Implementing Design Patterns in C# 11 and .NET 7

2nd Edition

Learn how to design and develop robust and scalable applications using design patterns

Alexandre F. Malavasi Cardoso

www.bpbonline.com

First published: 2021

Second published: 2024

Published by BPB Online
WeWork
119 Marylebone Road
London NW1 5PU

UK | UAE | INDIA | SINGAPORE

ISBN 978-93-55517-333

www.bpbonline.com

Dedicated to

My beloved wife:
Paula
and
My daughter ***Myla***

About the Author

Alexandre F. Malavasi Cardoso boasts over 16 years in software development, taking pivotal roles as a technical leader and engineer. He has spearheaded major projects using Microsoft Technologies for prominent companies across South America, Europe, and the U.S. Presently, he serves as the Head of Engineering at Propylon and as a Technical Advisor for Marelo. Alexandre's academic accolades include a postgraduate degree in Business and Systems Analysis and two master's degrees emphasizing Software Engineering with Agile Methods. He has earned multiple Microsoft certifications in Azure and Web Development. An active contributor to the global tech community, Alexandre often speaks at international IT conferences, authors technical articles, and has thrice been recognized by Microsoft as a Most Valuable Professional (MVP).

About the Reviewer

Based in Rabat, Morocco, **Nabil Tadili** stands out as a renowned Lead Software Architect with deep expertise in Full Stack development, predominantly in the realms of .NET and Angular. With an unwavering dedication to pioneering innovations and adhering to industry best practices, he's delved profoundly into these platforms, ensuring he's attuned to their evolving nuances. In addition, Nabil showcases mastery in Node, Svelte, NextJs, Python, and a plethora of databases and demonstrates an exceptional command over CI/CD methodologies, representing a comprehensive embodiment of contemporary software engineering.

At Orange Business, Nabil's key role in driving the shift to a Microservices architecture highlights his forward-thinking approach to technological progression. While actively delving into Machine Learning at the Georgia Institute of Technology, he further accentuates his multifaceted expertise by refining business workflows and leading groundbreaking digital projects. His freelance journey, punctuated by over 150 accomplished projects and a stellar 98%+ client satisfaction rate, underscores his flexibility and unwavering commitment.

Ever driven by the motto to challenge one's limits, Nabil is an ardent advocate for teamwork, innovation, and continuous learning. His passion extends to artificial intelligence, quantum computing, and science.

Acknowledgement

My heartfelt appreciation goes to my family and friends for their steadfast support during this book's creation, with a special nod to my wife Paula and daughter Myla. I extend my gratitude to BPB Publications, whose expertise was pivotal in finalizing this work—a journey enriched by the collaboration of reviewers, technical mavens, and editors.

I must also recognize the invaluable insights of my colleagues and coworkers from years in the tech sector. Their teachings and feedback have been indispensable. Lastly, to every reader who resonated with my book: your enthusiasm and encouragement have truly anchored this endeavor.

Preface

In the dynamic world of software development, the mastery of Design Patterns and the intricacies of the object-oriented programming paradigm can set one apart. This book aims to be your guide through these complexities, specifically tailored for the modern version of C# language and the .NET platform. Our journey spans from the foundational SOLID principles, traces the rich history of the .NET platform, and ventures into the core of Design Patterns, all illustrated through tangible, real-world examples and an intuitive, step-by-step methodology.

Our opening chapters lay the groundwork, elucidating the fundamental concepts of C# and .NET. As we traverse through SOLID principles and the essence of object-oriented programming, we set the stage for the immersive experiences that follow. The book's heart revolves around hands-on examples that emphasize best software development practices, all while setting the stage for the profound Design Patterns which dominate the contemporary market.

Upon turning the last page, you, the developer, will find yourself equipped with a holistic understanding of C# and the .NET platform. More than just knowledge, you'll possess the practical wisdom to implement best practices in real-world scenarios and a versatile toolbox of Design Patterns to tackle the myriad challenges of software development. Welcome to a transformative learning experience.

Chapter 1: C# Fundamentals – Diving deep into the C# landscape, this chapter covers the essential components of the language—from its syntax to control structures. Readers will become well-versed with data types, operators, and the nuances of C# development, setting a solid foundation for advanced topics.

Chapter 2: .NET Fundamentals – This section unveils the architecture of the .NET platform, detailing its core components, runtime environment, and the Common Language Runtime (CLR). Readers will learn about .NET's diverse class libraries, offering a comprehensive view of this powerful platform.

Chapter 3: Basic Concepts of Object-Oriented Programming in C# – Emphasizing the pillars of OOP—encapsulation, inheritance, polymorphism, and abstraction—this chapter offers a C#-centric view. Readers will grasp the power of object-oriented design, learning about classes, objects, interfaces, and more.

Chapter 4: SOLID Principles in C# – This chapter demystifies the SOLID principles, laying the foundation for designing robust and maintainable C# applications. Readers will understand the importance of each principle, from Single Responsibility to Dependency Inversion, and how they guide software craftsmanship.

Chapter 5: Introduction to Design Patterns – A comprehensive introduction awaits as readers embark on a journey to understand the rationale, significance, and application of design patterns, preparing them to tackle software design challenges with confidence.

Chapter 6: Singleton Pattern in .NET Applications – Exploring the Singleton pattern, this chapter showcases its role in ensuring a class has a single instance, guiding readers through its implementation nuances and real-world applications within the .NET environment.

Chapter 7: Abstract Factory Pattern with Blazor – Diving into the Abstract Factory Pattern, readers will comprehend its importance in abstracting the creation of related families of objects. The chapter highlights its synergy with the Blazor framework, offering practical insights.

Chapter 8: Prototype Pattern with ASP.NET Razor – This chapter delves into the Prototype Pattern, emphasizing its utility in creating object copies within the context of ASP.NET Razor. Through examples, readers will learn about object cloning and its benefits.

Chapter 9: Factory Method Pattern Using New Features on C# 11 – With a focus on the Factory Method Pattern, readers will explore its applicability in object creation. Leveraging C# 11's novel features, the chapter provides an enriched perspective on this creational pattern.

Chapter 10: Adapter Pattern with Entity Framework Core – This chapter illuminates the Adapter Pattern, illustrating its pivotal role in harmonizing incompatible interfaces. Using Entity Framework Core as a backdrop, readers will discover the pattern's application in real-world scenarios.

Chapter 11: Composite Pattern with ASP.NET MVC – Detailing the Composite Pattern, this section sheds light on managing hierarchies of objects. Within the ASP.NET MVC framework context, readers will grasp its essence and benefits, especially in UI rendering.

Chapter 12: Proxy Pattern with GRPC – The Proxy Pattern takes the spotlight here. Readers will learn its significance in controlling access to objects. The integration with the GRPC framework offers a fresh perspective on managing object communications.

Chapter 13: Command Pattern Using MediatR – This chapter dissects the Command Pattern, emphasizing its utility in encapsulating request-response mechanisms. With MediatR, readers will experience its potency in decoupling classes and managing operations.

Chapter 14: Strategy Pattern Using Azure C# and Azure Functions – Exploring the Strategy Pattern, this chapter showcases its role in defining interchangeable algorithm families. With C# and Azure Functions as the foundation, readers will learn to dynamically select algorithmic implementations in cloud-based solutions.

Chapter 15: Observer Pattern – Concluding with the Observer Pattern, readers will delve into the intricacies of maintaining consistency between objects. The chapter highlights the importance of real-time data synchronization and how the pattern ensures efficient object interactivity and communication.

Code Bundle and Coloured Images

Please follow the link to download the
Code Bundle and the ***Coloured Images*** of the book:

https://rebrand.ly/b4af92

The code bundle for the book is also hosted on GitHub at **https://github.com/bpbpublications/Implementing-Design-Patterns-in-C-Sharp-11-and-.NET-7-2nd-Edition**. In case there's an update to the code, it will be updated on the existing GitHub repository.

We have code bundles from our rich catalogue of books and videos available at **https://github.com/bpbpublications**. Check them out!

Errata

We take immense pride in our work at BPB Publications and follow best practices to ensure the accuracy of our content to provide with an indulging reading experience to our subscribers. Our readers are our mirrors, and we use their inputs to reflect and improve upon human errors, if any, that may have occurred during the publishing processes involved. To let us maintain the quality and help us reach out to any readers who might be having difficulties due to any unforeseen errors, please write to us at :

errata@bpbonline.com

Your support, suggestions and feedbacks are highly appreciated by the BPB Publications' Family.

Piracy

If you come across any illegal copies of our works in any form on the internet, we would be grateful if you would provide us with the location address or website name. Please contact us at **business@bpbonline.com** with a link to the material.

If you are interested in becoming an author

If there is a topic that you have expertise in, and you are interested in either writing or contributing to a book, please visit **www.bpbonline.com**. We have worked with thousands of developers and tech professionals, just like you, to help them share their insights with the global tech community. You can make a general application, apply for a specific hot topic that we are recruiting an author for, or submit your own idea.

Reviews

Please leave a review. Once you have read and used this book, why not leave a review on the site that you purchased it from? Potential readers can then see and use your unbiased opinion to make purchase decisions. We at BPB can understand what you think about our products, and our authors can see your feedback on their book. Thank you!

For more information about BPB, please visit **www.bpbonline.com**.

Join our book's Discord space

Join the book's Discord Workspace for Latest updates, Offers, Tech happenings around the world, New Release and Sessions with the Authors:

https://discord.bpbonline.com

Table of Contents

Chapter 1
C# Fundamentals

Introduction

With this chapter, we are starting our journey into design patterns using C# 11.0 and .NET 7, walking through the basic concepts of programming in C#, giving you familiarity with the language, its main operations, and instructions that will help you understand what you need to learn to make progress in the next sections of this book, such as object-oriented programming, design patterns, and .NET platform.

You will learn in this chapter how to create and work with variables, operators, and logical and conditional statements. Additionally, you will have the opportunity to have practical experience in implementing basic programs in C#. Further, you will be able to try the new Debugging Experience provided by Visual Studio 2022.

Getting familiar with the basic concepts of C# will allow you to understand how to apply the complex design patterns in the further chapter in this book. It will help you to build enterprise projects using the .NET platform.

Structure

In this chapter, we will discuss the following topics:

- Visual Studio 2022 and Visual Studio Code Installation Instructions

- Introduction to Visual Studio 2022 and Visual Studio Code
- Basic operations in C#
- Object Types in C#
- Loops and iterations in C#
- Error handling in C#
- New features introduced in C# 10 and C# 11

Objectives

In this unit, you'll learn to set up Visual Studio IDE and Visual Studio Code, create C# apps, grasp basic language operations, and employ new features from C# 10 and 11. These skills will provide a strong foundation for proficient C# programming and development.

Tools and environment setup

To get started with software development in .NET 7 and C# 11, you must install the latest Visual Studio 2022 version, a complete **Integrated Development Environment** (**IDE**) for creating, compiling, and building your .NET projects. Visual Studio is available for Windows and macOS, both available in the Community Edition for studying purposes. The tool can be downloaded from the official Visual Studio website.

Furthermore, Microsoft has provided the Visual Studio Code, a free, open-source alternative light version of code editor for .NET and C# applications, which is available not only for Windows and macOS but also for various Linux distributions. Considering this editor is an open-source, extensible project, technical communities, IT professionals, and companies around the globe have created numerous extensions for different languages apart from C# itself. Therefore, it is a suitable tool for creating cross-platform applications without any compatibility concerns. Visual Studio Code can be downloaded on the official website for free.

Installing Visual Studio 2022

After downloading Visual Studio 2022 from the official website, you must take the following steps for the installation:

1. Make sure your user has the necessary permissions to install the software on the operating system. Double-click the downloaded executable file.
2. Choose the desired workloads to be installed and set up with Visual Studio, including the additional native project templates from the .NET platform. For the examples of this book, the following workloads must be installed, as seen in *Figure 1.1*:
   - ASP.NET and web development
   - Azure Development
   - Node.js development
   - Mobile development with .NET
   - .NET desktop development
   - Universal Windows Platform development

   After making this step in the installation, the workload list will be shown as the following result in *Figure 1.1*:

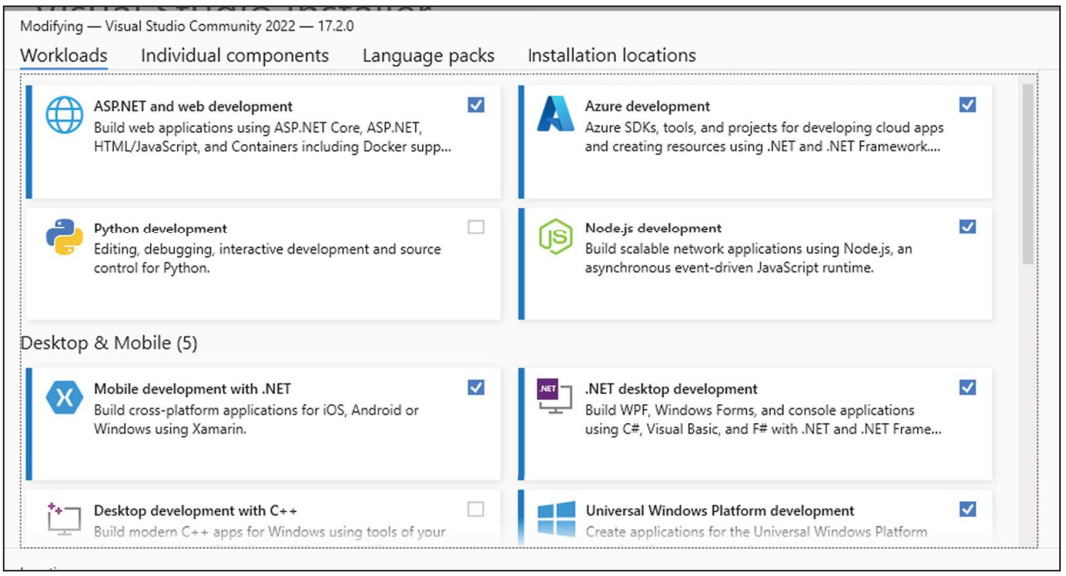

***Figure 1.1**: Visual Studio 2022 workloads*

3. After choosing the necessary workloads, click on the Install option.
4. Usually, Visual Studio is configured to use the same language and region settings as the operating system. If you want to setup a different one, you

can do that in the **Language Packs** option, where other ones will be available. Visual Studio was configured to the English language for the examples in this book by default. After finishing the installation, you will be already able to create .NET 7 solutions using all the project type available in Visual Studio that belongs to the workloads specified during installation.

Once Visual Studio 2022 is installed, you do not need to install the latest stable version of the .NET SDK, as it is already a part of the Visual Studio installation. By default, Visual Studio updates and follows the newest features introduced into .NET, such as library updates, minor and major changes, and new project templates. However, any new library updates after the initial installation will not automatically apply the upgrades to your existing projects. Each project targets a specific .NET version, and Visual Studio updates and modifies just the IDE, not the existing projects' configurations.

Installing Visual Studio Code

Visual Studio Code is a cross-platform alternative to Visual Studio IDE, and it is a good option if you want a free, lightweight cross-platform editor for .NET projects. Many companies, technical communities, and individual contributors have provided extensions that allow us to work with many different programming languages in Visual Studio Code. This light open-source IDE has become one of the most popular code editors used in the market. Furthermore, the editor is available for multiple operating systems, such as macOS,Windows, and multiple Linux distributions.

After downloading the executable from the official Visual Studio website, you must take the following steps to complete the installation:

1. Double-click the downloaded executable file. Ensure your user has the necessary permission in the operating system to install the software.
2. Download the Visual Studio extension for C# and Azure, as seen in the following figure:

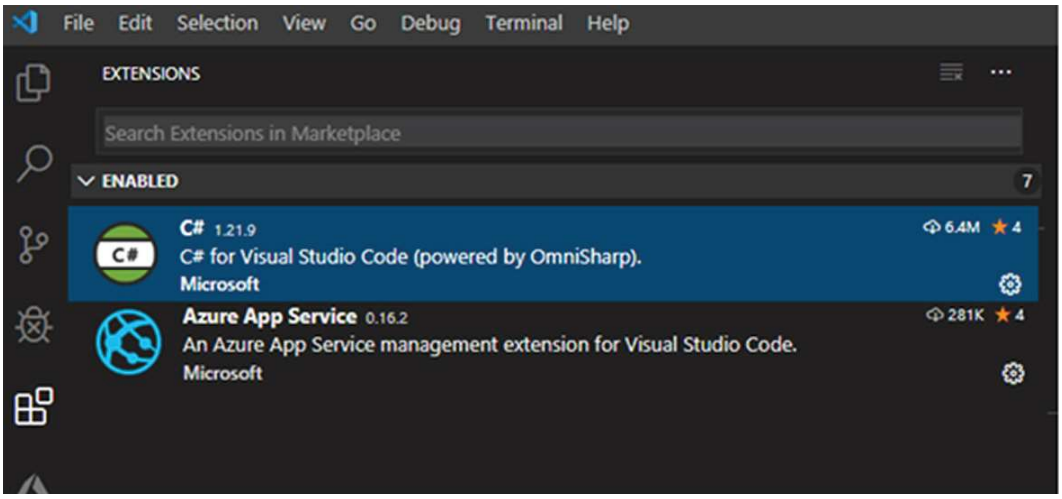

***Figure 1.2**: Visual Studio Code extensions*

These are the two extensions that can be used along with the code samples in this book, and Visual Studio Code can be used as an alternative to the Visual Studio 2022 IDE if you would like to use more command lines for .NET applications.

Introduction to Visual Studio 2022

The Visual Studio 2022 is such a powerful IDE for software development, which allows you to create, build, debug and deploy your .NET applications using a single tool with numerous plugins and extensions, giving you a rich user experience as a developer. It includes the possibility of having access to external resources such as databases and cloud services. In addition, the Visual Studio contains natively installed project templates along with the initial setup, including code samples for Web projects, Desktop, and Mobile applications, which give us a quick way to start a project without having to set up everything from scratch in terms of simple and common configurations. To access those templates, click the **File** option on the super menu and choose the **New Project** option, as seen in *Figure 1.3*:

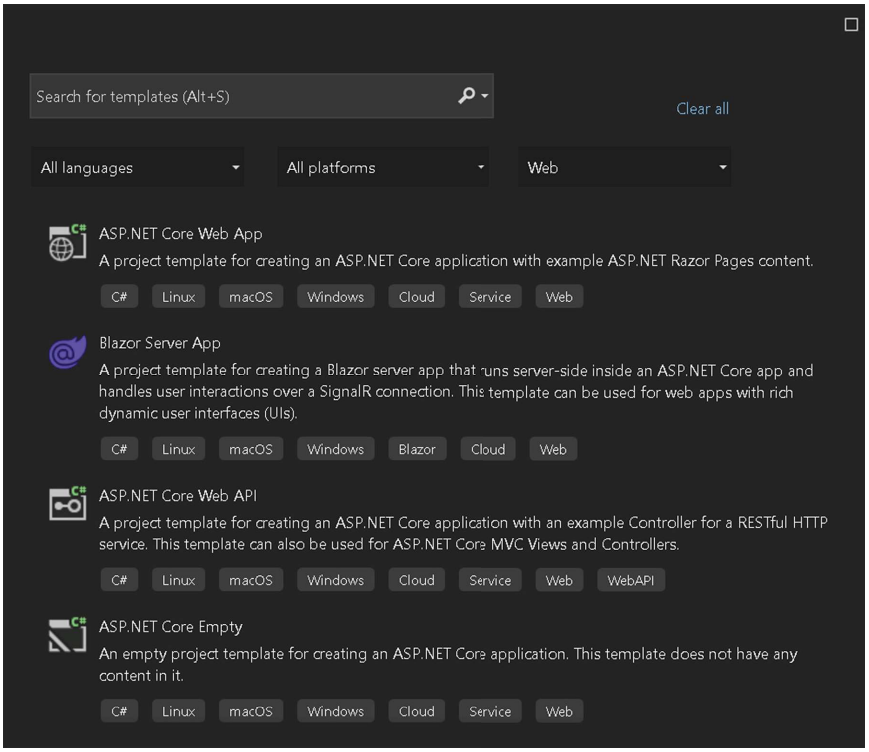

***Figure 1.3**: Visual Studio templates*

After installing the most recent Visual Studio version, the most popular project templates for .NET applications become available, according to the workloads chosen during the installation, such as Console App, ASP.NET Core Web API, Blazor App, and much more. Each project template has other sub-types ready to use, a variation of the main template. Those templates are time-saving, speeding up the software development process and helping learner developers understand how each type of project can be structured.

Open-source communities, companies, and individual developers who publish the templates as Visual Studio extensions on the Visual Studio Marketplace website develop and share many extra free templates. Additionally, Visual Studio extensions for many other purposes can be downloaded on the same website to get a better experience for software development within Visual Studio, getting integrations with third-party tools.

After creating any simple project, the template list available in Visual Studio, you are redirected to the Solution Explorer experience, having in a common place the code editor, access to the files and folder structure for your project, and access to any relevant external resources, as seen in *Figure 1.4*:

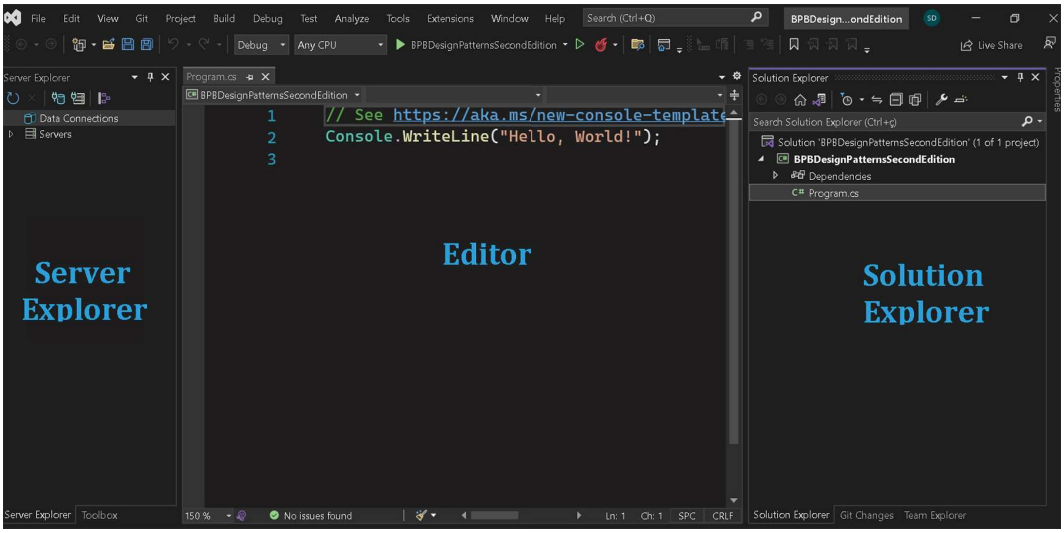

***Figure 1.4**: Visual Studio Code editor área*

On the **Solution Explorer** sidebar, you can access all the existing folders in files of your projects, including all the resources related to all the linked projects if you have multiple projects in the same solution. Furthermore, this **Solution Explorer** view allows you to have access to extra options for the configuration of your projects, such as the installation of extra packages, third-party libraries, and other resources available through Nuget Packages and project references by right-clicking on an individual project in the solution and choosing the **Manage NuGet Packages** option, as seen in *Figure 1.5*:

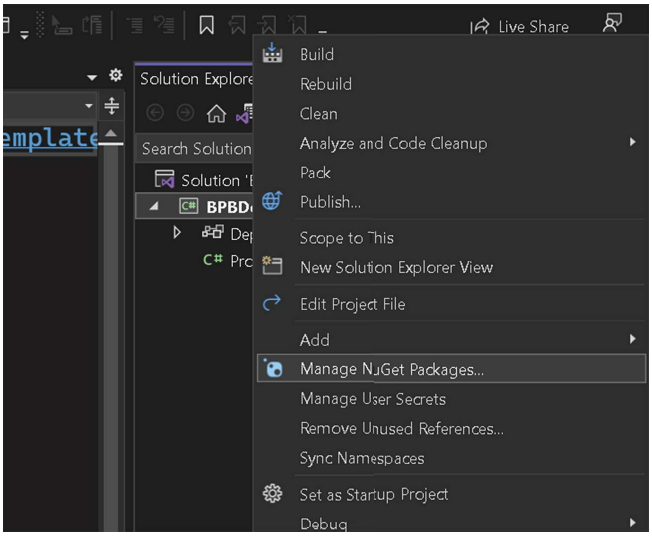

***Figure 1.5**: Manage NuGet Packages option*

After choosing the underlying **Manage Nuget Packages** option, a new window is available where it is possible to search and install any published Nuget packages, specifying the desired version, as seen in *Figure 1.6*:

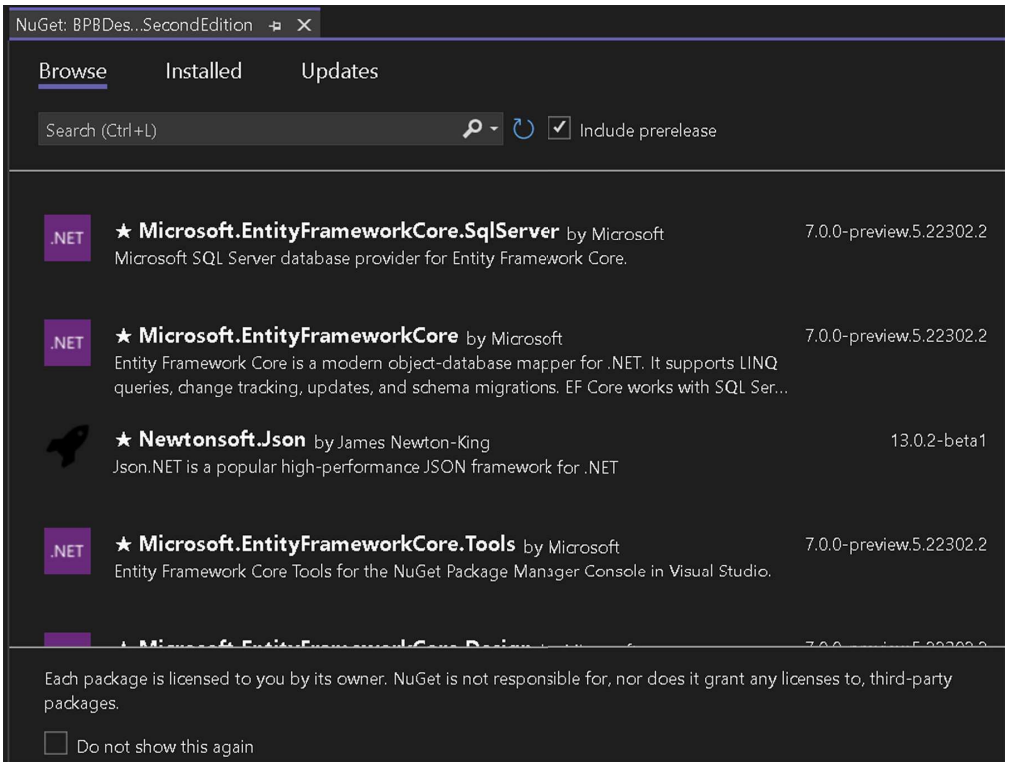

***Figure 1.6**: Nuget Packages window*

Another exciting option is to install packages using the commandline, which makes the process much easier and faster once the developer has enough familiarity with the available commands. Under the Nuget Packages Console window, which appears at the bottom of Visual Studio, it is possible to use the **`dotnet install`** command, followed by the name of the package, as seen in *Figure 1.7*:

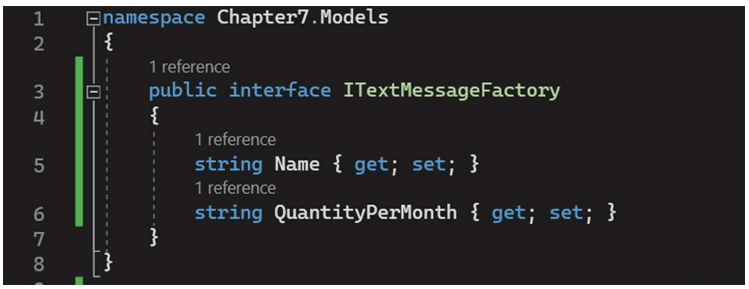

***Figure 1.7**: Dotnet CLI for Nuget Packages*

Continuing the Visual Studio coding area presented in *Figure 1.4*, the Editor sidebar is where you develop your code. It can get tips by Visual Studio on code syntax, indentation, codification error messages, navigating into classes, functions and functions and methods, including project references within the source code. The editor has a high level of customization, meaning many settings can be changed to adjust according to your needs. This customization involves background color, contrast level, font size, and more.

Finally, on the **Server Explorer** sidebar shown in *Figure 1.4*, it is possible to connect to external resources, including databases, application servers, cloud infrastructure, and on-premise features. With all the integrated features in Visual Studio, you can keep all the work in a single and shared place, which has an excellent value for high productivity. In the select Toolbar, there is an option to run applications, save pending changes, hot reload, debug capabilities, comment features, options to undo recent changes, and much more, as seen in *Figure 1.8*:

Figure 1.8: *Visual Studio Toolbar*

The .NET platform allows us to build applications using C#, Vb.Net, and F# languages. F# is a programming language based on the functional programming paradigm, and the Vb.NET is a programming language largely used in .NET projects since the beginning of the platform, despite the fact the C# language has taken over the preference of .NET developers in general to build .NET enterprise applications. Within Visual Studio, under the project creation dialog, you can choose the language for your project by choosing the correct and corresponding project template, as seen in *Figure 1.9*:

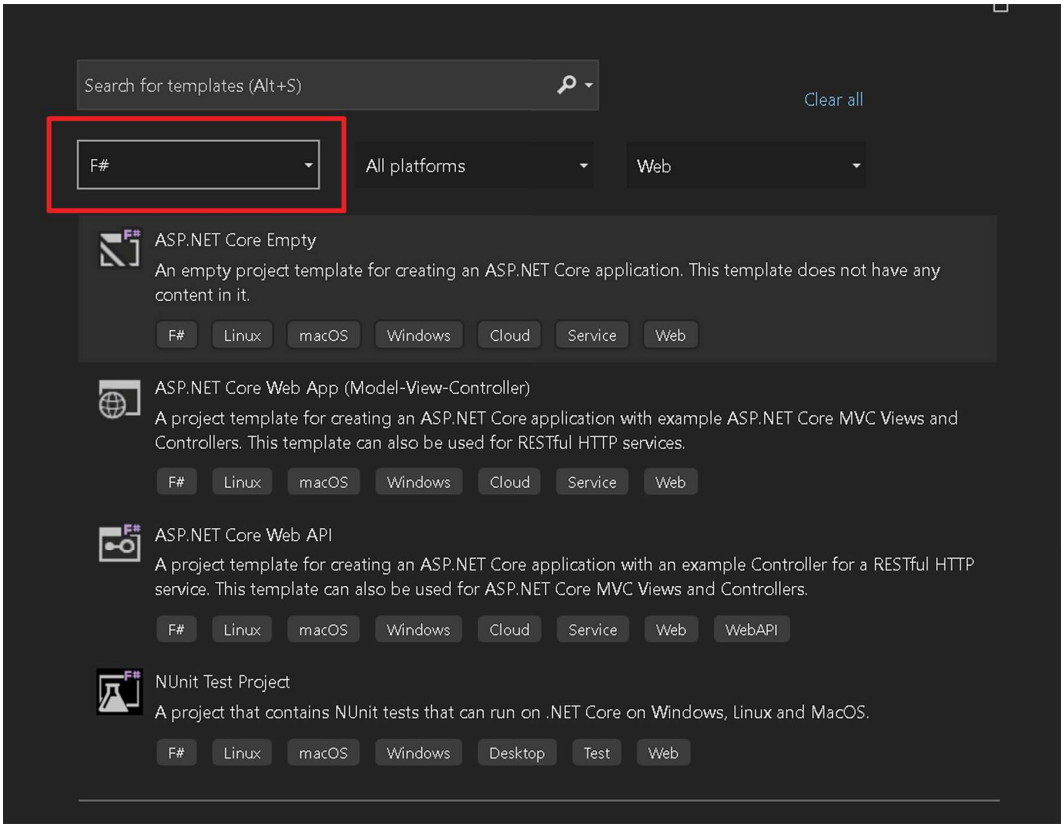

***Figure 1.9**: Visual Studio templates for F# language*

Given the example in *Figure 1.9*, the Visual Studio shows all the available project templates based on the F# language. Therefore, there are alternatives to the C# language within the .NET platform, including the possibility of multiple projects in the same solution, each targeting a different programming language as the core language.

Visual Studio is a complete IDE for .NET applications. It does not require the installation of extra tools to build and execute standard programs based on C# or any other language supported by .NET.

Introduction to Visual Studio Code

At first glance, Visual Studio Code seems quite different from Visual Studio 2022 in terms of user experience. Still, both have the same purposes: to create, debug, build and deploy applications using the .NET platform or other platforms. At its core, this light IDE contains many extensions to make our routine as developers easier and

more productive. It is certain to say that the main difference between Visual Studio Code and Visual Studio 2022 is the cross-platform capabilities for development once the first one is available for Linux, Windows, and macOS operating systems. Still, Visual Studio is available only for Windows and macOS, with limited capabilities in the macOS version. On the other hand, you must install and enable each extension and extra component on Visual Studio Code to get everything you need to develop specific applications. In terms of comparison with other Visual Studio versions, the Figure 1.10 shows that the options available for Visual Studio Code are pretty similar to the ones present in Visual Studio 2022:

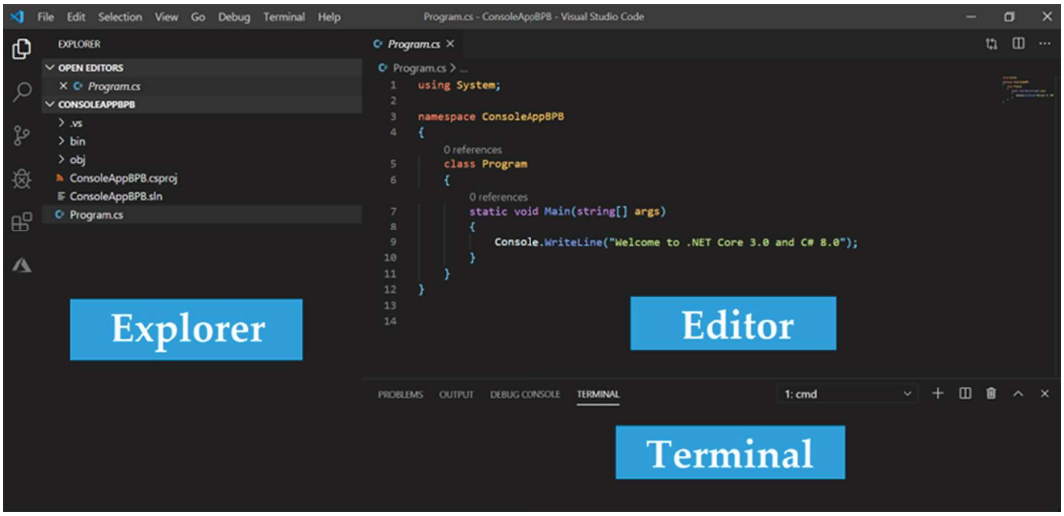

***Figure 1.10**: Visual Studio Code editor áreas*

On the **Explorer** sidebar, you can access all the existing folders and files of your projects, including the possibility of managing multiple and simultaneous projects in various windows in the same Explorer. As seen in *Figure 1.10*, the **Editor** sidebar is where you develop the actual code and can get a similar experience as Visual Studio regarding code tips, snippets, and much more.

Visual Studio Code is a command-line-based editor, meaning it may have fewer visual options in the IDE to manage and develop your projects. However, you can benefit from quickly running, building, and creating new files for your application using commands without having to use the mouseclicks for it, increasing productivity if you are familiar with the commands and shortcuts available.

The traditional complete Visual Studio 2022 IDE version relies on the existing specification within `.sln` (solution) files and specific extensions for projects, such as `.csproj` files for C# projects, to open the project solutions correctly. Although, Visual Studio Code allows you to run code by only opening simple folders and files, which

is useful in case you want to run other languages that are not .NET related in the same Visual Studio Code simultaneously.

Speaking of productivity and menu options, Visual Studio Code is more flexible and customizable than Visual Studio, as it contains many predefined commands and shortcuts to get files, a more friendly search option across folders and files, and much more, as seen in *Figure 1.11*:

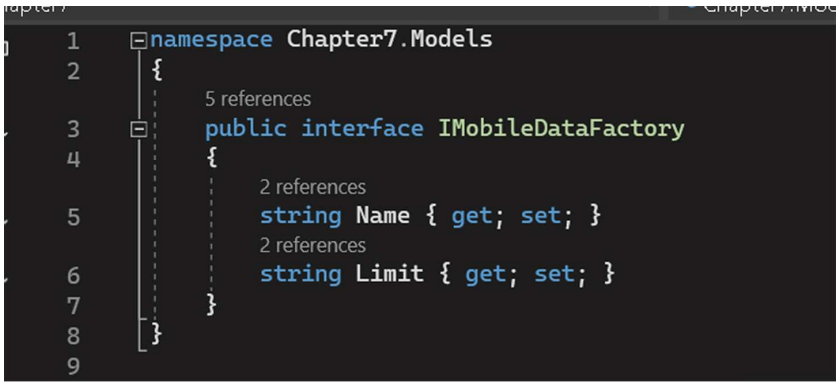

***Figure 1.11**: Visual Studio Code shortcuts*

Visual Studio is the most popular code editor in the market right now. The features available are increasing exponentially over time because of the intense contributions by active technical communities behind this project, providing a lot of extensions and new capabilities.

History of .NET

The .NET platform was created long before Web applications became popular and even before the first C# language. Despite the massive adoption of .NET these days, there is a first impression that the scenario in the market was always like the current state, especially for those who did not experience software development around the beginning of the 1990s. Looking back at almost three decades ago, when other programming languages were prevalent like Java and PHP at this time, it seemed essential for Microsoft to enter this great market of development tools for enterprise applications. In this context, the first version of the .NET platform was launched as a beta product in 2000. In February of 2022, the official 1.0 version was finally released, supporting Windows 98, Millennium, and XP.

The most important feature of this first version was the **Common Language Runtime (CLR)** capability, which allowed developers to create .NET applications using more than one programming language within the same solution. The .NET