Challenge 4

Wednesday, November 6, 2024

8:53 AM

|  |
| --- |
| Challenge 4:  - Integrate with HttpClientFactory for .Net MVC Project  - Use HttpClient to call category endpoint of .Net Core Minimal API to get all categories data  - Replace the hardcoded category data with data retrieved from API  - Run 2 projects and verify the results |

    /\* "https": {

      "commandName": "Project",

      "dotnetRunMessages": true,

      "launchBrowser": true,

      "applicationUrl": "[https://localhost:7066;http://localhost:5240](https://localhost:7066;http:/localhost:5240)",

      "environmentVariables": {

        "ASPNETCORE\_ENVIRONMENT": "Development"

      }

    }, \*/

Tạo 1 repo trên git

Mở visual studio code -> source control -> clone -> trỏ đến 1 directory để clone về

Copy required code vào thư mục mới đó, trừ git nếu có

Quay lại visual studio code -> click add những file cần commit

-> .. -> commit & push

============

now please continue guide step by step (include which file, which folder and all details, include what for 'using', where to put the code, what to update or replace) to do the Challenge 4: - Integrate with HttpClientFactory for .Net MVC Project - Use HttpClient to call category endpoint of .Net Core Minimal API to get all categories data - Replace the hardcoded category data with data retrieved from API - Run 2 projects and verify the results

**ChatGPT đã nói:**

ChatGPT

Great! Let’s go through each step in detail to complete Challenge 4. We’ll create a .NET Core Minimal API to serve as the category endpoint and use HttpClientFactory in your .NET MVC project to retrieve data from it.

**Step 1: Create a .NET Core Minimal API Project for Category Endpoint**

1. **Open a terminal in Visual Studio Code** and navigate to your project folder.
2. Run the following command to create a new Minimal API project:  
     
   bash  
   Sao chép mã  
   dotnet new web -n CategoryApi  
     
   This creates a new folder named CategoryApi with the project files.
3. **Open CategoryApi in Visual Studio Code**:  
     
   bash  
   Sao chép mã  
   cd CategoryApi  
   code .
4. Open Program.cs and replace the code with the following to set up a simple endpoint:  
     
   csharp  
   Sao chép mã  
   using Microsoft.AspNetCore.Builder;  
   using Microsoft.Extensions.DependencyInjection;  
   using Microsoft.Extensions.Hosting;  
     
   var builder = WebApplication.CreateBuilder(args);  
   builder.Services.AddEndpointsApiExplorer();  
   builder.Services.AddSwaggerGen();  
     
   var app = builder.Build();  
     
   if (app.Environment.IsDevelopment())  
   {  
    app.UseSwagger();  
    app.UseSwaggerUI();  
   }  
     
   app.UseHttpsRedirection();  
     
   var categories = new List<string> { "Electronics", "Books", "Clothing", "Furniture" };  
     
   app.MapGet("/categories", () => categories);  
     
   app.Run();  
     
   This code defines a single GET endpoint at /categories that returns a hardcoded list of category data.
5. Run the Minimal API to make sure it’s working:  
     
   bash  
   Sao chép mã  
   dotnet run  
     
   The API should be running on <https://localhost:5001> (or another port shown in the terminal). Verify that <https://localhost:5001/categories> returns the list of categories in JSON format.

**Step 2: Configure HttpClientFactory in the MVC Project**

1. **Go back to your MVC project in Visual Studio Code** (in the CategoryProject folder).
2. Open Startup.cs or Program.cs (depending on your MVC project's setup).
3. In the ConfigureServices method (or equivalent), add the following code to configure HttpClientFactory:  
     
   csharp  
   Sao chép mã  
   services.AddHttpClient("CategoryApi", client =>  
   {  
    client.BaseAddress = new Uri("https://localhost:5001");  
   });  
     
   This registers an HttpClient for the API endpoint with a base address.

**Step 3: Modify the CategoryController to Use HttpClient to Call the API**

1. In **Visual Studio Code**, open the Controllers/CategoryController.cs file.
2. Update the controller to use HttpClient instead of hardcoded data:  
     
   csharp  
   Sao chép mã  
   using Microsoft.AspNetCore.Mvc;  
   using System.Collections.Generic;  
   using System.Net.Http;  
   using System.Text.Json;  
   using System.Threading.Tasks;  
     
   namespace CategoryProject.Controllers  
   {  
    public class CategoryController : Controller  
    {  
    private readonly IHttpClientFactory \_httpClientFactory;  
     
   public CategoryController(IHttpClientFactory httpClientFactory)  
    {  
    \_httpClientFactory = httpClientFactory;  
    }  
     
   public async Task<IActionResult> Index()  
    {  
    var client = \_httpClientFactory.CreateClient("CategoryApi");  
    var response = await client.GetAsync("/categories");  
     
   if (!response.IsSuccessStatusCode)  
    {  
    // Handle error (optional)  
    return View(new List<string>());  
    }  
     
   var responseContent = await response.Content.ReadAsStringAsync();  
    var categories = JsonSerializer.Deserialize<List<string>>(responseContent);  
     
   return View(categories);  
    }  
    }  
   }  
     
   This code uses HttpClientFactory to create a client, call the /categories endpoint, and deserialize the response into a list of categories.

**Step 4: Modify the View to Display API Data**

Since we already set up the view to display categories in a table format, there’s no need to change Views/Category/Index.cshtml. It should work with the updated data from the API.

**Step 5: Run Both Projects and Verify the Results**

1. **Run the Minimal API** (CategoryApi) in one terminal:  
     
   bash  
   Sao chép mã  
   cd CategoryApi  
   dotnet run  
     
   Ensure that it’s running on [**https://localhost:5001**](https://localhost:5001).
2. **Run the MVC Project** (CategoryProject) in another terminal:  
     
   bash  
   Sao chép mã  
   cd CategoryProject  
   dotnet run
3. Open a browser and navigate to <http://localhost:5000/Category> (or the appropriate port for your MVC project).

**Verification**

* The "Category" page in the MVC project should display a table with the categories retrieved from the Minimal API instead of the hardcoded data.
* If you don’t see the data, verify that both projects are running and check the console for any errors related to API requests.

Let me know if you encounter any issues or if you need further help!

4o

**Bạn đã nói:**

how to use swagger? I encounter a problem: 'IServiceCollection' does not contain a definition for 'AddSwaggerGen' and no accessible extension method 'AddSwaggerGen' accepting a first argument of type 'IServiceCollection' could be found (are you missing a using directive or an assembly reference?)

**ChatGPT đã nói:**

ChatGPT

The error you're encountering is because the AddSwaggerGen method is part of the **Swashbuckle.AspNetCore** library, which provides Swagger support in .NET Core. You'll need to install this package in your Minimal API project to use Swagger.

**Step 1: Install the Swashbuckle.AspNetCore Package**

1. Open a terminal in Visual Studio Code.
2. Navigate to the CategoryApi project folder.
3. Run the following command to install the Swagger package:  
     
   bash  
   Sao chép mã  
   dotnet add package Swashbuckle.AspNetCore

This will add the Swashbuckle.AspNetCore package to your project, which includes the AddSwaggerGen method.

**Step 2: Update Program.cs to Configure Swagger**

After the package is installed, you can set up Swagger in your Minimal API project as follows:

1. Open Program.cs in the CategoryApi project.
2. Ensure you have the necessary using directive at the top of the file:  
     
   csharp  
   Sao chép mã  
   using Microsoft.OpenApi.Models;
3. Update your code to configure Swagger. Your Program.cs should look like this:  
     
   csharp  
   Sao chép mã  
   using Microsoft.AspNetCore.Builder;  
   using Microsoft.Extensions.DependencyInjection;  
   using Microsoft.Extensions.Hosting;  
   using Microsoft.OpenApi.Models;  
     
   var builder = WebApplication.CreateBuilder(args);  
     
   // Add services to the container.  
   builder.Services.AddEndpointsApiExplorer();  
   builder.Services.AddSwaggerGen(c =>  
   {  
    c.SwaggerDoc("v1", new OpenApiInfo { Title = "Category API", Version = "v1" });  
   });  
     
   var app = builder.Build();  
     
   // Configure the HTTP request pipeline.  
   if (app.Environment.IsDevelopment())  
   {  
    app.UseSwagger();  
    app.UseSwaggerUI(c =>  
    {  
    c.SwaggerEndpoint("/swagger/v1/swagger.json", "Category API V1");  
    });  
   }  
     
   app.UseHttpsRedirection();  
     
   var categories = new List<string> { "Electronics", "Books", "Clothing", "Furniture" };  
     
   app.MapGet("/categories", () => categories);  
     
   app.Run();

**Step 3: Run the API and Access Swagger**

1. In the terminal, run the API:  
     
   bash  
   Sao chép mã  
   dotnet run
2. Once the API is running, open a browser and navigate to the Swagger UI at <https://localhost:5001/swagger> (or the appropriate port shown in your terminal).

You should see the Swagger interface, which lists the /categories endpoint. You can test the endpoint directly from this interface by clicking on it and selecting "Try it out."

Swagger provides a convenient way to explore and test your API endpoints, making it especially useful for debugging and verifying API responses. Let me know if this resolves the issue or if you encounter any other problems!

4o

**Bạn đã nói:**

thank you, the problem is resolved

**ChatGPT đã nói:**

ChatGPT

Đã cập nhật bộ nhớ

You're very welcome! I'm glad to hear the problem is resolved. If you have any more questions about Swagger, integrating APIs, or anything else, feel free to reach out. Happy coding! 😊

4o

**Bạn đã nói:**

I have problem with the Program.cs in CategoryProject: The name 'services' does not exist in the current context

**ChatGPT đã nói:**

ChatGPT

The error you're seeing usually occurs because the services variable is not defined or is outside the scope in Program.cs. This can happen if you're using a newer version of .NET, which may have a different setup for Program.cs compared to previous versions.

Let’s fix it by ensuring you’re using the correct approach to configure services in Program.cs.

**Solution**

If you're using .NET 6 or later, the Program.cs file follows a minimal hosting model, where there is no separate Startup.cs. Here’s the updated approach for defining services in Program.cs:

1. Open Program.cs in the **CategoryProject**.
2. Modify the code to ensure services is accessible. Here’s an example structure that works with .NET 6 or later:  
     
   csharp  
   Sao chép mã  
   using Microsoft.AspNetCore.Builder;  
   using Microsoft.Extensions.DependencyInjection;  
   using Microsoft.Extensions.Hosting;  
     
   var builder = WebApplication.CreateBuilder(args);  
     
   // Add services to the container  
   builder.Services.AddControllersWithViews();  
     
   // Register HttpClient for API access  
   builder.Services.AddHttpClient("CategoryApi", client =>  
   {  
    client.BaseAddress = new Uri("https://localhost:5001");  
   });  
     
   var app = builder.Build();  
     
   // Configure the HTTP request pipeline  
   if (!app.Environment.IsDevelopment())  
   {  
    app.UseExceptionHandler("/Home/Error");  
    app.UseHsts();  
   }  
     
   app.UseHttpsRedirection();  
   app.UseStaticFiles();  
     
   app.UseRouting();  
     
   app.UseAuthorization();  
     
   app.MapControllerRoute(  
    name: "default",  
    pattern: "{controller=Home}/{action=Index}/{id?}");  
     
   app.Run();

In this approach:

* builder.Services is used to add services like AddControllersWithViews() and AddHttpClient.
* builder refers to the WebApplicationBuilder, so we don’t need a separate services variable.

**Explanation**

In the minimal hosting model:

* **builder.Services** is used to register services like HttpClient and MVC controllers.
* **app.Use... methods** configure middleware for handling requests.

**Additional Notes**

Make sure:

* using statements for Microsoft.AspNetCore.Builder, Microsoft.Extensions.DependencyInjection, and Microsoft.Extensions.Hosting are included at the top of Program.cs.

After making these changes, save the file and try running the project again. This should resolve the 'services' does not exist error. Let me know if you have further questions!

4o

**Bạn đã nói:**

thank you, my mistake to register the service after define var app = builder.Build()

**ChatGPT đã nói:**

ChatGPT

You're very welcome! That’s a common mistake, especially with the newer, streamlined Program.cs structure in .NET 6 and later. The order can be a bit tricky because all service registrations must happen before builder.Build().

I’m glad we caught it! If you have any more questions or run into other issues, don’t hesitate to ask. Good luck with the rest of your project! 😊

4o

**Bạn đã nói:**

I access the <http://localhost:5223/Category> with the error An unhandled exception occurred while processing the request. AuthenticationException: Cannot determine the frame size or a corrupted frame was received. System.Net.Security.SslStream.GetFrameSize(ReadOnlySpan<byte> buffer) HttpRequestException: The SSL connection could not be established, see inner exception. System.Net.Http.ConnectHelper.EstablishSslConnectionAsync(SslClientAuthenticationOptions sslOptions, HttpRequestMessage request, bool async, Stream stream, CancellationToken cancellationToken)

**ChatGPT đã nói:**

ChatGPT

The error you’re encountering usually happens when there's an issue with the SSL/TLS connection between your .NET MVC app and the Minimal API. This can happen if the Minimal API is running on HTTPS (<https://localhost:5001>), but your MVC project is trying to connect to it in a way that doesn’t handle HTTPS properly.

Here are a few steps to troubleshoot and resolve this error:

**Solution 1: Ensure Both Projects Use HTTPS**

Make sure both the MVC project and the Minimal API project are running on HTTPS (SSL/TLS). By default, .NET uses HTTPS for local development, so let's verify that the URLs match.

1. **Check the Minimal API URL**:
   * **Run the Minimal API project, and note the URL it runs on (e.g.,** [**https://localhost:5001**](https://localhost:5001)**). Ensure it's using HTTPS.**
2. **Configure the MVC Project to Use the Correct URL**:
   * **In the MVC project, open Program.cs where you set up HttpClient.**
   * **Make sure the BaseAddress for HttpClient is set to the correct HTTPS URL of the Minimal API:  
       
     csharp  
     Sao chép mã  
     builder.Services.AddHttpClient("CategoryApi", client =>  
     {  
      client.BaseAddress = new Uri("https://localhost:5001"); // Use the HTTPS URL  
     });**

**Solution 2: Disable SSL Verification Temporarily (for Development Only)**

If you’re still encountering SSL issues even after ensuring the URLs match, you can bypass SSL certificate validation as a quick workaround for local development. **Only use this in a development environment, as it’s not secure**.

1. Update the HttpClient setup in Program.cs as follows:  
     
   csharp  
   Sao chép mã  
   builder.Services.AddHttpClient("CategoryApi", client =>  
   {  
    client.BaseAddress = new Uri("https://localhost:5001");  
   }).ConfigurePrimaryHttpMessageHandler(() =>  
    new HttpClientHandler  
    {  
    ServerCertificateCustomValidationCallback = (message, cert, chain, errors) => true  
    });  
     
   This code disables SSL verification, allowing requests to proceed even if there’s an issue with the SSL certificate.

**Solution 3: Use HTTP Instead of HTTPS (for Development Only)**

Alternatively, you can configure both projects to use HTTP instead of HTTPS, which can sometimes avoid SSL-related issues.

1. **Run the Minimal API on HTTP**:
   * **Open Properties/launchSettings.json in the Minimal API project.**
   * **Under the profiles section, set the applicationUrl to use HTTP instead of HTTPS:  
       
     json  
     Sao chép mã  
     "applicationUrl": "**[**http://localhost:5001**](http://localhost:5001)**"**
   * **Save the file, and restart the Minimal API project. It should now run on** [**http://localhost:5001**](http://localhost:5001)**.**
2. **Update the MVC Project to Use HTTP**:
   * **In the MVC project’s Program.cs, set the BaseAddress for HttpClient to** [**http://localhost:5001**](http://localhost:5001) **(HTTP instead of HTTPS):  
       
     csharp  
     Sao chép mã  
     builder.Services.AddHttpClient("CategoryApi", client =>  
     {  
      client.BaseAddress = new Uri("http://localhost:5001"); // Use HTTP URL  
     });**

**Solution 4: Trust the Development Certificate**

If your project was generated with HTTPS enabled and you have SSL issues due to an untrusted certificate, you can try trusting the development certificate:

1. In a terminal, run the following command to trust the .NET development certificate:  
     
   bash  
   Sao chép mã  
   dotnet dev-certs https --trust
2. Restart both projects after running this command.

After applying one of these solutions, try running both projects again and access <http://localhost:5223/Category>. Let me know if this resolves the issue!

4o

**Bạn đã nói:**

I still have error: fail: Microsoft.AspNetCore.Diagnostics.DeveloperExceptionPageMiddleware[1] An unhandled exception has occurred while executing the request. System.Net.Http.HttpRequestException: The SSL connection could not be established, see inner exception. ---> System.Security.Authentication.AuthenticationException: Cannot determine the frame size or a corrupted frame was received. at System.Net.Security.SslStream.GetFrameSize(ReadOnlySpan1 buffer) at System.Net.Security.SslStream.EnsureFullTlsFrameAsync[TIOAdapter](CancellationToken cancellationToken, Int32 estimatedSize) at System.Runtime.CompilerServices.PoolingAsyncValueTaskMethodBuilder1.StateMachineBox1.System.Threading.Tasks.Sources.IValueTaskSource<TResult>.GetResult(Int16 token) at System.Net.Security.SslStream.ReceiveHandshakeFrameAsync[TIOAdapter](CancellationToken cancellationToken) at System.Net.Security.SslStream.ForceAuthenticationAsync[TIOAdapter](Boolean receiveFirst, Byte[] reAuthenticationData, CancellationToken cancellationToken) at System.Net.Http.ConnectHelper.EstablishSslConnectionAsync(SslClientAuthenticationOptions sslOptions, HttpRequestMessage request, Boolean async, Stream stream, CancellationToken cancellationToken) --- End of inner exception stack trace --- at System.Net.Http.ConnectHelper.EstablishSslConnectionAsync(SslClientAuthenticationOptions sslOptions, HttpRequestMessage request, Boolean async, Stream stream, CancellationToken cancellationToken) at System.Net.Http.HttpConnectionPool.ConnectAsync(HttpRequestMessage request, Boolean async, CancellationToken cancellationToken) at System.Net.Http.HttpConnectionPool.CreateHttp11ConnectionAsync(HttpRequestMessage request, Boolean async, CancellationToken cancellationToken) at System.Net.Http.HttpConnectionPool.AddHttp11ConnectionAsync(QueueItem queueItem) at System.Threading.Tasks.TaskCompletionSourceWithCancellation1.WaitWithCancellationAsync(CancellationToken cancellationToken) at System.Net.Http.HttpConnectionPool.SendWithVersionDetectionAndRetryAsync(HttpRequestMessage request, Boolean async, Boolean doRequestAuth, CancellationToken cancellationToken) at System.Net.Http.DiagnosticsHandler.SendAsyncCore(HttpRequestMessage request, Boolean async, CancellationToken cancellationToken) at System.Net.Http.RedirectHandler.SendAsync(HttpRequestMessage request, Boolean async, CancellationToken cancellationToken) at Microsoft.Extensions.Http.Logging.LoggingHttpMessageHandler.<SendCoreAsync>g\_\_Core|5\_0(HttpRequestMessage request, Boolean useAsync, CancellationToken cancellationToken) at Microsoft.Extensions.Http.Logging.LoggingScopeHttpMessageHandler.<SendCoreAsync>g\_\_Core|5\_0(HttpRequestMessage request, Boolean useAsync, CancellationToken cancellationToken) at System.Net.Http.HttpClient.<SendAsync>g\_\_Core|83\_0(HttpRequestMessage request, HttpCompletionOption completionOption, CancellationTokenSource cts, Boolean disposeCts, CancellationTokenSource pendingRequestsCts, CancellationToken originalCancellationToken) at CategoryProject.Controllers.CategoryController.Index() in C:\Data-cKhanh\.NET\Challenges\Challenge04\CategoryProject\Controllers\CategoryController.cs:line 34 at Microsoft.AspNetCore.Mvc.Infrastructure.ActionMethodExecutor.TaskOfIActionResultExecutor.Execute(ActionContext actionContext, IActionResultTypeMapper mapper, ObjectMethodExecutor executor, Object controller, Object[] arguments) at Microsoft.AspNetCore.Mvc.Infrastructure.ControllerActionInvoker.<InvokeActionMethodAsync>g\_\_Awaited|12\_0(ControllerActionInvoker invoker, ValueTask1 actionResultValueTask) at Microsoft.AspNetCore.Mvc.Infrastructure.ControllerActionInvoker.<InvokeNextActionFilterAsync>g\_\_Awaited|10\_0(ControllerActionInvoker invoker, Task lastTask, State next, Scope scope, Object state, Boolean isCompleted) at Microsoft.AspNetCore.Mvc.Infrastructure.ControllerActionInvoker.Rethrow(ActionExecutedContextSealed context) at Microsoft.AspNetCore.Mvc.Infrastructure.ControllerActionInvoker.Next(State& next, Scope& scope, Object& state, Boolean& isCompleted) at Microsoft.AspNetCore.Mvc.Infrastructure.ControllerActionInvoker.<InvokeInnerFilterAsync>g\_\_Awaited|13\_0(ControllerActionInvoker invoker, Task lastTask, State next, Scope scope, Object state, Boolean isCompleted) at Microsoft.AspNetCore.Mvc.Infrastructure.ResourceInvoker.<InvokeNextResourceFilter>g\_\_Awaited|25\_0(ResourceInvoker invoker, Task lastTask, State next, Scope scope, Object state, Boolean isCompleted) at Microsoft.AspNetCore.Mvc.Infrastructure.ResourceInvoker.Rethrow(ResourceExecutedContextSealed context) at Microsoft.AspNetCore.Mvc.Infrastructure.ResourceInvoker.Next(State& next, Scope& scope, Object& state, Boolean& isCompleted) at Microsoft.AspNetCore.Mvc.Infrastructure.ResourceInvoker.<InvokeFilterPipelineAsync>g\_\_Awaited|20\_0(ResourceInvoker invoker, Task lastTask, State next, Scope scope, Object state, Boolean isCompleted) at Microsoft.AspNetCore.Mvc.Infrastructure.ResourceInvoker.<InvokeAsync>g\_\_Awaited|17\_0(ResourceInvoker invoker, Task task, IDisposable scope) at Microsoft.AspNetCore.Mvc.Infrastructure.ResourceInvoker.<InvokeAsync>g\_\_Awaited|17\_0(ResourceInvoker invoker, Task task, IDisposable scope) at Microsoft.AspNetCore.Authorization.AuthorizationMiddleware.Invoke(HttpContext context) at Microsoft.AspNetCore.Authentication.AuthenticationMiddleware.Invoke(HttpContext context) at Microsoft.AspNetCore.Diagnostics.DeveloperExceptionPageMiddlewareImpl.Invoke(HttpContext context)

**ChatGPT đã nói:**

ChatGPT

This SSL connection issue can happen for several reasons, such as certificate mismatches, port conflicts, or SSL configuration issues. Here are some more targeted troubleshooting steps to try:

**Step 1: Check if Both Projects Are Running on HTTPS and Matching Ports**

Make sure both the **CategoryProject (MVC)** and **CategoryApi (Minimal API)** are running on HTTPS with the correct port configurations.

1. **Confirm the Port in CategoryApi**:
   * **Start the CategoryApi project.**
   * **Note the HTTPS URL in the console output, like** [**https://localhost:5001**](https://localhost:5001)**.**
2. **Verify the HttpClient BaseAddress in CategoryProject**:
   * **Open Program.cs in the CategoryProject.**
   * **Ensure the BaseAddress for HttpClient matches the HTTPS URL used by the Minimal API:  
       
     csharp  
     Sao chép mã  
     builder.Services.AddHttpClient("CategoryApi", client =>  
     {  
      client.BaseAddress = new Uri("https://localhost:5001"); // Replace with the actual HTTPS URL if different  
     });**

**Step 2: Bypass SSL Verification (for Local Development Only)**

If the SSL error persists, you can configure HttpClient to ignore SSL verification temporarily, which is suitable for development.

Update your **CategoryProject** Program.cs like this:

csharp

Sao chép mã

builder.Services.AddHttpClient("CategoryApi", client =>  
{  
 client.BaseAddress = new Uri("https://localhost:5001"); // Use your Minimal API HTTPS URL  
}).ConfigurePrimaryHttpMessageHandler(() =>  
 new HttpClientHandler  
 {  
 ServerCertificateCustomValidationCallback = (message, cert, chain, sslPolicyErrors) => true  
 });

**Step 3: Use HTTP Instead of HTTPS (for Local Development Only)**

If bypassing SSL verification doesn't resolve the issue, another approach is to switch both projects to use HTTP instead of HTTPS.

1. **Update CategoryApi to Use HTTP**:
   * **Open Properties/launchSettings.json in the CategoryApi project.**
   * **Under the profiles section, set the applicationUrl to use HTTP:  
       
     json  
     Sao chép mã  
     "applicationUrl": "**[**http://localhost:5001**](http://localhost:5001)**"**
   * **Restart CategoryApi to use HTTP.**
2. **Update CategoryProject to Use HTTP**:
   * **In CategoryProject Program.cs, change the BaseAddress for HttpClient to use HTTP:  
       
     csharp  
     Sao chép mã  
     builder.Services.AddHttpClient("CategoryApi", client =>  
     {  
      client.BaseAddress = new Uri("http://localhost:5001");  
     });**

**Step 4: Trust the .NET Development Certificate**

If you haven’t already, you can try trusting the .NET development certificate, which may resolve SSL-related issues locally.

Run this command in a terminal:

bash

Sao chép mã

dotnet dev-certs https --trust

Then restart both projects.

**Step 5: Clear SSL Cache and Restart**

Sometimes, SSL issues can be due to cached certificates. You can try restarting your machine to clear any cached SSL settings and ensure a fresh connection.

After trying these steps, run both the MVC and API projects again and try accessing <http://localhost:5223/Category> or <https://localhost:5223/Category>. Let me know if the error persists!

4o

**Bạn đã nói:**

can I continue? I run API project first, then run the MVC project and the Category web page now show the list of categories

**ChatGPT đã nói:**

ChatGPT

Đã cập nhật bộ nhớ

That's fantastic news! It sounds like everything is working smoothly. If you need any more guidance or have additional features to add, feel free to ask. Happy coding! 😊

4o

**Bạn đã nói:**

sorry my mistake, The Category web page not show the list of categories, seem it cannot retrieve the API correctly

**ChatGPT đã nói:**

ChatGPT

No problem! Let's go through some troubleshooting steps to make sure your MVC project is set up correctly to retrieve data from the API.

**Step 1: Confirm the API Endpoint URL**

Ensure that the **Minimal API** project is running and note the exact URL for the /categories endpoint. You can test it by navigating to the URL in your browser or using a tool like [Postman](https://www.postman.com/) or [curl](https://curl.se/).

* For example, the URL should be something like <https://localhost:5001/categories>.

**Step 2: Verify HttpClient Configuration in Program.cs**

In the **MVC project** (CategoryProject), open Program.cs and check that you have configured HttpClient to use the correct base address:

csharp

Sao chép mã

builder.Services.AddHttpClient("CategoryApi", client =>  
{  
 client.BaseAddress = new Uri("https://localhost:5001"); // Make sure this matches the API's HTTPS URL  
});

**Step 3: Update the Controller to Use the HttpClient Factory Correctly**

In your **CategoryController.cs** file, make sure you're calling the API correctly. The code should look something like this:

1. **Ensure the using directives** for System.Net.Http and System.Threading.Tasks are included at the top of the file.
2. Use IHttpClientFactory to create the HttpClient instance. Here's an example setup:  
     
   csharp  
   Sao chép mã  
   using Microsoft.AspNetCore.Mvc;  
   using System.Net.Http;  
   using System.Threading.Tasks;  
   using System.Collections.Generic;  
   using Newtonsoft.Json;  
     
   namespace CategoryProject.Controllers  
   {  
    public class CategoryController : Controller  
    {  
    private readonly IHttpClientFactory \_httpClientFactory;  
     
   public CategoryController(IHttpClientFactory httpClientFactory)  
    {  
    \_httpClientFactory = httpClientFactory;  
    }  
     
   public async Task<IActionResult> Index()  
    {  
    var client = \_httpClientFactory.CreateClient("CategoryApi");  
     
   // Attempt to retrieve data from the API  
    HttpResponseMessage response = await client.GetAsync("/categories");  
     
   if (response.IsSuccessStatusCode)  
    {  
    var data = await response.Content.ReadAsStringAsync();  
    var categories = JsonConvert.DeserializeObject<List<string>>(data);  
    return View(categories);  
    }  
    else  
    {  
    // Handle API error response  
    ViewBag.Error = "Could not retrieve categories.";  
    return View(new List<string>()); // Return an empty list or error view  
    }  
    }  
    }  
   }

**Step 4: Verify the Razor View**

In **CategoryProject/Views/Category/Index.cshtml**, make sure your view is set up to display the list of categories. Here’s a simple example:

html

Sao chép mã

@model List<string>

<h2>Categories</h2>

@if (ViewBag.Error != null)  
{  
 <p style="color: red;">@ViewBag.Error</p>  
}  
else  
{  
 <table>  
 <thead>  
 <tr>  
 <th>Category Name</th>  
 </tr>  
 </thead>  
 <tbody>  
 @foreach (var category in Model)  
 {  
 <tr>  
 <td>@category</td>  
 </tr>  
 }  
 </tbody>  
 </table>  
}

**Step 5: Run Both Projects and Check Logs**

1. Start the API project first and make sure it’s running.
2. Then start the MVC project.
3. If the list still doesn’t display, check the console output of both projects for any error messages.

Let me know how this works out, or if any errors show up in the console output!

4o

**Bạn đã nói:**

facing error: CategoryProject\Controllers\CategoryController.cs(31, 38): error CS0103: The name 'JsonConvert' does not exist in the current context [C:\Data-cKhanh\.NET\ Challenges\.NET-Challenge04\CategoryProject\CategoryProject.csproj] The build failed. Fix the build errors and run again.

**ChatGPT đã nói:**

ChatGPT

The error indicates that the JsonConvert class, which is part of the **Newtonsoft.Json** library, is not recognized. To fix this, you need to add a reference to the **Newtonsoft.Json** package in your project.

**Step 1: Install Newtonsoft.Json**

In Visual Studio Code, open the integrated terminal and run the following command to install the **Newtonsoft.Json** package:

bash

Sao chép mã

dotnet add package Newtonsoft.Json

This will add the **Newtonsoft.Json** library to your project, which includes JsonConvert.

**Step 2: Import Newtonsoft.Json in Your Controller**

Once the package is installed, add the following using directive at the top of your **CategoryController.cs** file:

csharp

Sao chép mã

using Newtonsoft.Json;

**Step 3: Rebuild and Run the Project**

After adding the package and import statement, try rebuilding and running the project:

bash

Sao chép mã

dotnet build  
dotnet run

This should resolve the JsonConvert error. Now, check if the categories display correctly on the Category page. Let me know if this works or if you encounter any further issues!

4o

**Bạn đã nói:**

facing error: fail: Microsoft.Extensions.Hosting.Internal.Host[11] Hosting failed to start System.IO.IOException: Failed to bind to address <http://127.0.0.1:5240>: address already in use. ---> Microsoft.AspNetCore.Connections.AddressInUseException: Only one usage of each socket address (protocol/network address/port) is normally permitted. ---> System.Net.Sockets.SocketException (10048): Only one usage of each socket address (protocol/network address/port) is normally permitted. at System.Net.Sockets.Socket.UpdateStatusAfterSocketErrorAndThrowException(SocketError error, Boolean disconnectOnFailure, String callerName) at System.Net.Sockets.Socket.DoBind(EndPoint endPointSnapshot, SocketAddress socketAddress) at System.Net.Sockets.Socket.Bind(EndPoint localEP) at Microsoft.AspNetCore.Server.Kestrel.Transport.Sockets.SocketTransportOptions.CreateDefaultBoundListenSocket(EndPoint endpoint) at Microsoft.AspNetCore.Server.Kestrel.Transport.Sockets.SocketConnectionListener.Bind() --- End of inner exception stack trace --- at Microsoft.AspNetCore.Server.Kestrel.Transport.Sockets.SocketConnectionListener.Bind() at Microsoft.AspNetCore.Server.Kestrel.Transport.Sockets.SocketTransportFactory.BindAsync(EndPoint endpoint, CancellationToken cancellationToken) at Microsoft.AspNetCore.Server.Kestrel.Core.Internal.Infrastructure.TransportManager.BindAsync(EndPoint endPoint, ConnectionDelegate connectionDelegate, EndpointConfig endpointConfig, CancellationToken cancellationToken) at Microsoft.AspNetCore.Server.Kestrel.Core.KestrelServerImpl.<>c\_\_DisplayClass28\_01.<<StartAsync>g\_\_OnBind|0>d.MoveNext() --- End of stack trace from previous location --- at Microsoft.AspNetCore.Server.Kestrel.Core.Internal.AddressBinder.BindEndpointAsync(ListenOptions endpoint, AddressBindContext context, CancellationToken cancellationToken) --- End of inner exception stack trace --- at Microsoft.AspNetCore.Server.Kestrel.Core.Internal.AddressBinder.BindEndpointAsync(ListenOptions endpoint, AddressBindContext context, CancellationToken cancellationToken) at Microsoft.AspNetCore.Server.Kestrel.Core.LocalhostListenOptions.BindAsync(AddressBindContext context, CancellationToken cancellationToken) at Microsoft.AspNetCore.Server.Kestrel.Core.Internal.AddressBinder.AddressesStrategy.BindAsync(AddressBindContext context, CancellationToken cancellationToken) at Microsoft.AspNetCore.Server.Kestrel.Core.Internal.AddressBinder.BindAsync(ListenOptions[] listenOptions, AddressBindContext context, Func2 useHttps, CancellationToken cancellationToken) at Microsoft.AspNetCore.Server.Kestrel.Core.KestrelServerImpl.BindAsync(CancellationToken cancellationToken) at Microsoft.AspNetCore.Server.Kestrel.Core.KestrelServerImpl.StartAsync[TContext](IHttpApplication1 application, CancellationToken cancellationToken) at Microsoft.AspNetCore.Hosting.GenericWebHostService.StartAsync(CancellationToken cancellationToken) at Microsoft.Extensions.Hosting.Internal.Host.<StartAsync>b\_\_15\_1(IHostedService service, CancellationToken token) at Microsoft.Extensions.Hosting.Internal.Host.ForeachService[T](IEnumerable1 services, CancellationToken token, Boolean concurrent, Boolean abortOnFirstException, List1 exceptions, Func3 operation) Unhandled exception. System.IO.IOException: Failed to bind to address <http://127.0.0.1:5240>: address already in use. ---> Microsoft.AspNetCore.Connections.AddressInUseException: Only one usage of each socket address (protocol/network address/port) is normally permitted. ---> System.Net.Sockets.SocketException (10048): Only one usage of each socket address (protocol/network address/port) is normally permitted. at System.Net.Sockets.Socket.UpdateStatusAfterSocketErrorAndThrowException(SocketError error, Boolean disconnectOnFailure, String callerName) at System.Net.Sockets.Socket.DoBind(EndPoint endPointSnapshot, SocketAddress socketAddress) at System.Net.Sockets.Socket.Bind(EndPoint localEP) at Microsoft.AspNetCore.Server.Kestrel.Transport.Sockets.SocketTransportOptions.CreateDefaultBoundListenSocket(EndPoint endpoint) at Microsoft.AspNetCore.Server.Kestrel.Transport.Sockets.SocketConnectionListener.Bind() --- End of inner exception stack trace --- at Microsoft.AspNetCore.Server.Kestrel.Transport.Sockets.SocketConnectionListener.Bind() at Microsoft.AspNetCore.Server.Kestrel.Transport.Sockets.SocketTransportFactory.BindAsync(EndPoint endpoint, CancellationToken cancellationToken) at Microsoft.AspNetCore.Server.Kestrel.Core.Internal.Infrastructure.TransportManager.BindAsync(EndPoint endPoint, ConnectionDelegate connectionDelegate, EndpointConfig endpointConfig, CancellationToken cancellationToken) at Microsoft.AspNetCore.Server.Kestrel.Core.KestrelServerImpl.<>c\_\_DisplayClass28\_01.<<StartAsync>g\_\_OnBind|0>d.MoveNext() --- End of stack trace from previous location --- at Microsoft.AspNetCore.Server.Kestrel.Core.Internal.AddressBinder.BindEndpointAsync(ListenOptions endpoint, AddressBindContext context, CancellationToken cancellationToken) --- End of inner exception stack trace --- at Microsoft.AspNetCore.Server.Kestrel.Core.Internal.AddressBinder.BindEndpointAsync(ListenOptions endpoint, AddressBindContext context, CancellationToken cancellationToken) at Microsoft.AspNetCore.Server.Kestrel.Core.LocalhostListenOptions.BindAsync(AddressBindContext context, CancellationToken cancellationToken) at Microsoft.AspNetCore.Server.Kestrel.Core.Internal.AddressBinder.AddressesStrategy.BindAsync(AddressBindContext context, CancellationToken cancellationToken) at Microsoft.AspNetCore.Server.Kestrel.Core.Internal.AddressBinder.BindAsync(ListenOptions[] listenOptions, AddressBindContext context, Func2 useHttps, CancellationToken cancellationToken) at Microsoft.AspNetCore.Server.Kestrel.Core.KestrelServerImpl.BindAsync(CancellationToken cancellationToken) at Microsoft.AspNetCore.Server.Kestrel.Core.KestrelServerImpl.StartAsync[TContext](IHttpApplication1 application, CancellationToken cancellationToken) at Microsoft.AspNetCore.Hosting.GenericWebHostService.StartAsync(CancellationToken cancellationToken) at Microsoft.Extensions.Hosting.Internal.Host.<StartAsync>b\_\_15\_1(IHostedService service, CancellationToken token) at Microsoft.Extensions.Hosting.Internal.Host.ForeachService[T](IEnumerable1 services, CancellationToken token, Boolean concurrent, Boolean abortOnFirstException, List1 exceptions, Func3 operation) at Microsoft.Extensions.Hosting.Internal.Host.StartAsync(CancellationToken cancellationToken) at Microsoft.Extensions.Hosting.HostingAbstractionsHostExtensions.RunAsync(IHost host, CancellationToken token) at Microsoft.Extensions.Hosting.HostingAbstractionsHostExtensions.RunAsync(IHost host, CancellationToken token) at Microsoft.Extensions.Hosting.HostingAbstractionsHostExtensions.Run(IHost host) at Program.<Main>$(String[] args) in C:\Data-cKhanh\.NET\Challenges\.NET-Challenge04\CategoryProject\Program.cs:line 39

**ChatGPT đã nói:**

ChatGPT

This error indicates that the port 5240 is already in use by another application, preventing your **CategoryProject** (MVC project) from starting on that port.

Here's how to resolve it:

**Option 1: Change the Port Number**

1. Open **launchSettings.json** located in CategoryProject/Properties.
2. Look for the section under "profiles" where "applicationUrl" is defined.
3. Update the port to a different number, such as 5241:  
     
   json  
   Sao chép mã  
   "applicationUrl": "<http://localhost:5241>"
4. Save the file and try running the application again.

**Option 2: Close the Process Using the Port**

If you want to keep using port 5240, find and stop the process currently using it:

* **On Windows:**
  1. Open Command Prompt as Administrator.
  2. Run the following command to find the process using port 5240:  
       
     bash  
     Sao chép mã  
     netstat -aon | findstr :5240
  3. Note the **PID** (Process ID) from the output, then run:  
       
     bash  
     Sao chép mã  
     taskkill /PID <PID> /F
* **On macOS/Linux:**
  1. Open a terminal.
  2. Run:  
       
     bash  
     Sao chép mã  
     lsof -i :5240
  3. Note the **PID**, then stop it with:  
       
     bash  
     Sao chép mã  
     kill -9 <PID>

After releasing the port, try running the application again.

4o

**Bạn đã nói:**

seem the error is caused by execute dotnet run for the CategoryApi first, then run for the CategoryProject after that

**ChatGPT đã nói:**

ChatGPT

You're correct; running both projects sequentially with dotnet run may cause port conflicts if both are set to run on similar ports. Here are two ways to handle running both projects without port conflicts.

**Option 1: Use Different Ports for Each Project**

1. **Open launchSettings.json** for both projects and assign each project a unique port if they are using the same one.
   * **For example:**
     + **In CategoryApi: set it to** [**http://localhost:5001**](http://localhost:5001)**.**
     + **In CategoryProject: set it to** [**http://localhost:5241**](http://localhost:5241)**.**
2. **Run Each Project Individually**:
   * **Run the API project first by navigating to its directory and using dotnet run.**
   * **Then run the MVC project in its directory with dotnet run.**

**Option 2: Use Debug Configuration in Visual Studio Code to Run Both Projects Simultaneously**

1. Open the **launch.json** file in .vscode and create configurations for both projects:  
     
   json  
   Sao chép mã  
   {  
    "version": "0.2.0",  
    "configurations": [  
    {  
    "name": "Launch CategoryApi",  
    "type": "coreclr",  
    "request": "launch",  
    "program": "${workspaceFolder}/CategoryApi/bin/Debug/net6.0/CategoryApi.dll",  
    "args": [],  
    "cwd": "${workspaceFolder}/CategoryApi",  
    "stopAtEntry": false,  
    "serverReadyAction": {  
    "action": "openExternally",  
    "pattern": "\\bNow listening on:\\s+(https?://\\S+)"  
    },  
    "env": {  
    "ASPNETCORE\_ENVIRONMENT": "Development"  
    },  
    "preLaunchTask": "build",  
    "console": "internalConsole"  
    },  
    {  
    "name": "Launch CategoryProject",  
    "type": "coreclr",  
    "request": "launch",  
    "program": "${workspaceFolder}/CategoryProject/bin/Debug/net6.0/CategoryProject.dll",  
    "args": [],  
    "cwd": "${workspaceFolder}/CategoryProject",  
    "stopAtEntry": false,  
    "serverReadyAction": {  
    "action": "openExternally",  
    "pattern": "\\bNow listening on:\\s+(https?://\\S+)"  
    },  
    "env": {  
    "ASPNETCORE\_ENVIRONMENT": "Development"  
    },  
    "preLaunchTask": "build",  
    "console": "internalConsole"  
    }  
    ]  
   }
2. After setting this up, open the **Run and Debug** panel in VS Code and choose **Launch CategoryApi** to start the API project, followed by **Launch CategoryProject** to start the MVC project.

Using either method, you should be able to run both projects without port conflicts. Let me know if this resolves the issue!

4o

**Bạn đã nói:**

if run 2 projects on different ports, can I retrieve data from the CategoryApi?

**ChatGPT đã nói:**

ChatGPT

Yes, you can retrieve data from **CategoryApi** even when it’s running on a different port than **CategoryProject**. You’ll just need to ensure that the **HttpClient** in your **CategoryProject** (the MVC project) points to the correct URL for the **CategoryApi**.

**Steps to Access CategoryApi from CategoryProject on Different Ports**

1. **Identify the URL of the API**:
   * **If CategoryApi runs on** [**http://localhost:5001**](http://localhost:5001)**, that’s the URL you’ll use in CategoryProject to fetch the data.**
2. **Update the HttpClient Configuration** in CategoryProject:
   * **In CategoryController.cs or wherever you’re making the HttpClient request, use the full URL of the CategoryApi.  
     Here’s an example of how to configure the HttpClient call in CategoryController.cs:  
       
     csharp  
     Sao chép mã  
     using System.Net.Http;  
     using System.Threading.Tasks;  
     using Microsoft.AspNetCore.Mvc;  
     using Newtonsoft.Json;  
     using CategoryProject.Models;  
       
     public class CategoryController : Controller  
     {  
      private readonly IHttpClientFactory \_clientFactory;  
       
     public CategoryController(IHttpClientFactory clientFactory)  
      {  
      \_clientFactory = clientFactory;  
      }  
       
     public async Task<IActionResult> Index()  
      {  
      var client = \_clientFactory.CreateClient();  
      client.BaseAddress = new Uri("http://localhost:5001"); // URL of the CategoryApi project  
        
      HttpResponseMessage response = await client.GetAsync("/api/categories"); // Adjust the route as necessary  
      if (response.IsSuccessStatusCode)  
      {  
      var jsonString = await response.Content.ReadAsStringAsync();  
      var categories = JsonConvert.DeserializeObject<List<Category>>(jsonString);  
      return View(categories);  
      }  
      else  
      {  
      // Handle error (e.g., display a message or log the error)  
      return View(new List<Category>()); // Return an empty list if the API call fails  
      }  
      }  
     }**
   * **Replace** [**http://localhost:5001**](http://localhost:5001) **with the actual port of your CategoryApi project.**
3. **Test the Connection**:
   * **Run CategoryApi on its designated port (e.g.,** [**http://localhost:5001**](http://localhost:5001)**).**
   * **Then run CategoryProject on its designated port (e.g.,** [**http://localhost:5241**](http://localhost:5241)**).**
   * **Access the Category page in CategoryProject; it should now retrieve data from CategoryApi.**

**Key Points**

* The port difference doesn’t prevent communication; as long as **CategoryProject** points to the correct **CategoryApi** URL, it should work fine.
* Ensure CORS is enabled in **CategoryApi** if you ever need to access it from a different domain (though this doesn’t apply if both projects run locally).

Let me know if you encounter any other issues with the setup!

4o

**Bạn đã nói:**

CategoryProject > CatergoryController.cs - facing the error: The type or namespace name 'Category' could not be found (are you missing a using directive or an assembly reference?)

**ChatGPT đã nói:**

ChatGPT

The error occurs because the **Category** model isn’t available in **CategoryProject**. To resolve this, make sure **CategoryProject** has access to the **Category** model by either:

1. **Adding the Category Model Directly in CategoryProject** (if it only exists in CategoryApi).
2. **Sharing the Category Model** between **CategoryApi** and **CategoryProject** by creating a **shared project** or **class library**.

**Option 1: Add the Category Model Directly in CategoryProject**

1. **Create a Models Folder** in **CategoryProject** if it doesn’t already exist.
2. **Create a New Class** in this folder called Category.cs.
3. **Define the Category Model** with properties that match the **CategoryApi** model.  
   For example:  
     
   csharp  
   Sao chép mã  
   namespace CategoryProject.Models  
   {  
    public class Category  
    {  
    public int Id { get; set; }  
    public string Name { get; set; }  
    public string Description { get; set; }  
    }  
   }
4. In **CategoryController.cs**, add the following using directive at the top to access the Category model:  
     
   csharp  
   Sao chép mã  
   using CategoryProject.Models;

**Option 2: Share the Category Model Using a Class Library**

If you want to reuse the **Category** model across multiple projects:

1. **Create a New Class Library** Project:
   * **Run the following command in the root folder of your solution:  
       
     bash  
     Sao chép mã  
     dotnet new classlib -n SharedModels**
   * **This creates a SharedModels project with a basic structure.**
2. **Define the Category Model** in the SharedModels Project:
   * **In the SharedModels project, add a new file called Category.cs with the necessary properties.**
3. **Add a Reference to the Class Library** in Both Projects:
   * **Add the reference in each project by running:  
       
     bash  
     Sao chép mã  
     dotnet add CategoryProject reference ../SharedModels  
     dotnet add CategoryApi reference ../SharedModels**
4. **Update the Using Directive** in **CategoryController.cs**:
   * **In CategoryProject’s CategoryController.cs, add:  
       
     csharp  
     Sao chép mã  
     using SharedModels;**

After following either option, the **CategoryProject** should now recognize the **Category** model, resolving the error. Let me know if you need more guidance on either option!

4o

**Bạn đã nói:**

seem CatergoryApi does not have any Model before

**ChatGPT đã nói:**

ChatGPT

Đã cập nhật bộ nhớ

In that case, let’s add a **Category model** to both **CategoryApi** and **CategoryProject**. This way, both projects can share the same structure.

Here’s how to add the **Category model** to each project:

**Step 1: Add the Category Model to Both Projects**

1. **In CategoryApi**:
   * **Create a Models folder in the CategoryApi project if it doesn’t already exist.**
   * **Add a new file named Category.cs in the Models folder.**
   * **Define the Category model with properties like this:  
       
     csharp  
     Sao chép mã  
     namespace CategoryApi.Models  
     {  
      public class Category  
      {  
      public int Id { get; set; }  
      public string Name { get; set; }  
      public string Description { get; set; }  
      }  
     }**
2. **In CategoryProject**:
   * **Similarly, create a Models folder if it doesn’t already exist.**
   * **Add a new file named Category.cs in the Models folder.**
   * **Copy the same Category class definition:  
       
     csharp  
     Sao chép mã  
     namespace CategoryProject.Models  
     {  
      public class Category  
      {  
      public int Id { get; set; }  
      public string Name { get; set; }  
      public string Description { get; set; }  
      }  
     }**

**Step 2: Update Using Directives**

1. In **CategoryController.cs** of **CategoryProject**, add the following using directive:  
     
   csharp  
   Sao chép mã  
   using CategoryProject.Models;
2. In **CategoryApi**, make sure any code accessing Category (like your controller or API endpoints) has the following directive:  
     
   csharp  
   Sao chép mã  
   using CategoryApi.Models;

**Step 3: Use the Model in API and MVC Projects**

* Now, you can use the **Category** model in **CategoryApi** to define your endpoints and in **CategoryProject** to retrieve data and display it.

This setup should allow the **CategoryProject** to retrieve data from **CategoryApi** and display it, now with a shared **Category** model structure across both projects. Let me know if you need further assistance!

4o

**Bạn đã nói:**

follow the steps, and facing below error when access <http://localhost:5241/Category>: An unhandled exception occurred while processing the request. SocketException: No connection could be made because the target machine actively refused it. System.Net.Sockets.Socket+AwaitableSocketAsyncEventArgs.ThrowException(SocketError error, CancellationToken cancellationToken) HttpRequestException: No connection could be made because the target machine actively refused it. (localhost:5001) System.Net.Http.HttpConnectionPool.ConnectToTcpHostAsync(string host, int port, HttpRequestMessage initialRequest, bool async, CancellationToken cancellationToken)

**ChatGPT đã nói:**

ChatGPT

The error indicates that **CategoryProject** (the MVC project) is trying to connect to **CategoryApi** at localhost:5001, but the connection is being refused. This usually happens if:

1. **CategoryApi is not running**.
2. **The port number** in **CategoryProject** does not match the actual port **CategoryApi** is running on.

Here’s how to resolve this:

**Step 1: Confirm CategoryApi is Running**

1. **Run CategoryApi** by navigating to its folder in the terminal and executing:  
     
   bash  
   Sao chép mã  
   dotnet run
2. Confirm the port **CategoryApi** is running on. If it’s set to localhost:5001, continue; if not, note the actual port.

**Step 2: Check and Update the Port in CategoryProject**

1. Open **CategoryController.cs** in **CategoryProject**.
2. Ensure the **base address** for **CategoryApi** is set to the correct port. If it’s running on a different port, update it in your HttpClient setup:  
     
   csharp  
   Sao chép mã  
   var client = \_httpClientFactory.CreateClient();  
   client.BaseAddress = new Uri("http://localhost:<actual-port>");  
     
   Replace <actual-port> with the port number **CategoryApi** is running on.

**Step 3: Restart Both Projects**

1. Run **CategoryApi** first.
2. Then, run **CategoryProject** and try accessing the Category page again.

Let me know if this resolves the issue or if you see any additional error messages.

4o

**Bạn đã nói:**

another issue: An unhandled exception occurred while processing the request. InvalidOperationException: The model item passed into the ViewDataDictionary is of type 'System.Collections.Generic.List1[CategoryProject.Models.Category]', but this ViewDataDictionary instance requires a model item of type 'System.Collections.Generic.List1[System.String]'. Microsoft.AspNetCore.Mvc.ViewFeatures.ViewDataDictionary.EnsureCompatible(object value)

**ChatGPT đã nói:**

ChatGPT

This error typically occurs when there’s a mismatch between the data type passed from the controller to the view and the type expected in the view.

In this case, it seems that the view is expecting a list of string items (e.g., List<string>) but is receiving a list of Category objects (List<CategoryProject.Models.Category>).

Here's how to resolve this:

**Step 1: Check the View Model Type in the Razor View**

1. Open the **Category view file** (e.g., Index.cshtml) located in Views/Category.
2. At the top of this file, ensure it expects a List<CategoryProject.Models.Category>. Update it if necessary:  
     
   csharp  
   Sao chép mã  
   @model List<CategoryProject.Models.Category>

**Step 2: Ensure Controller is Passing the Correct Model**

1. In **CategoryController.cs**, check that you’re passing a list of Category objects to the view. For example:  
     
   csharp  
   Sao chép mã  
   public async Task<IActionResult> Index()  
   {  
    var client = \_httpClientFactory.CreateClient();  
    client.BaseAddress = new Uri("http://localhost:<actual-port>");  
      
    // Call the API to get the list of categories  
    var response = await client.GetAsync("api/categories");  
      
    if (response.IsSuccessStatusCode)  
    {  
    var jsonData = await response.Content.ReadAsStringAsync();  
    var categories = JsonConvert.DeserializeObject<List<Category>>(jsonData); // Ensure this is a List<Category>  
    return View(categories); // Pass the list of Category objects to the view  
    }  
      
    return View(new List<Category>()); // Return an empty list if the API call fails  
   }  
     
   Make sure the API endpoint returns a list of Category objects that match the structure defined in **Category.cs**.

**Step 3: Update the View to Display Category Properties**

Ensure the **Index.cshtml** view file iterates over the Category model and displays its properties:

html

Sao chép mã

@model List<CategoryProject.Models.Category>

<table>  
 <thead>  
 <tr>  
 <th>ID</th>  
 <th>Name</th>  
 <th>Description</th>  
 </tr>  
 </thead>  
 <tbody>  
 @foreach (var category in Model)  
 {  
 <tr>  
 <td>@category.Id</td>  
 <td>@category.Name</td>  
 <td>@category.Description</td>  
 </tr>  
 }  
 </tbody>  
</table>

This setup should align the data passed from the controller with what the view expects, resolving the type mismatch error. Let me know if this fixes it!