AWS Architecture to Develop and Host a Web Application

Prepared in the partial fulfillment of the Summer Internship Program on AWS

AT

Under the guidance of

Mrs. Sumana Bethala, APSSDC

Mr. Anil Kumar Varanasi, APSSDC

Submitted by

(Batch-258)

***Harsha Chenna** (23MH1A42D8) -Aditya College of Engineering and Technology (Surampalem)*

***Geetha Sravya Kadali** (23MH1A05Q5)-Aditya College of Engineering and Technology (Surampalem)*

***Modini Anupoju** (23MH1A42J3)-Aditya College of Engineering and Technology (Surampalem)*

***Dwaraka Pinapathruni** (23MH1A42J4 -Aditya College of Engineering and Technology (Surampalem)*

***Surya Teja Veeravalli** (23MH1A42I7)-Aditya College of Engineering and Technology (Suampalem)*

ACKNOWLEDGEMENT

I would like to express my heartfelt gratitude to all those who have contributed to the successful completion of my summer internship project at Andhra Pradesh Skill Development Corporation (APSSDC). This opportunity has been an enriching and transformative experience for me, and I am truly thankful for the support, guidance, and encouragement I have received along the way
.
First and foremost, I extend my sincere regards to Mrs Sumana Bethala and Mr Anil Kumar Varanasi, my supervisors and mentors, for providing me with valuable insights, constant guidance, and unwavering support throughout the duration of the internship. Their expertise and encouragement have been instrumental in shaping the direction of this project.

I would like to thank the entire team at Andhra Pradesh Skill Development Corporation (APSSDC) for fostering a collaborative and innovative environment. The camaraderie, knowledge sharing, and feedback I received from my colleagues significantly contributed to the development and success of this project.

In conclusion, I am honoured to have been a part of this internship program, and I look forward to leveraging the skills and knowledge gained to contribute positively to future endeavours.

Thank you.

Sincerely,

Harsha chenna (23MH1A42D8)

Kadali Geetha Sravya(23MH1A05Q5)

Modini Anupoju (23MH1A42J3)

Dwaraka Pinapathruni(23MH1A42J4)

Surya Teja Veeravalli(23MH1A42I7)

Abstract

This project presents the design and deployment of a scalable and secure two-tier web application architecture using Amazon Web Services (AWS). The system focuses on a Course Registration Form as a practical application to demonstrate key cloud infrastructure capabilities.The architecture separates the application tier (hosted on EC2 instance) from the database tier(Powered by Amazon RDS), enabling enhanced scalability, manageability and security.

The project leverages key AWS components such as Virtual Private Cloud (VPC), EC2, and RDS to ensure controlled networking, robust computation, and reliable data storage. The application is built using PHP for the frontend logic, interacting with a MySQL database hosted on RDS for backend operations. VPC subnets, routing rules, security groups, and NAT gateways are configured to manage network traffic and secure the infrastructure.

Through AWS Console, the deployment process is streamlined, offering real-time monitoring, automated backups, seamless scaling, and simplified maintenance. The proposed architecture ensures high availability, fault tolerance, and cost-effectiveness, making it ideal for real-world web applications with dynamic user loads.

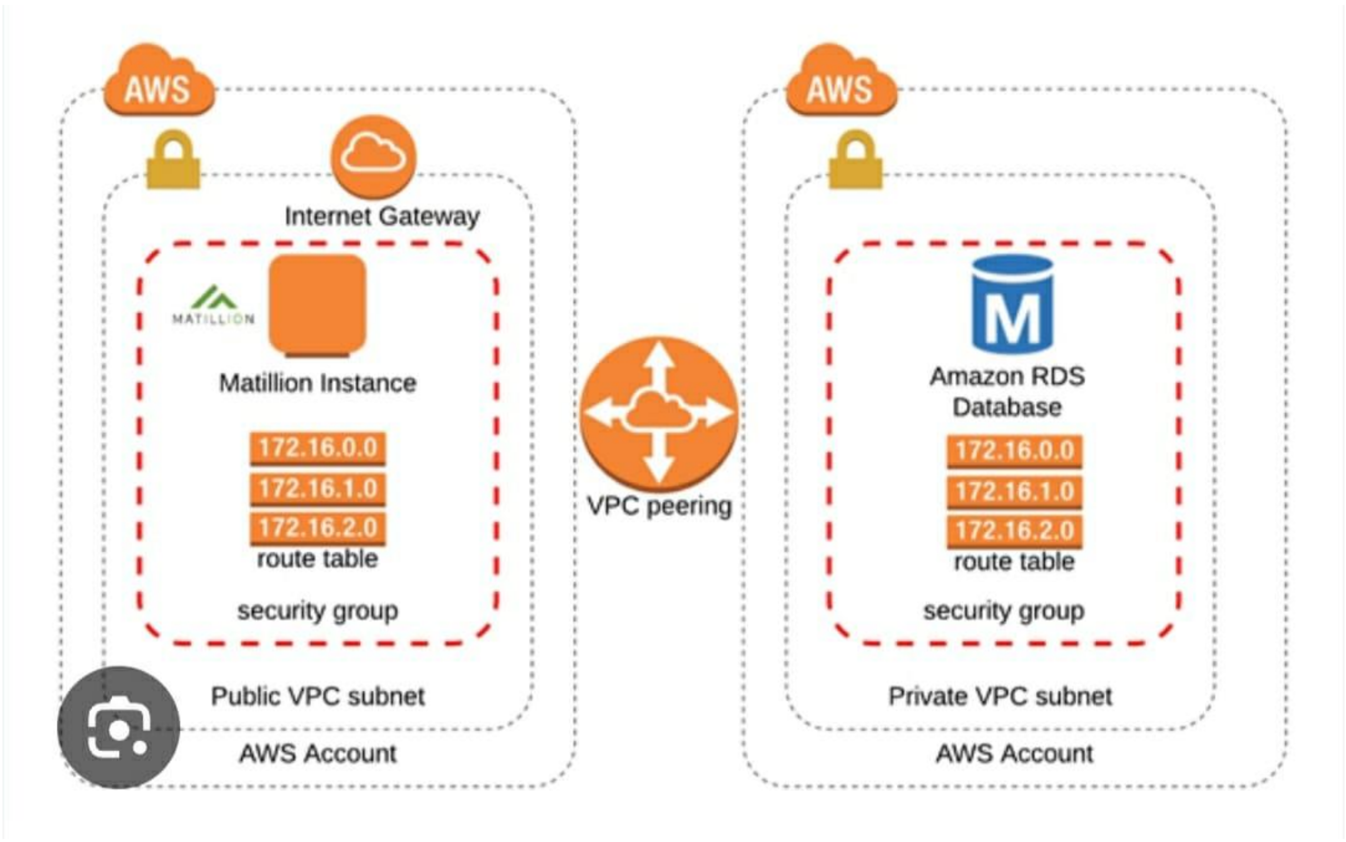
This project demonstrates how cloud-native development using AWS can deliver robust, maintainable, and efficient solutions for modern web services.

AWS architecture to develop and host a Web Application.

Services used:

- EC2 (Elastic Compute Cloud)
- VPC (Virtual Private Cloud)
  - Subnets
  - Route Tables
  - Internet Gateway
- Relation Database Services

Final architecture:



Description:

A Virtual Private Cloud (VPC) in AWS is a logically isolated network environment where users can launch AWS resources with complete control over networking configurations, such as IP address ranges, route tables, subnets, and security settings. It acts like a private data center in the cloud, offering enhanced security and flexibility. Inside a VPC, you can create multiple subnets, which are smaller segments of the network. Subnets can be categorized as public or private based on whether they have direct access to the internet. Public subnets are typically used for resources like web servers that need internet connectivity, while private subnets are used for backend systems like databases that should not be exposed to the public. When deploying a relational database using Amazon RDS (Relational Database Service), it is recommended to place it in private subnets to ensure security and prevent unauthorized external access. These subnets must be included in a DB Subnet Group, which is required by RDS to determine where to place the database instances across multiple Availability Zones for high availability and fault tolerance.

To allow an application or web server hosted on an EC2 instance in a public subnet to interact with the RDS database in a private subnet, proper networking and security configurations must be set up. This includes defining security groups that allow traffic on specific database ports (such as port 3306 for MySQL) from the EC2 instance to the RDS instance. Additionally, route tables should be configured to ensure proper routing within the VPC. Although the RDS database cannot be accessed from the internet, EC2 instances within the VPC can communicate with it securely using its private IP address. This architecture ensures that the database is protected from unauthorized access while still supporting efficient communication between application layers. By leveraging VPC, subnets, and RDS together, AWS enables users to build secure, scalable, and high-performing applications in the cloud.

Cloud computing

Cloud computing is on-demand access, via the internet, to computing resources—applications, servers (physical servers and virtual servers), data storage, development tools, networking capabilities, and more—hosted at a remote data center managed by a cloud services provider (or CSP). The CSP makes these resources available for a monthly subscription fee or bills them according to usage.

Compared to traditional on-premises IT, and depending on the cloud services you select, cloud computing helps do the following:

- **Lower IT costs:** Cloud lets you offload some or most of the costs and effort of purchasing, installing, configuring, and managing your own on-premises infrastructure.

- **Improve agility and time-to-value:** With cloud, your organization can start using enterprise applications in minutes, instead of waiting weeks or months for IT to respond to a request, purchase and configure supporting hardware, and install software. Cloud also lets you empower certain users—specifically developers and data scientists.

- **Scale more easily and cost-effectively:** Cloud provides elasticity—instead of purchasing excess capacity that sits unused during slow periods, you can scale capacity up and down in response to spikes and dips in traffic. You can also take advantage of your cloud provider's global network to spread your applications closer to users around the world.

The term 'cloud computing' also refers to the technology that makes cloud work. This includes some form of virtualized IT infrastructure—servers, operating system software, networking, and other infrastructure that's abstracted, using special software, so that it can be pooled and divided irrespective of physical hardware boundaries. For example, a single hardware server can be divided into multiple virtual servers.

Cloud Computing Services:

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

are the three most common models of cloud services, and it's not uncommon for an organization to use all three.

IaaS (Infrastructure-as-a-Service)

IaaS provides on-demand access to fundamental computing resources—physical and virtual servers, networking, and storage—over the internet on a pay-as-you-go basis. IaaS enables end users to scale and shrink resources on an as-needed basis, reducing the need for high, up-front capital expenditures or unnecessary on-premises or 'owned' infrastructure and for overbuying resources to accommodate periodic spikes in usage.

In contrast to SaaS and PaaS (and even newer PaaS computing models such as containers and serverless), IaaS provides the users with the lowest-level control of computing resources in the cloud.

IaaS was the most popular cloud computing model when it emerged in the early 2010s. While it remains the cloud model for many types of workloads, use of SaaS and PaaS is growing at a much faster rate.

PaaS (Platform-as-a-service)

PaaS provides software developers with on-demand platform—hardware, complete software stack, infrastructure, and even development tools—for running, developing, and managing applications without the cost, complexity, and inflexibility of maintaining that platform on-premises.

With PaaS, the cloud provider hosts everything—servers, networks, storage, operating system software, middleware, databases—at their data center. Developers simply pick from a menu to 'spin up' servers and environments they need to run, build, test, deploy, maintain, update, and scale applications.

Today, PaaS is often built around *containers*, a virtualized compute model one step removed from virtual servers. containers virtualize the operating system, enabling developers to package the application with only the operating system services it needs to run on any platform, without modification and without need for middleware.

SaaS (Software-as-a-Service)

SaaS—also known as cloud-based software or cloud applications—is application software that's hosted in the cloud, and that user's access via a web browser, a dedicated desktop client, or an API that integrates with a desktop or mobile operating system. In most cases, SaaS users pay a monthly or annual subscription fee; some may offer 'pay-as-you-go' pricing based on your actual usage.

In addition to the cost savings, time-to-value, and scalability benefits of cloud, SaaS offers the following:

- **Automatic upgrades:** With SaaS, users take advantage of new features as soon as the provider adds them, without having to orchestrate an on-premises upgrade.

- **Protection from data loss:** Because SaaS stores application data in the cloud with the application, users don't lose data if their device crashes or breaks.

SaaS is the primary delivery model for most commercial software today—there are hundreds of thousands of SaaS solutions available, from the most focused industry and departmental applications to powerful enterprise software database and AI (artificial intelligence) software.

Cloud Service Providers:

- Amazon Web Services
- Microsoft Azure
- Google Cloud Platform
- Oracle
- IBM cloud
- Salesforce

Amazon Web Services:

Amazon Web Services, Inc. (AWS) is a subsidiary of Amazon that provides on-demand cloud computing platforms and APIs to individuals, companies, and governments, on a metered, pay-as-you-go basis. Oftentimes, clients will use this in combination with autoscaling (a process that allows a client to use more computing in times of high application usage, and then scale down to reduce costs when there is less traffic). These cloud computing web services provide various services related to networking, computing, storage, middleware, IoT and other processing capacity, as well as software tools via AWS server farms. This frees clients from managing, scaling, and patching hardware, and operating systems.

One of the foundational services is Amazon Elastic Compute Cloud (EC2), which allows users to have at their disposal a virtual cluster of computers, with extremely high availability, which can be interacted with over the internet via REST APIs, a CLI or the AWS console. AWS's virtual computers emulate most of the attributes of a real computer, including hardware central processing units (CPUs) and graphics processing units (GPUs) for processing; local/RAM memory; hard disk /SSD storage; a choice of operating systems; networking; and pre-loaded application software such as web servers, databases, and customer relationship management (CRM).

AWS services are delivered to customers via a network of AWS server farms located throughout the world. Fees are based on a combination of usage (known as a "Pay-as-you-go" model), hardware, operating system, software, or networking features chosen by the subscriber required availability, redundancy, security, and service options. Subscribers can pay for a single virtual AWS computer, a dedicated physical computer, or clusters of either.

Amazon provides select portions of security for subscribers (e.g., physical security of the data centers) while other aspects of security are the responsibility of the subscriber (e.g., account management, vulnerability scanning, patching). AWS operates for many global geographical regions including seven in North America.

Amazon markets AWS to subscribers as a way of obtaining large-scale computing capacity more quickly and cheaply than building an actual physical server farm. All services are billed based on usage, but each service measures usage in varying ways. As of 2021 Q4, AWS has

33% market share for cloud infrastructure while the next two competitors Microsoft Azure and Google Cloud have 21%, and 10% respectively, according to Synergy Group

Why AWS?

- **Easy to use:**
  AWS is designed to allow application providers, ISVs, and vendors to host your applications quickly and securely – whether an existing application or a new SaaS-based application. You can use the AWS Management Console or well-documented web services APIs to access AWS's application hosting platform**.**

- **Flexible:**
  AWS enables you to select the operating system, programming language, web application platform, database, and other services you need. With AWS, you receive a virtual environment that lets you load the software and services your application requires. This eases the migration process for existing applications while preserving options for building new solutions.

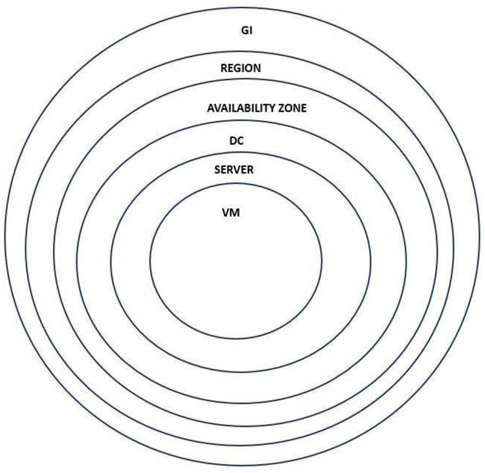
- **Cost-effective:**
  You pay only for the compute power, storage, and other resources you use, with no long-term contracts or up-front commitments. For more information on comparing the costs of other hosting alternatives with AWS, see the AWS Economics Center.

- **Reliable:**
  With AWS, you take advantage of a scalable, reliable, and secure global computing infrastructure, the virtual backbone of Amazon.com's multi-billion-dollar online business that has been honed for over a decade**.**

- **Scalable and High performance:**
  Using AWS tools, Auto Scaling, and Elastic Load Balancing, your application can scale up or down based on demand. Backed by Amazon's massive infrastructure, you have access to compute and storage resources when you need them.

- **Secure:**
  Using AWS tools, Auto Scaling, and Elastic Load Balancing, your application can scale up or down based on demand. Backed by Amazon's massive infrastructure, you have access to compute and storage resources when you need them.

Global Infrastructure:



1. Regions: AWS regions are geographic areas where AWS data centers are located. Each region is designed to be isolated from other regions to provide fault tolerance and resilience. AWS regions are further divided into Availability Zones.

2. Availability Zones (AZs): Availability Zones are physically separate data centers within a region. They are connected via high-speed, low-latency networks. The purpose of AZs is to provide redundancy and ensure that if one AZ goes down, services can automatically failover to another AZ within the same region.

3. Global Accelerator: AWS Global Accelerator is a service that utilizes the AWS global network to improve the performance and availability of your applications. It automatically routes traffic to the nearest AWS edge location, reducing latency and optimizing global network performance.

4. Direct Connect: AWS Direct Connect provides dedicated network connections between on-premises environments and AWS. These connections bypass the public internet, ensuring reliable and secure connectivity. Direct Connect locations are available in various regions around the world.

5. Regional Services: Some AWS services are region-specific, meaning they are available only in certain regions due to regional requirements or limitations. For example, certain compliance-related services or government-specific services may be available only in specific regions.

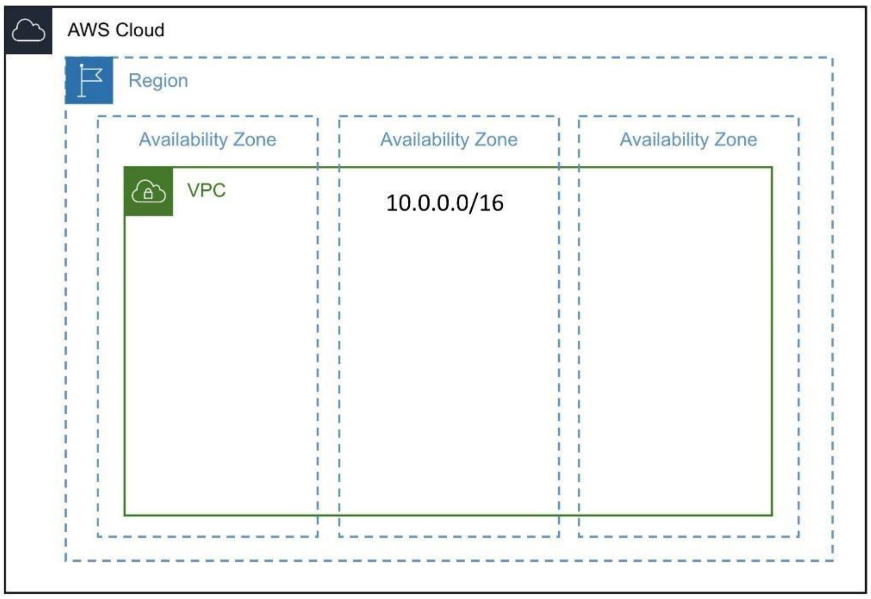
List of AWS Services:

Amazon, the preeminent cloud vendor, broke new ground by establishing the first cloud computing service, Amazon EC2, in 2008. AWS offers more solutions and features than any other provider and has free tiers with access to the AWS Console, where users can centrally control their ministrations.
Designed around ease-of-use for various skill sets, AWS is tailored for those unaccustomed to software development utilities. Web applications can be deployed in minutes with AWS facilities, without provisioning servers or writing additional code.

- Amazon EC2 (Elastic Compute Cloud)
- Amazon RDS (Relational Database Services)
- Amazon S3 (Simple Storage Service)
- Amazon Lambda
- Amazon Cognito
- Amazon Glacier
- Amazon SNS (Simple Notification Service)
- Amazon VPC (Virtual Private Cloud)
- Amazon Lightsail
- Amazon CloudWatch
- Amazon Cloud9
- Amazon Elastic Beanstalk
- Amazon CodeCommit
- Amazon IAM (Identity and Access Management)
- Amazon Inspector
- Amazon Kinesis
- Amazon Dynamo DB
- Amazon Codecatalyst
- Amazon Kinesis
- AWS Athena
- AWS Amplify
- AWS Quicksight
- AWS Cloudformation

Amazon VPC:

Amazon Virtual Private Cloud (VPC) is a commercial cloud computing service that provides a virtual private cloud, by provisioning a logically isolated section of Amazon Web Services (AWS) Cloud. Enterprise customers are able to access the Amazon Elastic Compute Cloud (EC2) over an IPsec based virtual private network. Unlike traditional EC2 instances which are allocated internal and external IP numbers by Amazon, the customer can assign IP numbers of their choosing from one or more subnets.



Amazon Web Services launched Amazon Virtual Private Cloud on 26 August 2009, which allows the Amazon Elastic Compute Cloud service to be connected to legacy infrastructure over an IPsec VPN.In AWS, the basic VPC is free to use, with users being charged by usage for additional features.EC2 and RDS instances running in a VPC can also be purchased using Reserved Instances, however will have a limitation on resources being guaranteed.[citation needed]

IBM Cloud launched IBM Cloud VPC on 4 June 2019, provides an ability to manage virtual machine-based compute, storage, and networking resources. Pricing for IBM Cloud Virtual Private Cloud is applied separately for internet data transfer, virtual server instances, and block storage used within IBM Cloud VPC.

Google Cloud Platform resources can be provisioned, connected, and isolated in a virtual private cloud (VPC) across all GCP regions. With GCP, VPCs are global resources and subnets within that VPC are regional resources. This allows users to connect zones and regions without the use of additional networking complexity as all data travels, encrypted in transit and at rest, on Google's own global, private network. Identity management policies and security rules allow for private access to Google's storage, big data, and analytics managed services. VPCs on Google Cloud Platform leverage the security of Google's data centers.

Amazon EC2:

Amazon Elastic Compute Cloud (EC2) is a part of Amazon.com's cloud- computing platform, Amazon Web Services (AWS), that allows users to rent virtual computers on which to run their own computer applications. EC2 encourages scalable deployment of applications by providing a web service through which a user can boot an Amazon Machine Image (AMI) to configure a virtual machine, which Amazon calls an "instance", containing any software desired. A user can create, launch, and terminate server- instances as needed, paying by the second for active servers – hence the term "elastic". EC2 provides users with control over the geographical location of instances that allows for latency optimization and high levels of redundancy. In November 2010, Amazon switched its own retail website platform to EC2 and AWS.

Amazon announced a limited public beta test of EC2 on August 25, 2006, offering access on a first-come, first-served basis. Amazon added two new instance types (Large and Extra-Large) on October 16, 2007. On May 29, 2008, two more types were added, High-CPU Medium and High-CPU Extra Large. There were twelve types of instances available.

Amazon added three new features on March 27, 2008, static IP addresses, availability zones, and user selectable kernels. On August 20, 2008, Amazon added Elastic Block Store (EBS) This provides persistent storage, a feature that had been lacking since the service was introduced.

Instance types:

Initially, EC2 used Xen virtualization exclusively. However, on November 6, 2017, Amazon announced the new C5 family of instances that were based on a custom architecture around the KVM hypervisor, called Nitro. Each virtual machine, called an "instance", functions as a virtual private server. Amazon sizes instances based on "Elastic Compute Units". The performance of otherwise identical virtual machines may vary. On November 28, 2017, AWS announced a bare-metal instance type offering marking a remarkable departure from exclusively offering virtualized instance types.

As of January 2019, the following instance types were offered:

- General Purpose: A1, T3, T2, M5, M5a, M4, T3a
- Compute Optimized: C5, C5n, C4
- Memory Optimized: R5, R5a, R4, X1e, X1, High Memory, z1d
- Accelerated Computing: P3, P2, G3, F1
- Storage Optimized: H1, I3, D2

As of April 2018, the following payment methods by instance were offered:

- On-demand: pay by the hour without commitment.

- Reserved: rent instances with one-time payment receiving discounts on the hourly charge.
- Spot: bid-based service runs the jobs only if the spot price is below the bid specified by bidder. The spot price is claimed to be supply-demand based, however a 2011 study concluded that the price was generally not set to clear the market but was dominated by an undisclosed reserve price.

1. Virtual Private Cloud (VPC) and Subnets in AWS

A **Virtual Private Cloud (VPC)** in Amazon Web Services (AWS) is a logically isolated section of the AWS cloud where users can define and control their virtual network. It allows complete control over resources such as servers (EC2), databases (RDS), and other AWS services, all within a secure and customizable network environment. A VPC is created with a defined **CIDR (Classless Inter-Domain Routing)** block, which represents the IP address range. Users can create subnets, assign IP addresses, configure route tables, and manage traffic flow using gateways and security mechanisms.

Within a VPC, the network is typically divided into **subnets**, which are smaller ranges of IP addresses within the larger VPC CIDR block. Subnets help organize the architecture, enable better traffic control, and enhance security. AWS best practices recommend placing subnets across multiple **Availability Zones (AZs)** to ensure high availability and fault tolerance. Subnets can be **public** or **private**, depending on the requirements of the resources deployed within them.

A **public subnet** is one that has a route to an **Internet Gateway**, allowing resources like web servers to communicate with the internet. These are used for hosting applications, APIs, or load balancers that require external access. In contrast, a **private subnet** has no direct internet connectivity and is ideal for resources that should not be publicly accessible, such as backend application servers, databases, and internal services. If internet access is needed from a private subnet (e.g., for updates), a **NAT Gateway** or **NAT instance** can be used to allow outbound traffic without exposing the resource to incoming requests.

Route tables control how traffic is directed within the VPC. Each subnet is associated with a route table, which defines where network traffic should be sent. Another critical layer of control is provided by **Security Groups** and **Network Access Control Lists (NACLs)**. Security groups act as virtual firewalls for instances, while NACLs provide stateless control at the subnet level.

By leveraging VPC and subnets, organizations can build secure, multi-tier applications. For example, a common architecture would involve a web server in a public subnet, an application server in a private subnet, and a database server in another private subnet. This separation ensures that sensitive data and operations are well protected. VPC also supports **peering**, **VPN**, and **Direct Connect**, enabling secure communication across multiple VPCs or with on-premises infrastructure.

In conclusion, AWS VPC and its subnets form the foundational networking layer for cloud applications. They provide a robust, flexible, and secure environment that supports complex and scalable infrastructure designs.

2. Amazon RDS and Its Role Within a VPC

Amazon RDS (Relational Database Service) is a fully managed database service provided by AWS that simplifies the process of setting up, operating, and scaling relational databases in the cloud. It supports several major database engines, including MySQL, PostgreSQL, MariaDB, Oracle, and Microsoft SQL Server. RDS removes the burden of infrastructure management, handling tasks like backups, patching, monitoring, and automatic failover.

When an RDS instance is launched, it is deployed inside an **Amazon VPC**, ensuring that the database resides in a secure, isolated network environment. This setup gives you complete control over who can access the database and from where. The RDS instance must be associated with a **DB Subnet Group**, which includes **private subnets** in at least two **Availability Zones (AZs)**. This design ensures high availability, as AWS can automatically switch to another AZ in case of failure, keeping the database online.

RDS instances placed in private subnets are protected from direct internet access, significantly improving security. Only resources within the VPC, such as EC2 instances or Lambda functions, can access the database, provided proper permissions are configured. Access is further controlled using **security groups**, which define inbound and outbound traffic rules. For example, an application server in a public subnet can connect to RDS on port 3306 (for MySQL), but only if allowed by both the RDS and EC2 security groups.

RDS also supports **Multi-AZ deployments**, where a standby instance is automaticallmaintained in another AZ. In case of hardware failure or network disruption, RDS fails over to the standby, ensuring minimal downtime. Additionally, **read replicas** can be configured to handle read-heavy workloads or support disaster recovery strategies.

RDS offers automated backups, snapshot capabilities, and point-in-time recovery, which aessential for protecting data. Storage can be scaled up easily without impacting performance, and different storage types (such as General Purpose SSD, Provisioned IOPS, or Magnetic) are available depending on workload needs.

An important aspect of deploying RDS inside a VPC is its **integration with other AWS services**. For example, applications running on EC2 can connect to RDS using private IPs. Services like AWS Lambda and ECS can also be configured to access RDS through **VPC networking interfaces**.

In summary, Amazon RDS provides a powerful, secure, and easy-to-manage solution for deploying relational databases in the cloud. Its tight integration with VPC, support for high availability, and automated management features make it an ideal choice for modern cloud-native applications.

3. MobaXterm as a Platform for AWS Access and Management

MobaXterm is a comprehensive remote computing tool designed for Windows users that brings together a wide range of Unix and networking commands into a single, portable application. It is particularly useful for developers and system administrators working with **cloud platforms like AWS**, where accessing Linux-based servers and managing resources is a daily necessity. MobaXterm supports **SSH, SFTP, RDP, VNC, and X11 forwarding**, allowing users to securely connect to and control remote servers.

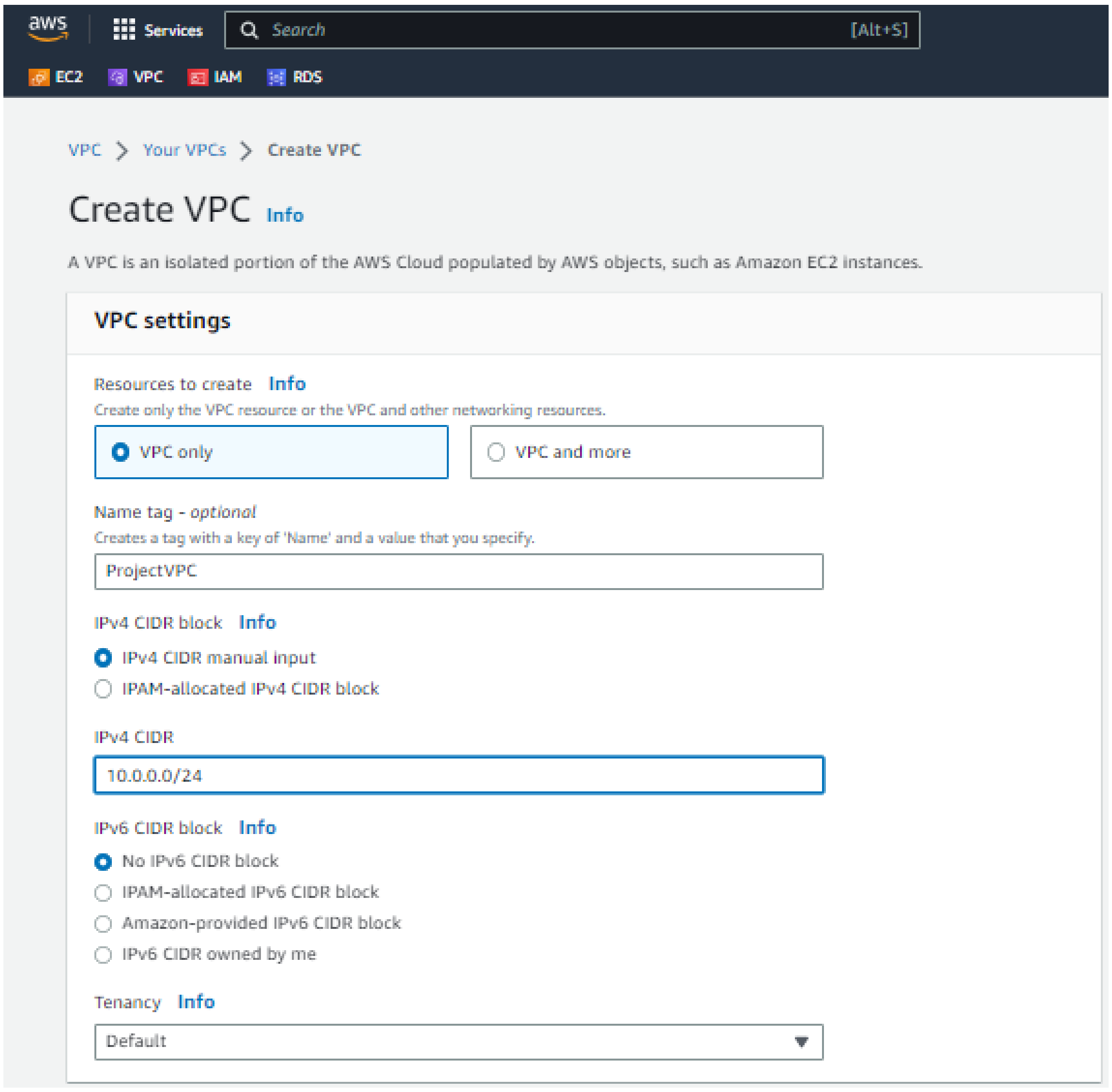
In the context of AWS, MobaXterm is commonly used to access **EC2 instances** hosted inside a **VPC (Virtual Private Cloud)**. For example, when a Linux server is launched in a **public subnet**, users can connect to it using **SSH** via the MobaXterm terminal. By uploading a private key (.pem file), users can easily establish a secure session and gain full command-line access to the remote instance. This access allows users to install software, configure services, run web servers, or manage backend scripts.

One of the powerful features of MobaXterm is its ability to act as a **bastion host interface**, also known as an **SSH jump host**. In scenarios where **RDS databases** are located inside private subnets and are not directly accessible from the internet, users can use MobaXterm to first connect to an EC2 instance in a public subnet. From there, they can create an **SSH tunnel** to securely access the RDS instance or other internal resources. This method maintains strict security while allowing necessary access for configuration and maintenance tasks.

In addition to command-line access, MobaXterm includes a built-in **SFTP browser** that opens automatically when you connect via SSH. This makes it incredibly easy to upload or download files from the remote EC2 instance without using a separate file transfer application. Whether you're uploading web application files, editing configuration files, or managing logs, the integrated file manager is efficient and user-friendly.

Another advantage is the ability to open multiple tabs or sessions, allowing users to manage several servers simultaneously. MobaXterm also supports macros and remote session saving, which helps streamline repetitive tasks and boosts productivity. Its X11 server enables GUI-based applications from the remote Linux server to be displayed on the local Windows machine.

In conclusion, MobaXterm is an essential tool for AWS users who require reliable and efficient access to cloud-based Linux environments. It simplifies remote management, enhances productivity, and enables secure access to even the most isolated AWS resources, making it an ideal companion for anyone working with EC2, RDS, and VPC-based architectures.



Create VPC Info

A VPC is an isolated portion of the AWS Cloud populated by AWS objects, such as Amazon EC2 instances.

VPC settings

Resources to create Info
Create only the VPC resource or the VPC and other networking resources.
[✓ VPC only] VPC and more

Name tag - *optional*
Creates a tag with a key of 'Name' and a value that you specify.
ProjectVPC

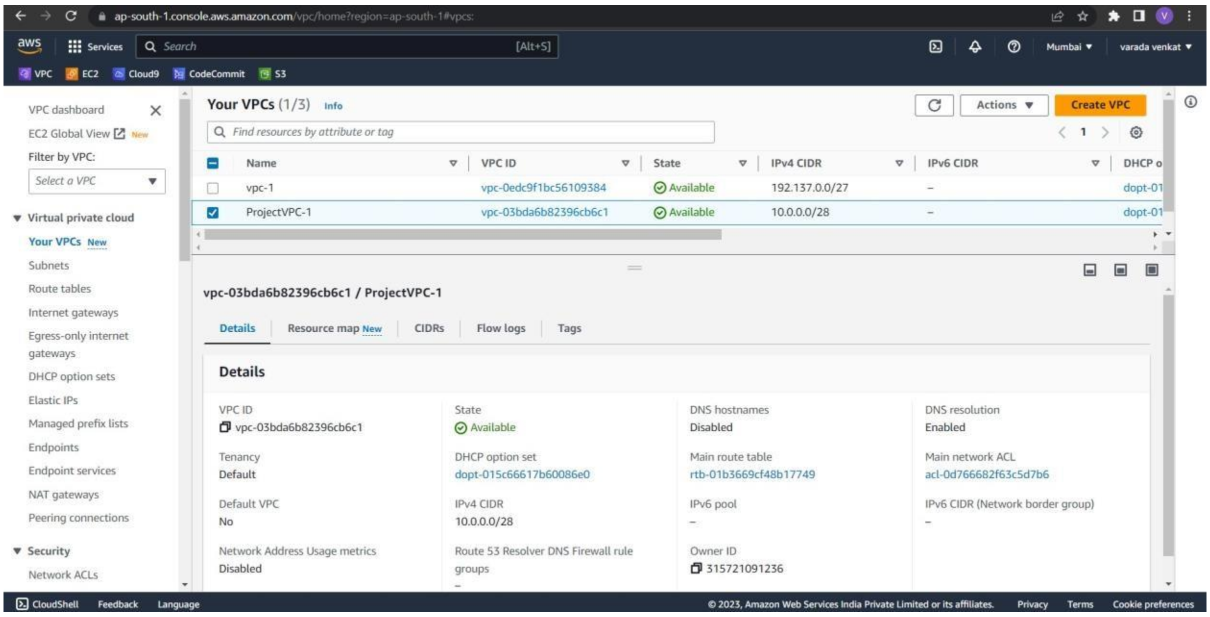
IPv4 CIDR block Info
[✓ IPv4 CIDR manual input]
IPAM-allocated IPv4 CIDR block

IPv4 CIDR
10.0.0.0/24

IPv6 CIDR block Info
[✓ No IPv6 CIDR block]
IPAM-allocated IPv6 CIDR block
Amazon-provided IPv6 CIDR block
IPv6 CIDR owned by me

Tenancy Info
Default

- After creating the VPC you can view it in your VPCs.



ap-south-1.console.aws.amazon.com/vpc/home?region=ap-south-1#vpcs:

Your VPCs (1/3) Info

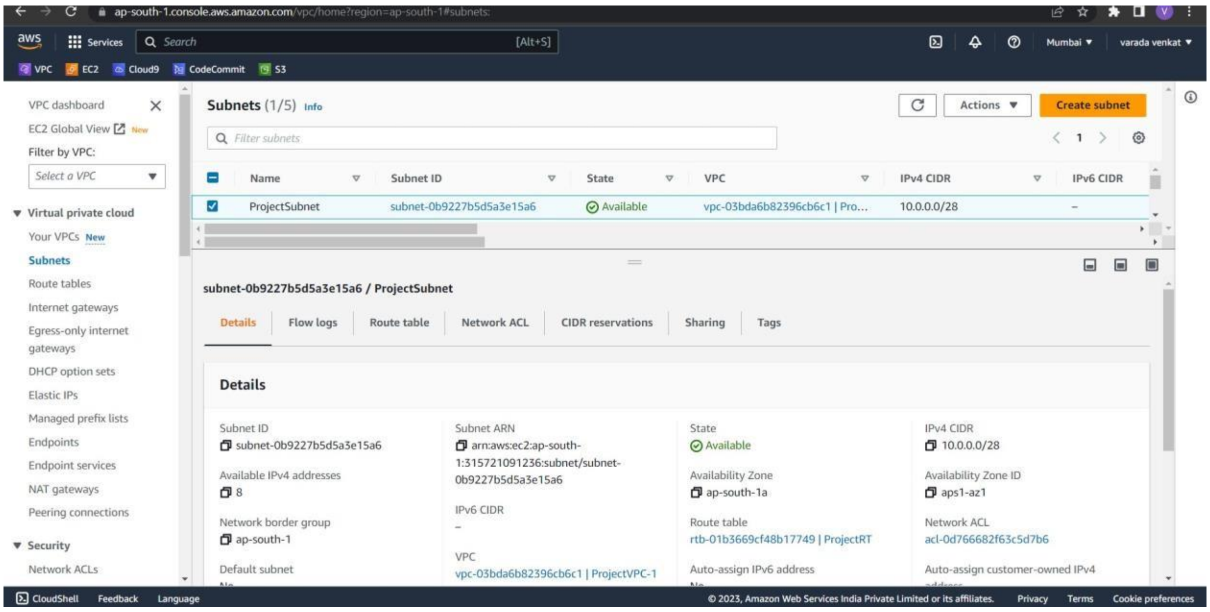
Name	VPC ID	State	IPv4 CIDR	IPv6 CIDR	DHCP o
vpc-1	vpc-0edc9f1bc56109384	Available	192.137.0.0/27	–	dopt-01
ProjectVPC-1	vpc-03bda6b82396cb6c1	Available	10.0.0.0/28	–	dopt-01

vpc-03bda6b82396cb6c1 / ProjectVPC-1

Details

VPC ID vpc-03bda6b82396cb6c1	State Available	DNS hostnames Disabled	DNS resolution Enabled
Tenancy Default	DHCP option set dopt-015c66617b60086e0	Main route table rtb-01b3669cf48b17749	Main network ACL acl-0d766682f63c5d7b6
Default VPC No	IPv4 CIDR 10.0.0.0/28	IPv6 pool –	IPv6 CIDR (Network border group) –
Network Address Usage metrics Disabled	Route 53 Resolver DNS Firewall rule groups –	Owner ID 315721091236	

- Create a subnet for the newly created VPC.



ap-south-1.console.aws.amazon.com/vpc/home?region=ap-south-1#subnets:

Subnets (1/5) Info

Name	Subnet ID	State	VPC	IPv4 CIDR	IPv6 CIDR
ProjectSubnet	subnet-0b9227b5d5a3e15a6	Available	vpc-03bda6b82396cb6c1 | Pro...	10.0.0.0/28	–

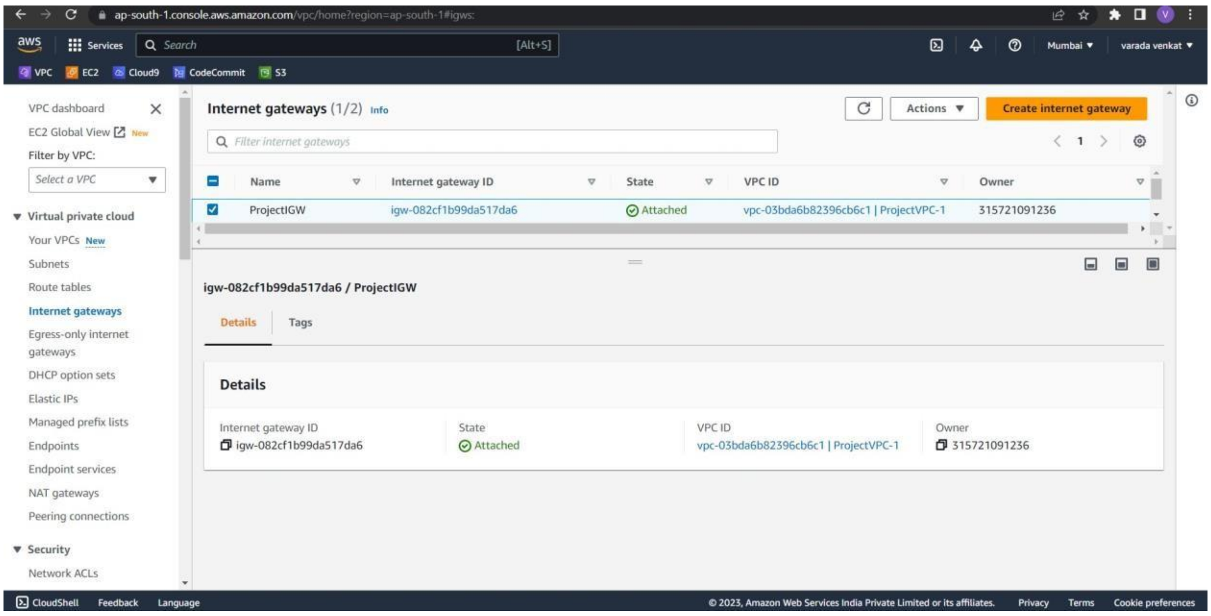
subnet-0b9227b5d5a3e15a6 / ProjectSubnet

Details

Subnet ID subnet-0b9227b5d5a3e15a6	Subnet ARN arn:aws:ec2:ap-south-1:315721091236:subnet/subnet-0b9227b5d5a3e15a6	State Available	IPv4 CIDR 10.0.0.0/28
Available IPv4 addresses 8	IPv6 CIDR –	Availability Zone ap-south-1a	Availability Zone ID aps1-az1
Network border group ap-south-1	VPC vpc-03bda6b82396cb6c1 | ProjectVPC-1	Route table rtb-01b3669cf48b17749 | ProjectRT	Network ACL acl-0d766682f63c5d7b6
Default subnet No		Auto-assign IPv6 address	Auto-assign customer-owned IPv4 address

- Create an Internet Gateway for the instances for communication and share data.

- Attach the newly created Internet Gateway (IGW) to the VPC.



Internet gateways (1/2) Info

Name	Internet gateway ID	State	VPC ID	Owner
ProjectIGW	igw-082cf1b99da517da6	Attached	vpc-03bda6b82396cb6c1 | ProjectVPC-1	315721091236

igw-082cf1b99da517da6 / ProjectIGW

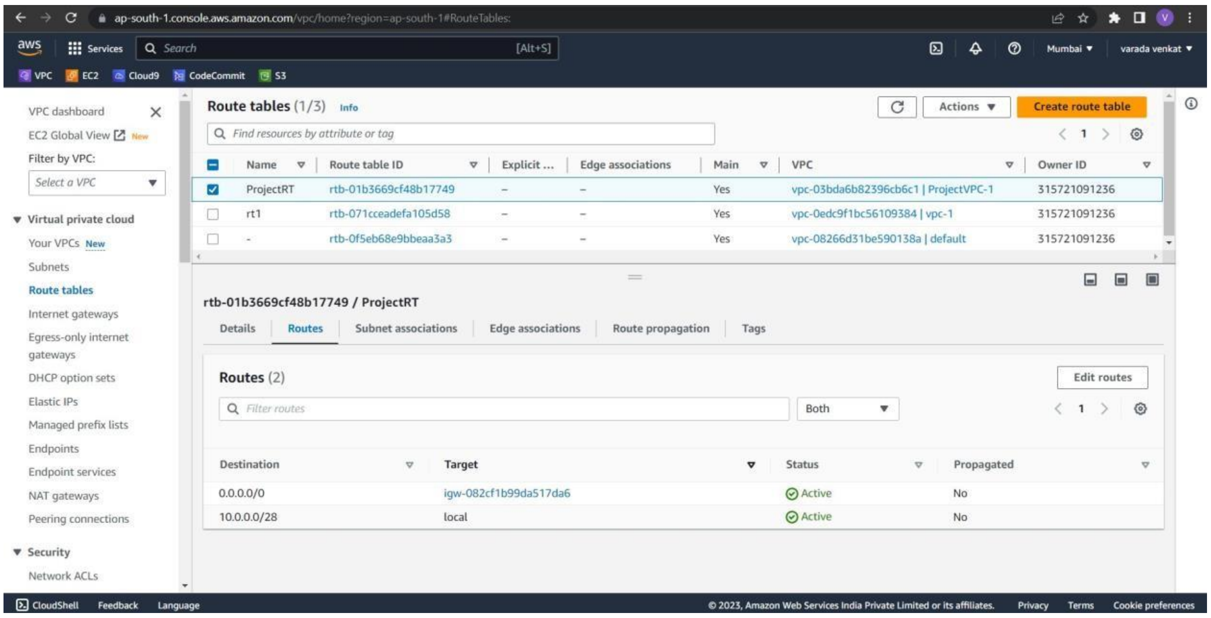
Details | Tags

Details

Internet gateway ID	State	VPC ID	Owner
igw-082cf1b99da517da6	Attached	vpc-03bda6b82396cb6c1 | ProjectVPC-1	315721091236

© 2023, Amazon Web Services India Private Limited or its affiliates.

- Enter the address of the IGW in the respective route table of the VPC.



Route tables (1/3) Info

Name	Route table ID	Explicit ...	Edge associations	Main	VPC	Owner ID
ProjectRT	rtb-01b3669cf48b17749	–	–	Yes	vpc-03bda6b82396cb6c1 | ProjectVPC-1	315721091236
rt1	rtb-071cceadefa105d58	–	–	Yes	vpc-0edc9f1bc56109384 | vpc-1	315721091236
-	rtb-0f5eb68e9bbeaa3a3	–	–	Yes	vpc-08266d31be590138a | default	315721091236

rtb-01b3669cf48b17749 / ProjectRT

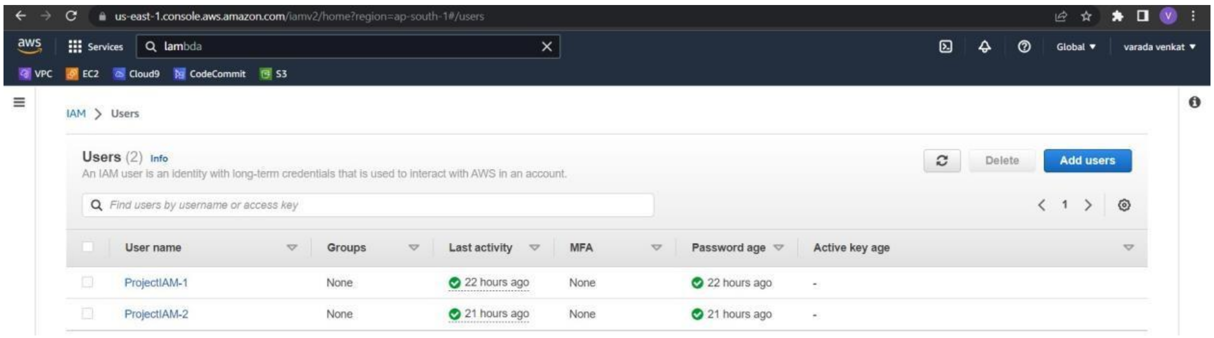
Details | Routes | Subnet associations | Edge associations | Route propagation | Tags

Routes (2)

Destination	Target	Status	Propagated
0.0.0.0/0	igw-082cf1b99da517da6	Active	No
10.0.0.0/28	local	Active	No

© 2023, Amazon Web Services India Private Limited or its affiliates.

- Create two IAM users for the root users.



us-east-1.console.aws.amazon.com/iamv2/home?region=ap-south-1#/users

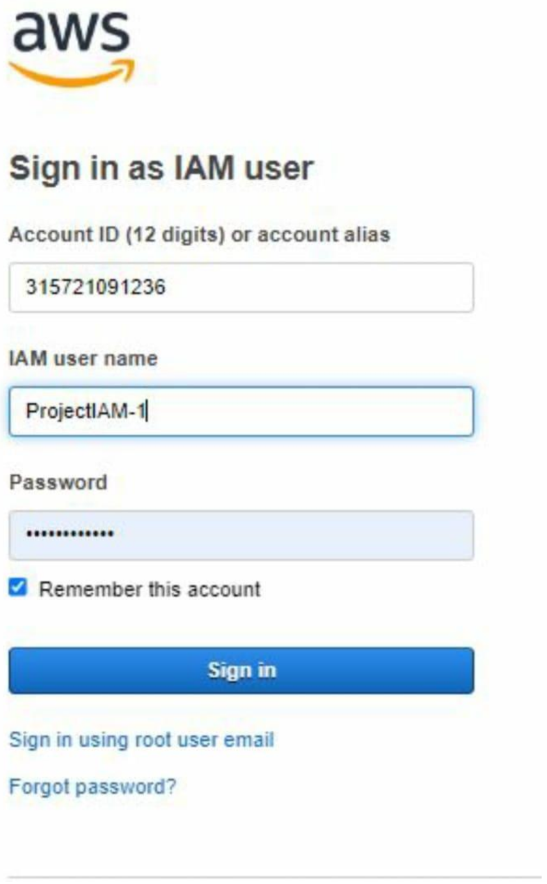
IAM > Users

Users (2) Info

An IAM user is an identity with long-term credentials that is used to interact with AWS in an account.

User name	Groups	Last activity	MFA	Password age	Active key age
ProjectIAM-1	None	22 hours ago	None	22 hours ago	-
ProjectIAM-2	None	21 hours ago	None	21 hours ago	-

- Login with the credentials given for the IAM users.



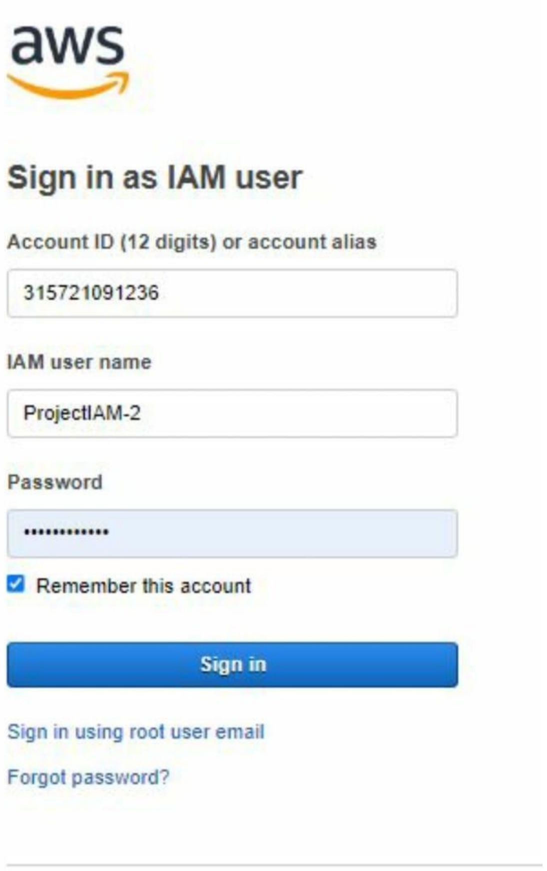
Sign in as IAM user

Account ID (12 digits) or account alias: 315721091236

IAM user name: ProjectIAM-1

Password: ••••••••••••

☑ Remember this account



Sign in as IAM user

Account ID (12 digits) or account alias: 315721091236

IAM user name: ProjectIAM-2

Password: ••••••••••••

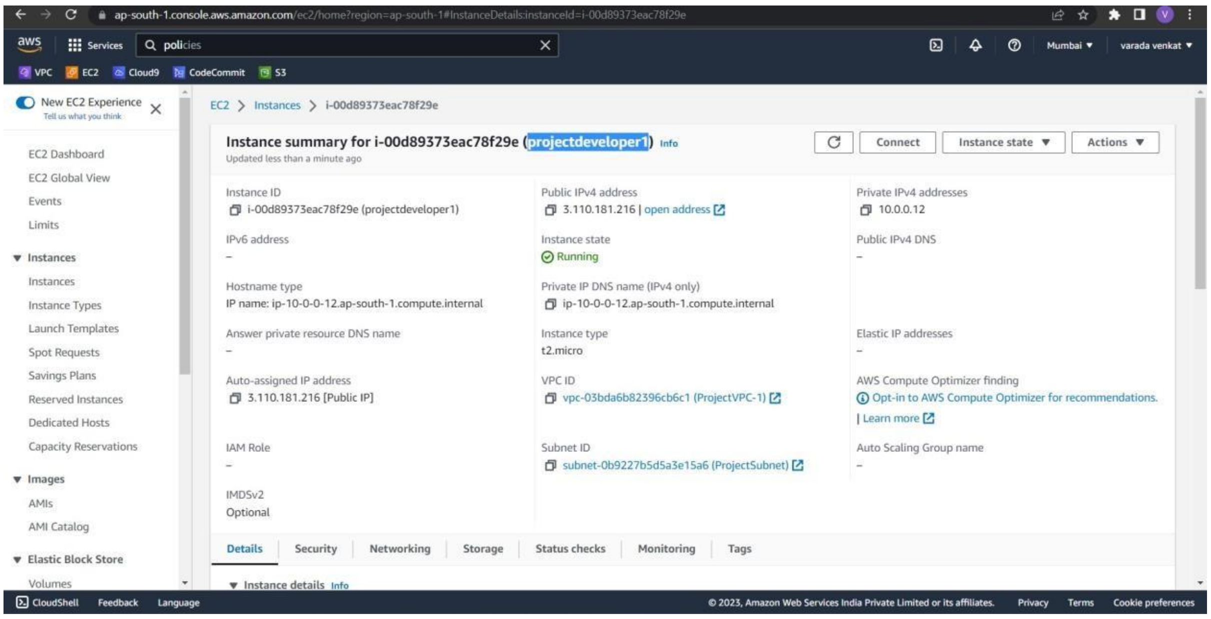
☑ Remember this account

Creating instances in VPC (Virtual Private Cloud):

Steps to create an instance:

1. Goto launch instance.
2. Enter Name, AMI (Amazon Machine Image), Instance(T2-micro), Key-Pair, VPC, Security Group, Subnet.
3. Allow SSH, All Traffic and HTTP in the Security group.

The Instance created in the first IAM User:



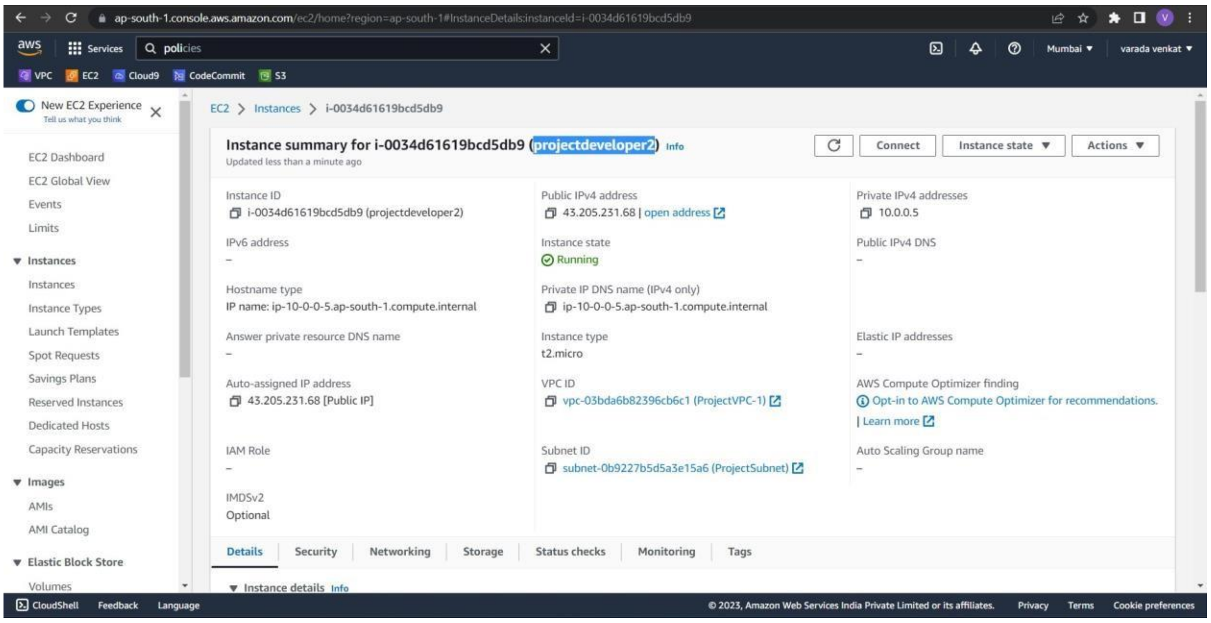
ap-south-1.console.aws.amazon.com/ec2/home?region=ap-south-1#InstanceDetails:instanceId=i-00d89373eac78f29e

EC2 > Instances > i-00d89373eac78f29e

Instance summary for i-00d89373eac78f29e (projectdeveloper1) Info
Updated less than a minute ago

Instance ID: i-00d89373eac78f29e (projectdeveloper1)
Public IPv4 address: 3.110.181.216 | open address
Private IPv4 addresses: 10.0.0.12
IPv6 address: –
Instance state: Running
Public IPv4 DNS: –
Hostname type: IP name: ip-10-0-0-12.ap-south-1.compute.internal
Private IP DNS name (IPv4 only): ip-10-0-0-12.ap-south-1.compute.internal
Answer private resource DNS name: –
Instance type: t2.micro
Elastic IP addresses: –
Auto-assigned IP address: 3.110.181.216 [Public IP]
VPC ID: vpc-03bda6b82396cb6c1 (ProjectVPC-1)
AWS Compute Optimizer finding: Opt-in to AWS Compute Optimizer for recommendations. | Learn more
IAM Role: –
Subnet ID: subnet-0b9227b5d5a3e15a6 (ProjectSubnet)
Auto Scaling Group name: –
IMDSv2: Optional

The Instance created in the second IAM User:



ap-south-1.console.aws.amazon.com/ec2/home?region=ap-south-1#InstanceDetails:instanceId=i-0034d61619bcd5db9

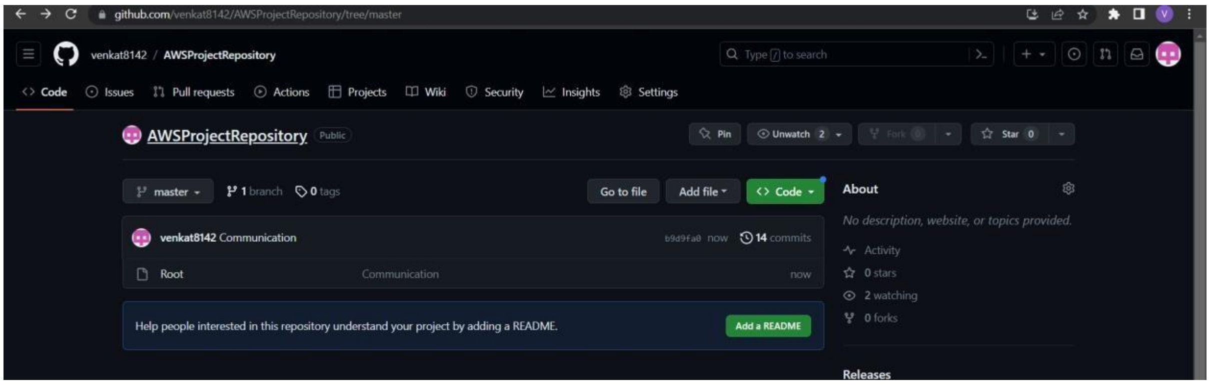
EC2 > Instances > i-0034d61619bcd5db9

Instance summary for i-0034d61619bcd5db9 (projectdeveloper2) Info
Updated less than a minute ago

Instance ID: i-0034d61619bcd5db9 (projectdeveloper2)
Public IPv4 address: 43.205.231.68 | open address
Private IPv4 addresses: 10.0.0.5
IPv6 address: –
Instance state: Running
Public IPv4 DNS: –
Hostname type: IP name: ip-10-0-0-5.ap-south-1.compute.internal
Private IP DNS name (IPv4 only): ip-10-0-0-5.ap-south-1.compute.internal
Answer private resource DNS name: –
Instance type: t2.micro
Elastic IP addresses: –
Auto-assigned IP address: 43.205.231.68 [Public IP]
VPC ID: vpc-03bda6b82396cb6c1 (ProjectVPC-1)
AWS Compute Optimizer finding: Opt-in to AWS Compute Optimizer for recommendations. | Learn more
IAM Role: –
Subnet ID: subnet-0b9227b5d5a3e15a6 (ProjectSubnet)
Auto Scaling Group name: –
IMDSv2: Optional

Creating a Github repository:

>Open Github account and create a repository for communication between the root user and developer.



github.com/venkat8142/AWSProjectRepository/tree/master

venkat8142 / AWSProjectRepository

Code | Issues | Pull requests | Actions | Projects | Wiki | Security | Insights | Settings

AWSProjectRepository Public

Pin | Unwatch 2 | Fork 0 | Star 0

master | 1 branch | 0 tags | Go to file | Add file | Code

venkat8142 Communication	b9d9fa0 now	14 commits
Root	Communication	now

Help people interested in this repository understand your project by adding a README. Add a README

About

No description, website, or topics provided.

Activity

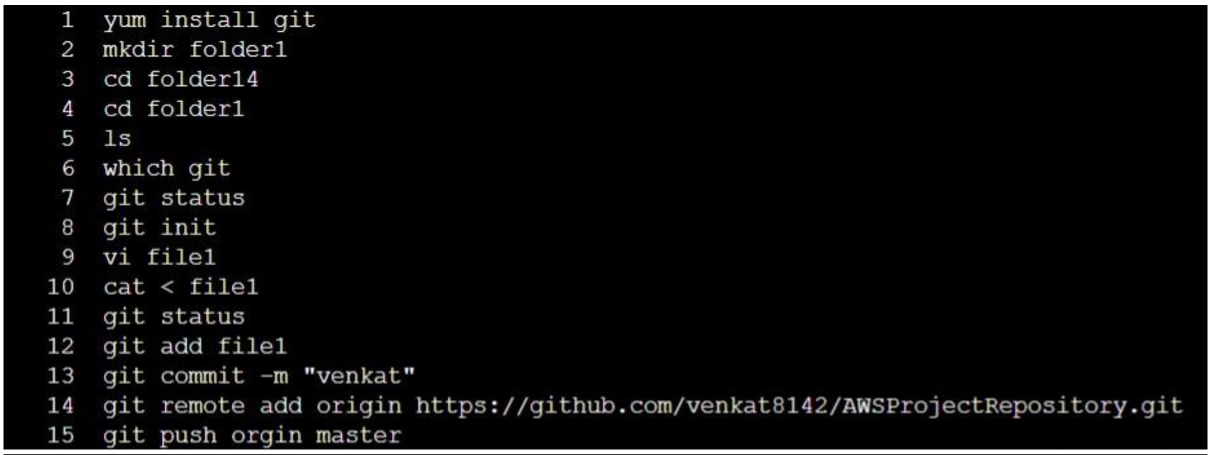
0 stars

2 watching

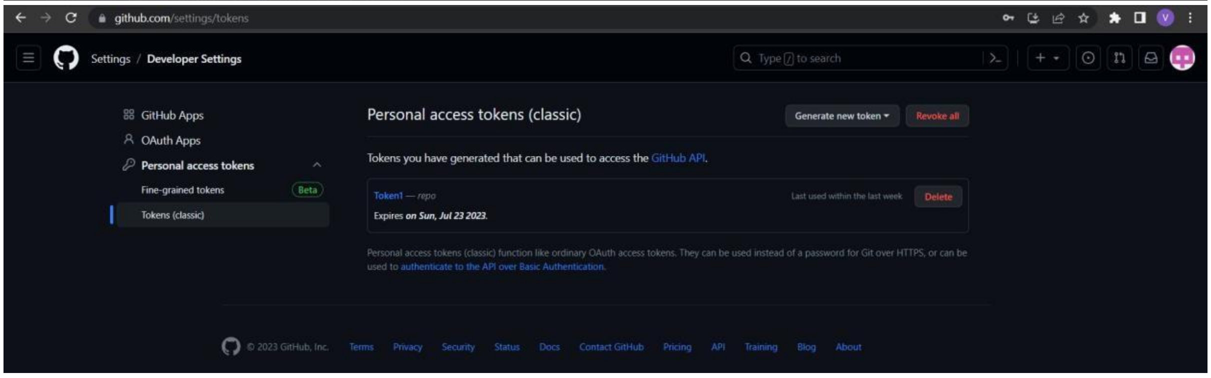
0 forks

Releases

>Connect the EC2 instance and enter the given commands to push a file to Git repository:



```
yum install git
mkdir folder1
cd folder14
cd folder1
ls
which git
git status
git init
vi file1
cat < file1
git status
git add file1
git commit -m "venkat"
git remote add origin https://github.com/venkat8142/AWSProjectRepository.git
git push orgin master
```



github.com/settings/tokens

Settings / Developer Settings

GitHub Apps

OAuth Apps

Personal access tokens

Fine-grained tokens Beta

Tokens (classic)

Personal access tokens (classic) | Generate new token | Revoke all

Tokens you have generated that can be used to access the GitHub API.

Token1 — *repo* | Last used within the last week | Delete

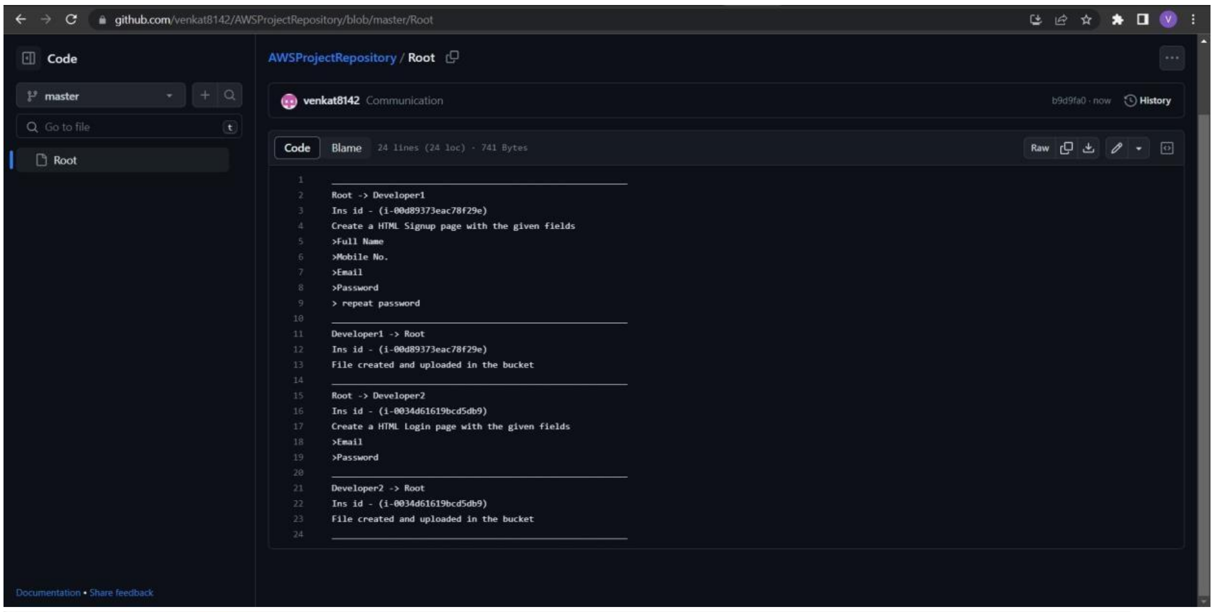
Expires on Sun, Jul 23 2023.

Personal access tokens (classic) function like ordinary OAuth access tokens. They can be used instead of a password for Git over HTTPS, or can be used to authenticate to the API over Basic Authentication.

© 2023 GitHub, Inc. Terms Privacy Security Status Docs Contact GitHub Pricing API Training Blog About

>For accesing the repository we need to generate a token and enter the token id in the command prompt instead of password.

>The IAM users will use the pull command for retrieving the information provided buy the root user.



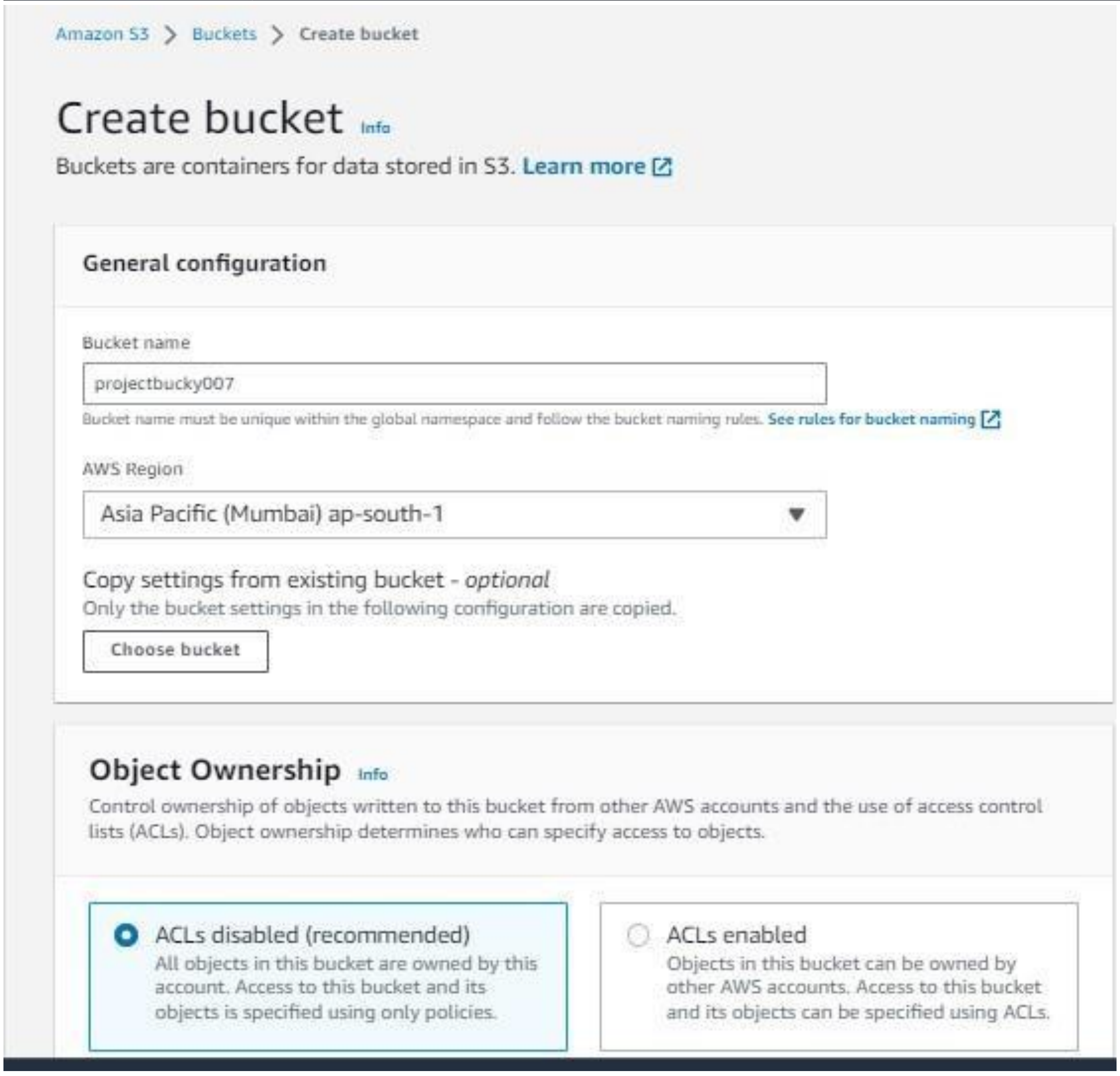
github.com/venkat8142/AWSProjectRepository/blob/master/Root

AWSProjectRepository / Root

venkat8142 Communication

24 lines (24 loc) · 741 Bytes

```
________________________________________
Root -> Developer1
Ins id - (i-00d89373eac78f29e)
Create a HTML Signup page with the given fields
>Full Name
>Mobile No.
>Email
>Password
> repeat password
________________________________________
Developer1 -> Root
Ins id - (i-00d89373eac78f29e)
File created and uploaded in the bucket
________________________________________
Root -> Developer2
Ins id - (i-0034d61619bcd5db9)
Create a HTML Login page with the given fields
>Email
>Password
________________________________________
Developer2 -> Root
Ins id - (i-0034d61619bcd5db9)
File created and uploaded in the bucket
________________________________________
```



Amazon S3 > Buckets > Create bucket

Create bucket Info

Buckets are containers for data stored in S3. Learn more

General configuration

Bucket name: projectbucky007

Bucket name must be unique within the global namespace and follow the bucket naming rules. See rules for bucket naming

AWS Region: Asia Pacific (Mumbai) ap-south-1

Copy settings from existing bucket - *optional*

Only the bucket settings in the following configuration are copied.

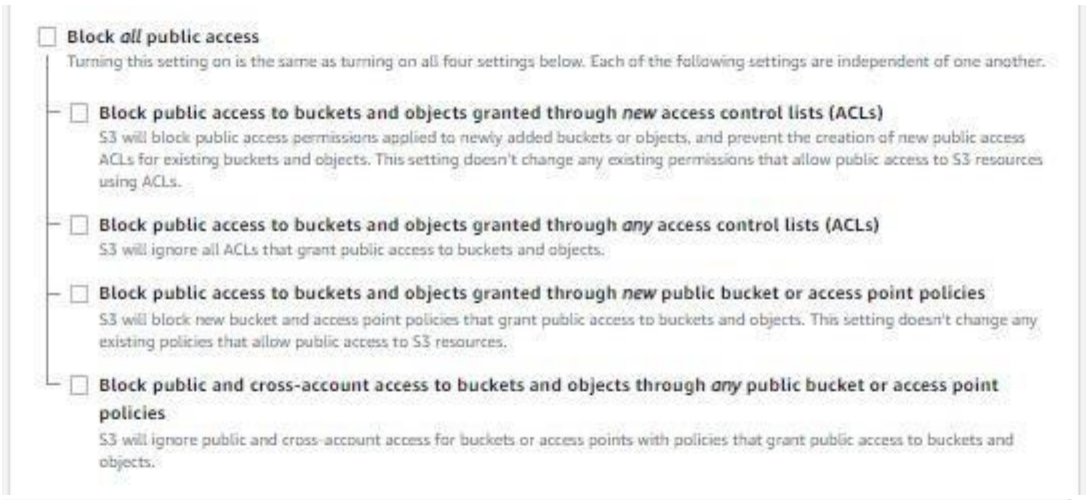
Object Ownership Info

Control ownership of objects written to this bucket from other AWS accounts and the use of access control lists (ACLs). Object ownership determines who can specify access to objects.

[✓ ACLs disabled (recommended)] All objects in this bucket are owned by this account. Access to this bucket and its objects is specified using only policies.

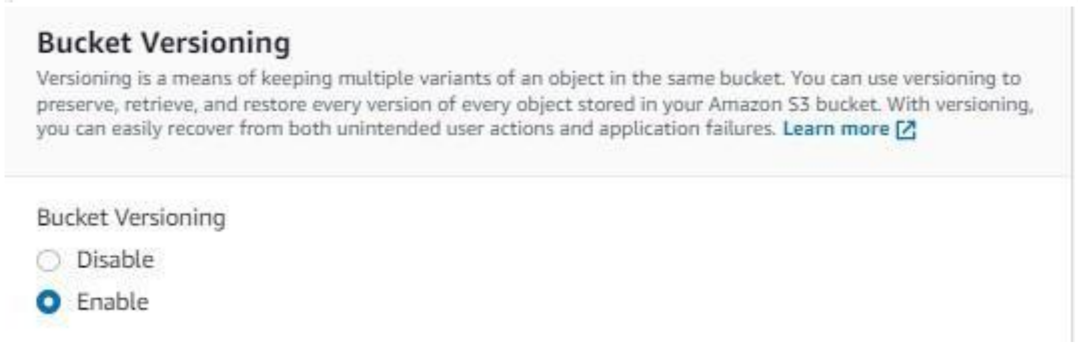
ACLs enabled: Objects in this bucket can be owned by other AWS accounts. Access to this bucket and its objects can be specified using ACLs.

r



☐ **Block *all* public access**
Turning this setting on is the same as turning on all four settings below. Each of the following settings are independent of one another.

- ☐ **Block public access to buckets and objects granted through *new* access control lists (ACLs)**
  S3 will block public access permissions applied to newly added buckets or objects, and prevent the creation of new public access ACLs for existing buckets and objects. This setting doesn't change any existing permissions that allow public access to S3 resources using ACLs.
- ☐ **Block public access to buckets and objects granted through *any* access control lists (ACLs)**
  S3 will ignore all ACLs that grant public access to buckets and objects.
- ☐ **Block public access to buckets and objects granted through *new* public bucket or access point policies**
  S3 will block new bucket and access point policies that grant public access to buckets and objects. This setting doesn't change any existing policies that allow public access to S3 resources.
- ☐ **Block public and cross-account access to buckets and objects through *any* public bucket or access point policies**
  S3 will ignore public and cross-account access for buckets or access points with policies that grant public access to buckets and objects.



Bucket Versioning
Versioning is a means of keeping multiple variants of an object in the same bucket. You can use versioning to preserve, retrieve, and restore every version of every object stored in your Amazon S3 bucket. With versioning, you can easily recover from both unintended user actions and application failures. Learn more

Bucket Versioning
Disable [✓ Enable]

Creating a S3 Bucket :

>Enter a Unique Bucket name.

>Enter the region for the Bucket.

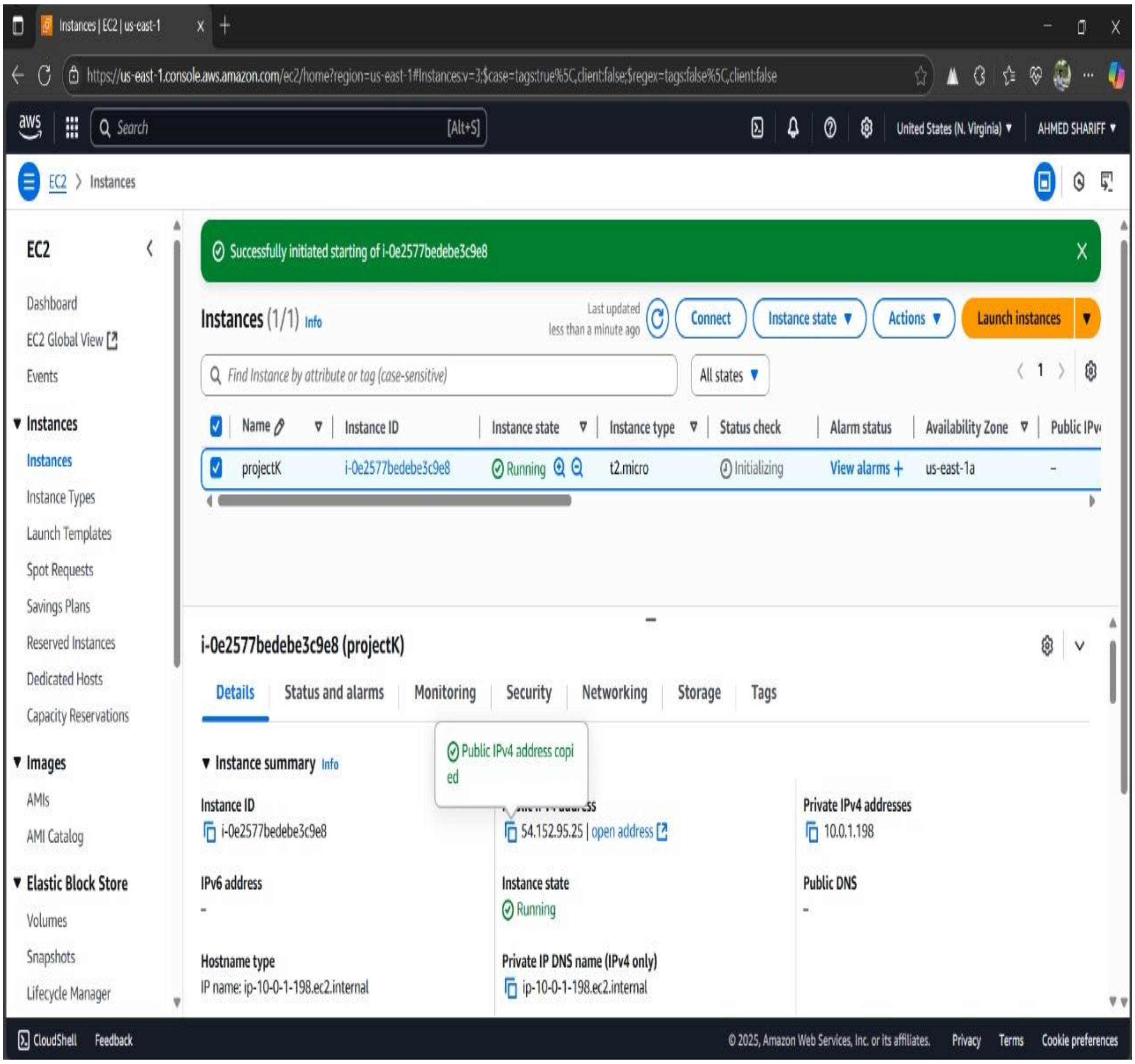
>Enable the ACL (Access Control List) option , Objects in this bucket can be owned by other AWS accounts. Access to this bucket and its objects can be specified using ACLs.

>Click the checkbox for block all public access.

>Enable Bucket versioning (Versioning is a means of keeping multiple variants of an object in the same bucket. You can use versioning to preserve, retrieve, and restore every version of every object stored in your Amazon S3 bucket. With versioning, you can easily recover from both unintended user actions and application failures.)

>Click the Create Bucket button.

➢ EC2 instance running with the public IPV4 address



Successfully initiated starting of i-0e2577bedebe3c9e8

Instances (1/1)

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv
projectK	i-0e2577bedebe3c9e8	Running	t2.micro	Initializing	View alarms +	us-east-1a	–

i-0e2577bedebe3c9e8 (projectK)

Instance summary

Public IPv4 address copied

Instance ID: i-0e2577bedebe3c9e8

Public IPv4 address: 54.152.95.25 | open address

Private IPv4 addresses: 10.0.1.198

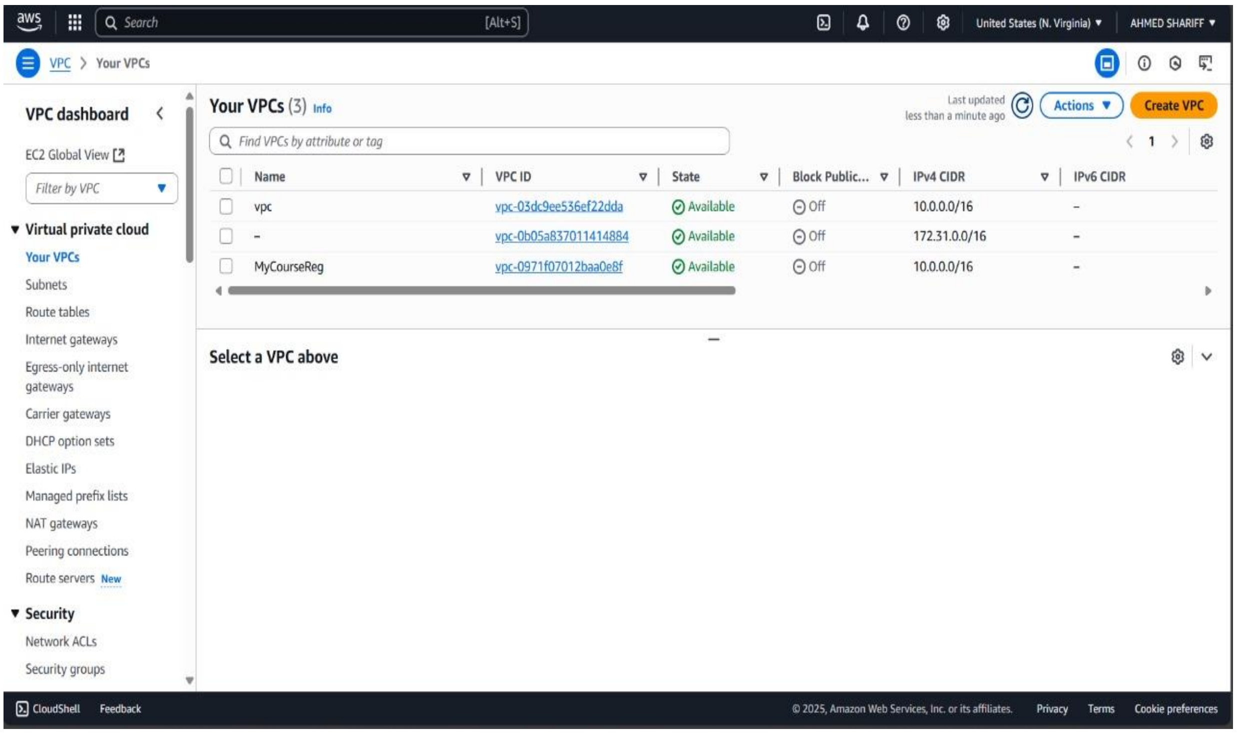
IPv6 address: –

Instance state: Running

Public DNS: –

Hostname type: IP name: ip-10-0-1-198.ec2.internal

Private IP DNS name (IPv4 only): ip-10-0-1-198.ec2.internal



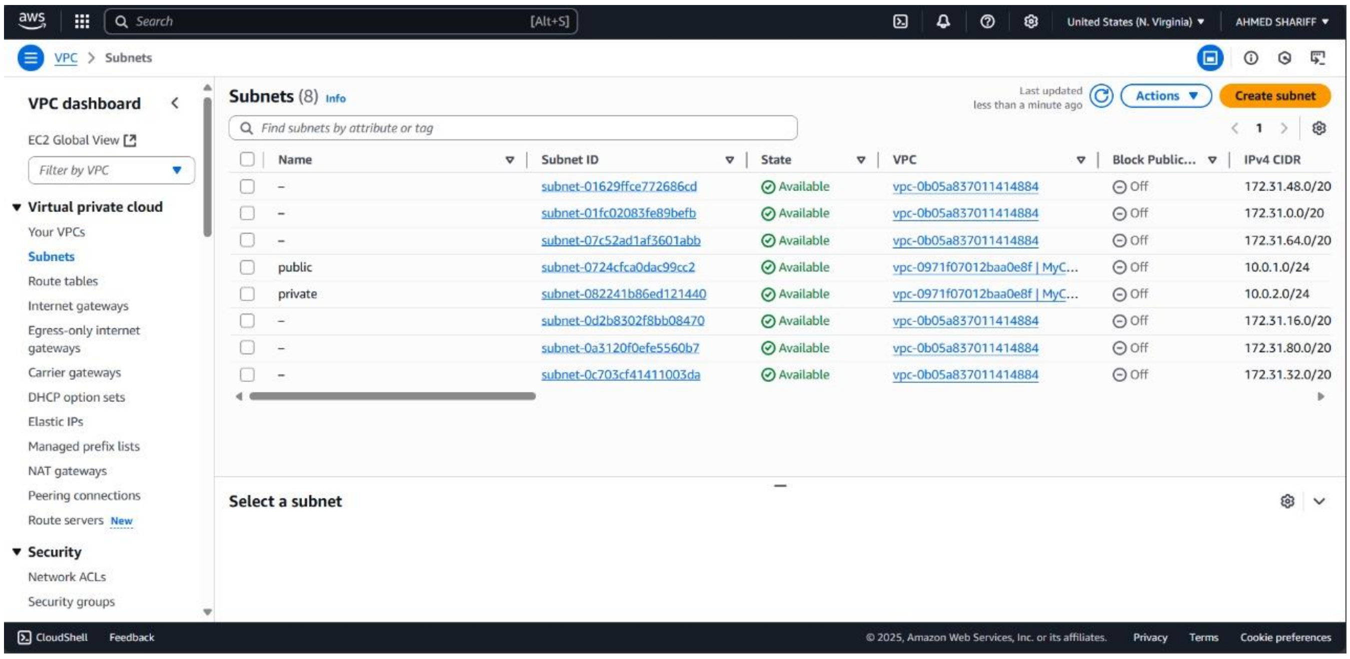
VPC > Your VPCs

Your VPCs (3)

Name	VPC ID	State	Block Public...	IPv4 CIDR	IPv6 CIDR
vpc	vpc-03dc9ee536ef22dda	Available	Off	10.0.0.0/16	–
–	vpc-0b05a837011414884	Available	Off	172.31.0.0/16	–
MyCourseReg	vpc-0971f07012baa0e8f	Available	Off	10.0.0.0/16	–

Select a VPC above

- Available subnets and their details in AWS VPC dashboard

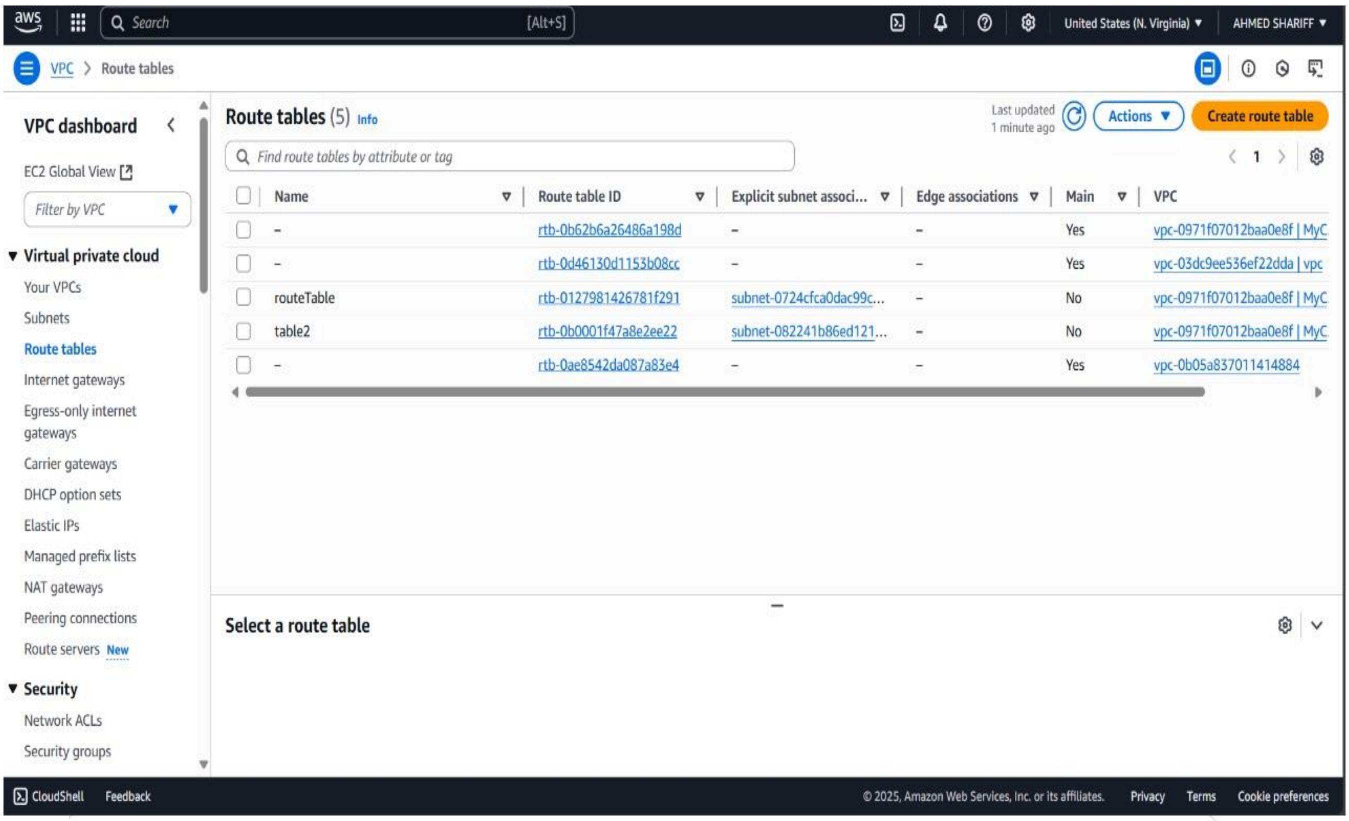


VPC > Subnets

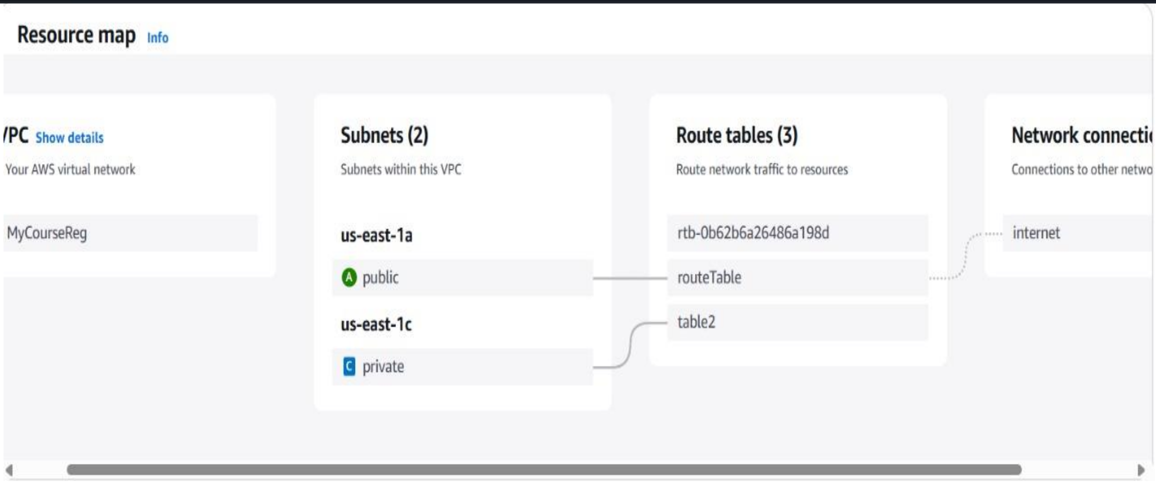
Subnets (8)

Name	Subnet ID	State	VPC	Block Public...	IPv4 CIDR
–	subnet-01629ffce772686cd	Available	vpc-0b05a837011414884	Off	172.31.48.0/20
–	subnet-01fc02083fe89befb	Available	vpc-0b05a837011414884	Off	172.31.0.0/20
–	subnet-07c52ad1af3601abb	Available	vpc-0b05a837011414884	Off	172.31.64.0/20
public	subnet-0724cfca0dac99cc2	Available	vpc-0971f07012baa0e8f | MyC...	Off	10.0.1.0/24
private	subnet-082241b86ed121440	Available	vpc-0971f07012baa0e8f | MyC...	Off	10.0.2.0/24
–	subnet-0d2b8302f8bb08470	Available	vpc-0b05a837011414884	Off	172.31.16.0/20
–	subnet-0a3120f0efe5560b7	Available	vpc-0b05a837011414884	Off	172.31.80.0/20
–	subnet-0c703cf41411003da	Available	vpc-0b05a837011414884	Off	172.31.32.0/20

Select a subnet

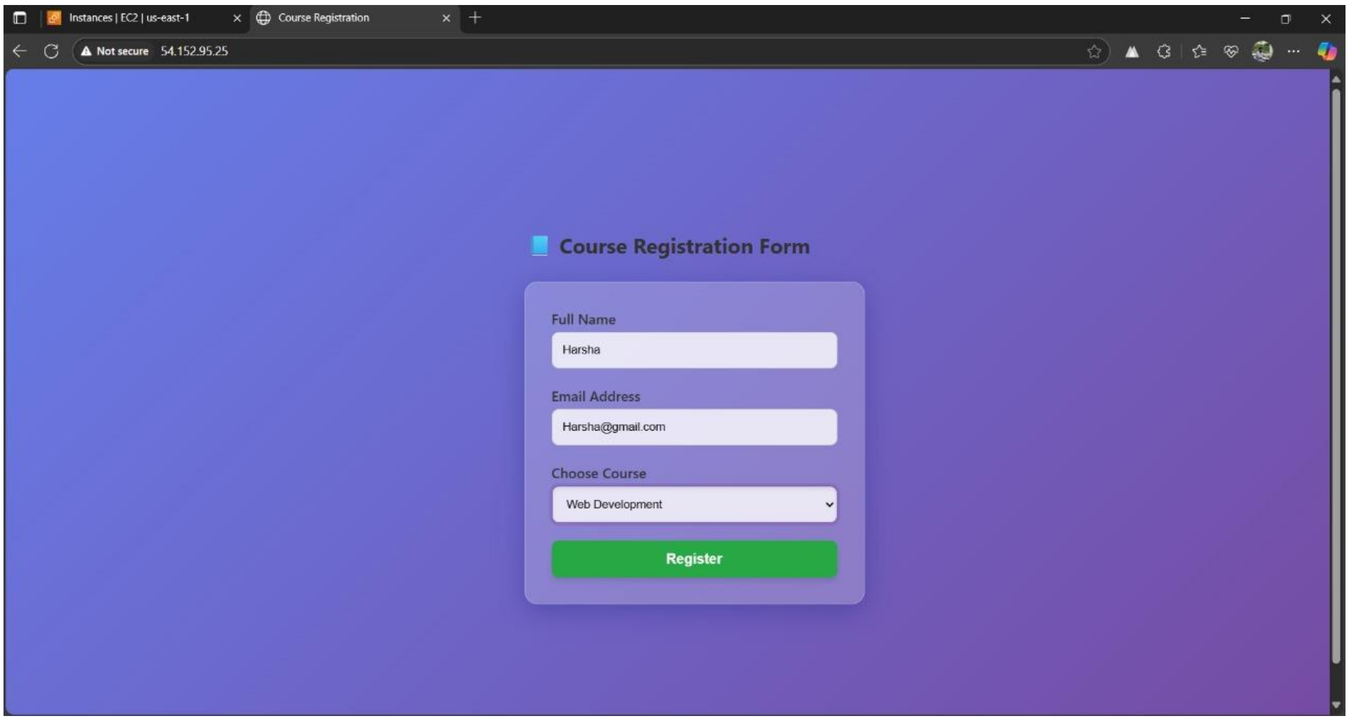


Name	Route table ID	Explicit subnet associ...	Edge associations	Main	VPC
–	rtb-0b62b6a26486a198d	–	–	Yes	vpc-0971f07012baa0e8f | MyC
–	rtb-0d46130d1153b08cc	–	–	Yes	vpc-03dc9ee536ef22dda | vpc
routeTable	rtb-0127981426781f291	subnet-0724cfca0dac99c...	–	No	vpc-0971f07012baa0e8f | MyC
table2	rtb-0b0001f47a8e2ee22	subnet-082241b86ed121...	–	No	vpc-0971f07012baa0e8f | MyC
–	rtb-0ae8542da087a83e4	–	–	Yes	vpc-0b05a837011414884



AWS VPC resource map with one VPC with two subnets

- Web Based Course Registration Form



Course Registration Form

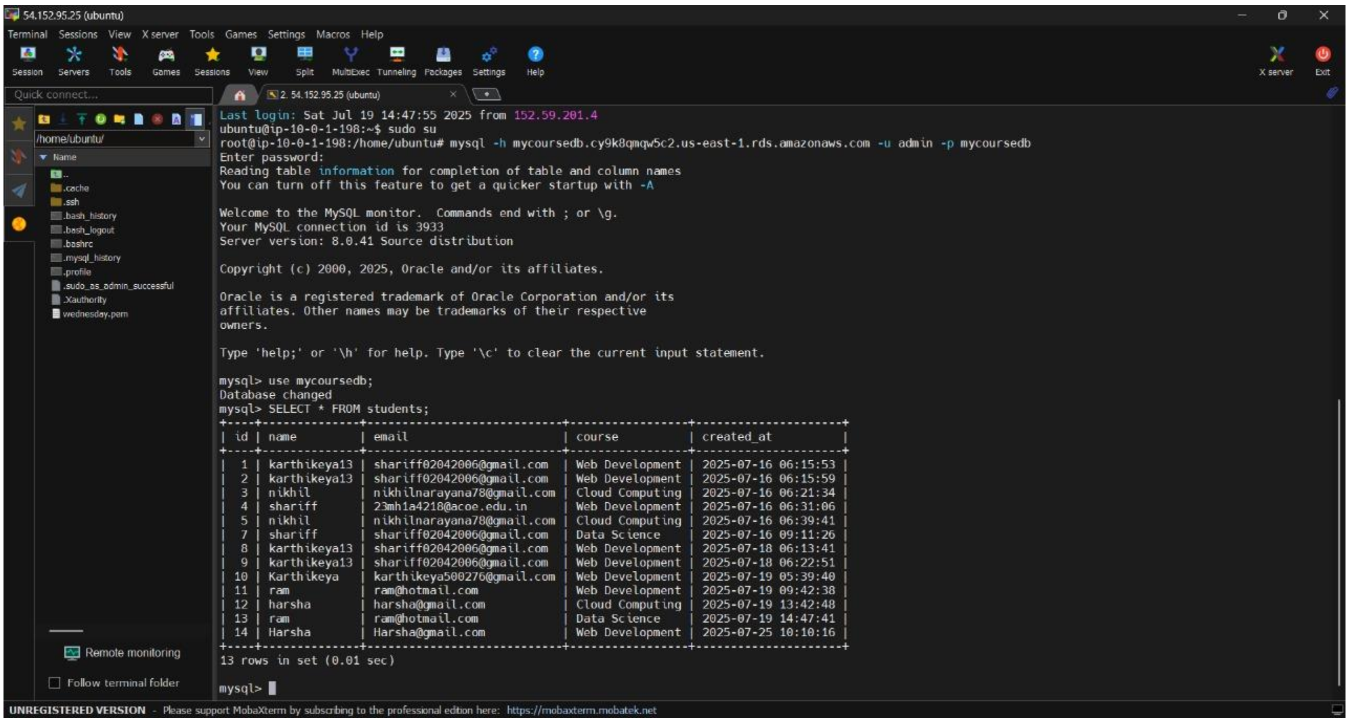
Full Name: Harsha

Email Address: Harsha@gmail.com

Choose Course: Web Development

Register

- **MYSQL** database hosted on Amazon RDS


54.152.95.25 (ubuntu)
Terminal Sessions View X server Tools Games Settings Macros Help
Session Servers Tools Games Sessions View Split MultiExec Tunneling Packages Settings Help
X server Exit
2. 54.152.95.25 (ubuntu)
/home/ubuntu/
Remote monitoring
Follow terminal folder

```
Last login: Sat Jul 19 14:47:55 2025 from 152.59.201.4
ubuntu@ip-10-0-1-198:~$ sudo su
root@ip-10-0-1-198:/home/ubuntu# mysql -h mycoursedb.cy9k8qmqw5c2.us-east-1.rds.amazonaws.com -u admin -p mycoursedb
Enter password:
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 3933
Server version: 8.0.41 Source distribution

Copyright (c) 2000, 2025, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> use mycoursedb;
Database changed
mysql> SELECT * FROM students;
+----+-------------+----------------------------+-----------------+---------------------+
| id | name        | email                      | course          | created_at          |
+----+-------------+----------------------------+-----------------+---------------------+
|  1 | karthikeya13 | shariff02042006@gmail.com  | Web Development | 2025-07-16 06:15:53 |
|  2 | karthikeya13 | shariff02042006@gmail.com  | Web Development | 2025-07-16 06:15:59 |
|  3 | nikhil      | nikhilnarayana78@gmail.com | Cloud Computing | 2025-07-16 06:21:34 |
|  4 | shariff     | 23mh1a4218@acoe.edu.in     | Web Development | 2025-07-16 06:31:06 |
|  5 | nikhil      | nikhilnarayana78@gmail.com | Cloud Computing | 2025-07-16 06:39:41 |
|  7 | shariff     | shariff02042006@gmail.com  | Data Science    | 2025-07-16 09:11:26 |
|  8 | karthikeya13 | shariff02042006@gmail.com  | Web Development | 2025-07-18 06:13:41 |
|  9 | karthikeya13 | shariff02042006@gmail.com  | Web Development | 2025-07-18 06:22:51 |
| 10 | Karthikeya  | karthikeya500276@gmail.com | Web Development | 2025-07-19 05:39:40 |
| 11 | ram         | ram@hotmail.com            | Web Development | 2025-07-19 09:42:38 |
| 12 | harsha      | harsha@gmail.com           | Cloud Computing | 2025-07-19 13:42:48 |
| 13 | ram         | ram@hotmail.com            | Data Science    | 2025-07-19 14:47:41 |
| 14 | Harsha      | Harsha@gmail.com           | Web Development | 2025-07-25 10:10:16 |
+----+-------------+----------------------------+-----------------+---------------------+
13 rows in set (0.01 sec)

mysql> 
```

UNREGISTERED VERSION - Please support MobaXterm by subscribing to the professional edition here: https://mobaxterm.mobatek.net