

# Infrastructure Provisioning using Ansible

## Problem:

provisioning cloud resources using Ansible to set up a basic web application infrastructure.

infrastructure consist of the following components:

1. An AWS EC2 instance running a basic web server.
2. An AWS RDS MySQL database for storing application data.

## Requirements:

Create an EC2 instance with the following specifications:

1. Instance Type: t2.micro
2. AMI: Ubuntu Server 20.04 LTS
3. Security Group: Allow incoming HTTP (port 80) and SSH (port 22) traffic.

Create an RDS MySQL instance with the following specifications:

1. Instance Class: db.t2.micro
2. Engine: MySQL
3. Allow incoming traffic on port 3306 from the EC2 instance's security group.

## ANISBLE PLAYBOOK FOR CREATING EC2 AND RDS INFRASTRUCTURE

---

**- name: Provision AWS Infrastructure**

**hosts: localhost**

**gather\_facts: no**

**vars:**

**aws\_access\_key: "AKIA6ODU7LJYX2CVOHHA"**

**aws\_secret\_key: "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"**

**aws\_region: "us-west-2"**

**ec2\_instance\_type: "t2.micro"**

**ec2\_ami: "ami-0c66c6068e0b884e1"**

**ec2\_security\_group: "web-server-sg"**  
**rds\_instance\_class: "db.t3.micro"**  
**rds\_engine: "mysql"**  
**rds\_engine\_version: "5.7.44"**  
**rds\_db\_name: "webapp\_db"**  
**rds\_master\_username: "admin"**  
**rds\_master\_password: "password"**  
**rds\_security\_group: "rds-sg"**  
**key\_pair\_name: "KEY1"**

**tasks:**

- name: Create security group for EC2

**amazon.aws.ec2\_security\_group:**  
**name: web\_sg**  
**description: Security group for web server**  
**region: "{{ aws\_region }}"**  
**rules:**

- proto: tcp

**ports:**

- 22
- 80

**cidr\_ip: 0.0.0.0/0**  
**state: present**  
**register: ec2\_sg**

- name: Create security group for RDS

**amazon.aws.ec2\_security\_group:**  
**name: rds\_sg**  
**description: Security group for RDS**  
**region: "{{ aws\_region }}"**  
**rules:**

- proto: tcp  
ports:  
- 3306  
group\_id: "{{ ec2\_sg.group\_id }}"  
state: present  
register: rds\_sg

- name: Launch EC2 instance  
amazon.aws.ec2\_instance:  
key\_name: "{{ key\_pair\_name }}"  
instance\_type: "{{ ec2\_instance\_type }}"  
image\_id: "{{ ec2\_ami }}"  
wait: yes  
security\_groups: "{{ ec2\_sg.group\_id }}"  
region: "{{ aws\_region }}"  
count: 1  
tags:  
Name: web\_server  
wait: yes  
user\_data: |  
#!/bin/bash  
sudo apt update  
sudo apt install -y apache2  
sudo systemctl start apache2  
sudo systemctl enable apache2  
register: ec2\_instance

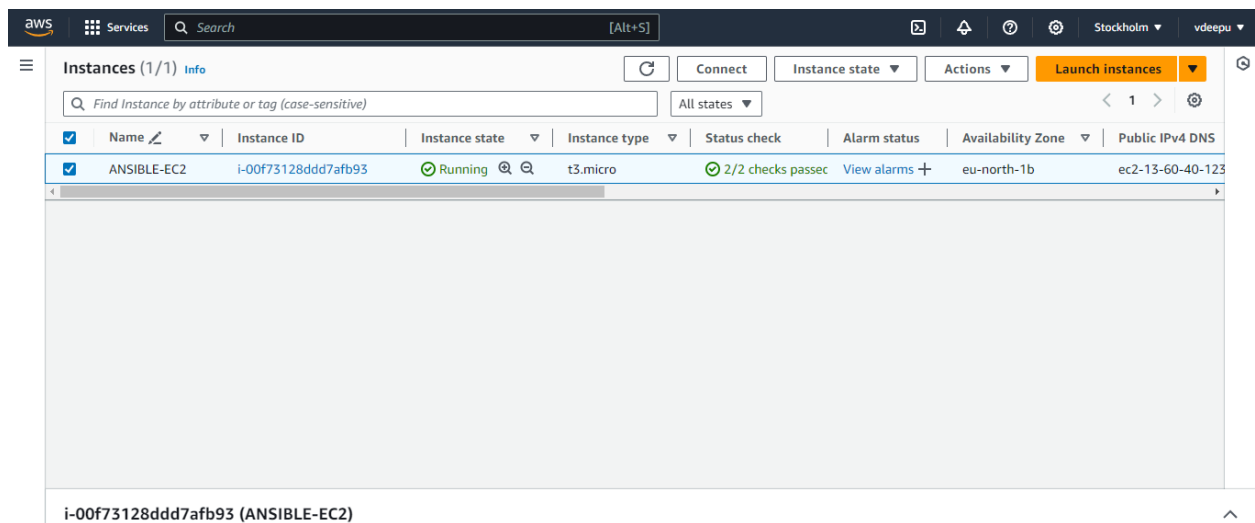
- name: Create RDS instance  
amazon.aws.rds\_instance:  
db\_instance\_identifier: mydbinstance

```

engine: "{{ rds_engine }}"
db_instance_class: "{{ rds_instance_class }}"
master_username: "{{ rds_master_username }}"
master_user_password: "{{ rds_master_password }}"
allocated_storage: 20
vpc_security_group_ids: "{{ rds_sg.group_id }}"
db_name: "{{ rds_db_name }}"
region: "{{ aws_region }}"
publicly_accessible: yes
wait: yes
aws_access_key: "{{ aws_access_key }}"
aws_secret_key: "{{ aws_secret_key }}"

```

## 1. Creating a control node EC2 instance for installing Ansible and provisioning the infrastructure.



## 2. In that control node installing ansible and verifying its version.

```
ubuntu@ip-172-31-32-160: ~  
ubuntu@ip-172-31-32-160:~$ ansible --version  
ansible [core 2.16.8]  
  config file = /etc/ansible/ansible.cfg  
  configured module search path = ['/home/ubuntu/.ansible/plugins/modules', '/usr/share/ansible/plugins/modules']  
  ansible python module location = /usr/lib/python3/dist-packages/ansible  
  ansible collection location = /home/ubuntu/.ansible/collections:/usr/share/ansible/collections  
  executable location = /usr/bin/ansible  
  python version = 3.12.3 (main, Apr 10 2024, 05:33:47) [GCC 13.2.0] (/usr/bin/python3)  
  jinja version = 3.1.2  
  libyaml = True  
ubuntu@ip-172-31-32-160:~$
```

## 3. Installing aws cli in control node that allows us to manage AWS resources directly.

```
root@ip-172-31-36-41: ~  
root@ip-172-31-36-41:~# aws --version  
aws-cli/2.17.13 Python/3.11.9 Linux/6.8.0-1009-aws exe/x86_64.ubuntu.24  
root@ip-172-31-36-41:~#
```

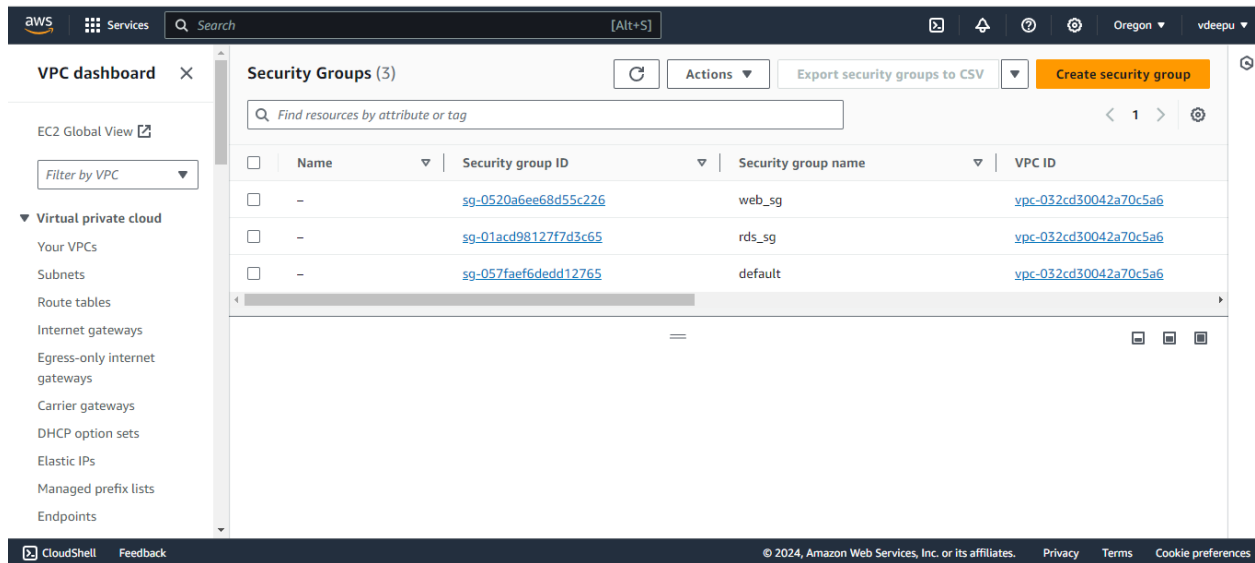
## 4. Configuring aws configure in control node.

```
root@ip-172-31-36-41: ~  
root@ip-172-31-36-41:~# aws configure  
AWS Access Key ID [None]: AKIA6ODU7LJYX2CVOHHA  
AWS Secret Access Key [None]: 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  
Default region name [None]: eu-north-1  
Default output format [None]: json  
root@ip-172-31-36-41:~#
```

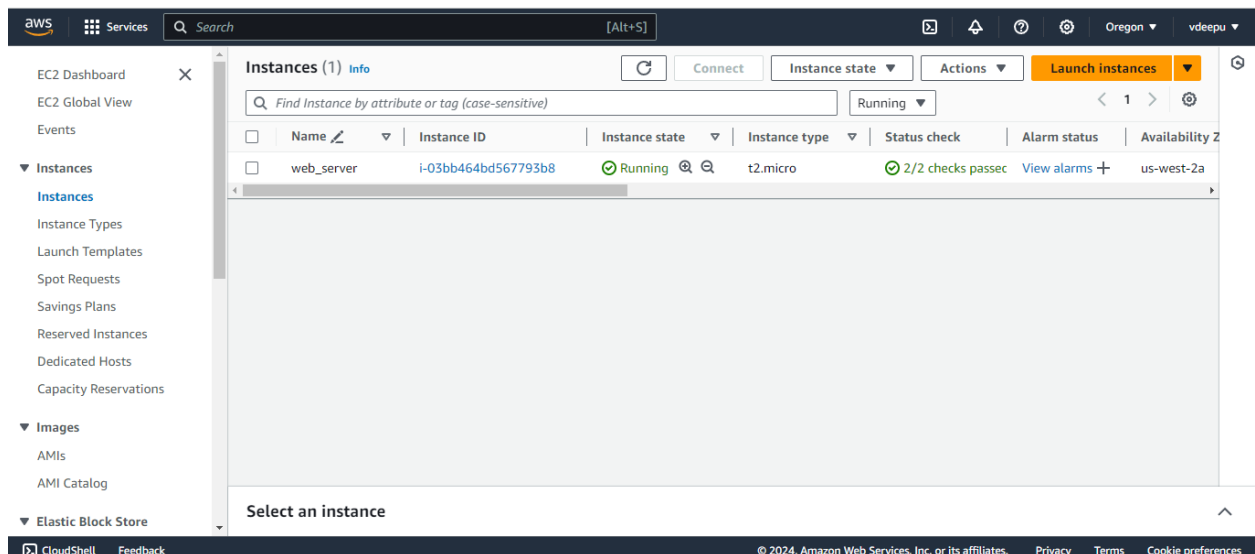
## 5. Running a Ansible playbook for creating a ec2 and rds infrastructure.

```
root@ip-172-31-36-41: ~/ansible  
root@ip-172-31-36-41:~/ansible# ls  
group_vars  myenv  playbook.yml  
root@ip-172-31-36-41:~/ansible# vi playbook.yml  
root@ip-172-31-36-41:~/ansible# ansible-playbook playbook.yml  
[WARNING]: provided hosts list is empty, only localhost is available. Note that the implicit localhost does not match 'all'  
  
PLAY [Provision AWS Infrastructure] *****  
  
TASK [Create security group for EC2] *****  
ok: [localhost]  
  
TASK [Create security group for RDS] *****  
ok: [localhost]  
  
TASK [Launch EC2 instance] *****  
changed: [localhost]  
  
TASK [Create RDS instance] *****  
ok: [localhost]  
  
PLAY RECAP *****  
localhost : ok=4  changed=1  unreachable=0  failed=0  skipped=0  rescued=0  ignored=0  
  
root@ip-172-31-36-41:~/ansible#
```

**6. Security group is created for EC2 and RDS with specific ports opened for allowing incoming traffics from ports ssh(22) , http(80) and mysql( 3306) for allowing incoming traffics from ec2 instance security group.**



**7. EC2 Instance is created with a basic web server- Apache**





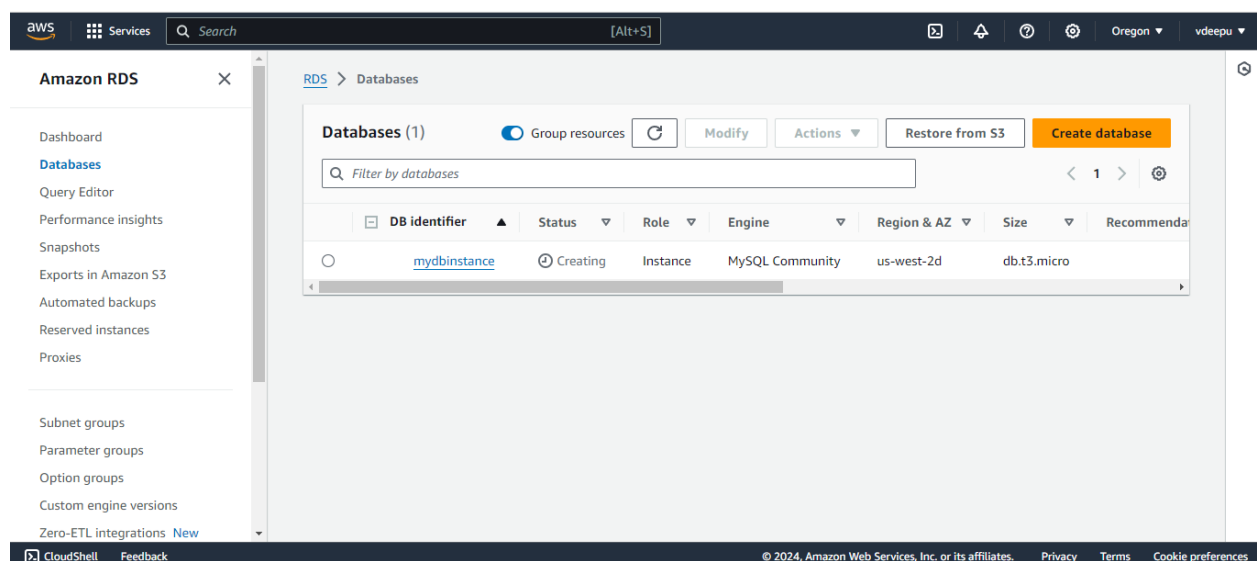




## Welcome to My Simple Web Page

This is a simple Apache web server.

**Note: For RDS Database the db.t2.micro instance type is no longer supported for new RDS MySQL database instance creations as of June 1, 2024, and support will end entirely on December 31, 2024. Instead, i used the db.t3.micro instance type, which offers better performance and cost efficiency.**



## 9. RDS Mysql Database is created with a instance type of db.t3.micro

