



BLACKBUCKS INTERNSHIP REPORT

**An Architecture of Bastion Host with
Auto Scaling**

SUBMITTED BY

Ms. SATHI JYOTHSNA MALIKA

RegdNo:20B91A5756

Ms. PINNAMANENI SREEJA

RegdNo:20B91A5752

Ms. KUNCHE SOWMYA SREE

RegdNo:20B91A5735

UNDER THE GUIDANCE OF MR. AASHU DEV

B.Tech, AWS solution Architect (2x) certified,

AWS Academy Accredited Educator



**Blackbuck Engineers Pvt, Ltd
Road No 36, Jubilee Hills, Hyderabad**

BLACKBUCK INTERNSHIP WORK

Team Members:

- SATHI JYOTHSNA MALIKA (20B91A5756)
- PINNAMANENI SREEJA (20B91A5752)
- KUNCHE SOWMYA SREE(20B91A5735)

Title:

AWS architecture of Bastion Host with Auto Scaling

Abstract:

The above architecture is to access EC2 private instances through EC2 public instances by using bastion host that provides security to a network from external sources and creating an Auto Scaling group which helps in creating instances to meet desired capacity. It maintains a few instances by performing periodic health checks on the instances in the group.

Table of Contents

1.Introduction to Amazon web services (AWS)	1-2
2. Why AWS	2-5
3. AWS global Infrastructure	5-8
4. List of top AWS services	8-18
5. Implementation of the Architecture of Bastion Host with Auto Scaling	19
5.1. Introduction	19-20
5.2.AWS services used	20
5.3. Rough Architecture	20
5.4. Final Architecture	21
5.5. Workflow	21-22
6. Implementation of the project	22
6.1. Service 1: VPC	22-27
6.2. Service 2: EC2	27-30
6.3. Service 3: AMI	30-32
6.4. Service 4: IAM User	33-38
6.5. Service 5: Cloud Shell	39-42
6.7. Creating an auto scaling group	43-53
6.8. Conclusion	54

Introduction To Amazon Web Services (AWS):

- **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, compute, 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 in 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.

Uses of AWS:

- A small manufacturing organization uses their expertise to expand their business by leaving their IT management to the AWS.
- A large enterprise spread across the globe can utilize the AWS to deliver the training to the distributed workforce.
- An architecture consulting company can use AWS to get the high compute rendering of construction prototype.
- A media company can use the AWS to provide different types of content such as ebox or audio files to the worldwide files.

Why AWS?

There are several reasons why AWS has become a popular choice for cloud computing:

1. **Broad and Comprehensive Service Offering:** AWS offers a wide range of services to meet various computing needs. Whether you require computer power, storage, databases, machine learning, analytics, networking, or other capabilities, AWS provides a comprehensive set of services to fulfill these requirements.
2. **Scalability and Flexibility:** AWS allows users to scale their resources up or down based on demand. Whether you need to handle a sudden surge in traffic or want to reduce costs during

periods of lower activity, AWS provides the flexibility to adjust your resources accordingly. This scalability ensures that your applications can handle varying workloads effectively.

3. Global Infrastructure: AWS has a vast global infrastructure comprising numerous data centers and availability zones spread across different regions. This infrastructure enables users to deploy their applications and services closer to their target audience, resulting in reduced latency and improved performance.

4. Reliability and Availability: AWS has built a reputation for providing highly reliable and available services. With its multiple availability zones and data replication mechanisms, AWS ensures that your applications and data remain accessible even in the face of hardware failures or natural disasters. Service Level Agreements (SLAs) guarantee a certain level of uptime for many AWS services.

5. Security: AWS places a strong emphasis on security. It provides a wide range of security features and tools to help users protect their applications and data. This includes encryption options, network security controls, identity and access management, and compliance certifications. AWS adheres to industry best practices to maintain a secure environment for its customers.

6. Integration and Ecosystem: AWS integrates well with various third-party tools, technologies, and services. It offers extensive APIs and SDKs, making it easier to integrate AWS services into existing applications or build new solutions from scratch. The AWS ecosystem also includes a vibrant community, documentation, training resources, and support services, facilitating development and troubleshooting.

7. Cost-Effectiveness: AWS follows a pay-as-you-go pricing model, allowing users to pay only for the resources they consume. This eliminates the need for upfront investments in hardware and infrastructure. Additionally, AWS provides cost optimization tools and features to help users monitor and control their spending, ensuring cost-effectiveness.

8. Innovation and Continuous Improvement: AWS continues to innovate and expand its services, introducing new capabilities and features regularly. It invests heavily in research and development to stay at the forefront of cloud technology. This commitment to innovation ensures that users have access to the latest tools and advancements in cloud computing.

These factors, among others, contribute to the popularity and success of AWS as a cloud computing provider. However, it's important to note that the choice of cloud provider should be based on your specific needs, requirements, and preferences. It's worth evaluating multiple cloud platforms to determine the best fit for your organization.

Advantages of AWS:

1. Flexibility
2. Cost-effectiveness
3. Scalability/Elasticity
4. Security

1) Flexibility

- We can get more time for core business tasks due to the instant availability of new features and services in AWS.
- It provides effortless hosting of legacy applications. AWS does not require learning new technologies and migration of applications to the AWS provides advanced computing and efficient storage.
- AWS also offers a choice that whether we want to run the applications and services together or not. We can also choose to run a part of the IT infrastructure in AWS and the remaining part in data centers.

2) Cost-effectiveness

AWS requires no upfront investment, long-term commitment, and minimum expense when compared to traditional IT infrastructure that requires a huge investment.

3) Scalability/Elasticity

Through AWS, autoscaling and elastic load balancing techniques are automatically scaled up or down, when demand increases or decreases respectively. AWS techniques are ideal for handling

unpredictable or very high loads. Due to this reason, organizations enjoy the benefits of reduced cost and increased user satisfaction.

4) Security

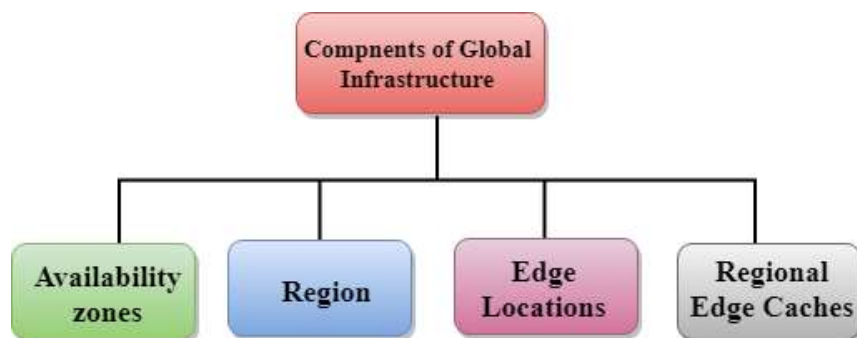
- AWS provides end-to-end security and privacy to customers.
- AWS has a virtual infrastructure that offers optimum availability while managing full privacy and isolation of their operations.
- Customers can expect high-level physical security because of Amazon's several years of experience in designing, developing, and maintaining large-scale IT operation centers.
- AWS ensures the three aspects of security, i.e., Confidentiality, integrity, and availability of users.

AWS Global Infrastructure

- AWS is a cloud computing platform which is globally available.
- Global infrastructure is a region around the world in which AWS is based. Global infrastructure is a bunch of high-level IT services which is shown below:
- AWS is available in 19 regions, and 57 availability zones in December 2018 and 5 more regions 15 more availability zones for 2019.

The following are the components that make up the AWS infrastructure:

- Availability Zones
- Region
- Edge locations
- Regional Edge Caches



In Amazon Web Services (AWS), an Availability Zone (AZ) refers to a distinct, physically separate data center within a specific region. AZs are designed to provide fault tolerance and high availability by isolating failures and minimizing the impact of any disruption.

Each AZ is equipped with independent power, cooling, networking infrastructure, and is connected to other AZs within the same region through high-speed, low-latency links. They are strategically located to minimize the risk of natural disasters affecting multiple zones simultaneously.

By distributing resources across multiple AZs, you can design highly reliable and resilient architectures in AWS. When you launch resources like EC2 instances, databases, or storage volumes, you have the option to select the AZ in which they should be provisioned.

The primary benefits of utilizing Availability Zones in AWS include:

Fault tolerance: By deploying resources in different AZs, you protect your applications from single points of failure. If one AZ experiences an issue, your applications can continue running in other AZs, ensuring minimal downtime.

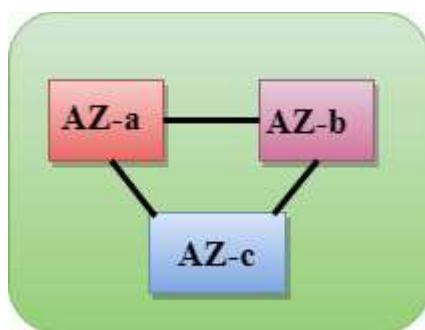
High availability: Distributing resources across AZs allows you to achieve high availability for your applications. Services like load balancers can be configured to route traffic across multiple AZs, automatically diverting traffic to healthy instances if one AZ becomes unavailable.

Availability zone as a Data Center

- An availability zone is a facility that can be somewhere in a country or in a city. Inside this facility, i.e., Data Centre, we can have multiple servers, switches, load balancing, firewalls. The things which interact with the cloud sit inside the data centers.
- An availability zone can be a several data centers, but if they are close together, they are counted as 1 availability zone.

Region

- A region is a geographical area. Each region consists of 2 more availability zones.
- A region is a collection of data centers which are completely isolated from other regions.
- A region consists of more than two availability zones connected to each other through links.



- Availability zones are connected through redundant and isolated metro fibers.

Edge Locations

- Edge locations are the endpoints for AWS used for caching content.
- Edge locations consist of CloudFront and Amazon's Content Delivery Network (CDN).
- Edge locations are more than regions. Currently, there are over 150 edge locations.
- Edge location is not a region but a small location that AWS have. It is used for caching the content.
- Edge locations are mainly located in most of the major cities to distribute the content to end users with reduced latency.
- For example, some user accesses your website from Singapore; then this request would be redirected to the edge location closest to Singapore where cached data can be read.

Regional Edge Cache

- AWS announced a new type of edge location in November 2016, known as a Regional Edge Cache.
- Regional Edge cache lies between CloudFront Origin servers and the edge locations.
- A regional edge cache has a larger cache than an individual edge location.
- Data is removed from the cache at the edge location while the data is retained at the Regional Edge Caches.
- When the user requests the data, then data is no longer available at the edge location. Therefore, the edge location retrieves the cached data from the regional edge cache instead of the Origin servers that have high latency.

List of top AWS Services:

AWS is the widely used cloud platform worldwide, from start-ups to large enterprises. Though AWS services were introduced to the market by 2006, their revenue from Public Cloud SaaS has hit 145.5 billion USD by 2021. Presently, Amazon Web Services are a one-stop solution for all cloud services ranging from data storage to analytics. AWS services provide easy, simple, cost-effective cloud services, which drive businesses to achieve increased efficiency and performance. Besides, these services have many more features to serve customers in multiple ways.

Now, let's have a look at the most popular AWS services in 2023. In this blog, you can learn what is the objective, features, and benefits of each AWS service.

Here is the list of Top 30 AWS Services List:

1. Amazon EC2 [Elastic Compute Cloud]

Amazon EC2 is one of the fastest-growing cloud computing AWS services, which offers virtual servers to manage any kind of workload. It facilitates the computing infrastructure with the best suitable processors, networking facilities, and storage systems. As a result, it supports adapting to the workloads precisely. Amazon EC2 provides a highly secure, reliable, performing computing infrastructure meeting business demands. And it helps you to access resources quickly and dynamically scale capacities as per demands.



2. Amazon S3

Another popular addition to the AWS services list is Amazon S3, which is an object storage AWS service, which is highly scalable. It mainly helps users to access any quantity of data from anywhere. Here, data is stored in 'storage classes' to reduce costs without any extra investment and manage it comfortably. The data is highly secure and supports meeting audit and compliance requirements. You can handle any volume of data with Amazon S3's robust access controls, replication tools, and higher visibility. Moreover, it supports maintaining data version controls and preventing accidental deletion.



3. AWS Aurora

Amazon Aurora is the next addition to this list of top AWS services in demand. Why? It is a MySQL and PostgreSQL compatible relational database with high performance. Believe it or not, it is five times faster than standard MySQL databases. And it allows for automating crucial tasks such as hardware provisioning, database setup and backups, and patching. Amazon Aurora is a distributed, fault-tolerant, self-healing storage system that could scale automatically as per needs. Besides, you can even reduce costs significantly and enhance databases' security, availability, and reliability.



4. Amazon DynamoDB

DynamoDB is a promising addition to this list of AWS services. DynamoDB is a fully managed and serverless NoSQL database AWS service. And it is a fast and flexible database system that provides innovative opportunities to developers at low costs. It gives you single-digit millisecond performance with unlimited throughput and storage. DynamoDB has in-built tools to generate actionable insights, useful analytics, and monitor traffic trends in applications.



5. Amazon RDS

Amazon RDS would be the next entry in this discussion on AWS services. Amazon RDS is the managed Relational Database AWS Service (RDS) for MySQL, PostgreSQL, Oracle, SQL Server, and MariaDB. It allows the setup, operation, and scale of a relational database in the cloud quickly. Also, it achieves high performance by automating the tasks such as hardware provisioning, database setup, patching, and backups. When you use Amazon RDS, you don't need to install and maintain the database software. Overall, you can optimize costs by embracing this service and achieve high availability, security, and compatibility for your resources.



6. Amazon Lambda

AWS Lambda is also a promising addition to the list of AWS services. Amazon Lambda is a serverless and event-driven computing AWS service. It helps to run codes automatically without worrying about servers and clusters. Simply put, codes can be uploaded directly to run without worrying about provisioning or managing infrastructure. So, this service automatically accepts 'code execution requests' irrespective of its scale. Besides, you can pay the price only for the computer time, so AWS Lambda makes effective cost-control.



7. Amazon VPC

Amazon VPC is the Virtual Private Cloud, which is an isolated cloud resource. It controls the virtual networking environment, such as resource placement, connectivity, and security. And it allows you to build and manage compatible VPC networks across cloud AWS resources and on-premise resources. Here, it improves security by applying rules for inbound and outbound connections. Also, it monitors VPC flow logs delivered to Amazon S3 and Amazon Cloudwatch to gain visibility over network dependencies and traffic patterns. Amazon VPC also detects anomalies in the patterns, prevents data leakage, and troubleshoots network connectivity and configuration issues.



8. Amazon CloudFront

Amazon CloudFront is another credible mention in the list of renowned Amazon Web Services. This AWS service delivers content globally, which offers high performance and security. Mainly, it delivers data with high speed and low latency. Here, content is delivered to destinations successfully with automated network mapping and intelligent routing mechanisms. The security of data is enhanced with traffic encryption methods and access controls. Also, data can be transferred within milliseconds with its in-built data compression, edge computing capabilities, and field-level encryption. Besides, you gear up streaming high-quality video using AWS media services to any device quickly and consistently using Amazon CloudFront.



9. AWS Elastic Beanstalk

This AWS service supports running and managing web applications. Elastic Beanstalk allows for the easy deployment of applications from capacity provisioning, load balancing, and auto-scaling to application health monitoring. With its auto-scaling properties, this service simplifies demands in scaling to adjust to the needs of the business. It helps to manage peaks in workloads and traffic with minimum costs. Basically, AWS Elastic Beanstalk is a developer-friendly tool since it manages servers, load balancers, firewalls, and networks simply. As a result, this service allows developers to show much more focus on coding.



10. Amazon EC2 Auto-scaling

This AWS service scales computing capacity to meet the demands accurately. And it is achieved by adding or removing EC2 instances automatically. There are two types of scaling such as dynamic scaling and predictive scaling. Here, dynamic scaling responds to the presently changing demands, whereas predictive scaling responds based on predictions. Through Amazon EC2 Auto-scaling, you can identify the unhealthy EC2 instances, terminate them, and replace them with new instances.



11. Amazon ElastiCache

Amazon ElastiCache is a fully managed, flexible, in-memory caching AWS service. It supports increasing the performance of your applications and database. And this service helps to reduce the load in a database by caching data in memory. Amazon ElastiCache accesses data from in-memory with high speed, microsecond latency, and high throughput. With a self-managed cache service, you can reduce costs and eliminate the operational overhead of your business.



12. Amazon S3 Glacier

Amazon S3 Glacier is the archive storage in the cloud at a low cost. It is built with three storage classes such as S3 Glacier instant retrieval, flexible retrieval, and deep archive. Here, the instant class supports immediate access to data, and the flexible class allows flexible access within minutes to hours with no cost. The third one, deep archive, helps archive compliance data and digital media. Overall, they support you to access data from archives faster.



13. Amazon LightSail

Amazon LightSail is the website and applications building AWS service. This service offers Virtual Private Server instances, containers, databases, and storage. It allows a serverless computing service with AWS Lambda. With Amazon LightSail, you can create websites using pre-configured applications such as WordPress, Magento, Prestashop, and Joomla in a few clicks and at a low cost. In addition to this, it is the best tool for testing, so you can create, test, and delete sandboxes with your new ideas.



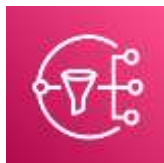
14. Amazon SageMaker

Amazon SageMaker is the AWS service that allows building, training, and deploying Machine Learning (ML) models at a large capacity. It is an analytical tool that functions based on Machine Learning power to analyze data more efficiently. With its single toolset, you can build high-quality ML models quickly. Amazon SageMaker not only generates reports but provides the purpose for generating predictions too. In addition, Amazon Ground Truth Plus creates datasets without labeling applications.



15. Amazon SNS

It is the Amazon Simple Notification Service (SNS). It is a messaging service between Application to Application (A2P) and Application to Person (A2Person). Here, A2P helps many-to-many messaging between distributed systems, microservices, and event-driven serverless applications. And A2P supports applications to send messages to many users via mail, SMS, etc. For instance, you can send up to ten messages in a single API request. With effective filtering systems, subscribers will receive messages that they are interested in. Besides, Amazon SNS works alongside Amazon SQS to deliver messages accurately and consistently.



16. Amazon EBS

Amazon Elastic Block Store (EBS) is a block storage service. It supports scaling high-performance workloads such as SAP, Oracle, and Microsoft products. And it provides better

protection against failures up to 99.999%. It helps to resize clusters for big data analytics engines such as Hadoop and Spark. Also, you can build storage volumes, optimize storage performance, and reduce costs. Amazon EBS's lifecycle management creates policies that help create and manage backups effectively.



17. Amazon Kinesis

It is the AWS service that analyses video as well as data streams. Amazon Kinesis collects, processes, and analyzes all types of streaming data. Here, the data may be audio, video, application logs, website clickstreams, and IoT telemetry. Then, it generates real-time insights within seconds once the data has arrived. With the help of Amazon Kinesis, you could stream and process a large quantity of real-time data with low latencies, very simply.



18. Amazon Elastic File System (EFS)

Amazon EFS is the fully managed file system for Amazon EC2. And it is a simple and serverless elastic file system. You can create and configure file systems without provisioning, deploying, patching, and maintenance using Amazon EFS. Here, files can be added and deleted as per the scaling needs. Especially, you can pay only for the used space, hence this service helps to reduce costs.



19. AWS IAM

It is the Identity and Access Management (IAM) service offered by AWS to securely access the applications and resources. It regulates access to various resources based on roles and access policies; as a result, you can achieve a fine-grained access control on your resources. The AWS IAM access analyzer helps streamline permission management through setting, verifying, and refining. In addition, AWS IAM attribute-based access control helps create fine-grained permissions based on user attributes such as department, job role, team name, etc.



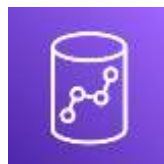
20. Amazon SQS

Amazon SQS is a fully managed message queuing service. There are two types of message queuing services: SQS Standard and SQS FIFO. Here, the SQS standard offers features such as maximum throughput, best-effort ordering, and quick delivery. And SQS FIFO processes messages only once in the same order by which they have been sent. Also, Amazon SQS allows decoupling or scaling microservices, distributed systems, and serverless applications. It helps you send, receive, and manage messages in a large volume. Moreover, there is no need to install and maintain other messaging software, reducing costs significantly. Besides, scaling is carried out quickly and automatically in this service.



21. Amazon RedShift

Amazon Redshift is a quick, simple, and cost-effective data warehousing service. You can gain insights about cloud data warehousing in an easy, faster, and more secure way. It allows analysis of all the data in operational databases, data lakes, data warehouses, and third-party data. And Amazon Redshift helps analyze a large volume of data and run complex analytical queries. With its automation capabilities, this service increases query speed and provides the best price performance.



22. Amazon Cloudwatch

This AWS service monitors the cloud resources and applications keenly. It is a single platform that helps to monitor all AWS resources and applications; it increases visibility to respond to issues quickly. Mainly, Amazon Cloudwatch provides actionable insights to optimize monitoring applications, systemwide performance changes, and resource utilization. And you can get a complete view of the health of AWS resources, applications, and services running on AWS and on-premises. In addition, Amazon CloudWatch helps to detect anomalies in the behavior of the cloud environment, set alarms, visualize logs and metrics, make automated actions, troubleshoot issues, and discover insights.



23. Amazon Chime

Amazon Chime is a communication service. It is a single solution that offers audio calling, video calling, and screen sharing capabilities. With the help of this service, you can make quality meetings, chat, and video calls both inside and outside of your organization. And more features can be added to this service as per your business needs. Mainly, you can set calls for a pre-defined time to automatically make calls on time. Amazon Chime helps you not to miss a meeting amidst your hectic schedule at work. Besides, you can pay as per the usage of resources by which you can reduce the costs significantly.



24. Amazon Cognito

It is the identity management AWS service. Amazon Cognito manages identities for accessing your applications and resources. Mainly, this service helps to add sign-in, sign-up, and access control the web and mobile apps quickly. It can support millions of users to sign in with familiar applications such as Apple, Facebook, Google, and Amazon. In Amazon Cognito, the feature 'Cognito user pools' can be set up quickly without any infrastructure, and the pool members will have a directory profile. It supports multi-factor authentication and encryption of data-at-rest and data-in-transit.



25. Amazon Inspector

Amazon Inspector is an automated vulnerability management service. This service offers continuous and automated vulnerability management for Amazon EC2 and Amazon ECR. It allows scanning AWS workloads for software vulnerabilities and unwanted network exposure. Amazon Inspector quickly identifies vulnerabilities, which helps to take immediate actions to resolve them before it worsens the applications. Moreover, it supports meeting compliance requirements and reduces meantime-to-remediate vulnerabilities. And it provides you with accurate risk scores and streamlined workflow.



26. AWS Firewall Manager

It is the central management service of firewall rules. The firewall manager supports managing firewall rules across all the applications and accounts. The common security rules help to manage new applications included over time. It is the one-time solution for consistently creating firewall rules and security policies and implementing them across the infrastructure. AWS firewall manager helps you audit VPC security groups for compliance requirements and control network traffic effectively.



27. Amazon Appflow

Amazon Appflow is a no-code service that allows the integration of SaaS applications and AWS services effortlessly. To be more precise, it securely automates dataflows integrating third-party applications and AWS services without using codes. You can transfer data between SaaS applications such as Salesforce, SAP, Zendesk, etc. since

Amazon Appflow can be integrated with other applications in a few clicks. Especially, a large volume of data can be moved without breaking it up into batches using this service.



28. Amazon Route 53

It is a scalable cloud Domain Name System (DNS) service. It allows end-users to connect with Amazon EC2, Elastic load balancers, Amazon S3 buckets, and even outside AWS. In this service, the feature 'Route 53 application recovery controllers' configure DNS health checks and helps to monitor the ability of systems to recover from failures. And 'Route 53 traffic flow' helps manage traffic across the globe using routing methods such as latency-based routing, Geo DNS,



Geoproximity, and weighted round-robin.

29. AWS Cloud Formation

This AWS service creates and manages resources with templates. It is a single platform that can handle all AWS accounts across the globe. It automates resource management with AWS service integration and offers turnkey application distribution and governance controls. Also, AWS Cloud Formation can automate, test, and deploy infrastructure with continuous integration and delivery. And you can run applications right from AWS EC2 to complex multi-region applications using this service.



30. AWS Key Management Service (KMS)

AWS KMS manages the creation and control of encryption keys. It means that AWS KMS creates cryptographic keys and controls their uses across various applications. You can achieve a secure and resilient service using hardware resilient modules to protect keys. This service can be integrated with AWS Cloudtrail to provide logs of all key usage to precisely fulfil compliance and regulatory requirements.



Implementation of the Architecture of Bastion Host with Auto Scaling

Introduction:

Bastion Host :

A bastion host is a server whose purpose is to provide access to a private network from an external network, such as the Internet. Because of its exposure to potential attack, a bastion host must minimize the chances of penetration.

Why use Bastion Host?

The complete scenario can be explained as suppose there are clusters of instances in your public network. The public cloud allows you to create some private or isolated section of the cloud which can be used by the user for launching other services which are known as VPC (Virtual Private Network). So, the user wants to create a medium or a communication channel to your VPC insecure environment. So, there are many methods through which you can do this. The first decision you might make is to provide an external IP address. You can assign some services with an external IP address to access it over the internet. But some users might not want to use external IP addresses and want to use SSH tool for more security to connect to the VPC. So now if you are not providing it with the external IP address then the alternate remains are that create another instance on the network which becomes a gateway for the private network to the internet. It acts as a trusted relay for inbound.

Auto Scaling Group:

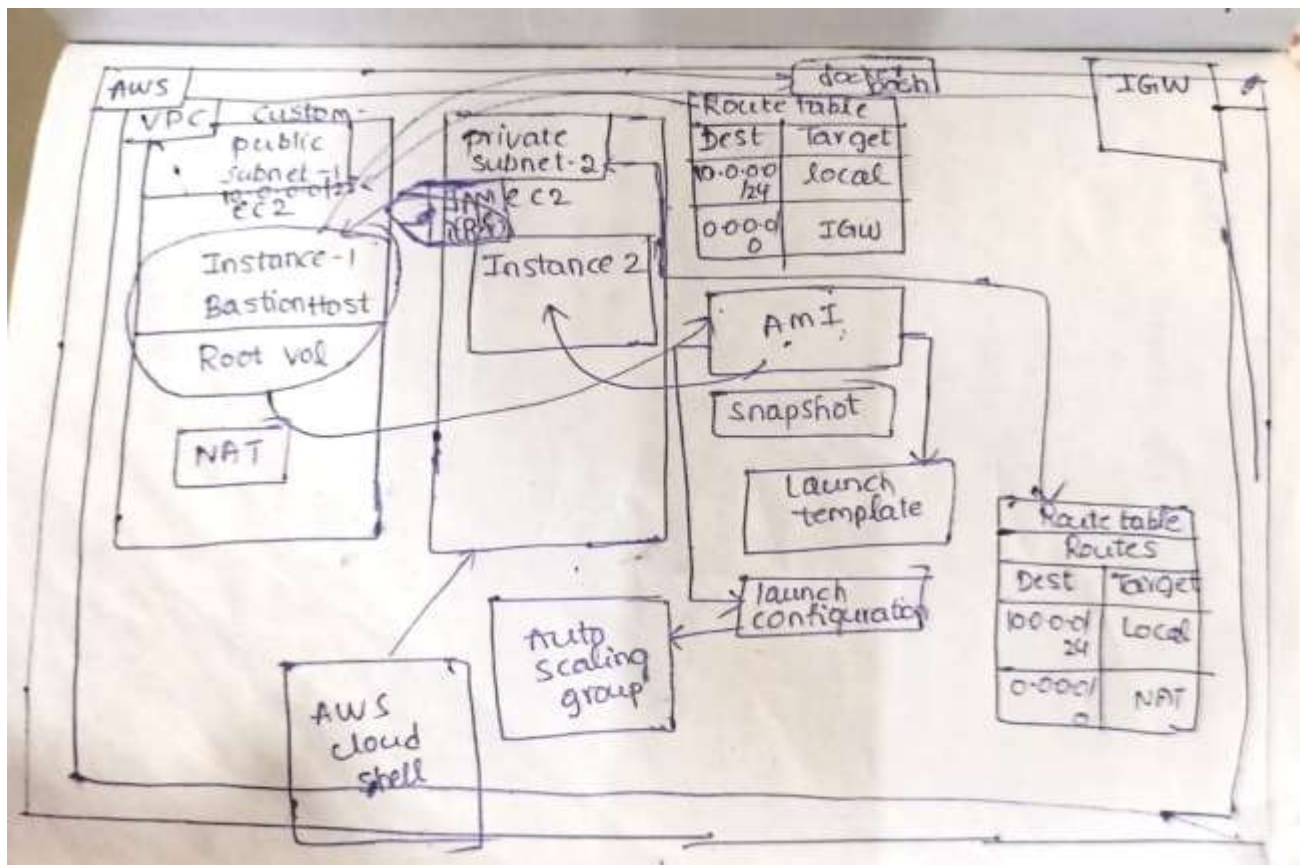
An Auto Scaling group contains a collection of EC2 instances that are treated as a logical grouping for the purposes of automatic scaling and management. An Auto Scaling group also lets you use Amazon EC2 Auto Scaling features such as health check replacements and scaling policies. Both maintaining the number of instances in an Auto Scaling group and automatic scaling are the core functionality of the Amazon EC2 Auto Scaling service. The size of an Auto Scaling group depends on the number of instances that you set as the desired capacity. You can adjust its size to meet demand, either manually or by using automatic scaling. An Auto Scaling group starts by launching enough instances to meet its desired capacity.

It maintains this number of instances by performing periodic health checks on the instances in the group. The Auto Scaling group continues to maintain a fixed number of instances even if an instance becomes unhealthy. If an instance becomes unhealthy, the group terminates the unhealthy instance and launches another instance to replace it.

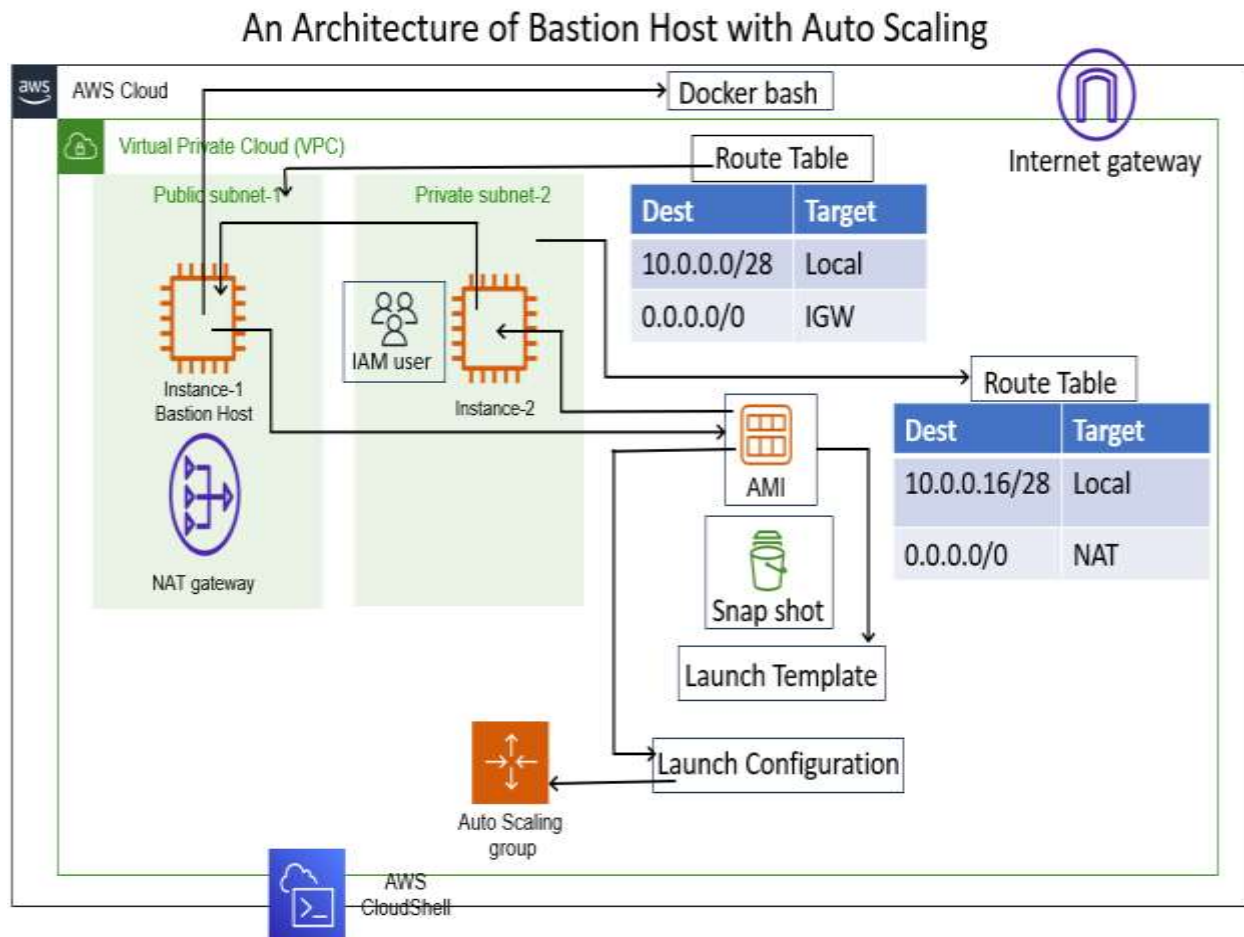
AWS Services Used:

- Virtual Private Cloud(VPC)
- Elastic Compute Cloud(EC2)
- Identity and Access Management(IAM)
- AWS Cloud Shell
- Auto Scaling Group

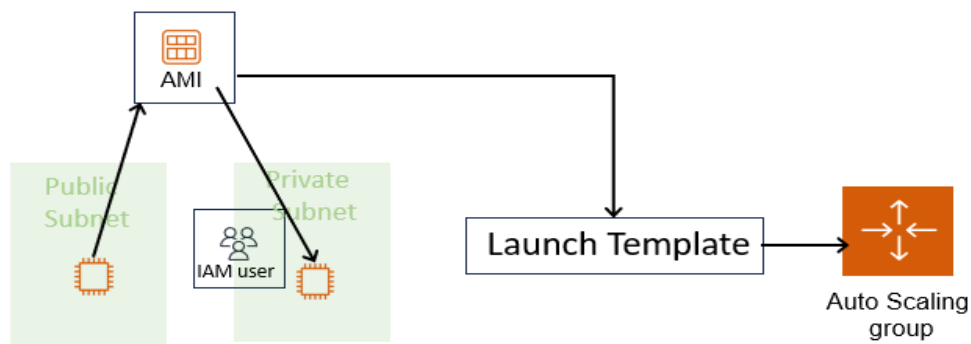
Rough Architecture:



Final Architecture:



Workflow:



- Create a VPC and edit the route table associated with it. Create a subnet and name it as public subnet. Edit the corresponding subnet association, create an internet gateway, and attach it to custom VPC. Now launch an ec2 instance with created VPC. Create an AMI for public instance.
- Create an IAM user and repeat the above process to the same VPC as IAM user. Now create a NAT gate and add it to the private route table. Thus, we can access the private instance through public instance.
- Now launch a template for AMI of public instance and create an auto scaling group. An instance is created automatically which reduces the utilization of CPU and allows the user to access the instance.

Implementation of the Project

Service 1: VPC



To create a VPC with the name "VPC1," follow these steps:

- Sign into the AWS Management Console.
- Open the Virtual Private Cloud Service.
- Click on the "Create VPC" option.
- Give the VPC name as vpc1 and CIDR value as 10.0.0.0/24.
- Click on create VPC.

VPC settings

Resources to create [help](#)
 Create only the VPC resource or the VPC and other networking resources.

☒ VPC only ☐ VPC and more

Name tag - optional [help](#)
 Creates a tag with a key of "Name" and a value that you specify.

IPv4 CIDR block [help](#)
☒ IPv4 CIDR manual input
☐ IPAM-allocated IPv4 CIDR block

IPv4 CIDR

IPv6 CIDR block [help](#)
☒ No IPv6 CIDR block
☐ IPAM-allocated IPv6 CIDR block
☐ Amazon-provided IPv6 CIDR block
☐ IPv6 CIDR owned by me

Your VPCs (2) [help](#)

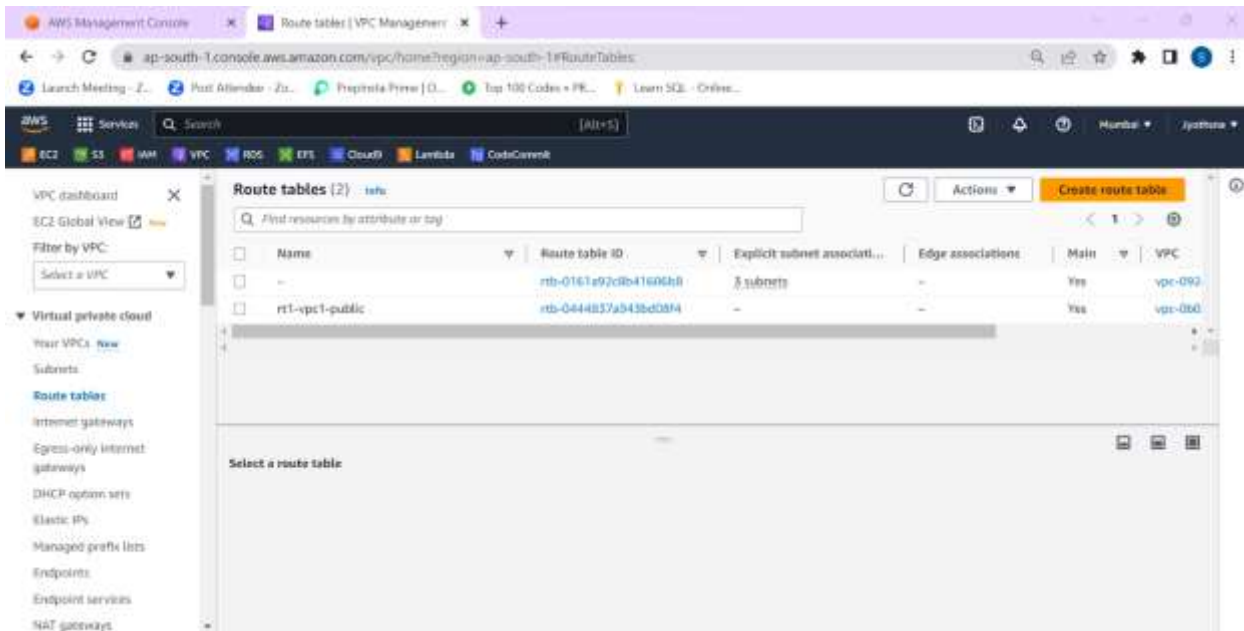
<input type="checkbox"/>	Name	VPC ID	State	IPv4 CIDR	IPv6 CIDR
<input type="checkbox"/>	vpc1	vpc-0b0392996214289e4	Available	10.0.0.0/24	-
<input type="checkbox"/>	-	vpc-0924b6f251ebf0f0	Available	172.11.0.0/16	-

Select a VPC above

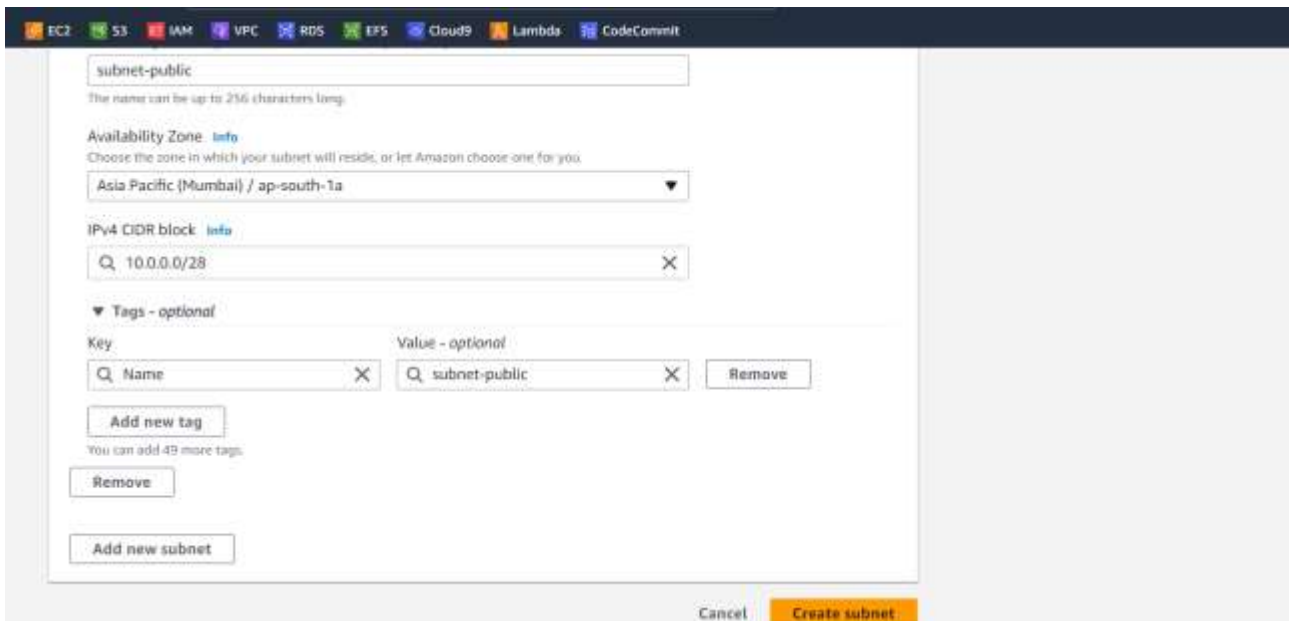
Virtual private cloud

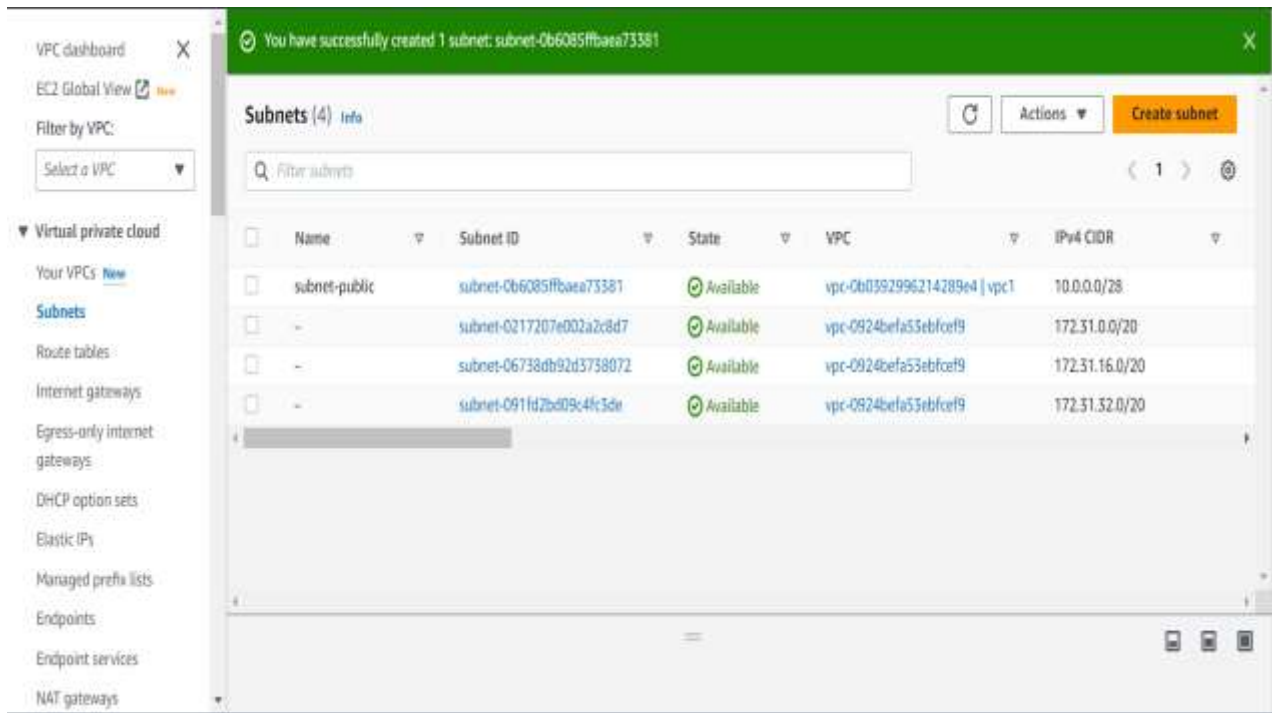
- Your VPCs [New](#)
- Subnets
- Route tables
- Internet gateways
- Egress-only Internet gateways
- DHCP option sets
- Elastic IPs
- Managed prefix lists
- Endpoints
- Endpoint services
- NAT gateways

- Go to route table and edit the name of route table as rt1-vpc1-public that is created along with vpc1.

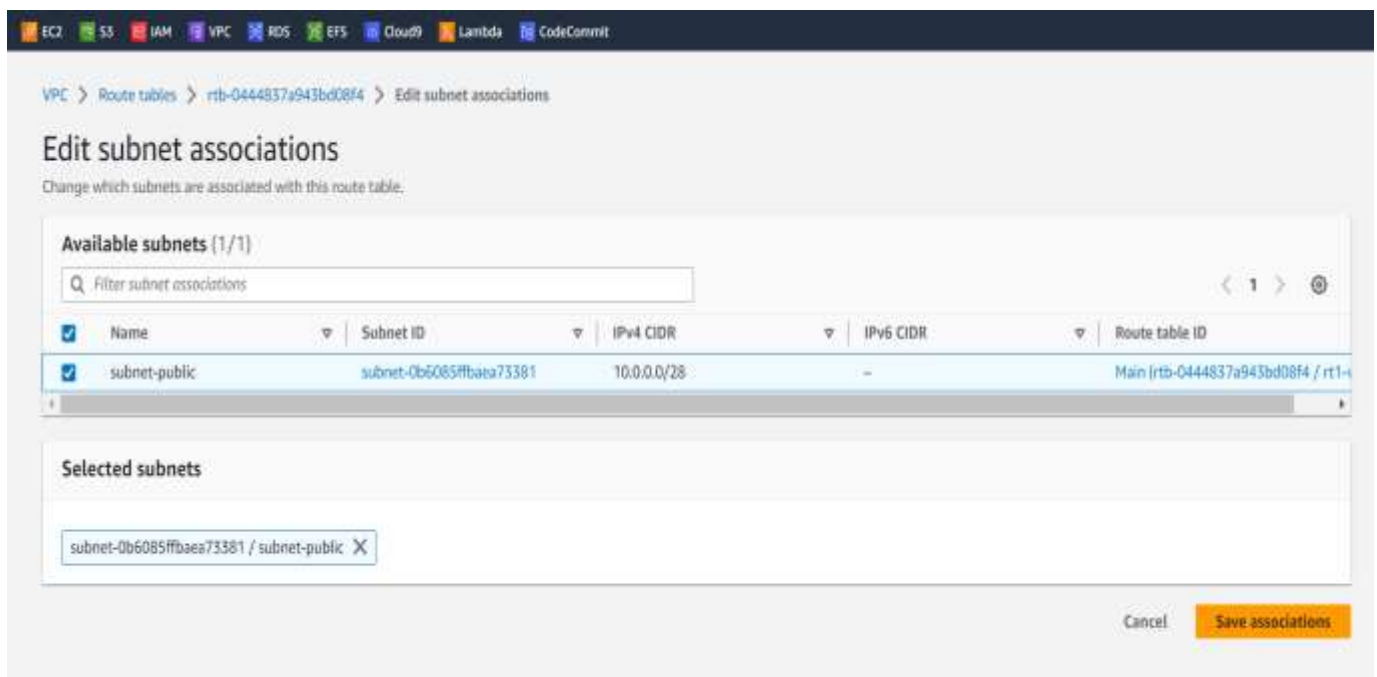


- Go to subnets and create a subnet and name it as public subnet. Select the availability zone as south-1a, CIDR as 10.0.0.0/28 and select create subnet.





- Go to route tables, select the public route table, and go to subnet associations and select public subnet and click on save associations.



- Create an internet gateway with the name igw1-vpc1.

Create internet gateway Info

An internet gateway is a virtual router that connects a VPC to the internet. To create a new internet gateway specify the name for the gateway below.

Internet gateway settings

Name tag
Creates a tag with a key of 'Name' and a value that you specify.

igw1-vpc1

Tags - optional
A tag is a label that you assign to an AWS resource. Each tag consists of a key and an optional value. You can use tags to search and filter your resources or track your AWS costs.

Key	Value - optional	
Name	igw1-vpc1	Remove

Add new tag
You can add 49 more tags.

Cancel **Create Internet gateway**

- Attach custom IGW to custom VPC.

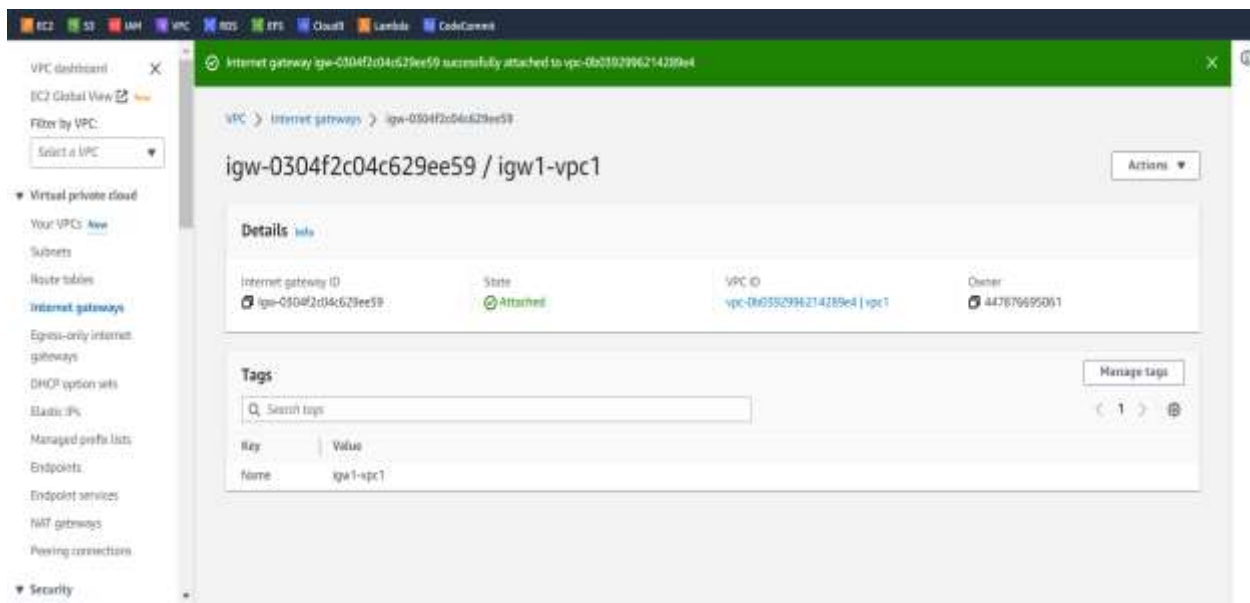
igw-0304f2c04c629ee59 / igw1-vpc1 Actions

Details Info

Internet gateway ID	State	VPC ID	Owner
igw-0304f2c04c629ee59	Detached	-	447876695061

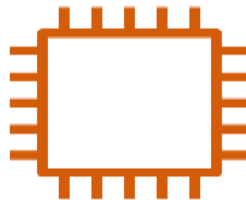
Tags Manage tags

Key	Value
Name	igw1-vpc1



- Internet Gate Way is successfully attached to VPC.
- Now go to route tables, select rt1-vpc1-public, click on edit routes, go to add routes, and add 0.0.0.0/0 at destination and select our internet gateway in destination and save it.

Service 2: EC2



Creating an EC2 instance in a public subnet as a Bastion Host:

- Name your instance as public instance,
- Select "Amazon Linux 2 AMI".
- Instance type "t2. micro".
- Select your existing key pair.
- Select your custom VPC, public subnet and enable the auto-assign public IP.
- In the security group section, select availability zone as south-1a.
- create a new security group, add security groups that supports SSH and all the traffic.

- Launch your instance.

EC2 > Instances > Launch an instance

Launch an instance [Info](#)

Amazon EC2 allows you to create virtual machines, or instances, that run on the AWS Cloud. Quickly get started by following the simple steps below:

Name and tags [Info](#)

Name

[Add additional tags](#)

▼ Application and OS Images (Amazon Machine Image) [Info](#)

An AMI is a template that contains the software configuration (operating system, application server, and applications) required to launch your instance. Search or Browse for AMIs if you don't see what you are looking for below.

[Quick Start](#)

▼ Summary

Number of instances [Info](#)

Software image (AMI)

Amazon Linux 2 Kernel 5.10 AMI...[read more](#)
ami-012b9156f755804f5

Virtual server type (instance type)

t2.micro

Firewall (security group)

New security group

Storage (volumes)

1 volume(s) - 8 GiB

[Free tier: In your first year includes](#) [X](#)

[Cancel](#) [Launch instance](#) [Review commands](#)

- Here we name the instance as “instance1-public”.

Amazon Linux macOS Ubuntu Windows Red Hat [Browse more AMIs](#)

Amazon Machine Image (AMI)

Amazon Linux 2 AMI (HVM) - Kernel 5.10, SSD Volume Type [Free tier eligible](#)

ami-012b9156f755804f5 (64-bit x86) / ami-0a0f75018c0da79 (64-bit ARM)

Virtualization: true - 8GB enabled: true - Root device type: xfs

Description

Amazon Linux 2 Kernel 5.10 AMI 2.0.20230612.0 x86_64 HVM gp2

Architecture

AMI ID

64-bit (x86) ami-012b9156f755804f5 [Verified provider](#)

▼ Instance type [Info](#)

Instance type

t2.micro

Family: t2 1 vCPU 1 GiB Memory Current generation: true [Free tier eligible](#) [All generations](#)

▼ Summary

Number of instances [Info](#)

Software image (AMI)

Amazon Linux 2 Kernel 5.10 AMI...[read more](#)
ami-012b9156f755804f5

Virtual server type (instance type)

t2.micro

Firewall (security group)

New security group

Storage (volumes)

1 volume(s) - 8 GiB

[Free tier: In your first year includes](#) [X](#)

[Cancel](#) [Launch instance](#) [Review commands](#)

- We choose “Amazon Linux 2 AMI(HVM) – Kernel 5.10.SSD Volume Type in AMI.

Firewall (security groups) [Info](#)

A security group is a set of firewall rules that control the traffic for your instance. Add rules to allow specific traffic to reach your instance.

☒ Create security group ☐ Select existing security group

Security group name - required

publicsg

This security group will be added to all network interfaces. The name can't be edited after the security group is created. Max length is 255 characters. Valid characters: a-z, A-Z, 0-9, spaces, and _-:/@#.%+=&|!\$*

Description - required [Info](#)

launch-wizard-3 created 2023-06-26T04:15:06.332Z

Inbound Security Group Rules

▼ Security group rule 1 (TCP, 22, 0.0.0.0/0)

Type [Info](#) Protocol [Info](#) Port range [Info](#)

ssh TCP 22

Source type [Info](#) Source [Info](#) Description - optional [Info](#)

Anywhere Add CIDR, prefix list or security group e.g. SSH for admin desktop

0.0.0.0/0 x ::/0 x

▼ Security group rule 2 (TCP, 0)

[Remove](#)

[Remove](#)

Summary

Number of instances [Info](#)

1

Software Image (AMI)

Amazon Linux 2 Kernel 5.10 AMI...[read more](#)
ami-012b9156f755804f5

Virtual server type (instance type)

t2.micro

Firewall (security group)

New security group

Storage (volumes)

1 volume(s) - 8 GiB

[Free tier](#) In your first hour includes

[Cancel](#) [Launch instance](#) [Review commands](#)

- Add “Type: All traffic”, “source type: Anywhere” in inbound security Group rules.

Type [Info](#) Protocol [Info](#) Port range [Info](#)

All traffic All All

Source type [Info](#) Source [Info](#) Description - optional [Info](#)

Anywhere Add CIDR, prefix list or security group e.g. SSH for admin desktop

0.0.0.0/0 x ::/0 x

[Rules with source of 0.0.0.0/0 allow all IP addresses to access your instance. We recommend setting security group rules to allow access from known IP addresses only.](#)

[Add security group rule](#)

[Advanced network configuration](#)

▼ Configure storage [Info](#) [Advanced](#)

1x 8 GiB gp2 Root volume (Not encrypted)

[Free tier eligible customers can get up to 30 GB of EBS General Purpose \(SSD\) or Magnetic storage](#)

Summary

Number of instances [Info](#)

1

Software Image (AMI)

Amazon Linux 2 Kernel 5.10 AMI...[read more](#)
ami-012b9156f755804f5

Virtual server type (instance type)

t2.micro

Firewall (security group)

New security group

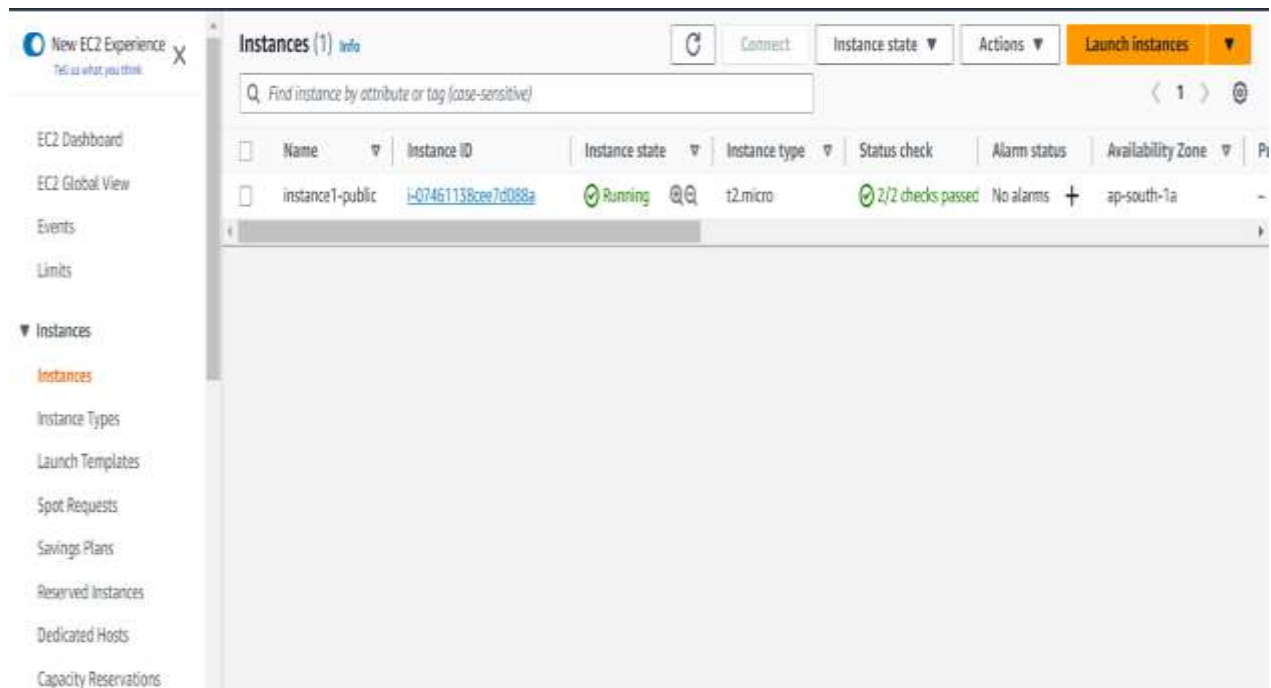
Storage (volumes)

1 volume(s) - 8 GiB

[Free tier](#) In your first hour includes

[Cancel](#) [Launch instance](#) [Review commands](#)

- Make sure that in configure storage GiB : 8 and root volume : gp2 are present.



- Now “instance1-public” is successfully launched.

Service 3: AMI



- To create an AMI first we need to create an image, for that look at the following procedure:
- Select the public instance.
- Go to action.
- Go to image and templates.
- Click on create image, give image name, and create the image.

Create image [Info](#)

An image (also referred to as an AMI) defines the programs and settings that are applied when you launch an EC2 instance. You can create an image from the configuration of an existing instance.

Instance ID

Image name

 Maximum 127 characters. Can't be modified after creation.

Image description - optional

 Maximum 255 characters.

No reboot
☐ Enable

Instance volumes

- After images have been created go to AMI, there we can find an AMI.
- Whenever an AMI is created a snapshot gets created along with it.

Amazon Machine Images (AMIs) (1) [Info](#)

Owned by me

Name	AMI ID	AMI name	Source	Owner
image1	ami-0ebc5bae0825ee88	forbastionhost	447876695061/forbastionhost	447876695061

Select an AMI

- After successfully creating an AMI go to instances and connect “instance1-public”.

Connect to instance Info

Connect to your instance i-07461138cee7d088a (instance1-public) using any of these options

EC2 Instance Connect | Session Manager | SSH client | EC2 serial console

Instance ID
i-07461138cee7d088a (instance1-public)

Connection Type

☒ **Connect using EC2 Instance Connect**
Connect using the EC2 Instance Connect browser-based client, with a public IPv4 address.

☐ **Connect using EC2 Instance Connect Endpoint**
Connect using the EC2 Instance Connect browser-based client, with a private IPv4 address and a VPC endpoint.

Public IP address
13.235.244.214

User name
Enter the user name defined in the AMI used to launch the instance. If you didn't define a custom user name, use the default user name, ec2-user.
ec2-user

Note: In most cases, the default user name, ec2-user, is correct. However, read your AMI usage instructions to check if the AMI owner has changed the default AMI user name.

Cancel **Connect**

- execute the following commands:

```
db_upgrade          grub2-efi-path      mountpoint          plymouth            rayno               thl
dbus-cleanups-sockets grub2-mount         sptat              pmcp                rayalog-recover-gl.pl tcptraceroute
dbus-daemon         grub2-script-check  asgattrib           pm-is-supported     runcon              tcsh
dbus-monitor        gsettings          asgcat             pod2html            run-parts           testd
dbus-run-session    gaoelix            asgcomp            pod2man             rvi                 testdctl
dbus-send           qtat               asgcomm            pod2text            rview               testnl

[root@ip-10-0-0-10 bin]# history
1  mkdir jyothona
2  cd jyothona
3  cat >file1
4  yum install docker -y
5  systemctl status docker
6  systemctl start docker
7  systemctl status docker
8  docker run -it ubuntu /bin/bash
9  docker images
10 docker ps -a
11 which docker
12 cd /bin
13 ls
14 history
[root@ip-10-0-0-10 bin]#
```

i-07461138cee7d088a (instance1-public)

Public IP: 13.235.244.214 Private IP: 10.0.0.10

Service 4: IAM User

Create a User in IAM:

To create a user, follow these steps:

- Open the AWS Management Console in your web browser.
- Navigate to the IAM service.
- In the left navigation pane, click on "Users" and then click on the "Add user" button.

User Details:

- Provide a name for the user in the "Username" field, such as "malika".
- Optionally, you can enable the checkbox for "Programmatic access" to allow the user to interact with AWS services programmatically through APIs.
- If necessary, enable the checkbox for "AWS Management Console access" to grant the user access to the AWS Management Console.

Password:

- If you selected "AWS Management Console access" in the previous step, choose one of the following options to set the user's password:
- "Autogenerated password" will create a randomly generated password for the user.
- "Custom password" will allow you to set a specific password for the user.
- "Require password reset" will prompt the user to change their password upon first login.

Permissions:

- In the "Set permissions" section, you can choose to attach existing policies to the user or create a custom policy. This will define the user's permissions and access to AWS resources.
- To attach existing policies, click on the "Add permissions" button, select the desired policies from the list, and click on the "Next: Tags" button.

- To create a custom policy, click on the "Create group" button and follow the steps to define and attach the policy to the user.

Tags:

- Optionally, you can add tags to the user for better organization and management. Tags are key-value pairs that provide metadata about the user.

Review:

- Review the user details, permissions, and tags to ensure they are correct.
- Click on the "Create user" button to create the user.
- The user "malika" will be created in IAM with the specified permissions and access settings.
Make sure to securely manage and share the user's access credentials, such as access key and secret key, if programmatic access is enabled.
- Copy console sign in link another Browser
- Enter proper credentials and login.

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.

User groups (1/1)

Search

Group name	Users	Attached policies	Created
alpha	0	AdministrationAccess	2023-06-24

Set permissions boundary - optional

Cancel Previous Next

User details

The user name can have up to 64 characters. Valid characters: A-Z, a-z, 0-9, and +, -, @, _.

 Provide user access to the AWS Management Console - optional
If you're granting 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?

☐ Specify a user or identity Carrier - Recommended

We recommend that you use Identity Center to provide unified 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 IBM user](#)

We recommend that you create API users only if you need to enable programmatic access through access keys, service-specific credentials for AWS CodeCommit or Amazon SageMaker, or a listing credential for emergency incident access.

☐ Autogenerated password

You can view the preview after you create the user.

Custom permitted

[Price is a double-precision floating-point value]

Abstract

* Please see all units & franchise in group.

[View user](#)

You can view and download the user's password and email instructions for signing in to the AWS Management Console.

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

© Copied

en-us

 <https://447876695061.signin.aws.amazon.com/console>

User name

 malika

Console password

 [Show](#)[Download .csv file](#)[Return to users list](#)

-
- The screenshot displays the AWS Management Console's VPC dashboard. A green notification banner at the top indicates the successful creation of a subnet: 'subnet-0b62201e0b5a1360b'. The 'Subnets (5)' section is selected, showing a table of subnets. The table includes columns for Name, ID, Status, VPC, and CIDR block. Three subnets are listed: 'subnet-public', 'subnet-02117207e602a2c8d7', and 'subnet2-private', all with a status of 'Available'. The 'subnet2-private' row is highlighted. The left sidebar shows the 'Virtual private cloud' section expanded, with various VPC-related services listed.
- | Name | ID | Status | VPC | CIDR block |
|-----------------|---------------------------|-----------|------------------------------|---------------|
| subnet-public | subnet-0b6085ffbaea73381 | Available | vpc-0b0592996214289e4 vpc1 | 10.0.0.0/24 |
| subnet | subnet-02117207e602a2c8d7 | Available | vpc-0924bfa55ebfc0f9 | 172.31.0.0/24 |
| subnet2-private | subnet-0b62201e0b5a1360b | Available | vpc-0b0392996214289e4 vpc1 | 10.0.0.16/24 |

AWS Management Console | VPC Management Console

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

Create route table

A route table specifies how packets are forwarded between the subnets within your VPC, the internet, and your VPN connection.

Route table settings

Name - optional
Create a tag with a key of Name and a value that you specify.

VPC
The VPC to use for this route table.

vpc-0b0392996214289e4 (vpc1)

Tags

A tag is a label that you assign to an AWS resource. Each tag consists of a key and an optional value. You can use tags to search and filter your resources or track your AWS costs.

Key Value - optional

Name X rt1-vpc1-private X Remove

Add new tag

You can add 49 more tags.

Cancel Create route table

AWS Management Console | VPC Management Console

ap-south-1.console.aws.amazon.com/vpc/home?region=ap-south-1#EditRouteTableSubnetAssociations:RouteTableId=rtb-061af5649ee2dba7a

Edit subnet associations

Change which subnets are associated with this route table.

Available subnets (7/2)

Filter subnet associations

	Name	Subnet ID	IPv4 CIDR	IPv6 CIDR	Route table ID
<input type="checkbox"/>	subnet-public	subnet-0b6085f9baca73381	10.0.0.0/28	-	rtb-0444837a943bd08f4 / rt1-vpc1-p
<input checked="" type="checkbox"/>	subnet2-private	subnet-0b62201e0b3a1360b	10.0.0.16/28	-	Main (rtb-0444837a943bd08f4 / rt1-v

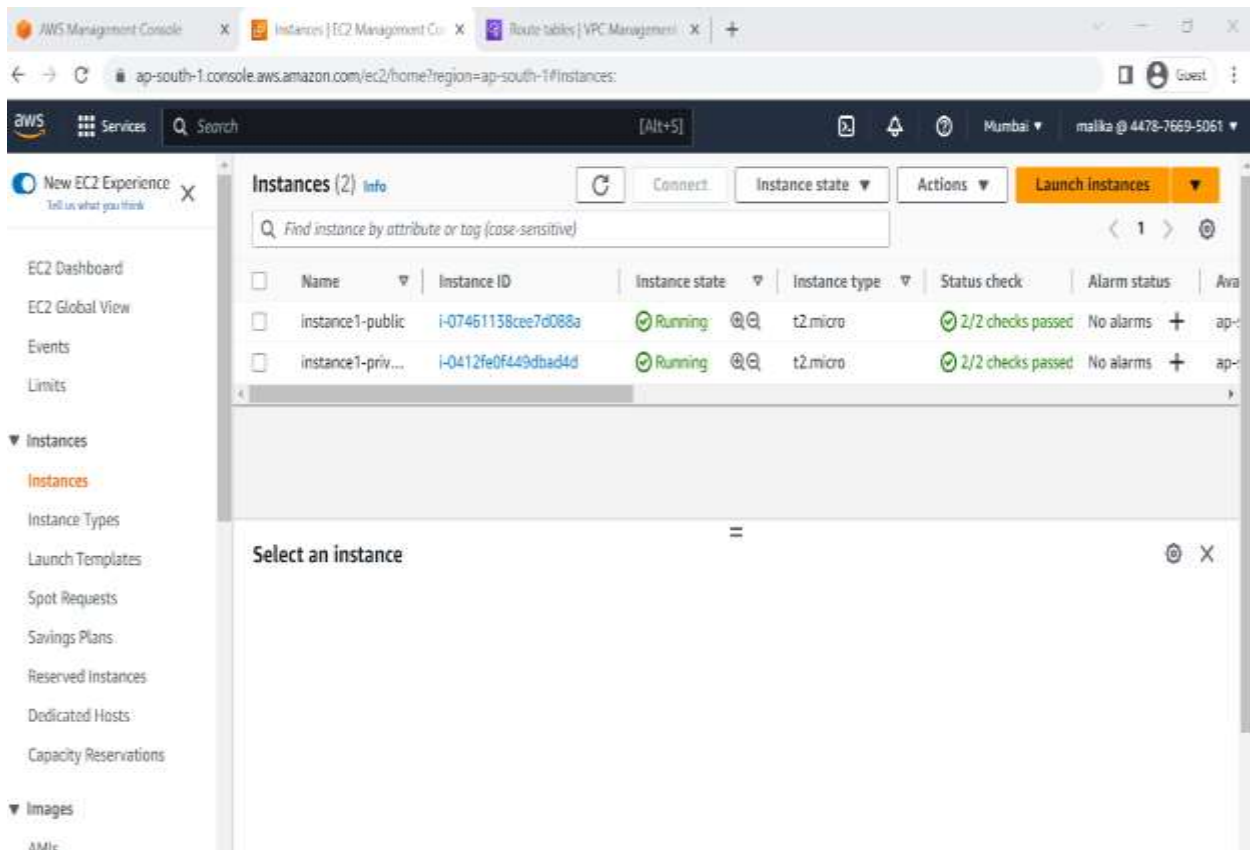
Selected subnets

subnet-0b62201e0b3a1360b / subnet2-private X

Cancel Save associations

Now launch an ec2 instance:

- Choose instance name as “instance-private”.
- Select “Amazon Linux 2 AMI”.
- Instance type “t2. micro”.
- Select your existing key pair which is used in instance-public.
- Select your custom VPC, public subnet and disable the auto-assign public IP.
- In the security group section, select availability zone as south-1b.
- Choose existing security group that is used in instance-public.
- Launch your instance.



Service 5: CloudShell

Go to CloudShell and execute the following commands:

The screenshot shows the AWS CloudShell interface in a web browser. The terminal window displays the following commands and output:

```
[cloudshell-user@ip-10-4-41-26 ~]$ ls
[cloudshell-user@ip-10-4-41-26 ~]$ ls
malika.pem
[cloudshell-user@ip-10-4-41-26 ~]$ chmod 400 malika.pem
[cloudshell-user@ip-10-4-41-26 ~]$ scp -i malika.pem malika.pem ec2-user@13.235.244.214:/home/ec2-user
The authenticity of host '13.235.244.214 (13.235.244.214)' can't be established.
ECDSA key fingerprint is SHA256:pMEHL39eeH058hzCQ6rCG34hIHRKESlgdWdLbk.
ECDSA key fingerprint is MD5:74:e8:97:18:74:28:11:ae:07:51:ba:67:2d:53:d0:e4.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '13.235.244.214' (ECDSA) to the list of known hosts.
malika.pem
[cloudshell-user@ip-10-4-41-26 ~]$
```

A green notification box at the bottom right states: "File upload successful. malika.pem was successfully uploaded to the following directory: /home/cloudshell-user."

- After running on the cloudshell go to instances and again select instance-public and connect the instance.

Run the following commands:

The screenshot shows a terminal window with the following commands and output:

```
Last login: Mon Jun 26 04:37:22 2023 from ec2-13-233-177-3.ap-south-1.compute.amazonaws.com

 _ _ | _ _ | _ )
 _ | ( _ _ | /   Amazon Linux 2 AMI
 _ _ | \ _ _ | _ _ |

https://aws.amazon.com/amazon-linux-2/
[ec2-user@ip-10-0-0-10 ~]$ sudo su
[root@ip-10-0-0-10 ec2-user]# ls
jyothana malika.pem
[root@ip-10-0-0-10 ec2-user]# cd jyothana
[root@ip-10-0-0-10 jyothana]# ls
file1
[root@ip-10-0-0-10 jyothana]# cd /home/ec2-user
[root@ip-10-0-0-10 ec2-user]# ssh -i "malika.pem" ec2-user@10.0.0.20
The authenticity of host '10.0.0.20 (10.0.0.20)' can't be established.
ECDSA key fingerprint is SHA256:TYMagGvj2qrABYN3FtlqlbugTW5OqNTZbvs3j0it2Is.
ECDSA key fingerprint is MD5:ce:ce:ce:ce:4e:5e:c2:45:23:86:b3:6d:0f:e7:73:25.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '10.0.0.20' (ECDSA) to the list of known hosts.
```

- SSH command in Linux: The ssh command provides a secure encrypted connection between two hosts over an insecure network.
- By running this command with the
- We can use the private instance as another AMI is created for private instance.

```

      _ _ _ _ _
     _ _ _ _ _ Amazon Linux 2 AMI
    _ _ _ _ _

https://aws.amazon.com/amazon-linux-2/
[ec2-user@ip-10-0-0-20 ~]$ which docker
/usr/bin/which: no docker in (/usr/local/bin:/usr/bin:/usr/local/sbin:/usr/sbin:/home/ec2-user/.local/bin:/home/ec2-user/bin)
[ec2-user@ip-10-0-0-20 ~]$ ping google.com
PING google.com (142.250.199.142) 56(84) bytes of data.
^C
--- google.com ping statistics ---
4 packets transmitted, 0 received, 100% packet loss, time 3048ms

[ec2-user@ip-10-0-0-20 ~]$

```

- AS the route table is not attached to the NAT gateway, we are unable to receive the transmitted packets.

Process to successfully run a private instance.

- Creating a NAT gateway:
- If we want to receive the transmitted packets without any packet loss in a private instance then a NAT gateway must be attached to the route table.
- Go to VPC
- Go to NAT gateway
- Click on create NAT gateway
- Give a name for NAT gate like "natg"
- Allocate subnet-public at choose subnet

- Click on allocate elastic IP then an elastic IP is automatically allocated
- Click on create NAT gateway

✓ Elastic IP address 43.204.50.240 (eipalloc-0dd4cc1a420cae47e) allocated.

VPC > NAT gateways > Create NAT gateway

Create NAT gateway [Info](#)

A highly available, managed Network Address Translation (NAT) service that instances in private subnets can use to connect to services in other VPCs, on-premises networks, or the internet.

NAT gateway settings

Name - optional
Create a tag with a key of 'Name' and a value that you specify.

nsq

The name can be up to 256 characters long.

Subnet
Select a subnet in which to create the NAT gateway.

subnet-0b6085ffb9ea73381 (subnet-public) ▼

Connectivity type
Select a connectivity type for the NAT gateway.

Select a connectivity type for the NAT gateway.

☒ Public
☐ Private

Elastic IP allocation ID [Info](#)
Assign an Elastic IP address to the NAT gateway.

eipalloc-0dd4cc1a420cae47e ▼ Allocate Elastic IP

► **Additional settings** [Info](#)

Tags

A tag is a label that you assign to an AWS resource. Each tag consists of a key and an optional value. You can use tags to search and filter your resources or track your AWS costs.

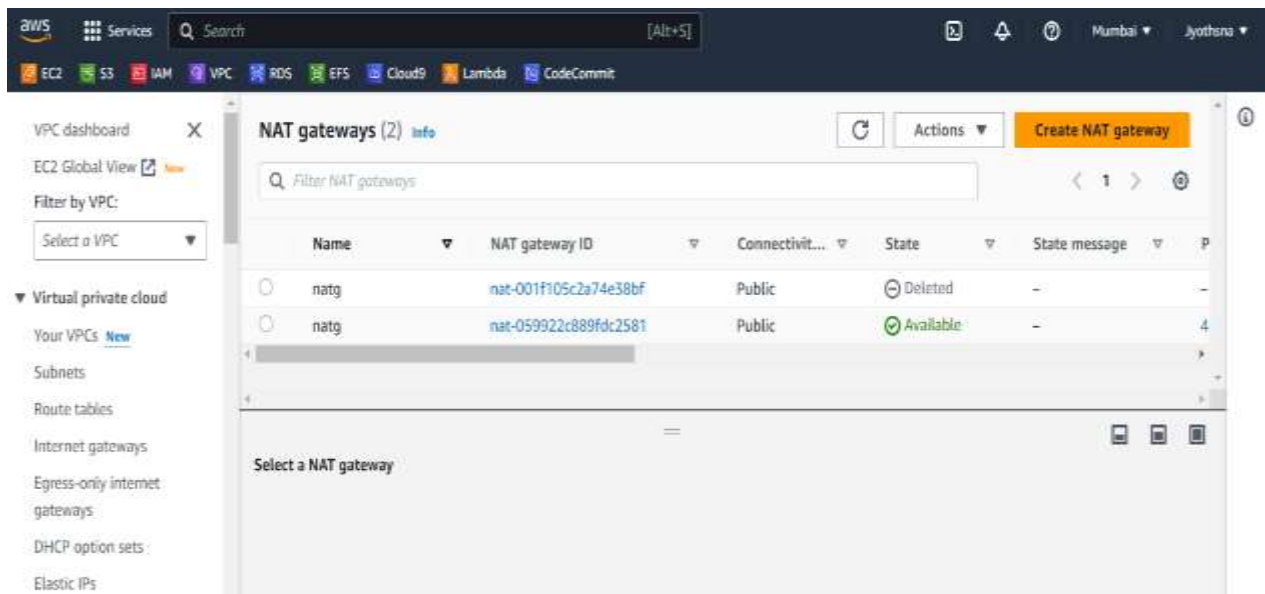
Key	Value - optional	
Q Name X	Q nsq X	Remove

Add new tag

You can add 49 more tags.

Cancel Create NAT gateway

- Now a NAT gate has been successfully created.



- Now go to route tables and select “rt1-vpc1-private” and go to edit routes.
- Click on add routes and add 0.0.0.0/0 at local and select the created NAT gateway and click on save.
- NAT gate is successfully added to routes.
- Go to connected public instance again and run the command “ping google.com“, we find that transmitted packets are successfully received without any loss.

```
[ec2-user@ip-10-0-0-20 ~]$ ping google.com
PING google.com (142.250.183.142) 56(84) bytes of data.
^C
--- google.com ping statistics ---
6 packets transmitted, 0 received, 100% packet loss, time 5101ms

[ec2-user@ip-10-0-0-20 ~]$ ping google.com
PING google.com (142.250.182.238) 56(84) bytes of data.
64 bytes from bom07s29-in-fl14.1e100.net (142.250.182.238): icmp_seq=1 ttl=109 time=2.55 ms
64 bytes from bom07s29-in-fl14.1e100.net (142.250.182.238): icmp_seq=2 ttl=109 time=1.81 ms
64 bytes from bom07s29-in-fl14.1e100.net (142.250.182.238): icmp_seq=3 ttl=109 time=1.87 ms
64 bytes from bom07s29-in-fl14.1e100.net (142.250.182.238): icmp_seq=4 ttl=109 time=1.82 ms
64 bytes from bom07s29-in-fl14.1e100.net (142.250.182.238): icmp_seq=5 ttl=109 time=1.79 ms
64 bytes from bom07s29-in-fl14.1e100.net (142.250.182.238): icmp_seq=6 ttl=109 time=1.84 ms
^C
--- google.com ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 5009ms
rtt min/avg/max/mdev = 1.796/1.951/2.554/0.272 ms
[ec2-user@ip-10-0-0-20 ~]$
```

Creating an Auto Scaling Group

- An auto scaling group creates a running instance itself which reduces the CPU utilization and provides users to access an instance with all the features.
- So, we create a template and auto scaling group through existing AMI.
- Launching a Template:
- Launching a Template includes the following steps:
- Go to ec2.
- Go to launch template.
- Give the name of a template, for suppose “my template.”
- Select AMI in Application & OS Images and choose my AMI and click at owned by me it displays the existing images then select image.
- Give instance type as “t2 micro.”
- Select the existing key pair.
- Go to network settings and choose the existing security group.
- Click on launch template.

Launch template name and description

Launch template name - /required/

Must be unique to this account. Max 128 chars. No spaces or special characters like \$, %, @.

Template version description

Max 255 chars

Auto Scaling guidance: [Info](#)
 Select this if you intend to use this template with EC2 Auto Scaling.
☐ Provide guidance to help me set up a template that I can use with EC2 Auto Scaling.

► Template tags
 ► Source template

Summary

Software image (AMI)
 forbastionhost
 ami-0ebc1ba0325ee88

Virtual server type (instance type)
 -

Firewall (security group)
 -

Storage (volumes)
 1 volume(s) - 8 GiB

Cancel **Create launch template**

Application and OS Images (Amazon Machine Image) [Info](#)
 An AMI is a template that contains the software configuration (operating system, application server, and applications) required to launch your instance. Search or Browse for AMIs if you don't see what you are looking for below.

Recently **My AMIs** Quick Start

☐ Don't include in launch template ☒ Owned by me [Browse more AMIs](#)
including AMIs from AWS, Marketplace and the Community

☐ Shared with me

Amazon Machine Image (AMI)

AMI ID	Virtualization format	OS	Architecture	Root device type
forbastionhost ami-0ebc1ba0325ee88 2023-09-26T04:27:00.000Z	Virtualization format	OS	Architecture	Root device type

Summary

Software image (AMI)
 forbastionhost
 ami-0ebc1ba0325ee88

Virtual server type (instance type)
 -

Firewall (security group)
 -

Storage (volumes)
 1 volume(s) - 8 GiB

Cancel **Create launch template**

Instance type [Info](#) Advanced

Instance type

t2.micro Free tier eligible

Family: t2 | 1 vCPU | 1 GiB Memory | Current generation: true

On-Demand Linux pricing: 0.0124 USD per Hour

On-Demand Windows pricing: 0.017 USD per Hour

On-Demand RHES pricing: 0.0724 USD per Hour

On-Demand SUSE pricing: 0.0124 USD per Hour

☐ All generations

[Compare instance types](#)

▼ Key pair (login) [Info](#)

You can use a key pair to securely connect to your instance. Ensure that you have access to the selected key pair before you launch the instance.

Key pair name

malika

[Create new key pair](#)

▼ Summary

Software Image (AMI)

forbastionhost
ami-0ebc5bae0825ev88

Virtual server type (instance type)

t2.micro

Firewall (security group)

-

Storage (volumes)

1 volume(s) - 8 GiB

Cancel [Create launch template](#)

Subnet [Info](#)

subnet-0b6085fbaea73361 subnet-public

VPC: vpc-060392996214289e4 | Owner: 447876695061

Availability Zone: ap-south-1a | IP addresses available: 9 | CIDR: 10.0.0.0/28

[Create new subnet](#)

When you specify a subnet, a network interface is automatically added to your template.

Firewall (security groups) [Info](#)

A security group is a set of firewall rules that control the traffic for your instance. Add rules to allow specific traffic to reach your instance.

☒ Select existing security group ☐ Create security group

Common security groups [Info](#)

Select security groups

publicsg sg-0bf4d63321245d481 X

VPC: vpc-060392996214289e4

[Compare security group rules](#)

Security groups that you add or remove here will be added to or removed from all your network interfaces.

▶ Advanced network configuration

▼ Summary

Software Image (AMI)

forbastionhost
ami-0ebc5bae0825ev88

Virtual server type (instance type)

t2.micro

Firewall (security group)

publicsg

Storage (volumes)

1 volume(s) - 8 GiB

Cancel [Create launch template](#)

- After successfully launching a template go to Auto Scaling groups and click on create auto scaling group
- Give the name as “asg.”
- Click on launch template, it shows a created template, click on that.
- Click on next.

EC2 > Auto Scaling groups > Create Auto Scaling group

Step 1
Choose launch template

Step 2
Choose instance launch options

Step 3 - optional
Configure advanced options

Step 4 - optional
Configure group size and scaling policies

Step 5 - optional
Add notifications

Step 6 - optional
Add tags

Step 7
Review

Choose launch template [info](#)

Specify a launch template that contains settings common to all EC2 instances that are launched by this Auto Scaling group.

Name

Auto Scaling group name
Enter a name to identify the group.

myg

Must be unique to this account in the current Region and no more than 255 characters.

Launch template [info](#)

For accounts created after May 31, 2023, the EC2 console only supports creating Auto Scaling groups with launch templates. Creating Auto Scaling groups with launch configurations is not recommended but still available via the CLI and API until December 31, 2023.

Launch template
Choose a launch template that contains the instance-level settings, such as the Amazon Machine Image (AMI), instance type, key pair, and security groups.

mytemplate

[Create a launch template](#)

Version
Default (1)

[Create a launch template version](#)

Description	Launch template mytemplate lt-0430006d84d30d5b8	Instance type t2.micro
AMI ID ami-0ebcb5bae0825ee88	Security groups -	Request Spot Instances No
Key pair name malika	Security group IDs sg-0bf4d63321245d481	

Additional details

Storage (volumes) -	Date created Mon Jun 26 2023 11:11:26 GMT+0530 (India Standard Time)
------------------------	--

Cancel **Next**

- In vpc select vpc1
- Select all the shown availability zones.
- Click on next

Choose the VPC network environment that your instances are launched into, and customize the instance types and purchase options.

Network info

For most applications, you can use multiple Availability Zones and let EC2 Auto Scaling balance your instances across the zones. The default VPC and default subnets are suitable for getting started quickly.

VPC
Choose the VPC that defines the virtual network for your Auto Scaling group.

vpc-060392996214289ed (vpc-1)
10.0.0.0/24

[Create a VPC](#)

Availability Zones and subnets
Define which Availability Zones and subnets your Auto Scaling group will use in the chosen VPC.

Select Availability Zones and subnets

ap-south-1a (subnet-0b60b5f8b0a75581) (subnet-public) 10.0.0.0/28	✕
ap-south-1b (subnet-0b62201e0b8a1340b) (subnet2-private) 10.0.0.16/28	✕

[Create a subnet](#)

- Click on the attach to the new load balancer and choose the existing target group.
- Click on next.

Use the options below to attach your Auto Scaling group to an existing load balancer, or to a new load balancer that you define.

☐ No load balancer
Traffic to your Auto Scaling group will not be fronted by a load balancer.

☐ Attach to an existing load balancer
Choose from your existing load balancers.

☒ Attach to a new load balancer
Quickly create a basic load balancer to attach to your Auto Scaling group.

Attach to a new load balancer

Define a new load balancer to create for attachment to this Auto Scaling group.

Load balancer type
Choose from the load balancer types offered below. Type selection cannot be changed after the load balancer is created. If you need a different type of load balancer than those offered here, visit the [Load Balancing console](#).

☒ Application Load Balancer
HTTP, HTTPS

☐ Network Load Balancer
TCP, UDP, TLS

Load balancer name
Name cannot be changed after the load balancer is created.

asg-1

Load balancer scheme
Scheme cannot be changed after the load balancer is created.

☒ Internal

☐ Internet-facing

- In configure group size and scaling policies choose the maximum capacity as 10 and continue with the other things

The screenshot shows the 'Create Auto Scaling group' wizard in the AWS Management Console, specifically Step 4: 'Configure group size and scaling policies'. The left sidebar lists the steps: Step 1: Choose launch template, Step 2: Choose instance launch options, Step 3 - optional: Configure advanced options, Step 4 - optional: Configure group size and scaling policies (current step), Step 5 - optional: Add notifications, Step 6 - optional: Add tags, and Step 7: Review.

The main content area is titled 'Configure group size and scaling policies - optional'. It includes a sub-section 'Group size - optional' with the instruction: 'Specify the size of the Auto Scaling group by changing the desired capacity. You can also specify minimum and maximum capacity limits. Your desired capacity must be within the limit range.' Below this, there are three input fields: 'Desired capacity' (set to 1), 'Minimum capacity' (set to 1), and 'Maximum capacity' (set to 10).

- In target tracking scaling policy set the target value to 65 and all others remain same
- Click on next.

The screenshot shows the 'Scaling policies - optional' configuration screen. It prompts the user to 'Choose whether to use a scaling policy to dynamically resize your Auto Scaling group to meet changes in demand.' There are two radio button options: 'Target tracking scaling policy' (selected) and 'None'. The 'Target tracking scaling policy' option includes a sub-instruction: 'Choose a desired outcome and leave it to the scaling policy to add and remove capacity as needed to achieve that outcome.'

Below the radio buttons, there are several input fields: 'Scaling policy name' (set to 'Target Tracking Policy'), 'Metric type' (set to 'Average CPU utilization'), 'Target value' (set to 65), and 'Instances need' (set to 300 seconds warm up before including in metric). At the bottom, there is a checkbox labeled 'Disable scale in to create only a scale-out policy' which is currently unchecked.

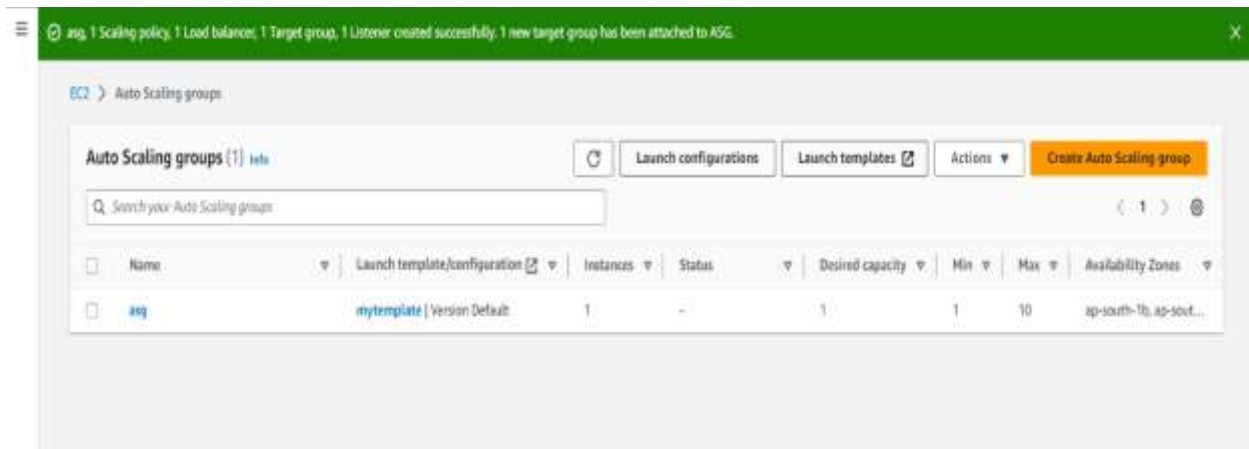
- Click on next.
- Again, click on next.

The screenshot shows the 'Add tags - optional' step in the AWS Management Console. On the left, a sidebar lists steps 1 through 7. Step 6, 'Add tags', is the current step. The main area has a title 'Add tags - optional' with a link to the AWS documentation. Below the title is a text box explaining that tags help search, filter, and track resources. A blue callout box contains a tip about adding tags to instances and EBS volumes. Below this is a section titled 'Tags (0)' with an 'Add tag' button and a note '0/1 remaining'. At the bottom right are 'Cancel', 'Previous', and 'Next' buttons.

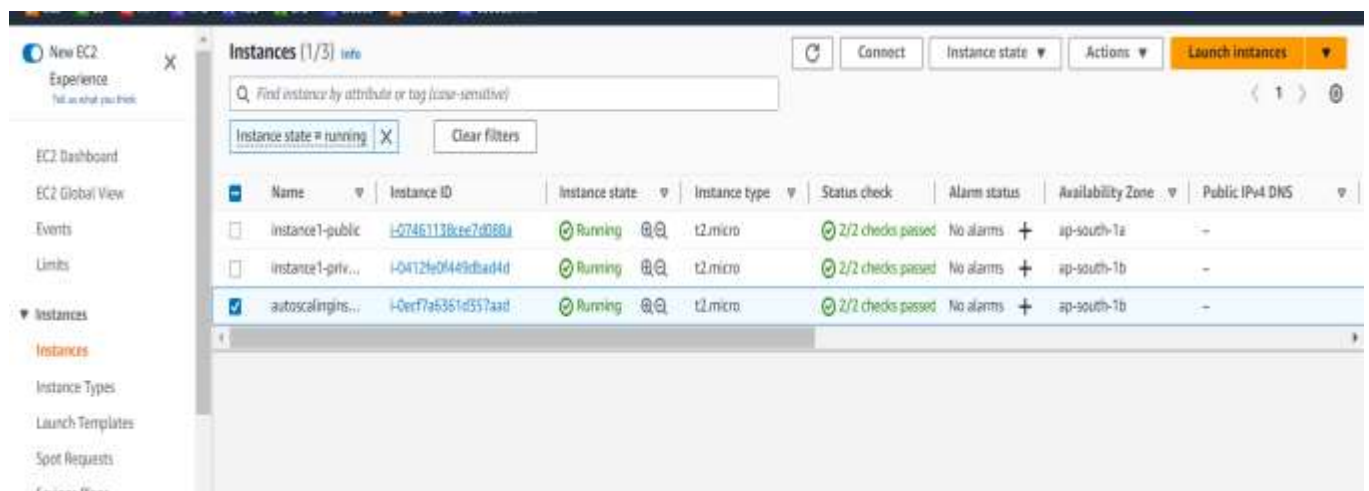
- Click on create Auto scaling group.

The screenshot shows two sections of the AWS Management Console. The first section is 'Step 5: Add notifications', which includes a toggle for 'Instance scale-in protection' (currently disabled) and an 'Edit' button. Below this is a 'Notifications' section showing 'No notifications'. The second section is 'Step 6: Add tags', which includes an 'Edit' button and a 'Tags (0)' section. The 'Tags (0)' section has a table with columns 'Key', 'Value', and 'Tag new instances'. The table is currently empty, showing 'No tags'. At the bottom are 'Cancel', 'Previous', and 'Create Auto Scaling group' buttons.

- An auto scaling group was successfully created.



- We can see the instance created along with Auto Scaling group.
- Name the instance as “autoscaling instance.”



- An auto scaling instance doesn't have an IPv4 address. So, we cannot connect the instance directly.
- We need to create an end point to connect an auto scaling instance.

Creating an endpoint:

- Go to vpc.
- Go to the end points.
- Click on create end point.
- Give name to an end point ex: “ep.”
- Choose ec2 instance connect end point.
- Select vpc1 in vpc.
- Select existing security group.
- Select “subnet-private” in subnet.
- Click on create end point.

VPC > Endpoints > Create endpoint

Create endpoint Info

There are three types of VPC endpoints – Interface endpoints, Gateway Load Balancer endpoints, and Gateway endpoints. Interface endpoints and Gateway Load Balancer endpoints are powered by AWS PrivateLink, and use an Elastic Network Interface (ENI) as an entry point for traffic destined to the service. Interface endpoints are typically accessed using the public or private DNS name associated with the service, while Gateway endpoints and Gateway Load Balancer endpoints serve as a target for a route in your route table for traffic destined for the service.

Endpoint settings

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

ep

Service category
Select the service category

<input type="radio"/> AWS services Services provided by Amazon	<input type="radio"/> PrivateLink Ready partner services Services with an AWS Service Ready designation	<input type="radio"/> AWS Marketplace services Services that you've purchased through AWS Marketplace
<input checked="" type="radio"/> EC2 Instance Connect Endpoint An elastic network interface that	<input type="radio"/> Other endpoint services Find services shared with you by service name	

Select the VPC in which to create the endpoint.

VPC
The VPC in which to create your endpoint.

vpc-0b0392996214289e4 (vpc1) ↕ ↻

► **Additional settings**

Security groups (1/2) Info ↻

Find resources by attribute or tag

<input type="checkbox"/>	Group ID	Group name	VPC ID
<input type="checkbox"/>	sg-079d8fb5f6be531f7	default	vpc-0b0392996214289e4
<input checked="" type="checkbox"/>	sg-0bf4d63321245d481	publicsg	vpc-0b0392996214289e4

sg-0bf4d63321245d481 ✕

Subnet
Select the Subnet in which to create the endpoint.

Subnet
Select the subnets in which to create the endpoint.

subnet-0b62201e0b3a1360b (subnet2-private) ↕ ↻

- An end point is created successfully.

Successfully created VPC endpoint
eice-0b7842ce650f4b91d

Endpoints (1/1) Info ↻ Actions Create endpoint

Find resources by attribute or tag

<input checked="" type="checkbox"/>	Name	VPC endpoint ID	VPC ID	Service name	Endpoint type
<input checked="" type="checkbox"/>	ep	eice-0b7842ce650f4b91d	vpc-0b0392996214289e4 vpc1	eice-0b7842ce650f4b91d.1B42e51e.ec2-instance-comm...	EC2 Instance

eice-0b7842ce650f4b91d / ep

[Details](#) [Security Groups](#) [Subnets](#) [Tags](#)

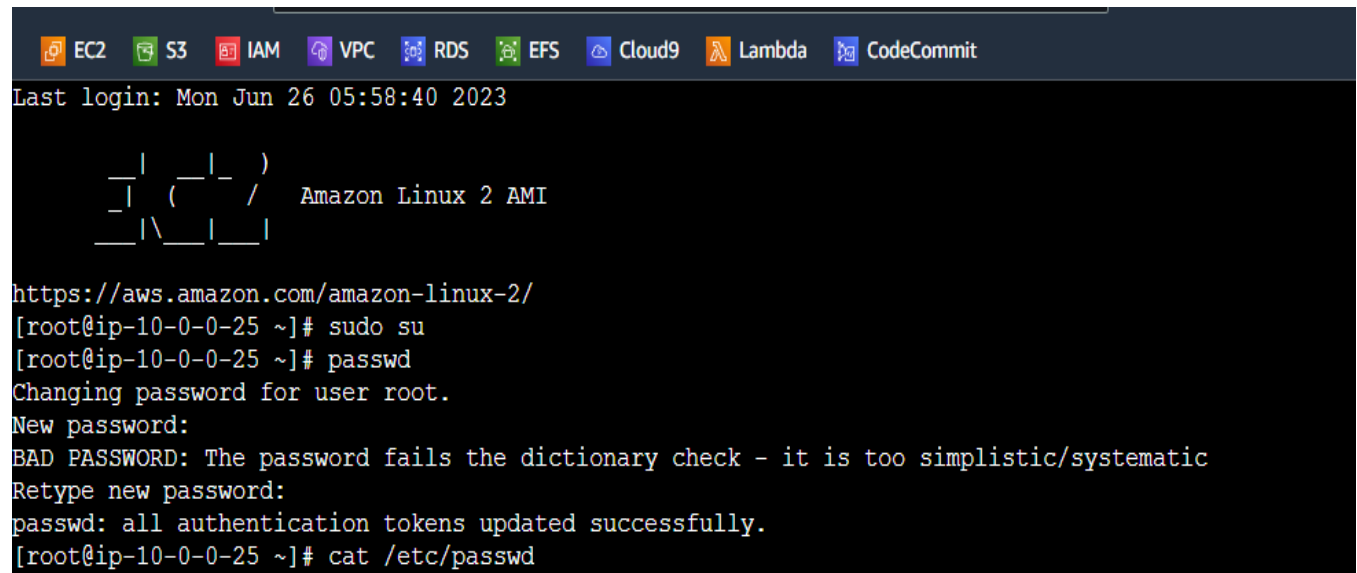
- Now go to instances, select the “autoscalinginstance” and click on connect.
- Click on connect using end point.
- Select the end point.
- Click on connect.

The screenshot shows the 'Connect to instance' page in the AWS Management Console. The page title is 'Connect to instance' with an 'Info' link. Below the title, it says 'Connect to your instance i-0ecf7a6361d357aad (autoscalinginstance) using any of these options'. There are four tabs: 'EC2 Instance Connect' (selected), 'Session Manager', 'SSH client', and 'EC2 serial console'. Under the 'EC2 Instance Connect' tab, the 'Instance ID' is 'i-0ecf7a6361d357aad (autoscalinginstance)'. The 'Connection Type' section has two options: 'Connect using EC2 Instance Connect' (unselected) and 'Connect using EC2 Instance Connect Endpoint' (selected). The 'Private IP address' is '10.0.0.25'. The 'User name' field contains 'root'. The 'Max tunnel duration (seconds)' field contains '3600'. The 'EC2 Instance Connect Endpoint' field contains 'eice-0b7842ce650f4b91d'.

- Instance is connected and AMI is launched.
- Auto scaling instance allows users to run commands, create files, and perform all kinds of computations very easily with less CPU utilization.
- We can also create a new user in the running instance and can run commands.

- Auto scaling instance allows users to run commands, create files, and perform all kinds of computations very easily with less CPU utilization.
- We can also create a new user in the running instance and can run commands.

Execution Results:



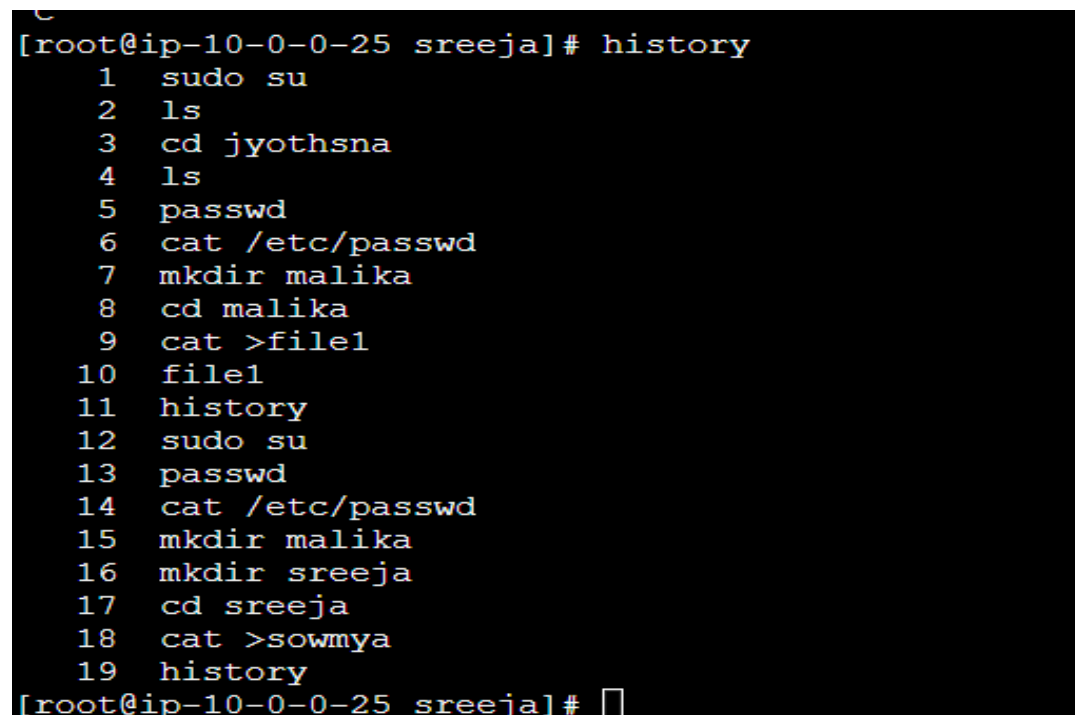
```

Last login: Mon Jun 26 05:58:40 2023

 _ _ | _ _ | _ _ )
 _ | ( _ _ /   Amazon Linux 2 AMI
 _ | \ _ _ | _ _ |

https://aws.amazon.com/amazon-linux-2/
[root@ip-10-0-0-25 ~]# sudo su
[root@ip-10-0-0-25 ~]# passwd
Changing password for user root.
New password:
BAD PASSWORD: The password fails the dictionary check - it is too simplistic/systematic
Retype new password:
passwd: all authentication tokens updated successfully.
[root@ip-10-0-0-25 ~]# cat /etc/passwd

```



```

C
[root@ip-10-0-0-25 sreeja]# history
 1  sudo su
 2  ls
 3  cd jyothsna
 4  ls
 5  passwd
 6  cat /etc/passwd
 7  mkdir malika
 8  cd malika
 9  cat >file1
10  file1
11  history
12  sudo su
13  passwd
14  cat /etc/passwd
15  mkdir malika
16  mkdir sreeja
17  cd sreeja
18  cat >sowmya
19  history
[root@ip-10-0-0-25 sreeja]# 

```


Conclusion:

This architect helps access EC2 private instances through EC2 public instances by using bastion host that provides security to a network from external sources and creating an Auto Scaling group which helps in creating instances to meet desired capacity. It maintains a few instances by performing periodic health checks on the instances in the group.