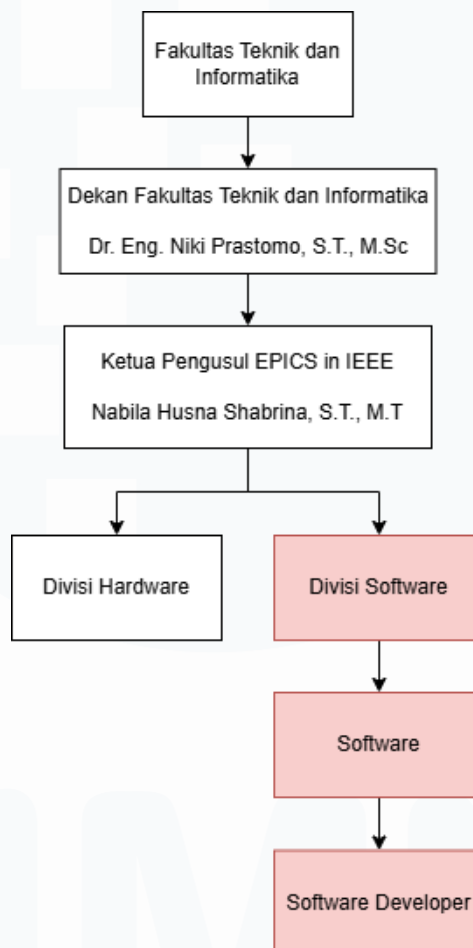


BAB III

PELAKSANAAN KERJA MAGANG

3.1 Kedudukan dan Koordinasi



Gambar 3.1 Bagan Struktur Kedudukan Magang

Universitas Multimedia Nusantara (UMN) membuat proyek *An AIoT-powered Smart Agricultural System for Pests Forecasting and Management* dengan pendanaan dari EPICS in IEEE. Struktur kedudukan magang ini bisa dilihat pada Gambar 3.1, proyek diketuai oleh Ibu Nabila Husna Shabrina sebagai Ketua Pengusul EPICS in IEEE dan juga Pembimbing kerja magang. Pekerjaan proyek magang ini dibagi menjadi dua bagian, divisi *hardware*, dan divisi *software*. Divisi

software memiliki bagian *software* yang di isi dengan *software* developer, bagian *software* bertanggung jawab atas pengembangan dan pembuatan aplikasi MySalak.

Dalam program magang ini, penulis mendapatkan posisi sebagai *software* developer khususnya pada bagian *deployment* aplikasi dari periode *development* ke *production*. *Software Developer* memiliki tugas seperti merancang dan mengimplementasikan proses *deployment* aplikasi dari tahap pengembangan hingga produksi, memastikan aplikasi dapat berjalan dengan optimal di lingkungan produksi, mengelola infrastruktur dan otomatisasi *deployment* untuk meningkatkan efisiensi, serta memantau kinerja aplikasi guna menjaga stabilitas dan skalabilitas sistem. Pada proyek ini, penulis memiliki tugas untuk *deploy* aplikasi ke *website* dan memastikan aplikasi berjalan dengan normal.

3.2 Tugas dan Uraian Kerja Magang

3.2.1 Tugas

Sebelum aplikasi MySalak dapat di-*deploy*, diperlukan penelitian terlebih dahulu mengenai spesifikasi *Virtual Private Server* (VPS) yang sesuai untuk menampung dan menjalankan aplikasi tersebut. Setelah spesifikasi VPS ditentukan, perlu dilakukan pemilihan dan registrasi nama domain yang akan digunakan untuk *website* aplikasi.

Untuk memastikan aplikasi berjalan dengan stabil dan menghindari perbedaan lingkungan antara *development* dan *production*, digunakan perangkat lunak *Docker*. *Source code* dari MySalak akan diambil (pull) dari repositori utama menggunakan *GitHub*. *Source code* ini terbagi menjadi tiga komponen utama, yaitu *frontend*, *backend*, dan *Application Programming Interface* (API).

Bagian *frontend* merupakan antarmuka yang akan diakses dan digunakan oleh pengguna. *Backend* berfungsi sebagai inti sistem yang bertanggung jawab dalam pengelolaan data, pemrosesan logika bisnis, serta interaksi dengan basis

data. Sementara itu, API memungkinkan komunikasi antara backend dengan perangkat keras dan sistem *Artificial Intelligence*.

Setelah itu, VPS akan dikonfigurasi agar dapat terhubung dengan domain yang telah didaftarkan. Setelah koneksi berhasil, VPS akan digunakan sebagai lingkungan untuk menampung dan menjalankan aplikasi MySalak.

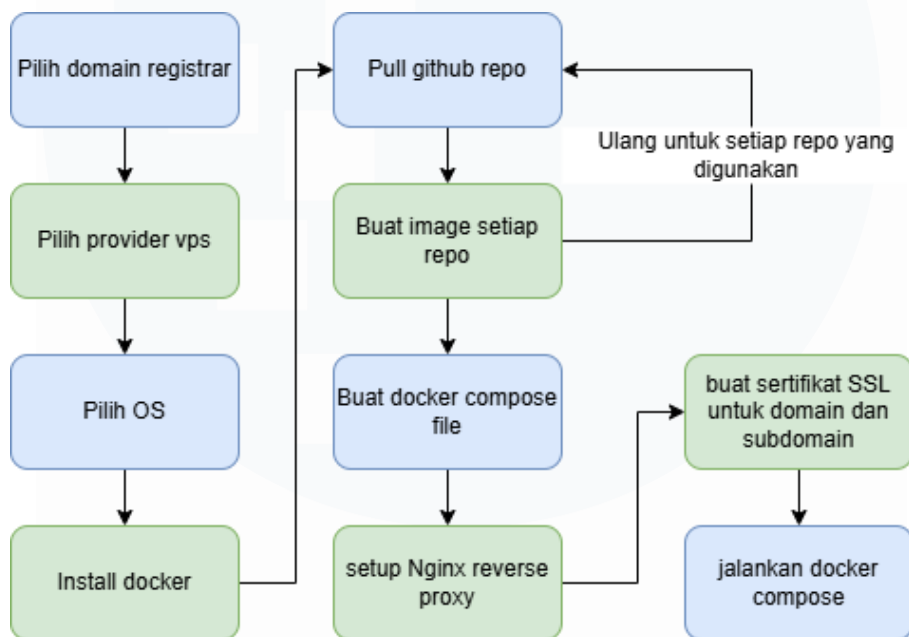
3.2.2 Uraian

Berikut adalah tabel pekerjaan magang per minggu yang dilakukan penulis selama empat bulan terakhir.

Tabel 3.1 Tabel kerja magang

MINGGU	RINCIAN TUGAS YANG DILAKUKAN
1	Melakukan perbandingan provider domain
2	Melakukan perbandingan provider VPS
3	Studi literatur penggunaan VPS
4	Setup VPS dan domain
5	Buat Docker image
6	Buat Docker image
7	Buat Docker compose
8	Setup Nginx reverse proxy
9	Deploy aplikasi MySalak untuk tes
10	Tes aplikasi, bug fixing
11	Deploy aplikasi MySalak secara resmi
12	Memastikan aplikasi berjalan tanpa error
13	Memastikan aplikasi berjalan tanpa error

14	Memastikan aplikasi berjalan tanpa error
15	Membuat laporan Magang
16	Membuat laporan Magang
17	Membuat laporan Magang



Gambar 3.2 alur kerja deploy aplikasi MySalak

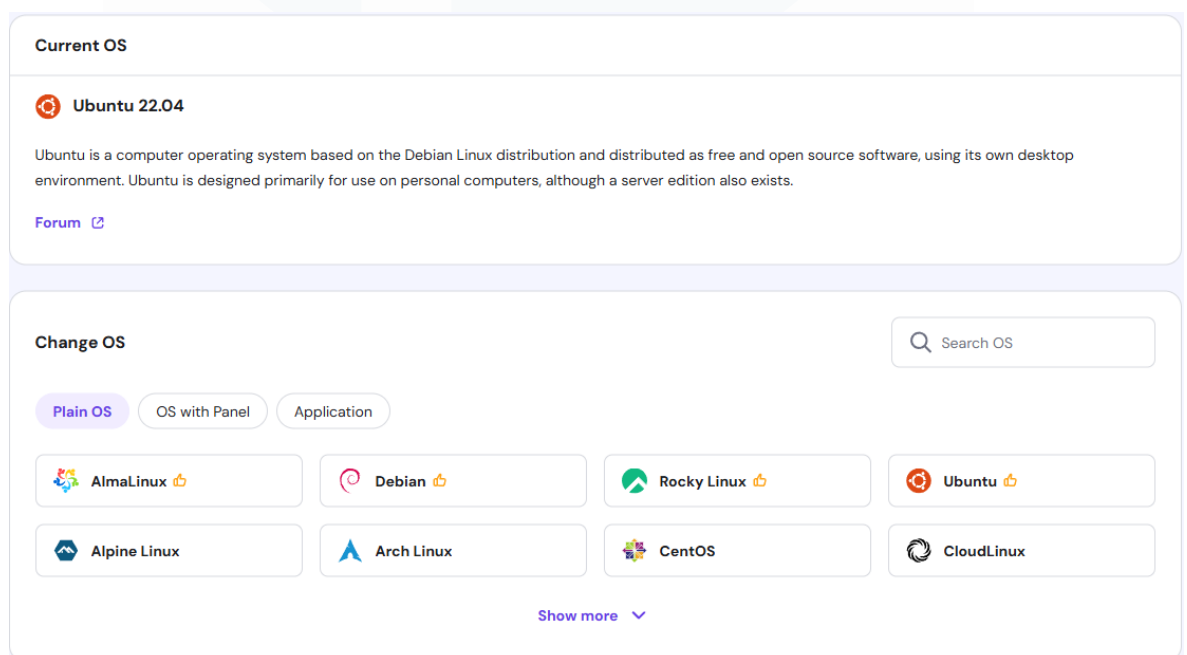
Seperti yang ditunjukkan pada Gambar 3.2, tugas penulis mencakup menyiapkan berbagai keperluan untuk deployment aplikasi. Persiapan ini meliputi konfigurasi domain agar aplikasi dapat diakses oleh pengguna, penggunaan VPS sebagai server yang terhubung ke domain, serta penerapan Docker untuk memastikan aplikasi memiliki semua dependensinya dan berjalan dalam lingkungan yang konsisten dengan pengembangannya, dan *debugging* jika Docker image tidak sesuai berjalan. Selain itu, Nginx digunakan sebagai reverse proxy untuk mengoptimalkan distribusi permintaan dan mencegah kelebihan beban pada server VPS, sementara sertifikat SSL diterapkan untuk memastikan standar keamanan pada domain. Setelah proses deployment selesai,

penulis bertanggung jawab untuk memantau kinerja aplikasi dan status VPS guna memastikan operasionalnya tetap optimal.

3.2.3 Setup VPS

Virtual Private Server (VPS) merupakan server pribadi yang disimulasikan melalui teknik virtualisasi pada sebuah server fisik berkapasitas besar, sehingga dapat digunakan oleh banyak pengguna secara terisolasi [3]. Dengan menggunakan VPS, aplikasi MySalak dapat dikembangkan lebih lanjut dengan fleksibilitas yang lebih tinggi tanpa memerlukan server fisik pribadi.

Untuk memperoleh VPS yang sesuai, dilakukan survei terhadap berbagai layanan penyedia web *hosting*. Berdasarkan hasil survei, Hostinger dipilih sebagai penyedia layanan yang paling sesuai, dengan mempertimbangkan keseimbangan antara harga yang ditawarkan dan spesifikasi VPS yang mendukung kebutuhan aplikasi MySalak.

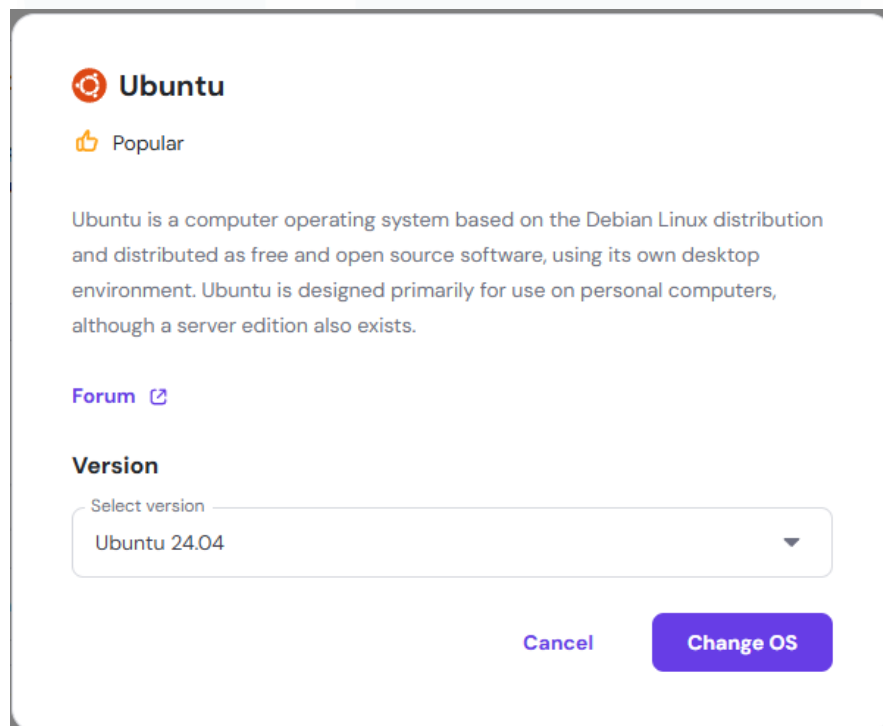


Gambar 3.3 Tampilan bagian pemasangan OS VPS

Setelah proses penyewaan VPS selesai, server belum dapat diakses karena belum memiliki sistem operasi (OS). Oleh karena itu, Ubuntu versi 22.04 dipilih

sebagai sistem operasi yang digunakan pada VPS, mengingat popularitasnya dalam lingkungan VPS serta kemudahan dalam penggunaannya.

Sebagaimana ditunjukkan pada Gambar 3.3, proses pemilihan dan pemasangan OS dilakukan dengan langkah yang sederhana. Pengguna hanya perlu memilih sistem operasi yang diinginkan, menentukan versi yang diperlukan, lalu mengeklik tombol *Change OS* sebagaimana diperlihatkan pada Gambar 3.4. Setelah itu, sistem operasi akan secara otomatis terpasang pada VPS.



Gambar 3.4 Tampilan pemilihan versi OS yang ingin dipasang

Setelah VPS dapat digunakan, langkah selanjutnya adalah mengonfigurasinya agar dapat terhubung dengan domain yang telah dimiliki. Proses ini dilakukan melalui *DNS Manager* yang dapat dilihat pada Gambar 3.5. Dengan menggunakan *DNS Manager*, domain dapat disambungkan ke VPS dengan mengganti *nameserver* yang ada pada domain menjadi *nameserver* milik Hostinger.

Add Domain

mysalak.com

- 1 Login to your domain provider**
Login and find the DNS Zone management
- 2 Change to Hostinger Nameservers**
Change your current nameservers to the Hostinger nameservers.
 ns1.dns-parking.com ns2.dns-parking.com
- 3 Confirmation**
☐ I've changed the nameservers

NS record require up to 24 hours to work worldwide. After that, you can manage juantestsite.web.id records from Hostinger.

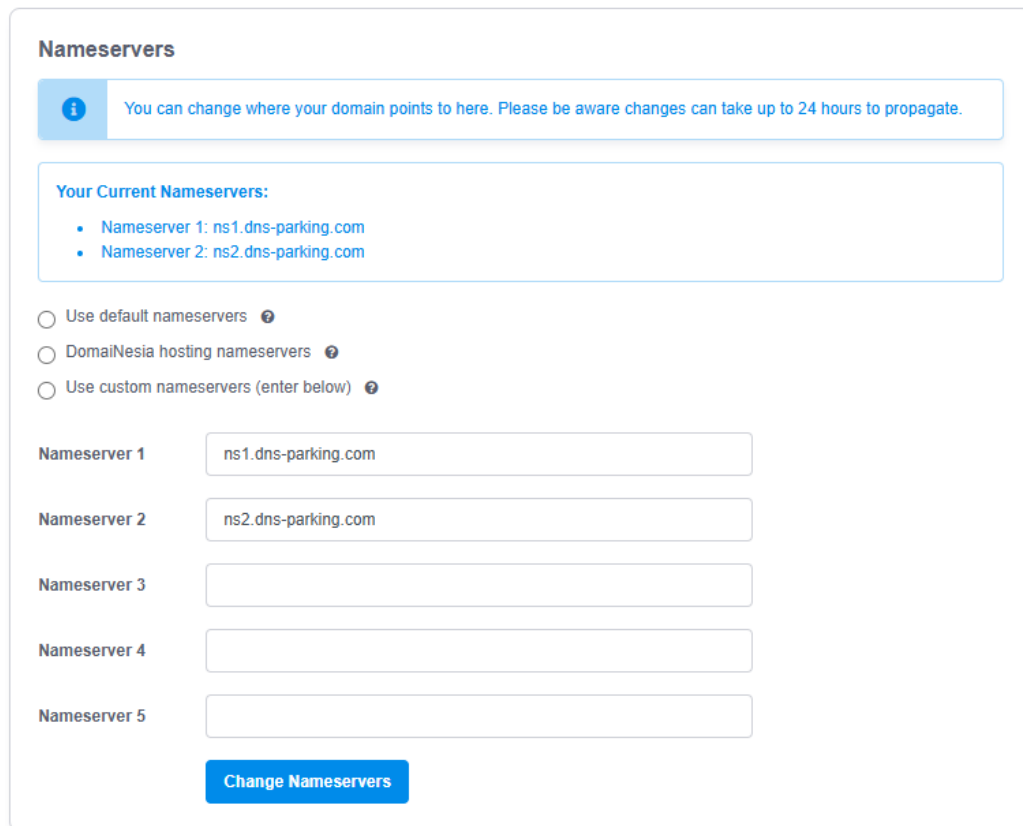
Cancel Finish

Gambar 3.5 DNS Manager VPS

3.2.4 Setup Domain

Domain merupakan alamat unik yang digunakan untuk mengakses sebuah *website* dan mempermudah pengguna dalam mengingat serta menggunakannya [4]. Tanpa domain, *website* tetap dapat diakses, tetapi pengguna harus memasukkan serangkaian angka spesifik yang merupakan alamat *Internet Protocol* (IP) dari server tempat *website* tersebut dihosting. Oleh karena itu, penggunaan domain yang berbentuk kata unik lebih disarankan karena lebih mudah diingat dan diakses oleh pengguna.

Untuk mendapatkan domain yang cocok, dilakukan survei terhadap berbagai platform penyedia layanan registrasi domain, DomaiNesia dipilih sebagai penyedia layanan yang paling sesuai berdasarkan pertimbangan harga serta ketersediaan domain yang memenuhi kebutuhan. Nama domain yang dipilih untuk digunakan dalam aplikasi ini adalah MySalak.com.



Nameservers

You can change where your domain points to here. Please be aware changes can take up to 24 hours to propagate.

Your Current Nameservers:

- Nameserver 1: ns1.dns-parking.com
- Nameserver 2: ns2.dns-parking.com

☐ Use default nameservers ?

☐ DomaiNesia hosting nameservers ?

☐ Use custom nameservers (enter below) ?

Nameserver 1: ns1.dns-parking.com

Nameserver 2: ns2.dns-parking.com

Nameserver 3:

Nameserver 4:

Nameserver 5:

Change Nameservers

Gambar 3.6 Pengantian Nameserver domain MySalak.com

Setelah proses registrasi selesai, domain dikonfigurasi agar terhubung dengan VPS yang telah disiapkan. Konfigurasi ini dilakukan dengan mengganti *nameserver* bawaan domain, sebagaimana ditunjukkan pada Gambar 3.6, dengan *nameserver* yang sesuai dengan VPS. *Nameserver* berfungsi sebagai web server yang mengarahkan domain ke server tertentu. Dalam konteks ini, *nameserver* digunakan untuk menghubungkan domain MySalak.com dengan lokasi server VPS yang telah dimiliki.

Pada Gambar 3.6 dapat dilihat bahwa terdapat dua *nameserver* yang dikonfigurasi untuk domain tersebut. Penggunaan dua *nameserver* bertujuan sebagai langkah redundansi; apabila *nameserver* utama mengalami kendala atau tidak dapat diakses, *nameserver* kedua akan secara otomatis menggantikannya hingga layanan kembali normal.

Manage DNS records

These records define how your domain behaves. Common uses include pointing your domain at web servers or configuring email delivery for your domain.

Type

A

Name

@

Points to

TTL

14400

Add Record

Q Search

Type	Name	Priority	Content	TTL		
A	www	0	91.108.111.189	3600	Delete	Edit
A	flask	0	91.108.111.189	3600	Delete	Edit
A	node	0	91.108.111.189	3600	Delete	Edit

Gambar 3.7 DNS manager domain

Untuk menghubungkan domain dengan VPS, diperlukan konfigurasi *Address Record* (Record A) pada *DNS Manager* yang dapat dilihat pada Gambar 3.7. Proses konfigurasi ini dimulai dengan memilih tipe A, kemudian memasukkan *subdomain* yang diinginkan, seperti simbol @ untuk mengarahkan ke *root domain*. Setelah itu, alamat IP VPS yang akan digunakan perlu dimasukkan, dan nilai *Time To Live* (TTL) dapat disesuaikan sesuai kebutuhan atau dibiarkan dalam pengaturan *default*. Mengingat bahwa data yang diproses dalam sistem bersifat krusial, nilai TTL untuk *subdomain* ini ditetapkan sebesar 3600 detik atau setara dengan 5 menit guna memastikan pembaruan yang lebih cepat.

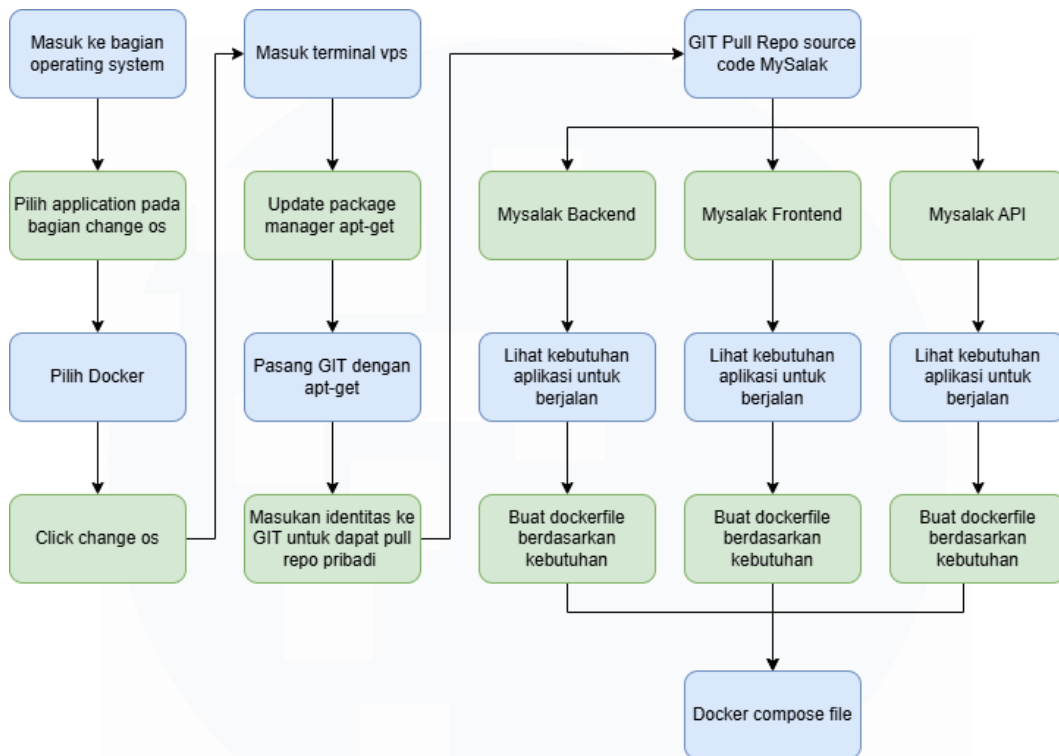
Karena aplikasi MySalak bergantung pada data dari API dan *hardware* pengukur, diperlukan konfigurasi tambahan untuk *subdomain* yang berfungsi sebagai penghubung dengan kedua komponen tersebut. *Subdomain* untuk API dan perangkat keras dibuat dengan konfigurasi serupa, dengan perbedaan pada bagian *name*, di mana *subdomain flask* digunakan untuk API, sementara *subdomain node* digunakan untuk *hardware* pengukur. Setelah seluruh

konfigurasi DNS selesai dilakukan, diperlukan waktu propagasi antara 1 hingga 24 jam untuk memastikan bahwa perubahan telah diterapkan sepenuhnya dan domain dapat berfungsi dengan baik.

3.2.5 Setup Docker, image, dan compose

Docker adalah layanan yang memastikan aplikasi yang dibuat pada komputer lokal dapat berjalan pada komputer lain dengan mengemas aplikasi ke dalam kontainer yang memiliki semua *library* dan *dependency* yang diperlukan aplikasi [5]. *Docker image* merupakan *blueprint* atau rancangan dari suatu kontainer yang berisi semua komponen yang diperlukan untuk menjalankan aplikasi, termasuk sistem operasi, dependensi, serta konfigurasi yang diperlukan. Dengan menggunakan *image* ini, pengguna dapat membuat banyak instansi kontainer yang identik, sehingga memudahkan proses *deployment* dan skala aplikasi.

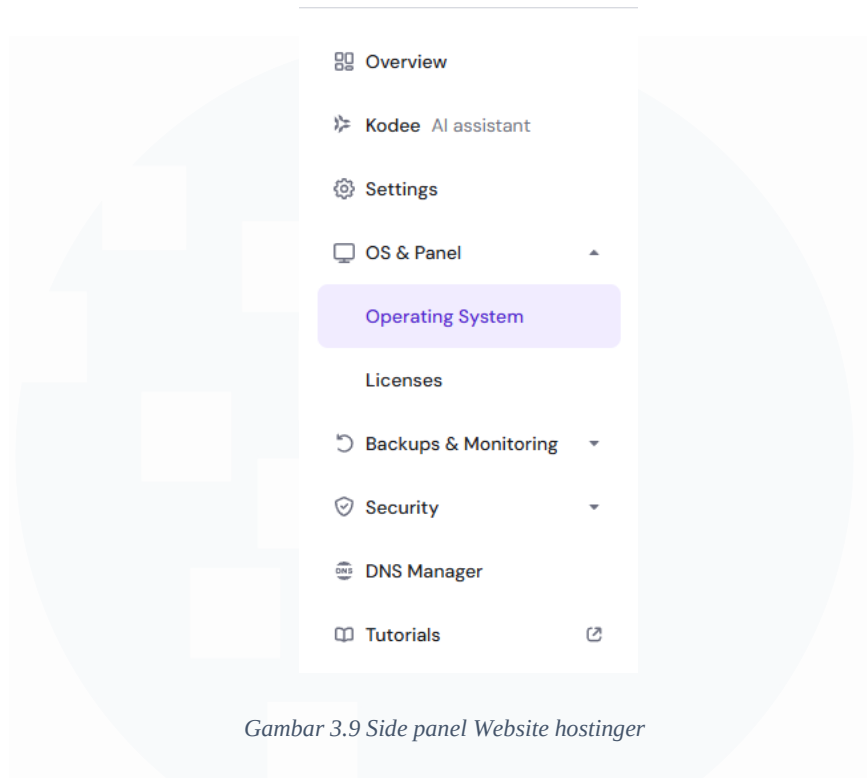
Selain itu, *Docker Compose* adalah alat yang digunakan untuk mendefinisikan dan mengelola aplikasi multi-kontainer menggunakan *file* konfigurasi berformat YAML, yaitu *Docker-compose.yml*. Dengan *Docker Compose*, berbagai layanan yang saling terhubung, seperti *backend*, *frontend*, dan *database*, dapat dikelola secara lebih efisien dalam satu perintah. Penggunaan *Docker Compose* memungkinkan pengembang untuk menjalankan seluruh *stack* aplikasi dengan sekali eksekusi menggunakan perintah *Docker compose up*, yang akan secara otomatis membangun *image*, mengatur jaringan antar-kontainer, serta memastikan semua layanan berjalan sesuai konfigurasi yang telah ditetapkan.



Gambar 3.8 Flowchart pemasangan Docker dan penggunaannya

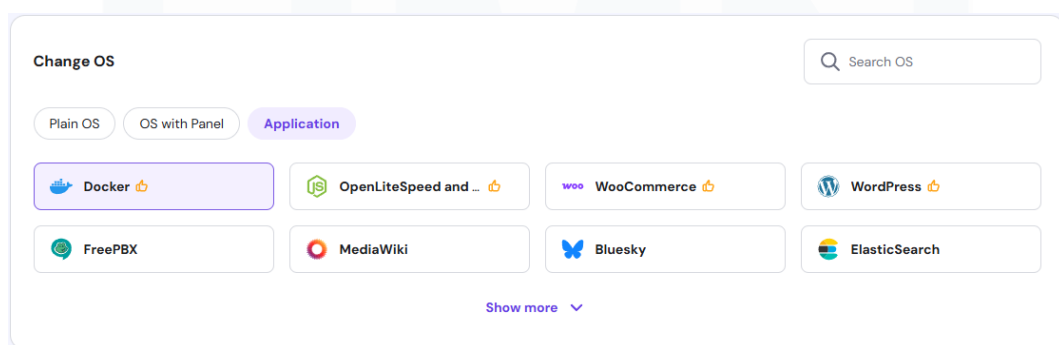
Seperti yang ditunjukkan pada Gambar 3.8, alur pemasangan Docker pada VPS dimulai dengan pengaturan sistem operasi dan aktivasi Docker. Setelah itu, pengguna mengakses terminal VPS untuk memperbarui package manager, menginstal GIT, dan mengonfigurasi identitas GIT guna menarik kode sumber MySalak.

Deployment terbagi menjadi tiga bagian utama: backend, frontend, dan API. Setiap komponen dianalisis untuk menentukan kebutuhan lingkungan eksekusinya, lalu dibuat Dockerfile yang sesuai. Terakhir, Docker Compose file disusun untuk menjalankan seluruh layanan, memastikan aplikasi berjalan optimal dan mudah dikelola.



Gambar 3.9 Side panel Website hostinger

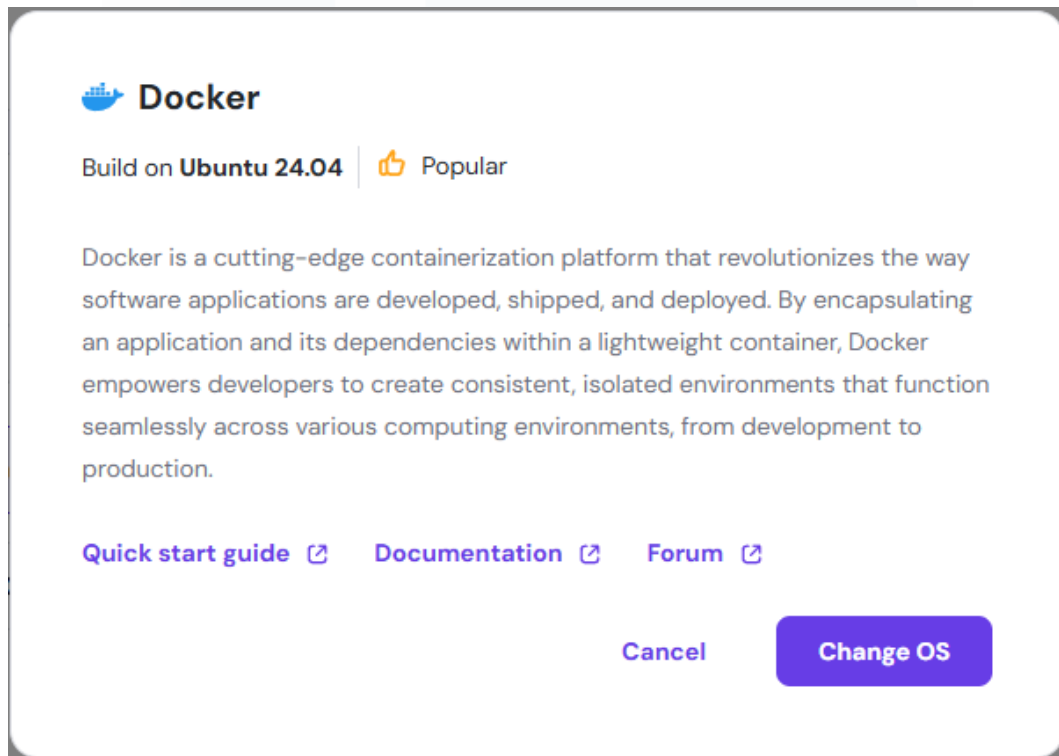
Untuk menggunakan Docker, VPS harus terlebih dahulu dipasangkan dan dikonfigurasi. Proses instalasi Docker pada VPS dapat dilakukan melalui side panel yang tersedia di platform Hostinger. Pengguna dapat memilih bagian *operating system*, sebagaimana ditunjukkan pada Gambar 3.9, sebelum melanjutkan dengan pemasangan Docker.



Gambar 3.10 Bagian pemilihan aplikasi yang ingin dipasang pada VPS

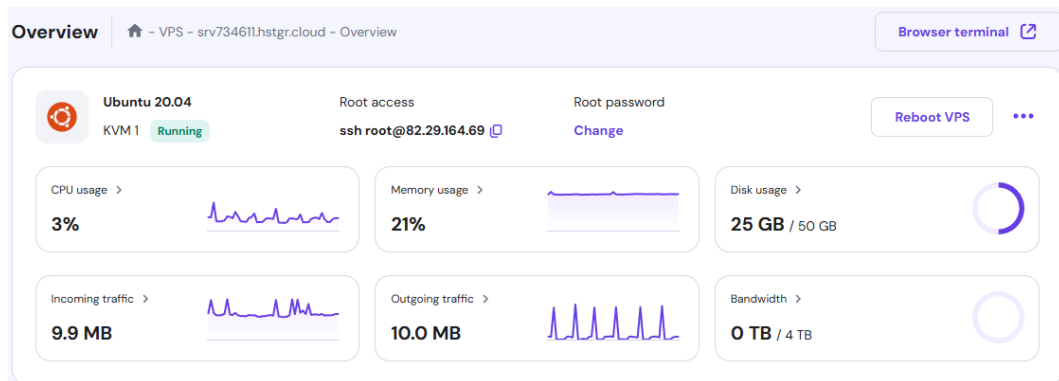
Selanjutnya, pada bagian *Change OS*, pengguna perlu memilih tab *Application* dan kemudian memilih Docker, sebagaimana ditunjukkan pada

Gambar 3.10. Setelah opsi Docker dipilih, akan muncul jendela konfirmasi seperti yang terlihat pada Gambar 3.11. Jendela ini menampilkan informasi mengenai versi aplikasi yang akan dipasang. Jika versi yang ditampilkan sudah sesuai, pengguna dapat mengklik tombol *Change OS* untuk memulai proses instalasi Docker pada VPS.



Gambar 3.11 window konfirmasi pemasangan aplikasi ke VPS

Untuk membuat dan menjalankan *Docker image* pada VPS, langkah pertama adalah masuk ke terminal VPS dengan mengklik tombol browser terminal pada sisi kanan atas pada *website hostinger VPS* yang dapat dilihat pada Gambar 3.12. Setelah itu, pengguna akan dipindahkan ke *website* lain di mana pengguna dapat secara langsung memberikan perintah melalui *command line* yang akan dijalankan oleh VPS



Gambar 3.12 Overview VPS

Untuk memasang layanan atau aplikasi pada VPS, diperlukan penggunaan *apt-get*, sebuah alat yang digunakan untuk mengambil paket *software* dari server distribusi dan memasangnya pada sistem. *Apt-get* berfungsi sebagai manajer paket yang memungkinkan pengambilan serta instalasi perangkat lunak dari sumber yang terpercaya.

Sebelum menggunakannya, penting untuk memastikan bahwa *apt-get* telah diperbarui ke versi terbaru guna menghindari potensi masalah kompatibilitas atau paket tidak tersedia. Oleh karena itu, perintah *sudo apt-get update* perlu dijalankan, sebagaimana ditunjukkan pada Gambar 3.13, untuk memperbarui daftar paket dan dependensi yang tersedia di sistem.

```

sudo apt-get update
Hit:1 http://in.archive.ubuntu.com/ubuntu focal InRelease
Get:2 http://in.archive.ubuntu.com/ubuntu focal-updates InRelease [128 kB]
Hit:3 https://download.docker.com/linux/ubuntu focal InRelease
Get:4 http://in.archive.ubuntu.com/ubuntu focal-backports InRelease [128 kB]
Get:5 http://in.archive.ubuntu.com/ubuntu focal-security InRelease [128 kB]
Hit:6 https://repository.monarx.com/repository/ubuntu-focal focal InRelease
Get:7 http://in.archive.ubuntu.com/ubuntu focal-updates/main amd64 Packages [3800 kB]
Get:8 http://in.archive.ubuntu.com/ubuntu focal-updates/main Translation-en [579 kB]
Get:9 http://in.archive.ubuntu.com/ubuntu focal-updates/universe amd64 Packages [1257 kB]
Fetched 6019 kB in 4s (1406 kB/s)
Reading package lists... Done

```

Gambar 3.13 update apt-get VPS

Untuk menyalin kode aplikasi MySalak ke VPS, diperlukan alat *Git*. *Git* merupakan sistem kontrol versi yang berfungsi sebagai alat kolaborasi dalam pengembangan sebuah proyek. Dengan menggunakan *Git*, pengembang dapat menyimpan versi terbaru dari proyek, menyalin seluruh komponen proyek ke

komputer lokal, serta mencatat setiap perubahan yang dilakukan, termasuk informasi mengenai siapa yang melakukan perubahan tersebut.

Untuk memasang layanan *Git* pada VPS, pengguna perlu menjalankan perintah `sudo apt-get install git`, sebagaimana ditunjukkan pada Gambar 3.14. Perintah ini akan mengunduh dan memasang *Git* dari repositori resmi, sehingga sistem dapat menggunakan fungsionalitasnya untuk mengelola kode sumber aplikasi secara lebih efisien.

```
root@srv734611:~# sudo apt-get install git
Reading package lists... Done
Building dependency tree
Reading state information... Done
git is already the newest version (1:2.25.1-1ubuntu3.14).
0 upgraded, 0 newly installed, 0 to remove and 9 not upgraded.
```

Gambar 3.14 pasang layanan git

Setelah *Git* berhasil dipasang pada VPS, langkah selanjutnya adalah mengonfigurasinya agar dapat mengakses repositori proyek. Untuk itu, *Git* memerlukan identitas pengguna yang harus ditetapkan terlebih dahulu. Konfigurasi ini dilakukan dengan menentukan nama pengguna dan alamat email menggunakan perintah `git config --global user.name "Modbilas"` dan `git config --global user.email "238.juanjen@gmail.com"`. Perintah tersebut memastikan bahwa setiap perubahan yang dilakukan dalam repositori akan dicatat dengan identitas yang diberikan. Setelah konfigurasi selesai, repositori yang berisi kode untuk *API*, *backend*, dan *frontend* aplikasi *MySalak* dapat disalin ke VPS menggunakan perintah `git clone`, sebagaimana ditunjukkan pada Gambar 3.15. Perintah ini akan menyalin seluruh isi repositori dari *remote server* ke VPS, sehingga kode dapat diakses dan dijalankan dalam lingkungan *server-side*.

```
# git clone https://github.com/MichaelHaes/mysalak_flask_api
```

Gambar 3.15 perintah salin repository

Setelah repositori berhasil diunduh, langkah selanjutnya adalah membuat *file Dockerfile* pada setiap folder utama proyek. *File Dockerfile* berisi instruksi yang diperlukan untuk membangun *Docker image*, sehingga aplikasi dapat dijalankan dalam lingkungan yang terisolasi. Pembuatan *Dockerfile* dilakukan dengan menjalankan perintah nano *Dockerfile*, yang memungkinkan pengguna menuliskan konfigurasi yang diperlukan, sebagaimana ditunjukkan pada Gambar 3.16. Setelah *Dockerfile* selesai dibuat, proses pembangunan *image* dilakukan dengan menjalankan perintah *Docker build -t modbilas/MySalak-flask-api:1.0*. Dalam perintah ini, *flag -t* digunakan untuk memberikan *tag* pada *image* guna menandai versi pembuatannya, sedangkan *modbilas/MySalak-flask-api* merupakan nama dari *Docker image* yang dihasilkan. Proses pembuatan *Docker image* ini diterapkan pada seluruh komponen aplikasi *MySalak*, termasuk *API*, *backend*, dan *frontend*, sehingga setiap bagian dapat dijalankan secara independen dalam lingkungan *containerized* yang lebih fleksibel dan terstandarisasi.

```

GNU nano 4.8                                Dockerfile
FROM python:3.12
RUN apt-get update && \
    apt-get install -y libhdf5-dev && \
    apt-get install -y libgl1-mesa-glx && \
    apt-get install git -y && \
    apt-get clean
WORKDIR /mysalak_flask_api
COPY requirements.txt ./
RUN pip install --no-cache-dir -r requirements.txt
RUN git clone https://github.com/gilbertzaini/yolov5-mysalak.git
COPY . .
EXPOSE 8888
CMD ["waitress-serve", "--host", "0.0.0.0", "--port", "8888", "server:app"]

[ Read 13 lines ]
^G Get Help  ^O Write Out ^W Where Is  ^R Cut Text  ^J Justify   ^C Cur Pos
^X Exit      ^R Read File ^\ Replace   ^U Paste Text ^T To Spell  ^_ Go To Line

```

Gambar 3.16 isi file *Dockerfile*

Selanjutnya, *file Docker-compose.yml* dibuat dengan menjalankan perintah *nano Docker-compose.yml* pada *root* folder proyek. *File Docker-compose* ini digunakan untuk mengelola layanan yang terdiri dari dua atau lebih *Docker container*, sehingga memungkinkan aplikasi berjalan dalam lingkungan yang terintegrasi.

Dalam *file Docker-compose.yml*, sejumlah parameter perlu ditentukan, antara lain nama layanan (*service*), *image* yang digunakan untuk masing-masing layanan, variabel lingkungan yang diperlukan, lokasi layanan dalam jaringan lokal (*network*), serta nama *database* jika dibutuhkan. Dengan konfigurasi ini, *Docker-compose* dapat secara otomatis mengatur dan menghubungkan berbagai *container* yang diperlukan untuk menjalankan aplikasi, sebagaimana ditunjukkan pada Gambar 3.17.

```
GNU nano 4.8                                docker-compose.yml
services:
  mysqldb:
    image: mysql:latest
    restart: unless-stopped
    environment:
      - MYSQL_DATABASE=mysalak
      - MYSQL_ROOT_PASSWORD=salak
    ports:
      - '3306:3306'
    volumes:
      - db:/var/lib/mysql
  mysalak_flask_backend:
    image: mysalak-flask-api:1.0
    ports:
      - '8888:8888'
  mysalak_express_backend:
    depends_on:
      - mysqldb
      - mysalak_flask_backend
    image: mysalak-express-api:1.0
[ Read 46 lines ]
^G Get Help  ^C Write Out ^W Where Is  ^R Cut Text  ^J Justify   ^C Cur Pos
^X Exit      ^R Read File ^\ Replace   ^U Paste Text ^T To Spell  ^_ Go To Line
```

Gambar 3.17 isi *Docker-compose.yml*

Setelah konfigurasi selesai, layanan dapat dijalankan dengan perintah *Docker compose up* seperti Gambar 3.18, yang akan menjalankan semua kontainer yang telah didefinisikan dalam *file Docker-compose.yml*. Dengan

demikian, aplikasi dapat berjalan secara terisolasi di dalam lingkungan kontainer pada VPS.

```
root@srv734611:~/mysalak_flask_api# docker compose up
[+] Running 4/4
✓ Container root-mysalak_flask_backend-1    Created           0.0s
✓ Container root-mysqldb-1                  Created           0.0s
✓ Container root-mysalak_express_backend-1   Created           0.0s
✓ Container root-mysalak-1                  Created           0.0s
Attaching to mysalak-1, mysalak_express_backend-1, mysalak_flask_backend-1, mysqldb-1
mysqldb-1 | 2025-03-01 10:43:06+00:00 [Note] [Entrypoint]: Entrypoint
mysalak_express_backend-1 |
mysalak_express_backend-1 | > mysalak-express-backend@1.0.0 dev
mysalak_express_backend-1 | > nodemon server
mysalak_express_backend-1 |
mysqldb-1 | 2025-03-01 10:43:09+00:00 [Note] [Entrypoint]: Switching
mysqldb-1 | 2025-03-01 10:43:09+00:00 [Note] [Entrypoint]: Entrypoint
```

Gambar 3.18 perintah Docker compose up berjalan

3.2.6 Setup Nginx reverse proxy

Agar aplikasi dapat diakses melalui *domain* MySalak, diperlukan konfigurasi Nginx sebagai *reverse proxy*. *Reverse proxy* merupakan server yang berfungsi sebagai perantara antara klien dan *backend server* [6]. Dalam proses ini, klien tidak berinteraksi langsung dengan *backend server*, melainkan mengirimkan permintaan ke *reverse proxy*, yang kemudian meneruskan permintaan tersebut ke server yang sesuai. Setelah *backend server* memproses permintaan, *reverse proxy* mengembalikan respons kepada klien.

Dalam implementasinya, Nginx dipilih sebagai *reverse proxy* karena memiliki berbagai fitur unggulan, seperti *load balancing*, *caching*, dan *rate limiting*. *Load balancing* memungkinkan distribusi lalu lintas ke beberapa server untuk meningkatkan keandalan dan kinerja aplikasi. *Caching* membantu mengurangi beban server dengan menyimpan sementara data yang sering diakses, sedangkan *rate limiting* berfungsi untuk mengontrol jumlah permintaan dari klien guna mencegah penyalahgunaan sumber daya. Dengan konfigurasi Nginx sebagai *reverse proxy*, aplikasi MySalak dapat diakses secara lebih efisien dan aman melalui domain yang telah ditentukan.

```

root@srv734611:~# sudo apt-get install nginx
Reading package lists... Done
Building dependency tree
Reading state information... Done
nginx is already the newest version (1.18.0-0ubuntu1.7).
30 upgraded, 0 newly installed, 0 to remove and 9 not upgraded.

```

Gambar 3.19 pemasangan Nginx

Untuk memasang Nginx pada VPS, instalasi dapat dilakukan menggunakan perintah *apt-get*. Dengan menjalankan perintah *sudo apt-get install Nginx* seperti Gambar 3.19, perangkat lunak Nginx akan dipasang pada sistem. Setelah proses instalasi selesai, status layanan Nginx dapat diperiksa menggunakan perintah *sudo systemctl status Nginx*. Jika layanan telah berjalan dengan baik, *output* perintah tersebut akan menampilkan status *active*, yang menandakan bahwa Nginx telah berhasil dipasang dan berjalan di sistem.

Langkah selanjutnya adalah masuk ke *direktori* Nginx dan berpindah ke folder *sites-available* dengan menjalankan perintah *cd /etc/nginx/sites-available/*. Folder *sites-available* berisi *file* konfigurasi yang mewakili *website* yang akan dihubungkan ke server Nginx. Untuk menambahkan domain serta *subdomain* yang akan digunakan oleh MySalak, perlu dibuat *file* konfigurasi baru di dalam folder ini. Penamaan *file* konfigurasi disesuaikan dengan fungsinya, di mana *file* untuk *frontend* dan *backend* dinamakan MySalak, sementara *file* untuk *server* API diberi nama *flask*.

```

GNU nano 6.2
server {
    server_name flask.mysalak.com;

    location / {
        proxy_pass http://127.0.0.1:8888/;
        proxy_set_header Host $host; # Forwarded host
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_redirect off;
    }
}

```

Gambar 3.20 kode penyambung domain ke server Nginx

Setiap *file* konfigurasi diisi dengan kode yang dapat dilihat pada Gambar 3.20, dengan bagian *server_name* dan *proxy_pass* yang disesuaikan berdasarkan *subdomain* dan *port* yang digunakan. Setelah semua domain dan *subdomain* memiliki file konfigurasi Nginx, *file* tersebut harus disalin ke dalam folder *sites-enabled* menggunakan perintah `sudo ln -s /etc/Nginx/sites-available/flask /etc/Nginx/sites-enabled/`. Penggunaan perintah `ln -s` memungkinkan pembuatan *symbolic link* yang menunjuk ke *file* asli, sehingga jika terjadi perubahan pada *file* konfigurasi utama, tidak perlu membuat ulang *file* tautan tersebut.

Setelah semua *file* konfigurasi domain dan *subdomain* telah dipindahkan ke folder *sites-enabled*, konfigurasi Nginx perlu diuji untuk memastikan tidak ada kesalahan dalam sintaknya. Pengujian dilakukan dengan menjalankan perintah `sudo Nginx -t`, yang akan memverifikasi setiap *file* konfigurasi yang baru dibuat. Jika tidak ditemukan kesalahan, Nginx kemudian perlu dimulai ulang dengan menjalankan perintah `sudo systemctl restart Nginx` guna menerapkan perubahan yang telah dilakukan.

Setelah proses *restart* selesai, *domain* akan dapat menampilkan aplikasi MySalak setelah menjalankan perintah `Docker compose up`. Namun, pada tahap ini, aplikasi hanya dapat diakses melalui HTTP dengan alamat `http://MySalak.com`, karena belum dilengkapi dengan sertifikat keamanan (SSL).

3.2.7 Certbot

Agar aplikasi MySalak dapat diakses melalui HTTPS, domain dan *subdomain* yang digunakan memerlukan sertifikasi keamanan. Untuk memperoleh sertifikat tersebut, digunakan Certbot, yaitu alat otomatis yang berfungsi untuk mendapatkan dan memperbarui sertifikat keamanan domain secara gratis melalui *Let's Encrypt* [7].

Proses pemasangan Certbot dapat dilakukan dengan menjalankan perintah `sudo apt-get install Certbot`. Setelah Certbot terpasang, langkah selanjutnya adalah mengaktifkan *firewall* guna memastikan keamanan server dengan menjalankan perintah `ufw enable`. Selanjutnya, aturan pada *firewall* perlu dikonfigurasi agar mengizinkan lalu lintas melalui *port* yang diperlukan, yaitu *port* 80 (HTTP), *port* 443 (HTTPS), serta *port* lain yang digunakan oleh aplikasi MySalak. Konfigurasi ini dapat dilakukan dengan perintah `ufw allow 80/tcp` dan `ufw allow 443/tcp`. Untuk memastikan aturan *firewall* telah diterapkan dengan benar, statusnya dapat diperiksa menggunakan perintah `ufw status`. Setelah konfigurasi ini selesai, Certbot dapat digunakan untuk menghasilkan sertifikat SSL yang memungkinkan aplikasi MySalak diakses dengan protokol HTTPS, meningkatkan keamanan komunikasi antara klien dan server.

```
root@srv734611:~# ufw status
Status: active

To Action From
--
3000/tcp ALLOW Anywhere
8888/tcp ALLOW Anywhere
8000/tcp ALLOW Anywhere
80/tcp ALLOW Anywhere
443/tcp ALLOW Anywhere
443 ALLOW Anywhere
3000/tcp (v6) ALLOW Anywhere (v6)
8888/tcp (v6) ALLOW Anywhere (v6)
8000/tcp (v6) ALLOW Anywhere (v6)
80/tcp (v6) ALLOW Anywhere (v6)
443/tcp (v6) ALLOW Anywhere (v6)
443 (v6) ALLOW Anywhere (v6)
```

Gambar 3.21 status firewall setelah ditambahkan port yang digunakan

Setelah semua *port* yang diperlukan telah dimasukkan ke dalam aturan *firewall*, tahap selanjutnya adalah pembuatan sertifikat keamanan untuk domain dan setiap *subdomain* secara individual. Proses ini dilakukan menggunakan perintah `sudo Certbot --Nginx -d MySalak.com -d www.MySalak.com`, yang secara otomatis mengonfigurasi Nginx untuk menggunakan sertifikat SSL dari

Let's Encrypt. Jika proses ini berhasil tanpa kendala, pengguna akan diberikan opsi untuk menentukan apakah pengunjung yang mengakses situs melalui HTTP akan secara otomatis diarahkan ke HTTPS atau tidak. Dalam kasus aplikasi MySalak, opsi yang dipilih adalah opsi 2, yang mengaktifkan *redirect* dari HTTP ke HTTPS guna memastikan keamanan dan enkripsi data selama komunikasi antara klien dan server. Setelah proses ini selesai, sistem akan menampilkan konfirmasi keberhasilan seperti yang ditunjukkan pada Gambar 3.22.

```
-----
Your existing certificate has been successfully renewed, and the new certificate
has been installed.

The new certificate covers the following domains: https://juantestsite.web.id
and https://www.juantestsite.web.id

You should test your configuration at:
https://www.ssllabs.com/ssltest/analyze.html?d=juantestsite.web.id
https://www.ssllabs.com/ssltest/analyze.html?d=www.juantestsite.web.id
-----

IMPORTANT NOTES:
- Congratulations! Your certificate and chain have been saved at:
  /etc/letsencrypt/live/juantestsite.web.id/fullchain.pem
  Your key file has been saved at:
  /etc/letsencrypt/live/juantestsite.web.id/privkey.pem
  Your cert will expire on 2025-05-30. To obtain a new or tweaked
  version of this certificate in the future, simply run certbot again
  with the "certonly" option. To non-interactively renew *all* of
  your certificates, run "certbot renew"
- If you like Certbot, please consider supporting our work by:

  Donating to ISRG / Let's Encrypt:  https://letsencrypt.org/donate
  Donating to EFF:                  https://eff.org/donate-le
```

Gambar 3.22 output hasil Certbot saat berhasil membuat sertifikat keamanan

Setelah proses pembuatan sertifikat selesai, Certbot secara otomatis menambahkan beberapa baris kode pada file konfigurasi yang terletak di direktori `/etc/Nginx/sites-available/`. Penambahan konfigurasi ini bertujuan untuk memastikan bahwa Nginx menggunakan sertifikat SSL yang telah diperoleh, sehingga domain dan *subdomain* dapat diakses melalui protokol HTTPS. Modifikasi yang dilakukan mencakup pengaturan jalur sertifikat dan kunci privat, serta penyesuaian konfigurasi keamanan untuk memastikan

komunikasi yang terenkripsi antara server dan klien. Perubahan tersebut dapat dilihat pada bagian bawah Gambar 3.23.

```

GNU nano 4.8          juantestsite.web.id
server {
    server_name juantestsite.web.id www.juantestsite.web.id;

    location / {
        proxy_pass http://127.0.0.1:3000/;
        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_set_header X-Forwarded-Proto $scheme;
        proxy_redirect off;
    }

    listen 443 ssl; # managed by Certbot
    ssl_certificate /etc/letsencrypt/live/juantestsite.web.id/fullchain.pem; # >
    ssl_certificate_key /etc/letsencrypt/live/juantestsite.web.id/privkey.pem; >
    include /etc/letsencrypt/options-ssl-nginx.conf; # managed by Certbot
    ssl_dhparam /etc/letsencrypt/ssl-dhparams.pem; # managed by Certbot
}

[ Read 40 lines ]
^G Get Help  ^C Write Out ^W Where Is  ^K Cut Text  ^J Justify   ^C Cur Pos
^X Exit      ^R Read File ^\ Replace   ^U Paste Text ^T To Spell  ^_ Go To Line

```

Gambar 3.23 kode sertifikat dari Certbot

3.3 Kendala yang Ditemukan

Selama proses *deployment* aplikasi MySalak muncul kendala-kendala yang perlu di selesaikan sebelum aplikasi dapat bekerja, yaitu:

1. Saat aplikasi dipindahkan ke VPS, aplikasi tidak bekerja dengan benar.
2. Peneliti tidak memiliki pengalaman dalam menggunakan Docker.
3. Aplikasi MySalak tidak dapat di akses saat masuk lewat domain.

3.4 Solusi atas Kendala yang Ditemukan

Berdasarkan kendala yang ditemukan selama *deployment* aplikasi, berikut solusi yang didapatkan:

1. Dengan menggunakan Docker, yang memastikan lingkungan aplikasi sama persis dengan lingkungan *development*.

2. Melakukan studi literatur tentang Docker dan melakukan latihan pemakaian Docker.
3. Agar dapat diakses saat pengguna mengetik domain MySalak.com, domain memerlukan sertifikat keamanan (SSL) yang didapatkan menggunakan Certbot.