```
# STEP 1: Install Required Packages
!pip install --quiet gdown easyocr opencv-python-headless
```

```
WARNING: Retrying (Retry(total=4, connect=None, read=None, redirect=None, status=None)) after connection broken by 'Protocol
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 2.9/2.9 MB 26.6 MB/s eta 0:00:00
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 363.4/363.4 MB 3.6 MB/s eta 0:00:00
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 13.8/13.8 MB 66.0 MB/s eta 0:00:00
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 24.6/24.6 MB 60.2 MB/s eta 0:00:00
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 883.7/883.7 kB 40.6 MB/s eta 0:00:00
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 664.8/664.8 MB 876.2 kB/s eta 0:00:00
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 211.5/211.5 MB 6.0 MB/s eta 0:00:00
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 56.3/56.3 MB 15.7 MB/s eta 0:00:00
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 127.9/127.9 MB 7.4 MB/s eta 0:00:00
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 207.5/207.5 MB 6.6 MB/s eta 0:00:00
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 188.7/188.7 MB 6.2 MB/s eta 0:00:00
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 21.1/21.1 MB 87.4 MB/s eta 0:00:00
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 422.8/422.8 kB 26.3 MB/s eta 0:00:00
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 969.6/969.6 kB 40.1 MB/s eta 0:00:00
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 292.9/292.9 kB 22.2 MB/s eta 0:00:00
```

```
# STEP 2: Import Required Libraries
import os
import gdown
import zipfile
import pandas as pd
from matplotlib import pyplot as plt
from google.colab import files
from IPython.display import display
```

```
# STEP 3: Download and Extract the Dataset
file_id = "15FthdQ1TVjB29_4GWodngor2fDKwgsJL"
zip_path = "plates_dataset.zip"
dataset_dir = "registered_plates"

gdown.download(f"https://drive.google.com/uc?id={file_id}", zip_path, quiet=False)

os.makedirs(dataset_dir, exist_ok=True)
with zipfile.ZipFile(zip_path, 'r') as zip_ref:
    zip_ref.extractall(dataset_dir)

print("✅ Dataset successfully downloaded and unzipped.")
```

```
Downloading...
From (original): https://drive.google.com/uc?id=15FthdQ1TVjB29_4GWodngor2fDKwgsJL
From (redirected): https://drive.google.com/uc?id=15FthdQ1TVjB29_4GWodngor2fDKwgsJL&confirm=t&uuid=02ae99b5-741b-41d3-a46a-
To: /content/plates_dataset.zip
100%|██████████| 213M/213M [00:04<00:00, 45.3MB/s]
✅ Dataset successfully downloaded and unzipped.
```

```
import easyocr

reader = easyocr.Reader(['en'])

def extract_plate_number(img_path):
    print(f"📷 Reading with EasyOCR: {img_path}")

    results = reader.readtext(img_path, detail=0, paragraph=False)

    # Combine and clean result
    if results:
        plate_text = ' '.join(results).upper()
        print(f"📝 EasyOCR Output: {plate_text}")
        return plate_text
    else:
        print("⚠️ EasyOCR failed to detect any text.")
        return "NOT_DETECTED"
```

```
WARNING:easyocr.easyocr:Neither CUDA nor MPS are available - defaulting to CPU. Note: This module is much faster with a GPU.
```

```
# STEP 5: Process Dataset Images and Extract Plates
reader = easyocr.Reader(['en'])

print("\n🔍 Scanning dataset images...")

registered_plates_data = []

# ✅ Change: Recursively search for image files in all subdirectories
```

```
image_extensions = ('.jpg', '.jpeg', '.png')
files_in_dataset = []

for root, dirs, files in os.walk(dataset_dir):
    for file in files:
        if file.lower().endswith(image_extensions):
            full_path = os.path.join(root, file)
            files_in_dataset.append(full_path)

print(f"📁 Found {len(files_in_dataset)} image files in dataset.")

if not files_in_dataset:
    print("❌ No image files found. Please check dataset structure.")

# ✅ Loop over full image paths found
for image_path in files_in_dataset:
    print(f"\n📷 Processing image: {image_path}")

    plate_text = extract_plate_number(image_path)
    print(f"📝 OCR Output: {plate_text}")

    registered_plates_data.append({
        'Filename': os.path.basename(image_path),
        'Plate': plate_text
    })

    print(f"✅ {os.path.basename(image_path)} ➜ Plate Detected: '{plate_text}'")
```

```
WARNING:easyocr.easyocr:Neither CUDA nor MPS are available - defaulting to CPU. Note: This module is much faster with a GP

🔍 Scanning dataset images...
📁 Found 433 image files in dataset.

📷 Processing image: registered_plates/images/Cars48.png
📷 Reading with EasyOCR: registered_plates/images/Cars48.png
📝 EasyOCR Output: ALR 486
📝 OCR Output: ALR 486
✅ Cars48.png ➜ Plate Detected: 'ALR 486'

📷 Processing image: registered_plates/images/Cars93.png
📷 Reading with EasyOCR: registered_plates/images/Cars93.png
📝 EasyOCR Output: CUSR
📝 OCR Output: CUSR
✅ Cars93.png ➜ Plate Detected: 'CUSR'

📷 Processing image: registered_plates/images/Cars169.png
📷 Reading with EasyOCR: registered_plates/images/Cars169.png
📝 EasyOCR Output: PRIV ATE
📝 OCR Output: PRIV ATE
✅ Cars169.png ➜ Plate Detected: 'PRIV ATE'

📷 Processing image: registered_plates/images/Cars310.png
📷 Reading with EasyOCR: registered_plates/images/Cars310.png
📝 EasyOCR Output: LGEIT QIL STRANGEBEAVER CON
📝 OCR Output: LGEIT QIL STRANGEBEAVER CON
✅ Cars310.png ➜ Plate Detected: 'LGEIT QIL STRANGEBEAVER CON'

📷 Processing image: registered_plates/images/Cars98.png
📷 Reading with EasyOCR: registered_plates/images/Cars98.png
📝 EasyOCR Output: ALI KULT MIRRY MCTCO RARROTHTAD CLUTE ALAMY STOCK PHOTO
📝 OCR Output: ALI KULT MIRRY MCTCO RARROTHTAD CLUTE ALAMY STOCK PHOTO
✅ Cars98.png ➜ Plate Detected: 'ALI KULT MIRRY MCTCO RARROTHTAD CLUTE ALAMY STOCK PHOTO'

📷 Processing image: registered_plates/images/Cars127.png
📷 Reading with EasyOCR: registered_plates/images/Cars127.png
⚠️ EasyOCR failed to detect any text.
📝 OCR Output: NOT_DETECTED
✅ Cars127.png ➜ Plate Detected: 'NOT_DETECTED'

📷 Processing image: registered_plates/images/Cars370.png
📷 Reading with EasyOCR: registered_plates/images/Cars370.png
📝 EasyOCR Output: CHO1 ANOOO1
📝 OCR Output: CHO1 ANOOO1
✅ Cars370.png ➜ Plate Detected: 'CHO1 ANOOO1'

📷 Processing image: registered_plates/images/Cars423.png
📷 Reading with EasyOCR: registered_plates/images/Cars423.png
📝 EasyOCR Output: P3RV_P AE
📝 OCR Output: P3RV_P AE
✅ Cars423.png ➜ Plate Detected: 'P3RV_P AE'

📷 Processing image: registered_plates/images/Cars237.png
📷 Reading with EasyOCR: registered_plates/images/Cars237.png
📝 EasyOCR Output: @OA
📝 OCR Output: @OA
✅ Cars237.png ➜ Plate Detected: '@OA'
```

```
# STEP 6: Create DataFrame
df_registered = pd.DataFrame(registered_plates_data)

print("\n🧾 Columns in DataFrame:", df_registered.columns.tolist())

if df_registered.empty:
    print("⚠️ No plates found! Check OCR or dataset.")
else:
    print(f"✅ Total registered plates: {len(df_registered)}")
    display(df_registered)
```

```
🧾 Columns in DataFrame: ['Filename', 'Plate']
✅ Total registered plates: 433
```

	Filename	Plate
0	Cars48.png	ALR 486
1	Cars93.png	CUSR
2	Cars169.png	PRIV ATE
3	Cars310.png	LGEIT QIL STRANGEBEAVER CON
4	Cars98.png	ALI KULT MIRRY MCTCO RARROTHTAD CLUTE ALAMY ST...
...	...	...
428	Cars114.png	CCGRAPHICS AP 29 BP 585
429	Cars184.png	ELIL
430	Cars182.png	VZLAF
431	Cars398.png	8 @16 M SATOYOIA
432	Cars118.png	JA6Z UAR

433 rows × 2 columns

```
# STEP 7: Convert to List for Matching
registered_plates = df_registered['Plate'].tolist()
```

```
# Re-import in case 'files' got overwritten
from google.colab import files

# STEP 8: Upload a Test Image
print("\n📤 Upload an image of a vehicle license plate to verify...")
uploaded = files.upload()

if uploaded:
    user_img_name = list(uploaded.keys())[0]
    user_img_path = os.path.join(os.getcwd(), user_img_name)

    # Validate image file type
    if user_img_name.lower().endswith(('.jpg', '.jpeg', '.png')):
        # STEP 9: Extract Plate from Uploaded Image
        user_plate = extract_plate_number(user_img_path)
        print(f"\n🔍 Detected License Plate: {user_plate}")

        # STEP 10: Match and Display Result
        if user_plate in registered_plates:
            print("✅ Welcome, You are allowed to park your vehicle.")
        else:
            print("⚠️ Warning, Your vehicle is not registered.")
    else:
        print("⛔ Unsupported file type. Please upload a .jpg, .jpeg, or .png image.")
else:
    print("❌ No image uploaded.")
```

```
📤 Upload an image of a vehicle license plate to verify...
```

Choose Files No file chosen Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.

```
Saving Cars10.png to Cars10.png
📷 Reading with EasyOCR: /content/Cars10.png
📝 EasyOCR Output: TN 37 CS 2765

🔍 Detected License Plate: TN 37 CS 2765
✅ Welcome, You are allowed to park your vehicle.
```