OPSWAT DevOps CloudOps Training Program 2025

Final Exam Announcement

Exam Structure | Submission | Timeline

Exam Structure

We have already prepared the container image for the web application. The images and their environment variables are: (Repo link: https://hub.docker.com/u/cuongopswat)

- cuongopswat/go-coffeeshop-web
  - REVERSE_PROXY_URL: <proxy_service:port>
  - WEB_PORT: 8888
- cuongopswat/go-coffeeshop-proxy
  - APP_NAME: <Any Name>
  - GRPC_PRODUCT_HOST: <product service>
  - GRPC_PRODUCT_PORT: 5001
  - GRPC_COUNTER_HOST: <counter service>
  - GRPC_COUNTER_PORT: 5002
- cuongopswat/go-coffeeshop-barista
  - APP_NAME: <Any Name>
  - IN_DOCKER: "true"
  - PG_URL: <Postgresql connection string>
  - PG_DSN_URL: <host=... user=... password=... dbname=... sslmode=disable> sslmode=disable
  - RABBITMQ_URL: <RabbitMQ connection string>
- cuongopswat/go-coffeeshop-kitchen
  - APP_NAME: <Any Name>
  - IN_DOCKER: "true"
  - PG_URL: <Postgresql connection string>
  - PG_DSN_URL: <host=... user=... password=... dbname=... sslmode=disable> sslmode=disable
  - RABBITMQ_URL: <RabbitMQ connection string>
- cuongopswat/go-coffeeshop-counter
  - APP_NAME:<Any Name>
  - IN_DOCKER: "true"
  - PG_URL: <Postgresql connection string>
  - PG_DSN_URL: <host=... user=... password=... dbname=... sslmode=disable>
  - RABBITMQ_URL: <RabbitMQ connection string>
  - PRODUCT_CLIENT_URL: <product-service:port>

- cuongopswat/go-coffeeshop-product
  - APP_NAME: <Any Name>
- postgres:14-alpine
  - POSTGRES_DB
  - POSTGRES_USER
  - POSTGRES_PASSWORD
- rabbitmq:3.11-management-alpine
  - RABBITMQ_DEFAULT_USER
  - RABBITMQ_DEFAULT_PASS

Start order:

1. PostgreSQL:
2. RabbitMQ:
3. Product
4. Counter
5. The remaining services

Port expose:

- Postgresql: 5432
- RabbiMQ: 5672 and 15672
- proxy: 5000
- product: 5001
- counter: 5002
- web: 8888

Here is your exam:

	Section	Subsection	Weight	Requirement
1	Preparation			1. The infrastructure is hosted on AWS which can be used with a personal or the OPSWAT Devops training account. 2. Your code should be stored on any VCS such as GitHub, Gitlab… It's supposed to share your code with the examiner
2	Infrastructure as code (Terraform and awscli)		20%	1. Resources: (You should read the full exam to clarify all the requirements for each resource) 2. All resources are created by Terraform or AWS CLI. 3. Public modules are allowed, but not with basic infrastructure (VPC, Subnet, SG) 4. Use a different workspace for each environment. 5. Using remote S3 backend.
3	Application deployment	Prepare		Pull the above images and push to your Docker private registry (You can change the image name if you want)
4		Dev environment	10%	1. Create the EC2 and install Docker 2. Prepare Docker-Compose to deploy the dev environment

				3. Ensure healthcheck settings
5		Prod environment	20%	1. Create the EKS cluster with minimal resource capacity 2. Create in-cluster resources by the Declarative method. 3. Create YAML files for all resources 4. Create a Horizontal Pod Autoscaling to apply scale 5. Ensuring security both at rest and in transit. 6. Ensure healthcheck settings
6		Database	10%	Using Postgresql by using RDS (use Free-tier option, t3.micro, no HA, no read-replica setup), ensure using Secret Manager to store sensitive data
7	CI/CD		20%	CI/CD pipeline (can choose AWS CodePipeline or TeamCity, or Github Actions) to support application integration and delivery. The pipeline should be: 1. Scan the image with the Trivy software 2. Push the image to the Docker private registry 3. Deploy the application with the latest image, can choose one of the 2 mechanisms below: ◦ (Optional) Using Argocd to apply the GitOps pull mechanism. ◦ (Optional) Using Helm/Kubeclt to apply the GitOps push mechanism. (You can spin up a new EC2 to deploy TeamCity and Argocd for your own)
8	Monitoring System		20%	1. Monitoring system (can choose to use Amazon CloudWatch/Grafana) to monitor the system. a. A dashboard to monitor: Nodes CPU and Memory, Pods CPU and Memory, API requests, 4xx, 5xx http requests. b. Alert if: ▪ The autoscaling scales to the maximum number. ▪ There's an anomaly change in the ELB's RequestCount. ▪ High memory or CPU usage in any component of application ▪ 5xx errors

Submission

There are 2 parts:

1. **Provide your repository with a friendly format README to describe the solution for the exam requirement, including sections**:
   a. Summary: Describe the summary of the solution.
   b. Architecture: The architecture of the solution.
   c. Component description: The Description of each component in the architecture.
   d. The homepage of the application

    e. User guideline: The guideline on how to use the source code to provision/deploy components in the architecture.

2. **Demonstration**:
    a. **DevOps Intern Learners**: offline demonstration with 30-45 minutes per person.
    b. **Regular Learners (not DevOps Intern)**: a short video to demonstrate your deployment. You can cut the deployment progress because it takes a long time to complete.

Once done, you can **send your submission to** cuong.tran@opswat.com.

Timeline

- For Regular Learners, submission deadline: May 20, 2025
- For DevOps Intern Learners, demonstration date: May 20, 2025
- Trainer team's evaluation: from May 21, 2025 to May 28, 2025
- Result announcement: May 29, 2025

If you have any concerns, feel free to contact @Cuong Tran Nguyen Huy or @Ly Doan .