# Full Stack Java Developer

# Capstone Project –

# Write Up

----------------------------------------------

By Giannis Zoulias

Contents

[1. Creating the Code 3](#_Toc151214347)

[2. Building the AWS Infrastructure 4](#_Toc151214348)

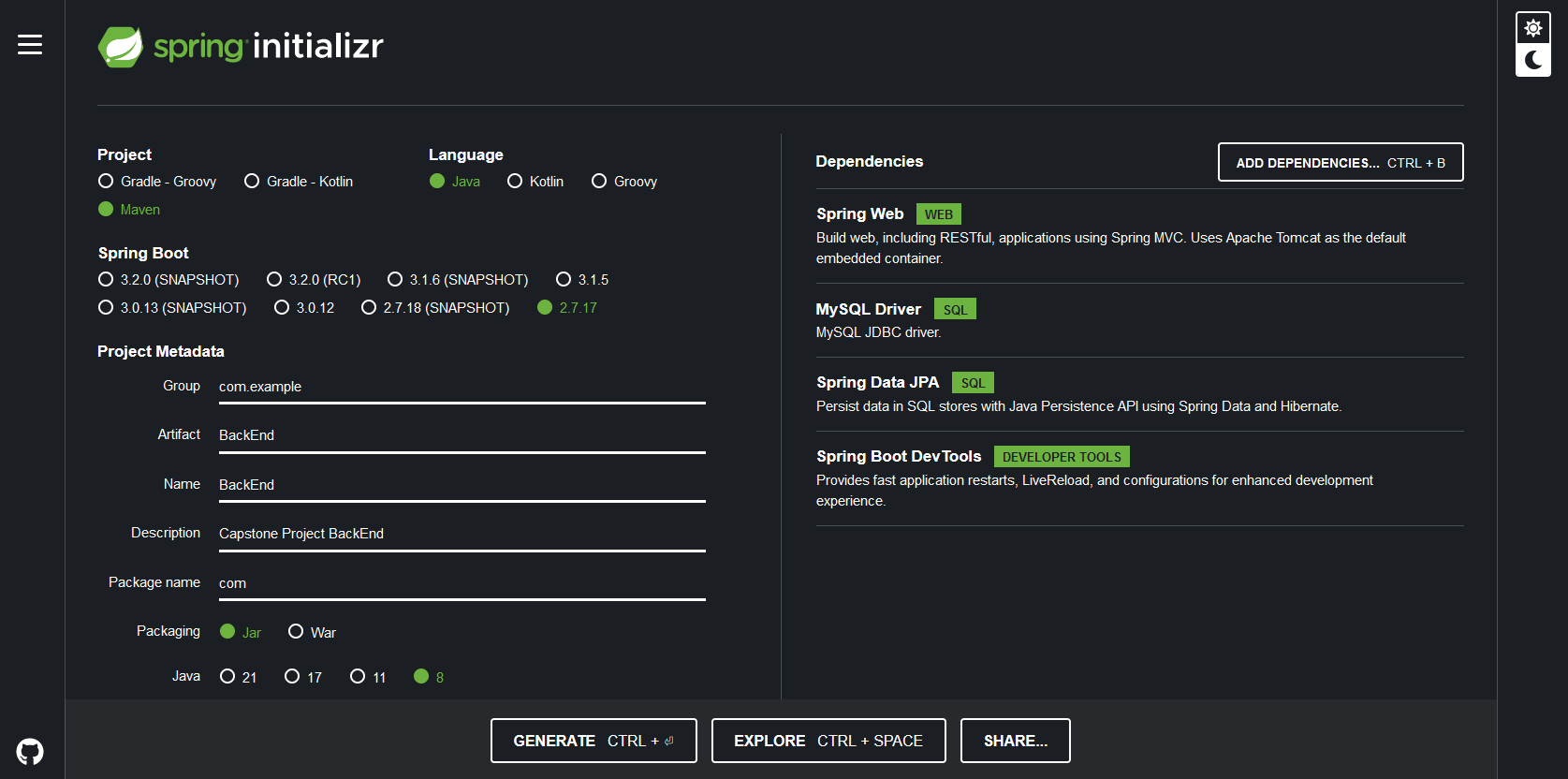
[3. Updating Code with the AWS IPs 11](#_Toc151214349)

[4. BackEnd Installation 13](#_Toc151214350)

[5. FrontEnd Installation 20](#_Toc151214351)

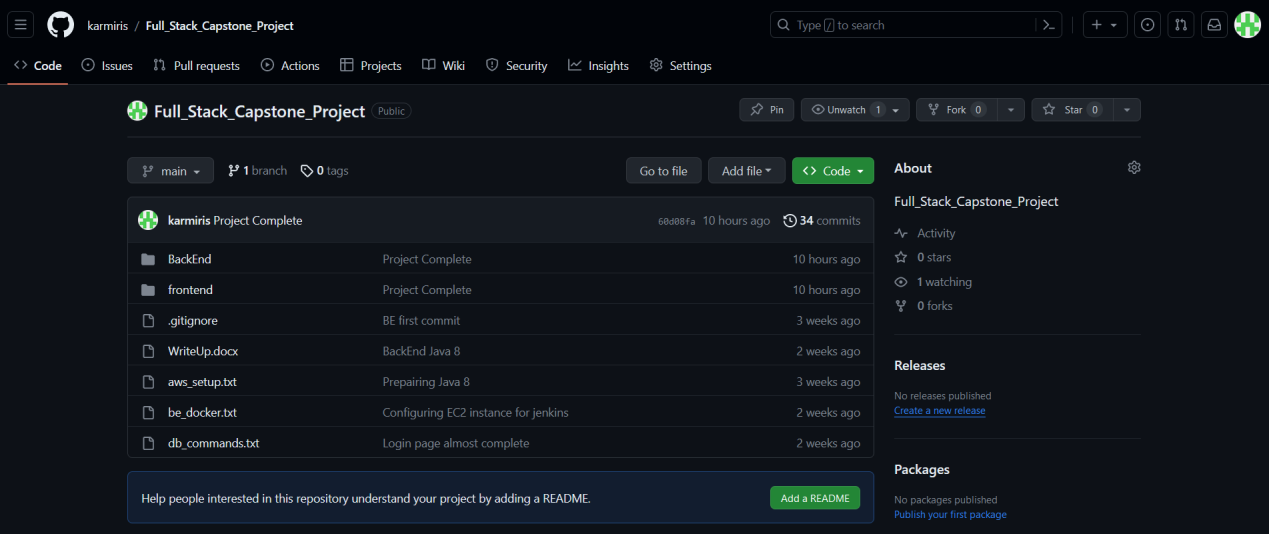
# Creating the Code

We first build our code. For the frontend we use React JS. For the backend we use spring-boot with the following options:



We upload the code in GitHub:

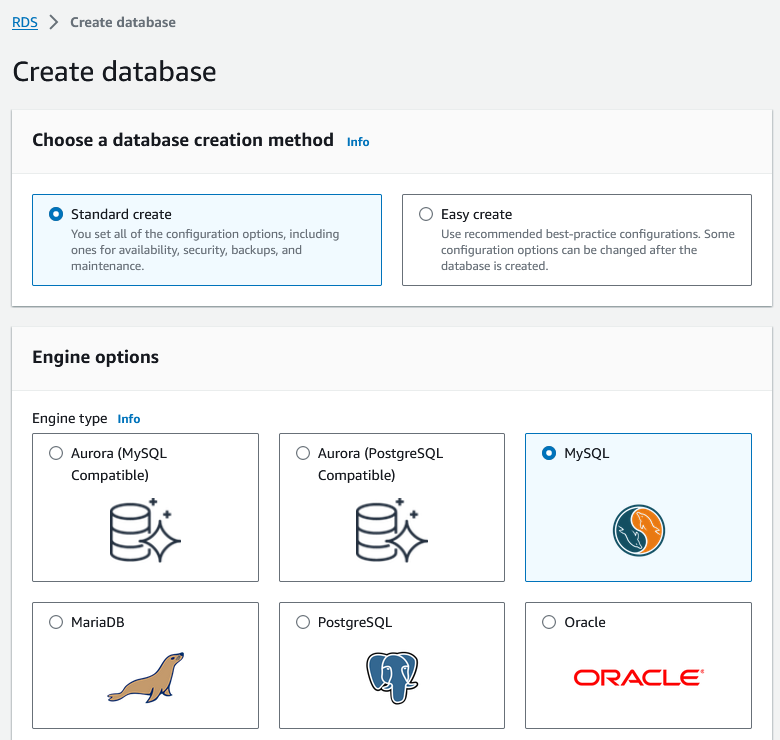
<https://github.com/karmiris/Full_Stack_Capstone_Project>



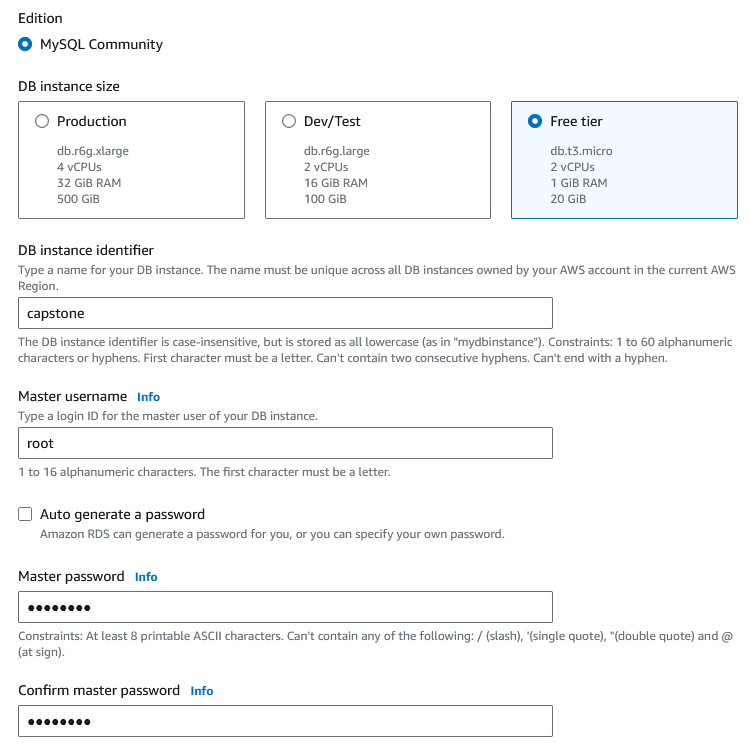
# Building the AWS Infrastructure

We then connect to AWS and create a database (RDS). In all settings that are not mentioned we leave default values:

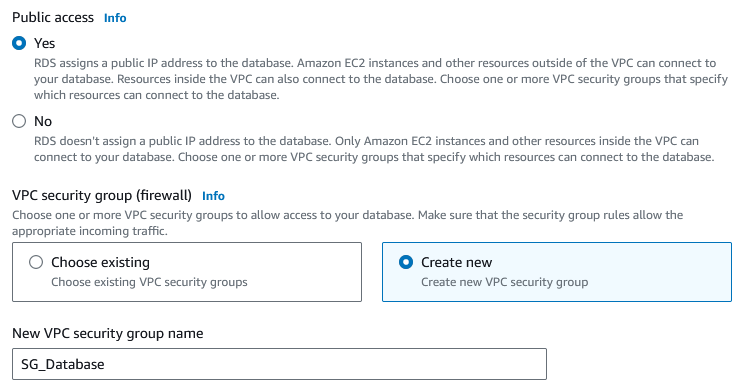
* Engine Type: MySQL
* DB instance size: Free Tier



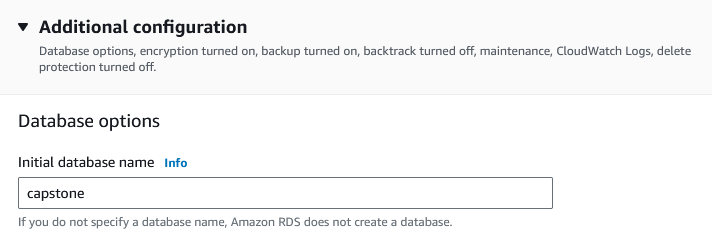
* DB instance identifier: capstone
* Master username: root
* Master password: rootroot



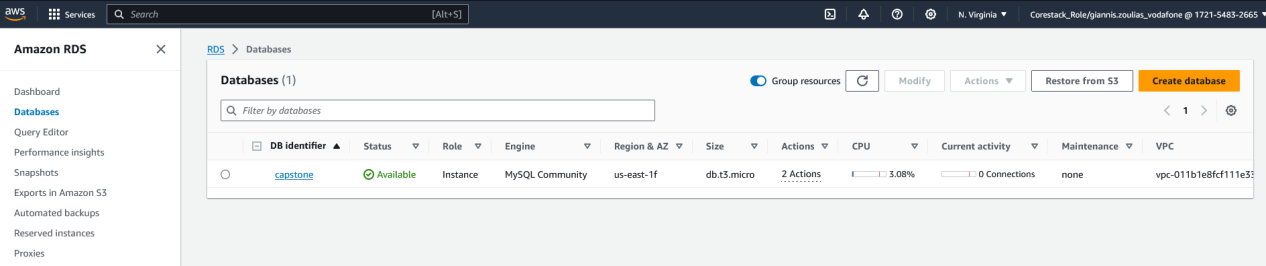
* Public access: Yes
* VPC security group (firewall): Create New
* New VPC security group name: SG\_Database



* Initial database name: capstone



Finally, we create the database:

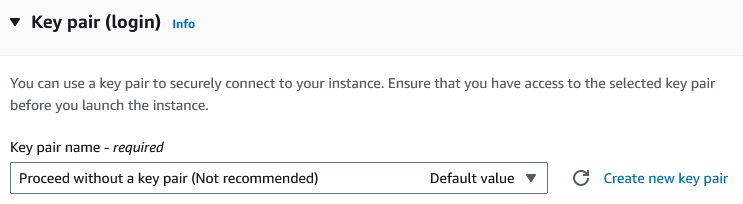


Then we switch to EC2 and create the Backend instance. In all settings that are not mentioned we leave default values:

* Name: BackEnd



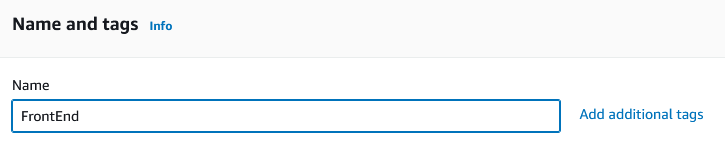
* Key pair: Proceed without a key pair (we can connect through the AWS console)



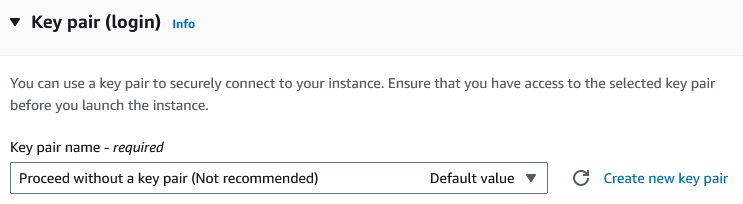
And we launch the instance.

We then create a second instance for the Frontend:

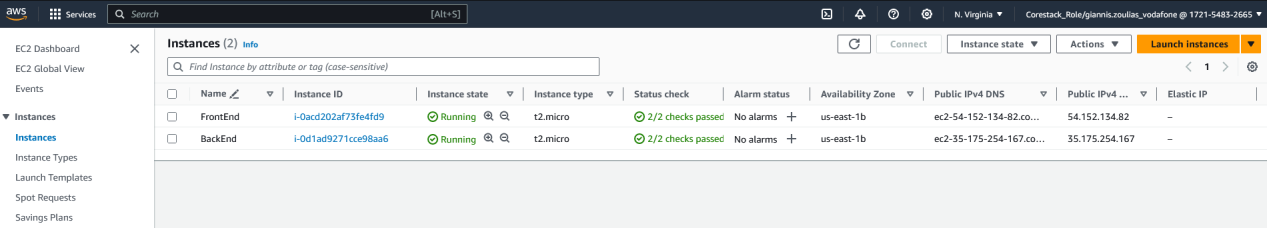
* Name: FrontEnd



* Key pair: Proceed without a key pair



We check that both instances are up and running:



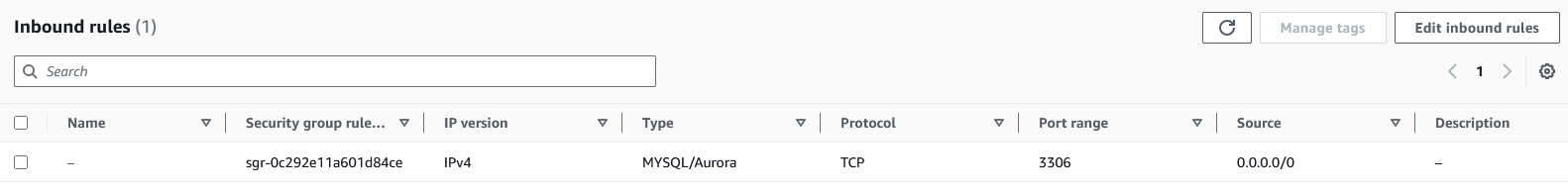
So, we sum up the IPs:

* Database: capstone.crealx72j3np.us-east-1.rds.amazonaws.com
* Database Port: 3306
* Backend: 3.84.93.62
* Frontend: 54.80.234.163

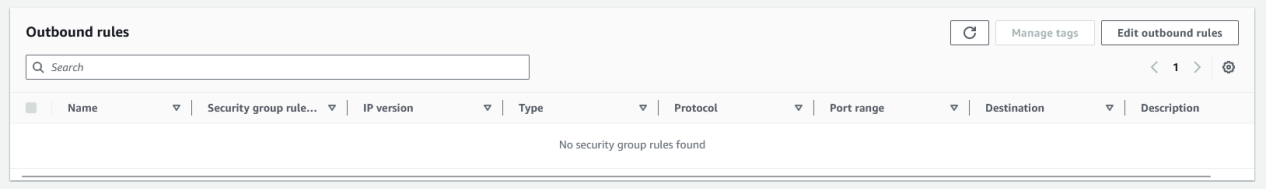
Now we setup the Security Groups (Firewall) Traffic for all elements:

* For the database:

Inbound traffic for port 3306:

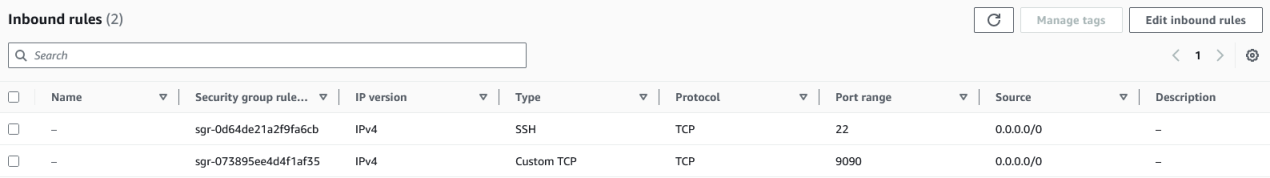


No outgoing traffic:

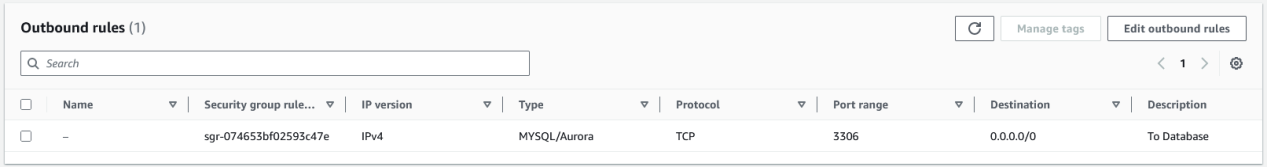


* For the backend:

Inbound traffic from port 9090 and for SSH:

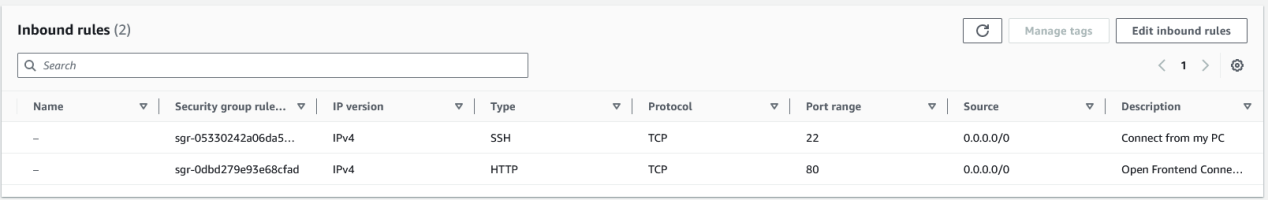


Outgoing traffic to the database (port 3306):

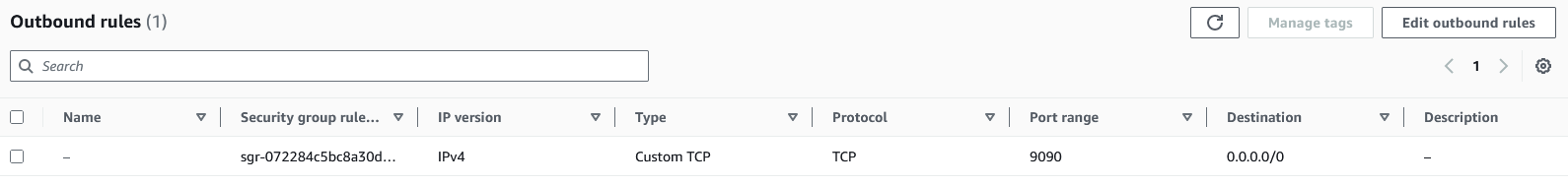


* For the frontend:

Inbound HHTP and SSH traffic:



Outgoing traffic to port 9090:

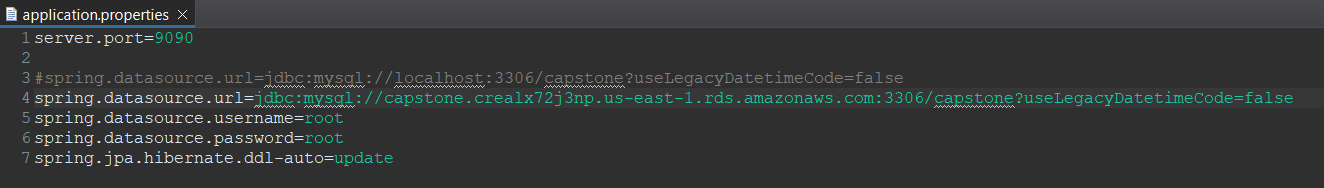


# Updating Code with the AWS IPs

Now we know the AWS endpoints, we can update our code configuration with them.

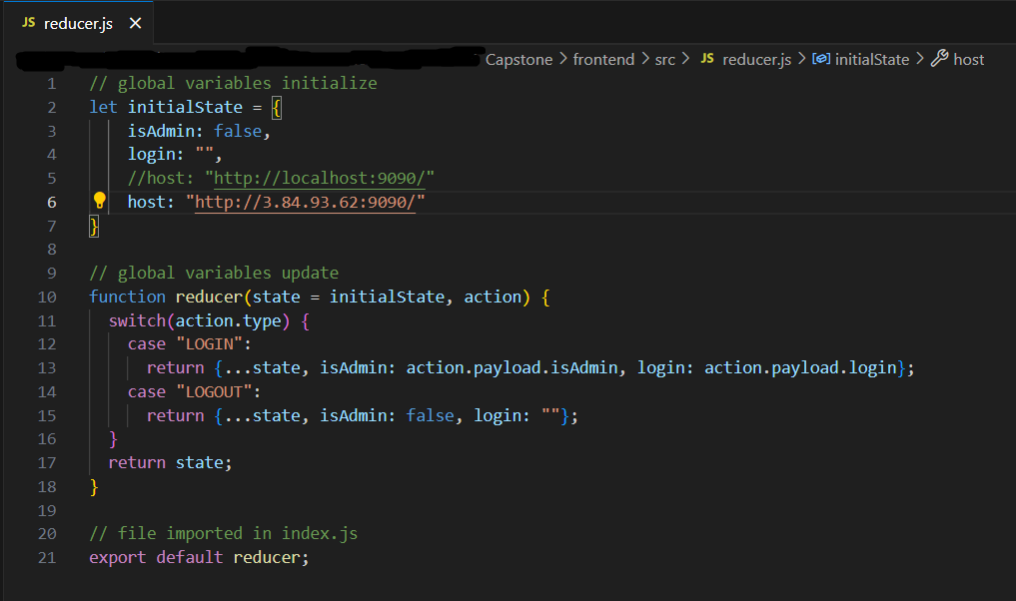
For the backend to connect to the database, we edit the file:

\BackEnd\src\main\resources\application.properties

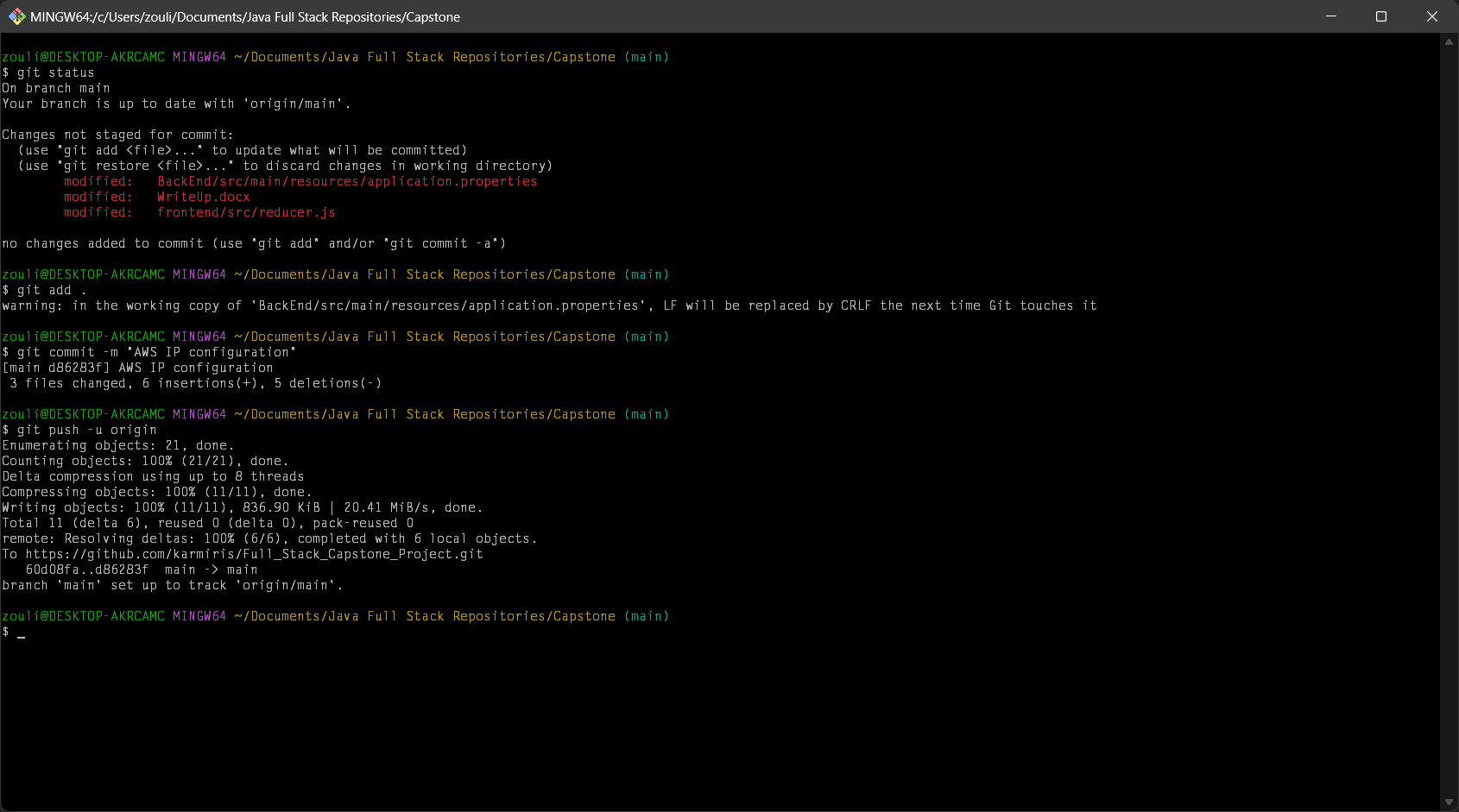


For the frontend to communicate with the backend, we edit the file:

\frontend\src\reducer.js

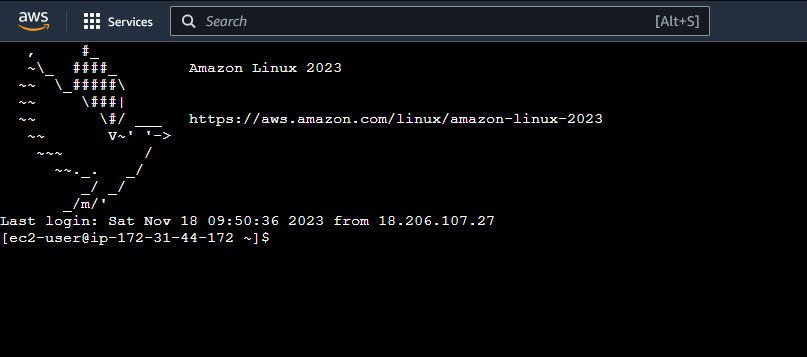


Finally, we commit the changes in GitHub:



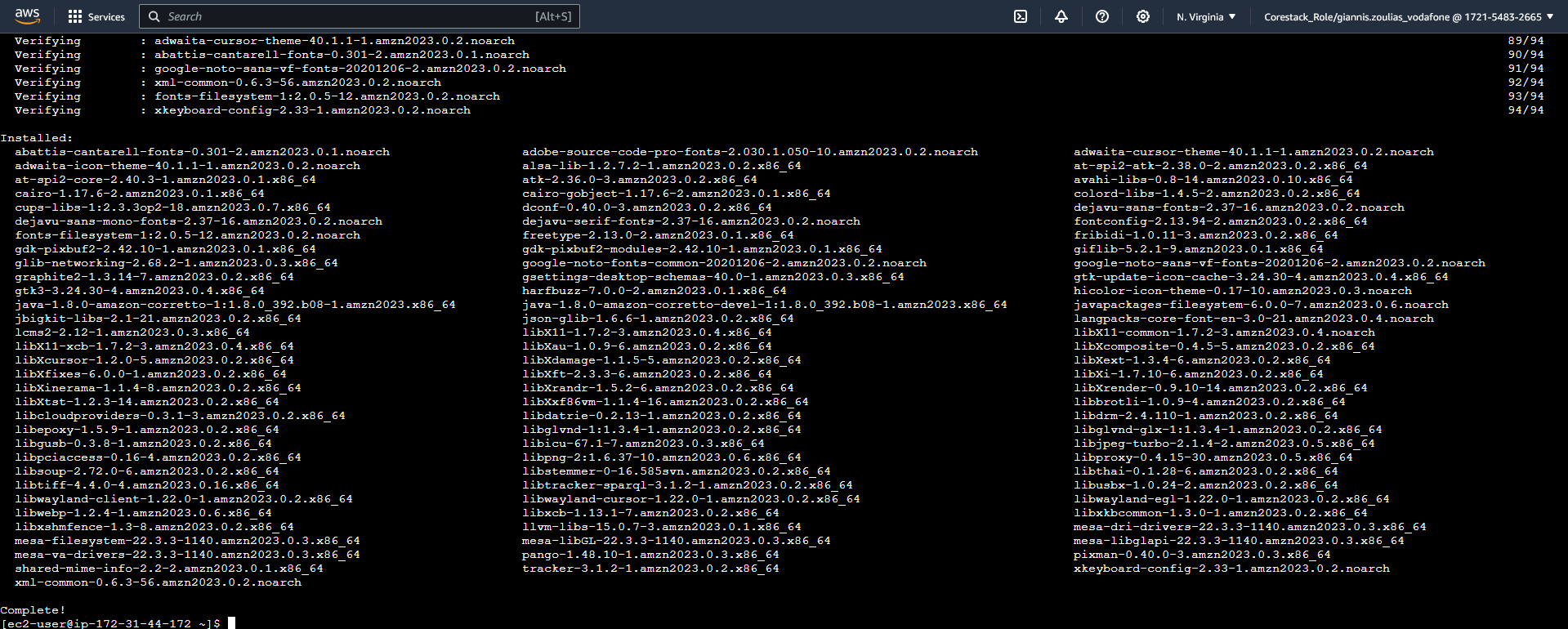
# BackEnd Installation

We connect to the Backend instance and we start installing the needed software. We temporarily have to allow outgoing traffic in the security group for the needed packages to get downloaded:



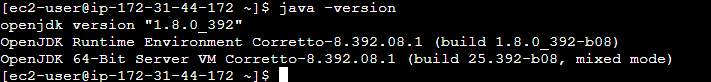
First, we install Java 1.8:

*sudo yum install -y java-1.8.0-devel*

**

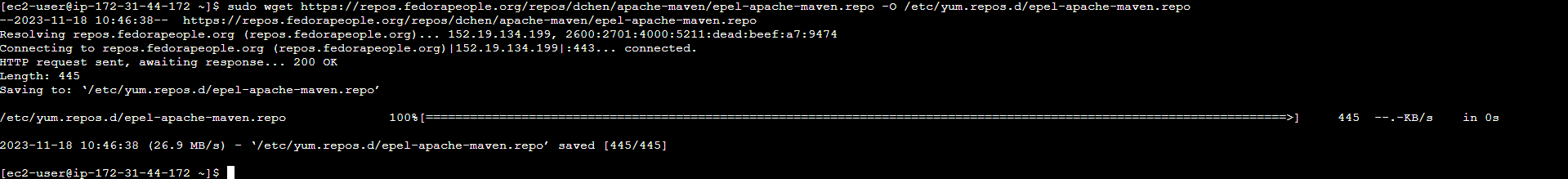
And we confirm the installation has completed successfully:

*java –version*



We then install Maven. First we add the Maven repo:

*sudo wget https://repos.fedorapeople.org/repos/dchen/apache-maven/epel-apache-maven.repo -O /etc/yum.repos.d/epel-apache-maven.repo*

**

Then we specify the version to be installed:

*sudo sed -i s/\$releasever/6/g /etc/yum.repos.d/epel-apache-maven.repo*

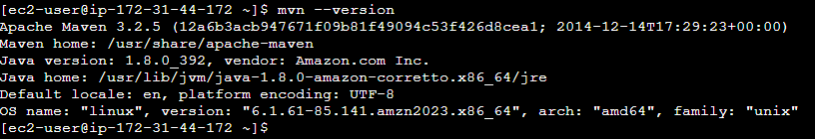
And we run the installer:

*sudo yum install -y apache-maven*



Finally, we confirm that this installation also has run smoothly:

*mvn --version*



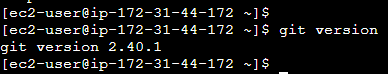
We also need to install Git:

*sudo yum install -y git*

**

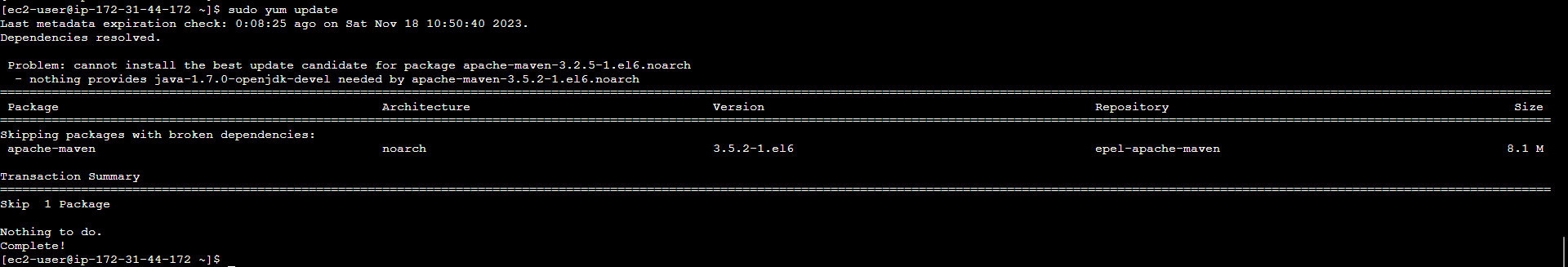
And confirm its installation:

*git version*

**

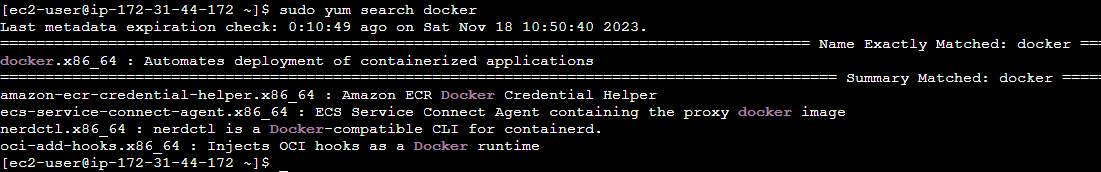
To complete the installations, we will also need Docker. We refresh the yum list:

*sudo yum update*

**

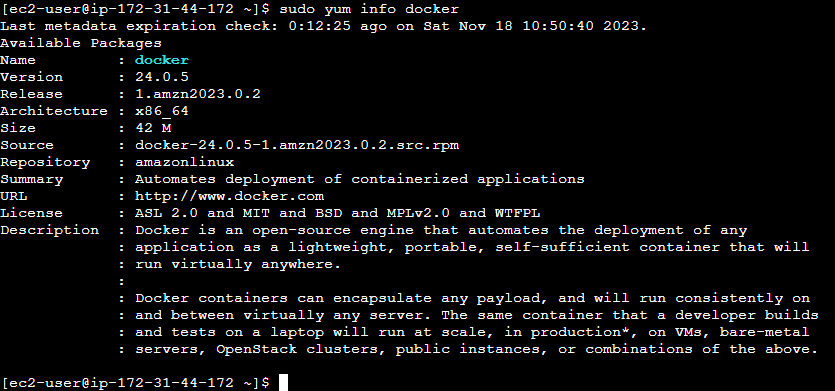
We find the Docker image:

*sudo yum search docker*

**

We get Docker version:

*sudo yum info docker*

**

Then we run the installer:

*sudo yum install -y docker*

**

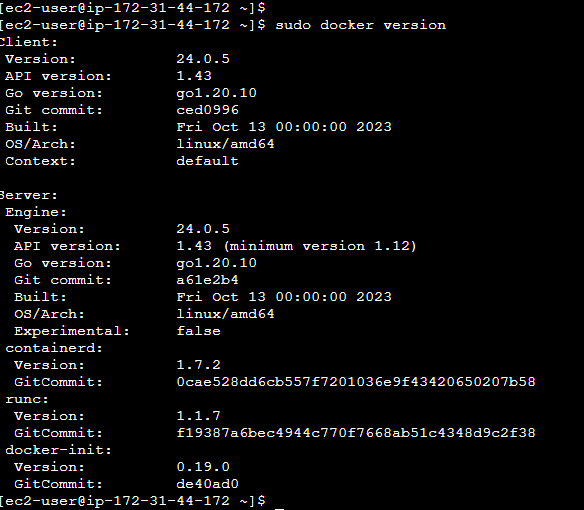
We start Docker:

*sudo service docker start*

**

And we verify all is well done:

*sudo docker version*

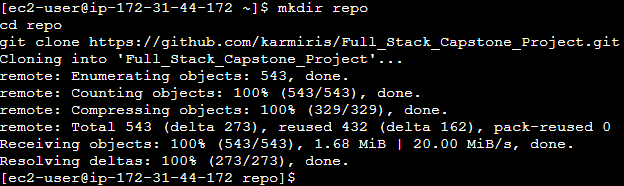
**

We now clone our code in the instance:

*mkdir repo*

*cd repo*

*git clone https://github.com/karmiris/Full\_Stack\_Capstone\_Project.git*

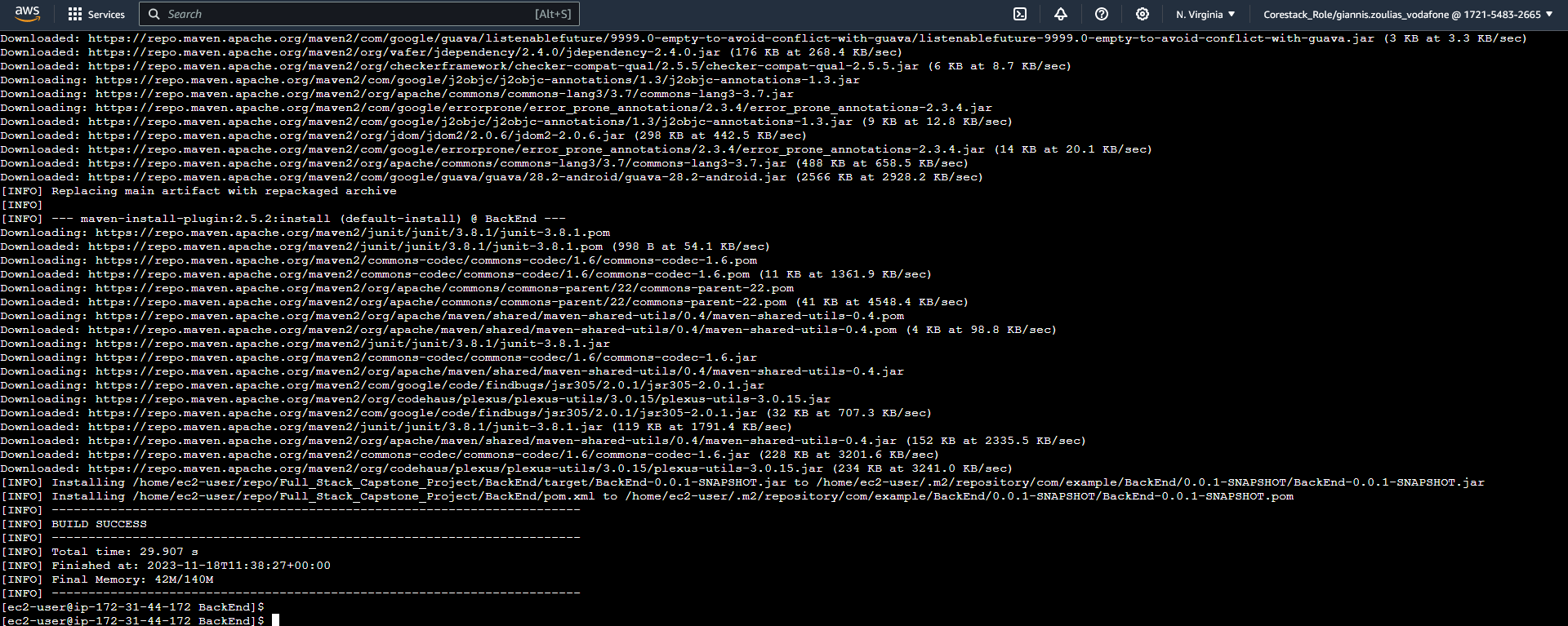


Then we build our Backend code (at this point we may need to loosen the incoming traffic rules of the database too):

*cd Full\_Stack\_Capstone\_Project*

*cd BackEnd*

*mvn clean install*



We create the Docker image:

*sudo docker build --tag=back\_end:latest .*

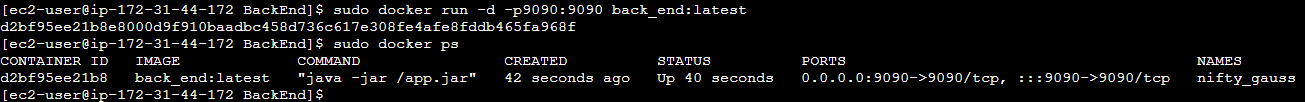
*sudo docker images*

**

And we run it in background, forwarding the same port (9090):

*sudo docker run -d –p9090:9090 back\_end:latest*

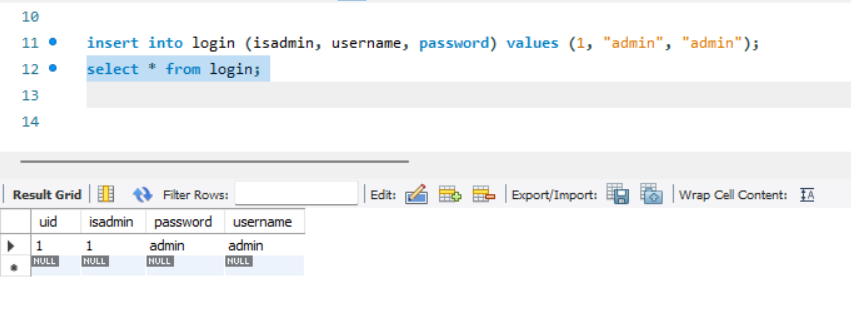
*sudo docker ps*

**

The backend is now running and has connected to the databases, creating the tables. So we connect to the database to create the admin user:

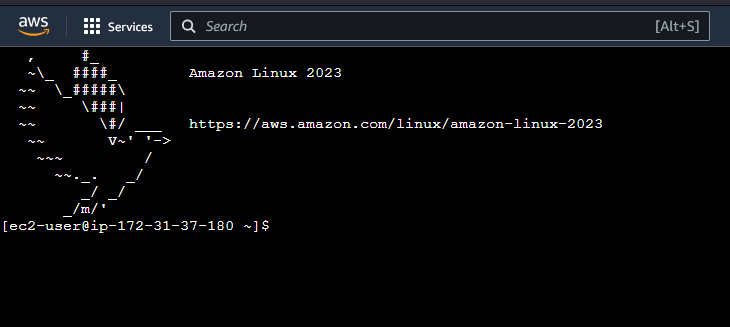
*insert into login (isadmin, username, password) values (1, "admin", "admin");*

*select \* from login;*

**

# FrontEnd Installation

We now connect to the Frontend instance to install the needed software. Again, we temporarily have to loosen the outgoing traffic rules for the installations to download the needed files:

**

At this point we can tighten up all the Security Group rules again.