



Basic Computing Concepts

© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Welcome to Basic Computing Concepts.

What you will learn

At the core of the lesson

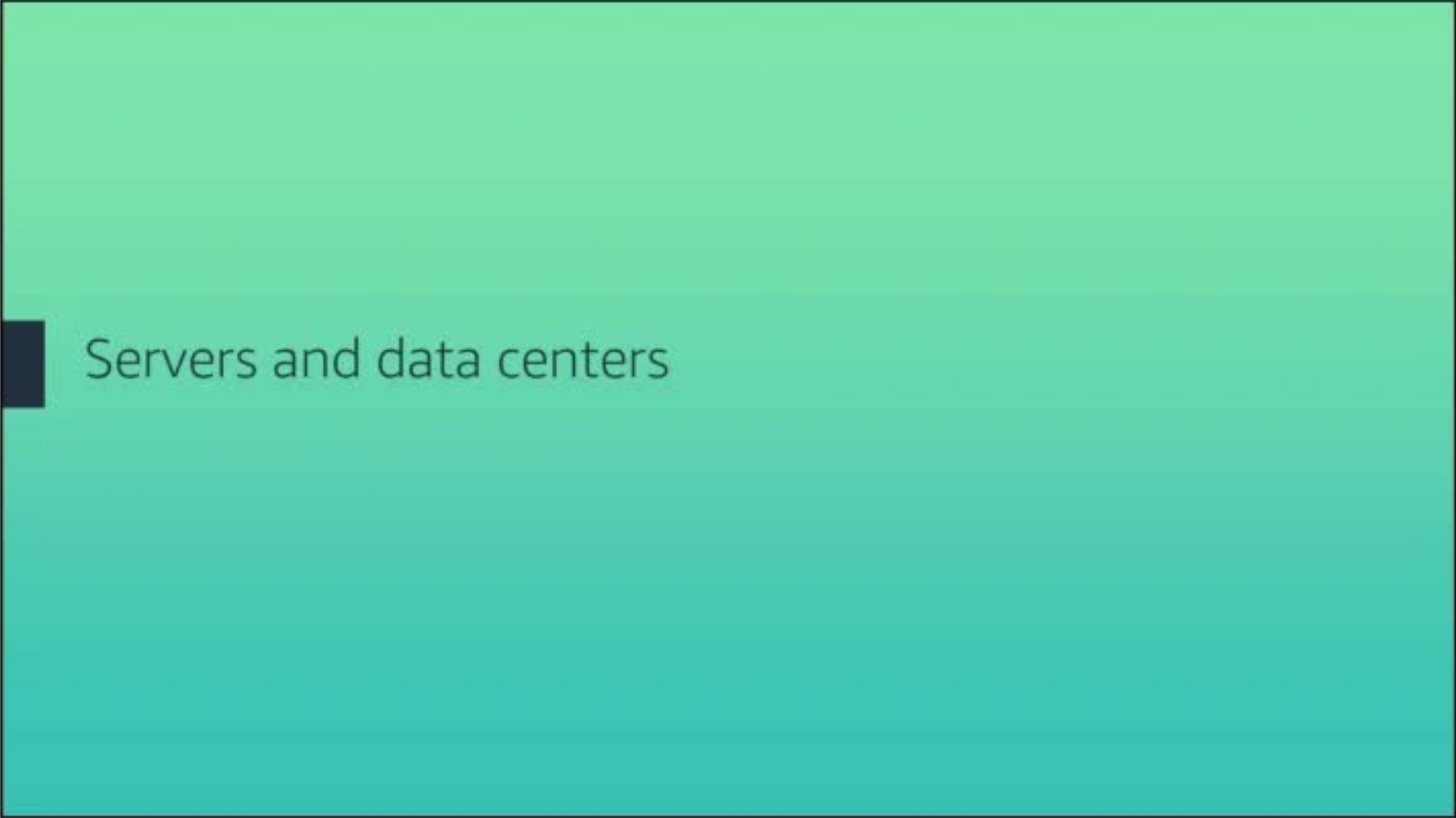
You will learn how to:

- Describe servers and data centers
- Discuss computing technology that enables cloud computing
- Describe how software is developed



After completing this module, you should be able to:

- Describe servers and data centers
- Discuss computing technology that enables cloud computing
- Describe how software is developed



Servers and data centers

What is a server?

A computer that provides data or services to other computers

- A server provides a response to a request from a client computer over a network
- Server hardware typically differs from desktop hardware to support –
 - More memory and multiple CPUs
 - Redundant power supplies and network interfaces
 - Smaller form factor
- Examples of servers –
 - Web server
 - Database server
 - Mail server



A server is a computer that provides resources or services to other computers over a network.

The many different types of servers include:

- Web server – Is used by web applications to serve Hypertext Markup Language (HTML) pages to a requesting client
- Database server – Hosts database software that applications use to store and retrieve data
- Mail server – Is used to send and receive email from and to clients

Client/server example: Web application

Web application that runs on a web server and accesses a database server



5

aws re/start

Running a web application is an example use case for servers. A web application is typically deployed to a *web server*, which is responsible for directing client requests to it. A web application usually stores its application data on a database server. The database server runs a special type of software that's called a *database management system (DBMS)*. The DBMS controls the organization, security, and access of the data. Examples of a DBMS are MySQL, an open source relational database management system; and Oracle, a relational DBMS that Oracle Corporation owns and offers.

The following list shows the flow of information in the example:

1. The user opens a browser on a client machine and enters the address of the web application's homepage. This address is called its home *Uniform Resource Locator (URL)*, for example: <https://anycompanywebapp.com>.
2. The web server receives the client request and directs it to the appropriate web application.
3. The web application sends a request to the database server to access its application data.
4. The database server returns the requested data to the web application.
5. The web application builds the response webpage and passes it to the web

server, which returns the page to the client browser.

Where does a server reside?

Servers reside in a data center



A data center hosts all of an organization's computer and networking equipment, including:

- Servers
- Storage devices
- Network devices (routers, switches, and hubs)
- Cooling equipment
- Uninterruptable power supplies (UPS)

Servers reside in a data center. A data center is a physical location that is used to host computer systems and associated components such as networks, storage devices, and power supplies.

Data centers are designed to be secure and to provide an ideal climate for the contained equipment to operate. They must protect the equipment from many types of failures and accidents, including power losses and fire.

Who owns the data center?

Traditional *on-premises* model

- You own the data center and host it at your location
- You buy, install, configure, and manage all of the hardware and software in your own facility
- You hire the staff who are responsible for managing and maintaining the data center
- You use your own data center resources

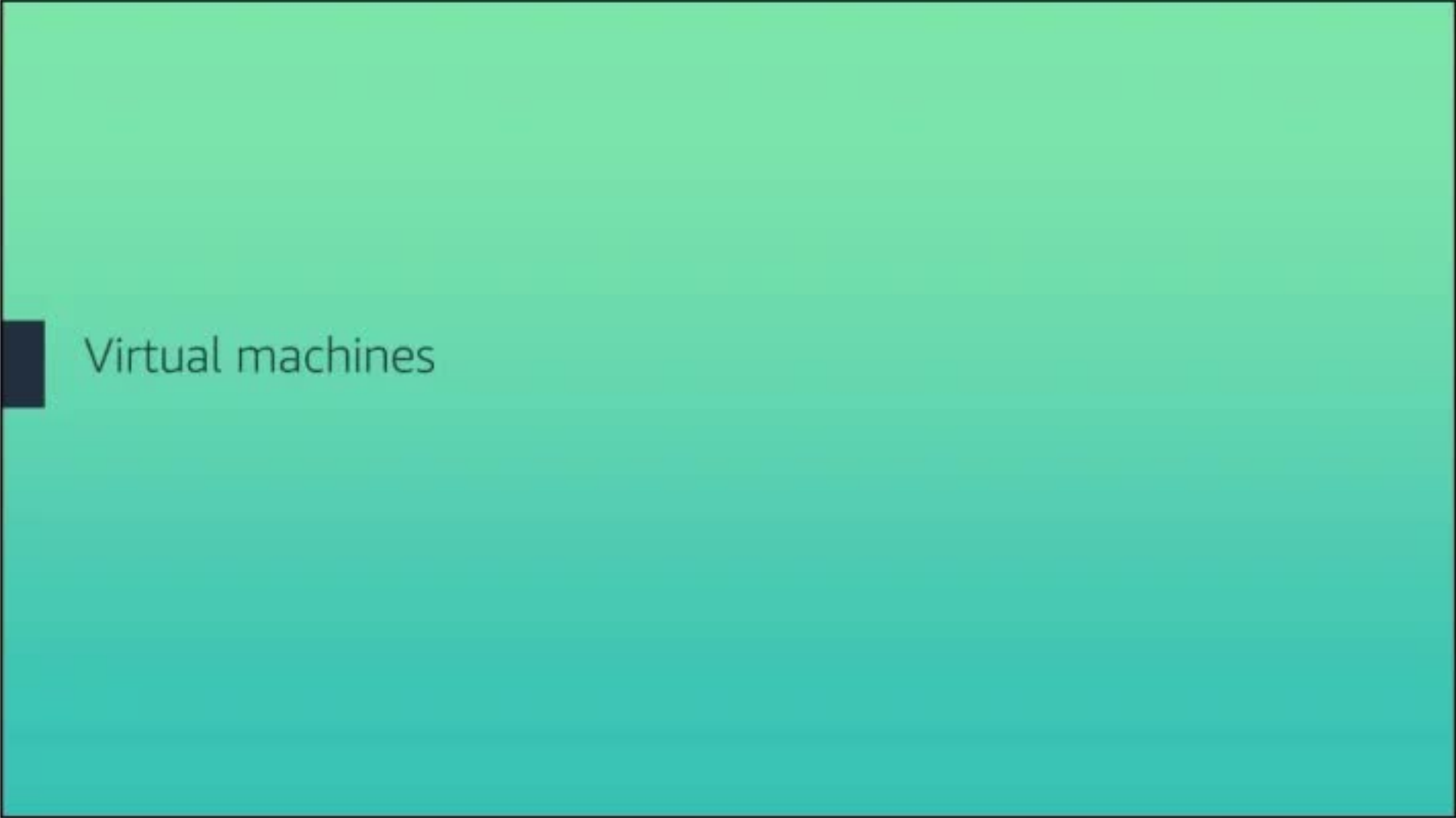
Cloud model

- A cloud services provider owns the data center
- The cloud services provider buys the hardware and infrastructure software for their own facility
- The cloud services provider hires the personnel to support the data center
- You pay to use the cloud service provider's data center resources

Traditionally, organizations owned their data centers. The equipment is *on-premises* at a location that the company owns. If you follow this model, you are the one who buys, installs, configures, and manages all the hardware and software in your own facility. You are responsible for installation, maintenance, and numerous other costs. You also must hire the staff who are responsible for the maintenance of the hardware, software, and the facility itself.

The cloud model provides another option: a cloud services provider buys the hardware and infrastructure software in their own facility. They manage and maintain it with their own personnel. You bring your application or *workload* to run on their servers and pay for the services that they offer.

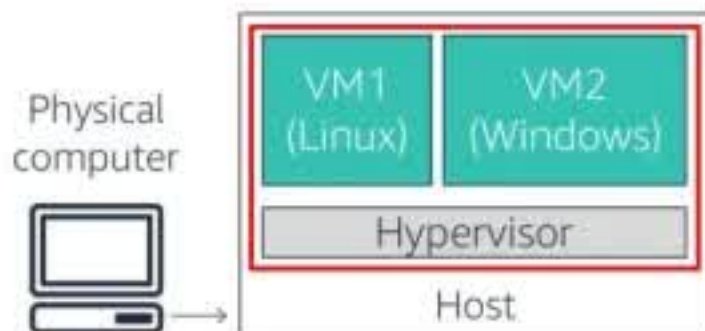
Later, you will learn the advantages of the cloud model as compared with the on-premises model.



Virtual machines

What is a virtual machine?

A virtual machine (VM) is a **software-based computer**



- A VM runs on a physical computer, which is called a **host**
- A software layer, which is called a **hypervisor**, provides access to the resources of the physical computer (CPU, memory, disk, network) to the VM
- The VM runs **its own operating system (OS)** and interacts with the host through the hypervisor
- **Multiple VMs** can be provisioned on a single host

Virtualization enables you to create multiple VMs, each with its own OS and applications, on a single physical machine.

A virtual machine (VM) is a computer that's emulated through software. It is *virtual* because it isn't a physical computer. Instead, specialized software, which is called *virtualization* software, runs inside a physical computer to provide the computing capabilities of a VM. In other words, a VM is a software-based computer that runs inside a physical computer.

The physical computer where a VM runs is called a *host*. The VM provides computing capabilities by accessing the physical resources of the host through a software layer that's called a *hypervisor*. The hypervisor shares the host's physical resources—such as its CPU, memory, disk drives, and networking capabilities—among the VMs that run on the host. A VM can run its own operating system, and multiple VMs can run on a single host. Virtualization enables you to separate your operating system and applications from the computer's hardware. Use cases for VMs include virtual desktops, multiple operating support, and cloud computing.

Benefits of VMs

Cost savings

- Running multiple VMs on a single physical machine reduces the need to buy a new computer.

Efficiency

- Running multiple VMs on a single physical computer increases its utilization.

Reusability and portability

- You can copy a VM image on the same physical host or move it to a different host to duplicate the VM's computing environment.

The benefits of using a VMs include:

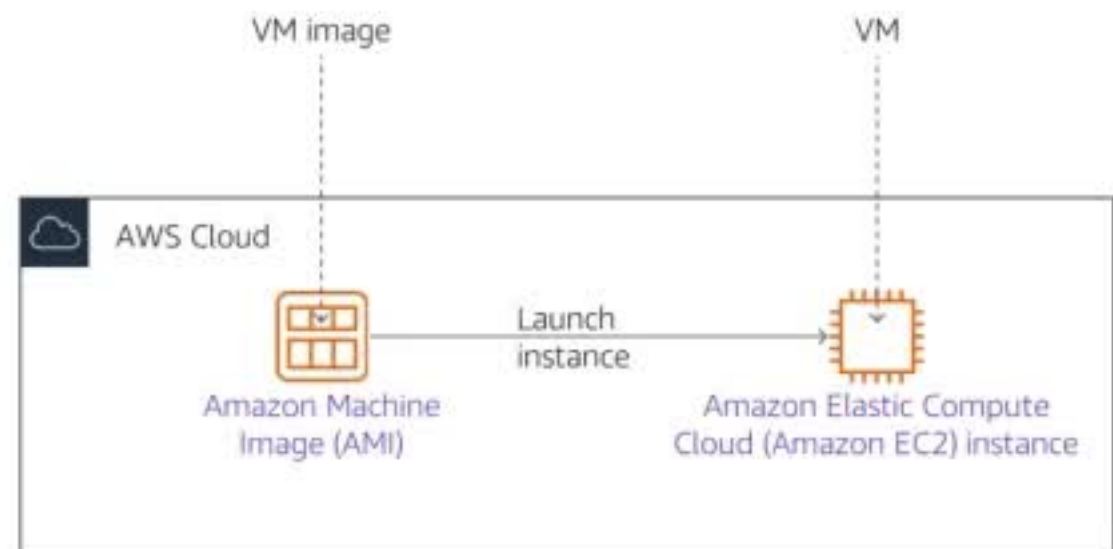
- Cost savings – For example, you don't need to buy a new machine if you want to run a different operating system (OS) on your existing machine. You create a VM with the new OS and run it on your machine with virtualization software.
- Efficiency – You can run multiple VMs on a single physical computer to handle different types of workloads and increase its utilization. VMs enable you to reduce computing resource waste due to under-utilized servers.
- Reusability and portability – A *virtual machine image* defines all of the configuration, software, and applications that are installed in a VM. You can duplicate a VM image on one or more physical hosts without creating a new VM from scratch. This duplication promotes reusability and portability. For example, creating multiple copies of the same VM to respond to incoming requests can improve your applications' performance when the number of requests increases. You can also copy a VM image to a different host for backup purposes.

VMs in the cloud

VMs are the fundamental unit of computing in the cloud


VMs enable:

- Self-service
- Pay for what you use
- Scalability



VMs enable computing in the cloud. In the AWS Cloud, the core service that offers computing capabilities is Amazon Elastic Compute Cloud (Amazon EC2).

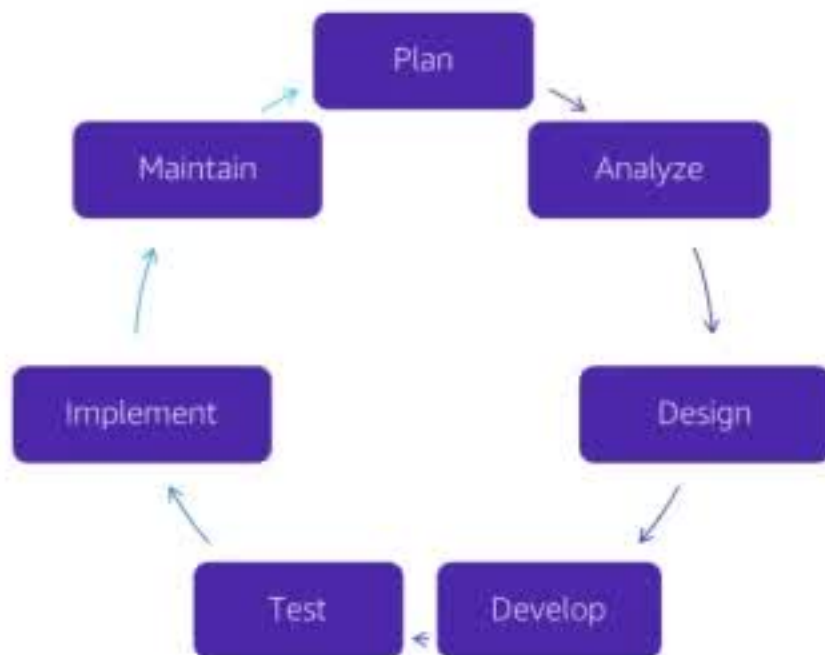
You can use Amazon EC2 to provision virtual servers, and you can completely control the computing resources of those servers. You can obtain and start new server instances in minutes. From a cost perspective, you pay only for the capacity that you use. In addition, you can quickly scale capacity both up and down as your computing requirements change.



Software Development Life Cycle

How software is developed

Software Development Life Cycle (SDLC)



- Plan: What is the problem and what resources do you need to solve it?
- Analyze: What do you want from a solution?
- Design: How will you build what you want?
- Develop: Build what you designed.
- Test: Did you get what you want?
- Implement: Start to use what you built.
- Maintain: Improve what you built.

13

aws re/start

The Software Development Life Cycle (SDLC) is a process that's used to produce software in a disciplined and organized way. When it's used correctly, it usually results in high-quality software that meets the customer's requirements.

At a high level, the purpose of each phase can be described by the following questions and actions:

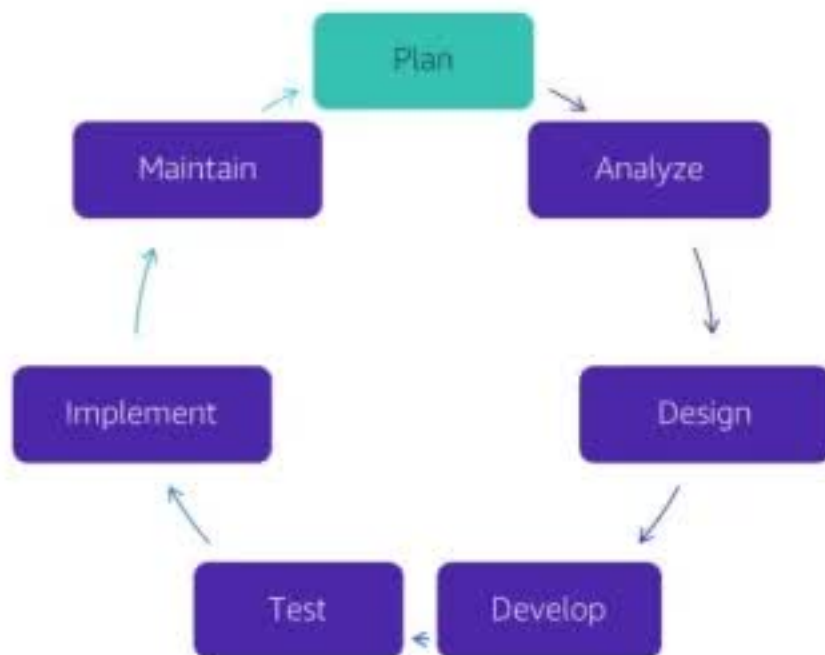
- Plan – What is the problem and what resources do you need to solve it?
- Analyze – What do you want from a solution?
- Design – How will you built what you want?
- Develop – Build what you have designed.
- Test – Did you get what you want?
- Implement – Start to use what you built.
- Maintain – Improve what you built.

The SDLC is repeated over the lifetime of an application. It's used to create, update, fix, and maintain the application.

The next slides discuss each phase in more detail.

Plan

Software Development Life Cycle (SDLC)

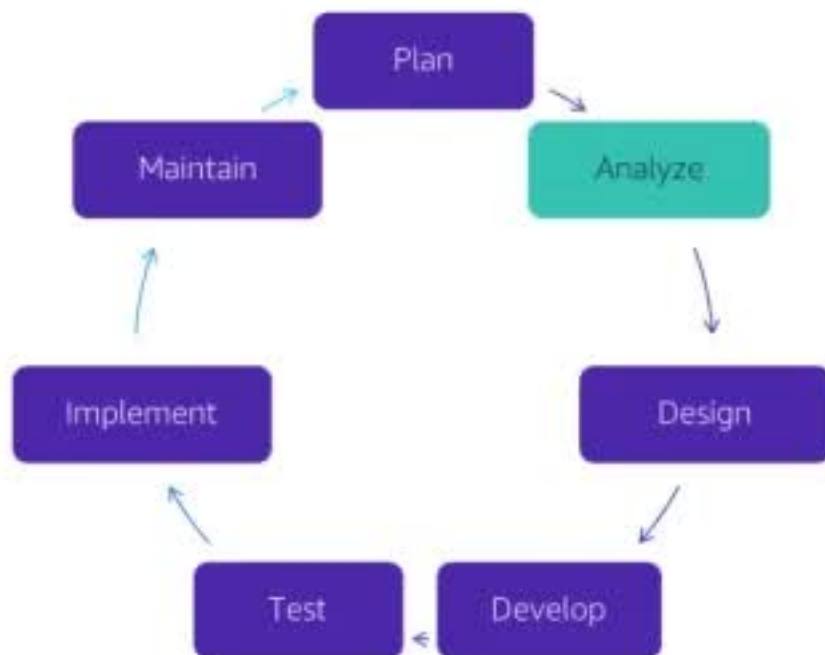


- The goals of the project are identified, along with the resources that are required to implement them. The output of this phase is a [project plan](#).
- Many factors are taken into account at this stage, including the economical, operational, and technical aspects of project implementation.
- Planning for quality assurance also happens at this stage.

In this organizational phase, a plan is formulated to identify the goals of the project and the resources that are required to implement them. Many factors, such as costs, human resources, and tools are considered and defined. The output of the Plan phase is a project plan.

Analyze

Software Development Life Cycle (SDLC)

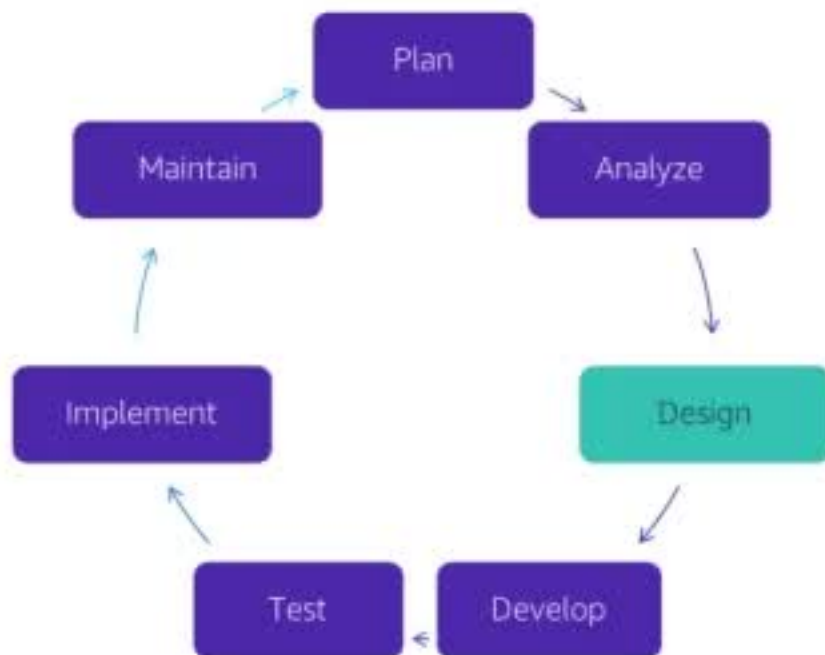


- Product requirements are clearly defined and documented in a Software Requirement Specification (SRS).
- The customer then approves the requirements.
- An SRS is used as a reference tool at every subsequent step of the SDLC.

The Analyze phase focuses on gathering the requirements for the application from the user and documenting them in a Software Requirement Specification (SRS) document.

Design

Software Development Life Cycle (SDLC)

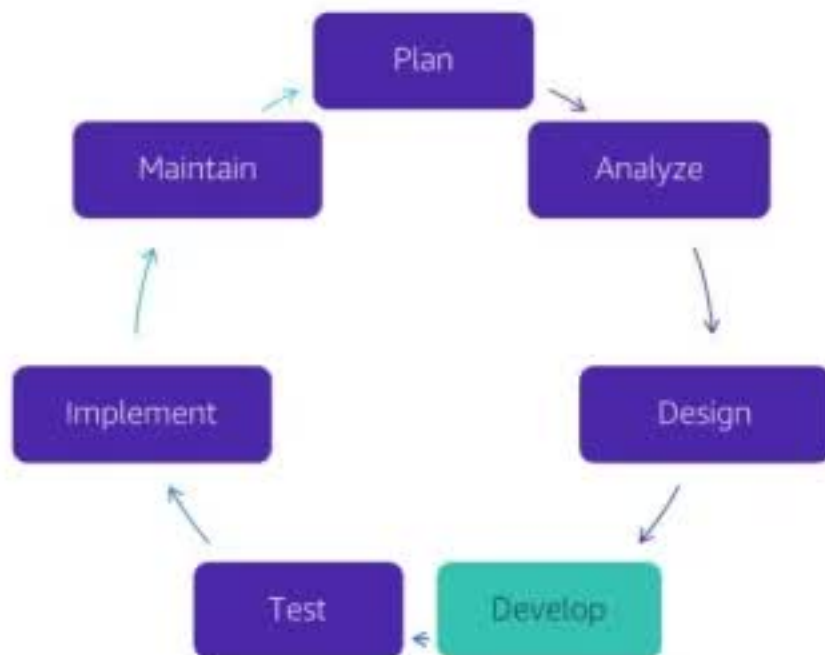


- Using the SRS, different types of architecture are evaluated for the best use in the project.
- More than one design approach is created and proposed in a design specification document.
- The design specification document contains detailed functional descriptions and other information, such as user interface descriptions.
- The design options are reviewed for risk, budget, and time constraints. The best design is then selected.

In the Design phase, user requirements are translated into a technical design. The output of the Design phase is a design specification document.

Develop

Software Development Life Cycle (SDLC)



- The actual writing of the computer code happens in this phase, and the product is built
- The code is written according to the design specification document, and the organization's software development standards and guidelines
- The programming language is chosen, and is based on the type of software that will be created

During the Develop phase, the code for the application is written according to the organization's software development standards and guidelines. The programming language that's used is chosen based on what best suits the application.

Test

Software Development Life Cycle (SDLC)



- One of the most important steps in the SDLC
- Code can be written to test other code. This process is called *automated testing*
- Common types of testing include –
 - Unit test
 - Integration test
 - Security test
 - Performance test

The purpose of the Test phase is to validate that the application components function as intended. This phase is also used to uncover and correct defects before the application is released to users.

Most defects, also called *bugs*, should be discovered and fixed during this phase. It's important to correct them in this phase because they are usually more expensive to fix when they are found in later phases.

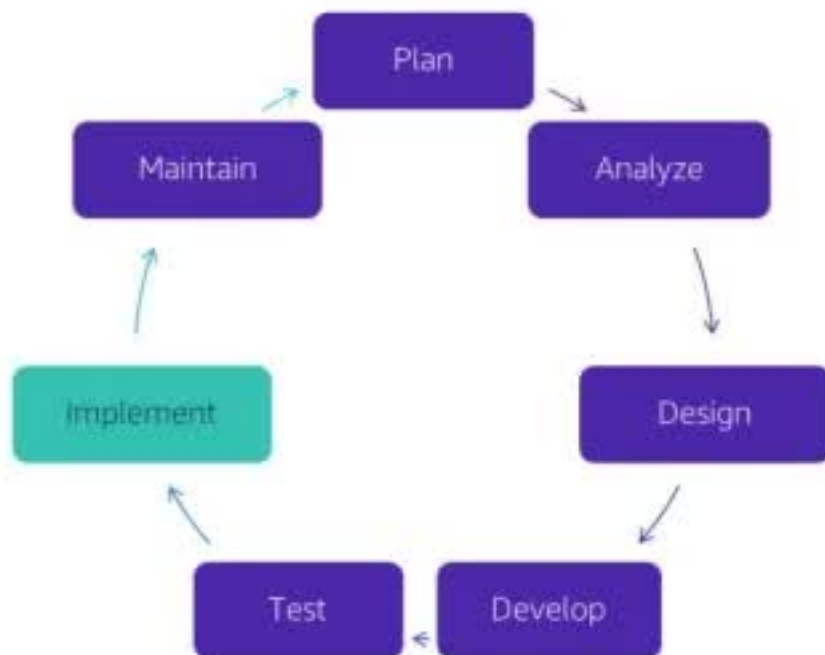
The many types of testing include:

- Unit testing – Tests individual application components at the program level. The programmer usually does this test.
- Integration testing – Tests the combination of multiple application components to verify that they work together correctly.
- Security testing – Tests to see whether the application is vulnerable from internal or external threats.

- Performance testing – Tests to see whether the application meets its expected performance requirements.

Implement

Software Development Life Cycle (SDLC)

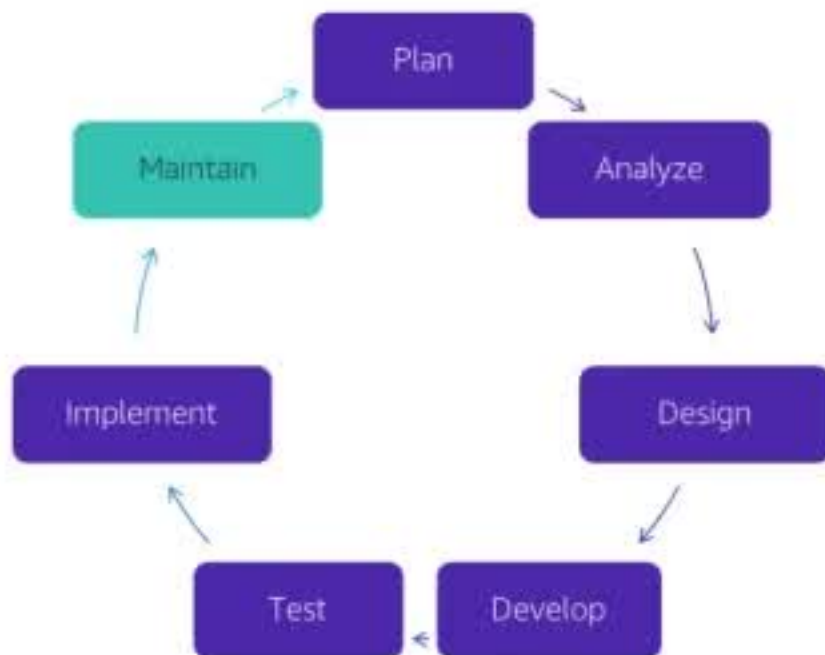


- Implementation is often called *deployment*
- The customer approves and signs off the completion of the application
- The application is released and used in production

In this phase, the finished application is deployed to a final environment, or the *production environment*, where users can start to use it.

Maintain

Software Development Life Cycle (SDLC)



- While in production, applications must be monitored constantly to ensure their correct operation.
- The need for maintenance can arise from different reasons –
 - Defect or error identified → **Corrective** maintenance
 - Changes in application environment → **Adaptive** maintenance
 - Changes in application requirements → **Perfective** maintenance
 - Prevent the occurrence of errors → **Preventive** maintenance

As soon as the application is in production, it must be monitored and maintained. The four general types of application maintenance are:

- **Corrective maintenance** – Is used to fix a problem that occurs and for which a solution is identified.
- **Adaptive maintenance** – Is required when something in the application's runtime environment is going to change. For example, an upgrade is planned for the database software that the application uses.
- **Perfective maintenance** – Occurs when new or revised functionality is identified for the application. For example, the user requests a change in the user interface.
- **Preventive maintenance** – Consists of changes that are made to avoid potential issues in the future. For example, code is redesigned or restructured for easier maintenance.

Key takeaways



© 2020 Amazon Web Services, Inc. or its affiliates. All rights reserved.

21

- A server is a computer that provides data or services to other computers.
- A data center is a physical location where an organization stores and operates its computer and networking equipment.
- Hardware virtualization enables you to create VMs on a physical computer. It's a fundamental technology that's used in cloud computing.
- The phases of the Software Development Life Cycle are –
 - Plan
 - Analyze
 - Design
 - Develop
 - Test
 - Implement
 - Maintain

aws re/start

Some key takeaways from this lesson include:

- A server is a computer that provides data or services to other computers.
- A data center is a physical location where an organization stores and operates its computer and networking equipment.
- Hardware virtualization enables you to create VMs on a physical computer. It's a fundamental technology that's used in cloud computing.
- The phases of the Software Development Life Cycle are:
 - Plan
 - Analyze
 - Design
 - Develop
 - Test
 - Implement
 - Maintain