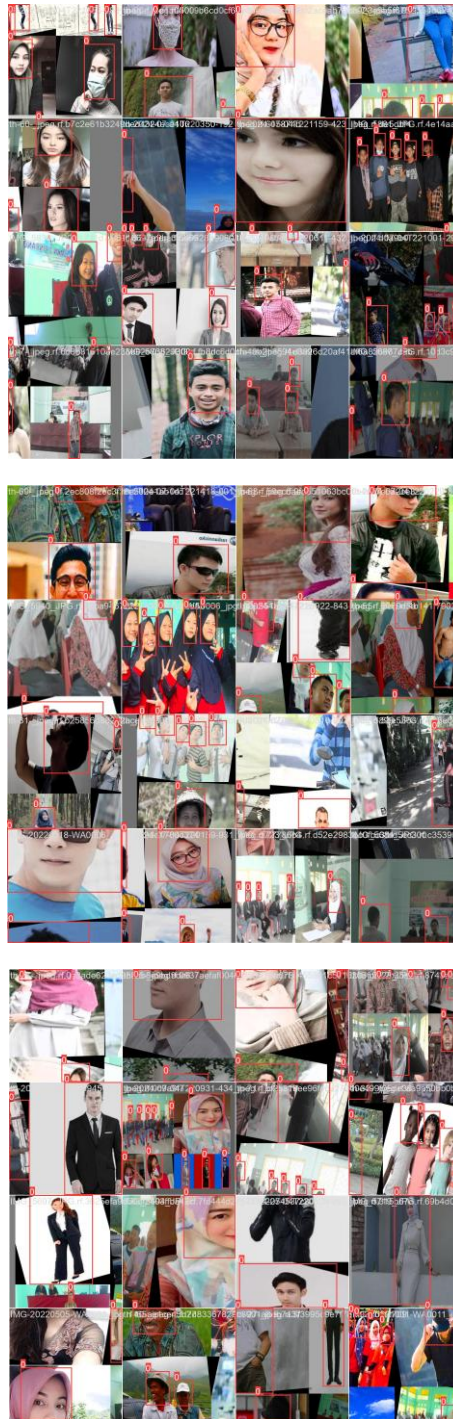


LAMPIRAN

Lampiran -1 Dokumentasi Penelitian





Kartu Monitoring Proposal

 KARTU MONITORING BIMBINGAN MAHASISWA PROGRAM STUDI TEKNIK INFORMATIKA FAKULTAS TEKNIK UNIVERSITAS MUHAMMADIYAH PAREPARE			
PROPOSAL			
Mahasiswa : MUH. YUSUF		Pembimbing I : Muhammad Basri, S.T., MT.	
NIM : 220280179		Pembimbing II : Ir. Untung Suwardoyo, S.Kom., MT.	
Judul Skripsi : IMPLEMENTASI ALGORITMA YOLOv5 PADA PERHITUNGAN PENCUCIAN MOBIL DAN MOTOR			
ARAHAN PEMBIMBING I	HARI/TGL & PARAF PEMBIMBING	ARAHAN PEMBIMBING II	HARI/TGL & PARAF PEMBIMBING
Konsultasi 1 <i>Pelaksanaan</i>		Konsultasi 1 <i>Tentukan terlebih dahulu</i>	16/5/2024 <i>[Signature]</i>
Konsultasi 2		Konsultasi 2 1. Latar belakang 2. Rumusan masalah 3. Bab 2. 4. User Interface	<i>[Signature]</i>
Konsultasi 3		Konsultasi 3 1. Bekerja 3 interface 2. Jelaskan tiap gambar 3. Jelaskan gambar 4. Rumus mobil dan motor	<i>[Signature]</i>
Konsultasi 4 <i>Acc</i>	<i>[Signature]</i>	Konsultasi 4 <i>Lengkap - Png dan</i> 1. Rumus Pneu 2. Gambar cara 3. YOLOv5	<i>[Signature]</i>
Konsultasi 5		Konsultasi 5 <i>Acc lanjut Arahensi</i> <i>ke Pan. 1</i>	<i>[Signature]</i>

Lanjut ke halaman sebelah...

Perhatian :

1. Mahasiswa wajib konsultasi minimal 5 kali
2. Kartu ini wajib dibawa oleh mahasiswa disetiap konsultasi dan diisi oleh Pembimbing
3. Kartu ini wajib dilampirkan pada laporan skripsi dan menjadi salah satu persyaratan untuk ikut seminar proposal/ujian skripsi
4. Kartu ini dicetak di atas kertas karton berwarna hijau muda dan dicetak timbal balik

Kartu Monitoring Proposal

Lanjutan ...

ARAHAN PEMBIMBING I	HARI/TGL & PARAF PEMBIMBING	ARAHAN PEMBIMBING II	HARI/TGL & PARAF PEMBIMBING
Konsultasi 6		Konsultasi 6	
Konsultasi 7		Konsultasi 7	
Konsultasi 8		Konsultasi 8	
Konsultasi 9		Konsultasi 9	
Konsultasi 10		Konsultasi 10	

Mengetahui
Ketua Program Studi

Marlina
Marlina, S.Kom., M.Kom.
NBM. 1162 680

Parepare, 16 Mei 2024

Mahasiswa

MUH YUSUF
MUH, YUSUF
NIM. 220280179

Perhatian :

1. Mahasiswa wajib konsultasi minimal 5 kali
2. Kartu ini wajib dibawa oleh mahasiswa disetiap konsultasi dan diisi oleh Pembimbing
3. Kartu ini wajib dilampirkan pada laporan skripsi dan menjadi salah satu persyaratan untuk ikut seminar proposal/ujian skripsi
4. Kartu ini dicetak di atas kertas karton berwarna hijau muda dan dicetak timbal balik

Kartu Monitoring Skripsi


KARTU MONITORING BIMBINGAN
 MAHASISWA PROGRAM STUDI TEKNIK INFORMATIKA
 FAKULTAS TEKNIK
 UNIVERSITAS MUHAMMADIYAH PAREPARE

HASIL/ SKRIPSI

Mahasiswa : MUH. YUSUF	Pembimbing I : Muh. Basri, ST.,MT
NIM : 220280179	Pembimbing II : Ir.Untung Suwardoyo, S.Kom.,M.T.,IPP
Judul Skripsi : IMPLEMENTASI ALGORITMA YOLOv5 PADA PERHITUNGAN KELUAR MASUK MANUSIA	

ARAHAN PEMBIMBING I	HARI/TGL & PARAF PEMBIMBING	ARAHAN PEMBIMBING II	HARI/TGL & PARAF PEMBIMBING
Konsultasi 1		Konsultasi 1 Latar Belakang, rumusan masalah, Tujuan, manfaat, dan kerangka pikir	
Konsultasi 2		Konsultasi 2 Rancangan sistem, Flowchart, ~~Tampilkan scrip pada detail sistem~~	
Konsultasi 3		Konsultasi 3 ~~Hasil Penelitian dan Kesimpulan~~ ~~dan Saran~~ Tampilkan scrip pada detail sistem	
Konsultasi 4		Konsultasi 4 Hasil Penelitian, kesimpulan dan saran.	
Konsultasi 5		Konsultasi 5 Abstrak, rumusan masalah, Tujuan, dan Batasan masalah	

Lanjut ke halaman sebelah...

Perhatian :

1. Mahasiswa wajib konsultasi minimal 5 kali
2. Kartu ini wajib dibawa oleh mahasiswa disetiap konsultasi dan diisi oleh Pembimbing
3. Kartu ini wajib dilampirkan pada laporan skripsi dan menjadi salah satu persyaratan untuk ikut seminar proposal/ujian skripsi
4. Kartu ini dicetak di atas kertas karton berwarna hijau muda dan dicetak timbal balik

Kartu Monitoring Skripsi

Lanjutan ...

ARAHAN PEMBIMBING I	HARI/TGL & PARAF PEMBIMBING	ARAHAN PEMBIMBING II	HARI/TGL & PARAF PEMBIMBING
Konsultasi 6		Konsultasi 6 Ace hasil	
Konsultasi 7		Konsultasi 7 Lampiran Rasi pengisi	
Konsultasi 8		Konsultasi 8 Ace ke pm 1.	
Konsultasi 9		Konsultasi 9	
Konsultasi 10		Konsultasi 10	

Parepare, 26 November 2024

Mengetahui
Ketua Program Studi

Marlina, S.Kom., M.Kom.
NBM. 1162 680

Mahasiswa

MUH. YUSUF
NIM. 220280179

Perhatian :

1. Mahasiswa wajib konsultasi minimal 5 kali
2. Kartu ini wajib dibawa oleh mahasiswa disetiap konsultasi dan diisi oleh Pembimbing
3. Kartu ini wajib dilampirkan pada laporan skripsi dan menjadi salah satu persyaratan untuk ikut seminar proposal/ujian skripsi
4. Kartu ini dicetak di atas kertas karton berwarna hijau muda dan dicetak timbal balik

Lampiran -2 Detail Tubuh dan Pose Manusia

1. Kepala dan Wajah

a. Rambut:

- **Warna:** Hitam, coklat, pirang, merah, abu-abu.
- **Tekstur:** Lurus, bergelombang, keriting.
- **Gaya:** Pendek, panjang, terikat, belah tengah, atau botak.
- **Bagian Depan:** Poni atau garis rambut dapat menjadi ciri khas.
- **Bagian Belakang:** Pola rambut, misalnya cepak atau panjang tergerai.

b. Wajah:

1) Bentuk Wajah:

- **Oval:** Proporsi seimbang, dagu melengkung.
- **Bulat:** Lebar wajah hampir sama dengan panjangnya.
- **Persegi:** Rahang tegas dengan dahi yang lebar.
- **Panjang:** Wajah lebih tinggi dibandingkan lebarnya.

2) Fitur Wajah:

➤ Mata:

- **Bentuk:** Almond, bulat, atau sipit.
- **Warna iris:** Hitam, coklat, biru, hijau.
- **Posisi:** Sejajar dan simetris di kedua sisi hidung.

➤ Hidung:

- **Bentuk:** Mancung, pesek, atau lebar.
- **Panjang:** Proporsi hidung terhadap wajah biasanya $\frac{1}{3}$ panjang wajah.

➤ **Mulut:**

- **Bentuk bibir:** Tipis, tebal, atau asimetris.
- **Posisi:** Berada sejajar dengan ujung bawah hidung.

➤ **Telinga:**

- **Letak:** Sejajar dengan alis dan ujung bawah hidung.
- **Bentuk:** Bulat atau sedikit lonjong.

c. **Pose:**

- **Menghadap Depan:** Semua fitur wajah terlihat simetris.
- **Menyamping:** Profil wajah dominan: hidung menonjol, satu mata terlihat jelas, bibir setengah terlihat.
- **Membelakangi:** Tidak ada tampilan fitur wajah, fokus pada belakang kepala (misalnya rambut atau bagian leher).

2. Leher

- a. **Panjang Leher:** Proporsional, biasanya 1/4 panjang kepala.
- b. **Tekstur:** Kulit leher cenderung halus dengan sedikit kerutan.
- c. **Pose:**

- **Tegak:** Leher tampak lurus.
- **Miring atau memutar:** Otot leher terlihat lebih tegas.

3. Bahu

- a. **Lebar Bahu:** Lebar bahu biasanya proporsional dengan tinggi tubuh.
- b. **Bentuk:**
 - **Tegap:** Bahu lurus.
 - **Menurun:** Bahu sedikit miring ke bawah.

c. Pose:

- **Menghadap Depan:** Bahu sejajar dan terlihat simetris.
- **Menyamping:** Salah satu bahu terlihat lebih menonjol ketika menyamping.
- **Membelakangi:** Bahu terlihat menyatu dengan punggung, termasuk tulang belikat

4. Lengan dan Tangan

a. Lengan:

- **Panjang:** Lengan atas hampir sama lengan bawah.
- **Otot:** Terlihat pada bagian atas (bicep) dan bawah (forearm).
- **Pose:**
 - **Lurus:** Posisi lengan sejajar dengan tubuh.
 - **Membengkok:** Siku membentuk sudut.

b. Tangan:

- **Telapak Tangan:**
 - Ukuran telapak mendekati panjang wajah.
 - Memiliki garis menyerupai bentuk huruf M
- **Jari:**
 - **Panjang:** Jari tengah paling panjang, ibu jari terpendek.
 - **Sendi:** Terlihat pada lipatan jari ketika membengkok.
- **Pose Tangan:**
 - **Menggenggam:** Jari-jari melipat sepenuhnya ke telapak.
 - **Terbuka:** Jari menyebar lebar.

- **Memegang Objek:** Tekanan pada bagian ujung jari dan telapak.

5. Torso (Bagian Tengah Tubuh)

a. Dada:

- **Bentuk:** Datar atau sedikit melengkung.
- **Proporsi:** Lebar dada hampir sama lebar bahu.

b. Perut:

- **Bentuk:** Rata, buncit, atau berotot (tergantung individu).
- **Pusat:** Pusat terletak di tengah tubuh bagian bawah.

c. Pose:

- **Menghadap Depan:** Torso terlihat simetris.
- **Menyamping:** Siluet dada dan perut lebih jelas terlihat.
- **Membelakangi:** Punggung terlihat penuh, termasuk pola tulang belikat dan bagian atas punggung.

6. Pinggul dan Panggul

a. Lebar Pinggul:

- Lebar pinggul mendekati 80%-100% lebar bahu (berbeda antara pria dan wanita).

b. Pose:

- **Menghadap Depan:** Pinggul sejajar.
- **Menyamping:** Salah satu sisi pinggul lebih tinggi, lekukan tubuh lebih jelas.

- **Membelakangi:** Pinggul terlihat lebih jelas dengan proporsi tubuh yang seimbang.

7. Kaki

a. Paha:

- **Bentuk:** Bagian atas lebih lebar dari betis.
- **Pose:**
 - **Menghadap Depan:** Kedua paha sejajar dan simetris.
 - **Menyamping:** Satu sisi paha terlihat penuh.
 - **Membelakangi:** Bagian belakang paha dan otot terlihat jelas.

b. Lutut:

- **Struktur:** Sendi besar yang menghubungkan paha dan betis.
- **Pose:**
 - **Lurus:** Lutut terkunci.
 - **Menekuk:** Sendi lutut terlihat menonjol.

c. Betis dan Pergelangan Kaki:

- **Bentuk Betis:** Bagian atas lebih lebar, meruncing ke bawah.
- **Pergelangan Kaki:** Area sempit sebelum telapak kaki.
- **Pose:**
 - **Menghadap Depan:** Betis simetris, pergelangan kaki terlihat.
 - **Menyamping:** Siluet otot lebih jelas terlihat pada bagian samping.

- **Membelakangi:** Pola otot betis dominan.

d. Kaki (Telapak):

➤ **Ukuran:** Panjang telapak kaki mendekati 15% dari tinggi tubuh.

➤ **Jari Kaki:**

- Lima jari, dengan jempol kaki yang paling besar.
- Kuku terlihat di ujung jari kaki.

➤ **Pose:**

- **Berdiri:** Telapak kaki rata dengan lantai.
- **Berjinjit:** Berat tubuh bertumpu pada jari kaki.

8. Analisis Pose

- a. Menghadap Depan:** Semua bagian tubuh terlihat penuh dan simetris, memudahkan analisis proporsi tubuh.
- b. Menyamping:** Hanya satu sisi tubuh terlihat, memberikan fokus pada profil samping (hidung, dada, pinggul, dan kaki).
- c. Membelakangi:** Bagian belakang tubuh menjadi fokus, dengan analisis rambut, bahu, punggung, dan bagian belakang kaki yang lebih dominan.

Lampiran -3 Listing Program

```

import cv2
import numpy as np
import mysql.connector
import tkinter as tk
from tkinter import messagebox, ttk, filedialog
from tkcalendar import DateEntry
from reportlab.lib.pagesizes import A4
from reportlab.platypus import SimpleDocTemplate, Table,
TableStyle
from reportlab.lib import colors

# Variabel global untuk tanggal yang dipilih
selected_date = None

def connect_to_database():
    try:
        connection = mysql.connector.connect(
            host="localhost",
            user="root",
            password="", # di sesuaikan
            database="pelabuhan_parepare"
        )
        if connection.is_connected():
            print("Berhasil terhubung ke database")
            return connection
    except mysql.connector.Error as err:
        messagebox.showerror("Error", f"Error: {err}")
        return None

def save_to_database(jumlah_masuk, jumlah_keluar):
    connection = connect_to_database()
    if connection:
        cursor = connection.cursor()
        try:
            query = "INSERT INTO deteksi_penumpang
(jumlah_masuk, jumlah_keluar, Waktu) VALUES (%s, %s, NOW())"
            cursor.execute(query, (jumlah_masuk, jumlah_keluar))
            connection.commit()
            print("Data berhasil disimpan ke database")
        except mysql.connector.Error as err:
            messagebox.showerror("Error", f"Gagal menyimpan
data: {err}")
        finally:
            cursor.close()
            connection.close()

def start_detection():

```

```

    count_in = 0 # Variabel untuk menghitung jumlah objek yang
    melewati garis masuk
    count_out = 0 # Variabel untuk menghitung jumlah objek yang
    melewati garis keluar
    classes = ["manusia-lpAE"] # Daftar kelas objek yang
    terdeteksi

    cap = cv2.VideoCapture(0) # Mengakses kamera
    if not cap.isOpened(): # Memeriksa apakah kamera berhasil
    diakses
        messagebox.showerror("Error", "Kamera tidak dapat
    diakses.") # Menampilkan pesan error jika tidak
        return
    cap.set(cv2.CAP_PROP_FRAME_WIDTH, 640) # Mengatur lebar
    frame
    cap.set(cv2.CAP_PROP_FRAME_HEIGHT, 480) # Mengatur tinggi
    frame

    net = cv2.dnn.readNetFromONNX("yolov5s.onnx") # Memuat
    model YOLOv5 dari file ONNX

    line_in_y = 220 # Posisi Y untuk garis masuk
    line_out_y = 230 # Posisi Y untuk garis keluar

    line_crossed = {} # Dictionary untuk melacak objek yang
    telah melewati garis

    while True: # Loop utama untuk menangkap dan memproses
    frame
        ret, img = cap.read() # Mengambil frame dari kamera

        if not ret: # Memeriksa apakah frame berhasil diambil
            print("Warning: Tidak dapat mengambil frame dari
    video.") # Peringatan jika gagal
            break

        # Menyiapkan blob untuk input ke model
        blob = cv2.dnn.blobFromImage(img, scalefactor=1/255,
    size=(640, 640), mean=[0, 0, 0], swapRB=True, crop=False)
        net.setInput(blob) # Mengatur input untuk model
        detections = net.forward()[0] # Melakukan inferensi dan
    mendapatkan deteksi

        img_width, img_height = img.shape[1], img.shape[0] #
    Mendapatkan ukuran gambar
        x_scale = img_width / 640 # Skala X
        y_scale = img_height / 640 # Skala Y

        boxes = [] # Daftar untuk menyimpan bounding boxes
        classes_ids = [] # Daftar untuk menyimpan ID kelas
        confidences = [] # Daftar untuk menyimpan tingkat
    kepercayaan

```

```

        # Menggambar garis masuk
        cv2.line(img, (0, line_in_y), (img_width, line_in_y),
(0, 255, 0), 2)
        cv2.putText(img, "In", (10, line_in_y - 10),
cv2.FONT_HERSHEY_SIMPLEX, 0.5, (0, 255, 0), 2)

        # Menggambar garis keluar
        cv2.line(img, (0, line_out_y), (img_width, line_out_y),
(0, 0, 255), 2)
        cv2.putText(img, "Out", (10, line_out_y - 10),
cv2.FONT_HERSHEY_SIMPLEX, 0.5, (0, 0, 255), 2)

    for i in range(detections.shape[0]): # Memproses setiap
deteksi
        row = detections[i] # Mendapatkan informasi deteksi
        confidence = row[4] # Mendapatkan nilai kepercayaan
        if confidence > 0.2: # Memeriksa apakah kepercayaan
cukup tinggi
            classes_score = row[5:] # Skor kelas
            ind = np.argmax(classes_score) # ID kelas
            dengan skor tertinggi
            if ind < len(classes) and classes_score[ind] >
0.5: # Memeriksa apakah kelas valid
                classes_ids.append(ind) # Menyimpan ID
kelas
                confidences.append(confidence) # Menyimpan
kepercayaan
                cx, cy, w, h = row[:4] # Mendapatkan pusat
dan ukuran bounding box
                x1 = int((cx - w / 2) * x_scale) #
Koordinat X1
                y1 = int((cy - h / 2) * y_scale) #
Koordinat Y1
                width = int(w * x_scale) # Lebar bounding
box
                height = int(h * y_scale) # Tinggi bounding
box
                boxes.append([x1, y1, width, height]) #
Menyimpan bounding box

        # Non-Maximum Suppression untuk menghindari deteksi
ganda
        if len(boxes) > 0:
            indices = cv2.dnn.NMSBoxes(boxes, confidences, 0.4,
0.4)

            for i in indices: # Memproses setiap bounding box
yang tersisa
                box_index = i[0] if isinstance(i, np.ndarray)
else i
                box = boxes[box_index]

```

```

        x1, y1, width, height = box

        # Menghitung titik tengah bounding box
        center_x = int(x1 + width / 2)
        center_y = int(y1 + height / 2)

        # Menggambar titik tengah
        cv2.circle(img, (center_x, center_y), 2, (0, 0,
255), -1) # Titik tengah berwarna merah

        # Menghasilkan ID unik untuk setiap objek
terdeteksi
        object_id =
f"{classes[classes_ids[box_index]]}_{box_index}"

        # Toleransi dalam piksel untuk deteksi garis
        tolerance = 5

        # Cek untuk garis masuk
        if object_id not in line_crossed:
            if line_in_y - tolerance < center_y <
line_in_y + tolerance and center_y < line_in_y:
                count_in += 1 # Increment count_in
                line_crossed[object_id] = 'in' # Tandai
objek telah melewati garis masuk
                cv2.rectangle(img, (x1, y1), (x1 +
width, y1 + height), (0, 255, 0), 2)

        # Cek untuk garis keluar
            if line_out_y - tolerance < center_y <
line_out_y + tolerance and center_y > line_out_y:
                count_out += 1 # Increment count_out
                line_crossed[object_id] = 'out' #
Tandai objek telah melewati garis keluar
                cv2.rectangle(img, (x1, y1), (x1 +
width, y1 + height), (0, 0, 255), 2)

        # Menghapus objek yang sudah terhitung
        if object_id in line_crossed:
            if line_crossed[object_id] == 'in' and
center_y > line_in_y:
                del line_crossed[object_id] # Menghapus
objek dari tracking
            elif line_crossed[object_id] == 'out' and
center_y < line_out_y:
                del line_crossed[object_id] # Menghapus
objek dari tracking

        # Menambahkan label pada objek terdeteksi
        label = f"{classes[classes_ids[box_index]]}:
{confidence:.2f}"

```

```

        cv2.putText(img, label, (x1, y1 - 10),
cv2.FONT_HERSHEY_SIMPLEX, 0.5, (255, 255, 255), 2)

        # Menampilkan hasil deteksi
        cv2.putText(img, f"In: {count_in}, Out: {count_out}",
(10, 30), cv2.FONT_HERSHEY_SIMPLEX, 1, (255, 0, 0), 2)

        cv2.imshow("Deteksi orang", img) # Menampilkan gambar
dengan deteksi
        if cv2.waitKey(1) & 0xFF == 27: # Menunggu tombol 'Esc'
untuk keluar
            break

        cap.release() # Melepaskan kamera
        cv2.destroyAllWindows() # Menutup semua jendela OpenCV
        save_to_database(count_in, count_out) # Menyimpan data ke
database

def view_database_report():
    def fetch_data_by_date(date):
        global selected_date
        selected_date = date
        query = "SELECT * FROM deteksi_penumpang WHERE
DATE(Waktu) = %s"
        connection = connect_to_database()
        if connection:
            cursor = connection.cursor()
            try:
                cursor.execute(query, (date,))
                rows = cursor.fetchall()

                for row in tree.get_children():
                    tree.delete(row)

                for row in rows:
                    tree.insert("", "end", values=row)
            except mysql.connector.Error as err:
                messagebox.showerror("Error", f"Error: {err}")
            finally:
                cursor.close()
                connection.close()

    def fetch_all_data():
        query = "SELECT * FROM deteksi_penumpang"
        connection = connect_to_database()
        if connection:
            cursor = connection.cursor()
            try:
                cursor.execute(query)
                rows = cursor.fetchall()

```



```

        ('ALIGN', (0, 0), (-1, 0), 'CENTER'),
        ('ALIGN', (0, 1), (-1, -1), 'CENTER'),
        ('VALIGN', (0, 0), (-1, -1), 'MIDDLE'),
        ('LEFTPADDING', (0, 0), (-1, -1), 6),
        ('RIGHTPADDING', (0, 0), (-1, -1), 6),
        ('TOPPADDING', (0, 0), (-1, -1), 6),
        ('BOTTOMPADDING', (0, 0), (-1, -1), 6),
    ]))

    elements.append(table)
    doc.build(elements)
    messagebox.showinfo("Sukses", "Laporan
berhasil dicetak sebagai PDF.")
except mysql.connector.Error as err:
    messagebox.showerror("Error", f"Error: {err}")
finally:
    cursor.close()
    connection.close()

# Buat jendela laporan
report_window = tk.Toplevel(root)
report_window.title("Laporan Database")
report_window.geometry("900x600")

# Frame untuk pencarian
search_frame = tk.Frame(report_window, bg="#2c3e50",
pady=10)
search_frame.pack(fill='x')

tk.Label(search_frame, text="Pilih Tanggal:",
font=("Helvetica", 14), bg="#2c3e50",
fg="#ecf0f1").pack(side="left")

date_entry = DateEntry(search_frame, date_pattern='yyyy-mm-
dd', background='darkblue', foreground='white', borderwidth=2)
date_entry.pack(side="left", padx=10)

search_button = tk.Button(search_frame, text="Cari",
command=lambda:
fetch_data_by_date(date_entry.get_date().strftime('%Y-%m-%d')),
font=("Helvetica", 12), bg="#e74c3c", fg="#ecf0f1")
search_button.pack(side="left")

# Frame untuk tombol cetak
print_frame = tk.Frame(report_window, bg="#34495e", pady=10)
print_frame.pack(fill='x', anchor="ne")

print_button = tk.Button(print_frame, text="Cetak ke PDF",
command=print_to_pdf, font=("Helvetica", 12), bg="#e74c3c",
fg="#ecf0f1")
print_button.pack(side="right")

```

```

# Frame untuk laporan
report_frame = tk.Frame(report_window)
report_frame.pack(expand=True, fill='both')

vsb = tk.Scrollbar(report_frame, orient="vertical")
vsb.pack(side="right", fill="y")

# Treeview untuk menampilkan data
tree = ttk.Treeview(report_frame, columns=("ID", "Jumlah
Masuk", "Jumlah Keluar", "Waktu"), show="headings",
yscrollcommand=vsb.set)
tree.heading("ID", text="ID", anchor="center")
tree.heading("Jumlah Masuk", text="Jumlah Masuk",
anchor="center")
tree.heading("Jumlah Keluar", text="Jumlah Keluar",
anchor="center")
tree.heading("Waktu", text="Waktu", anchor="center")

tree.column("ID", anchor="center")
tree.column("Jumlah Masuk", anchor="center")
tree.column("Jumlah Keluar", anchor="center")
tree.column("Waktu", anchor="center")

tree.pack(side="left", expand=True, fill="both")

vsb.config(command=tree.yview)

# Ambil dan tampilkan semua data pada awalnya
fetch_all_data()

# Buat jendela utama
root = tk.Tk()
root.title("Interface Pendeteksian")

root.geometry("500x300")
root.configure(bg="#34495e")

frame = tk.Frame(root, bg="#34495e")
frame.pack(expand=True, fill='both')

button_style = {
    "font": ("Helvetica", 16, "bold"),
    "fg": "#ecf0f1",
    "bg": "#e74c3c",
    "activebackground": "#c0392b",
    "activeforeground": "#ecf0f1",
    "bd": 5,
    "relief": "raised",
    "width": 20,
    "height": 2
}

```

```
btn_detection = tk.Button(frame, text="Mulai Pendeteksian",
command=start_detection, **button_style)
btn_detection.pack(pady=20)

btn_report = tk.Button(frame, text="Laporan Database",
command=view_database_report, **button_style)
btn_report.pack(pady=20)

root.mainloop()
```