

Task : 3-Tier Architecture: Dockerize Angular-Frontend and Java-Backend Application with RDS Integration

Step 1: Create and ec2 instance and launch it by using custom user data to install docker in it.

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

angular-java

Add additional tags

▼ Application and OS Images (Amazon Machine Image) Info

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

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

Recents

Quick Start

Amazon Linux

aws

macOS

Mac

Ubuntu

ubuntu

Windows

Microsoft

Red Hat

Red Hat

SUSE LI

SUSE

Q

Browse more AMIs

Including AMIs from AWS, Marketplace and the Community

Amazon Machine Image (AMI)

Ubuntu Server 24.04 LTS (LVM) - SSD Volume Type

Free tier eligible

t2.medium
Family: t2 2 vCPU 4 GiB Memory Current generation: true
On-Demand Linux base pricing: 0.0496 USD per Hour
On-Demand Windows base pricing: 0.0676 USD per Hour
On-Demand RHEL base pricing: 0.0784 USD per Hour
On-Demand SUSE base pricing: 0.1496 USD per Hour

☐ All generations
[Compare Instance types](#)

Additional costs apply for AMIs with pre-installed software

▼ 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*

seytan_cloud

Create new key pair

▼ Network settings Info

Edit

Network Info

vpc-09066077d7529c57f

Subnet Info

No preference (Default subnet in any availability zone)

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

V2 only (token required)

⚠ For V2 requests, you must include a session token in all Instance metadata requests. Applications or agents that use V1 for Instance metadata access will break.

Metadata response hop limit Info

2

Allow tags in metadata Info

Select

User data - optional Info

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

Choose file

```
#!/bin/bash

for pkg in docker.io docker-doc docker-compose docker-compose-v2 podman-
docker containerd runc; do sudo apt-get remove $pkg; done

# Add Docker's official GPG key:
sudo apt-get update -y
sudo apt-get install ca-certificates curl -y
sudo install -m 0755 -d /etc/apt/keyrings
sudo curl -fsSL https://download.docker.com/linux/ubuntu/gpg -o /etc/apt/
keyrings/docker.asc
sudo chmod a+r /etc/apt/keyrings/docker.asc

# Add the repository to Apt sources:
```

☐ User data has already been base64 encoded

Step 2: Take ssh of the instance.

```
seytan@seytan-Inspiron-3501:~$ ssh -i seytan_cloud.pem ubuntu@3.108.194.171
The authenticity of host '3.108.194.171 (3.108.194.171)' can't be established.
ED25519 key fingerprint is SHA256:Tz4cGr40106agsgeLweFfMYmbF9UThRRB9FY0yqCzfQ.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '3.108.194.171' (ED25519) to the list of known hosts.
Welcome to Ubuntu 24.04 LTS (GNU/Linux 6.8.0-1012-aws x86_64)

* Documentation:  https://help.ubuntu.com
* Management:    https://landscape.canonical.com
* Support:       https://ubuntu.com/pro

System information as of Fri Sep 20 05:40:25 UTC 2024

System load:  0.7                      Processes:            131
Usage of /:   24.7% of 6.71GB          Users logged in:     0
Memory usage: 7%                      IPv4 address for enX0: 172.31.34.179
Swap usage:   0%

Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status

The list of available updates is more than a week old.
To check for new updates run: sudo apt update

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.
```

Step 3: Now git clone the repository .

git clone <https://github.com/rajatpzade/angular-java.git>

```
Sep 20 11:10
ubuntu@ip-172-31-34-179:~$ git clone https://github.com/rajatpzade/angular-java.git
Cloning into 'angular-java'...
remote: Enumerating objects: 80, done.
remote: Counting objects: 100% (80/80), done.
remote: Compressing objects: 100% (62/62), done.
remote: Total 80 (delta 3), reused 80 (delta 3), pack-reused 0 (from 0)
Receiving objects: 100% (80/80), 268.11 KiB | 9.24 MiB/s, done.
Resolving deltas: 100% (3/3), done.
ubuntu@ip-172-31-34-179:~$ _
```

Step 4: Now update the os and install the mariadb -server .

```
Sep 20 11:11
ubuntu@ip-172-31-34-179:~$ sudo apt update
sudo apt install mariadb-server
sudo systemctl start mariadb
sudo systemctl enable mariadb
Hit:1 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu noble InRelease
Hit:2 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease
Hit:3 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease
Hit:4 https://download.docker.com/linux/ubuntu noble InRelease
Hit:5 http://security.ubuntu.com/ubuntu noble-security InRelease
Reading package lists... 84%
```

Step 5: Now create a rds database with user admin and password .

[RDS](#) > Create database

Create database [Info](#)


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.


Engine options


Engine type [Info](#)


☐ Aurora (MySQL Compatible) 

☐ Aurora (PostgreSQL Compatible) 

☐ MySQL 

☒ MariaDB 

☐ PostgreSQL 

☐ Oracle 

database-1

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

▼ **Credentials Settings**

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

admin

1 to 16 alphanumeric characters. 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 **Very weak**

Minimum constraints: At least 8 printable ASCII characters. Can't contain any of the following symbols: / * @

Confirm master password [Info](#)

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**

Step 6: Connect to database using following command and perform some actions.

mysql -h rds-endpoint -u username -p

```
ubuntu@ip-172-31-34-179: ~  
ubuntu@ip-172-31-34-179:~$ mysql -h database-1.c7kqi8k66m89.ap-south-1.rds.amazonaws.com -u admin -p12345678  
Welcome to the MariaDB monitor.  Commands end with ; or \g.  
Your MariaDB connection id is 26  
Server version: 10.11.8-MariaDB-Log managed by https://aws.amazon.com/rds/  
  
Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.  
  
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.  
  
MariaDB [(none)]> CREATE DATABASE springbackend;  
Query OK, 1 row affected (0.005 sec)  
  
MariaDB [(none)]> GRANT ALL PRIVILEGES ON springbackend.* TO 'username'@'localhost' IDENTIFIED BY 'your_password';  
Query OK, 0 rows affected (0.003 sec)  
  
MariaDB [(none)]>  
MariaDB [(none)]> GRANT ALL PRIVILEGES ON springbackend.* TO 'admin'@'3.108.194.171' IDENTIFIED BY '12345678';  
Query OK, 0 rows affected (0.004 sec)  
  
MariaDB [(none)]> FLUSH PRIVILEGES;  
Query OK, 0 rows affected (0.009 sec)  
  
MariaDB [(none)]> exit  
Bye  
ubuntu@ip-172-31-34-179:~$ mysql -h database-1.c7kqi8k66m89.ap-south-1.rds.amazonaws.com -u admin -p12345678 springbackend < angular-java/springbackend.sql  
ubuntu@ip-172-31-34-179:~$ _
```

Also copy the springbackend.sql file to the database springbackend in the rds.

Step 7: Now go to the spring-backend folder and create a Dockerfile .

Also make changes in the application.properties file and add the database endpoint and username and password there.

```
ubuntu@ip-172-31-34-179:~/angular-java/spring-backend  
ubuntu@ip-172-31-34-179:~/angular-java/spring-backend$ cd angular-java/spring-backend/  
ubuntu@ip-172-31-34-179:~/angular-java/spring-backend$ nano Dockerfile  
ubuntu@ip-172-31-34-179:~/angular-java/spring-backend$ cat Dockerfile  
FROM ubuntu:latest  
  
RUN apt update -y && \  
    apt install openjdk-8-jdk -y && \  
    apt install maven -y  
RUN rm -rf /var/lib/apt/lists/*  
WORKDIR /app  
  
COPY . /app  
  
# Create the directory for the properties file  
  
RUN mvn clean package -Dmaven.test.skip=true  
EXPOSE 8080  
  
CMD ["java", "-jar", "target/spring-backend-v1.jar"]  
ubuntu@ip-172-31-34-179:~/angular-java/spring-backend$ nano src/main/resources/application.properties  
ubuntu@ip-172-31-34-179:~/angular-java/spring-backend$ cat src/main/resources/application.properties  
spring.datasource.url=jdbc:mysql://database-1.c7kqi8k66m89.ap-south-1.rds.amazonaws.com:3306/springbackend?useSSL=false  
spring.datasource.username=admin  
spring.datasource.password=12345678  
  
spring.jpa.generate-ddl=true  
ubuntu@ip-172-31-34-179:~/angular-java/spring-backend$
```

Step 8: Now run the command docker build -t angular:backend to build the backend docker image.

```
ubuntu@ip-172-31-34-179: ~/angular-java/spring-backend$ sudo docker build -t angular:backend .
[+] Building 102.5s (13/13) FINISHED
=> [internal] load build definition from Dockerfile
=> => transferring dockerfile: 743B
=> [internal] load metadata for docker.io/library/ubuntu:latest
=> [internal] load .dockerignore
=> => transferring context: 2B
=> [1/8] FROM docker.io/library/ubuntu:latest@sha256:dfc10878be8d8fc9c61cbff33166cb1dfe44391539243783c72766894fa834a
=> => resolve docker.io/library/ubuntu:latest@sha256:dfc10878be8d8fc9c61cbff33166cb1dfe44391539243783c72766894fa834a
=> => sha256:b1e9cef312977f8b9dd19eb9ae84f83b315f88fe4f5c5651fedf41482c12432f7 2.38kB / 2.38kB
=> => sha256:dafa2b0c44d2cfb0bee721f079092ddf15dc8bc537fb07fe7c3264c15cb2e0e6 29.75MB / 29.75MB
=> => sha256:dfc10878be8d8fc9c61cbff33166cb1dfe44391539243783c72766894fa834a 1.34kB / 1.34kB
=> => sha256:774b57f089366f7d16615704a5b3e124d72484e2f03b3c2832233f1029b06a 424B / 424B
=> => extracting sha256:dafa2b0c44d2cfb0bee721f079092ddf15dc8bc537fb07fe7c3264c15cb2e0e6
=> [internal] load build context
=> => transferring context: 87.19kB
=> [2/8] RUN apt update -y && apt install openjdk-8-jdk -y && apt install maven -y
=> [3/8] RUN rm -rf /var/lib/apt/lists/*
=> [4/8] WORKDIR /app
=> [5/8] COPY . /app
=> [6/8] RUN rm -f ./src/main/resources/application.properties
=> [7/8] RUN echo "spring.datasource.url=jdbc:mysql://database-1.c7kql8k6m89.ap-south-1.rds.amazonaws.com:3306/springbackend?useSSL=false \nspring.datasource.username=admin \
=> [8/8] RUN mvn clean package -Dmaven.test.skip=true
=> exporting to image
=> => exporting layers
=> => writing image sha256:e802e66e4c7292cf2973582bf801474074d80b0b3a90422508fd1e24b1c25eb2
=> => naming to docker.io/library/angular:backend
ubuntu@ip-172-31-34-179: ~/angular-java/spring-backend$ sudo docker images
REPOSITORY TAG IMAGE ID CREATED SIZE
angular backend e802e66e4c72 42 seconds ago 859MB
ubuntu@ip-172-31-34-179: ~/angular-java/spring-backend$
```

Step 9: Now create another Dockerfile for the frontend in the folder angular-frontend.

In the line number 11 change the ip with your public ip of the instance.

OR you can change it manually in the worker.service.ts file .

```
ubuntu@ip-172-31-34-179: ~/angular-java/angular-frontend$ nano Dockerfile
ubuntu@ip-172-31-34-179: ~/angular-java/angular-frontend$ cat Dockerfile
FROM node:14 AS build

RUN apt update -y
WORKDIR /app

COPY package*.json ./

RUN npm install

COPY . .
RUN sed -i 's/localhost/3.108.194.171/g' ./src/app/services/worker.service.ts

RUN npm run build

FROM nginx:latest

WORKDIR /usr/share/nginx/html

COPY --from=build /app/dist/angular-frontend /usr/share/nginx/html

EXPOSE 80

CMD ["nginx", "-g", "daemon off;"]
ubuntu@ip-172-31-34-179: ~/angular-java/angular-frontend$
```

Step 10: Now Build the Docker image for the frontend also.

```
ubuntu@ip-172-31-34-179: ~/angular-java/angular-frontend$ sudo docker build -t angular:frontend .
[+] Building 105.2s (17/17) FINISHED
=> [internal] load build definition from Dockerfile
=> => transferring dockerfile: 407B
=> [internal] load metadata for docker.io/library/node:14
=> [internal] load metadata for docker.io/library/nginx:latest
=> [internal] load .dockerignore
=> transferring context: 2B
=> [build 1/8] FROM docker.io/library/node:14@sha256:a158d3b9b4e3fa813fa6c8c59bb8f0a860e015ad4e59bbce5744d2f6fd8461aa
=> resolve docker.io/library/node:14@sha256:a158d3b9b4e3fa813fa6c8c59bb8f0a860e015ad4e59bbce5744d2f6fd8461aa
=> sha256:3d2201bd995cccf12851a50820de03d34a17011dcbb9ac9fd3a50c952cbb131 10.00MB / 10.00MB
=> sha256:2cfa31bb0b0529ee4726b4f599ec27ee557ea3dea7019182323b3779959927f 2.21kB / 2.21kB
=> sha256:1d12470fa662a2a5cb50378dc8ea228c1735747db410bbefb8e2d9144b5452 7.51kB / 7.51kB
=> sha256:b253aeafeaa7e0671bb60080df01de101a38a045ff7bc656e3b0fbfc7c05cca5 7.86MB / 7.86MB
=> sha256:a158d3b9b4e3fa813fa6c8c59bb8f0a860e015ad4e59bbce5744d2f6fd8461aa 776B / 776B
=> sha256:2ff1d7c41c74a25258bfa6f0b8adb0a727f64518f55f65ca845ebc747976c408 50.45MB / 50.45MB
=> sha256:1de76e268b103d05fa8960e0f77951ff54b912b63429c34f5d6adfd09f5f9ee2 51.88MB / 51.88MB
=> sha256:d9a8df5894511ce28a05e2925a75e8a4acbd0634c39ad734fd0ba8e23d1b1569 191.85MB / 191.85MB
=> extracting sha256:2ff1d7c41c74a25258bfa6f0b8adb0a727f64518f55f65ca845ebc747976c408
=> sha256:4f51ee005deac0d99898e41b8ce60ebf259eb1a31a0b03f613aeb6bc9b83d8 4.19kB / 4.19kB
=> sha256:5f32ed3c3f279edda4fc571c880b5277355a29a0f52b52cdf965f058378a590 35.24MB / 35.24MB
=> sha256:9c8cc2f24a4dc64e602e006fc944600a541e8acd9ad72d2e90df3ba22f158b3 2.29MB / 2.29MB
=> sha256:0d27a8e861329007574c6766fba946d48e20d2c8e964e873de352603f22c4ceb 450B / 450B
=> extracting sha256:b253aeafeaa7e0671bb60080df01de101a38a045ff7bc656e3b0fbfc7c05cca5
=> extracting sha256:3d2201bd995cccf12851a50820de03d34a17011dcbb9ac9fd3a50c952cbb131
=> extracting sha256:1de76e268b103d05fa8960e0f77951ff54b912b63429c34f5d6adfd09f5f9ee2
=> extracting sha256:d9a8df5894511ce28a05e2925a75e8a4acbd0634c39ad734fd0ba8e23d1b1569
=> extracting sha256:6f51ee005deac0d99898e41b8ce60ebf259eb1a31a0b03f613aeb6bc9b83d8
=> extracting sha256:5f32ed3c3f279edda4fc571c880b5277355a29a0f52b52cdf965f058378a590
=> extracting sha256:9c8cc2f24a4dc64e602e006fc944600a541e8acd9ad72d2e90df3ba22f158b3
=> extracting sha256:0d27a8e861329007574c6766fba946d48e20d2c8e964e873de352603f22c4ceb
```

Step 11: Now check the images by command `docker images` .
After that run the docker images of both backend and frontend .

`sudo docker run -d -p 8080:8080 image_id/name && sudo docker run -d -p 80:80 image_id/name`

Afer that check the containers are running or not by command `docker ps`.

```
ubuntu@ip-172-31-34-179: ~$ sudo docker images
REPOSITORY   TAG       IMAGE ID       CREATED        SIZE
angular      frontend  0fb805cfe35f   45 seconds ago 188MB
angularr     backend  e802e66e4c72   10 minutes ago 858MB

ubuntu@ip-172-31-34-179: ~$ sudo docker run -d -p 8080:8080 e80 && sudo docker run -d -p 80:80 0fb
ed6ec2c20f19c317b57f312cb41f7669c18e646ca81dcfa29e174fc5a4030518
d24ef5466e78b3160cf415996d32ffefa0f12aebc57b3eb071ec60c8c0d64327

ubuntu@ip-172-31-34-179: ~$ sudo docker ps
CONTAINER ID   IMAGE     COMMAND                  CREATED        STATUS        PORTS                               NAMES
d24ef5466e78   0fb      "/docker-entrypoint..." 7 seconds ago  Up 6 seconds  0.0.0.0:80->80/tcp, :::80->80/tcp   elated_goldwasser
ed6ec2c20f19   e80      "java -jar target/sp..." 7 seconds ago  Up 6 seconds  0.0.0.0:8080->8080/tcp, :::8080->8080/tcp  recursing_franklin
ubuntu@ip-172-31-34-179: ~$
```


Output: Once the both containers are running paste the public ip in the web browser and check the results.

Workers

[Add Worker](#)

Order	First Name	Last Name	Status	Edit Button	Delete Button
1.	Ivan	Holicek	Working	Edit	<button>Delete</button>
2.	Marko	Markovic	Vacation	Edit	<button>Delete</button>
3.	Ivo	Ivica	Working	Edit	<button>Delete</button>
4.	Luka	Lukovic	Working	Edit	<button>Delete</button>
5.	Filip	Filipovic	Working	Edit	<button>Delete</button>
6.	sfs	sfs	fsf	Edit	<button>Delete</button>

Modified By CloudBlitz