LAPORAN IMPLEMENTASI API SEDERHANA MENGGUNAKAN FASTAPI

Laporan Dirancang Sebagai Tugas Mata Kuliah Teknologi *Cloud Computing*

Oleh :

Rizal Hanifa Pratama 123210114

PROGRAM STUDI INFORMATIKA

JURUSAN INFORMATIKA

FAKULTAS TEKNIK INDUSTRI

UNIVERSITAS PEMBANGUNAN NASIONAL "VETERAN"

YOGYAKARTA

2024

BAB II
PENJELASAN

2.1 Penjelasan Struktur dan Fungsi Endpoint

1. Endpoint Root (/)

   Metode HTTP yang digunakan adalah GET, yang berfungsi untuk mengembalikan daftar endpoint yang tersedia dengan informasi singkat tentang masing-masing endpoint. Fungsi ini memberikan daftar lengkap dari endpoint-endpoint yang dapat diakses oleh pengguna, masing-masing dilengkapi dengan `id`, `nama`, dan `endpoint`. Output dari fungsi ini adalah sebuah daftar yang berisi informasi mengenai berbagai endpoint, termasuk identifikasi unik (`id`), nama endpoint (`nama`), dan URL endpoint (`endpoint`). Hal ini memungkinkan pengguna untuk dengan mudah mengetahui dan mengakses berbagai layanan yang disediakan oleh API.

2. Endpoint Hero (/hero)

   Metode HTTP yang digunakan adalah GET, dengan fungsi untuk mengembalikan daftar hero yang berisi informasi singkat, termasuk `id`, `hero`, dan `ras`. Output dari fungsi ini adalah daftar hero yang mencakup `id`, `hero`, dan `ras`, memberikan gambaran umum tentang setiap hero yang tersedia. Selain itu, terdapat endpoint tambahan, yaitu Endpoint Hero Detail (`/hero/{hero_id}`), yang memungkinkan pengguna untuk mendapatkan detail lengkap dari hero tertentu berdasarkan `id` yang diberikan. Endpoint ini berguna untuk memperoleh informasi yang lebih mendalam tentang setiap hero.

3. Metode HTTP: GET

   Fungsi ini bertujuan untuk mengembalikan detail lengkap dari seorang hero berdasarkan `hero_id` yang diberikan. Output dari fungsi ini mencakup informasi rinci mengenai hero, termasuk `id`, `hero`, `ras`, `senjata`, `umur`, dan `quotes`, yang memberikan gambaran menyeluruh tentang karakter tersebut. Selain itu, terdapat penanganan kesalahan yang dilakukan dengan mengembalikan respons HTTP 404 jika hero dengan `hero_id` yang diminta tidak ditemukan. Hal ini memastikan bahwa pengguna menerima tanggapan yang sesuai bahkan jika mereka mencoba mengakses hero yang tidak tersedia dalam database.

4. Endpoint Role (/role)

   Metode HTTP yang digunakan adalah GET, dimana fungsi ini bertujuan untuk mengembalikan daftar role yang tersedia. Output dari fungsi ini adalah daftar role yang mencakup informasi mengenai `id` dan `Role`. Setiap entri dalam daftar tersebut memberikan informasi singkat tentang role yang tersedia, seperti identifikasi unik (`id`) dan nama role (`Role`). Hal ini memudahkan pengguna untuk memahami berbagai peran yang ada dalam sistem dan memungkinkan mereka untuk memilih peran yang sesuai dengan kebutuhan mereka.

5. Endpoint Type (/type)

   Metode HTTP yang digunakan adalah GET, dengan tujuan untuk mengembalikan daftar type yang tersedia. Fungsi ini bertujuan untuk memberikan pengguna informasi singkat tentang berbagai jenis yang tersedia dalam sistem. Output dari fungsi ini adalah daftar type yang mencakup `id` dan `Type`, yang memberikan informasi tentang jenis-jenis yang dapat digunakan. Setiap entri dalam daftar ini memberikan identifikasi unik (`id`) dan nama jenis (`Type`), memungkinkan pengguna untuk dengan mudah memahami berbagai jenis yang ada dalam sistem dan memilih jenis yang sesuai dengan kebutuhan mereka.

6. Endpoint Lane (/lane)

   Metode HTTP yang digunakan adalah GET, dengan tujuan untuk mengembalikan daftar lane yang tersedia. Fungsi ini bertujuan untuk memberikan pengguna informasi singkat tentang berbagai jalur yang dapat digunakan dalam sistem. Output dari fungsi ini adalah daftar lane yang mencakup `id` dan `Lane`, memberikan informasi tentang berbagai jalur yang tersedia. Setiap entri dalam daftar ini menyediakan identifikasi unik (`id`) dan nama jalur (`Lane`), memungkinkan pengguna untuk dengan mudah memahami berbagai jalur yang ada dalam sistem dan memilih jalur yang sesuai dengan kebutuhan mereka.

2.2 Kode Lengkap dengan Komentar/Dokumentasi

```
from fastapi import FastAPI, HTTPException

# Membuat instance dari FastAPI
app = FastAPI()

# Endpoint root
@app.get('/')
```

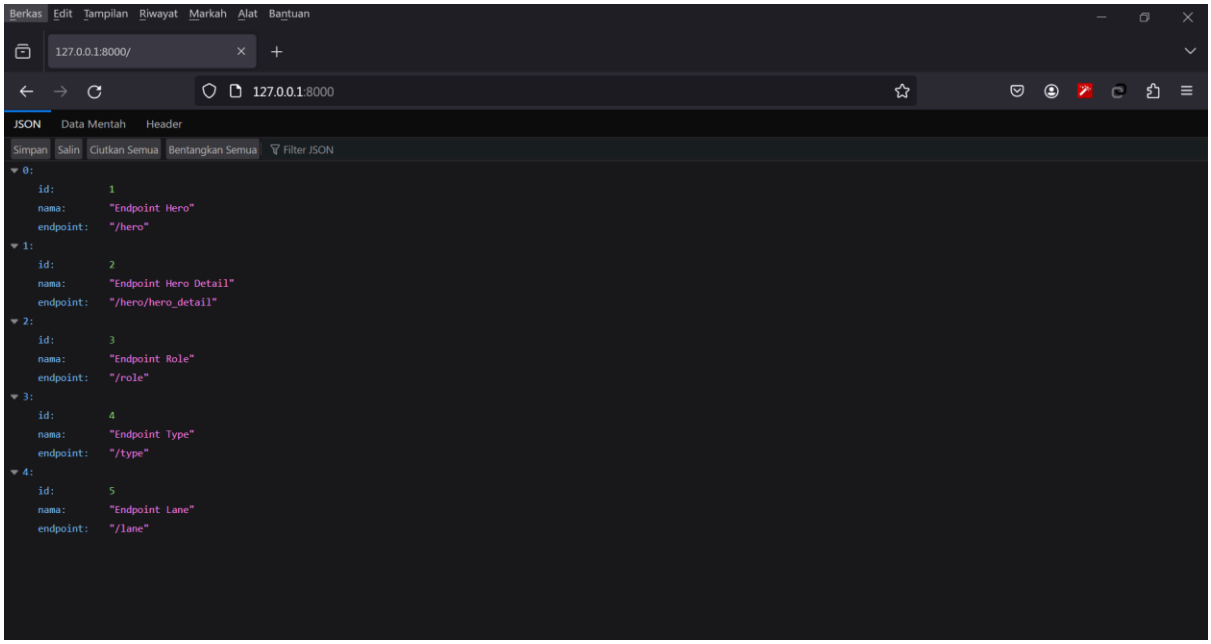
```
async def root():
    """
    Mengembalikan daftar endpoint yang tersedia dengan informasi
singkat.
    """
    return [
        {"id": 1, "nama": "Endpoint Hero", "endpoint": "/hero"},
        {"id": 2, "nama": "Endpoint Hero Detail", "endpoint":
"/hero/hero_detail"},
        {"id": 3, "nama": "Endpoint Role", "endpoint": "/role"},
        {"id": 4, "nama": "Endpoint Type", "endpoint": "/type"},
        {"id": 5, "nama": "Endpoint Lane", "endpoint": "/lane"},
    ]

# Data dummy untuk hero
heros = [
    {
        "id": 1,
        "hero": "Alucard",
        "ras": "Manusia",
        "senjata": "Pedang",
        "umur": 28,
        "quotes": "Para iblis akan bersimbah darah, cahaya hanya
milik orang yang benar."
    },
    {
        "id": 2,
        "hero": "Natalia",
        "ras": "Manusia",
        "senjata": "Cakar",
        "umur": 18,
        "quotes": "Orang bijak sejati tak akan memberitahumu
keberanannya."
    },
    {
        "id": 3,
        "hero": "Argus",
```

```
        "ras": "Peri",
        "senjata": "Pedang",
        "umur": 40,
        "quotes": "Kekuatan adalah keabadian."
    },
]

# Endpoint untuk mendapatkan data hero
@app.get('/hero')
async def hero_data():
    """
    Mengembalikan daftar hero dengan informasi singkat (id, hero,
ras).
    """
    # Menyederhanakan data hero agar hanya mengandung id, hero, dan
ras
    simplified_heros = [
        {"id": hero["id"], "hero": hero["hero"], "ras": hero["ras"]}
        for hero in heros
    ]
    return {
        "Message": "Success fetch heros data",
        "Data": simplified_heros
    }

# Endpoint untuk mendapatkan detail hero berdasarkan id
@app.get('/hero/{hero_id}')
async def hero_detail(hero_id: int):
    """
    Mengembalikan detail hero berdasarkan hero_id.
    Jika hero tidak ditemukan, mengembalikan HTTP 404.
    """
    # Mencari hero berdasarkan id
    for hero in heros:
        if hero["id"] == hero_id:
            return {
                "Message": "Success fetch hero detail",
```

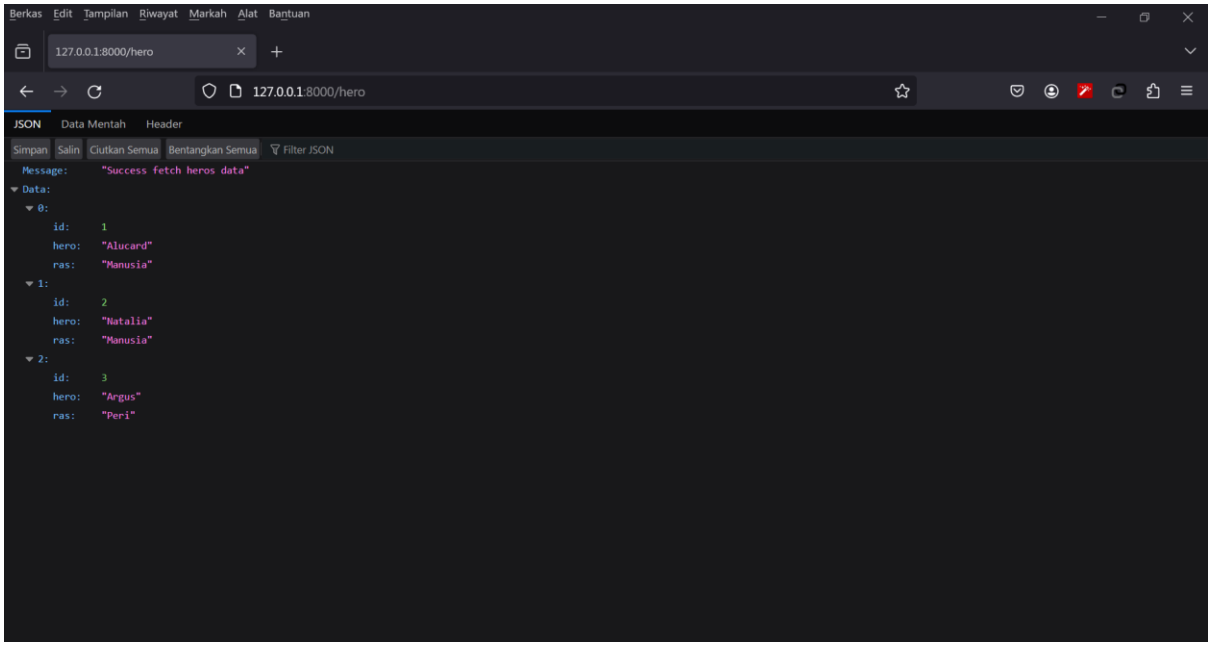
```
                "Data": hero
            }
    # Jika hero tidak ditemukan, mengembalikan HTTP 404
    raise HTTPException(status_code=404, detail="Hero not found")

# Endpoint untuk mendapatkan data Role
@app.get('/role')
async def role_data():
    """
    Mengembalikan daftar role yang tersedia.
    """
    # Data dummy role
    return {
        "Message": "Success fetch role data",
        "Data": [
            {"id": 1, "Role": "Fighter"},
            {"id": 2, "Role": "Assasin"},
            {"id": 3, "Role": "Tank"},
        ]
    }

# Endpoint untuk mendapatkan data type
@app.get('/type')
async def type_data():
    """
    Mengembalikan daftar type yang tersedia.
    """
    # Data dummy type
    return {
        "Message": "Success fetch type data",
        "Data": [
            {"id": 1, "Type": "Magic"},
            {"id": 2, "Type": "Deffend"},
            {"id": 3, "Type": "Phsycal"},
        ]
    }
```

```
# Endpoint untuk mendapatkan data lane
@app.get('/lane')
async def lane_data():
    """
    Mengembalikan daftar lane yang tersedia.
    """
    # Data dummy lane
    return {
        "Message": "Success fetch lane data",
        "Data": [
            {"id": 1, "Lane": "EXP Lane"},
            {"id": 2, "Lane": "Gold lane"},
            {"id": 3, "Lane": "Mid Lane"},
        ]
    }
```

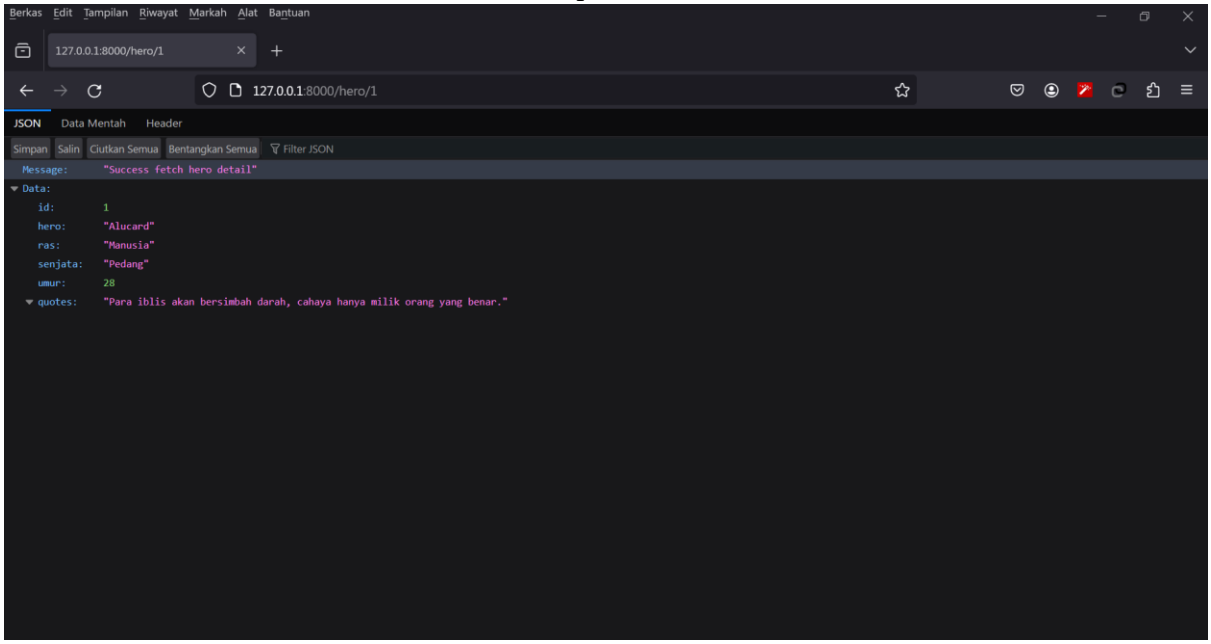
2.3 Hasil Running Program



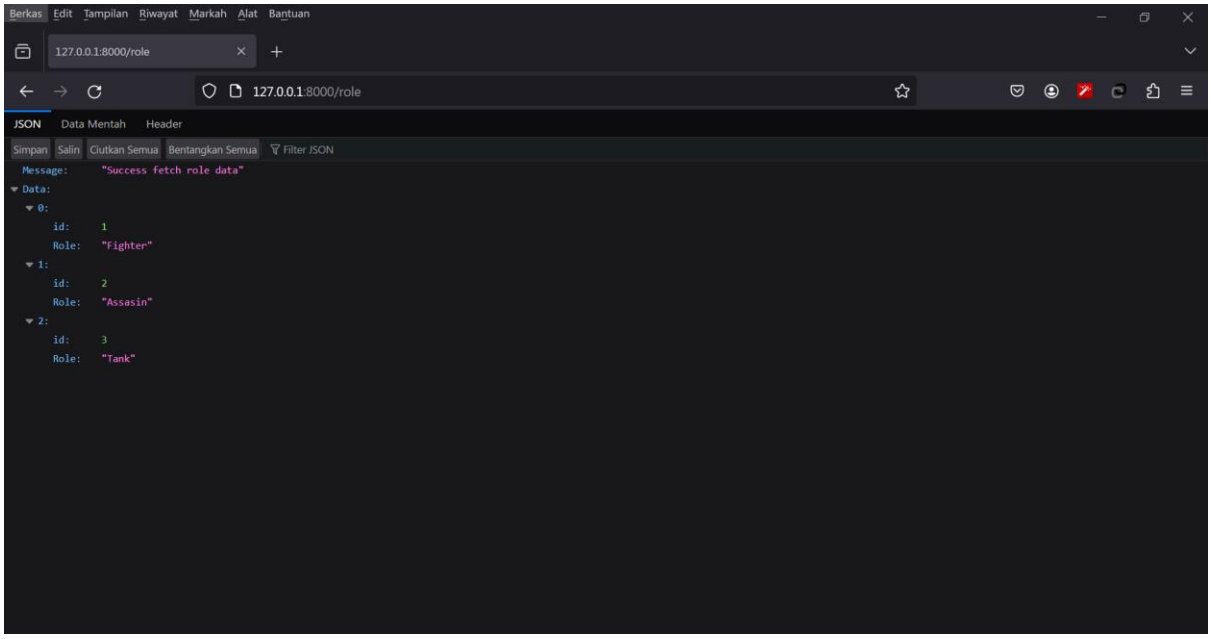
Gambar 1 *running* di *localhost* 8000



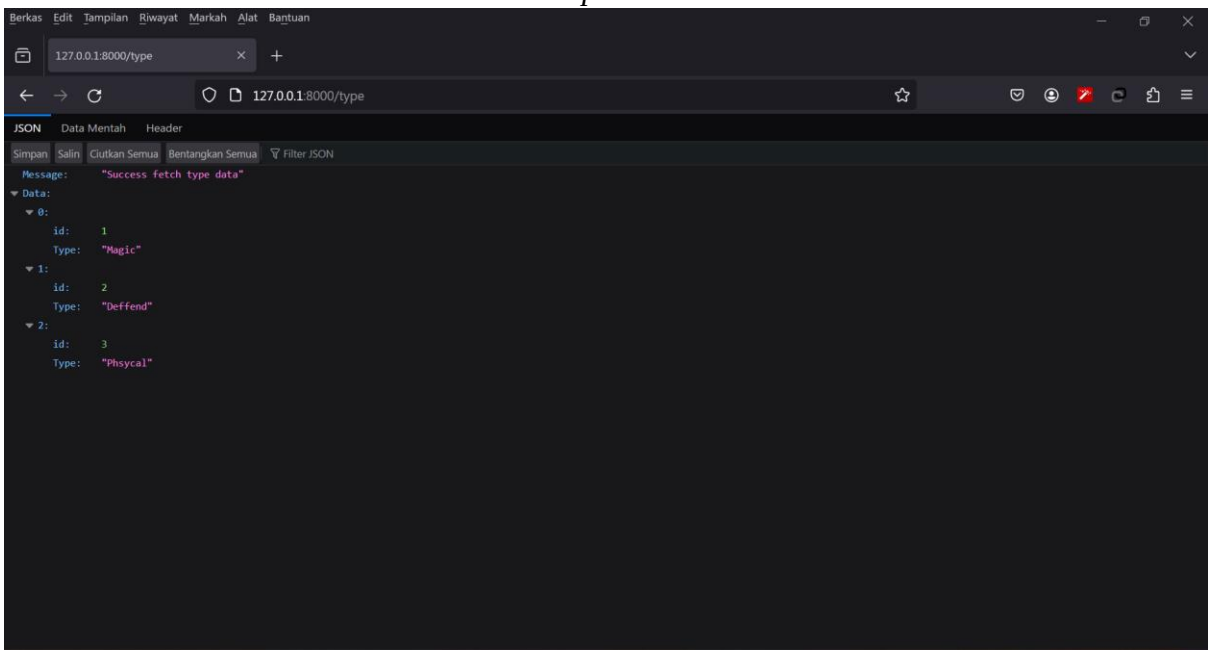
Gambar 2 *endpoint* /hero



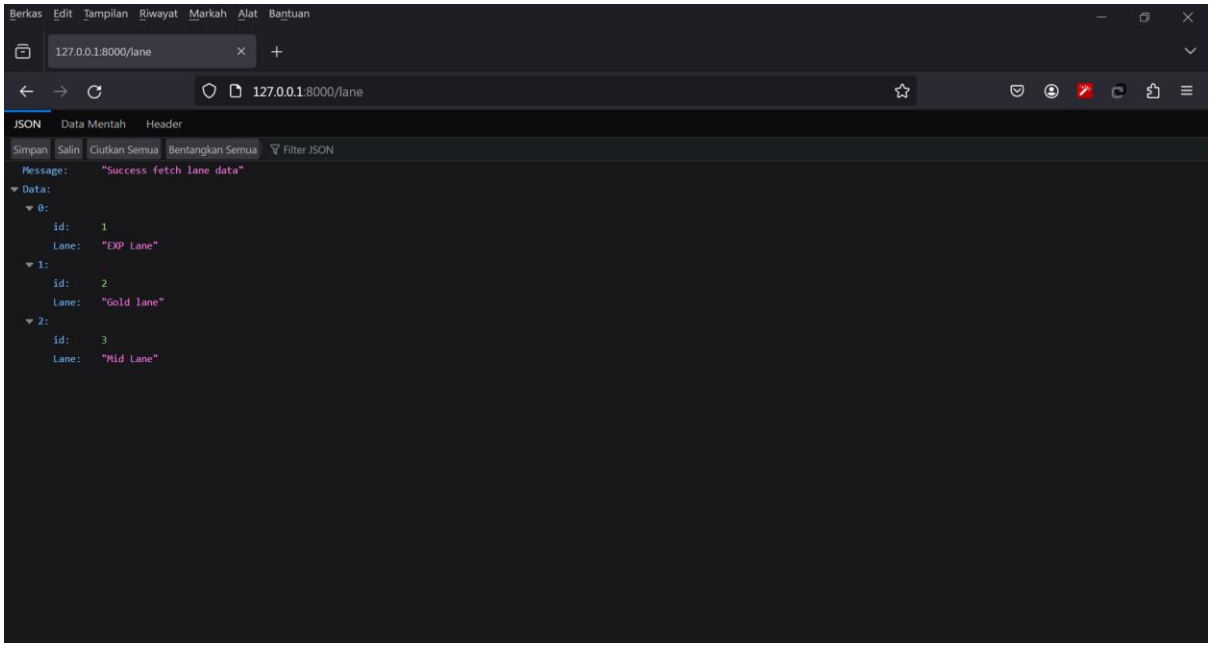
Gambar 3 *endpoint* /hero/1 untuk detail hero id 1



Gambar 4 *endpoint* /role



Gambar 5 *endpoint* /type



Gambar 6 *endpoint* /lane