

**PENGEMBANGAN APLIKASI REKOMENDASI BEASISWA  
BERBASIS KNOWLEDGE GRAPH DENGAN INTEGRASI  
FASTAPI, STREAMLIT, DAN DOCKER**



Disusun Oleh :

**Ahmad Fahim Nidhom**

**32602200033**

**PROGRAM STUDI TEKNIK INFORMATIKA  
FAKULTAS TEKNOLOGI INDUSTRI  
UNIVERSITAS ISLAM SULTAN AGUNG  
SEMARANG  
2025**

## DAFTAR ISI

<b>DAFTAR ISI .....</b>	<b>ii</b>
<b>DAFTAR GAMBAR .....</b>	<b>iii</b>
<b>DAFTAR TABEL .....</b>	<b>iv</b>
<b>BAB I PENDAHULUAN .....</b>	<b>1</b>
1.1    Latar Belakang .....	1
1.2    Tujuan.....	1
<b>BAB II TINJAUAN PUSTAKA .....</b>	<b>3</b>
2.1    Sistem Rekomendasi dan Knowledge Graph .....	3
2.2    Python dan Pandas .....	3
2.3    Docker .....	3
2.4    Streamit dan FastAPI .....	3
2.5    NetworkX & PyVis .....	3
2.6    TextBlob dan Sentiment Analysis .....	3
<b>BAB III IMPLEMENTASI .....</b>	<b>4</b>
3.1    Struktur Project .....	4
3.2    Konfigurasi Dockerfile.....	4
3.3    DockerHub .....	5
<b>BAB IV HASIL DAN PEMBAHASAN .....</b>	<b>6</b>
4.1    Instalasi Docker.....	6
4.2    Proses Implementasi.....	6
4.3    Tampilan Aplikasi .....	7
4.4    Unggah Aplikasi Ke Dockerhub .....	8
<b>BAB V PENUTUP .....</b>	<b>10</b>
5.1    Kesimpulan .....	10
<b>REFRENSI .....</b>	<b>11</b>
<b>LAMPIRAN .....</b>	<b>12</b>

## DAFTAR GAMBAR

Gambar 3. 1 Struktur Folder Project .....	4
Gambar 3. 2 Konfigurasi Dockerfile.....	5
Gambar 4. 1 Tampilan Docker setelah di install .....	6
Gambar 4. 2 Tampilan Docker version .....	6
Gambar 4. 3 Tampilan Docker build image .....	7
Gambar 4. 4 Docker PS.....	7
Gambar 4. 5 Docker run.....	7
Gambar 4. 6 Tampilan Utama Aplikasi.....	8
Gambar 4. 7 Tampilan DockerHub .....	9

## **DAFTAR TABEL**

# BAB I

## PENDAHULUAN

### 1.1 Latar Belakang

Perkembangan teknologi informasi telah mendorong transformasi dalam sistem pendukung keputusan, termasuk dalam pemberian rekomendasi beasiswa kepada mahasiswa. Banyak mahasiswa mengalami kesulitan dalam menemukan beasiswa yang sesuai karena beragamnya kriteria seperti jenjang studi, lokasi, dan jenis pendanaan. Salah satu pendekatan yang berkembang untuk mengatasi masalah ini adalah penggunaan *knowledge graph*, yang mampu merepresentasikan hubungan kompleks antar entitas dan meningkatkan akurasi sistem rekomendasi (Zhu et al. 2021)

Dalam pengembangan sistem rekomendasi berbasis web, framework seperti *FastAPI* dan *Streamlit* semakin populer karena kemampuannya membangun antarmuka web interaktif dan layanan API secara cepat dan sederhana (Ganaphati and Sharma 2022). Namun, tantangan deployment masih menjadi masalah umum, terutama karena perbedaan konfigurasi sistem. Solusi yang banyak diadopsi adalah penggunaan Docker, yang memungkinkan aplikasi dijalankan dalam container yang terisolasi dan konsisten di berbagai platform (Vinaya Durga 2020). Dengan mengintegrasikan pendekatan *knowledge graph*, API, antarmuka Streamlit, dan container Docker, sistem rekomendasi beasiswa menjadi lebih fleksibel, skalabel, dan siap diterapkan secara luas.

### 1.2 Tujuan

Adapun tujuan dari makalah ini :

1. Menjelaskan proses pengembangan sistem rekomendasi beasiswa berbasis *knowledge graph* menggunakan data akademik dan non-akademik mahasiswa.

2. Menyajikan tahapan implementasi aplikasi dengan framework **FastAPI** dan **Streamlit** sebagai backend dan antarmuka pengguna.
3. Menjelaskan proses containerisasi aplikasi menggunakan **Docker** untuk memastikan lingkungan kerja yang konsisten.
4. Menyampaikan langkah-langkah publikasi aplikasi ke **DockerHub** sebagai repositori cloud yang dapat diakses publik.

## **BAB II**

### **TINJAUAN PUSTAKA**

#### **2.1 Sistem Rekomendasi dan Knowledge Graph**

Sistem rekomendasi menggunakan *knowledge graph* mampu memodelkan hubungan semantik antar entitas, seperti mahasiswa, beasiswa, dan atribut lainnya, sehingga menghasilkan rekomendasi yang lebih kontekstual dan relevan (Zhu et al. 2021).

#### **2.2 Python dan Pandas**

Python dengan pustaka Pandas sangat efektif untuk pembersihan data, analisis, dan manipulasi dataset besar selama proses rekomendasi (Kabir 2024).

#### **2.3 Docker**

Docker adalah platform containerisasi yang membungkus aplikasi beserta seluruh dependensinya dalam satu paket yang konsisten dan portabel, memungkinkan deployment yang andal di berbagai lingkungan (Vinaya Durga 2020).

#### **2.4 Streamlit dan FastAPI**

Streamlit memungkinkan pembuatan antarmuka pengguna berbasis web secara interaktif, sementara FastAPI menyediakan layanan API yang cepat dan mudah digunakan untuk mendukung aplikasi berbasis data (Ganaphati and Sharma 2022).

#### **2.5 NetworkX & PyVis**

NetworkX digunakan untuk membangun struktur graf, dan PyVis memungkinkan visualisasi graf interaktif di web, memberikan pengalaman eksplorasi data yang lebih informatif dan visual (Perrone, Unpingco, and Lu 2020).

#### **2.6 TextBlob dan Sentiment Analysis**

TextBlob adalah pustaka Python yang sederhana namun efektif untuk analisis sentimen, digunakan untuk mengevaluasi opini atau nada dalam teks seperti judul beasiswa (Suanpang, Jamjuntr, and Kaewyong 2021).

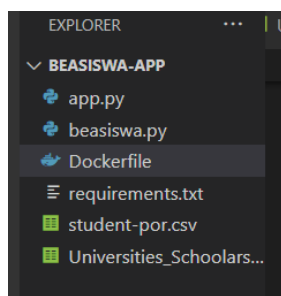
## BAB III

### IMPLEMENTASI

#### 3.1 Struktur Project

Project ini merupakan sistem rekomendasi beasiswa berbasis *knowledge graph* yang dibangun menggunakan bahasa pemrograman Python. Aplikasi terdiri dari dua komponen utama, yaitu API menggunakan **FastAPI** dan antarmuka pengguna menggunakan **Streamlit**. Seluruh proses data preprocessing, pembuatan graf, dan deployment dilakukan dalam satu folder project bernama `beasiswa-app`, yang dikemas ke dalam container menggunakan **Docker**. Struktur file utama meliputi:

1. `beasiswa.py` (file utama preprocessing & FastAPI)
2. `app.py` (streamlit interface)
3. `Dockerfile` (untuk membuat image)
4. `requirements.txt` (daftar dependensi)
5. file CSV data : `Universities_Scholarships_All_Around_the_World.csv` dan `student-por.csv`



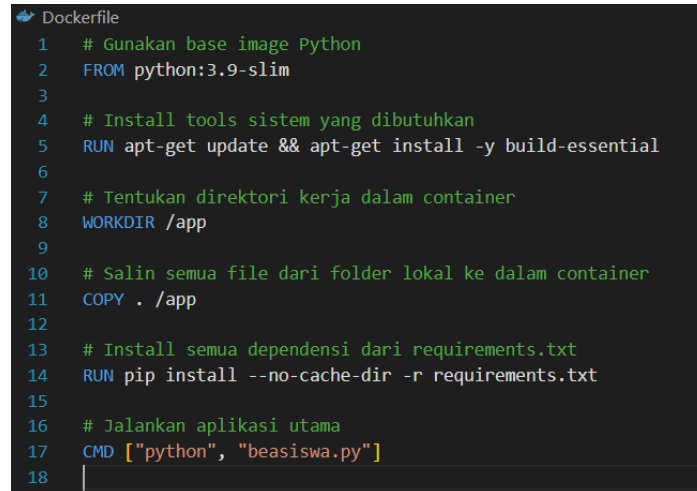
Gambar 3. 1 Struktur Folder Project

#### 3.2 Konfigurasi Dockerfile

File Dockerfile berfungsi untuk mendefinisikan proses pembuatan image Docker dari aplikasi. Dalam Dockerfile ini, digunakan **python:3.10** sebagai *base image*, kemudian semua dependensi di-*install* melalui `requirements.txt`, dan file Python utama dijalankan menggunakan perintah CMD. Dengan



demikian, proses setup dan deployment dapat dilakukan secara otomatis dan konsisten, baik di mesin lokal maupun server.

A screenshot of a code editor showing a Dockerfile. The file is titled 'Dockerfile' with a small icon. The code is as follows:

```
1 # Gunakan base image Python
2 FROM python:3.9-slim
3
4 # Install tools sistem yang dibutuhkan
5 RUN apt-get update && apt-get install -y build-essential
6
7 # Tentukan direktori kerja dalam container
8 WORKDIR /app
9
10 # Salin semua file dari folder lokal ke dalam container
11 COPY . /app
12
13 # Install semua dependensi dari requirements.txt
14 RUN pip install --no-cache-dir -r requirements.txt
15
16 # Jalankan aplikasi utama
17 CMD ["python", "beasiswa.py"]
18
```

Gambar 3. 2 Konfigurasi *Dockerfile*

### 3.3 DockerHub

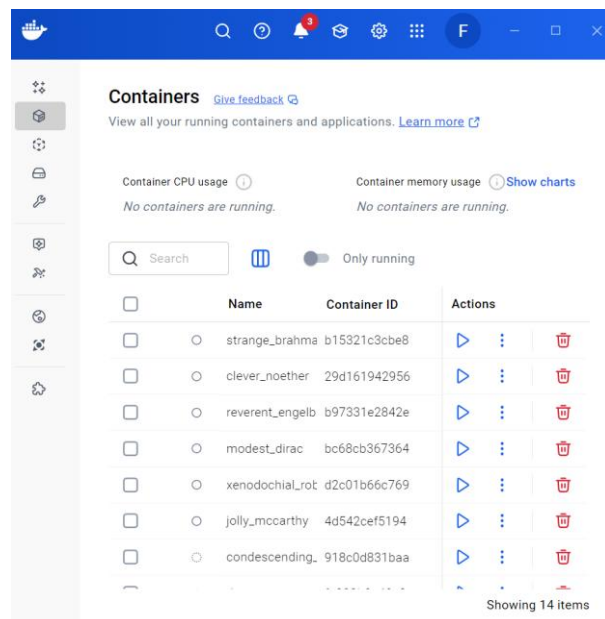
Setelah proses containerisasi berhasil, image kemudian ditandai (*tagging*) dan di-*push* ke akun Docker Hub pengguna. Proses ini memungkinkan image dapat diakses dari perangkat lain dengan perintah `docker pull`.

## BAB IV

### HASIL DAN PEMBAHASAN

#### 4.1 Instalasi Docker

Docker diinstal pada sistem operasi Windows menggunakan installer resmi dari <https://www.docker.com/products/docker-desktop>.



Gambar 4. 1 Tampilan Docker setelah di install

Setelah instalasi, perintah `docker -v` digunakan untuk memastikan bahwa Docker telah terpasang dan berjalan dengan baik.

```
PS C:\Users\Fahim Ahmad\beasiswa-app> docker -v
Docker version 28.3.0, build 38b7060
```

Gambar 4. 2 Tampilan Docker version

#### 4.2 Proses Implementasi

Setelah konfigurasi docker selesai, jalankan :

```
docker build -t beasiswa-app
```

```

PS C:\Users\Fahim Ahmad\beasiswa-app>
[+] Building 169.6s (11/11) FINISHED
=> [internal] load build definition from Dockerfile
=> => transferring dockerfile: 486B
=> [internal] load metadata for docker.io/library/python:3.9-slim
=> [auth] library/python:pull token for registry-1.docker.io
=> [internal] load .dockerignore
=> => transferring context: 2B
=> [1/5] FROM docker.io/library/python:3.9-slim@sha256:c2a0feb07dedbf91498883c2f8e1e5:
=> => resolve docker.io/library/python:3.9-slim@sha256:c2a0feb07dedbf91498883c2f8e1e5:
=> [internal] load build context
=> => transferring context: 538.23kB
=> CACHED [2/5] RUN apt-get update && apt-get install -y build-essential
=> CACHED [3/5] WORKDIR /app
=> [4/5] COPY . /app
=> [5/5] RUN pip install --no-cache-dir -r requirements.txt

```

Gambar 4. 3 Tampilan Docker build image

Perintah `docker build -t beasiswa-app .` digunakan untuk membangun image Docker dari Dockerfile yang ada di direktori saat ini. Opsi `-t beasiswa-app` memberi nama (tag) pada image yang dihasilkan. Titik (.) menunjukkan lokasi build context. Perintah ini dijalankan saat pertama kali membuat image atau setelah ada perubahan pada file proyek. Kemudian cek status container :

`docker ps`

Hasilnya menampilkan daftar container aktif lengkap dengan nama, status, dan port mapping.

```

PS C:\Users\Fahim Ahmad> cd beasiswa-app
PS C:\Users\Fahim Ahmad\beasiswa-app> docker ps

```

CONTAINER ID	IMAGE	COMMAND	CREATED NAMES	STATUS
b15321c3cbe8	beasiswa-app	"python beasiswa.py"	20 hours ago	Up 38 s
00/tcp, 0.0.0.0:8501->8501/tcp, [::]:8501->8501/tcp			strange_brahmagupta	

Gambar 4. 4 Docker PS

### 4.3 Tampilan Aplikasi

Setelah konfigurasi selesai, kemudian jalankan project nya dengan mengakses :

`docker run -p 8501:8501 beasiswa-app`

```

PS C:\Users\Fahim Ahmad\beasiswa-app> docker run -p 8501:8501 beasiswa-app
INFO:      Started server process [1]
INFO:      Waiting for application startup.
INFO:      Application startup complete.
INFO:      Uvicorn running on http://0.0.0.0:8000 (Press CTRL+C to quit)

Collecting usage statistics. To deactivate, set browser.gatherUsageStats to false.

You can now view your Streamlit app in your browser.

Local URL: http://localhost:8501
Network URL: http://172.17.0.2:8501
External URL: http://114.10.44.227:8501

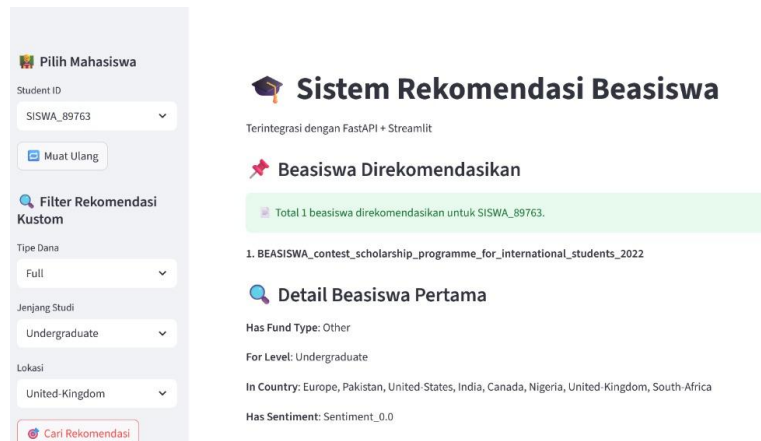
```

Gambar 4. 5 Docker run

setelah container running kemudian akses alamat kemudian copy ke browser :

<http://localhost:8501>

Kemudian akan muncul tampilan halaman utama aplikasi rekomendasi, kalau website sudah muncul maka konfigurasi docker nya sudah benar.



Gambar 4. 6 Tampilan Utama Aplikasi

#### 4.4 Unggah Aplikasi Ke Dockerhub

Sebelum unggah aplikasi ke dockerhub terlebih dahulu masuk ke akun docker.

```
docker login
```

kemudian tag pada image lokal beasiswa-app agar sesuai dengan format penamaan DockerHub (username/nama-repo:tag). Ini penting agar image bisa dikenali saat proses push.

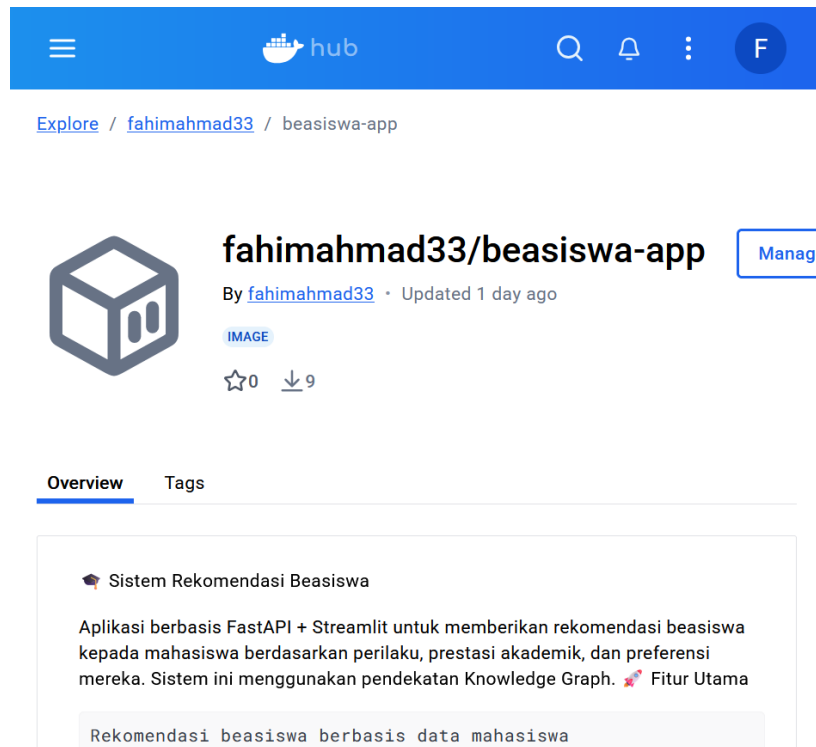
```
docker tag beasiswa-app fahimahmad33/beasiswa-app:latest
```

kemudian unggah image yang sudah ditandai tadi ke repository DockerHub.

```
docker run -p 8501:8501 beasiswa-app
```

setelah di unggah link image bisa di akses melalui :

<https://hub.docker.com/r/fahimahmad33/beasiswa-app>



Gambar 4. 7 Tampilan DockerHub

Melalui tautan ini, siapa pun dapat melihat dan mengunduh image Docker bernama beasiswa-app yang telah dipublikasikan . Image tersebut dapat diunduh menggunakan perintah

```
docker pull fahimahmad33/beasiswa-app:latest
```

Dijalankan secara lokal untuk mengakses aplikasi sistem rekomendasi beasiswa berbasis FastAPI dan Streamlit.

## **BAB V**

### **PENUTUP**

#### **5.1 Kesimpulan**

Pengembangan sistem rekomendasi beasiswa berbasis *knowledge graph* memberikan pendekatan baru yang lebih kontekstual dan terstruktur dalam menyajikan rekomendasi kepada mahasiswa. Dengan memanfaatkan kombinasi algoritma preprocessing, analisis sentimen, dan relasi antar entitas melalui graf, sistem ini mampu menyajikan hasil rekomendasi yang lebih informatif dan fleksibel berdasarkan karakteristik dan preferensi pengguna.

Penggunaan Docker dalam project ini memberikan kontribusi besar dalam hal efisiensi deployment. Dengan mengemas seluruh dependensi Python dan file sistem ke dalam container, proses instalasi aplikasi dapat dilakukan secara konsisten di berbagai platform tanpa khawatir terhadap perbedaan lingkungan pengembangan. Penggunaan Dockerfile juga menyederhanakan distribusi aplikasi, serta memungkinkan integrasi langsung ke layanan cloud atau CI/CD pipeline jika dibutuhkan di masa mendatang.

Secara keseluruhan, integrasi antara Python, FastAPI, Streamlit, dan Docker dalam sistem rekomendasi ini menunjukkan bagaimana teknologi kontainerisasi dan pemrograman modern dapat digunakan untuk membangun aplikasi cerdas yang ringan, modular, dan mudah diakses. Project ini juga dapat dikembangkan lebih lanjut, misalnya dengan menambahkan fitur autentikasi pengguna, penyimpanan berbasis database, serta antarmuka yang lebih dinamis.

## REFRENSI

- Ganaphati, Manjunatha, and K V Sharma. 2022. "Effect of Graphene Reinforcement on Damping Properties of Epoxy/Glass Hybrid Composites." *International Journal for Research in Applied Science and Engineering Technology* 10(6): 1868–74.
- Kabir, Mohammad Anowarul. 2024. "Python For Data Analytics: A Systematic Literature Review Of Tools, Techniques, And Applications." *Academic Journal on Science, Technology, Engineering & Mathematics Education* 4(04): 134–54.
- Perrone, Giancarlo, Jose Unpingco, and Haw-minn Lu. 2020. "Network Visualizations with Pyvis and VisJS." *Proceedings of the 19th Python in Science Conference (Scipy)*: 58–62.
- Suanpang, Pannee, Pitchaya Jamjuntr, and Phuripoj Kaewyong. 2021. "Sentiment Analysis With a Textblob Package." *Journal of Management Information and Decision Sciences* 24(6): 1–9.
- Vinaya Durga, Ganapathi Sharma. 2020. "Analysis of K-Mean Algorithm." 6(1): 133–36.
- Zhu, Feida et al. 2021. *KDD ' 21 Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*.

## **LAMPIRAN**

Link gituhub :

<https://github.com/fahimahmad21/beasiswa-app-project-cloud-computing->