

# HASIL STRESS TEST ENGINE FR BARU – MENGGUNAKAN LIBRARY OPEN SOURCE DEEPFACE

## Skenario 1 : tanpa embedded file

- Embedded file berfungsi mempercepat waktu processing dengan memanfaatkan processing yang telah dilakukan sebelumnya
- Embedded file harus dihapus agar kemudian digenerate ulang untuk dapat mendeteksi perubahan gambar (penambahan wajah baru atau penghapusan wajah lama)

- a) Test 1: 50 foto di identify dengan 1 foto. Hasilnya engine berhasil melakukan processing. Menghasilkan beberapa url foto yang cocok. Durasi processing adalah 19 detik.

The screenshot shows the VS Code interface with two main panes. On the left, the code editor displays a Python file named `main.py` containing code for a DeepFace-Flask application. It includes imports for DeepFace, os, and json, along with logic for reading images from a folder, performing face detection, and saving results. On the right, the Postman application is open, showing a POST request to `http://127.0.0.1:5000/core/identify`. The request body is a JSON object with fields: "transient": true, "face\_detector": "dlib", and "image": "data:image/jpeg;base64,[REDACTED]". The response pane shows a JSON array of 12 elements, each representing a comparison result between the uploaded photo and one of the 50 images in the database. The array includes fields like "name", "image", "similar", and "matches\_filename".

- b) Test 2: 100 foto di identify dengan 1 foto. Hasilnya engine berhasil melakukan processing. Menghasilkan beberapa url foto yang cocok. Durasi processing adalah 41 detik.

This screenshot is similar to the previous one but shows a more recent execution of the code. The terminal output indicates that 100 images were processed in 2.451 seconds. The Postman response shows 12 matches, with the first few matching images being "foto-0qyXc\_U" and "foto-0qyKz\_U". The JSON response is identical to the one in the previous screenshot, listing 12 matches with their respective names, images, similarity scores, and matched filenames.

- c) Test 3: 200 foto di identify dengan 1 foto. Hasilnya engine berhasil melakukan processing. Menghasilkan beberapa url foto yang cocok. Durasi processing adalah 77 detik.

The screenshot shows a development setup with two windows. On the left, the VS Code Explorer shows a folder named 'DEEPCODE-FLASK' containing 'main.py'. The code in 'main.py' includes logic for file conversion and AI processing. On the right, a Postman window displays a POST request to 'http://127.0.0.1:5000/core/identify'. The response body shows JSON data representing a comparison between two images.

```

    {
      "name": "compare_images",
      "message": "success to compare two images",
      "code": 200,
      "data": {
        "is_match": true
      }
    }
  
```

- d) Test 4: 500 foto di identify dengan 1 foto. Hasilnya engine berhasil melakukan processing. Menghasilkan beberapa ulr foto yang cocok. Durasi processing adalah 3menit 18detik.

This screenshot is similar to the previous one, showing the same development environment. The code in 'main.py' has been modified to handle 500 images. The Postman response shows a successful comparison result for 500 images.

```

    {
      "name": "compare_images",
      "message": "success to compare two images",
      "code": 200,
      "data": {
        "is_match": true
      }
    }
  
```

- e) Test 5: 1000 foto di identify dengan 1 foto. Hasilnya engine berhasil melakukan processing. Menghasilkan beberapa ulr foto yang cocok. Durasi processing adalah 6menit 51detik.

Screenshot of a developer's workspace showing two open tabs in a code editor and a browser.

**Code Editor (left):**

- File: main.py
- Content: Python code for DeepFace Flask, specifically the identify\_images function. It uses OpenCV to convert base64 strings to images, finds faces in them, and compares them against a database. A warning about deprecated TensorFlow code is present.

**Browser Tab (right):**

- Title: Deepface Flask / identify
- Method: POST
- URL: http://127.0.0.1:5000/core/identify
- Body (JSON): A JSON object containing tenant information, face detection parameters, and a data field with a large string of base64 encoded images.
- Response: 200 OK, 6 m 51.60 s, 24.98 KB

Bottom status bar: 30°C, Cerah, 3:31 PM, 12/29/2023

## Skenario 2 : dengan embedded file

- Dengan adanya embedded file waktu processing menjadi semakin singkat
- Dengan catatan tidak ada perubahan terhadap database foto
- Apabila ada perubahan pada database foto maka harus mengenerate embedded file baru

- a) Test 1: 50 foto di identify dengan 1 foto memanfaatkan embedded file. Hasilnya engine sukses melakukan processing, total waktunya adalah 0.4 detik Menghasilkan beberapa URL foto yang cocok.

The screenshot shows a code editor with Python code for a DeepFace-Flask application. The code includes logic for identifying faces in images and handling file uploads. A terminal window shows the application running on port 5000. To the right, a Postman interface is used to make a POST request to the '/core/identify' endpoint with a JSON body containing a base64 encoded image. The response is a 200 OK status with a JSON object containing a URL for the identified photo.

```

    # Konversi base64 ke file .jpg dan local write .jpg
    filename = str(uuid.uuid4()) + ".jpg"
    base64_to_jpg(base64_image1, filename)

    # AI:
    try:
        df = DeepFace.find(os.path.join(
            tempDir, filename), db_path="images")
        df = df[0]
    finally:
        # LOCAL DELETE : delete file .jpg in 'temp' directory
        os.remove(os.path.join(tempDir, filename))

```

- b) Test 2: 100 foto di identify dengan 1 foto memanfaatkan embedded file. Hasilnya engine sukses melakukan processing, total waktunya adalah 0.4 detik Menghasilkan beberapa URL foto yang cocok.

This screenshot is similar to the previous one, showing the same code editor and terminal output. The difference is in the Postman request, which now shows a larger list of URLs for identified photos, indicating multiple matches found for the query image. The response time remains at 0.4 seconds.

```

    # Konversi base64 ke file .jpg dan local write .jpg
    filename = str(uuid.uuid4()) + ".jpg"
    base64_to_jpg(base64_image1, filename)

    # AI:
    try:
        df = DeepFace.find(os.path.join(
            tempDir, filename), db_path="images", enforce_detection=False)
        df = df[0]
    finally:
        # LOCAL DELETE : delete file .jpg in 'temp' directory
        os.remove(os.path.join(tempDir, filename))

```

- c) Test 3: 200 foto di identify dengan 1 foto memanfaatkan embedded file. Hasilnya engine sukses melakukan processing, total waktunya adalah 0.5 detik Menghasilkan beberapa URL foto yang cocok.

```

main.py > identify_images
    ... (code for DeepFace.find and file processing)
    finally:
        # LOCAL DELETE : delete file .jpg in 'temp' directory
        os.remove(os.path.join(tempDir, filename))

    # CHECK : is pandas dataframe empty or not
    # empty -> face not match, not empty -> face match
    # define default value
    isMatch = False
    tempMatchesfilename = []
    if df.shape[0] > 0:
        print("user sudah terdaftar, silahkan LOGIN")

```

POST http://127.0.0.1:5000/core/identify

Body:

```

1 ["name": "compare_images", "message": "success to compare two images", "code": 200,
2   "data": {"is_match": true},
3   "current_filename": "temp/418554ab-6e4b-41d6-a3a-add24aa7f1a.jpg", "matches_filename":
4     ["temp/130ufY1CuB56yF.png", "images/foto-130ufY1CuB56yF.png", "images/
      foto-14pMzQkEgkTq8Iy.png"],
5   "images": ["foto-1WvWmWpuugl7v5.png", "images/foto-1Ksxt7050x50cce.C.png", "images/
      foto-28WzAsd4ppHca8.png"],
6   "images": ["foto-YXfKwMCwfEY76m.png", "images/foto-2QveQ4MhJOpn47.png", "images/
      foto-2t7mWpElfhkbqj.png"], ...
12 "images": ["foto-1g7GhuwzIBRR1RM0.png", "images/foto-20KdxVvxtQtmobL.png", "images/
      foto-9CmNdwsv7K1M61.png"],
13 "images": ["foto-0PSUCh1Dipy6eA.png", "images/foto-8Usa9fQuoZzLifNY.png", "images/
      foto-2KXBKyEmtSw41P.png"],
14 "images": ["foto-8871wBqhw9WeElu.png", "images/foto-1czGwKheVbjjsG0.png", "Images/
      foto-28p1wB8KBeq9qk.png"],
15 "images": ["foto-1sh1LWsjxSAFH.png", "images/foto-14cAUHtEzkN3a9r.png", "Images/
      foto-8Kvd2zLTfKtMqJk.png"],
16 "images": ["foto-1no1NqJNrtM697cp8.png", "images/foto-2HNgpbQkMvzcKF.png", "Images/
      foto-2xWzA2hVzK9xfu.png"]
]

```

- d) Test 4: 500 foto di identify dengan 1 foto memanfaatkan embedded file. Hasilnya engine sukses melakukan processing, total waktunya adalah 0.9 detik Menghasilkan beberapa URL foto yang cocok.

```

main.py > identify_images
    ... (code for DeepFace.find and file processing)
    finally:
        # LOCAL DELETE : delete file .jpg in 'temp' directory
        os.remove(os.path.join(tempDir, filename))

    # CHECK : is pandas dataframe empty or not
    # empty -> face not match, not empty -> face match
    # define default value
    isMatch = False
    tempMatchesfilename = []
    if df.shape[0] > 0:
        print("user sudah terdaftar, silahkan LOGIN")

```

POST http://127.0.0.1:5000/core/identify

Body:

```

1 {"name": "compare_images", "message": "success to compare two images", "code": 200,
2   "data": {"is_match": true},
3   "current_filename": "temp/418554ab-6e4b-41d6-a3a-add24aa7f1a.jpg", "matches_filename":
4     ["temp/130ufY1CuB56yF.png", "images/foto-130ufY1CuB56yF.png", "images/
      foto-14pMzQkEgkTq8Iy.png"],
5   "images": ["foto-1WvWmWpuugl7v5.png", "images/foto-1Ksxt7050x50cce.C.png", "images/
      foto-28WzAsd4ppHca8.png"],
6   "images": ["foto-YXfKwMCwfEY76m.png", "images/foto-2QveQ4MhJOpn47.png", "images/
      foto-2t7mWpElfhkbqj.png"], ...
12 "images": ["foto-1g7GhuwzIBRR1RM0.png", "images/foto-20KdxVvxtQtmobL.png", "images/
      foto-9CmNdwsv7K1M61.png"],
13 "images": ["foto-0PSUCh1Dipy6eA.png", "images/foto-8Usa9fQuoZzLifNY.png", "images/
      foto-2KXBKyEmtSw41P.png"],
14 "images": ["foto-8871wBqhw9WeElu.png", "images/foto-1czGwKheVbjjsG0.png", "Images/
      foto-28p1wB8KBeq9qk.png"],
15 "images": ["foto-1sh1LWsjxSAFH.png", "images/foto-14cAUHtEzkN3a9r.png", "Images/
      foto-8Kvd2zLTfKtMqJk.png"],
16 "images": ["foto-1no1NqJNrtM697cp8.png", "images/foto-2HNgpbQkMvzcKF.png", "Images/
      foto-2xWzA2hVzK9xfu.png"]
]

```

- e) Test 5: 1000 foto di identify dengan 1 foto memanfaatkan embedded file. Hasilnya engine sukses melakukan processing, total waktunya adalah 1.3 detik Menghasilkan beberapa URL foto yang cocok.

Screenshot of a development environment showing a DeepFace Flask application.

**Code Editor:** The main editor shows `main.py` with code related to image processing and DeepFace identification.

```

    def identify_images():
        base64_image = body.get("data")
        # Konversi base64 ke file .jpg dan local write .jpg
        filename = str(uuid.uuid4()) + ".jpg"
        base64_to_jpg(base64_image, filename)
        # AI:
        try:
            df = DeepFace.find(os.path.join(tempDir, filename), db_path="images", enforce_det=True)
            df = df[0]
        finally:
            # LOCAL DELETE : delete file .jpg di temp directory
            os.remove(os.path.join(tempDir, filename))
        # CHECK : is pandas dataframe empty or not
        if len(df) == 0:
            # define default value
            isMatch = False
            tempMatchesFilename = []
        else:
            if df.shape[0] > 0:
                print("User sudah terdaftar, silahkan LOGIN")
    
```

**Terminal:** Shows logs from a production WSGI server.

```

    * Running on http://127.0.0.1:5000
    Press CTRL+C to quit
    Restarting with stat
    WARNING:TensorFlow:From C:\Users\jokodho\AppData\Local\Programs\Python\Python310\lib\site-packages\tensorflow\src\losses\py\_2976: The name tf.losses.sparse_softmax_cross_entropy is deprecated. Please use tf.compat.v1.losses.sparse_softmax_cross_entropy instead.
    * Debugger is active!
    * Debugger PIN: 122-041-852
    File representations_vgg_face.pkl | 1000/1000 [06:50:00:00, 2.46it/s]
    23-12-29 15:11:51 - Representations stored in images/representations_vgg_face.pkl file.
    Please delete this file when you add new identities in your database.
    23-12-29 15:11:52 - Find function lasts 411.4649219512995 seconds
    user sudah terdaftar, silahkan LOGIN
    23-12-29 15:11:52 - [29/Oct/2023 15:11:52] "POST /core/identify HTTP/1.1" 200 -
    1018 representations found in representations_vgg_face.pkl
    23-12-29 15:11:52 - There are 1018 representations found in representations_vgg_face.pkl
    23-12-29 15:11:52 - Find function lasts 1.3504108988020312 seconds
    user sudah terdaftar, silahkan LOGIN
    23-12-29 15:11:52 - [29/Oct/2023 15:11:52] "POST /core/identify HTTP/1.1" 200 -
    
```

**Postman:** A POST request is being made to `http://127.0.0.1:5000/core/identify`.

**Body:**

```

    {
        "tenant": "devdips",
        "face_detector": 0,
        "data": "/9j/4AAQSkZJngABQAAAABIAAO/4QCMXhpZgAATUAGkAAAAABQESAAAABAEAAAaAUAAAABAAAASgeBAUAAAABAAAUsgeAMAAA
    
```

**Network:** Shows a successful 200 OK response with 1425 ms duration and 24.98 KB size.