Thadomal Shahani Engineering College

Bandra (W.), Mumbai- 400 050.

CERTIFICATE

Certify that Mr./Miss Aarnav Malvia

of Computer Department, Semester VI with

Roll No. 2103109 has completed a course of the necessary

experiments in the subject Cloud Computing under my

supervision in the **Thadomal Shahani Engineering College**

Laboratory in the year 2023 - 2024

Teacher In-Charge

Head of the Department

Date 10/4/24.

Principal

THADOMAL SHAHANI TSEC ENGINEERING COLLEGE

CONTENTS

THADOMAL SHAHANI TSEC ENGINEERING COLLEGE

Experiment : 1

Aim :- Introduction and overview of cloud computing

Q1) Definition of cloud computing.

Ans1) Cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to shared pool of configurable computing computing ~~or~~ resources that can be rapidly provisioned and released with minimal management effort or service provider interaction. This cloud model is composed of five essential characteristics, three service models and four deployment models. Users can obtain technology services such as processing power, storage and databases from a cloud provider, eliminating the need for purchasing, operating and maintaining on premises physical data centres and servers.

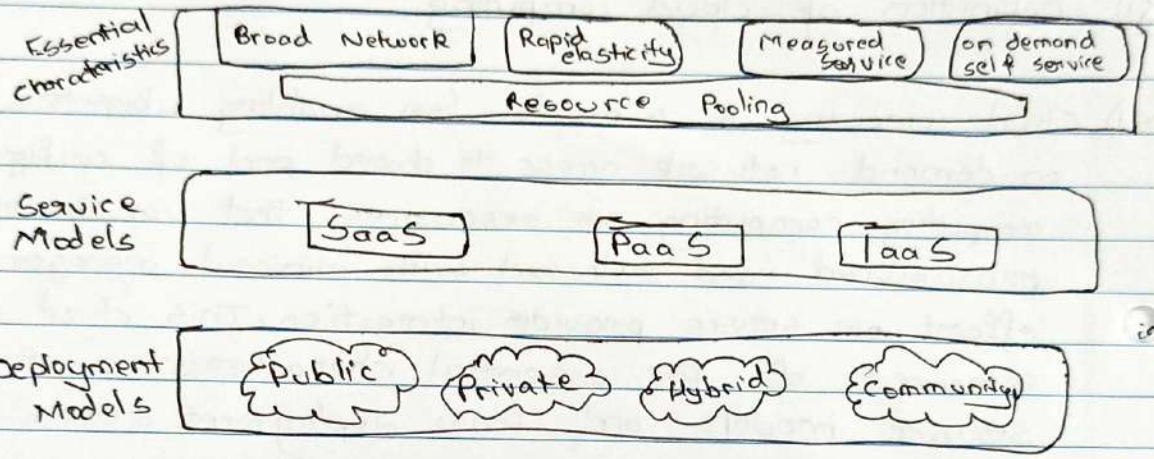
Q2) Characteristics of cloud computing

Ans2) 1) On demand self service :- Cloud computing services does not require any human administrators.
2) Broad network access :- Computing services are generally provided standard networks and heterogeneous devices.
3) Rapid elasticity
4) Resource pooling
5) virtualisation
6) Measured service
7) Security
8) Automation.

THADOMAL SHAHANI TSEC ENGINEERING COLLEGE

Q3) NIST cloud computing model.

Ans 3)



The NIST defines cloud computing as a model for enabling convenient, on demand network access to a shared pool of configurable computing resources that can be rapidly provisioned and released with minimal management effort or service provider interaction.

Q4 Different types of cloud computing.

Ans 4 Deployment models:

1) Public cloud
2) Private cloud
3) Hybrid cloud
4) Community cloud
5) Multi cloud

Service models:

1) IaaS Infrastructure as a Service
2) PaaS Platform as a service
3) SaaS Software as a service.

THADOMAL SHAHANI TSEC ENGINEERING COLLEGE

Q5) Explaination of architecture of cloud computing.

Ans 5) Cloud architecture is divided into two parts:-

1) Frontend
2) Backend

Frontend:- This refers to the client side of cloud computing system. It contains of the UI and applications which are used by the client.

Backend:- This refers to the cloud itself which is used by the service provider. It contains the resources as well as manages the resources and provides security mechanisms. It includes huge storages, virtual applications, virtual machines, traffic control mechanisms, etc.

Q6) Benefits and limitations of cloud computing.

Ans 6) Benefits:-

1) Back-up and restore data
2) Improved collaboration
3) Excellent accessibility
4) Low maintenance cost
5) Mobility
7) Unlimited storage capacity
8) Data Security.

Limitations:-

1) Internet connectivity
2) Vendor lock-in
3) Limited Control
4) Security

A.
6/24/24

Name : Arnav Malvia

Roll No.: 2103109

Batch : C23

Experiment 1: Introduction and overview of cloud computing

Q1) Definition of cloud computing

Ans 1) Cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction. This cloud model is composed of five essential characteristics, three service models, and four deployment models.

Cloud computing is a general term for the delivery of hosted computing services and IT resources over the internet with pay-as-you-go pricing. Users can obtain technology services such as processing power, storage and databases from a cloud provider, eliminating the need for purchasing, operating and maintaining on-premises physical data centers and servers.

A cloud can be private, public or a hybrid. A public cloud sells services to anyone on the internet. A private cloud is a proprietary network or a data center that supplies hosted services to a limited number of people, with certain access and permissions settings. A hybrid cloud offers a mixed computing environment where data and resources can be shared between both public and private clouds. Regardless of the type, the goal of cloud computing is to provide easy, scalable access to computing resources and IT services.

Cloud infrastructure involves the hardware and software components required for the proper deployment of a cloud computing model. Cloud computing can also be thought of as utility computing or on-demand computing.

Q2) Characteristics of cloud computing

Ans 2) These are the characteristics of cloud computing:-

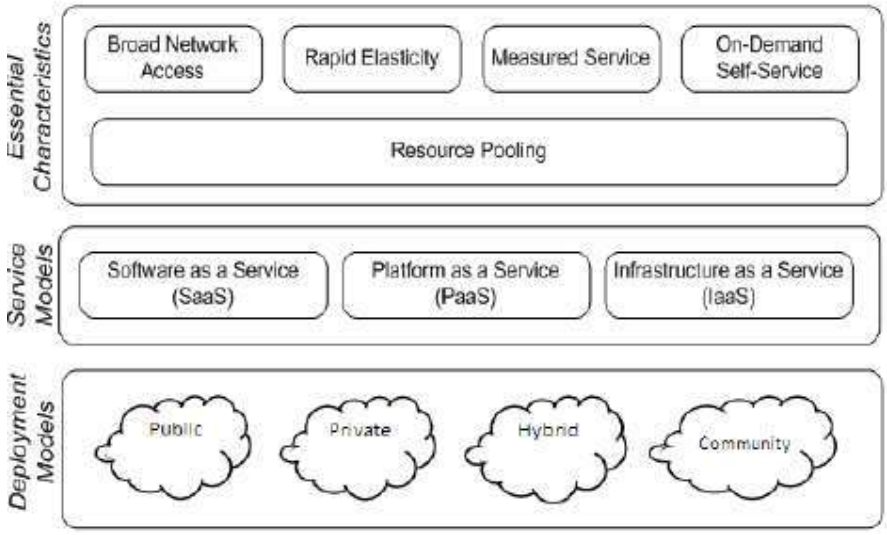
1. **On-demand self-services:** The Cloud computing services does not require any human administrators, user themselves are able to provision, monitor and manage computing resources as needed.
2. **Broad network access:** The Computing services are generally provided over standard networks and heterogeneous devices.
3. **Rapid elasticity:** The Computing services should have IT resources that are able to scale out and in quickly and on as needed basis. Whenever the user

require services it is provided to him and it is scale out as soon as its requirement gets over.

4. **Resource pooling:** The IT resource (e.g., networks, servers, storage, applications, and services) present are shared across multiple applications and occupant in an uncommitted manner. Multiple clients are provided service from a same physical resource.
5. **Measured service:** The resource utilization is tracked for each application and occupant, it will provide both the user and the resource provider with an account of what has been used. This is done for various reasons like monitoring billing and effective use of resource.
6. **Multi-tenancy:** Cloud computing providers can support multiple tenants (users or organizations) on a single set of shared resources.
7. **Virtualization:** Cloud computing providers use virtualization technology to abstract underlying hardware resources and present them as logical resources to users.
8. **Resilient computing:** Cloud computing services are typically designed with redundancy and fault tolerance in mind, which ensures high availability and reliability.
9. **Flexible pricing models:** Cloud providers offer a variety of pricing models, including pay-per-use, subscription-based, and spot pricing, allowing users to choose the option that best suits their needs.
10. **Security:** Cloud providers invest heavily in security measures to protect their users' data and ensure the privacy of sensitive information.
11. **Automation:** Cloud computing services are often highly automated, allowing users to deploy and manage resources with minimal manual intervention.
12. **Sustainability:** Cloud providers are increasingly focused on sustainable practices, such as energy-efficient data centers and the use of renewable energy sources, to reduce their environmental impact.

Q3) NIST cloud computing Model

Ans 3)



The National Institute of Standards and Technology (NIST) defines cloud computing as a "model for enabling convenient, on-demand network access to a shared pool of configurable computing resources that can be rapidly provisioned and released with minimal management effort or service provider interaction." Cloud is no longer an outlier but an essential part of business transformation strategies. Still, many organizations struggle to overcome the barriers of cloud computing to fully adopt it into their infrastructure.

Hybrid cloud computing offers a means to tackle these obstacles and create an adaptable, efficient business operation. A deployment model of cloud computing, hybrid cloud combines an on-premises datacenter with a public cloud and allows data and applications to be shared between them. This form of cloud computing relies on adaptability, resilience, and flexibility to meet the needs of an organization and their workloads. With a Modern Operations strategy at its foundation, the disciplines of hybrid cloud can help to mitigate the challenges of cloud computing so that businesses can be future-ready to evolve and thrive in any situation.

Q4) Different models of cloud computing (Service and Deployment)

Ans 4) Cloud Deployment Model functions as a virtual computing environment with a deployment architecture that varies depending on the amount of data you want to store and who has access to the infrastructure.

Types of Cloud Computing Deployment Models

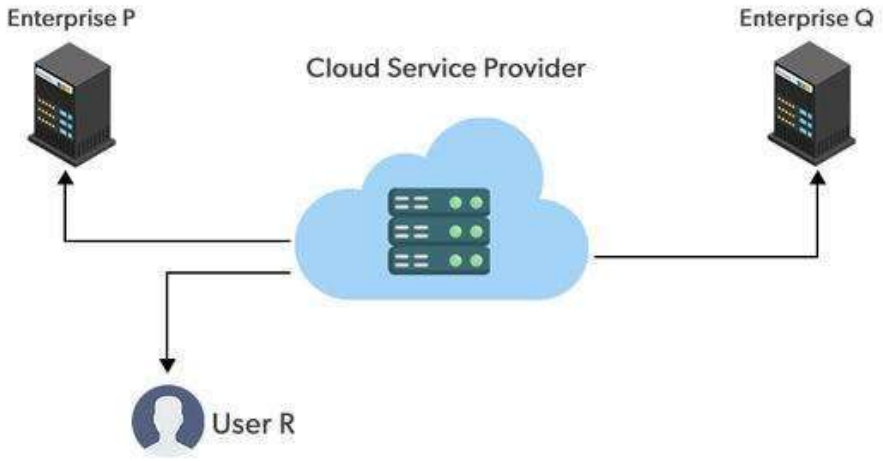
The cloud deployment model identifies the specific type of cloud environment based on ownership, scale, and access, as well as the cloud's nature and purpose. The location of the servers you're utilizing and who controls them are defined by a cloud deployment model. It specifies how your cloud infrastructure will look, what you can change, and whether you will be given services or will have to create everything yourself. Relationships between the infrastructure and your users are also defined by cloud deployment types. Different types of cloud computing deployment models are described below.

- Public Cloud

- Private Cloud
- Hybrid Cloud
- Community Cloud
- Multi-Cloud

Public Cloud

The public cloud makes it possible for anybody to access systems and services. The public cloud may be less secure as it is open to everyone. The public cloud is one in which cloud infrastructure services are provided over the internet to the general people or major industry groups. The infrastructure in this cloud model is owned by the entity that delivers the cloud services, not by the consumer. It is a type of cloud hosting that allows customers and users to easily access systems and services. This form of cloud computing is an excellent example of cloud hosting, in which service providers supply services to a variety of customers. In this arrangement, storage backup and retrieval services are given for free, as a subscription, or on a per-user basis. For example, Google App Engine etc.



Public Cloud

Advantages of the Public Cloud Model

- **Minimal Investment:** Because it is a pay-per-use service, there is no substantial upfront fee, making it excellent for enterprises that require immediate access to resources.
- **No setup cost:** The entire infrastructure is fully subsidized by the cloud service providers, thus there is no need to set up any hardware.
- **Infrastructure Management is not required:** Using the public cloud does not necessitate infrastructure management.
- **No maintenance:** The maintenance work is done by the service provider (not users).
- **Dynamic Scalability:** To fulfill your company's needs, on-demand resources are accessible.

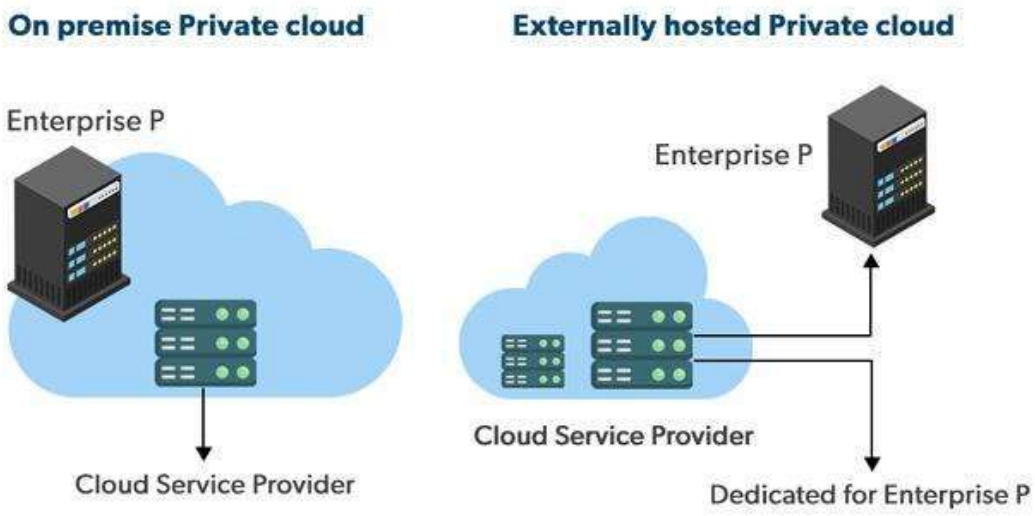
Disadvantages of the Public Cloud Model

- **Less secure:** Public cloud is less secure as resources are public so there is no guarantee of high-level security.

- **Low customization:** It is accessed by many public so it can't be customized according to personal requirements.

Private Cloud

The private cloud deployment model is the exact opposite of the public cloud deployment model. It's a one-on-one environment for a single user (customer). There is no need to share your hardware with anyone else. The distinction between private and public clouds is in how you handle all of the hardware. It is also called the "internal cloud" & it refers to the ability to access systems and services within a given border or organization. The cloud platform is implemented in a cloud-based secure environment that is protected by powerful firewalls and under the supervision of an organization's IT department. The private cloud gives greater flexibility of control over cloud resources.



Private Cloud

Advantages of the Private Cloud Model

- **Better Control:** You are the sole owner of the property. You gain complete command over service integration, IT operations, policies, and user behavior.
- **Data Security and Privacy:** It's suitable for storing corporate information to which only authorized staff have access. By segmenting resources within the same infrastructure, improved access and security can be achieved.
- **Supports Legacy Systems:** This approach is designed to work with legacy systems that are unable to access the public cloud.
- **Customization:** Unlike a public cloud deployment, a private cloud allows a company to tailor its solution to meet its specific needs.

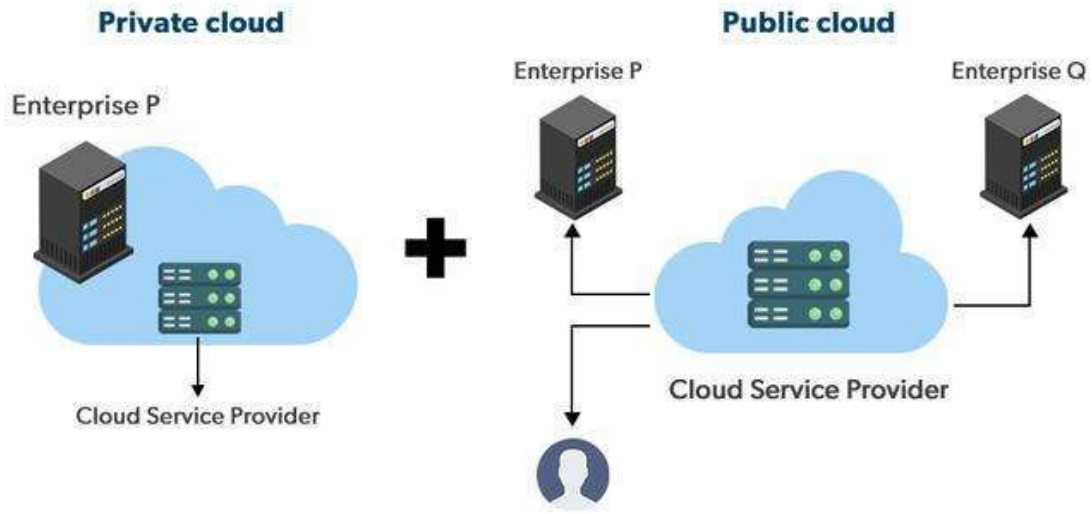
Disadvantages of the Private Cloud Model

- **Less scalable:** Private clouds are scaled within a certain range as there is less number of clients.
- **Costly:** Private clouds are more costly as they provide personalized facilities.

Hybrid Cloud

By bridging the public and private worlds with a layer of proprietary software, hybrid cloud computing gives the best of both worlds. With a hybrid solution, you may host the app in a safe environment while taking advantage of the public cloud's cost savings. Organizations

can move data and applications between different clouds using a combination of two or more cloud deployment methods, depending on their needs.



Hybrid Cloud

Advantages of the Hybrid Cloud Model

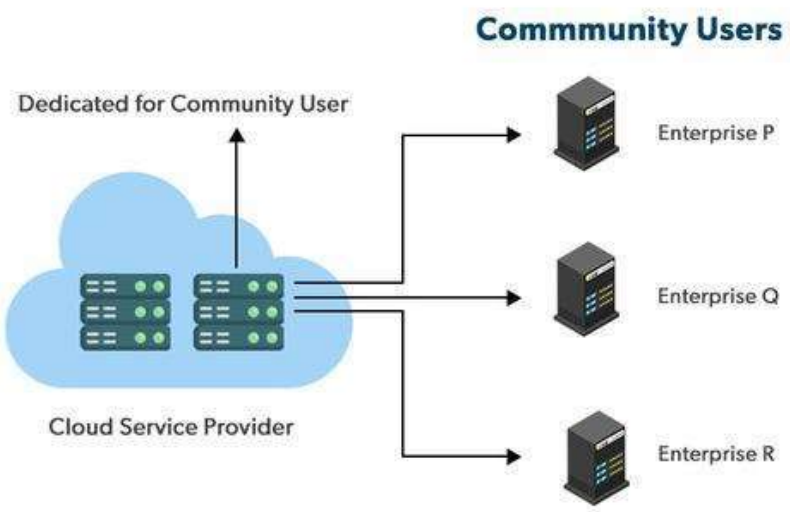
- **Flexibility and control:** Businesses with more flexibility can design personalized solutions that meet their particular needs.
- **Cost:** Because public clouds provide scalability, you'll only be responsible for paying for the extra capacity if you require it.
- **Security:** Because data is properly separated, the chances of data theft by attackers are considerably reduced.

Disadvantages of the Hybrid Cloud Model

- **Difficult to manage:** Hybrid clouds are difficult to manage as it is a combination of both public and private cloud. So, it is complex.
- **Slow data transmission:** Data transmission in the hybrid cloud takes place through the public cloud so latency occurs.

Community Cloud

It allows systems and services to be accessible by a group of organizations. It is a distributed system that is created by integrating the services of different clouds to address the specific needs of a community, industry, or business. The infrastructure of the community could be shared between the organization which has shared concerns or tasks. It is generally managed by a third party or by the combination of one or more organizations in the community.



Community Cloud

Advantages of the Community Cloud Model

- **Cost Effective:** It is cost-effective because the cloud is shared by multiple organizations or communities.
- **Security:** Community cloud provides better security.
- **Shared resources:** It allows you to share resources, infrastructure, etc. with multiple organizations.
- **Collaboration and data sharing:** It is suitable for both collaboration and data sharing.

Disadvantages of the Community Cloud Model

- **Limited Scalability:** Community cloud is relatively less scalable as many organizations share the same resources according to their collaborative interests.
- **Rigid in customization:** As the data and resources are shared among different organizations according to their mutual interests if an organization wants some changes according to their needs they cannot do so because it will have an impact on other organizations.

Multi-Cloud

We're talking about employing multiple cloud providers at the same time under this paradigm, as the name implies. It's similar to the hybrid cloud deployment approach, which combines public and private cloud resources. Instead of merging private and public clouds, multi-cloud uses many public clouds. Although public cloud providers provide numerous tools to improve the reliability of their services, mishaps still occur. It's quite rare that two distinct clouds would have an incident at the same moment. As a result, multi-cloud deployment improves the high availability of your services even more.

Advantages of the Multi-Cloud Model

- You can mix and match the best features of each cloud provider's services to suit the demands of your apps, workloads, and business by choosing different cloud providers.
- **Reduced Latency:** To reduce latency and improve user experience, you can choose cloud regions and zones that are close to your clients.

- **High availability of service:** It's quite rare that two distinct clouds would have an incident at the same moment. So, the multi-cloud deployment improves the high availability of your services.

Disadvantages of the Multi-Cloud Model

- **Complex:** The combination of many clouds makes the system complex and bottlenecks may occur.
- **Security issue:** Due to the complex structure, there may be loopholes to which a hacker can take advantage hence, makes the data insecure.

Cloud Computing helps in rendering several services according to roles, companies, etc. Cloud computing models are explained below.

- Infrastructure as a service (IaaS)
- Platform as a service (PaaS)
- Software as a service (SaaS)

1. Infrastructure as a service (IaaS)

Infrastructure as a Service (IaaS) helps in delivering computer infrastructure on an external basis for supporting operations. Generally, IaaS provides services to networking equipment, devices, databases, and web servers.
Infrastructure as a Service (IaaS) helps large organizations, and large enterprises in managing and building their IT platforms. This infrastructure is flexible according to the needs of the client.

Advantages of IaaS

- IaaS is cost-effective as it eliminates capital expenses.
- IaaS cloud provider provides better security than any other software.
- IaaS provides remote access.

Disadvantages of IaaS

- In IaaS, users have to secure their own data and applications.
- Cloud computing is not accessible in some regions of the World.

2. Platform as a service (PaaS)

Platform as a Service (PaaS) is a type of cloud computing that helps developers to build applications and services over the Internet by providing them with a platform.
PaaS helps in maintaining control over their business applications.

Advantages of PaaS

- PaaS is simple and very much convenient for the user as it can be accessed via a web browser.
- PaaS has the capabilities to efficiently manage the lifecycle.

Disadvantages of PaaS

- PaaS has limited control over infrastructure as they have less control over the environment and are not able to make some customizations.

- PaaS has a high dependence on the provider.

3. Software as a service (SaaS)

Software as a Service (SaaS) is a type of cloud computing model that is the work of delivering services and applications over the Internet. The SaaS applications are called Web-Based Software or Hosted Software.
SaaS has around 60 percent of cloud solutions and due to this, it is mostly preferred by companies.

Advantages of SaaS

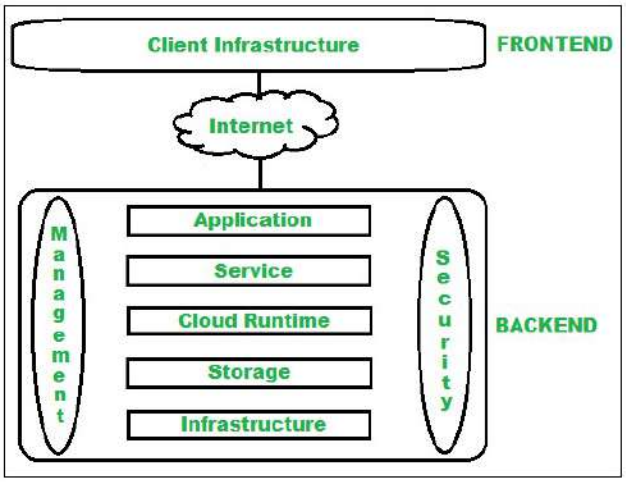
- SaaS can access app data from anywhere on the Internet.
- SaaS provides easy access to features and services.

Disadvantages of SaaS

- SaaS solutions have limited customization, which means they have some restrictions within the platform.
- SaaS has little control over the data of the user.
- SaaS are generally cloud-based, they require a stable internet connection for proper working.

Q5) Explanation of Architecture of cloud computing with suitable diagram.

Ans 5)



The cloud architecture is divided into 2 parts i.e.

1. Frontend
2. Backend

1. Frontend :

Frontend of the cloud architecture refers to the client side of cloud computing system. Means it contains all the user interfaces and applications which are used by the client to access the cloud computing services/resources. For example, use of a web browser to access the cloud platform.

- **Client Infrastructure –** Client Infrastructure is a part of the frontend component. It contains the applications and user interfaces which are required to access the cloud platform.
- In other words, it provides a GUI(Graphical User Interface) to interact with the cloud.

2. Backend :

Backend refers to the cloud itself which is used by the service provider. It contains the resources as well as manages the resources and provides security mechanisms. Along with this, it includes huge storage, virtual applications, virtual machines, traffic control mechanisms, deployment models, etc.

1. **Application –**
   Application in backend refers to a software or platform to which client accesses. Means it provides the service in backend as per the client requirement.
2. **Service –**
   Service in backend refers to the major three types of cloud based services like SaaS, PaaS and IaaS. Also manages which type of service the user accesses.
3. **Runtime Cloud-**
   Runtime cloud in backend provides the execution and Runtime platform/environment to the Virtual machine.
4. **Storage –**
   Storage in backend provides flexible and scalable storage service and management of stored data.
5. **Infrastructure –**
   Cloud Infrastructure in backend refers to the hardware and software components of cloud like it includes servers, storage, network devices, virtualization software etc.
6. **Management –**
   Management in backend refers to management of backend components like application, service, runtime cloud, storage, infrastructure, and other security mechanisms etc.
7. **Security –**
   Security in backend refers to implementation of different security mechanisms in the backend for secure cloud resources, systems, files, and infrastructure to end-users.
8. **Internet –**
   Internet connection acts as the medium or a bridge between frontend and

backend and establishes the interaction and communication between frontend and backend.

9. **Database**– Database in backend refers to provide database for storing structured data, such as SQL and NOSQL databases. Example of Databases services include Amazon RDS, Microsoft Azure SQL database and Google CLoud SQL.
10. **Networking**– Networking in backend services that provide networking infrastructure for application in the cloud, such as load balancing, DNS and virtual private networks.
11. **Analytics**– Analytics in backend service that provides analytics capabillities for data in the cloud, such as warehousing, bussness intellegence and machine learning.

Benefits of Cloud Computing Architecture :

- Makes overall cloud computing system simpler.
- Improves data processing requirements.
- Helps in providing high security.
- Makes it more modularized.
- Results in better disaster recovery.
- Gives good user accessibility.
- Reduces IT operating costs.
- Provides high level reliability.
- Scalability.

Q6) Benefits and limitations of cloud computing

Ans 6)

Advantages of Cloud Computing

1) Back-up and restore data

Once the data is stored in the cloud, it is easier to get back-up and restore that data using the cloud.

2) Improved collaboration

Cloud applications improve collaboration by allowing groups of people to quickly and easily share information in the cloud via shared storage.

3) Excellent accessibility

Cloud allows us to quickly and easily access store information anywhere, anytime in the whole world, using an internet connection. An internet cloud infrastructure increases organization productivity and efficiency by ensuring that our data is always accessible.

4) Low maintenance cost

Cloud computing reduces both hardware and software maintenance costs for organizations.

5) Mobility

Cloud computing allows us to easily access all cloud data via mobile.

6) IServices in the pay-per-use model

Cloud computing offers Application Programming Interfaces (APIs) to the users for access services on the cloud and pays the charges as per the usage of service.

7) Unlimited storage capacity

Cloud offers us a huge amount of storing capacity for storing our important data such as documents, images, audio, video, etc. in one place.

8) Data security

Data security is one of the biggest advantages of cloud computing. Cloud offers many advanced features related to security and ensures that data is securely stored and handled.

Limitations of Cloud Computing

A list of the disadvantage of cloud computing is given below -

1) Internet Connectivity

As you know, in cloud computing, every data (image, audio, video, etc.) is stored on the cloud, and we access these data through the cloud by using the internet connection. If you do not have good internet connectivity, you cannot access these data. However, we have no any other way to access data from the cloud.

2) Vendor lock-in

Vendor lock-in is the biggest disadvantage of cloud computing. Organizations may face problems when transferring their services from one vendor to another. As different vendors provide different platforms, that can cause difficulty moving from one cloud to another.

3) Limited Control

As we know, cloud infrastructure is completely owned, managed, and monitored by the service provider, so the cloud users have less control over the function and execution of services within a cloud infrastructure.

4) Security

Although cloud service providers implement the best security standards to store important information. But, before adopting cloud technology, you should be aware that you will be sending all your organization's sensitive information to a third party, i.e., a cloud computing

service provider. While sending the data on the cloud, there may be a chance that your organization's information is hacked by Hackers.

Experiment :- 2

Aim:- To study and implement Hosted Virtualization using VirtualBox and KVM.

Q1) Virtualization in cloud computing.

Ans1) Virtualisation is a technique how to separate a service from the underlying physical delivery of that service. It is the process of creating a ~~visual~~ virtual version of something like computer hardware. It was initially developed during the mainframe era. It involves using specialised software to create a virtual or software created version of computing resource rather than the actual version of the same resource. With the help of virtualisation, multiple operating systems and applications can run on the same machine and its same hardware at the same time, increasing the utilization and flexibility of hardware.

Q2) Benefits of Virtualisation

Ans2) 1) More flexible and ~~efficient~~ allocation of resources
2) Pay per-use of IT infrastructure on demand.
3) Enables running multiple operating systems.
4) Reduced expense
5) Resiliency
6) Increased ~~efficiency~~
7) High availability
8) Greener IT.

Q3) Hypervisor

Ans 3) A hypervisor is a software that creates and runs virtual machines (VMs). Also known as Virtual Machine Monitor (VMM) isolates the hypervisor operating system and resources from the virtual machines and enables the creation and management of those VMs. Physical hardware when used as hypervisor is called as host while the many VMs that use its resources are guests.

Two types of Hypervisors:-

1) Native or bare metal hypervisor (Type 1)
It runs directly through the host's hardware to manage guest operating systems.
This is the most common type of hypervisor.
eg: KVM, VMware, Microsoft Hyper-V, etc.

2) Hosted or embedded hypervisor (Type 2)
It runs on a conventional OS as a software layer or application. This type is better for individual users who want to run multiple operating systems on one computer.
eg:- Oracle, Virtual Box, VMware WorkStation

THADOMAL SHAHANI TSEC ENGINEERING COLLEGE

Q4) Comparison between VirtualBox and KVM.

KVM	VirtualBox
1) It is a Type 1 hypervisor, ∴ provides better performance.	1) It is a Type 2 hypervisor. ∴ slower in performance.
2) It is usually used in Linux environments.	2) It is compatible with Windows, Mac OS, Linux, etc.
3) Allows dynamic allocation of resources to virtual machines.	3) Has limitations in resource management.
4) Gives better hardware support.	4) Does not have same level of hardware support.
5) It is managed through command line tools and APIs, thus making it suitable for advanced users.	5) It provides a user friendly graphical interface, suitable for beginners.

A+
5/
24/1/24

Name : Arnav Malvia

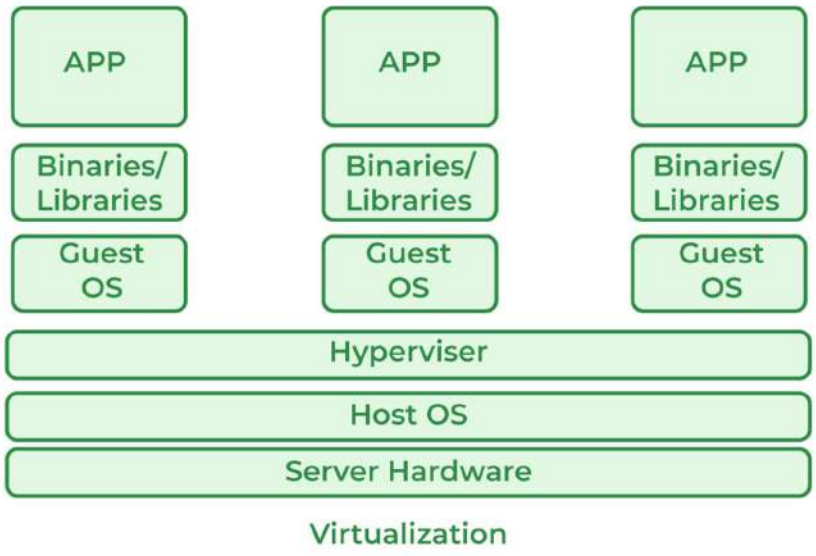
Roll No.: 2103109

Batch : C23

Experiment 2:- To study and implement Hosted Virtualization using VirtualBox& KVM

Q1) Virtualization in cloud computing

Ans 1) Virtualization is a technique how to separate a service from the underlying physical delivery of that service. It is the process of creating a virtual version of something like computer hardware. It was initially developed during the mainframe era. It involves using specialized software to create a virtual or software-created version of a computing resource rather than the actual version of the same resource. With the help of Virtualization, multiple operating systems and applications can run on the same machine and its same hardware at the same time, increasing the utilization and flexibility of hardware.
In other words, one of the main cost-effective, hardware-reducing, and energy-saving techniques used by cloud providers is Virtualization. Virtualization allows sharing of a single physical instance of a resource or an application among multiple customers and organizations at one time. It does this by assigning a logical name to physical storage and providing a pointer to that physical resource on demand. The term virtualization is often synonymous with hardware virtualization, which plays a fundamental role in efficiently delivering Infrastructure-as-a-Service (IaaS) solutions for cloud computing. Moreover, virtualization technologies provide a virtual environment for not only executing applications but also for storage, memory, and networking.



Virtualization

Types of Virtualization

1. Application Virtualization
2. Network Virtualization
3. Desktop Virtualization
4. Storage Virtualization
5. Server Virtualization
6. Data virtualization

1. Application Virtualization: Application virtualization helps a user to have remote access to an application from a server. The server stores all personal information and other characteristics of the application but can still run on a local workstation through the internet. An example of this would be a user who needs to run two different versions of the same software. Technologies that use application virtualization are hosted applications and packaged applications.
2. Network Virtualization: The ability to run multiple virtual networks with each having a separate control and data plan. It co-exists together on top of one physical network. It can be managed by individual parties that are potentially confidential to each other. Network virtualization provides a facility to create and provision virtual networks, logical switches, routers, firewalls, load balancers, Virtual Private Networks (VPN), and workload security within days or even weeks.
3. Desktop Virtualization: Desktop virtualization allows the users' OS to be remotely stored on a server in the data center. It allows the user to access their desktop virtually, from any location by a different machine. Users who want specific operating systems other than Windows Server will need to have a virtual desktop. The main benefits of desktop virtualization are user mobility, portability, and easy management of software installation, updates, and patches.
4. Storage Virtualization: Storage virtualization is an array of servers that are managed by a virtual storage system. The servers aren't aware of exactly where their data is stored and instead function more like worker bees in a hive. It makes managing storage from multiple sources be managed and utilized as a single repository. storage virtualization software maintains smooth operations, consistent performance, and a continuous suite of advanced functions despite changes, breaks down, and differences in the underlying equipment.
5. Server Virtualization: This is a kind of virtualization in which the masking of server resources takes place. Here, the central server (physical server) is divided into multiple different virtual servers by changing the identity number, and processors. So, each system can operate its operating systems in an isolated manner. Where each sub-server knows the identity of the central server. It causes an increase in performance and reduces the operating cost by the deployment of main server resources into a sub-server resource. It's beneficial in virtual migration, reducing energy consumption, reducing infrastructural costs, etc.
6. Data Virtualization: This is the kind of virtualization in which the data is collected from various sources and managed at a single place without knowing more about the technical information like how data is collected, stored & formatted then arranged that data logically so that its virtual view can be accessed by its interested people and stakeholders, and users through the various cloud services remotely. Many big giant companies are providing their services like Oracle, IBM, At scale, Cdata, etc.

Q2) Benefits of virtualization

Ans 2)

1) More flexible and efficient allocation of resources.
2) Pay peruse of the IT infrastructure on demand.
3) Enables running multiple operating systems.
4)Reduced expenses:- Computing power comes at a price. If the only way to get more resources is to purchase new hardware, that price becomes hefty. With virtualization tactics, you can take a hard look at your existing infrastructure and identify wasted or idle computing resources.Too often, organizations deploy servers to run applications that consume only a fraction of their available resources. Such servers never make use of their full potential. To make matters worse, when their applications are not running, these servers sit entirely idle.In a virtualized environment, you can assign each VM precisely the amount of computing power it needs to do its job. The remaining resources are then available for other VMs and their

applications.Virtualization costs are almost always lower than the cost of purchasing and maintaining additional hardware.

5)Resiliency:- A virtualized server environment is not bound to hardware like a traditional environment. You can easily back up, copy, and clone VMs to different physical hardware.Waiting for new hardware to be ready for deployment can take days, weeks, or even months. Meanwhile, you can deploy a VM backup in a matter of minutes. When Murphy's Law finally catches up with you, you will be thankful you can rapidly deploy your VM on a different machine, in a different location, with minimal hassle.

6)High availability:- Since you can clone a VM almost effortlessly, you can easily set up redundant virtualized environments with exceptionally high availability. By automatically monitoring VM status and rapidly switching to backup VMs in an outage, virtualization provides an extremely reliable system with no single point of failure in hardware or software.These "failover" systems enable you to seamlessly continue operating your VM from its last working state. This maximizes service availability no matter what goes wrong.You can remotely monitor, configure, and restart your entire virtual environment. This provides developers constant access, no matter how far they may be from the physical hardware, which helps further mitigate any potential downtime.

7)Increased efficiency:- Virtual environments are much easier to maintain than physical environments. Rather than managing numerous physical servers requiring individual attention, virtualization enables you to configure, monitor, and update all your VMs from a single machine. This saves time deploying updates, implementing security patches, and installing new software.With less physical hardware to worry about, your IT department spends less time maintaining physical machines. Your developers enjoy the efficiency of rapidly spinning up a VM without having to worry about adding new hardware.By nature, virtual environments are inherently scalable. You can easily deploy many instances of the same VM to help handle heavy load, offering efficient scalability that is always ready to support and sustain growth.

8)DevOps made easy:- Virtualization is a developer's best friend. It effectively segments production and development environments without extra hardware.It is simple to clone a VM to set up testing environments. You can test features and squash bugs without affecting your live product.With traditional hardware-based environments, developers need to manage all the updates and maintenance for their development machines. Maintaining an accurate representation of live servers for testing is also an ongoing challenge.VMs quickly solve all these issues. Virtualization provides on-demand access to an infinite number of perfectly replicated virtual machines for developers to play with.Developers take advantage of virtualization to help them expedite updates, improve software security, and maintain an efficient pipeline between development, testing, and deployment.

9)Greener IT:- In the long run, virtualization is an eco-friendly approach to IT. Reducing hardware requirements also reduces power consumption, ultimately minimizing our carbon footprint.This is good for both the environment and your bottom line. Power savings make it cheaper to maintain servers and data centers. You can then invest all that money in other ventures, like making your business even more environmentally friendly.

Q3) What is Hypervisor? Give examples of Hypervisor

Ans 3) A hypervisor is a form of virtualization software used in Cloud hosting to divide and allocate the resources on various pieces of hardware. The program which provides partitioning, isolation, or abstraction is called a virtualization hypervisor. The hypervisor is a hardware virtualization technique that allows multiple guest operating systems (OS) to run on a single host system at the same time. A hypervisor is sometimes also called a virtual machine manager(VMM).

Examples:-

1) VMware hypervisors like vSphere
2)Microsoft Hyper-V
3)Oracle VM Server
4)Citrix Hypervisor
5) Oracle VirtualBox
6)Oracle Solaris Zones
7)Oracle VM Server for x86

Q4) Explain the types of Hypervisors

Ans 4) There are two main types of hypervisor:

1) Native or “bare metal” hypervisors
2) Hosted or “embedded” hypervisors

A bare metal hypervisor is installed directly on the hardware of your machine, whereas a hosted hypervisor is installed on your operating system.

Bare metal hypervisors are typically faster and more efficient because they have direct access to the underlying hardware and don’t need to go through the operating system layer. Since they don’t have to compete with other applications or the OS, they can take all the available physical hardware power and allocate it to virtual machines. They also tend to be more secure, because, without an operating system on the host, less attack surface is available for malicious intruders.
However, hosted hypervisors are significantly easier to set up and get running, as you can use the more user-friendly operating system. They’re often used for testing and development purposes, as they can run on the OS to try out new programs or features without affecting the host OS.
VMware and Hyper-V are two key examples of hypervisor, with VMware owned by Dell and Hyper-V created by Microsoft. VMware software is made for cloud computing and virtualization, and it can install a hypervisor on your physical servers to allow multiple virtual machines to run at the same time. Hyper-V does the same thing, but you can also virtualize servers. Hyper-V comes pre-installed with Windows 10. Both are bare metal (native) hypervisors. Oracle VM VirtualBox is a hosted hypervisor.

Q5) Comparisons between VirtualBox and KVM

Ans 5) Kernel-based Virtual Machine (KVM) and Oracle VM VirtualBox are Linux-based virtualization solutions that allow users to run various operating systems without “bare-metal” hardware. Both are full virtualization solutions and open source. The key differences are in how they work and the types of features and use cases they support.

What Is KVM?

KVM, also known as the KVM hypervisor, is a virtualization module that turns the Linux kernel into a hypervisor. The Linux kernel is the main component of a Linux operating system (OS) that acts as the core interface between a computer's hardware and its processes. A hypervisor is software that creates and runs virtual machines (VMs), allowing a host computer to support multiple guest VMs by virtually sharing its resources, such as memory and processing.

Advantages of KVM

A KVM hypervisor enables full virtualization capabilities, giving each VM all the features of a physical system, including the basic input/output system (BIOS) and hardware, such as processors, memory, storage, and network cards.

The primary advantages of KVM are:

- **Performance**: KVM is a type 1 or "bare metal" hypervisor, meaning it runs directly on the host machine's physical hardware. This means it doesn't have to load an underlying OS and has direct access to the underlying hardware without having to contend for virtualization with other software such as other operating systems and device drivers. This gives KVM an inherent advantage in terms of performance and efficiency.
- **Maturity**: KVM is more than 15 years old and has more than 1,000 code contributors. This high level of maturity means it's well developed and very debugged. There are plenty of experts to go to for support and questions.
- **Scalability**: The KVM hypervisor automatically scales to respond to heavy loads once the number of VMs increases. It also enables clustering for thousands of nodes, which helps set the foundation for a cloud-based infrastructure.
- **Security**: As part of the Linux kernel source code, KVM benefits from rigorous development and testing processes, as well as continuous security patching.
- **Affordability**: Since it's open source and available as a Linux kernel module, KVM costs nothing out of the box.

What Is VirtualBox?

Developed by Oracle, VirtualBox is an open source virtualization software that is a type 2 hypervisor. That means it runs on a conventional OS just as other computer programs do and abstracts guest operating systems from the host OS. Type 2 hypervisors like VirtualBox are sometimes called "hosted" hypervisors because they rely on the host machine's pre-existing OS to manage calls to CPU, memory, storage, and network resources. VirtualBox supports the following guest operating systems:

- Windows 10, 8, 7, XP, Vista, 2000, NT, and 98
- Linux distributions based on Linux kernel 2.4 and newer, including Ubuntu, Debian, OpenSUSE, Mandriva/Mandrake, Fedora, RHEL, and Arch Linux
- Solaris and OpenSolaris
- macOS X Server Leopard and Snow Leopard
- OpenBSD and FreeBSD
- MS-DOS
- OS/2
- QNX
- BeOS R5
- Haiku
- ReactOS

Advantages of VirtualBox

VirtualBox offers many features that should be attractive to IT and development professionals:

- **Easy to use**: VirtualBox is very lightweight, making installation and usage very easy. It uses the .ova format, which makes it very easy to export and import appliances, and it simplifies networking, shared folders, guest addition, and virtual media management.
- **Powerful**: With the latest AMD and Intel hardware support, VirtualBox provides quick execution and chip-level virtualization support. It allows you to easily run up to 32 vCPUs while enjoying a range of virtual storage controllers. It also supports video acceleration and 3D graphics, remote display, USB and serial connections, crisp audio, and more.
- **Easy resource management**: VirtualBox has an array of features that make it easy to manage compute resources:
    - It allows you to throttle or cap CPU execution, network I/O, disk read and writes, and other host resources. This means that if hackers or rogue guests access your virtual machines, they can't consume more than your set caps.
    - You can use a web service API to remotely control your VirtualBox.
    - It allows up to 36 vNICs or virtual network interface cards so that you can test even the most complex configurations and setups for your network.
    - You can easily clone your VMs and get branched or multigenerational snapshots, which give you the option to revert to a previous state.
- **Robust community support**: VirtualBox is free and open source and supported by a large community of Oracle users. There are many support resources available, such as forums.virtualbox.org.

Q6) Write Steps of installation for the above-mentioned experiment along with screen shots

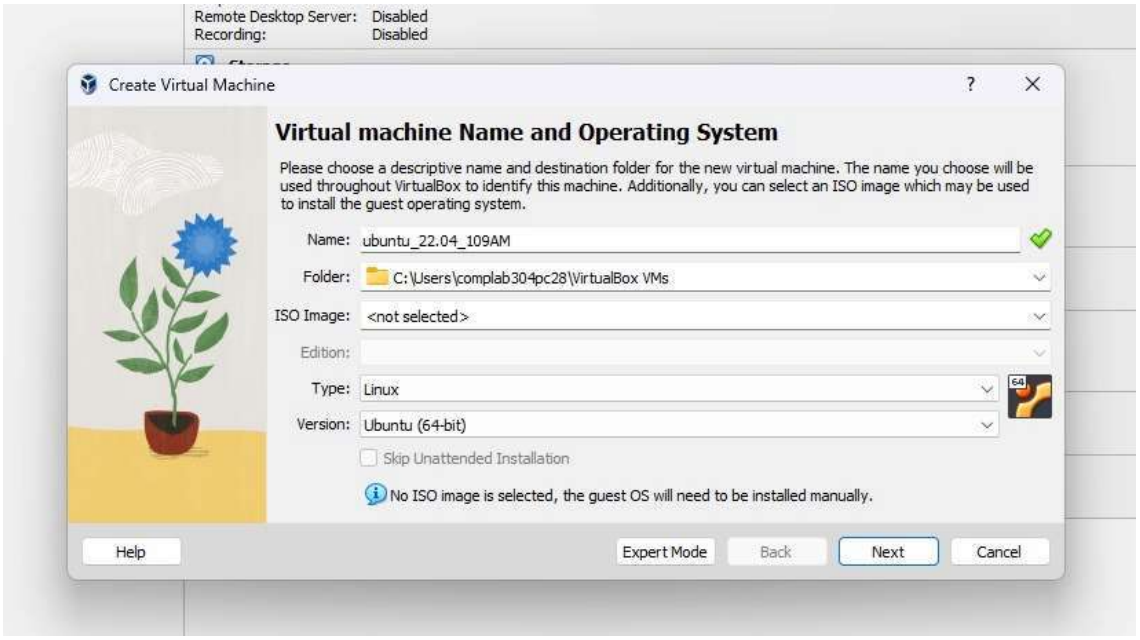
Ans 6)

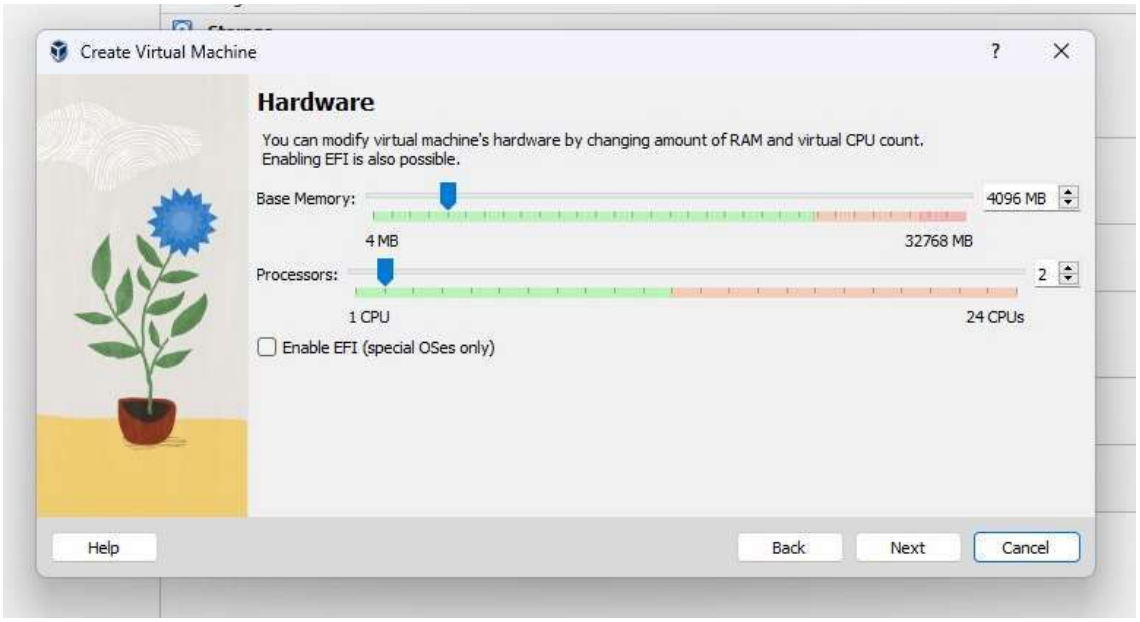
1) Creating a new system

2) Changing the network adapter to bridged adapter

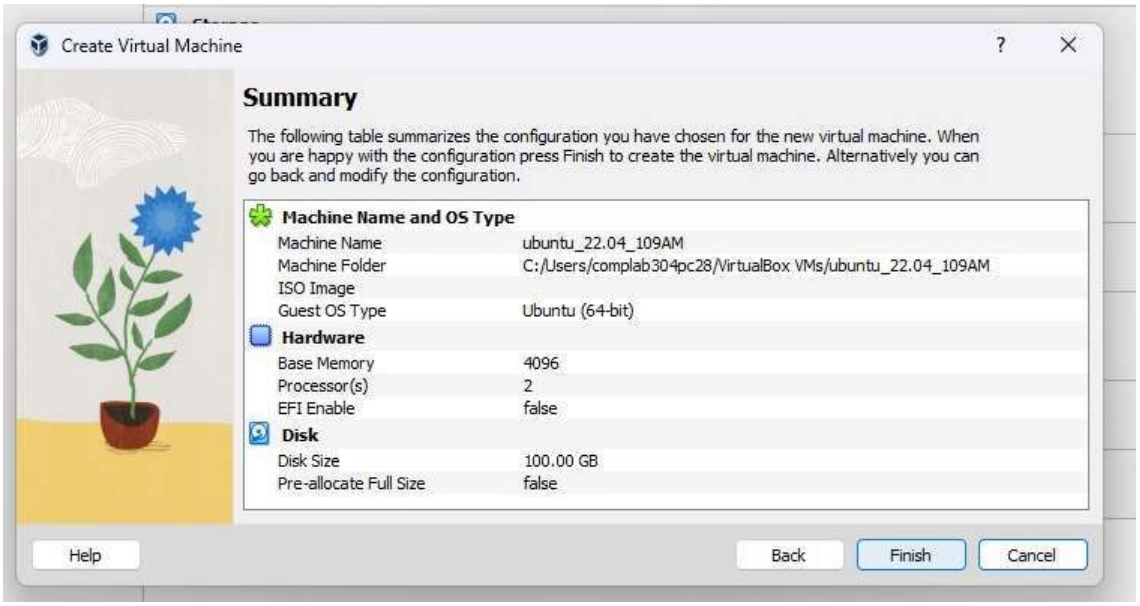
3) Adjusting the system's RAMs and no of CPUs required

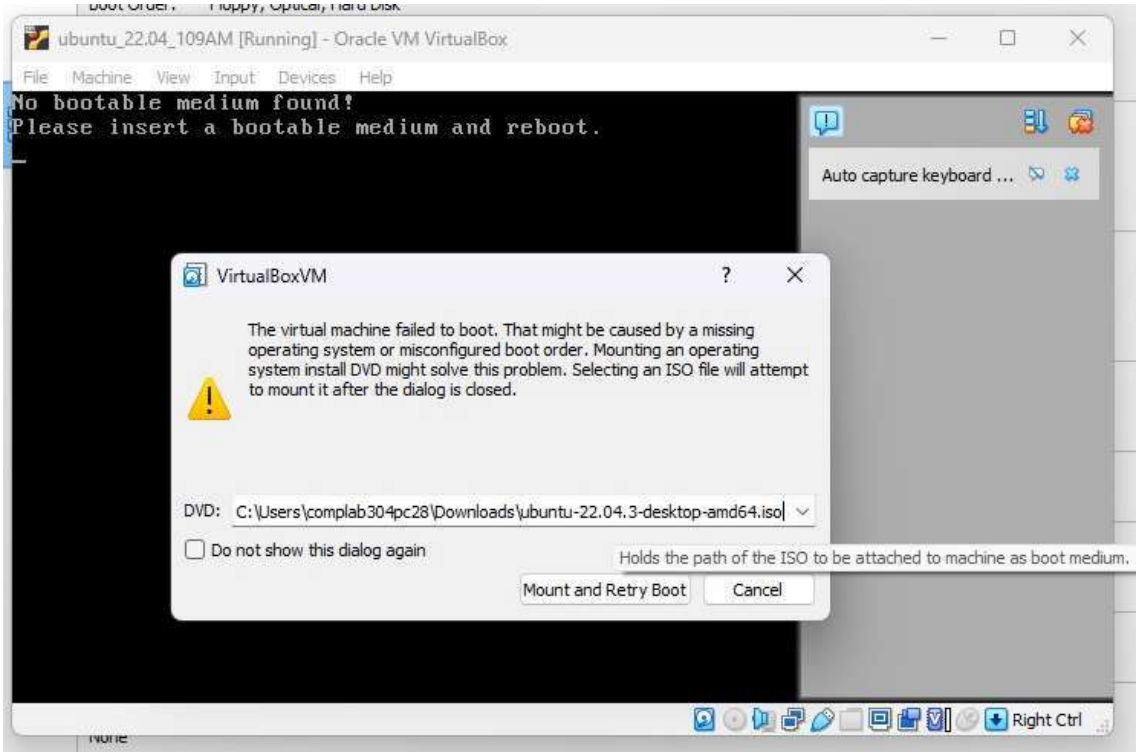
4) Installing Ubuntu virtual system

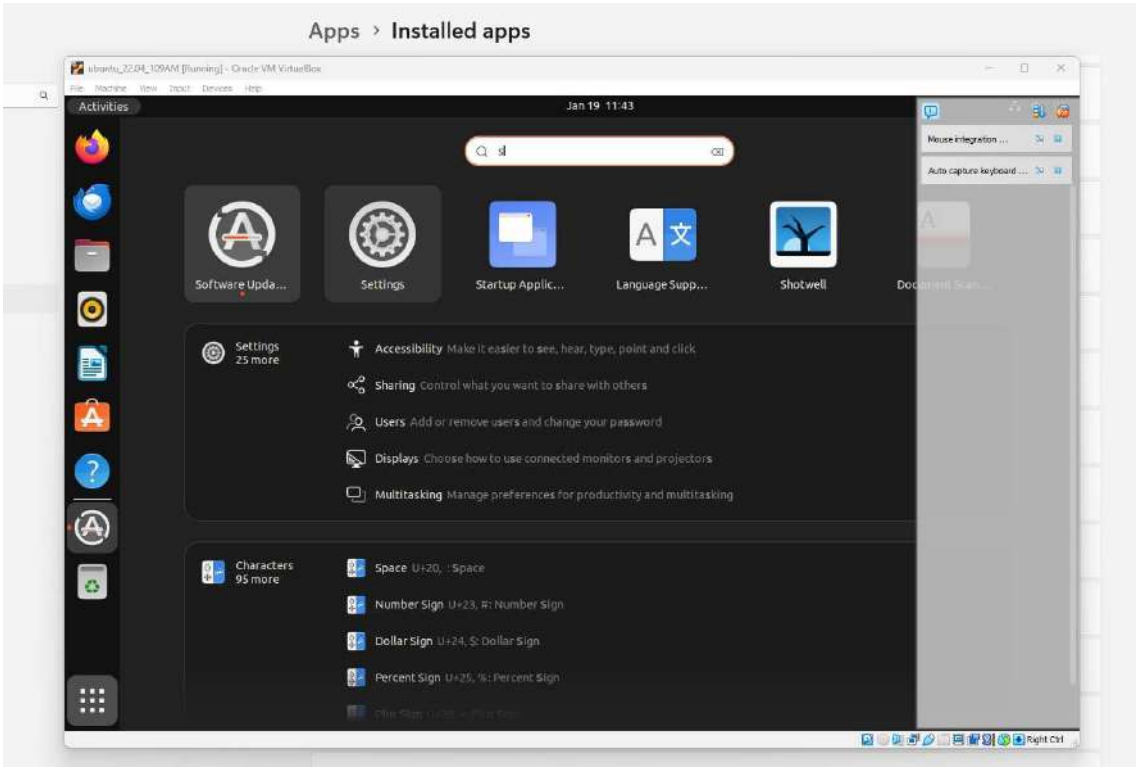

Remote Desktop Server: Disabled
Recording: Disabled
Create Virtual Machine
Virtual machine Name and Operating System
Please choose a descriptive name and destination folder for the new virtual machine. The name you choose will be used throughout VirtualBox to identify this machine. Additionally, you can select an ISO image which may be used to install the guest operating system.
Name: ubuntu_22.04_109AM
Folder: C:\Users\complab304pc28\VirtualBox VMs
ISO Image: <not selected>
Edition:
Type: Linux
Version: Ubuntu (64-bit)
Skip Unattended Installation
No ISO image is selected, the guest OS will need to be installed manually.
Help
Expert Mode
Back
Next
Cancel

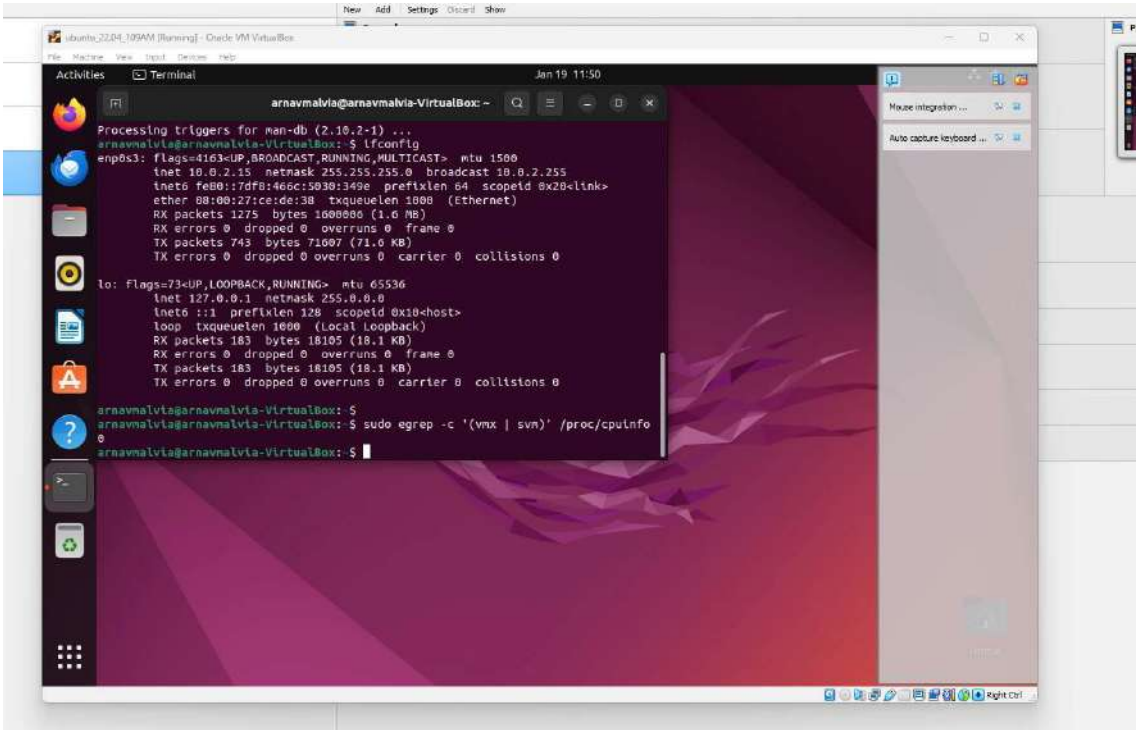

Create Virtual Machine
Hardware
You can modify virtual machine's hardware by changing amount of RAM and virtual CPU count.
Enabling EFI is also possible.
Base Memory:
4096 MB
4 MB
32768 MB
Processors:
2
1 CPU
24 CPUs
Enable EFI (special OSes only)
Help
Back
Next
Cancel

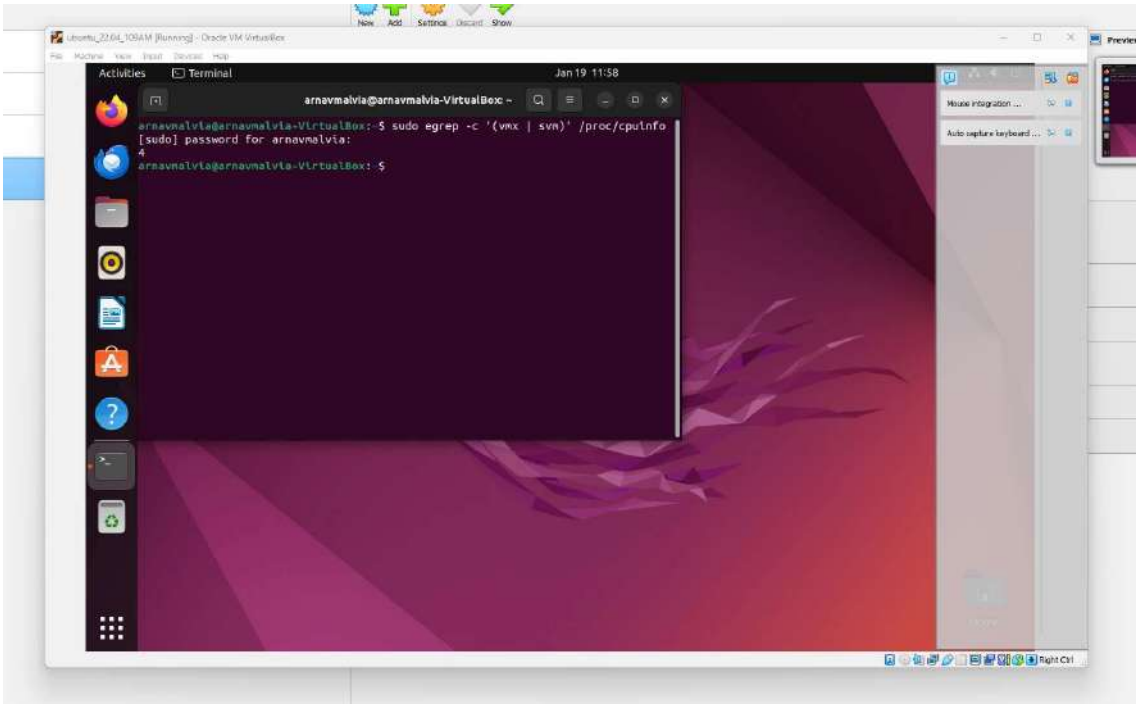

Create Virtual Machine
Virtual Hard disk
If you wish you can add a virtual hard disk to the new machine. You can either create a new hard disk file or select an existing one. Alternatively you can create a virtual machine without a virtual hard disk.
Create a Virtual Hard Disk Now
Disk Size:
100.00 GB
4.00 MB
2.00 TB
Pre-allocate Full Size
Use an Existing Virtual Hard Disk File
ubuntu2020.vdi (Normal, 100.00 GB)
Do Not Add a Virtual Hard Disk
Help
Back
Next
Cancel


Create Virtual Machine
Summary
The following table summarizes the configuration you have chosen for the new virtual machine. When you are happy with the configuration press Finish to create the virtual machine. Alternatively you can go back and modify the configuration.
Machine Name and OS Type
Machine Name ubuntu_22.04_109AM
Machine Folder C:/Users/complab304pc28/VirtualBox VMs/ubuntu_22.04_109AM
ISO Image
Guest OS Type Ubuntu (64-bit)
Hardware
Base Memory 4096
Processor(s) 2
EFI Enable false
Disk
Disk Size 100.00 GB
Pre-allocate Full Size false
Help
Back
Finish
Cancel


ubuntu_22.04_109AM [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
No bootable medium found!
Please insert a bootable medium and reboot.
Auto capture keyboard ...
VirtualBoxVM
The virtual machine failed to boot. That might be caused by a missing operating system or misconfigured boot order. Mounting an operating system install DVD might solve this problem. Selecting an ISO file will attempt to mount it after the dialog is closed.
DVD: C:\Users\complab304pc28\Downloads\ubuntu-22.04.3-desktop-amd64.iso
Do not show this dialog again
Holds the path of the ISO to be attached to machine as boot medium.
Mount and Retry Boot
Cancel
Right Ctrl


Apps › Installed apps
Activities
Jan 19 11:43
Software Upda...
Settings
Startup Applic...
Language Supp...
Shotwell
Settings
25 more
Accessibility Make it easier to see, hear, type, point and click
Sharing Control what you want to share with others
Users Add or remove users and change your password
Displays Choose how to use connected monitors and projectors
Multitasking Manage preferences for productivity and multitasking
Characters
95 more
Space U+20, : Space
Number Sign U+23, #: Number Sign
Dollar Sign U+24, $: Dollar Sign
Percent Sign U+25, %: Percent Sign


Activities
Terminal
Jan 19 11:50
arnavmalvia@arnavmalvia-VirtualBox: ~
Processing triggers for man-db (2.10.2-1) ...
arnavmalvia@arnavmalvia-VirtualBox:~$ ifconfig
enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
inet 10.0.2.15 netmask 255.255.255.0 broadcast 10.0.2.255
inet6 fe80::7df8:466c:5030:349e prefixlen 64 scopeid 0x20<link>
ether 08:00:27:ce:de:38 txqueuelen 1000 (Ethernet)
RX packets 1275 bytes 1600800 (1.6 MB)
RX errors 0 dropped 0 overruns 0 frame 0
TX packets 743 bytes 71607 (71.6 KB)
TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
inet 127.0.0.1 netmask 255.0.0.0
inet6 ::1 prefixlen 128 scopeid 0x10<host>
loop txqueuelen 1000 (Local Loopback)
RX packets 183 bytes 18105 (18.1 KB)
RX errors 0 dropped 0 overruns 0 frame 0
TX packets 183 bytes 18105 (18.1 KB)
TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0
arnavmalvia@arnavmalvia-VirtualBox:~$
arnavmalvia@arnavmalvia-VirtualBox:~$ sudo egrep -c '(vmx | svm)' /proc/cpuinfo
0
arnavmalvia@arnavmalvia-VirtualBox:~$
Mouse integration ...
Auto capture keyboard ...
Right Ctrl


Activities
Terminal
Jan 19 11:58
arnavmalvia@arnavmalvia-VirtualBox: ~
arnavmalvia@arnavmalvia-VirtualBox:~$ sudo egrep -c '(vmx | svm)' /proc/cpuinfo
[sudo] password for arnavmalvia:
4
arnavmalvia@arnavmalvia-VirtualBox:~$
Mouse integration ...
Auto capture keyboard ...
Right Ctrl

CCL

THADOMAL SHAHANI TSEC ENGINEERING COLLEGE

Experiment :- 3

Aim:- To study and implement ~~base~~ bare metal virtualisation using XEN.

Q1) Explain functions performed by Base-Metal hypervisor.

Ans) Hardware Abstraction :-

A base metal hypervisor's main function is to abstract the underlying physical hardware. It creates a virtualisation layer that allows multiple virtual machines to run independently on the same server.

Resource Allocation :-

The hypervisor manages and allocates physical resources such as CPU, memory, storage and N/W bandwidth among the virtual machines. It ensures fair and efficient allocation.

Isolation :-

They provide strong isolation b/w ~~wt~~ virtual machines. It embraces the security and stability preventing one VM from affecting others.

Live migration :-

Bare metal hypervisors often support live migrations allowing VMs to be moved from one physical host to another without disrupting their operation. This functionality is useful for load balancing, maintenance and ensuring high availability.

Q2 Compare Hosted and bare metal hypervisors.

Hosted	Bare Metal
1) Hosted runs on a conventional operating system.	1) Bare-metal runs directly on the system with VMs running on them.
2) OS virtualisation	2) Hardware Virtualisation.
3) Runs as an application on the host OS.	3) Guest OS and applications
4) Not so stable because of its reference on the underlying OS.	4) Better stability.
5) Slower because of the systems dependency.	5) Faster.
6) Less secure	6) More secure
7) Example:- Vmware ESXI, Microsoft Hyper-V.	7) Example:- VMware workstation player, Microsoft Virtual PC.

THADOMAL SHAHANI TSEC ENGINEERING COLLEGE

Q3 Explain the following terms:-

Ans 3 1) Horizontal and Vertical Scaling:-

In horizontal scaling, we enhance the performance of the server by adding more machines/ servers to the network, sharing the ~~cost~~ processing and memory workload across multiple devices and it helps to distribute the load among these servers. There is data inconsistency and utilizes network calls.

In vertical scaling upgrading the capacity of a single machine or moving to a new machine with more power is called vertical scaling. You can add more power to your machine by adding better processors, increasing RAM or other power increasing adjustments.

2) Autoscalings:-

Auto scaling is a feature in cloud computing that allows a cloud based application to automatically adjust the resources it uses such as servers, compute instances based on demand. The goal of Auto scaling is to ensure that the application has sufficient resources to meet performance goals and maintain availability, while also optimizing resource utilisation and ~~mimi~~ minimizing costs.

3) Load Balancing:-

A load balancer is a network device that fits between a set of backend servers and clients. It distributes the incoming traffic to multiple servers to reduce ~~the~~ the load. ~~balance~~ Load balancers typically use various algorithms such as round-robin to determine

which server to send incoming traffic. Load balancers can also provide features such as SSL terminations and health checks to maintain the server's health.

Q4) Write steps of installation for the above mentioned experiment along with screenshots.

Ans4) Step 1: Install XenServer

- Insert bootable CD into CD ROM or bootable pendrive and make first boot device from Bios.
- Press F2 for advanced options and make enabled virtualisation technology from Bios.
- We begin by choosing the keymap i.e keyboard layout.
- Press enter to load device drivers.
- Press enter to accept end user agreement
- Select Appropriate disk on which you want to install Xen Server.
- Select appropriate Installation Media.
- Press yes to select additional packages for installation, otherwise press No.
- Pick a root password, you will need this for loging in via the console or via the XenServer client on windows.
- The motherboard we are installing have 2 ethernet ports, both of which are supported by Xen Server.
- Select time zone

- Specify NTP server address and start installation to set your time.
- Press Enter to start installation of XenServer.

Step 2: Connect XenCenter to XenServer

- Download the XenCenter a management utility from XenServer IP address as a URL on browser. Install XenCenter and open it from start Menu ~~fro~~ of Windows on Machine 2.
- Once you click on Citrix XenCenter.
- To ~~center~~ connect to the XenServer host which is configured earlier, click ADD a server.
- Enter IP address of XenServer and Enter User login credentials and click Add.
- Once you clicked on Add, it will be able to configure a master password for all the XenServers.
- XenServer is added ~~to~~ now to XenCenter.

Step 3: Create Local ISO Storage

Before creating Virtual machine we have to create storage repository which is nothing but shared directory on XenCenter.

Using local command shell

First view XenDirectory structure #df -h

Now make folder to store ISO images #mkdir/var/ISO-images

Using wget command download iso files.

Step 4: Installation of Virtual Machine from XenCenter.

- Right Click on Xen Server icon on Xen Centre and select New VM.
- Select VM template
- Name the virtual machine.
- Locate the operating system installation media.
- Allocate CPU and memory to VM.
- Select networking
- Finish

B+

CP 5

28/2/24

Arnav Malvia
C23
2103109

CCL
Experiment 3

Aim: To study and Implement Bare-metal Virtualization using Xen.

Objective: To understand the concept of hypervisors, its types and functions in CloudComputing.

Theory:

Virtualization in Cloud Computing

Virtualization is the "creation of a virtual (rather than actual) version of something, such as a server, a desktop, a storage device, an operating system or network resources".

In other words, Virtualization is a technique, which allows to share a single physical instance of a resource or an application among multiple customers and organizations. It does by assigning a logical name to a physical storage and providing a pointer to that physical resource when demanded.

What is the concept behind the Virtualization?

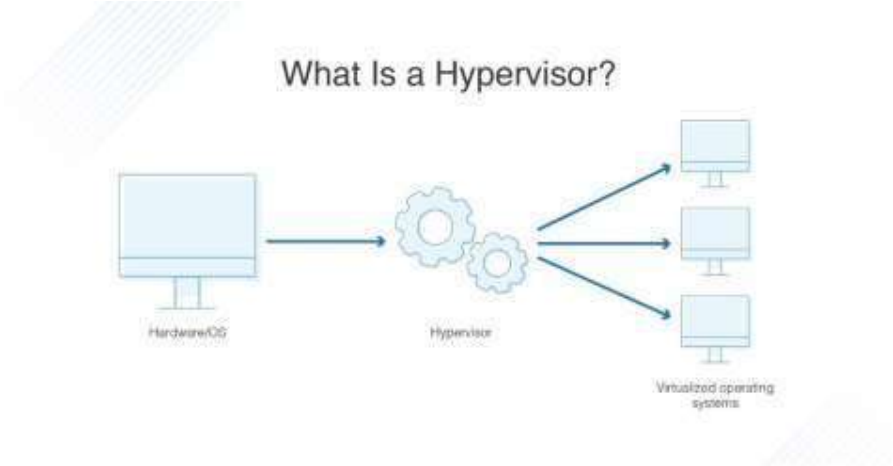
Creation of a virtual machine over existing operating system and hardware is known asHardware Virtualization. A Virtual machine provides an environment that is logically separated from the underlying hardware.

The machine on which the virtual machine is going to create is known as Host Machine andthat virtual machine is referred as a Guest Machine.

What is a hypervisor?

A hypervisor, also known as a virtual machine monitor or VMM. The hypervisor is a piece of software that allows us to build and run virtual machines which are abbreviated as VMs.

A hypervisor allows a single host computer to support multiple virtual machines (VMs) by sharing resources including memory and processing.



What is the use of a hypervisor?

Hypervisors allow the use of more of a system's available resources and provide greater ITversatility because the guest VMs are independent of the host hardware which is one of themajor benefits of the Hypervisor.

In other words, this implies that they can be quickly switched between servers. Since a hypervisor with the help of its special feature, it allows several virtual machines to operate ona single physical server. So, it helps us to reduce:

- The Space efficiency
- The Energy uses
- The Maintenance requirements of the server.

Kinds of hypervisors

There are two types of hypervisors: "Type 1" (also known as "bare metal") and "Type 2" (also known as "hosted"). A type 1 hypervisor functions as a light operating system that operates directly on the host's hardware, while a type 2 hypervisor functions as a softwarelayer on top of an operating system, similar to other computer programs.

Since they are isolated from the attack-prone operating system, bare-metal hypervisors areextremely stable.

Furthermore, they are usually faster and more powerful than hosted hypervisors. For these purposes, the majority of enterprise businesses opt for bare-metal hypervisors for their datacenter computing requirements.

While hosted hypervisors run inside the OS, they can be topped with additional (anddifferent) operating systems.

The hosted hypervisors have longer latency than bare-metal hypervisors which is a very major disadvantage of the it. This is due to the fact that contact between the hardware and thehypervisor must go through the OS's extra layer.

The Type 1 hypervisor

The native or bare metal hypervisor, the Type 1 hypervisor is known by both names.

It replaces the host operating system, and the hypervisor schedules VM services directly tothe hardware.

The type 1 hypervisor is very much commonly used in the enterprise data center or otherserver-based environments.

It includes KVM, Microsoft Hyper-V, and VMware vSphere. If we are running the updated version of the hypervisor then we must have already got the KVM integrated into the Linux kernel in 2007.

The Type 2 hypervisor

It is also known as a hosted hypervisor, The type 2 hypervisor is a software layer orframework that runs on a traditional operating system.

It operates by separating the guest and host operating systems. The host operating systemschedules VM services, which are then executed on the hardware.

Individual users who wish to operate multiple operating systems on a personal computershould use a form 2 hypervisor.

This type of hypervisor also includes the virtual machines with it.

Hardware acceleration technology improves the processing speed of both bare-metal andhosted hypervisors, allowing them to build and handle virtual resources more quickly.

On a single physical computer, all types of hypervisors will operate multiple virtual servers for multiple tenants. Different businesses rent data space on various virtual servers from public cloud service providers. One server can host multiple virtual servers, each of which isrunning different workloads for different businesses.

Benefits of hypervisors

Using a hypervisor to host several virtual machines has many advantages:

- Speed: The hypervisors allow virtual machines to be built instantly unlike bare-metal servers. This makes provisioning resources for complex workloads much simpler.
- Efficiency: Hypervisors that run multiple virtual machines on the resources of a single physical machine often allow for more effective use of a single physical server.
- Flexibility: Since the hypervisor distinguishes the OS from the underlying hardware, the program no longer relies on particular hardware devices or drivers, bare-metal hypervisors enable operating systems and their related applications to operate on a variety of hardware types.
- Portability: Multiple operating systems can run on the same physical server thanks tohypervisors (host machine). The hypervisor's virtual machines are portable because they are separate from the physical computer.

As an application requires more computing power, virtualization software allows it to access additional machines without interruption.

Difference between Type-1 and Type-2 Hypervisors:

Criteria	Type 1 hypervisor	Type 2 hypervisor
AKA	Bare-metal or Native	Hosted
Definition	Runs directly on the system with VMs running on them	Runs on a conventional Operating System
Virtualization	Hardware Virtualization	OS Virtualization
Operation	Guest OS and applications run on the hypervisor	Runs as an application on the host OS
Scalability	Better Scalability	Not so much, because of its reliance on the underlying OS.
Setup/Installation	Simple, as long as you have the necessary hardware support	Lot simpler setup, as you already have an Operating System.

System Independence	Has direct access to hardware along with virtual machines it hosts	Are not allowed to directly access the host hardware and its resources
Speed	Faster	Slower because of the system's dependency
Performance	Higher-performance as there's no middle layer	Comparatively has reduced performance rate as it runs with extra overhead
Security	More Secure	Less Secure, as any problem in the base operating system affects the entire system including the protected Hypervisor
Examples	• VMware ESXi • Microsoft Hyper-V • Citrix XenServer	• VMware Workstation Player • Microsoft Virtual PC • Sun's VirtualBox

What is Cloud Scaling?

In cloud computing, scaling is the process of adding or removing compute, storage, and network services to meet the demands a workload makes for resources in order to maintainavailability and performance as utilization increases. Scaling generally refers to adding or reducing the number of active servers (instances) being leveraged against your workload'sresource demands. Scaling up and scaling out refer to two dimensions across which resources—and therefore, capacity—can be added.

What Factors Impact Cloud Resource Demands?

The demands of your cloud workloads for computational resources are usually determinedby:

- The number of incoming requests (front-end traffic)
- The number of jobs in the server queue (back-end, load-based)
- The length of time jobs have waited in the server queue (back-end, time-based)

Scaling Up & Scaling Out

Scaling up refers to making an infrastructure component more powerful—larger or faster—soit can handle more load, while scaling out means spreading a load out by adding additional components in parallel.

<u>Scale Up (Vertical Scaling)</u>

Scaling up is the process of resizing a server (or replacing it with another server) to give itsupplemental or fewer CPUs, memory, or network capacity.

Benefits of Scaling Up

- Vertical scaling minimizes operational overhead because there is only one server to manage. There is no need to distribute the workload and coordinate among multiple servers.
- Vertical scaling is best used for applications that are difficult to distribute. For example, when a relational database is distributed, the system must accommodate

transactions that can change data across multiple servers. Major relational databasescan be configured to run on multiple servers, but it's often easier to vertically scale.

Vertical Scaling Limitations

- There are upper boundaries for the amount of memory and CPU that can allocated to a single instance, and there are connectivity ceilings for each underlying physical host
- Even if an instance has sufficient CPU and memory, some of those resources may sitidle at times, and you will continue pay for those unused resources

Scale Out (Horizontal Scaling)

Instead of resizing an application to a bigger server, scaling out splits the workload acrossmultiple servers that work in parallel.

Benefits of Scaling Out

- Applications that can sit within a single machine—like many websites—are well-suited to horizontal scaling because there is little need to coordinate tasks between servers. For example, a retail website might have peak periods, such as around the end-of-year holidays. During those times, additional servers can be easily committedto handle the additional traffic.
- Many front-end applications and microservices can leverage horizontal scaling. Horizontally-scaled applications can adjust the number of servers in use according tothe workload demand patterns.

Horizontal Scaling Limitations

- The main limitation of horizontal scaling is that it often requires the application to be architected with scale out in mind in order to support the distribution of workloads across multiple servers.

What is Cloud Autoscaling?

Autoscaling (sometimes spelled auto scaling or auto-scaling) is the process of automatically increasing or decreasing the computational resources delivered to a cloud workload based on need. The primary benefit of autoscaling, when configured and managed properly, is that your workload gets exactly the cloud computational resources it requires (and no more or less) at any given time. You pay only for the server resources you need, when you need them.

Load balancing in Cloud Computing

Cloud load balancing is defined as the method of splitting workloads and computing properties in a cloud computing. It enables enterprise to manage workload demands or application demands by distributing resources among numerous computers, networks orservers. Cloud load balancing includes holding the circulation of workload traffic and demands that exist over the Internet.

As the traffic on the internet growing rapidly, which is about 100% annually of the present traffic. Hence, the workload on the server growing so fast which leads to the overloading of servers mainly for popular web server. There are two elementary solutions to overcome theproblem of overloading on the servers-

- First is a single-server solution in which the server is upgraded to a higherperformance server. However, the new server may also be overloaded soon, demanding another upgrade. Moreover, the upgrading process is arduous and expensive.
- Second is a multiple-server solution in which a scalable service system on a cluster of servers is built. That's why it is more cost effective as well as more scalableto build a server cluster system for network services.

Load balancing is beneficial with almost any type of service, like HTTP, SMTP, DNS, FTP, and POP/IMAP. It also rises reliability through redundancy. The balancing service is provided by a dedicated hardware device or program. Cloud-based servers farms can attain more precise scalability and availability using server load balancing.

Load balancing solutions can be categorized into two types –

- Software-based load balancers: Software-based load balancers run on standardhardware (desktop, PCs) and standard operating systems.
- Hardware-based load balancer: Hardware-based load balancers are dedicated boxeswhich include Application Specific Integrated Circuits (ASICs) adapted for a particular use. ASICs allows high speed promoting of network traffic and are frequently used for transport-level load balancing because hardware-based load balancing is faster in comparison to software solution.

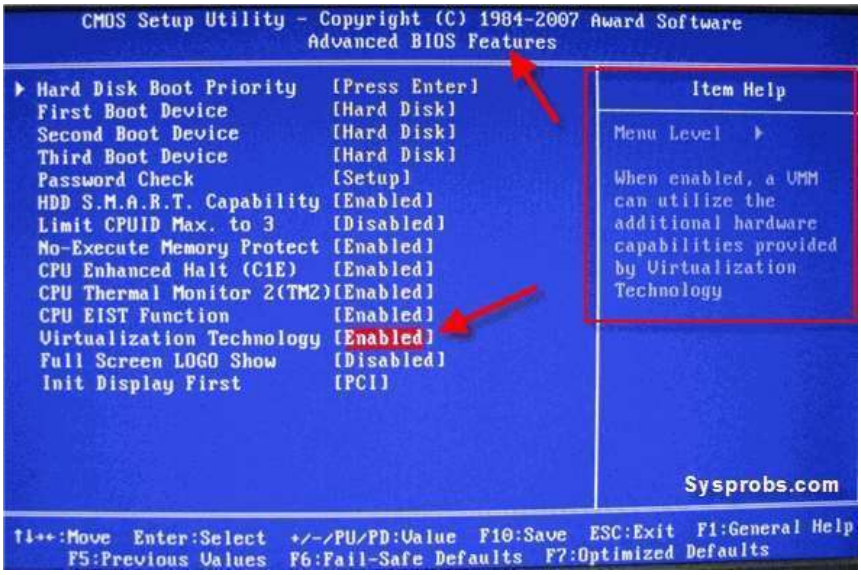
Major Examples of Load Balancers –

- Direct Routing Requesting Dispatching Technique: This approach of request dispatching is like to the one implemented in IBM's Net Dispatcher. A real server and load balancer share the virtual IP address. In this, load balancer takes an interface constructed with the virtual IP address that accepts request packets and it directly routes the packet to the selected servers.
- Dispatcher-Based Load Balancing Cluster: A dispatcher does smart load balancing byutilizing server availability, workload, capability and other user-defined criteria to regulate where to send a TCP/IP request. The dispatcher module of a load balancer can split HTTP requests among various nodes in a cluster. The dispatcher splits the load among many servers in a cluster so the services of various nodes seem like a virtual service on an only IP address; consumers interrelate as if it were a solo server,without having an information about the back-end infrastructure.
- Linux Virtual Load Balancer: It is an opensource enhanced load balancing solution used to build extremely scalable and extremely available network services such as HTTP, POP3, FTP, SMTP, media and caching and Voice Over Internet Protocol (VoIP). It is simple and powerful product made for load balancing and fail-over. The load balancer itself is the primary entry point of server cluster systems and can execute Internet Protocol Virtual Server (IPVS), which implements transport-layer load balancing in the Linux kernel also known as Layer-4 switching.

Output:

1. Install XenServer: Insert Bootable CD into CDROM or Bootable Pen drive and make first boot device from BIOS.

2. Press F2 for advance options and Make Enabled Virtualization Technology fromBIOS

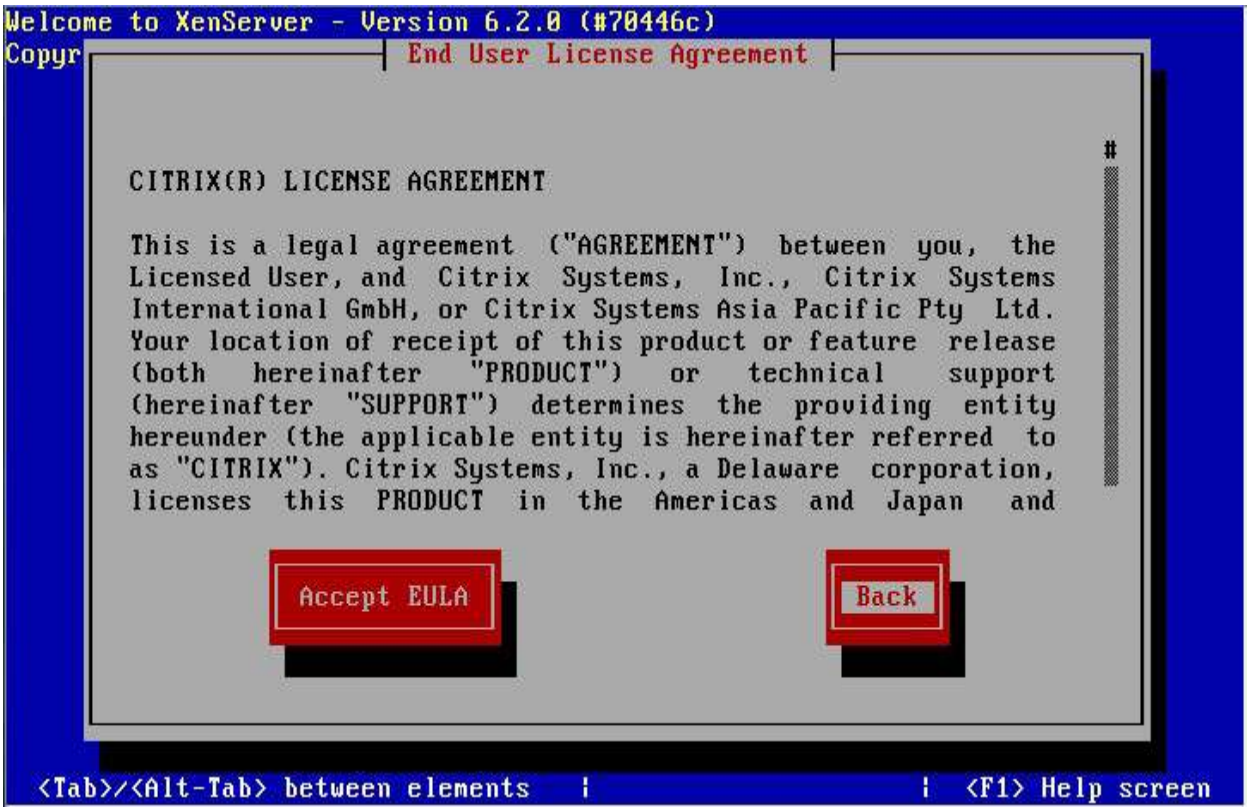


3. Save and reboot

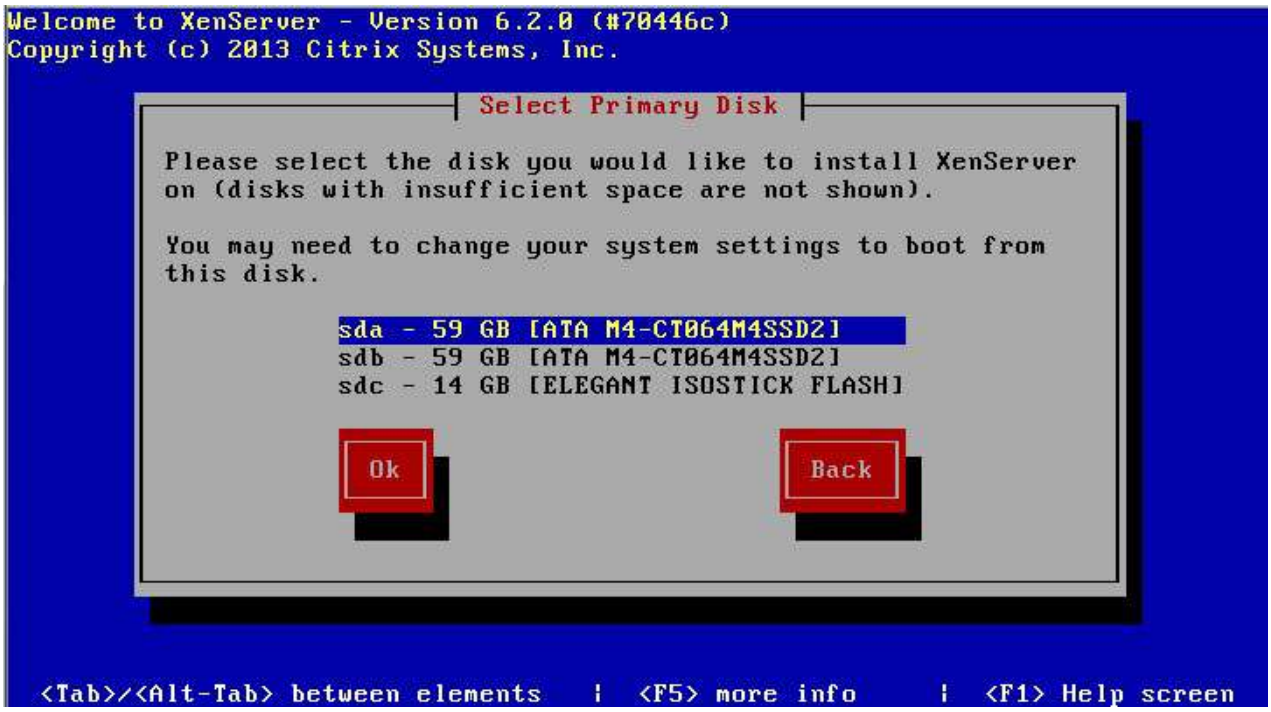
4. We begin by choosing the keymap i.e. Keyboard Layout

5. Press Enter to load Device Drivers

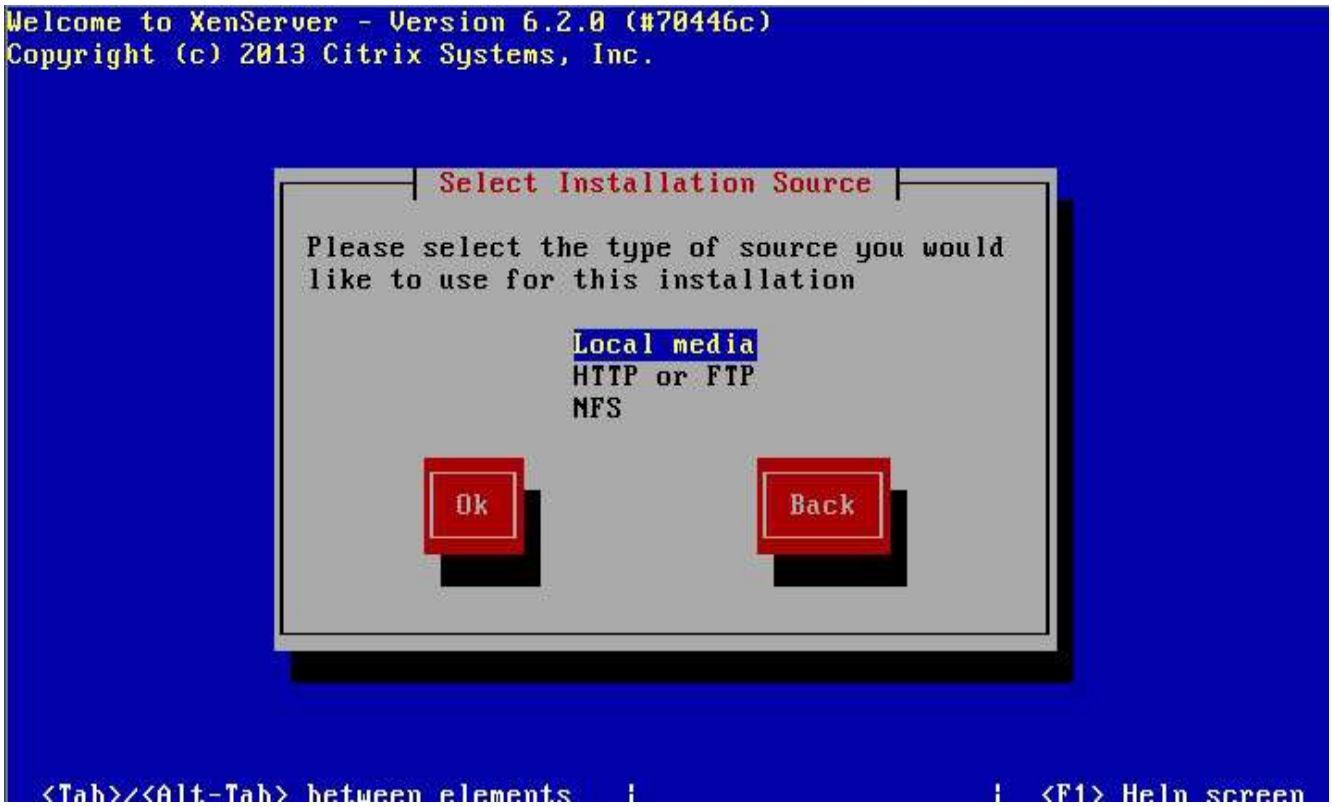
6. Press Enter to accept End User License Agreement



7. Select Appropriate disk on which you want to install Xenserver

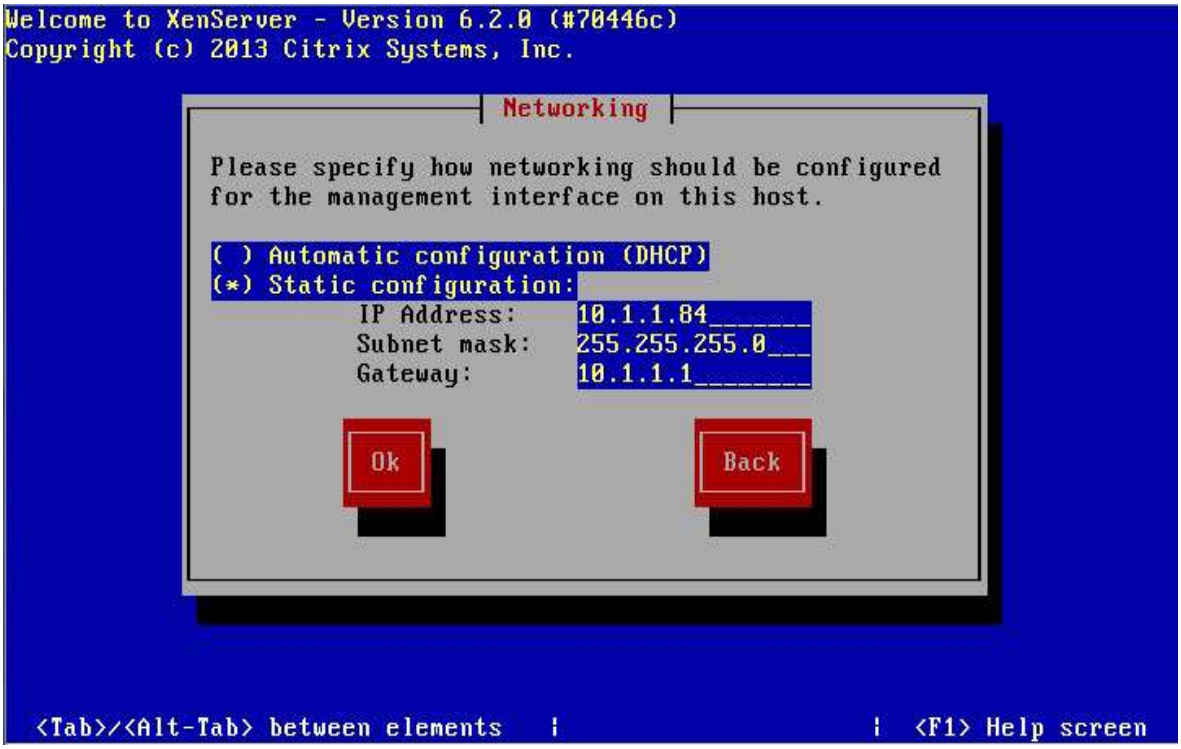


8. Select appropriate Installation Media



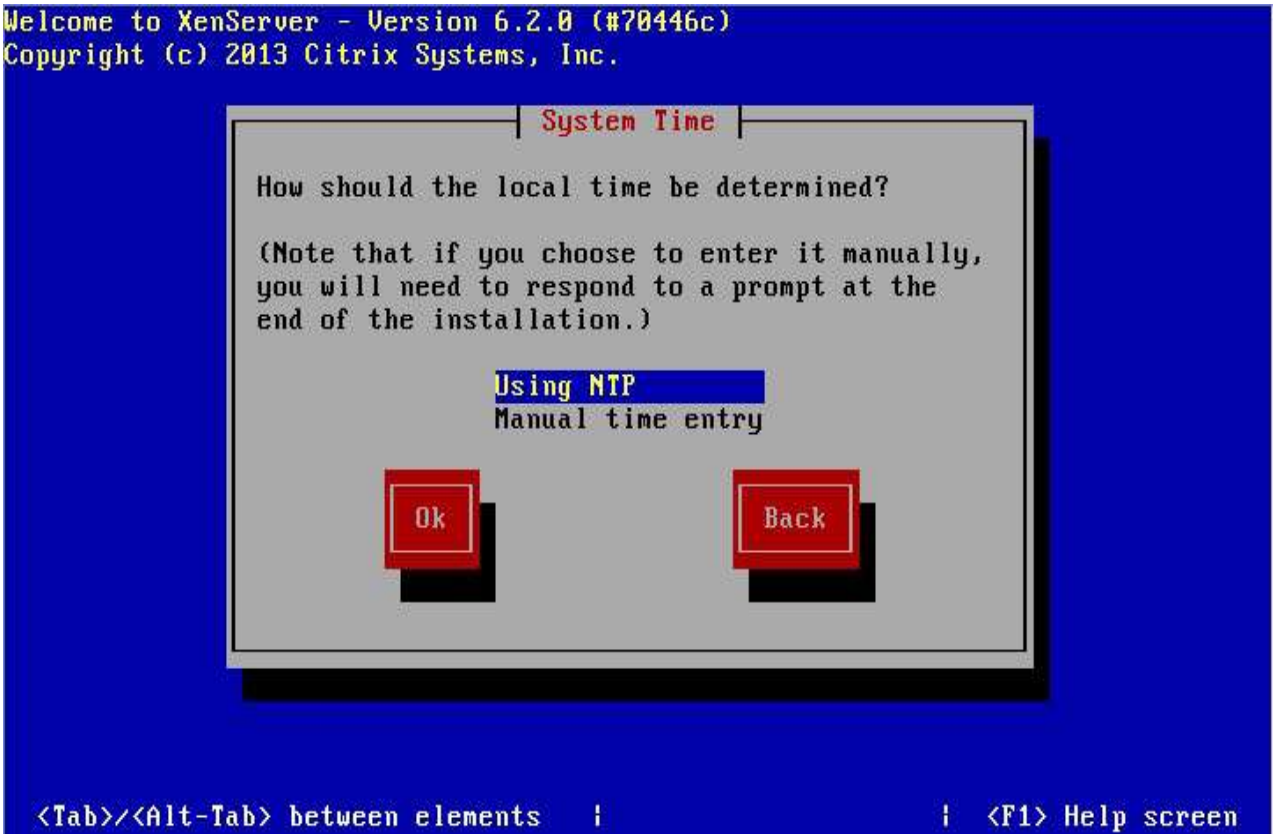
9. Press yes to Select additional packages for installation , Otherwise Press No

10. The motherboard we are installing have two Ethernet ports, both of which are supported by XenServer Choose the one you wish to use for the management network
– you can change this later. Here we get to choose the networking settings for ourmanagement network.

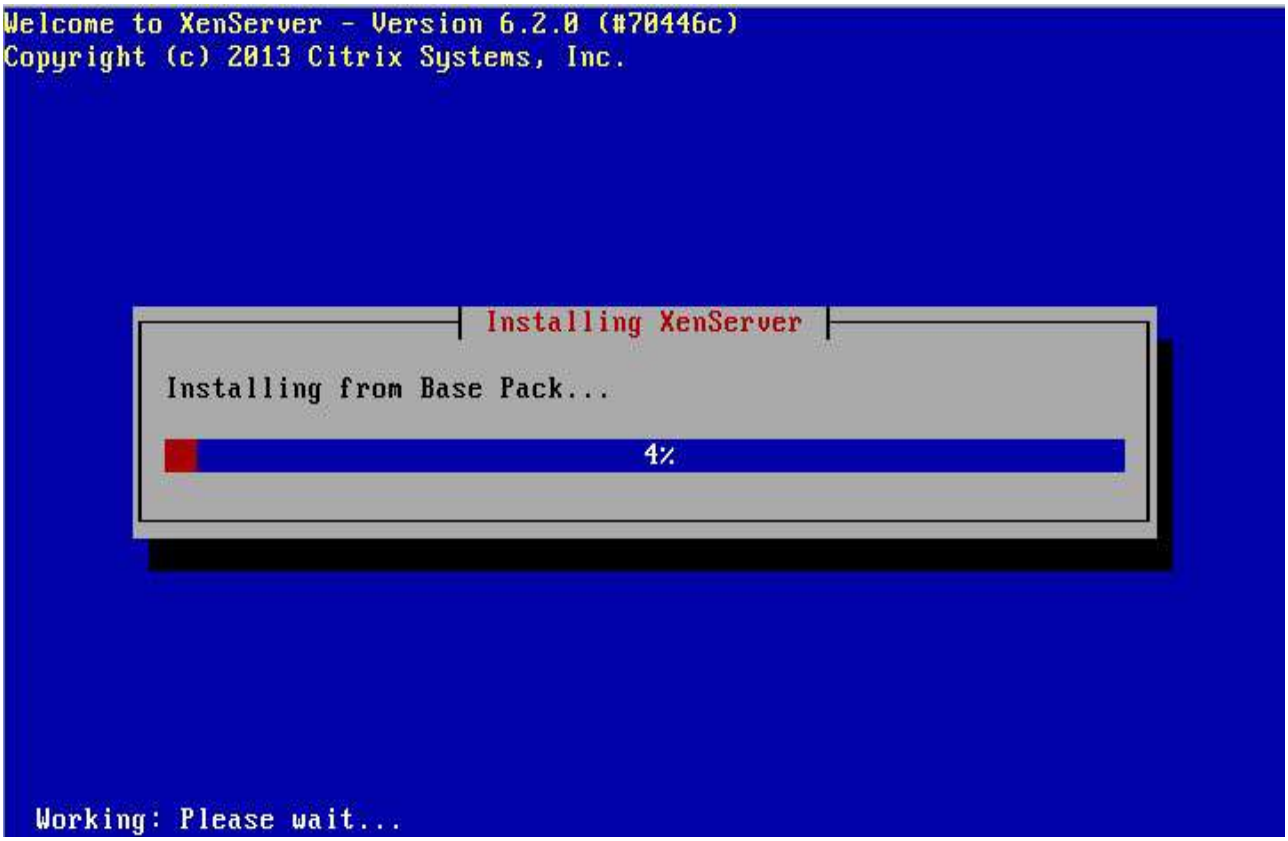


11. Select Time zone

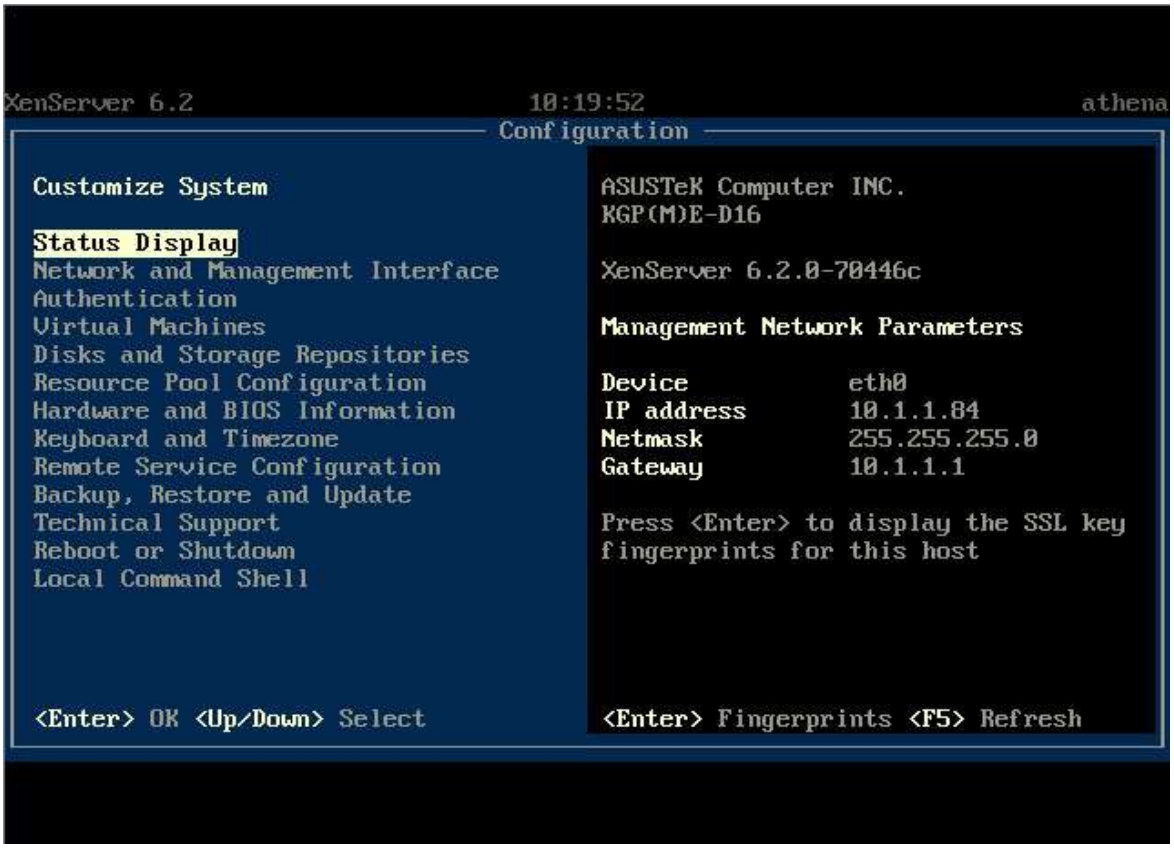
12. Specify NTP Server address and start installation to set your time



13. Press Enter to start installation of XenServer

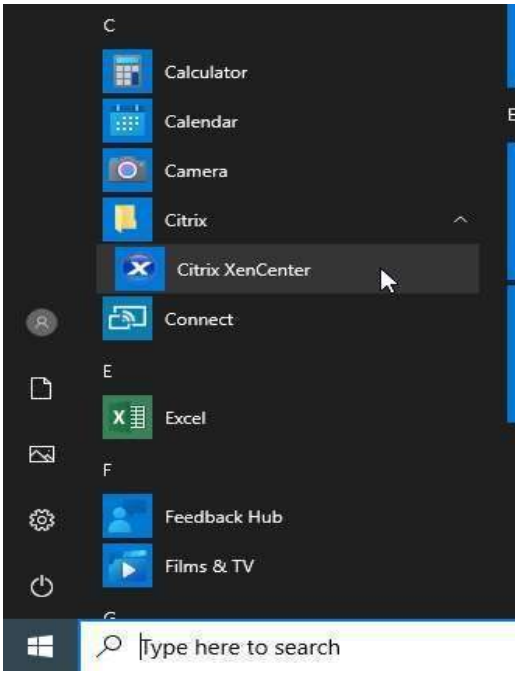


14. This is the loading screen and console for installed new XenServer



Step 2: Connect XenCenter to XenServer

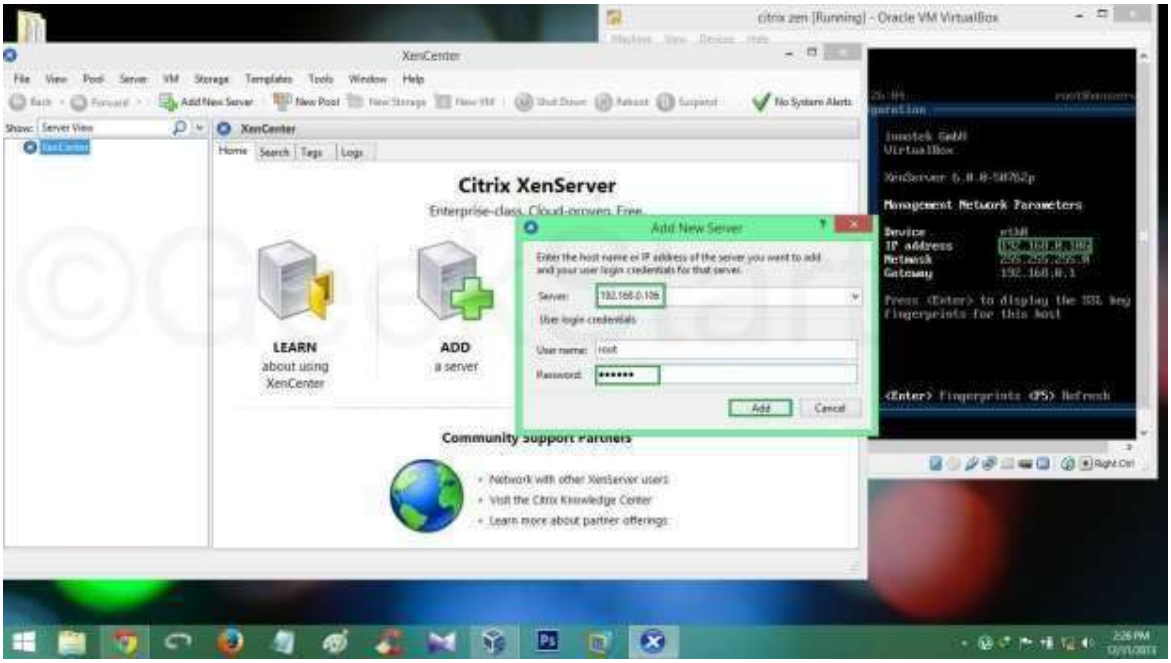
1. Download the XenCenter a management utility from XenServer IP address as a URL on browser. Install XenCenter and open it from start Menu of Windows on Machine 2



2. Once you click on Citrix XenCenter ..It Looks like as below:

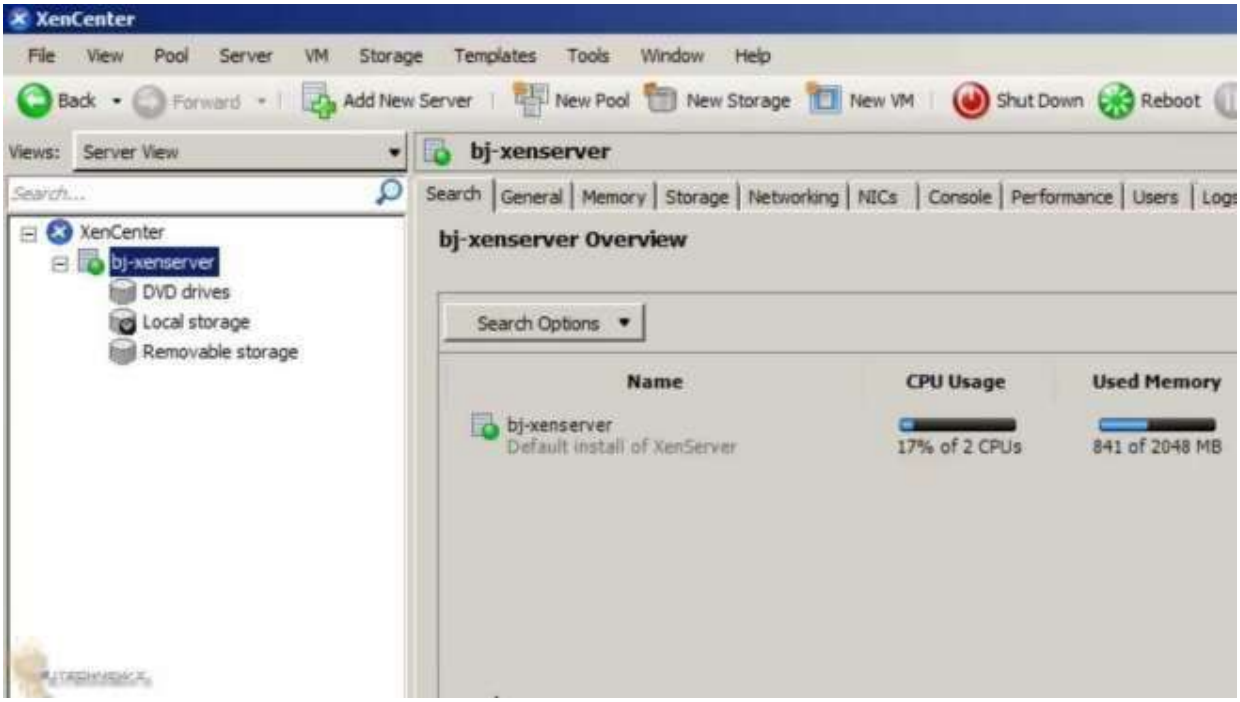
3. To Connect to the XenServer host which is configured earlier, click ADD a Server

4. Enter IP address of XenServer and Enter User login credentials and click Add



5. Once you clicked on Add , it will ask to configure a master password for all the XenServers.

6. Xenserver is added now to XenCenter



Step 3: Create Local ISO Storage .

Now before creating Virtual Machine we have to Create storage repository which is nothing but shared directory on XenCenter which holds all iso files and which is required to install Operating system on XenServer .

Using Local command Shell

i. First View Xen Directory Structrure # df – h
ii. Now make folder to store ISO images # mkdir /var/ISO_images
iii. Using wget command download iso files . for example:
#wget http:// releases.ubuntu.com/16.04/ubuntu-16.04.6-desktop-

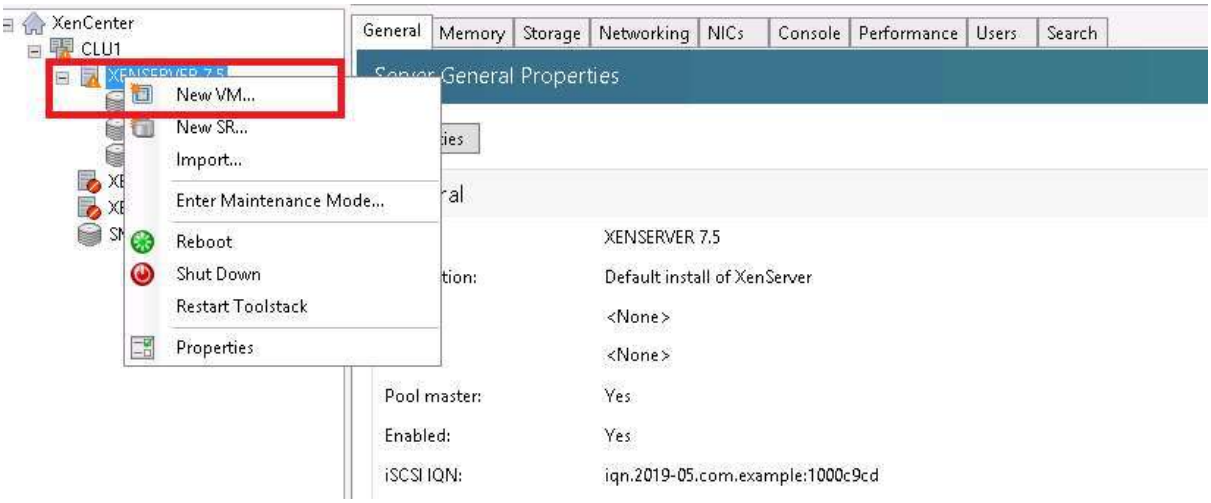
amd64.isoOR

USING PENDRIVE (need to Mount Pendrive)

a) #mkdir /mnt/myusb
b) #ls – l /dev/sdb1 to check drive for removable disk.
c) #mount –t vfat –o rw,users /dev/sdb1 /mnt/ myusb
d) #cd / mnt/myusb
e) # ls (content of pendrive)
f) # cp Ubuntu -16.04.5-desktop-i386.iso /var/ISO_images
g) Reboot or shutdown Xenserver from Xencenter or from console of Xenserver

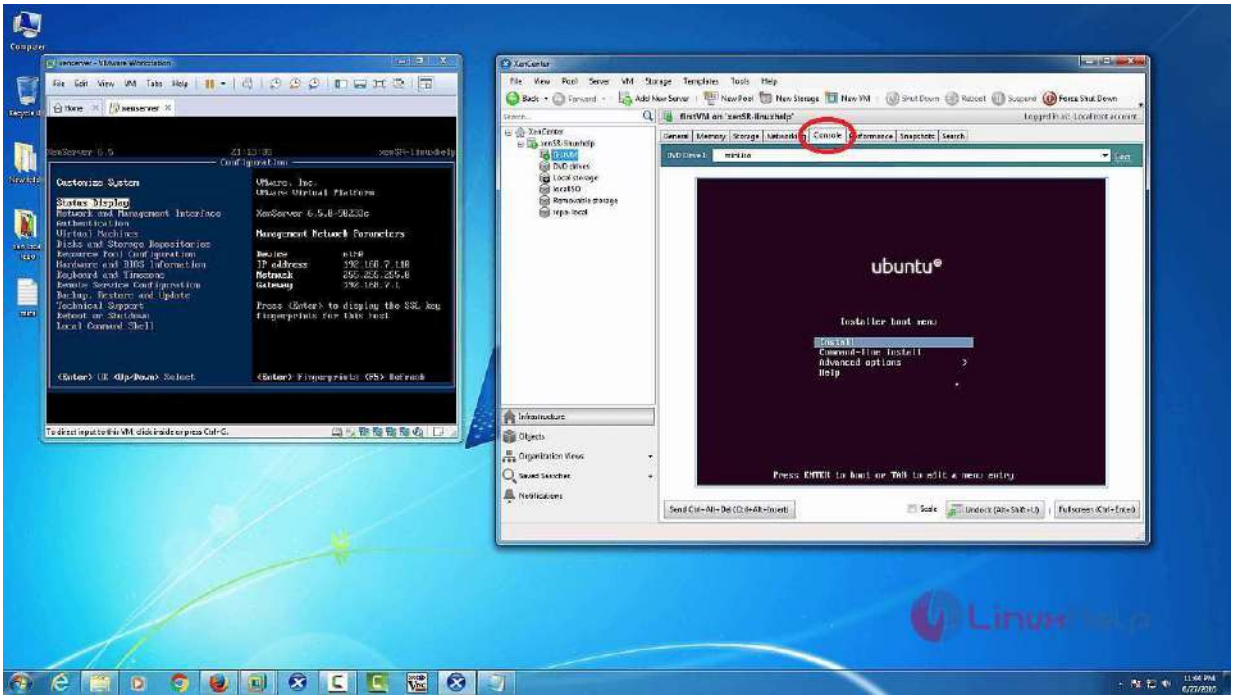
Step 4: Installation of Virtual Machine from Xencenter

i. Right click on Xenserver icon on Xen center and select New VM.



ii. Select VM template

iii. Name the virtual machine
iv. Locate the operating system installation media To select appropriate OS iso file.
v. Allocate CPU and memory to VM
vi. Select networking
vii. finish



Conclusion: Thus we have successfully created a virtual machine in xenserver usingXenCenter tool.

Experiment :- 4

Aim:- To study and implement infrastructure as a service using AWS.

Theory:-

EC2= Amazon Elastic Compute Cloud, a VM service in AWS for running customizable virtual computers.

Features= On-demand computing, scalable capacity, no upfront hardware investments, easy app deploy. and scaling based on traffic site.

AMI= Amazon Machine Image required to launch instances. Contains EBS snapshots, root template, launch permissions, etc.

Storage= AMI's are stored in Amazon S3; identified by unique ID's. Can be created from scratch or bundled from existing EC2 instances.

Types of EC2 computing instances:

General Purpose: Balanced compute, memory and networking resources for diverse workloads like web servers.

Compute Optimized: High-performance processors for compute bound tasks like batch processing.

Memory optimized: Fast performance for memory intensive applications like databases.

Accelerated Computing: Hardware accelerators for tasks like ml and graphics processing.

Storage optimised: High, sequential read/write access for large data sets.

Elastic IP addresses: These are static IPv4 addresses for dynamic cloud computing. You can also use them in DNS records for domain pointing. They are public IPv4 addresses reachable from net.

Remote Desktop ~~Protocol~~ Protocol (RDP): It allows remote usage of desktop computers. It's commonly used, especially with windows. Users can access, edit files, run apps. Remote desktop access is more about accessing physical desktops remotely.

Implementation: Open EC2 page on amazon, click launch instance, select Ubuntu server 20.0.4 and select Instance type, configure Installation Details, add a new volume and tag. Create a new Security Group. Create a new Key Pair and Download it. Launch and click connect.

Conclusion:- Hence, an EC2 instance was setup and run on Amazon Web Services.

A+
3/24

Arnav Malvia
2103109
C23

Experiment 4

Aim: To study and Implement Infrastructure as a Service using AWS.

Theory:

EC2

EC2 stands for Amazon Elastic Compute Cloud. Amazon Ec2 is a basic virtual machine withcustomizable hardware components and an OS. The system allows you to run various virtualcomputers and manage the same with a single hardware.

Elastic Compute Cloud is the highly used and primary service system in the massive AWS ecosystem.The cloud system provides multiple features, for instance, it facilitates computing on-demand and scales the Computing capacity in the Amazon cloud system.

Amazon instances free you from making additional up-front investments for hardware. Also, no extra baggage of maintaining rented hardware. The all-in-one virtual hardware is easy to use and lets you create and run applications at a higher speed. Adapting Elastic Computing Cloud in AWS allows you to launch multiple virtual servers. It also provides the control to scaling up or scaling down in correspondence with the rate of the site traffic.

Also, the system works with multi-volume workloads and is capable of provisioning and de-provisioning resources with respect to the on-going demand. This adaptable behavior of the systemconstitutes the word Elastic in Elastic Computing Cloud.

AMI

AMI stands for Amazon Machine Image. The information required to launch an instance is provided by the Amazon Machine Image . AMI must be specified when an instance is launched. One AMI canbe chosen to launch multiple instances when the configuration required is the same for all the instances. However, multiple AMIs can be used to launch instances with different configurations.

An AMI consists of the following:

One or more EBS snapshots
For instance-store-backed AMIs, a template for the root volume of the instance is provided. Thisincludes the operating system, an application server, and applications.
Launch permissions that control which AWS accounts can use the AMI to launch instances.
A block device mapping that specifies the volumes to attach to the instance when it's launched
AMIs are stored in the Amazon S3 and identified by a unique identifier in the form of AMI-xxxxxx and a manifest XML file. They can be created from scratch or bundled from existing EC2 instances.Once an AMI is created, it is stored in an S3 bucket and the user can decide whether to make it available to others users or keep it for personal use.

You can also associate a product code with a given AMI, thus, allowing the owner of the AMI to get revenue every time this AMI is used to create EC2 instances.

Types of EC2 computing instances

Amazon EC2 provides a wide selection of instance types optimized to fit different use cases. Instance types comprise varying combinations of CPU, memory, storage, and networking capacity and give you the flexibility to choose the appropriate mix of resources for your applications. Each instance typeincludes one or more instance sizes, allowing you to scale your resources to the requirements of your target workload.

General Purpose

General purpose instances provide a balance of compute, memory and networking resources, and canbe used for a variety of diverse workloads. These instances are ideal for applications that use these resources in equal proportions such as web servers and code repositories.

Use Cases:

Developing, building, testing, and signing iOS, iPadOS, macOS, WatchOS, and tvOS applications onthe Xcode IDE

Compute Optimized

Compute Optimized instances are ideal for compute bound applications that benefit from high performance processors. Instances belonging to this family are well suited for batch processing workloads, media transcoding, high performance web servers, high performance computing (HPC), scientific modeling, dedicated gaming servers and ad server engines, machine learning inference andother compute intensive applications.

Use Cases:

High performance computing (HPC), batch processing, ad serving, video encoding, gaming, scientificmodelling, distributed analytics, and CPU-based machine learning inference.

Memory Optimized

Memory optimized instances are designed to deliver fast performance for workloads that processlarge data sets in memory.

Use Cases:

Memory-intensive applications such as open-source databases, in-memory caches, and real time bigdata analytics

Accelerated Computing

Accelerated computing instances use hardware accelerators, or co-processors, to perform functions,such as floating point number calculations, graphics processing, or data pattern matching, more efficiently than is possible in software running on CPUs.

Use Cases:

Machine learning, high performance computing, computational fluid dynamics, computational finance, seismic analysis, speech recognition, autonomous vehicles, and drug discovery.

Storage Optimized

Storage optimized instances are designed for workloads that require high, sequential read and write access to very large data sets on local storage. They are optimized to deliver tens of thousands of low-latency, random I/O operations per second (IOPS) to applications.

Use Cases:

These instances maximize the number of transactions processed per second (TPS) for I/O intensive and business-critical workloads which have medium size data sets and can benefit from high computeperformance and high network throughput such as relational databases (MySQL, MariaDB, and

PostgreSQL), and NoSQL databases (KeyDB, ScyllaDB, and Cassandra). They are also an ideal fitfor workloads that require very fast access to medium size data sets on local storage such as searchengines and data analytics workloads.

Elastic IP addresses

An Elastic IP address is a static IPv4 address designed for dynamic cloud computing. An Elastic IP address is allocated to your AWS account, and is yours until you release it. By using an Elastic IP address, you can mask the failure of an instance or software by rapidly remapping the address to another instance in your account. Alternatively, you can specify the Elastic IP address in a DNS recordfor your domain, so that your domain points to your instance. For more information, see the documentation for your domain registrar, or Set up dynamic DNS on Your Amazon Linux instance.

An Elastic IP address is a public IPv4 address, which is reachable from the internet. If your instance does not have a public IPv4 address, you can associate an Elastic IP address with your instance to enable communication with the internet. For example, this allows you to connect to your instance from your local computer.

AWS currently does not support Elastic IP addresses for IPv6.

Remote Desktop Protocol (RDP)

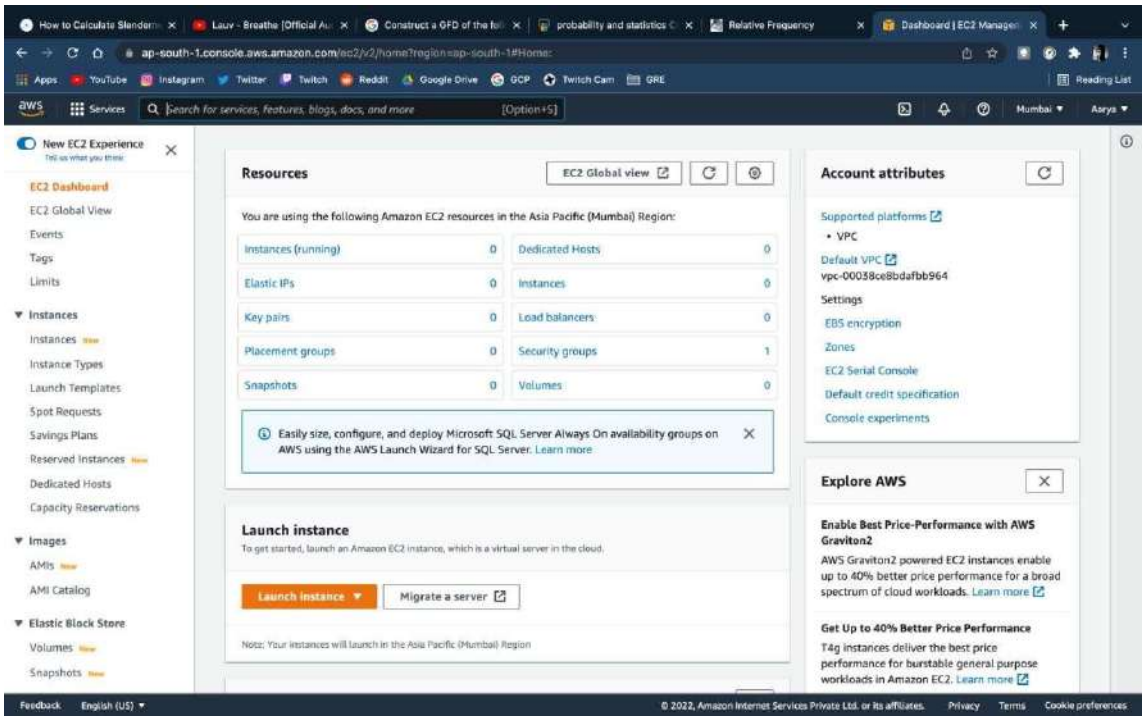
The Remote Desktop Protocol (RDP) is a protocol, or technical standard, for using a desktop computerremotely. Remote desktop software can use several different protocols, including RDP, Independent Computing Architecture (ICA), and virtual network computing (VNC), but RDP is the most commonlyused protocol. RDP was initially released by Microsoft and is available for most Windows operating systems, but it can be used with Mac operating systems too.

Remote desktop is the ability to connect with and use a faraway desktop computer from a separate computer. Remote desktop users can access their desktop, open and edit files, and use applications as ifthey were actually sitting at their desktop computer. Employees often use remote desktop software to access their work computers when they are traveling or working from home.

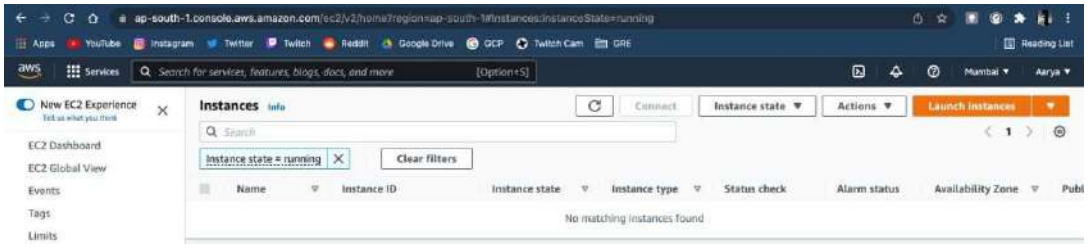
Remote desktop access is very different from cloud computing, even though both allow employees to work remotely. In cloud computing, users access files and applications that are stored in the cloud — specifically, in cloud servers. In contrast, when using remote desktop software, users are actually accessing their physical desktop computer, and can only use files and applications saved locally on thatdesktop. Cloud computing is sometimes easier to use and more efficient to implement for remote workforces, but many companies have not migrated to the cloud, or cannot for security or regulatory reasons.

Implementation:

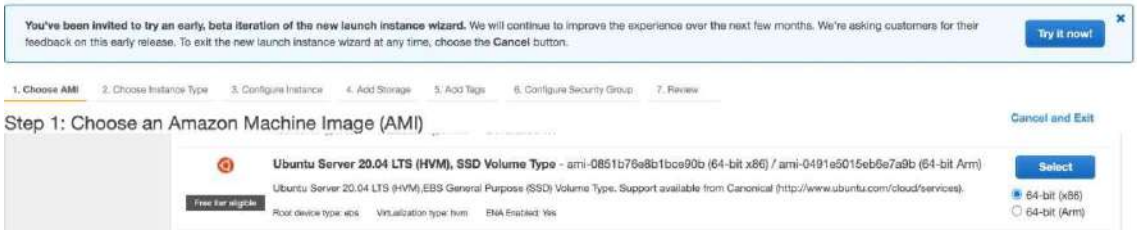
1. Open the EC2 page on amazon.



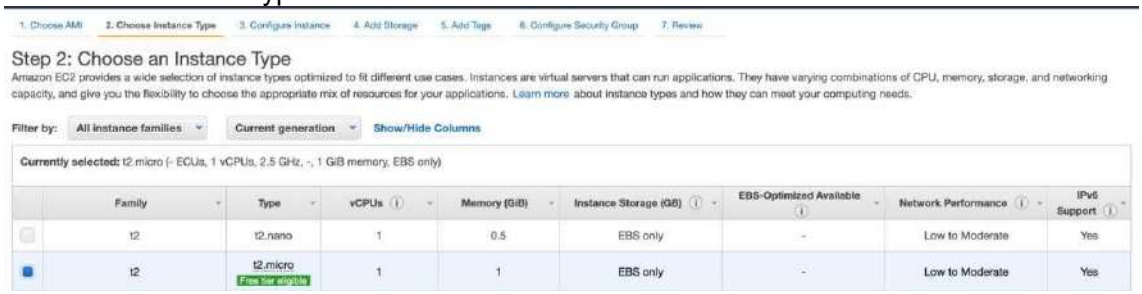
2. Click the launch instance



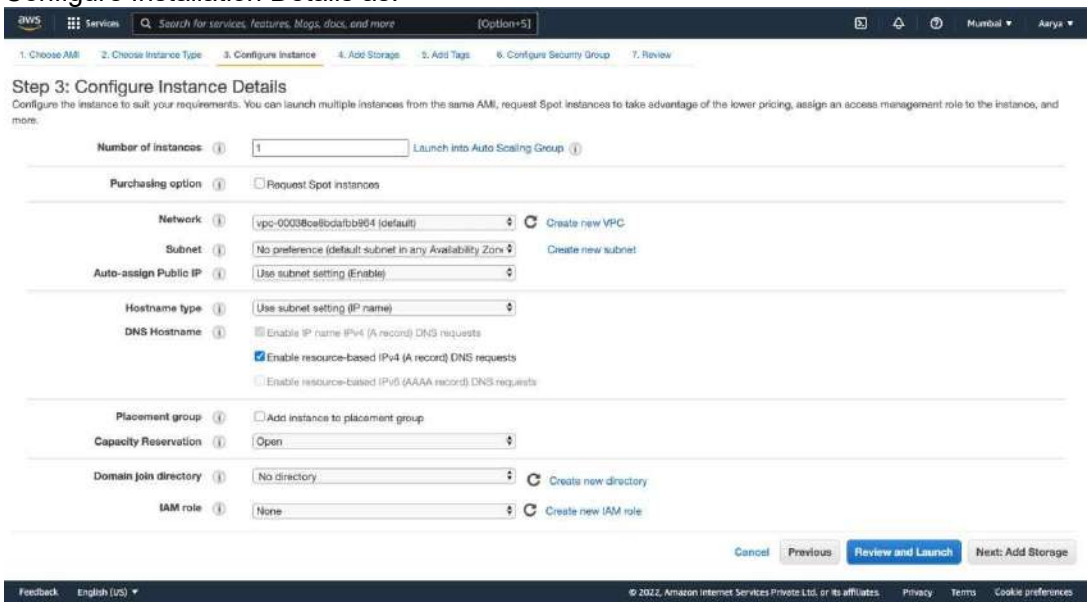
3. Select Ubuntu Server 20.0.4



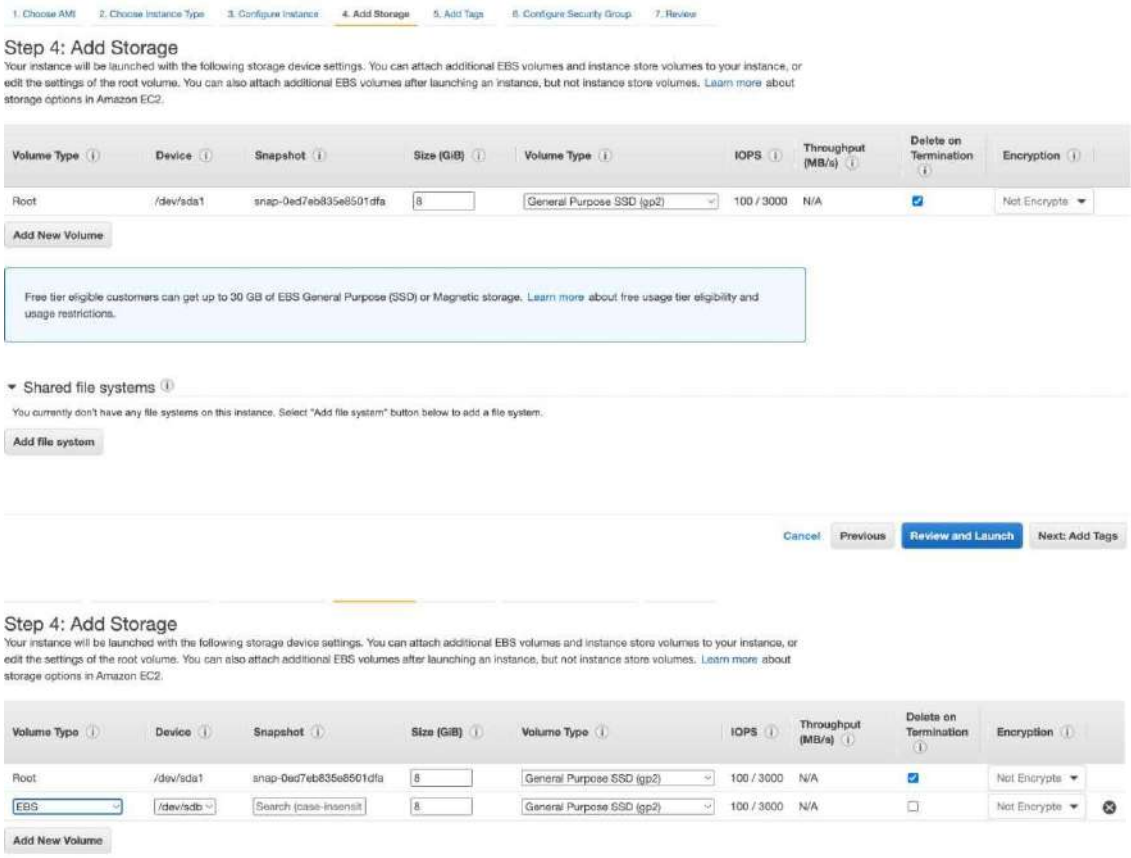
4. Select the Instance Type.



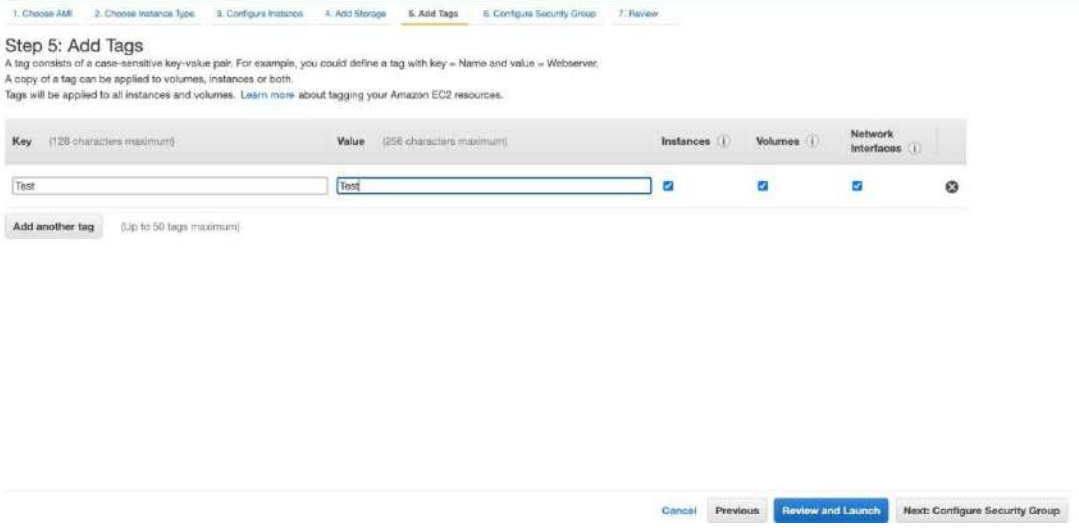
5. Configure Installation Details as:



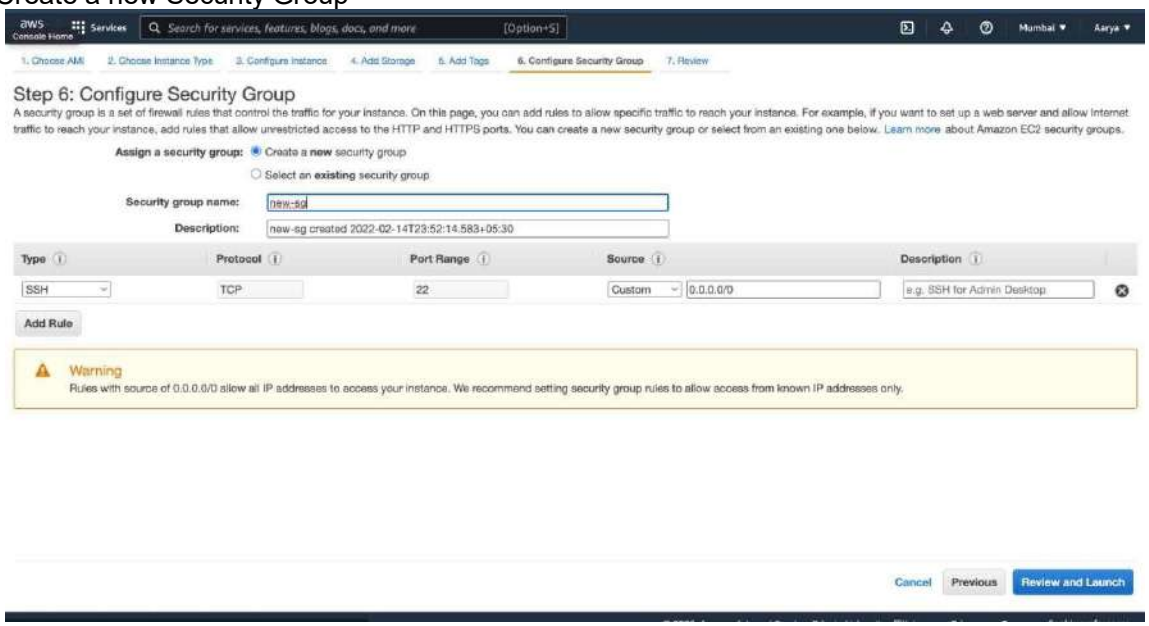
6. Add a new volume



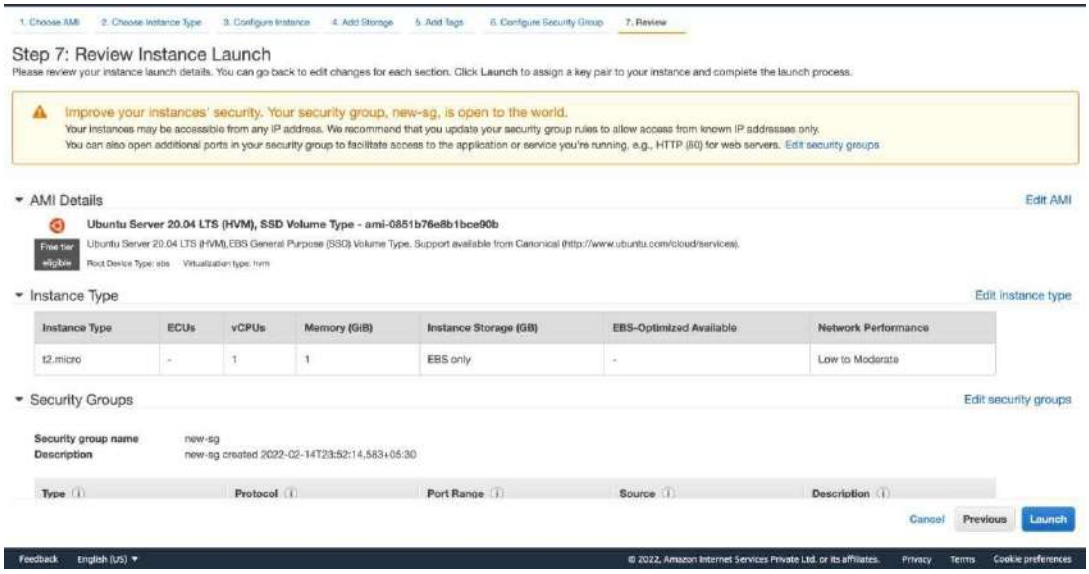
7. Add a Tag



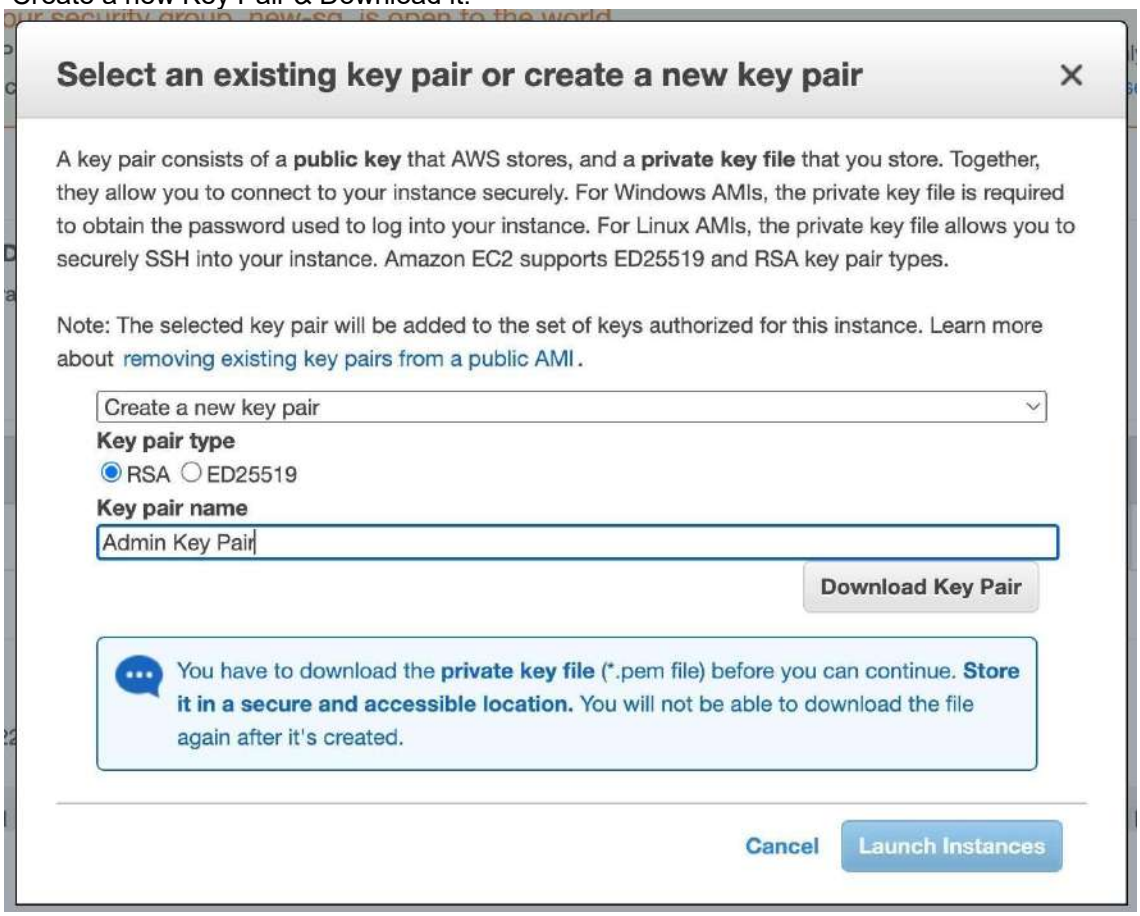
8. Create a new Security Group



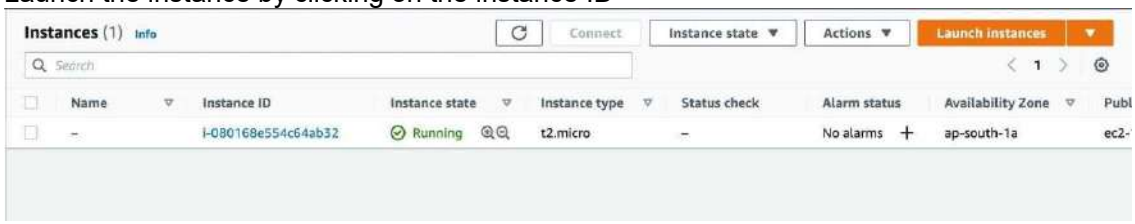
9. Review all details and launch



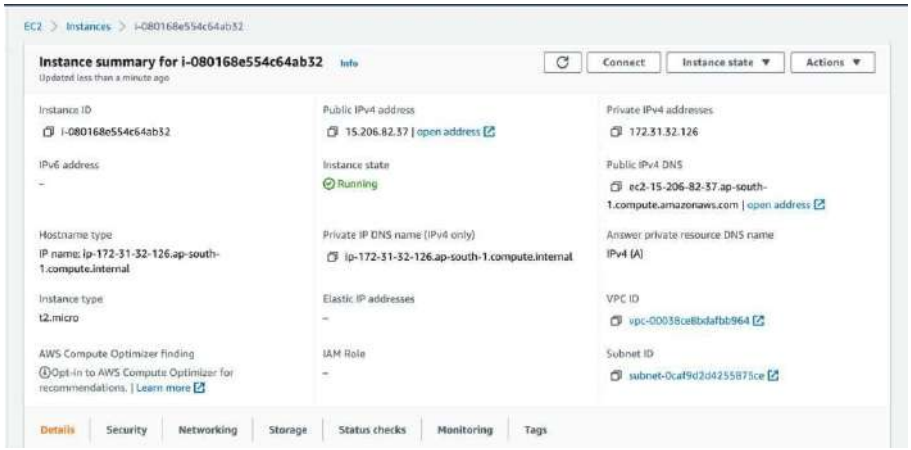
10. Create a new Key Pair & Download it.



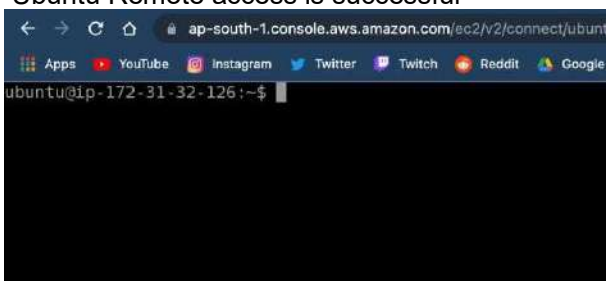
11. Launch the instance by clicking on the instance ID



Then click connect.



12. Ubuntu Remote access is successful



Testing Linux commands:

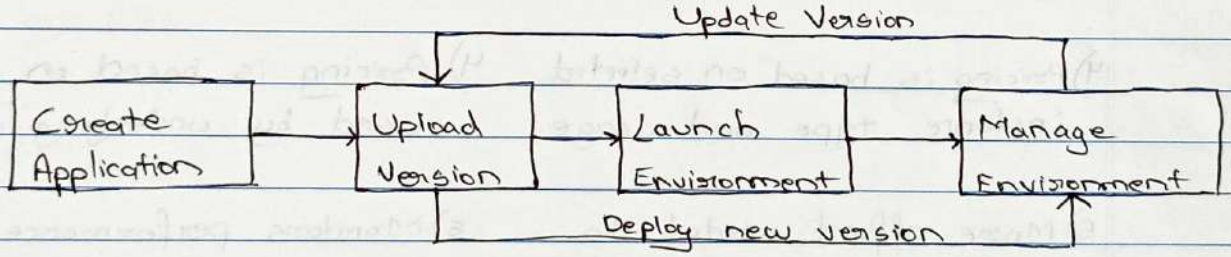
Conclusion: Hence, an EC2 Instance was setup and run on Amazon Web Services.

Experiment :- 5

Aim :- To study and implement platform as a service using AWS Elastic Beanstalk service.

Theory :-

AWS Elastic beanstalk service - With elastic beanstalk one can quickly deploy and control apps in AWS cloud without having to learn about infrastructure that runs those apps. Elastic Beanstalk reduces restricting choice or ctrl. The user simply uploads the application and it automatically handles details of capacity provisions, load balancing, scaling and app health monitoring.



Elastic beanstalk supports apps developed in Go, Java, .NET, Node.js, PHP, Python and Ruby. When you deploy your application, it builds selected supported platform version and provisions for one or more AWS resources such as Amazon EC2 instances, to run your apps.

THADOMAL SHAHANI
TSEC
ENGINEERING COLLEGE

Compare EC2 and Elastic Beanstalk.

EC2	Elastic Beanstalk
1) It provides raw virtual machines in cloud. Users have full control over the OS and networking.	1) It abstracts away the solution infrastructure details and provides platform as a service solution.
2) Users are responsible for deploying application managing software updates.	2) EBS takes care of the underlying infrastructure.
3) Supports a wide range of OS and services.	3) Supports specific programming languages and frameworks.
4) Pricing is based on selected instance type and usage.	4) Pricing is based on resources used by underlying infra.
5) More effort needed to handle increased traffic.	5) Monitors performance and provides easier scalability.

Elastic Load Balancing:-
ELB automatically distributes your incoming traffic across multiple targets such as EC2 instances, containers and IP addresses, in one or more ~~to~~ zones. It monitors health of its registered targets and routes traffic only to healthy ~~foor~~ targets. Elastic Load Balancing scales your load balancer capacity automatically in response to changes in traffic.

THADOMAL SHAHANI
TSEC
ENGINEERING COLLEGE

Steps for deploying web apps / web services on AWS Elastic Beanstalk

1) Login to AWS and go to EBS.
2) Click on create Application.
3) Write app information : Name, tag, Platform, etc.
4) In app code, select sample application and then click on create application button.
5) Click on environments → check the health of environments till it becomes 'ok'.
6) Click the URL.

Conclusion :-

By performing the above experiment, we were able to understand EBS and its applications and deploy services on EBS.

A+

15/3/24.

Arnav Malvia
2103109
C23

Cloud Computing

Experiment 5

Aim: To study and Implement Platform as a Service using AWS Elastic BeanstalkService.

Theory:

AWS Elastic Beanstalk

AWS Elastic Beanstalk is an AWS managed service for web applications. Elastic beanstalk is a pre-configured EC2 server that can directly take up your application code and environment configurations and use it to automatically provision and deploy the required resources within AWS to run the web application. Unlike EC-2 which is Infrastructure as a service, Elastic beanstalk is a Platform As A Service (PAAS) as it allows users to directly use a pre-configured server for their application. Of course, you can deploy applications without ever having to use elastic beanstalk but that would mean having to choose the appropriate service from the vast array of services offered by AWS, manually provisioning these AWS resources, and stitching them up together to form a complete web application. Elastic beanstalk abstracts the underlying configuration work and allows you as a user to focus on more pressing matters.

This raises a concern that if elastic beanstalk configures most of the resources itself the and abstracts the underlying details. Can developers change the configuration if needed? The answer is Yes. Elastic Beanstalk is provided to make application deployment simpler but at no level will it restrict the developers from changing any configurations.

Features:

Application: Elastic Beanstalk directly takes in our project code. So the Elastic Beanstalk application is named the same as your project home directory.
Application Environments: Users may want their application to run on different environments like DEV, UAT, and PROD. You can create and configure different environments to run applications on different stages.
Environment Health: One of the most lucrative features about running applications on AWS or most of the other cloud platforms is the automated health checks. AWS runs automatic health checks on all EC-2 deployments (Elastic Beanstalk is a managed EC-2 service) which can be monitored from the AWS console. For example, in case of web

applications, AWS will regularly, as scheduled by the developers, ping the application tocheck if the response is status code 200 and the application is running as expected.

Health check responses:
Red: Application failed all health tests.
Yellow: The application failed some of the health tests.
Grey: The application is updating.
Green: Application passed health check successfully.
Isolated: All environments within a single application are isolated from each other (independent of each others' running status). Needless to say, two different applicationsare also isolated.
Scalability: Using Auto-Scaling within Elastic beanstalk makes the application dynamically scalable.
Elastic Load Balancing: All the web requests to the application are not directly relayed to application instances. They first hit the Elastic Load Balancer (ELB), which, as the name suggests, balances the load across all the application instances.
Language support: Elastic Beanstalk supports the applications developed with Java, .NET, PHP, Node.js, Python, Ruby, Go, and Docker on familiar servers such as Apache,Nginx, Passenger, and IIS.
Pricing: There is no extra charge for using Elastic Beanstalk. Users ar only required topay for the services and resources provisioned by Elastic Beanstalk Service.
Automatic Provisioning: Elastic Beanstalk takes away the burden of choosing the right services and configuring their security groups to work together.
Impossible to Outgrow: AWS claims that since Elastic Beanstalk uses Auto Scalingfeature it can, in theory, handle any amount of internet traffic.

Elastic Beanstalk supported Language/Frameworks
AWS Elastic Beanstalk provides managed platforms that support running web applications developed for specific programming languages, frameworks, and web containers. Elastic Beanstalk offers one or more platform versions for each platform. When you create an environment and choose a platform, Elastic Beanstalk provisions the resources that your application needs, including one or more Amazon Elastic Compute Cloud (Amazon EC2) instances. The software stack running on the Amazon EC2 instances depends on the platform version you chose.

For more information about platforms, see AWS Elastic Beanstalk Platforms in the AWSElastic Beanstalk Developer Guide. Detailed release notes are available for recent releases at AWS Elastic Beanstalk Release Notes.

The following sections provide information about all current platform versions. All current Linux-based platform versions run on Amazon Linux 2018.03 (64-bit). For lists of historical platform versions and the date ranges they were current, see Platform history.

Elastic Beanstalk has scheduled some platform versions for retirement, because some of their components are reaching their End of Life (EOL). These platform versions remain available until the published retirement date of their retiring components. For a list of component retirement dates, see AWS Elastic Beanstalk Platform Support Policy in the AWS Elastic Beanstalk Developer Guide. For a list of platform versions scheduled for retirement, see Elastic Beanstalk platform versions scheduled for retirement.

Supported Languages / Frameworks:

- Docker
- Go
- Java SE
- Tomcat
- .NET Core on Linux
- .NET on Windows Server
- Node.js
- PHP
- Python
- Ruby
- Elastic Beanstalk platform versions scheduled for retirement
- Elastic Beanstalk platform versions in public beta

Docker

Docker is a container platform that allows you to define your own software stack and store it in an image that can be downloaded from a remote repository. Use the Docker platform if you only need to run a single Docker container on each instance in your environment. The Docker platform includes an nginx proxy server.

See Deploying Elastic Beanstalk Applications from Docker Containers in the *AWS Elastic Beanstalk Developer Guide* for more information about the Docker platform.

Platform Version and *Solution Stack Name*	AMI	Docker	Docker Compose	Proxy Server
Docker AL2 version 3.4.12 *64bit Amazon Linux 2 v3.4.12 running Docker*	2.0.20220207	20.10.7-5	1.29.2	nginx 1.20.0

For information about platform versions scheduled for retirement as published in Platform Support Policy, see Single Container Docker on the *Retiring Platform Versions* page. For information about previous platform versions, see Docker platform history.

Go

Elastic Beanstalk supports the following Go platform versions.

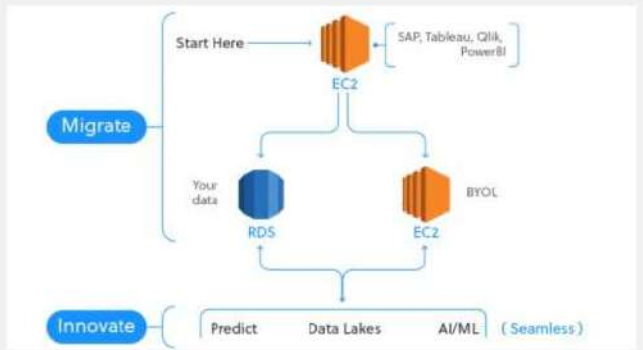
Platform Version and *Solution Stack Name*	AMI	Language	AWS X-Ray	Proxy Server
Go 1 AL2 version 3.4.6 *64bit Amazon Linux 2 v3.4.6 running Go 1*	2.0.20220207	Go 1.17.7	3.2.0	nginx 1.20.0

For information about platform versions scheduled for retirement as published in Platform Support Policy, see Go on the *Retiring Platform Versions* page. For information about previous platform versions, see Go platform history.

EC2 Compared to Elastic Beanstalk

Differences between the AWS Services of Elastic Beanstalk and EC2:

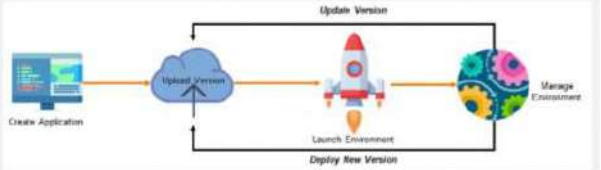
How does the EC2 actually work Vs Elastic Beanstalk?



Elastic Beanstalk Vs EC2 – EC2 Vs Elastic Beanstalk

EC2 allows users to create and launch servers in the cloud. The EC2 instances offer a total web services API for accessing the many different services that are available on AWS platform.

How does the Elastic Beanstalk actually work Vs EC2?

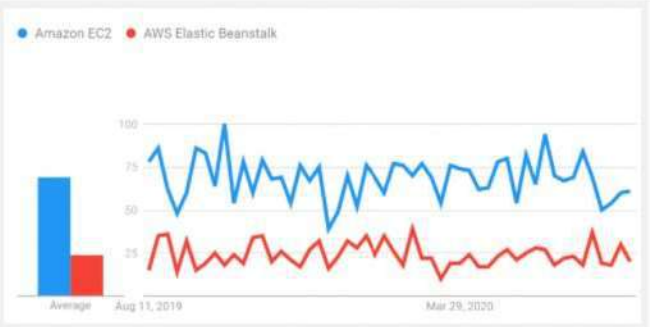


Elastic Beanstalk Vs EC2 – Elastic Beanstalk Vs EC2

- The Elastic Beanstalk service from AWS provides developers with a platform for the deployment of apps on the AWS cloud, as well as gets them connected to other AWS services.
- This means that Elastic Beanstalk cannot be regarded as something which could be discovered as time goes by, but on the contrary it should be studied in terms of its underlying AWS services, and the way they work in concert through the help of Elastic Beanstalk.
- Elastic Beanstalk connects services such as S3, EC2 and Auto Scaling in order to deploy elastic cloud apps.
- When an environment gets launched, Elastic Beanstalk will merely use an already defined AMI which is accompanied with an operating system that is installed, then goes ahead with launching a newly created instance having the same type of your specification.
- Additionally, Beanstalk would set up an elastic load balancer making it respond to a unique URL.
- Due to the fact that Elastic Beanstalk would orchestrate various services, you will find extra procedures of interaction with those services.

Regardless of the fact that everything would be organized and verified by default, you will get the possibility to actually work with and change the assets managed by Elastic Beanstalk, where you've got the opportunity to either overwrite, adjust or bypass whatever Elastic Beanstalk performs. You can also get things customized based on what you require.

Comparison of Interest over time for Elastic Beanstalk Vs EC2:



Elastic Beanstalk Vs EC2 – Elastic Beanstalk and EC2 Interest Over Time

As it can be concluded from the above graph, Amazon EC2 has a higher interest over time while AWS Elastic Beanstalk has a low interest over time rate. This means that with the use of Amazon EC2 you can get a higher rate of interest as time goes by, while with AWS Elastic Beanstalk this rate will be way less.

Advantages and Disadvantages of AWS Elastic Beanstalk Vs EC2:

Check out in the table below some of the major advantages and disadvantages of AWS Beanstalk and those of Amazon EC2.

AWS Service	Advantages	Disadvantages
Elastic Beanstalk	1. Integrates with a variety of AWS services 2. Easy deployment 3. Quick 4. Painless 5.Neatly Documented	You will get charged directly upon exceeding the free quota
EC2	1. Fast and reliable cloud servers 2. Scalable 3. Easily managed 4. Low costing 5. Auto-scaling	Ui needs extra work Poor CPU performance High learning curve

Elastic Load Balancing (ELB)

Elastic Load Balancing (ELB) is a load-balancing service for Amazon Web Services (AWS) deployments. ELB automatically distributes incoming application traffic and scales resources to meet traffic demands.

ELB helps an IT team adjust capacity according to incoming application and network traffic. Users enable ELB within a single availability zone or across multiple availability zones to maintain consistent application performance.

Historically, load balancing divides the amount of work that a computer has to do amongmultiple computers so that users, in general, get served faster. ELB offers enhanced features including:

- Detection of unhealthy Elastic Compute Cloud (EC2) instances.
- Spreading instances across healthy channels only.
- Flexible cipher support.
- Centralized management of Secure Sockets Layer (SSL) certificates.
- Optional public key authentication.
- Support for both IPv4 and IPv6.

High availability

The most well-known service that relies on ELB is Amazon's EC2, as ELB performs a health check to ensure an instance is still running before sending traffic to it. When an instance fails or is unhealthy, ELB routes traffic to the remaining healthy EC2 instances.If all EC2 instances in a particular availability zone are unhealthy, ELB can route traffic to other availability zones until the original instances restore to a healthy state.

A developer can integrate Amazon Route 53 and domain name system (DNS) failover to further boost application resiliency. Route 53 can route traffic to another healthy ELB and fail over across AWS regions.

Automatic scaling

A developer can use AWS' Auto Scaling feature to guarantee he or she has enough EC2 instances running behind an ELB. The developer sets Auto Scaling conditions, and when a condition is met, a new EC2 instance can spin up to meet the desired minimum.A developer can also set a condition to spin up new EC2 instances to reduce latency.

Security

ELB supports applications within an Amazon Virtual Private Cloud for stronger network security. An IT team can specify whether it wants an internet-facing or internal load balancer. The latter option enables a developer to route traffic through an ELB using private IP addresses. A developer could also route traffic between different tiers of an application by using multiple internet-facing and internal load balancers; this approach allows an IT team to use a security group along with private IP addresses while exposing only the web-facing tier and its public IP addresses.

In addition to certificate management, ELB allows SSL/Transport Layer Security (TLS) decryption.

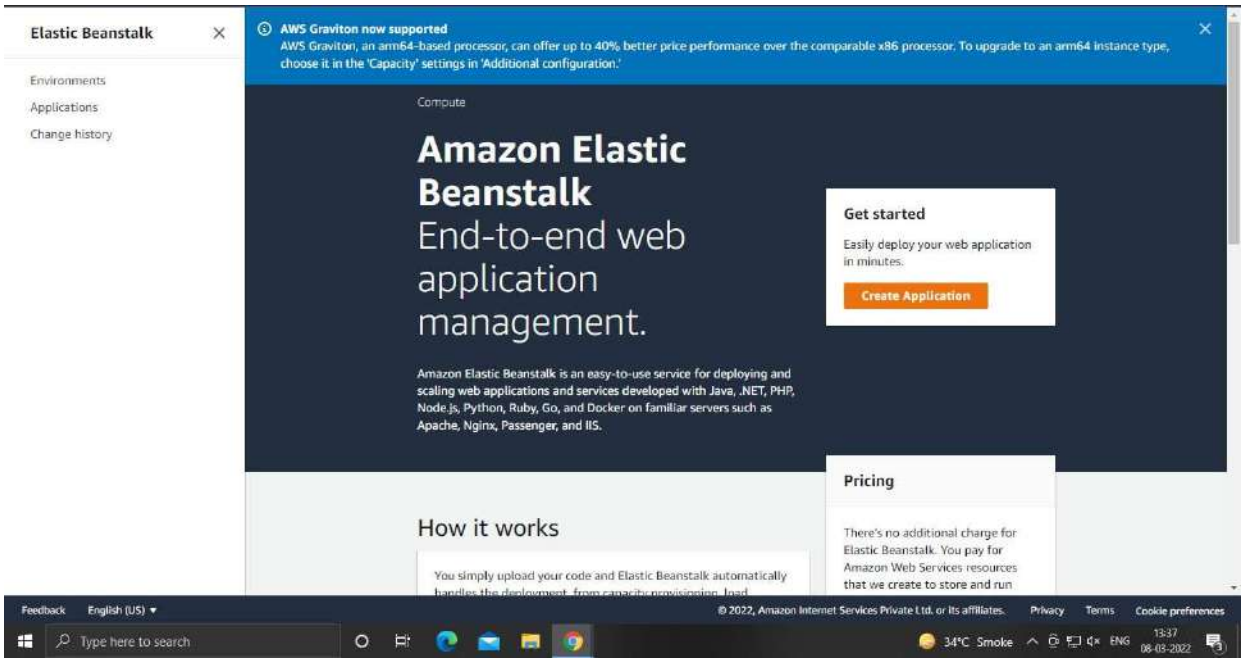
Types of load balancers

ELB offers two different load balancer features, which help provide scalable cloud computing capacity. The Application Load Balancer handles advanced traffic routing from other services or containers at the application level, while the Classic Load Balancer spreads app or network traffic across EC2 instances.

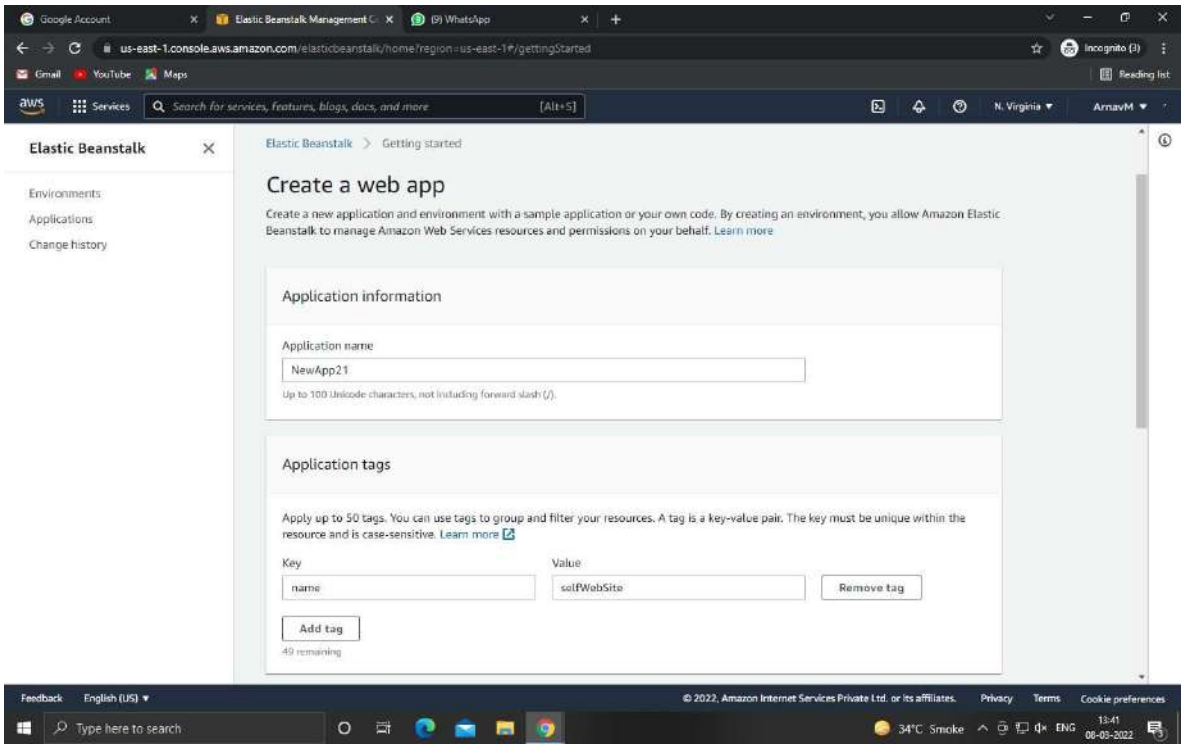
Other vendors also offer tools to load balance workloads. ScaleBase offers an ELB solution with real-time elasticity, which simplifies the ability to scale the MySQL relational database management system (RDBMS) without requiring infrastructure changes or taking services offline.

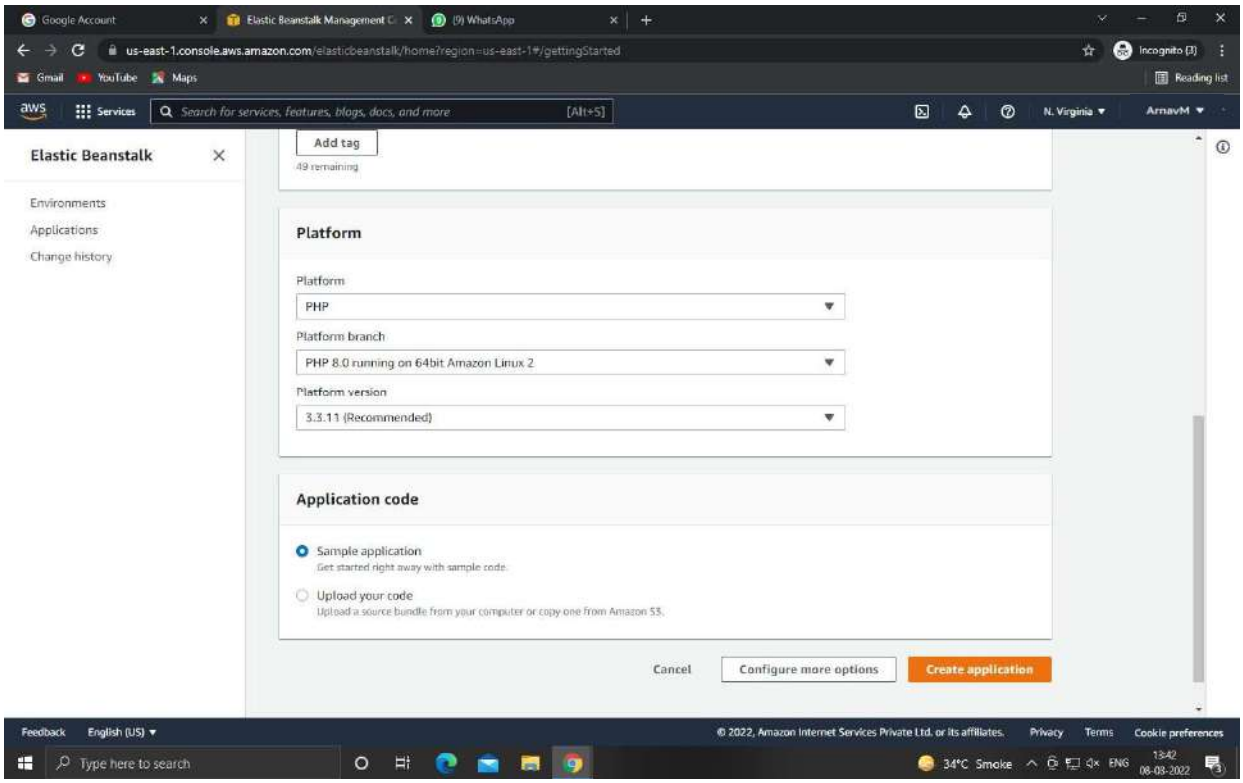
IMPLEMENTATION

Click on Create Application

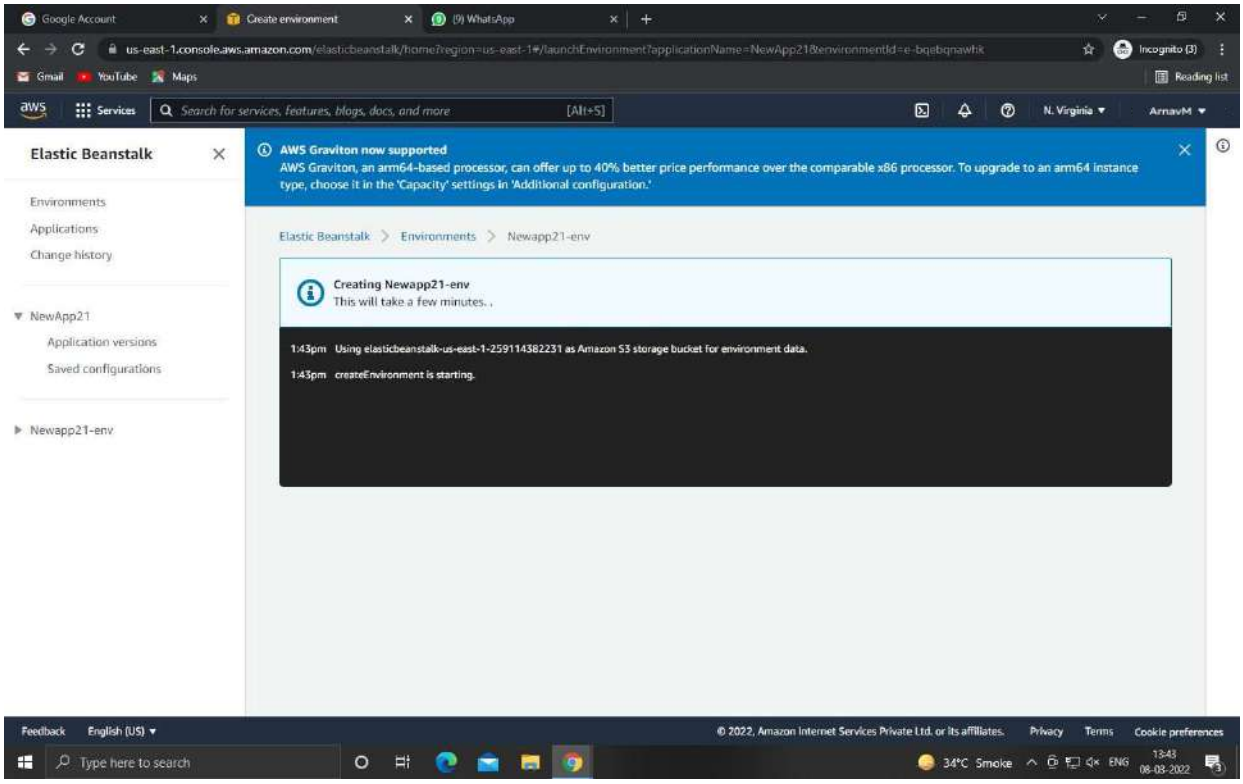


Write Application information : Name, Tag,Platform etc.

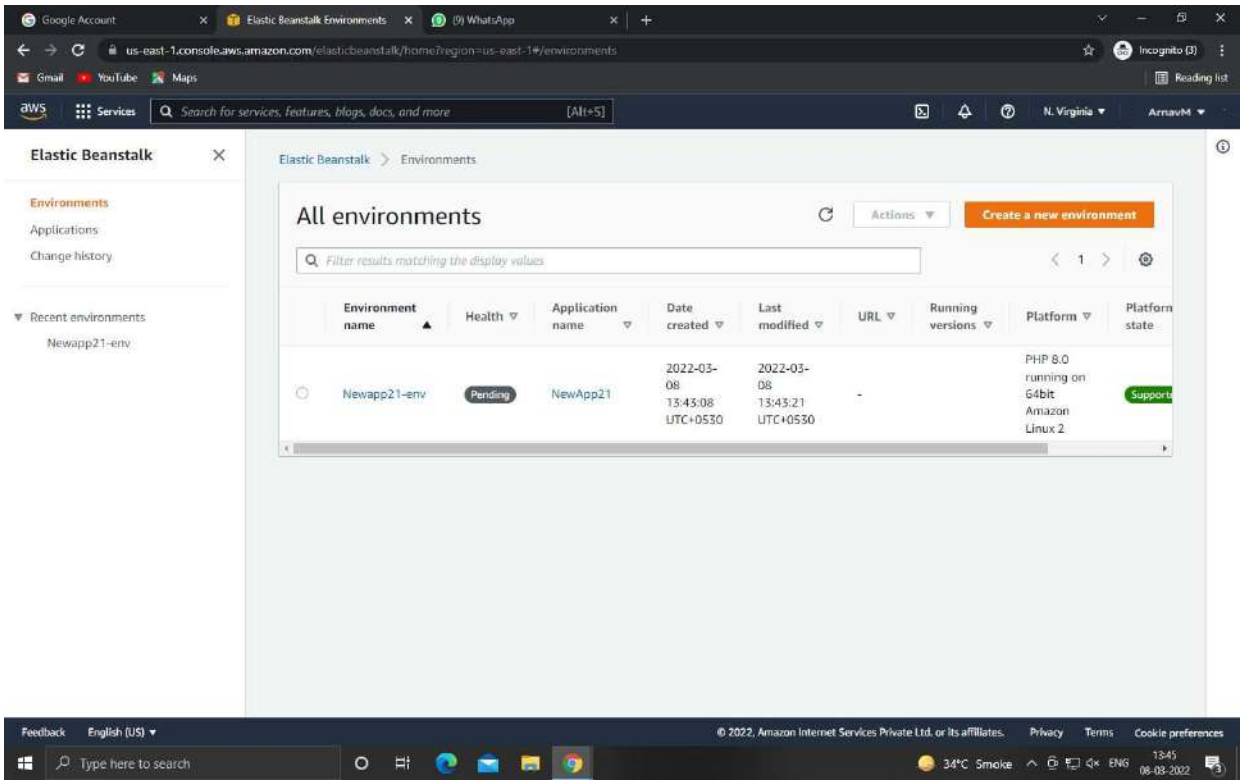


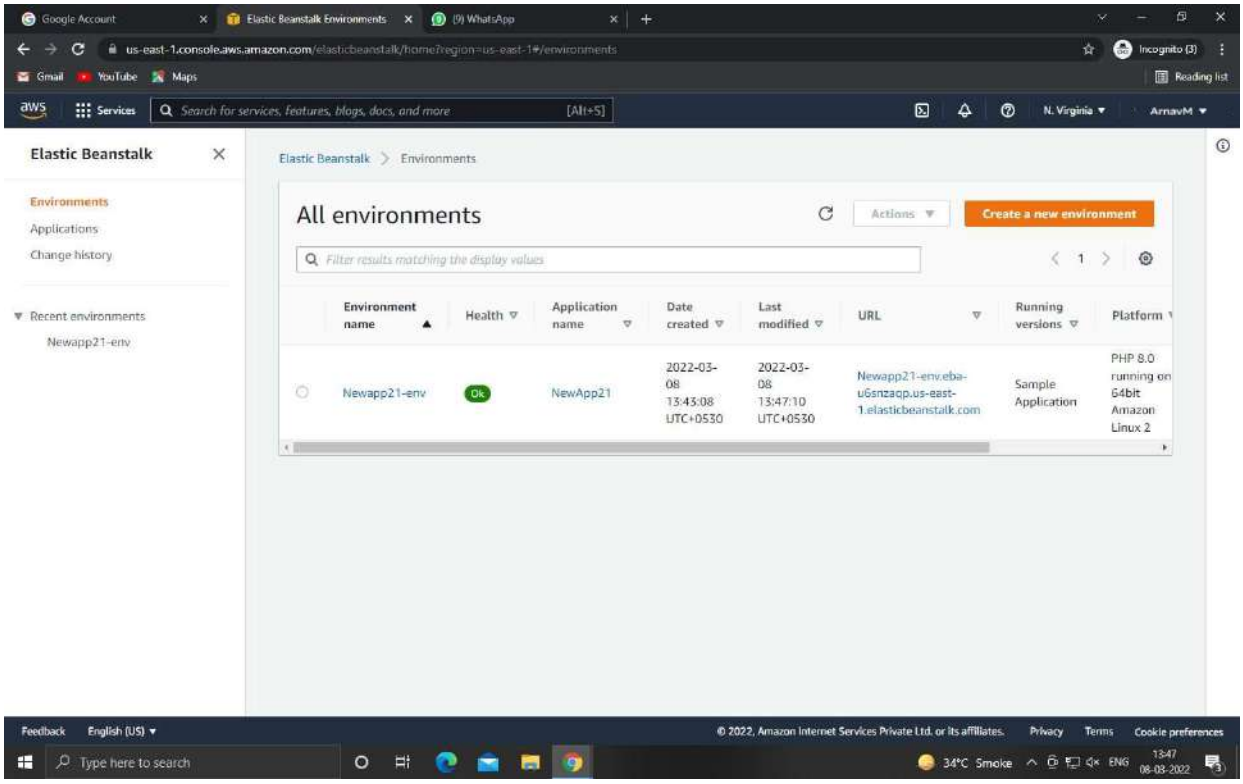


In Application Code: select sample application and then Click on button Create Application

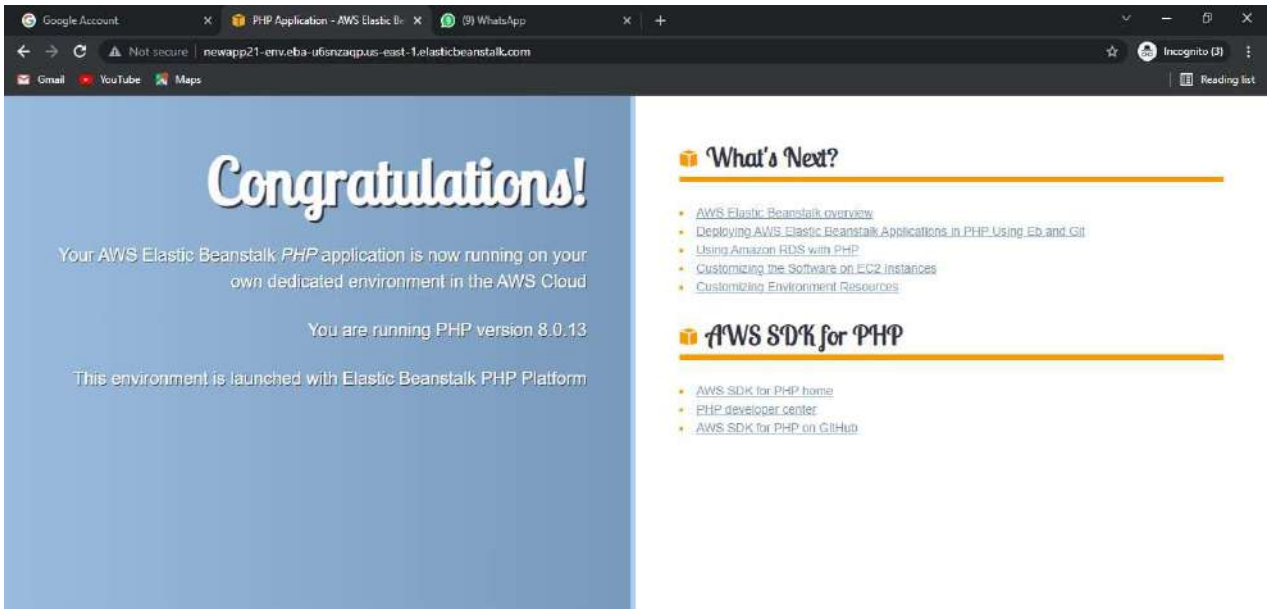


Click on Environments -> Check the health of Environment wait till it becomes ‘OK’





Click on the URL



Conclusion: Platform as a Service using AWS Elastic Beanstalk Service is successfully Implemented.

THADOMAL SHAHANI TSEC ENGINEERING COLLEGE

Experiment:- 6

Aim:- To study and implement storage as a service using own cloud.

Theory:-

Cloud storage- It allows you to save data and files in an off site location that you access either through the public internet or a dedicated private network connection. Data that you transfer off site for storage becomes responsibility of a third party cloud provider. The provider hosts, secures, manages and maintains the servers and associated infrastructure and ensures you have access to the data whenever you require it.

Own Cloud:- It is a file server that enables secure storage, collaboration and sharing. It is convenient to store files in cloud so they are available on any device and can be shared with a few clicks.

Features of own cloud-

1) Access your data
2) Sync your data
3) Share your data
4) Versioning
5) Encryption
6) Drag and Drop upload
7) Theming

Advantages of storage as a service-

1) More affordable short-term costs.
2) Scalability.
3) Security
4) Data Redundancy

Limitations -
1) Greater cost overtime
2) Data transfer speed.
3) Dependency on provider
4) Potential Downtime.

Object storage - It is a technology that stores and manages data in an unstructured format called objects. Modern organisations create and analyze large volumes of unstructured data across different types like photos, videos, email, audio files, etc

File Storage - Many applications need shared file access, This has been traditionally served by network-attached storage (NAS) services.

Block storage - It is a technology that controls data storage and storage devices. It takes any data, like a file or database entry and divides it into blocks of equal sizes. Think of ~~a~~ block storage as a more direct pipeline to the data.

Popular storage as a service vendor along with services:
1) Amazon Simple storage Service (s3) -
It is an object storage service offering industry leading sustainability, security and performance.

2) Microsoft Azure Blob Storage -
It's microsofts object storage solution within Azure cloud platform. It is designed for storage and managing large amount of unstructured data.

THADOMAL SHAHANI
TSEC
ENGINEERING COLLEGE

3) Google Cloud Storage-
It is a managed service for storing large amounts of unstructured data.

4) IBM cloud object storage-
It is an enterprise grade object storage solution that provides scalable and secure data storage. It is designed to support large scales.

5) Backblaze B2 cloud storage-
It offers cost effective object storage solutions. It is known for simplicity and competitive pricing. Key features are affordability, data durability, scalable storage and ease of use.

Conclusion:-
By performing the above experiment we understood how to implement storage as a service using own cloud.

A+
20/3/24

Name : Arnav Malvia
Batch : C23
Roll No.: 2103109

Experiment 6:

Aim: To study and Implement Storage as a Service using Own Cloud.

Theory:-

Introduction:
In Experiment 6, we delve into the implementation and study of Storage as a Service (SaaS) using OwnCloud. This experiment aims to explore the concept of cloud storage, understand OwnCloud and its features, discuss the advantages and limitations of SaaS, explain different types of storage, and provide an overview of popular SaaS vendors. Additionally, we will provide step-by-step installation instructions for OwnCloud and demonstrate its capabilities as a Storage as a Service solution.

1. Concept of Cloud Storage
Cloud storage refers to storing data on remote servers accessed over the internet instead of storing it locally on a physical disk or hard drive. It provides users with the ability to access their data from any location and device with internet connectivity. Cloud storage services typically offer scalability, reliability, and redundancy features, allowing users to store, manage, and retrieve data seamlessly.

2. OwnCloud and its Features
OwnCloud is an open-source cloud storage platform that allows users to store, sync, and share files securely. Its features include:

- File Synchronization: Sync files across devices and platforms.
- File Sharing: Share files and folders with individuals or groups, with customizable permissions.
- Security: OwnCloud offers encryption capabilities to ensure data security.
- Collaboration: Collaborate on documents in real-time with built-in editing features.
- Integration: Integration with other services such as calendars, contacts, and productivity tools.
- Scalability: OwnCloud can scale to meet the needs of both individuals and enterprises.

3. Advantages and Limitations of Storage as a Service
Advantages:
- Cost Savings: Eliminates the need for purchasing and maintaining physical storage infrastructure.
- Scalability: Easily scale storage resources up or down based on demand.
- Accessibility: Data can be accessed from anywhere with an internet connection.
- Redundancy and Disaster Recovery: Cloud storage providers often offer redundancy and backup options to ensure data availability and disaster recovery.
- Collaboration: Facilitates easy collaboration and sharing of files among users.

Limitations:
- Dependency on Internet: Access to data is reliant on internet connectivity.
- Security Concerns: Data security and privacy concerns may arise due to reliance on third-party cloud providers.
- Data Transfer Speed: Transfer speeds may be slower compared to local storage, especially for large files.
- Limited Control: Users may have limited control over data management and security policies compared to on-premises storage solutions.
4. Types of Storages
Object Storage:
Object storage is a data storage architecture that manages data as objects rather than as blocks or files. Each object typically includes the data itself, metadata, and a unique identifier.

Object storage is highly scalable and suitable for storing large amounts of unstructured data such as documents, images, and multimedia files.

Block Level Storage:
Block-level storage involves dividing data into blocks, each with its address and identifier. It is typically used in Storage Area Networks (SANs) and provides high-performance storage suitable for databases and transactional applications.

5. Popular Storage-as-a-Service Vendors
Amazon S3 (Simple Storage Service):
Amazon S3 is a highly scalable, secure, and durable object storage service offered by Amazon Web Services (AWS). It provides a simple web services interface to store and retrieve any amount of data from anywhere on the web.

Google Cloud Storage:
Google Cloud Storage is an object storage service provided by Google Cloud Platform. It offers high availability, durability, and scalability, with various storage classes to optimize costs based on access frequency and latency requirements.

Microsoft Azure Blob Storage:
Azure Blob Storage is Microsoft's object storage solution within the Azure cloud platform. It provides scalable storage for unstructured data, including documents, images, and media files, with features such as encryption and access control.
Dropbox:
Dropbox is a popular file hosting service that offers cloud storage, file synchronization, and personal cloud features. It provides easy collaboration and sharing options for individuals and businesses.

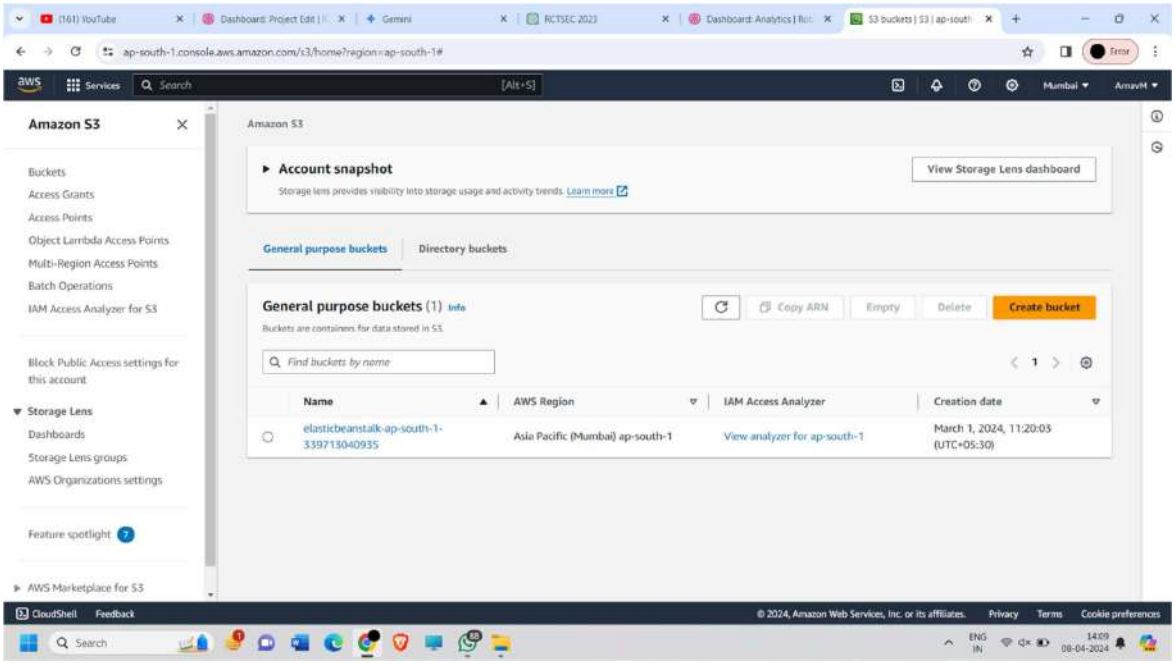
Box:
Box is a cloud content management and file sharing service designed for businesses. It offers secure storage, collaboration, and workflow automation features, with integrations with various productivity tools and applications.

Installation and Demonstration of OwnCloud as Storage as a Service
Step 1: Installation of OwnCloud
1. Download the OwnCloud package from the official website.
2. Install the required dependencies such as Apache, MySQL, and PHP.
3. Extract the OwnCloud package to the web server directory.
4. Configure the database settings and complete the installation via the web interface.
5. Set up user accounts and configure storage options.

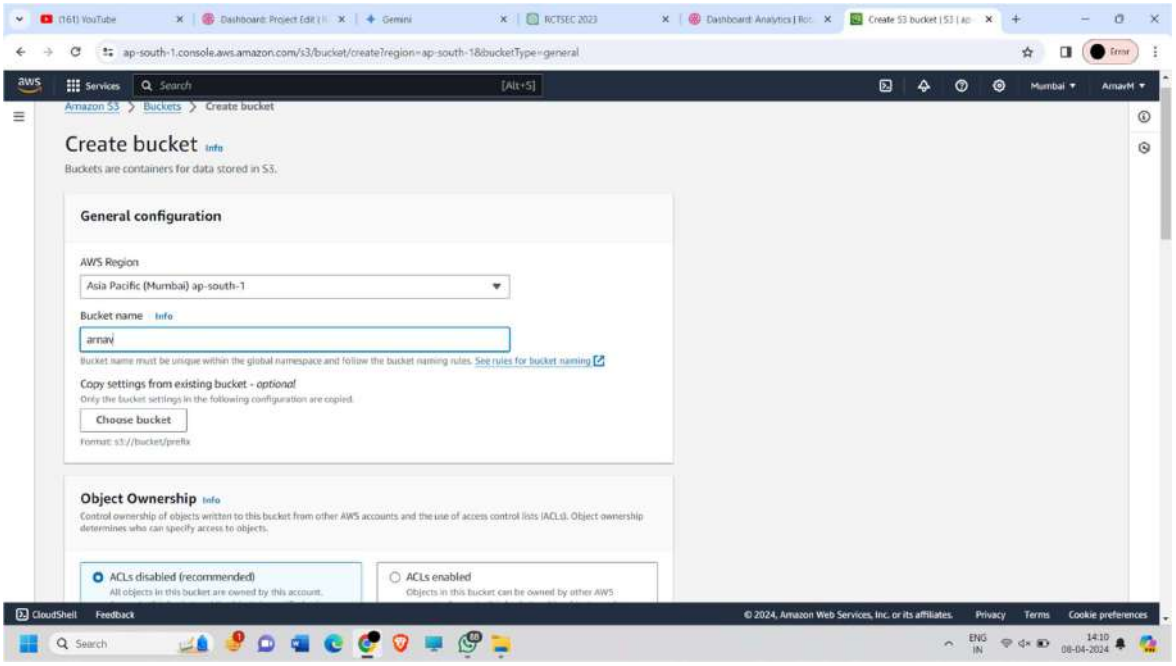
Step 2: Demonstration of Storage as a Service
1. Upload files to OwnCloud from different devices.
2. Access the uploaded files from the OwnCloud web interface.
3. Share files with other users and set permissions.
4. Sync files across devices using the OwnCloud desktop or mobile client.
5. Collaborate on documents in real-time using built-in editing features.

Implementation:

1. Open amazon web service and search for S3
2. Click on create bucket



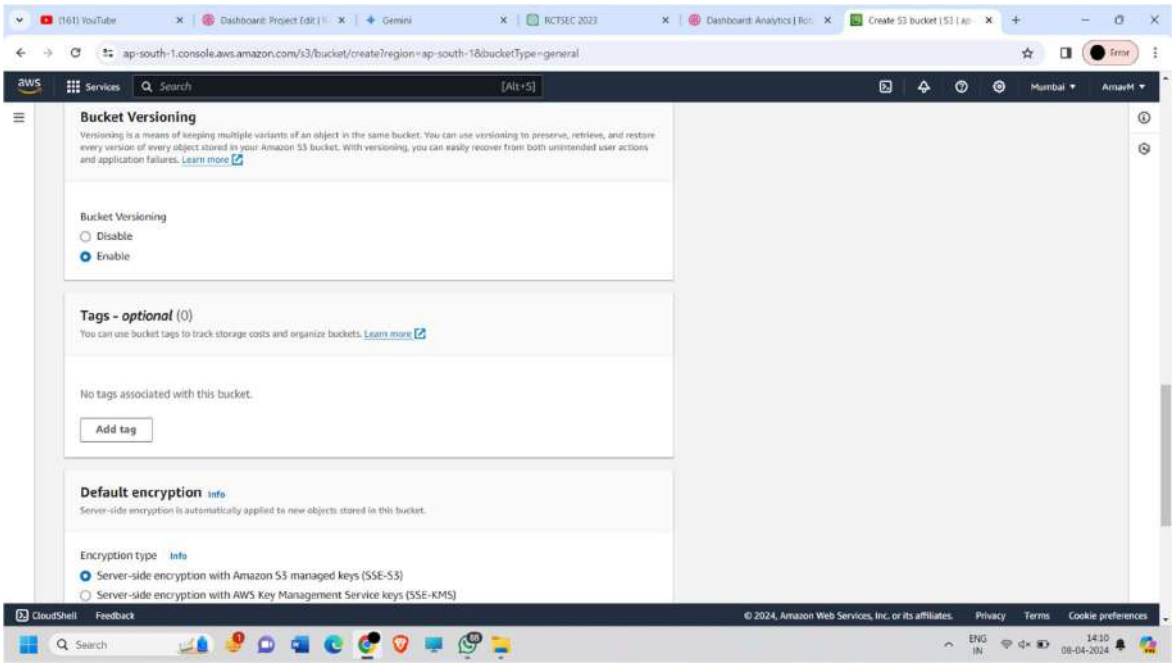
3. Add general information



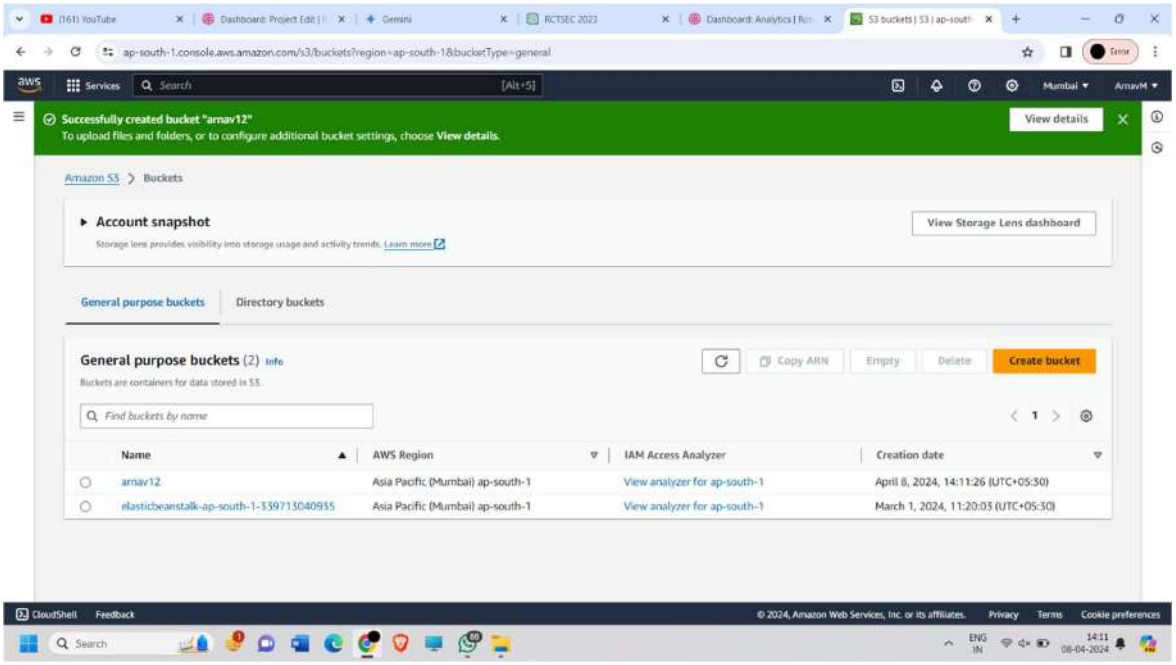
4. Select object ownership

5. Disable the bucket version

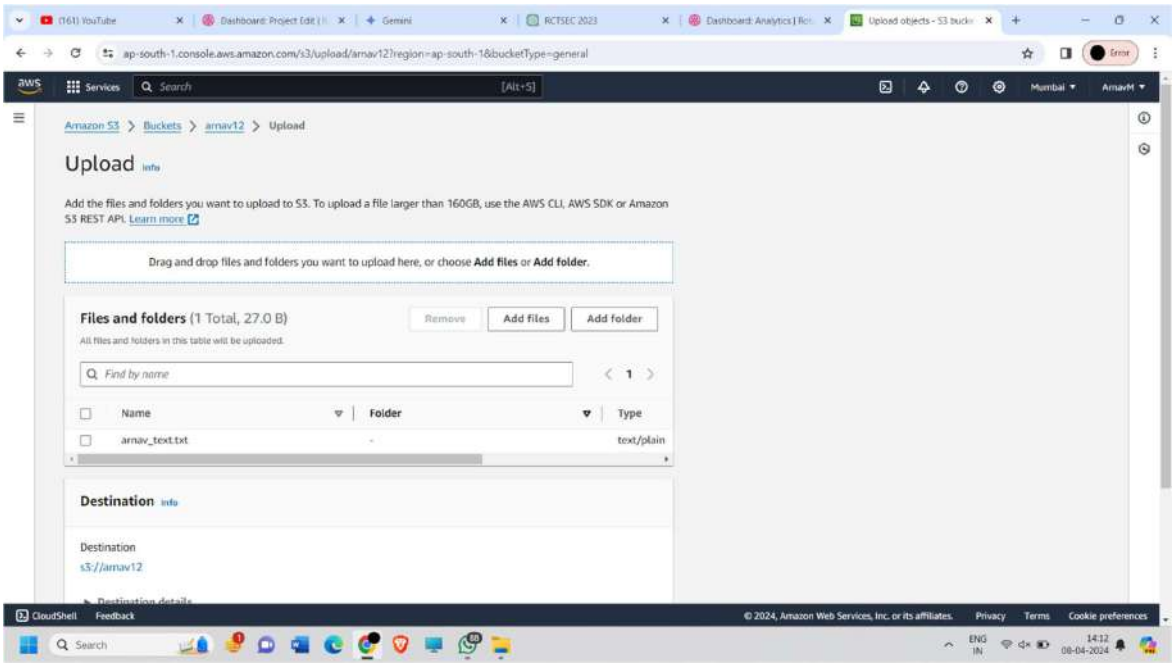
6. Add default encryption details



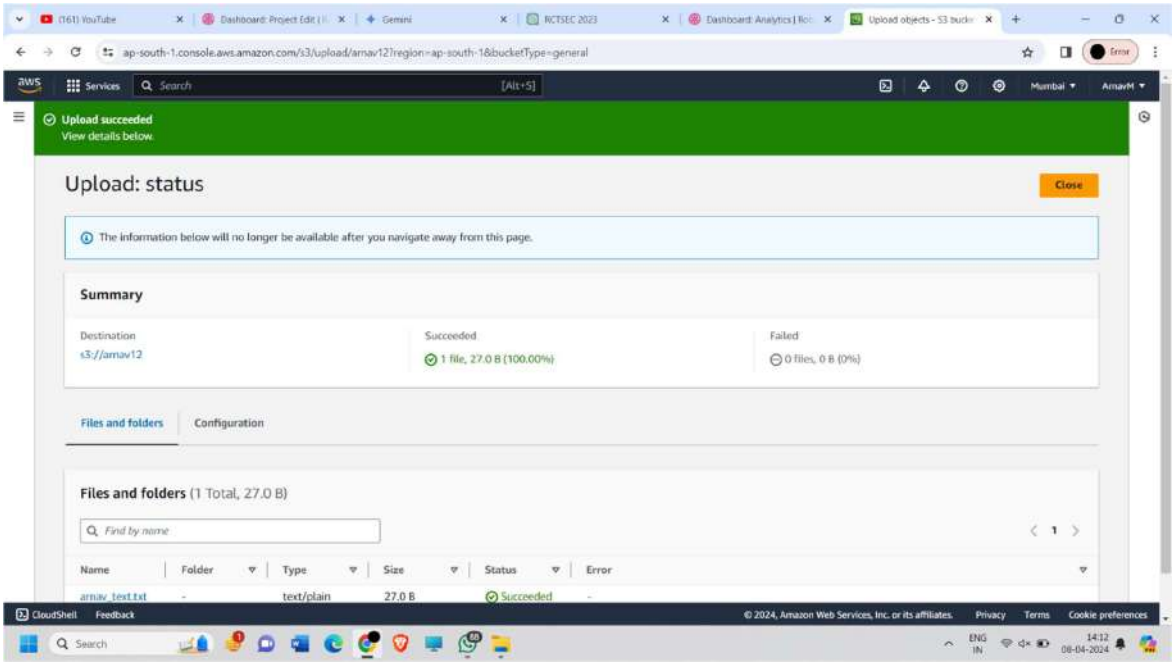
7. The bucket has been created



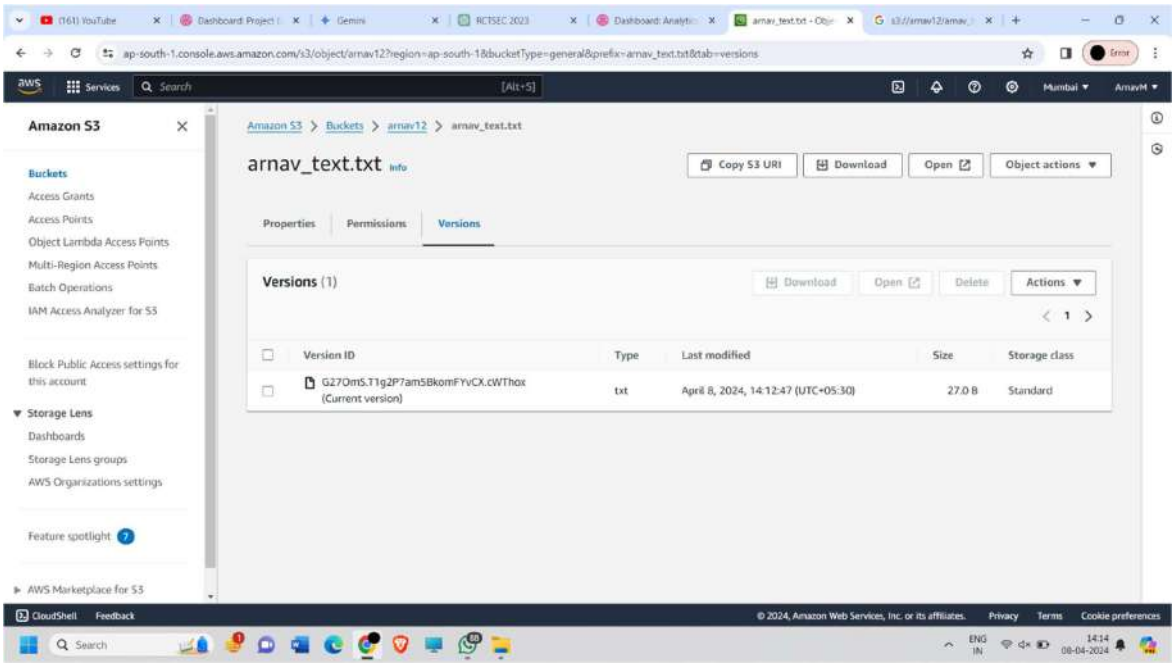
8. Click on upload



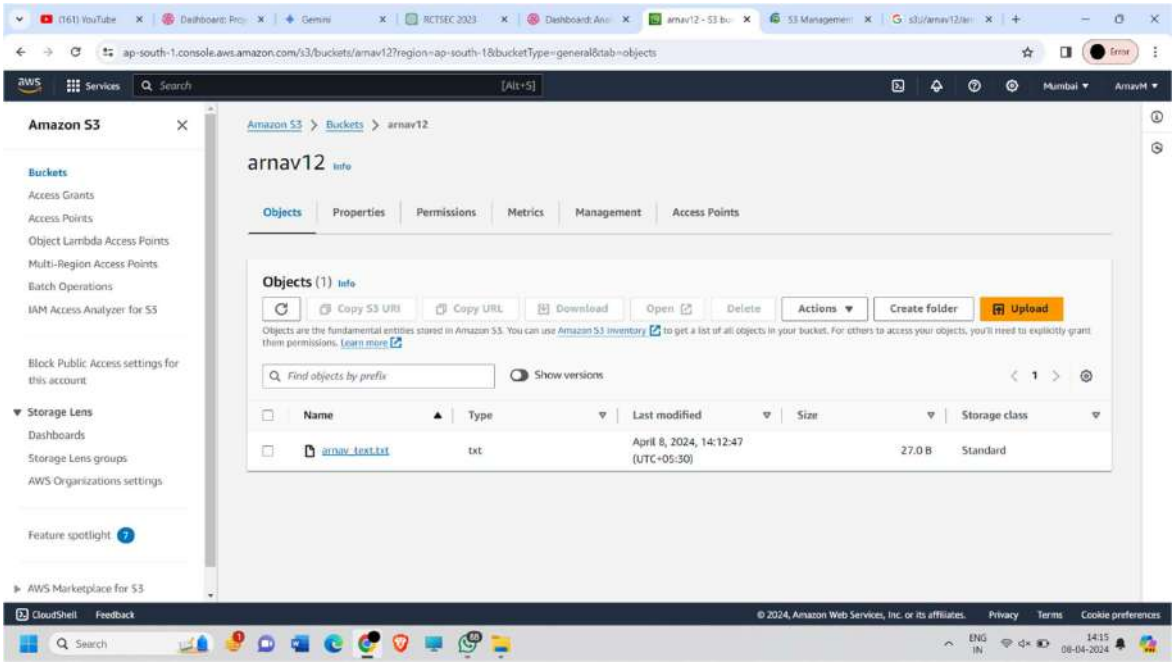
9. Upload



10. Object is created



11. Properties



12. Edit static website hosting

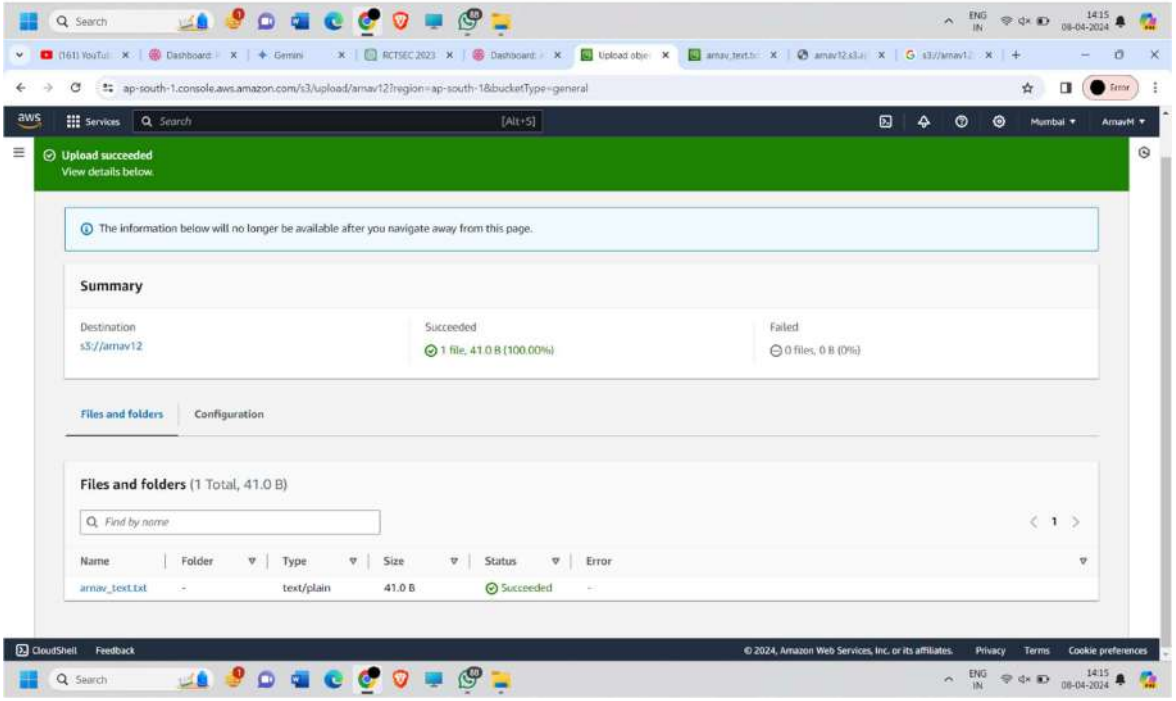
13. Insert document

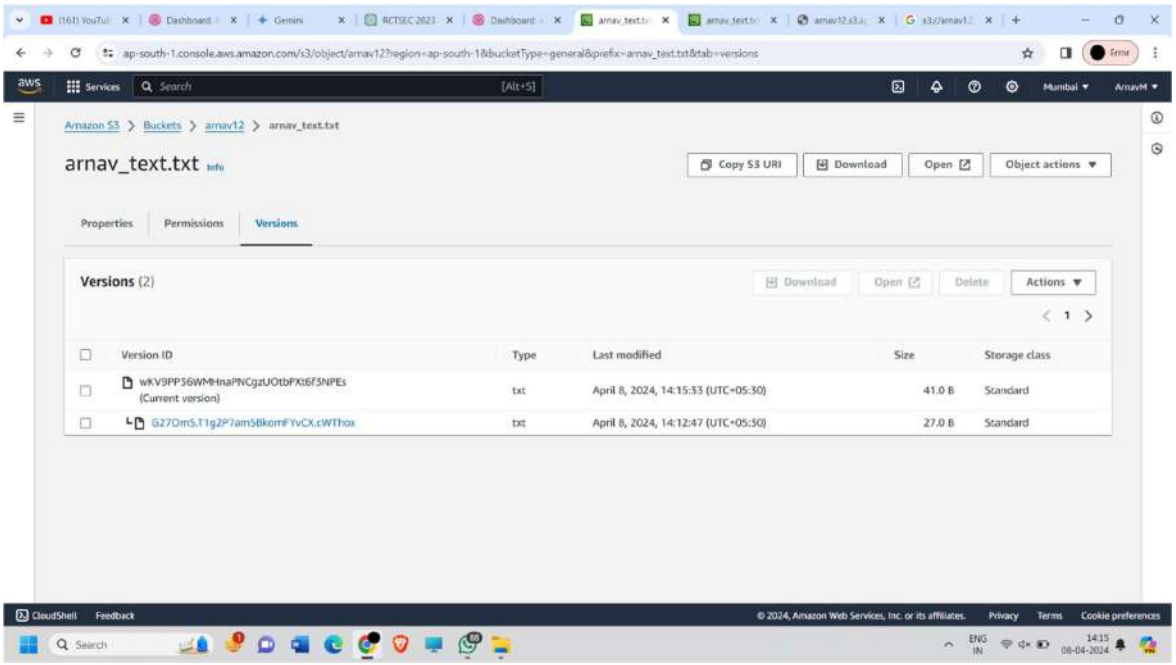
14. Permissions

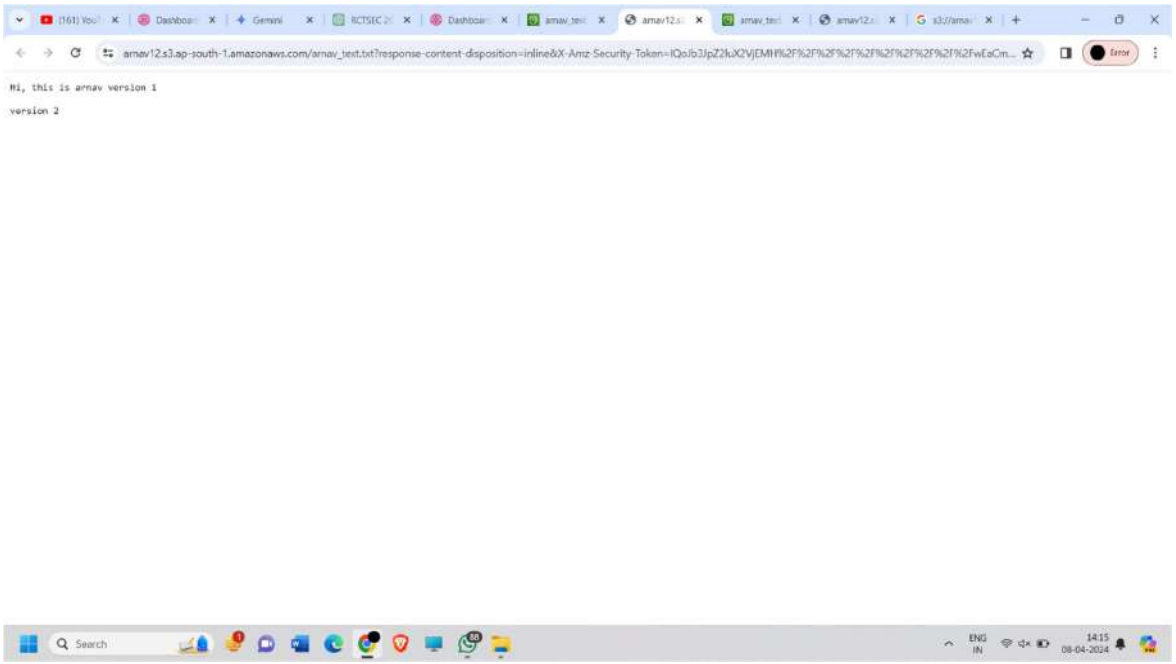
15. Edit bucket policy

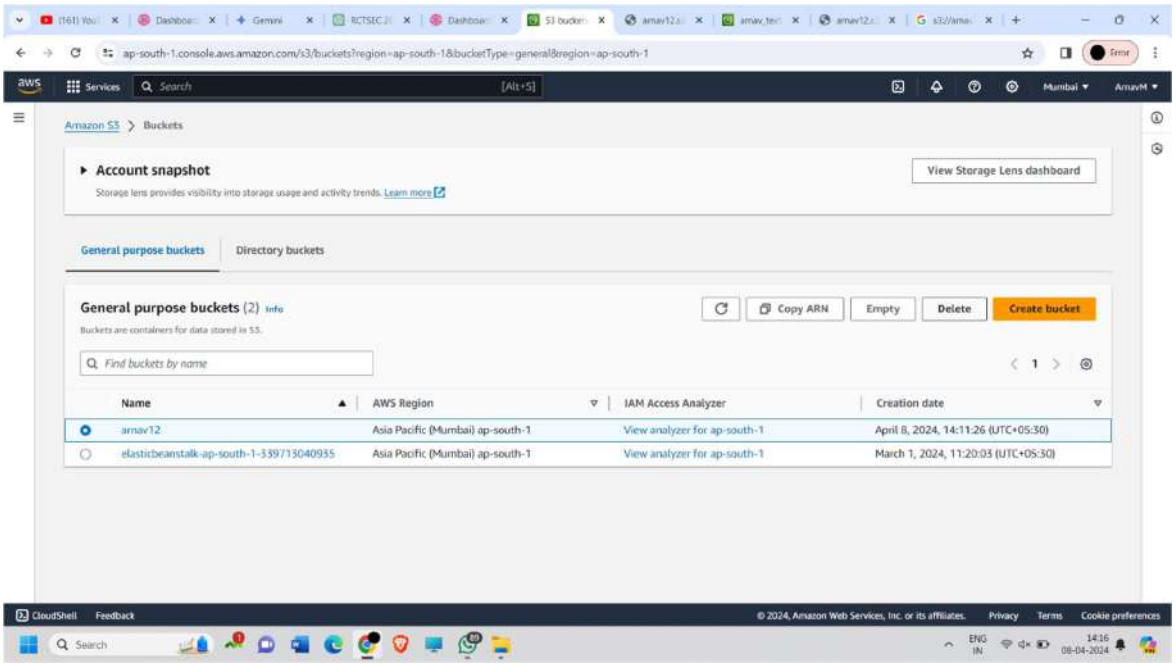
16. Check the object overview


arnav12.s3.ap-south-1.amazonaws.com/arnav_text.txt?response-content-disposition=inline&X-Amz-Security-Token=IQoJb3JpZ2luX2VjEMH%2F%2F%2F%2F%2F%2F%2F%2F%2F%2FwEaCm...
Hi, this is arnav version 1


ap-south-1.console.aws.amazon.com/s3/upload/arnav12?region=ap-south-1&bucketType=general
Upload succeeded
View details below.
The information below will no longer be available after you navigate away from this page.
Summary
Destination
s3://arnav12
Succeeded
1 file, 41.0 B (100.00%)
Failed
0 files, 0 B (0%)
Files and folders
Configuration
Files and folders (1 Total, 41.0 B)
Find by name
Name
Folder
Type
Size
Status
Error
arnav_text.txt
-
text/plain
41.0 B
Succeeded
-
CloudShell
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences


ap-south-1.console.aws.amazon.com/s3/object/arnav12?region=ap-south-1&bucketType=general&prefix=arnav_text.txt&tab=versions
Amazon S3 > Buckets > arnav12 > arnav_text.txt
arnav_text.txt Info
Copy S3 URI
Download
Open
Object actions
Properties
Permissions
Versions
Versions (2)
Download
Open
Delete
Actions
Version ID
Type
Last modified
Size
Storage class
wKV9PP36WMHnaPNCgzUOtbFXt6f5NPEs (Current version)
txt
April 8, 2024, 14:15:33 (UTC+05:30)
41.0 B
Standard
G27Om5.T1g2P7am5BkomFYvCX.cWThox
txt
April 8, 2024, 14:12:47 (UTC+05:30)
27.0 B
Standard
CloudShell
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences

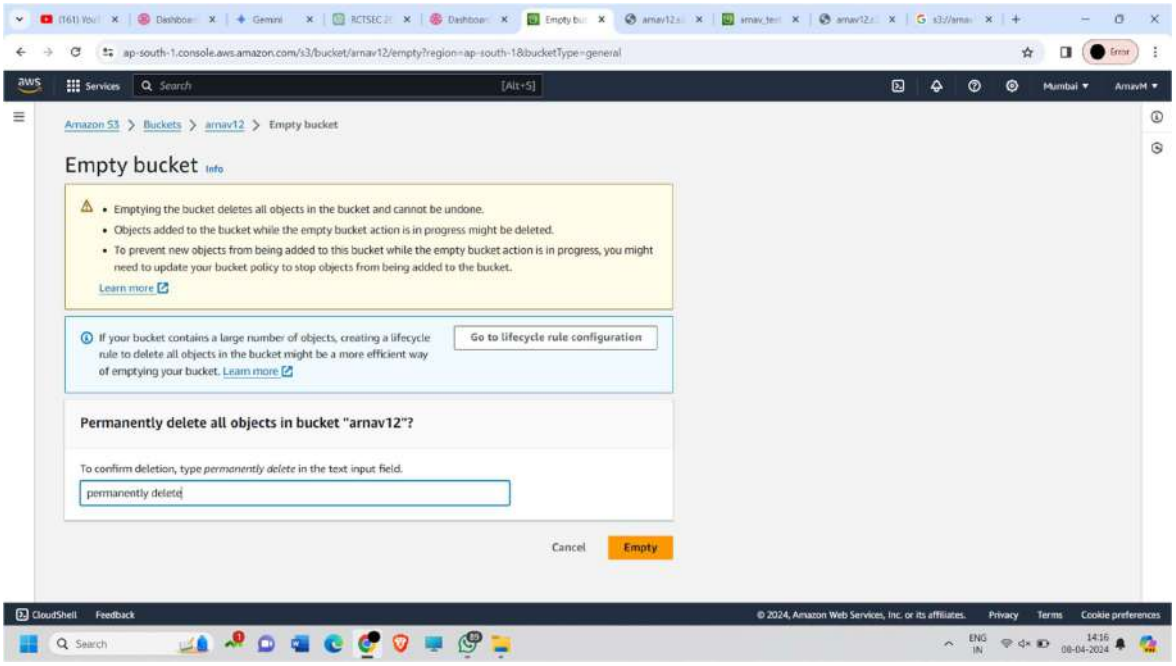

Hi, this is arnav version 1
version 2

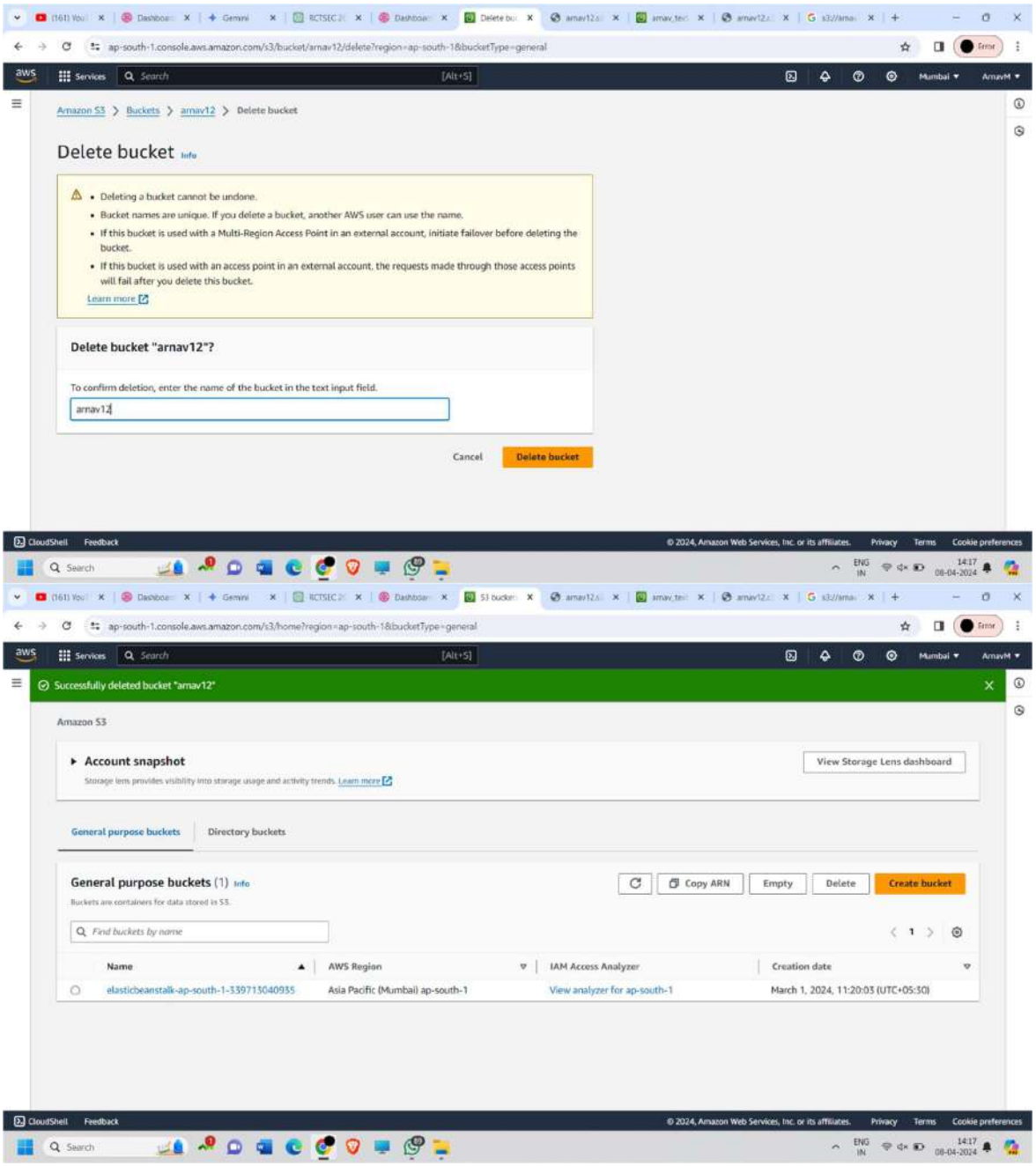


17. Delete objects

18. Select the bucket

19. Delete the bucket




ap-south-1.console.aws.amazon.com/s3/bucket/arnav12/delete?region=ap-south-1&bucketType=general
Services
Search
[Alt+S]
Mumbai
Amazon S3 > Buckets > arnav12 > Delete bucket
Delete bucket Info
Deleting a bucket cannot be undone.
Bucket names are unique. If you delete a bucket, another AWS user can use the name.
If this bucket is used with a Multi-Region Access Point in an external account, initiate failover before deleting the bucket.
If this bucket is used with an access point in an external account, the requests made through those access points will fail after you delete this bucket.
Learn more
Delete bucket "arnav12"?
To confirm deletion, enter the name of the bucket in the text input field.
arnav12
Cancel
Delete bucket
CloudShell
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences

ap-south-1.console.aws.amazon.com/s3/home?region=ap-south-1&bucketType=general
Successfully deleted bucket "arnav12"
Amazon S3
Account snapshot
Storage lens provides visibility into storage usage and activity trends. Learn more
View Storage Lens dashboard
General purpose buckets
Directory buckets
General purpose buckets (1) Info
Buckets are containers for data stored in S3.
Copy ARN
Empty
Delete
Create bucket
Find buckets by name
Name
AWS Region
IAM Access Analyzer
Creation date
elasticbeanstalk-ap-south-1-339713040935
Asia Pacific (Mumbai) ap-south-1
View analyzer for ap-south-1
March 1, 2024, 11:20:03 (UTC+05:30)
CloudShell
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences

THADOMAL SHAHANI
TSEC
ENGINEERING COLLEGE

Experiment :- 7

Aim :- To study and implement Identity and Access Management (IAM) practices on AWS.

Theory :-

Identity and Access Management (IAM) is the cornerstone of ensuring that only authorized individuals and services can interact with cloud resources. Access management refers to the practice of controlling and monitoring access to resources within a system. In cloud environments like AWS, where resources are abstracted and accessed remotely, ensuring secure access is crucial. The need arises from the requirement to safeguard sensitive data, prevent unauthorised usage of resources and comply with regulatory standards.

IAM and its components :-

1) Users
2) Groups
3) Roles
4) Policies

Inline and Custom Policies :-

Inline are policies that are embedded directly into a single user, group or a role. Inline policies are tightly coupled with the entity to which they are attached and cannot be reused across multiple entities.

Custom policies are standalone documents that can be attached to multiple users, groups or roles. They

offer more flexibility and reusability compared to inline policies.

Multi-Factor Authentication (MFA) in AWS:-
MFA adds an extra layer of security to AWS accounts by requiring users to present two or more forms of verification before granting access. This typically involves something the user knows and something they possess. MFA can be enabled for IAM users and can be modified or forced for specific actions.

Conclusion:-
Understood and implemented Identity and Access management (IAM) practices on AWS.

A
20/3/24

Arnav Malvia
C23
2103109

Cloud Computing Experiment 7

Aim: To study and implement Identity and Access Management (IAM) practices on AWS.

Theory:

Introduction:

In the realm of cloud computing, maintaining secure access to resources is paramount. Identity and Access Management (IAM) is the cornerstone of ensuring that only authorized individuals and services can interact with cloud resources. This write-up delves into the concept of access management, the components of AWS IAM, a comparison between root users and IAM users, roles and policies, types of policies, and Multi-Factor Authentication (MFA) on AWS.

Concept and Need of Access Management:

Access management refers to the practice of controlling and monitoring access to resources within a system. In cloud environments like AWS, where resources are abstracted and accessed remotely, ensuring secure access is crucial. The need arises from the requirement to safeguard sensitive data, prevent unauthorized usage of resources, and comply with regulatory standards.

IAM and Its Components:

IAM in AWS is a service that enables you to manage access to AWS services and resources securely. Its components include:

Users: Represent individuals or services interacting with AWS resources. Each user has unique security credentials.

Groups: A collection of users, simplifying the management of permissions by applying policies to groups rather than individual users.

Roles: Similar to users, but intended for services or entities outside of AWS, allowing them temporary access to AWS resources.

Policies: JSON documents defining permissions. These can be attached to users, groups, or roles to specify the actions they are allowed or denied.

Root Users vs. IAM Users:

Root Users: When an AWS account is created, the email address and password used become the root user credentials. Root users have complete access to all resources and services in the account. However, it's recommended to avoid using root credentials for day-to-day activities due to security risks.

IAM Users: These are users created within AWS IAM. IAM users have specific permissions assigned to them through policies. They don't have inherent access to all resources like root users, but their permissions can be finely tuned.

Roles and Policies:

Roles: IAM roles are meant to be assumed by entities within or outside of AWS, such as AWS services, applications, or users. Roles define a set of permissions, and temporary security credentials are provided when a role is assumed.

Policies: IAM policies are documents that define permissions. They can be attached to users, groups, or roles. Policies specify what actions are allowed or denied, and on which resources.

Inline and Custom Policies in AWS:

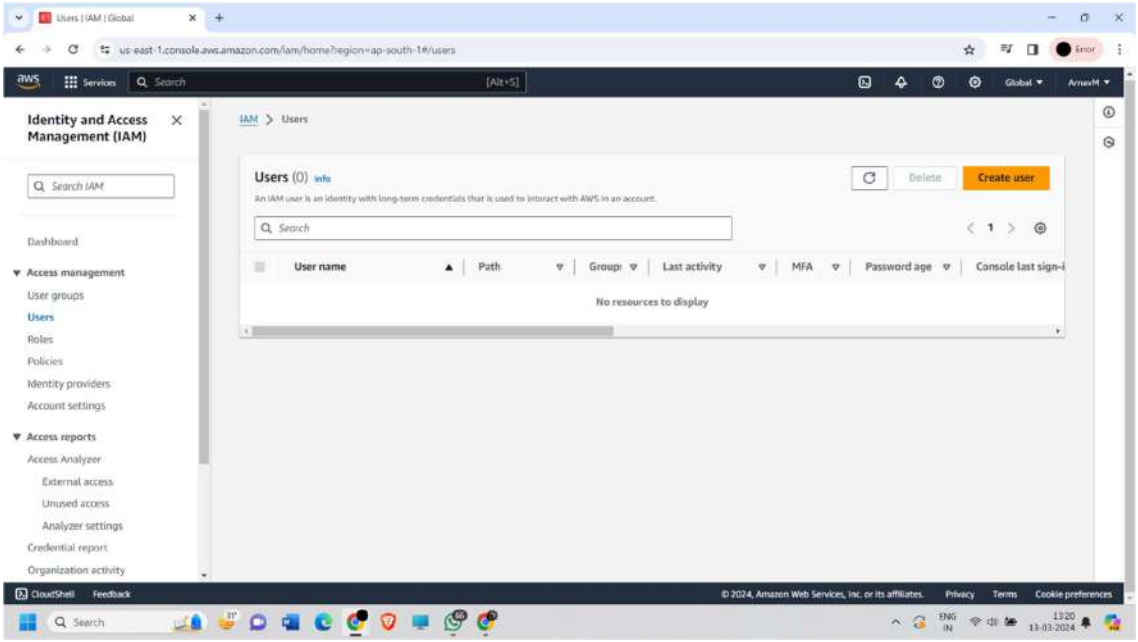
Inline Policies: These are policies that are embedded directly into a single user, group, or role. Inline policies are tightly coupled with the entity to which they are attached and cannot be reused across multiple entities.

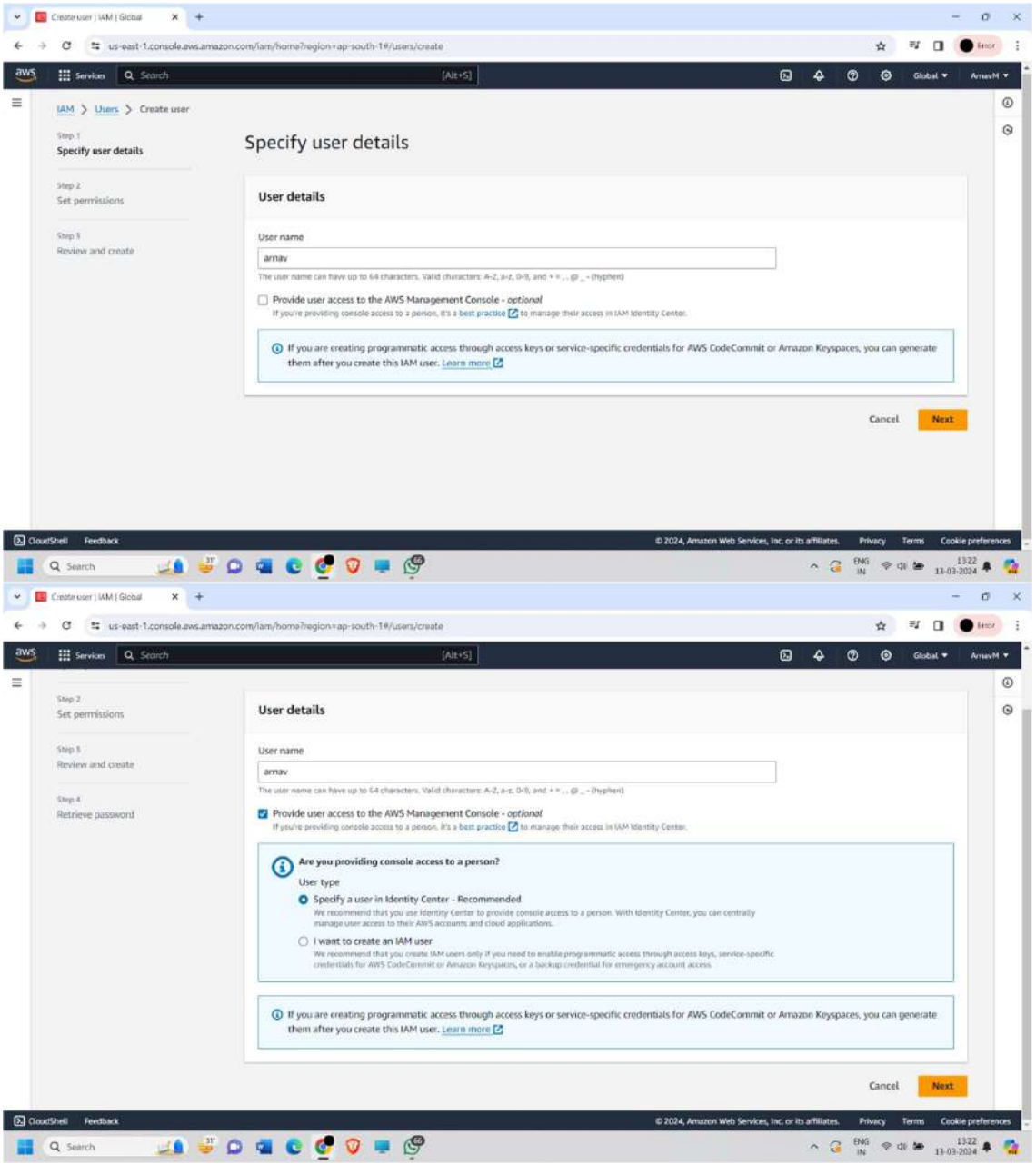
Custom Policies: Custom policies are standalone documents that can be attached to multiple users, groups, or roles. They offer more flexibility and reusability compared to inline policies.

Multi-Factor Authentication (MFA) in AWS:

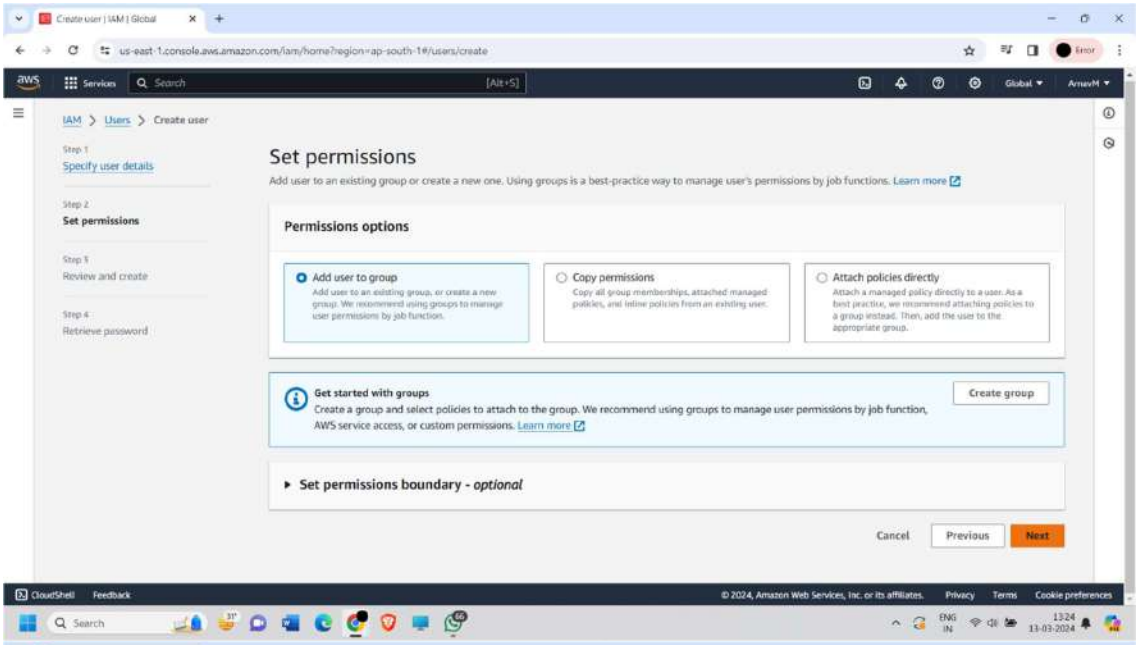
MFA adds an extra layer of security to AWS accounts by requiring users to present two or more forms of verification before granting access. This typically involves something the user knows (like a password) and something they possess (like a mobile device or hardware token). MFA can be enabled for IAM users and can be enforced for specific actions or console logins, enhancing security by mitigating the risks associated with compromised passwords.

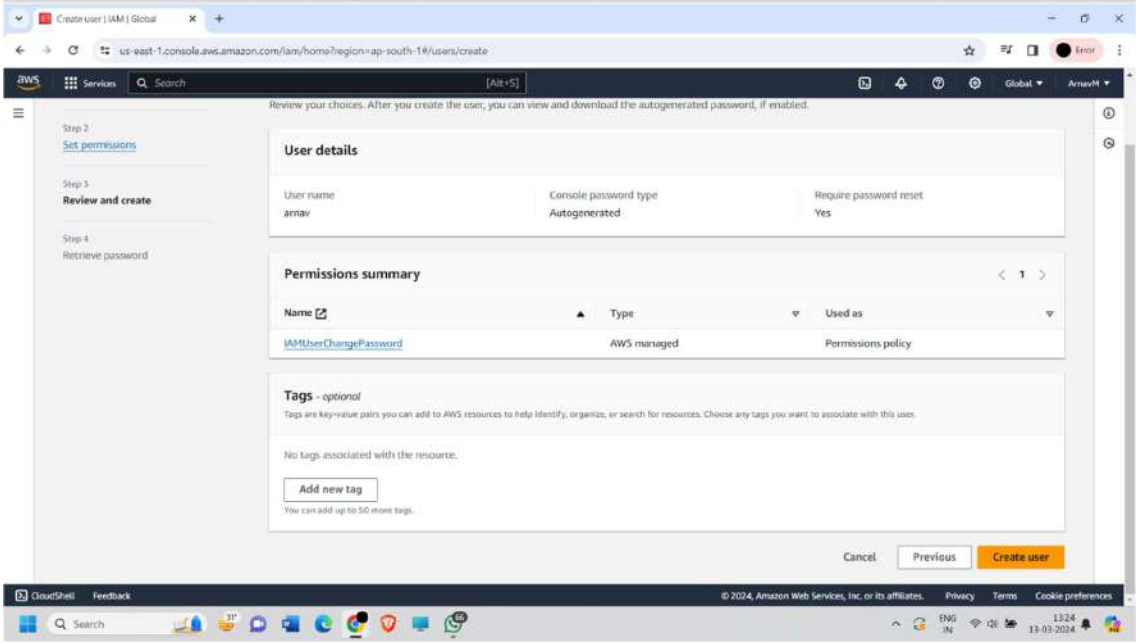
Screenshots

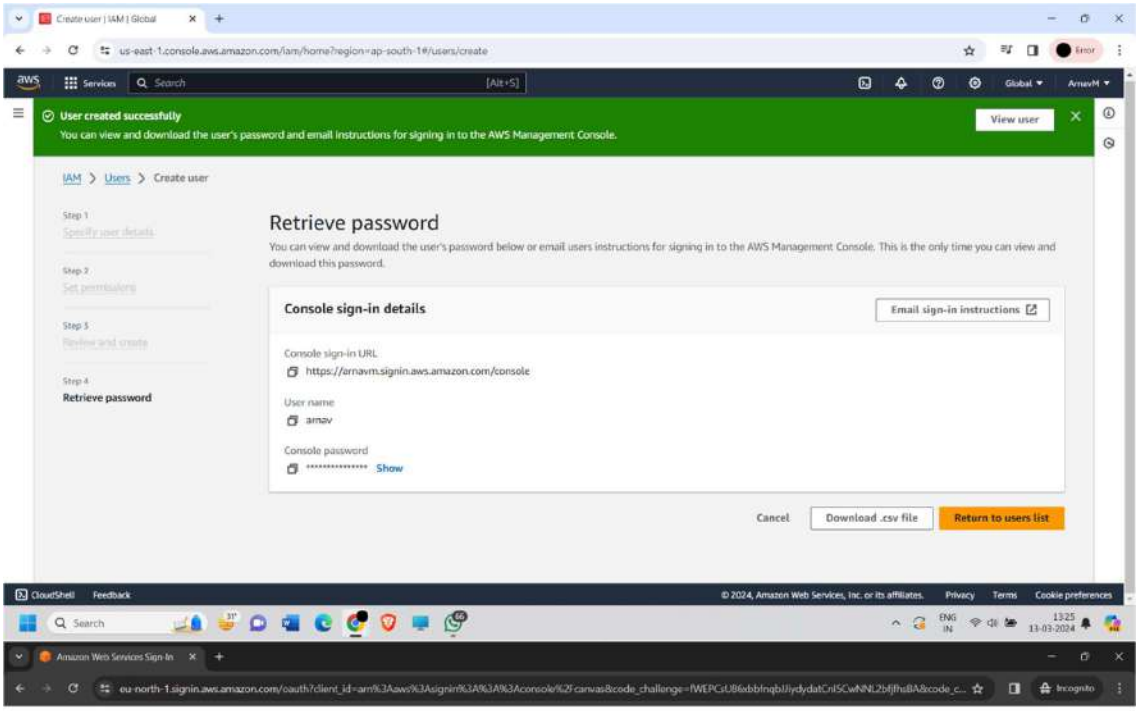


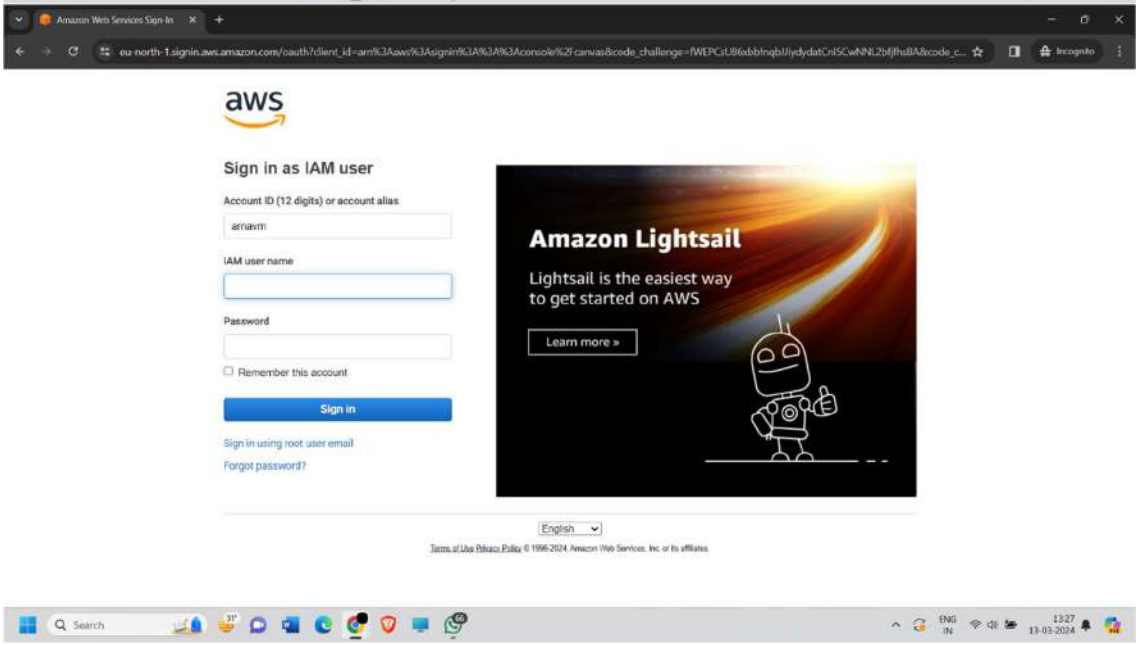

Create user | IAM | Global
us-east-1.console.aws.amazon.com/iam/home?region=ap-south-1#/users/create
Services
Search
[Alt+S]
Global
ArnavM
IAM > Users > Create user
Step 1
Specify user details
Step 2
Set permissions
Step 3
Review and create
Specify user details
User details
User name
arnav
The user name can have up to 64 characters. Valid characters: A-Z, a-z, 0-9, and + = , . @ _ - (hyphen)
Provide user access to the AWS Management Console - optional
If you're providing console access to a person, it's a best practice to manage their access in IAM Identity Center.
If you are creating programmatic access through access keys or service-specific credentials for AWS CodeCommit or Amazon Keyspaces, you can generate them after you create this IAM user. Learn more
Cancel
Next
CloudShell
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences
Search
ENG
IN
13:22
13-03-2024

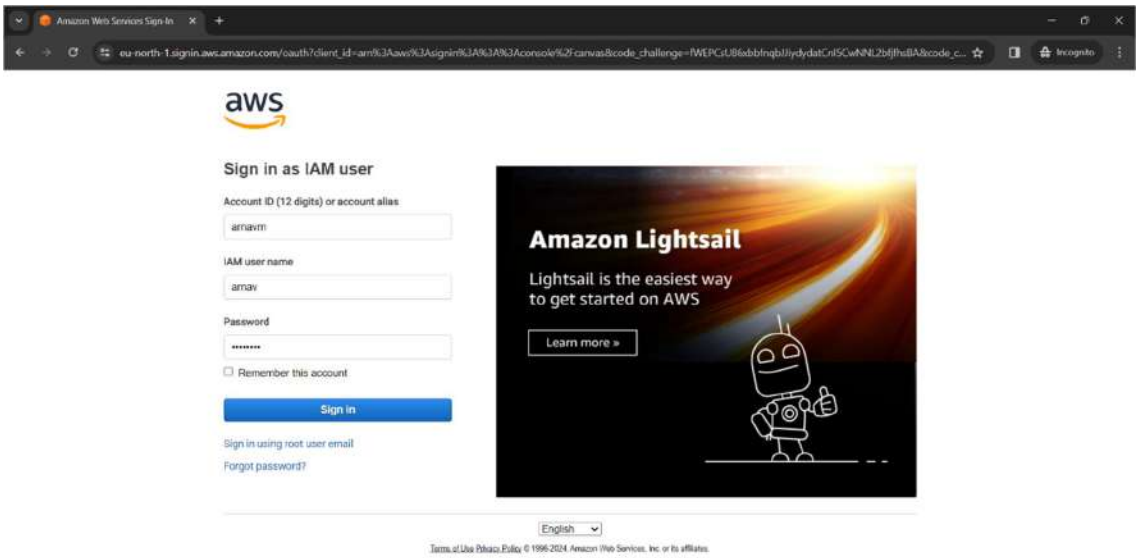
Create user | IAM | Global
us-east-1.console.aws.amazon.com/iam/home?region=ap-south-1#/users/create
Services
Search
[Alt+S]
Global
ArnavM
Step 2
Set permissions
Step 3
Review and create
Step 4
Retrieve password
User details
User name
arnav
The user name can have up to 64 characters. Valid characters: A-Z, a-z, 0-9, and + = , . @ _ - (hyphen)
Provide user access to the AWS Management Console - optional
If you're providing console access to a person, it's a best practice to manage their access in IAM Identity Center.
Are you providing console access to a person?
User type
Specify a user in Identity Center - Recommended
We recommend that you use Identity Center to provide console access to a person. With Identity Center, you can centrally manage user access to their AWS accounts and cloud applications.
I want to create an IAM user
We recommend that you create IAM users only if you need to enable programmatic access through access keys, service-specific credentials for AWS CodeCommit or Amazon Keyspaces, or a backup credential for emergency account access.
If you are creating programmatic access through access keys or service-specific credentials for AWS CodeCommit or Amazon Keyspaces, you can generate them after you create this IAM user. Learn more
Cancel
Next
CloudShell
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences
Search
ENG
IN
13:22
13-03-2024

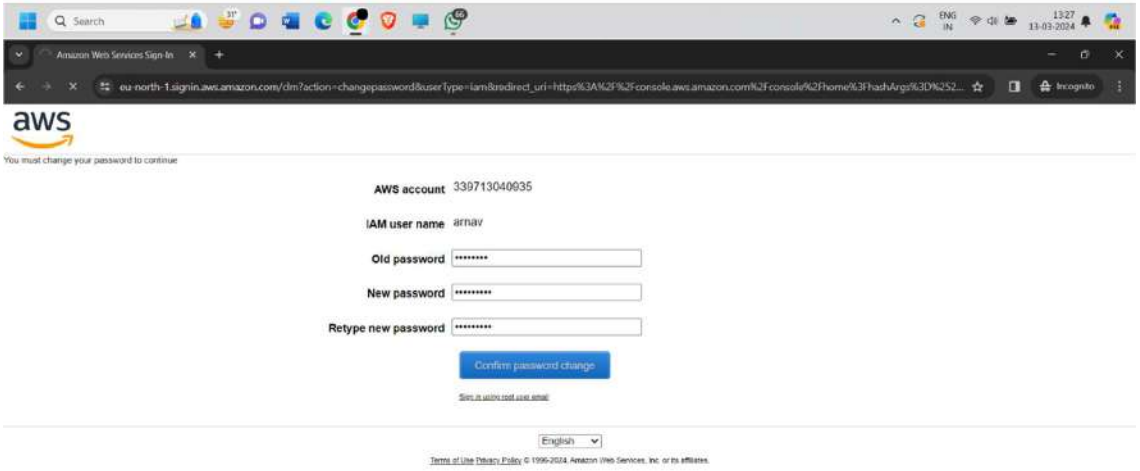

Create user | IAM | Global
us-east-1.console.aws.amazon.com/iam/home?region=ap-south-1#/users/create
Services
Search
[Alt+S]
Global
ArnavM
IAM > Users > Create user
Step 1
Specify user details
Step 2
Set permissions
Step 3
Review and create
Step 4
Retrieve password
Set permissions
Add user to an existing group or create a new one. Using groups is a best-practice way to manage user's permissions by job functions. Learn more
Permissions options
Add user to group
Add user to an existing group, or create a new group. We recommend using groups to manage user permissions by job function.
Copy permissions
Copy all group memberships, attached managed policies, and inline policies from an existing user.
Attach policies directly
Attach a managed policy directly to a user. As a best practice, we recommend attaching policies to a group instead. Then, add the user to the appropriate group.
Get started with groups
Create a group and select policies to attach to the group. We recommend using groups to manage user permissions by job function, AWS service access, or custom permissions. Learn more
Create group
Set permissions boundary - optional
Cancel
Previous
Next
CloudShell
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences


Create user | IAM | Global
us-east-1.console.aws.amazon.com/iam/home?region=ap-south-1#/users/create
Services
Search
[Alt+S]
Global
ArnavM
Review your choices. After you create the user, you can view and download the autogenerated password, if enabled.
Step 2
Set permissions
Step 3
Review and create
Step 4
Retrieve password
User details
User name
arnav
Console password type
Autogenerated
Require password reset
Yes
Permissions summary
Name
Type
Used as
IAMUserChangePassword
AWS managed
Permissions policy
Tags - optional
Tags are key-value pairs you can add to AWS resources to help identify, organize, or search for resources. Choose any tags you want to associate with this user.
No tags associated with the resource.
Add new tag
You can add up to 50 more tags.
Cancel
Previous
Create user
CloudShell
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences

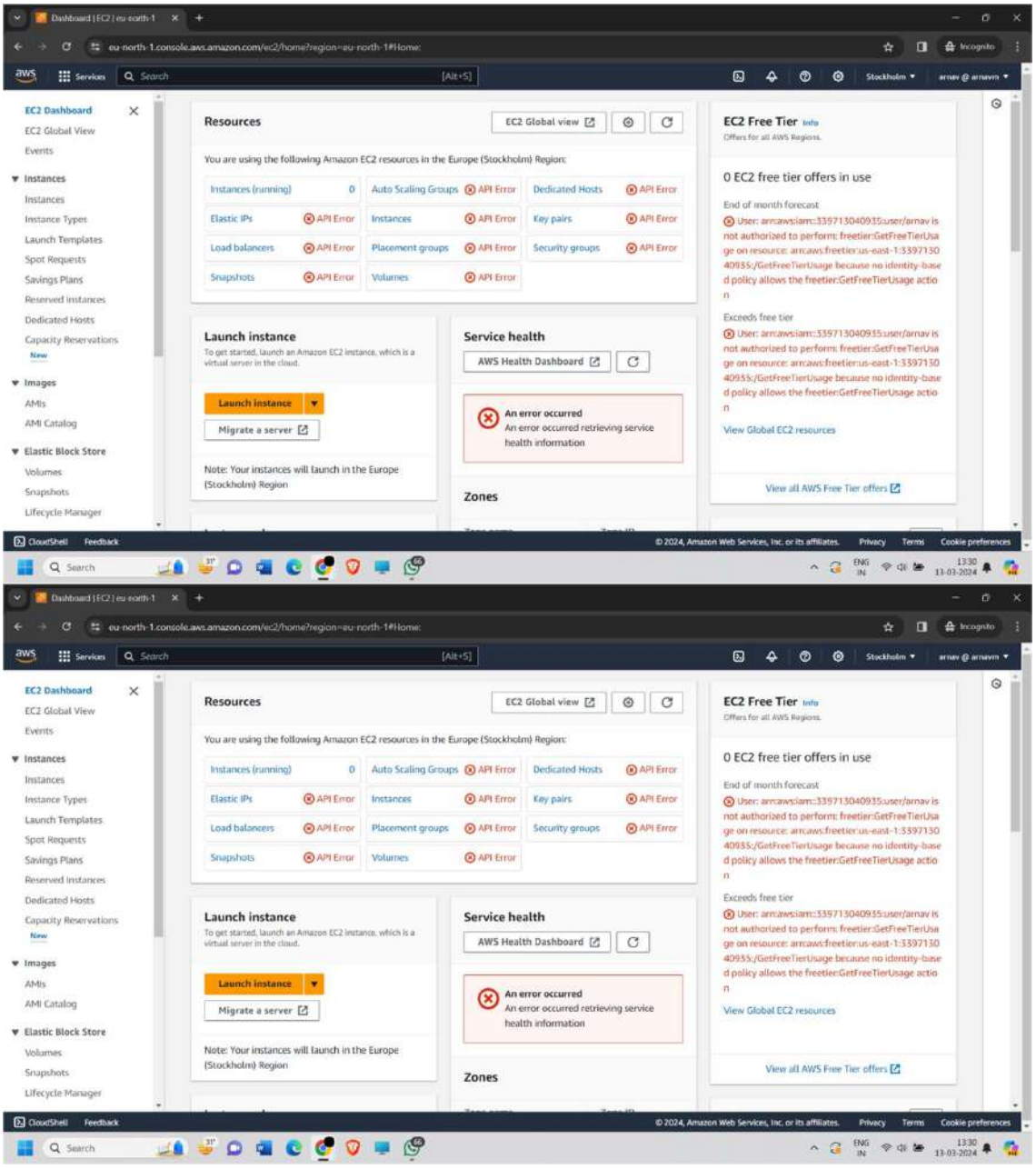

Create user | IAM | Global
us-east-1.console.aws.amazon.com/iam/home?region=ap-south-1#/users/create
Services
Search
[Alt+S]
Global
AnavM
User created successfully
You can view and download the user's password and email instructions for signing in to the AWS Management Console.
View user
IAM > Users > Create user
Step 1
Specify user details
Step 2
Set permissions
Step 3
Review and create
Step 4
Retrieve password
Retrieve password
You can view and download the user's password below or email users instructions for signing in to the AWS Management Console. This is the only time you can view and download this password.
Console sign-in details
Email sign-in instructions
Console sign-in URL
https://arnavm.signin.aws.amazon.com/console
User name
arnav
Console password
Show
Cancel
Download .csv file
Return to users list
CloudShell
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences
Search
ENG
IN
1325
13-03-2024


Amazon Web Services Sign-In
eu-north-1.signin.aws.amazon.com/oauth?client_id=arn%3Aaws%3Asignin%3A%3A%3Aconsole%2Fcanvas&code_challenge=fWEPCsUB6xbbfnqbUiydydatCnISCwNNL2bfjfhsBIA&code_c...
Incognito
aws
Sign in as IAM user
Account ID (12 digits) or account alias
arnavm
IAM user name
Password
Remember this account
Sign in
Sign in using root user email
Forgot password?
Amazon Lightsail
Lightsail is the easiest way to get started on AWS
Learn more »
English
Terms of Use Privacy Policy © 1996-2024, Amazon Web Services, Inc. or its affiliates.
Search
ENG
IN
1327
13-03-2024

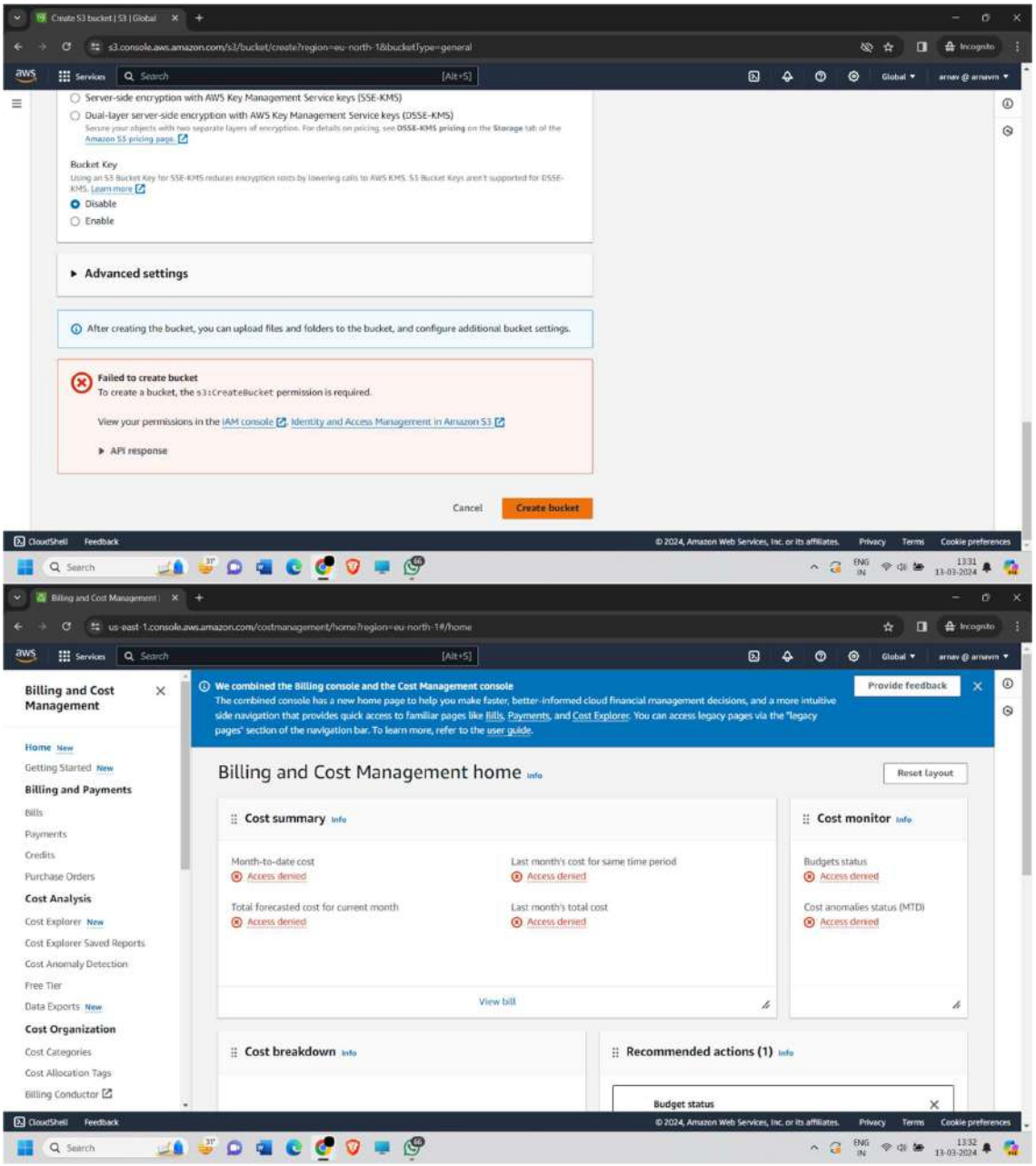

Amazon Web Services Sign-In
eu-north-1.signin.aws.amazon.com/oauth?client_id=arn%3Aaws%3Asignin%3A%3A%3Aconsole%2Fcanvas&code_challenge=fWEPCsU8f6xbbfnqbIJiydydatCnISCwNNL2bfjfhsBA&code_c...
Incognito
aws
Sign in as IAM user
Account ID (12 digits) or account alias
arnavm
IAM user name
arnav
Password
Remember this account
Sign in
Sign in using root user email
Forgot password?
Amazon Lightsail
Lightsail is the easiest way to get started on AWS
Learn more »
English
Terms of Use Privacy Policy © 1996-2024, Amazon Web Services, Inc. or its affiliates.


Search
ENG IN
13:27 13-03-2024
Amazon Web Services Sign-In
eu-north-1.signin.aws.amazon.com/cfm?action=changepassword&userType=iam&redirect_uri=https%3A%2F%2Fconsole.aws.amazon.com%2Fconsole%2Fhome%3FhashArgs%3D%252...
Incognito
aws
You must change your password to continue
AWS account 339713040935
IAM user name arnav
Old password
New password
Retype new password
Confirm password change
Sign in using root user email
English
Terms of Use Privacy Policy © 1996-2024, Amazon Web Services, Inc. or its affiliates.

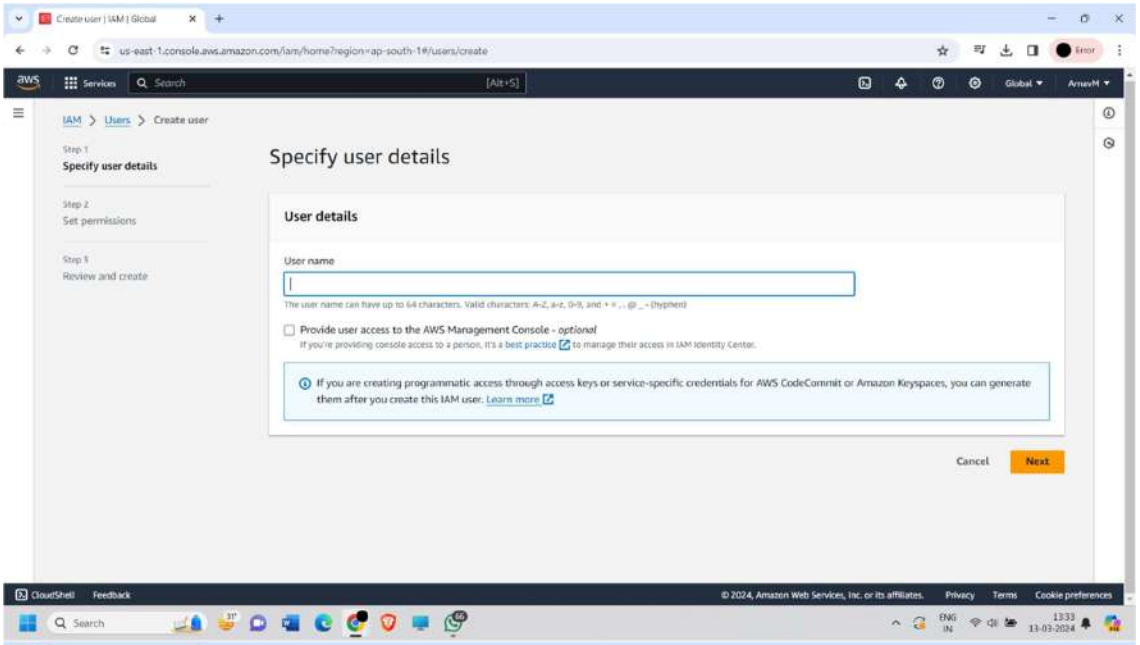

https://eu-north-1.signin.aws.amazon.com/cfm?action=changepassword&userType=iam&redirect_uri=https%3A%2F%2Fconsole.aws.amazon.com%2Fconsole%2Fhome%3FhashArgs%3D%2523%26isauthcode%3Dtrue%26state%3DhashArgsFromTB_eu-north-1_25f9668bbf6572c6c...
Search
ENG IN
13:28 13-03-2024

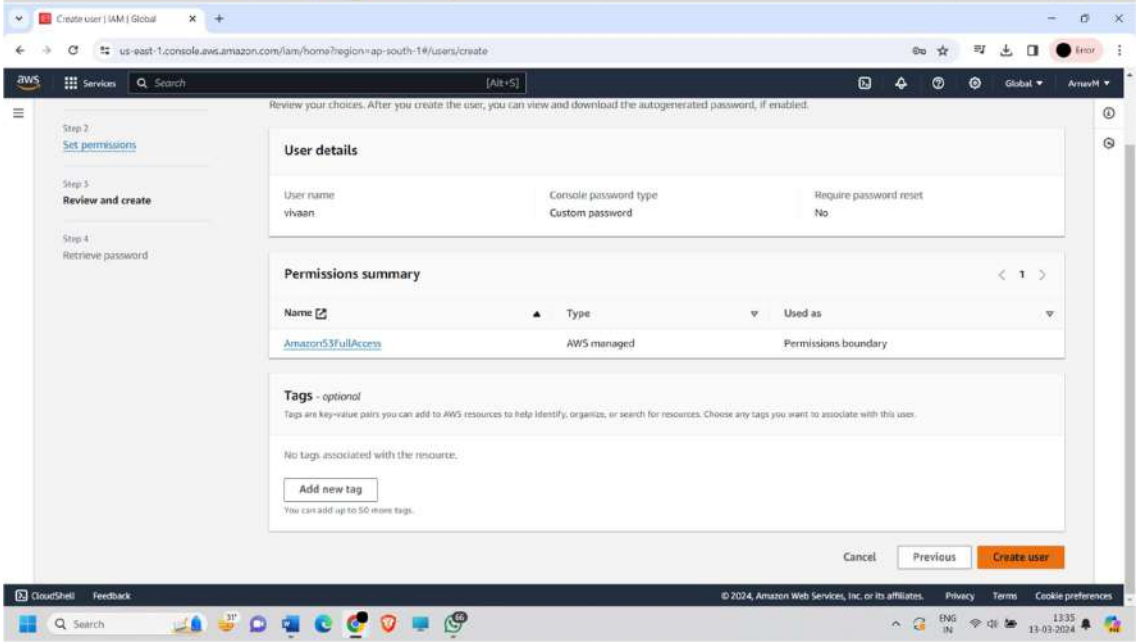


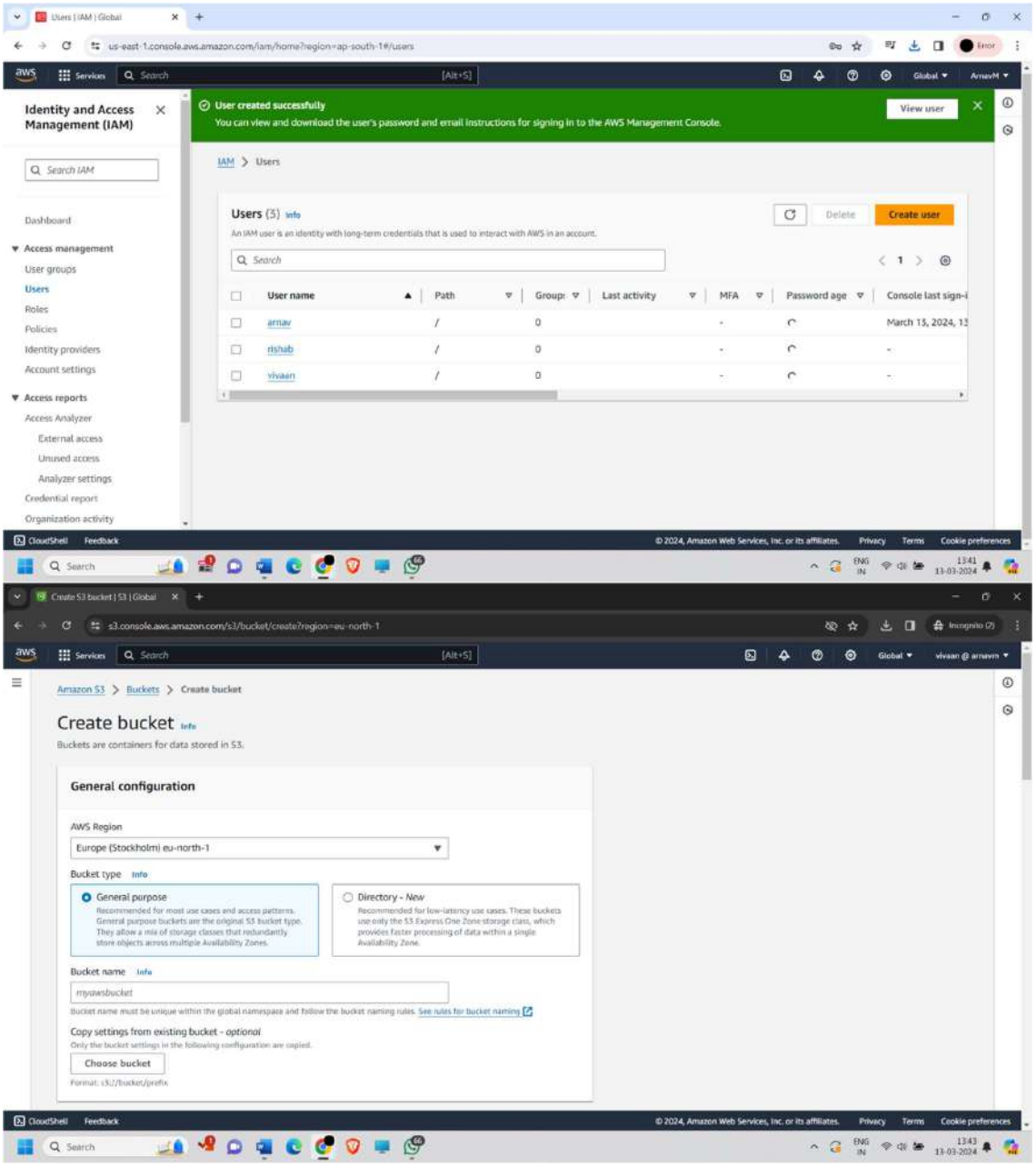
Dashboard | EC2 | eu-north-1
eu-north-1.console.aws.amazon.com/ec2/home?region=eu-north-1#Home:
Incognito
Services
Search
[Alt+S]
Stockholm
arnav @ arnavm
EC2 Dashboard
EC2 Global View
Events
Instances
Instances
Instance Types
Launch Templates
Spot Requests
Savings Plans
Reserved Instances
Dedicated Hosts
Capacity Reservations
New
Images
AMIs
AMI Catalog
Elastic Block Store
Volumes
Snapshots
Lifecycle Manager
Resources
EC2 Global view
You are using the following Amazon EC2 resources in the Europe (Stockholm) Region:
Instances (running) 0
Auto Scaling Groups API Error
Dedicated Hosts API Error
Elastic IPs API Error
Instances API Error
Key pairs API Error
Load balancers API Error
Placement groups API Error
Security groups API Error
Snapshots API Error
Volumes API Error
Launch instance
To get started, launch an Amazon EC2 instance, which is a virtual server in the cloud.
Launch instance
Migrate a server
Note: Your instances will launch in the Europe (Stockholm) Region
Service health
AWS Health Dashboard
An error occurred
An error occurred retrieving service health information
Zones
EC2 Free Tier Info
Offers for all AWS Regions.
0 EC2 free tier offers in use
End of month forecast
User: arn:aws:iam::339713040935:user/arnav is not authorized to perform: freetier:GetFreeTierUsage on resource: arn:aws:freetier:us-east-1:339713040935:/GetFreeTierUsage because no identity-based policy allows the freetier:GetFreeTierUsage action
Exceeds free tier
User: arn:aws:iam::339713040935:user/arnav is not authorized to perform: freetier:GetFreeTierUsage on resource: arn:aws:freetier:us-east-1:339713040935:/GetFreeTierUsage because no identity-based policy allows the freetier:GetFreeTierUsage action
View Global EC2 resources
View all AWS Free Tier offers
CloudShell
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences
Search
ENG
IN
1330
13-03-2024


Create S3 bucket | S3 | Global
s3.console.aws.amazon.com/s3/bucket/create?region=eu-north-1&bucketType=general
Server-side encryption with AWS Key Management Service keys (SSE-KMS)
Dual-layer server-side encryption with AWS Key Management Service keys (DSSE-KMS)
Bucket Key
Disable
Enable
Advanced settings
After creating the bucket, you can upload files and folders to the bucket, and configure additional bucket settings.
Failed to create bucket
To create a bucket, the s3:CreateBucket permission is required.
View your permissions in the IAM console. Identity and Access Management in Amazon S3
API response
Cancel
Create bucket
CloudShell
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences

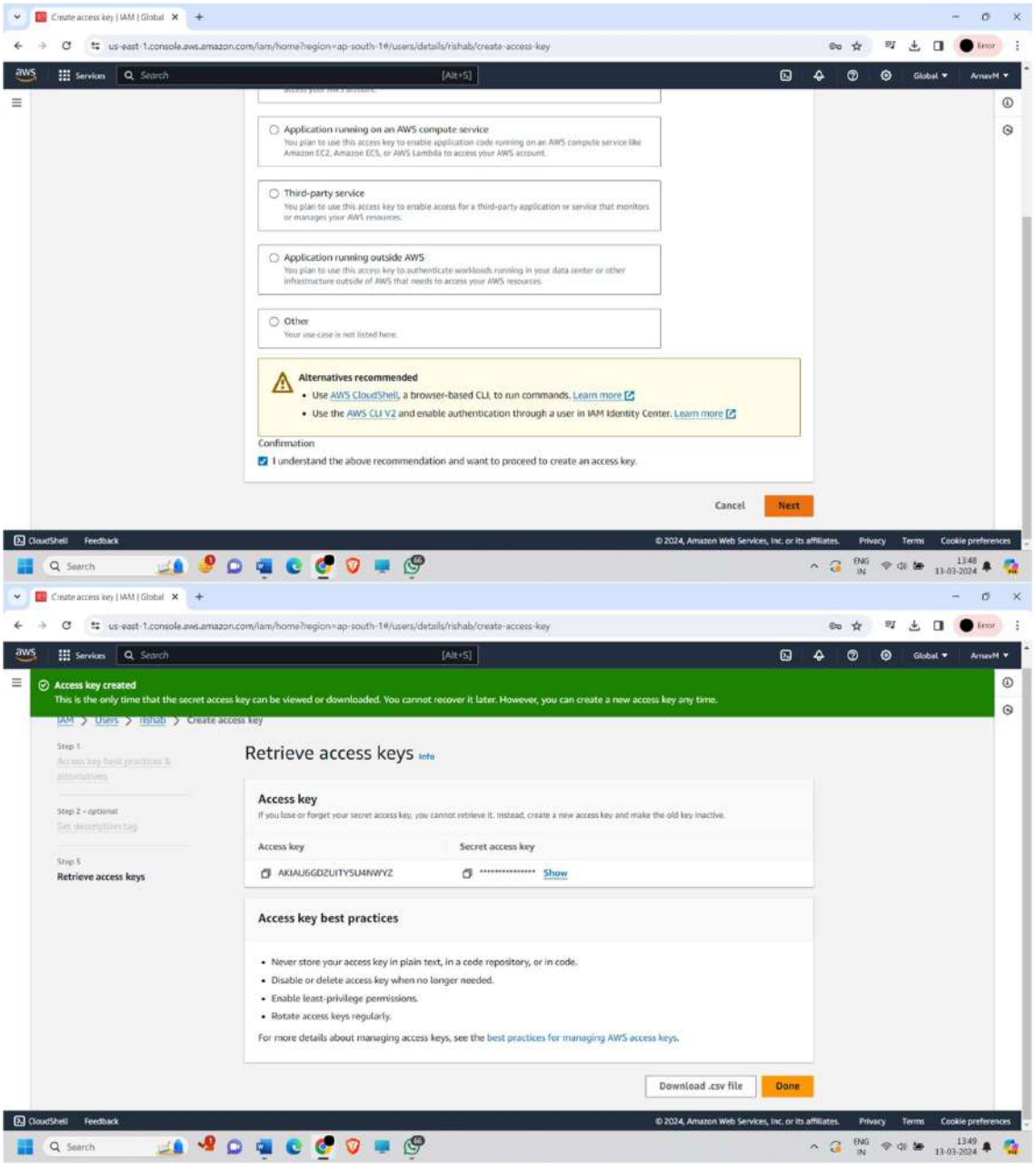
Billing and Cost Management
us-east-1.console.aws.amazon.com/costmanagement/home?region=eu-north-1#/home
Billing and Cost Management
Home New
Getting Started New
Billing and Payments
Bills
Payments
Credits
Purchase Orders
Cost Analysis
Cost Explorer New
Cost Explorer Saved Reports
Cost Anomaly Detection
Free Tier
Data Exports New
Cost Organization
Cost Categories
Cost Allocation Tags
Billing Conductor
We combined the Billing console and the Cost Management console
The combined console has a new home page to help you make faster, better-informed cloud financial management decisions, and a more intuitive side navigation that provides quick access to familiar pages like Bills, Payments, and Cost Explorer. You can access legacy pages via the "legacy pages" section of the navigation bar. To learn more, refer to the user guide.
Provide feedback
Billing and Cost Management home Info
Reset layout
Cost summary Info
Month-to-date cost
Access denied
Last month's cost for same time period
Access denied
Total forecasted cost for current month
Access denied
Last month's total cost
Access denied
View bill
Cost monitor Info
Budgets status
Access denied
Cost anomalies status (MTD)
Access denied
Cost breakdown Info
Recommended actions (1) Info
Budget status
CloudShell
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences


Create user | IAM | Global
us-east-1.console.aws.amazon.com/iam/home?region=ap-south-1#/users/create
Services
Search
[Alt+S]
Global
ArnavM
IAM > Users > Create user
Step 1
Specify user details
Step 2
Set permissions
Step 3
Review and create
Specify user details
User details
User name
The user name can have up to 64 characters. Valid characters: A-Z, a-z, 0-9, and + = , . @ _ - (hyphen)
Provide user access to the AWS Management Console - optional
If you're providing console access to a person, it's a best practice to manage their access in IAM Identity Center.
If you are creating programmatic access through access keys or service-specific credentials for AWS CodeCommit or Amazon Keyspaces, you can generate them after you create this IAM user. Learn more
Cancel
Next
CloudShell
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences
Search
ENG
IN
1333
13-03-2024

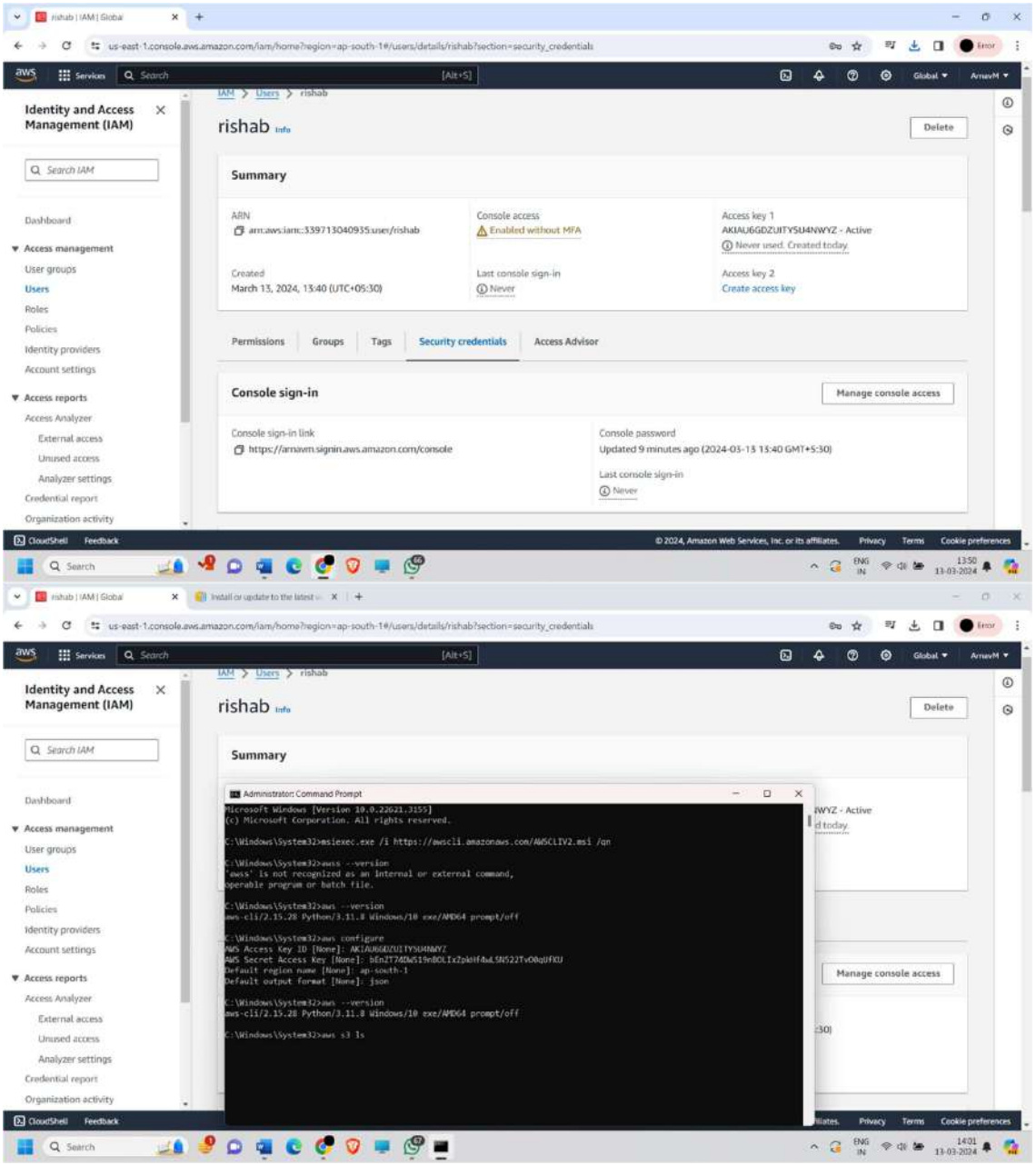

Create user | IAM | Global
us-east-1.console.aws.amazon.com/iam/home?region=ap-south-1#/users/create
Services
Search
[Alt+S]
Global
ArnavM
Review your choices. After you create the user, you can view and download the autogenerated password, if enabled.
Step 2
Set permissions
Step 3
Review and create
Step 4
Retrieve password
User details
User name
vivaan
Console password type
Custom password
Require password reset
No
Permissions summary
Name
Type
Used as
Amazon53FullAccess
AWS managed
Permissions boundary
Tags - optional
Tags are key-value pairs you can add to AWS resources to help identify, organize, or search for resources. Choose any tags you want to associate with this user.
No tags associated with the resource.
Add new tag
You can add up to 50 more tags.
Cancel
Previous
Create user
CloudShell
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences
Search
ENG
IN
1335
13-03-2024


Users | IAM | Global
us-east-1.console.aws.amazon.com/iam/home?region=ap-south-1#/users
Identity and Access Management (IAM)
Search IAM
Dashboard
Access management
User groups
Users
Roles
Policies
Identity providers
Account settings
Access reports
Access Analyzer
External access
Unused access
Analyzer settings
Credential report
Organization activity
User created successfully
You can view and download the user's password and email instructions for signing in to the AWS Management Console.
View user
IAM > Users
Users (3) Info
An IAM user is an identity with long-term credentials that is used to interact with AWS in an account.
Delete
Create user
Search
User name
Path
Group:
Last activity
MFA
Password age
Console last sign-i
arnav
rishab
vivaan
March 13, 2024, 1
CloudShell
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences

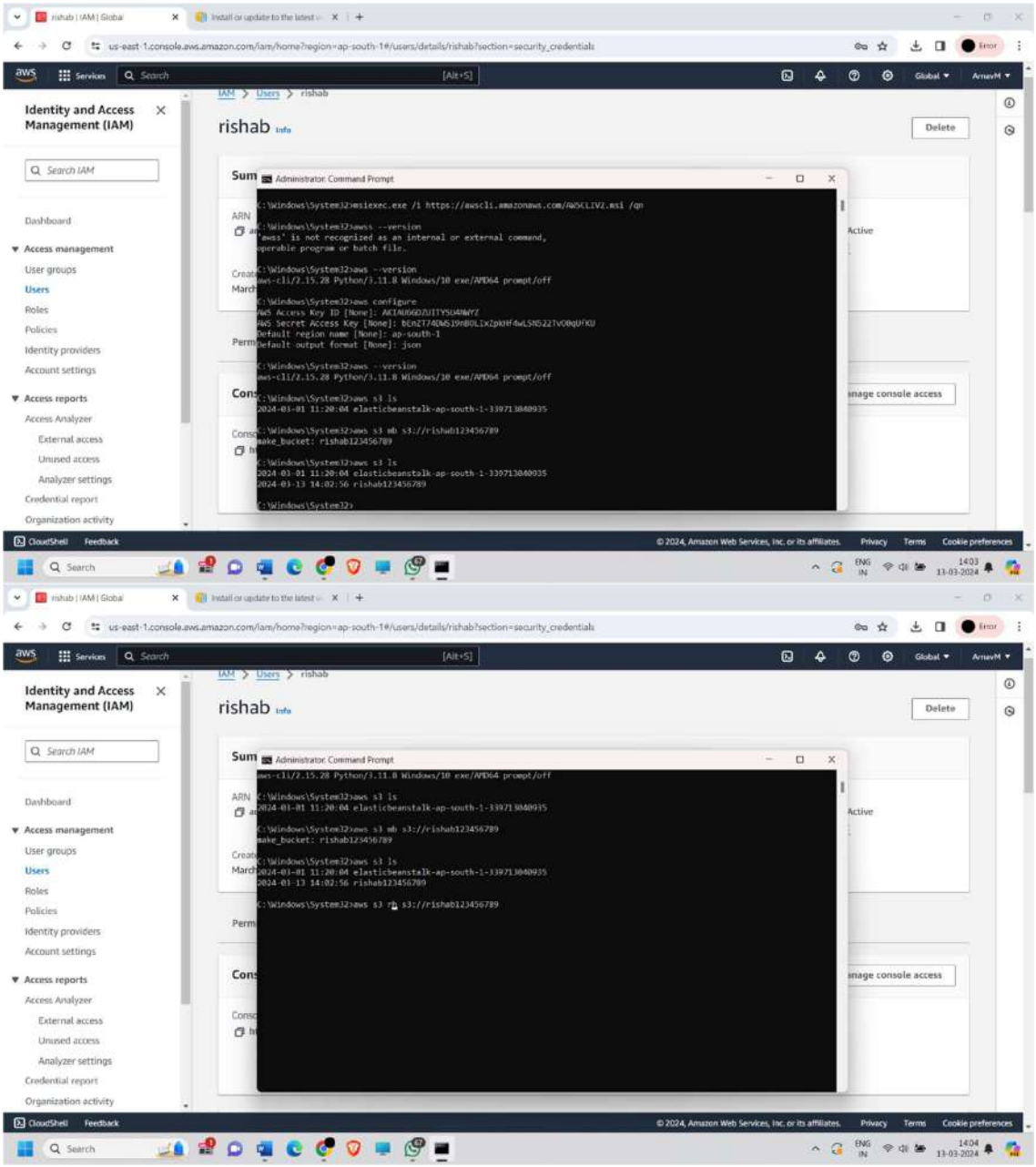
Create S3 bucket | S3 | Global
s3.console.aws.amazon.com/s3/bucket/create?region=eu-north-1
Amazon S3 > Buckets > Create bucket
Create bucket Info
Buckets are containers for data stored in S3.
General configuration
AWS Region
Europe (Stockholm) eu-north-1
Bucket type Info
General purpose
Recommended for most use cases and access patterns. General purpose buckets are the original S3 bucket type. They allow a mix of storage classes that redundantly store objects across multiple Availability Zones.
Directory - New
Recommended for low-latency use cases. These buckets use only the S3 Express One Zone storage class, which provides faster processing of data within a single Availability Zone.
Bucket name Info
myawsbucket
Bucket name must be unique within the global namespace and follow the bucket naming rules. See rules for bucket naming
Copy settings from existing bucket - optional
Only the bucket settings in the following configuration are copied.
Choose bucket
Format: s3://bucket/prefix
CloudShell
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences


Create access key | IAM | Global
us-east-1.console.aws.amazon.com/iam/home?region=ap-south-1#/users/details/rishab/create-access-key
Application running on an AWS compute service
You plan to use this access key to enable application code running on an AWS compute service like Amazon EC2, Amazon ECS, or AWS Lambda to access your AWS account.
Third-party service
You plan to use this access key to enable access for a third-party application or service that monitors or manages your AWS resources.
Application running outside AWS
You plan to use this access key to authenticate workloads running in your data center or other infrastructure outside of AWS that needs to access your AWS resources.
Other
Your use case is not listed here.
Alternatives recommended
Use AWS CloudShell, a browser-based CLI, to run commands. Learn more
Use the AWS CLI V2 and enable authentication through a user in IAM Identity Center. Learn more
Confirmation
I understand the above recommendation and want to proceed to create an access key.
Cancel
Next
CloudShell
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences

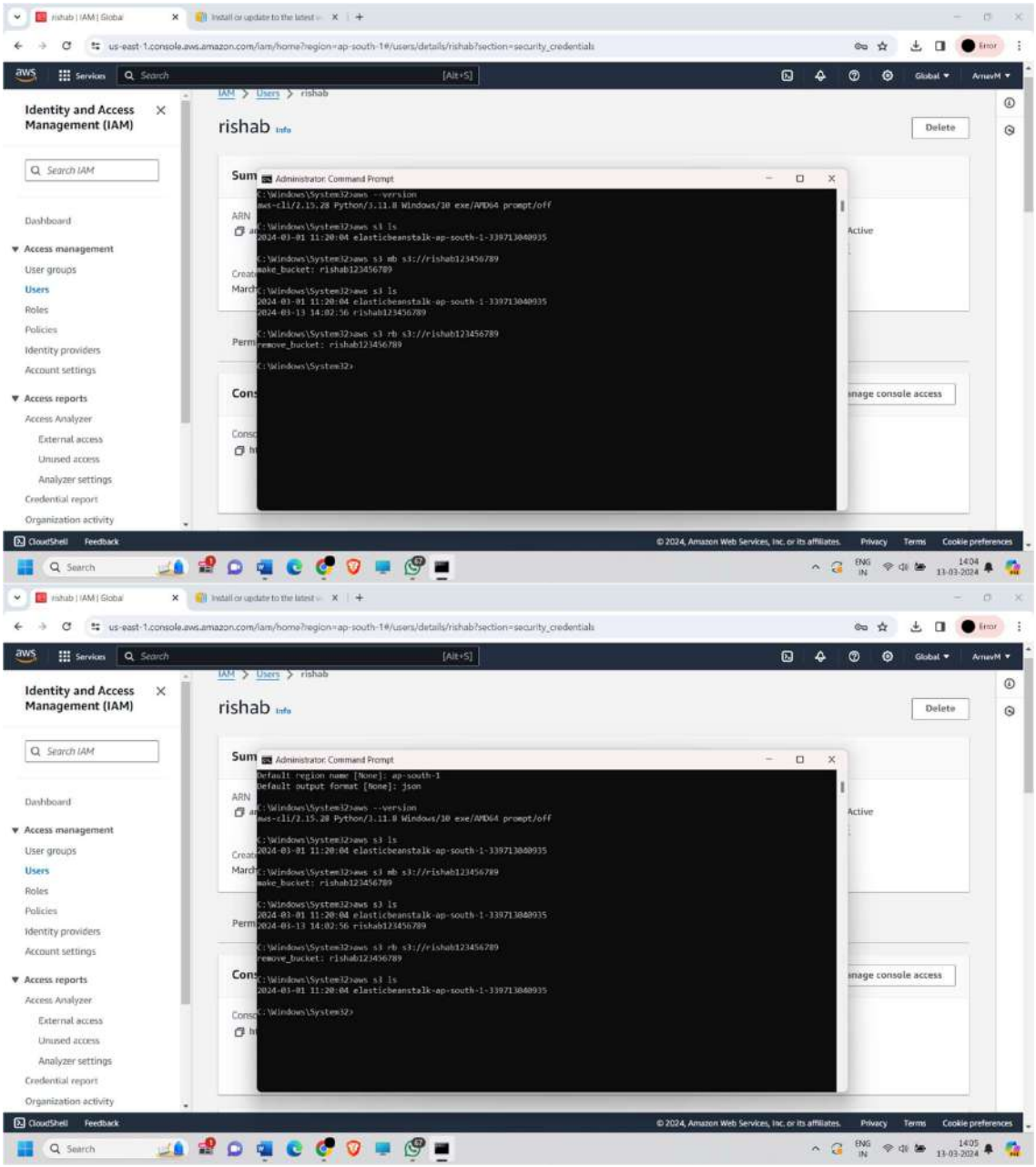
Create access key | IAM | Global
us-east-1.console.aws.amazon.com/iam/home?region=ap-south-1#/users/details/rishab/create-access-key
Access key created
This is the only time that the secret access key can be viewed or downloaded. You cannot recover it later. However, you can create a new access key any time.
IAM > Users > rishab > Create access key
Step 1
Step 2 - optional
Step 3
Retrieve access keys
Retrieve access keys Info
Access key
If you lose or forget your secret access key, you cannot retrieve it. Instead, create a new access key and make the old key inactive.
Access key
Secret access key
AKIAU6GDZUITY5U4NWYZ
Show
Access key best practices
Never store your access key in plain text, in a code repository, or in code.
Disable or delete access key when no longer needed.
Enable least-privilege permissions.
Rotate access keys regularly.
For more details about managing access keys, see the best practices for managing AWS access keys.
Download .csv file
Done
CloudShell
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences


rishab | IAM | Global
us-east-1.console.aws.amazon.com/iam/home?region=ap-south-1#/users/details/rishab?section=security_credentials
Services
Search
[Alt+S]
Global
ArnavM
Identity and Access Management (IAM)
Search IAM
Dashboard
Access management
User groups
Users
Roles
Policies
Identity providers
Account settings
Access reports
Access Analyzer
External access
Unused access
Analyzer settings
Credential report
Organization activity
IAM > Users > rishab
rishab Info
Delete
Summary
ARN
arn:aws:iam::339713040935:user/rishab
Console access
Enabled without MFA
Access key 1
AKIAU6GDZUITYSU4NWYZ - Active
Never used. Created today.
Created
March 13, 2024, 13:40 (UTC+05:30)
Last console sign-in
Never
Access key 2
Create access key
Permissions
Groups
Tags
Security credentials
Access Advisor
Console sign-in
Manage console access
Console sign-in link
https://arnavm.signin.aws.amazon.com/console
Console password
Updated 9 minutes ago (2024-03-13 13:40 GMT+5:30)
Last console sign-in
Never
CloudShell
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences
Search
ENG
IN
13:50
13-03-2024

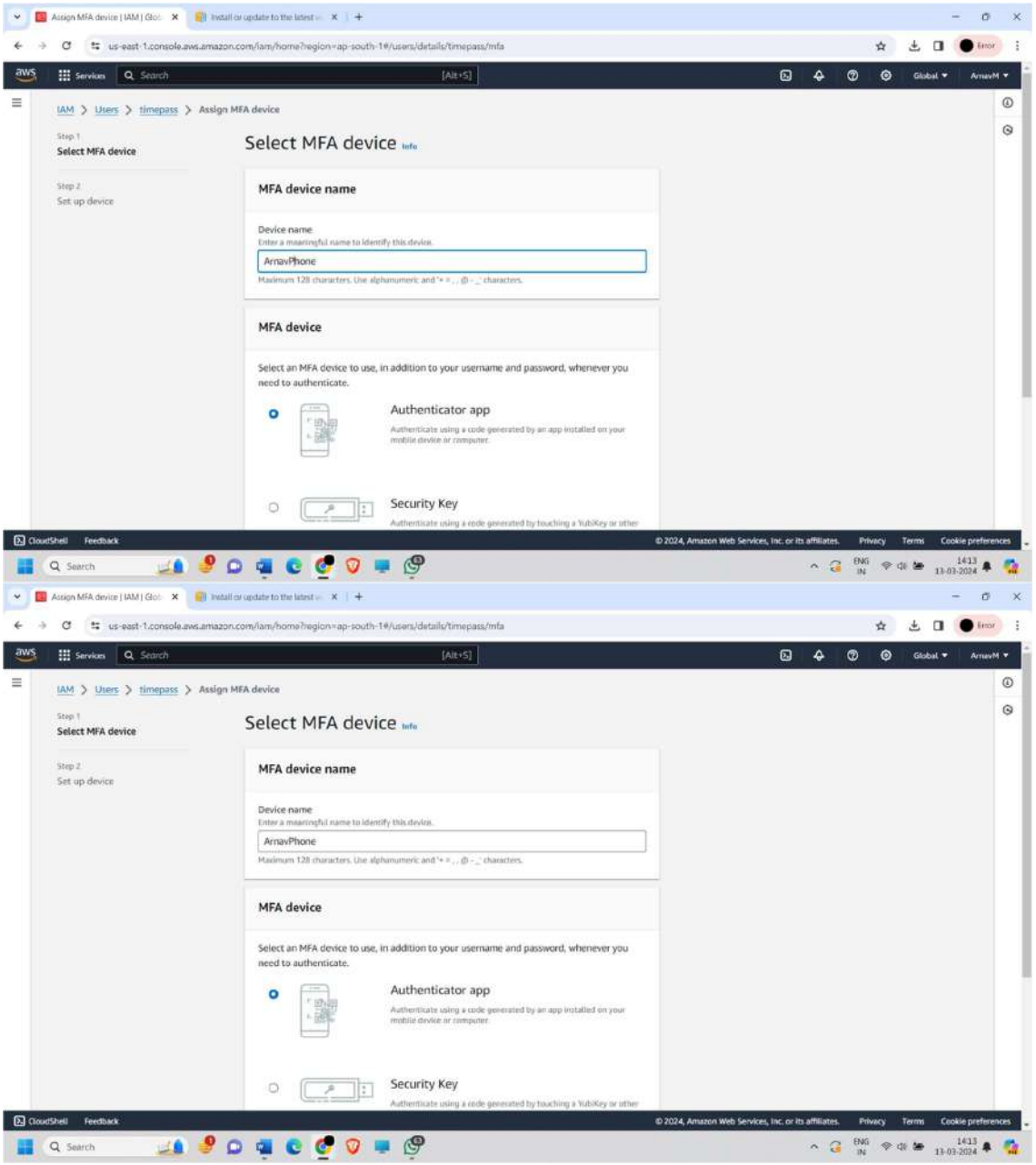
rishab | IAM | Global
Install or update to the latest
us-east-1.console.aws.amazon.com/iam/home?region=ap-south-1#/users/details/rishab?section=security_credentials
Identity and Access Management (IAM)
rishab Info
Delete
Summary
Administrator: Command Prompt
Microsoft Windows [Version 10.0.22621.3155]
(c) Microsoft Corporation. All rights reserved.
C:\Windows\System32>msiexec.exe /i https://awscli.amazonaws.com/AWSCLIV2.msi /qn
C:\Windows\System32>awss --version
'awss' is not recognized as an internal or external command,
operable program or batch file.
C:\Windows\System32>aws --version
aws-cli/2.15.28 Python/3.11.8 Windows/10 exe/AMD64 prompt/off
C:\Windows\System32>aws configure
AWS Access Key ID [None]: AKIAU6GDZUITYSU4NWYZ
AWS Secret Access Key [None]: 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
Default region name [None]: ap-south-1
Default output format [None]: json
C:\Windows\System32>aws --version
aws-cli/2.15.28 Python/3.11.8 Windows/10 exe/AMD64 prompt/off
C:\Windows\System32>aws s3 ls
Manage console access
CloudShell
Feedback
Privacy
Terms
Cookie preferences
Search
ENG
IN
14:01
13-03-2024


Identity and Access Management (IAM)
rishab
Delete
Administrator: Command Prompt
C:\Windows\System32>msiexec.exe /i https://awscli.amazonaws.com/AWSCLIV2.msi /qn
C:\Windows\System32>aws --version
'aws' is not recognized as an internal or external command,
operable program or batch file.
C:\Windows\System32>aws --version
aws-cli/2.15.28 Python/3.11.8 Windows/10 exe/AMD64 prompt/off
C:\Windows\System32>aws configure
AWS Access Key ID [None]: AKIAU66GDJUITYSU4NWY2
AWS Secret Access Key [None]: bCnZT74DWSI9nBOLIxZpkHf4wLSN5227vO0qUfKU
Default region name [None]: ap-south-1
Default output format [None]: json
C:\Windows\System32>aws --version
aws-cli/2.15.28 Python/3.11.8 Windows/10 exe/AMD64 prompt/off
C:\Windows\System32>aws s3 ls
2024-03-01 11:20:04 elasticbeanstalk-ap-south-1-339713040935
C:\Windows\System32>aws s3 mb s3://rishab123456789
make_bucket: rishab123456789
C:\Windows\System32>aws s3 ls
2024-03-01 11:20:04 elasticbeanstalk-ap-south-1-339713040935
2024-03-13 14:02:56 rishab123456789
C:\Windows\System32>
CloudShell
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences

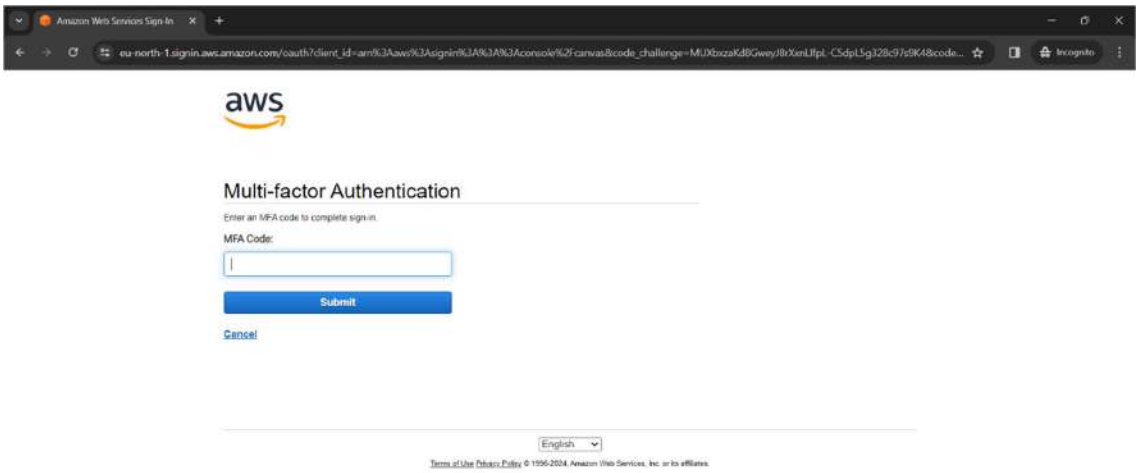
Identity and Access Management (IAM)
rishab
Delete
Administrator: Command Prompt
aws-cli/2.15.28 Python/3.11.8 Windows/10 exe/AMD64 prompt/off
C:\Windows\System32>aws s3 ls
2024-03-01 11:20:04 elasticbeanstalk-ap-south-1-339713040935
C:\Windows\System32>aws s3 mb s3://rishab123456789
make_bucket: rishab123456789
C:\Windows\System32>aws s3 ls
2024-03-01 11:20:04 elasticbeanstalk-ap-south-1-339713040935
2024-03-13 14:02:56 rishab123456789
C:\Windows\System32>aws s3 rb s3://rishab123456789
CloudShell
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences

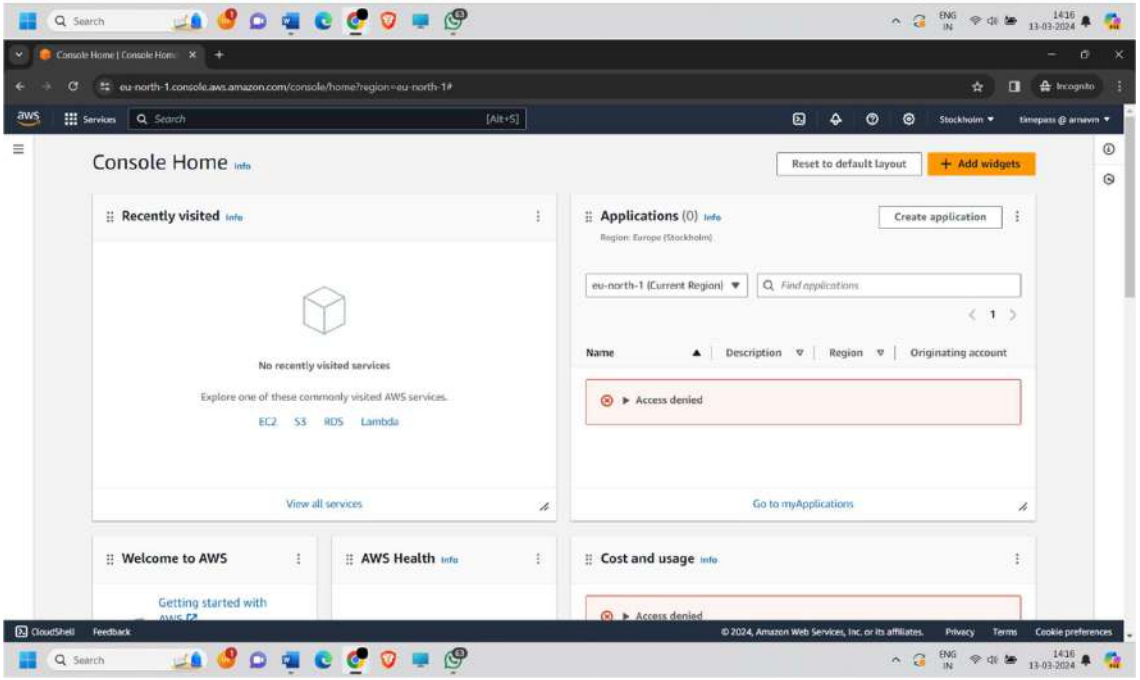

Identity and Access Management (IAM)
rishab
Delete
Administrator: Command Prompt
C:\Windows\System32>aws --version
aws-cli/2.15.28 Python/3.11.8 Windows/10 exe/AMD64 prompt/off
C:\Windows\System32>aws s3 ls
2024-03-01 11:20:04 elasticbeanstalk-ap-south-1-339713040935
C:\Windows\System32>aws s3 mb s3://rishab123456789
make_bucket: rishab123456789
C:\Windows\System32>aws s3 ls
2024-03-01 11:20:04 elasticbeanstalk-ap-south-1-339713040935
2024-03-13 14:02:56 rishab123456789
C:\Windows\System32>aws s3 rb s3://rishab123456789
remove_bucket: rishab123456789
C:\Windows\System32>
CloudShell
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences

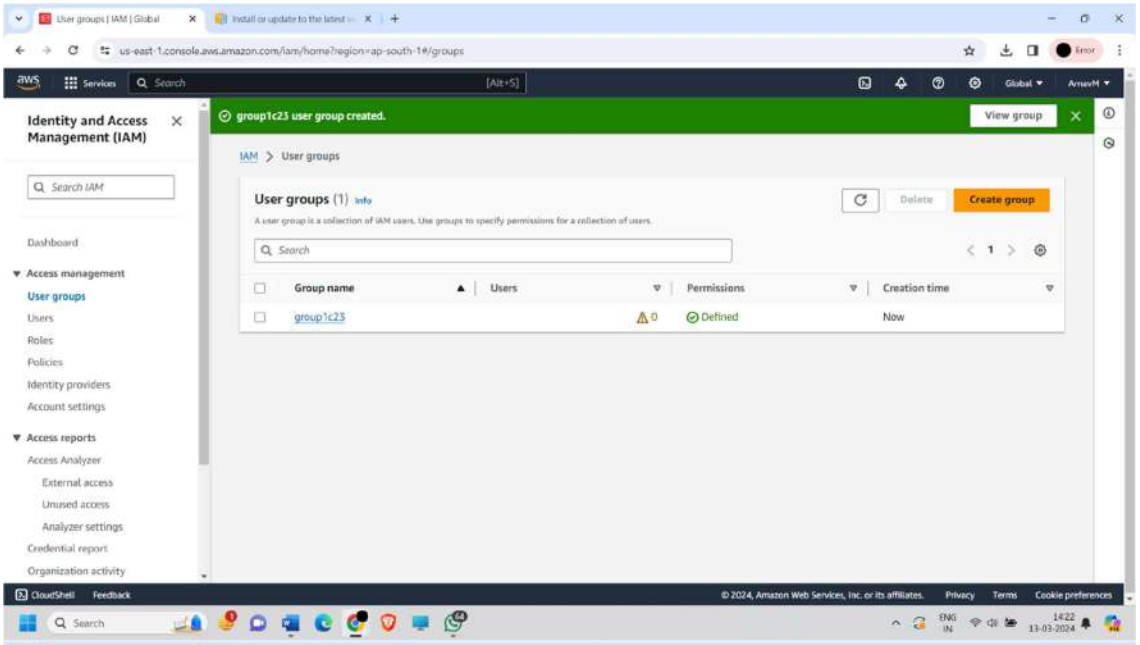
Identity and Access Management (IAM)
rishab
Delete
Administrator: Command Prompt
Default region name [None]: ap-south-1
Default output format [None]: json
C:\Windows\System32>aws --version
aws-cli/2.15.28 Python/3.11.8 Windows/10 exe/AMD64 prompt/off
C:\Windows\System32>aws s3 ls
2024-03-01 11:20:04 elasticbeanstalk-ap-south-1-339713040935
C:\Windows\System32>aws s3 mb s3://rishab123456789
make_bucket: rishab123456789
C:\Windows\System32>aws s3 ls
2024-03-01 11:20:04 elasticbeanstalk-ap-south-1-339713040935
2024-03-13 14:02:56 rishab123456789
C:\Windows\System32>aws s3 rb s3://rishab123456789
remove_bucket: rishab123456789
C:\Windows\System32>aws s3 ls
2024-03-01 11:20:04 elasticbeanstalk-ap-south-1-339713040935
C:\Windows\System32>
CloudShell
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences

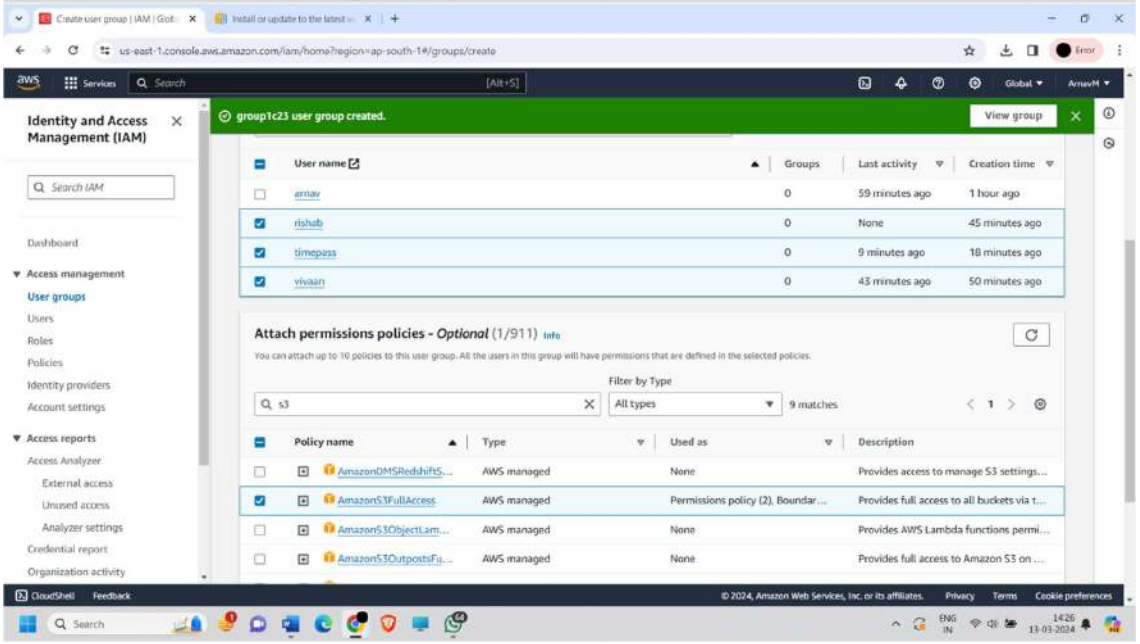


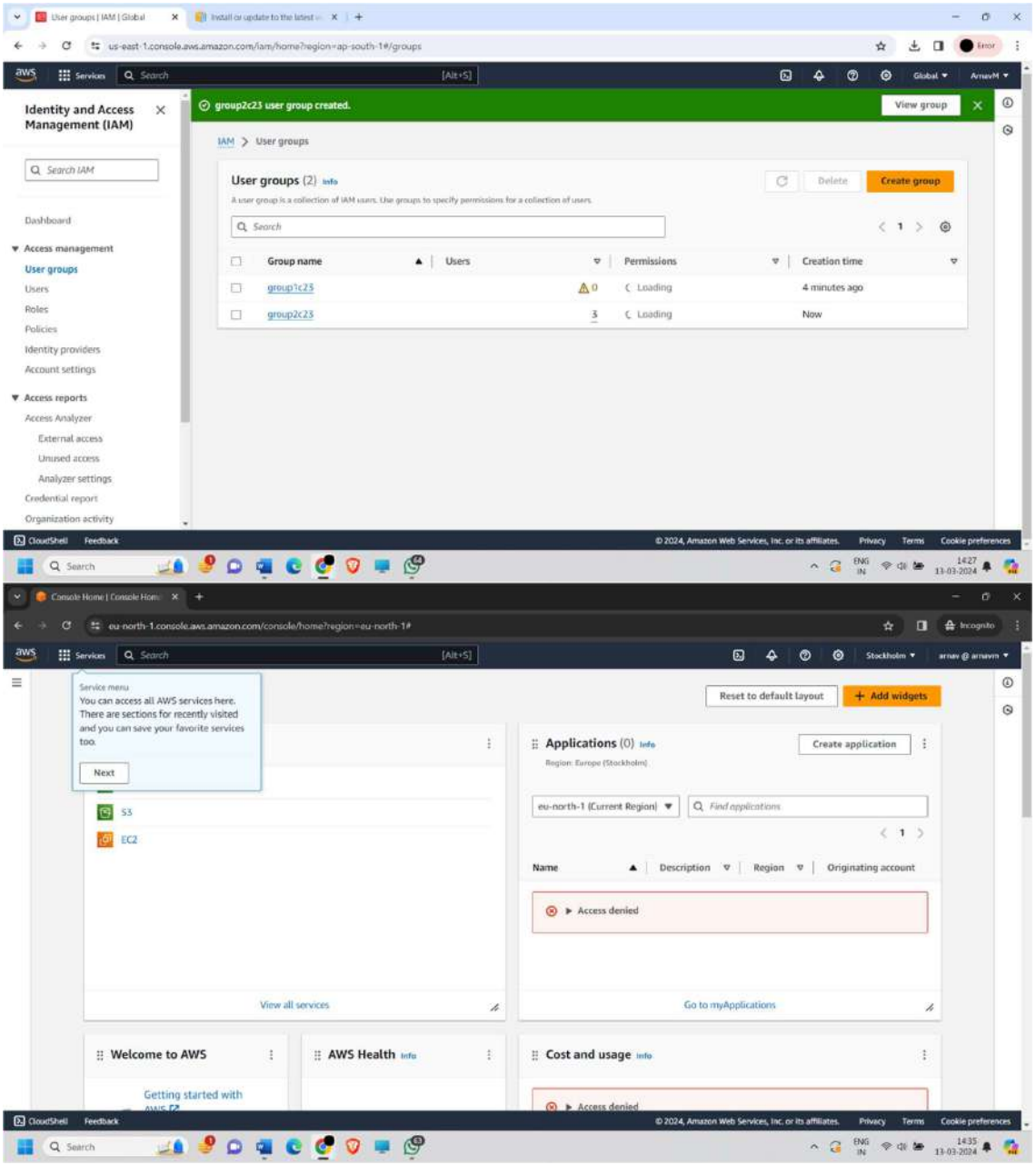
Assign MFA device | IAM | Glob
Install or update to the latest
us-east-1.console.aws.amazon.com/iam/home?region=ap-south-1#/users/details/timepass/mfa
Error
aws
Services
Search
[Alt+S]
Global
ArnavM
IAM > Users > timepass > Assign MFA device
Step 1
Select MFA device
Step 2
Set up device
Select MFA device Info
MFA device name
Device name
Enter a meaningful name to identify this device.
ArnavPhone
Maximum 128 characters. Use alphanumeric and '+ = , . @ - _' characters.
MFA device
Select an MFA device to use, in addition to your username and password, whenever you need to authenticate.
Authenticator app
Authenticate using a code generated by an app installed on your mobile device or computer.
Security Key
Authenticate using a code generated by touching a YubiKey or other
CloudShell
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences
Search
ENG
IN
14:13
13-03-2024

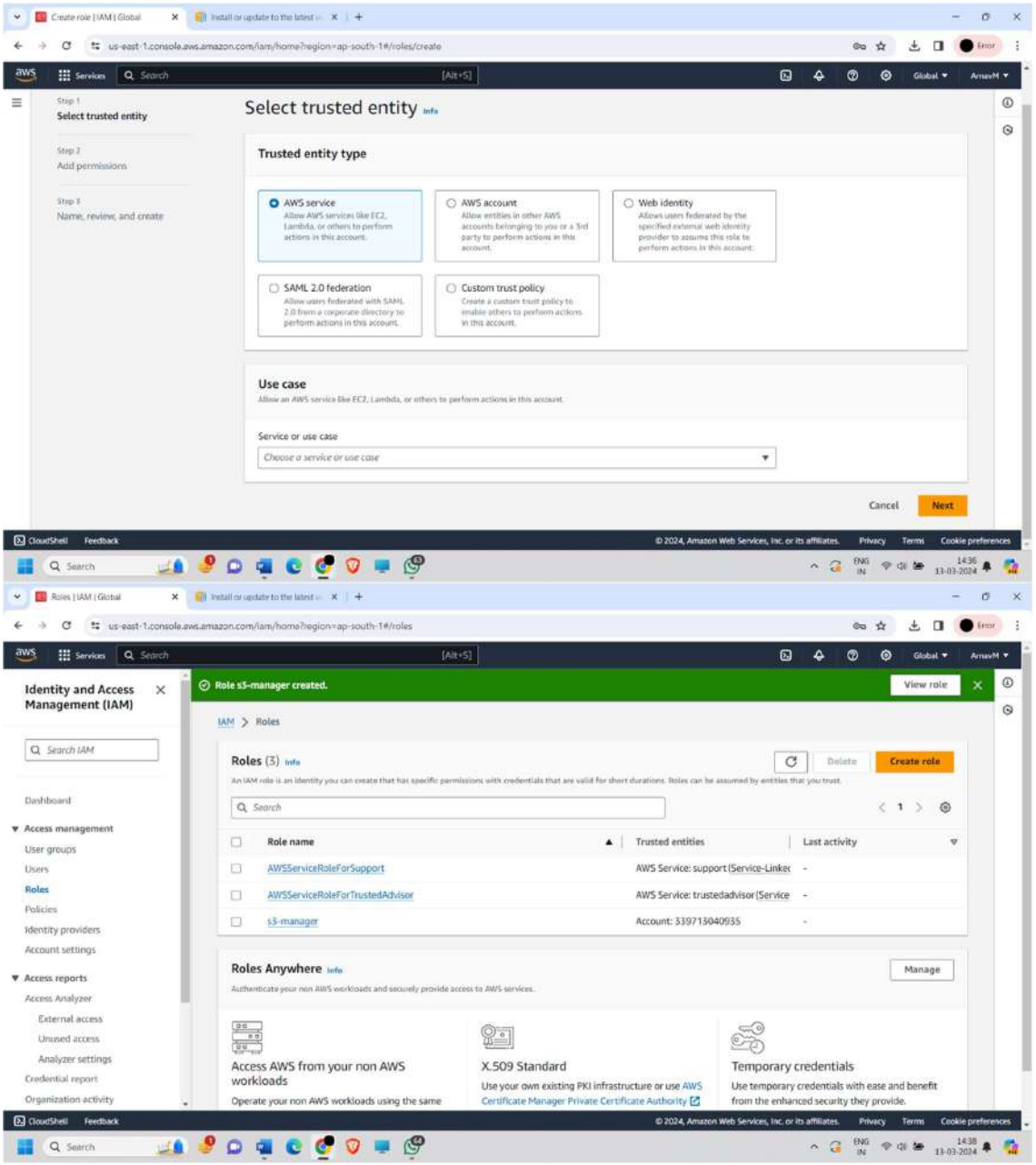

Amazon Web Services Sign-In
eu-north-1.signin.aws.amazon.com/oauth?client_id=arn%3Aaws%3Asignin%3A%3A%3Aconsole%2Fcanvas&code_challenge=MUXbxzaKd8GweyJ8rXenLIfpL-C5dpL5g328c97s9K48&code...
Incognito
aws
Multi-factor Authentication
Enter an MFA code to complete sign-in.
MFA Code:
Submit
Cancel
English
Terms of Use Privacy Policy © 1996-2024, Amazon Web Services, Inc. or its affiliates.


Search
ENG
IN
14:16
13-03-2024
Console Home | Console Home
eu-north-1.console.aws.amazon.com/console/home?region=eu-north-1#
Incognito
aws
Services
Search
[Alt+S]
Stockholm
timepass @ arnavm
Console Home Info
Reset to default layout
+ Add widgets
Recently visited Info
No recently visited services
Explore one of these commonly visited AWS services.
EC2 S3 RDS Lambda
View all services
Applications (0) Info
Region: Europe (Stockholm)
Create application
eu-north-1 (Current Region)
Find applications
1
Name
Description
Region
Originating account
Access denied
Go to myApplications
Welcome to AWS
Getting started with
AWS
AWS Health Info
Cost and usage Info
Access denied
CloudShell
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences
Search
ENG
IN
14:16
13-03-2024

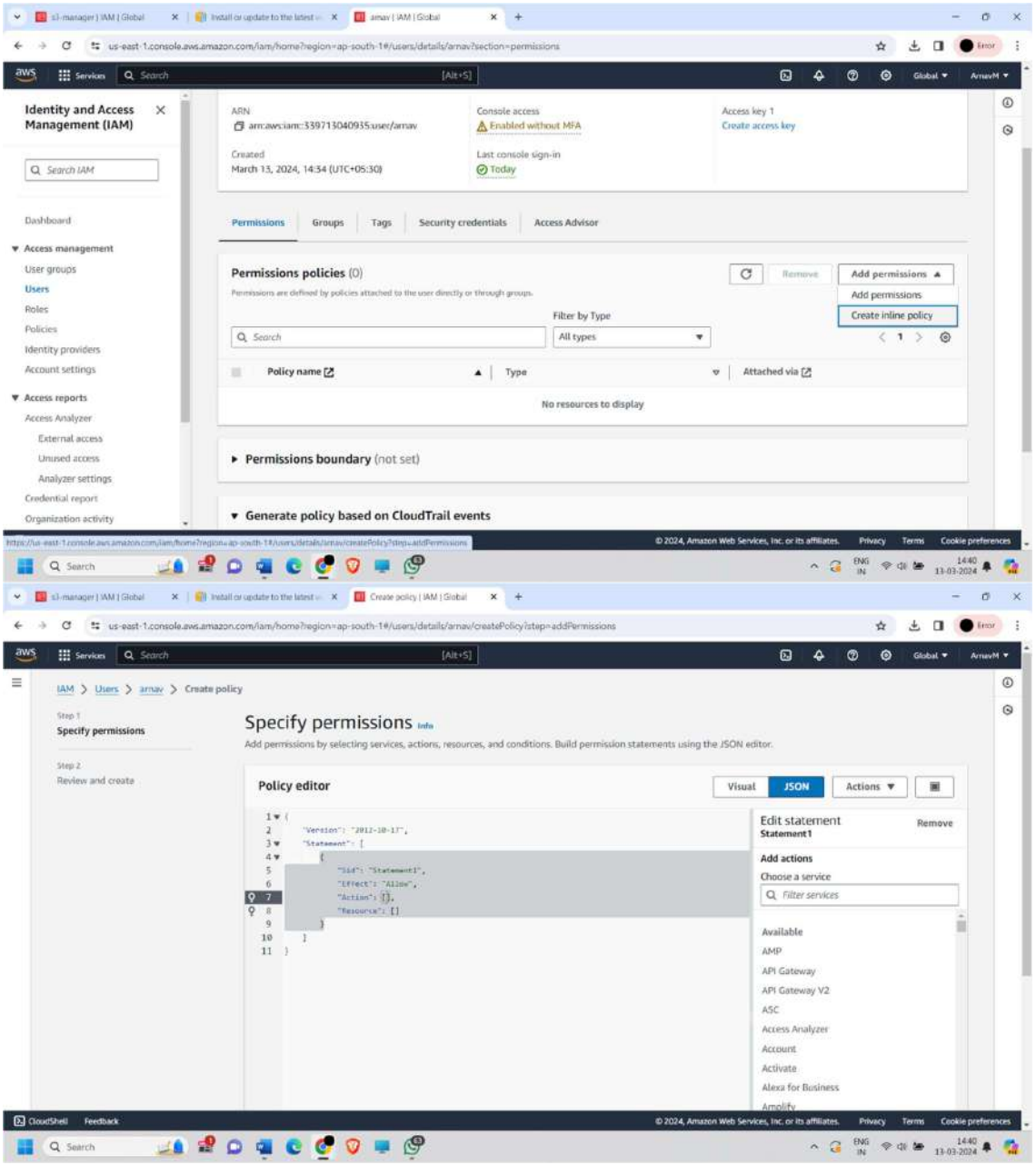

group1c23 user group created.
View group
Identity and Access Management (IAM)
IAM > User groups
User groups (1) Info
A user group is a collection of IAM users. Use groups to specify permissions for a collection of users.
Delete
Create group
Search
Group name
Users
Permissions
Creation time
group1c23
0
Defined
Now
Dashboard
Access management
User groups
Users
Roles
Policies
Identity providers
Account settings
Access reports
Access Analyzer
External access
Unused access
Analyzer settings
Credential report
Organization activity
CloudShell
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences


group1c23 user group created.
View group
Identity and Access Management (IAM)
User name
Groups
Last activity
Creation time
arnav
0
59 minutes ago
1 hour ago
rishab
0
None
45 minutes ago
timepass
0
9 minutes ago
18 minutes ago
vivaan
0
43 minutes ago
50 minutes ago
Attach permissions policies - Optional (1/911) Info
You can attach up to 10 policies to this user group. All the users in this group will have permissions that are defined in the selected policies.
Filter by Type
s3
All types
9 matches
Policy name
Type
Used as
Description
AmazonDMSRedshiftS...
AWS managed
None
Provides access to manage S3 settings...
AmazonS3FullAccess
AWS managed
Permissions policy (2), Boundar...
Provides full access to all buckets via t...
AmazonS3ObjectLam...
AWS managed
None
Provides AWS Lambda functions permi...
AmazonS3OutpostsFu...
AWS managed
None
Provides full access to Amazon S3 on ...
CloudShell
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences

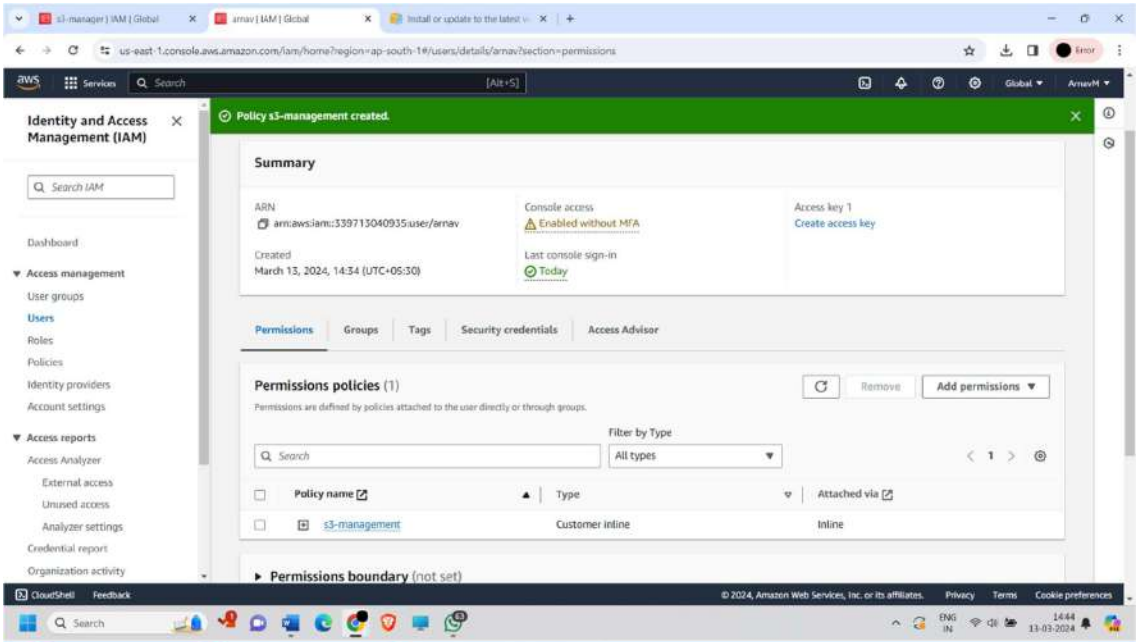

User groups | IAM | Global
us-east-1.console.aws.amazon.com/iam/home?region=ap-south-1#/groups
Identity and Access Management (IAM)
group2c23 user group created.
View group
IAM > User groups
User groups (2) Info
A user group is a collection of IAM users. Use groups to specify permissions for a collection of users.
Delete
Create group
Search IAM
Dashboard
Access management
User groups
Users
Roles
Policies
Identity providers
Account settings
Access reports
Access Analyzer
External access
Unused access
Analyzer settings
Credential report
Organization activity
Group name
Users
Permissions
Creation time
group1c23
Loading
4 minutes ago
group2c23
Loading
Now
CloudShell
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences
Console Home | Console Home
eu-north-1.console.aws.amazon.com/console/home?region=eu-north-1#
Incognito
Stockholm
Service menu
You can access all AWS services here. There are sections for recently visited and you can save your favorite services too.
Next
Reset to default layout
Add widgets
S3
EC2
Applications (0) Info
Region: Europe (Stockholm)
Create application
eu-north-1 (Current Region)
Find applications
Name
Description
Region
Originating account
Access denied
View all services
Go to myApplications
Welcome to AWS
Getting started with
AWS Health Info
Cost and usage Info
Access denied

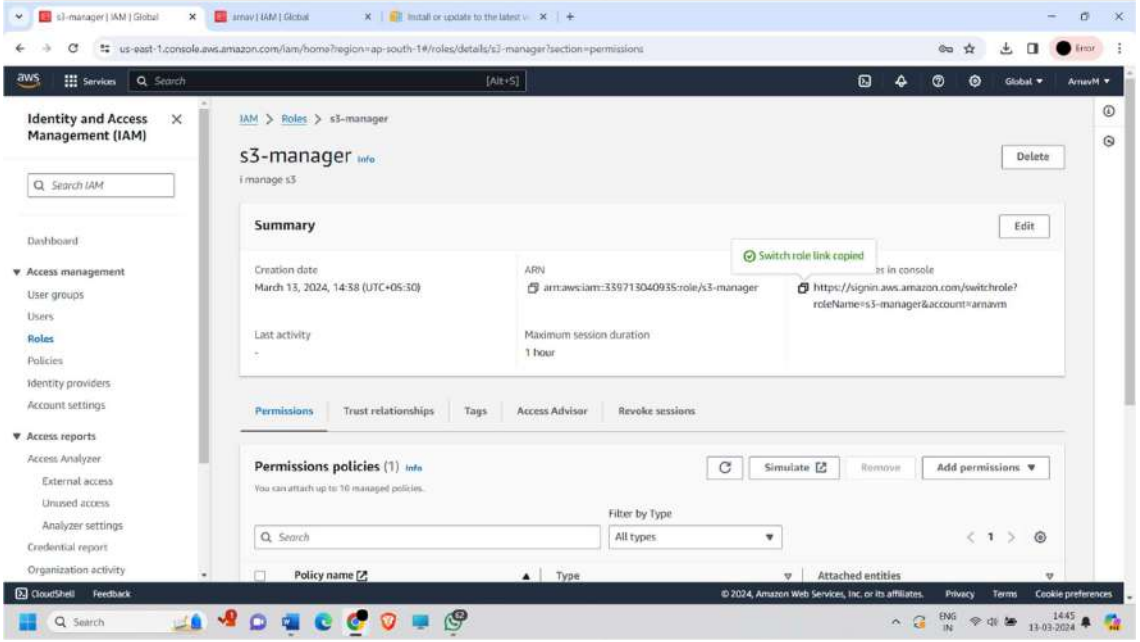

Create role | IAM | Global
us-east-1.console.aws.amazon.com/iam/home?region=ap-south-1#/roles/create
Services
Search
[Alt+S]
Global
ArnavM
Step 1
Select trusted entity
Step 2
Add permissions
Step 3
Name, review, and create
Select trusted entity Info
Trusted entity type
AWS service
Allow AWS services like EC2, Lambda, or others to perform actions in this account.
AWS account
Allow entities in other AWS accounts belonging to you or a 3rd party to perform actions in this account.
Web identity
Allows users federated by the specified external web identity provider to assume this role to perform actions in this account.
SAML 2.0 federation
Allow users federated with SAML 2.0 from a corporate directory to perform actions in this account.
Custom trust policy
Create a custom trust policy to enable others to perform actions in this account.
Use case
Allow an AWS service like EC2, Lambda, or others to perform actions in this account.
Service or use case
Choose a service or use case
Cancel
Next
CloudShell
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences
Search
ENG
IN
14:36
13-03-2024

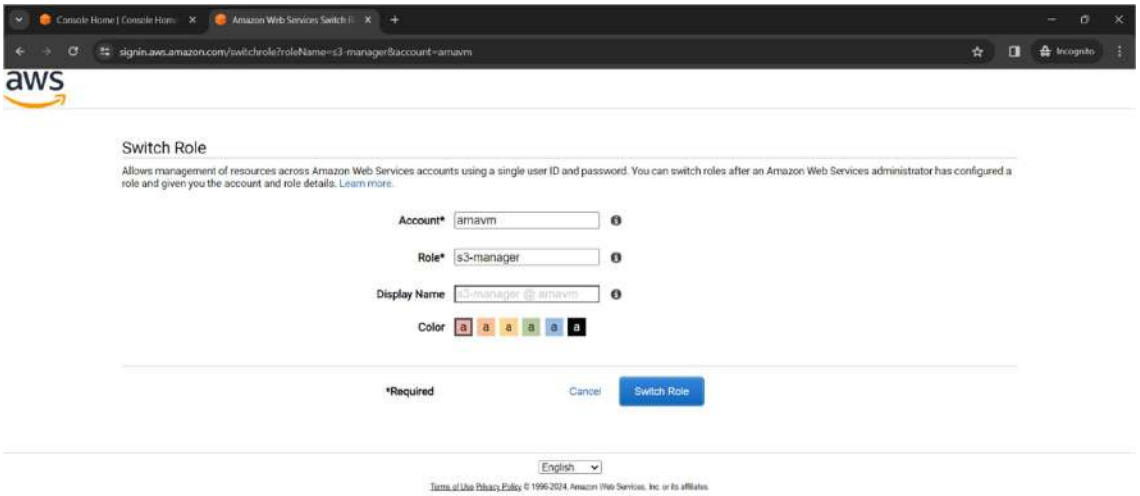
Roles | IAM | Global
us-east-1.console.aws.amazon.com/iam/home?region=ap-south-1#/roles
Services
Search
[Alt+S]
Global
ArnavM
Identity and Access Management (IAM)
Search IAM
Dashboard
Access management
User groups
Users
Roles
Policies
Identity providers
Account settings
Access reports
Access Analyzer
External access
Unused access
Analyzer settings
Credential report
Organization activity
Role s3-manager created.
View role
IAM > Roles
Roles (3) Info
An IAM role is an identity you can create that has specific permissions with credentials that are valid for short durations. Roles can be assumed by entities that you trust.
Delete
Create role
Search
Role name
Trusted entities
Last activity
AWSServiceRoleForSupport
AWS Service: support (Service-Linke
AWSServiceRoleForTrustedAdvisor
AWS Service: trustedadvisor (Service
s3-manager
Account: 339713040935
Roles Anywhere Info
Authenticate your non AWS workloads and securely provide access to AWS services.
Manage
Access AWS from your non AWS workloads
Operate your non AWS workloads using the same
X.509 Standard
Use your own existing PKI infrastructure or use AWS Certificate Manager Private Certificate Authority
Temporary credentials
Use temporary credentials with ease and benefit from the enhanced security they provide.
CloudShell
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences
Search
ENG
IN
14:38
13-03-2024

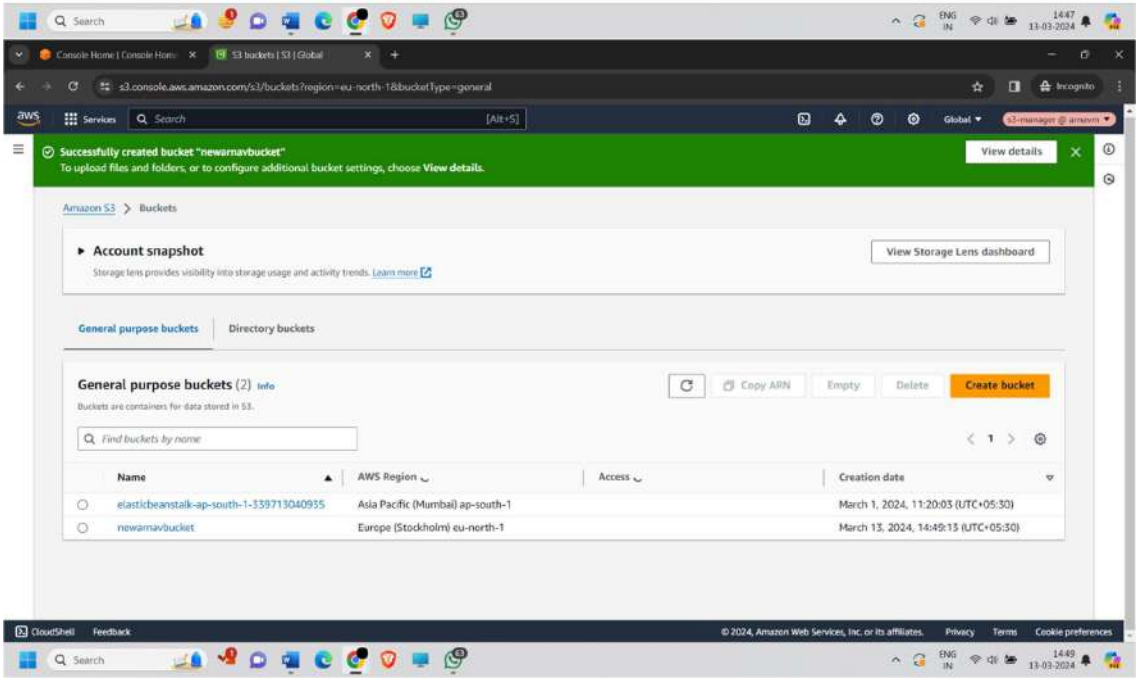

Identity and Access Management (IAM)
Search IAM
Dashboard
Access management
User groups
Users
Roles
Policies
Identity providers
Account settings
Access reports
Access Analyzer
External access
Unused access
Analyzer settings
Credential report
Organization activity
ARN
arn:aws:iam::339713040955:user/arnav
Console access
Enabled without MFA
Access key 1
Create access key
Created
March 13, 2024, 14:34 (UTC+05:30)
Last console sign-in
Today
Permissions
Groups
Tags
Security credentials
Access Advisor
Permissions policies (0)
Permissions are defined by policies attached to the user directly or through groups.
Remove
Add permissions
Add permissions
Create inline policy
Filter by Type
Search
All types
Policy name
Type
Attached via
No resources to display
Permissions boundary (not set)
Generate policy based on CloudTrail events
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences

IAM > Users > arnav > Create policy
Step 1
Specify permissions
Step 2
Review and create
Specify permissions
Info
Add permissions by selecting services, actions, resources, and conditions. Build permission statements using the JSON editor.
Policy editor
Visual
JSON
Actions
{
"Version": "2012-10-17",
"Statement": [
{
"Sid": "Statement1",
"Effect": "Allow",
"Action": [],
"Resource": []
}
]
}
Edit statement
Statement1
Remove
Add actions
Choose a service
Filter services
Available
AMP
API Gateway
API Gateway V2
ASC
Access Analyzer
Account
Activate
Alexa for Business
Amplify
CloudShell
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences


Policy s3-management created.
Summary
ARN
arn:aws:iam::339713040935:user/arnav
Console access
Enabled without MFA
Access key 1
Create access key
Created
March 13, 2024, 14:34 (UTC+05:30)
Last console sign-in
Today
Permissions
Groups
Tags
Security credentials
Access Advisor
Permissions policies (1)
Permissions are defined by policies attached to the user directly or through groups.
Remove
Add permissions
Filter by Type
All types
Policy name
Type
Attached via
s3-management
Customer inline
Inline
Permissions boundary (not set)


IAM > Roles > s3-manager
s3-manager Info
i manage s3
Delete
Summary
Edit
Switch role link copied
Creation date
March 13, 2024, 14:38 (UTC+05:30)
ARN
arn:aws:iam::339713040935:role/s3-manager
https://signin.aws.amazon.com/switchrole?roleName=s3-manager&account=arnavm
Last activity
-
Maximum session duration
1 hour
Permissions
Trust relationships
Tags
Access Advisor
Revoke sessions
Permissions policies (1) Info
You can attach up to 10 managed policies.
Simulate
Remove
Add permissions
Filter by Type
All types
Policy name
Type
Attached entities





Conclusion:

In conclusion, mastering IAM practices on AWS is crucial for maintaining a secure and compliant cloud environment. Understanding the differences between root users and IAM users, roles and policies, and implementing MFA adds layers of security necessary for protecting valuable resources and data in the cloud.

THADOMAL SHAHANI TSEC ENGINEERING COLLEGE

Experiment :- 8

Aim:- To study and Implement Database as a service on SQL databases using AWS RDS.

Theory :-

Implementing a database as a service (DBaaS) using Amazon Web Services (AWS) Relational Database Service (RDS) involves leveraging AWS's managed database service to deploy and operate SQL databases in the cloud without the need for managing the underlying infrastructure.

AWS RDS supports various SQL database engines such as MySQL, Postgre SQL, Oracle, SQL Server and Maria DB providing users with flexibility and scalability. By utilising RDS users can easily provision, configure and scale database instances based on their application requirements, while AWS handles tasks like backups, software patching, monitoring and maintenance.

This approach not only reduces the operational overhead associated with managing databases but also ensures high availability, durability and security of data.

Additionally AWS RDS offers features like automated backups, multi-AZ deployment for high availability, read replicas for scalability and encryption at rest and in transit, making it an ideal choice for deploying SQL databases in a cloud-native environment.

THADOMAL SHAHANI
TSEC
ENGINEERING COLLEGE

RDS vs Aurora

RDS	Aurora
1) Data is not ~~compte~~ continuously backed up.	1) Data is continuously backed up.
2) Less durable than aurora.	2) Multiple copies makes it more durable.
3) Standard Recovery.	3) Fast recovery from failure.
4) Less compared to aurora.	4) Expensive than RDS

Conclusion:-
Understood and implemented database as a service on SQL databases using AWS RDS.

20/3/24

Arnav Malvia
C23
2103109

Experiment 8 :-

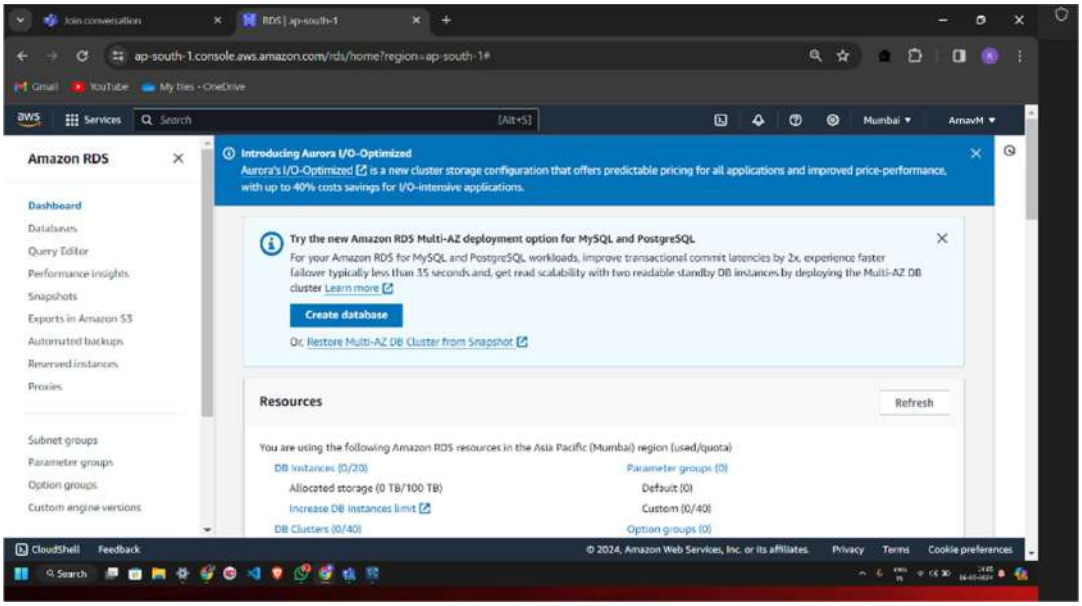
Aim: To study and Implement a Database as a Service on SQL databases Using AWS RDS.

Theory:

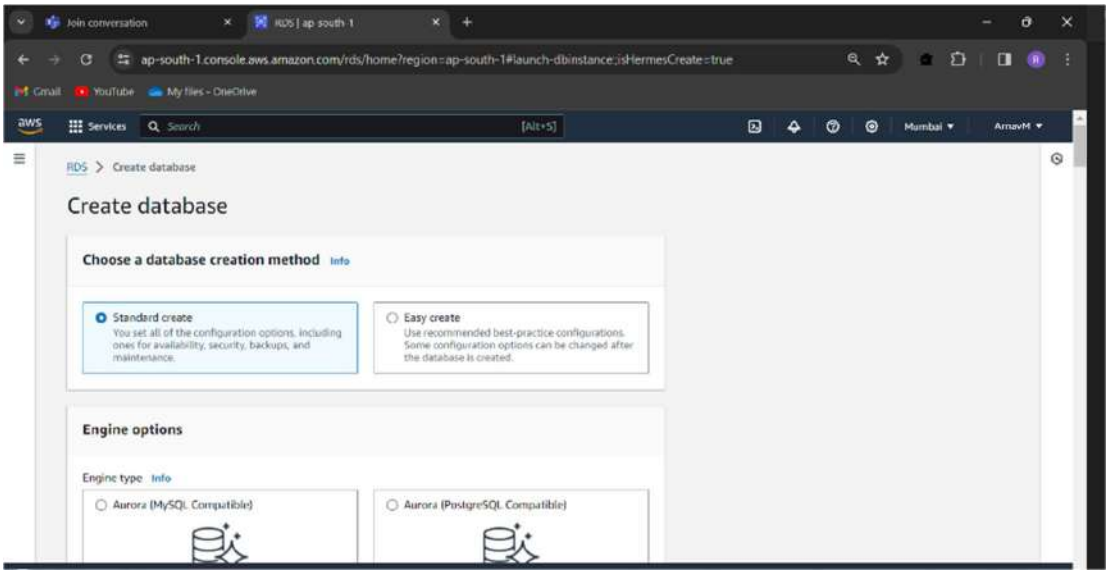
Implementing a Database as a Service (DBaaS) using Amazon Web Services (AWS) Relational Database Service (RDS) involves leveraging AWS's managed database service to deploy and operate SQL databases in the cloud without the need for managing the underlying infrastructure. AWS RDS supports various SQL database engines such as MySQL, PostgreSQL, Oracle, SQL Server, and MariaDB, providing users with flexibility and scalability. By utilizing RDS, users can easily provision, configure, and scale database instances based on their application requirements, while AWS handles tasks like backups, software patching, monitoring, and maintenance. This approach not only reduces the operational overhead associated with managing databases but also ensures high availability, durability, and security of the data. Additionally, AWS RDS offers features like automated backups, multi-AZ deployment for high availability, read replicas for scalability, and encryption at rest and in transit, making it an ideal choice for deploying SQL databases in a cloud-native environment.

Steps for creation and usage of database as a service : -

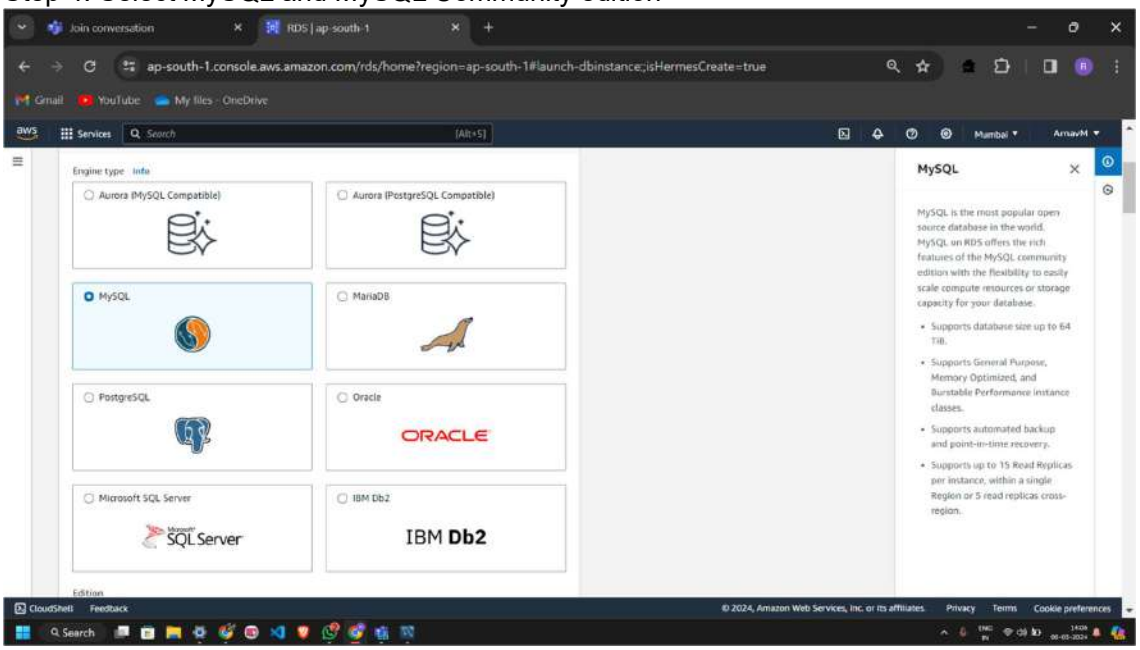
Step1 : Login to aws console and search RDS



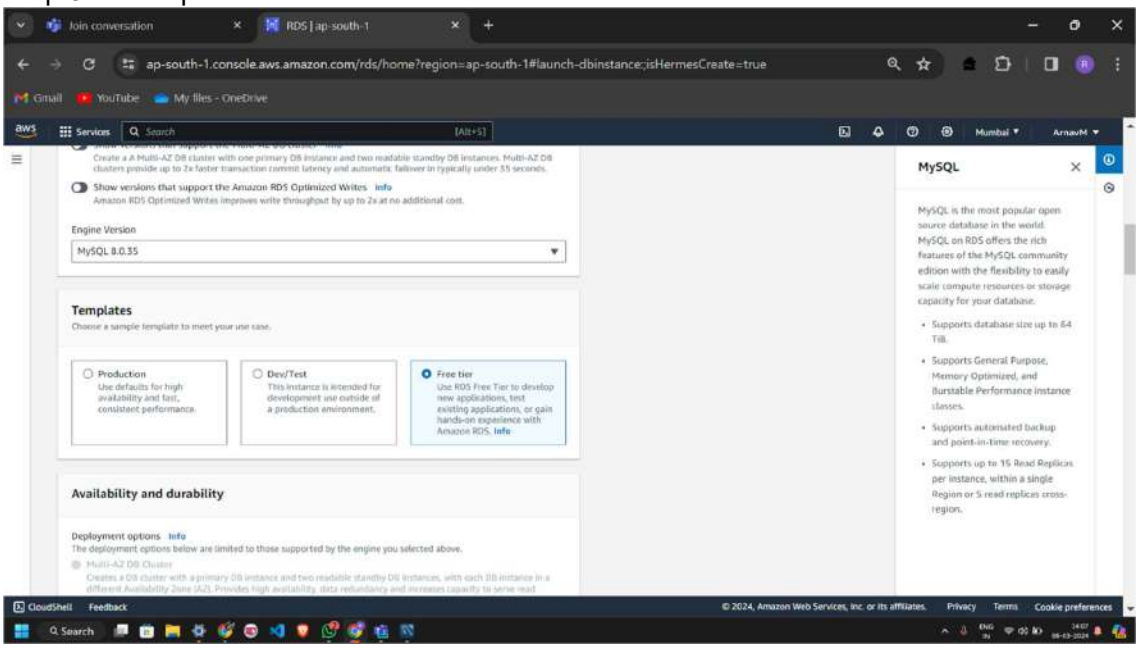
Step2: Click on to RDS and create database



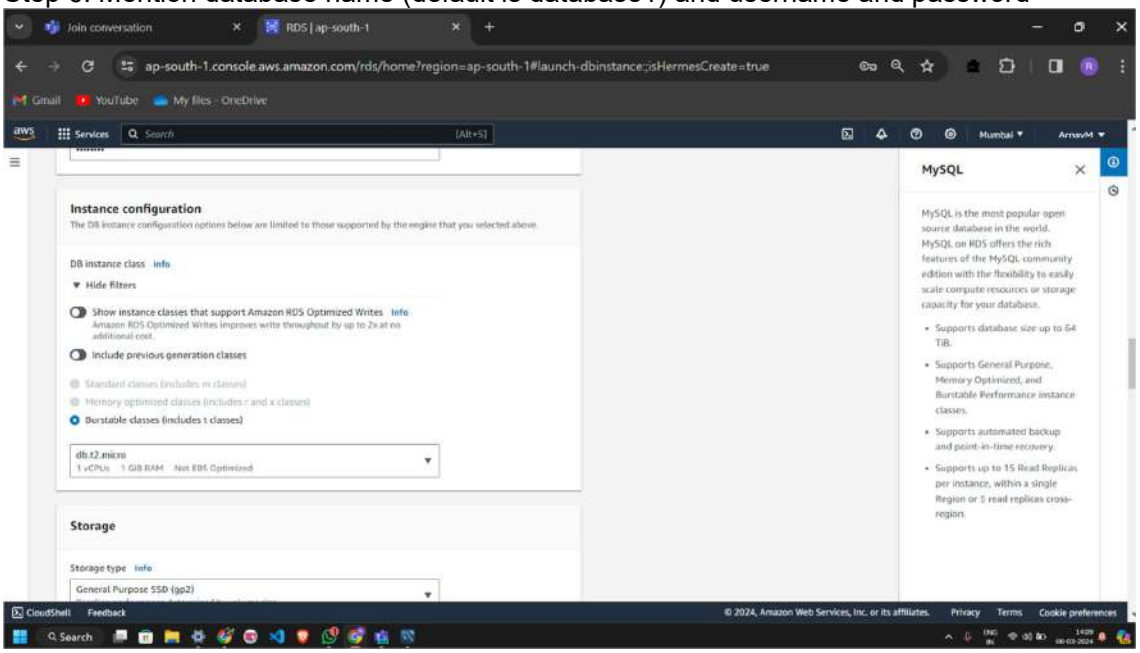
Step 3: Select standard database
Step 4: Select MySQL and MySQL Community edition



Step 5: In Templates select Free tier



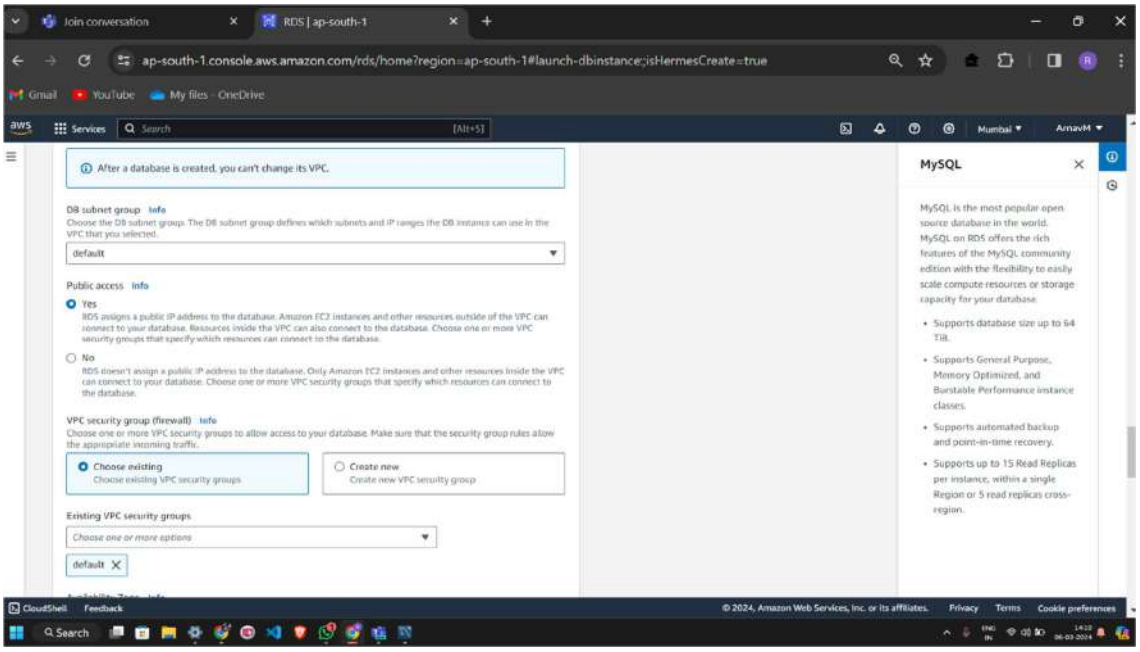
Step 6: Mention database name (default is database1) and username and password



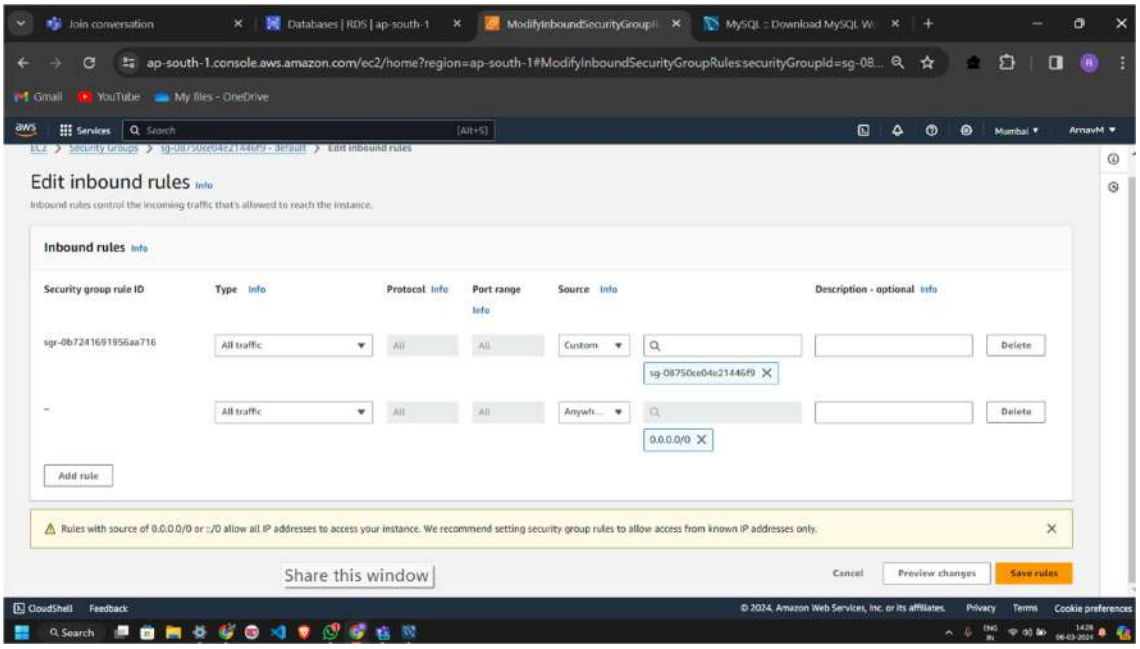
Step 7: Instance is t2.micro

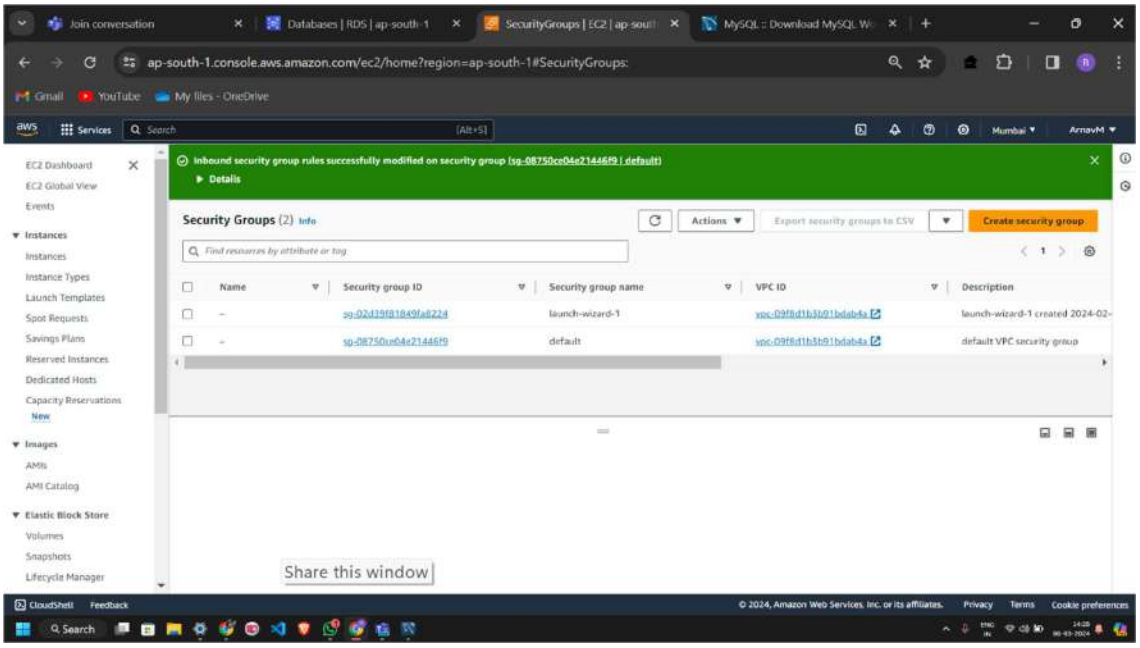
Rest of things keep default

Step 8: Select Public Acess -Yes



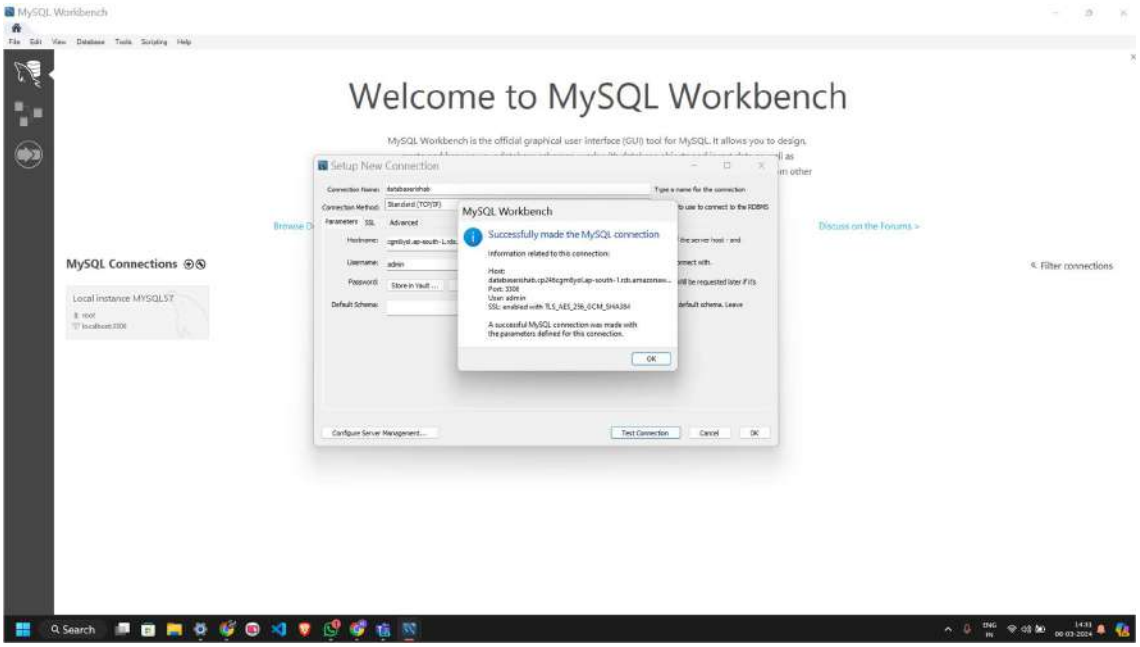
Step 9: Click on to create Database





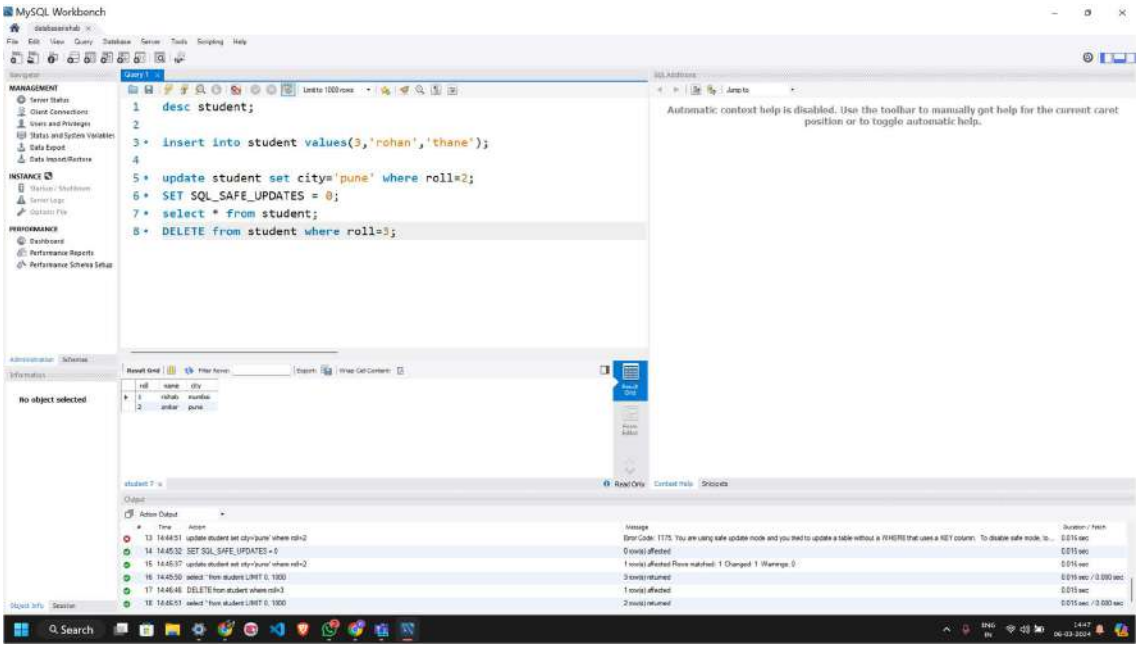
Click on to MySQL connection

Paste copied endpoint in Hostname
Connection Name: databasearnav
Username: admin
Click on to Test Connection



Click on Ok button
Go to workbench double click on connection,

Write query and execute



Now delete the instance (once you have done with it)
Select instance go to action stop instance and then delete instance

Conclusion:

In conclusion, leveraging AWS RDS for Database as a Service offers a streamlined approach to deploying and managing SQL databases in the cloud, providing scalability, reliability, and security for applications relying on SQL data storage.

THADOMAL SHAHANI
TSEC
ENGINEERING COLLEGE

Experiment :- 9

Aim :- To study and implement containerization using Docker.

Theory :-

→ Docker uses a client server architecture. Docker is a containerisation platform that provides a complete system for building and running software containers. It uses a client server architecture where the client (usually CLI) sends request to a seperate process that's responsible for carrying out the required actions. The docker architecture has a - Docker client
- Docker Daemon
- Docker images
- Docker containers
- Docker registries

Docker is a powerful ~~tool~~ tool that can help you to develop, deploy and manage applications more efficiently.

→ Benefits of containerisation :-
Containerisation is a technology that produces executable software application packages that are abstracted from host operating system.
Benefits include :- Portability
- Agility
- Efficiency
- Faster Delivery
- Flexibility

THADOMAL SHAHANI TSEC ENGINEERING COLLEGE

→ Container: A docker container is a standalone executable package of software that includes everything needed to run an application: code, runtime, system tools, system libraries.

A docker image is a standalone executable package of software that includes everything. They are templates that contain instructions for creating a docker container. A docker container is a running instance of a docker image.
Docker file is a text ~~docke~~ document containing all the commands the user requires to call on the commands on the user requires to call on the command line to assemble an image.

→ Containers and virtual machines (VMs) are complementary technologies that can be used ~~as~~ together to make applications independent from IT infrastructure resources. An image is a snapshot of an environment and a container runs the software. An image is a template that contains instructions for running a container. A container is a runnable software application or service.

Conclusion:-
Understood and implemented containerisation using docker.

A+

27/3/24

Arnav Malvia
C23
2103109

Experiment 9

Q1. What is Docker and Explain its architecture.

Docker is a set of platform as a service products that use OS-level virtualization to deliver software in packages called containers. Containers isolate software from its surroundings, for example differences between development and staging environments and help reduce conflicts between teams running different software on the same infrastructure.

Docker architecture uses a client-server model and comprises of the Docker Client, Docker Host, Network and Storage components, and the Docker Registry / Hub.

- Docker Client:

The Docker client provides a command line interface (CLI) that allows you to issue build, run, and stop application commands to a Docker daemon. It is responsible for sending commands to the Docker daemon.

- Docker Daemon:

The Docker daemon is a service that runs on the Docker host and is responsible for building, running, and distributing Docker containers. It is responsible for carrying out the commands received from the Docker client.

- Docker Host:

The Docker host is the physical or virtual machine that runs the Docker daemon. It provides the resources that the Docker daemon needs to build, run, and distribute Docker containers.

- Docker Registry:

The Docker registry is a repository that stores Docker images. Docker images are read-only templates that contain instructions for creating Docker containers. Docker images can be pulled from the Docker registry and used to create Docker containers.

- Docker Hub:

Docker Hub is a public Docker registry that is maintained by Docker, Inc. It stores Docker images that can be pulled and used by anyone.

Docker architecture is designed to be modular and extensible. This makes it easy to add new features and functionality to Docker. Docker architecture is also designed to be portable. This means that Docker can be used on a variety of platforms, including Linux, Windows, and macOS.

Docker is a powerful tool that can be used to automate the deployment of applications in lightweight containers. Docker containers can be used to run applications in a variety of environments, including development, staging, and production. Docker containers can also be used to create microservices architectures.

Q2. Benefits of Containerization

Containerization can provide many benefits to businesses, including:

- Faster deployment: Containers can speed up the process of configuring and deploying applications.
- Increased agility: Containers are platform-agnostic, so developers can use them regardless of the operating system or platform.
- Greater speed: Containers share a machine's OS, so they're not overburdened by excess overheads.
- Fault isolation: Each container application is isolated and operates independently.
- Improved security: Containerized applications run independently in separate containers, each with its own level of security.
- Resource maximization: Containers use fewer resources than virtual machines.
- Easier management: Containers allow for easier and more consistent management of applications and services.
- Portability: Containers can be run on different platforms and clouds.
- Consistency: DevOps teams know applications in containers will run the same, regardless of where they are deployed.

Q3. Explain following w.r.t Docker

• Container

• Images

• **DockerFile**

A Docker container is a lightweight, standalone, executable package of software that includes everything needed to run an application: code, runtime, system tools, system libraries and settings. Containers isolate software from its surroundings, for example differences between development and staging environments and help reduce conflicts between teams running different software on the same infrastructure.

Containers are a great way to package and deploy applications because they are:

- Standardized units of software

that package up code and all its dependencies so that the application runs quickly and reliably from one computing environment to another.

- Portable and efficient

because they virtualize the operating system instead of hardware. This means that containers use fewer resources than virtual machines and can be deployed on any machine that has the Docker engine installed.

- Take up less space than VMs

(container images are typically tens of MBs in size) and can handle more applications, requiring fewer VMs and Operating systems.

- Fast to create and start

(seconds versus minutes for VMs) and can run anywhere, from your laptop to the cloud.

Docker images are templates that contain instructions for creating Docker containers. Containers are isolated, lightweight, and portable, and can be used to run applications on any platform.

Docker images are made up of layers, each of which contains a set of instructions for modifying the file system. The first layer is the base layer, which is usually an existing Docker image. The other layers are added on top of the base layer, and each layer contains the changes that need to be made to the file system to create the desired container.

Docker images are stored in a registry, which is a server that stores and distributes Docker images. The Docker Hub is the official registry for Docker images, but there are also many other registries available.

To create a Docker image, you need to create a Dockerfile. A Dockerfile is a text file that contains instructions for building a Docker image. The Dockerfile tells Docker what base image to use, what commands to run, and what files to copy into the image.

Once you have created a Dockerfile, you can build the Docker image using the docker build command. The docker build command will read the Dockerfile and create a Docker image based on the instructions in the file.

Once you have created a Docker image, you can run it using the docker run command. The docker run command will create a Docker container from the image and run the application that is contained in the image.

Docker images are a powerful tool for developing and deploying applications. They allow you to create isolated, lightweight, and portable containers that can be run on any platform.

A Dockerfile is a text file that contains instructions for building a Docker image. Docker images are used to create Docker containers. Docker containers are isolated, lightweight, and portable environments that can run applications.

Dockerfiles are made up of a series of commands, each of which tells Docker how to build the image. The commands are executed in order, and each one creates a new layer in the image. The final layer is the image that is used to create containers.

Q4. Compare the following

• Container and Virtual Machine

• Images and Container

differences between containers and virtual machines (VMs):

- Size

VM image files are larger, measuring in gigabytes, while container files are smaller, measuring in megabytes.

- Operating system

VMs include an operating system instance, while containers do not.

- Virtualization

VMs virtualize an entire machine, including the hardware layers, while containers are lightweight software packages that contain all the dependencies required to execute the contained software application.

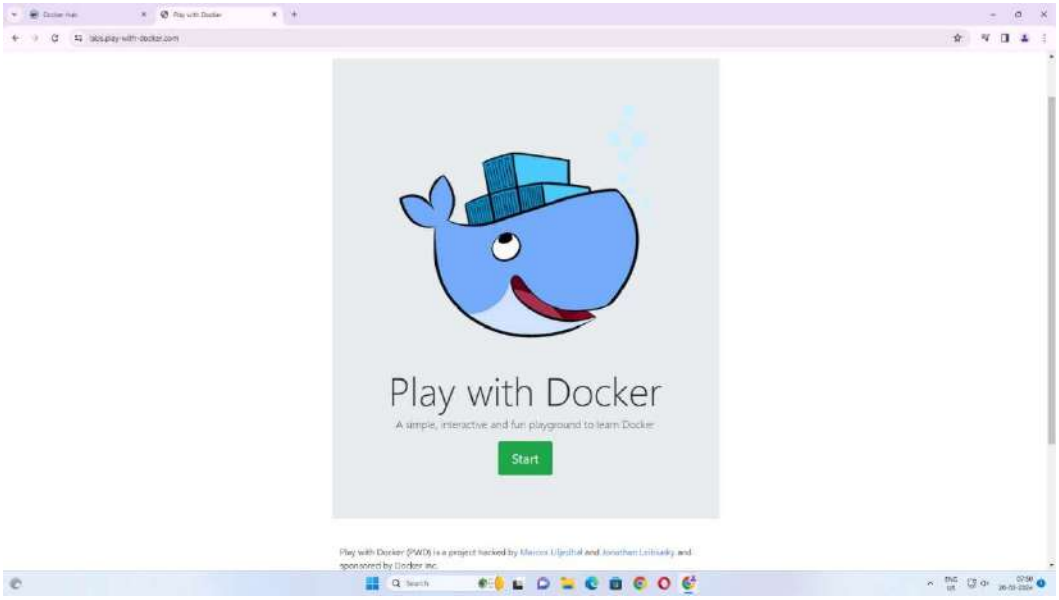
- Awareness

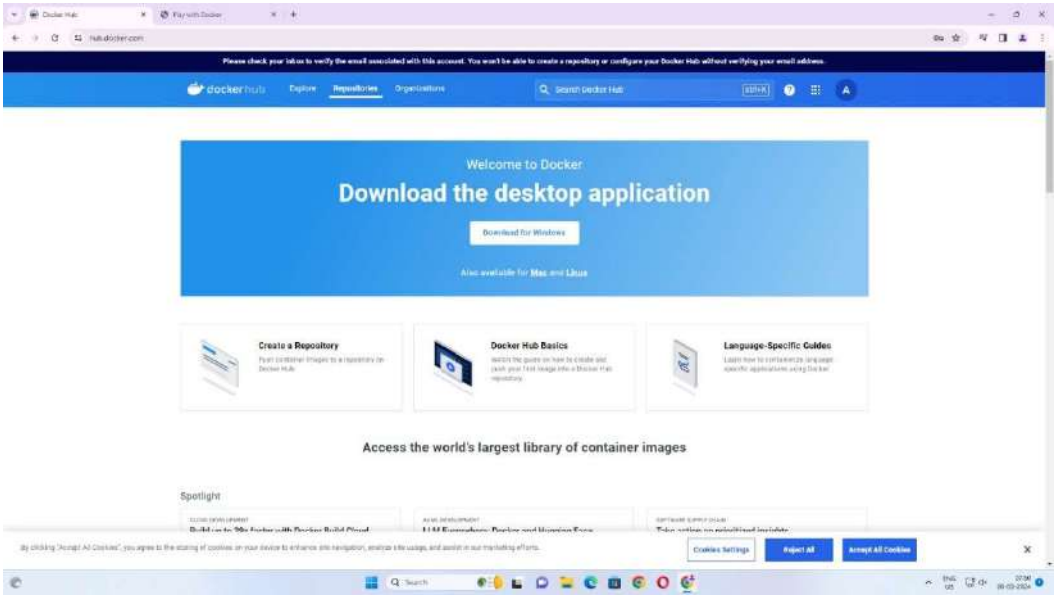
Containers have no awareness of other processes running on the host OS and have limited access to hardware.

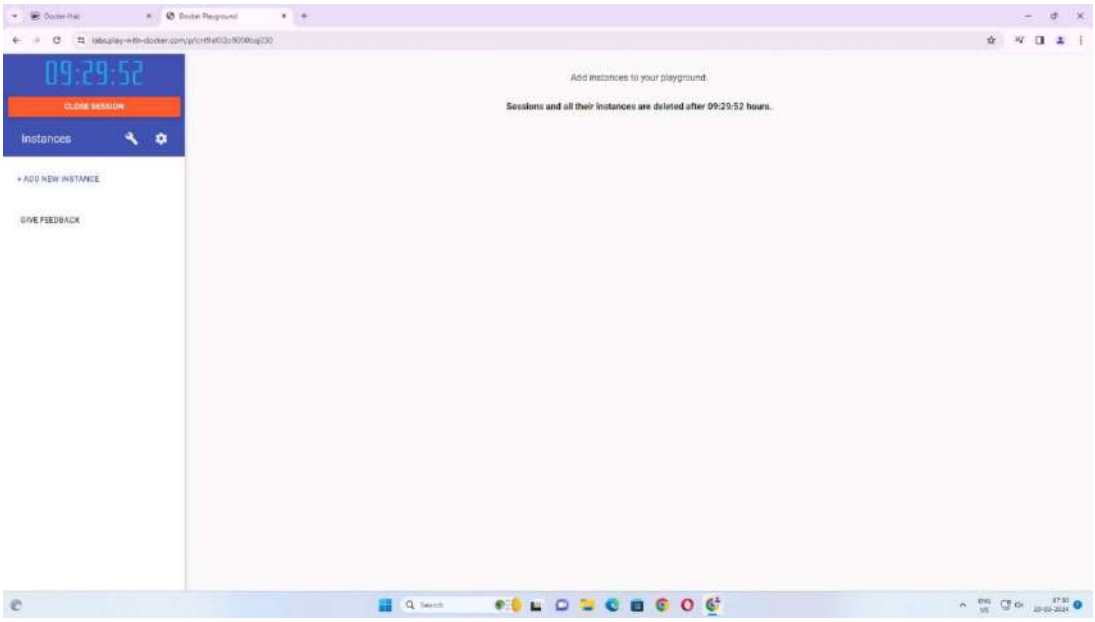
- Software

VMs are a piece of software that allows you to install other software inside of it, while a container is software that allows different functionalities of an application independently.

Q5. Write steps and attach screenshots of Implement Containerization using Docker





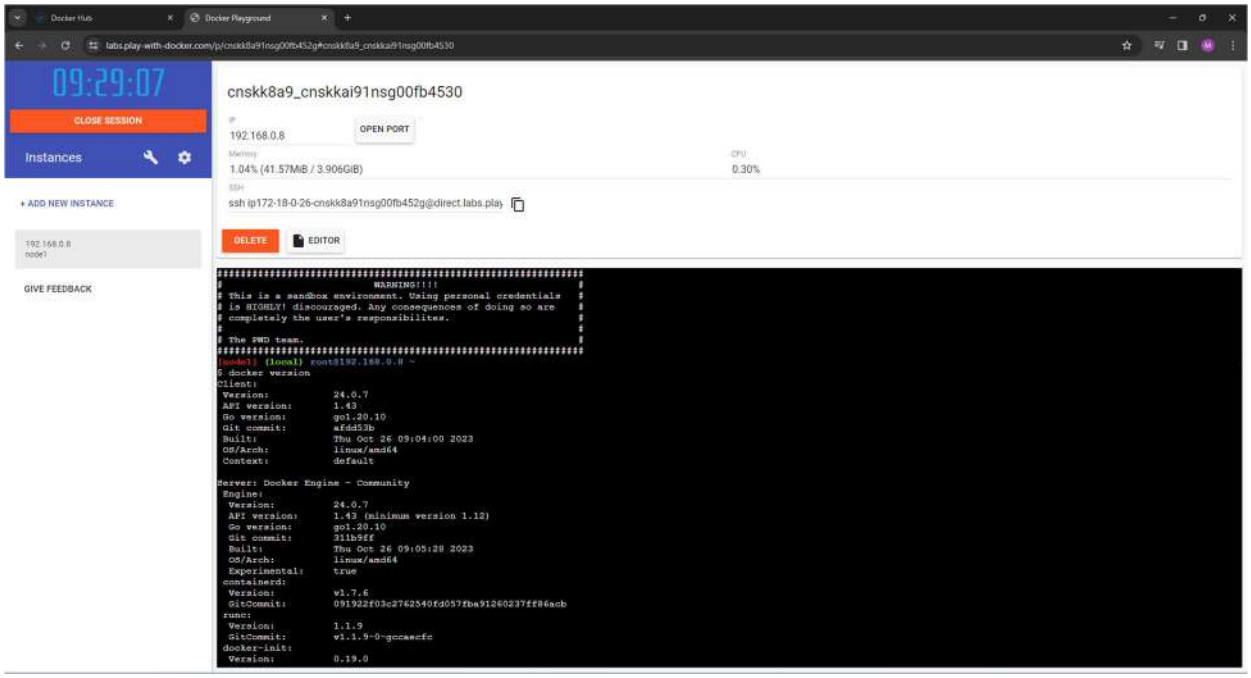


1. Make a docker playground login user.

```
  stop        Stop one or more running containers
  top         Display the running processes of a container
  unpause     Unpause all processes within one or more containers
  update      Update configuration of one or more containers
  wait        Block until one or more containers stop, then print their exit codes

Run 'docker container COMMAND --help' for more information on a command.
[node1] (local) root@192.168.0.8 ~
$ docker container ps -a
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES
[node1] (local) root@192.168.0.8 ~
$ docker pull ubuntu
Using default tag: latest
latest: Pulling from library/ubuntu
bccd10f490ab: Pull complete
Digest: sha256:77906da86b60585ce12215807090eb327e7386c8fafb5402369e421f44eff17e
Status: Downloaded newer image for ubuntu:latest
docker.io/library/ubuntu:latest
[node1] (local) root@192.168.0.8 ~
$ docker image ls -a
REPOSITORY   TAG       IMAGE ID       CREATED       SIZE
ubuntu       latest    ca2b0f26964c   2 weeks ago   77.9MB
[node1] (local) root@192.168.0.8 ~
$ docker pull ubuntu:20.04
20.04: Pulling from library/ubuntu
17d0386c2fff: Pull complete
Digest: sha256:80ef4a44043dec4490506e6cc4289eeda2d106a70148b74b5ae91ee670e9c35d
Status: Downloaded newer image for ubuntu:20.04
docker.io/library/ubuntu:20.04
[node1] (local) root@192.168.0.8 ~
$ docker image ls -a
REPOSITORY   TAG       IMAGE ID       CREATED       SIZE
ubuntu       latest    ca2b0f26964c   2 weeks ago   77.9MB
ubuntu       20.04     3cff1c6ff37e   4 weeks ago   72.8MB
[node1] (local) root@192.168.0.8 ~
$
```

Execute the docker container command

Use -ls , -a command with docker container to view it.

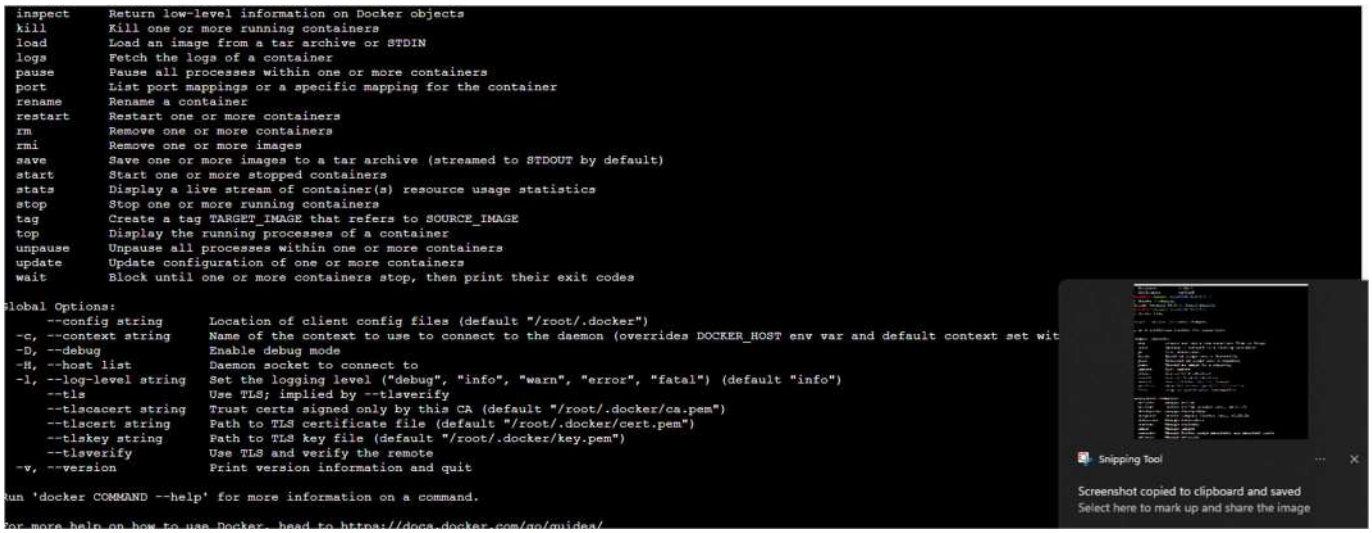



```
Run 'docker container COMMAND --help' for more information on a command.
[node1] (local) root@192.168.0.8 ~
$ docker container ps -a
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES
[node1] (local) root@192.168.0.8 ~
$ docker pull ubuntu
Using default tag: latest
latest: Pulling from library/ubuntu
bccd10f490ab: Pull complete
Digest: sha256:77906da86b60585ce12215807090eb327e7386c8fafb5402369e421f44eff17e
Status: Downloaded newer image for ubuntu:latest
docker.io/library/ubuntu:latest
[node1] (local) root@192.168.0.8 ~
$
```

Docker pull ubuntu command to switch to ubuntu.

```
 Version:           0.19.0
 GitCommit:         de40ad0
[node1] (local) root@192.168.0.8 ~
$ docker --version
Docker version 24.0.7, build afdd53b
[node1] (local) root@192.168.0.8 ~
$ docker help

Usage:  docker [OPTIONS] COMMAND

A self-sufficient runtime for containers

Common Commands:
  run         Create and run a new container from an image
  exec        Execute a command in a running container
  ps          List containers
  build       Build an image from a Dockerfile
  pull        Download an image from a registry
  push        Upload an image to a registry
  images      List images
  login       Log in to a registry
  logout      Log out from a registry
  search      Search Docker Hub for images
  version     Show the Docker version information
  info        Display system-wide information

Management Commands:
  builder     Manage builds
  buildx*     Docker Buildx (Docker Inc., v0.11.2)
  checkpoint  Manage checkpoints
  compose*    Docker Compose (Docker Inc., v2.23.0)
  container   Manage containers
  context     Manage contexts
  image       Manage images
  manifest    Manage Docker image manifests and manifest lists
  network     Manage networks
```

Docker version command to check the version details of the docker



Writing echo hello world to print the statement using docker run -it

Ubuntu sleep 200.

Docker kill container id kills the container which is running.

```
$ docker run -it ubuntu sleep 10; echo "hello world"
hello world
[node1] (local) root@192.168.0.8 ~
$ docker run -dit ubuntu sleep 10; echo "hello world"
e54b56021791b140091f6b51829642c320af80f4fe5a9d3f75680ded40736263
hello world
[node1] (local) root@192.168.0.8 ~
$ docker container ls
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES
[node1] (local) root@192.168.0.8 ~
$ docker run -dit ubuntu sleep 300; echo "hello world"
694169f7f91d951a3593c26791b1a6f72d536319ce2b066c2d6b1e7b29460011
hello world
[node1] (local) root@192.168.0.8 ~
$ docker container ls
CONTAINER ID   IMAGE     COMMAND       CREATED         STATUS         PORTS     NAMES
694169f7f91d   ubuntu    "sleep 300"   6 seconds ago   Up 6 seconds             serene_shannon
[node1] (local) root@192.168.0.8 ~
$ docker container stop 694169f7f9
694169f7f9
[node1] (local) root@192.168.0.8 ~
$ docker container ls
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES
[node1] (local) root@192.168.0.8 ~
$ docker container kill  694169
Error response from daemon: Cannot kill container: 694169: Container 694169f7f91d951a3593c26791b1a6f72d536319ce2b066c2d6b1e7b29460011 is not running
[node1] (local) root@192.168.0.8 ~
$ docker run -dit ubuntu sleep 300; echo "hello world"
5fab4b0b92fc66ad4351202afa9d6df78316ad1f334c2dca45a38832410e801c
hello world
[node1] (local) root@192.168.0.8 ~
$ docker container ls
CONTAINER ID   IMAGE     COMMAND       CREATED          STATUS          PORTS     NAMES
5fab4b0b92fc   ubuntu    "sleep 300"   19 seconds ago   Up 18 seconds             serene_dirac
[node1] (local) root@192.168.0.8 ~
$
```

THADOMAL SHAHANI
TSEC
ENGINEERING COLLEGE

Experiment :- 10

Aim :- To study and implement the container observation using Kubernetes.

Theory :-

Container Orchestration -

It automatically provides, deploys, scales and manages containerised applications without worrying about the underlying infrastructure. Developers can implement container orchestration anywhere containers are, allowing them to automate the lifecycle management of containers.

Need of container orchestration -

1) Reliable application development
2) Scalability
3) Lower Costs
4) Enhanced Security
5) High availability

Kubernetes -

Kubernetes is a portable, extensible, open source platform for managing containerised workloads and services. that facilitates both declarative configuration and automation. It has a large rapidly growing ecosystem.

They are a robust container orchestration platform that offeres advanced features. Docker swarm is a simple and lightweight container orchestration, platform and provide basic features.

THADOMAL SHAHANI TSEC ENGINEERING COLLEGE

Services offered by Kubernetes:-
1) Container orchestration
2) Service Discovery and Load Balancing
3) Automated Rollouts and Roll Backs
4) Storage Orchestration
5) Automated Scaling

Kubernetes Components:-
1) Control Plane Components
2) Kube- apiserver components
3) etcd
4) Nodes
5) Image Registry and Pods.

Kubernetes follow modular architecture with multiple components, while Docker swarm has simple architecture with fewer components, including swarm manager and swarm agents.

Conclusion:-
Understood and implemented the container orchestration using Kubernetes.

10/4/2024

Arnav Malvia

Roll No. :2103109

Batch : C23

Experiment 10

Q1. Explain need of container orchestration tool

Container orchestration tools automate the lifecycle of containers, including provisioning, deployment, and scaling. This allows organizations to benefit from containerization at scale without additional maintenance overhead.

Container orchestration tools manage and automate the complete lifecycle of containers, including provisioning, deployment, and scaling. This allows organizations to capture the benefits of containerization at scale without incurring additional maintenance overhead.

Benefits of container orchestration include:

- Efficient resource management
- Cost savings
- Greater agility
- Simplified deployments
- Interoperability
- Easy scaling of applications and infrastructure
- Ideal for microservices architecture
- Improved governance and security

Container orchestration also automates time-consuming tasks like:

- Recreating, scaling, and upgrading containers
- Managing networking and storage capabilities
- Container provisioning and deployment
- Configuration and scheduling
- Resource allocation
- Scaling containers to balance the application workload
- Ensuring container availability
- Load balancing of service discovery
- Container performance monitoring

Container orchestration tools like Google Kubernetes Engine (GKE) make it easier to deploy and run containerized applications and microservices. Container orchestrators typically apply their own methodologies and offer varying capabilities, but they all enable organizations to automatically coordinate, manage, and monitor containerized applications.

Container orchestration automatically provisions, deploys, scales, and manages containerized applications without worrying about the underlying infrastructure. Developers can implement container orchestration anywhere containers are, allowing them to automate the life cycle management of containers.

Q2. What is Kubernetes? Describe its features

Kubernetes, also known as K8s, is an open-source platform for managing containerized applications. It automates the tasks of deploying, scaling, and managing applications, and provides features like orchestration, resource management, and service discovery. Kubernetes can be used to deploy a variety of applications, including web, machine learning, and big data applications.

some features of Kubernetes:

- Service discovery and load balancing
- Storage orchestration
- Automated rollouts and rollbacks
- Automatic bin packing
- Self-healing
- Horizontal scaling
- Container balancing
- Backward-compatible model

Kubernetes also allows pods to be treated as VMs in terms of naming, port allocation, application configuration, migration, and service discovery.

Kubernetes, often abbreviated as “K8s”, orchestrates containerized applications to run on a cluster of hosts. The K8s system automates the deployment and management of cloud native applications using on-premises infrastructure or public cloud platforms. It distributes application workloads across a Kubernetes cluster and automates dynamic container networking needs. Kubernetes also allocates storage and persistent volumes to running containers, provides automatic scaling, and works continuously to maintain the desired state of applications, providing resiliency.

Q3. Explain Kubernetes Components, its working and architecture

Kubernetes is an open-source platform for deploying and managing containers. Kubernetes architecture follows a distributed model, comprising multiple components working together to ensure the seamless execution of containerized applications. At the core of this architecture lies the master node, responsible for

managing the entire cluster and orchestrating the operations across worker nodes.

In this blog, we are going to cover the *Kubernetes Cluster architecture, Kubernetes components, Managed Kubernetes Services.* Also, we will be discussing the *Kubernetes master node & worker node* and their components.

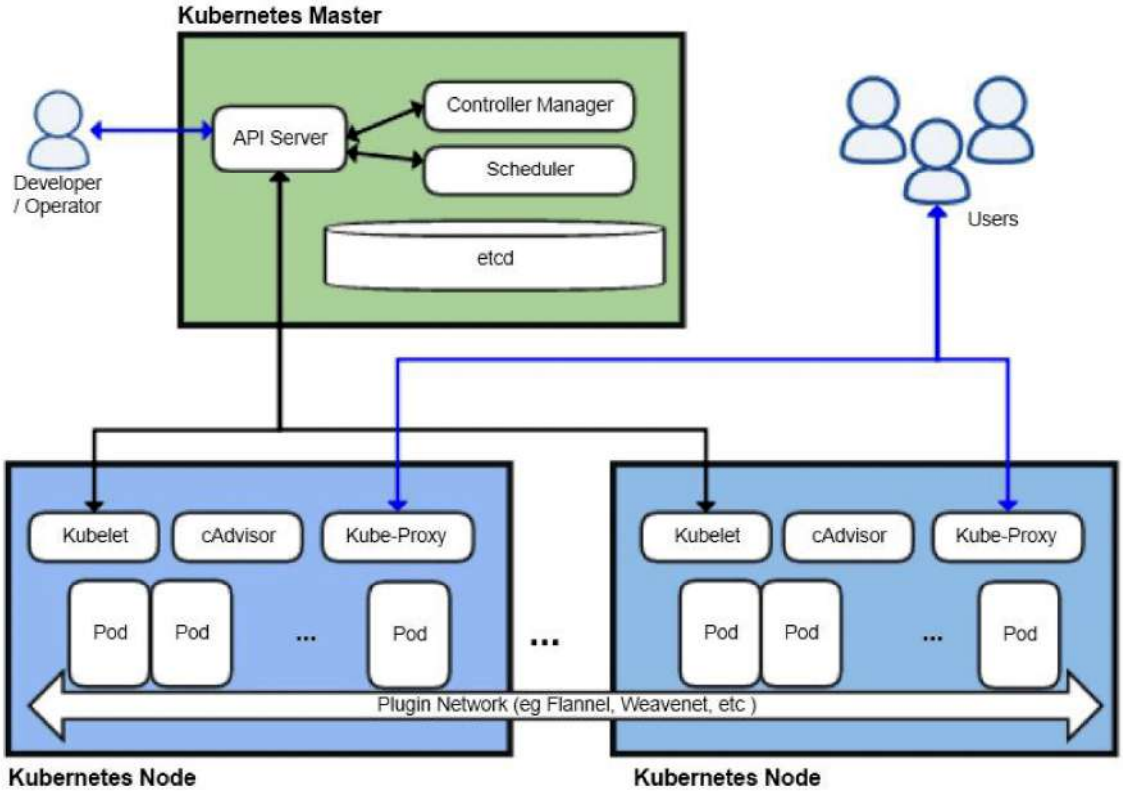
What Is Kubernetes?

In organizations, multiple numbers of containers run on multiple hosts at a time. So it becomes very hard to manage all the containers together, a simple solution to this would be Kubernetes. Kubernetes is an open-source platform for managing containerized workloads and services. Kubernetes take care of scaling and failover for our application running on the container.

Read More : **Kubernetes for beginners**

What is Kubernetes Architecture?

Kubernetes provides a flexible architecture for discovering services within a cluster while maintaining loose connections. A Kubernetes cluster comprises a set of control planes and compute nodes. The primary role of the control plane is to oversee the entire cluster, expose the API, and manage the scheduling of compute nodes according to the desired settings. The compute nodes run container runtimes such as Docker, alongside a communication agent called kubelet, which interacts with the control plane. These nodes can consist of physical servers, virtual machines (VMs) in on-premises or cloud environments.



1) In the Kubernetes architecture diagram above you can see, there is one master and multiple nodes.

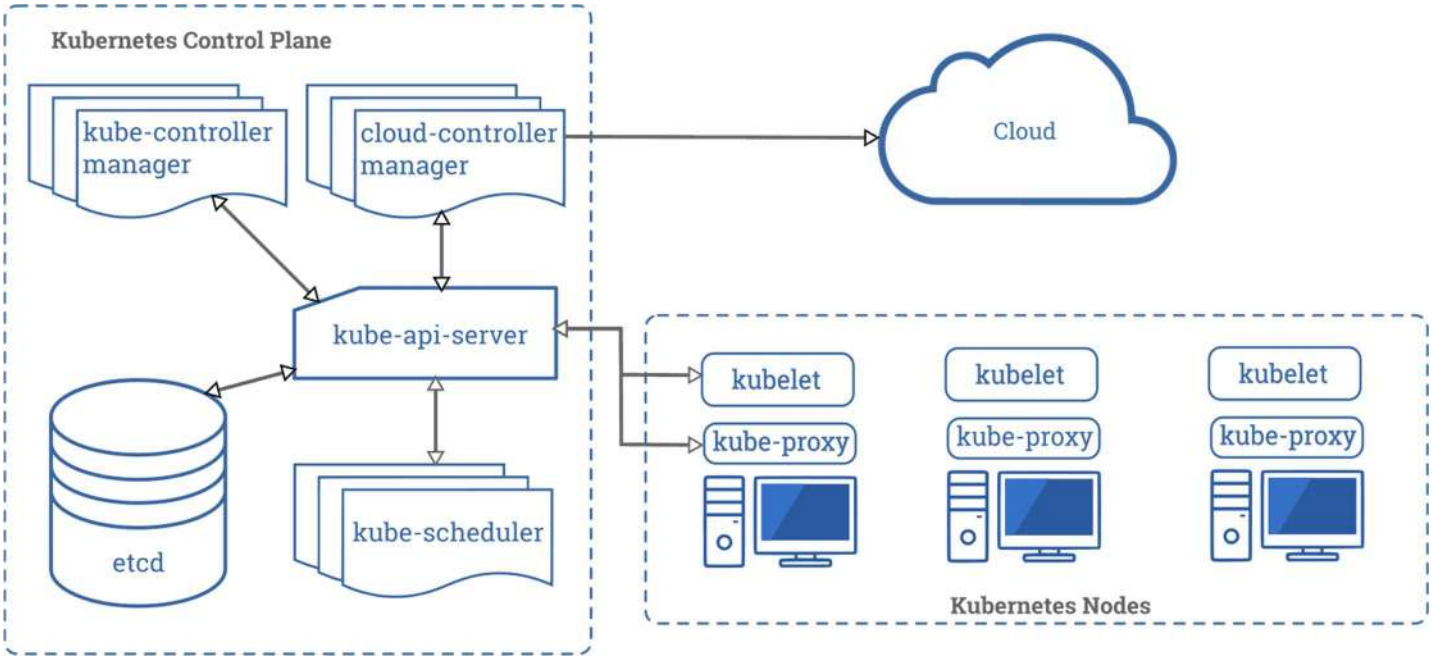
2) The Master node communicates with Worker nodes using Kube API-server to kubelet communication.

3) In the Worker node, there can be one or more pods and pods can contain one or more containers.

4) Containers can be deployed using the image also can be deployed externally by the user.

Read more about Kubernetes Network Policy here.

Kubernetes Architecture Components



Read more about Container Orchestration and Management Options

Kubernetes Master Node

In Kubernetes (k8s), a master node is the **control plane component** responsible for **managing the cluster**. It coordinates and schedules tasks, maintains cluster state, and monitors node health. It includes components like API server, scheduler, and controller manager, ensuring overall cluster functionality and orchestration of containerized applications.

Master Node Components:

1) Kube API server handles administrative tasks on the master node. Users send REST commands in YAML/JSON to the API server, which processes and executes them. The Kube API server acts as the front end of the Kubernetes control plane.

2) etcd, a distributed key-value store, maintains the cluster state and configuration details like subnets and config maps in Kubernetes' database. It's where Kubernetes stores its information.

Read More: Kubernetes for Testers.

3) Kube-scheduler assigns tasks to worker nodes and manages new requests from the API Server, ensuring they are directed to healthy nodes.

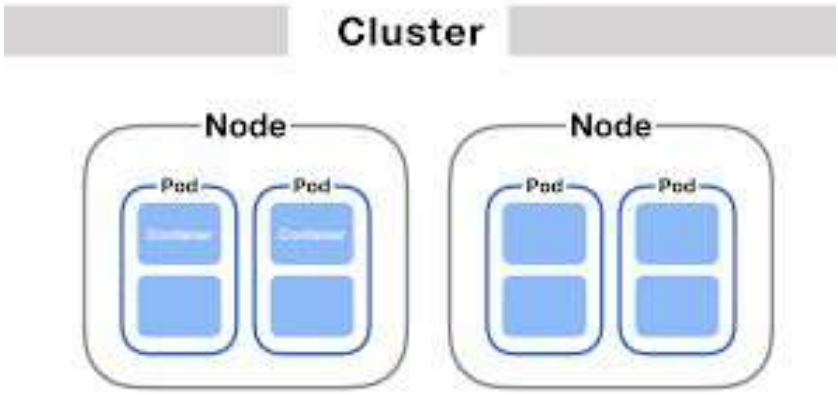
4) Kube Controller Manager task is to retrieve the desired state from the API Server. If the desired state does not match the current state of the object, corrective steps are taken by the control loop to align the current state with the desired state.

There are different types of control manager in Kubernetes architecture:

- **Node Manager:** It oversees nodes, creating new ones in case of unavailability or destruction.
- **Replication Controller:** It ensures the desired container count is maintained within the replication group.
- **Endpoints Controller:** This controller populates the endpoints object, connecting Services & Pods.

Q4. Difference between POD and node?

	Pod	Node
Description	The smallest deployable unit in a Kubernetes cluster	A physical or virtual machine
Role	Isolates containers from underlying servers to boost portability Provides the resources and instructions for how to run containers optimally	Provides the compute resources (CPU, volumes, etc) to run containerized apps
What it hosts	Application containers, supporting volumes, and similar IP addresses for logically similar containers	Pods with application containers inside them, kubelet

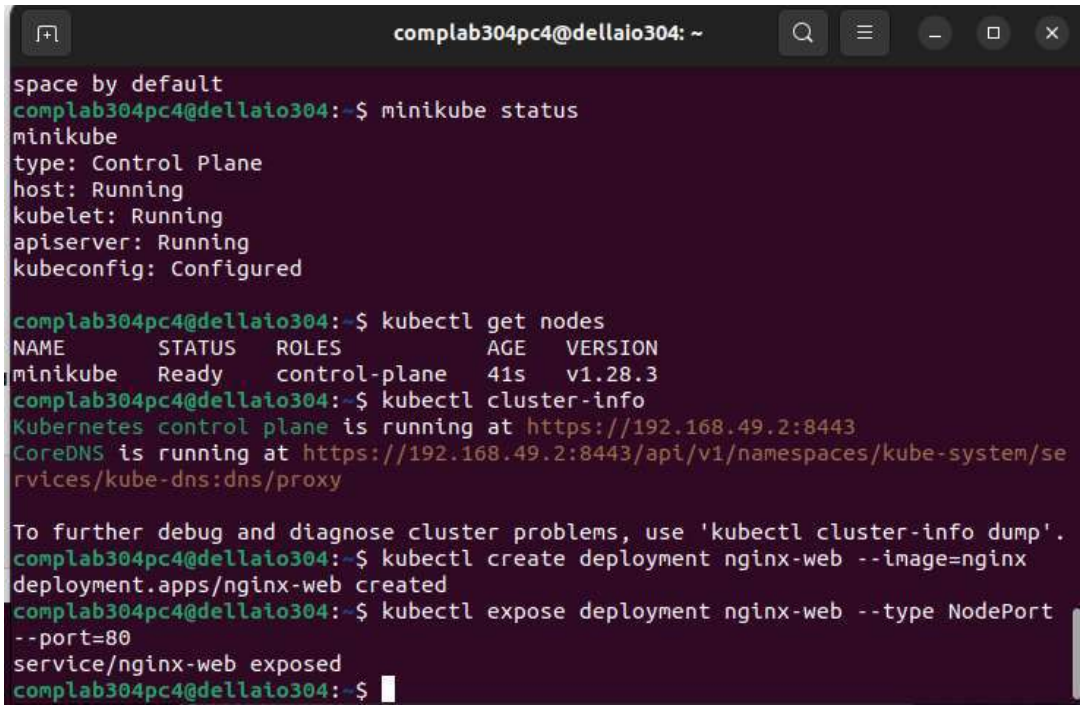


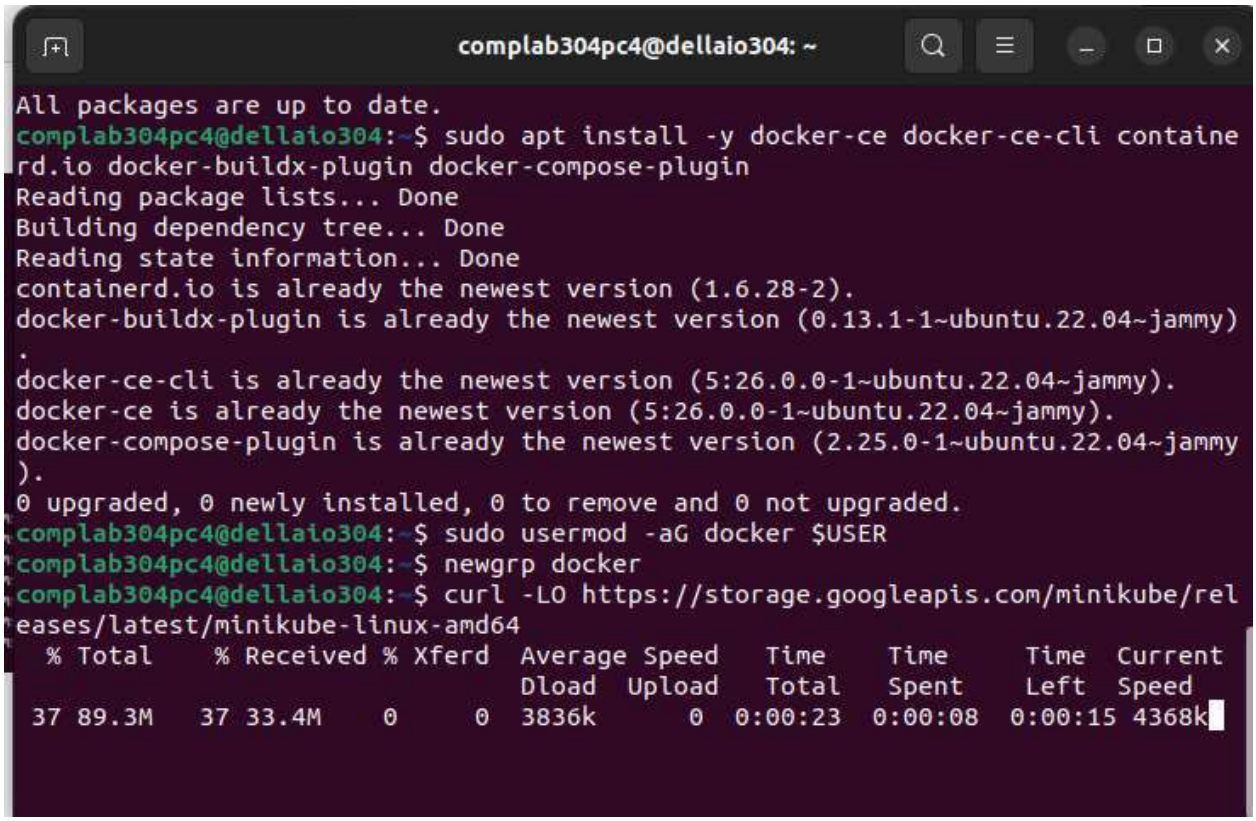
Q5. Compare Kubernetes and Docker Swarm

Point of comparison	Kubernetes	Docker Swarm
Installation	Complex	Comparatively simple
Learning curve	Heavy	Lightweight
GUI	Detailed view	No GUI, needs third party
Cluster setup	Easy	Easy
Availability features	multiple	minimal
Scalability	All-in-one scaling based on traffic	Values scaling quickly (approx. 5x faster than

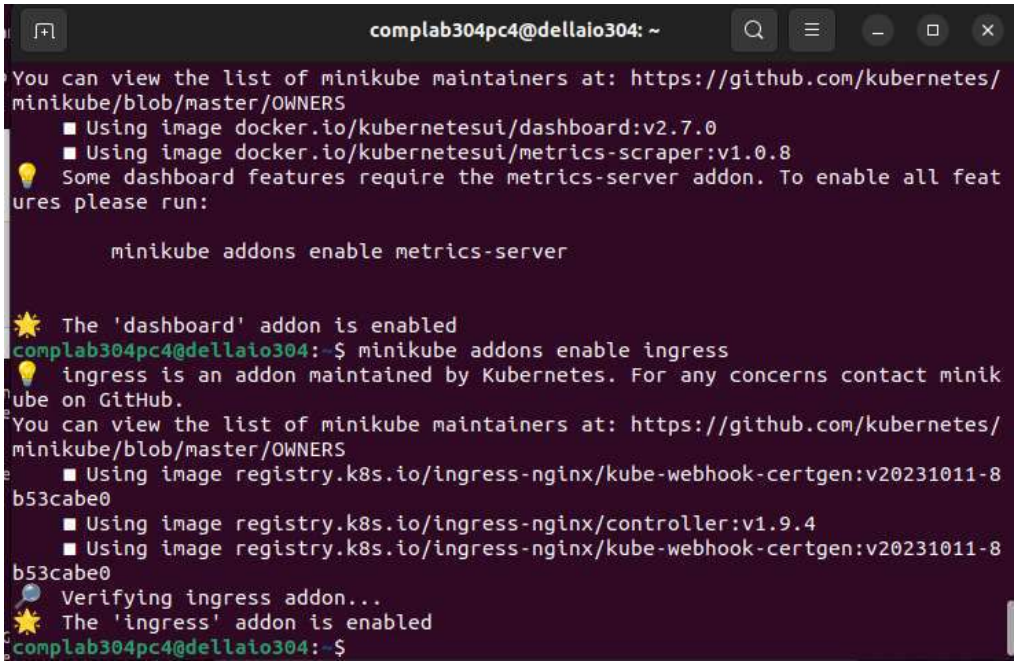
		K8s) over scaling automatically
Horizontal auto-scaling	Yes	No
Monitoring capabilities	Yes, built-in	No, needs third party
Load balancing	No built-in internal auto load balancing	Internal load balancing
Security features	Supports multiple security features	Supports multiple security features

Q6. Write steps and attach screenshots of implement container orchestration using Kubernetes

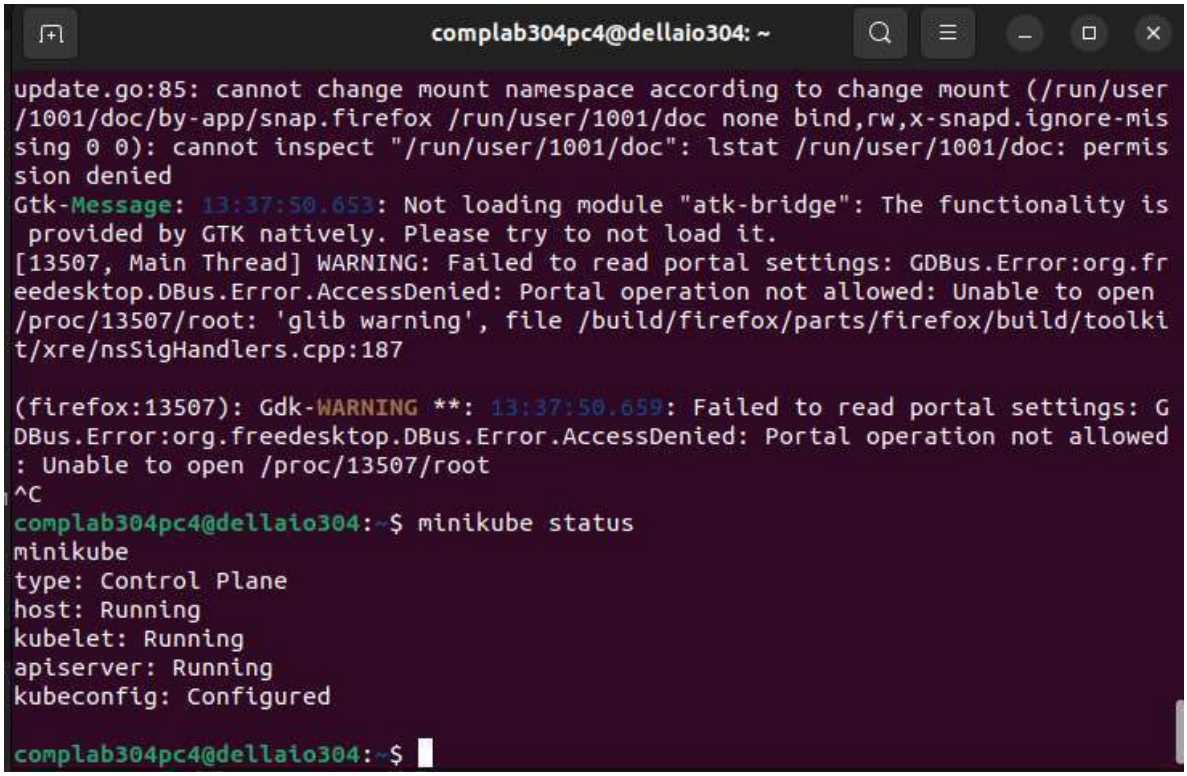

complab304pc4@dellaio304: ~
space by default
complab304pc4@dellaio304:~$ minikube status
minikube
type: Control Plane
host: Running
kubelet: Running
apiserver: Running
kubeconfig: Configured
complab304pc4@dellaio304:~$ kubectl get nodes
NAME STATUS ROLES AGE VERSION
minikube Ready control-plane 41s v1.28.3
complab304pc4@dellaio304:~$ kubectl cluster-info
Kubernetes control plane is running at https://192.168.49.2:8443
CoreDNS is running at https://192.168.49.2:8443/api/v1/namespaces/kube-system/services/kube-dns:dns/proxy
To further debug and diagnose cluster problems, use 'kubectl cluster-info dump'.
complab304pc4@dellaio304:~$ kubectl create deployment nginx-web --image=nginx
deployment.apps/nginx-web created
complab304pc4@dellaio304:~$ kubectl expose deployment nginx-web --type NodePort --port=80
service/nginx-web exposed
complab304pc4@dellaio304:~$


complab304pc4@dellaio304: ~
All packages are up to date.
complab304pc4@dellaio304:~$ sudo apt install -y docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-compose-plugin
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
containerd.io is already the newest version (1.6.28-2).
docker-buildx-plugin is already the newest version (0.13.1-1~ubuntu.22.04~jammy).
docker-ce-cli is already the newest version (5:26.0.0-1~ubuntu.22.04~jammy).
docker-ce is already the newest version (5:26.0.0-1~ubuntu.22.04~jammy).
docker-compose-plugin is already the newest version (2.25.0-1~ubuntu.22.04~jammy).
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
complab304pc4@dellaio304:~$ sudo usermod -aG docker $USER
complab304pc4@dellaio304:~$ newgrp docker
complab304pc4@dellaio304:~$ curl -LO https://storage.googleapis.com/minikube/releases/latest/minikube-linux-amd64
% Total % Received % Xferd Average Speed Time Time Time Current
Dload Upload Total Spent Left Speed
37 89.3M 37 33.4M 0 0 3836k 0 0:00:23 0:00:08 0:00:15 4368k

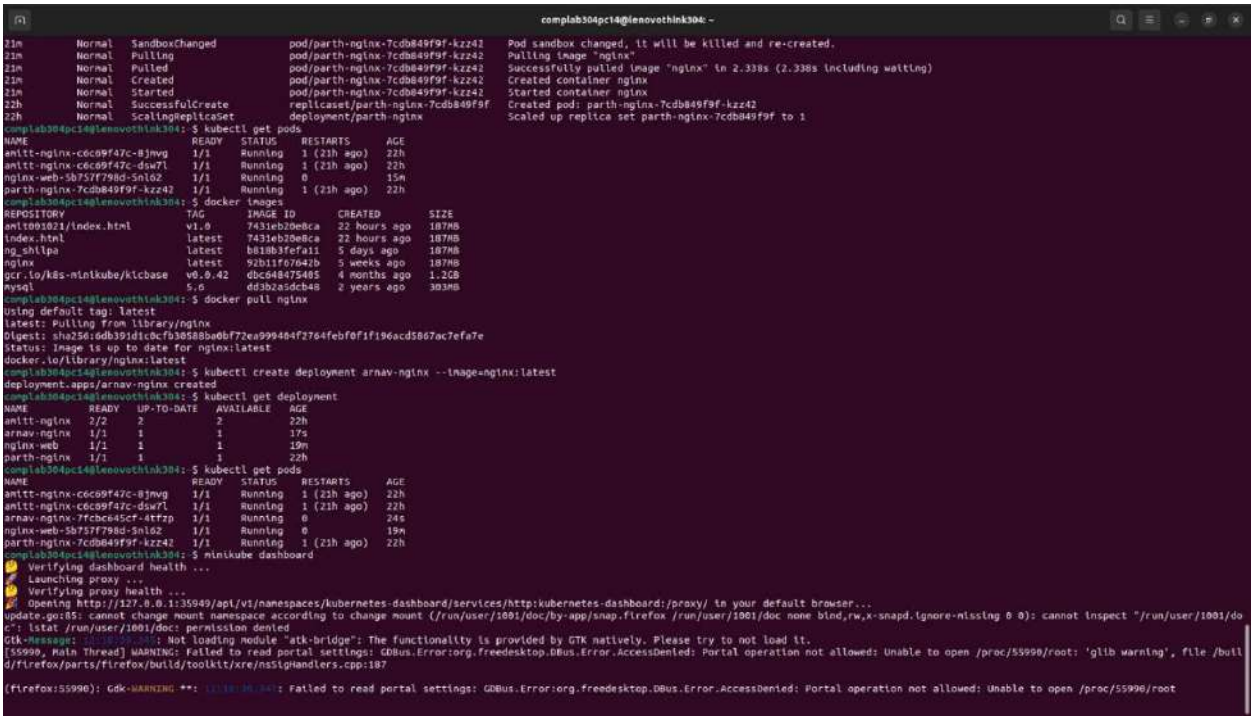
Installing Docker

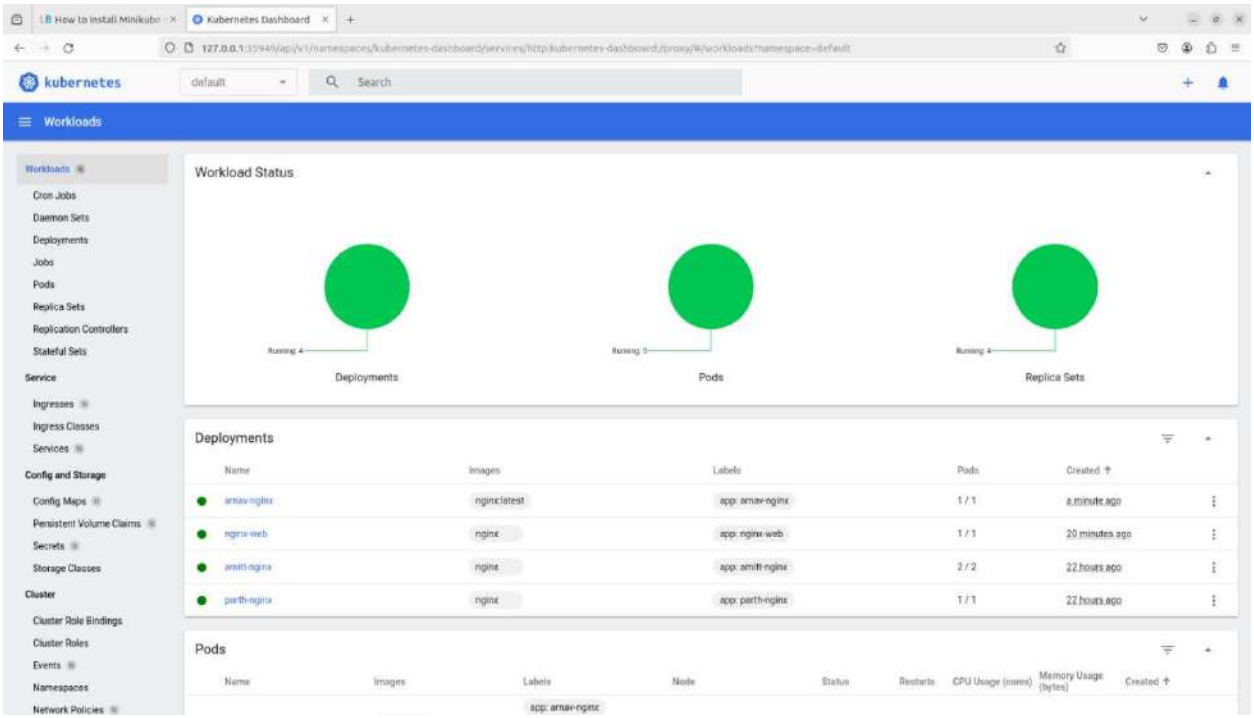


Minikube addons command to enable ingress

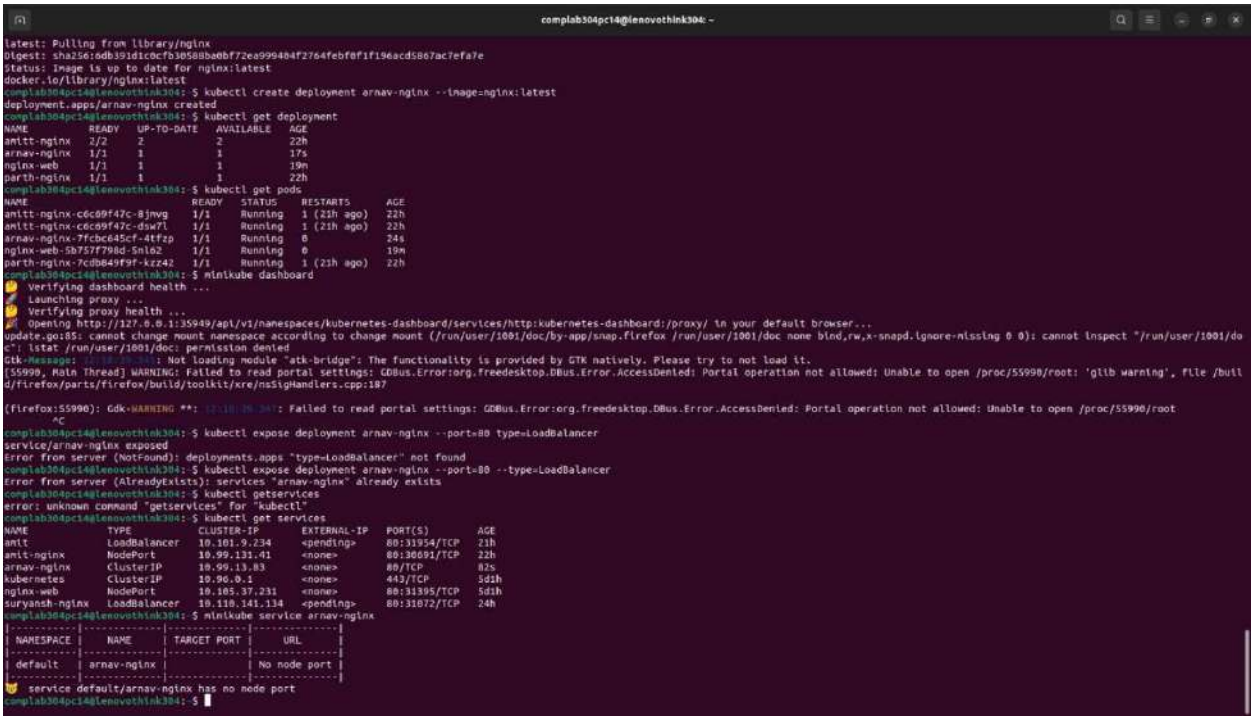


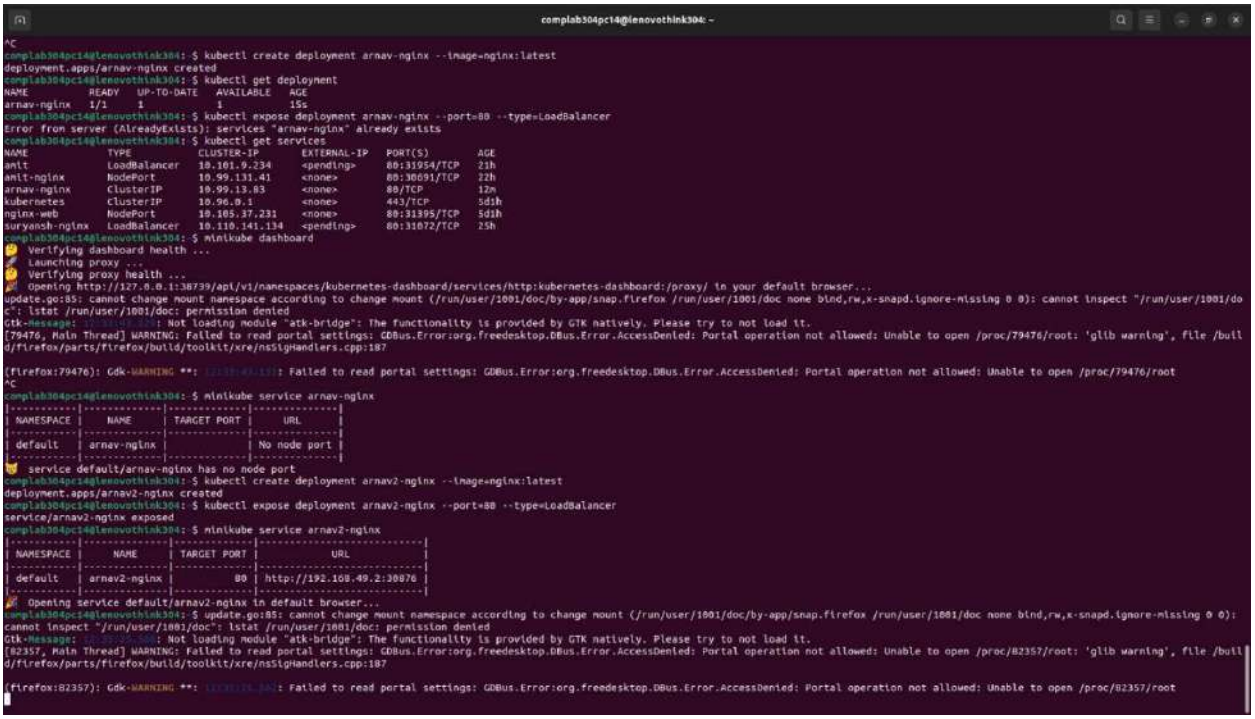
Minikube status to check the current status including the type, host, kubelet, apiserver etc



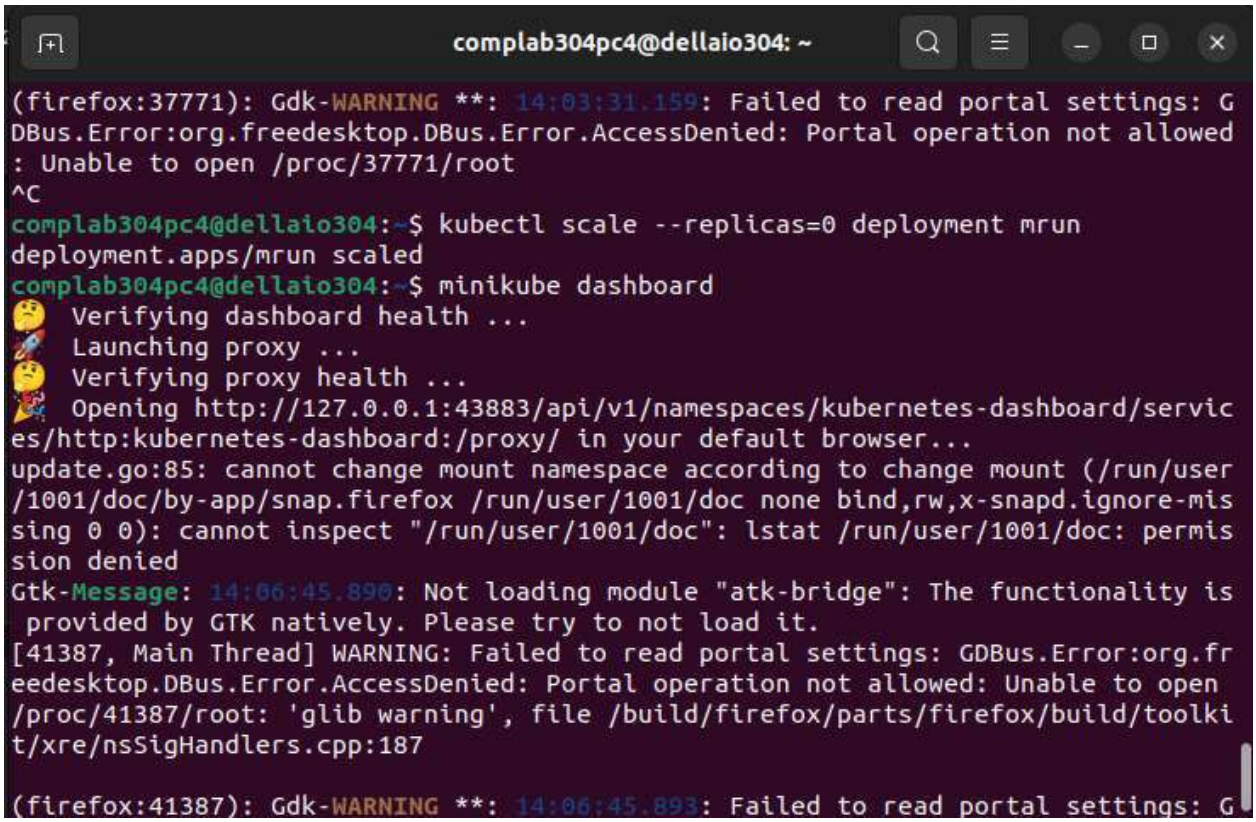


Minikube Dashboard which shows the no of pods, services, nodes which are currently running

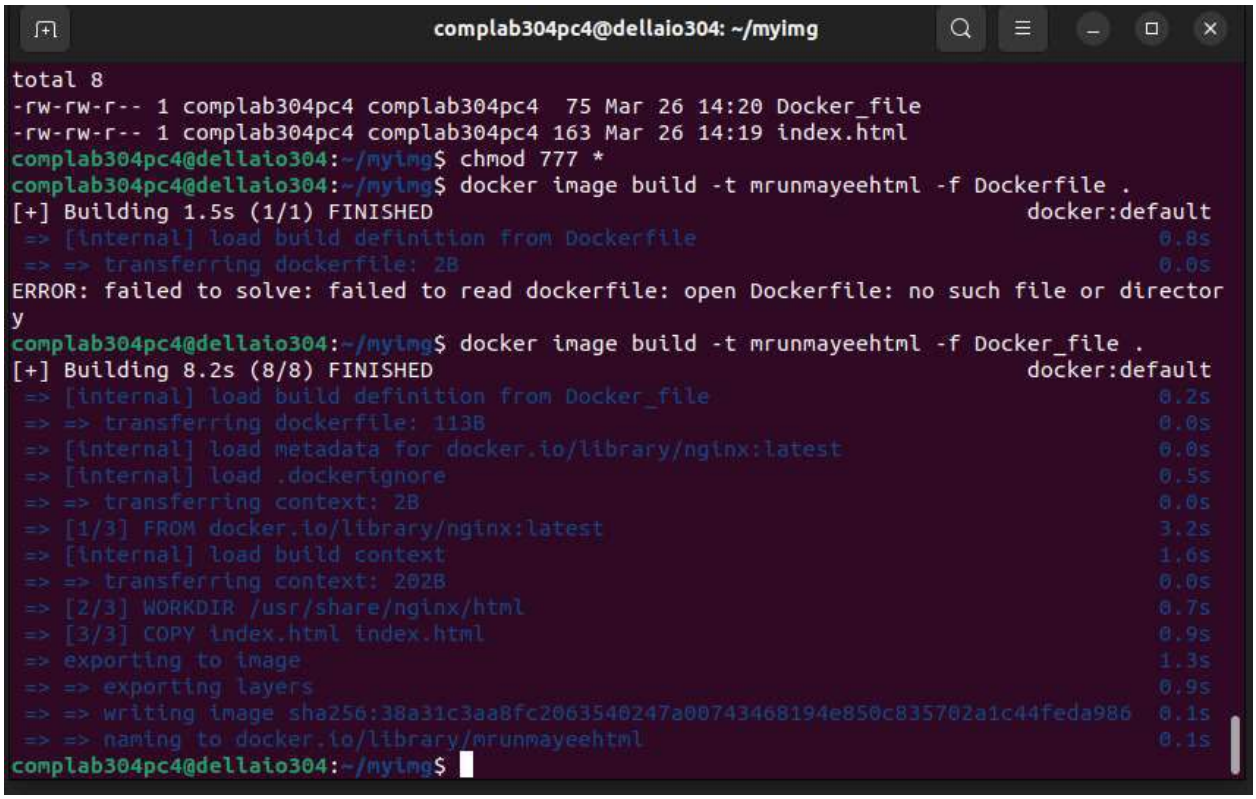

complab304pc14@lenovothink304: ~
latest: Pulling from library/nginx
Digest: sha256:6db391d1c0cfb30588ba0bf72ea999404f2764febf0f1f196acd5867ac7efa7e
Status: Image is up to date for nginx:latest
docker.io/library/nginx:latest
complab304pc14@lenovothink304:~$ kubectl create deployment arnav-nginx --image=nginx:latest
deployment.apps/arnav-nginx created
complab304pc14@lenovothink304:~$ kubectl get deployment
NAME READY UP-TO-DATE AVAILABLE AGE
amitt-nginx 2/2 2 2 22h
arnav-nginx 1/1 1 1 17s
nginx-web 1/1 1 1 19m
parth-nginx 1/1 1 1 22h
complab304pc14@lenovothink304:~$ kubectl get pods
NAME READY STATUS RESTARTS AGE
amitt-nginx-c6c89f47c-8jnvg 1/1 Running 1 (21h ago) 22h
amitt-nginx-c6c89f47c-dsw7l 1/1 Running 1 (21h ago) 22h
arnav-nginx-7fcbc645cf-4tfzp 1/1 Running 0 24s
nginx-web-5b757f798d-5nl62 1/1 Running 0 19m
parth-nginx-7cdb849f9f-kzz42 1/1 Running 1 (21h ago) 22h
complab304pc14@lenovothink304:~$ minikube dashboard
Verifying dashboard health ...
Launching proxy ...
Verifying proxy health ...
Opening http://127.0.0.1:35949/api/v1/namespaces/kubernetes-dashboard/services/http:kubernetes-dashboard:/proxy/ in your default browser...
update.go:85: cannot change mount namespace according to change mount (/run/user/1001/doc/by-app/snap.firefox /run/user/1001/doc none bind,rw,x-snapd.ignore-missing 0 0): cannot inspect "/run/user/1001/doc": lstat /run/user/1001/doc: permission denied
Gtk-Message: : Not loading module "atk-bridge": The functionality is provided by GTK natively. Please try to not load it.
[55990, Main Thread] WARNING: Failed to read portal settings: GDBus.Error:org.freedesktop.DBus.Error.AccessDenied: Portal operation not allowed: Unable to open /proc/55990/root: 'glib warning', file /build/firefox/parts/firefox/build/toolkit/xre/nsSigHandlers.cpp:187
(firefox:55990): Gdk-WARNING **: : Failed to read portal settings: GDBus.Error:org.freedesktop.DBus.Error.AccessDenied: Portal operation not allowed: Unable to open /proc/55990/root
^C
complab304pc14@lenovothink304:~$ kubectl expose deployment arnav-nginx --port=80 type=LoadBalancer
service/arnav-nginx exposed
Error from server (NotFound): deployments.apps "type=LoadBalancer" not found
complab304pc14@lenovothink304:~$ kubectl expose deployment arnav-nginx --port=80 --type=LoadBalancer
Error from server (AlreadyExists): services "arnav-nginx" already exists
complab304pc14@lenovothink304:~$ kubectl getservices
error: unknown command "getservices" for "kubectl"
complab304pc14@lenovothink304:~$ kubectl get services
NAME TYPE CLUSTER-IP EXTERNAL-IP PORT(S) AGE
amit LoadBalancer 10.101.9.234 <pending> 80:31954/TCP 21h
amit-nginx NodePort 10.99.131.41 <none> 80:30691/TCP 22h
arnav-nginx ClusterIP 10.99.13.83 <none> 80/TCP 82s
kubernetes ClusterIP 10.96.0.1 <none> 443/TCP 5d1h
nginx-web NodePort 10.105.37.231 <none> 80:31395/TCP 5d1h
suryansh-nginx LoadBalancer 10.110.141.134 <pending> 80:31872/TCP 24h
complab304pc14@lenovothink304:~$ minikube service arnav-nginx
| NAMESPACE | NAME | TARGET PORT | URL |
| default | arnav-nginx | | No node port |
service default/arnav-nginx has no node port
complab304pc14@lenovothink304:~$


complab304pc14@lenovothink304: ~
^C
complab304pc14@lenovothink304:~$ kubectl create deployment arnav-nginx --image=nginx:latest
deployment.apps/arnav-nginx created
complab304pc14@lenovothink304:~$ kubectl get deployment
NAME READY UP-TO-DATE AVAILABLE AGE
arnav-nginx 1/1 1 1 15s
complab304pc14@lenovothink304:~$ kubectl expose deployment arnav-nginx --port=80 --type=LoadBalancer
Error from server (AlreadyExists): services "arnav-nginx" already exists
complab304pc14@lenovothink304:~$ kubectl get services
NAME TYPE CLUSTER-IP EXTERNAL-IP PORT(S) AGE
amit LoadBalancer 10.101.9.234 <pending> 80:31954/TCP 21h
amit-nginx NodePort 10.99.131.41 <none> 80:30691/TCP 22h
arnav-nginx ClusterIP 10.99.13.83 <none> 80/TCP 12m
kubernetes ClusterIP 10.96.0.1 <none> 443/TCP 5d1h
nginx-web NodePort 10.105.37.231 <none> 80:31395/TCP 5d1h
suryansh-nginx LoadBalancer 10.110.141.134 <pending> 80:31872/TCP 25h
complab304pc14@lenovothink304:~$ minikube dashboard
Verifying dashboard health ...
Launching proxy ...
Verifying proxy health ...
Opening http://127.0.0.1:38739/api/v1/namespaces/kubernetes-dashboard/services/http:kubernetes-dashboard:/proxy/ in your default browser...
update.go:85: cannot change mount namespace according to change mount (/run/user/1001/doc/by-app/snap.firefox /run/user/1001/doc none bind,rw,x-snapd.ignore-missing 0 0): cannot inspect "/run/user/1001/doc": lstat /run/user/1001/doc: permission denied
Gtk-Message: : Not loading module "atk-bridge": The functionality is provided by GTK natively. Please try to not load it.
[79476, Main Thread] WARNING: Failed to read portal settings: GDBus.Error:org.freedesktop.DBus.Error.AccessDenied: Portal operation not allowed: Unable to open /proc/79476/root: 'glib warning', file /build/firefox/parts/firefox/build/toolkit/xre/nsSigHandlers.cpp:187
(firefox:79476): Gdk-WARNING **: : Failed to read portal settings: GDBus.Error:org.freedesktop.DBus.Error.AccessDenied: Portal operation not allowed: Unable to open /proc/79476/root
^C
complab304pc14@lenovothink304:~$ minikube service arnav-nginx
| NAMESPACE | NAME | TARGET PORT | URL |
| default | arnav-nginx | | No node port |
service default/arnav-nginx has no node port
complab304pc14@lenovothink304:~$ kubectl create deployment arnav2-nginx --image=nginx:latest
deployment.apps/arnav2-nginx created
complab304pc14@lenovothink304:~$ kubectl expose deployment arnav2-nginx --port=80 --type=LoadBalancer
service/arnav2-nginx exposed
complab304pc14@lenovothink304:~$ minikube service arnav2-nginx
| NAMESPACE | NAME | TARGET PORT | URL |
| default | arnav2-nginx | 80 | http://192.168.49.2:30876 |
Opening service default/arnav2-nginx in default browser...
complab304pc14@lenovothink304:~$ update.go:85: cannot change mount namespace according to change mount (/run/user/1001/doc/by-app/snap.firefox /run/user/1001/doc none bind,rw,x-snapd.ignore-missing 0 0): cannot inspect "/run/user/1001/doc": lstat /run/user/1001/doc: permission denied
Gtk-Message: : Not loading module "atk-bridge": The functionality is provided by GTK natively. Please try to not load it.
[82357, Main Thread] WARNING: Failed to read portal settings: GDBus.Error:org.freedesktop.DBus.Error.AccessDenied: Portal operation not allowed: Unable to open /proc/82357/root: 'glib warning', file /build/firefox/parts/firefox/build/toolkit/xre/nsSigHandlers.cpp:187
(firefox:82357): Gdk-WARNING **: : Failed to read portal settings: GDBus.Error:org.freedesktop.DBus.Error.AccessDenied: Portal operation not allowed: Unable to open /proc/82357/root

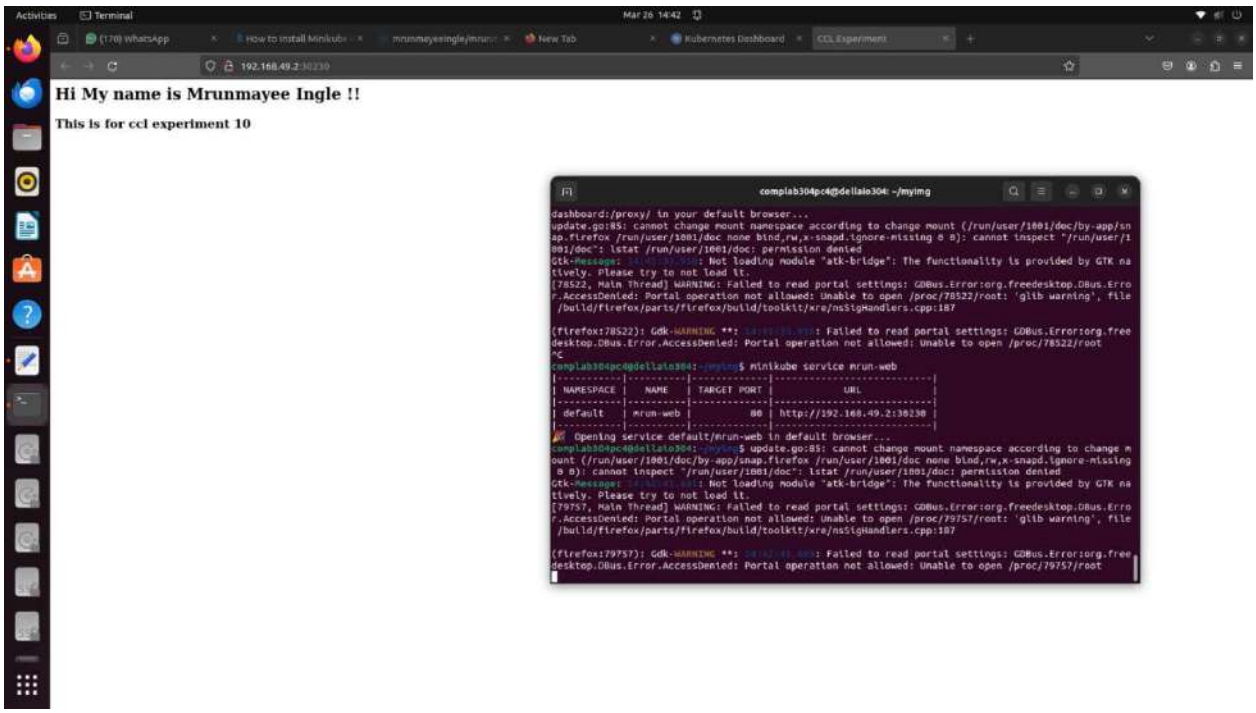
Creating a new deployment mrunmayee-nginx using kubectl.



Viewing all the services and pods currently running



Pushing an image to the Docker hub and then displaying it using kubectl



CCL

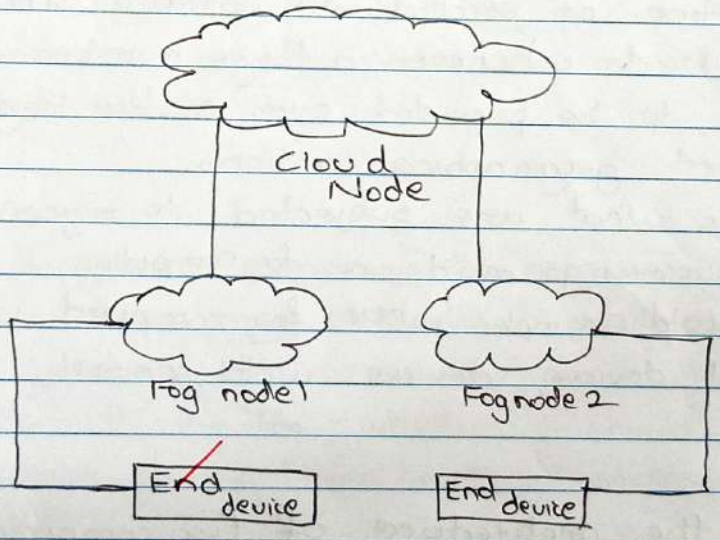
THADOMAL SHAHANI TSEC ENGINEERING COLLEGE

Assignment :- 1

Q1 What is fog computing? Explain the need of fog computing.

Ans 1 Fog computing is the term coined by cisco that refers to extending cloud computing to an edge of the enterprise's network. It facilitates the operation of computing, storage and networking services between end devices and computing data centres.

(Figure A)



Fog Computing Architecture

1) The devices comprising the fog infrastructure are known as fog nodes
2) In fog computing all storage capabilities, computation capabilities, data along with the applications are placed between the cloud and physical host.
3) All these functionalities are placed more towards the host. This makes processing faster as it is done almost at the place where data is created.

4) It improves the efficiency of the system and is also used to ensure increased security.

Need of fog computing:-

1) It is used when only selected data is required to send to the cloud. This selected data is chosen for long term storage and is less frequently accessed by the host.
2) It is used when data should be analyzed within a fraction of seconds i.e Latency should be low.
3) It is used whenever a large number of services needed to be provided over a ~~too~~ large area at different geographical locations.
4) Devices what are subjected to rigorous computations and processings must use fog computing.
5) Real world examples where fog computed is used are In IoT devices, devices with sensors, cameras, etc.

Q2 Explain the architecture of Fog computing.

Ans 2 Refer Figure A for architecture diagram.

1) Physical and virtual nodes (end devices):-
End devices serve as the points of contact to the real world, be it application servers, edge routers, end devices such as mobile phones and smartwatches, or sensors. These devices are data generators and can span a large spectrum of technology. This means they may have varying storage and processing capacities and different underlying software and hardware.

THADOMAL SHAHANI TSEC ENGINEERING COLLEGE

2) Fog nodes:-
They are independent devices that pick up the generated information. Fog nodes fall under 3 categories: fog devices, fog servers and gateways. These devices store necessary data while fog servers also compute this data to decide the course of action.

3) Monitoring Services:-
Monitoring services usually include application programming interfaces that keep track of systems performance and resource availability. Monitoring systems ensure that all end devices and fog nodes are up and communication isn't stalled.

4) Data processors:-
Data processors are programs that run on fog nodes. They filter, trim and sometimes even reconstruct faulty data that flows from end devices. Data processors are in charge of deciding what to do with the data - whether it should be stored locally on a fog server or sent for long-term storage in the cloud.

5) Resource manager:-
The resource manager allocates and deallocates resources to various nodes and schedules data transfer between nodes and the cloud. Also takes care of data backup, ensuring zero data loss.

6) Security tools:-
Encryption is a must since all communication tends to happen over wireless networks. End users directly ask the fog nodes for data in some cases.

7) Applications:-
Applications provide services to end-users. They use data provided by fog computing system to provide quality servire while ensuring cost-effectiveness.

Q3 Applications of fog computing:-
1) It can be used to monitor and analyze the patient's condition. In case of emergency, doctors can be alerted.
2) It can be used for real time rail monitoring as for high speed trains we want as little latency as possible.
3) It can be used for gas and oils pipeline optimization. It generates a huge amount of data and is ineffecient to store all data into the cloud for analysis.

Q4 Compare Fog and Cloud computing.
Ans 4

Cloud computing	Fog Computing
1) Has high latency.	1) Has low latency
2) Low response time.	2) High response time.
3) Has low security.	3) Has high security.
4) Access speed is high depending on the VM connectivity	4) Even higher speed compared to cloud computing.
5) Multiple data sources can be integrated.	5) Multiple data sources and devices can be integrated.
6) Mobility is limited.	6) Mobility is supported.
7) Few number of server nodes.	7) Large number of server nodes.
8) Services are provided within the internet.	8) Services are provided at the edge of local network.

THADOMAL SHAHANI TSEC ENGINEERING COLLEGE

Q5 Case study of fog computing.

Ans 5 Case Study: Fog Computing in Precision Agriculture

Introduction:-
Precision agriculture utilizes technology to optimise farming practices by collecting and analyzing data from various sources like sensors, drones and satellites.

Challenges faced by Traditional Computing:-
1) Latency: Real time decision making in agriculture is crucial. Sending data to the cloud and receiving responses can introduce significant delays.
2) Bandwidth: Uploading large amounts of agricultural data, especially real time sensor readings, can be bandwidth-intensive and expensive.
3) Connectivity: Remote farms may have unreliable or limited internet connectivity, hindering communication with the cloud.

Solution: Fog Computing
Fog computing addresses these challenges by processing data closer to the source, often on edge devices or local servers. In precision agriculture, fog nodes can be deployed:
1) On-farm: Servers or dedicated ~~for~~ devices collect data from sensors and perform initial processing like filtering and aggregation.
2) At field edges: Mobile fog nodes can be mounted on tractors or drones, processing data collected during operations.

Benefits :-

1) Reduced latency : Fog computing allows for real time data analysis, enabling farmers to make prompt decisions on irrigation, spraying and harvesting.
2) Bandwidth optimisation : Only processed data instead of raw sensor readings, is sent to the cloud, significantly reducing bandwidth usage.
3) Offline Functionality : Fog nodes can operate even with limited or no internet connectivity, ensuring critical operations continue uninterrupted.
4) Improved decision making :- local data analysis can provide valuable insights for farmers.

Challenges :-

1) Security :- Securing fog nodes and ensuring data privacy requires care consideration and implementation of security protocols.
2) Resource Management : Managing and maintaining distributed fog nodes across a large farm can be complex.
3) Standardisation : The lack of standardised fog computing platforms can pose challenges in scalability.

24/2/24

CCL

THADOMAL SHAHANI TSEC ENGINEERING COLLEGE

Assignment :- 2

Q1 Explore and compare the similar types of services provided by AWS, Azure and Google Cloud Platform.

Ans 1 1) Compute Services :

AWS - Amazon Elastic Compute Cloud (EC2) -

Provides resizable compute capacity in the cloud, allowing users to quickly scale virtual servers up or down. Supports a wide range of instance types optimised for different use cases, such as general purpose, compute-optimized, memory optimized and storage optimised instances

Azure - Azure Virtual Machines (VMs) -

Offers a variety of VM sizes and types, including general-purpose, compute optimised, memory optimised and storage optimised VMs. Provides features like Azure spot VMs for cost effective, interruptible compute capacity, Azure Virtual Machine scale sets for automatic scaling of VM instances and Azure Dedicated Host for physical servers dedicated to hosting VMs.

GCP - Google Compute Engine (GCE)

Offers virtual machines with customisable CPU, memory and disk configurations, including predefined machine types and custom machine types. Provides features like managed instance groups for auto-scaling and load balancing, preemptible VMs for short lived, cost effective compute capacity and sole-tenant nodes for dedicated physical servers.

2) Storage Services:

AWS - Amazon Simple Storage Service (S3) -
Provides highly durable and scalable object storage with a simple web service interface. offer features like versioning for maintaining multiple versions of objects, lifecycle policies for automating data management and encryption for securing data at rest and in transit. Supports various storage classes, including Standard, Standard IA, One zone - IA and Glacier Deep Archive with different pricing and availability options.

Azure - Azure Blob Storage -
offers scalable object storage for storing unstructured data such as documents, images, videos and logs. Provides features like lifecycle management for automating data tiering and deletion, blob versioning for maintaining multiple versions of blobs and Azure Blob Index for efficiently querying metadata. Supports various access tiers including Hot, Cool and Archive with different pricing based on data access frequency and retention duration.

GCP - Google Cloud Storage -
Provides durable and scalable object storage with global edge-caching capabilities for fast content delivery. offers features like object lifecycle management for automating data retention and deletion, object versioning for maintaining historical versions of objects, and object-level access control for fine grained security. Supports different storage classes, including standard, nearline, coldline, and archive with different pricing based on data access frequency and availability requirements.

3) Database Services:

AWS - Amazon Relational Database Service (RDS) -
offers managed relational database services for popular database engines such as MySQL, Postgre SQL, Maria DB, oracle and SQL server. Provides features like automated backups, automated scaling, multi-Az deployments for high availability and read replicas for scaling read workloads. Support features like encryption at rest and in transit, database parameter groups for customising database configurations, and database snapshots for point-in-time recovery.

Azure - Azure SQL Database -
Provides fully managed SQL database services with built in high availability, automatic tuning and intelligent performance optimization. offers features like automated backups, geo-replication for disaster recovery, transparent data encryption and advanced threat protection for securing data. Supports features like Azure SQL Managed instances for migrating on premises SQL Server workloads to the cloud with minimal changes.

GCP - Google Cloud SQL -
offers managed MySQL, Postgre SQL and SQL server databases with automated backups, replication and scaling. Provides features like automated patching, point in time recovery and cross region replication for dist disaster recovery. Supports features like private IP connectivity, IAM database authentication, and database flags for customizing database behaviour.

4) Networking Services:

AWS - Amazon Virtual Private Cloud (VPC) -
Allows users to create isolated sections of the AWS cloud with configurable IP addressing route tables and network access control policies. Offers features like VPC peering for connecting VPCs within the same or different AWS accounts, AWS Direct connect for dedicated network connectivity and AWS VPN for secure site to site VPN connections.

Azure - Azure Virtual Network -
Provides networking functionality such as virtual private network (VPN), Azure Express Route for dedicated private connectivity and Azure Bastion for secure RDP and SSH access to VMs. offers features like Azure Virtual Network Peering for connecting virtual networks within the same or different Azure regions, Azure Load Balancer for distributing inoming traffic and Azure Application Gateway for application level load balancing.

GCP - Google Virtual Private Cloud (VPC)
Offers global virtual networking for creating and managing private, secure networks with customisable IP addressing and routing. Provides feature like VPC peering for connecting VPCs within the same or different GCP projects, cloud VPN for secure site to site VPN connection and cloud interconnect for dedicated connectivity to Google Cloud.

5) Serverless Computing:

AWS - AWS lambda -
Allows users to run code without provisioning for manging servers, paying only for the compute time consumed. Offers features like AWS lambda layers for sharing code and libraries across

functions, AWS Lambda @Edge for running code at the edge locations of Amazon Cloud Front, and AWS Lambda Destinations for asynchronous error handling and retries.

Azure- Azure Functions-

Enables serveless computing with an event driven on demand execution of code in response to events like HTTP requests, ~~cost~~ messages, queues and timers. Offers ~~fa~~ features like durable functions for orchestrating long running workflows, Azure functions proxies for managing API endpoints and routes.

GCP - Google Cloud Functions-

Provides serverless execution of code in response to events from google cloud services like cloud storage, cloud Pub/Sub and cloud Firestore. offers features like background functions for processing events asynchronously, HTTP functions for serving HTTP requests and cloud functions for Firebase for building serverless applications on Firebase.

6) Machine Learning Services:

AWS - Amazon SageMaker -

Provides fully managed machine learning services for building training and deploying machine learning models at scale. offers features like built in algorithms for common ML tasks, automated model tuning ~~options~~ for optimizing model performance and model hosting for deploying models as endpoints for inference.

Azure - Azure Machine Learning -

offers a comprehensive set of tools and services for building and deploying machine ~~learni~~ learning models, including automated ML, model interpretability, and ML ops. Provides features like Azure Machine Learning Designer for building ML pipelines without writing code, Azure machine learning pipelines for orchestrating and automating ML workflows and Azure machine learning interpretability for understanding model predictions.

THADOMAL SHAHANI TSEC ENGINEERING COLLEGE

GCP - Google Cloud AI Platform -
Provides machine learning services for data scientists and developers to build, train and deploy AI models including managed ML services, AutoML and TensorFlow. Offers features like Cloud AI platform Notebooks for collaborative ML development in Jupyter notebooks, Cloud AI platform Jobs for running distributed training jobs and cloud AI prediction for serving ML models at scale.

7) Big Data and Analytics Services:
AWS - Amazon EMR (Elastic MapReduce) -
offers managed Hadoop and spark clusters for processing large scale data with scalable compute capacity and storage. Provides features like EMRFS for accessing data stored in Amazon S3, EMR notebooks for interactive data analysis in Zeppelin or Jupyter notebooks and EMR studio for building and collaborating on big data applications.

Azure - Azure HDInsight -
Provides managed Hadoop, spark and other big data clusters in the cloud with scalable compute and storage resources. Offers features like HDInsight Interactive Query for ad-hoc querying of big data with low latency responses, HDInsight Kafka for building real time streaming applications and HDInsight Enterprise Security Package for securing data and clusters.

GCP - Google Cloud Dataproc -
offers managed Apache Spark and Hadoop clusters for big data processing with scalable compute and storage resources. Provides features like Dataproc Workflow Templates for automating and orchestrating complex data workflows, Dataproc Jobs API for submitting and managing batch and streaming jobs and Dataproc Metastore for managing and serving metadata for data assets.

THADOMAL SHAHANI
TSEC
ENGINEERING COLLEGE

8) Container Services:

AWS- Amazon Elastic Kubernetes Service (EKS)-
Provides managed kubernetes clusters for orchestrating containerized applications with scalable compute and storage resources. Offers features like EKS fargate for running kubernetes pods without managing underlying infrastructure, EKS Anywhere for deploying and managing kubernetes clusters on premises or in hybrid environments and EKS Add ons for extending cluster capabilities with monitoring, logging and networking.

Azure- Azure Kubernetes Service (AKS)-
Offers managed kubernetes orchestration for deploying and managing containerised applications with built-in monitoring, scaling and security features. Provides features like AKS Virtual Nodes for running kubernetes pods on Azure container instances and AKS Service Mesh for managing microservices communication.

GCP - Google Kubernetes Engine (GKE)-
Provides managed kubernetes clusters with automated scaling, monitoring and upgrades for deploying containerized applications at scale. Offers features like GKE Autopilot for fully managed kubernetes clusters with automated operations and optimisation, GKE Anthos for deploying and managing kubernets clusters across hybrid and multi coloured environments.

9) Identity and Access Management (IAM):

AWS- AWS Identity and Access Management (IAM)-
Provides centralised control and management of user access to AWS services and resource with fine grained permissions and policies. Offers features like IAM Roles for delegating permissions to entities within or outside AWS accounts, IAM policies for defining permissions at a granular level and IAM access Analyzer for identifying resources with unintended access permissions.

Azure - Azure Active Directory (AAD)-
Provides identity and access management services for Azure and other Microsoft services with single sign on, multi factor authentication and conditional access capabilities. offers features like Azure AD B2C for customer identity and access management, Azure AD B2B for external user collaboration and Azure AD Domain services for extending on promise Active Directory to the cloud.
GCP - Google Cloud Identity and Access Management (Cloud IAM)-
offers centralised access control for managing user permissions and resources across Google cloud services with fine grained access control and resource hierarchies. Provides features like IAM conditions for adding context based access controls to IAM policies, IAM roles for defining granular permissions for specific tasks and IAM service accounts for authenticating applications.

10) Content Delivery Network (CDN):
AWS - Amazon CloudFront -
Provides a global content delivery network for catching content with low latency and high transfer speeds to users worldwide. offers features like edge locations for catching content closer to end users, real time logs for monitoring and analysing CDN traffic and AWS WAF for protecting web applications from common security threats.
Azure - Azure Content Delivery Network (CDN)-
offers global distribution of content with edge catching and fast delivery for websites, applications and media assets. Provide features like dynamic site acceleration for improving web performance, real time analytics for monitoring CDN traffic and custom rules engine for configuring CDN behaviour.
GCP - Google Cloud CDN -
Provides low latency content delivery with Google's global

network infrastructure for delivering static and dynamic content with high reliability and scalability. Offers features like cache invalidation for refreshing cached content, origin authentication for securing communication between CDN and origin servers and load balancing for distributing traffic across multiple CDN endpoints.

A+
20/3/24

THADOMAL SHAHANI
TSEC
ENGINEERING COLLEGE

CCL :- Mini Project

Shopkart : Ecommerce website for clothes.

Shopkart is an innovative ecommerce platform specialising in clothing, designed to provide customers with a seamless shopping experience. Built using React and Tailwind CSS, Shopkart offers a visually appealing and user friendly interface, ensuring a delightful browsing and purchasing journey for users. What sets Shopkart apart is its robust backend infrastructure using - AWS Cognito
- AWS DynamoDB
- AWS Amplify.

Amazon cognito ensures secure authentication and authorisation process, safeguarding sensitive user data.

Dynamo DB as a serverless no sql database, enables efficient storage and retrieval of product information ensuring real time updates on inventory and seamless read.

AWS Amplify serves as the linchpin, facilitating seamless communication between the frontend and backend components while also streamlining development and deployment processes. The synergy between React, Tailwind, Cognito, DynamoDB and Amplify empowers Shopkart to deliver a high performance, scalable and secure eCommerce platform, redefining experience.

10/4/24

CCL MINI-PROJECT

Title: E-commerce Website for Clothes Deployed in AWS Amplify

Team Members:

Arnav Malvia: 2103109
Rishab Mandal: 2103110
Tanay Mihani: 2103117

Table of Contents

Problem Statement:

In today's rapidly evolving digital landscape, the realm of online shopping has become increasingly integral to consumers' daily lives. However, amidst the plethora of e-commerce platforms available, the domain of clothing retailing remains ripe with challenges and untapped potential. Existing platforms often fall short in providing a tailored and immersive shopping experience, leaving users frustrated and dissatisfied. Consequently, there is a critical need to address these shortcomings and develop a sophisticated e-commerce solution tailored specifically for the clothing retail sector.

Enter "Shop Kart," a visionary e-commerce platform poised to revolutionize the online clothing shopping experience. Shop Kart recognizes the inherent challenges faced by users in navigating existing e-commerce websites, from limited personalization options to inefficient product discovery mechanisms. Our mission is to create a dynamic and user-centric platform that not only meets but exceeds the expectations of today's discerning shoppers.

At the heart of our endeavour lies the quest for personalized shopping experiences. Shop Kart aims to leverage advanced machine learning algorithms and data analytics to offer tailored product recommendations, size suggestions, and style inspirations based on users' preferences and past interactions. By harnessing the power of cloud computing technologies, we envision a platform that learns and adapts to each user's unique tastes and preferences, enhancing their overall shopping journey.

Moreover, Shop Kart seeks to revolutionize the way users discover new and exciting clothing items. Through intuitive search algorithms, visual browsing features, and curated collections, we aim to streamline the product discovery process, empowering users to explore and find their perfect wardrobe additions effortlessly. Our commitment to user experience extends beyond mere functionality; we envision a seamless and engaging interface that captivates users from their very first interaction with the platform.

In addition to elevating the front-end user experience, Shop Kart places significant emphasis on fortifying its back-end operations, particularly in the realm of inventory management. Recognizing the pivotal role inventory management plays in the seamless functioning of an eCommerce platform, Shop Kart commits to implementing robust systems that guarantee real-time stock tracking, size availability updates, and efficient

product categorization. Leveraging cloud-based infrastructure, the platform aims to revolutionize inventory management processes by harnessing the power of Amazon DynamoDB, Amazon Cognito, and AWS Amplify.

Amazon DynamoDB emerges as the backbone of Shop Kart's inventory management systems, offering unparalleled scalability, performance, and reliability. With DynamoDB, the platform can effortlessly handle the complexities of real-time stock tracking, accommodating fluctuations in inventory levels while ensuring lightning-fast access to critical data. Furthermore, DynamoDB's serverless architecture aligns seamlessly with Shop Kart's goal of optimizing operational efficiency, allowing the platform to scale dynamically based on workload demands without incurring unnecessary overhead costs.

Amazon Cognito, integrated seamlessly into Shop Kart's infrastructure, serves as the linchpin of authentication and authorization processes, safeguarding sensitive inventory data and ensuring that access is tightly controlled. By leveraging Cognito's robust identity management capabilities, Shop Kart fortifies its inventory management systems against unauthorized access, thereby bolstering data security and compliance.

Complementing DynamoDB and Cognito, AWS Amplify emerges as a game-changer in streamlining the development and deployment of inventory management features. With Amplify, Shop Kart can accelerate the implementation of inventory management functionalities while maintaining a laser focus on efficiency and scalability. Together, DynamoDB, Cognito, and Amplify form a formidable trifecta, propelling Shop Kart towards its mission of optimizing back-end operations and delivering unparalleled value to its customers.

Furthermore, Shop Kart prioritizes security and reliability in every aspect of its operations. Through seamless integration with trusted payment gateways and adherence to industry-leading security protocols, we strive to provide users with a secure and hassle-free shopping environment. Users can shop with confidence, knowing that their financial information is protected and their transactions are processed swiftly and securely.

In conclusion, Shop Kart embodies a vision of innovation, personalization, and excellence in online clothing retailing. Through cutting-edge technology, user-centric design, and a relentless commitment to quality, Shop Kart aims to redefine the online shopping experience, one satisfied customer at a time. Join us on this transformative journey as we pave the way for a new era of digital commerce in the world of fashion.

Technologies Used:

React and Tailwind:

At the heart of Shop Kart's front-end development lies React, a powerful JavaScript library renowned for its efficiency and flexibility in building interactive user interfaces. By adopting React's component-based architecture, we were able to break down complex UI elements into modular components, facilitating code reuse, maintainability, and scalability. Additionally, Tailwind CSS emerged as our preferred styling framework, offering a utility-first approach that streamlined the design process and empowered developers to create visually stunning interfaces with minimal effort. Together, React and Tailwind formed the backbone of Shop Kart's front-end infrastructure, enabling us to deliver a seamless and visually captivating user experience that sets the platform apart in the competitive e-commerce landscape.

AWS Services:

In the realm of cloud computing, Amazon Web Services (AWS) stands as a beacon of innovation and reliability, providing a comprehensive suite of services to power modern applications and services. Shop Kart leverages several AWS services to underpin its backend infrastructure and enhance its capabilities:

1. Amazon AWS Cognito:

With Amazon Cognito, you can add user sign-up and sign-in features and control access to your web and mobile applications. Amazon Cognito provides an identity store that scales to millions of users, supports social and enterprise identity federation, and offers advanced security features to protect your consumers and business. Built on open identity standards, Amazon Cognito supports various compliance regulations and integrates with frontend and backend development resources. Amazon Cognito serves as the backbone of Shop Kart's authentication and user management system, providing a secure and scalable solution for verifying user identities and managing user access to the platform. With Cognito, Shop Kart ensures a seamless and intuitive user experience, allowing users to sign up and sign in effortlessly across multiple devices and platforms. Cognito's robust authentication mechanisms, including multi-factor authentication (MFA) and adaptive authentication, enhance security and protect against unauthorized access. Additionally, Cognito's user pools and identity pools enable Shop Kart to define custom user attributes, manage user permissions, and integrate with external identity providers seamlessly. By leveraging Cognito's

features, Shop Kart builds trust and confidence among its users while maintaining the highest standards of security and compliance.

2. Amazon AWS DynamoDB:

Amazon DynamoDB is a serverless, NoSQL database service that enables you to develop modern applications at any scale. As a serverless database, you only pay for what you use and DynamoDB scales to zero, has no cold starts, no version upgrades, no maintenance windows, no patching, and no downtime maintenance. DynamoDB offers a broad set of security controls and compliance standards. DynamoDB global tables is a multi-Region, multi-active database with a 99.999% availability SLA and increased resilience. DynamoDB reliability is supported with managed backups, point-in-time recovery, and more. With DynamoDB streams, you can build serverless event-driven applications. Amazon DynamoDB powers Shop Kart's data storage and retrieval system, providing unmatched scalability, performance, and reliability for handling the platform's growing volume of data. DynamoDB's flexible data model and seamless scalability allow Shop Kart to store and retrieve product information, user profiles, and transactional data with ease, ensuring optimal performance and responsiveness. DynamoDB's built-in encryption at rest and in transit, along with granular access controls and fine-grained permissions, ensure that sensitive data is securely stored and protected against unauthorized access. Furthermore, DynamoDB's global tables and multi-region replication capabilities enable Shop Kart to deliver a seamless and consistent user experience to customers worldwide, regardless of their geographic location. By leveraging DynamoDB's features, Shop Kart maintains a competitive edge in the e-commerce market, offering users a reliable and high-performance platform for their shopping needs.

3. Amazon AWS Amplify:

AWS Amplify empowers Shop Kart's backend infrastructure, offering a comprehensive set of tools and services for building, deploying, and scaling cloud-based applications. Amplify's rich feature set, including authentication, data storage, and serverless computing, accelerates the development process and enables Shop Kart to deliver new features and updates to market quickly. Amplify's pre-built UI components, libraries, and SDKs simplify common development tasks, reducing development time and effort. Additionally, Amplify's seamless integration with other AWS services, such as Amazon Cognito and Amazon DynamoDB, ensures interoperability and compatibility across the platform. Amplify's built-in analytics and monitoring capabilities provide valuable insights into user behavior, application performance, and resource utilization, enabling Shop Kart to optimize the user experience and drive business growth. By harnessing Amplify's features, Shop Kart maintains agility, scalability, and reliability, positioning itself as a leader in the competitive e-commerce landscape.

Requirement Specification:

1. Functional Requirements:

a. User Management:

- User registration: Allow users to create accounts with email and password or social media authentication.

- User login: Provide secure authentication mechanisms for users to log in to their accounts.

- User profile management: Allow users to update their profile information, including shipping addresses and payment methods.

b. Product Management:

- Product listing: Display products in various categories with details such as name, description, price, and images.

- Product filtering: Enable users to filter products based on criteria such as price range, brand, size, color, etc.

- Product recommendations: Implement recommendation algorithms to suggest related or recommended products to users based on their browsing and purchase history.

c. Shopping Cart and Checkout:

- Shopping cart: Allow users to add products to their cart, view cart contents, update quantities, and remove items.

- Checkout process: Provide a seamless checkout process with multiple payment options, order summary, and shipping details.

- Order tracking: Enable users to track the status of their orders and view order history.

d. Security:

- Secure authentication: Implement secure authentication mechanisms using encryption protocols and best practices to protect user credentials.

- Data encryption: Encrypt sensitive data such as user information, payment details, and order information to ensure data privacy and security.

- Secure payments: Integrate with secure payment gateways to process transactions securely and protect users' financial information.

2. Non-functional Requirements:

a. Performance:

- Response time: Ensure fast response times for page loads, search queries, and checkout processes to provide a seamless user experience.

- Load balancing: Implement load balancing mechanisms to distribute incoming traffic evenly across multiple servers to prevent overload.

b. Reliability:

- High availability: Ensure the website is always available and accessible to users with minimal downtime.

- Fault tolerance: Implement redundancy and failover mechanisms to mitigate the impact of server failures or system errors.

c. Usability:

- User-friendly interface: Design an intuitive and easy-to-navigate user interface with clear navigation, consistent layout, and accessible features.

- Mobile responsiveness: Ensure the website is optimized for mobile devices and provides a seamless browsing and shopping experience across different screen sizes.

d. Security:

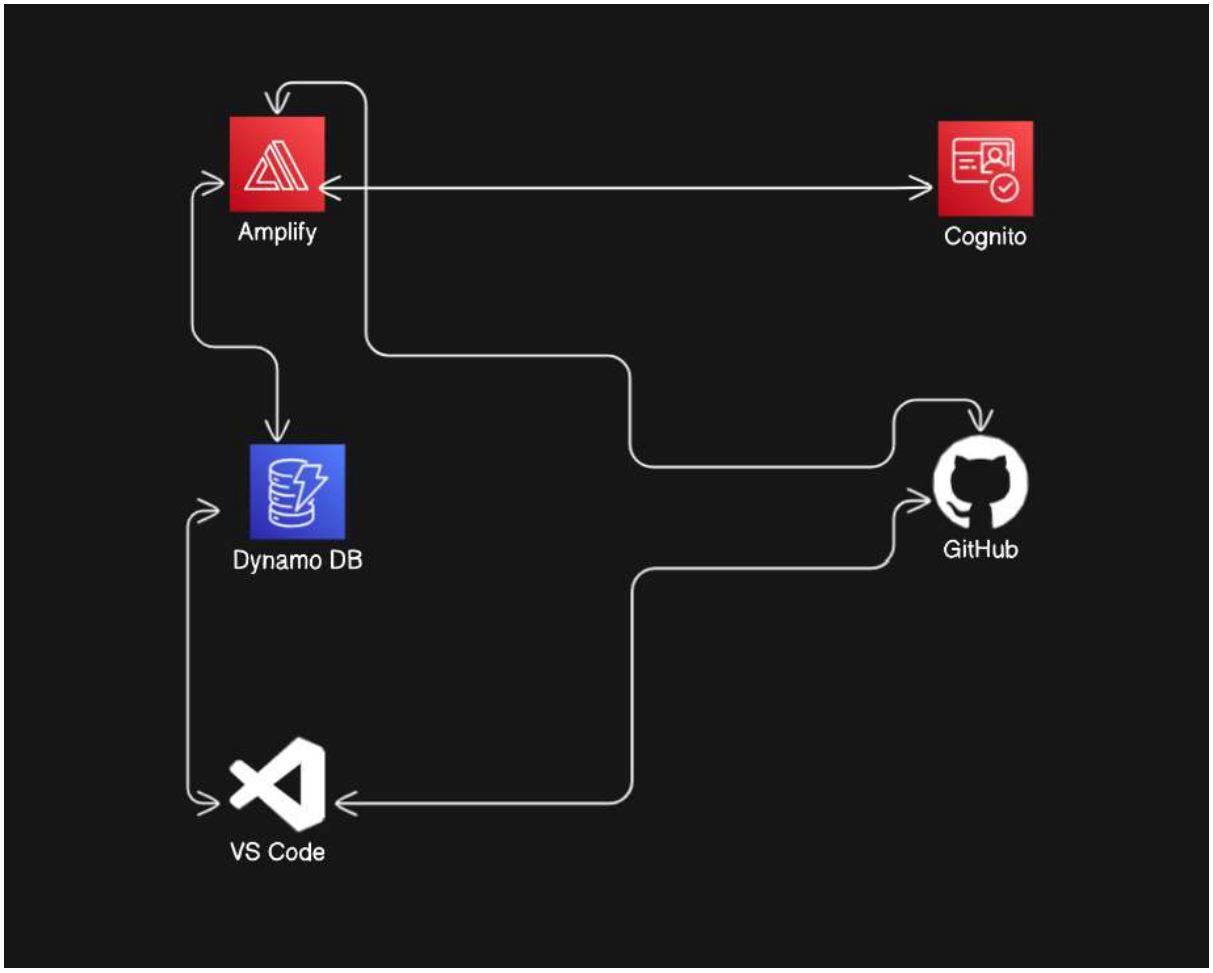
- Data protection: Implement measures to protect user data, including encryption, access controls, and secure transmission protocols.

e. Scalability:

- Elasticity: Design the architecture to scale resources dynamically based on demand to handle fluctuations in traffic and workload effectively.

- Resource optimization: Optimize resource utilization and minimize costs by scaling resources up or down as needed without overprovisioning.

Architecture/Flow Chart:



In the development workflow depicted in the flow chart, frontend developers use Visual Studio Code (VS Code) editor to write code for the application. The codebase is stored and managed in a GitHub repository, facilitating version control and collaboration. Meanwhile, AWS Amplify serves as a pivotal tool in simplifying the development and deployment process. Developers leverage Amplify to configure backend resources such as AWS Cognito for authentication and AWS DynamoDB for database storage. Amplify's intuitive interface enables seamless integration of backend services with the frontend application. Through Amplify, developers can define APIs, manage authentication workflows, and orchestrate data storage operations. This tight integration between Amplify, Cognito, and DynamoDB streamlines development efforts and ensures efficient communication between frontend and backend components. Overall, this workflow enables developers to rapidly iterate on features, maintain code quality, and deliver a robust and scalable application to end-users.

CODE:

App.js

```
import React from "react";
import "./App.css";

// Imports the Amplify library from 'aws-amplify' package. This is used to configure your app to interact with AWS services.
import { Amplify } from "aws-amplify";

// Imports the Authenticator and withAuthenticator components from '@aws-amplify/ui-react'.
// Authenticator is a React component that provides a ready-to-use sign-in and sign-up UI.
// withAuthenticator is a higher-order component that wraps your app component to enforce authentication.
import { Authenticator, withAuthenticator } from "@aws-amplify/ui-react";

// Imports the default styles for the Amplify UI components. This line ensures that the authenticator looks nice out of the box.
import "@aws-amplify/ui-react/styles.css";

// Imports the awsExports configuration file generated by the Amplify CLI. This file contains the AWS service configurations (like Cognito, AppSync, etc.) specific to your project.
import awsExports from "./aws-exports";
import Routers from "./Routers";

// Configures the Amplify library with the settings from aws-exports.js, which includes all the AWS service configurations for this project.
Amplify.configure(awsExports);

function App() {
  return (
    <div className="App">
      <Authenticator>
        {({ signOut, user }) => (
          <main>
            <Routers signOut={signOut} user={user} />
          </main>
        )}
      </Authenticator>
    </div>
  );
}
```

```
export default withAuthenticator(App);
```

Routers.jsx

```
import React from "react";
import { HashRouter as Router, Routes, Route } from "react-router-dom";
import { Navbar } from "./components/navbar";
import { Shop } from "./pages/shop/shop";
import { Contact } from "./pages/contact";
import { Cart } from "./pages/cart/cart";
import { ShopContextProvider } from "./context/shop-context";
import Footer from "./components/footer";
import AddressDetailsPage from "./pages/checkout/address";
import PaymentMethod from "./pages/checkout/PaymentMethod";
import PaymentPage from "./pages/checkout/PaymentPage";
import OTPpage from "./pages/checkout/OTPpage";
import DeliveryPage from "./pages/checkout/deliveryPage";

const Routers = ({ signOut, user }) => {
  return (
    <div className="">
      <ShopContextProvider>
        <Router>
          <Navbar signOut={signOut} user={user} />
          <Routes>
            <Route path="/" element={<Shop />} />
            <Route path="/contact" element={<Contact />} />
            <Route path="/cart" element={<Cart />} />
            <Route path="/checkout" element={<AddressDetailsPage />} />
            <Route path="/payment" element={<PaymentMethod />} />
            <Route path="/paymentpage" element={<PaymentPage />} />
            <Route path="/OTPpage" element={<OTPpage />} />
            <Route path="/deliveryPage" element={<DeliveryPage />} />
          </Routes>
          <Footer />
        </Router>
      </ShopContextProvider>
    </div>
  );
};

export default Routers;
```

Navbar.jsx

```
import React from "react";
import { Link } from "react-router-dom";
import { ShoppingCart } from "phosphor-react";
import "./navbar.css";

export const Navbar = ({ signOut, user }) => {
  return (
    <div className="navbar justify-between pl-5 sticky top-0 z-10">
      <div className="text-white text-3xl font-bold">ShopKart</div>
      <div className="links">
        <Link to="/"> Shop </Link>
        <Link to="/contact"> Contact </Link>
        <Link to="/cart">
          <ShoppingCart size={32} />
        </Link>
        <button
          // onClick={signOut}
          className="border border-white text-white ml-5 px-3 py-2 rounded-full"
        >
          Account
        </button>
        <button
          onClick={signOut}
          className="border border-white text-white ml-5 px-3 py-2 rounded-full"
        >
          Sign Out
        </button>
      </div>
    </div>
  );
};
```

Footer.jsx

```jsx
import React from "react";

function Footer() {
  return (
    <footer className="bg-gray-800 text-white py-8">
      <div className="container mx-auto px-4">
        <div className="flex flex-wrap justify-center">
          <div className="w-full md:w-1/2 lg:w-1/4 mb-4 md:mb-0">
            <h2 className="text-lg font-semibold mb-4">About Shopkart</h2>
            <p>
              A leading online store for all your shopping needs. Find the best
              deals on electronics, fashion, home goods, and more!
            </p>
          </div>
          <div className="w-full md:w-1/2 lg:w-1/4 mb-4 md:mb-0">
            <h2 className="text-lg font-semibold mb-4">Quick Links</h2>
            <ul>
              <li>
                <a href="#">Home</a>
              </li>
              <li>
                <a href="#">Shop</a>
              </li>
              <li>
                <a href="#">About Us</a>
              </li>
              <li>
                <a href="#">Contact Us</a>
              </li>
            </ul>
          </div>
          <div className="w-full md:w-1/2 lg:w-1/4 mb-4 md:mb-0">
            <h2 className="text-lg font-semibold mb-4">Follow Us</h2>
            <ul>
              <li>
                <a href="#">Facebook</a>
              </li>
              <li>
                <a href="#">Twitter</a>
              </li>
              <li>
                <a href="#">Instagram</a>
              </li>
              <li>
                <a href="#">LinkedIn</a>
              </li>
```

```
          </ul>
        </div>
      </div>
      <hr className="border-gray-700 my-6" />
      <p className="text-center">
        &copy; 2024 Shopkart. All rights reserved.
      </p>
    </div>
  </footer>
 );
}

export default Footer;
```

Context: shop-context.jsx

```
import { createContext, useEffect, useState } from "react";
import { PRODUCTS } from "../products";

export const ShopContext = createContext(null);

const getDefaultCart = () => {
 let cart = {};
 for (let i = 1; i < PRODUCTS.length + 1; i++) {
  cart[i] = 0;
 }
 return cart;
};

export const ShopContextProvider = (props) => {
 const [cartItems, setCartItems] = useState(getDefaultCart());

 const getTotalCartAmount = () => {
  let totalAmount = 0;
  for (const item in cartItems) {
   if (cartItems[item] > 0) {
    let itemInfo = PRODUCTS.find((product) => product.id === Number(item));
    totalAmount += cartItems[item] * itemInfo.price;
   }
  }
  return totalAmount;
 };
```

```
  const addToCart = (itemId) => {
    setCartItems((prev) => ({ ...prev, [itemId]: prev[itemId] + 1 }));
  };

  const removeFromCart = (itemId) => {
    setCartItems((prev) => ({ ...prev, [itemId]: prev[itemId] - 1 }));
  };

  const updateCartItemCount = (newAmount, itemId) => {
    setCartItems((prev) => ({ ...prev, [itemId]: newAmount }));
  };

  const checkout = () => {
    // setCartItems(getDefaultCart());
  };

  const contextValue = {
    cartItems,
    addToCart,
    updateCartItemCount,
    removeFromCart,
    getTotalCartAmount,
    checkout,
  };

  return (
    <ShopContext.Provider value={contextValue}>
      {props.children}
    </ShopContext.Provider>
  );
};
```

Cart.jsx

```
import React, { useContext } from "react";
import { ShopContext } from "../../context/shop-context";
import { PRODUCTS } from "../../products";
import { CartItem } from "./cart-item";
import { useNavigate } from "react-router-dom";

import "./cart.css";
```

```jsx
export const Cart = () => {
  const { cartItems, getTotalCartAmount, checkout } = useContext(ShopContext);
  const totalAmount = getTotalCartAmount();

  const navigate = useNavigate();

  return (
    <div className="cart min-h-[85vh]">
      <div>
        <h1 className="text-4xl font-bold py-5">Your Cart Items</h1>
      </div>
      <div className="cart">
        {PRODUCTS.map((product) => {
          if (cartItems[product.id] !== 0) {
            return <CartItem data={product} />;
          }
        })}
      </div>

      {totalAmount > 0 ? (
        <div className="checkout text-2xl font-semibold">
          <p> Subtotal: ${totalAmount} </p>
          <button className="p-3" onClick={() => navigate("/")}>
            {" "}
            Continue Shopping{" "}
          </button>
          <button
            className="p-3"
            onClick={() => {
              checkout();
              navigate("/checkout");
            }}
          >
            {" "}
            Checkout{" "}
          </button>
        </div>
      ) : (
        <h1 className="text-4xl font-bold py-5">
          {" "}
          Your Shopping Cart is Empty
        </h1>
      )}
    </div>
  );
};
```

Cart-item.jsx

```
import React, { useContext } from "react";
import { ShopContext } from "../../context/shop-context";

export const CartItem = (props) => {
  const { id, productName, price, productImage } = props.data;
  const { cartItems, addToCart, removeFromCart, updateCartItemCount } =
    useContext(ShopContext);

  return (
    <div className="cartItem">
      <img src={productImage} />
      <div className="description">
        <p>
          <b>{productName}</b>
        </p>
        <p> Price: ${price}</p>
        <div className="countHandler">
          <button onClick={() => removeFromCart(id)}> - </button>
          <input
            value={cartItems[id]}
            onChange={(e) => updateCartItemCount(Number(e.target.value), id)}
          />
          <button onClick={() => addToCart(id)}> + </button>
        </div>
      </div>
    </div>
  );
};
```

Checkout pages:

1. Address.jsx

```jsx
import React, { useState } from "react";
import { useNavigate } from "react-router-dom";

function AddressDetailsPage() {
  const [formData, setFormData] = useState({
    fullName: "",
    email: "",
    addressLine1: "",
    addressLine2: "",
    city: "",
    state: "",
    zipCode: "",
    country: "",
    phone: "",
  });

  const handleChange = (e) => {
    const { name, value } = e.target;
    setFormData((prevState) => ({
      ...prevState,
      [name]: value,
    }));
  };

  const navigate = useNavigate();

  const handleSubmit = (e) => {
    e.preventDefault();
    // Add logic to handle form submission
    navigate("/payment");
  };

  return (
    <div className="bg-gray-100 min-h-screen py-8">
      <div className="container mx-auto px-4">
        <h1 className="text-3xl font-semibold mb-8">
          Fill Address and Other Details
        </h1>
        <form onSubmit={handleSubmit}>
          <div className="grid grid-cols-1 sm:grid-cols-2 gap-6">
            <div>
              <label
                htmlFor="fullName"
                className="block text-sm font-semibold mb-1"
              >
                Full Name
```

```
            </label>
            <input
              type="text"
              id="fullName"
              name="fullName"
              value={formData.fullName}
              onChange={handleChange}
                className="w-full px-4 py-2 border border-gray-300 rounded-md focus:outline-none
focus:border-red-500"
              // // required
            />
          </div>
          <div>
            <label
              htmlFor="email"
              className="block text-sm font-semibold mb-1"
            >
              Email
            </label>
            <input
              type="email"
              id="email"
              name="email"
              value={formData.email}
              onChange={handleChange}
                className="w-full px-4 py-2 border border-gray-300 rounded-md focus:outline-none
focus:border-red-500"
              // // required
            />
          </div>
          <div>
            <label
              htmlFor="addressLine1"
              className="block text-sm font-semibold mb-1"
            >
              Address Line 1
            </label>
            <input
              type="text"
              id="addressLine1"
              name="addressLine1"
              value={formData.addressLine1}
              onChange={handleChange}
                className="w-full px-4 py-2 border border-gray-300 rounded-md focus:outline-none
focus:border-red-500"
              // required
```

```
          />
        </div>
        <div>
          <label
            htmlFor="addressLine2"
            className="block text-sm font-semibold mb-1"
          >
            Address Line 2
          </label>
          <input
            type="text"
            id="addressLine2"
            name="addressLine2"
            value={formData.addressLine2}
            onChange={handleChange}
              className="w-full px-4 py-2 border border-gray-300 rounded-md focus:outline-none
focus:border-red-500"
          />
        </div>
        <div>
          <label
            htmlFor="city"
            className="block text-sm font-semibold mb-1"
          >
            City
          </label>
          <input
            type="text"
            id="city"
            name="city"
            value={formData.city}
            onChange={handleChange}
              className="w-full px-4 py-2 border border-gray-300 rounded-md focus:outline-none
focus:border-red-500"
            // required
          />
        </div>
        <div>
          <label
            htmlFor="state"
            className="block text-sm font-semibold mb-1"
          >
            State
          </label>
          <input
            type="text"
```

```
            id="state"
            name="state"
            value={formData.state}
            onChange={handleChange}
              className="w-full px-4 py-2 border border-gray-300 rounded-md focus:outline-none
focus:border-red-500"
            // required
           />
          </div>
          <div>
           <label
            htmlFor="zipCode"
            className="block text-sm font-semibold mb-1"
           >
            Zip Code
           </label>
           <input
            type="text"
            id="zipCode"
            name="zipCode"
            value={formData.zipCode}
            onChange={handleChange}
              className="w-full px-4 py-2 border border-gray-300 rounded-md focus:outline-none
focus:border-red-500"
            // required
           />
          </div>
          <div>
           <label
            htmlFor="country"
            className="block text-sm font-semibold mb-1"
           >
            Country
           </label>
           <input
            type="text"
            id="country"
            name="country"
            value={formData.country}
            onChange={handleChange}
              className="w-full px-4 py-2 border border-gray-300 rounded-md focus:outline-none
focus:border-red-500"
            // required
           />
          </div>
          <div>
```

```
            <label
              htmlFor="phone"
              className="block text-sm font-semibold mb-1"
            >
              Phone
            </label>
            <input
              type="text"
              id="phone"
              name="phone"
              value={formData.phone}
              onChange={handleChange}
               className="w-full px-4 py-2 border border-gray-300 rounded-md focus:outline-none
focus:border-red-500"
              // required
            />
          </div>
        </div>
        <div className="text-center mt-6">
          <button
            type="submit"
                 className="btn bg-red-500 hover:bg-red-600 text-white font-semibold px-6 py-2
rounded-md transition duration-300"
          >
            Submit
          </button>
        </div>
      </form>
    </div>
  </div>
 );
}

export default AddressDetailsPage;
```

2. PaymentPage.jsx

```
import React, { useContext, useEffect } from "react";
import { useState } from "react";
import { NavLink } from "react-router-dom";
```

```
import { ShopContext } from "../../context/shop-context";

export default function PaymentPage() {
  const { cartItems, getTotalCartAmount, checkout } = useContext(ShopContext);
  const cartPrice = getTotalCartAmount();
  const [visible, setvisible] = useState(false);

  useEffect(() => {
    setTimeout(() => {
      setvisible(true);
    }, 1000);
  }, []);

  return (
    <div className="bg-gray-200">
      {/* // Redirecting  */}
      {!visible && (
        <div className="text-left text-lg min-h-[80vh] p-4">
          <div className="text-3xl my-4 font-semibold">
            Securely redirecting you to ShopKart Pay...
          </div>
          <div className="mt-10">Please do not close or leave this page.</div>
          <div className="my-5">
            You will return to ShopKart.in when finished.
          </div>
            <button className="bg-red-400 text-md py-2 max-w-[499px] w-full rounded-lg my-2
hover:bg-red-500">
            Continue
          </button>
        </div>
      )}
      {/* // RazorPay  */}
      {visible && (
        <div className="py-12">
          <div className="p-12 border text-left bg-gray-100 w-fit xs:w-screen mx-auto border-gray-
300 rounded-xl shadow-md">
            <div className="mb-5">
              <img
                src="https://upload.wikimedia.org/wikipedia/en/thumb/8/89/Razorpay_logo.svg/800px-
Razorpay_logo.svg.png?20171127075036"
                alt=""
                className="h-10 rounded-lg"
              />
            </div>
            <div className="flex">
```

```
          <div className="w-[200px] mb-4 rounded-lg border-4 border-blue-500 text-blue-500 p-4 bg-white">
            <svg
              xmlns="http://www.w3.org/2000/svg"
              fill="none"
              viewBox="0 0 24 24"
              strokeWidth={1.5}
              stroke="currentColor"
              className="w-6 h-6"
            >
              <path
                strokeLinecap="round"
                strokeLinejoin="round"
                 d="M2.25 8.25h19.5M2.25 9h19.5m-16.5 5.25h6m-6 2.25h3m-3.75 3h15a2.25 2.25 0 002.25-2.25V6.75A2.25 2.25 0 0019.5 4.5h-15a2.25 2.25 0 00-2.25 2.25v10.5A2.25 2.25 0 004.5 19.5z"
              />
            </svg>
            <div className="font-bold text-xl">Card</div>
          </div>
          <div></div>
        </div>
        <div className="text-xl mb-1">Card number</div>
        <input
          type="number"
          placeholder="1234 1234 1234 1234"
          //   onChange={(event) => {}}
          className="p-4 text-xl w-full shadow-md rounded-lg border border-gray-300 bg-white"
        />
        <div className="flex xs:flex-col mt-3 space-x-4 xs:space-x-0">
          <div>
            <div className="text-xl mb-1">Expiry</div>
            <input
              type="text"
              placeholder="MM / YY"
              className="p-4 text-xl shadow-md rounded-lg border border-gray-300 bg-white"
            />
          </div>
          <div>
            <div className="text-xl mb-1">CVV</div>
            <input
              type="number"
              placeholder="CVV"
              className="p-4 text-xl shadow-md rounded-lg border border-gray-300 bg-white"
            />
          </div>
```

```
          </div>
          <div className="flex my-3 xs:flex-col space-x-4 xs:space-x-0">
            <div>
              <div className="text-xl mb-1">Country</div>
              <input
                type="text"
                placeholder="India"
                className="p-4 text-xl shadow-md rounded-lg border border-gray-300 bg-white"
              />
            </div>
            <div>
              <div className="text-xl mb-1">Mobile number</div>
              <input
                type="number"
                placeholder="Eg. 9714008888"
                className="p-4 text-xl shadow-md rounded-lg border border-gray-300 bg-white"
              />
            </div>
          </div>
          <div className="text-gray-600 my-1 text-lg max-w-[650px]">
            By providing your card information, you allow Amazon to charge
            your card for future payments in accordance with their terms.
          </div>
          <NavLink to="/OTPpage">
            <button className="bg-blue-500 hover:bg-blue-600 font-bold text-xl text-white w-full
p-4 rounded-lg my-4">
              Pay Now ₹{cartPrice}.00
            </button>
          </NavLink>
        </div>
      </div>
    )}
  </div>
 );
}
```

3. PaymentMethod.jsx

```
import React, { useContext, useRef, useState } from "react";
```

```
import { useEffect } from "react";
import { NavLink } from "react-router-dom";
import { ShopContext } from "../../context/shop-context";

export default function PaymentMethod() {
  const { cartItems, getTotalCartAmount, checkout } = useContext(ShopContext);
  const cartPrice = getTotalCartAmount();
  const [coupons, setCoupons] = useState(["NEWYEAR60", "GET60"]);
  const [couponInputVal, setcouponInputVal] = useState();
  const discount = 0.4; // i.e 60 % discount
  const [discountedPrice, setdiscountedPrice] = useState();

  const couponInputRef = useRef();

  useEffect(() => {
    // setcartPrice(15000);
    setdiscountedPrice(cartPrice);
  }, []);

  useEffect(() => {
    coupons.includes(couponInputRef.current.value) &&
      cartPrice &&
      setdiscountedPrice(cartPrice * discount);
  }, [couponInputVal]);

  return (
    <div className="bg-gray-200 min-h-[60vh]">
      <div className="flex p-4">
        {/* left part  */}
        <div className="w-3/5 bg-white text-left p-3 text-black">
          <div className="text-2xl mb-3 font-semibold">
            Select a payment method
          </div>
          <hr />
          <div className="mt-4">
            <div className="my-2 font-semibold text-gray-700">AMAZON PAY</div>
            <div className="flex border rounded-md p-4">
              <input type="checkbox" className="mr-4 w-4" />
              <div className="font-semibold text-xl">
                Amazon Pay (Coming soon)
                <div className="font-normal text-gray-700">
                  Get started with amazon pay UPI
                </div>
              </div>
            </div>
          </div>
```

```
        <div className="">
          <div className="my-3 font-semibold text-gray-700">
            MORE WAYS TO PAY
          </div>
          <div className="flex border rounded-md p-4">
            <input type="checkbox" className="mr-4 w-4" />
            <div className="font-semibold text-xl">
              Pay with Debit/Credit/ATM Cards
              <div className="font-normal text-gray-700">
                You can save your cards as per new RBI guidelines
              </div>
            </div>
          </div>
          <div className="flex border rounded-md p-4">
            <input type="checkbox" className="mr-4 w-4" />
            <div className="font-semibold text-xl">
              EMI (Coming soon)
              <div className="font-normal text-gray-700"></div>
            </div>
          </div>
          <div className="border p-4 text-xl text-blue-600">
            Add Gift Card or Promo Code
            <div className="flex flex-wrap mt-2 space-x-4 text-black">
              <input
                ref={couponInputRef}
                className="border border-gray-600 py-3 px-6 rounded-md"
                placeholder="Enter Code"
                type="text"
              />
              <button
                onClick={() =>
                  couponInputRef.current.value != "" &&
                  setcouponInputVal(couponInputRef.current.value)
                }
                className="border py-3 px-6 rounded-md"
              >
                Apply
              </button>
            </div>
          </div>
        </div>
      </div>
      {/* right part  */}
      <div className=" w-5/12 px-3">
        <div className="bg-white text-left p-4 text-black">
          <NavLink to="/paymentpage">
```

```
            <button className="bg-red-500 text-white text-lg font-semibold py-2 w-full rounded-lg
my-2 hover:bg-red-600">
              Place Your Order And Pay
            </button>
          </NavLink>
          <div className="text-sm my-2">
            You can review this order before it's final.
          </div>
          <div className="font-semibold">Order Summary</div>
          <div className="flex">
            <div>Items:</div>
            <div className="ml-auto">₹{cartPrice}.00</div>
          </div>
          <div className="flex">
            <div>Delivery:</div>
            <div className="ml-auto">₹0.00</div>
          </div>
          <div className="flex">
            <div>Total:</div>
            <div className="ml-auto">₹{cartPrice}.00</div>
          </div>
          <div className="flex">
            <div>Promotion(s) Applied:</div>
            {/* <div className="ml-auto">- ₹{cartPrice}.00</div> */}
            {/* Coupon discount NEWYEAR60 below */}
            <div className="ml-auto">
              {/* - ₹{cartPrice - cartPrice * discount}.00 */}- ₹
              {cartPrice - discountedPrice}.00
            </div>
          </div>
          <div className="flex text-xl text-red-600 font-bold my-2">
            <div>Order Total:</div>
            <div className="ml-auto">₹{discountedPrice}.00</div>
          </div>
          <div className="font-semibold">
            Order Totals include GST.
            <a className="text-sm cursor-pointer ml-1 text-blue-500 hover:underline">
              See details
            </a>
          </div>
        </div>
      </div>
    </div>
  </div>
 );
}
```

4. OTPpage.jsx

```
import React from "react";
import { NavLink } from "react-router-dom";

export default function OTPpage() {

  return (
    <div>
      <div className=" bg-gray-200">
        <div className="py-12">
          <div className="p-12 border text-left bg-gray-100 w-fit mx-auto border-gray-300 rounded-xl shadow-md">
            <div className="mb-5">
              <img
                src="https://upload.wikimedia.org/wikipedia/en/thumb/8/89/Razorpay_logo.svg/800px-Razorpay_logo.svg.png?20171127075036"
                alt=""
                className="h-10 rounded-lg"
              />
            </div>
            <div className="flex">
              <div className="w-[200px] mb-4 rounded-lg border-4 border-blue-500 text-blue-500 p-4 bg-white">
                <svg
                  xmlns="http://www.w3.org/2000/svg"
                  fill="none"
                  viewBox="0 0 24 24"
                  strokeWidth={1.5}
                  stroke="currentColor"
                  className="w-6 h-6"
                >
                  <path
                    strokeLinecap="round"
                    strokeLinejoin="round"
                    d="M2.25 8.25h19.5M2.25 9h19.5m-16.5 5.25h6m-6 2.25h3m-3.75 3h15a2.25 2.25 0 002.25-2.25V6.75A2.25 2.25 0 0019.5 4.5h-15a2.25 2.25 0 00-2.25 2.25v10.5A2.25 2.25 0 004.5 19.5z"
                  />
```

```
            </svg>
            <div className="font-bold text-xl">Card</div>
          </div>
          <div></div>
        </div>
        <div className="text-2xl font-semibold mb-4">
          Enter the OTP sent to +91 XXXXXXXXXX
        </div>
        <input
          type="number"
          placeholder="OTP"
          className="p-4 text-xl w-full shadow-md rounded-lg border border-gray-300 bg-white"
        />
        <div className="text-blue-500 text-right mt-2 mb-20 font-bold cursor-pointer ml-auto">
          Resend OTP
        </div>
        <div className="text-gray-600 text-sm my-1 max-w-[650px]">
          By clicking "Submit OTP", I agree to
          <a className="ml-1 font-semibold text-blue-500 cursor-pointer hover:underline">
            Terms and Conditions and Service Agreement
          </a>
        </div>
        <NavLink to="/deliveryPage">
          <button className="bg-blue-500 hover:bg-blue-600 font-bold text-xl text-white w-full
p-4 rounded-lg my-4">
            Submit OTP
          </button>
        </NavLink>
      </div>
    </div>
  </div>
 );
}
```

5. deliveryPage.jsx

```
import React from "react";
import { NavLink } from "react-router-dom";
```

```
const DeliveryPage = () => {
  return (
    <div className="bg-gray-100 min-h-screen py-8">
      <div className="container mx-auto px-4">
        <div className="max-w-lg mx-auto flex flex-col bg-white p-6 rounded-lg shadow-md text-
center">
          <img
            src="https://imgs.search.brave.com/HqUu9kQy_dqrc8_KzMih89eySS1wuYW1_IV8dOn
bW_c/rs:fit:500:0:0/g:ce/aHR0cHM6Ly90NC5m/dGNkbi5uZXQvanBn/LzA1LzQ5LzM4Lzcz/L
zM2MF9GXzU0OTM4/NzM4N19SQ2txVnRm/MnQ1UGJOWUdRZ1FF/SFpNcEZ1SGJwVHZ
K/ei5qcGc"
            alt="Success"
            className="h-[45vh] object-contain"
          />
          <h1 className="text-2xl font-bold mb-4">Payment Successful!</h1>
          <p className="text-gray-700 mb-4">
            Thank you for shopping with Shopkart. Your order has been
            successfully placed.
          </p>
          <NavLink
            to="/"
           className="btn bg-red-500 hover:bg-red-600 text-white font-semibold px-6 py-2 rounded-
md transition duration-300"
          >
            Continue Shopping
          </NavLink>
        </div>
      </div>
    </div>
  );
};

export default DeliveryPage;
```

Contact.jsx

```
import React from "react";

export const Contact = () => {
  return (
```

```
  <div className="contact-page bg-gray-100 min-h-screen py-12">
    <div className="container mx-auto px-4">
      <div className="contact-title text-center mb-8">
        <h1 className="text-4xl font-bold">Contact Us</h1>
        <p className="text-gray-600">We'd love to hear from you!</p>
      </div>

      <div className="contact-form max-w-lg mx-auto bg-white p-6 rounded-lg shadow-md">
        <form>
          <div className="mb-4">
            <label
              htmlFor="name"
              className="block text-gray-700 font-semibold mb-2"
            >
              Your Name
            </label>
            <input
              type="text"
              id="name"
              name="name"
                 className="w-full px-4 py-2 border border-gray-300 rounded-md focus:outline-none
focus:border-red-500"
            />
          </div>
          <div className="mb-4">
            <label
              htmlFor="email"
              className="block text-gray-700 font-semibold mb-2"
            >
              Your Email
            </label>
            <input
              type="email"
              id="email"
              name="email"
                 className="w-full px-4 py-2 border border-gray-300 rounded-md focus:outline-none
focus:border-red-500"
            />
          </div>
          <div className="mb-4">
            <label
              htmlFor="message"
              className="block text-gray-700 font-semibold mb-2"
            >
              Message
            </label>
```

```
            <textarea
              id="message"
              name="message"
              rows="4"
                classNameName="w-full px-4 py-2 border border-gray-300 rounded-md focus:outline-none
focus:border-red-500"
            ></textarea>
          </div>
          <div classNameName="text-center">
            <button
              type="submit"
                  classNameName="btn bg-red-500 hover:bg-red-600 text-white font-semibold px-6 py-2
rounded-md transition duration-300"
            >
              Send Message
            </button>
          </div>
        </form>
      </div>
    </div>
  </div>
 );
};
```

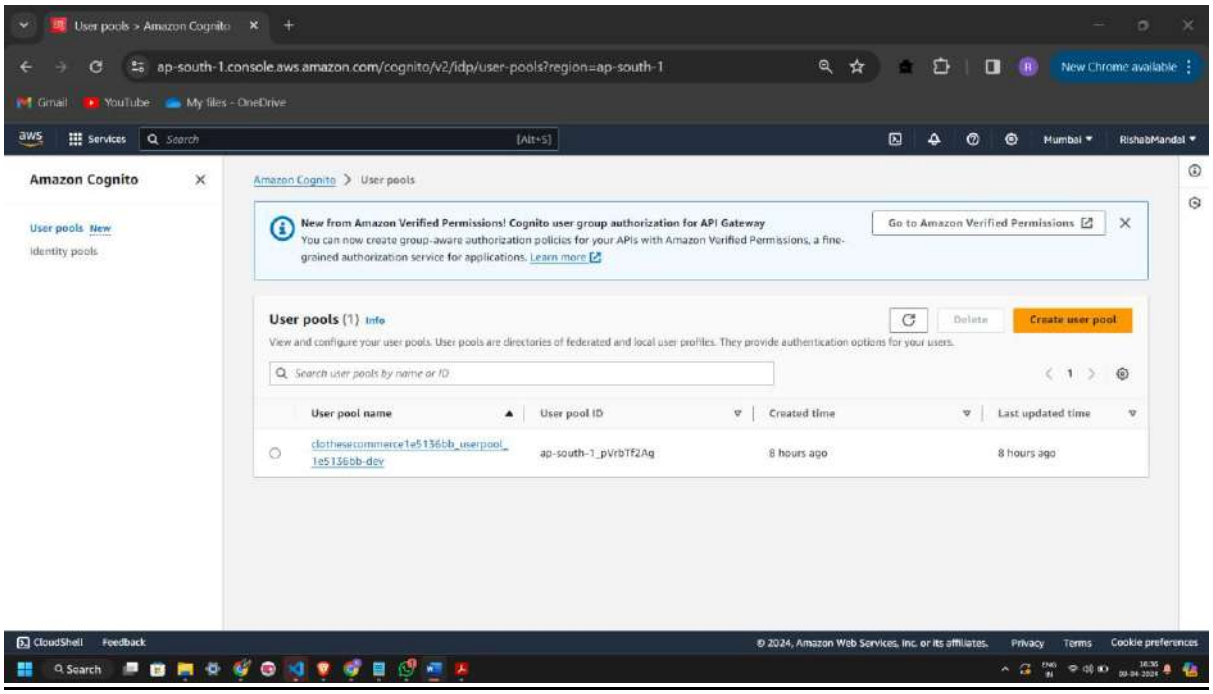
AWS DynamoDB Setup with product.js

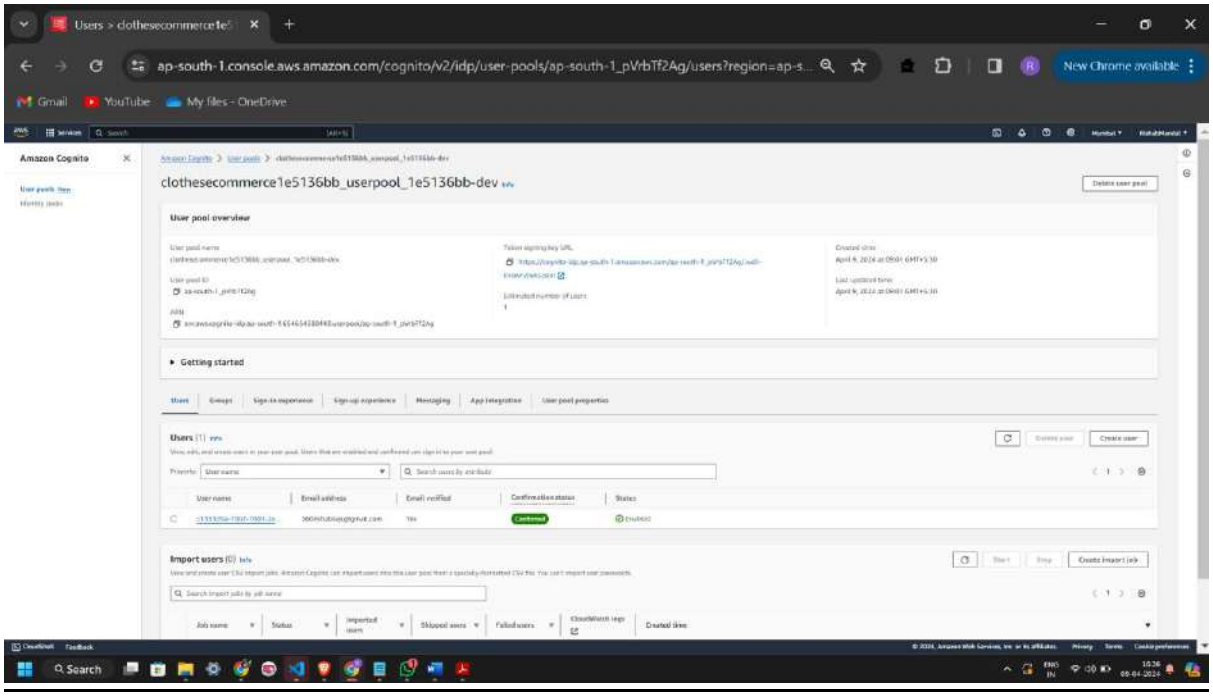
```
import AWS from "aws-sdk";

const dynamoDB = new AWS.DynamoDB.DocumentClient({
 region: "ap-south-1",
 accessKeyId: "AKIAZQ3DRDGN6ONMIEW2",
 secretAccessKey: "gsOuaBlfnqCE6d5ZMWJF/R3NnH0CDGD6qDtxcv7p",
});

const fetchProductsFromDynamoDB = async () => {
 try {
   const params = {
     TableName: "Products", // Replace 'YourTableName' with your DynamoDB table name
   };
   const data = await dynamoDB.scan(params).promise();
```

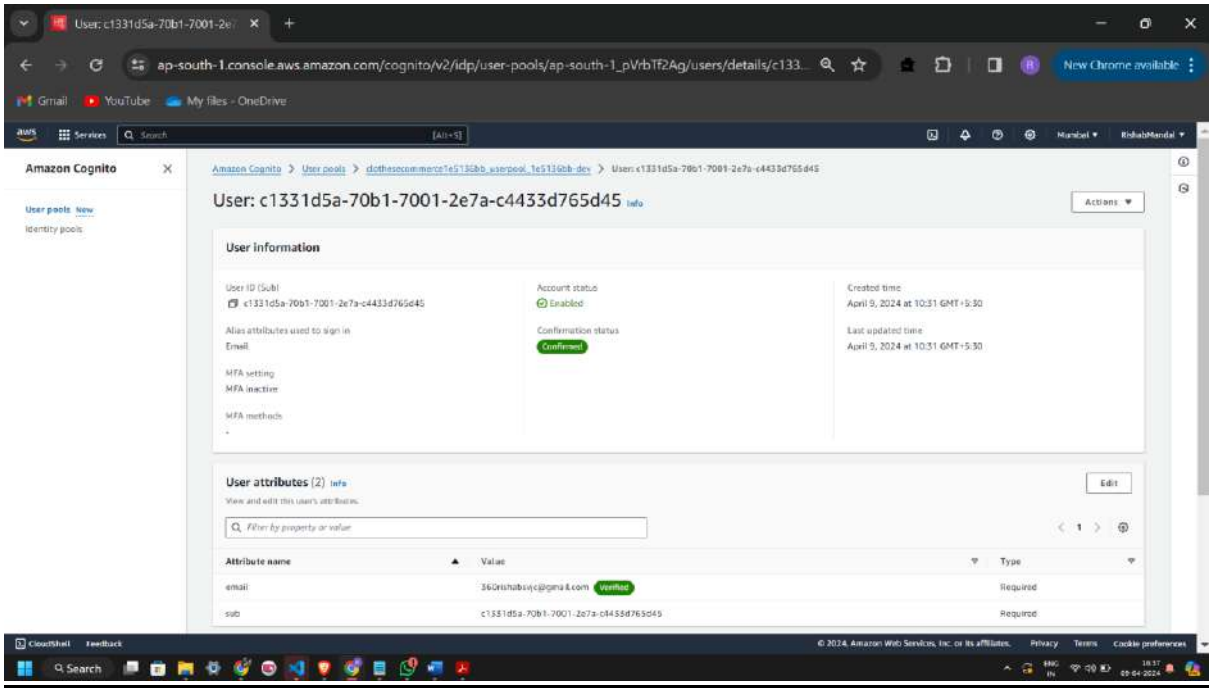
```
    // console.log(data);
    return data.Items.map((item) => ({
      id: parseInt(item.id),
      productName: item.productName,
      price: parseFloat(item.price),
      productImage: item.productImage,
    }));
  } catch (error) {
    console.error("Error fetching products from DynamoDB:", error);
    return [];
  }
};

export const PRODUCTS = await fetchProductsFromDynamoDB();
```

Screenshots of the project

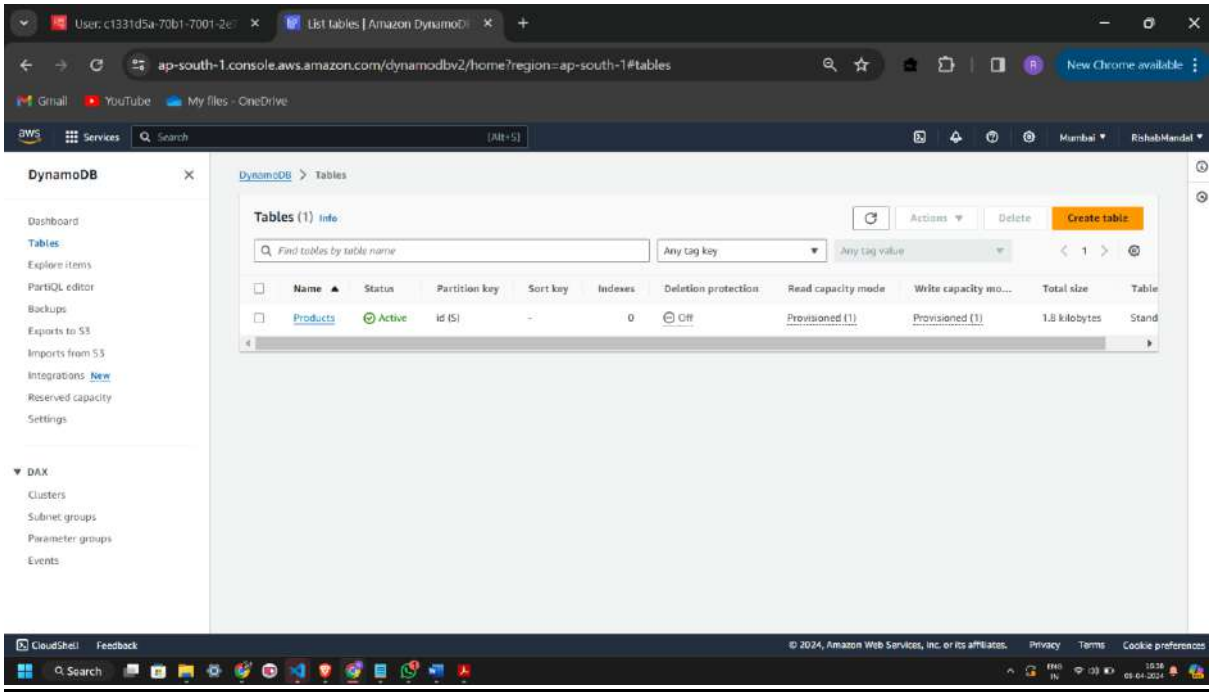
AWS-Cognito:

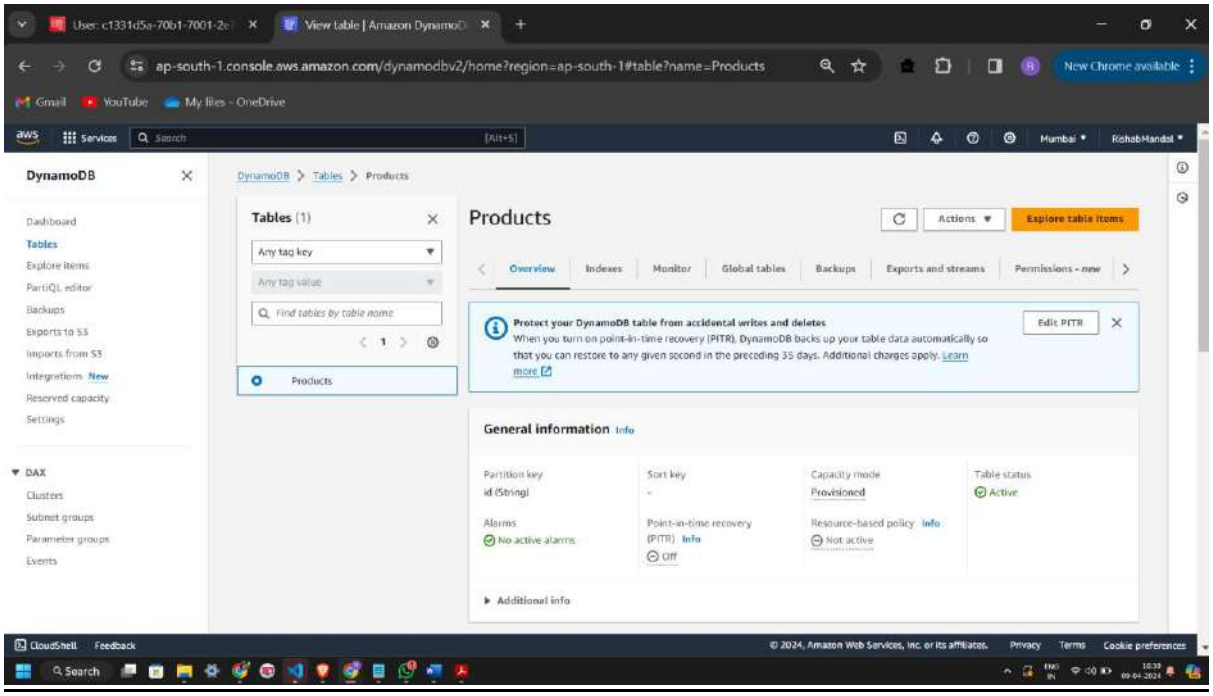


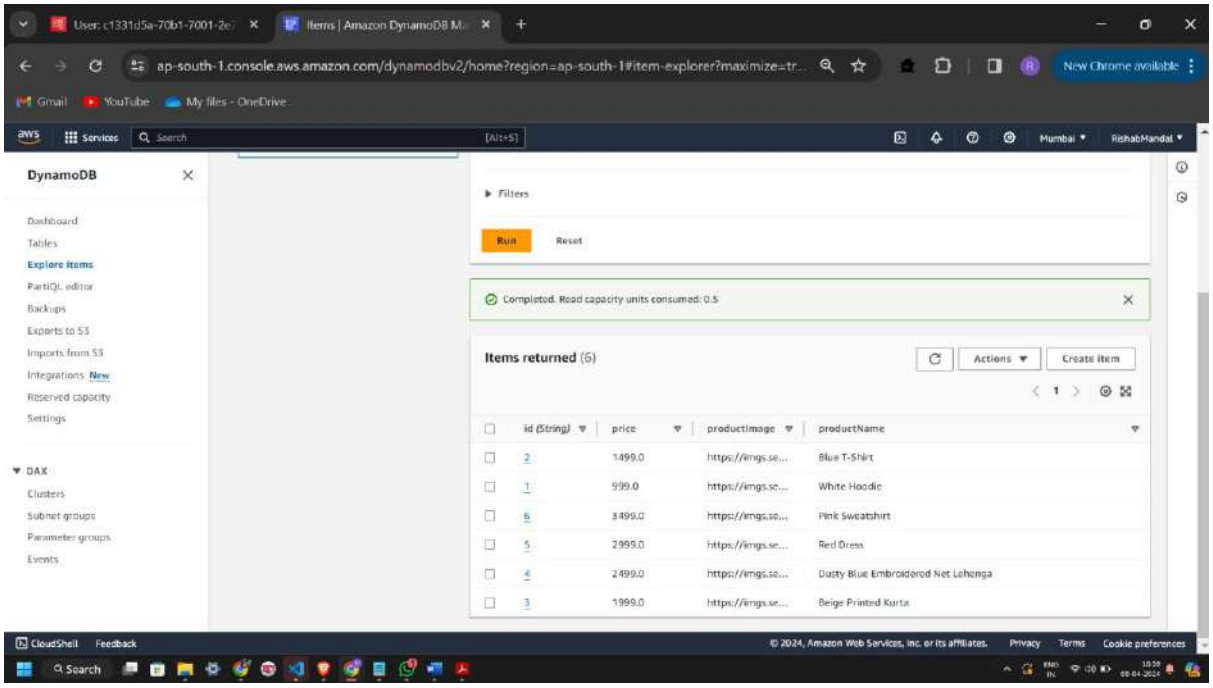

Users > clothesecommerce1e5
ap-south-1.console.aws.amazon.com/cognito/v2/idp/user-pools/ap-south-1_pVrbTf2Ag/users?region=ap-s...
New Chrome available
Gmail
YouTube
My files - OneDrive
Amazon Cognito
clothesecommerce1e5136bb_userpool_1e5136bb-dev
User pool overview
Getting started
Users
Import users
Search


User: c1331d5a-70b1-7001-2e7
ap-south-1.console.aws.amazon.com/cognito/v2/idp/user-pools/ap-south-1_pVrbTf2Ag/users/details/c133...
New Chrome available
Gmail
YouTube
My files - OneDrive
Amazon Cognito
User pools New
Identity pools
Amazon Cognito > User pools > clothesecommerce1e5136bb_userpool_1e5136bb-dev > User: c1331d5a-70b1-7001-2e7a-c4433d765d45
User: c1331d5a-70b1-7001-2e7a-c4433d765d45 Info
Actions
User information
User ID (Sub)
c1331d5a-70b1-7001-2e7a-c4433d765d45
Account status
Enabled
Created time
April 9, 2024 at 10:31 GMT+5:30
Alias attributes used to sign in
Email
Confirmation status
Confirmed
Last updated time
April 9, 2024 at 10:31 GMT+5:30
MFA setting
MFA inactive
MFA methods
User attributes (2) Info
Edit
Attribute name
Value
Type
email
Verified
Required
sub
c1331d5a-70b1-7001-2e7a-c4433d765d45
Required
Search

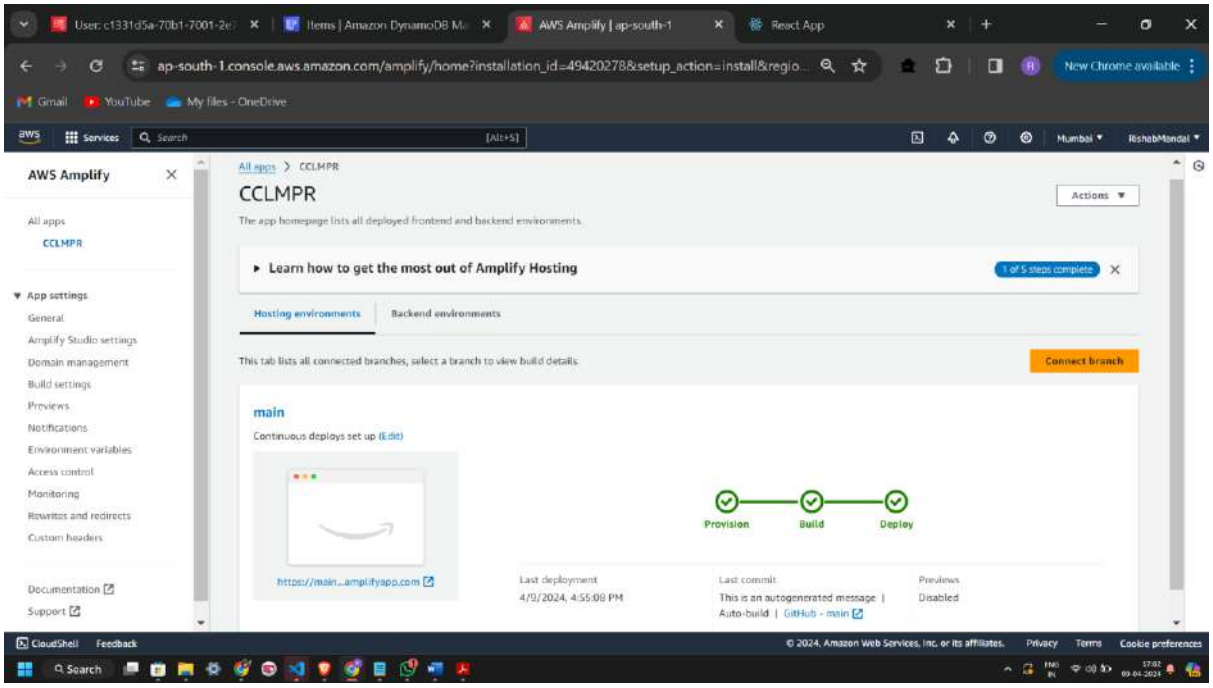
AWS-DynamoDB:



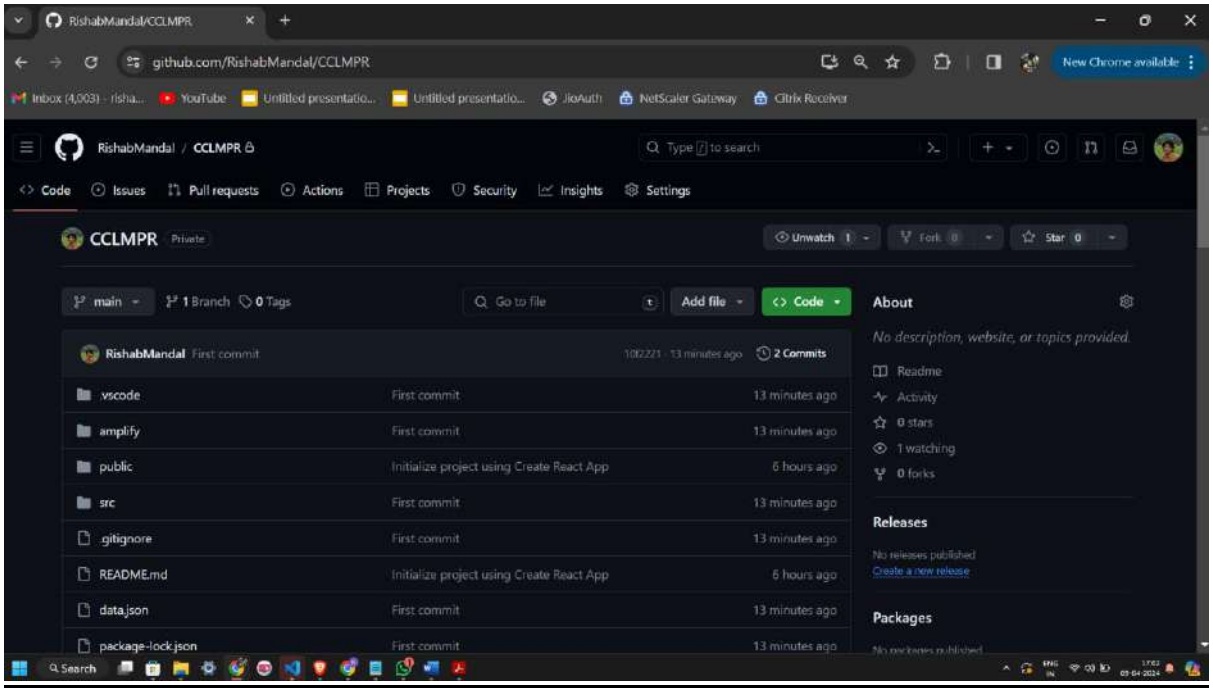




AWS-Amplify:

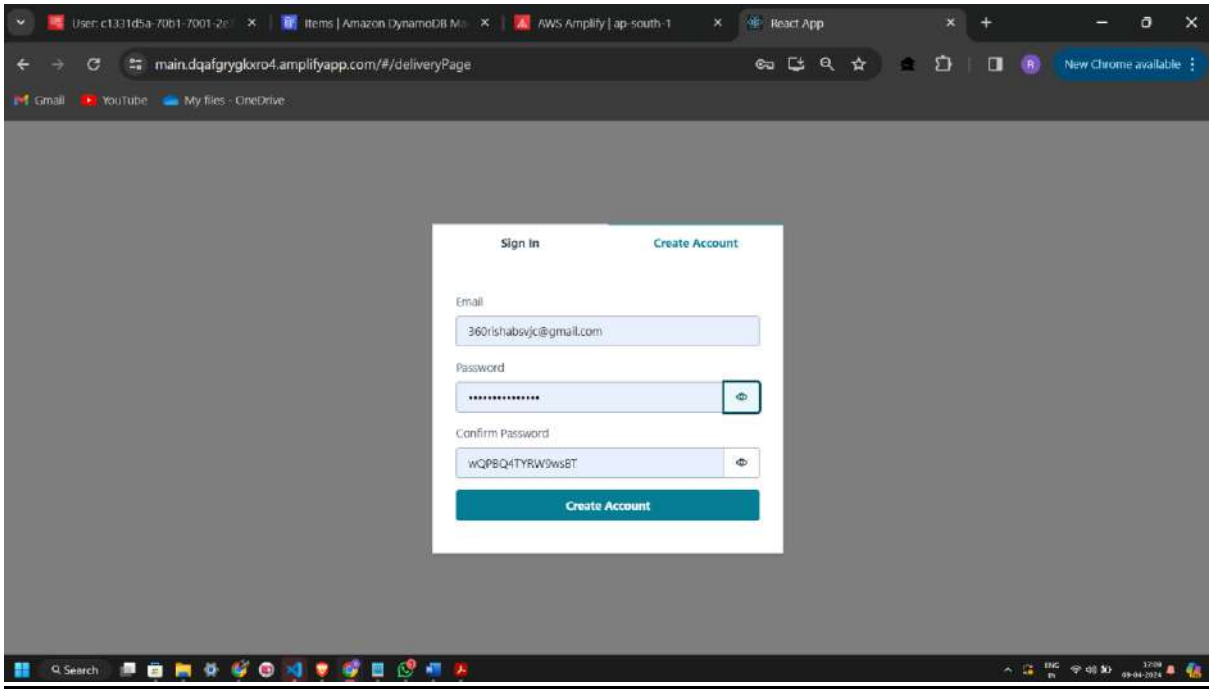


GitHub:

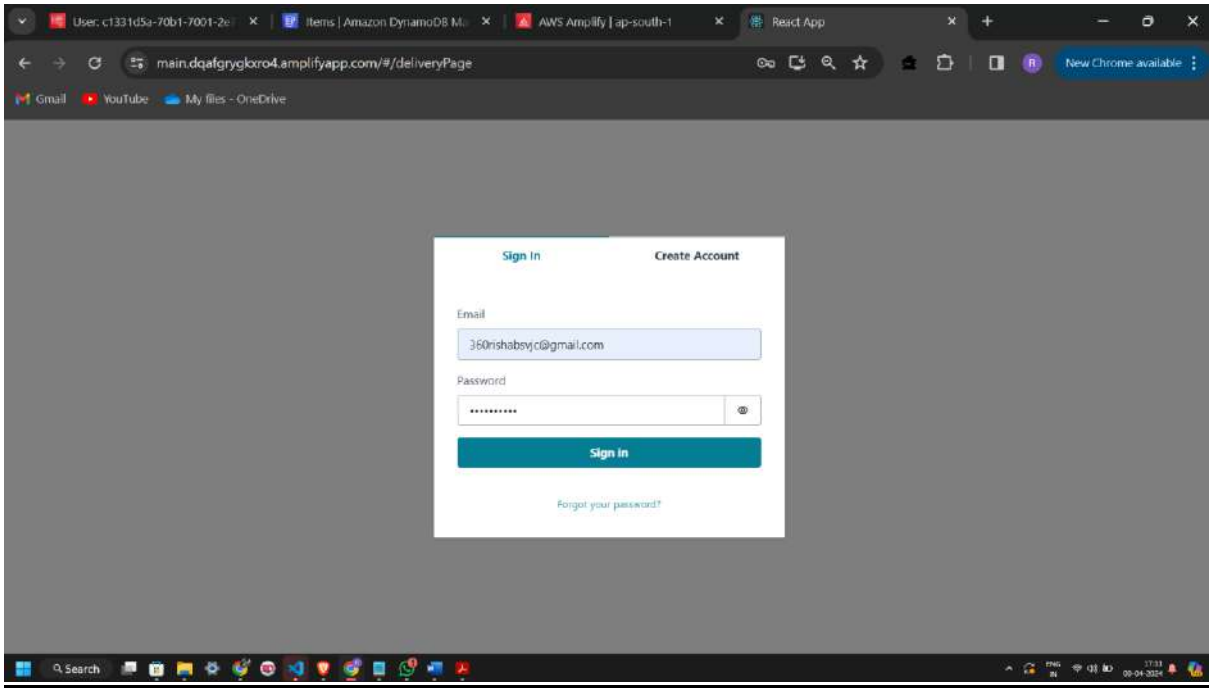


Website:

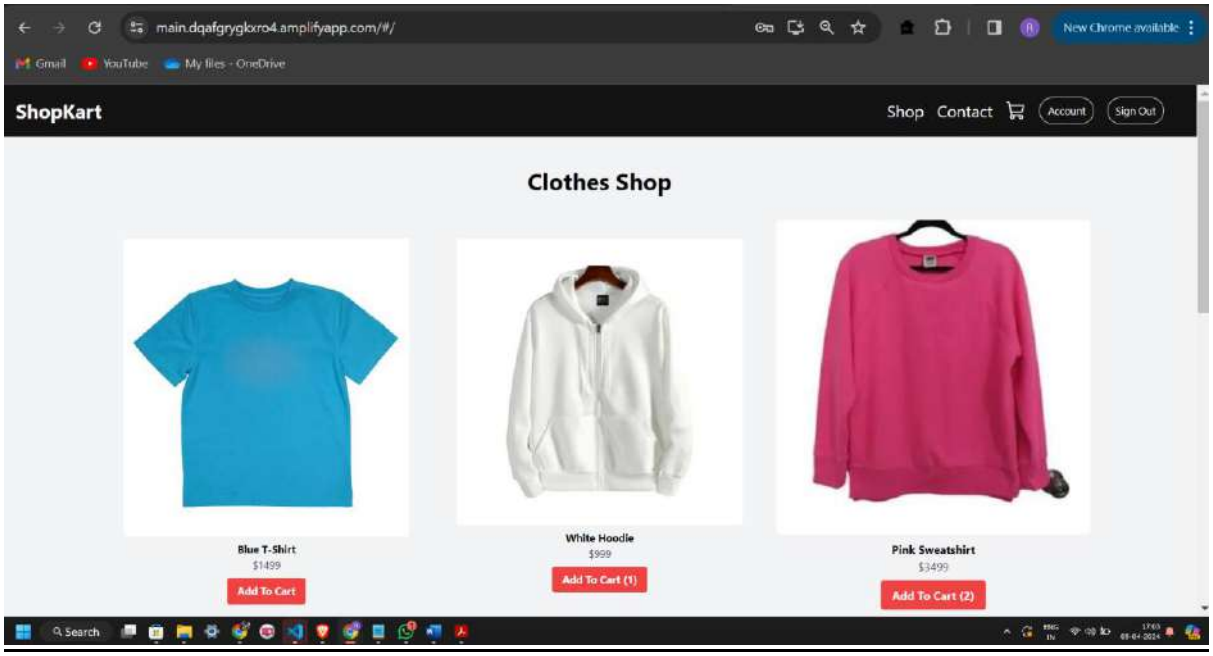
Login page

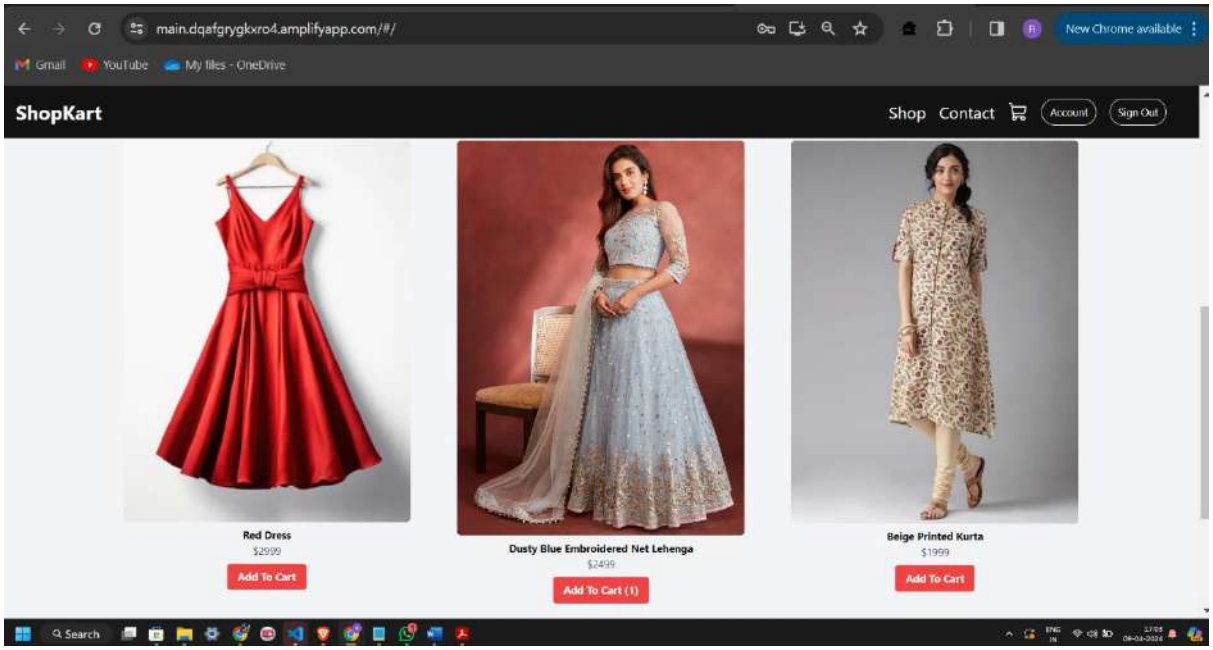


Sign-up page

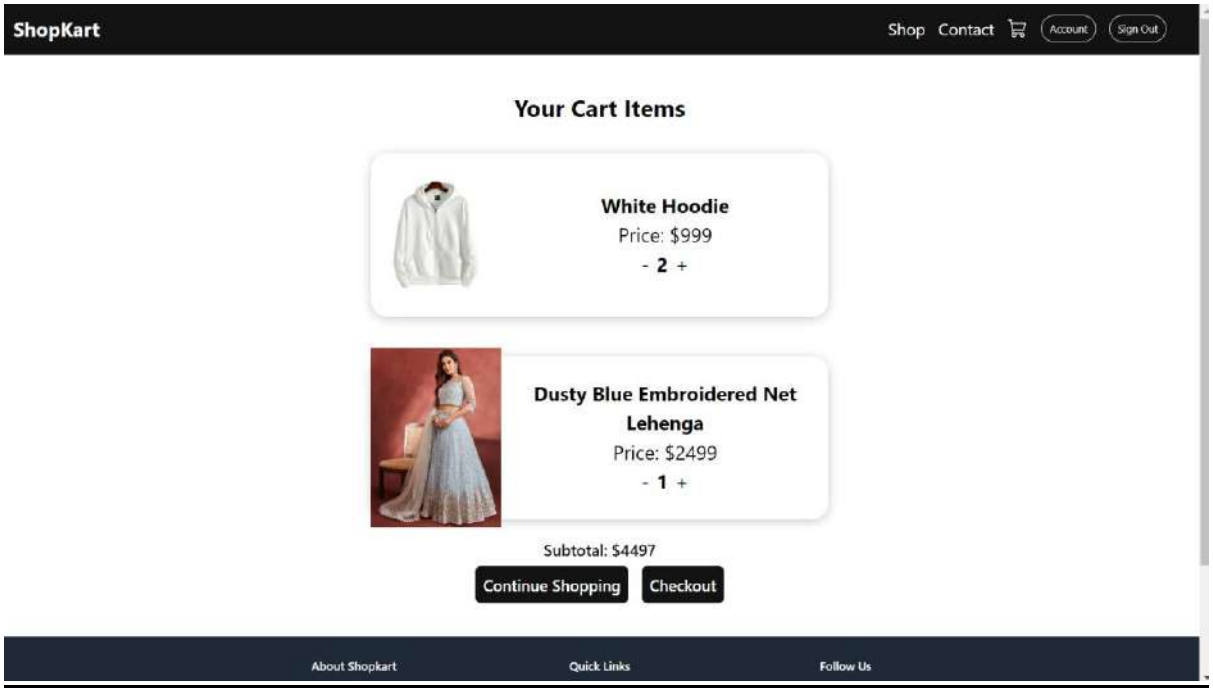


Landing page

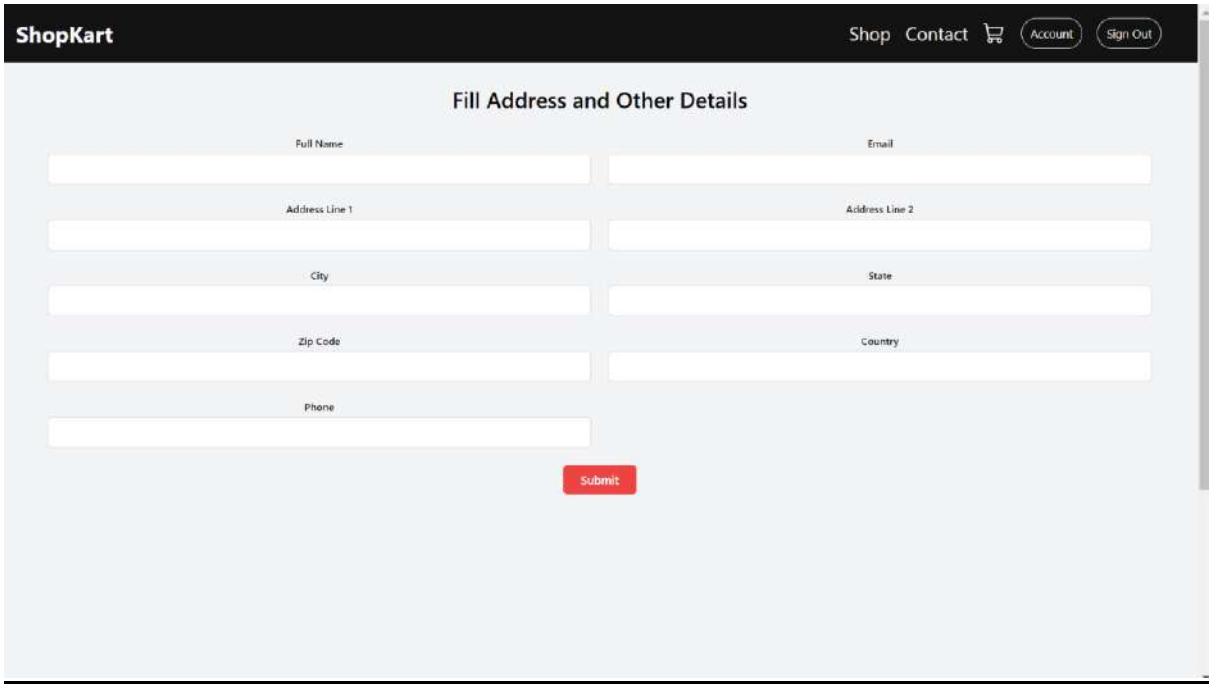



main.dqafgrygkxro4.amplifyapp.com/#/
New Chrome available
Gmail
YouTube
My files - OneDrive
ShopKart
Shop
Contact
Account
Sign Out
Red Dress
$2999
Add To Cart
Dusty Blue Embroidered Net Lehenga
$2499
Add To Cart (1)
Beige Printed Kurta
$1999
Add To Cart
Search

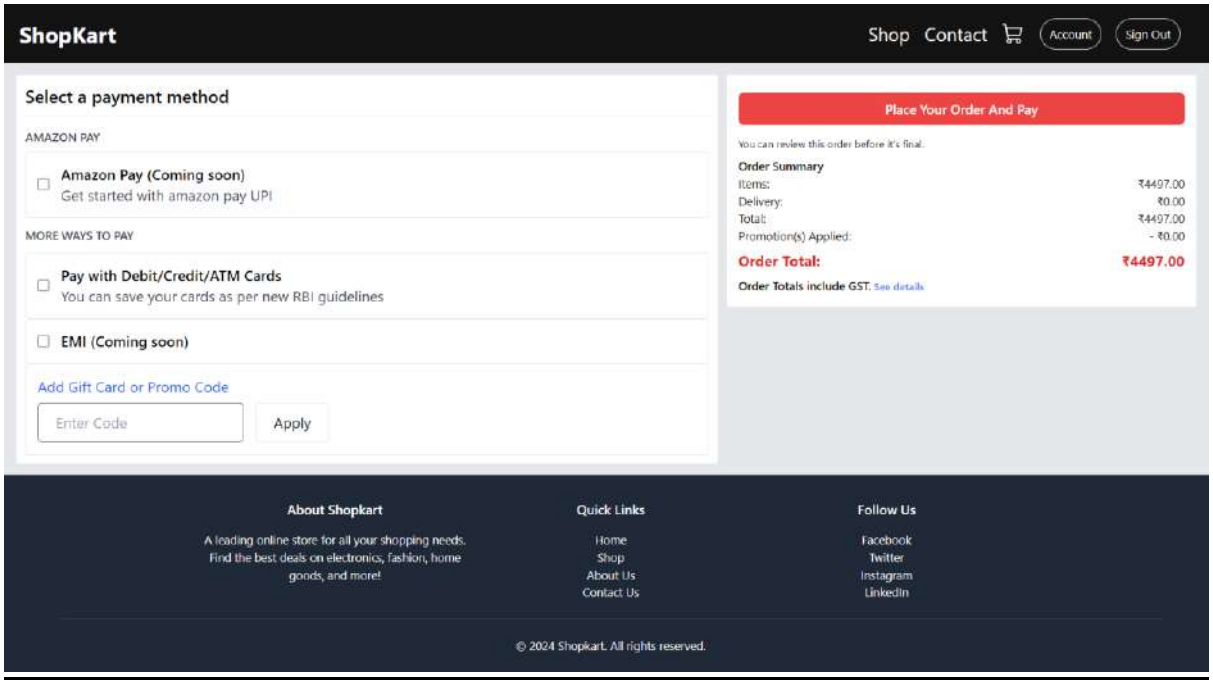
Cart page


ShopKart
Shop
Contact
Account
Sign Out
Your Cart Items
White Hoodie
Price: $999
- 2 +
Dusty Blue Embroidered Net Lehenga
Price: $2499
- 1 +
Subtotal: $4497
Continue Shopping
Checkout
About Shopkart
Quick Links
Follow Us

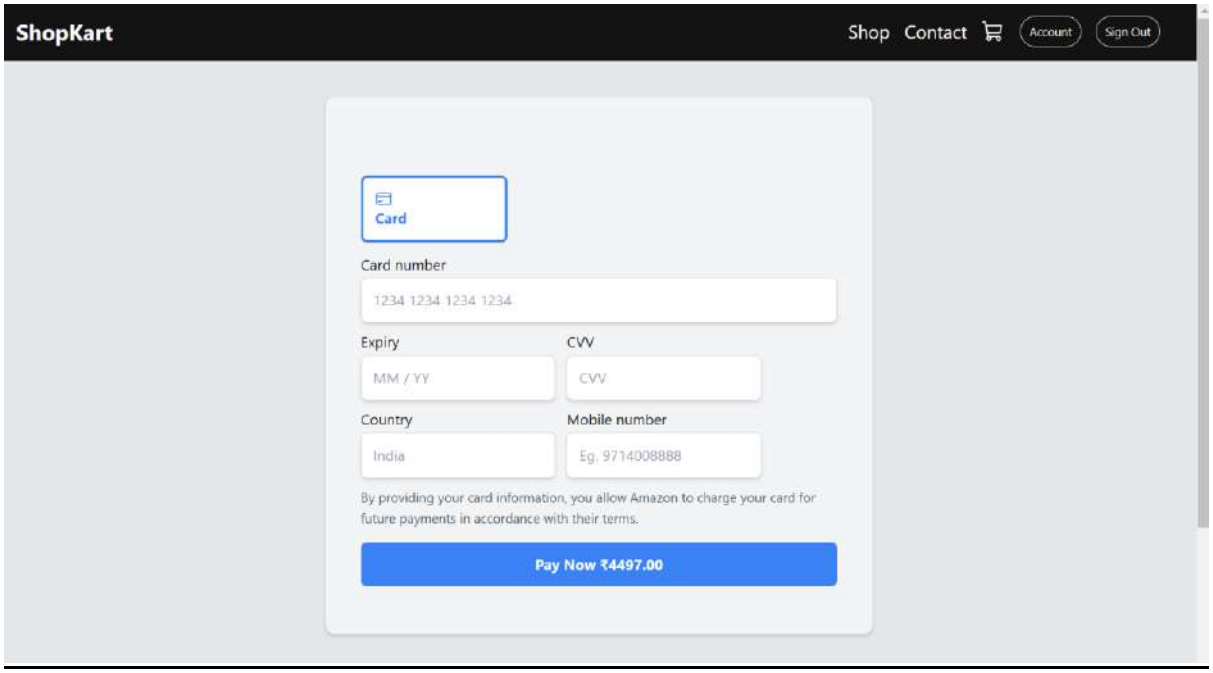
Fill in address details page

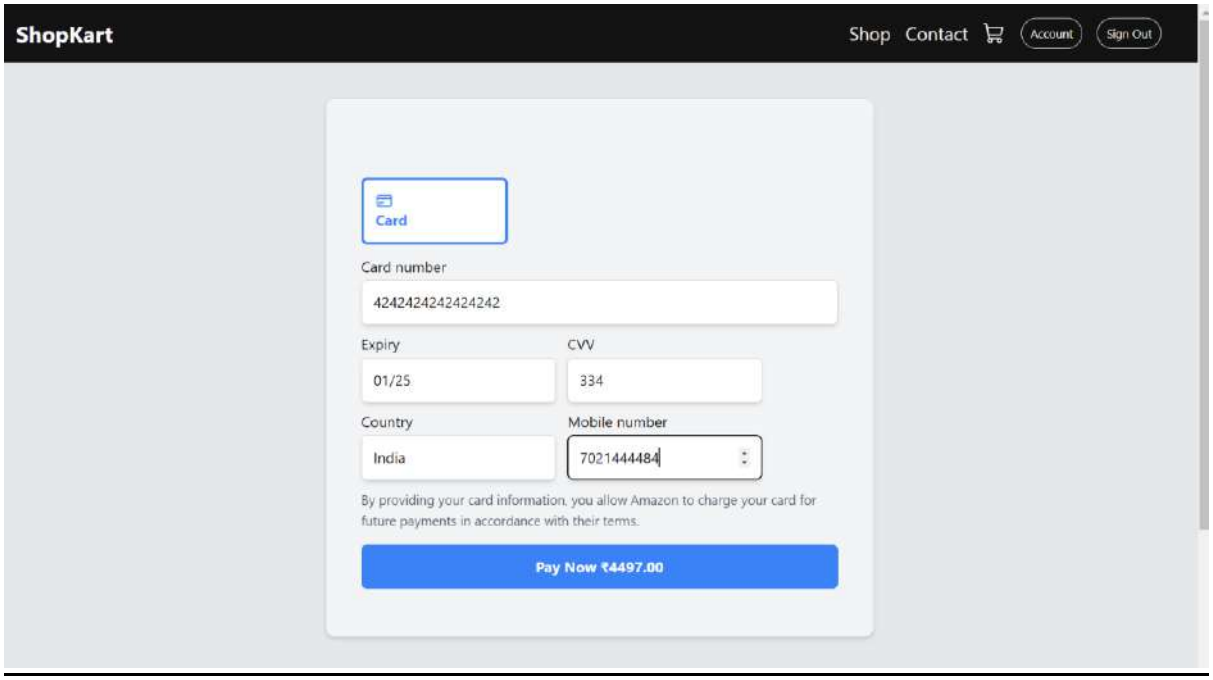


Select the payment method page

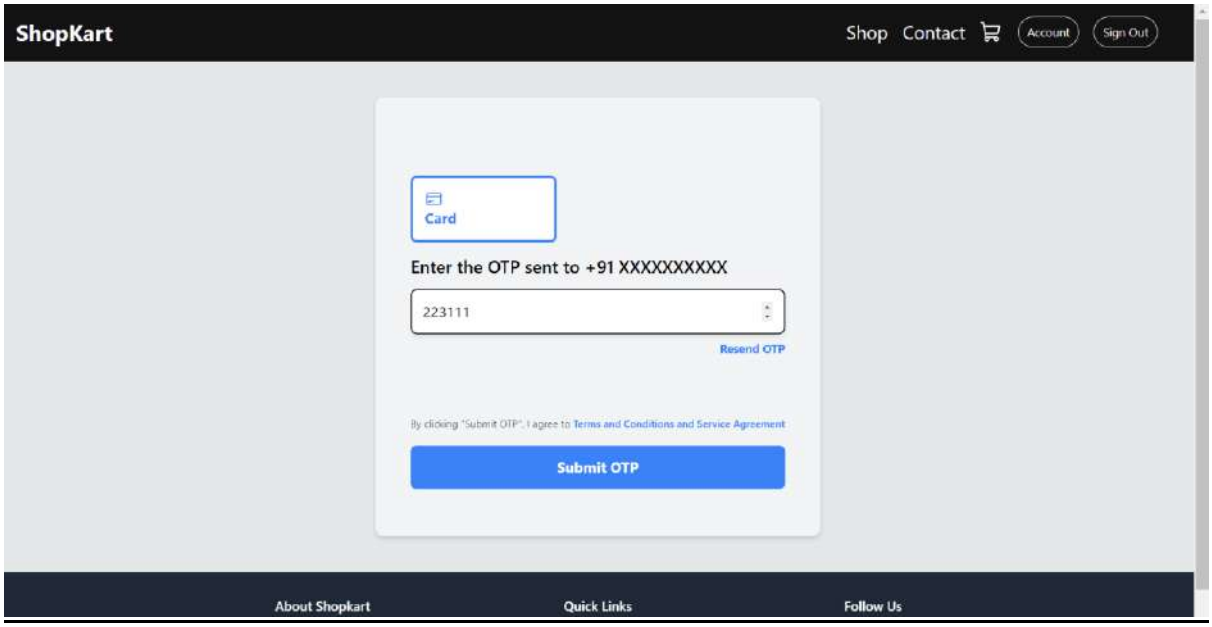


Enter the required payment method details page

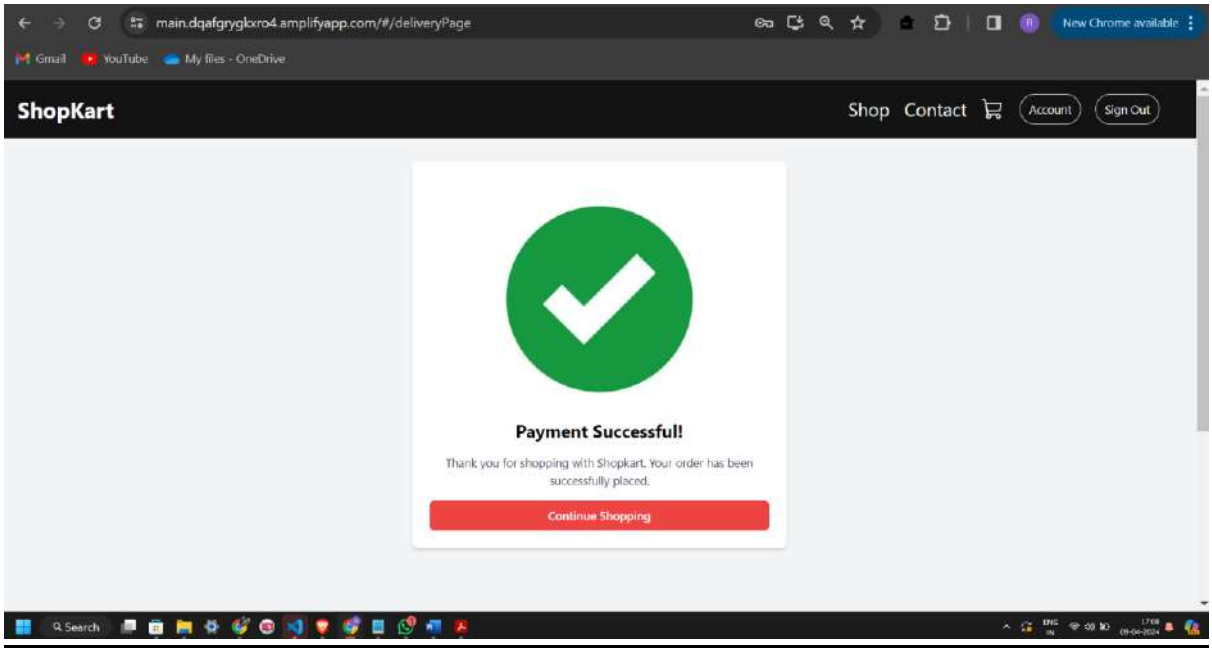




OTP authentication page



Order placed successfully/failed to place order status page



Conclusion:

In conclusion, Shop Kart is a cutting-edge e-commerce platform designed to offer users a seamless shopping experience for clothing and fashion items. Leveraging modern technologies like Amazon DynamoDB, AWS Amplify, and Amazon Cognito, Shop Kart ensures robustness, scalability, and security in its operations. With a user-friendly interface, extensive product catalogue, and secure checkout process, Shop Kart aims to become a top choice for online shoppers, catering to their diverse needs and preferences.