CLO24 -- Kubernetes -- Niklas Häll (2025)

Index

[Project: Description here 2](#_Toc210384964)

[1. Introduction 2](#_Toc210384965)

[1.1 The entire project can be found at GitHub 2](#_Toc210384966)

[1.2 Prerequisites (knowledge, accounts and software installs) 2](#_Toc210384967)

[2. Overview of the project 2](#_Toc210384968)

[2.1 Choices, choices.. why when what how? 2](#_Toc210384969)

[2.2 General overview: Infrastructure 3](#_Toc210384970)

[2.3 General overview: Kubernetes 3](#_Toc210384971)

[2.4 General overview: CI/CD 3](#_Toc210384972)

[2.5 General overview: .NET 3](#_Toc210384973)

[2.6 General overview: LLM in this project 3](#_Toc210384974)

[2.7 The system in action 3](#_Toc210384975)

[2.7.1 bla bla flowchart 3](#_Toc210384976)

[2.9.2 blu blu flowchart 3](#_Toc210384977)

[2.9.3 User interaction flowchart 3](#_Toc210384978)

[3. Infrastructure in-depth 4](#_Toc210384979)

[3.1 Verification and troubleshooting: Infrastructure 4](#_Toc210384980)

[4. Kubernetes in-depth 4](#_Toc210384981)

[4.1 Verification and troubleshooting: AWS 4](#_Toc210384982)

[9. Where to go (improvement possibilities) 4](#_Toc210384983)

[9.1 Security 4](#_Toc210384984)

[9.2 CI/CD analysis 4](#_Toc210384985)

[9.3 Scalability analysis 4](#_Toc210384986)

[9.4 Cost analysis 5](#_Toc210384987)

[9.5 Logs and metrics 5](#_Toc210384988)

[9.6 Other areas that might be worth considering 5](#_Toc210384989)

[10. Challenges and lessons learned 5](#_Toc210384990)

[10.1 Number one 5](#_Toc210384991)

[10.2 Number two 5](#_Toc210384992)

[10.3 Number three 5](#_Toc210384993)

[11. Build timeline 5](#_Toc210384994)

[11.1 Phase one 6](#_Toc210384995)

[12. References and links 6](#_Toc210384996)

[11.1 Our class’ studyguide/tutorial 6](#_Toc210384997)

[11.2 LLM: Partners (and enemies) in crime 6](#_Toc210384998)

[11.3 More references? 6](#_Toc210384999)

# 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 the progression, to show how the system came to life.

## 11.1 Phase one, step one

Rudimentary infrastructure and a landing page:

1. Laid down the general architecture and plan for the project, and created a repo (and project locally, with directories pre-set).
2. Created a ngxinx-health-check-page.yaml to function as a verification page that will help me progress through the project.
3. Prepared my Hetzner Cloud VM by:
   1. running apt update + apt upgrade + rebooted the VM
   2. then I installed K3s with Traefik disabled (since my VM run Caddy)
   3. set my user to access kubeconfig (chmod)
   4. set KUBECONFIG in my .bashrc – now I am ready to run kubectl without sudo
4. Next up I could chose to create my nginx-health-check-page on the VM directly, through SSH, or remote kubectl from my local machine. Since I will implement CI/CD later and this is just a health page: I cat the nginx-page straight to the VM.
5. We need to expose the service, it is only visible internally for now, so I add a Caddy route to expose it: sudo nano /etc/caddy/Caddyfile to edit Caddy and then I add a block for my subdomain > reverse proxy > internal IP:port
6. Added a DNS pointer from the Subdomain: tested, all works

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

## 11.2 Phase one, step two

Initial CI/CD for the Hetzner cloud (will become the backend) deployment:

1. Create an Index.html-landingpage to replace the nginx-template, create a Dockerfile and a deploy\_healthcheck.yml for the basic deploy cycle
2. Update the nginx-link in the k3s-yaml to use the new custom image
3. Add secrets to GitHub for the deployment
4. Clone the repo to the VM
5. Git push and try the workflow

# 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

ChatGPT and Claude

## 11.3 More references?

Provided here