Java Application Deployment using 3 Servers

This document provides step-by-step instructions for deploying a Java-based web application using three servers:

1. **Build Server** - For compiling and building the Java code using Maven.
2. **Deployment Server** - For running the built artifact (WAR file) using Apache Tomcat.
3. **Database Server** - For hosting and managing the application database (MySQL).

Artifacts will be securely copied between servers using the `scp` command.

Step 1: Build Server Setup (Java + Maven)

1. Launch a Linux server (e.g., Ubuntu) for the build process.
2. Install Java Development Kit (JDK):

```
sudo apt update
sudo apt install openjdk-17-jdk -y
java -version
```

3. Install Maven:

```
sudo apt install maven -y
mvn -version
```

4. Clone your Java project repository:

```
git clone <your_repo_url>
cd <project_folder>
```

```
[INFO] Packaging webapp
[INFO] Assembling webapp [webapp] in [/home/ubuntu/JavaWebCal/target/webapp]
[INFO] Processing war project
[INFO] Copying webapp resources [/home/ubuntu/JavaWebCal/src/main/webapp]
[INFO] Building war: /home/ubuntu/JavaWebCal/target/webapp.war
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time:  20.638 s
[INFO] Finished at: 2025-10-07T06:35:36Z
[INFO] ------------------------------------------------------------------------
ubuntu@ip-172-31-27-0:~/JavaWebCal$
```

5. Build the project and generate the WAR/JAR file:

```
mvn clean package
```

6. Verify that the artifact (e.g., `target/app.war`) is created successfully.

Step 2: Transfer Artifact using SCP

Use the `scp` command to securely transfer the WAR/JAR file from the Build Server to the Deployment Server.

Example:

scp /home/ubuntu/project/target/app.war
ubuntu@<deployment_server_ip>:/home/ubuntu/

Note: Ensure that SSH keys or proper credentials are configured to allow secure copy between servers.

```
ubuntu@ip-172-31-33-236:~/aws-rds-java$ cat ls.pub
ssh-ed25519 AAAAC3NzaC1lZDI1NTE5AAAAINYSP5zfjd9yKu/I96ieLKQ8UdUKLLq78TGgpW/xaiJ7 ubuntu@ip-1
72-31-33-236
ubuntu@ip-172-31-33-236:~/aws-rds-java$
```

Step 3: Deployment Server Setup (Java + Tomcat)

1. Launch another Linux server for deployment.
2. Install Java:

```
sudo apt update
sudo apt install openjdk-17-jdk -y
java -version
```

3. Download and install Apache Tomcat:

```
wget https://dlcdn.apache.org/tomcat/tomcat-9/v9.0.91/bin/apache-tomcat-
9.0.91.tar.gz
tar -xvzf apache-tomcat-9.0.91.tar.gz
mv apache-tomcat-9.0.91 tomcat
```

```
ubuntu@ip-172-31-42-202:~$ ls
apache-tomcat-9.0.110.tar.gz
ubuntu@ip-172-31-42-202:~$ tar -xvf apache-tomcat-9.0.110.tar.gz
apache-tomcat-9.0.110/conf/
apache-tomcat-9.0.110/conf/catalina.policy
apache-tomcat-9.0.110/conf/catalina.properties
```

4. Deploy the WAR file to Tomcat:

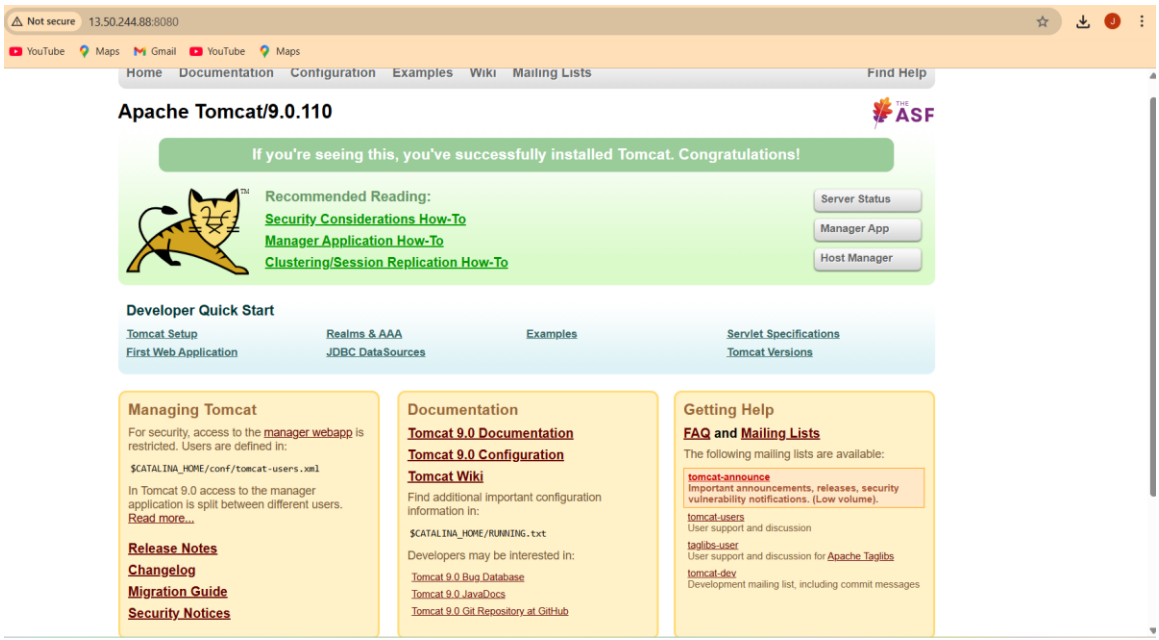
mv /home/ubuntu/app.war /home/ubuntu/tomcat9/webapps/

```
ubuntu@ip-172-31-33-236:~$ scp /home/ubuntu/aws-rds-java/target/*.war ubuntu@13.60.228.88:/h
ome/ubuntu/tomcat/webapps
LoginWebApp.war                                   100% 3829KB  22.3MB/s   00:00
```

5. Start Tomcat:

cd /home/ubuntu/tomcat9/bin
./startup.sh

6. Access the application in a browser:
`http://<deployment_server_ip>:8080/app`



Step 4: Database Server Setup (MySQL)

1. Launch a third Linux server for the database.
2. Install MySQL Server:
```bash
sudo apt update
sudo apt install mysql-server -y
sudo systemctl start mysql
sudo systemctl enable mysql
```
3. Secure MySQL installation:
```bash
```

sudo mysql_secure_installation
```

4. Login to MySQL and create a database and user:
```bash
mysql -u root -p
CREATE DATABASE appdb;
CREATE USER 'appuser'@'%' IDENTIFIED BY 'password';
GRANT ALL PRIVILEGES ON appdb.* TO 'appuser'@'%';
FLUSH PRIVILEGES;
```
5. Edit MySQL config to allow remote access:
```bash
sudo vim /etc/mysql/mysql.conf.d/mysqld.cnf
```
Change `bind-address` from `127.0.0.1` to `0.0.0.0`.
6. Restart MySQL:

sudo systemctl restart mysql

Step 5: Connect Application to Database

1. Update your application's database configuration file (e.g., `application.properties` or `context.xml`) with the following details:
properties
spring.datasource.url=jdbc:mysql://<database_server_ip>:3306/appdb
spring.datasource.username=appuser
spring.datasource.password=password

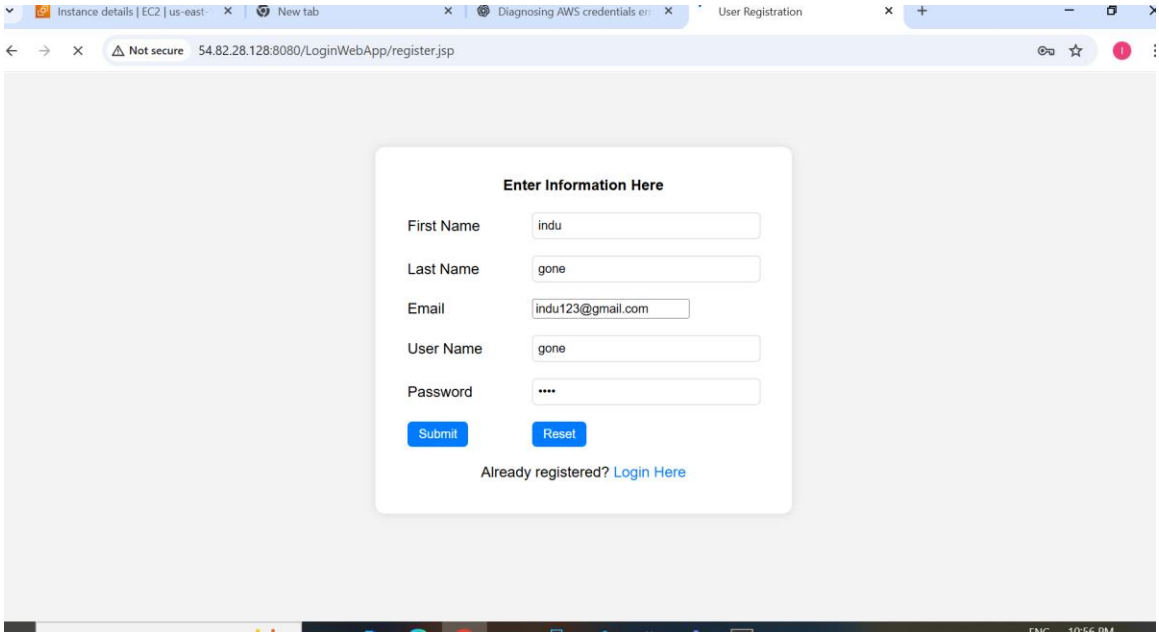
2. Redeploy the application if necessary.
3. Verify database connectivity from the Deployment Server:

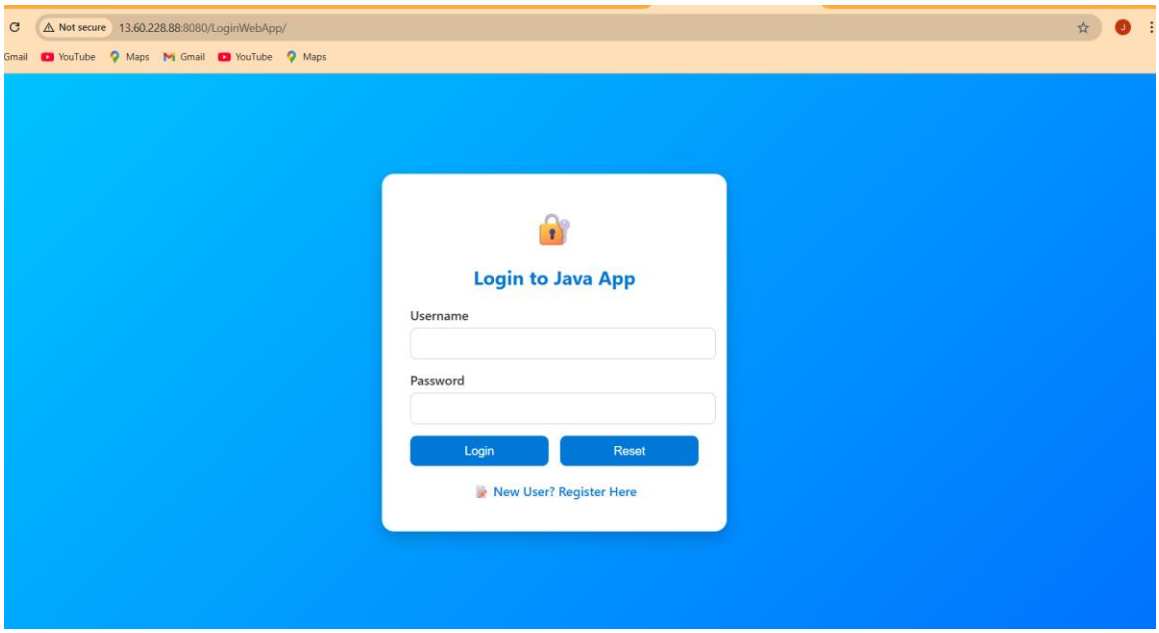
mysql -h <database_server_ip> -u appuser -p

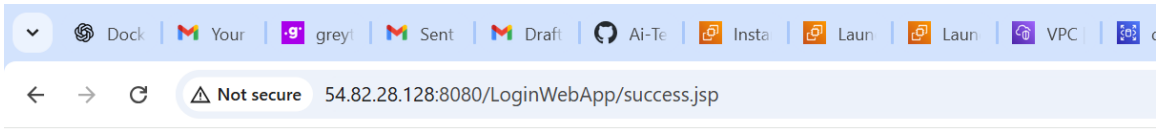
Step 6: Verification

1. Verify that the Tomcat service is running:

2. Access the application from your browse.
```


Instance details | EC2 | us-east-
New tab
Diagnosing AWS credentials err
User Registration
Not secure
54.82.28.128:8080/LoginWebApp/register.jsp
Enter Information Here
First Name
indu
Last Name
gone
Email
indu123@gmail.com
User Name
gone
Password
Submit
Reset
Already registered? Login Here


Not secure
13.60.228.88:8080/LoginWebApp/
Gmail
YouTube
Maps
Gmail
YouTube
Maps
Login to Java App
Username
Password
Login
Reset
New User? Register Here



Welcome indu Log out