DevSecOps for .NET Core

Securing Modern Software Applications

Afzaal Ahmad Zeeshan

Apress®

DevSecOps for .NET Core

Securing Modern Software Applications

Afzaal Ahmad Zeeshan

Apress®

DevSecOps for .NET Core: Securing Modern Software Applications

Afzaal Ahmad Zeeshan
Rabwah, Pakistan

ISBN-13 (pbk): 978-1-4842-5849-1 ISBN-13 (electronic): 978-1-4842-5850-7
https://doi.org/10.1007/978-1-4842-5850-7

Managing Director, Apress Media LLC: Welmoed Spahr
Acquisitions Editor: Smriti Srivastava
Development Editor: Matthew Moodie
Coordinating Editor: Shrikant Vishwakarma

Cover designed by eStudioCalamar

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

Distributed to the book trade worldwide by Springer Science+Business Media New York, 233 Spring Street, 6th Floor, New York, NY 10013. 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 rights@apress.com, or visit http://www.apress.com/rights-permissions.

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/978-1-4842-5849-1. For more detailed information, please visit http://www.apress.com/source-code.

Printed on acid-free paper

I dedicate this book to all the bots running our automation jobs on the servers, making the lives of software engineers better. They deserve credit, too.

Table of Contents

About the Author

Afzaal Ahmad Zeeshan is a software engineer based in Rabwah, Pakistan. He likes .NET Core for regular day development and has experience with Cloud, mobile, and API development. Afzaal Ahmad has experience with the Azure platform and likes to build cross-platform libraries/software with .NET Core. He received the MVP Award by Alibaba Cloud for cloud expertise. He was twice recognized as a Microsoft MVP for his work in the field of software development, was recognized four times as a CodeProject MVP for technical writing and mentoring, and was recognized four times as a C# Corner MVP in the same field.

About the Technical Reviewer

Iqra Ali is a software engineer at StackFinity in the UK. Iqra spends most of her time contributing to the open source world and authoring technical articles on CodeProject and C# Corner. Iqra has diversified experience in architecture design, development, and automation. She has also worked on multiple DevOps-related projects during her professional career. Iqra has published courses on asynchronous programming and DevOps in collaboration with online publishers such as Packt. Furthermore, she has reviewed technical books and courses. In her free time, she listens to podcasts and reads books.

Acknowledgments

Well, I admit I am a very smug person, but that doesn't stop me from acknowledging the people who helped me, not only with this project but also with my learning ventures, earning ventures, and what not. My mom, dad, brothers, sisters, and friends—I would like to thank you for never giving up on me and encouraging me to continue learning.

I would also like to thank many seniors from the online communities, who taught me the skill of "questioning." The best answers I have ever received in my experience were by asking the right questions.

I also want to give a shout out to C# Corner, CodeProject, Microsoft, Alibaba Cloud, and Digital Ocean for the awards, certificates, free goodies, and the motivation they provided me throughout the years. Last but not least, special credit to Eminem, as his songs were by my side as I went through the hardships of software development.

Introduction

As of the year 2020, software development has become a broader domain within computer science and even overshadows several other domains. Every now and then, people find tutorials about programming languages, tools, and design patterns. The only thing that changes is the name of the pattern. From the old days of waterfall development, all the way to service-oriented architectures. We have come to the age of microservices and DevOps. I have written articles that showcase the DevOps pipelines I built for my own projects as well as for customers over the years. Now, I decided to write a book covering good practices. (There are no best practices in DevOps and computer science as a whole; only good use cases for each situation.)

This book is for beginners who want to learn about DevOps and the introduction of security into the pipeline. If you have experience with DevOps pipelines, you can use this book as a guide to learn how to enforce code quality and security policies on your repositories. If you are new to DevOps, this will give you a good start to understanding where the industry is currently doing the research and user studies. Many organizations are moving toward an open source environment, so they should focus on how to build a secure and collaborative environment for their contributors and engineers. A CI pipeline that builds and runs tests is no longer valid and complete. Online projects include contributors from all across the globe. Different people bring different coding styles to the repository. This book will help you understand how to apply code checks to the contributions added to the repository.

I have developed DevOps pipelines for many customers and clients that I have worked with in the past. I always recommend that they maintain at least a CI pipeline (if they think DevOps is a bit overkill)

for their projects. From single-developer indie projects, to large-scale enterprise product development and maintenance. DevOps is a silver bullet against all bugs, code smells, performance issues, and security problems in your projects. This book will not provide you with the silver bullet, nor will it tell you about the seller, rather it will help you understand the workings of DevOps and how you can develop your own. My core focus throughout this book is to teach you the methods used to automate the pipeline and where to apply the appropriate techniques. I often see new developers applying complex DevOps tools and pipeline stages to a job that might have just needed four lines of code. Sometimes the service that is provided out of the box is not the perfect one. Sometimes, just sometimes, the status quo of your DevOps is wrong. Perhaps someone in the past implemented a tool they got for free at a conference. My goal throughout this book is to tell you that DevOps is here to help you, and DevSecOps is here to make sure that your DevOps pipelines are implemented to test every possible use-case of your product.

You should use this book as a consultation for your DevOps servers—as a checklist to see if you have created the stages correctly in the server. This book is useful for students, too, the developer group that I encounter regularly. It is important for students to explore the practices of DevOps before they start working in industry. Software companies teach DevOps practices and other code stability techniques, but a student with DevOps skills will land a better job than their fellow students who know a little of the operations and automation.

Use this book however you like; I welcome you to check out the GitHub repository and submit the changes. I will try my best to check all the issues that you have with the code, or with DevOps.

—Afzaal Ahmad Zeeshan

CHAPTER 1

Modern Software Engineering

We stumble upon software in our everyday life, from handheld mobile devices to intelligent components of a smart home all the way to big machines we employ for regular daily tasks. As a software engineer, you get to see your work being used in every aspect of life. As interesting and energizing as it might sound, it has its own problems that one needs to tackle. The same products that help millions "achieve more" could someday cease to function due to a bug. Even worse, that bug might be an attempt to hijack the system for illegal activity. A lot of the software deployed to production follows poor design architecture, gets published in non-tested environments, or is being used by customers who are unaware of the socially engineered risks of software exploitation. Software packages are no longer just executable files sent to a customer upon receipt of payment. Software packages have become more Internet-based, agile, adaptable, accessible, and intelligent.

Authoring a software package in the 1990s was different than it is today. Modern solutions start with a complex front-end design and integrate serverless components with distributed data sources. There are several payment options integrated into the software products that your customers rely on. In my five years of experience working in the open source community at CodeProject and C# Corner, I have experienced young (and passionate) software engineers writing software that manages

A. Ahmad Zeeshan, *DevSecOps for .NET Core*,
https://doi.org/10.1007/978-1-4842-5850-7_1

or (in some ways) handles financial tasks of an organization or an entity. Perhaps everyone wants to start with a *fintech* (finance-technology) startup as a "hello world" approach to software development. The problem is not with the software being written or who is writing the software, rather it's with the approach that is being taken to write the software. I do not blame anyone for dreaming big. Unlike other fields of engineering, software engineering is broader in the sense that there are more opinions than solutions. Don't believe me? Count the number of JavaScript frameworks and libraries and tell me which one to use. I will wait.

There are so many "optimal" ways to do a thing in modern software that one can almost forget the "my" way to do a thing. Innovations happen not because everyone is doing something by the book, rather they are doing something they felt good about. When you're doing things that you feel good about, you break a few things, learn from them, and swear not to do them again. A gentleman will try to stop others from doing the things that could harm them. Agile, DevOps, DevSecOps,[1] or *modern software engineering* aims to do just that in the world of software.

Agile development introduced principles that put software development, quality, and collaboration above contracts and plans. This improved the software quality and development approach, because now customers were involved in the development phase and engineers had a clear definition of requirements. As the requirements change, engineers can adapt quickly and work on what is important for the customer. In the Agile methodology, your projects do not follow a strict sprint approach, rather, they adapt to the changing requirements. This removes the problems that previous models, such as waterfall, introduced in software engineering. Agile makes the software development customer- and

[1]There is a manifesto that mentions the main points that DevSecOps tends to add or resolve with other development methodologies; you can read it here: https://www.devsecops.org/.

product-oriented, rather than contract-, tool-, or plan-oriented. You plan according to the customer's needs.

DevOps is the set of principles, rules, or manifestos that are used to increase automation as the code is developed, built, and deployed. Many concepts that are part of DevOps pipeline, such as continuous integration, are used by teams that do not follow DevOps completely. A complete DevOps pipeline recommends process automation to be used from the inception phase to the deployment and monitoring phases. The main feature of DevOps is to remove the silos between multiple teams of an organization or between organizations working on a project. The tools used for DevOps often use issue trackers or ticketing systems to include communication channels in the development process. This removes the need for email communication, which is slow and causes friction between processes.

DevSecOps, DevOpsSec, or SecDevOps is primarily the addition of security, performance, and stability to the DevOps cycle. DevSecOps builds on top of DevOps and adds extra checks and validation rules at each stage. For example, your development tools and IDEs should have code analysis tools to check for bugs in the code and notify the development "before" the code is checked in to the repository. Similarly, during the build phase, your code should be checked for common code smells and bad practices such as SQL Injection, XSS attacks, and so on.

DevSecOps does not stop there; it requires you to continuously monitor the application and test it. In this book, we will look at how we can run these tests in production. Another interesting aspect of DevOps is the automation of complete infrastructure, thanks to cloud computing. Now our infrastructure management is done on the automation processes, ensuring a secure deployment. Keeping everything in check helps build trust. Your developers can rely on the security of the infrastructure while developing the applications, and the Ops can depend on a secure package being deployed on the production environment.

A quick differentiation between SecDevOps, DevSecOps, and DevOpsSec. It depends on where in the pipeline the "Sec" part happens. Of course, this is just a naming convention and DevSecOps is used as it sounds familiar and is easier to understand—Sec being added to Dev and Ops. Naming is not important though; what is important is how you apply the analysis tools. A consensus is that SecDevOps is the recommended approach[2] to addition of security. In this approach, we think of security and performance before writing the code. Then the code writing happens, and the code is built. It is similar in nature to the test-driven development approaches where you write the test before writing the code.

DevSecOps tools help you develop and publish your software with confidence. This confidence comes from the automated test and quality assurance checks built into the systems. By adding the checks to the system in each stage, you can verify that the code passes a rigorous validation check before moving to the next step. Secondly, DevSecOps recommends that you add bugs and code vulnerabilities to your testing system to verify that the package does not contain any bugs and code exploitation possibilities. Your software can be a web application, a mobile application, an automation script, or a game. Your pipeline should run tests tailored to each deployment domain for your product.

DevOps is more than just automation; it is a practice to prevent making mistakes when you know they hurt the business. DevOps tends to discourage human intervention in the systems and recommends using scripts and jobs to run the tasks. In a distributed environment, you can rely on a job that verifies the code quality for each contribution. If your team is working on open source software—say hosted on GitHub or GitLab—then you need to enforce some code standards and policies before the code and contribution is added to your software.

[2]Read this website post to learn more about the three different approaches for security considerations with DevOps, https://blog.ariacybersecurity.com/blog/devsecops-vs-secdevops-blog.

Teams use a different model to communicate the requirements and tasks. Some use Scrum, some use Kanban boards in their teams, some prefer to use Git Workflow and issues on GitHub, while some do stand ups in the mornings.[3] There is no argument that can prove one method wrong, or another method better. It is just the team that chooses which model to select. You can perform everything using the Agile model if you minimize human intervention.

Secondly, human intervention can be a part of the development process, but the tools and processes can continue to run. If you do this, you are doing a good-enough DevOps. But, is there a good-enough DevOps? If there is, and you decide that you can leave a process in place that will take care of the execution of current and foreseeable future contributions, then yes, you have a good DevOps. What this means is that you define a set of rules and jobs that run and verify the contributions and code that is being added to your repository. You do not need to use the modern DevOps tools like GitLab. You can perform this on your own machine.[4] Or you can introduce tools that "enforce" DevOps practices and standards to improve code quality and stability and increase the productivity of your teams by a major factor.

In this book, I will not discourage the practices of Scrum or Agile, or the use of Kanban boards in favor of Git issues. What this book does is help you visualize what an automation cycle in DevOps means. It is nothing more than the removal of human intervention in each step. It's a workflow connecting the developer's machine to production virtual machines or servers. The workflow, with a set of predefined testing, verification, and quality checks, should pass before the code can be pushed to a next stage. The automation helps the teams release the code quickly and rapidly

[3]Or evenings. A global team is more likely to have a stand up in mornings/evenings. For half the team, it's morning, and for others it's evening.

[4]Check out Git hooks for more on this. You can also check Jenkins CI server to learn how a new commit is verified to be stable.

and implement a required feature quickly, without having to wait for a complete cycle renewal as in other methods.

It is thought[5] that DevOps brings a complex set of principles that needs to be implemented in all software. This is not true. DevOps is not complicated, and not every software program must integrate DevOps tools to function. Most DevOps requirements change with the size of the team. If your team is 10 or 20 people, you do not require large-scale ticketing systems, like Jira. Your work can be done with a simple Git issue-management system. GitLab tends to offer a beautiful issue tracking system. In four out of six of my active projects, I am currently using GitLab's issue tracker. If your team size is bigger than that, try using Azure DevOps' Kanban boards and see if that fits your requirements. In a nutshell, you do not need to invest your money just because some ABC company uses a certain product.

That is the purpose of this book, to guide you to the best practices for DevOps, and to simplify the concepts that modern software tools use as marketing terminology; security, performance, feature flags, continuous monitoring, and so on. I will also introduce code readability improvements and ways to remove common practices that add smell to your code.

Software Design

Software development, deployment, and management practices have evolved in the past decade. The terms of Agile and SREs that were specific to an organization are now being adapted by a broader audience. Freelance projects are also being moved away from version control and managed environments. Product owners are inviting external contributions to

[5]Read this article on leading open source news and blogs website with statements from DevOps leads and VPs of engineering from firms that are implementing DevOps for themselves or for their customers, at https://opensource.com/article/18/3/4-hardest-things-devops-transformation.

collaborate on software architecture, design, and development. I have worked, and I am still working, with clients on their own infrastructure using VPNs, version control tools, and OAuth for authentication/authorization. Code owners trust that the software that is deployed on-premises will protect their properties and provide them with control over every action that takes place. The design of software is changing at a faster rate. Organizations are unable to train their employees at the pace that is necessary to adopt to the software market. This is bringing a change in the methodologies as well. There are new trends and new buzzwords that include the methodology shift to DevOps, DevSecOps, and so on, and the development changes from monolith to service-oriented, microservices, service mesh, and more. Software is written to aid software engineers to develop secure and scalable software. This intelligence is due to the advanced and complex models used by the software and IDEs.

We have come a long way in the development approach that we take today. From writing software that targets a problem, to writing the solution that solves a problem for a customer. It took a decade of research and investment to study the principles needed to be applied to software engineering as well as software development. The software that is written today is adaptable, accessible, intelligent, and secure. Organizations invest in an international business to gather the talent from across the world to work on a problem for a common solution. Take the example of Facebook, WordPress, Instagram, and other giants that provide solutions to customers, solving one problem at a time.

Software development and deployment methods have changed drastically, but they can be listed in the following points:

- Software has become complex in architecture, but simple in nature and development. You can write a small program with a smaller memory footprint and CPU requirements, and then add it to a broader system that becomes a solution.

- Your databases are no longer stored on a central hub, and your databases are no longer relational-only in nature. Modern software requires databases of different nature for different purposes. A mixture of SQL and NoSQL databases is being used in production environments everywhere.
- Different paradigms of software development use different toolsets. IoT, for example, uses SQLite as a database, whereas a web application is likely to use a Microsoft SQL Server or MySQL for the deployment.
- Cloud platforms have changed the way you push software to production. This means that the development pattern changes to accommodate the deployment techniques. You write an application with containerization in mind when deploying with Docker runtime.
- You integrate third-party or out-proc logging and monitoring systems. Online monitoring tools are made available to study the patterns of application usage and user statistics. Google Analytics, Azure Application Insights, and so on are a few examples. You avoid using console logging mechanisms due to privacy, security, and regulation requirements. You cannot expose a password, despite being a test password, on the console logs.

Organizations are multicultural, meaning the solutions no longer take an English sentence for granted and only provide that sentence as a response to every request. Instead, they have invested linguists and technical writers of different languages and regions to write the content in different languages for customers that speak multiple languages. Software

is becoming tangible once again. Our devices, such as handhelds and PDAs, are becoming smarter. The software written to operate these devices needs to be smarter and secure. A software bug in one program of the PDA can be used to exploit the entire software stack of an organization or the customer for that PDA company. The largest businesses are being hacked every day, and their engineers are being socially engineered for a back door in the company's code.

Solutions on the Internet

When you develop software for web applications, the requirements totally change. You are no longer able to completely secure your source code, API keys, and app data by applying ProGuard[6] only to obfuscate the source code. Your application's source code and resources are downloaded by the clients before they can be used on their browsers. With these methods, you need to think differently. When thinking differently, you need to enforce how a hacker might think. Several hacking attempts are made when you leave the security out of the code. Consider the examples of session hijacking, XSS scripting, and so on.

An average customer might be cheated into entering their username/password details into a web page that does not belong to an authentic website, by means of phishing. What if that website belonged to you, and one of your customers enters their details in the website that was sent by someone they just met? Imagine your website was tasked with processing monetary transactions and the hacker has access to the customer's account. How would you tackle these scenarios? The answer is, you cannot. If you plan to take an action on an event that has happened, you are already too late. Perhaps the hacker has the username/password to

[6]ProGuard is a Java tool used by Android developers to shrink application size by removing unnecessary code from the binaries and to obfuscate the bytecode generated by Java compiler for JVM.

one of their accounts and now they can use this account to send an email/message to one of their peers for their details. Their wife or daughter perhaps. Natural computing and artificial intelligence helps us understand how we can integrate biometric authentication with the applications. Each time you log in to your laptop, Windows[7] requests a pin code or provides a secure mode of authentication. Why would they do so, when you can just use your password? The reason is that even if someone has your password, they still cannot get into your personal devices. Even if someone knows your pin code, they still cannot get into your online accounts. There must be a barrier that one has to cross before gaining access to someone's properties.

Note Software written on one platform must be used on another platform. .NET Core supports this aim of the developers. Microsoft provides authentication options with industry standards, such as OAuth, OpenID Connect, and so on. You should use and develop those frameworks instead of writing your own software. They have a better track record for performance, security, and trust in the community that will use your product as a consumer or client. In open source, project owners trust the software when it has a track record of stability.

If you work in industries where you are writing software for network-based applications, here is the first rule—never reinvent the wheel. Major software companies have invested years of research in developing secure frameworks and libraries to be used for development. .NET Framework (the parent framework of .NET Core) is more than a decade old. It was released in 2002 and the team has over 18 years of experience in writing the libraries and software stacks for development SDKs. You can rely

[7]Or Linux, or MacOS, etc.

on these frameworks to provide you with security, scalability, and performance. This is critical when you are working with security related elements in your applications. Passwords, for example, should be hashed. Period. You must hash the passwords with a strong hashing algorithm that is provided by the framework. There are algorithms that provide hashing, but they are not recommended for hashing in purposes like password hashing. MD5 is one such hash. I will discuss code security and how to add security in your code base in later chapters. Those chapters will also discuss how you can enforce the code policies in your repositories to prevent any unwanted bugs.

The software connected to servers has different options to control how a patch is implemented in the system. In this book, I will discuss a few control structures defined in DevOps and automation that allow teams to push a release patch before the release is loaded on the screen. This requires that your frameworks also support the latest technology and automation. Xamarin supports these trends and we can implement them in our mobile applications. If you are a store manager, you can easily tell your customers that your store is about to have a sale and dynamically change the price of goods and discount your products. All this can be done without changing any code in your software. Writing the software is difficult, and testing it is more difficult. Teams need to run many tests on the UI, integration, and QA standards in order to ensure that the latest update does not break anything in the application or degrade the overall UX. If you make a small mistake in the values, or change your mind later, you will need to apply the patches quickly and expect the customers to receive the updates on time before the sale goes live. Thanks to DevOps and automation, you can do that in less time than it took you to read this sentence. This process will be discussed in later chapters, especially when I discuss the release managements and production environments.

Multicultural Customers

Today, software is used by a global audience, unless you scope it down to a specific audience. People come to visit your solutions with different language and accessibility requirements. Your software needs to add these requirements to the checklist of QA and testing. Frameworks used by the developers introduce the tools and libraries that can be used to create the experience that can support accessibility needs. Your application does not need to support the English language. I have travelled to Malaysia, Qatar, and Turkey, and I have felt safer during the mobility in Qatar and Malaysia because of the language support that their software provides. In Turkey, they use the Turkish language as their primary language and most of their software does not include English support. As a customer I tend to avoid their local software when roaming and use Google's software for navigation and local searches.

Markets have evolved and software that targets a niche environment and groups has emerged. Not only the software, but also the software development communities, have started to raise awareness of the multicultural aspects of the community. Web solutions are changing the way they are rendered. A web page is not only an HTML document that is rendered by the web browser. Instead, a web page is now being used by the screen-readers and ARIA-tags are being enforced by technical standards. Global software vendors do not want to miss anyone during the process of globalization. Your DevOps team should also enforce these practices to ensure the software is ready to accept all traffic that it will receive. You cannot miss an opportunity to convert a visitor to a customer in the long run. Sometimes you will be writing the software that other software engineers will use. That is the use-case of GitHub or GitLab and other teams within Microsoft, Amazon, and Google. You need to investigate the language that they are using. Imagine giving software programs perfectly tuned for C or C++ developers to a Python engineer.

The IDE will not care if their indentation is okay, but the engineer will go crazy each time the IDE recommends adding a semicolon or a curly brace for block scoping. Similarly, when writing software for customers, you need to take care of their preferences. For many mobile application developers, it is not difficult to hire someone to write the content for their application marketplaces. If you can write the content in English, there will be online service providers who can translate the content into their native language. You can even use Google Translate and translate the content to a native language. I will not discuss the cultural impact on the automation, DevOps, or the development aspect. This part you need to investigate yourself, and it is not that difficult. But it can mean new markets for your application.

The Ever-Changing Market

The market is always changing, and today's open source world contains a plethora of libraries and frameworks for a regular use. Just try to find a suitable message queuing[8] library for your application. You have a thousand options to decide from. Cloud platforms have already removed the barriers of cost and integration. You can quickly integrate a solution into your applications with a single click. You only need to pay as you use a resource. Organizations have started to write their software in a cloud-native[9] approach and provide the solutions on the cloud. Customers only need to pay for the resources that they are using, for the time that they are

[8]Message queues are used in different aspects of your solutions. They are used with serverless applications to process the messages on demand by a serverless job/function. They are also used to balance the load on a worker thread, especially if your worker threads or nodes are weaker in compute power.

[9]A cloud-native design is one in which an application uses the cloud platform to its fullest, by fully utilizing its capabilities to scale, is highly available, and is fault tolerant.

using them. In order to succeed, you need to adopt this pattern. You write software online and provide it as a multi-tenant solution to the customers.

Design patterns[10] and architectures of applications are also changing. You need to think about the most important factors of your application and add availability to them. You need to think about the cost factor for the meat of your business and add redundancy to it. You also need to think about the marketing trends and how a customer conversion happens and add caching/compute power to that. All these elements make up a single cloud-native solution. One book, one tutorial, or a single hack-a-thon cannot teach you all these. You need to practice these by hand on your own subscriptions. At Apress, we have a library of hands-on practice resources that can help you understand how to build a cloud-native solution. You do not need to know a certain language/runtime in order to write cloud-native solutions. But to better adapt and be productive, you need to know the basics of how to write scalable applications.

The market is changing, not only for the customers or programs. The market is changing for software engineers as well. The days of desktop application developers are a bit behind the cloud now. The hottest trends and roles belong to the cloud, data, and machine learning. .NET Core happens to target all these roles. If you want to be relevant to today's market, you need to practice these roles and these sciences before it's too late. This book will not teach you how to become a data scientist or a cloud architect, but the patterns and designs discussed in this book will help you apply automation with confidence when you do.

[10]Read this extensive guide by Microsoft about Cloud Design Patterns, at https://docs.microsoft.com/en-us/azure/architecture/patterns/.

Security and Compliance Requirements

Finally, software is more than just the services running on a device. Today, laws have made software complicated (in the name of security and user privacy) by applying regulations and compliance requirements to them. The software, as well as the organization producing the software, is responsible for complying with rules set by the local authorities. Since your solution is likely to go global, you need to meet global regulations. In this book, I will mention a few products that include compliance and regulation checks. Software packages that target these requirements are premium and finding free software will come with its own limitations and requirements.

There are regulations that discuss how software should behave when in the hands of children (called COPPA), and there are regulations in place that control how data left by the customer must be processed and removed if not needed anymore (such as GDPR, CCPA, etc.). Software firms are paying heavy fines when they do not adhere to these regulations. Chapter 7 discusses how you can provide an automated way to add these checks to your code bases. It also covers how you can prevent[11] unnecessary legal actions against your software by providing a transparent view of your software and policies. This can be done by exposing the audit reports to the auditors, thus inviting external auditors to perform checks.

Prerequisites

Before starting the book, I want to mention the prerequisites and requirements needed to make the most out of this book. This book requires that you have a hosted account with any of your favorite DevOps tools.

[11]This book is not a legal recommendation, and I am not providing legal advice in this book or in the final chapter. You must consult a legal entity to learn more about how to work closely with regulations and to protect your and your customers' data, privacy, and property.

I recommend using GitHub, GitLab, or BitBucket. There are several other hosted code-repository service providers, such as Azure DevOps (hosted or on-premises), AWS CodeCommit, and so on, but I recommend going with GitHub or GitLab. Most of the concepts and scenarios will be explained with GitHub or GitLab as the environment. You can create a new account with GitHub at `https://github.com` and for GitLab at `https://gitlab.com`.

Since the entire topic revolves on the concept of version control and source controls, it is highly recommended that you know the basics of Git version control. In later chapters, I will explain some concepts of how Git version control works, but it will be an 1,000-feet overview of Git and not specifics.

This book discusses DevOps from a .NET Core's developer's point of view, so the hosting environment differs. .NET Core applications run on a multiverse of hosting environments, ranging from servers, mobile devices, ML hosting environments, and so on. You need to prepare a subscription for different platforms if you want to follow along. Most platforms for .NET Core provide a free 30-day trial, and some have a freemium offer to the customers. You can use those offers on the following:

- Microsoft Azure
- Heroku Platform
- AWS
- Alibaba Cloud
- Google Cloud Platform

For mobile applications, you can run the applications on your own accounts and devices. If you want to move to a production environment, you will need to purchase a subscription/license from the store. Different stores have different policies set for their development SDKs and for publishing executables. You will need to consult their store websites to learn more about joining these programs.

What to Expect in This Book

This book is a guide about the latest trends of DevOps and DevSecOps. The book is a theoretical approach to explain the DevOps concepts to a beginner or a professional who is currently working in automation and wants to learn more about DevOps. I added some sample scenarios that discuss real-world problems that need to be tackled.

To keep the book on-topic and succinct, I will avoid deep dives into different concepts and practices. You will find references in each chapter that can help you understand the technicalities about a specific subject. Similarly, I will list my own repositories and code samples that you can use for your own homework tasks. There is no assignment section in this book because the book does not contain any recipe and is not a cookbook. The assignment for this book is to read all the references that are provided in each chapter. You should consult the support material provided in this book in each chapter before moving to the next. The chapters are designed to support a DevOps lifecycle (discussion, development, testing, QA, release, customer support, and legal). Each chapter will complement a stage in the lifecycle of a project.

If you are using Microsoft Azure and want to follow along with the Azure samples, you can find the Azure templates in the GitHub repository for this project. The resource templates are provided only for complex architectures. I will switch between different products and tools based on simplicity. It might not be a recommended approach to follow in a real-world scenario. But to explain the concepts, it is always best to use the product that can do the job with the smallest number of clicks. This book will equally support different cloud vendors and DevOps tools.

What *Not* to Expect in This Book

You cannot expect a hands-on guide to development and software engineering. If you are looking for a guide or a hands-on approach to configure Terraform for your infrastructure, then this book might not