Preface

The world of web development, as we see today, has undergone many dynamic changes shaped by multiple new technologies and platforms. Over the last few years Microsoft ASP.NET has quickly evolved to become one of the most famous platforms for developing web-based solutions. Since early 2002, when the first version (1.0) of ASP.NET was released, Microsoft has continuously added many out-of-the-box features and components, making web development easier for the end developer. In a very short time span, the ASP.NET platform has grown and matured into a stable object-oriented framework, with a large set of useful tools and a huge class library, attracting widespread interest in the developer communities around the world. With the introduction of LINQ, MS AJAX, WCF, WPF, and a lot of exciting new tools, the .NET framework has not only grown large but also flexible, in terms of the choices and options being offered to the developers.

With all of these new technologies hogging the limelight, an ever-increasing gap was created in the mindset of new developers, due to a shift in priorities. Developers, especially beginners, were attracted by the buzz created by these new, cool tools, and started interpreting them as a solution for better architecture and design, losing focus on the core concepts in the process. A developer, who has just learnt the basics of ASP.NET, was more eager to devote his or her time to technologies such as AJAX and LINQ instead of learning and implementing design patterns.

One reason for this paradigm shift was the lack of books that could showcase a better way to structure and develop ASP.NET-based web solutions, explaining with examples how to use different architectural styles and design patterns in real-life ASP.NET code. This book aims to bridge that gap.

I won't be focusing on deep and detailed theoretical concepts, as this book is not a "pure" architecture and design guide. Rather, the goal is to show you how to design a web site in ASP.NET the correct way, focus on different design options, analyze and study what architectural options we have, and decide when to use which architectural solution. It is very important to understand that there is no one perfect or best way in architecture and design. We need to improvise, and adapt to each project's unique requirements. Understanding core application architecture and design patterns can be tough for many developers, and so this book aims to elucidate these through the use of real-life examples and code in ASP.NET. This book will also shed some light on the basics of better application structure, coding practices, and database design, and will demonstrate, with suitable examples, how the correct architectural decisions can greatly impact overall application stability and performance.

What This Book Covers

Chapter 1 will introduce you to architecture and design in ASP.NET, including tiers, layers, and logical structuring.

Chapter 2 discusses the advantages and disadvantages of using the simplest and easiest 1-tier, 1-layer default architecture in ASP.NET. You will also understand when and why we should use out-of-the-box data source controls, and how the 1-tier, 1-layer style is tightly-coupled and is not flexible or scalable.

Chapter 3 discusses what an ER diagram is, the domain model, the basics of UML, and what an n-layer design is, and how it increases the flexibility and maintainability of the code when compared to a 1-layer architecture. A sample project is explained with code in a 3-layer model. The drawbacks or limitations of this model are also discussed.

Chapter 4 talks about n-tier architecture in ASP.NET and how to implement it. It also explains Data Transfer Objects and how to use them with 4-tier and 5-tier web solutions.

In *Chapter 5*, you will learn and understand what MVC design is, and how the ASP.NET MVC framework helps us quickly implement MVC design in our web applications.

In *Chapter 6*, you will learn how and when to use the most common design patterns in ASP.NET: Factory, Dependency Injection, Singleton, and others.

Chapter 7 explains why we need SOA, explaining the advantages of SOA for a beginner. A sample project using SOA architecture is discussed. The chapter also explains how the Windows Communication Framework (WCF) compliments SOA.