A circular graphic in the top right corner of the cover features a topographic map with blue contour lines and a dashed grid, set against a dark blue background. A thin yellow arc is visible on the left side of the circle.

Beginning Entity Framework Core 5

From Novice to Professional

—

Eric Vogel

Apress®

Beginning Entity Framework Core 5

From Novice to Professional

Eric Vogel

Apress®

Beginning Entity Framework Core 5: From Novice to Professional

Eric Vogel
Okemos, MI, USA

ISBN-13 (pbk): 978-1-4842-6881-0
<https://doi.org/10.1007/978-1-4842-6882-7>

ISBN-13 (electronic): 978-1-4842-6882-7

Copyright © 2021 by Eric Vogel

This work is subject to copyright. All rights are reserved by the Publisher, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilms or in any other physical way, and transmission or information storage and retrieval, electronic adaptation, computer software, or by similar or dissimilar methodology now known or hereafter developed.

Trademarked names, logos, and images may appear in this book. Rather than use a trademark symbol with every occurrence of a trademarked name, logo, or image we use the names, logos, and images only in an editorial fashion and to the benefit of the trademark owner, with no intention of infringement of the trademark.

The use in this publication of trade names, trademarks, service marks, and similar terms, even if they are not identified as such, is not to be taken as an expression of opinion as to whether or not they are subject to proprietary rights.

While the advice and information in this book are believed to be true and accurate at the date of publication, neither the authors nor the editors nor the publisher can accept any legal responsibility for any errors or omissions that may be made. The publisher makes no warranty, express or implied, with respect to the material contained herein.

Managing Director, Apress Media LLC: Welmoed Spahr
Acquisitions Editor: Jonathan Gennick
Development Editor: Laura Berendson
Coordinating Editor: Jill Balzano

Cover image designed by Freepik (www.freepik.com)

Distributed to the book trade worldwide by Springer Science+Business Media LLC, 1 New York Plaza, Suite 4600, New York, NY 10004. Phone 1-800-SPRINGER, fax (201) 348-4505, e-mail orders-ny@springer-sbm.com, or visit www.springeronline.com. Apress Media, LLC is a California LLC and the sole member (owner) is Springer Science + Business Media Finance Inc (SSBM Finance Inc). SSBM Finance Inc is a **Delaware** corporation.

For information on translations, please e-mail booktranslations@springernature.com; for reprint, paperback, or audio rights, please e-mail bookpermissions@springernature.com.

Apress titles may be purchased in bulk for academic, corporate, or promotional use. eBook versions and licenses are also available for most titles. For more information, reference our Print and eBook Bulk Sales web page at <http://www.apress.com/bulk-sales>.

Any source code or other supplementary material referenced by the author in this book is available to readers on GitHub via the book's product page, located at www.apress.com/9781484268810. For more detailed information, please visit <http://www.apress.com/source-code>.

Printed on acid-free paper

Table of Contents

About the Author	xi
About the Technical Reviewer	xiii
Introduction	xv
Part I: Getting Started	1
Chapter 1: Installation	3
Install Visual Studio	3
Create a Project	5
Install Entity Framework	9
Install the Core Tools Package	12
SQL Server Database	13
Summary.....	14
Chapter 2: Project Structure.....	15
Core Project	15
Unit Testing Overview	18
Data Access Layer Project	19
Main Project References.....	20
Summary.....	20
Part II: Core Features.....	23
Chapter 3: Entities	25
Person Entity.....	25
Address Entity	26
Simple Navigation Property	26
Entity Property Constraints	27

TABLE OF CONTENTS

- Entity Schema Attributes..... 29
- More Entity Schema Attributes 31
- Mapping of the Primary Key Column 31
- Enum Mapping 32
- Entity Inheritance Mapping 33
 - Table-per-Hierarchy..... 33
 - Table-per-Type..... 34
- Summary..... 36
- Chapter 4: Database Context 37**
 - Creating a Simple Database Context..... 37
 - Connecting to Our Database 37
 - Accessing Entities in a Database Context..... 39
 - Saving Entity Changes 40
 - Configuring a Database Context..... 40
 - Set a Database Schema for All Entities 40
 - Composite Key Constraint 41
 - Primary Key Constraint..... 41
 - Default Property Value 42
 - Foreign Key Relationships 43
 - Summary..... 43
- Chapter 5: Creating a Database from Code..... 45**
 - Setting the Connection String 45
 - Creating the Initial Migration 48
 - Creating the Database from the Migration..... 53
 - Connect to the New Database from Visual Studio 54
 - Summary..... 56
- Chapter 6: Seeding Data 57**
 - Populating Lookup Data 57
 - Seeding Test Persons..... 63

Seeding Addresses	64
Summary.....	64
Chapter 7: Getting Data	65
LINQ Queries	65
Select Anonymous Types	66
Select Object Transformation	66
Joins in LINQ.....	67
Select the Navigation Property	68
Sorting Data.....	68
Method Syntax Queries.....	69
Testing Our Data.....	70
Summary.....	75
Chapter 8: Inserting Data	77
Inserting the Root Entity.....	77
Inserting Child Records.....	78
Primary Key Values	80
Identity Seeded Primary Key	80
Guid Primary Key	80
Non-computed Primary Key	80
Foreign Key Values.....	80
Default Values	81
Record Insertion Integration Tests	81
Summary.....	85
Chapter 9: Updating Data.....	87
Updating the Root Entity	87
Updating a Child Entity.....	88
Integration Test	89
Summary.....	95

TABLE OF CONTENTS

- Chapter 10: Deleting Data..... 97**
 - Deleting the Root Entity 97
 - Deleting a Child Entity..... 97
 - Cascade Delete..... 98
 - Client Set Null Delete Behavior..... 99
 - Restrict Delete Behavior..... 100
 - Set Null Delete Behavior..... 101
 - Integration Test 101
 - Summary..... 104

- Chapter 11: Navigation Properties 105**
 - Mapping a Parent Entity in a One-to-Many Relationship 105
 - Mapping a Related Entity to Parent..... 106
 - One-to-Many Integration Test 107
 - Many-to-Many Relationships..... 110
 - Map a Relationship Through Navigation Properties 110
 - Create and Run Migration..... 111
 - Many-to-Many Integration Test 114
 - Summary..... 130

- Part III: Advanced Features 131**

- Chapter 12: Aggregations..... 133**
 - Unit Test Setup 133
 - Count..... 134
 - Adding Age to Person..... 135
 - Min 137
 - Max 137
 - Average 138
 - Average Unit Test..... 138
 - Sum..... 138
 - Group By 139

Direct Group By	139
Group By with Count.....	140
Group By with Min	140
Group By with Max	141
Group By with Average	142
Group By with Sum.....	142
Summary.....	143
Chapter 13: Stored Procedures.....	145
Add Stored Procedures to a Database	145
Set Up Unit Tests	149
Test the GetPersonsByState Stored Procedure	150
Test the AddLookUpItem Stored Procedure.....	151
Summary.....	152
Chapter 14: Migrations	153
What Is a Migration?	153
How to Add a Migration?	153
The Migration API	154
Common Schema Commands.....	154
Add a Column to a Table.....	154
Change a Column in a Table	155
Common Data Commands	155
Inserting Data.....	155
Deleting Data.....	156
Update Data in a Migration.....	157
How to Run Migrations.....	158
Update to the Most Recent Migration.....	159
Migrate to a Specific Migration	159
Run Migration from Code	159
Automatic Data Migrations	159
Summary.....	161

- Part IV: A Model Web Application 163**
- Chapter 15: Authentication on the Web 165**
 - Install Identity NuGet Packages 165
 - Initialize Identity on App Startup 166
 - Scaffold UI..... 172
 - Update Database..... 174
 - Updating UI..... 175
 - Summary..... 180
- Chapter 16: Displaying Data on the Web 181**
 - Scaffolding UI..... 181
 - Generated List View and Model..... 183
 - Generated Details View and Model 187
 - Adding Addresses to the Details View..... 190
 - Adding Contacts to Navigation..... 194
 - Testing the App 197
 - Adding Some Polish 199
 - Summary..... 203
- Chapter 17: Inserting Data on the Web..... 205**
 - Generated Create Razor Page 205
 - Generated View 205
 - Generated Model 207
 - Person Model Validation 209
 - Adding Addresses 210
 - Address Model Validation 210
 - Updating UI 214
 - Update the Razor Page Model 219
 - Running the App 221

Adding Some Polish	223
Update the Model	223
Update the UI	226
Summary.....	231
Chapter 18: Updating Data on the Web.....	233
Generated Edit Razor Page View.....	233
Generated Edit Razor Model.....	235
Add a Friendly Address Update	238
Update the Model	238
Update the UI.....	242
Running the App	247
Summary.....	248
Chapter 19: Deleting Data on the Web.....	249
Generated Delete Razor Page View.....	249
Generated Model.....	251
Updating Page Controller Code to Delete Addresses	253
Running the App.....	255
Showing a Person’s Address on the Delete Form	256
Update the Page Controller.....	256
Update the UI.....	257
Summary.....	263
Chapter 20: Reporting on the Web.....	265
Creating the Razor Page	265
Updating the Razor Model.....	266
Updating the Razor View	267
Add a Link to Navigation	269
Running the Report	272
Adding Pagination	272
Adding Sample Data	272
Adding Pagination Support.....	275

TABLE OF CONTENTS

- Updating the Model for Pagination 277
- Add Pagination to the Razor View..... 278
- Running the Finished Report 279
- Summary..... 279
- Chapter 21: Authorization on the Web..... 281**
- Enable Roles 281
- Add Test Users 281
- Add and Assign Roles on Startup..... 282
- Enforce Authorization..... 284
- Authorize the Contacts Menu Item 285
- Secure Razor Pages 287
- Summary..... 305
- Part V: Learning More 307**
- Chapter 22: Delving Deeper 309**
- Chapter 23: Conclusion..... 311**
- Index..... 313**

About the Author

Eric Vogel is a seasoned contributor to *Visual Studio Magazine* and Senior Software Developer at Red Cedar Solutions Group. He has been developing .NET Framework web and desktop solutions for 13 years. He holds a Bachelor of Science degree in computer science from Michigan State University. He is Acting President of the Greater Lansing User Group for .NET.

About the Technical Reviewer



Pieter Nijs is a Belgian .NET architect with a passion for mobile and cloud development. He has played a key role in several projects ranging from large consumer-facing healthcare, telecom, and media apps to smaller LOB applications. As a mobile development expert at Xpirit Belgium, he loves helping customers implement mobile-first and cloud-first applications. Pieter is primarily interested in the Microsoft stack, so his interest and expertise translate to technologies like .NET, C#, XAML, Xamarin, UWP, Azure,

Azure DevOps, and so on. Both at work and in his spare time, Pieter is constantly working and playing with these and other new technologies. He likes to tell everybody about the things he does, sharing his knowledge. Hence, you can find him speaking at conferences, giving trainings, and blogging at blog.pieeatingninjas.be. Since 2017, Pieter has been receiving the Microsoft MVP Award in the Windows Development category for sharing his passion and expertise with the community.

Introduction

This book is aimed at readers who have a beginner's knowledge of the .NET Framework who are looking to use Entity Framework (EF) Core 5 for a side project or business application. No prior knowledge of Entity Framework Core 5 is required. The book guides the user through the basics of Entity Framework Core 5 up to some more advanced concepts and culminates in creating an ASP.NET Core Razor Pages web application that has full create, read, update, and delete (CRUD) capability.

We will be using the NUnit Framework to test Entity Framework Core 5 behavior before using it in a full web application. You will first go over how to query data and then how to insert, update, and delete data. Later in the book, we will go over more advanced techniques like how to aggregate data, use navigation properties to get related data, and call custom raw SQL and stored procedures.

The later chapters also cover basic authentication, authorization, and reporting in an ASP.NET Core Razor Pages web application. You will learn how to query and manipulate a SQL Server database by testing each facet through NUnit integration tests.

PART I

Getting Started

CHAPTER 1

Installation

In order to use Entity Framework Core 5, you will need some tools. These tools include Visual Studio (VS) 2019 and some NuGet packages for your solution.

Install Visual Studio

The first step is getting a version of Visual Studio 2019. The Community Edition of Visual Studio 2019 for Windows can be used for all code in this book. If you are running macOS, you can also opt to install Visual Studio for Mac. I'll be using Windows for the remainder of the book, but the steps are about the same for macOS.

You can download Visual Studio 2019 from <https://visualstudio.microsoft.com/downloads/> as seen in Figure 1-1.

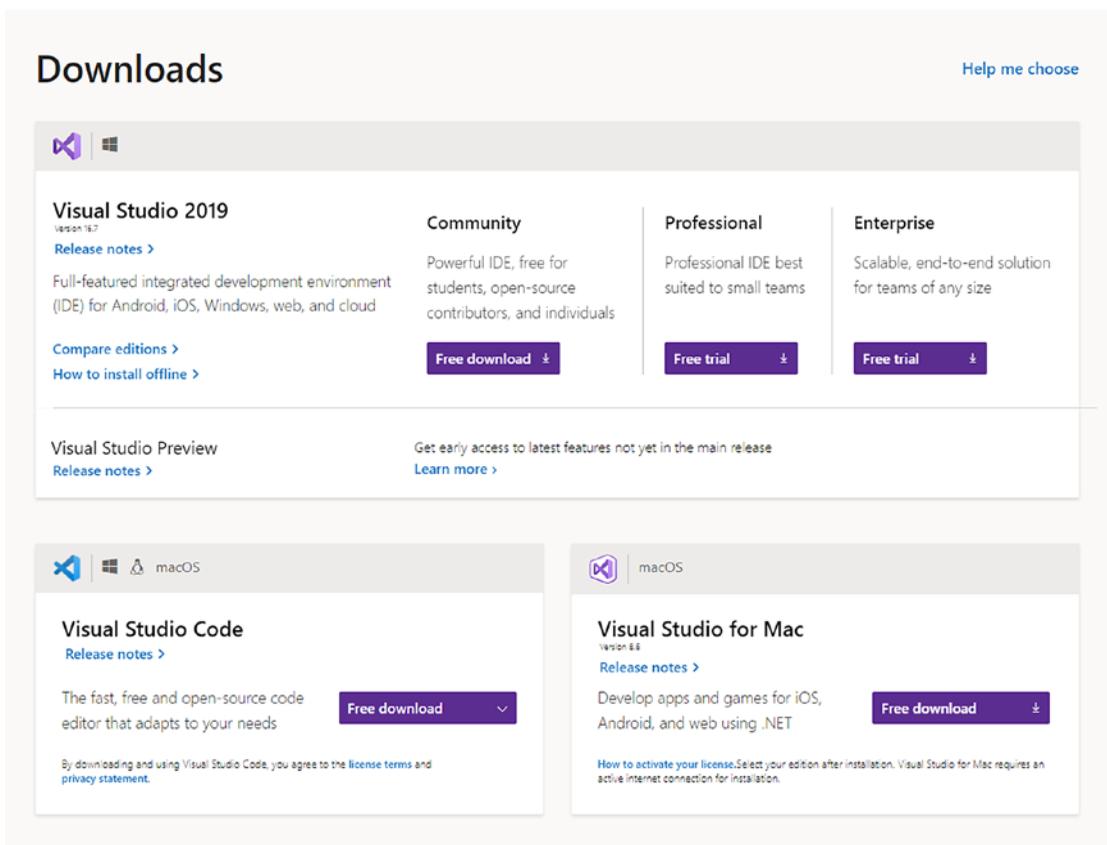


Figure 1-1. Download VS 2019

Next, install the latest Visual Studio 2019 and select the .NET Core option as seen in Figure 1-2.

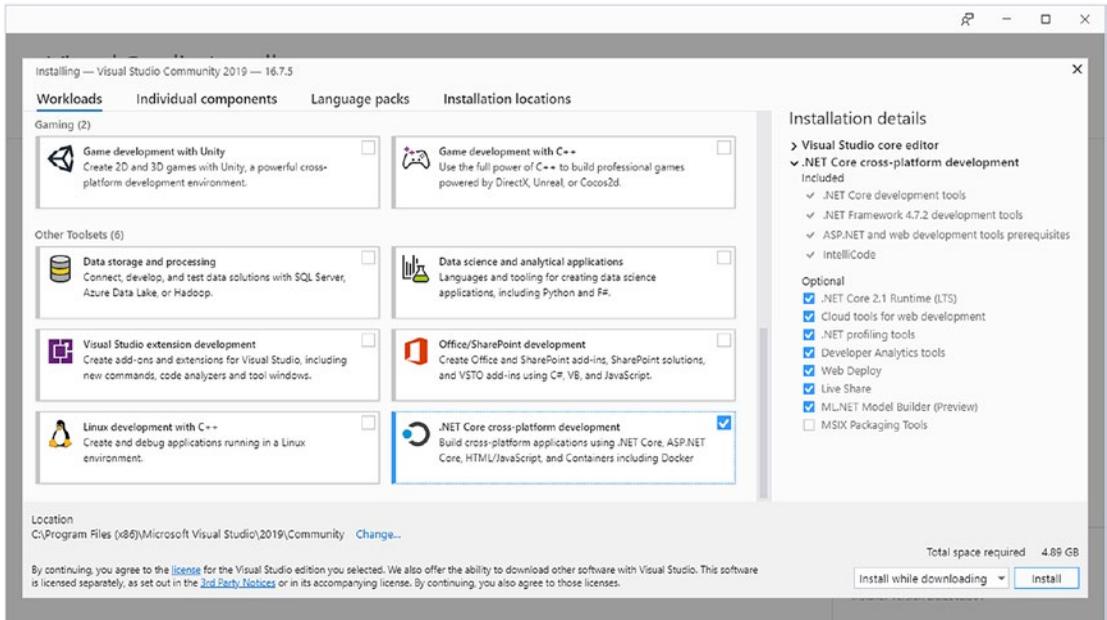


Figure 1-2. Install .NET Core Support

Entity Framework Core 5 will run on both .NET Core 3.1 and .NET 5. For this book, we will be using .NET Core 3.1.

Create a Project

Entity Framework Core 5 works on a variety of application types from console apps, desktop apps, and web apps. For this book, we will be creating an ASP.NET Core MVC (Model-View-Controller) app.

We will now create the project in Visual Studio and install all the required tools to use Entity Framework Core 5. First, create a new project in Visual Studio as seen in Figure 1-3.

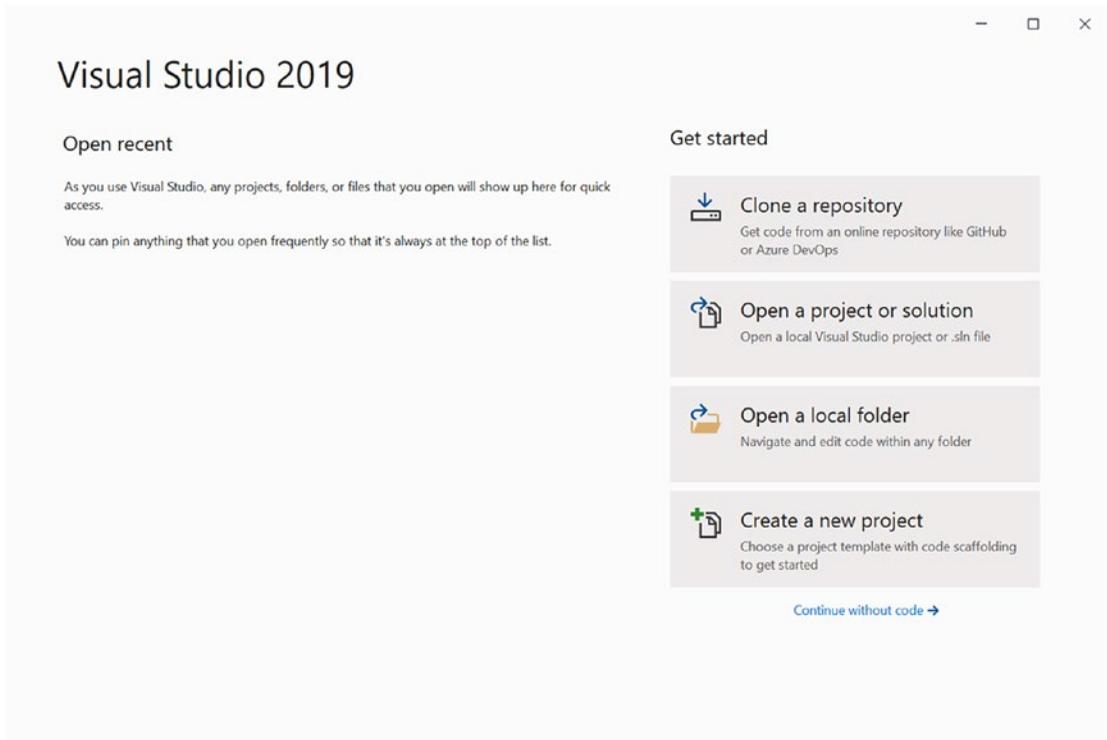


Figure 1-3. *Create a New Visual Studio 2019 Project*

Then create a new ASP.NET Core web application as seen in Figure 1-4.

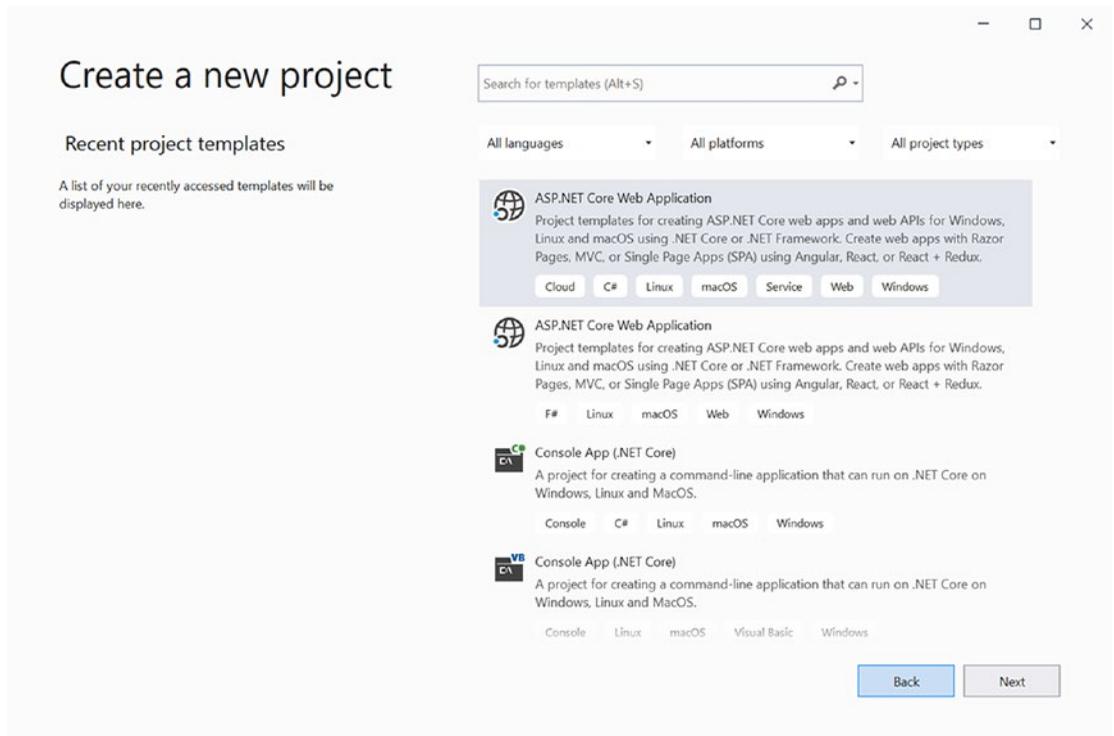


Figure 1-4. *New ASP.NET Core Web App*

Next, name your ASP.NET Core web application. I named my app `EFCore5WebApp` as seen in Figure 1-5.

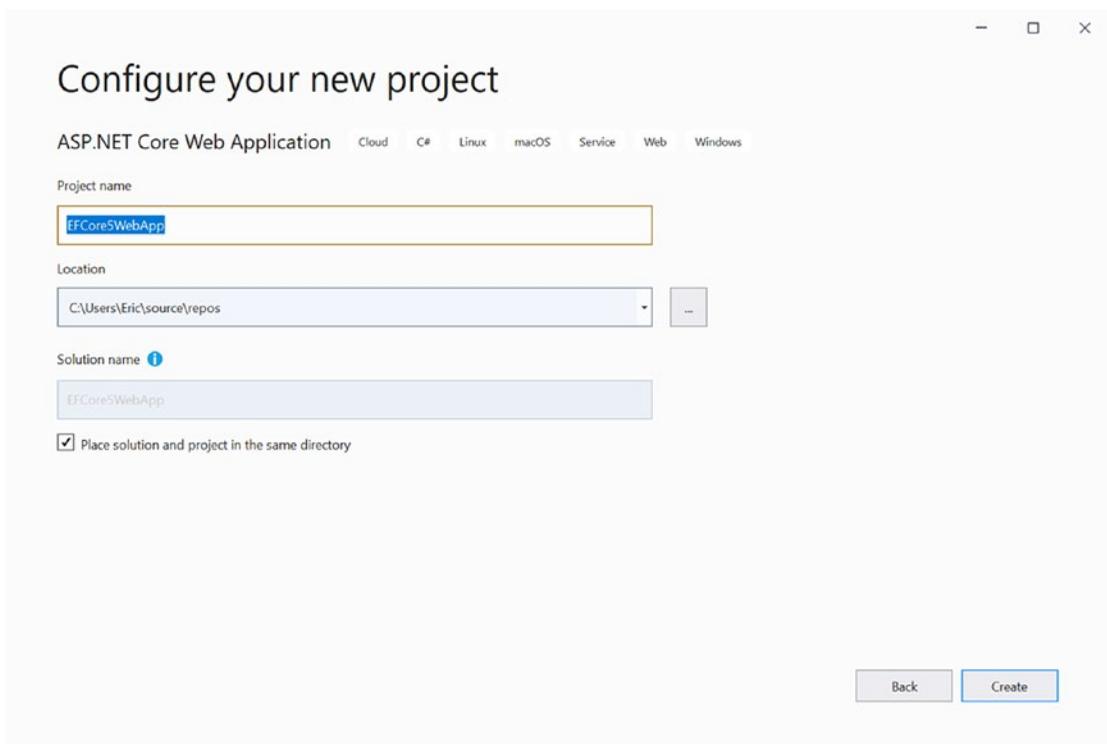


Figure 1-5. Naming the Web App

After that, select ASP.NET Core 3.1 as the target framework and select the Model-View-Controller template as seen in Figure 1-6.

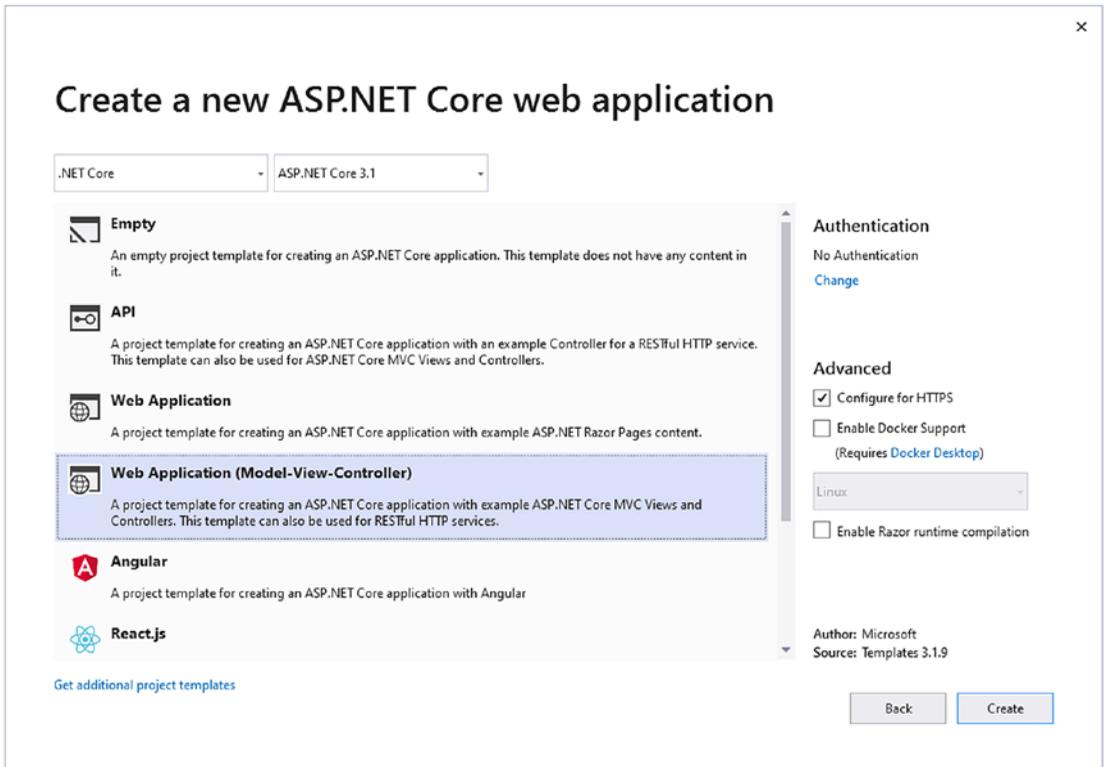


Figure 1-6. *Creating an ASP.NET Core 5.0 MVC App*

Then click the “Create” button to create the project and solution. After this, you will see the generated project opened in Visual Studio.

Install Entity Framework

Entity Framework Core 5 has providers for Microsoft SQL Server, SQLite, Cosmos, and in-memory databases. For this book, we will be using the SQL Server provider.

Entity Framework is installed through the NuGet Package Manager. We will be installing the Entity Framework Core 5 SQL Server NuGet package, which will allow us to interact with a Microsoft SQL Server instance.

Now it is time to install the Entity Framework Core 5 SQL Server NuGet package into the web app. Open the NuGet installer and search for “Entity Framework” and click the “Include prerelease” checkbox as shown in Figure 1-7.

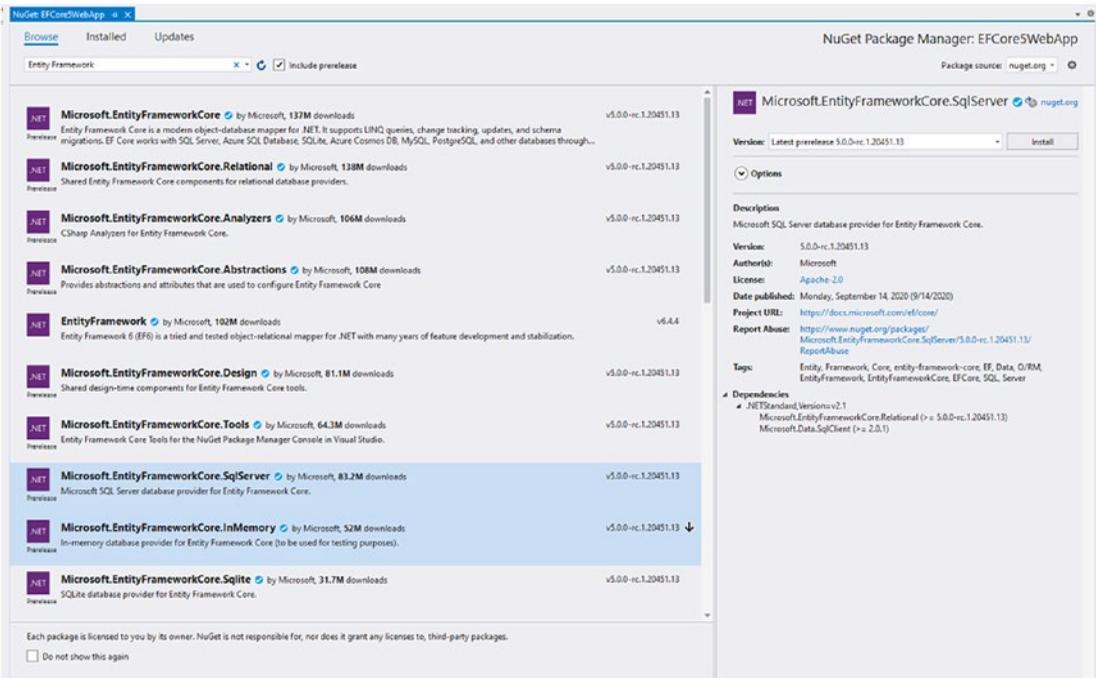


Figure 1-7. Install the EF Core 5 SQL Server NuGet Package

Click the Install button, and you should be prompted to confirm your installation as seen in Figure 1-8. Make sure the version is 5.* and that the package name is Microsoft.EntityFrameworkCore.SqlServer, which will install Entity Framework Core 5 with SQL Server support.

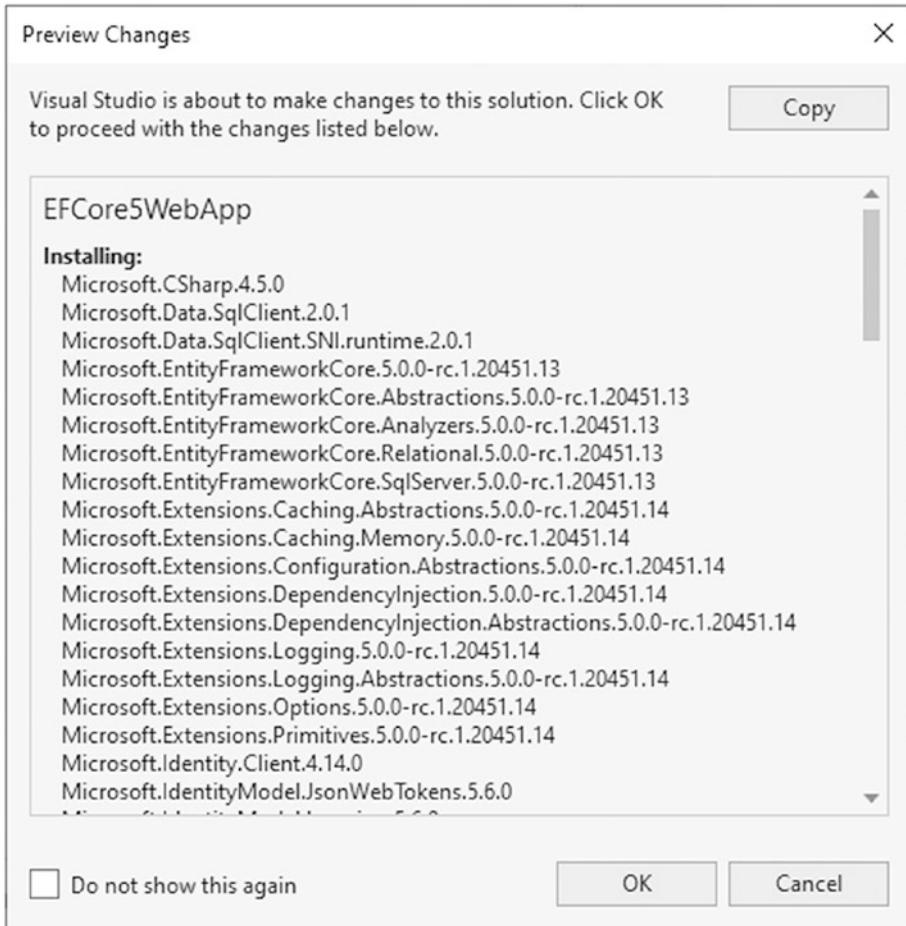


Figure 1-8. Confirm EF Core SQL Server NuGet Installation

Lastly, you'll be prompted to accept the license to install the EF Core 5 NuGet package as seen in Figure 1-9.

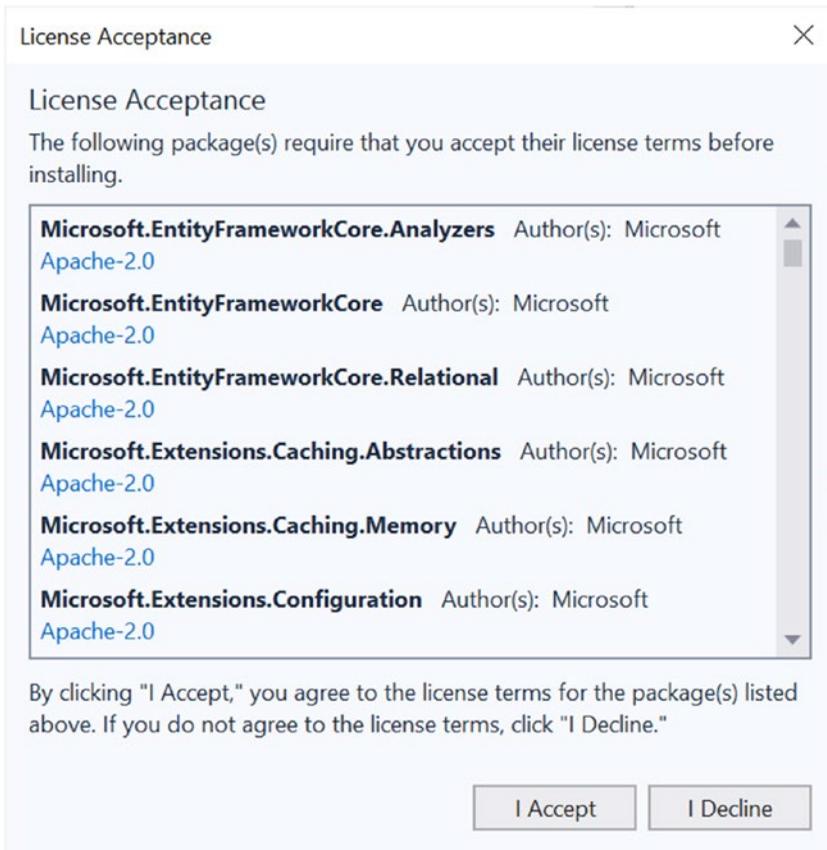


Figure 1-9. *Confirm EF Core 5 NuGet License*

Click the I Accept button, and the package will be installed.

Install the Core Tools Package

Finally, you’ll need to install the Entity Framework 5 Core Tools package, which will allow you to create and update your database from the NuGet Package Manager Console. Open up the NuGet Package Manager and search for “Entity Framework Tools” and install the package as seen in Figure 1-10. Make sure to install the package named “Microsoft.EntityFrameworkCore.Tools” and that the version is 5.*.

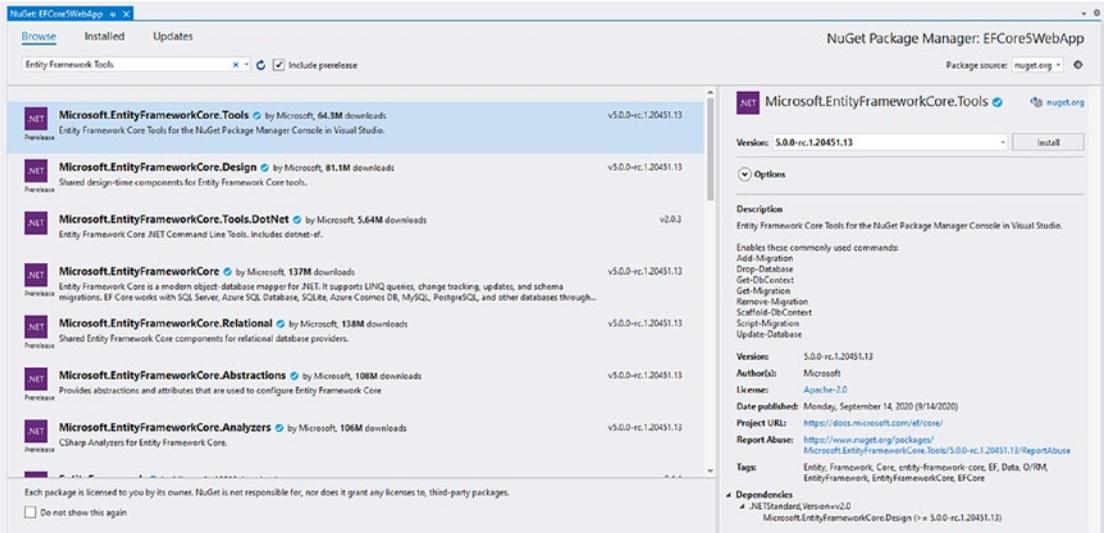


Figure 1-10. Install the EF Core 5 Tools NuGet Package

You have now successfully installed the needed tools to use Entity Framework Core 5. In the next chapter, we will cover how to move the Entity Framework 5 NuGet installation to a data access layer (DAL) project and how to structure your application using an N-tier architecture. You are well on your way to learning how to effectively use Entity Framework Core 5.

SQL Server Database

For this book, we will be using SQL Server Express LocalDb that is installed automatically by Visual Studio 2019 Preview as part of the .NET Core workload. Feel free to use your own full SQL Server instance. I will cover how to set the database connection string and create the database from code in Chapter 5.

Summary

In this chapter, you've installed the tools needed to use Entity Framework Core 5 with a Microsoft SQL Server instance. You've installed the latest Visual Studio 2019 and Entity Framework Core 5 into our solution.

With this groundwork in, we can move on to structuring our application in the next chapter to effectively use Entity Framework Core 5. A good architecture goes a long way in allowing the application to be easily tested through unit and integration tests and eases maintenance of the application over time.

CHAPTER 2

Project Structure

Creating the structure of your application is an important step. In this chapter, I will detail how to use a multilayered architecture to separate concerns in the application. This will make your application easier to maintain and extend.

Core Project

The Core project is where the commonly used code shared across the entire application will reside. This is where the entity classes used by Entity Framework will live. This is also where common interfaces and utility code will exist.

To get started, open the solution you created in Chapter 1. Then add a new .NET Standard C# class library project named “EFCore5WebApp.Core” to the solution as seen in Figure 2-1.

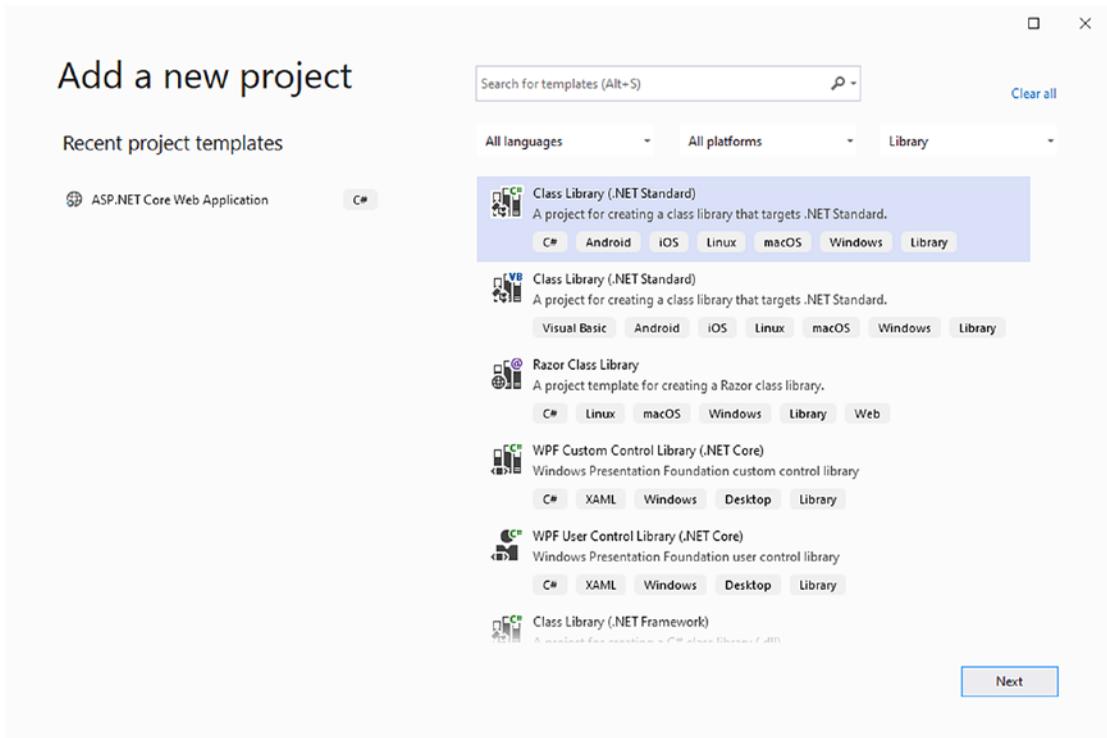


Figure 2-1. *New Class Library*

On the next screen, name your project “EFCore5WebApp.Core” as seen in Figure 2-2.

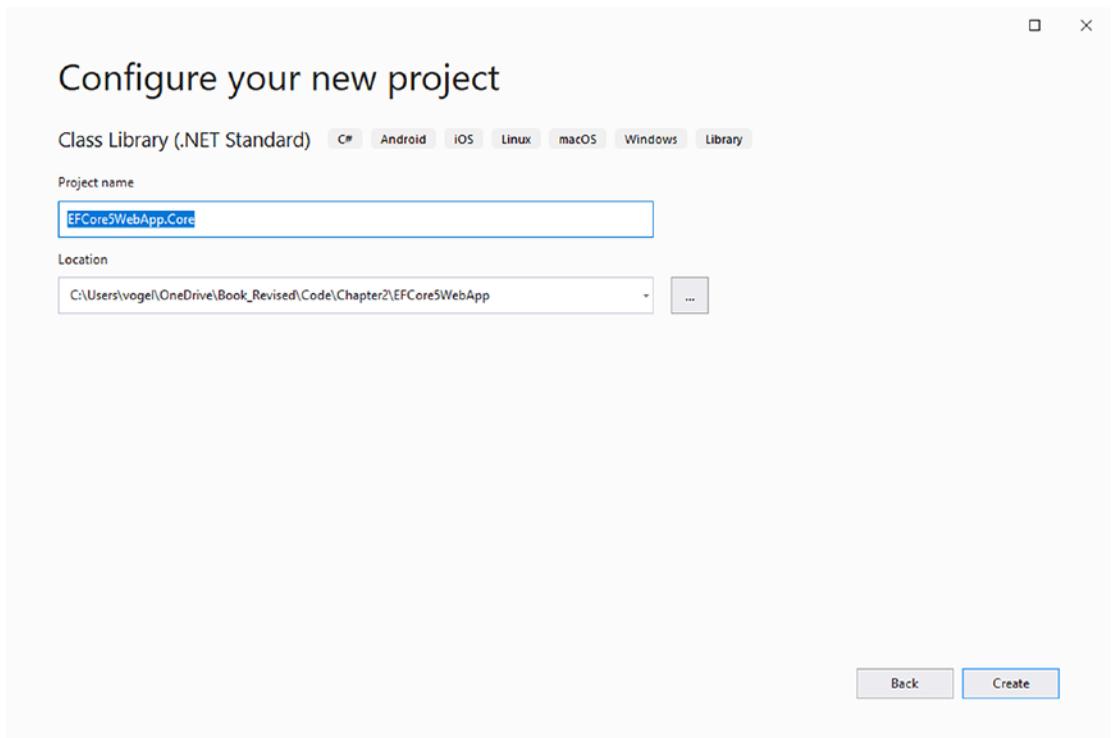


Figure 2-2. Naming the Core Project

Next, create a new NUnit Test Project (.NET Core) template project named “EFCore5WebApp.Core.Tests”. This project will be where we’ll create any needed unit tests for shared application code. You can easily find the correct template by searching for “NUnit” in the template dialog as seen in Figure 2-3.