTE DEV CONTAINERS REFERENCE | Code in any language Write code in your favorite programming language. | Version control Built-in support for git and many other source control providers. | Debugging Debug your code without lea editor. | aving your | | | |
| | - | | | | | | |





The **Installer** for **Visual Studio Code** will begin **Downloading** and once it has been **Downloaded** it will show in **Downloads** in your **Browser** where you can **Open** it to launch the **Installer** as follows:

| Icense Agreement | | |
|---|---|-----------------|
| Please read the following important information before continuing. | | 2 |
| Please read the following License Agreement. You must accept the terms of continuing with the installation. | this agreement before | • |
| This license applies to the Visual Studio Code product. Source | e Code for Visua | l |
| Studio Code is available at <u>https://github.com/Microsoft/vsc</u> | ode under the M | IT |
| license agreement at <u>https://github.com/microsoft/vscode/b</u> | lob/main/LICENS | <u>SE.txt</u> . |
| Additional license information can be found in our FAQ at | | |
| https://code.visualstudio.com/docs/supporting/taq. | | |
| | | |
| MICROSOFT SOFTWARE LICENSE TER | MS | |
| MICROSOFT SOFTWARE LICENSE TER MICROSOFT VISUAL STUDIO CODE | MS | |
| MICROSOFT SOFTWARE LICENSE TERM MICROSOFT VISUAL STUDIO CODE These license terms are an agreement between you and Micros based on where you live, one of its affiliates). They apply to the | MS oft Corporation (or software named | e |
| MICROSOFT SOFTWARE LICENSE TERM MICROSOFT VISUAL STUDIO CODE These license terms are an agreement between you and Micros based on where you live, one of its affiliates). They apply to the I accept the agreement | MS oft Corporation (or software named | |
| MICROSOFT SOFTWARE LICENSE TERM MICROSOFT VISUAL STUDIO CODE These license terms are an agreement between you and Micros based on where you live, one of its affiliates). They apply to the I accept the agreement I do not accept the agreement | MS oft Corporation (or software named | r |

You will need to select **I accept the agreement** and then select **Next** and keep selecting **Next** for the rest of the **Installer** as you don't need to change anything, once you get to the end select **Finish** as follows:









Launch Visual Studio Code

Once you have installed **Visual Studio Code** the **Installer** will have launched **Visual Studio Code**, but if **Visual Studio Code** was already **Installed**, then on **Mac** you need to go to **Finder** and then search for **Visual Studio Code** and then select it to **Launch** it, or if using **Windows** you need to go to **Start**, and then search for **Visual Studio Code** and then select it to **Launch** it as follows:

| 刘 File Edit Selection View Go Run Terminal Help | \leftrightarrow \rightarrow | ₽ Search | ₿× | |
|--|---|--|--|-------------|
| Welcome × | | | | |
| ↓ Weekcome × ↓ ↓ < | Visual Studio Code Editing evolved Start Cone Git Repository Cone Git Repository Cone Git Repository Cone Git Repository Cone Git Repository | Walkthr Cuto Cuto Cuto Cuto GitHu Co Get S Cet S More_ | DUGhS tarted with VS Code mice your editor, learn the basics, and start coding the Fundamentals the Copilot Destina tarted with C# Dev Kit Usabled tarted with Python Development Usable | |
| | | | | |
| | | Show welcome page on startup | | |
| | | | | A () |

Then in Visual Studio Code from the Side Bar select the top option of Explorer as follows:









Next in **Visual Studio Code** within **Explorer** select **Open Folder** and locate the **Folder** for your **Project**, this will be the one in **Terminal** on **Mac** or **Command Prompt** on **Windows** e.g. *C*:*Workshop\Blazor.Podcast.Al* and then choose **Select Folder** or if you cannot find it *Type* in the **Terminal** on **Mac** or **Command Prompt** on **Windows** the **Command** of **code** . followed by **Enter**, which should open the **Folder** as follows:

| 🗙 File Edit Selection View Go Run Terminal Help | $\leftarrow \rightarrow$ | , Blazor, Podcast, Al | · arian and a second s | |
|---|--|---|---|----|
| LOPLORER ···· ★] Welcome ★ ✓ ELAZOR PODCASTAI ···· ···· ★] ✓ > Luyout · ···· ···· ★] ✓ > Poportios ···· ···· ···· ···· ↓ ⑥ > Poportios ···· ···· ···· ···· ↓ ⑧ ···· ···· ···· ···· ···· ↓ ↓ ● ···· ···· ····· ···· ↓ ↓ ↓ ● ···· ···· ····· ···· ↓ ↓ ↓ ● ···· ····· ····· ↓ ↓ ↓ ● ···· ····· ····· ↓ ↓ ↓ ● ···· ······ ······ ↓ ↓ ↓ ● ···· ······ ······ ↓ ↓ ↓ ● ······· ······· <t< td=""><td>Visual Studi Editing evolved</td><td>o Code</td><td></td><td></td></t<> | Visual Studi Editing evolved | o Code | | |
| | Start ☐: New File ①: Open File >: Connect to Recent You have no recer | Do you trust the authors of the files in this folder? Code provides features that may automatically execute files in this ridide. Tyou don't trust the authors of these files, we recommend to continue m restricted mode as the files may be malicious. See our doct to learn mestic demode as the files may be malicious. See our doct to learn mestic demode as the files may be malicious. See our doct to learn mestic demode as the files may be malicious. See our doct to learn mestic demode as the files may be malicious. See our doct to learn mestic demode as the files may be malicious. See our doct to learn mestic demode as the files may be malicious. See our doct to learn mestic demode as the files may be malicious. See our doct to learn mestic demode as the files may be malicious. See our doct to learn mestic demode as the files may be malicious. See our doct to learn mestic demode as the files may be malicious. See our doct to learn mestic demode as the files may be malicious. See our doct to learn mestic demode as the files may be malicious. See our doct to learn mestic demode as the files may be malicious. See our doct to learn mestic demode as the files may be malicious. See our doct to learn mestic demode as the files may be malicious. See our doct to learn mestic demode as the files may be malicious. See our doct to learn mestic demode as the files may be malicious. See our doct to learn mestic demode as the files may be malicious. See our doct to learn mestic demode as the files may be malicious. See our doct to learn mestic demode as the files may be malicious. See our doct to learn mestic demode as the files may be malicious. See our doct to learn mestic demode as the files may be malicious. See our doct to learn mestic demode as the files may be malicious. See our doct to learn mestic demode as the files may be malicious. See our doct to learn mestic demode as the files may be malined as t | oughs tarted with VS Code mize your editor, learn the basics, and start coding the Fundamentals tarted with Python Development Tupdated tarted with Jupyter Notebooks Tupdated | |
| ⊘ > OUTLINE > > IMMENSE >> 0 0 0 ∆ 0 | | | | 80 |

Then choose **Yes**, **I trust the authors** in **Do you trust the authors of the files in this folder?** as this is the **Folder** for the **Project** you created and then once done select **Program.cs** from **Explorer** as follows:



Information – **Program.cs** is where the application is **Initialised** you will see **using** statements at the top for features in **.NET** or **Packages** along with **WebAssemblyHostBuilder** which is set up by default.





Update Project

Return to **Visual Studio Code** and with **Program.cs** selected in **Explorer** you will need to add some **using** statements, so below the line of **using Blazor.Podcast.AI** or the last **using** statement you need to *Copy* and *Paste* the following **Usings**:

using OpenAI; using System.ClientModel; using Microsoft.Extensions.AI;

Information – The first **using** statement will include functionality to use **OpenAI**, the next is needed for the **Credentials** and the last for functionality needed from the **Package** of **Microsoft.Extensions.AI.OpenAI**.

Then within **Program.cs** above the line of **await builder.Build().RunAsync();** you need to *Copy* and *Paste* the following **Code**:

```
var credential = new ApiKeyCredential("PersonalAccessToken");
var options = new OpenAIClientOptions()
{
    Endpoint = new Uri("https://models.inference.ai.azure.com")
};
var client = new OpenAIClient(credential, options);
var chat = client.GetChatClient("gpt-4o-mini").AsIChatClient();
builder.Services.AddSingleton(chat);
// Register Provider
```

Information – This Code is for the Client needed to use GitHub Models, starting with ApiKeyCredential with a placeholder for a Personal Access Token to be provided later in the Workshop. This is followed by an OpenAIClientOptions created with the Endpoint needed to access GitHub Models. Then there is an OpenAIClient using ApiKeyCredential and OpenAIClientOptions used to set up IChatClient which will use the AI Model from OpenAI of GPT-40 Mini. This is Registered with Dependency Injection where functionality such as IChatClient can be provided where it is needed. Finally, there is a Comment for Register Provider in Program.cs which will be returned to later when the Provider has been Implemented in the Workshop to Register it for Dependency Injection.







Once you have updated **Program.cs** with **Usings** and **Code** in **Visual Studio Code** it should be as follows:

| | le <u>E</u> dit <u>S</u> election <u>V</u> iew <u>G</u> o <u>R</u> un <u>T</u> e | erminal <u>H</u> elp | | ♀ Blazor.Podcast.Al | 8× | | |
|---------|---|---|--|--|------------|--------------------|------------|
| ſЪ | EXPLORER | C Program.cs | | | | | |
| E & & Q | ALZOR.PODCASTAL bin bin bin bin playout obj Pages Properties wwwroot usymotot usymotot Apprazor Apprazor Alazor.PodcastAl.csproj O Program.cs | O Program.cs using Microsoft.AspNetCore.Co using Blazor.Podcast.AI; using OpenAI; using System.ClientHodel; using Microsoft.Estensions.AI varbuilder.RootComponents.Add/eb ubuilder.RootComponents.Add/eb ubuilder.Services.AddScoped(sp ubuilder.services.AddScoped(sp accredential = new ApikeyCr | <pre>mponents.Web; pponents.WebAssembly.Hosting; ; Builder.CreateDefault(angs); >>("mapp"); adoutletv("head::after"); -> new HttpClient { BaseAddress - new dential("PresonalAccessToken");</pre> | Uri(builder.HostEnvironment.BaseAddress) }); | | | |
| | | <pre>15 Var options - new OpenAlCien 16 { 17 Endpoint - new OpenAlCien 19 var Clent - new OpenAlCien 19 var Clent - new OpenAlCien 10 var Clent - clent Activation 10 var Clent - clent Activation 11 builder.services.AddSingleton 22 24 25 await builder.Build().RunAsyn 26 </pre> | <pre>toptions() (/models.inference.si.azure.com[*]) (credential.options); ((*gpt-40-a=ini[*]).AsiChatClient(); (chat); c();</pre> | | | | |
| | | | | | | | |
| | > OUTLINE | | | | | | |
| × 6 | | | | | 1- 32 C-11 | Second LITE & CDIE | () C# P9 0 |

Visual Studio Code will need to be kept Open throughout the Workshop but don't worry if you Close it by mistake as you can just Launch it again by, if using a Mac you need to go to Finder, search for Visual Studio Code and then select it, or if using Windows you need to go to Start, search for Visual Studio Code and select it so that Visual Studio Code is Opened again. Then from Welcome in Visual Studio Code select Blazor.Podcast.AI from Recent and then select Program.cs from Explorer.

This completes **Visual Studio Code** being **Installed** and / or **Launched** for **Windows** or **Mac** along with opening the **Project** for **Blazor.Podcast.Al** and updating **Program.cs** as needed for **GitHub Models**.







Create GitHub Account

If you don't have an existing **GitHub Account** you will need to **Sign up** for one, to do so use a **Browser** and visit the **Website** at <u>github.com</u> which also includes more about **GitHub**.



Information – **GitHub** is where you can collaboratively store and share code for any language or platform such as **.NET** with **Repositories** where you can track and propose changes to help build and ship software.

Then select **Sign up** where you can input your **Email** then a **Password** and **Username** to use for **GitHub** along with your **Country/Region** such as **United Kingdom** and then select **Continue**.









Then you may be asked to Verify your Account and then to Confirm the Email Address for your Account.



Next you need to check your **Email** for one for **Your GitHub launch code** that would have been **Sent** to the **Email Address** you provided for your new **Account** which you can then *Copy* and *Paste* below **Enter Code** on the **Confirm your email address** and then select **Continue** to create your **GitHub Account**.

| 🔞 🗊 🗘 Sign in to GitHub - GitHub - GitHub - K + | | | | - | c | × |
|---|-----|-----|---|------|---|---|
| ← C (≜) https://github.com/login?return_to=https%3A%2F%2Fgithub.com%2Fdashboard | P 8 | 5 û | G | £≞ 1 | • | 🧑 |
| 0 | | | | | | |
| Sign in to GitHub | | | | | | |
| Your account was created × successfully. Please sign in to continue | | | | | | |
| Username or email address | | | | | | |
| Password Forgot password? | | | | | | |
| Sign in | | | | | | |
| Sign in with a passkey New to GitHub? Create an account | | | | | | |
| | | | | | | |
| Terms Privacy Docs Contact SitHub Support Manage cookies Do not share my personal information | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |

Information – This completes creating a new **GitHub Account** than can then be used to get a **Personal Access Token** to access **GitHub Models**.







Get Personal Access Token

| O D Addes-Gibbu Markeplace x + | - o × |
|--|-----------------------------------|
| ← C (a) https://github.com/marketplace/models | 표 \land ☆) ଓ 輝 🞓 … 🧔 |
| E 🕄 🔿 Marketplace / Models | e 🕖 to search 🛛 🕲 🔹 🗎 + 🔹 💿 🕅 😂 🧊 |
| Model: Select a Model \sim | (Preview) Give feedback |
| | |
| | |
| 0 | |
| Welcome to GitHub Models | |
| A catalog and playground of AI models to help you build AI features and products. | |
| Model switching: A single API key for all models & billing. | |
| Quick personal setup: GitHub PAT to install models in your projects. | |
| Free to start: No charges until you hit our rate limits. | |
| | |
| | |
| | |
| Select a model to get started, or <u>explore the full model catalog</u> . | |
| OpenAl GPT-4.1 Model gpt-4.1 outperforms gpt-40 across the board, with major gains in coding, instruction DeepSeek-V3-0324 Model DeepSeek-V3-0324 Model DeepSeek-V3-0324 Model | . (Model) is great at |
| | |
| | |
| | |
| | |
| | |

Information – **GitHub Models** is a catalogue and playground for **AI Models** including those from **OpenAI** to help build AI features and products easily accessed using a single **Endpoint** and **Personal Access Token** from **GitHub**. You can find out more about **GitHub Models** or try them out at <u>github.com/models</u>.

| 10 🗈 🐵 OpenAl x + | | | | - 0 | × |
|--|--|-----|------|----------|----------|
| \leftarrow C \bigcirc https://openai.com | Ą | ♠ ☆ | 3 √≡ | * | 0 |
| OpenAl 🛛 | | | Q | Log in | Î |
| Research Safety ChatGPT | | | | | |
| Sora API Platform For Business | What can I help with? | | | | |
| Stories Company News | Quiz me on vocabulary | | | | |
| | Search with ChatGPT Talk with ChatGPT Research Sora More | | | | |
| ຕມ ↓ Bcroll1 | o explore | | | | |

Information - **OpenAI** develops advanced **AI Models** including GPT or Generative Pre-trained Transformer that powers services such as ChatGPT that can understand and generate human-like text or DALL·E that can be used to create realistic images from text descriptions. To find out more about **OpenAI** visit <u>openai.com</u>.







If you have a **GitHub Account**, use a **Browser** and visit the **Website** at <u>github.com</u> and select **Sign in**, or once you have created your **GitHub Account** input your **Username or email address** and **Password**.



Then select Sign in to your GitHub Account and once done you will be taken to the Dashboard as follows:

| 🕼 🔲 🖸 GitHub Dashboard x + | | | | - o × |
|--|---|--|---|---|
| \leftarrow C \triangle https://github.com/dashboard | | | E | 표 🖂 🏠 🗘 🎼 🔹 🔹 🦉 |
| Dashboard | | | Q Type 🖉 to search | 8 • + • O n 🛆 😱 |
| Create your first project Ready to start building? Create a repository for a new idea or bring over an existing repository to keep contributing to it. Create repository Import repository | Home Ask Copilot | | GittHub Copilo Available for free Up 15 90 cdots and 2000 code completions per month, access Christ 2 55 centus and centum | ot entry * |
| | Learn with a tutorial project Introduction to GitHub Get started using GitHub in less than an hour. | GitHub Pages Create a site or blog from your GitHub repositories with GitHub Pages | Open Cc | ppilot |
| | Code with Copilot Develop with Al-powered code suggestions using GitHub Copilot, Codespaces, and VS Code. | Hello GitHub Actions Create a GitHub Action and use it in a workflow. | A virtual file system adapter for / | Azure Blob storage |
| | See more tutorial projects | | icon-theme Material Design icons for VS Coc | de |
| | Start a new repository for tutorialr-workshop A repository contains all of your project's files, revision history, and collaborator discussion. | Introduce yourself with a profile README Share information about yourself by creating a profile README, which appears at the top of your profile page. | webpack / webpack-de Serves a webpack app. Updates Documentation https://webpack server/. | vv-server the browser on changes. jsorg/configuration/dev- |
| | Repository name * name your new repository | tutorialr-workshop/README.md Cree | te Arrow Contract te Contract | |
| | Public Anyone on the internet can see this repository Private You choose who can see and commit to this repository | M 1'm interested in T'm currently learning Q 1'm looking to collaborate on Q How to reach me Q Pronous: M 6art: | | |

Information – **Dashboard** is where you get started with **GitHub** including being able to create your own **Repositories** where you can store and share any **Code** including being able to collaborate with others or contribute **Code** to others on **GitHub**.





tutorialr.com

Once in the **Dashboard** for your **GitHub Account** select the **Avatar** from the top of the **Dashboard** to display the **Account Menu** as follows:

| 🕅 🗊 🖸 GitHub Dashboard x + | | | | - | o × |
|---|---|--|---|--|------|
| ← C | | | B | ⓒ [☆) ଓ ৫ 🔮 | 🥠 |
| Dashboard | | | Q Type () to search | tutorialr-workshop | ₹× |
| Create your first project Ready to start building? Create a repository for a new idea or bring over an existing repository to keep contributing to it. Create repository Import repository | Home Ask Copilot | nt O Find issues assigned to me | GitHub Copilot Available for free Up to 50 chats and 2000 code completions per month, access to Claude 35 Segmet, and more. Open Core | Set status A Your profile Your repositories S Your Copilot I'' Your projects '' Your stars Our stars Our stars | |
| | Learn with a tutorial project Introduction to GitHub Get started using GitHub in less than an hour. | GitHub Pages Create a site or blog from your GitHub repositories with GitHub Pages. | Explore repositories | Your organizations Your organizations Your enterprises Your sponsors Try Enterprise | Free |
| | Code with Copilot Develop with Al-powered code suggestions using GitHub Copilot, Codespaces, and VS Code. | Hello GitHub Actions Create a GitHub Action and use it in a workflow. | ☆ 727 ● Go | Settings GitHub Website | New |
| | See more tutorial projects | | Material Design icons for VS Code | 대 GitHub Docs 요 GitHub Support 炅 GitHub Community | |
| | Start a new repository for tutorialr-workshop A repository contains all of your project's files, revision history, and collaborator discussion. | Introduce yourself with a profile README Share information about yourself by creating a profile README, which appears at the top of your profile page. | webpack / webpack-dev Serves a webpack app. Updates th Documentation https://webpack.js server/. 2 7 8 | [→ Sign out | |
| | name your new repository | tutorialr-workshop/README.md Creat | Explore more → | | |
| | Public Anyone on the internet can see this repository Private You choose who can see and commit to this repository | 2 - M I'm interested in 3 - ∑ I'm currently learning 4 - ∉ I'm looking to collaborate on 5 - ∉ Now to reach me 6 - @ Promouns: 7 - 4 Fun fort | | | |

Then from the Account Menu select Settings to display the Settings for your GitHub Account.

| 🕼 🗖 🕥 Your profile | × + | | | - @ × |
|---------------------|---|--|---|---------------------|
| ← C | m/settings/profile | | | 😬 \land 🏠 🌀 🎼 🍫 🚥 🧔 |
| ■ ○ Settings | | | Q Type 📝 to search |) 🛛 • + • 💿 n 🖻 💼 |
| E Settings | tutorialr-workshop (tu your personal account A Public profile Account Account Account Accessibility Notifications Access Billing and licensing Emails Password and authentication SSH and GPC keys Conganizations Entreprises Code, planning, and automation Repositories Code, planning, and automation Repositories Code planel Code planel Code planel Saved replies | torialr-workshop) Public profile Name Vour name may appear around Giffub where you contribute or are mentioned. You can remove it at any time. Public email Select a verified email to display Vou can winage verified email addresses in your email settings Bio Tell us a little bit about yourself Tell us a little bit about yourself Don't specify VR CR CRCD ID CRCD ID CRCD DID Social acounts V Into social profile I Verified CRCD Pooled CRCD ID Social acounts Verified CRCD Pooled | C to your personal profile Profile picture ↓ Edit | |
| | Security ① Code security | Ø Link to social profile 2 Ø Link to social profile 3 | | |





tutorialr.com

Next from **Settings** select **Developer settings** from the bottom of the options that includes **Public profile**.

| 🗇 🗖 🖸 GitHub Apps | < + | | | - a × |
|---------------------------------|------------------------|--|-------------------------|--------------------------|
| ← C | /apps | | | 표 \Lambda ☆) ଓ 🕼 🎓 🚥 🥠 |
| E Settings / Developer Settings | | | Q Type 🕜 to search | |
| | 8 GitHub Apps | GitHub Apps | | |
| | A OAuth Apps | | | |
| | Personal access tokens | × | | |
| | | | | |
| | | 88 | | |
| | | No GitHub Apps | | |
| | | Want to build something that integrates with and extends GitHub? Regist | ter a new GitHub App to | |
| | | get started developing on the did tab Ari. | | |
| | | New GitHub App | | |
| | | View documentation | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | © 2025 GitHub, Inc. Terms Privacy Security Status Docs Contact Manage cookies Do not share | my personal information | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |

Then in **Developer Settings** select **Personal access tokens** and then select **Fine-grained tokens**.

| Fine-grained Personal Access Tole x + | | - | Ċ | x c |
|---|--------------------------------|-------------|---|-----|
| ← C | 田 A* ☆ ③ | £≞ 1 | 8 | 🥠 |
| | Q. Type [] to search (3) + + + | 0 11 | | |
| 88 GitHub Apps A OAuth Apps Personal access tokens Time-grained tokens Tokens (classic) Nee • 2025 GitHub, Inc. | International access tokens | | | |

Information – This is where Personal Access Tokens are created needed to access GitHub Models.







Next in **Fine-grained tokens** select **Generate new token** and then enter a **Token name** which can be anything unique to your **GitHub Account**, for example *GitHub Models*, as follows:

| O New Fine-grained Personal Acces × + | | - a × |
|---|--|--------------------------|
| ← C | | 표 \Lambda ☆) ଓ 🎓 🎓 … 🥠 |
| E O Settings / Developer Settings | Q. Type [] to searc | h 🛛 🖉 🖓 + - O 🏗 🛆 😱 🕯 |
| 88 GitHub Apps A OAuth Apps Personal access tokens Fine-grained tokens Tokens (classic) | <section-header><text><text><section-header><section-header><section-header><section-header><section-header><section-header><section-header><section-header><section-header></section-header></section-header></section-header></section-header></section-header></section-header></section-header></section-header></section-header></text></text></section-header> | |
| | | |

Then at the bottom of **New fine-grained personal access token** select **Account permissions** and then next to **Models** select **Access** and choose the **Read-only** option as follows:

| C New Fine-grained Personal Acces × + | | | | | | - | ٥ | × |
|---------------------------------------|--|---------------------|---|--------------------|------|---|---|---|
| ← C | | | E | ± A ^N ☆ | 3 ∖≙ | 3 | | ø |
| | Git SSH keys Git SSH keys | Access: No access 👻 | | | | | | |
| | Interaction limits Interaction limits on repositories <u>Learn more.</u> | Access: No access 👻 | | | | | | |
| | Knowledge bases View knowledge bases for a user, <u>Learn more</u> . | Access: No access 👻 | | | | | | |
| | Models (Selected) Allows access to GitHub Models. <u>Learn more.</u> | Access: Read-only 👻 | | | | | | |
| | Plan View a user's plan. <u>Learn more.</u> | Access: No access 👻 | | | | | | |
| | Private repository invitations View a user's invitations to private repositories <u>Learn more.</u> | Access: No access 👻 | | | | | | |
| | Profile Manage a user's profile settings. <u>Learn more</u> | Access: No access 👻 | | | | | | |
| | SSH signing keys View and manage a user's SSH signing keys: <u>Learn more.</u> | Access: No access 👻 | | | | | | |
| | Starring List and manage repositories a user is starring. <u>Learn more.</u> | Access: No access 👻 | | | | | | I |
| | Watching List and change repositories a user is subscribed to. <u>Learn more</u> | Access: No access 👻 | | | | | | |
| | Generate token Cancel This token will be ready for use immediately. | | | | | | | |
| Q = 2025 | GitHub, Inc. Terms Privacy Security Status Docs Contact Manage cookies Do not share my perso | onal information | | | | | | |

Information – This will give the Personal Access Token to be created Permission to use GitHub Models.







Next at the bottom of **New fine-grained personal access token** select **Generate token** and in **New personal access token** select **Generate token** as follows:

| | | | 3 | • | |
|--|---|---------------------|--------|---|--|
| C O I inclos//gitmub.com/settings/personal-access-tokens/new | erene ane meanly a reac a flare and commences terms to the | | ມ ມ | ٢ | |
| | Git SSH keys Git SSH keys Learn more. | Access: No access 💌 | | | |
| | Interaction limits Interaction limits on repositories Learn more. | Access: No access 👻 | | | |
| | Knowledge bases View knowledge bases for a user. <u>Learn more</u> | Access: No access 👻 | | | |
| | Models (Selected) Allows access to GitHub Models. <u>Learn more.</u> | Access: Read-only 💌 | | | |
| | New personal access token X | ccess: No access 👻 | | | |
| | | | | | |
| | Your new personal access token GitHub Models will be ready for use immediately. It will expire on Thursday, July 10, 2025. | ccess: No access 👻 | | | |
| | GitHub Models grants you 1 account permission: | ccess: No access 👻 | | | |
| | Models (Read-only) | | | | |
| | | ccess: No access 👻 | | | |
| | Cancel Generate token | | | | |
| | List and manage repositories a user is starring. Learn more. | Access: No access + | | | |
| | Watching List and change repositories a user is subscribed to. <u>Learn more.</u> | Access: No access 👻 | | | |
| | Consult taken | | | | |
| | This token will be ready for use immediately. | | | | |
| | | | | | |
| 0 0 200 | 5 GitHub, Inc. Terms Privacy Security Status Docs Contact Manage.cookies Do not share my personal | linformation | | | |
| NP | | | | | |

Once Generate token has been selected the Personal Access Token will be created as follows:

| C Fine-grained Personal Access Tole X | + | | | | | - | ٥ | × |
|---------------------------------------|--|---|---------------------|---------|-----|----|---|----------|
| ← C | personal-access-tokens | | | B A ☆ C | £_≡ | \$ | | 0 |
| | | | Q Type [] to search | 8 - + | • • | n | 0 | |
| | 8 GitHub Apps A OAuth Apps Personal access tokens ∧ Fine-grained tokens Tokens (classic) C ● 20 | Comparison of the set | ation | | | | | |

Information – This **Personal Access Token** only has **Permission** to access **GitHub Models** so it won't be able to do anything to your **GitHub Account**, but you should keep it secret and not share it with anyone.







Then you need to *Copy* the **Personal Access Token** from **Fine-grained personal access tokens** and return to **Visual Studio Code** where **Program.cs** should be selected in **Explorer** as follows:



Then within **Program.cs** you will need to replace the placeholder of **PersonalAccessToken** by *Pasting* the **Personal Access Token** that you *Copied* from **Fine-grained personal access tokens** in **GitHub**.



Information – Your **Personal Access Token** can be used here like this as it will just be used on your **Mac** or **Windows** computer but should be it back to **PersonalAccessToken** if sharing anywhere including **GitHub**.







Next, within Visual Studio Code from the Menu select File and then Save as follows:

| × | File Edit Selection View Go Run Ter | rminal Help | \leftarrow \rightarrow | P Blazor.Podcast.Al | 8 ~ | |
|------|-------------------------------------|------------------------|---|---|------------------|--|
| ſ, | New Text File Ctrl+N | Program.cs | | | | II |
| | New File Ctrl+Alt+Windows+N | C Program.cs | | | | |
| ρ | New Window Ctrl+Shift+N | | Microsoft.AspNetCore.Components.Web; | | | La Contra de Cartos de Car |
| | New Window with Profile > | 2 using | Microsoft.AspNetCore.Components.WebAssembly.Hostin Blazor Podcast AT | | | Particular and the second |
| દુરુ | Open File Ctrl+O | 4 using | OpenAI; | | | |
| | Open Folder Ctrl+K Ctrl+O | 5 using | System.ClientModel; | | | |
| æ | Open Workspace from File | 7 5 USING | Microsoft.Extensions.Al; | | | |
| | Open Recent > | 8 var bu | uilder = WebAssemblyHostBuilder.CreateDefault(args) | | | |
| Б | Add Falder to Minderson | 9 builde 10 builde | er.RootComponents.Add <app>("#app"); er.RootComponents.Add<headoutlet>("head::after");</headoutlet></app> | | | |
| | Add Folder to Workspace | 10 00110 | | | | |
| | Save workspace AS | 12 builde | er.Services.AddScoped(sp -> new HttpClient { BaseAd | ldress = new Uri(builder.HostEnvironment. | BaseAddress) }); | |
| | Duplicate workspace | 13 14 var cr | <pre>redential = new ApiKeyCredential("github_pat");</pre> | | | |
| | Save Ctrl+S | 15 var op | <pre>ptions = new OpenAIClientOptions()</pre> | | | |
| | Save As Ctrl+Shift+S | 16 { | ndpoint = new Uni("https://models.inference.ai.azur | ve.com") | | |
| | Save All Ctrl+K S | | | | | |
| | Share | 19 var cl 20 var cl | <pre>lient = new OpenAIClient(credential, options); hat = client.GetChatClient("got-4o-mini").AsIChatCl</pre> | Lient(): | | |
| | Auto Save | 21 builde | er.Services.AddSingleton(chat); | | | |
| | Preferences | | | | | |
| | | | | | | |
| | Revert File | | <pre>builder.Build().RunAsync();</pre> | | | |
| | Close Editor Ctrl+F4 | | | | | |
| | Close Folder Ctrl+K F | | | | | |
| | Close Window Alt+F4 | | | | | |
| | Exit | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| (8) | | | | | | |
| | > OUTLINE | | | | | |
| | > TIMELINE | | | | | |
| × - | ⊗ 0 ∆ 0 | | | | | Ln 14, Col 54 Spaces: 4 UTF-8 CRLF () C# 🔀 🚨 |

Information – This will **Save** the changes you have made to **Program.cs** which includes the **Usings** and **Code** added for the **Client** along with the **Personal Access Token** to use **GitHub Models**.

This completes the process of getting a **Personal Access Token** from a **GitHub Account** and updating **Program.cs** in the **Project** for **Blazor.Podcast.AI** to include the **Personal Access Token** to be able to access **GitHub Models**.







Launch Project in Browser

Return to the Terminal on Mac or Command Prompt on Windows that was opened earlier as follows:



Information – If you closed the **Terminal** on **Mac** then you need to go to **Finder**, search for **Terminal** and then select it to **Open** it, or if you closed the **Command Prompt** on **Windows** you need to go to **Start**, search for **Command Prompt** and then select it to **Open** it. Then once opened you need to change directory using **cd** to the location for your **Project**, for example *cd Blazor.Podcast.Al*.

Then in **Terminal** on **Mac** or **Command Prompt** on **Windows** you then need to *Copy* and *Paste* the following **Command** and then press **Enter**:

| dotnet watch | |
|--------------|--|
|--------------|--|

Information – If **Terminal** on **Mac** or **Command Prompt** on **Windows** displays any **Errors**, then make sure that everything was entered correctly into **Program.cs** by going over the previous **Steps** to double check it matches what you have but once any corrections have been made and **Saved** or there are no **Errors** then the **Build** should proceed.







Once the **Command** has completed successfully the **Terminal** on **Mac** or **Command Prompt** on **Windows** should display as follows:

| Command Prompt - dotnet w × + v | - | | × |
|--|----------------------------|---------------------|------------|
| <pre>info : OK https://api.nuget.org/v3/registration5-gz-semver2/microsoft.extensions.ai.openai/index.json 273ms info : Restoring packages for C:\Workshop\Blazor.Podcast.AI\Blazor.Podcast.AI.csproj info : CACHE https://api.nuget.org/v3/vulnerabilities/index.json info : CACHE https://api.nuget.org/v3-vulnerabilities/2025.06.13.23.35.10/vulnerability.base.json info : CACHE https://api.nuget.org/v3-vulnerabilities/2025.06.13.23.35.10/vulnerability.base.json info : CACHE https://api.nuget.org/v3-vulnerabilities/2025.06.13.23.35.10/2025.06.14.11.35.11/vulnerability.updat info : Package 'Microsoft.Extensions.AI.OpenAI' is compatible with all the specified frameworks in project 'C:\Work Podcast.AI\Blazor.Podcast.AI.csproj'. info : PackageReference for package 'Microsoft.Extensions.AI.OpenAI' version '9.6.0-preview.1.25310.2' added to fil op\Blazor.Podcast.AI\Blazor.Podcast.AI.csproj'. info : Generating MSBuild file C:\Workshop\Blazor.Podcast.AI\obj\Blazor.Podcast.AI.csproj.nuget.g.targets. info : Writing assets file to disk. Path: C:\Workshop\Blazor.Podcast.AI\obj\project.assets.json log : Restored C:\Workshop\Blazor.Podcast.AI\Blazor.Podcast.AI.csproj (in 205 ms).</pre> | e.json shop\l e 'C:` | n Blazo \Work | or. Ksh |
| <pre>C:\Workshop\Blazor.Podcast.AI>dotnet watch dotnet watch</pre> | | | |

Information – You will be using *Copy* and *Paste* for each piece of **Code** but to avoid any issues the key thing to remember in **C#** is balance, so a **Curly Brace** of { will always have a counterpart of } also applies to **Square Brackets** which should have both [and] and **Brackets** which should have always have (and) along with **Double Quotes** which should be in pairs so if see " at the start there should be another " at the end. Also make sure where you see any **Semi-Colons** or ; to include them where needed as it can often be the smallest mistake that is easiest to fix makes your **Code** work when corrected although any indentation including **Tabs** won't affect any behaviour. **Errors** will give you an idea where to look including the line number of the **File** which will make them easier to find and give you some idea of what you did wrong so you can correct any mistake.







Also once the **Command** has completed in the **Terminal** on **Mac** or **Command Prompt** on **Windows** and there are no **Errors** then the following will be displayed in a **Browser**:



Information – If you don't see anything like this in a **Browser** then check anything you might have missed in the **Workshop**. Otherwise, this already is your first or working **Blazor** application that you can even use and interact with, including **Counter** and **Weather**, although it doesn't yet perform any of the actions specific to **Blazor.Podcast.Al**.

Don't Close the Visual Studio Code for your Project of Blazor.Podcast.AI but if Visual Studio Code is Closed, then if using a Mac you need to go to Finder, search for Visual Studio Code and then select it to Open it again, or if using Windows you need to go to Start, search for Visual Studio Code and select it so it is Open again. Then from Welcome in Visual Studio Code select Blazor.Podcast.AI from Recent.

Don't **Close** the **Command Prompt** on **Windows** or **Terminal** on **Mac** but if it is **Closed** then you need to go to **Finder**, search for **Terminal** and then select it to **Open** it, or if you **Closed** the **Command Prompt** on **Windows** you need to go to **Start**, search for **Command Prompt** and then select it to **Open** it. Then once opened you need to change directory using **cd** to the location for your **Project**, for example *cd Blazor.Podcast.Al* and then you need to type **dotnet watch** followed by **Enter**.

Don't **Close** the **Browser** but if it is **Closed** within **Terminal** on **Mac** or **Command Prompt** on **Windows** press the **Ctrl** key along with **C** on your **Keyboard** or on a **Mac** press **Command** along with **C** and then type **dotnet** watch followed by **Enter** which should relaunch the **Browser**.

This completes the process of launching the Project of Blazor.Podcast.Al in a Browser.







Implement

Provider Class

In Visual Studio Code select Program.cs in Explorer then choose New File... next to Blazor.Podcast.Al.



Then *Type* in the following **Name** and press **Enter** after which you should see or select a blank **Provider.cs** in **Explorer** within **Visual Studio Code**.

| Provider.cs |
|-------------|
|-------------|







🗢 tutorialr.com

Then within Visual Studio Code in Provider.cs you need to Copy and Paste the following Code:

using Microsoft.Extensions.AI; namespace Blazor.Podcast.AI; public class Provider(IChatClient chat) { // System Prompt // Cancellation // Properties // Send Method // Cancel & New Methods }

Information – This forms the outline of a class for Provider, a Class is used to group together Code or can be used to represent an Object. The first line is a using for functionality needed from the Package of Microsoft.Extensions.Al.OpenAl. Then there is a namespace for Blazor.Podcast.Al which is followed by the class which is defined for Provider including IChatClient which was Registered in Program.cs which is provided to the class with Dependency Injection in a Primary Constructor which is used to provide anything needed for a class in a concise readable manner. There are also Comments which are the lines starting with // which will help you place Code from the next few Steps of the Workshop.

Next within **Visual Studio Code** in **Provider.cs** underneath **// System Prompt** you need to *Copy* and *Paste* the following **Code**:

private const string system = @" You are a friendly and useful assistant that will help with podcast planning that is based on answers to questions, always state detailed opinions on anything asked of you then suggest title and short description for the podcast, segments for each episode, first five episode ideas, ideas to make it unique and generate a script for a trailer. Only use simple html and no markdown to format responses";

Information – This **Code** defines a **Constant** or **const** which is something that does not change when your application is running and is **private** as it is only used within the **class** of **Provider**. It is a **string** which is a **Types** which defines the kind of data I can have which in this case is text for a **System Prompt** within a pair of **Double Quotes** and ending with a **Semi-Colon**. A **System Prompt** is a set of instructions given to an **AI Model** that acts as a set of rules to guide how it behaves and responds along with any capabilities. In this case it will be a friendly and useful assistant that will help with **Podcast** planning including suggesting titles, description, segments, episode ideas, how to make it unique plus a generate a script for a trailer. The **System Prompt** can also be used to control how the **Response** will be formatted which normally uses **Markdown**, a special language for formatting text, but instead we want to use simple **HTML** which is used to format output for a **Browser**.







Then within **Visual Studio Code** in **Provider.cs** underneath **// Cancellation** you need to *Copy* and *Paste* the following **Code**:

private CancellationTokenSource? cancel;

Information – This **Code** defines a **Variable** which is something that does change when your application is running and is **private** as it is only used within the **class** of **Provider**, these kinds of **Variable** within a **class** are also known as **Members**. This **Variable** is a **Type** of **CancellationTokenSource** which will be used to provide a **Cancellation Token** to **Cancel** any **Response** and it is followed by a **Question Mark** or **?** meaning that it is **Nullable** which means it can have no **Value** before it is given one with **new()** later.

Next within **Visual Studio Code** in **Provider.cs** underneath **// Properties** you need to *Copy* and *Paste* the following **Code**:

Information – The first Property, which are normally public meaning they can be used inside and outside the class of Provider, is a string for a Title only has a get as it just returns the Value, the second Property is for a string for a Label which also only has a get, then this is followed by set of Questions which also just returns a Value and the Type of this Property is a Dictionary which is used to store a Value which in this case it is a List of text or string using a Key. The Keys are items such as "Podcast is about" and the Values for the List are set using Square Brackets to a set with one string such as string.Empty representing a Value that is blank or a multiple string values such as "Solo" which will define an Answer to a Question. The second Property represents a List of ChatMessage from Package of Microsoft.Extensions.Al.OpenAl which defines Requests or Responses set to a ChatMessage using Square Brackets to the System Prompt using ChatRole.System. The final two Properties with a Type known as a Boolean or bool which can be true or false will control when Questions or Generating should be displayed.







While still within **Provider.cs** in **Visual Studio Code** underneath **// Send Method** you need to *Copy* and *Paste* the following **Code**:

```
public async Task Send(string message)
{
    cancel = new();
    IsGenerating = true;
    IsQuestions = false;
    Messages.Add(new ChatMessage(ChatRole.User, message));
    var response = await chat.GetResponseAsync([.. Messages], null, cancel.Token);
    var assistant = new TextContent(response.Text);
    Messages.Add(new ChatMessage(ChatRole.Assistant, [assistant]));
    IsGenerating = false;
}
```

Information – Send is a Method, which is a way to create a reusable block of Code, they can be provided with Values known as Parameters which will be a string for message, and it is public which means it can be used outside the class of Provider. This Method also Returns a Task, used for Code that runs in the background, within the Method the Member of cancel is Initialised with new() and below this the two Properties of IsGenerating and IsQuestions are set as needed. The message that was provided to the Method is added to the Property of the List of ChatMessage as ChatRole.User which represents any Requests. Then a Variable of response is set using await which is Code that will run in the background to the result of the Method of GetResponseAsync for getting the Response using IChatClient that was provided to the class with Parameters for the List of ChatMessage, an unused Parameter provided with null and then a Token from the Cancellation Token. Then a Variable with the Type of TextContent of assistant is created using the Value of response which is added to the List of ChatMessage as ChatRole.Assistant which represents Responses, and IsGenerating set to false to indicate Generating has completed.







Next within **Provider.cs** in **Visual Studio Code** underneath **// Cancel & New Methods** you need to *Copy* and *Paste* the following **Code**:

```
public void Cancel()
{
    cancel?.Cancel();
    IsGenerating = false;
}
public void New()
{
    Cancel();
    IsQuestions = true;
    Messages = [new(ChatRole.System, system)];
}
```

Information – Theses **Methods** don't **Return** any values, so they are **void** and are **public** which means they can be used outside the **class** of **Provider**. The first **Method** of **Cancel** is used to stop any current **Response** from the **AI Model** and the **Method** of **New** will start a new conversation with the **AI Model** which includes resetting the **Property** for the **List** of **ChatMessage** to the **System Prompt**.

Next, within Visual Studio Code from the Menu select File and then Save as follows:



Information – If the **Terminal** on **Mac** or **Command Prompt** on **Windows** displays any **Errors**, then make sure that everything was entered correctly into **Provider.cs** by going over previous **Steps** to double check it matches what you have but once any corrections have been made and **Saved** or there are no **Errors** then the **Build** should proceed.







< tutorialr.com

Then within Visual Studio Code select Program.cs from Explorer as follows:

| X . | File <u>E</u> dit <u>S</u> election <u>V</u> iew <u>Go</u> <u>R</u> un <u>T</u> en | minal <u>H</u> elp | \leftrightarrow \rightarrow | D Blazor.Podcast.Al | 8 ~ | |
|------------|--|--|---|---|------------------|--|
| (J) | EXPLORER ···· | C Program.cs X C Provider.cs | | | | |
| - | V BLAZOR.PODCAST.AI | C Program.cs | | | | |
| ρ | | 1 using Microsoft.AspNet0 | ore.Components.Web; | | | La Carlo de Carlos de Carl |
| | > Layout | 2 using Microsoft.AspNetC | ore.Components.WebAssembly.He | | | VETTAMENTON AND AND AND AND AND AND AND AND AND AN |
| | > obj | 3 using Blazor.Podcast.Al | | | | |
| ٦ | > Pages | 4 using OpenAl; | | | | |
| | > Properties | 5 Using System.ClientMode | ans AT | | | |
| a'> | > www.root | 7 | | | | |
| | ♀ _Imports.razor | 8 var builder = WebAssemb | lyHostBuilder.CreateDefault(| args); | | |
| B | App.razor | 9 builder.RootComponents. | Add <app>("#app");</app> | | | |
| | Blazor.Podcast.Al.csproj | 10 builder.RootComponents. | Add <headoutlet>("head::after'</headoutlet> | | | |
| | Program.cs | | | | | |
| | Provider.cs | 12 Dullder.Services.AddSco | ped(sp -> new HttpClient { Ba | aseAddress - new Uri(Duilder.HostEnvironment. | BaseAddress) }); | |
| | | 14 var credential = new Ar | iKevCredential("github nat | | | |
| | | 15 var options = new OpenA | IClientOptions() | | | |
| | | | | | | |
| | | 17 Endpoint = new Uri(| | | | |
| | | | | | | |
| | | 19 Var client - new OpenAl | client(credential, options); | | | |
| | | 20 Var chat = cilent.Gettr 21 builder Services AddSir | <pre>alclient(gpt-40-mini).Asic aleton(chat):</pre> | accitenc(); | | |
| | | 22 // Register Provider | Brecon(chac); | | | |
| | | | | | | |
| | | | | | | |
| | | 25 await builder.Build().F | unAsync(); | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| Q | | | | | | |
| 0 | | | | | | |
| | > OUTLINE | | | | | |
| | > TIMELINE | | | | | |
| × (| 8040 | | | | | Ln 23. Col 1 Spaces: 4 UTF-8 CRLF () C# 83 🕰 |

In **Program.cs** below **Comment** of **// Register Provider** need to *Copy* and *Paste* the following **Code**:

builder.Services.AddSingleton<Provider>();

Finally, within Visual Studio Code from the Menu select File and then Save as follows:

| ×1 | ile Edit Selection View Go Run Te | minal Help | \leftarrow \rightarrow | , | 8 ~ | |
|-----|-----------------------------------|--------------------------------------|--|--|-------------------|--|
| Ð | New Text File Ctrl+N | Program.cs • C* Pr | | | | |
| | New File Ctrl+Alt+Windows+N | C Program.cs | | | | |
| 2 | New Window with Profile | 1 using Microso 2 using Microso | <pre>t.AspNetCore.Components.Web; ft.AspNetCore.Components.WebAssembly.</pre> | Hosting; | | |
| 99 | | 3 using Blazor. | | | | |
| 5 | Open File Ctrl+O | 5 using System. | | | | |
| a l | Open Workspace from File | | | | | |
| | Open Recent > | 8 var builder = | WebAssemblyHostBuilder.CreateDefault | t(args); | | |
| -⊞ | Add Folder to Workenses | 9 builder.RootO 10 builder.RootO | <pre>omponents.Add<app>("#app"); omponents.Add<headoutlet>("head::aft</headoutlet></app></pre> | er"): | | |
| | Save Workspace As | | | | | |
| | Duplicate Workspace | 12 builder.Servi | <pre>ces.AddScoped(sp => new HttpClient {</pre> | BaseAddress = new Uri(builder.HostEnvironmen | (BaseAddress) }); | |
| | Save Ctrl+S | 14 var credentia | <pre>1 = new ApiKeyCredential("github_pat_ new OpenAIClientOptions())</pre> | | | |
| | Save As Ctrl+Shift+S | 16 { | new openatorrencopcions() | | | |
| | Save All Ctrl+K S | 17 Endpoint | = new Uri(" <u>https://models.inference.a</u> | ai.azure.com") | | |
| | Share > | 19 var client = | new OpenAIClient(credential, options) |); [[hetClimet()] | | |
| | Auto Saua | 20 var chat = Ci 21 builder.Servi | ces.AddSingleton(chat); | chatchent(); | | |
| | Preferences | 22 // Register P 23 builder.Servi | rovider ces.AddSingleton <provider>():</provider> | | | |
| | Devest File | | | | | |
| | Close Editor Ctrl+F4 | 25 await builder 26 | .Build().RunAsync(); | | | |
| | Close Folder Ctrl+K F | | | | | |
| | Close Window Alt+F4 | | | | | |
| | Exit | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| 8 | | | | | | |
| 533 | > OUTLINE | | | | | |
| 225 | > TIMELINE | | | | | |
| × (| 90A0 | | | | | Ln 23, Col 43 Spaces: 4 UTF-8 CRLF () C# 🚳 🗘 |

This completes the process in Visual Studio Code of creating and Registering the Provider.





Item Component

In Visual Studio Code select Program.cs in Explorer then choose New File... next to Blazor.Podcast.AI.



Then *Type* in the following **Name** and press **Enter** after which you should see or select a blank **Item.razor** in **Explorer** within **Visual Studio Code**.

Item.razor









Then within Visual Studio Code in Item.razor you need to Copy and Paste the following Razor:

```
@using Microsoft.Extensions.AI

@* Messages *@

@code {
    [Parameter]
    public required ChatMessage Message { get; set; }
}
```

Information – This forms the outline of a **Component** in **Razor** which combines **HTML** used to output in a **Browser** with **C#** in **Blazor**. The first line of **using** is for functionality needed in the **Component** from the **Package** of **Microsoft.Extensions.AI.OpenAI**. There is also a **Comment** which in **Razor** starts with **@*** and ends with ***@** and **code** with the **Property** of **ChatMessage** which will be a **Parameter** of the **Component**.

Then within **Visual Studio Code** in **Item.razor** underneath **@* Messages *@** you need to *Copy* and *Paste* the following **Razor**:

```
@if (Message.Role == ChatRole.User)
{
    <div class="card text-white bg-info m-2">
        <div class="card-header">User</div>
        <div class="card-body">
             <mark>@</mark>Message.Text
        </div>
    </div>
}
else if(Message.Role == ChatRole.Assistant)
{
    foreach (var content in Message.Contents)
    {
        if (content is TextContent { Text: { Length: > 0 } text })
             <div class="card bg-light m-2">
                 <div class="card-header">Assistant</div>
                 <div class="card-body">
                     @((MarkupString)text)
                 </div>
             </div>
        }
    }
}
```

Information - This will be used to output a ChatMessage for ChatRole.User or ChatRole.Assistant
using an if statement to check it is Equal with == to the Role from ChatMessage. Should the Role be
ChatRole.User then the first block of Razor will be output for a Request otherwise with else if the Role
is ChatRole.Assistant then the second block of Razor will be output for a Response and only if this is
TextContent with a Length greater than 0 using > and this will be in HTML so MarkupString will convert
it so that it can be output correctly in the Browser.







Next, within Visual Studio Code from the Menu select File and then Save as follows:

| X | ile Edit Selection View Go Run Ten | minal Help | | $\leftarrow \rightarrow$ | | 8 ~ | | o × |
|----------|------------------------------------|------------------------------|--|--------------------------|---|------------|--|-----------|
| ch. | New Text File Ctrl+N | Program.cs | Item.razor | | | | | |
| 6 | New File Ctrl+Alt+Windows+N | | | | | | | |
| ρ | New Window Ctrl+Shift+N | | | | | | | gur. |
| <u>í</u> | New Window with Profile > | 2 3 A* Mor | 53805 *A | | | | | |
| ુર | Open File Ctrl+O | 4 @if (N | essage.Role == C | hatRole.User) | | | | |
| | Open Folder Ctrl+K Ctrl+O | | | | | | | |
| æ | Open Workspace from File | | iv class="card to (div class="card | ext-white bg-in | o m-2"> | | | |
| | Open Recent > | | <div class="ca</td><td>rd-body"></div> | | | | | |
| - H8- | | | @Message.T | ext | | | | |
| | Add Folder to Workspace | | | | | | | |
| | Save Workspace As | | | | | | | |
| | Duplicate Workspace | | f(Message.Role = | - ChatRole.Assi | tant) | | | |
| | Save Ctrl+S | 14 1 15 f o | reach (var conte | nt in Message.Co | ontents) | | | |
| | Save As Ctrl+Shift+S | 16 { | | | | | | _ |
| | Save All Ctrl+K S | | if (content is | TextContent (| <pre>fext: { Length: > 0 } text })</pre> | | | |
| - | | | | -"card bg-light | | | | |
| | Share | | | lass="card-head | r">Assistant | | | |
| | Auto Save | | ματό C. (| (MarkupString)t | ext) | | | |
| | Preferences > | | | | | | | |
| | Revert File | | | | | | | |
| | Close Editor Ctrl+E4 | | | | | | | |
| | Close Folder Ctrl+K F | | | | | | | |
| | Close Window Alt+F4 | 28 29 @code | | | | | | |
| - | | | | | | | | |
| | Exit | | blic required Ch | atMessage Messa | | | | |
| | | | | | | | | |
| | | | | | | | | |
| | | | | | | | | |
| | | | | | | | | |
| | | | | | | | | |
| | | | | | | | | |
| | | | | | | | | |
| | | | | | | | | |
| _ | | | | | | | | |
| 8 | | | | | | | | |
| ~~ | | | | | | | | |
| દુરુટ | > TIMELINE | | | | | | | |
| × (| 30∆0 | | | | | | Ln 28, Col 1 Spaces: 4 UTF-8 CRLF () R | Razor 🐯 🚨 |

You should double check that everything was entered correctly as if the **Terminal** on **Mac** or **Command Prompt** on **Windows** displays any **Errors** this will be only for any **Code** that was entered into **Item.razor**, so go over the previous **Steps** to double check it matches what you have but once any corrections have been made and **Saved** or there are no **Errors** then the **Build** should proceed.

This completes the **Component** of **Item** to be used to output each **Message** for a **Request** or **Response**.







Send Component

In Visual Studio Code select Program.cs in Explorer then choose New File... next to Blazor.Podcast.Al.



Then *Type* in the following **Name** and press **Enter** after which you should see or select a blank **Send.razor** in **Explorer** within **Visual Studio Code**.

Send.razor









Then within Visual Studio Code in Send.razor you need to Copy and Paste the following Razor:

```
@* Edit Form *@
@code {
    // Members
    // Parameters
    // Submit Method
    // OnInitialized Method
}
```

Information – This forms the outline of a **Component** in **Razor** which combines **HTML** used to output in a **Browser** with **C#** in **Blazor**. There is a **Comment** which in **Razor** starts with **@*** and ends with ***@** along with a **code** section with **Comments** starting with **//** which will help you place **Code** for the next few **Steps**.

While within **Visual Studio Code** in **Send.razor** below the **Comment** of **// Members** you need to *Copy* and *Paste* the following **Code**:

```
private Dictionary<string, string> answers = new();
private string? message;
```

Information – These **Members** will only be used within the **Component** so they are **private**, the first **Member** is a **Dictionary** where the **Key** will match the ones from any **Questions** but there will be only a single **Answer**, so the **Value** is just a single **string**. The second **Member** is a **string**? that can be nothing.

Next within **Visual Studio Code** in **Send.razor** below the **Comment** of **// Parameters** you need to *Copy* and *Paste* the following **Code**:

```
[Parameter]
public string? TextAreaLabel { get; set; }
[Parameter]
public bool IsQuestions { get; set; }
[Parameter]
public Dictionary<string, List<string>> Questions { get; set; } = new();
[Parameter]
public EventCallback<string> OnSend { get; set; }
```

Information – These **Parameters** will be provided to the **Component** and include a **string**? to be used for the **Label** of a **Text Area** and a **bool** to control whether the Questions will be shown or not. There is also a **Parameter** for **Questions** which is the same **Type** as that defined in the **Provider** and there is a **Parameter** for an **EventCallback** which will notify another **Component** when a **Message** needs to be **Sent** from this **Component**.







Then within **Visual Studio Code** in **Send.razor** below the **Comment** of **// Submit Method** you need to *Copy* and *Paste* the following **Code**:

```
private async Task Submit()
{
    if (IsQuestions)
    Ł
        if (message?.Length > 0)
            answers.Add(string.Empty, message);
        if (answers.Values.All(a => !string.IsNullOrWhiteSpace(a)))
            message = string.Join(",",
            answers.Select(value => $"{value.Key} {value.Value}"));
        }
    if (message is { Length: > 0 } text)
    Ł
        message = null;
        await OnSend.InvokeAsync(text);
    }
}
```

Information – The Method of Submit is used to Send a Message, this could be the Questions, which will be indicated if the value of IsQuestions is true. Should this be the case then if there was a message also Sent with the Questions this will be added to answers with a Key that is blank. This is followed by another if that check to make sure that all Questions have been Answered using All which is a special Method using LINQ and it checks to make sure All values are !IsNullOrWhiteSpace which means not null, blank or any other whitespace, if they are they are combined into message using another Method using LINQ of Select into one long Message separated by a Comma using the Method of Join for a string. Finally, there is a check to see if the message has any contents using the Length which produces a Value of text and then message is reset and OnSend is triggered with the Value of text.

Next within **Visual Studio Code** in **Send.razor** below the **Comment** of **// OnInitialized Method** you need to *Copy* and *Paste* the following **Code**:

```
protected override void OnInitialized() =>
    answers = Questions.ToDictionary(kvp => kvp.Key, kvp => string.Empty);
```

Information – This is a special **Method** as it replaces the **Method** normally used to set up a **Component** of **OnInitialized** to perform any custom setup, in this case it is setting up **answers** by converting **Questions** into the correct kind of **Dictionary** that it needs with the **Method** of **ToDictionary** with the same **Keys** but a blank value to become an **Answer** to a **Question** that will be input or selected in **Blazor Podcast AI**.







Then within **Visual Studio Code** in **Item.razor** underneath **@* Edit Form *@** you need to *Copy* and *Paste* the following **Razor**:

```
<EditForm Model="@this" OnValidSubmit="@Submit" class="m-2">
   @if(IsQuestions)
       @foreach (var question in Questions.Keys)
           <div class="form-group">
               <label class="form-label">@question</label>
               @if (Questions[question].Count > 1)
                   <select class="form-control mb-2" @bind="@answers[question]">
                      <option value="">Select Option</option>
                      @foreach (var answer in Questions[question])
                          <option value="@answer">@answer</option>
                  </select>
               }
               else
               {
                   <textarea class="form-control mb-2" rows="1"
                  @bind="@answers[question]" />
           </div>
       }
   }
   <div class="form-group">
       <label class="form-label">@TextAreaLabel</label>
       </div>
   <button type="submit" class="btn btn-primary mb-2">
       Send
   </button>
   <button type="reset" class="btn btn-secondary mb-2">
       Reset
   </button>
</EditForm>
```

Information – EditForm is used for input, if the Value of IsQuestions is true it will Loop through the Questions with the foreach and if the Answers for a Question is more 1 than using > then it will use the Select from HTML to produce a Dropdown with the Answers to choose from, getting each Answer from the Dictionary by Key using the Square Brackets, if there is just one Answer then it will use a TextArea from HTML instead where you can input Text. This is then followed by another TextArea which has a Label from HTML using the Parameter of TextAreaLabel and finally there is a button of Submit which will trigger the Method of Submit set in the EditForm and a button to Reset the EditForm.







Next, within Visual Studio Code from the Menu select File and then Save as follows:



Once **Saved** then return to the **Terminal** on **Mac** or **Command Prompt** on **Windows** and you will see an **Error** that **Adding an abstract method or overriding an inherited method requires restarting the application**, and it will ask **Do you want to restart your app? Yes (y) / No (n) / Always (a) / Never (v)** if you *Type* in **y** this will restart the application and reload your **Browser**.



Information – If you look out for **Hot reload succeeded** in **Terminal** on **Mac** or **Command Prompt** on **Windows** this will indicate that there are no more **Errors**.







However, if the **Terminal** on **Mac** or **Command Prompt** on **Windows** still displays any **Errors** other than that one and **Exited with error code -1** after restarting then you should double check everything was entered correctly as any **Errors** will only for any **Code** that was entered into **Send.razor**, so go over the previous **Steps** to double check it matches what you have but once any corrections have been made and **Saved** or there are no **Errors** then the **Build** should proceed as follows:

| Command Prompt - dotnet v × + ~ | | × |
|--|-----|----|
| dotnet watch 🖕 [Blazor.Podcast.AI (net9.0)] Hot reload succeeded. | | |
| dotnet watch 🕗 File updated: .\Program.cs | | |
| dotnet watch 💧 [Blazor.Podcast.AI (net9.0)] Hot reload succeeded. | | |
| dotnet watch 🕗 File added: .\Item.razor | | |
| dotnet watch 🧅 [Blazor.Podcast.AI (net9.0)] Hot reload succeeded. | | |
| dotnet watch 🖉 File updated: .\Item.razor | | |
| dotnet watch 🖕 [Blazor.Podcast.AI (net9.0)] Hot reload succeeded. | | |
| dotnet watch 🖉 File added: .\Send.razor | | |
| dotnet watch O [Blazor.Podcast.AI (net9.0)] Hot reload succeeded. | | |
| dotnet watch 🕐 File updated: .\Send.razor | | |
| dotnet watch of Unable to apply hot reload, restart is needed to apply the changes. | | |
| dotnet watch X C: Workshop/Blazor.Podcast.Al/Send.razor(80,5): error ENC0023: Adding an abstract method or overriding an | inh | er |
| 1 ted method requires restarting the application. | | |
| bo you want to restart your app? Yes (y) / No (n) / Atways (a) / Never (V) | | |
| f y f | | |
| dotnet watch A [blazbr.poldast.in (hets.o)] Exited with effort ode -1 | | |
| dotnet watch & Build Succeeded: C: Workshop Blazon Dodest AT Blazon Dodest AT Staroni | | |
| Using Jaugh strings from C. Workshon Blazon Dodast AT Dronovies Jaugh Settings ison | | |
| info: Microsoft Hosting ifetime[14] | | |
| Now listening on http://localhost:5136 | | |
| info: Microsoft.Hosting.Lifetime[0] | | |
| Application started. Press Ctrl+C to shut down. | | |
| info: Microsoft.Hosting.Lifetime[0] | | |
| Hosting environment: Development | | |
| info: Microsoft.Hosting.Lifetime[0] | | |
| Content root path: C:\Workshop\Blazor.Podcast.AI | | |
| | | |
| | | |

This completes the **Component** that will be used to **Send** a **Request** including the initial set of **Questions**.







Home Component

In Visual Studio Code in Explorer select Pages and then select Home.razor as follows:



Information – This defines the **Content** for the **Page** that is displayed in your **Browser** using **Razor**.

Then you need to remove everything from **Home.razor** so it appears as follows:









Once within Home.razor in Visual Studio Code you need to Copy and Paste the following Razor:

@page "/"
@inject Provider provider
<PageTitle>@provider.Title</PageTitle>
<h1>@provider.Title</h1>
@* New & Items *@
@* Output *@

Information – This forms the outline of a Page in Razor which combines HTML used to output in Browser with C# in Blazor which is indicated with the Directive of page at the top which is then followed by the Provider which is provided by Dependency Injection with the Directive of inject. There is a new PageTitle and H1 which uses Title from the Provider, followed by Comments which in Razor start with @* and end with *@ which will help you place Code from the next few Steps.

Then within **Visual Studio Code** in **Home.razor** underneath **@* New & Items *@** you need to *Copy* and *Paste* the following **Razor**:

Information – There is a **Button** in **HTML** for **New** which when **Clicked** will trigger the **Method** of **New** in the **Provider** using **onclick**. This is followed by a **foreach** which will **Loop** through each **Message** in the **Property** of **Messages** in the **Provider** and then use the **Component** of **Item** to output each **Message**.







Then within **Visual Studio Code** in **Home.razor** underneath **@* Output *@** you need to *Copy* and *Paste* the following **Razor**:

```
@if (provider.IsGenerating)
{
    <div class="card bg-light m-2">
        <div class="card-header">Generating</div>
        <div class="card-body">
            <div class="spinner-border text-black" role="status">
                <span class="sr-only"></span>
            </div>
        </div>
        <div class="card-footer">
            <button class="btn btn-secondary m-2" <br/>
@onclick="@provider.Cancel">
                 Cancel
            </button>
        </div>
    </div>
}
else
{
    <Send IsQuestions="@provider.IsQuestions"</pre>
    Questions="@provider.Questions" OnSend="@provider.Send"
    TextAreaLabel="@provider.Label"/>
}
```

Information – When IsGenerating is true then it will output the block of **Razor** that also includes a **button** to **Cancel** which will trigger the **Method** of **Cancel** in the **Provider** using **onclick**. However, when **IsGenerating** is **false** it will output the **Component** of **Send** and will provide the **Parameters** from the **Provider** including **IsQuestions** and **Questions** along with when **OnSend** is triggered that this is handled by the **Method** of **Send** in the **Provider** and set the **TextAreaLabel** to **Label**.







Finally, within Visual Studio Code from the Menu select File and then Save as follows:

| X I | ile Edit Selection View Go Run Term | nal Help ← → | |
|------------|-------------------------------------|--|---|
| Ð | New Text File Ctrl+N | Program.cs O Item.razor O Send.razor I Home.razor I Provider.cs | II ··· |
| | New File Ctrl+Alt+Windows+N | Yages > ↔ Homerazor | |
| ρ | New Window Ctrl+Shift+N | 1 @page "/" | El Son any research and a second Second and a second a second a second a se |
| | New Window with Profile > | 2 ginject Provider provider | Patients Patients "D@@fames |
| ço | Open File Ctrl+O | 4 <pagetitle>@provider.Title</pagetitle> | The second se |
| | Open Folder Ctrl+K Ctrl+O | 5 6 - Ahtomoutidan Titla/Ahto | |
| æ> | Open Workspace from File | 7 7 | |
| | Open Recent > | | |
| HE - | | 9 <button @onclick="@provider.New" class="btn btn-outline-primary m-2"></button> | |
| | Add Folder to Workspace | | |
| | Save Workspace As | | |
| | Duplicate Workspace | 13 @foreach (var message in @provider.Messages) | |
| | Save Ctrl+S | 14 { 15 <[them:@kevs="@message" Message" /> | |
| | Save As Ctrl+Shift+S | | |
| | Save All Ctrl+K S | 17 17 - 01 State 10 | |
| | | 18 @ fut (provider_1sGenerating) | |
| | Share > | | |
| | Auto Save | 21 cdiv class="rand hs_light m_2"> | |
| | Preferences > | 23 <iiv class="card-header">Generating</iiv> | - |
| | | | |
| | Kevert Hie | 25 <div class-"spinner-border="" role-"status"="" text-black"=""> 25</div> | |
| | Close Editor Ctri+F4 | 20 Span Cassa Should Arspan | |
| | Close Folder Ctrl+K F | | |
| | Close Window Alt+F4 | 29 (div class="card-footer"> | |
| | Exit | 30 | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | 38 <send isquestions-"@provider.isquestions"<="" td=""><td></td></send> | |
| | | 39 Questions= @provider.guestions onsena @provider.sena 48 TextApealabel="@provider.label="/> | |
| | | | |
| | | | |
| | | | |
| 8 | | | |
| | | | |
| £63 | > TIMELINE | | |
| × (| 0 & 0 | | Ln 42, Col 1 Spaces: 4 UTF-8 with BOM CRLF () Razor 🚳 🚨 |

You should double check that everything was entered correctly as if the **Terminal** on **Mac** or **Command Prompt** on **Windows** displays any **Errors** this will be only for any **Code** that was entered into **Home.razor**, so go over the previous **Steps** to double check it matches what you have but once any corrections have been made and **Saved** or there are no **Errors** then the **Build** should proceed.

Don't **Close** the **Visual Studio Code** for your **Project** of **Blazor.Podcast.AI** but if **Visual Studio Code** is **Closed**, then if using a **Mac** you need to go to **Finder**, search for **Visual Studio Code** and then select it to **Open** it again, or if using **Windows** you need to go to **Start**, search for **Visual Studio Code** and select it so it is **Open** again. Then from **Welcome** in **Visual Studio Code** select **Blazor.Podcast.AI** from **Recent**.

Don't **Close** the **Command Prompt** on **Windows** or **Terminal** on **Mac** but if it is **Closed** then you need to go to **Finder**, search for **Terminal** and then select it to **Open** it, or if you **Closed** the **Command Prompt** on **Windows** you need to go to **Start**, search for **Command Prompt** and then select it to **Open** it. Then once opened you need to change directory using **cd** to the location for your **Project**, for example *cd Blazor.Podcast.Al* and then you need to type **dotnet watch** followed by **Enter**.

Don't **Close** the **Browser** but if it is **Closed** within **Terminal** on **Mac** or **Command Prompt** on **Windows** press the **Ctrl** key along with **C** on your **Keyboard** or on a **Mac** press **Command** along with **C** and then type **dotnet** watch again followed by **Enter** which should relaunch the **Browser**.

This completes the **Page** of **Home** that will be used to output or input any **Messages** including **Questions** and this also completes creating a **Podcast Assistant** in **Blazor** using **GitHub Models** for the **Workshop**.







Generate

Podcast Assistant

Switch over to the **Browser** that would have been opened by the **Terminal** on **Mac** or **Command Prompt** on **Windows** as follows:

| 🔞 🗊 🥑 Blazor Podcast Al | x + | | | - | ٥ | × |
|-----------------------------------|---|---|---|-----|-----|----------|
| \leftarrow C (i) localhost:5046 | | ☆ | G | ি 🔹 | | 0 |
| Blazor.Podcast.Al | | | | | Abo | out |
| i Home | Blazor Podcast Al | | | | | |
| + Counter | New | | | | | |
| 😑 Weather | Podcast is about | | | | | |
| | | | | | | 1 |
| | Host of podcast is | | | | | |
| | | | | | , | le la |
| | Listener of podcast is | | | | | |
| | | | | | | le le |
| | Format of podcast is | | | | | |
| | Select Option | | | | | |
| | Purpose of podcast is | | | | | |
| | Select Option | | | | | |
| | Optionally refine with details or questions | | | | | |
| | | | | | | |
| | Send Reset | | | | | 6 |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |
| | | | | | | |

Information – If your **Home** in the **Browser** does not look like this then **Refresh**. If it still does not look like this then go through the previous **Steps** in the **Workshop** to see if you missed anything or got anything wrong, pay particular attention to any **Razor** as this may not have triggered an **Error** but could be incorrect or in the wrong place.

You can start a new **Conversation** with **New** or **Cancel** any **Response** being **Generated**. **GitHub Models** does limit any **Requests** that are **Sent** and **Responses** that are **Generated** each day as it is free to use. When you do hit the limit then **Blazor Podcast AI** will stop working but it will start working again after twenty-four hours as the limit will be reset for **GitHub Models**.

Information – **GitHub Models** can be replaced by paid-for **AI Models** that can be used as often as you want, although they cost money depending on how many **Requests** are made or how large they are along with how many **Responses** are **Generated** and how large those are. These costs vary depending on the **Model** used but generally they can be quite small, but this can add up but when experimenting then using them for free is ideal such as with **GitHub Models**.







Once in the **Browser** for **Blazor Podcast AI** start by **Answering** the **Questions**, the first **Podcast is about** can be answered with any topic such as *Blazor*, *GitHub Models* or anything you want, the second **Host of podcast is** can be answered with your name and the third **Listener of podcast is** can be the kind of listener such as beginner or details about the person you're aiming to reach. There are also **Questions** for **Format of podcast is**, and **Purpose of podcast is** where you can pick from a set of **Answers** in a **Dropdown** to pick the format or purpose that you think would make a good podcast. **Optionally** you can refine with more information, but you can leave it blank, but you must **Answer** the **Questions** for example as follows:

| 🕼 🗖 🍼 Blazor Podcast Al | x + | | | - | o × |
|-----------------------------------|---|---|---|------|-------|
| \leftarrow C (] localhost:5046 | | ☆ | G | £≞ 🎓 | 🥠 |
| Blazor.Podcast.Al | | | | | About |
| 🕺 Home | Blazor Podcast Al | | | | |
| + Counter | New | | | | |
| 😑 Weather | Podcast is about | | | | |
| | Blazor | | | | |
| | Host of podcast is | | | | |
| | Peter | | | | |
| | Listener of podcast is | | | | |
| | Beginners | | | | |
| | Format of podcast is | | | | |
| | Solo | | | | |
| | Purpose of podcast is | | | | |
| | Education | | | | |
| | Optionally refine with details or questions | | | | |
| | | | | | |
| | Send Reset | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |

Then in the **Browser** for **Blazor Podcast Al** select **Send** to start **Generating** as follows:

| 🔞 🗖 🥑 Blazor Podcast Al | x + | | | - | o × |
|-----------------------------------|---|----------|-----|----|-------|
| \leftarrow C (i) localhost:5046 | | ث | G (| \$ | 🥠 |
| Blazor.Podcast.Al | | | | | About |
| i Home | Blazor Podcast Al | | | | |
| + Counter | New | | | | |
| E Weather | User Podcast is about Blazor, Host of podcast is Peter, Listener of podcast is Beginners, Format of podcast is Solo, Purpose of podcast is Education | | | | |
| | Generating | | | | |
| | о О | | | | |
| | Cancel | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |







Once **Generating** has completed in the **Browser** for **Blazor Podcast AI** you will see a **Response** from the **Assistant** which for example will be as follows:

| 🔞 🔲 🚿 Blazor Podcast Al | x + - o | × |
|-----------------------------------|---|-----|
| \leftarrow C (i) localhost:5046 | 요 요 요 요 | - 🥠 |
| Blazor.Podcast.Al | Abou | ,t |
| i Home | Blazor Podcast Al | |
| + Counter | New | |
| 😑 Weather | User | |
| | Podcast is about Blazor, Host of podcast is Peter, Listener of podcast is Beginners, Format of podcast is Solo, Purpose of podcast is Education | |
| | Assistant | |
| | That's a fantastic idea for a podcast! Blazor is a powerful framework for building web applications with C# and .NET, and providing educational content, especially aimed at beginners, can create a great resource for those looking to learn. Let's break down several components to help with your podcast planning. | |
| | Podcast Title | . 1 |
| | "Blazor Basics with Peter" | . 1 |
| | Short Description | . 1 |
| | Join Peter in "Blazor Basics," where beginners embark on their journey into web application development using Blazor. Each episode dives into fundamental concepts, tips, and techniques to help you understand and master this versatile framework. | |
| | Proposed Segments for Each Episode | |
| | Introduction: Brief intro to the episode's topic. Main Content: Deep dive into the topic with evaluations, examples, and practical advice. | |
| | Content: Deep the the object will be provide the parallelistic, standproved and the content of the object of the parallelistic standproved and the content of the parallelistic standproved and the parallelistin standproved and the parallelistic standproved and the paralleli | |
| | Tips and incks: Quick ups related to the episode's content. Conclusion: Recap of the episode and a teaser for the next one. | |
| | First Five Episode Ideas | |
| | Episode 1: "Getting Started with Blazor: What Is It and Why Use It?" | |
| | Episode 2: "Setting Up Your Development Environment for Blazor." Episode 3: "Creating Your First Blazor Application: A Step-by-Step Guide." | |
| | Episode 4: "Components in Blazor: Building Blocks of Your Applications." Episode 5: "Working with Data: Integrating APIs in Blazor." | |

Response may be just want you want but generally when using **AI**, you need to amend or adjust until you get what you want by asking questions or providing information to refine a **Response**, you can do this by **Scrolling** down the **Page** and then select the **Text Area** for **Optionally refine with details or questions** and for example you may have changed your mind about the topic so *Type* in **Make it about .NET instead** or anything else you want to change as follows:

| 🔞 🔲 🥑 Blazor Podcast Al | × + | - | ٥ | × |
|-----------------------------------|--|-------|------|----|
| \leftarrow C (i) localhost:5046 | ☆) G ☆ | \$ | | ø |
| Blazor.Podcast.Al | | A | bout | • |
| if Home | Join Peter in "Blazor Basics," where beginners embark on their journey into web application development using Blazor. Each episode dives into fundamental concepts, tips, and techniques to help you understar master this versatile framework. | d and | | |
| + Counter | Proposed Segments for Each Episode Introduction: Brief intro to the episode's topic. Main Content: Deep dive into the topic with explanations, examples, and practical advice. Q&A Section: Answer questions submitted by listeners regarding Blazor and web development. Typs and Tricks: Quick tips related to the episode's content. Conclusion: Recap of the episode and a teaser for the next one. | | | |
| | First Five Episode Ideas | | | |
| | Episode 1: "Getting Started with Blazor: What Is It and Why Use It?" Episode 2: "Setting Up Your Development Environment for Blazor," Episode 3: "Creating Your First Blazor Application: A Step-by-Step Guide." Episode 4: "Components in Blazor: Building Blocks of Your Applications." Episode 5: "Working with Data: Integrating APIs in Blazor." | | | |
| | Ideas to Make It Unique | | | -I |
| | Add a "Listener Spotlight" segment where you feature a question or project from a listener. Include live coding sessions or demos (depending on the format) to showcase concepts in action. Create a community forum where listeners can discuss episodes and ask questions, fostering engagement. Host occasional guest experts from the Blazor community for special episodes. Compile resources, code snippets, and additional reading material for listeners to access after each episode. | | | |
| | Trailer Script | | | -I |
| | "Welcome to 'Blazor Basics with Peterl' Are you a beginner eager to dive into web development with Blazor? You've come to the right placel In each episode, I'll guide you through everything you need to know- the foundational concepts of Blazor to creating your very first application. Whether you're setting up your development environment or integrating APIs, I'll equip you with tips and tricks to navigate this exciting framework. Join me on this educational journey to unlock your potential as a web developert Subscribe now and let's get started!" | from | | |
| | Optionally refine with details or questions | | | ł |
| | Make it about .NET instead | | | |
| | Send Reset | | | ļ |







Then in the **Browser** for **Blazor Podcast Al** select **Send** to start **Generating** and once completed you should see your refined output for example the podcast will now be about **.NET** as follows:

| 🔞 🗊 🚿 Blazor Podcast Al | x + | - | ٥ | \times |
|-----------------------------------|---|------|-------|----------|
| \leftarrow C (i) localhost:5046 | ☆) G ☆ | \$ | | Ø |
| Blazor.Podcast.Al | | A | About | • |
| A Home | User | | | |
| + Counter | Make it about NET instead | | | |
| Counter Weather | Assistant Great choicel Expanding the content focus to .NET will allow for a broader range of topics, appealing to beginners interested in various aspects of the .NET framework, not just Blazor. This can encompass ASPJ. NET Core, desktop applications, and more. Here is an updated plan for your podcast. Podcast Title "NET Fundamentals with Peter" Short Description Welcome to "NET Fundamentals" with Peter, your go-to podcast for all things. NET. Designed specifically for beginners, each episode breaks down essential concepts, tools, and techniques to help you navigate world of .NET development and kickstart your programming journey. Proposed Segments for Each Episode • Introduction: Biel overview of what will be discussed in the episode. • Main Content: In-depth discussion of the chosen topic with examples and insights. • QAS Section: Answer listener-submitted questions about NET development. • Tips and Tricks: Practical advice and resources relevant to the episode topic. • Conclusion: Summary of what was covered and sneak peek into the next episode. • Episode 1: "Introduction to. NET: An Overview of the Framework." • Episode 1: "Introduction to .NET is NET Conside Applications "Resp-Step Tourial." • Episode 1: "Introduction to .NET is NET Conside Applications the Applications with ASP NET. A Beginner's Guide." • Episode 1: "Introduction to .NET is NET Conside Applications Studie." • Episode 2: "Setting Up Your. NET Development Environment. A Beginn | ↓ET, | | |
| | Feature a "Listener Project Showcase" where you highlight a project from a listener each month. Incorporate mini-challenges or coding tasks for listeners based on episode content. Create a dedicated social media group for listeners to discuss episodes, share knowledge, and network. | | | |

You could also expand on suggestions for example by *Typing* in **Can you expand on episode one?** in the **Text Area** for **Optionally refine with details or questions** and then select **Send** and once **Generating** has completed you can then review any **Response** for example as follows:



This completes using the Podcast Assistant in Blazor with GitHub Models for the Workshop.







Custom Assistant

Podcast Assistant allows you to get started with you own **Podcast**, but with some modifications you can create a **Custom Assistant** that can do anything else, so return to **Visual Studio Code** as follows:



Next from Explorer in Visual Studio Code select Provider.cs as follows:



Information – You can also select Provider.cs from the Tabs at the top of Visual Studio Code.







Then in **Visual Studio Code** within **Provider.cs** in the **Method** of **New** you need to change **IsQuestions** from being assigned to **true** to being assigned to **false** instead, so the **Method** of **New** is as follows:



Information – This change will make sure that the **Questions** are not shown when choosing **New** in the **Browser** as this is only needed for the **Podcast Assistant**.

Next in **Visual Studio Code** within **Provider.cs** you need to change the **Property** of **IsQuestions** from it being initialised from **true** to **false** instead, so the **Property** of **IsQuestions** is as follows:

public bool IsQuestions { get; set; } = false;

Information – This change will make sure that the **Questions** are not shown when starting the application in the **Browser** as this is only needed for the **Podcast Assistant**.

Then in **Visual Studio Code** within **Provider.cs** you need to change the **Property** of **Label** from it being initialised from **Optionally refine with details or questions** to the following:

Provide content for social media or questions

The **Property** of Label should now be as follows:

public string Label { get; } = "Optionally refine with details or questions";

Information – This can be anything that makes sense for your **Custom Assistant**, so people know what to do when using it for the first time or using it throughout to **Generate** what they need.

Next in **Visual Studio Code** within **Provider.cs** you need to change the **Property** of **Title** from it being initialised from **Blazor Podcast AI** to the following:

Social Media AI

The **Property** of **Title** should now be as follows:

public string Title { get; } = "Social Media AI";

Information – This can be changed to anything that makes sense for your Custom Assistant.







Then you need to update the **System Prompt**, this is the **const** for the **string** of **system** near the top of the **Provider** to change the behaviour from a **Podcast Assistant** to your own **Custom Assistant**, you should leave the **You are a friendly and useful assistant** part of the **System Prompt** along with **Only use simple html and no markdown to format responses** but otherwise you can change the **System Prompt** to behave like anything such as a **Social Media Assistant** as follows:

private const string system = @" You are a friendly and useful assistant that will help with social media to create engaging posts adapted for main social media platforms also include any suitable hashtags, offer advice on content scheduling, latest trends and any best practices to maximise visibility. Only use simple html and no markdown to format responses";

Information – You can define any behaviour using a **System Prompt** for your **Custom Assistant** in this case it is one that helps with social media to create engaging posts for the main social media platforms and include any suitable hashtags along with offering advice on scheduling content plus latest trends and best practices to maximise visibility. Your **Custom Assistant** could be a creative catalyst for suggesting ideas, provoke challenging conversations or it could be a specialised expert in a specific area to offer guidance. You can even have fun, what would happen if you added *talk like a pirate* to your **System Prompt**?

Finally, within Visual Studio Code from the Menu select File and then Save as follows:







tutorialr.com

Once **Saved** then return to the **Terminal** on **Mac** or **Command Prompt** on **Windows** and you will see an **Error** that **Adding an abstract method or overriding an inherited method requires restarting the application**, and it will ask **Do you want to restart your app? Yes (y) / No (n) / Always (a) / Never (v)** if you *Type* in **y** this will restart the application and reload your **Browser**.



If the **Terminal** on **Mac** or **Command Prompt** on **Windows** still displays any **Errors** other than that one and **Exited with error code -1** after restarting, then you should double check everything was updated correctly in **Provider.cs** from the previous **Steps** of the **Workshop**, or if there are no other **Errors** then the **Build** should proceed as follows:









In Visual Studio Code you have completed everything for your Custom Assistant so now you can switch over to the Browser that would have been opened by the Terminal on Mac or Command Prompt on Windows as follows:

| 🕼 🔲 🍼 Social Media Al | × + | | - | o) | × |
|-----------------------------------|---|-----------|-----|-------|---|
| \leftarrow C (] localhost:5046 | | \$ GIN | 1 🕈 | 🍕 | 0 |
| Blazor.Podcast.Al | | | | About | |
| A Home | Social Media Al | | | | |
| + Counter | New | | | | |
| = Weather | Provide content for social media or questions | | | | |
| | | | | 6 | |
| | Send Reset | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |

Information – It should display something like this in the **Browser** depending on what **Custom Assistant** you create and in the **Text Area** for **Provide content for social media or questions** you can *Type* in something such as **GitHub Models**, select **Send** and once **Generating** has completed will be like as follows:



This completes creating a **Custom Assistant** in **Blazor** with **GitHub Models** and concludes the **Workshop** so you can close **Visual Studio Code**, **Browser** and **Terminal** on **Mac** or **Command Prompt** on **Windows**.



CC