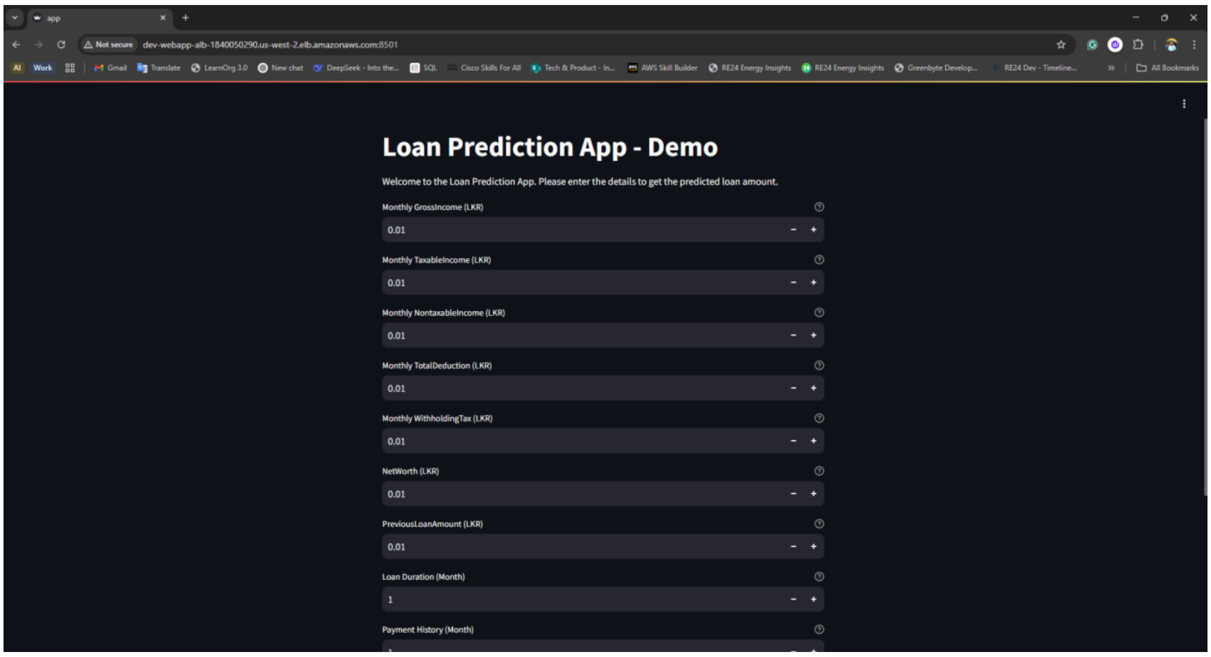
Platform Engineer Assignment

Deploying a Containerised Application Using IaC

Akila Hettiarachchi

Simple Web Application

URL: http://dev-webapp-alb-1840050290.us-west-2.elb.amazonaws.com:8501



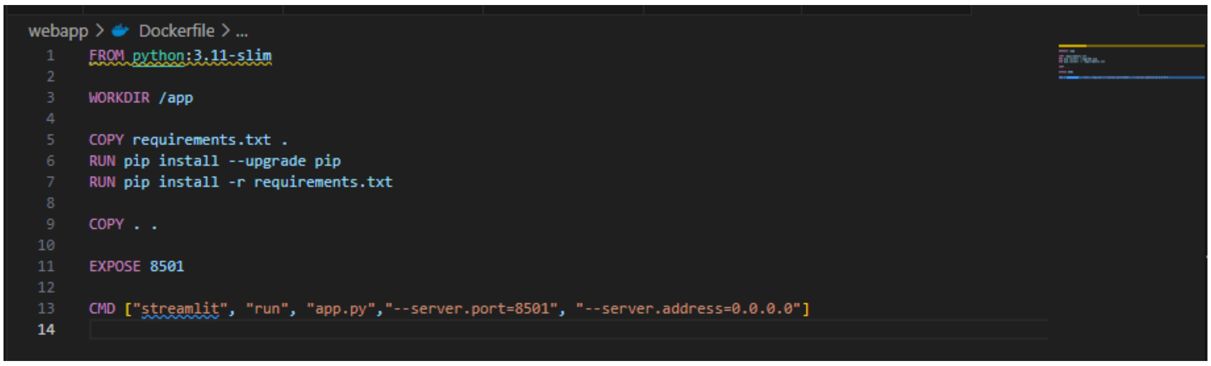
Loan Prediction App - Demo

Welcome to the Loan Prediction App. Please enter the details to get the predicted loan amount.

Monthly GrossIncome (LKR)	0.01
Monthly TaxableIncome (LKR)	0.01
Monthly NontaxableIncome (LKR)	0.01
Monthly TotalDeduction (LKR)	0.01
Monthly WithholdingTax (LKR)	0.01
NetWorth (LKR)	0.01
PreviousLoanAmount (LKR)	0.01
Loan Duration (Month)	1
Payment History (Month)	

Step 1: Containerization

I developed a demo web application using Python and Streamlit. I then containerised it using Docker, based on the python:3.11-slim image. The Docker container exposes port 8501, the default port Streamlit uses.



webapp > Dockerfile > ...

```dockerfile
FROM python:3.11-slim

WORKDIR /app

COPY requirements.txt .
RUN pip install --upgrade pip
RUN pip install -r requirements.txt

COPY . .

EXPOSE 8501

CMD ["streamlit", "run", "app.py","--server.port=8501", "--server.address=0.0.0.0"]
```

Figure 01: Docker file

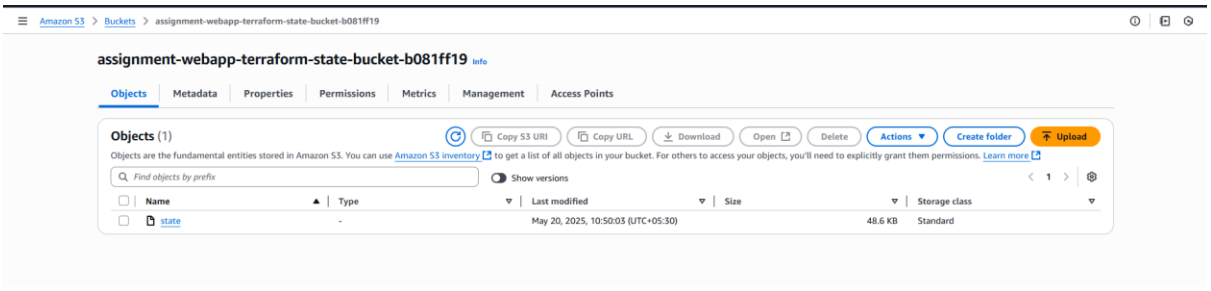
Step 2: Infrastructure as Code (IaC) with Terraform

2.1 Define AWS ECS Cluster, Services, and Task Definitions, etc., using Terraform.

I set up a remote backend using AWS S3 and DynamoDB to store the Terraform state file and manage state locking. This setup ensures that Terraform works seamlessly both locally and in the GitLab pipeline, and it also facilitates team collaboration.

Using Terraform, I created an Amazon ECR repository to store the built web application image. I then defined an ECS cluster, service, and task definition to run the stored image. Additionally, I created a VPC and subnets to support the ECS infrastructure. A security group was configured to manage ECS network access, allowing public access on port 8501.

I assigned an IAM role for ECS task execution and configured CloudWatch Logs to view application logs. Finally, I set up an Application Load Balancer (ALB) and a target group to handle incoming web application traffic.



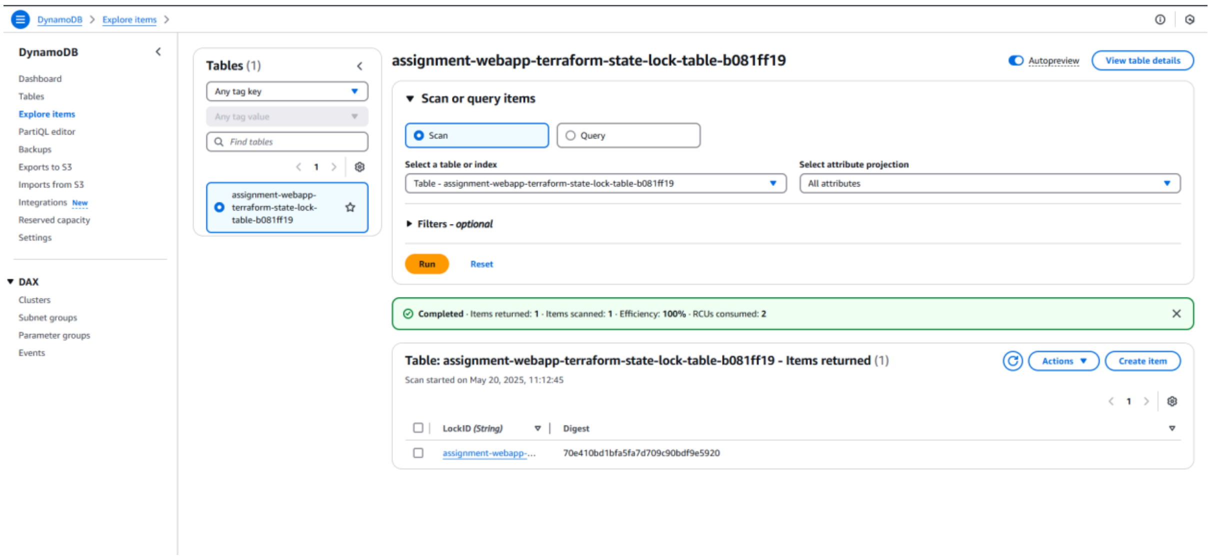
assignment-webapp-terraform-state-bucket-b081ff19 Info

Objects (1)

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

Name	Type	Last modified	Size	Storage class
state	-	May 20, 2025, 10:50:03 (UTC+05:30)	48.6 KB	Standard

Figure 02: S3 bucket for state store



Tables (1)

assignment-webapp-terraform-state-lock-table-b081ff19

assignment-webapp-terraform-state-lock-table-b081ff19

Scan or query items

[✓ Scan] Query

Select a table or index: Table - assignment-webapp-terraform-state-lock-table-b081ff19

Select attribute projection: All attributes

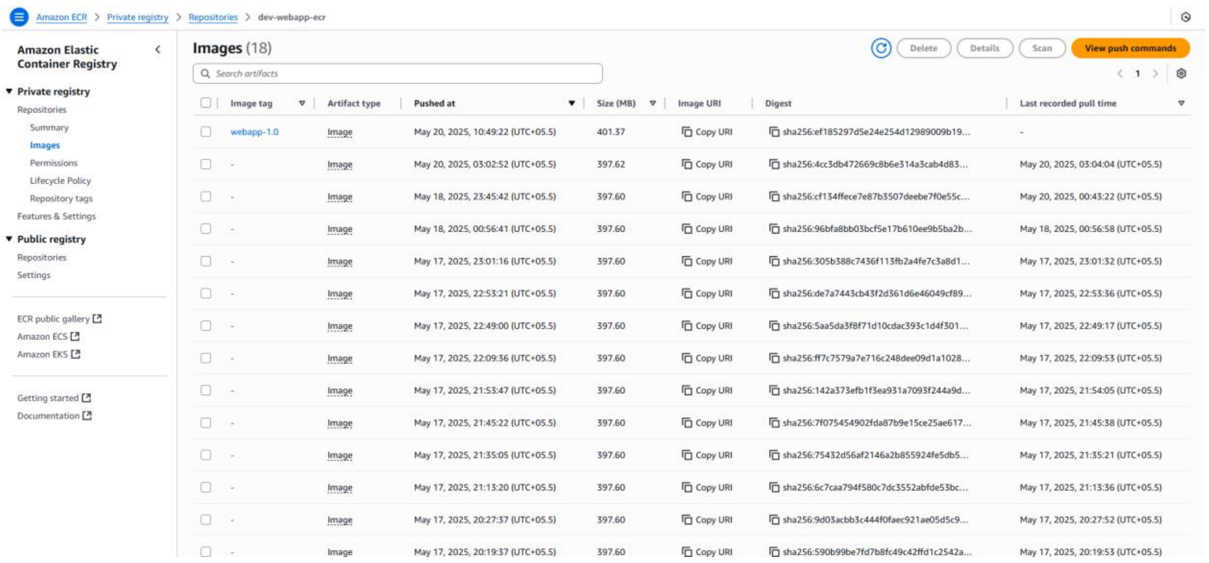
Completed · Items returned: 1 · Items scanned: 1 · Efficiency: 100% · RCUs consumed: 2

Table: assignment-webapp-terraform-state-lock-table-b081ff19 - Items returned (1)

Scan started on May 20, 2025, 11:12:45

LockID (String)	Digest
assignment-webapp-...	70e410bd1bfa5fa7d709c90bdf9e5920

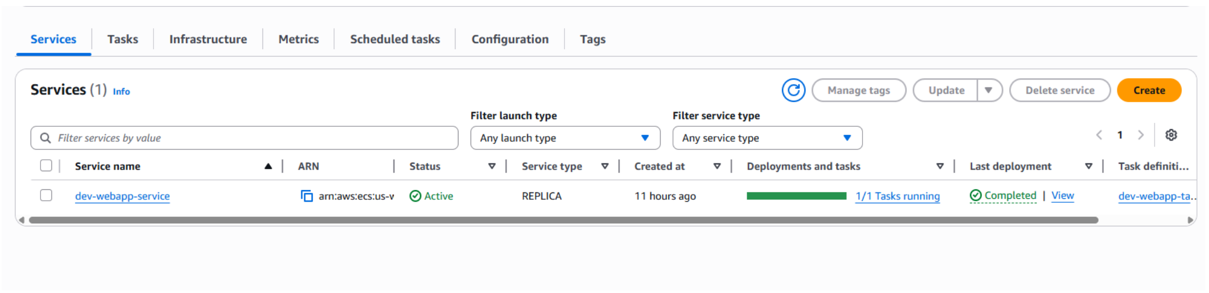
Figure 03: DynamoDB for state locking



Images (18)

Image tag	Artifact type	Pushed at	Size (MB)	Image URI	Digest	Last recorded pull time
webapp-1.0	Image	May 20, 2025, 10:49:22 (UTC+05.5)	401.37	Copy URI	sha256:ef185297d5e24e254d12989009b19...	-
-	Image	May 20, 2025, 03:02:52 (UTC+05.5)	397.62	Copy URI	sha256:4cc3db472669c8b6e314a3cab4d83...	May 20, 2025, 03:04:04 (UTC+05.5)
-	Image	May 18, 2025, 23:45:42 (UTC+05.5)	397.60	Copy URI	sha256:cf134ffece7e87b3507deebe7f0e55c...	May 20, 2025, 00:43:22 (UTC+05.5)
-	Image	May 18, 2025, 00:56:41 (UTC+05.5)	397.60	Copy URI	sha256:96bfa8bb03bcf5e17b610ee9b5ba2b...	May 18, 2025, 00:56:58 (UTC+05.5)
-	Image	May 17, 2025, 23:01:16 (UTC+05.5)	397.60	Copy URI	sha256:305b388c7436f113fb2a4fe7c3a8d1...	May 17, 2025, 23:01:32 (UTC+05.5)
-	Image	May 17, 2025, 22:53:21 (UTC+05.5)	397.60	Copy URI	sha256:de7a7443cb43f2d361d6e46049cf89...	May 17, 2025, 22:53:36 (UTC+05.5)
-	Image	May 17, 2025, 22:49:00 (UTC+05.5)	397.60	Copy URI	sha256:5aa5da3f8f71d10cdac393c1d4f301...	May 17, 2025, 22:49:17 (UTC+05.5)
-	Image	May 17, 2025, 22:09:36 (UTC+05.5)	397.60	Copy URI	sha256:ff7c7579a7e716c248dee09d1a1028...	May 17, 2025, 22:09:53 (UTC+05.5)
-	Image	May 17, 2025, 21:53:47 (UTC+05.5)	397.60	Copy URI	sha256:142a373efb1f3ea931a7093f244a9d...	May 17, 2025, 21:54:05 (UTC+05.5)
-	Image	May 17, 2025, 21:45:22 (UTC+05.5)	397.60	Copy URI	sha256:7f075454902fda87b9e15ce25ae617...	May 17, 2025, 21:45:38 (UTC+05.5)
-	Image	May 17, 2025, 21:35:05 (UTC+05.5)	397.60	Copy URI	sha256:75432d56af2146a2b855924fe5db5...	May 17, 2025, 21:35:21 (UTC+05.5)
-	Image	May 17, 2025, 21:13:20 (UTC+05.5)	397.60	Copy URI	sha256:6c7caa794f580c7dc3552abfde53bc...	May 17, 2025, 21:13:36 (UTC+05.5)
-	Image	May 17, 2025, 20:27:37 (UTC+05.5)	397.60	Copy URI	sha256:9d03acbb3c444f0faec921ae05d5c9...	May 17, 2025, 20:27:52 (UTC+05.5)
-	Image	May 17, 2025, 20:19:37 (UTC+05.5)	397.60	Copy URI	sha256:590b99be7fd7b8fc49c42ffd1c2542a...	May 17, 2025, 20:19:53 (UTC+05.5)

Figure 04: ECR Repo

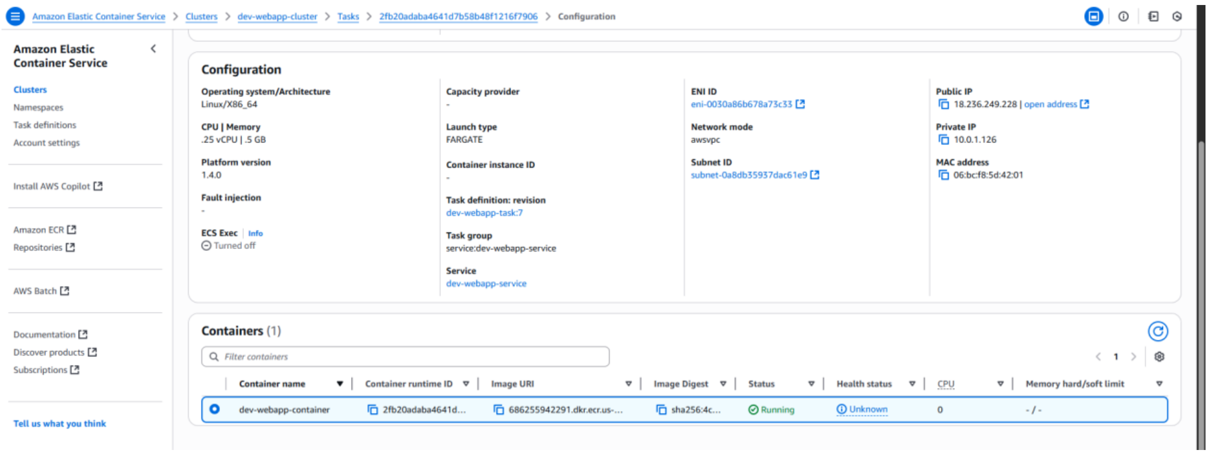


Services (1)

Filter launch type: Any launch type
Filter service type: Any service type

Service name	ARN	Status	Service type	Created at	Deployments and tasks	Last deployment	Task definiti...
dev-webapp-service	arn:aws:ecs:us-v	Active	REPLICA	11 hours ago	1/1 Tasks running	Completed | View	dev-webapp-ta...

Figure 05: ECS Service



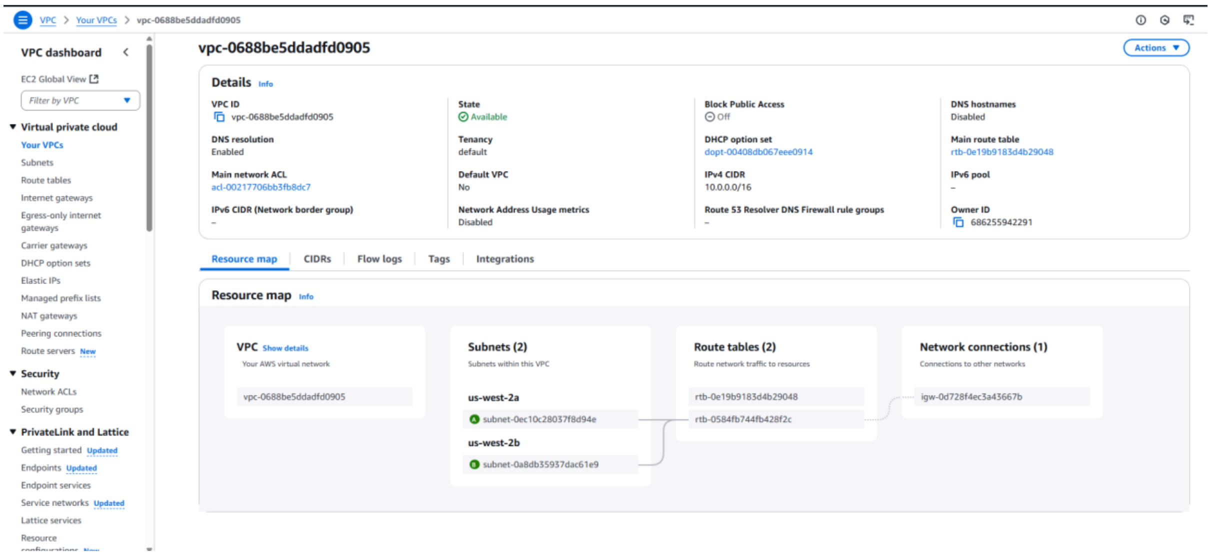
Configuration

Operating system/Architecture Linux/X86_64	**Capacity provider** -	**ENI ID** eni-0030a86b678a73c33	**Public IP** 18.236.249.228 | open address
CPU | Memory .25 vCPU | .5 GB	**Launch type** FARGATE	**Network mode** awsvpc	**Private IP** 10.0.1.126
Platform version 1.4.0	**Container instance ID** -	**Subnet ID** subnet-0a8db35937dac61e9	**MAC address** 06:bc:f8:5d:42:01
Fault injection -	**Task definition: revision** dev-webapp-task:7		
ECS Exec Info Turned off	**Task group** service:dev-webapp-service		
	Service dev-webapp-service		

Containers (1)

Container name	Container runtime ID	Image URI	Image Digest	Status	Health status	CPU	Memory hard/soft limit
dev-webapp-container	2fb20adaba4641d...	686255942291.dkr.ecr.us-...	sha256:4c...	Running	Unknown	0	- / -

Figure 06: ECS Task



VPC > Your VPCs > vpc-0688be5ddadfd0905

VPC dashboard

EC2 Global View

Filter by VPC

- Virtual private cloud
  - Your VPCs
  - Subnets
  - Route tables
  - Internet gateways
  - Egress-only internet gateways
  - Carrier gateways
  - DHCP option sets
  - Elastic IPs
  - Managed prefix lists
  - NAT gateways
  - Peering connections
  - Route servers New
- Security
  - Network ACLs
  - Security groups
- PrivateLink and Lattice
  - Getting started Updated
  - Endpoints Updated
  - Endpoint services
  - Service networks Updated
  - Lattice services
  - Resource configurations New

vpc-0688be5ddadfd0905

Details Info

VPC ID vpc-0688be5ddadfd0905	**State** Available	**Block Public Access** Off	**DNS hostnames** Disabled
DNS resolution Enabled	**Tenancy** default	**DHCP option set** dopt-00408db067eee0914	**Main route table** rtb-0e19b9183d4b29048
Main network ACL acl-00217706bb3fb8dc7	**Default VPC** No	**IPv4 CIDR** 10.0.0.0/16	**IPv6 pool** –
IPv6 CIDR (Network border group) –	**Network Address Usage metrics** Disabled	**Route 53 Resolver DNS Firewall rule groups** –	**Owner ID** 686255942291

Resource map | CIDRs | Flow logs | Tags | Integrations

Resource map Info

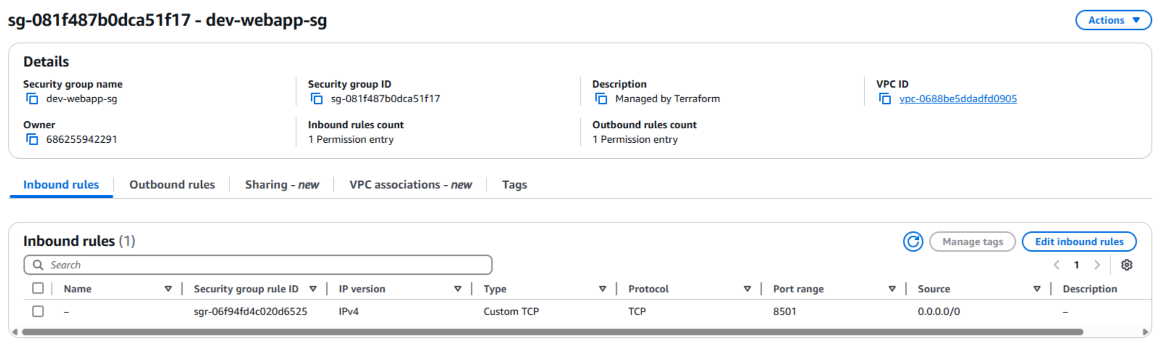
VPC Show details
Your AWS virtual network
vpc-0688be5ddadfd0905

Subnets (2)
Subnets within this VPC
us-west-2a
subnet-0ec10c28037f8d94e
us-west-2b
subnet-0a8db35937dac61e9

Route tables (2)
Route network traffic to resources
rtb-0e19b9183d4b29048
rtb-0584fb744fb428f2c

Network connections (1)
Connections to other networks
igw-0d728f4ec3a43667b

Figure 07: VPC



sg-081f487b0dca51f17 - dev-webapp-sg

Details

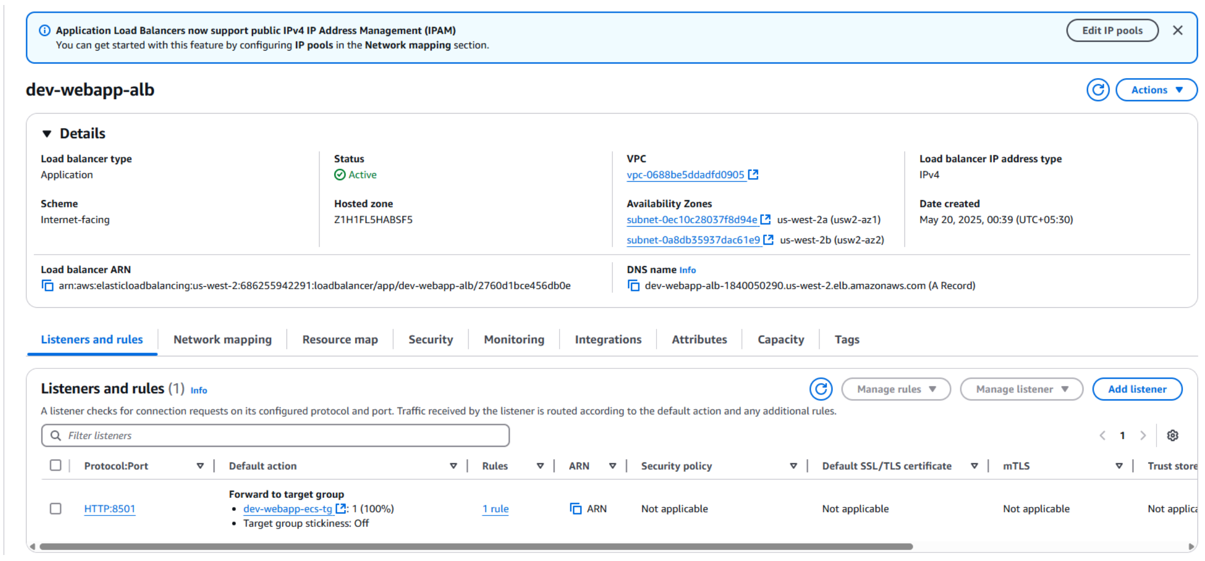
Security group name dev-webapp-sg	**Security group ID** sg-081f487b0dca51f17	**Description** Managed by Terraform	**VPC ID** vpc-0688be5ddadfd0905
Owner 686255942291	**Inbound rules count** 1 Permission entry	**Outbound rules count** 1 Permission entry	

Inbound rules | Outbound rules | Sharing - *new* | VPC associations - *new* | Tags

Inbound rules (1)

Name	Security group rule ID	IP version	Type	Protocol	Port range	Source	Description
–	sgr-06f94fd4c020d6525	IPv4	Custom TCP	TCP	8501	0.0.0.0/0	–

Figure 08: Security group



Application Load Balancers now support public IPv4 IP Address Management (IPAM)
You can get started with this feature by configuring **IP pools** in the **Network mapping** section.

dev-webapp-alb

Details

Load balancer type Application	**Status** Active	**VPC** vpc-0688be5ddadfd0905	**Load balancer IP address type** IPv4
Scheme Internet-facing	**Hosted zone** Z1H1FL5HABSF5	**Availability Zones** subnet-0ec10c28037f8d94e us-west-2a (usw2-az1) subnet-0a8db35937dac61e9 us-west-2b (usw2-az2)	**Date created** May 20, 2025, 00:39 (UTC+05:30)

Load balancer ARN
arn:aws:elasticloadbalancing:us-west-2:686255942291:loadbalancer/app/dev-webapp-alb/2760d1bce456db0e

DNS name Info
dev-webapp-alb-1840050290.us-west-2.elb.amazonaws.com (A Record)

Listeners and rules | Network mapping | Resource map | Security | Monitoring | Integrations | Attributes | Capacity | Tags

Listeners and rules (1) Info

A listener checks for connection requests on its configured protocol and port. Traffic received by the listener is routed according to the default action and any additional rules.

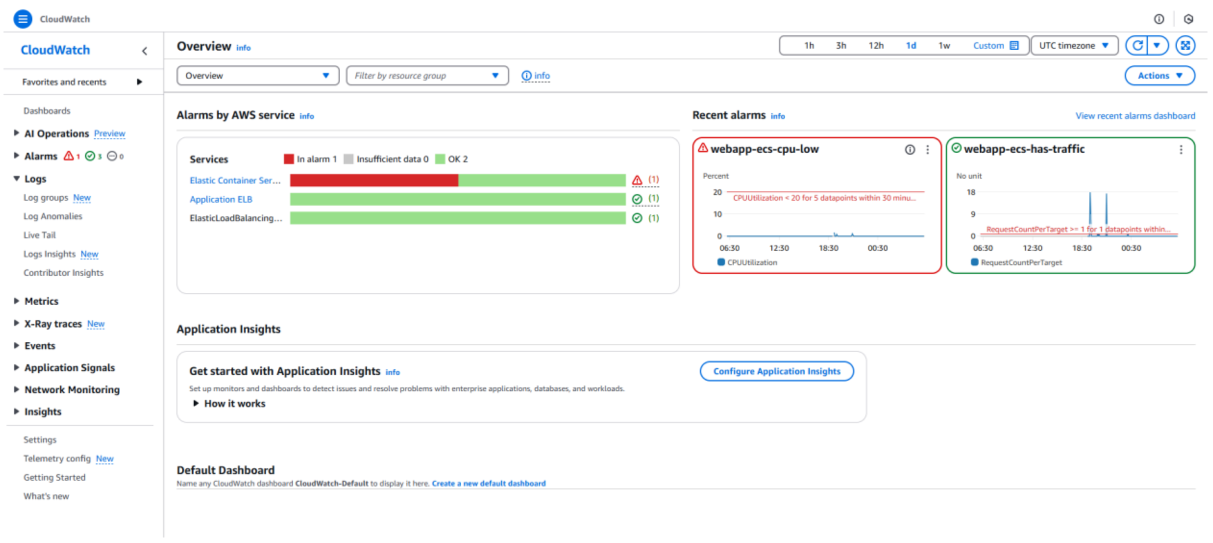
Protocol:Port	Default action	Rules	ARN	Security policy	Default SSL/TLS certificate	mTLS	Trust store
HTTP:8501	**Forward to target group** • dev-webapp-ecs-tg: 1 (100%) • Target group stickiness: Off	1 rule	ARN	Not applicable	Not applicable	Not applicable	Not applica

Figure 09: Load Balancer

2.2 Implement vertical scaling.

AWS does not provide a native service for vertical auto scaling. Therefore, I implemented a custom solution using CloudWatch Alarms and Lambda functions. CloudWatch metrics monitor ECS service CPU usage and ALB target group activity.

If CPU usage exceeds 80%, a CloudWatch alarm triggers the scale-up Lambda function, which vertically scales the ECS task by increasing vCPU and memory. Conversely, if CPU usage drops below 20% and there is at least one active request, a scale-down Lambda function is triggered to reduce vCPU and memory allocation. It helps efficiently manage resources while maintaining application performance.



Alarms by AWS service

Services	🟥 In alarm 1 ▨ Insufficient data 0 🟩 OK 2
Elastic Container Ser...	⚠ (1)
Application ELB	✓ (1)
ElasticLoadBalancing...	✓ (1)

Recent alarms

webapp-ecs-cpu-low: CPUUtilization < 20 for 5 datapoints within 30 minu...

webapp-ecs-has-traffic: RequestCountPerTarget >= 1 for 1 datapoints within...

Application Insights

Get started with Application Insights

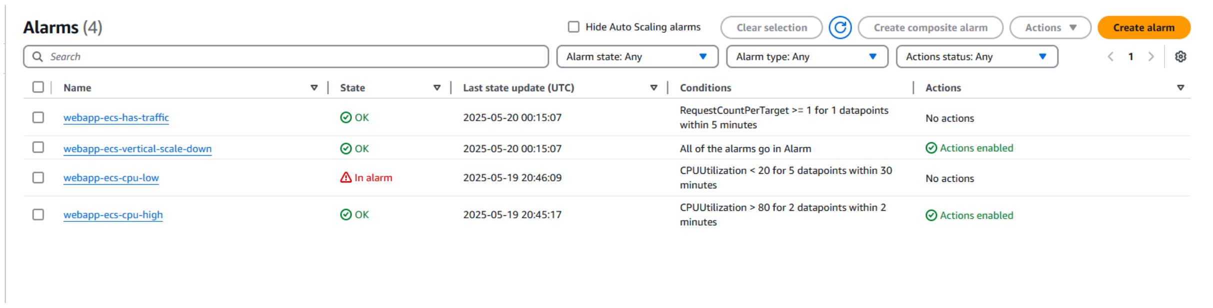
Set up monitors and dashboards to detect issues and resolve problems with enterprise applications, databases, and workloads.

Configure Application Insights

Default Dashboard

Name any CloudWatch dashboard **CloudWatch-Default** to display it here. Create a new default dashboard

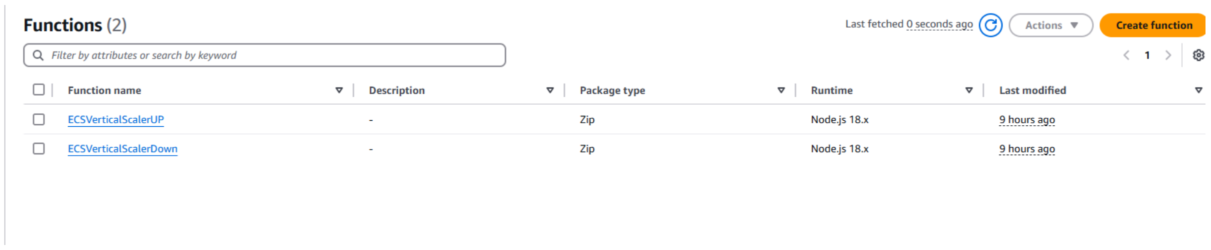
Figure 10: CloudWatch alarm overview



Alarms (4)

Name	State	Last state update (UTC)	Conditions	Actions
webapp-ecs-has-traffic	✓ OK	2025-05-20 00:15:07	RequestCountPerTarget >= 1 for 1 datapoints within 5 minutes	No actions
webapp-ecs-vertical-scale-down	✓ OK	2025-05-20 00:15:07	All of the alarms go in Alarm	✓ Actions enabled
webapp-ecs-cpu-low	⚠ In alarm	2025-05-19 20:46:09	CPUUtilization < 20 for 5 datapoints within 30 minutes	No actions
webapp-ecs-cpu-high	✓ OK	2025-05-19 20:45:17	CPUUtilization > 80 for 2 datapoints within 2 minutes	✓ Actions enabled

Figure 11: CloudWatch alarms list



Functions (2)

Last fetched 0 seconds ago

Function name	Description	Package type	Runtime	Last modified
ECSVerticalScalerUP	-	Zip	Node.js 18.x	9 hours ago
ECSVerticalScalerDown	-	Zip	Node.js 18.x	9 hours ago

Figure 12: Lambda Functions

Step 3: GitLab CI/CD Pipeline

I defined a GitLab CI/CD pipeline with three stages:

1. Run Tests

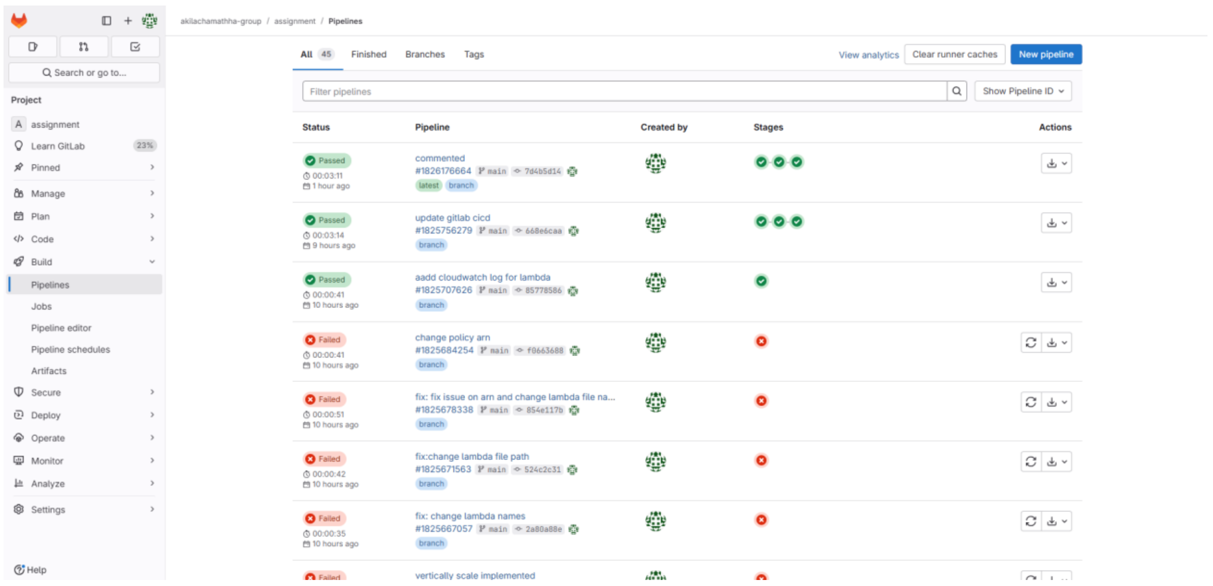
Used the “python:3.10-slim” image to execute web application tests. The tests were run using “pytest”.

2. Build Image

Used the “docker:28.1.1” image and “docker:28.1.1-dind” as the service image. In this stage, the pipeline logs into Amazon ECR using the AWS CLI, builds the Docker container, and pushes it to the ECR repository.

3. Deploy Image

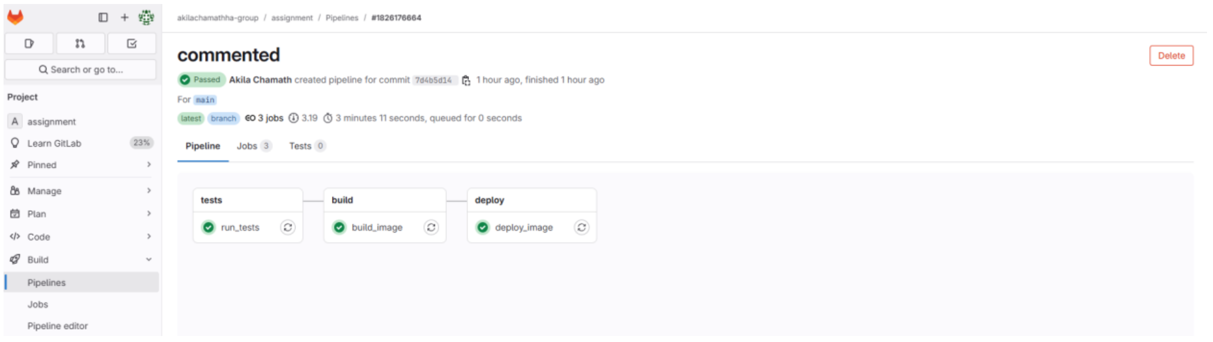
Used the “alpine:3.20” image. A Python virtual environment is created, and the AWS CLI is installed and configured. Then, Terraform is installed using the official binary. Finally, the Terraform configuration is applied to provision infrastructure and deploy the newly built image to ECS.



akilachamathha-group / assignment / Pipelines

Status	Pipeline	Created by	Stages	Actions
Passed 00:03:11 1 hour ago	commented #1826176664 main 7d4b5d14 latest branch		✓ ✓ ✓	
Passed 00:03:14 9 hours ago	update gitlab cicd #1825756279 main 668e6caa branch		✓ ✓ ✓	
Passed 00:00:41 10 hours ago	aadd cloudwatch log for lambda #1825707626 main 85778586 branch		✓	
Failed 00:00:41 10 hours ago	change policy arn #1825684254 main f0663688 branch		✗	
Failed 00:00:51 10 hours ago	fix: fix issue on arn and change lambda file na... #1825678338 main 854e117b branch		✗	
Failed 00:00:42 10 hours ago	fix:change lambda file path #1825671563 main 524c2c31 branch		✗	
Failed 00:00:35 10 hours ago	fix: change lambda names #1825667057 main 2a80a88e branch		✗	
Failed	vertically scale implemented		✗	

Figure 13: GitLab Pipeline runs



akilachamathha-group / assignment / Pipelines / #1826176664

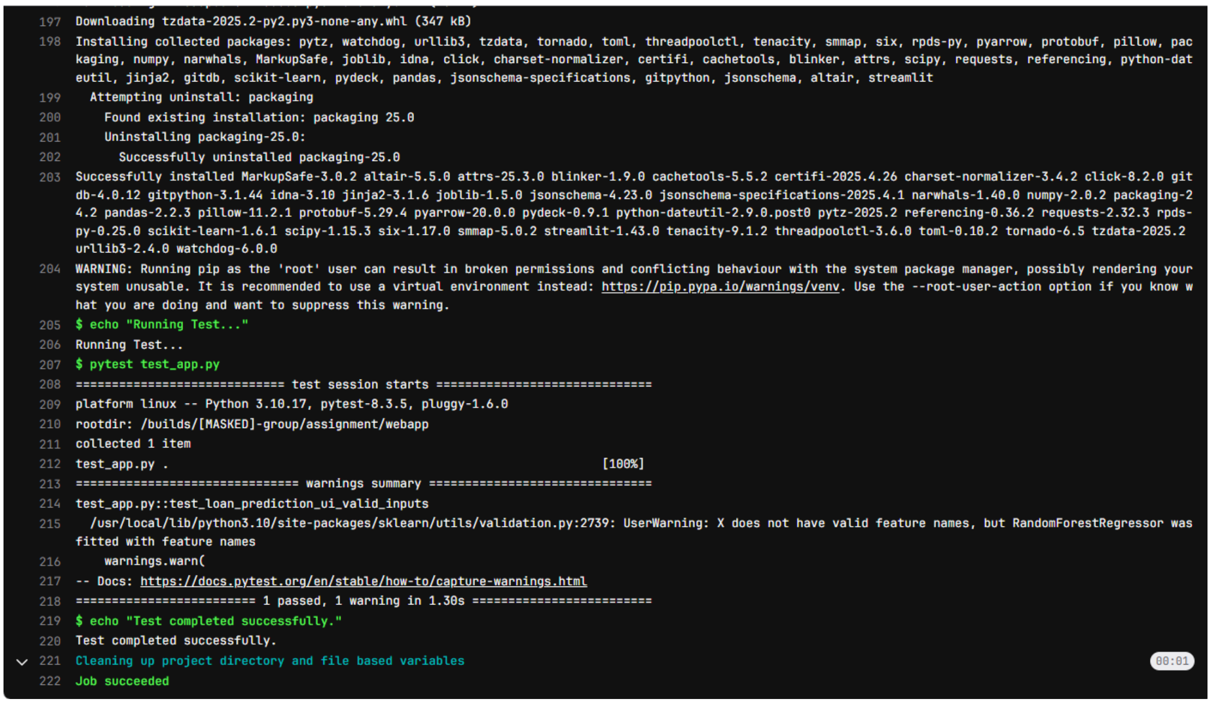
commented

Passed Akila Chamath created pipeline for commit 7d4b5d14 1 hour ago, finished 1 hour ago

For main

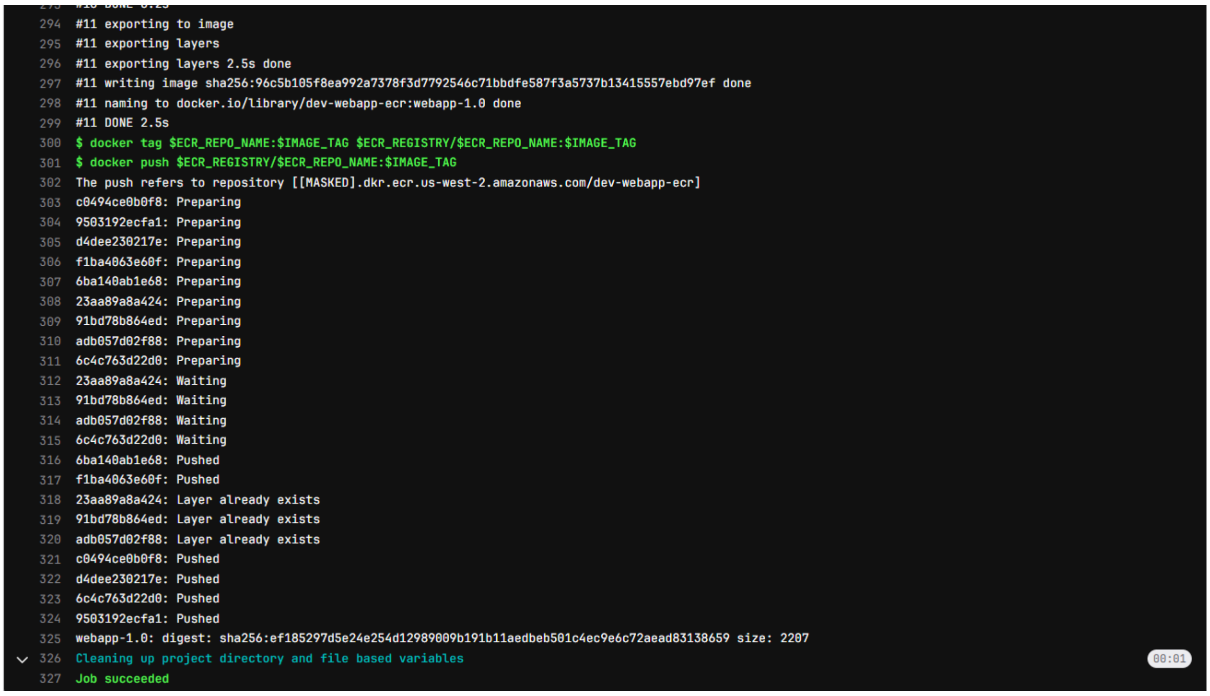
latest branch 3 jobs 3.19 3 minutes 11 seconds, queued for 0 seconds

Figure 14: GitLab Pipeline Stages



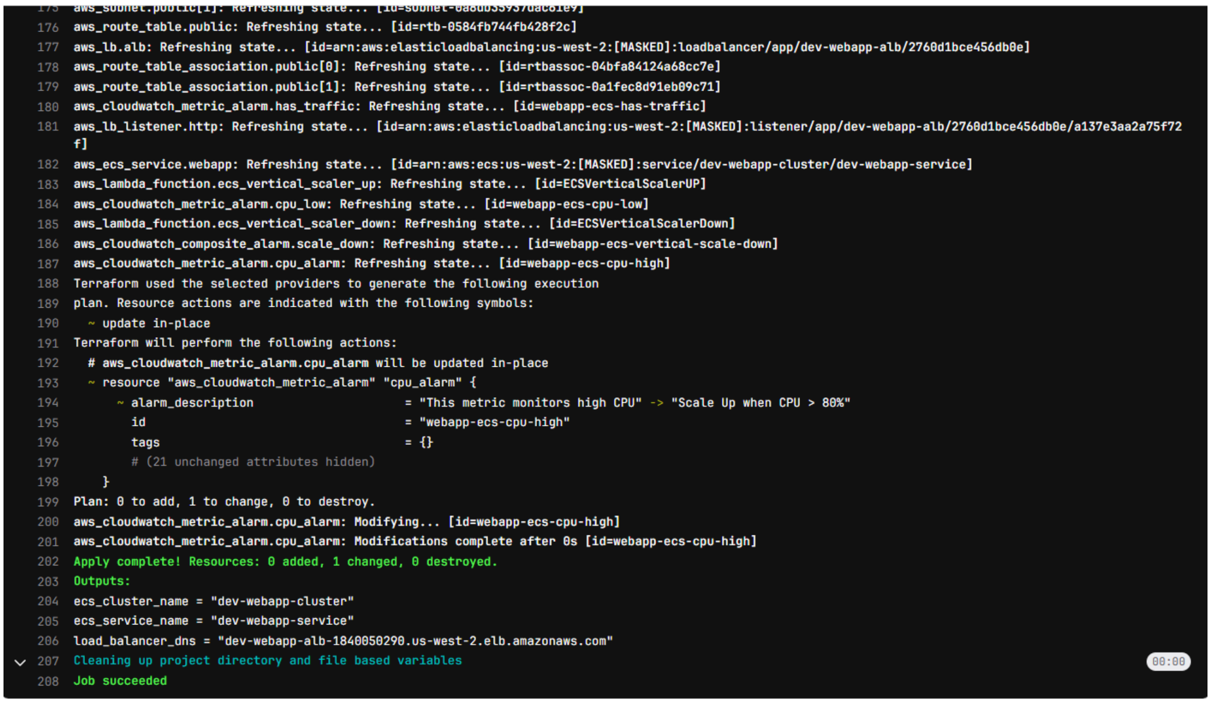
```
Downloading tzdata-2025.2-py2.py3-none-any.whl (347 kB)
Installing collected packages: pytz, watchdog, urllib3, tzdata, tornado, toml, threadpoolctl, tenacity, smmap, six, rpds-py, pyarrow, protobuf, pillow, pac
kaging, numpy, narwhals, MarkupSafe, joblib, idna, click, charset-normalizer, certifi, cachetools, blinker, attrs, scipy, requests, referencing, python-dat
eutil, jinja2, gitdb, scikit-learn, pydeck, pandas, jsonschema-specifications, gitpython, jsonschema, altair, streamlit
  Attempting uninstall: packaging
    Found existing installation: packaging 25.0
    Uninstalling packaging-25.0:
      Successfully uninstalled packaging-25.0
Successfully installed MarkupSafe-3.0.2 altair-5.5.0 attrs-25.3.0 blinker-1.9.0 cachetools-5.5.2 certifi-2025.4.26 charset-normalizer-3.4.2 click-8.2.0 git
db-4.0.12 gitpython-3.1.44 idna-3.10 jinja2-3.1.6 joblib-1.5.0 jsonschema-4.23.0 jsonschema-specifications-2025.4.1 narwhals-1.40.0 numpy-2.0.2 packaging-2
4.2 pandas-2.2.3 pillow-11.2.1 protobuf-5.29.4 pyarrow-20.0.0 pydeck-0.9.1 python-dateutil-2.9.0.post0 pytz-2025.2 referencing-0.36.2 requests-2.32.3 rpds-
py-0.25.0 scikit-learn-1.6.1 scipy-1.15.3 six-1.17.0 smmap-5.0.2 streamlit-1.43.0 tenacity-9.1.2 threadpoolctl-3.6.0 toml-0.10.2 tornado-6.5 tzdata-2025.2
urllib3-2.4.0 watchdog-6.0.0
WARNING: Running pip as the 'root' user can result in broken permissions and conflicting behaviour with the system package manager, possibly rendering your
system unusable. It is recommended to use a virtual environment instead: https://pip.pypa.io/warnings/venv. Use the --root-user-action option if you know w
hat you are doing and want to suppress this warning.
$ echo "Running Test..."
Running Test...
$ pytest test_app.py
============================= test session starts ==============================
platform linux -- Python 3.10.17, pytest-8.3.5, pluggy-1.6.0
rootdir: /builds/[MASKED]-group/assignment/webapp
collected 1 item
test_app.py .                                                            [100%]
=============================== warnings summary ===============================
test_app.py::test_loan_prediction_ui_valid_inputs
  /usr/local/lib/python3.10/site-packages/sklearn/utils/validation.py:2739: UserWarning: X does not have valid feature names, but RandomForestRegressor was
fitted with feature names
    warnings.warn(
-- Docs: https://docs.pytest.org/en/stable/how-to/capture-warnings.html
========================= 1 passed, 1 warning in 1.30s =========================
$ echo "Test completed successfully."
Test completed successfully.
Cleaning up project directory and file based variables                                                                   00:01
Job succeeded
```

Figure 15: GitLab Pipeline Run_Test Log



```
#11 exporting to image
#11 exporting layers
#11 exporting layers 2.5s done
#11 writing image sha256:96c5b105f8ea992a7378f3d7792546c71bbdfe587f3a5737b13415557ebd97ef done
#11 naming to docker.io/library/dev-webapp-ecr:webapp-1.0 done
#11 DONE 2.5s
$ docker tag $ECR_REPO_NAME:$IMAGE_TAG $ECR_REGISTRY/$ECR_REPO_NAME:$IMAGE_TAG
$ docker push $ECR_REGISTRY/$ECR_REPO_NAME:$IMAGE_TAG
The push refers to repository [[MASKED].dkr.ecr.us-west-2.amazonaws.com/dev-webapp-ecr]
c0494ce0b0f8: Preparing
9503192ecfa1: Preparing
d4dee230217e: Preparing
f1ba4063e60f: Preparing
6ba140ab1e68: Preparing
23aa89a8a424: Preparing
91bd78b864ed: Preparing
adb057d02f88: Preparing
6c4c763d22d0: Preparing
23aa89a8a424: Waiting
91bd78b864ed: Waiting
adb057d02f88: Waiting
6c4c763d22d0: Waiting
6ba140ab1e68: Pushed
f1ba4063e60f: Pushed
23aa89a8a424: Layer already exists
91bd78b864ed: Layer already exists
adb057d02f88: Layer already exists
c0494ce0b0f8: Pushed
d4dee230217e: Pushed
6c4c763d22d0: Pushed
9503192ecfa1: Pushed
webapp-1.0: digest: sha256:ef185297d5e24e254d12989009b191b11aedbeb501c4ec9e6c72aead83138659 size: 2207
Cleaning up project directory and file based variables                                                                   00:01
Job succeeded
```

Figure 16: GitLab Pipeline Build_Image Log



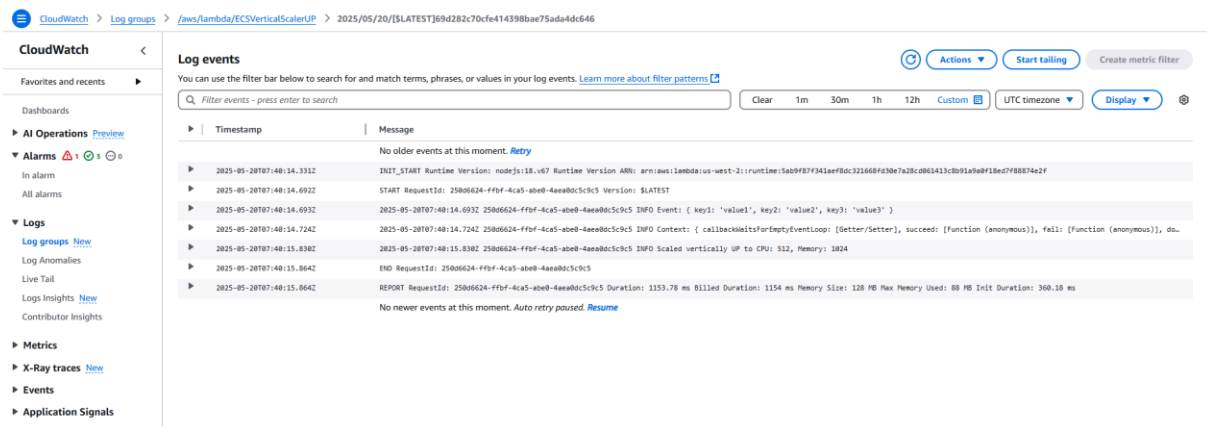
```
aws_subnet.public[1]: Refreshing state... [id=subnet-0a8db35937dac81e9]
aws_route_table.public: Refreshing state... [id=rtb-0584fb744fb428f2c]
aws_lb.alb: Refreshing state... [id=arn:aws:elasticloadbalancing:us-west-2:[MASKED]:loadbalancer/app/dev-webapp-alb/2760d1bce456db0e]
aws_route_table_association.public[0]: Refreshing state... [id=rtbassoc-04bfa84124a68cc7e]
aws_route_table_association.public[1]: Refreshing state... [id=rtbassoc-0a1fec8d91eb09c71]
aws_cloudwatch_metric_alarm.has_traffic: Refreshing state... [id=webapp-ecs-has-traffic]
aws_lb_listener.http: Refreshing state... [id=arn:aws:elasticloadbalancing:us-west-2:[MASKED]:listener/app/dev-webapp-alb/2760d1bce456db0e/a137e3aa2a75f72
f]
aws_ecs_service.webapp: Refreshing state... [id=arn:aws:ecs:us-west-2:[MASKED]:service/dev-webapp-cluster/dev-webapp-service]
aws_lambda_function.ecs_vertical_scaler_up: Refreshing state... [id=ECSVerticalScalerUP]
aws_cloudwatch_metric_alarm.cpu_low: Refreshing state... [id=webapp-ecs-cpu-low]
aws_lambda_function.ecs_vertical_scaler_down: Refreshing state... [id=ECSVerticalScalerDown]
aws_cloudwatch_composite_alarm.scale_down: Refreshing state... [id=webapp-ecs-vertical-scale-down]
aws_cloudwatch_metric_alarm.cpu_alarm: Refreshing state... [id=webapp-ecs-cpu-high]
Terraform used the selected providers to generate the following execution
plan. Resource actions are indicated with the following symbols:
  ~ update in-place
Terraform will perform the following actions:
  # aws_cloudwatch_metric_alarm.cpu_alarm will be updated in-place
  ~ resource "aws_cloudwatch_metric_alarm" "cpu_alarm" {
      ~ alarm_description                     = "This metric monitors high CPU" -> "Scale Up when CPU > 80%"
        id                                    = "webapp-ecs-cpu-high"
        tags                                  = {}
        # (21 unchanged attributes hidden)
    }
Plan: 0 to add, 1 to change, 0 to destroy.
aws_cloudwatch_metric_alarm.cpu_alarm: Modifying... [id=webapp-ecs-cpu-high]
aws_cloudwatch_metric_alarm.cpu_alarm: Modifications complete after 0s [id=webapp-ecs-cpu-high]
Apply complete! Resources: 0 added, 1 changed, 0 destroyed.
Outputs:
ecs_cluster_name = "dev-webapp-cluster"
ecs_service_name = "dev-webapp-service"
load_balancer_dns = "dev-webapp-alb-1840050290.us-west-2.elb.amazonaws.com"
Cleaning up project directory and file based variables                                                              00:00
Job succeeded
```

Figure 17: GitLab Pipeline Deploy_Image Log

Step 4: Monitoring & Logging

4.1 Enable ECS service logs for vertical scaling.

I created a CloudWatch Log Group to monitor and view the ECS service auto scaling logs generated by the Lambda functions.



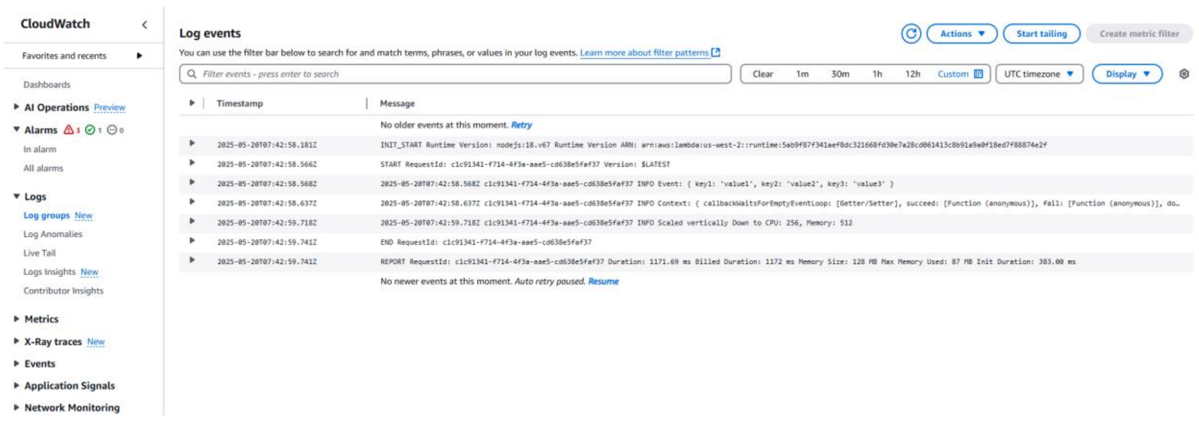
CloudWatch > Log groups > /aws/lambda/ECSVerticalScalerUP > 2025/05/20/[$LATEST]69d282c70cfe414398bae75ada4dc646

Log events

You can use the filter bar below to search for and match terms, phrases, or values in your log events. Learn more about filter patterns

```
Timestamp                  Message
                           No older events at this moment. Retry
2025-05-20T07:40:14.331Z   INIT_START Runtime Version: nodejs:18.v67 Runtime Version ARN: arn:aws:lambda:us-west-2::runtime:5ab9f87f341aef8dc321668fd30e7a28cd061413c8b91a9a0f18ed7f88874e2f
2025-05-20T07:40:14.692Z   START RequestId: 250d6624-ffbf-4ca5-abe0-4aea0dc5c9c5 Version: $LATEST
2025-05-20T07:40:14.693Z   2025-05-20T07:40:14.693Z 250d6624-ffbf-4ca5-abe0-4aea0dc5c9c5 INFO Event: { key1: 'value1', key2: 'value2', key3: 'value3' }
2025-05-20T07:40:14.724Z   2025-05-20T07:40:14.724Z 250d6624-ffbf-4ca5-abe0-4aea0dc5c9c5 INFO Context: { callbackWaitsForEmptyEventLoop: [Getter/Setter], succeed: [Function (anonymous)], fail: [Function (anonymous)], do…
2025-05-20T07:40:15.830Z   2025-05-20T07:40:15.830Z 250d6624-ffbf-4ca5-abe0-4aea0dc5c9c5 INFO Scaled vertically UP to CPU: 512, Memory: 1024
2025-05-20T07:40:15.864Z   END RequestId: 250d6624-ffbf-4ca5-abe0-4aea0dc5c9c5
2025-05-20T07:40:15.864Z   REPORT RequestId: 250d6624-ffbf-4ca5-abe0-4aea0dc5c9c5 Duration: 1153.78 ms Billed Duration: 1154 ms Memory Size: 128 MB Max Memory Used: 88 MB Init Duration: 360.18 ms
                           No newer events at this moment. Auto retry paused. Resume
```

Figure 18: Lambda Log for auto scale up

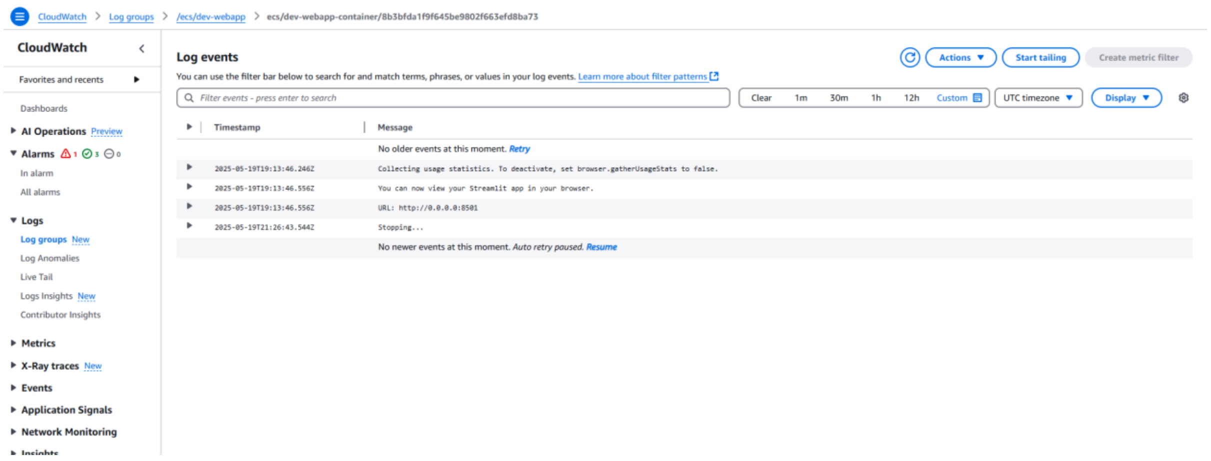


```
Timestamp                  Message
                           No older events at this moment. Retry
2025-05-20T07:42:58.181Z   INIT_START Runtime Version: nodejs:18.v67 Runtime Version ARN: arn:aws:lambda:us-west-2::runtime:5ab9f87f341aef8dc321668fd30e7a28cd061413c8b91a9a0f18ed7f88874e2f
2025-05-20T07:42:58.566Z   START RequestId: c1c91341-f714-4f3a-aae5-cd638e5faf37 Version: $LATEST
2025-05-20T07:42:58.568Z   2025-05-20T07:42:58.568Z c1c91341-f714-4f3a-aae5-cd638e5faf37 INFO Event: { key1: 'value1', key2: 'value2', key3: 'value3' }
2025-05-20T07:42:58.637Z   2025-05-20T07:42:58.637Z c1c91341-f714-4f3a-aae5-cd638e5faf37 INFO Context: { callbackWaitsForEmptyEventLoop: [Getter/Setter], succeed: [Function (anonymous)], fail: [Function (anonymous)], do…
2025-05-20T07:42:59.718Z   2025-05-20T07:42:59.718Z c1c91341-f714-4f3a-aae5-cd638e5faf37 INFO Scaled vertically Down to CPU: 256, Memory: 512
2025-05-20T07:42:59.741Z   END RequestId: c1c91341-f714-4f3a-aae5-cd638e5faf37
2025-05-20T07:42:59.741Z   REPORT RequestId: c1c91341-f714-4f3a-aae5-cd638e5faf37 Duration: 1171.69 ms Billed Duration: 1172 ms Memory Size: 128 MB Max Memory Used: 87 MB Init Duration: 383.00 ms
                           No newer events at this moment. Auto retry paused. Resume
```

Figure 19: Lambda Log for auto scale down

4.2 Capture ECS service logs using AWS CloudWatch.

I created an AWS CloudWatch Log Group for the ECS service to monitor and view its logs



```
Timestamp                  Message
                           No older events at this moment. Retry
2025-05-19T19:13:46.246Z   Collecting usage statistics. To deactivate, set browser.gatherUsageStats to false.
2025-05-19T19:13:46.556Z   You can now view your Streamlit app in your browser.
2025-05-19T19:13:46.556Z   URL: http://0.0.0.0:8501
2025-05-19T21:26:43.544Z   Stopping...
                           No newer events at this moment. Auto retry paused. Resume
```

Figure 20: ECS Service Log