DEPLOYING WORDPRESS IN A VPC

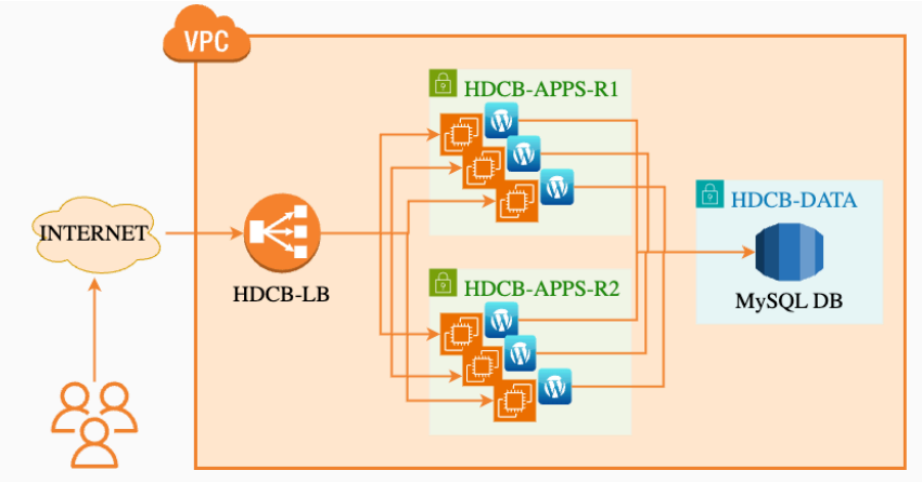
1. INTRODUCTION

The aim of this exercise is to deploy wordpress in a VPC in order to learn about network configuration in AWS. Through the development of this exercise, we will explore several concepts like VPCs, subnets, Route Tables, Internet Gateway and Load Balancers.

We will also learn how to deploy EC2 and RDS instances.

The idea is to create a VPC that contains three subnets (two subnets for the application, named R1 and R2, and the other for the database).
The users have to be able to make queries through the load balancer and the load balancer will then be used to communicate the traffic to the EC2 instances

This is the architecture of the exercise:

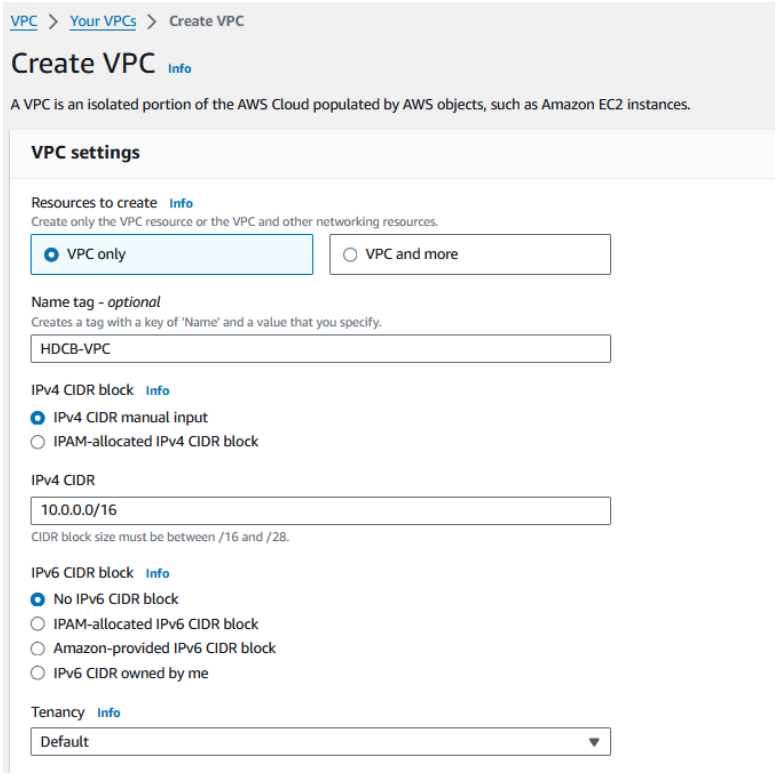


2. NETWORKING

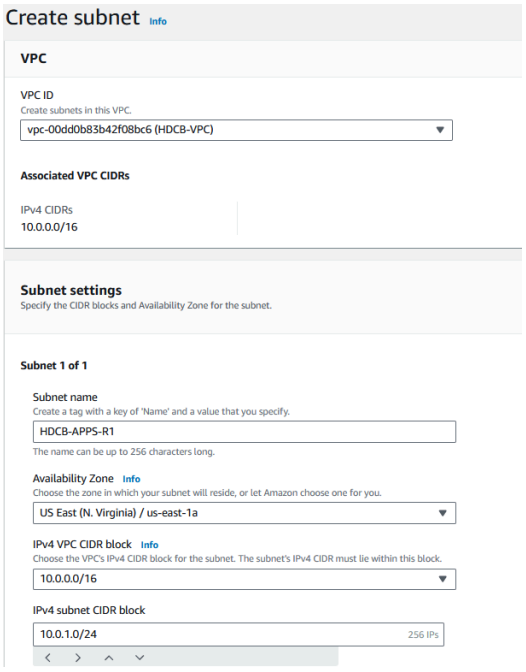
A) VPC AND SUBNETS

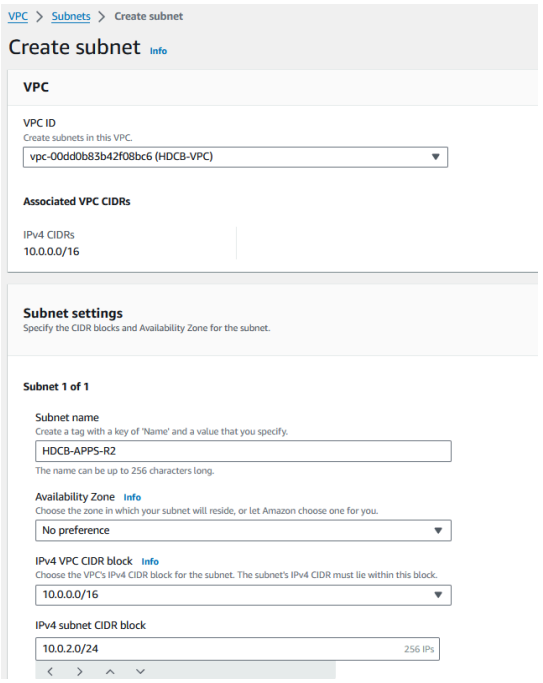
Amazon Virtual Private Cloud (VPC) is a virtual network infrastructure provided that allows you to create isolated sections of the AWS cloud where you can launch AWS resources in a logically isolated environment. Subnets, on the other hand, are subdivisions of a VPC's IP address range and are used to segment and organize the resources within a VPC.

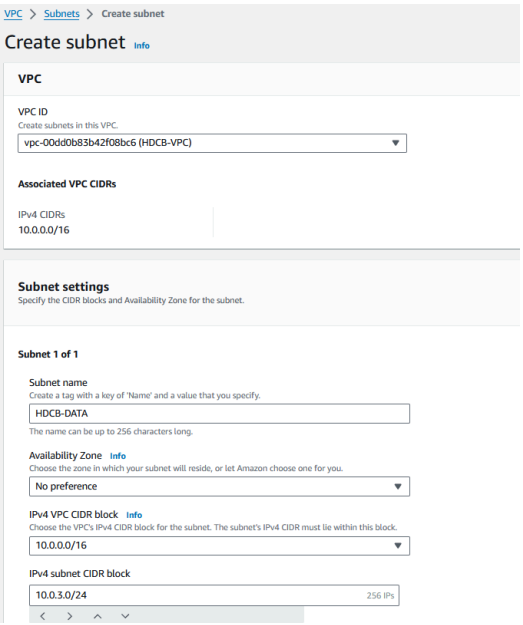
We create the VPC named “HDCB-VPC”:



After creating the VPC we have to create the subnets that reside inside that VPC:



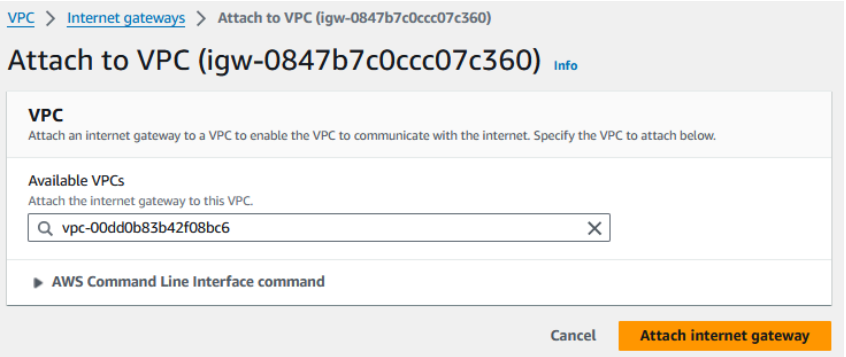




Load balancers need to use at least two subnets, and to make it more real we will place the subnets in different zones. In that way, if an ability zone is down, the architecture will still work in another availability zone.

These subnets are now completely isolated, so we have to create an Internet Gateway. An Internet Gateway in AWS provides connectivity between a VPC and the internet, enabling resources within the VPC to access internet resources and allowing internet-based users or services to access resources within the VPC.
Internet Gateways are typically used for resources in public subnets within a VPC, where those resources require direct connectivity to the internet. Publicly accessible services, such as web servers or load balancers, often reside in public subnets with an associated route to the Internet Gateway.

After creating it, we have to attach the internet Gateway to our VPC:



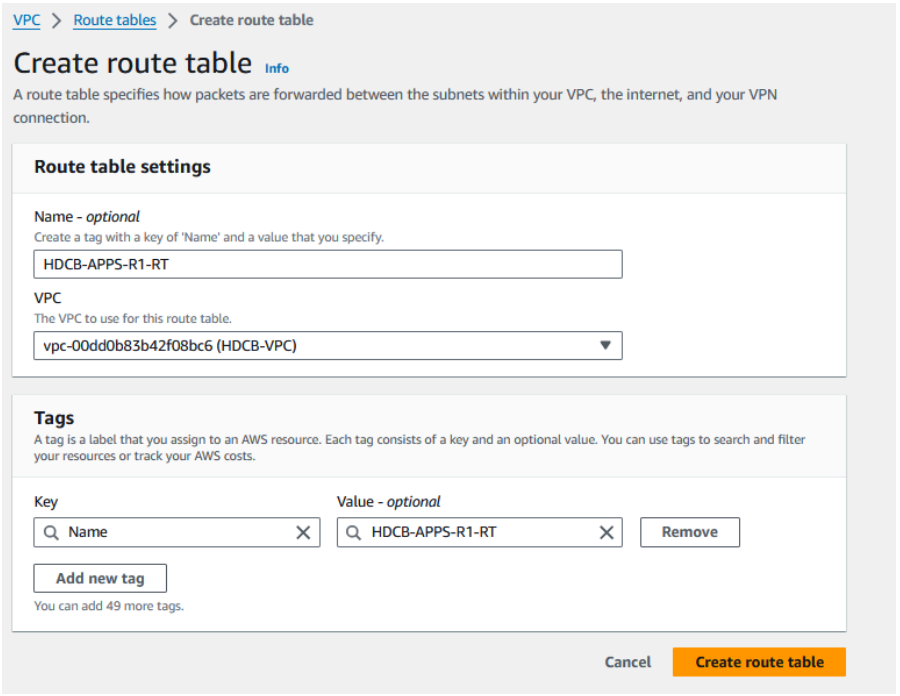
Now that we created the Internet Gateway we must connect it to the subnets via the Route Tables.

B) ROUTE TABLES

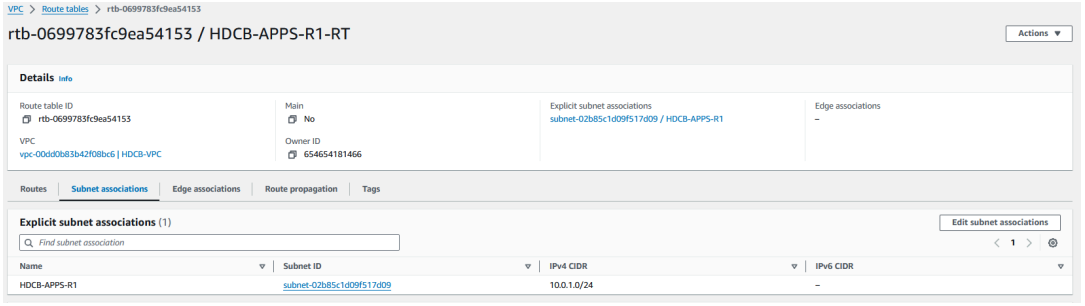
In AWS, a route table is a set of rules, called routes, that are used to determine where network traffic from the VPC should be directed. Each subnet in a VPC must be associated with a route table, and the route table controls the routing of traffic within that subnet.

We have to create two Route Tables in our VPC, one for the DCB-APPS-R1 subnet and the other for the DCB-APPS-R2 subnet. Both need to access the internet. The database does not need to access the internet, so we do not need to route HDCB-DATA subnet.

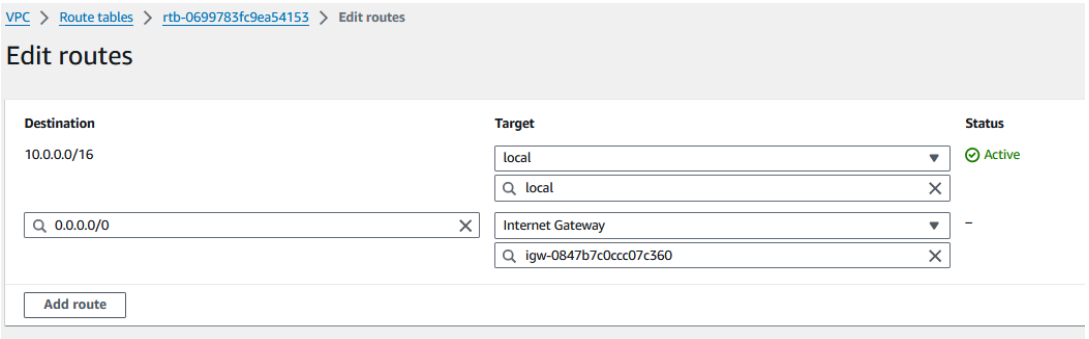
We will exemplify the creation of the route table for DCB-APPS-R1, although the same process is used for DCB-APPS-R2:



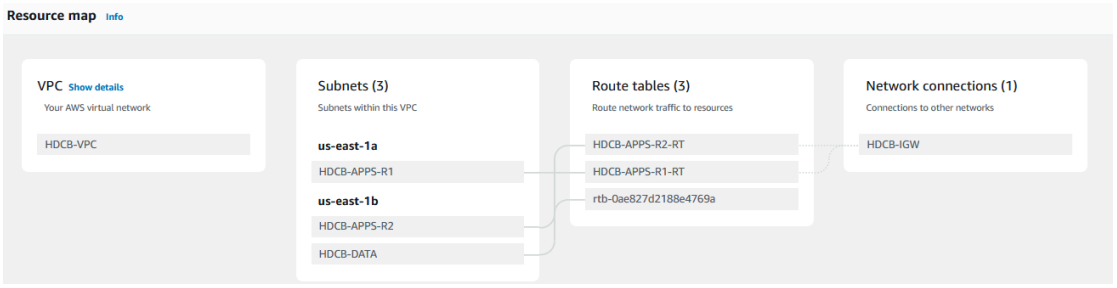
After creating the route tables we make the association with the correspondent subnet:



Then we add a new route to allow all traffic throughout the internet gateway specified before:

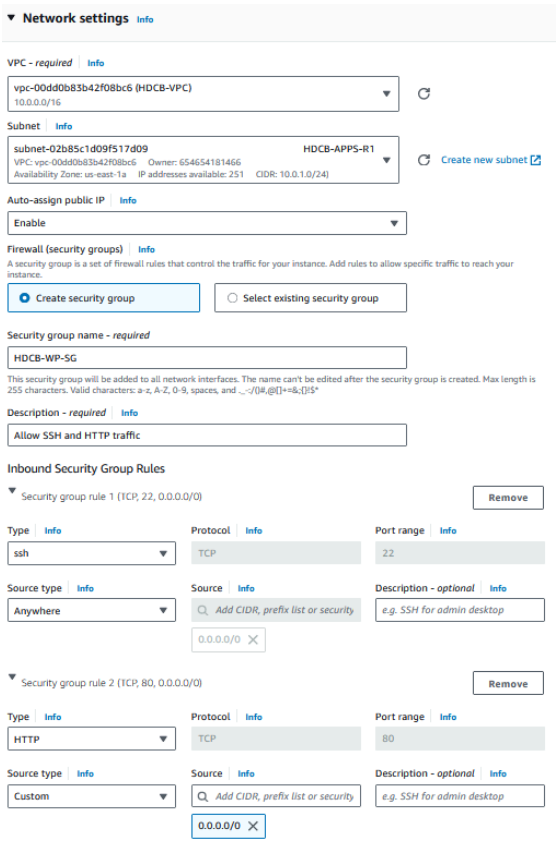


The following scheme shows the final network configuration:



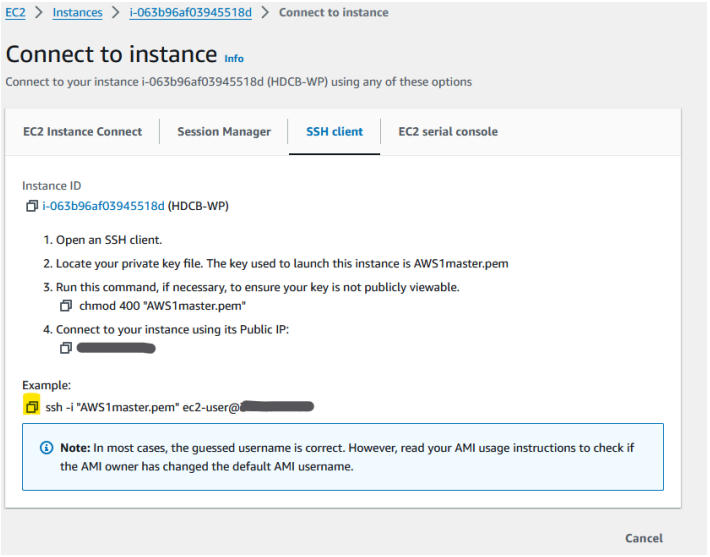
3. DEPLOYING AN EC2 INSTANCE

We launch and EC2 instance with the name “HDCB-WP”, keeping the default AMI and configurations. In the networking part, we create a new security group as follows:



In AWS, security groups act as virtual firewalls for EC2 instances and other resources within the VPC. They basically control inbound and outbound traffic at the instance level, and this enables us to specify which traffic is allowed to reach our instances.

Afterwards, we go to our terminal and connect to the instance by copying the example command and using our private key:



In this virtual machine, we will install wordpress:

vi install_wp.sh

We type “i” and insert the following code:

```
#!/bin/bash
# sudo bash install_wp.sh

# Check if the script is run with sudo
if [ "$(id -u)" -ne 0 ]; then
        echo "Please run this script with sudo or as root."
        exit 1
fi

# Install necessary packages
sudo dnf install -y wget php-mysqlnd httpd php-fpm php-mysqli php-json php php-devel php-gd

# Download and extract WordPress
cd /tmp
wget https://wordpress.org/latest.tar.gz
tar -xzf latest.tar.gz

# Start and Enable Apache
sudo systemctl start --now httpd

# Move WordPress files to the web directory
cp -r wordpress/* /var/www/html/

# Configure Apache
sed -i 's/AllowOverride None/AllowOverride All/g' /etc/httpd/conf/httpd.conf

# Set permissions
chown -R apache:apache /var/www
chmod 2775 /var/www

# Restart Apache
systemctl restart httpd
```

To save the bash script we press esc and type :wq, and then press enter.

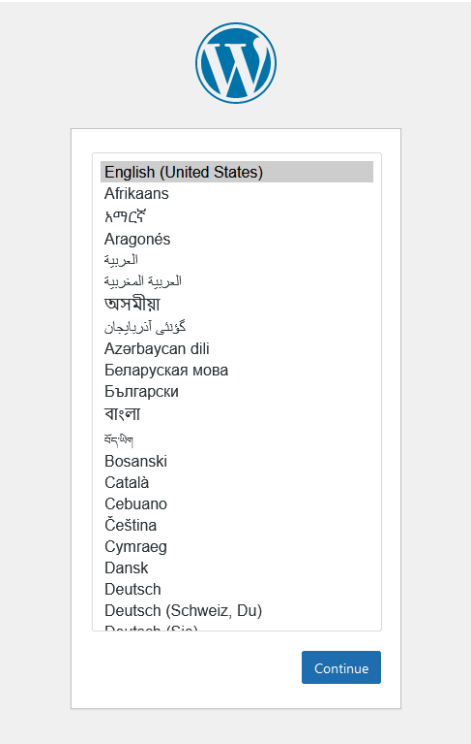
To install wordpress we have to run the bash script we just saved. We do that by executing the following command:

sudo bash install_wp.sh

Then, we copy the public IP of the instance and go to the browser:

http://PUBLIC_IP

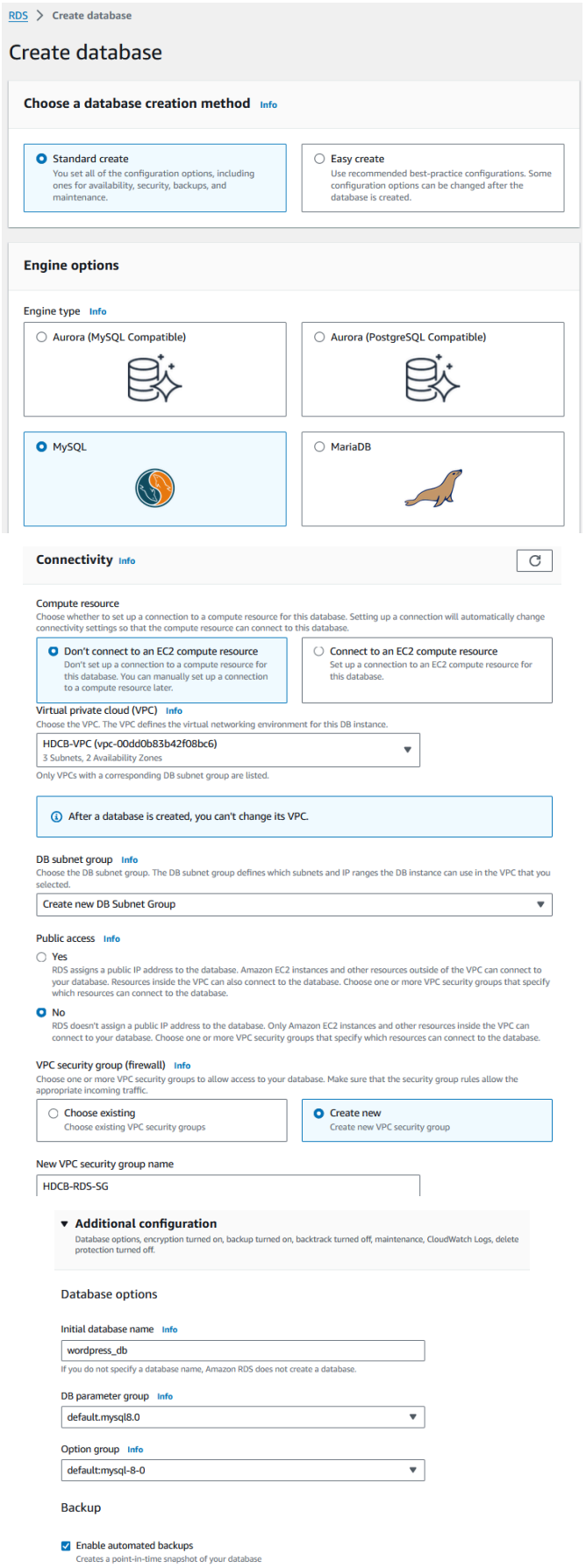
We will see that we are accessing the installation of wordpress:



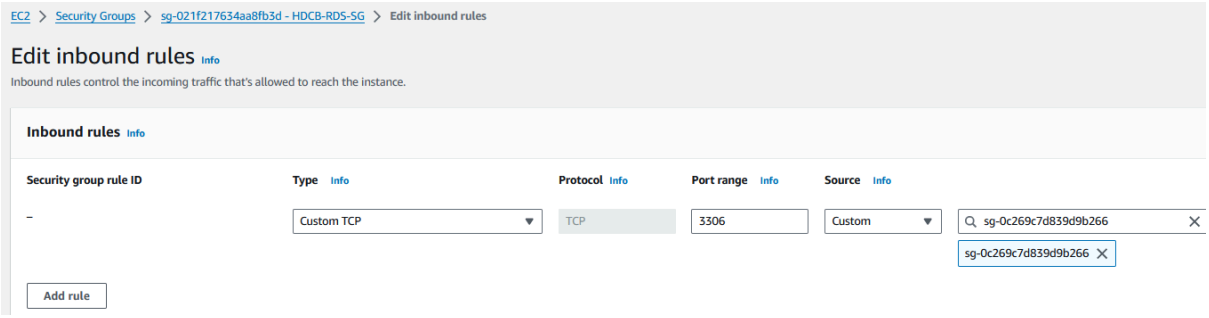
4. RDS AND DATABASE TESTING

Amazon RDS is a fully managed relational database service that simplifies the process of deploying, operating, and scaling relational databases in the cloud.

We go to the RDS service and create a database:

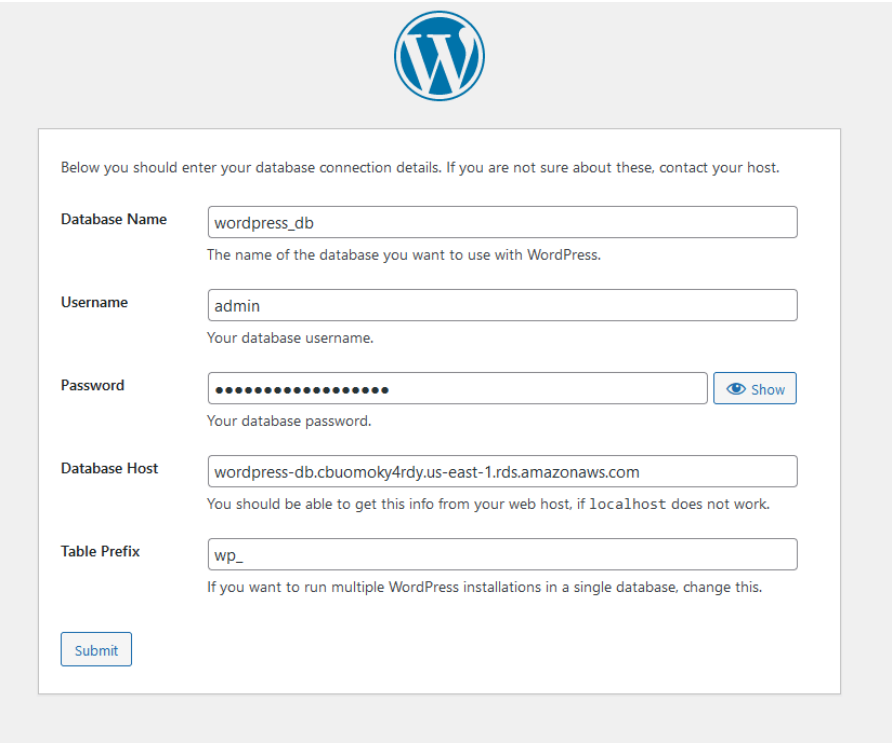

RDS > Create database
Create database
Choose a database creation method Info
Standard create
You set all of the configuration options, including ones for availability, security, backups, and maintenance.
Easy create
Use recommended best-practice configurations. Some configuration options can be changed after the database is created.
Engine options
Engine type Info
Aurora (MySQL Compatible)
Aurora (PostgreSQL Compatible)
MySQL
MariaDB
Connectivity Info
Compute resource
Choose whether to set up a connection to a compute resource for this database. Setting up a connection will automatically change connectivity settings so that the compute resource can connect to this database.
Don't connect to an EC2 compute resource
Don't set up a connection to a compute resource for this database. You can manually set up a connection to a compute resource later.
Connect to an EC2 compute resource
Set up a connection to an EC2 compute resource for this database.
Virtual private cloud (VPC) Info
Choose the VPC. The VPC defines the virtual networking environment for this DB instance.
HDCB-VPC (vpc-00dd0b83b42f08bc6)
3 Subnets, 2 Availability Zones
Only VPCs with a corresponding DB subnet group are listed.
After a database is created, you can't change its VPC.
DB subnet group Info
Choose the DB subnet group. The DB subnet group defines which subnets and IP ranges the DB instance can use in the VPC that you selected.
Create new DB Subnet Group
Public access Info
Yes
RDS assigns a public IP address to the database. Amazon EC2 instances and other resources outside of the VPC can connect to your database. Resources inside the VPC can also connect to the database. Choose one or more VPC security groups that specify which resources can connect to the database.
No
RDS doesn't assign a public IP address to the database. Only Amazon EC2 instances and other resources inside the VPC can connect to your database. Choose one or more VPC security groups that specify which resources can connect to the database.
VPC security group (firewall) Info
Choose one or more VPC security groups to allow access to your database. Make sure that the security group rules allow the appropriate incoming traffic.
Choose existing
Choose existing VPC security groups
Create new
Create new VPC security group
New VPC security group name
HDCB-RDS-SG
Additional configuration
Database options, encryption turned on, backup turned on, backtrack turned off, maintenance, CloudWatch Logs, delete protection turned off.
Database options
Initial database name Info
wordpress_db
If you do not specify a database name, Amazon RDS does not create a database.
DB parameter group Info
default.mysql8.0
Option group Info
default:mysql-8-0
Backup
Enable automated backups
Creates a point-in-time snapshot of your database

Next, we edit the security group that was automatically created. We delete the inbound rule and create a new one that allows all traffic coming from the wordpress security group:

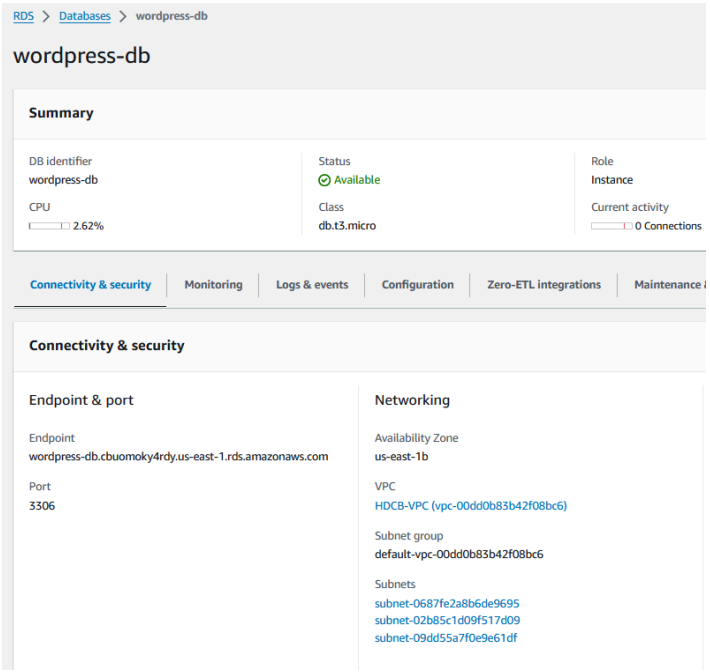


This means that all the instances attached to this HDCB-WP-SG security group are going to be allowed to send traffic to this port. In this case, we are only accepting inbound traffic so we can delete the outbound traffic set by default.

Now, let's test that everything works by trying to connect to the database using this configuration:



Note that the database host is the endpoint we got when creating the database:

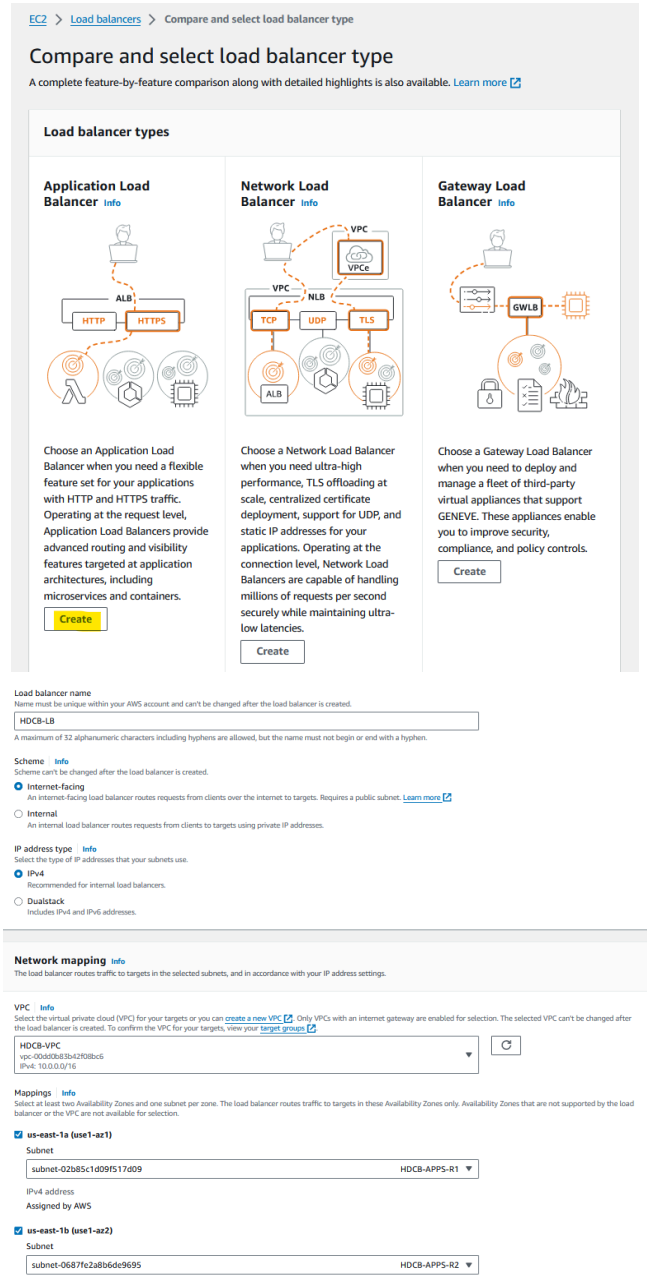


5. LOAD BALANCER CONFIGURATION

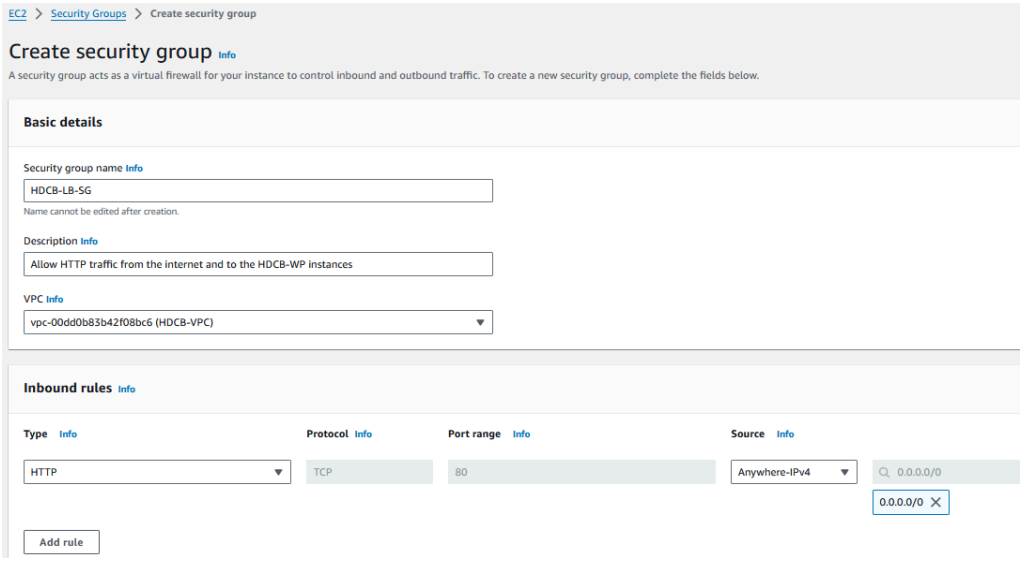
When we deploy an EC2 instance, the IP is dynamic, which means that if we stop the instance or re-launch it, this IP can change. For a web service, having a dynamic IP is not the best thing, since we would have to reconfigure the web service for every time our IP changes. In this case, we can create an elastic IP (which is static) and assign it to our instances.

A load balancer is something that has an elastic IP and also is able to distribute the traffic across different services. Load balancers are critical components in distributed computing and networking architectures since they help to distribute incoming network traffic across multiple servers (or EC2 instances in our case) to ensure efficient use of resources, optimize performance, and enhance the availability and reliability of applications.

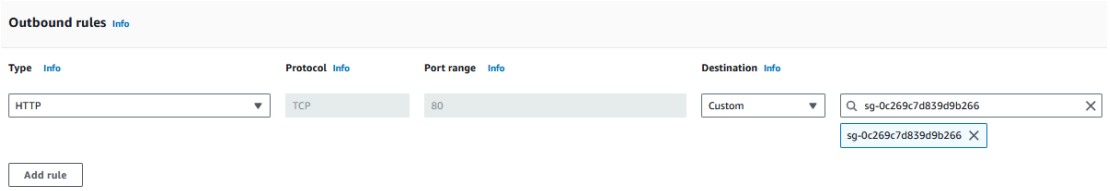
We create an application load balancer:



We also create a security group for the load balancer. We want everyone to be able to access the load balancer from the internet (HTTP) using port 80:

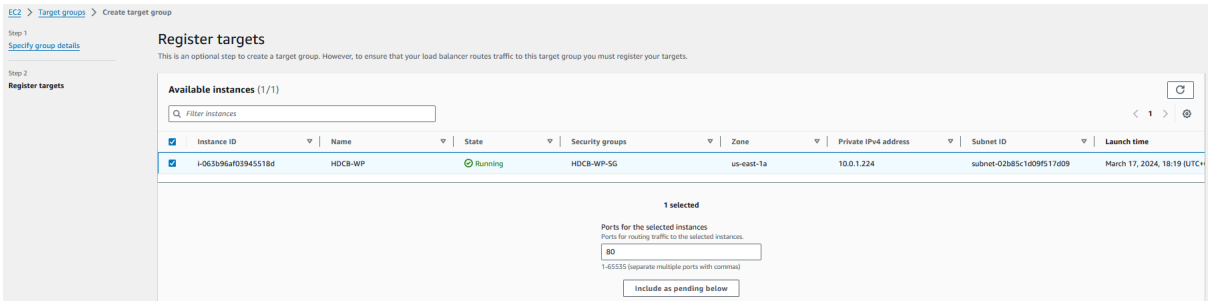


However, the outbound traffic should only be redirected to the wordpress instances, so the destination should be the security group of wordpress:



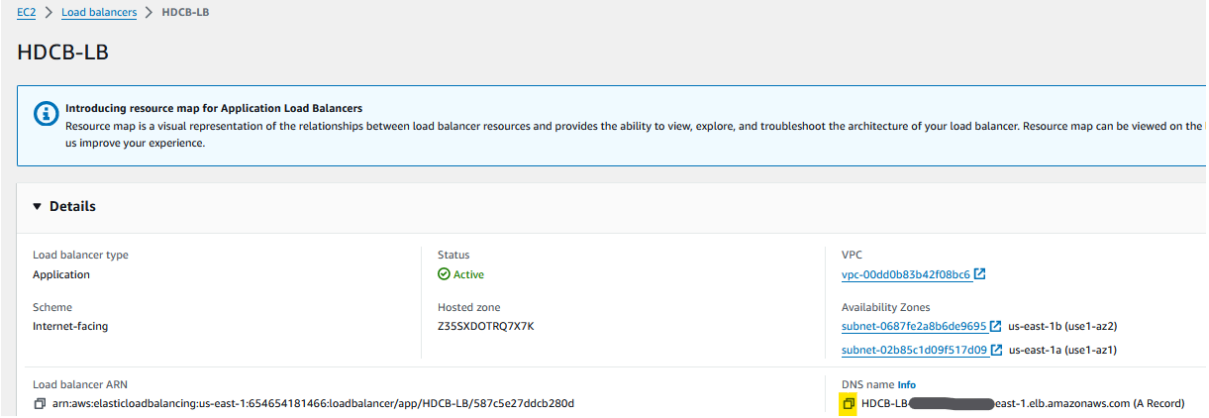
Finally, we have to create a **target group**. Target groups define where the load balancer directs traffic after receiving it. They play a crucial role in distributing traffic across multiple targets and ensuring the availability and scalability of applications hosted behind the load balancer.

In this case our target is the EC2 instance.

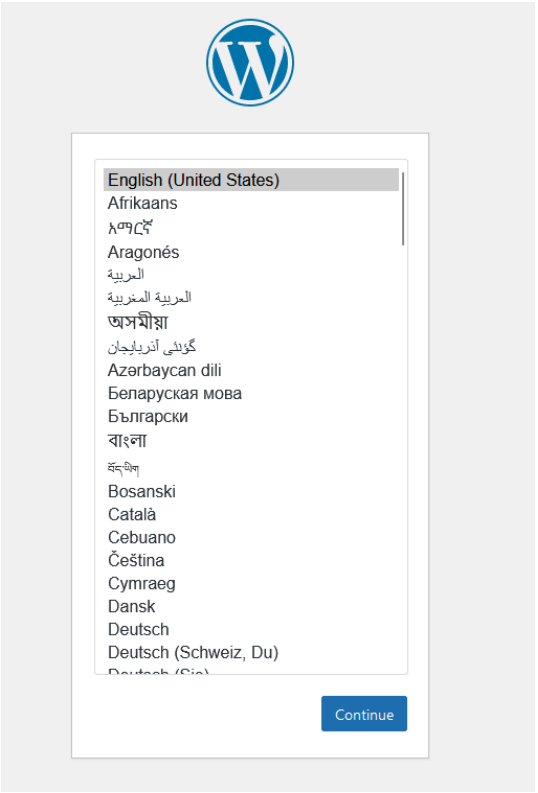


Finally, we deploy the load balancing. Once we have the load balancing working, we can configure wordpress using this elastic IP.

When the load balancer is active, we can copy the DNS and paste it into the browser:

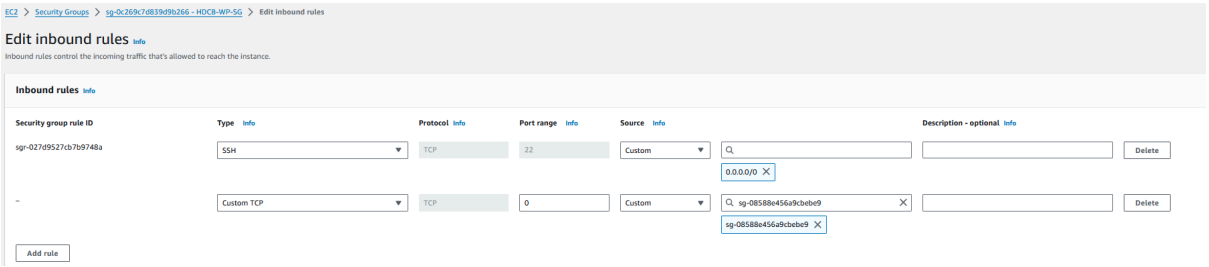


If we do that, we will also be redirected to the wordpress installation page:



We can follow the steps and install it.

Finally, we have to modify the inbound rules for the wordpress security group, only to allow traffic from the load balancer security group:



6. REFERENCES

This exercise was proposed and provided by Jordi Mateo, professor of the subject High Performance and Distributed Computing for Big Data (URV).

https://github.com/JordiMateoUdL