CLO24 -- Kubernetes -- Niklas Häll (2025)

Index

[Project: Description here 2](#_Toc210400299)

[1. Introduction 2](#_Toc210400300)

[1.1 The entire project can be found at GitHub 2](#_Toc210400301)

[1.2 Prerequisites (knowledge, accounts and software installs) 2](#_Toc210400302)

[2. Overview of the project 3](#_Toc210400303)

[2.1 Choices, choices.. why when what how? 3](#_Toc210400304)

[2.2 General overview: Infrastructure 3](#_Toc210400305)

[2.3 General overview: Kubernetes 3](#_Toc210400306)

[2.4 General overview: CI/CD 3](#_Toc210400307)

[2.5 General overview: .NET 3](#_Toc210400308)

[2.6 General overview: LLM in this project 3](#_Toc210400309)

[2.7 The system in action 3](#_Toc210400310)

[2.7.1 bla bla flowchart 3](#_Toc210400311)

[2.9.2 blu blu flowchart 4](#_Toc210400312)

[2.9.3 User interaction flowchart 4](#_Toc210400313)

[3. Infrastructure in-depth 4](#_Toc210400314)

[3.1 Verification and troubleshooting: Infrastructure 4](#_Toc210400315)

[4. Kubernetes in-depth 4](#_Toc210400316)

[4.1 Verification and troubleshooting: AWS 4](#_Toc210400317)

[9. Where to go (improvement possibilities) 4](#_Toc210400318)

[9.1 Security 4](#_Toc210400319)

[9.2 CI/CD analysis 5](#_Toc210400320)

[9.3 Scalability analysis 5](#_Toc210400321)

[9.4 Cost analysis 5](#_Toc210400322)

[9.5 Logs and metrics 5](#_Toc210400323)

[9.6 Other areas that might be worth considering 5](#_Toc210400324)

[10. Challenges and lessons learned 5](#_Toc210400325)

[10.1 Number one 5](#_Toc210400326)

[10.2 Number two 5](#_Toc210400327)

[10.3 Number three 6](#_Toc210400328)

[11. Build timeline 6](#_Toc210400329)

[11.1 Phase one – Let the Kube-Snake loose! 6](#_Toc210400330)

[11.1.1 Phase one, step one – Local Development & Remote Infrastructure 6](#_Toc210400331)

[11.1.2 Phase one, step two – CI/CD Pipeline Implementation 7](#_Toc210400332)

[11.2 Phase two, microscopic services under the hood 7](#_Toc210400333)

[Goal: Implement GitOps tooling (Helm + ArgoCD) and deploy MongoDB as first stateful workload. 7](#_Toc210400334)

[11.2.1 Phase two, step one – who’s at the HELM anyway? 7](#_Toc210400335)

[11.2.2 Phase two, step two – ArgoCD: GitOps in Action(s) 8](#_Toc210400336)

[11.2.3 Phase two, step three – GitOps Workflow Integration 8](#_Toc210400337)

[11.2.4 Phase two, step four – MongDB: Stateful workloads 8](#_Toc210400338)

[11.2.5 Phase two, step five – .NET API Integration 8](#_Toc210400339)

[12. References and links 8](#_Toc210400340)

[11.1 Our class’ studyguide/tutorial 8](#_Toc210400341)

[11.2 LLM: Partners (and enemies) in crime 9](#_Toc210400342)

[11.3 More references? 9](#_Toc210400343)

# Project: Description here

# 1. Introduction

Lorum Ipsum

## 1.1 The entire project can be found at GitHub

Made you look! <https://github.com/mymh13/kube-snake>

## 1.2 Prerequisites (knowledge, accounts and software installs)

To replicate this system, you are expected to know and have:

* Point 1
* Point 2
* Point 3

# 2. Overview of the project

Lorum Ipsum overview-um

## 2.1 Choices, choices.. why when what how?

A rough overview would be this: Lorum Ipsum

## 2.2 General overview: Infrastructure

choices

## 2.3 General overview: Kubernetes

choices

## 2.4 General overview: CI/CD

choices

## 2.5 General overview: .NET

choices

## 2.6 General overview: LLM in this project

Explanation here

## 2.7 The system in action

An overview of the natural flow within the system:

### 2.7.1 bla bla flowchart

1. List 1
2. List 2
3. List 3
4. List 4

### 2.9.2 blu blu flowchart

1. Further listing
2. Further listing 2

### 2.9.3 User interaction flowchart

1. More lists
2. More lists 2

Example below: can look like this

# 3. Infrastructure in-depth

Intro text

## 3.1 Verification and troubleshooting: Infrastructure

Common issues and where to look:

* list

# 4. Kubernetes in-depth

Text

## 4.1 Verification and troubleshooting: AWS

* List

# 9. Where to go (improvement possibilities)

There are quite a lot of roads leading to Rome, but let us look at a few..

## 9.1 Security

Strengths: strengths listed

Improvements: improvements listed

## 9.2 CI/CD analysis

Strengths: strengths listed

Improvements: improvements listed

## 9.3 Scalability analysis

Strengths: strengths listed

Improvements: improvements listed

## 9.4 Cost analysis

Thoughts on the topic here

## 9.5 Logs and metrics

Strengths: strengths listed

Improvements: improvements listed

## 9.6 Other areas that might be worth considering

Lorum ipsum

# 10. Challenges and lessons learned

List here

## 10.1 Number one

Wow

## 10.2 Number two

Wow

## 10.3 Number three

Wowowow

# 11. Build timeline

The GitHub repository will provide a more detailed history, which I also outline more specifically in modules in the markdown-documents inside the project (/docs/phase\_<phase-number>). But this serves as a rough outline of progression, to show how the system came to life.

## 11.1 Phase one – Let the Kube-Snake loose!

Goal: Establish K3s infrastructure and CI/CD pipeline with automated healthcheck deploy.

### 11.1.1 Phase one, step one – Local Development & Remote Infrastructure

Rudimentary infrastructure and a landing page:

1. Project Planning & Setup

* Laid down the general architecture and plan for the project
* Created GitHub repository and local project structure with pre-set directories
* Created architecture documentation (architecture.md)

2. Local K3s Testing (Docker Desktop)

* Verified K3s cluster working in Docker Desktop environment
* Created nginx-health-check-page.yaml deployment manifest
* Tested basic kubectl commands and internal cluster networking

3. Remote Infrastructure Preparation (Hetzner VM)

* Updated VM: apt update && apt upgrade && reboot
* Installed K3s with Traefik disabled: curl -sfL https://get.k3s.io | sh -s - --disable=traefik
* Configured kubectl access: set user permissions on kubeconfig, added KUBECONFIG to .bashrc

4. Deploy Healthcheck to Remote Cluster

* Deployed nginx healthcheck directly to VM via SSH using kubectl apply
* Verified internal service running (ClusterIP: 10.43.39.44:80)

5. External Access Configuration

* Configured Caddy reverse proxy: edited /etc/caddy/Caddyfile with subdomain block
* Added DNS A record pointing subdomain to VM IP
* Verified SSL certificate auto-provisioned by Caddy
* Confirmed external access at https://kube-snake.mymh.dev

Flow: Internet > Caddy (VM) > K3s Service > nginx container (pod)

### 11.1.2 Phase one, step two – CI/CD Pipeline Implementation

Automated deployment workflow using GitHub Actions

1. Custom Healthcheck Container

* Created custom index.html landing page with status information
* Created Dockerfile using nginx:alpine base image
* Added styles.css for consistent styling and favicon.ico

2. GitHub Actions Workflow Setup

* Created deploy-healthcheck.yml with build and deploy jobs
* Configured workflow to build Docker image and push to GitHub Container Registry (GHCR)
* Updated nginx-health-check-page.yaml to use custom GHCR image

3. Secrets & Authentication Configuration

* Added GitHub repository secrets: SSH\_HOST, SSH\_USER, SSH\_PRIVATE\_KEY
* Cloned repository to VM for kubectl access during deployment
* Resolved SSH authentication issues (key format, passphrase handling)

4. Deployment Automation & Testing

* Implemented automated deployment: Git push > Docker build > GHCR push > kubectl deploy
* Fixed image caching issue: added imagePullPolicy: Always and SHA-based tagging
* Modified workflow to use kubectl set image with commit SHA tags for reliable updates
* Verified end-to-end CI/CD pipeline working successfully

Updated Flow**:** Git Push > GitHub Actions > Build Image > GHCR > SSH to VM > kubectl set image > Pod Update

## 11.2 Phase two, microscopic services under the hood

### Goal: Implement GitOps tooling (Helm + ArgoCD) and deploy MongoDB as first stateful workload.

### 11.2.1 Phase two, step one – who’s at the HELM anyway?

Converting from raw manifests to reusable Helm charts:

1. First
2. Second

### 11.2.2 Phase two, step two – ArgoCD: GitOps in Action(s)

Installing and configuring continuous deployment automation:

1. First
2. Second

### 11.2.3 Phase two, step three – GitOps Workflow Integration

Transitioning from GitHub Actions deployment to ArgoCD-managed deployments:

1. First
2. Second

### 11.2.4 Phase two, step four – MongDB: Stateful workloads

First database deployment with persistent storage:

1. First
2. Second

### 11.2.5 Phase two, step five – .NET API Integration

Testing the full stack with a simple backend service:

1. First
2. Second

# 12. References and links

## 11.1 Our class’ studyguide/tutorial

<https://cloud-developer.educ8.se/clo/4.-run-cloud-applications/3.-kubernetes/index.html>

That webpage and the content is the property of Lars Appel, I am just referring to it.

## 11.2 LLM: Partners (and enemies) in crime

Claude 4.5 and ChatGPT 5

## 11.3 More references?

Provided here