# Im Dripping

(real dripping)

[ sorry ga terlalu jago, nikmati dokumentasi ampas saya ]

# Requirements

Pertama-tama saya mau memberi tahu terlebih dahulu jikalau saya mengerjakan soal ini pada Windows. Sehingga segala hal yang mungkin berbeda pada macOS atau Linux (saya angkat tangan). Ok let’s go, ada beberapa hal yang dibutuhkan untuk mengerjakan soal ini.

* Source code web aplikasi yang akan di-deploy
* Docker
* Kubernetes
* Minikube (saya make minikube)

\*Note: ini saya pake minikube, tapi tetep di-local kalau di-cloud saya bingung karena udah cukup mabok sama pelabuhan, tapi pake minikube 😊

## Install

1. Docker: <https://docs.docker.com/desktop/install/windows-install/>
2. Kubernetes: <https://kubernetes.io/docs/tasks/tools/install-kubectl-windows/>
3. Minikube: <https://minikube.sigs.k8s.io/docs/start/?arch=%2Fwindows%2Fx86-64%2Fstable%2F.exe+download>

# Setup

Yak, kita masuk ke bagian setup. Sebelum itu ada beberapa hal lagi yang disiapkan yaitu, script deployment dan service yang nantinya akan digunakan.

Service.yml

|  |
| --- |
| apiVersion: v1 # versi dari API yang digunakan  kind: Service # jenis resource yang akan dibuat  metadata: # metadata dari resource yang akan dibuat  name: expense-tracker-service # nama service  spec: # spesifikasi dari service  type: NodePort # tipe service yang akan dibuat  selector: # selector untuk menentukan pod yang akan diatur oleh service  app: expense-tracker # label app dengan value expense-tracker  ports: # port yang akan diexpose oleh service  - protocol: TCP # protokol yang digunakan  port: 80 # port 80  targetPort: 80 # port yang dituju  nodePort: 30000 # port yang akan diexpose ke node |

Deployment.yml

|  |
| --- |
| # bagian ini untuk mendefinisikan deployment  apiVersion: apps/v1  # kind untuk mendefinisikan jenis resource yang akan dibuat  kind: Deployment  # metadata untuk mendefinisikan metadata dari resource yang akan dibuat  metadata:  name: expense-tracker-deployment # nama deployment  spec: # spesifikasi dari deployment  replicas: 1 # jumlah replika yang akan dibuat  selector: # selector untuk menentukan pod yang akan diatur oleh deployment  matchLabels: # label yang akan dijadikan acuan oleh selector  app: expense-tracker # label app dengan value expense-tracker  template: # template untuk mendefinisikan pod yang akan dibuat  metadata: # metadata dari pod yang akan dibuat  labels: # label dari pod yang akan dibuat  app: expense-tracker # label app dengan value expense-tracker  spec: # spesifikasi dari pod yang akan dibuat  containers: # container yang akan dijalankan oleh pod  - name: expense-tracker # nama container  image: kiiiro/kubernetes-dripping:latest # image yang akan digunakan  ports: # port yang akan diexpose oleh container  - containerPort: 80 # port 80 |

Ok, lanjut

1. Pertama buat image terlebih dahulu dari webapp yang akan di-deploy. Berikut cuplikan dari dockefile yang ada

A screenshot of a computer program

Description automatically generated

Figure 0‑1 dockerfile untuk membuat image dari webapp

Selanjutnya, buat sebuah repository pada <https://hub.docker.com> yang nantinya image akan di-push ke docker hub (tbh, saya ga yakin part ini buat apa, karena keknya image yang dipake tetep yang local).

Eits, pastikan terminal ada di posisi yang sama dengan adanya dockerfile.

|  |
| --- |
| # build image  docker build -t kiiiro/Kubernetes-dripping .  # push docker image ke docker hub  Docker push kiiiro/Kubernetes-dripping |

Nah, harusnya di local sudah muncul docker image yang telah di-build barusan

A screen shot of a computer

Description automatically generated

Figure 0‑2 Hasil Docker Image

1. Langkah selanjutnya adalah pembuatan cluster pada Kubernetes
2. Jalankan command

* minikube start -p <nama\_profile>

A screen shot of a computer

Description automatically generated

Oke, ini berhasil dibuat

1. Cek, apakah kubectl menggunakan context yang sudah anda buat atau belum, dengan menggunakan command

* kubectl config current-context

A black background with white text

Description automatically generated

Kalo semisal, berbeda bisa gunakan perintah ini

* kubectl config use-context <nama\_cluster>

contoh kasus:

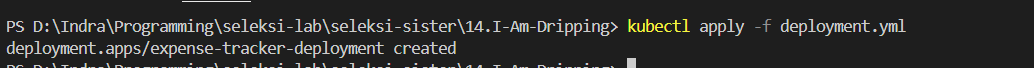
A screen shot of a computer program

Description automatically generated

1. Nah, karena kubectl sudah menggunakan cluster yang sesuai kita bisa apply script service.yml dan deployment.yml pada cluster “dripping”

Gunakan command:

* kubectl apply -f <script\_deploy\_klen.yml>
* kubectl apply -f <script\_service.klen.yml>





1. Last one, jalankan command

* minikube service <nama\_service>

command di atas digunakan untuk memulai service dalam kasus saya yaitu

* + minikube service expense-tracker-service

A screenshot of a computer program

Description automatically generated  
tada, hasilnya bisa dilihat pada URL yang diberikan, dan ini hasilnya.

A screenshot of a computer

Description automatically generated

Note: saya pake webapp yang sama kayak soal Pelabuhan bedanya Pelabuhan 3 layer, yang ini gausah deh saya dah pusing kena azure kemaren.

TERIMA KASIH