Nama : Kristian Danuarta

Nim : 230741100

Prodi : Ilmu Komputer

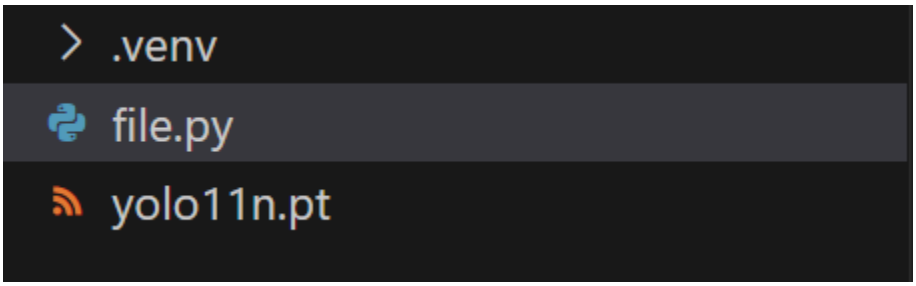
Matkul : Kecerdasan Buatan / artificial intelligent (AI)

Fakultas : Teknik Dan Sains

PENJELASAN TOTORIAL DEPLOY

DETEKSI REAL-TIME CAMERA , DETEKSI VIDEO AND DETEKSI GAMBAR FOTO UPLOAD

Langkah 1



1. Sebelum melakukan deteksi objeck , siapkan folder sesuai nama di WIN C: , lalu isi file yollo 11 yang sudah diterakan oleh bpk terhomat kita zikri wayuzi, lalu buatlah 1 file python (app.py)
2. Lalu buka visual code lalu isi coding yang sudah di terakan di whatsaap

Langkah 2

1. Installasilah library sesuai dengan tugas :

pip install ultralytics opencv-python-headless streamlit pillow numpy

codingan library :

```
from ultralytics import YOLO
import cv2
import streamlit as st
from PIL import Image
import numpy as np
from collections import Counter
```

PENJELASAN :

1. **from ultralytics import YOLO**
   Ultralytics:
   Library Python resmi yang dibuat oleh tim pengembang YOLO.
   Mendukung berbagai versi YOLO, termasuk YOLOv5, YOLOv6, dan YOLOv8.
   Mempermudah implementasi model YOLO untuk tugas seperti:
   - Deteksi objek (object detection).
   - Segmentasi (segmentation).
   - Deteksi pose (pose estimation).

   YOLO:
   Kelas utama yang disediakan oleh library ini untuk memuat, melatih, dan menggunakan model YOLO.
   Mengabstraksikan berbagai tugas seperti pelatihan model, validasi, dan inferensi (prediksi).

2. **import cv2** cv2 adalah library OpenCV (Open Source Computer Vision) untuk Python. Library ini digunakan untuk pemrosesan gambar dan video.
   Fungsi Utama:
   Manipulasi Gambar: Membaca, menulis, mengubah ukuran, memutar, atau memotong gambar.
   Analisis Gambar: Mendeteksi tepi, objek, wajah, atau gerakan.
   Pemrosesan Video: Membaca, menulis, dan menganalisis alur video frame by frame.
   Visi Komputer Lanjut: Menerapkan algoritma seperti segmentasi, optical flow, dan object tracking.

3. **import streamlit as st** import
   streamlit as st
   streamlit adalah framework Python untuk membangun aplikasi web berbasis data secara cepat dan interaktif.
   Fungsi Utama:
   Membuat Antarmuka Pengguna: Mempermudah pembuatan UI dengan elemen seperti slider, tombol, form, dll.
   Visualisasi Data: Mendukung grafik dan visualisasi interaktif menggunakan library seperti Matplotlib, Plotly, atau Altair.
   Komunikasi Real-Time: Memungkinkan pengguna untuk berinteraksi langsung dengan model atau data yang Anda buat.

4. **from PIL import Image** from PIL
   import Image
   PIL (Python Imaging Library) adalah library untuk memanipulasi gambar. Versi modernnya adalah Pillow.
   Fungsi Utama:
   Membuka, memproses, dan menyimpan file gambar dalam berbagai format (JPEG, PNG, GIF, dll.).
   Mendukung operasi manipulasi gambar seperti rotasi, cropping, scaling, dan filter.

5. **import numpy as np** import
   numpy as np
   numpy adalah library untuk komputasi numerik di Python, terutama untuk array dan operasi matematika linier.
   Fungsi Utama:
   Array Manipulation: Membuat dan memproses array multidimensi.
   Operasi Matematika: Perkalian matriks, statistik, transformasi Fourier, dll.
   Pemrosesan Gambar: Banyak library visi komputer (seperti OpenCV) menggunakan array NumPy untuk merepresentasikan gambar.

6. **from collections import Counter**
   from collections import Counter
   Counter adalah bagian dari modul collections di Python yang digunakan untuk menghitung elemen dalam iterable.

Fungsi Utama:

Menghitung frekuensi elemen dalam list, string, atau iterable lainnya.

Memberikan hasil berupa dictionary dengan elemen sebagai kunci dan frekuensi sebagai nilai.

Langkah 3

```
8     # Load YOLO model
9     @st.cache_resource
10    def load_model(model_path):
11        return YOLO(model_path)
```

Penjelasan ;

1. **@st.cache_resource: Decorator Streamlit** untuk menyimpan hasil fungsi di cache, sehingga fungsi hanya dijalankan sekali untuk parameter yang sama.
2. **load_model:** Fungsi untuk memuat model YOLO dari file model (model_path).

Keuntungan:

- Efisiensi Waktu: Model hanya dimuat sekali.
- Pengalaman Lebih Cepat: Mengurangi delay saat aplikasi Streamlit digunakan ulang.
- Hemat Resource: Menghindari pemrosesan ulang yang tidak perlu.

```python
# Process and display the detection results
def display_results(image, results):
    boxes = results.boxes.xyxy.cpu().numpy()  # [x1, y1, x2, y2]
    scores = results.boxes.conf.cpu().numpy()  # Confidence scores
    labels = results.boxes.cls.cpu().numpy()  # Class indices
    names = results.names  # Class names

    detected_objects = []

    for i in range(len(boxes)):
        if scores[i] > 0.5:  # Confidence threshold
            x1, y1, x2, y2 = boxes[i].astype(int)
            label = names[int(labels[i])]
            score = scores[i]
            detected_objects.append(label)
            cv2.rectangle(image, (x1, y1), (x2, y2), (0, 255, 0), 2)
            cv2.putText(image, f"{label}: {score:.2f}", (x1, y1 - 10), cv2.FONT_H

    return image, detected_objects
```

Penjelasan :

Fungsi display_results

Fungsi ini memproses hasil deteksi dari model YOLO, menampilkan kotak pembatas (bounding boxes), label, dan skor kepercayaan pada gambar, serta mengembalikan daftar objek yang terdeteksi.

Penjelasan Komponen:

1. **Input Parameter:**
   - image: Gambar asli yang akan ditampilkan dengan hasil deteksi.
   - results: Hasil deteksi model YOLO (termasuk bounding boxes, skor, dan label).
   - confidence_threshold: Batas minimum untuk skor kepercayaan agar deteksi dianggap valid (default 0.5).
2. **Proses Deteksi:**
   - results.boxes.xyxy: Koordinat kotak pembatas [x1, y1, x2, y2].
   - results.boxes.conf: Skor kepercayaan (confidence score) untuk setiap kotak.
   - results.boxes.cls: Indeks kelas (label numerik) objek.
   - results.names: Nama kelas (label teks).
3. **Iterasi Hasil:**

- Periksa apakah skor kepercayaan lebih besar dari confidence_threshold.
- Gambar kotak pembatas (cv2.rectangle) dan tambahkan teks label serta skor (cv2.putText) pada gambar.

4. **Hasil:**
   - image: Gambar dengan hasil deteksi (bounding box dan label).
   - detected_objects: Daftar label objek yang terdeteksi.

Cara Kerja:

1. Gambar diberi anotasi kotak pembatas untuk setiap deteksi yang memenuhi confidence_threshold.
2. Teks label dan skor kepercayaan ditambahkan pada gambar.
3. Fungsi mengembalikan gambar hasil anotasi dan daftar label objek.

```
34  def main():
35      st.title("☹KRISTIANO RONALDO😉")
36      st.sidebar.title("Settings")
37
38      model_path = "yolo11n.pt"  # Path to your YOLO model
39      model = load_model(model_path)
40
41      # Create the checkbox once
42      run_detection = st.sidebar.checkbox("Start/Stop Object Detection", key="detec
43
44      # Open video capture if checkbox is active
45      if run_detection:
46          cap = cv2.VideoCapture(0)
47          st_frame = st.empty()  # Placeholder for video frames
48          st_detection_info = st.empty()  # Placeholder for detection information
49
50          while True:
51              ret, frame = cap.read()
52              if not ret:
```

PENJELASAN :

1. **main()** ○ Fungsi utama aplikasi Streamlit yang menjalankan seluruh proses deteksi objek.
2. **st.set_page_config** ○ Mengatur konfigurasi halaman Streamlit (judul halaman dan layout lebar penuh).
3. **Judul Aplikasi** ○ Menampilkan judul utama dan sidebar menggunakan st.title dan st.sidebar.title.

4. **Memuat Model YOLO** ○ Model YOLO dimuat menggunakan fungsi load_model, dengan path model ditentukan di variabel model_path.
5. **Sidebar: Pilih Mode** ○ Opsi untuk memilih mode deteksi objek:
   - **Real-time Kamera**: Menggunakan kamera untuk deteksi langsung.
   - **Unggah Gambar**: Memungkinkan pengguna mengunggah gambar untuk dianalisis.
6. **Sidebar: Confidence Threshold** ○ Slider untuk mengatur ambang kepercayaan (confidence threshold) model YOLO dalam mendeteksi objek (rentang 0.1 hingga 1.0).
7. **Mode "Unggah Gambar"** ○ Menampilkan form untuk mengunggah file gambar.
   - Jika gambar diunggah:
     - Dibaca menggunakan Pillow (Image.open).
     - Dikoversi menjadi array NumPy (np.array) agar kompatibel dengan YOLO.

Fungsi Utama

- Interaktivitas: Pengguna dapat memilih mode deteksi dan mengatur parameter.
- Fleksibilitas: Mendukung deteksi pada gambar atau video secara real-time. Alur Utama

1. Konfigurasi aplikasi → Load model YOLO → Pilih mode → Deteksi objek berdasarkan input (gambar/kamera).

4o

```
            frame = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)  # Convert to RGB for
            results = model.predict(frame, imgsz=640)  # Perform detection

            # Draw results and collect detected objects
            frame, detected_objects = display_results(frame, results[0])

            # Display video feed
            st_frame.image(frame, channels="RGB", use_column_width=True)

            # Display detection information
            if detected_objects:
                object_counts = Counter(detected_objects)
                detection_info = "\n".join([f"{obj}: {count}" for obj, count in ob
            else:
                detection_info = "No objects detected."

            st_detection_info.text(detection_info)  # Update detection info text
```

PENJELASAN :

- **Deteksi pada Gambar:**
  - model.predict(image_np, imgsz=640): Mendeteksi objek dalam gambar (image_np) dengan ukuran gambar yang disesuaikan (imgsz=640).
  - display_results(image_np, results[0], confidence_threshold): Menampilkan hasil deteksi pada gambar dan meng-filter objek berdasarkan confidence_threshold. Fungsi ini mengembalikan gambar dengan objek yang terdeteksi dan daftar objek yang terdeteksi.
- **Menampilkan Hasil Deteksi Gambar:**
  - st.image(image_np, caption="Hasil Deteksi", use_column_width=True): Menampilkan gambar yang sudah terdeteksi objeknya di aplikasi Streamlit**.**
- **Menampilkan Info Deteksi:**
  - Jika objek terdeteksi, informasi tentang objek yang terdeteksi ditampilkan menggunakan Counter untuk menghitung jumlah setiap objek. Info tersebut ditampilkan dalam format list.
  - Jika tidak ada objek terdeteksi, aplikasi menampilkan pesan "Tidak ada objek terdeteksi."
- **Deteksi Real-time Kamera:**
  - Mode "Real-time Kamera" memungkinkan deteksi objek secara langsung menggunakan kamera.

- st.sidebar.checkbox("Mulai Deteksi"): Opsi untuk memulai deteksi real-time.
- cv2.VideoCapture(0): Membuka kamera untuk deteksi real-time.
- Placeholder st.empty() digunakan untuk menampilkan video dan info deteksi secara dinamis selama deteksi berlangsung.

```
    while True:
        ret, frame = cap.read()
        if not ret:
            st.warning("❌ Gagal menangkap gambar dari kamera.")
            break

        # Konversi frame ke RGB dan lakukan deteksi
        frame = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
        results = model.predict(frame, imgsz=640)
        frame, detected_objects = display_results(frame, results[0], confidence_threshold)

        # Tampilkan hasil video dan info deteksi
        st_frame.image(frame, channels="RGB", use_column_width=True)

        if detected_objects:
            object_counts = Counter(detected_objects)
            detection_info = "\n".join([f"{obj}: {count}" for obj, count in object_counts.items()])
        else:
            detection_info = "❌ Tidak ada objek terdeteksi."

        st_detection_info.text(detection_info)

        # Hentikan deteksi jika checkbox dimatikan
        if not st.session_state.detection_control:
            break
```

PENJELASAN :

Penjelasan Singkat Kode:

1. **while True::** Loop utama yang terus berjalan untuk menangkap frame dari kamera dan melakukan deteksi objek secara terus-menerus.
2. **ret, frame =** cap.read(): Membaca frame dari kamera (video stream). Jika gagal, program akan memberikan peringatan dan keluar dari loop.
3. **Konversi Frame ke RGB:**
    - cv2.cvtColor(frame, cv2.COLOR_BGR2RGB): Mengkonversi frame dari format BGR (yang digunakan oleh OpenCV) menjadi RGB (format yang digunakan oleh Streamlit dan model YOLO).
4. **Deteksi dengan YOLO:**

- model.predict(frame, imgsz=640): Melakukan deteksi objek pada frame dengan ukuran gambar 640x640.
- display_results(frame, results[0], confidence_threshold): Menampilkan hasil deteksi pada frame dan mengfilter objek berdasarkan confidence threshold. Mengembalikan frame yang telah dideteksi dan daftar objek yang terdeteksi.

5. **Tampilkan Hasil Video dan Info Deteksi:**
   - st_frame.image(frame, channels="RGB", use_column_width=True): Menampilkan frame video dengan objek yang terdeteksi di Streamlit. ○ Menampilkan informasi deteksi objek (nama dan jumlah) atau pesan jika tidak ada objek yang terdeteksi.
6. **Hentikan Deteksi:**
   - if not st.session_state.detection_control:: Jika checkbox untuk memulai deteksi dimatikan, loop berhenti dan deteksi berakhir.

Kesimpulan:

- Loop Video Real-time: Menangkap frame video dari kamera, melakukan deteksi objek, dan menampilkan hasil deteksi beserta informasi objek yang terdeteksi.
- Kontrol Deteksi: Deteksi dapat dihentikan jika pengguna mematikan kontrol deteksi (checkbox).

```
79            cap.release()
80
81  if __name__ == "__main__":
82      main()
```

PENJELASAN :

1. cap.release():

- Menutup dan membebaskan perangkat kamera setelah selesai digunakan. Fungsi ini memastikan kamera dilepaskan dan sumber daya yang digunakan dapat dibebaskan.

2. **st.success("🎉 Deteksi objek dihentikan."):**
   - Menampilkan pesan sukses di aplikasi Streamlit untuk memberi tahu pengguna bahwa deteksi objek telah dihentikan.
3. **st.sidebar.markdown("---"):**
   - Menambahkan garis pemisah di sidebar untuk memperjelas tampilan antarmuka.
4. **st.sidebar.info("👩‍💻 Dibuat dengan cinta oleh hendri menggunakan Streamlit dan YOLO 11."):** ○ Menampilkan pesan informasi di sidebar yang memberikan kredit kepada pembuat aplikasi.
5. **if __name__ == "__main__"::**
   - Memastikan fungsi main() hanya dijalankan jika skrip ini dijalankan langsung (bukan diimpor sebagai modul).
6. **main():**
   - Memanggil fungsi utama untuk menjalankan aplikasi.

JIKA SUDAH DI JELASKAN SEMUA KITA LANJUT TOTORIAL MEMBUKA STREAMLIT deploy

1. BUKA CMD DAN MASUKAN VARIABEL FOLDER YANG SUDAH DIBIKIN
2. LALU MASUKAN VERTUALMENT DENGAN PRINTAH **(VENV\SCRIPTS\ACTIPATE)**
3. LALU TEST DENGAN CODINGAN **(STREAMLIT RUN (NAMA FILE PYTHON))**

JIKA SUDAH MUNCUL SEPRTI INI :
(.venv) C:\uas-kecerdasanbuatan>streamlit run app.py

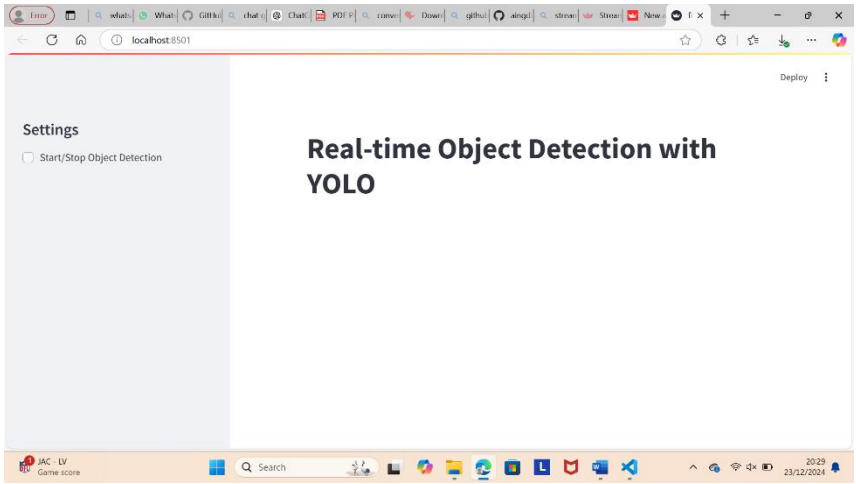
You can now view your Streamlit app in your browser.

Local URL: http://localhost:8501
Network URL: http://172.20.10.6:8501

2024-12-23 15:13:55.459 Examining the path of torch.classes raised: Tried to instantiate class '__path__._path', but it does not exist! Ensure that it is registered via torch::class_
2024-12-23 15:22:01.859 Examining the path of torch.classes raised: Tried to instantiate class '__path__._path', but it does not exist! Ensure that it is registered via torch::class_

BERATI SUDAH JALAN DAN TINGGAL MASUK KE DEPLOY NYA

DENGAN PRINTAH **http://localhost:8501**

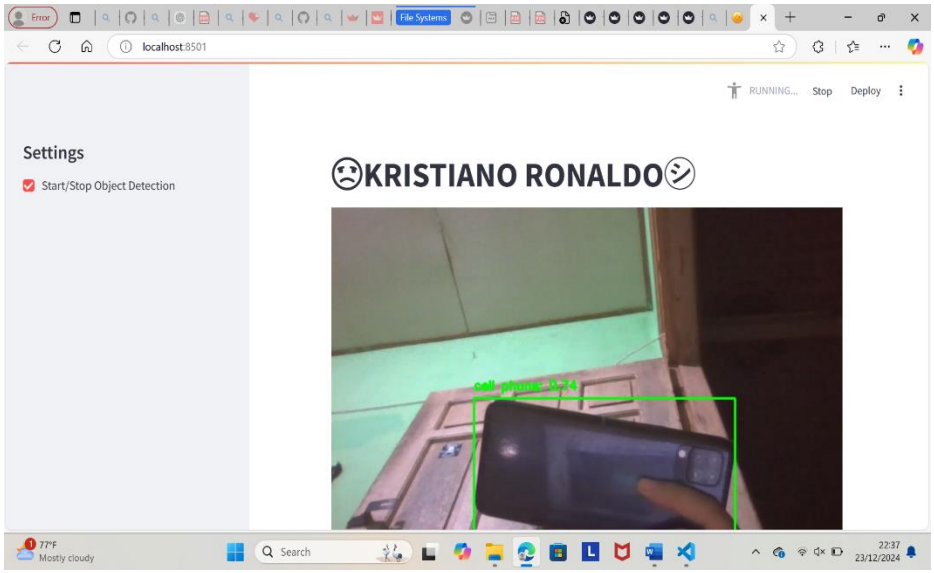


NATI AKAN MUNCUL GAMBAR

JIKA SUDAH TAMPIL SEPERTI GAMABR ITU
NANTI ADA PILIHAN YANG MAU DI DETEKSI
JIKA INGIN MENDETEKSI BAGIAN **REAL-TIME** TINGGAL TEKAN AJA
LALU CENTANGKAN **MULAI DETEKSI** MAKA AKAN MUNCUL
GAMBAR DETEKSI WAJAH DENGAN KAMERA SEPERTI CONTOH DI

BAWAH INI

JIKA SUDAH SEPRTI INI BERARTI REAL-TIME PADA KAMERA



BERHASIL DI DETEKSI JADI TAHAP BERIKUTNYA **UNGGAH GAMBAR**

Lalu unggah foto yang ingin di deteksi dan buka folrder dengan foto deteksi ukuran limit 200 mb (jpg,peg dan png)
Jika sudeh input nanti dia akan mendeteksi gamabar seprti ini :
Dia akan mendekripsikan orang kursi dan object di gambar seprti dibawahb Chair ada 5 objeck dan person ada 6 object dan deteksi siap dan sempurna untuk dicoba dalam berbentuk file and kamera jika ingin berbentuk video tinggal modifikasi coding ptrhonnya

Jika ingin menambah diploy deteksi video bisa buka file python lalu masukan codingan

```
# Sidebar: Pilih mode (gambar, video, atau kamera)
mode = st.sidebar.radio("Pilih mode deteksi:", ("Unggah Gambar", "Unggah Video", "Real-time Kamera"))
```

Ini digunakan untuk daftar menu atau sidebar pada pilihan pengen di deteksi

Stelah itu tambahkan

```
elif mode == "Unggah Video":
    st.subheader("📤 **Unggah Video untuk Deteksi Objek**")
    uploaded_file = st.file_uploader("Pilih file video:", type=["mp4", "avi", "mov"])

    if uploaded_file is not None:
        temp_video_path = "temp_video.mp4"
        with open(temp_video_path, "wb") as f:
            f.write(uploaded_file.read())

        cap = cv2.VideoCapture(temp_video_path)
        st_frame = st.empty()

        while cap.isOpened():
            ret, frame = cap.read()
            if not ret:
                break

            # Run YOLO deteksi pada setiap frame
            frame = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
            results = model.predict(frame, imgsz=640)
            frame, detected_objects = display_results(frame, results[0], confidence_threshold)

            # Tampilkan frame dengan hasil deteksi
            st_frame.image(frame, channels="RGB", use_column_width=True)

        cap.release()
        st.success("🎉 Deteksi video selesai.")
```

- **elif mode == "Unggah Video":**:
  - Mengecek jika mode yang dipilih adalah "Unggah Video" untuk deteksi objek pada video.
- **st.subheader("📤 *Unggah Video untuk Deteksi Objek*")**:
  - Menampilkan subjudul untuk memberi tahu pengguna bahwa mereka dapat mengunggah video.
- **uploaded_file = st.file_uploader("Pilih file video:", type=["mp4", "avi", "mov"])**:
  - Menyediakan uploader file untuk memilih video dengan format yang didukung (MP4, AVI, MOV).
- **Menyimpan Video yang Diunggah**:
  - Jika file video diunggah, file tersebut disimpan sementara di temp_video_path menggunakan mode write-binary.
- **cap = cv2.VideoCapture(temp_video_path)**:
  - Membuka video untuk diproses menggunakan OpenCV (cv2.VideoCapture).
- **Loop untuk Memproses Setiap Frame Video**:
  - Selama video masih dapat dibaca (cap.isOpened()), kode ini membaca dan mengonversi setiap frame menjadi format RGB, kemudian menjalankan deteksi objek YOLO pada setiap frame.
- **Menampilkan Hasil Deteksi**:
  - Frame yang sudah terdeteksi objeknya ditampilkan menggunakan st_frame.image(). **cap.release()**:

- Menutup video setelah selesai diproses.

 st.success(🎉 Deteksi video selesai."): • Menampilkan pesan sukses setelah deteksi objek pada video selesai.

```
if run_detection:
    cap = cv2.VideoCapture(0)  # Buka kamera
    st_frame = st.empty()  # Placeholder untuk video
    st_detection_info = st.empty()  # Placeholder untuk info deteksi
```

 if run_detection::

- Mengecek apakah deteksi objek harus dijalankan berdasarkan status checkbox run_detection di sidebar. Jika checkbox dicentang, kode di dalam blok ini akan dijalankan.

 cap = cv2.VideoCapture(0):

- Membuka kamera perangkat untuk menangkap video. Argumen 0 merujuk pada kamera default (biasanya kamera laptop atau webcam).

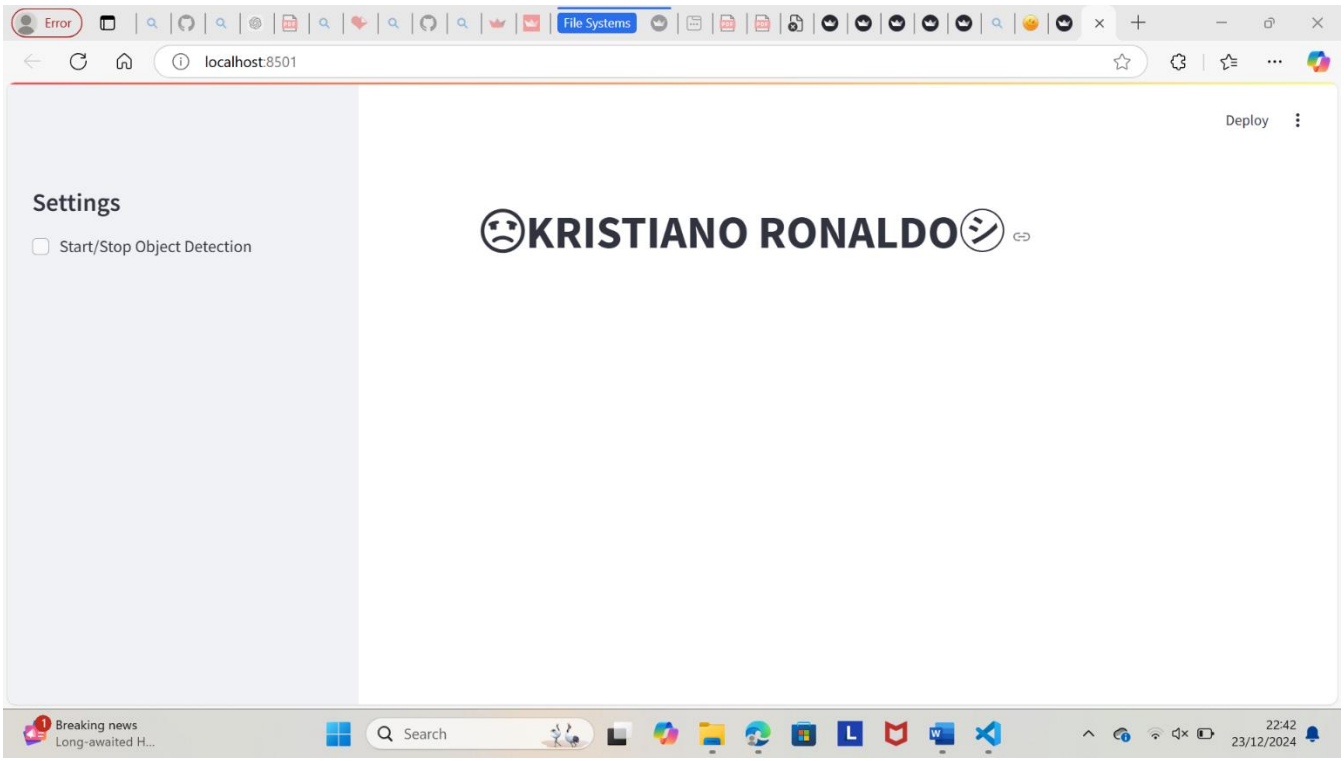
 st_frame = st.empty():

- Membuat placeholder kosong di Streamlit untuk menampilkan frame video secara dinamis. Placeholder ini akan diisi dengan gambar video yang ditangkap.

 st_detection_info = st.empty():

- Membuat placeholder kosong lainnya untuk menampilkan informasi deteksi objek (misalnya nama objek yang terdeteksi dan jumlahnya) secara dinamis.

JIKA SUDAH DITAMBAHKAN LALU BUKA LAGI WEBSITENYA DAN CEK LAGI **http://localhost:8501**
MAKA AKAN MUNCUL SEPERTI INI



Jika sudah tampilan seperti ini berarti codingan python tadi berhasil di input di tambahkan jadi tinggal menguplouad file **(mp4 , avi , mov dan mpeg4)**

Jika sudah di upload file berbentuk video maka muncul kotak deteksi berarti coding PYTHON suluruhnya dari : realtime kamera , uplod foto dan upload video Berhasil.