# Phase-6-README

This is Team 1 submission for phase 6 of the final project in ENPM818N.

# Deliverables

- [x] 1. A fully functional CI/CD pipeline automating build, test, and deployment.
- [x] 2. Clear pipeline configuration files with integration into a container.
- [x] 3. Logs or screenshots showcasing successful builds, tests, deployments.
- [x] 4. Validation of rollback functionality ensuring recovery from deployment.
- [x] 5. Confirmation of secure management of sensitive data.

# Proof

## 1. A fully functional CI/CD pipeline automating build, test, and deployment

## 2. Clear pipeline configuration files with integration into a container

1. **build-images.yml (Link):** This workflow automates the process of building Docker images for all microservices within the project. Upon code commits to the main branch, it systematically constructs each service's Docker image and securely pushes them to the GitHub Container Registry (GHCR). This ensures that the latest application versions are readily available for deployment and testing, maintaining consistency and traceability through versioned image tags.
2. **run-integration-tests.yml (Link):** Activated upon the successful completion of build-images.yml, this workflow executes comprehensive integration tests. It checks out the latest code, rebuilds necessary Docker images, prunes unused Docker resources to optimize the environment, and initiates trace-based testing after a brief pause to allow services to stabilize. This process verifies the seamless interaction and functionality of all integrated services, ensuring system integrity before deployment.
3. **deploy.yml (Link):** After successful completion of run-integration-tests.yml This deployment workflow leverages Helm to generate updated Kubernetes manifests based on the latest Docker images from GHCR. It applies these manifests directly to the AWS EKS cluster using `kubectl`, ensuring seamless and automated updates to the production environment. Additionally, it incorporates a rollback mechanism that monitors the deployment's health for five minutes post-deployment, automatically reverting to the previous stable state if any issues are detected, thereby enhancing system reliability and minimizing downtime.

## 3. Logs or screenshots showcasing successful builds, tests, deployments

- Check `phase-6-build-images_run_example.log`, `phase-6-run-integration-tests_run.log` and `phase-6-deploy_run.log` to see full logs of successful builds, tests and deployments.
- Chcek `phase-6-build-images_run_pic.png`, `phase-6-run-integration-tests_run_pic.png` and `phase-6-deploy_run_pic.png` to see screenshots of successful builds, tests and deployments.

## 4. Validation of rollback functionality ensuring recovery from deployment.

Since it's hard to simulate a real issue after successful building and testing, here is a sinppent for the logs showing the rollback logic being recongnized and employed. In Addition, I have attached the result of testing the conditional statement to ensure the output does indeed catch healhy and unhealthy deployment.

```
======CMD======
[45](:46)# Wait for 5 minutes
[46](:47)sleep 30
[47](:48)# Check for unhealthy pods
[48](:49)UNHEALTHY_PODS=$(kubectl get pods -n otel-demo --field-selector=status.phase!=Running
--no-headers 2> /dev/null | wc -l)
[49](:50)if [ "$UNHEALTHY_PODS" -ne 0 ]; then
[50](:51) echo "Detected issues with deployment. Rolling back..."
[51](:52) # Restore previous manifest
[52](:53) if [ -f /home/ec2-user/kubernetes/opentelemetry-demo-backup.yaml ]; then
[53](:54) kubectl apply -f /home/ec2-user/kubernetes/opentelemetry-demo-backup.yaml
[54](:55) else
[55](:56) echo "No backup manifest found. Cannot rollback."
[56](:57) exit 1
[57](:58) fi
[58](:59)else
[59](:60) echo "Deployment is healthy."
[60](:61)fi
[61](:62)======END======
[64](:65)out: Deployment is healthy.
```

### Testing Condition in EC2

```
[ec2-user@ip-10-0-0-57 ~]$ kubectl get pods -n otel-demo --field-selector=status.phase=Running
--no-headers
opentelemetry-demo-accountingservice-559bc4fd75-9wrvg       1/1   Running   0     4m46s
opentelemetry-demo-adservice-5df4bf56d6-tprfv               1/1   Running   0     32m
opentelemetry-demo-cartservice-694df79ff7-2d4lf             1/1   Running   0     32m
opentelemetry-demo-checkoutservice-744b6cdd6-h5dnp          1/1   Running   0     32m
opentelemetry-demo-currencyservice-6b8fcd9bd6-fqggg         1/1   Running   0     32m
opentelemetry-demo-emailservice-66d8476fdc-2v5x8            1/1   Running   0     32m
opentelemetry-demo-flagd-5946c56f88-9kvf5                   2/2   Running   0     32m
opentelemetry-demo-frauddetectionservice-85d48f855f-6p2zd   1/1   Running   0     32m
opentelemetry-demo-frontend-59bccd8fdb-hj9h8                1/1   Running   0     32m
opentelemetry-demo-frontendproxy-f46cf7d-r8msf              1/1   Running   0     32m
opentelemetry-demo-grafana-69b6bd5dd4-vg529                 1/1   Running   0     32m
opentelemetry-demo-imageprovider-7466d894fb-kc5px           1/1   Running   0     32m
opentelemetry-demo-jaeger-7785549bb-nq2jw                   1/1   Running   0     32m
opentelemetry-demo-kafka-76d4d9f48b-wqjdr                   1/1   Running   0     31m
opentelemetry-demo-loadgenerator-6994f5db8-9q8kz            1/1   Running   0     31m
opentelemetry-demo-otelcol-5c757cfcf-lllvf                  1/1   Running   0     32m
opentelemetry-demo-paymentservice-857974bcdb-tnmwb          1/1   Running   0     31m
opentelemetry-demo-productcatalogservice-6bc98644f9-rqqvj   1/1   Running   0     31m
opentelemetry-demo-prometheus-server-57cd8f9d46-mjtt6       1/1   Running   0     32m
opentelemetry-demo-quoteservice-5658c58fd7-zzs64            1/1   Running   0     31m
opentelemetry-demo-recommendationservice-6f576fd574-lst2d   1/1   Running   0     31m
opentelemetry-demo-shippingservice-fcfc7765-xncrg           1/1   Running   0     31m
opentelemetry-demo-valkey-68b4cb4498-59jkq                  1/1   Running   0     31m
otel-demo-opensearch-0                                      1/1   Running   0     32m

[ec2-user@ip-10-0-0-57 ~]$ kubectl get pods -n otel-demo --field-selector=status.phase!=Running
--no-headers 2> /dev/null | wc -l
0

[ec2-user@ip-10-0-0-57 ~]$ kubectl get pods -n otel-demo --field-selector=status.phase=Running
--no-headers 2> /dev/null | wc -l
24
```

You can also check `phase-6-rollback-condition-tested.png` to see a screenshot of this result.

## 5. Confirmation of secure management of sensitive data.

This is evident in the yaml files where we can see there are no sensitive data, instead I used secret variables. Examples:

```
deploy.yml
33:           host: ${{ secrets.EC2_HOST }}
35:           key: ${{ secrets.EC2_SSH_KEY }}

component-build-images.yml
166:           password: ${{ secrets.GITHUB_TOKEN }}
```

And you can check `phase-6-github-secret-pic.png` for a screenshot from GitHub.