ДОДАТОК А

Лістинг коду системи керування роботом телеприсутності

Частина перша

```
import json
import os
import sys
import time
import logging
import requests
from datetime import datetime, timedelta
import spidev as SPI
sys.path.append("..")
from lib import LCD_1inch28
from PIL import Image, ImageDraw, ImageFont

# Raspberry Pi pin configuration
RST = 27
DC = 25
BL = 18
bus = 0
device = 0
logging.basicConfig(level=logging.DEBUG)



API_KEY = '345cfdcc1b4ef42a7bc3a4123d88eeb0'
```

```
    CITY = 'Poltava,UA'
    URL = f'https://api.openweathermap.org/data/2.5forecast?q={CITY}&units=metric&lang=en&appid={API_KEY}'

    image_folder = "/home/serhii/Desktop/Images"
    weather_images = {
        'clear': 'sunny.png',
        'clouds': 'cloudy.png',
        'rain': 'rain.png',
        'storm': 'storm.png',
        'snow': 'snow.png',
        'default': 'cloudy.png'
    }

    def fetch_weather_data():
        try:
            response = requests.get(URL)
            response.raise_for_status()
            data = response.json()

            with open('response.json', 'w', encoding='utf-8') as f:
                json.dump(data, f, ensure_ascii=False, indent=4)

            print("File 'response.json' created!")
            return data

        except Exception as e:
```

```
        print(f"Error getting data: {e}")
        return None

def load_weather_data():
    if not os.path.exists('response.json'):
        print("File 'response.json' not found. Fetching data...")
        return fetch_weather_data()
    else:
        try:
            with open('response.json', 'r', encoding='utf-8') as f:
                data = json.load(f)
            if not data:
                print("File 'response.json' is empty. Fetching data...")
                return fetch_weather_data()
            return data
        except json.JSONDecodeError as e:
            print(f"Error reading JSON file: {e}")
            return fetch_weather_data()

def get_tomorrow_forecast(data):
    now = datetime.now()
    tomorrow_date = (now + timedelta(days=1)).strftime("%Y-%m-%d")
    target_time = f"{tomorrow_date} 12:00:00"

    for forecast in data["list"]:
        if forecast["dt_txt"] == target_time:
            weather_data = forecast.get("weather", [{}])
            description = weather_data[0].get("description", "No description")
```

```
            return {
                "time": forecast["dt_txt"],
                "temperature": forecast["main"]["temp"],
                "feels_like": forecast["main"]["feels_like"],
                "description": description,
                "wind_speed": forecast["wind"]["speed"],
                "main": forecast["weather"][0]["main"].lower(),
            }

    return None

def display_forecast(disp, font, forecast):
    if forecast is None:
        message = "The forecast for tomorrow is not found!"
        image = Image.new("RGB", (disp.width, disp.height), "BLACK")
        draw = ImageDraw.Draw(image)
        bbox = draw.textbbox((0, 0), message, font=font)
        w, h = bbox[2] - bbox[0], bbox[3] - bbox[1]
        draw.text(
            ((disp.width - w) // 2, (disp.height - h) // 2),
            message,
            fill="WHITE",
            font=font,
        )
        disp.ShowImage(image.rotate(180))
        return
```

```
weather_time = forecast["time"]
date_obj = datetime.strptime(weather_time, "%Y-%m-%d %H:%M:%S")
formatted_date = date_obj.strftime("%d-%m-%Y")

image = Image.new("RGB", (disp.width, disp.height), "BLACK")
draw = ImageDraw.Draw(image)

main_weather = forecast.get("main", "default").lower()
image_file = weather_images.get(main_weather, weather_images["default"])
image_path = os.path.join(image_folder, image_file)

text1 = f"Forecast for tomorrow: "
text2 = f"Time: {formatted_date}"
text3 = f"Temperature: {forecast['temperature']}°C"
text4 = f"Feels like: {forecast['feels_like']}°C"
text5 = f"Description: {forecast['description']}"
text6 = f"Wind speed: {forecast['wind_speed']} m/s"

draw.text((30, 40), text1, fill="WHITE", font=font_small)
draw.text((20, 55), text2, fill="WHITE", font=font_small)
draw.text((10, 70), text3, fill="WHITE", font=font_small)
draw.text((10, 85), text4, fill="WHITE", font=font_small)
draw.text((10, 100), text5, fill="WHITE", font=font_small)
draw.text((10, 115), text6, fill="WHITE", font=font_small)

main_weather = forecast.get("main", "default").lower()
image_file = weather_images.get(main_weather, weather_images["default"])
image_path = os.path.join(image_folder, image_file)
```

```
    if os.path.exists(image_path):
        weather_icon = Image.open(image_path).resize((80, 80))
        image.paste(weather_icon, ((disp.width - 80) // 2, 145))

    disp.ShowImage(image.rotate(180))
    time.sleep(10)

try:
    disp = LCD_1inch28.LCD_1inch28()
    disp.Init()
    disp.clear()
    disp.bl_DutyCycle(50)

    font_common = ImageFont.truetype("./Font/Font01.ttf", 30)
    font_small = ImageFont.truetype("./Font/Font01.ttf", 21)

    line_spacing = 50

    data = load_weather_data()

    image_path_for_femoji = "/home/serhii/Desktop/Emoji/First_emoji.png"
    image_path_for_semoji = "/home/serhii/Desktop/Emoji/Second_emoji.png"
    image_path_for_temoji = "/home/serhii/Desktop/Emoji/Third_emoji.png"
    image_path_for_foemoji = "/home/serhii/Desktop/Emoji/Forth_emoji.png"
    image_path_for_lemoji = "/home/serhii/Desktop/Emoji/Last_emoji.png"

    while True:
```

```
            if os.path.exists(image_path_for_femoji):
                image = Image.open(image_path_for_femoji)
                image = image.resize((disp.width, disp.height))
                disp.ShowImage(image.rotate(180))
            else:
                print(f"File {image_path_for_femoji} is not found")
            time.sleep(6)

            for _ in range(10):
                image = Image.new("RGB", (disp.width, disp.height), "BLACK")
                draw = ImageDraw.Draw(image)

                current_date = datetime.now().strftime("%d-%m-%Y")
                current_time = datetime.now().strftime("%H:%M:%S")
                current_weekday = datetime.now().strftime("%A")

                start_y = 40

                draw.text((20, start_y), f"Date: {current_date}", fill="WHITE",
font=font_common)
                draw.text((20, start_y + line_spacing), f"Time: {current_time}",
fill="WHITE", font=font_common)
                draw.text((20, start_y + 2 * line_spacing), f"Weekday:
{current_weekday}", fill="WHITE", font=font_common)

                image_rotated = image.rotate(180)
                disp.ShowImage(image_rotated)
                time.sleep(1)
```

```
            if os.path.exists(image_path_for_emoji2):
                image = Image.open(image_path_for_semoji)
                image = image.resize((disp.width, disp.height))
                disp.ShowImage(image.rotate(180))
            else:
                print(f"File {image_path_for_femoji} is not found")
            time.sleep(6)


            for img_name in ["rain.png", "snow.png", "storm.png", "sunny.png",
"cloudy.png"]:
                img_path = os.path.join(image_folder, img_name)
                if os.path.exists(img_path):
                    img = Image.open(img_path).resize((disp.width, disp.height))
                    img_rotated = img.rotate(180)
                    disp.ShowImage(img_rotated)
                    time.sleep(5)

            if os.path.exists(image_path_for_temoji):
                image = Image.open(image_path_for_temoji)
                image = image.resize((disp.width, disp.height))
                disp.ShowImage(image.rotate(180))
            time.sleep(6)

            forecast = get_tomorrow_forecast(data)

            display_forecast(disp, font_common, forecast)
```

```
        time.sleep(10)

        if os.path.exists(image_path_for_foemoji):
            image = Image.open(image_path_for_foemoji)
            image = image.resize((disp.width, disp.height))
            disp.ShowImage(image.rotate(180))
        time.sleep(6)

        if os.path.exists(image_path_for_lemoji):
            image = Image.open(image_path_for_lemoji)
            image = image.resize((disp.width, disp.height))
            disp.ShowImage(image.rotate(180))
        time.sleep(6)

except KeyboardInterrupt:
    disp.module_exit()
    logging.info("Program terminated")
```

Частина друга

```
import RPi.GPIO as GPIO
import time

servo_pin_1 = 3
servo_pin_2 = 2
servo_pin_3 = 26

GPIO.setwarnings(False)
```

```
GPIO.setmode(GPIO.BCM)
GPIO.setup(servo_pin_1, GPIO.OUT)
GPIO.setup(servo_pin_2, GPIO.OUT)
GPIO.setup(servo_pin_3, GPIO.OUT)

pwm1 = GPIO.PWM(servo_pin_1, 50)
pwm2 = GPIO.PWM(servo_pin_2, 50)
pwm3 = GPIO.PWM(servo_pin_3, 50)

pwm1.start(0)
pwm2.start(0)
pwm3.start(0)

def smooth_set_angle(pwm, pin, start_angle, end_angle, step=1, delay=0.02):
    start_duty = 2 + (start_angle / 18)
    end_duty = 2 + (end_angle / 18)

    if start_duty <= end_duty:
        duty_range = range(int(start_duty * 10), int(end_duty * 10) + 1, step)
    else:
        duty_range = range(int(start_duty * 10), int(end_duty * 10) - 1, -step)

    for duty in duty_range:
        GPIO.output(pin, True)
        pwm.ChangeDutyCycle(duty / 10)
        time.sleep(delay)
        GPIO.output(pin, False)
```

```
        pwm.ChangeDutyCycle(0)

    def set_fixed_angle(pwm, pin, angle):
        duty = 2 + (angle / 18)
        GPIO.output(pin, True)
        pwm.ChangeDutyCycle(duty)
        time.sleep(0.5)
        GPIO.output(pin, False)

    def first_try():
        set_fixed_angle(pwm1, servo_pin_1, 35)
        smooth_set_angle(pwm2, servo_pin_2, start_angle=5, end_angle=180, step=1,
delay=0.06)
        smooth_set_angle(pwm3, servo_pin_3, start_angle=180, end_angle=90, step=1,
delay=0.06)

    def second_try():
        set_fixed_angle(pwm2, servo_pin_2, 90)
        smooth_set_angle(pwm1, servo_pin_1, start_angle=25, end_angle=110, step=1,
delay=0.06)
        smooth_set_angle(pwm3, servo_pin_3, start_angle=110, end_angle=25, step=1,
delay=0.06)

    def third_try():
        set_fixed_angle(pwm2, servo_pin_2, 90)
        set_fixed_angle(pwm1, servo_pin_1, 35)
        smooth_set_angle(pwm2, servo_pin_2, start_angle=25, end_angle=170, step=1,
delay=0.06)
```

```
        smooth_set_angle(pwm3, servo_pin_3, start_angle=170, end_angle=15, step=1, delay=0.06)
        set_fixed_angle(pwm2, servo_pin_2, 5)

    try:
        while True:
            first_try()
            second_try()
            third_try()

    except KeyboardInterrupt:
        pwm1.stop()
        pwm2.stop()
        pwm3.stop()
        GPIO.cleanup()
```

ДОДАТОК Б

Демонстраційний матеріал у вигляді презентації

№	Позначення	Найменування	Дод. відомості
		Текстові документи	
1	ГЮИК. 505400.019 ПЗ	Пояснювальна записка	90 с., ф.А4
		Додаткові матеріали	
2		Лістинг коду	А4, 12 с.
3		Демонстраційний	А4, 12 с.
		матеріал у вигляді	*.ppt
		презентації	

					ГЮИК. 505400.019 ВД			
Змін.	Арк.	№ документа	Підп.	Дата.	Розробка системи автоматичного керування роботом телеприсутності «Відомість кваліфікаційної роботи»	Літ.	Аркуш.	Аркушів
Розроб.		Шматко						1
Перевірив.		Євсєєв				ХНУРЕ Кафедра КІТАР		
Н. контр.		Демська						
Затвер.		Невлюдов						