

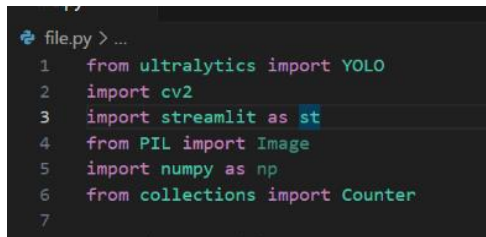
UAS KECERDASAN BUATAN
PENJELASAN KODE DAN TUTORIAL DEPLOY

Nama : Imelda Kafitri

Nim : 230741117

Matkul : kecerdasan buatan

PENJELASAN KODE



```
1 from ultralytics import YOLO
2 import cv2
3 import streamlit as st
4 from PIL import Image
5 import numpy as np
6 from collections import Counter
7
```

Pertama ada library

❖ **from ultralytics import YOLO :**

Mengimpor kelas atau fungsi YOLO dari pustaka **Ultralytics** untuk mendukung implementasi model YOLO (You Only Look Once) dalam deteksi objek.

❖ **Import cv2**

Mengimpor pustaka **OpenCV**, yang digunakan untuk pemrosesan gambar dan video, seperti membaca, menulis, dan manipulasi gambar

❖ **Import streamlit as st**

Mengimpor pustaka **Streamlit**, yang berguna untuk membuat antarmuka web interaktif dan ringan, sering digunakan untuk memvisualisasikan hasil analisis atau model.

❖ **From PIL import image**

Mengimpor kelas **Image** dari pustaka **Pillow** untuk memanipulasi gambar, seperti membuka, mengubah ukuran, atau menyimpan gambar.

❖ **Import numpy as np**

Mengimpor library NumPy untuk operasi numerik, seperti manipulasi array/matriks.

❖ `from collections import Counter`

Mengimpor Counter untuk menghitung frekuensi elemen dalam koleksi data.

```
7
8 # Load YOLO model
9 @st.cache_resource
10 def load_model(model_path):
11     return YOLO(model_path)
12 # Process and display the detection res
```

Potongan kode tersebut mendefinisikan sebuah fungsi yang memanfaatkan dekorator dari Streamlit untuk menyimpan model YOLO yang dimuat demi efisiensi. Ketika dipanggil dengan , ia mengembalikan contoh model YOLO dari jalur yang ditentukan. Mekanisme penyimpanan data ini menghindari pemuatan ulang model setiap kali fungsi dipanggil, sehingga meningkatkan kinerja selama penggunaan berulang dalam konteks aplikasi [`web.load_model@st.cache_resource`](#)`model_path`.

`@st.cache_resource` : Menyimpan output fungsi untuk meningkatkan kinerja dengan menghindari pemuatan berulang.

`def load_model(model_path)` : Menentukan fungsi yang mengambil argumen `model_path`.

`return YOLO(model_path)` : Memuat dan mengembalikan model YOLO menggunakan jalur yang ditentukan.

```
# 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_HERSHEY_SIMPLEX, 0.5, (0, 255, 0), 2)

file.py > ...
3 def display_results(image, results):
4
5     return image, detected_objects
6 # Main Streamlit app
```

`def display_results(image, results)`: Baris ini mendefinisikan sebuah fungsi bernama `display_results` yang menerima dua parameter:

`image`: Gambar di mana objek terdeteksi.

results: Output dari model deteksi objek, yang berisi informasi tentang objek yang terdeteksi.

boxes = results.boxes.xyxy.cpu().numpy()

scores = results.boxes.conf.cpu().numpy()

labels = results.boxes.cls.cpu().numpy()

names = results.names

i sini, kita mengekstrak berbagai informasi dari objek results:

boxes: Koordinat bounding box untuk objek yang terdeteksi dalam format [x1, y1, x2, y2], yang dikonversi menjadi array NumPy.

scores: Skor kepercayaan untuk setiap deteksi, yang menunjukkan seberapa yakin model terhadap prediksi tersebut.

labels: Indeks kelas yang sesuai dengan setiap objek yang terdeteksi.

names: Daftar atau kamus yang memetakan indeks kelas ke nama kelas (misalnya, "kucing", "anjing").

detected_objects = [] : Sebuah daftar kosong bernama detected_objects diinisialisasi untuk menyimpan nama objek yang terdeteksi yang memenuhi ambang batas kepercayaan tertentu.

for i in range(len(boxes)): Loop ini mengiterasi setiap objek yang terdeteksi berdasarkan jumlah bounding box.

if scores[i] > 0.5: Di dalam loop, kita memeriksa apakah skor kepercayaan untuk deteksi saat ini lebih besar dari 0.5. Ambang batas ini membantu menyaring deteksi dengan kepercayaan rendah.

x1, y1, x2, y2 = boxes[i].astype(int)

label = names[int(labels[i])]

score = scores[i] : Koordinat bounding box diambil dan dikonversi menjadi bilangan bulat.

Label (nama kelas) yang sesuai dengan indeks kelas diambil.

Skor kepercayaan disimpan untuk digunakan nanti.

detected_objects.append(label) : Label dari objek yang terdeteksi ditambahkan ke dalam daftar detected_objects.

cv2.rectangle(image, (x1, y1), (x2, y2), (0, 255, 0), 2) : Menggambar bounding box pada gambar

cv2.putText(image, f"{label}: {score:.2f}", (x1, y1 - 10), cv2.FONT_HERSHEY_SIMPLEX, 0.5, (0, 255, 0), 2) : Menambahkan teks (label dan skor) pada gambar.

return image, detected_objects : Setelah semua deteksi diproses, fungsi ini mengembalikan gambar yang telah diproses serta daftar objek yang terdeteksi.

Fungsi display_result ini sangat berguna untuk memvisualisasikan hasil deteksi objek dalam sebuah gambar dengan menggambar bounding box dan menampilkan informasi terkait setiap objek yang terdeteksi.

```
# Main Streamlit app
def main():
    st.title("Real-time Object Detection with YOLO")
    st.sidebar.title("Settings")

    model_path = "yolo11n.pt" # Path to your YOLO model
    model = load_model(model_path)
```

Fungsi yang merupakan struktur dasar untuk aplikasi Streamlit yang mengimplementasikan deteksi objek secara real-time menggunakan model YOLO.

def main(): Baris ini mendefinisikan fungsi utama dari aplikasi Streamlit. Dalam Python, fungsi didefinisikan dengan kata kunci def diikuti dengan nama fungsi dan tanda kurung.

st.title("Real-time Object Detection with YOLO") : Baris ini menetapkan judul aplikasi Streamlit menjadi "Deteksi Objek Real-time dengan YOLO". Fungsi st.title() digunakan untuk membuat judul di antarmuka web aplikasi.

st.sidebar.title("Settings") : Baris ini membuat sidebar di aplikasi Streamlit dengan judul "Settings" (Pengaturan). Sidebar berguna untuk input pengguna dan konfigurasi tanpa membuat antarmuka utama menjadi berantakan.

model_path = "yololin.pt" : Di sini, sebuah variabel model_path didefinisikan untuk menyimpan jalur ke file model YOLO (dalam hal ini, "yololin.pt"). Model ini akan dimuat nanti untuk keperluan deteksi objek.

model = load_model(model_path) : Baris ini memanggil fungsi load_model() dengan model_path sebagai argumen, yang diharapkan dapat memuat model YOLO dari jalur yang ditentukan. Fungsi ini seharusnya menangani semua pengaturan yang diperlukan untuk menggunakan model dalam prediksi.

```
model = load_model(model_path)

# Create the checkbox once
run_detection = st.sidebar.checkbox("Start/Stop Object Detection", key="detection_control")

# Open video capture if checkbox is active
if run_detection:
    cap = cv2.VideoCapture(0)
    st_frame = st.empty() # Placeholder for video frames
    st_detection_info = st.empty() # Placeholder for detection information

    while True:
        ret, frame = cap.read()
        if not ret:
            st.warning("Failed to capture image.")
            break
```

merupakan bagian dari aplikasi Streamlit untuk deteksi objek menggunakan OpenCV.

run_detection = st.sidebar.checkbox("Start/Stop Object Detection", key="detection control") = Baris ini membuat checkbox di sidebar aplikasi Streamlit. Label untuk checkbox adalah "Start/Stop Object Detection".

key="detection control": Ini adalah pengidentifikasi unik untuk checkbox, yang memungkinkan Streamlit mengelola statusnya secara independen dari widget lain.

run_detection: Variabel ini akan bernilai True jika checkbox dicentang (menunjukkan bahwa deteksi objek harus dimulai) dan False jika tidak dicentang.

if run_detection: Kondisi ini memeriksa apakah checkbox dicentang. Jika iya, blok kode berikutnya akan dieksekusi.

cap = cv2.VideoCapture(0): Baris ini menginisialisasi pengambilan video dari kamera default (biasanya webcam bawaan). Argumen 0 merujuk pada perangkat kamera pertama.

st_frame = st.empty() = ini membuat placeholder kosong dalam aplikasi Streamlit di mana frame video akan ditampilkan.

st_detection_info.empty() = Baris ini tampaknya dimaksudkan untuk membuat placeholder untuk menampilkan informasi deteksi, tetapi tampaknya ada kesalahan di sini. Seharusnya diinisialisasi dengan cara yang mirip dengan st_frame.

while True: Ini menciptakan loop tak terbatas yang akan terus menangkap frame dari kamera selama checkbox tetap dicentang.

ret, frame = cap.read() : Baris ini mencoba membaca sebuah frame dari objek pengambilan video cap. Variabel ret akan bernilai True jika frame berhasil ditangkap dan frame akan berisi data gambar yang sebenarnya.

if not ret: Kondisi ini memeriksa apakah frame tidak berhasil ditangkap.

st.warning("Failed to capture image.") : Jika pengambilan gagal, baris ini menampilkan pesan peringatan di aplikasi Streamlit.

Break : Pernyataan ini keluar dari loop tak terbatas jika pengambilan gagal.

Potongan kode ini mengatur antarmuka sederhana dalam aplikasi Streamlit yang memungkinkan pengguna untuk memulai atau menghentikan proses deteksi objek menggunakan webcam mereka. Ketika diaktifkan, ia terus-menerus menangkap frame dari kamera dan memberikan umpan balik jika pengambilan gagal.

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

display_results: Fungsi ini menggambar hasil deteksi objek pada frame dan mengembalikan frame yang dimodifikasi serta informasi objek yang terdeteksi.

frame: Gambar/video dengan hasil deteksi.

detected_objects: Data tentang objek yang terdeteksi.

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

st_frame.image(...): Fungsi ini digunakan untuk menampilkan gambar atau frame video dalam aplikasi Streamlit.

frame: Variabel ini berisi data gambar yang ingin ditampilkan (biasanya frame yang telah diproses).

channels="RGB": Menunjukkan bahwa data gambar menggunakan format warna RGB (Red, Green, Blue).

use_column_width=True: Mengatur agar gambar menyesuaikan lebar kolom tempat gambar ditampilkan, sehingga responsif terhadap ukuran tampilan.

```
# Display detection information
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 = "No objects detected."

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

if detected_objects:: Memeriksa apakah ada objek yang terdeteksi

object_counts = Counter(detected_objects): Menggunakan Counter dari modul collections untuk menghitung jumlah setiap jenis objek yang terdeteksi.

detection_info = "\n".join(...): Membuat string yang berisi informasi tentang setiap objek dan jumlahnya, dipisahkan dengan baris baru.

else:: Jika tidak ada objek yang terdeteksi, menetapkan detection_info sebagai **"No objects detected."**

st_detection_info.text(detection_info): Memperbarui teks di antarmuka Streamlit untuk menampilkan informasi deteksi yang telah dibuat.

```

# Break the loop if checkbox is unchecked
if not st.session_state.detection_control:
    break

cap.release()

if __name__ == "__main__":
    main()

```

Kode ini menghentikan loop pengambilan video jika checkbox tidak dicentang, melepaskan sumber daya kamera, dan menjalankan fungsi utama jika skrip dijalankan secara langsung.

if not st.session_state.detection_control:: Memeriksa apakah checkbox untuk kontrol deteksi objek tidak dicentang. Jika tidak dicentang (False), maka blok berikutnya akan dieksekusi.

break: Menghentikan loop yang sedang berjalan (misalnya, loop pengambilan video) jika checkbox tidak dicentang.

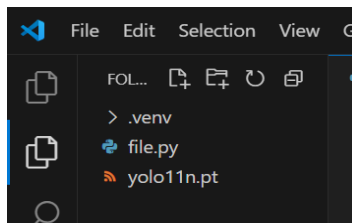
cap.release(): Memanggil metode ini untuk melepaskan sumber daya yang digunakan oleh objek cap, yang merupakan pengambilan video. Ini penting untuk memastikan bahwa kamera dapat digunakan kembali oleh aplikasi lain setelah deteksi selesai.

if __name__ == "__main__": Memeriksa apakah skrip ini dijalankan sebagai program utama. Jika ya, maka blok berikutnya akan dieksekusi.

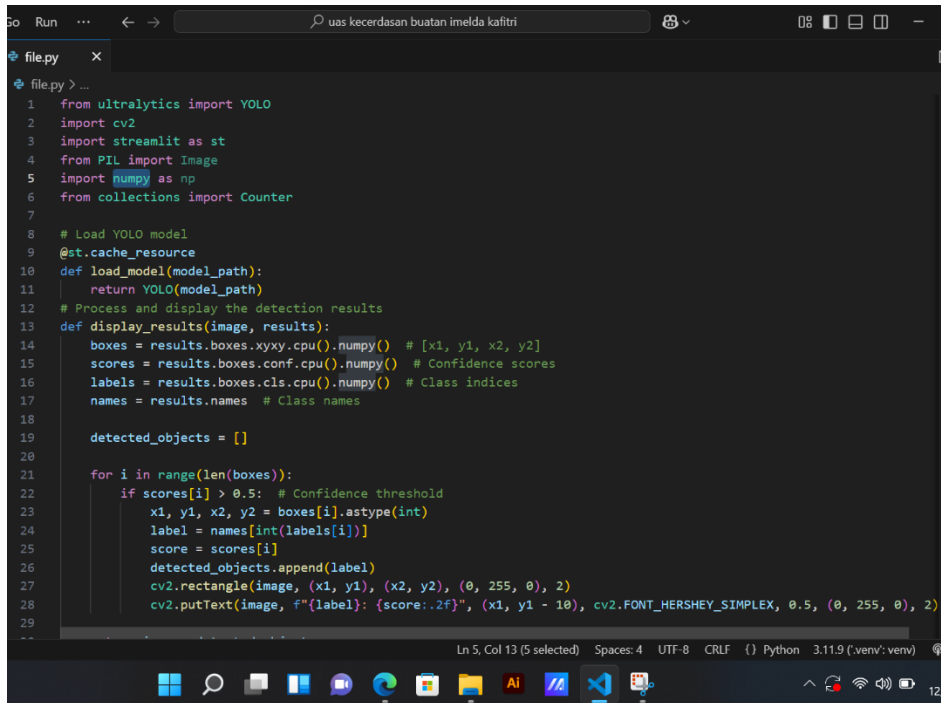
main(): Memanggil fungsi main(), yang biasanya berisi logika utama dari aplikasi.

TUTORIAL DEPLOY

Petama tama kita buat file py dan masukan yolo dalam folder tugas kita.



Lalu setelah itu kita masukan kode kode untuk menjalankan nya

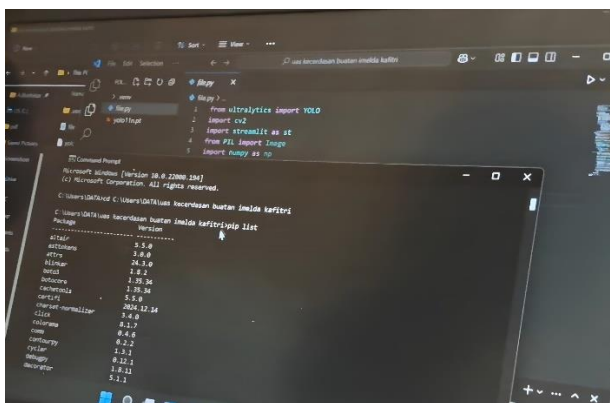


```
1 from ultralytics import YOLO
2 import cv2
3 import streamlit as st
4 from PIL import Image
5 import numpy as np
6 from collections import Counter
7
8 # Load YOLO model
9 @st.cache_resource
10 def load_model(model_path):
11     return YOLO(model_path)
12
13 # Process and display the detection results
14 def display_results(image, results):
15     boxes = results.boxes.xyxy.cpu().numpy() # [x1, y1, x2, y2]
16     scores = results.boxes.conf.cpu().numpy() # Confidence scores
17     labels = results.boxes.cls.cpu().numpy() # Class indices
18     names = results.names # Class names
19
20     detected_objects = []
21
22     for i in range(len(boxes)):
23         if scores[i] > 0.5: # Confidence threshold
24             x1, y1, x2, y2 = boxes[i].astype(int)
25             label = names[int(labels[i])]
26             score = scores[i]
27             detected_objects.append(label)
28             cv2.rectangle(image, (x1, y1), (x2, y2), (0, 255, 0), 2)
29             cv2.putText(image, f"{label}: {score:.2f}", (x1, y1 - 10), cv2.FONT_HERSHEY_SIMPLEX, 0.5, (0, 255, 0), 2)
```

Install terlebih dahulu ultralytics dengan perintah = pip install ultralytics di terminal atau cmd.

Lalu install juga streamlit dengan perintah = pip install streamlit

Kita bisa menggunakan perintah pip list untuk melihat apakah ultralytics dan streamlitnya sudah terinstall



Jika sudah terinstall kita bisa masuk ke cmd dan menulis perintah seperti di bawah ini


```
Command Prompt - streamlit run file.py
Microsoft Windows [Version 10.0.22000.194]
(c) Microsoft Corporation. All rights reserved.

C:\Users\DATA>cd ..

C:\Users>cd ..

C:\>C:\Users\DATA\uas kecerdasan buatan imelda kafitri
'C:\Users\DATA\uas' is not recognized as an internal or external command,
operable program or batch file.

C:\>cd C:\Users\DATA\uas kecerdasan buatan imelda kafitri

C:\Users\DATA\uas kecerdasan buatan imelda kafitri>.venv\scripts\activate

(.venv) C:\Users\DATA\uas kecerdasan buatan imelda kafitri>streamlit run file.py
```

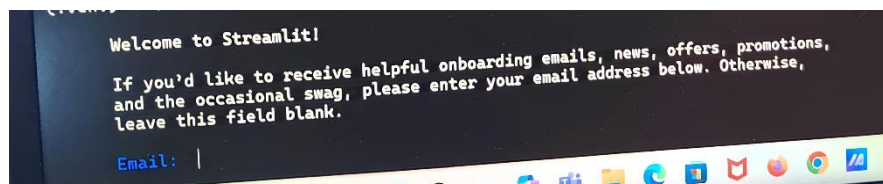
Pertama tama kita sambungkan terlebih dahulu cmdnya dengan folder yang telah kita buat.

Disini nama folder saya uas kecerdasan buatan Imelda kafitri

Lalu setelah itu tambahkan perintah `.venv\scripts\ activate`.

Perintah `.venv\Scripts\activate` (untuk Windows) digunakan untuk mengaktifkan virtual environment di Python, yang berfungsi untuk memisahkan dependensi paket proyek tertentu dari instalasi global. Dengan mengaktifkan virtual environment, Anda dapat mengelola versi pustaka yang spesifik, menghindari konflik antar proyek, dan memastikan bahwa semua perintah Python dan pip merujuk pada lingkungan yang terisolasi, sehingga memudahkan pengembangan, pemeliharaan, dan penyebaran aplikasi.

Lalu setelah itu kita jalankan atau run kan file.py nya



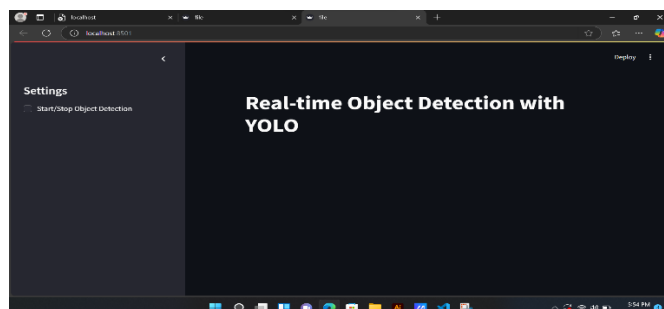
Nanti system akan meminta kita untuk mengisi email, tinggal isi saja email pribadi, lalu enter.

```
(.venv) C:\Users\DATA\uas kecerdasan buatan imelda kafitri>streamlit run file.py

You can now view your Streamlit app in your browser.

Local URL: http://localhost:8502
Network URL: http://172.20.10.4:8502
```

Jika berhasil akan memiliki tampilan seperti ini dan system akan otomatis masuk ke deploy.



Jika ingin melakukan deteksi tekan saja pada start object detection, maka dia akan mulai mendeteksi.

