

AWS Workshop: Highly available & scalable three-tier application deployment on AWS

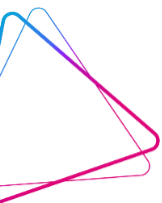
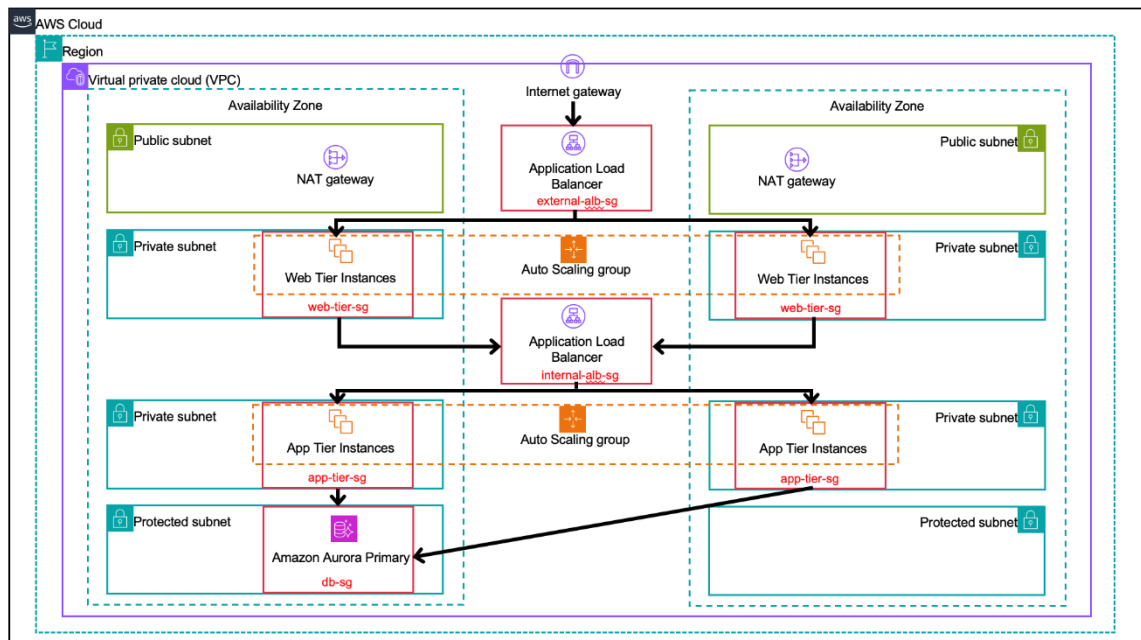


Table of Contents

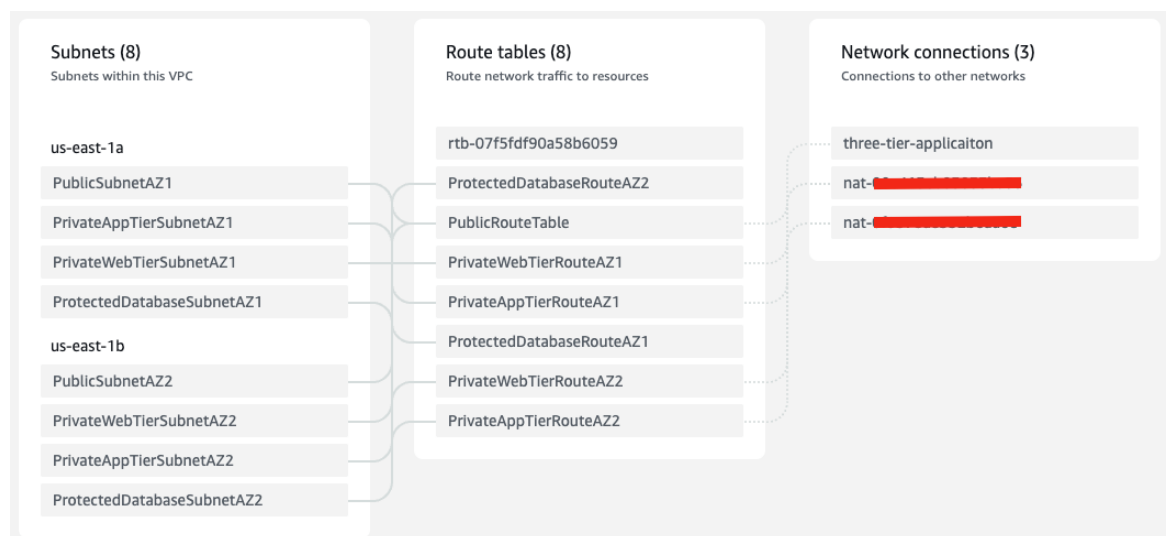
AWS Architecture	3
Step 1: Setting up networking & IAM roles as pre-requisite	4
Step 2: Creating a Web Server using EC2 Instance.....	6
Step 3: Creating App Server using EC2 Instance	9
Step 4: Connect Web Server with App Server.....	11
Step 5: Create Database	13
Step 6: Connect Database with App Tier.....	16
Step 7: Create Load Balancer and Autoscaling for Backend	18
Step 8: Create Load Balancer and Autoscaling for Frontend	24

AWS Architecture

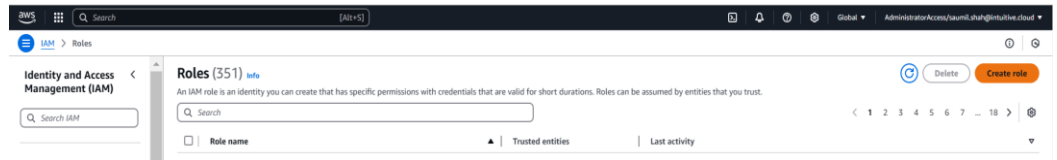


Step 1: Setting up networking & IAM roles as pre-requisite

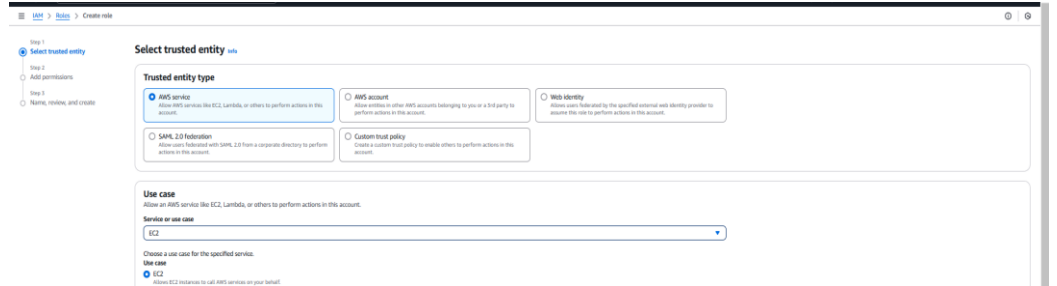
1. Go to CloudFormation service.
2. Click on **Create a stack With new resources**.
3. Download the '[CFT link](#)' to deploy the pre-requisite for the LAB
4. Choose **Upload a template** file and upload the file downloaded.
5. You can leave all parameters with default values.
6. Click on **Next** on **Step 2 (Specify stack details)** and **Step 3 (Configure stack options)**.
7. Finally, under review section click on **Submit**.
8. This stack creates the following resources:
 - **VPC with 2 public, 4 private, and 2 protected subnets.** Two public subnets would be connected to a common route table, having network connections to the **internet gateway**. Four private subnet will have 4 separate route tables, each route table will have network connects to to the **NAT gateway**. Protected subnets will have no path to the NAT gateways. Two private subnets will be used for frontend (web tier) logic and the other two private subnets for backend (app tier) logic. Protected subnets will have our RDS database.



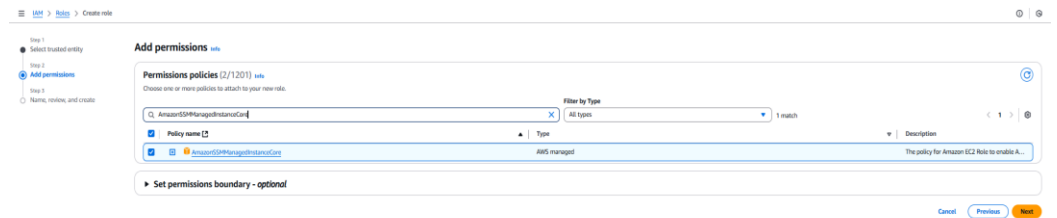
- Security groups namely
 1. WebTierSecurityGroup – To be used for all WebTier resources [EC2, ELB]
 2. AppTierSecurityGroup – To be used for AppTier resources- [EC2]
 3. DatabaseSecurityGroup- To be used for DatabaseTier [RDS]
9. Create an **IAM instance profile** for EC2
 1. Open AWS Console and go to the **IAM** service.
 2. Click on **Roles**, then click **Create role**.



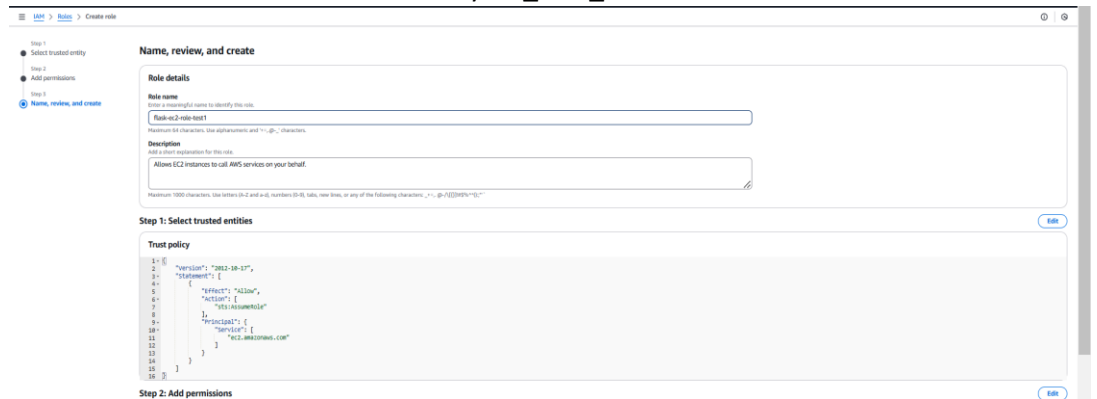
3. Select AWS service as the **trusted entity** and choose **EC2**. Click **Next**.



4. Search for and attach the **'AmazonS3FullAccess'** and **'AmazonSSMManagedInstanceCore'** policies. Click **Next**.



5. Enter the role name as **flask-ec2-role-<your_user_id>** and click **Create role**.



Step 2: Creating a Web Server using EC2 Instance

1. Open the Amazon EC2. From the EC2 console dashboard, in the Launch instance pane, choose **Launch instance**.



2. Under Name and tags, for name enter Webserver.

Launch an instance [Info](#)

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

Name and tags [Info](#)

Name

[Add additional tags](#)

3. Under Application and OS Images (Amazon Machine Image). Choose **Quick Start** and then choose the operating system (OS) for your instance. From Amazon Machine Image (AMI), select **Amazon Linux 2AMI**.

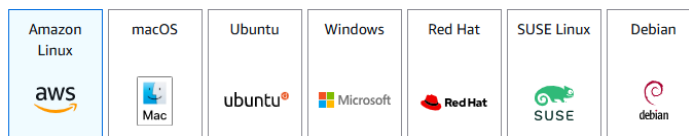
▼ Application and OS Images (Amazon Machine Image) [Info](#)

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

Recents

My AMIs

Quick Start



Amazon Machine Image (AMI)

Amazon Linux 2 AMI (HVM) - Kernel 5.10, SSD Volume Type
ami-02a53b0d62d37a757 (64-bit (x86)) / ami-08523976443f71beb (64-bit (Arm))
Virtualization: hvm ENA enabled: true Root device type: ebs

Free tier eligible ▼

4. Under Instance type, for Instance type, choose **t2.micro**.

▼ Instance type [Info](#) | [Get advice](#)

Instance type

t2.micro

Free tier eligible

Family: t2 1 vCPU 1 GiB Memory Current generation: true
On-Demand Windows base pricing: 0.0162 USD per Hour On-Demand Ubuntu Pro base pricing: 0.0134 USD per Hour
On-Demand SUSE base pricing: 0.0116 USD per Hour On-Demand RHEL base pricing: 0.026 USD per Hour
On-Demand Linux base pricing: 0.0116 USD per Hour

☐ All generations

[Compare instance types](#)

[Additional costs apply for AMIs with pre-installed software](#)

5. Under Key pair (login), **Proceed without a key pair**.

▼ 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*

Default value ▼

[Create new key pair](#)

6. Under Network settings, select **Edit**, under VPC choose **three-tier-application** VPC created in step 1. Under subnet select **PublicSubnetAZ1**. For Auto-assign public IP select **Enable**. Choose Select **existing security group** and choose **WebTierSecurityGroup**.

▼ **Network settings** [Info](#)

VPC - *required* | [Info](#)

vpc-092494c8b0675df25 (three-tier-application)
10.0.0.0/16

Subnet | [Info](#)

subnet-066d4848342aecb96 PublicSubnetAZ1
VPC: vpc-092494c8b0675df25 Owner: 185713903852 Availability Zone: us-east-1a
Zone type: Availability Zone IP addresses available: 247 CIDR: 10.0.0.0/24

Auto-assign public IP | [Info](#)

Enable

Additional charges apply when outside of free tier allowance

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

three-tier-application-WebTierSecurityGroup-tslFZDj8DRCS sg-08e57ac99e83ab2f2 X
VPC: vpc-092494c8b0675df25

Compare security group rules

► **Advanced network configuration**

7. Under Advanced Details section, for **IAM instance profile** select **flask-ec2-role-
<your_user_id>** created in step-1.

▼ **Advanced details** [Info](#)

Domain join directory | [Info](#)

Select

Create new directory

IAM instance profile | [Info](#)

flask-ec2-role-test1
arn:aws:iam::185713903852:instance-profile/flask-ec2-role-test1

Create new IAM profile

Hostname type | [Info](#)

IP name

DNS Hostname | [Info](#)

☒ Enable IP name IPv4 (A record) DNS requests

8. Scroll down to **User Data** and copy contents from this [Link](#).

User data - optional | [Info](#)

Upload a file with your user data or enter it in the field.

Choose file

```
#!/bin/bash
yum update -y
yum install -y python3 python3-pip git
git clone https://github.com/Saumil-Shah-ity/flask-three-tier.git /home/ec2-user/flaskapp
cd /home/ec2-user/flaskapp/frontend
sudo pip3 install flask flask-cors mysql-connector-python requests pymysql
sudo pip3 install --upgrade urllib3==1.26.16
sudo python3 /home/ec2-user/flaskapp/frontend/frontend.py
```

- Click on **Launch Instance**.
- Once the instance is up and running in **Healthy** state with **2/2 checks passed**. Select the instance and copy the public IP.

Instances (1/1) [Info](#)

Find Instance by attribute or tag (case-sensitive) Running

☒ Name ☐ Instance ID ☐ Instance state ☐ Instance type ☐ Status check ☐ Alarm

<input checked="" type="checkbox"/>	WebServer	i-0ad216cfd9f8caa23	Running	t2.micro	2/2 checks passed	View
-------------------------------------	-----------	---------------------	---------	----------	-------------------	----------------------

i-0ad216cfd9f8caa23 (WebServer)

[Details](#) | [Status and alarms](#) | [Monitoring](#) | [Security](#) | [Networking](#) | [Storage](#) | [Tags](#)

▼ **Instance summary** [Info](#)

Instance ID
[i-0ad216cfd9f8caa23](#)

IPv6 address
-

Hostname type
IP name: ip-10-0-0-243.ec2.internal

Answer private resource DNS name
-

Auto-assigned IP address
[34.206.64.23](#) (Public IP)

Public IPv4 address
[34.206.64.23](#) | [open address](#)

Instance state
[Running](#)

Private IP DNS name (IPv4 only)
[ip-10-0-0-243.ec2.internal](#)

Instance type
t2.micro

VPC ID
[vpc-092494c8b0675df25](#) (three-tier-application)

Private IPv4 addresses
[10.0.0.243](#)

Public IPv4 DNS
[ec2-34-206-64-23.compute-1.amazonaws.com](#) | [open address](#)

Elastic IP addresses
-

AWS Compute Optimizer finding
No recommendations available for this instance.

- Open your browser and search for `http://<public_id>:80`. The screen below should appear.

To-Do List

Frontend Dummy Task 1

☒

Frontend Dummy Task 2

☒

Step 3: Creating App Server using EC2 Instance

1. Open the Amazon EC2. From the EC2 console dashboard, in the Launch instance pane, choose **Launch instance**.



2. Under Name and tags, for name enter **Appserver**.

A screenshot of the 'Name and tags' section in the EC2 console. It features a 'Name' label and a text input field containing the text 'Appserver'. To the right of the input field is a blue link that says 'Add additional tags'.

3. Under Application and OS Images (Amazon Machine Image). Choose **Quick Start** and then choose the operating system (OS) for your instance. From Amazon Machine Image (AMI), select **Amazon Linux 2AMI**.

A screenshot of the 'Application and OS Images (Amazon Machine Image)' section. It includes a search bar with the placeholder text 'Search our full catalog including 1000s of application and OS images'. Below the search bar are tabs for 'Recents', 'My AMIs', and 'Quick Start'. The 'Quick Start' tab is active, showing a grid of AMI cards for Amazon Linux, macOS, Ubuntu, Windows, Red Hat, SUSE Linux, and Debian. The 'Amazon Linux' card is highlighted with a blue border. Below the grid, the details for the selected 'Amazon Linux 2 AMI (HVM) - Kernel 5.10, SSD Volume Type' are shown, including its ID and pricing information. A 'Free tier eligible' badge is visible on the right.

4. Under Instance type, for Instance type, choose **t2.micro**.

A screenshot of the 'Instance type' section. It shows the 't2.micro' instance type selected. Below the name, it lists specifications: 'Family: t2', '1 vCPU', '1 GiB Memory', and 'Current generation: true'. Pricing information for various operating systems is also displayed. A 'Free tier eligible' badge is present. To the right, there is a radio button for 'All generations' and a link to 'Compare instance types'. At the bottom, a note states 'Additional costs apply for AMIs with pre-installed software'.

5. Under Key pair (login), **Proceed without a key pair**.

A screenshot of the 'Key pair (login)' section. It includes an informational message: '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.' Below this, there is a label 'Key pair name - required' and a text input field containing the text 'Proceed without a key pair (Not recommended)'. To the right of the input field is a 'Default value' dropdown menu and a blue link to 'Create new key pair'.

- Under Network settings, select **Edit**, under VPC choose **three-tier-application** VPC created in step 1. Under subnet select **PrivateAppTierSubnetAZ1**. Choose **Select existing security group** and choose **AppTierSecurityGroup**.

▼ Network settings [Info](#)

VPC - required [Info](#)

vpc-092494c8b0675df25 (three-tier-application)
10.0.0.0/16

Subnet [Info](#)

subnet-0ab04e2f322f47dc1 PrivateAppTierSubnetAZ1
VPC: vpc-092494c8b0675df25 Owner: 185713903852 Availability Zone: us-east-1a
Zone type: Availability Zone IP addresses available: 249 CIDR: 10.0.4.0/24

Auto-assign public IP [Info](#)

Disable

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

three-tier-application-AppTierSecurityGroup-uyPkV2h43C29 sg-06be5a20b60d8da1e X
VPC: vpc-092494c8b0675df25

[Compare security group rules](#)

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

[▶ Advanced network configuration](#)

- Under Advanced Details section, for **IAM instance profile** select **flask-ec2-role-<your_user_id>** created in step-1.

▼ Advanced details [Info](#)

Domain join directory [Info](#)

Select

[Create new directory](#)

IAM instance profile [Info](#)

flask-ec2-role-test1
arn:aws:iam::185713903852:instance-profile/flask-ec2-role-test1

[Create new IAM profile](#)

Hostname type [Info](#)

IP name

DNS Hostname [Info](#)

☒ Enable IP name IPv4 (A record) DNS requests

- Scroll down to **User Data** and copy contents from this [Link](#).

User data - optional [Info](#)
Upload a file with your user data or enter it in the field.

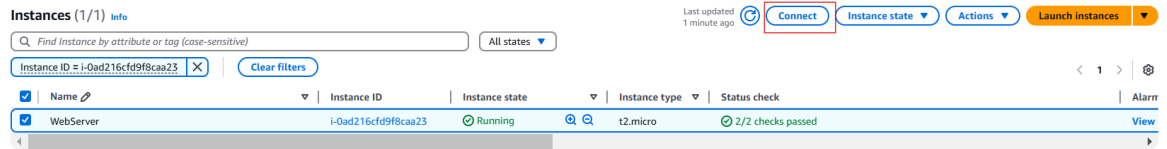
[Choose file](#)

```
#!/bin/bash
yum update -y
yum install -y python3 python3-pip git
git clone https://github.com/Saumil-Shah-ity/flask-three-tier.git /home/ec2-user/flaskapp
cd /home/ec2-user/flaskapp/backend
sudo pip3 install flask flask-cors mysql-connector-python requests pymysql
sudo pip3 install --upgrade urllib3==1.26.16
sudo python3 /home/ec2-user/flaskapp/backend/backend.py
```

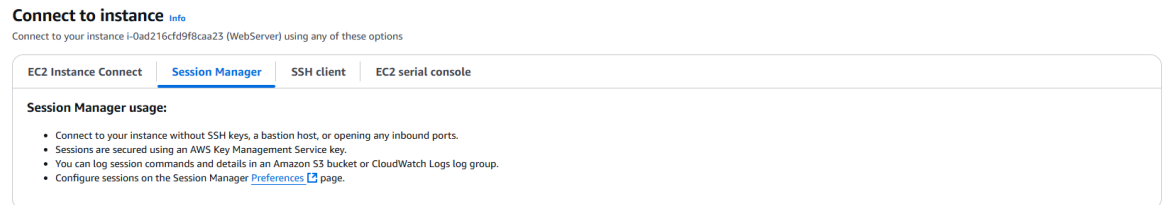
- Click on **Launch Instance**.
- Once the instance is up and running in **Healthy** state with **2/2 checks passed**.

Step 4: Connect Web Server with App Server

1. Navigate to EC2 Instance and choose **Webserver**. Select **Connect**.



2. Select **Session Manager** tab and click on **Connect**.



3. Execute following steps:

- a. `sudo su ec2-user`
- b. `cd ~`
- c. `cd flaskapp/frontend/`

```
sh-4.2$ sudo su ec2-user
[ec2-user@ip-10-0-0-243 bin]$ cd ~
[ec2-user@ip-10-0-0-243 ~]$ cd flaskapp/frontend/
[ec2-user@ip-10-0-0-243 frontend]$ sudo vim frontend.py
[ec2-user@ip-10-0-0-243 frontend]$
```

- d. `sudo vim frontend.py`
- e. Press the **i** key to enter insert mode and replace the **BACKEND_API** variable with the **Private IP** address of the **Appserver**.

```
from flask import Flask, render_template, request, redirect, url_for
import requests

app = Flask(__name__)

# Backend API URL - Replace with your actual backend DNS or IP
BACKEND_API = "http://10.0.4.40:5000" # Update this with Load Balancer later

# Dummy data for fallback
tasks = [{"id": 1, "name": "Frontend Dummy Task 1"}, {"id": 2, "name": "Frontend Dummy Task 2"}]

@app.route("/", methods=["GET", "POST"])
def index():
    global tasks

    if request.method == "POST":
        task_name = request.form.get("task_name")
        try:
            requests.post(f"{BACKEND_API}/add", json={"name": task_name})
        except requests.exceptions.RequestException:
            new_id = max([task["id"] for task in tasks]) + 1 if tasks else 1
            tasks.append({"id": new_id, "name": task_name})

        return redirect(url_for("index"))

    try:
        response = requests.get(f"{BACKEND_API}/tasks")
        response.raise_for_status()
        tasks = response.json()
    except requests.exceptions.RequestException as e:
        print(f"Backend API failed: {e}")

    return render_template("index.html", tasks=tasks)

@app.route("/delete/<int:task_id>")
def delete_task(task_id):
    global tasks
    try:
        requests.delete(f"{BACKEND_API}/delete/{task_id}")
    except requests.exceptions.RequestException:
        pass

:wq!
```

- f. Press the **Esc** key, type `:wq!`, and hit **Enter** to save and exit.
- g. `sudo fuser -k 80/tcp`

- h. `sudo nohup python3 frontend.py --port 80 &`


```
[ec2-user@ip-10-0-4-40 backend]$ sudo nohup python3 backend.py --port 5000 &
[1] 1169
[ec2-user@ip-10-0-4-40 backend]$ nohup: ignoring input and appending output to 'nohup.out'
[ec2-user@ip-10-0-4-40 backend]$
```

- i. Hit **Enter** twice.


10. Open your browser and search for `http://<public_id_of_webserver>:80`. The screen below should appear.

To-Do List

Dummy Task 1



Dummy Task 2



Step 5: Create Database

1. Navigate to the RDS dashboard in the AWS console and click on **Subnet groups** on the left-hand side. Click **Create DB subnet group**.
2. Name the subnet group as **three-tier-subnet-group** and choose VPC named **three-tier-application**

Subnet group details
Name
You won't be able to modify the name after your subnet group has been created.

Must contain from 1 to 255 characters. Alphanumeric characters, spaces, hyphens, underscores, and periods are allowed.
Description

VPC
Choose a VPC identifier that corresponds to the subnets you want to use for your DB subnet group. You won't be able to choose a different VPC identifier after your subnet group has been created.

three-tier-application (vpc-092494c8b0675df25)
8 Subnets, 2 Availability Zones

3. Under **Availability Zones** select **us-east-1a** and **us-east-1b**. For subnet choose **ProtectedDatabaseSubnetAZ1** and **ProtectedDatabaseSubnetAZ2**. Click **Create**.

Add subnets
Availability Zones
Choose the Availability Zones that include the subnets you want to add.

Choose an availability zone

us-east-1a X us-east-1b X

Subnets
Choose the subnets that you want to add. The list includes the subnets in the selected Availability Zones.

Select subnets

ProtectedDatabaseSubnetAZ1
Subnet ID: subnet-0ca4ef7788334d98c CIDR: 10.0.6.0/24

ProtectedDatabaseSubnetAZ2
Subnet ID: subnet-064365bf84efa6833 CIDR: 10.0.7.0/24

For Multi-AZ DB clusters, you must select 3 subnets in 3 different Availability Zones.

Subnets selected (2)

Availability zone	Subnet name	Subnet ID	CIDR block
us-east-1a	ProtectedDatabaseSubnetAZ1	subnet-0ca4ef7788334d98c	10.0.6.0/24
us-east-1b	ProtectedDatabaseSubnetAZ2	subnet-064365bf84efa6833	10.0.7.0/24

4. Sign in to the AWS Management Console and open the Amazon RDS console.
5. In the navigation pane, choose **Databases**.
6. Choose **Create database**.

Databases (1)

Group resources Modify Actions Restore from S3 Create database

Filter by databases

DB identifier	Status	Role	Engine	Region ...	Size	Recommendations	CPU	Current activity	Mai
database-1	Available	Instance	MySQL Co...	us-east-1a	db.t3.micro		3.06%	0 Connections	non

7. Choose **Standard create**.

Choose a database creation method









Standard create
You set all of the configuration options, including ones for availability, security, backups, and maintenance.

Easy create
Use recommended best-practice configurations. Some configuration options can be changed after the database is created.

8. For Engine type, choose **MySQL**.

Engine options

Engine type [Info](#)

<input type="radio"/> Aurora (MySQL Compatible) 	<input type="radio"/> Aurora (PostgreSQL Compatible) 
<input checked="" type="radio"/> MySQL 	<input type="radio"/> PostgreSQL 
<input type="radio"/> MariaDB 	<input type="radio"/> Oracle 
<input type="radio"/> Microsoft SQL Server 	<input type="radio"/> IBM Db2 

9. In Templates, choose the template that matches your use case. Choose **Free Tier**.

Templates

Choose a sample template to meet your use case.

<input type="radio"/> Production Use defaults for high availability and fast, consistent performance.	<input type="radio"/> Dev/Test This instance is intended for development use outside of a production environment.	<input checked="" type="radio"/> Free tier Use RDS Free Tier to develop new applications, test existing applications, or gain hands-on experience with Amazon RDS. Info
---	---	---

10. In the Settings section **name** the database as **three-tier-db** and open Credential management and select **Self managed**. Enter the same password in Master password and Confirm password.

Settings

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

The DB instance identifier is case-insensitive, but is stored as all lowercase (as in "mydbinstance"). Constraints: 1 to 63 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.

1 to 16 alphanumeric characters. The first character must be a letter.

Credentials management
You can use AWS Secrets Manager or manage your master user credentials.

☐ Managed in AWS Secrets Manager - most secure
RDS generates a password for you and manages it throughout its lifecycle using AWS Secrets Manager.

☒ **Self managed**
Create your own password or have RDS create a password that you manage.

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

Master password [Info](#)

Password strength [Info](#) Neutral
Minimum constraints: At least 8 printable ASCII characters. Can't contain any of the following symbols: / ' * @

Confirm master password [Info](#)

8. Under **Instance configuration** section choose **Burstable classes** and select **db.t3.micro** instance type.

Instance configuration

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

DB instance class [Info](#)

▼ **Hide filters**

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

☒ Include previous generation classes

☐ Standard classes (includes m classes)

☐ Memory optimized classes (includes r and x classes)

☒ **Burstable classes (includes t classes)**

2 vCPUs 1 GiB RAM Network: Up to 2,085 Mbps

9. Under **Connectivity** section, under VPC select **three-tier-application** and for subnet group select **three-tier-subnet group**.

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.

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

three-tier-application (vpc-092494c8b0675df25)
8 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.

private-three-tier
2 Subnets, 2 Availability Zones

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.

10. For the **VPC Security Group**, select **"Existing"** and choose **three-tier-application-DatabaseSecurityGroup**.

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

Existing VPC security groups
Choose one or more options

three-tier-application-DatabaseSecurityGroup-Sp5PSnckHFQ ✕

Availability Zone [Info](#)
No preference

RDS Proxy
RDS Proxy is a fully managed, highly available database proxy that improves application scalability, resiliency, and security.

☐ Create an RDS Proxy [Info](#)
RDS automatically creates an IAM role and a Secrets Manager secret for the proxy. RDS Proxy has additional costs. For more information, see [Amazon RDS Proxy pricing](#).

Certificate authority - optional [Info](#)
Using a server certificate provides an extra layer of security by validating that the connection is being made to an Amazon database. It does so by checking the server certificate that is automatically installed on all databases that you provision.

rds-ca-rsa2048-g1 (default)
Expiry: May 26, 2061

If you don't select a certificate authority, RDS chooses one for you.

► **Additional configuration**

11. Scroll and create database.

► **Additional configuration**
Database options, encryption turned on, backup turned on, backtracking turned off, maintenance, CloudWatch Logs, delete protection turned off.

Estimated monthly costs
The Amazon RDS Free Tier is available to you for 12 months. Each calendar month, the free tier will allow you to use the Amazon RDS resources listed below for free:

- 750 hrs of Amazon RDS in a Single-AZ db.t2.micro, db.t3.micro or db.t4g.micro Instance.
- 20 GB of General Purpose Storage (SSD).
- 20 GB for automated backup storage and any user-initiated DB Snapshots.

[Learn more about AWS Free Tier.](#)

When your free usage expires or if your application use exceeds the free usage tiers, you simply pay standard, pay-as-you-go service rates as described in the [Amazon RDS Pricing page](#).

ⓘ You are responsible for ensuring that you have all of the necessary rights for any third-party products or services that you use with AWS services.

[Cancel](#) [Create database](#)

12. Once the database is in **running** state. Select the database and navigate to **Endpoint & port** copy the **endpoint**.

Connectivity & security | Monitoring | Logs & events | Configuration | Zero-ETL integrations | Maintenance & backups | Data migrations - new | Tags | Recommendations

Connectivity & security

Endpoint & port

Endpoint
three-tier-db.cvmxj8kxlm.us-east-1.rds.amazonaws.com

Port
3306

Networking

Availability Zone
us-east-1b

VPC
three-tier-application (vpc-092494c8b0675df25)

Subnet group
private-three-tier

Subnets
subnet-064365bf84efa6833
subnet-0ca4ef7788334d98c

Security

VPC security groups
three-tier-application-DatabaseSecurityGroup-Sp5PSnckHFQ (sg-055512455bee6bc38)
Active

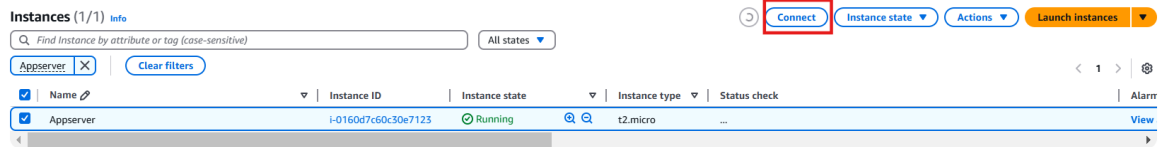
Publicly accessible
No

Certificate authority [Info](#)
rds-ca-rsa2048-g1

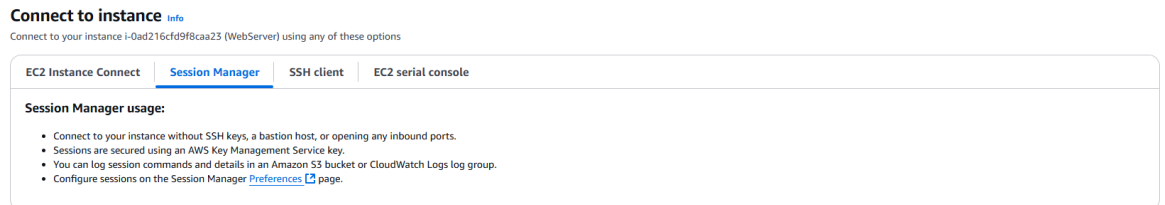
Certificate authority date
May 26, 2061, 05:04 (UTC+05:30)

Step 6: Connect Database with App Tier

1. Navigate to EC2 Instance and choose **Appserver**. Select **Connect**.



2. Select **Session Manager** tab and click on **Connect**.



3. Perform the following steps to create a table.
 - a. For connecting to the database execute following commands:

- i. `sudo su ec2-user`
- ii. `sudo yum install mysql -y`

```
[ec2-user@ip-10-0-4-40 bin]$ sudo yum install mysql -y
Loaded plugins: extra_suggestions, langpacks, priorities, update-motd
amzn2-core                               | 3.4 kB  00:00:00
Resolving Dependencies
--> Running transaction check
--> Package mariadb.x86_64 1:5.5.68-1.amzn2.0.1 will be installed
--> Finished Dependency Resolution

Dependencies Resolved

Package               Arch          Version              Repository           Size
--
Installing:
mariadb               x86_64        1:5.5.68-1.amzn2.0.1 amzn2-core           8.8 M

Transaction Summary
--
Install 1 Package
--
Total download size: 8.8 M
Installed size: 49 M
Downloading packages:
mariadb-5.5.68-1.amzn2.0.1.x86_64.rpm | 8.8 MB  00:00:00
Running transaction check
Running transaction test
Transaction test succeeded
Running transaction
  Installing : 1:mariadb-5.5.68-1.amzn2.0.1.x86_64
  Verifying  : 1:mariadb-5.5.68-1.amzn2.0.1.x86_64
Installed:
mariadb.x86_64 1:5.5.68-1.amzn2.0.1
Complete!
```

- iii. `mysql -h <database-endpoint> -u admin -p`

```
[ec2-user@ip-10-0-4-40 bin]$ mysql -h three-tier-db.cvmgxj8kximi.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 30
Server version: 8.0.40 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)]>
```

- iv. Once the database is connected, execute the SQL queries from the provided [Link](#).

```
MySQL [(none)]> CREATE DATABASE todo;
Query OK, 1 row affected (0.01 sec)

MySQL [(none)]> USE todo;
Database changed
MySQL [todo]> CREATE TABLE tasks (
  ->     id INT AUTO INCREMENT PRIMARY KEY,
  ->     name VARCHAR(255) NOT NULL
  -> );
Query OK, 0 rows affected (0.07 sec)

MySQL [todo]> exit
Bye
[ec2-user@ip-10-0-4-40 bin]$
```


4. Perform the following steps to connect to the database to app tier.
 - a. `cd /home/ec2-user/flaskapp/backend/`
 - b. `sudo vim backend.py`
 - c. Press the **i** key to enter insert mode and replace the database config as:
 - i. Host -> **database endpoint**
 - ii. User -> **admin**
 - iii. Password -> **<your_database_password>**
 - iv. Database -> **todo**

```
from flask import Flask, jsonify, request
import pymysql

app = Flask(__name__)

# ✅ Replace with your RDS MySQL details
DB_CONFIG = {
    "host": "three-tier-db.cvmgxj8kximi.us-east-1.rds.amazonaws.com",
    "user": "admin",
    "password": "<type-your-password-here>",
    "database": "todo"
}

# ✅ Try to connect to DB
```

- d. Press the **Esc** key, type **:wq!**, and hit **Enter** to save and exit.
- e. `sudo fuser -k 5000/tcp`
- f. `sudo nohup python3 backend.py --port 5000 &`

```
[ec2-user@ip-10-0-4-40 backend]$ sudo nohup python3 backend.py --port 5000 &
[1] 1169
[ec2-user@ip-10-0-4-40 backend]$ nohup: ignoring input and appending output to 'nohup.out'
[ec2-user@ip-10-0-4-40 backend]$
```

- g. Hit **Enter** twice.

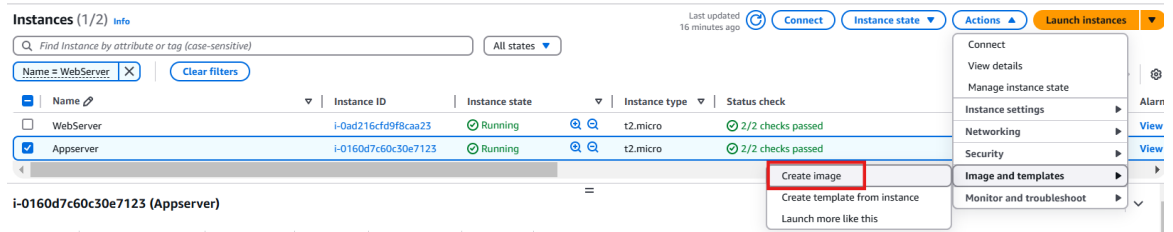
5. Open your browser and search for `http://<public_id_of_webserver>:80`. The screen below should appear.

To-Do List

Add Task

Step 7: Create Load Balancer and Autoscaling for Backend

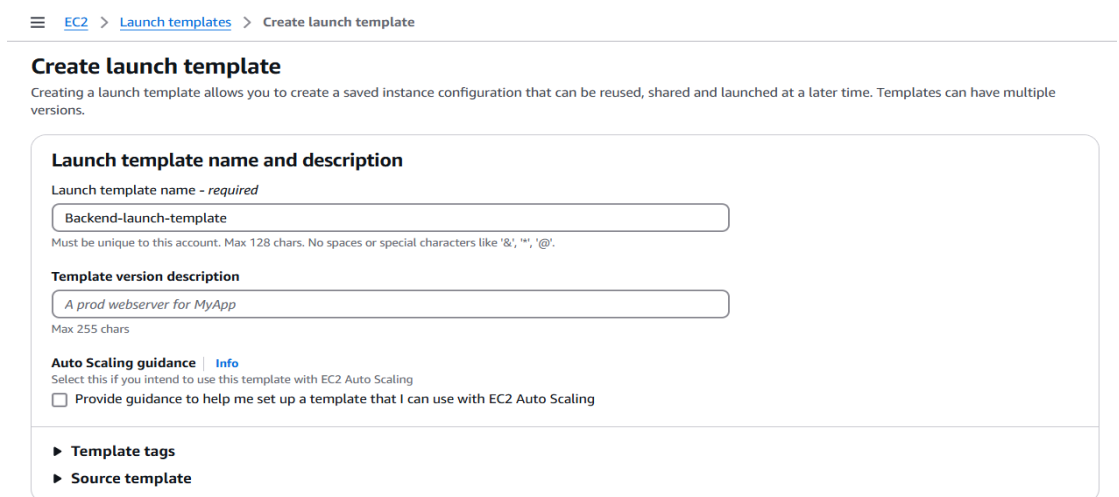
1. Create Image for Autoscaling group, Click on actions & select **'Image & template'** create an image using the instance. Provide the **Image name** as **Backend-AMI** and **Create Image**. Make sure its status appears **Available**.



The screenshot shows the AWS Management Console 'Instances' page. A table lists two instances: 'WebServer' and 'Appserver'. The 'Appserver' instance (ID: i-0160d7c60c30e7123) is selected. The 'Actions' menu is open, and the 'Create image' option is highlighted. The 'Create image' option is also visible in the 'Image and templates' section of the left sidebar.

Name	Instance ID	Instance state	Instance type	Status check
WebServer	i-0ad216cf898caa23	Running	t2.micro	2/2 checks passed
Appserver	i-0160d7c60c30e7123	Running	t2.micro	2/2 checks passed

2. Navigate to EC2 Console. Click Launch Templates in the left sidebar and click on create a **New Launch Template**. Enter Launch template name **backend-launch-template**.



The screenshot shows the 'Create launch template' page in the AWS Management Console. The 'Launch template name' is 'Backend-launch-template' and the 'Template version description' is 'A prod webserver for MyApp'. The 'Auto Scaling guidance' section is expanded, showing the 'Template tags' and 'Source template' options.

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

Launch template name - required

Backend-launch-template

Must be unique to this account. Max 128 chars. No spaces or special characters like '&', '@', '!'.

Template version description

A prod webserver for MyApp

Max 255 chars

Auto Scaling guidance | Info

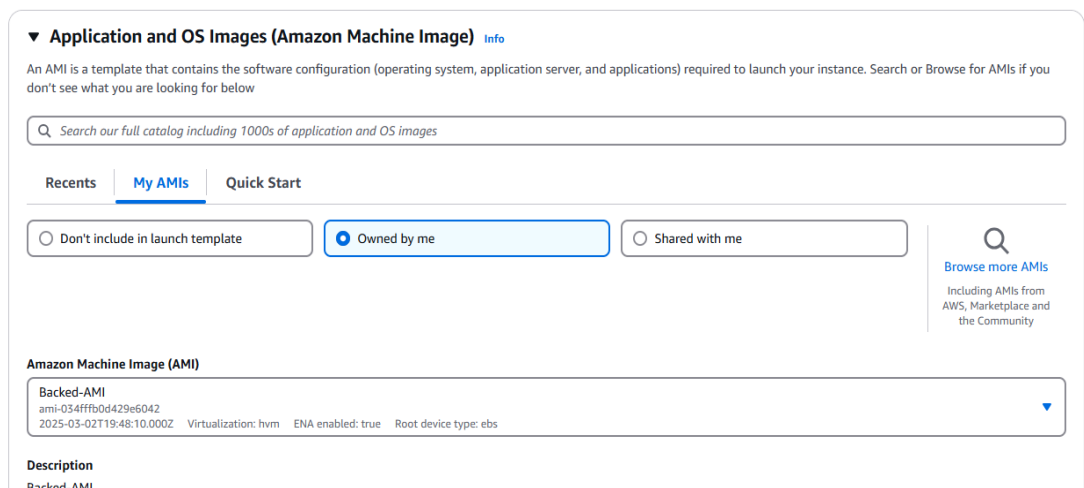
Select this if you intend to use this template with EC2 Auto Scaling

☐ Provide guidance to help me set up a template that I can use with EC2 Auto Scaling

► Template tags

► Source template

3. Under **Amazon Machine Image (AMI)**, select **'Backend-AMI'** created above



The screenshot shows the 'Application and OS Images (Amazon Machine Image)' page in the AWS Management Console. The 'Backend-AMI' is selected under the 'Owned by me' filter. The 'Description' section shows the AMI ID 'ami-034f7b0d429e6042' and the creation date '2025-03-02T19:48:10.000Z'.

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

Recents | **My AMIs** | Quick Start

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

Amazon Machine Image (AMI)

Backend-AMI
ami-034f7b0d429e6042
2025-03-02T19:48:10.000Z Virtualization: hvm ENA enabled: true Root device type: ebs

Description

Backend-AMI

4. Choose an **Instance Type** (e.g., 't2.micro')

▼ Instance type Info | Get advice

Advanced

Instance type

t2.micro

Family: t2 1 vCPU 1 GiB Memory Current generation: true On-Demand Windows base pricing: 0.0162 USD per Hour Free tier eligible

On-Demand Ubuntu Pro base pricing: 0.0134 USD per Hour On-Demand SUSE base pricing: 0.0116 USD per Hour

On-Demand RHEL base pricing: 0.026 USD per Hour On-Demand Linux base pricing: 0.0116 USD per Hour

All generations

Compare instance types

Additional costs apply for AMIs with pre-installed software

5. Under Key pair (login), **Proceed without a key pair**.

▼ 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

Proceed without a key pair (Not recommended)

Default value ▼

Create new key pair

6. Keep default network settings as '**don't include in launch template**'. For security group select **three-tier-application-AppTierSecurityGroup**.

▼ Network settings Info

Subnet | Info

Don't include in launch template

Create new subnet

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

Firewall (security groups) | Info

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

Select existing security group

Create security group

Security groups | Info

Select security groups

three-tier-application-AppTierSecurityGroup-uyPkV2h43C29 sg-06be5a20b60d8da1e X

VPC: vpc-092494c8b0675df25

Compare security group rules

► Advanced network configuration

7. Under Advanced Details section, for **IAM instance profile** select **flask-ec2-role-<your_user_id>** created in step-1.

▼ Advanced details Info

Domain join directory | Info

Select

Create new directory

IAM instance profile | Info

flask-ec2-role-test1

arn:aws:iam::185713903852:instance-profile/flask-ec2-role-test1

Create new IAM profile

Hostname type | Info

IP name

DNS Hostname | Info

☒ Enable IP name IPv4 (A record) DNS requests

11. Scroll down to **User Data** and copy contents from this [Link](#).

User data - optional | [Info](#)

Upload a file with your user data or enter it in the field.

[Choose file](#)

```
#!/bin/bash
yum update -y
yum install -y python3 python3-pip git
git clone https://github.com/Saamil-Shah-ity/flask-three-tier.git /home/ec2-user/flaskapp
cd /home/ec2-user/flaskapp/backend
sudo pip3 install flask flask-cors mysql-connector-python requests pymysql
sudo pip3 install --upgrade urllib3==1.26.16
sudo python3 /home/ec2-user/flaskapp/backend/backend.py
```

12. Click on **Create Launch Template**.
13. Open the **AWS Auto Scaling Console**. Click **Create an Auto Scaling group**
14. Enter ASG Name **backend-asg**. Under **Launch Template**, select the one you created **backend-launch-template**

- Step 1: **Choose launch template or configuration**
- Step 2: Choose instance launch options
- Step 3 - optional: Integrate with other services
- Step 4 - optional: Configure group size and scaling
- Step 5 - optional: Add notifications
- Step 6 - optional: Add tags
- Step 7: Review

Choose launch template or configuration [Info](#)

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

Name

Auto Scaling group name

Enter a name to identify the group.

Backend-ASG

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

Launch template [Info](#)

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.

Backend-launch-template

[Create a launch template](#)

Version

Default (1)

[Create a launch template version](#)

Description

-

AMI ID

ami-0a6622b53b5f6cd0e

Key pair name

-

Launch template

[Backend-launch-template](#)

lt-09f40c0cd7aed1e55

Security groups

-

Security group IDs

-

Instance type

t2.micro

Request Spot Instances

No

Additional details

Storage (volumes)

-

Date created

Sun Mar 02 2025 22:58:17 GMT+05:30 (India Standard Time)

[Cancel](#)

[Next](#)

15. Choose the Network. Select the **‘three-tier-application’** VPC. Choose two private **PrivateAppTierSubnetAZ1** and **PrivateAppTierSubnetAZ2**.

Step 1: Choose launch template or configuration
Step 2: **Choose instance launch options**
Step 3 - optional: Integrate with other services
Step 4 - optional: Configure group size and scaling
Step 5 - optional: Add notifications
Step 6 - optional: Add tags
Step 7: Review

Choose instance launch options [info](#)

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

Instance type requirements [info](#)

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	Version	Description
Backend-launch-template lt-09f40c0c7aed1e55	Default	-

Instance type
t2.micro

[Override launch template](#)

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-092494c8b0675df25 (three-tier-application)
10.0.0.0/16
[Create a VPC](#)

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
us-east-1b | subnet-0a904b05d31027a0f (PrivateAppTierSubnetAZ2)
10.0.5.0/24
us-east-1a | subnet-0ab04c2f322f47dc1 (PrivateAppTierSubnetAZ1)
10.0.4.0/24
[Create a subnet](#)

Availability Zone distribution - new
Auto Scaling automatically balances instances across Availability Zones. If launch failures occur in a zone, select a strategy.

☒ **Balanced best effort**
If launches fail in one Availability Zone, Auto Scaling will attempt to launch in another healthy Availability Zone.

☐ **Balanced only**
If launches fail in one Availability Zone, Auto Scaling will continue to attempt to launch in the unhealthy Availability Zone to preserve balanced distribution.

[Cancel](#) [Skip to review](#) [Previous](#) [Next](#)

16. Add Health Check & Load Balancer. Create and attach a **new load-balancer**

[EC2](#) > [Auto Scaling groups](#) > Create Auto Scaling group

Step 1: Choose launch template or configuration
Step 2: Choose instance launch options
Step 3 - optional: Integrate with other services
Step 4 - optional: Configure group size and scaling
Step 5 - optional: Add notifications
Step 6 - optional: Add tags
Step 7: Review

Integrate with other services - optional [info](#)

Use a load balancer to distribute network traffic across multiple servers. Enable service-to-service communications with VPC Lattice. Shift resources away from impaired Availability Zones with zonal shift. You can also customize health check replacements and monitoring.

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 routed 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 new load balancer to attach to your Auto Scaling group.

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

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

☒ **Application Load Balancer**
HTTP, HTTPS

☐ Network Load Balancer
TCP, UDP, TLS

Load balancer name
Name cannot be changed after the load balancer is created.
Backend-ASG-1

17. Select **‘Internal load-balancer’** & **‘private’** subnets, under port type **‘5000’**. Under default routing select **‘Create a target group’**

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

☒ **Internal**

☐ Internet-facing

Network mapping
Your new load balancer will be created using the same VPC and Availability Zone selections as your Auto Scaling group. You can select different subnets and add subnets from additional Availability Zones.

VPC
vpc-092494c8b0675df25 [three-tier-application](#)

Availability Zones and subnets
You must select a single subnet for each Availability Zone enabled. Only public subnets are available for selection to support DNS resolution.

☒ us-east-1b
subnet-0a904b05d31027a0f

☒ us-east-1a
subnet-0ab04c2f322f47dc1

Listeners and routing
If you require secure listeners, or multiple listeners, you can configure them from the [Load Balancing console](#) after your load balancer is created.

Protocol
HTTP

Port
5000

Default routing (forward to)
[Create a target group](#)

New target group name
An instance target group with default settings will be created.
Backend-ASG-1

Tags - optional
Consider adding tags to your load balancer. Tags enable you to categorize your AWS resources so you can more easily manage them.

[Add tag](#)
50 remaining

18. Update the scaling configuration by setting the '**max desired capacity**' to '2' and selecting the '**target tracking scaling policy**' under 'Automatic scaling' options:

Scaling [Info](#)
You can resize your Auto Scaling group manually or automatically to meet changes in demand.

Scaling limits
Set limits on how much your desired capacity can be increased or decreased.

Min desired capacity

Equal or less than desired capacity

Max desired capacity

Equal or greater than desired capacity

Automatic scaling - optional
Choose whether to use a target tracking policy | [Info](#)
You can set up other metric-based scaling policies and scheduled scaling after creating your Auto Scaling group.

☐ No scaling policies
Your Auto Scaling group will remain at its initial size and will not dynamically resize to meet demand.

☒ Target tracking scaling policy
Choose a CloudWatch metric and target value and let the scaling policy adjust the desired capacity in proportion to the metric's value.

Scaling policy name

Metric type | [Info](#)
Monitored metric that determines if resource utilization is too low or high. If using EC2 metrics, consider enabling detailed monitoring for better scaling performance.

Target value

Instance warmup | [Info](#)
 seconds

☐ Disable scale in to create only a scale-out policy

19. **Review & create** autoscaling group.
20. Stress test the EC2 instance via SSM login to webserver, use the below command on primary webserver created by autoscaling group this will trigger the second instance in ASG to launch:
- python3 -c "while True: x = [i**2 for i in range(1000000)]"
21. Wait for the **second instance** to spin up by ASG, Observe the different instance ID on the webpage by accessing the webpage via Loadbalancer, refresh the page to see different instance id's

To-Do List

Add Task

Frontend Dummy Task 1



Frontend Dummy Task 2



Served by EC2 Instance ID: i-03581407478a3b88e

To-Do List

Add Task

Frontend Dummy Task 1



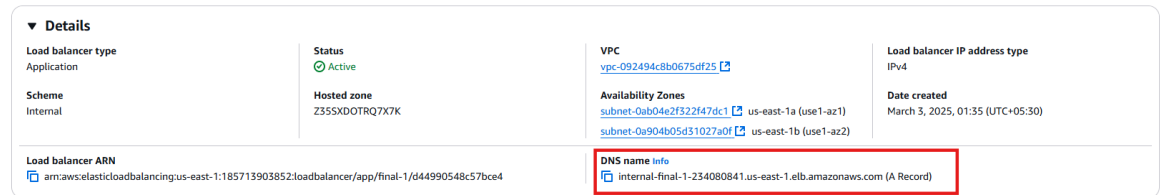
Frontend Dummy Task 2



Served by EC2 Instance ID: i-0dbee27b5a9a4c635

Step 8: Create Load Balancer and Autoscaling for Frontend

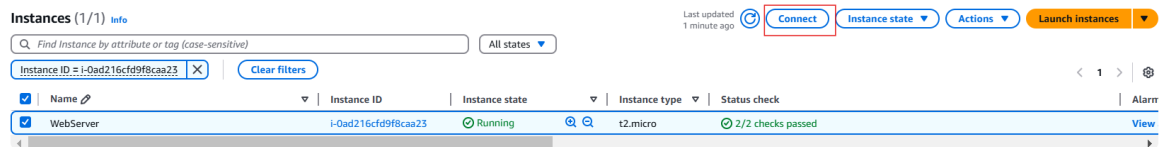
1. Navigate to **load balancer** in console and make sure internal load balancer is in Available state. Copy its **DNS name**.



▼ Details

Load balancer type Application	Status Active	VPC vpc-092494c8b0675df25	Load balancer IP address type IPv4
Scheme Internal	Hosted zone Z355XDTRQ7X7K	Availability Zones subnet-0ab04e2f322f47dc1 us-east-1a (use1-az1) subnet-0a904b05d31027a0f us-east-1b (use1-az2)	Date created March 3, 2025, 01:35 (UTC+05:30)
Load balancer ARN arn:aws:elasticloadbalancing:us-east-1:185713903852:loadbalancer/app/final-1/d44990548c57bce4		DNS name info internal-final-1-234080841.us-east-1.elb.amazonaws.com (A Record)	

2. Navigate to EC2 Instance and choose **Webserver**. Select **Connect**.



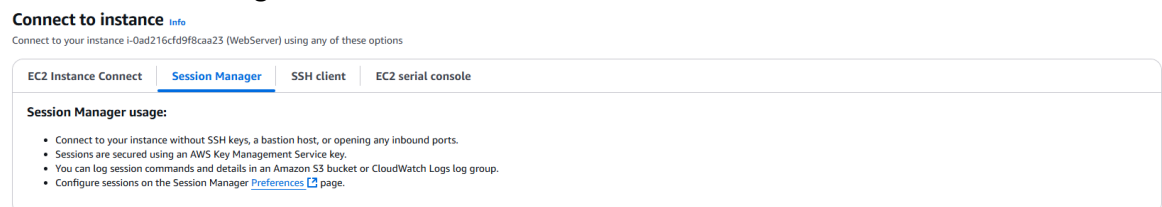
Instances (1/1) Info

Find Instance by attribute or tag (case-sensitive) All states

Instance ID: i-0ad216cfd9f8caa23 Clear filters

Name	Instance ID	Instance state	Instance type	Status check
WebServer	i-0ad216cfd9f8caa23	Running	t2.micro	2/2 checks passed

3. Select **Session Manager** tab and click on **Connect**.



Connect to instance Info

Connect to your instance i-0ad216cfd9f8caa23 (WebServer) using any of these options

EC2 Instance Connect Session Manager SSH client EC2 serial console

Session Manager usage:

- Connect to your instance without SSH keys, a bastion host, or opening any inbound ports.
- Sessions are secured using an AWS Key Management Service key.
- You can log session commands and details in an Amazon S3 bucket or CloudWatch Logs log group.
- Configure sessions on the Session Manager Preferences page.

4. Execute following steps:

- b. `sudo su ec2-user`
- c. `cd ~`
- d. `cd flaskapp/frontend/`

```
sh-4.2$ sudo su ec2-user
[ec2-user@ip-10-0-0-243 bin]$ cd ~
[ec2-user@ip-10-0-0-243 ~]$ cd flaskapp/frontend/
[ec2-user@ip-10-0-0-243 frontend]$ sudo vim frontend.py
[ec2-user@ip-10-0-0-243 frontend]$
```

- e. `sudo vim frontend.py`
- f. Press the **i** key to enter insert mode and replace the **BACKEND_API** variable with the endpoint of the **LoadBalancer** in the format of `http://<loadbalancer_endpoint> 5000/`

```
from flask import Flask, render_template, request, redirect, url_for
import requests

app = Flask(__name__)

# Backend API URL - Replace with your actual backend DNS or IP
BACKEND_API = "http://internal-final-1-234080841.us-east-1.elb.amazonaws.com:5000" # Update this with Load Balancer later
```

- g. Press the **Esc** key, type `:wq!`, and hit **Enter** to save and exit.
- h. `sudo fuser -k 80/tcp`
- i. `sudo nohup python3 frontend.py --port 80 &`

```
[ec2-user@ip-10-0-4-40 backend]$ sudo nohup python3 backend.py --port 5000 &
[1] 1169
[ec2-user@ip-10-0-4-40 backend]$ nohup: ignoring input and appending output to 'nohup.out'
[ec2-user@ip-10-0-4-40 backend]$
```

- j. Hit **Enter** twice.

13. Open your browser and search for `http://<public_id_of_webserver>:80`. The screen below should appear.

To-Do List

Add Task

14. Create Image for Autoscaling group, Click on actions & select **'Image & template'** create an image using the instance. Provide the **Image name** as **Frontend-AMI** and **Create Image**. Make sure its status appears **Available**.

The screenshot shows the AWS Management Console 'Instances' page. Two instances are listed: 'WebServer' (ID: i-0ad216cfd9f8caa23) and 'Appserver' (ID: i-0160d7c60c30e7123). Both are in a 'Running' state. The 'WebServer' instance is selected, and the 'Actions' menu is open, showing options like 'Connect', 'View details', 'Manage instance state', 'Instance settings', 'Networking', 'Security', 'Image and templates', and 'Monitor and troubleshoot'. The 'Create image' option is highlighted in red.

15. Navigate to EC2 Console. Click Launch Templates in the left sidebar and click on create a **New Launch Template**. Enter Launch template name **frontend-launch-template**.

Launch template name and description

Launch template name - *required*

Must be unique to this account. Max 128 chars. No spaces or special characters like '&', '*', '!', '@'.

Template version description

Max 255 chars

Auto Scaling guidance

Select this if you intend to use this template with EC2 Auto Scaling

☐ Provide guidance to help me set up a template that I can use with EC2 Auto Scaling

► Template tags

► Source template

16. Under **Amazon Machine Image (AMI)**, select **'Frontend-AMI'** created above

The screenshot shows the AWS Management Console 'Application and OS Images (Amazon Machine Image)' page. It displays a search bar, tabs for 'Recents', 'My AMIs', and 'Quick Start', and filters for 'Don't include in launch template', 'Owned by me', and 'Shared with me'. Under the 'My AMIs' tab, the 'Frontend-AMI' is listed with details: AMI ID: ami-017882ef1226949d8, Virtualization: hvm, ENA enabled: true, Root device type: ebs. The 'Description' is 'Frontend-AMI' and the 'Architecture' is 'x86_64'.

17. Choose an **Instance Type** (e.g., 't2.micro')

▼ Instance type Info | Get advice

Advanced

Instance type

t2.micro

Family: t2 1 vCPU 1 GiB Memory Current generation: true On-Demand Windows base pricing: 0.0162 USD per Hour Free tier eligible

On-Demand Ubuntu Pro base pricing: 0.0134 USD per Hour On-Demand SUSE base pricing: 0.0116 USD per Hour

On-Demand RHEL base pricing: 0.026 USD per Hour On-Demand Linux base pricing: 0.0116 USD per Hour

All generations

Compare instance types

Additional costs apply for AMIs with pre-installed software

18. Under Key pair (login), **Proceed without a key pair**.

▼ 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

Proceed without a key pair (Not recommended)

Default value ▼

Create new key pair

19. Keep default network settings as '**don't include in launch template**'. For security group select **three-tier-application-WebTierSecurityGroup**.

▼ Network settings Info

Subnet Info

Don't include in launch template ▼

Create new subnet

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

Firewall (security groups) Info

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

Select existing security group

Create security group

Security groups Info

Select security groups

three-tier-application-WebTierSecurityGroup-tsLFZDj8DRCS sg-08e57ac99e83ab2f2 X

Compare security group rules

► Advanced network configuration

20. Under Advanced Details section, for **IAM instance profile** select **flask-ec2-role-
<your_user_id>** created in step-1.

▼ Advanced details Info

Domain join directory Info

Select ▼

Create new directory

IAM instance profile Info

flask-ec2-role-test1

arn:aws:iam::185713903852:instance-profile/flask-ec2-role-test1

Create new IAM profile

Hostname type Info

IP name ▼

DNS Hostname Info

☒ Enable IP name IPv4 (A record) DNS requests

21. Scroll down to **User Data** and copy contents from this [Link](#).

User data - optional [Info](#)

Upload a file with your user data or enter it in the field.

[Choose file](#)

```
#!/bin/bash
yum update -y
yum install -y python3 python3-pip git
pip3 install flask flask-cors mysql-connector-python requests pymsql
cd /home/ec2-user/flaskapp/frontend/
python3 /home/ec2-user/flaskapp/frontend/frontend.py
```

22. Click on **Create Launch Template**.

23. Open the **AWS Auto Scaling Console**. Click **Create an Auto Scaling group**

24. Enter ASG Name **frontend-asg**. Under **Launch Template**, select the one you created **frontend-launch-template**

Name

Auto Scaling group name

Enter a name to identify the group.

frontend-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.

frontend-launch-template

[Create a launch template](#)

Version

Default (1)

[Create a launch template version](#)

Description

-

AMI ID

ami-017882ef1226949d8

Key pair name

-

Launch template

frontend-launch-template
lt-0575f5ca7e16222dc

Security groups

-

Security group IDs

-

Instance type

t2.micro

Request Spot Instances

No

25. Choose the Network. Select the **'three-tier-application'** VPC. Choose two private **PublicSubnetAZ1** and **PublicTierSubnetAZ2**.

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-092494c8b0675df25 (three-tier-application)
10.0.0.0/16

[Create a VPC](#)

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

us-east-1a | subnet-066d4848342aecb96 (PublicSubnetAZ1) X
10.0.0.0/24

us-east-1b | subnet-0fa0ab3ca9af653ee (PublicSubnetAZ2) X
10.0.1.0/24

[Create a subnet](#)

Availability Zone distribution - new

Auto Scaling automatically balances instances across Availability Zones. If launch failures occur in a zone, select a strategy.

☒ Balanced best effort

If launches fail in one Availability Zone, Auto Scaling will attempt to launch in another healthy Availability Zone.

☐ Balanced only

If launches fail in one Availability Zone, Auto Scaling will continue to attempt to launch in the unhealthy Availability Zone to preserve balanced distribution.

[Cancel](#)

[Skip to review](#)

[Previous](#)

[Next](#)

26. Add Health Check & Load Balancer. Create and attach a **new load-balancer**
27. Select **'Internet facing load-balancer'** & **'public'** subnets, under port type **'80'**. Under default routing select **'Create a target group'**

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

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

☒ Application Load Balancer
HTTP, HTTPS

☐ Network Load Balancer
TCP, UDP, TLS

Load balancer name
Name cannot be changed after the load balancer is created.
frontend-asg-1

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

☐ Internal

☒ Internet-facing

Network mapping
Your new load balancer will be created using the same VPC and Availability Zone selections as your Auto Scaling group. You can select different subnets and add subnets from additional Availability Zones.

VPC
vpc-092494c8b0675df25 [three-tier-application](#)

Availability Zones and subnets
You must select a single subnet for each Availability Zone enabled. Only public subnets are available for selection to support DNS resolution.

☒ us-east-1b
subnet-0fa0ab3ca9af653ee

☒ us-east-1a
subnet-066d4848342aecb96

Listeners and routing
If you require secure listeners, or multiple listeners, you can configure them from the [Load Balancing console](#) after your load balancer is created.

Protocol
HTTP

Port
80

Default routing (forward to)
Create a target group

28. Update the scaling configuration by setting the **'max desired capacity'** to **'2'** and selecting the **'target tracking scaling policy'** under **'Automatic scaling'** options:

Scaling [Info](#)
You can resize your Auto Scaling group manually or automatically to meet changes in demand.

Scaling limits
Set limits on how much your desired capacity can be increased or decreased.

Min desired capacity
1
Equal or less than desired capacity

Max desired capacity
2
Equal or greater than desired capacity

Automatic scaling - optional
Choose whether to use a target tracking policy | [Info](#)
You can set up other metric-based scaling policies and scheduled scaling after creating your Auto Scaling group.

☐ No scaling policies
Your Auto Scaling group will remain at its initial size and will not dynamically resize to meet demand.

☒ Target tracking scaling policy
Choose a CloudWatch metric and target value and let the scaling policy adjust the desired capacity in proportion to the metric's value.

Scaling policy name
Target Tracking Policy

Metric type | [Info](#)
Monitored metric that determines if resource utilization is too low or high. If using EC2 metrics, consider enabling detailed monitoring for better scaling performance.

Average CPU utilization

Target value
50

Instance warmup | [Info](#)
300 seconds

☐ Disable scale in to create only a scale-out policy

29. **Review & create** autoscaling group.
30. The application now can be accessed using DNS name of the new **frontend load balancer**.