We will try to simulate incident with one of the instances and see how AWS handles and how long it takes

As we can see around 20:18 both instances are working as expected and return the internal hostname, in this case

ip-10-0-1-46.eu-west-3.compute.internal ip-10-0-2-94.eu-west-3.compute.internal

∾ % date "+%H:%M:%S" && http http://app-lb-2017845830.eu-west-3.elb.amazonaws.com 20:18:20 Connection: keep-alive Content-Length: 39 Content-Type: text/plain; charset=utf-8 Date: Thu, 03 Apr 2025 16:18:20 GMT server: uvicorn ip-10-0-1-46.eu-west-3.compute.internal ∼ % date "+%H:%M:%S" && http http://app-lb-2017845830.eu-west-3.elb.amazonaws.com 20:18:20 Connection: keep-alive Content-Length: 39 Content-Type: text/plain; charset=utf-8 Date: Thu, 03 Apr 2025 16:18:21 GMT server: uvicorn ip-10-0-2-94.eu-west-3.compute.internal

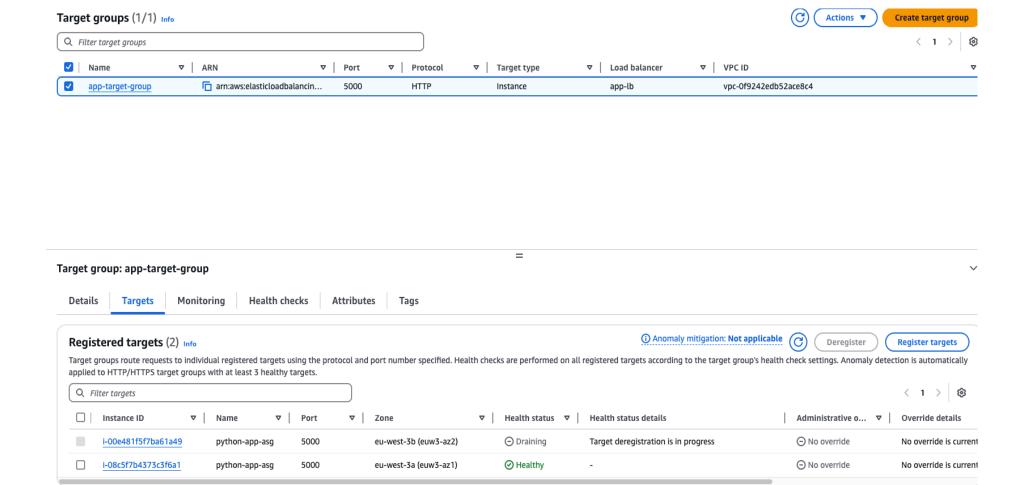
And the same goes for the health check. Both health checks are working, and both instances are healthy.

```
~ % date "+%H:%M:%S" && http http://app-lb-2017845830.eu-west-3.elb.amazonaws.com/healthcheck 20:20:22
HTTP/1.1 200 OK
Connection: keep-alive
Content-Length: 6
Content-Type: text/plain; charset=utf-8
Date: Thu, 03 Apr 2025 16:20:23 GMT
server: uvicorn
OK 200
```

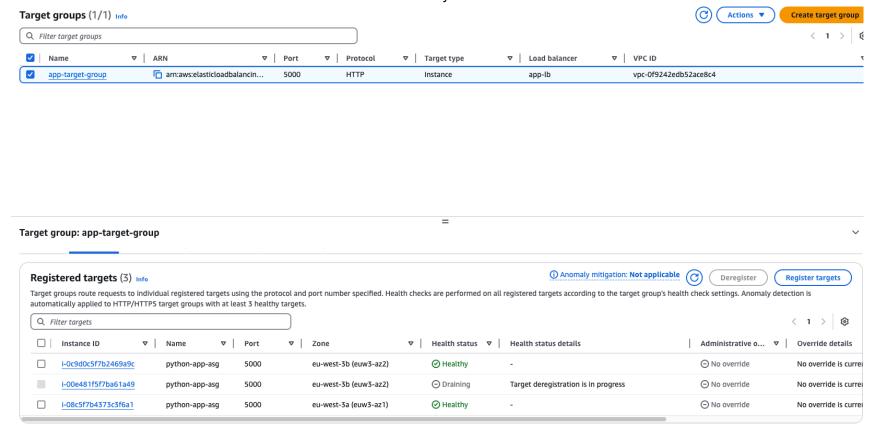
Around 20:22, we introduced a fault in the system by triggering the /terminate-instance path, which changes the status code to 404 for the /healthcheck path. As you can see, after a few seconds, one of the hosts returns a "Healthcheck Fails" with 404 error code, while the other instance works.

```
∼ % date "+%H:%M:%S" && http http://app-lb-2017845830.eu-west-3.elb.amazonaws.com/terminate-instance
20:22:10
Connection: keep-alive
Content-Length: 29
Content-Type: text/plain; charset=utf-8
Date: Thu, 03 Apr 2025 16:22:11 GMT
server: uvicorn
Healthcheck status set to 404
                     ∼ % date "+%H:%M:%S" && http http://app-lb-2017845830.eu-west-3.elb.amazonaws.com/healthcheck
20:22:13
Connection: keep-alive
Content-Length: 21
Content-Type: text/plain; charset=utf-8
Date: Thu, 03 Apr 2025 16:22:13 GMT
server: uvicorn
Healthcheck Fails 404
                   ∼ % date "+%H:%M:%S" && http http://app-lb-2017845830.eu-west-3.elb.amazonaws.com/healthcheck
20:22:15
Connection: keep-alive
Content-Length: 6
Content-Type: text/plain; charset=utf-8
Date: Thu, 03 Apr 2025 16:22:15 GMT
server: uvicorn
OK 200
```

After a few moments, we can see that one of the instances in the target group became unhealthy and is draining. This means the Auto Scaling group detected the unhealthy instance and is creating a new one.



After a minute or so we can see a new instance created and healthy.



New host with internal host ip-10-0-2-173.eu-west-3.compute.internal

```
% date "+%H:%M:%S" && http http://app-lb-2017845830.eu-west-3.elb.amazonaws.com/
20:27:24
HTTP/1.1 200 OK
Connection: keep-alive
Content-Length: 40
Content-Type: text/plain; charset=utf-8
Date: Thu, 03 Apr 2025 16:27:24 GMT
server: uvicorn
ip-10-0-2-173.eu-west-3.compute.internal
```

Incident Summary

One instance in the autoscaling group failed its health check with a 404 response. The autoscaling mechanism terminated the unhealthy instance and launched a replacement, which passed the health check and functioned normally.

However, there are scenarios where the autoscaling group may not be able to heal the system:

- When S3 is unavailable
- When there is a configuration error in the Python file or in the AWS Terraform configuration
- When three availability zones are not available