

**MAKALAH**  
**DEPLOYMENT DAN DISTRIBUSI APLIKASI RESEPKU**  
**BERBASIS DOCKER DAN DOCKER HUB**



Disusun oleh:  
Muhammad Rafli Aldian Jamil  
(32602300044)

UNIVERSITAS ISLAM SULTAN AGUNG SEMARANG  
FAKULTAS TEKNOLOGI INDUSTRI  
PRODI TEKNIK INFORMATIKA  
2025

## DAFTAR ISI

<b>BAB I PENDAHULUAN .....</b>	<b>4</b>
2.1    Latar Belakang .....	5
2.2    Rumusan Masalah .....	5
2.3    Tujuan.....	5
<b>BAB II LANDASAN TEORI .....</b>	<b>5</b>
2.1    Docker .....	5
2.2    Docker Compose .....	6
2.3    Docker Hub .....	6
2.4    MariaDB.....	6
2.5    Flask .....	6
2.6    PhpMyAdmin.....	6
<b>BAB III METODOLOGI .....</b>	<b>6</b>
3.1    Analisis Kebutuhan .....	7
3.2    Perancangan Sistem .....	7
3.3    Implementasi .....	7
3.4    Pengujian.....	7
<b>BAB IV IMPLEMENTASI.....</b>	<b>8</b>
4.1    Struktur Folder Proyek.....	8
4.2    Pembuatan Dockerfile .....	8
4.3    Pembuatan docker-compose.yml .....	9
4.4    Build dan Testing Lokal .....	10
4.5    Build dan Push ke Docker Hub .....	11
4.6    Update docker-compose.yml.....	13
4.7    Penjelasan Fitur Aplikasi .....	14

4.7.1	Tampilan Home .....	14
4.7.2	Tambah Resep .....	15
4.7.3	Cari Resep .....	15
<b>BAB V HASIL DAN PEMBAHASAN .....</b>		<b>19</b>
5.1	Hasil .....	19
5.1.1	Build dan Testing Lokal .....	19
5.1.2	Tampilan Home Aplikasi.....	19
5.1.3	Pengelolaan Database dengan phpMyAdmin .....	19
5.1.4	Build dan Push Image ke Docker Hub.....	19
5.1.5	Update dan Deploy dari Docker Hub.....	20
5.1.6	Pengujian Fitur Aplikasi.....	20
5.2	Pembahasan.....	20
5.2.1	Analisis Proses Deployment .....	20
5.2.2	Analisis Fitur Aplikasi .....	21
5.2.3	Kendala dan Solusi.....	21
5.2.4	Evaluasi.....	21
<b>BAB VI KESIMPULAN .....</b>		<b>22</b>
<b>DAFTAR PUSTAKA.....</b>		<b>24</b>

## DAFTAR GAMBAR

Gambar 4. 1 Struktur folder proyek .....	8
Gambar 4. 2 Isi file Dockerfile .....	8
Gambar 4. 3 Isi file docker-compose.yml .....	9
Gambar 4. 4 Hasil build di terminal .....	10
Gambar 4. 5 Tampilan home aplikasi .....	10
Gambar 4. 6 Tampilan phpMyAdmin .....	11
Gambar 4. 7 Proses Build Image di Terminal .....	11
Gambar 4. 8 Proses push image ke Docker Hub .....	12
Gambar 4. 9 Repository Aplikasi di Docker Hub .....	12
Gambar 4. 10 Tampilan Saat Menjalankan Aplikasi .....	14
Gambar 4. 11 Tampilan Home Aplikasi .....	14
Gambar 4. 12 Tampilan Tambah Resep .....	15
Gambar 4. 13 Tampilan Cari Resep .....	15
Gambar 4. 14 Tampilan Hasil Pencarian Resep .....	16
Gambar 4. 15 Tampilan Detail Resep .....	16
Gambar 4. 16 Form Edit Resep .....	17
Gambar 4. 17 Tampilan Hapus Resep .....	17
Gambar 4. 18 Tampilan Random Resep .....	18
Gambar 4. 19 Tampilan Riwayat Pencarian .....	18

## **BAB I**

### **PENDAHULUAN**

#### **2.1 Latar Belakang**

Di era digital, kebutuhan akan aplikasi manajemen resep semakin meningkat. Banyak orang ingin menyimpan, mencari, dan mengelola resep masakan secara digital agar lebih mudah diakses dan tidak mudah hilang. Namun, deployment aplikasi web seringkali menghadapi tantangan seperti perbedaan environment, dependency, dan konfigurasi server. Docker hadir sebagai solusi untuk mengemas aplikasi beserta seluruh dependensinya ke dalam container, sehingga deployment menjadi lebih mudah, konsisten, dan dapat direplikasi di berbagai platform. Docker Hub memudahkan distribusi image aplikasi ke berbagai server atau anggota tim.

#### **2.2 Rumusan Masalah**

Bagaimana membangun, mendistribusikan, dan mendemokan aplikasi manajemen resep berbasis web menggunakan Docker dan Docker Hub agar deployment menjadi lebih mudah dan konsisten?

#### **2.3 Tujuan**

1. Membangun aplikasi manajemen resep berbasis web.
2. Melakukan deployment aplikasi menggunakan Docker.
3. Mendistribusikan aplikasi ke Docker Hub.
4. Mendemokan aplikasi secara lokal.

## **BAB II**

### **LANDASAN TEORI**

#### **2.1 Docker**

Docker adalah platform open-source yang memungkinkan developer mengemas aplikasi beserta seluruh dependensinya ke dalam sebuah container. Container ini dapat dijalankan di berbagai environment tanpa perlu konfigurasi ulang.

## **2.2 Docker Compose**

Docker Compose adalah tool untuk mendefinisikan dan menjalankan multi-container Docker applications menggunakan file YAML. Dengan Compose, developer dapat mengatur beberapa service sekaligus, seperti aplikasi, database, dan tool pendukung lainnya.

## **2.3 Docker Hub**

Docker Hub adalah layanan registry image Docker yang memungkinkan developer menyimpan dan mendistribusikan image secara online. Dengan Docker Hub, image aplikasi dapat di-pull dan dijalankan di mana saja.

## **2.4 MariaDB**

MariaDB adalah sistem manajemen basis data relasional yang merupakan fork dari MySQL. MariaDB banyak digunakan karena performa dan kompatibilitasnya yang tinggi.

## **2.5 Flask**

Flask adalah framework web berbasis Python yang ringan dan mudah digunakan untuk membangun aplikasi web. Flask mendukung pengembangan aplikasi dengan arsitektur modular dan fleksibel.

## **2.6 PhpMyAdmin**

PhpMyAdmin adalah aplikasi berbasis web untuk mengelola database MySQL/MariaDB. Dengan phpMyAdmin, pengguna dapat membuat, mengedit, dan menghapus database serta tabel secara visual.

# **BAB III**

## **METODOLOGI**

Penelitian ini menggunakan metode kualitatif dengan pendekatan rekayasa perangkat lunak. Metode kualitatif dipilih karena penelitian ini berfokus pada proses pengembangan, deployment, dan distribusi aplikasi secara deskriptif, tanpa melibatkan pengumpulan data numerik atau statistik.

Tahapan metodologi yang dilakukan meliputi:

### **3.1 Analisis Kebutuhan**

Pada tahap ini dilakukan identifikasi kebutuhan fungsional dan non-fungsional aplikasi, seperti:

1. Pengguna dapat menambah, mencari, mengedit, menghapus, dan melihat resep.
2. Aplikasi dapat dijalankan secara containerized.
3. Database terintegrasi dan mudah dikelola melalui phpMyAdmin.
4. Deployment dan distribusi aplikasi mudah dilakukan.

### **3.2 Perancangan Sistem**

Tahap ini meliputi:

1. Perancangan arsitektur aplikasi (menggunakan Docker Compose untuk multi-container).
2. Perancangan struktur folder proyek.
3. Perancangan Dockerfile untuk membungkus aplikasi Flask.
4. Perancangan docker-compose.yml untuk mengatur service Flask, MariaDB, dan phpMyAdmin.

### **3.3 Implementasi**

ada tahap ini dilakukan:

1. Pembuatan kode program aplikasi dengan Flask.
2. Pembuatan Dockerfile dan docker-compose.yml.
3. Build image aplikasi.
4. Testing lokal menggunakan Docker Compose.
5. Push image ke Docker Hub.

### **3.4 Pengujian**

Aplikasi diuji secara lokal menggunakan Docker Compose untuk memastikan semua fitur berjalan dengan baik, serta diuji deployment-nya dengan menarik image dari Docker Hub. Pengujian dilakukan dengan menjalankan aplikasi, mengakses fitur-fitur utama, dan memastikan integrasi antar service berjalan lancar.

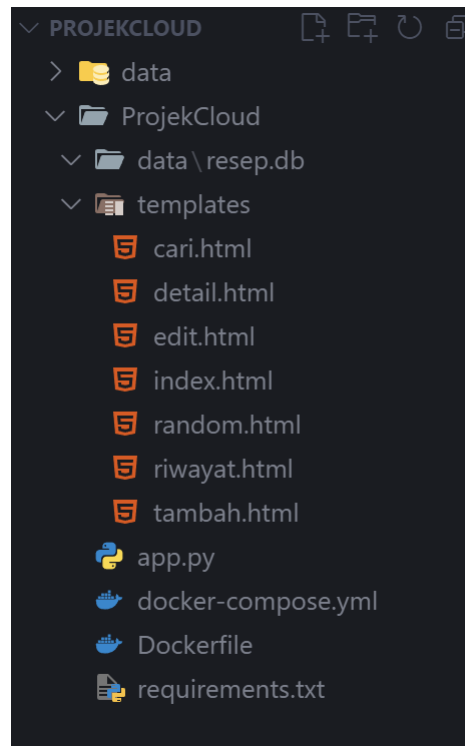
Setiap tahapan didokumentasikan secara naratif dan didukung dengan screenshot serta penjelasan langkah-langkah yang dilakukan.

## BAB IV

### IMPLEMENTASI

#### 4.1 Striktur Folder Proyek

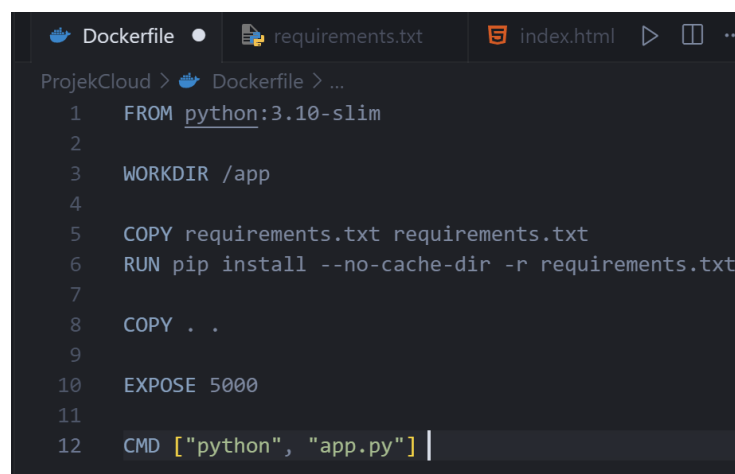
Struktur folder aplikasi Resepku adalah sebagai berikut:



Gambar 4. 1 Struktur folder proyek

#### 4.2 Pembuatan Dockerfile

Dockerfile digunakan untuk membungkus aplikasi Flask ke dalam sebuah image. Berikut adalah isi Dockerfile:

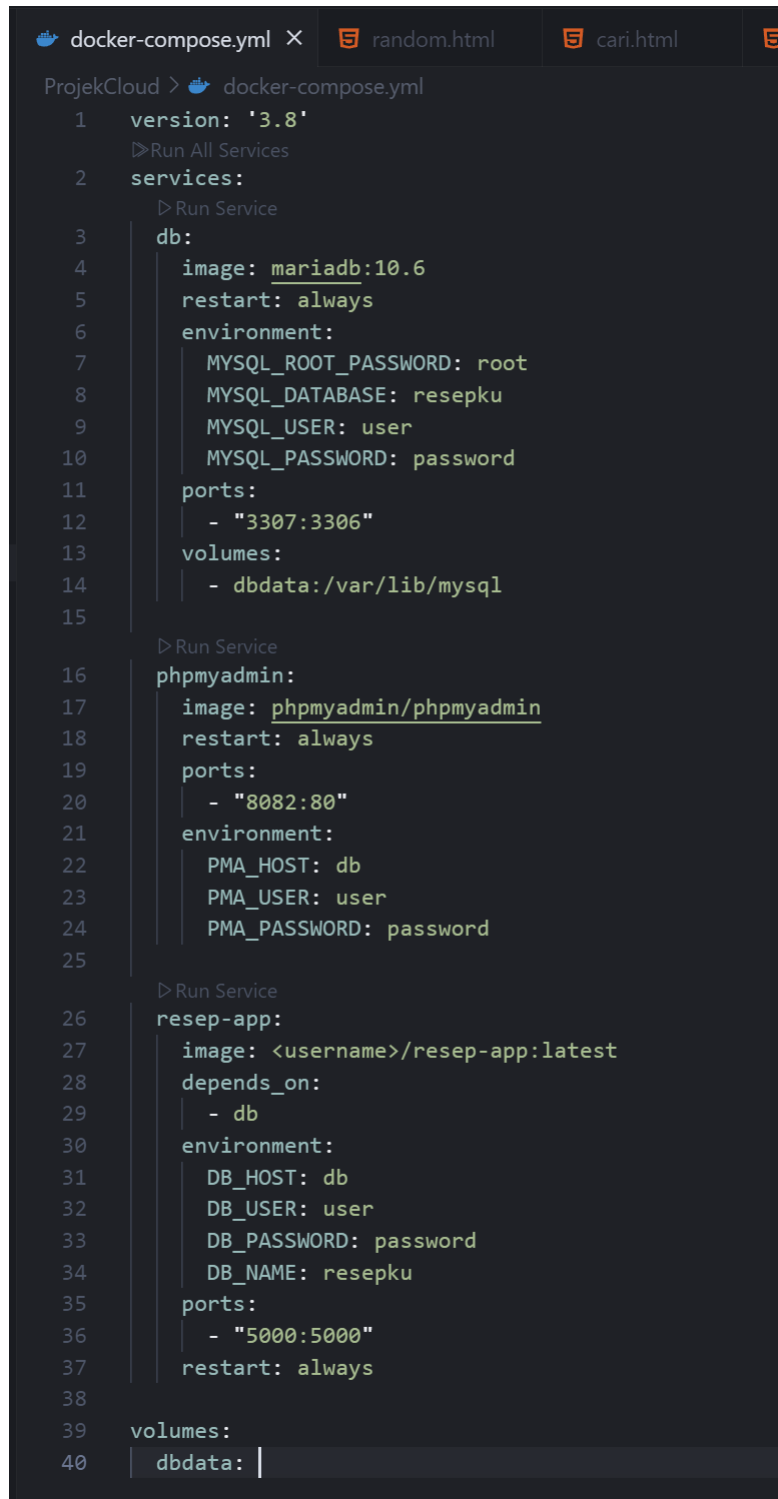


Gambar 4. 2 Isi file Dockerfile



### 4.3 Pembuatan docker-compose.yml

File docker-compose.yml digunakan untuk mengatur tiga service utama, yaitu aplikasi Flask, database MariaDB, dan phpMyAdmin.



```
1  version: '3.8'
2  services:
3    db:
4      image: mariadb:10.6
5      restart: always
6      environment:
7        MYSQL_ROOT_PASSWORD: root
8        MYSQL_DATABASE: resepku
9        MYSQL_USER: user
10       MYSQL_PASSWORD: password
11      ports:
12        - "3307:3306"
13      volumes:
14        - dbdata:/var/lib/mysql
15
16    phpmyadmin:
17      image: phpmyadmin/phpmyadmin
18      restart: always
19      ports:
20        - "8082:80"
21      environment:
22        PMA_HOST: db
23        PMA_USER: user
24        PMA_PASSWORD: password
25
26    resep-app:
27      image: <username>/resep-app:latest
28      depends_on:
29        - db
30      environment:
31        DB_HOST: db
32        DB_USER: user
33        DB_PASSWORD: password
34        DB_NAME: resepku
35      ports:
36        - "5000:5000"
37      restart: always
38
39  volumes:
40    dbdata:
```

Gambar 4. 3 Isi file docker-compose.yml

## 4.4 Build dan Testing Lokal

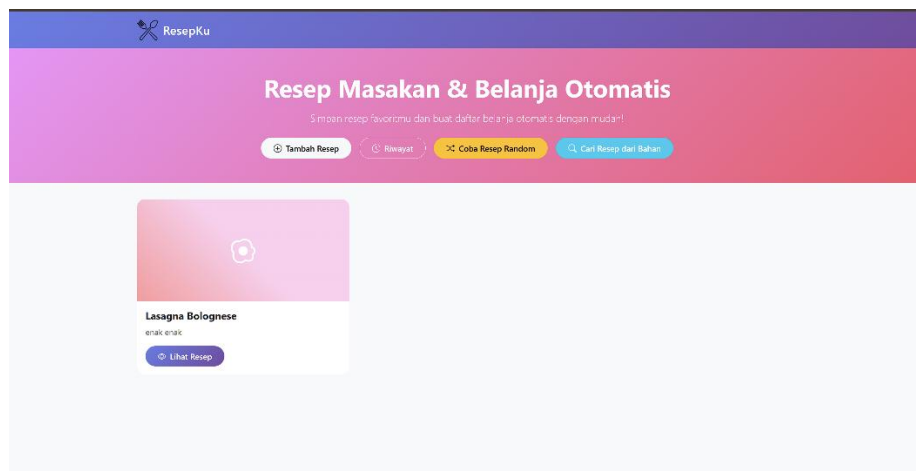
Build dan testing aplikasi dilakukan dengan perintah `docker-compose up --build`.

```
C:\Users\duima\Documents\ProjekCloud\ProjekCloud>docker-compose up --build
[*] Running 39/39
✓db Pulled                                369.7s
✓resep-app Pulled                         48.0s
✓phpmyadmin Pulled                       269.6s

[*] Running 5/5
✓Network projekcloud_default              Created      0.1s
✓Volume "projekcloud_dbdata"              Created      0.0s
✓Container projekcloud-db-1                Created      0.4s
✓Container projekcloud-phpmyadmin-1        Created      0.4s
✓Container projekcloud-resep-app-1         Created      0.0s
Attaching to db-1, phpmyadmin-1, resep-app-1
db-1 | 2025-07-14 12:59:55+08:00 [Note] [Entrypoint]: Entrypoint script for MariaDB Server 1:10.6.22+maria-ubu2004 started.
phpmyadmin-1 | AH00558: apache2: Could not reliably determine the server's fully qualified domain name, using 172.18.0.2. Set the 'ServerName' dire
phpmyadmin-1 | AH00558: apache2: Could not reliably determine the server's fully qualified domain name, using 172.18.0.2. Set the 'ServerName' dire
phpmyadmin-1 | ctive globally to suppress this message
phpmyadmin-1 | [Mon Jul 14 12:59:56.278224 2825] [mpm_prefork:notice] [pid 1:tid 1] AH00163: Apache/2.4.62 (Debian) PHP/8.2.27 configured -- resumi
ng normal operations
phpmyadmin-1 | [Mon Jul 14 12:59:56.278283 2825] [core:notice] [pid 1:tid 1] AH00094: Command line: 'apache2 -D FOREGROUND'
db-1 | 2025-07-14 12:59:56+08:00 [Note] [Entrypoint]: Switching to dedicated user 'mysql'
resep-app-1 | 2025-07-14 12:59:56+08:00 [Note] [Entrypoint]: Entrypoint script for MariaDB Server 1:10.6.22+maria-ubu2004 started.
resep-app-1 | File "/usr/local/lib/python3.10/site-packages/mysql/connector/connection_cext.py", line 354, in _open_connection
resep-app-1 | self._cmysql.connect(**cnx_kwargs)
resep-app-1 | _mysql_connector.MySQLInterfaceError: Can't connect to MySQL server on 'db:3306' (111)
resep-app-1 | The above exception was the direct cause of the following exception:
db-1 | 2025-07-14 12:59:56+08:00 [Note] [Entrypoint]: Initializing database files
resep-app-1 | Traceback (most recent call last):
resep-app-1 | File "/app/app.py", line 175, in <module>
resep-app-1 |     init_db()
resep-app-1 | File "/app/app.py", line 19, in init_db
resep-app-1 |     conn = get_db_connection()
```

Gambar 4. 4 Hasil build di terminal

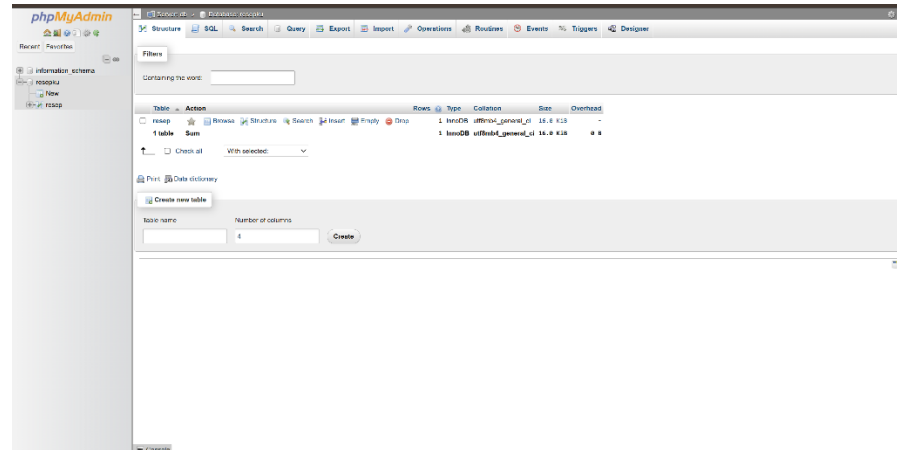
Setelah build selesai, aplikasi dapat diakses melalui browser pada alamat `http://localhost:5000` dan phpMyAdmin pada <http://localhost:8082>.



Gambar 4. 5 Tampilan home aplikasi

Gambar 4.5 menunjukkan tampilan halaman utama (home) aplikasi Resepku setelah berhasil dijalankan di browser pada alamat `http://localhost:5000`. Pada halaman ini, pengguna dapat melihat daftar resep yang sudah tersimpan di database. Terdapat juga navigasi ke fitur-fitur lain seperti tambah resep, cari resep, random resep, dan riwayat pencarian.

Tampilan home ini menjadi pusat akses utama bagi pengguna untuk mengelola dan mencari resep yang diinginkan.



Gambar 4. 6 Tampilan phpMyAdmin

Gambar 4.6 memperlihatkan tampilan phpMyAdmin yang diakses melalui <http://localhost:8082>. phpMyAdmin digunakan untuk memudahkan pengelolaan database MariaDB secara visual. Pada tampilan ini, pengguna dapat melihat daftar database, tabel-tabel yang ada, serta melakukan operasi seperti menambah, mengedit, atau menghapus data secara langsung. phpMyAdmin sangat membantu dalam proses administrasi database tanpa perlu menggunakan perintah SQL secara manual.

#### 4.5 Build dan Push ke Docker Hub

Setelah aplikasi berjalan baik secara lokal, langkah selanjutnya adalah build image untuk Docker Hub:

```
C:\Users\dwina\Documents\ProjekCloud\ProjekCloud>docker build -t aldirafli/resep-app:latest .
[+] Building 396.7s (11/11) FINISHED                                docker:desktop-linux
=> [internal] load build definition from Dockerfile                  0.0s
=> => transferring dockerfile: 262B                                  0.0s
=> [internal] load metadata for docker.io/library/python:3.10-slim  3.5s
=> [auth] library/python:pull token for registry-1.docker.io        0.0s
=> [internal] load .dockerignore                                     0.0s
=> => transferring context: 4B                                         0.0s
=> [1/5] FROM docker.io/library/python:3.10-slim@sha256:9dd6774a1276178f94b0cc1fb1f0edd98825d0ea7634847af9940b1b6273c13  0.0s
=> resolve docker.io/library/python:3.10-slim@sha256:9dd6774a1276178f94b0cc1fb1f0edd98825d0ea7634847af9940b1b6273c13  0.1s
=> => sha256:9dd6774a1276178f94b0cc1fb1f0edd98825d0ea7634847af9940b1b6273c13 9.13kB / 9.13kB  0.0s
=> => sha256:c5dbdb2b857830a0a0d0765ea519c1514a0bda11b4c40e8352072fab4379d5 1.75kB / 1.75kB  0.0s
=> => sha256:563d4fbdc42697a70677ee28bb6b67ad9074e3c9f753043523a38dc4f4f553 5.37kB / 5.37kB  0.0s
=> [internal] load build context                                     0.0s
=> => transferring context: 45.47kB                                    0.0s
=> [2/5] WORKDIR /app                                              0.0s
=> [3/5] COPY requirements.txt requirements.txt                    0.0s
=> [4/5] RUN pip install --no-cache-dir -r requirements.txt       392.5s
=> [5/5] COPY . .                                                  0.0s
=> exporting to image                                              0.4s
=> writing image sha256:32f3e804fb06b475f05e59dca6d89e306d2232daa3619f69de9c5f1f94624 0.0s
=> naming to docker.io/aldirafli/resep-app:latest                 0.0s
```

Gambar 4. 7 Proses Build Image di Terminal

Proses ini dilakukan di terminal pada direktori proyek. Pada tahap ini, Docker akan membaca instruksi dari file Dockerfile, menginstall dependencies, menyalin source code, dan menghasilkan sebuah image

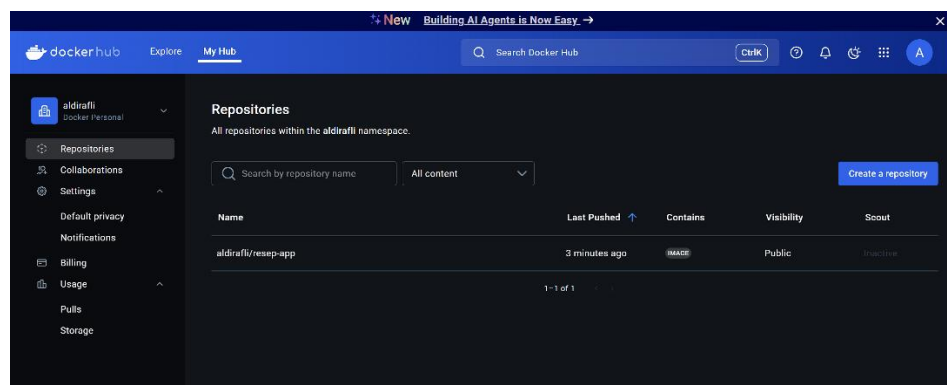
aplikasi yang siap untuk dijalankan atau di-push ke Docker Hub. Output di terminal menampilkan tahapan build dan status keberhasilan build image.

```
C:\Users\dwima\Documents\ProjekCloud\ProjekCloud>docker push aldirafli/resep-app:latest
The push refers to repository [docker.io/aldirafli/resep-app]
51818d53b94a: Pushed
6078db42e914: Pushed
ab6f97c74861: Pushed
1d601b70893b: Pushed
31abbb538de0: Layer already exists
a2ccf7527b3c: Layer already exists
abd92ba2021e: Layer already exists
1bb35e8b4de1: Layer already exists
latest: digest: sha256:bc0ecc52944954e6d5fb54a06289097031502f6e4ae1ac777b2bbd855c20c306 size: 1992
```

Gambar 4. 8 Proses push image ke Docker Hub

Gambar 4.8 memperlihatkan proses push image aplikasi ke Docker Hub menggunakan perintah:

Pada tahap ini, image yang sudah berhasil dibuild akan diunggah ke repository Docker Hub milik pengguna. Proses push ini memungkinkan image aplikasi dapat diakses dan digunakan di komputer atau server lain tanpa perlu build ulang. Output di terminal menunjukkan progress upload dan konfirmasi jika push berhasil.



Gambar 4. 9 Repository Aplikasi di Docker Hub

Gambar 4.9 menampilkan halaman repository aplikasi di Docker Hub melalui browser. Pada halaman ini, dapat dilihat bahwa image aplikasi dengan tag latest sudah berhasil diunggah dan tersedia secara publik atau privat sesuai pengaturan repository. Informasi seperti nama repository, tag, ukuran image, dan waktu upload juga dapat dilihat di halaman ini. Hal ini menandakan bahwa aplikasi sudah siap untuk didistribusikan dan di-deploy di lingkungan lain.

## 4.6 Update docker-compose.yml

Setelah image tersedia di Docker Hub, file docker-compose.yml diupdate agar service resep-app menggunakan image dari Docker Hub:

```
services:
  db:
    image: mariadb:10.6
    restart: always
    environment:
      MYSQL_ROOT_PASSWORD: root
      MYSQL_DATABASE: resepku
      MYSQL_USER: user
      MYSQL_PASSWORD: password
    ports:
      - "3307:3306"
    volumes:
      - dbdata:/var/lib/mysql

  phpmyadmin:
    image: phpmyadmin/phpmyadmin
    restart: always
    ports:
      - "8082:80"
    environment:
      PMA_HOST: db
      PMA_USER: user
      PMA_PASSWORD: password

  resep-app:
    image: aldirafli/resep-app:latest
    depends_on:
      - db
    environment:
      DB_HOST: db
      DB_USER: user
      DB_PASSWORD: password
      DB_NAME: resepku
    ports:
```

```

- "5000:5000"
restart: always

volumes:
  dbdata:

Lalu jalankan aplikasi dengan docker-compose up -d
C:\Users\dwima\Documents\ProjekCloud\ProjekCloud>docker-compose up -d
[+] Running 3/3
✓Container projekcloud-resep-app-1   Started
✓Container projekcloud-phpmyadmin-1 Started
✓Container projekcloud-db-1         Started

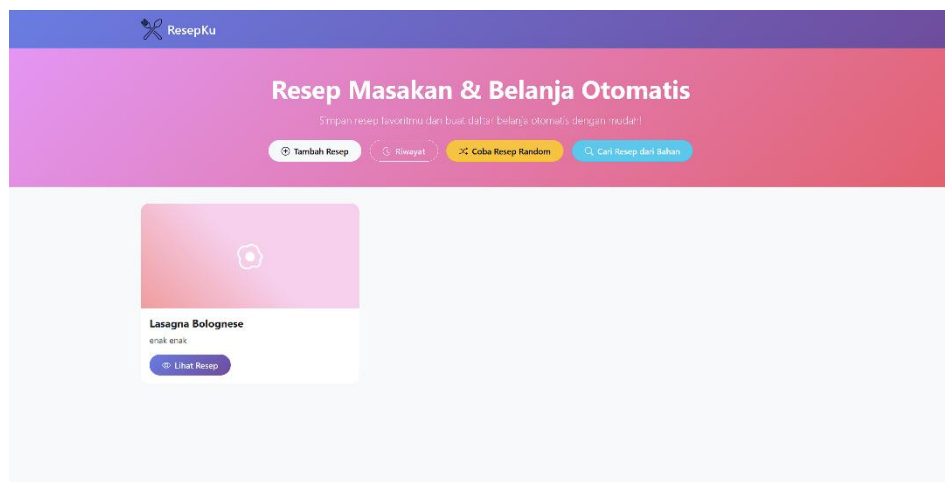
```

Gambar 4. 10 Tampilan Saat Menjalankan Aplikasi

## 4.7 Penjelasan Fitur Aplikasi

Aplikasi Resepku memiliki beberapa fitur utama, yaitu:

### 4.7.1 Tampilan Home



Gambar 4. 11 Tampilan Home Aplikasi

Gambar ini menunjukkan halaman utama aplikasi Resepku yang diakses melalui <http://localhost:5000>. Pada halaman ini, pengguna dapat melihat daftar resep yang sudah tersimpan di database. Terdapat juga menu navigasi ke fitur-fitur lain seperti tambah resep, cari resep, random resep, dan riwayat pencarian. Tampilan home ini menjadi pusat akses utama bagi pengguna untuk mengelola dan mencari resep.

## 4.7.2 Tambah Resep

The screenshot shows the 'Tambah Resep Baru' (Add New Recipe) form in the ResepKu application. The form is titled 'Tambah Resep Baru' with a subtitle 'Isi detail resep favoritmu di bawah ini'. It contains four main sections: 'Nama Resep' (Recipe Name), 'Deskripsi' (Description), 'Daftar Bahan' (List of Ingredients), and 'Langkah-langkah Memasak' (Cooking Steps). Each section has a text input field and a 'Simpan Resep' (Save Recipe) button. The 'Daftar Bahan' section includes a list of ingredients: '2 siung bawang putih', '1 sendok makan garam', and '500g daging ayam'. The 'Langkah-langkah Memasak' section includes a list of steps: '1. Panaskan minyak dalam wajan', '2. Tumis bawang putih hingga harum', and '3. Masukkan daging ayam'.

Gambar 4. 12 Tampilan Tambah Resep

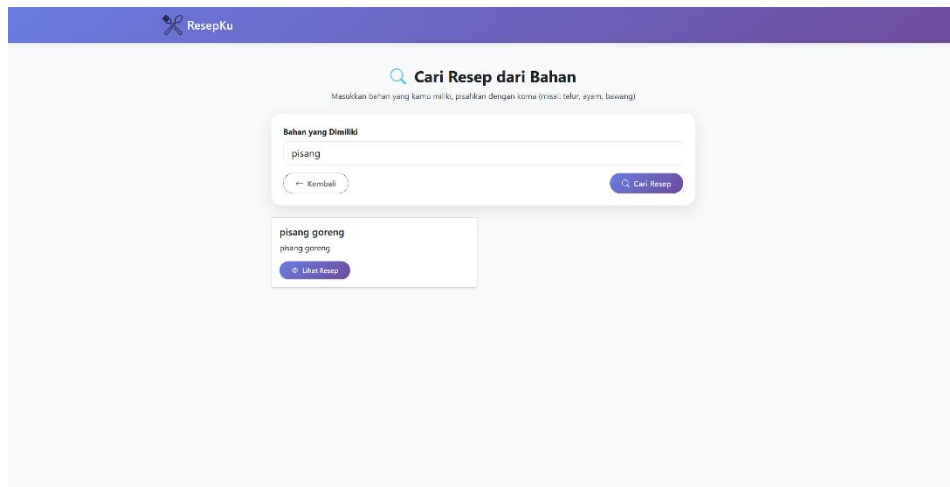
Gambar 4.12 memperlihatkan form tambah resep yang dapat diakses melalui menu “Tambah Resep”. Pengguna dapat mengisi nama resep, bahan-bahan, dan langkah-langkah pembuatan pada form ini.

## 4.7.3 Cari Resep

The screenshot shows the 'Cari Resep dari Bahan' (Search Recipe by Ingredient) page in the ResepKu application. The page has a title 'Cari Resep dari Bahan' and a subtitle 'Masukkan bahan yang kamu miliki, pilihkan dengan koma (misal: telur, ayam, bawang)'. There is a search input field with the text 'paku' and a 'Cari Resep' button. Below the search field, there is a message 'Tidak ada resep yang cocok' (No recipe is suitable) and a subtitle 'Coba kurangi jumlah bahan atau tambah resep baru' (Try reducing the number of ingredients or adding a new recipe).

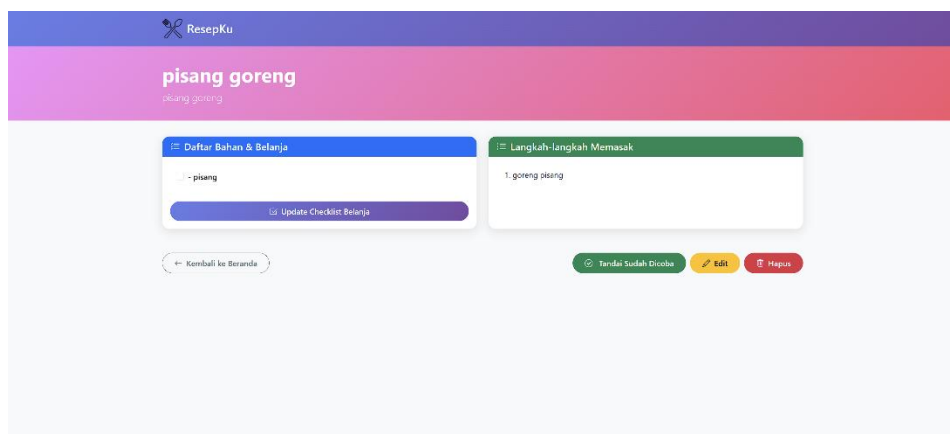
Gambar 4. 13 Tampilan Cari Resep

Gambar 4.13 menampilkan halaman pencarian resep, di mana pengguna dapat memasukkan kata kunci untuk mencari resep berdasarkan nama atau bahan tertentu.



Gambar 4. 14 Tampilan Hasil Pencarian Resep

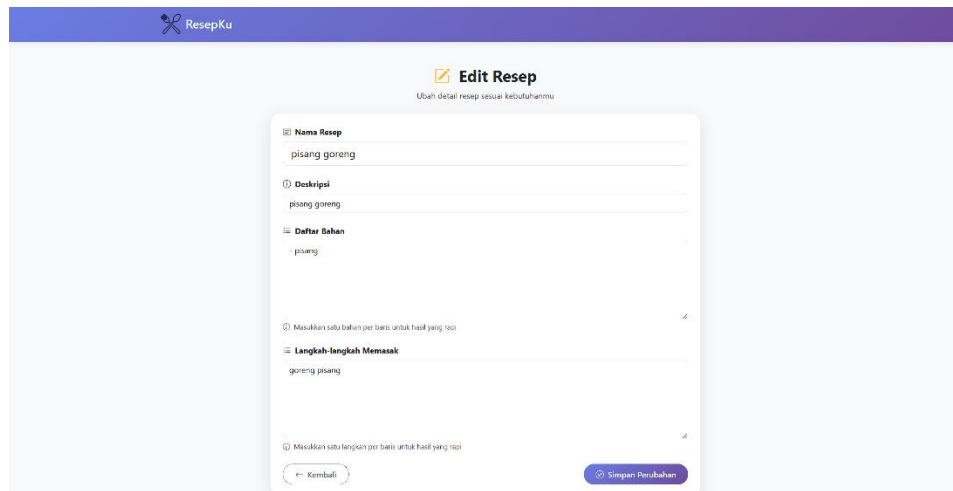
Setelah melakukan pencarian, aplikasi akan menampilkan daftar hasil resep yang sesuai dengan kata kunci yang dimasukkan. Pengguna dapat memilih salah satu resep untuk melihat detailnya.



Gambar 4. 15 Tampilan Detail Resep

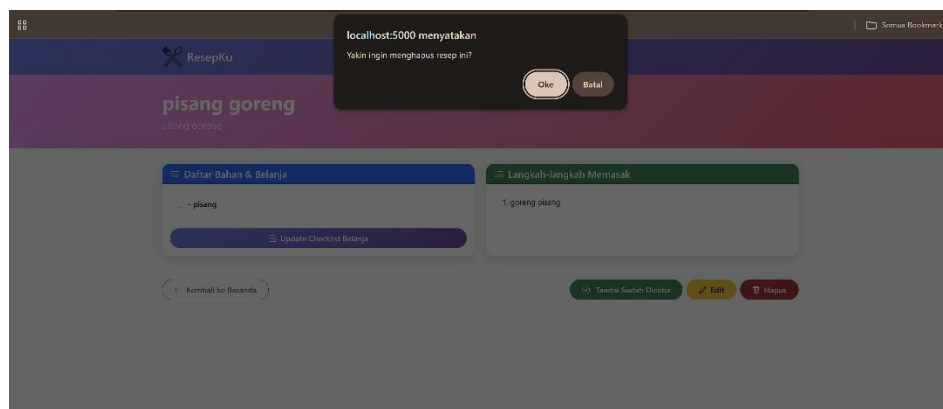
Gambar ini menunjukkan halaman detail resep yang menampilkan informasi lengkap mengenai resep yang dipilih, seperti nama resep, bahan-bahan, dan langkah-langkah pembuatan.





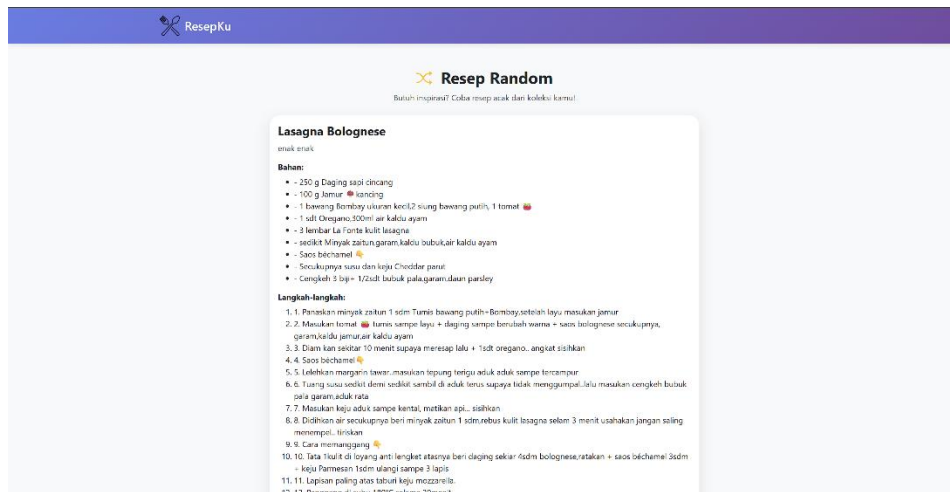
Gambar 4. 16 Form Edit Resep

Gambar ini memperlihatkan form edit resep yang dapat diakses dari halaman detail resep. Pengguna dapat memperbarui informasi resep yang sudah ada melalui form ini.



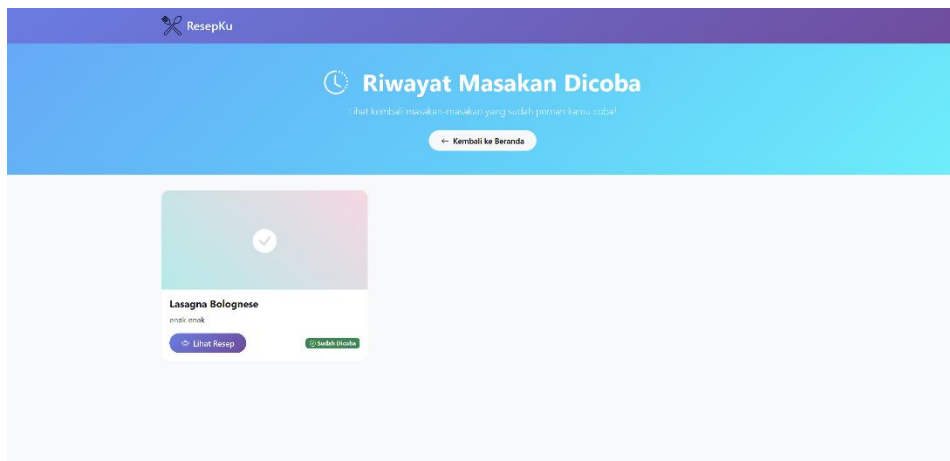
Gambar 4. 17 Tampilan Hapus Resep

Gambar ini menampilkan tampilan konfirmasi sebelum menghapus resep. Fitur ini memastikan bahwa pengguna benar-benar ingin menghapus resep dari database.



Gambar 4. 18 Tampilan Random Resep

Gambar ini memperlihatkan fitur random resep, di mana aplikasi akan menampilkan satu resep secara acak dari database. Fitur ini membantu pengguna yang bingung ingin memasak apa.



Gambar 4. 19 Tampilan Riwayat Pencarian

Gambar 4.19 menampilkan halaman riwayat pencarian, di mana pengguna dapat melihat daftar kata kunci atau resep yang pernah dicari sebelumnya. Fitur ini memudahkan pengguna untuk mengakses kembali resep yang pernah dicari.

## **BAB V**

### **HASIL DAN PEMBAHASAN**

#### **5.1 Hasil**

Pada bab ini dijelaskan hasil dari proses implementasi, deployment, serta pengujian aplikasi Resepku yang telah dilakukan secara bertahap. Setiap tahapan didokumentasikan dengan narasi dan screenshot untuk memperjelas hasil yang diperoleh.

##### **5.1.1 Build dan Testing Lokal**

Setelah seluruh file konfigurasi dan kode aplikasi selesai dibuat, proses build dan testing dilakukan secara lokal menggunakan Docker Compose. Perintah yang digunakan adalah:

```
docker-compose up -build
```

Hasil build ditunjukkan pada Gambar 4.4, di mana seluruh service berhasil dijalankan tanpa error. Aplikasi dapat diakses melalui browser pada alamat <http://localhost:5000> dan phpMyAdmin pada <http://localhost:8082>.

##### **5.1.2 Tampilan Home Aplikasi**

Setelah aplikasi berjalan, halaman utama aplikasi Resepku dapat diakses. Pada Gambar 4.5, terlihat daftar resep yang sudah tersimpan di database, serta menu navigasi ke fitur-fitur lain. Hal ini menunjukkan bahwa integrasi antara aplikasi Flask dan database MariaDB berjalan dengan baik.

##### **5.1.3 Pengelolaan Database dengan phpMyAdmin**

Pada Gambar 4.6, tampak tampilan phpMyAdmin yang digunakan untuk mengelola database MariaDB. Melalui phpMyAdmin, dapat dilihat struktur tabel, data resep, serta melakukan operasi CRUD secara visual. Hal ini memudahkan proses administrasi database tanpa perlu perintah SQL manual.

##### **5.1.4 Build dan Push Image ke Docker Hub**

Proses build image aplikasi dilakukan dengan perintah:

```
docker build -t aldirafli/resep-app:latest .
```

Hasil build image ditunjukkan pada Gambar 4.7. Setelah itu, image di-push ke Docker Hub menggunakan perintah:

```
docker push aldirafli/resep-app:latest
```

Proses push ini berhasil ditunjukkan pada Gambar 4.8. Verifikasi di Docker Hub dapat dilihat pada Gambar 4.9, di mana image aplikasi sudah tersedia di repository Docker Hub.

#### **5.1.5 Update dan Deploy dari Docker Hub**

Setelah image tersedia di Docker Hub, file docker-compose.yml diupdate agar service resep-app menggunakan image dari Docker Hub. Aplikasi kemudian dijalankan kembali dan dapat diakses seperti sebelumnya, menandakan proses distribusi dan deployment berjalan lancar.

#### **5.1.6 Pengujian Fitur Aplikasi**

Setiap fitur aplikasi diuji dan didokumentasikan sebagai berikut:

1. Tampilan Home: Menampilkan daftar resep dan menu navigasi (Gambar 4.11).
2. Tambah Resep: Form tambah resep dapat digunakan (Gambar 4.12).
3. Cari Resep: Fitur pencarian menampilkan hasil sesuai kata kunci (Gambar 4.13 dan 4.14).
4. Detail Resep: Informasi lengkap resep dapat diakses (Gambar 4.15).
5. Edit Resep: Form edit dapat digunakan untuk memperbarui data resep (Gambar 4.16).
6. Hapus Resep: Konfirmasi hapus muncul sebelum data dihapus (Gambar 4.17).
7. Random Resep: Fitur random menampilkan resep acak (Gambar 4.18).
8. Riwayat Pencarian: Daftar riwayat pencarian tampil dengan baik (Gambar 4.19).

### **5.2 Pembahasan**

#### **5.2.1 Analisis Proses Deployment**

Proses deployment menggunakan Docker dan Docker Hub memberikan banyak keuntungan, di antaranya:

1. Konsistensi Lingkungan: Dengan Docker, aplikasi dapat dijalankan di berbagai sistem operasi tanpa perlu khawatir perbedaan environment.

2. Kemudahan Distribusi: Image aplikasi yang sudah di-push ke Docker Hub dapat di-pull dan dijalankan di komputer atau server mana pun dengan mudah.
3. Manajemen Multi-Container: Docker Compose memudahkan pengelolaan beberapa service sekaligus (aplikasi, database, phpMyAdmin) dalam satu perintah.

### **5.2.2 Analisis Fitur Aplikasi**

Setiap fitur aplikasi berjalan sesuai dengan kebutuhan pengguna:

1. CRUD Resep: Pengguna dapat menambah, mencari, mengedit, dan menghapus resep dengan mudah.
2. Pencarian dan Random: Fitur pencarian dan random resep memudahkan pengguna menemukan inspirasi masakan.
3. Riwayat Pencarian: Fitur ini membantu pengguna mengakses kembali resep yang pernah dicari.
4. phpMyAdmin: Memudahkan pengelolaan database secara visual, sangat membantu untuk debugging dan administrasi data.

### **5.2.3 Kendala dan Solusi**

Beberapa kendala yang dihadapi selama proses implementasi antara lain:

1. Error pada build/push image: Biasanya disebabkan oleh kesalahan penulisan nama image atau belum login ke Docker Hub. Solusi: pastikan nama image benar dan sudah login.
2. Masalah port bentrok: Jika port yang digunakan sudah dipakai aplikasi lain, perlu mengganti port di file docker-compose.yml.
3. Integrasi antar service: Pastikan environment variable dan dependensi antar service sudah benar diatur di docker-compose.

### **5.2.4 Evaluasi**

Secara keseluruhan, aplikasi Resepku berhasil di-deploy dan didistribusikan dengan baik menggunakan Docker dan Docker Hub. Semua fitur berjalan sesuai harapan, dan deployment menjadi lebih mudah, cepat, dan konsisten.

## **BAB VI**

### **KESIMPULAN**

Berdasarkan hasil implementasi, pengujian, dan pembahasan yang telah dilakukan, dapat diambil beberapa kesimpulan sebagai berikut:

1. Aplikasi Resepku berhasil dibangun dan diimplementasikan menggunakan framework Flask untuk backend, MariaDB sebagai database, serta phpMyAdmin sebagai alat bantu administrasi database. Seluruh fitur utama aplikasi, seperti tambah, cari, edit, hapus, random resep, dan riwayat pencarian, dapat berjalan dengan baik sesuai kebutuhan pengguna.
2. Penggunaan Docker dan Docker Compose sangat memudahkan proses deployment aplikasi. Dengan mengemas aplikasi dan seluruh dependensinya ke dalam container, proses setup menjadi lebih cepat, konsisten, dan minim error, baik di lingkungan pengembangan maupun produksi.
3. Distribusi aplikasi melalui Docker Hub terbukti efektif untuk membagikan image aplikasi ke berbagai komputer atau server. Proses build, push, dan pull image dapat dilakukan dengan mudah, sehingga aplikasi dapat dijalankan di mana saja tanpa perlu konfigurasi ulang yang rumit.
4. Manajemen multi-container dengan Docker Compose memungkinkan integrasi yang lancar antara aplikasi, database, dan phpMyAdmin. Semua service dapat dijalankan, dihentikan, dan dikelola secara bersamaan hanya dengan satu perintah, sehingga sangat efisien untuk pengembangan dan deployment.
5. Pengujian aplikasi menunjukkan bahwa seluruh fitur berjalan sesuai harapan. Pengguna dapat mengelola resep dengan mudah, melakukan pencarian, melihat detail, mengedit, menghapus, serta mendapatkan rekomendasi resep secara acak. Fitur riwayat pencarian juga menambah kenyamanan pengguna dalam mengakses resep yang pernah dicari.
6. phpMyAdmin sangat membantu dalam administrasi database, baik untuk keperluan pengujian, debugging, maupun pemeliharaan data. Pengelolaan

database menjadi lebih mudah dan visual tanpa perlu perintah SQL manual.

7. Dokumentasi dan penjelasan langkah-langkah deployment yang detail sangat penting untuk memastikan proses dapat direplikasi oleh siapa saja. Dengan dokumentasi yang baik, anggota tim lain atau pengguna baru dapat dengan mudah memahami dan menjalankan aplikasi.
8. Kendala yang muncul selama proses implementasi, seperti error pada build image, push ke Docker Hub, atau pengaturan environment, dapat diatasi dengan troubleshooting yang tepat. Hal ini menunjukkan pentingnya pemahaman terhadap tools yang digunakan serta ketelitian dalam mengikuti setiap langkah.
9. Penggunaan teknologi container seperti Docker sangat relevan untuk pengembangan aplikasi modern, terutama dalam konteks DevOps, Continuous Integration, dan Continuous Deployment (CI/CD). Pengalaman ini menjadi bekal penting untuk pengembangan aplikasi skala lebih besar di masa depan.
10. Secara keseluruhan, proyek ini membuktikan bahwa deployment aplikasi berbasis web dapat dilakukan dengan cara yang efisien, terstandarisasi, dan mudah didistribusikan menggunakan Docker dan Docker Hub, sehingga aplikasi siap digunakan dan dikembangkan lebih lanjut.

## **DAFTAR PUSTAKA**

Docker Documentation. <https://docs.docker.com/>

Docker Hub. <https://hub.docker.com/>

Flask Documentation. <https://flask.palletsprojects.com/>

MariaDB Documentation. <https://mariadb.com/kb/en/documentation/>

phpMyAdmin Documentation. <https://www.phpmyadmin.net/docs/>

[Link GitHub Proyek]

[https://github.com/Fxlyn/WebApp\\_Resepku/tree/main/ProjekCloud](https://github.com/Fxlyn/WebApp_Resepku/tree/main/ProjekCloud)