Project 7 - Use Scripting and User Data for 2-Tier App Deployment

contents:

What problem is this task solving

- as a part of changing and testing the app a developer would have to set up the app with the required dependency each time
- this can be a time-consuming process and take up time
- devs will want to ensure the same dependency are present each time and that the set up is uniform and not prone to human error

Automation workflow for this project

1. manual app and database deployment - setting up using command line
   - this is a good way to test the setup steps one by one and make a script that does not require user input
   - can be used to debug and make sure all the necessary commands are set up
2. Bash scripting
   - taking the above steps used in command line to develop a script that can be run within that virtual machine
   - making sure all commands can be run using linux - in my case replacing text using `sed` instead of manually using `nano`
3. User data
   - taking the scripts used to set up virtual machine and putting it into the user data field of the VM setup
   - means that the setup does not require ssh access to a virtual machine, which will increase security as this can be removed from security groups once the scripts have been tested
   - this is the place where using `sudo` for the `npm install` step can cause issues
4. making images and a `run-app-only.sh` script
   - the setup for both the db and app vm are almost identical each time, the only functional difference being the ip of the mongo db connection
   - this means images can be utalised to provide a fully provisioned app and db
   - then a `run-app-only.sh` script can be utalised on the app set up to provide the `DB_HOST` variable

Manual deployment

General points:

- whilst setting up this manually the aim is to make sure there are no instances were user input is required, as the end goal is to have this running in a script
- a number of install commands often require user input, but this can be dealt with using `DEBIAN_FRONTEND=noninteractive` variable with `apt-get`
  - if used consistently this can run exported as a variable `export DEBIAN_FRONTEND=noninteractive`
  - `-y` flag can also be used to give default of 'yes' were user input is required

db deployment

```
#update and upgrade packages without asking for user input
sudo DEBIAN_FRONTEND=noninteractive apt-get update && sudo DEBIAN_FRONTEND=noninteractive apt-get dist-upgrade -y
#install mongo db
sudo apt-get install gnupg curl
# import gpg key
curl -fsSL https://www.mongodb.org/static/pgp/server-7.0.asc | \
   sudo gpg -o /usr/share/keyrings/mongodb-server-7.0.gpg \
   --dearmor
#create file list
echo "deb [ arch=amd64,arm64 signed-by=/usr/share/keyrings/mongodb-server-7.0.gpg ] https://repo.mongodb.org/apt/ubuntu jammy/mongod
# reload package db
sudo apt-get update
# Install MongoDB Community Server
sudo DEBIAN_FRONTEND=noninteractive apt-get install -y mongodb-org=7.0.6 mongodb-org-database=7.0.6 mongodb-org-server=7.0.6 mongodb
#start
sudo systemctl start mongod
#configure the bind ip
sudo nano /etc/mongod.conf
## change bind IP to 0.0.0.0
#restart the mongo db service
sudo systemctl restart mongod
```

app deployment

```
#!/bin/bash
#upgrade and update all
sudo DEBIAN_FRONTEND=noninteractive apt-get update && sudo DEBIAN_FRONTEND=noninteractive apt-get dist-upgrade -y
#needed installs
sudo DEBIAN_FRONTEND=noninteractive apt install nginx -y
sudo systemctl enable nginx
sudo git clone https://github.com/lowndes96/tech501-sparta-app-cicd.git /repo
sudo DEBIAN_FRONTEND=noninteractive bash -c "curl -fsSL https://deb.nodesource.com/setup_20.x | bash -" && \
sudo DEBIAN_FRONTEND=noninteractive apt-get install -y nodejs
sudo npm install -g pm2
export DB_HOST=mongodb://10.0.3.35:27017/posts
# set up reverse proxy
sudo nano /etc/nginx/sites-available/default
## replace try files line with proxy pass line
sudo chown -R $USER:$USER /repo/app
# reload
sudo nginx -t
sudo systemctl reload nginx
#move to app
cd /repo/app
#run npm install
npm install
pm2 start app.js
```

Bash scripting

general points:

- script must always start with `#!bin/bash` so that the script is executed using bash
- all `nano` commands replaced with `sed`
    - I used regex for some of this, which in retrospect may have been overkill - but was focussing on making code reliable
- This is a good time to start using log files as demonstrated in the completed app script to aid any debugging

use of sed

- sed is a streamlined editor
- in this case we are using it to find and replace text
- basic syntax:

```
sed [options] 'command' [inputfile...]
```

- options:

Option	Description
-i	Edit the file in place without printing to the console (overwrite the file).
-n	Suppress automatic printing of lines.
-e	Allows multiple commands to be executed.
-f	Reads sed commands from a file instead of the command line.
-r	Enables extended regular expressions.

- command:
  - example "s/emily/alex" this would substitute "emily" for "alex" default is the first instance on each line
  - regex can also be used here

db:

sed command for setting bind ip:

```
#configure the bind ip
sudo sed -i 's/^  bindIp: [0-9]\{1,3\}\.[0-9]\{1,3\}\.[0-9]\{1,3\}\.[0-9]\{1,3\}/  bindIp: 0.0.0.0/' /etc/mongod.conf
```

- regex ensures any num.num.num.num pattern is replaced with 0.0.0.0 (maybe unnecessary)

app

sed command for setting proxy pass:

```
# set proxy pass
sudo sed -i "s/try_files \$uri \$uri\/ =404/proxy_pass http:\/\/localhost:3000/g" /etc/nginx/sites-available/default
```

- this replaces the try files line in two locations, which doesn't affect the end result, but worth knowing
-

user data

db:

- need to enable mongod to start on boot, or this code will not work properly as part of user data (discovered during debugging)

```
# enabling mongod to start on boot, starts it
sudo systemctl enable mongod
sudo systemctl is-enabled mongod # checking it's enabled
```

app

full script:

```
#!/bin/bash

# Define log file
LOG_FILE="/emily_custom_data.log"

# Redirect stdout and stderr to the log file
exec > >(sudo tee -a "$LOG_FILE") 2>&1

#upgrade and update all
echo "fetching and upgrading packages..."
sudo DEBIAN_FRONTEND=noninteractive apt-get update && sudo DEBIAN_FRONTEND=noninteractive apt-get dist-upgrade -y

#needed installs
echo "Installing nginx..."
sudo DEBIAN_FRONTEND=noninteractive apt install nginx -y
sudo systemctl enable nginx

echo "Download and install node.js v20..."
sudo DEBIAN_FRONTEND=noninteractive bash -c "curl -fsSL https://deb.nodesource.com/setup_20.x | bash -" && \
sudo DEBIAN_FRONTEND=noninteractive apt-get install -y nodejs

echo "checking nodejs and npm versions:"
node -v
npm -v

echo "install of pm2 using npm..."
sudo npm install -g pm2

# set up reverse proxy
echo "setup reverse proxy..."
sudo sed -i "s/try_files \$uri \$uri\/ =404/proxy_pass http:\/\/localhost:3000/g" /etc/nginx/sites-available/default
sudo nginx -t
sudo systemctl reload nginx


#setup of db connection (remove as needed)
echo "setting uup db connection..."
export DB_HOST=mongodb://10.0.3.35:27017/posts

#accessing and running app
echo "cloning app folder and accessing..."
sudo git clone https://github.com/lowndes96/tech501-sparta-app-cicd.git /repo
sudo chown -R $USER:$USER /repo/app
cd /repo/app

echo "seeding the database..."
npm install

echo "stopping old instances of the app and starting again..."
pm2 delete app.js
pm2 start app.js

echo "Script execution completed. Logs stored in $LOG_FILE"
```

images

db image

- make db_vm using user data + make image

```
#!bin/bash
#update and upgrade packages without asking for user input
sudo DEBIAN_FRONTEND=noninteractive apt-get update && sudo DEBIAN_FRONTEND=noninteractive apt-get dist-upgrade -y
#install mongo db
sudo apt-get install gnupg curl
# import gpg key
curl -fsSL https://www.mongodb.org/static/pgp/server-7.0.asc | \
   sudo gpg -o /usr/share/keyrings/mongodb-server-7.0.gpg \
   --dearmor
#create file list
echo "deb [ arch=amd64,arm64 signed-by=/usr/share/keyrings/mongodb-server-7.0.gpg ] https://repo.mongodb.org/apt/ubuntu jammy/mongod
# reload package db
sudo apt-get update
# Install MongoDB Community Server
sudo DEBIAN_FRONTEND=noninteractive apt-get install -y mongodb-org=7.0.6 mongodb-org-database=7.0.6 mongodb-org-server=7.0.6 mongodb
#start
sudo systemctl start mongod
#configure the bind ip
sudo sed -i 's/^  bindIp: [0-9]\{1,3\}\.[0-9]\{1,3\}\.[0-9]\{1,3\}\.[0-9]\{1,3\}/  bindIp: 0.0.0.0/' /etc/mongod.conf
# enabling mongod to start on boot, starts it
sudo systemctl enable mongod
sudo systemctl is-enabled mongod # checking it's enabled
#restart the mongo db service
sudo systemctl restart mongod
```

app image

- use the script above in the 'user data section without the DB_HOST export line, as this is added using our `run-app-only.sh` script when making a new app from the provisioned app image

`run-app-only.sh` script:

```
#!/bin/bash
#move to app
sudo chown -R $USER:$USER /repo/app
cd /repo/app
export DB_HOST=mongodb://new.db.private.ip:27017/posts
#run npm install
npm install
pm2 delete app.js
pm2 start app.js
```

Blockers – Suggestion: what was the issue, reason for the issue, solution

- needed to change the ownership of the app repo after it was downloaded
  - in order to store the folder in root the sudo command needed to be used
  - but don't want to run the app using sudo as this creates other downstream issues
  - hence the `chown` command needs to be used to change the ownership of the repo folder once added
- enabling of mongodb
  - I encountered some issues when transitioning from running scripts to user data, discovered via group debugging that you have to enable some programmes, such as mongodb to run on boot and added the appropriate commands into my script
- Discovered that `waagent deprovision user` command was not needed to make images run correctly
  - may need to brush up on exactly what this does.
- had an issue with my `#!/bin/bash` line missing a `/` which caused issues for some time

What you learnt

- go slowly and troubleshoot step by step
  - all my issues arose from trying to add too much at once
  - even if confident test the steps, troubleshooting takes longer that just checking as you go
- importance of enabling things on boot
- should have used terraform to aid in this, I dove in and should have spent more time setting things up to make it easier
- how to make log files, was incredibly useful once I got stuck, wish I'd utilised this earlier
  - also including checks (i.e npm -v) in the scripts to look back on

Benefits you saw personally from the project

- the end result was very satisfying and works quickly and consistently
- working through the workflow at the top helped me determine why certain lines of code are necessary and when they become important