

Deploy a Simple Web Application Using AWS EC2 and RDS

In this tutorial a simple Java web application from Github will be deployed on an EC2 instance and RDS database.

1. Log in to the AWS console with your username and password. Follow the steps given in the *Creating a Linux EC2 instance and connect pdf* and create a Linux EC2 instance. This will be used to host our web application.
2. Follow the steps in the Create a *MySQL Database Instance in RDS and connect pdf* file to create a RDS database instance. This will be used to host our web application database. Note down the username and password created for the RDS instance. Also note down the end point to connect to RDS instance.
3. Install Tomcat Server on the EC2 instance.
Connect to your EC2 instance using putty application. Run following commands.

```
sudo apt update -y

sudo apt install default-jdk -y

java -version
```

Now we will install Tomcat server to host our Java web application.

```
cd /tmp

sudo wget https://dlcdn.apache.org/tomcat/tomcat-9/v9.0.94/bin/apache-
tomcat-9.0.94.tar.gz

sudo mkdir /opt/tomcat

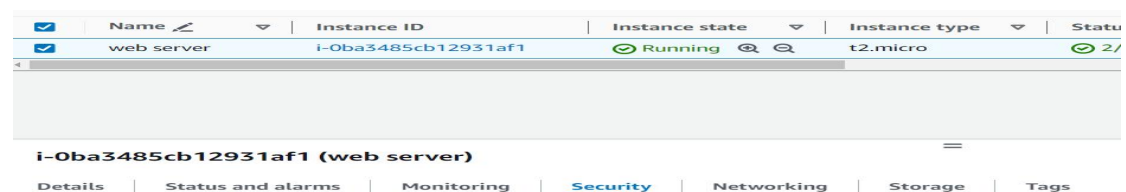
sudo tar xzvf /tmp/apache-tomcat-9.0.94.tar.gz -C /opt/tomcat
--strip-components=1

cd /opt/tomcat
```

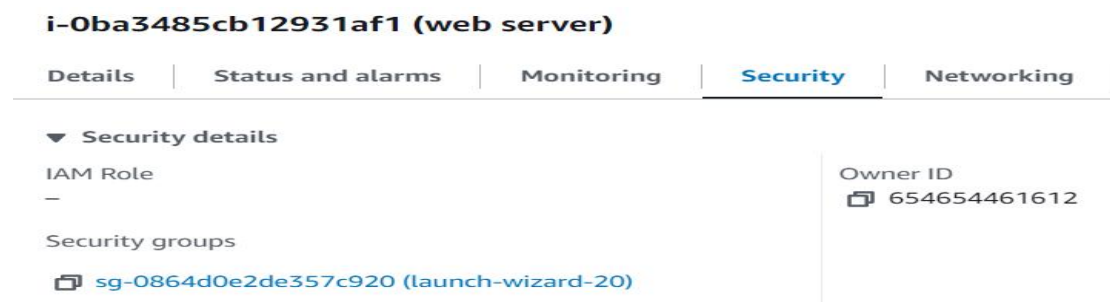
Start the tomcat server to check if it is working.

```
sudo sh bin/startup.sh
```

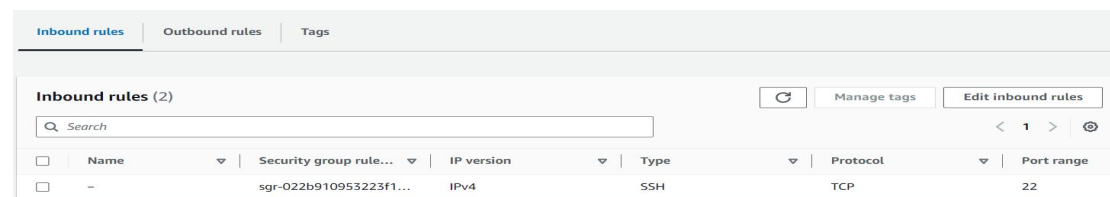
Now go to the EC2 console. Select your EC2 instance. Select Security option as shown below.



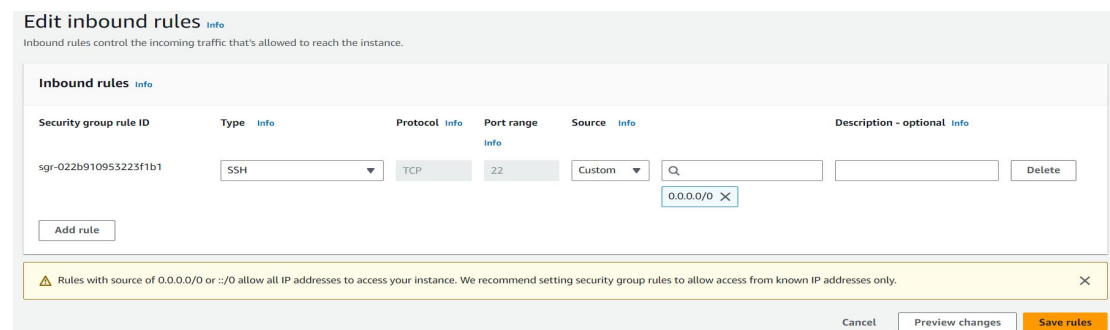
In the Security tab below click on the name of the security group shown in the blue colour as shown below.



The following page is displayed.

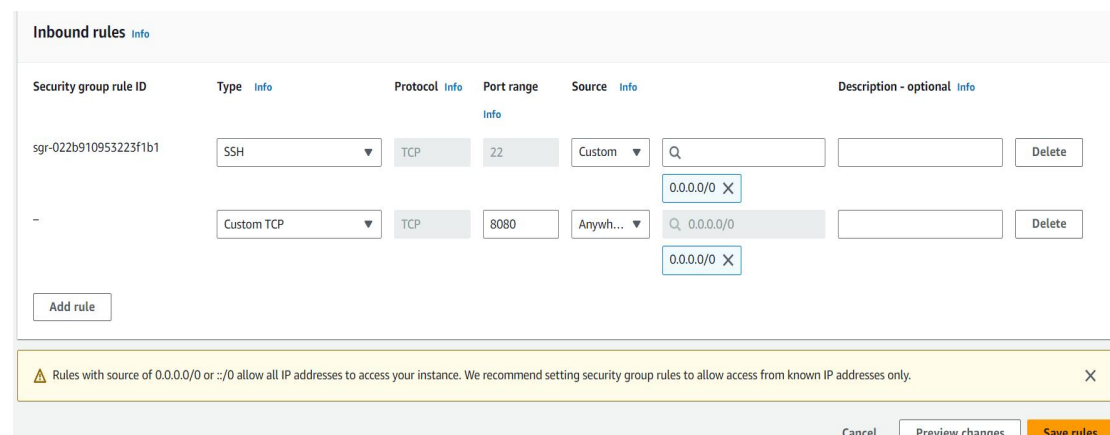


Click Edit Inbound Rules button. Following Window opens.



Click Add Rule button.

A new rule will be added below the existing SSH rule as shown below.



In the port range box type 8080. This is the port used by the Tomcat server. Click In the empty box in the source field. Select 0.0.0.0/0 option. Then click Save Rules button.

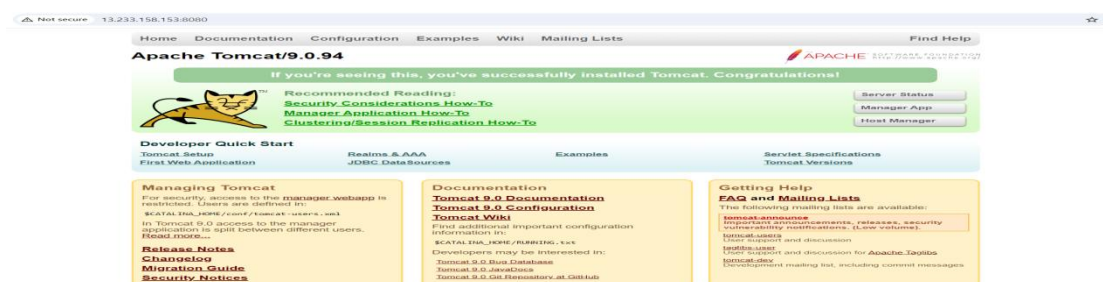
Now we will try to access the Tomcat Server using EC2 instance public IP.

Open a browser on your local computer. Type following URL.

<http://your-ec2-public-ip:8080>

Make sure you replace the actual public IP address of your EC2 instance.

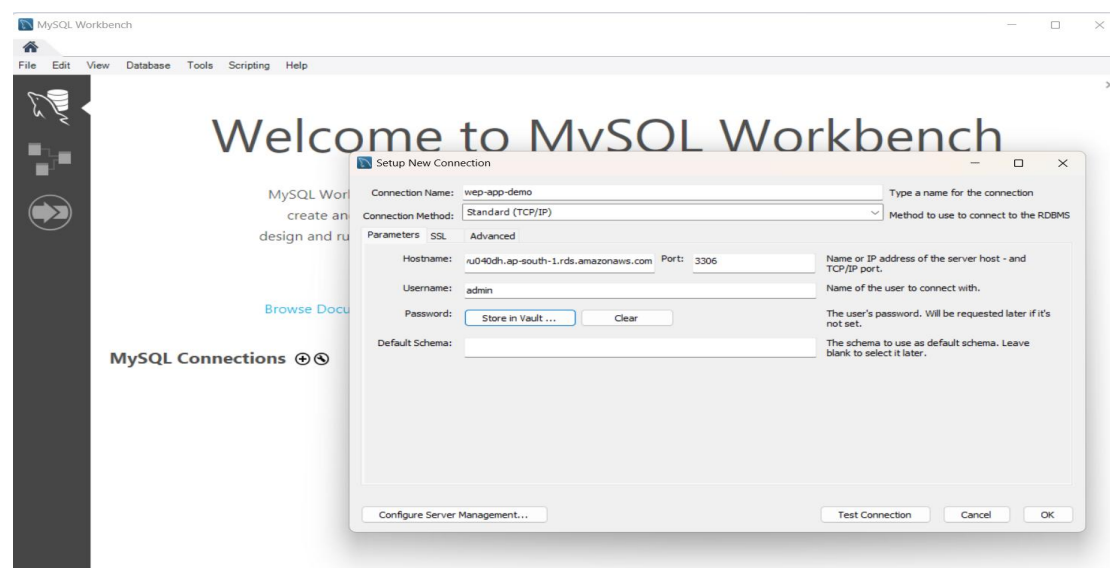
If all the steps are properly performed, you should be able to get following page in the browser.



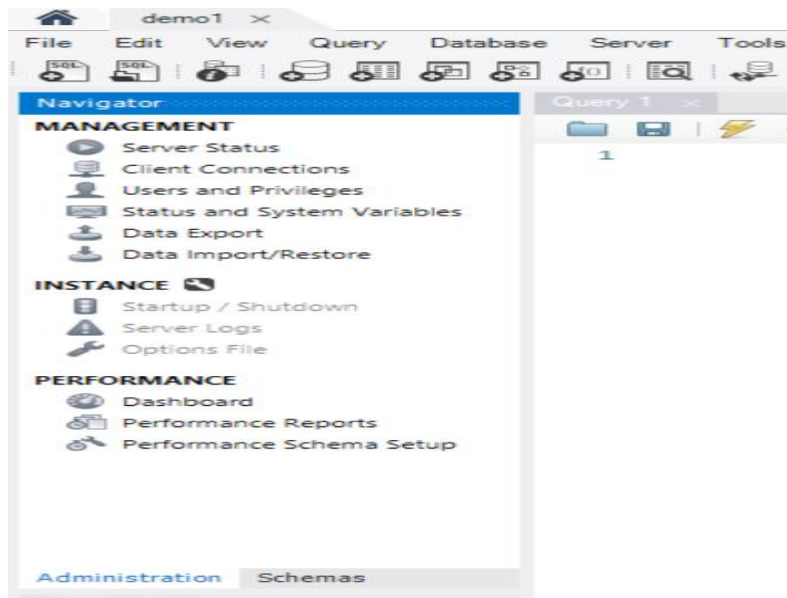
Thus the tomcat server is working properly.

4. Create the required table in the database.

Open MySQL Workbench. If the connection is not saved earlier, click the plus sign in front of MySQL Connections option. In the window that opens, provide details of your RDS database. Click Test Connection button. If it is successful, Connect to the RDS database.



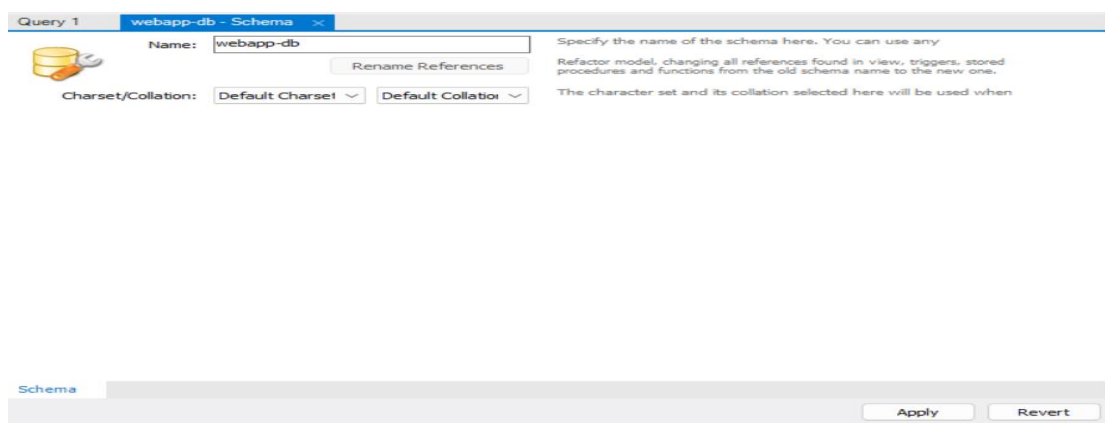
Once you connect to the database, click schemas option on the left side next to Administration option(appears in blue colour)



Click the following option to add a new schema. The option appears below the File menu bar.



Provide a name for the schema and click apply button.



Again click Apply button and click Finish to create the new schema.

The new schema will be shown on the left side.



Expand the new created schema. Tables option will be displayed.

[illegible]

Click apply button to create the table. Click Apply button again. This will create the table.

5. Download the application from Github and configure

Clone the repository of the application using following command.

```
git clone https://github.com/santoshkatageri/Java-Mysql-Simple-Login-Web-application.git
```

Go to the directory that contains the actual code files.

```
cd Java-Mysql-Simple-Login-Web-application/src/main/webapp
```

Edit the login.jsp file.

```
nano login.jsp
```

The following line will be present.

```
class.forName("com.mysql.jdbc.Driver");  
Connection con = DriverManager.getConnection("jdbc:mysql://mysql:3306/sample", "root", "Qwerty@12345");
```

This line provides the details for connecting to the database server. Put your RDS instance details here.

Replace mysql:3306 with Your-RDS-Endpoint:3306 .Replace sample database with actual database name (schema created above). Replace root username with admin. Replace Qwerty@12345 password with your own password.

This will look as shown below.

```
Connection con = DriverManager.getConnection("jdbc:mysql://database-1.cvy6wyu040dh.ap-south-1.rds.amazonaws.com:3306/webapp-db", "admin", "Demo_Admin24");
```

Save the file.

Now edit the userRegistration.jsp file.

```
Nano userRegistration.jsp
```

In this file also you will find the same line as above. Make the same modifications. Save the file.

6. Install Apache maven to build the war file from the source code.

```
sudo apt install maven -y
```

Once the installation is over, to verify type the following command. It will display the Maven version.

```
mvn -v
```

Then go back to main directory using

```
cd ../../..
```

Edit the pom.xml file.

```
nano pom.xml
```

Scroll down to the following section.

```
<dependency>
    <groupId>mysql</groupId>
    <artifactId>mysql-connector-java</artifactId>
    <version>5.1.30</version>
</dependency>
</dependencies>
<build>
```

Change the version from 5.1.30 to 8.0.32 as shown below.

```
<dependency>
    <groupId>mysql</groupId>
    <artifactId>mysql-connector-java</artifactId>
    <version>8.0.32</version>
</dependency>
```

Modify the build section in the file as shown below.

```
<build>
  <plugins>
    <plugin>
      <groupId>org.apache.maven.plugins</groupId>
      <artifactId>maven-war-plugin</artifactId>
      <version>3.3.2</version>
    </plugin>
  </plugins>
  <finalName>LoginWebApp</finalName>
</build>
```

Save the file.

Now build the package using following command.

```
mvn package
```

Once you get Build Success message, a war file in the target directory.

7. Deploy the war file using tomcat.

Connect to your EC2 instance using putty. Copy the war file created by maven to the Tomcat webapps directory. For this give following commands.

```
cd /opt/tomcat
```

```
sudo cp ~/Java-Mysql-Simple-Login-Web-application/target/LoginWebApp.war  
webapps
```

8. Access the webapp

Go to the following URL

<http://your-ec2-public-ip:8080>

Make sure you are getting the Tomcat server page.

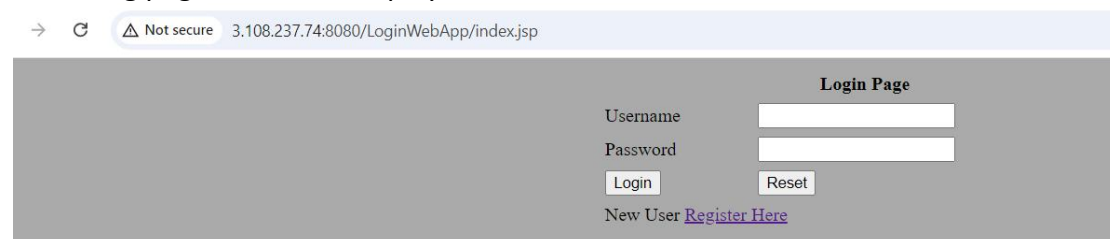
If you are not getting the page then the Tomcat Server is not running. In the putty screen give following command.

```
sudo sh bin/startup.sh
```

Then try above URL. If you are getting the Tomcat page then try following

<http://your-ec2-public-ip:8088/LoginWebAppppp>

Following page should be displayed.



→ ↻ ⚠ Not secure 3.108.237.74:8080/LoginWebApp/index.jsp

Login Page

Username

Password

New User [Register Here](#)

Try Register Here option and create a user. Then Try to login using This User.

This is how you have deployed a simple Java web application using AWS EC2 and RDS services.

#####Important #####

Please make sure you delete the EC2 instance and database instance immediately.

Clean up

Make sure you delete all the resources created for this practical.

1. Go to the RDS console. Select the database you created, click Actions button above and click delete.

Make sure you uncheck the Create final snapshot Check box. (Clear the check box).

Delete demo-db1 instance?

Are you sure you want to Delete the **demo-db1** DB Instance?

☐ **Create final snapshot?**
Determines whether a final DB Snapshot is created before the DB instance is deleted.

☐ **Retain automated backups**
Determines whether retaining automated backups for 7 days after deletion

☒ **I acknowledge that upon instance deletion, automated backups, including system snapshots and point-in-time recovery, will no longer be available.**

To confirm deletion, type *delete me* into the field

⚠ We strongly recommend taking a final snapshot before instance deletion since after your instance is deleted, automated backups will no longer be available.

Cancel **Delete**

(Do not include space after delete me. Else delete button will not be enabled)
Click delete button.

2. Delete the EC2 instance created. Go to EC2 console. In instances select your EC2 instance check box. Click Instance state and click terminate.

the navigation panel. We'll release updates continuously based on customer feedback.

Instances (1/1) Info

Filter instances

Instance state: running X Clear filters

<input checked="" type="checkbox"/>	Name	Instance ID	Instance state
<input checked="" type="checkbox"/>	-	i-0b53530471f2066ab	Running

Instance state menu:

- Stop instance
- Start instance
- Reboot instance
- Hibernate instance
- Terminate instance