

AWS Well-Architected Framework

Publication date: **November 6, 2024** ([Document revisions](#))

The AWS Well-Architected Framework helps you understand the pros and cons of decisions you make while building systems on AWS. By using the Framework you will learn architectural best practices for designing and operating reliable, secure, efficient, cost-effective, and sustainable systems in the cloud.

Introduction

The AWS Well-Architected Framework helps you understand the pros and cons of decisions you make while building systems on AWS. Using the Framework helps you learn architectural best practices for designing and operating secure, reliable, efficient, cost-effective, and sustainable workloads in the AWS Cloud. It provides a way for you to consistently measure your architectures against best practices and identify areas for improvement. The process for reviewing an architecture is a constructive conversation about architectural decisions, and is not an audit mechanism. We believe that having well-architected systems greatly increases the likelihood of business success.

AWS Solutions Architects have years of experience architecting solutions across a wide variety of business verticals and use cases. We have helped design and review thousands of customers' architectures on AWS. From this experience, we have identified best practices and core strategies for architecting systems in the cloud.

The AWS Well-Architected Framework documents a set of foundational questions that help you to understand if a specific architecture aligns well with cloud best practices. The framework provides a consistent approach to evaluating systems against the qualities you expect from modern cloud-based systems, and the remediation that would be required to achieve those qualities. As AWS continues to evolve, and we continue to learn more from working with our customers, we will continue to refine the definition of well-architected.

This framework is intended for those in technology roles, such as chief technology officers (CTOs), architects, developers, and operations team members. It describes AWS best practices and strategies to use when designing and operating a cloud workload, and provides links to further implementation details and architectural patterns. For more information, see the [AWS Well-Architected homepage](#).

AWS also provides a service for reviewing your workloads at no charge. The [AWS Well-Architected Tool](#) (AWS WA Tool) is a service in the cloud that provides a consistent process for you to review and measure your architecture using the AWS Well-Architected Framework. The AWS WA Tool provides recommendations for making your workloads more reliable, secure, efficient, and cost-effective.

To help you apply best practices, we have created [AWS Well-Architected Labs](#), which provides you with a repository of code and documentation to give you hands-on experience implementing best practices. We also have teamed up with select AWS Partner Network (APN) Partners, who are members of the [AWS Well-Architected Partner program](#). These AWS Partners have deep AWS knowledge, and can help you review and improve your workloads.

Definitions

Every day, experts at AWS assist customers in architecting systems to take advantage of best practices in the cloud. We work with you on making architectural trade-offs as your designs evolve. As you deploy these systems into live environments, we learn how well these systems perform and the consequences of those trade-offs.

Based on what we have learned, we have created the AWS Well-Architected Framework, which provides a consistent set of best practices for customers and partners to evaluate architectures, and provides a set of questions you can use to evaluate how well an architecture is aligned to AWS best practices.

The AWS Well-Architected Framework is based on six pillars — operational excellence, security, reliability, performance efficiency, cost optimization, and sustainability.

Table 1. The pillars of the AWS Well-Architected Framework

Name	Description
Operational excellence	The ability to support development and run workloads effectively, gain insight into their operations, and to continuously improve supporting processes and procedures to deliver business value.
Security	The security pillar describes how to take advantage of cloud technologies to protect

Name	Description
	data, systems, and assets in a way that can improve your security posture.
Reliability	The reliability pillar encompasses the ability of a workload to perform its intended function correctly and consistently when it's expected to. This includes the ability to operate and test the workload through its total lifecycle . This paper provides in-depth, best practice guidance for implementing reliable workloads on AWS.
Performance efficiency	The ability to use computing resources efficiently to meet system requirements, and to maintain that efficiency as demand changes and technologies evolve.
Cost optimization	The ability to run systems to deliver business value at the lowest price point.
Sustainability	The ability to continually improve sustainability impacts by reducing energy consumption and increasing efficiency across all components of a workload by maximizing the benefits from the provisioned resources and minimizing the total resources required.

In the AWS Well-Architected Framework, we use these terms:

- A **component** is the code, configuration, and AWS Resources that together deliver against a requirement. A component is often the unit of technical ownership, and is decoupled from other components.
- The term **workload** is used to identify a set of components that together deliver business value. A workload is usually the level of detail that business and technology leaders communicate about.

- We think about **architecture** as being how components work together in a workload. How components communicate and interact is often the focus of architecture diagrams.
- **Milestones** mark key changes in your architecture as it evolves throughout the product lifecycle (design, implementation, testing, go live, and in production).
- Within an organization the **technology portfolio** is the collection of workloads that are required for the business to operate.
- The **level of effort** is categorizing the amount of time, effort, and complexity a task requires for implementation. Each organization needs to consider the size and expertise of the team and the complexity of the workload for additional context to properly categorize the level of effort for the organization.
 - **High:** The work might take multiple weeks or multiple months. This could be broken out into multiple stories, releases, and tasks.
 - **Medium:** The work might take multiple days or multiple weeks. This could be broken out into multiple releases and tasks.
 - **Low:** The work might take multiple hours or multiple days. This could be broken out into multiple tasks.


When architecting workloads, you make trade-offs between pillars based on your business context. These business decisions can drive your engineering priorities. You might optimize to improve sustainability impact and reduce cost at the expense of reliability in development environments, or, for mission-critical solutions, you might optimize reliability with increased costs and sustainability impact. In ecommerce solutions, performance can affect revenue and customer propensity to buy. Security and operational excellence are generally not traded-off against the other pillars.

On architecture

In on-premises environments, customers often have a central team for technology architecture that acts as an overlay to other product or feature teams to verify they are following best practice. Technology architecture teams typically include a set of roles such as: Technical Architect (infrastructure), Solutions Architect (software), Data Architect, Networking Architect, and Security Architect. Often these teams use [TOGAF](#) or the [Zachman Framework](#) as part of an enterprise architecture capability.

At AWS, we prefer to distribute capabilities into teams rather than having a centralized team with that capability. There are risks when you choose to distribute decision making authority, for

example, verifying that teams are meeting internal standards. We mitigate these risks in two ways. First, we have *practices* (ways of doing things, process, standards, and accepted norms) that focus on allowing each team to have that capability, and we put in place experts who verify that teams raise the bar on the standards they need to meet. Second, we implement *mechanisms* that carry out automated checks to verify standards are being met.

 “Good intentions never work, you need good mechanisms to make anything happen” — Jeff Bezos.

This means replacing a human's best efforts with mechanisms (often automated) that check for compliance with rules or process. This distributed approach is supported by the [Amazon leadership principles](#), and establishes a culture across all roles that *works back* from the customer. Working backward is a fundamental part of our innovation process. We start with the customer and what they want, and let that define and guide our efforts. Customer-obsessed teams build products in response to a customer need.

For architecture, this means that we expect every team to have the capability to create architectures and to follow best practices. To help new teams gain these capabilities or existing teams to raise their bar, we activate access to a virtual community of principal engineers who can review their designs and help them understand what AWS best practices are. The principal engineering community works to make best practices visible and accessible. One way they do this, for example, is through lunchtime talks that focus on applying best practices to real examples. These talks are recorded and can be used as part of onboarding materials for new team members.

AWS best practices emerge from our experience running thousands of systems at internet scale. We prefer to use data to define best practice, but we also use subject matter experts, like principal engineers, to set them. As principal engineers see new best practices emerge, they work as a community to verify that teams follow them. In time, these best practices are formalized into our internal review processes, and also into mechanisms that enforce compliance. The Well-Architected Framework is the customer-facing implementation of our internal review process, where we have codified our principal engineering thinking across field roles, like Solutions Architecture and internal engineering teams. The Well-Architected Framework is a scalable mechanism that lets you take advantage of these learnings.

By following the approach of a principal engineering community with distributed ownership of architecture, we believe that a Well-Architected enterprise architecture can emerge that is driven

by customer need. Technology leaders (such as CTOs or development managers), carrying out Well-Architected reviews across all your workloads will permit you to better understand the risks in your technology portfolio. Using this approach, you can identify themes across teams that your organization could address by mechanisms, training, or lunchtime talks where your principal engineers can share their thinking on specific areas with multiple teams.

General design principles

The Well-Architected Framework identifies a set of general design principles to facilitate good design in the cloud:

- **Stop guessing your capacity needs:** If you make a poor capacity decision when deploying a workload, you might end up sitting on expensive idle resources or dealing with the performance implications of limited capacity. With cloud computing, these problems can go away. You can use as much or as little capacity as you need, and scale in and out automatically.
- **Test systems at production scale:** In the cloud, you can create a production-scale test environment on demand, complete your testing, and then decommission the resources. Because you only pay for the test environment when it's running, you can simulate your live environment for a fraction of the cost of testing on premises.
- **Automate with architectural experimentation in mind:** Automation permits you to create and replicate your workloads at low cost and avoid the expense of manual effort. You can track changes to your automation, audit the impact, and revert to previous parameters when necessary.
- **Consider evolutionary architectures:** In a traditional environment, architectural decisions are often implemented as static, onetime events, with a few major versions of a system during its lifetime. As a business and its context continue to evolve, these initial decisions might hinder the system's ability to deliver changing business requirements. In the cloud, the capability to automate and test on demand lowers the risk of impact from design changes. This permits systems to evolve over time so that businesses can take advantage of innovations as a standard practice.
- **Drive architectures using data:** In the cloud, you can collect data on how your architectural choices affect the behavior of your workload. This lets you make fact-based decisions on how to improve your workload. Your cloud infrastructure is code, so you can use that data to inform your architecture choices and improvements over time.
- **Improve through game days:** Test how your architecture and processes perform by regularly scheduling game days to simulate events in production. This will help you understand where

improvements can be made and can help develop organizational experience in dealing with events.