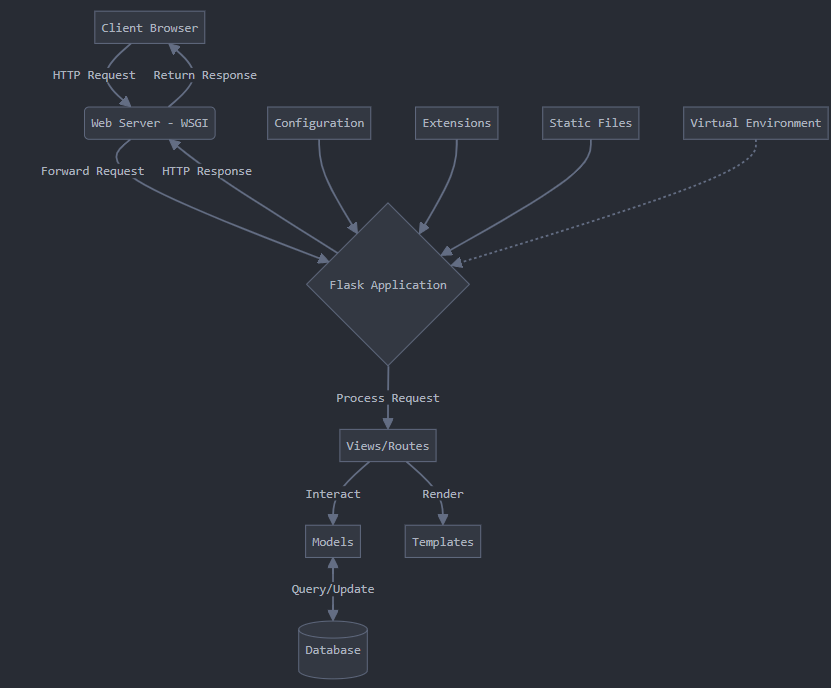
**Menjelaskan Dan Meggambarkan Arsitektur Project Flask Dalam Suatu Environment  
( Tugas 1.5 No,2 )**Lembar ini Disusun untuk Memenuhi Tugas Mata Kuliah Pemrograman Web Praktik II  
**Dosen pengampu : Adam Sekti Aji, S.kom.,M.kom**



**Disusun Oleh :**

ZIDAN ALFIAN MUBAROK (5230411107)

***TEKNIK INFORMATIKA  
FAKULTAS SAINS DAN TEKNOLOGI  
UNIVERSITAS TEKNOLOGI YOGYAKARTA*2024**

**Flask** adalah microframework Python yang populer untuk pengembangan web, dan arsitekturnya cukup fleksibel. Mari kita bahas komponen-komponen utamanya dan bagaimana mereka berinteraksi dalam suatu environment.

Penjelasan komponen-komponen utama dalam arsitektur Flask:

* Client Browser:
  + Mengirim HTTP request ke server.
  + Menerima dan menampilkan HTTP response dari server.
* Web Server (WSGI):
  + Biasanya menggunakan Gunicorn atau uWSGI dalam produksi.
  + Menerima HTTP request dan meneruskannya ke aplikasi Flask.
  + Mengirimkan HTTP response kembali ke client.
* Flask Application:
  + Inti dari proyek, menangani logika aplikasi.
  + Terdiri dari beberapa komponen utama: a. Views/Routes:
    - Mendefinisikan endpoint dan logika penanganan request.
    - Berinteraksi dengan models untuk mengakses/memodifikasi data.
    - Merender templates untuk menghasilkan respons.

b. Models:

* + - Merepresentasikan struktur data dan logika bisnis.
    - Berinteraksi dengan database (biasanya menggunakan ORM seperti SQLAlchemy).

c. Templates:

* + - File HTML dengan placeholder untuk data dinamis.
    - Dirender oleh views untuk menghasilkan output HTML.
* Database:
  + Menyimpan data aplikasi.
  + Bisa menggunakan berbagai jenis database (SQL atau NoSQL).
* Configuration:
  + Mengatur parameter aplikasi (misalnya, koneksi database, secret key).
  + Bisa berbeda untuk environment yang berbeda (development, testing, production).
* Extensions:
  + Menambahkan fungsionalitas tambahan ke Flask (misalnya, Flask-SQLAlchemy, Flask-Login).
* Static Files:
  + File CSS, JavaScript, gambar, dll.
  + Disajikan langsung oleh web server dalam produksi.
* Virtual Environment:
  + Isolasi dependencies proyek.
  + Memastikan konsistensi environment antar pengembang dan di produksi.

Implementasi Praktis:

1. Struktur Folder:

myflasklatihan/

├── app/

│ ├── \_\_init\_\_.py

│ ├── models.py

│ ├── views.py

│ └── templates/

├── config.py

├── requirements.txt

├── run.py

└── venv/

Inisialisasi Aplikasi (app/\_\_init\_\_.py):

from flask import Flask

from flask\_sqlalchemy import SQLAlchemy

app = Flask(\_\_name\_\_)

app.config.from\_object('config')

db = SQLAlchemy(app)

from app import views, models

Konfigurasi (config.py):

import os

BASE\_DIR = os.path.abspath(os.path.dirname(\_\_file\_\_))

SQLALCHEMY\_DATABASE\_URI = 'sqlite:///' + os.path.join(BASE\_DIR, 'app.db')

SECRET\_KEY = 'your-secret-key'

Model (app/models.py):

from app import db

class User(db.Model):

id = db.Column(db.Integer, primary\_key=True)

username = db.Column(db.String(64), unique=True)

# ... field lainnyaaa

View (app/views.py)

from flask import render\_template

from app import app

@app.route('/')

def index():

return render\_template('index.html')

Template (app/templates/index.html):

from flask import render\_template

from app import app

@app.route('/')

def index():

return render\_template('index.html')

Run Script (run.py):

from app import app

if \_\_name\_\_ == '\_\_main\_\_':

app.run(debug=True)

Penggunaan Virtual Environment:  
python -m venv venv   
source venv/bin/activate  
  
Arsitektur ini memberikan struktur yang terorganisir dan modular untuk proyek Flask, memudahkan pengembangan, pemeliharaan, dan skalabilitas. Setiap komponen memiliki tanggung jawab yang jelas, memungkinkan pengembang untuk bekerja pada bagian-bagian tertentu tanpa mempengaruhi keseluruhan sistem.