```
import RPi.GPIO as GPIO
from time import sleep, time
import subprocess
import numpy as np
import os
import shutil
from tensorflow.keras.models import load_model
from tensorflow.keras.preprocessing import image
import firebase_admin
from firebase_admin import credentials, storage

# Load the trained model
model = load_model(’/home/fypml/Desktop/FYP/fyp_model_96_test.h5’)

# Initialize Firebase
cred =
credentials.Certificate("/home/fypml/Desktop/FYP/fyp-disease-detection-firebase-adminsdk-s44wo-f7254b
97c2.json")
firebase_admin.initialize_app(cred, {
    ’storageBucket’: ’fyp-disease-detection.appspot.com’,
    ’databaseURL’: ’https://fyp-disease-detection-default-rtdb.europe-west1.firebasedatabase.app’
})

# Set up GPIO mode and pin numbering
GPIO.setmode(GPIO.BOARD)
GPIO.setwarnings(False)

# Define GPIO pins for motor control
left_motor_pin = 15
right_motor_pin = 13
left_sensor_pin = 7
right_sensor_pin = 10

# Set up GPIO pins for motor control
GPIO.setup(left_motor_pin, GPIO.OUT)
GPIO.setup(right_motor_pin, GPIO.OUT)

# Set up GPIO pins for sensor inputs
GPIO.setup(left_sensor_pin, GPIO.IN)
GPIO.setup(right_sensor_pin, GPIO.IN)

# Define motor control functions
def leftOn():
    GPIO.output(left_motor_pin, GPIO.HIGH)

def leftOff():
    GPIO.output(left_motor_pin, GPIO.LOW)

def rightOn():
    GPIO.output(right_motor_pin, GPIO.HIGH)

def rightOff():
    GPIO.output(right_motor_pin, GPIO.LOW)

def stopAll():
    GPIO.output(left_motor_pin, GPIO.LOW)
    GPIO.output(right_motor_pin, GPIO.LOW)

# Define LED pins
LED_PINS = {
    ’Aphids’: 11,
    ’Army Worm’: 16,
    ’Bacterial Blight’: 37,
    ’Powdery Mildew’: 29,
    ’Target Spot’: 36,
    ’Healthy’: 22
}

# Set up LED pins as outputs
for pin in LED_PINS.values():
    GPIO.setup(pin, GPIO.OUT)

def turn_on_led(label):
    # Get the corresponding GPIO pin for the label
    pin = LED_PINS.get(label)
    if pin is not None:
        GPIO.output(pin, GPIO.HIGH)
        print(f"LED for {label} turned on.")
    else:
        print(f"No LED pin defined for {label}.")

def turn_off_all_leds():
    # Turn off all LEDs
    for pin in LED_PINS.values():
        GPIO.output(pin, GPIO.LOW)
    print("All LEDs turned off.")

def capture_image():
    image_path = ’/home/fypml/Desktop/FYP/pic/captured_image.jpg’  # Define the path to save the
captured image
    subprocess.run([’libcamera-still’, ’-o’, image_path])
    print(f"Image captured and saved to {image_path}")
    return image_path

def classify_image(image_path):
    img = image.load_img(image_path, target_size=(256, 256))
    img_array = image.img_to_array(img)
    img_array = np.expand_dims(img_array, axis=0)
    img_array /= 255.0  # Normalize the image data to 0-1 range

    # Make prediction
    predictions = model.predict(img_array)
    predicted_class = np.argmax(predictions, axis=1)[0]  # Get the index of max value

    # Map the model’s predicted index to specific class numbers
    class_labels = {0: ’Aphids’, 1: ’Army Worm’, 2: ’Bacterial Blight’, 3: ’Healthy’, 4: ’Powdery Mildew’, 5:
’Target Spot’}
    predicted_label = class_labels[predicted_class]
    print(f"Predicted class: {predicted_label} ({predicted_class})")
    return predicted_label

def upload_to_firebase(image_path, predicted_label):
    # Rename the image file with the predicted label
    base_path = os.path.dirname(image_path)
    new_image_path = os.path.join(base_path, f"{predicted_label}.jpg")
    shutil.move(image_path, new_image_path)

    # Put your local file path
    blob = storage.bucket().blob(’images/’ + os.path.basename(new_image_path))

    # Open the file in binary mode
    with open(new_image_path, ’rb’) as image_file:
        blob.upload_from_file(image_file, content_type=’image/jpg’)

    # Get the URL of the uploaded image file
    url = blob.public_url
    print(f’Image uploaded to {url}’)
    print(f’Image deleted from Raspberrypi’)
    return url

try:

    sleep(3)  # Wait for 3 seconds before capturing the next image
    while True:
        # Run for 5 seconds
        print(f"Car started Running in 2 secs")
        sleep(2)
        start_time = time()
        while time() - start_time < 5:
            if GPIO.input(left_sensor_pin) == 0 and GPIO.input(right_sensor_pin) == 0:
                leftOff()
                rightOff()
            elif GPIO.input(left_sensor_pin) == 1 and GPIO.input(right_sensor_pin) == 1:
                leftOn()
                rightOn()
            elif GPIO.input(left_sensor_pin) == 1 and GPIO.input(right_sensor_pin) == 0:
                leftOn()
                rightOff()
            elif GPIO.input(left_sensor_pin) == 0 and GPIO.input(right_sensor_pin) == 1:
                leftOff()
                rightOn()

        # Stop for 8 seconds
        stopAll()
        sleep(8)

        image_path = capture_image()
        sleep(2)
        predicted_label = classify_image(image_path)  # Classify the image
        turn_off_all_leds()  # Turn off all LEDs before turning on the specific one
        print(f"Classification result: {predicted_label}")

        turn_on_led(predicted_label)  # Turn on the LED corresponding to the predicted label
        upload_to_firebase(image_path, predicted_label)  # Upload the image to Firebase
        turn_off_all_leds()  # Turn off all LEDs before turning on the specific one
        sleep(5)  # Wait for 5 seconds before capturing the next image

except KeyboardInterrupt:
    print("Keyboard interrupt detected. Exiting...")
finally:
    GPIO.cleanup()
```