CC de data pipeline :

Question 1)
La région Stockholm sur AWS -> eu-north-1

Terraaform plan :

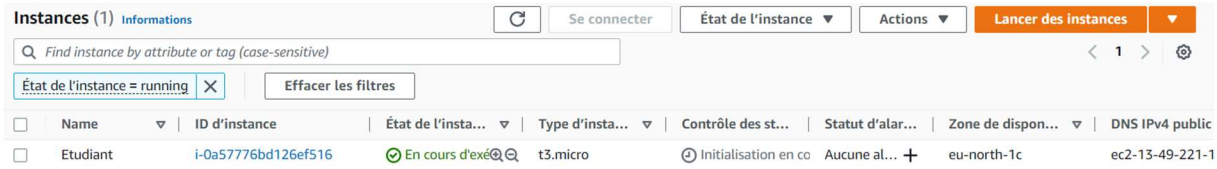
```
$ terraform plan

Terraform used the selected providers to generate the following executio
plan. Resource actions are indicated with the following symbols:
  + create

Terraform will perform the following actions:

  # aws_instance.ec2_vm will be created
  + resource "aws_instance" "ec2_vm" {
      + ami                                  = "ami-02384a901b5df8024"
      + arn                                  = (known after apply)
      + associate_public_ip_address          = (known after apply)
      + availability_zone                    = (known after apply)
      + cpu_core_count                       = (known after apply)
      + cpu_threads_per_core                 = (known after apply)
      + disable_api_termination              = (known after apply)
      + ebs_optimized                        = (known after apply)
      + get_password_data                    = false
      + host_id                              = (known after apply)
      + id                                   = (known after apply)
      + instance_initiated_shutdown_behavior = (known after apply)
      + instance_state                       = (known after apply)
      + instance_type                        = "t3.micro"
      + ipv6_address_count                   = (known after apply)
      + ipv6_addresses                       = (known after apply)
      + key_name                             = "prisca-clio-key"
      + monitoring                           = (known after apply)
      + outpost_arn                          = (known after apply)
      + password_data                        = (known after apply)
      + placement_group                      = (known after apply)
      + placement_partition_number           = (known after apply)
      + primary_network_interface_id         = (known after apply)
      + private_dns                          = (known after apply)
      + private_ip                           = (known after apply)
      + public_dns                           = (known after apply)
      + public_ip                            = (known after apply)
      + secondary_private_ips                = (known after apply)
      + security_groups                      = (known after apply)
      + source_dest_check                    = true
      + subnet_id                            = (known after apply)
      + tags                                 = {
          + "Name"  = "Etudiant"
          + "Owner" = "prisca.clio@etu.u-pec.fr"
        }
      + tags_all                             = {
          + "Name"  = "Etudiant"
          + "Owner" = "prisca.clio@etu.u-pec.fr"
        }
      + tenancy                              = (known after apply)
      + user_data                            = (known after apply)
      + user_data_base64                     = (known after apply)
      + vpc_security_group_ids               = (known after apply)

      + capacity_reservation_specification {
```

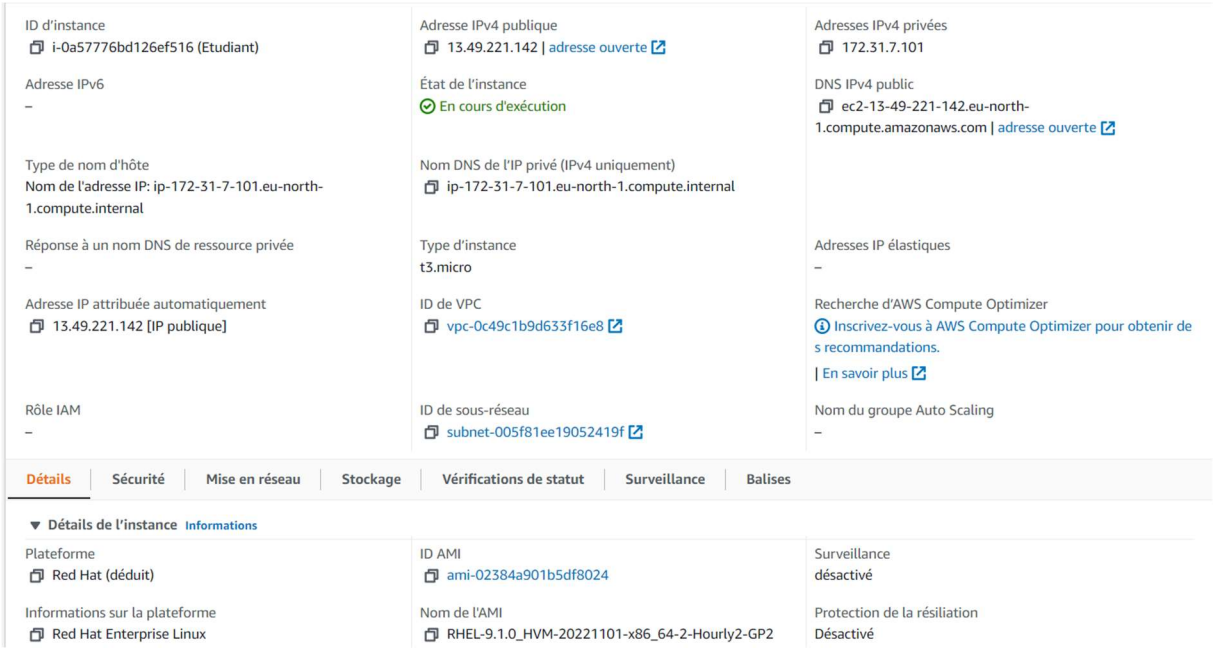
```
+ capacity_reservation_specification {
    + capacity_reservation_preference = (known after apply)

    + capacity_reservation_target {
        + capacity_reservation_id = (known after apply)
      }
  }

+ ebs_block_device {
    + delete_on_termination = (known after apply)
    + device_name           = (known after apply)
    + encrypted             = (known after apply)
    + iops                  = (known after apply)
    + kms_key_id            = (known after apply)
    + snapshot_id           = (known after apply)
    + tags                  = (known after apply)
    + throughput            = (known after apply)
    + volume_id             = (known after apply)
    + volume_size           = (known after apply)
    + volume_type           = (known after apply)
  }

+ enclave_options {
    + enabled = (known after apply)
  }

+ ephemeral_block_device {
    + device_name  = (known after apply)
    + no_device    = (known after apply)
    + virtual_name = (known after apply)
  }

+ metadata_options {
    + http_endpoint               = (known after apply)
    + http_put_response_hop_limit = (known after apply)
    + http_tokens                 = (known after apply)
    + instance_metadata_tags      = (known after apply)
  }

+ network_interface {
    + delete_on_termination = (known after apply)
    + device_index          = (known after apply)
    + network_interface_id  = (known after apply)
  }

+ root_block_device {
    + delete_on_termination = (known after apply)
    + device_name           = (known after apply)
    + encrypted             = (known after apply)
    + iops                  = (known after apply)
    + kms_key_id            = (known after apply)
    + tags                  = (known after apply)
    + throughput            = (known after apply)
    + volume_id             = (known after apply)
    + volume_size           = (known after apply)
    + volume_type           = (known after apply)
```

```
          + kms_key_id            = (known after apply)
          + tags                  = (known after apply)
          + throughput            = (known after apply)
          + volume_id             = (known after apply)
          + volume_size           = (known after apply)
          + volume_type           = (known after apply)
        }
    }

  # aws_key_pair.mykey will be created
  + resource "aws_key_pair" "mykey" {
      + arn             = (known after apply)
      + fingerprint     = (known after apply)
      + id              = (known after apply)
      + key_name        = "prisca-clio-key"
      + key_name_prefix = (known after apply)
      + key_pair_id     = (known after apply)
      + public_key      = "ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQDp1iJ685uh0wg6Z
CJ3+1qWXhNScb/CjMdxXcZ8yWbnJTUmI90lfC3E7dsBN8Rb+CBASKVk29DX6O1U6v+lAUSiDOaGoB7S8
pDdYnHRSWw+OdcCq72HV5Iy2pmqKlwOt8xuFDNxAHlPh/1txHGQF1LFCCjZf6cmCm7sxFYea51K6bV7a
WA4qHQDkOSAN9DtoHMTTeUKPOtTJGx+QgUfXl8KTPXg2NBWr2xsQu7WrYv3W5EACpPYXQyZxhxDy/VQp
qtNQ+kzeYO1Qu7tqhbNdyer9m5LWIlwwHDiUH9n6sswtqjXHtWjXMYLRTLg7H/XGs7xhX8DCQYjJze3M
Ssf/86T Prisca@LAPTOP-9KJ7VP6F"
      + tags_all        = (known after apply)
    }

  # aws_security_group.default will be created
  + resource "aws_security_group" "default" {
      + arn                    = (known after apply)
      + description            = "Managed by Terraform"
      + egress                 = [
          + {
              + cidr_blocks      = [
                  + "0.0.0.0/0",
                ]
              + description      = ""
              + from_port        = 0
              + ipv6_cidr_blocks = []
              + prefix_list_ids  = []
              + protocol         = "-1"
              + security_groups  = []
              + self             = false
              + to_port          = 0
            },
        ]
      + id                     = (known after apply)
      + ingress                = [
          + {
              + cidr_blocks      = [
                  + "0.0.0.0/0",
                ]
              + description      = ""
              + from_port        = 22
              + ipv6_cidr_blocks = []
              + prefix_list_ids  = []
              + protocol         = "tcp"
              + security_groups  = []
              + self             = false
```

```
            + from_port        = 0
            + ipv6_cidr_blocks = []
            + prefix_list_ids  = []
            + protocol         = "-1"
            + security_groups  = []
            + self             = false
            + to_port          = 0
          },
      ]
    + id                     = (known after apply)
    + ingress                = [
        + {
            + cidr_blocks      = [
                + "0.0.0.0/0",
              ]
            + description      = ""
            + from_port        = 22
            + ipv6_cidr_blocks = []
            + prefix_list_ids  = []
            + protocol         = "tcp"
            + security_groups  = []
            + self             = false
            + to_port          = 22
          },
        + {
            + cidr_blocks      = [
                + "0.0.0.0/0",
              ]
            + description      = ""
            + from_port        = 80
            + ipv6_cidr_blocks = []
            + prefix_list_ids  = []
            + protocol         = "tcp"
            + security_groups  = []
            + self             = false
            + to_port          = 80
          },
      ]
    + name                   = (known after apply)
    + name_prefix            = (known after apply)
    + owner_id               = (known after apply)
    + revoke_rules_on_delete = false
    + tags_all               = (known after apply)
    + vpc_id                 = (known after apply)
  }

Plan: 3 to add, 0 to change, 0 to destroy.

Note: You didn't use the -out option to save this plan, so Terraform can't
guarantee to take exactly these actions if you run "terraform apply" now.
```

Capture d’écran du tableau de bord AWS montrant la VM démarrée et son « tag » :

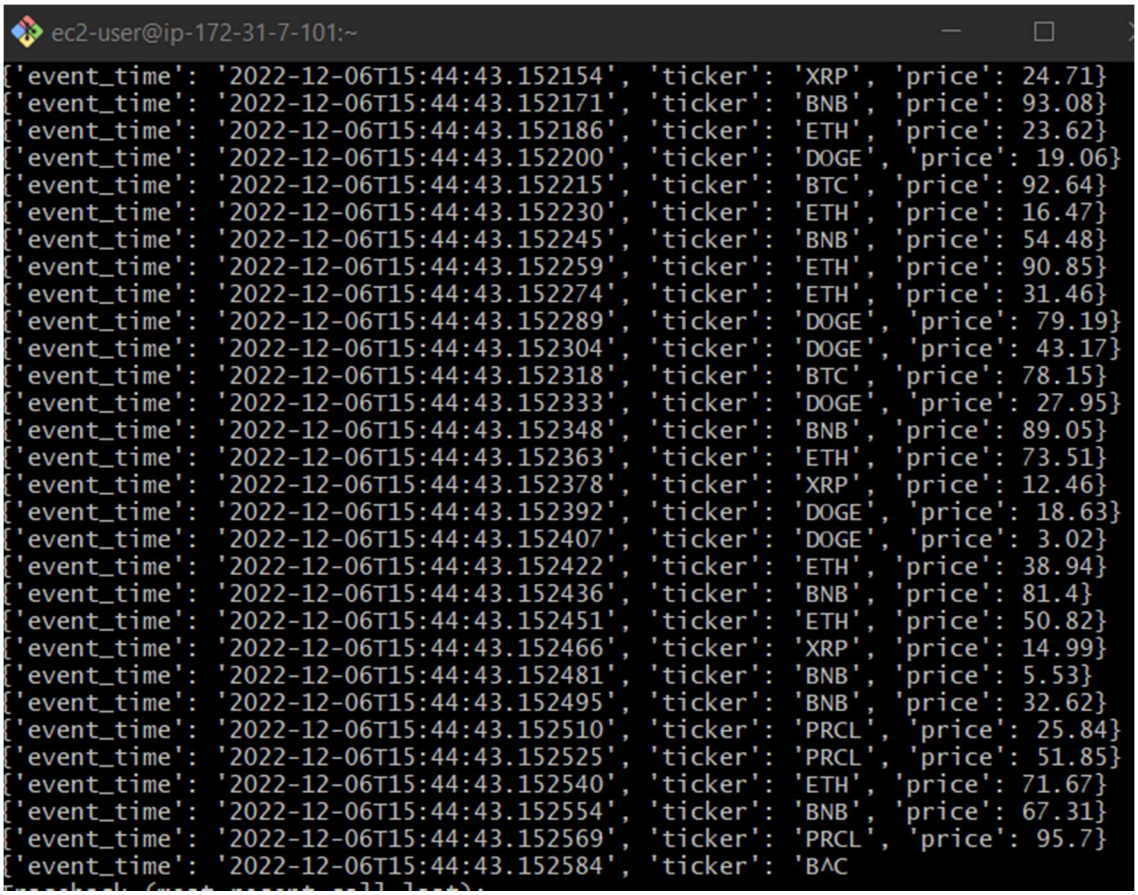




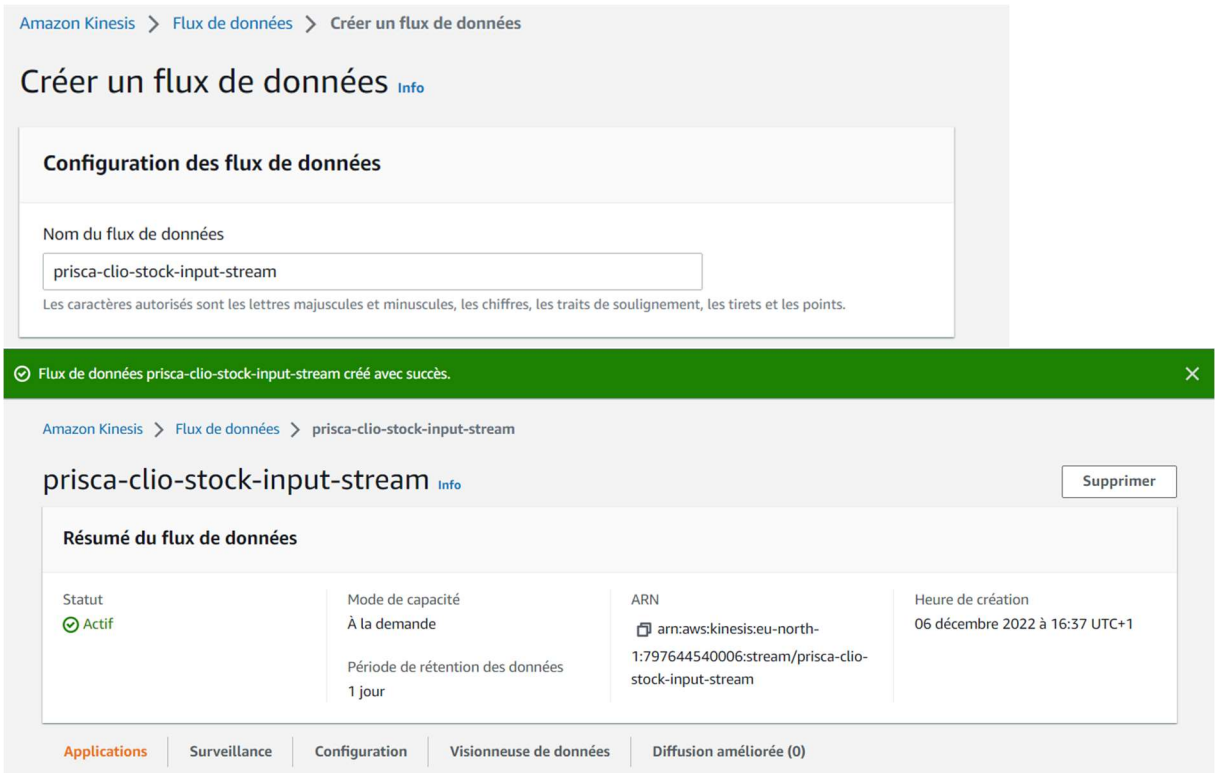
Question 2)
Se connecter :
ssh -i "prisca-clio-key.pem" ec2-user@ec2-13-49-221-142.eu-north-1.compute.amazonaws.com

sudo yum install -y python
sudo yum install -y pip
sudo pip install boto3

```
[ec2-user@ip-172-31-7-101 ~]$ ls
stock.py
[ec2-user@ip-172-31-7-101 ~]$ cat stock.py
import datetime
import json
import random
import boto3

STREAM_NAME = "input-stream"
REGION = "us-east-1"

def get_data():
    return {
        'event_time': datetime.datetime.now().isoformat(),
        'ticker': random.choice(["BTC","ETH","BNB", "XRP", "DOGE","PRCL"]),
        'price': round(random.random() * 100, 2)}

def generate(stream_name, kinesis_client):
    while True:
        data = get_data()
        print(data)
       #kinesis_client.put_record(StreamName=stream_name,Data=json.dumps(data),P
artitionKey="partitionkey")

if __name__ == '__main__':
    generate(STREAM_NAME, boto3.client('kinesis', region_name=REGION))
[ec2-user@ip-172-31-7-101 ~]$
```


ec2-user@ip-172-31-7-101:~
{'event_time': '2022-12-06T15:44:43.152154', 'ticker': 'XRP', 'price': 24.71}
{'event_time': '2022-12-06T15:44:43.152171', 'ticker': 'BNB', 'price': 93.08}
{'event_time': '2022-12-06T15:44:43.152186', 'ticker': 'ETH', 'price': 23.62}
{'event_time': '2022-12-06T15:44:43.152200', 'ticker': 'DOGE', 'price': 19.06}
{'event_time': '2022-12-06T15:44:43.152215', 'ticker': 'BTC', 'price': 92.64}
{'event_time': '2022-12-06T15:44:43.152230', 'ticker': 'ETH', 'price': 16.47}
{'event_time': '2022-12-06T15:44:43.152245', 'ticker': 'BNB', 'price': 54.48}
{'event_time': '2022-12-06T15:44:43.152259', 'ticker': 'ETH', 'price': 90.85}
{'event_time': '2022-12-06T15:44:43.152274', 'ticker': 'ETH', 'price': 31.46}
{'event_time': '2022-12-06T15:44:43.152289', 'ticker': 'DOGE', 'price': 79.19}
{'event_time': '2022-12-06T15:44:43.152304', 'ticker': 'DOGE', 'price': 43.17}
{'event_time': '2022-12-06T15:44:43.152318', 'ticker': 'BTC', 'price': 78.15}
{'event_time': '2022-12-06T15:44:43.152333', 'ticker': 'DOGE', 'price': 27.95}
{'event_time': '2022-12-06T15:44:43.152348', 'ticker': 'BNB', 'price': 89.05}
{'event_time': '2022-12-06T15:44:43.152363', 'ticker': 'ETH', 'price': 73.51}
{'event_time': '2022-12-06T15:44:43.152378', 'ticker': 'XRP', 'price': 12.46}
{'event_time': '2022-12-06T15:44:43.152392', 'ticker': 'DOGE', 'price': 18.63}
{'event_time': '2022-12-06T15:44:43.152407', 'ticker': 'DOGE', 'price': 3.02}
{'event_time': '2022-12-06T15:44:43.152422', 'ticker': 'ETH', 'price': 38.94}
{'event_time': '2022-12-06T15:44:43.152436', 'ticker': 'BNB', 'price': 81.4}
{'event_time': '2022-12-06T15:44:43.152451', 'ticker': 'ETH', 'price': 50.82}
{'event_time': '2022-12-06T15:44:43.152466', 'ticker': 'XRP', 'price': 14.99}
{'event_time': '2022-12-06T15:44:43.152481', 'ticker': 'BNB', 'price': 5.53}
{'event_time': '2022-12-06T15:44:43.152495', 'ticker': 'BNB', 'price': 32.62}
{'event_time': '2022-12-06T15:44:43.152510', 'ticker': 'PRCL', 'price': 25.84}
{'event_time': '2022-12-06T15:44:43.152525', 'ticker': 'PRCL', 'price': 51.85}
{'event_time': '2022-12-06T15:44:43.152540', 'ticker': 'ETH', 'price': 71.67}
{'event_time': '2022-12-06T15:44:43.152554', 'ticker': 'BNB', 'price': 67.31}
{'event_time': '2022-12-06T15:44:43.152569', 'ticker': 'PRCL', 'price': 95.7}
{'event_time': '2022-12-06T15:44:43.152584', 'ticker': 'B^C

Question 3)


Amazon Kinesis > Flux de données > Créer un flux de données
Créer un flux de données Info
Configuration des flux de données
Nom du flux de données
prisca-clio-stock-input-stream
Les caractères autorisés sont les lettres majuscules et minuscules, les chiffres, les traits de soulignement, les tirets et les points.
Flux de données prisca-clio-stock-input-stream créé avec succès.
Amazon Kinesis > Flux de données > prisca-clio-stock-input-stream
prisca-clio-stock-input-stream Info
Supprimer
Résumé du flux de données
Statut
Actif
Mode de capacité
À la demande
Période de rétention des données
1 jour
ARN
arn:aws:kinesis:eu-north-1:797644540006:stream/prisca-clio-stock-input-stream
Heure de création
06 décembre 2022 à 16:37 UTC+1
Applications
Surveillance
Configuration
Visionneuse de données
Diffusion améliorée (0)

Question 4)

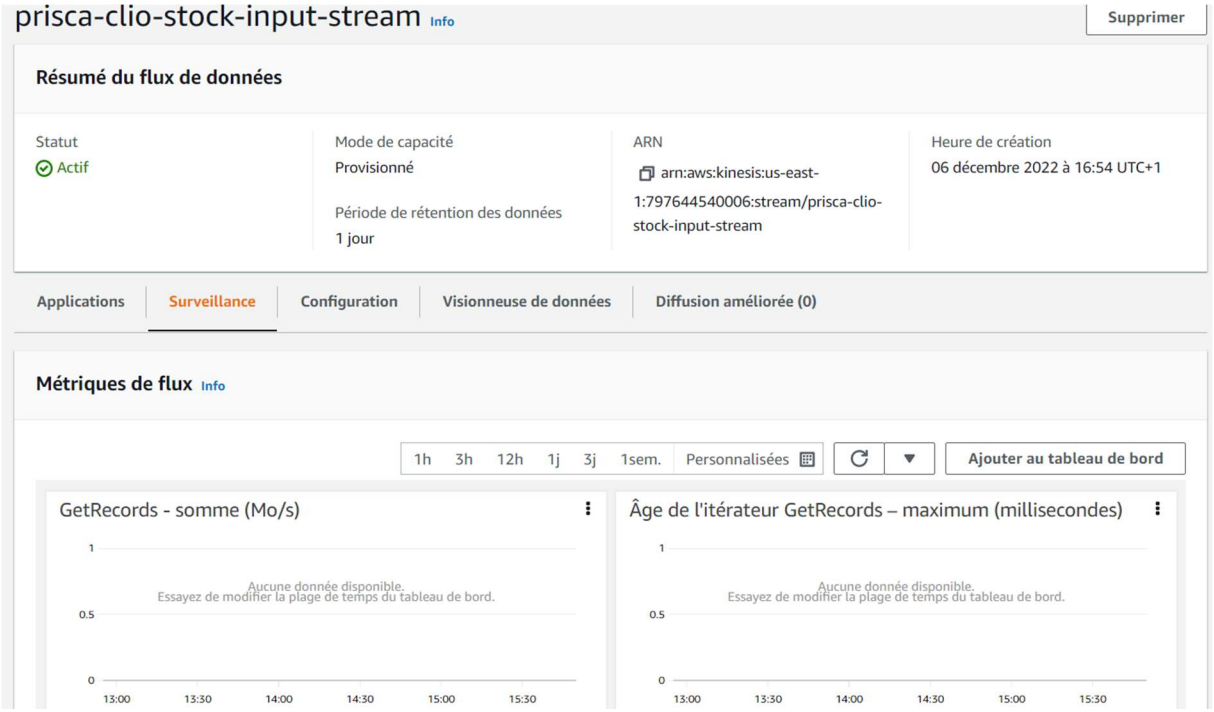
1) Supprimer le data stream

aws kinesis delete-stream --stream-name prisca-clio-stock-input-stream

2) Créer un data stream

```
aws kinesis create-stream --stream-name prisca-clio-stock-input-stream --shard-count 1
```

```
C:\Users\Prisca>aws kinesis create-stream --stream-name prisca-clio-stock-input-stream --shard-count 1

C:\Users\Prisca>
```

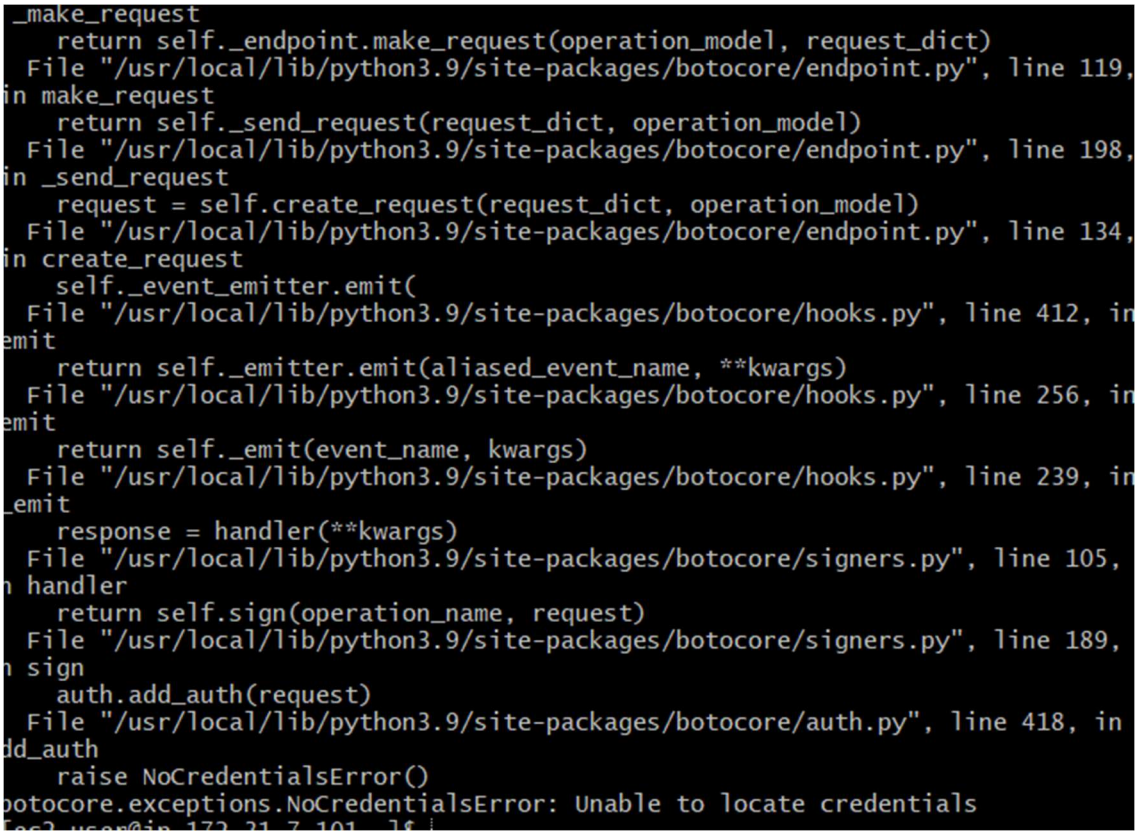


Question 5)

1)Modification du code python

```
[ec2-user@ip-172-31-7-101 ~]$ cat stock.py
import datetime
import json
import random
import boto3

STREAM_NAME = "prisca-clio-stock-input-stream"
REGION = "us-east-1"

def get_data():
    return {
        'event_time': datetime.datetime.now().isoformat(),
        'ticker': random.choice(["BTC","ETH","BNB", "XRP", "DOGE","PRCL"]),
        'price': round(random.random() * 100, 2)}

def generate(stream_name, kinesis_client):
    while True:
        data = get_data()
        print(data)
       #kinesis_client.put_record(StreamName=stream_name,Data=json.dumps(data),P
artitionKey="partitionkey")

if __name__ == '__main__':
    generate(STREAM_NAME, boto3.client('kinesis', region_name=REGION))
```

Exécution :



2)

3)

Pour supprimer le data stream on fait comme pour la question précedante et pour supprimer la vm on fait terraform destroy

Question 6)

```
$ terraform plan

Terraform used the selected providers to generate the following execution
plan. Resource actions are indicated with the following symbols:
  + create

Terraform will perform the following actions:

  # aws_kinesis_stream.test_stream will be created
  + resource "aws_kinesis_stream" "test_stream" {
      + arn                       = (known after apply)
      + encryption_type           = "NONE"
      + enforce_consumer_deletion = false
      + id                        = (known after apply)
      + name                      = "prisca-clio-stock-input-stream"
      + retention_period          = 24
      + shard_count               = 1
      + tags                      = {
          + "Name"  = "Etudiant"
          + "Owner" = "prisca.clio@etu.u-pec.fr"
        }
      + tags_all                  = {
          + "Name"  = "Etudiant"
          + "Owner" = "prisca.clio@etu.u-pec.fr"
        }

      + stream_mode_details {
          + stream_mode = (known after apply)
        }
    }

Plan: 1 to add, 0 to change, 0 to destroy.

Note: You didn't use the -out option to save this plan, so Terraform can't
guarantee to take exactly these actions if you run "terraform apply" now.
```

```
Do you want to perform these actions?
  Terraform will perform the actions described above.
  Only 'yes' will be accepted to approve.

  Enter a value: yes

aws_kinesis_stream.test_stream: Creating...
aws_kinesis_stream.test_stream: Still creating... [10s elapsed]
aws_kinesis_stream.test_stream: Still creating... [20s elapsed]
aws_kinesis_stream.test_stream: Creation complete after 23s [id=arn:aws:kinesis:
eu-north-1:797644540006:stream/prisca-clio-stock-input-stream]

Apply complete! Resources: 1 added, 0 changed, 0 destroyed.

Prisca@LAPTOP-9KJ7VP6F MINGW64 ~/Documents/EPISEN SI ING3/SEMESTRE 1/Pipiline tr
aintement cloud/CC/terraform
```