# Appendix A

# Answers to the Test Your Knowledge Questions

This appendix has the answers to the questions in the **Test Your Knowledge** section at the end of each chapter.

## Chapter 1 – Introducing Apps and Services with .NET

Suggested answers to these questions:

1. Why is it good practice to add the following settings to your project files? And when should you not set it?

   ```
   <TreatWarningsAsErrors>true</TreatWarningsAsErrors>
   ```

   **Answer:** The compiler gives warnings about issues that the developer should fix. But compiler warnings can be ignored by default. Enabling the `TreatWarningsAsErrors` option forces the developer to fix the issues so that they can compile and run their project.

   In a .NET project that uses the current gRPC tools, you cannot enable this option because the tools generate types with all-lowercase names. .NET 7 or later gives a warning if you use all lowercase for a type name. You can learn more about this warning wave at the following link: `https://docs.microsoft.com/en-us/dotnet/csharp/language-reference/compiler-messages/warning-waves#cs8981---the-type-name-only-contains-lower-cased-ascii-characters`.

   If you find that you get too many errors after enabling this, you can disable specific warnings by using the `<WarningsNotAsErrors>` element with a comma-separated list of warning codes, as shown in the following markup:

   ```
   <WarningsNotAsErrors>0219,CS8981</WarningsNotAsErrors>
   ```

2. Which service technology requires a minimum HTTP version of 2?

   **Answer**: The gRPC service technology requires a minimum of HTTP/2.

3. In 2010, your organization created a service using .NET Framework and Windows Communication Foundation. What is the best technology to migrate to and why?

   **Answer**: Two options to migrate a WCF service and client are (1) use the Core WCF open-source project, or (2) re-implement the service and client using gRPC. If Core WCF has all the features you need and you want to use minimum effort, then choose Core WCF. If Core WCF does not support all the features you need and you have the time and resources to re-implement the service, then migrate to gRPC.

4. Which code editor or IDE should you install for .NET development?

   **Answer**: Most developers will benefit from installing both Visual Studio 2026 and VS Code and using the best for different project types. More experienced developers often tout the benefits of third-party choices like Rider.

5. What should you beware of when creating Azure resources?

   **Answer**: Many Azure resources have a financial cost. You should delete resources as soon as they are no longer needed to reduce your costs.

# Chapter 1A (online-only section) – What's New in Modern C# and .NET

Suggested answers to these questions:

1. Which type of .NET release is higher quality, STS or LTS?

   **Answer**: Both STS and LTS releases have the same high quality. The only difference is the length of support, 2 years for STS and 3 years for LTS.

2. In new .NET projects, nullable checks are enabled. What are two ways to disable them?

   **Answer**: You can disable nullable checks in the project file by either setting the `<Nullable>` element to `disabled` or by deleting it completely. This is because nullable checks are off by default if the element is not explicitly set to `enabled`.

3. If you define any types in a top-level program, where must they go in the `Program.cs` file?

   **Answer**: At the bottom of the file below any executable statements, or you will see compiler error `CS8803`, as shown at the following link: `https://learn.microsoft.com/en-us/dotnet/csharp/language-reference/compiler-messages/cs8803`.

4. How do you import a class like `Console` so that its static members like `WriteLine` are available in all code files throughout a project?

**Answer**: In the project file, add a `<Using>` element to an `<ItemGroup>` element. The `<Using>` element must have its `Include` attribute set to the full namespace and name of the class, and its `Static` attribute set to `true`. For example, to import the `Console` class so that its static members like `WriteLine` or `ForegroundColor` are available in all code files throughout a project, add the following markup:

```
<ItemGroup>
  <Using Include="System.Console" Static="true" />
</ItemGroup>
```

5. What is the best new C# 14 language feature and why?

   **Answer**: Extension members is the best new C# 14 language feature because it finally implements extension block syntax. This enables extension members like properties and static methods, as well as instance methods. And it retains support for traditional syntax definitions of extension methods for backwards compatibility. This allows cleaner APIs and fluent libraries, better expressiveness, code evolution without breaking changes, all with no new runtime cost.

## Chapter 1B (online-only section) – Benchmarking Performance and Testing

Suggested answers to these questions:

1. What information can you find out about a process?

   **Answer:** The `Process` class has many properties, including `ExitCode`, `ExitTime`, `Id`, `MachineName`, `PagedMemorySize64`, `ProcessorAffinity`, `StandardInput`, `StandardOutput`, `StartTime`, `Threads`, and `TotalProcessorTime`.

2. How accurate is the `Stopwatch` class?

   **Answer**: The `Stopwatch` class can be accurate to within a nanosecond (a billionth of a second), but you shouldn't rely on that.

# Chapter 2 - Building Mobile Apps Using .NET MAUI

Suggested answers to these questions:

1. What are the four categories of .NET MAUI user interface components, and what do they represent?

   **Answer**: The four categories of .NET MAUI user interface components are:

   - Pages: This represents mobile application screens.
   - Layouts: This represents the structure of a combination of the user interface components.
   - Views: This represents a single user interface component.
   - Cells: This represents a single item in a list or table view.

2. What is the benefit of the `Shell` component, and what kinds of UI does it implement?

   **Answer**: The benefit of the `Shell` component is simplified app development by providing standardized navigation and search capabilities. It defines either a tab bar or a flyout for user interface navigation.

3. How can you enable a user to perform an action on a cell in a list view?

   **Answer**: To enable a user to perform an action on a cell in a list view, you can set some context actions, which are menu items that raise an event, as shown in the following markup:

```
<TextCell Text="{Binding CompanyName}"
          Detail="{Binding Location}"
          TextColor="{DynamicResource PrimaryTextColor}"
          DetailColor="{DynamicResource PrimaryTextColor}" >
  <TextCell.ContextActions>
    <MenuItem Clicked="Customer_Phoned" Text="Phone" />
    <MenuItem Clicked="Customer_Deleted" Text="Delete"
              IsDestructive="True" />
  </TextCell.ContextActions>
</TextCell>
```

4. When would you use an `Entry` instead of an `Editor`?

   **Answer**: Use an `Entry` for a single line of text and an `Editor` for multiple lines of text.

5. What is the effect of setting `IsDestructive` to `true` for a menu item in a cell's context actions?

   **Answer**: The menu item is colored red as a warning to the user.

6. You have defined a `Shell` with a content page, but no navigation is shown. Why might this be?

   **Answer**: A shell with only one content page does not show any navigation because there is nothing to navigate to. You must have at least two shell content items.

7. What is the difference between `Margin` and `Padding` for an element like a `Button`?

   **Answer**: The difference between `Margin` and `Padding` for an element like a `Button` is that `Margin` is outside the `Border`, and `Padding` is inside the `Border`.

8. How are event handlers attached to an object using XAML?

   **Answer**: Event handlers are attached to an object using XAML by setting an attribute for the event name to the name of a method in the code-behind class, as shown in the following markup:

```
<Button Clicked="SaveButton_Clicked">
```

9. What do XAML styles do?

   **Answer**: XAML styles enable the setting of one or more properties.

10. Where can you define resources?

    **Answer**: You can define resources in any element depending on where you want to share those resources.

    - To share resources throughout an app, define resources in the `<Application.Resources>` element.
    - To share resources only within a page, define resources in its `<Page.Resources>` element.
    - To share resources only in a single element like a button, define resources in its `<Button.Resources>` element.

# Chapter 3 – Building Desktop Apps Using Avalonia

Suggested answers to these questions:

1. You create a new Avalonia app with the project template, and it runs fine using the example code:

   ```
   <Window xmlns="https://github.com/avaloniaui"
           ...>

     Welcome to Avalonia!

   </Window>
   ```

   But when you add a button under the existing text inside the `<Window>` element, as shown highlighted in the following code, you get a compile error: `Unable to find a setter that allows multiple assignments to the property Content of type Avalonia.Controls:Avalonia.Controls.ContentControl`. What is the problem, and how do you fix it?

   ```
   <Window xmlns="https://github.com/avaloniaui"
           ...>

     Welcome to Avalonia!

     <Button>Test</Button>

   </Window>
   ```

**Answer**: The `<Window>` element is a `ContentControl`, meaning that it has a property named `Content` that can be any *single* element. But the markup contains *two* elements: The "Welcome to Avalonia!" text and the button. You must wrap those two elements together using another control, for example, a `StackPanel` or `Grid`, as shown in the following code:

```
<Window xmlns="https://github.com/avaloniaui"
        ...>

  <StackPanel>
    Welcome to Avalonia!
    <Button>Test</Button>
  </StackPanel>

</Window>
```

2. Why does Avalonia use the file extension `.axaml`?

   **Answer**: Avalonia uses its own XAML dialect, inspired by WPF, UWP, and Xamarin.Forms, but it's implemented from scratch. Avalonia's parser and schema are independent of Microsoft's XAML infrastructure (System.Xaml and the Windows Presentation Foundation). If Avalonia used plain .xaml files, it would cause confusion for IDEs and build tools (which might assume WPF or UWP semantics), code analyzers (that use Microsoft's XML/XAML schemas), and developers switching between frameworks. So the Avalonia team chose `.axaml` to make it explicit that this is Avalonia-specific XAML, not WPF XAML.

3. What are attached properties, and how are they used?

   **Answer**: Attached properties are one of the more subtle but powerful ideas in XAML-based frameworks like WPF, UWP, and Avalonia. They're a design pattern that lets you "attach" extra behavior or metadata to objects that don't natively have those properties. For example:

```
<Button Grid.Row="1" Grid.Column="2" Content="Click me"/>
```

   In the preceding code, `Grid.Row` and `Grid.Column` are attached properties. They belong to the `Grid` class, not to `Button`. But they're attached to the `Button` element to tell the `Grid` how to position it. Even though `Button` doesn't have `Row` or `Column` members, XAML allows you to use `Grid.Row="1"` and `Grid.Column="2"` because the `Grid` defines them as attached properties.

4. How does Avalonia handle the layout and positioning of controls?

   **Answer**: Avalonia's layout system is built around a two-pass layout process (measure and arrange) and a hierarchy of layout containers that determine how child controls are sized and positioned. Avalonia goes through two passes for layout: **Measure Pass** where each control reports how much space it wants based on its content, margins, padding, and layout constraints, and **Arrange Pass** once all desired sizes are known, the parent decides where to place each child within its bounds. Avalonia provides a set of layout panels, each implementing a particular positioning strategy. These are the building blocks of most user interfaces. Layout updates automatically when bound data or properties change.

5. How do you define styles for controls?

   **Answer**: Styles are defined declaratively and are used to change the appearance and behavior of controls without modifying their code. Avalonia's styling system is closer to CSS than XAML in spirit: you can target controls by type, name, class, or more complex selectors, and you can also use dynamic and data-driven styling. A `Style` in Avalonia is a collection of property setters (and optionally triggers) that apply to one or more controls based on matching criteria. Each style defines a selector that determines which controls it applies to, a set of property setters that define visual or behavioral changes, and optional triggers that react to control states or data changes. The following style applies automatically to every `Button` in the scope where it's defined (for example, inside a `Window.Resources` block):

   ```
   <Style Selector="Button">
       <Setter Property="FontSize" Value="18"/>
       <Setter Property="Background" Value="LightBlue"/>
       <Setter Property="CornerRadius" Value="8"/>
   </Style>
   ```

6. What is the purpose of data templates?

   **Answer**: Data templates define how data objects are visually represented in the user interface. When you bind a control (like a `ListBox`, `ComboBox`, or `ContentControl`) to a collection of data objects, a data template provides a visual blueprint to define how to display each item.

7. How can you bind a property in a ViewModel to a control?

   **Answer**: Binding a property from your ViewModel to a control is the fundamental way to connect your UI to your application logic. It's built around the MVVM (Model–View–ViewModel) pattern, where the ViewModel exposes data and commands, and the View (your XAML) displays and interacts with them via data bindings. For example:

   ```
   <TextBox Text="{Binding FirstName}"/>
   ```

   In the preceding markup, Avalonia connects the `Text` property of the `TextBox` to the `FirstName` property of the ViewModel. Changes in either direction can propagate depending on the binding mode.

8. How does Avalonia's rendering pipeline differ from WPF's?

   **Answer:** WPF's rendering model is deeply tied to the Windows graphics stack. It sits atop DirectX (specifically Direct3D) and uses a retained-mode composition engine. Because it depends on DirectX and DWM, WPF is tightly bound to Windows and can't run natively on Linux, macOS, or mobile platforms. Avalonia was designed from the ground up to not depend on Windows graphics APIs. Instead, it uses an abstraction layer that can target multiple backends. Similar to WPF, Avalonia builds a retained-mode visual tree composed of Visual objects, each with geometry, brushes, transforms, and effects. The tree is handed to an `IRenderer`, which decides how to draw it.

Avalonia ships with several implementations, but the default is `SkiaRenderer`. Skia is a high-performance, cross-platform 2D graphics library used by Chrome, Android, and Flutter. Avalonia uses SkiaSharp, the .NET binding to Skia, to rasterize everything: text, shapes, bitmaps, and so on. This is the main reason Avalonia looks the same on Windows, Linux, and macOS.

9. What is the purpose of `INotifyPropertyChanged` and `ReactiveObject` in Avalonia MVVM applications?

   **Answer**: In MVVM (Model–View–ViewModel), your ViewModel exposes properties that the View (your XAML) binds to. For example, if your ViewModel's `Username` property changes after the UI is rendered, the framework needs a way to tell the UI. `INotifyPropertyChanged` is a standard .NET interface that allows an object to notify listeners like Avalonia's binding engine when one of its properties changes. When a property value changes, your ViewModel must raise the `PropertyChanged` event. But implementing `INotifyPropertyChanged` is tedious. `ReactiveObject` is part of the ReactiveUI framework and serves as a base class for reactive ViewModels. It implements `INotifyPropertyChanged` and adds convenient, strongly-typed helpers.

10. How do you implement theming and dynamic styling?

    **Answer**: You theme an Avalonia app by loading a base theme, then drive Light/Dark and any per-subtree overrides with `RequestedThemeVariant`. You put your colors, brushes, and other look-and-feel bits in resource dictionaries, optionally split into `ThemeDictionaries` for Light vs Dark. You reference those values with `{DynamicResource}` so they live-update when the theme changes. For dynamic styling, you flip classes and respond to pseudo-classes and data triggers in styles.

# Chapter 4 – Building Web Apps Using Blazor

Suggested answers to these questions:

1. What is the benefit of the new Blazor Full Stack hosting model in .NET 8 compared to legacy hosting models like Blazor Server?

   **Answer**: Legacy hosting models like Blazor Server limit where components can run. The new Blazor Full Stack hosting model provides complete flexibility and even adds more options. Components can execute on the server and generate static markup, and each individual component can be switched to any of the following: streaming rendering, interactive server-side with live updates of the COM using SignalR, or interactive client-side with WebAssembly. The Blazor WebAssembly hosting model is still useful for standalone apps hosted on a static website.

2. Does Blazor WebAssembly support all features of the latest .NET APIs?

   **Answer**: No. Due to limitations imposed by running inside a web browser, not all features are supported. Use of an unsupported feature will cause a `PlatformNotSupportedException` to be thrown. A browser compatibility analyzer will warn of issues during development time.

3. What is the file extension for Blazor components?

   **Answer:** `.razor`.

4. How do you set the default layout for all Blazor page components?

   **Answer:** In `App.razor`, set the `DefaultLayout` property of the `<RouteView>` element to a Razor expression that sets the type of a layout class, as shown highlighted in the following markup:

```
<Router AppAssembly="@typeof(App).Assembly">
  <Found Context="routeData">
    <RouteView RouteData="@routeData" DefaultLayout="@typeof(MainLayout)"
/>
    ...
  </Found>
  ...
</Router>
```

5. How do you register a route for a Blazor page component?

   **Answer:** At the top of the `.razor` file for a page component, add a `@page` directive with the relative route, as shown in the following code: `@page "my-route"`

6. When would you set the `Match` property of a `<NavLink>` component to `NavLinkMatch.All`?

   **Answer:** When multiple `<NavLink>` component instances could have a partial match on their relative paths; for example, by default, `/customers` would match both `/customers` and `/customers/USA`, so you would want the `<NavLink>` for `/customers` to be set to `NavLinkMatch.All`.

7. You have imported a custom namespace in the `_Imports.razor` file, but when you try to use a class in that namespace in a code-behind file for the Blazor component, the class is not found. Why? How can you fix the issue?

   **Answer:** The `_Imports.razor` file only applies to `.razor` files. If you use code-behind `.cs` files to implement component code, then they must have namespaces imported separately. To fix the issue, use global usings to implicitly import the namespace.

8. What must you do to a property in a component class to have it set to a query string parameter automatically?

   **Answer:** You must decorate the property with the `[Parameter]` and `[SupplyParameterFromQuery]` attributes, as shown in the following code:

```
[Parameter]
[SupplyParameterFromQuery(Name = "country")]
public string? Country { get; set; }
```

9. What is QuickGrid?

   **Answer**: QuickGrid is an open-source basic grid Blazor component from Microsoft.

10. How can a Blazor component access browser features like local storage?

    **Answer**: A Blazor component accesses browser features like local storage by using JavaScript interop, using the interface named `IJSRuntime`. It can dynamically load a JavaScript module file using its `InvokeAsync<IJSObjectReference>` method. The JavaScript file can then interact with all browser features.

# Chapter 5 – Implementing Popular Third-Party Libraries

Suggested answers to these questions:

1. What is the most downloaded third-party NuGet package of all time?

   **Answer**: `Newtonsoft.Json`.

2. What method do you call on the ImageSharp `Image` class to make a change like resizing the image or replacing colors with grayscale?

   **Answer**: `Mutate`.

3. What is a key benefit of using Serilog for logging?

   **Answer**: Serilog can be told to write serialized structured data to the log. The `@` symbol prefixing a parameter tells Serilog to serialize the object passed in, instead of just the result of calling the `ToString` method. Later, that complex object can be queried for improved search and sort capabilities in the logs.

4. What is a Serilog sink?

   **Answer**: A Serilog sink is where you record your logs. For example, there are sinks for writing to the console, to a file, to SQL Server, and to Azure's Application Insights.

5. Should you always use a package like AutoMapper to map between objects?

   **Answer**: No. There is a debate about when AutoMapper should be used that you can read about at the following link: `https://www.anthonysteele.co.uk/AgainstAutoMapper.html`.

6. Which FluentAssertions method should you call to start a fluent assertion on a value?

   **Answer**: `Should`.

7. Which FluentAssertions method should you call to assert that all items in a sequence conform to a condition, like a `string` item must have fewer than six characters?

   **Answer**: `OnlyContain`.

8. Which FluentValidation class should you inherit from to define a custom validator?

   **Answer**: `AbstractValidator<T>`.

9. With FluentValidation, how can you set a rule to only apply in certain conditions?

   **Answer**: Call the `When` method with a lambda expression that returns `true`. You can optionally also call the `Otherwise` method to run if the lambda expression returns `false`.

10. With QuestPDF, which interface must you implement to define a document for a PDF, and what methods of that interface must you implement?

    **Answer**: You must implement the `IDocument` interface and, therefore, implement the `Compose` and `GetMetadata` methods.

# Chapter 6 – Handling Dates, Times, and Internationalization

Suggested answers to these questions:

1. What is the difference between localization, globalization, and internationalization?

   **Answer**:

   - Localization affects the user interface of your application. Localization is controlled by a neutral (language only) or specific (language and region) culture. You provide multiple language versions of text and other values. For example, the label of a text box might be “First name” in English, and “Prénom” in French.
   - Globalization affects the data of your application. Globalization is controlled by a specific (language and region) culture, for example, `en-GB` for British English, or `fr-CA` for Canadian French. The culture must be specific because a decimal value formatted as a currency must know to use Canadian dollars instead of French euros.
   - Internationalization is the combination of localization and globalization.

2. What is the smallest measurement of time available in .NET?

   **Answer**: Nanoseconds.

3. How long is a “tick” in .NET?

   **Answer**: One tick is 100 nanoseconds.

4. In what scenario might you use a `DateOnly` value instead of a `DateTime` value?

   **Answer**: Use a `DateOnly` value when you are mapping to a date column in SQL Server, or when you do not need to store the time or know the time zone.

5. For a time zone, what does its `BaseUtcOffset` property tell you?

   **Answer**: The `BaseUtcOffset` property is a `TimeSpan` that represents the difference between this time zone and the UTC time zone, ignoring any potential Daylight Saving adjustments.

6. How can you get information about the local time zone in which your code executes?

   **Answer**: Read the properties of the `TimeZoneInfo.Local` object.

7. For a `DateTime` value, what does its `Kind` property tell you?

   **Answer**: The `Kind` property is a `DateTimeKind` value that indicates if the `DateTime` value is `Unspecified`, `Utc`, or `Local`.

8. How can you control the current culture for your executing code?

   **Answer**: Set the `CultureInfo.CurrentCulture`, `CultureInfo.CurrentUICulture`, `Thread.CurrentThread.CurrentCulture`, or `Thread.CurrentThread.CurrentUICulture` property to an appropriate `CultureInfo` instance.

9. What is the ISO culture code for Welsh?

   **Answer**: `cy-GB`.

10. How do localization resource file fallbacks work?

    **Answer**: Resource files are XML files with the `.resx` extension. The filename includes a culture code, for example, `PacktResources.en-GB.resx` or `PacktResources.da-DK.resx`. The automatic culture fallback search path for resources goes from a specific culture (language and region) to a neutral culture (language only) to an invariant culture. If the current thread culture is `en-AU` (Australian English), then it will search for the resource file in the following order:

    - Australian English: `PacktResources.en-AU.resx`
    - Neutral English: `PacktResources.en.resx`
    - Invariant: `PacktResources.resx`

# Chapter 7 – Managing Relational Data Using SQL

Suggested answers to these questions:

1. Which NuGet package should you reference in a .NET project to get the best performance when working with data in SQL Server?

   **Answer**: `Microsoft.Data.SqlClient`.

2. What is the safest way to define a database connection string for SQL Server?

   **Answer**: Create an instance of `SqlConnectionStringBuilder`, set its properties like `DataSource`, `InitialCatalog`, `UserID`, and `Password`, and then read its `ConnectionString` property.

3. What must T-SQL parameters and variables be prefixed with?

   **Answer**: T-SQL parameters and variables must be prefixed with the `@` character.

4. What must you do before reading an output parameter of a command executed using `ExecuteReader`?

   **Answer**: You must close the data reader before reading an output parameter.

5. What type does Dapper add its extension methods to?

   **Answer**: Dapper adds its extension methods to any type that implements `IDbConnection`.

6. What are the two most commonly used extension methods provided by Dapper?

   **Answer**: The `Query<T>` extension method and the `Execute` extension method.

# Chapter 8 – Building Entity Models Using EF Core

Suggested answers to these questions:

1. What can the `dotnet-ef` tool be used for?

   **Answer**: The `dotnet-ef` tool can perform design-time development tasks, like creating or applying migrations and generating a model from an existing database.

2. What type would you use for the property that represents a table, for example, the `Products` property of a data context?

   **Answer**: `DbSet<T>`, where `T` is the entity model type in the table, for example, `Product`.

3. What type would you use for the property that represents a one-to-many relationship, for example, the `Products` property of a `Category` entity?

   **Answer**: `ICollection<T>`, where `T` is the entity model type in the related table, for example, `Product`.

4. What is the EF Core convention for primary keys?

   **Answer**: The property named `ID` or `Id` or `ClassNameID` or `ClassNameId` is assumed to be the primary key. If the type of that property is any of the following, then the property is also marked as being an `IDENTITY` column: `tinyint`, `smallint`, `int`, `bigint`, or `guid`.

5. Why might you choose the Fluent API in preference to annotation attributes?

   **Answer**: You might choose Fluent API in preference to annotation attributes when you want to keep your entity classes free from extraneous code that is not needed in all scenarios. For example, when creating a .NET Standard 2.0 class library for entity classes, you might want to only use validation attributes so that the metadata can be read by Entity Framework Core, and by technologies like ASP.NET Core model binding validation and .NET MAUI desktop and mobile apps. However, you might want to use Fluent API to define Entity Framework Core-specific functionality, like mapping to a different table or column name.

6. Why might you implement the `IMaterializationInterceptor` interface in an entity type?

   **Answer**: EF Core interceptors enable interception, modification, and/or suppression of EF Core operations. The `IMaterializationInterceptor` interface allows interception before and after an entity instance is created, and before and after properties of that instance are initialized. This enables setting properties or calling methods needed for validation, computed values, or flags, using a factory to create instances, and creating a different entity instance than EF would normally create, such as an instance from a cache.

# Chapter 9 – Building a Custom LLM-based Chat Service

Suggested answers to these questions:

1. What do you need to call an OpenAI cloud-based LLM?

   **Answer**: You need an API key to call an OpenAI cloud-based LLM. You can get one from the following link: `https://platform.openai.com/settings/organization/api-keys`.

2. What is Microsoft Agent Framework?

   **Answer**: Microsoft Agent Framework is an open-source SDK from Microsoft that lets you easily build agents that can call your existing code. You can use Agent Framework with models from OpenAI, Azure OpenAI, Hugging Face, and others. By combining your existing C# and .NET projects with these models, you can build agents that answer questions and automate processes.

3. What are the benefits of using the AI Chat Web App project template?

   **Answer**: The benefits of using the AI Chat Web App project template are that it already references the Agent Framework packages, it implements a basic chat interface, it supports any model (either local or cloud), and it has the built-in capability of ingesting any PDF and Markdown files that you add to the project.

4. What does the `AIFunctionFactory.Create` method do?

   **Answer**: It allows the general LLM capabilities to be extended with custom functions. The sequence of steps for function calling is as follows:

   - The user sends a message along with a set of functions mapped to .NET methods.
   - The LLM can choose to call one or more functions as well as use its standard model.
   - The application executes the method and sends the method output to the LLM.
   - The LLM uses the method output as additional context to generate an appropriate response.

5. Why is it important to change the default system prompt when using custom functions?

   **Answer**: It is important to change the default system prompt when using custom functions because it instructs the LLM when to use each of its tools aka custom functions.

6. What is Hugging Face?

   **Answer**: Hugging Face is a company known for its innovations in the field of artificial intelligence, particularly in natural language processing (NLP). It has become highly popular for its open-source contributions and its role in developing and maintaining the transformers library, which provides state-of-the-art machine learning models for a variety of NLP tasks. Hugging Face's transformers library is widely used in the AI community. It offers pre-trained models that can be used for tasks like text classification, information extraction, question answering, summarization, translation, and more. Alongside the transformers library, Hugging Face operates a model hub that hosts thousands of pre-trained models contributed by the community. These models can be easily downloaded and used in various NLP projects.

# Chapter 10 – Building and Securing Minimal API Web Services

Suggested answers to these questions:

1. List six method names that can be specified in an HTTP request.

   **Answer**: `GET`, `HEAD`, `POST`, `PUT`, `PATCH`, and `DELETE`. Others include `TRACE`, `OPTIONS`, and `CONNECT`.

2. List six status codes and their descriptions that can be returned in an HTTP response.

   **Answer**: `200` OK, `201` Created, `301` Moved Permanently, `400` Bad Request, `404` Not Found (missing resource), and `500` Internal Server Error. Others include `101` Switching Protocols (e.g., from HTTP to WebSocket), `202` Accepted, `204` No Content, `304` Not Modified, `401` Unauthorized, `403` Forbidden, `406` Not Acceptable (for example, requesting a response format that is not supported by a website), and `503` Service Unavailable.

3. How is the ASP.NET Core Minimal API service technology different from the ASP.NET Core Web API service technology?

   **Answer**: An ASP.NET Core Web API service requires a controller class to define the endpoints. An ASP.NET Core Minimal API service does not need a controller. A Minimal API service can therefore be simpler and implemented in fewer statements, hence the name.

4. With the ASP.NET Core Minimal APIs service technology, how do you map an HTTP `PUT` request to `api/customers` to a lambda statement block?

   **Answer**: Call the `MapPut` method, and pass a route pattern and a lambda statement block, as shown in the following code:

   ```
   app.MapPut("api/customers", () =>
     {
       // Do something.
     });
   ```

5. With the ASP.NET Core Minimal API service technology, how do you map a method or lambda parameter to a value in a route, query string, or the body of the request?

   **Answer**: Decorate the parameter with `[FromRoute]`, `[FromQuery]`, or `[FromBody]`.

6. Does enabling CORS increase security for a web service?

   **Answer**: No. Enabling CORS deliberately weakens security for a web service to allow functionality to work by relaxing the same origin policy of web browsers in specific scenarios. Enabling CORS has no effect on non-web browser clients.

7. You have added statements to `Program.cs` to enable HTTP logging, but HTTP requests and responses are not being logged. What is the most likely reason, and how can you fix it?

   **Answer**: If no HTTP activity is being logged, then the most likely reason is that the log level is set to `Warning` (2) or `Error` (1). The log level for HTTP logging must be `Information` (3) or higher. In the `appsettings.json` file for your environment, make sure that HTTP logging is at a suitable level, as shown highlighted in the following configuration:

   ```
   {
     "Logging": {
       "LogLevel": {
         "Default": "Information",
         "Microsoft.AspNetCore": "Warning",
         "Microsoft.AspNetCore.HttpLogging": "Information"
       }
     }
   }
   ```

8. How do you limit the rate of requests for a specific client using the `AspNetCoreRateLimit` package?

   **Answer**: First, configure the name of the HTTP request header that the client must send, as shown in the following configuration:

   ```
   "ClientRateLimiting": {
     "ClientIdHeader": "X-Client-Id",
   ```

   Second, provide the client with a unique ID that they must set that header to in their requests, for example, a GUID value for their account. Third, configure rules for that client ID, as shown in the following configuration:

   ```
   "ClientRateLimitPolicies": {
     "ClientRules": [
       {
         "ClientId": "abc123-...",
         "Rules": [
   ```

9. How do you limit the rate of requests for a specific endpoint using the `Microsoft.AspNetCore.RateLimiting` package?

   **Answer**: First, create a `RateLimiterOptions` object, configured with the limits that you want, and a name for the policy, and then pass it to a call to the `UseRateLimiter` method. Second, call the `RequireRateLimiting` method on the endpoint and pass in the name of the policy.

10. What does JWT mean?

    **Answer:** **JSON Web Token** (**JWT**) is an open standard that defines a compact and self-contained way to transmit information as a JSON object. It is commonly used for authorization and other small data exchanges.

# Chapter 11 – Caching, Queuing, and Resilient Background Services

Suggested answers to these questions:

1. How much longer does it take to read 1 MB of data from an SSD compared to memory?

   **Answer:** It takes about four times longer to read 1 MB of data from an SSD compared to memory.

2. What is the difference between absolute and sliding expirations?

   **Answer:** Absolute is a fixed date/time. Sliding is a time span that resets if the item is read, so it can potentially last forever.

3. What unit of measurement is used by `Size` for the in-memory cache?

   **Answer:** The unit of measurement is arbitrary.

4. You have written the following statement to get information about in-memory caching, but `stats` is `null`. What must you do to fix this issue?

   ```
   MemoryCacheStatistics? stats = _memoryCache.GetCurrentStatistics();
   ```

   **Answer:** You must set `TrackStatistics` to `true` in the `MemoryCacheOptions` object when registering the `MemoryCache` as a dependency service, as shown in the following code:

   ```
   builder.Services.AddSingleton<IMemoryCache>(new MemoryCache(
     new MemoryCacheOptions
     {
       TrackStatistics = true,
       // ...set other options.
     }));
   ```

5. What data types can be stored in (a) an in-memory cache, and (b) a distributed cache?

   **Answer:** Any data type can be stored in an in-memory cache. Only byte arrays can be stored in a distributed cache.

6. What are the differences between the Retry and Circuit Breaker patterns?

   **Answer:** The **Retry** pattern enables clients to automatically retry a failed action with the expectation that the fault will succeed if retried after a short delay. The **Circuit Breaker** pattern prevents calls when a threshold of faults is reached. In effect, it is a way for a service to detect if a fault is *not* transient, or not transient enough to keep retrying.

7. When using the RabbitMQ default direct exchange, what must the routing key be for a queue named `product`?

   **Answer**: The routing key must match the queue name, so it must be `product`.

8. What is the difference between a fanout and a topic exchange?

   **Answer**: A Fanout exchange delivers messages to all queues that are bound to it and the routing key is ignored. A Topic exchange delivers messages based on a routing key and criteria defined in the binding between the exchange and a queue.

9. What port does RabbitMQ listen on by default?

   **Answer**: `5672`.

10. When inheriting from the `BackgroundService` class, what method must you override that is called automatically by the host to run your service?

    **Answer**: `ExecuteAsync`.

# Chapter 12 – Broadcasting Real-Time Communication Using SignalR

Suggested answers to these questions:

1. What transport does SignalR use, and which is the default?

   **Answer**: SignalR prefers to use WebSockets as its transport, then it will fall back to Server-Side Events, and finally, it will use Long Polling if neither of the others is supported by the client and server.

2. What is a good practice for RPC method signature design?

   **Answer**: A good practice for RPC method signature design is to define a single parameter using a complex type. This allows additional properties to be added to the type in the future without breaking the contract between the client and server.

3. What tool can you use to download the SignalR JavaScript library?

   **Answer**: At the command line or terminal, you can use the Library Manager CLI tool. Visual Studio 2022 includes a graphical tool; right-click a web project, and then choose **Add** | **Client Side Libraries**.

4. What happens if you send a SignalR message to a client with a connection ID that does not exist?

   **Answer**: Nothing.

5. What are the benefits of separating a SignalR service from other ASP.NET Core components?

   **Answer**: Once you separate the SignalR hosting, you can take advantage of Azure SignalR Service. This offers global reach and a world-class data center and network, and scales up to millions of connections, while meeting SLAs like providing compliance and high security.

# Chapter 13 – Combining Data Sources Using GraphQL

Suggested answers to these questions:

1. What transport protocol does a GraphQL service use?

   **Answer**: GraphQL can use HTTP or others, like WebSocket.

2. What media type does GraphQL use for its queries?

   **Answer**: GraphQL can use its own media type, `application/graphql`.

3. How can you parameterize GraphQL queries?

   **Answer**: In the `query` definition, you define parameter names prefixed with `$` with a data type, and then reference them in the query, as shown in the following query:

   ```
   query getOrdersByDateAndCountry($country: String, $orderDate: String) {
     order(orderDate: $orderDate {
       orderId
       orderDate
       customer(country: $country) {
         companyName
         country
       }
     }
   }
   ```

4. What are the benefits of using Strawberry Shake over a regular HTTP client for GraphQL queries?

   **Answer**: Strawberry Shake includes a tool to generate strongly typed proxies that call a GraphQL service and can be more easily used by a .NET client, instead of manually making requests to the service.

5. How might you insert a new product into the Northwind database?

   **Answer**: You would define a mutation named `addProduct`, consisting of a type to represent the input, a type to represent the payload, and a class with a method to perform the insertion. Then, you could submit a mutation, as shown in the following GraphQL document:

   ```
   mutation AddProduct {
     addProduct(
       input: {
         productName: "Tasty Burgers"
         supplierId: 1
         categoryId: 2
         quantityPerUnit: "6 per box"
         unitPrice: 40
         unitsInStock: 0
   ```

```
        unitsOnOrder: 0
        reorderLevel: 0
        discontinued: false
      }
    )
    {
      product {
        productId
        productName
      }
    }
  }
```

# Chapter 14 – Building Efficient Microservices Using gRPC

Suggested answers to these questions:

1. What are three benefits of gRPC that make it a good choice for implementing services?

   **Answer**: Three benefits of gRPC that make it a good choice for implementing services are (1) its Protobuf binary serialization that minimizes network usage, (2) its requirement of HTTP/2 that provides significant performance benefits, and (3) its support by almost all languages and platforms.

2. How are contracts defined in gRPC?

   **Answer**: In gRPC, contracts are defined using `.proto` files.

3. Which of the following .NET types require extensions to be imported: `int`, `double`, or `DateTime`?

   **Answer**: `DateTime` is not supported by default and so requires an extension to be imported.

4. Why should you set a deadline when calling a gRPC method?

   **Answer**: Setting a deadline for a gRPC call is recommended practice because it controls the upper limit on how long a gRPC call can run for. It prevents gRPC services from potentially consuming too many server resources.

5. Why should you be cautious if implementing your own custom type to handle `decimal` values?

   **Answer**: There is no standard for handling `decimal` values, so any clients will need to know how you have handled them so that they can do so in the same way. This is difficult to guarantee.

6. What do you have to do to use date and time values in a gRPC message?

   **Answer**: You must use well-known type extensions: `google.protobuf.Timestamp` and `google.protobuf.Duration`.

7. What are some scenarios when you might implement an interceptor?

   **Answer**: Interceptors are a way to perform additional processing during requests and responses in the client or service. For example, they can be used for logging, monitoring, and validation.

8. What are the benefits of enabling gRPC JSON transcoding to a gRPC service hosted in ASP.NET Core?

   **Answer**: gRPC has some strict requirements that not all clients can support. Making HTTP requests and passing JSON objects is supported by all clients. Enabling gRPC JSON transcoding for a gRPC service therefore enables the best of both worlds.

# Chapter O1 (online-only section) – Managing NoSQL Data Using Azure Cosmos DB

Suggested answers to these questions:

1. What are the APIs supported by Azure Cosmos DB?

   **Answer**: The APIs supported by Azure Cosmos DB are the Core (SQL) API, Gremlin (Graph) API, MongoDb API, Azure Table Storage API, and Cassandra API.

2. At what level do you select the API: account, database, container, or partition?

   **Answer**: You select the API at the account level, and that applies to every database, container, or partition in that account.

3. What does *embed* mean regarding data modeling with Cosmos DB?

   **Answer**: Embedding means storing data, like the category and supplier information for a product, within the product data, even if that means duplicating that data across many products. It is like the concept of denormalization in a relational database.

4. What is the unit of measurement for throughput for Cosmos DB, and what does 1 unit represent?

   **Answer**: Throughput is measured as **request units per second** (**RU/s**). A single **request unit** (**RU**) is about the cost of performing a `GET` request for a 1 KB document, using its unique identifier.

5. What package should you reference to programmatically work with Cosmos DB resources?

   **Answer**: `Microsoft.Azure.Cosmos`.

6. What language do you use to write Cosmos DB Core (SQL) API user-defined functions and stored procedures?

   **Answer**: JavaScript.

7. What is the difference between a vertex and an edge in a graph database?

   **Answer**: A vertex represents a noun like a customer or product. An edge represents a relationship between two vertices, like bought or viewed.

8. What package should you reference to programmatically execute Gremlin scripts?

   **Answer**: `Gremlin.net`.

9. What Gremlin command returns all vertices that have a label of `product` and have a property named `unitsInStock`, with more than 10 units in stock?

   **Answer**: `g.V().has('product', 'unitsInStock', gt(10))`

10. Why do some edges have weights?

    **Answer**: A common property for an edge is a `weight` that could be used to give importance to the relationship, or the cost of traversing the edge in time, money, or distance.

# Chapter 02 (online-only section) – Building Serverless Nanoservices Using Azure Functions

Suggested answers to these questions:

1. What is the difference between the in-process and isolated hosting models for Azure Functions?

   **Answer**: The in-process hosting model requires your Azure Function to be loaded alongside other code and to target a predefined version of an LTS release, like .NET Core 3.1 or .NET 6. The isolated hosting model allows your Azure Function to load in its own process, and it can use any version of .NET that you choose.

2. What attribute do you use to cause a function to trigger when a message arrives in a queue?

   **Answer**: Decorate the `Run` method parameter that represents the received message with the `[QueueTrigger]` attribute, and specify the queue name, as shown highlighted in the following code:

```
public static async Task Run(
  [QueueTrigger("checksQueue")] QueueMessage message,
  ILogger log)
{
```

3. What attribute do you use to make a queue available to send messages to?

   **Answer**: Decorate the `Run` method parameter that represents the queue with the `[Queue]` attribute, and specify the queue name, as shown highlighted in the following code:

```
public static async Task<IActionResult> Run(
  [HttpTrigger(AuthorizationLevel.Anonymous,
    "get", "post", Route = null)] HttpRequest req,
  [Queue("checksQueue")] ICollector<string> collector,
  ILogger log)
{
  ...
  collector.Add("This is the message."); // Send message to queue.
  ...
```

4. What schedule does the following NCRONTAB expression define?

   ```
   0 0 */6 * 6 6
   ```

   **Answer**: Assuming the occurrences are during the year 2023, the NCRONTAB expression defines that they occur on Saturdays (the last 6 digit) during June (the preceding 6 digit) in 2023, on any day of the month, at zero minutes and seconds (the first and second 0 digits) of every 6 hours (`*/6`). For example:

   ```
   Sat, 03 Jun 2023 00:00:00
   Sat, 03 Jun 2023 06:00:00
   Sat, 03 Jun 2023 12:00:00
   Sat, 03 Jun 2023 18:00:00
   Sat, 10 Jun 2023 00:00:00
   Sat, 10 Jun 2023 06:00:00
   Sat, 10 Jun 2023 12:00:00
   Sat, 10 Jun 2023 18:00:00
   Sat, 17 Jun 2023 00:00:00
   Sat, 17 Jun 2023 06:00:00
   Sat, 17 Jun 2023 12:00:00
   Sat, 17 Jun 2023 18:00:00
   Sat, 24 Jun 2023 00:00:00
   Sat, 24 Jun 2023 06:00:00
   Sat, 24 Jun 2023 12:00:00
   Sat, 24 Jun 2023 18:00:00
   ```

5. How can you configure a dependency service for use in a function?

   **Answer**: To register a dependency service for use in a function, create a class that inherits from the `FunctionsStartup` class and override its `Configure` method. Add the `[FunctionsStartup]` assembly attribute to specify the class name registered for startup. Add services to the `IFunctionsHostBuilder` instance passed to the method.