

# **”TEKNOPLAT”**

## **Manual Sistem Pengenalan Otomatis Plat Nomor Dengan Metode YOLO & EasyOCR**



**Penyusun:**

5200411389    Fahri Putra Herlambang

**PROGRAM STUDI INFORMATIKA  
FAKULTAS SAINS & TEKNOLOGI  
UNIVERSITAS TEKNOLOGI YOGYAKARTA**

**2023/2024**

## Daftar Isi

Daftar Isi .....	i
Daftar Gambar .....	ii
Bagian 1 Pengenalan Antar Muka Sistem ANPR .....	1
1.1    Halaman Awal Upload Image .....	1
1.1.1    Tampilan Antar Muka Halaman Awal Upload Image .....	1
1.1.2    Potongan Kode Program Halaman Beranda / Upload Image .....	2
1.1.3    Potongan Kode Fungsi Create .....	3
1.2    Halaman Database .....	3
1.2.1    Tampilan Antar Muka Halaman Database .....	3
1.2.2    Potongan Kode Program Halaman Database .....	5
1.2.3    Potongan Kode Fungsi Database Read, Update, Delete .....	7
1.3    Halaman <i>Training &amp; Testing</i> .....	7
1.3.1    Tampilan Antar Muka Halaman Training & Testing .....	8
1.3.2    Potongan Kode Program Halaman <i>Training &amp; Testing</i> .....	11
1.3.3    Potongan Kode Program <i>Training &amp; Testing</i> Google Collab .....	12
1.4    Halaman <i>Predict</i> .....	12
1.4.1    Tampilan Antar Muka Halaman Predict .....	13
1.4.2    Potongan Kode Program Halaman Predict .....	14
1.4.3    Potongan Kode Program Fungsi Predict .....	15
1.5    Halaman ANPR .....	18
1.5.1    Tampilan Antar Muka Halaman ANPR .....	18
1.5.2    Potongan Kode Program Halaman ANPR .....	20
1.5.3    Potongan Kode Program Fungsi ANPR .....	21
Bagian 2 Manual Penggunaan Sistem .....	23
2.1    Step by step Proses Pelatihan YoloV5 .....	23
2.2    Step by step Proses Pengujian Deteksi Plat Nomor (Satu File) .....	28
2.3    Step by step Proses Pengujian Deteksi Plat Nomor ( <i>Multiple File</i> ) .....	32
2.4    Step by step Proses Instalasi Program .....	36

## Daftar Gambar

Gambar 1. 1 Tampilan Halaman Upload Image .....	1
Gambar 1. 2 Tampilan Halaman Ketika Upload Gambar.....	2
Gambar 1. 3 Tampilan Halaman Database Bagian Awal .....	3
Gambar 1. 4 Tampilan Halaman Database Bagian Tengah .....	4
Gambar 1. 5 Tampilan Halaman Database Bagian Bawah .....	4
Gambar 1. 6 Tampilan Awal Halaman Google Collab .....	8
Gambar 1. 7 Tampilan Google Collab Ketika Training.....	9
Gambar 1. 8 Tampilan Arsitektur Yolov5 .....	9
Gambar 1. 9 Tampilan Metriks Pelatihan Yolov5 .....	10
Gambar 1. 10 Tampilan Google Collab Ketika Testing .....	10
Gambar 1. 11 Tampilan Google Collab Hasil Testing.....	11
Gambar 1. 12 Tampilan Awal Halaman Predict .....	13
Gambar 1. 13 Tampilan Ketika Melakukan Prediksi.....	13
Gambar 1. 14 Tampilan Hasil Prediksi .....	14
Gambar 1. 15 Tampilan Awal Halaman ANPR.....	18
Gambar 1. 16 Tampilan Ketika Melakukan ANPR .....	19
Gambar 1. 17 Tampilan Hasil Prediksi ANPR .....	19

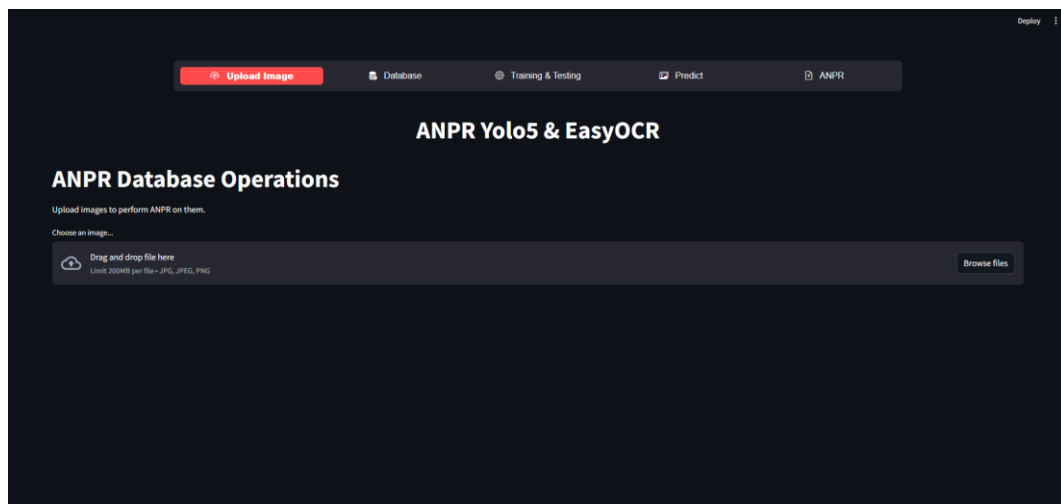
# Bagian 1

## Pengenalan Antar Muka Sistem ANPR

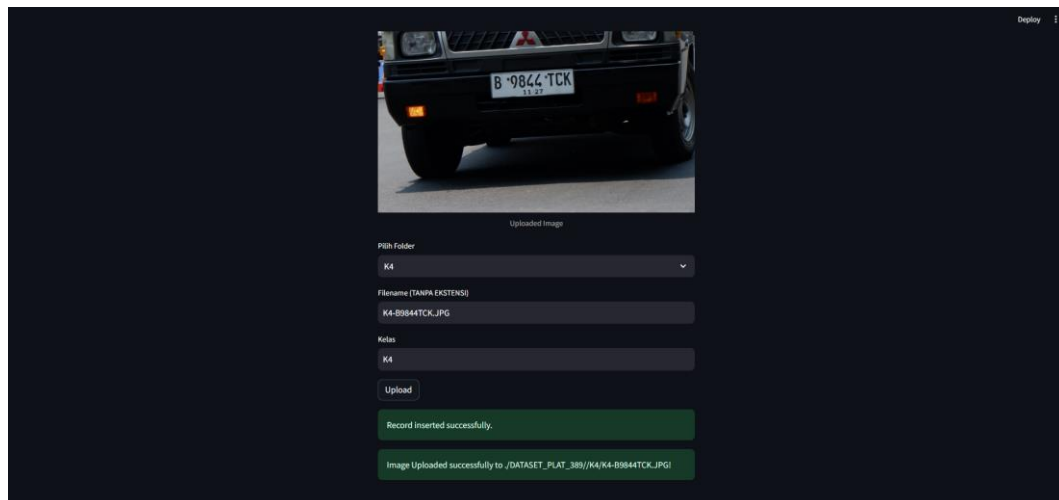
### 1.1 Halaman Awal Upload Image

Halaman awal adalah halaman yang pertama kali muncul saat pengguna menggunakan sistem. Halaman ini memungkinkan pengguna untuk berpindah halaman menggunakan 5 menu yang tersedia pada bagian samping yaitu: Upload Image, Database, Training & Testing, Predict, dan ANPR. Menu “Upload Image” digunakan untuk menampilkan halaman awal dan digunakan sebagai tempat untuk mengupload gambar pada database. Sedangkan detail untuk keempat menu lainnya akan dijelaskan pada bagian berikutnya.

#### 1.1.1 Tampilan Antar Muka Halaman Awal Upload Image



Gambar 1. 1 Tampilan Halaman Upload Image



**Gambar 1. 2 Tampilan Halaman Ketika Upload Gambar**

### **1.1.2 Potongan Kode Program Halaman Beranda / Upload Image**

```

if selected == "Upload Image":
    #title center
    st.markdown("<h1 style='text-align: center; color: White;'>ANPR Yolo5 &
EasyOCR</h1>", unsafe_allow_html=True)
    st.title('ANPR Database Operations')
    st.write('Upload images to perform ANPR on them.')
    uploaded_file = st.file_uploader("Choose an image...", type=["jpg", "jpeg", "png"])
    if uploaded_file is not None:
        image = Image.open(uploaded_file)
        #resize image
        # image = image.resize((720, 720))
        path_image= "./DATASET_PLAT_389/"
        col1, col2,col3 = st.columns(3)
        with col1:
            st.empty()
        with col2:
            st.image(image, caption="Uploaded Image", use_column_width=True)
            selected_folder = st.selectbox("Pilih Folder", ['K1', 'K2', 'K3', 'K4'])
            name_img = st.text_input("Filename (TANPA EKSTENSI)",
value=uploaded_file.name)
            class_img = st.text_input("Kelas", value=selected_folder)
            if st.button("Upload"):
                with st.spinner("Uploading..."):
                    db = connect_db()
                    db.create_record(path_image,selected_folder,name_img,class_img)
                    image.save(os.path.join(path_image, selected_folder, name_img))
                    st.success(f"Image Uploaded successfully to
{path_image}/{selected_folder}/{name_img}!")
                with col3:
                    st.empty()

```

### 1.1.3 Potongan Kode Fungsi Create

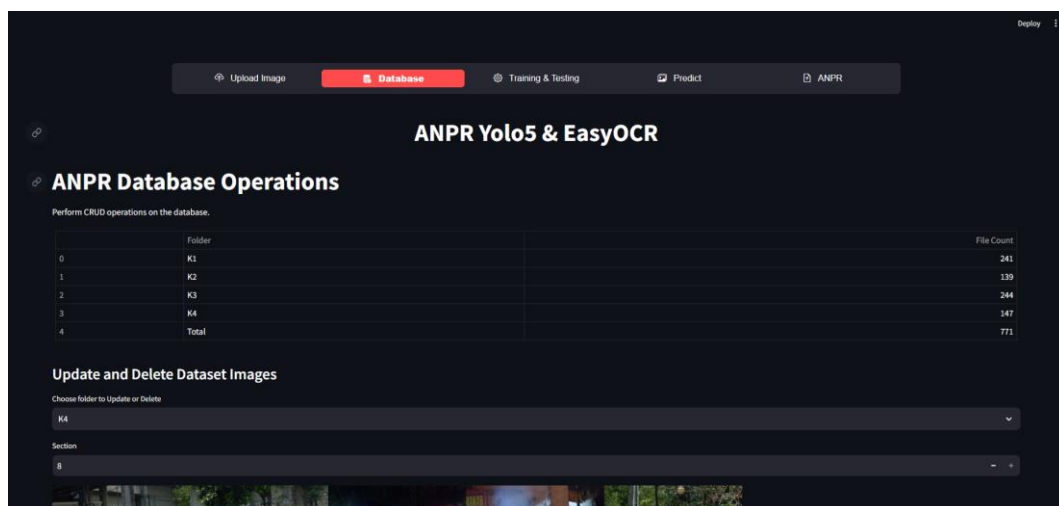
```
def create_record(self, path, folder, filename, class_name):
    sql = "INSERT INTO `image_dataset` (`path`, `folder`, `filename`, `class`)"
    VALUES (%s, %s, %s, %s)"
    val = (path, folder, filename, class_name)
    self.cursor.execute(sql, val)
    self.connection.commit()
    st.success("Record inserted successfully.")
```

## 1.2 Halaman Database

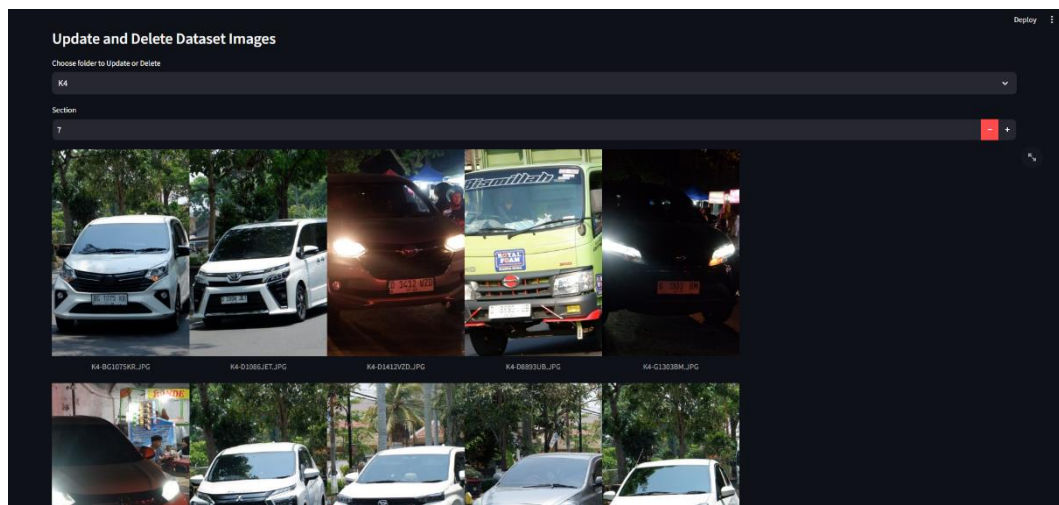
Halaman Database adalah halaman yang digunakan untuk melihat seluruh data citra kendaraan. Halaman ini memuat tabel yang terdiri dari 4 kolom yaitu: “Filename”, dan “Format”.

Pada halaman ini pengguna dapat melihat berapa jumlah data, lalu pengguna juga dapat melihat proporsi data pada setiap folder yang ada. Kemudian pada halaman ini juga menampilkan setiap gambar yang ada pada database. Dengan mengubah folder dan halaman pengguna dapat mencari gambar secara spesifik

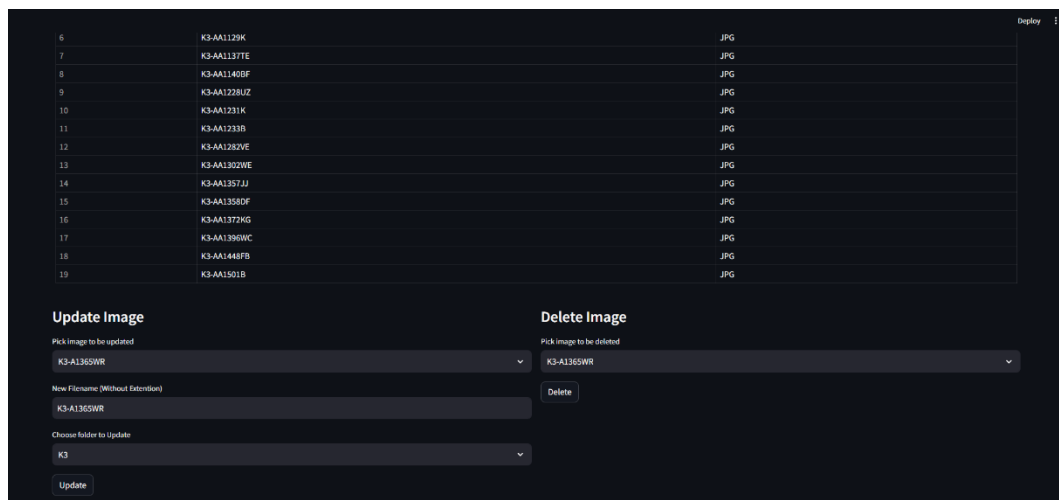
### 1.2.1 Tampilan Antar Muka Halaman Database



Gambar 1. 3 Tampilan Halaman Database Bagian Awal



**Gambar 1. 4 Tampilan Halaman Database Bagian Tengah**



**Gambar 1. 5 Tampilan Halaman Database Bagian Bawah**

### 1.2.2 Potongan Kode Program Halaman Database

```
if selected == "Database":
    #title center
    st.markdown("<h1 style='text-align: center; color: White;'>ANPR Yolo5 &
EasyOCR</h1>", unsafe_allow_html=True)
    st.title('ANPR Database Operations')
    st.write("Perform CRUD operations on the database.")

    # Folders to search for .JPG files
    folders = ['K1', 'K2', 'K3', 'K4']

    # Count image files in each folder
    folder_counts = {}
    total_count = 0
    for folder in folders:
        count = count_image_files(folder)
        folder_counts[folder] = count
        total_count += count

    # Create a DataFrame with folder counts
    df = pd.DataFrame(list(folder_counts.items()), columns=['Folder', 'File Count'])

    # Add total count
    df.loc[len(df.index)] = ['Total', total_count]

    st.table(df)
    st.subheader("Update and Delete Dataset Images")
    # Show a selectbox to choose the folder
    update_delete_folder = st.selectbox("Choose folder to Update or Delete", ['K1', 'K2',
'K3', 'K4'])

    folder_path_for_update_delete = get_folder_path(update_delete_folder)

    # Menyertakan valid_image_extensions untuk mendapatkan daftar file gambar
    valid_image_extensions = ['.jpg', '.jpeg', '.png']
    all_image_files = [file for file in os.listdir(folder_path_for_update_delete) if
os.path.isfile(os.path.join(folder_path_for_update_delete, file)) and
any(file.lower().endswith(ext) for ext in valid_image_extensions)]

    # Pilihan gambar untuk update atau delete
    page_number = st.number_input("Section", min_value=1,
max_value=(len(all_image_files) // 20) + 1, value=1, step=1)
    start_index = (page_number - 1) * 20
    end_index = min(page_number * 20, len(all_image_files))
```



```

# Organize images into rows of five
rows_of_images = [all_image_files[i:i+5] for i in range(start_index, end_index, 5)]

# Display each row of images
for row_images in rows_of_images:
    st.image([os.path.join(folder_path_for_update_delete, image) for image in
row_images], width=250, caption=row_images)

    table_data = {"Filename": [file.split(".")[0] for file in
all_image_files[start_index:end_index]],
"Format": [file.split(".")[1] for file in
all_image_files[start_index:end_index]]}

# Tampilkan data Filename dan format dalam tabel dengan indeks dimulai dari 1
st.table(pd.DataFrame(table_data).reset_index(drop=True))

col1, col2 = st.columns(2)
with col1:
    st.subheader("Update Image")
    # Pilih gambar untuk diupdate
    selected_image_for_update = st.selectbox("Pick image to be updated",
table_data["Filename"])
    if selected_image_for_update is not None:
        db = connect_db()
        #read record if exist and get ID
        get_id = db.read_record(selected_image_for_update)

# Form untuk mengupdate Filename
new_file_name = st.text_input("New Filename (Without Extention)",
selected_image_for_update)
# Show a selectbox to choose the folder
new_class = st.selectbox("Choose folder to Update", ['K1', 'K2', 'K3', 'K4'],
index=folders.index(update_delete_folder))
# Tampilkan tombol update
update_button = st.button("Update")

if update_button:
    if get_id is not None:
        db.update_record(get_id[0],new_file_name,new_class)
    # Lakukan pembaruan Filename
    old_file_path = os.path.join(folder_path_for_update_delete,
f"{selected_image_for_update}.jpg")
    path_image = "/DATASET_PLAT_389/"
    new_file_path = os.path.join(path_image, new_class, f"{new_file_name}.jpg")

```

```
os.rename(old_file_path, new_file_path)
st.success(f"Image Name {selected_image_for_update} updated to
{new_file_name}.jpg")
```

### 1.2.3 Potongan Kode Fungsi Database Read, Update, Delete

```
def read_record(self, filename):
    sql = "SELECT * FROM `image_dataset` WHERE `filename` = %s LIMIT 1"
    self.cursor.execute(sql, (filename,))
    record = self.cursor.fetchone()
    return record

def update_record(self, id, new_filename, new_class_name):
    sql = "UPDATE `image_dataset` SET `folder` = %s, `filename` = %s, `class` = %s
    WHERE `id` = %s LIMIT 1"
    val = (new_class_name, new_filename, new_class_name, id)
    self.cursor.execute(sql, val)
    self.connection.commit()
    st.success("Record updated successfully.")

def delete_record(self, id):
    sql = "DELETE FROM `image_dataset` WHERE `id` = %s LIMIT 1"
    self.cursor.execute(sql, (id,))
    self.connection.commit()
    st.success("Record deleted successfully.")
```

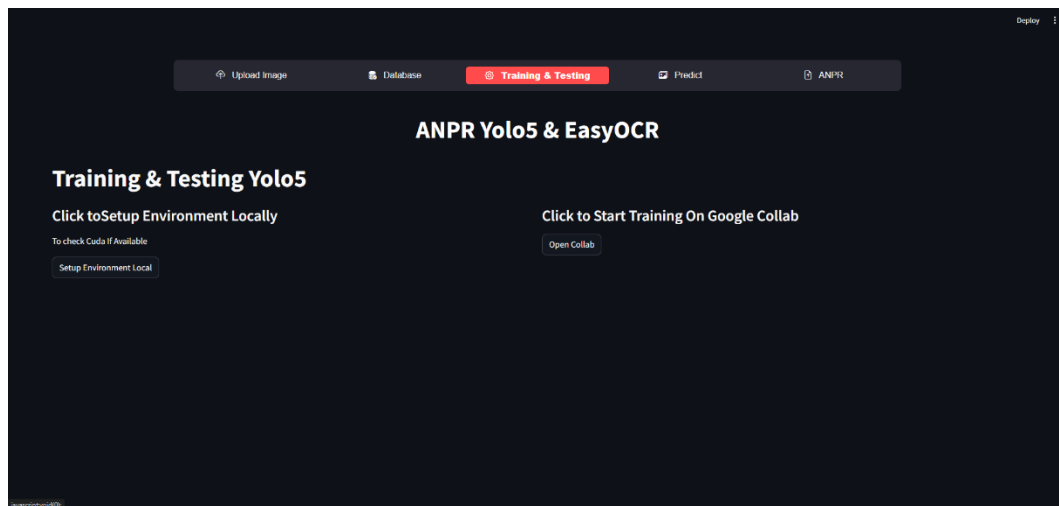
## 1.3 Halaman *Training & Testing*

Halaman *Training & Testing* dapat diakses oleh pengguna dengan memilih menu “*Training & Testing*”. Halaman ini digunakan untuk menjalankan tahap pelatihan pada YOLOv5 yang akan digunakan untuk melakukan pengenalan model plat nomor kendaraan. Sebelum memulai pelatihan, ada beberapa hal yang perlu diperhatikan oleh pengguna antara lain: apakah data sudah dilakukan pelabelan/annotasi, dan beberapa hyperparameter seperti Epoch & Batch. Nilai pada epoch menunjukkan maksimum epoch yang akan dijalankan selama tahap pelatihan. Nilai batch menunjukkan jumlah citra yang akan diinputkan sebagai citra latih dalam satu kali step pelatihan.

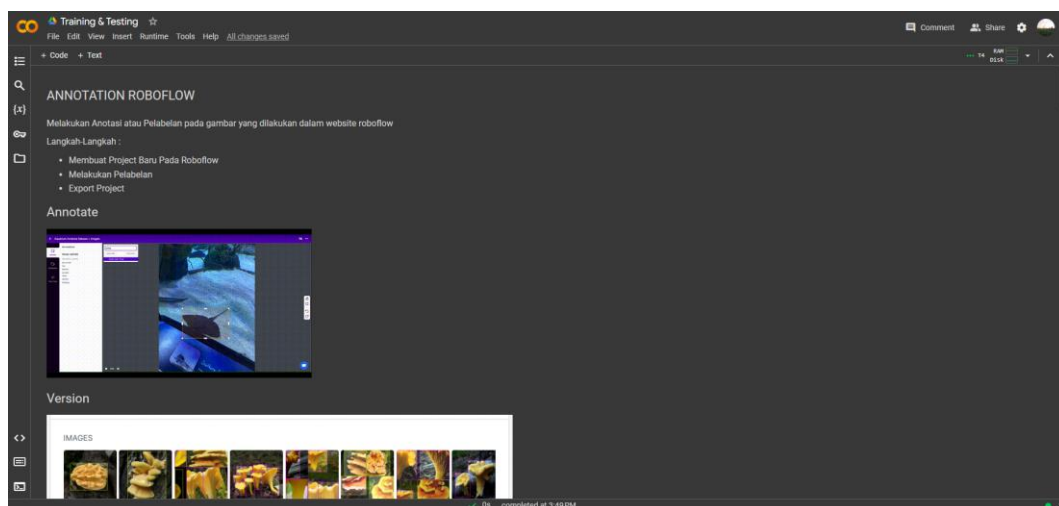
Selain memiliki isian hyperparameter, halaman ini juga memiliki 2 tombol dengan label “*Setup Environment Local*” dan “*Open Collab*” yang masing masing digunakan untuk melakukan pelatihan dan pengujian. Tombol “*Setup Environment*

*Local*” digunakan untuk mempersiapkan device secara local agar dapat melakukan prediksi/deteksi plat nomor, kemudian “*Open Collab*” digunakan untuk membuka google collab yang digunakan sebagai media training. Sebagai luaran dari halaman ini, 3 grafik akan muncul setelah proses pelatihan selesai yang akan ditampilkan pada *Tensorboard*. Grafik ini masing-masing adalah grafik F1-Score dan PRCurve. Pengguna juga dapat melihat ringkasan arsitektur Yolo pada saat Training dan Log Pelatihan pada Tensorboard.

### 1.3.1 Tampilan Antar Muka Halaman Training & Testing



**Gambar 1.1 Tampilan Awal Halaman Pelatihan**



**Gambar 1. 6 Tampilan Awal Halaman Google Collab**

```

python train.py --img 416 --batch 16 --epochs 100 --data (dataset.location)/data.yaml --weights yolov5s.pt --cache

Epoch: 20/99    GPU mem 1.73G box_loss 0.82813 obj_loss 0.80582 cls_loss 0.80292 Instances 4 Size 416 mAP50 100/101 [00:16:00:00, 6.2911/s]
Class Images Instances P 8 mAP50 mAP50-95: 100X 5/5 [00:01:00:00, 3.6311/s]
all 154 154 0.562 0.583 0.58 0.659

Epoch: 21/99    GPU mem 1.73G box_loss 0.82798 obj_loss 0.80419 cls_loss 0.80287 Instances 9 Size 416 mAP50 101/101 [00:16:00:00, 6.0811/s]
Class Images Instances P 8 mAP50 mAP50-95: 100X 5/5 [00:01:00:00, 3.0911/s]
all 154 154 0.564 0.589 0.590 0.711

Epoch: 22/99    GPU mem 1.73G box_loss 0.82825 obj_loss 0.80472 cls_loss 0.80248 Instances 9 Size 416 mAP50 101/101 [00:15:00:00, 6.1211/s]
Class Images Instances P 8 mAP50 mAP50-95: 100X 5/5 [00:01:00:00, 4.0111/s]
all 154 154 0.588 0.587 0.587 0.659

Epoch: 23/99    GPU mem 1.73G box_loss 0.82731 obj_loss 0.80553 cls_loss 0.80282 Instances 5 Size 416 mAP50 101/101 [00:15:00:00, 6.1711/s]
Class Images Instances P 8 mAP50 mAP50-95: 100X 5/5 [00:01:00:00, 3.9411/s]
all 154 154 0.585 0.589 0.581 0.681

Epoch: 24/99    GPU mem 1.73G box_loss 0.827 obj_loss 0.80599 cls_loss 0.8019 Instances 6 Size 416 mAP50 101/101 [00:15:00:00, 6.2711/s]
Class Images Instances P 8 mAP50 mAP50-95: 100X 5/5 [00:01:00:00, 3.9911/s]
all 154 154 0.573 0.592 0.583 0.705

Epoch: 25/99    GPU mem 1.73G box_loss 0.8271 obj_loss 0.80563 cls_loss 0.80262 Instances 4 Size 416 mAP50 101/101 [00:16:00:00, 6.0811/s]
Class Images Instances P 8 mAP50 mAP50-95: 100X 5/5 [00:01:00:00, 2.9711/s]
all 154 154 0.587 0.585 0.585 0.727

Epoch: 26/99    GPU mem 1.73G box_loss 0.82665 obj_loss 0.80587 cls_loss 0.80247 Instances 3 Size 416 mAP50 101/101 [00:15:00:00, 6.1211/s]
Class Images Instances P 8 mAP50 mAP50-95: 100X 5/5 [00:01:00:00, 3.9811/s]
all 154 154 0.581 0.592 0.59 0.72

Epoch: 27/99    GPU mem 1.73G box_loss 0.82699 obj_loss 0.80572 cls_loss 0.80287 Instances 9 Size 416 mAP50 101/101 [00:16:00:00, 6.1811/s]
Class Images Instances P 8 mAP50 mAP50-95: 100X 5/5 [00:01:00:00, 3.9811/s]
all 154 154 0.581 0.592 0.59 0.72
  
```

Gambar 1. 7 Tampilan Google Collab Ketika Training

```

100X 14.10/14.14 [00:00:00:00, 17396/s]
Overriding model.yaml nc=80 with nc=2

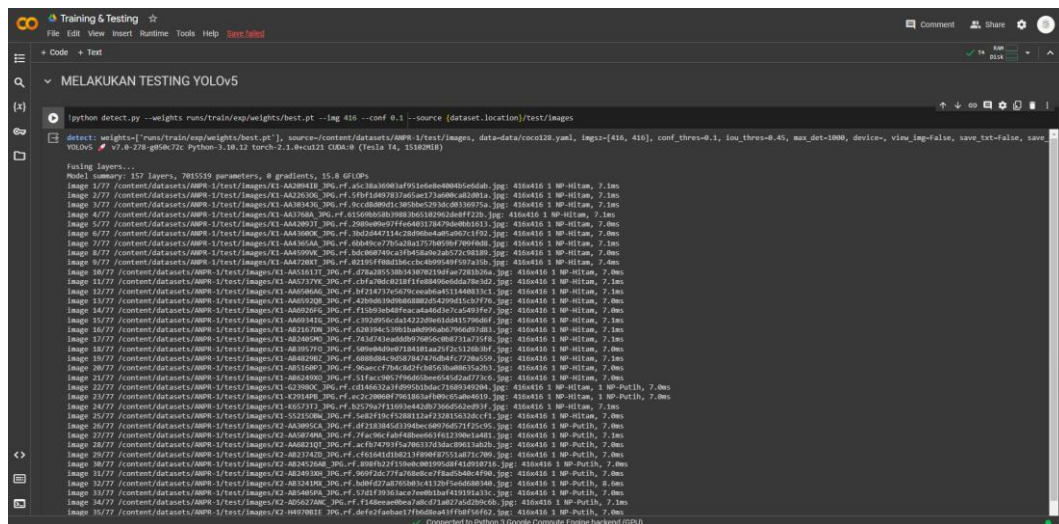
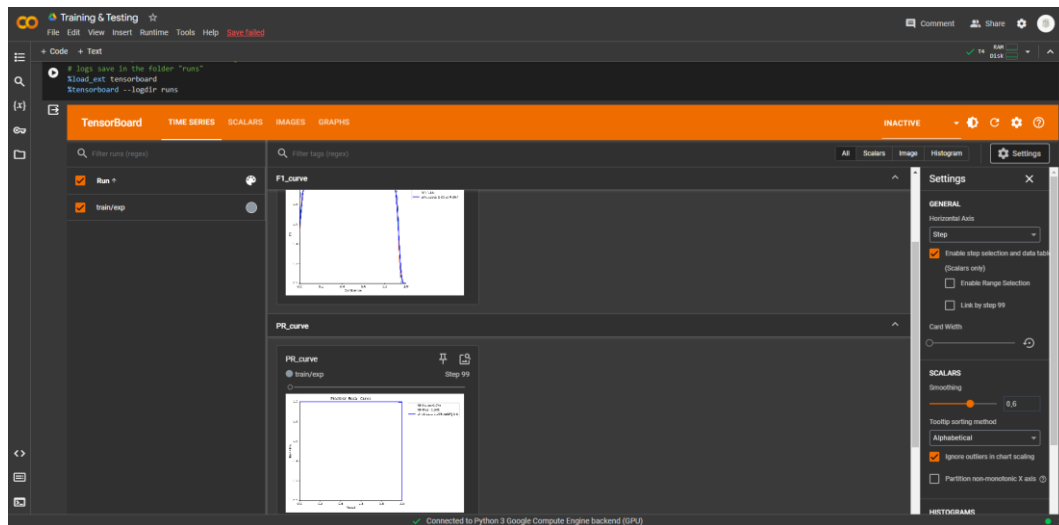
from n params module arguments
1 1 18208 models.common.Conv [1, 32, 6, 2, 2]
2 1 18816 models.common.C3 [64, 64, 1]
3 1 7808 models.common.C3 [64, 128, 1, 2]
4 1 215712 models.common.C3 [128, 128, 1]
5 1 295424 models.common.Conv [128, 256, 3, 2]
6 1 625152 models.common.C3 [256, 256, 1]
7 1 1188672 models.common.Conv [256, 512, 3, 2]
8 1 1182720 models.common.C3 [512, 512, 1]
9 1 656896 models.common.SPP [512, 512, 1]
10 1 131584 models.common.Concat [512, 256, 1, 1]
11 1 0 torch.nn.modules.upsampling.Upsample [None, 1, 'nearest']
12 [-1, 0] 1 0 models.common.Concat [1]
13 1 361888 models.common.C3 [512, 256, 1, False]
14 1 138016 models.common.Conv [256, 128, 1, 1]
15 1 0 torch.nn.modules.upsampling.Upsample [None, 2, 'nearest']
16 [-1, 0] 1 0 models.common.Concat [1]
17 1 98880 models.common.C3 [256, 128, 1, False]
18 1 147712 models.common.Conv [128, 128, 3, 2]
19 [-1, 14] 1 0 models.common.Concat [1]
20 1 296448 models.common.C3 [256, 256, 1, False]
21 1 158416 models.common.Conv [256, 256, 3, 2]
22 [-1, 10] 1 0 models.common.Concat [1]
23 1 1182720 models.common.C3 [512, 512, 1, False]
24 [17, 20, 21] 1 18879 models.yolo.Detect [2, [[16, 13, 36, 30, 33, 21], [30, 61, 62, 45, 59, 119], [116, 90, 156, 104, 373, 320]], [128, 256, 512]]

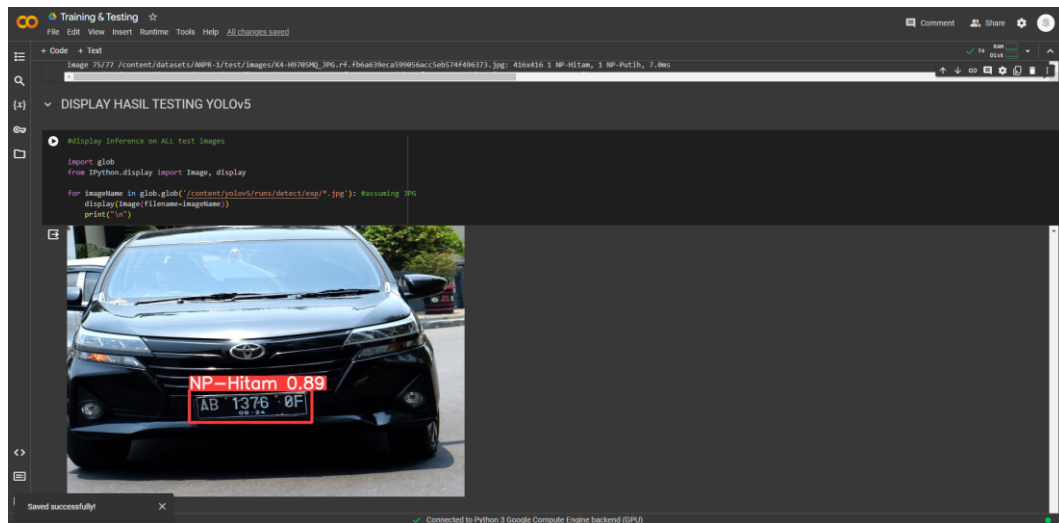
Model summary: 214 layers, 7821823 parameters, 7821823 gradients, 16.0 GFLOPs

Transferred 343/349 items from yolov5s.pt
AMP checks passed
optimizer: SGD(lr=0.01) with parameter groups 57 weight(decay=0.0), 60 weight(decay=0.0005), 60 bias
augmentations: blur(p=0.01, blur_limit=(1, 7)), medianBlur(p=0.01, blur_limit=(3, 7)), ToTensor(p=0.01), CLAHE(p=0.01, clip_limit=(1, 4.0), tile_grid_size=(8, 8))
train: Scanning /content/datasets/AMP9-1/train/labels.cache
train: New cache created: /content/datasets/AMP9-1/train/labels.cache
train: Caching images (0.000 rows): 100X 100/100 [00:12:00:00, 130.5411/s]
val: Scanning /content/datasets/AMP9-1/valid/labels.cache
val: New cache created: /content/datasets/AMP9-1/valid/labels.cache
val: Caching images (0.108 rows): 100X 154/154 [00:02:00:00, 67.8911/s]

AutoAnchor: 4.02 anchors/target, 1.888 Best Possible Recall (BPR). Current anchors are a good fit to dataset
Plotting labels to runs/train/amp/labels.jpg...
  
```

Gambar 1. 8 Tampilan Arsitektur Yolov5





**Gambar 1. 11 Tampilan Google Collab Hasil Testing**

### 1.3.2 Potongan Kode Program Halaman *Training & Testing*

```

if selected == "Training & Testing":
    st.title('Training & Testing Yolo5')
    if 'trainer' not in st.session_state:
        st.session_state.trainer = None
    col1, col2 = st.columns(2)
    with col1:
        st.subheader("Click to Setup Environment Locally")
        st.write("To check Cuda If Available")
        if st.button("Setup Environment Local"):
            if st.session_state.trainer == None:
                st.session_state.trainer = Train()
                st.write(f"Setup complete. Using torch {torch.__version__}
                ({torch.cuda.get_device_properties(0).name if torch.cuda.is_available() else
                'CPU'})")
            else:
                st.warning("Please Setup Environment Locally.")
    with col2:
        st.subheader("Click to Start Training On Google Collab")
        if st.button("Open Collab"):
            # Link to your Google Colab notebook
            colab_link =
            "https://colab.research.google.com/drive/1AQkYvTr0JdS_fsCEO120VUbYHOn5bBYR?
            usp=sharing"

            # Open the link in a new tab when the button is clicked
            webbrowser.open_new_tab(colab_link)
            st.success("Training started. Please check the Colab notebook for progress.")

```

### 1.3.3 Potongan Kode Program *Training & Testing* Google Collab

```
!pip install roboflow
from roboflow import Roboflow
rf = Roboflow(api_key="2KkH5m1vRk6crjqB6QHj")
project = rf.workspace("universitas-teknologi-yogyakarta-luprw").project("anpr-zlpm1")
dataset = project.version(1).download("yolov5")
```

```
!python train.py --img 416 --batch 16 --epochs 100 --data {dataset.location}/data.yaml -
-weights yolov5s.pt --cache
```

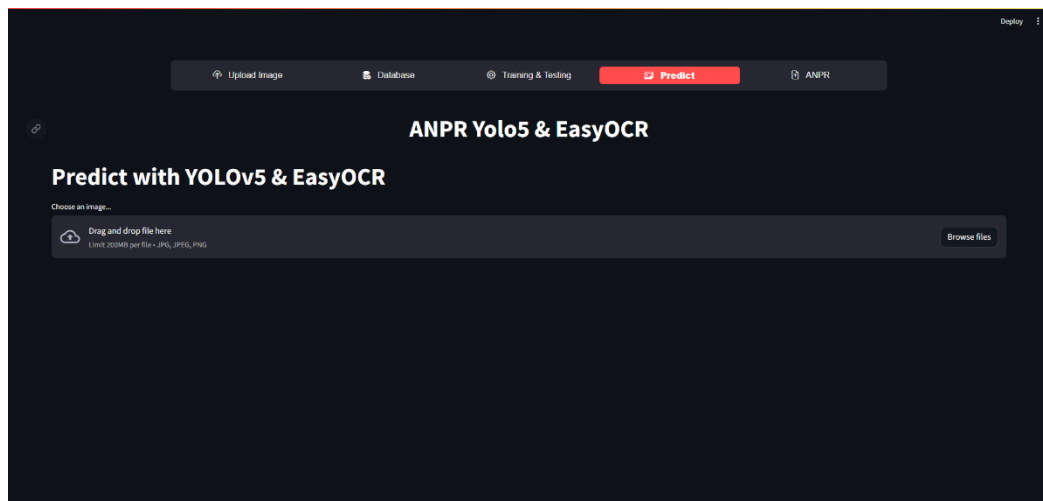
```
!python detect.py --weights runs/train/exp/weights/best.pt --img 416 --conf 0.1 --
source {dataset.location}/test/images
```

## 1.4 Halaman *Predict*

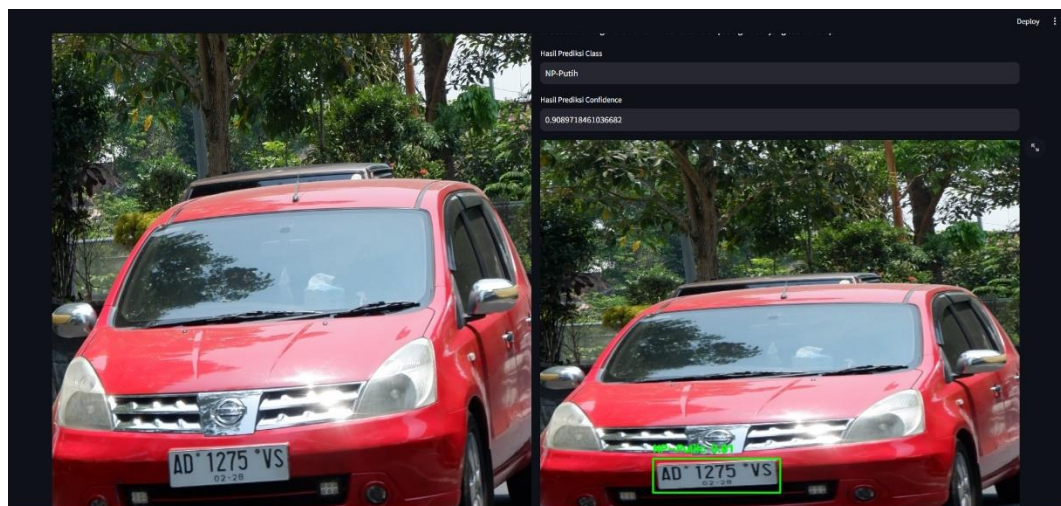
Halaman *Predict* adalah halaman yang digunakan untuk melihat proses deteksi plat nomor dan rekognisi plat nomor secara detail. Pada halaman ini pengguna akan melakukan upload gambar yang akan dilakukan prediksi, setelah pengguna melakukan upload gambar maka akan muncul sebuah button “Predict”.

Setelah button predict ini ditekan maka akan muncul step-step rekognisi plat nomor step tersebut bermula dari deteksi plat nomor yang dilakukan oleh Yolov5, kemudian dilakukan Segmentasi pada plat nomor tersebut, kemudian setelah mendapatkan gambar plat nomor, dilakukan beberapa proses *preprocessing* sehingga plat nomor tersebut menjadi gambar hitam putih, kemudian dengan menggunakan EasyOCR dilakukan deteksi karakter pada plat nomor tersebut, hasil deteksi tersebut kemudian di urutkan dan dilakukan filtering sehingga menghasilkan hanya huruf dan angka yang sesuai dengan plat nomor.

### 1.4.1 Tampilan Antar Muka Halaman Predict

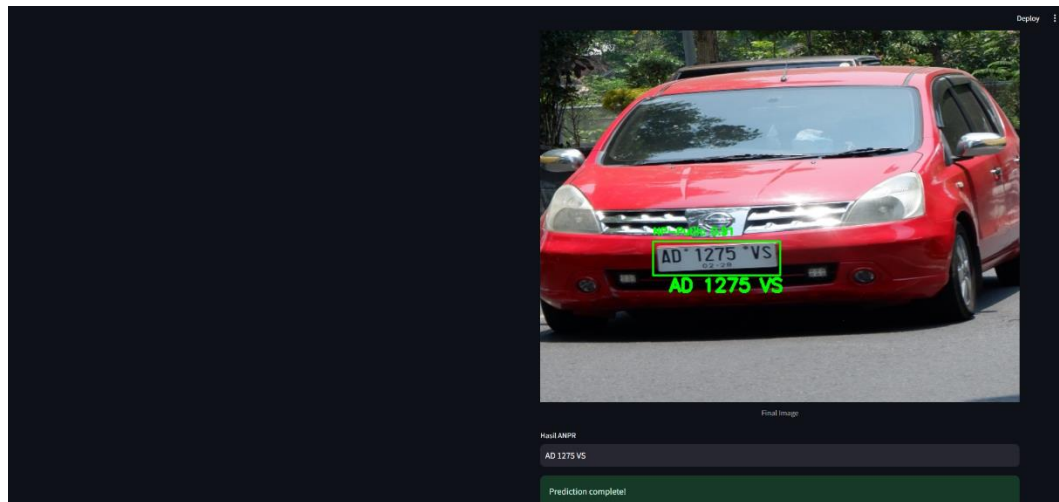


Gambar 1. 12 Tampilan Awal Halaman Predict



Gambar 1. 13 Tampilan Ketika Melakukan Prediksi





**Gambar 1. 14 Tampilan Hasil Prediksi**

#### **1.4.2 Potongan Kode Program Halaman Predict**

```

if selected == 'Predict':
    #title center
    st.markdown("<h1 style='text-align: center; color: White;'>ANPR Yolo5 &
EasyOCR</h1>", unsafe_allow_html=True)
    st.title('Predict with YOLOv5 & EasyOCR')
    anpr = ANPR()

    uploaded_file = st.file_uploader("Choose an image...", type=["jpg", "jpeg", "png"])

    if uploaded_file is not None:
        image = Image.open(uploaded_file)
        image_in = cv2.cvtColor(np.array(image), cv2.COLOR_RGB2BGR)
        col1, col2 = st.columns(2)
        with col1:
            st.image(image, caption="Uploaded Image", use_column_width=True)
            resized_image = None
            annotated_image = None
            st.subheader("Click to Predict with YOLOv5")
            button_pred = st.button("Predict")
        with col2:
            if button_pred:
                with st.spinner("Predicting..."):
                    model_path = "best_V5.pt" # Replace this with your model's path
                    resized_image, annotated_image, result_anpr =
anpr.predict_image(image_in, model_path=model_path)
                    st.text_input("Hasil ANPR", value=result_anpr)
                    st.success("Prediction complete!")

```

### 1.4.3 Potongan Kode Program Fungsi Predict

```
import streamlit as st
import cv2
import torch
import numpy as np
import easyocr
import re

@st.cache_resource()
class ANPR:
    def __init__(self):
        self.device = "cuda" if torch.cuda.is_available() else "cpu"
        print("Using Device:", self.device)
        self.model = None

    def load_model(self, model_path):
        self.model = torch.hub.load('ultralytics/yolov5', 'custom', path=model_path,
force_reload=True)

    def Prep_Hitam_Putih(self, cropped_image, label):
        if label == 'NP-Hitam':
            cropped_image = cv2.cvtColor(cropped_image, cv2.COLOR_BGR2GRAY) #
Convert to grayscale
            st.image(cropped_image, caption='Grayscale Image', use_column_width=True)
            st.write("Pada gambar diatas merupakan hasil konversi gambar dari RGB ke
Grayscale. Konversi ini dilakukan untuk mempermudah proses preprocessing
selanjutnya.")
            cropped_image = cv2.convertScaleAbs(cropped_image, alpha=0.85, beta=0.1) #
decrease contrast
            st.image(cropped_image, caption='Decrease Contrast Image',
use_column_width=True)
            st.write("Pada gambar diatas merupakan hasil penurunan kontras dari gambar
yang telah dikonversi ke Grayscale. Penurunan kontras ini dilakukan untuk
mempermudah proses preprocessing selanjutnya.")
            cropped_image = cv2.blur(cropped_image,(2,2)) # apply blur to image
            st.image(cropped_image, caption='Blur Image', use_column_width=True)
            st.write("Pada gambar diatas merupakan hasil blur dari gambar yang telah
dikonversi ke Grayscale dan diturunkan kontrasnya. Blur ini dilakukan untuk
mempermudah proses preprocessing selanjutnya.")
            cropped_image = cv2.bilateralFilter(cropped_image, d=4, sigmaColor=8,
sigmaSpace=12) # Bilateral filter for noise reduction
            st.image(cropped_image, caption='Bilateral Filter Image',
use_column_width=True)
            st.write("Pada gambar diatas merupakan hasil bilateral filter dari gambar yang
telah dikonversi ke Grayscale, diturunkan kontrasnya, dan di blur. Bilateral filter ini
dilakukan untuk mempermudah proses preprocessing selanjutnya.")
```

```

        cropped_image = cv2.threshold(cropped_image, 10, 255,
cv2.THRESH_BINARY_INV | cv2.THRESH_OTSU)[1] # apply binary threshold
inverse
        st.image(cropped_image, caption='Binary Threshold Inverse Image',
use_column_width=True)
        st.write("Pada gambar diatas merupakan hasil binary threshold inverse dari
gambar yang telah dikonversi ke Grayscale, diturunkan kontrasnya, di blur, dan di
bilateral filter. Binary threshold inverse ini dilakukan untuk mempermudah proses
preprocessing selanjutnya.")
        cropped_image = cv2.bitwise_not(cropped_image) # Invert colors
        st.image(cropped_image, caption='Invert Colors Image',
use_column_width=True)
        st.write("Pada gambar diatas merupakan hasil invert colors dari gambar yang
telah dikonversi ke Grayscale, diturunkan kontrasnya, di blur, di bilateral filter, dan
di binary threshold inverse. Invert colors ini dilakukan untuk mempermudah proses
preprocessing selanjutnya.")
        cropped_image = cropped_image[0:cropped_image.shape[0] - 10,
0:cropped_image.shape[1]] #menghapus tanggal pajak
        elif label == 'NP-Putih':
            cropped_image = cv2.cvtColor(cropped_image, cv2.COLOR_BGR2GRAY) #
Convert to grayscale
            st.image(cropped_image, caption='Grayscale Image', use_column_width=True)
            st.write("Pada gambar diatas merupakan hasil konversi gambar dari RGB ke
Grayscale. Konversi ini dilakukan untuk mempermudah proses preprocessing
selanjutnya.")
            cropped_image = cv2.convertScaleAbs(cropped_image, alpha=1.15, beta=0) #
Increase contrast
            st.image(cropped_image, caption='Increase Contrast Image',
use_column_width=True)
            st.write("Pada gambar diatas merupakan hasil peningkatan kontras dari
gambar yang telah dikonversi ke Grayscale. Peningkatan kontras ini dilakukan untuk
mempermudah proses preprocessing selanjutnya.")
            cropped_image = cv2.blur(cropped_image,(2,2)) #apply blur to image
            st.image(cropped_image, caption='Blur Image', use_column_width=True)
            st.write("Pada gambar diatas merupakan hasil blur dari gambar yang telah
dikonversi ke Grayscale dan ditingkatkan kontrasnya. Blur ini dilakukan untuk
mempermudah proses preprocessing selanjutnya.")
            cropped_image = cv2.bilateralFilter(cropped_image, d=4, sigmaColor=8,
sigmaSpace=12) # Bilateral filter for noise reduction
            st.image(cropped_image, caption='Bilateral Filter Image',
use_column_width=True)
            st.write("Pada gambar diatas merupakan hasil bilateral filter dari gambar yang
telah dikonversi ke Grayscale, ditingkatkan kontrasnya, dan di blur. Bilateral filter ini
dilakukan untuk mempermudah proses preprocessing selanjutnya.")
            cropped_image = cv2.threshold(cropped_image, 10, 255,

```

```

cv2.THRESH_BINARY | cv2.THRESH_OTSU)[1] # apply binary treshold inverse
    st.image(cropped_image, caption='Binary Threshold Image',
use_column_width=True)
    st.write("Pada gambar diatas merupakan hasil binary treshold dari gambar
yang telah dikonversi ke Grayscale, ditingkatkan kontrasnya, di blur, dan di bilateral
filter. Binary treshold ini dilakukan untuk mempermudah proses preprocessing
selanjutnya.")
    cropped_image = cropped_image[0:cropped_image.shape[0] - 10,
0:cropped_image.shape[1]] #menghapus tanggal pajak

else:
    print("Label Tidak Diketahui")
    return None

# Display the resulting cropped image
return cropped_image
def read_text(self, result_image):
    reader = easyocr.Reader(['id'], gpu=True)
    results = reader.readtext(result_image)
    return results

# Function to sort OCR results based on X-axis coordinates
def sort_by_x_axis(self, results):
    return sorted(results, key=lambda x: x[0][0][0])

# Sort OCR results by X-axis coordinates
def filter_results(self, results):
    sorted_results = self.sort_by_x_axis(results)
    plate_number = ""

    for result in sorted_results:
        text = re.sub(r'^A-Za-z0-9', "", result[1]).upper() # Remove special characters

        # Condition to include shorter alpha sequences (<= 2 characters)
        if text.isalpha() and len(text.replace(" ", "")) <= 2:
            filter = ".join([char if char.isalnum() or char == ' ' else " for char in text]) + ' '
            plate_number += filter
        # Conditions for other types of sequences (digits, longer alpha sequences, etc.)
        elif len(text.replace(" ", "")) >= 4:
            filter = ".join([char if char.isalnum() or char == ' ' else " for char in text]) + ' '
            plate_number += filter
        elif text.isdigit() and len(plate_number.replace(" ", "")) <= 4:
            filter = ".join([char if char.isalnum() or char == ' ' else " for char in text]) + ' '
            plate_number += filter
        elif text.isalpha() and len(text.replace(" ", "")) <= 3:
            filter = ".join([char if char.isalnum() or char == ' ' else " for char in text]) + ' '
            plate_number += filter

```

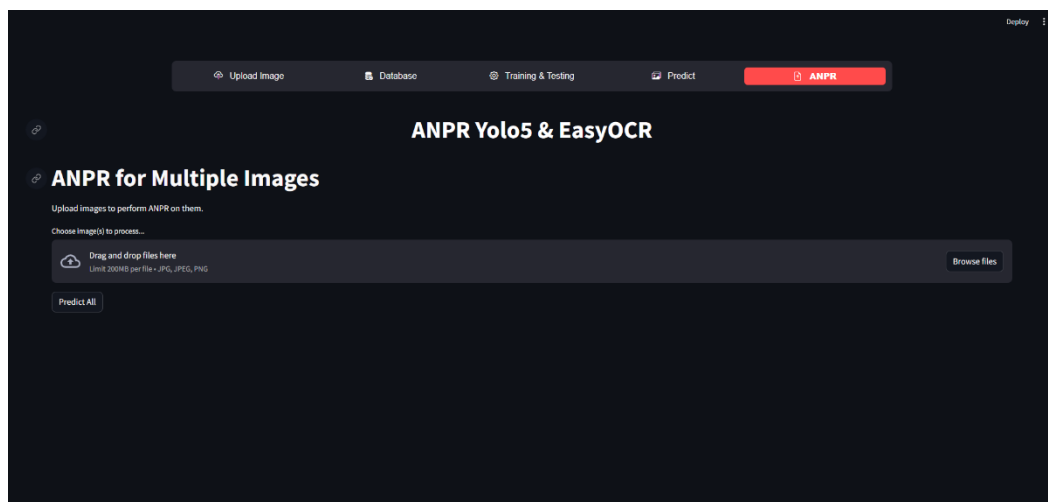
## 1.5 Halaman ANPR

Pada halaman ANPR ini pengguna dapat melakukan prediksi dan rekognisi plat nomor dengan menggunakan banyak file gambar sekaligus, sehingga memudahkan pengguna untuk melakukan deteksi terhadap banyak kendaraan.

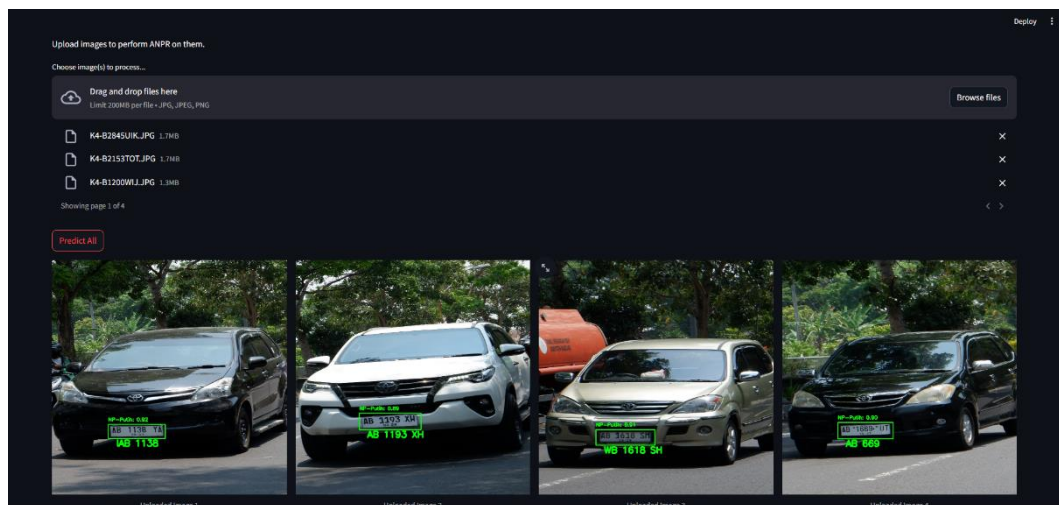
Pada halaman ANPR ini sama seperti halaman Predict namun pada halaman ANPR ini tidak menampilkan secara detail proses deteksi dan proses rekognisi karakter yang dilaksanakan sehingga membuat tampilan yang lebih sederhana.

Lalu pada halaman ANPR ini pengguna dapat menyimpan hasil prediksi dalam bentuk file txt. Pada file tersebut akan berisi nama file gambar dan hasil prediksinya

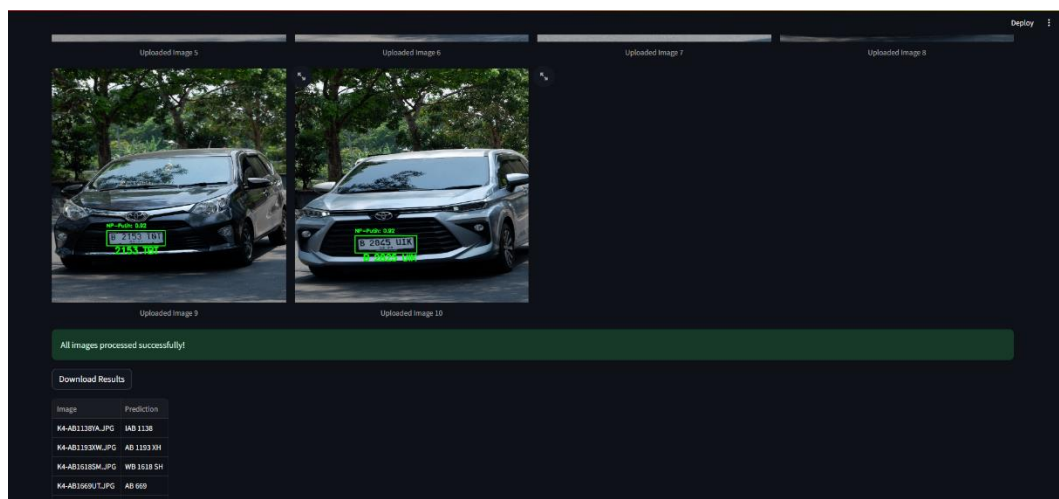
### 1.5.1 Tampilan Antar Muka Halaman ANPR



Gambar 1. 15 Tampilan Awal Halaman ANPR



Gambar 1. 16 Tampilan Ketika Melakukan ANPR



Gambar 1. 17 Tampilan Hasil Prediksi ANPR

### 1.5.2 Potongan Kode Program Halaman ANPR

```
if selected == 'ANPR':
    st.title('ANPR for Multiple Images')
    st.write('Upload images to perform ANPR on them.')

    model_path = "best_V5.pt" # Replace this with your model's path
    anpr_folder = ANPR_Folder()

    uploaded_files = st.file_uploader("Choose image(s) to process...", type=["jpg", "jpeg",
"png"], accept_multiple_files=True)

    if uploaded_files is not None:
        all_results = []
        if st.button("Predict All"):
            with st.spinner("Predicting..."):
                images_count = len(uploaded_files)
                images_per_row = 4
                rows_count = -(images_count // images_per_row) # Ceiling division to get
the total number of rows

                for i in range(rows_count):
                    cols = st.columns(images_per_row)
                    for j in range(images_per_row):
                        idx = i * images_per_row + j
                        if idx < images_count:
                            uploaded_file = uploaded_files[idx]
                            image = Image.open(uploaded_file)
                            image = np.array(image)

                            resized_image, result_text, class_label =
anpr_folder.predict_image(image, model_path) # Provide model_path
                            all_results.append({'Image': uploaded_file.name, 'Prediction':
result_text})

                            # Display uploaded image
                            with cols[j]:
                                st.image(resized_image, caption=f"Uploaded Image {idx + 1}",
use_column_width=True)
                            st.success("All images processed successfully!")
                            # Save results to a text file
                            results_text = ""
                            for result in all_results:
                                results_text += f"Filename {result['Image']} => Prediction
{result['Prediction']}\n"
```

```

dt_now = dt.datetime.now().strftime("%d-%m-%Y_%H-%M")
st.download_button(label="Download Results", data=result_text,
file_name=f"results_{images_count}_{dt_now}.txt", mime="text/plain")
st.dataframe(all_results)

```

### 1.5.3 Potongan Kode Program Fungsi ANPR

```

import streamlit as st
import cv2
import torch
import easyocr
import re
from io import BytesIO

@st.cache_resource()
class ANPR_Folder:
    def __init__(self):
        self.device = "cuda" if torch.cuda.is_available() else "cpu"
        print("Using Device:", self.device)
        self.model = None

    def load_model(self, model_path):
        self.model = torch.hub.load('ultralytics/yolov5', 'custom', path=model_path,
force_reload=True)

    def Prep_Hitam_Putih(self, cropped_image, label):
        if label == 'NP-Hitam':
            cropped_image = cv2.cvtColor(cropped_image, cv2.COLOR_BGR2GRAY) #
Convert to grayscale
            cropped_image = cv2.convertScaleAbs(cropped_image, alpha=0.85, beta=0.1) #
decrease contrast
            cropped_image = cv2.blur(cropped_image,(2,2)) # apply blur to image
            cropped_image = cv2.bilateralFilter(cropped_image, d=4, sigmaColor=8,
sigmaSpace=12) # Bilateral filter for noise reduction
            cropped_image = cv2.threshold(cropped_image, 10, 255,
cv2.THRESH_BINARY_INV | cv2.THRESH_OTSU)[1] # apply binary treshhold
inverse
            cropped_image = cv2.bitwise_not(cropped_image) # Invert colors
            cropped_image = cropped_image[0:cropped_image.shape[0] - 10,
0:cropped_image.shape[1]] #menghapus tanggal pajak

```



```

else:
    print("Label Tidak Diketahui")
    return None

# Display the resulting cropped image
return cropped_image
def read_text(self, result_image):
    reader = easyocr.Reader(['id'], gpu=True)
    results = reader.readtext(result_image)
    return results

# Function to sort OCR results based on X-axis coordinates
def sort_by_x_axis(self, results):
    return sorted(results, key=lambda x: x[0][0][0])

# Sort OCR results by X-axis coordinates
def filter_results(self, results):
    sorted_results = self.sort_by_x_axis(results)
    plate_number = ""

    for result in sorted_results:
        text = re.sub(r'[^\A-Za-z0-9]', "", result[1]).upper() # Remove special characters

        # Condition to include shorter alpha sequences (<= 2 characters)
        if text.isalpha() and len(text.replace(" ", "")) <= 2:
            filter = ".join([char if char.isalnum() or char == ' ' else " for char in text]) + ' '
            plate_number += filter
        # Conditions for other types of sequences (digits, longer alpha sequences, etc.)
        elif len(text.replace(" ", "")) >= 4:
            filter = ".join([char if char.isalnum() or char == ' ' else " for char in text]) + ' '
            plate_number += filter
        elif text.isdigit() and len(plate_number.replace(" ", "")) <= 4:
            filter = ".join([char if char.isalnum() or char == ' ' else " for char in text]) + ' '
            plate_number += filter
        elif text.isalpha() and len(text.replace(" ", "")) <= 3:
            filter = ".join([char if char.isalnum() or char == ' ' else " for char in text]) + ' '
            plate_number += filter
        else:
            continue
    plate_number = plate_number.upper().strip().replace(" ", "")
    plate_number = re.sub(r"([A-Z])([0-9])", r"\1 \2", plate_number)
    plate_number = re.sub(r"([0-9])([A-Z])", r"\1 \2", plate_number)
    return plate_number
def predict_image(self, image, model_path=None):
    if model_path:
        self.load_model(model_path)

```

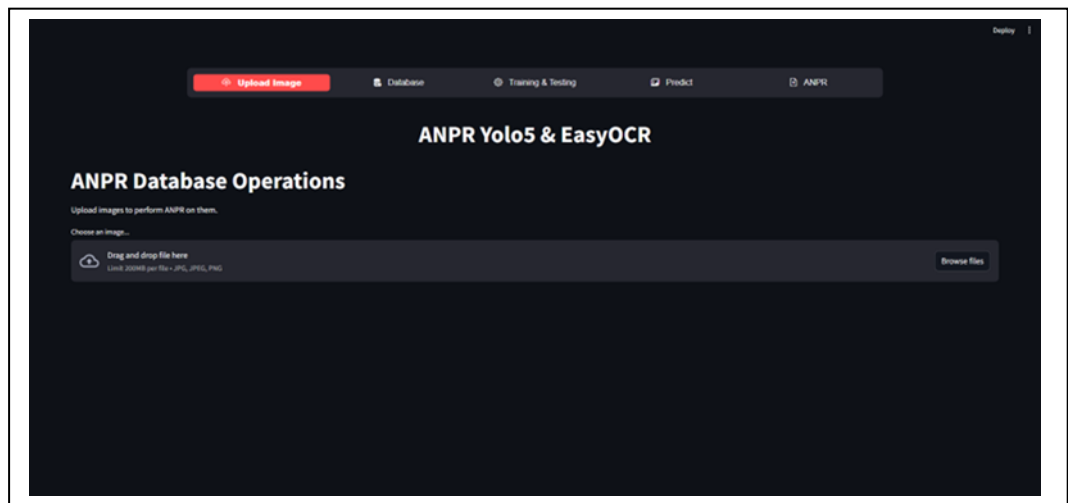
## Bagian 2

### Manual Penggunaan Sistem

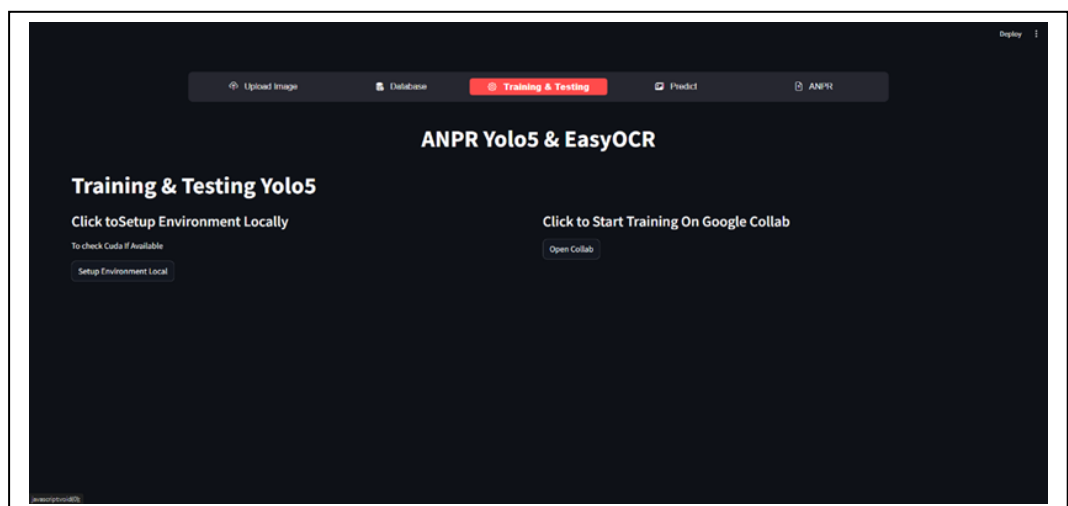
#### 2.1 Step by step Proses Pelatihan YoloV5

Langkah-langkah untuk melakukan pelatihan metode YoloV5 pada sistem yang dibuat adalah sebagai berikut.

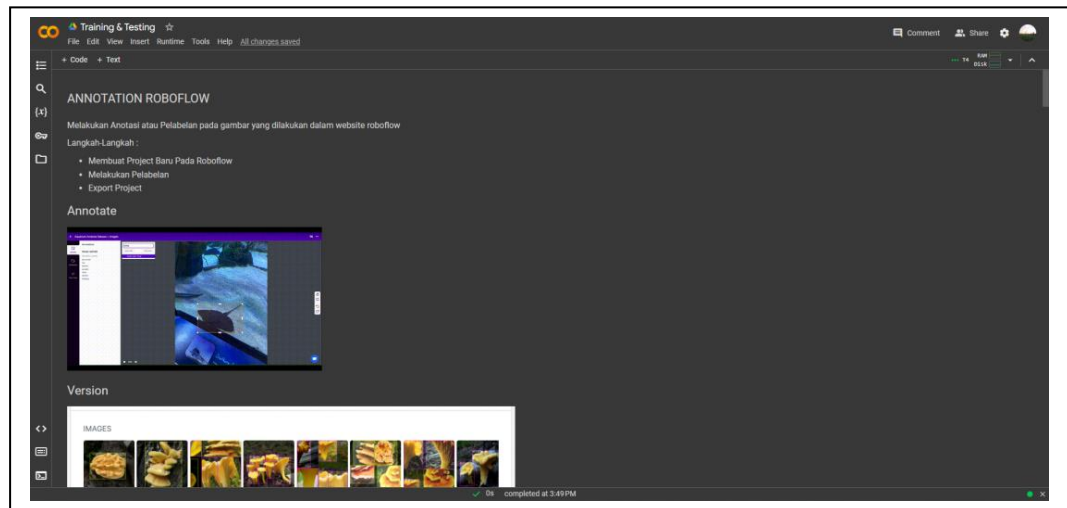
1. Buka sistem hingga halaman beranda muncul.



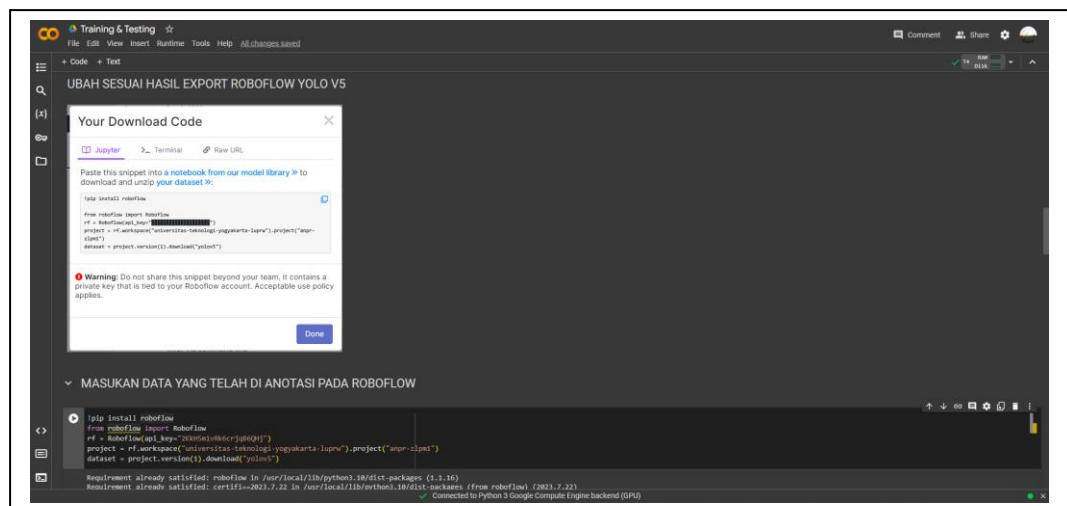
2. Pilih menu “Setup Environment Local” terlebih dahulu untuk melakukan instalasi kebutuhan sistem secara local untuk mempersiapkan proses pengujian, lalu pilih menu “Open Collab”



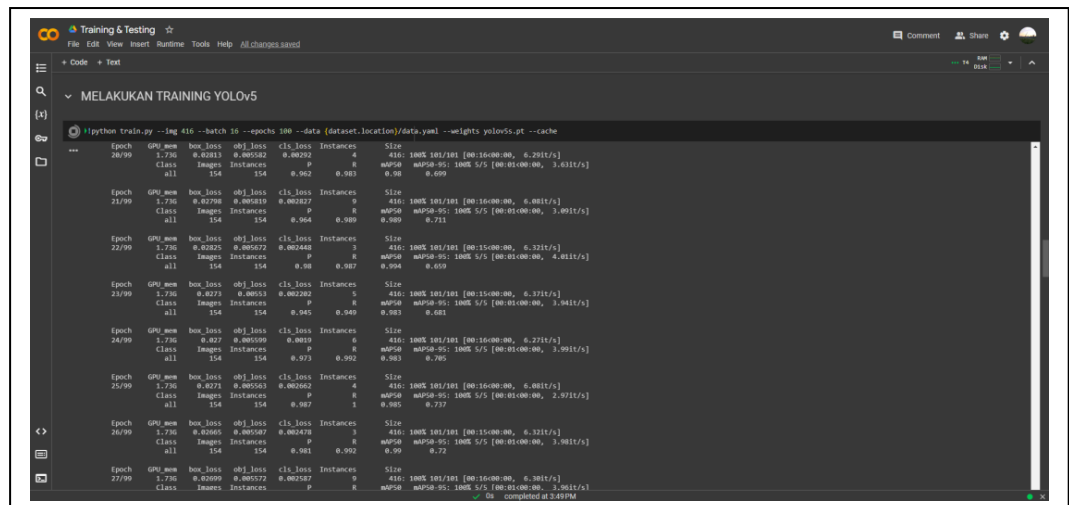
3. Tunggu hingga halaman pelatihan muncul, setelah memilih “Open Collab”, akan muncul halaman Google Collab yang berisi proses pelatihan.



4. Persiapkan data yang telah di anotasi sebelum menggunakan YoloV5, apabila telah melakukan anotasi maka masukan API dataset yang telah dilakukan anotasi tadi, dengan melakukan export data.



5. Setelah berhasil melakukan import dataset yang telah dilakukan anotasi, selanjutnya kita dapat mengubah *hyperparameter* dari metode YoloV5 seperti Epoch dan Batch, lalu jalankan program.

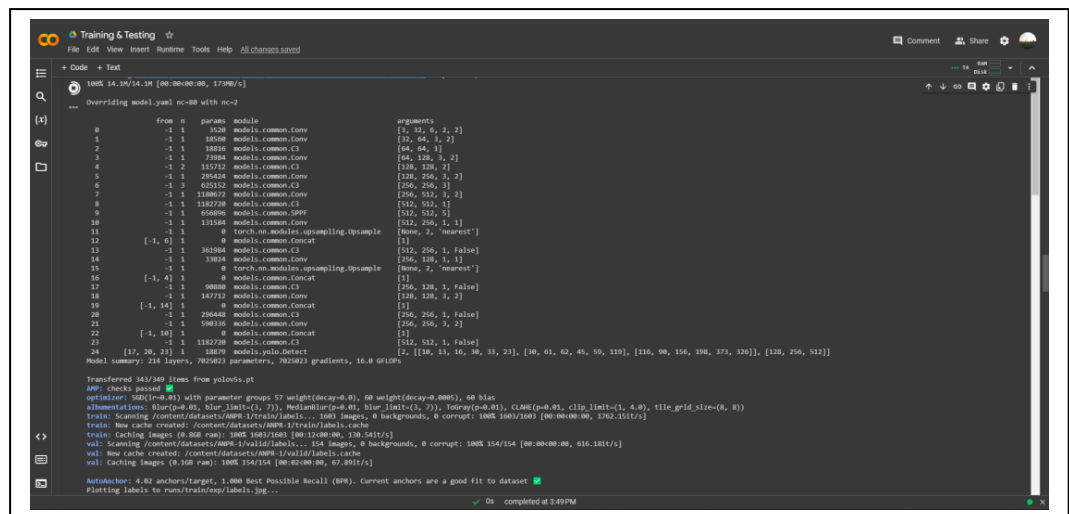


```
!python train.py --img 416 --batch 16 --epochs 100 --data (dataset.location)/data.yaml --weights yolo5s.pt --cache
```

Epoch	GPU_mem	box_loss	obj_loss	cls_loss	Instances	Size
20/99	1.73G	0.42811	0.400582	0.00292	4	416: 100% 101/101 [00:16:00.00, 6.201it/s]
Class	Images	Instances	P	R	mAP50	mAP50-95: 100% 5/5 [00:01:00.00, 3.631it/s]
all	154	154	0.962	0.983	0.98	0.699
Epoch	GPU_mem	box_loss	obj_loss	cls_loss	Instances	Size
21/99	1.73G	0.42798	0.400419	0.002827	9	416: 100% 101/101 [00:16:00.00, 6.401it/s]
Class	Images	Instances	P	R	mAP50	mAP50-95: 100% 5/5 [00:01:00.00, 3.801it/s]
all	154	154	0.964	0.989	0.989	0.711
Epoch	GPU_mem	box_loss	obj_loss	cls_loss	Instances	Size
22/99	1.73G	0.42829	0.400572	0.002448	3	416: 100% 101/101 [00:15:00.00, 6.321it/s]
Class	Images	Instances	P	R	mAP50	mAP50-95: 100% 5/5 [00:01:00.00, 4.011it/s]
all	154	154	0.98	0.987	0.994	0.619
Epoch	GPU_mem	box_loss	obj_loss	cls_loss	Instances	Size
23/99	1.73G	0.4271	0.40051	0.002262	5	416: 100% 101/101 [00:15:00.00, 6.371it/s]
Class	Images	Instances	P	R	mAP50	mAP50-95: 100% 5/5 [00:01:00.00, 3.941it/s]
all	154	154	0.945	0.940	0.983	0.681
Epoch	GPU_mem	box_loss	obj_loss	cls_loss	Instances	Size
24/99	1.73G	0.427	0.400599	0.0019	6	416: 100% 101/101 [00:16:00.00, 6.271it/s]
Class	Images	Instances	P	R	mAP50	mAP50-95: 100% 5/5 [00:01:00.00, 3.991it/s]
all	154	154	0.973	0.992	0.983	0.705
Epoch	GPU_mem	box_loss	obj_loss	cls_loss	Instances	Size
25/99	1.73G	0.4271	0.400563	0.002662	4	416: 100% 101/101 [00:16:00.00, 6.401it/s]
Class	Images	Instances	P	R	mAP50	mAP50-95: 100% 5/5 [00:01:00.00, 2.971it/s]
all	154	154	0.987	1	0.985	0.727
Epoch	GPU_mem	box_loss	obj_loss	cls_loss	Instances	Size
26/99	1.73G	0.42665	0.400587	0.002478	3	416: 100% 101/101 [00:15:00.00, 6.321it/s]
Class	Images	Instances	P	R	mAP50	mAP50-95: 100% 5/5 [00:01:00.00, 3.901it/s]
all	154	154	0.981	0.992	0.99	0.72
Epoch	GPU_mem	box_loss	obj_loss	cls_loss	Instances	Size
27/99	1.73G	0.42699	0.400572	0.002587	9	416: 100% 101/101 [00:16:00.00, 6.301it/s]
Class	Images	Instances	P	R	mAP50	mAP50-95: 100% 5/5 [00:01:00.00, 3.961it/s]

On completed at 5:49 PM

6. Setelah program berjalan, maka sistem akan memperlihatkan proses pelatihan, pada awal proses pelatihan akan memperlihatkan arsitektur YoloV5 kemudian memperlihatkan setiap epoch yang berjalan



```
!python train.py --img 416 --batch 16 --epochs 100 --data (dataset.location)/data.yaml --weights yolo5s.pt --cache
```

```
Model summary: 214 layers, 761063 parameters, 761063 gradients, 16.8 GFLOPs
```

```
Transferred 242/249 items from yolov5s.pt
```

```
AMP: checks passed
```

```
optimizer: SGD(lr=0.01) with parameter groups 37 weight(decay=0.0), 98 weight(decay=0.0001), 48 bias
```

```
albuotetimes: Blur(p=0.01, blur_limit=(3, 7)), MedianBlur(p=0.01, blur_limit=(3, 7)), TopDown(p=0.01, CLAW(p=0.01, clip_limit=(1, 4.0), tile_grid_size=(8, 8))
```

```
train: Scanning dataset/(dataset.location)/train/labels.cache... 100% images, 0 backgrounds, 0 corrupt: 100% 100/100 [00:00:00.00, 1762.151it/s]
```

```
train: Now cache created: /content/dataset/(dataset.location)/train/labels.cache
```

```
train: Caching images (0.808 ram): 100% 100/100 [00:12:00.00, 130.541it/s]
```

```
val: Scanning /content/dataset/(dataset.location)/valid/labels.cache... 114 images, 0 backgrounds, 0 corrupt: 100% 114/114 [00:00:00.00, 616.181it/s]
```

```
val: Now cache created: /content/dataset/(dataset.location)/valid/labels.cache
```

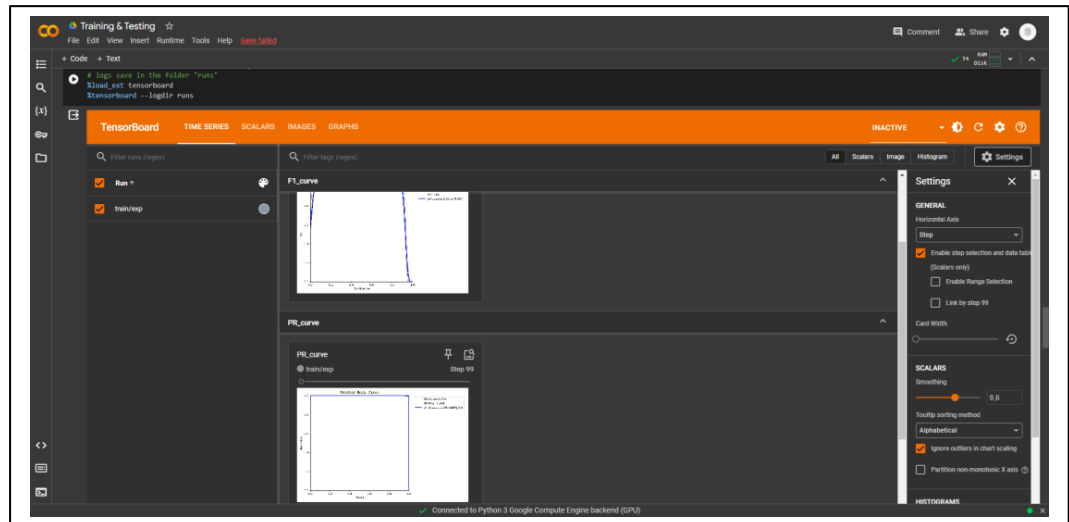
```
val: Caching images (0.108 ram): 100% 114/114 [00:02:00.00, 67.491it/s]
```

Autoscheduler: 4.82 anchors/target, 1.000 best Possible Recall (BPR). Current anchors are a good fit to dataset

Plotting labels to run/train/exp/labels.jpg...

On completed at 5:49 PM

7. Proses pelatihan selesai ditandai dengan tampilnya grafik hasil pelatihan yang ada pada tensorboard, tensorboard ini akan menampilkan berbagai metrik sebagai parameter penilaian kita tentang seberapa bagus model kita.



8. Setelah menjalankan tensorboard, kita akan menjalankan proses testing pada setiap gambar test, ketika menjalankan proses testing akan menampilkan seperti berikut.

```
Training & Testing
File Edit View Insert Runtime Tools Help Save failed
+ Code + Test
+ Logs save in the folder "runs"
+ Detect, test, tensorboard
+ Monitorboard --logdir runs

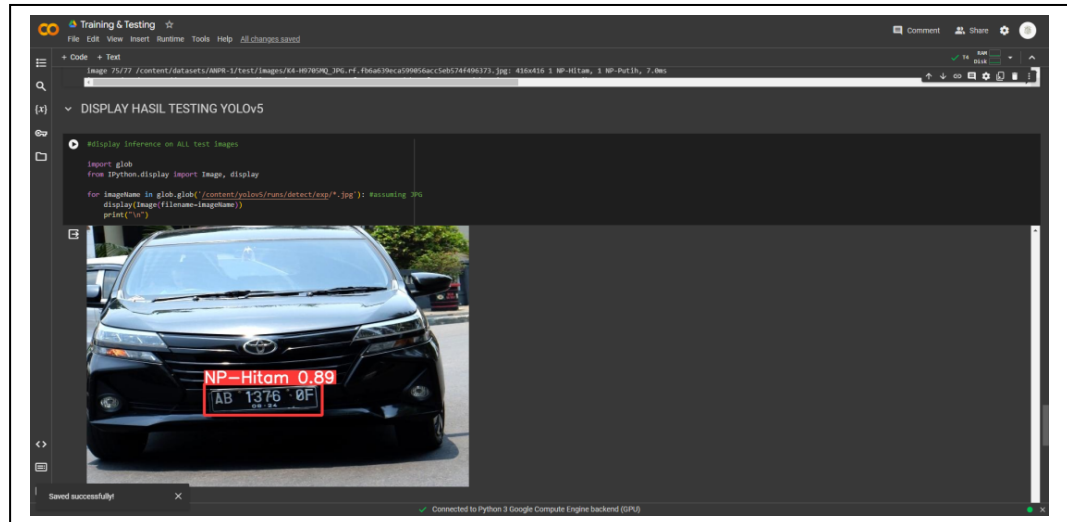
TensorBoard
TIME SERIES SCALARS IMAGES GRAPHS
INACTIVE
Filter tags (tags)
Run +
Train/Log

F1_score
PR_curve
PR_curve
Step 99

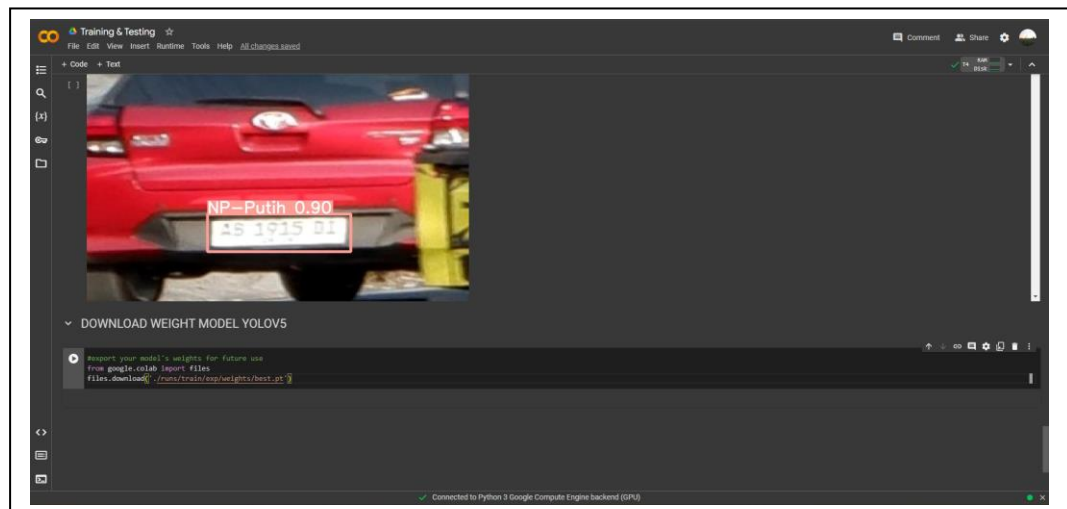
detect: weights=train/weights/best.pt, source=content/datasets/ANPR-1/test/images, data=data/coco128.yaml, imgsz=[416, 416], conf_thres=0.1, iou_thres=0.45, max_det=1000, device=, view_img=False, save_txt=False, save
YOLOv5 v7.0-218-gf06c72c Python-3.10.12 torch-2.1.0+cu121 CUDA-0 (Tesla T4, 1510MiB)

Fusing layers...
Model Summary: 152 layers, 785559 parameters, 0 gradients, 15.8 GLOPs
Image 1/77 /content/datasets/ANPR-1/test/images/K1-AA200418.jpg, ref. d5c3ba3a0e1af91efeb0e400400b0dab.jpg: 416x416 1 WP-Hitam, 7.1ms
Image 2/77 /content/datasets/ANPR-1/test/images/K1-AA200418.jpg, ref. d5c3ba3a0e1af91efeb0e400400b0dab.jpg: 416x416 1 WP-Hitam, 7.1ms
Image 3/77 /content/datasets/ANPR-1/test/images/K1-AA200418.jpg, ref. d5c3ba3a0e1af91efeb0e400400b0dab.jpg: 416x416 1 WP-Hitam, 7.1ms
Image 4/77 /content/datasets/ANPR-1/test/images/K1-AA200418.jpg, ref. d5c3ba3a0e1af91efeb0e400400b0dab.jpg: 416x416 1 WP-Hitam, 7.1ms
Image 5/77 /content/datasets/ANPR-1/test/images/K1-AA200418.jpg, ref. d5c3ba3a0e1af91efeb0e400400b0dab.jpg: 416x416 1 WP-Hitam, 7.1ms
Image 6/77 /content/datasets/ANPR-1/test/images/K1-AA200418.jpg, ref. d5c3ba3a0e1af91efeb0e400400b0dab.jpg: 416x416 1 WP-Hitam, 7.1ms
Image 7/77 /content/datasets/ANPR-1/test/images/K1-AA200418.jpg, ref. d5c3ba3a0e1af91efeb0e400400b0dab.jpg: 416x416 1 WP-Hitam, 7.1ms
Image 8/77 /content/datasets/ANPR-1/test/images/K1-AA200418.jpg, ref. d5c3ba3a0e1af91efeb0e400400b0dab.jpg: 416x416 1 WP-Hitam, 7.1ms
Image 9/77 /content/datasets/ANPR-1/test/images/K1-AA200418.jpg, ref. d5c3ba3a0e1af91efeb0e400400b0dab.jpg: 416x416 1 WP-Hitam, 7.1ms
Image 10/77 /content/datasets/ANPR-1/test/images/K1-AA200418.jpg, ref. d5c3ba3a0e1af91efeb0e400400b0dab.jpg: 416x416 1 WP-Hitam, 7.1ms
Image 11/77 /content/datasets/ANPR-1/test/images/K1-AA200418.jpg, ref. d5c3ba3a0e1af91efeb0e400400b0dab.jpg: 416x416 1 WP-Hitam, 7.1ms
Image 12/77 /content/datasets/ANPR-1/test/images/K1-AA200418.jpg, ref. d5c3ba3a0e1af91efeb0e400400b0dab.jpg: 416x416 1 WP-Hitam, 7.1ms
Image 13/77 /content/datasets/ANPR-1/test/images/K1-AA200418.jpg, ref. d5c3ba3a0e1af91efeb0e400400b0dab.jpg: 416x416 1 WP-Hitam, 7.1ms
Image 14/77 /content/datasets/ANPR-1/test/images/K1-AA200418.jpg, ref. d5c3ba3a0e1af91efeb0e400400b0dab.jpg: 416x416 1 WP-Hitam, 7.1ms
Image 15/77 /content/datasets/ANPR-1/test/images/K1-AA200418.jpg, ref. d5c3ba3a0e1af91efeb0e400400b0dab.jpg: 416x416 1 WP-Hitam, 7.1ms
Image 16/77 /content/datasets/ANPR-1/test/images/K1-AA200418.jpg, ref. d5c3ba3a0e1af91efeb0e400400b0dab.jpg: 416x416 1 WP-Hitam, 7.1ms
Image 17/77 /content/datasets/ANPR-1/test/images/K1-AA200418.jpg, ref. d5c3ba3a0e1af91efeb0e400400b0dab.jpg: 416x416 1 WP-Hitam, 7.1ms
Image 18/77 /content/datasets/ANPR-1/test/images/K1-AA200418.jpg, ref. d5c3ba3a0e1af91efeb0e400400b0dab.jpg: 416x416 1 WP-Hitam, 7.1ms
Image 19/77 /content/datasets/ANPR-1/test/images/K1-AA200418.jpg, ref. d5c3ba3a0e1af91efeb0e400400b0dab.jpg: 416x416 1 WP-Hitam, 7.1ms
Image 20/77 /content/datasets/ANPR-1/test/images/K1-AA200418.jpg, ref. d5c3ba3a0e1af91efeb0e400400b0dab.jpg: 416x416 1 WP-Hitam, 7.1ms
Image 21/77 /content/datasets/ANPR-1/test/images/K1-AA200418.jpg, ref. d5c3ba3a0e1af91efeb0e400400b0dab.jpg: 416x416 1 WP-Hitam, 7.1ms
Image 22/77 /content/datasets/ANPR-1/test/images/K1-AA200418.jpg, ref. d5c3ba3a0e1af91efeb0e400400b0dab.jpg: 416x416 1 WP-Hitam, 7.1ms
Image 23/77 /content/datasets/ANPR-1/test/images/K1-AA200418.jpg, ref. d5c3ba3a0e1af91efeb0e400400b0dab.jpg: 416x416 1 WP-Hitam, 7.1ms
Image 24/77 /content/datasets/ANPR-1/test/images/K1-AA200418.jpg, ref. d5c3ba3a0e1af91efeb0e400400b0dab.jpg: 416x416 1 WP-Hitam, 7.1ms
Image 25/77 /content/datasets/ANPR-1/test/images/K1-AA200418.jpg, ref. d5c3ba3a0e1af91efeb0e400400b0dab.jpg: 416x416 1 WP-Hitam, 7.1ms
Image 26/77 /content/datasets/ANPR-1/test/images/K1-AA200418.jpg, ref. d5c3ba3a0e1af91efeb0e400400b0dab.jpg: 416x416 1 WP-Hitam, 7.1ms
Image 27/77 /content/datasets/ANPR-1/test/images/K1-AA200418.jpg, ref. d5c3ba3a0e1af91efeb0e400400b0dab.jpg: 416x416 1 WP-Hitam, 7.1ms
Image 28/77 /content/datasets/ANPR-1/test/images/K1-AA200418.jpg, ref. d5c3ba3a0e1af91efeb0e400400b0dab.jpg: 416x416 1 WP-Hitam, 7.1ms
Image 29/77 /content/datasets/ANPR-1/test/images/K1-AA200418.jpg, ref. d5c3ba3a0e1af91efeb0e400400b0dab.jpg: 416x416 1 WP-Hitam, 7.1ms
Image 30/77 /content/datasets/ANPR-1/test/images/K1-AA200418.jpg, ref. d5c3ba3a0e1af91efeb0e400400b0dab.jpg: 416x416 1 WP-Hitam, 7.1ms
Image 31/77 /content/datasets/ANPR-1/test/images/K1-AA200418.jpg, ref. d5c3ba3a0e1af91efeb0e400400b0dab.jpg: 416x416 1 WP-Hitam, 7.1ms
Image 32/77 /content/datasets/ANPR-1/test/images/K1-AA200418.jpg, ref. d5c3ba3a0e1af91efeb0e400400b0dab.jpg: 416x416 1 WP-Hitam, 7.1ms
Image 33/77 /content/datasets/ANPR-1/test/images/K1-AA200418.jpg, ref. d5c3ba3a0e1af91efeb0e400400b0dab.jpg: 416x416 1 WP-Hitam, 7.1ms
Image 34/77 /content/datasets/ANPR-1/test/images/K1-AA200418.jpg, ref. d5c3ba3a0e1af91efeb0e400400b0dab.jpg: 416x416 1 WP-Hitam, 7.1ms
Image 35/77 /content/datasets/ANPR-1/test/images/K1-AA200418.jpg, ref. d5c3ba3a0e1af91efeb0e400400b0dab.jpg: 416x416 1 WP-Hitam, 7.1ms
Image 36/77 /content/datasets/ANPR-1/test/images/K1-AA200418.jpg, ref. d5c3ba3a0e1af91efeb0e400400b0dab.jpg: 416x416 1 WP-Hitam, 7.1ms
Image 37/77 /content/datasets/ANPR-1/test/images/K1-AA200418.jpg, ref. d5c3ba3a0e1af91efeb0e400400b0dab.jpg: 416x416 1 WP-Hitam, 7.1ms
Image 38/77 /content/datasets/ANPR-1/test/images/K1-AA200418.jpg, ref. d5c3ba3a0e1af91efeb0e400400b0dab.jpg: 416x416 1 WP-Hitam, 7.1ms
Image 39/77 /content/datasets/ANPR-1/test/images/K1-AA200418.jpg, ref. d5c3ba3a0e1af91efeb0e400400b0dab.jpg: 416x416 1 WP-Hitam, 7.1ms
Image 40/77 /content/datasets/ANPR-1/test/images/K1-AA200418.jpg, ref. d5c3ba3a0e1af91efeb0e400400b0dab.jpg: 416x416 1 WP-Hitam, 7.1ms
Image 41/77 /content/datasets/ANPR-1/test/images/K1-AA200418.jpg, ref. d5c3ba3a0e1af91efeb0e400400b0dab.jpg: 416x416 1 WP-Hitam, 7.1ms
Image 42/77 /content/datasets/ANPR-1/test/images/K1-AA200418.jpg, ref. d5c3ba3a0e1af91efeb0e400400b0dab.jpg: 416x416 1 WP-Hitam, 7.1ms
Image 43/77 /content/datasets/ANPR-1/test/images/K1-AA200418.jpg, ref. d5c3ba3a0e1af91efeb0e400400b0dab.jpg: 416x416 1 WP-Hitam, 7.1ms
Image 44/77 /content/datasets/ANPR-1/test/images/K1-AA200418.jpg, ref. d5c3ba3a0e1af91efeb0e400400b0dab.jpg: 416x416 1 WP-Hitam, 7.1ms
Image 45/77 /content/datasets/ANPR-1/test/images/K1-AA200418.jpg, ref. d5c3ba3a0e1af91efeb0e400400b0dab.jpg: 416x416 1 WP-Hitam, 7.1ms
Image 46/77 /content/datasets/ANPR-1/test/images/K1-AA200418.jpg, ref. d5c3ba3a0e1af91efeb0e400400b0dab.jpg: 416x416 1 WP-Hitam, 7.1ms
Image 47/77 /content/datasets/ANPR-1/test/images/K1-AA200418.jpg, ref. d5c3ba3a0e1af91efeb0e400400b0dab.jpg: 416x416 1 WP-Hitam, 7.1ms
Image 48/77 /content/datasets/ANPR-1/test/images/K1-AA200418.jpg, ref. d5c3ba3a0e1af91efeb0e400400b0dab.jpg: 416x416 1 WP-Hitam, 7.1ms
Image 49/77 /content/datasets/ANPR-1/test/images/K1-AA200418.jpg, ref. d5c3ba3a0e1af91efeb0e400400b0dab.jpg: 416x416 1 WP-Hitam, 7.1ms
Image 50/77 /content/datasets/ANPR-1/test/images/K1-AA200418.jpg, ref. d5c3ba3a0e1af91efeb0e400400b0dab.jpg: 416x416 1 WP-Hitam, 7.1ms
Image 51/77 /content/datasets/ANPR-1/test/images/K1-AA200418.jpg, ref. d5c3ba3a0e1af91efeb0e400400b0dab.jpg: 416x416 1 WP-Hitam, 7.1ms
Image 52/77 /content/datasets/ANPR-1/test/images/K1-AA200418.jpg, ref. d5c3ba3a0e1af91efeb0e400400b0dab.jpg: 416x416 1 WP-Hitam, 7.1ms
Image 53/77 /content/datasets/ANPR-1/test/images/K1-AA200418.jpg, ref. d5c3ba3a0e1af91efeb0e400400b0dab.jpg: 416x416 1 WP-Hitam, 7.1ms
Image 54/77 /content/datasets/ANPR-1/test/images/K1-AA200418.jpg, ref. d5c3ba3a0e1af91efeb0e400400b0dab.jpg: 416x416 1 WP-Hitam, 7.1ms
Image 55/77 /content/datasets/ANPR-1/test/images/K1-AA200418.jpg, ref. d5c3ba3a0e1af91efeb0e400400b0dab.jpg: 416x416 1 WP-Hitam, 7.1ms
Image 56/77 /content/datasets/ANPR-1/test/images/K1-AA200418.jpg, ref. d5c3ba3a0e1af91efeb0e400400b0dab.jpg: 416x416 1 WP-Hitam, 7.1ms
Image 57/77 /content/datasets/ANPR-1/test/images/K1-AA200418.jpg, ref. d5c3ba3a0e1af91efeb0e400400b0dab.jpg: 416x416 1 WP-Hitam, 7.1ms
Image 58/77 /content/datasets/ANPR-1/test/images/K1-AA200418.jpg, ref. d5c3ba3a0e1af91efeb0e400400b0dab.jpg: 416x416 1 WP-Hitam, 7.1ms
Image 59/77 /content/datasets/ANPR-1/test/images/K1-AA200418.jpg, ref. d5c3ba3a0e1af91efeb0e400400b0dab.jpg: 416x416 1 WP-Hitam, 7.1ms
Image 60/77 /content/datasets/ANPR-1/test/images/K1-AA200418.jpg, ref. d5c3ba3a0e1af91efeb0e400400b0dab.jpg: 416x416 1 WP-Hitam, 7.1ms
Image 61/77 /content/datasets/ANPR-1/test/images/K1-AA200418.jpg, ref. d5c3ba3a0e1af91efeb0e400400b0dab.jpg: 416x416 1 WP-Hitam, 7.1ms
Image 62/77 /content/datasets/ANPR-1/test/images/K1-AA200418.jpg, ref. d5c3ba3a0e1af91efeb0e400400b0dab.jpg: 416x416 1 WP-Hitam, 7.1ms
Image 63/77 /content/datasets/ANPR-1/test/images/K1-AA200418.jpg, ref. d5c3ba3a0e1af91efeb0e400400b0dab.jpg: 416x416 1 WP-Hitam, 7.1ms
Image 64/77 /content/datasets/ANPR-1/test/images/K1-AA200418.jpg, ref. d5c3ba3a0e1af91efeb0e400400b0dab.jpg: 416x416 1 WP-Hitam, 7.1ms
Image 65/77 /content/datasets/ANPR-1/test/images/K1-AA200418.jpg, ref. d5c3ba3a0e1af91efeb0e400400b0dab.jpg: 416x416 1 WP-Hitam, 7.1ms
Image 66/77 /content/datasets/ANPR-1/test/images/K1-AA200418.jpg, ref. d5c3ba3a0e1af91efeb0e400400b0dab.jpg: 416x416 1 WP-Hitam, 7.1ms
Image 67/77 /content/datasets/ANPR-1/test/images/K1-AA200418.jpg, ref. d5c3ba3a0e1af91efeb0e400400b0dab.jpg: 416x416 1 WP-Hitam, 7.1ms
Image 68/77 /content/datasets/ANPR-1/test/images/K1-AA200418.jpg, ref. d5c3ba3a0e1af91efeb0e400400b0dab.jpg: 416x416 1 WP-Hitam, 7.1ms
Image 69/77 /content/datasets/ANPR-1/test/images/K1-AA200418.jpg, ref. d5c3ba3a0e1af91efeb0e400400b0dab.jpg: 416x416 1 WP-Hitam, 7.1ms
Image 70/77 /content/datasets/ANPR-1/test/images/K1-AA200418.jpg, ref. d5c3ba3a0e1af91efeb0e400400b0dab.jpg: 416x416 1 WP-Hitam, 7.1ms
Image 71/77 /content/datasets/ANPR-1/test/images/K1-AA200418.jpg, ref. d5c3ba3a0e1af91efeb0e400400b0dab.jpg: 416x416 1 WP-Hitam, 7.1ms
Image 72/77 /content/datasets/ANPR-1/test/images/K1-AA200418.jpg, ref. d5c3ba3a0e1af91efeb0e400400b0dab.jpg: 416x416 1 WP-Hitam, 7.1ms
Image 73/77 /content/datasets/ANPR-1/test/images/K1-AA200418.jpg, ref. d5c3ba3a0e1af91efeb0e400400b0dab.jpg: 416x416 1 WP-Hitam, 7.1ms
Image 74/77 /content/datasets/ANPR-1/test/images/K1-AA200418.jpg, ref. d5c3ba3a0e1af91efeb0e400400b0dab.jpg: 416x416 1 WP-Hitam, 7.1ms
Image 75/77 /content/datasets/ANPR-1/test/images/K1-AA200418.jpg, ref. d5c3ba3a0e1af91efeb0e400400b0dab.jpg: 416x416 1 WP-Hitam, 7.1ms
Image 76/77 /content/datasets/ANPR-1/test/images/K1-AA200418.jpg, ref. d5c3ba3a0e1af91efeb0e400400b0dab.jpg: 416x416 1 WP-Hitam, 7.1ms
Image 77/77 /content/datasets/ANPR-1/test/images/K1-AA200418.jpg, ref. d5c3ba3a0e1af91efeb0e400400b0dab.jpg: 416x416 1 WP-Hitam, 7.1ms
Connected to Python 3 Google Compute Engine backend (GPU)
```

9. Pengguna dapat melihat contoh hasil deteksi plat nomor, hasil deteksi ini memperlihatkan kualitas model yang dihasilkan



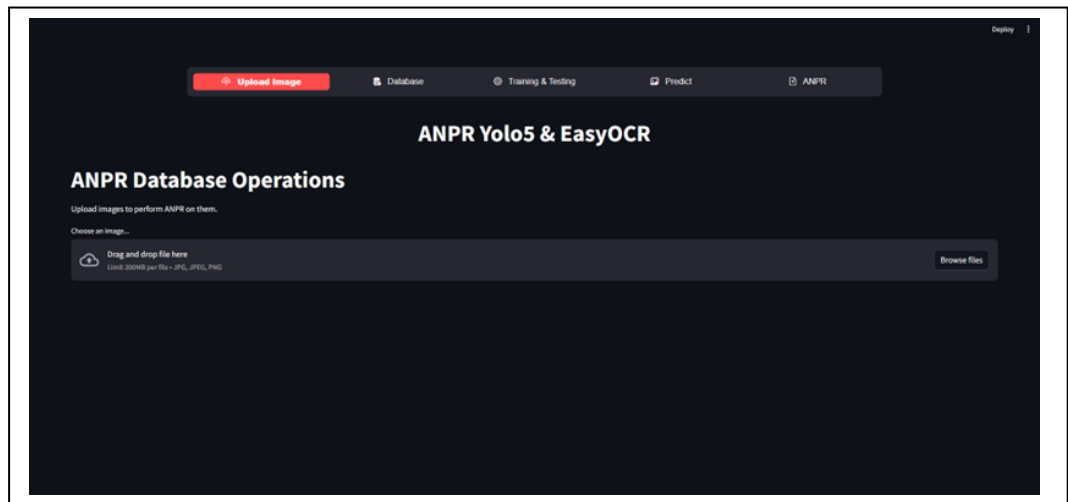
10. Setelah melihat model yang dihasilkan, pengguna dapat mengunduh bobot yang dihasilkan oleh YoloV5 untuk dimasukkan ke dalam sistem.



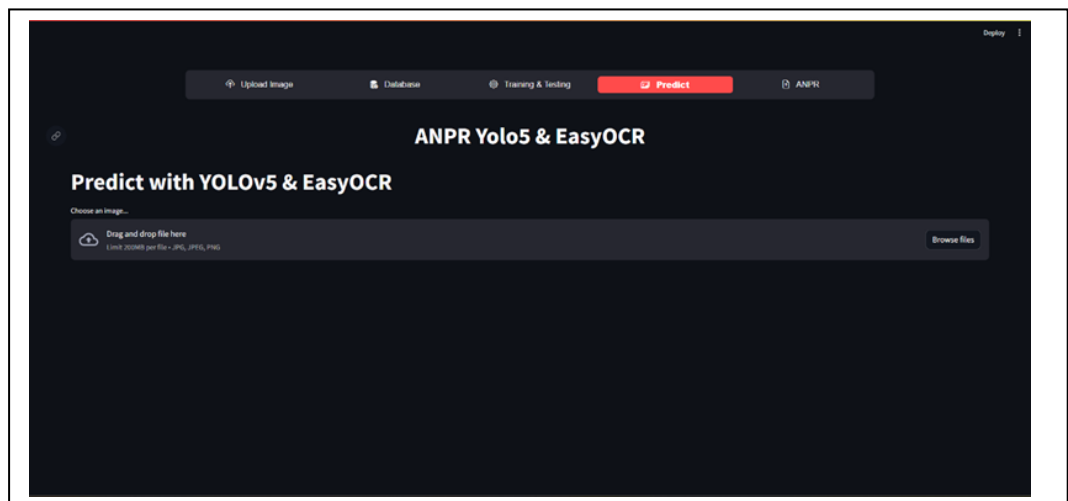
## 2.2 Step by step Proses Pengujian Deteksi Plat Nomor (Satu File)

Langkah-langkah untuk melakukan pengujian YoloV5 dan EasyOCR pada sistem yang dibuat adalah sebagai berikut.

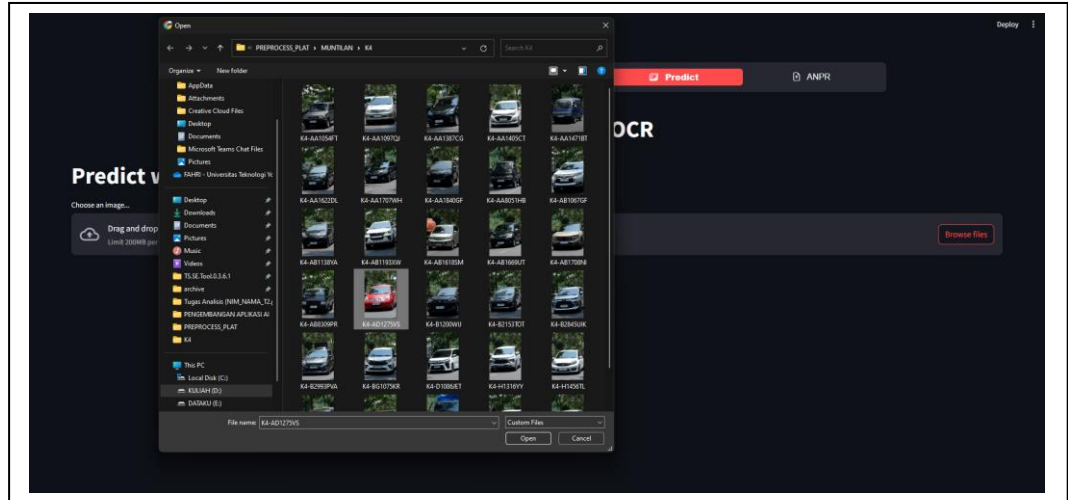
1. Buka sistem hingga halaman beranda muncul. Jangan lupa ubah file Best\_V5.pt dengan hasil training kalian yang berada pada lokasi file sistem ini.



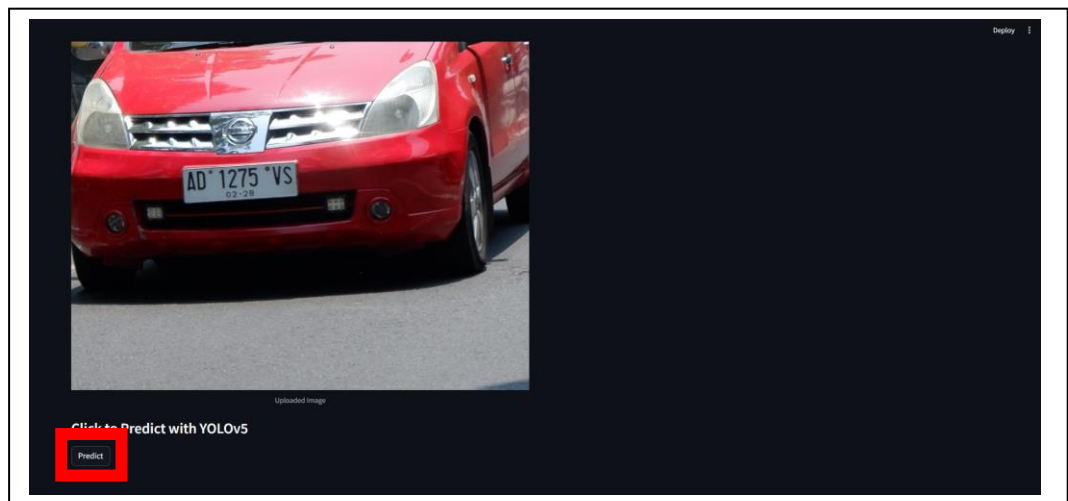
2. Pilih menu “Predict” pada bagian atas (menu nomor 4), dan tunggu hingga halaman predict muncul.



3. Tekan tombol “Browse Files”, dan pilih file gambar yang akan di upload.



4. Setelah gambar terpilih sistem akan menampilkan gambar tersebut, kemudian tekan tombol “Predict” untuk melakukan deteksi plat nomor dan prediksi karakter

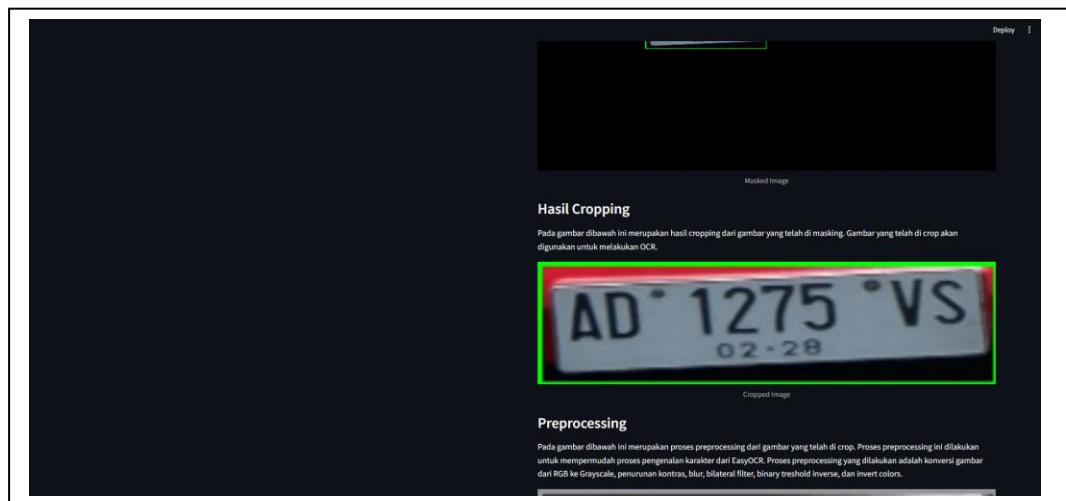




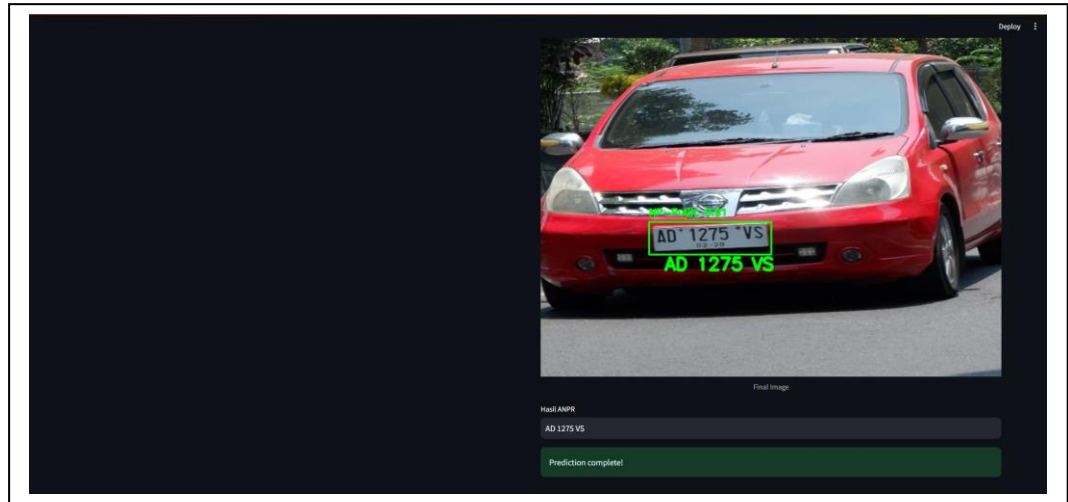
- Setelah tombol “Predict” dipilih maka akan muncul hasil prediksi dari deteksi plat nomor menggunakan YoloV5



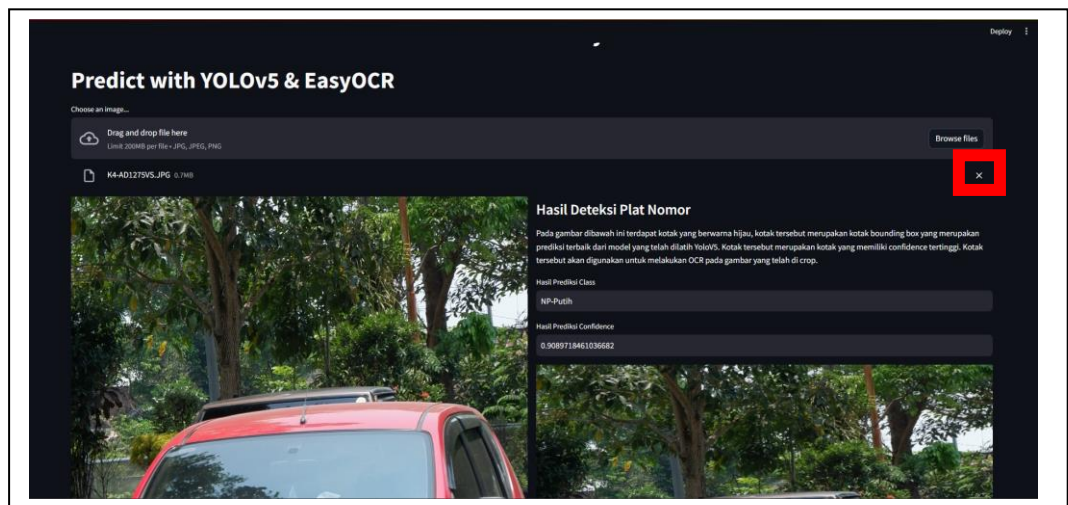
- Kemudian sistem memperlihatkan setiap proses yang dilakukan kemudian akan memberikan hasil pengenalan karakter menggunakan EasyOCR.



7. Setelah berhasil melakukan pengenalan karakter sistem akan memberikan tampilan prediksi plat nomor dan pengenalan karakter, kemudian menampilkan hasil pengenalan karakter.



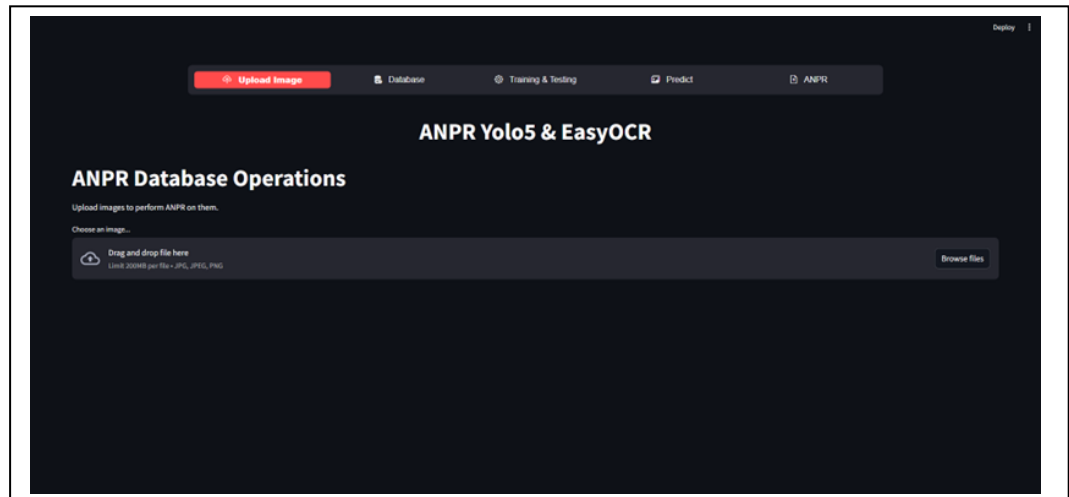
8. Pengguna dapat mengembalikan halaman pengujian ini kembali bersih seperti pada langkah ke-1 dengan meng-klik tombol "X" (kotak merah).



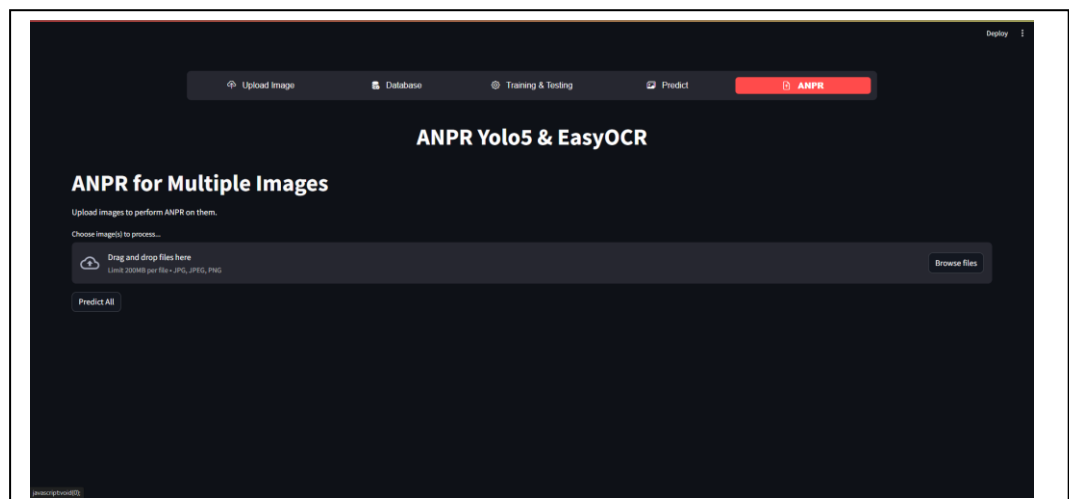
### 2.3 Step by step Proses Pengujian Deteksi Plat Nomor (*Multiple File*)

Langkah-langkah untuk melakukan pengujian jaringan pada sistem yang dibuat adalah sebagai berikut.

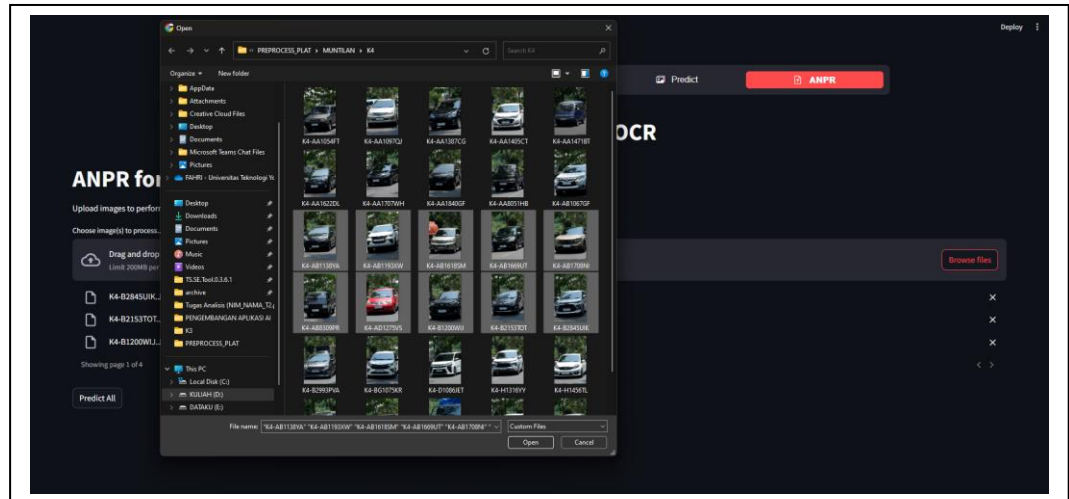
1. Buka sistem hingga halaman beranda muncul. Jangan lupa ubah file Best\_V5.pt dengan hasil training kalian.



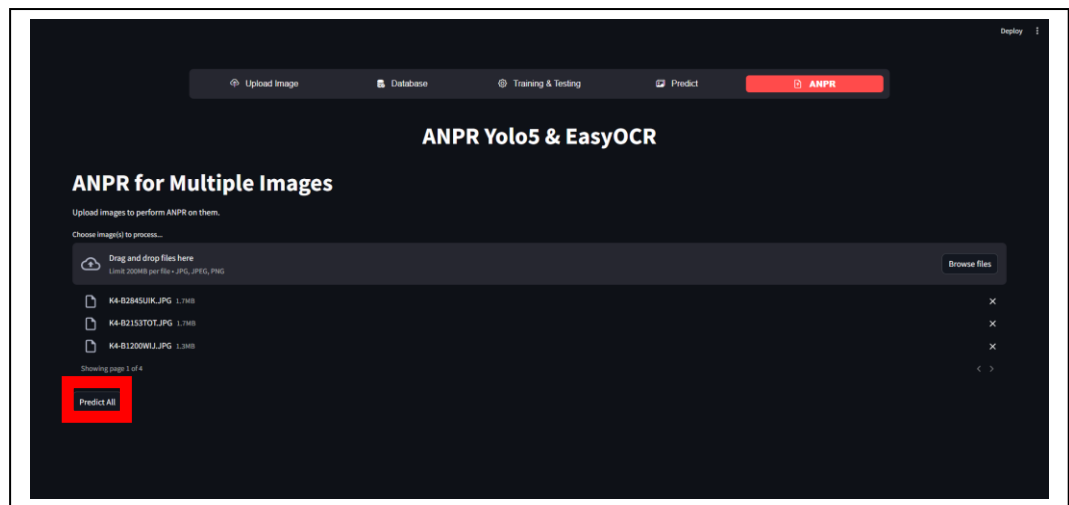
2. Pilih menu “ANPR” pada bagian atas (menu nomor 5), dan tunggu hingga halaman predict muncul.



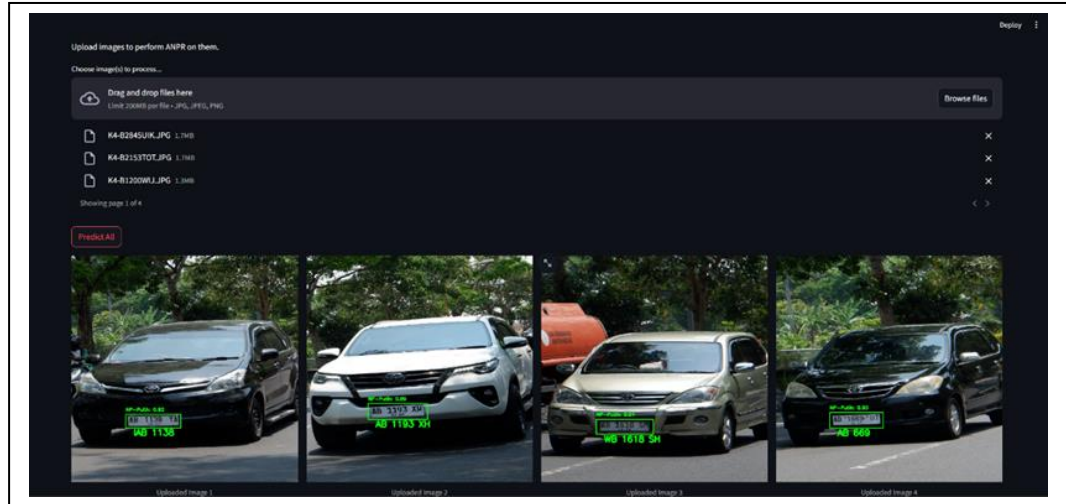
3. Tekan tombol “Browse Filse”, dan pilih beberapa file yang akan dilakukan deteksi plat dan pengenalan karakter.



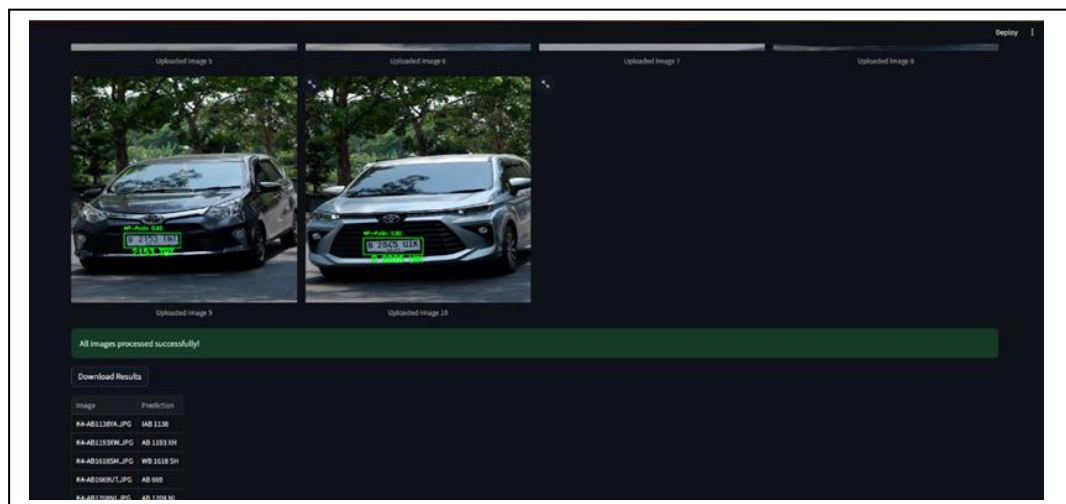
4. Tekan tombol “Predict All” (kotak merah) untuk melakukan prediksi plat nomor dan pengenalan karakter pada setiap gambar yang telah di upload.



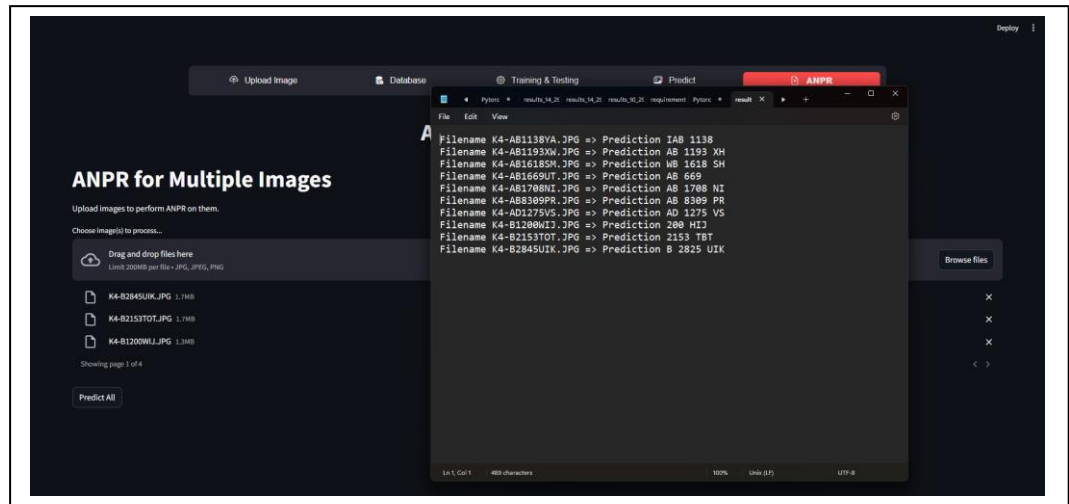
5. Jika tombol “Predict All” sudah dipilih maka sistem akan melakukan prediksi plat nomor dan pengenalan karakter pada setiap gambar.



6. Setelah sistem berhasil melakukan prediksi plat nomor dan pengenalan karakter, maka akan muncul tabel hasil pengenalan karakter, dan terdapat tombol “Download Results”.



7. Setelah proses pengujian selesai, maka hasil pengujian akan tampil di bagian bawah dalam bentuk table, namun kita dapat mengunduh hasil pengujian tersebut dalam bentuk file txt.



## 2.4 Step by step Proses Instalasi Program

Langkah-langkah untuk melakukan pengujian jaringan pada sistem yang dibuat adalah sebagai berikut.

1. Clone / unduh file program pada laman Github berikut:

<https://github.com/FahriPutra00/ANPR-Yolo5-EasyOCR-Streamlit>

2. Lakukan Instalasi Conda dan Jalankan CommandPromt pada PyTorch\_Cuda.Txt

```
conda create -n ANPR-Py python=3.10
conda activate ANPR-Py
conda install -c conda-forge cudatoolkit=11.8 cudnn=8.8.0
pip3 install torch torchvision torchaudio --index-url
https://download.pytorch.org/whl/cu118
pip3 install streamlit
pip3 install streamlit_option_menu
pip3 install easyocr
```

3. Import image\_dataset.sql ke DBMS seperti MySQL dan PostGreSQL
4. Untuk menjalankan program menggunakan virtual environment yang telah dibuat tadi

```
streamlit run main.py
```