

MAKALAH
Kontainerisasi Aplikasi E-Voting (e-Pilketos) Menggunakan
Docker untuk Menciptakan Lingkungan Pengembangan yang
Portabel



Disusun Oleh:

Nama : Akhmad Syaifudin

Nim : 32602300014

PROGRAM STUDI S1 TEKNIK INFORMATIKA
FAKULTAS TEKNOLOGI INDUSTRI
UNIVERSITAS ISLAM SULTAN AGUNG
SEMARANG

2025

ABSTRAK

Pengembangan aplikasi web modern sering kali menghadapi tantangan terkait konsistensi lingkungan antara mesin pengembang yang satu dengan yang lain, serta antara lingkungan pengembangan dan produksi. Praktikum ini bertujuan untuk mengatasi masalah tersebut dengan menerapkan teknologi kontainerisasi pada aplikasi e-voting berbasis PHP bernama "e-Pilketos". Dengan menggunakan Docker dan Docker Compose, seluruh komponen aplikasi termasuk server web Apache, runtime PHP, dan database MySQL dibungkus dalam kontainer yang terisolasi. Laporan ini merinci proses penyiapan lingkungan, mulai dari pembuatan `Dockerfile` untuk mendefinisikan image aplikasi, hingga konfigurasi `docker-compose.yml` untuk mengorkestrasi layanan-layanan yang ada. Hasil dari praktikum ini adalah sebuah lingkungan pengembangan yang portabel, konsisten, dan mudah didistribusikan, yang dapat dijalankan pada sistem operasi apa pun yang mendukung Docker hanya dengan satu perintah. Implementasi ini secara efektif menyelesaikan masalah "it works on my machine" dan menyederhanakan proses onboarding bagi pengembang baru serta proses deployment ke server.

Kata Kunci: Docker, Docker Compose, Kontainerisasi, PHP, MySQL, Lingkungan Pengembangan Portabel, e-Pilketos.

KATA PENGANTAR

Puji syukur penulis panjatkan kepada ALLAH SWT, yang telah memberikan rahmat, taufik serta hidayah-Nya, sehingga laporan Algoritma dan Struktur Data dapat terselesaikan.

Tanpa lupa penulis mengucapkan terima kasih kepada :

1. Rektor UNISSULA Bapak Prof. Dr. H. Gunarto, S.H., M.H yang mengijinkan penulis menimba ilmu di kampus ini
2. Dekan Fakultas Teknologi Industri Ibu Dr. Ir. Hj.Novi Marlyana, S.T., M.T
3. Dosen pengampu penulis Sam Farisa Chaerul Haviana, ST., M.Kom yang telah memberi ilmu tentang Cloud Computing
4. Orang tua penulis yang telah mengijinkan untuk menyelesaikan makalah ini,
5. Dan kepada semua pihak yang tidak dapat saya satu persatu.

Penulis menyadari bahwa dalam penyusunan laporan ini masih terdapat banyak kekurangan, untuk itu penulis mengharap kritik dan saran dari pembaca untuk sempurnanya laporan ini. Semoga dengan ditulisnya laporan ini dapat menjadi sumber ilmu bagi setiap pembaca.

Semarang, Juli 2025

Akhmad Syaifudin

DAFTAR ISI

| | |
|--|------------|
| HALAMAN JUDUL | i |
| ABSTRAK | ii |
| KATA PENGANTAR..... | iii |
| DAFTAR ISI..... | iv |
| DAFTAR GAMBAR..... | v |
| DAFTAR TABEL | vi |
| BAB I: PENDAHULUAN | 1 |
| 1.2 Rumusan Masalah..... | 1 |
| 1.3 Tujuan Praktikum | 2 |
| BAB II: DASAR TEORI | 3 |
| 2.1 Aplikasi e-Pilketos | 3 |
| 2.2 Teknologi Kontainerisasi | 3 |
| 2.3 Docker..... | 3 |
| BAB III METODOLOGI PRAKTIKUM..... | 5 |
| 3.1 Alat dan Bahan..... | 5 |
| 3.2 Alur Kerja Aplikasi (<i>Flowchart</i>) | 7 |
| 3.3.1 Flowchart Alur Pengguna Siswa | 7 |
| 3.3.2 Flowchart Alur Pengguna Admin..... | 9 |
| 3.3 Prosedur Kontainerisasi | 9 |
| BAB IV: HASIL DAN PEMBAHASAN | 14 |
| 4.1 Hasil Eksekusi Lingkungan. | 14 |
| 4.2 Demonstrasi Fungsionalitas Aplikasi | 14 |
| 4.3 Analisis Konfigurasi | 17 |
| BAB V: PENUTUP | 18 |
| 5.1 Kesimpulan | 18 |
| 5.2 Saran | 18 |
| DAFTAR PUSTAKA | 19 |

DAFTAR GAMBAR

| | |
|---|----|
| Gambar 3. 1 Docker Dekstop | 5 |
| Gambar 3. 2 Proses Instalasi | 5 |
| Gambar 3. 3 Tampilan Awal Daftar Docker | 6 |
| Gambar 3. 4 Tampilan Setelah Daftar | 6 |
| Gambar 3. 5 Struktur Folder | 7 |
| Gambar 3. 6 Flowchart Siswa | 8 |
| Gambar 3. 7 Flowchart Siswa | 8 |
| Gambar 3. 8 Flowchart Admin | 9 |
| Gambar 3. 9 Docker File | 9 |
| Gambar 3. 10 Docker Compose-yml | 11 |
| Gambar 3. 11 Menjalankan Lingkungan | 13 |
| Gambar 3. 12 Menjalankan Lingkungan | 13 |
| Gambar 4. 1 Hasil Eksekusi Lingkungan | 14 |
| Gambar 4. 2 Akses Aplikasi | 14 |
| Gambar 4. 3 Akses Registrasi | 15 |
| Gambar 4. 4 Login Akun yang Sudah di Registrasi | 15 |
| Gambar 4. 5 Tampilan Sukses Login | 15 |
| Gambar 4. 6 Tampilan Kandidat | 16 |
| Gambar 4. 7 Konfirmasi Suara | 16 |
| Gambar 4. 8 Hasil Setelah Vote | 16 |
| Gambar 4. 9 Tampilan Hasil Vote | 17 |
| Gambar 4. 10 Tampilan Database | 17 |

DAFTAR TABEL

| | |
|--|-------------------------------------|
| Tabel 1. 1 Daftar Versi Ubuntu..... | Error! Bookmark not defined. |
| Tabel 3. 1 Tabel <i>Direktori</i> | Error! Bookmark not defined. |
| Tabel 5. 1 Tabel Perintah Manajemen <i>User</i> | Error! Bookmark not defined. |
| Tabel 5. 2 Tabel Perintah Manajemen <i>Group</i> | Error! Bookmark not defined. |
| Tabel 5. 3 Tabel Tugas 1 | Error! Bookmark not defined. |

BAB I: PENDAHULUAN

1.1 Latar Belakang

Dalam siklus hidup pengembangan perangkat lunak, salah satu tantangan utama adalah memastikan konsistensi lingkungan. Sering kali, sebuah aplikasi yang berjalan sempurna di komputer seorang pengembang gagal berfungsi di komputer lain atau di server produksi. Masalah ini, yang populer dengan sebutan "it works on my machine", umumnya disebabkan oleh perbedaan versi pustaka, konfigurasi sistem operasi, atau dependensi lainnya.

Proyek e-Pilketos, sebuah sistem pemilihan ketua OSIS berbasis web dengan PHP dan MySQL, merupakan contoh aplikasi yang rentan terhadap masalah ini. Untuk menjalankannya, seorang pengembang harus menginstal server web (seperti Apache), versi PHP yang sesuai, ekstensi PHP yang dibutuhkan (seperti `mysqli`), dan server database MySQL. Proses instalasi manual ini tidak efisien, rawan kesalahan, dan sulit direplikasi secara identik.

Teknologi containerisasi, yang dipopulerkan oleh Docker, menawarkan solusi elegan untuk masalah ini. Dengan membungkus aplikasi beserta seluruh dependensinya ke dalam unit standar yang disebut kontainer, Docker memastikan bahwa aplikasi akan berjalan dengan cara yang sama di mana pun ia dijalankan.

1.2 Rumusan Masalah

Berdasarkan latar belakang di atas, rumusan masalah dalam praktikum ini adalah sebagai berikut:

1. Bagaimana cara mendefinisikan lingkungan yang dibutuhkan oleh aplikasi e-Pilketos (Apache, PHP, MySQL) secara deklaratif?
2. Bagaimana cara mengemas aplikasi e-Pilketos dan semua dependensinya ke dalam sebuah image yang portabel?
3. Bagaimana cara mengelola dan menjalankan beberapa layanan (aplikasi web dan database) secara bersamaan dan terhubung satu sama lain menggunakan Docker?

1.3 Tujuan Praktikum

Tujuan dari praktikum ini adalah:

1. Mampu membuat `Dockerfile` untuk membangun image Docker kustom bagi aplikasi e-Pilketos.
2. Mampu membuat file `docker-compose.yml` untuk mendefinisikan, mengkonfigurasi, dan menjalankan arsitektur multi-kontainer.
3. Membangun sebuah lingkungan pengembangan yang terisolasi, konsisten, dan mudah didistribusikan kepada anggota tim lain atau untuk keperluan deployment.

BAB II: DASAR TEORI

2.1 Aplikasi e-Pilketos

e-Pilketos adalah sebuah aplikasi berbasis web yang dirancang untuk memfasilitasi proses pemilihan ketua OSIS secara elektronik. Pengembangan sistem e-voting seperti ini merupakan topik yang relevan di lingkungan pendidikan untuk memperkenalkan proses demokrasi digital yang transparan dan efisien (Pradana et al., 2024). Arsitektur aplikasi ini terdiri dari:

- Frontend: HTML, CSS, dan JavaScript untuk antarmuka pengguna.
- Backend: Bahasa pemrograman PHP untuk memproses logika bisnis, otentikasi pengguna, dan pemrosesan suara.
- Database: MySQL untuk menyimpan data pengguna, kandidat, dan hasil suara.
- Web Server: Apache untuk melayani permintaan HTTP dari pengguna.

2.2 Teknologi Kontainerisasi

Kontainerisasi adalah sebuah metode virtualisasi tingkat sistem operasi yang memungkinkan sebuah aplikasi beserta seluruh dependensinya dibungkus dan dijalankan dalam sebuah unit terisolasi yang disebut kontainer. Setiap kontainer berjalan secara konsisten di lingkungan komputasi mana pun, yang secara efektif memecahkan masalah perbedaan konfigurasi antar mesin pengembang (Ekaputra & Affandi, 2023).

2.3 Docker

Docker adalah platform perangkat lunak open-source terdepan yang mengimplementasikan teknologi kontainerisasi. Docker menyederhanakan proses pembuatan, penyebaran, dan pengelolaan aplikasi di dalam kontainer. Pendekatan ini sangat bermanfaat untuk mengelola aplikasi web agar lebih efisien (Bik, 2017). Konsep utama Docker meliputi:

- **Dockerfile:** Sebuah file teks yang berisi serangkaian instruksi untuk membangun sebuah image Docker.

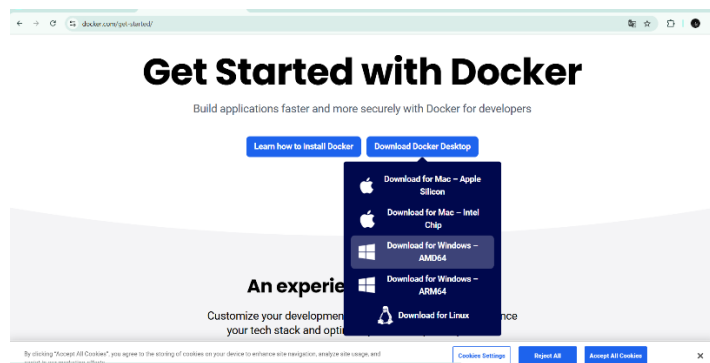
- Image: Sebuah paket *read-only* yang berisi semua yang dibutuhkan untuk menjalankan aplikasi: kode, runtime, pustaka, dan variabel lingkungan.
- Container: Sebuah instans dari sebuah image yang sedang berjalan.

BAB III METODOLOGI PRAKTIKUM

3.1 Alat dan Bahan

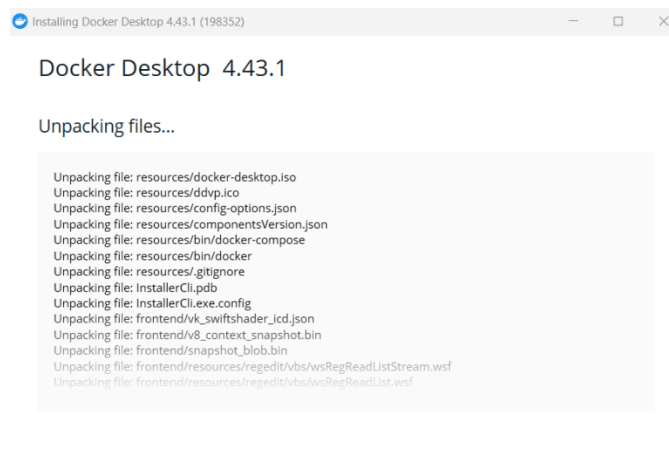
Perangkat Keras: Komputer/Laptop dengan sistem operasi Windows, macOS, atau Linux.

1. Docker Desktop



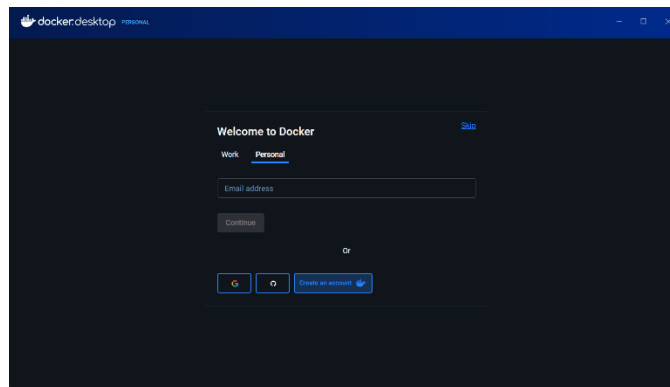
Gambar 3. 1 docker dekstop

Yang pertama anda mendownload aplikasinya melalui website <https://www.docker.com/>, setelah itu anda memilih sesuai spesifikasi laptop yang anda gunakan



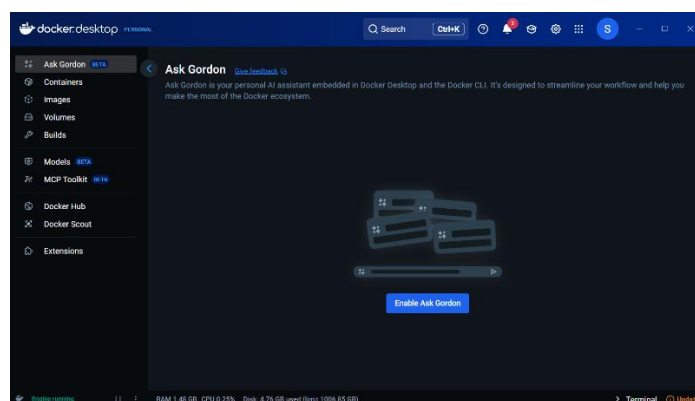
Gambar 3. 2 Proses Instalasi

Setelah download anda selesai, mulai set up install pencet tombol *next* terus saja dan tunggu install selesai



Gambar 3. 3 Tampilan awal daftar docker

Jika anda sudah mempunyai akun silahkan login, dan jika belum maka anda membuat akun terlebih dahulu. Jika membuat akun pilihlah sesuai kebutuhan anda untuk *Work / Personal*.



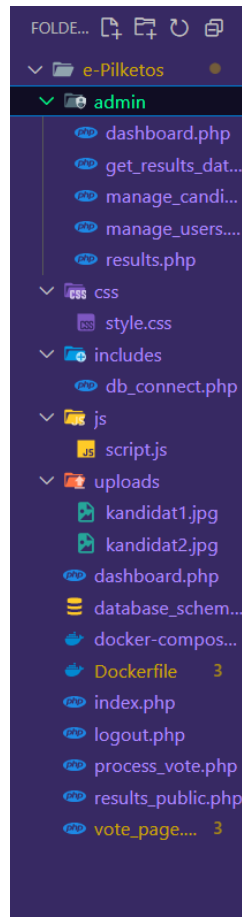
Gambar 3. 4 Tampilan Setelah Daftar

Setelah membuat akun maka tampilan akan seperti diatas menunjukkan berhasil install docker

2. Visual Studio Code (atau teks editor lain)
3. Source code proyek e-Pilketos

3.1 Struktur Proyek

Proyek ini diorganisir dengan struktur folder sebagai berikut, yang penting untuk pemetaan volume di Docker.



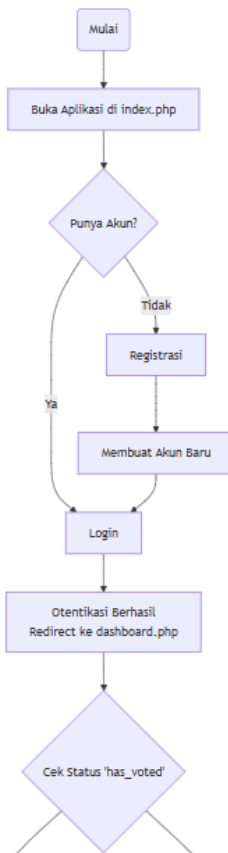
Gambar 3.5 Struktur Folder

3.2 Alur Kerja Aplikasi (*Flowchart*)

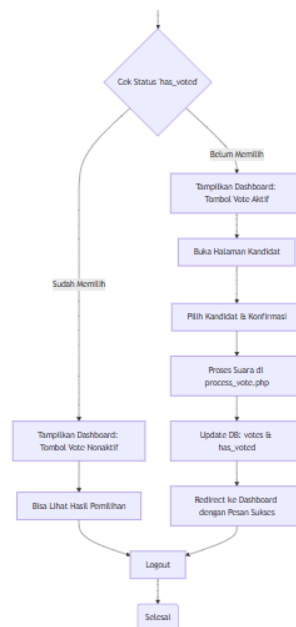
Sebelum melakukan kontainerisasi, penting untuk memahami alur kerja (flowchart) dari aplikasi e-Pilketos. Aplikasi ini memiliki dua alur utama berdasarkan peran pengguna: Siswa dan Admin.

3.3.1 Flowchart Alur Pengguna Siswa

Flowchart ini menggambarkan perjalanan pengguna dengan peran 'siswa', mulai dari halaman awal hingga selesai memberikan suara



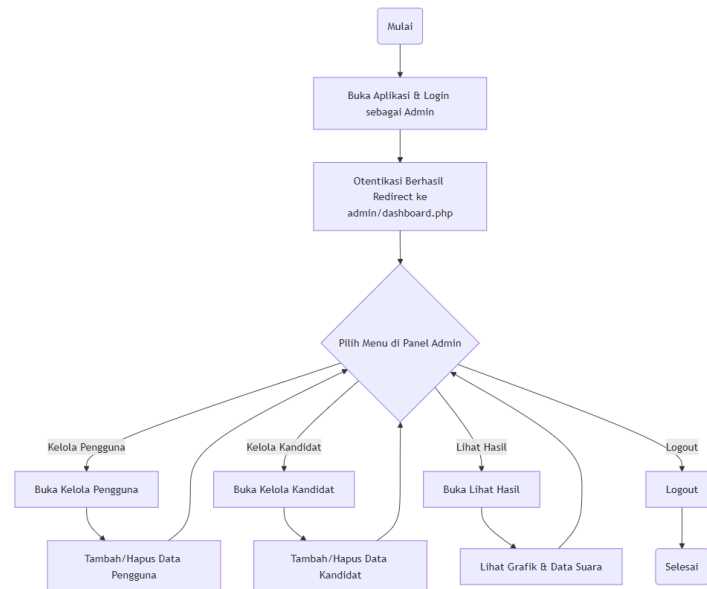
Gambar 3. 6 Flowchart Siswa



Gambar 3. 7 Flowchart Siswa

3.3.2 Flowchart Alur Pengguna Admin

Flowchart ini menggambarkan perjalanan pengguna dengan peran 'admin' untuk mengelola sistem.

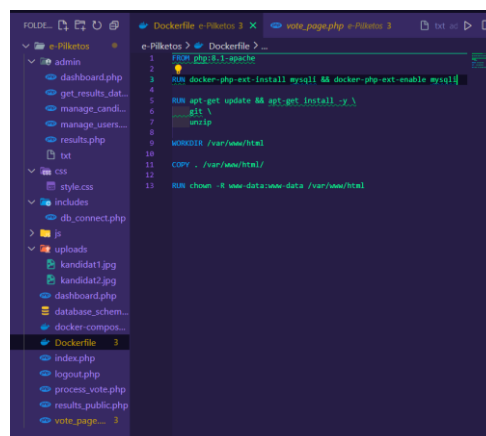


Gambar 3. 8 *Flowchart* Admin

3.3 Prosedur Kontainerisasi

Langkah-langkah untuk melakukan kontainerisasi aplikasi adalah sebagai berikut:

1. Pembuatan `Dockerfile` `Dockerfile` digunakan untuk membuat image dari aplikasi PHP kita.



Gambar 3. 9 Docker File

a. *Source code*

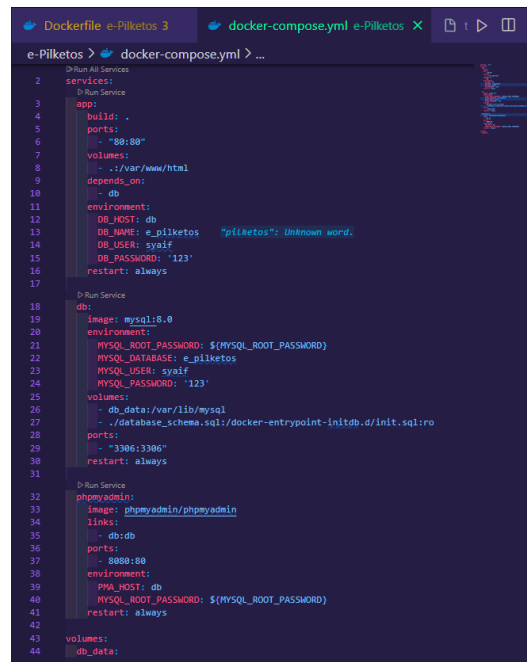
```
FROM php:8.1-apache
RUN docker-php-ext-install mysqli && docker-php-ext-enable
mysqli
RUN apt-get update && apt-get install -y \
    git \
    unzip
WORKDIR /var/www/html
COPY . /var/www/html/
RUN chown -R www-data:www-data /var/www/html
```

b. *Penjelasan*

Proses pembangunan image Docker untuk aplikasi ini diawali dengan menggunakan image dasar resmi `php:8.1-apache`, yang kemudian diperkaya dengan instalasi ekstensi `mysqli` untuk memastikan konektivitas database. Selanjutnya, direktori kerja di dalam kontainer diatur ke `/var/www/html`, lalu seluruh file proyek disalin dari direktori lokal ke lokasi tersebut. Sebagai langkah akhir, kepemilikan file diubah menjadi milik pengguna server web `www-data` untuk memberikan hak akses yang diperlukan agar aplikasi dapat berjalan dengan benar.

2. Pembuatan `docker-compose.yml`

File ini mengorkestrasi dua layanan utama: `app` (aplikasi PHP) dan `db` (database MySQL).



Gambar 3. 10 Docker Compose-yml

a. Source code

```

version: '3.8'

services:

  app:
    build: .
    ports:
      - "80:80"
    volumes:
      - ./var/www/html
    depends_on:
      - db
    environment:
      DB_HOST: db
      DB_NAME: e_pilketos
      DB_USER: syaif
      DB_PASSWORD: '123'
    restart: always

  db:
    image: mysql:8.0
    environment:

```

```

MYSQL_ROOT_PASSWORD: ${MYSQL_ROOT_PASSWORD}
MYSQL_DATABASE: e_pilketos
MYSQL_USER: syaif
MYSQL_PASSWORD: '123'
volumes:
- db_data:/var/lib/mysql
- ./database_schema.sql:/docker-entrypoint-
initdb.d/init.sql:ro
ports:
- "3306:3306"
restart: always

phpmyadmin:
image: phpmyadmin/phpmyadmin
links:
- db:db
ports:
- 8080:80
environment:
PMA_HOST: db
MYSQL_ROOT_PASSWORD: ${MYSQL_ROOT_PASSWORD}
restart: always
volumes:
db_data:

```

b. Penjelasan

File `docker-compose.yml` ini mendefinisikan arsitektur multi-kontainer dengan dua layanan utama: `app` dan `db`. Layanan `app` dibangun dari `Dockerfile` lokal, memetakan port 80 untuk akses web, dan menghubungkan kode sumbernya secara langsung untuk pengembangan live. Layanan `db` menggunakan image resmi `mysql:8.0`, memastikan data tersimpan permanen melalui volume, serta mengimpor skema database secara otomatis saat pertama kali dijalankan. Keduanya terhubung melalui jaringan internal Docker, di mana `app` menggunakan `db` sebagai alamat host databasenya yang diatur via variabel lingkungan,

dan depends_on menjamin database selalu siap sebelum aplikasi dimulai.

3. Menjalankan Lingkungan docker-compose up -d --build / docker-compose up -d

```

Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\fitri\Downloads\le-Pilketos\le-Pilketos> docker-compose up -d
time="2025-07-10T13:06:12+07:00" level=warning msg="C:\Users\fitri\Downloads\le-Pilketos\le-Pilketos\docker-compose.yml: the attribute 'build' is deprecated, use 'service' instead"
[+] Building 89.0s (14/14) FINISHED
=> [internal] load local bake definitions 0.0s
=> => reading from stdin 0.0s
=> [internal] load build definition from Dockerfile 0.0s
=> => transferring dockerfile: 1.69kB 0.0s
=> [internal] load metadata for docker.io/library/php:8.1-apache 4.2s
=> [auth] library/php:pull token for registry-1.docker.io 0.0s
=> [internal] load dockerignore 0.0s
=> => transferring context: 2B 0.0s
=> [1/6] FROM docker.io/library/php:8.1-apache@sha256:f1b7b9ac4a103c6cde1a1eb3b5e06a5b4e855c763fe3b122d8 53.2s
=> => resolve docker.io/library/php:8.1-apache@sha256:f1b7b9ac4a103c6cde1a1eb3b5e06a5b4e855c763fe3b122d8 0.0s
=> => sha256:fed346587590935feb9715f96f08771a631c2a75c0475757e9b05d1404f 892B / 892B 0.4s
=> => sha256:1215c71937e9f773a9e809bc796d591445f2857d1ca1c752ddad8c89814c654 243B / 243B 0.7s
=> => sha256:89476a2bbdb0cee95c565732670434d22b06b23c3535a161d7769cdca18ccd 248B / 248B 1.0s
=> => sha256:225778fdeccf587ad6c1e79895e290ba7e44c0467087cd3798cf92cab360ab3 2.40kB / 2.40kB 1.0s
=> => sha256:18c41e55d4f22f34b0a5e93aa63acc41bcb99b52df5b97038f229f8cddd 11.16MB / 11.16MB 7.8s
=> => sha256:bfa4126235f78d5e12ca7de5e5807e3f407df479938ec41d3d45c80e02edbf4 489B / 489B 0.4s
=> => sha256:41f34298a3d069fb6091d421046702726f5c6f14cd1f369c1e9dc56c93e7b 12.03MB / 12.03MB 16.8s
=> => sha256:1660f69bd2c2f7e54052a49d168e6c56b0a421a9f92a56fc2d6e4c3f310 488B / 488B 0.7s
=> => sha256:cd80672d0b5d2126a0418e40918611ab31aa7a3aba566c7a262b9a014 227B / 227B 0.7s
=> => sha256:a041ea15491fc46539cf844ab5b9a32599fa6837da558bb2b2a08d10475b370 28.12MB / 28.12MB 15.7s
=> => extracting sha256:cd80672d0b5d2126a0418e40918611ab31aa7a3aba566c7a262b9a014 0.0s
=> => sha256:b5331372a28a37fc362c544cf3f7d373a9c7208516d23db137f1716f2f6595 0.0s
=> => sha256:9ab567f0c2e38352baba8a117cc5c1e6d7908aa17cf05485a211c2a2907 225B / 225B 0.8s
=> => sha256:3f90d7cb2cb0e842ba415bc0a183f35beb66ca67bb5672d179132768f8a2cd 104.33MB / 104.33MB 45.3s
=> => extracting sha256:3f90d7cb2cb0e842ba415bc0a183f35beb66ca67bb5672d179132768f8a2cd 2.6s
=> => extracting sha256:9ab567f0c2e38352baba8a117cc5c1e6d7908aa17cf05485a211c2a2907 0.0s
=> => sha256:a041ea15491fc46539cf844ab5b9a32599fa6837da558bb2b2a08d10475b370 0.0s
=> => extracting sha256:b5331372a28a37fc362c544cf3f7d373a9c7208516d23db137f1716f2f6595 0.0s
=> => extracting sha256:1660f69bd2c2f7e54052a49d168e6c56b0a421a9f92a56fc2d6e4c3f310 0.0s
=> => extracting sha256:41f34298a3d069fb6091d421046702726f5c6f14cd1f369c1e9dc56c93e7b 0.1s
=> => extracting sha256:bfa4126235f78d5e12ca7de5e5807e3f407df479938ec41d3d45c80e02edbf4 0.6s
=> => extracting sha256:88cd18855df225f3db04ac93aa63acc41bcb99b52df5b97038f229f8cddd 0.3s
=> => extracting sha256:225778fdeccf587ad6c1e79895e290ba7e44c0467087cd3798cf92cab360ab3 0.0s
=> => extracting sha256:89476a2bbdb0cee95c565732670434d22b06b23c3535a161d7769cdca18ccd 0.0s
=> => extracting sha256:1215c71937e9f773a9e809bc796d591445f2857d1ca1c752ddad8c89814c654 0.6s
=> => extracting sha256:fed346587590935feb9715f96f08771a631c2a75c0475757e9b05d1404f 0.6s
=> => extracting sha256:4f49f708e6f54461cfa02571ae0b9a0dc1e0cd5577484a6d75e68dc38e8ac1 0.1s
=> [internal] load build context 0.1s
=> => transferring context: 638.40kB 0.1s
=> [2/6] RUN docker-php-ext-install mysqli && docker-php-ext-enable mysqli 10.2s
=> [3/6] RUN apt-get update && apt-get install -y git unzip 15.1s
=> [4/6] WORKDIR /var/www/html/ 0.1s
=> [5/6] COPY ./var/www/html/ 0.1s
=> [6/6] RUN chown -R www-data:www-data /var/www/html 0.4s
=> => exporting to image 4.6s
=> => exporting layers 3.6s
=> => exporting manifest sha256:3226c6549905d1b0d3d1f4fdd9daf4f2935b863f1b1a41cb23576704dbelaw7 0.0s
=> => exporting config sha256:b113be69aefc551e62773c583c7d87d48dac0e676321c252755d6f1f074be 0.0s
=> => exporting attestation manifest sha256:2a82cc4ad074c68abb34a3b6c204ff5f5fc21aaf5a3d4db38bd53be5707 0.0s
=> => exporting manifest list sha256:04c30d12ae0f640497aa566caba90f4bbf0c88cd66978c505bd348e0620f51 0.0s
=> => naming to docker.io/library/e-pilketos-app:latest 0.0s
=> => unpacking to docker.io/library/e-pilketos-app:latest 0.8s
=> => resolving provenance for metadata file 0.0s
[+] Running 5/6
✔app Built 0.0s
✔Network e-pilketos_default Created 0.1s

```

Gambar 3. 11 Menjalankan Lingkungan

```

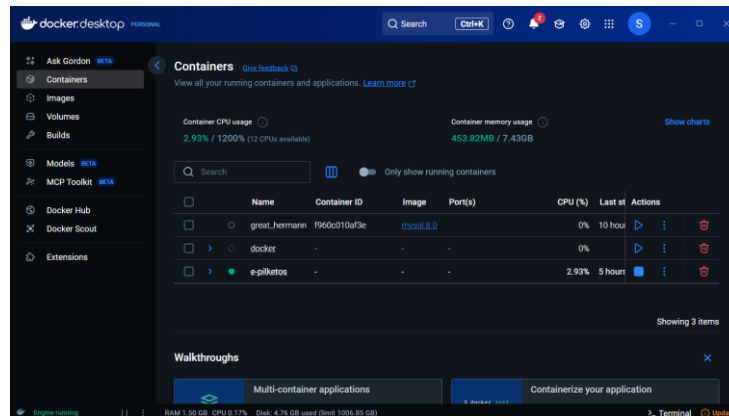
PS C:\Users\fitri\Downloads\le-Pilketos\le-Pilketos> docker-compose up -d
time="2025-07-10T13:12:32+07:00" level=warning msg="C:\Users\fitri\Downloads\le-Pilketos\le-Pilketos\docker-compose.yml: the attribute 'build' is deprecated, use 'service' instead"
[+] Building 89.0s (14/14) FINISHED
=> [internal] load local bake definitions 0.0s
=> => reading from stdin 0.0s
=> [internal] load build definition from Dockerfile 0.0s
=> => transferring dockerfile: 1.69kB 0.0s
=> [internal] load metadata for docker.io/library/php:8.1-apache 4.2s
=> [auth] library/php:pull token for registry-1.docker.io 0.0s
=> [internal] load dockerignore 0.0s
=> => transferring context: 2B 0.0s
=> [1/6] FROM docker.io/library/php:8.1-apache@sha256:f1b7b9ac4a103c6cde1a1eb3b5e06a5b4e855c763fe3b122d8 53.2s
=> => resolve docker.io/library/php:8.1-apache@sha256:f1b7b9ac4a103c6cde1a1eb3b5e06a5b4e855c763fe3b122d8 0.0s
=> => sha256:fed346587590935feb9715f96f08771a631c2a75c0475757e9b05d1404f 892B / 892B 0.4s
=> => sha256:1215c71937e9f773a9e809bc796d591445f2857d1ca1c752ddad8c89814c654 243B / 243B 0.7s
=> => sha256:89476a2bbdb0cee95c565732670434d22b06b23c3535a161d7769cdca18ccd 248B / 248B 1.0s
=> => sha256:225778fdeccf587ad6c1e79895e290ba7e44c0467087cd3798cf92cab360ab3 2.40kB / 2.40kB 1.0s
=> => sha256:18c41e55d4f22f34b0a5e93aa63acc41bcb99b52df5b97038f229f8cddd 11.16MB / 11.16MB 7.8s
=> => sha256:bfa4126235f78d5e12ca7de5e5807e3f407df479938ec41d3d45c80e02edbf4 489B / 489B 0.4s
=> => sha256:41f34298a3d069fb6091d421046702726f5c6f14cd1f369c1e9dc56c93e7b 12.03MB / 12.03MB 16.8s
=> => sha256:1660f69bd2c2f7e54052a49d168e6c56b0a421a9f92a56fc2d6e4c3f310 488B / 488B 0.7s
=> => sha256:cd80672d0b5d2126a0418e40918611ab31aa7a3aba566c7a262b9a014 227B / 227B 0.7s
=> => sha256:a041ea15491fc46539cf844ab5b9a32599fa6837da558bb2b2a08d10475b370 28.12MB / 28.12MB 15.7s
=> => extracting sha256:cd80672d0b5d2126a0418e40918611ab31aa7a3aba566c7a262b9a014 0.0s
=> => sha256:b5331372a28a37fc362c544cf3f7d373a9c7208516d23db137f1716f2f6595 0.0s
=> => sha256:9ab567f0c2e38352baba8a117cc5c1e6d7908aa17cf05485a211c2a2907 225B / 225B 0.8s
=> => sha256:3f90d7cb2cb0e842ba415bc0a183f35beb66ca67bb5672d179132768f8a2cd 104.33MB / 104.33MB 45.3s
=> => extracting sha256:3f90d7cb2cb0e842ba415bc0a183f35beb66ca67bb5672d179132768f8a2cd 2.6s
=> => extracting sha256:9ab567f0c2e38352baba8a117cc5c1e6d7908aa17cf05485a211c2a2907 0.0s
=> => sha256:a041ea15491fc46539cf844ab5b9a32599fa6837da558bb2b2a08d10475b370 0.0s
=> => extracting sha256:b5331372a28a37fc362c544cf3f7d373a9c7208516d23db137f1716f2f6595 0.0s
=> => extracting sha256:1660f69bd2c2f7e54052a49d168e6c56b0a421a9f92a56fc2d6e4c3f310 0.0s
=> => extracting sha256:41f34298a3d069fb6091d421046702726f5c6f14cd1f369c1e9dc56c93e7b 0.1s
=> => extracting sha256:bfa4126235f78d5e12ca7de5e5807e3f407df479938ec41d3d45c80e02edbf4 0.6s
=> => extracting sha256:88cd18855df225f3db04ac93aa63acc41bcb99b52df5b97038f229f8cddd 0.3s
=> => extracting sha256:225778fdeccf587ad6c1e79895e290ba7e44c0467087cd3798cf92cab360ab3 0.0s
=> => extracting sha256:89476a2bbdb0cee95c565732670434d22b06b23c3535a161d7769cdca18ccd 0.0s
=> => extracting sha256:1215c71937e9f773a9e809bc796d591445f2857d1ca1c752ddad8c89814c654 0.6s
=> => extracting sha256:fed346587590935feb9715f96f08771a631c2a75c0475757e9b05d1404f 0.6s
=> => extracting sha256:4f49f708e6f54461cfa02571ae0b9a0dc1e0cd5577484a6d75e68dc38e8ac1 0.1s
=> [internal] load build context 0.1s
=> => transferring context: 638.40kB 0.1s
=> [2/6] RUN docker-php-ext-install mysqli && docker-php-ext-enable mysqli 10.2s
=> [3/6] RUN apt-get update && apt-get install -y git unzip 15.1s
=> [4/6] WORKDIR /var/www/html/ 0.1s
=> [5/6] COPY ./var/www/html/ 0.1s
=> [6/6] RUN chown -R www-data:www-data /var/www/html 0.4s
=> => exporting to image 4.6s
=> => exporting layers 3.6s
=> => exporting manifest sha256:3226c6549905d1b0d3d1f4fdd9daf4f2935b863f1b1a41cb23576704dbelaw7 0.0s
=> => exporting config sha256:b113be69aefc551e62773c583c7d87d48dac0e676321c252755d6f1f074be 0.0s
=> => exporting attestation manifest sha256:2a82cc4ad074c68abb34a3b6c204ff5f5fc21aaf5a3d4db38bd53be5707 0.0s
=> => exporting manifest list sha256:04c30d12ae0f640497aa566caba90f4bbf0c88cd66978c505bd348e0620f51 0.0s
=> => naming to docker.io/library/e-pilketos-app:latest 0.0s
=> => unpacking to docker.io/library/e-pilketos-app:latest 0.8s
=> => resolving provenance for metadata file 0.0s
[+] Running 5/6
✔app Built 0.0s
✔Network e-pilketos_default Created 0.1s
✔Container e-pilketos-db-1 Started 0.0s
✔Container e-pilketos-app-1 Started 1.0s
✔Container e-pilketos-phpmyadmin-1 Started 1.0s

```

Gambar 3. 12 Menjalankan Lingkungan

BAB IV: HASIL DAN PEMBAHASAN

4.1 Hasil Eksekusi Lingkungan.



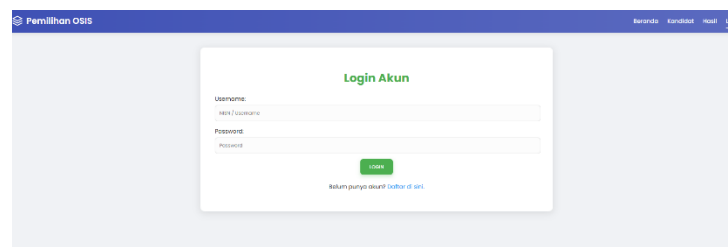
Gambar 4. 1 Hasil Eksekusi Lingkungan

Setelah menjalankan perintah `docker-compose up -d --build`, Docker berhasil membangun image aplikasi dan menjalankan semua layanan. Pengecekan status menggunakan perintah `docker-compose ps` menunjukkan bahwa semua kontainer berada dalam keadaan "Up" (berjalan), yang mengindikasikan bahwa penyiapan lingkungan kontainer berhasil

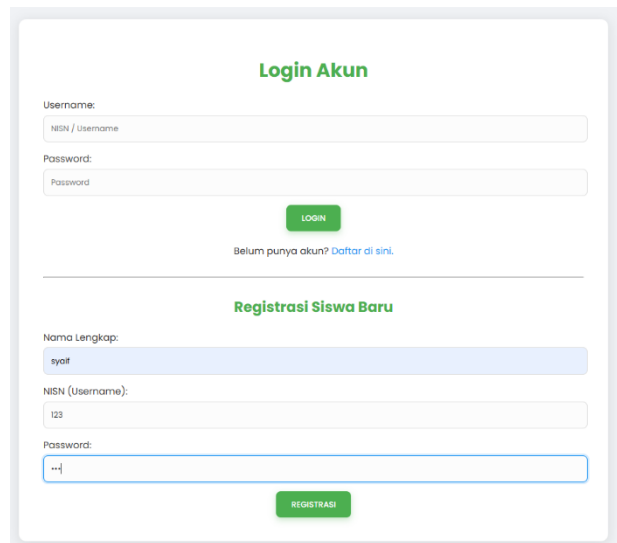
4.2 Demonstrasi Fungsionalitas Aplikasi

Untuk memverifikasi bahwa aplikasi berjalan dengan benar di dalam lingkungan Docker, serangkaian pengujian fungsionalitas dilakukan. Alur pengujian mencakup registrasi pengguna baru, login, proses voting, hingga pengecekan hasil.

1. Mengakses Aplikasi dan Registrasi Pengguna



Gambar 4. 2 Akses Aplikasi

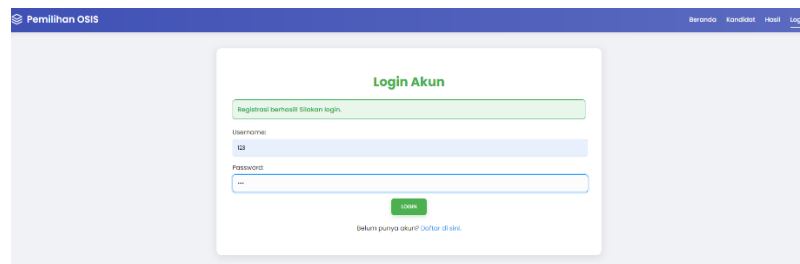


The image shows two forms stacked vertically. The top form is titled "Login Akun" and has fields for "Username:" (with a hint "NISN / Username") and "Password:". Below these is a green "LOGIN" button and a link "Belum punya akun? [Daftar di sini.](#)". The bottom form is titled "Registrasi Siswa Baru" and has fields for "Nama Lengkap:" (with the value "syait"), "NISN (Username):" (with the value "123"), and "Password:". Below these is a green "REGISTRASI" button.

Gambar 4. 3 Akses Registrasi

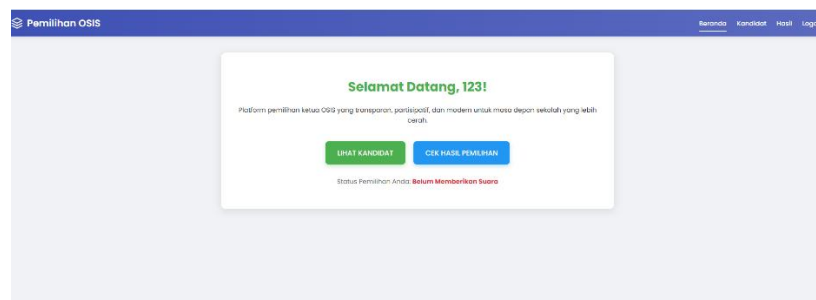
Aplikasi diakses pada `http://localhost:80`. Halaman login muncul, dan pengguna baru dapat melakukan registrasi.

2. Login dan Tampilan Dashboard



The image shows the "Login Akun" form within a larger application context. The header bar is blue and contains "Pemilihan OSIS" on the left and "Beranda", "Kandidat", "Hasil", and "Login" on the right. The login form itself is white and contains a green success message "Registrasi berhasil! Silakan login.", the "Username:" field with the value "123", the "Password:" field with a masked value, a green "LOGIN" button, and the link "Belum punya akun? [Daftar di sini.](#)".

Gambar 4. 4 Login Akun yang Sudah di Registrasi

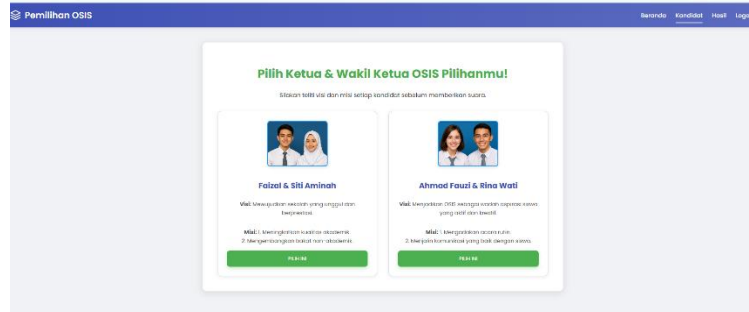


The image shows the dashboard after a successful login. The header bar is the same as in the previous screenshot. The main content area is white and features a green greeting "Selamat Datang, 123!". Below this is a descriptive sentence: "Platform pemilihan ketua OSIS yang transparan, partisipatif, dan modern untuk masa depan sekolah yang lebih cerah." There are two buttons: a green "LIHAT KANDIDAT" button and a blue "CEK HASIL PEMILIHAN" button. At the bottom, it says "Status Pemilihan Anda: Belum Memberikan Suara".

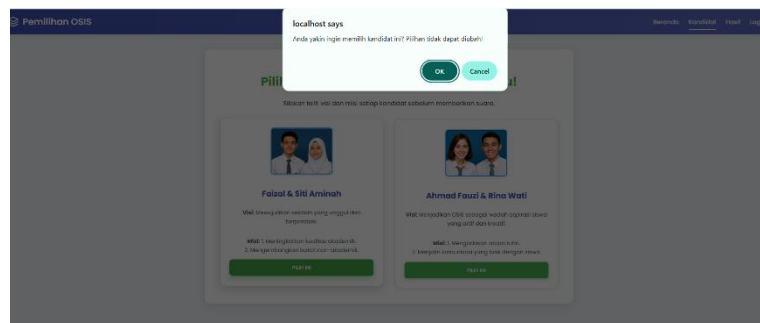
Gambar 4. 5 Tampilan Sukses Login

Setelah registrasi, pengguna login dan diarahkan ke dashboard, yang menampilkan status "Belum Memberikan Suara".

3. Proses Voting



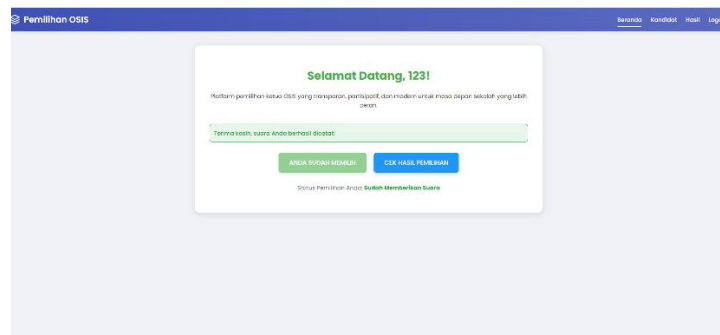
Gambar 4. 6 Tampilan Kandidat



Gambar 4. 7 Konfirmasi Suara

Pengguna memilih kandidat dari halaman pemilihan dan mengkonfirmasi suaranya.

4. Verifikasi Hasil Setelah Voting



Gambar 4. 8 Hasil Setelah Vote

Setelah voting, dashboard pengguna diperbarui. Status berubah menjadi "Sudah Memberikan Suara".

Hasil pemilihan dapat dilihat secara langsung, di mana suara yang baru saja diberikan sudah tercatat pada grafik.

The screenshot shows the phpMyAdmin web interface. The browser address bar displays 'localhost:8080/index.php?route=/sql&pos=0&db=e_phpmyadmin&table=users'. The interface title is 'phpMyAdmin'. The left sidebar shows a tree view with 'localhost' selected, containing 'phpmyadmin' and 'information_schema'. The main content area shows the 'users' table structure and data. The table has 7 columns: id, username, password, name, lastname, email, and last_login. Two rows of data are displayed: user 1 (admin) and user 123 (syed). The interface includes navigation tabs (Structure, SQL, Search, Insert, etc.), a status bar indicating 'Showing rows 1 - 2 total. Query took 0.0002 seconds.', and a 'Query results operations' section at the bottom.

Pengecekan melalui phpMyAdmin pada `http://localhost:8080` menunjukkan data pengguna baru telah tersimpan dan status `has_voted` telah diperbarui, membuktikan transaksi database berjalan sukses.

Kunci keberhasilan konektivitas antara aplikasi dan database terletak pada jaringan virtual internal yang dibuat oleh Docker Compose dan penggunaan variabel lingkungan. Aplikasi di dalam kontainer `app` berhasil menghubungi kontainer `db` menggunakan `db` sebagai hostname. Ini menunjukkan kekuatan Docker dalam menyederhanakan arsitektur microservices dan menciptakan lingkungan yang terdefinisi dengan baik.

BAB V: PENUTUP

5.1 Kesimpulan

Praktikum ini telah berhasil mendemonstrasikan proses kontainerisasi sebuah aplikasi web PHP multi-komponen (e-Pilketos) menggunakan Docker dan Docker Compose. Dengan mendefinisikan infrastruktur sebagai kode, berhasil diciptakan sebuah lingkungan pengembangan yang terisolasi, konsisten, dan sangat portabel. Pendekatan ini secara efektif memecahkan masalah dependensi dan konfigurasi manual, serta menyederhanakan alur kerja pengembangan secara keseluruhan.

5.2 Saran

Untuk pengembangan lebih lanjut, beberapa perbaikan dapat dipertimbangkan:

1. Keamanan: Menggunakan pengguna non-root untuk koneksi database di lingkungan produksi.
2. Optimasi Image: Menerapkan *multi-stage builds* pada `Dockerfile` untuk mengurangi ukuran image akhir.
3. CI/CD: Mengintegrasikan Docker ke dalam alur kerja *Continuous Integration/Continuous Deployment* untuk otomatisasi testing dan deployment.

DAFTAR PUSTAKA

- Bik, M. F. R. (2017). Implementasi Docker untuk pengelolaan banyak aplikasi web (Studi kasus: Jurusan Teknik Informatika UNESA). *Jurnal Manajemen Informatika*, 7(2).
- Ekaputra, A. R., & Affandi, A. S. (2023). Pemanfaatan layanan cloud computing dan docker container untuk meningkatkan kinerja aplikasi web. *Journal of Information System and Application Development*, 1(2), 138–147.
- Pradana, S. A., Andika, R., Wibowo, M. A. P., Hutagalung, M. R. S., Sipahutar, H. K., & Rizal, C. (2024). Perancangan Sistem Informasi E-Voting Berbasis Web Untuk Pemilihan Ketua Himpunan Di UIN Sumatera Utara Medan. *Jurnal Komputer Teknologi Informasi Dan Sistem Informasi (JUKTISI)*, 3(2), 782–793.