

Configure and Connect a MySQL Database Instance with a Web Server

Description:

You are working as a database administrator for an IT firm. You have been asked to create a new database instance on AWS cloud and connect it with the employee management portal hosted on a web server.

Problem statement:

Organization wants to deploy a new multi-tier application. The application will take live inputs from the employees and it will be hosted on a web server running on the AWS cloud.

The development team has asked you to set up the web server and configure it to scale automatically in cases of a traffic surge, to make the application highly available. They have also asked you to take the inputs from the employees and store them securely in the database.

Must use the following:

- Create a Database Instance with the following specifications:
 - Database creation method: Standard Create
 - Engine: MySQL
 - Database Instance size: db.t2.micro
- Create an EC2 Instance with the following specifications:
 - AMI: Amazon Linux
 - Region: Use only US East (N Virginia), us-east-1, and us-east-2
 - Instance types: t2.micro and t3.micro
 - Allowed EBS types: GP2 and Standard

Following requirements should be met:

- Follow the above-mentioned specifications
- Make sure that the Availability Zone is similar throughout the instances and volumes
- Ensure that the server scales automatically and the traffic is optimally routed among the scaled servers
- Document the step-by-step process involved in completing this task

Solution Overview and Configuration Steps

This multi-tier architecture consists of front end webserver and backend database. The frontend webserver will provide the application access to the users and take live inputs like Employee name and address from the user and insert the data into the backend data base.

For deploying the application, used AWS cloud services like Virtual private cloud, Internet Gateway, Security groups, subnets, EC2 Instances, RDS MySQL, Load balancer, Auto scaling group. This document will explain about how to create and configure all the required AWS services to design, deploy and test the application. Let's start with AWS services creation and configurations to achieve the final goal of this project.

The below diagram is high level architecture of the multi-tier application and we will follow this diagram for to creating services. I will explain more about each service information and configuration below

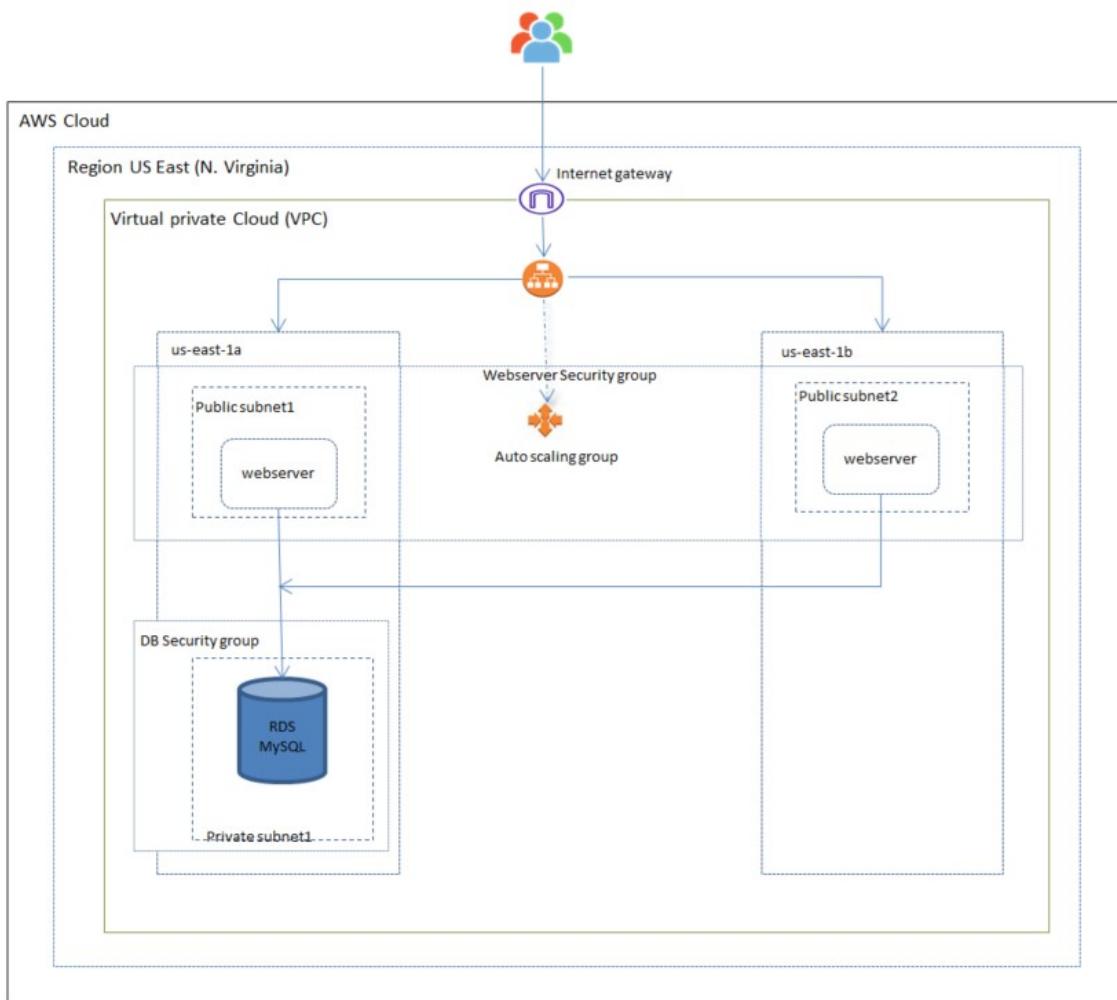


Diagram1: Application architecture

Create a Virtual private Cloud and all required components

- Looking at above architecture diagram, need to create VPC with below specified steps
- Logging into the AWS console, navigate to the VPC resource. Click “Create VPC” and you will see two options: “VPC only” or “VPC and more.” Select VPC and more because, we are creating a VPC with multiple subnets, availability zones, and more. So it's will create internet Gateway, subnets, Route tables and attach internet Gateway to VPC.
 - a) Logging into the AWS console



- b) Navigate to the VPC

- c) Click “Create VPC” and select “VPC and more” , during VPC creation selected the following inputs

IPv4 CIDR block: **10.0.0.0/16**

Tenancy: **Default**

Availability zones: **2 (us-east-1a & us-east-1b)**

Number of public subnets: **2**

Number of private subnets: **2**

NAT gateways: **None**

VPC endpoints: **None**

Enable DNS hostnames: **check**

Enable DNS resolutions: **check**

- d) After all input parameters filled click create VPC and will create the VPC with below workflow

- Create Security groups for EC2 instance
 - Navigated to Security group and click create security Group
 - Provided Security group name as “project” and update description
 - Select the VPC which was created as above step
 - Add inbound rules with SSH and HTTP for webserver

Type	Protocol	Port range	Source	Description
HTTP	TCP	80	Anywhere-IPv4	Allow webserver access from external
SSH	TCP	22	Anywhere-IPv4	Allow webserver access from external

- e) Click create Security Group

The screenshot shows the AWS VPC Security Groups console. A success message at the top indicates 'Security group sg-0762e59c0959bc5c7 - Project was created successfully'. The main card displays the security group details: name 'sg-0762e59c0959bc5c7 - Project', owner 'Y325331651673', and a description 'Allow webserver access from external'. It also shows the VPC ID 'vpc-0202aaefbd77d7a5bf'. The 'Inbound rules' section contains two entries:

Name	Security group rule...	IP version	Type	Protocol	Port range	Source	Description
sg-0762e59c0959bc5c7	sg-0762e59c0959bc5c7	IPv4	HTTP	TCP	80	0.0.0.0/0	
-	sg-0762e59c0959bc5c7	IPv4	SSH	TCP	22	0.0.0.0/0	

- Create Security group for RDS Database instance.

- Navigated to Security group and click create security Group
- Provided Security group name as “project-DB-SG” and update description
- Select the VPC which was created as above step
- Add inbound rules with SSH and HTTP for webserver

The screenshot shows the AWS EC2 Security Groups console. A success message at the top indicates 'Security group project-DB-SG was created successfully'. The main card displays the security group details: name 'project-DB-SG', description 'Allow db connection to EC2 instance', and VPC 'vpc-0f941d41a15436beb'. The 'Inbound rules' section contains one entry:

Type	Protocol	Port range	Source	Description - optional
MySQL/Aurora	TCP	3306	Anywhere-IPv4	0.0.0.0/0

- Click create Security Group

The screenshot shows the AWS EC2 Security Groups console. A success message at the top indicates 'Security group sg-0ff31f144ca9ccfe9 - project-DB-SG was created successfully'. The main card displays the security group details: name 'sg-0ff31f144ca9ccfe9 - project-DB-SG', description 'Allow db connection to EC2 instance', and VPC 'vpc-0f941d41a15436beb'. The 'Inbound rules' section contains one entry:

Name	Security group rule...	IP version	Type	Protocol	Port range	Source	Description
-	sg-0ff31f144ca9ccfe9	IPv4	MySQL/Aurora	TCP	3306	0.0.0.0/0	

Create a webserver Tier

In this section created first tier EC2 instance and it represents the webserver front end application for user inputs.

Launch an EC2 instance

Navigate to the EC2 Console and Click launch instance

During EC2 instance selected the following inputs

Updated name as “webserver”

Selected AMI “Amazon Linux 2023 AMI”

Instance type t2.micro

Created key pair with name project-key

Selected VPC “Project-vpc” which was created above

Selected subnet ua-east-1a public subnet

Selected auto assign public IP enable

Selected existing service group which was created above for EC2 instance called “project”

Create EC2 instance

Name and tags Info

Name
Webserver [Add additional tags](#)

Application and OS Images (Amazon Machine Image) Info

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

Quick Start

Amazon Linux macOS Ubuntu Windows Red Hat SUSE [Browse more AMIs](#)
Including AMIs from AWS, Marketplace and the Community

Amazon Machine Image (AMI)

Amazon Linux 2023 AMI
ami-06ca3ca175f37d6 (64-bit (x86)) / ami-0006abfd85caddfb2 (64-bit (Arm))
Virtualization: hvm ENA enabled: true Root device type: ebs [Free tier eligible](#)

Description
Amazon Linux 2023 AMI 2023.1.20230705.0 x86_64 HVM kernel-6.1

Architecture AMI ID
64-bit (x86) ami-06ca3ca175f37dd66 [Verified provider](#)

Summary

Number of instances Info
1

Software Image (AMI)
Amazon Linux 2023 AMI 2023.1.2... [read more](#)
ami-06ca3ca175f37d66

Virtual server type (instance type)
t2.micro

Firewall (security group)
New security group

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

Free tier: In your first year includes 750 hours of t2.micro (or t3.micro in the Regions in which t2.micro is unavailable) instance usage on free tier AMIs per month, 30 GiB of EBS storage, 2 million I/Os, 1 GiB of snapshots, and 100 GiB of bandwidth to the internet.

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

Instance type

t2.micro Family: t2 1 vCPU 1 GiB Memory Current generation: true Free tier eligible
On-Demand Windows pricing: 0.0162 USD per Hour
On-Demand Linux pricing: 0.0116 USD per Hour
On-Demand RHEL pricing: 0.0716 USD per Hour
On-Demand Linux pricing: 0.0116 USD per Hour

[All generations](#) [Compare instance types](#)

Key pair (login) Info

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

Key pair name - required project-key [Create new key pair](#)

Network settings Info

VPC - required info
vpc-0262aebdb75d7e38f (project-vpc) [Create new subnet](#)

Subnet info
subnet-03fb5bc1e7aa447e3 project-subnet-public1-us-east-1a
VPC: vpc-0262aebdb75d7e38f Owner: 152531651673 Availability Zone: us-east-1a IP addresses available: 4091 CIDR: 10.0.0.0/20

Auto-assign public IP info
Enable

Firewall (security groups) info
A security group is a set of firewall rules that control the traffic for your instance. Add rules to allow specific traffic to reach your instance.
 Create security group Select existing security group

Common security groups info
Select security groups
Project sg-0762e59c0959bc5c7 [Create new security group](#) [Compare security group rules](#)

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

Summary

Number of instances Info
1

Software Image (AMI)
Amazon Linux 2023 AMI 2023.1.2... [read more](#)
ami-06ca3ca175f37d66

Virtual server type (instance type)
t2.micro

Firewall (security group)
Project

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

Free tier: In your first year includes 750 hours of t2.micro (or t3.micro in the Regions in which t2.micro is unavailable) instance usage on free tier AMIs per month, 30 GiB of EBS storage, 2 million I/Os, 1 GiB of snapshots, and 100 GiB of bandwidth to the internet.

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

Now the EC2 Instance created successfully.

<https://us-east-1.console.aws.amazon.com/ec2/home?region=us-east-1#LaunchInstances>

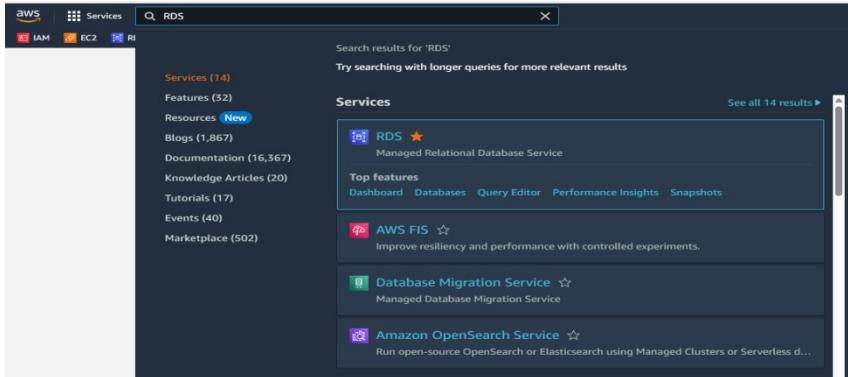
EC2 > Instances > Launch an instance

Success
Successfully initiated launch of instance i-0b75b141d7761e
[Launch log](#)

Next Steps

Create RDS MySQL Instance

Navigate to RDS and click RDS



Selected “Standard Create”

Selected Engine options MySQL

Selected Free tier and DB instance name database-1

Kept the username as admin and provided Master password for connect database

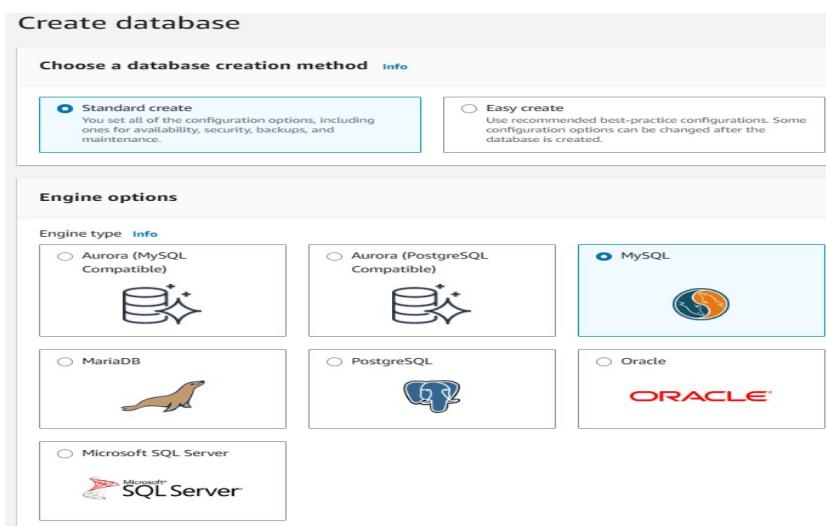
Selected instance class t2.micro

Selected max storage threshold value 100

Selected connectivity to an EC2 compute resources and selected the EC2 instance name which was created above.

Selected Security group “project-db-sg” which was created for DB server instance

Click create Database



IAM EC2 RDS VPC

Show versions that support the Multi-AZ DB Cluster [Info](#)
 Create a A Multi-AZ DB cluster with one primary DB instance and two readable standby DB instances. Multi-AZ DB clusters provide up to 2x faster transaction commit latency and automatic failover in typically under 35 seconds.

Show versions that support the Amazon RDS Optimized Writes [Info](#)
 Amazon RDS Optimized Writes improves write throughput by up to 2x at no additional cost.

Engine Version

MySQL 8.0.33

Templates
 Choose a sample template to meet your use case.

Production
 Use defaults for high availability and fast, consistent performance.

Dev/Test
 This instance is intended for development use outside of a production environment.

Free tier
 Use RDS Free Tier to develop new applications, test existing applications, or gain hands-on experience with Amazon RDS.
[Info](#)

Availability and durability

Deployment options [Info](#)
 The deployment options below are limited to those supported by the engine you selected above.

Multi-AZ DB Cluster - new
 Creates a DB cluster with a primary DB instance and two readable standby DB instances, with each DB instance in a different Availability Zone (AZ). Provides high availability, data redundancy and increases capacity to serve read workloads.

Multi-AZ DB instance (not supported for Multi-AZ DB cluster snapshot)
 Creates a primary DB instance and a standby DB instance in a different AZ. Provides high availability and data redundancy, but the standby DB instance doesn't support connections for read workloads.

Single DB instance (not supported for Multi-AZ DB cluster snapshot)
 Creates a single DB instance with no standby DB instances.

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.

database-1

The DB instance identifier is case-insensitive, but is stored as all lowercase (as in "mydbinstance"). Constraints: 1 to 60 alphanumeric characters or hyphens. 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.

admin

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

Manage master credentials in AWS Secrets Manager
 Manage master user credentials in Secrets Manager. RDS can generate a password for you and manage it throughout its lifecycle.

If you manage the master user credentials in Secrets Manager, some RDS features aren't supported.
[Learn more](#)

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 master password [Info](#)

Instance configuration

The DB instance configuration options below are limited to those supported by the engine that you selected above.

Amazon RDS Optimized Writes - new [Info](#)
 Show instance classes that support Amazon RDS Optimized Writes

DB instance class [Info](#)
 Standard classes (includes m classes)
 Memory optimized classes (includes r and x classes)
 Burstable classes (includes t classes)

db.t2.micro
1 vCPUs 1 GiB RAM Not EBS Optimized

Include previous generation classes

Storage

Storage type [Info](#)
General Purpose SSD (gp2)
Baseline performance determined by volume size

Maximum storage threshold [Info](#)

Charges will apply when your database autoscales to the specified threshold

100 GiB

The minimum value is 22 GiB and the maximum value is 6,144 GiB

Connectivity [Info](#)

Compute resource
Choose whether to set up a connection to a compute resource for this database. Setting up a connection will automatically change connectivity settings so that the compute resource can connect to this database.

Don't connect to an EC2 compute resource
Don't set up a connection to a compute resource for this database. You can manually set up a connection to a compute resource later.

Connect to an EC2 compute resource
Set up a connection to an EC2 compute resource for this database.

EC2 instance [Info](#)
Choose the EC2 instance to add as the compute resource for this database. A VPC security group is added to this EC2 instance. A VPC security group is also added to the database with an inbound rule that allows the EC2 instance to access the database.

i-0b75b1f417d77616e
Webserver

Some VPC settings can't be changed when a compute resource is added
Adding an EC2 compute resource automatically selects the VPC, DB subnet group, and public access settings for this database. To allow the EC2 instance to access the database, a VPC security group rds-ec2-X is added to the database and another called ec2-rds-X to the EC2 instance. You can remove the new security group for the database only by removing the compute resource.

Virtual private cloud (VPC) [Info](#)
Choose the VPC. The VPC defines the virtual networking environment for this DB instance.

project-vpc (vpc-0262aehdb75d7e38f)
4 Subnets, 2 Availability Zones

Only VPCs with a corresponding DB subnet group are listed.

Choose the VPC. The VPC defines the virtual networking environment for this DB instance.
project-vpc (vpc-0262aehdb75d7e38f)
4 Subnets, 2 Availability Zones

Only VPCs with a corresponding DB subnet group are listed.

After a database is created, you can't change its VPC.

DB subnet group [Info](#)
Choose the DB subnet group. The DB subnet group defines which subnets and IP ranges the DB instance can use in the VPC that you selected.

Choose existing
Choose existing DB subnet group

Automatic setup
RDS creates a new subnet group for you or reuses an existing subnet group

DB subnet group name
rds-ec2-db-subnet-group-1

New DB subnet group created.

Public access [Info](#)
 Yes
RDS assigns a public IP address to the database. Amazon EC2 instances and other resources outside of the VPC can connect to your database. Resources inside the VPC can also connect to the database. Choose one or more VPC security groups that specify which resources can connect to the database.

No
RDS doesn't assign a public IP address to the database. Only Amazon EC2 instances and other resources inside the VPC can connect to your database. Choose one or more VPC security groups that specify which resources can connect to the database.

VPC security group (firewall) [Info](#)
Choose one or more VPC security groups to allow access to your database. Make sure that the security group rules allow the appropriate incoming traffic.

Choose existing
Choose existing VPC security groups

Create new
Create new VPC security group

Additional VPC security group
Choose one or more options
project-DB-SG X

Creating database database-1
Your database might take a few minutes to launch.
You can use settings from database-1 to simplify configuration of suggested database add-ons while we finish creating your DB for you.
How was your experience creating an Amazon RDS database? [Provide feedback](#)

RDS > Databases

Databases (1)

DB identifier	Status	Role	Engine	Region & AZ	Size	Actions	CPU	Current activity
database-1	Creating	Instance	MySQL Community	us-east-1a	db.t2.micro	-	-	-

Successfully set up a connection between database-1 and EC2 instance i-0cd6bdca2a206537f3.

Successfully created database database-1
You can use settings from database-1 to simplify configuration of suggested database add-ons while we finish creating your DB for you.
How was your experience creating an Amazon RDS database? [Provide feedback](#)

RDS > Databases

Databases (1)

DB identifier	Status	Role	Engine	Region & AZ	Size	Actions	CPU	Current activity	Maintenance
database-1	Available	Instance	MySQL Community	us-east-1a	db.t2.micro	-	24.07%	0 Connections	none

Install an Apache webserver with PHP and MariaDB

Connect to Webserver EC2 instance which was created above.

Navigate to EC2 instance and select the webserver EC2 instance and click connect

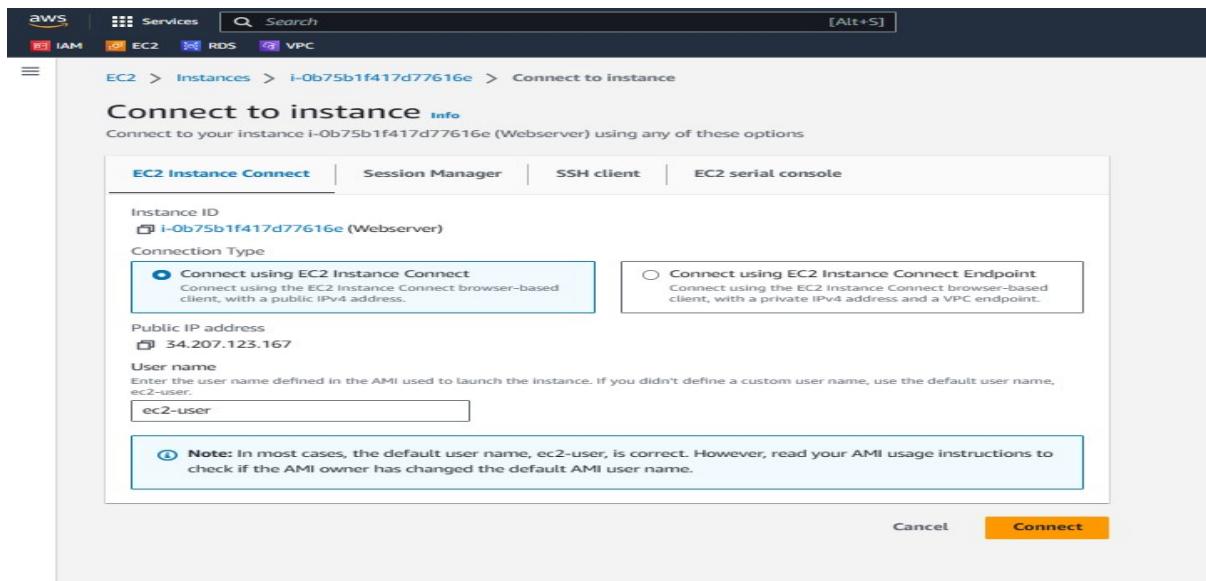
Connect to EC2 instance connect and click connect

New EC2 Experience Tell us what you think

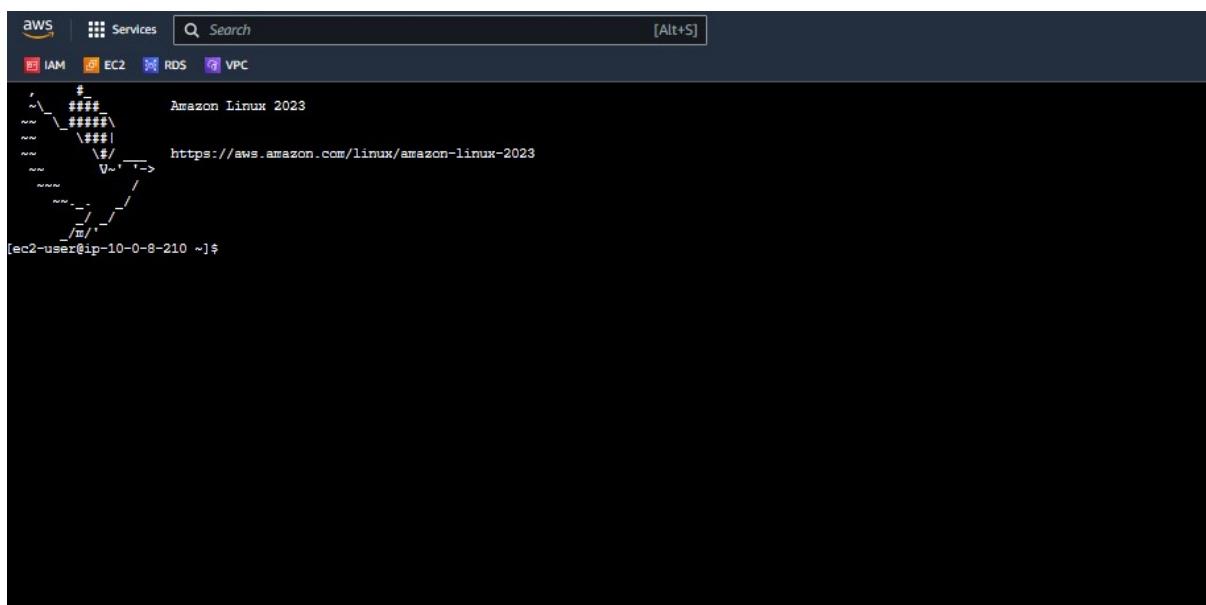
EC2 > Instances > i-0b75b1f417d77616e

Instance summary for i-0b75b1f417d77616e (Webserver) [Info](#)
Updated less than a minute ago

Instance ID	i-0b75b1f417d77616e (Webserver)	Public IPv4 address	34.207.123.167 Open address	Private IPv4 address	10.0.8.210
IPv6 address	-	Instance state	Running	Public IPv6 DNS	i-ec2-34-207-123-167.compute-1.amazonaws.com Open address
Hostname type	IP name: ip-10-0-8-210.ec2.internal	Private IP DNS name (IPv4 only)	ip-10-0-8-210.ec2.internal	Elastic IP addresses	-
Answer private resource DNS name	-	Instance type	t2.micro	AWS Compute Optimizer finding	Opt-in to AWS Compute Optimizer for recommendations. Learn more
Auto-assigned IP address	34.207.123.167 (Public IP)	VPC ID	i-0c4032aeb0d7567e3ff (project-vpc)	Auto Scaling Group name	-
UAMI Role	-	Subnet ID	subnet-03f850e1e7aa447e3 (project-subnet-public1-us-east-1a)		
BIDS/2	Required				



Connected EC2 instance console



Install an Apache web server with PHP and MariaDB

Connect EC2 instance using



Get the latest bug fixes and security updates by updating the software on your EC2 instance
sudo dnf update -y

install the Apache web server, PHP, and MariaDB software using the following command.
`sudo dnf install -y httpd php php-mysqli mariadb105`

Start the web server with the command shown following.

Configure the web server to start with each system boot using the `sudo systemctl` command.

```
$sudo systemctl start httpd
```

```
$sudo systemctl enable httpd
```

```
[ec2-user@ip-10-0-8-210 ~]$ sudo systemctl start httpd
[ec2-user@ip-10-0-8-210 ~]$ sudo systemctl enable httpd
Created symlink /etc/systemd/system/multi-user.target.wants/httpd.service → /usr/lib/systemd/system/httpd.service.
[ec2-user@ip-10-0-8-210 ~]$ [REDACTED]
```

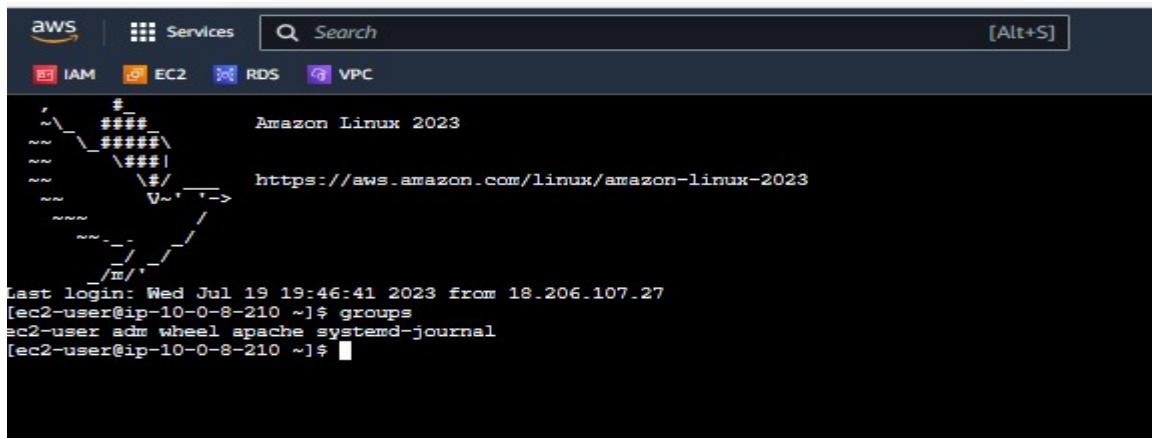
To set file permissions for the Apache web server

Add the ec2-user user to the apache group.

```
$sudo usermod -a -G apache ec2-user
```

```
[ec2-user@ip-10-0-8-210 ~]$ sudo usermod -a -G apache ec2-user
[ec2-user@ip-10-0-8-210 ~]$ [REDACTED]
```

Exit and Log back in again and verify that the **apache** group exists with the **groups** command.



```
#          Amazon Linux 2023
#          #####
#          #####
#          \###|
#          \#/   https://aws.amazon.com/linux/amazon-linux-2023
#          V~'-->
#          ~~~
#          ~~~
#          /m/
Last login: Wed Jul 19 19:46:41 2023 from 18.206.107.27
[ec2-user@ip-10-0-8-210 ~]$ groups
ec2-user adm wheel apache systemd-journal
[ec2-user@ip-10-0-8-210 ~]$ [REDACTED]
```

Change the group ownership of the **/var/www** directory and its contents to the **apache** group.

```
$sudo chown -R ec2-user:apache /var/www
```

Change the directory permissions of **/var/www** and its subdirectories to add group write permissions and set the group ID on subdirectories created in the future.

```
$sudo chmod 2775 /var/www
```

```
$find /var/www -type d -exec sudo chmod 2775 {} \;
```

Recursively change the permissions for files in the **/var/www** directory and its subdirectories to add group write permissions.

```
$find /var/www -type f -exec sudo chmod 0664 {} \;
```

Connected Apache web server to DB instance
Check the DB endpoint from database instance

AWS Services Search [Alt+S]

RDS > Databases > database-1

database-1

Summary

DB identifier database-1	CPU <div style="width: 4.83%;">4.83%</div>	Status Available
Role Instance	Current activity <div style="width: 0%;">0 Connections</div>	Engine MySQL Community

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

Connectivity & security

Endpoint & port	Networking	Security
Endpoint database-1.curuv6e7ll.us-east-1.rds.amazonaws.com	Availability Zone us-east-1a	VPC security groups rds-ec2-1 (sg-071e7112753a19f2c) Active project-DB-SG (sg-0dfb7b87ff23d7bf) Active
Port 3306	VPC project-vpc (vpc-032aa3ebf25c6d4a5)	

Create DATABASE in MYSQL DB

```
[ec2-user@ip-10-0-8-210 ~]$ mysql -h database-2.ccies94klf2v.us-east-1.rds.amazonaws.com -u admin -p
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MySQL connection id is 22
Server version: 8.0.33 Source distribution

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

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

MySQL [(none)]> CREATE DATABASE sample;
Query OK, 1 row affected (0.007 sec)
```

```
MySQL [(none)]> SHOW DATABASES;
+-----+
| Database |
+-----+
| information_schema |
| mysql |
| performance_schema |
| sample |
| sys |
+-----+
5 rows in set (0.006 sec)

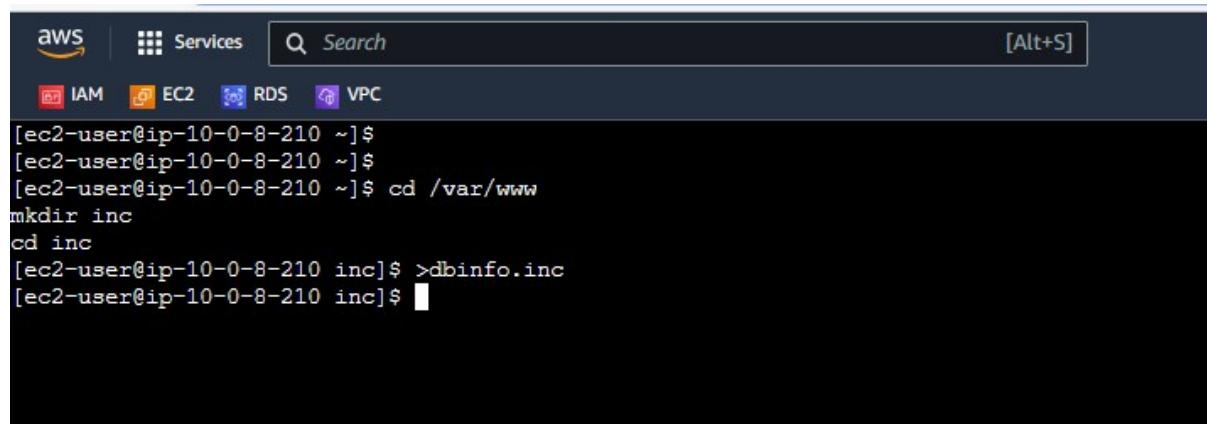
MySQL [(none)]> █
```

To add content to the Apache web server that connects to your DB instance

```
$cd /var/www
$mkdir inc
$cd inc
```

Created a new file in the `inc` directory named `dbinfo.inc`, and then edit the file by calling nano (or the editor of your choice).

```
>dbinfo.inc
```



The screenshot shows a terminal window within the AWS Cloud9 IDE. At the top, there's a navigation bar with the AWS logo, a 'Services' dropdown, a search bar, and a '[Alt+S]' keyboard shortcut. Below the bar, there are links for IAM, EC2, RDS, and VPC. The main area of the terminal shows a command-line session:

```
[ec2-user@ip-10-0-8-210 ~]$
[ec2-user@ip-10-0-8-210 ~]$
[ec2-user@ip-10-0-8-210 ~]$ cd /var/www
mkdir inc
cd inc
[ec2-user@ip-10-0-8-210 inc]$ >dbinfo.inc
[ec2-user@ip-10-0-8-210 inc]$ █
```

Added the following contents to the `dbinfo.inc` file. Here, `db_instance_endpoint` is your DB instance endpoint, without the port, for your DB instance.

```
<?php

define('DB_SERVER', 'db_instance_endpoint');
define('DB_USERNAME', 'tutorial_user');
define('DB_PASSWORD', 'master password');
define('DB_DATABASE', 'sample');

?>
```

```
[ec2-user@ip-10-0-10-143 inc]$ cat dbinfo.inc
<?php

define('DB_SERVER', 'database-1.curiuv6e7lii.us-east-1.rds.amazonaws.com');
define('DB_USERNAME', 'admin');
define('DB_PASSWORD', 'REDACTED');
define('DB_DATABASE', 'sample');

?>
```

Save and close the dbinfo.inc file.
Change the directory to /var/www/html.
Add the following code in SamplePage.php

```
<?php include "../inc/dbinfo.inc"; ?>
<html>
<body>
<h1>Employees Data</h1>
<?php

/* Connect to MySQL and select the database. */
$connection = mysqli_connect(DB_SERVER, DB_USERNAME, DB_PASSWORD);

if (mysqli_connect_errno()) echo "Failed to connect to MySQL: " . mysqli_connect_error();

$database = mysqli_select_db($connection, DB_DATABASE);

/* Ensure that the EMPLOYEES table exists. */
VerifyEmployeesTable($connection, DB_DATABASE);

/* If input fields are populated, add a row to the EMPLOYEES table. */
$employee_name = htmlentities($_POST['NAME']);
$employee_address = htmlentities($_POST['ADDRESS']);

if (strlen($employee_name) || strlen($employee_address)) {
    AddEmployee($connection, $employee_name, $employee_address);
}
?>

<!-- Input form -->
<form action="<?PHP echo $_SERVER['SCRIPT_NAME'] ?>" method="POST">
<table border="0">
    <tr>
        <td>NAME</td>
        <td>ADDRESS</td>
    </tr>
    <tr>
```

```

<td>
  <input type="text" name="NAME" maxlength="45" size="30" />
</td>
<td>
  <input type="text" name="ADDRESS" maxlength="90" size="60" />
</td>
<td>
  <input type="submit" value="Add Data" />
</td>
</tr>
</table>
</form>

<!-- Display table data. -->
<table border="1" cellpadding="2" cellspacing="2">
<tr>
  <td>ID</td>
  <td>NAME</td>
  <td>ADDRESS</td>
</tr>

<?php

$result = mysqli_query($connection, "SELECT * FROM EMPLOYEES");

while($query_data = mysqli_fetch_row($result)) {
  echo "<tr>";
  echo "<td>",$query_data[0], "</td>",
    "<td>",$query_data[1], "</td>",
    "<td>",$query_data[2], "</td>";
  echo "</tr>";
}
?>

</table>

<!-- Clean up. -->
<?php

mysqli_free_result($result);
mysqli_close($connection);

?>

</body>
</html>

<?php

/* Add an employee to the table. */

```

```

function AddEmployee($connection, $name, $address) {
    $n = mysqli_real_escape_string($connection, $name);
    $a = mysqli_real_escape_string($connection, $address);

    $query = "INSERT INTO EMPLOYEES (NAME, ADDRESS) VALUES ('$n', '$a');";

    if(!mysqli_query($connection, $query)) echo("<p>Error adding employee data.</p>");
}

/* Check whether the table exists and, if not, create it. */
function VerifyEmployeesTable($connection, $dbName) {
    if(!TableExists("EMPLOYEES", $connection, $dbName))
    {
        $query = "CREATE TABLE EMPLOYEES (
            ID int(11) UNSIGNED AUTO_INCREMENT PRIMARY KEY,
            NAME VARCHAR(45),
            ADDRESS VARCHAR(90)
        )";

        if(!mysqli_query($connection, $query)) echo("<p>Error creating table.</p>");
    }
}

/* Check for the existence of a table. */
function TableExists($tableName, $connection, $dbName) {
    $t = mysqli_real_escape_string($connection, $tableName);
    $d = mysqli_real_escape_string($connection, $dbName);

    $checktable = mysqli_query($connection,
        "SELECT TABLE_NAME FROM information_schema.TABLES WHERE TABLE_NAME = '$t' AND
        TABLE_SCHEMA = '$d'");
}

if(mysqli_num_rows($checktable) > 0) return true;

return false;
}
?>

```

```
[ec2-user@ip-10-0-8-210 inc]$ cd /var/www/html
[ec2-user@ip-10-0-8-210 html]$ >SamplePage.php
[ec2-user@ip-10-0-8-210 html]$ vi SamplePage.php
[ec2-user@ip-10-0-8-210 html]$ █
```

Verify that your web server successfully connects to your DB instance by opening a web browser and browsing

Get the public IP4 DNS name from the webserver EC2 instance.

The screenshot shows the AWS EC2 Instance Summary page for an instance named 'i-0b75b1f417d77616e (Webserver)'. The Public IPv4 DNS field, which contains 'ec2-34-207-123-167.compute-1.amazonaws.com', is highlighted with a red box.

Copy the webserver Instance public IP4 DNS name and try to open the URL from web browser. Web browser automatically redirected to the secure https site but changed the URL to use http. Success! I am able to see the webpage with input variables and table

The screenshot shows a web browser displaying a PHP page titled 'Employees Data'. The page contains a table with columns 'NAME' and 'ADDRESS'. The table data is highlighted with a red box.

ID	NAME	ADDRESS
1	Nageswara Rao Goparaju	Hyderabad
2	Harsha Vardhan Goparaju	Bangalore

Now adding data into the database using input values from the webserver. Successfully able to add data into the database and table updated with the updated information

The screenshot shows a web browser displaying a PHP page titled 'Employees Data'. The page contains a table with columns 'NAME' and 'ADDRESS'. A new row has been added to the table, containing the data for 'Harsha Vardhan Goparaju'.

ID	NAME	ADDRESS
1	Nageswara Rao Goparaju	Hyderabad
2	Harsha Vardhan Goparaju	Bangalore

Creating Golden Image AMI and Instance template

Now created a Golden image of the webserver and will be used for Instance template. Go to EC2 instances and select the webserver and click Actions → image and templates → Create Image

The screenshot shows the AWS EC2 Instances page. A single instance named 'webserver' is listed, which is currently running. The Actions menu is open, and the 'Image and templates' option is highlighted.

Provided Image name as webserver-golden-image and click image

The screenshot shows the 'Create Image' wizard. Step 1: Set instance details. The instance ID is i-0c57987cae3f07f83. The image name is set to 'webserver-golden-image'. The image description is 'Golden image for webserver'. Under 'Instance volumes', there is one EBS volume of size 8 GiB, type General Purpose SSD, with throughput of 3000 MiB/s and encryption enabled.

Create EC2 instance Template.

Go to EC2 instances and select the webserver and click Actions → image and templates → Create template from instance

Provided Launch template name “webserver-template”

Selected owned by me “webserver-golden-image”

Selected instance type t2.micro

Selected the existing key pair name project-key

Selected the subnet from us-east-1a public

Selected the existing security group which was created for webserver

Click create launch instance

The screenshot shows the AWS EC2 Instances page. The 'webserver' instance is still running. The Actions menu is open, and the 'Image and templates' option is highlighted, indicating the creation process is still in progress.

Create launch template

Creating a launch template allows you to create a saved instance configuration that can be reused, shared and launched at a later time. Templates can have multiple versions.

Launch template name and description

Source instance: i-0c57987cae5f07f83

Launch template name - required: webserver-template
Must be unique to this account. Max 128 chars. No spaces or special characters like '&', '*', '@'.

Template version description: A webserver
Max 255 chars

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

► Template tags

Launch template contents
Specify the details of your launch template below. Leaving a field blank will result in the field not being included in the launch template.

Summary

Software Image (AMI)
webserver image
ami-04e2c62be8bd77e46

Virtual server type (instance type)
t2.micro

Firewall (security group)
2 security groups

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

Free tier: In your first year includes 750 hours of t2.micro (or t3.micro in the Regions in which t2.micro is unavailable) instance usage on free tier AMIs per month, 30 GiB of EBS storage, 2 million IOs, 1 GB of snapshots, and 100 GB of bandwidth to the internet.

Create launch template

AMI from catalog | **Recents** | **My AMIs** | **Quick Start**

Don't include in launch template Owned by me Shared with me

Amazon Machine Image (AMI)

webserver-golden-image
ami-04e2c62be8bd77e46
2023-07-22T07:38:40.00Z Virtualization: hvm ENA enabled: true Root device type: ebs boot mode: uefi-preferred

Description: webserver image

Architecture: x86_64 AMI ID: ami-04e2c62be8bd77e46

Instance type [Info](#)

Manually select instance type
Select an instance type that meets your computing, memory, networking, or storage needs

Specify instance type attributes
Specify instance attributes that match your compute requirements

Instance type

t2.micro Family: t2 1 vCPU 1 GiB Memory Current generation: true
On-Demand Windows pricing: 0.0162 USD per Hour
On-Demand SUSE pricing: 0.0116 USD per Hour
On-Demand RHEL pricing: 0.0716 USD per Hour
On-Demand Linux pricing: 0.0116 USD per Hour

Free tier eligible All generations

Summary

Software Image (AMI)
webserver image
ami-04e2c62be8bd77e46

Virtual server type (instance type)
t2.micro

Firewall (security group)
2 security groups

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

Free tier: In your first year includes 750 hours of t2.micro (or t3.micro in the Regions in which t2.micro is unavailable) instance usage on free tier AMIs per month, 30 GiB of EBS storage, 2 million IOs, 1 GB of snapshots, and 100 GB of bandwidth to the internet.

Create launch template

Key pair name: project-key

Template value:

Network settings

Subnet info: subnet-0c05909e5a18fac9b (project-subnet-public1-us-east-1a)

VPC: vpc-0765d503dd08f4e60 Owner: 667778984468 Availability Zone: us-east-1a IP addresses available: 4090 CIDR: 10.0.0.0/20

Create new subnet

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

Firewall (security groups)

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

Select existing security group (selected) or Create security group

Common security groups: Select security groups

project-web sg-08882068c03a8469f (VPC: vpc-0765d503dd08f4e60)

Show all selected (+1)

Compare security group rules

Storage (volumes)

EBS Volumes

Volume 1 (Template and AMI Root) (8 GiB, EBS, General purpose SSD (gp3))

AMI Volumes are not included in the template unless modified

Summary

Software image (AMI): webserver image ami-04e2c62be8bd77e46

Virtual server type (instance type): t2.micro

Firewall (security group): 2 security groups

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

Free tier: In your first year includes 750 hours of t2.micro (or t3.micro in the Regions in which t2.micro is unavailable) instance usage on free tier AMI per month, 30 GiB of EBS storage, 2 million I/Os, 1 GB of snapshots, and 100 GB of bandwidth to the internet.

Cancel

Successfully Instance template created and this will be used for Auto scaling group

Success Successfully created webserver-template (lt-0cc3488610265ff6e)

Actions log

Next steps

Launch an instance With On-Demand Instances, you pay for compute capacity by the second (for Linux, with a minimum of 60 seconds) or by the hour (for all other operating systems) with no long-term commitments or upfront payments. Launch an On-Demand Instance from your launch template.

Launch instance from this template

Create an Auto Scaling group from your template Amazon EC2 Auto Scaling helps you maintain application availability and allows you to scale your Amazon EC2 capacity up or down automatically according to conditions you define. You can use Auto Scaling to help ensure that you are running your desired number of Amazon EC2 instances during demand spikes to maintain performance and decrease capacity during lulls to reduce costs.

Create Auto Scaling group

Create Spot Fleet A Spot Instance is an unused EC2 instance that is available for less than the On-Demand price. Because Spot Instances enable you to request unused EC2 instances at steep discounts, you can lower your Amazon EC2 costs significantly. The hourly price for a Spot Instance (of each instance type in each Availability Zone) is set by Amazon EC2, and adjusted gradually based on the long-term supply of and demand for Spot Instances. Spot instances are well-suited for data-analysis, batch jobs, background processing, and optional tasks.

Create Spot Fleet

Creating Auto scaling group

Create Target Group with following inputs:

Target type: Instances

Target group name: Project-lb-TG

Protocol: HTTP port 80

VPC: Select the created VPC "project-vpc"

Create Target Group

The screenshot shows the AWS CloudWatch Metrics console. A log stream named 'AWS CloudWatch Metrics' is selected. The log entries are listed in a table with columns for 'Time' (UTC), 'Metric Name', 'Value', and 'Source ARN'. The first entry is from 'aws_lambda_function_invocation' with a value of '1' and a source ARN of 'arn:aws:lambda:us-east-1:123456789012:function:my-lambda-function'. The second entry is from 'aws_lambda_invocation_error' with a value of '1' and a source ARN of 'arn:aws:lambda:us-east-1:123456789012:function:my-lambda-function'.

Target group name

Project-lb-TG

A maximum of 32 alphanumeric characters including hyphens are allowed, but the name must not begin or end with a hyphen.

Protocol	Port
HTTP	: 80 1-65535

VPC

Select the VPC with the instances that you want to include in the target group.

project-vpc
vpc-0262aebdb75d7e38f
IPv4: 10.0.0.0/16

Protocol version

HTTP1
Send requests to targets using HTTP/1.1. Supported when the request protocol is HTTP/1.1 or HTTP/2.

HTTP2
Send requests to targets using HTTP/2. Supported when the request protocol is HTTP/2 or gRPC, but gRPC-specific features are not available.

gRPC
Send requests to targets using gRPC. Supported when the request protocol is gRPC.

Successfully created Target Group

Screenshot of the AWS EC2 Target groups 'Create target group' wizard, Step 2: Register targets.

Register targets

This is an optional step to create a target group. However, to ensure that your load balancer routes traffic to this target group you must register your targets.

Available instances (1)

Instance ID	Name	State	Security groups	Zone	Subnet ID
i-0c57987cae3f07fb3	webserver	Running	ec2-rds-1, project-web	us-east-1a	subnet-0c05909e5a18fac9b

Ports for the selected instances

Ports for routing traffic to the selected instances.
80
1-65535 (separate multiple ports with comma)
Include as pending below

Screenshot of the AWS EC2 Target groups 'Create target group' wizard, Step 3: Review and Create.

Successfully created target group: project-lb-TG

Target groups (1) info

Name	ARN	Port	Protocol	Target type	Load balancer	VPC ID
project-lb-TG	arn:aws:elasticloadbalanc...	80	HTTP	Instance	None associated	vpc-0765d503dd008f4e60

Created application load balancer

Load balancer types

Application Load Balancer [Info](#)

Choose an Application Load Balancer when you need a flexible feature set for your 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.

[Create](#)

Network Load Balancer [Info](#)

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 applications. Operating at the connection level, Network Load Balancers are capable of handling millions of requests per second securely while maintaining ultra-low latencies.

[Create](#)

Gateway Load Balancer [Info](#)

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.

[Create](#)

VPC [Info](#)

Select the virtual private cloud (VPC) for your targets or you can [create a new VPC](#). Only VPCs with an internet gateway are enabled for selection. The selected VPC can't be changed after the load balancer is created. To confirm the VPC for your targets, view your [target groups](#).

project-vpc vpc-0765d503dd08f4e60 IPv4: 10.0.0.16	C
---	-------------------

Mappings [Info](#)

Select at least two Availability Zones and one subnet per zone. The load balancer routes traffic to targets in these Availability Zones only. Availability Zones that are not supported by the load balancer or the VPC are not available for selection.

<input checked="" type="checkbox"/> us-east-1a (use1-az2)	
Subnet	
subnet-0c05909e5a18fac9b	project-subnet-public1-us-east-1a ▾
IPv4 address	
Assigned by AWS	
<input checked="" type="checkbox"/> us-east-1b (use1-az4)	
Subnet	
subnet-0c616971c0b098f0b	project-subnet-public2-us-east-1b ▾
IPv4 address	
Assigned by AWS	

Security groups

Select up to 5 security groups

project-web sg-08882068c03a8469f X	VPC: vpc-0765d503dd08f4e60
------------------------------------	----------------------------

Listeners and routing [Info](#)

A listener is a process that checks for connection requests using the port and protocol you configure. The rules that you define for a listener determine how the load balancer routes requests to its registered targets.

▼ Listener HTTP:80

Protocol: HTTP	Port: 80	Default action: Info	Remove
Forward to: project-lb-TG	Target type: Instance, IPv4	HTTP	C
Create target group			

Listener tags - optional

Consider adding tags to your listener. Tags enable you to categorize your AWS resources so you can more easily manage them.

[Add listener tag](#)

You can add up to 50 more tags.

aws Services Search [Alt+S]

IAM EC2 RDS VPC

▼ Add-on services - optional

Additional AWS services can be integrated with this load balancer at launch. You can also add these and other services after your load balancer is created by reviewing the "Integrated Services" tab for the selected load balancer.

AWS Global Accelerator Info

Create an accelerator to get static IP addresses and improve the performance and availability of your applications. [Additional charges apply](#)

► Load balancer tags - optional

Consider adding tags to your load balancer. Tags enable you to categorize your AWS resources so you can more easily manage them. The 'Key' is required, but 'Value' is optional. For example, you can have Key = production-webserver, or Key = webserver, and Value = production.

Summary

Review and confirm your configurations. [Estimate cost](#)

Basic configuration Edit	Security groups Edit	Network mapping Edit	Listeners and routing Edit
project-lb <ul style="list-style-type: none">Internet-facingIPv4	project-web sg-08882068c03a8469f	VPC vpc-0765d503dd08f4e60 project-vpc <ul style="list-style-type: none">us-east-1a subnet-0c05909e5a18fac9b project-subnet-public1-us-east-1aus-east-1b subnet-0c616971c0b098f0b project-subnet-public2-us-east-1b	<ul style="list-style-type: none">HTTP:80 defaults to <i>Target group not defined</i>

aws Services Search [Alt+S]

IAM EC2 RDS VPC

⌚ Successfully created load balancer: project-lb

Note: It might take a few minutes for your load balancer to be fully set up and ready to route traffic. Targets will also take a few minutes to complete the registration process and pass initial health checks.

EC2 > Load balancers > project-lb > Create Application Load Balancer

Create Application Load Balancer

Suggested next steps

- Review, customize, or configure attributes for your load balancer and listeners using the **Description** and **Listeners** tabs within [project-lb](#).
- Discover other services that you can integrate with your load balancer. Visit the **Integrated services** tab within [project-lb](#).

[View load balancer](#)

Created Auto scaling group:

Amazon EC2 Auto Scaling helps maintain the availability of your applications

Auto Scaling groups are collections of Amazon EC2 instances that enable automatic scaling and fleet management features. These features help you maintain the health and availability of your applications.

Create Auto Scaling group

Get started with EC2 Auto Scaling by creating an Auto Scaling group.

Create Auto Scaling group

How it works

Pricing

Amazon EC2 Auto Scaling features have no additional fees beyond the service fees for Amazon EC2, CloudWatch (for scaling policies), and the other AWS resources that you use. Visit the pricing page of each service to learn more.

Getting started

Step 3 - optional
Configure advanced options

Step 4 - optional
Configure group size and scaling policies

Step 5 - optional
Add notifications

Step 6 - optional
Add tags

Step 7
Review

Auto Scaling group name
Enter a name to identify the group.
Project-ASG
Must be unique to this account in the current Region and no more than 255 characters.

Launch template **Info** **Switch to launch configuration**

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

Create a launch template **C**

Version
Default (1) **C**

Create a launch template version **C**

Description A webserver	Launch template webserver-template C lt-0cc3488610265ff6e	Instance type t2.micro
AMI ID ami-04e2c62be8bd77e46	Security groups -	Request Spot Instances No
Key pair name project-key	Security group IDs sg-0950a2bc8e490890d C sg-08882068c03a8469f C	

Additional details

Storage (volumes) /dev/xvda	Date created Sat Jul 22 2023 08:47:38 GMT+0100 (Irish Standard Time)
--------------------------------	--

Step 2
Choose instance launch options

Step 3 - optional
Configure advanced options

Step 4 - optional
Configure group size and scaling policies

Step 5 - optional
Add notifications

Step 6 - optional
Add tags

Step 7
Review

options.

Network **Info**

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

VPC
Choose the VPC that defines the virtual network for your Auto Scaling group.
vpc-0765d503dd08f4e60 (project-vpc) **C**
10.0.0.0/16

Create a VPC **C**

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

Select Availability Zones and subnets **C**

us-east-1a | subnet-0c05909e5a18fac9b (project-subnet-public1-us-east-1a) **X**
10.0.0.0/20

us-east-1b | subnet-0c616971c0b098f0b (project-subnet-public2-us-east-1b) **X**
10.0.16.0/20

Create a subnet **C**

Instance type requirements **Info** **Override launch template**

You can keep the same instance attributes or instance type from your launch template, or you can choose to override the launch template by specifying different instance attributes or manually adding instance types.

Launch template webserver-template C lt-0cc3488610265ff6e	Version Default	Description A webserver
---	--------------------	----------------------------

Choose instance launch options

Step 3 - optional
Configure advanced options

Step 4 - optional
Configure group size and scaling policies

Step 5 - optional
Add notifications

Step 6 - optional
Add tags

Step 7
Review

Load balancing 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

Existing load balancer target groups

Only instance target groups that belong to the same VPC as your Auto Scaling group are available for selection.

Select target groups ▾ ✖

Project-lb-TG | HTTP ✖
Application Load Balancer: project-lb

VPC Lattice integration options Info

To improve networking capabilities and scalability, integrate your Auto Scaling group with VPC Lattice. VPC Lattice facilitates communications between AWS services and helps you connect and manage your applications across compute services in AWS.

Select VPC Lattice service to attach

No VPC Lattice service
VPC Lattice will not manage your Auto Scaling group's network access and connectivity with other services.

Attach to VPC Lattice service
Incoming requests associated with specified VPC Lattice target groups will be routed to your Auto Scaling group.

Step 1
Choose launch template or configuration

Step 2
Choose instance launch options

Step 3 - optional
Configure advanced options

Step 4 - optional
Configure group size and scaling policies

Step 5 - optional
Add notifications

Step 6 - optional
Add tags

Configure group size and scaling policies - optional Info

Set the desired, minimum, and maximum capacity of your Auto Scaling group. You can optionally add a scaling policy to dynamically scale the number of instances in the group.

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
5

AWS Services Search [Alt+S]

Step 6 - optional Add tags

Maximum capacity 5

Step 7 Review Scaling policies - *optional*

Choose whether to use a scaling policy to dynamically resize your Auto Scaling group to meet changes in demand. [Info](#)

Target tracking scaling policy Choose a desired outcome and leave it to the scaling policy to add and remove capacity as needed to achieve that outcome.

None

Scaling policy name Target Tracking Policy

Metric type Average CPU utilization

Target value 50

Instances need 50 seconds warm up before including in metric

Disable scale in to create only a scale-out policy

Successfully created auto scaling group

aws Services Search [Alt+F]

Project-ASG 1 Scaling policy created successfully

EC2 > Auto Scaling groups

Auto Scaling groups (1) Info

Search your Auto Scaling groups

Name Launch template/configuration Instances Status Desired capacity Min Max Availability Zones

Project-ASG webserver-template | Version Default 0 0 Updating capacity... 2 2 5 us-east-1a, us-east-1b

Desired 2 webserver EC2 instances created and running

New EC2 Experience X Instances (3) Info

Find instance by attribute or tag (case-sensitive)

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS	Public IPv4
-	i-0a94400619bf14558	Running	t2.micro	2/2 checks passed	No alarms	+ us-east-1b	ec2-34-201-69-198.co...	34.201.69.198
webserver	i-055987cae3f07fb3	Running	t2.micro	2/2 checks passed	No alarms	+ us-east-1a	ec2-52-200-101-155.co...	52.200.101.155
-	i-025916e71e0e6282	Running	t2.micro	2/2 checks passed	No alarms	+ us-east-1a	ec2-52-87-192-95.com...	52.87.192.95

Check the LB DNS name and open the URL from the web browser.

The screenshot shows the AWS CloudWatch Metrics interface. On the left, there's a navigation pane with links like 'Metrics', 'Logs', 'CloudWatch Metrics Insights', 'CloudWatch Metrics Metrics Insights', and 'CloudWatch Metrics Metrics Insights'. The main area displays a chart titled 'HelloWorld' with a single data series. The chart has three data points: one at 100% at time 0, another at 100% at time 10, and a third at 100% at time 20. Below the chart, there's a table with columns 'Time' and 'Value'.

Successfully able to connect webserver page

The screenshot shows a web browser window with the URL project-lb-1650961892.us-east-1.elb.amazonaws.com/SamplePage.php. The page title is "Employees Data". It features a table with columns "NAME" and "ADDRESS". There is a "Add Data" button and a "Listeners and rules" section below the table.

ID	NAME	ADDRESS
1	Nageswara Rao Goparaju	Hyderabad
2	Harsha Vardhan Goparaju	Bangalore

Update more data to check the webserver and database are working as expected

The screenshot shows a web browser window with the same URL as the previous screenshot. The page title is "Employees Data". The table now contains four rows of data: Nageswara Rao Goparaju (Hyderabad), Harsha Vardhan Goparaju (Bangalore), Krishna (Delhi), and Shankar (Chennai). There is a "Add Data" button and a "Listeners and rules" section below the table.

ID	NAME	ADDRESS
1	Nageswara Rao Goparaju	Hyderabad
2	Harsha Vardhan Goparaju	Bangalore
3	Krishna	Delhi
4	Shankar	Chennai

Checking the High Availability using the termination of the running webserver

Terminated one webserver and check the application availability from the second instance

The screenshot shows the AWS EC2 Instances page. A green banner at the top indicates "Successfully terminated i-044612c36b74c800c". The table lists five instances:

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS	Public IPv4
Webserver	i-07363d4056dfef5dff	Terminated	t2.micro	-	No alarms	+ us-east-1b	-	-
webserver	i-0ca580db63120e0ec	Running	t2.micro	2/2 checks passed	No alarms	+ us-east-1b	-	-
webserver	i-044612c36b74c800c	Shutting-down	t2.micro	2/2 checks passed	No alarms	+ us-east-1a	-	-
Webserver	i-0fd4925cf924001cd	Terminated	t2.micro	-	No alarms	+ us-east-1a	-	-
Webserver	i-0b75b1f1417d77616	Stopped	t2.micro	2/2 checks passed	No alarms	+ us-east-1a	-	35.175.144

Still the application up and running

The screenshot shows a web browser displaying the URL project-lb-1650961892.us-east-1.elb.amazonaws.com/SamplePage.php. The page title is "Employees Data". It contains a table with four rows of employee data:

ID	NAME	ADDRESS
1	Nageswara Rao Goparaju	Hyderabad
2	Harsha Vardhan Goparaju	Bangalore
3	Krishna	Delhi
4	Shankar	Chennai

There is also a "Add Data" button.

Check the HA working to make it desired state 2

Within few mints second instances initiated and started running

The screenshot shows the AWS EC2 Instances page. A green banner at the top indicates "Successfully terminated i-025916e71ce0c6282". The table lists four instances:

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS	Public IPv4
-	i-094400619f14558	Running	t2.micro	2/2 checks passed	No alarms	+ us-east-1b	ec2-54-201-69-198.co...	34.201.69.198
webserver	i-0c57987cae3f07f83	Running	t2.micro	2/2 checks passed	No alarms	+ us-east-1a	ec2-52-200-101-155.co...	52.200.101.155
-	i-0209c917bf0f00edb	Pending	t2.micro	-	No alarms	+ us-east-1a	ec2-54-211-233-45.co...	54.211.233.45
-	i-025916e71ce0c6282	Terminated	t2.micro	-	No alarms	+ us-east-1a	-	-

Check the Stress testing to make it increase webserver instances

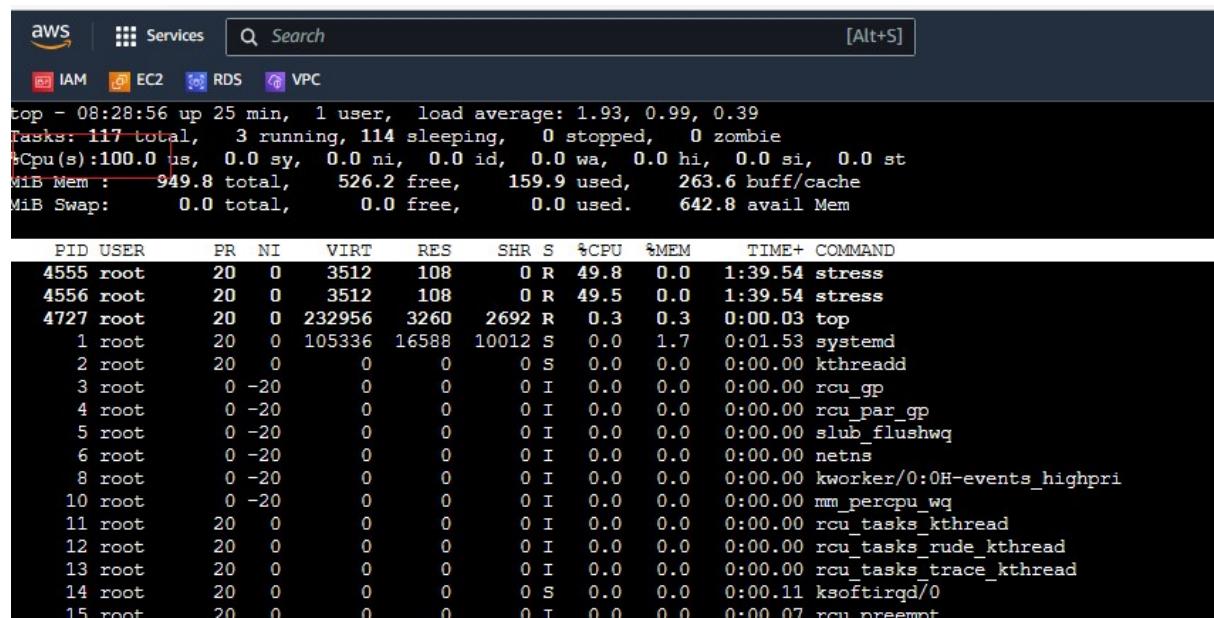
Install stress on EC2 instances and run stress command

```
yum install stress -y
```

```
[root@ip-10-0-31-125 ~]# yum install stress -y
last metadata expiration check: 1:02:43 ago on Sat Jul 22 07:20:33 2023.
Dependencies resolved.
=====
Package                           Architecture      Version
=====
Installing:
stress                            x86_64          1.0.4-28.amzn2023.0.2
Transaction Summary
Install 1 Package
Total download size: 37 k
Installed size: 78 k
Downloading Packages:
stress-1.0.4-28.amzn2023.0.2.x86_64.rpm
=====
Total
Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
Running transaction
  Preparing           :
  Installing         : stress-1.0.4-28.amzn2023.0.2.x86_64
  Running scriptlet: stress-1.0.4-28.amzn2023.0.2.x86_64
  Verifying          : stress-1.0.4-28.amzn2023.0.2.x86_64
Installed:
  stress-1.0.4-28.amzn2023.0.2.x86_64
Complete!
```

Stress applied on the CPU

```
Complete!
[root@ip-10-0-0-17 ~]# stress --cpu 2 --timeout 1000 &
[1] 3252
[root@ip-10-0-0-17 ~]# stress: info: [3252] dispatching hogs: 2 cpu, 0 io, 0 vm, 0 hdd
```



The screenshot shows the AWS CloudWatch Metrics Insights interface. At the top, there are navigation links for IAM, EC2, RDS, and VPC. Below that is a search bar and a key combination [Alt+S]. The main area displays a terminal session with the following content:

```
top - 08:28:56 up 25 min,  1 user,  load average: 1.93, 0.99, 0.39
Tasks: 117 total,   3 running, 114 sleeping,   0 stopped,   0 zombie
%Cpu(s): 100.0 us,  0.0 sy,  0.0 ni,  0.0 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
MiB Mem :  949.8 total,    526.2 free,   159.9 used,   263.6 buff/cache
MiB Swap:     0.0 total,      0.0 free,      0.0 used.   642.8 avail Mem

PID USER      PR  NI    VIRT    RES    SHR S %CPU %MEM TIME+ COMMAND
4555 root      20   0   3512   108    0 R 49.8  0.0  1:39.54 stress
4556 root      20   0   3512   108    0 R 49.5  0.0  1:39.54 stress
4727 root      20   0  232956  3260  2692 R  0.3  0.3  0:00.03 top
  1 root      20   0  105336 16588 10012 S  0.0  1.7  0:01.53 systemd
  2 root      20   0     0     0    0 S  0.0  0.0  0:00.00 kthreadd
  3 root      0 -20    0     0    0 I  0.0  0.0  0:00.00 rcu_gp
  4 root      0 -20    0     0    0 I  0.0  0.0  0:00.00 rcu_par_gp
  5 root      0 -20    0     0    0 I  0.0  0.0  0:00.00 slub_flushwq
  6 root      0 -20    0     0    0 I  0.0  0.0  0:00.00 netns
  8 root      0 -20    0     0    0 I  0.0  0.0  0:00.00 kworker/0:0H-events_highpri
 10 root      0 -20    0     0    0 I  0.0  0.0  0:00.00 mm_percpu_wq
 11 root      20   0     0     0    0 I  0.0  0.0  0:00.00 rcu_tasks_kthread
 12 root      20   0     0     0    0 I  0.0  0.0  0:00.00 rcu_tasks_rude_kthread
 13 root      20   0     0     0    0 I  0.0  0.0  0:00.00 rcu_tasks_trace_kthread
 14 root      20   0     0     0    0 S  0.0  0.0  0:00.11 ksoftirqd/0
 15 root      20   0     0     0    0 T  0.0  0.0  0:00.07 rcu_prempt
```

Started creating new Instances after its reaches 50% threshold

AWS Services Search [Alt+S]

New EC2 Experience Tell us what you think

EC2 Dashboard EC2 Global View Events

Instances Instances Instance Types Launch Templates Spot Requests Savings Plans Reserved Instances Dedicated Hosts Scheduled Instances

Instances (6) Info

Name	Instance ID	Instance state	Instance type	Status check	Alarm status
-	i-Off5781069246ae35	Pending	t2.micro	-	No alarms
-	i-0a94400619bf14558	Running	t2.micro	2/2 checks passed	No alarms
-	i-Ofcd73aea2c396f53	Pending	t2.micro	-	No alarms
webserver	i-0c57987cae3f07f83	Running	t2.micro	2/2 checks passed	No alarms
-	i-020d9c17b9f0a0edb	Running	t2.micro	2/2 checks passed	No alarms
-	i-025916e71ce0c6282	Terminated	t2.micro	-	No alarms

Reached max 5 instances count

AWS Services Search [Alt+S]

New EC2 Experience Tell us what you think

N. Virginia ec2_user_1023149 @ 6677-78

EC2 Dashboard EC2 Global View Events

Instances Instances Instance Types Launch Templates Spot Requests Savings Plans Reserved Instances Dedicated Hosts Scheduled Instances

Instances (7) Info

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS	Public IPv4 ...
-	i-Ofcf79561327b392de	Running	t2.micro	Initializing	No alarms	+ us-east-1b	ec2-54-242-233-107.co...	54.242.233.107
-	i-Off5781069246ae35	Running	t2.micro	Initializing	No alarms	+ us-east-1b	ec2-54-237-234-103.co...	54.237.234.103
-	i-0a94400619bf14558	Running	t2.micro	2/2 checks passed	No alarms	+ us-east-1b	ec2-34-201-69-198.co...	34.201.69.198
-	i-Ofcd73aea2c396f53	Running	t2.micro	Initializing	No alarms	+ us-east-1a	ec2-44-212-24-186.co...	44.212.24.186
webserver	i-0c57987cae3f07f83	Running	t2.micro	2/2 checks passed	No alarms	+ us-east-1a	ec2-52-209-101-155.co...	52.209.101.155
-	i-020d9c17b9f0a0edb	Running	t2.micro	2/2 checks passed	No alarms	+ us-east-1a	ec2-54-211-233-45.co...	54.211.233.45
-	i-025916e71ce0c6282	Terminated	t2.micro	-	No alarms	+ us-east-1a	-	-

Able to insert data

← ⌂ Not secure | project-lb-1650961892.us-east-1.elb.amazonaws.com/SamplePage.php

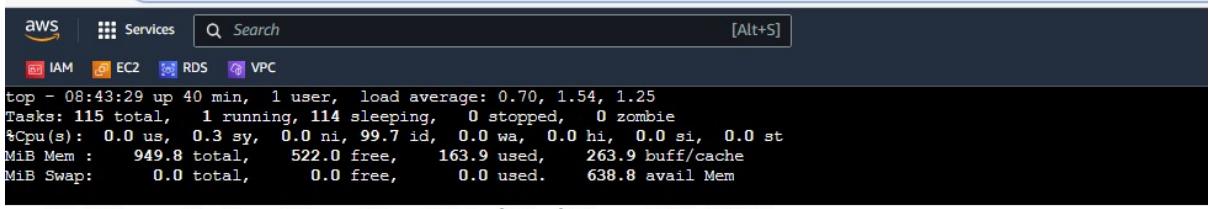
Employees Data

NAME	ADDRESS
<input type="text"/>	<input type="text"/>

Add Data

ID	NAME	ADDRESS
1	Nageswara Rao Goparaju	Hyderabad
2	Harsha Vardhan Goparaju	Bangalore
3	Krishna	Delhi
4	Shankar	Chennai
5	Ramarao	Hyderabad

Now the CPU load reduced to below 50%



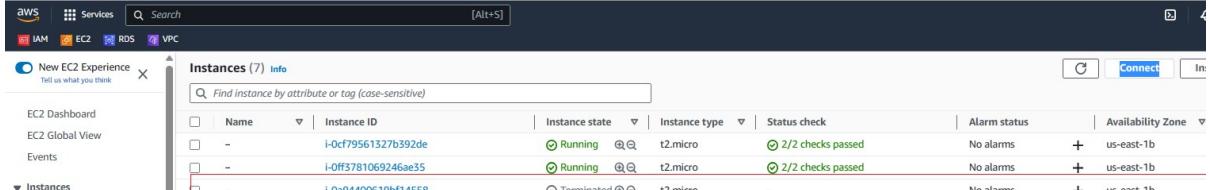
```

top - 08:43:29 up 40 min, 1 user, load average: 0.70, 1.54, 1.25
Tasks: 115 total, 1 running, 114 sleeping, 0 stopped, 0 zombie
%Cpu(s): 0.0 us, 0.3 sy, 0.0 ni, 99.7 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st
MiB Mem : 949.8 total, 522.0 free, 163.9 used, 263.9 buff/cache
MiB Swap: 0.0 total, 0.0 free, 0.0 used. 638.8 avail Mem

PID USER PR NI VIRT RES SHR S %CPU %MEM TIME+ COMMAND
3352 apache 20 0 1275964 8956 5496 S 0.3 0.9 0:00.24 httpd
5771 root 20 0 232956 3260 2692 R 0.3 0.3 0:00.02 top
1 root 20 0 105336 16588 10012 S 0.0 1.7 0:01.67 systemd
2 root 20 0 0 0 0 S 0.0 0.0 0:00.00 kthreadd
3 root 0 -20 0 0 0 I 0.0 0.0 0:00.00 rcu_gp
4 root 0 -20 0 0 0 I 0.0 0.0 0:00.00 rcu_par_gp
8 root 0 -20 0 0 0 I 0.0 0.0 0:00.00 kworker/0:0H-events_highpri
10 root 0 -20 0 0 0 I 0.0 0.0 0:00.00 mm_percpu_wq
11 root 20 0 0 0 0 I 0.0 0.0 0:00.00 rcu_tasks_kthread
12 root 20 0 0 0 0 I 0.0 0.0 0:00.00 rcu_tasks_rude_kthread
13 root 20 0 0 0 0 I 0.0 0.0 0:00.00 rcu_tasks_trace_kthread
14 root 20 0 0 0 0 S 0.0 0.0 0:00.16 ksoftirqd/0
15 root 20 0 0 0 0 I 0.0 0.0 0:00.09 rcu_preempt
16 root rt 0 0 0 0 S 0.0 0.0 0:00.01 migration/0
18 root 20 0 0 0 0 S 0.0 0.0 0:00.00 cpuhp/0
20 root 20 0 0 0 0 S 0.0 0.0 0:00.00 kdevtmpfs
21 root 0 0 0 0 0 S 0.0 0.0 0:00.00 ksoftirqd/1

```

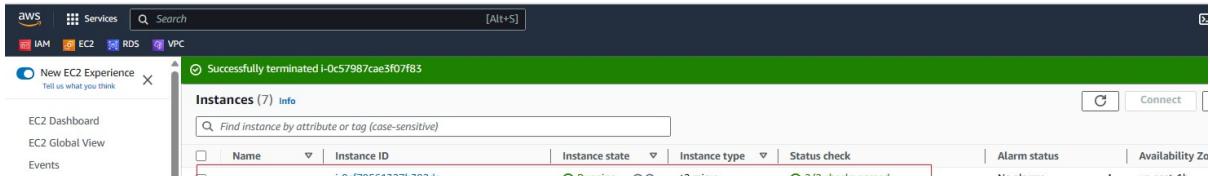
Started Scale in from Auto scaling



Instances (7) Info

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone
-	i-0cf79561327b392de	Running	t2.micro	2/2 checks passed	No alarms	us-east-1b
-	i-0ff3781069246ae35	Running	t2.micro	2/2 checks passed	No alarms	us-east-1b
-	i-0a94400619bf14558	Terminated	t2.micro	-	No alarms	us-east-1b
-	i-0cf73aea2c396f53	Running	t2.micro	2/2 checks passed	No alarms	us-east-1a
webserver	i-0c57987cae3f07f83	Running	t2.micro	2/2 checks passed	No alarms	us-east-1a
-	i-020d9c17b9f0a0edb	Running	t2.micro	2/2 checks passed	No alarms	us-east-1a
-	i-025916e71ce0c6282	Terminated	t2.micro	-	No alarms	us-east-1a

Now only running with desired instances



Successfully terminated i-0c57987cae3f07f83

Instances (7) Info

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone
-	i-0cf79561327b392de	Running	t2.micro	2/2 checks passed	No alarms	us-east-1b
-	i-0ff3781069246ae35	Terminated	t2.micro	-	No alarms	us-east-1b
-	i-0a94400619bf14558	Terminated	t2.micro	-	No alarms	us-east-1b
-	i-0cf73aea2c396f53	Running	t2.micro	2/2 checks passed	No alarms	us-east-1a
webserver	i-0c57987cae3f07f83	Terminated	t2.micro	-	No alarms	us-east-1a
-	i-020d9c17b9f0a0edb	Terminated	t2.micro	-	No alarms	us-east-1a
-	i-025916e71ce0c6282	Terminated	t2.micro	-	No alarms	us-east-1a

Result

Finally the webserver up and running with High availability using Auto scaling group as expected.

The screenshot shows a web browser window with the URL project-lb-1650961892.us-east-1.elb.amazonaws.com/SamplePage.php. The page title is "Employees Data". There are two input fields labeled "NAME" and "ADDRESS". Below them is a table with 5 rows of data:

ID	NAME	ADDRESS
1	Nageswara Rao Goparaju	Hyderabad
2	Harsha Vardhan Goparaju	Bangalore
3	Krishna	Delhi
4	Shankar	Chennai
5	Ramarao	Hyderabad

An "Add Data" button is located to the right of the table.

END of the Project