

ANGULAR JAVA PROJECT



❖ Prerequisites

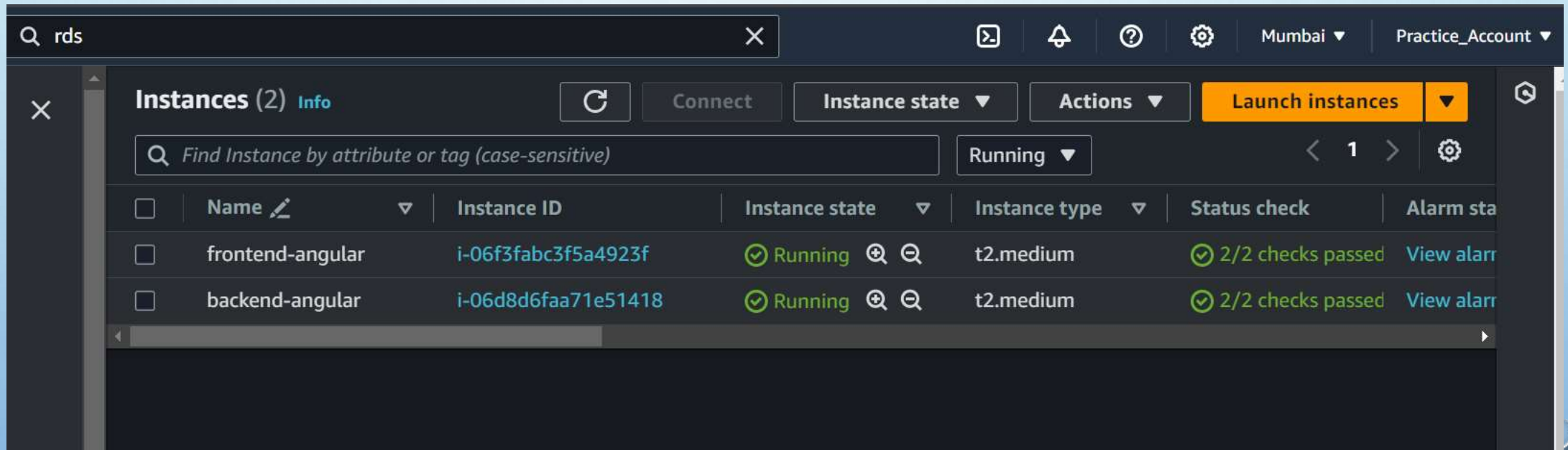
- Ubuntu Server
- Aws Account
- Admin Access

PROJECT DOCUMENT MADE BY:-

SAI PHAPALE

FOLLOW STEPS :-

- Login in into the AWS account Navigate to EC2 service
- Launch two instance with same Configuration named it as frontend-angular & backend-angular

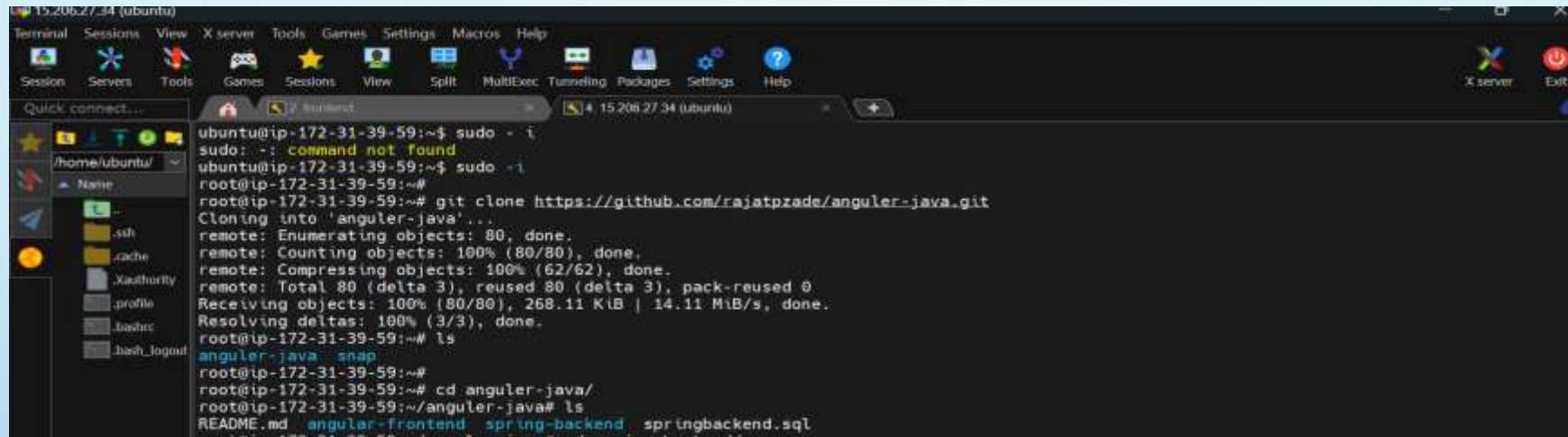


The screenshot displays the AWS Management Console interface for the EC2 service. At the top, there's a search bar with 'rds' entered. Below it, the 'Instances (2)' section is active, showing a list of two EC2 instances. The instances are 'frontend-angular' and 'backend-angular', both with 't2.medium' instance types and 'Running' states. The 'Status check' column shows '2/2 checks passed' for both. The 'Alarm state' column has a 'View alarm' link for each instance. The interface includes various controls like 'Connect', 'Instance state', 'Actions', and a 'Launch instances' button.

Name	Instance ID	Instance state	Instance type	Status check	Alarm state
frontend-angular	i-06f3fab3f5a4923f	Running	t2.medium	2/2 checks passed	View alarm
backend-angular	i-06d8d6faa71e51418	Running	t2.medium	2/2 checks passed	View alarm

- Connect your ubuntu server via ssh (EC2 direct connect , SSH Client , Mobaxterm , Cloudshell)

- Now Clone the repo in both Frontend and Backend Server using git clone command .

A screenshot of a terminal window on a Linux system. The terminal shows the user 'ubuntu' at IP '172-31-39-59' attempting to run 'sudo -i'. The command fails with 'command not found'. The user then successfully runs 'sudo -i' to become root. As root, the user runs 'git clone https://github.com/rajatpzade/angular-java.git'. The output shows the repository is cloned into 'angular-java'. Finally, the user runs 'ls' in the 'angular-java' directory, listing files: 'README.md', 'angular-frontend', 'spring-backend', and 'springbackend.sql'.

```
ubuntu@ip-172-31-39-59:~$ sudo -i
sudo: -: command not found
ubuntu@ip-172-31-39-59:~$ sudo -i
root@ip-172-31-39-59:~#
root@ip-172-31-39-59:~# 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
Receiving objects: 100% (80/80), 268.11 KiB | 14.11 MiB/s, done.
Resolving deltas: 100% (3/3), done.
root@ip-172-31-39-59:~# ls
angular-java  snap
root@ip-172-31-39-59:~#
root@ip-172-31-39-59:~# cd angular-java/
root@ip-172-31-39-59:~/angular-java# ls
README.md  angular-frontend  spring-backend  springbackend.sql
```

• Frontend Setup steps

- Install Nodejs and npm in the frontend server using command .
sudo apt install nodejs npm
- To confirm installation of Node.js and npm use .
node -v & npm -v
- Now install Angular CLI globally using
sudo npm install -g @angular/cli@14.2.1

- Now its time to install dependence that required to our project
 - npm install
 - ng build
 - ng serve --host 0.0.0.0 --port=80

```
✓ Browser application bundle generation complete.

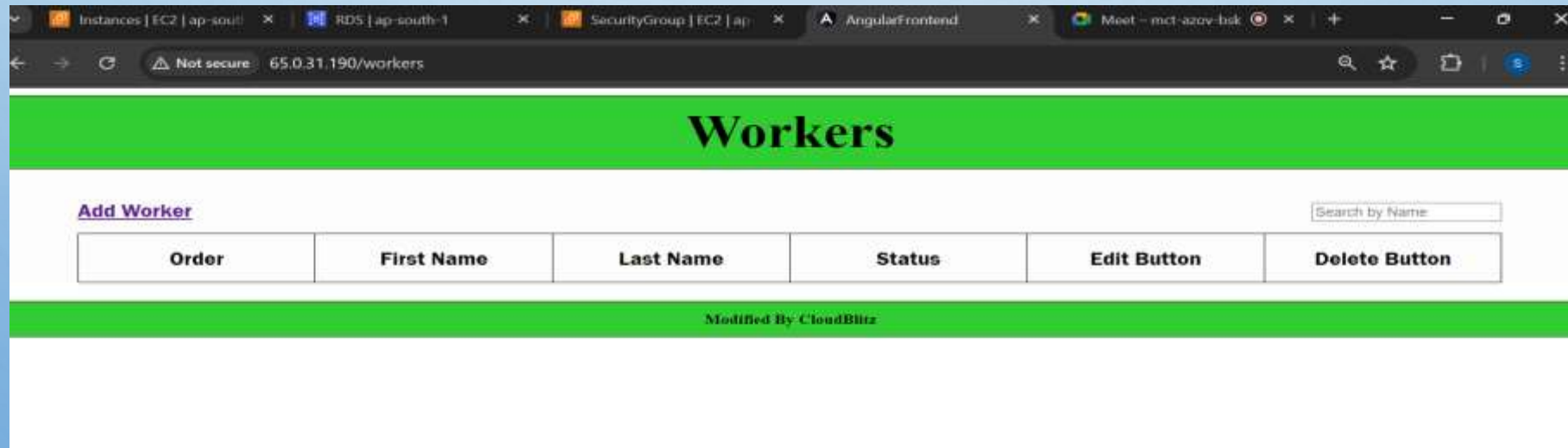
Initial Chunk Files | Names          | Raw Size
vendor.js           | vendor         | 2.47 MB
polyfills.js        | polyfills      | 318.02 kB
styles.css, styles.js | styles        | 210.10 kB
main.js             | main           | 32.69 kB
runtime.js          | runtime        | 6.53 kB
                    | Initial Total  | 3.03 MB

Build at: 2024-05-15T12:35:04.674Z - Hash: b237071b6c0acbab - Time: 23081ms

** Angular Live Development Server is listening on 0.0.0.0:80, open your browser on http://localhost:80/ **

✓ Compiled successfully.
```

- Now hit the public ip on the default browser you will See your Frontend has set-up successfully



• Backend Setup Steps:-

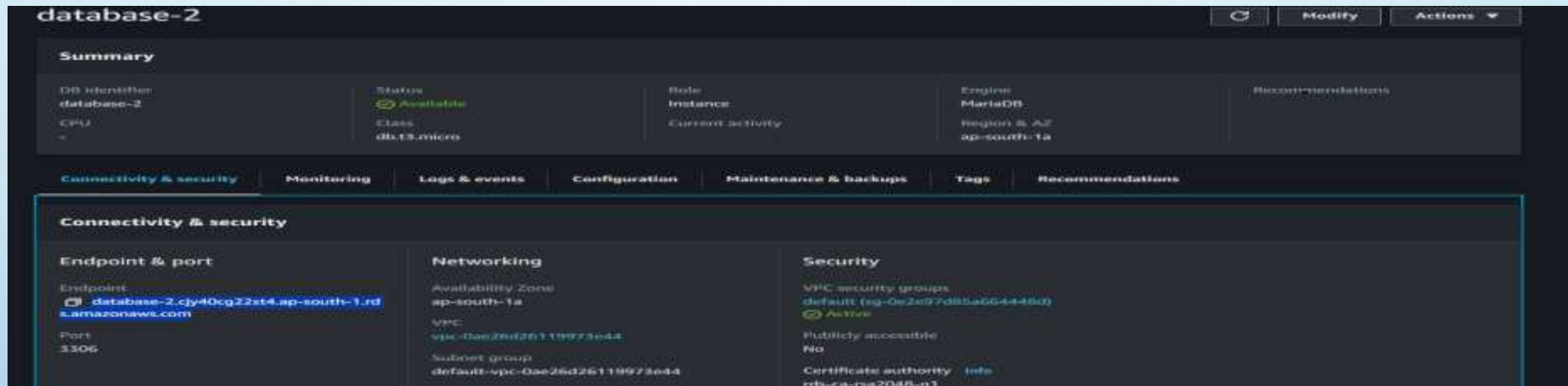
- First install openJdk 8 in your backend server using
`sudo apt install openjdk-8-jdk`
- To build java code install maven
`sudo apt install maven`
- Build the project using Maven
`mvn clean package -Dmaven.test.skip=true`
- Finally run your Application using command
`java -jar target/spring-backend-v1.jar`

```
[INFO] Not compiling test sources
[INFO]
[INFO] --- maven-surefire-plugin:2.22.2:test (default-test) @ spring-backend ---
[INFO] Tests are skipped.
[INFO]
[INFO] --- maven-jar-plugin:3.1.1:jar (default-jar) @ spring-backend ---
[INFO] Building jar: /root/angular-java/spring-backend/target/spring-backend-v1.jar
[INFO]
[INFO] --- spring-boot-maven-plugin:2.7.4:repackage (repackage) @ spring-backend ---
[INFO] Replacing main artifact with repackaged archive
[INFO]
[INFO] BUILD SUCCESS
[INFO]
[INFO] Total time: 4.066 s
[INFO] Finished at: 2024-06-18T06:49:57Z
[INFO]
root@ip-172-31-39-59:~/angular-java/spring-backend#
root@ip-172-31-39-59:~/angular-java/spring-backend# java -jar target/spring-backend-v1.jar

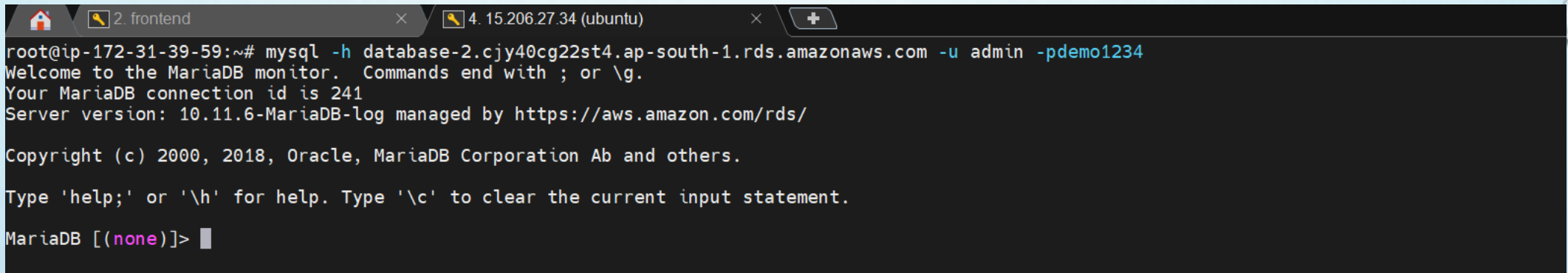
Spring Boot
(Spring Boot 2.7.4)

2024-06-18 06:50:14.457 INFO 5252 --- [main] c.e.s.SpringBackendApplication : Starting SpringBackendApplication v1
using Java 1.8.0_412 on ip-172-31-39-59 with PID 5252 (/root/angular-java/spring-backend/target/spring-backend-v1.jar started by root in /
root/angular-java/spring-backend)
2024-06-18 06:50:14.480 INFO 5252 --- [main] c.e.s.SpringBackendApplication : No active profile set, falling back to
1 default profile: "default"
2024-06-18 06:50:15.790 INFO 5252 --- [main] .s.d.r.c.RepositoryConfigurationDelegate : Bootstrapping Spring Data JPA repository
ries in DEFAULT mode.
2024-06-18 06:50:15.880 INFO 5252 --- [main] .s.d.r.c.RepositoryConfigurationDelegate : Finished Spring Data repository scanning
in 72 ms. Found 1 JPA repository interfaces.
2024-06-18 06:50:16.039 INFO 5252 --- [main] o.s.b.w.embedded.tomcat.TomcatWebServer : Tomcat initialized with port(s): 8080
(http)
2024-06-18 06:50:16.858 INFO 5252 --- [main] o.apache.catalina.core.StandardService : Starting service [Tomcat]
2024-06-18 06:50:16.859 INFO 5252 --- [main] org.apache.catalina.core.StandardEngine : Starting Servlet engine: [Apache Tomcat
/9.0.65]
2024-06-18 06:50:16.972 INFO 5252 --- [main] o.a.c.e.C.[Tomcat].[localhost].[/] : Initializing Spring embedded WebApplic
ationContext
2024-06-18 06:50:16.972 INFO 5252 --- [main] w.s.c.ServletWebServerApplicationContext : Root WebApplicationContext: initializa
tion completed in 2379 ms
```

- Installing MariaDB, Setting Password, and Importing Database on Ubuntu Linux
- Now, Nagivate to AWS RDS service to create the database , After successfully creating the database we will get the endpoint



- Install the MariaDB server on the backend server using commands
- `sudo apt update`
- `sudo apt install mariadb-server`
- `sudo systemctl start mariadb`
- `sudo systemctl enable mariadb`
- `sudo mysql_secure_installation`
- After installing MariaDB login in to database using RDS end point that we have got .
- `mysql -h (rds-Endpoint) -u (username) -p(password)`

A terminal window with two tabs. The first tab is titled '2. frontend' and the second is '4. 15.206.27.34 (ubuntu)'. The terminal shows the command 'mysql -h database-2.cjy40cg22st4.ap-south-1.rds.amazonaws.com -u admin -pdemo1234' being executed. The output includes a welcome message, connection ID, server version, copyright, and help instructions. The prompt 'MariaDB [(none)]>' is visible at the bottom.

```
root@ip-172-31-39-59:~# mysql -h database-2.cjy40cg22st4.ap-south-1.rds.amazonaws.com -u admin -pdemo1234
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 241
Server version: 10.11.6-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)]> 
```

- This output we will get when we enter inside database.
- ## Connectors Configuration for frontend and Backend

 - Frontend-Server
 - PATH = “ **cd /root/angular-java/angular-frontend/src/app/services** “
 - You will the file “**worker.service.ts**” enter the file put your public lp of backend server
(On next page is content of worker.service.ts file)


```

import { HttpClient } from '@angular/common/http';
import { Injectable } from '@angular/core';
import { Observable } from 'rxjs';
import { map } from 'rxjs/operators';
import { Worker } from '../models/worker';

@Injectable({
  providedIn: 'root'
})
export class WorkerService {

  private getUrl: string = "http://15.206.27.34:8080/api/v1/workers";

  constructor(private _httpClient: HttpClient) { }

  getWorkers(): Observable<Worker[]> {
    return this._httpClient.get<Worker[]>(this.getUrl).pipe(
      map(response => response)
    )
  }

  saveWorkers(worker: Worker): Observable<Worker> {
    return this._httpClient.post<Worker>(this.getUrl, worker);
  }

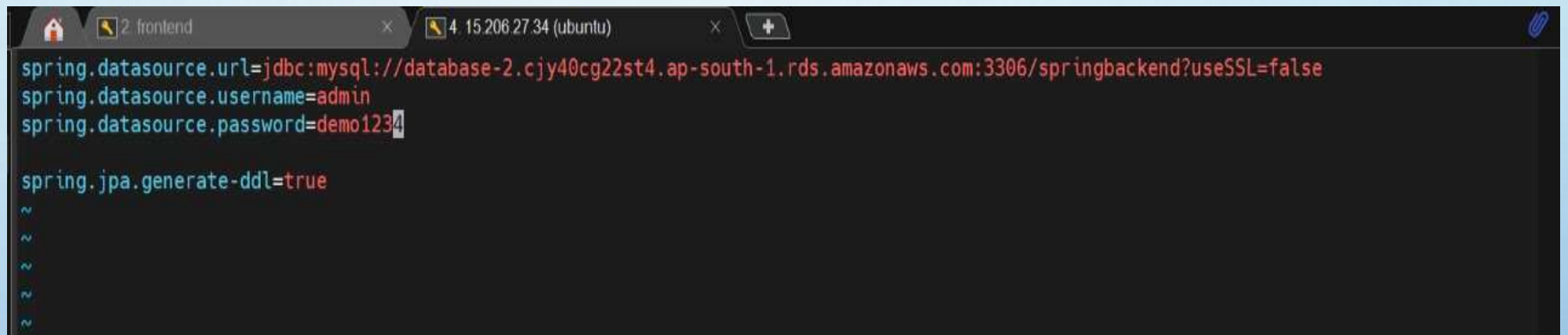
  getWorker(id: Number): Observable<Worker> {
    return this._httpClient.get<Worker>(`${this.getUrl}/${id}`).pipe(
      map(response => response)
    )
  }
}

```

- Where I have selected ip replace it with Public Ip of your backend server .
- Rebuild the dependences using
- ng build
- ng serve --host 0.0.0.0 --port=80

• Backend-Server

- PATH = “ **cd /root/angular-java/spring-backend/src/main/resource/**”
- You will the file “ **application.properties** “ enter the file put your endpoint of RDS database with username and password . (As shown in below)

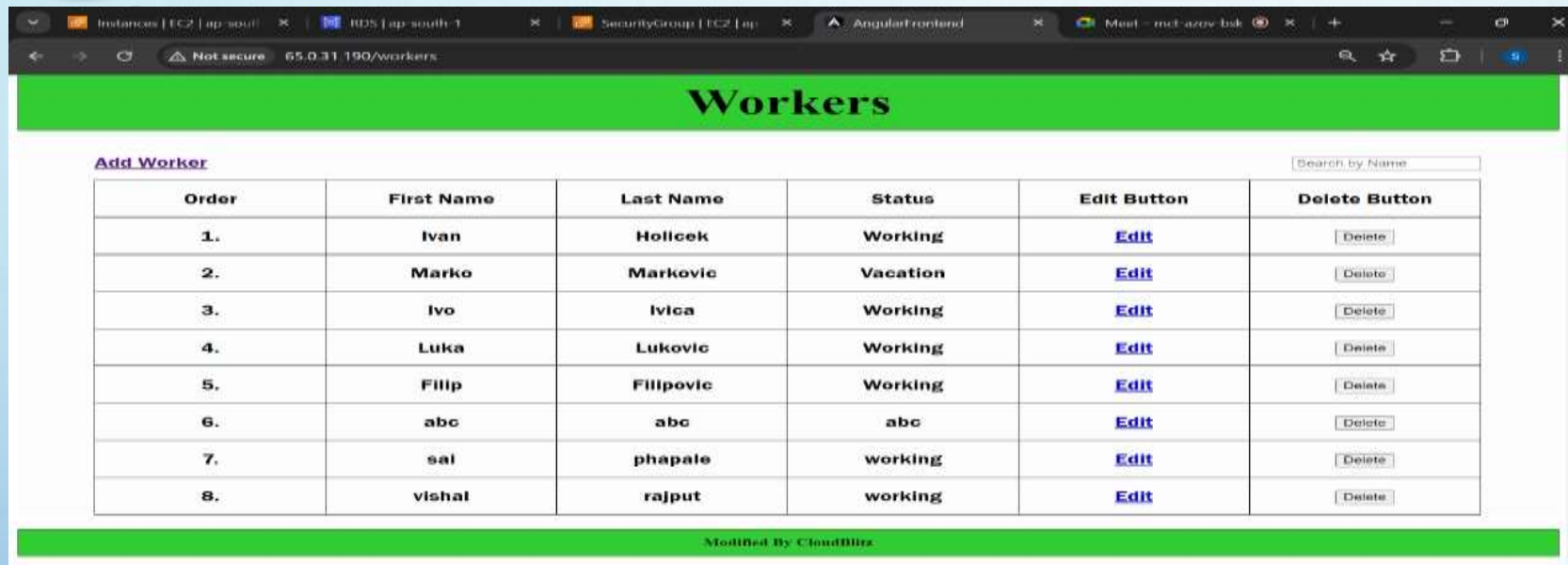


```
spring.datasource.url=jdbc:mysql://database-2.cjy40cg22st4.ap-south-1.rds.amazonaws.com:3306/springbackend?useSSL=false
spring.datasource.username=admin
spring.datasource.password=demo1234

spring.jpa.generate-ddl=true
~
~
~
~
~
```

- Rebuild the project of maven and run application using commands
 - mvn clean package -Dmaven.test.skip=true
 - java -jar target/spring-backend-v1.jar

- After doing all changes Properly, the Application Will work these are some Outputs :-



The screenshot shows a web browser window with the URL 65.0.31.190/workers. The page has a green header with the title 'Workers'. Below the header, there is a link 'Add Worker' and a search bar labeled 'Search by Name'. The main content is a table with 6 columns: Order, First Name, Last Name, Status, Edit Button, and Delete Button. The table contains 8 rows of worker data. At the bottom of the page, there is a green footer that says 'Modified By CloudBlitz'.

Order	First Name	Last Name	Status	Edit Button	Delete Button
1.	Ivan	Holicek	Working	Edit	<input type="button" value="Delete"/>
2.	Marko	Markovic	Vacation	Edit	<input type="button" value="Delete"/>
3.	Ivo	Ivica	Working	Edit	<input type="button" value="Delete"/>
4.	Luka	Lukovic	Working	Edit	<input type="button" value="Delete"/>
5.	Fillip	Fillipovic	Working	Edit	<input type="button" value="Delete"/>
6.	abc	abc	abc	Edit	<input type="button" value="Delete"/>
7.	sai	phapale	working	Edit	<input type="button" value="Delete"/>
8.	vishal	rajput	working	Edit	<input type="button" value="Delete"/>

This is application output where we can add, edit , delete workers .

```
MariaDB [(none)]> use springbackend;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
MariaDB [springbackend]> show tables;
+-----+
| Tables_in_springbackend |
+-----+
| tbl_workers              |
+-----+
1 row in set (0.001 sec)

MariaDB [springbackend]> select * from tbl_workers;
+----+-----+-----+-----+
| id  | status | workerfname | workerlname |
+----+-----+-----+-----+
| 37  | Vacation | Marko      | Markovic    |
| 40  | Working  | Ivo        | Ivica       |
| 41  | Working  | Luka       | Lukovic     |
| 42  | Working  | Filip      | Filipovic   |
| 43  | abc      | abc        | abc         |
| 45  | working  | vishal     | rajput      |
| 46  | working  | vishal     | rajput      |
+----+-----+-----+-----+
7 rows in set (0.004 sec)

MariaDB [springbackend]> ^DBye
```

The Workers data is also successfully stored inside database springbackend .