# LAMPIRAN A

Listing Program Python

```
import cv2
import numpy as np
from tensorflow.keras.models import load_model
from tensorflow.keras.preprocessing.image import img_to_array
import pygame

# Path to the audio files
audio_real_path = '/home/pi/Documents/model/assets/asli'
audio_fake_path = '/home/pi/Documents/model/assets/palsu'
audio_unknow_path                                                        =
'/home/pi/Documents/model/assets/tidak_tahu.mp3'
# Load the model
modeljadi                                                                =
load_model('/home/pi/Documents/model/assets/modelfakeorreal.h5')

# Function to play audio
def play_audio(audio_path):
    # Initialize pygame mixer
    pygame.mixer.init()
    # Load the audio file
    pygame.mixer.music.load(audio_path)
    # Play the audio file
    pygame.mixer.music.play()
    # Keep the program running until the audio finishes playing
    while pygame.mixer.music.get_busy():
        pygame.time.Clock().tick(10)

# Function to process image and make prediction
def predict_image(image, model, img_height=128, img_width=128):
    # Convert image to array and normalize
    img_array = img_to_array(image)
    img_array = img_array / 255.0
    # Resize image
    img_array = cv2.resize(img_array, (img_height, img_width))

    # Add batch dimension
    img_array = np.expand_dims(img_array, axis=0)

    # Predict the image
    prediction = model.predict(img_array)

    return prediction

def classify_image(saved_image_path):
    # Load and classify the saved image
    saved_image = cv2.imread(saved_image_path)
    image_rgb = cv2.cvtColor(saved_image, cv2.COLOR_BGR2RGB)
    prediction = predict_image(image_rgb, modeljadi)

    # Display result
    if prediction[0] > 0.5 and prediction[0] < 1:
        label = 'Real'
    elif prediction[0] == 1:
        label = 'tidak tahu'
    else:
        label = 'Fake'

    print(f'Prediksi: {label}')
    print(prediction[0])

    # Check label and play corresponding audio
    if label == 'Real':
        play_audio(audio_real_path)
    elif label == 'tidak tahu':
        play_audio(audio_unknow_path)
    else:
        play_audio(audio_fake_path)

    return label
```

```
import pygame
import cv2
import numpy as np
from tensorflow.keras.preprocessing import image
from tensorflow.keras.models import load_model

# Dictionary label for denominations and corresponding audio paths
class_labels = {
0:('1000','/content/drive/MyDrive/clinet/uangpalsuataureal/audio/satuk.mp3'),
1:('10000','/content/drive/MyDrive/clinet/uangpalsuataureal/audio/sepuluhk.mp3'),
2:('100000','/content/drive/MyDrive/clinet/uangpalsuataureal/audio/seratusk.mp3'),
3:('2000','/content/drive/MyDrive/clinet/uangpalsuataureal/audio/duak.mp3'),
4:('20000','/content/drive/MyDrive/clinet/uangpalsuataureal/audio/duapuluhk.mp3'),
5:('5000','/content/drive/MyDrive/clinet/uangpalsuataureal/audio/limak.mp3'),
6:('50000','/content/drive/MyDrive/clinet/uangpalsuataureal/audio/limapuluhk.mp3'),
7:('75000','/content/drive/MyDrive/clinet/uangpalsuataureal/audio/tujuhlimak.mp3')}

# Load the model
modeljadi=load_model('/content/drive/MyDrive/clinet/uangpalsuataureal/model/modelnominal.h5')

# Function to play audio
def play_audio(audio_path):
    # Initialize pygame mixer
    pygame.mixer.init()
    # Load the audio file
    pygame.mixer.music.load(audio_path)
    # Play the audio file
    pygame.mixer.music.play()
    # Keep the program running until the audio finishes playing
    while pygame.mixer.music.get_busy():
        pygame.time.Clock().tick(10)

def preprocess_image(img):
    # Convert to grayscale
    img = cv2.cvtColor(img, cv2.COLOR_RGB2GRAY)
    # Adjust contrast using histogram equalization
    img = cv2.equalizeHist(img)
    # Convert back to 3 channels
    img = cv2.cvtColor(img, cv2.COLOR_GRAY2RGB)
    return img

# Function to process image and make prediction
def prepare_image(image_path, target_size):
    img = image.load_img(image_path, target_size=target_size)
    img_array = image.img_to_array(img)
    # Preprocess image
    img_array = preprocess_image(img_array.astype(np.uint8))
    img_array = np.expand_dims(img_array, axis=0)  # Add batch dimension
    img_array = img_array / 255.0  # Normalize
    return img_array

def classify_nominal_image(saved_image_path):
    # Prepare the captured image for prediction
    prepared_image = prepare_image(saved_image_path, target_size=(128, 128))

    # Make prediction
    predictions = modeljadi.predict(prepared_image)

    # Get the class with the highest probability
    predicted_class = np.argmax(predictions, axis=1)[0]

    # Get the predicted class label and corresponding audio path
    predicted_class_label, audio_path = class_labels[predicted_class]
```

```
# Play the corresponding audio
play_audio(audio_path)

return predicted_class_label
```

## LAMPIRAN B

# Hasil Alat dan Pengujian

Gambar A. Alat Deteksi Uang Tampak Depan

Gambar B. Alat Deteksi Uang Tampak Belakang

## 1.1 Biodata Penulis

Nama : Febrian Syach Pangestu

Tempat/Tanggal Lahir : Cilacap, 04 Februari 2003

Alamat : JL. Merapi RT03/RW05, No.C2, Sidanegara, Cilacap.

Email : febriansyachpangestu77@gmail.com

Telepon/Hp : 081329085440

Hobi : Menonton Film

Moto : Bersyukur dan Tetap semangat

Riwayat Pendidikan :

- SD Negeri 06 Petungan Tahun 2009 – 2015
- SMP Negeri 1 Cilacap Tahun 2015 – 2018
- SMK Negeri 2 Cilacap Tahun 2018 – 2021
- Politeknik Negeri Cilacap Tahun 2021 – 2024
  Prodi D3 – Teknik Elektronika

Penulis telah mengikuti sidang Tugas Akhir pada tanggal 14 Agustus 2024 sebagai salah satu persyaratan untuk memperoleh gelar Ahli Madya (A.Md).