

UAS KECERDASAN TIRUAN



Nama : Anggun

Nim : 230741118

Prodi : Ilmu Komputer

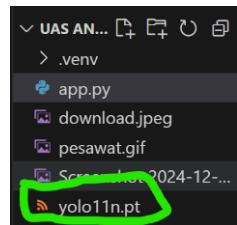
Fakultas : Tekhnik Dan Sains

Dosen Pengampu : Zikri Wahyuzi

Pertama saya membuat folder dulu dengan nama seperti dibawah ini

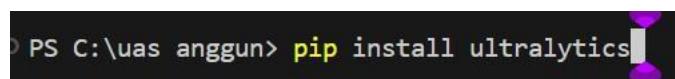


Setelah membuat folder saya langsung membuka aplikasi visual studio code, dan disana saya membuka folder yang telah saya bikin tadi, dan didalam folder tersebut saya akan membuat file yang bernama app.py/file.py dengan kode-kode yang akan dijelaskan dibawah ini.



Lalu saya ambahkan file yolo11 yang sudah saya download ke folder tadi

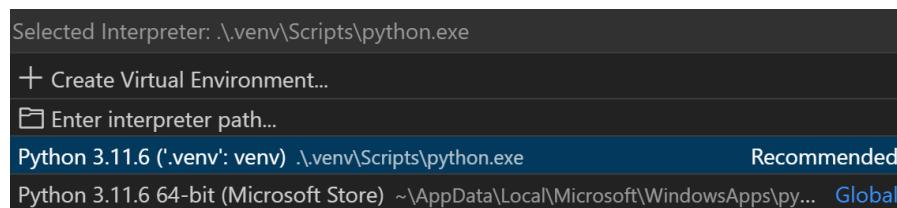
Sebelum memulai saya menginstal dulu library yang akan dibutuhkan



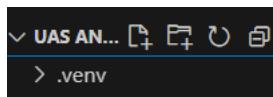
Kode ini untuk menginstal ultralytics



Kode ini untuk menginstal streamlit



Muncul lah :



Setelah menginstall kedua library dan memilih interpreter diatas selanjutnya saya akan menjelaskan kode-kode yang kita gunakan.

```
app.py > ...
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 import base64
```

Kode-kode ini kegunaannya adalah untuk mengimpor library yang akan kita gunakan, fungsinya adalah...

1. from ultralytics import YOLO

Fungsinya : Digunakan untuk mengimpor pustaka YOLO dari Ultralytics, yang menyediakan model deteksi objek yang sudah dilatih sebelumnya atau memudahkan pelatihan model kustom.

2. import cv2

Fungsinya : OpenCV digunakan untuk pengolahan gambar dan video.

3. import streamlit as st

Fungsinya : Membuat antarmuka web interaktif untuk aplikasi berbasis Python.

4. from PIL import Image

Fungsinya : Library Python untuk memanipulasi gambar.

5. import numpy as np

Fungsinya : Mengelola array numerik, terutama untuk pengolahan gambar dan data hasil deteksi.

6. from collections import Counter

Fungsinya : Menghitung jumlah kemunculan elemen dalam daftar.

7. import base64

Fungsinya : Encode dan decode data dalam format base64.

```
app.py > ...
...
9 # Muat model YOLO
10 @st.cache_resource
11 def load_model(model_path):
12     return YOLO(model_path)
13
```

Kode ini merupakan fungsi untuk memuat model YOLO dengan optimasi tertentu menggunakan Streamlit cache. Berikut adalah penjelasan rinci tentang kode ini:

- `@st.cache_resource`:
Streamlit menyimpan model di cache untuk menghindari pemuatan ulang saat aplikasi dijalankan ulang, mempercepat proses.
- `load_model(model_path)`:
Fungsi ini memuat model YOLO dari jalur file (`model_path`), misalnya model yang sudah dilatih seperti `yolov8n.pt`.
- `YOLO(model_path)`:
Membuat instance model YOLO dari pustaka ultralytics untuk digunakan pada deteksi objek.

```
15 # Memproses dan menampilkan hasil deteksi
16 def display_results(image, results):
17     boxes = results.boxes.xyxy.cpu().numpy() # [x1, y1, x2, y2]
18     scores = results.boxes.conf.cpu().numpy() # Skor kepercayaan diri
19     labels = results.boxes.cls.cpu().numpy() # Indeks kelas
20     names = results.names # Nama kelas
```

- `boxes = results.boxes.xyxy.cpu().numpy()`:
 1. Mengambil koordinat kotak pembatas (bounding box) dalam format `[x1, y1, x2, y2]` dari hasil deteksi objek.
 2. Dikonversi ke format NumPy array untuk manipulasi lebih lanjut.
- `scores = results.boxes.conf.cpu().numpy()`:
 1. Mengambil skor kepercayaan (confidence score) dari setiap deteksi objek.
 2. Skor ini menunjukkan seberapa yakin model terhadap deteksi tersebut.
- `labels = results.boxes.cls.cpu().numpy()`:
Mengambil indeks kelas dari setiap deteksi, yang merujuk ke jenis objek yang terdeteksi (misalnya "person" atau "car").
- `names = results.names`:
Mengambil nama kelas (label objek) yang sesuai dengan indeks kelas. Misalnya, indeks 0 mungkin merujuk ke nama "person".

```
detected_objects = []
```

Variabel ini digunakan untuk menyimpan daftar objek yang berhasil terdeteksi oleh model YOLO selama proses deteksi.

```

3     for i in range(len(boxes)):
4         if scores[i] > 0.5: # Ambang batas kepercayaan
5             x1, y1, x2, y2 = boxes[i].astype(int)
6             label = names[int(labels[i])]
7             score = scores[i]
8             detected_objects.append(label)
9             cv2.rectangle(image, (x1, y1), (x2, y2), (0, 255, 0), 2)
10            cv2.putText(image, f'{label}: {score:.2f}', (x1, y1 - 10), cv2.FONT_HERSHEY_SIMPLEX, 0.5, (0, 255, 0), 2)
11
12
13    return image, detected_objects
14

```

- for i in range(len(boxes)) : Melakukan iterasi untuk setiap kotak pembatas (bounding box) yang terdeteksi.
- if scores[i] > 0.5 : Memfilter hasil deteksi hanya jika skor kepercayaan lebih besar dari 0.5.
- Koordinat Bounding Box: x1, y1, x2, y2 = boxes[i].astype(int)
- Nama Label: label = names[int(labels[i])]
- Skor Kepercayaan: score = scores[i]
- detected_objects.append(label) : Menyimpan nama objek yang terdeteksi ke dalam daftar detected_objects.
- Kotak Pembatas: cv2.rectangle(image, (x1, y1), (x2, y2), (0, 255, 0), 2) Menggambar kotak hijau di sekitar objek yang terdeteksi.
- cv2.putText(image, f'{label}: {score:.2f}', ...) : Menampilkan nama label dan skor di atas kotak pembatas.
- return image, detected_objects : Mengembalikan gambar dengan anotasi dan daftar objek yang terdeteksi.

```

app.py > display_results
34
35      # Fungsi untuk menambahkan latar belakang
36      def set_background(image_path):
37          with open(image_path, "rb") as file:
38              base64_image = base64.b64encode(file.read()).decode()
39              css = f"""
40                  <style>
41                      .stApp {{
42                          background-image: url("data:image;base64,{base64_image}");
43                          background-size: cover;
44                          background-repeat: no-repeat;
45                          background-attachment: fixed;
46                      }}
47                  </style>
48                  """
49                  st.markdown(css, unsafe_allow_html=True)
50

```

Kode ini digunakan untuk menambahkan gambar latar belakang ke aplikasi Streamlit:

Baca Gambar:

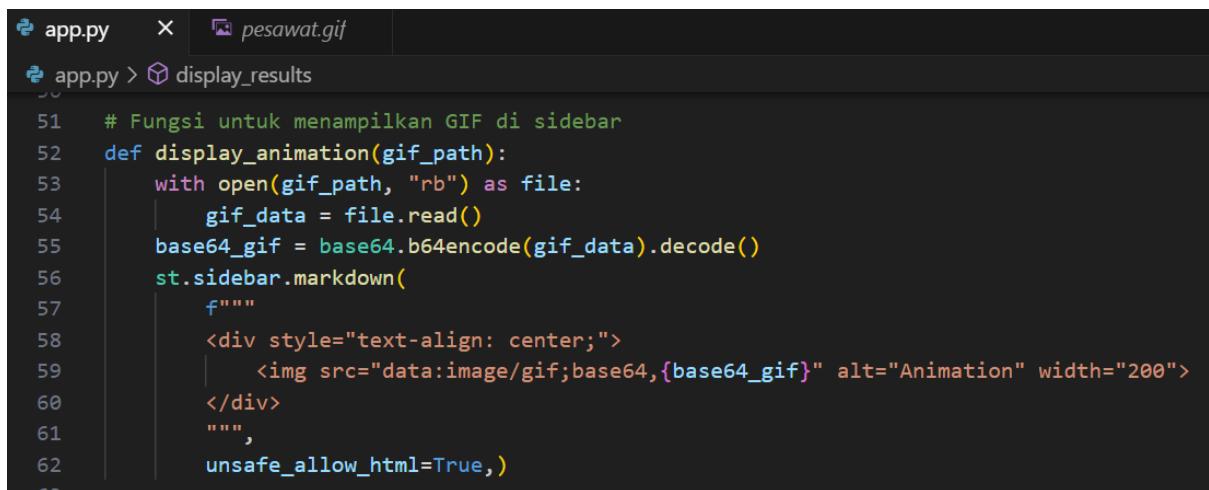
- with open(image_path, "rb") as file: Membuka gambar dalam mode baca biner.
- base64_image: Mengonversi gambar ke format Base64.

Tambahkan CSS:

- Mengatur gambar latar belakang (background-image) dengan CSS:
 - **Ukuran penuh**: background-size: cover;
 - **Tidak berulang**: background-repeat: no-repeat;
 - **Tetap**: background-attachment: fixed;

Terapkan ke Streamlit:

- st.markdown(css, unsafe_allow_html=True): Menyisipkan kode CSS ke dalam aplikasi Streamlit untuk mengatur gambar latar.



The screenshot shows a code editor with two tabs: 'app.py' and 'pesawat.gif'. The 'app.py' tab contains the following code:

```
51 # Fungsi untuk menampilkan GIF di sidebar
52 def display_animation(gif_path):
53     with open(gif_path, "rb") as file:
54         gif_data = file.read()
55     base64_gif = base64.b64encode(gif_data).decode()
56     st.sidebar.markdown(
57         f"""
58             <div style="text-align: center;">
59                 
60             </div>
61             """,
62         unsafe_allow_html=True,
63     )
```

Kode ini menampilkan GIF di sidebar aplikasi Streamlit:

Baca GIF:

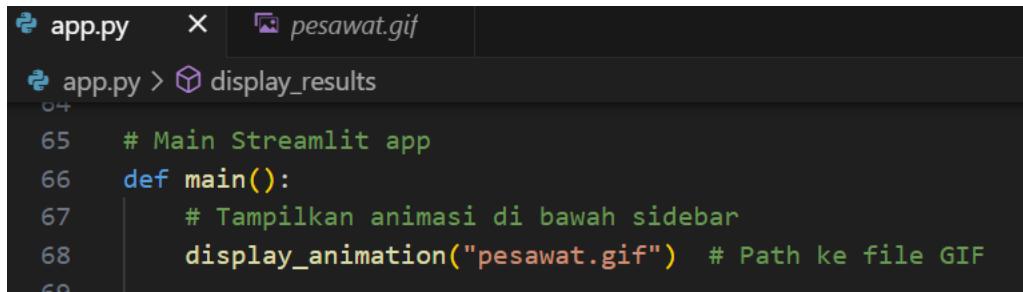
- Membuka file GIF (gif_path) dalam mode baca biner.
- Mengonversi data GIF menjadi format Base64 (base64.b64encode).

Tambahkan ke Sidebar:

- Menyisipkan HTML ke sidebar Streamlit dengan untuk menampilkan GIF.
- Gaya : GIF ditampilkan dengan lebar 200px dan teks di tengah.

Integrasi dengan Streamlit:

- st.sidebar.markdown: Menyisipkan HTML yang sudah diformat ke sidebar.



```
# Main Streamlit app
def main():
    # Tampilkan animasi di bawah sidebar
    display_animation("pesawat.gif") # Path ke file GIF
```

Kode ini adalah Menginisialisasi aplikasi Streamlit dan menampilkan animasi GIF di sidebar menggunakan file "pesawat.gif".

def main():

- Mendefinisikan fungsi utama aplikasi.

display_animation("pesawat.gif")

- Memanggil fungsi display_animation dengan file GIF "pesawat.gif" sebagai argumen.
- GIF akan ditampilkan di sidebar aplikasi Streamlit.



```
# Tampilkan Background
set_background("download.jpeg")

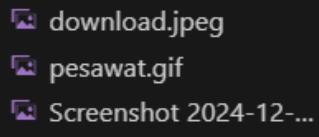
st.title("DETEKSI OBJEK ANGGUN")
st.sidebar.title("Settings⚙️")

model_path = "yolo11n.pt" # Path menuju model YOLO Anda
model = load_model(model_path)
```

Kode ini adalah bagian dari aplikasi Streamlit untuk mengatur tampilan dan memuat model deteksi objek:

- set_background("download.jpeg") : Menetapkan gambar latar belakang aplikasi menggunakan file download.jpeg.
- st.title("DETEKSI OBJEK ANGGUN") : Menampilkan judul utama aplikasi di bagian atas dengan teks "DETEKSI OBJEK ANGGUN".
- st.sidebar.title("Settings ⚙️") : Menampilkan judul di sidebar dengan teks "Settings ⚙️".
- model_path = "yolo11n.pt" : Mendefinisikan path menuju model YOLO yang akan digunakan (yolo11n.pt).
- model = load_model(model_path) : Memuat model YOLO menggunakan fungsi load_model dengan path yang telah ditentukan.

Pada set_background dan display_animation anda bisa menggunakan foto atau gambar yang sudah anda download, dan pada display_animation kalian bisa menambahkan video dengan format (gift) sebagai berikut:



```
# Buat tombol radio untuk kontrol on/off
detection_control = st.sidebar.radio("DETEKSI OBJEK", ("off🔴", "On🟢"), index=0)

# Buka perekaman video jika deteksi diatur ke "On"
if detection_control == "On🟢":
    cap = cv2.VideoCapture(0)
    st_frame = st.empty() # Tempat penampung untuk bingkai video
    st_detection_info = st.empty() # Tempat penampung untuk informasi deteksi

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

Kode ini menambahkan kontrol deteksi objek menggunakan tombol radio pada sidebar Streamlit dan membuka kamera jika deteksi diaktifkan.

- `detection_control = st.sidebar.radio("DETEKSI OBJEK", ("off🔴", "On🟢"), index=0)`
Menampilkan tombol radio di sidebar dengan dua opsi:
 - "off🔴" (deteksi mati secara default).
 - "On🟢" (deteksi aktif).
- `if detection_control == "On🟢":`
Mengeksekusi blok kode berikut jika deteksi diatur ke "On🟢".
- `cap = cv2.VideoCapture(0)`
Membuka kamera perangkat untuk menangkap video.
- `st_frame = st.empty()`: Tempatkan bingkai video di halaman Streamlit.
- `st_detection_info = st.empty()`: Tempatkan informasi hasil deteksi.
- `ret, frame = cap.read()`: Membaca bingkai dari kamera.
Jika gagal membaca bingkai:
`if not ret:` Menampilkan peringatan menggunakan `st.warning` dan keluar dari loop.

```

# Jalankan deteksi YOLO
frame = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB) # Konversi ke RGB untuk tampilan
results = model.predict(frame, imgsz=640) # Melakukan deteksi

# Menggambar hasil dan mengumpulkan objek yang terdeteksi
frame, detected_objects = display_results(frame, results[0])

# Menampilkan umpan video
st_frame.image(frame, channels="RGB", use_column_width=True)

# Menampilkan informasi deteksi
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) # Perbarui teks info deteksi

# Putuskan loop jika deteksi diatur ke "Off"
if detection_control == "off":
    break

cap.release()

```

Kode ini melakukan deteksi objek menggunakan YOLO secara langsung melalui umpan video dari kamera, serta menampilkan hasil deteksi dan informasi objek yang ditemukan di antarmuka **Streamlit**.

1. Konversi Warna Bingkai:

```
frame = cv2.cvtColor(frame, cv2.COLOR_BGR2RGB)
```

Mengonversi bingkai dari format BGR (default OpenCV) ke RGB agar kompatibel dengan tampilan Streamlit.

2. Deteksi Objek dengan YOLO:

```
results = model.predict(frame, imgsz=640)
```

Model YOLO dijalankan pada bingkai video dengan ukuran gambar input yang disesuaikan (imgsz=640).

3. Menggambar Hasil Deteksi:

```
frame, detected_objects = display_results(frame, results[0])
```

- Hasil deteksi diproses oleh fungsi display_results untuk menggambar kotak pembatas (bounding box) di sekitar objek yang terdeteksi.
- Juga, objek yang terdeteksi dikumpulkan dalam daftar detected_objects.

4. Menampilkan Video di Aplikasi Streamlit:

```
st_frame.image(frame, channels="RGB", use_column_width=True)
```

Bingkai video hasil deteksi ditampilkan secara langsung dalam halaman Streamlit.

5. Menghitung dan Menampilkan Informasi Deteksi:

```
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)
```

- Jika ada objek yang terdeteksi:
 - Menghitung jumlah setiap jenis objek menggunakan Counter.
 - Informasi dihimpun dalam format teks yang rapi (misalnya, "Kucing: 2, Anjing: 1").
- Jika tidak ada objek:
 - Menampilkan pesan "No objects detected."
- Informasi ini diperbarui pada antarmuka dengan st_detection_info.text.

6. Menghentikan Deteksi:

```
if detection_control == "off":  
    break
```

Loop dihentikan jika tombol kontrol deteksi di sidebar diatur ke "off".

7. Melepaskan Kamera:

```
cap.release()
```

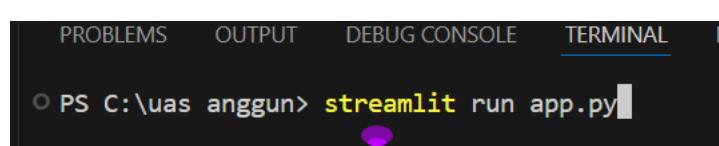
Kamera dilepaskan setelah loop selesai untuk memastikan sumber daya dilepaskan dengan benar.

```
118  
119  if __name__ == "__main__":  
120      main()
```

Kode ini memastikan program hanya berjalan jika file dijalankan langsung. Fungsi main() dipanggil untuk memulai aplikasi utama.

TUTORIAL MENJALANKAN APLIKASI DEPLOY

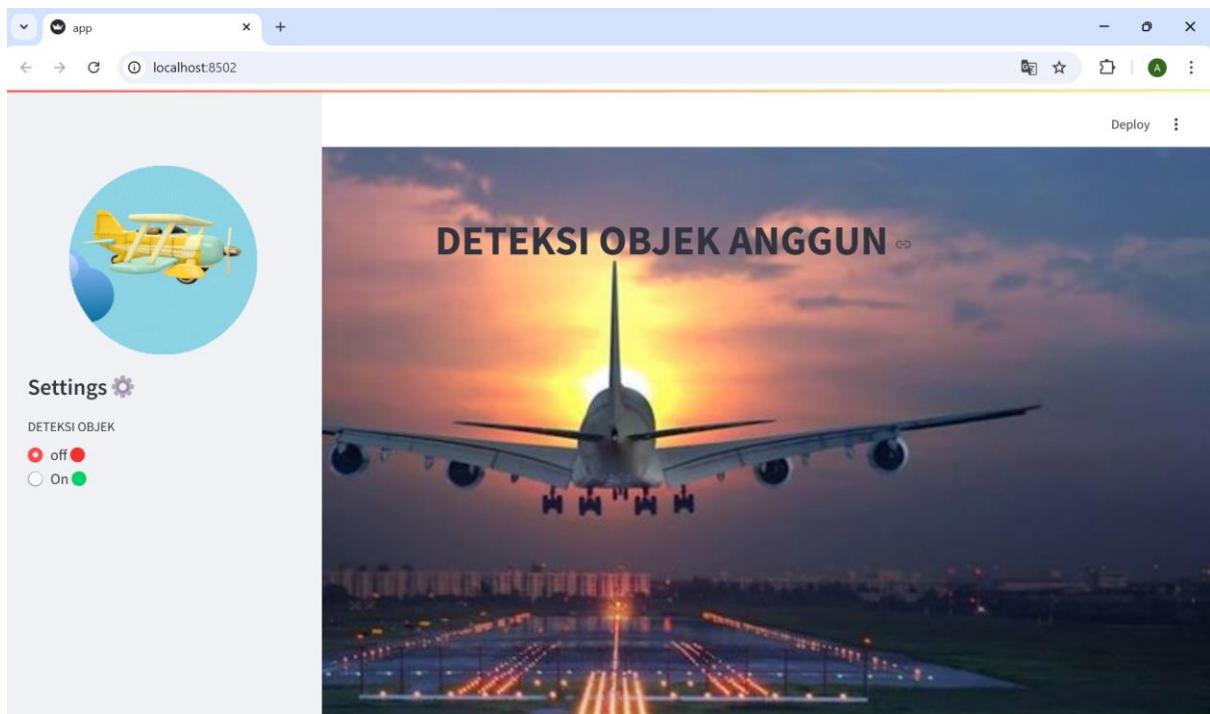
Seperti itulah penjelasan tentang kode-kode yang saya gunakan, sekarang kita beralih ke cara menjalankan kode dan hasilnya :



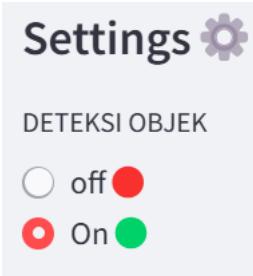
```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL  
PS C:\uas anggun> streamlit run app.py
```

Gunakan perintah ini untuk menjalankan kode-kode diatas

Kemudian saya diminta untuk memasukkan email untuk mengakses streamlit tadi dan akan diarahkan ke browser gmail yang kalian pilih tadi, tampilan nya sebagai berikut:



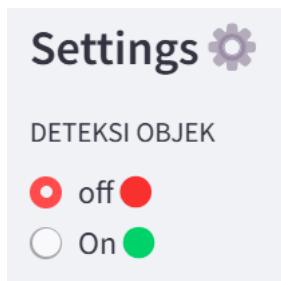
Lalu untuk mendeteksi tekan saja tombol “On” seperti ini :



Lalu tampilannya akan seperti ini :



Dan jika ingin mematikan deteksinya pencet saja tombol “Off” seperti ini :



LINK GITHUB SAYA : <https://github.com/anggunflaa/Uas-Kecerdasan-Tiruan-Anggun-/tree/main>

-TERIMAKASIH-