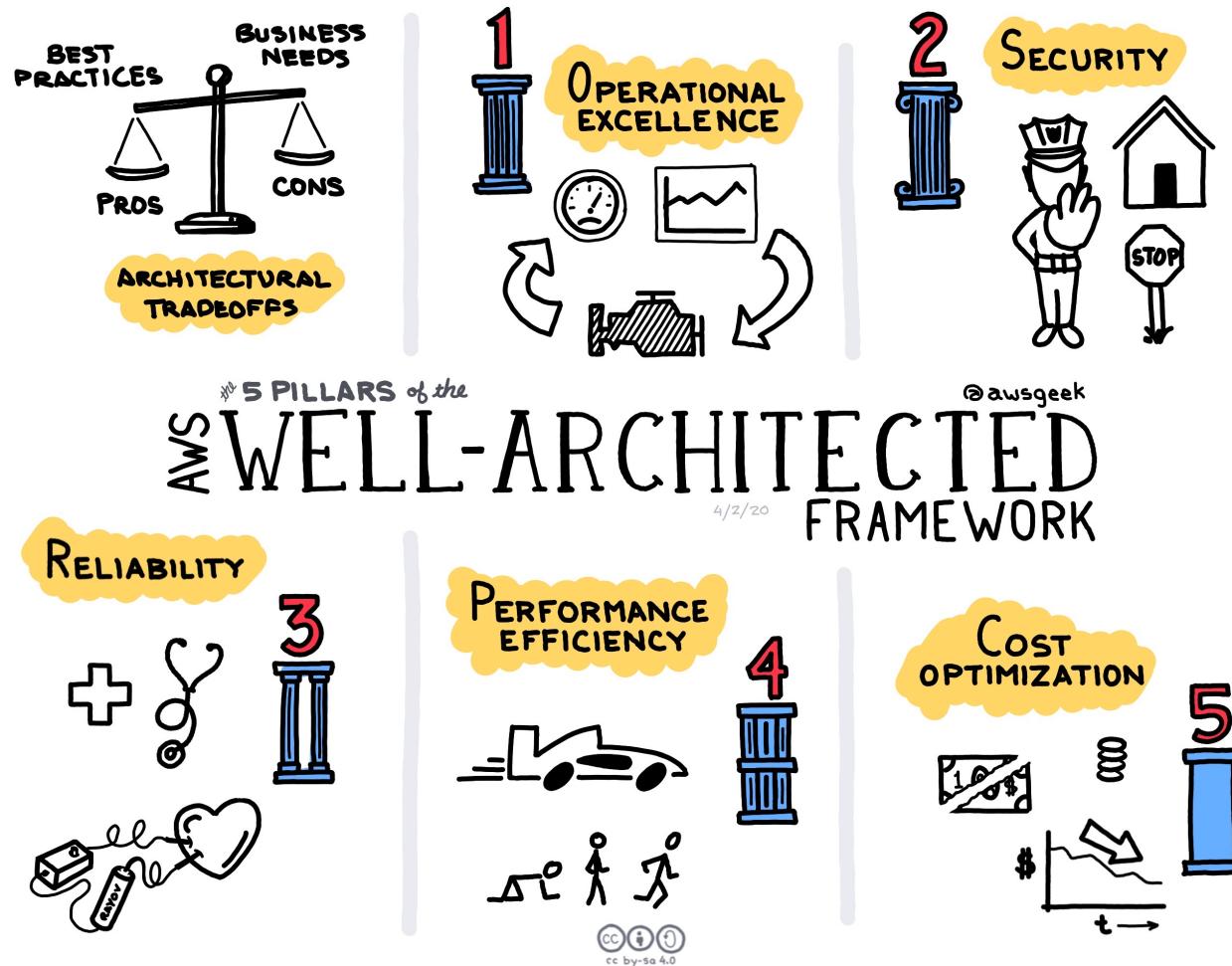


# AWS02: AWS Core Services

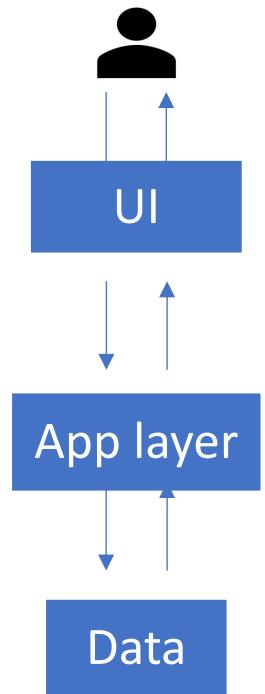


# Outline

- N-tiered Architecture of Web Applications
- AWS Three-tier Architecture
- Types of Core Services
  - Network: VPC
  - Compute: EC2, ELB, Auto Scaling
  - Monitoring- Cloudwatch
  - Storage: S3, EBS
  - Database- RDS (MySQL)
- AWS Three-tier Architecture using Core Services

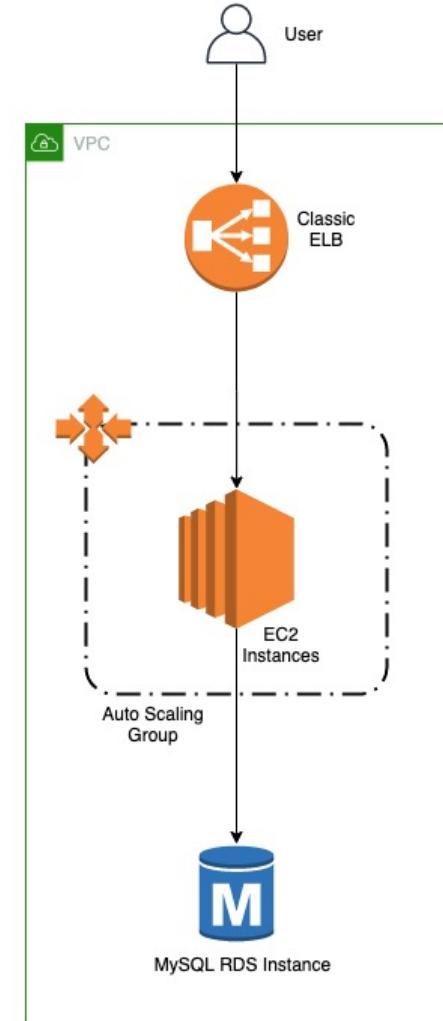
# N-tiered Architecture of Web Applications

- tiers based on resource functions
- advantages
  - Flexibility to add, remove or upgrade each tier independently
  - Easy scalability based on usage
- three-tiered architecture
  - Presentation (UI)
  - Application processing
  - Data storage and management
- layers interact using APIs

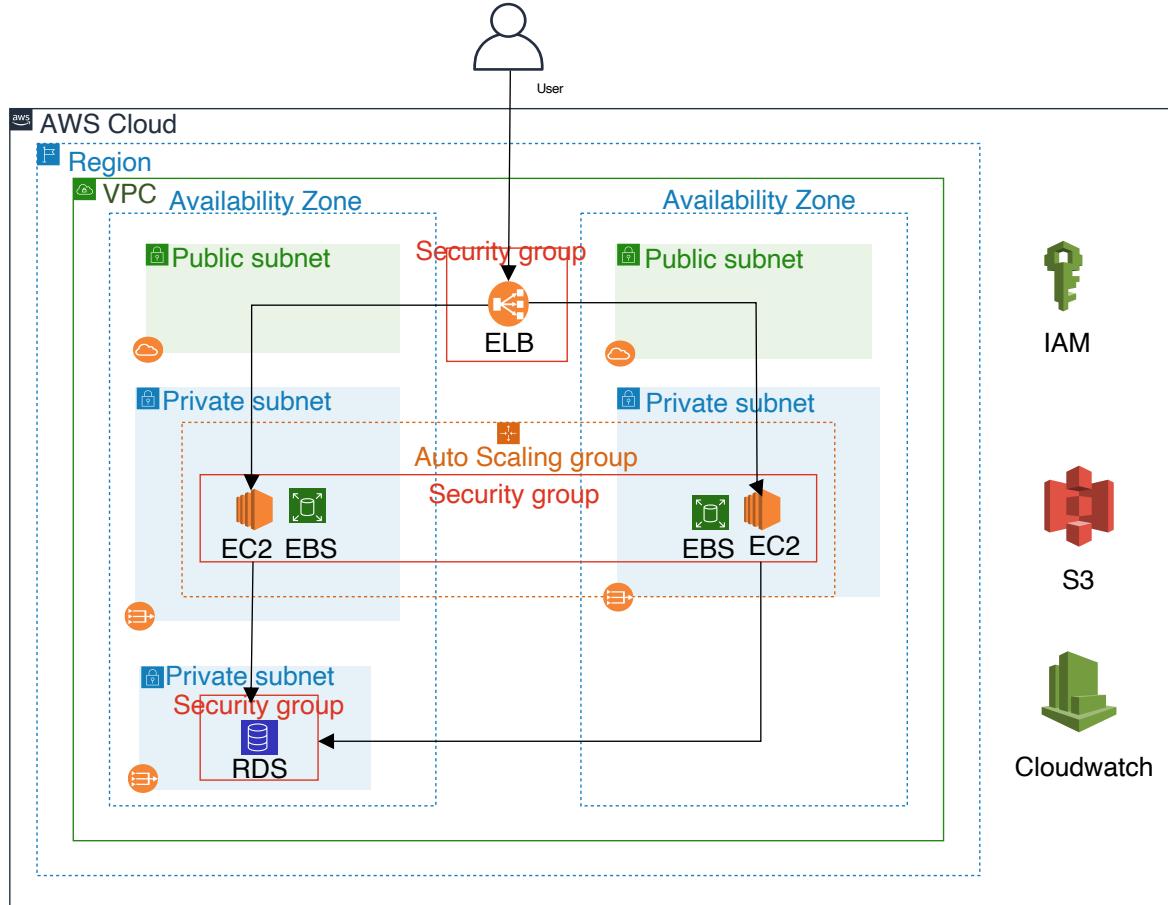


# AWS Three-tier Architecture

- Client
- Server
  - load balancer distributes client requests for high-availability
  - cluster of web (application) servers
    - serve user requests received from the load balancer
    - deployed as an Auto Scaling group
- Data -> database
  - Stores persistent data
  - Serves queries received from web servers



# AWS Three-tier Architecture using Core Services



## Components

- Network
- Compute & Storage
- Elastic Load Balancer
- MySQL RDS Database
- S3 bucket
- Monitoring: Cloudwatch

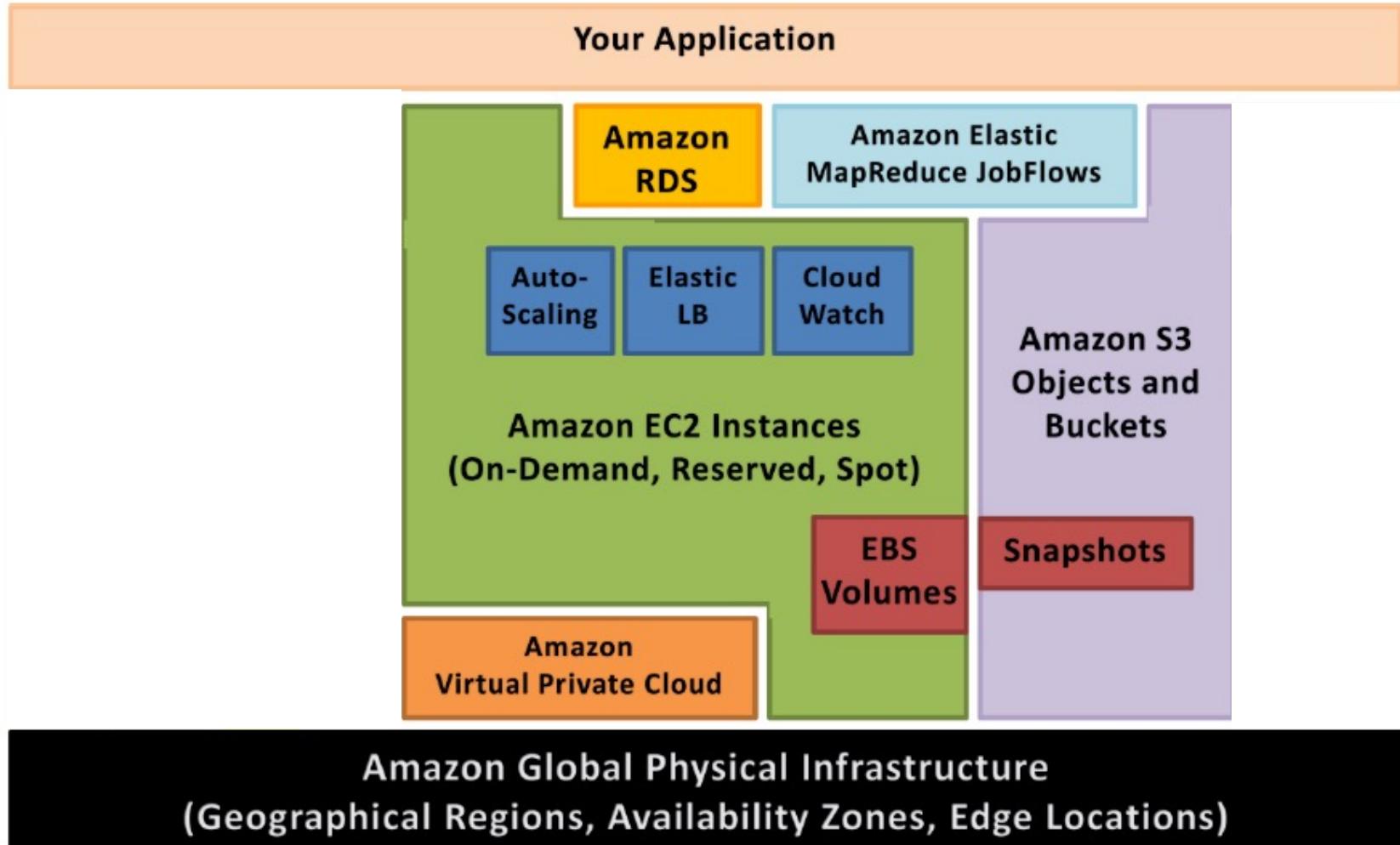
# Amazon Web Services (AWS) Ecosystem

Your Application

**Amazon Global Physical Infrastructure  
(Geographical Regions, Availability Zones, Edge Locations)**

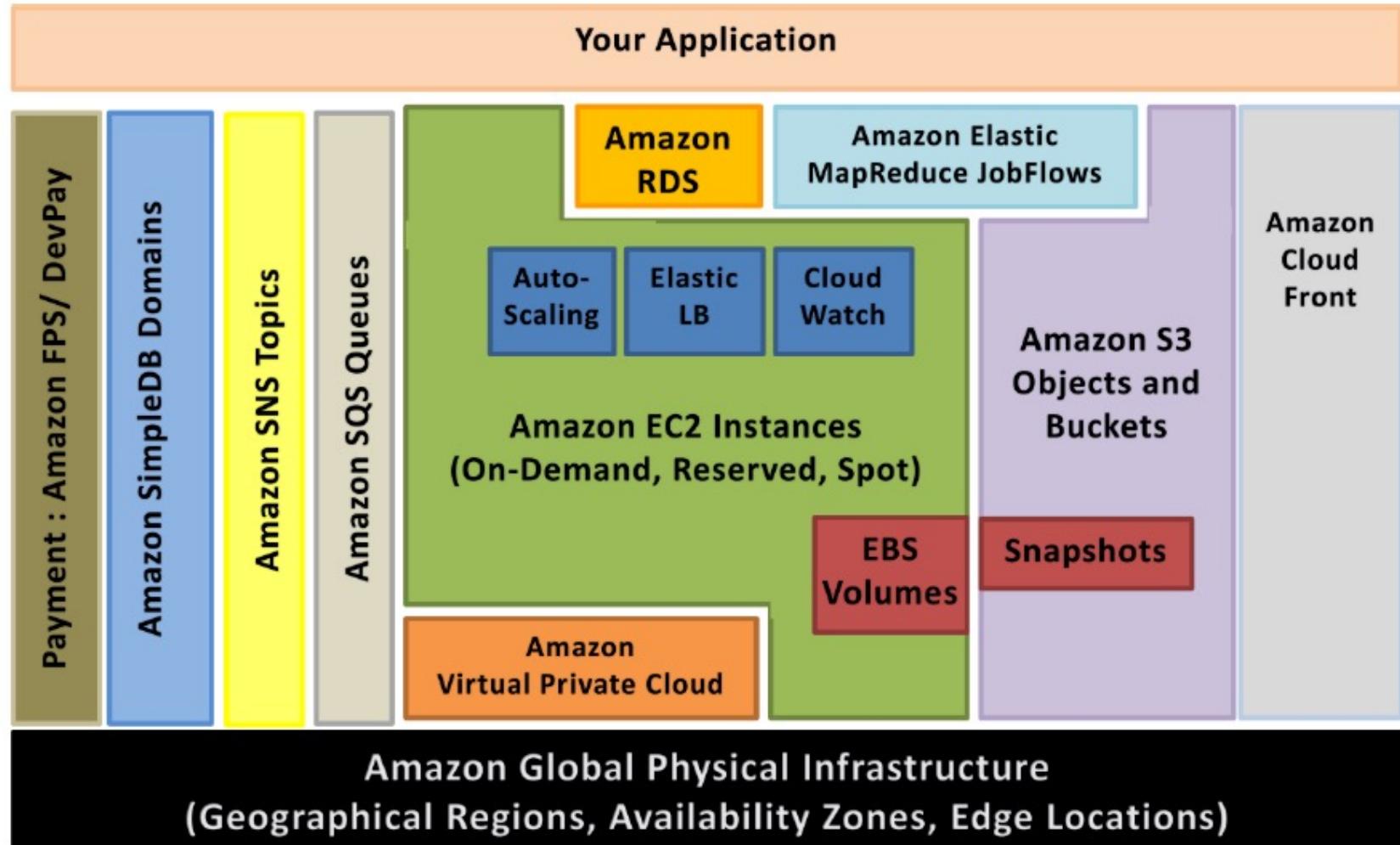
J. Varia, Architecting for the Cloud: Best Practices, Amazon Web Services, May 2010.

# Amazon Web Services (AWS) Ecosystem



J. Varia, Architecting for the Cloud: Best Practices, Amazon Web Services, May 2010.

# Amazon Web Services (AWS) Ecosystem



J. Varia, Architecting for the Cloud: Best Practices, Amazon Web Services, May 2010.

# Outline

- N-tiered Architecture of Web Applications
- AWS Three-tier Architecture
- Types of Core Services
  - Network: VPC
  - Compute: EC2, ELB, Auto Scaling
  - Monitoring- Cloudwatch
  - Storage: S3, EBS
  - Database- RDS (MySQL)
- AWS Three-tier Architecture using Core Services

# Network Services - VPC

- What is Virtual Private Cloud (VPC)?

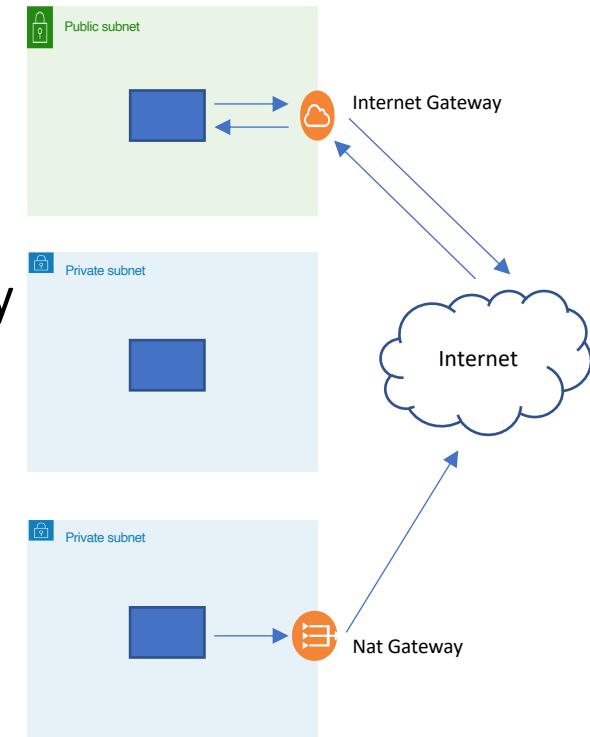
*"Amazon VPC enables you to launch Amazon Web Services (AWS) resources into a virtual network that you've defined. This virtual network closely resembles a traditional network that you'd operate in your own datacenter, with the benefits of using the scalable infrastructure of AWS"*

- Resembles an on-premise private corporate network
- In your VPC you can :
  - Define a CIDR block\*
- Create multiple public and/or private subnets
- Launch resources with your own private IP address
- Define security groups, Access Control Lists, and Route Tables
- No additional charge for using a VPC

\*read more - [www.digitalocean.com/community/tutorials/understanding-ip-addresses-subnets-and-cidr-notation-for-networking](http://www.digitalocean.com/community/tutorials/understanding-ip-addresses-subnets-and-cidr-notation-for-networking)

# Network Services - Subnets

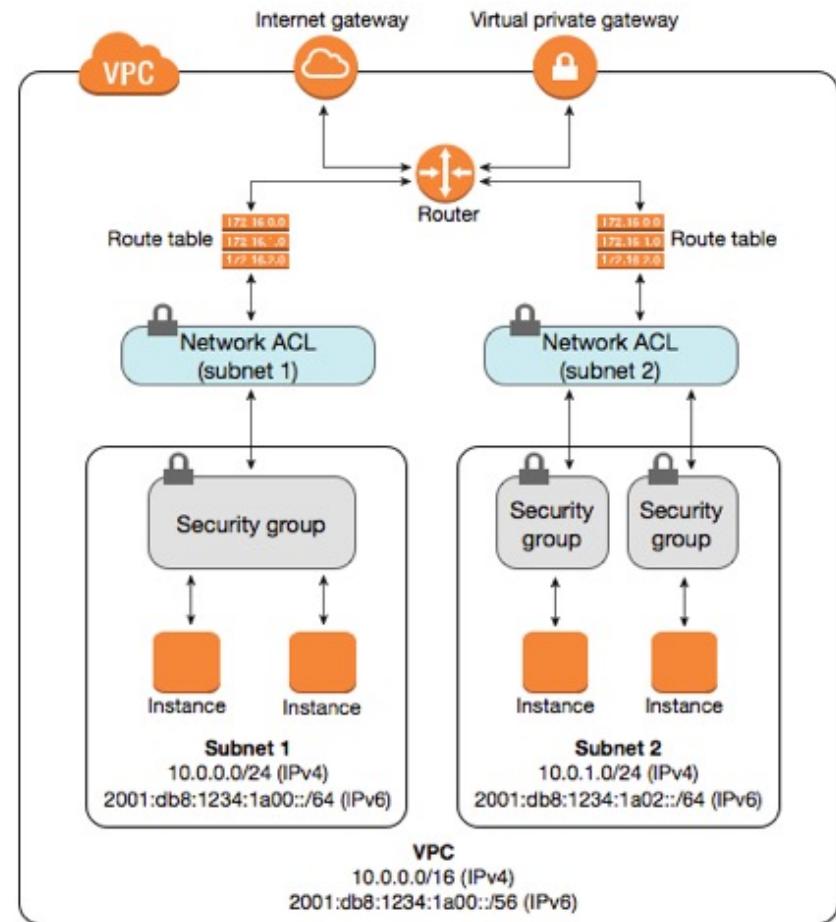
- Subnet is a range of IP addresses in a VPC, in one AZ
- Route table: Set of rules to determine where traffic is directed
- Types of Subnets:
  - Public Subnet
    - Network traffic directed to Internet Gateway
    - Resources have access to internet
    - Use for resources that are directly accessed
  - Private Subnet
    - By default, all new subnets are private
    - Resources do not have access to internet
    - To access internet, direct traffic to a NAT gateway



# Network Services - Security

Two main features for network security:

- Security Groups
  - Act as firewall for EC2 instance
- Network Access Control List (NACL)
  - Act as a firewall for subnets



# Outline

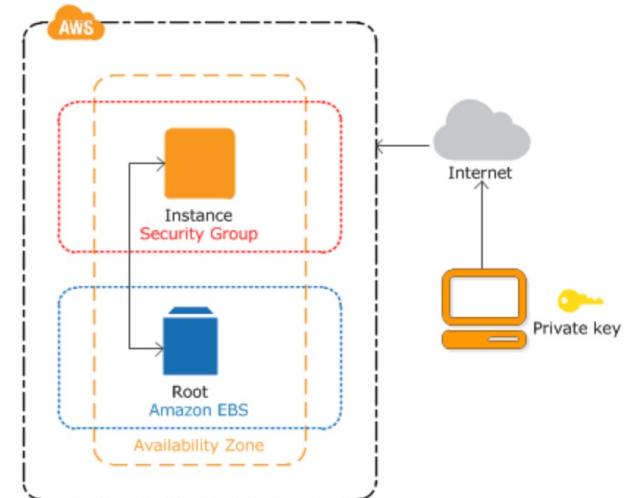
- N-tiered Architecture of Web Applications
- AWS Three-tier Architecture
- Types of Core Services
  - Network: VPC
  - Compute: EC2, ELB, Auto Scaling
  - Monitoring- Cloudwatch
  - Storage: S3, EBS
  - Database- RDS (MySQL)
- AWS Three-tier Architecture using Core Services

# Compute Services: EC2 Instances

- web service providing resizable computing capacity to build and host your software systems
- an **instance** is a virtual server (VM) with a specified set of resources including CPU cycles, main memory, secondary storage, communication and I/O bandwidth
- user chooses:
  - *region* and *availability zone* where virtual server should be placed
  - an *instance type* from a menu of instance types
- pricing: on-demand, reserved, spot

# EC2 Instances: Properties

- **Amazon Machine Image (AMI)**: Software configuration template (OS, and applications)
- Instance Types: Compute Optimized, Memory optimized, General Purpose, etc.
- Instance Sizes: Based on vCPUs, Memory and Network Performance
- EBS (Elastic Block Storage) Root/Secondary Volumes: For persistent storage
- Key Pair: For SSH (Linux) or RDP (Windows) access
- Network configurations:
  - VPC and Subnet
  - Public and Private IPs
- IAM Role: AWS credentials
- Security Groups



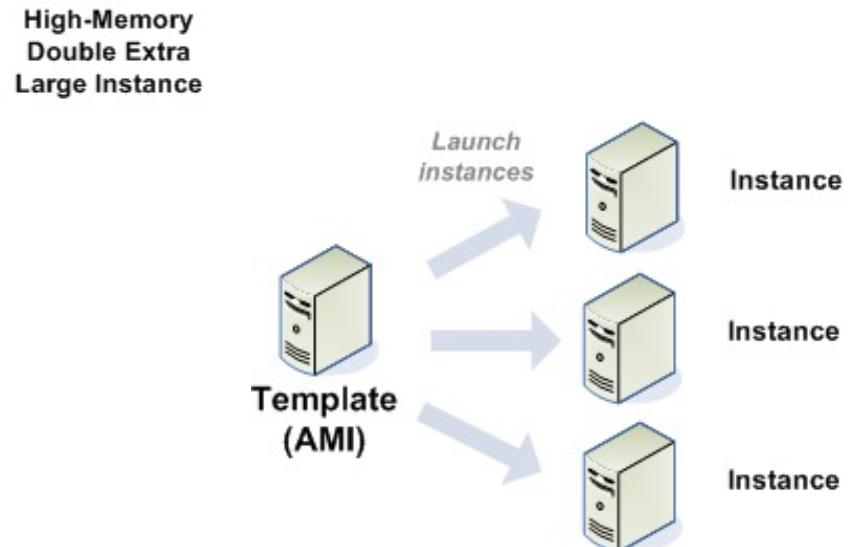
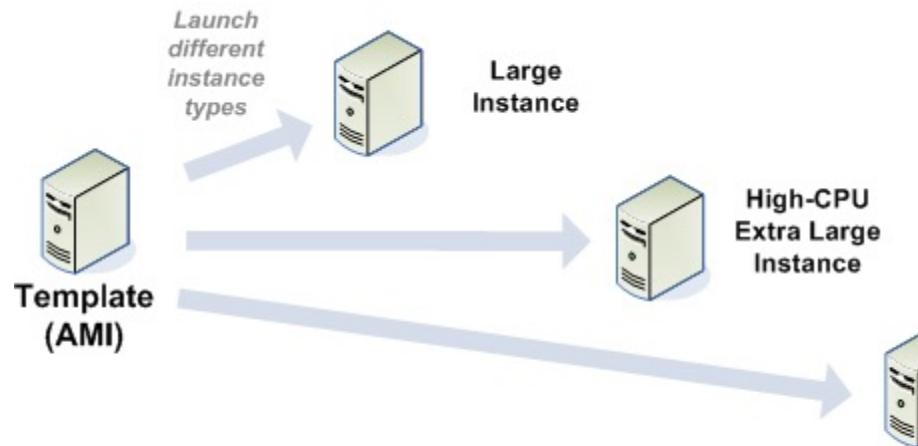
# Example: Instance Types

1. General Purpose: T2, M4
  2. Compute Optimized: C4, C5
  3. Memory Optimized: X1, R4
  4. Accelerated Computing: F1, P2
  5. Storage optimized: I3, D2
- 
- Each instance type is further divided into subtypes based on features like the number of virtual CPUs (**vCPUs**), memory availability etc
  - vCPU is defined as a hyper-thread of Intel Xeon E5-2670 v2 CPU (previously ECU (Elastic Compute Unit) – equivalent to CPU capacity of 2007 Opteron/Xeon processor)

# Example: Instance Types

|   | vCPU | ECU      | Memory (GiB) | Instance Storage (GB) | Linux/UNIX Usage  |
|---|------|----------|--------------|-----------------------|-------------------|
| <strong>General Purpose - Current Generation</strong> |      |          |              |                       |                   |
| t2.nano   | 1    | Variable | 0.5          | EBS Only              | \$0.0059 per Hour |
| t2.micro  | 1    | Variable | 1            | EBS Only              | \$0.012 per Hour  |
| t2.small  | 1    | Variable | 2            | EBS Only              | \$0.023 per Hour  |
| t2.medium   | 2    | Variable | 4            | EBS Only              | \$0.047 per Hour  |
| t2.large  | 2    | Variable | 8            | EBS Only              | \$0.094 per Hour  |
| t2.xlarge   | 4    | Variable | 16           | EBS Only              | \$0.188 per Hour  |
| t2.2xlarge  | 8    | Variable | 32           | EBS Only              | \$0.376 per Hour  |
| m4.large  | 2    | 6.5      | 8            | EBS Only              | \$0.1 per Hour    |
| m4.xlarge   | 4    | 13       | 16           | EBS Only              | \$0.2 per Hour    |
| m4.2xlarge  | 8    | 26       | 32           | EBS Only              | \$0.4 per Hour    |
| m4.4xlarge  | 16   | 53.5     | 64           | EBS Only              | \$0.8 per Hour    |
| m4.10xlarge   | 40   | 124.5    | 160          | EBS Only              | \$2 per Hour      |
| m4.16xlarge   | 64   | 188      | 256          | EBS Only              | \$3.2 per Hour    |

# EC2: AMI



# EC2 Instances: 3 Tasks

## 1. Launch an instance

- a launched instance is secured by specifying a key pair and security group
- to connect to your instance, you must specify the private key of the key pair that you specified when launching your instance

## 2. Connect to instance launched

- AWS management console, CLI

## 3. Terminate your instance

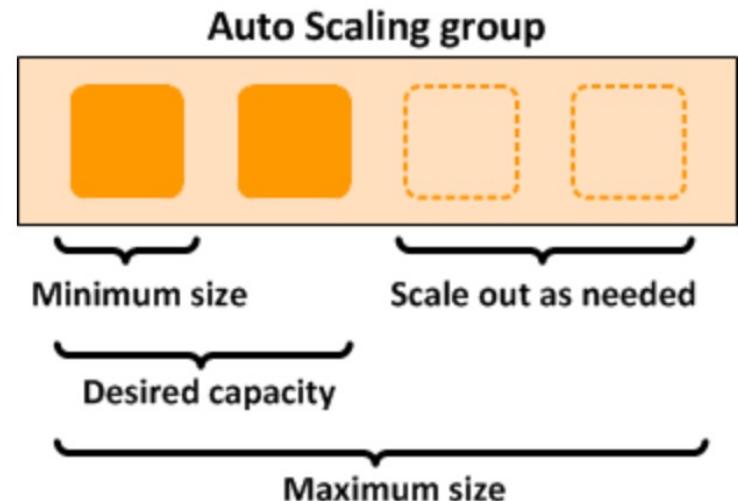
- deletes the instance and **stop incurring charges**

# AWS Instances & Network Services

- when launched, an instance is provided with a DNS name that maps to a
  - *private IP address* → for internal communication within the internal EC2 communication network
  - *public IP address* → for communication outside the internal Amazon network, e.g., communication with the user that launched the instance and assigned for the lifetime of an instance
- an instance can request an *elastic IP address* (rather than a public IP), a static public IP address allocated to an instance from the available pool of the availability zone
- an *elastic IP address* is not released when the instance is stopped or terminated but *released when no longer needed*

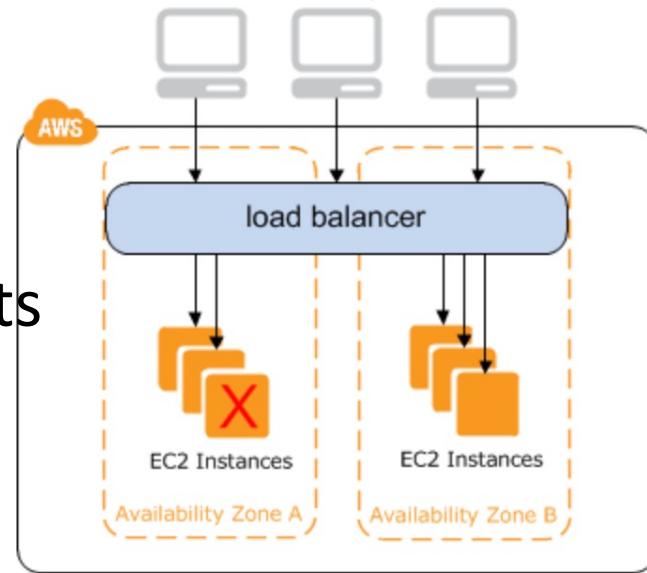
# Compute Services: Auto Scaling

- scale EC2 capacity automatically
- for applications with usage variability
- no additional charge for using an Auto Scaling group
- uses a template called “Launch Configuration” to create EC2 instances
- advantages
  - increased availability
  - increased fault-tolerance
  - better cost management



# Compute Services: Elastic Load Balancing

- Automatically distributes incoming traffic across targets
- Targets can be instances in one or more Availability Zones
- Single point of contact for the clients
- Routes requests only to healthy targets
- Advantages
  - Increases Fault-tolerance and availability
  - Targets can be added or removed without any disruption
  - Minimizes risk of overloading one server with requests
  - Supports operational monitoring and logging

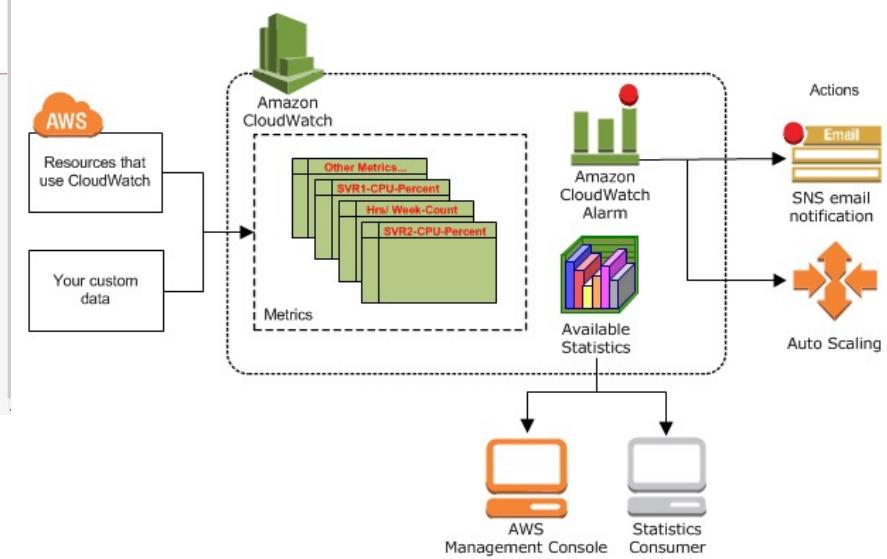
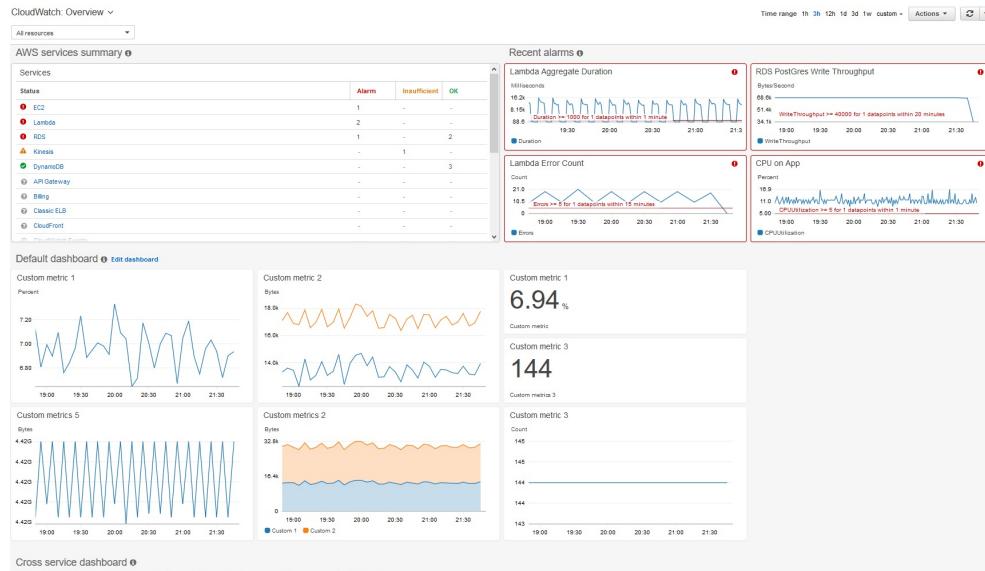


# Outline

- N-tiered Architecture of Web Applications
- AWS Three-tier Architecture
- Types of Core Services
  - Network: VPC
  - Compute: EC2, ELB, Auto Scaling
  - Monitoring- Cloudwatch
  - Storage: S3, EBS
  - Database- RDS (MySQL)
- AWS Three-tier Architecture using Core Services

# Monitoring Service: Cloudwatch

- Manage and monitor AWS services
- Collect and track metrics and create alerts
- Monitor custom metrics using API

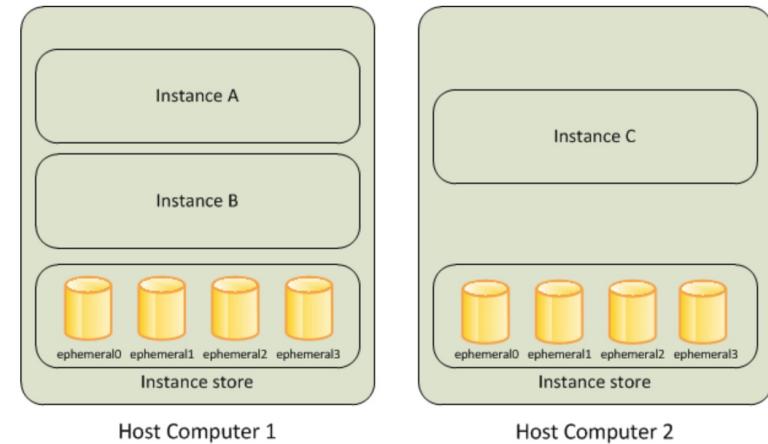


# Outline

- N-tiered Architecture of Web Applications
- AWS Three-tier Architecture
- Types of Core Services
  - Network: VPC
  - Compute: EC2, ELB, Auto Scaling
  - Monitoring- Cloudwatch
  - Storage: S3, EBS
  - Database- RDS (MySQL)
- AWS Three-tier Architecture using Core Services

# Storage Services: Instance Storage

- Disks physically attached to EC2 host instances
- Ideal for temporary storage that changes frequently
- Data lost once the instance is stop / terminated but persists during reboots
- Data doesn't persist in AMIs
- Size depends on the instance type
- Cost is included in instance cost
- Example:
  - buffers, caches, scratch data



# Storage Services

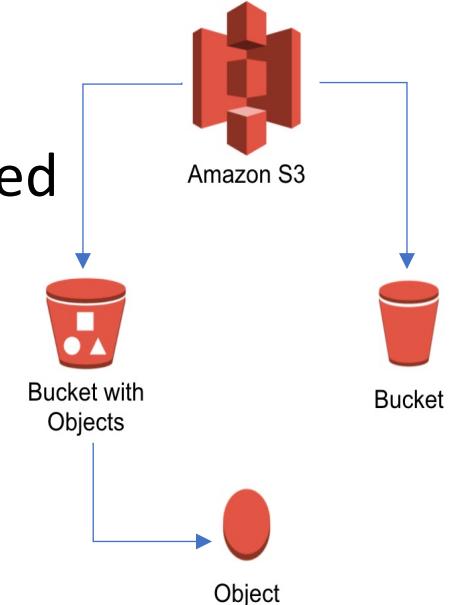
## EBS (Elastic Block Store) Volumes

- Persistent block level storage volumes for EC2 instances, e.g database applications, file systems and applications using raw data devices
- Automatically replicated within its AZ
- Can be attached to instance as root or secondary volume
- Advantages
  - Reliable, secure storage (Supports encryption optional )
  - Quickly Backup and restore using Incremental Snapshots
  - Offers the consistent and low-latency performance
  - Quickly scale up, easily scale down within minutes
  - Can be copied across various regions using snapshots
  - Paying a low price for only what you provision

# Storage Services: Simple Storage Service (S3)

- GLOBAL object-based storage service
- S3 buckets
  - objects are stored in buckets in a region selected by user and retrieved via unique names
  - set of functions: PUT (write), GET (read) and DELETE
  - User controls read/update access
- Advantages
  - Store/ retrieve any amount of data, anytime anywhere
  - Scalable, reliable, fast and durable
- Objects accessed through a URL:

<http://johnsmith.s3.amazonaws.com/photos/puppy.jpg>



# Outline

- N-tiered Architecture of Web Applications
- AWS Three-tier Architecture
- Types of Core Services
  - Network: VPC
  - Compute: EC2, ELB, Auto Scaling
  - Monitoring- Cloudwatch
  - Storage: S3, EBS
  - Database- RDS (**MySQL**)
- AWS Three-tier Architecture using Core Services

# Database Services

## Relational Database Service (RDS)

- Managed database service to set up, operate, and scale a relational database
- Basic building block is a DB instance
  - Isolated database environment
  - Contains multiple user-created databases
  - DB engines: MySQL, MariaDB, PostgreSQL, etc.
- Behavior controlled using Parameter Groups
- Available DB instance classes:
  - Standard: For general purpose workloads
  - Memory Optimized: For memory-intensive applications
  - Burstable Performance: provide a baseline performance level, with the ability to burst to full CPU usage
- Can be used in Multi-AZ deployment for High Availability

AWS manages the following features of your database:

Scaling

High availability

Database backups

DB s/w patches

DB s/w installs

OS patches

OS installation

Server maintenance

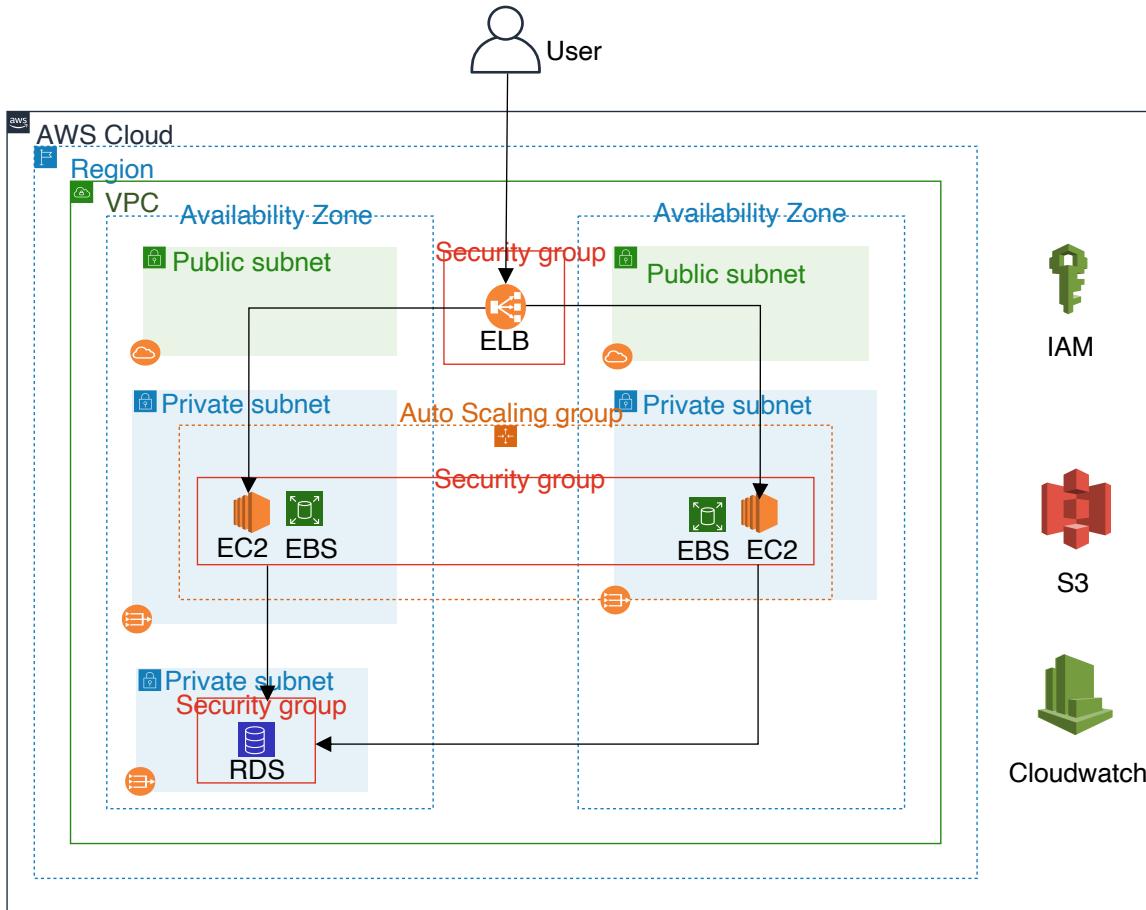
Rack & stack

Power, HVAC, net

# Outline

- N-tiered Architecture of Web Applications
- AWS Three-tier Architecture
- Types of Core Services
  - Network: VPC
  - Compute: EC2, ELB, Auto Scaling
  - Monitoring- Cloudwatch
  - Storage: S3, EBS
  - Database- RDS (MySQL)
- AWS Three-tier Architecture using Core Services

# AWS Three-tier Architecture using Core Services



## Steps

1. Select Region (us-east-1)
2. Create VPC
3. Create subnets in 2 AZs
  - 2 public subnets
  - 2 private subnets for web servers
  - 1 private DB subnet
4. Add IGW (Internet Gateway) and NAT (Network Address Translation Gateway)
5. Create Auto Scaling group (or 2 EC2 instances)
6. Create ELB
7. Create RDS MySQL Instance
8. Create S3 bucket
9. Monitor Cloudwatch metrics

# Useful AWS Links

- AWS documentation - [docs.aws.amazon.com](https://docs.aws.amazon.com)
- AWS knowledge center (FAQ) –  
[aws.amazon.com/premiumsupport/knowledge-center/](https://aws.amazon.com/premiumsupport/knowledge-center/)
- AWS discussion forums – [forums.aws.amazon.com/index.jspa](https://forums.aws.amazon.com/index.jspa)
- 12-month free tier – [aws.amazon.com/free](https://aws.amazon.com/free)
- Pricing - [aws.amazon.com/ec2/pricing](https://aws.amazon.com/ec2/pricing)
- AWS simple monthly calculator – [calculator.s3.amazonaws.com/index.html](https://calculator.s3.amazonaws.com/index.html)
- Total cost of ownership calculator – [aws.amazon.com/tco-calculator](https://aws.amazon.com/tco-calculator)
- AWS global infrastructure - [aws.amazon.com/about-aws/global-infrastructure/](https://aws.amazon.com/about-aws/global-infrastructure/)

# References

- **Reading**

- AWS® Certified Solution Architect Study Guide, 2017
- AWS 12-month Free Tier, [aws.amazon.com/free](http://aws.amazon.com/free)

- **User Guides**

- Network: [Working with VPCs](#)
- EC2 Instances: [Get Started with EC2](#)
- Auto Scaling: [Get Started with Auto Scaling](#)
- ELB: [Increase EC2 Availability](#)
- RDS: [Get Started with RDS](#)
- S3: [Backup files to Amazon S3](#)
- Cloudwatch: [Get Started with Cloudwatch](#)

- **Additional Readings**

- AWS CLI: [AWS CLI](#)
- Host static website on S3: [Static Website Hosting](#)
- AWS Solution Implementations: [Implementations](#)
- Security Groups vs NACL: [SG vs NACL](#)

# AWS02: AWS Core Services (Lab)



# Outline

- Objective
- Main Steps and Milestones
  - Network - VPC
  - Compute - EC2
  - Elastic Load Balancer
  - RDS DB instance
  - Cloudwatch Metrics
  - Autoscaling Group
- Cleanup
- Summary
- References

# Objectives

Setup three-tier architecture on AWS

- Milestone 1: Network
  - Create VPC
  - Create 5 subnets in 2 AZs
  - Create Internet Gateway in each AZ
  - Create NAT Gateway in each AZ
  - Configure Routing tables for subnets
- Milestone 2: Compute & Load Balancing
  - Create one EC2 instance in each AZ
  - Configure Elastic Load Balancer
- Milestone 3: Database and Scaling
  - Configure RDS MySQL DB Instance
  - Observe Cloudwatch metrics
  - Add Auto Scaling Group

# Main Steps - Network

- Sign in
- Select AWS Service
- Select Region
- Create VPC
- Create subnets
- Create Internet Gateway
- Create NAT Gateway
- Add to subnet routing tables

# **Step 1: Sign-in to AWS console**

**Use the IAM user account we have setup for you!**

# Step 2: Select VPC Service

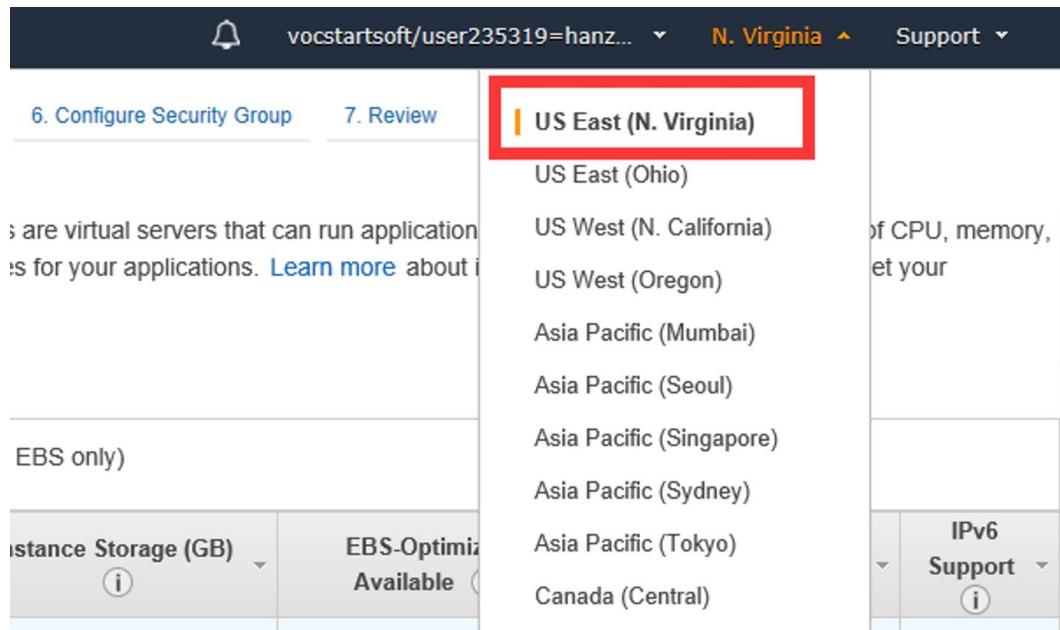
AWS Management Console

The screenshot shows the AWS Management Console interface. At the top, it says "AWS services". Below that, under "Recently visited services", there is a list of services: VPC, EC2, DynamoDB, CloudWatch, IAM, and RDS. The "VPC" service is highlighted with a red rectangular box around its icon and name. At the bottom of the list, there is a link "▶ All services".

| AWS services  |  |   |  |  |  |
|---|--|---|--|--|--|
| ▼ Recently visited services   |  |   |  |  |  |
|  <b>VPC</b>          |  <a href="#">DynamoDB</a>   |  <a href="#">IAM</a> |  |  |  |
|  <a href="#">EC2</a> |  <a href="#">CloudWatch</a> |  <a href="#">RDS</a> |  |  |  |
| ▶ All services  |  |   |  |  |  |

# Step 3: Select Region

- Top right of the menu bar, between username and help:
- Choose US East (N. Virginia)



# Step 4.1: Create VPC

## View Existing default VPC

The screenshot shows the AWS VPC Dashboard. On the left, there's a sidebar with navigation links for VPCs, Cloud, Security, Reachability, and AWS Network Firewall. The main area displays 'Resources by Region' for the US East (N. Virginia) region. A red box highlights the 'VPCs' section, which shows 1 VPC across all regions. Other resource counts are: Subnets (6), Route Tables (1), Internet Gateways (1), Egress-only Internet Gateways (0), DHCP options sets (1), Elastic IPs (0), Endpoints (0), Endpoint Services (0), NAT Gateways (0), VPC Peering Connections (0), Network ACLs (1), Customer Gateways (0), Virtual Private Gateways (0), Site-to-Site VPN Connections (0), and Running Instances (0).

| Resource Type                 | Count | Region      |
|-------------------------------|-------|-------------|
| VPCs                          | 1     | N. Virginia |
| Subnets                       | 6     | N. Virginia |
| Route Tables                  | 1     | N. Virginia |
| Internet Gateways             | 1     | N. Virginia |
| Egress-only Internet Gateways | 0     | N. Virginia |
| DHCP options sets             | 1     | N. Virginia |
| Elastic IPs                   | 0     | N. Virginia |
| Endpoints                     | 0     | N. Virginia |
| Endpoint Services             | 0     | N. Virginia |
| NAT Gateways                  | 0     | N. Virginia |
| Peering Connections           | 0     | N. Virginia |
| Network ACLs                  | 1     | N. Virginia |
| Customer Gateways             | 0     | N. Virginia |
| Virtual Private Gateways      | 0     | N. Virginia |
| Site-to-Site VPN Connections  | 0     | N. Virginia |
| Running Instances             | 0     | N. Virginia |
| Endpoint Services             | 0     | N. Virginia |

# Step 4.2: Create VPC

- View Existing default VPC

The screenshot shows the AWS VPC Dashboard. On the left, there's a sidebar with options like 'New VPC Experience', 'VPC Dashboard', and a list under 'VIRTUAL PRIVATE CLOUD' including 'Your VPCs', 'Subnets', 'Route Tables', etc. In the center, there are two buttons: 'Launch VPC Wizard' and 'Launch EC2 Instances'. Below them, a note says 'Note: Your Instances will launch in the US East (N. Virginia) region.' A section titled 'Resources by Region' shows various metrics: VPCs (1), NAT Gateways (0), Subnets (6), VPC Peering Connections (0), Route Tables (1), Network ACLs (1), Internet Gateways (1), and Security Groups (5). The 'VPCs' row is highlighted with a red box.

- Create VPC

The screenshot shows the 'Create VPC' wizard. It has a header with 'Actions' and a 'Create VPC' button, which is highlighted with a red box. Below is a table with columns: PC ID, State, IPv4 CIDR, IPv6 CIDR (Network border group), and IPv6 pool. There is one entry: 'oc-e5d4fa83' with 'Available' state, '172.31.0.0/16' CIDR, and '-' for IPv6 fields.

| PC ID       | State     | IPv4 CIDR     | IPv6 CIDR (Network border group) | IPv6 pool |
|-------------|-----------|---------------|----------------------------------|-----------|
| oc-e5d4fa83 | Available | 172.31.0.0/16 | -                                | -         |

# Step 4.3: Create VPC

- Configure Name
- Configure IPv4 VPC CIDR
- Create

Create VPC [Info](#)

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

**VPC settings**

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

IPv4 CIDR block [Info](#)

IPv6 CIDR block [Info](#)  
 No IPv6 CIDR block  
 Amazon-provided IPv6 CIDR block  
 IPv6 CIDR owned by me

Tenancy [Info](#)

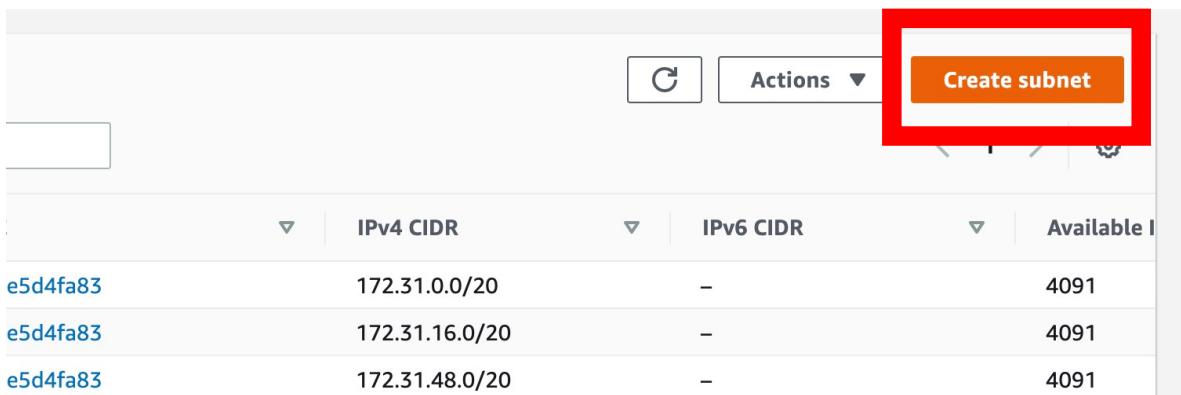
**Tags**  
A tag is a label that you assign to an AWS resource. Each tag consists of a key and an optional value. You can use tags to search and filter your resources or track your AWS costs.

| Key   | Value - <i>optional</i>  |
|---|--|
| <input type="text" value="Name"/> <a href="#">X</a> | <input type="text" value="my-vpc"/> <a href="#">X</a> <a href="#">Remove</a> |

[Add new tag](#)  
You can add 49 more tags.

[Cancel](#) [Create VPC](#)

# Step 5.1: Create Subnets



A screenshot of the AWS CloudFormation console showing a list of subnets. The interface includes a top navigation bar with a refresh icon, an 'Actions' dropdown, and a prominent orange 'Create subnet' button highlighted with a red box. Below the header is a search bar. The main area displays a table with columns: IP4 CIDR, IP6 CIDR, and Available I. Three rows of subnet data are listed:

|          | IPv4 CIDR      | IPv6 CIDR | Available I |
|----------|----------------|-----------|-------------|
| e5d4fa83 | 172.31.0.0/20  | -         | 4091        |
| e5d4fa83 | 172.31.16.0/20 | -         | 4091        |
| e5d4fa83 | 172.31.48.0/20 | -         | 4091        |

# Step 5.2: Create Subnets

- Choose VPC
- Configure Name, AZ and CIDR and Create

The screenshot shows two side-by-side configuration panels for creating a subnet.

**Create subnet** (Left Panel):

- VPC**: A dropdown menu is open, showing "VPC ID" and "Create subnets in this VPC." Below it, the selected VPC ID is "vpc-09b23f87f1ca7c8d5 (my-vpc)".
- Associated VPC CIDRs**: Shows "IPv4 CIDRs" and "10.0.0.0/16".

**Subnet settings** (Right Panel):

**Subnet 1 of 1**

- Subnet name**: "my-public-subnet-01" (highlighted with a red box).
- Availability Zone**: "US East (N. Virginia) / us-east-1a" (highlighted with a red box).
- IPv4 CIDR block**: "10.0.0.1/24" (highlighted with a red box).
- Tags - optional**: A table with one tag: "Name" (key) and "my-public-subnet-01" (value). Buttons for "Add new tag" and "Remove" are shown.

**Actions**: Buttons for "Cancel" and "Create subnet" (highlighted with a red box).

# Step 5.3: Create Subnets

Create more subnets with the following configuration settings:

| Subnet no | VPC    | Name                 | AZ         | CIDR        |
|-----------|--------|----------------------|------------|-------------|
| 1         | my-vpc | my-public-subnet-01  | us-east-1a | 10.0.0.0/24 |
| 2         | my-vpc | my-public-subnet-02  | us-east-1b | 10.0.1.0/24 |
| 3         | my-vpc | my-private-subnet-01 | us-east-1a | 10.0.2.0/24 |
| 4         | my-vpc | my-private-subnet-02 | us-east-1b | 10.0.3.0/24 |
| 5         | my-vpc | my-db-subnet-01      | us-east-1a | 10.0.4.0/24 |

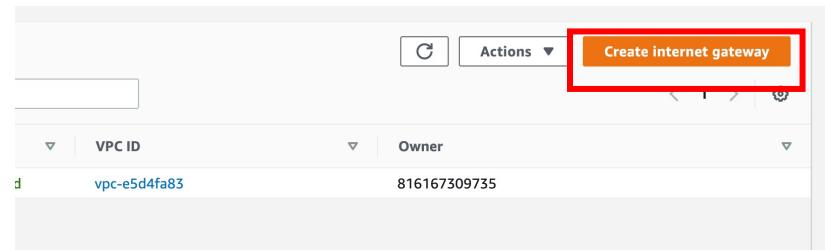
# Step 5.4: Create Subnets

Verify the subnets created ( filter by vpc)

| Subnets (5) <a href="#">Info</a>   |                   |                       |           |                         |             |         |                          |                   |
|--|-------------------|-----------------------|-----------|-------------------------|-------------|---------|--------------------------|-------------------|
| <input type="button" value="C"/> Actions ▾ <a href="#">Create subnet</a> |                   |                       |           |                         |             |         |                          |                   |
| <input type="button" value="Filter subnets"/>                            |                   |                       |           |                         |             |         |                          |                   |
| <input type="checkbox"/>   | Name              | Subnet ID             | State     | VPC                     | IPv4 Cl...  | IPv6... | Available IPv4 addresses | Availability Zone |
| <input type="checkbox"/>   | my-private-sub... | subnet-091275dcf...   | Available | vpc-084c62a7e74cc182... | 10.0.2.0/24 | -       | 251                      | us-east-1a        |
| <input type="checkbox"/>   | my-public-sub...  | subnet-0a61031ea6...  | Available | vpc-084c62a7e74cc182... | 10.0.1.0/24 | -       | 251                      | us-east-1b        |
| <input type="checkbox"/>   | my-public-sub...  | subnet-00483e7a3...   | Available | vpc-084c62a7e74cc182... | 10.0.0.0/24 | -       | 251                      | us-east-1a        |
| <input type="checkbox"/>   | my-private-sub... | subnet-046abb3f84...  | Available | vpc-084c62a7e74cc182... | 10.0.3.0/24 | -       | 251                      | us-east-1b        |
| <input type="checkbox"/>   | my-db-subnet-01   | subnet-0f3fcf793ea... | Available | vpc-084c62a7e74cc182... | 10.0.4.0/24 | -       | 251                      | us-east-1a        |

# Step 6.1: Create Internet Gateway

- Create Internet Gateway



- Configure Gateway Name

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.

**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   |
|-----------------------------------|--|
| <input type="text" value="Name"/> | <input type="text" value="my-internet-gateway-01"/> <a href="#">Remove</a> |

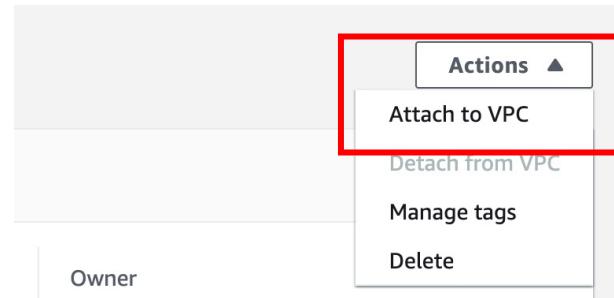
[Add new tag](#)  
You can add 49 more tags.

[Cancel](#) **Create internet gateway**

# Step 6.2: Create Internet Gateway

## Attach Internet Gateway to VPC

- Click on Actions
- Select “Attach to VPC”
- Choose VPC



# Step 7: Create NAT Gateway

- Configure NAT Gateway for each private subnet:
- Configure Name and subnet as:

| NAT Gateway Name  | Subnet               |
|-------------------|----------------------|
| My-nat-gateway-01 | My-private-subnet-01 |
| My-nat-gateway-02 | My-private-subnet-02 |
| My-nat-gateway-03 | My-db-subnet-01      |

- Allocate Elastic IP
- Create

The screenshot shows the 'Create NAT gateway' wizard in the AWS Management Console. The 'NAT gateway settings' section is visible, containing fields for 'Name - optional' (set to 'my-nat-gateway-01'), 'Subnet' (set to 'subnet-0e9d6a2747c15fa97 (my-private-subnet-01)'), and 'Elastic IP allocation ID' (set to 'Select an Elastic IP'). The 'Allocate Elastic IP' button is also highlighted with a red box. The 'Tags' section shows a single tag ('Name' key, 'my-nat-gateway-01 value') and the 'Create NAT gateway' button at the bottom is also highlighted with a red box.

# Step 8.1.1: Add to public subnet routing tables

Route Tables of Public subnets :

- Select a subnet
- View Route Table and click on the Route table
- Note: This is the default route table, which is currently attached to all subnets. For private subnets, we will create new route tables.

| Name                 | Subnet ID                | State     | VPC                           | IPv4 CIDR   | IPv6 CIDR |
|----------------------|--------------------------|-----------|-------------------------------|-------------|-----------|
| my-private-subnet-01 | subnet-091275dcbf6af268f | Available | vpc-084c62a7e74cc182e   my... | 10.0.2.0/24 | -         |
| my-public-subnet-02  | subnet-0a61031ea67702163 | Available | vpc-084c62a7e74cc182e   my... | 10.0.1.0/24 | -         |
| my-public-subnet-01  | subnet-00483e7a35980936d | Available | vpc-084c62a7e74cc182e   my... | 10.0.0.0/24 | -         |
| my-private-subnet-02 | subnet-046abb3f8431952d4 | Available | vpc-084c62a7e74cc182e   my... | 10.0.3.0/24 | -         |
| my-db-subnet-01      | subnet-0f3fcf793eaf96eb4 | Available | vpc-084c62a7e74cc182e   my... | 10.0.4.0/24 | -         |

subnet-00483e7a35980936d / my-public-subnet-01

Details | Flow logs | **Route table** | Network ACL | CIDR reservations | Sharing | Tags

Route table: **rtb-0b82412de837e61ff** | Edit route table association

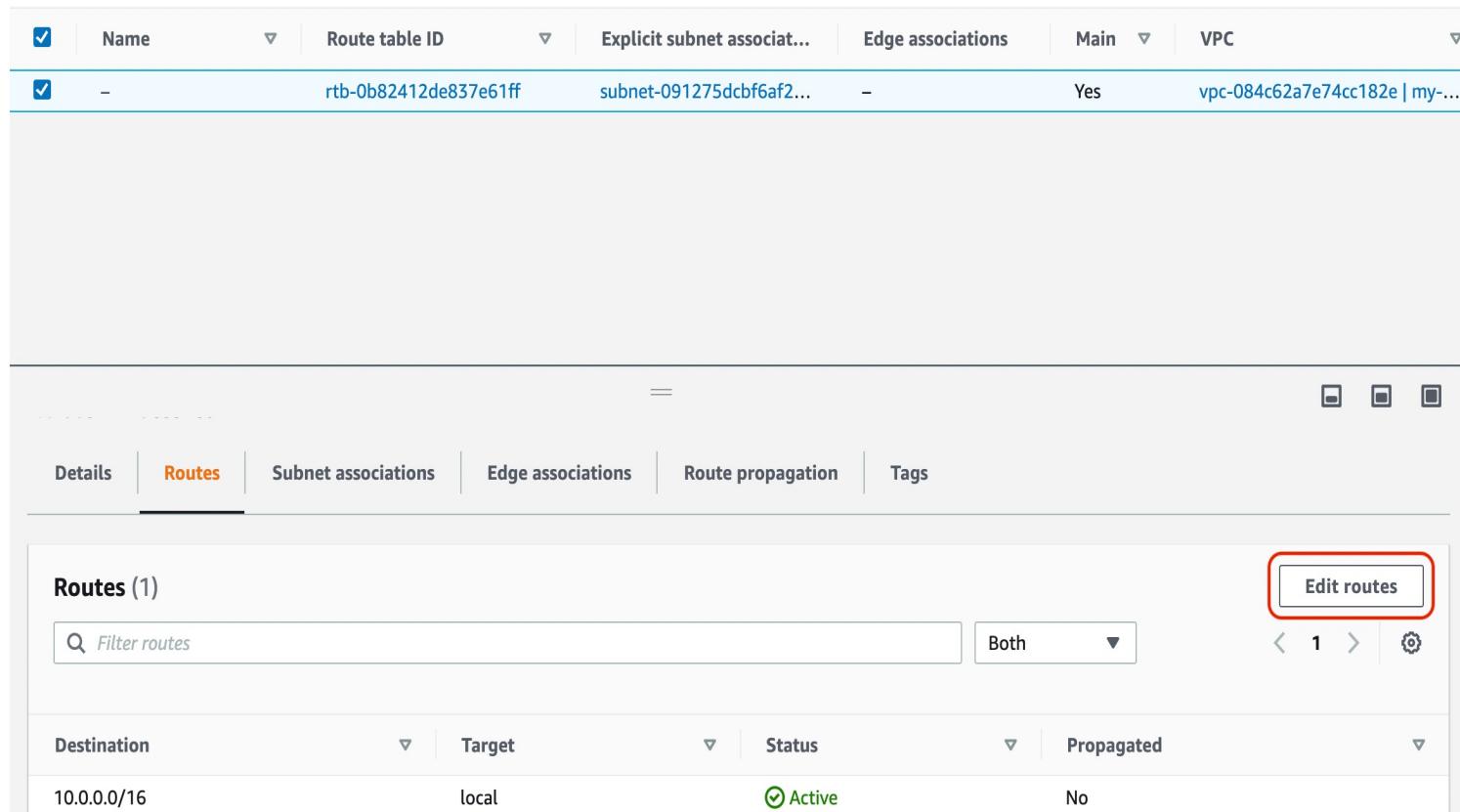
Routes (1)

Filter routes

< 1 > ⌂

# Step 8.1.2: Add to public subnet routing tables

- Click on Edit routes



| Name | Route table ID        | Explicit subnet associat... | Edge associations | Main | VPC                         |
|------|-----------------------|-----------------------------|-------------------|------|-----------------------------|
| -    | rtb-0b82412de837e61ff | subnet-091275dcbf6af2...    | -                 | Yes  | vpc-084c62a7e74cc182e   my- |

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

**Routes (1)**

Filter routes Both < 1 > ⚙️

| Destination | Target | Status | Propagated |
|-------------|--------|--------|------------|
| 10.0.0.0/16 | local  | Active | No         |

# Step 8.1.3: Add to public subnet routing tables

- Add route for all traffic to the internet gateway

Edit routes

| Destination   | Target  | Status                                     | Propagated             |
|---|---|--|------------------------|
| 10.0.0.0/16   | <input type="text" value="local"/> <span>X</span>                 | <input checked="" type="checkbox"/> Active | No                     |
| <input type="text" value="0.0.0.0/0"/> <span>X</span> | <input type="text" value="igw-0bf9a4bb604fa23d8"/> <span>X</span> | <input checked="" type="checkbox"/> Active | No <span>Remove</span> |
| <span>Add route</span>                                |   |  |                        |

# Step 8.1.4: Add to subnet routing tables

For the route table of the public subnet:

- Click on Edit subnet associations

The screenshot shows the AWS Route Table configuration page for route table ID `rtb-00a8d637b263f5372`. The 'Subnet associations' tab is selected. The 'Explicit subnet associations' section is empty, displaying the message 'No subnet associations' and 'You do not have any subnet associations.' A red box highlights the 'Edit subnet associations' button.

| Subnet ID                                | IPv4 CIDR | IPv6 CIDR |
|--|-----------|-----------|
| No subnet associations                   |           |           |
| You do not have any subnet associations. |           |           |

# Step 8.1.5: Add to subnet routing tables

For the route table of the public subnet:

- Associate the two public subnets with the route table.

The screenshot shows the 'Edit subnet associations' page for a specific route table. At the top, the breadcrumb navigation shows: VPC > Route tables > rtb-00a8d637b263f5372 > Edit subnet associations. The main section is titled 'Edit subnet associations' with the sub-instruction 'Change which subnets are associated with this route table.' Below this is a table titled 'Available subnets (2/5)'.

| Name  | Subnet ID                | IPv4 CIDR   | IPv6 CIDR | Route table ID               |
|---|--------------------------|-------------|-----------|------------------------------|
| <input checked="" type="checkbox"/> my-public-subnet-02 | subnet-0123dea1c219e73a8 | 10.0.1.0/24 | -         | Main (rtb-00a8d637b263f5372) |
| <input type="checkbox"/> my-db-subnet-01                | subnet-0fd834aceb104d66c | 10.0.4.0/24 | -         | Main (rtb-00a8d637b263f5372) |
| <input checked="" type="checkbox"/> my-public-subnet-01 | subnet-02eb43f71a8623443 | 10.0.0.0/24 | -         | Main (rtb-00a8d637b263f5372) |
| <input type="checkbox"/> my-private-subnet-01           | subnet-0c03134cf401550c7 | 10.0.2.0/24 | -         | Main (rtb-00a8d637b263f5372) |
| <input type="checkbox"/> my-private-subnet-02           | subnet-0c8cad422b2aa9b60 | 10.0.3.0/24 | -         | Main (rtb-00a8d637b263f5372) |

Below the table is a section titled 'Selected subnets' containing two entries: 'subnet-02eb43f71a8623443 / my-public-subnet-01' and 'subnet-0123dea1c219e73a8 / my-public-subnet-02'. At the bottom right are 'Cancel' and 'Save associations' buttons.

# Step 8.2.1: Add to private subnet routing tables

- Route Tables of 3 Private subnets :
  - Click on Create route table

The screenshot shows the AWS VPC Route Tables page. On the left, there's a sidebar with 'New VPC Experience' and 'VPC Dashboard'. Under 'VIRTUAL PRIVATE CLOUD', it lists 'Your VPCs' and 'Subnets', with 'Route Tables' currently selected. The main area has a title 'Route tables (2) Info' with a 'Create route table' button highlighted by a red box. A search bar says 'Filter route tables'. Below is a table with columns: Name, Route table ID, Explicit subnet associat..., Edge associations, Main, and VPC. Two rows are listed:

| Name | Route table ID        | Explicit subnet associat... | Edge associations | Main | VPC                           |
|------|-----------------------|-----------------------------|-------------------|------|-------------------------------|
| -    | rtb-9cf819f9          | -                           | -                 | Yes  | vpc-3843a35d                  |
| -    | rtb-0b82412de837e61ff | subnet-091275dcbf6af2...    | -                 | Yes  | vpc-084c62a7e74cc182e   my... |

4

# Step 8.2.2: Add to private subnet routing tables

- Configure Names
- Select VPC
- Create routing tables - 'my-private-route-table-01', 'my-private-route-table-02' and 'my-db-route-table-01'

Create route table [Info](#)

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.

**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 - <i>optional</i>                                | Remove                                |
|-----------------------------------|--|---------------------------------------|
| <input type="text" value="Name"/> | <input type="text" value="my-private-route-table-01"/> | <input type="button" value="Remove"/> |

You can add 49 more tags.

# Step 8.2.3: Add to private subnet routing tables

- For each newly created route table, configure Routes to NAT.
  - Select the routing table
  - Click on Edit Routes

| Route Table Name          | Target            |
|---------------------------|-------------------|
| my-private-route-table-01 | My-nat-gateway-01 |
| my-private-route-table-02 | My-nat-gateway-02 |
| my-db-route-table-01      | My-nat-gateway-03 |

The screenshot shows the AWS VPC Route Tables interface, specifically the 'Edit routes' page for a route table named 'rtb-03b7ad79847031e89'. The page displays a table of existing routes and allows for adding new routes.

| Destination | Target                                     | Status | Propagated |
|-------------|--|--------|------------|
| 10.0.0.0/16 | local                                      | Active | No         |
| 0.0.0.0/0   | nat-0e028d379737bd1d8 (my-nat-gateway-02)  | -      | No         |
|             | nat-07e4d1d99aaaf1ca7d (my-nat-gateway-03) |        |            |
|             | nat-03d21b0f55f900ef3 (my-nat-gateway-01)  |        |            |

At the bottom of the table, there is an 'Add route' button. Below the table, there are three buttons: 'Cancel', 'Preview', and 'Save changes'.

# Step 8.3: Add to subnet routing tables

For each newly created Route Table:

- Select Route Table
- Click on Edit subnet associations
- Configure as follows:

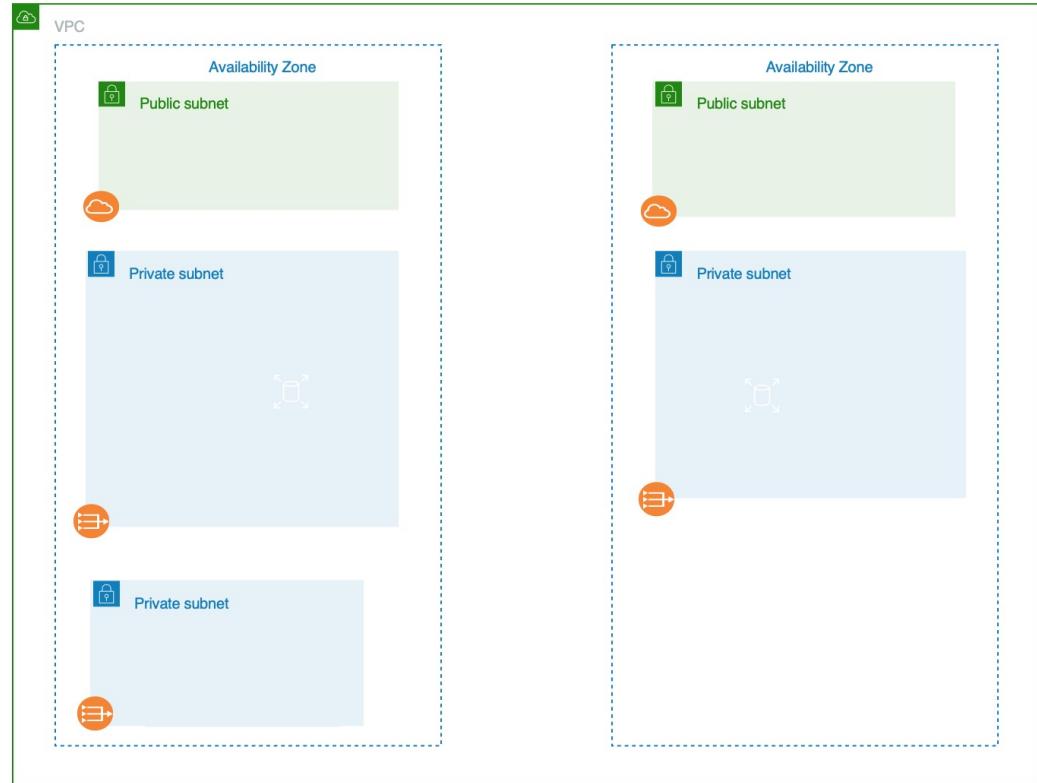
| Route Table Name          | Subnet               |
|---------------------------|----------------------|
| my-private-route-table-01 | My-private-subnet-01 |
| my-private-route-table-02 | My-private-subnet-02 |
| my-db-route-table-01      | My-db-subnet-01      |

The screenshot shows the AWS VPC Route Tables console. The top navigation bar includes 'VPC > Route tables > rtb-03b7ad79847031e89'. The main title is 'rtb-03b7ad79847031e89 / my-private-route-table-01'. On the right, there is an 'Actions ▾' button. Below the title, there are two tabs: 'Details' (selected) and 'Info'. The 'Details' tab displays route table ID (rtb-03b7ad79847031e89), VPC (vpc-084c62a7e74cc182e | my-vpc-1), Main status (No), and Owner ID (160864444630). Below the details, there are five tabs: 'Routes', 'Subnet associations' (highlighted in orange), 'Edge associations', 'Route propagation', and 'Tags'. Under the 'Subnet associations' tab, it says 'Explicit subnet associations (0)' and has a red box around the 'Edit subnet associations' button.

# Milestone 1

Verify the resources created:

- VPC
- Subnets
- Internet Gateway
- NAT Gateway
- Route Tables



# Main Steps - Computing

- Select AWS EC2 Service
- Select Region
- Launch instances
- View instance launched

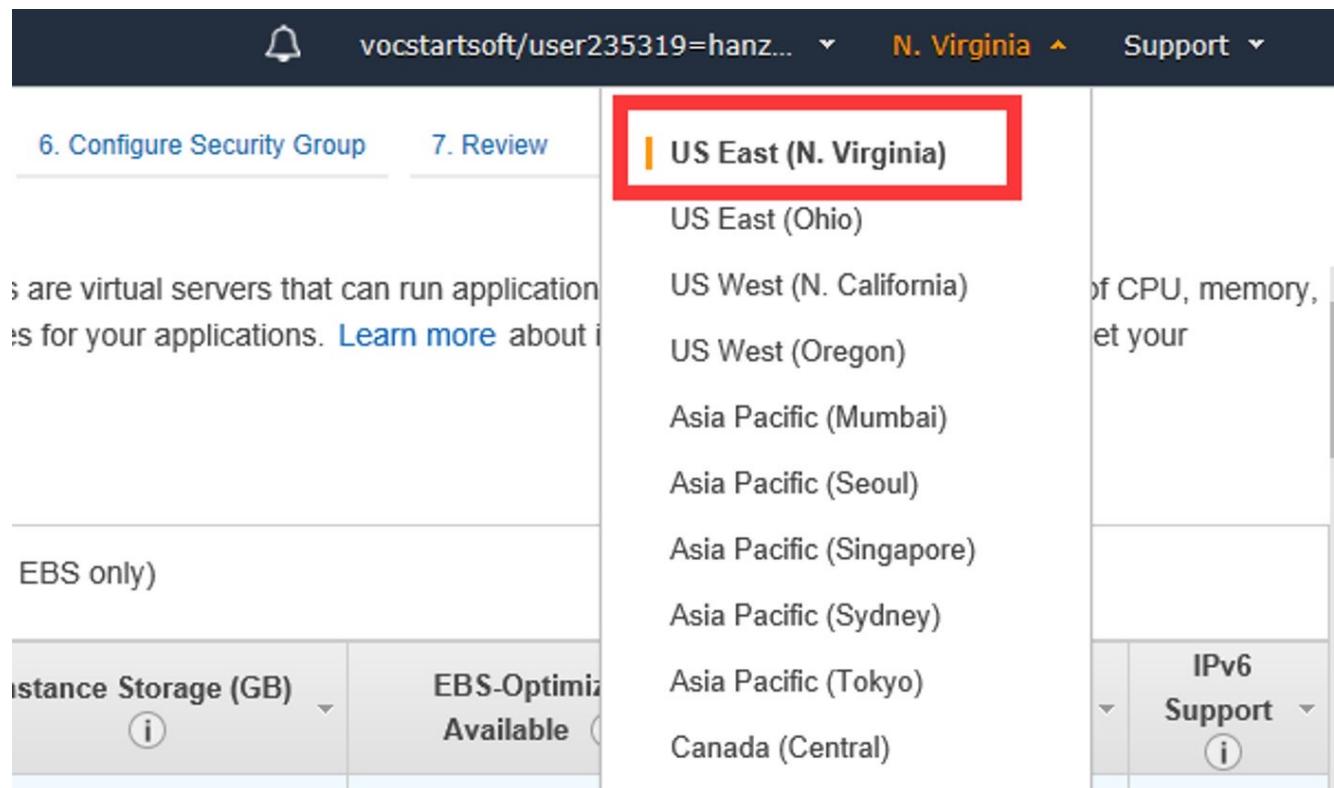
# Step 1: Select AWS EC2 Service

The screenshot shows the AWS Management Console homepage. At the top, there is a navigation bar with the AWS logo, 'Services' dropdown, 'Resource Groups' dropdown, a bell icon, the URL 'vocstartsoft/user235319=hanz...', 'N. Virginia' region dropdown, and 'Support' dropdown.

The main content area has a title 'AWS Management Console'. On the left, there is a sidebar titled 'AWS services' with a 'Find Services' search bar and a 'Recently visited services' section. The 'EC2' service is highlighted with a red box. Below this are sections for 'All services', 'Build a solution' (with options for launching a virtual machine or building a web app), and 'Explore AWS' (with links to AWS Marketplace, Amazon S3, AWS Fargate, and Amazon Redshift).

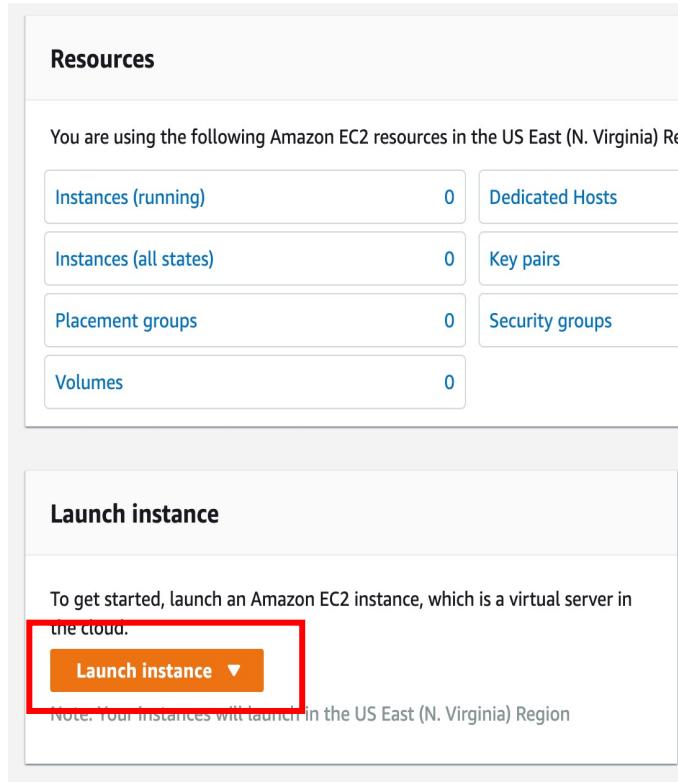
# Step 2: Select Region

Top right of the menu bar, between username and help: Choose N.Virginia



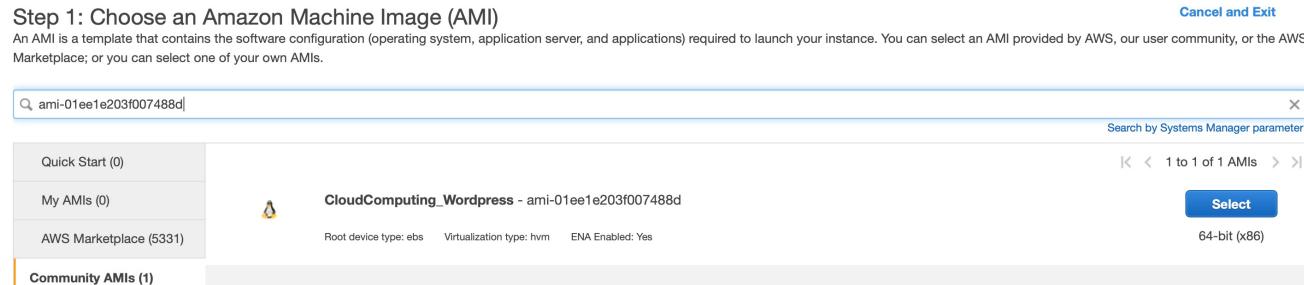
# Step 3.1: Launch Instances

- Launch 2 instances one by one
- Note: All parameters (except the subnet) are the same



# Step 3.2: Launch Instances

- Select AMI
- Search for 'ami-01ee1e203f007488d' under community ami's -> Select



- Select Instance Size (t2.micro)

Step 2: Choose an Instance Type

Amazon EC2 provides a wide selection of instance types optimized to fit different use cases. Instances are virtual servers that can run applications. They have varying combinations of CPU, memory, storage, and networking capacity, and give you the flexibility to choose the appropriate mix of resources for your applications. [Learn more](#) about instance types and how they can meet your computing needs.

The screenshot shows the AWS Lambda console with a search bar at the top containing 'ami-01ee1e203f007488d'. Below the search bar, there are tabs for 'Quick Start (0)', 'My AMIs (0)', 'AWS Marketplace (5331)', and 'Community AMIs (1)'. The 'Community AMIs' tab is selected. A search result for 'CloudComputing\_Wordpress - ami-01ee1e203f007488d' is displayed, showing details: Root device type: ebs, Virtualization type: hvm, ENA Enabled: Yes. To the right, there is a 'Select' button and a note '64-bit (x86)'. At the bottom right of the screenshot, there is a small red hand icon pointing towards the 'Select' button.

Filter by: All instance families ▾ Current generation ▾ Show/Hide Columns

Currently selected: t2.micro (- ECUs, 1 vCPUs, 2.5 GHz, -, 1 GiB memory, EBS only)

|                                     | Family | Type                           | vCPUs | Memory (GiB) | Instance Storage (GB) | EBS-Optimized Available | Network Performance | IPv6 Support |
|-------------------------------------|--------|--------------------------------|-------|--------------|-----------------------|-------------------------|---------------------|--------------|
| <input type="checkbox"/>            | t2     | t2.nano                        | 1     | 0.5          | EBS only              | -                       | Low to Moderate     | Yes          |
| <input checked="" type="checkbox"/> | t2     | t2.micro<br>Free tier eligible | 1     | 1            | EBS only              | -                       | Low to Moderate     | Yes          |
| <input type="checkbox"/>            | t2     | t2.small                       | 1     | 2            | EBS only              | -                       | Low to Moderate     | Yes          |
| <input type="checkbox"/>            | t2     | t2.medium                      | 2     | 4            | EBS only              | -                       | Low to Moderate     | Yes          |

# Step 3.3: Launch Instances

## Configure Instance

### Instance 1:

- Number of Instances: 1
- Network: my-vpc
- Subnet: my-private-subnet-01
- Auto-Assign Public IP: Disable
- Advanced Details – Userdata:  
#!/bin/bash  
sudo service apache2 restart

### Instance 2:

- Number of Instances: 1
- Network: my-vpc
- Subnet: my-private-subnet-02
- Auto-Assign Public IP: Disable
- Advanced Details – Userdata:  
#!/bin/bash  
sudo service apache2 restart

### Step 3: Configure Instance Details

Configure the instance to suit your requirements. You can launch multiple instances from the same AMI, request Spot instances to take advantage

|                               |   |   |
|-------------------------------|---|---|
| Number of instances           | <input type="text" value="1"/>  | Launch into Auto Scaling Group                                  |
| Purchasing option             | <input type="checkbox"/> Request Spot instances   |   |
| Network                       | vpc-09b23f87f1ca7c8d5   my-vpc  | <a href="#">Create new VPC</a>                                  |
| Subnet                        | subnet-0e9d6a2747c15fa97   my-private-subnet-01   | <a href="#">Create new subnet</a><br>250 IP Addresses available |
| Auto-assign Public IP         | <input type="checkbox"/> Enable   |   |
| Placement group               | <input type="checkbox"/> Add instance to placement group  |   |
| Capacity Reservation          | <input type="checkbox"/> Open   |   |
| Domain join directory         | No directory  | <a href="#">Create new directory</a>                            |
| IAM role                      | <input type="checkbox"/> None   | <a href="#">Create new IAM role</a>                             |
| CPU options                   | <input type="checkbox"/> Specify CPU options  |   |
| Shutdown behavior             | Stop  |   |
| Stop - Hibernate behavior     | <input type="checkbox"/> Enable hibernation as an additional stop behavior                                      |   |
| Enable termination protection | <input type="checkbox"/> Protect against accidental termination   |   |
| Monitoring                    | <input type="checkbox"/> Enable CloudWatch detailed monitoring<br><a href="#">Additional charges apply.</a>     |   |
| Tenancy                       | Shared - Run a shared hardware instance<br><a href="#">Additional charges will apply for dedicated tenancy.</a> |   |
| Elastic Inference             | <input type="checkbox"/> Add an Elastic Inference accelerator<br><a href="#">Additional charges apply.</a>      |   |

# Step 3.4: Launch Instances

Add a root EBS volume of size 8GB to the instance

## Step 4: Add Storage

Your instance will be launched with the following storage device settings. You can attach additional EBS volumes and instance store volumes to your instance, or edit the settings of the root volume. You can also attach additional EBS volumes after launching an instance, but not instance store volumes. [Learn more](#) about storage options in Amazon EC2.

| Volume Type | Device    | Snapshot               | Size (GiB) | Volume Type               | IOPS       | Throughput (MB/s) | Delete on Termination               | Encryption    |
|-------------|-----------|------------------------|------------|---------------------------|------------|-------------------|-------------------------------------|---------------|
| Root        | /dev/sda1 | snap-0846ce4394d115972 | 8          | General Purpose SSD (gp2) | 100 / 3000 | N/A               | <input checked="" type="checkbox"/> | Not Encrypted |

[Add New Volume](#)

# Step 3.5: Launch Instances

- Add Name Tags as:
  - Instance 1: my-ec2-01
  - Instance 2: my-ec2-02
- Optional: You can any other key-value tags

## Step 5: Add Tags

A tag consists of a case-sensitive key-value pair. For example, you could define a tag with key = Name and value = Webserver.

A copy of a tag can be applied to volumes, instances or both.

Tags will be applied to all instances and volumes. [Learn more](#) about tagging your Amazon EC2 resources.

| Key  | (128 characters maximum) | Value     | (256 characters maximum) | Instances                           | Volumes                             |                                  |
|------|--------------------------|-----------|--------------------------|-------------------------------------|-------------------------------------|----------------------------------|
| Name |                          | my-ec2-01 |                          | <input checked="" type="checkbox"/> | <input checked="" type="checkbox"/> | <input type="button" value="X"/> |

# Step 3.6: Launch Instances

- Create a single Security Group for both instances
  - Allow HTTP access from all sources
- Note: This will be changed after creating the ELB

## Instance 1:

Step 6: Configure Security Group

A security group is a set of firewall rules that control the traffic for your instance. On this page, you can add rules to allow specific traffic to reach your instance. For example, if you want to set up a web server and allow Internet traffic to reach your instance, add rules that allow unrestricted access to the HTTP and HTTPS ports. You can create a new security group or select from an existing one below. [Learn more](#) about Amazon EC2 security groups.

Assign a security group:  Create a new security group  Select an existing security group

Security group name: my-ec2-security-group-01  
Description: my-ec2-security-group-01

| Type | Protocol | Port Range | Source           | Description                |
|------|----------|------------|------------------|----------------------------|
| HTTP | TCP      | 80         | Custom 0.0.0.0/0 | e.g. SSH for Admin Desktop |

[Add Rule](#)

## Instance 2:

Step 6: Configure Security Group

A security group is a set of firewall rules that control the traffic for your instance. On this page, you can add rules to allow specific traffic to reach your instance. For example, if you want to set up a web server and allow Internet traffic to reach your instance, add rules that allow unrestricted access to the HTTP and HTTPS ports. You can create a new security group or select from an existing one below. [Learn more](#) about Amazon EC2 security groups.

Assign a security group:  Create a new security group  Select an existing security group

| Security Group ID  | Name                     | Description                | Actions                     |
|--|--------------------------|----------------------------|-----------------------------|
| <input type="checkbox"/> sg-06aea793e09a3aa53            | default                  | default VPC security group | <a href="#">Copy to new</a> |
| <input checked="" type="checkbox"/> sg-014d2b7d7d2cb9039 | my-ec2-security-group-01 | my-ec2-security-group-01   | <a href="#">Copy to new</a> |

# Step 3.7: Launch Instances

- Click on “Review and Launch”
- Click on “Launch”
- Configure a single Key Pair for both instances

## Instance 1:

Select an existing key pair or create a new key pair X

A key pair consists of a **public key** that AWS stores, and a **private key file** that you store. Together, they allow you to connect to your instance securely. For Windows AMIs, the private key file is required to obtain the password used to log into your instance. For Linux AMIs, the private key file allows you to securely SSH into your instance.

Note: The selected key pair will be added to the set of keys authorized for this instance. Learn more about [removing existing key pairs from a public AMI](#).

**Info** You have to download the **private key file (\*.pem file)** before you can continue. [Store it in a secure and accessible location](#). You will not be able to download the file again after it's created.

## Instance 2:

Select an existing key pair or create a new key pair X

A key pair consists of a **public key** that AWS stores, and a **private key file** that you store. Together, they allow you to connect to your instance securely. For Windows AMIs, the private key file is required to obtain the password used to log into your instance. For Linux AMIs, the private key file allows you to securely SSH into your instance.

Note: The selected key pair will be added to the set of keys authorized for this instance. Learn more about [removing existing key pairs from a public AMI](#).

I acknowledge that I have access to the selected private key file (my-ec2-key-pair.pem), and that without this file, I won't be able to log into my instance.

- Click on “Download Key Pair”
- Then launch instance.

# Step 4: View Instances

- Verify that instances have been successfully launched:
  - Check Instance State is Running
  - Status Checks should be passed (may take some time)
  - Select the instance to view more details and monitoring

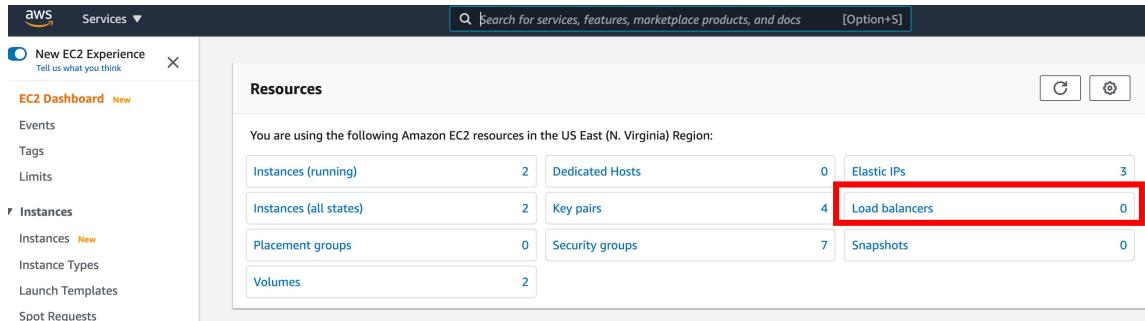
| Instances (2) <a href="#">Info</a>    |           | <a href="#">C</a> | Connect              | Instance state ▾ | Actions ▾                   |              |                     |                   |
|---------------------------------------|-----------|-------------------|----------------------|------------------|-----------------------------|--------------|---------------------|-------------------|
| <input type="text"/> Filter instances |           |                   |                      |                  |                             |              |                     |                   |
|                                       | Name ▾    | Instance ID       | Instance state ▾     | Instance type ▾  | Status check                | Alarm status | Availability Zone ▾ | Public IPv4 DNS ▾ |
| <input type="checkbox"/>              | my-ec2-01 |                   | <span>Running</span> | t2.micro         | <span>2/2 checks ...</span> | No alarms    | us-east-1a          | -                 |
| <input type="checkbox"/>              | my-ec2-02 |                   | <span>Running</span> | t2.micro         | <span>2/2 checks ...</span> | No alarms    | us-east-1b          | -                 |

# Main Steps – Elastic Load Balancer

- Select AWS Service
- Select Load Balancer Type
- Basic Configuration
- Security Group
- Configure Health Check
- Add EC2 instances
- Verify successful creation
- Add access to EC2 security group

# Step 1: Select AWS Service

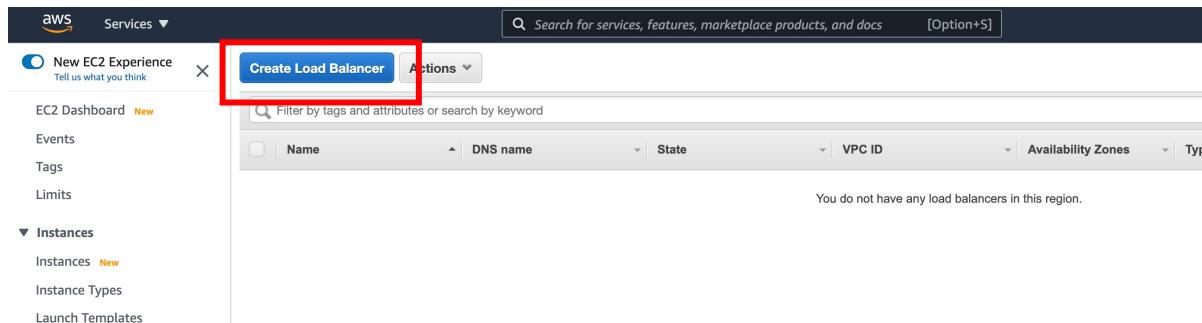
On the EC2 console:



The screenshot shows the AWS EC2 Dashboard. On the left, there's a sidebar with options like New EC2 Experience, EC2 Dashboard, Instances, and others. The main area is titled 'Resources' and displays various EC2 metrics. A red box highlights the 'Load balancers' section, which shows 0 items.

| Category               | Count |
|------------------------|-------|
| Instances (running)    | 2     |
| Dedicated Hosts        | 0     |
| Elastic IPs            | 3     |
| Instances (all states) | 2     |
| Key pairs              | 4     |
| Load balancers         | 0     |
| Placement groups       | 0     |
| Security groups        | 7     |
| Volumes                | 2     |
| Snapshots              | 0     |

## Create Load Balancer:



The screenshot shows the AWS EC2 Dashboard. The 'Create Load Balancer' button in the top navigation bar is highlighted with a red box. Below it, the main content area shows a table for managing load balancers, with a message indicating 'You do not have any load balancers in this region.'

# Step 2: Select Load Balancer Type

## Select Classic Load Balancer:

### Select load balancer type

Elastic Load Balancing supports four types of load balancers: Application Load Balancers, Network Load Balancers, Gateway Load Balancers, and Classic Load Balancers. Choose the load balancer type that meets your needs.  
[Learn more about which load balancer is right for you](#)

| Application Load Balancer   | Network Load Balancer   | Gateway Load Balancer   | Classic Load Balancer  |
|---|---|---|--|
| <br><a href="#">Create</a>   | <br><a href="#">Create</a>   | <br><a href="#">Create</a>   | <br><a href="#">Create</a>  |
| Choose an Application Load Balancer when you need a flexible feature set for your web applications with HTTP and HTTPS traffic. Operating at the request level, Application Load Balancers provide advanced routing and visibility features targeted at application architectures, including microservices and containers.<br><a href="#">Learn more &gt;</a> | Choose a Network Load Balancer when you need ultra-high performance, TLS offloading at scale, centralized certificate deployment, support for UDP, and static IP addresses for your application. Operating at the connection level, Network Load Balancers are capable of handling millions of requests per second securely while maintaining ultra-low latencies.<br><a href="#">Learn more &gt;</a> | Choose a Gateway Load Balancer when you need to deploy and manage a fleet of third-party virtual appliances that support GENEVE. These appliances enable you to improve security, compliance, and policy controls.<br><a href="#">Learn more &gt;</a> | PREVIOUS GENERATION<br>for HTTP, HTTPS, and TCP<br><br>Choose a Classic Load Balancer when you have an existing application running in the EC2-Classic network.<br><a href="#">Learn more &gt;</a> |

# Step 3: Basic configuration

- Name: my-load-balancer-01
- Create LB inside: my-vpc
- Select subnets: my-public-subnet-01, my-public-subnet-02

Note: Always assign public subnets to ELB in the same AZs as your EC2 instances.

Step 1: Define Load Balancer

Basic Configuration

This wizard will walk you through setting up a new load balancer. Begin by giving your new load balancer a unique name so that you can identify it from other load balancers you might create. You will also need to configure ports and protocols for your load balancer. Traffic from your clients can be routed from any load balancer port to any port on your EC2 instances. By default, we've configured your load balancer with a standard web server on port 80.

Load Balancer name:  Create LB Inside:  Create an internal load balancer:

Enable advanced VPC configuration:

Listener Configuration:

| Load Balancer Protocol | Load Balancer Port | Instance Protocol | Instance Port |
|------------------------|--------------------|-------------------|---------------|
| HTTP                   | 80                 | HTTP              | 80            |

Add

Select Subnets

You will need to select a Subnet for each Availability Zone where you wish traffic to be routed by your load balancer. If you have instances in only one Availability Zone, please select at least two Subnets in different Availability Zones to provide higher availability for your load balancer.

VPC vpc-09b23f87f1ca7c8d5 (10.0.0.0/16) | my-vpc

Available subnets

| Actions | Availability Zone | Subnet ID                | Subnet CIDR | Name                 |
|---------|-------------------|--------------------------|-------------|----------------------|
|         | us-east-1a        | subnet-06f7d489cef536ca  | 10.0.4.0/24 | my-db-subnet-01      |
|         | us-east-1a        | subnet-0e96d8a2747c15f97 | 10.0.2.0/24 | my-private-subnet-01 |
|         | us-east-1a        | subnet-04773e298ff9571f1 | 10.0.3.0/24 | my-private-subnet-02 |

Selected subnets

| Actions | Availability Zone | Subnet ID                | Subnet CIDR | Name                |
|---------|-------------------|--------------------------|-------------|---------------------|
|         | us-east-1a        | subnet-01f24ec1bb75ae6cf | 10.0.0.0/24 | my-public-subnet-01 |
|         | us-east-1b        | subnet-0e72b6d01522b4c63 | 10.0.1.0/24 | my-public-subnet-02 |

# Step 4: Security Group

- Create a new Security Group for the Load Balancer:
- Allow HTTP (port 80) access from your IP

## Step 2: Assign Security Groups

You have selected the option of having your Elastic Load Balancer inside of a VPC, which allows you to assign security groups to your load balancer. Please select the security groups to assign to this load balancer. This can be changed at any time.

Assign a security group:

Create a new security group  
 Select an existing security group

Security group name: my-lb-security-group-01

Description: Security Group for Load Balancer

| Type          | Protocol | Port Range | Source   |
|---------------|----------|------------|--|
| Custom TCP F▼ | TCP      | 80         | <input checked="" type="checkbox"/> Custom Anywhere<br>My IP |

CIDR, IP or Security Group

Add Rule

# Step 5: Configure Health Check

- Used by the load balancer to determine if instance is healthy

## Step 4: Configure Health Check

Your load balancer will automatically perform health checks on your EC2 instances and only route traffic to instances that pass the health check needs.

|               |  |
|---------------|--|
| Ping Protocol | <input type="text" value="HTTP"/>        |
| Ping Port     | <input type="text" value="80"/>          |
| Ping Path     | <input type="text" value="/index.html"/> |

### Advanced Details

---

|                     |   |
|---------------------|---|
| Response Timeout    | <input type="text" value="5"/> seconds  |
| Interval            | <input type="text" value="30"/> seconds |
| Unhealthy threshold | <input type="text" value="2"/>          |
| Healthy threshold   | <input type="text" value="10"/>         |

# Step 6: Add EC2 instances

- Add EC2 instances as targets for the ELB

Note: If you don't see the correct instances here, check the subnet configurations of EC2 and ELB

## Step 5: Add EC2 Instances

The table below lists all your running EC2 Instances. Check the boxes in the Select column to add those instances to this load balancer.

VPC vpc-09b23f87f1ca7c8d5 (10.0.0.0/16) | my-vpc

| Select                   | Instance            | Name      | State   | Security groups          | Zone       | Subnet ID         |
|--------------------------|---------------------|-----------|---------|--------------------------|------------|-------------------|
| <input type="checkbox"/> | i-0f958f44ff98cde18 | my-ec2-01 | running | my-ec2-security-group-01 | us-east-1a | subnet-0e9d6a2... |
| <input type="checkbox"/> | i-0a9bf59baea804c7c | my-ec2-02 | running | my-ec2-security-group-01 | us-east-1b | subnet-04773e2... |

## Availability Zone Distribution

2 instances in us-east-1a

- Enable Cross-Zone Load Balancing (i)  
 Enable Connection Draining (i)  seconds

# Step 7: Verify successful creation

- Verify creation
- Verify status of instances is “InService”
- Note down the DNS name of the ELB

# Step 8.1: Add access to EC2 security group

- On EC2 console, open security groups

The screenshot shows the AWS EC2 Resources page. At the top, it says "Resources" and "You are using the following Amazon EC2 resources in the US East (N. Virginia) Region:". Below this are several metrics with counts: Instances (running) 2, Dedicated Hosts 0, Elastic IPs 3, Instances (all states) 2, Key pairs 4, Load balancers 1, Placement groups 0, Security groups 8, Snapshots 0, and Volumes 2. The "Security groups" row is highlighted with a red box.

- Select security group and Edit Inbound Rules

The screenshot shows the AWS Security Groups page. It lists three security groups: "sg-014d2b7d7d2cb9039 - my-ec2-security-group-01" (selected with a checked checkbox and highlighted with a red box), "sg-0745e26cbbef4669 - default", and "sg-05a4e707-00-7aa5-tz - my-lb-security-group-01". Each row includes columns for Name, Security group ID, Description, Owner, and Inbound rules count. Below the table, there's a navigation bar with tabs: Details, Inbound rules (which is selected and highlighted with a red box), Outbound rules, and Tags. At the bottom right, there's a button labeled "Edit inbound rules" (also highlighted with a red box).

# Step 8.2: Add access to EC2 security group

- Edit Inbound rule
  - Protocol: HTTP
  - Port: 80
  - Source: Security group of ELB

Edit inbound rules [Info](#)

Inbound rules control the incoming traffic that's allowed to reach the instance.

| Inbound rules <a href="#">Info</a>   |                               |                                 |                             |  |                        |
|--|-------------------------------|---------------------------------|-----------------------------|--|------------------------|
| Type <a href="#">Info</a>  | Protocol <a href="#">Info</a> | Port range <a href="#">Info</a> | Source <a href="#">Info</a> | Description - optional <a href="#">Info</a>        | Delete                 |
| HTTP   | TCP                           | 80                              | Custom                      | <input type="text" value="sg-"/> <a href="#">X</a> | <a href="#">Delete</a> |
| <a href="#">Add rule</a>   |                               |                                 |                             |  |                        |
| <p><b>⚠</b> NOTE: Any edits made on existing rules will result in the edited rule being deleted and a new rule created with the new details. This means that rule to be dropped for a very brief period of time until the new rule can be created.</p> |                               |                                 |                             |  |                        |

**Type:** HTTP    **Protocol:** TCP    **Port range:** 80    **Source:** Custom    **Description - optional:**

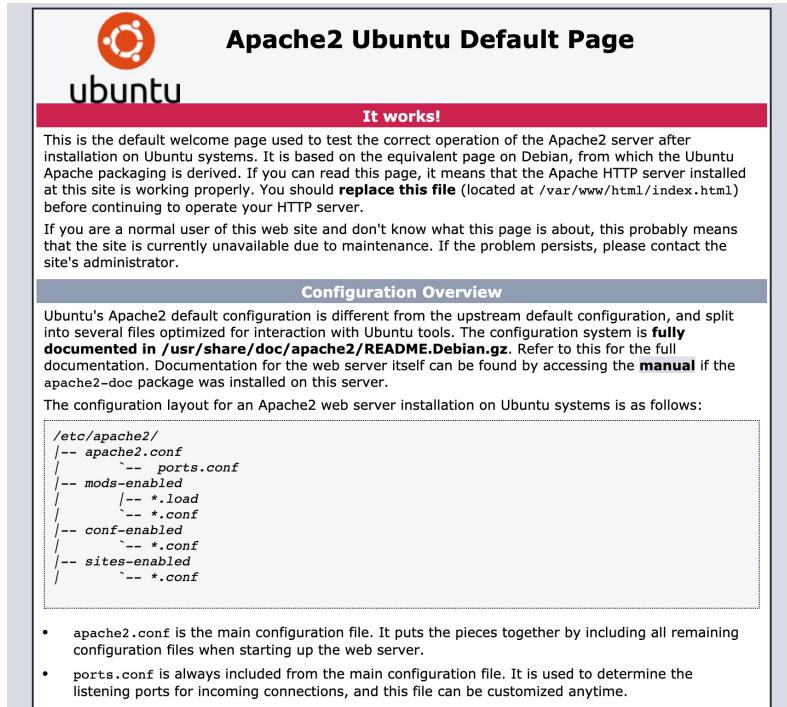
**Source:**  [X](#)

**Security Groups:**

- my-ec2-security-group | sg-014d2b7d7d2c9b9039
- default | sg-06aea793e09a3aa53
- my-lb-security-group | sg-0745e26cbef4669

# Milestone 2

You should be able to access application on EC2 using DNS name of ELB



The screenshot shows the Apache2 Ubuntu Default Page. At the top, there's a red bar with the text "It works!". Below it, the page title is "Apache2 Ubuntu Default Page". It features the Ubuntu logo. The main content area contains text about the default page, instructions to replace the file if it's not working, and a "Configuration Overview" section. The configuration overview details the layout of Apache2 files in /etc/apache2/.

This is the default welcome page used to test the correct operation of the Apache2 server after installation on Ubuntu systems. It is based on the equivalent page on Debian, from which the Ubuntu Apache packaging is derived. If you can read this page, it means that the Apache HTTP server installed at this site is working properly. You should [replace this file](#) (located at `/var/www/html/index.html`) before continuing to operate your HTTP server.

If you are a normal user of this web site and don't know what this page is about, this probably means that the site is currently unavailable due to maintenance. If the problem persists, please contact the site's administrator.

**Configuration Overview**

Ubuntu's Apache2 default configuration is different from the upstream default configuration, and split into several files optimized for interaction with Ubuntu tools. The configuration system is [fully documented in `/usr/share/doc/apache2/README.Debian.gz`](#). Refer to this for the full documentation. Documentation for the web server itself can be found by accessing the [manual](#) if the `apache2-doc` package was installed on this server.

The configuration layout for an Apache2 web server installation on Ubuntu systems is as follows:

```
/etc/apache2/
|-- apache2.conf
|   '-- ports.conf
|-- mods-enabled
|   '-- *.load
|   '-- *.conf
|-- conf-enabled
|   '-- *.conf
|-- sites-enabled
|   '-- *.conf
```

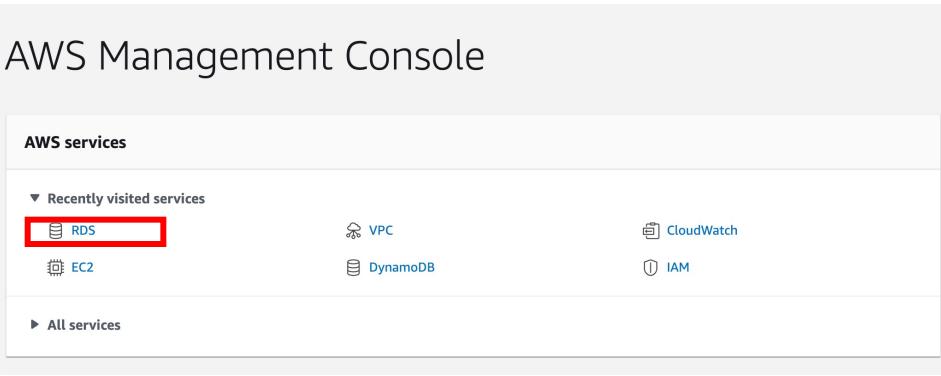
- `apache2.conf` is the main configuration file. It puts the pieces together by including all remaining configuration files when starting up the web server.
- `ports.conf` is always included from the main configuration file. It is used to determine the listening ports for incoming connections, and this file can be customized anytime.

# Main Steps – RDS DB Instance

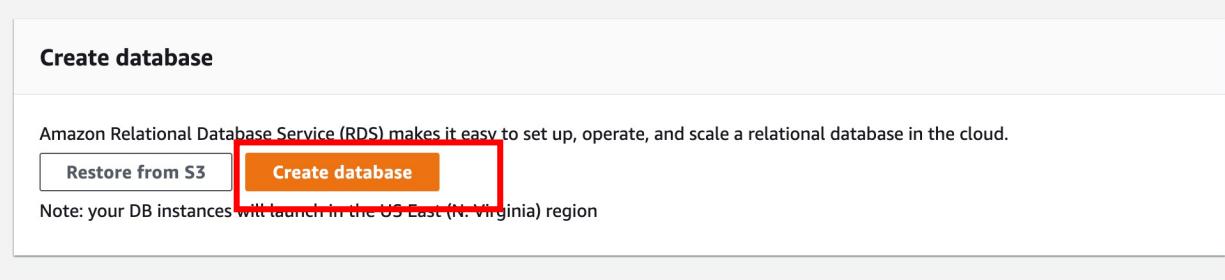
- Select AWS Service
- Create DB Instance
- Add inbound rule to DB Security Group
- Verify Creation

# Step 1: Select AWS Service

On AWS Console:



The screenshot shows the AWS Management Console homepage. In the top navigation bar, the text "AWS Management Console" is visible. Below it, the "AWS services" section is displayed. Under "Recently visited services", the "RDS" icon is highlighted with a red box. Other services listed include VPC, CloudWatch, EC2, DynamoDB, and IAM. A "All services" link is also present.

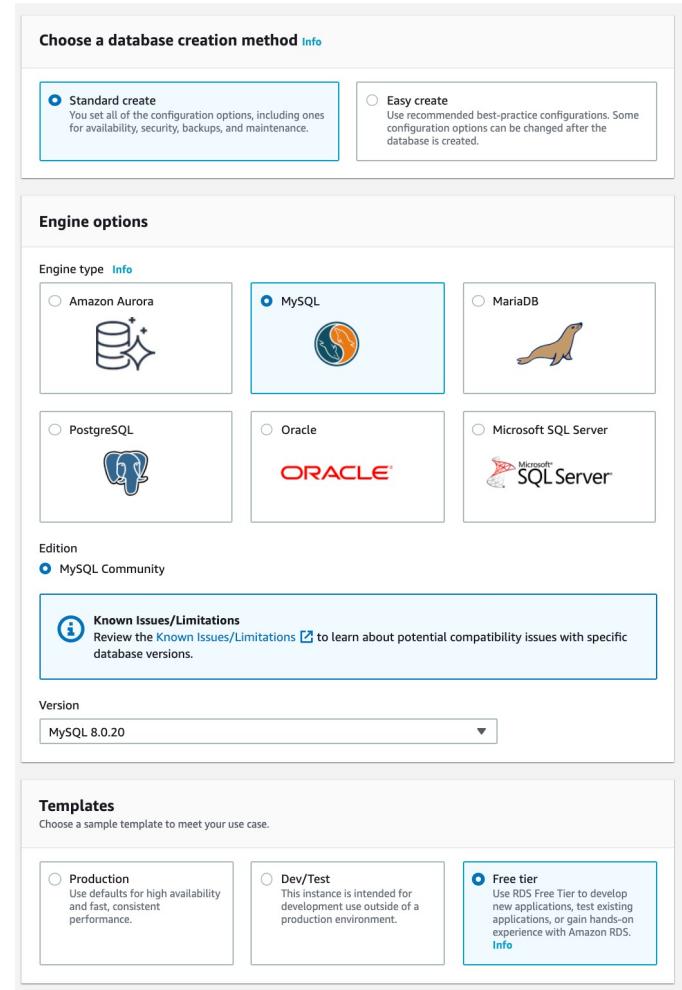
  


The screenshot shows the "Create database" page for Amazon RDS. At the top, the title "Create database" is shown. Below it, a descriptive text states: "Amazon Relational Database Service (RDS) makes it easy to set up, operate, and scale a relational database in the cloud." Two buttons are present: "Restore from S3" and a large orange "Create database" button, which is also highlighted with a red box. A note at the bottom indicates: "Note: your DB instances will launch in the US East (N. Virginia) region".

# Step 2.1: Create DB Instance

## DB Instance Configuration:

- Creation Method: Standard Create
- Engine Type: MySQL
- Template: Free Tier



# Step 2.2: Create DB Instance

## DB Instance Configuration:

- DB Instance Identifier: my-db-instance-01
- Master username: admin
- Master password: adminpassword

**Settings**

**DB instance identifier** [Info](#)  
Type a name for your DB instance. The name must be unique across all DB instances owned by your AWS account in the current AWS Region.

The DB instance identifier is case-insensitive, but is stored as all lowercase (as in "mydbinstance"). Constraints: 1 to 60 alphanumeric characters or hyphens (1 to 15 for SQL Server). First character must be a letter. Can't contain two consecutive hyphens. Can't end with a hyphen.

**Credentials Settings**

**Master username** [Info](#)  
Type a login ID for the master user of your DB instance.

1 to 16 alphanumeric characters. First character must be a letter

**Auto generate a password**  
Amazon RDS can generate a password for you, or you can specify your own password

**Master password** [Info](#)  
\*\*\*\*\*  
Constraints: At least 8 printable ASCII characters. Can't contain any of the following: / (slash), ' (single quote), " (double quote) and @ (at sign).

**Confirm password** [Info](#)  
\*\*\*\*\*

# Step 2.3: Create DB Instance

## DB Instance Configuration:

- VPC: my-vpc
- Public Access: No
- VPC Security Group: Create new
- New VPC security group name: my-db-security-group
- Availability Zone: us-east-1a
- Authentication: Password authentication
- Additional Configuration:
  - Initial database name: wordpress

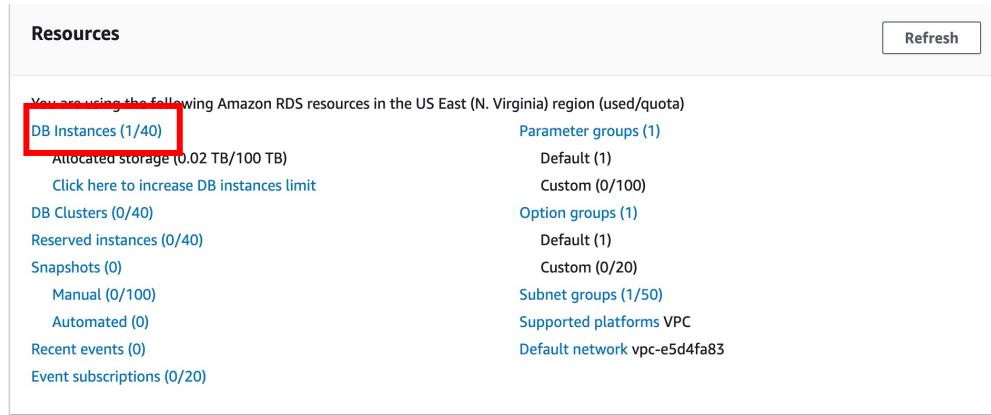
## Create Database

Note: It will take some time for the instance to be available. Move on to the next step.

The screenshot shows the 'Create DB Instance' configuration interface. The 'Connectivity' section is set to 'Virtual private cloud (VPC)' with 'my-vpc' selected. A note states 'After a database is created, you can't change the VPC selection.' The 'Subnet group' is set to 'Create new DB Subnet Group'. Under 'Public access', 'No' is selected. In the 'VPC security group' section, 'Create new' is selected and 'my-db-security-group' is entered. The 'New VPC security group name' field also contains 'my-db-security-group'. The 'Availability Zone' is set to 'us-east-1a'. The 'Additional configuration' section includes 'Database authentication' options: 'Password authentication' (selected), 'Password and IAM database authentication', and 'Password and Kerberos authentication' (disabled). A note at the bottom of this section states 'Choose a directory in which you want to allow authorized users to authenticate with this DB instance using Kerberos Authentication.'

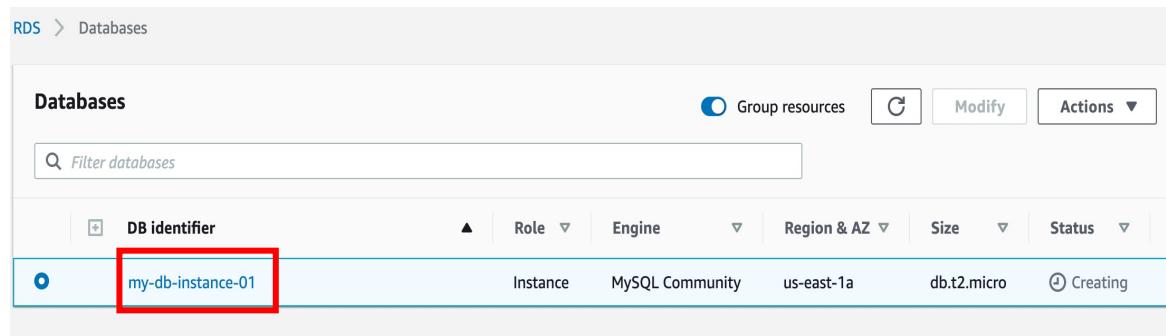
# Step 3.1: Add inbound rule to DB Security Group

- On RDS Console:



The screenshot shows the 'Resources' page in the AWS RDS console. At the top, it displays the number of resources used: 1/40 DB Instances, 1 Parameter group, 1 Option group, 1 Subnet group, and 1 Default network. Below this, there are links for Allocating storage, Click here to increase DB instances limit, DB Clusters, Reserved instances, Snapshots, Recent events, and Event subscriptions.

- Select DB instance:



The screenshot shows the 'Databases' page in the AWS RDS console. It lists a single database named 'my-db-instance-01'. The table columns include DB identifier, Instance, Engine, Region & AZ, Size, Status, and Actions. The 'my-db-instance-01' row is highlighted with a red box around its 'DB identifier' cell.

| DB identifier     | Instance        | Engine     | Region & AZ | Size     | Status | Actions   |
|-------------------|-----------------|------------|-------------|----------|--------|-----------|
| my-db-instance-01 | MySQL Community | us-east-1a | db.t2.micro | Creating | -      | [Actions] |

# Step 3.2: Add inbound rule to DB Security Group

- Select Security Group

RDS > Databases > my-db-instance-01

my-db-instance-01

**Summary**

|                                    |                  |                           |                           |
|------------------------------------|------------------|---------------------------|---------------------------|
| DB Identifier<br>my-db-instance-01 | CPU<br>-         | Status<br>Creating        | Class<br>db.t2.micro      |
| Role<br>Instance                   | Current activity | Engine<br>MySQL Community | Region & AZ<br>us-east-1a |

**Connectivity & security** | Monitoring | Logs & events | Configuration | Maintenance & backups | Tags

**Connectivity & security**

|                 |                                 |  |
|-----------------|---------------------------------|--|
| Endpoint & port | Networking                      | Security   |
| Endpoint<br>-   | Availability zone<br>us-east-1a | VPC security groups<br>my-db-security-group (sg-0c2315ae6f7b6c1cb)<br>[active] |

- Add Inbound Rule:
  - Port – 3306, Source – EC2 Security Group

Edit inbound rules [Info](#)

Inbound rules control the incoming traffic that's allowed to reach the instance.

**Inbound rules** [Info](#)

| Type         | Protocol | Port range | Source | Description - optional                                 |
|--------------|----------|------------|--------|--|
| MySQL/Aurora | TCP      | 3306       | Custom | <input type="text" value="sg"/> <a href="#">Delete</a> |

[Add rule](#)

⚠ NOTE: Any edits made on existing rules will result in the edited rule being deleted and a new rule created with the new details.

Security Groups

- my-ec2-security-group... | sg-014d2b7d7d2cb9039
- default | sg-06aea795e09a3aa53
- my-lb-security-group... | sg-0745e26ccbefd46669
- my-db-security-group | sg-0c2315ae6f7b6c1cb

on that rule to be dropped for a very brief period of time until the new rule can be

# Step 4: Verify Creation

Wait for “Available” status

Databases

Group resources  Modify Actions  Create database

Filter databases

| DB identifier     | Role     | Engine          | Region & AZ | Size        | Status                 | CPU   | Current activity |    |
|-------------------|----------|-----------------|-------------|-------------|------------------------|-------|------------------|----|
| my-db-instance-01 | Instance | MySQL Community | us-east-1a  | db.t2.micro | <span>Available</span> | 2.67% | 0 Connections    | no |

Click on the “DB identifier”  
Note down the endpoint of the  
DB instance

my-db-instance-01

Summary

DB identifier my-db-instance-01 CPU 2.37%

Role Instance Current activity 0 Connecti

Connectivity & security Monitoring Logs & events Con

Connectivity & security

Endpoint & port

Endpoint my-db-instance-01.us-east-1.rds.amazonaws.com

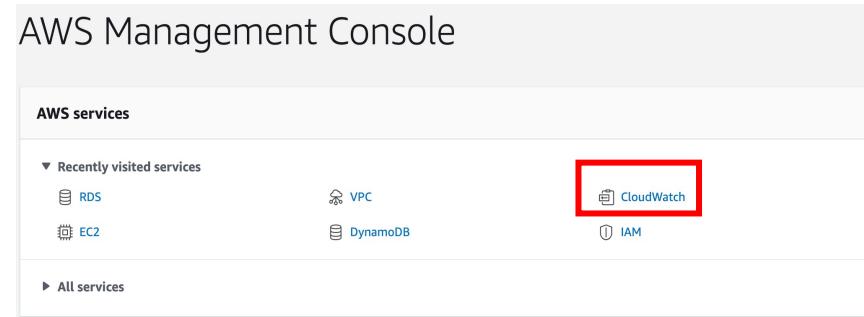
Port 3306

# Main Steps - Monitoring

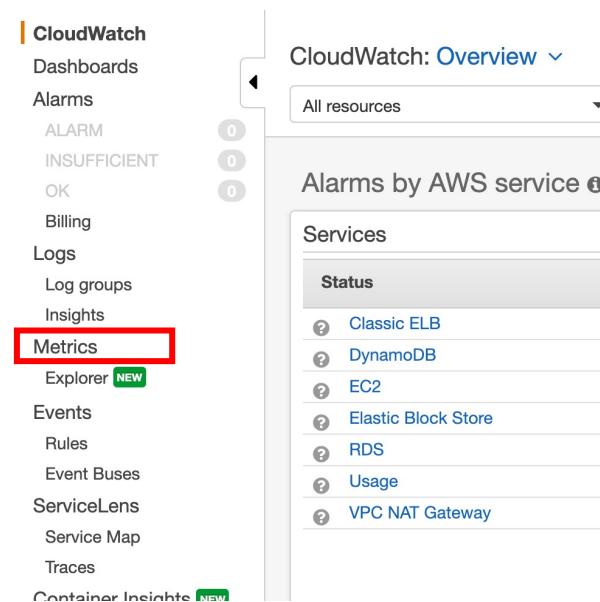
- Select AWS Service
- Observe EC2 CPU utilization

# Step 1: Select AWS Service

- On AWS Console:

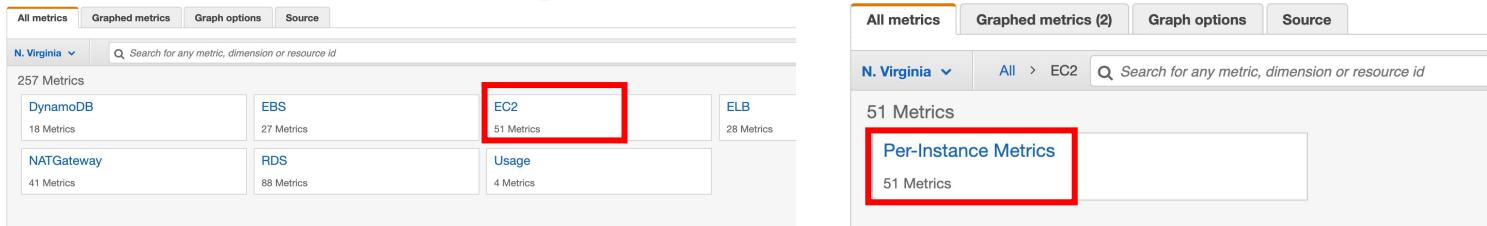


- Open Metrics:

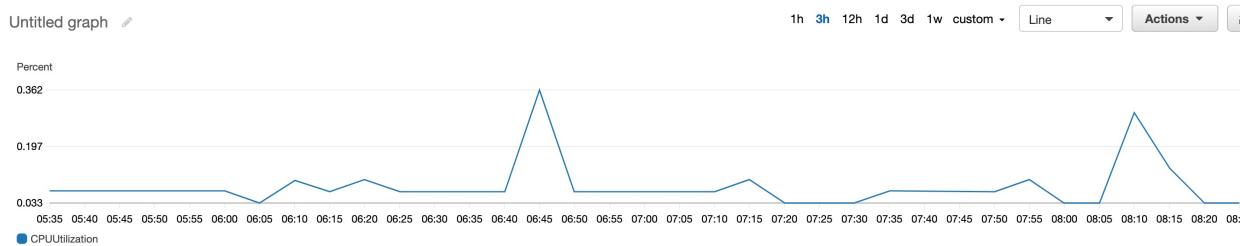


# Step 2: Observe EC2 CPU utilization

- Select EC2 -> Per-Instance Metrics



- Select CPU Utilization Metric for my-ec2-01



The figure is a screenshot of the AWS CloudWatch Metrics Insights interface. It shows a table titled "All metrics" for the N. Virginia region, filtered to show metrics for the instance "my-ec2-01". The table has four columns: "Instance Name (51)", "InstanceId", "Metric Name", and "Unit". The rows show three metrics for the instance: "my-ec2-01" with instance ID "i-0f958f44ff98cde18" and metric names "NetworkPacketsIn", "NetworkPacketsOut", and "CPUUtilization". The "CPUUtilization" row is highlighted by a red box.

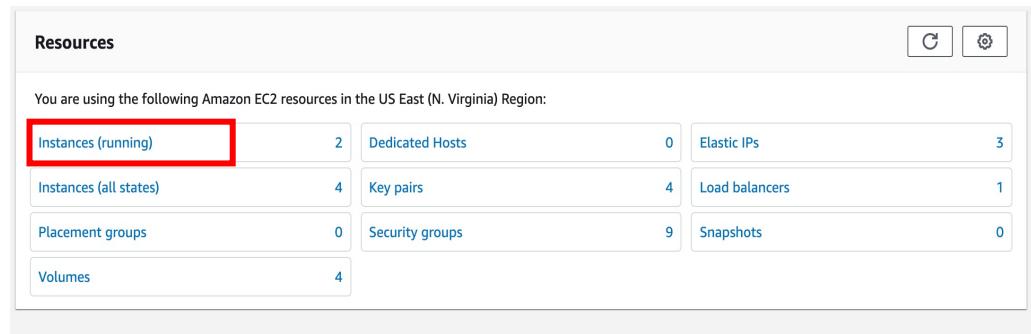
| All metrics                                   | Graphed metrics (1)                 | Graph options                                   | Source |
|---|-------------------------------------|---|--------|
| N. Virginia                                   | All > EC2 > Per-Instance Metrics    | Search for any metric, dimension or resource id | Graph  |
| <input type="checkbox"/> Instance Name (51)   | <input type="checkbox"/> InstanceId | <input type="checkbox"/> Metric Name            |        |
| <input type="checkbox"/> my-ec2-01            | i-0f958f44ff98cde18                 | NetworkPacketsIn                                |        |
| <input type="checkbox"/> my-ec2-01            | i-0f958f44ff98cde18                 | NetworkPacketsOut                               |        |
| <input checked="" type="checkbox"/> my-ec2-01 | i-0f958f44ff98cde18                 | CPUUtilization                                  |        |

# Main Steps – Add Auto Scaling

- Delete existing EC2 instances
- Create Launch Configuration
- Create Auto Scaling Group

# Step 1: Delete existing instances

- On EC2 Dashboard



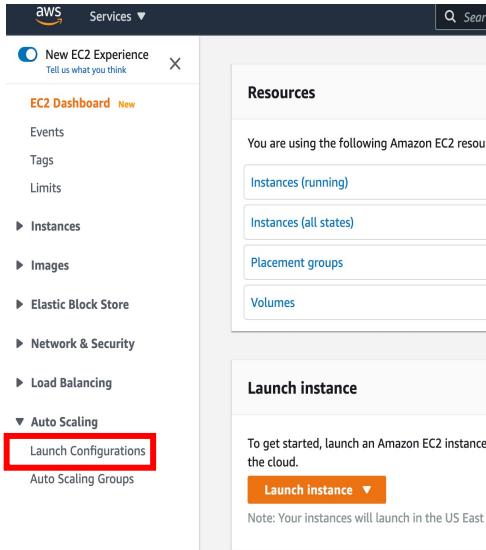
- Select instances and terminate them

| Instances (2/4) <a href="#">Info</a>  |                     |   |               |  |  |         |
|---|---------------------|---|---------------|--|--|---------|
| <input type="button" value="C"/> <a href="#">Connect</a> <a href="#">Instance state ▲</a> |                     |   |               |  |  |         |
| <input type="button" value="Filter instances"/>   |                     |   |               |  |  |         |
| Name  | Instance ID         | Instance state                              | Instance type | Status check                                       | Action   | Details |
| <input checked="" type="checkbox"/> my-ec2-01   | i-0f958f44ff98cde18 | <input checked="" type="checkbox"/> Running | t2.micro      | <input checked="" type="checkbox"/> 2/2 checks ... | <input type="button" value="Stop instance"/>   | No      |
| <input checked="" type="checkbox"/> my-ec2-02   | i-0a9bf59baea804c7c | <input checked="" type="checkbox"/> Running | t2.micro      | <input checked="" type="checkbox"/> 2/2 checks ... | <input type="button" value="Reboot instance"/> | No      |

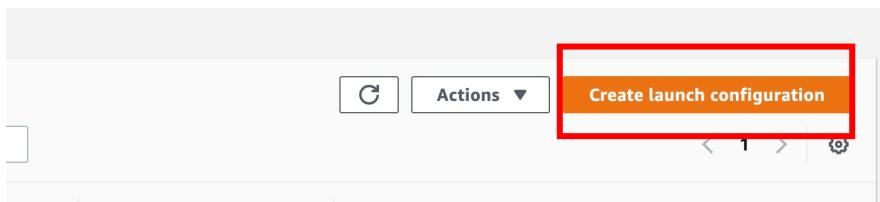
# Step 2: Create Launch Configuration

## Step 2.1: Select AWS Service

- On EC2 Dashboard



- Create Launch Configuration



# Step 2: Create Launch Configuration

## Step 2.2: Configuration

**Name:** my-launch-config-01

**AMI:** ami-01ee1e203f007488d

**Instance Type:** t2.micro

**Additional configuration -> Advanced details:**

### Userdata: (as text)

```
#!/bin/bash
sudo bash -c "cat >/etc/wordpress/config-us-east-1.elb.amazonaws.com.php" <<"EOT"
<?php
define('DB_NAME', 'wordpress');
define('DB_USER', 'admin');
define('DB_PASSWORD', 'adminpassword');
define('DB_HOST', '<REPLACE WITH RDS DB INSTANCE ENDPOINT>');
define('DB_COLLATE', 'utf8_general_ci');
define('WP_CONTENT_DIR', '/usr/share/wordpress/wp-content');
?>
EOT
sudo service apache2 restart
```

Username, password  
and hostname of RDS  
DB instance

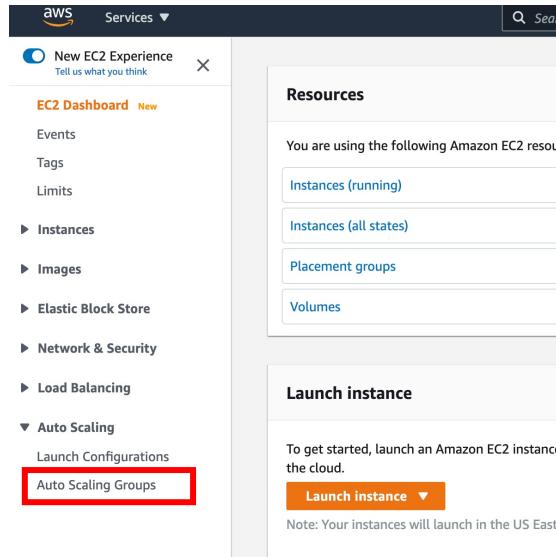
**Security Group:** my-ec2-security-group-01

**Key Pair:** my-ec2-key-pair

# Step 3: Create Auto Scaling Group

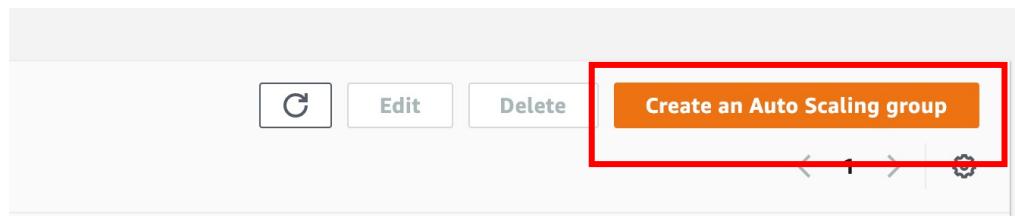
## Step 3.1: Select AWS Service

On EC2 Dashboard



The screenshot shows the AWS EC2 Dashboard. On the left sidebar, under the 'Auto Scaling' section, the 'Auto Scaling Groups' link is highlighted with a red box. The main content area displays sections for 'Resources' (Instances (running), Instances (all states), Placement groups, Volumes) and 'Launch instance' (with a 'Launch instance' button).

## Create Auto Scaling Group



The screenshot shows the first step of the 'Create Auto Scaling Group' wizard. It features a top navigation bar with 'Create', 'Edit', and 'Delete' buttons, followed by a large orange 'Create an Auto Scaling group' button. This button is highlighted with a red box. Below the button are navigation arrows and a gear icon.

# Step 3: Create Auto Scaling Group

## Step 3.2: Choose Launch Configuration

**Name:** my-auto-scaling-group

**Switch to launch configuration**

**Launch Template:** my-launch-config-01

Choose launch template or configuration [Info](#)

Specify a launch template that contains settings common to all EC2 instances that are launched by this Auto Scaling group. If you currently use launch configurations, you might consider migrating to launch templates.

|  |
|--|
| <b>Name</b>  |
| Auto Scaling group name<br>Enter a name to identify the group.<br><input type="text" value="my-auto-scaling group"/><br>Must be unique to this account in the current Region and no more than 255 characters.  |
| <b>Launch configuration</b> <a href="#">Info</a> <a href="#">Switch to launch template</a>   |
| Launch configuration<br>Choose a launch configuration that contains the instance-level settings, such as the Amazon Machine Image (AMI), instance type, key pair, and security groups.<br><input type="text" value="my-launch-config-02"/> <a href="#">▼</a> <a href="#">C</a> |

# Step 3: Create Auto Scaling Group

## Step 3.3: Configure Settings

- Instance Purchase Options:
  - Adhere to launch template
- Network:
  - VPC: my-vpc
  - Subnets: my-private-subnet-01
  - my-private-subnet-02

The screenshot shows the 'Configure Settings' step of the AWS Auto Scaling 'Create Auto Scaling Group' wizard. It is divided into two main sections: 'Instance purchase options' and 'Network'.

**Instance purchase options:** This section contains two options:

- Adhere to launch template**: Describes how the launch template determines the purchase option (On-Demand or Spot) and instance type.
- Combine purchase options and instance types**: Describes how to specify On-Demand and Spot capacity to launch multiple instance types.

**Network:** This section shows the selected VPC and subnets:

- VPC:** my-vpc (vpc-09b23f87f1ca7c8d5), 10.0.0.0/16
- Subnets:** Two subnets selected from us-east-1a:
  - us-east-1a | subnet-0e9d6a2747c15fa97 (my-private-subnet-01), 10.0.2.0/24
  - us-east-1a | subnet-04773e298ff9571f1 (my-private-subnet-02), 10.0.3.0/24

# Step 3: Create Auto Scaling Group

## Step 3.4: Configure advanced options

- Load balancing
  - Attach to an existing load balancer
  - Choose my-load-balancer-01
- Group Size
  - Set a constant capacity of 2
- Tags
  - Name: my-ec2
  - Tag new Instances - checked

**Load balancing - optional** Info

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 an existing load balancer**  
Select the load balancers that you want to attach to your Auto Scaling group.

Choose from your load balancer target groups  
This option allows you to attach Application, Network, or Gateway Load Balancers.

Choose from Classic Load Balancers

Classic Load Balancers  
Select Classic Load Balancers

my-load-balancer-01 X Classic Load Balancer

**Group size - optional** Info

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.

Desired capacity  
2

Minimum capacity  
2

Maximum capacity  
2 ▲ ▼

# Step 3: Create Auto Scaling Group

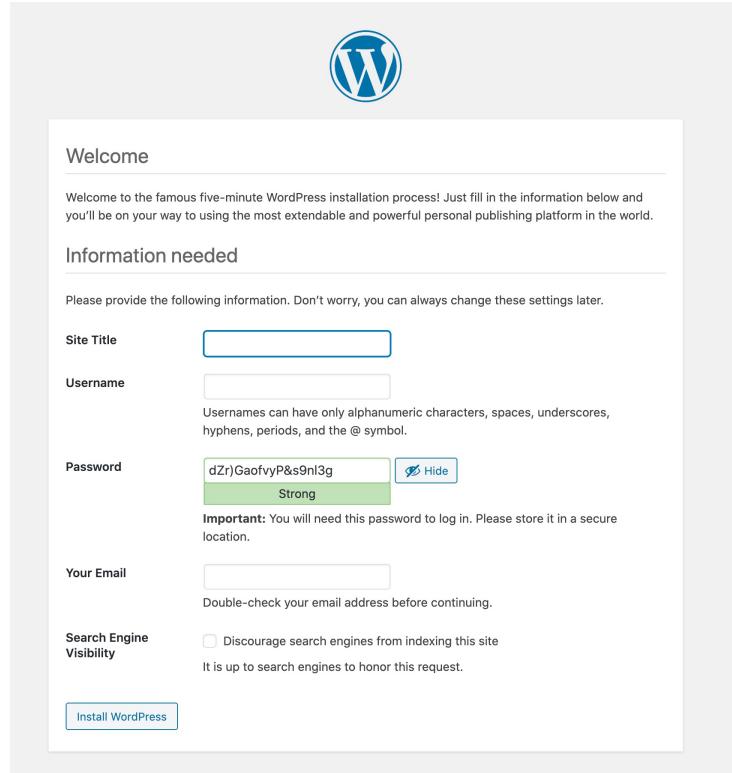
## Step 3.5: Verify

- Verify number of Instances (=2)
- Verify 2 Running Instances on EC2 dashboard

| Auto Scaling groups (1)                              |                                       |   |   |   |           |   |
|--|---------------------------------------|---|---|---|-----------|---|
| <input type="text"/> Search your Auto Scaling groups |                                       |   |   |   |           |   |
| <input type="checkbox"/>                             | Name                                  | ▼ | Launch template/configuration  | ▼ | Instances | ▼ |
| <input type="checkbox"/>                             | <a href="#">my-auto-scaling-group</a> |   | <a href="#">my-launch-config-01</a>   Version Default   |   | 2         |   |

# Milestone 3

Access ELB again as : <ELB-DNS-NAME>/blog



# Clean up

Remember to delete resources:

- Delete auto scaling group – deletes the instances too ([docs](#))
- Delete ELB ([docs](#))
- Delete RDS DB instance ([docs](#))
- Delete NAT Gateways ([docs](#))
- Release Elastic IPs associated with NATs ([docs](#))
- Delete Internet gateways ([docs](#))

# Summary

- Core services covered
  - Network: VPC
  - Compute: EC2, ELB, Auto Scaling
  - Storage: S3, EBS
  - Database: RDS
  - Monitoring: Cloudwatch