Task1
Create three Ec2's using the yaml file deploy8.yaml
Cd in .ssh folder and ssh-keygen
cd /etc/ansible/
Nano the hosts file and replace the ip with your public ip ec2

[ubuntu]
54.211.32.132 ansible_user=ubuntu ansible_ssh_private_key_file=~/.ssh/deploy08
[ubuntu1]
3.84.22.180 ansible_user=ubuntu ansible_ssh_private_key_file=~/.ssh/deploy08
[ubuntu2]
54.211.32.132 ansible_user=ubuntu ansible_ssh_private_key_file=~/.ssh/deploy08

Sudo nano dependes.yml
Replace the ips with your ec2 ip's and key with your keyname
sudo ansible-playbook dependes.yml (yml file playbook may not work so just do the commands manually)
Ssh into your ec2's and in the .ssh directory sudo known_hosts and copy the public key content into it
ansible all -m ping -v

ansible-playbook test.yml
Replace anything you need too

If installing docker fails for whatever reason check the third ec2 by sshing and do docker ps if its not there do sudo rm /usr/share/keyrings/docker-archive-keyring.gpg then rerun the playbook.

Part 2,3, and 4
We are going to create a Jenkinsfile and figure out how to build, test, and deploy.
Sudo apt install npm on the second ec2
Make a new agent for jenkins and connect the node
Create a multibranch pipeline on jenkins
Run the build

Testing
You can use npm test to do testing and also change app.test.js text to make it pass

Deploy
Generate a credential on dockerhub in the security settings of your account
Add credentials on Jenkins
Put token in password field and username should be your dockerhub name
Edit the jenkins pipeline script as needed
Build the pipeline

Part5
Monitoring
Start by running a audit on the application and saving the file somewhere
npm audit > ApplicationAudit
Put in a password like abc
Will create a file with .gpg extension after it
Now we want to encrypt the file…
We will use gpg which is most likely pre-installed
gpg -c filename
You can decrypt a file using the following
gpg -d data.gpg
Create a dashboard I will name mine AlarmDeploy and select alarm widget
Create an alarm
Select a metric
Select the cpu utilization for the ec2 you want to monitor
Alarm will trigger when cpu utilization is greater than 0.4 = 40%
Name the alarm and create it
sudo apt-get update -y
Install stress-ng
sudo apt-get install -y stress-ng
stress-ng -c 1 -p 55
Encrypt and upload the report to the repo using ansible encryption
ansible-vault encrypt CloudWatchReport.txt
To decrypt...
Ansible vault decrypt filename

Additional Info

Docker
Sudo nano Dockerfile
docker pull openjdk
docker build -t nameforimage . (Builds the image using Dockerfile)
Docker build -f .\Dockerfile . -t backend_test
docker run -dit -p 3000:3000 --name saiApp openjdk:latest (test the application on your browser)

Creating your github key so you can ssh and do git push
Ssh-keygen
Cd ~/.ssh
eval "$(ssh-agent -s)"
ssh-add ~/.ssh/id_ed25519
Click on developer settings on your profile

Create a personal access token
For scopes check full repo
Copy the key generated

How to Bridge containers
docker run -dit -p 3000:3000 --name Syip2 syiptest
docker run -dit -p 5000:5000 --name Syip1 backend_test
docker network create linker
docker network connect linker Syip2
docker network connect linker Syip1

File Encryption use ansible
https://docs.ansible.com/ansible/latest/user_guide/vault.html

MYSQL client may need a couple other packages
https://pypi.org/project/mysqlclient/

Cypress testing
Npx cypress open
You will need to move the test.spec.js file into the cypress/integration folder
You need cypress.json file in your repo
Your result will probably be somewhere in your ec2 that tests the app
My location: ~/workspace/Deploy08Branch_main/results