Project 7 - Bash Scripting and VM Automation

Introduction

The objective of this project was to create a series of bash scripts that would automate the deployment of a web application and database on AWS. The scripts needed to be idempotent, meaning they could run multiple times without causing issues, and work both as direct execution scripts and as user data for VM initialization.

This solves the problem of having to manually deploy the app and database on AWS. This is a time consuming task that can be automated with a script.

Automation Workflow

TODO: Leaving this space for a diagram of the process.

My automation process followed these stages:

1. Manual Deployment (Initial Testing)
2. Bash Script Development (prov-app.sh & prov-db.sh)
3. User Data Implementation

4. Image Creation
5. Streamlined Deployment (run-app-only.sh)

Methodology

I approached this project in three distinct stages:

- Stage 1 - Script Development:
    - Created prov-db.sh for database provisioning
    - Developed prov-app.sh for application deployment
    - Tested each script independently
    - Ensured idempotency through multiple test runs
- Stage 2 - User Data Integration:
    - Modified scripts to work as user data
    - Adjusted file paths for root execution
    - Added detailed logging for troubleshooting
    - Implemented error handling
- Stage 3 - Image Creation:
    - Created base images from successfully deployed instances
    - Developed run-app-only.sh for quick deployment
    - Tested image-based deployment process
    - Documented recovery procedures

Setting Up Instances (Stage 1)

I utilized AWS infrastructure with the following configuration:

Settings for App Instance:

- Name: **ameenah-sparta-app-vm**
- Instance Type: t2.micro
- Security Groups: Allow SSH, HTTP, and HTTPS
- VPC: Default VPC
- Subnet: Default subnet
- Network: Public subnet for internet access

Settings for DB Instance:

- Name: **ameenah-sparta-db-vm**
- Instance Type: t2.micro

- Security Groups: Allow SSH and port 27017 from App Instance
- VPC: Default VPC
- Subnet: Default subnet
- Network: Private subnet for database security

Network Configuration:

- Default VPC used for both instances
- App instance in public subnet for internet access
- DB instance in private subnet for security
- Security groups configured for minimal required access

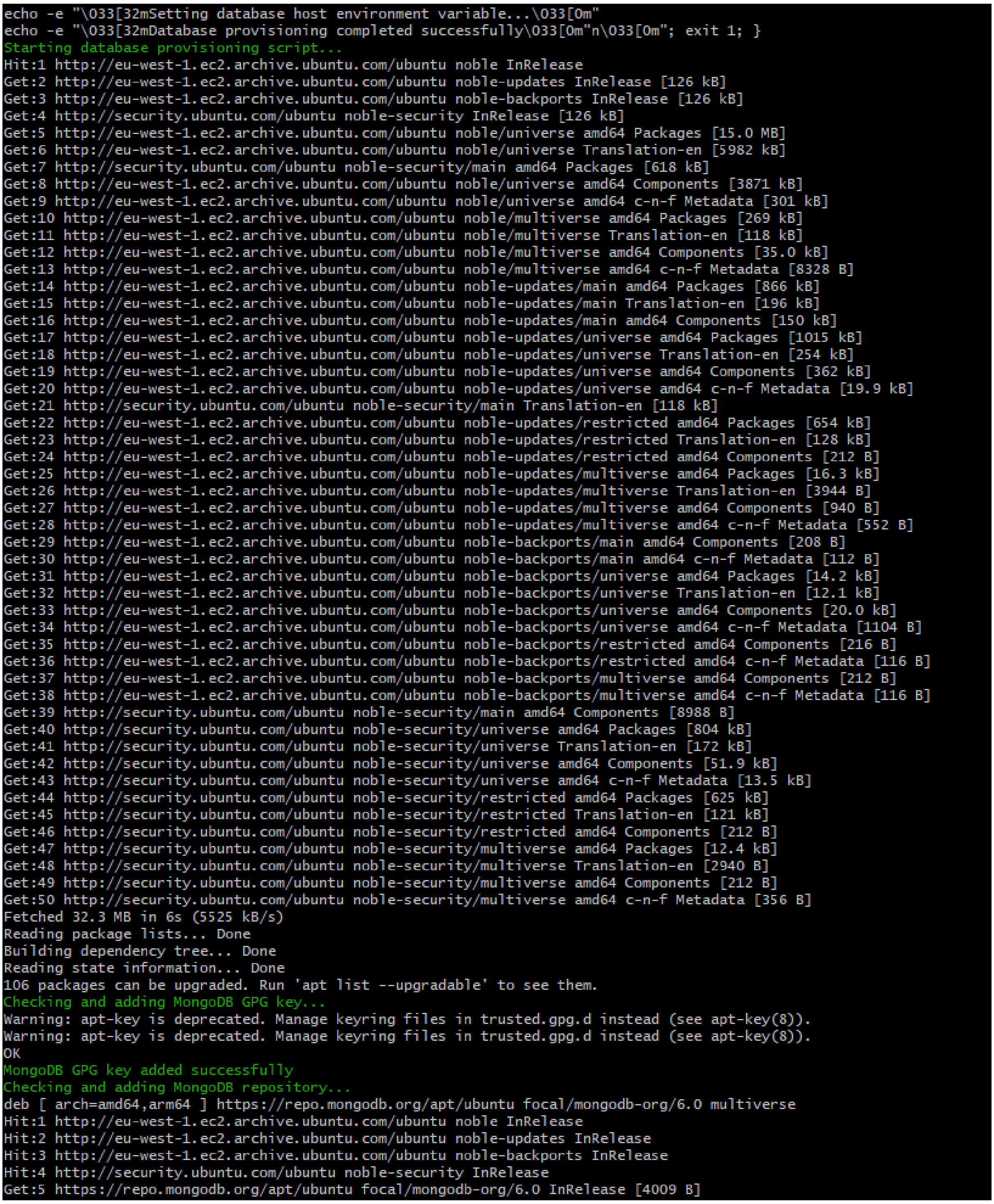
I utilized Terraform to create these VMs to save time when debugging on a multiple of occasions.

Creating the Scripts (Stage 2)

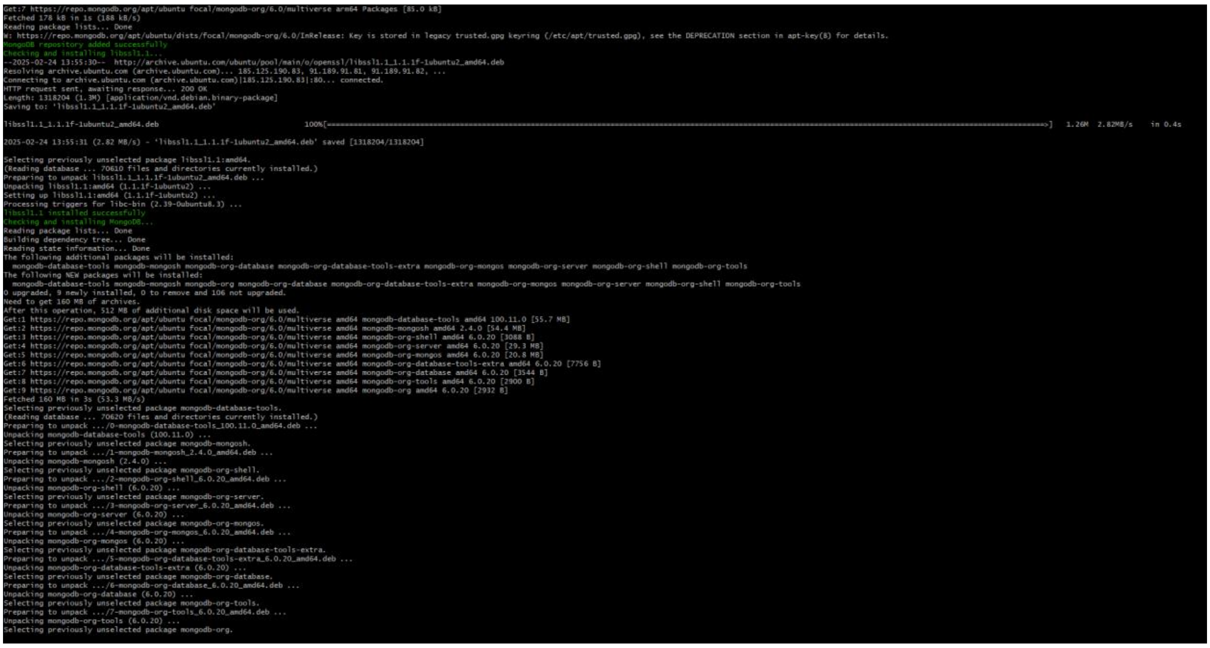
App Bash Script - Fresh Install (prov-app.sh)

I created a bash script with conditions that checks the success and failure of the app deployment script. The actions are displayed with an echo statement in green to differentiate from the rest of the output.

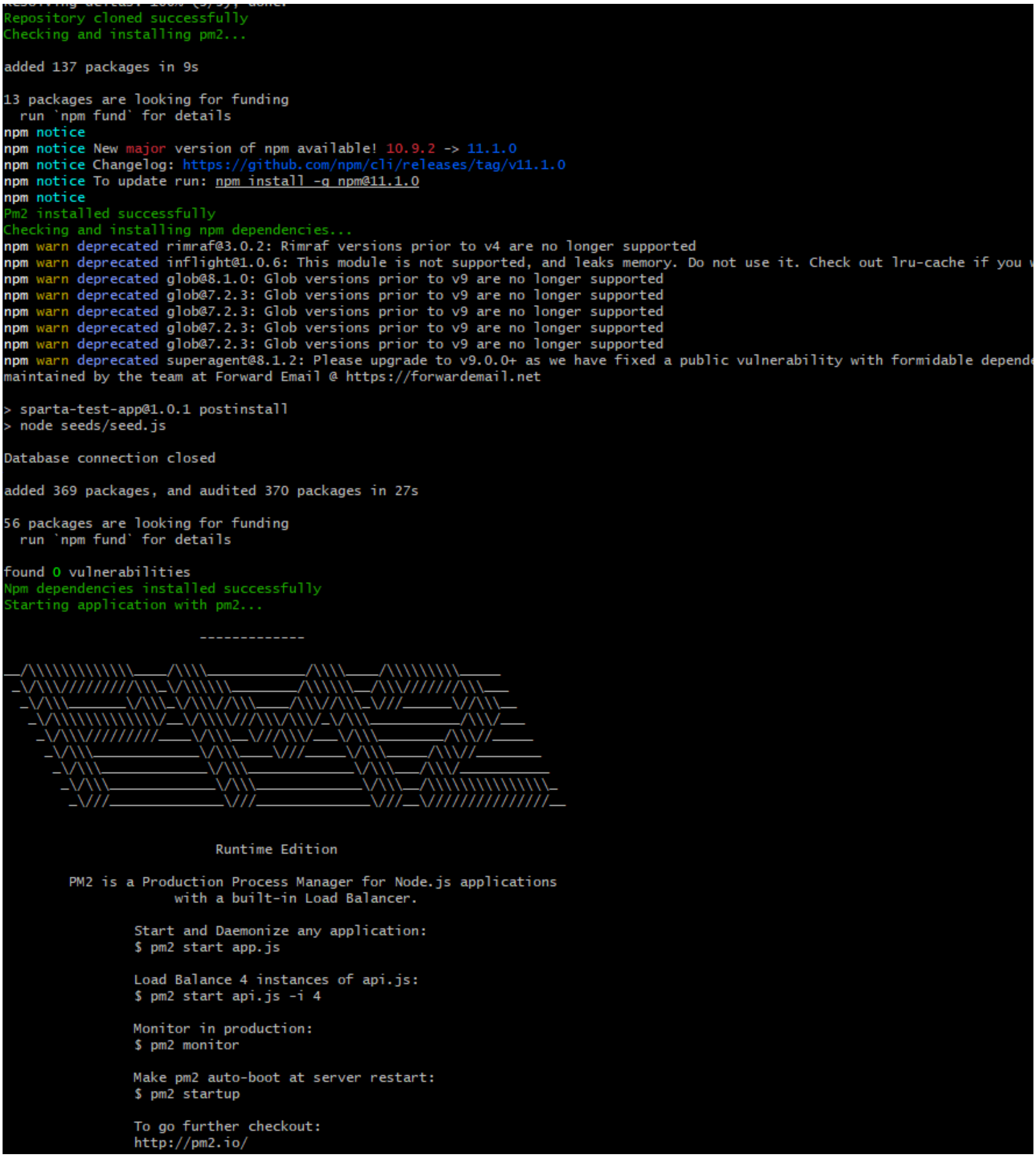
prov-app.sh



```
echo -e "\033[32mSetting database host environment variable...\033[0m"
echo -e "\033[32mDatabase provisioning completed successfully\033[0m"n\033[0m"; exit 1; }
Starting database provisioning script...
Hit:1 http://eu-west-1.ec2.archive.ubuntu.com/ubuntu noble InRelease
Get:2 http://eu-west-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease [126 kB]
Get:3 http://eu-west-1.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease [126 kB]
Get:4 http://security.ubuntu.com/ubuntu noble-security InRelease [126 kB]
Get:5 http://eu-west-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 Packages [15.0 MB]
Get:6 http://eu-west-1.ec2.archive.ubuntu.com/ubuntu noble/universe Translation-en [5982 kB]
Get:7 http://security.ubuntu.com/ubuntu noble-security/main amd64 Packages [618 kB]
Get:8 http://eu-west-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 Components [3871 kB]
Get:9 http://eu-west-1.ec2.archive.ubuntu.com/ubuntu noble/universe amd64 c-n-f Metadata [301 kB]
Get:10 http://eu-west-1.ec2.archive.ubuntu.com/ubuntu noble/multiverse amd64 Packages [269 kB]
Get:11 http://eu-west-1.ec2.archive.ubuntu.com/ubuntu noble/multiverse Translation-en [118 kB]
Get:12 http://eu-west-1.ec2.archive.ubuntu.com/ubuntu noble/multiverse amd64 Components [35.0 kB]
Get:13 http://eu-west-1.ec2.archive.ubuntu.com/ubuntu noble/multiverse amd64 c-n-f Metadata [8328 B]
Get:14 http://eu-west-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 Packages [866 kB]
Get:15 http://eu-west-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main Translation-en [196 kB]
Get:16 http://eu-west-1.ec2.archive.ubuntu.com/ubuntu noble-updates/main amd64 Components [150 kB]
Get:17 http://eu-west-1.ec2.archive.ubuntu.com/ubuntu noble-updates/universe amd64 Packages [1015 kB]
Get:18 http://eu-west-1.ec2.archive.ubuntu.com/ubuntu noble-updates/universe Translation-en [254 kB]
Get:19 http://eu-west-1.ec2.archive.ubuntu.com/ubuntu noble-updates/universe amd64 Components [362 kB]
Get:20 http://eu-west-1.ec2.archive.ubuntu.com/ubuntu noble-updates/universe amd64 c-n-f Metadata [19.9 kB]
Get:21 http://security.ubuntu.com/ubuntu noble-security/main Translation-en [118 kB]
Get:22 http://eu-west-1.ec2.archive.ubuntu.com/ubuntu noble-updates/restricted amd64 Packages [654 kB]
Get:23 http://eu-west-1.ec2.archive.ubuntu.com/ubuntu noble-updates/restricted Translation-en [128 kB]
Get:24 http://eu-west-1.ec2.archive.ubuntu.com/ubuntu noble-updates/restricted amd64 Components [212 B]
Get:25 http://eu-west-1.ec2.archive.ubuntu.com/ubuntu noble-updates/multiverse amd64 Packages [16.3 kB]
Get:26 http://eu-west-1.ec2.archive.ubuntu.com/ubuntu noble-updates/multiverse Translation-en [3944 B]
Get:27 http://eu-west-1.ec2.archive.ubuntu.com/ubuntu noble-updates/multiverse amd64 Components [940 B]
Get:28 http://eu-west-1.ec2.archive.ubuntu.com/ubuntu noble-updates/multiverse amd64 c-n-f Metadata [552 B]
Get:29 http://eu-west-1.ec2.archive.ubuntu.com/ubuntu noble-backports/main amd64 Components [208 B]
Get:30 http://eu-west-1.ec2.archive.ubuntu.com/ubuntu noble-backports/main amd64 c-n-f Metadata [112 B]
Get:31 http://eu-west-1.ec2.archive.ubuntu.com/ubuntu noble-backports/universe amd64 Packages [14.2 kB]
Get:32 http://eu-west-1.ec2.archive.ubuntu.com/ubuntu noble-backports/universe Translation-en [12.1 kB]
Get:33 http://eu-west-1.ec2.archive.ubuntu.com/ubuntu noble-backports/universe amd64 Components [20.0 kB]
Get:34 http://eu-west-1.ec2.archive.ubuntu.com/ubuntu noble-backports/universe amd64 c-n-f Metadata [1104 B]
Get:35 http://eu-west-1.ec2.archive.ubuntu.com/ubuntu noble-backports/restricted amd64 Components [216 B]
Get:36 http://eu-west-1.ec2.archive.ubuntu.com/ubuntu noble-backports/restricted amd64 c-n-f Metadata [116 B]
Get:37 http://eu-west-1.ec2.archive.ubuntu.com/ubuntu noble-backports/multiverse amd64 Components [212 B]
Get:38 http://eu-west-1.ec2.archive.ubuntu.com/ubuntu noble-backports/multiverse amd64 c-n-f Metadata [116 B]
Get:39 http://security.ubuntu.com/ubuntu noble-security/main amd64 Components [8988 B]
Get:40 http://security.ubuntu.com/ubuntu noble-security/universe amd64 Packages [804 kB]
Get:41 http://security.ubuntu.com/ubuntu noble-security/universe Translation-en [172 kB]
Get:42 http://security.ubuntu.com/ubuntu noble-security/universe amd64 Components [51.9 kB]
Get:43 http://security.ubuntu.com/ubuntu noble-security/universe amd64 c-n-f Metadata [13.5 kB]
Get:44 http://security.ubuntu.com/ubuntu noble-security/restricted amd64 Packages [625 kB]
Get:45 http://security.ubuntu.com/ubuntu noble-security/restricted Translation-en [121 kB]
Get:46 http://security.ubuntu.com/ubuntu noble-security/restricted amd64 Components [212 B]
Get:47 http://security.ubuntu.com/ubuntu noble-security/multiverse amd64 Packages [12.4 kB]
Get:48 http://security.ubuntu.com/ubuntu noble-security/multiverse Translation-en [2940 B]
Get:49 http://security.ubuntu.com/ubuntu noble-security/multiverse amd64 Components [212 B]
Get:50 http://security.ubuntu.com/ubuntu noble-security/multiverse amd64 c-n-f Metadata [356 B]
Fetched 32.3 MB in 6s (5525 kB/s)
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
106 packages can be upgraded. Run 'apt list --upgradable' to see them.
Checking and adding MongoDB GPG key...
Warning: apt-key is deprecated. Manage keyring files in trusted.gpg.d instead (see apt-key(8)).
Warning: apt-key is deprecated. Manage keyring files in trusted.gpg.d instead (see apt-key(8)).
OK
MongoDB GPG key added successfully
Checking and adding MongoDB repository...
deb [ arch=amd64,arm64 ] https://repo.mongodb.org/apt/ubuntu focal/mongodb-org/6.0 multiverse
Hit:1 http://eu-west-1.ec2.archive.ubuntu.com/ubuntu noble InRelease
Hit:2 http://eu-west-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease
Hit:3 http://eu-west-1.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease
Hit:4 http://security.ubuntu.com/ubuntu noble-security InRelease
Get:5 https://repo.mongodb.org/apt/ubuntu focal/mongodb-org/6.0 InRelease [4009 B]
```



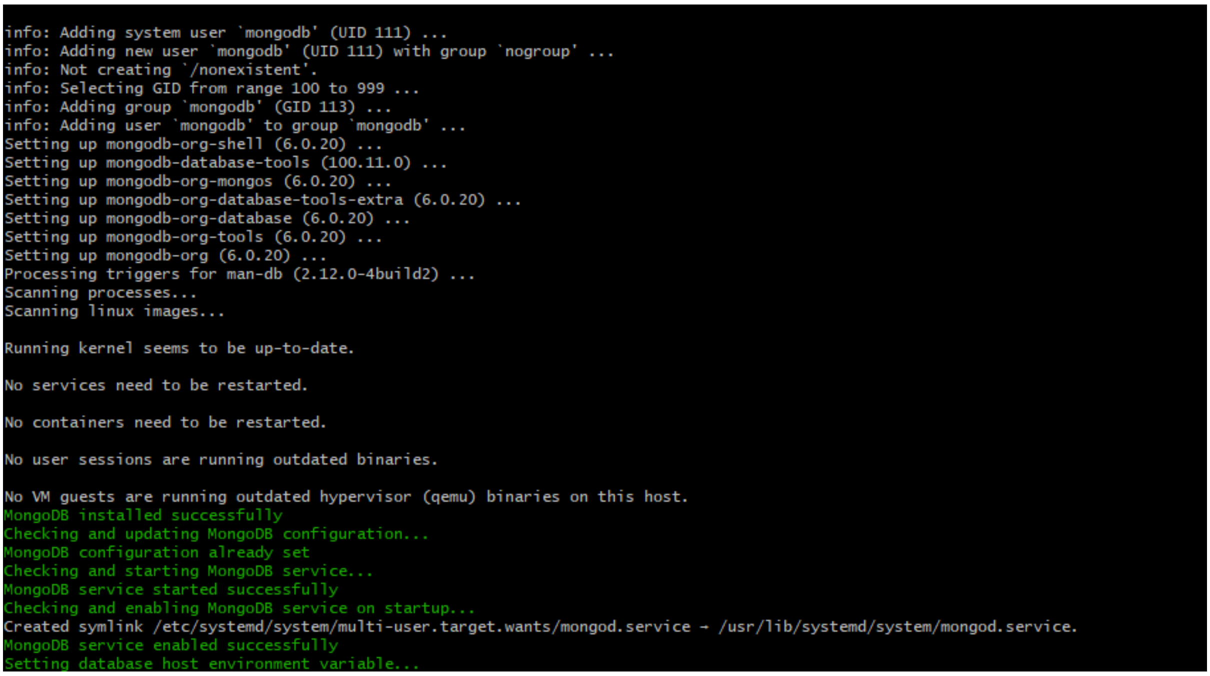
```
Get:7 https://repo.mongodb.org/apt/ubuntu focal/mongodb-org/6.0/multiverse arm64 Packages [85.0 kB]
Fetched 178 kB in 1s (188 kB/s)
Reading package lists... Done
W: https://repo.mongodb.org/apt/ubuntu/dists/focal/mongodb-org/6.0/InRelease: Key is stored in legacy trusted.gpg keyring (/etc/apt/trusted.gpg), see the DEPRECATION section in apt-key(8) for details.
MongoDB repository added successfully
Checking and installing libssl1.1...
--2025-02-24 13:55:30--  http://archive.ubuntu.com/ubuntu/pool/main/o/openssl/libssl1.1_1.1.1f-1ubuntu2_amd64.deb
Resolving archive.ubuntu.com (archive.ubuntu.com)... 185.125.190.83, 91.189.91.81, 91.189.91.82, ...
Connecting to archive.ubuntu.com (archive.ubuntu.com)|185.125.190.83|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 1318204 (1.3M) [application/vnd.debian.binary-package]
Saving to: 'libssl1.1_1.1.1f-1ubuntu2_amd64.deb'

libssl1.1_1.1.1f-1ubuntu2_amd64.deb                        100%[=============================================================================================================================================>]   1.26M  2.82MB/s    in 0.4s

2025-02-24 13:55:31 (2.82 MB/s) - 'libssl1.1_1.1.1f-1ubuntu2_amd64.deb' saved [1318204/1318204]

Selecting previously unselected package libssl1.1:amd64.
(Reading database ... 70610 files and directories currently installed.)
Preparing to unpack libssl1.1_1.1.1f-1ubuntu2_amd64.deb ...
Unpacking libssl1.1:amd64 (1.1.1f-1ubuntu2) ...
Setting up libssl1.1:amd64 (1.1.1f-1ubuntu2) ...
Processing triggers for libc-bin (2.39-0ubuntu8.3) ...
libssl1.1 installed successfully
Checking and installing MongoDB...
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  mongodb-database-tools mongodb-mongosh mongodb-org-database mongodb-org-database-tools-extra mongodb-org-mongos mongodb-org-server mongodb-org-shell mongodb-org-tools
The following NEW packages will be installed:
  mongodb-database-tools mongodb-mongosh mongodb-org mongodb-org-database mongodb-org-database-tools-extra mongodb-org-mongos mongodb-org-server mongodb-org-shell mongodb-org-tools
0 upgraded, 9 newly installed, 0 to remove and 106 not upgraded.
Need to get 160 MB of archives.
After this operation, 512 MB of additional disk space will be used.
Get:1 https://repo.mongodb.org/apt/ubuntu focal/mongodb-org/6.0/multiverse amd64 mongodb-database-tools amd64 100.11.0 [55.7 MB]
Get:2 https://repo.mongodb.org/apt/ubuntu focal/mongodb-org/6.0/multiverse amd64 mongodb-mongosh amd64 2.4.0 [54.4 MB]
Get:3 https://repo.mongodb.org/apt/ubuntu focal/mongodb-org/6.0/multiverse amd64 mongodb-org-shell amd64 6.0.20 [3088 B]
Get:4 https://repo.mongodb.org/apt/ubuntu focal/mongodb-org/6.0/multiverse amd64 mongodb-org-server amd64 6.0.20 [29.3 MB]
Get:5 https://repo.mongodb.org/apt/ubuntu focal/mongodb-org/6.0/multiverse amd64 mongodb-org-mongos amd64 6.0.20 [20.8 MB]
Get:6 https://repo.mongodb.org/apt/ubuntu focal/mongodb-org/6.0/multiverse amd64 mongodb-org-database-tools-extra amd64 6.0.20 [7756 B]
Get:7 https://repo.mongodb.org/apt/ubuntu focal/mongodb-org/6.0/multiverse amd64 mongodb-org-database amd64 6.0.20 [3544 B]
Get:8 https://repo.mongodb.org/apt/ubuntu focal/mongodb-org/6.0/multiverse amd64 mongodb-org-tools amd64 6.0.20 [2900 B]
Get:9 https://repo.mongodb.org/apt/ubuntu focal/mongodb-org/6.0/multiverse amd64 mongodb-org amd64 6.0.20 [2932 B]
Fetched 160 MB in 3s (53.3 MB/s)
Selecting previously unselected package mongodb-database-tools.
(Reading database ... 70620 files and directories currently installed.)
Preparing to unpack .../0-mongodb-database-tools_100.11.0_amd64.deb ...
Unpacking mongodb-database-tools (100.11.0) ...
Selecting previously unselected package mongodb-mongosh.
Preparing to unpack .../1-mongodb-mongosh_2.4.0_amd64.deb ...
Unpacking mongodb-mongosh (2.4.0) ...
Selecting previously unselected package mongodb-org-shell.
Preparing to unpack .../2-mongodb-org-shell_6.0.20_amd64.deb ...
Unpacking mongodb-org-shell (6.0.20) ...
Selecting previously unselected package mongodb-org-server.
Preparing to unpack .../3-mongodb-org-server_6.0.20_amd64.deb ...
Unpacking mongodb-org-server (6.0.20) ...
Selecting previously unselected package mongodb-org-mongos.
Preparing to unpack .../4-mongodb-org-mongos_6.0.20_amd64.deb ...
Unpacking mongodb-org-mongos (6.0.20) ...
Selecting previously unselected package mongodb-org-database-tools-extra.
Preparing to unpack .../5-mongodb-org-database-tools-extra_6.0.20_amd64.deb ...
Unpacking mongodb-org-database-tools-extra (6.0.20) ...
Selecting previously unselected package mongodb-org-database.
Preparing to unpack .../6-mongodb-org-database_6.0.20_amd64.deb ...
Unpacking mongodb-org-database (6.0.20) ...
Selecting previously unselected package mongodb-org-tools.
Preparing to unpack .../7-mongodb-org-tools_6.0.20_amd64.deb ...
Unpacking mongodb-org-tools (6.0.20) ...
Selecting previously unselected package mongodb-org.
```



```
Resolving deltas: 100% (3/3), done.
Repository cloned successfully
Checking and installing pm2...

added 137 packages in 9s

13 packages are looking for funding
  run `npm fund` for details
npm notice
npm notice New major version of npm available! 10.9.2 -> 11.1.0
npm notice Changelog: https://github.com/npm/cli/releases/tag/v11.1.0
npm notice To update run: npm install -g npm@11.1.0
npm notice
Pm2 installed successfully
Checking and installing npm dependencies...
npm warn deprecated rimraf@3.0.2: Rimraf versions prior to v4 are no longer supported
npm warn deprecated inflight@1.0.6: This module is not supported, and leaks memory. Do not use it. Check out lru-cache if you
npm warn deprecated glob@8.1.0: Glob versions prior to v9 are no longer supported
npm warn deprecated glob@7.2.3: Glob versions prior to v9 are no longer supported
npm warn deprecated glob@7.2.3: Glob versions prior to v9 are no longer supported
npm warn deprecated glob@7.2.3: Glob versions prior to v9 are no longer supported
npm warn deprecated glob@7.2.3: Glob versions prior to v9 are no longer supported
npm warn deprecated superagent@8.1.2: Please upgrade to v9.0.0+ as we have fixed a public vulnerability with formidable depend
maintained by the team at Forward Email @ https://forwardemail.net

> sparta-test-app@1.0.1 postinstall
> node seeds/seed.js

Database connection closed

added 369 packages, and audited 370 packages in 27s

56 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities
Npm dependencies installed successfully
Starting application with pm2...

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

__/\\\\\\\\\\\\\____/\\\\____________/\\\\____/\\\\\\\\\_____
 _\/\\\/////////\\\_\/\\\\\\________/\\\\\\__/\\\///////\\\___
  _\/\\\_______\/\\\_\/\\\//\\\____/\\\//\\\_\///______\//\\\__
   _\/\\\\\\\\\\\\\/__\/\\\\///\\\/\\\/_\/\\\___________/\\\/___
    _\/\\\/////////____\/\\\__\///\\\/___\/\\\________/\\\//_____
     _\/\\\_____________\/\\\____\///_____\/\\\_____/\\\//________
      _\/\\\_____________\/\\\_____________\/\\\___/\\\/___________
       _\/\\\_____________\/\\\_____________\/\\\__/\\\\\\\\\\\\\\\_
        _\///______________\///______________\///__\///////////////__


                          Runtime Edition

        PM2 is a Production Process Manager for Node.js applications
                     with a built-in Load Balancer.

                Start and Daemonize any application:
                $ pm2 start app.js

                Load Balance 4 instances of api.js:
                $ pm2 start api.js -i 4

                Monitor in production:
                $ pm2 monitor

                Make pm2 auto-boot at server restart:
                $ pm2 startup

                To go further checkout:
                http://pm2.io/
```

Database Bash Script - Fresh Install (prov-db.sh)

Similarly, one was created for the database provisioning script. prov-db.sh



```
info: Adding system user `mongodb' (UID 111) ...
info: Adding new user `mongodb' (UID 111) with group `nogroup' ...
info: Not creating `/nonexistent'.
info: Selecting GID from range 100 to 999 ...
info: Adding group `mongodb' (GID 113) ...
info: Adding user `mongodb' to group `mongodb' ...
Setting up mongodb-org-shell (6.0.20) ...
Setting up mongodb-database-tools (100.11.0) ...
Setting up mongodb-org-mongos (6.0.20) ...
Setting up mongodb-org-database-tools-extra (6.0.20) ...
Setting up mongodb-org-database (6.0.20) ...
Setting up mongodb-org-tools (6.0.20) ...
Setting up mongodb-org (6.0.20) ...
Processing triggers for man-db (2.12.0-4build2) ...
Scanning processes...
Scanning linux images...

Running kernel seems to be up-to-date.

No services need to be restarted.

No containers need to be restarted.

No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
MongoDB installed successfully
Checking and updating MongoDB configuration...
MongoDB configuration already set
Checking and starting MongoDB service...
MongoDB service started successfully
Checking and enabling MongoDB service on startup...
Created symlink /etc/systemd/system/multi-user.target.wants/mongod.service → /usr/lib/systemd/system/mongod.service.
MongoDB service enabled successfully
Setting database host environment variable...
```

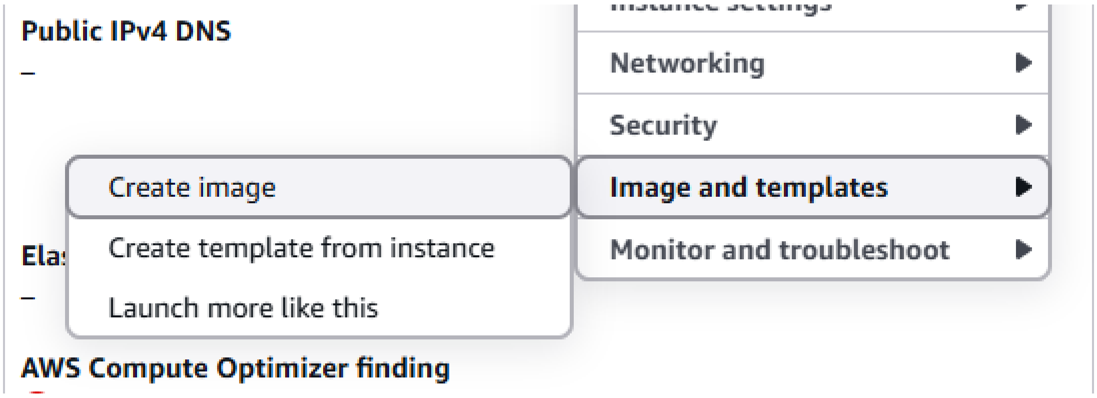
Run App Only Script - Fresh Install (run-app-only.sh)

run-app-only.sh

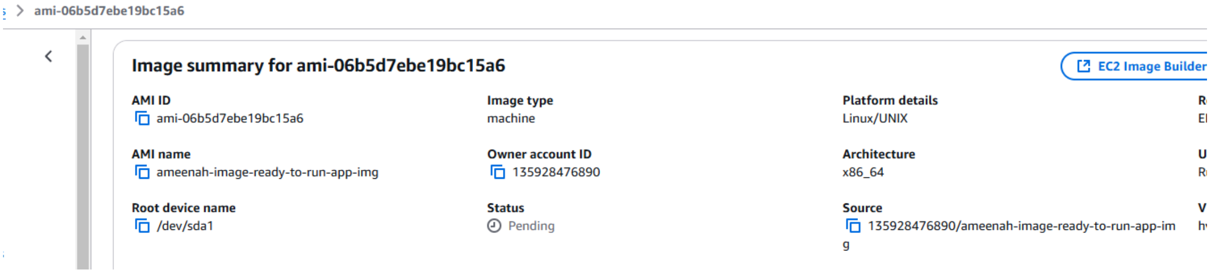
Image Creation (Stage 3)

I created two images from the successful instances. The first image was created from the app instance, and the second from the database instance.

I researched and found that waagent was not needed for AWS. I'll use my app VM as an example here:



All I needed was to make my new VMs using the images I exported, and to use the run-app-only.sh script to deploy the app from the image.



ami-06b5d7ebe19bc15a6

Image summary for ami-06b5d7ebe19bc15a6

EC2 Image Builder

AMI ID
ami-06b5d7ebe19bc15a6

Image type
machine

Platform details
Linux/UNIX

AMI name
ameenah-image-ready-to-run-app-img

Owner account ID
135928476890

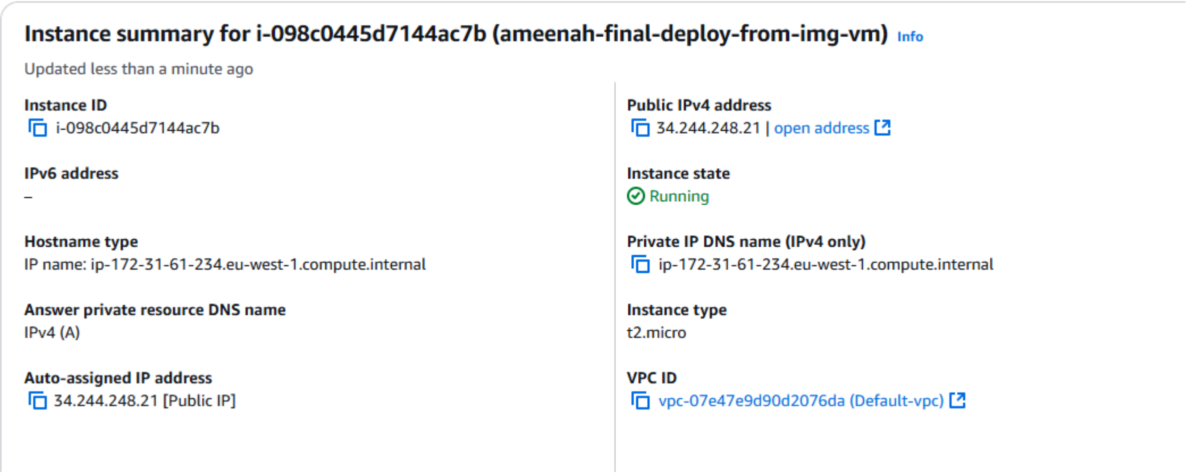
Architecture
x86_64

Root device name
/dev/sda1

Status
Pending

Source
135928476890/ameenah-image-ready-to-run-app-img

This was a successful deployment. A secondary set of VMs were created from the images and the app was deployed from the image. Both VMs were successfully deployed, and the post page worked as expected.



Instance summary for i-098c0445d7144ac7b (ameenah-final-deploy-from-img-vm) Info

Updated less than a minute ago

Instance ID
i-098c0445d7144ac7b

IPv6 address
–

Hostname type
IP name: ip-172-31-61-234.eu-west-1.compute.internal

Answer private resource DNS name
IPv4 (A)

Auto-assigned IP address
34.244.248.21 [Public IP]

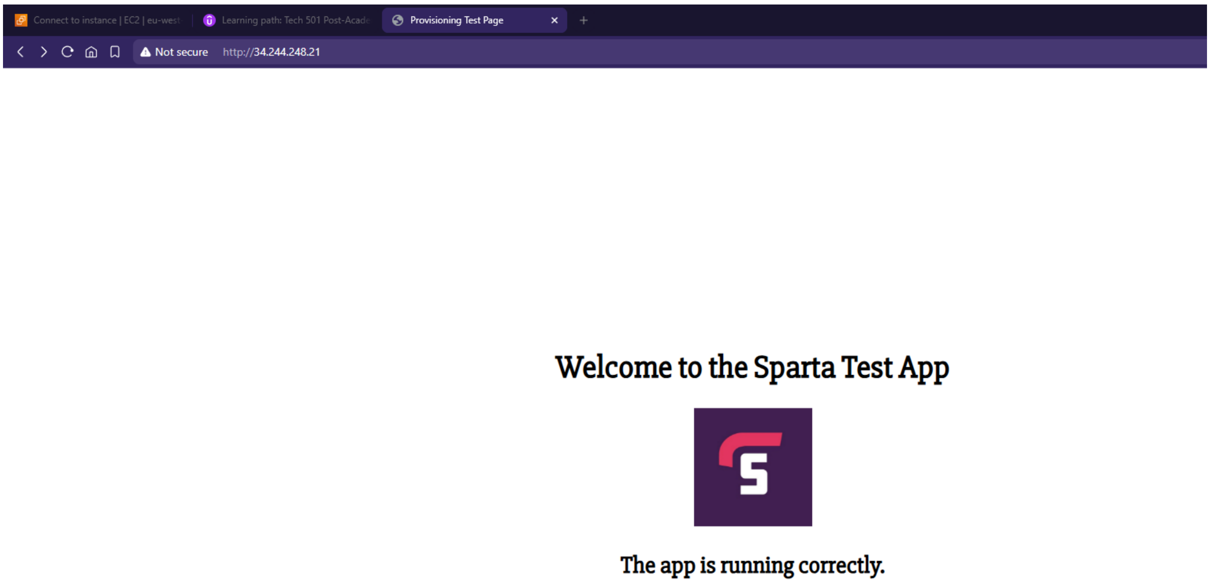
Public IPv4 address
34.244.248.21 | open address

Instance state
Running

Private IP DNS name (IPv4 only)
ip-172-31-61-234.eu-west-1.compute.internal

Instance type
t2.micro

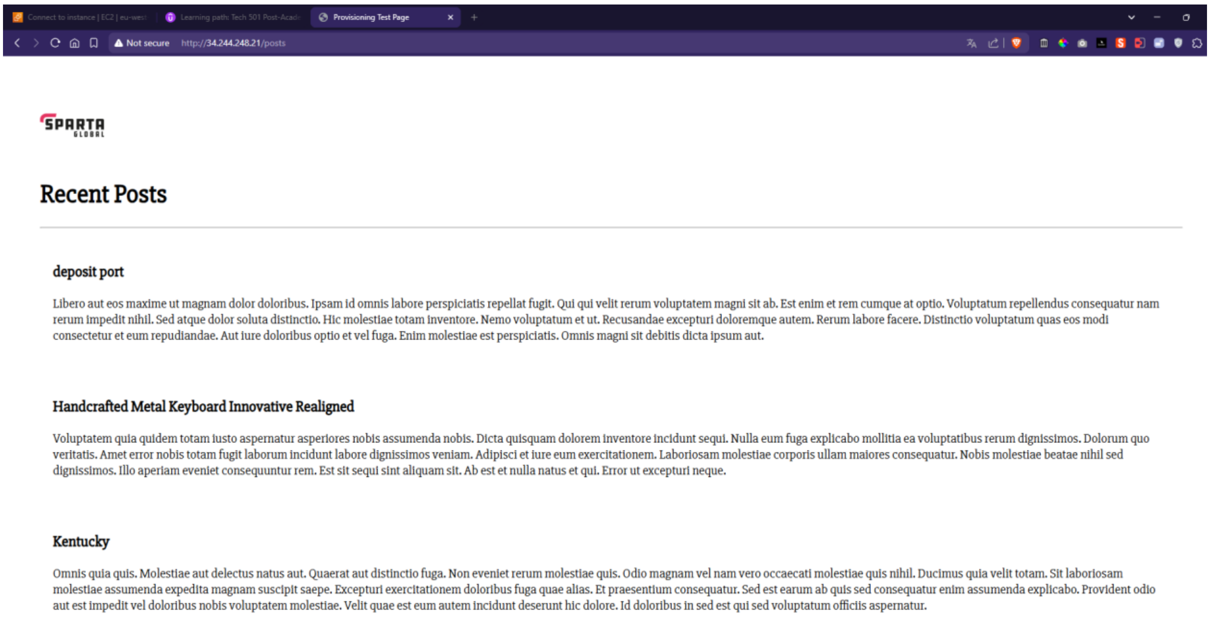
VPC ID
vpc-07e47e9d90d2076da (Default-vpc)



Connect to instance | EC2 | eu-west
Learning path: Tech 501 Post-Acade
Provisioning Test Page
Not secure
http://34.244.248.21

Welcome to the Sparta Test App

The app is running correctly.


Connect to instance | EC2 | eu-west
Learning path: Tech 501 Post-Acad
Provisioning Test Page
Not secure
http://34.244.248.21/posts
SPARTA GLOBAL

Recent Posts

deposit port

Libero aut eos maxime ut magnam dolor doloribus. Ipsam id omnis labore perspiciatis repellat fugit. Qui qui velit rerum voluptatem magni sit ab. Est enim et rem cumque at optio. Voluptatum repellendus consequatur nam rerum impedit nihil. Sed atque dolor soluta distinctio. Hic molestiae totam inventore. Nemo voluptatum et ut. Recusandae excepturi doloremque autem. Rerum labore facere. Distinctio voluptatum quas eos modi consectetur et eum repudiandae. Aut iure doloribus optio et vel fuga. Enim molestiae est perspiciatis. Omnis magni sit debitis dicta ipsum aut.

Handcrafted Metal Keyboard Innovative Realigned

Voluptatem quia quidem totam iusto aspernatur asperiores nobis assumenda nobis. Dicta quisquam dolorem inventore incidunt sequi. Nulla eum fuga explicabo mollitia ea voluptatibus rerum dignissimos. Dolorum quo veritatis. Amet error nobis totam fugit laborum incidunt labore dignissimos veniam. Adipisci et iure eum exercitationem. Laboriosam molestiae corporis ullam maiores consequatur. Nobis molestiae beatae nihil sed dignissimos. Illo aperiam eveniet consequuntur rem. Est sit sequi sint aliquam sit. Ab est et nulla natus et qui. Error ut excepturi neque.

Kentucky

Omnis quia quis. Molestiae aut delectus natus aut. Quaerat aut distinctio fuga. Non eveniet rerum molestiae quis. Odio magnam vel nam vero occaecati molestiae quis nihil. Ducimus quia velit totam. Sit laboriosam molestiae assumenda expedita magnam suscipit saepe. Excepturi exercitationem doloribus fuga quae alias. Et praesentium consequatur. Sed est earum ab quis sed consequatur enim assumenda explicabo. Provident odio aut est impedit vel doloribus nobis voluptatem molestiae. Velit quae est eum autem incidunt deserunt hic dolore. Id doloribus in sed est qui sed voluptatum officiis aspernatur.

Methods of Testing

I implemented a comprehensive testing strategy:

1. Clean Installation Testing
   - Deployed on fresh instances
   - Verified all dependencies
   - Developed and tested scripts
2. Idempotency Testing
   - Multiple script executions
   - Verified no duplicate services
   - Checked for error handling
3. Image Testing
   - Deployed from images
   - Verified app functionality
   - Checked service status
   - Tested user data

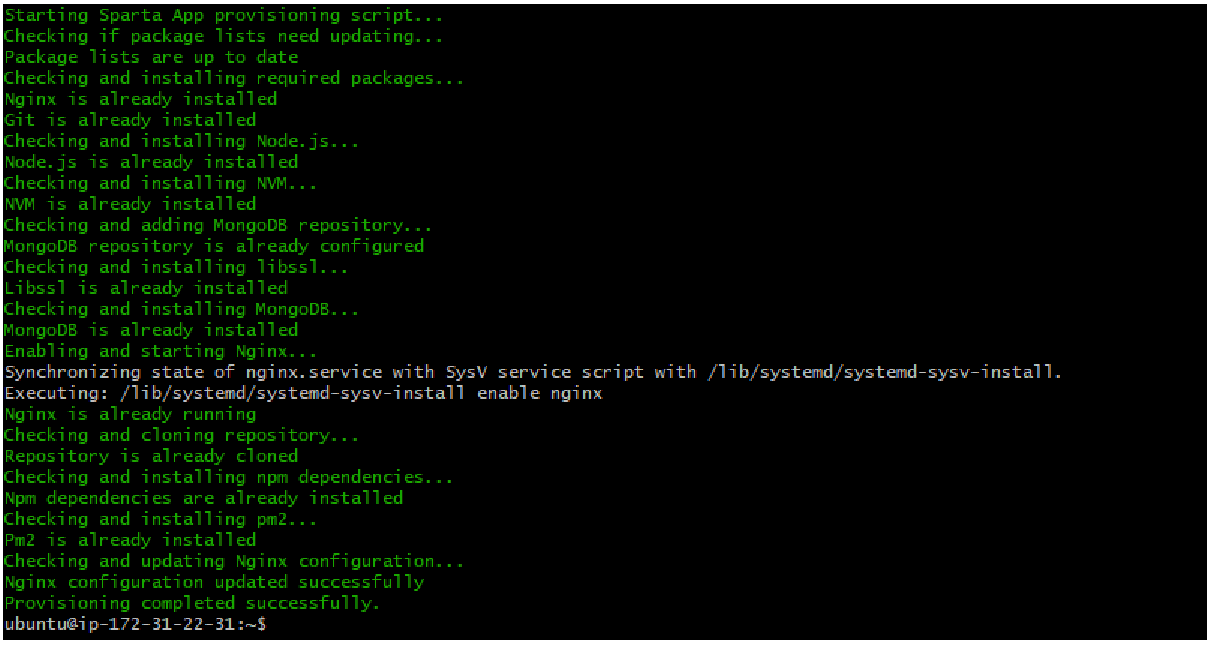
As I wanted to keep an incremental approach, I used Terraform to run the bash scripts on the app and database instances. I created a Terraform script that runs the bash scripts on the instances, this method was useful as it allowed me to test the scripts on the instances and see the results as I went along - correcting any errors as I went.

After each phase was done, I would go back and check to see if the script did what I wanted it to do, a good example of this was checking if the bindip was updating in it's file - this was something that could easily be missed if I was not taking a step back and checking the output.

I backed up my scripts ran them again on clean instances to ensure that the script was idempotent, and again on that VM to see what happened.

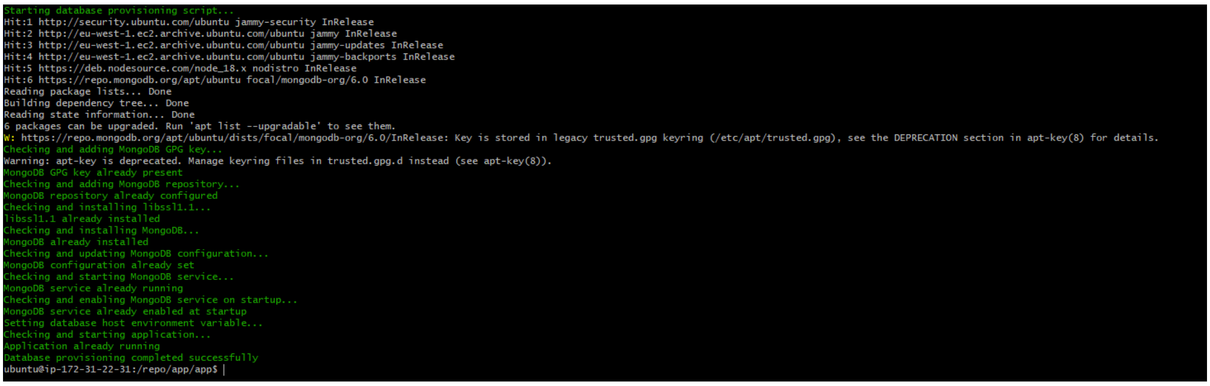
I also used them on my VMs that were deployed from the images to see if they worked.

Testing Idempotency of App script on post-setup:



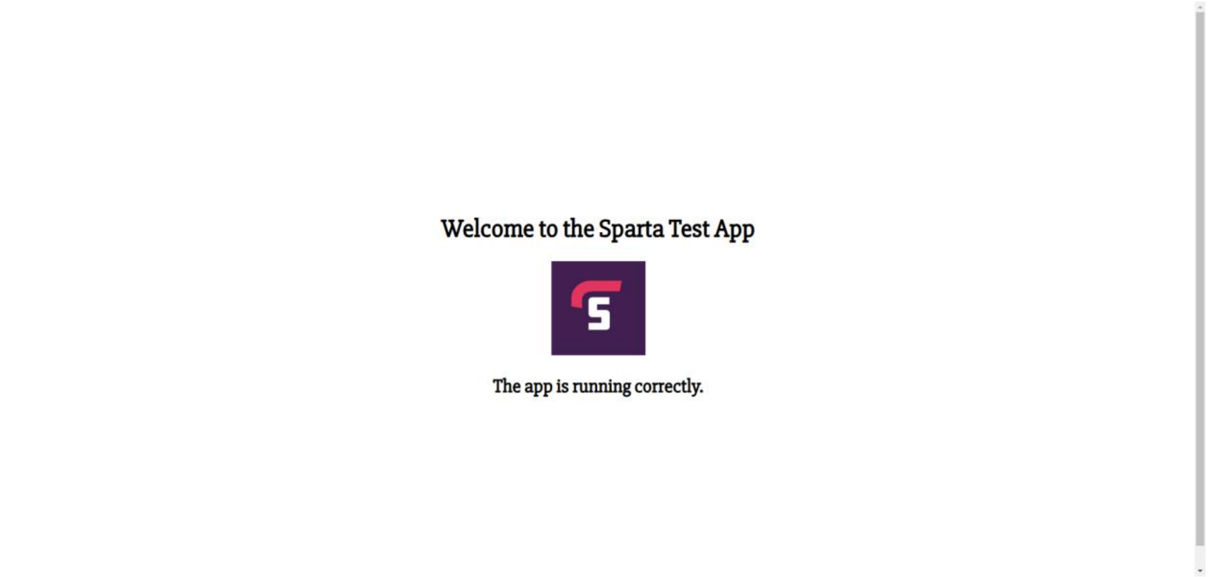
```
Starting Sparta App provisioning script...
Checking if package lists need updating...
Package lists are up to date
Checking and installing required packages...
Nginx is already installed
Git is already installed
Checking and installing Node.js...
Node.js is already installed
Checking and installing NVM...
NVM is already installed
Checking and adding MongoDB repository...
MongoDB repository is already configured
Checking and installing libssl...
Libssl is already installed
Checking and installing MongoDB...
MongoDB is already installed
Enabling and starting Nginx...
Synchronizing state of nginx.service with SysV service script with /lib/systemd/systemd-sysv-install.
Executing: /lib/systemd/systemd-sysv-install enable nginx
Nginx is already running
Checking and cloning repository...
Repository is already cloned
Checking and installing npm dependencies...
Npm dependencies are already installed
Checking and installing pm2...
Pm2 is already installed
Checking and updating Nginx configuration...
Nginx configuration updated successfully
Provisioning completed successfully.
ubuntu@ip-172-31-22-31:~$
```

Testing Idempotency of DB script on post-setup:



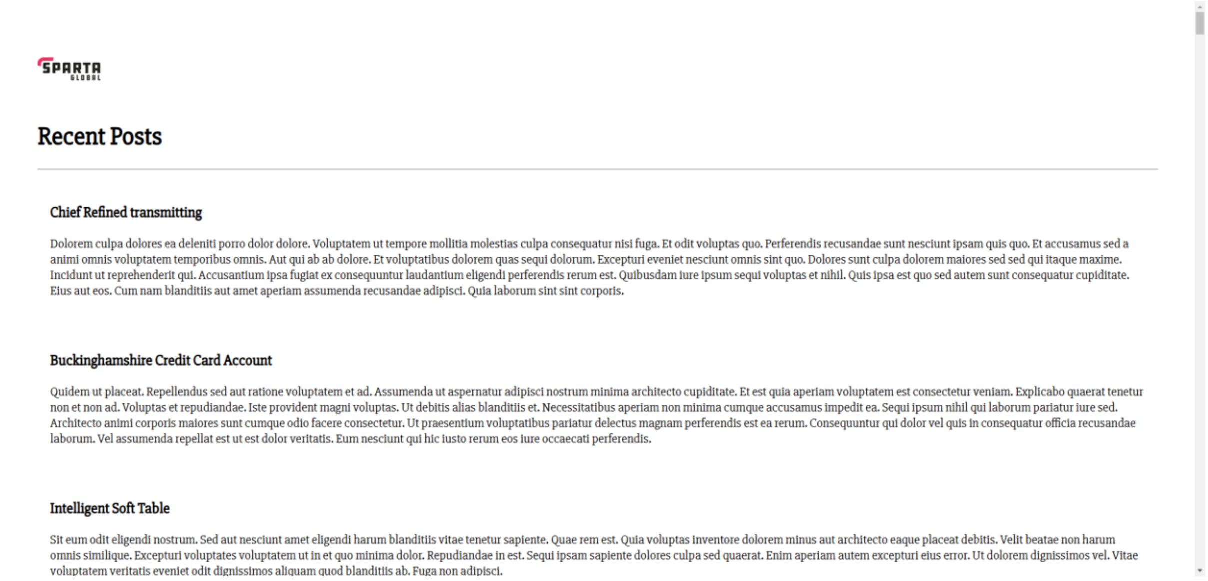
```
Starting database provisioning script...
Hit:1 http://security.ubuntu.com/ubuntu jammy-security InRelease
Hit:2 http://eu-west-1.ec2.archive.ubuntu.com/ubuntu jammy InRelease
Hit:3 http://eu-west-1.ec2.archive.ubuntu.com/ubuntu jammy-updates InRelease
Hit:4 http://eu-west-1.ec2.archive.ubuntu.com/ubuntu jammy-backports InRelease
Hit:5 https://deb.nodesource.com/node_18.x nodistro InRelease
Hit:6 https://repo.mongodb.org/apt/ubuntu focal/mongodb-org/6.0 InRelease
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
6 packages can be upgraded. Run 'apt list --upgradable' to see them.
W: https://repo.mongodb.org/apt/ubuntu/dists/focal/mongodb-org/6.0/InRelease: Key is stored in legacy trusted.gpg keyring (/etc/apt/trusted.gpg), see the DEPRECATION section in apt-key(8) for details.
Checking and adding MongoDB GPG key...
Warning: apt-key is deprecated. Manage keyring files in trusted.gpg.d instead (see apt-key(8)).
MongoDB GPG key already present
Checking and adding MongoDB repository...
MongoDB repository already configured
Checking and installing libssl1.1...
libssl1.1 already installed
Checking and installing MongoDB...
MongoDB already installed
Checking and updating MongoDB configuration...
MongoDB configuration already set
Checking and starting MongoDB service...
MongoDB service already running
Checking and enabling MongoDB service on startup...
MongoDB service already enabled at startup
Setting database host environment variable...
Checking and starting application...
Application already running
Database provisioning completed successfully
ubuntu@ip-172-31-22-31:/repo/app/app$ |
```

Displayed Outcome (Screenshots):



Welcome to the Sparta Test App

The app is running correctly.



Recent Posts

Chief Refined transmitting

Dolorem culpa dolores ea deleniti porro dolor dolore. Voluptatem ut tempore mollitia molestias culpa consequatur nisi fuga. Et odit voluptas quo. Perferendis recusandae sunt nesciunt ipsam quis quo. Et accusamus sed a animi omnis voluptatem temporibus omnis. Aut qui ab ab dolore. Et voluptatibus dolorem quas sequi dolorum. Excepturi eveniet nesciunt omnis sint quo. Dolores sunt culpa dolorem maiores sed sed qui itaque maxime. Incidunt ut reprehenderit qui. Accusantium ipsa fugiat ex consequuntur laudantium eligendi perferendis rerum est. Quibusdam iure ipsum sequi voluptas et nihil. Quis ipsa est quo sed autem sunt consequatur cupiditate. Eius aut eos. Cum nam blanditiis aut amet aperiam assumenda recusandae adipisci. Quia laborum sint sint corporis.

Buckinghamshire Credit Card Account

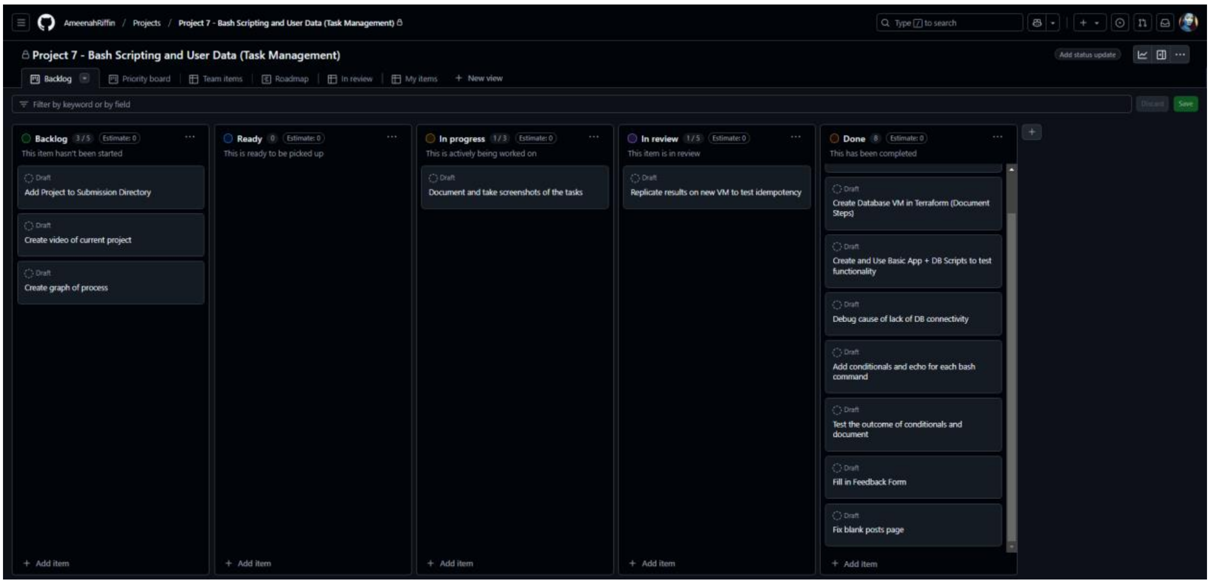
Quidem ut placeat. Repellendus sed aut ratione voluptatem et ad. Assumenda ut aspernatur adipisci nostrum minima architecto cupiditate. Et est quia aperiam voluptatem est consectetur veniam. Explicabo quaerat tenetur non et non ad. Voluptas et repudiandae. Iste provident magni voluptas. Ut debitis alias blanditiis et. Necessitatibus aperiam non minima cumque accusamus impedit ea. Sequi ipsum nihil qui laborum pariatur iure sed. Architecto animi corporis maiores sunt cumque odio facere consectetur. Ut praesentium voluptatibus pariatur delectus magnam perferendis est ea rerum. Consequuntur qui dolor vel quis in consequatur officia recusandae laborum. Vel assumenda repellat est ut est dolor veritatis. Eum nesciunt qui hic iusto rerum eos iure occaecati perferendis.

Intelligent Soft Table

Sit eum odit eligendi nostrum. Sed aut nesciunt amet eligendi harum blanditiis vitae tenetur sapiente. Quae rem est. Quia voluptas inventore dolorem minus aut architecto eaque placeat debitis. Velit beatae non harum omnis similique. Excepturi voluptates voluptatem ut in et quo minima dolor. Repudiandae in est. Sequi ipsam sapiente dolores culpa sed quaerat. Enim aperiam autem excepturi eius error. Ut dolorem dignissimos vel. Vitae voluptatem veritatis eveniet odit dignissimos aliquam quod blanditiis ab. Fuga non adipisci.

Task Management

As part of the project requirements, a task management project was created on the repo to track changes made to the project.



Project 7 - Bash Scripting and User Data (Task Management)

Backlog 3 / 5 Estimate: 0
This item hasn't been started
- Draft: Add Project to Submission Directory
- Draft: Create video of current project
- Draft: Create graph of process

Ready 0 Estimate: 0
This is ready to be picked up

In progress 1 / 3 Estimate: 0
This is actively being worked on
- Draft: Document and take screenshots of the tasks

In review 1 / 5 Estimate: 0
This item is in review
- Draft: Replicate results on new VM to test idempotency

Done 8 Estimate: 0
This has been completed
- Draft: Create Database VM in Terraform (Document Steps)
- Draft: Create and Use Basic App + DB Scripts to test functionality
- Draft: Debug cause of lack of DB connectivity
- Draft: Add conditionals and echo for each bash command
- Draft: Test the outcome of conditionals and document
- Draft: Fill in Feedback Form
- Draft: Fix blank posts page

Blockers

During my task, I encountered blockers and learned a few things that were very helpful in my learning journey.

1. **Understanding root spawned processes:**

While debugging the App VM bash script run from user dataa, I was not able to find the app files nor the PM2 instance. This stopped me from doing an app restart as I intended. I looked into this and realized that if these are installed by root, they will not be visible to the user account I'm using (ubuntu).

I found this solution:

- Find All Node-Related Services
- List all services and look for anything related to Node.js or app.js:

```
systemctl list-units --type=service | grep -i node
```

```
systemctl list-units --type=service | grep -i app
```

Inspect the /etc/.pm2 instance: Since it's running as root, check its process list:

```
sudo PM2_HOME=/etc/.pm2 pm2 list
```

If the app is listed, stop and start it:

```
sudo PM2_HOME=/etc/.pm2 pm2 stop app
```

```
sudo PM2_HOME=/etc/.pm2 pm2 start app
```

Or alternatively, delete it:

```
sudo PM2_HOME=/etc/.pm2 pm2 delete app
```

Additionally, DB_HOSTS needs to be set on root, as well. Unless you make it global.

Root can be accessed with: sudo su

I believe this information should be able help me in future deployments.

2. **Database connection issues:**

I encountered issues with the database connection, which was causing the app to fail. I spent a lot of time trying to fix this, but I was unable to resolve it. Remaking the VMs and subnets was one of the things I tried, which seemed not to work at the time but seemed to resolve it later.

I tried to debug by pinging the database from the app instance, but it was not working. The app and database were on the same subnet, so I thought that the issue was with the security group - however they were configured correctly. IE: TCP 3000, 22, and 80 were open for the app. Database was open for MongoDB's outbound port 27017 and inbound SSH (which I was able to access).

I looked further into this and found that the issue was with bindip not being set correctly in my userdata's sed command, along with needing to restart the app. There may have been other causes for my trouble, but I did not have enough time to do a full investigation.

I plan to do a more thorough investigation into that and document my findings.

Benefits and Learning Outcomes

Throughout this project, I developed my skills in bash scripting while creating deployment scripts.

The challenges I encountered, particularly with user permissions and database connectivity gave me a better insight on maintaining and debugging production environments.

Conclusion

I like this project. I learned a lot about keeping a consistent process with incremental approaches to building a bash script deploy a web application and database on AWS.

The scripts and images created provide a reliable, repeatable deployment process that significantly reduces manual intervention and potential human error.