



CheeseHead Hosting

Cheesehead hosting

Test report

Table of Contents

Introduction	2
Website functionality.....	3
Visiting the website over DNS.....	3
Registering a new user.....	4
Logging in	5
Deploy website environment.....	6
Ticket portal	7
Terraform	10
VPC overview Private	10
VPC overview Public.....	10
Check if the subnets are created.	11
Check if the routing table is created.....	11
Check if the security groups are created.	11
Check if the ECS cluster is created.	12
Check if the EC2 instances are created.....	12
Check if the EC2 instances are registered in the cluster.....	12
Check if the EC2 template is created.	13
Check if the Autoscaler is created.....	13
Check if the loadbalancer is created.....	13
Visit the website is accessible from the DNS name.	14
Connecting to sFTP container.	15
Stress testing.....	16
Testing load balancer & auto scaling.	16
Monitoring	18
Security Measures.....	19
Conclusion.....	21

Introduction

In the test we performed we login to the website with an account that was created and specify the details of the infrastructure we want to create. After the deploy button is clicked the API was triggered and the scripts triggered. This test was successful, all the scripts did their job, and we will show the results in this document.

Website functionality

Visiting the website over DNS

To evaluate if the website was accessible over DNS I browsed to: <https://www.cheeseheadhosting.nl/>

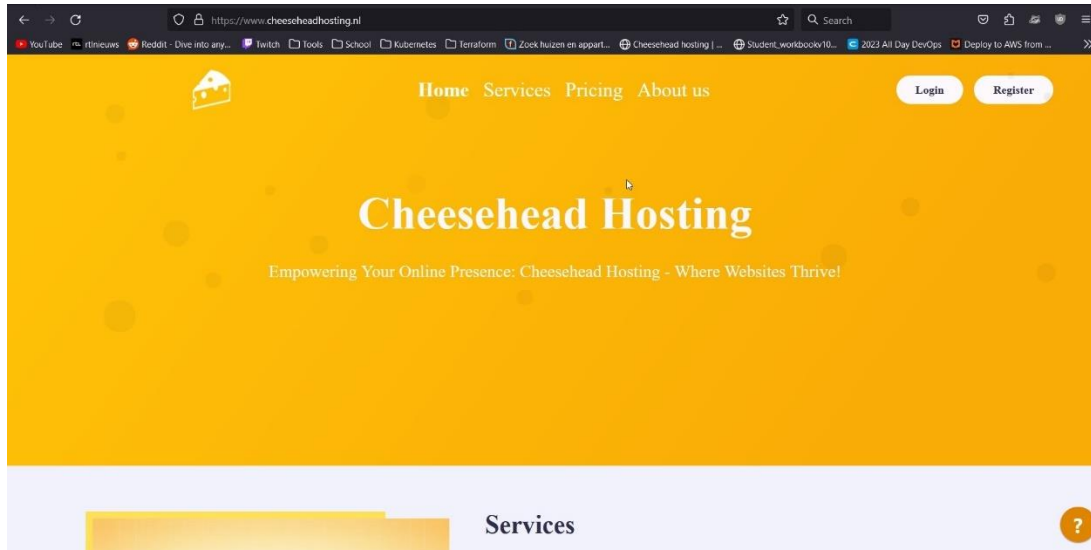


Figure 1

In figure 1 you can see that the website is accessible and responsive. The content is properly loaded, meaning that the server hosting and DNS record is working correctly. <https://www.cheeseheadhosting.nl/> is functioning correctly.

In figure 1 you could see that the website is accessible over HTTPS meaning that the certificate that is attached to the website is also working correctly. The certificate itself can be seen in figure 2.

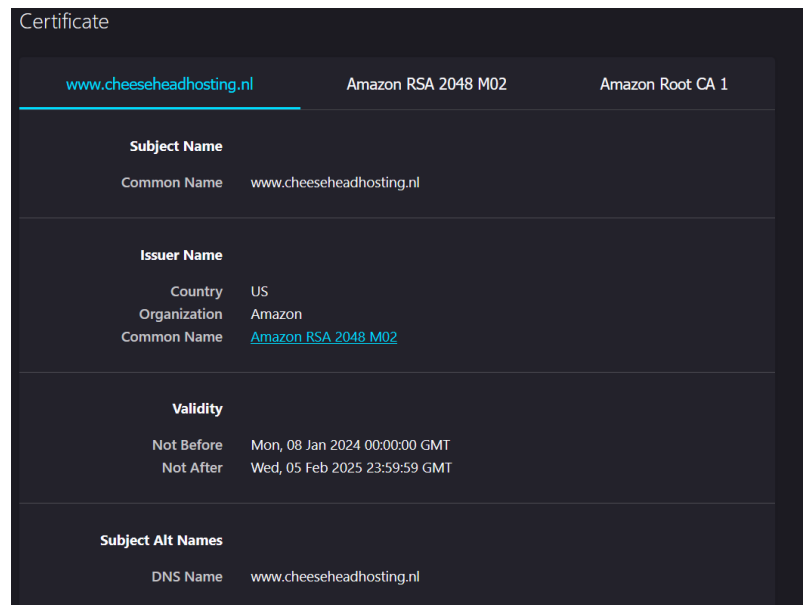


Figure 2

Registering a new user

To test if the creation of user accounts goes correctly, we visited <https://www.cheeseheadhosting.nl/register> and created an account using finalDemo as username and finalDemo@hotmail.com as email address. The Password must contain at least one uppercase letter, and one number and one special character. As can be seen in figure 3.

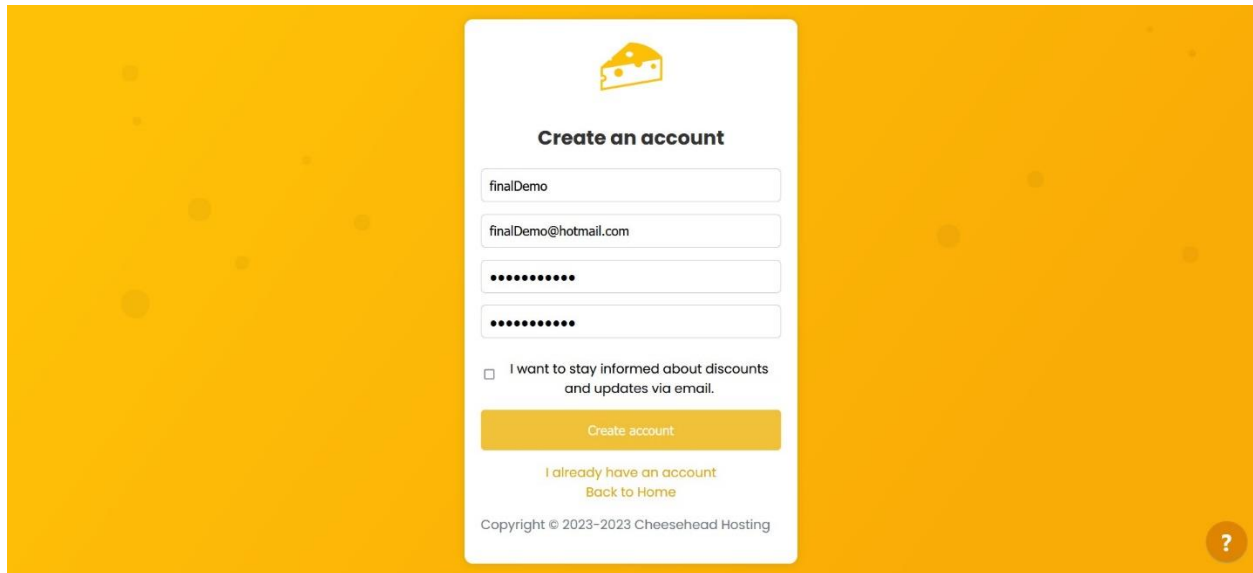


Figure 3

In figure 4 you can see the accounts table in our database. This shows that the account has successfully been created and registered in the database. As a future of security we also applied password salting and hashing.

```
mysql> SELECT * FROM accounts;
```

account_id	email	name	password
5	jellehamburg@gmail.com	jelle	\$2b\$12\$j5W1exHtUs0M4mYt08cxGuG4bG.cfD1ltWWEXJN78ux.agRSi9W5K
6	demo@gmail.com	demo	\$2b\$12\$1/3R0J3jrAs9z/Nis0eS/u80NiBQUShtKknEj9zsEWHv/MGZXrqAS
7	user@gmail.com	user	\$2b\$12\$8CJ1jMj202d8TjAeFXMTw0gCjp.5kI2QotTKvoYBU/.g/4eruKLJS
8	usermanual@gmail.com	usermanual	\$2b\$12\$6CLLEaUGjWcTcR.woEDFw09Yuy08y1siJNV49jff5NCPfbs0Dwsvp.
9	daanspronk29@gmail.com	Daan	\$2b\$12\$JTU.WINryclmBznBMnupbeXz.RCmiU8Duy/4mcoXFP/BB/V0gAyky
10	finalDemo@hotmail.com	finalDemo	\$2b\$12\$I8Emtq1dCm/ooAoj4isbGuyyistWEwDRtLxGGtAvR/Wr1Fp8BmLy2

Figure 4

Logging in

To test the login functionality we went to the login page of the website and used the credentials from the account we have made in the previous step phase, registering an account. This can be seen in figure 5.

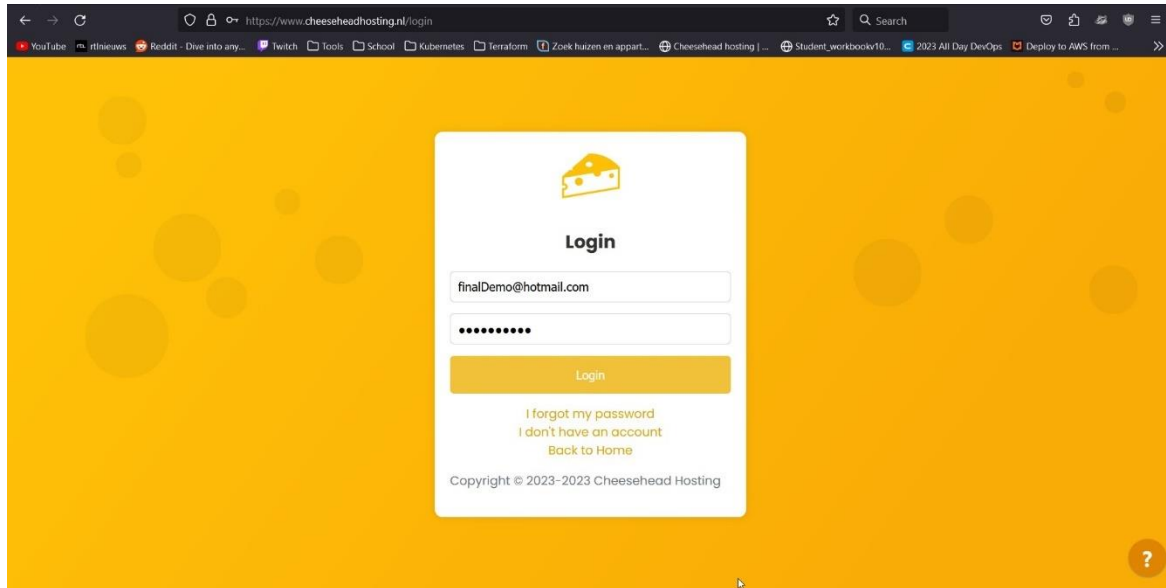


Figure 5

In figure 6 you can see that after using the correct credentials the user is send the home page of the customer portal.

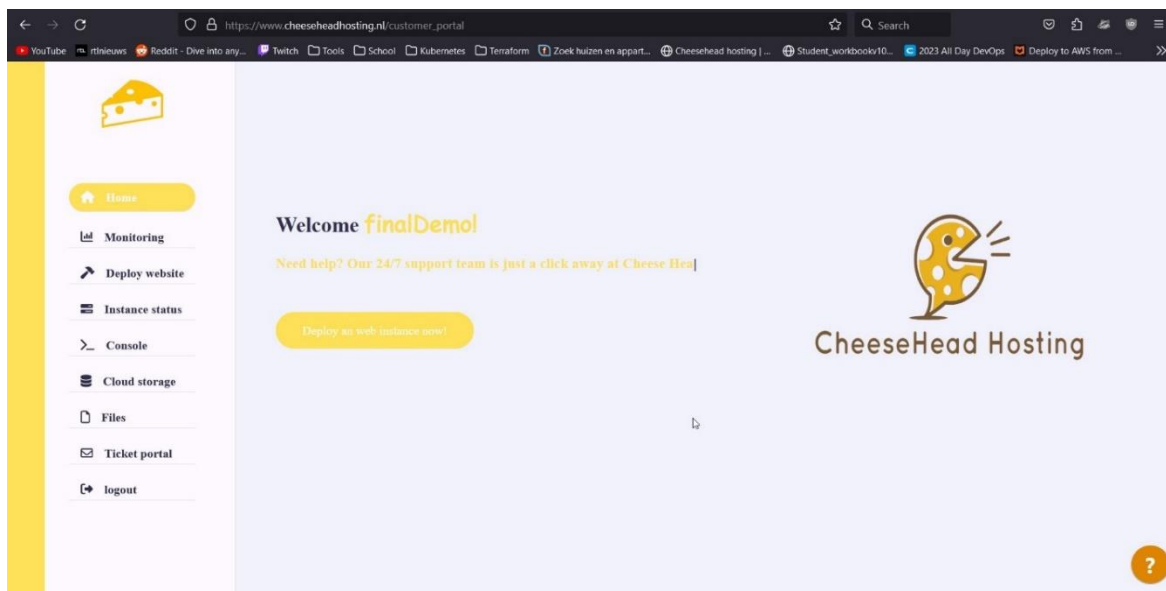


Figure 6

Deploy website environment.

In figure 7 you can see the deployment page. We have created a sample setup for the infrastructure to be deployment. We filled in the required sections and click on deploy.

Deploy Website and Database

Error: undefined

General-information

Preferred region: Ireland Redundancy: Yes Project name: finalDemo

Instance Setup

Programming Language: Python Select Database Type: MySQL Email for credentials: 508058@student.fontys.nl

Deploy website

Figure 7

In figure 8 you can see the email that we have received mentioning that the environment has been successfully deployed and thus that the website functionality is able to kick off the creation of an environment.

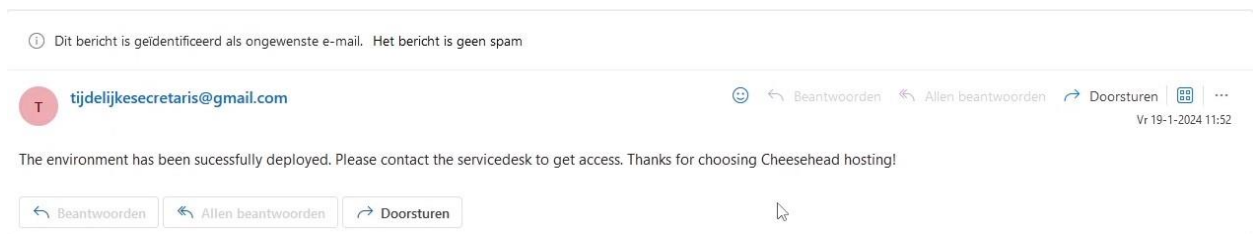


Figure 8

Ticket portal

The ticket portal is indicated with a “?” symbol on the bottom right of the screen. This is where customers can explain their issue and if there

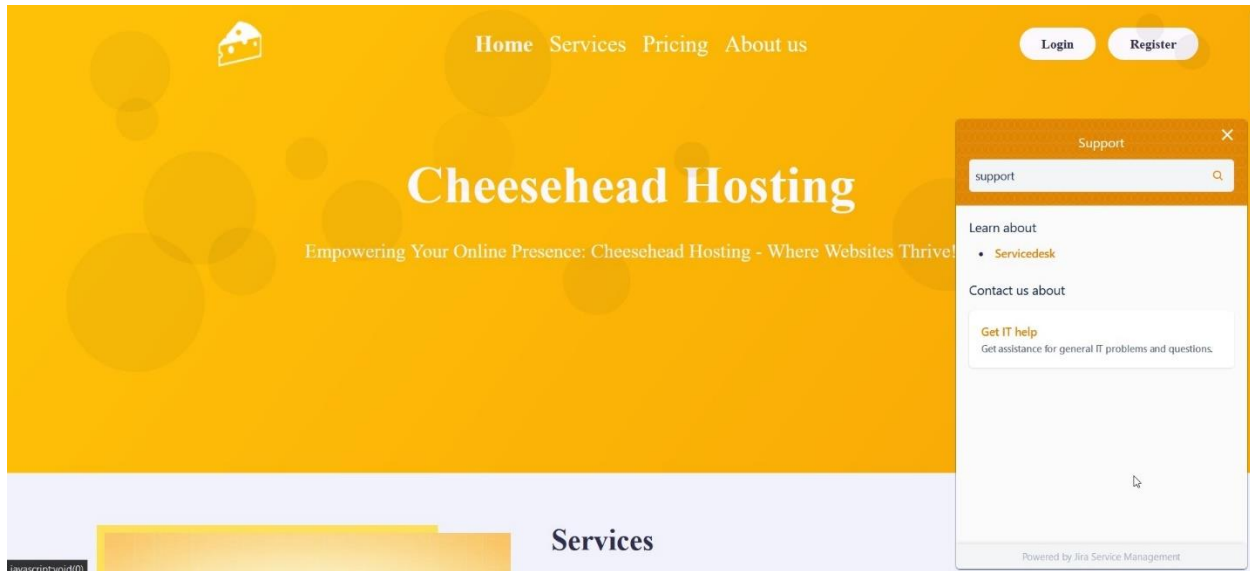
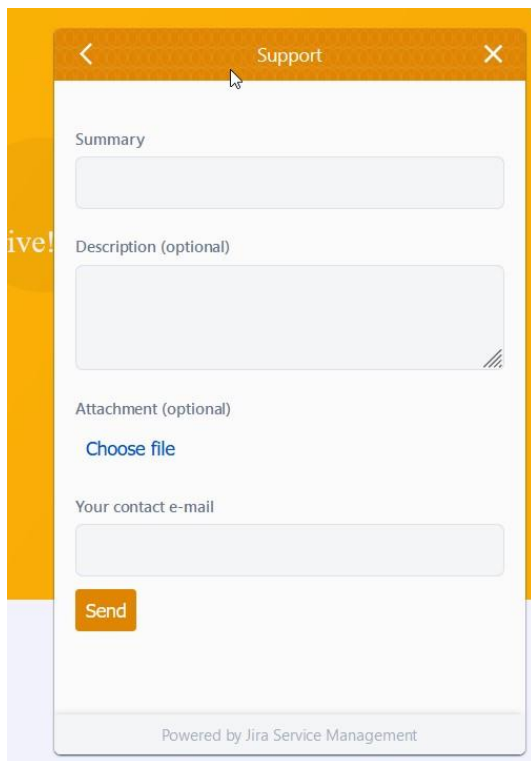


Figure 9

This figure shows the first greeting of the ticket portal where customers can either visit a service desk or get direct help from customer support by filling out a request.

A screenshot of a mobile application's 'Support' form. The form is titled 'Support' and has a back arrow and a close 'X' button in the top orange header. The form fields include: 'Summary' (a single-line text input), 'Description (optional)' (a multi-line text area with a clear icon), 'Attachment (optional)' (a section with a 'Choose file' link), and 'Your contact e-mail' (a single-line text input). An orange 'Send' button is located below the email field. At the bottom of the form, it says 'Powered by Jira Service Management'.

Support

Summary

Description (optional)

Attachment (optional)

Choose file

Your contact e-mail

Send

Powered by Jira Service Management

Figure 10

This figure shows the form the customers can fill out to request for help.



Figure 11

Once submitted, they are notified that they will be contacted about the issue soon.

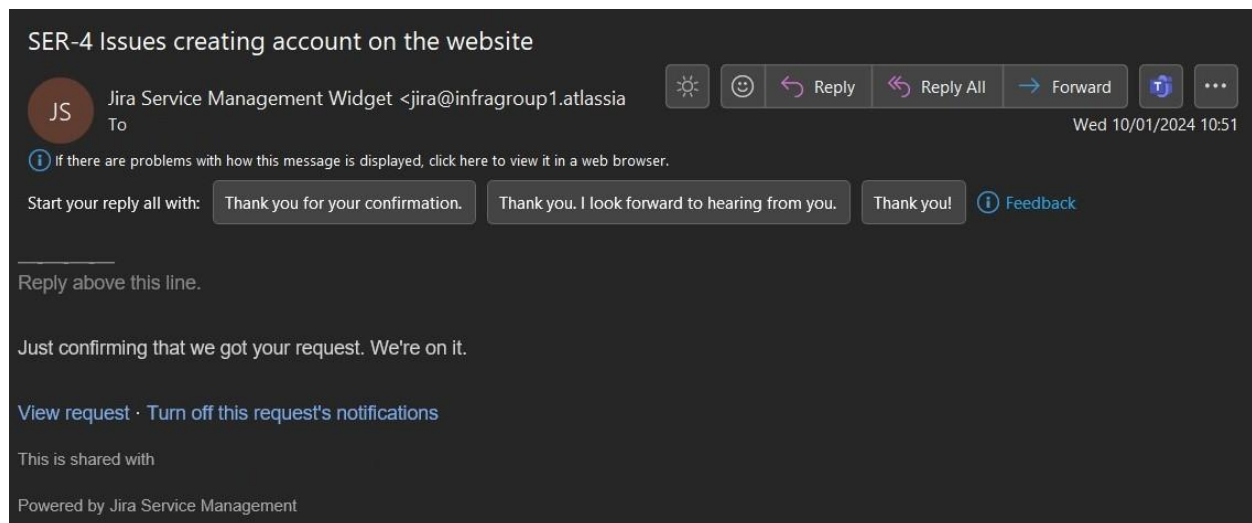


Figure 12

This figure shows our notification system once someone fills out the ticket form to receive assistance.

Terraform

Terraform successfully created the components specified in its script as will be shown below using screenshots of the final product in the AWS console. Keep in mind that “finalDemo” is the name provided as the project name on the website, this was stored in a variable and used for name convention.

VPC overview Private

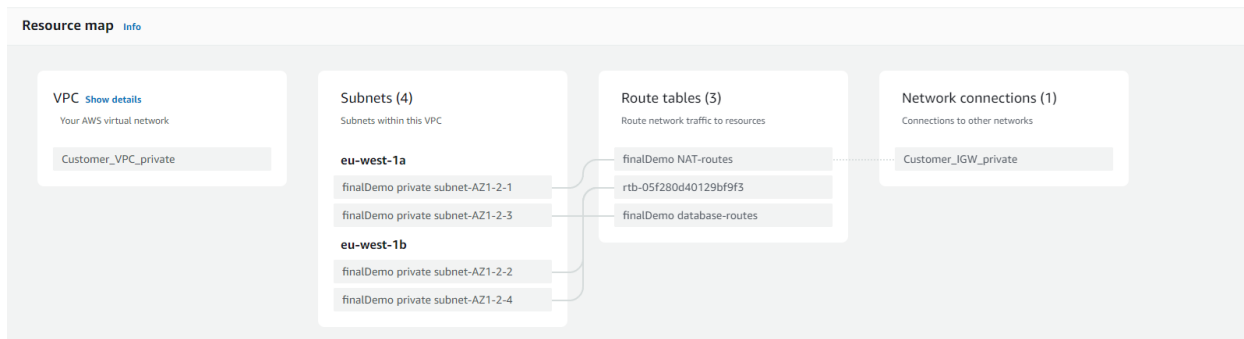


Figure 13, Resource map

In the figure above we can see the networking in the private customer VPC.

VPC overview Public

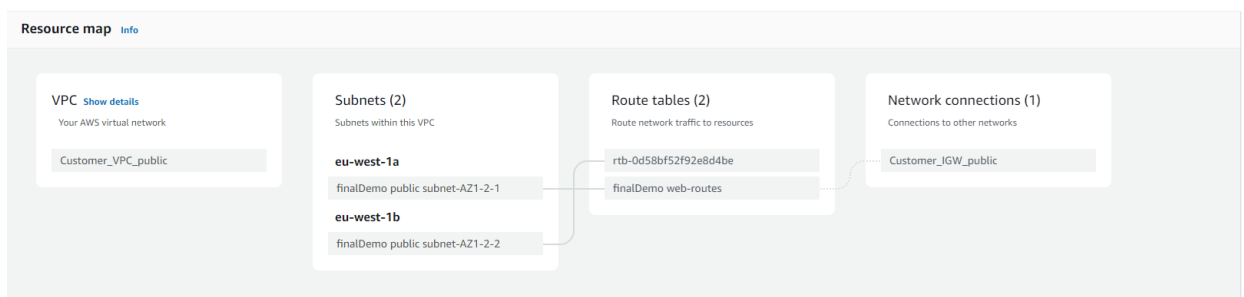


Figure 14, Resource map

In the figure above we can see the networking in the public customer VPC.

Check if the subnets are created.

	Name	Subnet ID	State	VPC	IPv4 CIDR	IPv6 CIDR	Available IPv4 addresses	Availability Zone
<input type="checkbox"/>	finalDemo private subnet-AZ1-2-2	subnet-0f7ac3ef9c3e78fc5	Available	vpc-007de754c295e6e53 Cust...	10.0.1.0/24	-	251	eu-west-1b
<input type="checkbox"/>	finalDemo private subnet-AZ1-2-4	subnet-05cbd465369944cca	Available	vpc-007de754c295e6e53 Cust...	10.0.3.0/24	-	251	eu-west-1b
<input type="checkbox"/>	finalDemo public subnet-AZ1-2-2	subnet-087d3db8f894fa4b8	Available	vpc-0f03071151601790f Cust...	10.1.1.0/24	-	249	eu-west-1b
<input type="checkbox"/>	finalDemo public subnet-AZ1-2-1	subnet-01acde546c140a321	Available	vpc-0f03071151601790f Cust...	10.1.0.0/24	-	247	eu-west-1a
<input type="checkbox"/>	finalDemo private subnet-AZ1-2-3	subnet-063f613916ab5e50e	Available	vpc-007de754c295e6e53 Cust...	10.0.2.0/24	-	251	eu-west-1a
<input type="checkbox"/>	finalDemo private subnet-AZ1-2-1	subnet-0ddc7de3a89cc865d	Available	vpc-007de754c295e6e53 Cust...	10.0.0.0/24	-	250	eu-west-1a

Figure 15, Subnet list

In the figure above we see the 6 subnets that are created for the client network.

Check if the routing table is created.

	Name	Route table ID	Explicit subnet associati...	Edge associations	Main	VPC	Owner ID
<input type="checkbox"/>	finalDemo NAT-routes	rtb-0ce15d57156ddaabf	subnet-0ddc7de3a89cc8...	-	No	vpc-007de754c295e6e53 Cust...	349962368193
<input type="checkbox"/>	finalDemo web-routes	rtb-0d5ad76d2be89fc51	subnet-01acde546c140a...	-	No	vpc-0f03071151601790f Cust...	349962368193
<input type="checkbox"/>	finalDemo database-routes	rtb-06cb3a6b15edd1095	subnet-063f613916ab5e...	-	No	vpc-007de754c295e6e53 Cust...	349962368193

Figure 16, Route tables

In this figure the client route tables are dispalyed.

Check if the security groups are created.

	Name	Security group ID	Security group name	VPC ID	Description	Owner	Inbound rule
<input type="checkbox"/>	finalDemo WEB-SG	sg-05c12b84c337f71be	web-sg	vpc-0f03071151601790f	Managed by Terraform	349962368193	7 Permission
<input type="checkbox"/>	finalDemo NAT-SG	sg-00d3922b556a7bfac	CLIENT_NAT_SG	vpc-007de754c295e6e53	Managed by Terraform	349962368193	4 Permission
<input type="checkbox"/>	finalDemo DB-SG	sg-019685952192fe214	db-sg	vpc-007de754c295e6e53	Managed by Terraform	349962368193	3 Permission

Figure 17, Security groups

The figure above show the client security groups.

Check if the ECS cluster is created.

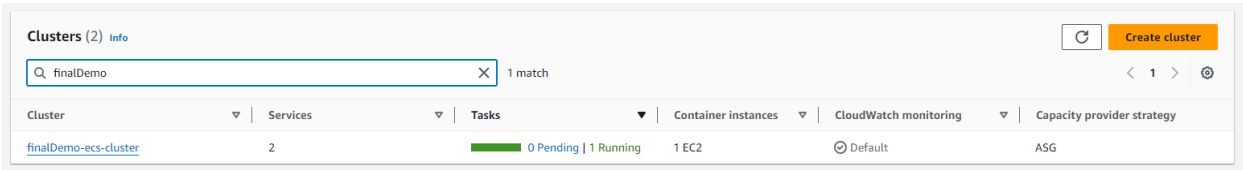


Figure 18, ECS cluster

In the image above the cluster is created.

Check if the EC2 instances are created.

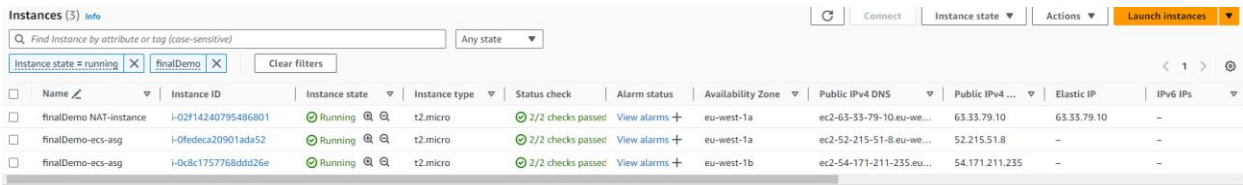


Figure 19, EC2 instances

Figure of the created EC2 instances that are created.

Check if the EC2 instances are registered in the cluster.

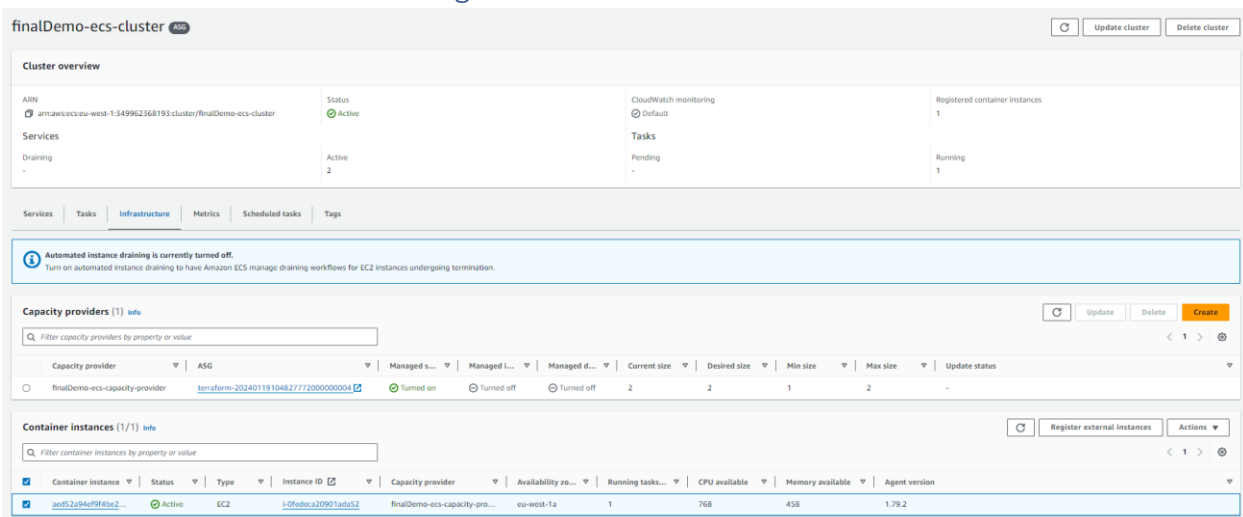
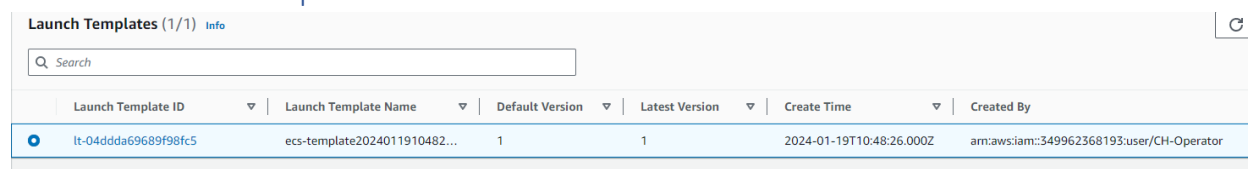


Figure 20, Container instance

One of the EC2 instances is registered to the ECS-cluster to run all the containers on.

Check if the EC2 template is created.

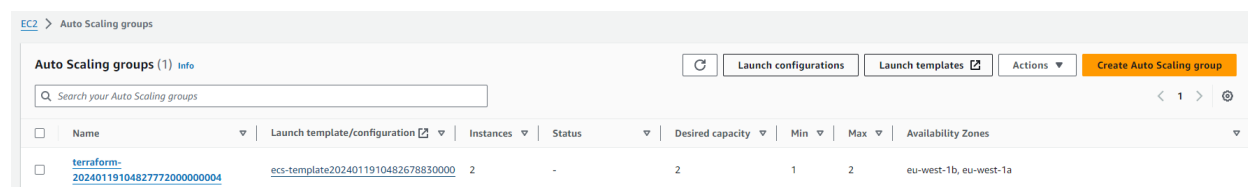


Launch Template ID	Launch Template Name	Default Version	Latest Version	Create Time	Created By
lt-04ddda69689f98fc5	ecs-template2024011910482...	1	1	2024-01-19T10:48:26.000Z	arn:aws:iam::349962368193:user/CH-Operator

Figure 21, Launch template

To spin up a EC2 instance using terraform we need to utilize a launch template that defines all the properties for the instance. This is displayed in the image above.

Check if the Autoscaler is created.

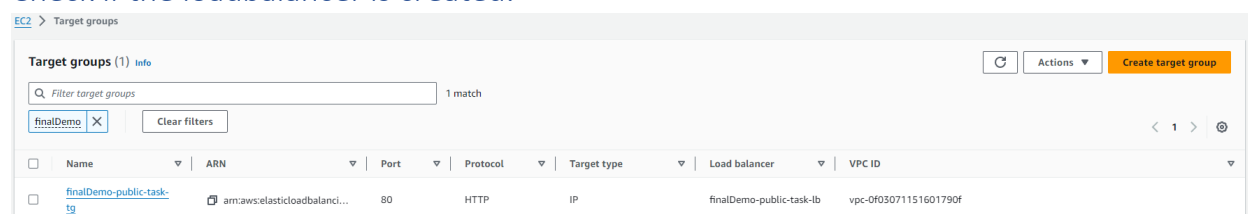


Name	Launch template/configuration	Instances	Status	Desired capacity	Min	Max	Availability Zones
terraform-20240119104827772000000004	ecs-template2024011910482678830000	2	-	2	1	2	eu-west-1b, eu-west-1a

Figure 22, Autoscaling group

In the figure above we can see that the autoscaling group is created.

Check if the loadbalancer is created.



Name	ARN	Port	Protocol	Target type	Load balancer	VPC ID
finalDemo-public-task-tg	arn:aws:elasticloadbalancing...	80	HTTP	IP	finalDemo-public-task-lb	vpc-0f03071151601790f

Figure 23, Target group

In the figure above we can see the target group for the loadbalancer is created.



Name	DNS name	State	VPC ID	Availability Zones	Type	Date created
finalDemo-public-task-lb	finalDemo-public-task-lb-1567140063.eu-west-1.elb.amazonaws.com	Active	vpc-0f030711516017...	2 Availability Zones	application	January 19, 2024, 11:48 (UTC+01:00)

Figure 24, Loadbalancer

The loadbalancer itself is also created as can be seen in the figure above.

Visit the website is accessible from the DNS name.

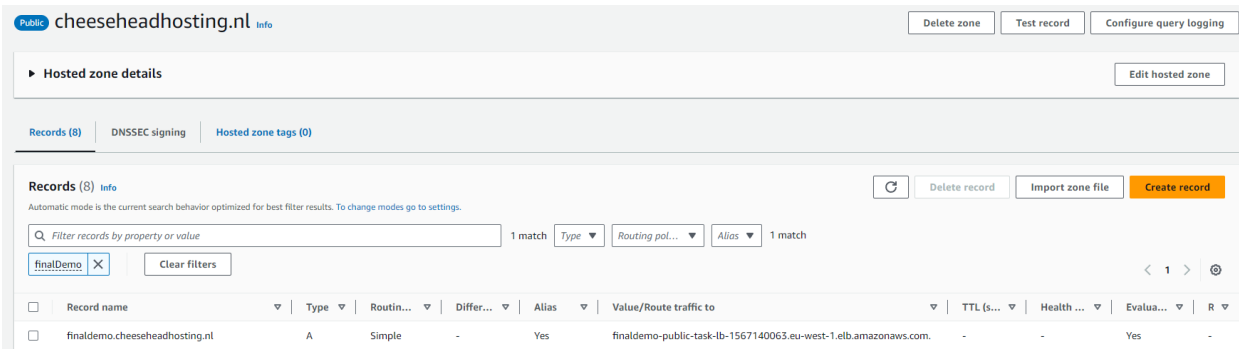


Figure 25, Hosted zone subdomain

In the figure above we can see the created subdomain for the client.



It works!

Figure 26, Client sample website

In the figure above we can see what happens when we visit the subdomain. It displays a sample website for the client.

Connecting to sFTP container.

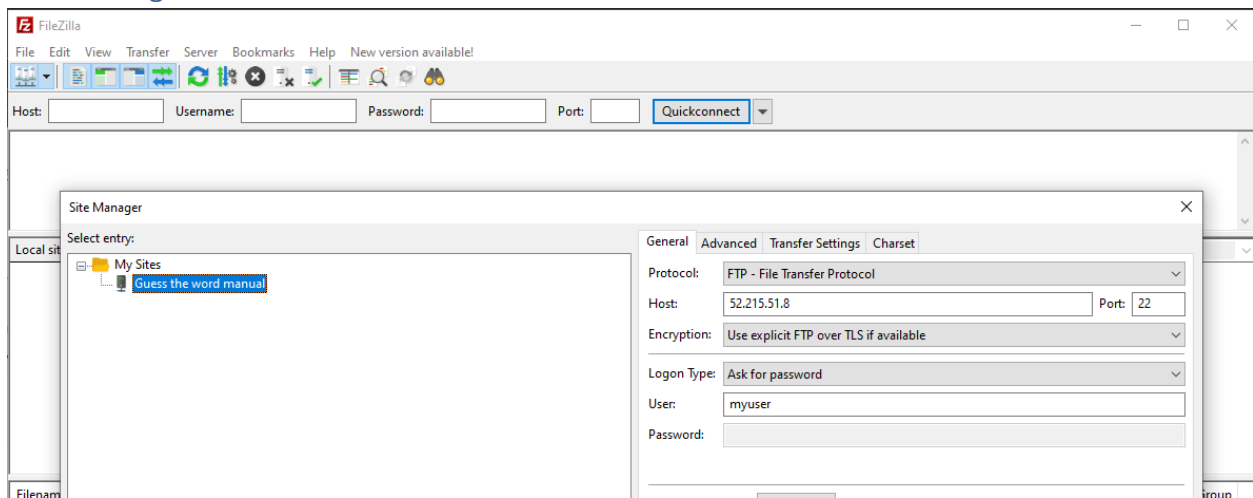


Figure 27, FTP credentials

In the figure above we see an attempt to connect to the FTP container.

```
Status: Connecting to 52.215.51.8:21...  
Status: Connection attempt failed with "ECONNREFUSED - Connection refused by server".  
Error: Could not connect to server
```

Figure 28, Failed connection

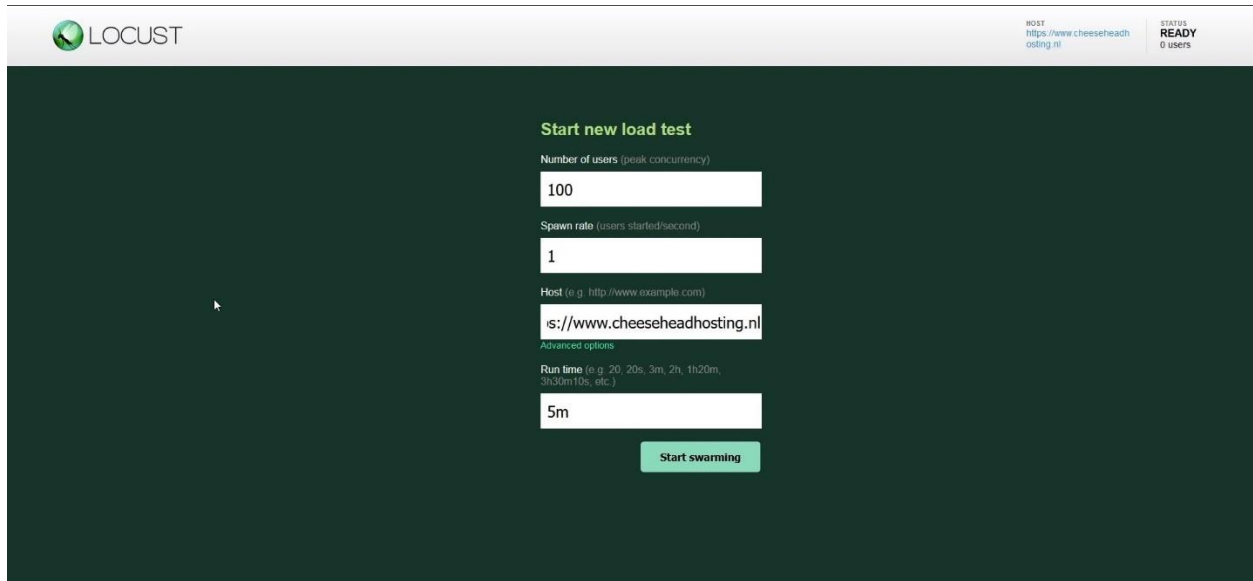
Unfortunately, we see in the figure above that the FTP connection is not successful.

This means that the Terraform script is able to create the infrastructure successfully unfortunately one component of our network doesn't work, the sFTP container.

Stress testing

Testing load balancer & auto scaling.

To test the load balancer and autoscaling functionality of our environment we have used Locust to start swarming our website. In figure 29 you can see the setup we have used. We kept things small to prevent bigger costs.



The screenshot shows the Locust web interface. At the top left is the Locust logo. At the top right, it shows the host URL 'https://www.cheeseheadhosting.nl' and the status 'READY 0 users'. The main area is titled 'Start new load test'. It contains several input fields: 'Number of users (peak concurrency)' set to 100, 'Spawn rate (users started/second)' set to 1, 'Host (e.g. http://www.example.com)' set to 'https://www.cheeseheadhosting.nl', and 'Run time (e.g. 20, 20s, 3m, 2h, 1h20m, 3h30m10s, etc.)' set to 5m. There is a 'Start swarming' button at the bottom right.

Figure 29

In figure 30 you can see that we swarmed the website for a while with clear indicators going over the 50% mark. We initially had it set to 70% but it would not scale up very often because it was harder to create the necessary load.

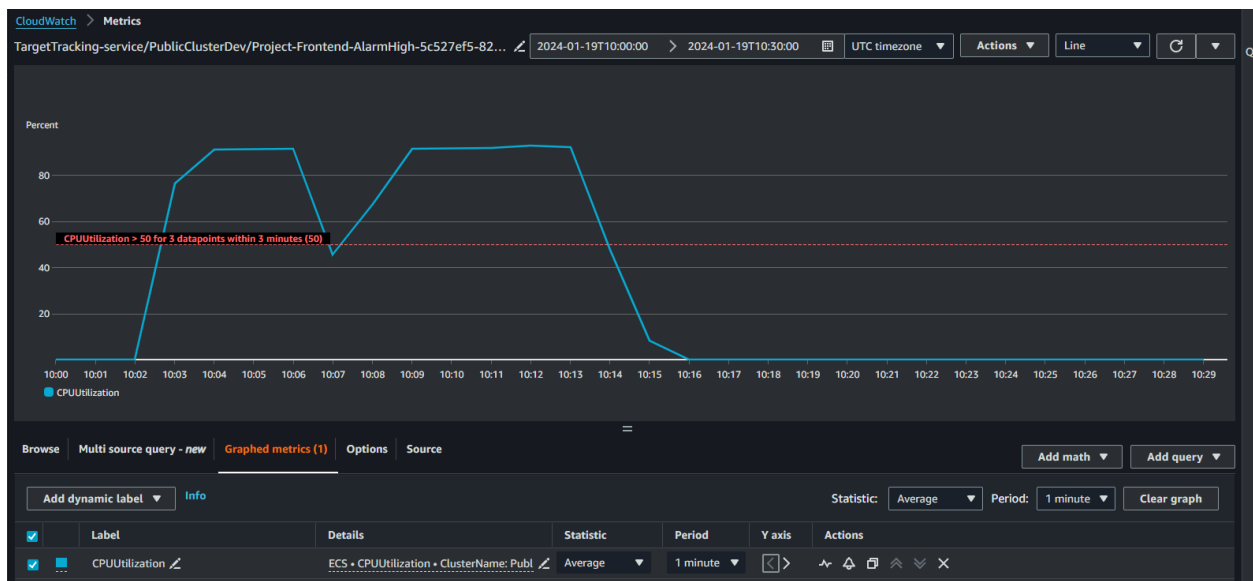


Figure 30

In figure 31 you can see that after going over the threshold a few times it is able to scale up.

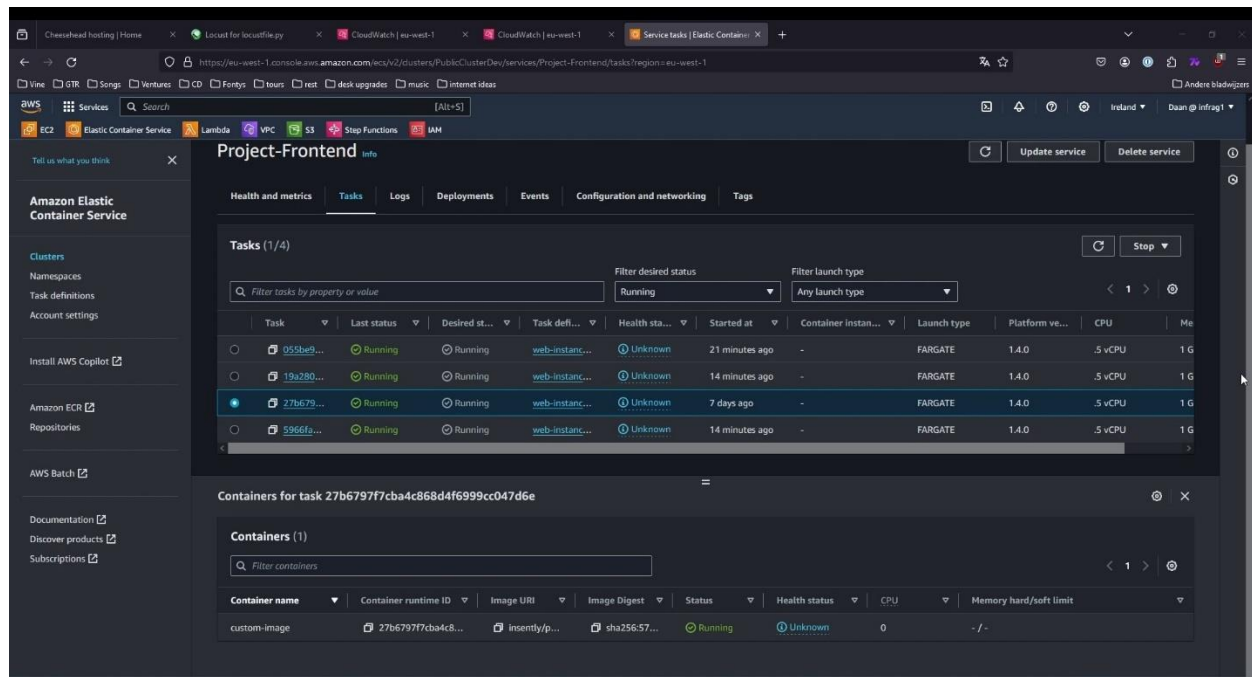


Figure 31

In figure 32 you can see that after we have ended our Locust script the auto scaler also scaled down accordingly.

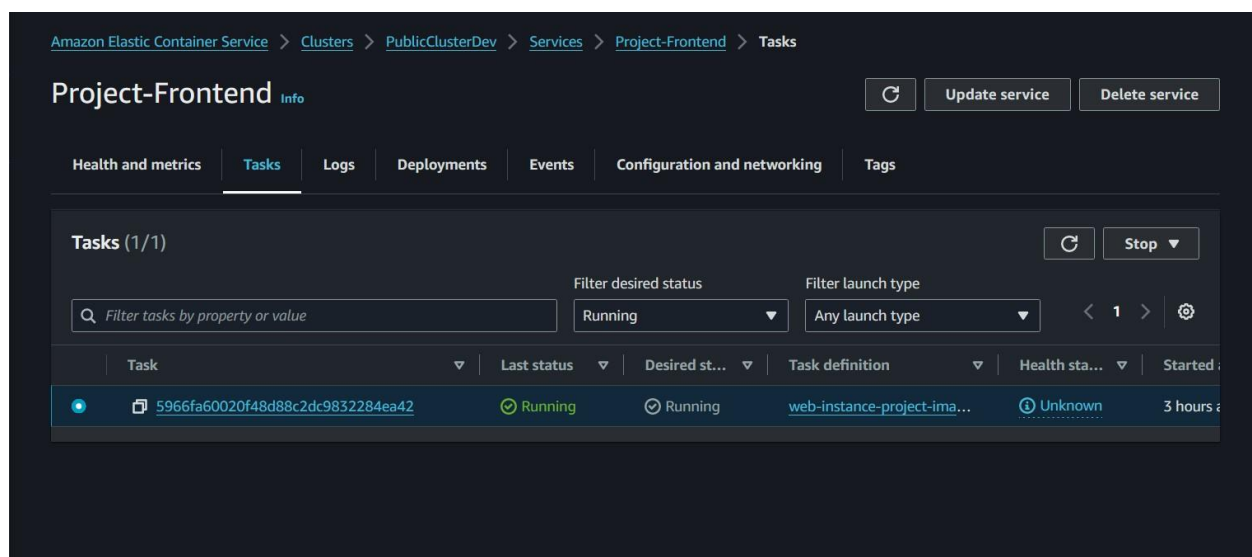


Figure 32

Monitoring

When setting the CloudTrail log group for CloudWatch, it can be seen that its data can be successfully used by CloudWatch and implemented into our Dashboard.

Log group: SNOWEEK16

#	@timestamp	@message	@logStream	@log
1	2024-01-19T12:29:28.856Z	{"eventVersion":"1.08","userIdentity":{"type":"IAMUser","principalId":"AIDAVC63... 349962368193_CloudTrail_eu-west-1_4	349962368193_CloudTrail_eu-west-1_4	349962368193_CloudTrail_eu-west-1_4
2	2024-01-19T12:29:15.629Z	{"eventVersion":"1.08","userIdentity":{"type":"IAMUser","principalId":"AIDAVC63... 349962368193_CloudTrail_eu-west-1_3	349962368193_CloudTrail_eu-west-1_3	349962368193_CloudTrail_eu-west-1_3
3	2024-01-19T12:28:46.809Z	{"eventVersion":"1.08","userIdentity":{"type":"AssumedRole","principalId":"AROA... 349962368193_CloudTrail_eu-west-1_3	349962368193_CloudTrail_eu-west-1_3	349962368193_CloudTrail_eu-west-1_3
4	2024-01-19T12:28:46.809Z	{"eventVersion":"1.08","userIdentity":{"type":"AssumedRole","principalId":"AROA... 349962368193_CloudTrail_eu-west-1_3	349962368193_CloudTrail_eu-west-1_3	349962368193_CloudTrail_eu-west-1_3
5	2024-01-19T12:28:46.121Z	{"eventVersion":"1.09","userIdentity":{"type":"IAMUser","principalId":"AIDAVC63... 349962368193_CloudTrail_eu-west-1	349962368193_CloudTrail_eu-west-1	349962368193_CloudTrail_eu-west-1
6	2024-01-19T12:28:45.514Z	{"eventVersion":"1.09","userIdentity":{"type":"IAMUser","principalId":"AIDAVC63... 349962368193_CloudTrail_eu-west-1	349962368193_CloudTrail_eu-west-1	349962368193_CloudTrail_eu-west-1
7	2024-01-19T12:28:37.587Z	{"eventVersion":"1.08","userIdentity":{"type":"AWSService","invokedBy":"dynamod... 349962368193_CloudTrail_eu-west-1_3	349962368193_CloudTrail_eu-west-1_3	349962368193_CloudTrail_eu-west-1_3
8	2024-01-19T12:28:29.943Z	{"eventVersion":"1.08","userIdentity":{"type":"IAMUser","principalId":"AIDAVC63... 349962368193_CloudTrail_eu-west-1_4	349962368193_CloudTrail_eu-west-1_4	349962368193_CloudTrail_eu-west-1_4
9	2024-01-19T12:28:11.611Z	{"eventVersion":"1.08","userIdentity":{"type":"AssumedRole","principalId":"AROA... 349962368193_CloudTrail_eu-west-1_3	349962368193_CloudTrail_eu-west-1_3	349962368193_CloudTrail_eu-west-1_3

Figure 33

This figure shows the CloudTrail log group being used on the CloudWatch dashboard to show recent activities such as EC2 instance shutdowns restarts, etc.

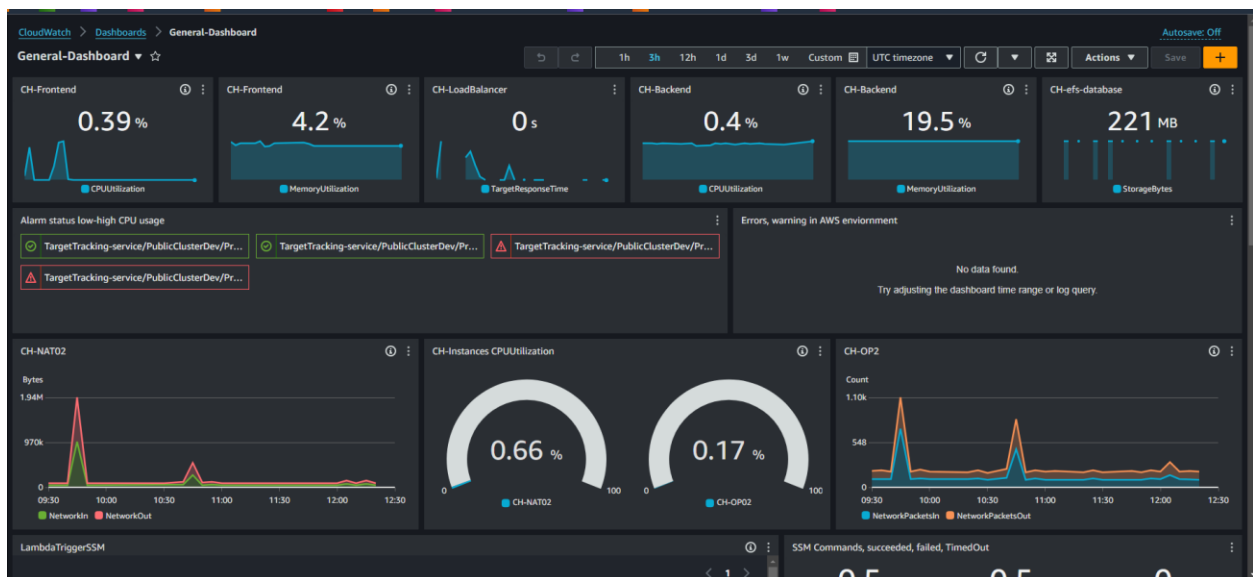


Figure 34

This figure shows the metrics for the CloudWatch dashboard working with no issues.

Security Measures

DDoS Detection

The lambda function to run the Athena query is successful and so are the notifications. This was tested by simply lowering the alarm threshold and then running the function since we could not test it any other way.

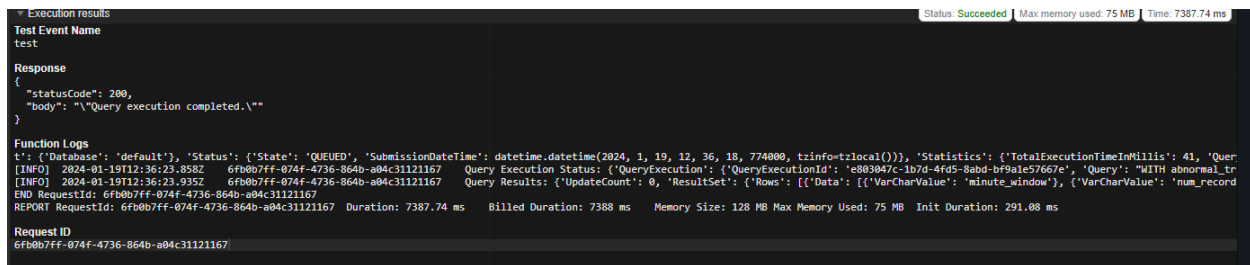


Figure 35

This figure shows the lambda function being successfully run with no issues. No alarm was sent because there was no attack.

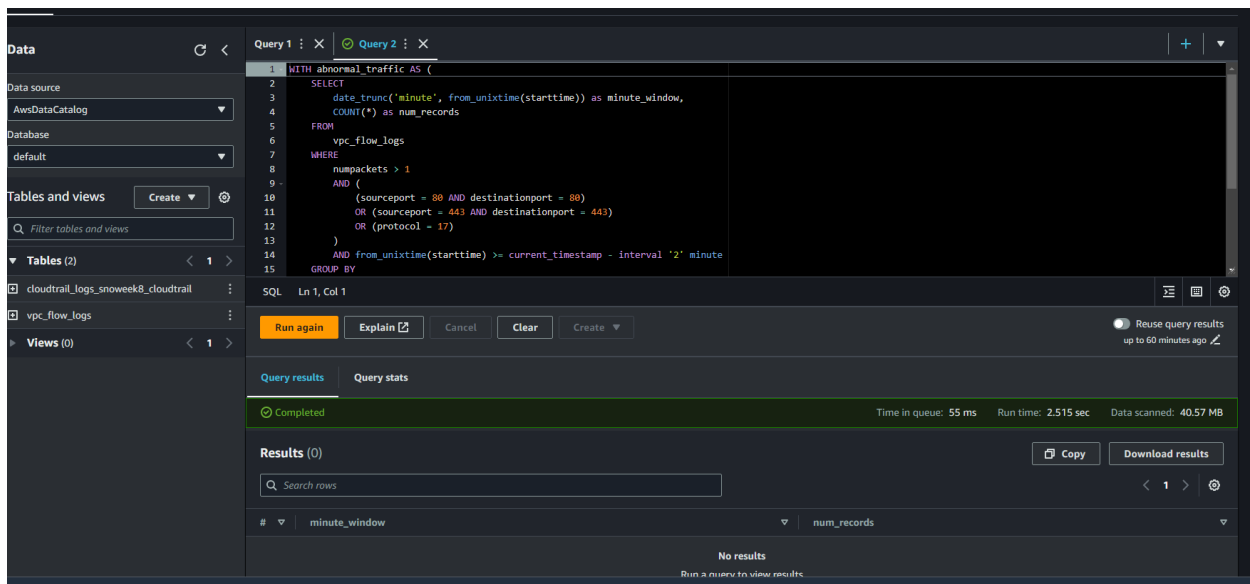


Figure 36

This figure shows the same query being run in Athena to show that there were no results. The query checks for packets received over 150000 from 443, 80 and UDP protocols (from UDP flooding possibilities) and there were no results.

athenaqueryddoss [Info](#)

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

Objects (22) [Info](#) [Copy S3 URI](#) [Copy URL](#) [Download](#) [Open](#) [Delete](#) [Actions](#) [Create folder](#) [Upload](#)

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 access your objects, you'll need to explicitly grant them permissions. [Learn more](#)

Find objects by prefix

	Name	Type	Last modified	Size	Storage class
<input type="checkbox"/>	40202142-5053-428e-98dc-17677e97b664.csv	csv	January 19, 2024, 11:10:34 (UTC+01:00)	30.0 B	Standard
<input type="checkbox"/>	40202142-5053-428e-98dc-17677e97b664.csv.metadata	metadata	January 19, 2024, 11:10:34 (UTC+01:00)	136.0 B	Standard
<input type="checkbox"/>	4b6a1e50-8e61-4eeb-86f3-893b503dff5.csv	csv	January 18, 2024, 17:20:38 (UTC+01:00)	114.0 B	Standard
<input type="checkbox"/>	4b6a1e50-8e61-4eeb-86f3-893b503dff5.csv.metadata	metadata	January 18, 2024, 17:20:38 (UTC+01:00)	449.0 B	Standard
<input type="checkbox"/>	5ce986b9-228f-4016-ba9b-b63318b1432f.csv	csv	January 19, 2024, 11:35:23 (UTC+01:00)	30.0 B	Standard
<input type="checkbox"/>	5ce986b9-228f-4016-ba9b-b63318b1432f.csv.metadata	metadata	January 19, 2024, 11:35:23 (UTC+01:00)	136.0 B	Standard
<input type="checkbox"/>	6a6dee2c-0c36-4146-99b5-6e0781ba70ac.csv	csv	January 19, 2024, 10:49:15 (UTC+01:00)	114.0 B	Standard

Figure 37

This figure shows where the athena queries are stored in S3.

AN **AWS Notifications** <no-reply@sns.amazonaws.com> [Sun](#) [Smile](#) [Reply](#) [Forward](#) [More](#)

To: [Balani, Divesh D.N.](#) Thu 1/18/2024 5:10 PM

The number of requests exceeded the threshold within the last minute.

--

If you wish to stop receiving notifications from this topic, please click or visit the link below to unsubscribe:
<https://eur01.safelinks.protection.outlook.com/?url=https%3A%2F%2Fsns.eu-west-1.amazonaws.com%2Funsubscribe.html%3FSubscriptionArn%3Darn%3Aaws%3Asns%3Aeu-west-1%3A349962368193%3Asnoweek8%3A00747aae-2bb1-4044-a86d-023ea1fde6af%26Endpoint%3Ddivesh.balani%40student.fontys.nl&data=05%7C02%7Cdivesh.balani%40student.fontys.nl%7Caec899b2f9cf4e6ec58108dc183ffb46%7Cc66b6765b7944a2b84ed845b341c086a%7C0%7C0%7C638411910292986447%7CUnknown%7CTWFpbGZsb3d8eyJWljiMC4wLjAwMDAilCJlQjoiV2luMzliLCJBTil6lk1haWwiLCJXVCi6Mn0%3D%7C3000%7C%7C&sdata=uZO6lFgaqJmlgwQS3oqcPJ5OGH4E54aGhAid7UMp5e4%3D&reserved=0>

Please do not reply directly to this email. If you have any questions or comments regarding this email, please contact us at <https://eur01.safelinks.protection.outlook.com/?url=https%3A%2F%2Faws.amazon.com%2Fsupport&data=05%7C02%7Cdivesh.balani%40student.fontys.nl%7Caec899b2f9cf4e6ec58108dc183ffb46%7Cc66b6765b7944a2b84ed845b341c086a%7C0%7C0%7C638411910292986447%7CUnknown%7CTWFpbGZsb3d8eyJWljiMC4wLjAwMDAilCJlQjoiV2luMzliLCJBTil6lk1haWwiLCJXVCi6Mn0%3D%7C3000%7C%7C&sdata=IWOXGS0S0HYzsRS2l%2B9yczkvJj0VBIAp7UkUJVNTgtaw%3D&reserved=0>

[Reply](#) [Forward](#)

Figure 38

Once I ran the same thing with a lower threshold, I received an alarm successfully.

Conclusion

This test was close to 95% successful. The website worked along with its login and register features. The automation scripts worked, Ansible triggered the Terraform script to create the infrastructure and it caught a standard error. Terraform in turn also worked as you may have seen in the chapter above. It created all the specified resources. The Lambda scripts worked, and a confirmation email was sent when the infrastructure was created.

This is all good, but there were a few things that did not work unfortunately. For starters, the FTP is not accessible. The container itself is healthy and running but is not accessible, probably because of a port issue. The other part that is not working is also a container, this is the MySQL container. This one does not start up at all, probably because there is an issue within the Terraform we couldn't figure out.

These are the results of this test report, as you can see a lot of it works except for a few minor functionalities but this is something that could be worked on in future sprints.