

به نام حضرت دوست

آموزش کاربردی Pro ASP.NET Core MVC 6

همراه با پیاده‌سازی کامل یک پروژه با MVC و صفحات Razor

Adam Freeman

مهندس نادر نبوی
انتشارات پندار پارس

سرشناسه	فریمن، آدام، ۱۹۷۲ م. - Freeman, Adam
عنوان و نام پدیدآور	آموزش کاربردی MVC 6 Pro ASP.NET Core همراه با پیاده‌سازی کامل یک پروژه با MVC و صفحات Razor/تألیف آدام فریمن؛ ترجمه نادر نبوی.
مشخصات نشر	تهران: پندار پارس، ۱۴۰۲.
مشخصات ظاهری	۸۲۴ ص.: مصور، جدول.
شابک	978-622-7785-17-3
وضعیت فهرست نویسی	فیبا
پاداشت	عنوان اصلی: Pro ASP.NET Core MVC 6: Develop Cloud-Ready Web Applications Using MVC, Blazor, and Razor
موضوع	Pages, 2022. میکروسافت داتنت فریم‌ورک Microsoft .NET Framework وبگاهها -- برنامه‌های تألیفی Web sites -- Authoring programs علوم کامپیوتر Computer science نرم‌افزار -- مهندسی Software engineering
شناسه افزوده	نبوی، نادر، ۱۳۴۰ -- مترجم
رده بندی کنگره	۷۴/۷۶QA
رده بندی دیویی	۲۷۶/۰۰۵
شماره کتابشناسی ملی	۹۱۷۰۰۵۸
اطلاعات رکورد کتابشناسی	فیبا

انتشارات پندارپارس



دفتر فروش: انقلاب، ابتدای کارگرجنوبی، کوی رشتچی، شماره ۱۴، واحد ۱۶ www.pendarepars.com
 تلفن: ۶۶۵۷۲۳۳۵ - ۶۶۹۲۶۵۷۸ همراه: ۰۹۱۲۲۴۵۲۳۴۸
info@pendarepars.com



نام کتاب : آموزش کاربردی MVC 6 Pro ASP.NET Core

ناشر : انتشارات پندار پارس

تألیف : آدام فریمن

ترجمه : نادر نبوی

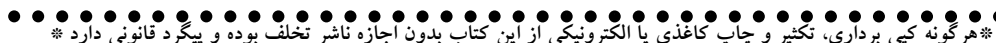
چاپ نخست : اردیبهشت ۱۴۰۲

شمارگان : ۱۰۰ نسخه

طرح جلد : رامین شکرالهی

چاپ، صحافی : روز

قیمت : ۵۵۰.۰۰۰ تومان شابک : ۹۷۸-۶۲۲-۷۷۸۵-۱۷-۳



*هرگونه کپی برداری، تکثیر و چاپ کاغذی یا الکترونیکی از این کتاب بدون اجازه ناشر تخلف بوده و پیگرد قانونی دارد *

فهرست

۱	فصل یکم: ASP.NET Core در عمل
۱	آشنایی با فریم‌ورک‌های MVC
۲	معرفی فریم‌ورک MVC
۲	معرفی صفحات Razor
۳	معرفی Blazor
۳	معرفی فریم‌ورک کمکی
۳	معرفی پلتفرم ANC
۴	سخنی در مورد ساختار کتاب
۴	نرم‌افزار مورد نیاز برای مثال‌های کتاب
۴	مطالب ارائه شده در کتاب
۵	فصل دوم: شروع به کار
۵	انتخاب ویرایشگر کد و محیط توسعه
۵	نصب ویژوال استدیو
۷	نصب NET SDK
۷	نصب Visual Studio code
۸	نصب SQL Server LocalDB
۱۰	ایجاد یک پروژه‌ی ANC
۱۱	باز کردن پروژه در ویژوال استدیو
۱۲	اجرای برنامه ANC
۱۴	آشنایی با Endpoint
۱۶	آشنایی با مفهوم مسیر
۱۶	چگونگی پردازش HTML
۱۹	خروجی پویا
۲۱	جمع‌بندی فصل دوم
۲۳	فصل سوم: ایجاد نخستین پروژه
۲۳	تنظیم سناریوی پروژه
۲۳	ایجاد پروژه
۲۴	آماده‌سازی پروژه
۲۵	افزودن مدل داده
۲۶	ایجاد نما و اکشن دوم
۲۸	متصل کردن اکشن‌ها به وسیله‌ی لینک

۲۹	ایجاد فرم ورود داده‌ها.....
۳۰	دریافت اطلاعات فرم.....
۳۲	استفاده از مقیدسازی مدل.....
۳۳	نخیره‌سازی اطلاعات فرم.....
۳۴	نمایش پاسخ‌ها.....
۳۶	اعتبارسنجی داده‌های فرم.....
۳۹	مشخص کردن فیلدهای نادرست فرم.....
۴۱	کار بر روی ظاهر سایت.....
۴۱	ظاهر نمای خوش‌آمد.....
۴۲	ظاهر نمای فرم.....
۴۳	ظاهر نمای Thanks.....
۴۴	ظاهر نمای ListResponses.cshtml.....
۴۷	فصل چهارم؛ آشنایی با ابزار توسعه.....
۴۷	ایجاد پروژه‌های ANC.....
۴۷	ایجاد پروژه با خط فرمان.....
۴۹	باز کردن پروژه.....
۵۰	افزودن کد و محتوا به پروژه.....
۵۱	کامپایل و اجرای برنامه‌ها.....
۵۲	استفاده از ویژگی Hot Reload.....
۵۴	کامپایل و اجرای برنامه‌ها در ویژوال استدیو کد.....
۵۴	کامپایل و اجرای برنامه‌ها در ویژوال استدیو.....
۵۴	مدیریت بسته‌ها.....
۵۵	مدیریت بسته‌های NuGet.....
۵۶	مدیریت بسته‌های ابزار.....
۵۶	مدیریت بسته‌های سمت مشتری.....
۵۸	دیبگ پروژه‌ها.....
۵۹	فصل پنجم؛ ویژگی‌های مهم C#.....
۶۰	ایجاد پروژه‌ی این فصل.....
۶۰	باز کردن پروژه.....
۶۰	فعال کردن ASP.NET Core MVC.....
۶۱	افزودن عناصر پروژه.....
۶۱	ایجاد مدل.....

۶۱	ایجاد نما و کنترلر.....
۶۲	انتخاب پورت HTTP.....
۶۳	آشنایی با عبارتهای سطح بالا.....
۶۳	آشنایی با عبارتهای سراسری using.....
۶۵	تحلیل وضعیت پوچ.....
۶۸	آزمایش مقادیر پوچ.....
۶۹	کاربرد عملگر شرطی Null.....
۷۰	لغو عمل تحلیل وضعیت پوچ.....
۷۱	لغو پیامهای هشدار وضعیت پوچ.....
۷۱	ترکیب رشتهها.....
۷۲	مقداردهی آغازین کلکسیونها و اشیاء.....
۷۴	عبارت new به شکل هوشمند.....
۷۵	آزمایش نوع اشیاء.....
۷۵	تشخیص نوع متغیر در switch.....
۷۶	استفاده از متدهای گسترش یافته.....
۷۸	کاربرد متدهای گسترش یافته در رابطه با اینترفیسها.....
۷۹	متدهای گسترش یافتهی فیلترکننده.....
۸۰	عبارتهای لاندا.....
۸۱	تعریف تابع با عبارت لاندا.....
۸۴	عبارتهای لاندا برای متدها و خصوصیتها.....
۸۵	بیان ضمنی نوع متغیر و انواع بی نام.....
۸۵	کاربرد انواع بی نام.....
۸۷	پیاده سازی پیش فرض در اینترفیس.....
۸۸	متدهای آسنکرون.....
۹۰	کاربرد await و async.....
۹۱	مقادیر شمارشی آسنکرون.....
۹۴	دسترسی به نامها.....
۹۷	فصل ششم: آزمایشهای واحد پروژههای ANC.....
۹۷	آماده سازی مقدمات نرم افزاری فصل.....
۹۷	باز کردن پروژه.....
۹۷	تنظیم پورت HTTP.....
۹۸	فعال ساختن MVC.....

۹۸	ایجاد مدل داده.....
۹۹	ایجاد نما و کنترلر.....
۱۰۰	ایجاد پروژه‌ی آزمایش.....
۱۰۱	نوشتن و اجرای کد آزمایش‌های واحد.....
۱۰۳	مرورگر آزمایش ویژوال استدیو.....
۱۰۴	اجرای آزمایش‌ها در ویژوال استدیو کد.....
۱۰۴	اجرای آزمایش‌ها با خط فرمان.....
۱۰۵	رفع خطای آزمایش.....
۱۰۶	جداسازی اجزای کد برای آزمایش واحد.....
۱۰۹	استفاده از نرم‌افزار مقلد.....
۱۱۰	ایجاد شیء Moq.....
۱۱۳	فصل هفتم؛ پروژه‌ی فروشگاه ورزشی.....
۱۱۴	ایجاد پروژه‌ها.....
۱۱۴	ایجاد پروژه‌ی آزمایش واحد.....
۱۱۴	باز کردن پروژه‌ها.....
۱۱۵	تنظیم پورت HTTP.....
۱۱۵	ایجاد پوشه‌های پروژه.....
۱۱۶	سرویس‌ها و زنجیره‌ی درخواست پروژه.....
۱۱۷	پیکربندی موتور نمای Razor.....
۱۱۸	ایجاد نما و کنترلر.....
۱۱۸	ایجاد مدل داده.....
۱۱۹	اجرای برنامه.....
۱۱۹	کار با داده‌ها.....
۱۲۰	نصب Entity Framework Core.....
۱۲۱	تعریف رشته‌ی اتصال.....
۱۲۱	کلاس‌های پایگاه داده.....
۱۲۲	پیکربندی EF Core.....
۱۲۲	ایجاد مخزن داده‌ها.....
۱۲۳	ثبت سرویس مخزن داده‌ها.....
۱۲۴	ایجاد اسکمای پایگاه داده.....
۱۲۵	ایجاد داده‌های نمونه.....
۱۲۶	نمایش لیستی از محصولات.....

۱۲۶	کنترلر.....
۱۲۷	آزمایش واحد کنترلر.....
۱۲۸	آماده کردن نما.....
۱۲۹	اجرای برنامه.....
۱۲۹	صفحه‌بندی داده‌های نما.....
۱۳۰	آزمایش واحد صفحه‌بندی.....
۱۳۱	نمایش لینک‌های صفحه‌ها.....
۱۳۱	بخش مدل نما.....
۱۳۲	کلاس Tag Helper.....
۱۳۳	آزمایش واحد لینک‌ها.....
۱۳۴	داده‌های نما-مدل.....
۱۳۵	آزمایش واحد داده‌های مدل نما.....
۱۳۷	نمایش لینک‌های صفحه‌ها.....
۱۳۷	بهبود URLها.....
۱۳۸	شکل‌دهی نماها.....
۱۳۹	نصب بسته‌ی Bootstrap.....
۱۴۲	ایجاد نمای جزئی.....
۱۴۳	فصل هشتم؛ پیمایش سایت و سبد خرید.....
۱۴۳	کنترل‌های پیمایش.....
۱۴۳	فیلتر کردن محصولات.....
۱۴۵	به روز کردن آزمایش‌های واحد.....
۱۴۶	آزمایش واحد فیلتر محصولات.....
۱۴۷	بازبینی طرح مسیریابی.....
۱۵۰	ایجاد فهرست گروه محصول.....
۱۵۱	لیست گروه محصول.....
۱۵۲	آزمایش واحد لیست گروه محصول.....
۱۵۳	ایجاد نما.....
۱۵۷	آزمایش واحد: گزارش گروه منتخب.....
۱۵۷	آزمایش واحد شمارش محصولات یک گروه.....
۱۵۸	سبد خرید.....
۱۵۹	پیکربندی صفحات Razor.....
۱۶۰	ایجاد صفحه‌ی Razor.....

۱۶۲	استفاده از نشست.....
۱۶۳	تعریف مدل سبد خرید.....
۱۶۳	آزمایش واحد سبد خرید.....
۱۶۶	متدهای توسعه‌یافته برای نشست‌ها.....
۱۶۶	تکمیل صفحه‌ی Razor.....
۱۷۱	فصل نهم؛ تکمیل سبد خرید.....
۱۷۱	بهبود سبد خرید با سرویس.....
۱۷۱	کلاس کمکی سبد خرید.....
۱۷۳	ثبت سرویس کمکی سبد.....
۱۷۴	ساده کردن صفحه‌ی Razor.....
۱۷۵	اصلاح آزمایش‌های واحد.....
۱۷۵	تکمیل کارآیی سبد خرید.....
۱۷۶	حذف کالا از سبد خرید.....
۱۷۸	لیست کالاهای سبد خرید.....
۱۷۸	استفاده از فونت‌های Awesome.....
۱۷۹	ایجاد نما و کلاس عنصر نما.....
۱۸۰	ثبت سفارش.....
۱۸۰	ایجاد کلاس مدل.....
۱۸۱	افزودن فرآیند ثبت سفارش.....
۱۸۲	ایجاد کنترلر و نما.....
۱۸۳	پردازش سفارش.....
۱۸۴	گسترش پایگاه داده.....
۱۸۴	مخزن داده‌های سفارش.....
۱۸۶	تکمیل کنترلر Order.....
۱۸۷	آزمایش واحد پردازش سفارشات.....
۱۸۹	نمایش خطاهای اعتبارسنجی.....
۱۹۰	نمایش صفحه‌ی پایانی.....
۱۹۳	فصل دهم؛ مدیریت پروژه.....
۱۹۳	آماده‌سازی سرور Blazor.....
۱۹۴	ایجاد فایل Startup.....
۱۹۵	ایجاد عناصر مسیریابی و الگو.....
۱۹۶	ایجاد عناصر Razor.....

۱۹۷.....	مدیریت سفارش.....
۱۹۷.....	تغییرات مدل.....
۱۹۸.....	نمایش سفارشات به مدیر.....
۲۰۱.....	مدیریت کالاها.....
۲۰۲.....	مخزن داده‌های کالا.....
۲۰۳.....	عنصر List.....
۲۰۵.....	ایجاد عنصر جزئیات.....
۲۰۶.....	ایجاد عنصر ویرایش.....
۲۰۸.....	حذف محصول.....
۲۱۱.....	فصل یازدهم؛ امنیت و انتشار پروژه.....
۲۱۱.....	پایگاه داده‌ی هویت‌ها.....
۲۱۱.....	بسته‌ی تشخیص هویت.....
۲۱۲.....	تعریف رشته‌ی اتصال.....
۲۱۲.....	پیکربندی پروژه.....
۲۱۳.....	همگام‌سازی پایگاه داده با مدل.....
۲۱۴.....	تعریف داده‌های پایه.....
۲۱۶.....	تعیین سیاست تشخیص هویت.....
۲۱۸.....	کنترلر حساب کاربری و نماهای آن.....
۲۲۱.....	انتشار پروژه.....
۲۲۱.....	پیکربندی مدیریت خطا.....
۲۲۳.....	ایجاد تنظیمات پیکربندی محیط تولید.....
۲۲۴.....	ایجاد تصویر Docker.....
۲۲۴.....	فایل پیکربندی داکر.....
۲۲۵.....	ایجاد تصویر پروژه و انتشار آن.....
۲۲۶.....	اجرای پایانی پروژه‌ی منتشر شده.....
۲۲۷.....	فصل دوازدهم؛ آشنایی با ASP.NET Core.....
۲۲۸.....	آماده‌سازی فصل.....
۲۲۹.....	اجرای پروژه.....
۲۲۹.....	بررسی پلتفرم ANC.....
۲۲۹.....	زنجیره‌ی درخواست و میان‌افزار.....
۲۳۰.....	سرویس‌ها.....
۲۳۱.....	ساختار پروژه‌ی ANC.....

۲۳۲	نقطه‌ی ورودی پروژه.....
۲۳۴	بررسی فایل پروژه.....
۲۳۵	میان‌افزارهای شخصی.....
۲۳۹	تعریف میان‌افزار توسط کلاس.....
۲۴۳	ایجاد شاخه در زنجیره‌ی درخواست.....
۲۴۵	میان‌افزار پایانی.....
۲۴۷	پیکربندی میان‌افزار.....
۲۴۹	پیکربندی میان‌افزار مبتنی بر کلاس.....
۲۵۱	فصل سیزدهم: آشنایی با مسیریابی.....
۲۵۲	آماده کردن پروژه‌ی فصل.....
۲۵۴	فهم مسیریابی آدرس.....
۲۵۵	تعریف میان‌افزار مسیریابی و نقطه‌ی پایانی.....
۲۵۹	آشنایی با الگوی مسیریابی.....
۲۶۰	کاربرد متغیر در مسیر.....
۲۶۲	ترکیب نقطه‌ی پایانی با میان‌افزار.....
۲۶۴	ایجاد URL از روی مسیر.....
۲۶۷	مدیریت تطبیق آدرس‌ها.....
۲۶۷	استخراج چندین مقدار از یک بخش آدرس.....
۲۶۸	مقادیر پیش‌فرض در متغیرهای بخشی.....
۲۶۹	مقادیر اختیاری در الگوی آدرس.....
۲۷۱	کاربرد بخش‌های پوششی.....
۲۷۲	مقیدسازی بخش‌های مسیر.....
۲۷۵	تعریف مسیرهای نهایی.....
۲۷۷	مسیریابی پیشرفته.....
۲۷۷	تعریف قیدهای شخصی.....
۲۷۸	خطا در برخورد با مسیرهای مبهم.....
۲۸۰	دسترسی به نقطه‌ی پایانی در میان‌افزار.....
۲۸۳	فصل چهاردهم: آشنایی با فرآیند تزریق وابستگی.....
۲۸۳	آماده‌سازی پروژه‌ی فصل.....
۲۸۴	ایجاد میان‌افزار و نقطه‌ی پایانی.....
۲۸۵	پیکربندی زنجیره‌ی درخواست.....
۲۸۶	مسئله‌ی محل سرویس.....

۲۸۷	آشنایی با مشکل عناصر وابسته به هم
۲۸۹	استفاده از تزریق وابستگی
۲۹۱	کاربرد سرویس در میان‌افزار
۲۹۲	کاربرد سرویس در نقطه‌ی پایانی
۲۹۷	استفاده از چرخه‌ی عمر سرویس
۲۹۸	سرویس‌های گذرا
۳۰۲	سرویس‌های محدوده
۳۰۴	دسترسی به سرویس‌های محدوده
۳۰۵	نمونه‌ای جدید از نقطه‌ی پایانی برای هر درخواست
۳۰۶	سرویس‌های محدوده در عبارتهای لاند
۳۰۶	ویژگی‌های دیگر در تزریق وابستگی
۳۰۶	وابستگی زنجیری
۳۰۸	دسترسی به سرویس‌ها در Program.cs
۳۰۹	کاربرد توابع سازنده
۳۱۱	ایجاد سرویس با چند پیاده‌سازی
۳۱۳	انواع نامقید در سرویس‌ها
۳۱۵	فصل پانزدهم؛ آشنایی با ویژگی‌های پلتفرم (۱)
۳۱۵	آماده کردن پروژه‌ی این فصل
۳۱۶	سرویس پیکربندی
۳۱۷	فایل پیکربندی محیطی
۳۱۸	دسترسی به تنظیمات پیکربندی
۳۱۹	کاربرد تنظیمات Program.cs
۳۲۰	داده‌های پیکربندی با الگوی انتخابی تنظیمات
۳۲۱	آشنایی با فایل launchSettings.json
۳۲۶	استفاده از سرویس محیط توسعه
۳۲۷	امنیت، ذخیره‌ی داده‌های حساس
۳۳۰	سرویس تهیه‌ی سیاهه (لاگ)
۳۳۱	تولید پیام‌های سیاهه
۳۳۳	پیام‌های سیاهه در Program.cs
۳۳۴	صفت‌های آرایشی در تهیه سیاهه
۳۳۵	تنظیم سطح پایه برای ایجاد سیاهه
۳۳۷	ضبط سیاهه‌ی درخواست‌ها و پاسخ‌ها

۳۳۹	محتوای استاتیک و بسته‌های سمت مشتری
۳۴۰	میان‌افزار محتوای ایستا
۳۴۱	تغییر تنظیمات پیش‌فرض میان‌افزار استاتیک
۳۴۲	کاربرد بسته‌های سمت مشتری
۳۴۴	نصب بسته‌های سمت مشتری
۳۴۵	استفاده از بسته‌ی سمت مشتری
۳۴۷	فصل شانزدهم؛ آشنایی با ویژگی‌های پلتفرم (۲)
۳۴۷	آماده کردن پروژه‌ی این فصل
۳۴۸	استفاده از کوکی‌ها
۳۵۰	فعال شدن دریافت رضایت کاربر
۳۵۲	مدیریت کسب رضایت کاربر
۳۵۵	کاربرد نشست
۳۵۵	بیکربندی سرویس و میان‌افزار نشست
۳۵۸	استفاده از داده‌های نشست
۳۶۰	اتصال‌های HTTPS
۳۶۰	فعال کردن اتصال HTTPS
۳۶۲	کشف درخواست‌های HTTPS
۳۶۳	تحمیل HTTPS به مشتری
۳۶۳	بیکربندی تغییر مسیر HTTPS
۳۶۴	پروتکل امنیت انتقال اطلاعات
۳۶۶	مدیریت استثناها و خطاها
۳۶۷	ایجاد پاسخ مربوط به خطا
۳۷۰	بهبود پاسخ کد وضعیت
۳۷۲	فیلترینگ درخواست
۳۷۵	فصل هفدهم؛ کار با داده‌ها
۳۷۵	آماده کردن پروژه‌ی این فصل
۳۷۶	کش کردن داده‌ها
۳۷۷	کش مقادیر داده‌ها
۳۸۱	استفاده از کش ماندگار و توزیع شده
۳۸۴	کش کردن داده‌های پاسخ
۳۸۶	کاربرد Entity Framework Core
۳۸۶	نصب EF Core

۳۸۷	ایجاد مدل داده
۳۸۸	پیکربندی سرویس پایگاه داده
۳۸۹	ایجاد اسکمای پایگاه داده
۳۹۱	داده‌ها در نقطه‌ی پایانی
۳۹۵	فصل هجدهم: ایجاد پروژه و فضای کار
۳۹۵	تولید فایل‌های پروژه
۳۹۶	مدل داده
۳۹۶	بسته‌ی NuGet
۳۹۷	ایجاد مدل داده
۳۹۸	داده‌های اولیه
۳۹۹	سرویس‌ها و میان‌افزارهای EF Core
۴۰۰	نصب فریم‌ورک CSS
۴۰۰	پیکربندی زنجیره‌ی درخواست
۴۰۱	اجرای برنامه
۴۰۳	فصل نوزدهم: ایجاد وب سرویس‌های REST
۴۰۳	آماده‌سازی پروژه‌ی فصل
۴۰۴	تعریف REST
۴۰۵	آشنایی با JSON
۴۰۵	ایجاد یک وب‌سرویس با API
۴۰۸	ایجاد وب‌سرویس با کنترلر
۴۰۸	فعال کردن MVC
۴۰۹	ایجاد کنترلر
۴۰۹	کلاس پایه‌ی کنترلرها
۴۱۰	صفت‌های کنترلر
۴۱۲	برگشتی متد اکشن
۴۱۲	تزریق وابستگی در کنترلر
۴۱۴	کاربرد مقیدسازی مدل در دسترسی به داده‌ها
۴۱۴	مقیدسازی مدل با بدنه‌ی درخواست
۴۱۶	اکشن‌های حذف و ویرایش
۴۱۷	بهبود کارکرد وب‌سرویس
۴۱۸	کاربرد اکشن‌های آسنکرون
۴۱۹	مشکل مقیدسازی مدل

۴۲۱Action Result کاربرد
۴۲۵تغییر مسیر به متد اکشن
۴۲۶هدایت به آدرسی در سیستم مسیریابی
۴۲۶اعتبارسنجی داده‌ها
۴۲۸ApiController صفت کاربرد
۴۲۹حذف خاصیت‌های پوچ
۴۳۳فصل بیستم؛ وب سرویس پیشرفته
۴۳۳آماده کردن پروژه‌ی فصل
۴۳۵کار با داده‌های مرتبط
۴۳۷شکست ارجاع چرخشی در داده‌های مرتبط
۴۳۸کاربرد متد HTTP PATCH
۴۳۸آشنایی با JSON PATCH
۴۳۹نصب بسته‌ی JSON Patch
۴۴۱آشنایی با قالب‌بندی محتوا
۴۴۱آشنایی با خط مشی پیش فرض محتوا
۴۴۳آشنایی با روش مذاکره بر روی محتوا
۴۴۴قالب‌بندی XML
۴۴۶پذیرش کامل هدرهای Accept
۴۴۷تعیین قالب خروجی اکشن
۴۴۸درخواست قالبی مشخص در URL
۴۵۰محدود کردن قالب‌ها در اکشن
۴۵۱مستندسازی وب سرویس‌ها
۴۵۲نصب و پیکربندی بسته‌ی Swashbuckle
۴۵۵تنظیم دقیق توضیحات API
۴۵۵اجرای API Analyzer
۴۵۷تعیین نوع برگشتی اکشن
۴۵۹فصل بیست و یکم؛ کنترلرها و نماها (۱)
۴۵۹آماده‌سازی پروژه‌ی فصل
۴۶۱شروع کار با نماها
۴۶۱پیکربندی پروژه
۴۶۲ایجاد کنترلر HTML
۴۶۳آشنایی با مسیریابی قراردادی

۴۶۴Razor با نمای
۴۶۵Razor ایجاد نمای
۴۶۶Razor بهبود نمای
۴۶۷انتخاب نما توسط نام آن
۴۶۹استفاده از نماهای مشترک
۴۷۰مشخص کردن محل نما
۴۷۱Razor کار با نماهای
۴۷۴تنظیم نوع مدل نما
۴۷۷View Imports استفاده از فایل نوع
۴۷۹Razor آشنایی با نوشتار
۴۷۹درک فرامین راهنما
۴۸۰درک عبارات محتوا
۴۸۱تعیین محتوای عنصر
۴۸۲تنظیم مقادیر صفات
۴۸۲کاربرد عبارات شرطی
۴۸۵پیمایش دنباله‌ها
۴۸۷Razor استفاده از کدبلاک در
۴۸۹فصل بیست و دوم؛ کنترلرها و نماها (۲)
۴۸۹آماده‌سازی پروژه‌ی فصل
۴۹۰حذف پایگاه داده
۴۹۱View Bag کار با
۴۹۳View Bag محل استفاده از
۴۹۳Temp Data کاربرد
۴۹۵TempData کاربرد صفت
۴۹۵Layout استفاده از
۴۹۷View Bag با Layout بیکربندی
۴۹۸View Start استفاده از فایل
۴۹۹Layout پیش‌فرض لغو
۵۰۲C# انتخاب Layout در کد
۵۰۳Layout بخش‌بندی در
۵۰۵Layout بخش‌های دلخواه در
۵۰۷Layout آزمایش وجود بخش‌های

۵۰۸.....	کاربرد نماهای جزئی.....
۵۰۸.....	فعال کردن نماهای جزئی.....
۵۰۹.....	ایجاد نمای جزئی.....
۵۱۰.....	تعیین مدل داده برای نمای جزئی.....
۵۱۱.....	آشنایی با رمزنگاری محتوا.....
۵۱۲.....	رمزنگاری HTML.....
۵۱۳.....	رمزنگاری JSON.....
۵۱۵.....	فصل بیست و سوم؛ صفحات Razor
۵۱۶.....	آماده‌سازی پروژه‌ی فصل.....
۵۱۷.....	آشنایی با صفحات Razor.....
۵۱۷.....	پیکربندی صفحات Razor.....
۵۱۸.....	ایجاد صفحه‌ی Razor.....
۵۲۰.....	آشنایی با روش مرسوم مسیریابی.....
۵۲۰.....	آشنایی با مدل صفحه.....
۵۲۱.....	آشنایی با نمای صفحه.....
۵۲۲.....	آشنایی با کلاس C# ایجاد شده.....
۵۲۳.....	آشنایی با مسیریابی در صفحات Razor.....
۵۲۴.....	تعریف الگوی مسیریابی در صفحه‌ی Razor.....
۵۲۶.....	افزودن مسیر به صفحه‌ی Razor.....
۵۲۸.....	آشنایی با کلاس مدل صفحه.....
۵۲۸.....	تعریف فایل کلاس پشت صفحه.....
۵۳۱.....	نوع Action Result در صفحات Razor.....
۵۳۵.....	کار با چندین متد HTTP.....
۵۳۷.....	انتخاب متد هندلر.....
۵۳۹.....	آشنایی با نمای صفحه‌ی Razor.....
۵۳۹.....	ایجاد Layout.....
۵۴۰.....	نماهای جزئی در صفحات Razor.....
۵۴۲.....	صفحات Razor بدون مدل.....
۵۴۵.....	فصل بیست و چهارم؛ کامپوننت‌ها
۵۴۶.....	آماده‌سازی پروژه‌ی فصل.....
۵۴۸.....	آشنایی با کامپوننت‌های نما.....
۵۴۸.....	ایجاد و کاربرد کامپوننت نما.....

۵۵۱کاربرد کامپوننت با استفاده از تگ کمکی
۵۵۲ViewComponentResult با نوع
۵۵۳ایجاد نمای جزئی
۵۵۶خروجی HTML
۵۵۸context دریافت داده‌های
۵۵۹داده‌های context از نمای اصلی
۵۶۲استفاده از پارامتر پیش‌فرض
۵۶۳کامپوننت‌های غیرسنکرون
۵۶۴ایجاد فایل‌های ترکیبی کنترلر/کامپوننت
۵۶۶ایجاد نماهای ترکیبی
۵۶۹فصل بیست و پنجم: آشنایی با تگ‌های کمکی
۵۶۹آماده‌سازی پروژه‌ی فصل
۵۷۱ایجاد یک تگ کمکی
۵۷۱ایجاد کلاس تگ کمکی
۵۷۲دریافت اطلاعات عنصر HTML
۵۷۳تولید خروجی
۵۷۴ثبت تگ کمکی
۵۷۵کاربرد تگ کمکی
۵۷۶مدیریت دامنه‌ی کارکرد تگ کمکی
۵۷۷گسترش دامنه‌ی کارکرد تگ کمکی
۵۷۹ویژگی‌های پیشرفته‌ی تگ‌های کمکی
۵۷۹ایجاد عناصر شخصی HTML
۵۸۲ایجاد عناصر با برنامه‌نویسی
۵۸۲جای‌گذاری تگ کمکی در محل مشخص
۵۸۷کار با مدل نما
۵۸۹کار با مدل صفحه
۵۹۲اشتراک داده‌ها بین تگ‌های کمکی
۵۹۳جلوگیری از نمایش عناصر HTML
۵۹۵کامپوننت تگ کمکی
۵۹۷گسترش دامنه‌ی انتخاب عناصر توسط کامپوننت تگ کمکی
۵۹۹فصل بیست و ششم: تگ‌های کمکی پیش‌ساخته
۵۹۹آماده‌سازی پروژه‌ی فصل

۶۰۰	افزودن فایل تصویری.....
۶۰۲	راه‌اندازی تگ‌های پیش‌ساخته.....
۶۰۲	تگ کمکی برای <a>.....
۶۰۴	تگ <a> برای صفحات Razor.....
۶۰۵	ایجاد URL (نه لینک).....
۶۰۶	تگ‌های کمکی جاوااسکریپت و CSS.....
۶۰۷	انتخاب فایل‌های جاوااسکریپت.....
۶۰۹	نکاتی در مورد فایل‌های جاوااسکریپت.....
۶۰۹	محدود کردن الگوی گلوبینگ.....
۶۱۱	کار با شبکه‌های تحویل محتوا.....
۶۱۴	مدیریت شیوه‌نامه‌های CSS.....
۶۱۶	کار با CDNها.....
۶۱۷	کار با عنصر تصویر.....
۶۱۸	حافظه‌ی پنهان داده.....
۶۲۰	زمان انقضای حافظه پنهان.....
۶۲۲	نگارش‌های مختلف داده در حافظه‌ی پنهان.....
۶۲۳	تگ کمکی محیط میزبانی.....
۶۲۵	فصل بیست و هفتم؛ تگ‌های کمکی فرم.....
۶۲۵	آماده‌سازی پروژه‌ی فصل.....
۶۲۷	آشنایی با الگوی مدیریت فرم.....
۶۲۸	کنترلر برای مدیریت فرم.....
۶۳۰	صفحه‌ی Razor برای مدیریت فرم.....
۶۳۱	تگ‌های کمکی در فرم‌ها.....
۶۳۲	کار با عناصر فرم.....
۶۳۲	تعیین کنترلر و اکشن هدف.....
۶۳۴	تبدیل دکمه‌های فرم.....
۶۳۵	کار با عناصر input.....
۶۳۶	صفت type در input.....
۶۳۸	فرمت مقادیر داده‌های فرم.....
۶۴۰	فرمت از طریق کلاس مدل.....
۶۴۲	نمایش مقادیر داده‌های مرتبط.....
۶۴۵	عنصر label.....

۶۴۷	کار با عنصر select.....
۶۴۸	انتشار عنصر select.....
۶۵۰	کار با عنصر TextArea.....
۶۵۱	ویژگی anti-forgery.....
۶۵۳	ویژگی anti-forgery در کنترلر.....
۶۵۴	ویژگی anti-forgery در صفحات Razor.....
۶۵۵	ویژگی anti-forgery در جاوااسکریپت.....
۶۵۹	فصل بیست و هشتم؛ مقیدسازی مدل.....
۶۶۰	آماده‌سازی پروژه‌ی فصل.....
۶۶۱	آشنایی با مقیدسازی مدل.....
۶۶۳	مقیدسازی انواع ساده.....
۶۶۴	مقیدسازی انواع ساده در صفحات Razor.....
۶۶۵	مقادیر پیش‌فرض در مقیدسازی مدل.....
۶۶۸	مقیدسازی انواع پیچیده.....
۶۷۰	مقیدسازی خصوصیات.....
۶۷۱	مقیدسازی انواع تو در تو.....
۶۷۳	تعریف پیشوندهای شخصی.....
۶۷۵	مقیدسازی خصوصیات انتخاب شده.....
۶۷۷	مقیدسازی آرایه و کلکسیون.....
۶۷۷	مقیدسازی آرایه‌ها.....
۶۸۰	مقیدسازی کلکسیون‌ها.....
۶۸۵	منبعی برای مقیدسازی مدل.....
۶۸۸	منبع مقیدسازی برای خاصیت.....
۶۸۸	هدر درخواست به عنوان منبع مقیدسازی.....
۶۹۰	بدنه‌ی درخواست به عنوان منبع مقیدسازی.....
۶۹۱	مقیدسازی به روش دستی.....
۶۹۳	فصل بیست و نهم؛ اعتبارسنجی مدل.....
۶۹۴	آماده‌سازی پروژه‌ی فصل.....
۶۹۶	آشنایی با اعتبارسنجی مدل.....
۶۹۷	اعتبارسنجی داده‌ها.....
۷۰۰	نمایش پیام‌های اعتبارسنجی.....
۷۰۲	اعتبارسنجی ضمنی.....

۷۰۳	اعتبارسنجی صریح.....
۷۰۶	بیکربندی پیام‌های خطای اعتبارسنجی.....
۷۰۹	پیام‌های اعتبارسنجی خاصیت‌ها.....
۷۱۰	پیام‌های سطح مدل.....
۷۱۲	اعتبارسنجی ضمنی در صفحات Razor.....
۷۱۵	تعیین قوانین اعتبارسنجی با متادیتا.....
۷۱۸	ایجاد صفت اعتبارسنجی برای خاصیت‌ها.....
۷۱۹	ایجاد صفت اعتبارسنجی برای مدل.....
۷۲۱	ایجاد صفت اعتبارسنجی برای مدل در صفحه‌ی Razor.....
۷۲۳	اعتبارسنجی سمت مشتری.....
۷۲۵	اعتبارسنجی ترکیبی.....
۷۲۸	اعتبارسنجی ترکیبی در صفحات Razor.....
۷۳۱	فصل سی‌ام؛ کاربرد فیلترها.....
۷۳۲	آماده‌سازی پروژه‌ی فصل.....
۷۳۵	استفاده از فیلترها.....
۷۳۸	کاربرد فیلتر در صفحات Razor.....
۷۳۹	فهم کارکرد فیلترها.....
۷۴۰	ایجاد فیلترهای شخصی.....
۷۴۱	استفاده از فیلترهای اعتبارسنجی.....
۷۴۳	استفاده از فیلتر منابع.....
۷۴۷	فیلترهای اکشن.....
۷۵۱	استفاده از فیلترهای صفحه‌ی Razor.....
۷۵۵	استفاده از متدهای مدل صفحه.....
۷۵۶	استفاده از فیلترهای Result.....
۷۵۷	فیلترهای always-run.....
۷۵۸	ایجاد فیلتری از نوع Result.....
۷۶۰	فیلترهای Exception.....
۷۶۱	ایجاد فیلتری از نوع Exception.....
۷۶۳	مدیریت چرخه‌ی عمر فیلتر.....
۷۶۶	تزریق وابستگی و فیلترها.....
۷۶۷	فیلترهای سراسری.....
۷۶۹	ترتیب اجرای فیلترها.....

۷۷۱	تغییر ترتیب اجرای فیلترها.....
۷۷۳	فصل سی و یکم؛ پروژه‌های فرم.....
۷۷۳	آماده‌سازی پروژه‌ی فصل.....
۷۷۶	ایجاد برنامه‌ی کاربردی MVC.....
۷۷۶	آماده کردن نما و مدل نما.....
۷۷۸	واکشی داده‌ها.....
۷۷۹	ایجاد داده‌ی جدید.....
۷۸۳	ویرایش داده‌ها.....
۷۸۵	حذف داده‌ها.....
۷۸۶	برنامه‌های کاربردی فرم‌ها در صفحات Razor.....
۷۸۸	کارآیی‌های متداول.....
۷۹۰	تعریف صفحه برای عملیات CRUD.....
۷۹۲	ایجاد نمونه‌هایی از داده‌های مرتبط.....
۷۹۶	ایجاد داده‌های جدید.....

پیش‌گفتار مترجم

کتابی که در دست دارید، ویرایش نوزدهم کتاب Pro ASP.NET Core 6 است که در سال ۲۰۲۲ میلادی نگارش یافته است. نویسنده در فصل نخست کتاب در مورد تکامل برنامه‌نویسی سمت سرور مایکروسافت، از پروژه‌های وب ASP.NET تا چرخش مثبتی که به سمت پروژه‌های MVC ایجاد شد و در پایان منجر به پروژه‌های ASP.NET Core شد، توضیح جامعی داده است. خوانندگانی که از پیش در برنامه‌نویسی فرم‌های وب با پروژه‌های ASP.NET سنتی آشنایی داشته‌اند، با مشکلات این پروژه‌ها و دردسرهای آنها در پیاده‌سازی پروژه‌های بزرگ و سازمانی، دست و پنجه نرم کرده‌اند.

خوشبختانه برای یادگیری معماری جدید ASP.NET Core با خواندن این کتاب، نیاز به آشنایی با واسط‌های برنامه‌نویسی قدیمی وب، که به آنها اشاره شد ندارید. به عنوان تنها پیش‌نیاز لازم، آشنایی مقدماتی با مفاهیم وب به همراه توانایی کار با HTML و CSS، زبان #C به همراه Entity Framework و نوشتار کوئری‌های LINQ، کافی است. معنی این گفته این است که می‌توانید برنامه‌نویسی سمت سرور وب را از ابتدا با همین کتاب شروع کنید.

نویسنده، متن اصلی کتاب را به سه بخش اصلی، که می‌توان آنها را مقدماتی تا پیشرفته نامید، تقسیم کرده است. در بخش نخست، از فصل‌های ۱ تا ۱۱، با پیاده‌سازی کامل یک پروژه برخورد خواهید داشت که می‌تواند نقطه‌ی شروع خوبی برای یادگیری ASP.NET Core باشد. همه‌ی کدهای کتاب به راحتی هم در محیط ویژوال استدیو (Visual Studio) و هم در محیط ویژوال استدیو کد (Visual Studio code) که بیشتر مورد کاربرد مهندسين برق و آنهایی که با برنامه‌نویسی میکروکنترلرها و نظایر آنها کار کرده‌اند، آشنا است، اجرا می‌شوند.

بخش دوم، از فصل‌های ۱۲ تا ۱۷، همه‌ی مطالب گفته شده در بخش نخست را با وارد شدن به جزئیات و معرفی ویژگی‌های مهم ASP.NET Core از جمله، سرویس‌ها، کار با داده‌ها و Sql Server، معرفی راه‌کار تزریق وابستگی (Dependency Injection)، مسیریابی پیشرفته، وب‌سرویس‌های REST و بسیاری نکات مهم دیگر، پی می‌گیرد. ولی آنچه در اینجا بیشتر از همه روی آن تأکید شده است، کاربرد صفحات Razor به عنوان همراه یا در مواردی، جایگزینی برای راه‌کار MVC سنتی است که توصیه می‌شود خواننده به خوبی به آن توجه کند.

بخش سوم، از فصل‌های ۱۸ تا ۳۱ کتاب را می‌توان جمع‌بندی همه‌ی مطالب گفته شده در بخش‌های پیش دانست. در اینجا تأکید زیادی بر روی جنبه‌های پیشرفته‌ی وب‌سرویس‌ها، عناصر نما، تگ‌های کمکی، مقیدسازی مدل و بسیاری نکات مهم دیگر شده است که همگی در برنامه‌نویسی پروژه‌های بزرگ از ابزار ضروری محسوب می‌شوند.

کد کامل پروژه‌ی اصلی کتاب و سایر کدهایی که به شکل مثال در فصل‌های مختلف آورده شده‌اند را می‌توانید با رفتن به آدرس زیر به راحتی به دست آورید:

<https://github.com/apress/pro-asp.net-core-6>

در پایان از خوانندگان عزیز تقاضا دارم که پرسش‌ها و مشکلات خود را در صفحه این کتاب در سایت انتشارات به نشانی www.pendarepars.com و یا مستقیماً به آدرس پست الکترونیکی خودم، nabavijobmail@yahoo.com در میان بگذارند.

نادر نبوی

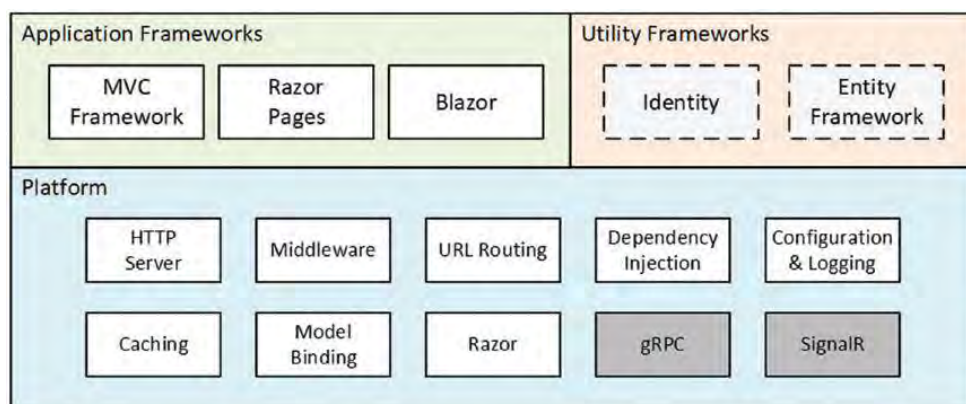
بهار سال ۱۴۰۲

فصل یکم

ASP.NET Core در عمل

ASP.NET Core MVC فریم ورک توسعه‌ی برنامه‌های کاربردی^۱ مایکروسافت است که اثر بخشی و سازماندهی مناسب معماری مدل-نما-کنترلر^۲ را با بهترین بخش‌های NET در هم آمیخته است. ASP.NET برای نخستین بار در سال ۲۰۰۲ عرضه شد و از آن زمان تا کنون در اثر همه تغییرات و تحولاتی که به خود دیده، تبدیل به نگارش ۶ ANC شده، که موضوع بحث این کتاب است.

همان‌گونه که در شکل ۱-۱ می‌بینید، ASP.NET Core (که از این پس در این کتاب آن را ANC خواهیم نامید) شامل پلت فرمی برای پردازش درخواست‌های HTTP، فریم‌ورک‌های اصلی برای ایجاد برنامه‌های کاربردی و فریم‌ورک‌های ثانویه‌ای برای ایجاد ویژگی‌های کمکی و سودمند^۳ مورد نیاز برنامه‌نویس، است.



شکل ۱-۱

آشنایی با فریم‌ورک‌های MVC

در شروع کار با ASP.NET CORE، تعداد زیاد فریم‌ورک‌های موجود می‌تواند گیج‌کننده باشد. همان‌گونه که خواهید دید بسیاری از این فریم‌ورک‌ها تکمیل‌کننده‌ی یکدیگر بوده و مشکلات مختلفی را حل می‌کنند، گرچه برخی از آنها یک مشکل را به روش‌های گوناگون پاسخ می‌دهند.

^۱ Application DevelopmentFramework

^۲ Model-View-Controller (MVC)

^۳ Utility Framework

معرفی فریم‌ورک MVC

MVC مدت‌ها پیش از ASP.NET CORE و NET 6، در دورانی که ASP.NET استفاده می‌شد، ارائه شد. ASP.NET نخستین بار بر پایه روش توسعه صفحات وب (Web Pages) استوار بود که تلاش می‌کرد همان تجربه توسعه‌ی فرم‌های ویندوز را، در مورد صفحات وب بکار گیرد^۱. ولی همان‌طور که می‌دانید این روش کارآمد نبود و مورد استقبال توسعه دهندگان وب قرار نگرفت. فریم‌ورک MVC در همان ایام بر پایه مدلی که بر ویژگی‌های HTTP و HTML استوار بود، ارائه شد.

MVC از الگوی مدل-نما-کنترلر برای ایجاد پروژه‌های وب استفاده می‌کند. الگوی MVC تاکید زیادی بر جداسازی دغدغه‌ها^۲ دارد که بر پایه آن، کارآیی‌های مختلف نرم‌افزار به صورت جداگانه و مستقل پیاده‌سازی می‌شوند. در این روش، کد مربوط به نمایش رابط کاربر در نما آورده می‌شود و مدل نماینده‌ی داده‌های برنامه است. کنترلرها، مسئول پردازش تقاضاهای رسیده و داده‌های مدل و افزون بر این، انتخاب و نمایش نمای مناسب به کاربر هستند. ولی اشکال کار در اینجا بود که نگارش‌های نخستین MVC بر پایه زیرساخت‌های ASP.NET ایجاد شده بودند (که از نظر کارکرد و معماری خیلی با MVC متفاوت است، م) و همین امر موجب پیدایش ویژگی‌های ناکارآمد و مشکل‌سازی شد. با حرکت به سوی ASP.NET Core، ASP.NET تبدیل به ANC شد و فریم‌ورک MVC بر پایه‌ی پلتفرم قابل توسعه‌ی جدید، دوباره از نو نوشته و ایجاد شد.

معرفی صفحات Razor

در برنامه‌های کاربردی MVC ANC، عنصر نرم‌افزاری موتور نما^۳، مسئول تولید محتوایی است که برای کاربران فرستاده می‌شود. موتور پیش‌فرض نما، Razor است که فایل‌های HTML را برای فرامینی که محتوای پویا تولید می‌کنند، پردازش می‌کند.

یکی از مشکلات کار با MVC زمان زیادی است که باید پیش از تولید محتوای اصلی صفحات وب مورد استفاده‌ی کاربر، صرف آماده‌سازی پروژه شود. در حالی که در صفحات قدیمی ASP.NET با وجود همه مشکلاتی که داشتند، ایجاد برنامه‌های کاربردی ساده، بیش از چند ساعت وقت نمی‌برد.

در صفحات Razor، کد (مثلاً به C#) با محتوای صفحه (HTML) در هم آمیخته می‌شوند. این روش همان سرعت ایجاد صفحات وب پیشین را بدون مشکلات خاص ASP.NET منسوخ شده، فراهم می‌کند. در این کتاب، صفحات Razor را در کنار MVC به کار خواهیم برد. در این روش، بخش‌های اصلی برنامه را با MVC نوشته و برای اهداف ثانویه مانند گزارش‌گیری، از صفحات Razor استفاده خواهیم کرد.

^۱ این به معنی سعی در پنهان کردن ویژگی‌های اصلی HTML و پروتکل HTTP بود که مهمترین آن‌ها بدون وضعیت، یا stateless بودن است. از این روی استفاده‌ی بیش از حد از کنترل‌هایی شبیه کنترل‌های ویندوز و سعی بر انتقال وضعیت این کنترل‌ها، در حجم وسیعی از داده، از یک صفحه به صفحه‌ی دیگر، عملاً کار با ASP.NET را بسیار مشکل کرده بود.

^۲ separation of concerns

^۳ View Engine

معرفی Blazor

پیشرفت و کاربرد روزافزون فریم‌ورک‌های سمت مشتری^۱ جاوا اسکریپت، از آنجا که برنامه‌نویسان را مجبور به یادگیری زبان جدیدی می‌کند، چالشی برای برنامه‌نویسان C# محسوب می‌شود. یادگیری زبانی که از نظر نوشتار و ساختار تفاوت‌های بسیاری با C# به عنوان زبان اصلی برنامه‌نویسی دارد، کار ساده‌ای نیست.

راه حل Blazor برای این مشکل، ایجاد امکان استفاده از C# برای برنامه‌نویسی سمت مشتری است. Blazor دارای دو نگارش به نام‌های Blazor Server سرور بلیزر و Blazor WebAssembly بلیزر وب است. سرور بلیزر بخشی از ANC بوده و با استفاده از ارتباطی که با سرور ANC ایجاد می‌کند (از طریق HTTP) کار می‌کند. همان‌گونه که می‌دانید سرور گفته شده دقیقاً همان جایی است که کدهای سمت سرور C# اجرا می‌شوند. بلیزر وب که هنوز در مرحله‌ی تجربه و آزمایش قرار دارد، گامی فراتر رفته و سعی می‌کند کدهای C# را در مرورگر کاربر (یعنی سمت مشتری) اجرا کند.

معرفی فریم‌ورک کمکی^۲

دو فریم‌ورک دیگر که به صورت تنگاتنگی با Core کار می‌کنند، عبارتند از Entity Framework و ASP.NET Identity. همان‌گونه که می‌دانید، Entity Framework نرم‌افزار نگاشت شیء-گرا-رابطه‌ای میکروسافت برای ارائه‌ی داده‌های ذخیره شده در یک مدل رابطه‌ای (مثل SqlServer)، به صورت شیء‌گرا است^۳. از این فریم‌ورک در همه‌ی محیط‌های NET می‌توان استفاده کرد. در این کتاب، از آن برای دسترسی به پایگاه داده، در پروژه‌های Core استفاده خواهیم کرد.

فریم‌ورک دوم، Entity، فریم‌ورکی است که برای اعتبارسنجی داده‌های لاگین کاربران، تعیین و محدودسازی سطوح دسترسی، مورد استفاده قرار می‌گیرد و در این کتاب هم دارای همین کاربرد است.

معرفی پلتفرم ANC

پلتفرم ANC شامل ویژگی‌های لازم برای دریافت و پردازش درخواست‌های HTTP و به دنبال آن، تولید پاسخ مناسب است. از مهم‌ترین اینها می‌توان از یک سرور HTTP، یک سیستم نرم‌افزاری میانی^۴ برای پردازش درخواست‌های رسیده و سایر ویژگی‌هایی که فریم‌ورک برنامه به آنها نیازمند است، مانند مسیریابی^۵ URL و موتور نمای^۶ Razor نام برد.

^۱ Client side

^۲ Utility Framework

^۳ ORM یا نرم‌افزاری که جداول رابطه‌ای پایگاه داده‌ای مانند SQL-Server را به شکل کلاس‌هایی که در یک سیستم شیء‌گرا از یکدیگر مشتق می‌شوند، در اختیار محیط برنامه‌نویسی NET قرار می‌دهد.

^۴ Middleware

^۵ URL Routing

^۶ Razor View Engine

سخنی در مورد ساختار کتاب

برای بهره‌برداری بهینه از کتاب، خواننده باید ضمن آشنایی با مفاهیم پایه‌ی توسعه‌ی وب، با کارکرد HTML و CSS آشنا بوده و علاوه بر اینها، دارای دانشی عملی در کار با #C باشد. از آنجا که تاکید کتاب بر کارکرد زبان #C با ANC است، تجربه‌ی کار با برنامه‌های توسعه‌ی سمت مشتری مانند JavaScript مورد نیاز نیست.

نرم‌افزار مورد نیاز برای مثال‌های کتاب

برای پیاده‌سازی مثال‌های کتاب (که اکیدا توصیه می‌شود) نیاز به یک ویرایشگر کد (خود ویزوال استدیو و یا نرم‌افزار Visual Studio Code)، کیت توسعه‌ی NET Core^۱ و نگارش LocalDB از Sqlserver خواهید داشت. همه‌ی موارد گفته شده، از سایت مایکروسافت بدون پرداخت هزینه‌ای قابل دسترس هستند (و در زمان ترجمه‌ی کتاب، بدون نیاز به VPN و با آی پی ایران قابل دانلود هستند).

کتاب برای ویندوز نوشته شده است. خود من ویندوز ۱۰ را بکار بردم ولی هر نوع نگارشی از ویندوز که ویزوال استدیو و NET Core در آن قابل اجرا باشند، می‌تواند مورد استفاده قرار گیرد. ANC می‌تواند بر روی پلتفرم‌های دیگری غیر از ویندوز اجرا شود، ولی بیشتر مثال‌های کتاب به SQL Server LocalDB نیاز دارند، که مختص ویندوز است.

مطالب ارائه شده در کتاب

بخش نخست کتاب، شامل فصل‌های ۱ تا ۱۱، به معرفی ANC می‌پردازد. افزون بر آماده‌سازی محیط برنامه‌نویسی و توسعه‌ی مورد نیاز و ایجاد اولین پروژه، با مهمترین ویژگی‌های #C در کار با ANC آشنا خواهید شد. بخش زیادی از این فصل‌ها به پیاده‌سازی پروژه‌ای به نام فروشگاه ورزشی (SportsStore Project) می‌پردازد که در طی آن از مراحل آغازین طراحی مفاهیم پروژه، تا پیاده‌سازی و انتشار آن، با ویژگی‌های اصلی و مهم ANC و چگونگی ترکیب این ویژگی‌ها با هم و استفاده از آنها، آشنا خواهید شد.

در بخش دوم که شامل فصل‌های ۱۲ تا ۱۷ می‌شود، وارد جزئیات ویژگی‌های اصلی ANC خواهیم شد. ضمن آشنایی با چگونگی پردازش درخواست‌های HTTP، موضوعاتی مانند ایجاد و بکارگیری عناصر میانی (Middleware Components)، ایجاد مسیره‌ها و مسیریابی (Routes)، تعریف و استفاده از سرویس‌ها و در پایان، کار با Entity Framework Core (که از این پس، به اختصار EF Core نامیده خواهد شد) را مورد بررسی قرار خواهیم داد. و در آخر در بخش سوم، فصل‌های ۱۸ تا پایان کتاب، با چگونگی ایجاد انواع مختلف پروژه آشنا خواهید شد. مواردی مانند وب سرویس‌های REST (RESTful web services)، کاربرد Razor و کنترلرها در تهیه‌ی پروژه‌های HTML و ویژگی‌های مهمی مانند نماها (Views) و عناصر آنها (View Components) و تگ‌های کمکی (Tag Helpers) بررسی خواهند شد.

^۱ .Net Core software development kit

فصل دوم

شروع به کار

بهترین راه برای آشنایی با یک فریم‌ورک توسعه‌ی نرم‌افزار، بکارگیری آن است. در این فصل با چگونگی آماده‌سازی و ایجاد یک محیط توسعه‌ی ANC و نحوه‌ی اجرای آن آشنا خواهید شد.

انتخاب ویرایشگر کد و محیط توسعه

مایکروسافت دو ابزار اصلی برای کدنویسی معرفی می‌کند؛ ویژوال استدیو و ویژوال استدیو کد (Visual Studio Code). ویژوال استدیو ابزار معمول توسعه‌ی برنامه‌های NET است که طیف وسیعی از راهکارها و ویژگی‌های مختلف را برای انواع برنامه‌های کاربردی NET در اختیار قرار می‌دهد. مشکل آن، مصرف زیاد منابع (حافظه و پردازش) و کند بودن آن است. برخی از ویژگی‌های آن هم، گرچه برای کمک به برنامه‌نویس ایجاد شده‌اند، ولی می‌توانند مانعی بر سر راه توسعه باشند.

"ویژوال استدیو کد" نگارش سبکی از ویژوال استدیو است که همه‌ی امکانات لازم برای توسعه‌ی ANC را در اختیار دارد.

همه‌ی مثال‌های این کتاب دارای دستورات لازم برای بکارگیری هر یک از ادیتورهای گفته شده هستند و شما می‌توانید هر کدام را که ترجیح می‌دهید، نصب و استفاده کنید.

ویژوال استدیو ابزار بیشتری برای ایجاد انواع فایل‌های مورد نیاز NET Core در اختیار می‌گذارد، بنابراین این اگر در برنامه‌نویسی NET مبتدی هستید، بهتر است که ویژوال استدیو را نصب کنید. به این ترتیب می‌توانید مطمئن باشید که به نتایج مورد نظر هر یک از مثال‌ها، به درستی، دست خواهید یافت.

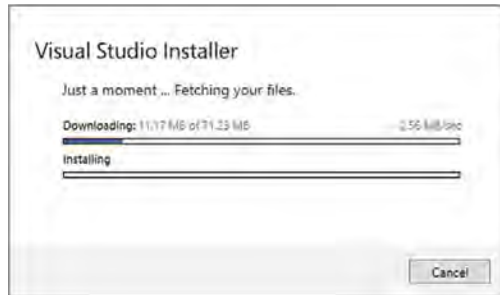
نصب ویژوال استدیو

برای 6 ANC نیاز به ویژوال استدیو ۲۰۲۲ دارید. در این کتاب از نگارش مجانی ویژوال استدیو (نگارش کامیونیتی Community Edition) استفاده شده که می‌توانید آن را از سایت www.visualstudio.com (با آی پی ایران و بدون نیاز به وی پی ان، در زمان ترجمه‌ی کتاب) دانلود کنید. پس از اقدام به دانلود و اجرای نصاب، با پنجره‌ی شکل ۲-۱ روبرو خواهید شد:



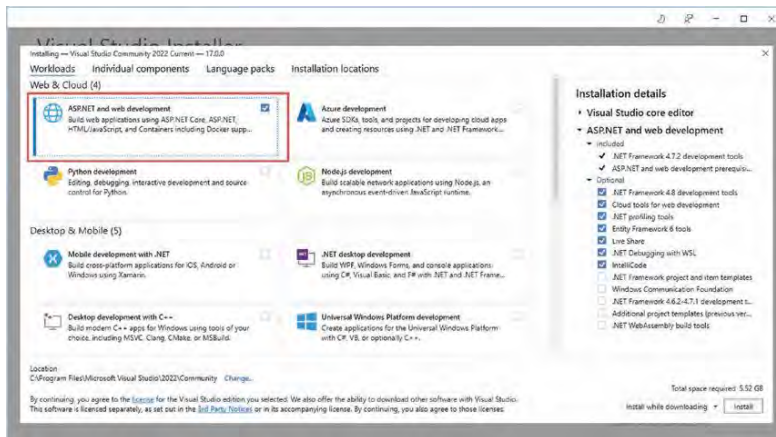
شکل ۲-۱

با کلیک بر روی "Continue" عمل دانلود فایل‌های لازم، مانند شکل ۲-۲، شروع خواهد شد:



شکل ۲-۲

با تکمیل دانلود، با گزینه‌هایی برای نصب روبرو می‌شوید. مطمئن شوید که مانند شکل ۳-۲، گزینه‌ی "ASP.NET and web development" حتما انتخاب شده باشد.



شکل ۳-۲

با توجه به شکل ۲-۴، با انتخاب بخش "Individual components"، حتما گزینه‌ی SQL Server Express 2019 LocalDB را انتخاب کنید. از این نرم‌افزار برای ذخیره‌سازی داده‌ها در مثال‌های آتی کتاب استفاده خواهیم کرد.



شکل ۴-۲

در پایان، کلیک بر روی دکمه‌ی install موجب دانلود فایل‌های لازم و نصب آنها خواهد شد. امکان دارد که برای تکمیل نصب، نیاز به ریست شدن کامپیوتر داشته باشید.

نصب .NET SDK

امکان دارد نگارشی از SDK که توسط نصب‌کننده‌ی ویژوال استدیو نصب می‌شود، همانی نباشد که مورد نیاز مثال‌های این کتاب است. بنابراین به آدرس زیر رفته و نگارش SDK 6.0.0 را دانلود و نصب کنید: <https://dotnet.microsoft.com/download/dotnet-core/6.0> پس از نصب، پنجره‌ای جدید برای Command prompt power shell ویندوز باز کرده و فرمان زیر را در آن وارد کنید:

```
dotnet --list-sdks
```

این فرمان، لیستی از SDK های نصب شده را نمایش می‌دهد. لیست زیر، نمایشی از ویندوزی است که .NET برای نخستین بار در آن نصب شده است:

```
6.0.100 [C:\Program Files\dotnet\sdk]
```

اگر از پیش، با نگارش‌های مختلفی از .NET کار کرده باشید، امکان دارد لیست طولانی‌تری ببینید:

```
3.1.101 [C:\Program Files\dotnet\sdk]
```

```
5.0.100 [C:\Program Files\dotnet\sdk]
```

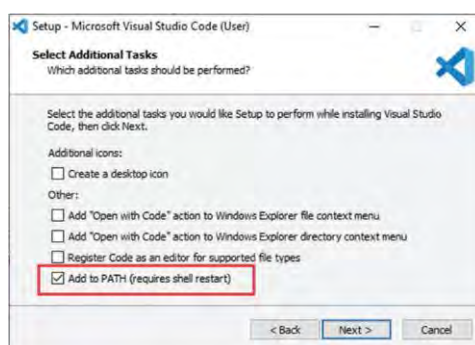
```
5.0.401 [C:\Program Files\dotnet\sdk]
```

```
6.0.100 [C:\Program Files\dotnet\sdk]
```

صرف نظر از هر تعداد SDK نصب شده، مطمئن شوید که حتما دارای نگارش 6.0.1xx هستید.

نصب Visual Studio code

در صورتی که تصمیم بر استفاده از Visual Studio Code دارید، برنامه‌ی نصب آن را از سایت <https://code.visualstudio.com> دانلود کنید. با اجرای برنامه‌ی نصب، مطمئن باشید که گزینه‌ی Add to PATH مانند شکل ۲-۵ انتخاب شده باشد:



شکل ۲-۵

از آنجا که "ویژوال استدیو کد" دارای .NET SDK نیست، باید آن را بطور جداگانه دانلود کنید. با رفتن به آدرس <https://dotnet.microsoft.com/download/dotnet-core/6.0> و انتخاب نگارش 6.0.0 آن را

دانلود کنید. پس از اجرای برنامه‌ی نصب کننده و پایان کار آن، پنجره‌ای جدید از خط فرمان ویندوز (PowerShell) باز کرده و فرمان زیر را در آن اجرا کنید:

```
dotnet --list-sdks
```

این فرمان، لیستی از SDK های نصب شده را نمایش می‌دهد. لیست زیر، نمایشی از ویندوزی است که .NET برای نخستین بار در آن نصب شده است:

```
6.0.100 [C:\Program Files\dotnet\sdk]
```

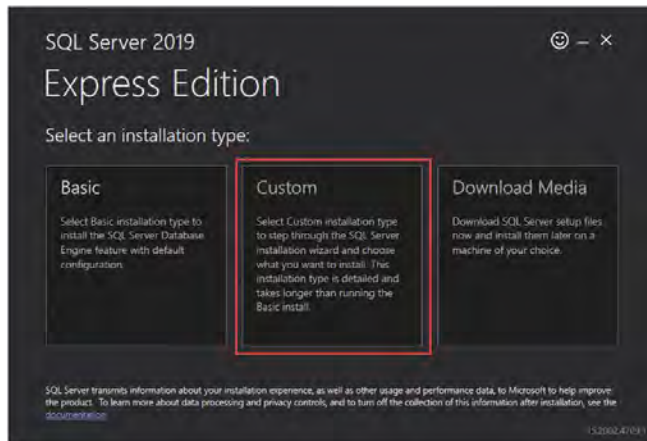
اگر از پیش با نگارش‌های مختلفی از .NET کار کرده باشید، امکان دارد لیست طولانی‌تری ببینید:

```
3.1.101 [C:\Program Files\dotnet\sdk]
5.0.100 [C:\Program Files\dotnet\sdk]
5.0.401 [C:\Program Files\dotnet\sdk]
6.0.100 [C:\Program Files\dotnet\sdk]
```

در اینجا هم، صرف نظر از هر تعداد SDK نصب شده، مطمئن شوید که حتما دارای نگارش 6.0.1xx هستید.

نصب SQL Server LocalDB

مثال‌های پایگاه داده‌ی این کتاب از LocalDB، که بخشی از نگارش SQL Express از SQL Server، با کمترین پیکربندی است، استفاده می‌کنند. این نرم‌افزار، به صورت کاملاً مجانی از آدرس <https://www.microsoft.com/en-in/sql-server/sql-server-downloads> قابل دانلود است. همان‌گونه که در شکل ۲-۶ می‌بینید، هنگام دانلود از بخش "Custom" استفاده کنید:



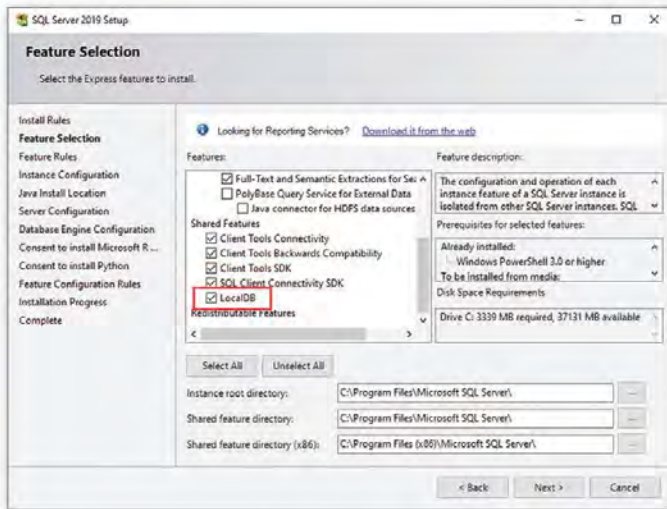
شکل ۲-۶

در ادامه، باید محلی را برای دانلود فایل‌های نصب شده بر روی کامپیوتر خود انتخاب کنید. پس از کلیک بر روی دکمه‌ی "Install"، عمل دانلود شروع می‌شود. وقتی پنجره‌ای مانند شکل ۲-۷ ظاهر شد، گزینه‌ی نخست را برای ایجاد وهله‌ی جدیدی از SQL Server، انتخاب کنید.



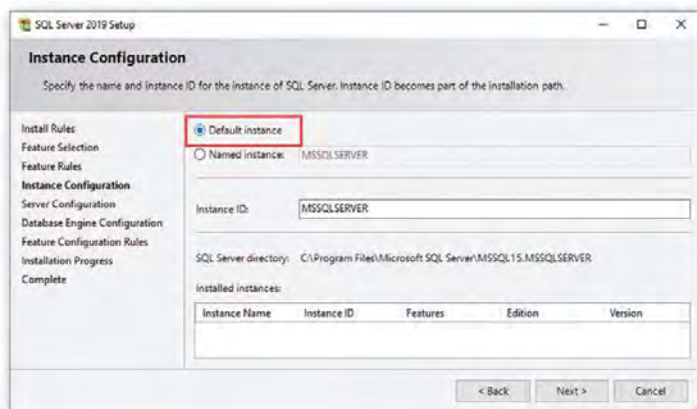
شکل ۲-۷

در ادامه، همه‌ی گزینه‌های پیش‌فرض را همانطور که نشان داده می‌شوند، قبول کنید. هنگامی که به پنجره‌ی انتخاب ویژگی‌های مورد نظر برای نصب، مانند شکل ۲-۸ می‌رسید، مطمئن باشید که گزینه‌ی مربوط به LocalDB انتخاب شده باشد. افزون بر این، ممکن است بخواهید گزینه‌های مربوط به R و Python را از حالت انتخاب خارج کنید که به هر حال، کاربردی در این کتاب نخواهند داشت (و البته به زمان زیادی برای دانلود و نصب نیاز دارند).



شکل ۲-۸

در صفحه‌ی انتخاب چگونگی پیکربندی و هله‌ی نصب شده، مانند شکل ۲-۹، گزینه‌ی "Default instance" را انتخاب کنید:



شکل ۲-۹

در پنجره‌های بعدی، گزینه‌های پیش‌فرض را قبول کنید. پس از پایان نصب، آخرین آپدیت SQL Server را دانلود و نصب کنید. در هنگام نگارش کتاب، آخرین آپدیت در آدرس زیر قابل دسترس است:

<https://support.microsoft.com/en-us/topic/kb5005679-cumulative-update-13-for-sql-server-2019-c1be850-460a-4be4-a569-fe11f0adc535>

البته دسترسی به این آدرس، با جستجوی KB5005679 در جستجویی مانند گوگل شاید راحت‌تر باشد. دقت کنید که در زمان مطالعه‌ی این کتاب، ممکن است نگارش‌های جدیدتری از این آپدیت در دسترس باشد.

ایجاد یک پروژه‌ی ANC

سراسرترین راه برای ایجاد پروژه، استفاده از خط فرمان است. پس از باز کردن یک پنجره‌ی جدید خط فرمان ویندوز (Powershell) و حرکت به درایو و پوشه‌ی مورد نظرتان برای ایجاد پروژه، فرامین زیر را مانند لیست ۲-۳ وارد کنید:

لیست ۲-۳ کد ایجاد پروژه و سالوشن

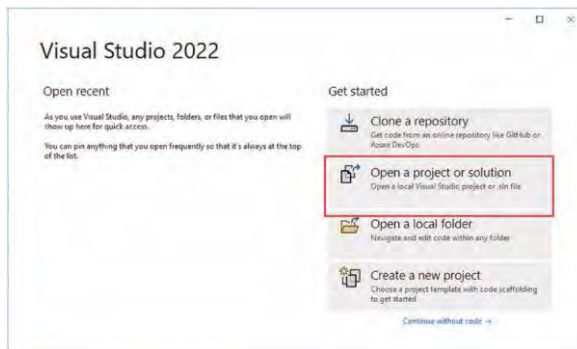
```
dotnet new globaljson --sdk-version 6.0.100 --output FirstProject
dotnet new mvc --no-https --output FirstProject --framework net6.0
dotnet new sln -o FirstProject
dotnet sln FirstProject add FirstProject
```

نخستین فرمان، پس از ایجاد پوشه‌ی FirstProject، فایل‌ی به نام global.json به همراه نگارشی از .NET که به کار خواهد رفت (۶.۰.۱۰۰)، را به آن اضافه می‌کند. این کار موجب حصول اطمینان از دستیابی به نتایج درست در اجرای مثال‌های کتاب خواهد شد. فرمان دوم، پروژه‌ی جدیدی به نام FirstProject، با استفاده از الگوی MVC ایجاد می‌کند. الگوی (Template) MVC، یکی از چندین الگوی ارائه شده برای ایجاد پروژه‌های جدید در ANC است. این الگو، پروژه‌ای ایجاد می‌کند که به طرز مناسبی برای فریم‌ورک MVC

پیکربندی شده است. اگر در حال حاضر چیزی از MVC یا به طور کلی الگوهای مختلف ASP.NET نمی‌دانید، نگران نباشید؛ تا پایان کتاب با جزئیات کار MVC به طور کامل آشنا خواهید شد. دو فرمان بعدی به ترتیب، ابتدا یک سالوشن ایجاد کرده و پس از آن، پروژه را به سالوشن اضافه می‌کنند (اسامی سالوشن و پروژه یکسان هستند).

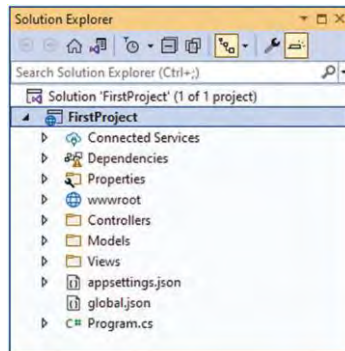
باز کردن پروژه در ویژوال استدیو

پس از اجرای ویژوال استدیو، مانند شکل ۲-۱۰، بر روی "Open a project or solution" کلیک کنید:



شکل ۲-۱۰

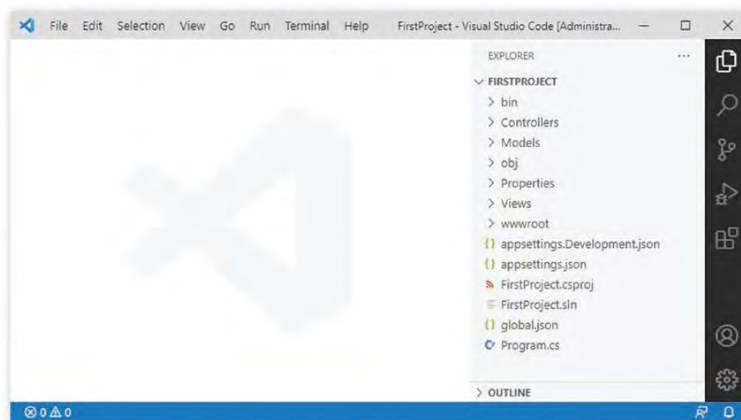
پس از ورود به پوشه‌ی FirstProject و انتخاب فایل FirstProject.sln (فایل سالوشن پروژه)، آن را با کلیک بر روی دکمه‌ی Open، باز کنید. ویژوال استدیو، پروژه را مانند شکل ۲-۱۱ باز کرده و محتویات آن را نمایش می‌دهد. همان‌گونه که قبلاً گفته شد، فایل‌های این پروژه بر پایه‌ی الگوی قالب MVC ایجاد شده‌اند.



شکل ۲-۱۱

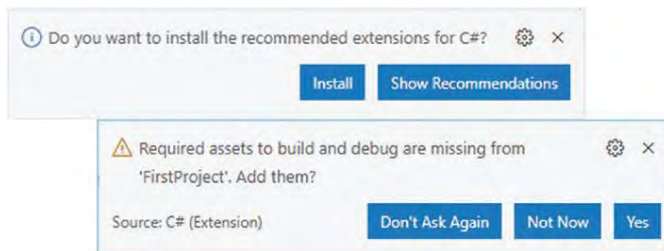
به عنوان روشی دیگر، برای این که پروژه را با ادیتور "ویژوال استدیو کد" باز کنید، پس از اجرای ادیتور، از منوی File گزینه‌ی Open Folder را انتخاب کنید. سپس به پوشه‌ی FirstProject بروید و در

پایان، بر روی دکمه‌ی Select Folder کلیک کنید. پروژه مانند شکل ۲-۱۲ در محیط ادیتور باز شده و محتویات آن نمایش داده می‌شود.



شکل ۲-۱۲

پروژه‌ای که برای نخستین بار در "ویژوال استدیو کد" باز می‌شود، نیاز به پیکربندی‌های دیگری هم دارد. به عنوان اولین گام، بر روی فایل Program.cs در پنجره‌ی کاوشگر (Explorer Pane) کلیک کنید تا باز شود. این کار، موجب نمایش پنجره‌ای مبنی بر درخواست تأیید برای افزودن ویژگی‌های مورد نیاز پروژه، مانند شکل ۲-۱۳ می‌شود. اگر هیچ پروژه‌ی C# تاکنون باز نکرده باشید، پنجره‌ای هم برای نصب ویژگی‌های لازم برای محیط C# باز خواهد شد (پنجره‌ی نخست در شکل ۲-۱۳).



شکل ۲-۱۳

آن طور که مناسب می‌دانید، بر روی دکمه‌ی Install یا Yes کلیک کنید. ویژوال استدیو اقدام به دانلود و نصب موارد مورد نیاز خواهد کرد.

اجرای برنامه ANC

اجرای برنامه‌ها، هم از طریق ویژوال استدیو و هم در محیط "ویژوال استدیو کد"، به طور مستقیم امکان‌پذیر است. با این حال در این کتاب از خط فرمان استفاده می‌کنیم که هم قابل اطمینان‌تر بوده و هم

این که از نظر آموزشی کارآیی بیشتری دارد. در آینده، پس از یادگیری مطالب، خودتان به هر روشی که برایتان راحت تر است، برنامه‌ها را اجرا خواهید کرد.

در زمان ایجاد پروژه، فایلی هم به نام LaunchSettings.json در پوشه‌ی Properties ایجاد می‌شود. محتویات این فایل تعیین کننده‌ی پورت HTTP است که برای گوش سپردن به درخواست‌های HTTP به کار خواهد رفت. پس از باز کردن این فایل با دو بار کلیک بر روی نام آن، در پنجره‌ی کاوشگر، شماره‌ی پورت گفته شده را با توجه به لیست ۲-۴، به ۵۰۰۰ تغییر دهید:

لیست ۲-۴

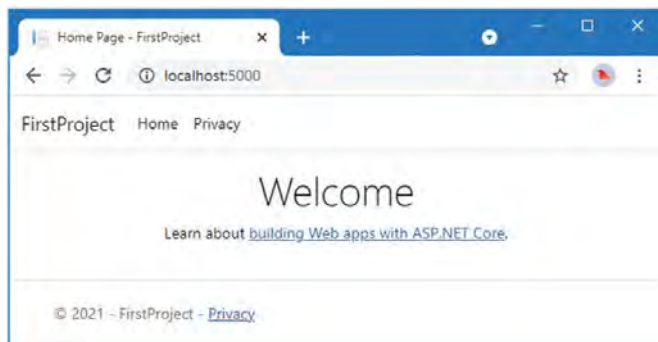
```
{
  "iisSettings": {
    "windowsAuthentication": false,
    "anonymousAuthentication": true,
    "iisExpress": {
      "applicationUrl": "http://localhost:5000",
      "sslPort": 0
    }
  },
  "profiles": {
    "FirstProject": {
      "commandName": "Project",
      "dotnetRunMessages": true,
      "launchBrowser": true,
      "applicationUrl": "http://localhost:5000",
      "environmentVariables": {
        "ASPNETCORE_ENVIRONMENT": "Development"
      }
    },
    "IIS Express": {
      "commandName": "IISExpress",
      "launchBrowser": true,
      "environmentVariables": {
        "ASPNETCORE_ENVIRONMENT": "Development"
      }
    }
  }
}
```

گرچه فقط آدرس موجود در بخش "profiles" بر روی ابزار خط فرمان NET. موثر است، در اینجا برای اطمینان بیشتر، هر دو مورد را تغییر داده‌ایم. پس از باز کردن پنجره‌ی جدید فرمان ویندوز (Powershell)، از طریق منوی شروع "start" ویندوز، به پوشه‌ی FirstProject که شامل فایل پروژه‌ی FirstProject (.csproj) است بروید و فرمان لیست ۲-۵ را برای اجرای برنامه وارد کنید.

لیست ۲-۵

```
dotnet run
```

فرمان dotnet run پروژه‌ی موجود در پوشه‌ی جاری را کامپایل و اجرا می‌کند. پس از اجرای برنامه، پنجره‌ی مرورگر را باز کنید و آدرس http://localhost:5000 را در نوار آدرس وارد کنید. این درخواست، پاسخ نشان داده شده در شکل ۲-۱۴ را ایجاد می‌کند:



شکل ۲-۱۴

در پایان، فشردن کلیدهای **Control+C** اجرای برنامه را متوقف می‌کند.

آشنایی با Endpoint

در برنامه‌های ANC درخواست‌های ورودی توسط نقاط پایانی (Endpoints)، دریافت و پردازش می‌شوند. نقطه‌ی پایانی تولیدکننده‌ی پاسخ مندرج در لیست ۲-۴، یک اکشن (Action) است. این اکشن، به عنوان یک متد استاندارد، به زبان C# نوشته شده است. هر اکشن در یک کنترلر (Controller) نوشته می‌شود که خود کلاسی مشتق شده از کلاس پایه‌ی `Microsoft.AspNetCore.Mvc.Controller` است.

به طور کلی، هر متد عمومی نوشته شده در یک کنترلر، یک اکشن است. این اکشن را می‌توان برای پردازش یک درخواست HTTP، فراخوانی کرد.^۱ روش معمول در پروژه‌های ANC این است که کلاس‌های کنترلر را در پوشه‌ای به نام `Controllers` قرار می‌دهند. در مثال حاضر، این پوشه از همان ابتدا توسط الگوی MVC که برای پروژه انتخاب کردید، ساخته شده است.

در اینجا نام کلاس کنترلر، `HomeController.cs` و نام خود کنترلر، `Home` است. بنابراین، فایل‌های کلاس‌های کنترلر، از یک نام دلخواه (در این مورد `Home`) و به دنبال آن کلمه‌ی `Controller` به صورت چسبیده به هم، استفاده می‌کنند. باید اشاره کرد که کنترلر `Home`، کنترلر پیش‌فرض در پروژه‌های ANC است.

اکنون فایل `HomeController.cs` را در پنجره‌ی کاوشگر ویژوال استدیو (یا ویژوال استدیو کد) پیدا، و بر روی آن کلیک کنید تا محتوایش مانند لیست ۲-۵ نمایش داده شود:

لیست ۲-۵

```
using System.Diagnostics;
using Microsoft.AspNetCore.Mvc;
using FirstProject.Models;
```

^۱ ساده‌ترین درخواست HTTP، وارد کردن آدرس یک صفحه توسط کاربر است. در این وضعیت، اکشنی که به صورت یک متد در یک کنترلر نوشته شده، در ابتدایی‌ترین حالت، محتوای صفحه‌ی مورد نظر را نمایش خواهد داد.

فصل ۲: شروع به کار / ۱۵

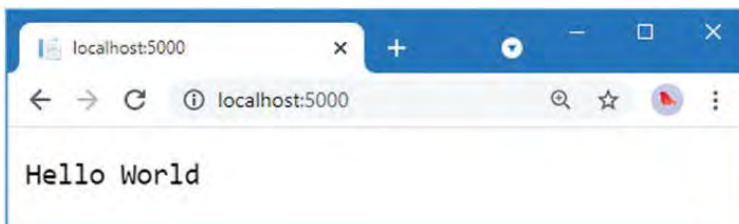
```
namespace FirstProject.Controllers;
public class HomeController : Controller {
    private readonly ILogger<HomeController> _logger;
    public HomeController(ILogger<HomeController> logger) {
        _logger = logger;
    }
    public IActionResult Index() {
        return View();
    }
    public IActionResult Privacy() {
        return View();
    }
    [ResponseCache(Duration = 0, Location = ResponseCacheLocation.None,
    NoStore = true)]
    public IActionResult Error() {
        return View(new ErrorViewModel { RequestId = Activity.Current?.Id
        ?? HttpContext.TraceIdentifier });
    }
}
```

در گام بعد، محتوای این کلاس را با آنچه که در لیست ۲-۶ می‌بینید، تغییر دهید. در اینجا، همراه با حذف عبارتهای using اضافی و همهی متدها به غیر از یکی از آنها، نوع خروجی متد و محتوای آن هم تغییر کرده است:

لیست ۲-۶

```
using Microsoft.AspNetCore.Mvc;
namespace FirstProject.Controllers {
    public class HomeController : Controller {
        public string Index() {
            return "Hello world";
        }
    }
}
```

در پی این تغییرات، در کنترلر Home تنها یک متد اکشن به نام Index تعریف شده است. این متد، عبارت Hello World را باز می‌گرداند. اکنون با استفاده از همان روش خط فرمان، پس از اجرای برنامه در پوشه‌ی FirstProject، آدرس `http://localhost:5000` را در مرورگر وارد کنید. این درخواست شما، توسط متد اکشن Index در کنترلر Home پردازش و پاسخ داده می‌شود. رشته‌ی تولید شده توسط این متد، Hello World، پاسخ به درخواست HTTP است (شکل ۲-۱۵).



شکل ۲-۱۵

آشنایی با مفهوم مسیر^۱

اینک این سوال مطرح می‌شود که در پاسخ یک درخواست، کدام یک از اکشن‌ها، که درون کنترلرها دسته‌بندی شده‌اند، اجرا شود؟ در ANC سیستم مسیریابی^۲ مسئول تعیین نقطه‌ی پایانی یا همان اکشنی است که باید به درخواست رسیده پاسخ دهد. مسیر یا Route، قانونی است که چگونگی پاسخ به درخواست را مشخص می‌کند. در هنگام ایجاد پروژه، یک مسیر پیش‌فرض برای شروع کار ایجاد شده است. با درخواست هر یک از URLهای زیر، همان اکشن Index در کنترلر Home، اجرا خواهد شد:

```
/
/Home
/Home/Index
```

بنابراین، وقتی مرورگری تقاضاهایی مانند http://yoursite/ یا http://yoursite/home را ارسال می‌کند، خروجی را از متد Index در کلاس HomeController دریافت می‌کند. این وضعیت را می‌توانید با تغییر آدرس مرورگر امتحان کنید. در حال حاضر، این آدرس http://localhost:5000 است که البته اگر از ویژوال استدیو استفاده می‌کنید، شماره‌ی پورت ممکن است متفاوت باشد. اگر رشته‌های /Home یا /Home/Index را به انتهای آدرس اضافه کنید، همان نتیجه‌ی Hello World را مشاهده خواهید کرد.

چگونگی پردازش HTML

خروجی مثال پیش فقط یک رشته متنی (Hello World) بود، این در حالی است که در بیشتر مواقع، خروجی متد اکشن باید صفحه‌ی HTML باشد. برای ایجاد پاسخ HTML در برابر درخواست مرورگر، نیاز به یک نما (View) داریم. نماها مشخص می‌کنند که ANC چگونه باید نتیجه‌ی تولید شده توسط متد Index را به شکل پاسخ HTML مناسب برای نمایش در مرورگر درآورد.

گام نخست، همان‌گونه که در لیست ۲-۷ نشان داده شده، تغییر مناسب در محتوای متد اکشن Index است. تغییرات به صورت پررنگ نشان داده شده‌اند.

لیست ۲-۷ پردازش و نمایش نما، در HomeController.cs در پوشه‌ی Controllers

```
using Microsoft.AspNetCore.Mvc;
namespace FirstProject.Controllers {
    public class HomeController : Controller {
        public IActionResult Index() {
            return View("MyView");
        }
    }
}
```

معرفی خروجی متد اکشن از نوع IActionResult، به معنی وادار کردن ANC به پردازش یک نماست. در بدنه‌ی متد، ایجاد خروجی از نوع IActionResult، با فراخوانی متد View() و مشخص کردن نام نمای مورد نظر (MyView) به عنوان پارامتر این متد، انجام می‌شود.

^۱ Route

^۲ Routing System