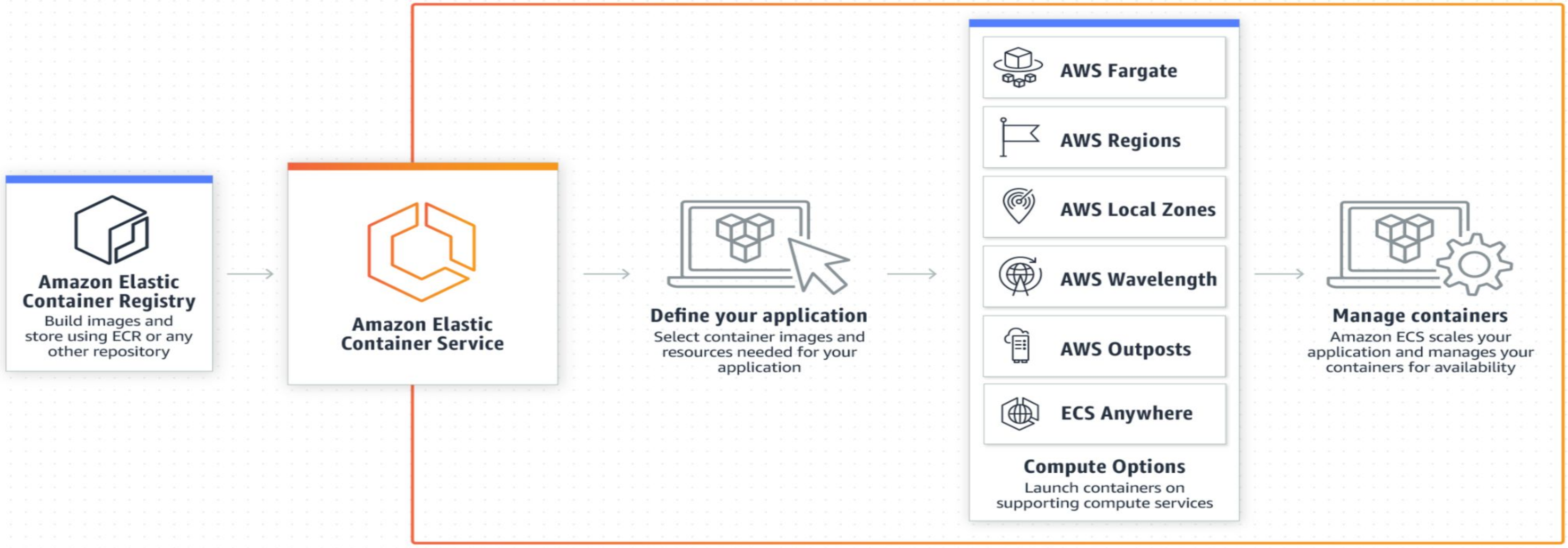
AWS : Serverless
BUILD SERVERLESS APPLICATION ON AWS CLOUD

Presented by :
Amrit Choudhary

ECS with Fargate Setup

ECS : Amazon ECS is a fully managed container orchestration service that makes it easy for you to deploy, manage, and scale containerized applications.

Fargate : It is **a technology that you can use with Amazon ECS to run containers without having to manage servers or clusters of Amazon EC2 instances.**

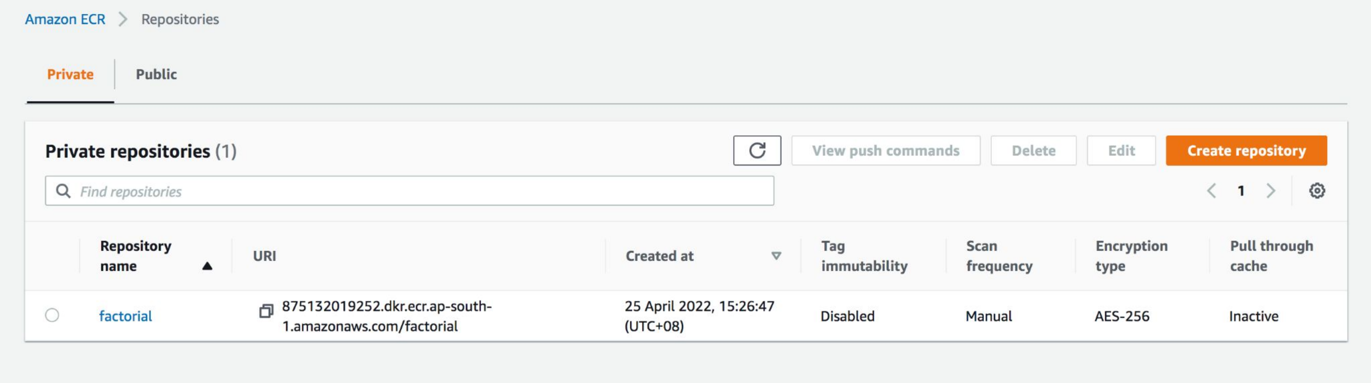


Amazon Elastic Container Registry
Build images and store using ECR or any other repository
Amazon Elastic Container Service
Define your application
Select container images and resources needed for your application
AWS Fargate
AWS Regions
AWS Local Zones
AWS Wavelength
AWS Outposts
ECS Anywhere
Compute Options
Launch containers on supporting compute services
Manage containers
Amazon ECS scales your application and manages your containers for availability

ECS with Fargate Setup : Manual

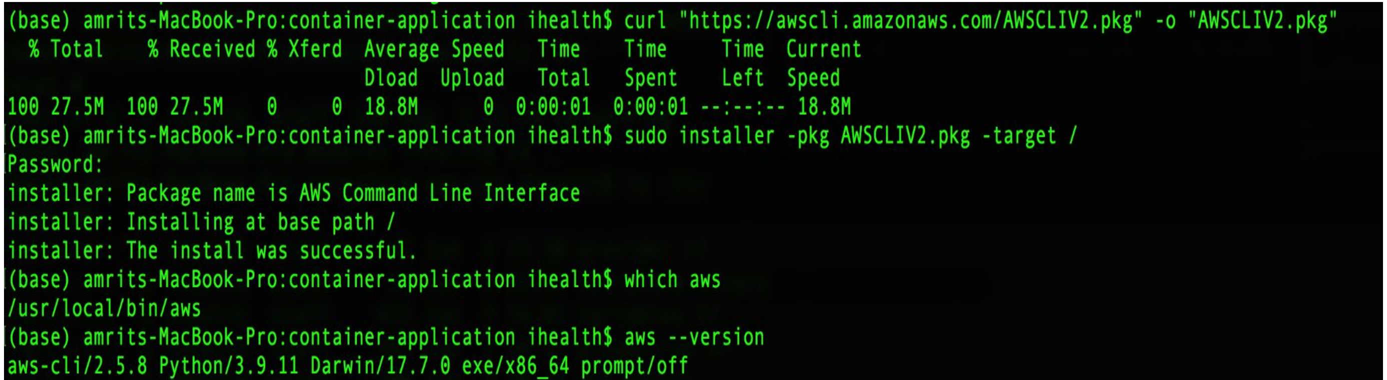
STEPS)

1) Create ECR repo :



	Repository name	URI	Created at	Tag immutability	Scan frequency	Encryption type	Pull through cache
○	factorial	875132019252.dkr.ecr.ap-south-1.amazonaws.com/factorial	25 April 2022, 15:26:47 (UTC+08)	Disabled	Manual	AES-256	Inactive

2) Install aws cli v2



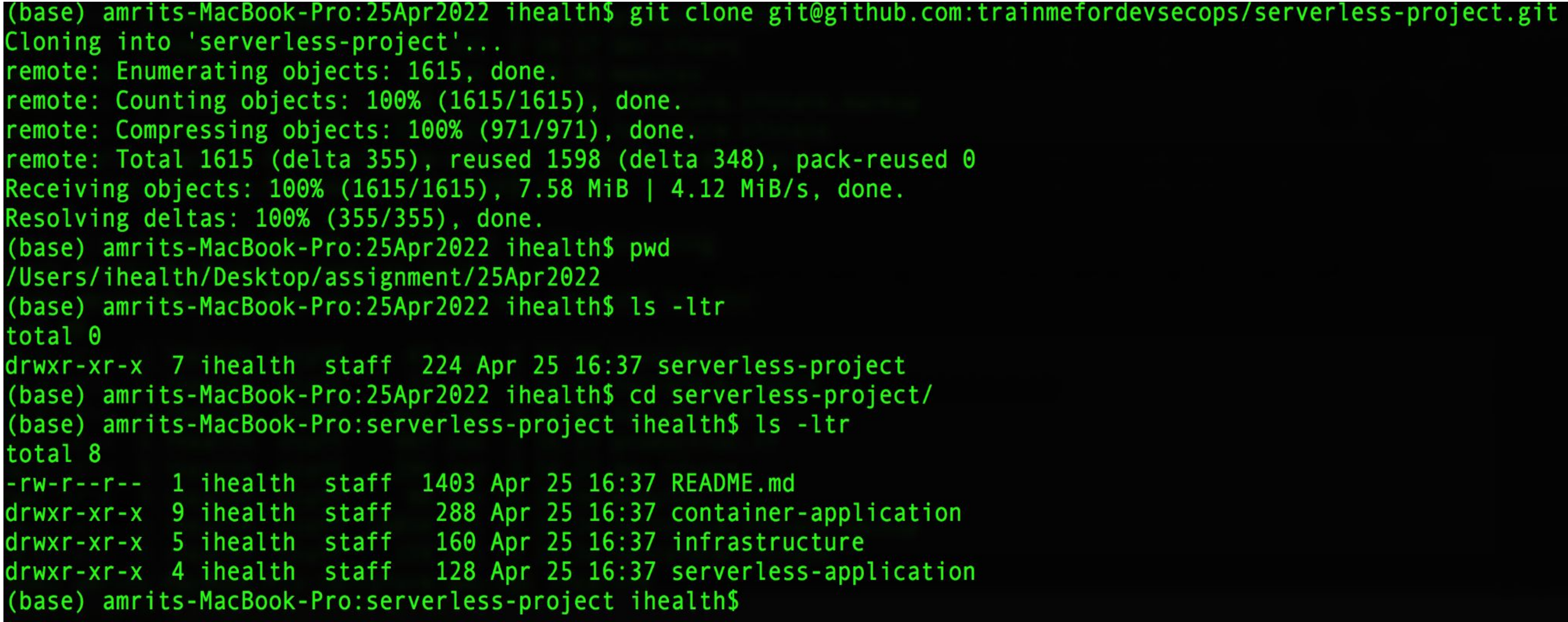
```
(base) amrits-MacBook-Pro:container-application ihealth$ curl "https://awscli.amazonaws.com/AWSCLIV2.pkg" -o "AWSCLIV2.pkg"
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100 27.5M  100 27.5M    0     0  18.8M      0  0:00:01  0:00:01 --:--:-- 18.8M
(base) amrits-MacBook-Pro:container-application ihealth$ sudo installer -pkg AWSCLIV2.pkg -target /
Password:
installer: Package name is AWS Command Line Interface
installer: Installing at base path /
installer: The install was successful.
(base) amrits-MacBook-Pro:container-application ihealth$ which aws
/usr/local/bin/aws
(base) amrits-MacBook-Pro:container-application ihealth$ aws --version
aws-cli/2.5.8 Python/3.9.11 Darwin/17.7.0 exe/x86_64 prompt/off
```

ECS with Fargate Setup : Automated with terraform

STEPS)

Git repository clone

git@github.com:trainmefordevsecops/serverless-project.git



```
(base) amrits-MacBook-Pro:25Apr2022 ihealth$ git clone git@github.com:trainmefordevsecops/serverless-project.git
Cloning into 'serverless-project'...
remote: Enumerating objects: 1615, done.
remote: Counting objects: 100% (1615/1615), done.
remote: Compressing objects: 100% (971/971), done.
remote: Total 1615 (delta 355), reused 1598 (delta 348), pack-reused 0
Receiving objects: 100% (1615/1615), 7.58 MiB | 4.12 MiB/s, done.
Resolving deltas: 100% (355/355), done.
(base) amrits-MacBook-Pro:25Apr2022 ihealth$ pwd
/Users/ihealth/Desktop/assignment/25Apr2022
(base) amrits-MacBook-Pro:25Apr2022 ihealth$ ls -ltr
total 0
drwxr-xr-x  7 ihealth  staff  224 Apr 25 16:37 serverless-project
(base) amrits-MacBook-Pro:25Apr2022 ihealth$ cd serverless-project/
(base) amrits-MacBook-Pro:serverless-project ihealth$ ls -ltr
total 8
-rw-r--r--  1 ihealth  staff  1403 Apr 25 16:37 README.md
drwxr-xr-x  9 ihealth  staff   288 Apr 25 16:37 container-application
drwxr-xr-x  5 ihealth  staff   160 Apr 25 16:37 infrastructure
drwxr-xr-x  4 ihealth  staff   128 Apr 25 16:37 serverless-application
(base) amrits-MacBook-Pro:serverless-project ihealth$
```

ECS with Fargate Setup : Manual

STEPS)

Aws cli v2 steps

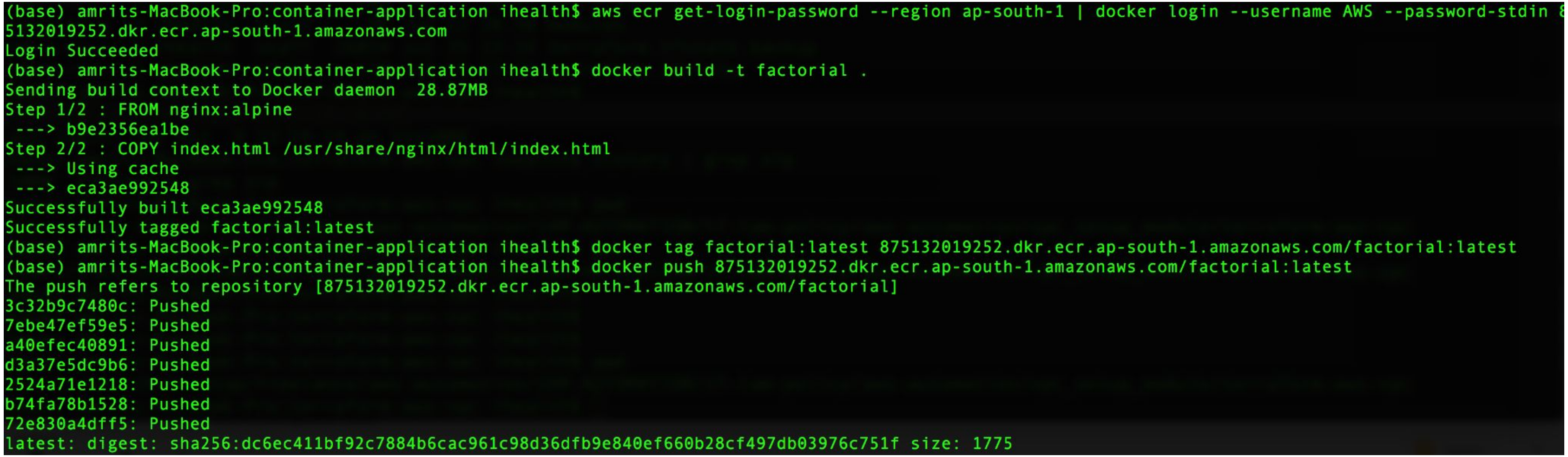
curl "https://awscli.amazonaws.com/AWSCLIV2.pkg" -o "AWSCLIV2.pkg"

sudo installer -pkg AWSCLIV2.pkg -target /

which aws

aws --version

3) Push docker image to ecr repository (using aws cli v2)



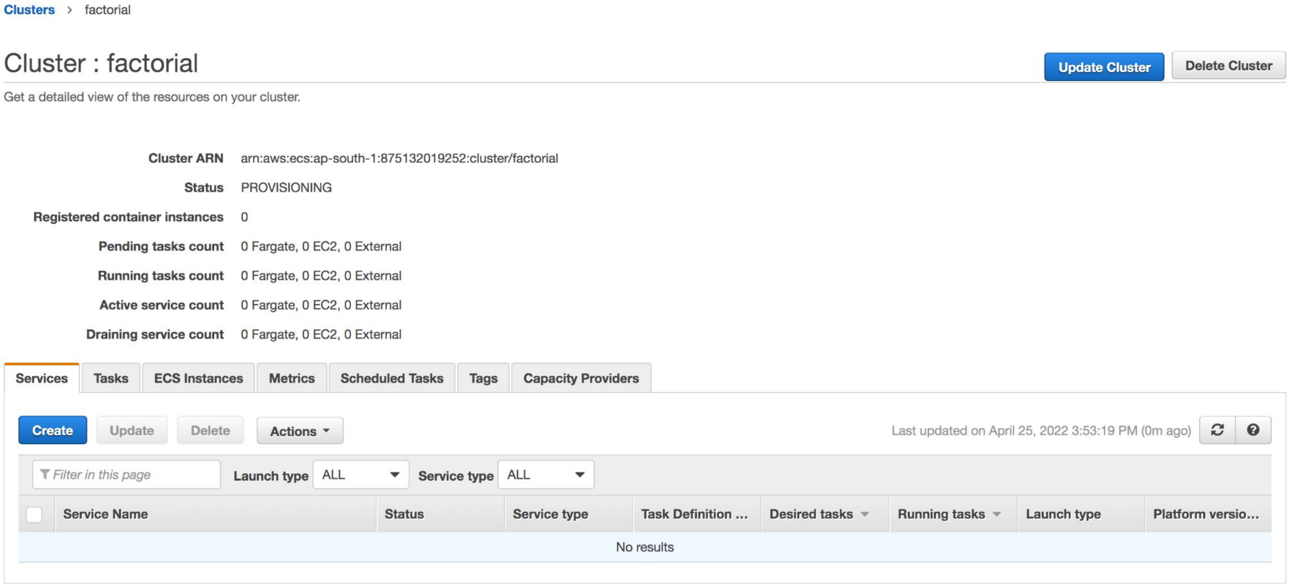
```
(base) amrits-MacBook-Pro:container-application ihealth$ aws ecr get-login-password --region ap-south-1 | docker login --username AWS --password-stdin 8
5132019252.dkr.ecr.ap-south-1.amazonaws.com
Login Succeeded
(base) amrits-MacBook-Pro:container-application ihealth$ docker build -t factorial .
Sending build context to Docker daemon  28.87MB
Step 1/2 : FROM nginx:alpine
 ---> b9e2356ea1be
Step 2/2 : COPY index.html /usr/share/nginx/html/index.html
 ---> Using cache
 ---> eca3ae992548
Successfully built eca3ae992548
Successfully tagged factorial:latest
(base) amrits-MacBook-Pro:container-application ihealth$ docker tag factorial:latest 875132019252.dkr.ecr.ap-south-1.amazonaws.com/factorial:latest
(base) amrits-MacBook-Pro:container-application ihealth$ docker push 875132019252.dkr.ecr.ap-south-1.amazonaws.com/factorial:latest
The push refers to repository [875132019252.dkr.ecr.ap-south-1.amazonaws.com/factorial]
3c32b9c7480c: Pushed
7ebe47ef59e5: Pushed
a40efec40891: Pushed
d3a37e5dc9b6: Pushed
2524a71e1218: Pushed
b74fa78b1528: Pushed
72e830a4dff5: Pushed
latest: digest: sha256:dc6ec411bf92c7884b6cac961c98d36dfb9e840ef660b28cf497db03976c751f size: 1775
```

ECS with Fargate Setup : Manual

3) Push docker image to ecr repository (using aws cli v2)

```
aws ecr get-login-password --region ap-south-1 | docker login --username AWS --password-stdin 875132019252.dkr.ecr.ap-south-1.amazonaws.com

docker build -t factorial .

docker tag factorial:latest 875132019252.dkr.ecr.ap-south-1.amazonaws.com/factorial:latest

docker push 875132019252.dkr.ecr.ap-south-1.amazonaws.com/factorial:latest
```

4) ECS Cluster Setup with Fargate



Cluster : factorial

Get a detailed view of the resources on your cluster.

Cluster ARN	arn:aws:ecs:ap-south-1:875132019252:cluster/factorial
Status	PROVISIONING
Registered container instances	0
Pending tasks count	0 Fargate, 0 EC2, 0 External
Running tasks count	0 Fargate, 0 EC2, 0 External
Active service count	0 Fargate, 0 EC2, 0 External
Draining service count	0 Fargate, 0 EC2, 0 External

Last updated on April 25, 2022 3:53:19 PM (0m ago)

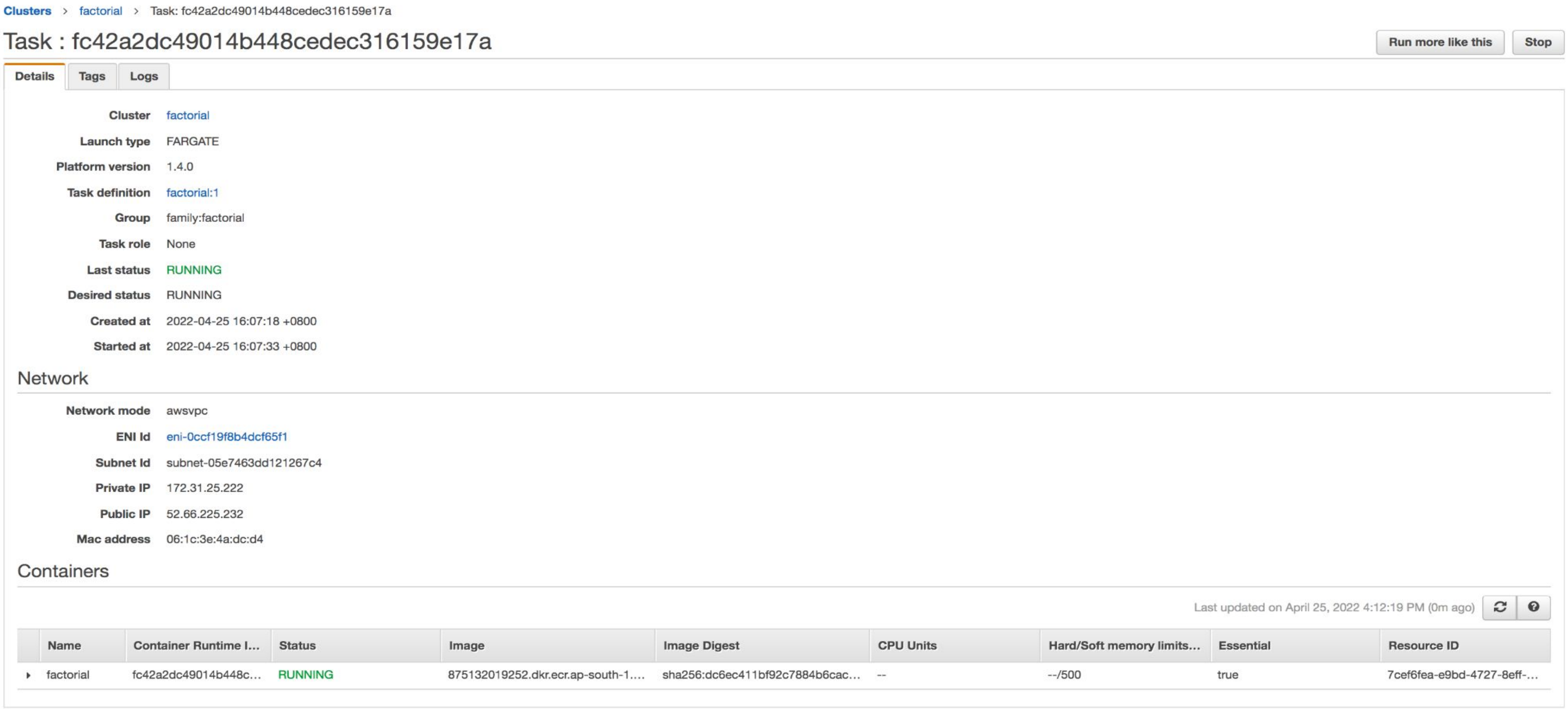
Launch type: ALL

Service type: ALL

□	Service Name	Status	Service type	Task Definition ...	Desired tasks	Running tasks	Launch type	Platform versio...
				No results				

ECS with Fargate Setup : Manual

3) create task definition



Clusters > factorial > Task: fc42a2dc49014b448cedec316159e17a

Task : fc42a2dc49014b448cedec316159e17a

Cluster	factorial
Launch type	FARGATE
Platform version	1.4.0
Task definition	factorial:1
Group	family:factorial
Task role	None
Last status	RUNNING
Desired status	RUNNING
Created at	2022-04-25 16:07:18 +0800
Started at	2022-04-25 16:07:33 +0800

Network

Network mode	awsvpc
ENI Id	eni-0ccf19f8b4dcf65f1
Subnet Id	subnet-05e7463dd121267c4
Private IP	172.31.25.222
Public IP	52.66.225.232
Mac address	06:1c:3e:4a:dc:d4

Containers

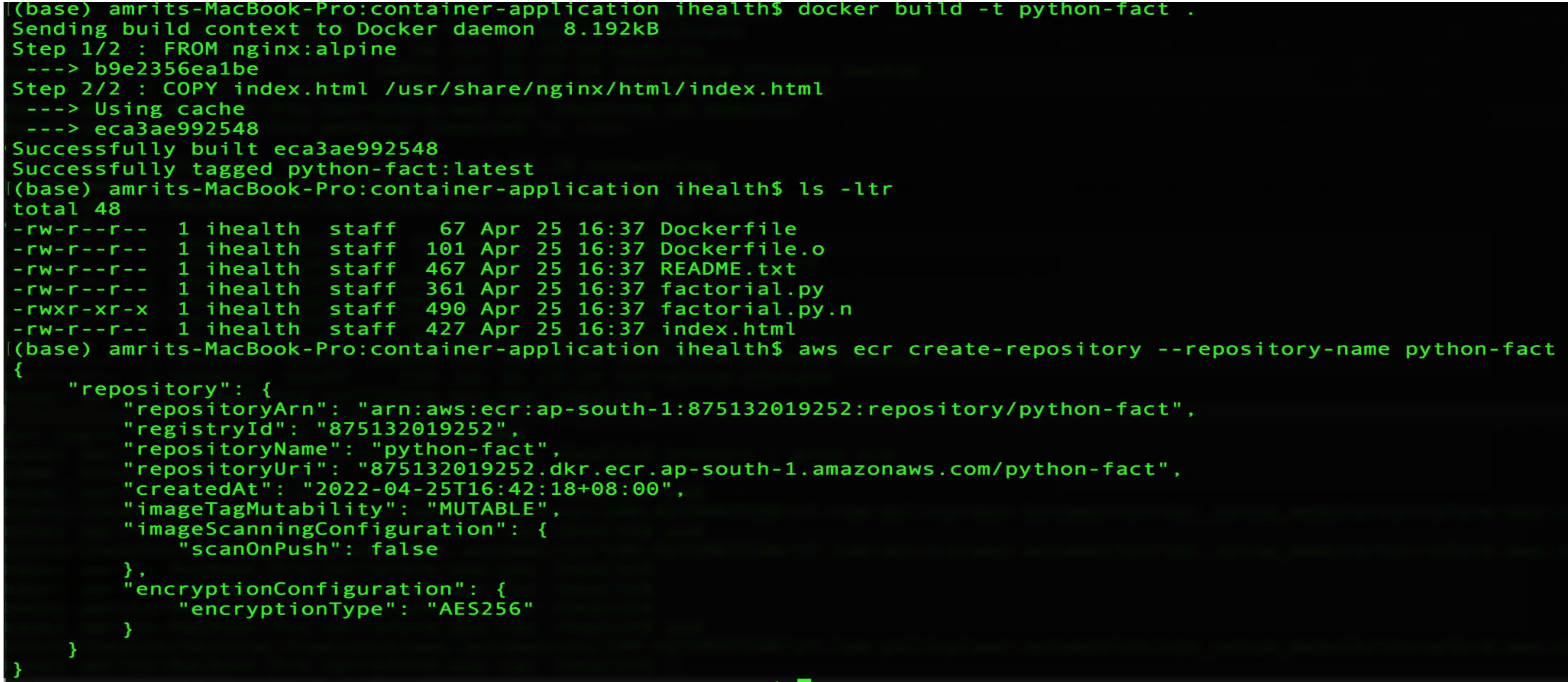
Last updated on April 25, 2022 4:12:19 PM (0m ago)

Name	Container Runtime I...	Status	Image	Image Digest	CPU Units	Hard/Soft memory limits...	Essential	Resource ID
factorial	fc42a2dc49014b448c...	RUNNING	875132019252.dkr.ecr.ap-south-1....	sha256:dc6ec411bf92c7884b6cac...	--	--/500	true	7cef6fea-e9bd-4727-8eff-...

ECS with Fargate Setup : Automated with terraform

STEPS)

1) Git repository clone
2) Build docker container image locally
3) Create factorial repository in aws ecr

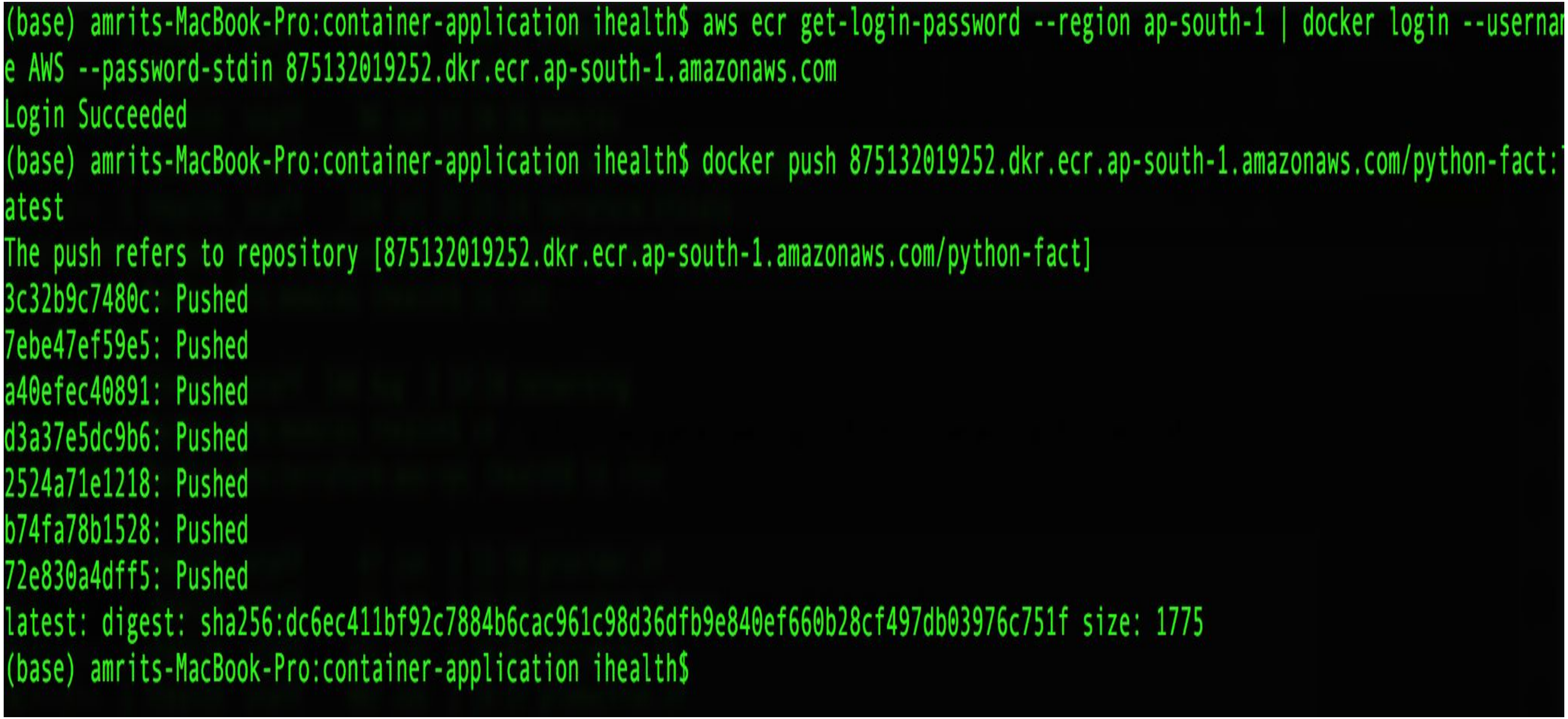


```
(base) amrits-MacBook-Pro:container-application ihealth$ docker build -t python-fact .
Sending build context to Docker daemon  8.192kB
Step 1/2 : FROM nginx:alpine
 ---> b9e2356ea1be
Step 2/2 : COPY index.html /usr/share/nginx/html/index.html
 ---> Using cache
 ---> eca3ae992548
Successfully built eca3ae992548
Successfully tagged python-fact:latest
(base) amrits-MacBook-Pro:container-application ihealth$ ls -ltr
total 48
-rw-r--r--  1 ihealth  staff   67 Apr 25 16:37 Dockerfile
-rw-r--r--  1 ihealth  staff  101 Apr 25 16:37 Dockerfile.o
-rw-r--r--  1 ihealth  staff  467 Apr 25 16:37 README.txt
-rw-r--r--  1 ihealth  staff  361 Apr 25 16:37 factorial.py
-rwxr-xr-x  1 ihealth  staff  490 Apr 25 16:37 factorial.py.n
-rw-r--r--  1 ihealth  staff  427 Apr 25 16:37 index.html
(base) amrits-MacBook-Pro:container-application ihealth$ aws ecr create-repository --repository-name python-fact
{
    "repository": {
        "repositoryArn": "arn:aws:ecr:ap-south-1:875132019252:repository/python-fact",
        "registryId": "875132019252",
        "repositoryName": "python-fact",
        "repositoryUri": "875132019252.dkr.ecr.ap-south-1.amazonaws.com/python-fact",
        "createdAt": "2022-04-25T16:42:18+08:00",
        "imageTagMutability": "MUTABLE",
        "imageScanningConfiguration": {
            "scanOnPush": false
        },
        "encryptionConfiguration": {
            "encryptionType": "AES256"
        }
    }
}
```

ECS with Fargate Setup : Automated with terraform

STEPS)

4) same steps as before in the manual part to push to the ecr repository

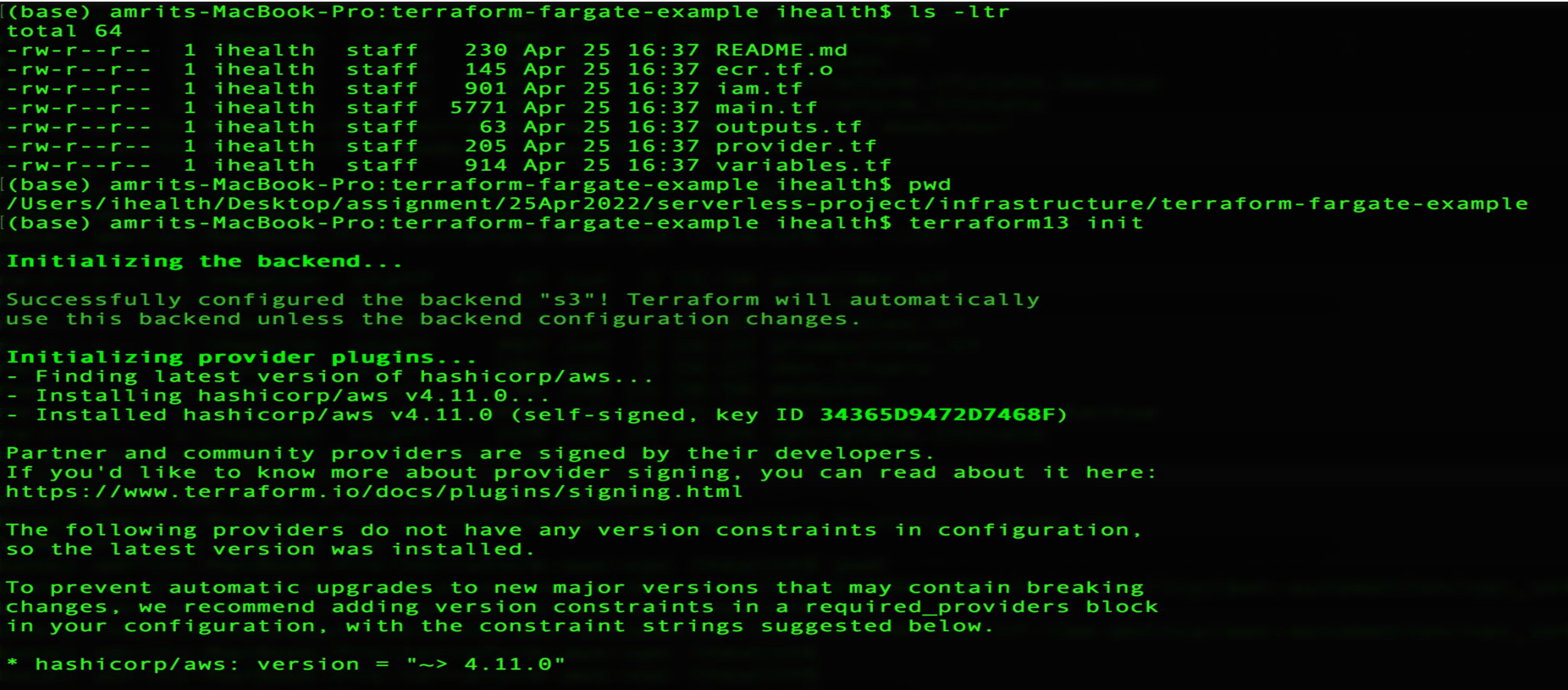


```
(base) amrits-MacBook-Pro:container-application ihealth$ aws ecr get-login-password --region ap-south-1 | docker login --usernam
e AWS --password-stdin 875132019252.dkr.ecr.ap-south-1.amazonaws.com
Login Succeeded
(base) amrits-MacBook-Pro:container-application ihealth$ docker push 875132019252.dkr.ecr.ap-south-1.amazonaws.com/python-fact:
atest
The push refers to repository [875132019252.dkr.ecr.ap-south-1.amazonaws.com/python-fact]
3c32b9c7480c: Pushed
7ebe47ef59e5: Pushed
a40efec40891: Pushed
d3a37e5dc9b6: Pushed
2524a71e1218: Pushed
b74fa78b1528: Pushed
72e830a4dff5: Pushed
latest: digest: sha256:dc6ec411bf92c7884b6cac961c98d36dfb9e840ef660b28cf497db03976c751f size: 1775
(base) amrits-MacBook-Pro:container-application ihealth$
```

ECS with Fargate Setup : Automated with terraform

STEPS)

5) terraform13 init (for provisioning infra , ecs, fargate by switching to terraform-fargate-example directory)

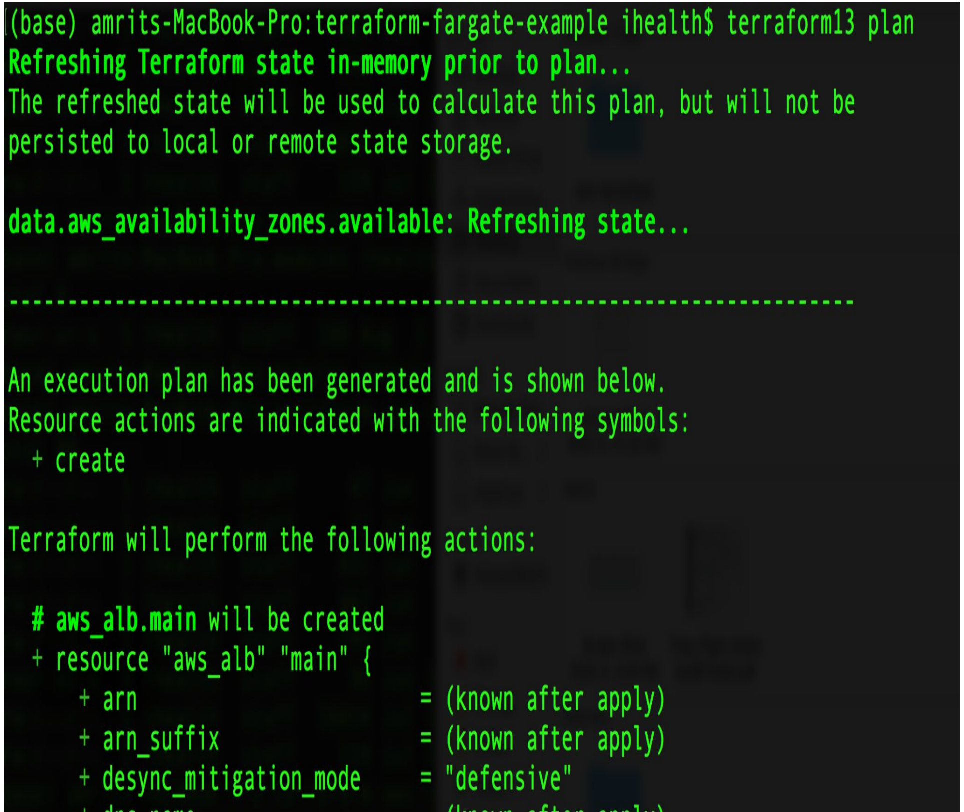


```
(base) amrits-MacBook-Pro:terraform-fargate-example ihealth$ ls -ltr
total 64
-rw-r--r--  1 ihealth  staff   230 Apr 25 16:37 README.md
-rw-r--r--  1 ihealth  staff   145 Apr 25 16:37 ecr.tf.o
-rw-r--r--  1 ihealth  staff   901 Apr 25 16:37 iam.tf
-rw-r--r--  1 ihealth  staff  5771 Apr 25 16:37 main.tf
-rw-r--r--  1 ihealth  staff    63 Apr 25 16:37 outputs.tf
-rw-r--r--  1 ihealth  staff   205 Apr 25 16:37 provider.tf
-rw-r--r--  1 ihealth  staff   914 Apr 25 16:37 variables.tf
(base) amrits-MacBook-Pro:terraform-fargate-example ihealth$ pwd
/Users/ihealth/Desktop/assignment/25Apr2022/serverless-project/infrastructure/terraform-fargate-example
(base) amrits-MacBook-Pro:terraform-fargate-example ihealth$ terraform13 init

Initializing the backend...

Successfully configured the backend "s3"! Terraform will automatically
use this backend unless the backend configuration changes.

Initializing provider plugins...
- Finding latest version of hashicorp/aws...
- Installing hashicorp/aws v4.11.0...
- Installed hashicorp/aws v4.11.0 (self-signed, key ID 34365D9472D7468F)

Partner and community providers are signed by their developers.
If you'd like to know more about provider signing, you can read about it here:
https://www.terraform.io/docs/plugins/signing.html

The following providers do not have any version constraints in configuration,
so the latest version was installed.

To prevent automatic upgrades to new major versions that may contain breaking
changes, we recommend adding version constraints in a required_providers block
in your configuration, with the constraint strings suggested below.

* hashicorp/aws: version = "~> 4.11.0"
```

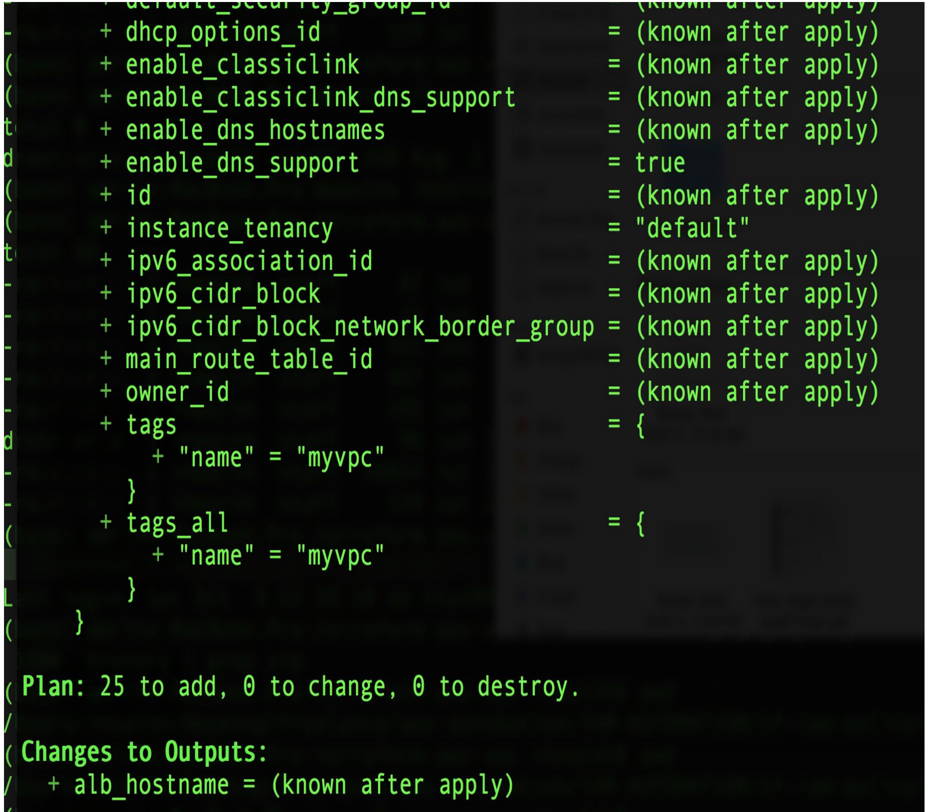
ECS with Fargate Setup : Automated with terraform

STEPS)

5) terraform13 plan (dry run to check infrastructure expected to created, modified, deleted), later terraform13 will run create the resources



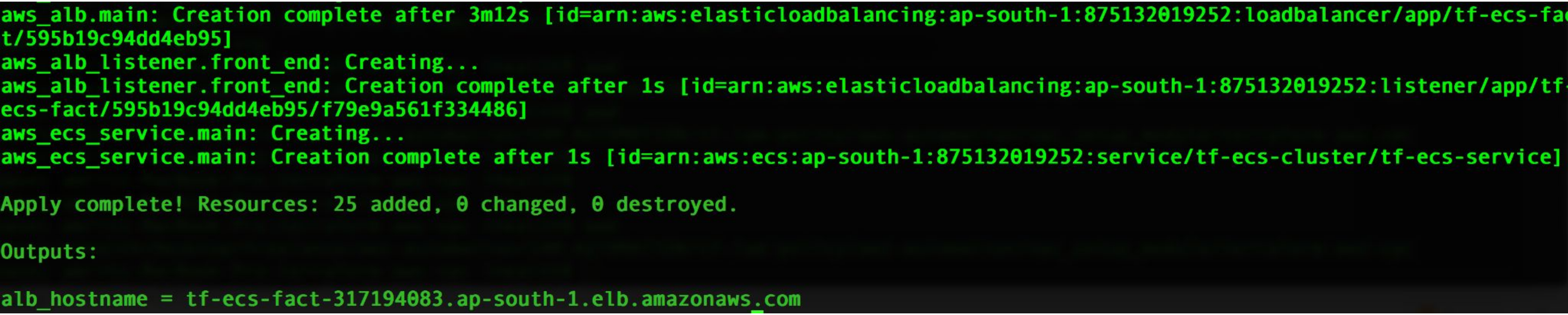
```
(base) amrits-MacBook-Pro:terraform-fargate-example ihealth$ terraform13 plan
Refreshing Terraform state in-memory prior to plan...
The refreshed state will be used to calculate this plan, but will not be
persisted to local or remote state storage.

data.aws_availability_zones.available: Refreshing state...

------------------------------------------------------------------------

An execution plan has been generated and is shown below.
Resource actions are indicated with the following symbols:
  + create

Terraform will perform the following actions:

  # aws_alb.main will be created
  + resource "aws_alb" "main" {
      + arn                        = (known after apply)
      + arn_suffix                 = (known after apply)
      + desync_mitigation_mode     = "defensive"
      + dns_name                   = (known after apply)
```



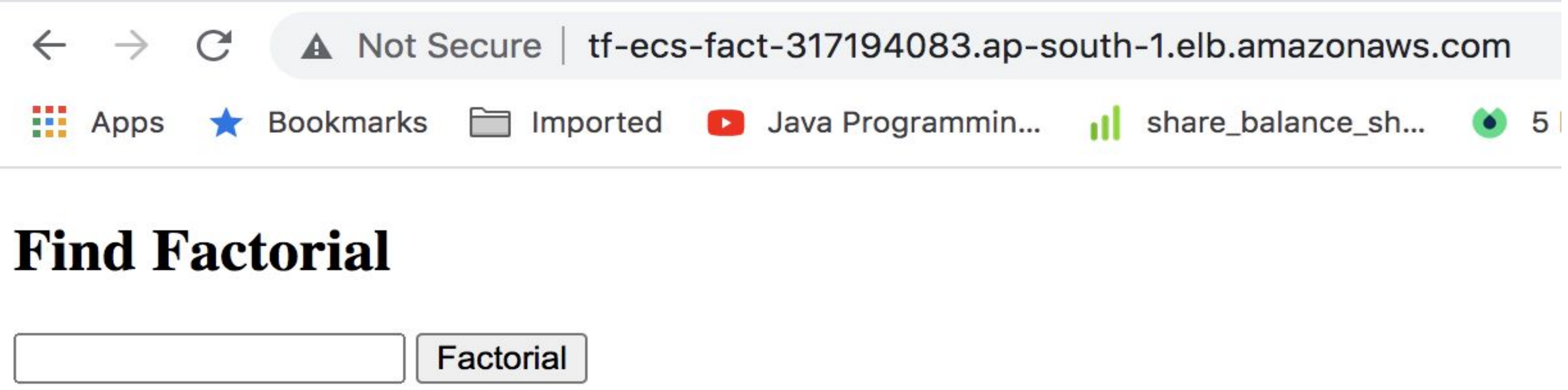
```
      + default_security_group_id                 = (known after apply)
      + dhcp_options_id                           = (known after apply)
      + enable_classiclink                        = (known after apply)
      + enable_classiclink_dns_support            = (known after apply)
      + enable_dns_hostnames                      = (known after apply)
      + enable_dns_support                        = true
      + id                                        = (known after apply)
      + instance_tenancy                          = "default"
      + ipv6_association_id                       = (known after apply)
      + ipv6_cidr_block                           = (known after apply)
      + ipv6_cidr_block_network_border_group = (known after apply)
      + main_route_table_id                       = (known after apply)
      + owner_id                                  = (known after apply)
      + tags                                      = {
          + "name" = "myvpc"
        }
      + tags_all                                  = {
          + "name" = "myvpc"
        }
    }

Plan: 25 to add, 0 to change, 0 to destroy.

Changes to Outputs:
  + alb_hostname = (known after apply)
```

ECS with Fargate Setup : Automated with terraform

OUTPUT



```
aws_alb.main: Creation complete after 3m12s [id=arn:aws:elasticloadbalancing:ap-south-1:875132019252:loadbalancer/app/tf-ecs-fac
t/595b19c94dd4eb95]
aws_alb_listener.front_end: Creating...
aws_alb_listener.front_end: Creation complete after 1s [id=arn:aws:elasticloadbalancing:ap-south-1:875132019252:listener/app/tf-
ecs-fact/595b19c94dd4eb95/f79e9a561f334486]
aws_ecs_service.main: Creating...
aws_ecs_service.main: Creation complete after 1s [id=arn:aws:ecs:ap-south-1:875132019252:service/tf-ecs-cluster/tf-ecs-service]

Apply complete! Resources: 25 added, 0 changed, 0 destroyed.

Outputs:

alb_hostname = tf-ecs-fact-317194083.ap-south-1.elb.amazonaws.com
```



tf-ecs-fact-317194083.ap-south-1.elb.amazonaws.com

Find Factorial

Factorial