

Project

AWS Infrastructure Deployment with Terraform

<https://www.loom.com/share/ba0fcc6f4b1f4fdea2d1391a2cad2d67?sid=5bfcf5aa-d374-49e0-8177-6a564c5eb762>

1. Project Overview

This project involved designing and deploying a scalable, secure, and containerized AWS environment using Terraform. The infrastructure included:

- Auto Scaling EC2 instances running Nginx, Docker, and Node.js.
- Private RDS databases (MySQL & PostgreSQL) with restricted access.
- HTTPS-enabled Application Load Balancer (ALB) for secure traffic routing.
- Business Intelligence (BI) tool deployment (Metabase) for live data visualization.
- Domain & SSL configuration for secure application access.

2. Infrastructure Deployment

2.1 Auto Scaling EC2 Instances

- Deployed three EC2 instances using a Launch Template with Amazon Linux 2.
- Configured user data scripts to auto-install Docker, Nginx, and Node.js.
- Implemented multi-stage Docker containers for frontend and backend applications.

2.2 Secure RDS Databases

- MySQL and PostgreSQL launched in private subnets (no public access).
- Configured security groups to allow only EC2 instances for secure communication.
- Used SSH tunneling to securely connect to databases from local machines.

2.3 Load Balancer & HTTPS Setup

- Deployed an Application Load Balancer (ALB) to distribute traffic across EC2 instances.
- Configured HTTPS (SSL/TLS) using AWS Certificate Manager (ACM).
- Set up HTTP-to-HTTPS redirection for secure connections.

2.4 BI Tool (Metabase) Deployment

- Installed Metabase via Docker on one of the EC2 instances.

3. Challenges Faced & Solutions

3.1 Unused Target Status in ALB

- Problem: The ALB showed "unused" targets even after registering EC2 instances.
- Root Cause:
 - Health check misconfiguration (incorrect path/port).
 - Security group rules blocking ALB traffic.
- Solution:
 - Updated health check settings to /api/health (Node.js) and / (React).
 - Adjusted security groups to allow ALB-to-EC2 communication.
 - Verified container logs to ensure apps were running correctly.

3.2 SSH Tunneling Issues

- Problem: Unable to connect to RDS via SSH tunnel.
- Root Cause:
 - Incorrect SSH key permissions.
 - Security group rules blocking port 22 (SSH).
- Solution:
 - Fixed permissions (chmod 400 key.pem).
 - Updated security groups to allow SSH from trusted IPs.

Some s.s are

```

ummulwara@devopsvm-02:~$ HTTP/1.1 200 OK
-bash: HTTP/1.1: No such file or directory
ummulwara@devopsvm-02:~$ # Check React routes (run on your EC2 instance)
curl -v http://localhost:3000 # Root path
curl -v http://localhost:3000/any-route # Test specific routes
* Host localhost:3000 was resolved.
* IPv6: ::1
* IPv4: 127.0.0.1
* Trying [::1]:3000...
* Connected to localhost (::1) port 3000
> GET / HTTP/1.1
> Host: localhost:3000
> User-Agent: curl/8.5.0
> Accept: */*
>
< HTTP/1.1 200 OK
< Content-Length: 644
< Content-Disposition: inline; filename="index.html"
< Accept-Ranges: bytes
< ETag: "48755a8ded14555da66a9bd9626274a3e065698a"

```

```

Warning: Permanently added '54.81.5.209' (ED25519) to the list of known hosts.
ec2-user@54.81.5.209: Permission denied (publickey,gssapi-keyex,gssapi-with-mic).
ummulwara@devopsvm-02:~$ curl http://localhost
systemctl status nginx # or apache2/httpd,
* nginx.service - A high performance web server and a reverse proxy server
   Loaded: loaded (/usr/lib/systemd/system/nginx.service; enabled; preset: enabled)
   Active: active (running) since Sun 2025-06-15 11:45:15 UTC; 2h 50min ago
     Docs: man:nginx(8)
  Process: 3996683 ExecStartPre=/usr/sbin/nginx -t -q -g daemon on; master_process on; (code=exited, status=0/SUCCESS)
  Process: 3996685 ExecStart=/usr/sbin/nginx -g daemon on; master_process on; (code=exited, status=0/SUCCESS)
 Main PID: 3996686 (nginx)
    Tasks: 5 (limit: 4613)
   Memory: 3.9M (peak: 4.5M)
      CPU: 23ms
   CGroup: /system.slice/nginx.service
           └─3996686 "nginx: master process /usr/sbin/nginx -g daemon on; master_process on;"
             └─3996687 "nginx: worker process"
               └─3996688 "nginx: worker process"
                 └─3996689 "nginx: worker process"
                   └─3996690 "nginx: worker process"
ummulwara@devopsvm-02:~$ curl -v https://ummul-project.apparelcorner.shop
* Host ummul-project.apparelcorner.shop:443 was resolved.
* IPv6: (none)
* IPv4: 52.1.10.17, 54.85.122.124
* Trying 52.1.10.17...

```

```

=> exporting to image
=> == exporting layers
=> == writing image sha256:58e40f4a971c108856778b2814b688d8cfe17b57e085fea57b13b59e3f44566
=> == naming to docker.io/library/nodeapp
0.0s
0.3s
0.0s
0.0s

1 warning found (use docker --debug to expand):
 - JSONArgsRecommended: JSON arguments recommended for CMD to prevent unintended behavior related to OS signals (line 6)
docker: Error response from daemon: Conflict. The container name "/nodeapp-container" is already in use by container "d827260d6867e64c30f1b507eb5035e1f2306aad30b658a255dd2e0b02ebb9df". You have to remove (or rename) that container to be able to reuse that name.

Run 'docker run --help' for more information
ummulwara@devopsvm-02:~/nodejs-1ba$ docker ps -a | grep nodeapp
d827260d6867    96533ebae235    "docker-entrypoint.s..."    4 minutes ago    Up 4 minutes    0.0.0.0:4000->5000/tcp
tcp, [::]:4000->5000/tcp    nodeapp-container
ummulwara@devopsvm-02:~/nodejs-1ba$ docker stop nodeapp-container
docker rm nodeapp-container
nodeapp-container
nodeapp-container
ummulwara@devopsvm-02:~/nodejs-1ba$ nano Dockerfile
ummulwara@devopsvm-02:~/nodejs-1ba$ docker build -t nodeapp .
2025/06/15 20:03:19 in: []string{}
2025/06/15 20:03:19 Parsed entitlements: {}
[+] Building 7.0s (9/9) FINISHED
                                docker:default
=> [internal] load build definition from Dockerfile
=> == transferring dockerfile: 124B
=> [internal] load metadata for docker.io/library/node:latest
=> [internal] load dockerignore
=> == transferring context: 1B
0.0s
0.0s
1.4s
0.0s
0.0s

```

```

* IPy4: 127.0.0.1
* Trying [::1]:4000...
* Connected to localhost (::1) port 4000
> GET /api/health HTTP/1.1
> Host: localhost:4000
> User-Agent: curl/8.5.0
> Accept: */*
>
< HTTP/1.1 200 OK
< X-Powered-By: Express
< Content-Type: application/json; charset=utf-8
< Content-Length: 77
< ETag: W/"4d-PR+LCYh97sx2FpMBSyHdgg9d1EQ"
< Date: Sun, 15 Jun 2025 20:28:58 GMT
< Connection: keep-alive
< Keep-Alive: timeout=5
<
* Connection #8 to host localhost left intact
{"status":"healthy","timestamp":"2025-06-15T20:28:58.949Z","version":"1.0.0"}ummulwara@devopsvm-02:~/nodejs-iba$ sudo netstat -tulnp
sudo netstat -tulnp | grep 4000
[sudo] password for ummulwara:
tcp        0      0 0.0.0.0:4000        0.0.0.0:*           LISTEN      2386812/docker-prox
tcp6       0      0 :::4000            :::*                LISTEN      2386820/docker-prox
ummulwara@devopsvm-02:~/nodejs-iba$ chmod 400 ~/Downloads/my-keypair.pem # Make the key read-only
chmod: cannot access '/home/ummulwara/Downloads/my-keypair.pem': No such file or directory
ummulwara@devopsvm-02:~/nodejs-iba$ ssh -i "path/to/your-key.pem" username@public-ip-or-dns
Warning: Identity file path/to/your-key.pem not accessible: No such file or directory.
ssh: Could not resolve hostname public-ip-or-dns: Temporary failure in name resolution
ummulwara@devopsvm-02:~/nodejs-iba$ java -jar metabase.jar
Error: Unable to access jarfile metabase.jar
ummulwara@devopsvm-02:~/nodejs-iba$

```

Search [Alt+S] United States (N. Virginia) ummulwara_125 @ 7837-6458-7672

Instances

Instances (4) info

Find instance by attribute or tag (case-sensitive) All states

Name	Instance ID	Instance state	Instance type	Status check
web-app-instance	i-0e7048be2ce771a03	Running	t3.micro	5/5 checks passed
project	i-0b3f029b6615fb3c5	Running	t2.micro	2/2 checks passed
web-app-instance	i-0d43555b9dbf0f93f	Running	t3.micro	3/3 checks passed
BI-Tool-Instance	i-0b6f4f40df4108c8b	Running	t3.medium	3/3 checks passed

Select an instance

Classic Load Balancing scales your web balancer capacity automatically in response to changes in incoming traffic.

Filter load balancers

Name	DNS name	State	VPC ID
assignment-web-app-alb	assignment-web-app-alb-2098232784.us-east-1.elb.a...	Active	vpc-066a92a2b6a0fa4d3

Load balancer: assignment-web-app-alb

Details Listeners and rules Network mapping Resource map Security Monitoring Integrations

Details

The image shows a VS Code editor with a Terraform configuration file named `ec2.tf` open. The file defines an AWS instance named `bi_instance` with the following configuration:

```
resource "aws_instance" "bi_instance" {
  ami           = "ami-09e6f87a47903347c"
  instance_type = "t3.xlarge" # Upgraded for BI tool
  subnet_id     = "subnet-8bd554e443dac9cb5"
  key_name       = var.key_name
  vpc_security_group_ids = [aws_security_group.bi_sg.id]

  # Add instance metadata options for security
  metadata_options {
    http_endpoint = "enabled"
    http_tokens   = "required" # IMDSv2 enforced
  }
}
```

The terminal output shows the execution of the Terraform command, indicating that the resources were successfully created:

```
aws_db_instance.postgres: Still creating... [4m0s elapsed]
aws_db_instance.postgres: Still creating... [4m10s elapsed]
aws_db_instance.postgres: Still creating... [4m20s elapsed]
aws_db_instance.postgres: Still creating... [4m30s elapsed]
aws_db_instance.postgres: Creation complete after 4m38s [id=db-MK0EP2EMBMALYEVXVJ2PTSM7U]

Apply complete! Resources: 17 added, 0 changed, 0 destroyed.
```

