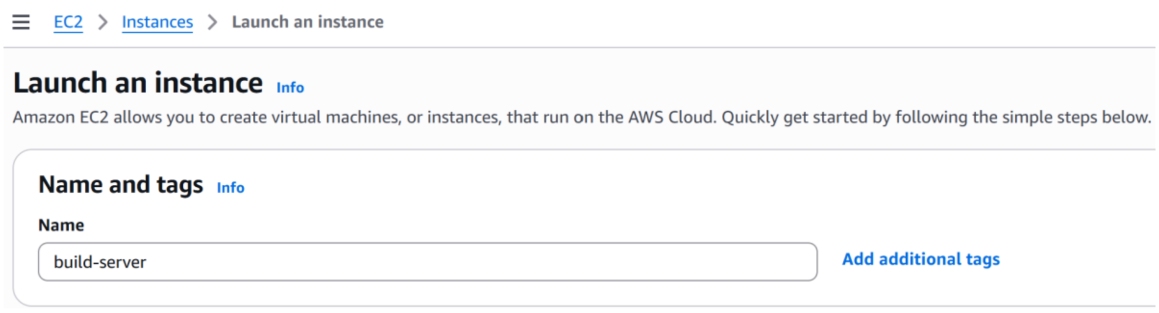
How to Build and Deploy a Java Application with a Database on the IaaS Model

Step 1: Create Three Virtual Machines (Build, Deploy, and Database Servers)

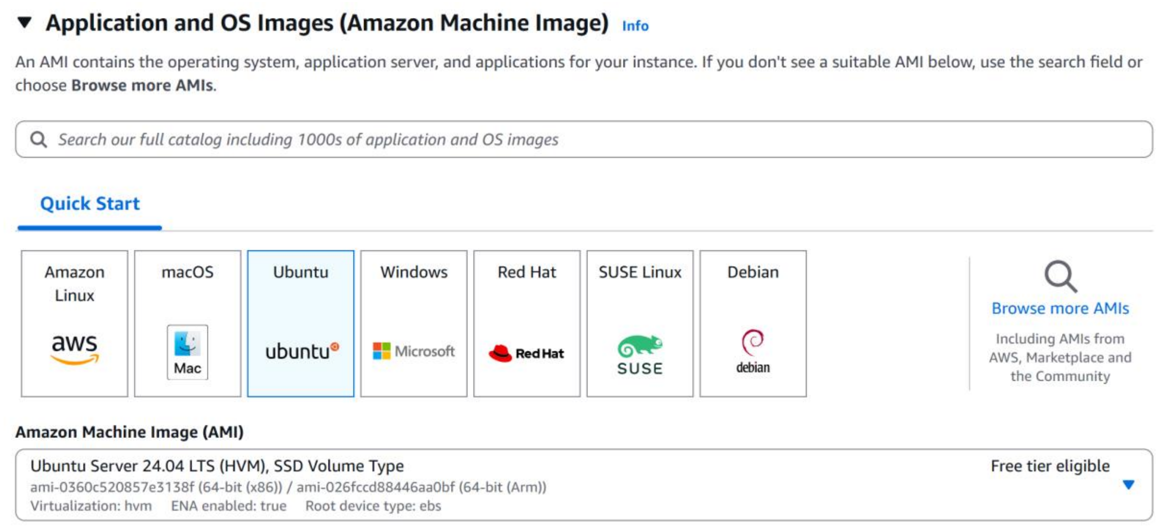
1. Go to EC2 Instances in AWS and click Launch Instance.
2. Enter a name for the instance.



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: build-server

3. Choose a Linux image (Ubuntu or Red Hat).

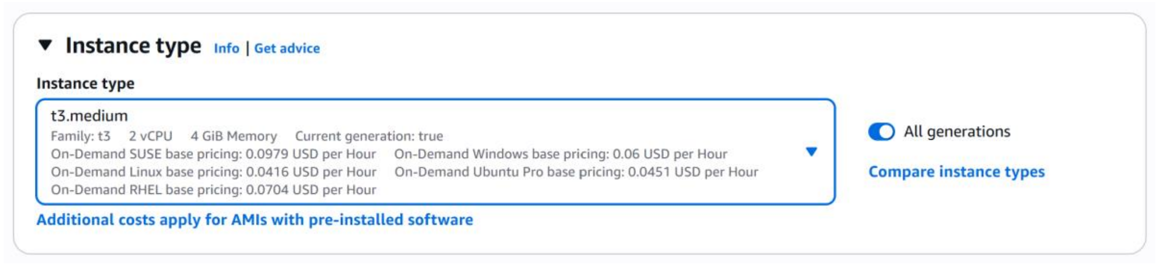

Application and OS Images (Amazon Machine Image) Info

An AMI contains the operating system, application server, and applications for your instance. If you don't see a suitable AMI below, use the search field or choose Browse more AMIs.

Search our full catalog including 1000s of application and OS images
Quick Start
Amazon Linux
macOS
Ubuntu
Windows
Red Hat
SUSE Linux
Debian
Browse more AMIs
Including AMIs from AWS, Marketplace and the Community

Amazon Machine Image (AMI)
Ubuntu Server 24.04 LTS (HVM), SSD Volume Type — Free tier eligible
ami-0360c520857e3138f (64-bit (x86)) / ami-026fccd88446aa0bf (64-bit (Arm))
Virtualization: hvm ENA enabled: true Root device type: ebs

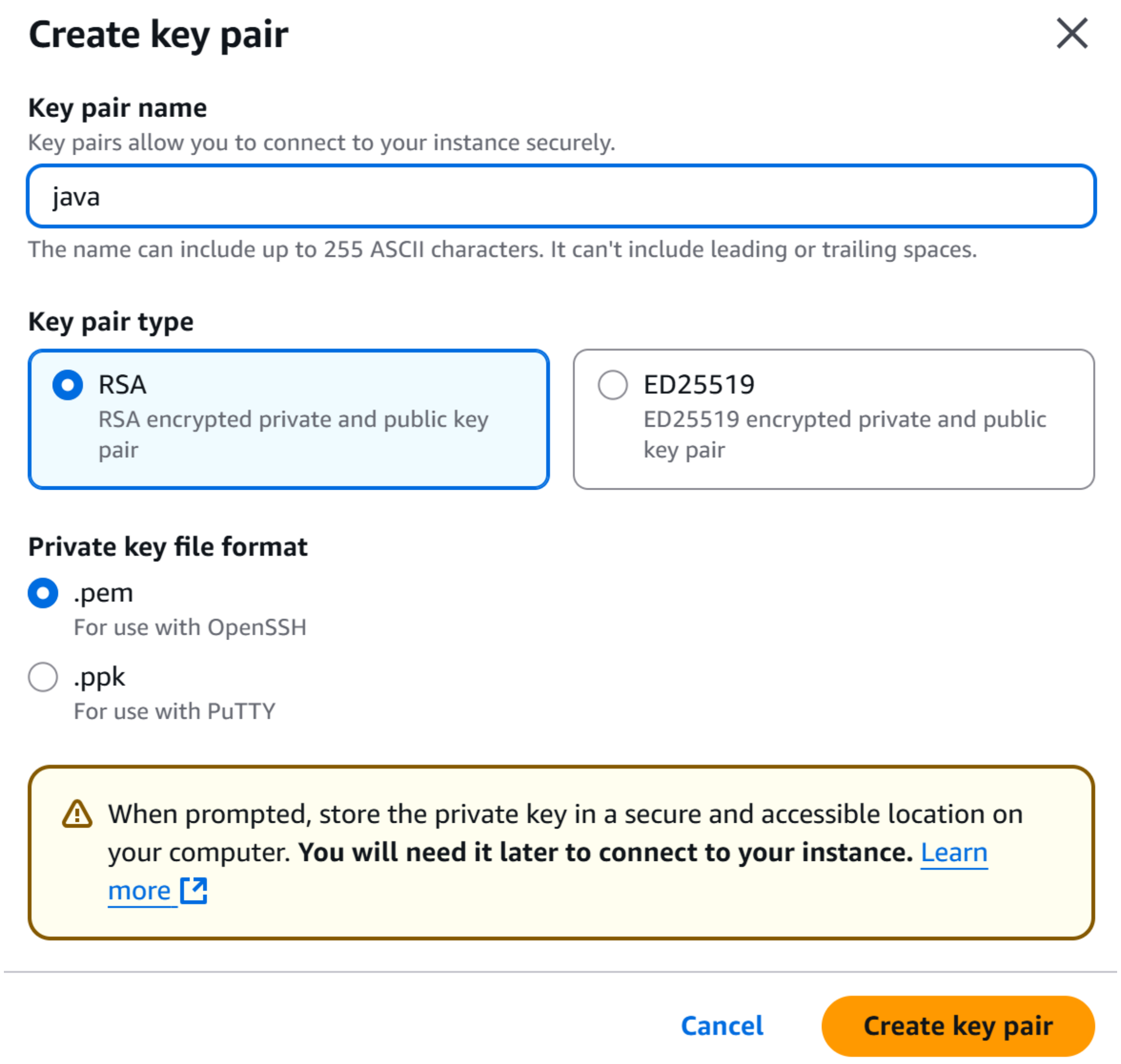
4. Select an instance type (e.g., t2.micro or t2.medium) as per your requirements.


Instance type Info | Get advice

Instance type: t3.medium
Family: t3 2 vCPU 4 GiB Memory Current generation: true
On-Demand SUSE base pricing: 0.0979 USD per Hour On-Demand Windows base pricing: 0.06 USD per Hour
On-Demand Linux base pricing: 0.0416 USD per Hour On-Demand Ubuntu Pro base pricing: 0.0451 USD per Hour
On-Demand RHEL base pricing: 0.0704 USD per Hour

All generations
Compare instance types
Additional costs apply for AMIs with pre-installed software

5. Create a key pair (.pem or .ppk format) and click Create Key Pair.


Create key pair

Key pair name

Key pairs allow you to connect to your instance securely.

java

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

Key pair type

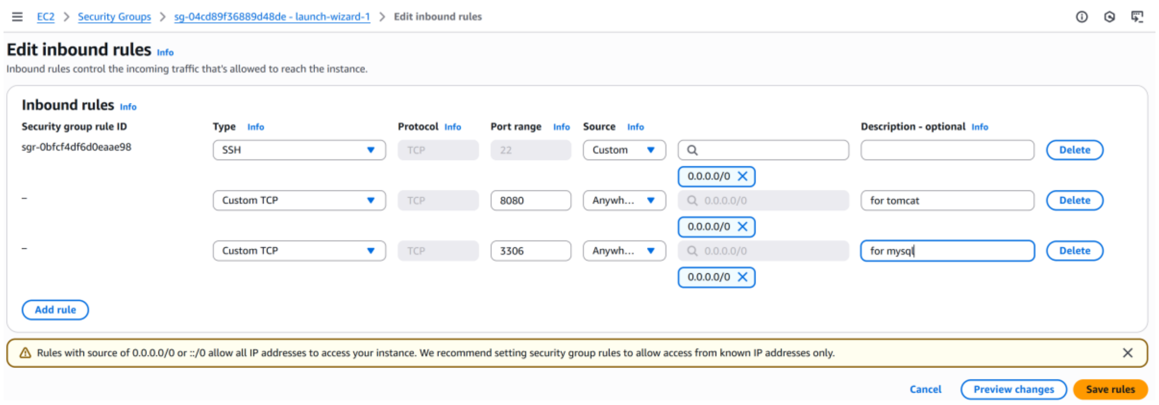
- [✓ RSA] — RSA encrypted private and public key pair
- ED25519 — ED25519 encrypted private and public key pair

Private key file format

- [✓ .pem] — For use with OpenSSH
- .ppk — For use with PuTTY

When prompted, store the private key in a secure and accessible location on your computer. **You will need it later to connect to your instance.** Learn more

6. Click Launch Instance.
7. In the Security Groups, edit Inbound Rules to allow:
   - 8080 (for Tomcat)
   - 3306 (for MySQL)


EC2 > Security Groups > sg-04cd89f36889d48de - launch-wizard-1 > Edit inbound rules

Edit inbound rules Info

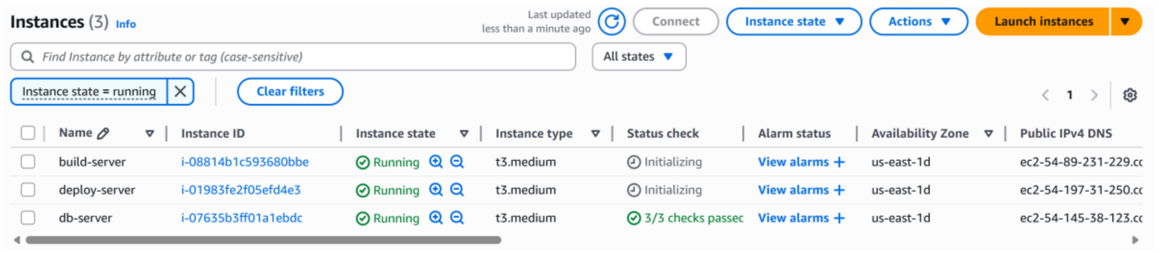
Inbound rules control the incoming traffic that's allowed to reach the instance.

Inbound rules Info

Security group rule ID	Type	Protocol	Port range	Source		Description - optional
sgr-0bfcf4df6d0eaae98	SSH	TCP	22	Custom	0.0.0.0/0	
–	Custom TCP	TCP	8080	Anywh...	0.0.0.0/0	for tomcat
–	Custom TCP	TCP	3306	Anywh...	0.0.0.0/0	for mysql

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

8. Connect to the instances using MobaXterm or any SSH client with the public IP and .pem file.


Instances (3) Info

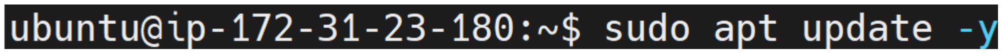
Last updated less than a minute ago

Instance state = running

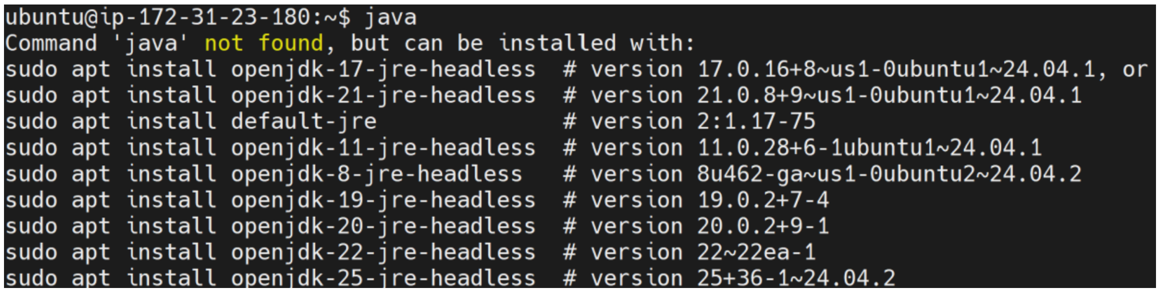
Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS
build-server	i-08814b1c593680bbe	Running	t3.medium	Initializing	View alarms +	us-east-1d	ec2-54-89-231-229.cc
deploy-server	i-01983fe2f05efd4e3	Running	t3.medium	Initializing	View alarms +	us-east-1d	ec2-54-197-31-250.cc
db-server	i-07635b3ff01a1ebdc	Running	t3.medium	3/3 checks passec	View alarms +	us-east-1d	ec2-54-145-38-123.cc

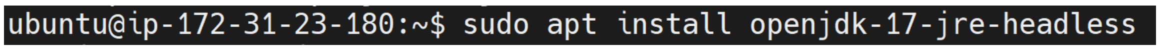
Step 2: Configure the Build Server

1. Update the server: sudo apt update -y

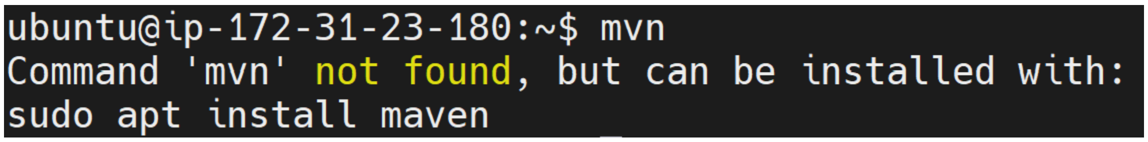

```
ubuntu@ip-172-31-23-180:~$ sudo apt update -y
```

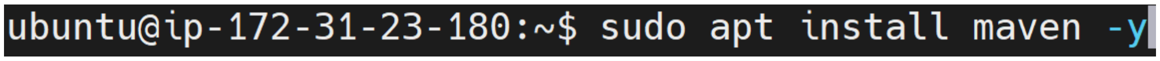
2. Check Java installation: java -version
   - If not installed: sudo apt install openjdk-17-jdk -y


```
ubuntu@ip-172-31-23-180:~$ java
Command 'java' not found, but can be installed with:
sudo apt install openjdk-17-jre-headless  # version 17.0.16+8~us1-0ubuntu1~24.04.1, or
sudo apt install openjdk-21-jre-headless  # version 21.0.8+9~us1-0ubuntu1~24.04.1
sudo apt install default-jre              # version 2:1.17-75
sudo apt install openjdk-11-jre-headless  # version 11.0.28+6-1ubuntu1~24.04.1
sudo apt install openjdk-8-jre-headless   # version 8u462-ga~us1-0ubuntu2~24.04.2
sudo apt install openjdk-19-jre-headless  # version 19.0.2+7-4
sudo apt install openjdk-20-jre-headless  # version 20.0.2+9-1
sudo apt install openjdk-22-jre-headless  # version 22~22ea-1
sudo apt install openjdk-25-jre-headless  # version 25+36-1~24.04.2
```


```
ubuntu@ip-172-31-23-180:~$ sudo apt install openjdk-17-jre-headless
```

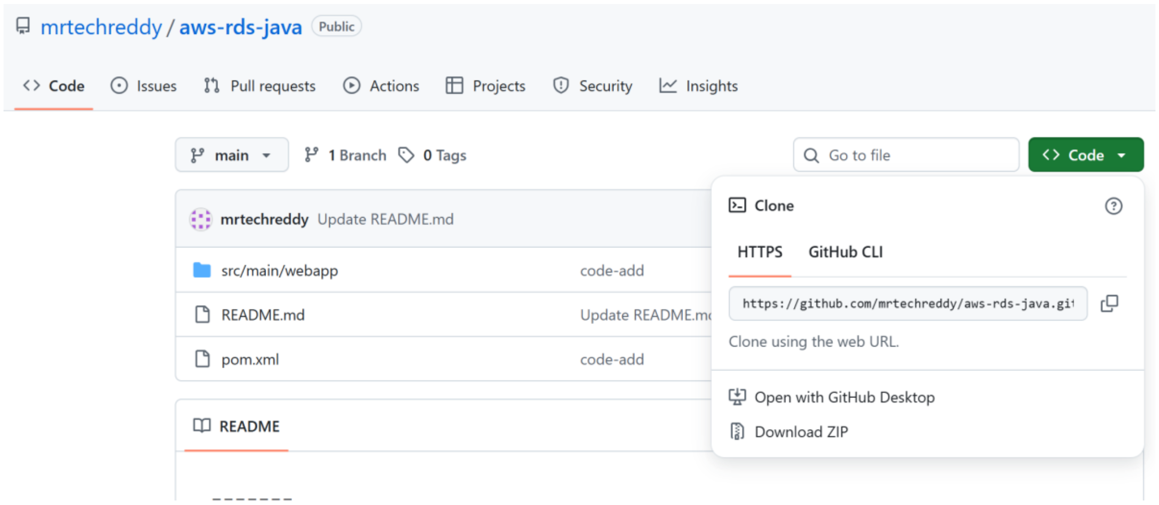
3. Check Maven installation: mvn -version
   - If not installed: sudo apt install maven -y

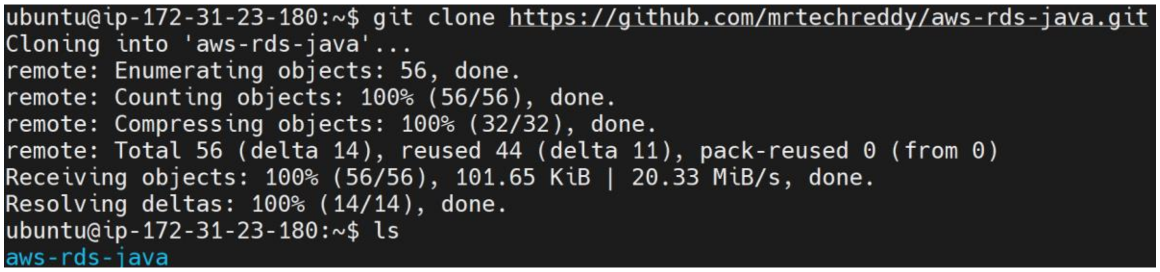

```
ubuntu@ip-172-31-23-180:~$ mvn
Command 'mvn' not found, but can be installed with:
sudo apt install maven
```



```
ubuntu@ip-172-31-23-180:~$ sudo apt install maven -y
```

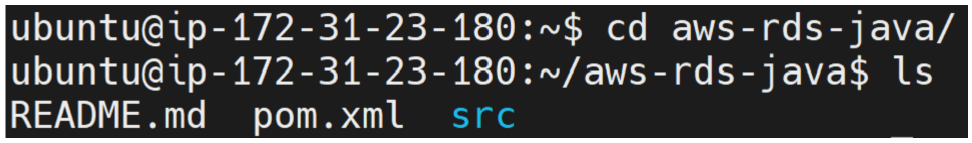
4. Clone the repository: git clone <repository_url>





```
ubuntu@ip-172-31-23-180:~$ git clone https://github.com/mrtechreddy/aws-rds-java.git
Cloning into 'aws-rds-java'...
remote: Enumerating objects: 56, done.
remote: Counting objects: 100% (56/56), done.
remote: Compressing objects: 100% (32/32), done.
remote: Total 56 (delta 14), reused 44 (delta 11), pack-reused 0 (from 0)
Receiving objects: 100% (56/56), 101.65 KiB | 20.33 MiB/s, done.
Resolving deltas: 100% (14/14), done.
ubuntu@ip-172-31-23-180:~$ ls
aws-rds-java
```

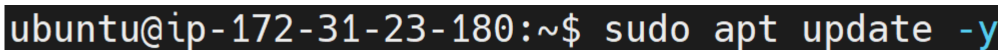
5. Navigate to the cloned directory: cd <repo_folder_name>



```
ubuntu@ip-172-31-23-180:~$ cd aws-rds-java/
ubuntu@ip-172-31-23-180:~/aws-rds-java$ ls
README.md  pom.xml  src
```

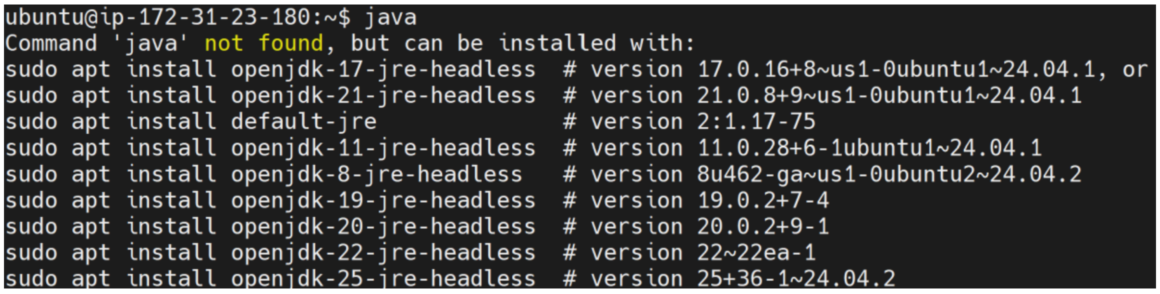
Step 3: Configure the Deploy Server

1. Update the server: sudo apt update -y

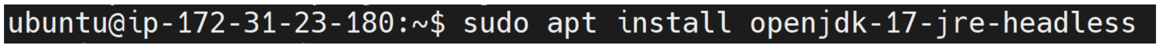


```
ubuntu@ip-172-31-23-180:~$ sudo apt update -y
```

2. Check Java installation: java -version
   - If not installed: sudo apt install openjdk-17-jdk -y



```
ubuntu@ip-172-31-23-180:~$ java
Command 'java' not found, but can be installed with:
sudo apt install openjdk-17-jre-headless  # version 17.0.16+8~us1-0ubuntu1~24.04.1, or
sudo apt install openjdk-21-jre-headless  # version 21.0.8+9~us1-0ubuntu1~24.04.1
sudo apt install default-jre              # version 2:1.17-75
sudo apt install openjdk-11-jre-headless  # version 11.0.28+6-1ubuntu1~24.04.1
sudo apt install openjdk-8-jre-headless   # version 8u462-ga~us1-0ubuntu2~24.04.2
sudo apt install openjdk-19-jre-headless  # version 19.0.2+7-4
sudo apt install openjdk-20-jre-headless  # version 20.0.2+9-1
sudo apt install openjdk-22-jre-headless  # version 22~22ea-1
sudo apt install openjdk-25-jre-headless  # version 25+36-1~24.04.2
```

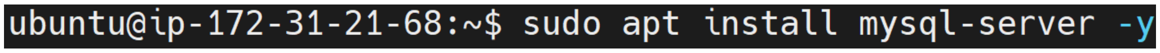


```
ubuntu@ip-172-31-23-180:~$ sudo apt install openjdk-17-jre-headless
```

3. Download and install Tomcat:
   wget https://dlcdn.apache.org/tomcat/tomcat-9/v9.0.110/bin/apache-tomcat-9.0.110.tar.gz
4. Extract and rename:
   tar -xvf apache-tomcat-9.0.110.tar.gz
   mv apache-tomcat-9.0.110 tomcat
5. Start Tomcat: cd tomcat/bin && ./startup.sh
6. Configure tomcat-users.xml, context.xml, and restart the Tomcat server.
7. Access Tomcat using http://<deploy-server-public-ip>:8080

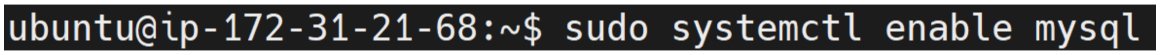
Step 4: Configure the Database Server

1. Update and install MySQL:
   sudo apt update -y
   sudo apt install mysql-server -y

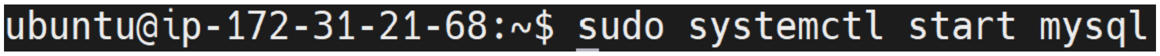


```
ubuntu@ip-172-31-21-68:~$ sudo apt install mysql-server -y
```

2. Enable and start MySQL:
   sudo systemctl enable mysql
   sudo systemctl start mysql

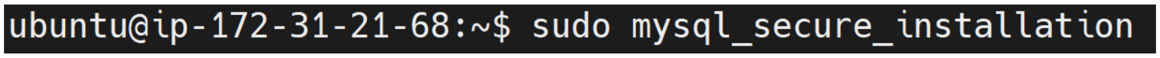


```
ubuntu@ip-172-31-21-68:~$ sudo systemctl enable mysql
```



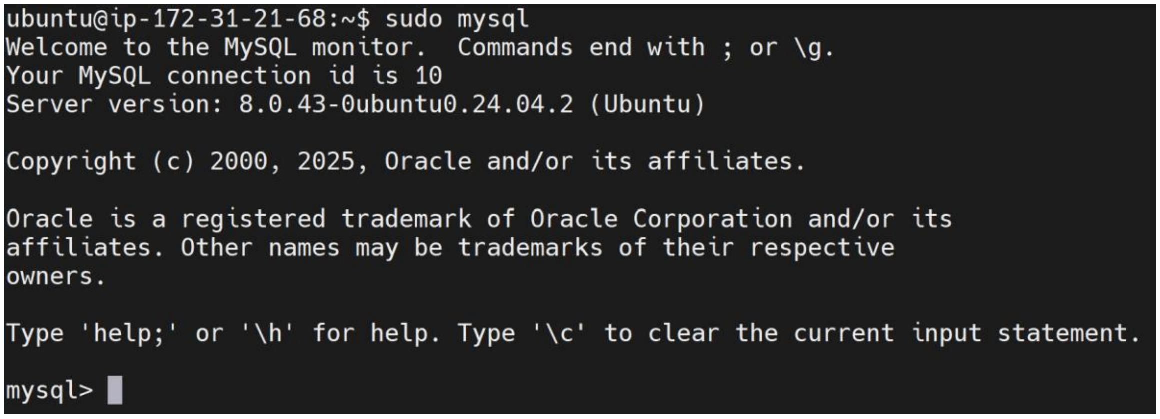
```
ubuntu@ip-172-31-21-68:~$ sudo systemctl start mysql
```

3. Secure MySQL: sudo mysql_secure_installation



```
ubuntu@ip-172-31-21-68:~$ sudo mysql_secure_installation
```

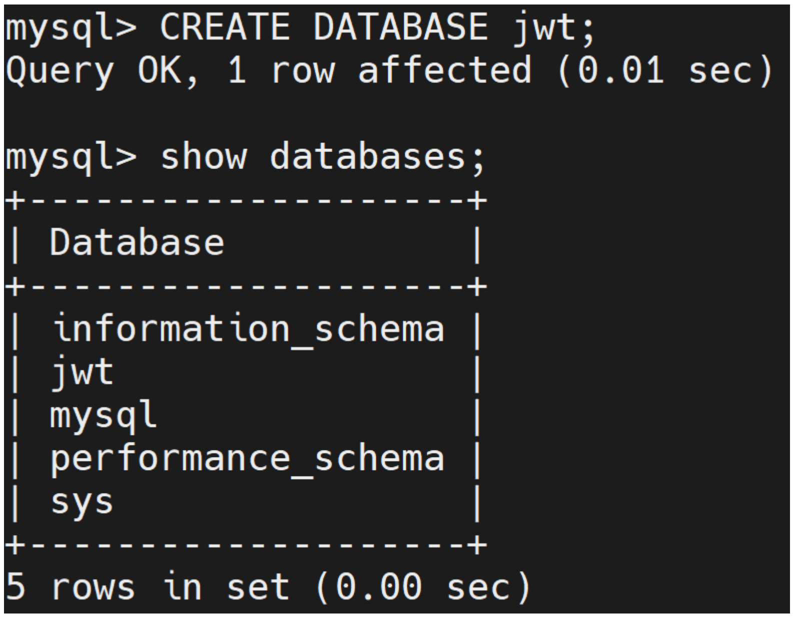
4. Login to MySQL: sudo mysql



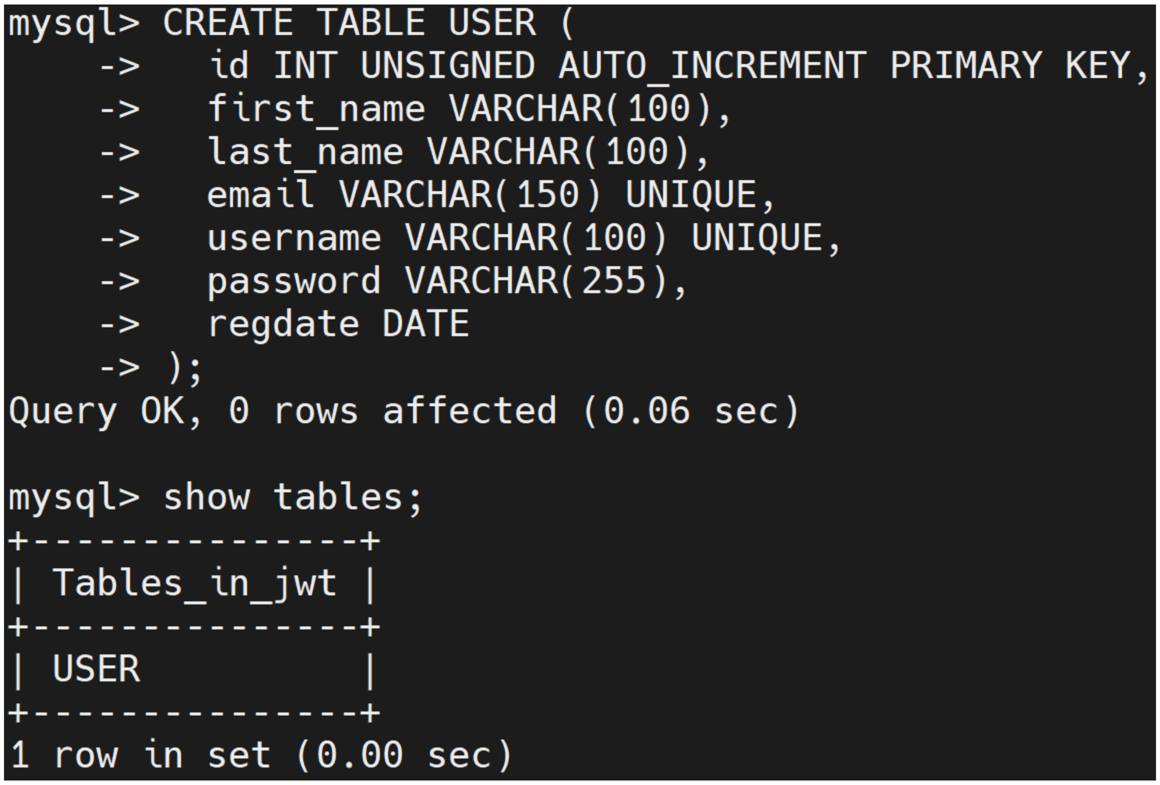
```
ubuntu@ip-172-31-21-68:~$ sudo mysql
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 10
Server version: 8.0.43-0ubuntu0.24.04.2 (Ubuntu)

Copyright (c) 2000, 2025, 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.

mysql> 
```

5. Create database and user:
   CREATE DATABASE jwt;
   CREATE USER 'appuser'@'%' IDENTIFIED BY 'password';

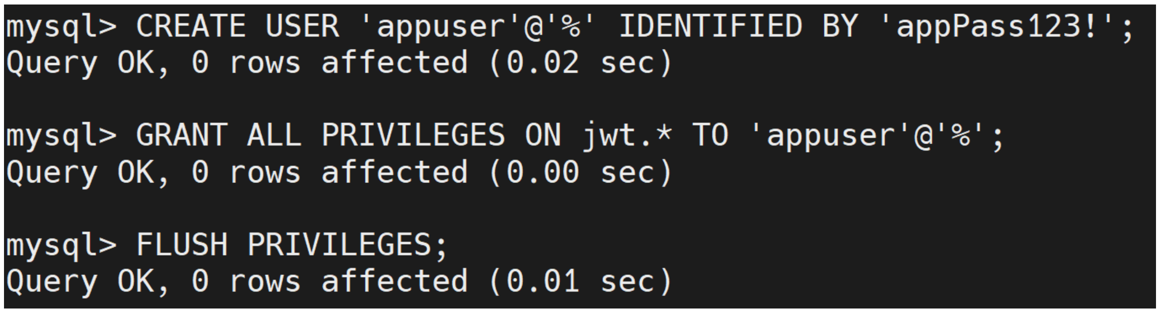
GRANT ALL PRIVILEGES ON jwt.* TO 'appuser'@'%';
FLUSH PRIVILEGES;



```
mysql> CREATE DATABASE jwt;
Query OK, 1 row affected (0.01 sec)

mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| jwt                |
| mysql              |
| performance_schema |
| sys                |
+--------------------+
5 rows in set (0.00 sec)
```

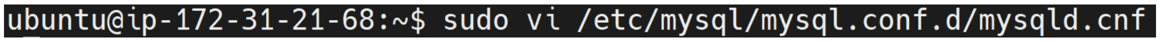


```
mysql> CREATE TABLE USER (
    ->   id INT UNSIGNED AUTO_INCREMENT PRIMARY KEY,
    ->   first_name VARCHAR(100),
    ->   last_name VARCHAR(100),
    ->   email VARCHAR(150) UNIQUE,
    ->   username VARCHAR(100) UNIQUE,
    ->   password VARCHAR(255),
    ->   regdate DATE
    -> );
Query OK, 0 rows affected (0.06 sec)

mysql> show tables;
+---------------+
| Tables_in_jwt |
+---------------+
| USER          |
+---------------+
1 row in set (0.00 sec)
```

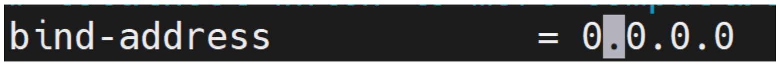


```
mysql> CREATE USER 'appuser'@'%' IDENTIFIED BY 'appPass123!';
Query OK, 0 rows affected (0.02 sec)

mysql> GRANT ALL PRIVILEGES ON jwt.* TO 'appuser'@'%';
Query OK, 0 rows affected (0.00 sec)

mysql> FLUSH PRIVILEGES;
Query OK, 0 rows affected (0.01 sec)
```

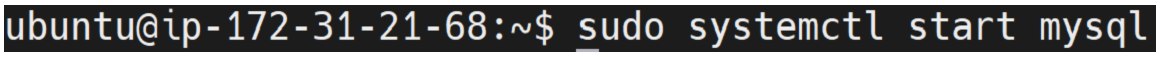
6. Edit bind-address to 0.0.0.0 and restart MySQL.



```
ubuntu@ip-172-31-21-68:~$ sudo vi /etc/mysql/mysql.conf.d/mysqld.cnf
```



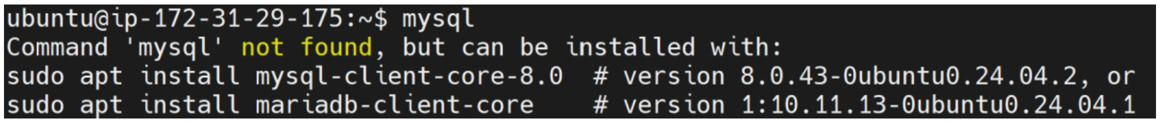
```
bind-address            = 0.0.0.0
```



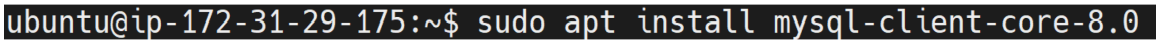
```
ubuntu@ip-172-31-21-68:~$ sudo systemctl start mysql
```

Step 5: Verify Database Connection from Deploy Server

1. Install MySQL client: sudo apt install mysql-client -y

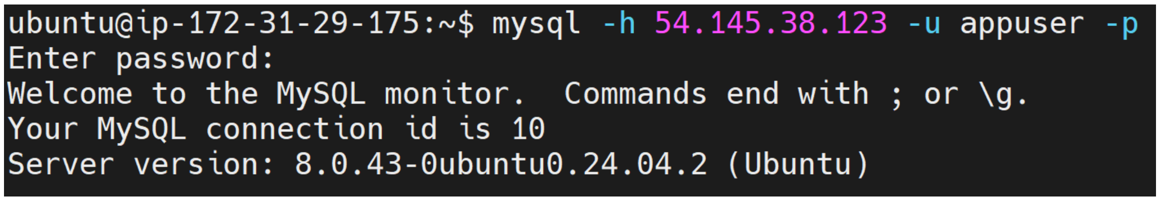


```
ubuntu@ip-172-31-29-175:~$ mysql
Command 'mysql' not found, but can be installed with:
sudo apt install mysql-client-core-8.0   # version 8.0.43-0ubuntu0.24.04.2, or
sudo apt install mariadb-client-core     # version 1:10.11.13-0ubuntu0.24.04.1
```



```
ubuntu@ip-172-31-29-175:~$ sudo apt install mysql-client-core-8.0
```

2. Connect: mysql -h <db-server-private-ip> -u appuser -p

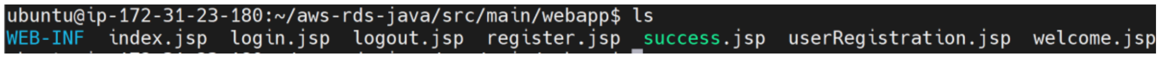


```
ubuntu@ip-172-31-29-175:~$ mysql -h 54.145.38.123 -u appuser -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 10
Server version: 8.0.43-0ubuntu0.24.04.2 (Ubuntu)
```

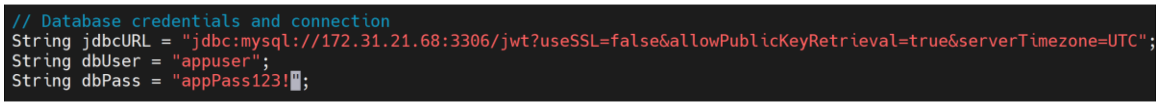
3. Verify: SHOW DATABASES; USE jwt; SHOW TABLES;

Step 6: Configure Application in Build Server

1. Update database connection details in JSP files: userRegistration.jsp and login.jsp

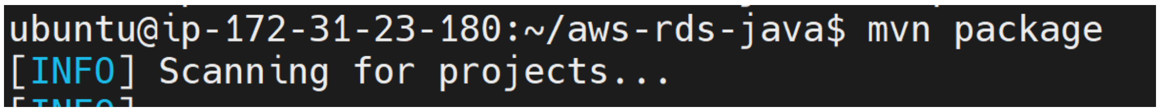


```
ubuntu@ip-172-31-23-180:~/aws-rds-java/src/main/webapp$ ls
WEB-INF  index.jsp  login.jsp  logout.jsp  register.jsp  success.jsp  userRegistration.jsp  welcome.jsp
```

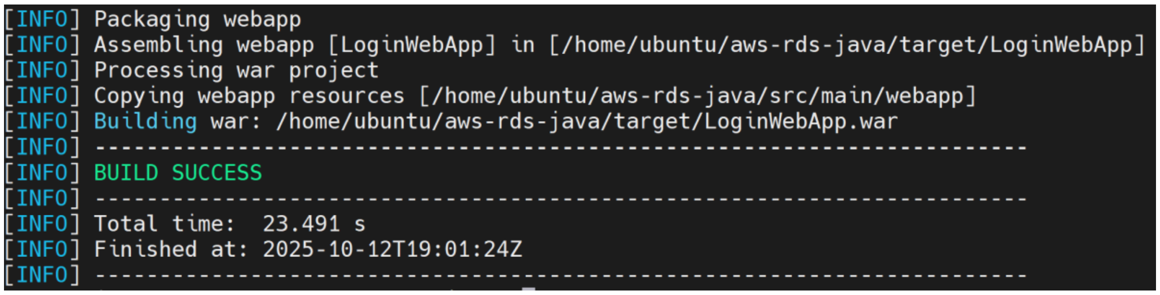


```
// Database credentials and connection
String jdbcURL = "jdbc:mysql://172.31.21.68:3306/jwt?useSSL=false&allowPublicKeyRetrieval=true&serverTimezone=UTC";
String dbUser = "appuser";
String dbPass = "appPass123!";
```

2. Build artifact: mvn package

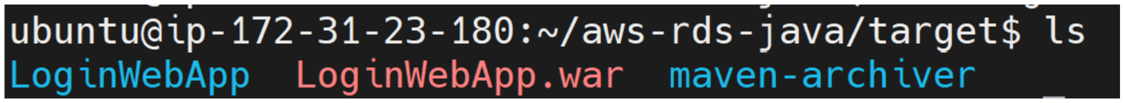


```
ubuntu@ip-172-31-23-180:~/aws-rds-java$ mvn package
[INFO] Scanning for projects...
```



```
[INFO] Packaging webapp
[INFO] Assembling webapp [LoginWebApp] in [/home/ubuntu/aws-rds-java/target/LoginWebApp]
[INFO] Processing war project
[INFO] Copying webapp resources [/home/ubuntu/aws-rds-java/src/main/webapp]
[INFO] Building war: /home/ubuntu/aws-rds-java/target/LoginWebApp.war
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time:  23.491 s
[INFO] Finished at: 2025-10-12T19:01:24Z
[INFO] ------------------------------------------------------------------------
```

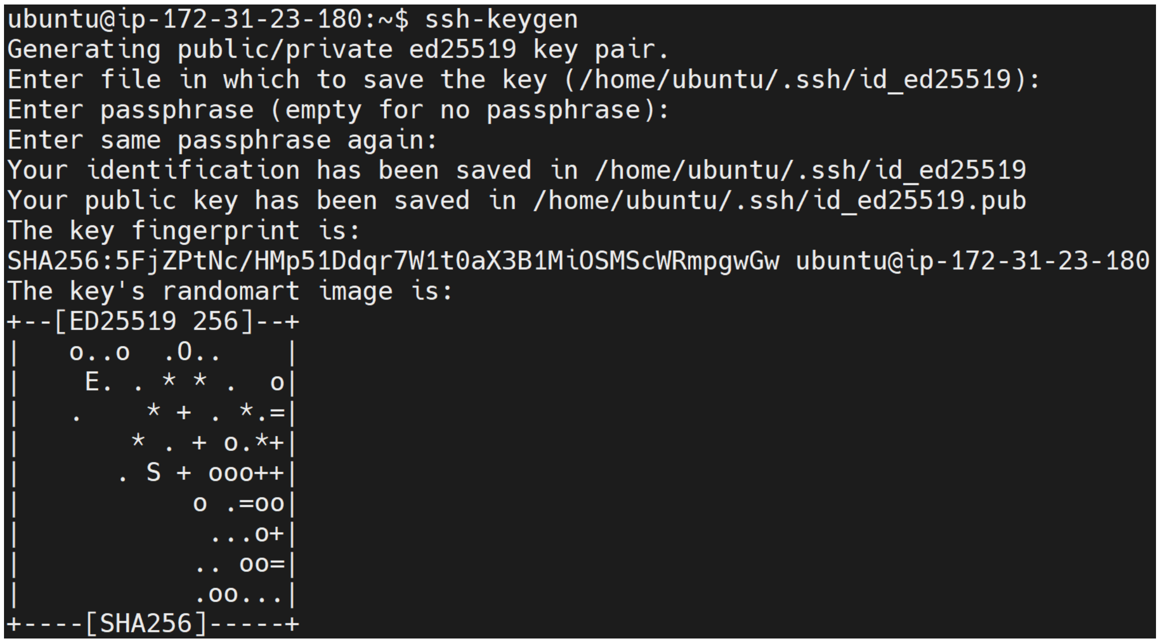
3. Verify .war file in target folder.



```
ubuntu@ip-172-31-23-180:~/aws-rds-java/target$ ls
LoginWebApp  LoginWebApp.war  maven-archiver
```

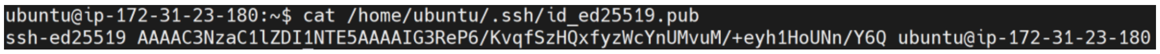
Step 7: Transfer Artifact to Deploy Server

1. Generate SSH key: ssh-keygen

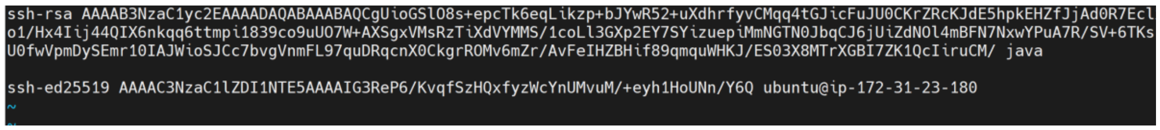


```
ubuntu@ip-172-31-23-180:~$ ssh-keygen
Generating public/private ed25519 key pair.
Enter file in which to save the key (/home/ubuntu/.ssh/id_ed25519):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/ubuntu/.ssh/id_ed25519
Your public key has been saved in /home/ubuntu/.ssh/id_ed25519.pub
The key fingerprint is:
SHA256:5FjZPtNc/HMp51Ddqr7W1t0aX3B1MiOSMScWRmpgwGw ubuntu@ip-172-31-23-180
The key's randomart image is:
+--[ED25519 256]--+
|    o..o  .O..   |
|     E. . * * . o|
|   .    * + . *.=|
|       * . + o.*+|
|      . S + ooo++|
|           o .=oo|
|            ...o+|
|           .. oo=|
|           .oo...|
+----[SHA256]-----+
```

2. Copy public key to deploy server authorized_keys.

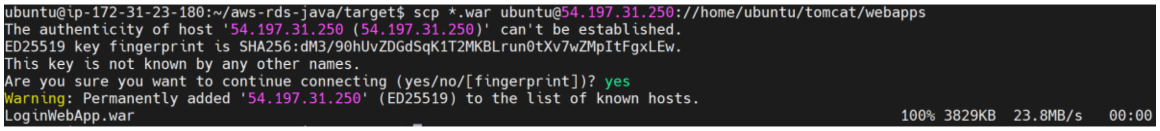


```
ubuntu@ip-172-31-23-180:~$ cat /home/ubuntu/.ssh/id_ed25519.pub
ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIG3ReP6/KvqfSzHQxfyzWcYnUMvuM/+eyh1HoUNn/Y6Q ubuntu@ip-172-31-23-180
```



```
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQCgUioGSlO8s+epcTk6eqLikzp+bJYwR52+uXdhrfyvCMqq4tGJicFuJU0CKrZRcKJdE5hpkEHZfJjAd0R7Ecl
o1/Hx4Iij44QIX6nkqq6ttmpi1839co9uUO7W+AXSgxVMsRzTiXdVYMMS/1coLl3GXp2EY7SYizuepiMmNGTN0JbqCJ6jUiZdNOl4mBFN7NxwYPuA7R/SV+6TKs
U0fwVpmDySEmr10IAJWioSJCc7bvgVnmFL97quDRqcnX0CkgrROMv6mZr/AvFeIHZBHif89qmquWHKJ/ES03X8MTrXGBI7ZK1QcIiruCM/ java

ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAIG3ReP6/KvqfSzHQxfyzWcYnUMvuM/+eyh1HoUNn/Y6Q ubuntu@ip-172-31-23-180
~
~
```

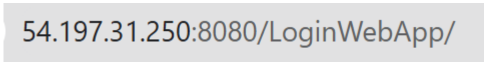
3. Transfer artifact:
   scp target/<artifact-name>.war ubuntu@<deploy-server-private-ip>:/home/ubuntu/tomcat/webapps/



```
ubuntu@ip-172-31-23-180:~/aws-rds-java/target$ scp *.war ubuntu@54.197.31.250://home/ubuntu/tomcat/webapps
The authenticity of host '54.197.31.250 (54.197.31.250)' can't be established.
ED25519 key fingerprint is SHA256:dM3/90hUvZDGdSqK1T2MKBLrun0tXv7wZMpItFgxLEw.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '54.197.31.250' (ED25519) to the list of known hosts.
LoginWebApp.war                                                                                       100% 3829KB  23.8MB/s   00:00
```

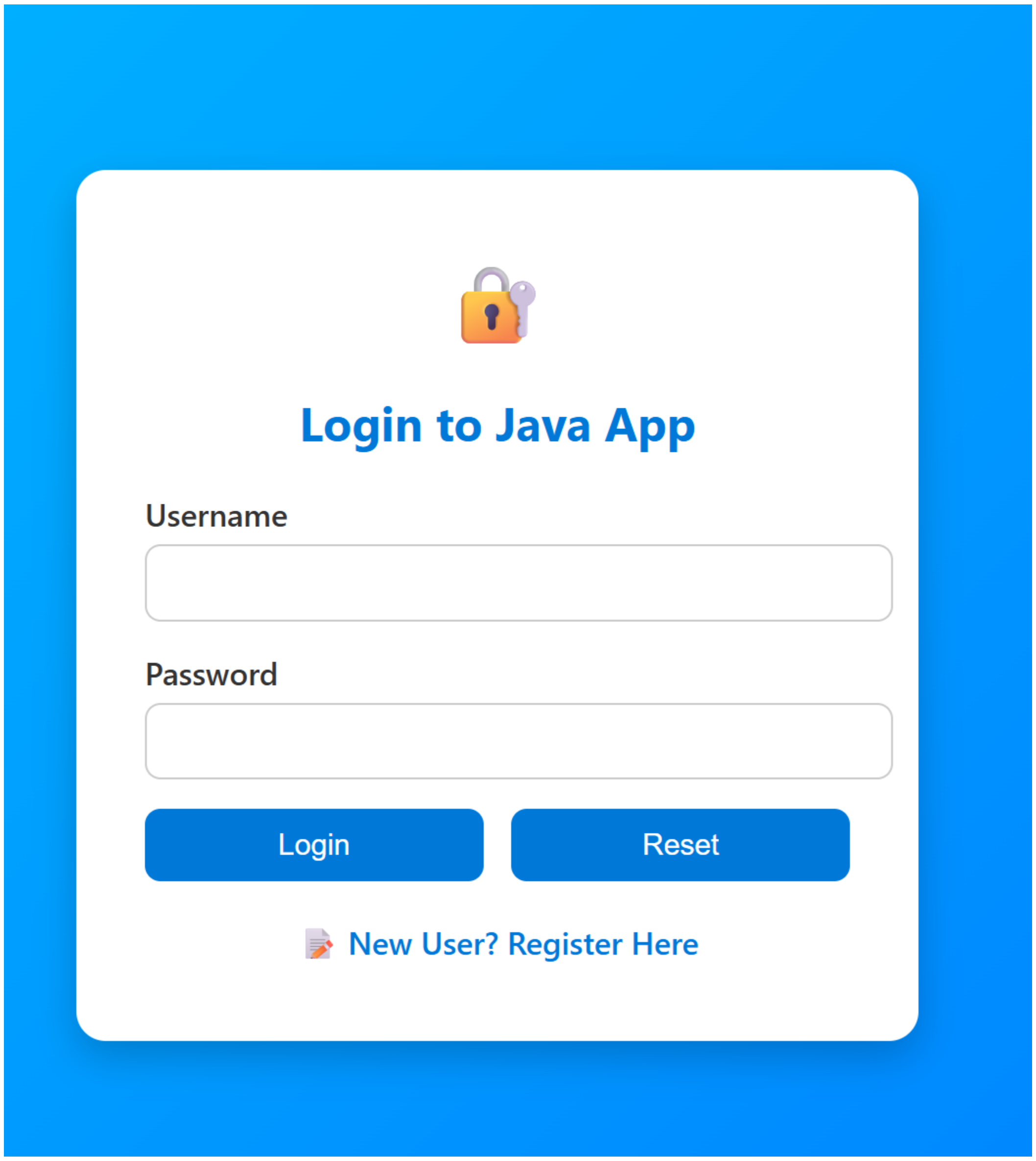
Step 8: Access the Application

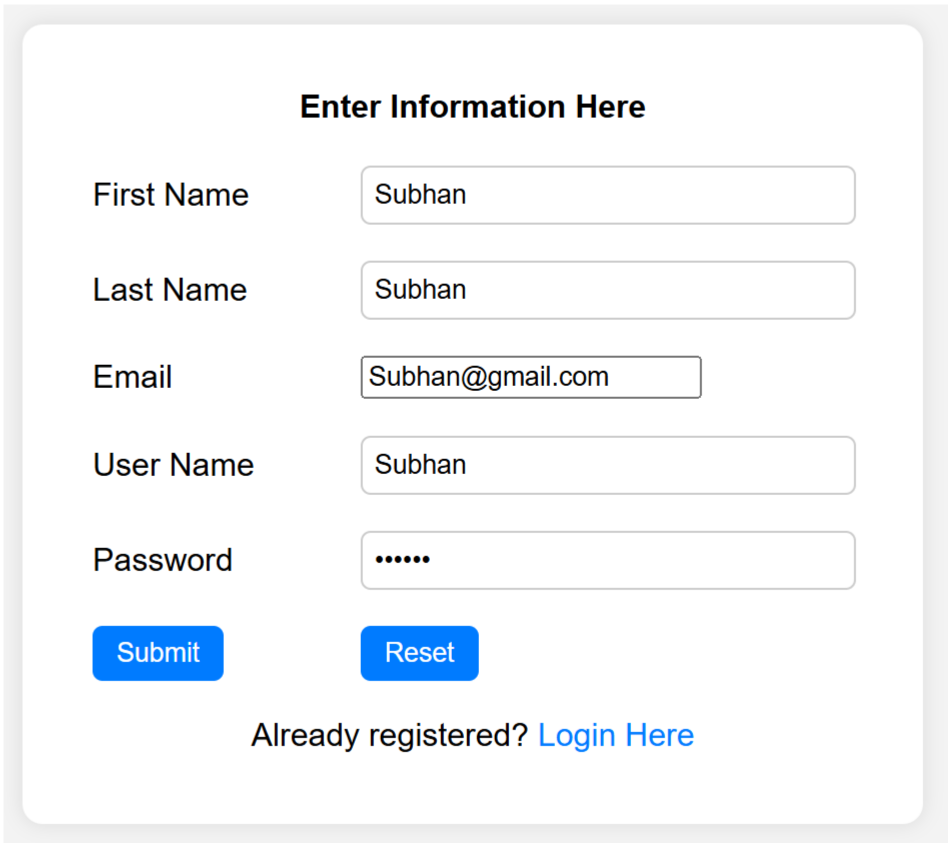
1. Open browser: http://<deploy-server-public-ip>:8080/<artifact-name>



54.197.31.250:8080/LoginWebApp/

2. Register new user → check success message.

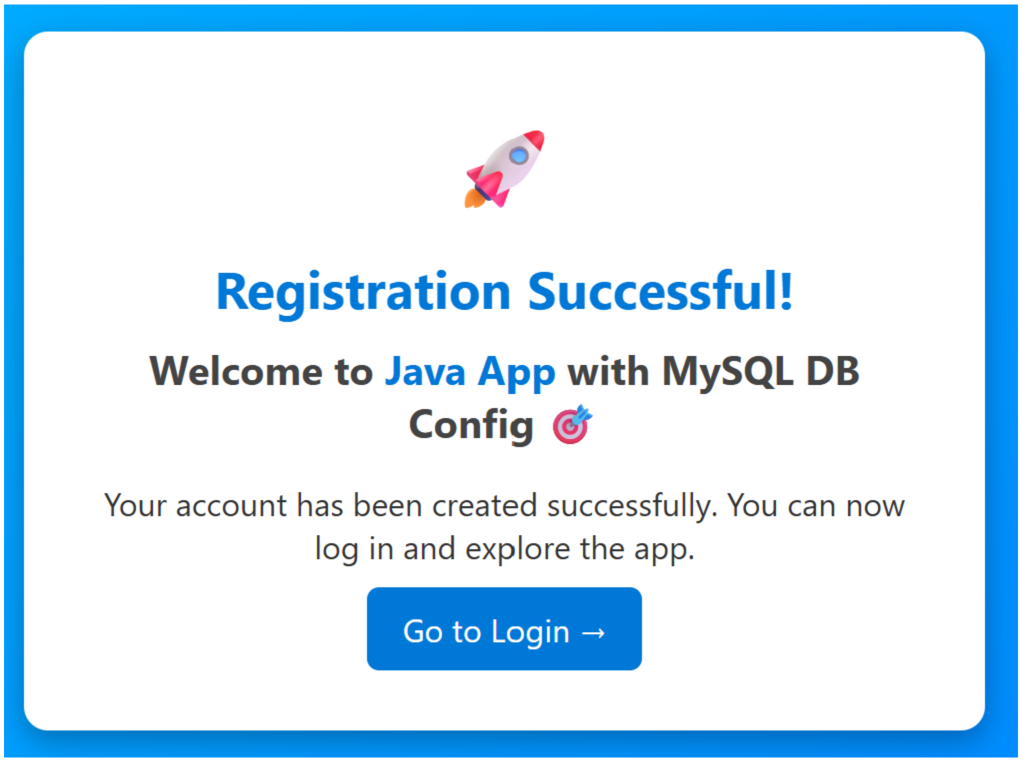




Enter Information Here

First Name	Subhan
Last Name	Subhan
Email	Subhan@gmail.com
User Name	Subhan
Password	••••••

Already registered? Login Here

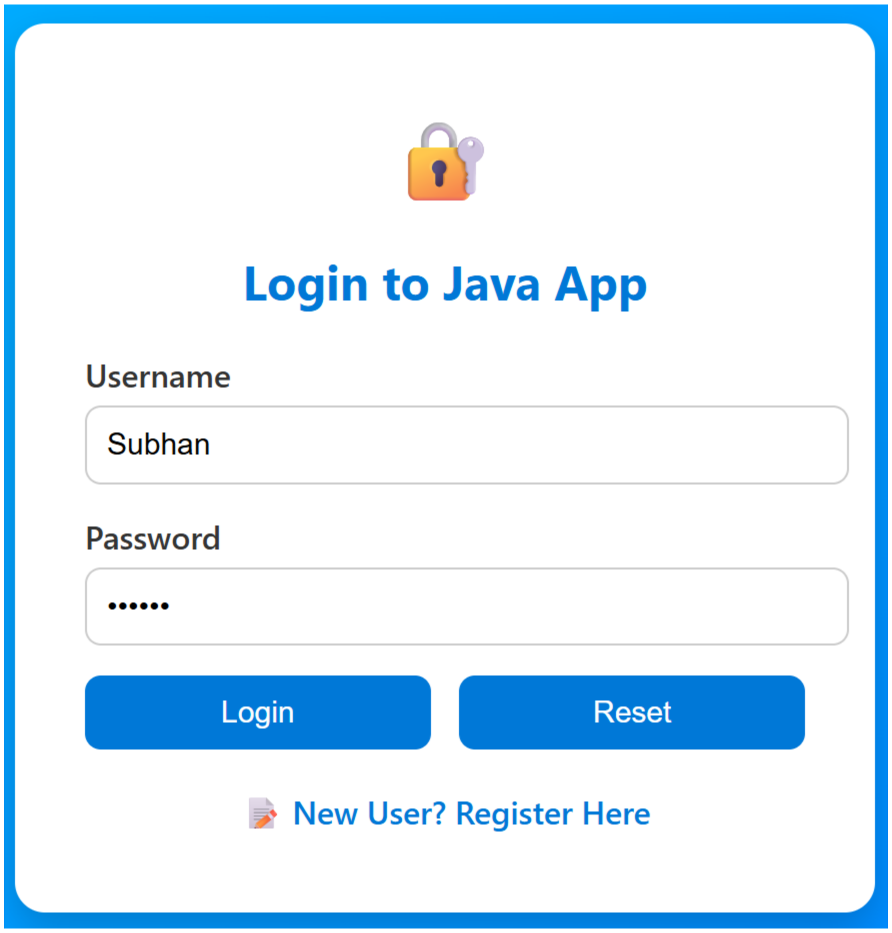


Registration Successful!

Welcome to Java App with MySQL DB Config 🎯

Your account has been created successfully. You can now log in and explore the app.

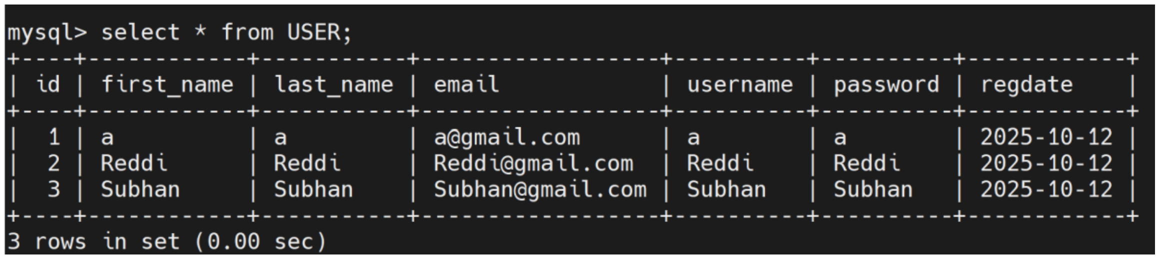
3. Login → verify welcome message.



Welcome Subhan Log out

Step 9: Verify Data in Database

1. Connect to MySQL: mysql -h <db-server-private-ip> -u appuser -p
2. View users: USE jwt; SELECT * FROM USER;



```
mysql> select * from USER;
+----+------------+-----------+------------------+----------+----------+------------+
| id | first_name | last_name | email            | username | password | regdate    |
+----+------------+-----------+------------------+----------+----------+------------+
|  1 | a          | a         | a@gmail.com      | a        | a        | 2025-10-12 |
|  2 | Reddi      | Reddi     | Reddi@gmail.com  | Reddi    | Reddi    | 2025-10-12 |
|  3 | Subhan     | Subhan    | Subhan@gmail.com | Subhan   | Subhan   | 2025-10-12 |
+----+------------+-----------+------------------+----------+----------+------------+
3 rows in set (0.00 sec)
```