

<b>Computed</b>	<b>4</b>
<b>Storage</b>	<b>15</b>
Creating a Default S3 Bucket for Internal Data Storage	15
Uploading Files	15
Setting Up a Static Website on S3 Using the AWS CLI	16
<b>Networking</b>	<b>22</b>
Create a VPC (Virtual Private Cloud)	23
Create Subnets	23
Create and Attach an Internet Gateway	23
Create a Route Table for the Public Subnet	24
Create a NAT Gateway for the Private Subnet	24
Create a Route Table for the Private Subnet	25
Test Connectivity	25
Step 1: Launch an EC2 Instance in the Public Subnet	26
Step 2: Launch an EC2 Instance in the Private Subnet	26
<b>Exercises Networking I</b>	<b>27</b>
Exercise 1: Configuring Security Groups for a Web Server	27
<b>Exercises Networking II</b>	<b>29</b>
Task 1: Add the instances to the Load Balancer	29
Task 2: Create a Load Balancer	29
<b>Database</b>	<b>32</b>
Task 1: Create an RDS postgresSQL database	32
Task 2: Create a db connection in pgadmin4	32
<b>Lambda functies</b>	<b>34</b>
Algemene kennis lambda	34
Exercise 1	35
Exercise 2	37
Exercise 3	38
Exercise 4	40
<b>Api Gateway</b>	<b>49</b>
Exercise 1	49
Exercise 2	51
Exercise 3	52

```
PS C:\Users\bened\.aws> aws s3 mb s3://aws723
make_bucket: aws723
PS C:\Users\bened\.aws> aws s3 list

usage: aws [options] <command> <subcommand> [<subcommand> ...] [
parameters]
To see help text, you can run:

    aws help
    aws <command> help
    aws <command> <subcommand> help

aws.exe: error: argument subcommand: Invalid choice, valid choices are:

ls                               | website
cp                               | mv
rm                               | sync
mb                               | rb
presign
```

```
PS C:\Users\bened> aws s3 sync .aws/ s3://aws723
Completed 921 Bytes/965 Bytes (3.4 KiB/s) with 2 file(s) remaining
upload: .aws\credentials to s3://aws723/credentials

Completed 921 Bytes/965 Bytes (3.4 KiB/s) with 1 file(s) remaining
Completed 965 Bytes/965 Bytes (1.3 KiB/s) with 1 file(s) remaining
upload: .aws\config to s3://aws723/config

PS C:\Users\bened> |
```

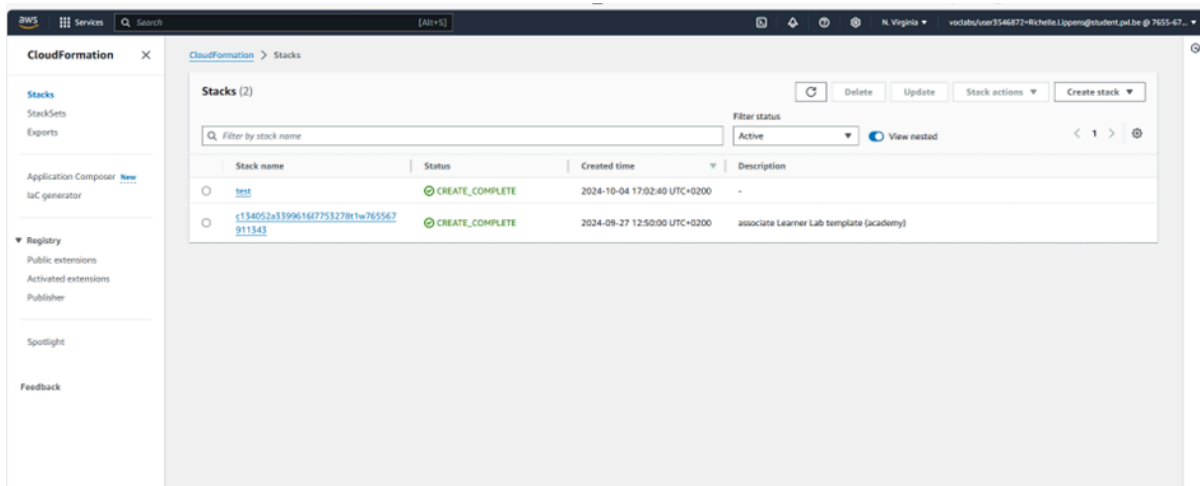


Lab: computed

Token controleren	aws sts get-caller-identity

## Computed

- What are the names of the 2 instances created?



- What is the name of the AMI that is used for both instances?

## Template

[View in Application Composer](#)



# This lab creates a VPC with instances, security groups, and key pairs  
 AWSTemplateFormatVersion: '2010-09-09'

[Copy](#)

### Resources:

# Create a VPC with the given name  
 Vpc: Vpc

```
# Solution
# instance names: instA, instB
# AMI name: Amazon Linux 2 AMI (HVM), SSD Volume Type
# Instance types: t2.micro, t2.large
# Public Ipv4: variable
# Keypair names: exc1
# security group names: ec2 incoming http
# rules for security group: allow incoming HTTP traffic on port 80
# why cost so high? t2.large instance type
```

- What is the instance type for both machines?

```
# Solution
# instance names: instA, instB
# AMI name: Amazon Linux 2 AMI (HVM), SSD Volume Type
# Instance types: t2.micro, t2.large
# Public Ipv4: variable
# Keypair names: exc1
# security group names: ec2 incoming http
# rules for security group: allow incoming HTTP traffic on port 80
# why cost so high? t2.large instance type
```

- What is the public IPv4 address for InstA?

klik op jouw stack > resources > kies jouw instance klik op jouw physical id  
 > selecteer jouw instance

instA i-0ebc6dc62a02279ff Running t2.micro 2/2 checks passed View alarms + us-east-1a ec2-3-91-39-76.comput... 3.91.39.76

**i-0ebc6dc62a02279ff (instA)**

Details Status and alarms Monitoring Security Networking Storage Tags

▼ Instance summary info

Instance ID  
i-0ebc6dc62a02279ff (instA)

Public IPv4 address  
3.91.39.76 open address

Private IPv4 addresses  
10.0.1.59

IPv6 address

Instance state

Public IPv4 DNS

instB i-01f993a489be4a249 Running t2.large 2/2 checks passed View alarms + us-east-1a ec2-54-174-230-53.co... 54.174.230.53

**i-01f993a489be4a249 (instB)**

Details Status and alarms Monitoring Security Networking Storage Tags

▼ Instance summary info

Instance ID  
i-01f993a489be4a249 (instB)

Public IPv4 address  
54.174.230.53 open address

Private IPv4 addresses  
10.0.1.45

IPv6 address

Instance state

Public IPv4 DNS

- What is the name for the keypair linked to both machines?

klik op jouw stack > resources > kijk naar jouw keypair > selecteer jouw instance (om te kijken of welk gekoppeld is)

Stack info Events Resources Outputs Parameters Template Change sets Git sync

**Resources (6)**

Search resources

Logical ID	Physical ID	Type	Status	Module
Ec2IncomingHttpSecurityGroup	sg-05098a0f287244d23	AWS::EC2::SecurityGroup	CREATE_COMPLETE	-
Exc1KeyPair	exc1	AWS::EC2::KeyPair	CREATE_COMPLETE	-
Labo2DebugVPC	vpc-08fe215974b29f272	AWS::EC2::VPC	CREATE_COMPLETE	-
Labo2InstanceA	i-0ebc6dc62a02279ff	AWS::EC2::Instance	CREATE_COMPLETE	-
Labo2InstanceB	i-01f993a489be4a249	AWS::EC2::Instance	CREATE_COMPLETE	-
PublicSubnet	subnet-05399ad8d8306782d	AWS::EC2::Subnet	CREATE_COMPLETE	-

<input checked="" type="checkbox"/>	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS	Public IPv4 ...	Elastic IP
<input checked="" type="checkbox"/>	instA	i-0ebc6dc62a02279ff	Running	t2.micro	2/2 checks passed	<a href="#">View alarms</a>	us-east-1a	ec2-3-91-39-76.comput...	3.91.39.76	-

**i-0ebc6dc62a02279ff (instA)**

Stop protection  
Disabled

Instance auto-recovery  
Default

AMI Launch index  
0

Credit specification  
standard

Usage protection

Launch time  
Thu Oct 10 2024 15:34:07 GMT+0200 (Midden-Europese zomertijd) (9 minutes)

Lifecycle  
normal

Key pair assigned at launch  
ec2-**ec2**

Kernel ID  
-

RAM disk ID

AMI location  
amazon/amzn2-ami-kernel-5.10-hvm-2.0.20240412.0-x86\_64-gp2

Stop-hibernate behavior  
Disabled

State transition reason  
-

State transition message  
-

Source

- Search for a linked security group. What is the name of this group?

<input checked="" type="checkbox"/>	Name	Security group ID	Security group name	VPC ID	Description	Owner
<input checked="" type="checkbox"/>	-	sg-05098a0f287244d23	ec2 incoming http	vpc-08fe215974b29f272	Allow incoming HTTP traffic on port 80	765567911343

Details

Inbound rules

Outbound rules

Tags

**Details**

Security group name  
ec2-**ec2 incoming http**

Owner  
765567911343

Security group ID  
sg-05098a0f287244d23

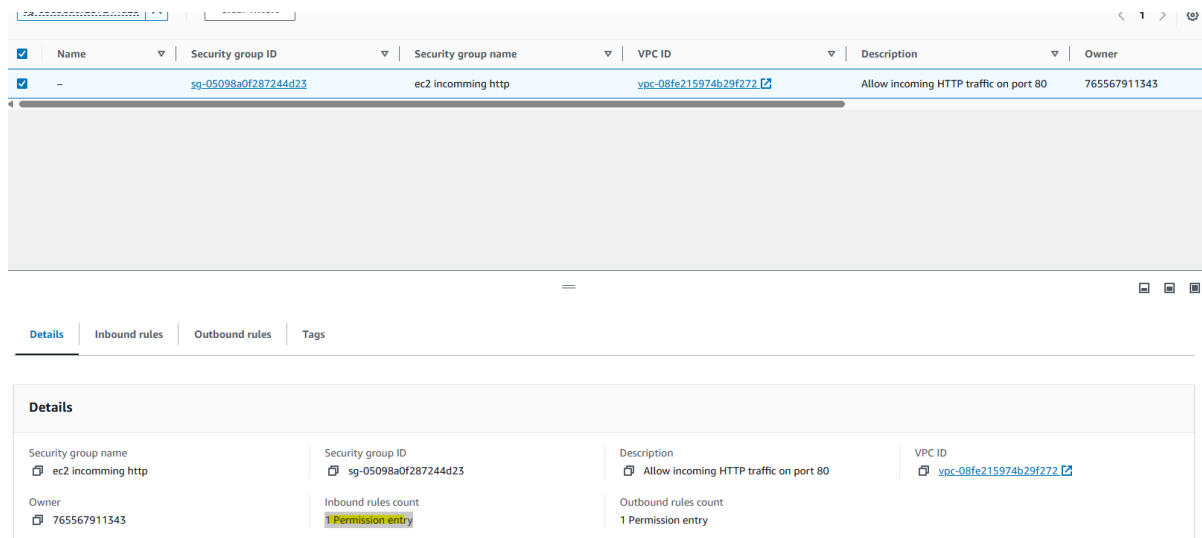
Inbound rules count  
1 Permission entry

Description  
Allow incoming HTTP traffic on port 80

Outbound rules count  
1 Permission entry

VPC ID  
vpc-08fe215974b29f272

- What rules are present in this security group?



- Our customer is complaining that the costs for his EC2 instances are expensive. What causes this issue and how would you fix this?

Verbinden met een instance	ssh -i /path/to/vockey.pem ec2-user@<public-ip-of-instance>
Controleer de cloud-init status	sudo cloud-init status --long
Bekijk de cloud-init logbestanden	sudo cat /var/log/cloud-init-output.log




Problemen met webserver

Stap1: verbinden met instance via ssh

```
ssh -i /path/to/vockey.pem ec2-user@<public-ip-of-instance>
```

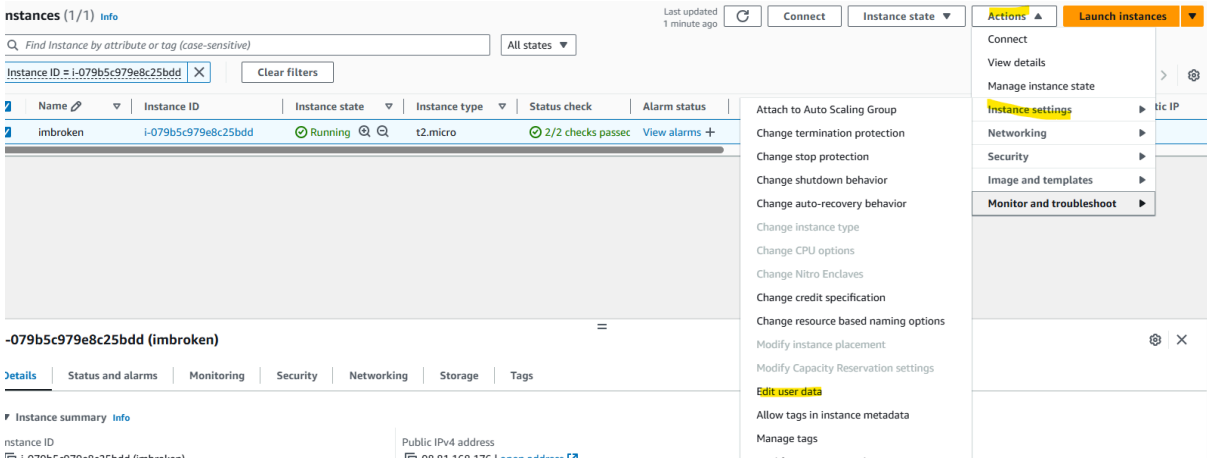
stap 2: kijk in de log file en zoek naar de foutmelding

sudo cat /var/log/cloud-init-output.log

```
Complete!
starting httpd service
Unknown operation 'starthttpd'.
enable httpd service
Created symlink from /etc/systemd/system/multi-user.target.wants/httpd.service to /usr/lib/systemd/system/httpd.service.
Cloud-init v. 19.3-46.amzn2.0.1 finished at Thu, 10 Oct 2024 13:55:11 +0000. Datasource DataSourceEc2.
```

stap 3: kijk in de console

resources > physical id > klik op de id van jouw instance > selecteer jouw instance > actions > instance settings > edit user data




stap 4: stop de intance, herlaad, en pas de user data correct aan

```
#!/bin/bash
echo "running yum update -y"
yum update -y
echo "running yum install httpd"
yum install -y httpd
echo "starting httpd service"
systemctl start httpd
echo "enable httpd service"
systemctl enable httpd
```

## Edit user data [Info](#)


Instance ID

 i-079b5c979e8c25bdd (imbroken)

### Current user data

User data currently associated with this instance

```
#!/bin/bash
echo "running yum update -y"
yum update -y
echo "running yum install httpd"
yum install -y httpd
echo "starting httpd service"
```

 Copy user data

### New user data

This user data will replace the current user data

☒ Modify user data as text

Add your user data below

☐ Modify user data by importing a file

Description of importing a file and what will happen to it

```
#!/bin/bash
echo "running yum update -y"
yum update -y
echo "running yum install httpd"
yum install -y httpd
echo "starting httpd service"
```


☐ Input is already base64-encoded

Cancel

Save

stap 5: open server en verwijder **s** in https









**Instance summary for i-01bb1239b18893121 (imbroken)** [Info](#)

 **Connect**

**Instance state** ▼

**Actions** ▼

Updated 1 minute ago

<b>Instance ID</b>  i-01bb1239b18893121 (imbroken)	<b>Public IPv4 address</b>  3.83.126.122   <a href="#">open address</a> 	<b>Private IPv4 addresses</b>  10.0.2.169
<b>IPv6 address</b> -	<b>Instance state</b>  <b>Running</b>	<b>Public IPv4 DNS</b>  ec2-3-83-126-122.compute-1.amazonaws.com   <a href="#">open address</a> 
<b>Hostname type</b> IP name: ip-10-0-2-169.ec2.internal	<b>Private IP DNS name (IPv4 only)</b>  ip-10-0-2-169.ec2.internal	

## Automatische software installeren met instance

stap1: launch an EC2 instance

stap2: ga naar advaced details ga onderaan naar User Data

stap3: vul een script in die automatische software installeert (bv. docker) en launch the instance

```
#!/bin/bash
sudo yum update -y
sudo yum install -y docker
sudo service docker start
docker run hello-world
```

Stap4: log in op ssh instance en voer volgend commando uit om te controleren

```
sudo cat /var/log/cloud-init-output.log
```

```
Digest: sha256:d211f485f2dd1dee407a80973c8f129f00d54604d2c90732e8e320e5038a0348
Status: Downloaded newer image for hello-world:latest

Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
```

stap5: controller docker image

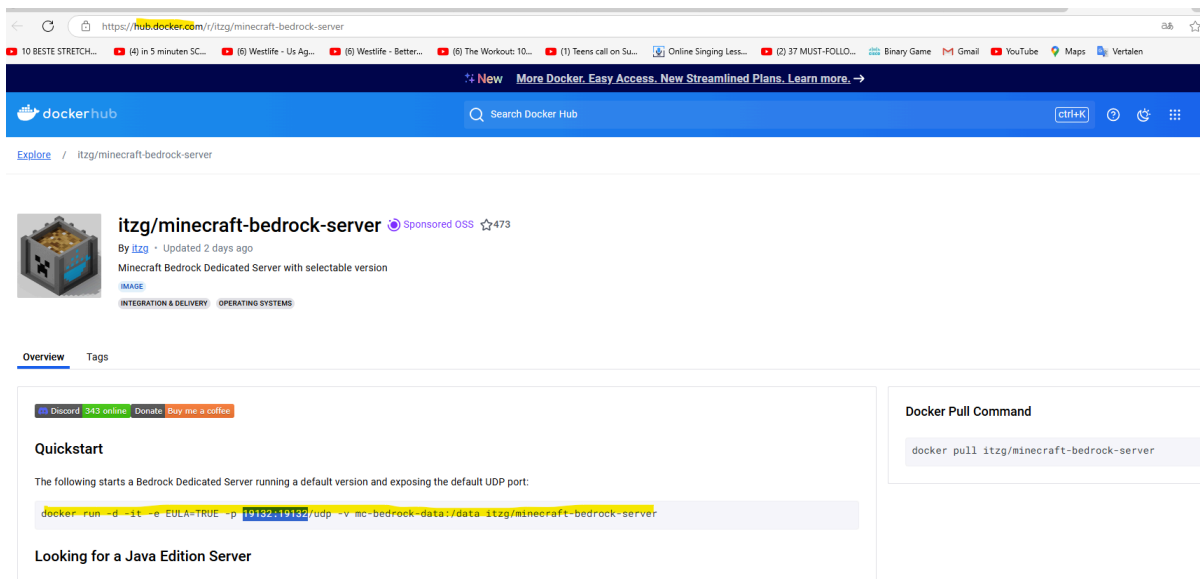
```
sudo docker container ls -a
```

```
[ec2-user@ip-172-31-39-82 ~]$ sudo docker container ls -a
CONTAINER ID   IMAGE          COMMAND                  CREATED        STATUS        PORTS          NAMES
6d287a96df43   hello-world    "/hello"                6 minutes ago  Exited (0)   6 minutes ago  sad_kalam
```

Image open in docker

Stap 1: voer commando uit om minecraft image te krijgen

```
docker run -d -it -e EULA=TRUE -p 25565:25565/udp -v mc-bedrock-data:/data itzg/minecraft-bedrock-server
```



https://hub.docker.com/r/itzg/minecraft-bedrock-server

New More Docker. Easy Access. New Streamlined Plans. Learn more. →

dockerhub

Search Docker Hub

Explore / itzg/minecraft-bedrock-server

itzg/minecraft-bedrock-server Sponsored OSS 473

By itzg · Updated 2 days ago

Minecraft Bedrock Dedicated Server with selectable version

IMAGE

INTEGRATION & DELIVERY OPERATING SYSTEMS

Overview Tags

Quickstart

The following starts a Bedrock Dedicated Server running a default version and exposing the default UDP port:

```
docker run -d -it -e EULA=TRUE -p 19132:19132/udp -v mc-bedrock-data:/data itzg/minecraft-bedrock-server
```

Looking for a Java Edition Server

Docker Pull Command

```
docker pull itzg/minecraft-bedrock-server
```

Stap 2: voeg unbound rules toe aan groups security

Auto-assigned IP address  
54.164.97.46 [Public IP]

IAM Role  
-

IMDSv2  
Required

VPC ID  
vpc-00062b62298760fe7

Subnet ID  
subnet-01f012b46b6b4d1f4

Instance ARN  
arn:aws:ec2:us-east-1:765567911343:instance/i-0a959de15cedf9520

AWS Compute Optimizer finding  
Opt-in to AWS Compute Optimizer for recom

Auto Scaling Group name  
-

Details

Status and alarms

Monitoring

Security

Networking

Storage

Tags

▼ Security details

IAM Role  
-

Owner ID  
765567911343

Launch time  
Thu Oct 10 2024 16:58:11 GMT+0200 (Midden-I

Security groups

sg-08bb09e76ca95edd1 (launch-wizard-3)

▼ Inbound rules

Filter rules

Name	Security group rule ID	Port range	Protocol	Source	Security groups
-	sgr-0f0b1952442a93743	22	TCP	0.0.0.0/0	launch-wizard-3

▼ Outbound rules


Filter rules

Inbound rules Info

Security group rule ID	Type Info	Protocol Info	Port range Info	Source Info	Description - optional Info
sgr-0aecfeca8af2e524b	Custom TCP	TCP	25565	An... <div>0.0.0.0/0 X</div>	<div></div> <div>Delete</div>
sgr-0b339356c575d7386	Custom TCP	TCP	25565	An... <div>:::0 X</div>	<div></div> <div>Delete</div>
sgr-0e5a81994a67144b5	SSH	TCP	22	Cus... <div>0.0.0.0/0 X</div>	<div></div> <div>Delete</div>


Add rule

Stap3: check of alles werkt. Conclusie het werkt niet en we mogen het nog eens vragen aan de leerkracht

 Minecraft Server Status

FAQ System status API

This site is ad-free and I would like to keep it that way. Please consider [donating](#) to keep it running. Thanks :) ×



# Minecraft Server Status

Get information about Minecraft servers quickly

Get server status

☐ Bedrock server? Minecraft Java (1.7+), Minecraft Bedrock or servers with `enable-query=true` are supported.

## Could not get the server status...

Did you type the correct address? The server may also be temporarily down. Please try again later.

# Storage

## Creating a Default S3 Bucket for Internal Data Storage

stap1: Open console >

[Amazon S3](#) > [Buckets](#) > **Create bucket**

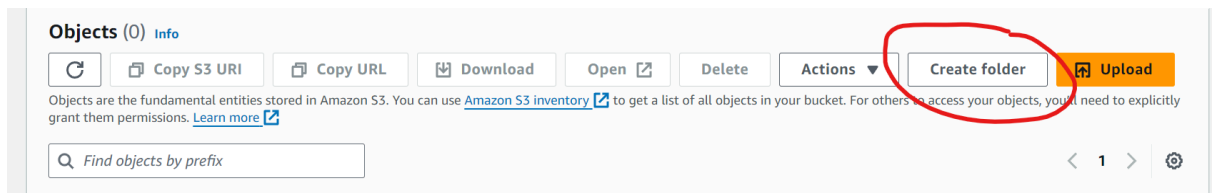
stap2: geef unique naam zonder hoofdletters

stap3: laat de rest staan

## Uploading Files

stap 1 : klik op de bucket

stap 2 : Create folder



stap 3 : Upload een file in je folder

Zie hier S3 and HTTP endpoints

**Summary**

Destination  
<s3://myuniquebucket2024/logs/>

Succeeded  
✔ 1 file, 172.0 B (100.00%)

Failed  
☹ 0 files, 0 B (0%)

**Files and folders** | Configuration

**Files and folders** (1 Total, 172.0 B)

Name	Folder	Type	Size	Status	Error
<a href="#">test.zip</a>	-	application/x...	172.0 B	✔ Succeeded	-

**Controleer** de setup met volgende commando's (kijk of het lijkt op de screenshot)

```
aws s3 ls
aws s3 ls s3://my-bucket/my-folder/
```

```
PS C:\Users\12101041\Desktop\Prive\CloudEssentials> aws s3 ls
2024-10-04 17:02:12 cf-templates-plw9b7g7o5s1-us-east-1
2024-10-17 15:38:17 myuniquebucket2024
PS C:\Users\12101041\Desktop\Prive\CloudEssentials> aws s3 ls s3://myuniquebucket2024/logs/
2024-10-17 15:39:50      0
2024-10-17 15:41:52    172 test.zip
PS C:\Users\12101041\Desktop\Prive\CloudEssentials>
```

## Setting Up a Static Website on S3 Using the AWS CLI

1) Create a new S3 bucket

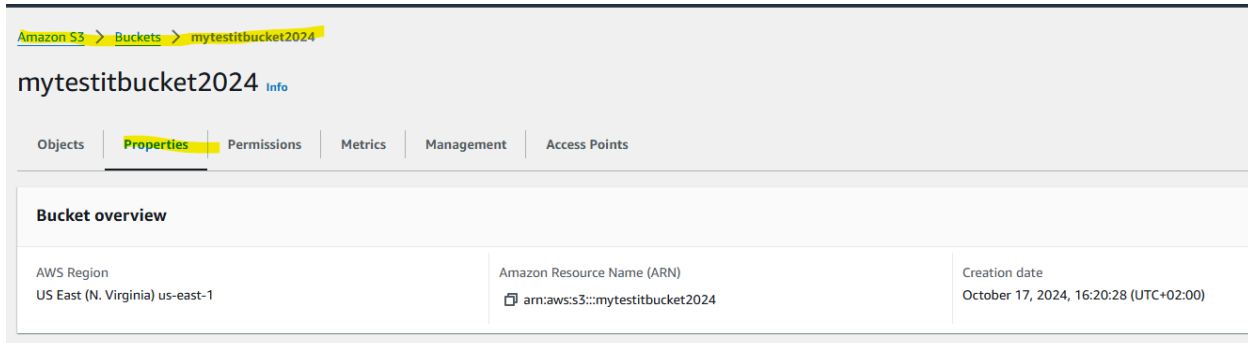
```
aws s3 mb s3://your-bucket-name
```

```
PS C:\Users\12101041\Desktop\Prive\CloudEssentials> aws s3 mb s3://myuniquebucket2024/
make_bucket: myuniquebucket2024
```

2) Enable Static Website Hosting

```
aws s3 website s3://your-bucket-name/ --index-document index.html
```





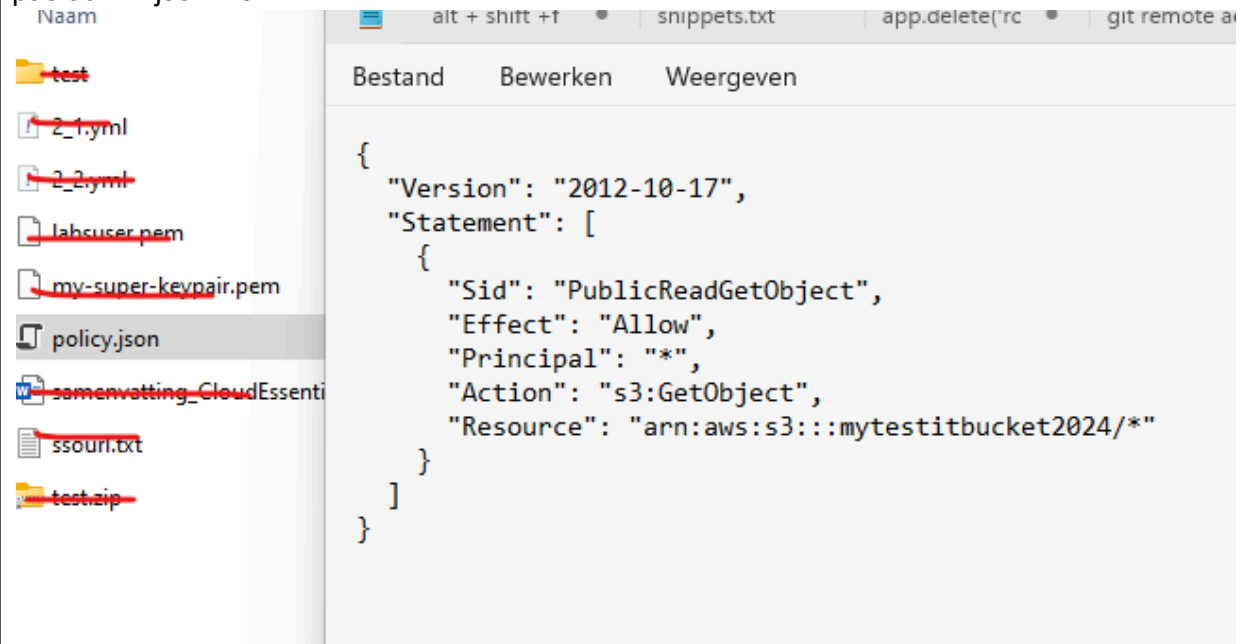
### 3) Disable Block Public Access

```
aws s3api put-public-access-block --bucket mytestitbucket2024
--public-access-block-configuration
"BlockPublicAcls=false,IgnorePublicAcls=false,BlockPublicPolicy=false,RestrictPublicBuckets
=false"
```

### 4) Add a Public Read Policy

```
aws s3api get-bucket-policy --bucket bucketname --query Policy --output text > policy.json
```

pas aan in json file



```
aws s3api put-bucket-policy --bucket bucketname --policy file://policy.json
```

### 5) Upload Your Website Content

Stap 1: creëer een html file

open powershell > voer commando **code .** in > maak een file index.html > voer commando **! +** tab in

```
aws s3 sync . s3://bucketname
```

## 6) Test Your Static Website

The screenshot shows the Amazon S3 console interface. At the top, the breadcrumb navigation indicates the path: Amazon S3 > Buckets > mytestitbucket2024 > s3/ > index.html. The main heading is 'index.html' with an 'Info' link. To the right of the heading are buttons for 'Copy S3 URI', 'Download', 'Open', and 'Object actions'. Below the heading are tabs for 'Properties', 'Permissions', and 'Versions'. The 'Properties' tab is selected, showing the 'Object overview' section. This section is divided into two columns. The left column contains the following information: Owner (awslabs0w6312615t1694813098), AWS Region (US East (N. Virginia) us-east-1), Last modified (October 17, 2024, 16:59:43 (UTC+02:00)), Size (231.0 B), Type (html), and Key (s3/index.html). The right column contains the following information: S3 URI (s3://mytestitbucket2024/s3/index.html), Amazon Resource Name (ARN) (arn:aws:s3:::mytestitbucket2024/s3/index.html), Entity tag (Etag) (654ee1fe0976a4778e3ac15810e3a302), and Object URL (https://mytestitbucket2024.s3.amazonaws.com/s3/index.html). Below the console, a browser window is shown with the address bar displaying the URL https://mytestitbucket2024.s3.amazonaws.com/s3/index.html. The browser's tab bar shows several tabs, including '10 BESTE STRETCH...', '(4) in 5 minuten SC...', '(6) Westlife - Us Ag...', '(6) Westlife - Better...', and '(6) The'. The main content area of the browser displays the word 'Success' in a large, bold, black font.

## Enabling Bucket Versioning and Using Pre-Signed URLs

Enable versioning on your bucket using the AWS CLI

```
aws s3api put-bucket-versioning --bucket mytestitbucket2024  
--versioning-configuration Status=Enabled
```

Upload a file to your bucket

```
aws s3 cp .\mytestitbucket2024.txt s3://mytestitbucket2024
```

Amazon S3 > Buckets > mytestitbucket2024


## mytestitbucket2024 [Info](#)

[Objects](#) | [Properties](#) | [Permissions](#) | [Metrics](#) | [Management](#) | [Access Points](#)

**Objects (1) [Info](#)** [Refresh](#) [Copy S3 URI](#) [Copy URL](#) [Down Arrow](#)

Objects are the fundamental entities stored in Amazon S3. You can use [Amazon S3 inventory](#) to get a list of all objects in your bucket. For others to acc

☐ Show versions

<input type="checkbox"/>	Name	Type	Last modified
<input type="checkbox"/>	 mytestitbucket2024.txt	txt	October 17, 2024, 17:19:20 (L

List all the versions of the file stored in your bucket

```
aws s3api list-object-versions --bucket bucketname --prefix filename.txt
```

```
PS C:\Users\12101041\Desktop\Prive\CloudEssentials> aws s3api list-object-versions --bucket mytestitbucket2024 --prefix mytestitbucket2024.txt
{
  "Versions": [
    {
      "ETag": "\"8ce0b6f65a26656c9839b2a7d435e838\"",
      "Size": 18,
      "StorageClass": "STANDARD",
      "Key": "mytestitbucket2024.txt",
      "VersionId": "57NZ_vfVJynkXw29GwPkaczFwi7VJKPd",
      "IsLatest": true,
      "LastModified": "2024-10-17T15:19:20+00:00",
      "Owner": {
        "DisplayName": "awslabsc0w6312615t1694813098",
        "ID": "9a29fa52b845ade2874f1c3693a260a23e1bb4db4fcac38b2212301d30184df8"
      }
    }
  ],
  "RequestCharged": null
}
```


Amazon S3 > Buckets > mytestitbucket2024 > mytestitbucket2024.txt

## mytestitbucket2024.txt Info

[Copy S3 URI](#) [Download](#) [Open](#) [Object a](#)

Properties | Permissions | **Versions**

Versions (1) [Download](#) [Open](#) [Delete](#) [Acti](#)

<input type="checkbox"/>	Version ID	Type	Last modified	Size	Storage class
<input type="checkbox"/>	 s57NZ_vfVjynkXw29GwPkaczFwi7V... (Current version)	txt	October 17, 2024, 17:19:20 (UTC+02:00)	18.0 B	Standard

Amazon S3 > Buckets > mytestitbucket2024

## mytestitbucket2024 Info

Objects | Properties | **Permissions** | Metrics | Management | Access Points

### Permissions overview

Access finding

Access findings are provided by IAM external access analyzers. Learn more about [How IAM analyzer findings work](#)

[View analyzer for us-east-1](#)

### Block public access (bucket settings)

Public access is granted to buckets and objects through access control lists (ACLs), bucket policies, access point policies, or all. In order to ensure that public access to all your S3 buckets and objects is blocked, turn on Block all public access for this bucket and its access points. AWS recommends that you turn on Block all public access, but before applying any of these settings, ensure that your applications will work correctly without public access. If you require public access for objects within, you can customize the individual settings below to suit your specific storage use cases. [Learn more](#)

```
aws s3 cp mytestitbucket2024.txt  
s3://mytestitbucket2024/mytestitbucket2024.txt --acl private
```

mytestitbucket2024.txt

Copy S3 URIDownloadOpenObject actions

PropertiesPermissionsVersions

Access control list (ACL)

Grant basic read/write permissions to AWS accounts. [Learn more](#)

This bucket has the bucket owner enforced setting applied for Object Ownership

When bucket owner enforced is applied, use bucket policies to control access. [Learn more](#)

Grantee	Object	Object ACL
Object owner (your AWS account) Canonical ID: 9a29fa52b845ade2874f1c3693a260a23e1bb4db4fcac38b2212301d30184df8	Read	Read, Write
Everyone (public access) Group: <a href="http://acs.amazonaws.com/groups/global/AllUsers">http://acs.amazonaws.com/groups/global/AllUsers</a>	-	-
Authenticated users group (anyone with an AWS account) Group: <a href="http://acs.amazonaws.com/groups/global/AuthenticatedUsers">http://acs.amazonaws.com/groups/global/AuthenticatedUsers</a>	-	-

Amazon S3 > Buckets > mytestitbucket2024 > Edit Object Ownership

Edit Object Ownership

Object Ownership

Control ownership of objects written to this bucket from other AWS accounts and the use of access control lists (ACLs). Object ownership determines who can specify access to objects.

ACLs disabled (recommended)

All objects in this bucket are owned by this account. Access to this bucket and its objects is specified using only policies.

ACLs enabled

Objects in this bucket can be owned by other AWS accounts. Access to this bucket and its objects can be specified using ACLs.

Object Ownership

Bucket owner enforced

Cancel

Save changes

## NAT gateway settings

### Name - optional

Create a tag with a key of 'Name' and a value that you specify.

MyNATGateway

The name can be up to 256 characters long.

### Subnet

Select a subnet in which to create the NAT gateway.

subnet-0f12f3082871810fd (PublicSubnet)

### Connectivity type

Select a connectivity type for the NAT gateway.

☒ Public

☐ Private

### Elastic IP allocation ID [Info](#)

Assign an Elastic IP address to the NAT gateway.

eipalloc-02e41be408157ab4e

[Allocate Elastic IP](#)

## VPC - required [Info](#)

vpc-08c3e76f8ad43bcee (MycustomVPC)  
10.0.0.0/16



### Subnet [Info](#)

subnet-0f12f3082871810fd  
VPC: vpc-08c3e76f8ad43bcee Owner: 852657213420  
Availability Zone: us-east-1a Zone type: Availability Zone  
IP addresses available: 250 CIDR: 10.0.1.0/24

PublicSubnet



[Create new subnet](#)

### Auto-assign public IP [Info](#)

Enable

Additional charges apply when outside of free tier allowance

### 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 reach your instance.

☒ Create security group

☐ Select existing security group

### Security group name - required

launch-wizard-4

This security group will be added to all network interfaces. The name can't be edited after the security group is created. Max

# Networking

---

## Create a VPC (Virtual Private Cloud)

- Go to the VPC Dashboard in the AWS Console.
- Click on **Create VPC**.
- In the VPC settings, choose **VPC only** and enter:
  - Name tag: **MyCustomVPC**.
  - IPv4 CIDR block: **10.0.0.0/16**. > Type dit zelf!!!!
  - Leave everything else as default and click Create VPC.

## Create Subnets

1. Go to Subnets under the VPC Dashboard.
2. Click Create subnet.
3. In the Create subnet section:
  - Select the VPC you just created (**MyCustomVPC**)
  - Subnet 1 (Public Subnet):
    - Name: **PublicSubnet**
    - Availability Zone: Select **us-east-1a**.
    - IPv4 CIDR block: **10.0.1.0/24**
    - Click Add another subnet.
  - Subnet 2 (Private Subnet):
    - Name: **PrivateSubnet**
    - Availability Zone: Use the same as the Public Subnet
    - IPv4 CIDR block: **10.0.2.0/24**
4. Click Create subnet.

## Create and Attach an Internet Gateway

1. Go to Internet Gateways under the VPC Dashboard.
2. Click Create Internet Gateway.
3. Provide the following details:
  - Name tag: **MyIGW**.
4. Click Create internet gateway.
5. After creation, select the new Internet Gateway and click Actions > Attach to VPC.
6. Choose the VPC (**MyCustomVPC**)
7. Click Attach Internet Gateway

## Create a Route Table for the Public Subnet

1. Go to Route Tables under the VPC Dashboard.
2. Click Create Route Table.
3. Provide the following details:
  - Name tag: **PublicRouteTable**.
  - VPC: Select **MyCustomVPC**.
4. Click Create route table.
5. After creation, select the PublicRouteTable and go to the Routes tab.
6. Click Edit routes and add a route:
  - Destination: **0.0.0.0/0**.
  - Target: Select **Internet Gateway** and choose **MyIGW**.
7. Click Save routes.
8. Now go to the Subnet Associations tab.
9. Click Edit subnet associations, select **PublicSubnet**,
10. click Save associations.

## Create a NAT Gateway for the Private Subnet

1. Go to Elastic IPs under the VPC Dashboard.
2. Click Allocate Elastic IP address and then



3. **click Allocate.**
4. Go to NAT Gateways under the VPC Dashboard.
5. **Click Create NAT Gateway** and provide the following details:
  - Name tag: **MyNATGateway**.
  - Subnet: Select **PublicSubnet**.
  - Elastic IP Allocation ID: Select the Elastic IP you just allocated.
6. Click **Create NAT Gateway** and wait until it becomes available.

## Create a Route Table for the Private Subnet

1. Go to Route Tables under the VPC Dashboard.
2. **Click Create Route Table.**
3. Provide the following details:
  - Name tag: **PrivateRouteTable**.
  - VPC: Select **MyCustomVPC**.
4. **Click Create route table.**
5. After creation, select the PrivateRouteTable and go to the Routes tab.
6. **Click Edit routes and add a route:**
  - Destination: **0.0.0.0/0**.
  - Target: Select NAT Gateway and choose **MyNATGateway**.
7. **Click Save routes.**
8. Now go to the Subnet Associations tab.
9. **Click Edit subnet associations**, select PrivateSubnet, and **click Save associations.**

## Test Connectivity

## Step 1: Launch an EC2 Instance in the Public Subnet

10. Go to EC2 Dashboard and click **Launch Instance**.
11. Provide the following details:
  - Name: **PublicInstance**
  - AMI: Select Amazon Linux
  - Instance Type: **t2.micro** (Free tier eligible)
  - **Edit** Network settings:
    - i. VPC: Select **MyCustomVPC**
    - ii. Subnet: Select **PublicSubnet**
    - iii. Auto-assign Public IP: **Enabled**
  - Security Group:
    - i. **Create security group** allowing SSH (port 22) from your IP address. **(My ip)**
12. **Launch the instance** and wait for it to be running.

## Step 2: Launch an EC2 Instance in the Private Subnet

1. Go back to the EC2 Dashboard and click Launch Instance.
2. Provide the following details:
  - Name: **PrivateInstance**.
  - AMI: Select Amazon Linux.
  - Instance Type: **t2.micro** (Free tier eligible).
  - Select **Vockey**
  - **Edit** Network settings:
    - VPC: Select **MyCustomVPC**.
    - Subnet: Select **PrivateSubnet**.
    - Auto-assign Public IP: Disabled.

- Security Group: Create a new security group allowing SSH (port 22) from ANY address.
- 3. **Launch the instance and wait for it to be running.**
- 4. SSH into the PublicInstance (using its public IP) and then SSH into the
- 5. `ssh ec2-user@<pubIP> -i "C:\Users\Josephine\Downloads\labsuser.pem"`
- 6. PrivateInstance from the PublicInstance using the private IP of the private instance:
- 7. `ssh ec2-user@<privateIP> -i "C:\Users\Josephine\Downloads\labsuser.pem"`

## Exercises Networking I

### Exercise 1: Configuring Security Groups for a Web Server

1. **Launch an EC2 instance in the PublicSubnet:**
  - Name: **WebServerInstance**.
  - AMI: Amazon Linux 2.
  - Instance Type: **t2.micro**.
  - Key pair: **vockey**
  - Subnet: **PublicSubnet**.
  - Auto-assign Public IP: Enabled.
2. Configure the security group for the instance:
  - Allow HTTP (port 80) traffic from anywhere (**0.0.0.0/0**).
  - Allow SSH (port 22) only from your IP address.
3. Install a web server:
  - SSH into the instance and run:
  - `sudo yum update -y sudo yum install httpd -y sudo systemctl start httpd sudo systemctl enable httpd`
  -

- Verify the web server is running by accessing the instance's public IP in a browser: **http://<public-ip>**.
4. Test: Ensure that the HTTP traffic is accessible publicly but SSH is restricted to your IP. → **kun je niet checken**

Problemen: SSH wel toegang mag niet waarom?

Security group rule 1 (TCP, 80, 0.0.0.0/0) Remove

Type <a href="#">Info</a>	Protocol <a href="#">Info</a>	Port range <a href="#">Info</a>
HTTP ▼	TCP	80
Source type <a href="#">Info</a>	Source <a href="#">Info</a>	Description - optional <a href="#">Info</a>
Anywhere ▼	<input type="text" value="0.0.0.0/0"/>	e.g. SSH for admin desktop

▼ Security group rule 2 (TCP, 22, 178.144.115.173/32) Remove

Type <a href="#">Info</a>	Protocol <a href="#">Info</a>	Port range <a href="#">Info</a>
ssh ▼	TCP	22
Source type <a href="#">Info</a>	Name <a href="#">Info</a>	Description - optional <a href="#">Info</a>
My IP ▼	<input type="text" value="178.144.115.173/32"/>	e.g. SSH for admin desktop

```
connection to 178.144.115.173 closed.
PS C:\Users\Josephine> ssh ec2-user@178.144.115.173 -i "C:\Users\Josephine\Downloads\labsuser.pem"
ssh: connect to host 178.144.115.173 port 22: Connection timed out
```

▼ Inbound rules

Filter rules					
	Security group rule ID	Port range	Protocol	Source	Security groups
	sgr-035089ab9ee51ac9f	22	TCP	178.144.115.173/32	<a href="#">launch-wizard-3</a>
	sgr-0e1deb6eef7bd67a	80	TCP	0.0.0.0/0	<a href="#">launch-wizard-3</a>

## Exercises Networking II

### Task 1: Add the instances to the Load Balancer

- Choose **Create target group**
- Choose a target type: Instances
- Target group name, enter: MyLabGroup
- Select my-VPC from the VPC drop-down menu.
- Choose **Next**. The Register targets screen appears.  
Note: Targets are the individual instances that will respond to requests from the Load Balancer.
- select the 5 EC2s
- Include as pending below 5 EC2s –
- **Create target group**

### Task 2: Create a Load Balancer

- **Create load balancer**
- choose Application Load Balancer
- vul **name** in en kies **my-vpc**
- **selecteer de 5 availability zones**

peering. To confirm the VPC for your targets, vi

my-vpc

vpc-0919554da08330594

IPv4 VPC CIDR: 10.0.0.0/16



## Mappings [Info](#)

Select at least two Availability Zones and one si  
in these Availability Zones only. Availability Zon  
are not available for selection.

### Availability Zones

☐ **us-east-1a (use1-az2)**

☐ **us-east-1b (use1-az4)**

☐ **us-east-1c (use1-az6)**

☐ **us-east-1d (use1-az1)**

● ☐ **us-east-1f (use1-az5)**

- selecteer een default security group als dat er is anders volg de stap beneden

#### 1. Create a Security Group for the Load Balancer

- **Name:** LoadBalancer-SG
- **Inbound Rule:** Allow HTTP (port 80) traffic from the internet
  - Protocol: HTTP
  - Port Range: 80
  - Source: 0.0.0.0/0 (to allow all incoming traffic on port 80)
- **Outbound Rule:** Default (Allow all outbound traffic, or restrict as per requirement)

#### 2. Create a Security Group for the Web Application Instances

- **Name:** WebAppInstances-SG
- **Inbound Rule:** Allow traffic from the Load Balancer on HTTP (port 80)
  - Protocol: HTTP
  - Port Range: 80
  - **Source:** LoadBalancer-SG (select the ID of the LoadBalancer-SG security group as the source)
- **Outbound Rule:** Default (Allow all outbound traffic, or restrict as per requirement)



- ga terug naar vorige tabblad en selecteer de nieuwe security group
- Listener HTTP:80 row, set the Default action to forward to MyLabGroup.
- create

# LabELB

Actions ▾

▼ Details			
Load balancer type	Status	VPC	Load balancer IP address type
Application	✔ Active	<a href="#">vpc-0919554da08330594</a> ⓘ	IPv4
Scheme	Hosted zone	Availability Zones	Date created
Internet-facing	Z35SXDOTRQ7X7K	<a href="#">subnet-05b375b925fcb2e26</a> ⓘ us-east-1f (use1-az5)	November 4, 2024, 10:44 (UTC+01:00)
		<a href="#">subnet-0160f4124a02faa4d</a> ⓘ us-east-1d (use1-az1)	
		<a href="#">subnet-0c0ae3c947e887dfa</a> ⓘ us-east-1a (use1-az2)	
		<a href="#">subnet-078422f599611ec82</a> ⓘ us-east-1b (use1-az4)	
		<a href="#">subnet-034774c6dc3f120e1</a> ⓘ us-east-1c (use1-az6)	
Load balancer ARN		DNS name <a href="#">Info</a>	
arn:aws:elasticloadbalancing:us-east-1:852657213420:loadbalancer/app/LabELB/2c3		LabELB-1882714066.us-east-1.elb.amazonaws.com (A Record)	

# Database

## Task 1: Create an RDS PostgreSQL database

RDS > **Create DB**

- kies standaard DB
- PostgreSQL
- For this exercise, use the "Free Tier" template
- Settings> Master Username → kies een username en write down the hostname to use later.
- kies "self managed" en stel een root password voor the database in schrijf het op! (**toorPostgreSQL**)
- Make sure it can connect to EC2 instances, especially the one in the "Labo5DebugVPC" (staat onder connectivity)
  - duid "Connect to an EC2 compute resource" aan
  - vul de EC2 instance in in de drop zone
- **Once created**

## Task 2: Create a db connection in pgadmin4

- Lookup the public IP of the EC2 in the Labo5DebugVPC.
- Open the IP in your browser **HTTP! Doe s weg!**

log in to the pgadmin4 app with following credentials:

- User email: student@pxl.be
- Password: SuperSecret
- "add new server" and use the hostname, login&password you write down earlier for the database.



Register - Server

General

Connection

Parameters

SSH Tunnel

Advanced

Name

test

Server group

Servers

Background

☐

Foreground

☐

Register - Server

General

Connection

Parameters

SSH Tunnel

Advanced

Host name/address

database-1.c1aqo6uos14s.us-east-1.rds.amazonaws.com

Port

5432

Maintenance database

postgres

Username

postgres

Kerberos authentication?

☐

Password

.....

Save password?

☐

Role

# Lambda functies

## Algemene kennis lambda

lambda > **create function**

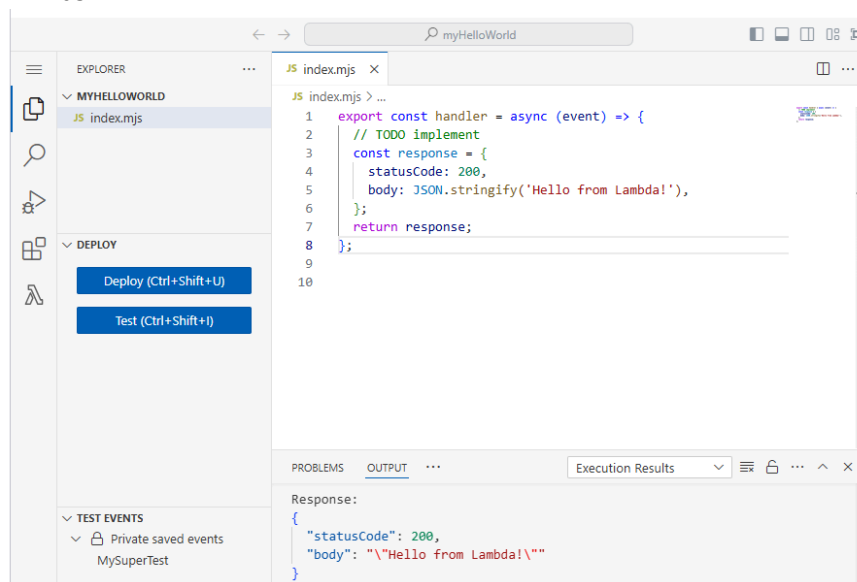
- Another from scratch
- Geef een functienaam
- kies je taal in dit geval runtime Node.js 22x
- architecture > x86\_64

Change default execution role

- use an existing role > **LabRole**

**Create function**

Je krijgt deze scherm

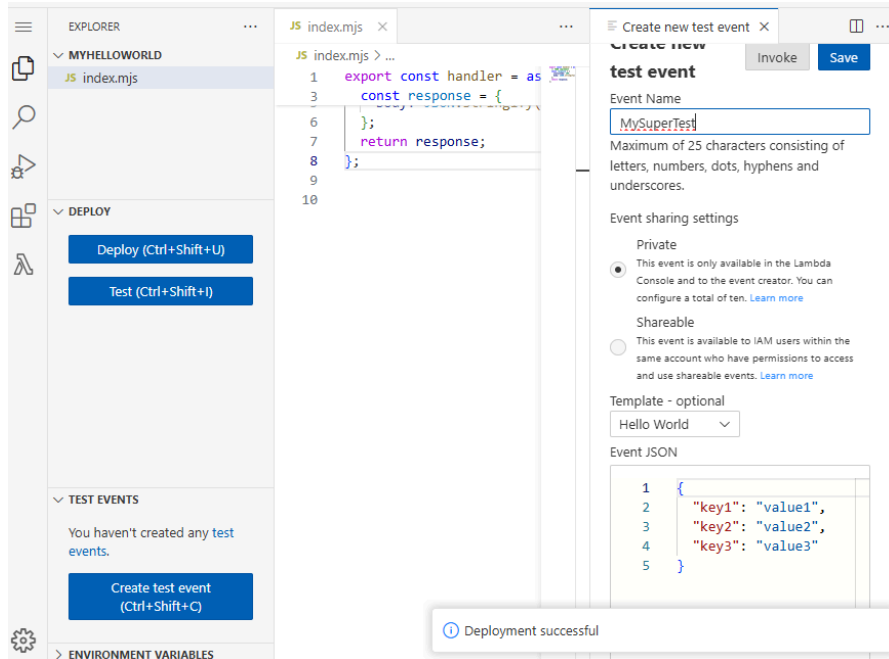


Tip om zwart balkje weg te halen in code > **esc**

- Schrijf je code
- **Druk op deploy**

Wil je testen

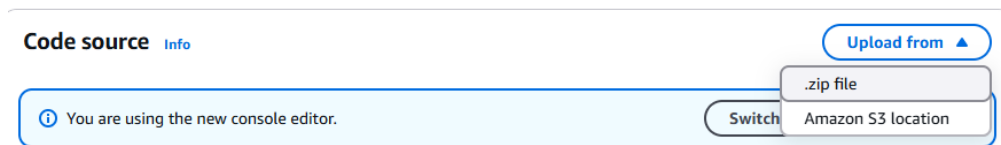
- Druk op Create test event of op + (onder events)
- zie een rechterscherf > geef naam van test en selecteer een template (s3 put, ...)
- Druk op **save**



- Druk op **test** onder deploy om te testen of het gerund heeft

#### Handige tips

- monitor > kun je gegevens analyseren
- monitor > view cloudwatch logs
- bovenaan vind je een **tab upload** from. Je kan een zip bestand uploaden (doen we niet kan wel)



#### Exercise 1

- maak een lambda event aan (**zie algemene info**)
- maak een test en kies template hello world
- plaats jouw json request erin

```
{
  "name": "Richelle"
}
```

- voeg een lambda-event aan toe in je code. Voor gemak maak een **try-catch**

In de oefening wordt gevraagd om een event op te halen. Door het event te verwerken en een variabele aan te maken, kun je de JSON-body uitlezen. Vervolgens maak je een response waarin je een bericht opneemt dat je wilt versturen. Dit zorgt ervoor dat er een JSON-response wordt teruggestuurd.

```
export const handler = async (event) => {
  try {
    const requestBody = event;
    console.log("Received request:", requestBody);
    const response = {
      message: `Hello, ${requestBody.name}!`,
    };

    return response;
  } catch (err) {
    console.error("Error parsing request body:", err);
    return {
      message: "Invalid request format",
    };
  }
};
```

- maak een deploy en test

```
Status: Succeeded
Test Event Name: test

Response:
{
  "message": "Hello, Richelle!"
}
```

## Exercise 2

- Maak een S3 bucket. **Zorg dat je permissies juist zijn.** (Zie eerdere s3 bucket )
- Druk op [Add trigger](#)
- maak een trigger: kies S3, je gekozen bucket en vink rechten aan

### Add trigger

Trigger configuration [Info](#)

S3  
aws asynchronous storage

**Bucket**  
Choose or enter the ARN of an S3 bucket that serves as the event source. The bucket must be in the same region as the function.  
s3/test12101041  
Bucket region: us-east-1

**Event types**  
Select the events that you want to have trigger the Lambda function. You can optionally set up a prefix or suffix for an event. However, for each bucket, individual events cannot have multiple configurations with overlapping prefixes or suffixes that could match the same object key.  
All object create events

- Schrijf een code
  - je hoeft geen response te krijgen om het antwoord te weten of het gelogd heeft

```
export const handler = async (event) => {
  try {
    const bucketname = event.Records[0].s3.bucket.name;
    const key = event.Records[0].s3.object.key;
    const size = event.Records[0].s3.object.size;
    console.log(`Bestand geüpload: ${key}, Grootte: ${size} bytes,
Bucket: ${bucketname}`)
  } catch (err) {
    console.log('Mislukt')
  }
};
```

- maak een test van en kies s3 put om te gaan testen.
  - verander de key value in een .txt file dat maakt het overzichtelijker
- voeg een file toe in je gekozen s3 bucket

Om te zien of het gelukt is

- ga naar monitor
- Druk op [View CloudWatch logs](#)
- ga naar logs > logs groups > kies je lambda functie
- nu ga je naar log streams > de eerste log stream is de meest recente
- Druk op de eerste link

Log streams

Tags

Anomaly detection

Metric filters

Subscription filters

Contributor Insights

Data protection

Field indexes - n

Log streams (18)

Filter log streams or try prefix search

Exact match

Show expired

Info

Log stream

2024/11/27/[\$LATEST]655ba7c5ddd64d75a7d75e3494a44e51

2024-11-27 10:12:55 (UTC)

2024/11/27/[\$LATEST]956ed75b7ae448c0919aa67dde3e4415

2024-11-27 10:12:30 (UTC)

- Je vind je log gegevens daar

CloudWatch > Log groups > /aws/lambda/test > 2024/11/27/[\$LATEST]dbc576de472b454d92bcf16e5e5b274f	
Log events	
You can use the filter bar below to search for and match terms, phrases, or values in your log events. <a href="#">Learn more about filter patterns</a>	
<input type="text" value="Filter events - press enter to search"/> <input type="button" value="Clear"/> <input type="button" value="1m"/> <input type="button" value="30m"/> <input type="button" value="1h"/> <input type="button" value="12h"/> <input type="button" value="Custom"/> <input type="button" value="UTC timezone"/> <input type="button" value="Display"/>	
▶	Timestamp   Message
	No older events at this moment. <a href="#">Retry</a>
▶	2024-11-27T10:09:50.352Z INIT_START Runtime Version: nodejs:22.v29 Runtime Version ARN: arn:aws:lambda:us-east-1::runtime:f494bf5385768c1a5f722eae98b6dd3d343c96ba7ec22b34f5c819e3e8511722
▶	2024-11-27T10:09:50.505Z START RequestId: 6a06c7ca-c5fa-4e4e-af94-dc374cd3c409 Version: \$LATEST
▶	2024-11-27T10:09:50.511Z 2024-11-27T10:09:50.511Z 6a06c7ca-c5fa-4e4e-af94-dc374cd3c409 INFO Bestand geüpload: <b>bertbissber-corrupt.txt</b> , Grootte: 183 bytes, Bucket: test12101041
▶	2024-11-27T10:09:50.533Z END RequestId: 6a06c7ca-c5fa-4e4e-af94-dc374cd3c409
▶	2024-11-27T10:09:50.533Z REPORT RequestId: 6a06c7ca-c5fa-4e4e-af94-dc374cd3c409 Duration: 27.45 ms Billed Duration: 28 ms Memory Size: 128 MB Max Memory Used: 70 MB Init Duration: 150.95 ms
	No newer events at this moment. Auto retry paused. <a href="#">Resume</a>

## Exercise 3

- maak een dynamodb aan. Druk op **create tabel**
- Geef de table name Todos
- Partition key geef de naam id
- **Create table**

Maak een lambda aan

- plaats de node.js code erin

```
import { DynamoDBClient } from "@aws-sdk/client-dynamodb";
import {
  DynamoDBDocumentClient,
  PutCommand,
} from "@aws-sdk/lib-dynamodb";

const client = new DynamoDBClient({});

const dynamo = DynamoDBDocumentClient.from(client);

export const handler = async (event) => {
  const item = {
```

```

    id: event.id,
    task: event.task,
    status: event.status
  }
  await dynamo.send(new PutCommand({
    TableName: "Todos",
    Item: item
  }));
};

```

- Maak vervolgens een test. Hier plaats je put gegevens erin. Geef het een naam en druk op save

```

{
  "id": "1",
  "task": "Learn AWS Lambda",
  "status": "pending"
}

```

- Vervolgens **deploy** en **test** je dat

controle of het gewerkt heeft

- ga naar Explore items > selecteer Todos (je dynamodb)
- Je vind een id 1

The screenshot shows the AWS DynamoDB console interface. On the left, the navigation pane includes 'DynamoDB', 'Dashboard', 'Tables', 'Explore items', ' PartiQL editor', 'Backups', 'Exports to S3', 'Imports from S3', 'Integrations', 'Reserved capacity', and 'Settings'. The main area is titled 'Todos' and shows a table with one item. The item has an 'id' of '1', a 'status' of 'pending', and a 'task' of 'Learn AWS Lambda'. A green status bar indicates 'Completed. Read capacity units consumed: 2'. The table headers are 'id (String)', 'status', and 'task'.

- Druk op 1
- Daar vind je je aangemaakte tabel

## Edit item

You can add, remove, or edit the attributes of an item. You can nest attributes inside other attributes up to 32 levels deep. [Learn more](#)

Form

JSON view

**Attributes** Add new attribute ▼

Attribute name	Value	Type	
id - Partition key	1	String	
status	pending	String	<button>Remove</button>
task	Learn AWS Lambda	String	<button>Remove</button>

Cancel Save Save and close

## Exercise 4

- maak een lambda functie aan (zie algemene oefening)
- Maak een ec2 server aan (zie eerdere documentatie)
- voeg code toe aan lambda functie

```
import { DescribeInstancesCommand, EC2Client } from
"@aws-sdk/client-ec2";
export const handler = async (event) => {
  const ec2Client = new EC2Client({ region: event.region });
  const instanceId = event.detail["instance-id"];
  try {
    const command = new DescribeInstancesCommand({
      InstanceIds: [instanceId],
    });
    const response = await ec2Client.send(command);
    const publicIp =
response.Reservations[0].Instances[0].PublicIpAddress;

    console.log(`Public IP of instance ${instanceId}: ${publicIp}`);
    return {
      statusCode: 200,
      body: `Public IP logged: ${publicIp}`,
    };
  } catch (error) {
    console.error("Error fetching instance data: ", error);
    throw error;
  }
};
```



```
import { DescribeInstancesCommand, EC2Client } from
"@aws-sdk/client-ec2";
export const handler = async (event) => {
  try {
    const client = new EC2Client({ region: event.region });
    const input = { // DescribeInstancesRequest
      InstanceIds: [ // InstanceIdStringList
        event.detail["instance-id"],
      ],
    };
    const command = new DescribeInstancesCommand(input);
    const response = await client.send(command);
    const Ipadress =
response.Reservations[0].Instances[0].PublicIpAddress
    console.log(Ipadress)
    return Ipadress;
  } catch (error) {
    console.log(error)
  }
};
```

- Om te weten te komen waar je de public ip adres te vinden is  
→ `const publicIp = response.Reservations[0].Instances[0].PublicIpAddress;`
- doe het volgende

```
import { DescribeInstancesCommand, EC2Client } from
"@aws-sdk/client-ec2";

export const handler = async (event) => {
  const ec2Client = new EC2Client({ region: event.region });
  const instanceId = event.detail["instance-id"];

  try {
    const command = new DescribeInstancesCommand({
      InstanceIds: [instanceId],
    });
  }
};
```

```

const response = await ec2Client.send(command);
return {
    statusCode: 200,
    message: response
};
} catch (error) {
    console.error("Error fetching instance data: ", error);
    throw error;
}
};

```

- in de response data zie je

```

Response:
{
  "statusCode": 200,
  "message": {
    "$metadata": {
      "httpStatusCode": 200,
      "requestId": "2ba59fb4-5c75-4d39-8c67-91a905d330f2",
      "attempts": 1,
      "totalRetryDelay": 0
    },
    "Reservations": [
      {
        "Groups": [],
        "Instances": [
          {
            "AmiLaunchIndex": 0,
            "ImageId": "ami-0453ec754f44f9a4a",
            "InstanceId": "i-07b9d918db3cd6b95",
            "InstanceType": "t2.micro",
            "KeyName": "vockey",
            "LaunchTime": "2024-12-01T08:56:52.000Z",
            "Monitoring": {
              "State": "disabled"
            },
            "Placement": {

```

```
    "AvailabilityZone": "us-east-1a",
    "GroupName": "",
    "Tenancy": "default"
  },
  "PrivateDnsName": "ip-172-31-26-18.ec2.internal",
  "PrivateIpAddress": "172.31.26.18",
  "ProductCodes": [],
  "PublicDnsName": "ec2-54-85-79-58.compute-1.amazonaws.com",
  "PublicIpAddress": "54.85.79.58",
  "State": {
    "Code": 16,
    "Name": "running"
  },
  "StateTransitionReason": "",
  "SubnetId": "subnet-0f9a40ec0716a8f67",
  "VpcId": "vpc-0d0a9b2daae79adaa",
  "Architecture": "x86_64",
  "BlockDeviceMappings": [
    {
      "DeviceName": "/dev/xvda",
      "Ebs": {
        "AttachTime": "2024-12-01T08:48:11.000Z",
        "DeleteOnTermination": true,
        "Status": "attached",
        "VolumeId": "vol-04e040d149d497f7e"
      }
    }
  ],
  "ClientToken": "bf9c8a51-25d6-42b5-a952-f37badc7e1fd",
  "EbsOptimized": false,
  "EnaSupport": true,
  "Hypervisor": "xen",
  "NetworkInterfaces": [
    {
      "Association": {
        "IpOwnerId": "amazon",
        "PublicDnsName":
"ec2-54-85-79-58.compute-1.amazonaws.com",
        "PublicIp": "54.85.79.58"
      }
    }
  ]
}
```

```
    },
    "Attachment": {
      "AttachTime": "2024-12-01T08:48:11.000Z",
      "AttachmentId": "eni-attach-0365b4c3b0112bf5f",
      "DeleteOnTermination": true,
      "DeviceIndex": 0,
      "Status": "attached",
      "NetworkCardIndex": 0
    },
    "Description": "",
    "Groups": [
      {
        "GroupName": "launch-wizard-1",
        "GroupId": "sg-026b35328c0e724b4"
      }
    ],
    "Ipv6Addresses": [],
    "MacAddress": "0a:ff:c5:e3:e5:e7",
    "NetworkInterfaceId": "eni-0d841ff317ca42c33",
    "OwnerId": "765567911343",
    "PrivateDnsName": "ip-172-31-26-18.ec2.internal",
    "PrivateIpAddress": "172.31.26.18",
    "PrivateIpAddresses": [
      {
        "Association": {
          "IpOwnerId": "amazon",
          "PublicDnsName":
"ec2-54-85-79-58.compute-1.amazonaws.com",
          "PublicIp": "54.85.79.58"
        },
        "Primary": true,
        "PrivateDnsName": "ip-172-31-26-18.ec2.internal",
        "PrivateIpAddress": "172.31.26.18"
      }
    ],
    "SourceDestCheck": true,
    "Status": "in-use",
    "SubnetId": "subnet-0f9a40ec0716a8f67",
    "VpcId": "vpc-0d0a9b2daae79adaa",
```

```
        "InterfaceType": "interface"
    }
],
"RootDeviceName": "/dev/xvda",
"RootDeviceType": "ebs",
"SecurityGroups": [
    {
        "GroupName": "launch-wizard-1",
        "GroupId": "sg-026b35328c0e724b4"
    }
],
"SourceDestCheck": true,
"Tags": [
    {
        "Key": "Name",
        "Value": "testInstance"
    }
],
"VirtualizationType": "hvm",
"CpuOptions": {
    "CoreCount": 1,
    "ThreadsPerCore": 1
},
"CapacityReservationSpecification": {
    "CapacityReservationPreference": "open"
},
"HibernationOptions": {
    "Configured": false
},
"MetadataOptions": {
    "State": "applied",
    "HttpTokens": "required",
    "HttpPutResponseHopLimit": 2,
    "HttpEndpoint": "enabled",
    "HttpProtocolIpv6": "disabled",
    "InstanceMetadataTags": "disabled"
},
"EnclaveOptions": {
    "Enabled": false
}
```

```

    },
    "BootMode": "uefi-preferred",
    "PlatformDetails": "Linux/UNIX",
    "UsageOperation": "RunInstances",
    "UsageOperationUpdateTime": "2024-12-01T08:48:11.000Z",
    "PrivateDnsNameOptions": {
      "HostnameType": "ip-name",
      "EnableResourceNameDnsARecord": true,
      "EnableResourceNameDnsAAAARecord": false
    },
    "MaintenanceOptions": {
      "AutoRecovery": "default"
    },
    "CurrentInstanceBootMode": "legacy-bios"
  }
],
  "OwnerId": "765567911343",
  "ReservationId": "r-08bd77e41480be412"
}
]
}
}

```

- Hier kun je response ip adres zoeken en toevoegen

Nu ga je testen

- maak een test aan gebruik **cloudwatch**
- Verander **region** naar jouw eigen regio en pas de detail-eigenschap aan op basis van de detail-structuur in jouw lab-map. Vervolgens vervang je de **instance-id** door het ID van jouw eigen aangemaakte instance.

```

{
  "id": "cdc73f9d-aea9-11e3-9d5a-835b769c0d9c",
  "detail-type": "Scheduled Event",
  "source": "aws.events",
  "account": "123456789012",
  "time": "1970-01-01T00:00:00Z",
  "region": "us-east-1",
  "resources": [

```

```

    "arn:aws:events:us-east-1:123456789012:rule/ExampleRule"
  ],
  "detail": {
    "instance-id": "i-07b9d918db3cd6b95"
  }
}

```

- Druk op **deploy** en vervolgens **test**

Test Event Name: details

Response:

```

{
  "statusCode": 200,
  "body": "Public IP logged: 54.85.79.58"
}

```

- ga nu naar monitor > cloudwatch logs > logs > log groups > log streams
- Je vindt daar je logs. **Ik weet niet waarom (moeten vragen) maar je laatste testen worden bij de eerste logs toegevoegd**

No older events at this moment. [Retry](#)

▶ 2024-12-01T08:54:24.462Z	INIT_START Runtime Version: nodejs:22.v29 Runtime Version ARN: arn:aws:lambda:us-east-1::runtime:f494bf5385768c1a5f722eae98b6dd3d343c96ba7ec22b34f5c819e3e8511722
▶ 2024-12-01T08:54:25.332Z	START RequestId: 7de11573-983f-4748-8a08-560af41c89eb Version: \$LATEST
▶ 2024-12-01T08:54:26.719Z	2024-12-01T08:54:26.719Z 7de11573-983f-4748-8a08-560af41c89eb INFO Public IP of instance i-07b9d918db3cd6b95: 54.234.253.205
▶ 2024-12-01T08:54:26.743Z	END RequestId: 7de11573-983f-4748-8a08-560af41c89eb
▶ 2024-12-01T08:54:26.743Z	REPORT RequestId: 7de11573-983f-4748-8a08-560af41c89eb Duration: 1408.92 ms Billed Duration: 1409 ms Memory Size: 128 MB Max Memory Used: 123 MB Init Duration: 866.81 ms
▶ 2024-12-01T08:58:19.320Z	START RequestId: 3cea7836-1aad-44b9-9ee5-5a408c51d04e Version: \$LATEST
▶ 2024-12-01T08:58:20.119Z	2024-12-01T08:58:20.119Z 3cea7836-1aad-44b9-9ee5-5a408c51d04e INFO Public IP of instance i-07b9d918db3cd6b95: 54.85.79.58
▶ 2024-12-01T08:58:20.162Z	END RequestId: 3cea7836-1aad-44b9-9ee5-5a408c51d04e

Nu wil je dat de ec2 instance **automatisch** worden gelogd en niet via testen

- Serve nu naar Amazon EventBridge
- Druk op **create rule**

- Kies een naam en description. In dit geval
  - naam: EC2RunningStateLambdaTrigger
  - description: EC2 Instance State-change Notification
- Druk op **next**
- laat alles staan in event source en sample event
- Maak hetzelfde event pattern aan als wat er in de lab staat

**Event pattern** Info

**Event source**  
AWS service or EventBridge partner as source  

AWS services

**AWS service**  
The name of the AWS service as the event source  

EC2

**Event type**  
The type of events as the source of the matching pattern  

EC2 Instance State-change Notification

**Event Type Specification 1**  
☐ Any state  
☒ Specific state(s)  

Specific state(s)

running

**Event Type Specification 2**  
☒ Any instance  
☐ Specific instance Id(s)

**Event pattern**  
Event pattern, or filter to match the events  

```

1 {
2   "source": ["aws.ec2"],
3   "detail-type": ["EC2 Instance State-change Notification"],
4   "detail": {
5     "state": ["running"]
6   }
7 }
        
```

Copy

Test pattern

Edit pattern

- Druk op **next**
- target selecteer lambda functie en naam van function bijvoorbeeld test

## Target 1

### Target types

Select an EventBridge event bus, EventBridge API destination (SaaS partner), or another AWS service as a target.

- ☐ EventBridge event bus  
☐ EventBridge API destination  
☒ AWS service

### Select a target Info

Select target(s) to invoke when an event matches your event pattern or when schedule is triggered (limit of 5 targets per rule)

Lambda function

### Function

test

► **Configure version/alias**

► **Additional settings**

- Druk de hele tijd op **next** en daarna op **create rule**
- stop nu je instance en herstart het. Ga vervolgens naar logstream. neem de eerste stream. Zonder te testen krijg je vind je je instance log al staan

Timestamp	Message
	No older events at this moment. <a href="#">Retry</a>
2024-12-01T09:47:43.982Z	INIT_START Runtime Version: nodejs:22.v29 Runtime Version ARN: arn:aws:lambda:us-east-1::runtime:f494bf5385768c1a5f722eae90b6dd3d343c96ba7ec22b34f5c819e3e8511722
2024-12-01T09:47:44.860Z	START RequestId: fcd1da5f-2777-40fa-a084-e92edf9feb95 Version: \$LATEST
2024-12-01T09:47:46.580Z	2024-12-01T09:47:46.580Z fcd1da5f-2777-40fa-a084-e92edf9feb95 INFO Public IP of instance i-07b9d918db3cd6b95: 44.223.69.118
2024-12-01T09:47:46.603Z	END RequestId: fcd1da5f-2777-40fa-a084-e92edf9feb95
2024-12-01T09:47:46.603Z	REPORT RequestId: fcd1da5f-2777-40fa-a084-e92edf9feb95 Duration: 1742.36 ms Billed Duration: 1743 ms Memory Size: 128 MB Max Memory Used: 126 MB Init Duration: 863.93 ms
	No newer events at this moment. Auto retry paused. <a href="#">Resume</a>



# Api Gateway

## Exercise 1

- create api
- HTTP API > build
- Geef een api naam en druk op next
- druk overal op next en vervolgens create
- develop > routes > create
- neem GET /

Maak een lambda zie **eerdere documentatie**

```
export const handler = async (event) => {  
  // TODO implement  
  const response = {  
    statusCode: 200,  
    body: JSON.stringify("Hello, World!")  
  };  
  return response;  
};
```

- ga naar develop > integration en druk op get
- neem create and attach an integration
- Kies lambda en neem jouw gemaakte lambda functie
- druk op create

API Gateway > APIs > hello (bmb884i840) > Integrations

**API Gateway**

- APIs
- Custom domain names
- Domain name access associations
- VPC links

---

API: hello(bmb884i840)

**Develop**

- Routes
- Authorization
- Integrations**
- CORS
- Reimport
- Export

**Deploy**

- Stages

### Create an integration

**Attach this integration to a route**

**Integration target**

Integration type

Lambda function

**Integration details**

**Integration target**

Choose the Lambda function that API Gateway invokes when the route receives a request.

**AWS Region**

us-east-1

**Lambda function**

▼ Advanced settings

- ga naar deploy > stages
- druk op [create](#)
- geef stage naam dev en enable stage development
- Druk op Deploy

GET  [Send](#)

Params **Authorization** Headers (8) Body • Scripts Settings [Cookies](#)

**Auth Type**

Inherit auth from parent

The authorization header will be automatically generated when you send the request. Learn more about [authorization](#).

This request is not inheriting any authorization helper at the moment. Save it in a collection to use the parent's authorization helper.

**Body** Cookies Headers (5) Test Results **200 OK** • 363 ms • 185 B •

Pretty Raw Preview Visualize Text

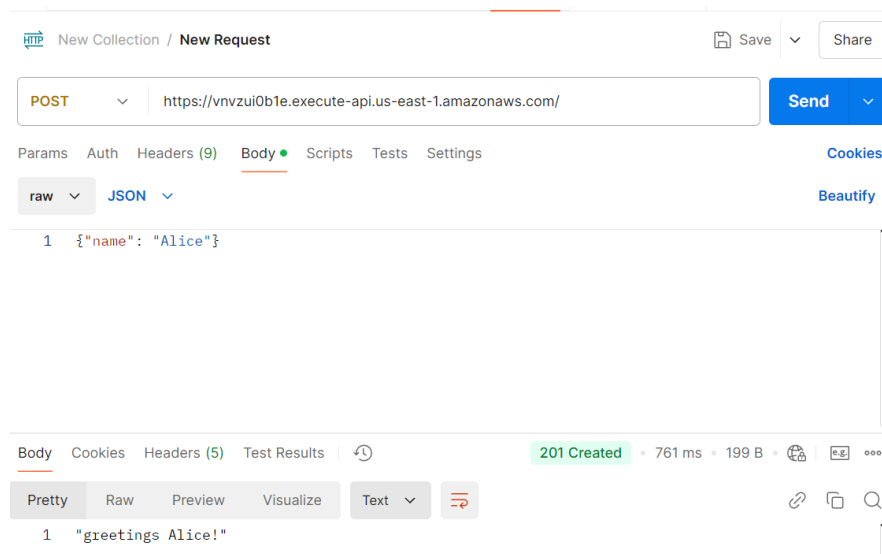
```
1 "Hello world!"
```

## Exercise 2

- Druk op **build** http API
- Geef een naam en druk de hele tijd op **next** en dan **create**
- ga naar route > **create**
- neem een **post** en geef eventueel een naam in dit geval laten we gewoon /
- Druk op **create**
  
- Maak een lambda functie

```
export const handler = async (event) => {  
  let body = JSON.parse(event.body);  
  const name = body.name || "World";  
  
  const response = {  
    statusCode: 201,  
    body: JSON.stringify(`greetings ${name}!`),  
  };  
  return response;  
};
```

- ga naar integrations > post > **create and attach an integration**
- kies lambda function > en vervolgens een naam
- Druk op **create**
- ga naar stages > neem default link >  
<https://vvnvzui0b1e.execute-api.us-east-1.amazonaws.com>



## Exercise 3

### Step 1: Create a DynamoDB table

- Kies **Create table**.
- Vul de Tabel naam in, vul http-crud-tutorial-items in.
- Voor Partition key, vul "id" in.
- Choose **Create table**.

### Step 2: Create a Lambda function

- Kies **Create function**.
- Voor Function name, voer in: http-crud-tutorial-function
- Voor Runtime, kies de nieuwste ondersteunde versie van Node.js of Python.
- Onder Permissions selecteer je Change default execution role.
- Klik op **Create function**.
- Open de Lambda-functie in de code-editor van de console en vervang de inhoud met de volgende code. Kies vervolgens **Deploy** om je functie bij te werken.

```
import { DynamoDBClient } from "@aws-sdk/client-dynamodb";
import {
  DynamoDBDocumentClient,
  ScanCommand,
  PutCommand,
  GetCommand,
  DeleteCommand,
} from "@aws-sdk/lib-dynamodb";

const client = new DynamoDBClient({});

const dynamo = DynamoDBDocumentClient.from(client);

const tableName = "http-crud-tutorial-items";

export const handler = async (event, context) => {
  let body;
  let statusCode = 200;
  const headers = {
    "Content-Type": "application/json",
  };

  try {
    switch (event.routeKey) {
```

```

case "DELETE /items/{id}":
    await dynamo.send(
        new DeleteCommand({
            TableName: tableName,
            Key: {
                id: event.pathParameters.id,
            },
        })
    );
    body = `Deleted item ${event.pathParameters.id}`;
    break;
case "GET /items/{id}":
    body = await dynamo.send(
        new GetCommand({
            TableName: tableName,
            Key: {
                id: event.pathParameters.id,
            },
        })
    );
    body = body.Item;
    break;
case "GET /items":
    body = await dynamo.send(
        new ScanCommand({ TableName: tableName })
    );
    body = body.Items;
    break;
case "PUT /items":
    let requestJSON = JSON.parse(event.body);
    await dynamo.send(
        new PutCommand({
            TableName: tableName,
            Item: {
                id: requestJSON.id,
                price: requestJSON.price,
                name: requestJSON.name,
            },
        })
    );

```

```

    );
    body = `Put item ${requestJSON.id}`;
    break;
  default:
    throw new Error(`Unsupported route: "${event.routeKey}"`);
  }
} catch (err) {
  statusCode = 400;
  body = err.message;
} finally {
  body = JSON.stringify(body);
}

return {
  statusCode,
  body,
  headers,
};
};

```

Stap 3: Een HTTP API maken:

- Kies **API create** en selecteer vervolgens HTTP API en klik op **build**.
- Voer bij API-naam de naam http-crud-tutorial-api in.
- Klik op **next**.
- Bij Routes configureren, klik op **next** om het maken van routes over te slaan. Je maakt de routes later.
- Bekijk het stadium dat API Gateway automatisch voor je aanmaakt en klik vervolgens op **next**.
- Klik op **create**.

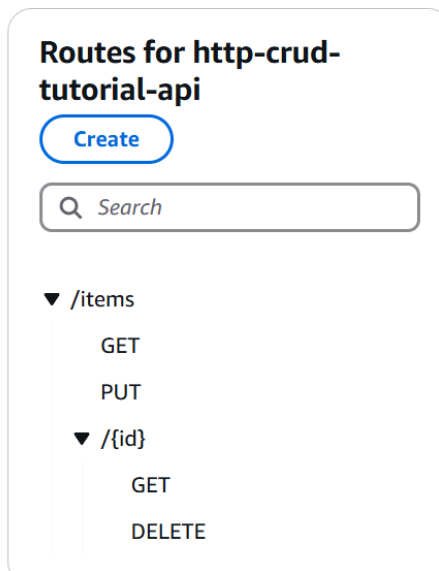
stap 4 Routes maken

### voorbeeld routes

Om routes te maken:

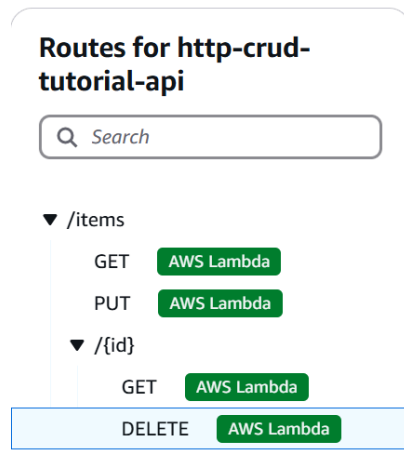
- Kies je API.

- Kies Routes.
- Klik op [create](#).
- Voor Methode, selecteer GET.
- Voor het pad voer je `/items/{id}` in.
- Klik op [create](#).
- Herhaal stappen 4-7 voor de volgende routes:
  - GET /items
  - DELETE /items/{id}
  - PUT /items.



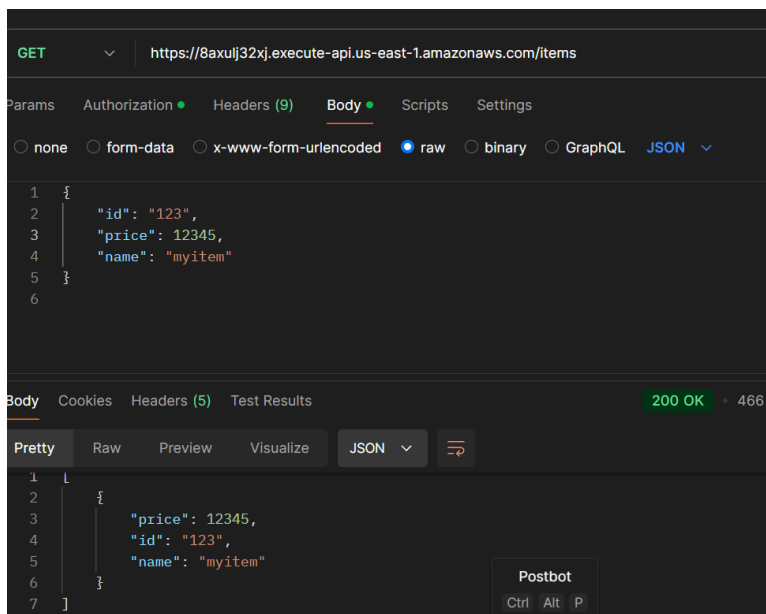
stap 5: Een integratie maken:

- Kies je API.
- Kies Integrations.
- Kies Manage integrations en vervolgens [Create](#).
- Sla Attach this integration to a route over. Dit wordt in een latere stap voltooid.
- Voor Integration type, selecteer Lambda function.
- Voor Lambda function, voer in: http-crud-tutorial-function.
- [Kies Create](#).
- Herhaal stap 4-5 met alle api's



## stap 6: testen

- ga naar stages en vervolgens neem je default url
- Open postman om vervolgens te testen
- let op neem de juiste CRUD en de juiste url bijvoorbeeld /items





Dashboard
Tables
Explore items
 PartiQL editor
 Backups
 Exports to S3
 Imports from S3
 Integrations [New](#)
 Reserved capacity
 Settings

Any tag value

Find tables

1 match

http-crud-tutorial-items

► Scan or query items

Expand to query or scan items.

Completed. Read capacity units consumed: 2

Items returned (2)

Actions

Create item

< 1 >

<input type="checkbox"/>	id (String)	name	price
<input type="checkbox"/>	123	myitem	12345
<input type="checkbox"/>	124	pizza	12344

## Exercise 4

- Maak een 2 lambda aan

Get request	Post request
<pre>import { DynamoDBClient } from "@aws-sdk/client-dynamodb"; import { DynamoDBDocumentClient, ScanCommand } from "@aws-sdk/lib-dynamodb";  const client = new DynamoDBClient({}); const dynamo = DynamoDBDocumentClient.from(client) ;  const tableName = "Todos";  export const handler = async () =&gt; {   let response;   try {     const result = await dynamo.send(new ScanCommand({ TableName: tableName }));     response = {       statusCode: 200,       body: </pre>	<pre>import { DynamoDBClient } from "@aws-sdk/client-dynamodb"; import { DynamoDBDocumentClient, PutCommand } from "@aws-sdk/lib-dynamodb";  const client = new DynamoDBClient({}); const dynamo = DynamoDBDocumentClient.from(client) ;  const tableName = "Todos";  export const handler = async (event) =&gt; {   let response;   try {     const body = JSON.parse(event.body);     const item = {       id: body.id,       task: body.task,       status: body.status, </pre>

```

JSON.stringify(result.Items),
    headers: {
        "Content-Type":
"application/json",
"Access-Control-Allow-Origin": "*",
    },
};
} catch (error) {
    response = {
        statusCode: 500,
        body: JSON.stringify({
error: error.message }),
        headers: {
            "Content-Type":
"application/json",
"Access-Control-Allow-Origin": "*",
        },
    };
}
return response;
};

```

```

};

    await dynamo.send(
        new PutCommand({
            TableName:
tableName,
            Item: item,
        })
    );

    response = {
        statusCode: 201,
        body: JSON.stringify({
message: "Todo added successfully",
item }),
        headers: {
            "Content-Type":
"application/json",
"Access-Control-Allow-Origin": "*",
        },
    };
} catch (error) {
    response = {
        statusCode: 500,
        body: JSON.stringify({
error: error.message }),
        headers: {
            "Content-Type":
"application/json",
"Access-Control-Allow-Origin": "*",
        },
    };
}
return response;
};

```

- In de oefening staat dat CORS origin enabled staat. Zorg er dus voor dat je `"Access-Control-Allow-Origin": "*" ,` toevoegt in de header
- maak een dynamoDB aan (**Zie eerdere oefening**)
- Maak een api-gateway aan
- Zorg ervoor dat je in route POST /api/todo en GET /api/todo instelt
- Voeg via integration je lambda toe
- ga naar default stage kies link en test in POST

The screenshot shows a REST client interface with the following details:

- Request:**
  - Method: **POST**
  - URL: `https://vj2f9bvq6f.execute-api.us-east-1.amazonaws.com/api/todo`
  - Body (raw): 

```
1 {
2   "id": "3",
3   "task": "Learn AWS Lambda",
4   "status": "pending"
5 }
6
```
- Response:**
  - Status: **201 Created**
  - Time: **2.02 s**
  - Size: **305 B**
  - Body (Pretty): 

```
1 {
2   "message": "Todo added successfully",
3   "item": {
4     "id": "3",
5     "task": "Learn AWS Lambda",
6     "status": "pending"
7   }
8 }
```

GET https://vj2f9bvq6f.execute-api.us-east-1.amazonaws.com/api/todo

Params Authorization Headers (9) Body Scripts Tests Settings

☐ none ☐ form-data ☐ x-www-form-urlencoded ☒ raw ☐ binary ☐ GraphQL .JSON

1  
2

Body Cookies Headers (6) Test Results 200 OK

Pretty Raw Preview Visualize JSON

```
1  [
2    {
3      "id": "2",
4      "task": "Learn AWS Lambda",
5      "status": "pending"
6    },
7    {
8      "id": "1"
9    },
10   {
11     "id": "4",
12     "task": "Learn AWS Lambda",
13     "status": "pending"
14   }
15 ]
```