1.2

#include <esp\_sleep.h>

#include "SimulatedQMI8658.h" //include accelerometer class

#include <Adafruit\_Sensor.h>

//gives insights of wake up frequency

RTC\_DATA\_ATTR int counterwake=0;

RTC\_DATA\_ATTR time\_t wakeuptime;

const sleeptime=2\*60\*100000 //2 minutes

SimulatedQMI8658 simulateacc; //represent accelerometer

void setup() {

Serial.begin(9600); // Initialize simulated accelerometer

if (simulateacc.begin()) {

Serial.println("Simulated accelerometer initialized.");

}

//define wake motion or working

simulateacc.setWakeupMotionEventCallBack(working);

simulateacc.configWakeOnMotion();

// handle motion event and its working

void working() {

Serial.println("Motion detected, wake up!");

}

// Going deep sleep

Serial.println("Going deep sleep");

esp\_deep\_sleep\_start();

}

void loop() {

Serial.print("Wake-up count: ");

Serial.println(counterWake);

counterWake++; // Increment wake-up counter

// Simulate checking for motion every loop iteration

simulateacc.simulateMotion();

}

1.3

#include <WiFi.h>

#include "TimeManager.h" // Include the TimeManager class

TimeManager timemanage;//creating object of time-related functions from TimeManager and that object name is timemanage

void setup() {

Serial.begin(9600);

// Connecting to Wi-Fi for NTP connection

WiFi.begin("SSID", "PASSWORD");

if (WiFi.status() != WL\_CONNECTED) {

Serial.println(" Wi-Fi connecting”);

}

delay(1000);

Serial.println(" Wi-Fi connected");

// starting time manage. Initializing RTC and setup NTP configuration

Timemanage.begin();

// Synchronize this time with the NTP server if Wi-Fi is available.

timemanage.syncTimeWithNTP();

// Printing of the synchronized time

timemanage.printCurrentTime();

}

void loop()

{

Timemanage.printCurrentTime(); // print time to monitor screen

delay(2000);

}

1.4

def activity(accelerate\_magnitude):

if acceleration\_magnitude < 0.5:

return "Resting"

else if 0.5 <= acceleration\_magnitude < 1.5:

return "Walking"

else if 1.5 <= acceleration\_magnitude < 3.0:

return "Running"

else:

return "Playing"

1.5

//Start time of the 10-minute interval using LittleFS

{

"timestamp": "YYYY-MM-DD HH:MM:SS",

"rest\_time": <seconds>,

"walk\_time": <seconds>,

"run\_time": <seconds>,

"play\_time": <seconds>

}

//Setting up LittleFS for microcontroller.

#include <LittleFS.h>

void setup() {

if (!LittleFS.begin()) {

Serial.println("Fail to mount file system");

return;

}

Serial.println("File system mounted");

}