

**EX N0: 01**

**DATE: 03/07/2023**

## **CREATING AN EC2 INSTANCE AND DEPLOYING A SAMPLE WEB APPLICATION**

### **Aim:**

The aim of this exercise is to demonstrate the process of creating an EC2 instance on Amazon Web Services (AWS) and deploying a sample web application on the instance.

### **Procedure:**

**Step 1:** Log in to your AWS Management Console.



### **Sign in**

**Root user**

Account owner that performs tasks requiring unrestricted access. [Learn more](#)

**IAM user**

User within an account that performs daily tasks. [Learn more](#)

**Root user email address**

22pds017@loyolacollege.edu

**Next**

Email: 22pds017@loyolacollege.edu

Password      [Forgot password?](#)

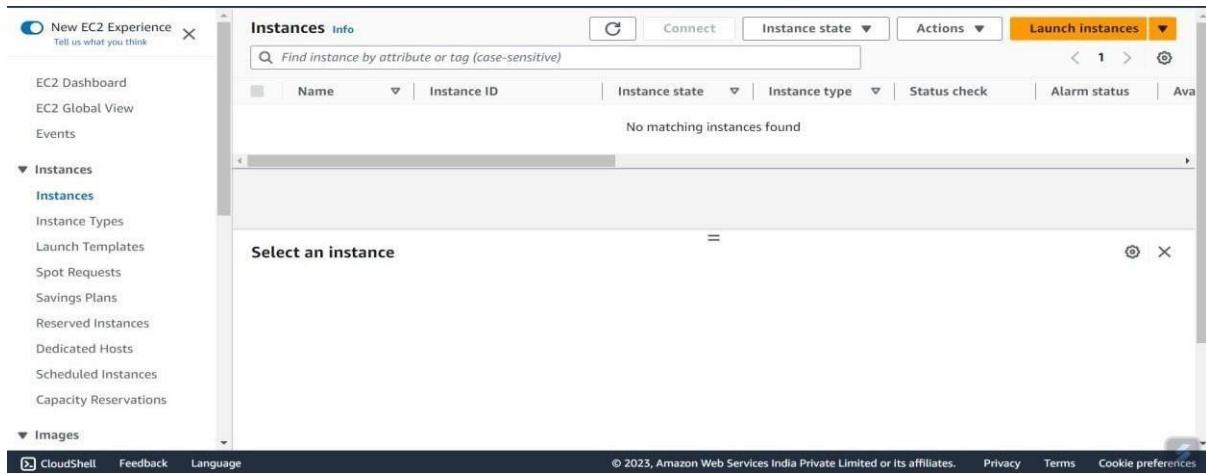
**Sign in**

The screenshot shows the AWS Console Home page. At the top, there's a search bar and navigation links for Services and N. Virginia. Below the search bar, there are three main sections: 'Recently visited' (with EC2, Billing, and Amazon SageMaker listed), 'Welcome to AWS' (with links for Getting started with AWS, Training and certification, and What's new with AWS), and 'AWS Health' (showing 0 open issues, 0 scheduled changes, and 0 other notifications). A blue button at the bottom right says 'Go to AWS Health'.

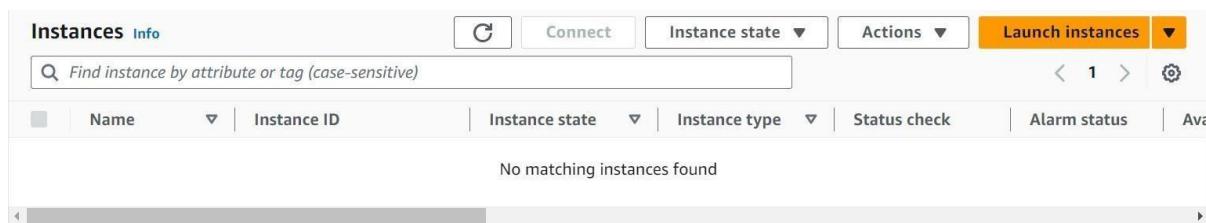
## Step 2: Navigate to the EC2 Dashboard.

The screenshot shows the AWS search results for 'ec2'. The search bar at the top has 'ec2' typed in. On the left, there's a sidebar with categories like Services (13), Features (52), Resources (New), Documentation (32,410), Knowledge Articles (20), Marketplace (2,463), Blogs (2,009), Events (30), and Tutorials (21). The main results section is titled 'Services' and shows four items: 'EC2' (Virtual Servers in the Cloud), 'EC2 Image Builder' (A managed service to automate build, customize and deploy OS images), 'Recycle Bin' (Protect resources from accidental deletion), and 'Amazon Inspector' (Continual vulnerability management at scale). There are also 'See all 13 results' and 'Features' sections below.

#EC2 Dashboard



**Step 3:** Click on "Launch Instance" to start the EC2 instance creation wizard.



- Choose an Amazon Machine Image (AMI) for your instance (e.g., Amazon Linux 2).

## ▼ 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.

Search our full catalog including 1000s of application and OS images

### Quick Start



Browse more AMIs  
Including AMIs from AWS, Marketplace and the Community

#### Amazon Machine Image (AMI)

Amazon Linux 2023 AMI  
ami-040d60c831d02d41c (64-bit (x86)) / ami-06c326063d3b494f1 (64-bit (Arm))  
Virtualization: hvm ENA enabled: true Root device type: ebs

Free tier eligible ▾

#### Description

Amazon Linux 2023 AMI 2023.1.20230725.0 x86\_64 HVM kernel-6.1

#### Architecture

64-bit (x86) ▾

#### AMI ID

ami-040d60c831d02d41c

Verified provider

- Select an instance type (e.g., t2.micro) based on your requirements.

## ▼ Instance type [Info](#)

#### Instance type

t3.micro  
Family: t3 2 vCPU 1 GiB Memory Current generation: true  
On-Demand RHEL pricing: 0.0708 USD per Hour  
On-Demand SUSE pricing: 0.0108 USD per Hour  
On-Demand Linux pricing: 0.0108 USD per Hour  
On-Demand Windows pricing: 0.02 USD per Hour

Free tier eligible

All generations

[Compare instance types](#)

- Create a new key pair or use an existing one to securely connect to the instance later.

## ▼ 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

Select ▾

Create new key pair

Give name and create key pair

### Create key pair

We noticed that you didn't select a key pair. If you want to be able to connect to your instance it is recommended that you create one.

Create new key pair     Proceed without key pair

**Key pair name**  
Key pairs allow you to connect to your instance securely.

The name can include up to 255 ASCII characters. It can't include leading or trailing spaces.

**Key pair type**

<input checked="" type="radio"/> RSA RSA encrypted private and public key pair	<input type="radio"/> ED25519 ED25519 encrypted private and public key pair
---	--

**Private key file format**

<input checked="" type="radio"/> .pem For use with OpenSSH	<input type="radio"/> .ppk For use with PuTTY
---	--

**Cancel** **Create key pair**

- Configure the instance details (e.g., network settings, security groups).

#### ▼ 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*

 Create new key pair

#### ▼ Network settings Info

**Edit**

Network Info

vpc-09f63052e479c3681

Subnet Info

No preference (Default subnet in any availability zone)

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

We'll create a new security group called 'launch-wizard-9' with the following rules:

Allow SSH traffic from

Helps you connect to your instance

0.0.0.0/0

Allow HTTPS traffic from the internet

To set up an endpoint, for example when creating a web server

Allow HTTP traffic from the internet

To set up an endpoint, for example when creating a web server

**⚠️** Rules with source of 0.0.0.0/0 allow all IP addresses to access your instance. We recommend setting security group rules to allow access from known IP addresses only. **X**

Add storage if needed.

Configure storage Info Advanced

1x 8 GiB gp3 Root volume (Not encrypted)

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

Add new volume

0 x File systems Edit

Review your settings and click on "Launch."

Summary

Number of instances Info

1

Software Image (AMI)

Amazon Linux 2023 AMI 2023.1.2...read more

ami-040d60c831d02d41c

Virtual server type (instance type)

t3.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 IOs, 1 GB of snapshots, and 100 GB of bandwidth to the internet. X

Cancel Launch instance Review commands

Success Successfully initiated launch of instance (i-03d3b6c3612abc6f)

**Step 5:** Connecting the Instance #Move to EC2 Dashboard, Select the newly created Instance and click Connect.



#Connect to the instance using EC2 Instance Connect

EC2 > Instances > i-0b9f3dc6aaf598f79 > Connect to instance

## Connect to instance Info

Connect to your instance i-0b9f3dc6aaf598f79 (Sample Web Application) using any of these options

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

Instance ID  
 i-0b9f3dc6aaf598f79 (Sample Web Application)

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

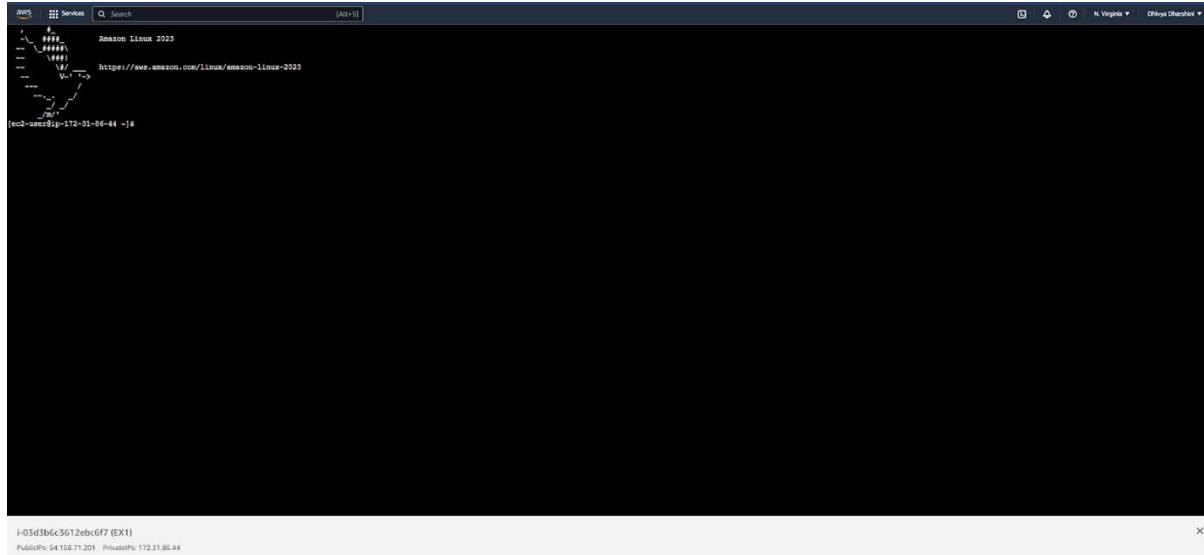
Public IP address  
 13.49.18.49

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

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

**Cancel** **Connect**

#After establishing connection



**Step 6:** Switch to the superuser (root) account in Linux systems

```
[ec2-user@ip-172-31-29-154 ~]$ sudo su -
```

**Step 7:** Update the package manager and install the Apache HTTP server.

```
[root@ip-172-31-29-154 ~]# yum update -y
```

```
[root@ip-172-31-29-154 ~]# yum install -y httpd
```

**Step 8:** Check the status of the Apache HTTP server

```
[root@ip-172-31-29-154 ~]# systemctl status httpd
```

```
httpd.service - The Apache HTTP Server
```

```
Loaded: loaded (/usr/lib/systemd/system/httpd.service; disabled; preset: disabled)
```

```
Active: inactive (dead)
```

```
Docs: man:httpd.service(8)
```

**Step 9:** Make a directory temp1 and change to that directory

```
[root@ip-172-31-29-154 ~]# mkdir temp1
```

```
[root@ip-172-31-29-154 ~]# cd temp1
```

**Step 10:** Download the template from free-css.com using wget command

```
[root@ip-172-31-93-109 temp1]# wget https://www.free-css.com/assets/files/free-csstemplates/download/page290/restoran.zip
```

**Step 11:** Unzip the file

```
[root@ip-172-31-93-109 temp1]# unzip restoran.zip
```

**Step 12:** List files and directories in the directory in long format.

```
[root@ip-172-31-29-154 temp1]# ls
```

```
-ltr total 1532
```

```
-rw-r--r--. 1 root root 1552247 Jan  7 2022 restoran.zip drwxr-xr-x. 7 root root 16384 Aug 10 17:34 bootstrap-restaurant-template
```

Step 13: Change directory to mobile-app-html-template

```
[root@ip-172-31-93-109 temp1]# cd bootstrap-restaurant-template
```

**Step 14:** Move all the files to the Apache web server's document root.

```
[root@ip-172-31-93-109 bootstrap-restaurant-template]# mv * /var/www/html
```

**Step 15:** Start the Apache web server

```
[root@ip-172-31-93-109 bootstrap-restaurant-template]# service httpd start
```

**Step 16:** Check the status of the Apache HTTP server

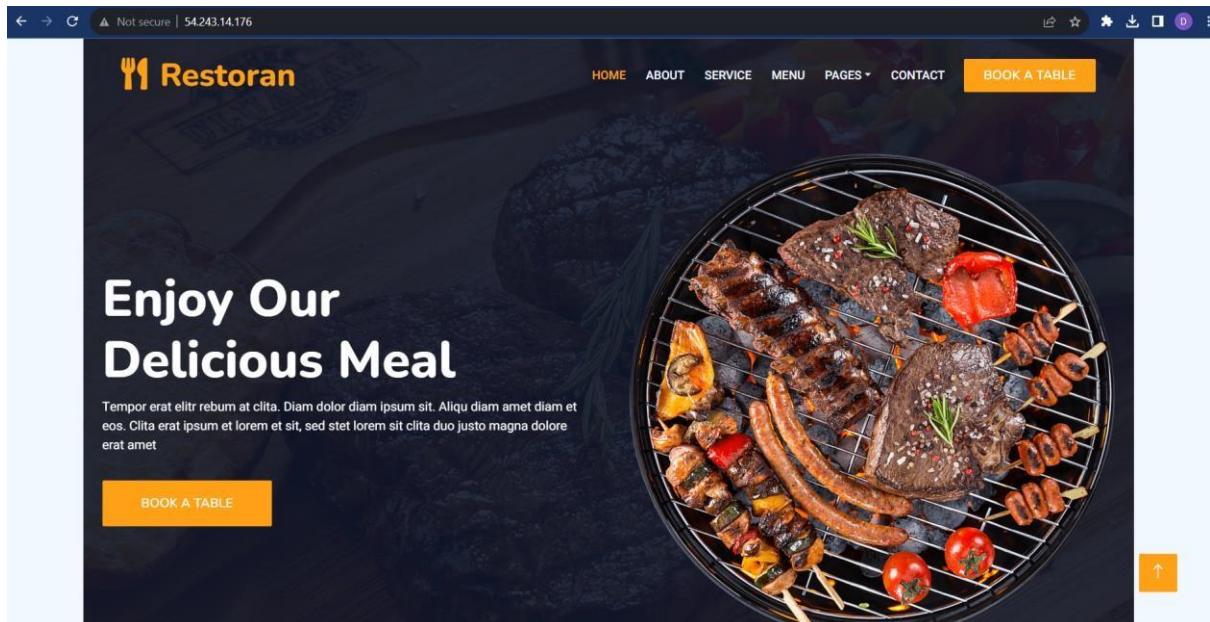
```
[root@ip-172-31-93-109 bootstrap-restaurant-template]# systemctl status httpd
```

httpd.service - The Apache HTTP Server

```
 Loaded: loaded (/usr/lib/systemd/system/httpd.service; disabled; preset: disabled)
```

```
 Active: active (running) since Sat 2023-08-05 18:02:26 UTC; 16s ago
          Docs: man:httpd.service(8)
```

## **Output:**



## **Result:**

The steps for creating an Amazon EC2 instance, installing the Apache HTTP Server, and deploying a sample web application (Restoran) have been successfully performed, resulting in a fully operational web server hosting the web application on the EC2 instance.

**EX N0: 02**

**DATE: 11/07/2023**

## **DEPLOYMENT OF A BASIC PYTHON WEB APPLICATION USING THE FLASK FRAMEWORK**

### **Aim:**

To demonstrate the deployment of a basic Python web application using the Flask framework on an Ubuntu EC2 instance.

### **Procedure:**

**Step 1:** Create an EC2 instance with Ubuntu in Amazon Machine Image allowing HTTP and HTTPS traffic.

**Step 2:** Update package information on the Ubuntu EC2 instance.

```
ubuntu@ip-172-31-46-36:~$ sudo apt update
```

**Step 3:** Install Python3 package manager (pip) on the instance.

```
ubuntu@ip-172-31-46-36:~$ sudo apt install python3-pip
```

**Step 4:** Check the version of pip to verify installation.

```
ubuntu@ip-172-31-46-36:~$ pip --version
```

*pip 22.0.2 from /usr/lib/python3/dist-packages/pip (python  
3.10)*

**Step 5:** Check the version of Git to verify installation

```
ubuntu@ip-172-31-46-36:~$ git --version
```

*git version 2.34.1*

**Step 6:** Clone the GitHub repository containing the Flask web application code.

```
ubuntu@ip-172-31-46-36:~$ git clone https://github.com/yeshwanthlm/docker-flaskdemo
```

**Step 7:** List files and directories in the directory in long format.

```
ubuntu@ip-172-31-46-36:~$ ls -ltr
```

*total 4 drwxrwxr-x 4 ubuntu ubuntu 4096 Jul 15 16:13  
docker-flask-demo*

**Step 8:** Navigate to the cloned directory.

```
ubuntu@ip-172-31-46-36:~$ cd docker-flask-demo
```

**Step 9:** List the contents of the directory to see the files.

```
ubuntu@ip-172-31-46-36:~/docker-flask-demo$ ls -ltr
```

*total 20  
drwxrwxr-x 2 ubuntu ubuntu 4096 Jul 15 16:13 templates  
-rw-rw-r-- 1 ubuntu ubuntu 5 Jul 15 16:13 requirements.txt  
-rw-rw-r-- 1 ubuntu ubuntu 2258 Jul 15 16:13 app.py  
-rw-rw-r-- 1 ubuntu ubuntu 687 Jul 15 16:13 Jenkinsfile  
-rw-rw-r-- 1 ubuntu ubuntu 112 Jul 15 16:13 Dockerfile*

**Step 10:** Install the Flask package using pip3.

```
ubuntu@ip-172-31-46-36:~/docker-flask-demo$ pip3 install flask
```

**Step 11:** Run the Flask web application.

```
ubuntu@ip-172-31-46-36:~/docker-flask-demo$ python3 app.py
```

*This is a sample web application that displays a colored background. A color can be specified in two ways.*

1. As a command line argument with --color as the argument. Accepts one of red,green,blue,blue2,pink,darkblue
2. As an Environment variable APP\_COLOR. Accepts one of red,green,blue,blue2,pink,darkblue
3. If none of the above then a random color is picked from the above list. Note: Command line argument precedes over environment variable.

*No command line argument or environment variable. Picking a Random Color =blue2*

*\* Serving Flask app 'app'*

*\* Debug mode: off*

*WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.*

*\* Running on all addresses (0.0.0.0)*

*\* Running on http://127.0.0.1:8080*

*\* Running on http://172.31.46.36:8080 Press CTRL+C to quit*

**Step 12:** Allow Custom TCP port 8080 in Inbound rules.

#Edit inbound rules

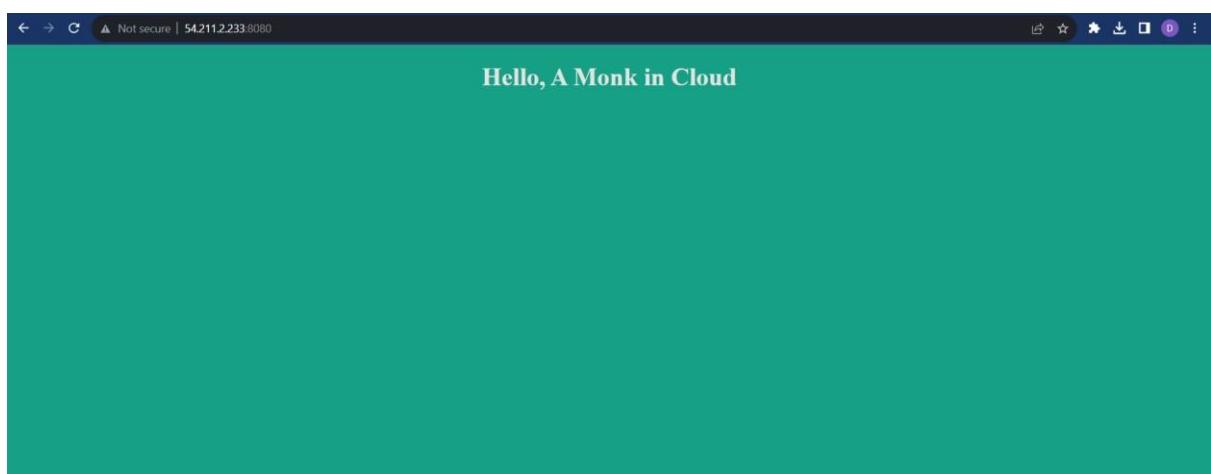
The screenshot shows the AWS Management Console interface for managing security groups. The URL in the address bar is `EC2 > Security Groups > sg-0133341ed1ee0848b - launch-wizard-10 > Edit inbound rules`. The page title is "Edit inbound rules".  
The main content area displays the "Inbound rules" table. It has two rows:

- Row 1: Security group rule ID is "sgr-00dc977eace3f20d8", Type is "SSH", Protocol is "TCP", Port range is "22", Source is "Custom" (with a dropdown menu showing "0.0.0.0/0"), and Description is empty. There is a "Delete" button to the right.
- Row 2: Security group rule ID is "sgr-07bbcf6d2a4556487", Type is "Custom TCP", Protocol is "TCP", Port range is "8080", Source is "Custom" (with a dropdown menu showing "0.0.0.0/0"), and Description is empty. There is a "Delete" button to the right.

At the bottom of the table, there is a "Add rule" button. At the very bottom of the page, there are three buttons: "Cancel", "Preview changes", and "Save rules".

**Step 13:** Now open <Public IP Address of EC2 Instance>:8080 in Browser

## **Output:**



## **Result:**

Thus, successfully deployed the basic Python web application using Flask on the Ubuntu EC2 instance, and the application was accessible through the EC2 instance's public IP address.

**EX N0: 03**

**DATE: 18/07/2023**

## **INSTALLING MYSQL IN UBUNTU INSTANCE AND PERFORMING CRUD OPERATIONS**

### **Aim:**

To demonstrate how to install MySQL on an Ubuntu EC2 instance and perform basic CRUD (Create, Read, Update, Delete) operations on a database using the MySQL command-line interface.

### **Procedure:**

**Step 1:** Create an EC2 instance with Ubuntu in Amazon

Machine Image **Step 2:** Update package information on the Ubuntu EC2 instance.

```
ubuntu@ip-172-31-45-247:~$ sudo apt update
```

**Step 3:** Install MySQL server on the instance.

```
ubuntu@ip-172-31-45-247:~$ sudo apt install mysql-server
```

**Step 4:** Check the status of the MySQL service to ensure it's running.

```
ubuntu@ip-172-31-45-247:~$ sudo systemctl status mysql
```

```
mysql.service - MySQL Community Server
   Loaded: loaded (/lib/systemd/system/mysql.service; enabled; vendor
   preset: enabled)
 Active: active (running) since Wed 2023-06-28 09:42:45 UTC; 8min ago
   Process: 2884 ExecStartPre=/usr/share/mysql/mysql-systemd-start pre
             (code=exited,
              status=0/SUCCESS) Main PID: 2892 (mysqld)
```

```
Status: "Server is operational"
  Tasks: 37 (limit: 1111)
Memory: 354.1M
CPU: 3.015s
CGroup: /system.slice/mysql.service
└─2892 /usr/sbin/mysqld

Jun 28 09:42:45 ip-172-31-45-247 systemd[1]: Starting MySQL
Community
Server...
Jun 28 09:42:45 ip-172-31-45-247 systemd[1]: Started MySQL Community
Server.
```

**Step 5:** Access the MySQL command-line interface (CLI) as the root user.

```
ubuntu@ip-172-31-45-247:~$ sudo mysql
```

```
Welcome to the MySQL monitor. Commands end with ; or \g.
Your MySQL connection id is 8
Server version: 8.0.33-0ubuntu0.22.04.2 (Ubuntu)
```

*Copyright (c) 2000, 2023, Oracle and/or its affiliates.*

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

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

Now you can run the SQL commands

**Step 6:** Create a new database named

'student'.

```
mysql> create database student;
```

*Query OK, 1 row affected (0.02 sec)*

**Step 7:** Show the list of databases to verify the creation of the 'student' database.

```
mysql> show databases;
```

```
+-----+  
| Database      |  
+-----+  
| information_schema |  
| mysql          |  
| performance_schema |  
| student         |  
| sys             |  
+-----+  
5 rows in set (0.20 sec)
```

**Step 8:** Switch to the 'student' database for further operations.

```
mysql> use student;
```

*Database changed*

**Step 9:** Create a table named 'name\_details' with two columns 'id' (integer) and 'name' (varchar).

```
mysql> create table name_details(id int,name varchar(45));
```

*Query OK, 0 rows affected (0.04 sec)*

**Step 10:** Insert three rows of data into the 'name\_details' table.

```
mysql> insert into name_details
```

```
values(1,'Dhivya'),(2,'Dharshini'),(3,'Anju');
```

*Query OK, 3 rows affected (0.01 sec)*

*Records: 3 Duplicates: 0 Warnings: 0*

**Step 11:** Retrieve and display all records from the 'name\_details' table.

```
mysql> select * from name_details;
```

```
+----+-----+  
| id | name |  
+----+-----+  
| 1 | Dhivya |  
| 2 | Dharshini |  
| 3 | Anju |  
+----+-----+  
3 rows in set (0.00 sec)
```

**Step 12:** Update a record from the 'name\_details' table.

```
mysql> UPDATE name_details SET name='Mahe'WHERE id= 3
```

*Query OK, 1 row affected (0.01 sec)*

*Rows matched: 1 Changed: 1 Warnings: 0*

**Step 13:** Delete a record from the 'name\_details' table.

```
mysql> DELETE FROM name_details WHERE id
```

*=2; Query OK, 1 row affected (0.00 sec)*

**Step 14:** Retrieve and display all records from the 'name\_details' table.

```
mysql> select * from name_details;
```

id	name
1	Dhivya
3	Mahe

2 rows in set (0.01 sec)

**Step 15:** Exit the MySQL CLI

```
mysql> exit
```

*Bye*

## **Result:**

Thus, successfully installed MySQL, created a database, and performed CRUD operations to manage data efficiently on the Ubuntu instance.

**EX N0: 04**

**DATE: 25/07/2023**

## **DEPLOYMENT OF FLASK WEB APPLICATION WITH DOCKER**

### **Aim:**

To deploy a Flask web application using Docker on an Ubuntu instance with seamless containerization and port mapping.

### **Procedure:**

**Step 1:** Create an EC2 instance with Ubuntu in Amazon Machine Image.

**Step 2:** Update package information on the Ubuntu EC2 instance.

```
ubuntu@ip-172-31-28-40:~$ sudo apt update
```

**Step 3:** Install Docker on the Ubuntu instance by downloading the installation script using

‘curl’ and then executing the script using the ‘sh’ shell to perform the installation

```
ubuntu@ip-172-31-28-40:~$ curl -fsSL https://get.docker.com -o get-docker.sh
```

```
ubuntu@ip-172-31-28-40:~$ sh get-docker.sh
```

**Step 4:** Check the version of docker to verify installation.

```
ubuntu@ip-172-31-28-40:~$ docker --version
```

*Docker version 24.0.5, build ced0996*

**Step 5:** Clone the GitHub repository containing the Flask web application and docker file.

```
ubuntu@ip-172-31-28-40:~$ git clone  
https://github.com/yeshwanthlm/docker-flaskdemo
```

**Step 6:** List files and directories in the directory in long format.

```
ubuntu@ip-172-31-28-40:~$ ls -ltr
```

```
total 28
-rw-rw-r-- 1 ubuntu ubuntu 21926 Jul 17 08:22 get-docker.sh
drwxrwxr-x 4 ubuntu ubuntu 4096 Jul 17 08:26 docker-flask-
demo
```

**Step 7:** Navigate to the cloned directory.

```
ubuntu@ip-172-31-28-40:~$ cd docker-flask-demo
```

**Step 8:** List the contents of the directory to see the files.

```
ubuntu@ip-172-31-28-40:~ docker-flask-demo$ ls -
```

```
|tr
```

```
total 20 drwxrwxr-x 2 ubuntu ubuntu 4096 Jul 15
16:13 templates

-rw-rw-r-- 1 ubuntu ubuntu 5 Jul 15 16:13 requirements.txt
-rw-rw-r-- 1 ubuntu ubuntu 2258 Jul 15 16:13 app.py
-rw-rw-r-- 1 ubuntu ubuntu 687 Jul 15 16:13 Jenkinsfile
-rw-rw-r-- 1 ubuntu ubuntu 112 Jul 15 16:13 Dockerfile
```

**Step 9:** Display the contents of the Dockerfile in the terminal.

```
ubuntu@ip-172-31-28-40:~/docker-flask-demo$ cat Dockerfile
```

```
FROM python:3.6
```

```
RUN pip install flask
```

```
COPY . /opt/
```

```
EXPOSE 8080
```

```
WORKDIR /opt
```

**Step 10:** Build the Docker image.

```
ubuntu@ip-172-31-28-40:~/docker-flask-demo$ sudo docker build -t
```

```
amc/flaskapp.
```

**Step 11:** List all the Docker images available on the system.

```
ubuntu@ip-172-31-28-40:~/docker-flask-demo$ sudo docker images
```

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
amc/flaskapp	latest	5f66ed14a643	2 days ago	913MB

**Step 12:** Run the Flask web application as a Docker container and map port 8080 to port 80 on the host.

```
ubuntu@ip-172-31-28-40:~/docker-flask-demo$ sudo docker run -d -p 80:8080  
e85ea9eeabdb
```

**Step 13:** List all the running containers on the system.

```
ubuntu@ip-172-31-28-40:~/docker-flask-demo$ sudo docker ps
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS
77904b80ac1d	5f66ed14a643	"python app.py"	19 seconds ago	Up 18 seconds

POR	NAMES
TS	
0.0.0.0:80->8080/tcp, :::80->8080/tcp	reverent_williamson

**Step 14:** Allow Custom TCP port 80 in Inbound rules.

**Step 15:** Now open <Public\_IP\_Address\_of\_EC2\_Instance>:80 in Browser

## **Output:**



## **Result:**

The Flask web application is deployed using Docker, and the container is running, and accessible on port 80 of the host machine.

**EX N0: 05**  
**DATE: 03/08/2023**

## **SETTING UP SIMPLE PIPELINE USING GIT, JENKINS, AND DOCKER IN LOCAL MODE ON AN UBUNTU INSTANCE**

### **Aim:**

To set up a simple pipeline using Git, Jenkins, and Docker in local mode on an Ubuntu instance.

### **Procedure:**

**Step 1:** Create an EC2 instance with Ubuntu in Amazon Machine Image.

**Step 2:** Update package information on the Ubuntu EC2 instance.

```
ubuntu@ip-172-31-47-227:~$ sudo apt-get update
```

```
ubuntu@ip-172-31-47-227:~$ sudo apt update
```

**Step 3:** Install Docker on the Ubuntu instance by downloading the installation script using

‘curl’ and then executing the script using the ‘sh’ shell to perform the installation

```
ubuntu@ip-172-31-47-227:~$ curl -fsSL https://get.docker.com -o get-docker.sh
```

```
ubuntu@ip-172-31-47-227:~$ sh get-docker.sh
```

**Step 4:** Check the version of docker to verify installation.

```
ubuntu@ip-172-31-47-227:~$ docker --version
```

*Docker version 24.0.4, build 3713ee1*

**Step 5:** Create a docker hub account in DockerHub website

**Step 6:** Login docker

```
ubuntu@ip-172-31-47-227:~$ sudo docker login -u <username> -p  
<password>
```

Replace <username> and <password> with DockerHub username and password.

**Step 7:** Install OpenJDK 17 (Java Runtime Environment).

```
ubuntu@ip-172-31-47-227:~$ sudo apt install openjdk-17-jre
```

**Step 8:** Check the Java version to ensure it is installed correctly.

```
ubuntu@ip-172-31-47-227:~$ java -version
```

*openjdk version "17.0.7" 2023-04-18*

*OpenJDK Runtime Environment (build 17.0.7+7-Ubuntu0ubuntu122.04.2)*

*OpenJDK 64-Bit Server VM (build 17.0.7+7-Ubuntu-0ubuntu122.04.2, mixed mode, sharing)*

**Step 9:** Import Jenkins Repository Key.

```
ubuntu@ip-172-31-47-227:~$ curl -fsSL  
https://pkg.jenkins.io/debian/jenkins.io2023.key |sudo tee  
/usr/share/keyrings/jenkins-keyring.asc > /dev/null
```

**Step 10:** Add Jenkins Repository to Sources List.

```
ubuntu@ip-172-31-47-227:~$ echo deb [signed-by=/usr/share/keyrings/Jenkins  
keyring.asc]https://pkg.jenkins.io/debian binary/ | sudo tee  
/etc/apt/sources.list.d/jenkins.list > /dev/null
```

**Step 11:** Update the package lists from all repositories, including the newly added Jenkins repository.

```
ubuntu@ip-172-31-47-227:~$ sudo apt-get update
```

**Step 12:** Install Jenkins.

```
ubuntu@ip-172-31-47-227:~$ sudo apt-get install Jenkins
```

**Step 13:** Start Jenkins service and check its status.

```
ubuntu@ip-172-31-47-227:~$ sudo systemctl start jenkins.service
```

```
ubuntu@ip-172-31-47-227:~$ sudo systemctl status Jenkins
```

• *jenkins.service - Jenkins Continuous Integration Server*

*Loaded: loaded (/lib/systemd/system/jenkins.service; enabled; vendor preset: enabled)*

*Active: active (running) since Fri 2023-07-21 12:50:18 UTC; 1min 57s ago*

*Main PID: 6680 (java)*

*Tasks: 38 (limit: 1111)*

*Memory: 292.6M*

*CPU: 1min 17.623s*

*CGroup: /system.slice/jenkins.service*

*└─6680 /usr/bin/java -Djava.awt.headless=true -jar /usr/share/java/jenkins.war --webroot=/var/cache/jenkins/war --httpPort=8080*

*Jul 21 12:49:30 ip-172-31-47-227 jenkins[6680]: c84c43b9adf14d60873f4185427328ba*

**Step 14:** Configure firewall to allow traffic on port 8080 for Jenkins, SSH

connections for remote access, port 8000 for container communication.

```
ubuntu@ip-172-31-47-227:~$ sudo ufw allow 8080
```

```
ubuntu@ip-172-31-47-227:~$ sudo ufw allow OpenSSH
```

```
ubuntu@ip-172-31-47-227:~$ sudo ufw allow 8000
```

**Step 15:** Enable the firewall.

```
ubuntu@ip-172-31-47-227:~$ sudo ufw enable
```

**Step 16:** Check the firewall status.

```
ubuntu@ip-172-31-47-227:~$ sudo ufw status
```

*Status: active*

<i>To</i>	<i>Action</i>	<i>From</i>
--	-----	---
8080	ALLOW	Anywhere
OpenSSH	ALLOW	Anywhere
8000	ALLOW	Anywhere
8080 (v6)	ALLOW	Anywhere (v6)
OpenSSH (v6)	ALLOW	Anywhere (v6)
8000 (v6)	ALLOW	Anywhere (v6)

**Step 17:** Retrieve the initialAdminPassword for Jenkins setup.

```
ubuntu@ip-172-31-47-227:~$ sudo cat
```

```
/var/lib/jenkins/secrets/initialAdminPassword
```

```
c84c43b9adf14d60873f4185427328ba
```

**Step 18:** Add the Jenkins user to the docker group for Docker access.

```
ubuntu@ip-172-31-25-220:~$ sudo usermod -aG docker Jenkins
```

**Step 19:** Restart Jenkins service to apply the changes.

```
ubuntu@ip-172-31-25-220:~$ sudo systemctl restart Jenkins
```

**Step 20:** Allow Custom TCP port 8080, 8000 in Inbound rules.

**Step 21:** Now open <Public\_IP\_Address\_of\_EC2\_Instance>:8080 in Browser to open Jenkins.

**Step 22:** Login with initialAdminPassword that we retrieved – Install suggested plugins – Setup new username and password.



**Welcome to Jenkins!**

dhivyadharshini

.....

Keep me signed in

**Sign in**

**Step 23:** Setup Credentials in Jenkins



# Jenkins

Dashboard >

+ New Item

People

Build History

Project Relationship

Check File Fingerprint

Manage Jenkins

My Views

<Next>

## Manage Jenkins

Search settings /

### System Configuration



#### System

Configure global settings and paths.



#### Tools

Configure tools, their locations and automatic installers.



#### Plugins

Add, remove, disable or enable plugins that can extend the functionality of Jenkins.



#### Nodes

Add, remove, control and monitor the various nodes that Jenkins runs jobs on.



#### Clouds

Add, remove, and configure cloud instances to provision agents on-demand.

### Security



#### Security

Secure Jenkins; define who is



#### Credentials

Configure credentials



#### Credential Providers

Configure the credential providers

<Next>

The screenshot shows the Jenkins 'Credentials' page under 'Manage Jenkins'. The top navigation bar includes 'Dashboard', 'Manage Jenkins', and 'Credentials'. The main title is 'Credentials'. Below it is a table header with columns: T, P, Store ↓, Domain, ID, and Name. A sub-section titled 'Stores scoped to Jenkins' is shown, featuring a table with a single row: 'System' (icon: gear) under '(global)'. There are also filter buttons for 'Icon', 'S', 'M', and 'L'.

<Next>

### Global credentials (unrestricted)

+ Add Credentials

Credentials that should be available irrespective of domain specification to requirements matching.

<Next>

<Next>

Now add your Docker hub User name and password and save it with Id (I have used test1).

Also make necessary changes in the Jenkinsfile like setting **credentials id** and **username of docker**

#### New credentials

Kind

Username with password

Scope ?

Global (Jenkins, nodes, items, all child items, etc)

Username ?

dhivyadd20

Treat username as secret ?

Password ?

.....

ID ?

test1

Description ?

docker credentials

Create

## Step 24: Create Pipeline in Jenkins



+ New Item

People

Build History

Project Relationship

Check File Fingerprint

Manage Jenkins

My Views

<Next>

Enter an item name of your wish.

Enter an item name

» Required field

**Freestyle project**  
This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system, and this can be even used for something other than software build.

**Pipeline**  
Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.

**Multi-configuration project**  
Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.

**OK**

<next>

Add some description

## General

Enabled 

### Description

automated builds

Plain text [Preview](#)

<next>

Build Triggers with Poll SCM.

This poll SCM with Schedule ‘\* \* \* \* \*’ will check the git repository every minute and if a commit is encounter it automatically build up...

### Build Triggers

Build after other projects are built [?](#)

Build periodically [?](#)

GitHub hook trigger for GITScm polling [?](#)

Poll SCM [?](#)

Schedule [?](#)

```
* * * * *
```

<next>

Display name – The same name given in item name

### Advanced Project Options

Advanced ^  Edited

Display Name [?](#)

app

<next>

Setting up pipeline definition from my GitHub Repository

## Pipeline

### Definition

Pipeline script from SCM

SCM ?

Git

Repositories ?

Repository URL ?

https://github.com/Dhivyadd20/jenkins.git

Credentials ?

- none -

Add ▾

Advanced ▾



<next>

Change the branch specifier according to your repository and save the pipeline

Branches to build ?

Branch Specifier (blank for 'any') ?

\*/main

Add Branch

Repository browser ?

(Auto)

Additional Behaviours

Add ▾

Script Path ?

Jenkinsfile

Lightweight checkout ?

Save Apply



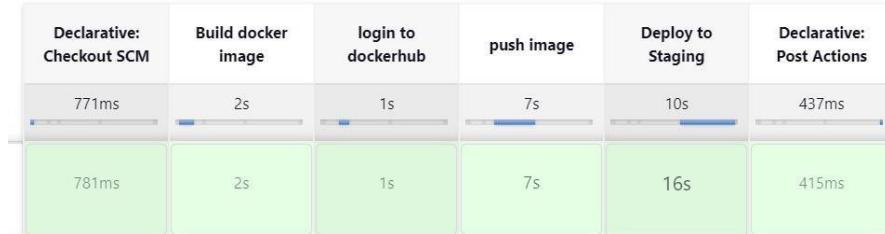
<next>

Now build the pipeline

- >Status
- Changes
- Build Now
- Configure
- Delete Pipeline
- Full Stage View
- Rename
- Pipeline Syntax
- Git Polling Log

<next>

Stages of pipeline will get executed.



**Step 25:** Now open <Public\_IP\_Address\_of\_EC2\_Instance>:8000 in Browser

## Result:

The local pipeline setup for Git, Jenkins, and Docker has been successfully established on the

Ubuntu instance, enabling efficient software development, testing, and deployment processes.

**EX N0: 06**

**DATE: 03/08/2023**

## **SETTING UP JENKINS ON AWS EC2 INSTANCE AND CREATING AN IMAGE**

### **Aim:**

To install Jenkins on an AWS EC2 instance, create an Amazon Machine Image (AMI) of the instance, launch a new EC2 instance from the AMI, and verify that Jenkins is installed on the new instance.

### **Procedure:**

**Step 1:** Create an EC2 instance with Ubuntu in Amazon Machine Image.

**Step 2:** Update package information on the Ubuntu EC2 instance.

```
ubuntu@ip-172-31-30-107:~$ sudo apt-get update
```

```
ubuntu@ip-172-31-30-107:~$ sudo apt update
```

**Step 3:** Install OpenJDK 17 (Java Runtime Environment).

```
ubuntu@ip-172-31-30-107:~$ sudo apt install openjdk-17-jre
```

**Step 4:** Check the Java version to ensure it is installed correctly.

```
ubuntu@ip-172-31-30-107:~$ java -version
```

*openjdk version "17.0.7" 2023-04-18*

**Step 5:** Import Jenkins Repository Key.

```
ubuntu@ip-172-31-30-107:~$ curl -fsSL https://pkg.jenkins.io/debian/jenkins.io-2023.key |  
sudo tee /usr/share/keyrings/jenkins-keyring.asc > /dev/null
```

**Step 6:** Add Jenkins Repository to Sources List.

```
ubuntu@ip-172-31-30-107:~$echo deb [signed-by=/usr/share/keyrings/jenkins-keyring.asc]  
https://pkg.jenkins.io/debian binary/ | sudo tee /etc/apt/sources.list.d/jenkins.list > /dev/null
```

**Step 7:** Update the package lists from all repositories, including the newly added Jenkins repository.

```
ubuntu@ip-172-31-30-107:~$ sudo apt-get update
```

**Step 8:** Install Jenkins.

```
ubuntu@ip-172-31-30-107:~$ sudo apt-get install jenkins
```

**Step 9:** Start Jenkins service and check its status.

```
ubuntu@ip-172-31-30-107:~$ sudo systemctl start jenkins.service
```

```
ubuntu@ip-172-31-30-107:~$ sudo systemctl status jenkins
```

● *jenkins.service - Jenkins Continuous Integration Server*

*Loaded: loaded (/lib/systemd/system/jenkins.service; enabled; vendor preset: enabled)*

*Active: active (running) since Sun 2023-09-24 12:50:18 UTC; 1min 57s ago*

*Main PID: 6680 (java)*

*Tasks: 38 (limit: 1111)*

*Memory: 292.6M*

*CPU: 1min 17.623s*

*CGroup: /system.slice/jenkins.service*

```
         └─6680 /usr/bin/java -Djava.awt.headless=true -jar  
          /usr/share/java/jenkins.war --webroot=/var/cache/jenkins/war --httpPort=8080
```

**Step 10:** Once Jenkins is installed and configured, navigate to the EC2 Management Console and stop the EC2 instance.

**Step 11:** Select your instance, right-click, and choose "Image" > "Create Image."

The screenshot shows the AWS EC2 Instances page. On the left, there's a sidebar with various navigation options like 'New EC2 Experience', 'Instances', 'Images', and 'Elastic Block Store'. The main area lists three instances: 'Docker + Jenkins' (stopped, t3.micro), 'Word press' (stopped, t3.micro), and 'Jenkins' (selected, stopped, t3.micro). A context menu is open over the 'Jenkins' instance, with the 'Create image' option highlighted by a yellow box. Other options in the menu include 'Launch instances', 'Launch instance from template', 'Migrate a server', 'Connect', 'Stop instance', 'Start instance', 'Reboot instance', 'Hibernate instance', 'Terminate instance', 'Instance settings', 'Networking', 'Security', and 'Image and templates' (which also has a 'Create image' option).

**Step 12:** Provide a name and description for the image and create it.

The screenshot shows the 'Create image' wizard. At the top, the path is EC2 > Instances > i-02b15e0b1e39e44a5 > Create image. The main form has fields for 'Image name' (containing 'Jenkins Image'), 'Image description - optional' (containing 'Image description'), and checkboxes for 'No reboot' and 'Enable'. Below this is the 'Instance volumes' section, which shows one EBS volume (/dev/sda) with a size of 8 GiB, EBS General Purpose S., IOPS 100, Throughput 100 MiB/s, Delete on termination checked, and Encrypted unchecked. There's a 'Create new snapshot from volume' button. A note says 'During the image creation process, Amazon EC2 creates a snapshot of each of the above volumes.' At the bottom, there's a 'Tags - optional' section with two radio buttons: 'Tag image and snapshots together' (selected) and 'Tag image and snapshots separately'. A note says '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.' Below that is an 'Add new tag' button and a note about adding up to 50 tags. At the very bottom right is a large 'Create image' button, which is highlighted with a yellow box.

**Step 13:** Navigate to Images > AMIs in the EC2 dashboard to see the images

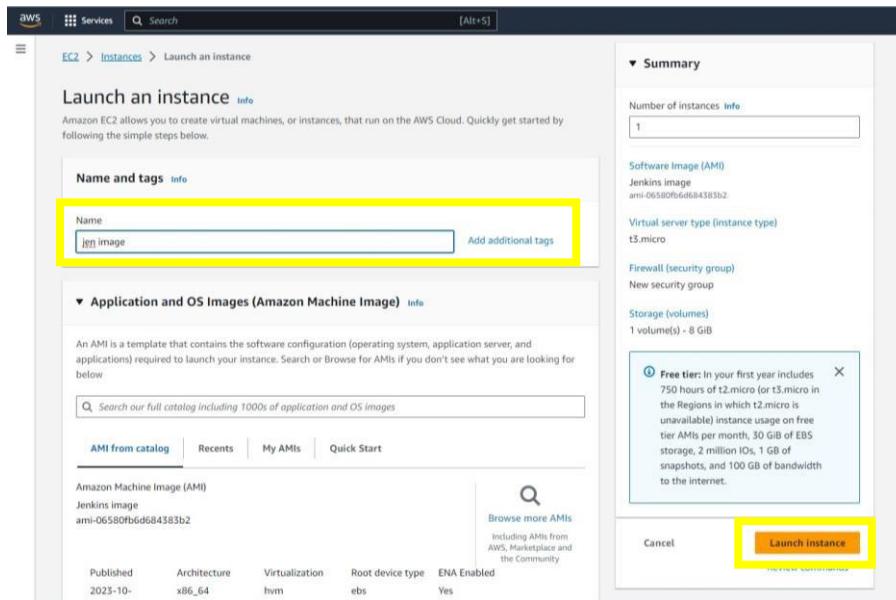
The screenshot shows the AWS EC2 Instances page. On the left, there's a sidebar with various navigation options like 'Instances', 'Images' (which is highlighted with a yellow box), and 'AMI Catalog'. The main area displays a table of three stopped t3.micro instances. A modal window titled 'Select an instance' is overlaid on the page, also showing the same three instances.

<after navigating>

**Step 14:** To launch a new instance with the created image, select the image and click Launch instance from AMI

The screenshot shows the AWS AMI (Amazon Machine Images) page. It lists a single AMI named 'Jenkins image' which is marked as 'Available'. The 'Launch instance from AMI' button at the top right of the table is highlighted with a yellow box.

**Step 15:** Follow the instance launch wizard, giving name, choosing key pair, configuring security groups and other settings as needed and launch.



**Step 16:** Now navigate to the EC2 instances dashboard and you can see the newly created instance from image.

Instances (4) <a href="#">Info</a>													
<a href="#">C</a> <a href="#">Connect</a> <a href="#">Instance state</a> <a href="#">Actions</a> <a href="#">Launch instances</a> <a href="#">▼</a>													
<input type="text"/> <a href="#">Find instance by attribute or tag (case-sensitive)</a>													
Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS	Public IPv4 IP	Elastic IP	IP	Port	Protocol	Interface
Docker + Jenkins	i-0ff8a141d1a933e9f5	<a href="#">Stopped</a>	<a href="#">@</a> <a href="#">Q</a>	t3.micro	-	No alarms	+	eu-north-1a	-	-	-	-	-
Word press	i-0f7bec8bf2f058ad5	<a href="#">Stopped</a>	<a href="#">@</a> <a href="#">Q</a>	t3.micro	-	No alarms	+	eu-north-1a	-	-	-	-	-
Jenkins	i-02b15e0b1e39ed4a45	<a href="#">Stopped</a>	<a href="#">@</a> <a href="#">Q</a>	t3.micro	-	No alarms	+	eu-north-1a	-	-	-	-	-
<b>jen image</b>	i-085f516e570f411b2	<a href="#">Running</a>	<a href="#">@</a> <a href="#">Q</a>	t3.micro	<a href="#">Initializing</a>	No alarms	+	eu-north-1a	ec2-51-20-83-139.eu-n...	51.20.83.139	-	-	-

**Step 17:** Connect to the new EC2 instance using SSH. Check if Jenkins is installed and running.

```
root@ip-172-31-24-134:~# jenkins --version
```

2.424

```
root@ip-172-31-24-134:~# sudo systemctl status jenkins
```

● *jenkins.service - Jenkins Continuous Integration Server*

*Loaded: loaded (/lib/systemd/system/jenkins.service; enabled; vendor preset: enabled)*

*Active: active (running) since Mon 2023-10-02 08:22:25 UTC; 2min 50s ago*

*Main PID: 455 (java)*

*Tasks: 38 (limit: 1111)*

*Memory: 308.8M*

*CPU: 1min 6.098s*

*CGroup: /system.slice/jenkins.service*

```
└─455 /usr/bin/java -Djava.awt.headless=true -jar  
/usr/share/java/jenkins.war --webroot=/var/cache/jenkins/war --httpPort=8080
```

## **Result:**

Thus, set up Jenkins on an EC2 instance, created a reusable image, and launch new instances with Jenkins pre-installed, making it easier to scale and manage Jenkins-based infrastructure in the cloud.

**EX N0: 07**

**DATE: 26/08/2023**

## **DEPLOYING MYSQL ON AWS, CREATING SNAPSHOTS**

### **Aim:**

To install and configure MySQL on an AWS EC2 instance, create a snapshot of the database, perform various database operations, and then restore the database from the snapshot.

### **Procedure:**

**Step 1:** Create an EC2 instance with Ubuntu in Amazon Machine Image

**Step 2:** Update package information on the Ubuntu EC2 instance.

```
ubuntu@ip-172-31-16-6:~$ sudo apt update
```

**Step 3:** Install MySQL server on the instance.

```
ubuntu@ip-172-31-16-6:~$ sudo apt install mysql-server
```

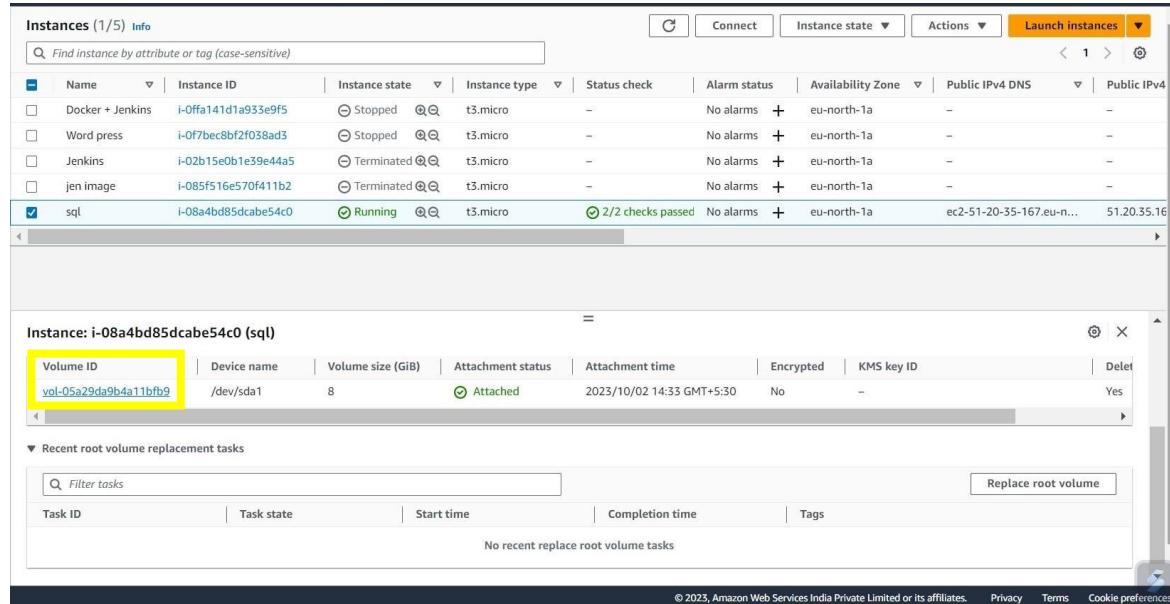
**Step 4:** Check the status of the MySQL service to ensure it's running.

```
ubuntu@ip-172-31-16-6:~$ sudo systemctl status mysql
```

```
● mysql.service - MySQL Community Server
   Loaded: loaded (/lib/systemd/system/mysql.service; enabled; vendor preset: enabled)
     Active: active (running) since Mon 2023-10-02 09:05:35 UTC; 11s ago
       Docs: man:mysqld(8)
   Process: 2901 ExecStartPre=/usr/share/mysql/mysql-systemd-start pre
             (code=exited, status=0/SUCCESS)
  Main PID: 2909 (mysqld)
     Tasks: 1 (limit=4672)
```

*Status: "Server is operational"*  
*Tasks: 38 (limit: 1111)*  
*Memory: 356.9M*  
*CPU: 1.052s*  
*CGroup: /system.slice/mysql.service*  
 └─2909 /usr/sbin/mysqld

## Step 5: Navigate to the EC2 Management Console and select your instance volume id.



The screenshot shows the AWS EC2 Instances page. The 'sql' instance is selected and highlighted with a yellow box. In the instance details section, the Volume ID 'vol-05a29da9b4a11bfb9' is also highlighted with a yellow box.

Volume ID	Device name	Volume size (GiB)	Attachment status	Attachment time	Encrypted	KMS key ID	Delete on termination
vol-05a29da9b4a11bfb9	/dev/sda1	8	Attached	2023/10/02 14:33 GMT+5:30	No	-	Yes

<navigates to volume dashboard>

## Step 6: Select the volume, Actions > Create snapshot



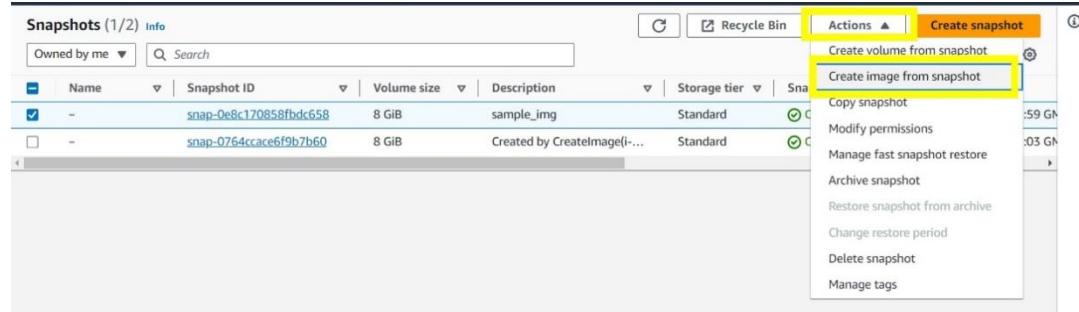
The screenshot shows the AWS Volumes page. The volume 'vol-05a29da9b4a11bfb9' is selected and highlighted with a yellow box. In the Actions menu, the 'Create snapshot' option is highlighted with a yellow box.

Name	Volume ID	Type	Size	IOPS	Throughput	Snapshot	Created
-	vol-05a29da9b4a11bfb9	gp2	8 GiB	100	-	snap-07ac292...	2023/10/12 14:33:20

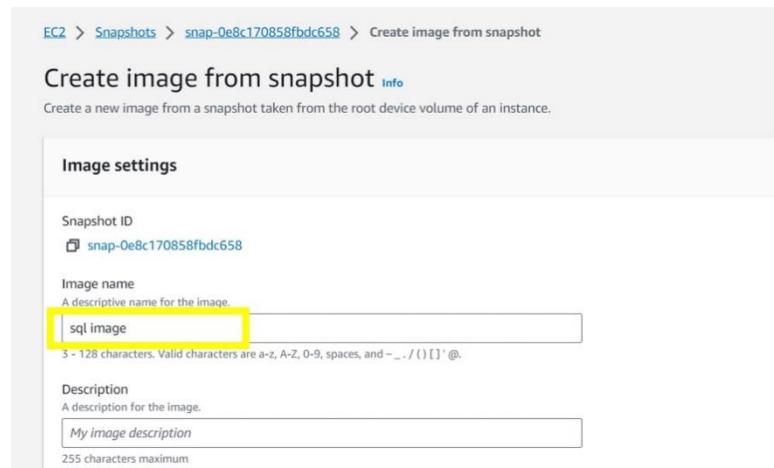
## Step 7: Navigate to Elastic Block Store > Snapshots in the EC2 dashboard to see the snapshots.



### Step 8: Select the snapshot, Actions > Create image from snapshot.



### Step 9: Give image name and create image.



**Block device mappings - optional** Info

Provisioned IOPS SSD (io2) volumes with a size greater than 16 TiB, IOPS greater than 64,000, or IOPS:GiB ratio greater than 500:1 are supported only with instance types that support io2 Block Express.

<b>Volume 1</b>		
Device type	Device name	Snapshot
Root	/dev/sda1	snap-0e8c170858fbdc658
Size (GiB)	Volume type	IOPS
8	General Purpose SSD (gp2)	100 / 3000
Throughput (MB/s)	Termination behavior	Encryption
-	<input checked="" type="checkbox"/> Delete on termination	<input type="checkbox"/> Encrypt volume
<a href="#">Add volume</a>		
<a href="#">Cancel</a> <a href="#" style="background-color: orange; color: white; padding: 5px;">Create image</a>		

**Step 10:** Navigate to Images > AMIs in the EC2 dashboard to see the images

New EC2 Experience

EC2 Dashboard EC2 Global View Events

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

Images AMIs AMI Catalog

Snapshots (1) Info

Name	Snapshot ID	Volume size	Description
-	snap-0fe1285c3ab303f8c	8 GiB	sql snapshot

<After Navigating>

**Step 11:** To launch a new instance with the created image, select the image and click Launch instance from AMI

New EC2 Experience

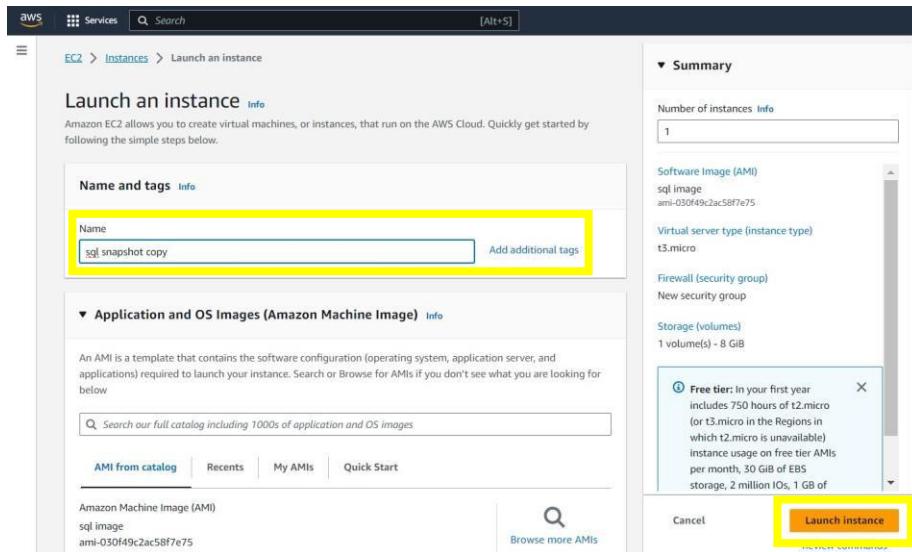
EC2 Dashboard EC2 Global View Events

Amazon Machine Images (AMIs) (1/1) Info

Name	AMI ID	AMI name	Source	Owner	Visibility	Status
<input checked="" type="checkbox"/>	ami-030f49c2ac58f7e75	sql image	290834569856/sql image	290834569856	Private	<span style="color: green;">Available</span>

[Launch instance from AMI](#)

**Step 12:** Follow the instance launch wizard, giving name, choosing key pair, configuring security groups and other settings as needed and launch.



**Step 13:** Now navigate to the EC2 instances dashboard and you can see the newly created instance from image.

Instances (6) <a href="#">Info</a>										
<a href="#">C</a> <a href="#">Connect</a> <a href="#">Instance state</a> <a href="#">Actions</a> <a href="#">Launch instances</a>										
<input type="text"/> 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	
Docker + Jenkins	i-Offa141d1a933e9f5	Stopped	t3.micro	-	No alarms	+ eu-north-1a	-	-	-	-
Word press	i-0f7becbbf2f038ad3	Stopped	t3.micro	-	No alarms	+ eu-north-1a	-	-	-	-
Jenkins	i-02b15e0b1e39e44a5	Terminated	t3.micro	-	No alarms	+ eu-north-1a	-	-	-	-
jen image	i-085f15e6570f411b2	Terminated	t3.micro	-	No alarms	+ eu-north-1a	-	-	-	-
sql	i-08a4bd85dcabe54c0	Running	t3.micro	2/2 checks passed	No alarms	+ eu-north-1a	ec2-51-20-35-167.eu-n...	51.20.35.16		
sql snapshot c...	i-00d1aa2961010df7c	Running	t3.micro	Initializing	No alarms	+ eu-north-1a	ec2-16-16-202-238.eu...	16.16.202.2		

**Step 14:** Connect to the new EC2 instance using SSH. Check if MySQL is installed and running.

```
ubuntu@ip-172-31-16-6:~$ sudo systemctl status mysql
```

- *mysql.service - MySQL Community Server*

*Loaded: loaded (/lib/systemd/system/mysql.service; enabled; vendor preset: enabled)*  
*Active: active (running) since Mon 2023-10-02 09:26:21 UTC; 53s ago*  
*Process: 2901 ExecStartPre=/usr/share/mysql/mysql-systemd-start pre (code=exited, status=0/SUCCESS)*  
*Main PID: 2909 (mysqld)*  
*Status: "Server is operational"*  
*Tasks: 38 (limit: 1111)*  
*Memory: 356.9M*  
*CPU: 1.052s*  
*CGroup: /system.slice/mysql.service*  
└─2909 /usr/sbin/mysqld

## **Result:**

Thus, set up MySQL on an EC2 instance, created a snapshot for backup and restored the database to another instance, providing data resilience and disaster recovery capabilities in the cloud.

**EX N0: 07**

**DATE: 02/09/2023**

## **DEPLOYING WORDPRESS USING BITNAMI'S AMAZON MARKETPLACE AMI AND CREATING A USER PROFILE**

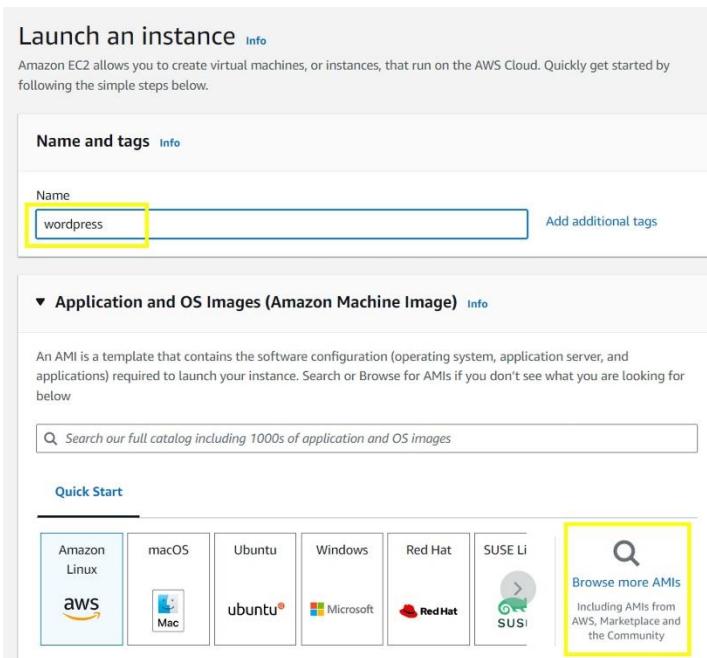
### **Aim:**

To deploy a WordPress instance on Amazon Web Services (AWS) using Bitnami's WordPress Amazon Machine Image (AMI) available on the AWS Marketplace and create a user profile on the WordPress website.

### **Procedure:**

**Step 1:** In the AWS Management Console, navigate to the EC2 service, Click on "Launch Instance."

- In the "AWS Marketplace" section, search for "Bitnami WordPress."
- Select the Bitnami WordPress AMI from the list.



## Choose an Amazon Machine Image (AMI)

An AMI is a template that contains the software configuration (operating system, application server, and applications) required to launch your instance. You can select an AMI provided by AWS, our user community, or the AWS Marketplace; or you can select one of your own AMIs.

<now launch the instance>

## Step 2: Open the public IP address of the created instance

<next>

[EC2](#) > [Instances](#) > i-0f7bec8bf2f038ad3

**Instance summary for i-0f7bec8bf2f038ad3 (Word press)** [Info](#)

Updated less than a minute ago

Instance ID	Public IPv4 address	Private IPv4 addresses
i-0f7bec8bf2f038ad3 (Word press)	<a href="#">13.53.170.213 [open address]</a>	<a href="#">172.31.23.85</a>
IPv6 address	Instance state	Public IPv4 DNS
-	<a href="#">Running</a>	<a href="#">ec2-13-53-170-213.eu-north-1.compute.amazonaws.com [open address]</a>

<after opening the public Ip address>

The screenshot shows a web browser window with three tabs:

- Instance details | EC2 | eu-north-1
- User's blog
- Generate Flask requirements.txt

The User's blog tab is active. The URL in the address bar is <https://13.53.170.213>. The page content is:

# Mindblown: a blog about philosophy.

## Hello world!

Welcome to WordPress. This is your first post.  
Edit or delete it, then start writing!

**Step 3: Move to the admin page by adding /wp-admin after the Ip address.**

The screenshot shows a web browser window with three tabs:

- Get system log | EC2 | eu-north-1
- User's blog
- Generate Flask requirements.txt

The User's blog tab is active. The URL in the address bar is <https://13.53.128.10/wp-admin>. The page content is:

Log In < User's blog — WordPress - <https://13.53.128.10/wp-admin>

Log In < User's blog — WordPress - [https://13.53.128.10/wp-login.php?redirect\\_to=https%3A%2F%2F13.53.128.10%2Fwp-admin%2F&reauth=1](https://13.53.128.10/wp-login.php?redirect_to=https%3A%2F%2F13.53.128.10%2Fwp-admin%2F&reauth=1)

<after adding, the page would be>

The screenshot shows the WordPress login page. The page features the classic blue 'W' logo at the top. Below it is a login form with fields for 'Username or Email Address' and 'Password'. There is also a 'Remember Me' checkbox and a 'Log In' button. At the bottom of the form, there are links for 'Lost your password?' and 'Go to User's blog'.

**Step 4: Get username and password from System log (Actions > Monitor and troubleshoot > Get system log)**

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone
Docker + Jenkins	i-0ffa141d1a933e9f5	Stopped	t3.micro	-	No alarms	eu-north-1a
Word press	i-0f7bec8bf2f038ad3	Running	t3.micro	Initializing	No alarms	eu-north-1a
sql snapshot c...	i-00d1aa2961010df7c	Terminated	t3.micro	-	No alarms	eu-north-1a
sql	i-08a4bd85dcabe54c0	Terminated	t3.micro	-	No alarms	eu-north-1a
wordpress	i-08fa8596ce9bba3f6	Terminated	t3.micro	-	No alarms	eu-north-1a

<you can see username and password here in system log>

```

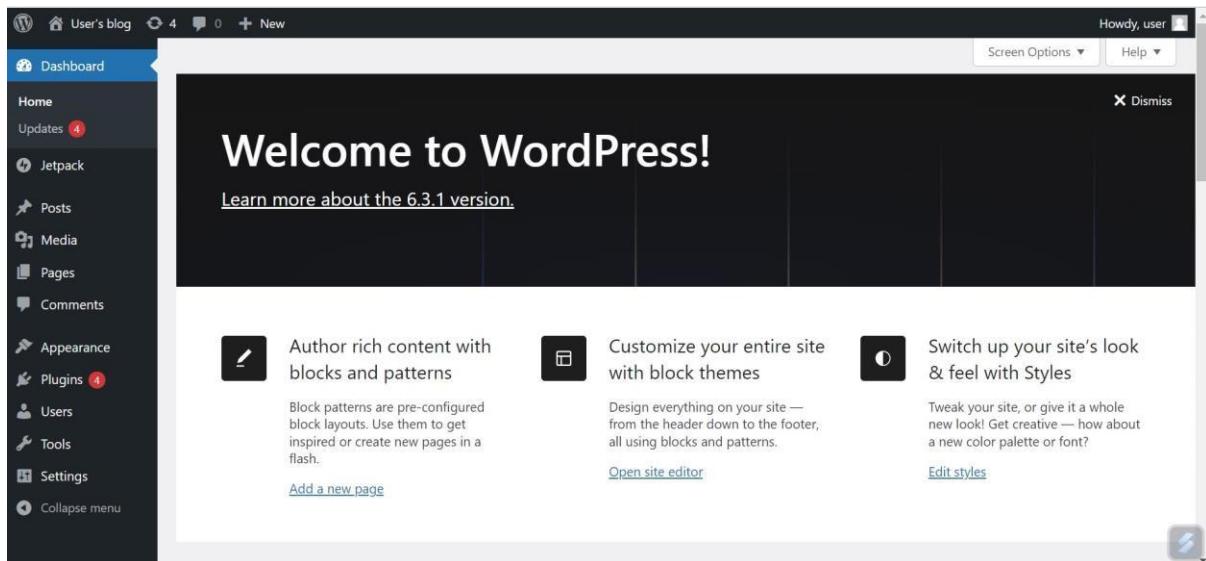
[ 60.917661] bitnami[618]: 650000+0 records in
[ 60.933642] bitnami[618]: 650000+0 records out
[ 60.934780] bitnami[618]: 665600000 bytes (666 MB, 635 MiB) copied, 3.37504 s, 197 MB/s
[ 62.041607] bitnami[618]: Setting up swapspace version 1, size = 634.8 MiB (65595904 bytes)
[ 62.043817] bitnami[618]: no label, UUID=b2269c54-c2c2-49f8-9a24-0f68d3b4a64a
[ 62.554841] Adding 649996k swap on /mnt/.bitnami.swap. Priority:-2 extents:26 across:305844k SSFS
[ 62.063205] bitnami[618]: ## 2023-10-02 10:27:45+00:00 ## INFO ## Running /opt/bitnami/var/init/pre-start/030_get_default_passwords...
[ 62.326108] bitnami[618]: #####
[ 62.328014] bitnami[618]: #
[ 62.329744] bitnami[618]: #      Setting Bitnami application password to 'pss:7V=SGy/U'
[ 62.331495] bitnami[618]: #      (the default application username is 'user')
[ 62.333050] bitnami[618]: #
[ 62.334527] bitnami[618]: #####
[ 62.380465] bitnami[607]: ## 2023-10-02 10:27:45+00:00 ## INFO ## Running first-boot...
[ 63.013055] bitnami[1117]: 2023-10-02T10:27:46.044Z - info: Saving configuration info to disk
[ 63.169996] bitnami[1117]: (node:1116) [DEP0005] DeprecationWarning: Buffer() is deprecated due to security and usability issues. Please use the Buffer.alloc(), Buffer.from() and Buffer.from()
[ 63.176109] bitnami[1117]: (Use `node --trace-deprecation ...` to show where the warning was created)
[ 64.117382] bitnami[1117]: 2023-10-02T10:27:47.149Z - info: Saving configuration info to disk
[ 64.123920] bitnami[1117]: 2023-10-02T10:27:47.155Z - warn: No peerAddress provided. Skipping hosts file section
[ 64.723489] bitnami[1117]: 2023-10-02T10:27:47.755Z - info: Initializing module bndDiagnostic

```

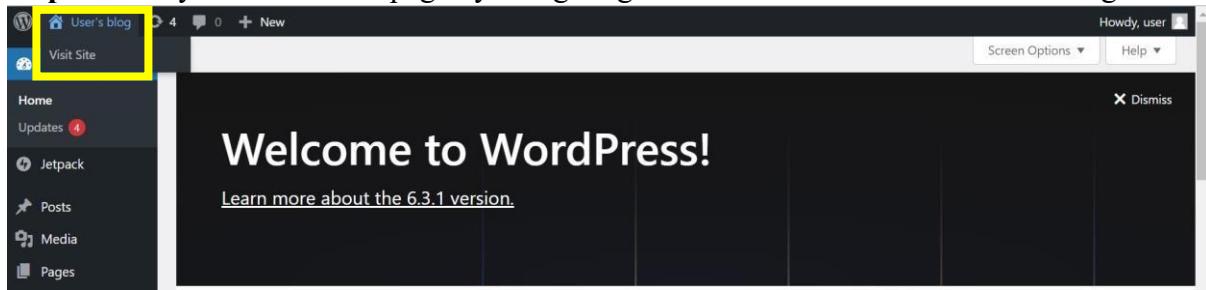
**Step 5: Login using the credentials and it will navigate to the dashboard page.**



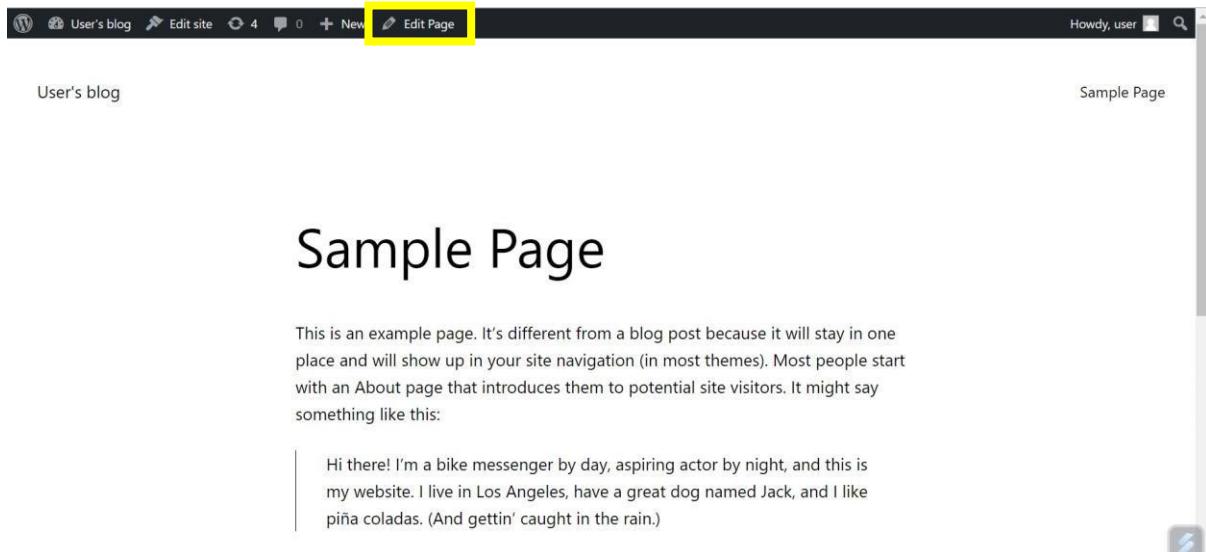
<Dashboard page>



**Step 6:** Now you can edit the page by navigating to User's blog > Visit Site > Edit Page



<next>



**Step 7:** Now you can start building your profile, after editing click on 'Update'.

Dhivya Dharshini M

**Summary:** Passionate data science student eager to earn and solve complex problems through data-driven insights.

**Contact Info:**

Phone Number: 8124888111

Email Id: dhivyadharshinim20@gmail.com

Visibility: Public  
Published: October 10, 2023 7:23 pm UTC+0  
Template: Pages  
URL: 3.81.110.140/dhivyadharshini-m/  
Author: user  
Buttons: Switch to draft, Move to trash

**Step 8:** Now navigate to the public Ip address, your contents will be updated.

# Dhivya Dharshini M

**Summary:** Passionate data science student eager to earn and solve complex problems through data-driven insights.

**Contact Info:**

Phone Number: 8124888111

Email Id: dhivyadharshinim20@gmail.com

## Result:

Thus, deployed a WordPress website using Bitnami's pre-configured AMI from the AWS Marketplace, created and updated a user profile on the WordPress platform.

**EX N0: 09**  
**DATE: 13/09/2023**

## DEPLOYING A WEBSITE IN AZURE WEB APP

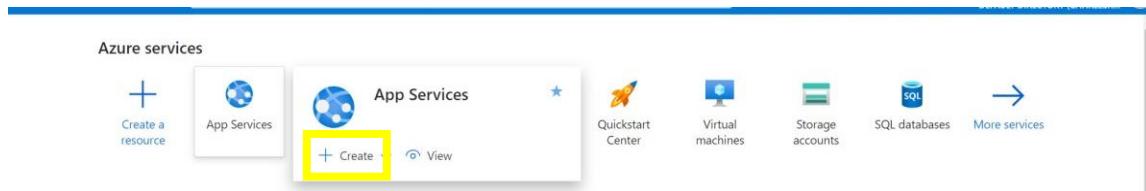
### Aim:

To deploy a website to Azure Web App using the Azure CLI, ensuring that the website is accessible online.

### Procedure:

#### Step 1: Create a Web App in Azure App Service

1. In the Azure Portal, navigate to the Azure App Service section.
2. Click on "Create" to create a new web app.



<next>

A screenshot of the Azure Portal's 'App Services' blade. The '+ Web App' button is highlighted with a yellow box. The blade shows various filtering options and sorting columns.

3. Provide a unique name for your web app. Choose your subscription, resource group, and App Service plan (or create a new one).
4. Select your runtime stack (e.g., .NET, Node.js, Python). Configure other settings like the operating system, region, and whether you want to enable or disable application insights.

## 5. Review your settings and click "Create" to provision the web app.

The screenshot shows the 'Create Web App' wizard in the Microsoft Azure portal. In the 'Project Details' section, the 'Subscription' dropdown is set to 'Azure for Students' and the 'Resource Group' dropdown is set to 'MOL-AppService'. In the 'Instance Details' section, the 'Name' field is filled with 'pizza-launchesss' and the 'Publish' dropdown is set to 'Code'. A yellow box highlights the 'Subscription' and 'Resource Group' dropdowns.

Project Details

Select a subscription to manage deployed resources and costs. Use resource groups like folders to organize and manage all your resources.

Subscription \* ⓘ

Resource Group \* ⓘ

Azure for Students

MOL-AppService

Create new

Instance Details

Need a database? Try the new Web + Database experience. ⓘ

Name \*

pizza-launchesss

.azurewebsites.net

Publish \*

Code  Docker Container  Static Web App

<next>

The screenshot shows the continuation of the 'Create Web App' wizard. In the 'Runtime stack' section, 'ASP.NET V4.8' is selected. In the 'Operating System' section, 'Windows' is selected. In the 'Region' section, 'East US' is selected. In the 'Pricing plans' section, 'Windows Plan (East US)' is selected with the value 'charlesreena2001\_asp\_7539 (F1)'. In the 'Zone redundancy' section, 'Free F1 (Shared infrastructure)' is selected. At the bottom, there are navigation buttons: 'Review + create', '< Previous', and 'Next : Deployment >'.

Runtime stack \*

ASP.NET V4.8

Operating System \*

Linux  Windows

Region \*

East US

Not finding your App Service Plan? Try a different region or select your App Service Environment.

Pricing plans

App Service plan pricing tier determines the location, features, cost and compute resources associated with your app.

Learn more ⓘ

Windows Plan (East US) \* ⓘ

charlesreena2001\_asp\_7539 (F1)

Create new

Pricing plan

Free F1 (Shared infrastructure)

Zone redundancy

Review + create

< Previous

Next : Deployment >

## Step 2: Connect to the cloud shell using Bash

The screenshot shows the Microsoft Azure portal interface. The top navigation bar includes the Azure logo, search bar, and user profile. The user profile shows the email 'charlesreena2001@gmail.com' and the text 'DEFAULT DIRECTORY (CHARLESREENA)'. Below the navigation bar, there is a toolbar with various icons. The 'Azure services' button is highlighted with a yellow box.

portal.azure.com/#home

Microsoft Azure

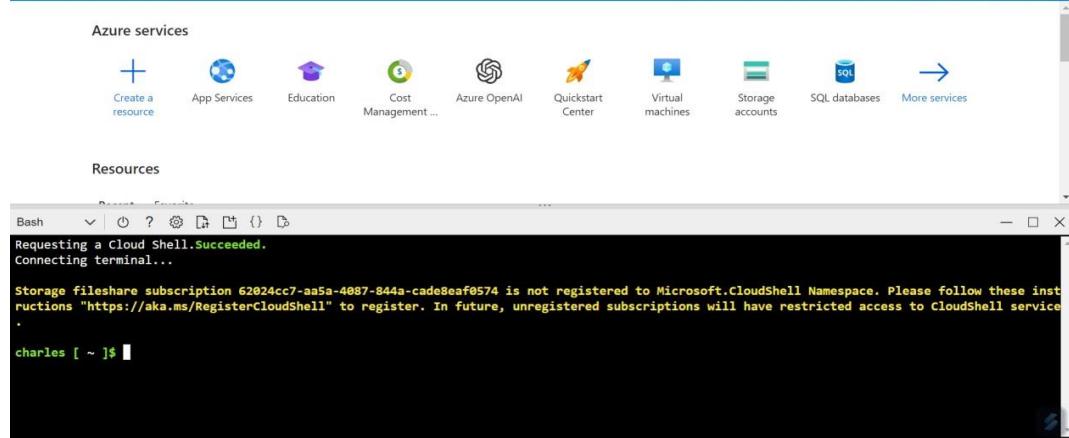
Search resources, services, and docs (G+ /)

charlesreena2001@gmail.com

DEFAULT DIRECTORY (CHARLESREENA)

Azure services

<then it shows the cloud shell>



**Step 3:** Use the ‘**wget**’ command to download the website files from a source URL. In this example, we downloaded a zip file named "antique-cafe.zip."

```
charles [ ~ ]$ wget https://www.free-css.com/assets/files/free-  
css-templates/download/page295/antique-cafe.zip
```

**Step 4:** Use the ‘**unzip**’ command to extract the contents of the downloaded zip file. This will create a directory containing the website files.

```
charles [ ~ ]$ ls  
antique-cafe.zip azure-mol-samples-2nd-ed clouddrive  
charles [ ~ ]$ unzip antique-cafe.zip
```

**Step 5:** Set your Git user email and name using the following commands

```
charles [ ~ ]$ git config --global user.email "your-email@example.com"  
charles [ ~ ]$ git config --global user.name "your-username"
```

**Step 6:** Change your working directory to the directory containing your website files

```
charles [ ~ ]$ ls
```

```
2126_antique_cafe antique-cafe.zip azure-mol-samples-2nd-ed clouddrive
```

```
charles [ ~ ]$ cd 2126_antique_cafe
```

```
charles [ ~/2126_antique_cafe ]$ ls -ltr
```

*total 36*

```
-rw-r--r-- 1 charles charles 510 Jul 30 2019 'ABOUT THIS
```

```
TEMPLATE.txt' drwxr-xr-x 2 charles charles 4096 Sep 29 2021 webfonts
```

```
drwxr-xr-x 2 charles charles 4096 Sep 29 2021 js drwxr-xr-x 2 charles
```

```
charles 4096 Sep 29 2021 img -rw-r--r-- 1 charles charles 15522 Sep 30
```

```
2021 index.html drwxr-xr-x 2 charles charles 4096 Sep 30 2021 css
```

### **Step 7:** Initialize a Git Repository

```
charles [ ~/2126_antique_cafe ]$ git init
```

### **Step 8: Add and Commit Website Files**

```
charles [ ~/2126_antique_cafe ]$ git add .
```

```
charles [ ~/2126_antique_cafe ]$ git commit -m "css!"
```

### **Step 9:** Use the Azure CLI to configure deployment credentials for your Azure Web App.

```
charles [ ~/2126_antique_cafe ]$ az webapp deployment source config-local-git --name  
pizza-launchess --resource-group MOL-AppService
```

### **Step 10:** Retrieve the publishing credentials for your Azure Web App using the Azure CLI.

```
charles [ ~/2126_antique_cafe ]$ az webapp deployment list-publishing-credentials --name  
pizza-launchess --resource-group MOL-AppService
```

```
{
  "id": "/subscriptions/62024cc7-aa5a-4087-844a-
cade8eaf0574/resourceGroups/MOL-
AppService/providers/Microsoft.Web/sites/pizzalaunchess/publishingcredentials/$pizz
a-launchess",
  "kind": null,
  "location": "East US",
  "name": "pizza-launchess",
  "publishingPassword": "6bBoiQrt2WINrzWlaXG3bKjDSsqmDlc6QetjJPdoynCckh1W6nKhkCljjcZ3",
  "publishingPasswordHash": null,
  "publishingPasswordHashSalt": null,
  "publishingUserName": "$pizza-launchess",
  "resourceGroup": "MOL-AppService",
  "scmUri": "https://$pizza-
launchess:6bBoiQrt2WINrzWlaXG3bKjDSsqmDlc6QetjJPdoynCckh1W6nKhkCljjcZ
3 @pizza-launchess.scm.azurewebsites.net",
  "type": "Microsoft.Web/sites/publishingcredentials"
}
```

**Step 11:** Add the Azure Git repository as a remote to your local Git repository. Use the actual credentials you obtained in previous step.

```
charles [ ~/2126_antique_cafe ]$ git remote add azure
'https://$pizzalaunchess:6bBoiQrt2WINrzWlaXG3bKjDSsqmDlc6QetjJPdoynCckh1W6nKhkCljj
cZ3@pizzalaunchess.scm.azurewebsites.net'
```

**Step 12: Push Website to Azure**

```
charles [ ~/2126_antique_cafe ]$ git push -f azure
master
```

**Step 13:** Navigate to the default domain you obtained from Web App Overview

Home >

**pizza-launchess** Web App

Search

Overview

Activity log

Access control (IAM)

Tags

Diagnose and solve problems

Microsoft Defender for Cloud

Events (preview)

Deployment

Deployment slots

Deployment Center

Browse Start Swap Restart Delete Refresh Download publish profile Reset publish profile Share to mobile ...

Essentials

Resource group (move) MOL-AppService

Status Stopped

Location (move) East US

Subscription (move) Azure for Students

Subscription ID 62024cc7-aa5a-4087-844a-cade8eaf0574

Default domain pizza-launchess.azurewebsites.net

App Service Plan ASP-MOLAppService-9b87

Operating System Windows

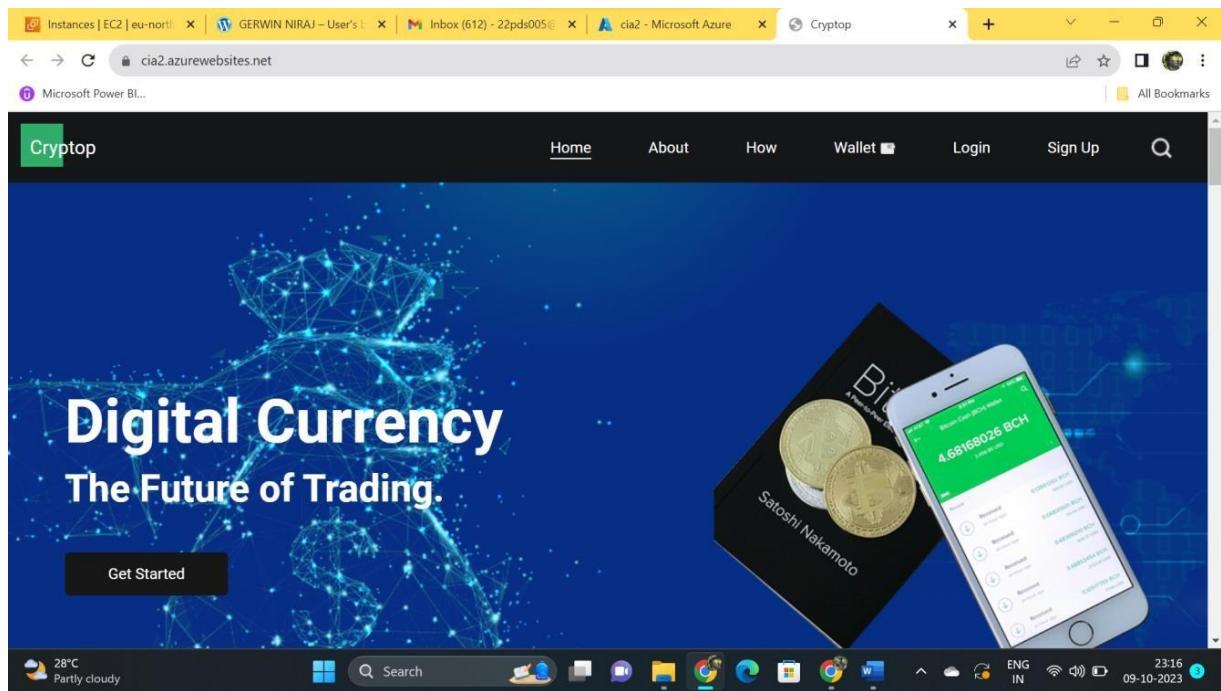
Health Check Not Configured

Git/Deployment username

Git clone url https://null@pizza-launchess.scm.azurewebsites.net/pizza-launchess.git

JSON View

## Output:



## Result:

Thus, our website is successfully deployed to Azure Web App, and it is accessible online via the provided URL.

**EX N0: 10**  
**DATE: 21/09/2023**

## CREATE AN AZURE CI/CD PIPELINE USING DEVOPS TOOLS

### Aim:

To Create an Azure Kubernetes Service (AKS) cluster, connect to it, and deploy a sample application to the cluster.

### Procedure:

#### Step 1: Creating a Kubernetes cluster

Create an AKS cluster using Azure portal.

Create resource → Containers → Azure Kubernetes Services → Create.



The screenshot shows the Microsoft Azure portal homepage. At the top, there is a search bar with the placeholder "Search resources, services, and docs (G+)" and a "Create" button. Below the search bar, there is a navigation bar with icons for "Azure services", "Resources", "Recent", and "Favorite". Under "Azure services", there is a grid of icons for various services, including "Create a resource" (which is highlighted with a yellow box), "Azure DevOps organizations", "App Services", "Education", "Cost Management ...", "Azure OpenAI", "Quickstart Center", "Virtual machines", "Storage accounts", and "More services".

<select containers>

Home >

### Create a resource

Get Started

Recently created

#### Categories

AI + Machine Learning

Analytics

Blockchain

Compute

Containers

Databases

<In Azure Kubernetes Services, Select Create>

## Create a resource

Get Started

Recently created

Popular Azure services See more in All services

Popular Marketplace products See more in Marketplace

**Categories**

- AI + Machine Learning
- Analytics
- Blockchain
- Compute
- Containers

**Azure Kubernetes Service (AKS)**  
Create | Docs | MS Learn

**Web App for Containers**  
Create | Docs | MS Learn

**Batch Service**  
Create | Docs | MS Learn

**NVIDIA GPU-Optimized VM**  
Create | Learn more

**Windows Server 2022 Core Datacenter Minimal OS**  
Create | Learn more

**Basic**  
Create | Learn more

## Step 2: Fill out the settings and configuration

Basics tab:

1. Give a resource group and a cluster name
2. Select region as US East
3. Cluster preset configuration as Dev/Test Nodes tab:

## Create Kubernetes cluster

Azure Kubernetes Service (AKS) manages your hosted Kubernetes environment, making it quick and easy to deploy and manage containerized applications without container orchestration expertise. It also eliminates the burden of ongoing operations and maintenance by provisioning, upgrading, and scaling resources on demand, without taking your applications offline.

[Learn more ↗](#)

### Project details

Select a subscription to manage deployed resources and costs. Use resource groups like folders to organize and manage all your resources.

Subscription \* ⓘ

Azure for Students

Resource group \* ⓘ

MOL-AppService

[Create new](#)

### Cluster details

Cluster preset configuration

Dev/Test

To quickly customize your Kubernetes cluster, choose one of the preset configurations above. You can modify these configurations at any time.  
[Learn more and compare presets](#)

## Create Kubernetes cluster

Cluster preset configuration

Dev/Test

To quickly customize your Kubernetes cluster, choose one of the preset configurations above. You can modify these configurations at any time. [Learn more and compare presets](#)

Kubernetes cluster name \* ⓘ

MyAKS

Region \* ⓘ

(US) East US

Availability zones ⓘ

Zones 1,2,3

AKS pricing tier ⓘ

Free

Kubernetes version \* ⓘ

1.26.6 (default)

Automatic upgrade ⓘ

Enabled with patch (recommended)

Choose between local accounts or Azure AD for authentication and Azure RBAC or Kubernetes RBAC for your authorization needs.

< Previous Next : Node pools > Review + create

<Networking tab>

### 1. Select Network configuration as kubenet

## Create Kubernetes cluster

Network configuration ⓘ

kubenet  
Best for smaller node pools. Each pod is assigned a logically different IP address from the subnet for simpler setup

Azure CNI  
Best for larger node pools. Each node and pod is assigned a unique IP for advanced configurations

Bring your own virtual network ⓘ

DNS name prefix \* ⓘ

MyAKS-dns

Network policy ⓘ

None  
Allow all ingress and egress traffic to the pods

Calico  
Open-source networking solution. Best for large-scale deployments with strict security requirements

Azure  
Native networking solution. Best for simpler deployments with basic security and networking requirements

< Previous Next : Integrations > Review + create

<Integrations tab>

Changes are not necessary. Proceed as it is.

## Create Kubernetes cluster ...

Basics Node pools Networking **Integrations** Advanced Tags Review + create

Connect your AKS cluster with additional services.

### Microsoft Defender for Cloud

Microsoft Defender for Cloud provides unified security management and advanced threat protection across hybrid cloud workloads. [Learn more](#)

Enable basic plan for free

### Azure Container Registry

Connect your cluster to an Azure Container Registry to enable seamless deployments from a private image registry.

[Learn more about Azure Container Registry](#)

Container registry

None

[Create new](#)

### Azure Monitor

In addition to the CPU and memory metrics included in AKS by default, you can enable Container Insights for more comprehensive data on the overall performance and health of your cluster. Billing is based on data ingestion and retention

< Previous

**Next : Advanced >**

Review + create

<Advanced tab>

Changes are not necessary. Proceed as it is.

## Create Kubernetes cluster ...

Basics Node pools Networking Integrations **Advanced** Tags Review + create

Enable secret store CSI driver

Infrastructure resource group

MC\_MOL-AppService\_MyAKS\_eastus

[Edit](#)

< Previous

**Next : Tags >**

Review + create

<Tags tab>

Changes are not necessary. Proceed as it is.

## Create Kubernetes cluster ...

Basics Node pools Networking Integrations Advanced Tags Review + create

Tags are name/value pairs that enable you to categorize resources and view consolidated billing by applying the same tag to multiple resources and resource groups. [Learn more about tags](#)

Note that if you create tags and then change resource settings on other tabs, your tags will be automatically updated.

Name ⓘ	Value ⓘ	Resource
<input type="text"/>	:	<input type="text"/> 2 selected ▾

< Previous Next : Review + create > **Review + create**

<Review and create tab>

Click create after validation is passed

Microsoft Azure Search resources, services, and docs (G+/-)

Home > Create a resource > Create Kubernetes cluster ...

Validation passed

Basics Node pools Networking Integrations Advanced Tags Review + create

**Basics**

Subscription	Azure for Students
Resource group	MOL-AppService
Region	East US
Kubernetes cluster name	MyAKS
Kubernetes version	1.26.6
Automatic upgrade	Patch

**Node pools**

< Previous Next > **Create** Download a template for automation

**Step 3:** Wait for the deployment to complete.

Home >

## microsoft.aks-20231004223512 | Overview

Deployment

Search | Delete | Cancel | Redeploy | Download | Refresh

- Overview
- Inputs
- Outputs
- Template

... Deployment is in progress

Deployment details

Resource	Type	Status	Operation details
InsightsActionGroupDep	Microsoft.Resources/d...	Created	<a href="#">Operation details</a>

Give feedback | Tell us about your experience with deployment

<after deployment, click on go to resources>

Home >

## microsoft.aks-20231004223512 | Overview

Deployment

Search | Delete | Cancel | Redeploy | Download | Refresh

- Overview
- Inputs
- Outputs
- Template

Your deployment is complete

Deployment name: microsoft.aks-202310... Start time: 10/4/2023, 10:40:58 PM  
Subscription: Azure for Students Correlation ID: aaa2b058-0e6f-4629-bb15-309840545c  
Resource group: MOL-AppService

Deployment details

Next steps

- Create a quick start application Recommended
- Create a Kubernetes deployment Recommended
- Integrate automatic deployments within your cluster Recommended
- Connect to cluster Recommended

[Go to resource](#) [Connect to cluster](#)

## Step 4: Connect to Cloud shell using PowerShell

Microsoft Azure | Search resources, services, and docs (G+)

Home > MyAKS | Kubernetes service

Search | Create | Connect | Start | Stop | Delete | Refresh | Open in mobile | Give feedback

- Overview
- Activity log
- Access control (IAM)
- Tags
- Diagnose and solve problems
- Microsoft Defender for Cloud

Essentials

Resource group	MOL-AppService	Kubernetes version	1.26.6
Status	Succeeded (Running)	API server address	myaks-dns-gd1w40js.hcp.eastus.azmk8s.io
Location	East US	Network type (plugin)	Kubenet
Subscription	Azure for Students	Node pools	1 node pool

JSON View

## 5: Connect to the AKS Cluster

Run the following command to connect to your AKS cluster

```
PS /home/charles> az aks get-credentials --resource-group MOL-AppService --name MyAKS
```

*Merged "MyAKS" as current context in /home/charles/.kube/config*

This command configures your local **kubectl** to use the AKS cluster.

2. Verify that you are connected to the AKS cluster (display information about the nodes in your AKS cluster).

```
PS /home/charles> kubectl get nodes
```

NAME	STATUS	ROLES	AGE	VERSION
aks-agentpool-11990967-vmss000000	Ready	agent	7m22s	v1.26.6
aks-agentpool-11990967-vmss000001	Ready	agent	7m28s	v1.26.6

## Step 6: Check for Existing Pods and Resources

1. Check for existing pods in the default namespace:

```
PS /home/charles> kubectl get pods
```

*No resources found in default namespace.*

2. Check for all resources (including services, deployments, and pods):

```
PS /home/charles> kubectl get all
```

<i>NAME</i>	<i>TYPE</i>	<i>CLUSTER-IP</i>	<i>EXTERNAL-IP</i>	<i>PORT(S)</i>	<i>AGE</i>
<i>service/kubernetes</i>	<i>ClusterIP</i>	<i>10.0.0.1</i>	<i>&lt;none&gt;</i>	<i>443/TCP</i>	<i>8m47s</i>

## 7: Create and Apply a Kubernetes Manifest

Create or edit a Kubernetes manifest file (in this case, **askapp.yml**):

```
PS /home/charles> vi askapp.yml
```

Paste the yaml contents and Finally add :wq! and click enter.

Contents are obtained from <https://github.com/Azure-Samples/azure-voting-appredis/blob/master/azure-vote-all-in-one-redis.yaml>

2. List the files in your current directory to ensure **askapp.yml** is present:

```
PS /home/charles> ls
```

```
2126_antique_cafe antique-cafe.zip askapp.yml azure-mol-samples-2nd-ed
clouddrive Microsoft
```

3. Apply the Kubernetes manifest to create the resources:

```
PS /home/charles> kubectl apply -f askapp.yml
```

*deployment.apps/azure-vote-back created service/azure-vote-back created deployment.apps/azure-vote-front created service/azure-vote-front created*

## 8: Check Deployments, Services, and Pods

Check the status of deployments:

```
PS /home/charles> kubectl get deployments
```

NAME	READY	UP-TO-DATE	AVAILABLE	AGE
azure-vote-back	1/1	1	1	30s
azure-vote-front	1/1	1	1	30s

2. Check the status of services (This should display the services and their external IP addresses).

```
PS /home/charles> kubectl get svc
```

NAME	TYPE	CLUSTER-IP	EXTERNAL-IP	PORT(S)	AGE
azure-vote-back	ClusterIP	10.0.80.231	<none>	6379/TCP	75s
azure-vote-front	LoadBalancer	10.0.233.213	52.149.236.217	80:32717/TCP	
kubernetes	ClusterIP	10.0.0.1	<none>	443/TCP	74s

3. Retrieve information about ReplicaSets

```
PS /home/charles> kubectl get rs
```

<i>NAME</i>	<i>DESIRED</i>	<i>CURRENT</i>	<i>READY</i>	<i>AGE</i>	<i>azure-vote-back-</i>
<i>66c88ccc8</i>	<i>1</i>	<i>1</i>	<i>1</i>	<i>2m18s</i>	<i>azure-vote-front-85dc674b97</i>
					<i>1</i>
					<i>2m18s</i>

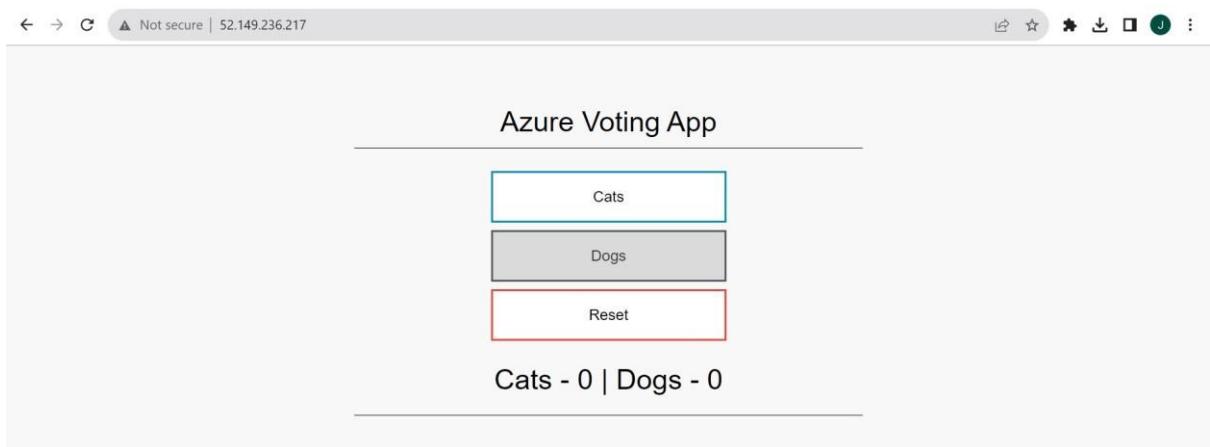
4. Verify that pods are running:

```
PS /home/charles> kubectl get pods
```

<i>NAME</i>	<i>READY</i>	<i>STATUS</i>	<i>RESTARTS</i>	<i>AGE</i>
<i>azure-vote-back-66c88ccc8-h572x</i>	<i>1/1</i>	<i>Running</i>	<i>0</i>	<i>2m33s</i>
<i>azure-vote-front-85dc674b97-c8zhh</i>	<i>1/1</i>	<i>Running</i>	<i>0</i>	<i>2m33</i>

**Step 9:** Paste the external IP address got in step 8.2

## Output:



## Result:

Thus, Created an Azure Kubernetes Service (AKS) cluster and deployed a sample application to the cluster.