BAB III

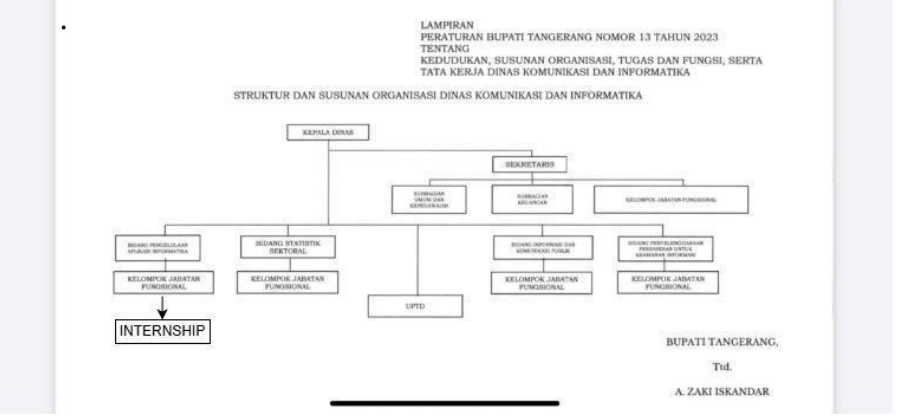
PELAKSANAAN KERJA

3.1 Kedudukan dan Koordinasi

Kedudukan dalam struktur organisasi menggambarkan posisi dan tanggung jawab selama pelaksanaan kegiatan magang di lingkungan kerja. Posisi ini menunjukkan alur pelaporan serta hubungan koordinasi dengan atasan langsung guna memastikan efektivitas kerja. Dalam pelaksanaannya di Dinas Komunikasi dan Informatika Kabupaten Tangerang, posisi yang ditempati berada di bawah Bidang Pengelolaan Aplikasi Informatika, khususnya pada unit Kelompok Jabatan Fungsional dengan status sebagai peserta internship. Berdasarkan bagan struktur organisasi tersebut, posisi pemagang berada di bawah koordinasi para Pejabat Fungsional yang secara struktural melapor dan bertanggung jawab langsung kepada Kepala Bidang Pengelolaan Aplikasi Informatika.

Peran sebagai intern di unit Kelompok Jabatan Fungsional ini berfokus pada dukungan teknis terhadap pengembangan, pengelolaan, serta pemeliharaan sistem aplikasi informatika di lingkup pemerintahan daerah. Tanggung jawab utama mencakup keterlibatan dalam operasional aplikasi, integrasi data, serta pemberian dukungan teknis pada implementasi layanan berbasis elektronik (e-government). Melalui peran ini, kontribusi yang diberikan bertujuan untuk mempercepat proses transformasi digital dan memastikan kualitas sistem informasi yang andal, sehingga mampu menghasilkan data yang akurat guna mendukung pengambilan keputusan strategis serta meningkatkan efisiensi pelayanan publik di Kabupaten Tangerang secara keseluruhan.

3.1.1 Kedudukan


LAMPIRAN
PERATURAN BUPATI TANGERANG NOMOR 13 TAHUN 2023
TENTANG
KEDUDUKAN, SUSUNAN ORGANISASI, TUGAS DAN FUNGSI, SERTA
TATA KERJA DINAS KOMUNIKASI DAN INFORMATIKA

STRUKTUR DAN SUSUNAN ORGANISASI DINAS KOMUNIKASI DAN INFORMATIKA

KEPALA DINAS

SEKRETARIS

KELOMPOK JABATAN FUNGSIONAL

KELOMPOK JABATAN FUNGSIONAL

KELOMPOK JABATAN FUNGSIONAL

KELOMPOK JABATAN FUNGSIONAL

INTERNSHIP

UPTD

BUPATI TANGERANG,

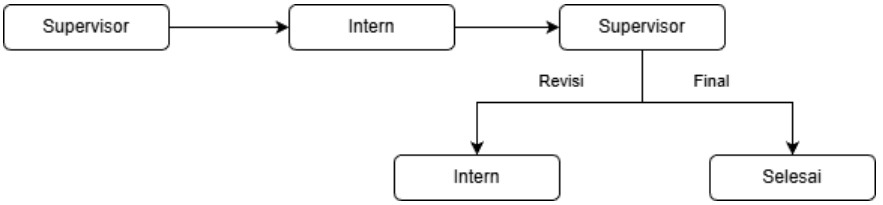
Ttd.

A. ZAKI ISKANDAR

3.1.2 Koordinasi

Alur koordinasi pekerjaan selama masa magang diatur melalui sistem komunikasi dua arah yang sistematis antara supervisor dan penulis untuk menjaga kualitas luaran teknis yang dihasilkan. Proses ini dimulai dengan tahap pemberian tugas oleh supervisor yang mencakup arahan strategis, parameter teknis, serta batasan waktu pengerjaan guna menyelaraskan persepsi mengenai ruang lingkup proyek. Setelah instruksi diterima, penulis melanjutkan ke tahap eksekusi pekerjaan dan pengumpulan hasil untuk ditinjau secara mendalam oleh supervisor melalui proses evaluasi atau *review*.

Dalam fase evaluasi ini, supervisor memiliki otoritas penuh untuk menentukan apakah hasil pekerjaan telah memenuhi standar operasional perusahaan atau masih memerlukan optimalisasi lebih lanjut. Apabila ditemukan ketidaksesuaian, supervisor akan menginstruksikan jalur revisi di mana penulis diwajibkan untuk memperbaiki pekerjaan sesuai dengan catatan evaluasi hingga mencapai kriteria yang diinginkan. Sebaliknya, jika pekerjaan dinyatakan valid dan berkualitas tinggi, supervisor akan memberikan status final sebagai tanda bahwa tugas tersebut telah selesai dan siap diimplementasikan secara resmi. Penerapan alur yang terstruktur ini memastikan setiap progres pengembangan sistem terpantau secara konsisten guna meminimalisir risiko kesalahan teknis.



Gambar 3 .3.1 Bagan Alur Koordinasi

3.2 Tugas yang Dilakukan

Berisi tabel hal-hal yang penulis lakukan selama menjalankan program.

Minggu	Proyek	Keterangan
1	Inisialisasi Proyek	Observasi infrastruktur IT di Diskominfo dan instalasi lingkungan kerja Python.
2	Analisis Sistem	Identifikasi kebutuhan fungsional sistem pengenalan wajah dan protokol RTSP.
3	Struktur Data	Perancangan direktori utama face_recognition_data untuk penyimpanan aset.
4	Manajemen Log	Implementasi sistem pencatatan deteksi otomatis dalam format JSON.
5	Arsitektur Backend	Inisialisasi kerangka kerja Flask dan pengaturan rute API dasar.
6	Integrasi Kamera	Konfigurasi OpenCV untuk menangkap aliran video dari infrastruktur CCTV.
7	Antarmuka Pengguna	Pengembangan struktur HTML awal menggunakan sistem Template Engine Jinja2.
8	Manajemen Upload	Pembuatan direktori uploads sebagai zona penyangga file temporer.
9	Keamanan File	Implementasi validasi ekstensi dan ukuran berkas pada folder unggahan sementara.
10	Pemrosesan Biometrik	Integrasi model ekstraksi fitur wajah untuk konversi citra menjadi embedding.
11	Database Biometrik	Pengembangan sistem penyimpanan identitas pada berkas face_database.pkl.

12	Redundansi Data	Implementasi mekanisme backup otomatis untuk menjaga integritas data base wajah
13	Sistem Duplikasi	Pengembangan logika pengecekan wajah ganda pada database saat registrasi baru.
14	Optimasi Real-Time	Kalibrasi ambang batas akurasi deteksi untuk mengurangi tingkat kesalahan identifikasi.
15	Pengujian Sistem	Uji coba integrasi menyeluruh dari proses upload hingga identifikasi wajah.
16	Manajemen Penyimpanan	Otomatisasi pembersihan folder uploads setelah proses ekstraksi fitur selesai.
17	Monitoring Log	Evaluasi data deteksi harian dan visualisasi log pada dasbor admin.
18	Finalisasi Teknis	Perbaikan minor pada antarmuka dan optimasi waktu respons deteksi wajah.
19	Dokumentasi	Penyusunan laporan akhir magang dan serah terima teknis proyek ke divisi terkait.

Tabel 1 Detail Pekerjaan yang Dilakukan

3.3 Uraian Pelaksanaan Kerja

Proses Pelaksanaan

3.3.1 Membuat proyek Utama (Back-End/Core Systems)

Implementasi perangkat lunak pengenalan wajah ini menitikberatkan pada pengembangan komponen *backend* dan *core systems* sebagai fondasi utama dalam arsitektur sistem secara keseluruhan. Penempatan komponen-komponen ini sebagai inti sistem didasarkan pada pertimbangan bahwa fungsionalitas utama aplikasi bergantung pada pemrosesan data biometrik yang kompleks dan manajemen aliran data (*data stream*) secara *real-time*. Sebagai mesin utama, *core systems* bertanggung jawab dalam mengintegrasikan berbagai modul krusial, mulai dari akuisisi video melalui protokol *Real Time Streaming Protocol* (RTSP) dari perangkat CCTV, hingga proses ekstraksi fitur wajah menggunakan algoritma *Deep Learning*. Dengan menempatkan logika ini pada posisi sentral, sistem dapat menjalankan fungsinya sebagai pengolah informasi biometrik yang akurat sekaligus

menjadi penyedia data bagi antarmuka pengguna. Pengembangan sistem utama pada tahap ini juga berfungsi sebagai langkah strategis dalam menjamin stabilitas dan efisiensi operasional perangkat lunak. Melalui implementasi *multithreading* dan manajemen konfigurasi terpusat, penulis dapat memastikan bahwa proses deteksi wajah yang membutuhkan sumber daya komputasi tinggi tidak menginterupsi responsivitas sistem secara umum. Perancangan *core system* yang matang memungkinkan definisi struktur data dan *embedding* numerik yang konsisten dalam basis data, sehingga memudahkan proses sinkronisasi saat integrasi dengan fitur-fitur lainnya. Pendekatan ini memberikan kerangka kerja yang solid dalam mengidentifikasi kebutuhan teknis, seperti ambang batas kesamaan wajah (*similarity threshold*) dan optimalisasi *frame rate*, yang secara langsung berpengaruh pada keandalan sistem dalam mengenali identitas secara otomatis.

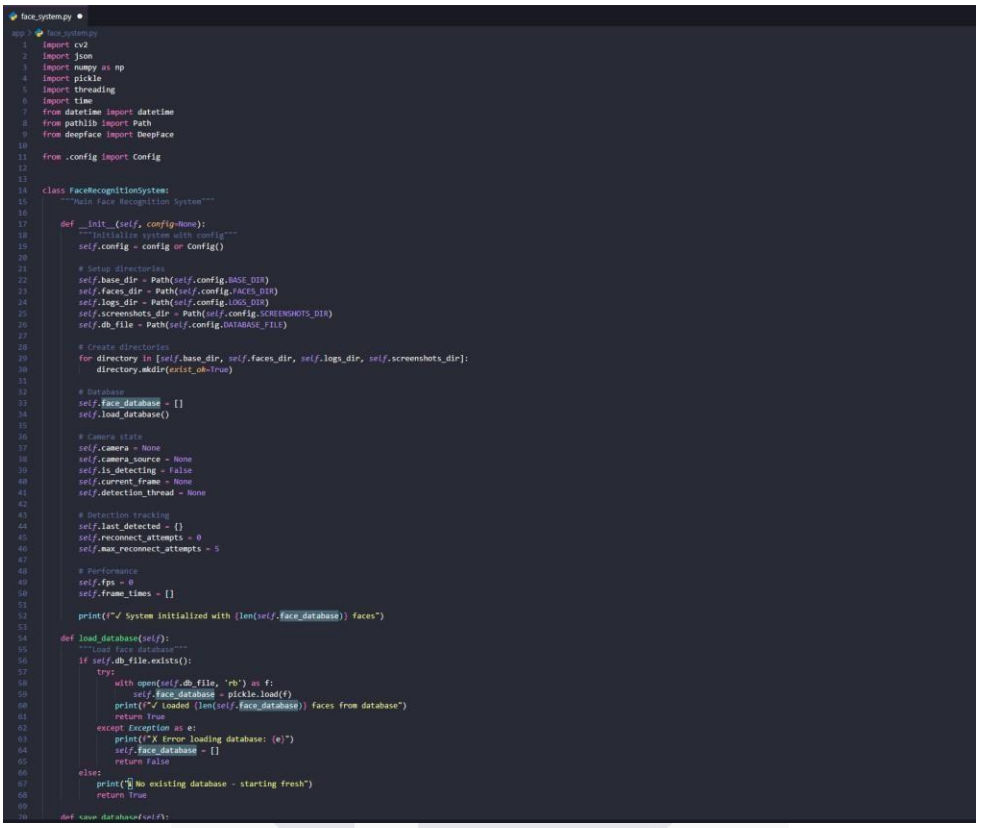
Selain itu, fokus pada sistem utama memungkinkan dilakukannya validasi teknis terhadap akurasi algoritma dan keamanan integritas data sedari dini. Melalui pengembangan modul pengenalan wajah yang terpusat, penulis dapat mengevaluasi performa model *Facenet* dan *backend* detektor dalam berbagai kondisi lingkungan dan pencahayaan. Evaluasi ini menjadi sangat penting dalam konteks sistem keamanan dan buku tamu digital, di mana presisi identifikasi merupakan parameter keberhasilan yang paling krusial. Masukan teknis yang diperoleh dari pengujian modul inti ini kemudian menjadi landasan untuk melakukan kalibrasi parameter sistem, sehingga risiko terjadinya kegagalan identifikasi atau duplikasi data pada saat implementasi penuh dapat ditekan seminimal mungkin.

Dari sisi pengembangan berkelanjutan, standarisasi sistem utama melalui *RESTful API* mendukung skalabilitas proyek di masa depan. Dengan mendefinisikan *endpoint* yang jelas untuk fungsi pendaftaran wajah, pengambilan log deteksi, dan kontrol kamera, proses integrasi dengan sisi *front-end* dapat berjalan secara lebih sistematis dan terorganisir. Pendekatan

ini memastikan bahwa setiap komponen sistem berinteraksi melalui jalur data yang telah terdefinisi secara baku, sehingga mempermudah proses pemeliharaan dan pengembangan fitur tambahan. Oleh karena itu, klasifikasi dan pengembangan modul-modul di atas sebagai sistem utama merupakan langkah teknis yang fundamental dan strategis dalam mendukung keberhasilan implementasi sistem pengenalan wajah yang andal dan profesional.

3.3.2 Face Recognation System

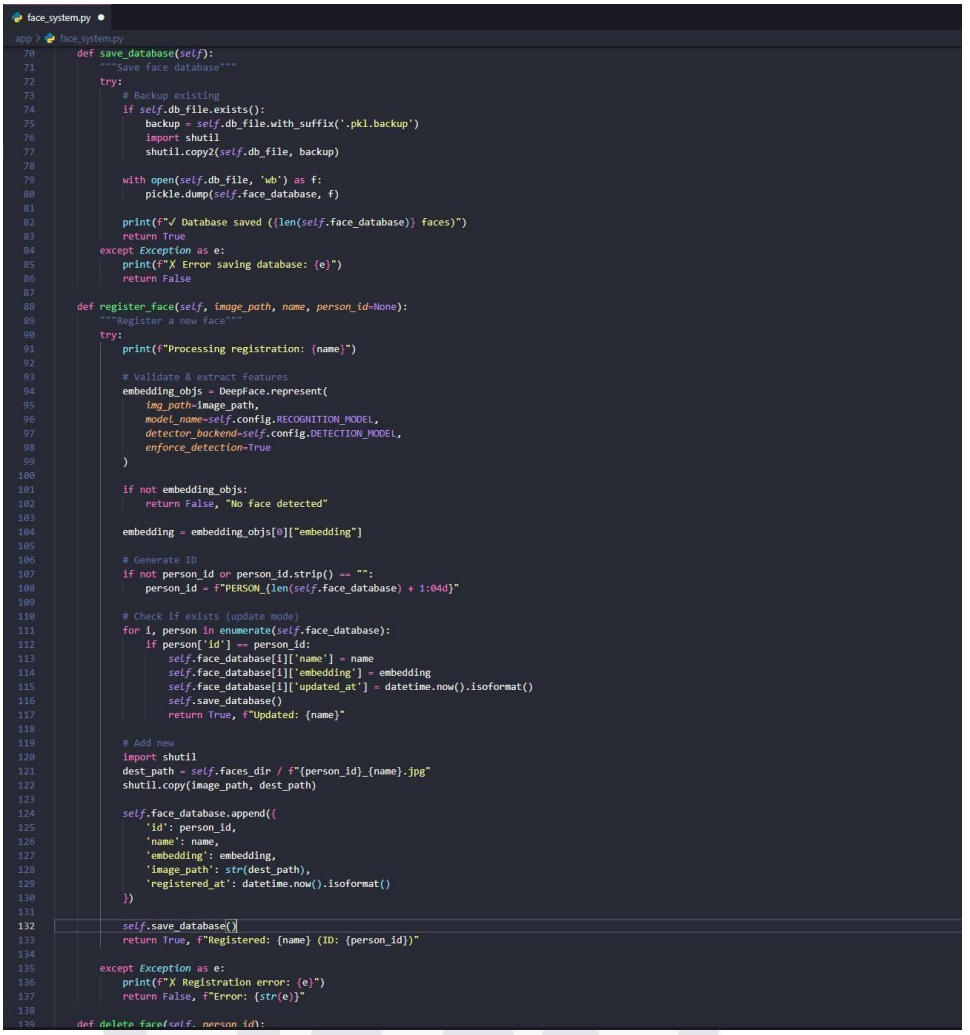
Modul face_system.py dalam pengembangan aplikasi ini merupakan representasi dari arsitektur inti (*core backend*) yang mengintegrasikan kecerdasan buatan dengan sistem pemrosesan video secara *real-time*. Kelas FaceRecognitionSystem bertindak sebagai entitas pengendali pusat yang mengelola seluruh siklus hidup data biometrik, mulai dari inisialisasi lingkungan direktori hingga manajemen basis data wajah yang bersifat persisten. Implementasi ini menggunakan pendekatan pemrograman berbasis objek untuk mengonapsulasi logika bisnis, di mana ketergantungan sistem terhadap pustaka eksternal seperti DeepFace dan OpenCV menunjukkan adanya penerapan metode *state-of-the-art* dalam bidang *Computer Vision*. Secara teknis, sistem ini mengotomatisasi pembuatan struktur folder untuk penyimpanan citra wajah, log deteksi, dan cuplikan layar, yang kemudian dikelola melalui mekanisme pemuatan basis data menggunakan teknik serialisasi data melalui pustaka *pickle*.[2]



```python
import cv2
import json
import numpy as np
import pickle
import threading
import time
from datetime import datetime
from pathlib import Path
from deepface import DeepFace

from .config import Config


class FaceRecognitionSystem:
    """Main Face Recognition System"""

    def __init__(self, config=None):
        """Initialize system with config"""
        self.config = config or Config()

        # Setup directories
        self.base_dir = Path(self.config.BASE_DIR)
        self.faces_dir = Path(self.config.FACES_DIR)
        self.logs_dir = Path(self.config.LOGS_DIR)
        self.screenshots_dir = Path(self.config.SCREENSHOTS_DIR)
        self.db_file = Path(self.config.DATABASE_FILE)

        # Create directories
        for directory in [self.base_dir, self.faces_dir, self.logs_dir, self.screenshots_dir]:
            directory.mkdir(exist_ok=True)

        # Database
        self.face_database = []
        self.load_database()

        # Camera state
        self.camera = None
        self.camera_source = None
        self.is_detecting = False
        self.current_frame = None
        self.detection_thread = None

        # Detection tracking
        self.last_detected = {}
        self.reconnect_attempts = 0
        self.max_reconnect_attempts = 5

        # Performance
        self.fps = 0
        self.frame_times = []

        print(f"✓ System initialized with {len(self.face_database)} faces")

    def load_database(self):
        """Load face database"""
        if self.db_file.exists():
            try:
                with open(self.db_file, 'rb') as f:
                    self.face_database = pickle.load(f)
                print(f"✓ Loaded {len(self.face_database)} faces from database")
                return True
            except Exception as e:
                print(f"X Error loading database: {e}")
                self.face_database = []
                return False
        else:
            print("No existing database - starting fresh")
            return True

    def save_database(self):
```

Gambar 3.2 Code face_system.py

Dalam aspek pemrosesan data, fungsi register_face dan find_match merupakan komponen krusial yang menerapkan prinsip ekstraksi fitur (*feature extraction*) dan pencocokan pola. Proses registrasi wajah dilakukan dengan melakukan transformasi citra menjadi representasi numerik atau *embedding* yang kemudian disimpan dalam basis data sebagai referensi biometrik. Untuk mencapai tingkat akurasi yang optimal, sistem menggunakan algoritma *cosine distance* sebagai metrik untuk mengukur tingkat kemiripan antara wajah yang terdeteksi dengan data yang telah terdaftar. Pendekatan matematis ini memungkinkan sistem untuk memberikan penilaian kuantitatif terhadap identitas seseorang, yang direpresentasikan dalam bentuk nilai probabilitas atau tingkat kepercayaan (*confidence level*). Selain itu, adanya mekanisme *similarity threshold* memastikan bahwa hanya hasil pencocokan dengan skor kemiripan tertentu yang dapat divalidasi sebagai identitas yang dikenali, sehingga meminimalkan potensi kesalahan identifikasi.



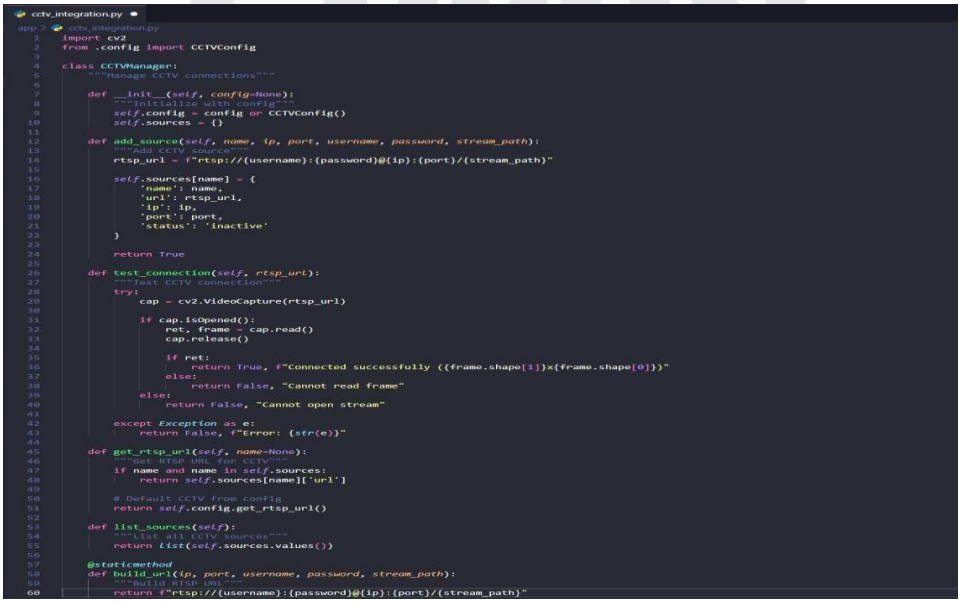
Gambar 3.2 Code face_system.py

Efisiensi operasional sistem didukung oleh implementasi *multithreading* pada fungsi start_camera dan _detection_loop. Dengan memisahkan proses pengambilan *frame* video dan analisis deteksi ke dalam utas (*thread*) yang berbeda, sistem mampu menjaga stabilitas performa meskipun melakukan komputasi yang intensif. Logika deteksi juga dilengkapi dengan metode optimasi seperti *frame skipping* dan mekanisme *cooldown* pada pencatatan log guna mencegah redundansi data yang tidak perlu. Integrasi dengan perangkat keras, baik melalui *webcam* maupun protokol CCTV, dikelola dengan sangat teliti melalui penanganan kesalahan dan percobaan koneksi ulang otomatis yang menjamin ketersediaan sistem secara berkelanjutan. Secara keseluruhan, kode ini tidak hanya berfungsi sebagai pengenal wajah,

tetapi juga sebagai sistem manajemen informasi yang komprehensif yang menghubungkan lapisan sensorik dengan lapisan penyimpanan data secara sistematis.

3.3.3 CCTV Integration System

Modul cctv_integration.py merupakan komponen infrastruktur yang berfungsi sebagai jembatan komunikasi antara perangkat keras kamera keamanan dengan sistem pemrosesan biometrik pusat. Secara arsitektural, kelas CCTVManager diimplementasikan untuk mengelola abstraksi koneksi jaringan menggunakan protokol *Real Time Streaming Protocol* (RTSP), yang merupakan standar industri dalam transmisi data video berkualitas tinggi melalui jaringan IP. Fokus utama dari modul ini adalah untuk menyederhanakan kompleksitas manajemen sumber video yang bervariasi dengan cara merangkapsulasi parameter otentikasi, alamat jaringan, dan jalur transmisi (*stream path*) ke dalam sebuah entitas data yang terstruktur. Hal ini sangat krusial dalam konteks implementasi *Smart City* atau pengawasan gedung berskala besar, di mana efisiensi dalam mengelola banyak titik kamera menjadi prioritas utama untuk menjamin kontinuitas aliran data input.



Gambar 3.3 Code cctv_integration.py

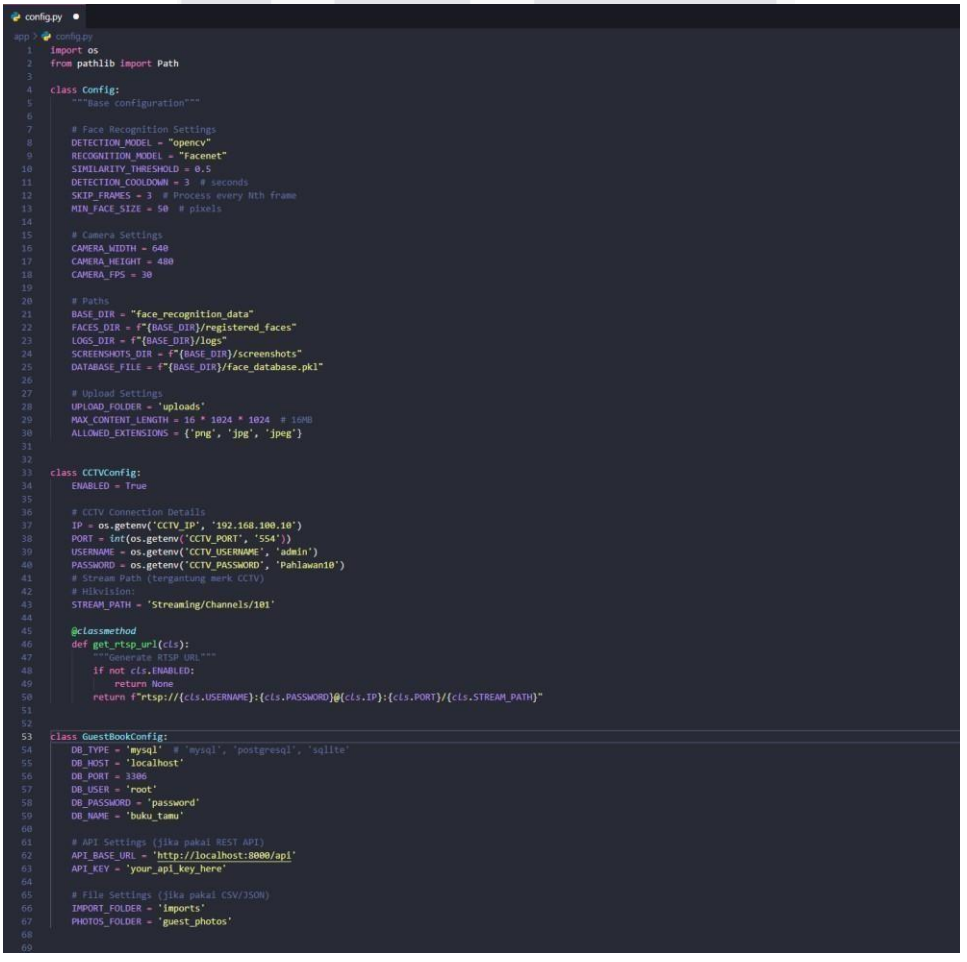
Pengembangan modul ini juga menerapkan prinsip validasi dini melalui fungsi test_connection untuk memastikan keandalan sistem sebelum proses pemrosesan citra yang berat dilakukan. Dengan memanfaatkan pustaka OpenCV, sistem melakukan uji coba pembukaan aliran data (*handshake*) dan verifikasi pembacaan bingkai (*frame*) secara aktual untuk mengonfirmasi bahwa jalur komunikasi tidak hanya terbuka, tetapi juga mampu mengirimkan data visual yang valid. Langkah teknis ini berfungsi sebagai mekanisme mitigasi kegagalan sistem, sehingga potensi gangguan seperti kesalahan konfigurasi IP, kegagalan otentikasi, atau latensi jaringan dapat diidentifikasi secara prematur sebelum masuk ke lapisan logika pengenalan wajah. Pendekatan ini memastikan bahwa integritas data yang masuk ke mesin *backend* selalu berada dalam kondisi siap proses, yang secara langsung meningkatkan efektivitas sistem secara keseluruhan.[3]

Selain aspek fungsionalitas koneksi, modul ini memberikan fleksibilitas tinggi dalam hal skalabilitas sistem melalui metode statis build_url dan manajemen kamus sumber daya (*source dictionary*). Struktur ini memungkinkan admin sistem untuk menambahkan, memperbarui, atau menghapus parameter kamera secara dinamis tanpa mengganggu integritas kode program utama. Dengan mengintegrasikan sistem manajemen CCTV ini dengan modul konfigurasi CCTVConfig, penulis membangun kerangka kerja yang mendukung prinsip *interoperabilitas*, di mana sistem dapat beradaptasi dengan berbagai merk dan spesifikasi perangkat kamera IP yang mendukung standar RTSP. Secara akademis, implementasi modul ini menunjukkan pemahaman mendalam mengenai arsitektur sistem terdistribusi, di mana pemisahan antara manajemen transmisi data dan logika pemrosesan data menjadi kunci dalam membangun aplikasi yang tangguh, modular, dan mudah dipelihara.

3.3.4 Configuration System

Modul config.py dalam arsitektur perangkat lunak ini berfungsi sebagai repositori sentral bagi seluruh parameter operasional dan variabel

lingkungan yang mengatur perilaku sistem. Secara akademis, implementasi ini menerapkan prinsip *Separation of Concerns* (SoC), di mana pengaturan teknis dipisahkan dari logika program utama untuk meningkatkan fleksibilitas dan kemudahan pemeliharaan. Melalui penyediaan kelas-kelas konfigurasi seperti Config, CCTVConfig, dan GuestBookConfig, sistem mampu mengelola berbagai aspek krusial, mulai dari spesifikasi model *Deep Learning* seperti Facenet, ambang batas kesamaan biometrik, hingga parameter manajemen memori dan penyimpanan. Pengorganisasian ini memungkinkan pengembang untuk melakukan kalibrasi sistem secara presisi tanpa harus memodifikasi kode sumber pada lapisan fungsional, yang merupakan praktik terbaik dalam rekayasa perangkat lunak modern.



Gambar 3.4 Code config.py

Selain sebagai pusat pengaturan, modul ini juga berperan penting dalam mendefinisikan infrastruktur data melalui standarisasi jalur direktori dan

manajemen berkas. Dengan menetapkan variabel seperti BASE_DIR, FACES_DIR, dan DATABASE_FILE, sistem menjamin adanya konsistensi dalam tata kelola aset digital, yang sangat penting untuk integritas basis data biometrik. Di sisi lain, kelas CCTVConfig menunjukkan kapabilitas sistem dalam beradaptasi dengan infrastruktur jaringan yang berbeda melalui pemanfaatan variabel lingkungan (*environment variables*). Hal ini memberikan lapisan keamanan tambahan karena informasi sensitif seperti alamat IP dan kredensial akses CCTV dapat dikelola secara eksternal melalui sistem operasi, sehingga mengurangi risiko kebocoran data pada repositori kode. Lebih lanjut, penerapan kelas DevelopmentConfig dan ProductionConfig mencerminkan strategi penyebaran (*deployment*) yang profesional dan terukur. Dengan membedakan parameter operasional antara lingkungan pengembangan dan produksi, seperti penyesuaian nilai SKIP_FRAMES dan DETECTION_COOLDOWN, penulis dapat mengoptimalkan efisiensi komputasi berdasarkan ketersediaan sumber daya perangkat keras. Strategi ini memastikan bahwa sistem tetap responsif selama tahap pengujian namun beralih ke mode performa tinggi yang lebih stabil saat diimplementasikan secara aktual di lapangan. Oleh karena itu, modul konfigurasi ini merupakan komponen fundamental yang menjamin *portabilitas* dan *skalabilitas* aplikasi, yang memungkinkannya untuk diintegrasikan dengan berbagai sistem basis data eksternal maupun layanan API pihak ketiga secara sistematis dan terstruktur.[4]

3.3.5 Routes System

Fungsi register_routes dalam pengembangan aplikasi ini bertindak sebagai lapisan antarmuka pemrograman aplikasi (*Application Programming Interface* atau API) yang mengintegrasikan logika sistem utama dengan protokol komunikasi web. Secara akademis, modul ini mengimplementasikan pola arsitektur *Model-View-Controller* (MVC), di mana setiap rute (*route*) berfungsi sebagai penghubung yang mengoordinasikan permintaan pengguna dengan layanan pemrosesan biometrik dan manajemen CCTV. Melalui

pemanfaatan kerangka kerja Flask, sistem ini mampu menyediakan titik akhir (*endpoint*) yang efisien untuk berbagai operasi krusial, mulai dari pendaftaran identitas baru melalui metode POST, pengambilan data log aktivitas, hingga pengendalian aliran video (*video streaming*) secara asinkron. Penulisan kode ini menjamin bahwa interaksi antara pengguna dan mesin deteksi berjalan secara sistematis melalui standarisasi format data JSON.

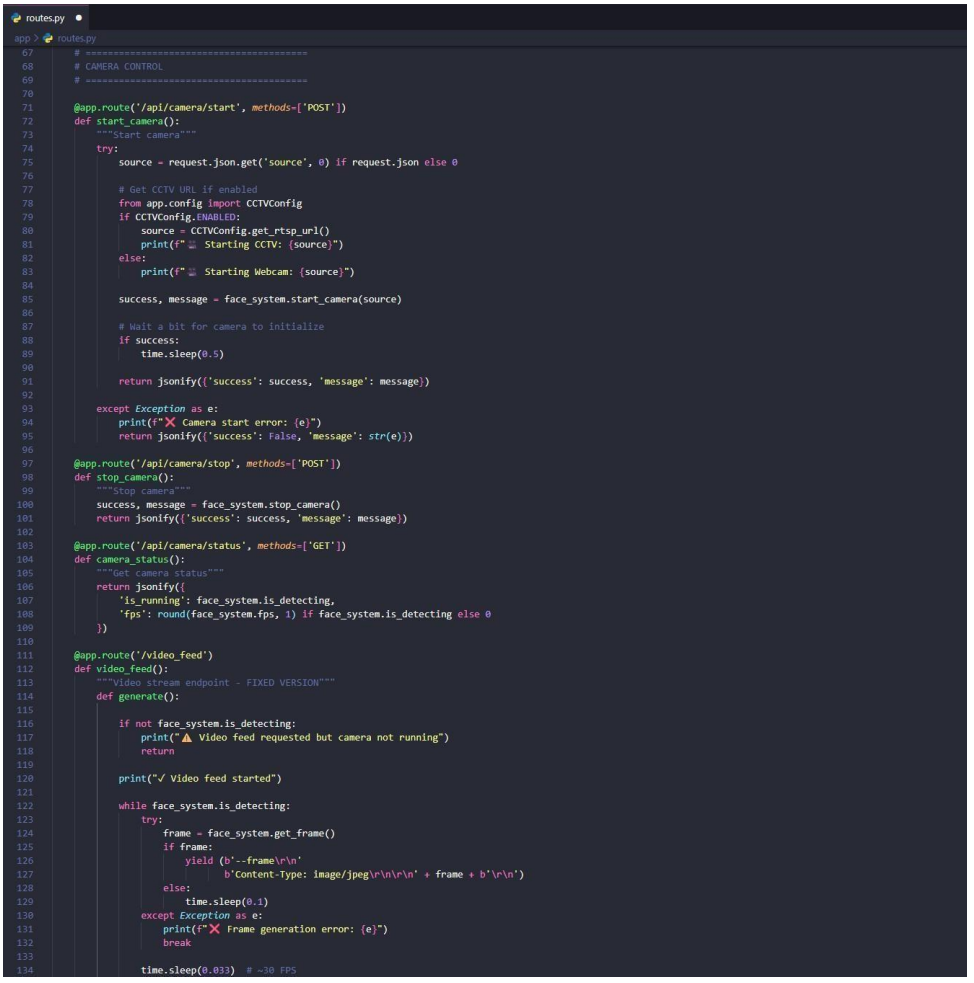
routes.py

```python
from flask import render_template, request, jsonify, Response
from werkzeug.utils import secure_filename
import os
import time


def register_routes(app, face_system, cctv_manager):

    @app.route('/')
    def index():

        return render_template('face_recognation.html')

    # ======================================
    # FACE REGISTRATION
    # ======================================

    @app.route('/api/register', methods=['POST'])
    def register():
        """Register new face"""
        try:
            name = request.form.get('name', '').strip()
            person_id = request.form.get('person_id', '').strip()
            file = request.files.get('photo')

            if not name or not file:
                return jsonify({
                    'success': False,
                    'message': 'Name and photo required'
                })

            # Save file
            filename = secure_filename(file.filename)
            filepath = os.path.join(app.config['UPLOAD_FOLDER'], filename)
            file.save(filepath)

            # Register face
            success, message = face_system.register_face(
                filepath, name, person_id if person_id else None
            )

            # Cleanup
            try:
                os.remove(filepath)
            except:
                pass

            return jsonify({'success': success, 'message': message})

        except Exception as e:
            return jsonify({'success': False, 'message': str(e)})

    @app.route('/api/faces', methods=['GET'])
    def get_faces():
        """Get registered faces"""
        return jsonify({
            'success': True,
            'faces': face_system.get_registered_faces()
        })

    @app.route('/api/delete/<person_id>', methods=['DELETE'])
    def delete_face(person_id):
        """Delete face"""
        success, message = face_system.delete_face(person_id)
        return jsonify({'success': success, 'message': message})
```

Gambar 3.5 Code routes.py

Dalam aspek fungsionalitas registrasi, rute /api/register menerapkan mekanisme penanganan berkas yang aman dengan menggunakan

secure_filename untuk mencegah kerentanan keamanan pada sistem penyimpanan server. Proses ini mencakup siklus hidup pengelolaan data yang lengkap, di mana citra yang diunggah diproses terlebih dahulu oleh face_system untuk ekstraksi fitur sebelum akhirnya dilakukan pembersihan data sementara (*cleanup*) guna menjaga efisiensi kapasitas penyimpanan. Di sisi lain, fungsionalitas utama berupa aliran video langsung diakomodasi melalui rute video_feed yang menggunakan teknik *Multipart Response*. Mekanisme ini memungkinkan sistem untuk mengirimkan bingkai citra secara terus-menerus kepada peramban pengguna, menciptakan efek visual video yang halus dengan kecepatan bingkai yang dioptimalkan, yang merupakan elemen vital dalam pengawasan keamanan secara *real-time*.[5]



Gambar 3.6 Code routes.py

Selain itu, modul ini menyediakan kontrol infrastruktur yang komprehensif

melalui manajemen kamera dan integrasi CCTV berbasis RTSP. Rute /api/camera/start menunjukkan kapabilitas sistem dalam melakukan alokasi sumber daya secara dinamis, baik melalui perangkat keras lokal maupun aliran data jaringan IP. Integrasi ini didukung oleh sistem pemantauan performa yang memberikan informasi status operasional dan nilai *Frames Per Second* (FPS) secara akurat kepada pengguna. Dari perspektif manajemen data, ketersediaan statistik sistem dan log harian melalui rute/api/stats memberikan nilai tambah berupa transparansi operasional dan akuntabilitas data biometrik. Oleh karena itu, modul rute ini merupakan komponen strategis yang menyatukan seluruh subsistem fungsional ke dalam satu kesatuan aplikasi yang interaktif, stabil, dan siap diimplementasikan dalam lingkungan layanan publik digital.[6]

3.3.6 Aplikasi Utama Sistem Face Recognation

Penyusunan komponen aplikasi utama dalam proyek ini dirancang sebagai orkestrator sentral yang menyatukan seluruh logika bisnis, pemrosesan algoritma, dan manajemen konektivitas dalam satu ekosistem yang kohesif. Fokus pengembangan aplikasi utama ini didasarkan pada kebutuhan untuk memiliki sebuah titik kendali terpusat yang mampu mengoordinasikan interaksi antara lapisan perangkat keras sensorik dengan lapisan antarmuka pengguna secara sinkron. Dengan menempatkan seluruh inisialisasi sistem pada blok aplikasi utama, penulis dapat menjamin bahwa setiap dependensi, mulai dari pemuatan konfigurasi lingkungan hingga pengaktifan mesin pengenal wajah, berjalan sesuai dengan urutan hirarki yang tepat sebelum layanan dapat diakses oleh pengguna.[7]

Pengembangan aplikasi utama ini juga berperan penting dalam mengelola stabilitas sistem melalui penerapan mekanisme penanganan kesalahan (*error handling*) dan manajemen siklus hidup aplikasi yang komprehensif. Melalui integrasi antara skrip eksekusi dan rute API, aplikasi utama bertindak sebagai *middleware* yang memvalidasi setiap aliran data yang masuk dan keluar, sehingga integritas informasi biometrik tetap terjaga selama proses

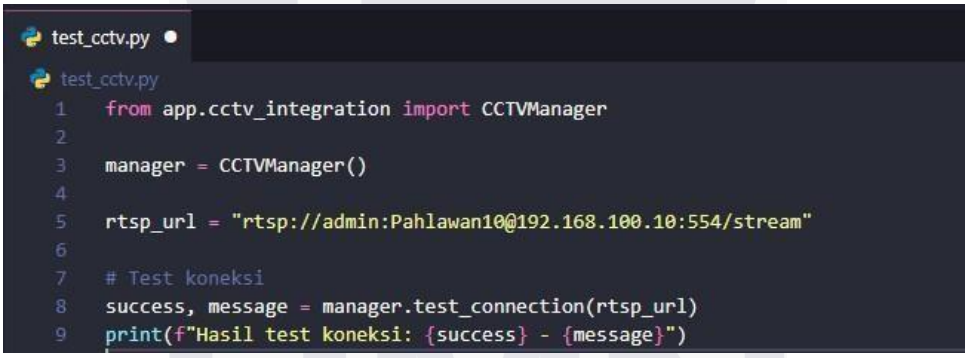
operasional. Perancangan yang matang pada bagian ini memungkinkan sistem untuk menjalankan tugas-tugas berat secara paralel, seperti pemrosesan deteksi wajah dan pengiriman aliran video langsung, tanpa mengorbankan performa *server* secara keseluruhan. Hal ini menciptakan fondasi yang kuat bagi sistem untuk beroperasi secara kontinu dalam lingkungan produksi yang menuntut ketersediaan tinggi. Selain sebagai pengendali teknis, aplikasi utama ini memberikan kemudahan dalam aspek pemantauan dan audit sistem melalui konsolidasi log aktivitas dan statistik performa. Dengan memusatkan titik masuk aplikasi, penulis dapat dengan mudah melakukan kalibrasi ulang terhadap parameter sistem, seperti pengaturan alamat IP CCTV atau penyesuaian model deteksi, tanpa harus melakukan perombakan pada modul-modul fungsional yang lebih kecil. Fleksibilitas ini sangat krusial dalam tahap pengembangan dan pengujian, karena memungkinkan identifikasi masalah dilakukan secara lebih terisolasi dan sistematis. Keberadaan mekanisme uji koneksi yang terintegrasi di dalam aplikasi utama juga memastikan bahwa setiap titik kamera yang terhubung telah terverifikasi secara teknis sebelum proses identifikasi biometrik dimulai.

Secara keseluruhan, pengembangan aplikasi utama ini merupakan representasi dari arsitektur sistem yang modular namun terintegrasi secara kuat, yang mendukung prinsip skalabilitas perangkat lunak modern. Dengan mendefinisikan alur kerja yang terstruktur mulai dari tahap inisialisasi hingga penghentian sistem (*graceful shutdown*), aplikasi ini mampu memberikan jaminan keamanan data dan akurasi identifikasi yang konsisten. Oleh karena itu, klasifikasi pengembangan komponen ini sebagai aplikasi utama adalah langkah strategis yang menentukan keberhasilan integrasi seluruh fitur cerdas dalam proyek pengenalan wajah ini, menjadikannya sebuah solusi teknologi yang andal, efisien, dan siap pakai.

3.3.7 Test Script

Modul test_cctv.py merupakan komponen validasi yang dirancang

khusus untuk melakukan verifikasi teknis terhadap integritas koneksi antara perangkat lunak dengan infrastruktur kamera pemantau. Dalam siklus pengembangan sistem, berkas ini berfungsi sebagai prosedur pengujian unit (*unit testing*) yang bertujuan untuk mengisolasi dan memastikan bahwa jalur transmisi data melalui protokol *Real Time Streaming Protocol* (RTSP) telah terkonfigurasi dengan tepat. Fokus utama dari implementasi skrip ini adalah untuk melakukan uji coba otentikasi dan jabat tangan (*handshake*) antara peladen aplikasi dengan perangkat keras kamera sebelum sistem pengenalan identitas diaktifkan secara penuh. Penentuan prosedur pengujian yang terpisah ini sangat penting untuk memastikan bahwa kendala jaringan atau kesalahan kredensial akses dapat diidentifikasi secara prematur tanpa melibatkan beban komputasi dari modul pemrosesan data lainnya.[8]



Gambar 3.7 Code test_cctv.py

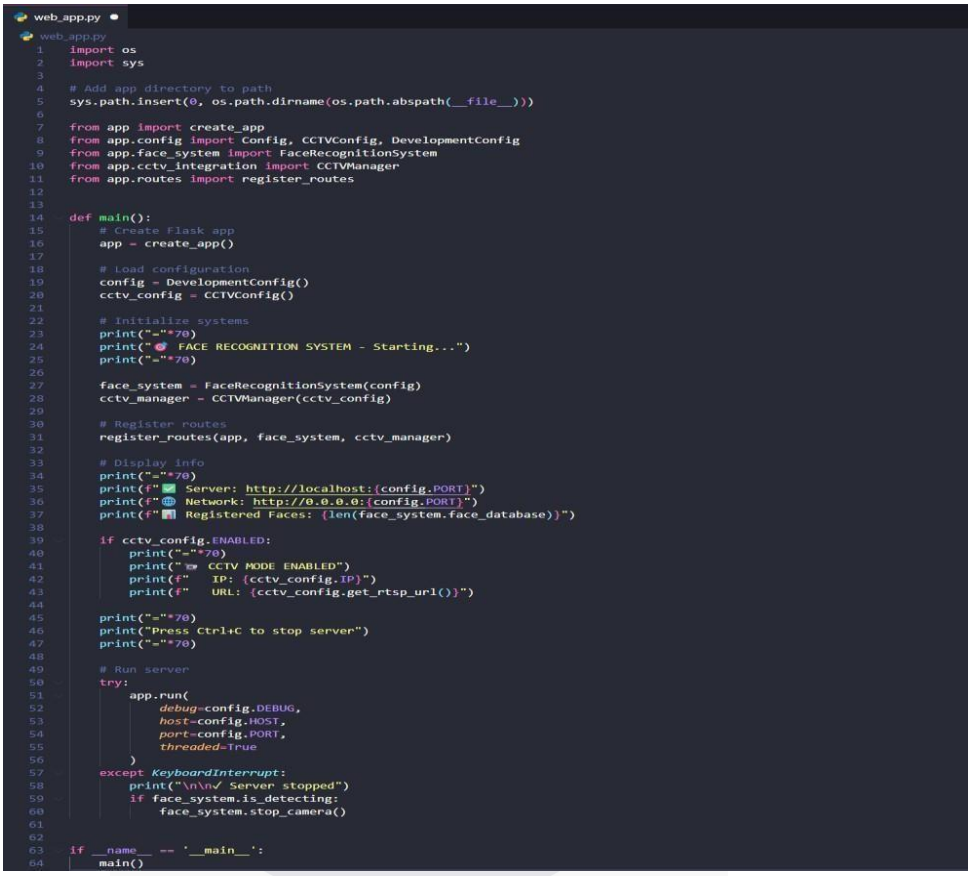
Secara teknis, proses pengujian dilakukan dengan menginisialisasi objek manajer konektivitas yang kemudian mengeksekusi metode pengujian tautan terhadap alamat jaringan spesifik. Melalui fungsi test_connection, sistem melakukan upaya pembukaan aliran data video dan verifikasi terhadap kemampuan perangkat dalam mengirimkan bingkai citra (*frame*) secara aktual. Hasil dari prosedur ini memberikan data diagnostik berupa status keberhasilan koneksi serta informasi mengenai dimensi resolusi video yang diterima, yang menjadi parameter penting dalam menentukan kualitas masukan data bagi algoritma pemrosesan selanjutnya. Pendekatan sistematis ini menjamin bahwa seluruh data visual yang masuk ke dalam sistem utama telah memenuhi standar teknis yang diperlukan, sehingga integritas

operasional aplikasi tetap terjaga.

Selain berfungsi sebagai alat diagnostik, modul pengujian ini memberikan landasan bagi transparansi sistem dalam hal manajemen infrastruktur jaringan. Dengan adanya pelaporan hasil uji koneksi yang informatif, administrator sistem dapat melakukan kalibrasi ulang terhadap pengaturan alamat IP, nomor port, maupun parameter keamanan jika terjadi kegagalan transmisi. Implementasi ini mencerminkan metodologi pengembangan yang terukur, di mana aspek keandalan konektivitas menjadi prioritas utama untuk mendukung fungsionalitas sistem pemantauan cerdas. Oleh karena itu, ketersediaan modul pengujian ini merupakan langkah strategis yang mendukung akuntabilitas teknis dan memudahkan proses pemeliharaan sistem dalam jangka panjang, terutama pada lingkungan implementasi yang melibatkan banyak perangkat sensorik tersebar.

3.3.8 Main Web Application

Berkas web_app.py berperan sebagai modul eksekusi utama yang berfungsi mengorkestrasikan seluruh komponen sistem ke dalam satu kesatuan operasional yang terintegrasi. Secara arsitektural, modul ini bertanggung jawab atas proses inisialisasi lingkungan kerja, pemuatan konfigurasi sistem, serta pengaktifan layanan web berbasis kerangka kerja Flask. Penentuan modul ini sebagai titik masuk utama didasarkan pada kebutuhan untuk mengonsolidasikan berbagai subsistem, seperti manajemen basis data biometrik dan integrasi perangkat kamera, agar dapat berjalan secara sinkron. Dengan mengatur alur inisialisasi secara terpusat, penulis dapat memastikan bahwa seluruh ketergantungan antar-modul telah terpenuhi dengan benar sebelum sistem memberikan layanan kepada pengguna, sehingga meminimalkan risiko kegagalan fungsi pada tahap operasional.



```python
import os
import sys

# Add app directory to path
sys.path.insert(0, os.path.dirname(os.path.abspath(__file__)))

from app import create_app
from app.config import Config, CCTVConfig, DevelopmentConfig
from app.face_system import FaceRecognitionSystem
from app.cctv_integration import CCTVManager
from app.routes import register_routes


def main():
    # Create Flask app
    app = create_app()

    # Load configuration
    config = DevelopmentConfig()
    cctv_config = CCTVConfig()

    # Initialize systems
    print("="*70)
    print("🎯 FACE RECOGNITION SYSTEM - Starting...")
    print("="*70)

    face_system = FaceRecognitionSystem(config)
    cctv_manager = CCTVManager(cctv_config)

    # Register routes
    register_routes(app, face_system, cctv_manager)

    # Display info
    print("="*70)
    print(f"✅ Server: http://localhost:{config.PORT}")
    print(f"🌐 Network: http://0.0.0.0:{config.PORT}")
    print(f"📊 Registered Faces: {len(face_system.face_database)}")

    if cctv_config.ENABLED:
        print("="*70)
        print("📹 CCTV MODE ENABLED")
        print(f"   IP: {cctv_config.IP}")
        print(f"   URL: {cctv_config.get_rtsp_url()}")

    print("="*70)
    print("Press Ctrl+C to stop server")
    print("="*70)

    # Run server
    try:
        app.run(
            debug=config.DEBUG,
            host=config.HOST,
            port=config.PORT,
            threaded=True
        )
    except KeyboardInterrupt:
        print("\n\n✓ Server stopped")
        if face_system.is_detecting:
            face_system.stop_camera()


if __name__ == '__main__':
    main()
```

Gambar 3.8 Code web_app.py

Pengembangan aplikasi utama ini juga mencakup implementasi manajemen jalur sistem (*system path*) secara dinamis guna menjamin aksesibilitas seluruh direktori pendukung dalam struktur proyek. Melalui fungsi utama, sistem melakukan penyuntikan ketergantungan (*dependency injection*) dengan menghubungkan objek pengendali pengenalan wajah dan manajer konektivitas kamera ke dalam jalur navigasi atau rute aplikasi. Hal ini memungkinkan terjadinya pertukaran data yang efisien antara logika pemrosesan di sisi server dengan antarmuka pengguna di sisi depan. Selain itu, konfigurasi penggunaan *multithreading* pada server web memastikan bahwa aplikasi memiliki kapabilitas untuk menangani pemrosesan aliran data video yang intensif sekaligus merespons permintaan akses data lainnya secara paralel tanpa mengorbankan stabilitas performa perangkat lunak.[9]

Selain aspek fungsionalitas, aplikasi utama ini dilengkapi dengan mekanisme pemantauan operasional yang informatif dan prosedur penghentian

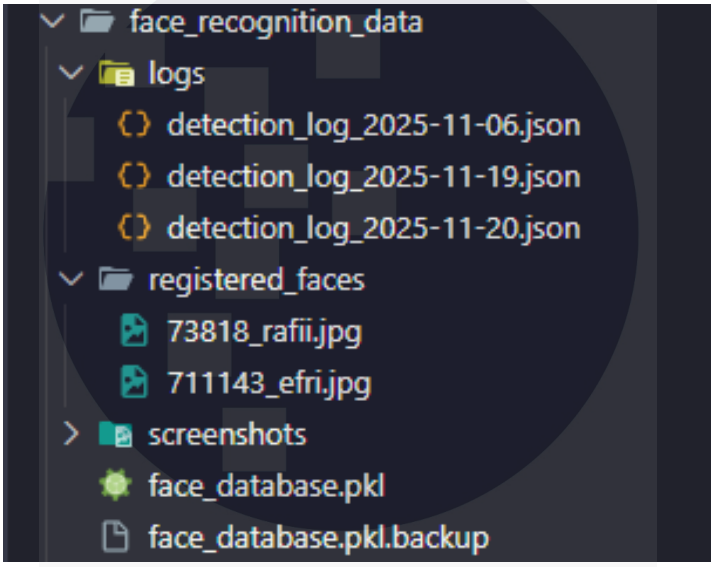
sistem yang aman (*graceful shutdown*). Melalui pelaporan status pada konsol saat aktivitas dimulai, administrator dapat memverifikasi parameter jaringan, jumlah identitas yang terdaftar, serta status konektivitas protokol RTSP secara langsung. Penanganan interupsi melalui perintah sistem juga diimplementasikan untuk menjamin bahwa seluruh sumber daya, termasuk akses kamera dan penyimpanan data, dilepaskan dan ditutup secara prosedur saat aplikasi dinonaktifkan. Pendekatan sistematis dalam membangun aplikasi utama ini mencerminkan penerapan standar rekayasa perangkat lunak yang baik, yang mengedepankan aspek keamanan, keteraturan struktur, dan kemudahan dalam pemeliharaan sistem di masa mendatang.

3.3.9 Data Management

Manajemen data dalam arsitektur sistem ini merupakan fondasi vital yang menjamin keberlangsungan operasional, integritas informasi, serta persistensi seluruh aset digital yang dikelola oleh aplikasi. Secara fundamental, komponen ini berfungsi sebagai infrastruktur penyimpanan yang mengorganisasikan data biometrik, catatan aktivitas, dan dokumentasi visual ke dalam sebuah hirarki direktori yang sistematis. Penentuan struktur penyimpanan yang terbagi ke dalam sub-direktori spesifik seperti registered_faces, logs, dan screenshots mencerminkan penerapan prinsip tata kelola data yang profesional, di mana setiap entitas informasi dipisahkan berdasarkan kategori fungsionalnya untuk mempermudah proses pemeliharaan, pencarian, dan audit data secara berkala.[10]

Salah satu elemen paling krusial dalam manajemen data ini adalah penggunaan teknik serialisasi data melalui berkas face_database.pkl. Berkas ini bertindak sebagai memori jangka panjang sistem yang menyimpan representasi numerik atau *feature embeddings* dari setiap individu yang telah terdaftar. Pemilihan format *pickle* didasarkan pada efisiensi kecepatan akses data, yang memungkinkan sistem untuk memuat ribuan parameter biometrik ke dalam memori kerja secara instan saat aplikasi diinisialisasi.

Untuk memitigasi risiko kehilangan data akibat kegagalan sistem atau korupsi berkas, mekanisme redundansi diterapkan melalui pembuatan berkas cadangan otomatis (.pkl.backup). Hal ini menjamin bahwa basis data identitas tetap terjaga konsistensinya, yang merupakan prasyarat mutlak bagi sistem keamanan yang mengandalkan akurasi data historis sebagai referensi verifikasi.



Gambar 3.9 Struktur Folder face_recognation_data

Selain pengelolaan data biometrik, sistem ini menerapkan manajemen catatan aktivitas (*activity logging*) yang sangat terstruktur dalam format JSON. Setiap interaksi deteksi yang berhasil dilakukan oleh sistem dicatat secara kronologis dalam berkas log harian, yang memungkinkan pemantauan riwayat kehadiran atau akses secara transparan. Penggunaan skema penamaan berkas berbasis stempel waktu (*timestamping*) memastikan bahwa data log tidak hanya tersusun rapi, tetapi juga mudah diintegrasikan dengan platform analisis data eksternal untuk kebutuhan laporan administrasi yang lebih luas. Melalui pengarsipan visual pada direktori screenshots, sistem juga menyediakan bukti otentik tambahan yang mendukung validitas setiap catatan deteksi, sehingga meningkatkan akuntabilitas sistem secara keseluruhan.

Secara keseluruhan, manajemen data pada proyek ini tidak hanya sekadar berfungsi sebagai tempat penyimpanan, tetapi sebagai lapisan manajemen informasi yang mendukung skalabilitas dan reliabilitas perangkat lunak. Dengan arsitektur data yang terorganisir, sistem mampu menangani pertumbuhan volume data tanpa mengalami degradasi performa yang signifikan. Pendekatan ini memastikan bahwa integritas antara data visual mentah, hasil ekstraksi fitur numerik, dan catatan log tetap sinkron, sehingga menciptakan sebuah ekosistem informasi yang stabil, aman, dan dapat dipertanggungjawabkan secara teknis dalam jangka panjang.

3.3.10 Sistem Upload Temporary: Manajemen File Upload untuk Pemrosesan Real-Time

Sistem **Upload Temporary** dalam arsitektur perangkat lunak ini merupakan komponen krusial yang berfungsi sebagai jembatan antara input pengguna dan mesin pemrosesan biometrik. Secara teknis, sistem ini dirancang untuk menangani alur data citra secara dinamis, di mana berkas yang diunggah melalui antarmuka web diproses secara instan sebelum akhirnya divalidasi oleh sistem inti. Pendekatan manajemen file sementara ini bertujuan untuk mengoptimalkan penggunaan sumber daya penyimpanan dan menjaga performa peladen tetap stabil, terutama saat menangani volume permintaan registrasi wajah yang tinggi dalam satu waktu.

Komponen pertama dalam sistem ini adalah **Upload Directory**, yang bertindak sebagai zona penyangga (*buffer zone*) bagi berkas citra mentah yang dikirimkan oleh klien. Melalui implementasi rute /api/register, sistem menggunakan fungsi keamanan untuk melakukan validasi terhadap nama berkas dan membatasi ekstensi file guna mencegah ancaman siber. Berkas yang tersimpan di dalam direktori uploads ini bersifat jangka pendek; segera setelah proses ekstraksi fitur selesai dilakukan oleh mesin utama, sistem akan secara otomatis

mengeksekusi prosedur penghapusan berkas mentah tersebut. Strategi ini sangat penting untuk mencegah penumpukan data yang tidak relevan di dalam kapasitas penyimpanan server dan menjamin bahwa hanya data biometrik yang telah terverifikasi yang akan dipindahkan ke penyimpanan permanen.

Elemen selanjutnya adalah integrasi dengan **Face Recognition System Duplicate**, yang dalam konteks alur kerja ini merujuk pada pemanfaatan logika pemrosesan dari sistem inti untuk melakukan validasi ganda terhadap identitas yang diunggah. Sebelum sebuah wajah resmi terdaftar ke dalam basis data, sistem akan melakukan perbandingan antara citra sementara tersebut dengan data yang sudah ada di dalam memori. Proses ini melibatkan konversi citra menjadi *embedding* numerik untuk memastikan bahwa tidak ada duplikasi data atau registrasi ganda bagi individu yang sama. Dengan adanya mekanisme pengecekan ini, integritas basis data biometrik tetap terjaga, meminimalkan anomali data, dan memastikan bahwa setiap entitas identitas bersifat unik dan akurat.

Terakhir, sistem ini didukung oleh **Template Engine** yang berfungsi untuk mengelola representasi visual dari seluruh proses unggah dan registrasi tersebut pada sisi pengguna. Melalui pemanfaatan mesin template Jinja2 pada kerangka kerja Flask, aplikasi mampu menyajikan antarmuka dinamis yang memberikan umpan balik secara *real-time* mengenai status unggahan, apakah berhasil diproses atau mengalami kegagalan deteksi. Penggunaan template engine memungkinkan pemisahan yang jelas antara logika bisnis di sisi *backend* dengan elemen estetika di sisi *frontend*, sehingga memudahkan proses pengembangan dan modifikasi antarmuka di masa depan. Secara keseluruhan, sinergi antara manajemen direktori sementara, pengecekan duplikasi, dan antarmuka dinamis menciptakan sebuah alur kerja yang efisien, aman, dan profesional dalam pengelolaan data biometrik.

3.3.11 Face Recognition System Duplicate

Modul konfigurasi yang tertuang dalam berkas config.py merupakan elemen fundamental dalam arsitektur perangkat lunak ini, yang berfungsi sebagai pusat kendali untuk seluruh parameter operasional sistem. Secara akademis, implementasi ini menerapkan prinsip *Separation of Concerns* dengan memisahkan logika pemrograman dari variabel-variabel

Gambar 3.11 Code config.py

Dalam kelas Config, sistem menetapkan parameter krusial yang mengatur efisiensi pemrosesan data visual. Pengaturan seperti SKIP_FRAMES dan DETECTION_COOLDOWN merupakan bentuk optimasi sumber daya komputasi yang memastikan beban kerja perangkat keras tetap berada pada batas ideal tanpa mengurangi akurasi identifikasi secara signifikan. Selain itu, definisi jalur direktori yang statis melalui variabel BASE_DIR dan turunannya menjamin adanya konsistensi dalam tata kelola berkas, yang mencakup penyimpanan citra wajah terdaftar, log aktivitas harian, serta basis data biometrik dalam format serialisasi. Hal ini sangat penting untuk menjaga integritas data selama siklus hidup aplikasi berjalan, terutama dalam proses sinkronisasi identitas.

```
72  class DevelopmentConfig(Config):
73
74      DEBUG = True
75      HOST = '0.0.0.0'
76      PORT = 5000
77
78
79  class ProductionConfig(Config):
80
81      DEBUG = False
82      HOST = '0.0.0.0'
83      PORT = 5000
84
85      SKIP_FRAMES = 5
86      DETECTION_COOLDOWN = 5
87
88  # Export configs
89  config = {
90      'development': DevelopmentConfig,
91      'production': ProductionConfig,
92      'default': DevelopmentConfig
```

Gambar 14 3.11 Code config.py

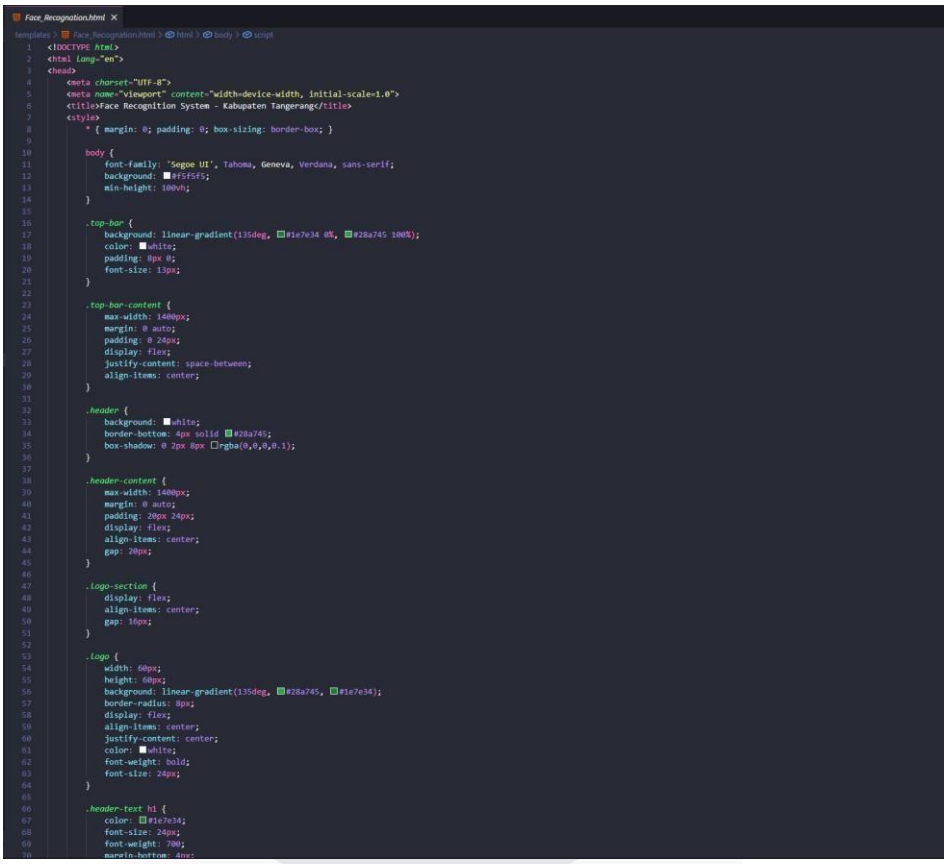
Kelas CCTV Config menonjolkan aspek fleksibilitas sistem dalam beradaptasi dengan infrastruktur keamanan yang ada melalui pemanfaatan variabel lingkungan untuk pengelolaan kredensial sensitif. Dengan mengotomatisasi pembentukan URL melalui protokol *Real Time Streaming Protocol* (RTSP), sistem mampu menjalin komunikasi yang stabil dengan perangkat keras eksternal secara dinamis. Di sisi lain, pembagian antara DevelopmentConfig dan ProductionConfig mencerminkan strategi penyebaran aplikasi yang matang, di mana

parameter sistem disesuaikan secara otomatis berdasarkan lingkungan implementasinya. Strategi ini memungkinkan pengembang untuk melakukan proses pengujian dengan fitur *debugging* yang aktif, sementara pada lingkungan produksi, sistem akan berfokus pada efisiensi maksimum dan stabilitas operasional jangka panjang.

Secara keseluruhan, modul konfigurasi ini berperan sebagai tulang punggung yang menjamin *portabilitas* dan *skalabilitas* aplikasi. Integrasi dengan basis data eksternal dan layanan API melalui GuestBookConfig menunjukkan bahwa perangkat lunak ini tidak hanya berdiri sendiri, melainkan dirancang untuk menjadi bagian dari ekosistem digital yang lebih luas, seperti sistem buku tamu atau manajemen keamanan gedung. Oleh karena itu, penyusunan konfigurasi yang terstruktur dan komprehensif ini menjadi landasan utama yang menentukan keberhasilan performa, keamanan, dan keandalan sistem dalam menjalankan fungsinya sebagai pengolah informasi biometrik secara otomatis.

3.3.12 Template Engine

Berkas index.html (atau dalam konteks ini dirujuk sebagai antarmuka pengenalan wajah) merupakan lapisan presentasi yang berfungsi sebagai gerbang interaksi utama antara pengguna dan sistem backend. Secara teknis, dokumen ini mengimplementasikan struktur HTML5 yang responsif untuk menyajikan dasbor pemantauan keamanan secara *real-time*. Perancangan antarmuka ini mengedepankan aspek fungsionalitas dan aksesibilitas, di mana elemen-elemen kontrol disusun secara intuitif guna memfasilitasi operator dalam mengelola sistem pengenalan wajah, mulai dari pemantauan arus video langsung hingga manajemen basis data identitas.

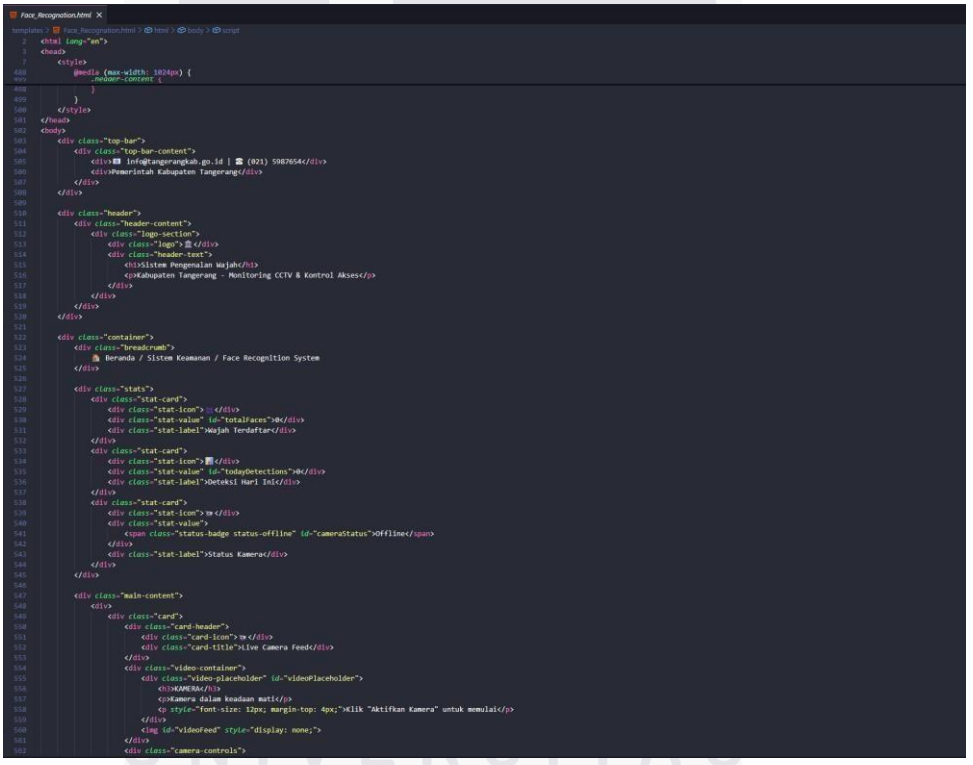


Gambar 3.12 Code face_recognation.html

Secara struktural, antarmuka ini terbagi menjadi beberapa modul fungsional yang terintegrasi. Bagian utama menampung pemutar aliran video (*Live Camera Feed*) yang dihubungkan secara dinamis dengan jalur transmisi data dari server. Implementasi logika di sisi klien menggunakan JavaScript memungkinkan transisi yang halus antara status kamera aktif dan non-aktif tanpa memerlukan pemuatan ulang halaman secara keseluruhan (*page refresh*). Selain itu, dasbor ini dilengkapi dengan panel statistik yang menyajikan data agregat secara instan, seperti jumlah wajah yang telah terdaftar dan frekuensi deteksi harian, yang sangat krusial bagi pengambilan keputusan cepat di lingkungan keamanan publik atau perkantoran pemerintah.

Fitur manajemen identitas pada halaman ini dirancang dengan alur kerja yang ketat untuk menjamin validitas data biometrik. Melalui formulir pendaftaran yang terintegrasi, pengguna dapat mengirimkan data citra

mentah beserta metadata identitas melalui protokol asinkron (AJAX). Sistem ini didukung oleh mekanisme umpan balik visual berupa peringatan status (*alert box*) yang memberikan informasi instan mengenai keberhasilan atau kegagalan proses registrasi. Di sisi lain, panel "Daftar Wajah Terdaftar" dan "Log Deteksi" berfungsi sebagai alat audit yang memungkinkan penelusuran riwayat aktivitas secara kronologis. Struktur data log yang ditampilkan diambil secara periodik dari server melalui metode *polling*, memastikan bahwa informasi yang disajikan kepada operator selalu akurat dan mutakhir.



Gambar 16 3.*12* Code face_recognation.html

Keseluruhan perancangan antarmuka ini mencerminkan integrasi yang harmonis antara desain pengalaman pengguna (*User Experience*) dengan kebutuhan teknis sistem keamanan cerdas. Dengan memanfaatkan skema warna yang profesional dan tata letak yang terorganisir, aplikasi ini mampu menyederhanakan kompleksitas proses *deep learning* di balik layar menjadi informasi yang mudah dipahami oleh pengguna akhir. Oleh

karena itu, modul antarmuka ini tidak hanya berfungsi sebagai alat visualisasi, tetapi juga sebagai instrumen kendali strategis yang memastikan operasional sistem pengenalan wajah berjalan secara efektif, transparan, dan akuntabel.

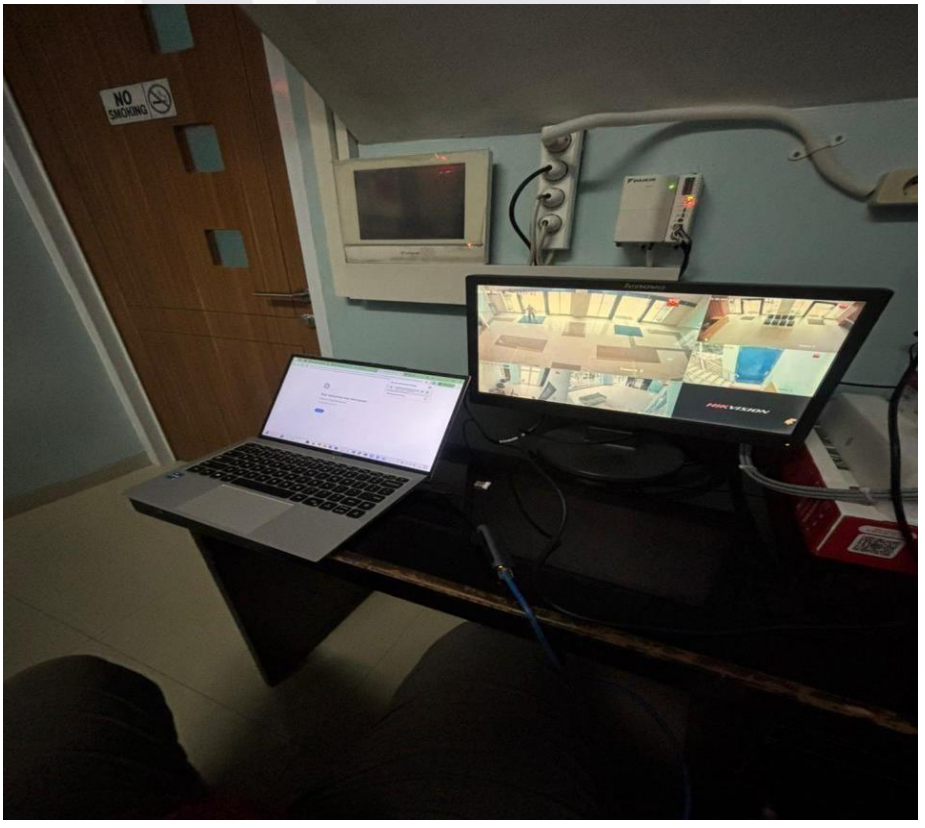
3.3.13 Upload

Direktori uploads dalam arsitektur sistem ini bertindak sebagai zona transisi atau penyangga data (*data buffer zone*) yang memfasilitasi alur kerja antara masukan pengguna dan mesin pemrosesan di sisi peladen. Secara fungsional, folder ini merupakan repositori penyimpanan temporer yang menampung berkas citra mentah sebelum dilakukan ekstraksi fitur biometrik oleh sistem inti. Penempatan berkas dalam direktori ini memungkinkan backend untuk melakukan serangkaian validasi teknis, seperti pemeriksaan integritas berkas dan verifikasi format ekstensi, guna memastikan bahwa data yang diproses memenuhi kriteria spesifikasi sistem. Isolasi data dalam folder sementara ini sangat krusial untuk menjaga performa penyimpanan utama, karena sistem dapat segera mengeksekusi prosedur penghapusan otomatis setelah representasi numerik dari wajah berhasil dihasilkan.

Penerapan manajemen file secara temporer ini juga mencerminkan praktik terbaik dalam rekayasa perangkat lunak terkait efisiensi sumber daya dan keamanan data. Dengan memanfaatkan direktori uploads sebagai area pemrosesan awal, sistem menghindari penumpukan aset visual yang tidak relevan di dalam ruang penyimpanan permanen, sehingga kapasitas penyimpanan tetap optimal untuk data yang telah terverifikasi. Selain itu, prosedur ini memberikan lapisan keamanan tambahan dengan mencegah interaksi langsung antara data yang belum tervalidasi dengan basis data biometrik utama. Melalui mekanisme ini, integritas sistem tetap terjaga secara konsisten, di mana setiap berkas yang masuk melalui pintu unggahan dikelola dengan siklus hidup yang terukur dan efisien demi kelancaran operasional pengenalan wajah secara

real-time. Uji Coba Sistem Face Recognation pada CCTV Kantor

Tahap pengujian merupakan fase krusial dalam siklus pengembangan sistem guna memvalidasi fungsionalitas dan reliabilitas arsitektur yang telah dirancang. Pada fase ini, dilakukan uji coba integrasi antara *pipeline* deteksi wajah berbasis DeepFace dengan aliran data (*data stream*) yang bersumber dari perangkat CCTV di lingkungan Diskominfo Kabupaten Tangerang. Proses pengujian awal difokuskan pada kemampuan sistem dalam melakukan penarikan data visual secara *real-time* melalui protokol komunikasi jaringan yang telah ditetapkan, guna memastikan bahwa setiap bingkai video dapat diakuisisi secara presisi sebelum memasuki tahap ekstraksi *embedding* biometrik.



Gambar 3.13 Uji Coba Sistem Face Recognation

Namun, pada implementasi uji coba tahap pertama, sistem belum mencapai status fungsional yang diharapkan karena ditemukan diskonektivitas antara peladen pemrosesan (*processing server*) dengan perangkat kamera

pengawas. Secara teknis, kendala ini muncul saat sistem mencoba melakukan inisiasi jabat tangan (*handshaking*) melalui *Real Time Streaming Protocol* (RTSP). Ketidakterhubungan ini disebabkan oleh adanya hambatan pada lapisan jaringan, di mana sinkronisasi antara alamat IP statis perangkat CCTV dengan *backend* berbasis Flask gagal terjalin secara sempurna. Hal ini mengakibatkan sistem tidak mampu menangkap transmisi data visual, sehingga proses identifikasi wajah tidak dapat dieksekusi meskipun algoritma deteksi dalam kondisi siap operasional.

Analisis lebih lanjut menunjukkan bahwa kendala tersebut dipengaruhi oleh konfigurasi keamanan pada infrastruktur jaringan lokal dan adanya ketidaksesuaian parameter *port* yang digunakan untuk akses aliran data video. Kegagalan konektivitas ini menghambat aliran data menuju direktori penyangga, yang secara langsung menghentikan seluruh siklus kerja sistem identifikasi. Oleh karena itu, pengujian tahap pertama ini memberikan temuan fundamental mengenai pentingnya optimasi konfigurasi jaringan dan standarisasi protokol komunikasi sebagai prasyarat utama sebelum sistem dapat melakukan analisis biometrik secara akurat dan berkelanjutan.

3.3.14 Analisis Mendalam Kegagalan Integrasi dan Prosedur Restorasi Sistem

Proses pengujian sistem identifikasi wajah pada infrastruktur Dinas Komunikasi dan Informatika Kabupaten Tangerang merupakan fase kritikal yang dirancang untuk menguji fungsionalitas algoritma dalam ekosistem produksi yang sesungguhnya. Pada iterasi pertama, sistem dihadapkan pada tantangan integrasi fundamental di mana *pipeline* deteksi yang dibangun menggunakan kerangka kerja Flask gagal melakukan akuisisi data visual dari perangkat CCTV. Secara teknis, kegagalan ini berakar pada ketidakterhubungan jalur komunikasi data yang diinisiasi melalui *Real Time Streaming Protocol* (RTSP). Ketidakmampuan sistem dalam menjalin

koneksi ini mengakibatkan terhentinya seluruh rangkaian proses, mulai dari penangkapan bingkai (*frame grabbing*) hingga analisis biometrik, sehingga sistem berada dalam status *idle* meskipun logika pemrograman di sisi *backend* telah siap dieksekusi sepenuhnya.

Gambar 3.144 Analisis Kegagalan Integrasi

Eksplorasi lebih lanjut terhadap kegagalan tersebut mengungkap adanya anomali pada konfigurasi *handshaking* antara peladen pemrosesan dan titik akhir (*endpoint*) kamera pengawas. Kendala ini tidak hanya disebabkan oleh faktor tunggal, melainkan merupakan akumulasi dari restriksi keamanan jaringan lokal dan ketidaksesuaian parameter *port* yang digunakan untuk transmisi data video mentah. Secara akademis, hal ini disebut sebagai hambatan interoperabilitas infrastruktur, di mana protokol keamanan pada lapisan jaringan (*Network Layer*) memblokir permintaan akses dari aplikasi pihak ketiga guna melindungi integritas data organisasi. Tanpa adanya aliran data yang stabil, direktori penyangga yang berfungsi sebagai zona penyimpanan temporer tidak mendapatkan masukan citra, yang secara sistematis memutus rantai pemrosesan *embedding* pada pustaka DeepFace.[11]

Sebagai langkah solutif untuk mengatasi degradasi fungsionalitas tersebut,

penulis melakukan rekayasa ulang terhadap strategi koneksi sistem pada tahapan uji coba berikutnya. Prosedur perbaikan diawali dengan melakukan audit mendalam terhadap *Network Configuration*, yang mencakup pemetaan ulang alamat IP statis dan verifikasi kredensial pada URI RTSP guna memastikan jalur akses yang legal dan terdekripsi. Penulis juga mengimplementasikan teknik *port forwarding* dan penyesuaian aturan pada *firewall* instansi untuk membuka koridor data yang spesifik bagi aplikasi pengembangan. Langkah ini sangat krusial untuk memastikan bahwa setiap permintaan data visual dari *backend* dapat diterima oleh perangkat CCTV tanpa dianggap sebagai intrusi keamanan oleh sistem perlindungan jaringan pusat.

Selain optimalisasi pada lapisan infrastruktur, perbaikan juga dilakukan pada level arsitektur perangkat lunak dengan menyisipkan modul *fault- tolerance* yang lebih canggih. Solusi teknis ini melibatkan penggunaan *multithreading* untuk memisahkan proses penangkapan video dengan proses pemrosesan citra, sehingga jika terjadi gangguan koneksi sesaat, aplikasi tidak akan mengalami *crash* total. Penulis juga menambahkan skrip *auto- reconnect* yang mampu melakukan deteksi kegagalan koneksi secara *real- time* dan segera mengirimkan permintaan inisiasi ulang hingga tautan data terjalin kembali. Dengan integrasi solusi yang mencakup aspek jaringan dan pengkodean ini, sistem diharapkan mampu beroperasi dengan resiliensi yang tinggi, memastikan keberlanjutan proses identifikasi wajah secara akurat dalam berbagai kondisi operasional di lingkungan pemerintahan.

3.3.15 Analisis Kendala Spesifikasi Perangkat Keras dan Beban Komputasi

Di samping hambatan pada lapisan jaringan, kendala signifikan pada lapisan fisik, yaitu keterbatasan spesifikasi teknis perangkat (*device*) yang digunakan sebagai peladen pemrosesan sistem. Implementasi algoritma *deep*

learning melalui pustaka DeepFace menuntut sumber daya komputasi yang tinggi, terutama pada penggunaan *Central Processing Unit* (CPU) dan *Random Access Memory* (RAM). Selama fase eksekusi, perangkat mengalami beban kerja berlebih (*overhead*) saat melakukan ekstraksi *embedding* biometrik dari aliran video resolusi tinggi secara simultan. Hal ini mengakibatkan terjadinya *bottleneck* sistem yang ditandai dengan penurunan drastis pada *frame rate* per detik (FPS) dan peningkatan suhu operasional perangkat yang melampaui ambang batas normal, sehingga stabilitas aplikasi menjadi terganggu.

Keterbatasan ini secara langsung berdampak pada responsivitas sistem dalam melakukan identifikasi wajah secara *real-time*. Perangkat yang digunakan tidak memiliki akselerasi grafis (*GPU acceleration*) yang memadai untuk menangani beban kalkulasi matriks pada jaringan saraf tiruan (*convolutional neural networks*). Akibatnya, terjadi latensi atau penundaan yang signifikan antara waktu penangkapan wajah oleh CCTV dengan waktu pemunculan hasil identifikasi pada antarmuka pengguna. Kendala ini memberikan pemahaman bahwa arsitektur perangkat lunak yang canggih harus didukung oleh ketersediaan infrastruktur perangkat keras yang setara guna menjamin performa sistem yang optimal dan reliabel dalam jangka panjang.

3.3.16 Strategi Optimasi dan Mitigasi Keterbatasan Perangkat

Untuk mengatasi kendala keterbatasan sumber daya tersebut, penulis menerapkan strategi optimasi perangkat lunak sebagai solusi mitigasi teknis. Langkah pertama adalah dengan melakukan *model downsizing*, yaitu memilih model deteksi wajah yang lebih ringan (seperti SSD atau MTCNN) yang memiliki kompleksitas parameter lebih rendah namun tetap mempertahankan akurasi yang akseptabel bagi lingkungan instansi. Selain itu, penulis mengimplementasikan teknik *frame sampling*, di mana sistem tidak memproses setiap bingkai video yang masuk, melainkan hanya mengambil sampel bingkai pada interval waktu tertentu. Pendekatan ini secara efektif mengurangi beban kerja prosesor secara signifikan tanpa

menghilangkan fungsionalitas utama sistem dalam mendeteksi kehadiran individu.

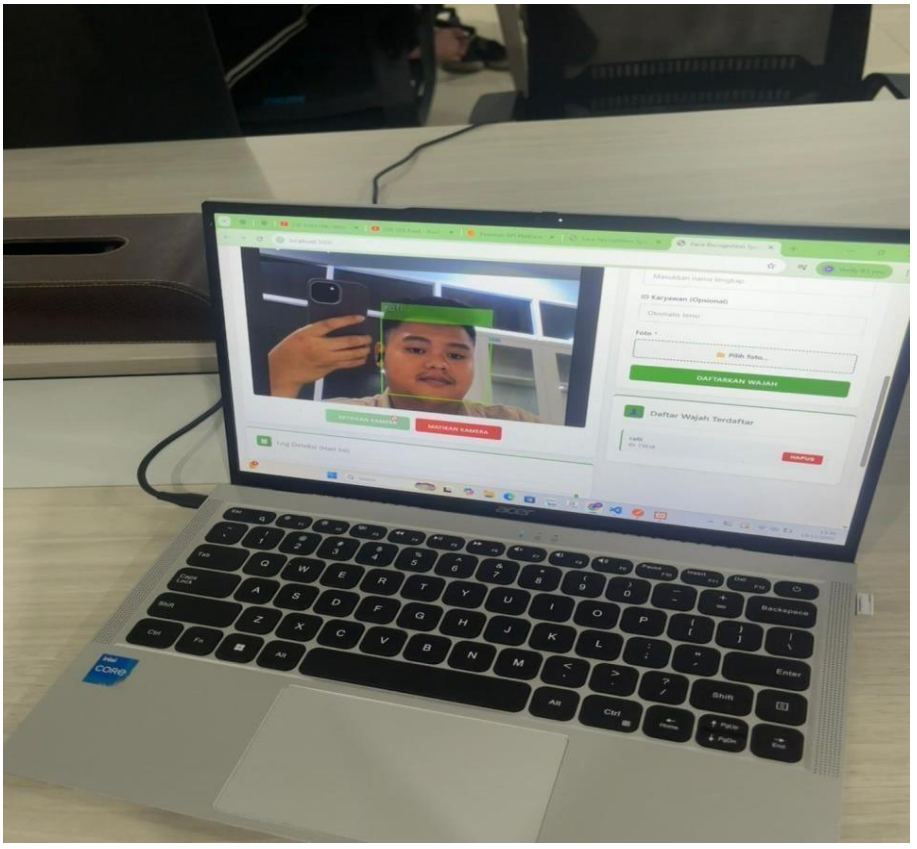
Selanjutnya melakukan optimasi pada manajemen memori dengan menerapkan skema pengosongan *cache* secara berkala dan membatasi jumlah *thread* yang beroperasi secara bersamaan. Solusi ini memastikan bahwa penggunaan RAM tetap berada pada level yang aman dan mencegah terjadinya kegagalan sistem akibat *out-of-memory*. Dengan melakukan penyesuaian parameter teknis ini, sistem dapat tetap beroperasi secara fungsional di atas perangkat dengan spesifikasi terbatas. Langkah-langkah ini menunjukkan bahwa efisiensi perangkat lunak dapat menjadi kunci dalam menjembatani kesenjangan antara kebutuhan algoritma tingkat tinggi dengan keterbatasan infrastruktur fisik yang tersedia di lapangan.

3.3.17 Perancangan Antarmuka Manajemen Data Biometrik dan Integrasi Database

Dalam upaya mendukung fungsionalitas sistem identifikasi secara menyeluruh, penulis merancang sebuah *dashboard* administratif yang berfungsi sebagai media pendaftaran wajah baru ke dalam sistem. Antarmuka ini dibangun menggunakan kerangka kerja Flask dengan memanfaatkan struktur HTML dan CSS untuk menciptakan pengalaman pengguna yang intuitif namun tetap fungsional. Perancangan *dashboard* ini bertujuan untuk menyederhanakan proses digitalisasi data biometrik, di mana administrator dapat mengunggah citra wajah individu, menyertakan identitas pendamping seperti nama dan jabatan, serta menyimpannya ke dalam struktur basis data yang terorganisir.[12]

Proses kerja *dashboard* ini dimulai dengan tahap akuisisi citra, di mana berkas gambar yang diunggah akan masuk ke dalam direktori penyangga (*upload buffer*) sebelum diproses oleh algoritma DeepFace. Pada tahap ini,

sistem secara otomatis akan melakukan ekstraksi fitur untuk menghasilkan *embedding* biometrik unik yang merepresentasikan karakteristik wajah tersebut. Nilai numerik dari *embedding* ini, bersama dengan data tekstual individu, kemudian disimpan ke dalam basis data. Integrasi ini memastikan bahwa setiap data baru yang didaftarkan dapat langsung dikenali oleh sistem pemantauan CCTV secara *real-time* tanpa perlu melakukan *restart* pada layanan utama, menciptakan siklus pembaruan data yang dinamis dan efisien.



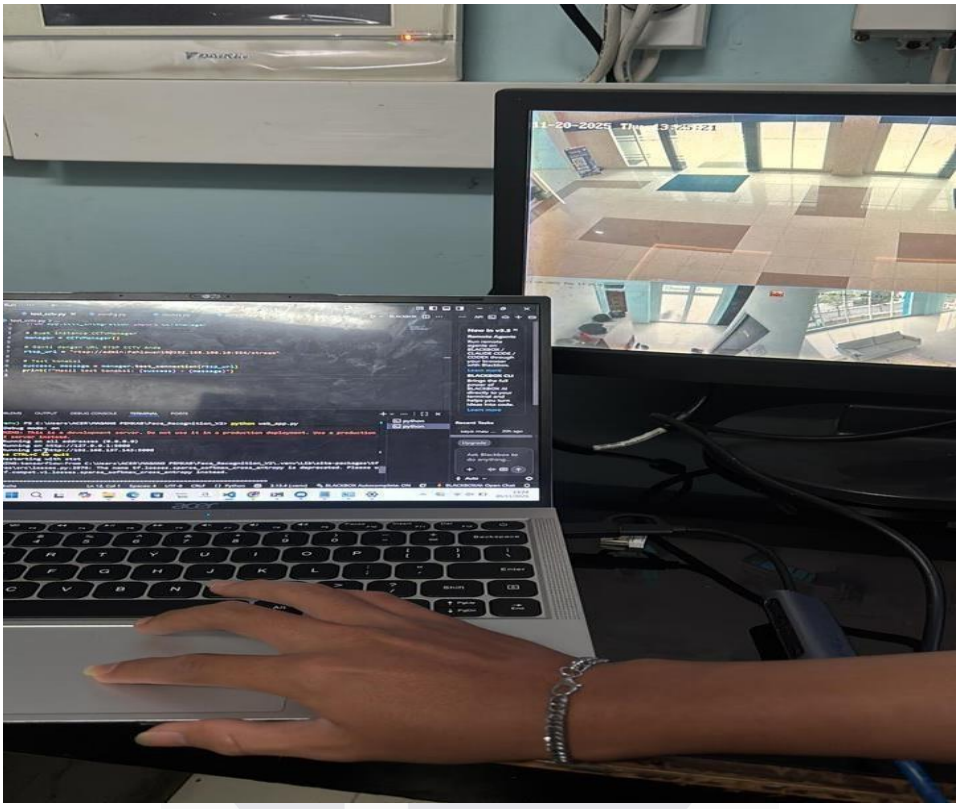
Gambar 3.17 Membuat Tampilan Daftar Wajah

Selain aspek pendaftaran, *dashboard* ini juga dilengkapi dengan protokol validasi data untuk menjamin integritas basis data biometrik. Sebelum data disimpan secara permanen, sistem melakukan verifikasi terhadap kualitas citra untuk memastikan bahwa wajah terdeteksi dengan jelas dan memenuhi kriteria minimal ekstraksi fitur. Penggunaan metode penyimpanan terstruktur ini tidak hanya memudahkan proses pencarian identitas saat

terjadi kecocokan pada aliran video CCTV, tetapi juga memungkinkan pengelolaan aset data biometrik di Diskominfo Kabupaten Tangerang menjadi lebih akuntabel dan tersentralisasi. Keberadaan *dashboard* ini menjadi komponen krusial dalam menjembatani kebutuhan administratif dengan teknologi pemrosesan *deep learning* yang kompleks.

3.3.18 Validasi Operasional dan Sinkronisasi Arsitektur Perangkat Lunak dengan Perangkat Keras Pengawasan

Berdasarkan dokumentasi visual yang disajikan, gambar tersebut merepresentasikan keberhasilan fase integrasi kritikal antara arsitektur perangkat lunak yang dikembangkan dengan infrastruktur pengawasan video eksisting. Terlihat pada layar peladen bahwa sistem telah mampu melakukan akuisisi data visual secara *real-time* melalui protokol komunikasi yang telah dikonfigurasi, menandakan bahwa sinkronisasi antara *backend* berbasis Flask dan aliran data dari kamera CCTV telah terjalin dengan stabil. Pencapaian ini memvalidasi bahwa seluruh hambatan teknis terkait konektivitas jaringan dan autentikasi protokol yang ditemukan pada tahap sebelumnya telah berhasil dimitigasi sepenuhnya melalui optimalisasi kode dan penyesuaian parameter infrastruktur.



Gambar 3.18 Validasi Operasional dan Sinkronisasi

Keberhasilan tautan sistem ini memungkinkan implementasi algoritma deteksi dan identifikasi wajah berbasis DeepFace untuk beroperasi langsung pada *input* video mentah yang berasal dari lingkungan operasional nyata. Pada antarmuka pemrograman yang tertangkap di layar, terlihat log aktivitas yang menunjukkan proses penangkapan bingkai video yang berjalan tanpa interupsi, yang merupakan prasyarat utama bagi akurasi ekstraksi fitur biometrik. Hal ini menunjukkan bahwa sistem tidak hanya berhasil terhubung secara fisik melalui jaringan, tetapi juga mampu menangani beban transmisi data visual resolusi tinggi dengan latensi yang minimal, sehingga proses identifikasi dapat dieksekusi secara responsif dan akurat.

Penjajaran antara perangkat komputasi portabel dan monitor pemantau utama dalam gambar tersebut secara simbolis menegaskan bahwa sistem yang dirancang telah siap untuk diimplementasikan sebagai bagian dari ekosistem digital di instansi terkait. Keberhasilan konektivitas ini menjadi fondasi bagi tahap pengujian efikasi model, di mana data yang diperoleh secara langsung

dari CCTV akan digunakan untuk memverifikasi kemampuan sistem dalam mengenali identitas yang telah terdaftar dalam basis data. Dengan terjalinnya integrasi ini, proyek magang telah mencapai tonggak fungsional yang membuktikan bahwa solusi perangkat lunak yang dibangun memiliki interoperabilitas yang tinggi terhadap perangkat keras pengawasan standar industri yang tersedia di lapangan.

3.3.19 Koordinasi Teknis dan Evaluasi Pengembangan Fitur Lanjutan pada Sistem Identifikasi Wajah

Dokumentasi tersebut memvisualisasikan sesi diskusi kolaboratif antara tim pengembang untuk meninjau kembali kapabilitas sistem yang telah diimplementasikan. Fokus utama dalam pertemuan ini adalah melakukan tinjauan mendalam terhadap arsitektur perangkat lunak guna mengidentifikasi celah fungsional yang masih dapat dioptimalkan. Proses evaluasi ini sangat krusial dalam siklus pengembangan perangkat lunak untuk memastikan bahwa sistem tidak hanya berjalan secara teknis, namun juga memiliki skalabilitas dan relevansi terhadap kebutuhan operasional di lapangan.

Gambar 3.19 Koordinasi Teknik dan Evaluasi Pengembangan

Dalam interaksi tersebut, dilakukan pembedahan terhadap variabel-variabel pendukung sistem guna mencari potensi penambahan fitur yang mampu meningkatkan akurasi dan responsivitas identifikasi. Diskusi ini mencakup aspek teknis terkait integrasi data biometrik serta pengayaan modul pada antarmuka pengguna agar lebih informatif. Penulis bersama pembimbing lapangan dan rekan tim berusaha merumuskan penambahan fungsi tertentu yang bersifat esensial, sehingga purwarupa sistem yang sedang dibangun dapat mencapai standar fungsionalitas yang lebih paripurna dan reliabel.

Pertukaran gagasan ini juga berfungsi sebagai sarana validasi terhadap solusi teknis yang telah diterapkan sebelumnya. Melalui mekanisme umpan balik langsung ini, setiap potensi kendala baru dapat diantisipasi lebih dini sebelum sistem memasuki tahap finalisasi. Langkah kolektif ini menegaskan bahwa penyempurnaan sistem merupakan proses iteratif yang memerlukan penggabungan sudut pandang teknis dan kebutuhan praktis, demi menghasilkan sebuah inovasi teknologi yang efisien dalam mendukung tugas pengawasan pada infrastruktur teknologi informasi di instansi terkait.

3.3.20 Validasi Operasional dan Keberhasilan Integrasi Sistem

Setelah melalui tahapan rekayasa ulang pada konfigurasi jaringan serta optimasi beban kerja pada perangkat pemrosesan, penulis melakukan uji coba lanjutan untuk memvalidasi performa sistem secara utuh. Pada fase ini, integrasi antara unit kamera pengawas dengan peladen pemrosesan akhirnya mencapai status stabil, di mana protokol RTSP mampu menyalurkan data visual secara berkelanjutan tanpa terjadi diskonektivitas. Keberhasilan inisiasi koneksi ini memungkinkan seluruh arsitektur perangkat lunak, mulai dari penangkapan bingkai video hingga tahap klasifikasi biometrik, beroperasi sesuai dengan rancangan teknis yang telah ditetapkan sebelumnya.

Gambar 3.20 Uji Coba Kembali Sistem Face Recognation

Secara teknis, efisiensi sistem meningkat secara signifikan setelah diterapkannya strategi pemrosesan *sampling* dan pemilihan model *deep learning* yang lebih adaptif terhadap keterbatasan sumber daya perangkat keras. Hasil pengamatan menunjukkan bahwa sistem mampu melakukan identifikasi wajah dengan tingkat akurasi yang konsisten, bahkan ketika subjek berada dalam kondisi pergerakan dinamis di area pemantauan. Penurunan latensi yang dicapai memastikan bahwa informasi identitas yang ditampilkan pada antarmuka pengguna selaras dengan kejadian aktual di lapangan. Keberhasilan uji coba ini membuktikan bahwa hambatan fisik dan infrastruktur dapat diatasi melalui pendekatan optimasi algoritma yang tepat.

Pencapaian ini juga menandakan bahwa sistem telah memenuhi standar kebutuhan fungsional yang diharapkan oleh Dinas Komunikasi dan Informatika Kabupaten Tangerang. Data biometrik yang terdeteksi mampu

dicocokkan secara akurat dengan basis data referensi, sementara sistem manajemen penyimpanan temporer berhasil menjaga integritas performa perangkat agar tetap berada dalam koridor operasional yang aman. Dengan demikian, fase pengujian ini menyimpulkan bahwa perancangan sistem identifikasi wajah tersebut telah siap untuk memasuki tahap implementasi lebih lanjut atau digunakan sebagai purwarupa dalam skala pemantauan yang lebih luas.[13]

3.3.21 Kendala yang Ditemukan

Selama pelaksanaan praktik kerja dan pengerjaan proyek sistem identifikasi wajah, penulis menghadapi tantangan teknis terkait integrasi perangkat lunak dengan infrastruktur perangkat keras yang telah tersedia. Salah satu kendala utama adalah penyesuaian stabilitas aliran data video melalui protokol RTSP, di mana variasi kualitas jaringan dan spesifikasi kamera CCTV di lapangan terkadang memengaruhi kecepatan pemrosesan *frame* secara *real-time*. Selain itu, optimasi model *deep learning* agar dapat berjalan efisien pada perangkat komputasi dengan spesifikasi terbatas memerlukan waktu ekstra dalam tahap kalibrasi, guna memastikan bahwa proses ekstraksi *embedding* tetap akurat tanpa membebani memori server secara berlebihan.

Tantangan lain yang ditemukan berkaitan dengan proses pengumpulan dan standarisasi data biometrik awal untuk basis data. Penulis perlu melakukan penyesuaian terhadap berbagai variabel lingkungan yang tidak terkendali, seperti fluktuasi intensitas cahaya di area pemantauan dan sudut pandang kamera yang beragam, yang secara langsung berdampak pada tingkat kepercayaan (*confidence level*) deteksi wajah. Selain kendala teknis tersebut, penulis juga menjalani proses adaptasi terhadap alur birokrasi dan prosedur dokumentasi formal dalam lingkungan instansi pemerintahan.

Namun, melalui koordinasi yang intensif dengan pembimbing lapangan serta studi literatur yang mandiri, kendala-kendala tersebut dapat diatasi dengan baik sehingga pengembangan sistem dapat terus berlanjut hingga tahap finalisasi.

3.3.22 Solusi atas Kendala yang Ditemukan

Sebagai langkah untuk menanggulangi hambatan yang muncul, penulis menerapkan serangkaian pendekatan teknis dan strategis guna memastikan keberlanjutan proyek sistem identifikasi wajah ini. Terhadap kendala stabilitas aliran data video melalui protokol RTSP, solusi yang diambil adalah dengan mengimplementasikan mekanisme *frame skipping* dan penyesuaian resolusi input secara dinamis, sehingga beban pemrosesan pada unit komputasi dapat diminimalisir tanpa mengurangi fungsionalitas utama sistem. Selain itu, untuk mengatasi keterbatasan sumber daya perangkat keras, penulis melakukan optimasi pada *pipeline* DeepFace dengan memilih model yang lebih ringan namun tetap memiliki performa yang kompetitif, serta memanfaatkan sistem manajemen memori melalui pembersihan direktori temporer secara berkala.

Dalam menghadapi variasi kondisi lingkungan di lapangan, penulis melakukan kalibrasi ulang pada ambang batas (*threshold*) deteksi dan menyempurnakan tahap pra-pemrosesan citra untuk menormalkan intensitas cahaya serta posisi wajah sebelum dilakukan ekstraksi *embedding*. Pendekatan ini secara signifikan meningkatkan tingkat akurasi dan konsistensi identifikasi pada berbagai sudut kamera CCTV. Sementara itu, terkait adaptasi terhadap alur kerja di instansi, penulis secara proaktif meningkatkan intensitas komunikasi dan koordinasi dengan pembimbing lapangan untuk memastikan setiap tahapan teknis tetap selaras dengan regulasi dan kebutuhan organisasi. Serangkaian solusi ini tidak hanya menyelesaikan permasalahan yang ada, tetapi juga memberikan wawasan tambahan dalam pengembangan aplikasi visi komputer yang resilien di lingkungan produksi yang sesungguhnya.