

# Experiment no. 0

**Aim :** Study of Linux Operating system, installation and configuration of services and command line basics, basics of Computer Networks, software Development Life cycle, cloud Computing and DevOps Ecosystem.

# Theory :

* **LINUX OPERATING SYSTEM -**

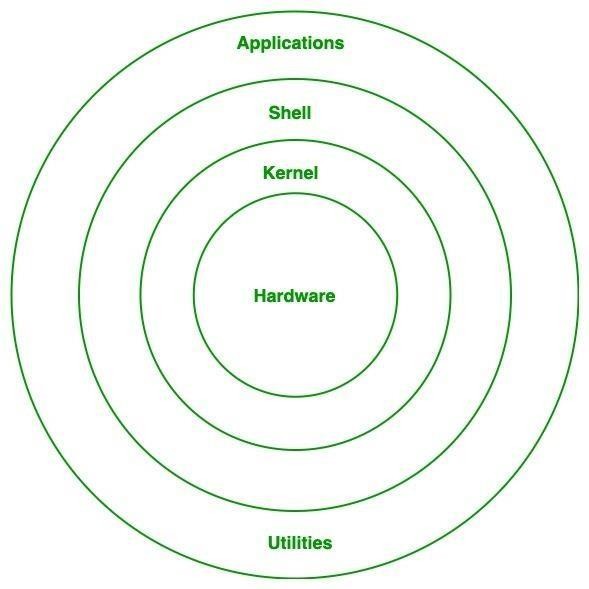
the Linux operating system is a powerful and flexible open-source software platform. It acts as the basis for a variety of devices, such embedded systems, cell phones, servers, and personal computers. Linux, that’s well-known for its reliability, safety, and flexibility, allows users to customize and improve their environment to suit specific needs. With an extensive and active community supporting it, Linux is an appealing choice for people as well as companies due to its wealth of resources and constant developments.

[Linux distribution](https://www.geeksforgeeks.org/what-are-linux-distributions/) is an operating system that is made up of a collection of software based on Linux kernel or you can say distribution contains the Linux kernel and supporting libraries and software. And you can get Linux-based operating system by downloading one of the Linux distributions and these distributions are available for different types of devices like embedded devices, personal computers, etc.

# Why use Linux?

Because it is free, open-source, and extremely flexible, Linux is widely utilized. For servers and developers, it is the ideal option because it offers strong security, stability, and performance. Generally interoperable hardware, a broad software library, and a vibrant community that offers support and regular updates are the many benefits of Linux. Due to its adaptability, users can customize the operating system according to their own needs, whether they become for personal or large enterprise use.

# Architecture of Linux



**Basic linux commands :**

1. Command : ls

Description: List files and directories in the current directory. Example: ls or ls /path/to/directory

1. Command: cd

Description: Change the current working directory. Example:

cd /path/to/directory

1. Command: pwd

Description: Print the current working directory. Example: pwd

1. Command: mkdir

Description: Create a new directory. Example:

mkdir new\_directory

1. Command: touch

Description: Create a new empty file. Example:

touch new\_file.txt

1. Command: cp

Description: Copy files or directories.

Example: cp file.txt /path/to/destination or cp -r directory /path/to/destination

1. Command: mvss

Description: Move or rename files or directories.

Example: mv file.txt /path/to/destination or mv old\_name.txt new\_name.txt

1. Command: rm

Description: Remove files or directories. Example: rm file.txt or rm -r directory

1. Command: grep

Description: Search for a specific pattern in a file or output Example: grep "keyword" file.txt

1. Command: cat

Description: Display the content of a file. Example: cat file.txt

1. Command: apt (for Debian-based systems like Ubuntu) Description: Advanced Package Tool - Manage software packages.

Example: sudo apt update (to update package lists) or sudo apt install package\_name

1. Command: yum(for Red Hat-based systems like CentOS)

Description: Yellow dog Updater, Modified - Install, update, and remove software packages.

Example: sudo yumupdate (to update packages) or sudo yum install package\_name

1. Command: useradd

Description: Create a new user account. Example: sudo useradd new\_user

1. Command: usermod

Description: Modify user account properties.

Example: sudo usermod -aG group\_name existing\_user (to add a user to a group)

1. Command: passwd

Description: Change a user's password.

Example: sudo passwd user\_name

# Installation and configuration of services and command line basics –

We will focus on Installation and Configuration of Services and Command Line Basics. These are essential skills for DevOps professionals as they are responsible for setting up and managing various services on a Linux system. Let's explore each point in more detail:

1. Experience in Installing and Configuring Software Services on a Linux System:
2. Gain hands-on experience in installing and configuring common software services on a Linux system to support different types of applications. DevOps professionals often need to install and configure software services like web servers (e.g., Apache, Nginx), databases (e.g., MySǪL, PostgreSǪL), and other applications to support the development and deployment of software applications.
3. Web Servers (e.g., Apache, Nginx):
4. Learn how to install and configure Apache and Nginx web servers to host websites and serve content over the internet. Web servers are crucial components for hosting websites and web applications. Apache and Nginx are popular web servers used in DevOps environments.
5. Databases (e.g., MySǪL, PostgreSǪL):
6. Explore how to install and configure MySǪL and PostgreSǪL databases to store and retrieve data for web applications. Databases are essential for storing and managing data in software applications. MySǪL and PostgreSǪL are widely used relational database management systems in DevOps.
7. Other Applications:
8. Familiarize yourself with the installation and configuration process of various applications commonly used in DevOps environments. Besides web servers and databases, DevOps professionals may need to install and configure other applications and tools, such as caching servers (e.g., Redis, Memcached),
9. messaging systems (e.g., RabbitMǪ, Apache Kafka), and more.
10. Understanding the Basics of the Command-Line Interface (CLI):
11. Understand the significance of the CLI in managing Linux systems, automating tasks, and executing commands for various operations. The command-line interface (CLI) is a text-based interface used to interact with the operating system and execute commands.
12. Common Commands for File Operations:
13. Master essential commands like mkdir, cp, mv, rm, and others to efficiently manage files and directories. The CLI offers powerful commands for file and directory manipulation, such as creating, copying, moving, renaming, and deleting files and directories.
14. Common Commands for Text Processing:
15. Learn how to use commands like grep to search for patterns, sed for text substitution, and awk for text manipulation. DevOps professionals often need to work with text files and process their contents. The CLI provides commands like grep, sed, and awk for text processing tasks.
16. Package Management:
17. Understand package management tools and commands to install, update, and remove software packages required for various applications. Package managers like apt (for Debian-based systems) and yum (for Red Hat-based systems) simplify the installation and management of software packages on Linux.

# COMPUTER NETWORKS -

Computer Networking is the practice of connecting computers together to enable communication and data exchange between them. In general, Computer Network is a collection of two or more computers. It helps users to communicate more easily. In this article, we are going to discuss the basics which everyone must know before going deep into Computer Networking.

# Basic Terminologies of Computer Networks

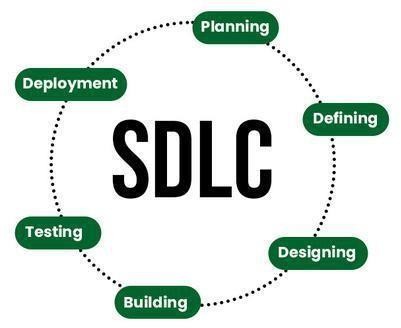
* + **Network:** A network is a collection of computers and devices that are connected together to enable communication and data exchange.
  + **Nodes:** Nodes are devices that are connected to a network. These can include computers, Servers, Printers, [Routers,](https://www.geeksforgeeks.org/introduction-of-a-router/) [Switches,](https://www.geeksforgeeks.org/types-of-switches-in-computer-network/) and other devices.
  + **Protocol:** A protocol is a set of rules and standards that govern how data is transmitted over a network. Examples of protocols include [TCP/IP,](https://www.geeksforgeeks.org/tcp-ip-model/) [HTTP,](https://www.geeksforgeeks.org/http-full-form/) and [FTP.](https://www.geeksforgeeks.org/file-transfer-protocol-ftp-in-application-layer/)
  + **Topology:** Network topology refers to the physical and logical arrangement of nodes on a network. The common network topologies include bus, star, ring, mesh, and tree.
  + **Service Provider Networks:** These types of Networks give permission to take Network Capacity and Functionality on lease from the Provider. Service Provider Networks include Wireless Communications, Data Carriers, etc.
  + **IP Address**: An IP address is a unique numerical identifier that is assigned to every device on a network. IP addresses a reused to identify devices and enable communication between them.
  + **DNS:** The [Domain Name System (DNS)](https://www.geeksforgeeks.org/domain-name-system-dns-in-application-layer/) is a protocol that is used to translate human-readable domain names (such as www.google.com) into IP addresses that computers can understand.
  + **Firewall:** A [firewall](https://www.geeksforgeeks.org/introduction-of-firewall-in-computer-network/) is a security device that is used to monitor and control incoming and outgoing network traffic. Firewalls are used to protect networks from unauthorized access and other security threats.

# Types of Enterprise Computer Networks

* + **LAN:** A [Local Area Network (LAN) i](https://www.geeksforgeeks.org/types-of-area-networks-lan-man-and-wan/)s a network that covers a small area, such as an office or a home. LANs are typically used to connect computers and other devices within a building or a campus.
  + **WAN:** A [Wide Area Network (WAN)](https://www.geeksforgeeks.org/wan-full-form/) is a network that covers a large geographic area, such as a city, country, or even the entire world. WANs are used to connect LANs together and are typically used for long- distance communication.
  + **Cloud Networks:** [Cloud Networks](https://www.geeksforgeeks.org/cloud-networking/) can be visualized with a Wide Area Network (WAN) as they can be hosted on public or private cloud service providers and cloud networks are available if there is a demand. Cloud Networks consist of Virtual Routers, Firewalls, etc.

# SOFTWARE DEVELOPMENT LIFE CYCLE (SDLC)

**Software development life cycle (SDLC) is a structured process that is used to design, develop, and test good- quality software.** SDLC, or software development life cycle, is a methodology that defines the entire procedure of softwaredevelopment step-by-step.



The [**SDLC model**](https://www.geeksforgeeks.org/sdlc-models-types-phases-use) **involves six phases or stages** while developing any software. SDLC is a collection of these six stages, and the stages of SDLC areas follows:

# Stage-1: Planning and Requirement Analysis

Planning is a crucial step in everything, just as in [software development](https://www.geeksforgeeks.org/software-development). In this same stage, [requirement](https://www.geeksforgeeks.org/activities-involved-in-software-requirement-analysis) [analysis](https://www.geeksforgeeks.org/activities-involved-in-software-requirement-analysis) is also performed by the developers of the organization. This is attained from customer inputs, and sales department/market surveys. The information from this analysis forms the building blocks of a basic project. The quality of the project is a result of planning.

# Stage-2: Defining Requirements

In this stage, all the requirements for the target software are specified. These requirements get approval from customers, market analysts, and stakeholders. This is fulfilled by utilizing SRS (Software Requirement Specification). This is a sort of document that specifies all those things that need to be defined and created during the entire project cycle.

# Stage-3: Designing Architecture

[SRS](https://www.geeksforgeeks.org/software-requirement-specification-srs-format) is a reference for software designers to come up with the best architecture for the software. Hence, with the requirements defined in SRS, multiple designs for the product architecture are present in the Design Document Specification (DDS).

This DDS is assessed by market analysts and stakeholders. After evaluating all the possible factors, the most practical and logical design is chosen for development.

# Stage-4: Developing Product

At this stage, the fundamental development of the product starts. For this, developers use a specific programming code as per the design in the DDS. Hence, it is important for the coders to follow the protocols

set by the association. Conventional programming tools like compilers, interpreters, debuggers, etc. are also put into use at this stage. Some popular languages like C/C++, Python, Java, etc. are put into use as per the software regulations.

# Stage-5: Product Testing and Integration

After the development of the product, testing of the software is necessary to ensure its smooth execution. Although, minimal testing is conducted at every stage of SDLC. Therefore, at this stage, all the probable flaws are tracked, fixed, and retested. This ensures that the product confronts the quality requirements of SRS.

# Stage-6: Deployment and Maintenance of Products

After detailed testing, the conclusive product is released in phases as per the organization’s strategy. Then it is tested in a real industrial environment. It is important to ensure its smooth performance. If it performs well, the organization sends out the product as a whole. After retrieving beneficial feedback, the company releases it as it is or with auxiliary improvements to make it further helpful for the customers. However, this alone is not enough. Therefore, along with the deployment, the [product’s supervision](https://www.geeksforgeeks.org/product-management).

# CLOUD COMPUTING –

Cloud computing is adopted by every company, whether it is an MNC or a startup many are still migrating towards it because of the cost-cutting, lesser maintenance, and the increased capacity of the data with the help of servers maintained by the cloud providers.

One more reason for this drastic change from the On-premises servers of the companies to the Cloud providers is the **‘Pay as you go**’ principle-based services provided by them i.e., you only have to pay for the service which you are using. The disadvantage On-premises server holds is that if the server is not in use the company still has to pay for it.

Cloud Computing?

Cloud Computing means storing and accessing the data and programs on remote servers that are hosted on the internet instead of the computer’s hard drive or local server. Cloud computing is also referred toas Internet-based computing, it is a technology where the resource is provided as a service through the Internet to the user. The data that is stored can be files, images, documents, or any other storable document.

The following are some of the Operations that can be performed with Cloud Computing

‐ Storage, backup, and recovery of data

‐ Delivery of software on demand

‐ Development of new applications and services

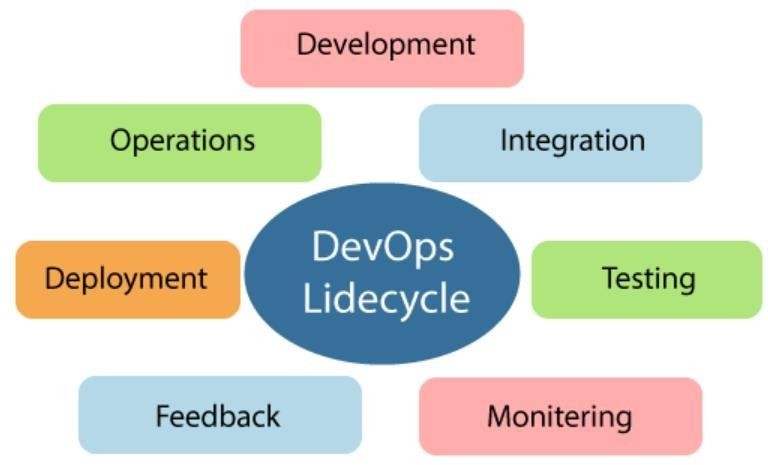
‐ Streaming videos and audio

# Understanding How Cloud Computing Works?

Cloud computing helps users in easily accessing computing resources like storage, and processing over internet rather than local hardwares. Here we discussing how it works in nutshell:

* + **Infrastructure:** Cloud computing depends on remote network servers hosted on internet for store, manage, and process the data.
  + **On-Demand Acess:** Users can access cloud services and resources based on-demand they can scale up or down the without having to invest for physical hardware.
  + **Types of Services:** Cloud computing offers various benefits such as cost saving, scalability, reliability and accessibility it reduces capital expenditures, improves efficiency.

# DEVOPS LIFECYCLE -



DevOps defines an agile relationship between operations and Development. It is a process that is practiced by the development team and operational engineers together from beginning to the final stage of the product.

# DevOps Lifecycle

DevOps is a set of practices that combine software development (Dev) and IT operations (Ops) to shorten the software development lifecycle while delivering features, fixes, and updates frequently in close alignment with business objectives.

# DevOps Tools

A wide range of tools support the DevOps lifecycle:

‐ **Version control:** Git, SVN

‐ **Continuous integration/continuous delivery (CI/CD):** Jenkins, GitLab CI/CD, CircleCI

‐ **Configuration management:** Ansible, Puppet, Chef

‐ **Containerization:** Docker, Kubernetes

‐ **Infrastructure as code:** Terraform

‐ **Monitoring and logging:** Prometheus, Grafana, ELK Stack

‐ **Cloud platforms:** AWS, Azure, GCP

# Benefits of DevOps

‐ Faster time-to-market

‐ Improved collaboration between teams

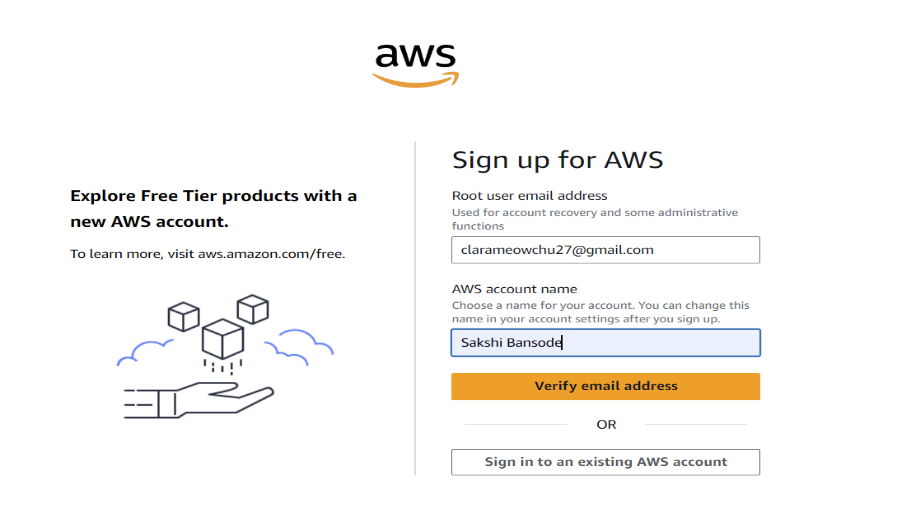
‐ Increased deployment frequency

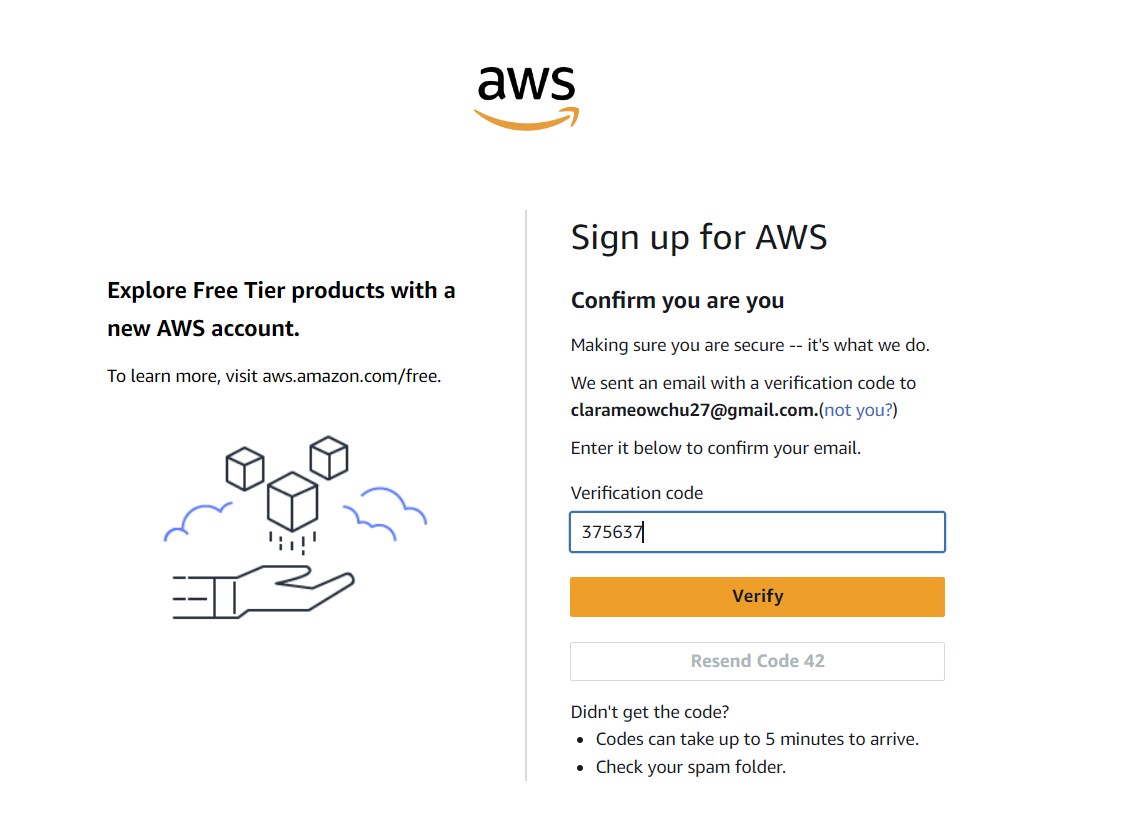
‐ Higher reliability and stability

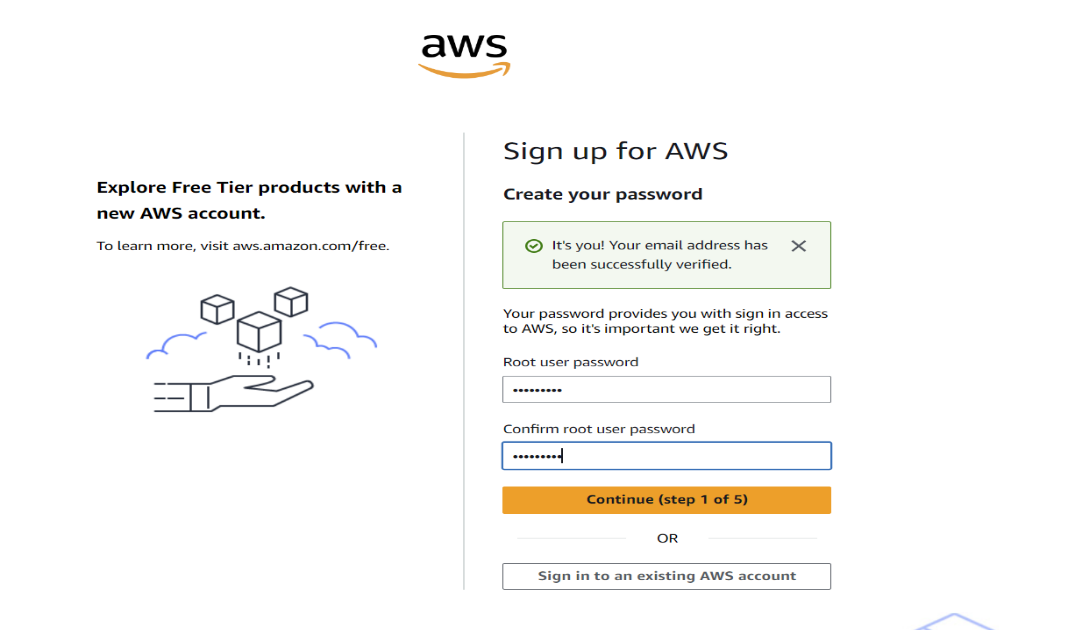
‐ Faster problem resolution

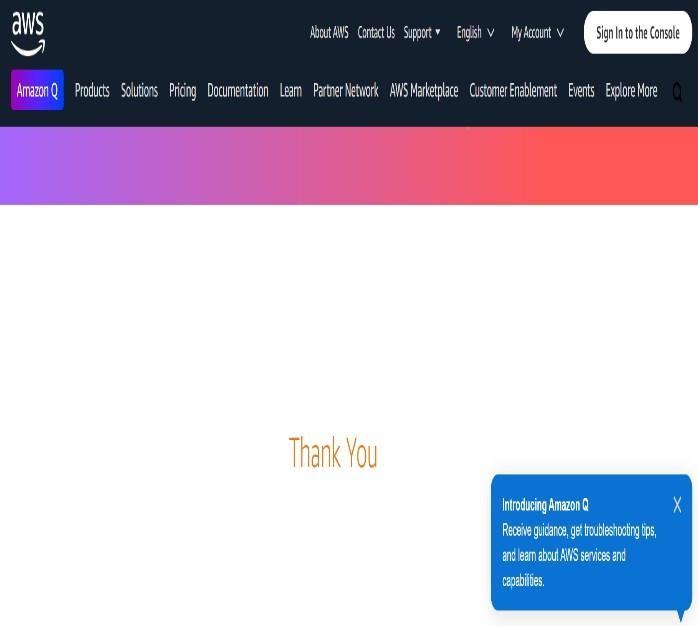
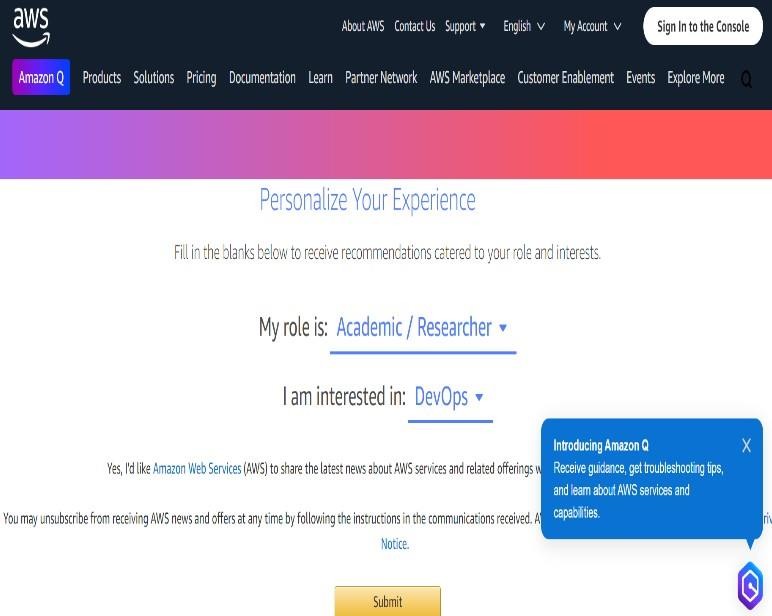
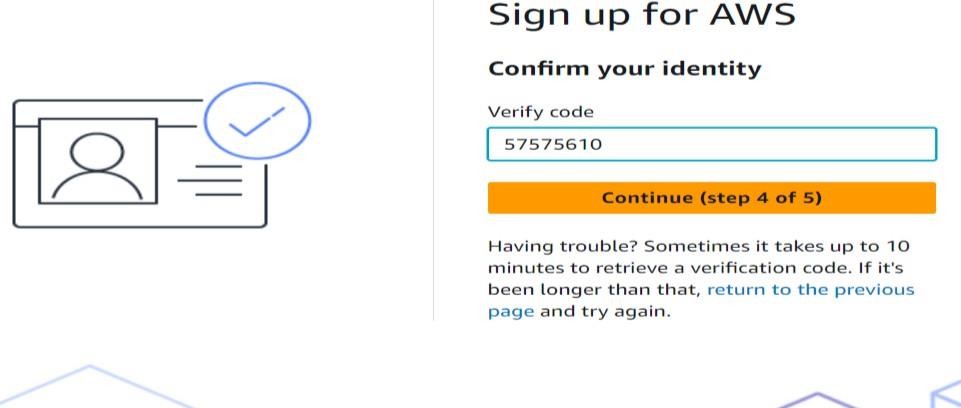
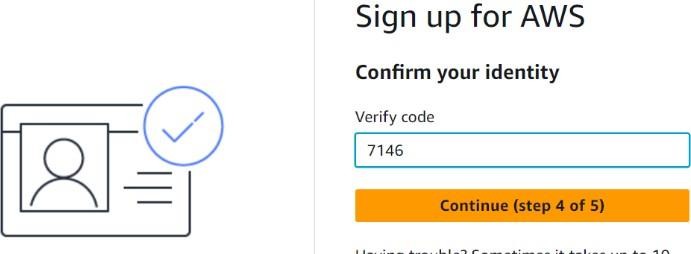
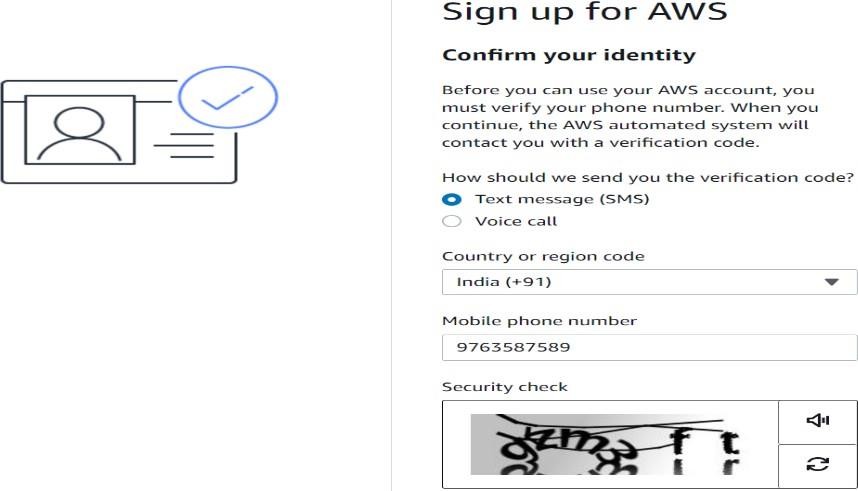
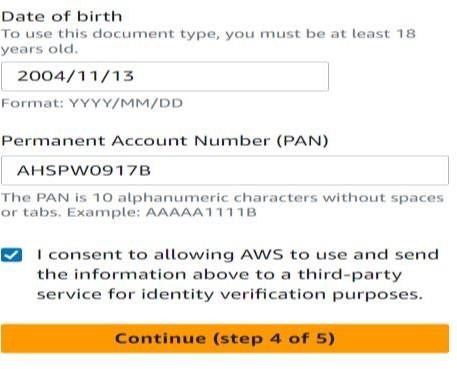
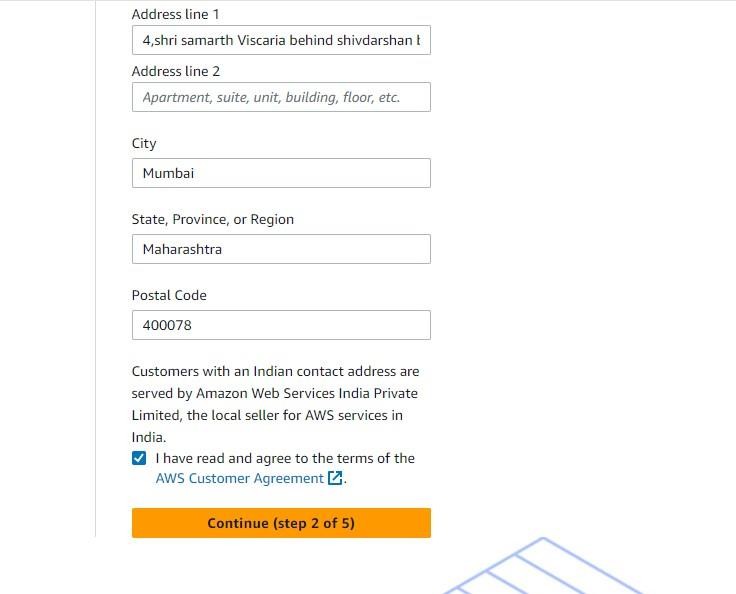
* Steps:

Sign up









* **Conclusion :** Hence we studied linux operating System , SDLC , computer networks, cloud computing Devops ecosystem.