SE2


# AWS Cloud & Big Data Architectures - Project

LESIEUX BENJAMIN AND THOMAS MARIOTTE

# Table of contents

*Note that the rest of the project is on the README.rd present in the github.*

# Task 1:

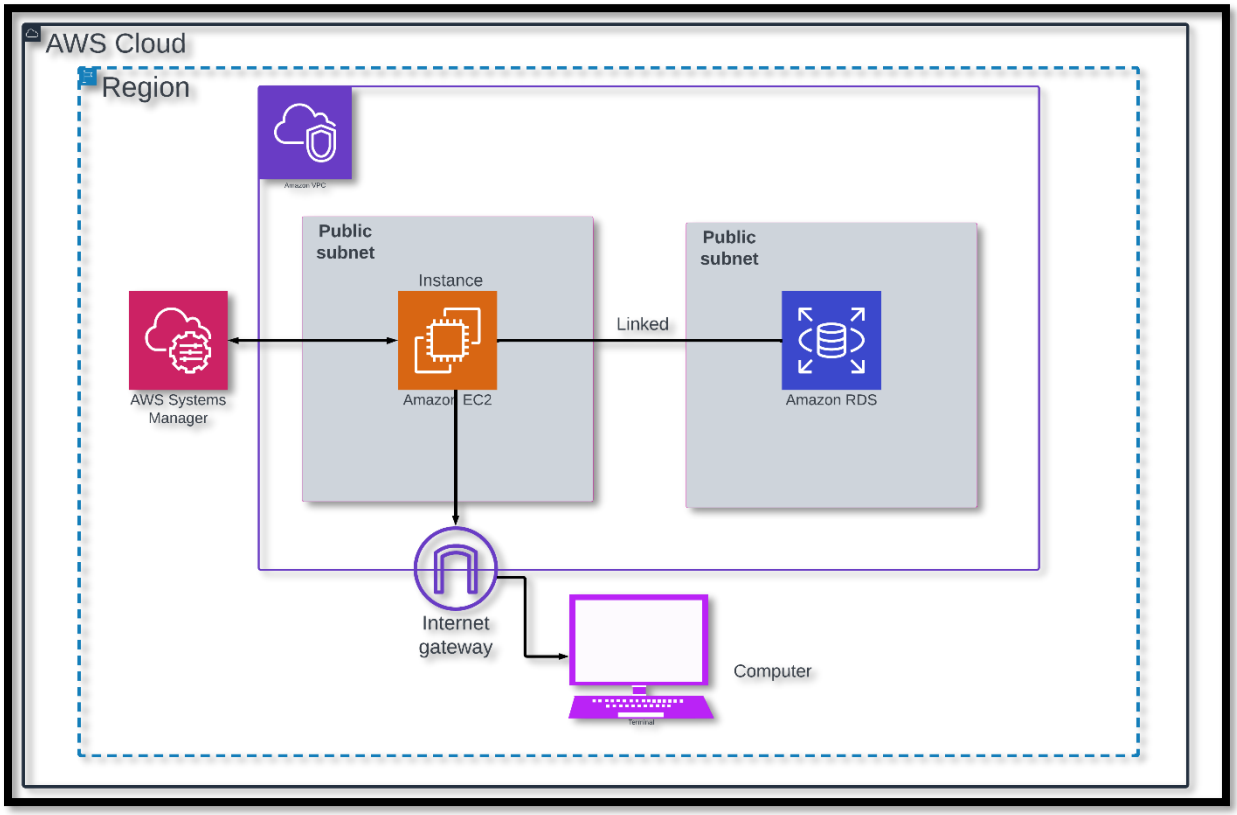
We had to create to deploy an AWS application and make sure that a Social Research Organization have his secure website.

## Our architecture:

We have designed our architecture as follows. In accordance with the requirements of the project, we created a PHP instance hosted on the Amazon EC2 service. This instance stores our website code and allows access from client computers via the internet gateway.

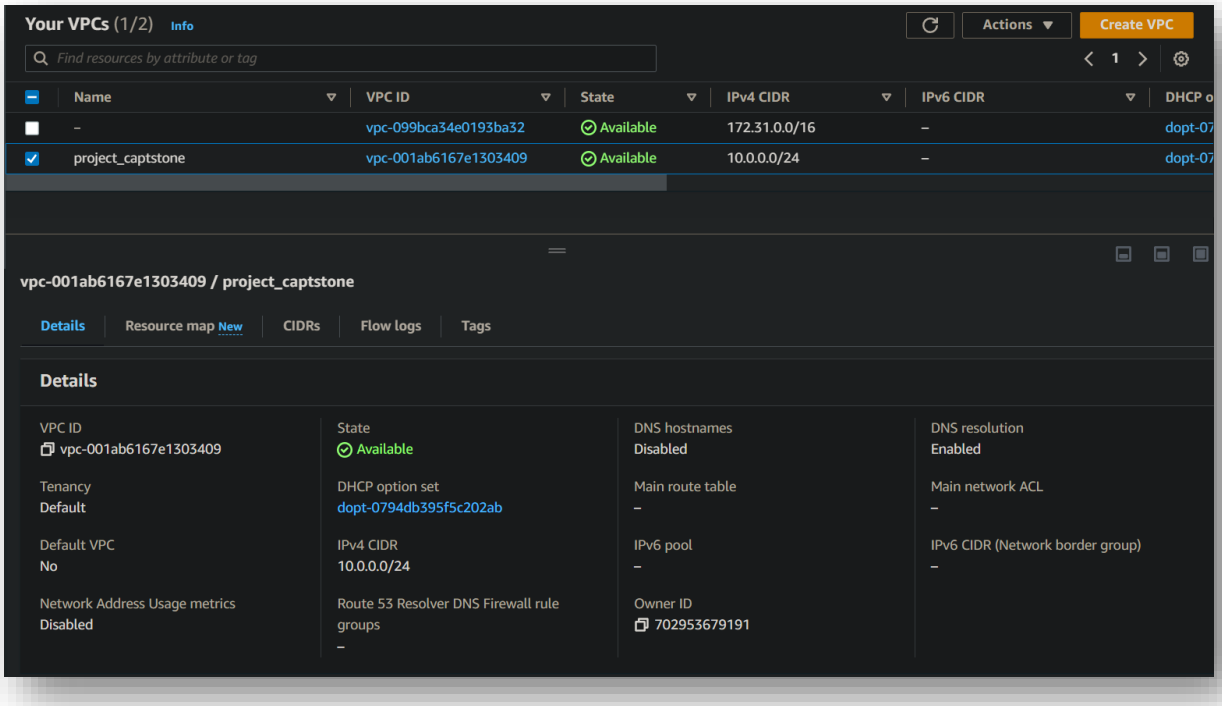
All our data is stored in a private subnet using the Amazon RDS service. The RDS service is configured to work with MariaDB and serves as the storage for all our data. To control access, we utilize the AWS Systems Manager Parameter Store to define the permissions for accessing the data.

This entire architecture is managed through a VPC (Virtual Private Cloud), which enables the creation of both public and private subnets. The public subnet is connected to our EC2 instance, while the private subnet ensures the security of our data by being linked to the RDS service.

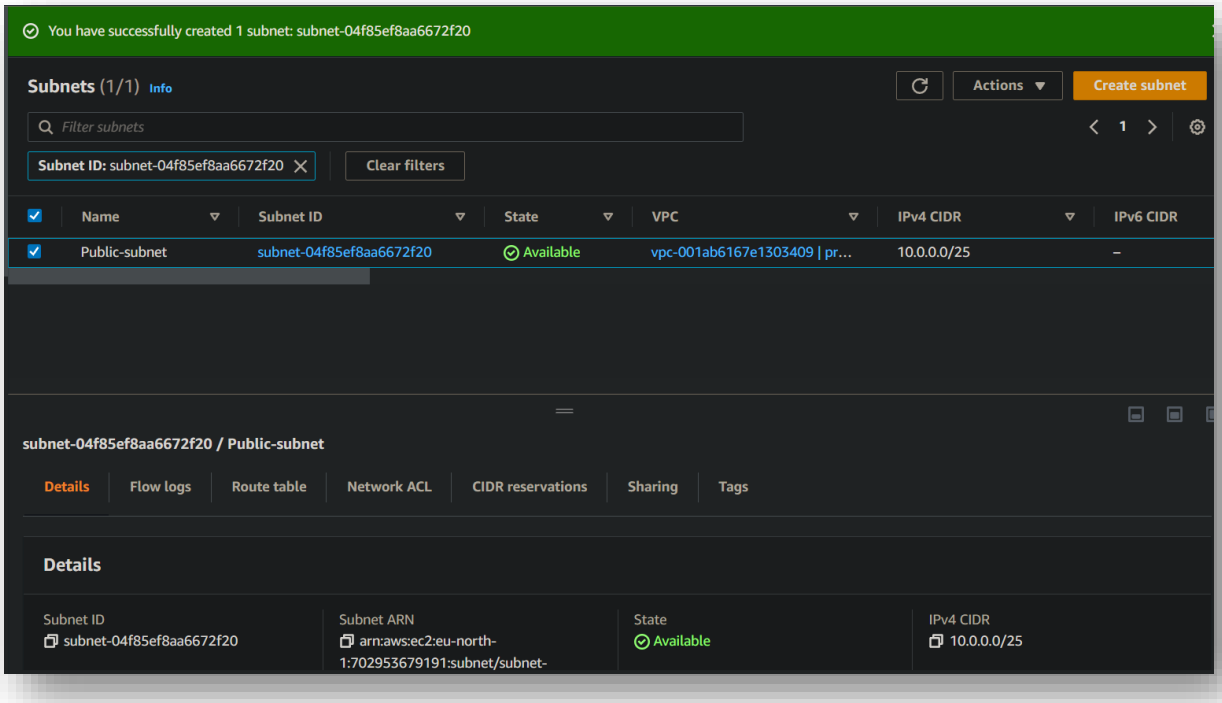


## VPC:

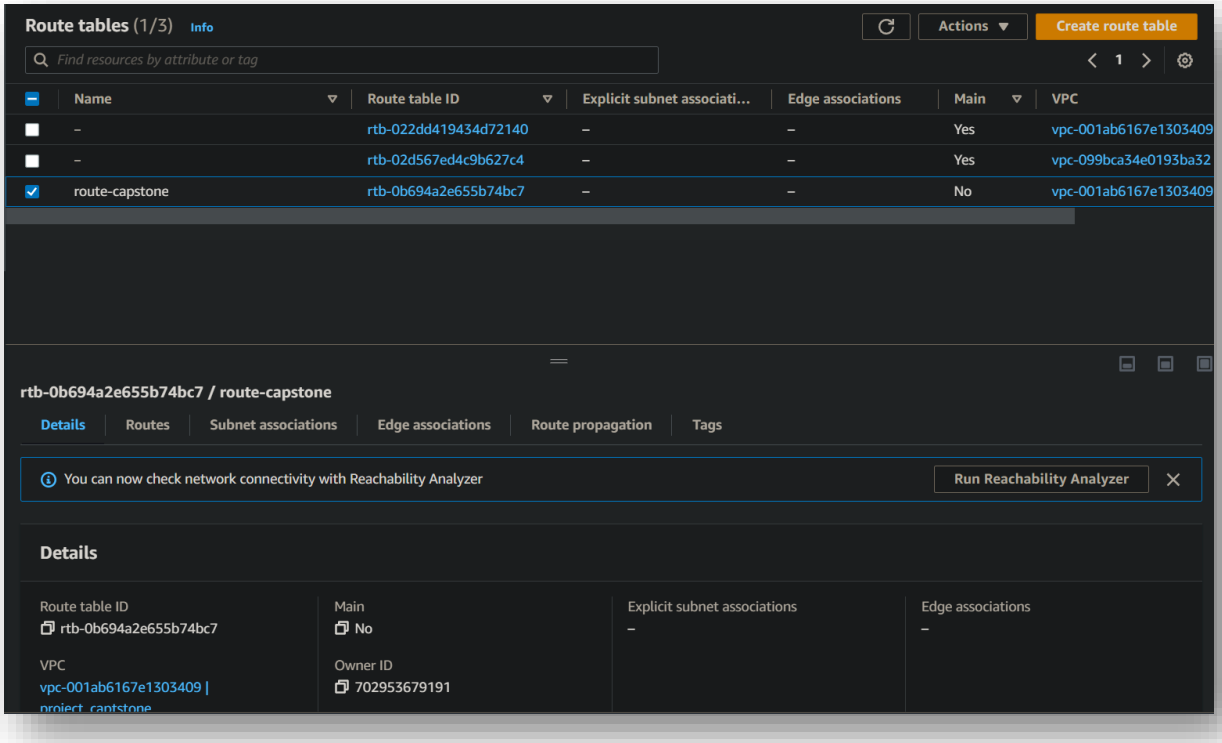
First, let's create a new VPC, separate from the default one, for our project. We will name it **"project-capstone"**.



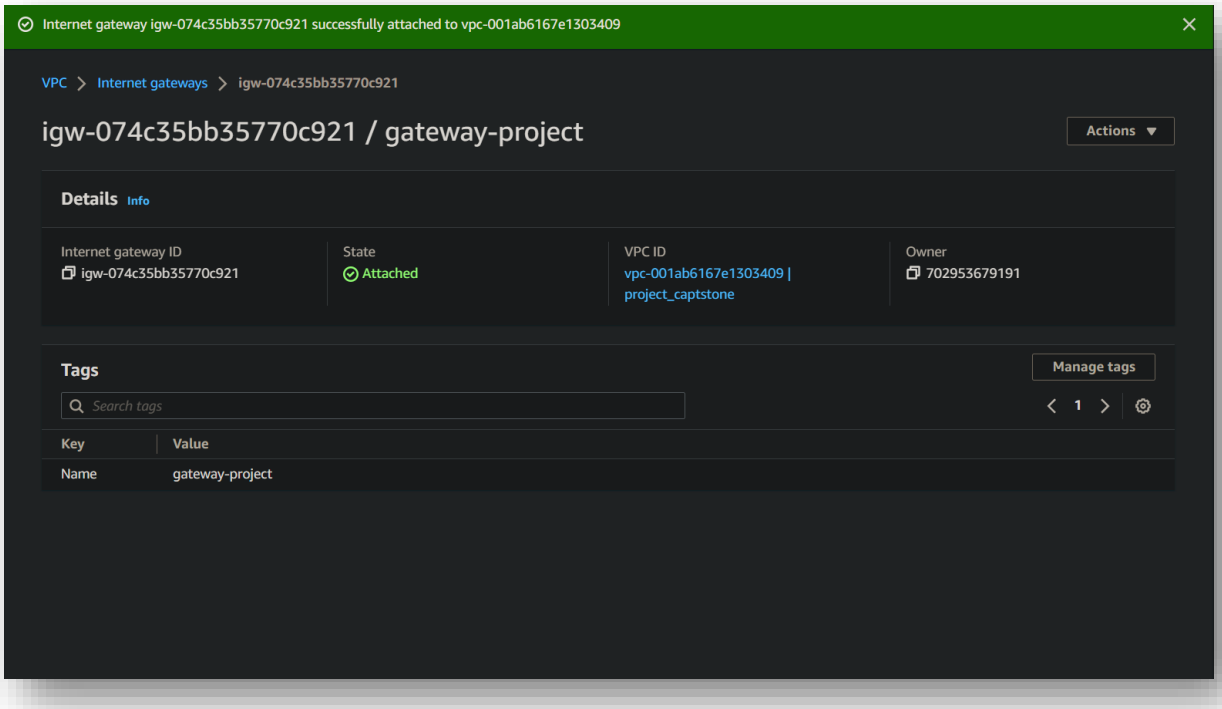

Then we must create a new public subnet that allow us to create an internet gateway and connect our future instance to internet.



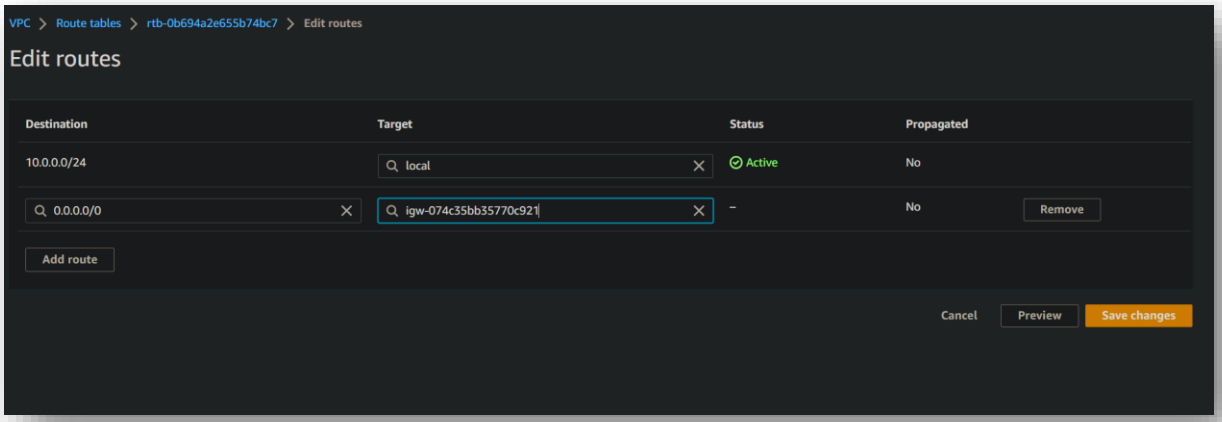
Then we have created our subnet we can create a VPC Route Table. We will create an Internet Gateway after that to connect our route table to internet.




Then our gateway that we attached to the previous VPC:



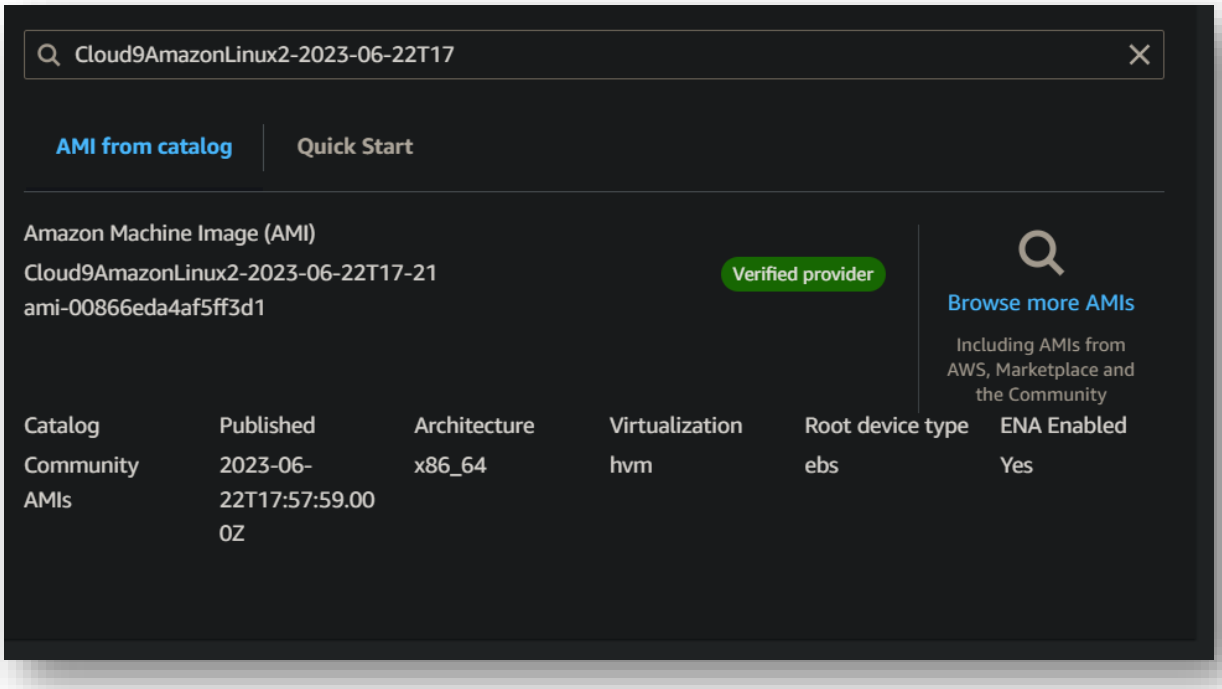
We added our internet gateway to our Route Table in 0.0.0.0/0 (anywhere).



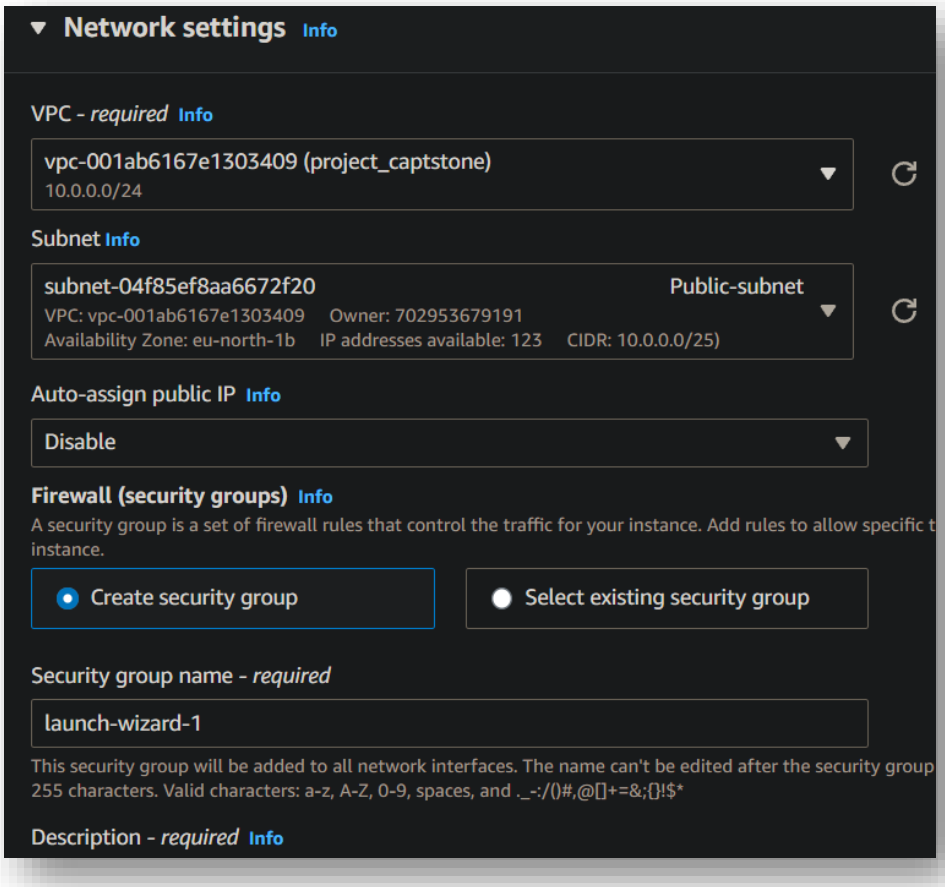

We have now finish with the VPC, let's start with the EC2 now:

## EC2:

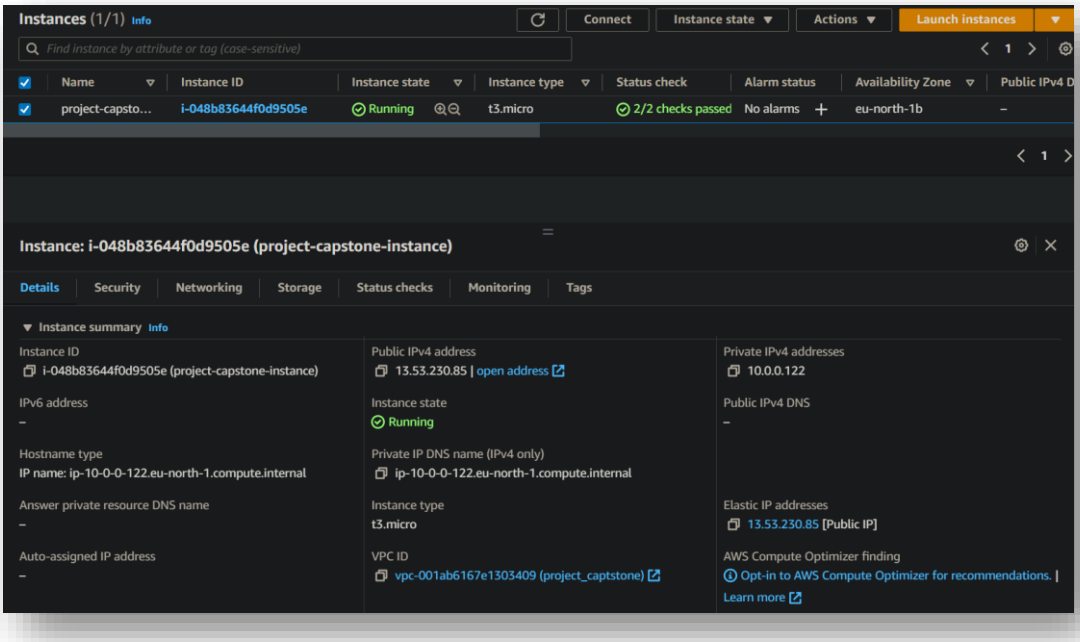
Let’s create an instance with the Linux AMI. We must link this instance to our public subnet.




We had to use this AMI because is provided in the subject.

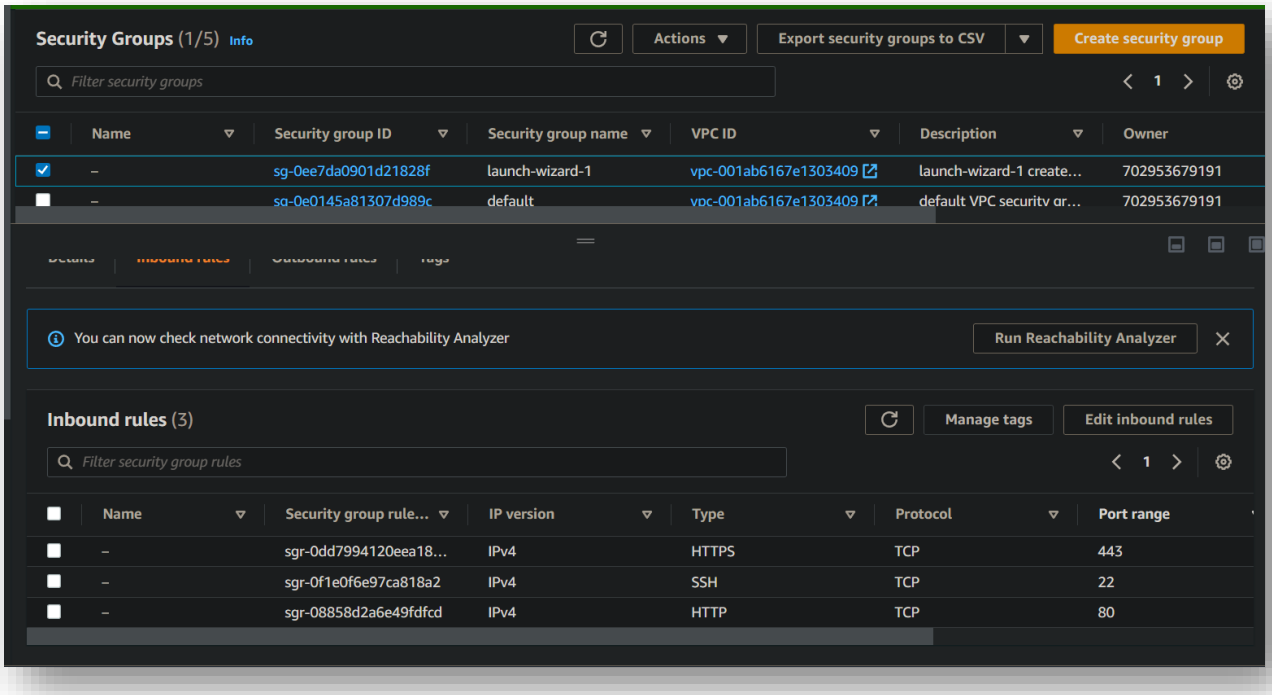



We choose our subnet and we create a security group name **launch-wizard-1** ( I forgot to change his name).




And there are the details of our instance. A public IPv4 address created thank Elastic IP services.

Next, we created Inbound Rules to allow our instance to establish internet connections, specifically using HTTP and HTTPS protocols. These rules ensure that our website is accessible to users.

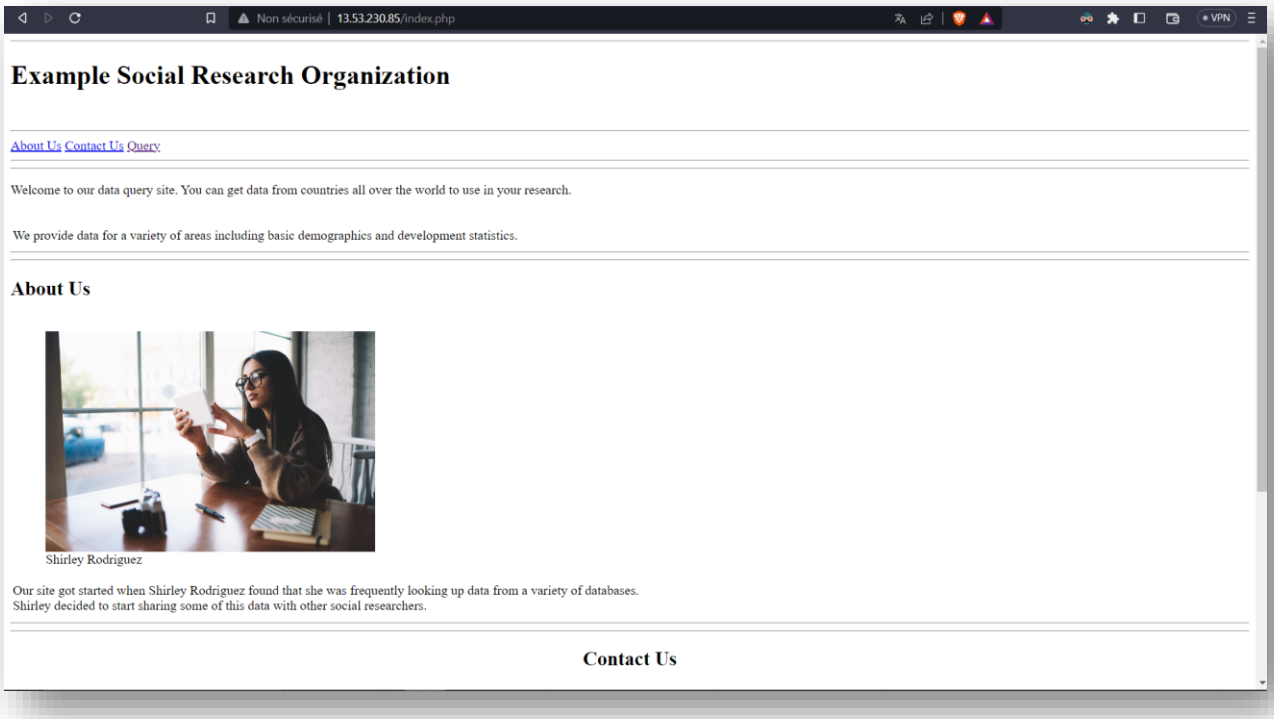



We will perform this following command to import to website (database and frontend in PHP) :

```
#!/bin/bash -ex
yum -y update
amazon-linux-extras install -y lamp-mariadb10.2-php7.2 php7.2
yum install -y httpd mariadb-server
chkconfig httpd on
service httpd start
cd /home/ec2-user
wget https://aws-tc-largeobjects.s3.us-west-2.amazonaws.com/CUR-TF-200-ACACAD-
2/21-course-project/s3/Countrydatadump.sql
chown ec2-user:ec2-user Countrydatadump.sql
cd /var/www/html
wget https://aws-tc-largeobjects.s3.us-west-2.amazonaws.com/CUR-TF-200-ACACAD-
2/21-course-project/s3/Example.zip
unzip Example.zip -d /var/www/html/
chown -R ec2-user:ec2-user /var/www/html
```

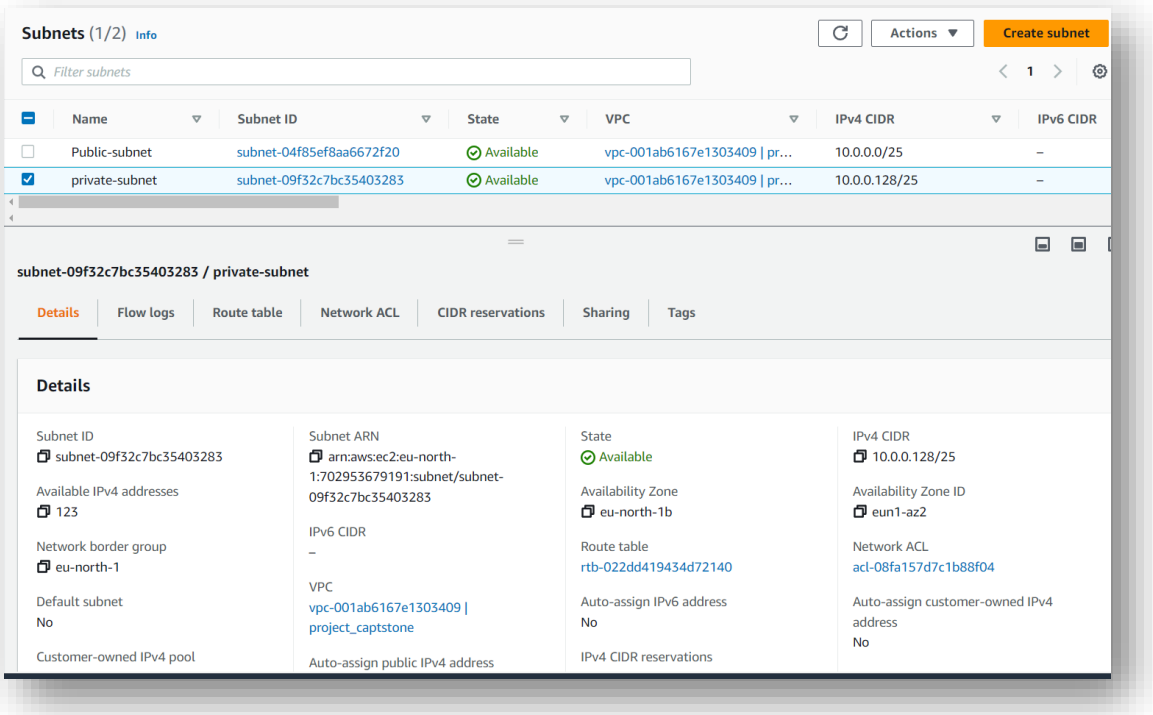
```
[cloudshell-user@ip-10-6-88-211 ~]$ sudo ssh -i "project.pem" cloudshell-user@13.53.230.85
Permission denied (publickey,gssapi-keyex,gssapi-with-mic).
[cloudshell-user@ip-10-6-88-211 ~]$ sudo ssh -i "project.pem" ec2-user@13.53.230.85

       __|  __|_  )
       _|  (     /   Amazon Linux 2 AMI
      ___|\___|___|

https://aws.amazon.com/amazon-linux-2/
:~ $
```

After all this command we can see that our website works in our IPV4 previously created:

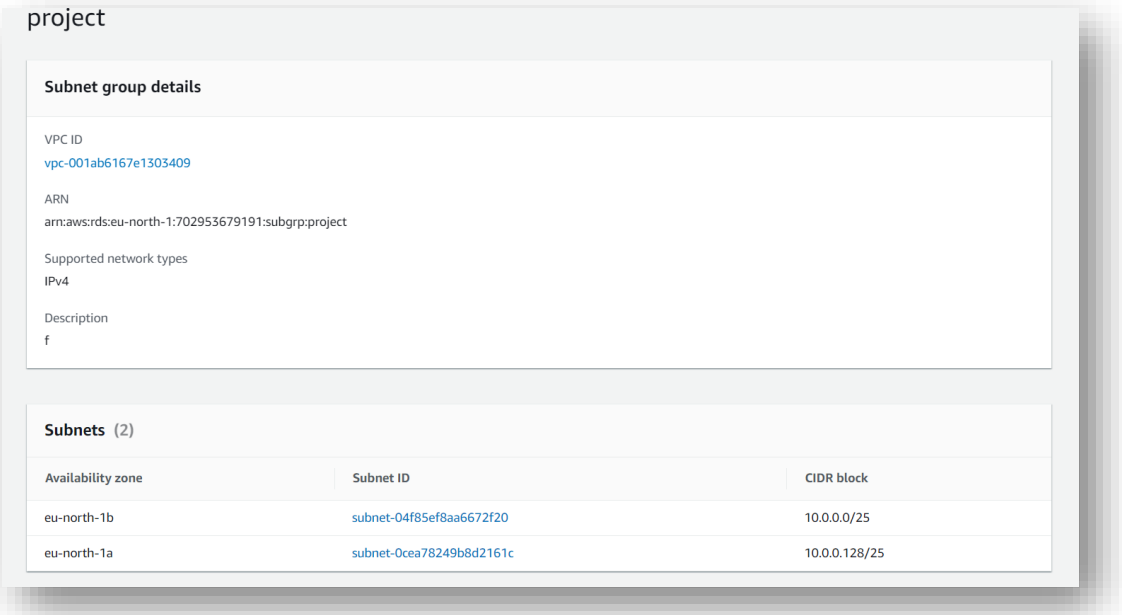


## Database:

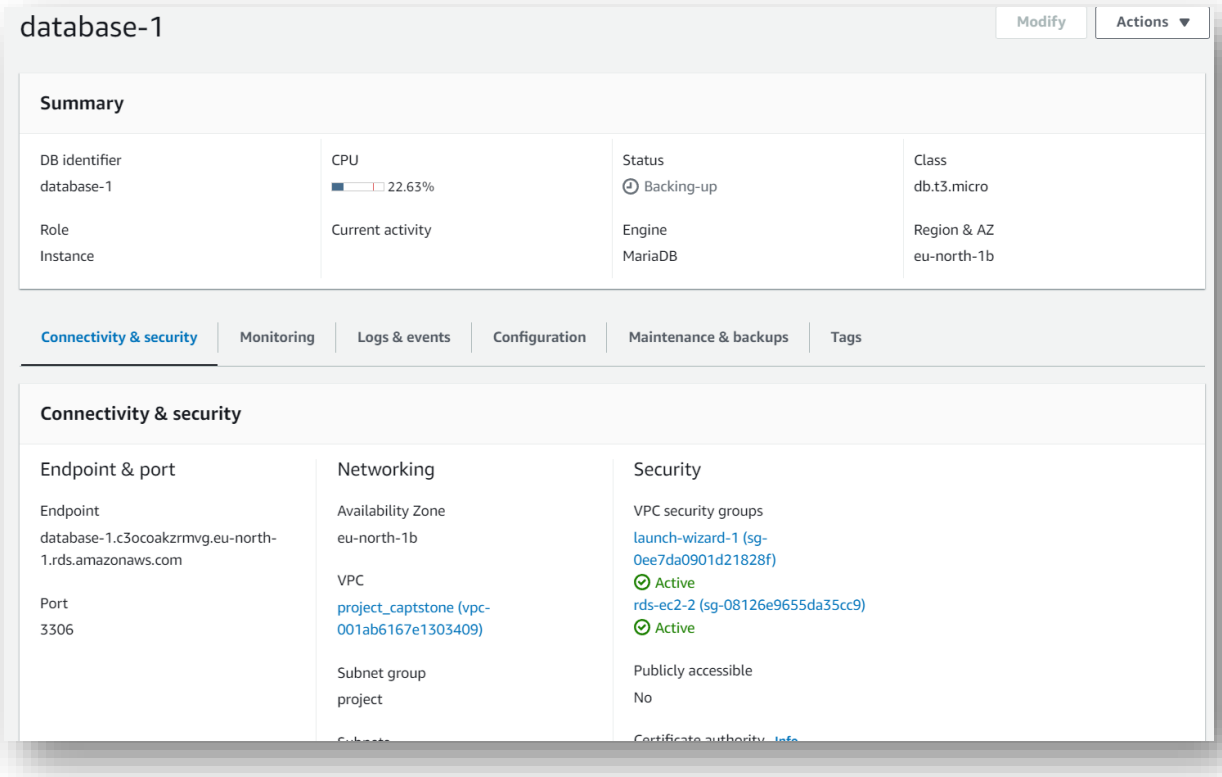
Now we must create another subnet which private, but this time for our Database. Create a private subnet allow us to keep our data safe. We have created this private subnet in another region than the public address. After that we created a group of subnet to connect it to our RDS database.




Once terminate add the subnet group and it's look like this :



So finally that what looks like our database:




## Connection:

We have now to connect with our database:

```
:~ $ mysql -u admin -h database-1.c3ocoakzrmvg.eu-north-1.rds.amazonaws.com -p
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 38
Server version: 10.6.10-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)]> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| innodb             |
| mysql              |
| performance_schema |
| sys                |
+--------------------+
5 rows in set (0.00 sec)
```

We can see that we have all files into our database.

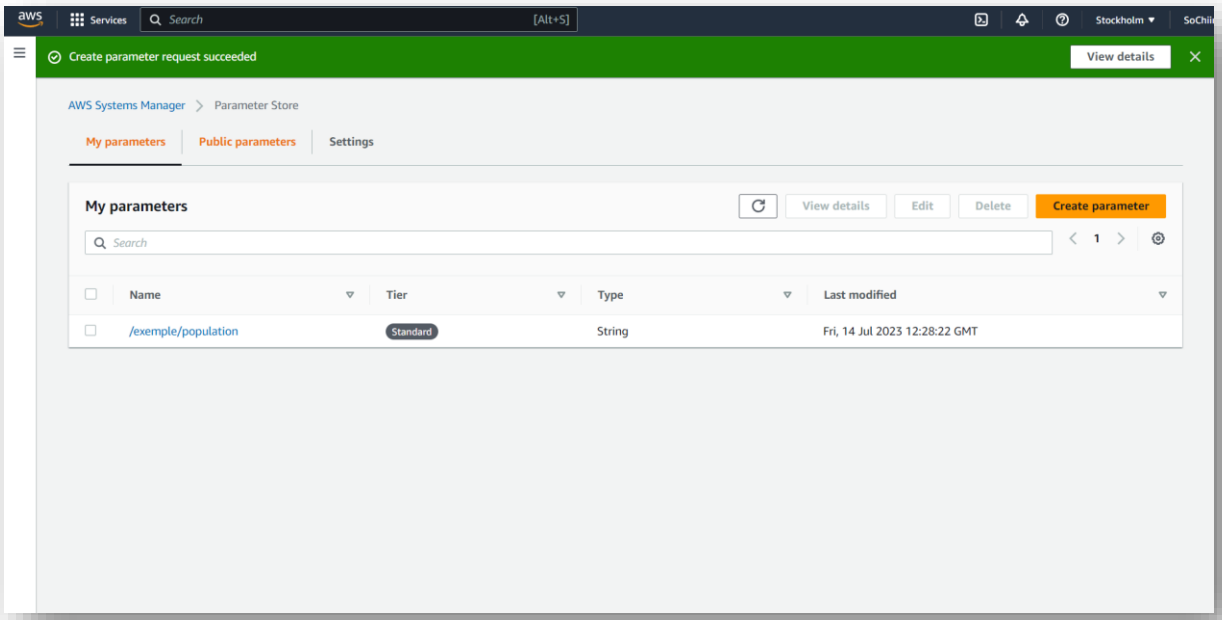
Then we create or own database:

```
MariaDB [(none)]> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| innodb             |
| mysql              |
| performance_schema |
| sys                |
+--------------------+
5 rows in set (0.01 sec)

MariaDB [(none)]> CREATE DATABASE database1;
Query OK, 1 row affected (0.00 sec)

MariaDB [(none)]> show databases;
+--------------------+
| Database           |
+--------------------+
| database1          |
| information_schema |
| innodb             |
| mysql              |
| performance_schema |
| sys                |
+--------------------+
6 rows in set (0.00 sec)

MariaDB [(none)]>
```

We input our SQL file :

```
MariaDB [(none)]> use database1
Database changed
MariaDB [database1]> source Countrydatadump.sql
Query OK, 0 rows affected (0.00 sec)

Query OK, 0 rows affected (0.00 sec)

Query OK, 0 rows affected (0.00 sec)

Query OK, 0 rows affected (0.00 sec)

Query OK, 0 rows affected (0.00 sec)

Query OK, 0 rows affected (0.00 sec)

Query OK, 0 rows affected (0.00 sec)

Query OK, 0 rows affected (0.00 sec)

Query OK, 0 rows affected (0.00 sec)

Query OK, 0 rows affected (0.00 sec)

Query OK, 0 rows affected (0.01 sec)

Query OK, 0 rows affected (0.00 sec)
```

Now, we need to create endpoints to provide access to our website. We have set up these endpoints using AWS Systems Manager. As an example, we created an endpoint named "example/population" to ensure accessibility to population data.



And when we compute the query, we have can see that we have all the data displayed. So, our website and our database is finally connected and we can do some query to this.

| Country Name | Population | Urban Population |
|---|---|---|
| Afghanistan | 26697430 | 5771984 |
| Albania | 3071856 | 1280964 |
| Algeria | 30533827 | 18259229 |
| American Samoa | 57625 | 51171 |
| Andorra | 64634 | 59722 |
| Angola | 13926373 | 6823923 |
| Antigua and Barbuda | 77656 | 24928 |
| Argentina | 36930709 | 33274569 |
| Armenia | 3076098 | 2002540 |
| Aruba | 90271 | 42157 |
| Australia | 19153000 | 16701416 |
| Austria | 8011566 | 5271610 |
| Azerbaijan | 8048600 | 4120883 |
| Bahamas, The | 297651 | 244074 |
| Bahrain | 638193 | 564163 |
| Bangladesh | 129592275 | 30583777 |
| Barbados | 267511 | 97106 |
| Belarus | 10005000 | 6993495 |
| Belgium | 10251250 | 9953964 |
| Belize | 249800 | 119404 |
| Benin | 6517810 | 2496321 |
| Bermuda | 62100 | 62100 |
| Bhutan | 571262 | 145101 |
| Bolivia | 8307248 | 5133879 |
| Bosnia and Herzegovina | 3693698 | 1595678 |
| Botswana | 1757925 | 935216 |
| Brazil | 174425387 | 141633414 |
| Brunei Darussalam | 327036 | 232523 |
| Bulgaria | 8170172 | 5629249 |
| Burkina Faso | 12294012 | 2040806 |
| Burundi | 6374347 | 529071 |
| Cambodia | 12446949 | 2103534 |
| Cameroon | 15678269 | 7823456 |
| Canada | 30769700 | 24461912 |
| Cape Verde | 437238 | 233485 |
| Cayman Islands | 40195 | 40195 |
| Central African Republic | 3701607 | 1391804 |
| Chad | 8222327 | 1924025 |
| Channel Islands | 145330 | 44326 |
| Chile | 15419820 | 13245625 |
| China | 1262645000 | 452026910 |
| Colombia | 39764166 | 28669964 |
| Comoros | 562469 | 158054 |
| Congo, Dem. Rep. | 49626200 | 14788608 |
| Congo, Rep. | 3135773 | 1828156 |
| Costa Rica | 3919180 | 2312316 |
| Cote d'Ivoire | 16581653 | 7213019 |
| Croatia | 4426000 | 2460856 |
| Cuba | 11104313 | 8394861 |
| Curacao | 133860 | 0 |
| Cyprus | 943294 | 647100 |
| Czech Republic | 10272322 | 7601518 |
| Denmark | 5339616 | 4544013 |
| Djibouti | 731930 | 609698 |
| Dominica | 69672 | 49537 |

## Links:

Link of the project: https://github.com/pascalito007/efrei-cloud-bigdata/tree/master/capstone-project

Link of the GitHub: https://github.com/SoChiiro/Cloud-Project