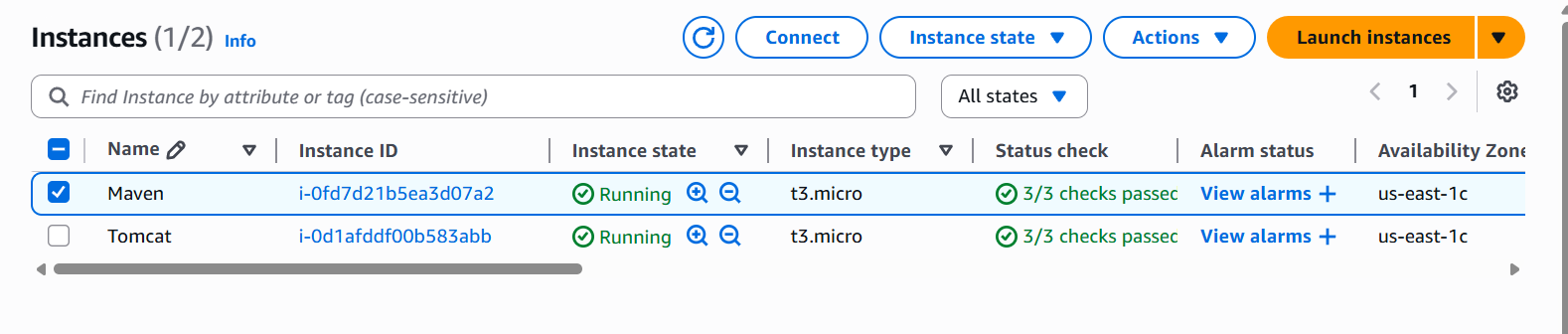
**Step 1: Launch Two AWS EC2 Instances**

Launch two EC2 instances from your AWS console:

* **Instance 1:** Maven (Build Server)
* **Instance 2:** Tomcat Server (Deploy Server)

After launching both instances, connect to them using any terminal (e.g., SSH) and start the configuration process.

**Note:** Since we are building a Java-based project, we need a build tool like **Maven**. Install **Java** first, as Maven automatically configures itself with the installed Java version to ensure compatibility.

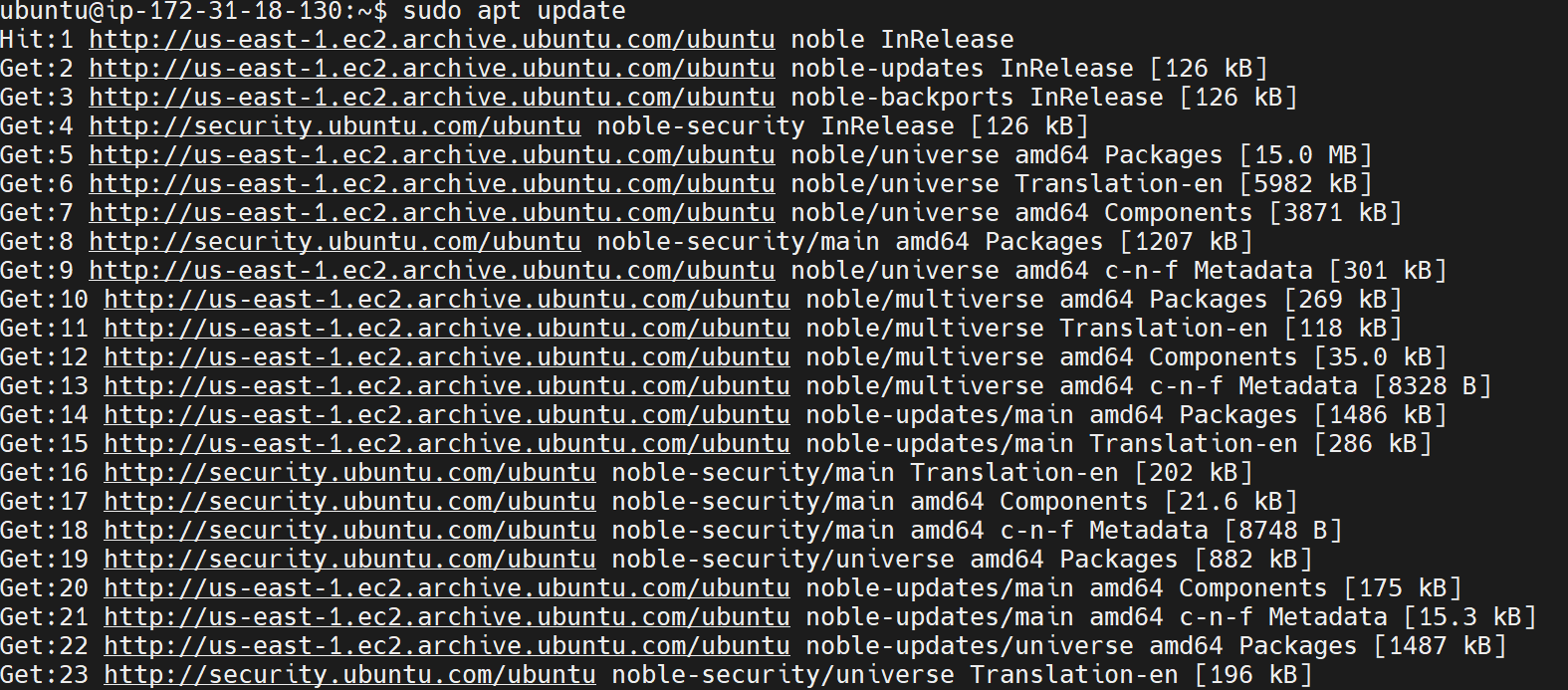
After launching both instances, connect to them using any terminal (e.g., SSH) and start the configuration process.

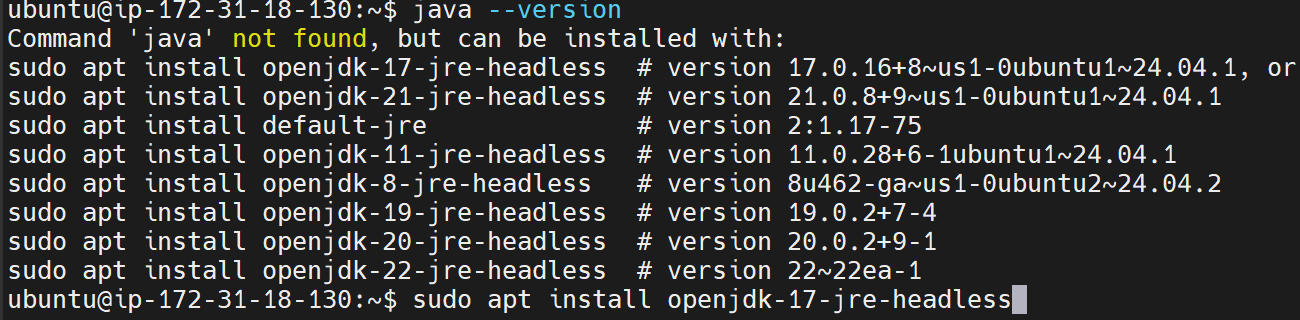
**Note:** Since we are building a Java-based project, we need a build tool like **Maven**. Install **Java** first, as Maven automatically configures itself with the installed Java version to ensure compatibility.  
**Step 2: Setting Up the Maven Server (Build Server)**

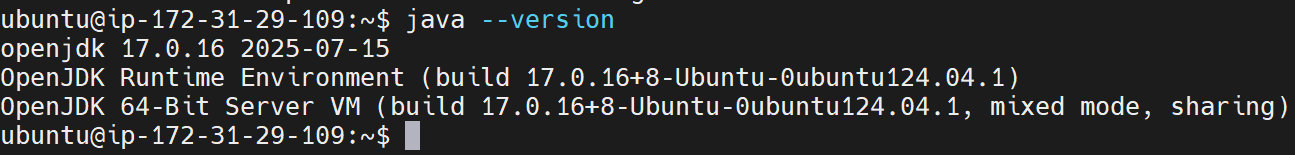
**2.1 Update the Server and Install Java**

**First, update your server and install Java (OpenJDK 17):**

Commands:  
sudo apt update



sudo apt install openjdk-17-jdk  
  
Verify Java installation:  
  
Command:

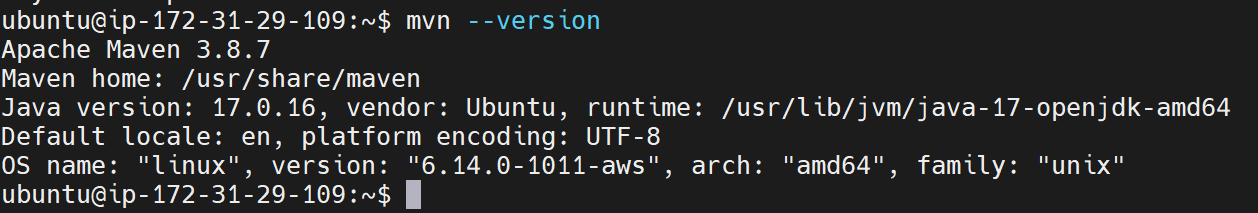
java --version   
  
**2.2 Install Maven**

Install Maven if it's not already installed:

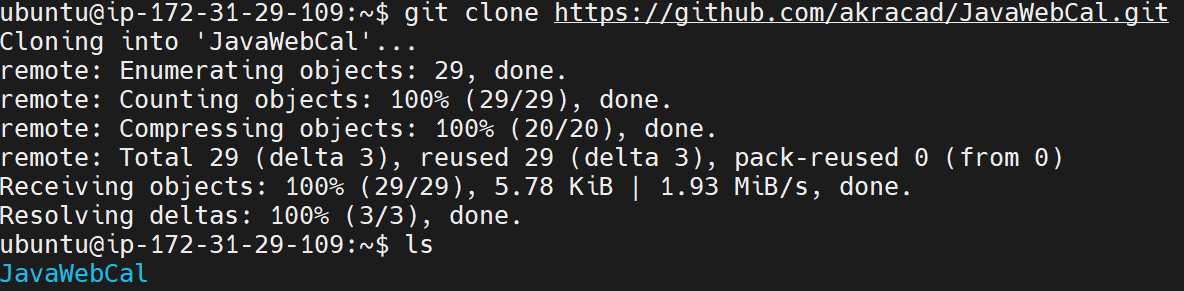
Commands:

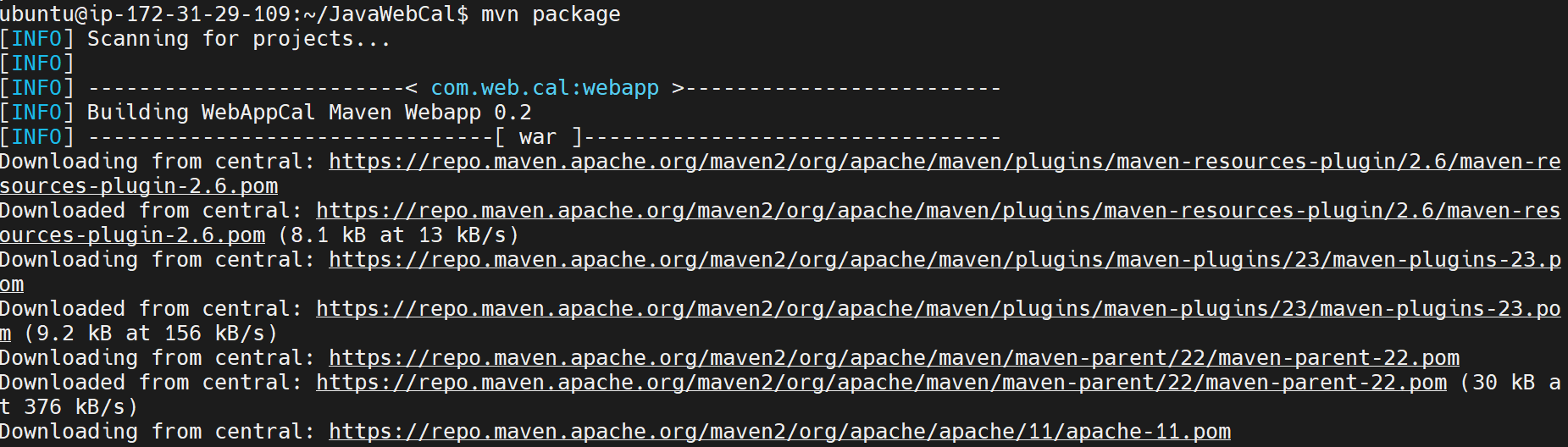
sudo apt install maven -y  
  
Verify Maven installation

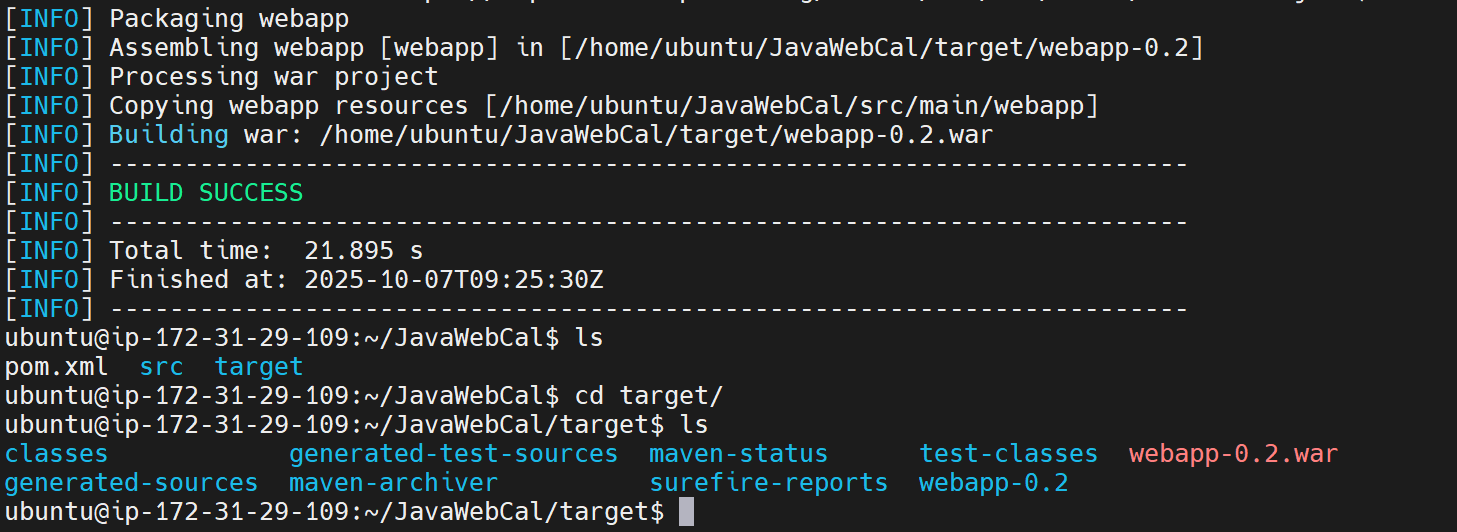
maven --version

  
  
**2.3 Clone the Repository**

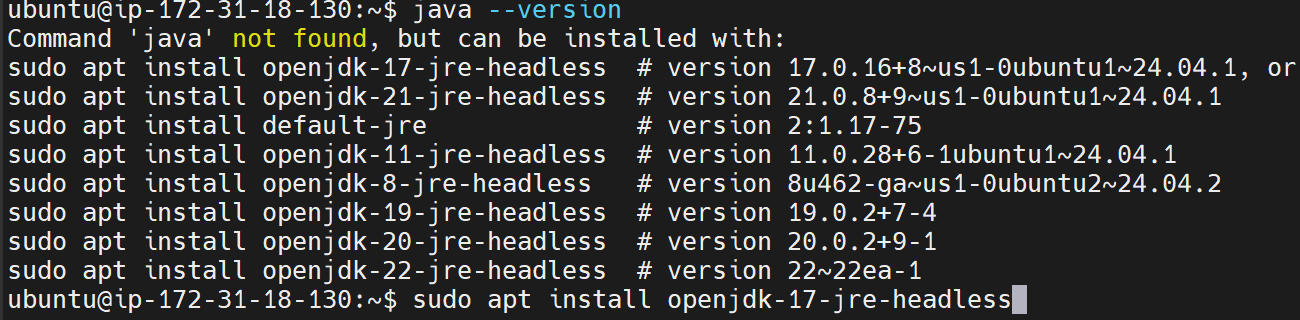
Clone your Java project repository from GitHub:

git clone <repo url> ****

After cloning the repository, use the ls command to list the files, and then move into the directory that contains the Java source code  
  
**2.4 Build the Artifact**  
Then generate the artifact with Command **mvn package**   
  
  
You will see **"BUILD SUCCESS"** in your terminal. Then, list the files using the ls command. You will notice a new **target** folder has been generated, and inside that folder, the **.war** file is present.

  
  
You have successfully generated the Artifact   
  
**Step 3: Setting Up the Tomcat Server (Deploy Server)**

3.1 Update Server and Install Java

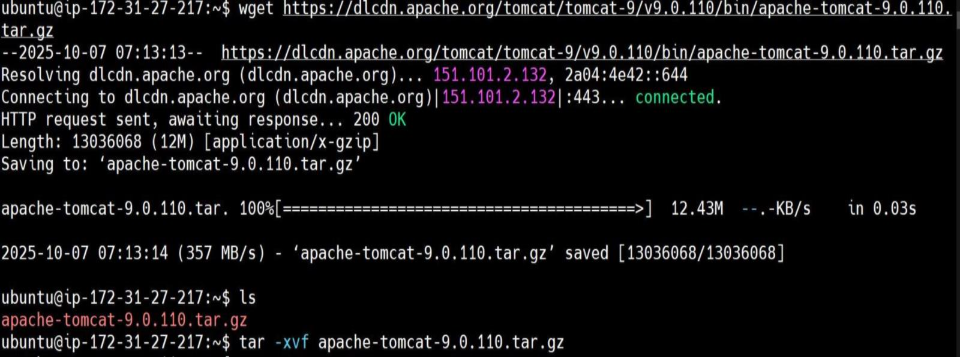
Update the server and install the same Java version (OpenJDK 17):  
  
  
  
**3.2 Install Apache Tomcat**

Go to Apache tomcat official website

Copy the tar extension file url link

Download it using wget command followed by the url link

After downloading tar file, extract the tar file using “tar -xvf  apache-tomcat-9.0.110.tar.gz” command



**3.3 Configure Tomcat**

a) Configure webapps/host-manager/META-INF/context.xml

Edit the file:  
After opening the context.xml file, locate the following lines

<Valve className="org.apache.catalina.valves.RemoteAddrValve"

  allow="127\.\d+\.\d+\.\d+|::1|0:0:0:0:0:0:0:1" and comment it out by enclosing it within <!-- and --> tags.



b) Configure webapps/manager/META-INF/context.xml on the Deploy Server.

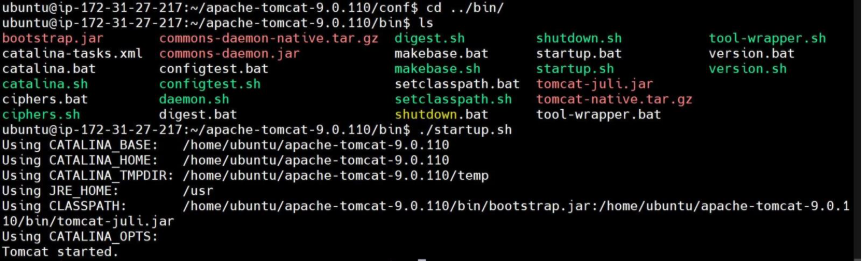
This step is the same as the previous one, but here you need to modify the context.xml file located in webapps/manager/META-INF/.

c) Configure conf/tomcat-users.xml

After opening file, locate for roles lines which are pre-defined and commented, those lines should be uncommented and passwords should be changed.

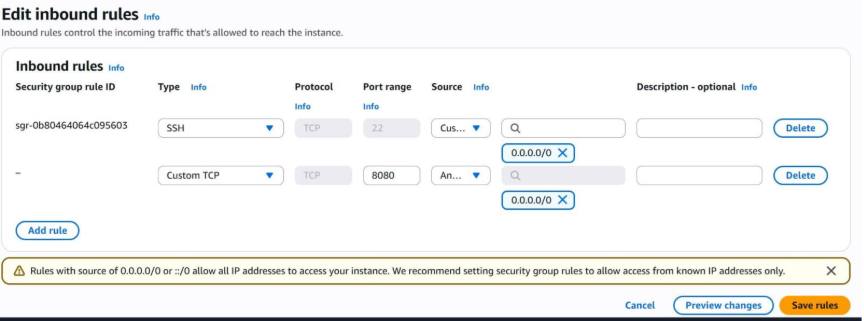


**3.4 Start the Tomcat Service**Navigate to the bin directory and start Tomcat:



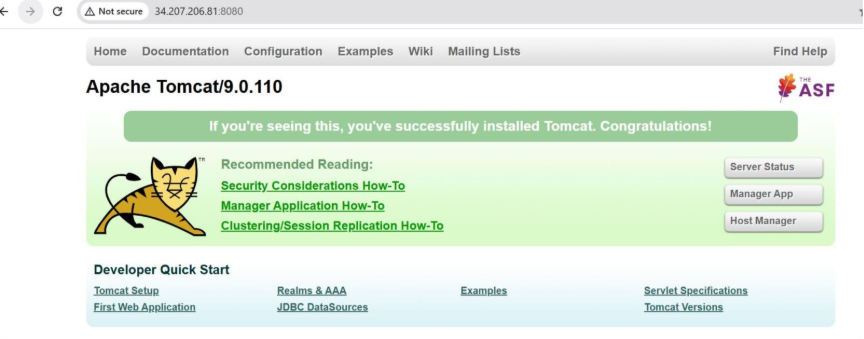
**3.5 Allow Port 8080 in Security Group**

In the AWS Management Console, update the **Inbound Rules** of your Tomcat instance’s **Security Group** to allow **port 8080 (TCP)**.



**3.6 Verify Tomcat Installation**

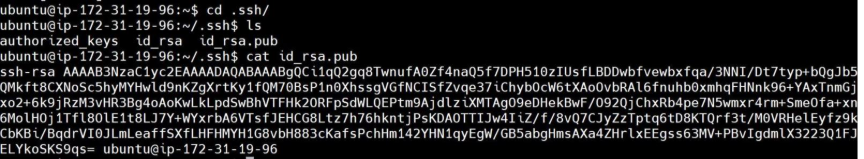
Access the Tomcat web interface in your browser using the public IP:



**Step 4: Transfer the .war File to the Tomcat Server**

**4.1 Set Up SSH Key Authentication**

On the Build Server, generate an SSH key pair:  
Copy the public key:



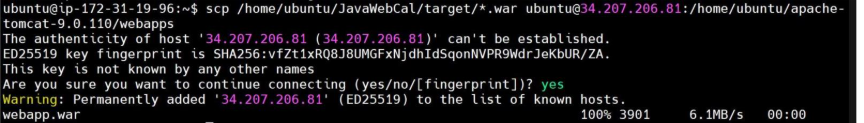
Paste the copied public keys of build server to authorized keys (by changing to .ssh directory) of Deploy server



**4.2 Copy the .war File Using SCP**

Use scp to copy the .war file from the Build Server to the Deploy Server’s webapps directory:

scp target/<your-artifact>.war ubuntu@<deploy-server-ip>:/opt/tomcat/webapps/

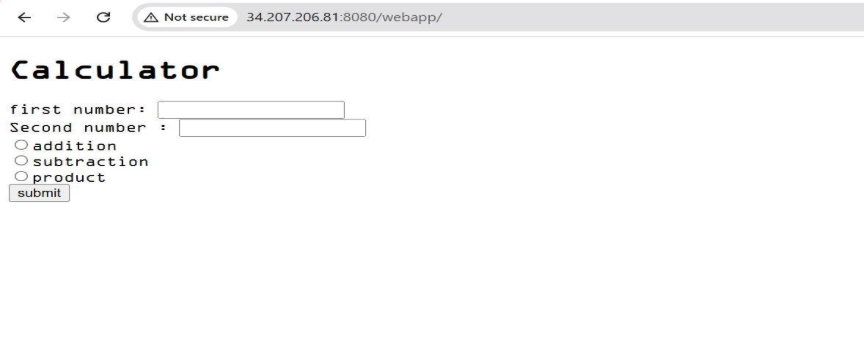


Verify that the .war file has been copied to /opt/tomcat/webapps/ on the Deploy Server



**4.3 Access the Deployed Application**

Restart Tomcat (optional) and access your deployed application in the browser:

  
Your Java application has been successfully deployed on the Apache Tomcat server!