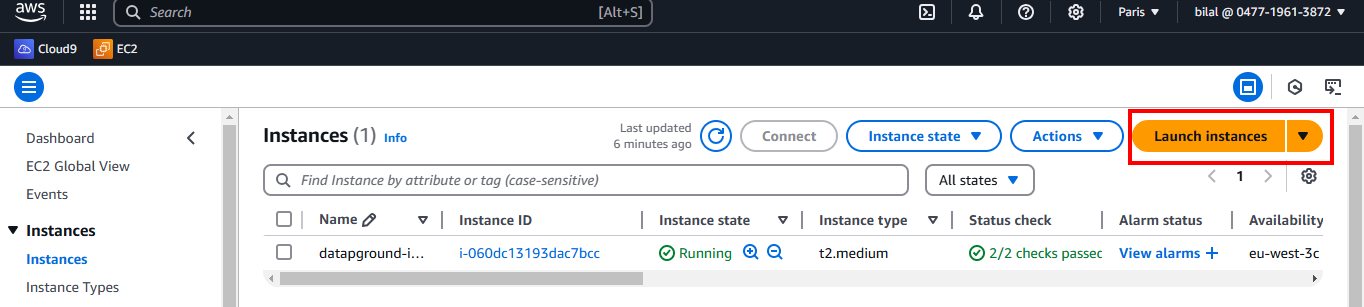
Documentation

### Create docker image for the Generator

* Create a docker image and Upload it to ECR:

**1.** You can work on an EC2 instance type t2.micro that is in a free tier in AWS. Or you can work on a Linux system.

Create the EC2 with the following configuration:



**Amazon Machine Image (AMI): Amazon Linux 2023 AMI**

**Instance type: t2.micro**

**Key pair (login): datapground-instance**

**Firewall (security groups): Select existing security group**

**Common security groups: datapground-instance-SG**

**(Leave the other configurations with default)**

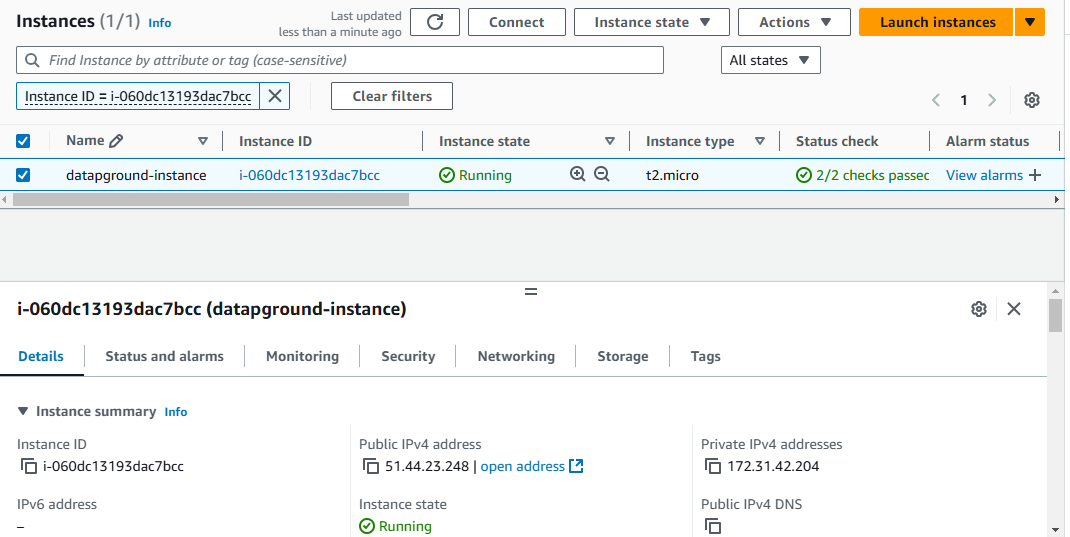
Then create the instance.

**2.** Connect to the instance with SSH using the key pair you have chosen “**datapground-instance”** and that you have downloaded earlier. This keypair is they key that is also working with the instance “datapground-instance”.

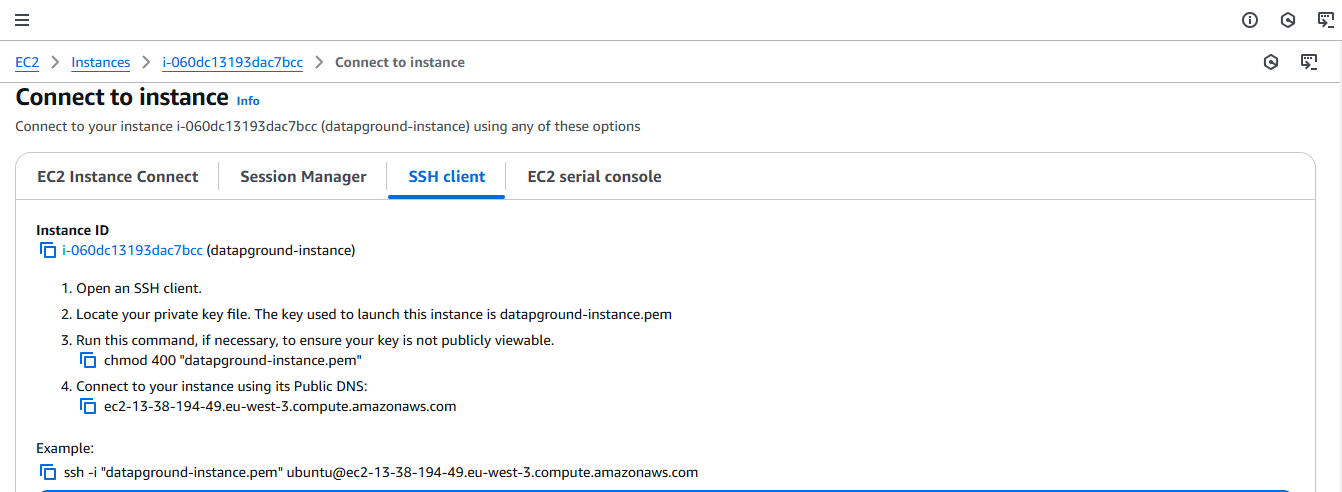
- Browse the directory where the key pair is and open terminal

- In your terminal type the SSH command, you can get the command for the instance you created from here: (see example below with the exitent instance)

Select the instance and click “Connect” then go to SSH tab:

****







Copy and paste this command in the terminal.

3. After opening the instance with SSH:

- Install docker with these commands:

Update the System

sudo yum update -y

Install Docker:

sudo dnf install docker -y

Start and Enable Docker:

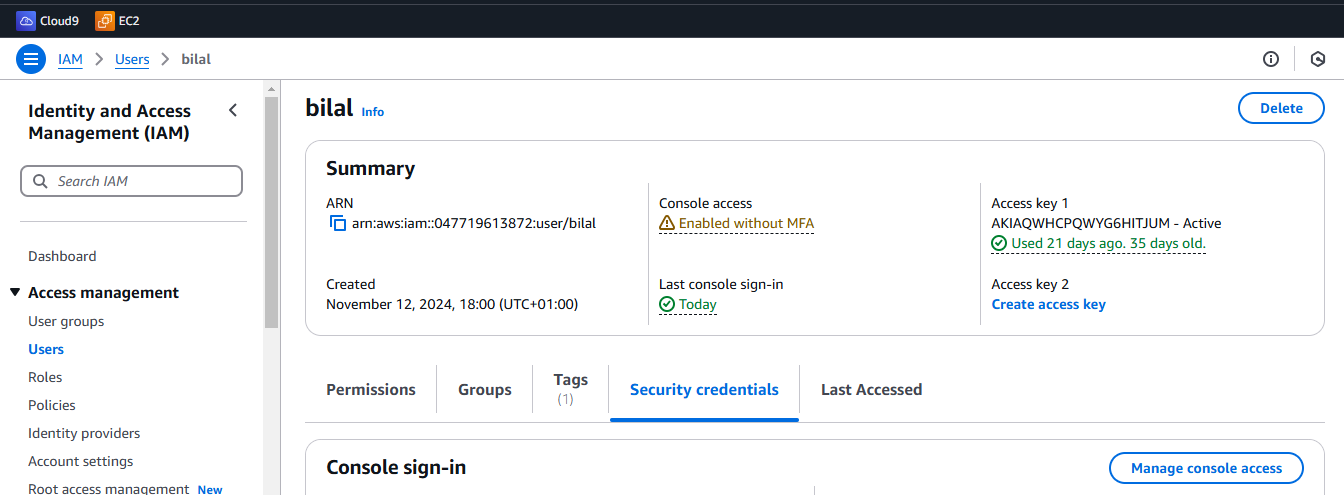
sudo systemctl start docker

sudo systemctl enable docker

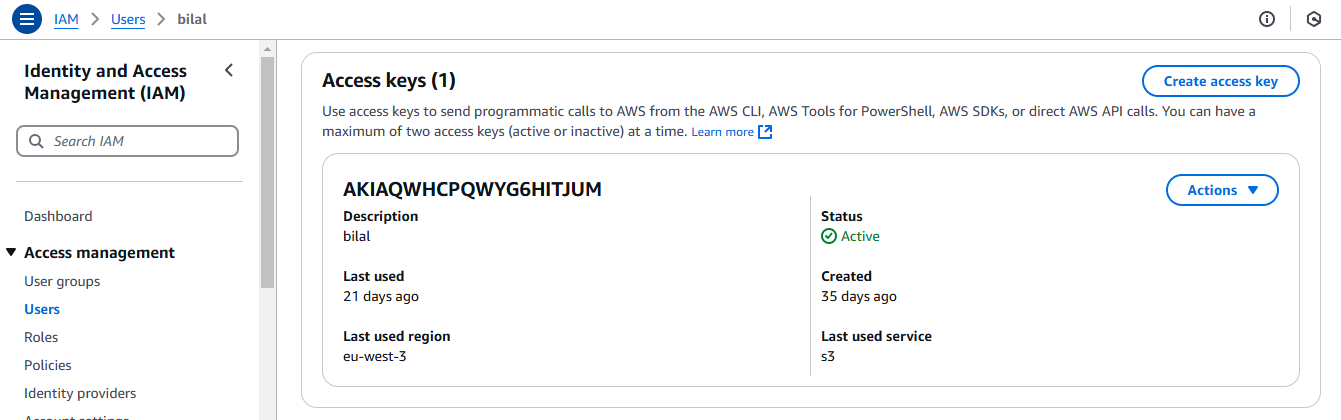
Add Your User to the Docker Group

sudo usermod -aG docker $USER

4. Get the access key and connect to AWS CLI:

- Get the access key, Go to IAM dashboard, users and click on your username, then go to tab Security credentials:  


- Go to Access keys and create access key, and download it:



- Install AWS CLI in the instance:

sudo dnf install aws-cli -y

- Run this command: aws configure

And enter the details you downloaded, below an example:

AWS Access Key ID [None]: AKIAEXAMPLE123   
AWS Secret Access Key [None]: wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY   
Default region name [None]: eu-west-3   
Default output format [None]: (you can leave it empty)

Connecting with CLI will allows you to push the docker image to the ECR

- Prepare the files needed to build the docker image all in a one folder:

I used an EC2 instance to create the docker image

- Create a directory and give it a name with command:

mkdir generator

- Create the files one by one and past the code inside using command:

nano Dockerfile

nano lambda\_function.py

nano requirements.txt

nano generators.py

google\_trans.py

- Dockerfile : file that build the docker image

- lambda\_function.py : code of lambda function

- requirements.txt : file of the required dependencies

- generators.py : file of the functions of the generator

- google\_trans.py

- Browse to the folder and build the docker image with command:

docker build -t lambda-generator .

- After building finished, tag the image for the ECR:

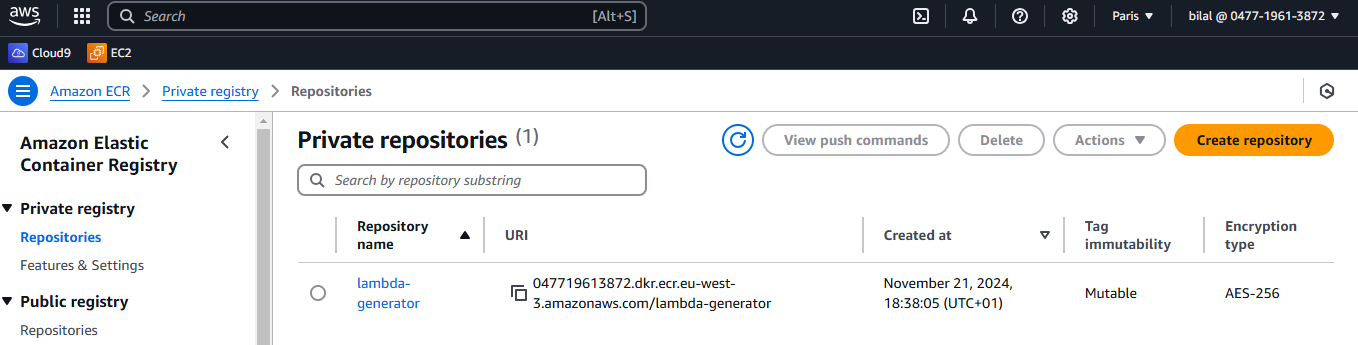
docker tag lambda-generator 047719613872.dkr.ecr.eu-west-3.amazonaws.com/lambda-generator

- Push the Image to ECR:

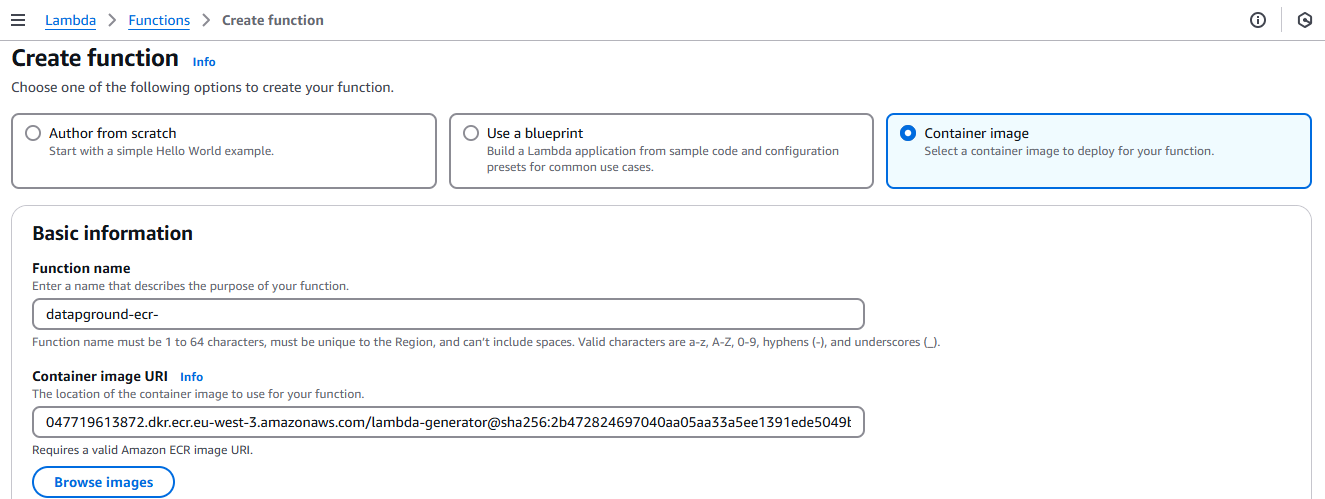
aws ecr get-login-password --region eu-west-3 | docker login --username AWS --password-stdin 047719613872.dkr.ecr.eu-west-3.amazonaws.com

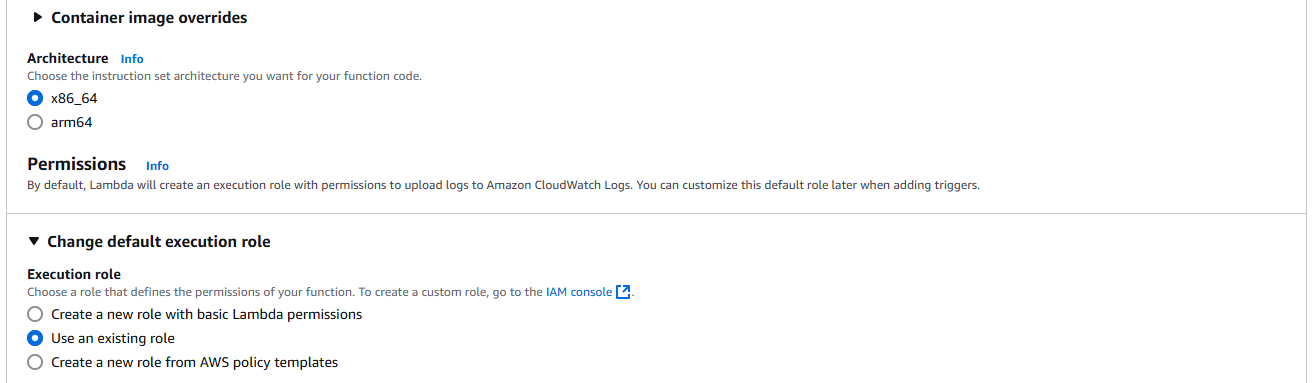
docker push 047719613872.dkr.ecr.eu-west-3.amazonaws.com/lambda-generator

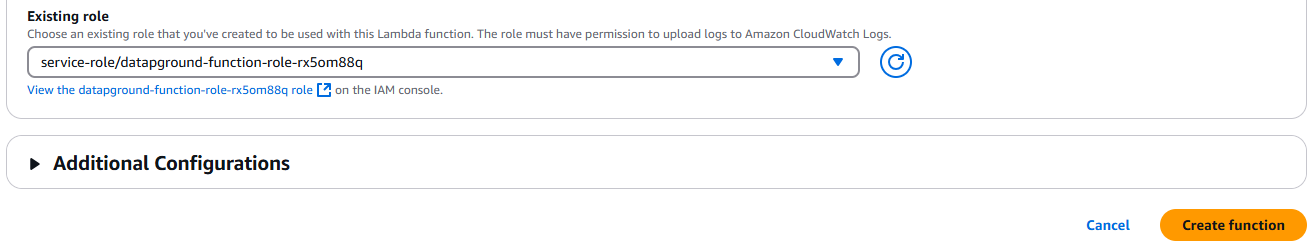
- Check the image in the ECR:



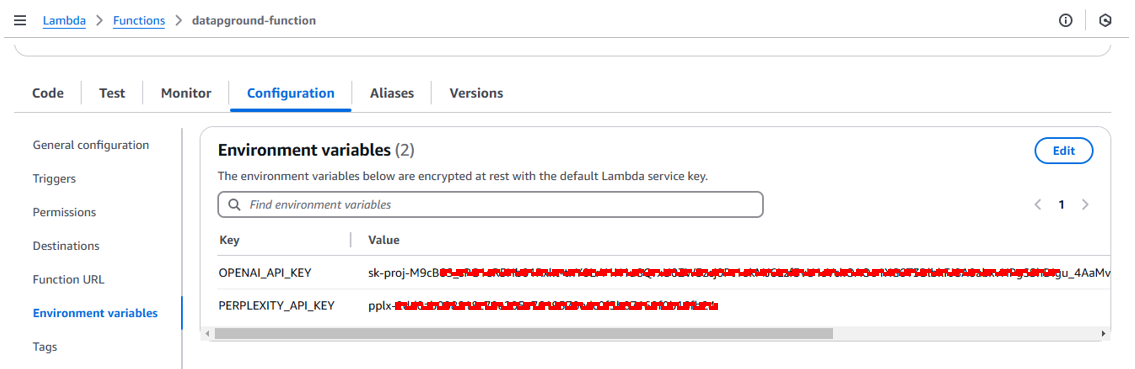
* Create lambda function:





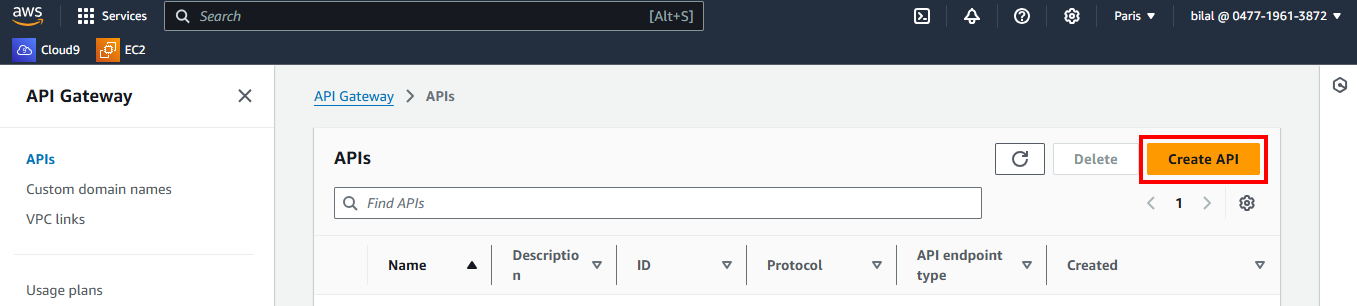


- Add API keys to the environment variables of the lambda function created:

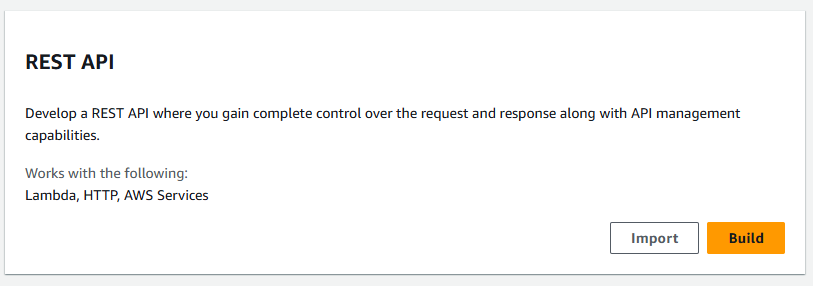


* Create API Gateway:

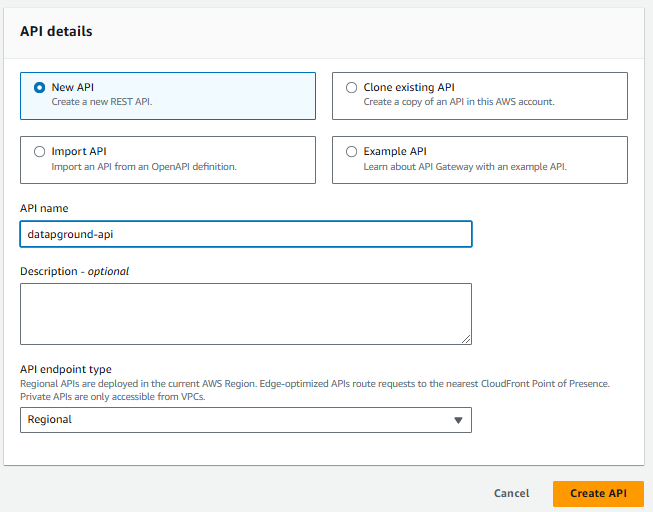
- Go the API Gateway page and click “Create API”:



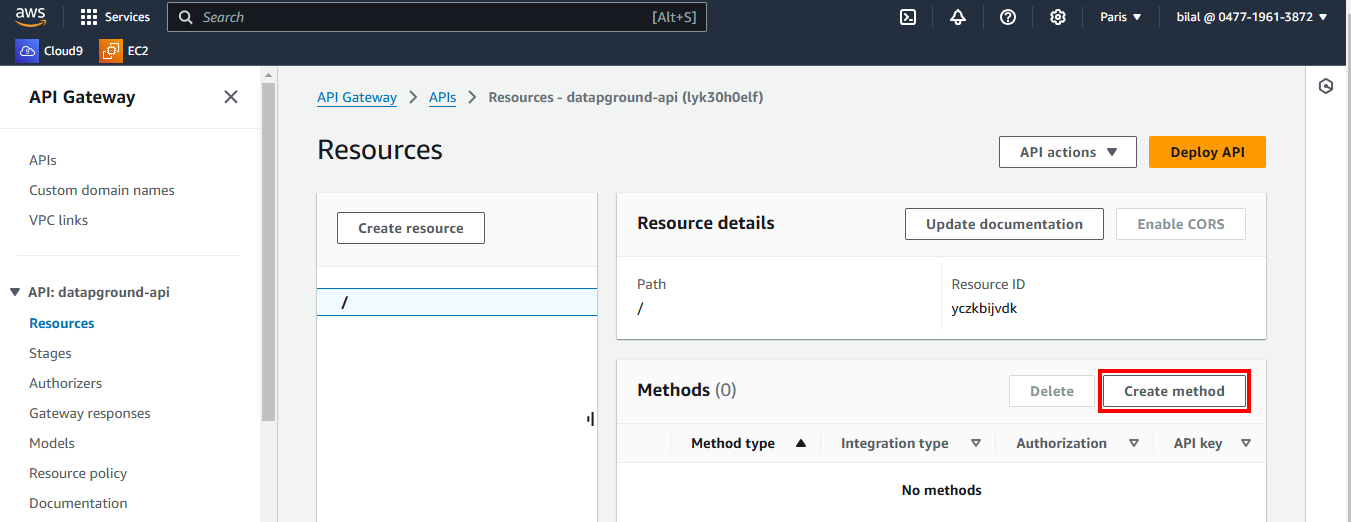
- Choose REST API:



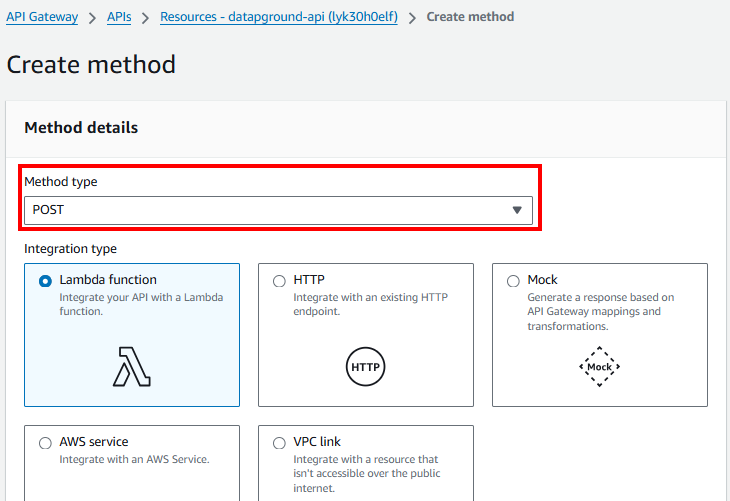
- Put a name, “[datapground-api](https://eu-west-3.console.aws.amazon.com/apigateway/main/apis/lyk30h0elf/resources?api=lyk30h0elf&experience=rest&region=eu-west-3)” and Create API:

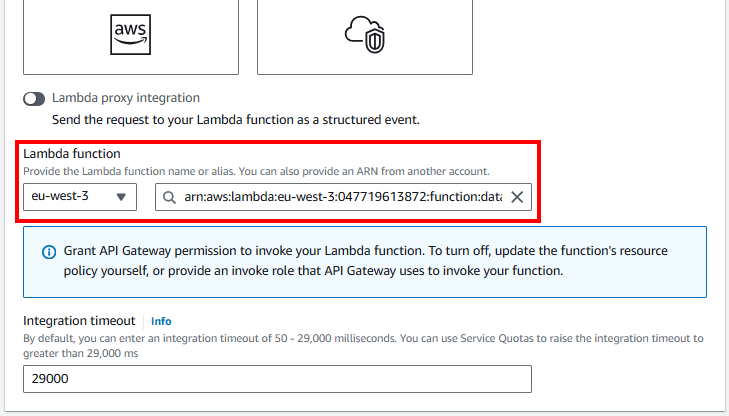


- Create Method:

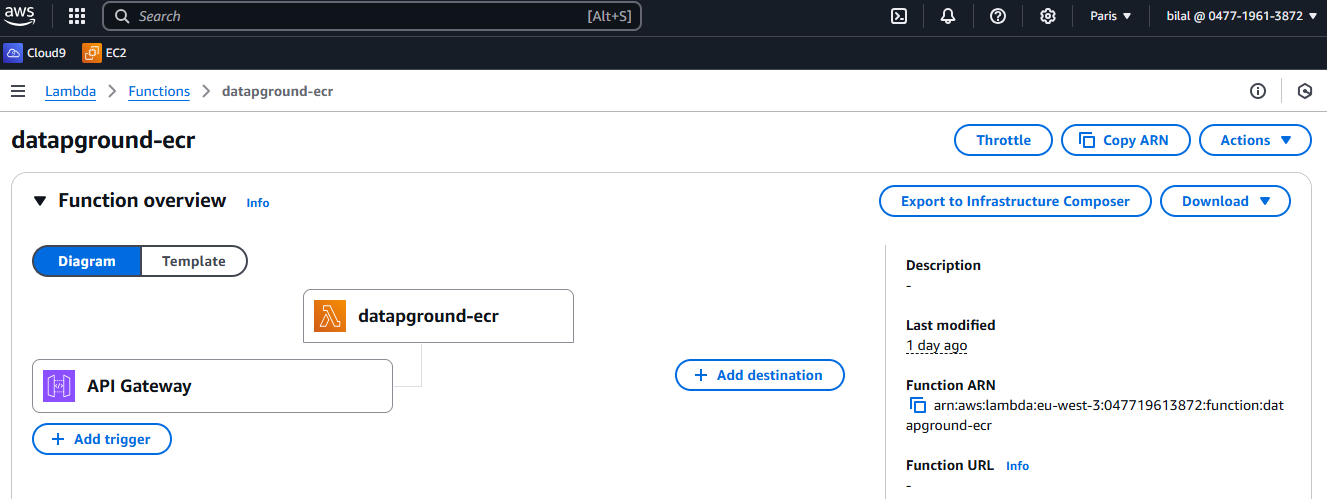


- Choose Method type “POST” and the lambda function created:

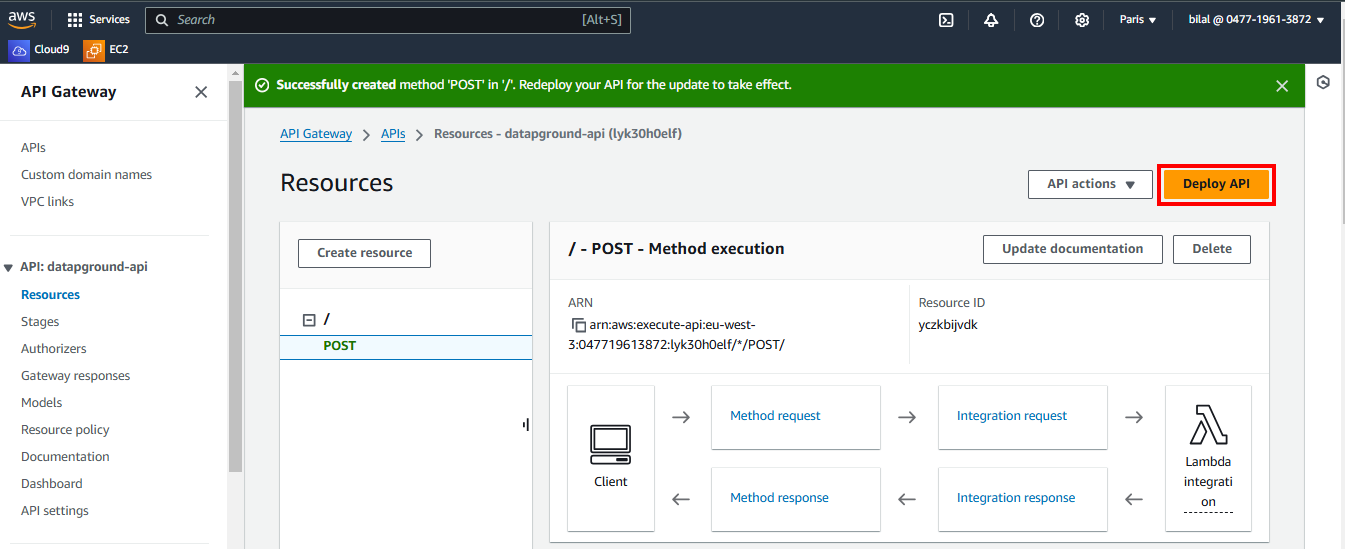




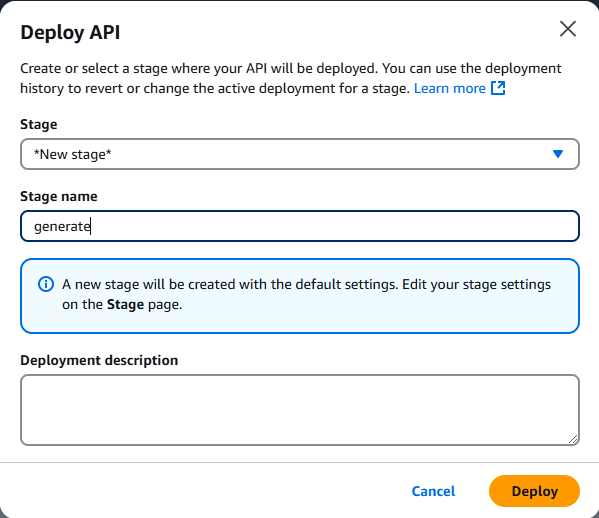
- If you check the lambda function you will find API Gateway as a trigger:



- Method POST created, Click “Deploy API”:

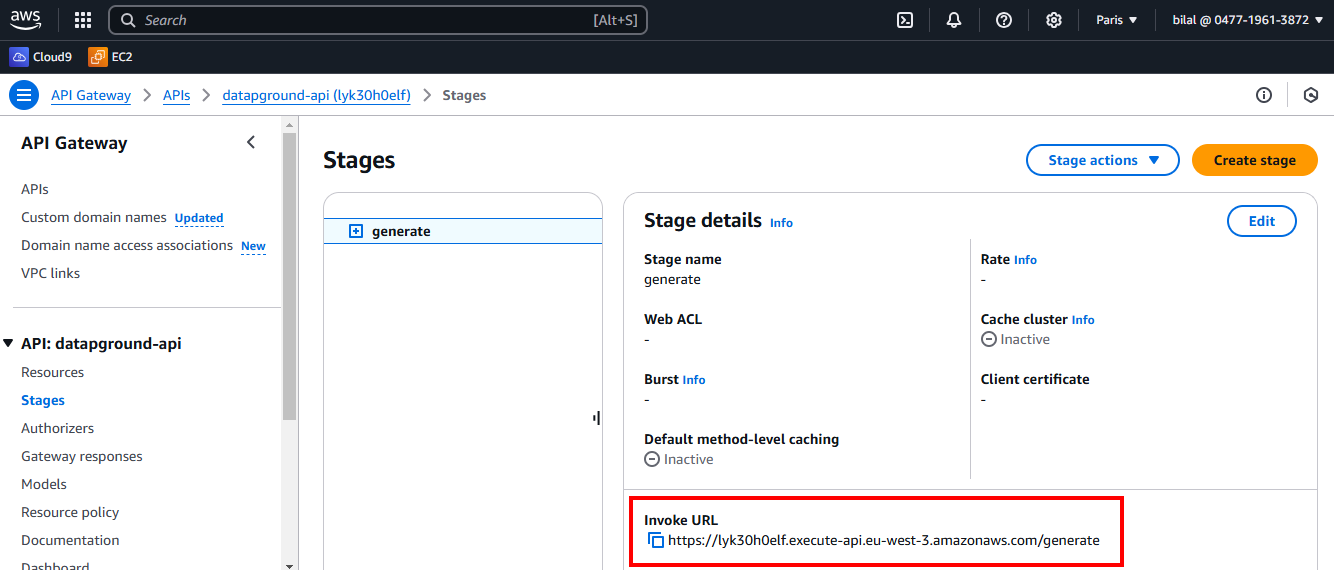


- Choose “New stage” and put a name:



- The deployment created, the invoke URL is the API endpoint:

<https://lyk30h0elf.execute-api.eu-west-3.amazonaws.com/generate>



- The Json code to use for testing with Postman and also to use it to send requests to the API:  
  
\* API endpoint:

https://lyk30h0elf.execute-api.eu-west-3.amazonaws.com/generate

\* Json code:

- THE MERLIN GENERATOR:

{

"body": "{\"generator\_type\": \"merlin\", \"theme\": \"Generate a dataset about AWS services with the fields: name, time, price.\", \"rows\": 100}"

}

- THE GOLD GENERATOR:

{

"body": "{\"generator\_type\": \"gold\", \"theme\": \"Generate the fields department, hire\_date and salary simulating this is a tech company. Make them so that there is a relation between the three. The previous hire\_dates correlate with better salaries mostly. And also the department is related with the range within the salary moves. I want a variety of 5 departments.\", \"rows\": 200}"

}

- THE PREMIUM GENERATOR:

{

"body": "{\"generator\_type\": \"premium\", \"theme\": \"Generate football data. Name, position, date of birth, height, team he plays for, scored goals, and some description about its game. Add some other fields you find relevant.\", \"rows\": 50}"

}

- THE ORACLE GENERATOR:

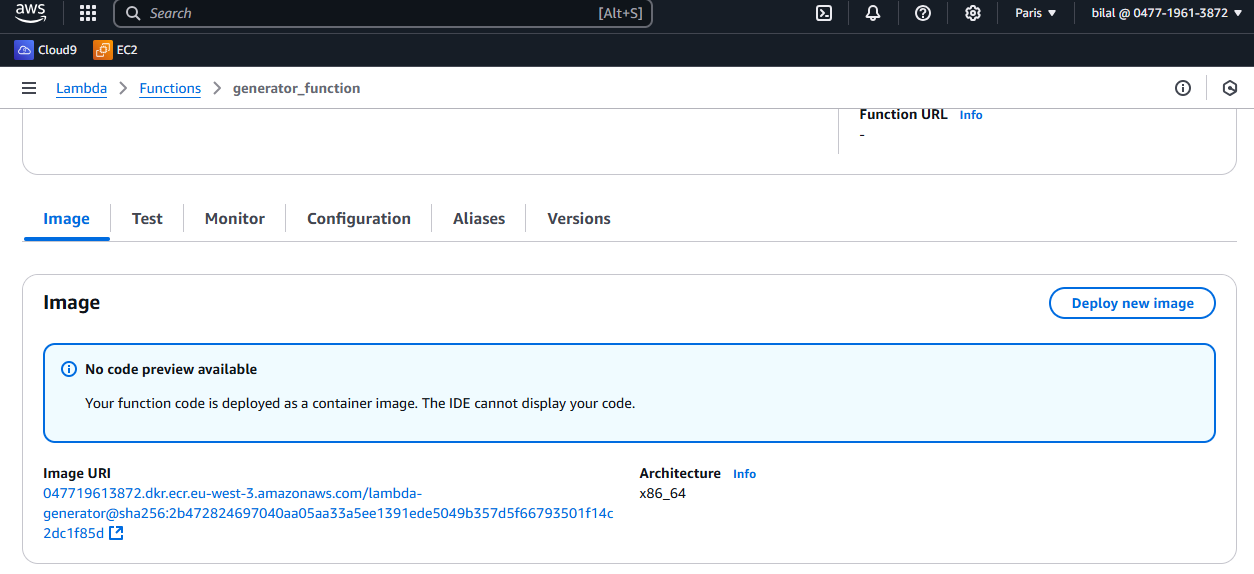
{

"body": "{\"generator\_type\": \"oracle\", \"theme\": \"Recent news that happend in the world of sports. Title, brief description, and date. Be specific about each new in the title.\", \"rows\": 15}"

}

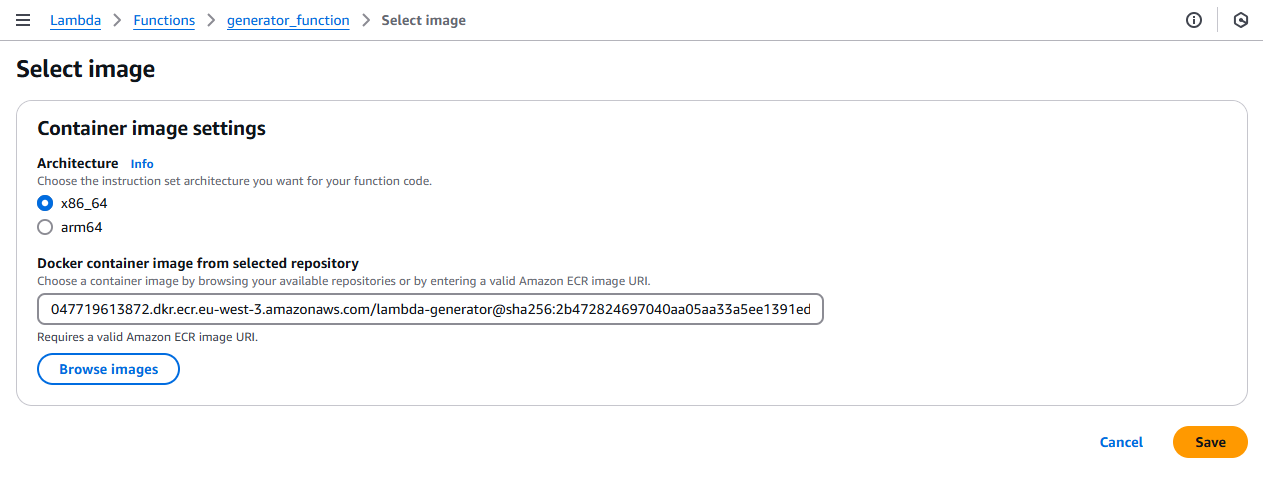
**\*\* To work with the same lambda function and you want just to change the image for the existent lambda function here is how to change it:**

- Go to lambda function and click “deploy new image”





- Change the image:





- Select the image:

