

LAPORAN PRAKTIKUM VIRTUALISASI KOMPUTER

MENJALANKAN APLIKASI SEDERHANA NODE.JS MENGGUNAKAN KUBERNETES



Agus Pranata Marpaung

13323033

DIII TEKNOLOGI KOMPUTER

**INSTITUT TEKNOLOGI DEL
FAKULTAS VOKASI**

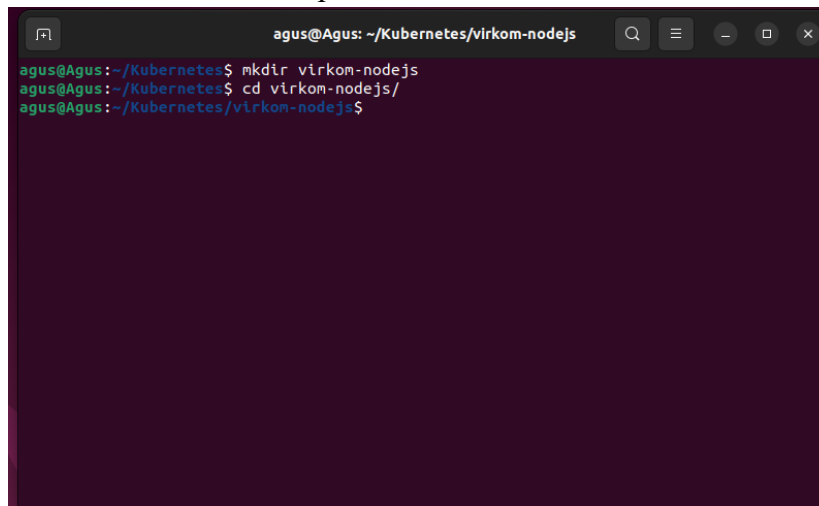
Judul Praktikum

Minggu/Sesi	:	XIV/3
Kode Mata Kuliah	:	4332103
Nama Mata Kuliah	:	VIRTUALISASI KOMPUTER
Setoran	:	Jawaban dalam bentuk <i>softcopy</i>
Batas Waktu Setoran	:	<i>Senin, 2 Desember 2024 jam 21:30</i>
Tujuan	:	1. Mahasiswa mampu membuat dan menjalankan aplikasi sederhana Node.js menggunakan Kubernetes.

Petunjuk

Praktikum

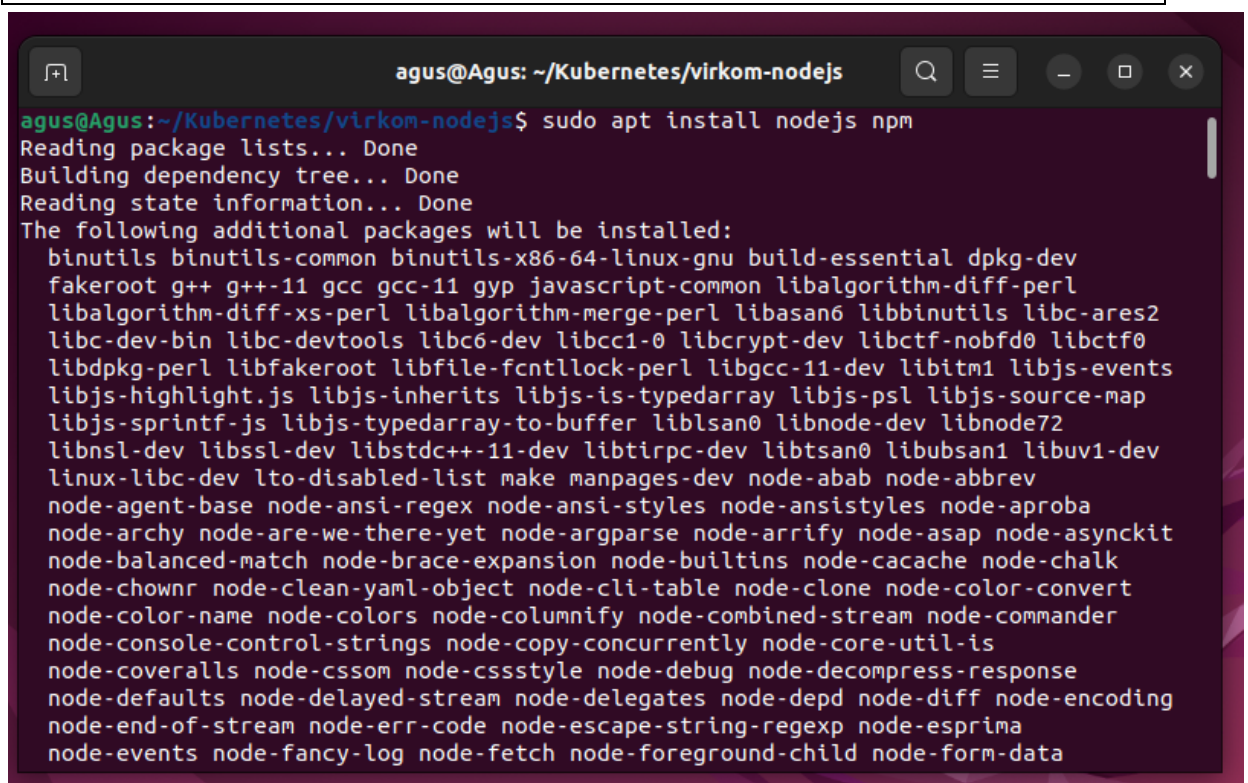
1. Pada praktikum kali ini Kita akan mencoba untuk membuat sebuah aplikasi atau website sederhana menggunakan **Nodejs**. Nodejs adalah sebuah *environment* yang berbasis Javascript dan dapat membantu Kita dalam mengembangkan aplikasi web. Kita akan diminta untuk melakukan instalasi Nodejs dalam host OS Kita. Untuk itu, Kita bisa membuat direktori baru pada host OS Kita.



```
agus@Agus: ~/Kubernetes/virkom-nodejs
agus@Agus:~/Kubernetes$ mkdir virkom-nodejs
agus@Agus:~/Kubernetes$ cd virkom-nodejs/
agus@Agus:~/Kubernetes/virkom-nodejs$
```

2. Kemudian Kita sekarang bisa menginstal node.js, namun sebelumnya Kita perlu melakukan update pada package yang ada pada OS Kita. Setelah itu Kita bisa menjalankan *command* berikut untuk melakukan instalasi node.js beserta dengan package managernya.

```
sudo apt install nodejs npm
```



```
agus@Agus: ~/Kubernetes/virkom-nodejs
agus@Agus:~/Kubernetes/virkom-nodejs$ sudo apt install nodejs npm
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
 binutils binutils-common binutils-x86-64-linux-gnu build-essential dpkg-dev
 fakeroot g++ g++-11 gcc gcc-11 gyp javascript-common libalgorithm-diff-perl
 libalgorithm-diff-xs-perl libalgorithm-merge-perl libasan6 libbinutils libc-ares2
 libc-dev-bin libc-devtools libc6-dev libcc1-0 libcrypt-dev libctf-nobfd0 libctf0
 libdpkg-perl libfakeroot libfile-fcntllock-perl libgcc-11-dev libitm1 libjs-events
 libjs-highlight.js libjs-inherits libjs-is-typedarray libjs-psl libjs-source-map
 libjs-sprintf.js libjs-typedarray-to-buffer liblsan0 libnode-dev libnode72
 libnsl-dev libssl-dev libstdc++-11-dev libtirpc-dev libtsan0 libubsan1 libuv1-dev
 linux-libc-dev lto-disabled-list make manpages-dev node-abab node-abbrev
 node-agent-base node-ansi-regex node-ansi-styles node-ansistyles node-aproba
 node-archy node-are-we-there-yet node-argparse node-arrify node-asap node-asynckit
 node-balanced-match node-brace-expansion node-builtins node-cacache node-chalk
 node-chownr node-clean-yaml-object node-cli-table node-clone node-color-convert
 node-color-name node-colors node-columnify node-combined-stream node-commander
 node-console-control-strings node-copy-concurrently node-core-util-is
 node-coveralls node-cssom node-cssstyle node-debug node-decompress-response
 node-defaults node-delayed-stream node-delegates node-depd node-diff node-encoding
 node-end-of-stream node-err-code node-escape-string-regexp node-esprima
 node-events node-fancy-log node-fetch node-foreground-child node-form-data
```

- Setelah terinstal, Kita bisa menginisialisasi project dengan konfigurasi default pada folder yang sudah Kita buat sebelumnya dengan menjalankan *command* berikut.

```
npm init -y
```

```
agus@Agus:~/Kubernetes/virkom-nodejs$ npm init -y
Wrote to /home/agus/Kubernetes/virkom-nodejs/package.json:

{
  "name": "virkom-nodejs",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "keywords": [],
  "author": "",
  "license": "ISC"
}
```

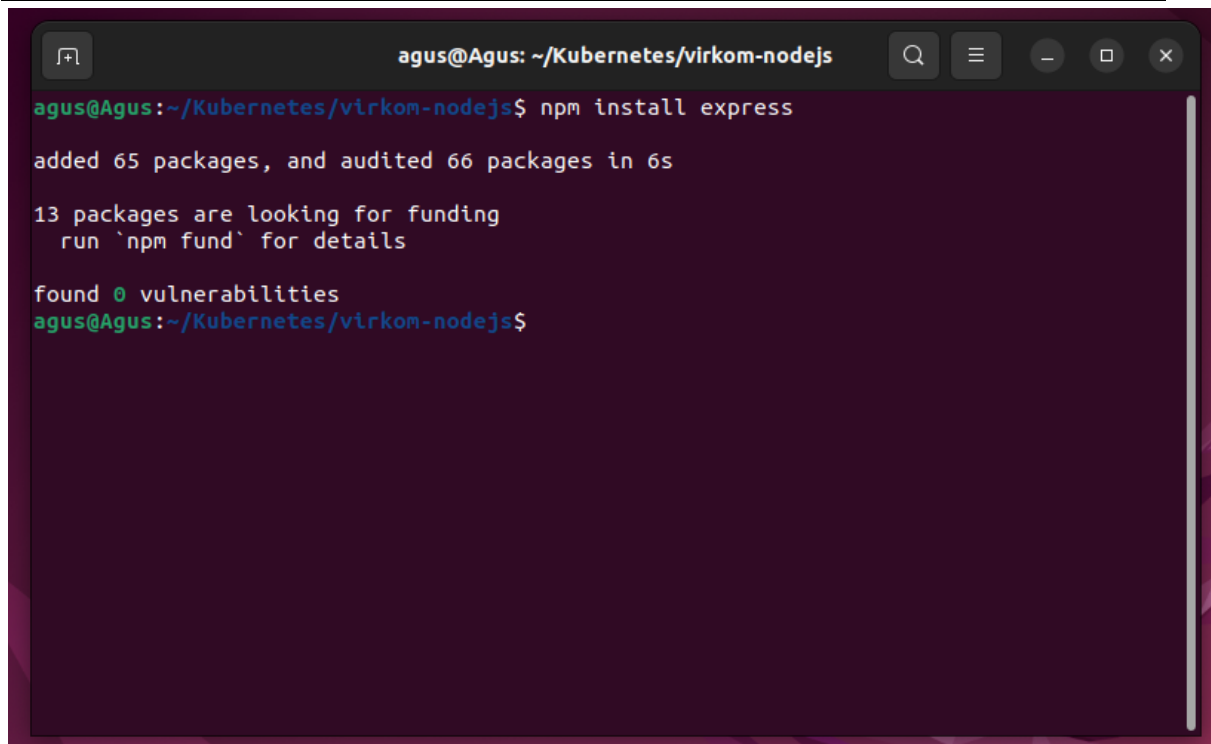
Maka akan terbentuk 2 file dengan ekstensi **json**.

- Lalu Kita bisa menghapus folder **node_modules** dalam *environment* local Kita, dikarenakan Kita tidak akan menjalankan nodejs dalam local Kita melainkan dengan minikube yang sudah Kita buat sebelumnya.

```
agus@Agus: ~/Kubernetes/virkom-nodejs
agus@Agus:~/Kubernetes/virkom-nodejs$ rm -rf node_modules/
agus@Agus:~/Kubernetes/virkom-nodejs$
```

5. Kemudian Kita install package `express.js` sebagai framework untuk website Kita dengan menjalankan *command* berikut.

```
npm install express
```

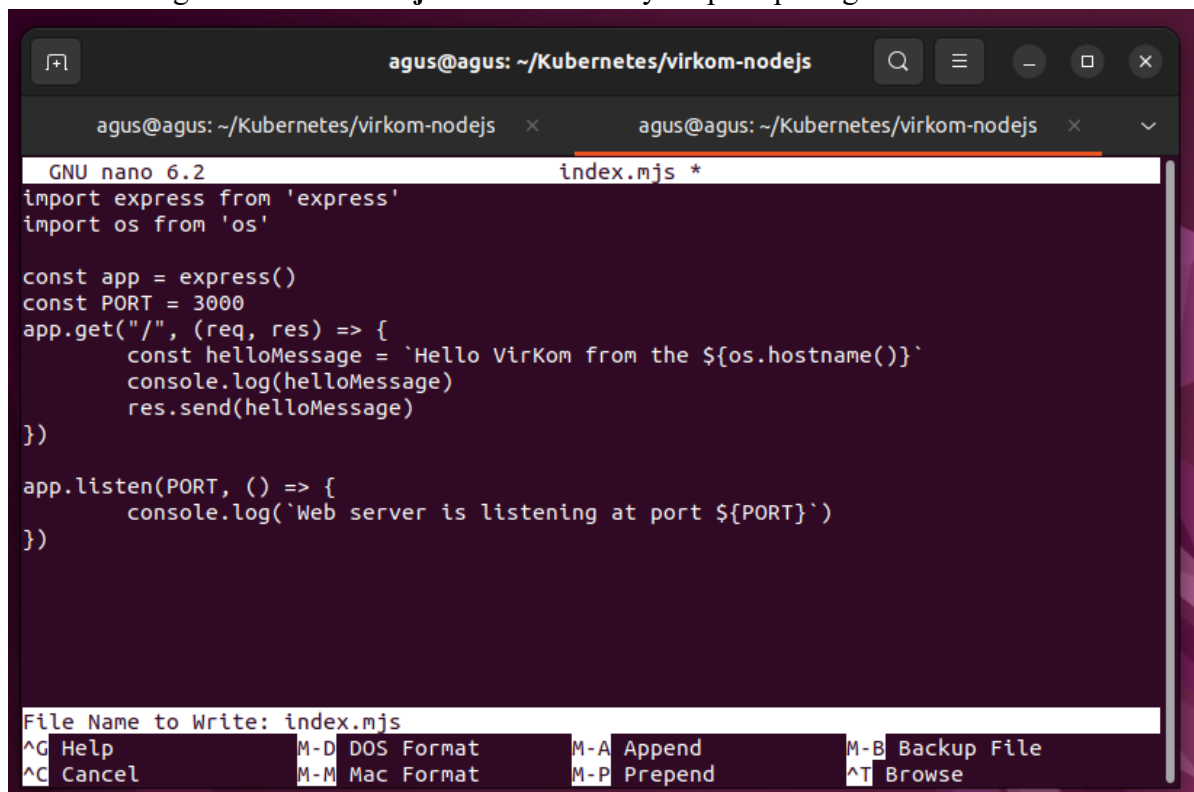


```
agus@Agus: ~/Kubernetes/virkom-nodejs
agus@Agus:~/Kubernetes/virkom-nodejs$ npm install express
added 65 packages, and audited 66 packages in 6s

13 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities
agus@Agus:~/Kubernetes/virkom-nodejs$
```

6. Buat file dengan nama **index.mjs** dan isikan filenya seperti pada gambar berikut.



```
GNU nano 6.2 index.mjs *
import express from 'express'
import os from 'os'

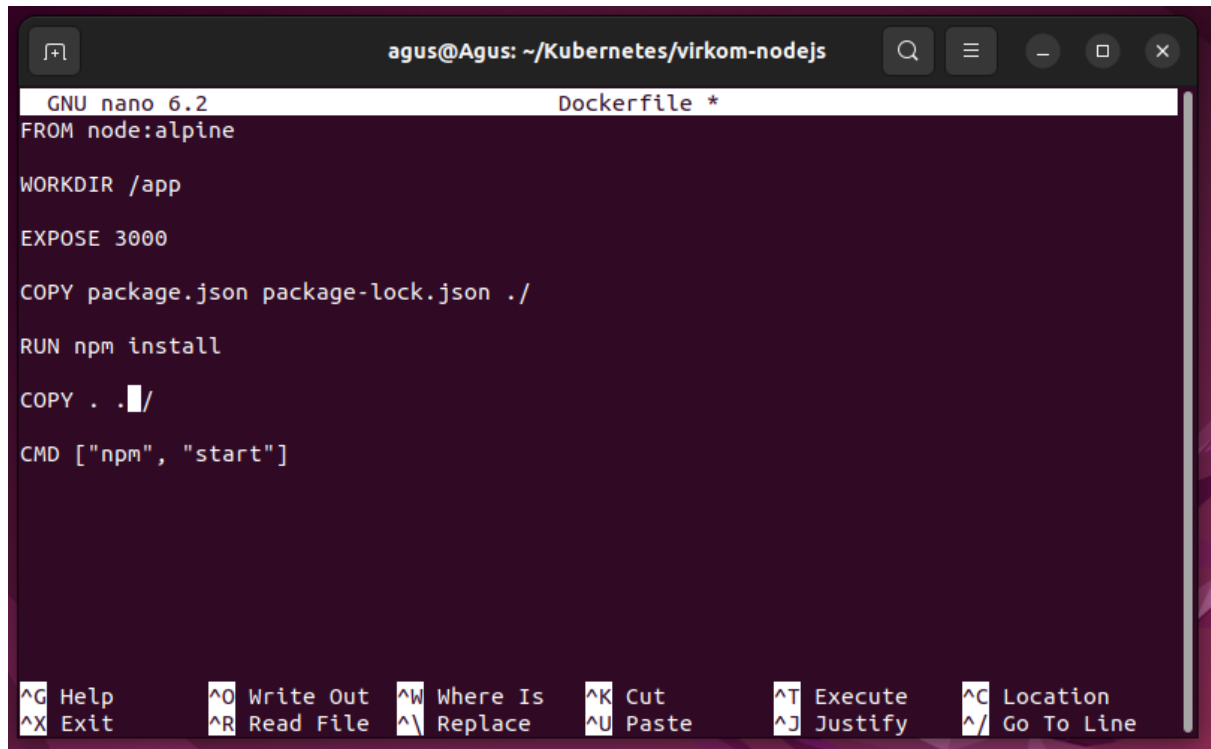
const app = express()
const PORT = 3000
app.get("/", (req, res) => {
  const helloMessage = `Hello VirKom from the ${os.hostname()}`
  console.log(helloMessage)
  res.send(helloMessage)
})

app.listen(PORT, () => {
  console.log(`Web server is listening at port ${PORT}`)
})

File Name to Write: index.mjs
^G Help      M-D DOS Format  M-A Append      M-B Backup File
^C Cancel    M-M Mac Format  M-P Prepend     ^T Browse
```

Lalu Kita simpan filenya.

7. Sekarang Kita akan diminta untuk mendockerize websitemu dengan membuatnya kedalam **Dockerfile**.



The screenshot shows a terminal window with the title bar "agus@Agus: ~/Kubernetes/virkom-nodejs". The editor is GNU nano 6.2, editing a file named "Dockerfile *". The content of the Dockerfile is as follows:

```
FROM node:alpine

WORKDIR /app

EXPOSE 3000

COPY package.json package-lock.json ./

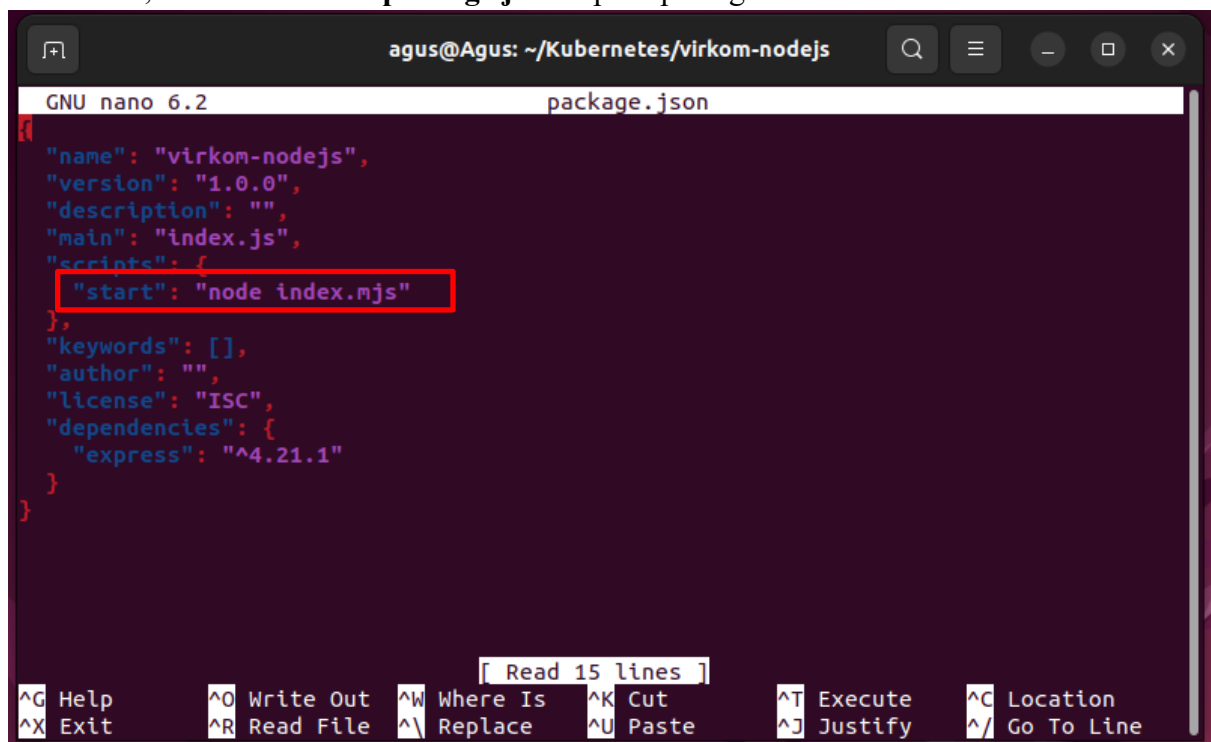
RUN npm install

COPY . ./

CMD ["npm", "start"]
```

The bottom status bar of the nano editor shows various keyboard shortcuts: ^G Help, ^X Exit, ^O Write Out, ^R Read File, ^W Where Is, ^_ Replace, ^K Cut, ^U Paste, ^T Execute, ^J Justify, ^C Location, and ^_ Go To Line.

8. Setelah itu, Kita ubah isi file **package.json** seperti pada gambar berikut.



The screenshot shows a terminal window with the title bar "agus@Agus: ~/Kubernetes/virkom-nodejs". The editor is GNU nano 6.2, editing a file named "package.json". The content of the package.json file is as follows:

```
{
  "name": "virkom-nodejs",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  "scripts": {
    "start": "node index.mjs"
  },
  "keywords": [],
  "author": "",
  "license": "ISC",
  "dependencies": {
    "express": "^4.21.1"
  }
}
```

The line `"start": "node index.mjs"` in the "scripts" object is highlighted with a red rectangle. The bottom status bar of the nano editor shows the same keyboard shortcuts as the previous image, with an additional "[Read 15 lines]" indicator.

9. Kemudian Kita bisa menjalankan **Dockerfile** dengan menjalankan *command* berikut.

```
agus@agus: ~/Kubernetes/virkom-nodejs
Usage:  docker buildx build [OPTIONS] PATH | URL | -

Start a build
agus@agus:~/Kubernetes/virkom-nodejs$ docker build . -t agus005/virkom-nodejs
[+] Building 4.3s (11/11) FINISHED                                docker:default
=> [internal] load build definition from Dockerfile                0.1s
=> => transferring dockerfile: 173B                               0.1s
=> [internal] load metadata for docker.io/library/node:alpine     3.2s
=> [auth] library/node:pull token for registry-1.docker.io        0.0s
=> [internal] load .dockerignore                                  0.0s
=> => transferring context: 2B                                     0.0s
=> [1/5] FROM docker.io/library/node:alpine@sha256:d03e75e7ba1385c2944f4 0.0s
=> [internal] load build context                                  0.1s
=> => transferring context: 34.59kB                                0.1s
=> CACHED [2/5] WORKDIR /app                                       0.0s
=> CACHED [3/5] COPY package.json package-lock.json ./           0.0s
=> CACHED [4/5] RUN npm install                                   0.0s
=> [5/5] COPY . ./                                                0.6s
=> exporting to image                                             0.1s
=> => exporting layers                                           0.1s
=> => writing image sha256:381e6d52efb4d34f9574c0f6d36e23aa287154cc8f989 0.0s
=> => naming to docker.io/agus005/virkom-nodejs                  0.0s
agus@agus:~/Kubernetes/virkom-nodejs$
```

10. Maka akan terbentuk sebuah custom image yang sudah Kita buat sebelumnya.

```
agus@agus:~/Kubernetes/virkom-nodejs$ docker images
```

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
agus005/virkom-nodejs	latest	381e6d52efb4	28 seconds ago	166MB
virkom-project2-web_php	latest	507e22e154e6	11 days ago	469MB
<none>	<none>	53551baa0344	11 days ago	190MB
image-agus	latest	c043ea34d940	11 days ago	190MB
<none>	<none>	ee392cff47c7	11 days ago	190MB
<none>	<none>	415be7c4c947	11 days ago	190MB
mongo	latest	df3f01eba940	5 weeks ago	854MB
ubuntu	latest	fec8bfd95b54	6 weeks ago	78.1M
B				
mysql	8.0	9f4b39935f20	6 weeks ago	590MB
nginx	latest	60c8a892f36f	8 weeks ago	192MB
gcr.io/k8s-minikube/kicbase	v0.0.45	aeed0e1d4642	2 months ago	1.28G
B				
httpd	latest	dad6ca1caf78	4 months ago	148MB
mongo	4	d896c071ac69	9 months ago	427MB
php	7.3-apache	35da9118b3c0	2 years ago	451MB

11. Lalu Kita juga bisa melakukan push custom image yang Kita buat ke Docker Hub. Pertama sekali, Kita harus login terlebih dahulu ke Docker Hub dengan menjalankan *command* berikut.

```
docker login
```

```
agus@agus:~/Kubernetes/virkom-nodejs$ docker login

USING WEB-BASED LOGIN
To sign in with credentials on the command line, use 'docker login -u <username>'

Your one-time device confirmation code is: DFJN-VQFB
Press ENTER to open your browser or submit your device code here: https://login.docker.com/activate

Waiting for authentication in the browser...
WARNING! Your password will be stored unencrypted in /home/agus/.docker/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credential-stores

Login Succeeded
```

Kita masukkan username dan password untuk akun Docker Hub yang sudah Kita buat sebelumnya.

12. Kemudian Kita bisa melakukan push pada image Kita dengan menjalankan *command* berikut.

Note: Sesuaikan dengan username yang sudah Kita buat sebelumnya.

```
docker push agus005/virkom-nodejs
```

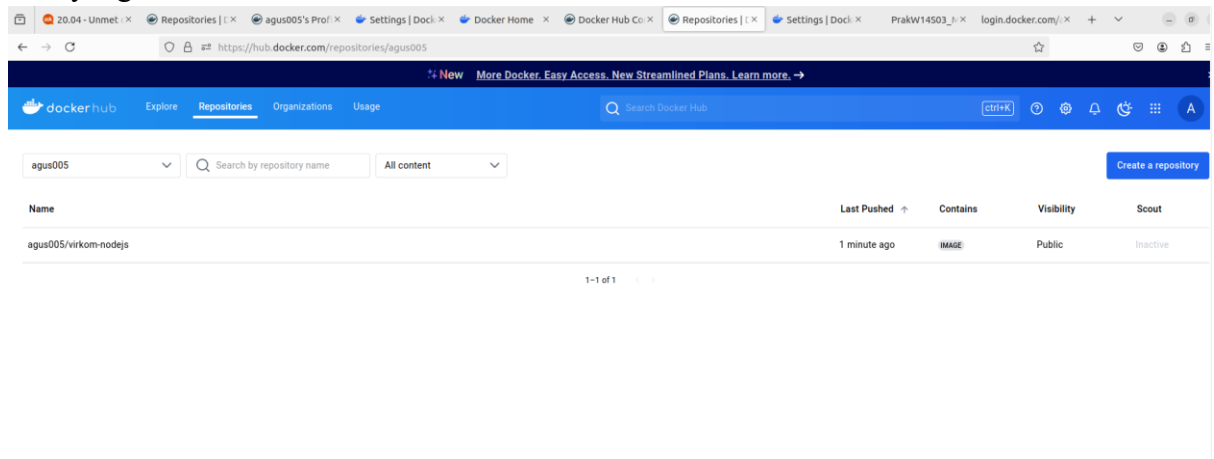
```
agus@agus: ~/Kubernetes/virkom-nodejs

Your one-time device confirmation code is: NMWQ-CQBN
Press ENTER to open your browser or submit your device code here: https://login.docker.com/activate

Waiting for authentication in the browser...
WARNING! Your password will be stored unencrypted in /home/agus/.docker/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credential-stores

Login Succeeded
agus@agus:~/Kubernetes/virkom-nodejs$ docker push agus005/virkom-nodejs
Using default tag: latest
The push refers to repository [docker.io/agus005/virkom-nodejs]
82f0997541e9: Pushed
70a8d42dbe54: Pushed
582a36a4ecd6: Pushed
77d4df81e3bd: Pushed
4abdbe5986fd: Pushed
f71a23576e0e: Pushing   5.61MB
93d7151534df: Pushing   4.96MB/147.2MB
75654b8eeebd: Pushing   3.691MB/7.798MB
```


13. Lalu Kita verifikasi apakah image yang baru Kita push sudah masuk kedalam repository Kita yang ada di Docker Hub.



Terlihat bahwa image berhasil di push.

14. Sekarang Kita akan diminta untuk membuat sebuah deployment yang baru dari image yang sudah Kita pull sebelumnya.

```
agus@agus:~/Kubernetes/virkom-nodejs$ kubectl create deployment virkom-nodejs-dep --image=agus005/virkom-nodejs
deployment.apps/virkom-nodejs-dep created
agus@agus:~/Kubernetes/virkom-nodejs$
```

15. Kita lihat apakah deployment dan pods berhasil terbuat dan berjalan.

```
agus@agus: ~/Kubernetes/virkom-nodejs
75654b8eeebd: Layer already exists
latest: digest: sha256:95d1370f6e88b8615eceba2cb95056bd5a3b29b9f7dd0cff6df10d9281259
abc size: 1993
agus@agus:~/Kubernetes/virkom-nodejs$ kubectl delete deployment --all --namespace=default
deployment.apps "virkom-nodejs-dep" deleted
agus@agus:~/Kubernetes/virkom-nodejs$ kubectl create deployment virkom-nodejs-dep --image=agus005/virkom-nodejs
deployment.apps/virkom-nodejs-dep created
agus@agus:~/Kubernetes/virkom-nodejs$ kubectl get deployments
NAME                READY    UP-TO-DATE    AVAILABLE    AGE
virkom-nodejs-dep   0/1      1             0            5s
agus@agus:~/Kubernetes/virkom-nodejs$ kubectl logs deployment/virkom-nodejs-dep
Error from server (BadRequest): container "virkom-nodejs" in pod "virkom-nodejs-dep-6cb45b7f77-4s4ds" is waiting to start: ContainerCreating
agus@agus:~/Kubernetes/virkom-nodejs$ kubectl get deployments
NAME                READY    UP-TO-DATE    AVAILABLE    AGE
virkom-nodejs-dep   1/1      1             1            17s
agus@agus:~/Kubernetes/virkom-nodejs$ kubectl get pods
NAME                                READY    STATUS    RESTARTS    AGE
virkom-nodejs-dep-6cb45b7f77-4s4ds  1/1      Running   0            2m46s
agus@agus:~/Kubernetes/virkom-nodejs$
```

16. Kemudian Kita juga diminta untuk membuat cluster IP untuk menghubungkan web server dengan node js dengan menjalankan *command* berikut.

```
agus@agus:~/Kubernetes/virkom-nodejs$ kubectl expose deployment virkom-nodejs-dep --port=3000
service/virkom-nodejs-dep exposed
agus@agus:~/Kubernetes/virkom-nodejs$
```

17. Lalu Kita verifikasi service yang baru dibuat.

```
agus@agus:~/Kubernetes/virkom-nodejs$ kubectl get services
NAME                                TYPE                CLUSTER-IP      EXTERNAL-IP      PORT(S)
agus-nodejs-dep-exposed             ClusterIP            10.103.243.122  <none>           3000/TCP
kubernetes                          ClusterIP            10.96.0.1       <none>           443/TCP
nodeapp-service                    LoadBalancer        10.108.94.226   <pending>        5000:311
10/TCP
virkom-nodejs-dep                  ClusterIP            10.101.227.181  <none>           3000/TCP
10s
```

18. Sekarang Kita coba untuk terhubung kedalam cluster IP dan juga port **3000** melalui node yang sudah Kita buat sebelumnya.

```
agus@agus:~/Kubernetes/virkom-nodejs$ minikube ssh
docker@minikube:~$ curl 10.108.94.226:3000
^X^[[A^C
docker@minikube:~$ curl 10.103.243.122:300
^X^C
docker@minikube:~$ curl 10.103.243.122:3000
curl: (7) Failed to connect to 10.103.243.122 port 3000 after 0 ms: Connection refused
docker@minikube:~$ curl 10.102.2.182:3000
Hello VirKom from the virkom-nodejs-dep-6cb45b7f77-4s4dsdocker@minikube:~$
```

19. Kemudian Kita akan melakukan scaling pada deployment yang sudah Kita buat sebelumnya yang dimana terdapat hanya 1 pod menjadi 4 pod.

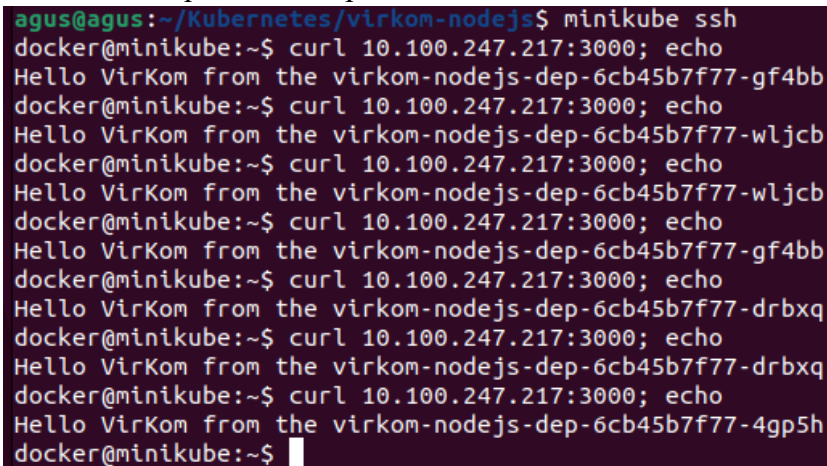
```
agus@agus:~/Kubernetes/virkom-nodejs$ kubectl scale deployment virkom-nodejs-dep --replicas=4
deployment.apps/virkom-nodejs-dep scaled
agus@agus:~/Kubernetes/virkom-nodejs$
```

20. Maka akan terdapat 4 pod yang terbentuk.

```
agus@agus:~/Kubernetes/virkom-nodejs$ kubectl get pods -o wide
NAME                                READY  STATUS   RESTARTS   AGE   IP
NODE    NOMINATED NODE  READINESS GATES
virkom-nodejs-dep-6cb45b7f77-4s4ds  1/1    Running  0          42m   10.244.0.36
minikube <none>          <none>
virkom-nodejs-dep-6cb45b7f77-d27p9  1/1    Running  0          21s   10.244.0.37
minikube <none>          <none>
virkom-nodejs-dep-6cb45b7f77-k6s92  1/1    Running  0          21s   10.244.0.38
minikube <none>          <none>
virkom-nodejs-dep-6cb45b7f77-nwch2  1/1    Running  0          21s   10.244.0.39
minikube <none>          <none>
agus@agus:~/Kubernetes/virkom-nodejs$
```

Dimana masing-masing pod akan memiliki hash yang specific dan IP yang unik atau berbeda dari pod lainnya.

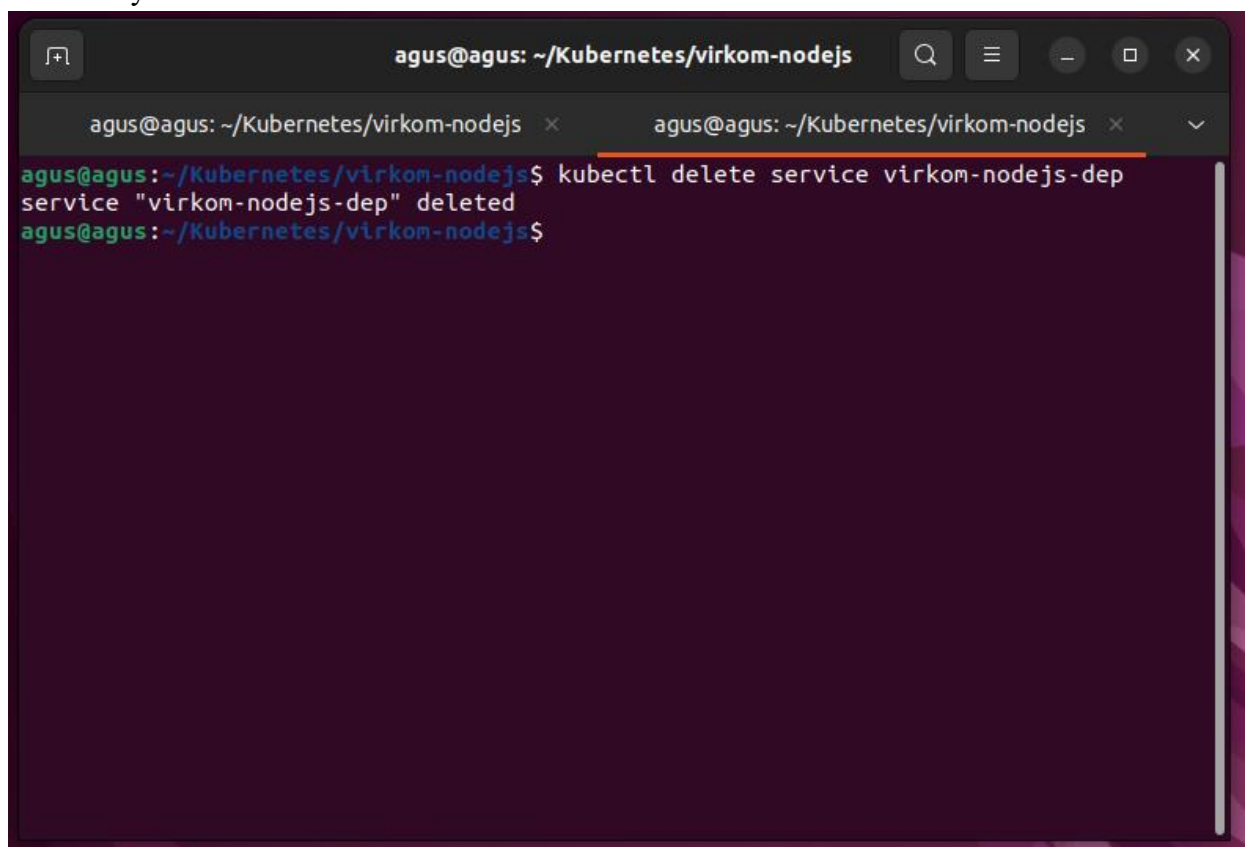
21. Kemudian Kita akan coba untuk mengakses clusterIP dari yang sudah Kita buat sebelumnya untuk melihat pod mana yang akan memberikan response disaat Kita melakukan request terhadap clusterIP.



```
agus@agus:~/Kubernetes/virkom-nodejs$ minikube ssh
docker@minikube:~$ curl 10.100.247.217:3000; echo
Hello VirKom from the virkom-nodejs-dep-6cb45b7f77-gf4bb
docker@minikube:~$ curl 10.100.247.217:3000; echo
Hello VirKom from the virkom-nodejs-dep-6cb45b7f77-wljcb
docker@minikube:~$ curl 10.100.247.217:3000; echo
Hello VirKom from the virkom-nodejs-dep-6cb45b7f77-gf4bb
docker@minikube:~$ curl 10.100.247.217:3000; echo
Hello VirKom from the virkom-nodejs-dep-6cb45b7f77-drxbq
docker@minikube:~$ curl 10.100.247.217:3000; echo
Hello VirKom from the virkom-nodejs-dep-6cb45b7f77-drxbq
docker@minikube:~$ curl 10.100.247.217:3000; echo
Hello VirKom from the virkom-nodejs-dep-6cb45b7f77-4gp5h
docker@minikube:~$
```

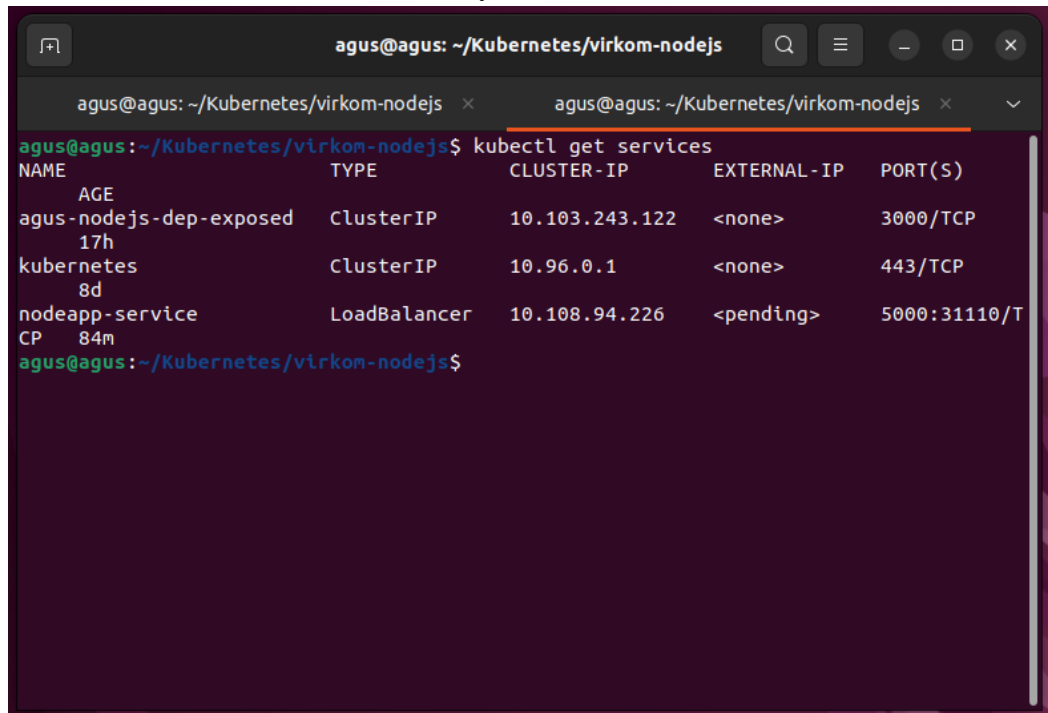
Terlihat pada gambar berikut, beberapa pods memberikan response yang berbeda saat Kita mencoba untuk melakukan request pada clusterIP yang sama, karena seperti itulah cara kerja pendistribusian *load*.

22. Lalu Kita coba membuat service baru dengan tipe NodePort dikarenakan untuk tipe ClusterIP hanya berfungsi didalam cluster saja. Dengan tipe NodePort, Kita mengekspos aplikasi yang berjalan di dalam cluster ke luar cluster melalui port tertentu. Namun terlebih dahulu Kita bisa menghapus service dengan tipe ClusterIP yang sudah Kita buat sebelumnya.



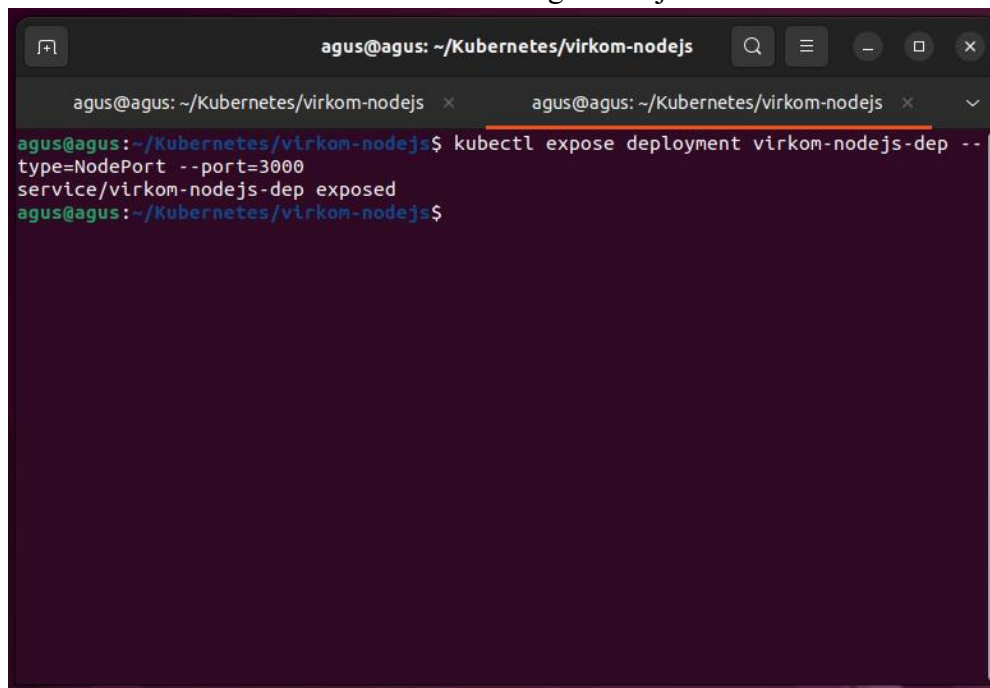
```
agus@agus: ~/Kubernetes/virkom-nodejs
agus@agus:~/Kubernetes/virkom-nodejs$ kubectl delete service virkom-nodejs-dep
service "virkom-nodejs-dep" deleted
agus@agus:~/Kubernetes/virkom-nodejs$
```

23. Maka tidak akan ditemukan servicenya.



```
agus@agus: ~/Kubernetes/virkom-nodejs
agus@agus: ~/Kubernetes/virkom-nodejs$ kubectl get services
NAME                                TYPE                CLUSTER-IP      EXTERNAL-IP      PORT(S)
agus-nodejs-dep-exposed             ClusterIP            10.103.243.122  <none>           3000/TCP
kubernetes                          ClusterIP            10.96.0.1       <none>           443/TCP
nodeapp-service                    LoadBalancer        10.108.94.226   <pending>        5000:31110/T
```

24. Maka Kita bisa membuat service baru dengan menjalankan *command* berikut.



```
agus@agus: ~/Kubernetes/virkom-nodejs
agus@agus: ~/Kubernetes/virkom-nodejs$ kubectl expose deployment virkom-nodejs-dep --
type=NodePort --port=3000
service/virkom-nodejs-dep exposed
agus@agus: ~/Kubernetes/virkom-nodejs$
```

25. Maka akan ditemukan service yang baru Kita buat.

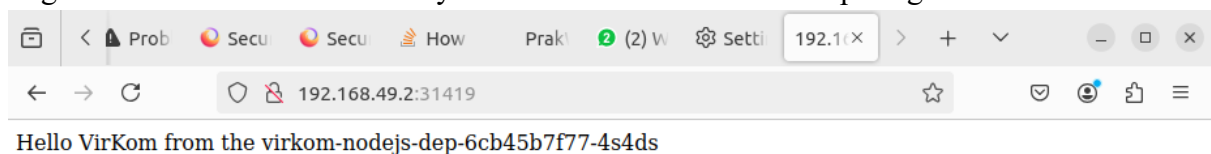
```
agus@agus:~/Kubernetes/virkom-nodejs$ kubectl get services
NAME                                TYPE                CLUSTER-IP      EXTERNAL-IP      PORT(S)
ag-us-nodejs-dep-exposed            ClusterIP            10.103.243.122  <none>           3000/TCP
kubernetes                          ClusterIP            10.96.0.1       <none>           443/TCP
nodeapp-service                    LoadBalancer        10.108.94.226   <pending>        5000:31110/T
virkom-nodejs-dep                  NodePort             10.99.252.66    <none>           3000:31419/T
agus@agus:~/Kubernetes/virkom-nodejs$
```

Pada gambar diatas akan terdapat keterangan type yaitu NodePort dan juga random port, yaitu 31344.

26. Lalu Kita akan coba untuk terhubung ke salah satu pod yang sudah Kita buat, melalui IP Address yang ada pada node Kita dan menggunakan random port yang tergenerate. Kita bisa mengetahui IP dari node Kita dengan menjalankan *command* berikut.

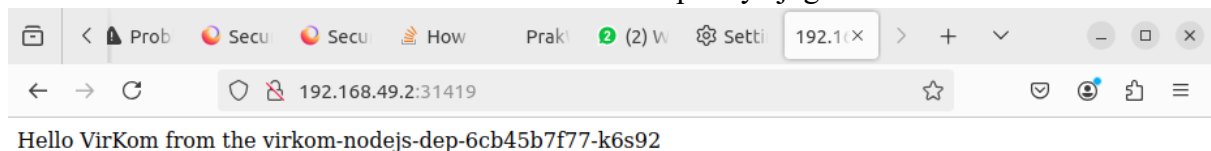
```
agus@agus:~/Kubernetes/virkom-nodejs$ minikube ip
192.168.49.2
agus@agus:~/Kubernetes/virkom-nodejs$
```

27. Kemudian coba akses IP Address tersebut bersama dengan random port yang sudah tergenerate dari service sebelumnya di web browser local Kita seperti gambar berikut.

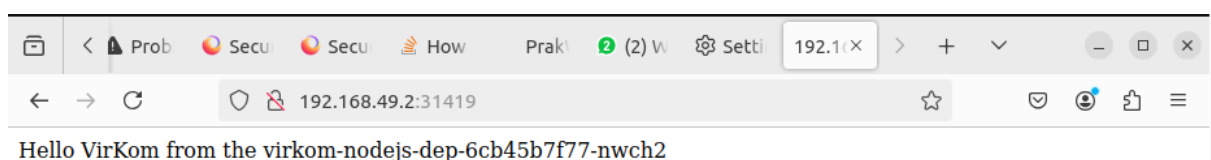


The screenshot shows a web browser window with the address bar set to `192.168.49.2:31419`. The page content displays the message: "Hello VirKom from the virkom-nodejs-dep-6cb45b7f77-4s4ds".

28. Jika Kita melakukan refresh terus menerus maka responnya juga akan berbeda.

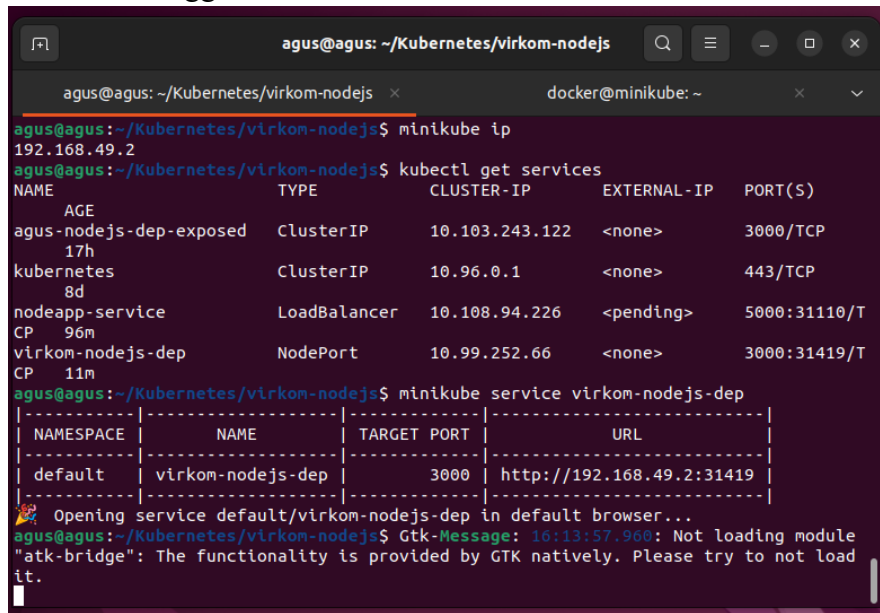


The screenshot shows a web browser window with the address bar set to `192.168.49.2:31419`. The page content displays the message: "Hello VirKom from the virkom-nodejs-dep-6cb45b7f77-k6s92".



The screenshot shows a web browser window with the address bar set to `192.168.49.2:31419`. The page content displays the message: "Hello VirKom from the virkom-nodejs-dep-6cb45b7f77-nwch2".

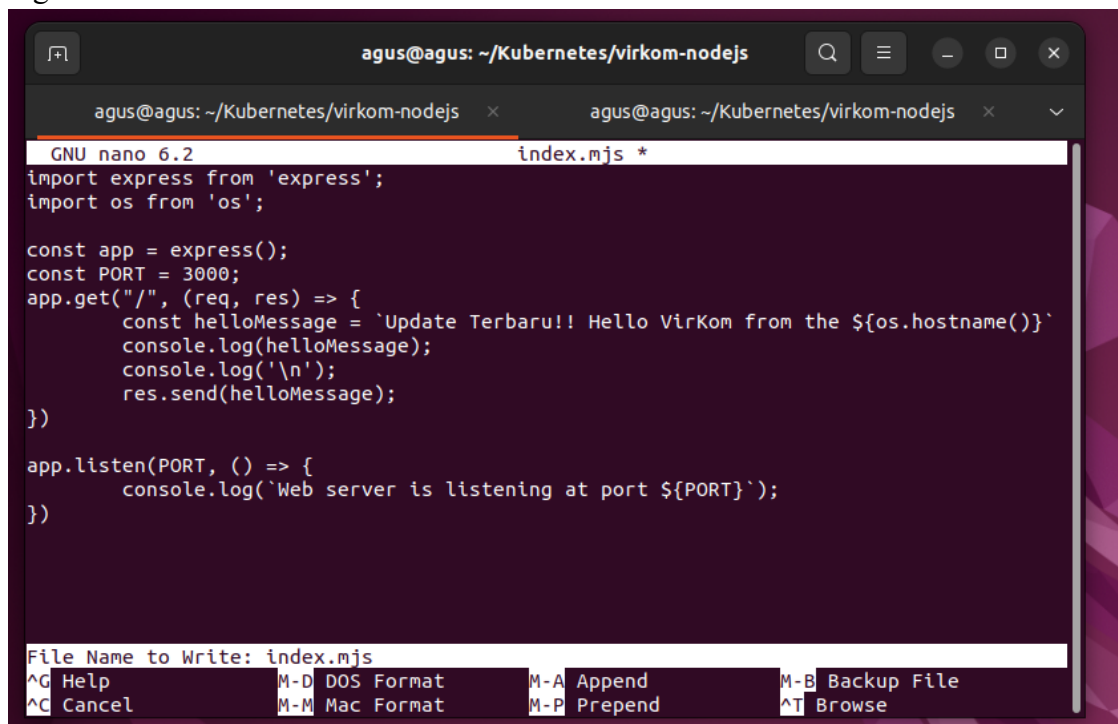
29. Kita juga bisa menjalankan *command* berikut untuk mengakses URL dari IP Address node tersebut sehingga web browser akan otomatis terbuka.



```
agus@agus: ~/Kubernetes/virkom-nodejs
agus@agus: ~/Kubernetes/virkom-nodejs$ minikube ip
192.168.49.2
agus@agus: ~/Kubernetes/virkom-nodejs$ kubectl get services
NAME                                TYPE          CLUSTER-IP    EXTERNAL-IP    PORT(S)
agus-nodejs-dep-exposed             ClusterIP     10.103.243.122 <none>         3000/TCP
kubernetes                          ClusterIP     10.96.0.1      <none>         443/TCP
nodeapp-service                    LoadBalancer 10.108.94.226  <pending>      5000:31110/T
virkom-nodejs-dep                  NodePort      10.99.252.66   <none>         3000:31419/T
agus@agus: ~/Kubernetes/virkom-nodejs$ minikube service virkom-nodejs-dep
-----
| NAMESPACE | NAME           | TARGET PORT | URL               |
|-----|-----|-----|-----|
| default | virkom-nodejs-dep | 3000 | http://192.168.49.2:31419 |
|-----|-----|-----|-----|
Opening service default/virkom-nodejs-dep in default browser...
agus@agus: ~/Kubernetes/virkom-nodejs$
```



30. Setelah itu, Kita diminta untuk melakukan update terhadap image yang Kita buat sebelumnya. Kita bisa melakukan edit pada file **index.mjs** dengan menambahkan line yang ingin Kita tambahkan.



```
GNU nano 6.2 index.mjs *
import express from 'express';
import os from 'os';

const app = express();
const PORT = 3000;
app.get("/", (req, res) => {
  const helloMessage = `Update Terbaru!! Hello VirKom from the ${os.hostname()}`;
  console.log(helloMessage);
  console.log('\n');
  res.send(helloMessage);
})

app.listen(PORT, () => {
  console.log(`Web server is listening at port ${PORT}`);
})

File Name to Write: index.mjs
^G Help      M-D DOS Format  M-A Append     M-B Backup File
^C Cancel    M-M Mac Format  M-P Prepend    ^T Browse
```

Kita simpan perubahan filenya.

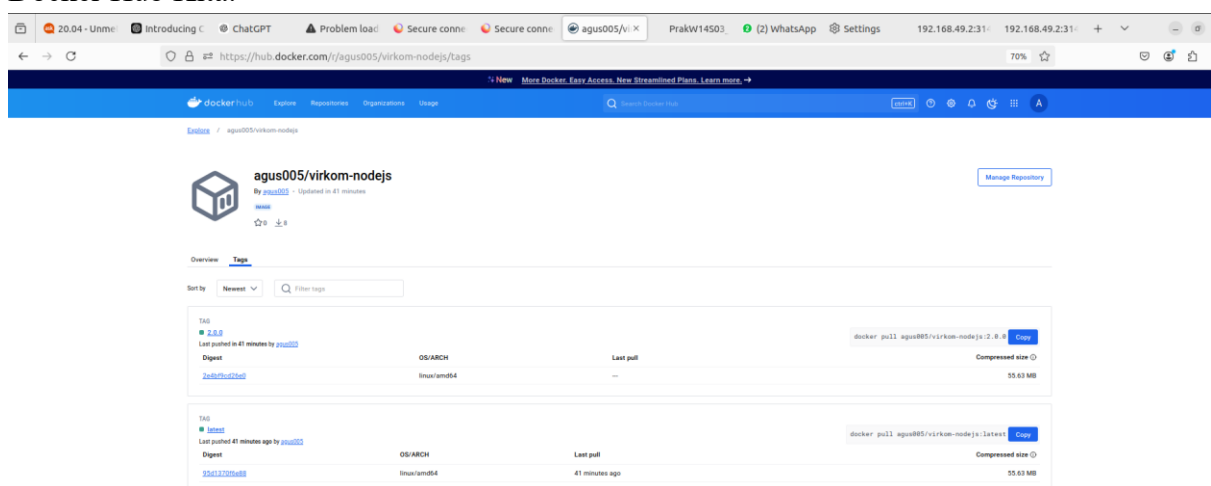
31. Lalu Kita buat image baru dengan tag yang baru.

```
agus@agus:~/Kubernetes/virkom-nodejs$ docker build . -t agus005/virkom-nodejs:2.0.0
[+] Building 3.3s (2/3)                                docker:default
=> [internal] load build definition from Dockerfile      0.0s
=> => transferring dockerfile: 173B                    0.0s
=> [internal] load metadata for docker.io/library/node:alpine 3.2s
=> [auth] library/node:pull token for registry-1.docker.io 0.0s
```

32. Kemudian Kita sekarang sudah bisa melakukan push pada image yang baru saja Kita update.

```
agus@agus:~/Kubernetes/virkom-nodejs$ docker push agus005/virkom-nodejs:2.0.0
The push refers to repository [docker.io/agus005/virkom-nodejs]
791508244604: Pushing 2.719MB
70a8d42dbe54: Layer already exists
582a36a4ecd6: Layer already exists
77d4df81e3bd: Layer already exists
4abdbe5986fd: Layer already exists
f71a23576e0e: Layer already exists
93d7151534df: Waiting
75654b8eeebd: Waiting
```

33. Kita bisa mengecek apakah image yang baru Kita push sudah masuk kedalam repository Docker Hub Kita.



Terlihat bahwa image baru berhasil di push.

34. Lalu Kita tetapkan image yang baru Kita push ke deployment dengan menjalankan *command* berikut.

```
kubectl set image deployment virkom-nodejs-dep virkom-nodejs=agus005/virkom-nodejs:2.0.0
```

```
agus@agus:~/Kubernetes/virkom-nodejs$ kubectl set image deployment virkom-nodejs-dep virkom-nodejs=agus005/virkom-nodejs:2.0.0
deployment.apps/virkom-nodejs-dep image updated
agus@agus:~/Kubernetes/virkom-nodejs$
```

35. Kemudian Kita bisa melakukan rollout pada deployment dengan menjalankan *command* berikut.

```
kubectl rollout status deployment virkom-nodejs-dep
```

```
agus@agus:~/Kubernetes/virkom-nodejs$ kubectl rollout status deployment virkom-nodejs-dep
deployment "virkom-nodejs-dep" successfully rolled out
agus@agus:~/Kubernetes/virkom-nodejs$
```

36. Maka pod yang lama akan diberhentikan dan pod yang baru akan dibuat untuk menggantikan pod yang lama.

```
agus@agus:~/Kubernetes/virkom-nodejs$ kubectl get pods
NAME                                READY   STATUS    RESTARTS   AGE
virkom-nodejs-dep-7864dc4bfc-798ft  1/1     Running   0           3m14s
virkom-nodejs-dep-7864dc4bfc-jwj9w  1/1     Running   0           3m10s
virkom-nodejs-dep-7864dc4bfc-qjtnb  1/1     Running   0           3m25s
virkom-nodejs-dep-7864dc4bfc-w6dcl  1/1     Running   0           3m25s
agus@agus:~/Kubernetes/virkom-nodejs$
```

37. Lalu Kita bisa verifikasi perubahannya dengan menjalankan *command* berikut.

The screenshot shows a terminal window on the left and a web browser on the right. The terminal displays the command `minikube service virkom-nodejs-dep` and its output, which includes a table of pod details and the URL `http://192.168.49.2:31419`. The browser on the right shows the page `Update Terbaru!! Hello VirKom from the virkom-nodejs-dep-7864dc4bfc-w6dcl`.

NAME	READY	STATUS	RESTARTS	AGE
virkom-nodejs-dep-7864dc4bfc-798ft	1/1	Running	0	3m14s
virkom-nodejs-dep-7864dc4bfc-jwj9w	1/1	Running	0	3m10s
virkom-nodejs-dep-7864dc4bfc-qjtnb	1/1	Running	0	3m25s
virkom-nodejs-dep-7864dc4bfc-w6dcl	1/1	Running	0	3m25s

NAMESPACE	NAME	TARGET PORT	URL
default	virkom-nodejs-dep	3000	http://192.168.49.2:31419

38. Jika Kita ingin balik ke settingan yang sebelumnya Kita bisa melakukan hal yang sama yaitu menentukan tag yang Kita gunakan sebelumnya, lalu lakukan perubahan.

39. Selesai!