LAMPIRAN

Lampiran 1 *Source Code* import library

```
import os
import sys
import logging
import tempfile
import time
import uuid
import threading
import requests
import json
  import cv2
  import numpy as np
  import pickle
  from datetime import datetime
  from flask import Flask, request, jsonify
  from flask_cors import CORS
  from scipy.spatial.distance import cosine, euclidean
  import warnings
  warnings.filterwarnings('ignore')
```

Lampiran 2 *Source Code* konfigurasi otomatis

```
PORT = 5001
DATASET_FOLDER = "dataset"  # Folder yang berisi subfolder per
identity
DATABASE_DIR = "face_database"
LARAVEL_URL = https://testcardeki.my.id/v1
```

Lampiran 3 *Source Code* set up log

```
logging.basicConfig(
level=logging.INFO,
format='%(asctime)s - %(levelname)s - %(message)s',
handlers=[
logging.FileHandler(f'{DATABASE_DIR}/logs/server.log'),
logging.StreamHandler()
]
)
logger = logging.getLogger(__name__)
```

Lampiran 4 *Source Code* install depencies

```
1. def install_dependencies():
2. required = ['deepface', 'scipy', 'opencv-python', 'tensorflow',
   'flask', 'flask-cors', 'pyngrok']
3. for package in required:
4. try:
5. __import__(package)
6. except ImportError:
7. os.system(f"{sys.executable} -m pip install {package}")
8. try:
9. import deepface
10.   return True
11.   except:
12.   return False
```

Lampiran 5 *Source Code* VGG-Face API

```
1. class SimpleVGGFaceAPI:
2. def __init__(self):
3. if not install_dependencies():
4. raise Exception("Failed to install dependencies")
5. from deepface import DeepFace
6. import tensorflow as tf
7. tf.config.threading.set_inter_op_parallelism_threads(0)
8. tf.config.threading.set_intra_op_parallelism_threads(0)
9. tf.config.set_visible_devices([], 'GPU')
10.   self.DeepFace = DeepFace
11.   self.embeddings_cache = {}
12.   self.embeddings_file =
   f"{DATABASE_DIR}/embeddings/vggface_cache.pkl"
13.   self.config = {
14.   'model_name': 'VGG-Face',
15.   'detector_backend': 'opencv',
16.   'threshold': 0.68
17.   }
18.   self.load_cache()
19.   self.auto_scan_dataset()
20.   def load_cache(self):
21.   if os.path.exists(self.embeddings_file):
22.   with open(self.embeddings_file, 'rb') as f:
23.   data = pickle.load(f)
24.   self.embeddings_cache = data.get('embeddings', data) if
   isinstance(data, dict) else data
25.   def save_cache(self):
26.   cache_data = {
27.   'embeddings': self.embeddings_cache,
28.   'saved_at': datetime.now().isoformat(),
```

```
29.   'model': 'VGG-Face'
30.   }
31.   with open(self.embeddings_file, 'wb') as f:
32.   pickle.dump(cache_data, f)
33.   def auto_scan_dataset(self):
34.   if not os.path.exists(DATASET_FOLDER):
35.   return
36.   person_folders = [f for f in os.listdir(DATASET_FOLDER) if
   os.path.isdir(os.path.join(DATASET_FOLDER, f))]
37.   for person_name in person_folders:
38.   person_path = os.path.join(DATASET_FOLDER, person_name)
39.   image_files = [f for f in os.listdir(person_path) if
   f.lower().endswith(('.jpg', '.jpeg', '.png', '.bmp'))]
40.   embeddings = []
41.   for image_file in image_files:
42.   embedding = self.compute_embedding(os.path.join(person_path,
   image_file))
43.   if embedding is not None:
44.   embeddings.append(embedding)
45.   if embeddings:
46.   avg_embedding = np.mean(embeddings, axis=0)
47.   avg_embedding = avg_embedding / np.linalg.norm(avg_embedding)
48.   self.embeddings_cache[person_name] = avg_embedding
49.   self.save_cache()
50.   def compute_embedding(self, image_path, max_retries=5):
51.   for attempt in range(max_retries):
52.   try:
53.   img = cv2.imread(image_path)
54.   if img is None:
55.   time.sleep(1)
56.   continue
57.   result = self.DeepFace.represent(
58.   img_path=image_path,
59.   model_name=self.config['model_name'],
60.   detector_backend=self.config['detector_backend'],
61.   enforce_detection=False,
62.   normalization='base'
63.   )
64.   embedding = np.array(result[0]['embedding']) if
   isinstance(result, list) else np.array(result['embedding'])
65.   return embedding / np.linalg.norm(embedding)
66.   except:
67.   time.sleep((attempt + 1) * 2)
68.   return None
69.   def verify_face(self, image_file, max_retries=3):
70.   temp_path = f"{DATABASE_DIR}/temp/{uuid.uuid4().hex}.jpg"
71.   image_file.save(temp_path)
72.   input_embedding = self.compute_embedding(temp_path)
73.   os.remove(temp_path)
74.   if input_embedding is None:
75.   return {'success': False, 'message': 'Failed to process
   image'}
```

```
76.   best_match = None
77.   best_similarity = 0.0
78.   for identity, cached_embedding in
   self.embeddings_cache.items():
79.   cosine_sim = 1 - cosine(input_embedding, cached_embedding)
80.   euclidean_dist = euclidean(input_embedding, cached_embedding)
81.   similarity = cosine_sim * 0.8 + (1 - euclidean_dist / 100) *
   0.2
82.   if similarity > best_similarity:
83.   best_similarity = similarity
84.   best_match = identity
85.   if best_similarity >= self.config['threshold']:
86.   return {'success': True, 'verified': True, 'user_id':
   best_match, 'confidence': best_similarity}
87.   else:
88.   return {'success': False, 'verified': False, 'message': 'Not
   recognized'}
```

Lampiran 6 *Source Code* fungsi utama

```
1. def main():
2. global vggface_api
3. vggface_api = SimpleVGGFaceAPI()
4. app = create_app()
5. app.run(host='0.0.0.0', port=PORT, debug=False)
6. if __name__ == '__main__':
7. main()
```

Lampiran 7 Hasil Cek Turnitin

KOMPARASI ANTARA YOLO DAN DEEPFACE UNTUK SISTEM PENCATATAN PEMINJAMAN MOBIL INVENTARIS PADA PT. ASTRA INTERNATIONAL DAIHATSU CABANG PRAMUKA

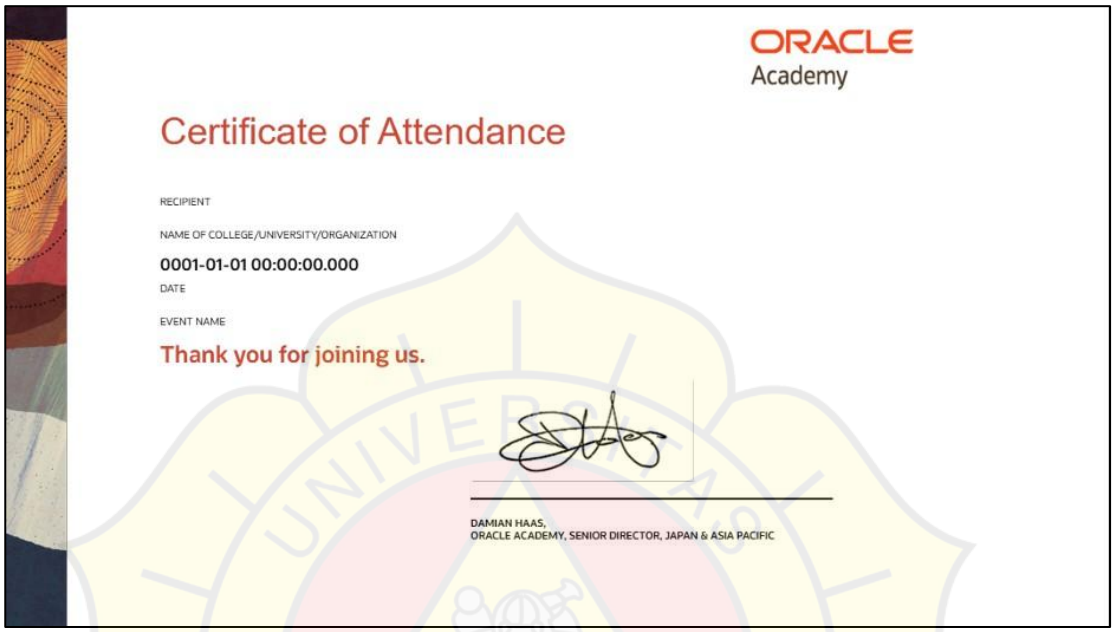
ORIGINALITY REPORT

17%	16%	7%	%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	jurnal.stmikasia.ac.id Internet Source	1 %
2	repository.its.ac.id Internet Source	1 %
3	ejournal.itn.ac.id Internet Source	1 %

Lampiran 8 Sertifikat Oracle Academy



ORACLE
Academy

Certificate of Attendance

RECIPIENT

NAME OF COLLEGE/UNIVERSITY/ORGANIZATION

0001-01-01 00:00:00.000
DATE

EVENT NAME

Thank you for joining us.

DAMIAN HAAS,
ORACLE ACADEMY, SENIOR DIRECTOR, JAPAN & ASIA PACIFIC

Lampiran 9 Sertifikat Cisco "PC Hardware and Software"

cisco Networking Academy

This certificate is awarded to

RIZKI ARDI KURNIAWAN KUSWORO

for successfully completing

IT Essentials: PC Hardware and Software

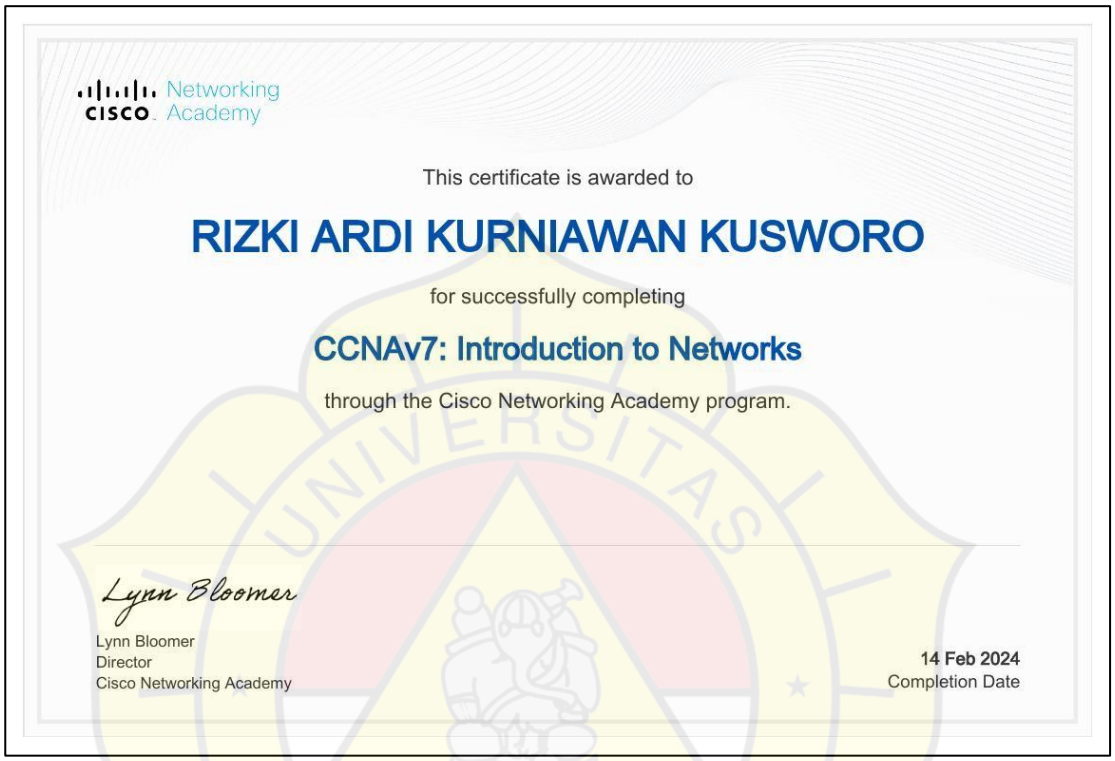
through the Cisco Networking Academy program.

Lynn Bloomer

Lynn Bloomer
Director
Cisco Networking Academy

28 Feb 2022
Completion Date

Lampiran 10 Sertifikat Cisco “CCNAv7: Introduction to Networks”



Networking Academy

This certificate is awarded to

RIZKI ARDI KURNIAWAN KUSWORO

for successfully completing

CCNAv7: Introduction to Networks

through the Cisco Networking Academy program.

Lynn Bloomer

Lynn Bloomer
Director
Cisco Networking Academy

14 Feb 2024
Completion Date