



Tecnológico de monterrey
Campus Querétaro

Construcción de software y toma de decisiones
TC2005B
Grupo 402

“Manual técnico de despliegue”

Deployment Instructions	3
1.- Creating a EC2 instance in aws	3
2.- Setting up the EC2 instance	8
Installing Required Packages	8
Downloading the Repository	9
Running the Application Persistently	10
Using a Proxy	11

Deployment Instructions

For the deployment of the application we used the following technologies, AWS, more specifically an EC2 instance, an Amazon Machine Image (AMI) and AlwaysData, for hosting the project's database, a cloud hosting service that offers both free and paid database hosting plans.

1.- Creating a EC2 instance in aws

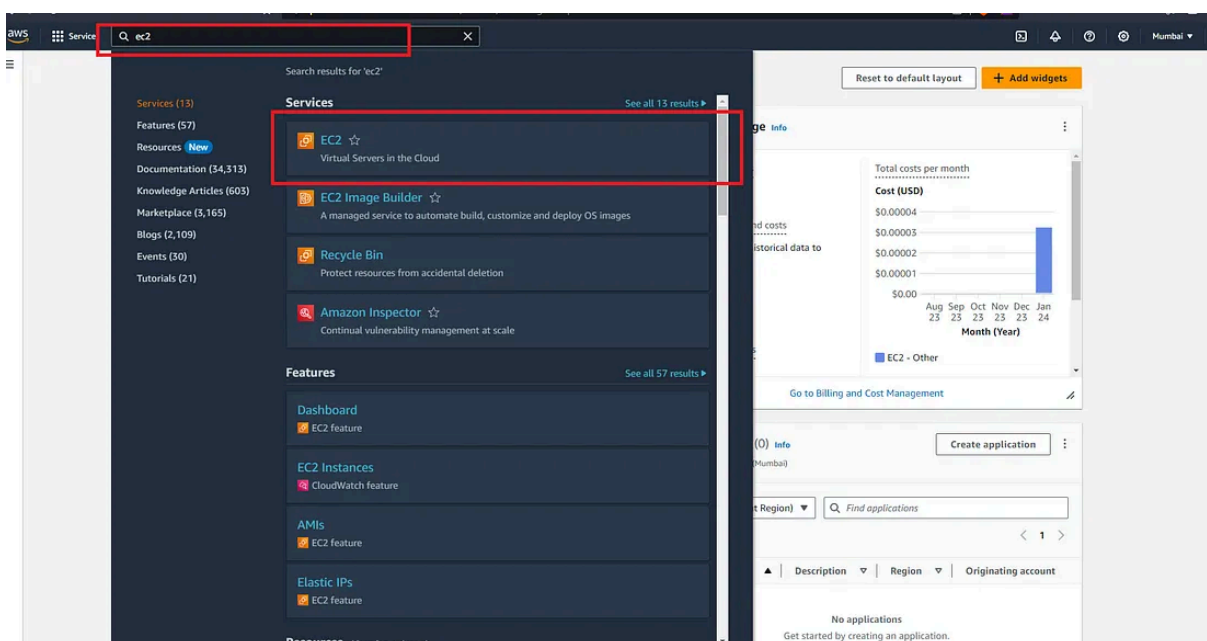
(<https://medium.com/@siddhantshaha28/step-by-step-guide-to-deploy-a-node-js-server-on-aws-ec2-72fd48f89cbd>)

Step 1: Sign in to AWS Console

Create a free account and Login to your AWS Management Console.

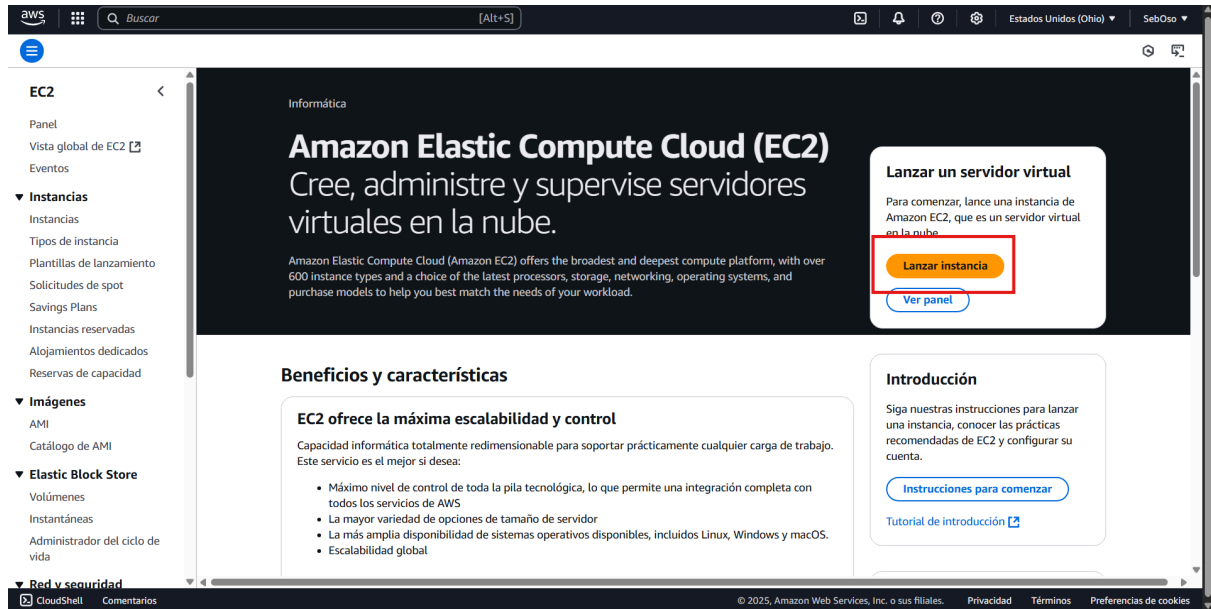
Step 2: Navigate to EC2 Dashboard

Once you login, you will be able to access the AWS console. Search for EC2 service in the search bar and select EC2 option.



Step 3: Launch Instance

Find and click on the **Launch Instance** option on the dashboard or inside the instances page.



Step 4: Choose an Amazon Machine Image (AMI)

Select an image for your instance. An image is a template for your instance with information like the operating system and required software for your virtual machine. For this example, choose a free-tier eligible Amazon Linux AMI or ubuntu.

EC2 > Instances > Launch an instance

Launch an instance [Info](#)

Amazon EC2 allows you to create virtual machines, or instances, that run on the AWS Cloud. Quickly get started by following the simple steps below.

Name and tags [Info](#)

Name


 [Add additional tags](#)

▼ Application and OS Images (Amazon Machine Image) [Info](#)


An AMI is a template that contains the software configuration (operating system, application server, and applications) required to launch your instance. Search or Browse for AMIs if you don't see what you are looking for below

Recents


Quick Start




Amazon Linux




macOS




Ubuntu




Windows



Red Hat



SUSE Linux



[Browse more AMIs](#)

Including AMIs from AWS, Marketplace and the Community

Amazon Machine Image (AMI)

Amazon Linux 2023 AMI

Free tier eligible

ami-0d3f444bc76de0a79 (64-bit (x86), uefi-preferred) / ami-07b4c3e2518ee4edd (64-bit (Arm), uefi)
Virtualization: hvm ENA enabled: true Root device type: ebs

Step 5: Choose an Instance Type

Select an instance type. The default, t2.micro, is eligible for the AWS Free Tier.

Step 6: Key Pair

It allows you to create Key pair (a private and a public key) for logging in to your virtual machine. If you have already created any Key pair select it from the dropdown. Or else, create a new Key pair.

Download the key pair file and store it as it can be used to log in to your instance.

Create key pair

Key pair name

Key pairs allow you to connect to your instance securely.

mykey

The name can include up to 255 ASCII characters. It can't include leading or trailing spaces.

Key pair type

☒ RSA

RSA encrypted private and public key pair

☐ ED25519

ED25519 encrypted private and public key pair



Private key file format

☒ .pem

For use with OpenSSH

☐ .ppk

For use with PuTTY

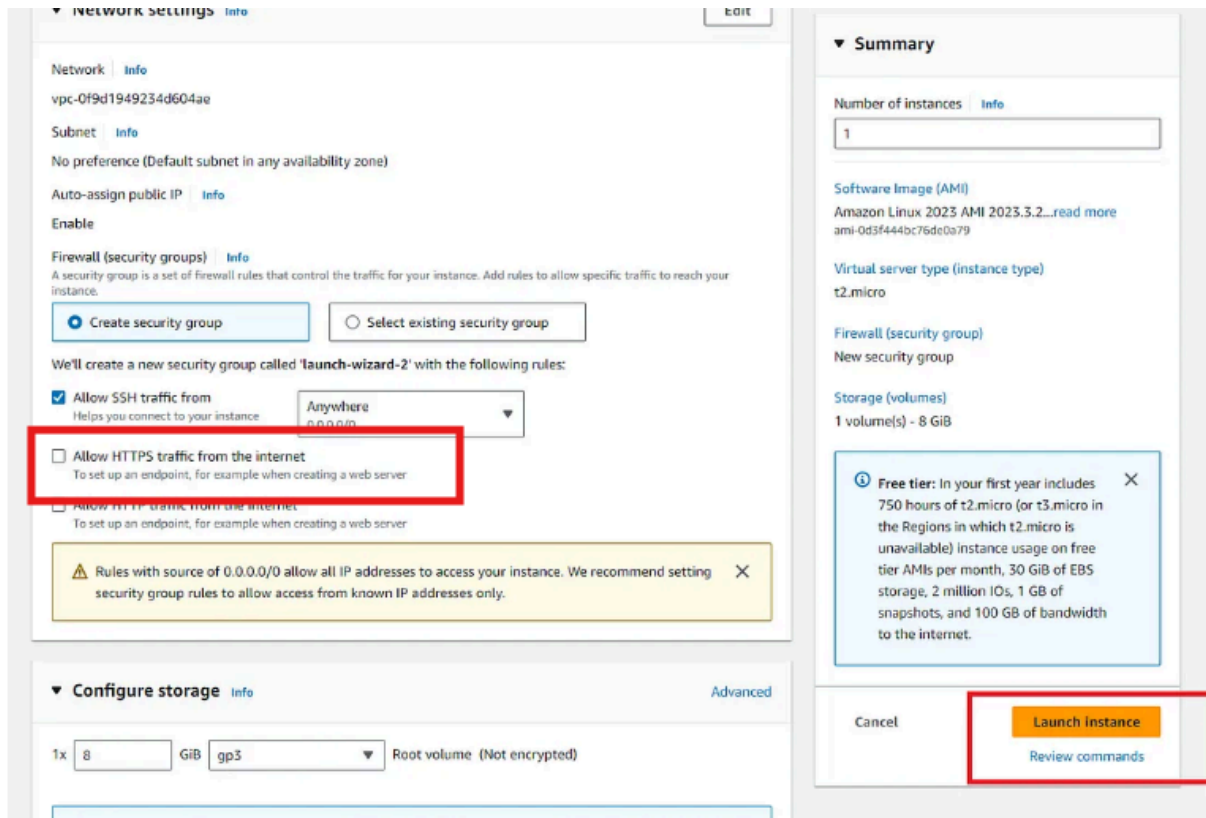
 When prompted, store the private key in a secure and accessible location on your computer. **You will need it later to connect to your instance.** [Learn more](#) 

Cancel

Create key pair

Step 7: Review and Launch

Review the instance and check the box where it says "Allow HTTPS traffic from the internet", to have https connection in our application and launch the instance.



2.- Setting up the EC2 instance

Once the server is created, access its console either via **SSH** or using the web console in aws.

Installing Required Packages

Before continuing, make sure to install the necessary packages to run the application. You can do this with the following command (or its equivalent for your distribution):

```
sudo apt update # to update the package manager
```

```
sudo apt install git nodejs npm mariadb-server nginx ufw -y
```

Downloading the Repository

It's not mandatory, but it's recommended to create a folder to organize your project on your server.


```
mkdir 'projects'  
cd 'projects'
```

Inside the folder, we clone our repository to have it locally:

```
git clone https://github.com/ian-hdzzz/nuclea.git
```

then move on to our project folder

```
cd 'nuclea'
```

Then, we will setup the `.env` file with the following command

You can use the following commands:

- `touch .env`

After creating the `.env`, you will have to modify this file with nano, and paste this code

- `nano .env`

PORT=4002

DB_HOST=mysql-flowitbd.alwaysdata.net

DB_USER=flowitbd_general

DB_PASSWORD=lalo123

DB_NAME=flowitbd_basededatos

DB_PORT=3306

SECRET=ian

GOOGLE_CLIENT_ID=474608711367-1jvug8coq4d5im2qr4p1sukshjfc4cju.apps.googleusercontent.com

GOOGLE_CLIENT_SECRET=GOCSPX-EQKvgxwo0mp1KAypEAPPHBntTBZ

GOOGLE_API_KEY=AlzaSyDS6YpnsLgZHFUmL37oNME9lwZXs0P5EjI

EMAIL_USER="flowitdb@gmail.com"

EMAIL_APP_PASSWORD="epro aziw pnx d awhl"

WHATSAPP_TOKEN=EAAIBZB3s9pDQBOyjhZCEaZAIzAEoaYZAahGpp3w3GcyN9S2Z
CgIVtrLKY7Kh5uv0ZAI3Oapxb6GYjnkOOwA0I1TiWeWUpzka5gwYZBjw1XAQdJNLTLF
oS22zxxU5KdjsjWgpykYko6quF7NjD30YoaupJUxJqe4dOQu44EZCdHjAqWcwOGx8Q
1QiPaA70ujHTvZBICeTZC7plkJB0Ao8RanXQmt4EGjblb3SoGs4Hcr7eSZAfGgZD

WHATSAPP_PHONE_NUMBER_ID=557558657437715

WHATSAPP_BUSINESS_ACCOUNT_ID=54762320175998

WHATSAPP_VERIFY_TOKEN=nuclea_whatsapp_verify_token

Once all the necessary files are in place, install the dependencies and run the project to verify everything works:

```
npm install
```

```
npm run
```

Running the Application Persistently

At this point, you should be able to access your project via its port. If not, try:

```
sudo ufw allow 4002/tcp # Or the port you are using
```

To keep the application running persistently, install PM2 inside your project folder:

```
npm install pm2
```

Then run it with:

```
npx pm2 start src/index.js
```

To stop the project:

```
npx pm2 delete 0
```

Using a Proxy

HTTPS Connection with a Self-Signed Certificate

First, install the necessary packages:

```
sudo apt install openssl
```

Then generate a private key:

```
sudo openssl genpkey -algorithm RSA -out /etc/ssl/private/server.key
```

Generate a certificate signing request:

```
sudo openssl req -new -key /etc/ssl/private/server.key -out /etc/ssl/certs/server.csr
```

After that command you will have to fill the form fields required:

Country Name:

State or Province Name(full name):

Locality Name

Finally, create the self-signed certificate (valid for one year):

Organization Name:

COMMON NAME (YOUR name):

the other ones you can leave them blank

After that run this code

```
sudo openssl x509 -req -in /etc/ssl/certs/server.csr -signkey /etc/ssl/private/server.key -out /etc/ssl/certs/server.crt -days 365
```

Update the Nginx configuration:

```
sudo nano /etc/nginx/sites-available/default
```

Comment all the lines after the editor opened up

And at the end of the code in the editor add this content:

```
server {  
  
    listen 80;  
  
    server_name _;  
  
  
    return 301 https://$host$request_uri;
```

```
}
```

```
server {
```

```
    listen 443 ssl;
```

```
    server_name _;
```

```
    ssl_certificate /etc/ssl/certs/server.crt;
```

```
    ssl_certificate_key /etc/ssl/private/server.key;
```

```
    location / {
```

```
        proxy_pass http://localhost:4002;
```

```
        proxy_set_header Host $host;
```

```
        proxy_set_header X-Real-IP $remote_addr;
```

```
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
```

```
        proxy_set_header X-Forwarded-Proto $scheme;
```

```
    }
```

```
}
```

Test and reload Nginx:

```
sudo nginx -t
```

```
sudo systemctl restart nginx
```

If everything worked out you should be able to go to the IP of your EC2 instance and see the application running.