Build modern applications on AWS

Manage less. Build fast. Innovate more.

Modern applications are changing how you deliver customer value

Organizations worldwide are focusing on innovation, customer satisfaction, and operational efficiency as top business priorities.[1] To achieve these goals, businesses will need their applications to play a key role in leading the way. However, many companies are building their applications the hard way—as they struggle to find a balance between managing technology and delivering new features.

While the cloud promises agility, it doesn't happen automatically. As organizations look to accelerate innovation, get more out of their data, and build new customer experiences, they need to modernize the way they build and operate applications. Modern applications are built with a combination of modular architecture patterns, serverless operational models, and agile developer processes.

In this eBook, we'll guide you through the three pathways that will help lay the foundation for modern application development in your own organization. We'll also explore how modern application development with AWS can help your organization innovate, reduce costs, accelerate time to market, and improve reliability.

[1] IDC: Future Enterprise Resiliency & Spending Survey - Wave 10, IDC, October, 2021

Modern apps empower digital innovators

Innovation means listening to your customers

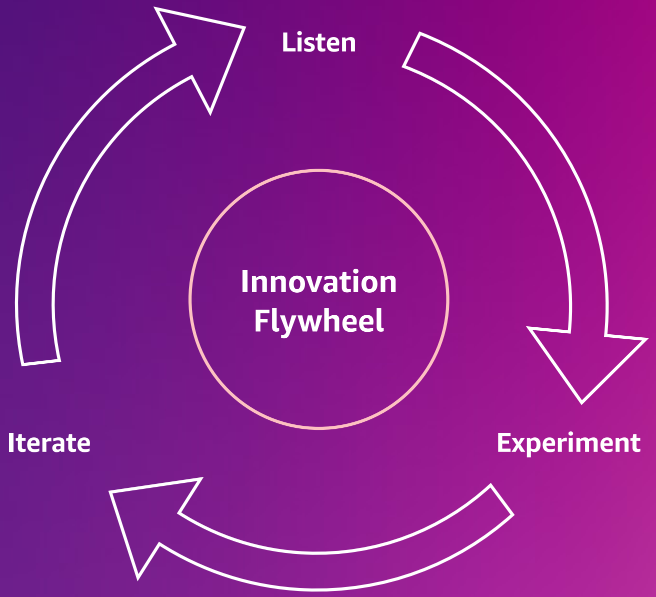
In a recent IDC Global CIO Advisory Board, the importance of retaining a customer-centric perspective was referenced as a critical element in driving successful digital innovation. A digital innovator does not present a solution in search of a problem. Rather, innovation is driven by viewing the customer's journey through their lens and building with the goal of removing points of frictions in their experience.

When you truly focus on your customer it means making business decisions by working backward from the customer's point of view. It means constantly evolving products and services to better deliver the outcomes that delight customers. It also means listening to what your customers truly care about so you can anticipate the experience that will keep them engaged as you continue inventing and iterating on their behalf. This approach is called the "innovation flywheel."

The basic idea is that the driver for any innovation begins with customer demand, improves with customer feedback, and repeats constantly (and profitably) until the demand changes and the whole cycle begins again. The faster your teams can get your own innovation flywheel spinning, the better you'll be able to build modern applications, and the more you will stand apart from competitors.

"Taking advantage of new MACH-ready architectures—microservices, API, cloud, and headless—allows organizations to quickly plug in new capabilities that fit their technology stack and address opportunities of growth as a digital-first customer-centric enterprise."[2]

Marci Maddox, Research Director, Digital Experience Strategies, IDC



[2] "IDC Perspective: Reframing the Digital-First Customer Journey: Key Takeaways from IDC's Global CIO Advisory Board — April 2022 Edition"

Building modern applications on AWS will make you **faster to market**. By accelerating the build and release cycle and offloading operational overhead, your developers can quickly build new features. You'll **increase innovation** with a modular architecture that frees teams to experiment with individual application components without risking the entire application. By automating test procedures and monitoring at every stage of the development lifecycle, you'll **improve reliability**. And you'll improve **total cost of ownership** (TCO) with a pay-for-value pricing model that reduces the cost of over-provisioning or paying for idle resources.

To build modern applications, you may need to reconsider the foundation on which they are built. While digital transformation can be dramatic at an organizational level, the process doesn't need to be brutal. Many organizations take an inspired leap to build new modern apps in the cloud, but plenty of others take a hybrid approach, which involves building apps to run both on premises and in the cloud. These paths don't have to be mutually exclusive. In fact, the process typically involves parallel paths—taking a team-by-team and workload-by-workload journey moving opportunistically, one step at a time.

50%

of IT spend will be directly allocated for digital transformation by 2023

67%

of executives believe they must pick up the pace to remain competitive

90%

of new applications are predicted to be cloud native by 2025

Three modernization pathways that generate real business value

Through our experience building applications for Amazon.com as well as from serving millions of AWS customers, we've observed three pathways that are customers can take for translating their vision of application modernization into a reality, generating value for business in the process.

1. **Build new apps with Kubernetes on AWS and extend existing Kubernetes to AWS for hybrid distribution.**
   Organizations running Kubernetes on premises or thinking about building new apps on Kubernetes, can run these workloads on AWS to increase reliability, security, and scalability. This approach provides server-level control, common operational tooling, and integrated access to the vast AWS service portfolio.

2. **Build new apps on managed containers and serverless architecture.**
   As organizations build new applications or features, we recommend managed containers, serverless technologies, and purpose-built databases to maximize agility and accelerate development while eliminating server management.

3. **Transform to a Modern DevOps model.**
   To create a cultural shift that delivers modern applications at scale, organizations can leverage DevOps services and tools to rapidly build and deliver products while maintaining a high bar on security and governance.

We'll explore each pathway in more detail, demonstrating how each can help lead to increased agility, lower costs, and building better apps that support business success. While you can modernize applications from any starting point, the outcome needs to be the same: applications that are secure, reliable, scalable, and quickly available for your customers and partners from the onset of your combined efforts.

Modern applications in practice

The three pathways of modern application development

Modern application development is a powerful approach to designing, building, and managing software in the cloud. This proven approach increases the agility of your development teams and the reliability and security of your applications, allowing you to build and release better products faster. We've worked with a wide range of organizations, from large, experienced operations teams looking to maintain maximum control with portable orchestration tools to resource-constrained teams preferring to offload operational tasks as they shift focus to building and iterating on applications.

From these experiences, we've identified the three solution pillars of modern application development to help you on your journey toward modernization:

1. **Build new and extend existing Kubernetes**
2. **Build on managed containers and serverless architecture**
3. **Transform to a Modern DevOps model**

Build new and extend existing Kubernetes

Today, more and more customers are taking a path of reinvention, building new applications that enable their organizations to take full advantage of the cloud. In fact, there's no single way to modernize; applications can coexist in different states (some highlighted in this eBook). Organizations worldwide are continuing to adopt and grow their use of containers as a lightweight and portable way to run and deploy many new applications. For many, containers have become the preferred route to support complex application architectures that require flexibility, rapid deployment, and continuous innovation in an environment that can run almost anywhere.

When moving to containerized applications, organizations need to consider where and how to invest their operational resources. Core to that consideration is determining the container orchestration platform best suited for the management of complex application deployments and the delivery of infrastructure automation. For those with a preference for open-source software (OSS), Kubernetes has gained significant adoption due to its vibrant ecosystem, community, consistent open-source APIs, and broad flexibility.

As an open-source project, Kubernetes is the ideal foundation for running containerized applications at scale in the cloud or in a hybrid cloud setup—providing a consistent, standardized way to scale and manage applications. Organizations planning to or already running Kubernetes—or thinking about building new applications on Kubernetes—can run them on Amazon Elastic Kubernetes Service (EKS) to increase reliability, security, and scalability, while retaining server-level control, common operational tooling, and integrated access to the broader AWS service portfolio.

For on-premises or hybrid deployments, Amazon EKS Anywhere brings the Amazon EKS experience to your data center, allowing you to retain latency sensitive or regulated applications in a private cloud on-premises all while leveraging a common toolset across environments.

Kubernetes is a powerful and extensible container orchestration technology that allows you to deploy and manage containerized applications at scale. The extensible nature of Kubernetes—and the large number of design choices available—makes building a tailored EKS cluster a long and time-consuming process. It involves integrating a wide range of open-source tools and AWS services that may require deep expertise in AWS and Kubernetes. To simplify that process, EKS Blueprints was built to help you integrate the landscape of Kubernetes tools. They make it easy to provision Amazon EKS to meet your specific application needs using familiar tools.

Why use Kubernetes today

Because Kubernetes is an open-source project, you can use it to run your containerized applications anywhere without needing to change your operational tooling. Kubernetes is maintained by a large community of volunteers and is always improving. Additionally, many other open-source projects and vendors build and maintain Kubernetes-compatible software that you can use to improve and extend your architecture.

Run applications at scale

Because Kubernetes lets you define complex containerized applications and run them at scale across a cluster of servers.

Seamlessly move applications

Using Kubernetes, containerized applications can be seamlessly moved from local development machines to production deployments on the cloud using the same operational tooling.

Run your apps anywhere

Run highly available and scalable Kubernetes clusters on AWS while maintaining full compatibility with your Kubernetes deployments running on-premises.

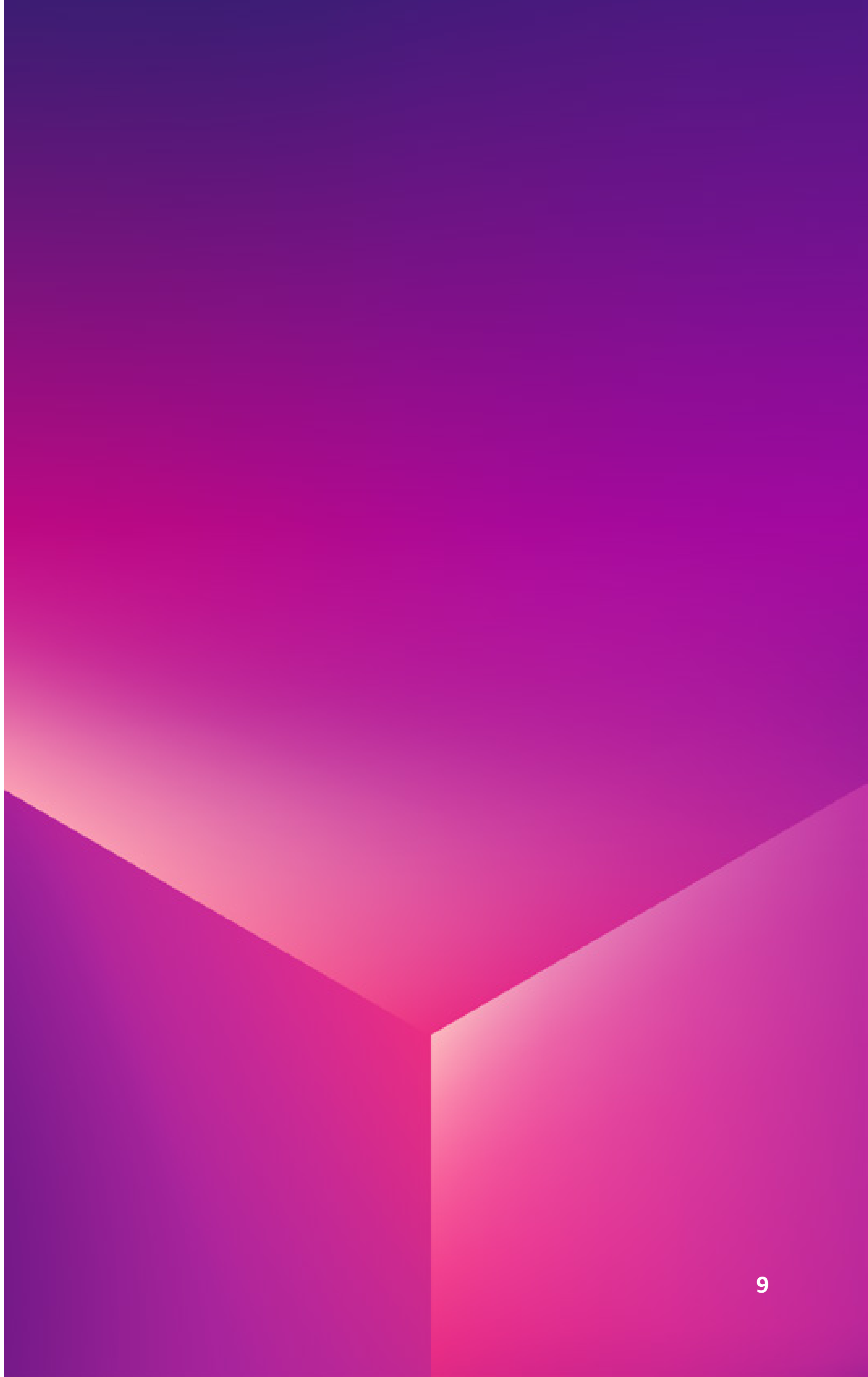
Add new functionality

Because Kubernetes is an open-source project, adding new functionality is easy. A large community of developers and companies build extensions, integrations, and plugins that help Kubernetes users do more.

Organizations possessing in-house Kubernetes and operational expertise can deploy Amazon EKS or Kubernetes on Amazon Elastic Compute Cloud (EC2) to leverage scalable, secure, and highly available infrastructure. This approach provides complete control over your compute instances, processes for deployment, maintenance, and scaling. With the most extensive global infrastructure, AWS Regions and Availability Zones provide maximum reliability, availability, and fault tolerance.

For organizations that want to use Amazon EKS without the complexity associated with running and scaling the underlying infrastructure, AWS Fargate is a serverless environment that removes the requirement to own, run, and manage the lifecycle of a compute infrastructure. With AWS Fargate, you can:

- **Eliminate operational overhead of scaling, patching, securing, and managing servers.**
- **Improve security through workload isolation by design (Amazon EKS pods run in their own dedicated runtime environment).**
- **Pay only for what you use (Fargate scales to closely match your resource needs).**



CUSTOMER SNAPSHOT

New Relic

New Relic is a leader in the observability space, providing engineers with a software-as-a-service (SaaS) platform that analyzes, troubleshoots, and optimizes their entire software stack. In 2020, New Relic began a journey to transform its business, moving from a host-based to a consumption-based pricing model. In addition, the company introduced a free service tier to all customers and prepared for a significant increase in demand on its platform. New Relic made the decision to migrate its entire platform to AWS to accommodate the projected trajectory of growth. In a period of just eight months, the business leveraged its own tools and processes, as well as AWS capabilities and offerings, to migrate over 20,000 servers and to refactor its entire services platform. The refactoring of the platform utilized **Amazon Elastic Kubernetes Service** (Amazon EKS). In this migration and refactoring program, New Relic saw material improvements in engineering efficiency and platform resiliency, along with a path for long-term scalability to support its aggressive growth targets.

Read more here ›

new relic

"Consuming AWS lets us focus on our core competencies, enabling us to release better products faster and more frequently."

Andrew Harnett,
Senior Director of Managed Services, New Relic

Build on managed container and serverless services

As we have already discussed, containers are a lightweight, portable way to run and deploy applications. Containerizing existing apps is often a first step in an organization's modernization journey. Many companies are moving applications to containers so their teams can spend their time and resources building applications instead of managing operational tasks. And increasingly, they are building their workloads on AWS managed services such as Amazon Elastic Kubernetes Service (Amazon EKS) or Amazon Elastic Container Service (Amazon ECS) with AWS Fargate. A managed container service helps to reduce operational burden while improving scalability, reliability, security, and availability. With managed container services on AWS, you no longer have to worry about managing containers. Instead, you can focus your resources on upskilling your teams to develop modern applications with serverless computing while accelerating application innovation.

CUSTOMER SNAPSHOT

Volkswagen Group

Global automotive manufacturer **Volkswagen Group** (Volkswagen) is building virtual reality (VR) and augmented reality (AR) applications to save time, reduce costs, and improve performance for use cases across the product lifecycle, from design reviews and training simulations to remote guidance in the factory. In its efforts to scale AR/VR, Volkswagen needed a faster, simpler, and more efficient way to prepare and deliver 3D content across its organization. The company developed a cloud architecture on which it can accomplish two things:

1. Automate Volkswagen's 3D data preparation pipelines for faster and simpler optimization of 3D models.
2. Test the ability to remotely render and stream 3D graphics to AR/VR headsets, replacing headsets tether-constrained to expensive workstations.

To achieve its goals, Volkswagen is using Amazon Web Services (AWS) to migrate specific VR rendering and 3D data optimization workloads to the cloud. It's also using a custom application, called Innoactive Portal, that was developed alongside VR software company **Innoactive** to deploy and manage VR applications. Using several AWS services, including **Amazon Elastic Container Service** (Amazon ECS)—a fully managed container orchestration service that makes it simple to deploy, manage, and scale containerized applications—Volkswagen is making its 3D data preparation pipeline more efficient and increasing its rendering power and speed in the cloud.

See full story here ›

VOLKSWAGEN GROUP

"It's simple and effective for us to use AWS because there are so many native services that support our goals and architecture drafts."

Jan-Paul Brückmann
Product Owner, Volkswagen Digital Realities Hub

Build new on serverless

Modernizing app development means adopting services, practices, and strategies that enable developers to build more agile applications. And with the speed and reliability of modern infrastructure, developers can deliver secure apps that scale from prototype to millions of users automatically, with the ability to innovate and respond to change faster.

Many modern applications are built serverless-first, a strategy that prioritizes the adoption of serverless services, so customers can increase agility throughout the application stack. With serverless technologies, you no longer have to manage physical servers and you'll benefit from automatic scaling, built-in high availability, and a pay-for-value billing model. Instead of worrying about managing and operating servers or runtimes, you can focus on product innovation while enjoying faster time-to-market.

In addition, front-end web and mobile tools and services can be built on top of AWS. The reliability of this infrastructure helps brands deliver secure, highly available apps that can scale automatically across the globe.

Key considerations for building scalable modern apps

Architectural patterns: microservices

Although your monolithic app might be easy to manage today, challenges often arise as you grow, including how to distribute ownership of the app across your teams. You can build a strong culture of ownership but still struggle to scale up if your application architecture includes hard dependencies that prevent teams from taking ownership of the final product. This is why we recommend building microservices architectures for apps that grow and change rapidly. Microservices are the architectural expression of a culture of ownership— they neatly divide complex applications into components that a single team can own and run independently.

With a monolith, you have many developers all pushing changes through a shared release pipeline, which causes friction at many points of the lifecycle. Upfront, during development, engineers need to coordinate their changes to make sure they're not breaking someone else's code. To upgrade a shared library to take advantage of a new feature, you need to convince everyone else to upgrade at the same time—a tough ask! And if you want to quickly push an important fix for your feature, you still need to merge it with changes in progress.

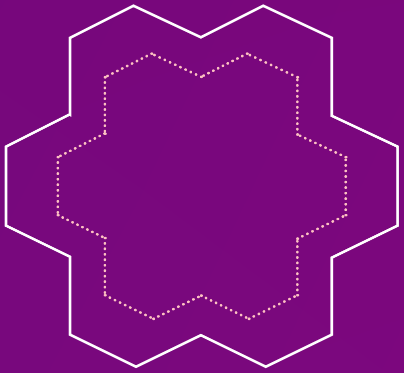
After development, you also face overhead when you're pushing the changes through the delivery pipeline. Even when making a one-line change in a tiny piece of code, engineers need to coordinate their changes ahead of time, merge their code, resolve conflicts within releases, rebuild the entire app, run all of the test suites, and redeploy once again.

With a microservices architecture, an application is made up of independent components that run each application process as a service. Services are built for business capabilities, and each service performs a single function. Because it runs independently and is managed by a single development team, each service can be updated, deployed, and scaled to meet the demand for specific functions of an application. For example, an online shopping cart built as a microservice can be used by many more users during a sale as it independently scales to support demand. Microservices communicate data with each other via well-defined interfaces, using lightweight APIs, events, or streams. Our customers are increasingly relying on event-driven architectures—those in which actions are triggered in response to changes in data—to improve application scalability and resiliency while also reducing costs.

Everything vs. One thing:

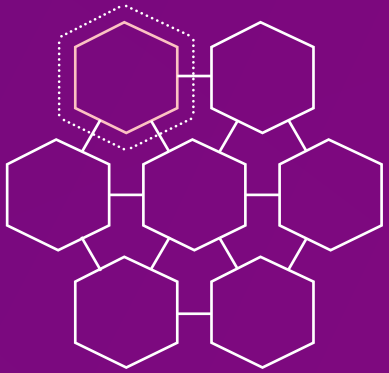
Two types of applications

Monolith apps



- Do everything
- Single app
- Must deploy entire app
- One database
- Organized around technology layers
- State in each runtime instance
- One technology stack for entire app

Microservices



- Do one thing
- Minimal function services
- Deployed separately, interact together
- Each has its own datastore
- Organized around business capabilities
- State is externalized
- Choice of technology for each microservice

CUSTOMER SNAPSHOT

Centrica

Centrica is a British multinational energy and services company that was looking to decrease its costs and increase its agility by changing the way its application was architected. The company opted to refactor to a microservices architecture and adopt a serverless strategy to achieve those goals. Building on that success, other teams in the organization have now also adopted the approach—serverless is now common across the organization. With serverless, Centrica is able to see and respond to customer issues in real time, something it had not been able to do before.

Watch the video ›

CUSTOMER SNAPSHOT

Liberty Mutual

With $40 billion in annual revenue, Liberty Mutual is the world's sixth-largest property and casualty insurance company and is an industry frontrunner in technological innovation. To achieve its goal of becoming a global digital company, the insurance provider focused on three main areas of transformation: customer-centricity, agility, and cloud-native development. To that end, the company made a strategic decision to pursue a serverless-first approach—a move designed to give it an edge in its competitive, global market. Liberty Mutual used AWS to migrate its on-premises systems to the cloud to modernize and drive business wide transformation. By using serverless architecture and letting AWS handle infrastructure management tasks like capacity provisioning and patching, Liberty Mutual reduced its operational burden and realized substantial cost savings. The company also used serverless solutions so that it could rapidly build more agile, high-quality applications. By eliminating operational overhead, the serverless architecture facilitates experimentation, empowering Liberty Mutual's teams to release quickly, get feedback, and iterate to get to market faster.

Read more here ›

Liberty Mutual. INSURANCE

"Our collaboration to figure things out feels like more than a customer-vendor relationship. It genuinely feels like AWS is part of our team."

Dave Anderson
Director of Technology, Liberty Mutual

As serverless as possible

As your architectural patterns and software delivery processes change, you will probably want to adopt an operational model that enables you to offload any activity that isn't a core competency of your business. To gain agility for rapid innovation, we recommend building microservices architecture, operating and deploying software using automation for things like monitoring, provisioning, cost management, deployment, security, and governance of apps. Choosing a serverless-first strategy—opting for serverless technologies wherever possible—enables you to maximize the operational benefits of AWS.

With a serverless operational model, you can build and run applications and services without provisioning and managing servers. This eliminates server management, provides flexible scaling, enables you to pay only for value, and automates high availability. This model lets you build and manage the aspects of your application that deliver customer value without having to worry about the underlying detail. Whether you are building net-new applications or migrating legacy, building with serverless primitives for compute, data, and integration will enable you to benefit from the most agility the cloud has to offer.

How do we define serverless at AWS?

When we say serverless, we mean the removal of the undifferentiated heavy lifting that is server operations. This is an important distinction because it allows you to focus on the building of the application rather than managing and scaling the infrastructure to support the application. The four tenets of a serverless operational model are:

1. **No server management –** There is no need to provision or maintain any servers. There is no software or runtime to install, maintain, or administer.
2. **Flexible scaling –** Your application can be scaled automatically or by adjusting its capacity through toggling the units of consumption (e.g., throughput, memory) rather than units of individual servers.
3. **Pay-for-value –** Instead of paying for server units, pay for what you value—consistent throughput or execution duration.
4. **Automated high availability –** Serverless provides built-in availability and fault tolerance. You don't need to architect for these capabilities since the services running the application provide them by default.

A serverless operational model is ideal for high-growth companies that want to innovate quickly. Serverless enables teams to move even faster and keep a laser focus on the activities that differentiate your business, so you can speed up your innovation flywheel.

Build on managed container and serverless services

Leveraging AWS Lambda and managed container services on AWS

With the rise of containers and serverless computing, instances are no longer your only cloud computing option. Choosing the optimal compute for your modern application starts with exploring several questions. Does self-managing infrastructure improve your business results? Do you have the expertise to do it? And will the extra effort ultimately drive value?

Increasingly, customers are choosing to offload server management by adopting container services like Amazon ECS and Amazon EKS or event-driven serverless compute services like **AWS Lambda**.

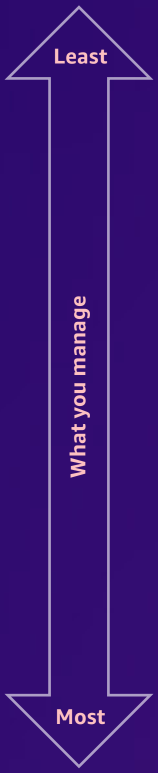
In reality, most customers use a combination of services.
About 80 percent of AWS containers customers have also adopted AWS Lambda.[3] Leveraging both options has its benefits, including fully managed services that have deep integration with AWS infrastructure, support for a wide range of use cases, abstraction from complexity, and a broad ecosystem of partners.

[3] Datalog State of Serverless, 2020

So how do you frame the decision?

Customers choose AWS Lambda when they have teams focused primarily on writing code and no limitations on existing instances or container platforms. AWS Lambda offers the maximum abstraction from infrastructure, enabling customers to release software at the fastest possible pace, which is why new applications are a great fit for AWS Lambda.

Customers often choose containers when they have existing containers investments, open source preferences for Kubernetes, or specific requirements for managing or configuring infrastructure. Containers are the most popular way to package code and are a great choice for modernizing legacy applications.



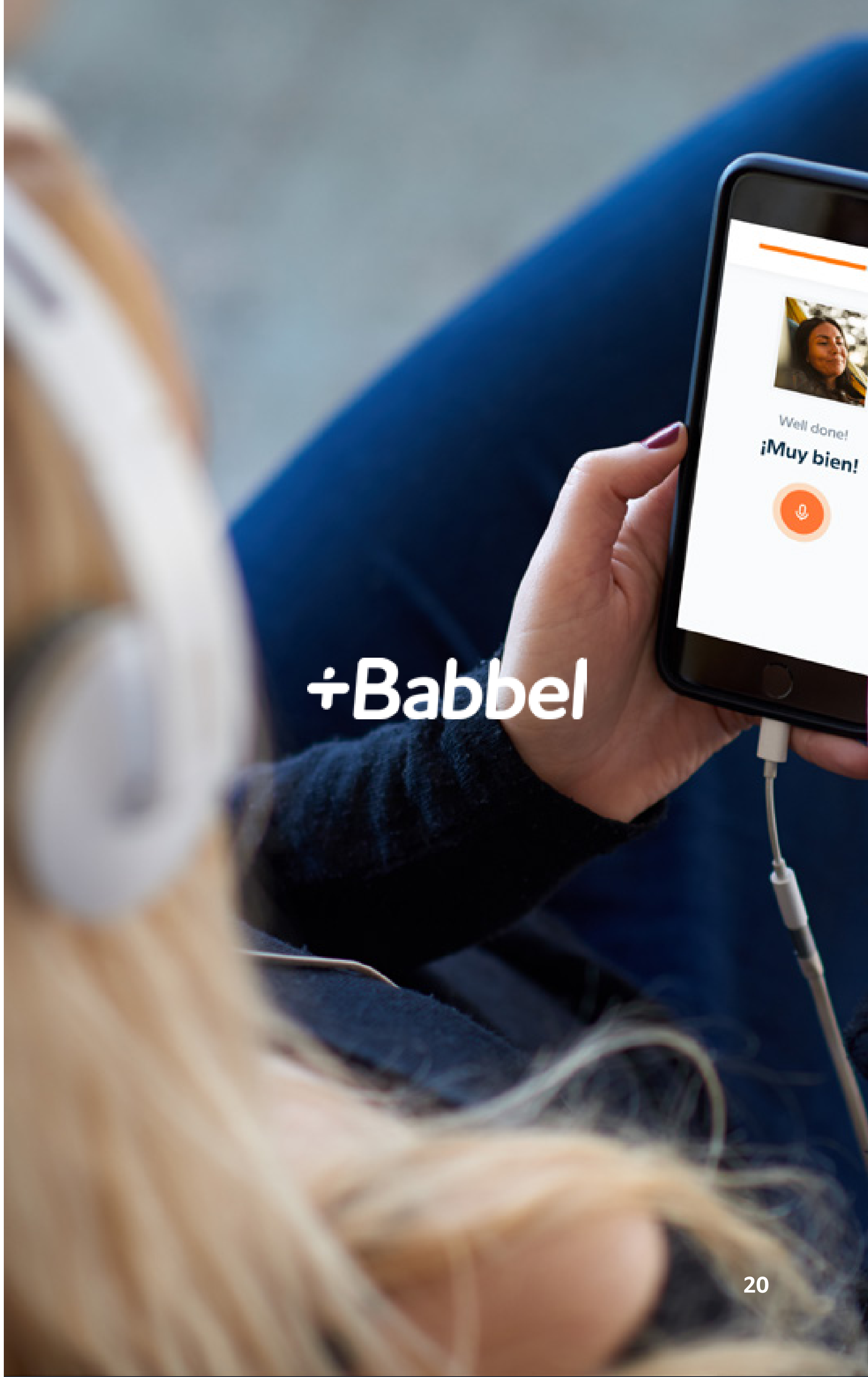
	AWS Manages	Customer Manages	
AWS Lambda Serverless functions	Data source integrations Physical hardware, software, networking, facilities Provisioning	Application code	
AWS Fargate Serverless containers	Container orchestration, provisioning Cluster scaling Physical hardware, host OS/kernel, networking, facilities	Application code Data source integrations Security config and updates, network config	Management tasks
Amazon ECS/ Amazon EKS Container management as a service	Container orchestration control plane Physical hardware, networking, facilities	Application code Data source integrations Security config and updates, network config, firewall	Work clusters Management tasks
Amazon EC2 Infrastructure as a service	Physical hardware, networking, facilities	Application code Data source integrations Scaling Management tasks	Security config and updates, network config Provisionining, managing scaling and patching of servers

CUSTOMER SNAPSHOT

Babbel

Babbel is a whole ecosystem of language learning offerings, including the world's best-selling language learning app. With more than 10 million subscriptions sold and over 60,000 courses for 14 languages, the company has created the No. 1 global destination for language learners. Babble has been running its platform on AWS from its first launch in 2007 and is frequently an early adopter of new service offerings from AWS. When Babbel was faced with an exponentially growing learner base, its original architecture was no longer scaling as needed to keep up with business growth. By migrating to Amazon ECS on AWS Fargate, and AWS Lambda, Babbel gained faster releases, rapid automated scalability, and reduced costs by utilizing computing workloads more efficiently—all while keeping up with the demand of its growing customer base.

See the full story ›



Taco Bell

One of America's most iconic food brands, Taco Bell, has over 7,000 restaurants in the United States. During the COVID-19 pandemic, Taco Bell needed to rapidly shift to meet consumer demand for delivery. The company runs nearly all of its infrastructure on AWS and uses serverless on AWS to focus less on managing servers and more on building business logic and data transformations to deliver real-time menu and restaurant information to its delivery partners.

Watch the video ›

"We have a menu that is very complex and has to be shared across multiple digital channels. Serverless fits that model really well."

Vadim Parizher
Vice President of Engineering and Analytics, Taco Bell

Coca Cola

When COVID-19 hit, consumer habits changed literally overnight. Coca-Cola responded rapidly with a no-touch drink dispensing experience to go along with its innovative Freestyle drink dispensers. Coca-Cola opted to build with AWS Lambda, and as a result, its team was able to focus on the application rather than security, latency, or scalability. Because with AWS Lambda, that's all built in. The new application launched in just 100 days, and now more than 52,000 machines have the touchless capability.

Watch the full story ›

"Low latency is essential to the user experience, which is why we're committed to a serverless solution on AWS."

Michael Connor
Chief Architect, Coca-Cola Freestyle Equipment Innovation Center

Transform to a modern DevOps model

Modern DevOps is the combination of cultural philosophies, practices, and tools that enable an organization to quickly and safely develop software, release it to production, and maintain its target availability and performance.

AWS has identified a set of common, broadly accepted practices that, when adopted, provide a mechanism for building a high-performing DevOps organization. This approach takes a simple idea—continuous improvement—and applies it to everything in the DevOps lifecycle from planning and code writing to deployment and monitoring. We call this approach Modern DevOps, and it's centered around bringing developers and operations closer by sharing operational tasks like compliance, observability, resilience, and infrastructure earlier in the development process and enhancing it with Artificial Intelligence and Machine Learning (AI/ML).

Developer agility: abstraction, automation, and standardization

Microservices architectures make teams agile and enable them to move faster, which means you're building more things that need to get released —great! However, you won't get new features to your customers faster if your build and release process does not keep up with your team's pace. Traditional development processes and release pipelines are slowed mainly by manual processes and custom code. Custom code is ultimately a long- term liability because it introduces the possibility for errors and long-term maintenance. Manual steps—from code changes and build requests to testing and deploying—are the greatest drag on release velocity. The solution involves abstraction, automation, and standardization.

To speed the development process, abstract away as much code as possible, particularly the lines of non-business logic code, required to develop and deliver production-ready apps. One way to do this is by employing frameworks and tooling that reduce the complexity of provisioning and configuring resources. This gives developers an ability to move quickly, while also enforcing best practices for security, privacy, reliability, performance, observability, and extensibility throughout the development process. Development frameworks give you confidence that your architecture will support your business growth long term.

By defining your software delivery process with best-practice templates, you can provide a standard for modeling and provisioning all infrastructure resources in a cloud environment. These “infrastructure as code” templates help teams get started on the right foot because the template provisions the entire technology stack for an application through code rather than using a manual process.

Through automation, you can create a repeatable motion that speeds up your software delivery lifecycle. Automating the release pipeline through continuous integration and continuous delivery (CI/CD) helps teams release high-quality code faster and more often. Teams that practice CI/CD ship more code, do it faster, and respond to issues quicker. In fact, according to the 2020 Puppet State of DevOps Report[4], teams that employ CI/CD have a failure rate that is five times lower, a commit-to-deploy rate that is 440 times faster, and a rate of deployment that is 46 times more frequent. Most notably, teams that practice CI/CD spend 44 percent more of their time creating new features and code instead of managing processes and tools.

CI/CD pipelines have become the new factory floor for building modern applications. At Amazon, we started using CI/CD to increase release velocity, and the results were dramatic—we have achieved millions of deployments per year, growing faster every year. To help companies benefit from our experience, we built a suite of developer tools based on the tools we use internally, so our customers can deliver code faster.

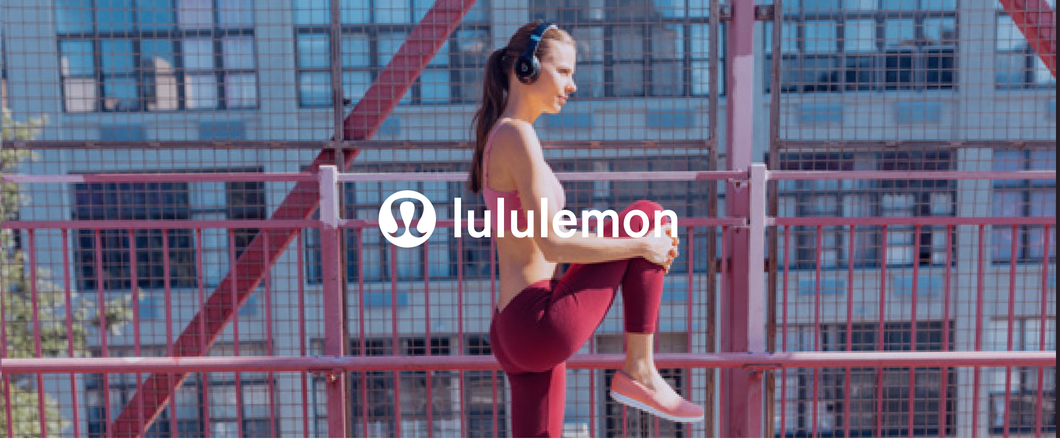
[4] Puppet 2020 State of Devops Report, 2020

A bit more detail

Continuous integration (CI) is a software development practice where developers regularly merge their code changes into a central repository, after which automated builds and tests are run. Continuous integration most often refers to the build or integration stage of the software release process and entails both an automation component (e.g., a CI or build service) and a cultural component (e.g., learning to integrate frequently).

Continuous delivery (CD) is a software development practice where code changes are automatically prepared for a release to production. Continuous delivery expands on continuous integration by deploying all code changes to a testing environment and/or a production environment after the build stage.

Learn how Amazon automates safe hands-off deployments ›



lululemon

With AWS, lululemon athletica can stand up development environments in minutes instead of days, automate its environments, and enable continuous integration and deployment. The Canadian company sells yoga-inspired apparel and other clothing at more than 350 locations throughout the world. lululemon runs its dev and test environments—as well as an upcoming mobile app—on the AWS Cloud.

"Any continuous integration and deployment pipeline should be automated, easy to manage, and discoverable, and that's exactly what we get using AWS. We get a level of simplicity and transparency we simply couldn't have in our previous on-premises environment."

Sam Keen
Director of Product Architecture, lululemon

The company decreased its time to build new production accounts from two days to a few minutes, using AWS CloudFormation templates and AWS CodePipeline. With that increased agility, lululemon dev teams can now experiment and get to the best solutions rather than having to settle for what they have resources committed to.

Watch the full story ›

HyperTrack

HyperTrack is a self-serve cloud platform for live location tracking through apps. When it launched in late 2015, HyperTrack needed to build a platform that could scale automatically to meet its anticipated growth, without reducing the time their developers spent on building new features. HyperTrack opted to use AWS Amplify for a mobile development framework and a serverless architecture to scale up and down automatically without engineering intervention. As a result, the company has realized a 30 percent cost savings compared to the architecture it was using before it switched to serverless. A big part of that savings comes from not needing operational resources to focus on server management —HyperTrack saves 40 hours of work, every single week, while managing millions of events.

See the full story ›

Transform to a modern DevOps model

Creating a culture of ownership: Manage less, innovate more with Modern DevOps

Innovation ultimately comes from people, and so enabling your people to deliver better customer outcomes is where modern application development starts. We use the concept of "products, not projects" to describe how this impacts team structure. Simply stated, it means that the teams that build products are responsible for running and maintaining them. It makes product teams accountable for the development of the whole product, not just a piece of it.

After more than a decade of building and running the highly scalable web application, Amazon.com, we've learned firsthand the importance of giving autonomy to our teams. When we gave our teams ownership of the complete application lifecycle, including taking customer input, planning the road map, and developing and operating the application, they became owners and felt empowered to develop and deliver new customer outcomes. Autonomy creates motivation, opens the door for creativity, and develops a risk-taking culture in an environment of trust.

While embracing a culture of ownership is not inherently technical, it remains one of the most challenging aspects of modern application development. Empowering teams to become product owners involves changing the mindset of your organization, the structure of your teams, and the work for which teams are responsible.

For most organizations, IT falls into one of two camps. It is viewed as a strategic competitive weapon or, more commonly, as a cost center necessary to support business growth.

Building a culture of innovation

1 Start with the customers – Every innovation should start with a customer need and ultimately delight your customers. Prioritize relentlessly to focus on customer demand.

2 Hire builders and let them build – Remove any obstacles that slow the process of building and releasing products and features for customers. The faster you iterate, the faster your flywheel spins.

3 Support builders with a belief system – Don't pay lip service to innovation. Live and breathe innovation in all areas of business from leadership, to sales, to support.

Manage less, innovate more

Modern applications create competitive differentiation by enabling rapid innovation. By adopting services, practices, and strategies that underscore speed and agility, you can shift resources from business as usual to differentiating activities with deep customer value. Experiment more and turn ideas into releases faster. Foster an environment where builders spend more time building, and less time managing. Modern applications are how organizations, including Amazon, innovate with speed and agility.

Why build modern applications on AWS?

Faster to Market	By speeding up the build and release cycle and offloading operational overhead, developers can quickly build new features. Automated test and release processes reduce error rates, so products are market-ready faster.	**See the proof:** **Urbanbase launches services 20x faster with AWS**
Increase Innovation	With a modular architecture, changes to any individual application component can be made quickly and with a lower risk to the whole application, so teams can experiment with new ideas more often.	**See the proof:** **iRobot uses AWS Lambda and the AWS IoT platform to manage its Roomba robotic vacuum cleaners**
Improve Reliability	By automating test procedures and monitoring at every stage of the development lifecycle, modern applications are reliable at deployment. Any issues can be evaluated and addressed in real time.	**See the proof:** **Siemens decreases customer control system alerts by 90 percent and reduces infrastructure costs by 85 percent**
Improve TCO	With a pay-for-value pricing model, modern applications reduce the cost of over-provisioning or paying for idle resources. By offloading infrastructure management, maintenance costs are also lower.	**See the proof:** **Save up to 80 percent on app maintenance with AWS Lambda**

Replatform to Managed Container Services

80% of all containerized applications running in the cloud run on AWS. 84% of all Kubernetes workloads in the cloud run on AWS.[5]

Resources

Amazon ECS workshop
Amazon EKS workshop
AWS AppRunner workshop

Recommended Training (Classroom)

Running Containers on Amazon Elastic Kubernetes Service (Amazon EKS)

Recommended Training (Online)

Amazon Elastic Container Service (ECS) Primer

[5] Nucleas, 2019

Build New Modern Applications with Serverless Technologies and Tools

Save up to 80% time on maintenance and approximately 70% on development when adopting a serverless-first strategy for building modern applications.[6]

Resources

Innovator Island – Serverless web application development workshop.
Build a serverless app video tutorial.

Recommended Training (Classroom)

Advanced Developing on AWS

Recommended Training (Online)

Architecting Serverless Solutions

[6] Deloitte, 2020

Transform to a Modern DevOps Model

60 percent of teams with more evolved DevOps practices fully remediate security vulnerabilities in less than a day.[7]

Resources

The Amazon Builder's Library
AWS DevOps services

Recommended Training (Classroom)

DevOps Engineering on AWS

Recommended Training (Online)

Getting started with DevOps on AWS

[7] Puppet, 2020

Learn more about building modern applications on AWS ›

Talk to an expert to implement the best practices of modern application development ›

Connect with an AWS partner to accelerate your modernization projects ›