

LAPORAN PRAKTIKUM

Docker Flask Web Peminjaman



Disusun Oleh:

Nama : 1. Muhammad Ajra Zemima Muda
2. L Hafidl Alkhair

Kelas : TRKJ 2.C

Jurusan : Teknologi Informasi dan Komputer

Program Studi : Teknologi Rekayasa Komputer Jaringan

Dosen Pembimbing : Indrawati, SST., M.T.



JURUSAN TEKNOLOGI INFORMASI KOMPUTER
PRODI TEKNOLOGI REKAYASA KOMPUTER JARINGAN
POLITEKNIK NEGERI LHOKSEUMAWE
TAHUN AJARAN 2024-202

LEMBAR PENGESAHAN

No Praktikum : 02/TIK/TRKJ-2C/Komputasi Awan
Judul : Laporan Praktikum Docker Flask Web Peminjaman
Nama : 1. Muhammad Ajra Zemima Muda
2. L Hafidl Alkhair
Kelas : TRKJ 2.C
Jurusan : Teknologi Informasi dan Komputer
Prodi : Teknologi Rekayasa Komputer Jaringan
Tanggal Praktikum : Jum'at 15 Mei
Tanggal Penyerahan : Jum'at, 21 Juni 2025

Buketrata, 21 Juni 2025

Dosen Pembimbing,

Indrawati, SST., M.T.

NIP. 19740815 200112 2 001

DAFTAR ISI

LEMBAR PENGESAHAN	i
DAFTAR ISI.....	ii
DAFTAR GAMBAR	iii
BAB I.....	1
A. Dasar Teori.....	1
B. Alat dan bahan	2
BAB II	3
A. Clone Program	3
1. Buka github <i>https://github.com/hafidalkhair/app-peminjaman</i>	3
2. Salin github.....	3
3. Buka cmd.....	5
4. cd <i>\app-peminjaman\</i>	5
B. Folder proyek app peminjaman.	5
C. Membuat database.....	6
E. Membangun docker dalam program app peminjaman.....	7
1. Buat Dockerfile.....	7
2. Membuat requirements.txt	7
3. Kode program yang terdapat pada file Dockerfile.	8
4. Penjelasan kode program	8
5. Build container.....	10
6. Jalankan docker dan ngrok.....	12
BAB III	16
A. Kesimpulan	16

DAFTAR GAMBAR

Gambar 1 1 tampilan github.....	3
Gambar 1 1 tampilan github.....	3
Gambar 1 2 copy url github	3
Gambar 1 2 copy url github	3
Gambar 1 3 git clone git hub di terminal.....	5
Gambar 1 3 git clone git hub di terminal.....	5
Gambar 1 4 masuk direktori app-peminjaman	5
Gambar 1 5 sturktur folder	5
Gambar 1 5 sturktur folder	5
Gambar 1 6 databases	6
Gambar 1 6 databases	6
Gambar 1 7 Membuat Dockerfile didalam Program app-peminjaman	7
Gambar 1 7 Membuat Dockerfile didalam Program app-peminjaman	7
Gambar 1 8 tampilan docker container yang kosong.....	10
Gambar 1 8 tampilan docker container yang kosong.....	10
Gambar 1 9 docker berhasil di buat.....	10
Gambar 1 9 docker berhasil di buat.....	10
Gambar 1 10 tampilan docker images yang telah dibuat	10
Gambar 1 10 tampilan docker images yang telah dibuat	10
Gambar 1 11 container id untuk container yang sedang berjalan.	11
Gambar 1 11 container id untuk container yang sedang berjalan.	11
Gambar 1 12 tampilan app setelah docker dijalan kan	12
Gambar 1 12 tampilan app setelah docker dijalan kan	12
Gambar 1 13 intall ngrok	12
Gambar 1 13 intall ngrok	12
Gambar 1 14 auth token ngrok	13
Gambar 1 14 auth token ngrok	13
Gambar 1 15 ip ngrok	13
Gambar 1 15 ip ngrok	13
Gambar 1 16 ngrok run	13
Gambar 1 16 ngrok run	13

Gambar 1 17 link dibagikan ngrok.....	14
Gambar 1 17 link dibagikan ngrok.....	14
Gambar 1 18 tampilan app peminjaman.....	15
Gambar 1 18 tampilan app peminjaman.....	15

BAB I

PENDAHULUAN

A. Dasar Teori

Docker adalah sebuah platform open-source yang digunakan untuk mengembangkan, mengirim, dan menjalankan aplikasi dalam sebuah container. Container adalah unit standar perangkat lunak yang mengemas kode dan semua dependensinya sehingga aplikasi dapat berjalan secara konsisten di berbagai lingkungan.

Menurut dokumentasi resmi Docker:

"Docker is an open platform for developing, shipping, and running applications. Docker enables you to separate your applications from your infrastructure so you can deliver software quickly." (Docker Docs, 2024)

“Docker adalah platform terbuka untuk mengembangkan, mendistribusikan, dan menjalankan aplikasi. Docker memungkinkan Anda untuk memisahkan aplikasi Anda dari infrastruktur sehingga Anda dapat mengirimkan perangkat lunak dengan lebih cepat.”

Aplikasi peminjaman barang merupakan sistem informasi berbasis web yang dirancang untuk mempermudah proses pencatatan dan pengelolaan peminjaman barang. Dengan menggunakan bahasa pemrograman Python dan framework Flask, aplikasi ini memungkinkan pengguna untuk melakukan login, registrasi, dan mengajukan peminjaman barang secara daring. Selain itu, Docker digunakan untuk mempermudah proses deployment dengan lingkungan yang terisolasi dan konsisten.

B. Alat dan bahan

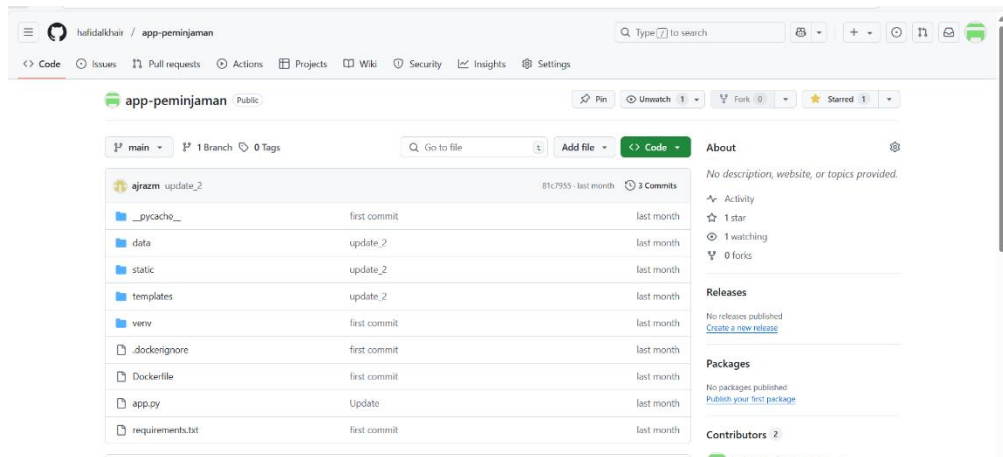
1. Bahasa Pemrograman: Python 3.x
2. Framework: Flask
3. Database: SQLite
4. Docker
5. Editor Kode: Visual Studio Code
6. Browser: Google Chrome atau Mozilla Firefox
7. Sistem Operasi: Windows/Linux

BAB II

LANGKAH PRAKTIKUM

A. Clone Program

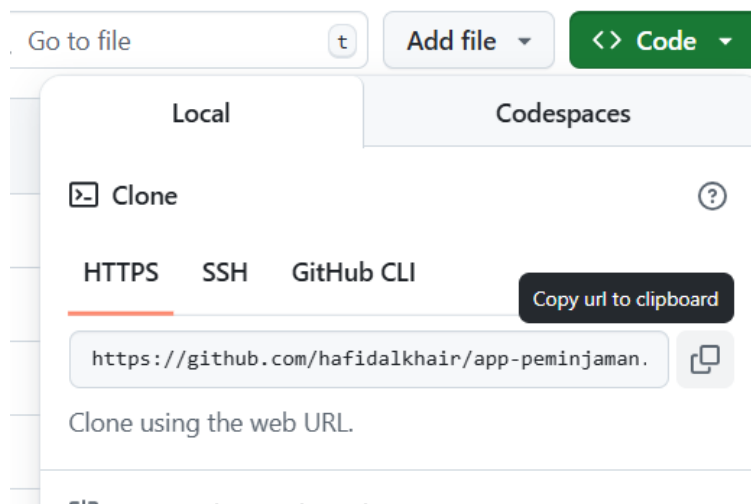
1. Buka github <https://github.com/hafidalkhair/app-peminjaman>



Gambar 1 1 tampilan github

Gambar 1 2 tampilan github

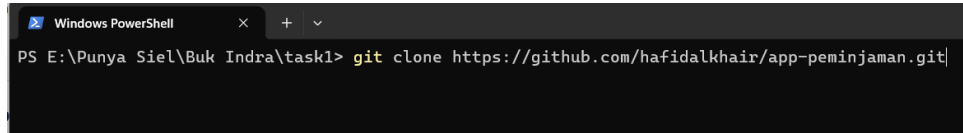
2. Salin github



Gambar 1 3 copy url github

Gambar 1 4 copy url github

3. Buka cmd

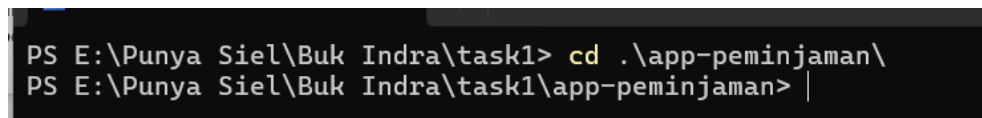


```
Windows PowerShell
PS E:\Punya Siel\Buk Indra\task1> git clone https://github.com/hafidalkhair/app-peminjaman.git
```

Gambar 1 5 git clone git hub di terminal

Gambar 1 6 git clone git hub di terminal

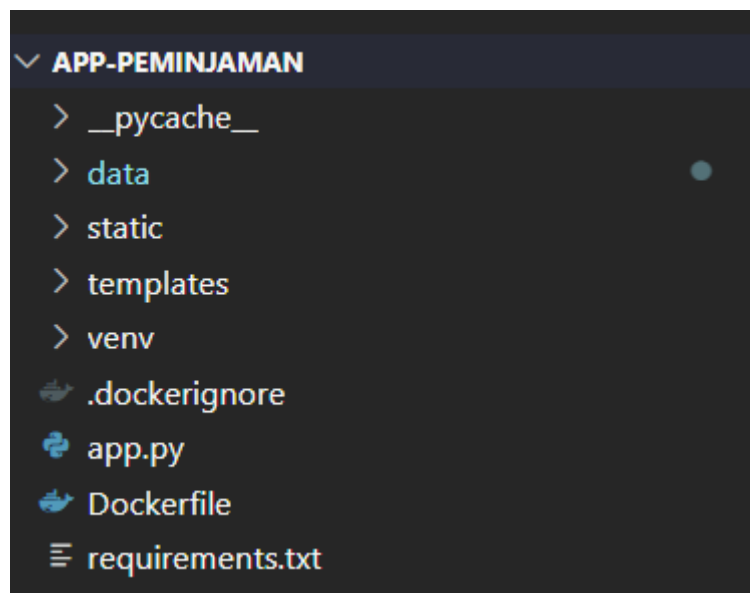
4. cd .\app-peminjaman\



```
PS E:\Punya Siel\Buk Indra\task1> cd .\app-peminjaman\
PS E:\Punya Siel\Buk Indra\task1\app-peminjaman> |
```

Gambar 1 7 masuk direktori app-peminjaman

B. Folder proyek app peminjaman.

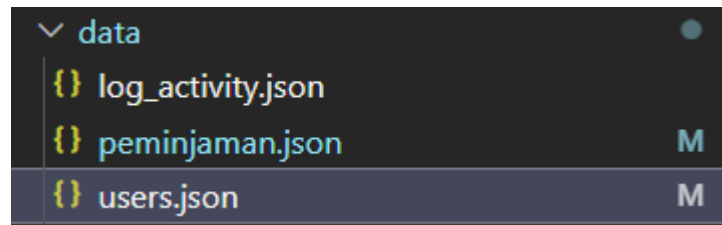


Gambar 1 8 sturktur folder

Gambar 1 9 sturktur folder

C. Membuat database

Untuk database nya itu sudah dibuat otomatis oleh app.py nya sendiri dan akan disimpan dalam data



Gambar 1 10 databases

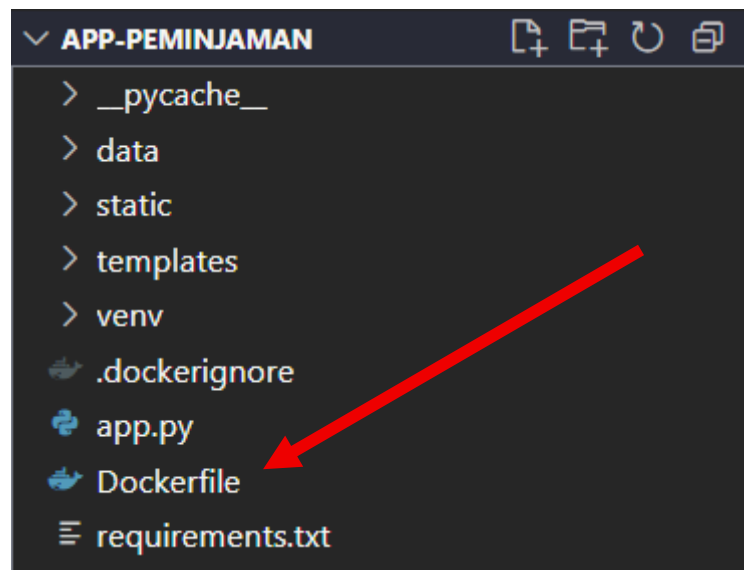
Gambar 1 11 databases

E. Membangun docker dalam program app peminjaman

Pada kasus ini, kami menggunakan situs web peminjaman yang telah kami buat bersama. Untuk memasukkan program ke dalam container Docker, ikuti langkah-langkah berikut:

1. Buat Dockerfile

Buat dockerfile didalam program web peminjaman



Gambar 1 12 Membuat Dockerfile didalam Program app-peminjaman

Gambar 1 13 Membuat Dockerfile didalam Program app-peminjaman

2. Membuat requirements.txt

Flask

Flask-Login

Werkzeug

3. Kode program yang terdapat pada file Dockerfile.

```
# Gunakan image dasar Python
FROM python:3.9-slim-buster

# Set working directory di dalam container
WORKDIR /app

# Salin requirements.txt dan instal dependensi
COPY requirements.txt .
RUN pip install --no-cache-dir -r requirements.txt

# Salin seluruh kode aplikasi
COPY . .

# Set variabel lingkungan
ENV FLASK_APP=app.py
ENV FLASK_ENV=production

# Port yang diekspos
EXPOSE 5000

# Perintah untuk menjalankan aplikasi
CMD ["flask", "run", "--host=0.0.0.0"]
```

4. Penjelasan kode program

a. *FROM python:3.9-slim-buster*

Menggunakan image dasar Python 3.9 yang ringan (slim) sebagai fondasi untuk aplikasi.

b. WORKDIR /app

Menentukan direktori kerja di dalam container. Semua perintah selanjutnya akan dijalankan di dalam direktori ini.

c. COPY requirements.txt .

d. RUN pip install --no-cache-dir -r requirements.txt

Menyalin file requirements.txt ke dalam container dan menginstal semua dependensi yang diperlukan untuk aplikasi Flask.

e. COPY . .

Menyalin seluruh kode aplikasi dari direktori lokal ke dalam container.

f. ENV FLASK_APP=app.py

g. ENV FLASK_ENV=production

Mengatur variabel lingkungan yang diperlukan oleh Flask. FLASK_APP menunjukkan file utama aplikasi, sedangkan FLASK_ENV diatur ke production untuk mode produksi.

h. EXPOSE 5000

Mengekspos port 5000, yang merupakan port default untuk aplikasi Flask.

i. CMD ["flask", "run", "--host=0.0.0.0"]

Perintah ini menjalankan aplikasi Flask ketika container dimulai, membuatnya tersedia untuk menerima permintaan dari luar.

5. Build container

a. docker ps

Untuk melihat daftar container Docker yang sedang berjalan.

```
PS E:\AJRA\S4\Komputasi Awan\App-Peminjaman\app-peminjaman> docker ps
CONTAINER ID   IMAGE                                COMMAND                  CREATED        STATUS        PORTS        NAMES
```

Gambar 1 14 tampilan docker container yang kosong

Gambar 1 15 tampilan docker container yang kosong

b. docker build -t app-peminjaman:1.0 .

```
[+] Building 12.4s (11/11) FINISHED                                docker:desktop-linux
-> [internal] load build definition from Dockerfile                0.0s
-> => transferring Dockerfile: 521B                                0.0s
-> [internal] load metadata for docker.io/library/python:3.9-slim-buster 3.9s
-> [auth] library/python:pull token for registry-1.docker.io       0.0s
-> [internal] load .dockerignore                                    0.0s
-> => transferring context: 99B                                       0.0s
-> [1/5] FROM docker.io/library/python:3.9-slim-buster@sha256:320a7a4258aba249f458872adcf92ee88dc6abd2d76dc5cf01cac9b53990 0.1s
-> => resolve docker.io/library/python:3.9-slim-buster@sha256:320a7a4258aba249f458872adcf92ee88dc6abd2d76dc5cf01cac9b53990 0.0s
-> => sha256:c84dfe38ddeb39e17d12128107f8354a9083b468a320a7780cd128f11c87 6.82kB / 6.82kB 0.0s
-> => sha256:320a7a4258aba249f458872adcf92ee88dc6abd2d76dc5cf01cac9b53990 988B / 988B 0.0s
-> => sha256:d5cca64dca85c37cf06721e36a93bf4331b0a04bfd3ef2c787386796535907 1.37kB / 1.37kB 0.0s
-> [internal] load build context                                    0.4s
-> => transferring context: 7.34kB 0.3s
-> [2/5] WORKDIR /app                                             0.0s
-> [3/5] COPY requirements.txt . 0.0s
-> [4/5] RUN pip install --no-cache-dir -r requirements.txt      7.8s
-> [5/5] COPY . . 0.0s
-> exporting to image                                             0.1s
-> => exporting layers 0.1s
-> => writing image sha256:2e12f060da377fe391ba2aah8963d3b26dd6689f5d3dacebfbb33b8514a1bd6 0.0s
-> => naming to docker.io/library/app-peminjaman:1.0             0.0s

View build details: docker-desktop://dashboard/build/desktop-linux/p1zdfjkav06dwtazsyurcdn
PS E:\AJRA\S4\Komputasi Awan\App-Peminjaman\app-peminjaman>
```

Gambar 1 16 docker berhasil di buat

Gambar 1 17 docker berhasil di buat

c. docker images

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
app-peminjaman	1.0	2e12f060da37	About a minute ago	134MB
fidk/komputasi	1.0	45672059bd77	4 days ago	137MB

Gambar 1 18 tampilan docker images yang telah dibuat

Gambar 1 19 tampilan docker images yang telah dibuat

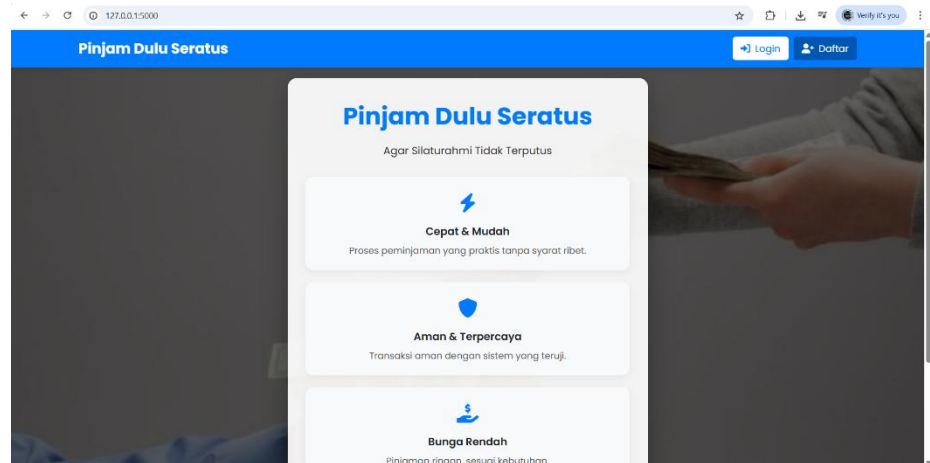
d. docker run -d -p 5000:5000 app-peminjaman:1.0

```
PS E:\AJRA\S4\Komputasi Awan\App-Peminjaman\app-peminjaman> docker run -d -p 5000:5000 app-peminjaman:1.0
1ac30a890549327bdde3a4cda612aea79275cb49f7c69cb445158c3b5807ca8c
```

Gambar 1 20 container id untuk container yang sedang berjalan.

Gambar 1 21 container id untuk container yang sedang berjalan.

e. tampilan app untuk 127.0.0.1:5000



Gambar 1 22 tampilan app setelah docker dijalankan

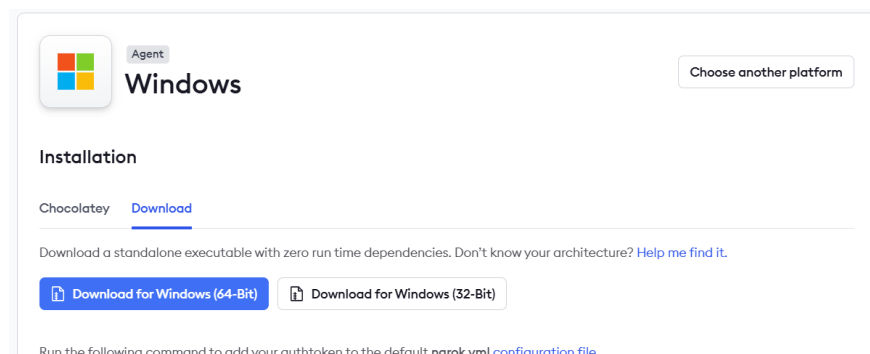
Gambar 1 23 tampilan app setelah docker dijalankan

6. Jalankan docker dan ngrok

Setelah docker dijalankan, install terlebih dahulu ngrok nya

a) Install ngrok windows/linux

Pergi ke web resmi ngrok kemudian login dan ikuti dokumentasi untuk download nya



Gambar 1 24 intall ngrok

Gambar 1 25 intall ngrok

b) Setelah install

Run the following command to add your authtoken to the default `ngrok.yml` [configuration file](#).

```
ngrok config add-authtoken 2xS2t60bCKw1Qt3n9LbZ4U6vkf9_6PVtr27vgZyWnj5n7Dg8Q
```

Gambar 1 26 auth token ngrok

Gambar 1 27 auth token ngrok

c) jalankan ngrok

```
>ngrok http http://localhost:5000|
```

Gambar 1 30 ngrok run

Gambar 1 31 ngrok run

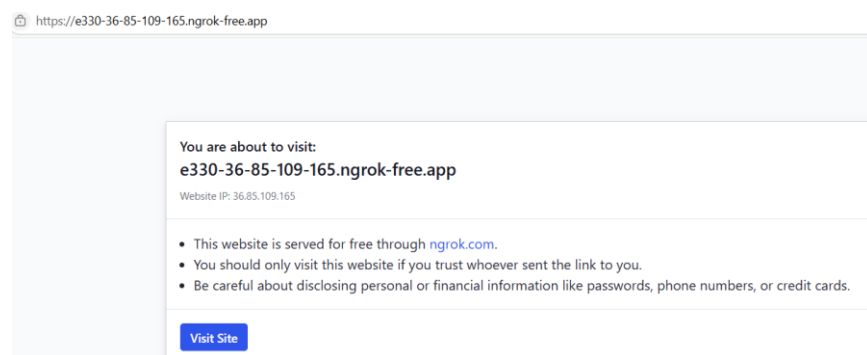
```
ngrok
👉 Load balance anything, anywhere with Endpoint Pools! https://ngrok.com/r/pools
Session Status      online
Account             ajrazm (Plan: Free)
Version             3.22.1
Region              Asia Pacific (ap)
Latency              49ms
Web Interface        http://127.0.0.1:4040
Forwarding            https://e330-36-85-109-165.ngrok-free.app -> http://localhost:5000
Connections          ttl      opn      rt1      rt5      p50      p90
                    0         0         0.00     0.00     0.00     0.00
```

Gambar 1 28 ip ngrok

Gambar 1 29 ip ngrok

untuk portnya harus di sesuaikan dengan docker setelah itu maka akan diberikan ip

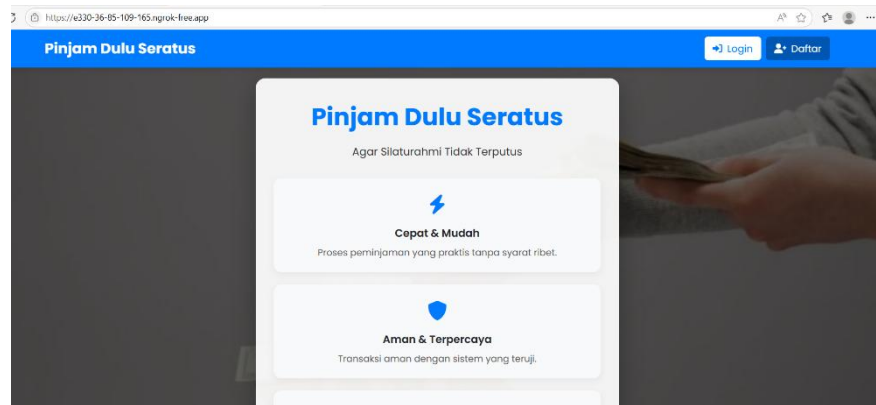
klik ip tersebut nanti akan di arahkan kesebuah link



Gambar 1 32 link dibagikan ngrok

Gambar 1 33 link dibagikan ngrok

visit site kemudian akan tampil app-peminjaman yang telah dibuat



Gambar 1 34 tampilan app peminjaman

Gambar 1 35 tampilan app peminjaman

link tersebut dapat dibagikan dan digunakan juga oleh pemilik link tersebut.

BAB III

PENUTUP

A. Kesimpulan

Praktikum pembuatan aplikasi peminjaman berbasis web dengan Flask dan Docker ini memberikan pemahaman mendalam tentang bagaimana mengembangkan dan mengelola aplikasi dengan lingkungan yang terisolasi serta mudah untuk didistribusikan. Aplikasi ini berhasil dibangun dengan fitur lengkap, seperti login, registrasi, verifikasi biodata, form peminjaman, serta dashboard untuk pengguna dan admin.

Dengan memanfaatkan Docker, aplikasi dapat dijalankan secara konsisten di berbagai sistem operasi tanpa harus mengatur lingkungan secara manual. Praktikum ini juga memberikan pengalaman nyata dalam pengelolaan data menggunakan file JSON dan penggunaan template HTML untuk antarmuka pengguna.

Secara keseluruhan, praktikum ini memperkuat keterampilan teknis dalam pengembangan web dan DevOps dasar, serta menumbuhkan pemahaman tentang arsitektur aplikasi yang modular dan efisien.