**Azure Fundamentals part 3: Describe core solutions and management tools on Azure**

* Choose the correct Azure Artificial Intelligence service to address different kinds of business challenges.
* Choose the best software development process tools and services for a given business scenario.
* Choose the correct cloud monitoring service to address different kinds of business challenges.
* Choose the correct Azure management tool to address different kinds of technical needs and challenges.
* Choose the right serverless computing technology for your business scenario.
* Choose the best Azure IoT service for a given business scenario.

# Choose the best tools to help organizations build better solutions

## Product options

At a high level, there are three primary offerings, each of which is aimed at a specific audience and use case and provides a diverse set of tools, services, programmatic APIs, and more.

### Azure DevOps Services

Azure DevOps Services is a suite of services that address every stage of the software development lifecycle.

* **Azure Repos** is a centralized source-code repository where software development, DevOps engineering, and documentation professionals can publish their code for review and collaboration.
* **Azure Boards** is an agile project management suite that includes Kanban boards, reporting, and tracking ideas and work from high-level epics to work items and issues.
* **Azure Pipelines** is a CI/CD pipeline automation tool.
* **Azure Artifacts** is a repository for hosting artifacts, such as compiled source code, which can be fed into testing or deployment pipeline steps.
* **Azure Test Plans** is an automated test tool that can be used in a CI/CD pipeline to ensure quality before a software release.

### GitHub and GitHub Actions

GitHub is arguably the world's most popular code repository for open-source software. Git is a decentralized source-code management tool, and GitHub is a hosted version of Git that serves as the primary remote. GitHub builds on top of Git to provide related services for coordinating work, reporting and discussing issues, providing documentation, and more. It offers the following functionality:

* It's a shared source-code repository, including tools that enable developers to perform code reviews by adding comments and questions in a web view of the source code before it can be merged into the main code base.
* It facilitates project management, including Kanban boards.
* It supports issue reporting, discussion, and tracking.
* It features CI/CD pipeline automation tooling.
* It includes a wiki for collaborative documentation.
* It can be run from the cloud or on-premises

Most relevant for this module, GitHub Actions enables workflow automation with triggers for many lifecycle events. One such example would be automating a CI/CD toolchain.

### Azure DevTest Labs

Azure DevTest Labs provides an automated means of managing the process of building, setting up, and tearing down virtual machines (VMs) that contain builds of your software projects. This way, developers and testers can perform tests across a variety of environments and builds. And this capability isn't limited to VMs. Anything you can deploy in Azure via an ARM template can be provisioned through DevTest Labs. Provisioning pre-created lab environments with their required configurations and tools already installed is a huge time saver for quality assurance professionals and developers.

Suppose you need to test a new feature on an old version of an operating system. Azure DevTest Labs can set up everything automatically upon request. After the testing is complete, DevTest Labs can shut down and deprovision the VM, which saves money when it's not in use. To control costs, the management team can restrict how many labs can be created, how long they run, and so on.

# Use Azure DevTest Labs to manage testing environments

# Choose the best monitoring service for visibility, insight, and outage mitigation

## The product options

At a high level, there are three primary Azure monitoring offerings, each of which is aimed at a specific audience and use case and provides a diverse set of tools, services, programmatic APIs, and more.

### Azure Advisor

[Azure Advisor](https://azure.microsoft.com/services/advisor/) evaluates your Azure resources and makes recommendations to help improve reliability, security, and performance, achieve operational excellence, and reduce costs. Advisor is designed to help you save time on cloud optimization. The recommendation service includes suggested actions you can take right away, postpone, or dismiss.

The recommendations are available via the Azure portal and the API, and you can set up notifications to alert you to new recommendations.

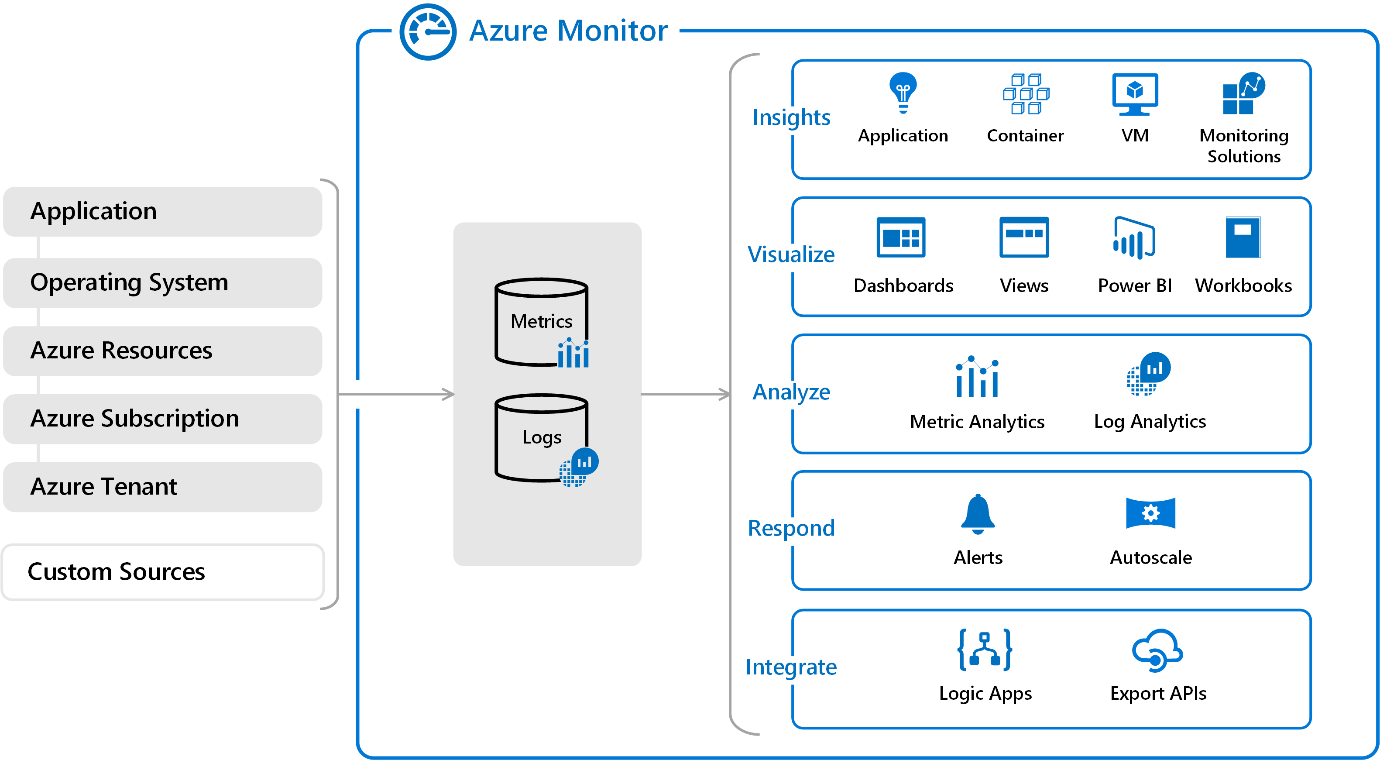
When you're in the Azure portal, the Advisor dashboard displays personalized recommendations for all your subscriptions, and you can use filters to home in on recommendations for specific subscriptions, resource groups, or services. The recommendations are divided into five categories:

* **Reliability**: Used to ensure and improve the continuity of your business-critical applications.
* **Security**: Used to detect threats and vulnerabilities that might lead to security breaches.
* **Performance**: Used to improve the speed of your applications.
* **Cost**: Used to optimize and reduce your overall Azure spending.
* **Operational Excellence**: Used to help you achieve process and workflow efficiency, resource manageability, and deployment best practices.

### Azure Monitor

[Azure Monitor](https://azure.microsoft.com/services/monitor/) is a platform for collecting, analyzing, visualizing, and potentially taking action based on the metric and logging data from your entire Azure and on-premises environment.

The following diagram illustrates just how comprehensive Azure Monitor is.



* On the left is a list of the sources of logging and metric data that can be collected at every layer in your application architecture, from application to operating system and network.
* In the center, you can see how the logging and metric data is stored in central repositories.
* On the right, the data is used in a number of ways. You can view real-time and historical performance across each layer of your architecture, or aggregated and detailed information. The data is displayed at different levels for different audiences. You can view high-level reports on the Azure Monitor Dashboard or create custom views by using Power BI and Kusto queries.

Additionally, you can use the data to help you react to critical events in real time, through alerts delivered to teams via SMS, email, and so on. Or you can use thresholds to trigger autoscaling functionality to scale up or down to meet the demand.

Some popular products such as Azure Application Insights, a service for sending telemetry information from application source code to Azure, uses Azure Monitor under the hood. With Application Insights, your application developers can take advantage of the powerful data-analysis platform in Azure Monitor to gain deep insights into an application's operations and diagnose errors without having to wait for users to report them.

### Azure Service Health

[Azure Service Health](https://azure.microsoft.com/features/service-health/) provides a personalized view of the health of the Azure services, regions, and resources you rely on. The status.azure.com website, which displays only major issues that broadly affect Azure customers, doesn't provide the full picture. But Azure Service Health displays both major and smaller, localized issues that affect you. Service issues are rare, but it's important to be prepared for the unexpected. You can set up alerts that help you triage outages and planned maintenance. After an outage, Service Health provides official incident reports, called root cause analyses (RCAs), which you can share with stakeholders.

Service Health helps you keep an eye on several event types:

* **Service issues** are problems in Azure, such as outages, that affect you right now. You can drill down to the affected services, regions, updates from your engineering teams, and find ways to share and track the latest information.
* **Planned maintenance** events can affect your availability. You can drill down to the affected services, regions, and details to show how an event will affect you and what you need to do. Most of these events occur without any impact to you and aren't shown here. In the rare case that a reboot is required, Service Health allows you to choose when to perform the maintenance to minimize the downtime.
* **Health advisories** are issues that require you to act to avoid service interruption, including service retirements and breaking changes. Health advisories are announced far in advance to allow you to plan.

1100 XP

# Choose the best tools for managing and configuring your Azure environment

## Your product options

Microsoft offers a variety of tools and services to manage your cloud environment, each aimed at different scenarios and users.

### The Azure portal

By using the Azure portal, a web-based user interface, you can access virtually every feature of Azure. The Azure portal provides a friendly, graphical UI to view all the services you're using, create new services, configure your services, and view reports. The Azure portal is how most users first experience Azure. But, as your Azure usage grows, you'll likely choose a more repeatable code-centric approach to managing your Azure resources.

### The Azure mobile app

The Azure mobile app provides iOS and Android access to your Azure resources when you're away from your computer. With it, you can:

* Monitor the health and status of your Azure resources.
* Check for alerts, quickly diagnose and fix issues, and restart a web app or virtual machine (VM).
* Run the Azure CLI or Azure PowerShell commands to manage your Azure resources.

### Azure PowerShell

Azure PowerShell is a shell with which developers and DevOps and IT professionals can execute commands called cmdlets (pronounced command-lets). These commands call the Azure Rest API to perform every possible management task in Azure. Cmdlets can be executed independently or combined into a script file and executed together to orchestrate:

* The routine setup, teardown, and maintenance of a single resource or multiple connected resources.
* The deployment of an entire infrastructure, which might contain dozens or hundreds of resources, from imperative code.

Capturing the commands in a script makes the process repeatable and automatable.

Azure PowerShell is available for Windows, Linux, and Mac, and you can access it in a web browser via Azure Cloud Shell.

Windows PowerShell has helped Windows-centric IT organizations automate many of their on-premises operations for years, and these organizations have built up a large catalog of custom scripts and cmdlets, as well as expertise.

### The Azure CLI

The Azure CLI command-line interface is an executable program with which a developer, DevOps professional, or IT professional can execute commands in Bash. The commands call the Azure Rest API to perform every possible management task in Azure. You can run the commands independently or combined into a script and executed together for the routine setup, teardown, and maintenance of a single resource or an entire environment.

In many respects, the Azure CLI is almost identical to Azure PowerShell in what you can do with it. Both run on Windows, Linux, and Mac, and can be accessed in a web browser via Cloud Shell. The primary difference is the syntax you use. If you're already proficient in PowerShell or Bash, you can use the tool you prefer.

### ARM templates

Although it's possible to write imperative code in Azure PowerShell or the Azure CLI to set up and tear down one Azure resource or orchestrate an infrastructure comprising hundreds of resources, there's a better way implement this functionality.

By using Azure Resource Manager templates (ARM templates), you can describe the resources you want to use in a declarative JSON format. The benefit is that the entire ARM template is verified before any code is executed to ensure that the resources will be created and connected correctly. The template then orchestrates the creation of those resources in parallel. That is, if you need 50 instances of the same resource, all 50 instances are created at the same time.

Ultimately, the developer, DevOps professional, or IT professional needs only to define the desired state and configuration of each resource in the ARM template, and the template does the rest. Templates can even execute PowerShell and Bash scripts before or after the resource has been set up.

# Choose the best Azure serverless technology for your business scenario

# Identify the product options

* 5 minutes

Serverless computing is a cloud-hosted execution environment that runs your code but abstracts the underlying hosting environment. The term serverless computing is a misnomer. After all, there is a server (or a group of servers) that executes your code or desired functionality.

The key idea is that you're not responsible for setting up or maintaining the server. You don't have to worry about scaling it when there's increased demand, and you don't have to worry about outages. The cloud vendor takes care of all maintenance and scaling concerns for you.

You create an instance of the service, and you add your code. No infrastructure configuration or maintenance is required, or even allowed. You configure your serverless apps to respond to events. An event could be a REST endpoint, a periodic timer, or even a message received from another Azure service. The serverless app runs only when it's triggered by an event. Scaling and performance are handled automatically, and you're billed only for the resources you use. You don't even need to reserve resources.

Serverless computing is ordinarily used to handle back-end scenarios. In other words, serverless computing is responsible for sending message from one system to another, or processing messages that were sent from other systems. It's not used for user-facing systems but, rather, it works in the background.

In this module, we'll cover two Azure serverless computing services: Azure Functions and Azure Logic Apps.

## Azure Functions

With the [Azure Functions](https://azure.microsoft.com/services/functions/) service, you can host a single method or function by using a popular programming language in the cloud that runs in response to an event. An example of an event might be an HTTP request, a new message on a queue, or a message on a timer.

Because of its atomic nature, Azure Functions can serve many purposes in an application's design. Functions can be written in many common programming languages, such as C#, Python, JavaScript, Typescript, Java, and PowerShell.

Azure Functions scales automatically, and charges accrue only when a function is triggered. These qualities make Azure Functions a solid choice when demand is variable. For example, you might be receiving messages from an IoT solution that monitors a fleet of delivery vehicles. You'll likely have more data arriving during business hours. Azure Functions can scale out to accommodate these busier times.

An Azure function is a stateless environment. A function behaves as if it's restarted every time it responds to an event. This feature is ideal for processing incoming data. And if state is required, the function can be connected to an Azure storage account.

Azure Functions can perform orchestration tasks by using an extension called Durable Functions, which allows developers to chain functions together while maintaining state.

The Azure Functions solution is ideal when you're concerned only with the code that's running your service and not the underlying platform or infrastructure. You use Functions most commonly when you need to perform work in response to an event. You do this often via a REST request, timer, or message from another Azure service, and when that work can be completed quickly, within seconds or less.

## Azure Logic Apps

[Logic Apps](https://azure.microsoft.com/services/logic-apps/) is a low-code/no-code development platform hosted as a cloud service. The service helps you automate and orchestrate tasks, business processes, and workflows when you need to integrate apps, data, systems, and services across enterprises or organizations. Logic Apps simplifies how you design and build scalable solutions, whether in the cloud, on-premises, or both. This solution covers app integration, data integration, system integration, enterprise application integration (EAI), and business-to-business (B2B) integration.

Azure Logic Apps is designed in a web-based designer and can execute logic that's triggered by Azure services without your having to write any code. You build an app by linking triggers to actions with connectors. A trigger is an event, such as a timer, that causes an app to execute, a new message to be sent to a queue, or an HTTP request. An action is a task or step that can execute. There are logic actions such as those you would find in most programming languages. Examples of actions include working with variables, decision statements and loops, and tasks that parse and modify data.

To build enterprise integration solutions with Azure Logic Apps, you can choose from a growing gallery of over 200 connectors. The gallery includes services such as Salesforce, SAP, Oracle DB, and file shares.

If you can't find the action or connector you need, you can build your own by using custom code.

## What are the differences between these services?

You can call Azure Functions from Azure Logic Apps, and vice versa. The primary difference between the two services is their intent. Azure Functions is a serverless compute service, and Azure Logic Apps is intended to be a serverless orchestration service. Although you can use Azure Functions to orchestrate a long-running business process that involves various connections, this was not its primary use case when it was designed.

Additionally, the two services are priced differently. Azure Functions pricing is based on the number of executions and the running time of each execution. Logic Apps pricing is based on the number of executions and the type of connectors that it utilizes.

900 XP

# Choose the best Azure IoT service for your application

[Azure IoT Hub](https://azure.microsoft.com/services/iot-hub/) is a managed service that's hosted in the cloud and that acts as a central message hub for bi-directional communication between your IoT application and the devices it manages. You can use Azure IoT Hub to build IoT solutions with reliable and secure communications between millions of IoT devices and a cloud-hosted solution back end. You can connect virtually any device to your IoT hub.

The IoT Hub service supports communications both from the device to the cloud and from the cloud to the device. It also supports multiple messaging patterns, such as device-to-cloud telemetry, file upload from devices, and request-reply methods to control your devices from the cloud. After an IoT hub receives messages from a device, it can route that message to other Azure services.

From a cloud-to-device perspective, IoT Hub allows for command and control. That is, you can have either manual or automated remote control of connected devices, so you can instruct the device to open valves, set target temperatures, restart stuck devices, and so on.

IoT Hub monitoring helps you maintain the health of your solution by tracking events such as device creation, device failures, and device connections.

## Azure IoT Central

[Azure IoT Central](https://azure.microsoft.com/services/iot-central/) builds on top of IoT Hub by adding a dashboard that allows you to connect, monitor, and manage your IoT devices. The visual user interface (UI) makes it easy to quickly connect new devices and watch as they begin sending telemetry or error messages. You can watch the overall performance across all devices in aggregate, and you can set up alerts that send notifications when a specific device needs maintenance. Finally, you can push firmware updates to the device.

To help you get up and running quickly, IoT Central provides starter templates for common scenarios across various industries, such as retail, energy, healthcare, and government. You then customize the design starter templates directly in the UI by choosing from existing themes or creating your own custom theme, setting the logo, and so on. With IoT Central, you can tailor the starter templates for the specific data that's sent from your devices, the reports you want to see, and the alerts you want to send.

## Azure Sphere

[Azure Sphere](https://azure.microsoft.com/services/azure-sphere/) creates an end-to-end, highly secure IoT solution for customers that encompasses everything from the hardware and operating system on the device to the secure method of sending messages from the device to the message hub. Azure Sphere has built-in communication and security features for internet-connected devices.

Azure Sphere comes in three parts:

* The first part is the Azure Sphere micro-controller unit (MCU), which is responsible for processing the operating system and signals from attached sensors. The following image displays the Seed Azure Sphere MT3620 Development Kit MCU, one of several different starter kits that are available for prototyping and developing Azure Sphere applications.



* The second part is a customized Linux operating system (OS) that handles communication with the security service and can run the vendor's software.
* The third part is Azure Sphere Security Service, also known as AS3. Its job is to make sure that the device has not been maliciously compromised. When the device attempts to connect to Azure, it first must authenticate itself, per device, which it does by using certificate-based authentication. If it authenticates successfully, AS3 checks to ensure that the device hasn't been tampered with. After it has established a secure channel of communication, AS3 pushes any OS or approved customer-developed software updates to the device.