

# 1 What is Amazon Web Services?

This chapter covers

- Overview of Amazon Web Services
- The benefits of using Amazon Web Services
- What you can do with Amazon Web Services
- Creating and setting up an AWS account

Almost every IT solution gets labeled with the term *cloud computing* or even just *cloud* nowadays. Buzzwords like these may help sales, but they're hard to work with when trying to teach—or learn—how to work with these technologies. So, for the sake of clarity, let's start this book by defining some terms.

Cloud computing, or the cloud, is a metaphor for supply and consumption of IT resources. The IT resources in the cloud aren't directly visible to the user; layers of abstraction exist in between. The level of abstraction offered by the cloud varies, from offering virtual machines (VMs) to providing Software as a Service (SaaS) based on complex distributed systems. Resources are available on demand in enormous quantities, and you pay for what you use.

The official definition from the National Institute of Standards and Technology follows:

*Cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (networks, virtual machines, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction.*

—National Institute of Standards and Technology

Also, NIST defines the following five essential characteristics for cloud computing:

- *On-demand self-service*—The cloud enables us to provision resources ad hoc with the click of a button or an API call.
- *Broad network access*—Capabilities are available over the network.
- *Resource pooling*—The cloud assigns resources based on a multitenant model, which means consumers share the same physical and virtual resources.
- *Rapid elasticity*—The cloud allows us to expand and shrink the provisioned capacity on demand.
- *Measured service*—The cloud provides metrics allowing consumers to gain insights into the utilization of their resources.

Besides that, offerings are often divided into the following three types:

- *Public*—A cloud managed by an organization and open to use by the general public
- *Private*—A cloud that virtualizes and distributes the IT infrastructure for a single organization
- *Hybrid*—A mixture of a public and a private cloud

Amazon Web Services (AWS) is a public cloud. By combining your on-premises data center with AWS, you are building a hybrid cloud.

Cloud computing services also have several classifications, described here:

- *Infrastructure as a Service (IaaS)*—Offers fundamental resources like computing, storage, and networking capabilities, using virtual machines such as Amazon EC2, Google Compute Engine, and Microsoft Azure Virtual Machines.
- *Platform as a Service (PaaS)*—Provides platforms to deploy custom applications to the cloud, such as AWS Lambda, AWS App Runner, Google App Engine, and Heroku.
- *Software as a Service (SaaS)*—Combines infrastructure and software running in the cloud, including office applications like Amazon WorkSpaces, Google Workspace, and Microsoft 365.

AWS is a cloud-computing provider with a wide variety of IaaS, PaaS, and SaaS offerings. Let's go into a bit more detail about what AWS is and does.

# I.1 What is Amazon Web Services (AWS)?

*Amazon Web Services (AWS)* is a platform of web services that offers solutions for computing, storing, and networking, at different layers of abstraction. For example, you can attach volumes to a virtual machine—a low level of abstraction—or store and retrieve data via a REST API—a high level of abstraction. Use the services provided by AWS to host websites, run enterprise applications, and mine tremendous amounts of data. *Web services* are accessible via the internet by using typical web protocols (such as HTTP) and are used by machines or by humans through a UI. The most prominent services provided by AWS are *EC2*, which offers virtual machines, and *S3*, which offers storage capacity. Services on AWS work well together: you can use them to migrate existing on-premises infrastructures or build from scratch. The pricing model for services is pay-per-use.

As an AWS customer, you can choose among different *data centers*. AWS data centers are distributed worldwide. For example, you can start a virtual machine in Japan in exactly the same way as you would start one in Ireland. This enables you to serve customers worldwide.

The map in figure 1.1 shows AWS's data centers. Access to some of them is limited: some data centers are accessible for US government organizations only, and special conditions apply for the data centers in China. Additional data centers have been announced for Canada, Spain, Switzerland, Israel, UAE, India, Australia, and New Zealand.



Figure 1.1 AWS data center locations

Now that we have defined the most important terms, the question is: what can you do with AWS?

## 1.2 What can you do with AWS?

You can run all sorts of application on AWS by using one or a combination of services. The examples in this section will give you an idea of what you can do.

### 1.2.1 Hosting a web shop

John is CIO of a medium-sized e-commerce business. He wants to develop a fast, reliable, and scalable web shop. He initially decided to host the web shop on-premises, and three years ago, he rented machines in a data center. A web server handles requests from customers, and a database stores product information and orders. John is evaluating how his company can take advantage of AWS by running the same setup on AWS, as shown in figure 1.2.

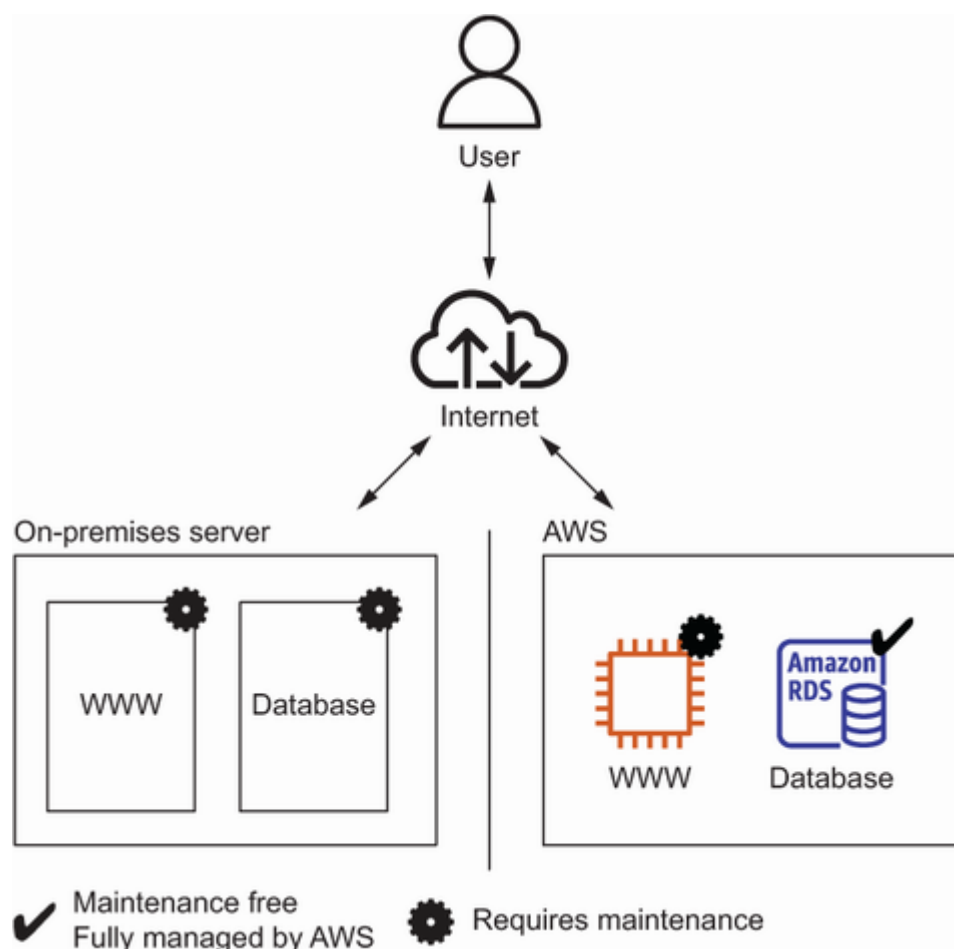


Figure 1.2 Running a web shop on-premises vs. on AWS

John not only wants to lift-and-shift his current on-premises infrastructure to AWS, he wants to get the most out of the advantages the cloud is offering. Additional AWS services allow John to improve his setup as follows:

- The web shop consists of dynamic content (such as products and their prices) and static content (such as the company logo). Splitting these up would reduce the load on the web servers and improve performance by delivering the static content over a content delivery network (CDN).
- Switching to maintenance-free services, including a database, an object store, and a DNS system, would free John from having to manage these parts of the system, decreasing operational costs and improving quality.
- The application running the web shop can be installed on virtual machines. Using AWS, John can run the same amount of resources he was using on his on-premises machine but split them into multiple, smaller virtual machines at no extra cost. If one of these virtual machines fails, the load balancer will send customer requests to the other virtual machines. This setup improves the web shop's reliability.

Figure 1.3 shows how John enhanced his web shop setup with AWS.

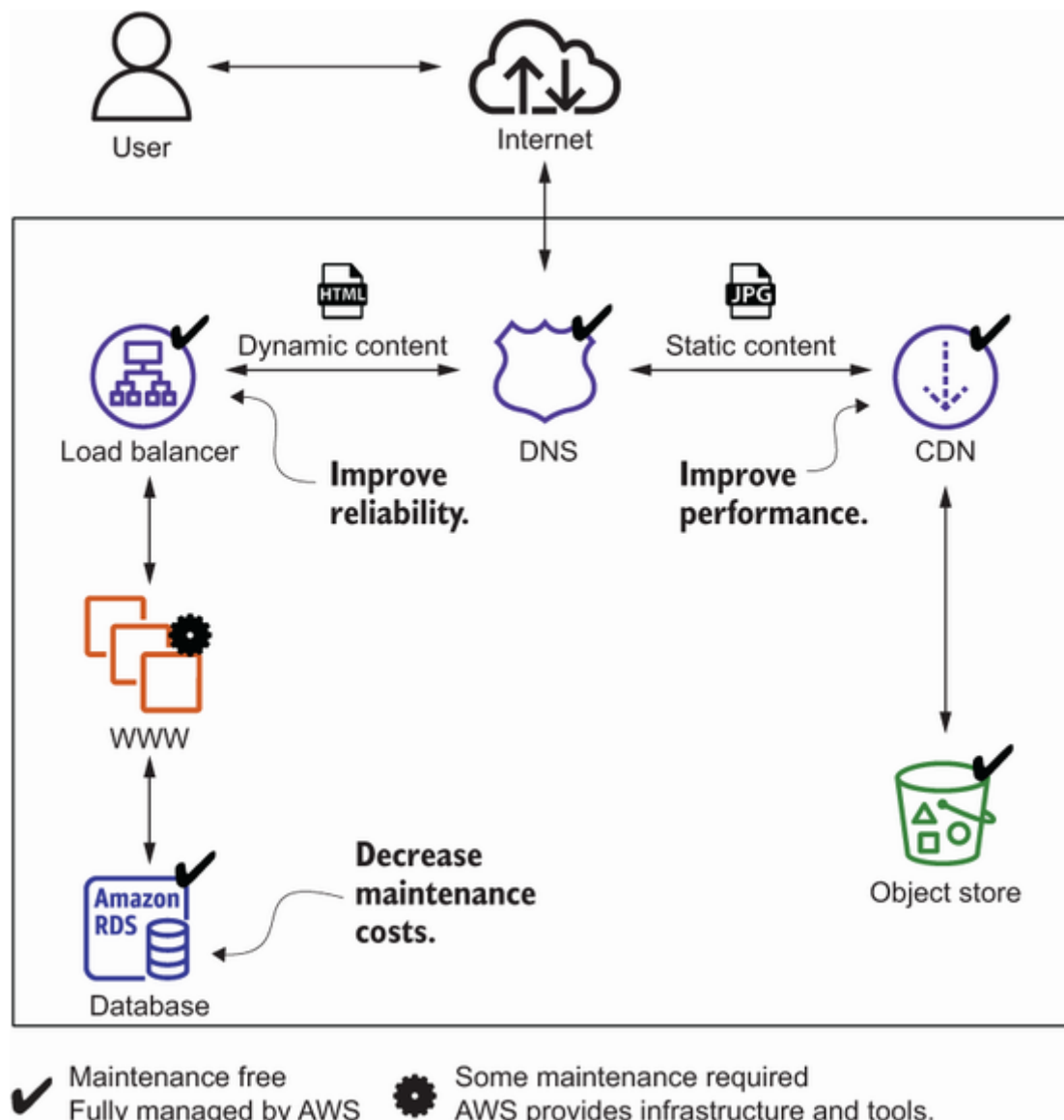


Figure 1.3 Running a web shop on AWS with CDN for better performance, a load balancer for high availability, and a managed database to decrease maintenance costs

John is happy with running his web shop on AWS. By migrating his company's infrastructure to the cloud, he was able to increase the reliability and performance of the web shop.

### 1.2.2 Running a Java EE application in your private network

Maureen is a senior system architect in a global corporation. She wants to move parts of her company's business applications to AWS when the data center contract expires in a few months, to reduce costs and gain flexibility. She would like to run enterprise applications (such as Java Enterprise Edition [EE] applications) consisting of an application server and an SQL database on AWS. To do so, she defines a virtual network in the cloud and connects it to the corporate network through a virtual private network (VPN) connection. She installs application servers on virtual machines to run the Java EE application. Maureen also wants to store data in an SQL

database service (such as Oracle Database EE or Microsoft SQL Server EE).

For security, Maureen uses subnets to separate systems with different security levels from each other. By using access-control lists, she can control ingoing and outgoing traffic for each subnet. For example, the database is accessible only from the Java EE server's subnet, which helps to protect mission-critical data. Maureen controls traffic to the internet by using network address translation (NAT) and firewall rules as well. Figure 1.4 illustrates Maureen's architecture.

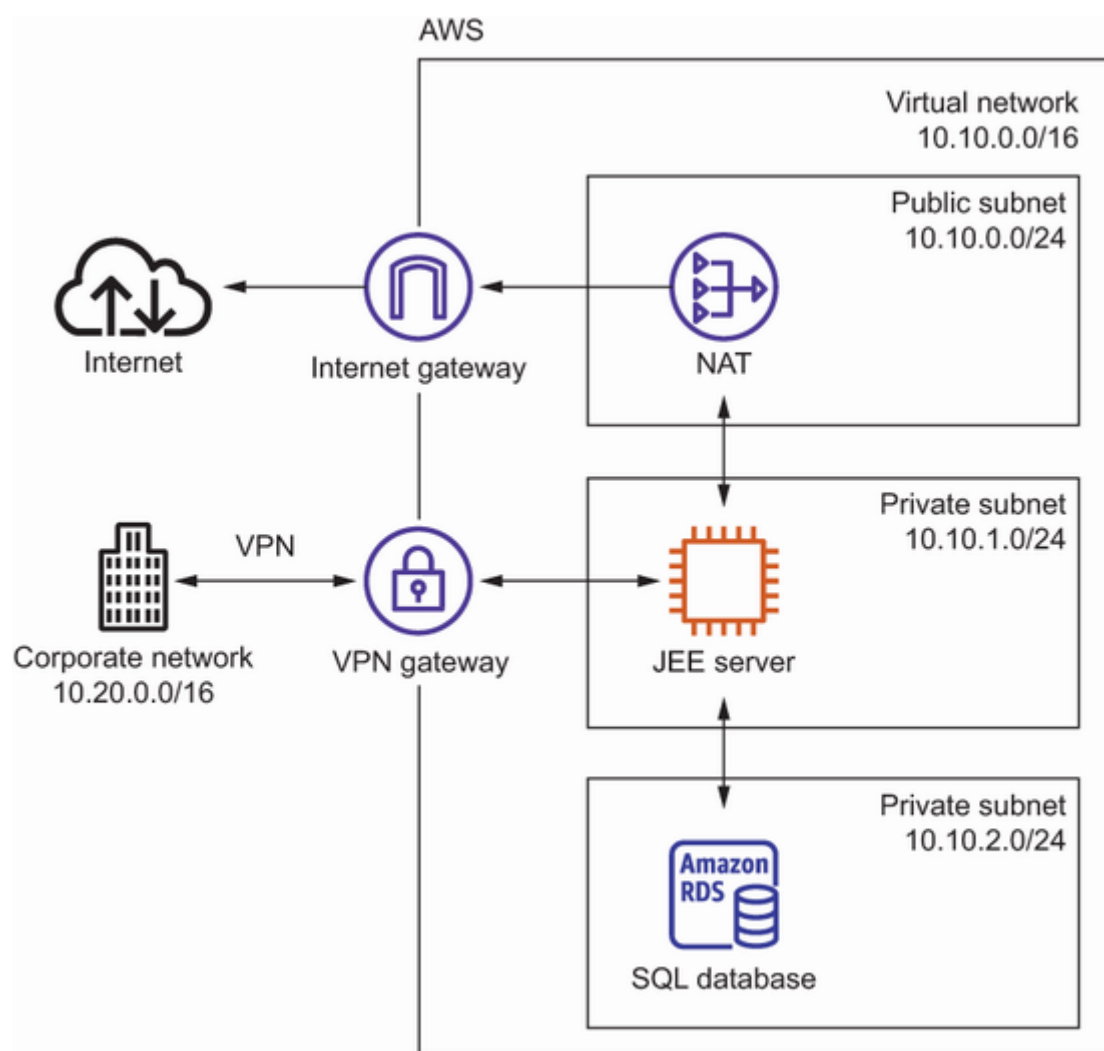


Figure 1.4 Running a Java EE application with enterprise networking on AWS improves flexibility and lowers costs.

Maureen has managed to connect the local data center with a private network running remotely on AWS to enable clients to access the Java EE server. To get started, Maureen uses a VPN connection between the local data center and AWS, but she is already thinking about setting up a dedicated network connection to reduce network costs and increase network throughput in the future.



The project was a great success for Maureen. She was able to reduce the time needed to set up an enterprise application from months to hours because AWS provides virtual machines, databases, and even the networking infrastructure on demand within a few minutes. Maureen's project also benefits from lower infrastructure costs on AWS, compared to using its own infrastructure on-premises.

### 1.2.3 Implementing a highly available system

Alexa is a software engineer working for a fast-growing startup. She knows that Murphy's Law applies to IT infrastructure: anything that can go wrong *will* go wrong. Alexa is working hard to build a highly available system to prevent outages from ruining the business. All services on AWS are either highly available or can be used in a highly available way. So, Alexa builds a system like the one shown in figure 1.5 with a high availability architecture. The database service is offered with replication and fail-over handling. In case the primary database instance fails, the standby database is promoted as the new primary database automatically. Alexa uses virtual machines acting as web servers. These virtual machines aren't highly available by default, but Alexa launches multiple virtual machines in different data centers to achieve high availability. A load balancer checks the health of the web servers and forwards requests to healthy machines.

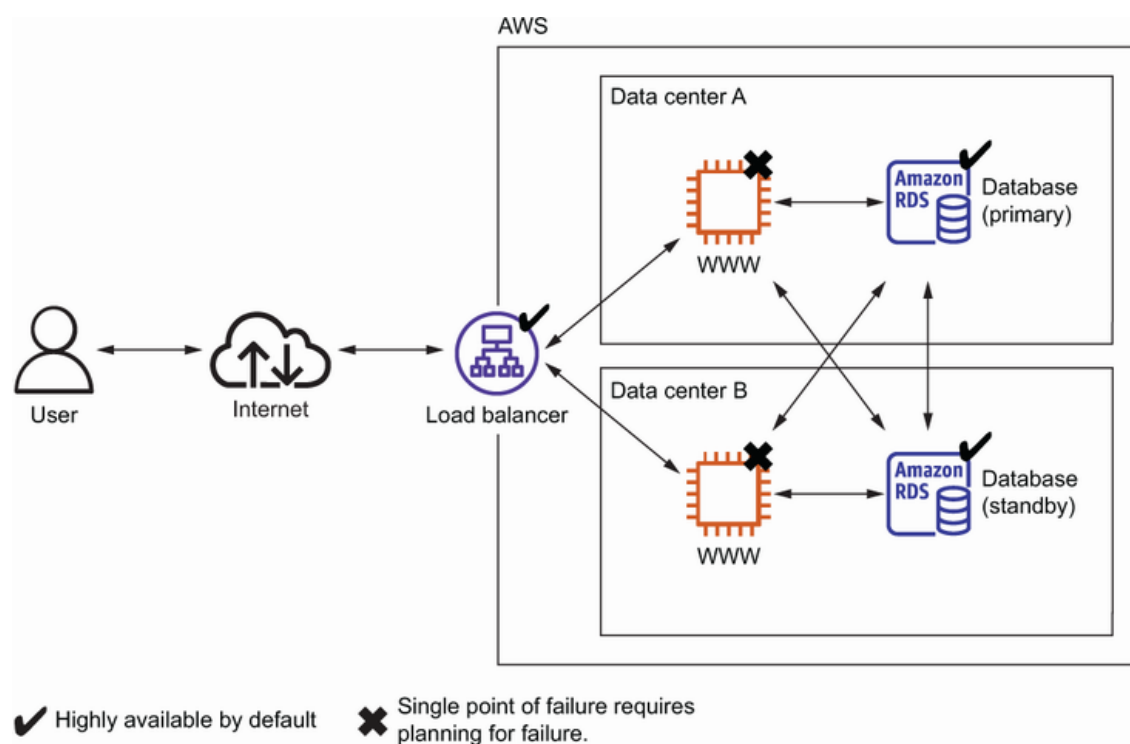


Figure 1.5 Building a highly available system on AWS by using a load balancer, multiple virtual machines, and a database with primary-standby replication



So far, Alexa has protected the startup from major outages. Nevertheless, she and her team are always planning for failure and are constantly improving the resilience of their systems.

#### 1.2.4 Profiting from low costs for batch processing infrastructure

Nick is a data scientist who needs to process massive amounts of measurement data collected from gas turbines. He needs to generate a daily report containing the maintenance condition of hundreds of turbines. Therefore, his team needs a computing infrastructure to analyze the newly arrived data once a day. Batch jobs are run on a schedule and store aggregated results in a database. A business intelligence (BI) tool is used to generate reports based on the data stored in the database.

Because the budget for computing infrastructure is very small, Nick and his team have been looking for a cost effective solution to analyze their data. He finds the following ways to make clever use of AWS's price model:

- *AWS bills virtual machines per second with a minimum of 60 seconds.* So Nick launches a virtual machine when starting a batch job and terminates it immediately after the job finishes. Doing so allows him to pay for computing infrastructure only when actually using it. This is a big game changer compared to the traditional data center where Nick had to pay a monthly fee for each machine, no matter how much it was used.
- *AWS offers spare capacity in their data centers at a substantial discount.* It is not important for Nick to run a batch job at a specific time. He can wait to execute a batch job until there is enough spare capacity available, so AWS offers him a virtual machine with a discount of 75%.

Figure 1.6 illustrates how Nick benefits from the pay-per-use price model for virtual machines.

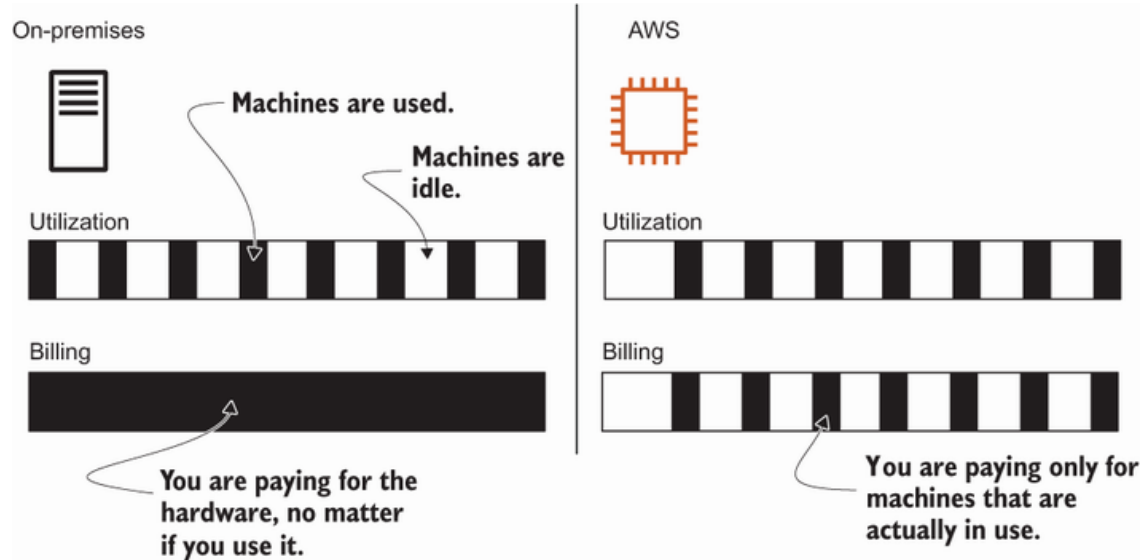


Figure 1.6 Making use of the pay-per-use price model of virtual machines

Nick is happy to have access to a computing infrastructure that allows his team to analyze data at low costs. You now have a broad idea of what you can do with AWS. Generally speaking, you can host any application on AWS. The next section explains the nine most important benefits AWS has to offer.

## 1.3 How you can benefit from using AWS

What's the most important advantage of using AWS? Cost savings, you might say. But saving money isn't the only advantage. Let's see how else you can benefit from using AWS by looking at some of its key features.

### 1.3.1 Innovative and fast-growing platform

AWS is announcing new services, features, and improvements constantly. Go to <https://aws.amazon.com/about-aws/whats-new/> to get an impression of the speed of innovation. We counted 2,080 announcements in 2021. Making use of the innovative technologies provided by AWS helps you to generate valuable solutions for your customers and thus achieve a competitive advantage.

Amazon reported net sales of \$62 billion for 2021. See <http://mng.bz/1RqB> if you are interested in the full report. We expect AWS to expand the size and extent of its platform in the upcoming years, for example, by adding additional services and data centers.

### **1.3.2 Services solve common problems**

As you've learned, AWS is a platform of services. Common problems such as load balancing, queuing, sending email, and storing files are solved for you by services. You don't need to reinvent the wheel. It's your job to pick the right services to build complex systems. Let AWS manage those services while you focus on your customers.

### **1.3.3 Enabling automation**

Because AWS is API driven, you can automate everything: write code to create networks, start virtual machine clusters, or deploy a relational database. Automation increases reliability and improves efficiency.

The more dependencies your system has, the more complex it gets. A human can quickly lose perspective, whereas a computer can cope with interconnected systems of any size. You should concentrate on tasks humans are good at—such as describing a system—while the computer figures out how to resolve all those dependencies to create the system.

Setting up an environment in the cloud based on your blueprints can be automated with the help of infrastructure as code, covered in chapter 4.

### **1.3.4 Flexible capacity (scalability)**

Flexible capacity reduces overcapacity. You can scale from one virtual machine to thousands of virtual machines. Your storage can grow from gigabytes to petabytes. You no longer need to predict your future capacity needs for the coming months and years to purchase hardware.

If you run a web shop, you have seasonal traffic patterns, as shown in figure 1.7. Think about day versus night, and weekday versus weekend or holiday. Wouldn't it be nice if you could add capacity when traffic grows and remove capacity when traffic shrinks? That's exactly what flexible capacity is about. You can start new virtual machines within minutes and throw them away a few hours after that.

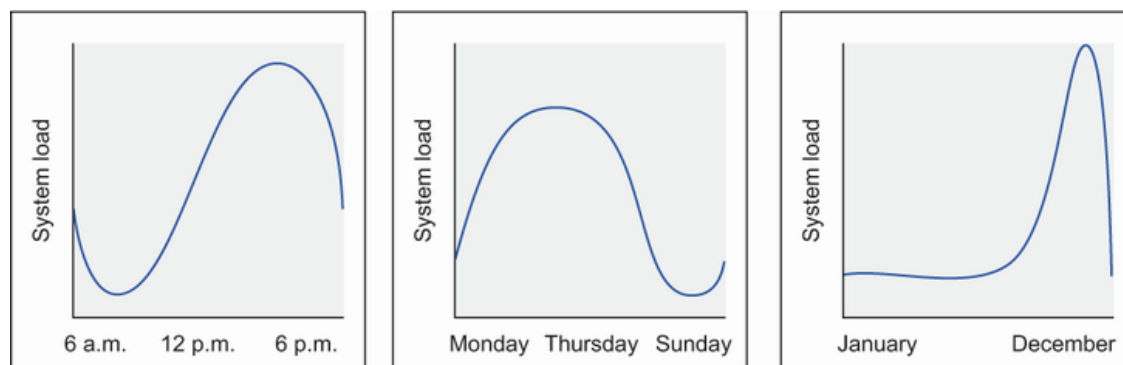


Figure 1.7 Seasonal traffic patterns for a web shop

The cloud has almost no capacity constraints. You no longer need to think about rack space, switches, and power supplies—you can add as many virtual machines as you like. If your data volume grows, you can always add new storage capacity.

Flexible capacity also means you can shut down unused systems. In one of our last projects, the test environment ran only from 7 a.m. to 8 p.m. on weekdays, allowing us to save 60%.

### 1.3.5 Built for failure (reliability)

Most AWS services are highly available or fault tolerant by default. If you use those services, you get reliability for free. Also, AWS provides tooling allowing you to build systems in a reliable way. It provides everything you need to create your own highly available or even fault-tolerant systems.

### 1.3.6 Reducing time to market

In AWS, you request a new virtual machine, and a few minutes later, that virtual machine is booted and ready to use. The same is true with any other AWS service available. You can use them all on demand.

Your development process will be faster because of the shorter feedback loops. You can eliminate constraints such as the number of test environments available; if you need another test environment, you can create it for a few hours.

### 1.3.7 Benefiting from economies of scale

AWS is increasing its global infrastructure constantly, and, therefore, AWS benefits from an economy of scale. As a customer, you will benefit partially from these effects.

AWS reduces prices for their cloud services every now and then. A few examples follow:

- In January 2019, AWS reduced the price for running containers on Fargate by 20% for vCPU and 65% for memory.
- In November 2020, AWS reduced prices for EBS volumes of type Cold HDD by 40%.
- In November 2021, AWS reduced prices for S3 storage by up to 31% in three storage classes.
- In April 2022, AWS removed additional costs for network traffic between data centers when using AWS PrivateLink, AWS Transit Gateway, and AWS Client VPN.

### 1.3.8 Global infrastructure

Are you serving customers worldwide? Making use of AWS's global infrastructure has the following advantages: having low network latencies between your customers and your infrastructure, being able to comply with regional data protection requirements, and benefiting from different infrastructure prices in different regions. AWS offers data centers in North America, South America, Europe, Africa, Asia, and Australia, so you can deploy your applications worldwide with little extra effort.

### 1.3.9 Professional partner

When you use AWS services, you can be sure that their quality and security follow the latest standards and certifications, such as the following:

- *ISO 27001*—A worldwide information security standard certified by an independent and accredited certification body.
- *ISO 9001*—A standardized quality management approach used worldwide and certified by an independent and accredited certification body.

- *PCI DSS Level 1*—A data security standard (DSS) for the payment card industry (PCI) to protect cardholders data.

Go to <https://aws.amazon.com/compliance/> if you want to dive into the details. If you're still not convinced that AWS is a professional partner, you should know that Expedia, Volkswagen, FINRA, Airbnb, Slack, and many more are running serious workloads on AWS. To read about AWS customer success, go to <https://aws.amazon.com/solutions/case-studies/>.

We have discussed a lot of reasons to run your workloads on AWS. But what does AWS cost? You will learn more about the pricing models in the next section.

## 1.4 How much does it cost?

A bill from AWS is similar to an electric bill. Services are billed based on use. You pay for the time a virtual machine was running, the used storage from the object store, or the number of running load balancers. Services are invoiced on a monthly basis. The pricing for each service is publicly available; if you want to calculate the monthly cost of a planned setup, you can use the AWS Pricing Calculator (<https://calculator.aws/>).

### 1.4.1 Free Tier

You can use some AWS services for free within the first 12 months of signing up. The idea behind the Free Tier is to enable you to experiment with AWS and get some experience using its services. Here is a taste of what's included in the Free Tier:

- 750 hours (roughly a month) of a small virtual machine running Linux or Windows. This means you can run one virtual machine for a whole month, or you can run 750 virtual machines for one hour.
- 750 hours (or roughly a month) of a classic or application load balancer.
- Object store with 5 GB of storage.
- Small relational database with 20 GB of storage, including backup.
- 25 GB of data stored on NoSQL database.

If you exceed the limits of the Free Tier, you start paying for the resources you consume without further notice. You'll receive a bill at the end of the month. We'll show you how to monitor your costs before you begin using AWS.

After your one-year trial period ends, you pay for all resources you use. But some resources are free forever. For example, the first 25 GB of the NoSQL database are free forever.

You get additional benefits, as detailed at <http://aws.amazon.com/free>. This book will use the Free Tier as much as possible and will clearly state when additional resources are required that aren't covered by the Free Tier.

### 1.4.2 Billing example

As mentioned earlier, you can be billed in the following ways:

- *Based on time of use*—A virtual machine is billed per second. A load balancer is billed per hour.
- *Based on traffic*—Traffic is measured in gigabytes or in number of requests, for example.
- *Based on storage usage*—Usage can be measured by capacity (e.g., 50 GB volume no matter how much you use) or real usage (such as 2.3 GB used).

Remember the web shop example from section 1.2? Figure 1.8 shows the web shop and adds information about how each part is billed.



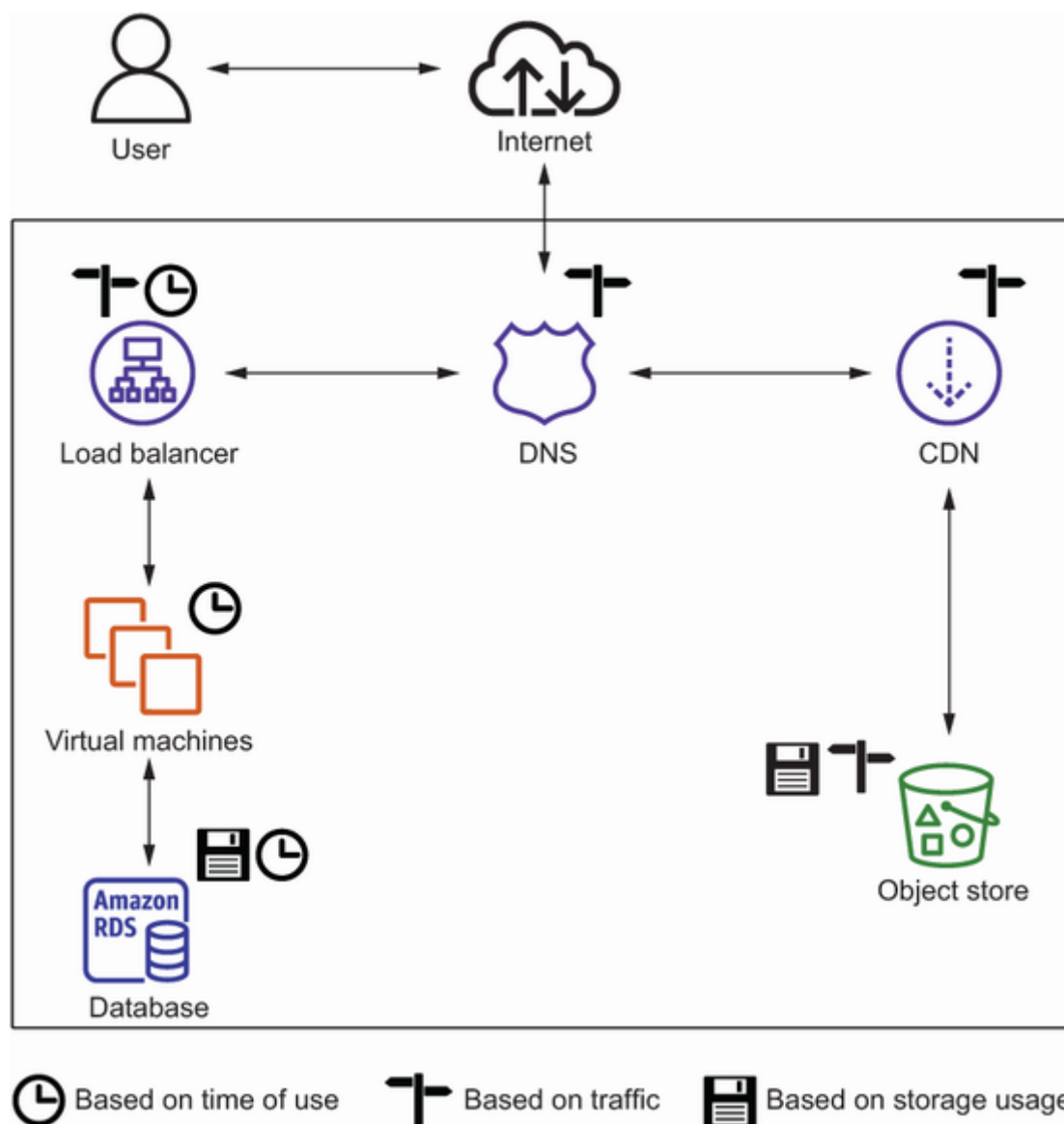


Figure 1.8 Some services are billed based on time of use, others by throughput or consumed storage.

Let's assume your web shop started successfully in January, and you ran a marketing campaign to increase sales for the next month. Lucky you: you were able to increase the number of visitors to your web shop fivefold in February. As you already know, you have to pay for AWS based on usage. Table 1.1 shows your bill for February. The number of visitors increased from 100,000 to 500,000, and your monthly bill increased from \$112 to \$473, which is a 4.2-fold increase. Because your web shop had to handle more traffic, you had to pay more for services, such as the CDN, the web servers, and the database. Other services, like the amount of storage needed for static files, didn't change, so the price stayed the same.

Table 1.1 How an AWS bill changes if the number of web shop visitors increases

Service	January usage	February usage	February charge	Increase
Visits to website	100,000	500,000		
CDN	25 M re-requests + 25 GB traffic	125 M re-requests + 125 GB traffic	\$115.00	\$100.00
Static files	50 GB used storage	50 GB used storage	\$1.15	\$0.00
Load balancer	748 hours + 50 GB traffic	748 hours + 250 GB traffic	\$19.07	\$1.83
Web servers	1 virtual machine = 748 hours	4 virtual machines = 2,992 hours	\$200.46	\$150.35
Database (748 hours)	Small virtual machine + 20 GB storage	Large virtual machine + 20 GB storage	\$133.20	\$105.47
DNS	2 M requests	10 M requests	\$4.00	\$3.20
<b>Total cost</b>			<b>\$472.88</b>	<b>\$360.85</b>

With AWS, you can achieve a linear relationship between traffic and costs. And other opportunities await you with this pricing model.

### 1.4.3 Pay-per-use opportunities

The AWS pay-per-use pricing model creates new opportunities. For example, the barrier for starting a new project is lowered, because you no longer need to invest in infrastructure up front. You can start virtual machines on demand and pay only per second of usage. You can stop using those virtual machines whenever you like, and you no longer have to pay for them. You don't need to make an upfront commitment regarding how much storage you'll use.

Another example: a big virtual machine costs exactly as much as two smaller ones with the same capacity. Thus, you can divide your systems into smaller parts, because the cost is the same. This makes fault tolerance affordable not only for big companies but also for smaller budgets.

## 1.5 Comparing alternatives

AWS isn't the only cloud computing provider. Microsoft Azure and Google Cloud Platform (GCP) are major players as well. The three major cloud providers share a lot in common. They all have the following:

- A worldwide infrastructure that provides computing, networking, and storage capabilities
- An IaaS offering that provides virtual machines on-demand: Amazon EC2, Azure Virtual Machines, Google Compute Engine
- Highly distributed storage systems able to scale storage and I/O capacity without limits: Amazon S3, Azure Blob Storage, Google Cloud Storage
- A pay-as-you-go pricing model

But what are the differences between the cloud providers?

AWS is the market leader in cloud computing, offering an extensive product portfolio. Although AWS has expanded into the enterprise sector during recent years, it is still obvious that AWS started with services to solve internet-scale problems. Overall, AWS is building great services based on innovative, mostly open source, technologies. AWS offers complicated but rock-solid ways to restrict access to your cloud infrastructure.

Microsoft Azure provides Microsoft’s technology stack in the cloud, recently expanding into web-centric and open source technologies as well. It seems like Microsoft is putting a lot of effort into catching up with Amazon’s market share in cloud computing.

The Google Cloud Platform (GCP) is focused on developers looking to build sophisticated distributed systems. Google combines their worldwide infrastructure to offer scalable and fault-tolerant services (such as Google Cloud Load Balancing). The GCP seems more focused on cloud-native applications than on migrating your locally hosted applications to the cloud, in our opinion.

There are no shortcuts to making an informed decision about which cloud provider to choose. Each use case and project is different—the devil is in the details. Also don’t forget where you are coming from. (Are you using Microsoft technology heavily? Do you have a big team consisting of system administrators or are you a developer-centric company?) Overall, in our opinion, AWS is the most mature and powerful cloud platform available at the moment.

## **1.6 Exploring AWS services**

In this section, you will get an idea of the range of services that AWS offers. We’ll also construct a mental model, with the help of some diagrams, to give you a high-level look at where those services sit in relation to the AWS setup as a whole.

Let’s start with the mental model overview. Hardware for computing, storing, and networking is the foundation of the AWS cloud. AWS runs services on this hardware, as shown in figure 1.9.

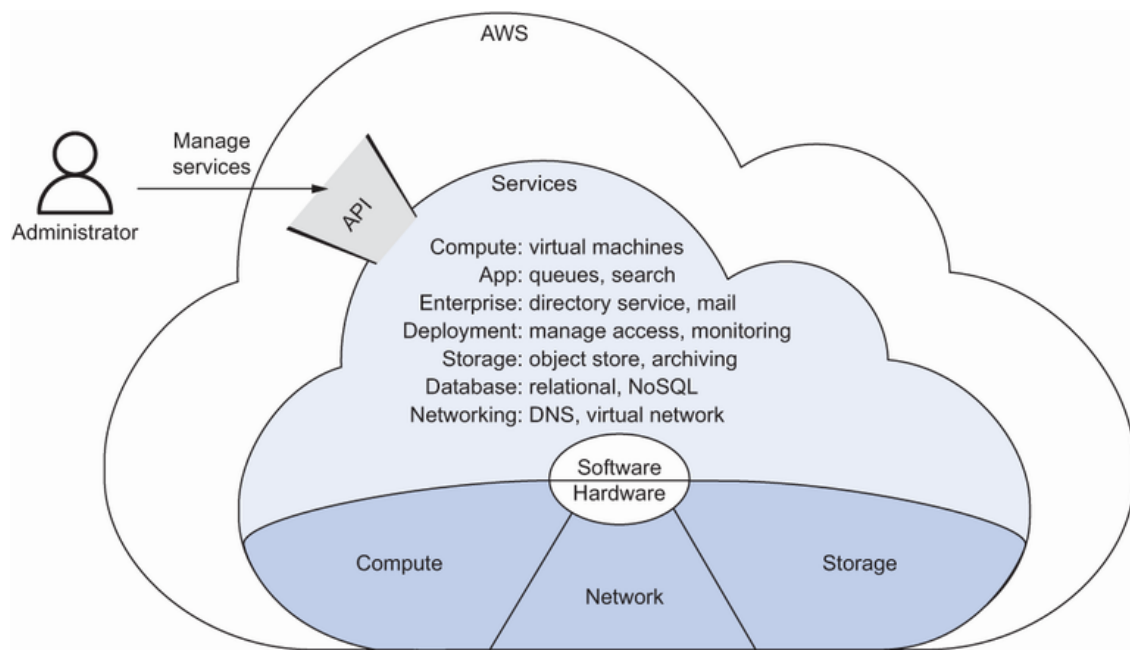


Figure 1.9 The AWS cloud is composed of hardware and software services accessible via an API.

You can manage services by sending requests to the API manually via a web-based GUI like the Management Console, a command-line interface (CLI), or programmatically via an SDK. Virtual machines have a special feature: you can connect to virtual machines through SSH, for example, and gain administrator access. This means you can install any software you like on a virtual machine. Other services, like the NoSQL database service, offer their features through an API and hide everything that's going on behind the scenes. Figure 1.10 shows an administrator installing a custom PHP web application on a virtual machine and managing dependent services such as a NoSQL database used by the application.

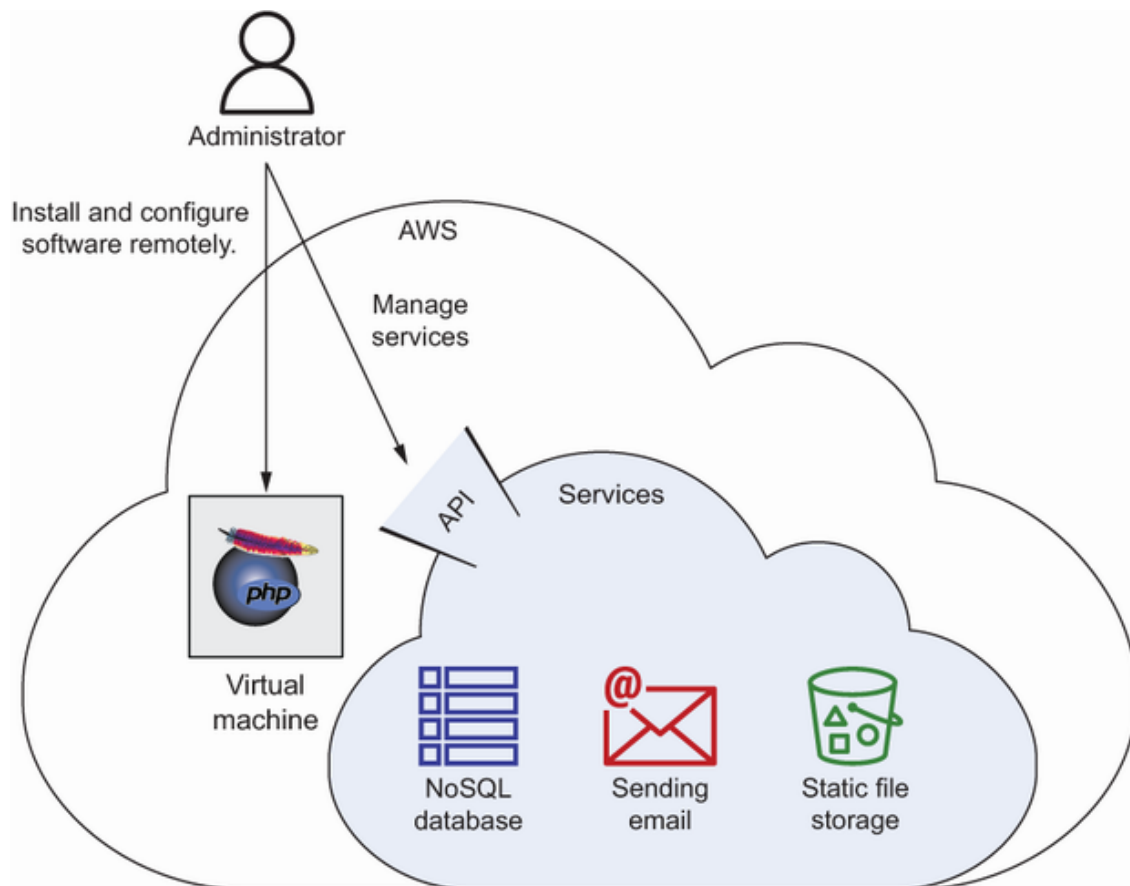


Figure 1.10 Managing a custom application running on a virtual machine and cloud-native services

Users send HTTP requests to a virtual machine. This virtual machine is running a web server along with a custom PHP web application. The web application needs to talk to AWS services to answer HTTP requests from users. For example, the application might need to query data from a NoSQL database, store static files, and send email. Communication between the web application and AWS services is handled by the API, as figure 1.11 shows.

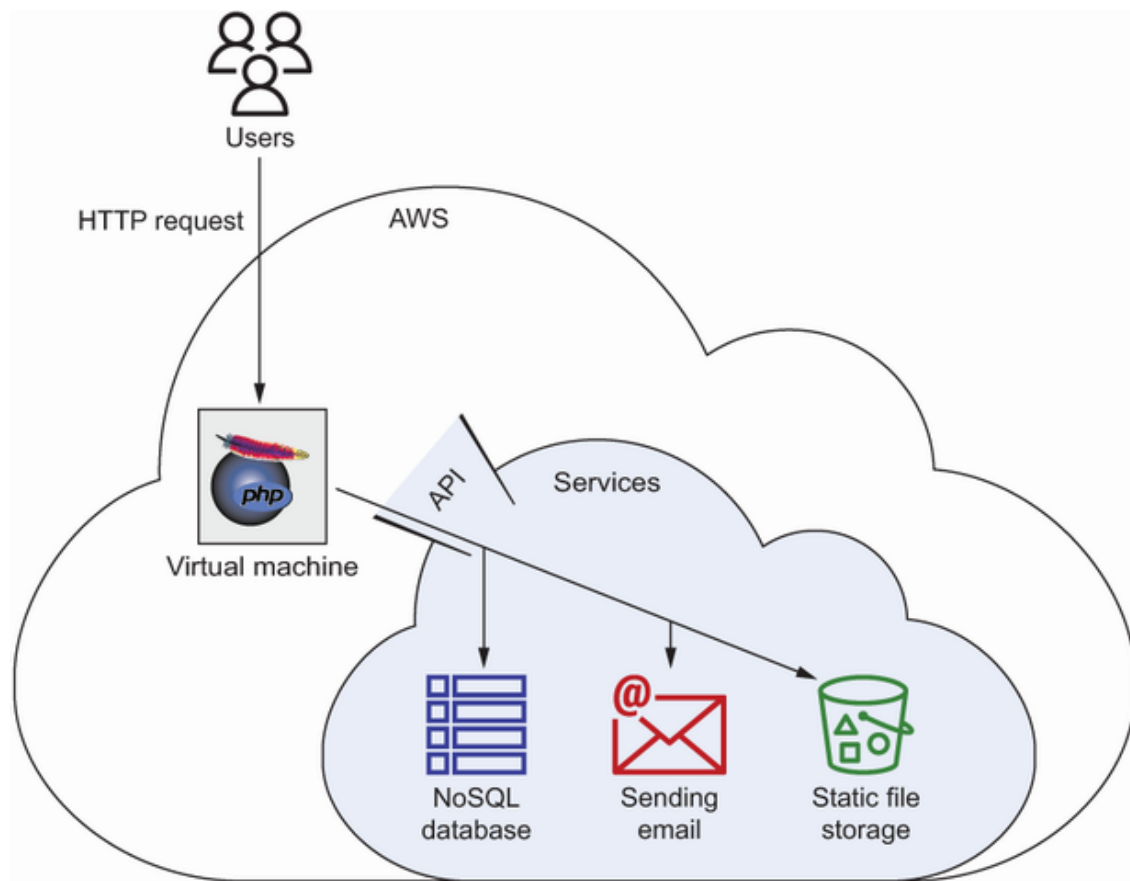


Figure 1.11 Handling an HTTP request with a custom web application using additional AWS services

The number of services available can be scary at the outset. When logging into AWS's web interface, you are presented with an overview listing around 200 services in 25 categories. On top of that, new services are announced constantly during the year and at the big conference in Las Vegas, AWS re:Invent, which takes place annually in November.

AWS offers services in the following categories:



- Analytics

- Application integration

- AR and VR

- AWS cost management

- Blockchain

- Business applications

- Compute

- Containers

- Customer enablement

- Database

- Developer tools

- End-user computing

- Frontend web and mobile

- Game Development

- Internet of Things

- Machine learning

- Management and governance

- Media services

- Migration and transfer

- Networking and content delivery

- Quantum technologies

- Robotics

- Satellite

- Security, identity, and compliance

- Storage

Obviously, it is impossible for us to cover all the services offered by AWS in one book. Therefore, in this book, we have selected for you the services that will help you get started quickly to build a fully capable, responsive, and dependable system, and then to grow and maintain that system. These are the most widely used services and will address most of your needs. After you've become more adept at working with AWS on these must-have services, feel free to investigate the nice-to-have services available to you.

The following services are covered in detail in our book:

- *EC2*—Virtual machines
- *ECS and Fargate*—Running and managing containers
- *Lambda*—Executing functions
- *S3*—Object store
- *Glacier*—Archiving data
- *EBS*—Block storage for virtual machines
- *EFS*—Network filesystem
- *RDS*—SQL databases
- *DynamoDB*—NoSQL database
- *ElastiCache*—In-memory key-value store
- *VPC*—Virtual network
- *ELB*—Load balancers
- *Simple Queue Service*—Distributed queues
- *CodeDeploy*—Automating code deployments
- *CloudWatch*—Monitoring and logging
- *CloudFormation*—Automating your infrastructure
- *IAM*—Restricting access to your cloud resources

We are missing at least three important topics that could fill their own books: continuous delivery, machine learning, and analytics. In fact, Manning has a whole book on doing machine learning and data analysis on AWS: *AI as a Service: Serverless Machine Learning with AWS* by Peter Elger and Eóin Shanaghy (<https://www.manning.com/books/ai-as-a-service>). You can read the first chapter free by following this link:

<http://mng.bz/BZvr>. We suggest you do so after you’ve finished our book, because we provide the foundational knowledge you need to work with any of these more advanced services.

Let’s return to a more immediate question: how exactly do you interact with an AWS service? The next section explains how to use the web interface, the CLI, and SDKs to manage and access AWS resources.

## 1.7 Interacting with AWS

When you interact with AWS to configure or use services, you make calls to the API. The API is the entry point to AWS, as figure 1.12 demonstrates.

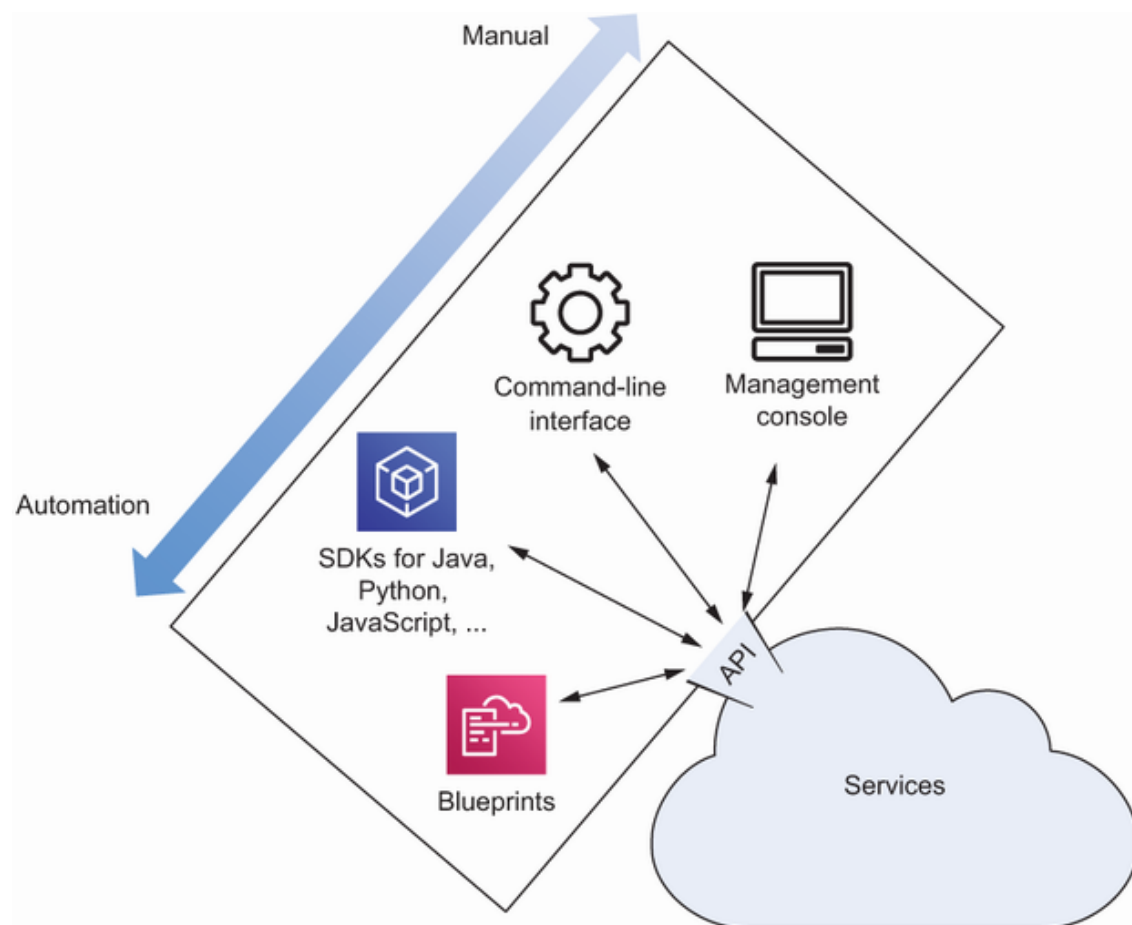


Figure 1.12 Different ways to access the AWS API, allowing you to manage and access AWS services

Next, we’ll give you an overview of the tools available for communicating with AWS’s APIs: the Management Console, the command-line interface, the SDKs, and infrastructure blueprints. We will compare the different tools, and you will learn how to use all of them while working your way through the book.

## 1.7.1 Management Console

The AWS Management Console allows you to manage and access AWS services through a graphical user interface (GUI), which works with modern browsers on desktop computers, laptops, and tablets. See figure 1.13.

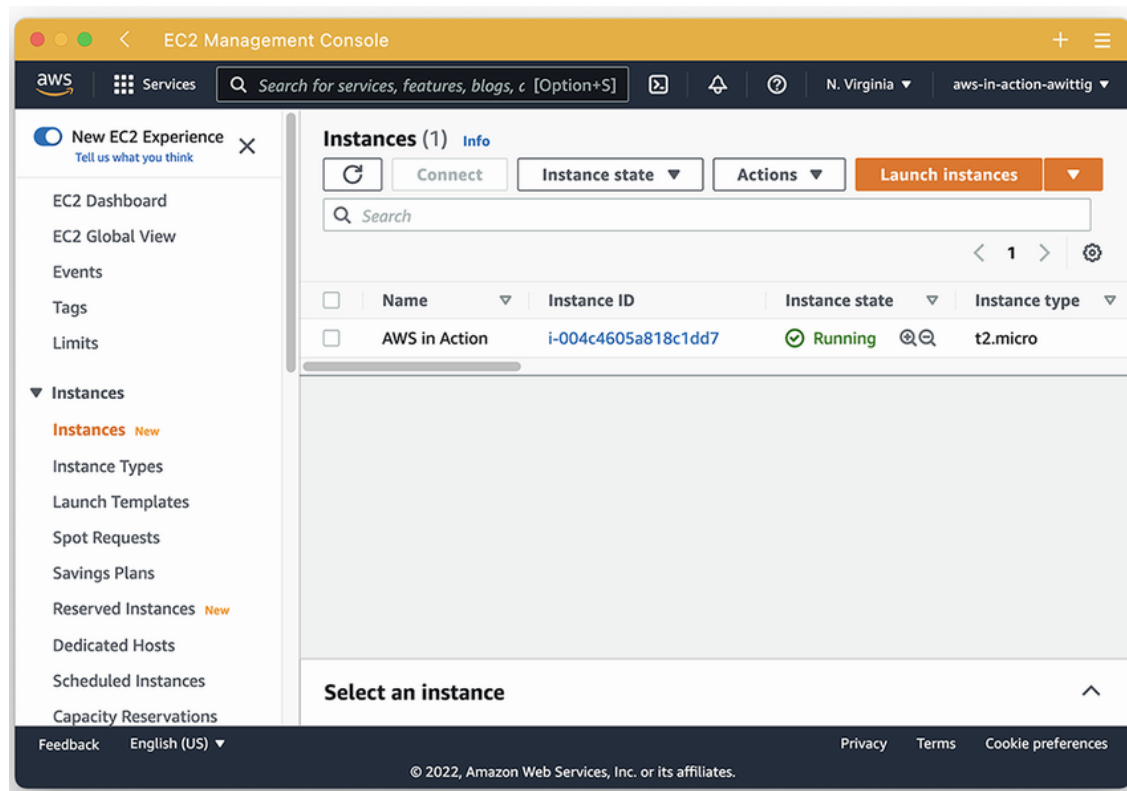
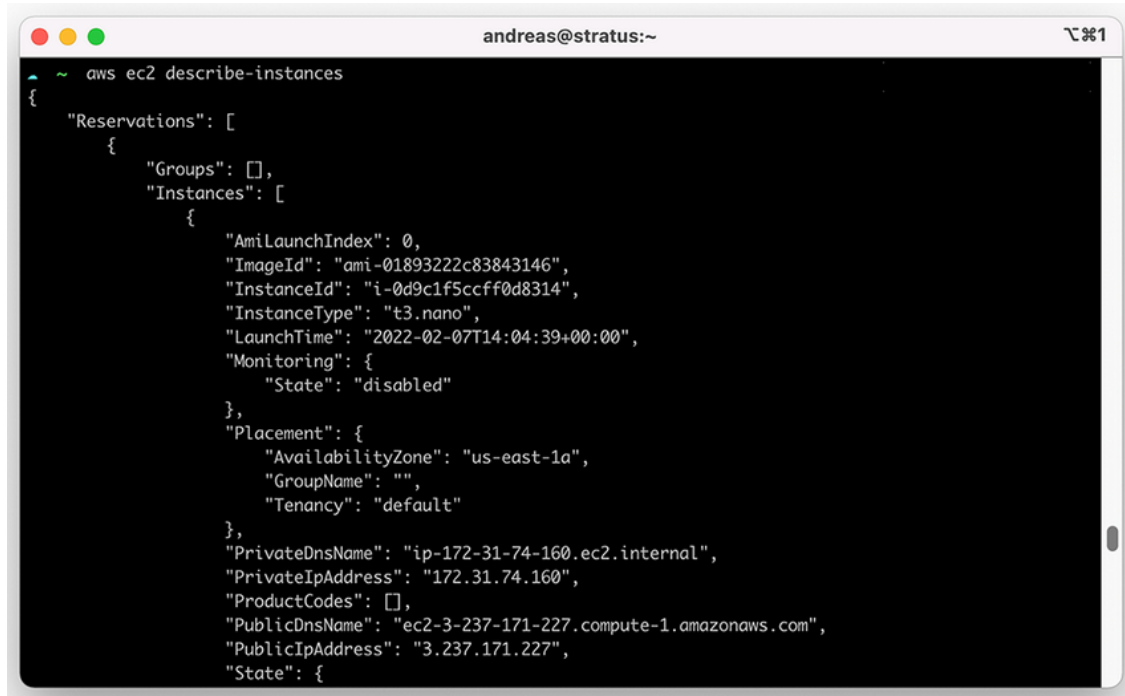


Figure 1.13 The AWS Management Console offers a GUI to manage and access AWS services.

When getting started or experimenting with AWS, the Management Console is the best place to start. It helps you to gain a quick overview of the different services. The Management Console is also a good way to set up a cloud infrastructure for development and testing.

## 1.7.2 Command-line interface

The command-line interface (CLI) allows you to manage and access AWS services within your terminal. Because you can use your terminal to automate or semi-automate recurring tasks, the CLI is a valuable tool. You can use the terminal to create new cloud infrastructures based on blueprints, upload files to the object store, or get the details of your infrastructure's networking configuration regularly. Figure 1.14 shows the CLI in action.

A terminal window titled 'andreas@stratus:~' showing the command 'aws ec2 describe-instances' and its output. The output is a JSON object representing a list of EC2 instances. The first instance is a t3.nano instance with ID 'i-0d9c1f5ccff0d8314', launched on 2022-02-07T14:04:39+00:00, in the 'us-east-1a' availability zone. The instance is in the 'disabled' state. The terminal window has a dark background and a light-colored border.

```
~ aws ec2 describe-instances
{
  "Reservations": [
    {
      "Groups": [],
      "Instances": [
        {
          "AmiLaunchIndex": 0,
          "ImageId": "ami-01893222c83843146",
          "InstanceId": "i-0d9c1f5ccff0d8314",
          "InstanceType": "t3.nano",
          "LaunchTime": "2022-02-07T14:04:39+00:00",
          "Monitoring": {
            "State": "disabled"
          },
          "Placement": {
            "AvailabilityZone": "us-east-1a",
            "GroupName": "",
            "Tenancy": "default"
          },
          "PrivateDnsName": "ip-172-31-74-160.ec2.internal",
          "PrivateIpAddress": "172.31.74.160",
          "ProductCodes": [],
          "PublicDnsName": "ec2-3-237-171-227.compute-1.amazonaws.com",
          "PublicIpAddress": "3.237.171.227",
          "State": {
```

Figure 1.14 The CLI allows you to manage and access AWS services from your terminal.

If you want to automate parts of your infrastructure with the help of a continuous integration server, like Jenkins, the CLI is the right tool for the job. The CLI offers a convenient way to access the API and combine multiple calls into a script.

You can even begin to automate your infrastructure with scripts by chaining multiple CLI calls together. The CLI is available for Windows, Mac, and Linux, as well as PowerShell.

### 1.7.3 SDKs

Use your favorite programming language to interact with the AWS API. AWS offers SDKs for the following platforms and languages:

- JavaScript
- Python
- PHP

- .NET
- Ruby
- Java

- Go
- Node.js
- C++

SDKs are typically used to integrate AWS services into applications. If you're doing software development and want to integrate an AWS service like a NoSQL database or a push-notification service, an SDK is the right choice for the job. Some services, such as queues and topics, must be used with an SDK.

### 1.7.4 Blueprints

A *blueprint* is a description of your system containing all resources and their dependencies. An Infrastructure as Code tool compares your blueprint with the current system and calculates the steps to create, update, or delete your cloud infrastructure. Figure 1.15 shows how a blueprint is transferred into a running system.

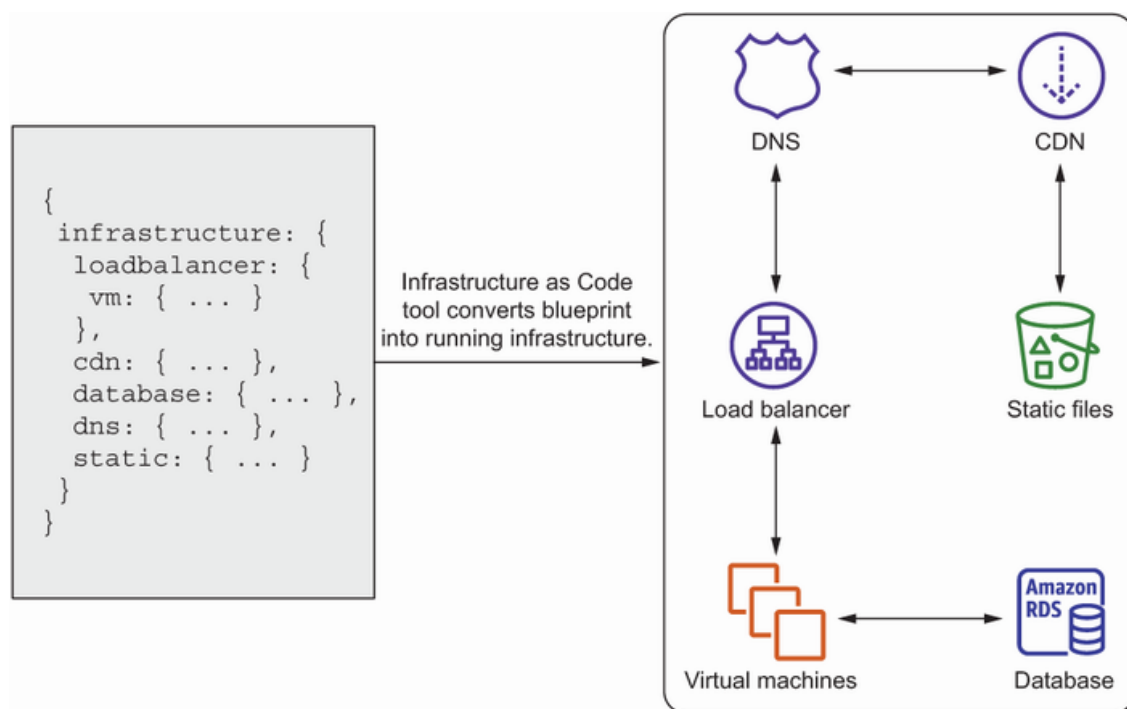


Figure 1.15 Infrastructure automation with blueprints

Consider using blueprints if you have to control many or complex environments. Blueprints will help you to automate the configuration of your infrastructure in the cloud. You can use them to set up a network and launch virtual machines, for example.



Automating your infrastructure is also possible by writing your own source code with the help of the CLI or the SDKs. Doing so, however, requires you to resolve dependencies, to make sure you are able to update different versions of your infrastructure, and to handle errors yourself. As you will see in chapter 4, using a blueprint and an Infrastructure-as-Code tool solves these challenges for you. It's time to get started creating your AWS account and exploring AWS practice after all that theory.

## I.8 Creating an AWS account

Before you can start using AWS, you need to create an account, which is a basket for all your cloud resources. You can attach multiple users to an account if multiple people need access to it; by default, your account will have one AWS account root user. To create an account, you need the following:

- A telephone number to validate your identity
- A credit card to pay your bills

**USING AN OLD ACCOUNT?** It is possible to use your existing AWS account while working through this book. In this case, your usage might not be covered by the Free Tier, so you might have to pay for the use.

Also, if you created your existing AWS account before December 4, 2013, please create a new one, because some legacy problems might cause trouble when following our examples.

**MULTIPLE AWS ACCOUNTS?** It is fine to create more than one AWS account. AWS even encourages you to do so, to isolate different workloads.

### 1.8.1 Signing up

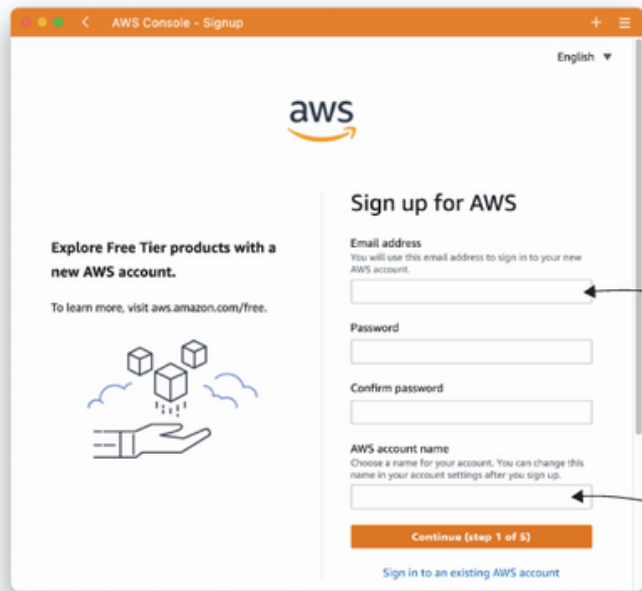
The sign-up process consists of five steps:

1. Providing login credentials
2. Providing contact information
3. Providing payment details
4. Verifying your identity
5. Choosing a support plan

Point your favorite web browser to <https://aws.amazon.com>, and click the Create an AWS Account button.

## 1. PROVIDING LOGIN CREDENTIALS

Creating an AWS account starts with defining a unique AWS account name, as shown in figure 1.16. The AWS account name has to be globally unique among all AWS customers. Try `aws-in-action-$yourname` and replace `$yourname` with your name. In addition to the account name, you have to specify an email address and a password used to authenticate the root user of your AWS account.



The screenshot shows the 'Sign up for AWS' page in the AWS Console. On the left, there is a promotional banner for the 'Free Tier' with an illustration of a hand holding blocks. The main form on the right contains the following fields:

- Email address:** A text input field with a note: 'You will use this email address to sign in to your new AWS account.'
- Password:** A text input field.
- Confirm password:** A text input field.
- AWS account name:** A text input field with a note: 'Choose a name for your account. You can change this name in your account settings after you sign up.'

Below the fields is an orange 'Continue (step 1 of 5)' button and a blue link for 'Sign in to an existing AWS account'. Two annotations with arrows point to the form fields:

- An arrow points to the 'Email address' field with the text: 'Type in your email address.'
- An arrow points to the 'AWS account name' field with the text: 'Use `aws-in-action-$yourname`, for example. Replace `$yourname`.'

Figure 1.16 First step of creating an AWS account: account name and password

We advise you to choose a strong password to prevent misuse of your account. *Use a password consisting of at least 20 characters.* Protecting your AWS account from unwanted access is crucial to avoid data breaches, data loss, or unwanted resource usage on your behalf.

## 2. PROVIDING CONTACT INFORMATION

The next step, as shown in figure 1.17, is adding your contact information. Fill in all the required fields, and continue.

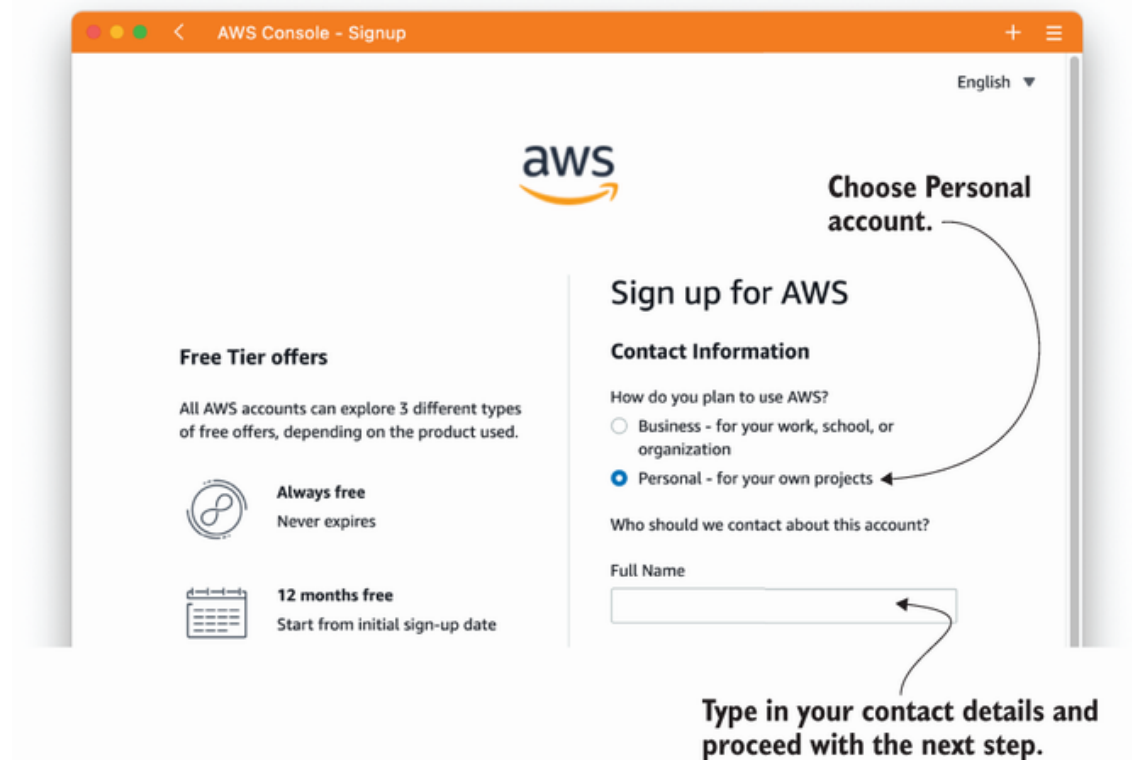


Figure 1.17 Second step of creating an AWS account: contact details

### 3. PROVIDING PAYMENT DETAILS

Next, the screen shown in figure 1.18 asks for your payment information. Provide your credit card information. There's an option to change the currency setting from USD to AUD, BRL, CAD, CHF, CNY, DKK, EUR, GBP, HKD, JPY, KRW, NOK, NZD, SEK, SGD, or ZAR later if that's more convenient for you. If you choose this option, the amount in USD is converted into your local currency at the end of the month.

**Secure verification**

**Billing Information**

Credit or Debit card number

Month Year

Cardholder's name

Billing address

☒ Use my contact address

☐ Use a new address

**Verify and Continue (step 3 of 5)**

You might be redirected to your bank's website to authorize the verification charge.

Figure 1.18 Third step of creating an AWS account: payment details

#### 4. VERIFYING YOUR IDENTITY

The next step is to verify your identity. Figure 1.19 shows the first step of the process.

After you complete the first part of the form, you'll receive a text message or call from AWS. All you need to do is to type in the verification code.

The screenshot shows the AWS Console - Signup page, specifically the 'Confirm your identity' step. The page has an orange header with the AWS logo and a language dropdown set to 'English'. On the left, there is an illustration of a person's profile with a checkmark, indicating successful verification. The main content area is titled 'Sign up for AWS' and 'Confirm your identity'. It explains that a phone number must be verified and that a verification code will be sent. Below this, there are two radio buttons for 'How should we send you the verification code?': 'Text message (SMS)' (selected) and 'Voice call'. There is a dropdown menu for 'Country or region code' and a text input field for 'Mobile phone number'. A 'Security check' section displays a CAPTCHA image with the characters 'xm8xpc' and 'vuu' overlaid with a large 'X'. To the right of the CAPTCHA are icons for a speaker and a refresh button. Below the CAPTCHA is a text input field labeled 'Type the characters as shown above'. At the bottom right, there is an orange button labeled 'Send SMS (step 4 of 5)'.

Figure 1.19 Fourth step of creating an AWS account: verify identity

## 5. CHOOSING A SUPPORT PLAN

The last step is to choose a support plan; see figure 1.20. For now, select the Basic plan, which is free. When running a production workload on AWS, we recommend at least a Developer plan to be able to ask questions about upcoming issues.



Figure 1.20 Fifth step of creating an AWS account: choose a support plan

High five! You're done. Click Go to the AWS Management Console, as shown in figure 1.21, to sign in to your AWS account for the first time.

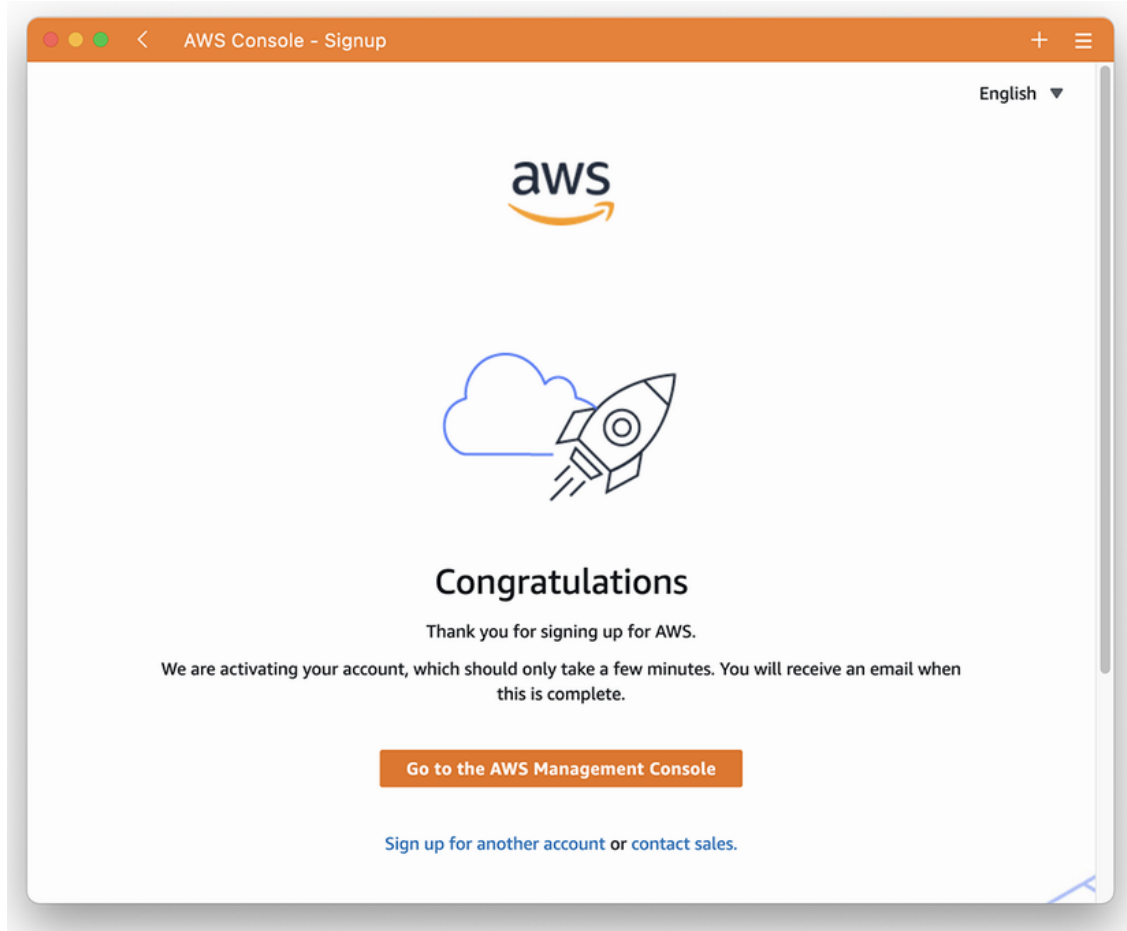


Figure 1.21 Fifth step of creating an AWS account: success!

## 1.8.2 Signing in

You now have an AWS account and are ready to sign in to the AWS Management Console. As mentioned earlier, the Management Console is a web-based tool you can use to control AWS resources; it makes most of the functionality of the AWS API available to you. Figure 1.22 shows the sign-in form at <https://console.aws.amazon.com>. Choose Root User and enter your email address, click Next, and then enter your password to sign in.



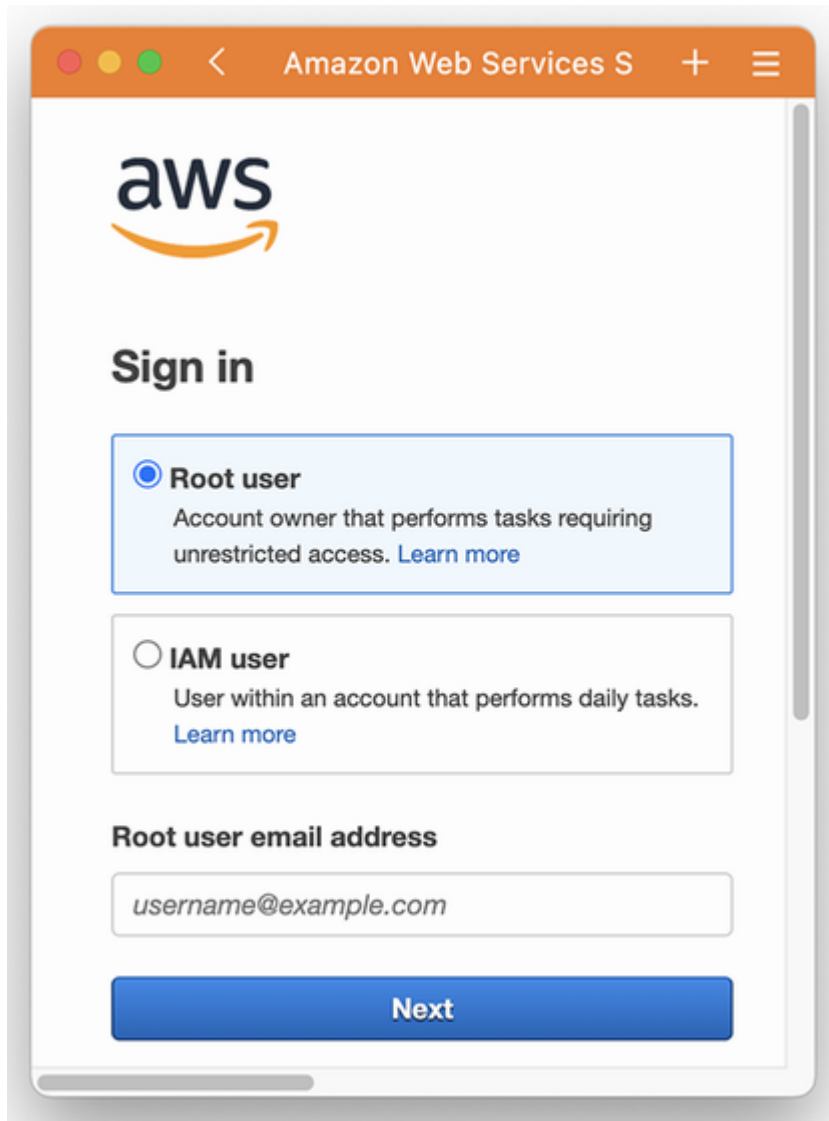


Figure 1.22 Sign in with the AWS account root user.

After you have signed in successfully, you are forwarded to the start page of the Management Console, as shown in figure 1.23.

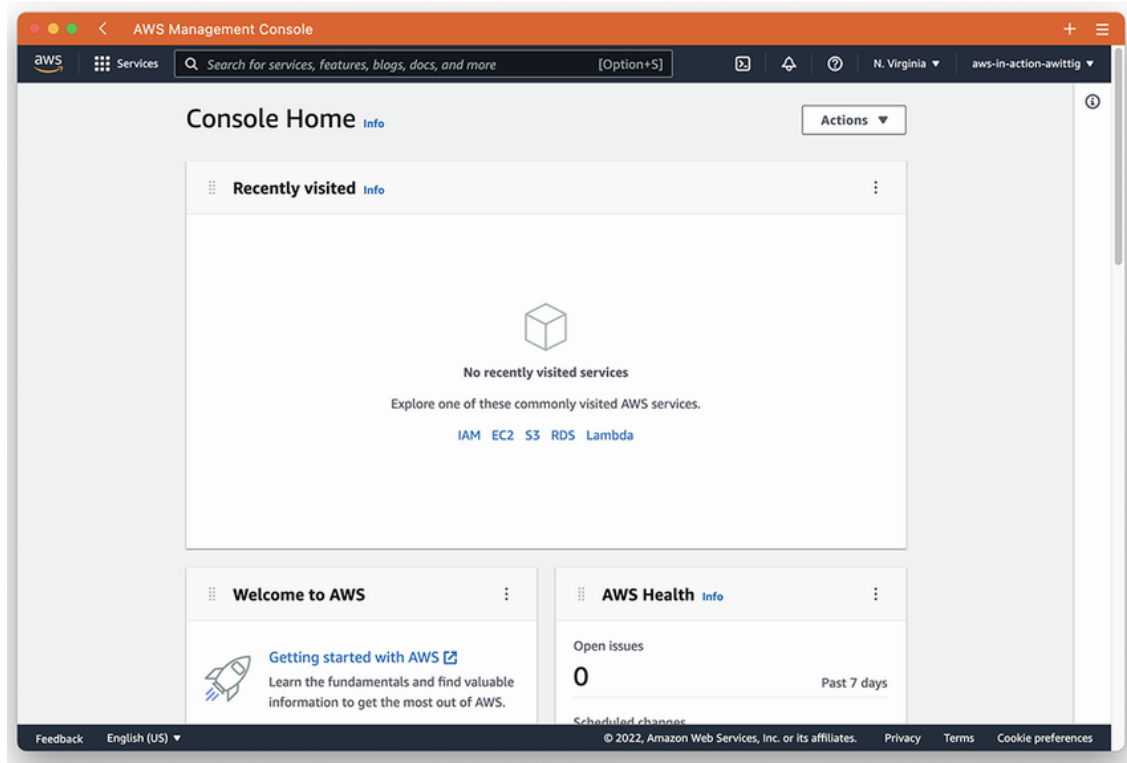


Figure 1.23 The dashboard after logging in for the first time

The most important part is the navigation bar at the top, shown in figure 1.24. It consists of the following eight sections:

- *AWS*—The dashboard of the Management Console shows an overview of your AWS account.
- *Services*—Provides access to all AWS services.
- *Search*—Allows you to search for services, features, and more.
- *Terminal*—Spin up a terminal with access to your cloud resources in the browser.
- *Notifications*—View alerts and notifications by AWS, for example, planned downtimes or outages.
- *Help*—Get support by the community, experts, or AWS support.
- *Region*—Select the region you want to manage.
- *Account*—Manage your AWS account.

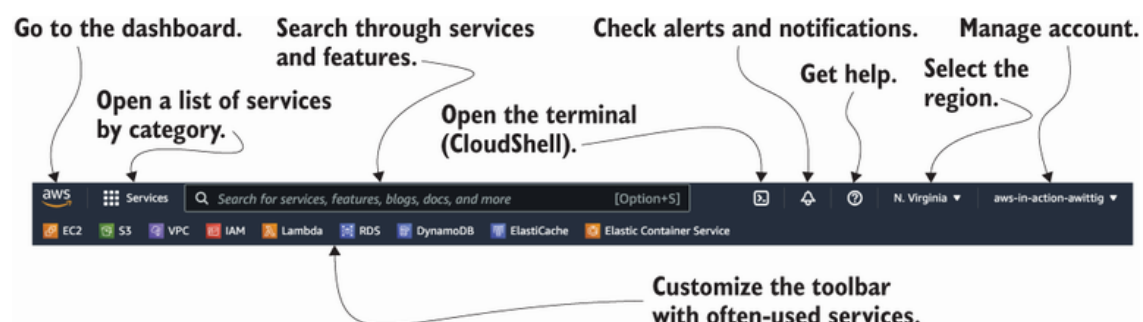


Figure 1.24 AWS Management Console navigation bar

Next, you'll make sure to avoid unwanted AWS costs.

## 1.9 Creating a budget alert to keep track of your AWS bill

At first, the pay-per-use pricing model of AWS might feel unfamiliar to you, because it is not 100% foreseeable what your bill will look like at the end of the month. Most of the examples in this book are covered by the Free Tier, so AWS won't charge you anything. Exceptions are clearly marked. To provide you with the peace of mind needed to learn about AWS in a comfortable environment, you will create a budget next. The budget will notify you via email in case your AWS bill will exceed \$5 so you can react quickly.

Before creating a budget, configure a Free Tier usage alert as follows:

1. Open the billing preferences of your AWS account at <http://mng.bz/5m8D>.
2. Enable Receive Free Tier Usage Alerts, and type in your email address as shown in figure 1.25.
3. Press the Save Preferences button.

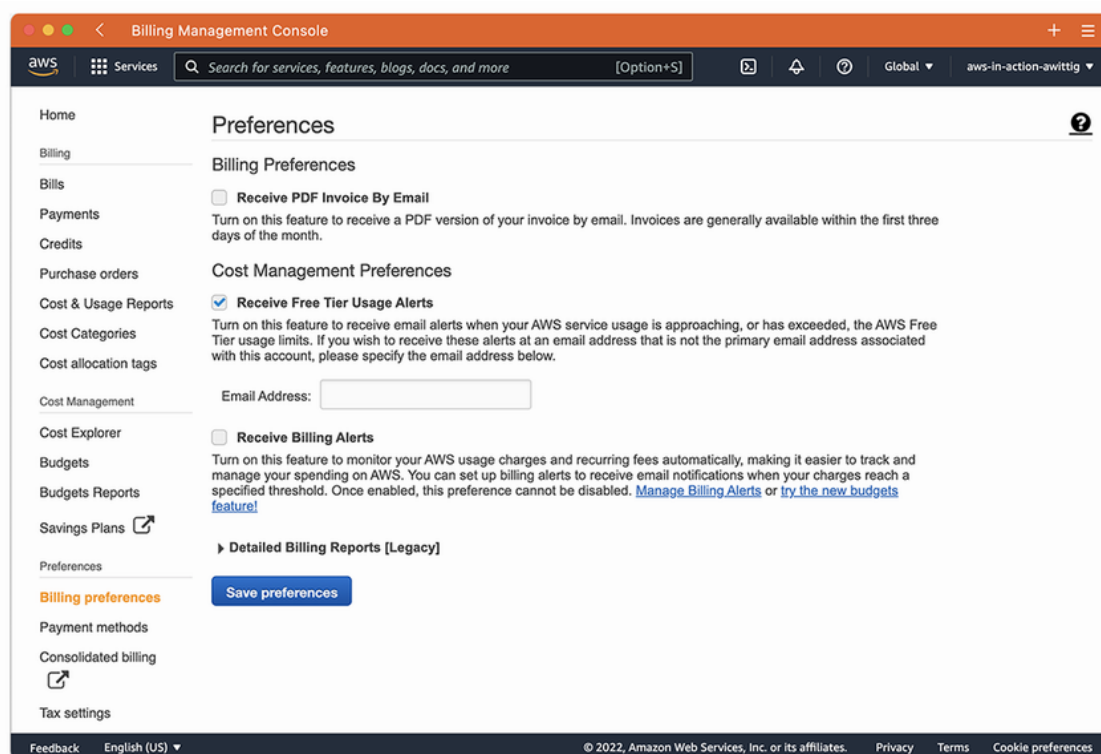


Figure 1.25 Creating a Free Tier usage alert to avoid unwanted costs

Next, you will create a budget that monitors costs incurred and forecasts costs to the end of the month, as described here:

1. Search and open Budgets in the Management Console's navigation bar.
2. Click Create Budget.
3. Select Cost Budget, as shown in figure 1.26.
4. Click Next.

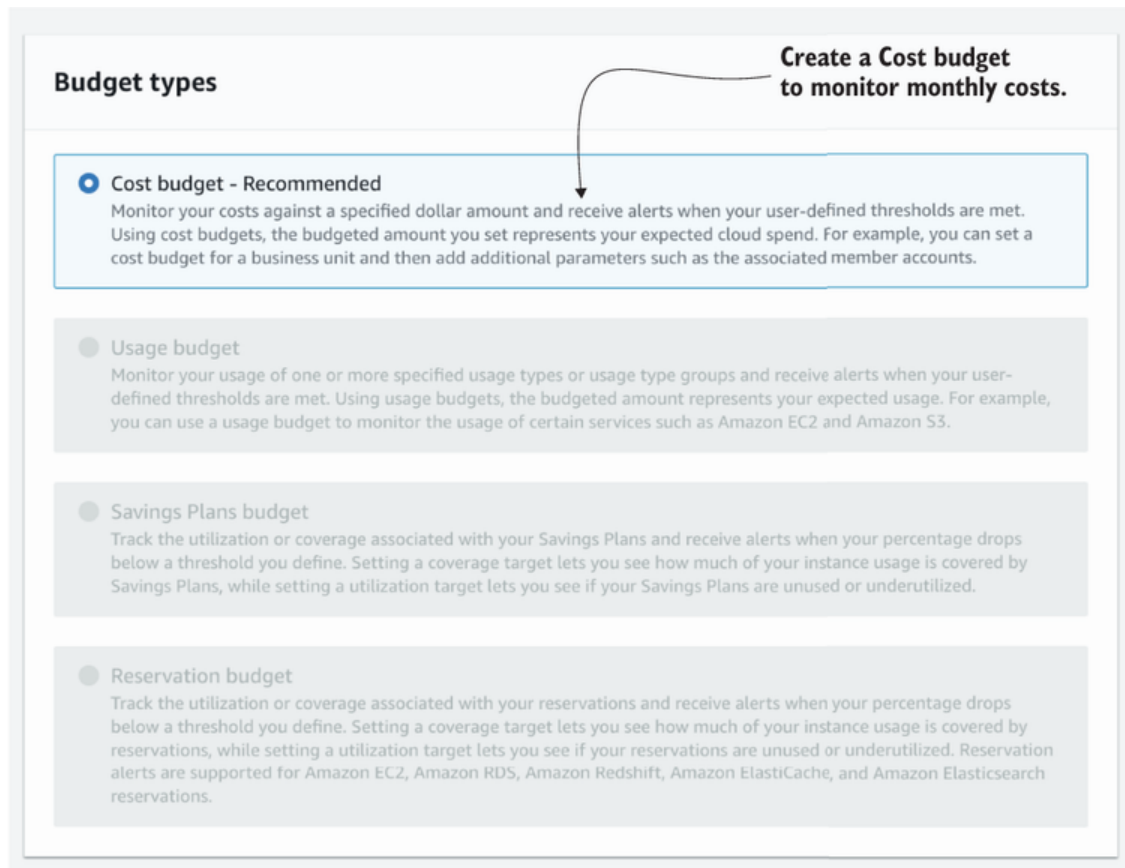


Figure 1.26 Creating a cost budget to monitor incurred and forecasted costs

Next, configure the cost budget as illustrated in figure 1.27 and described next:

1. Choose period Monthly.
2. Select Recurring Budget.
3. Choose the current month and year as start month.
4. Select Fixed to use the same budget for every month of the year.
5. Type in 5 to set the budget to \$5 per month.
6. Click Next.

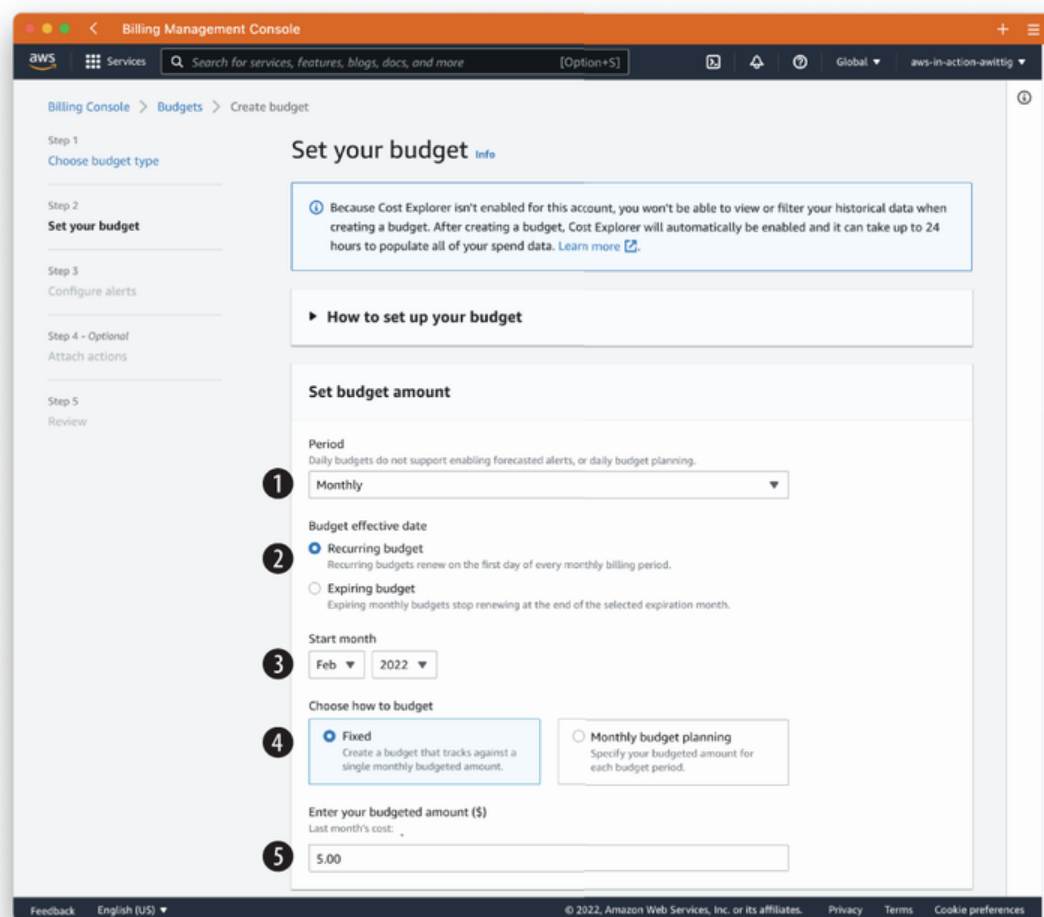


Figure 1.27 Creating a cost budget to monitor incurred and forecasted costs

After defining the budget, it is time to create alerts that will notify you via email as follows (figure 1.28):

1. Click Add Alert Threshold.
2. Type in 100% of budgeted amount.
3. Select trigger Actual to get notified when the incurred costs increase the budget.
4. Type in your email address.
5. Click Add Alert Threshold.
6. Type in 100% of budgeted amount.
7. Select trigger Forecasted to get notified when the forecasted costs increase the budget.
8. Type in your email address.
9. Click Next to proceed.
10. Review the budget, and click Create Budget.

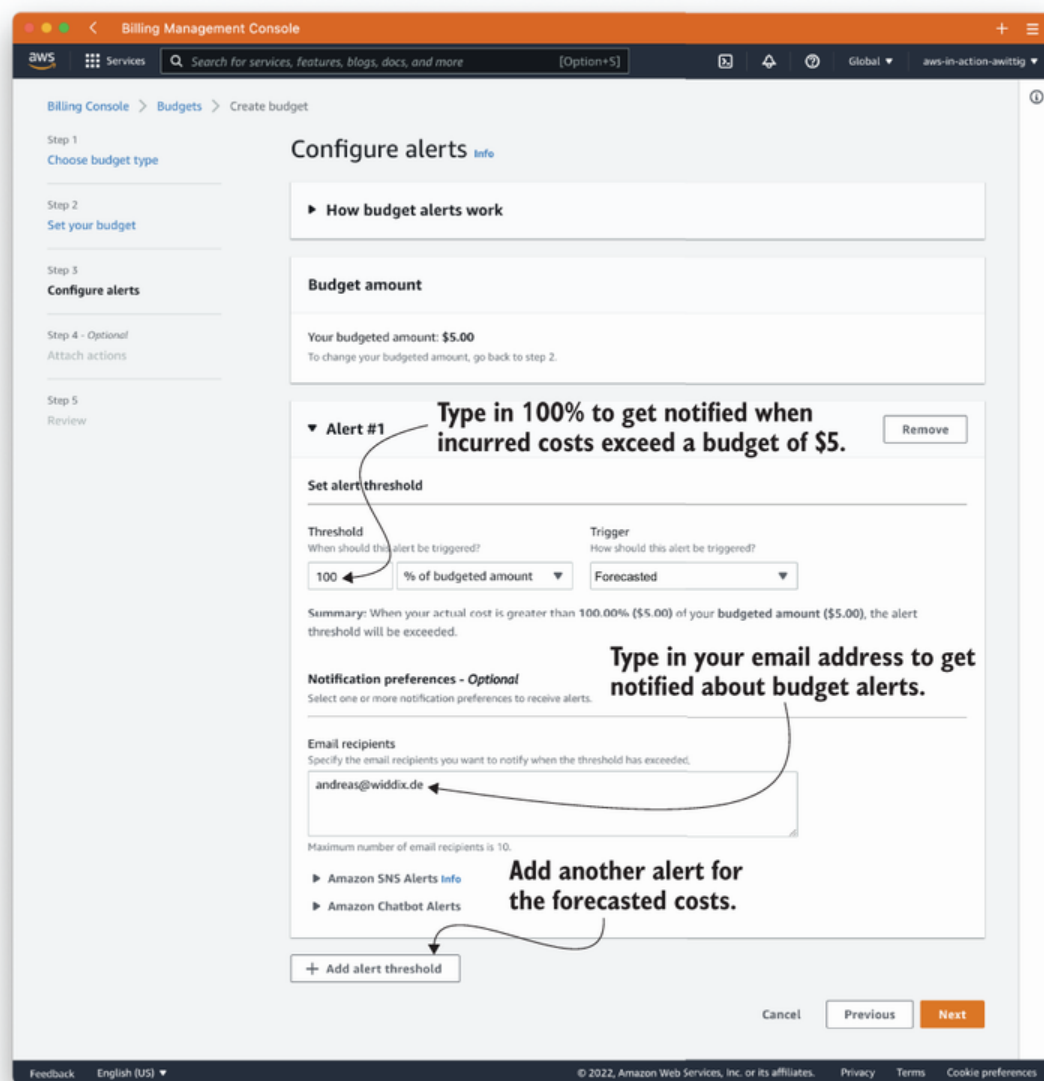


Figure 1.28 Define alerts to get notified when incurred or forecasted costs exceed your monthly budget.

That's it—you are ready to get started learning all the services and approaches we cover in the rest of this book. When you forget to shut down or delete resources when following our examples, AWS will notify you in case your monthly bill will exceed \$5.

## Summary

- Cloud computing, or the cloud, is a metaphor for supply and consumption of IT resources.
- Amazon Web Services (AWS) offers Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS).
- AWS is a platform of web services for computing, storing, and networking that work well together.

- Cost savings aren't the only benefit of using AWS. You'll also profit from an innovative and fast-growing platform with flexible capacity, fault-tolerant services, and a worldwide infrastructure.
- Almost any use case can be implemented on AWS, whether it's a widely used web application or a specialized enterprise application with an advanced networking setup.
- You can interact with AWS in many different ways. Control the different services by using the web-based user interface, use code to manage AWS programmatically from the command line or SDKs, or use blueprints to set up, modify, or delete your infrastructure on AWS.
- Pay-per-use is the pricing model for AWS services. Computing power, storage, and networking services are billed similarly to electricity.
- To create an AWS account, all you need is a telephone number and a credit card.
- Creating budget alerts allows you to keep track of your AWS bill and get notified whenever you exceed the Free Tier.