



AWS Cloud & Big Data Architectures - Project

LESIEUX BENJAMIN AND THOMAS MARIOTTE

Table of contents

Task 1:	2
Our architecture:	2
VPC:	3
EC2:	6
Database:	10
Connection:	11
Links:	16

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

Note that I used a dark Reader at the beginning so you can see that my background was black. But I turned off while I display the website, so my background turns white. You can remark that all name of my services are the same.

Task 1:

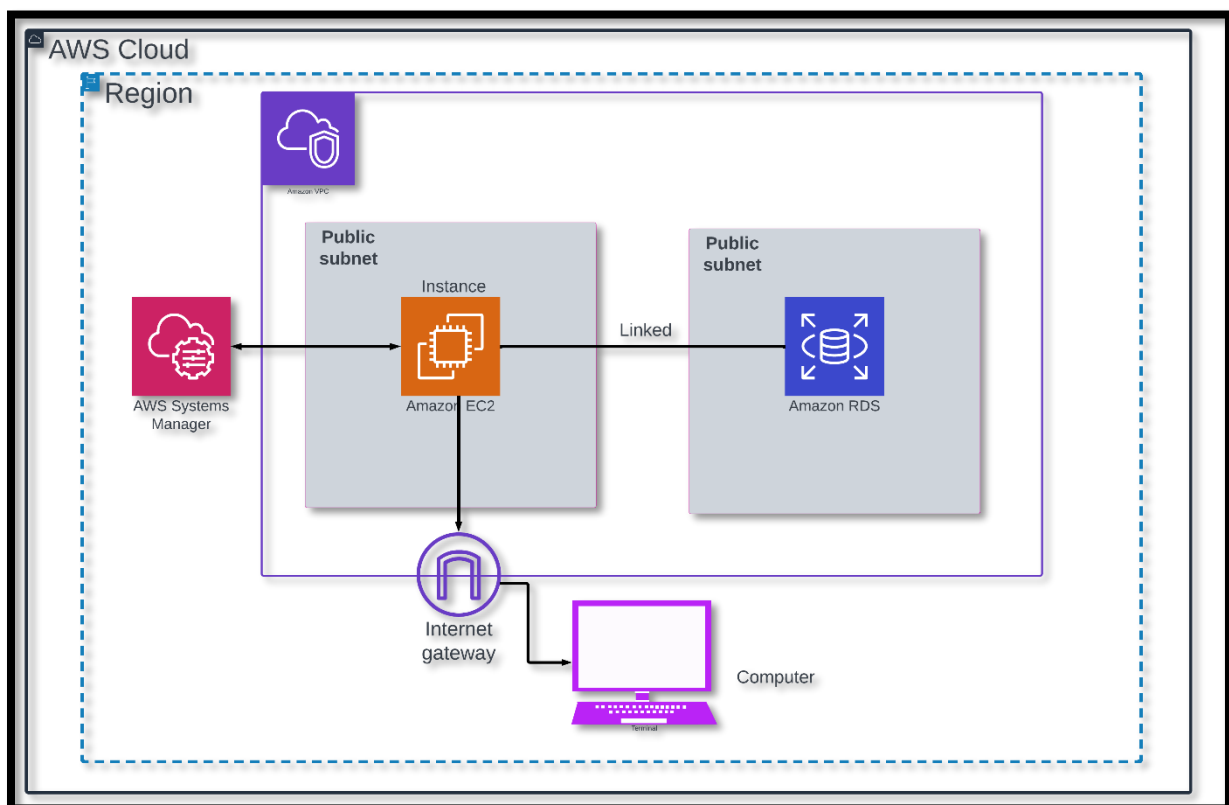
We had to create to deploy an AWS application and make sure that a Social Research Organization have his secure website.

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.

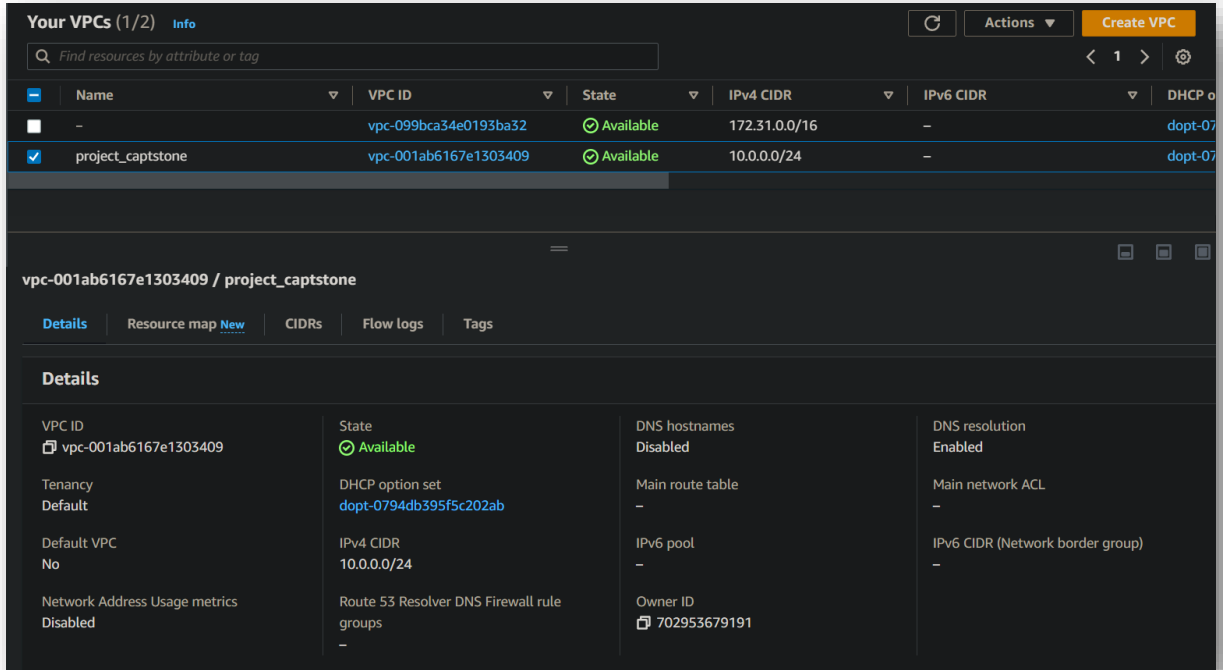
All our data is stored in a private subnet using the Amazon RDS service. The RDS service is configured to work with MariaDB and serves as the storage for all our data. To control access, we utilize the AWS Systems Manager Parameter Store to define the permissions for accessing the data.

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**".



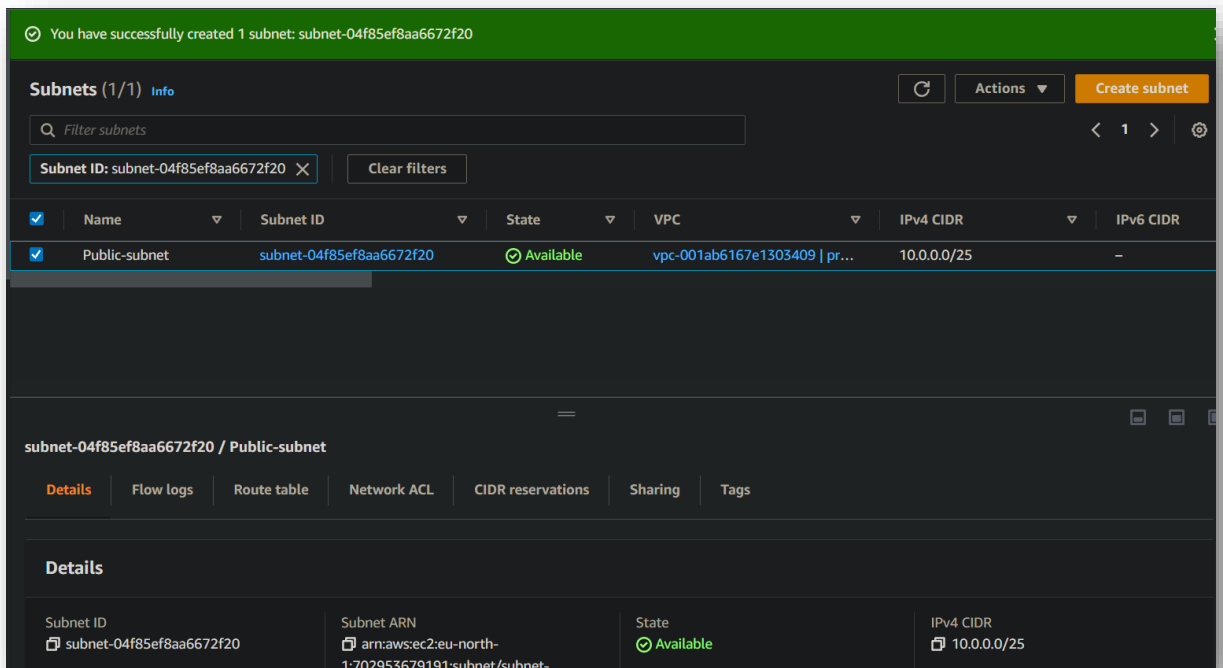
The screenshot shows the AWS Management Console interface for VPCs. At the top, there's a header "Your VPCs (1/2)" with a search bar and a "Create VPC" button. Below this is a table listing VPCs:

Name	VPC ID	State	IPv4 CIDR	IPv6 CIDR	DHCP
-	vpc-099bca34e0193ba32	Available	172.31.0.0/16	-	dopt-07
project_capstone	vpc-001ab6167e1303409	Available	10.0.0.0/24	-	dopt-07

Below the table, the details for the selected VPC "vpc-001ab6167e1303409 / project_capstone" are shown. The tabs include Details, Resource map, CIDRs, Flow logs, and Tags. The Details tab is active, showing the following information:

Property	Value
VPC ID	vpc-001ab6167e1303409
State	Available
Tenancy	Default
Default VPC	No
Network Address Usage metrics	Disabled
DHCP option set	dopt-0794db395f5c202ab
IPv4 CIDR	10.0.0.0/24
Route 53 Resolver DNS Firewall rule groups	-
DNS hostnames	Disabled
Main route table	-
IPv6 pool	-
Owner ID	702953679191
DNS resolution	Enabled
Main network ACL	-
IPv6 CIDR (Network border group)	-

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



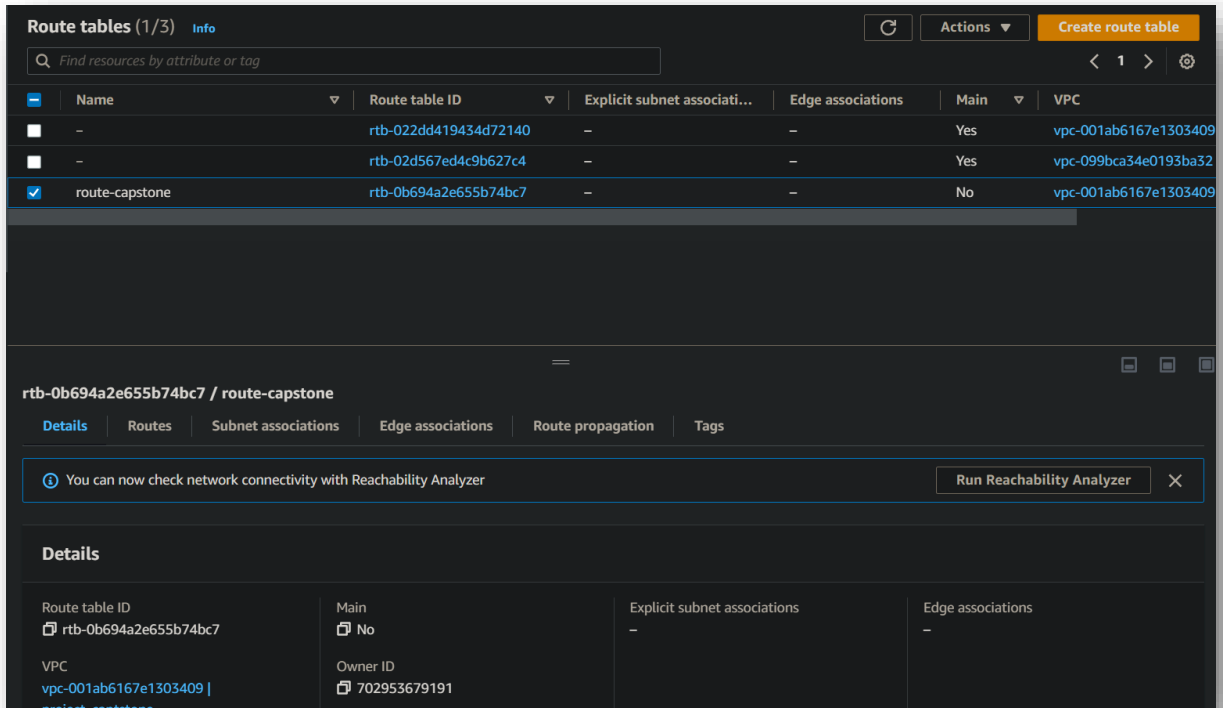
The screenshot shows the AWS Management Console interface for Subnets. At the top, there's a green notification bar: "You have successfully created 1 subnet: subnet-04f85ef8aa6672f20". Below this is a header "Subnets (1/1)" with a search bar and a "Create subnet" button. Below the header is a filter bar showing "Subnet ID: subnet-04f85ef8aa6672f20" and a "Clear filters" button. Below the filter bar is a table listing subnets:

Name	Subnet ID	State	VPC	IPv4 CIDR	IPv6 CIDR
Public-subnet	subnet-04f85ef8aa6672f20	Available	vpc-001ab6167e1303409 pr...	10.0.0.0/25	-

Below the table, the details for the selected subnet "subnet-04f85ef8aa6672f20 / Public-subnet" are shown. The tabs include Details, Flow logs, Route table, Network ACL, CIDR reservations, Sharing, and Tags. The Details tab is active, showing the following information:

Property	Value
Subnet ID	subnet-04f85ef8aa6672f20
Subnet ARN	arn:aws:ec2:eu-north-1:702953679191:subnet/subnet-
State	Available
IPv4 CIDR	10.0.0.0/25

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.



Route tables (1/3) [Info](#)

Find resources by attribute or tag

Name	Route table ID	Explicit subnet associati...	Edge associations	Main	VPC
-	rtb-022dd419434d72140	-	-	Yes	vpc-001ab6167e1303409
-	rtb-02d567ed4c9b627c4	-	-	Yes	vpc-099bca34e0193ba32
<input checked="" type="checkbox"/> route-capstone	rtb-0b694a2e655b74bc7	-	-	No	vpc-001ab6167e1303409

rtb-0b694a2e655b74bc7 / route-capstone

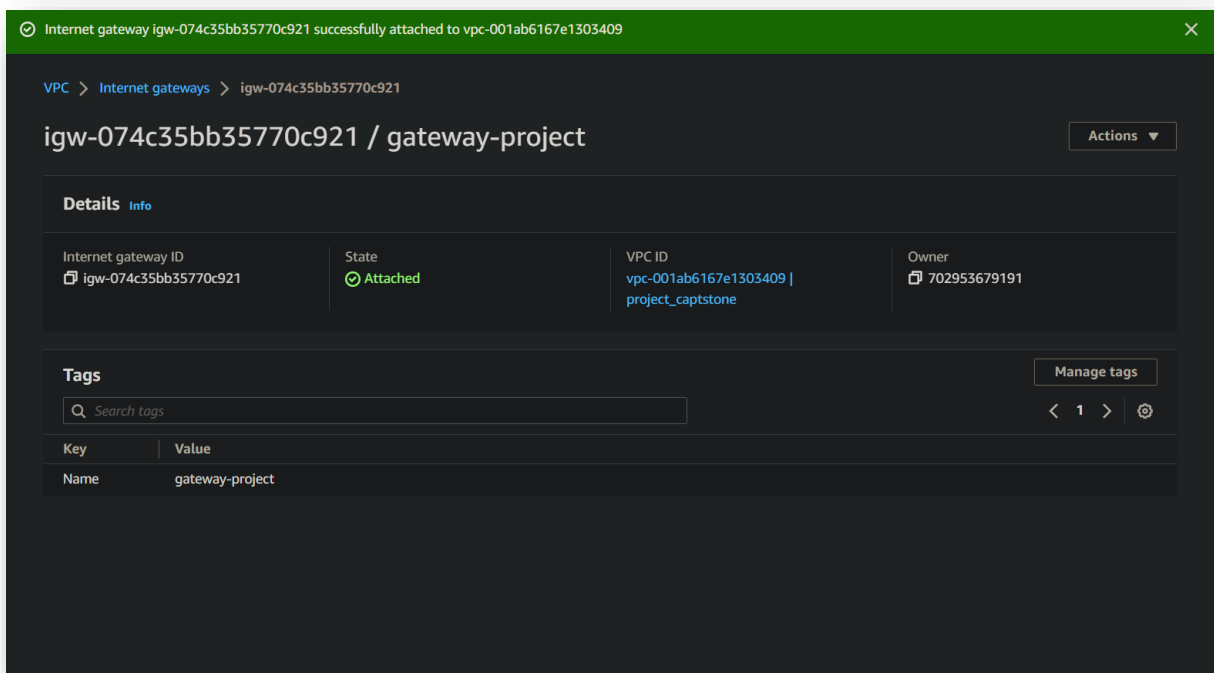
[Details](#) [Routes](#) [Subnet associations](#) [Edge associations](#) [Route propagation](#) [Tags](#)

You can now check network connectivity with Reachability Analyzer [Run Reachability Analyzer](#)

Details

Route table ID rtb-0b694a2e655b74bc7	Main No	Explicit subnet associations -	Edge associations -
VPC vpc-001ab6167e1303409 project_capstone	Owner ID 702953679191		

Then our gateway that we attached to the previous VPC:



Internet gateway igw-074c35bb35770c921 successfully attached to vpc-001ab6167e1303409

VPC > Internet gateways > igw-074c35bb35770c921

igw-074c35bb35770c921 / gateway-project [Actions](#)

Details [Info](#)

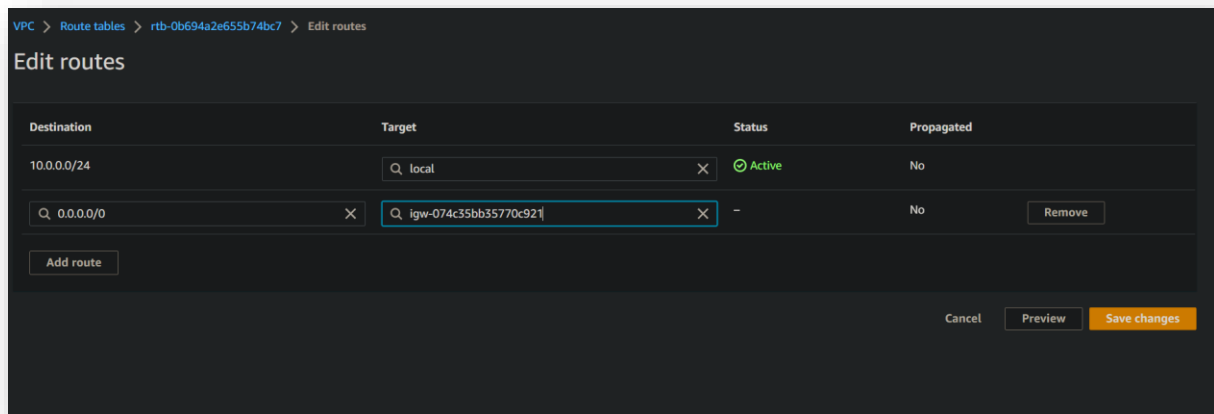
Internet gateway ID igw-074c35bb35770c921	State Attached	VPC ID vpc-001ab6167e1303409 project_capstone	Owner 702953679191
--	-------------------	--	-----------------------

Tags [Manage tags](#)

Search tags

Key	Value
Name	gateway-project

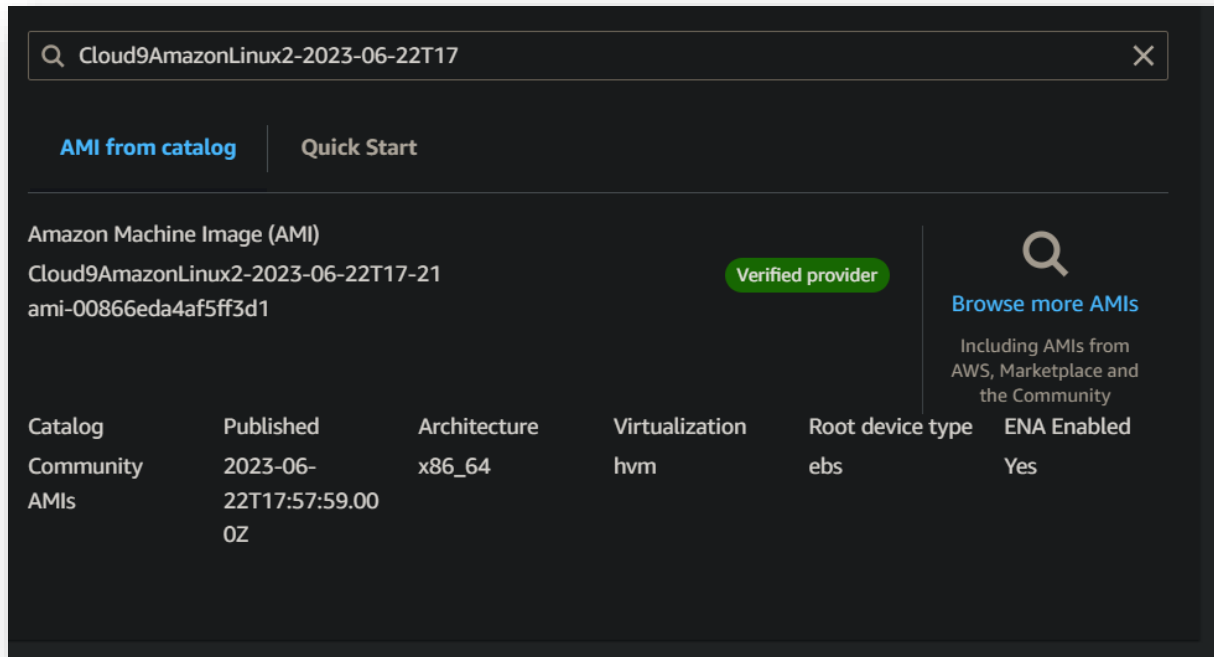
We added our internet gateway to our Route Table in 0.0.0.0/0 (anywhere).



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.

▼ **Network settings** [Info](#)

VPC - *required* [Info](#)

vpc-001ab6167e1303409 (project_capstone)
10.0.0.0/24

Subnet [Info](#)

subnet-04f85ef8aa6672f20 Public-subnet
VPC: vpc-001ab6167e1303409 Owner: 702953679191
Availability Zone: eu-north-1b IP addresses available: 123 CIDR: 10.0.0.0/25

Auto-assign public IP [Info](#)

Disable

Firewall (security groups) [Info](#)

A security group is a set of firewall rules that control the traffic for your instance. Add rules to allow specific traffic to your instance.

☒ Create security group ☐ Select existing security group

Security group name - *required*

launch-wizard-1

This security group will be added to all network interfaces. The name can't be edited after the security group is created. The name must be 1-255 characters. Valid characters: a-z, A-Z, 0-9, spaces, and . _ - / () # , @ [] + = & ; ' ! \$ *

Description - *required* [Info](#)

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

Instances (1/1) [Info](#)

Find instance by attribute or tag (case-sensitive)

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 address
project-capsto...	i-048b83644f0d9505e	Running	t3.micro	2/2 checks passed	No alarms	eu-north-1b	-

Instance: i-048b83644f0d9505e (project-capstone-instance)

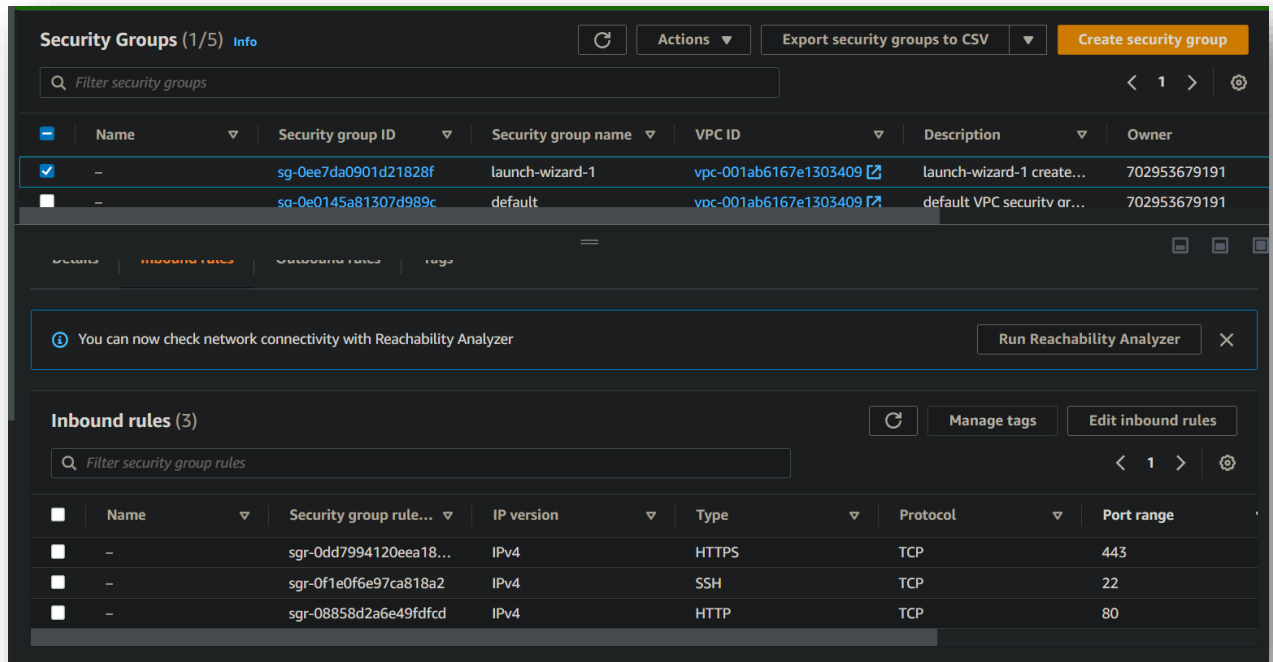
Details | Security | Networking | Storage | Status checks | Monitoring | Tags

▼ **Instance summary** [Info](#)

Instance ID i-048b83644f0d9505e (project-capstone-instance)	Public IPv4 address 13.53.230.85 open address	Private IPv4 addresses 10.0.0.122
IPv6 address -	Instance state Running	Public IPv4 DNS -
Hostname type IP name: ip-10-0-0-122.eu-north-1.compute.internal	Private IP DNS name (IPv4 only) ip-10-0-0-122.eu-north-1.compute.internal	Elastic IP addresses 13.53.230.85 [Public IP]
Answer private resource DNS name -	Instance type t3.micro	AWS Compute Optimizer finding Opt-in to AWS Compute Optimizer for recommendations. Learn more
Auto-assigned IP address -	VPC ID vpc-001ab6167e1303409 (project_capstone)	

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) :

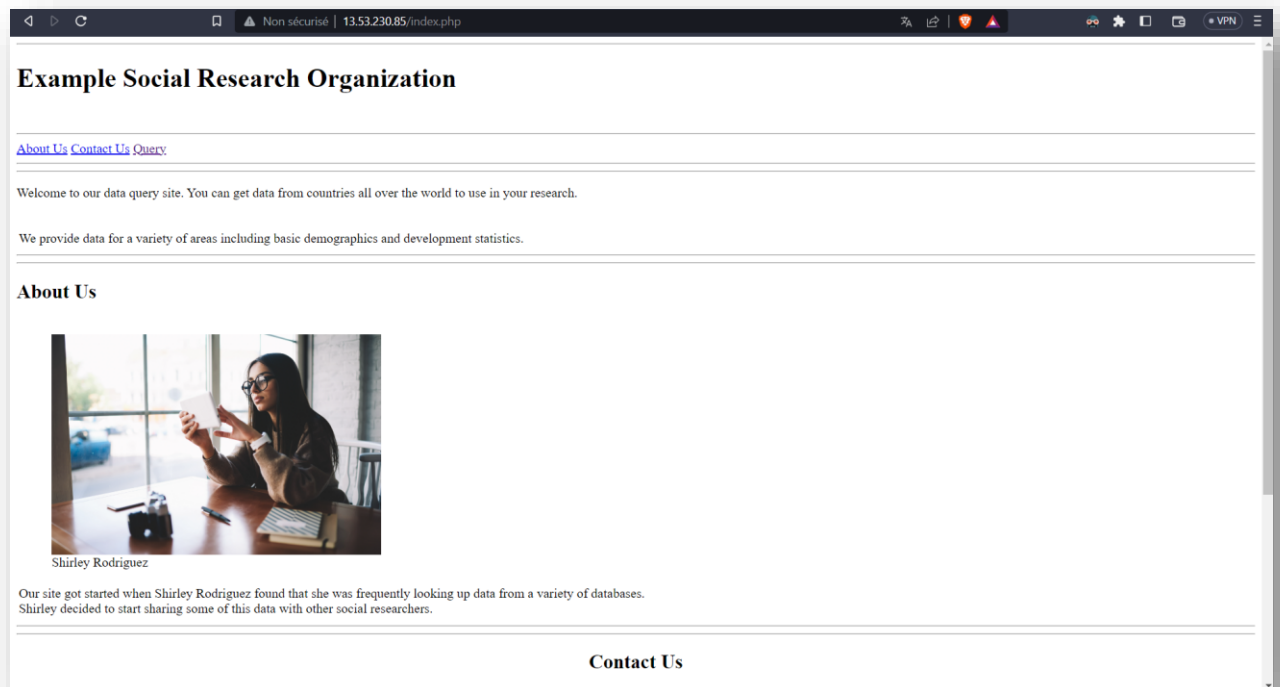
```
#!/bin/bash -ex
yum -y update
amazon-linux-extras install -y lamp-mariadb10.2-php7.2 php7.2
yum install -y httpd mariadb-server
chkconfig httpd on
service httpd start
cd /home/ec2-user
wget https://aws-tc-largeobjects.s3.us-west-2.amazonaws.com/CUR-TF-200-ACACAD-2/21-course-project/s3/Countrydatadump.sql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

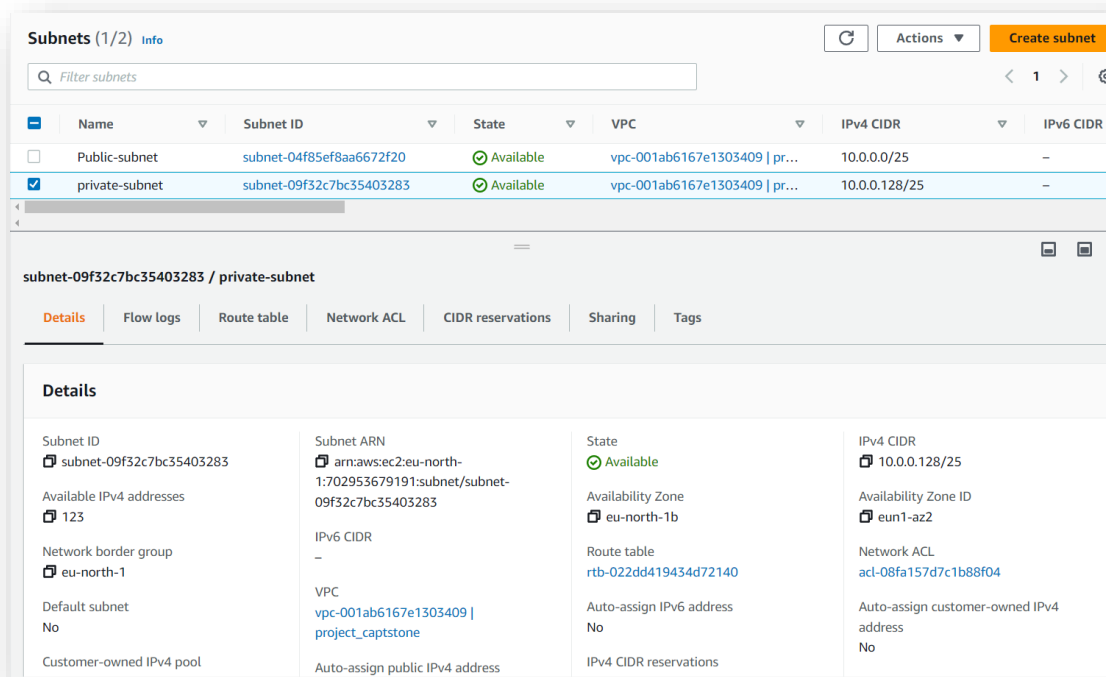
https://aws.amazon.com/amazon-linux-2/
:~ $
```

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.



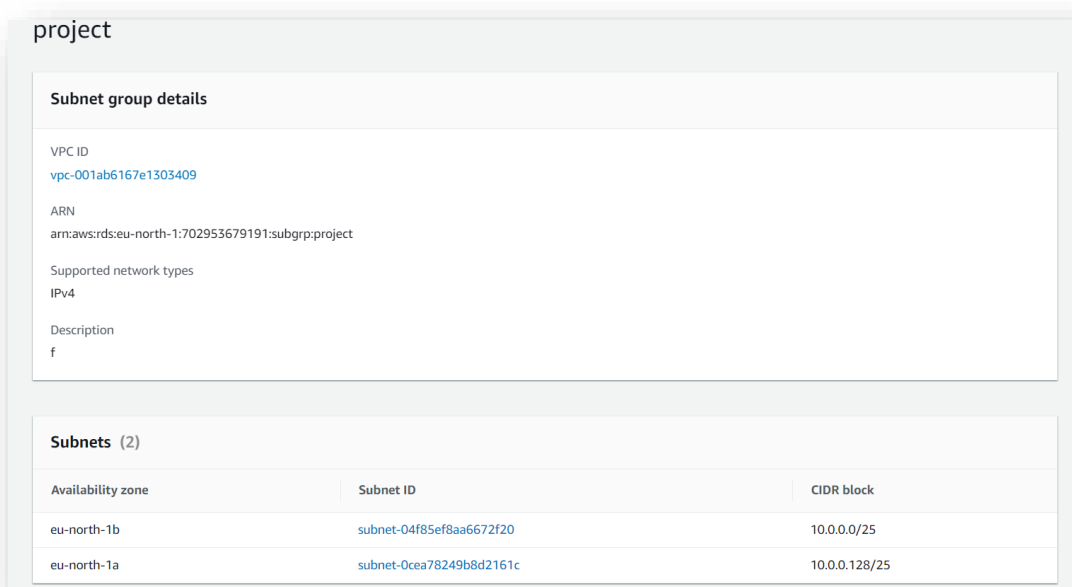
The screenshot shows the AWS Subnets console. At the top, there's a search bar and a table of subnets. The table has columns: Name, Subnet ID, State, VPC, IPv4 CIDR, and IPv6 CIDR. Two subnets are listed: 'Public-subnet' and 'private-subnet'. The 'private-subnet' is selected, and its details are shown below.

Name	Subnet ID	State	VPC	IPv4 CIDR	IPv6 CIDR
Public-subnet	subnet-04f85ef8aa6672f20	Available	vpc-001ab6167e1303409 pr...	10.0.0.0/25	-
private-subnet	subnet-09f32c7bc35403283	Available	vpc-001ab6167e1303409 pr...	10.0.0.128/25	-

The details for the selected subnet 'subnet-09f32c7bc35403283 / private-subnet' are shown below:

Details			
Subnet ID	Subnet ARN	State	IPv4 CIDR
subnet-09f32c7bc35403283	arn:aws:ec2:eu-north-1:702953679191:subnet/subnet-09f32c7bc35403283	Available	10.0.0.128/25
Available IPv4 addresses	IPV6 CIDR	Availability Zone	Availability Zone ID
123	-	eu-north-1b	eun1-az2
Network border group	VPC	Route table	Network ACL
eu-north-1	vpc-001ab6167e1303409 project_captstone	rtb-022dd419434d72140	acl-08fa157d7c1b88f04
Default subnet	Auto-assign IPv6 address	Auto-assign IPv6 address	Auto-assign customer-owned IPv4 address
No	No	No	No
Customer-owned IPv4 pool	Auto-assign public IPv4 address	IPv4 CIDR reservations	

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



The screenshot shows the AWS Subnet group details page. It displays the VPC ID, ARN, supported network types, and a list of subnets.

Subnet group details

VPC ID
vpc-001ab6167e1303409

ARN
arn:aws:rds:eu-north-1:702953679191:subgrp:project

Supported network types
IPv4

Description
f

Subnets (2)

Availability zone	Subnet ID	CIDR block
eu-north-1b	subnet-04f85ef8aa6672f20	10.0.0.0/25
eu-north-1a	subnet-0cea78249b8d2161c	10.0.0.128/25

So finally that what looks like our database:

database-1 Modify Actions ▼

Summary			
DB identifier database-1	CPU <div><div></div></div> 22.63%	Status ⌚ Backing-up	Class db.t3.micro
Role Instance	Current activity	Engine MariaDB	Region & AZ eu-north-1b

[Connectivity & security](#) | [Monitoring](#) | [Logs & events](#) | [Configuration](#) | [Maintenance & backups](#) | [Tags](#)

Connectivity & security		
Endpoint & port Endpoint database-1.c3ocoakzrmvg.eu-north-1.rds.amazonaws.com Port 3306	Networking Availability Zone eu-north-1b VPC project_captstone (vpc-001ab6167e1303409) Subnet group project	Security VPC security groups launch-wizard-1 (sg-0ee7da0901d21828f) ⬢ Active rds-ec2-2 (sg-08126e9655da35cc9) ⬢ Active Publicly accessible No Certificate authority: default

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 our 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)

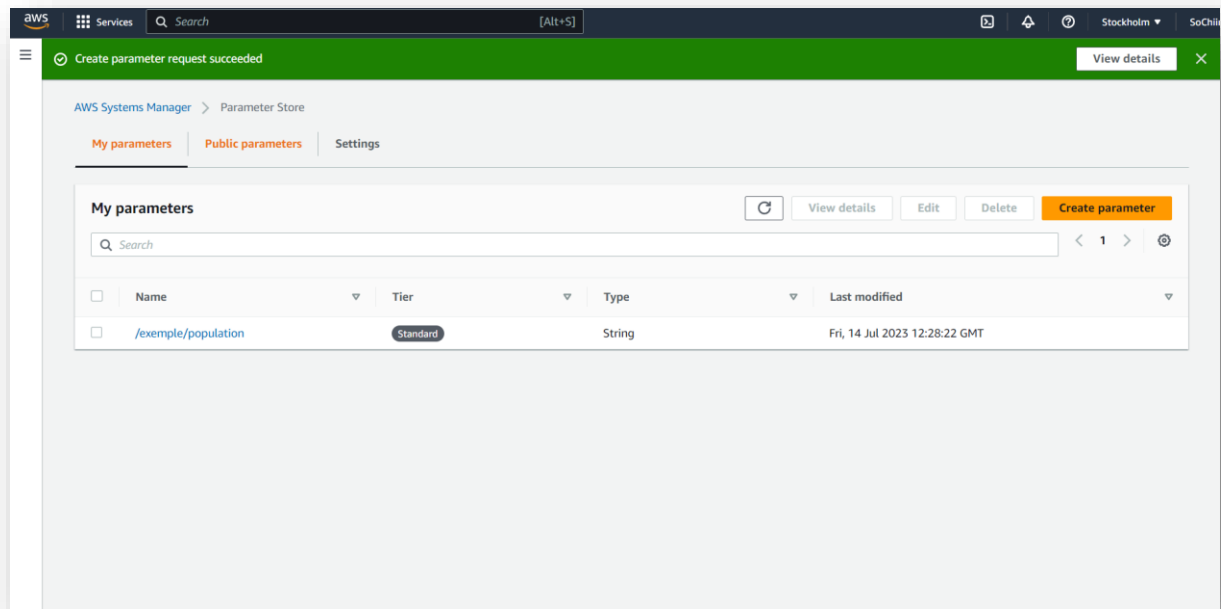
Query OK, 0 rows affected (0.00 sec)

Query OK, 0 rows affected (0.00 sec)

Query OK, 0 rows affected (0.01 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
Curaçao	133860	0
Cyprus	943294	647300
Czech Republic	10272322	7605518
Denmark	5339616	4544013
Djibouti	731930	609696
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/SoChihiro/Cloud-Project>