```
#include <Wire.h>
#include <Adafruit_MPU6050.h>
#include <Adafruit_Sensor.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>
#include "DFRobotDFPlayerMini.h"

// OLED Display
#define SCREEN_WIDTH 128
#define SCREEN_HEIGHT 64
Adafruit_SSD1306 display(SCREEN_WIDTH, SCREEN_HEIGHT, &Wire, -1);

// MPU6050 Accelerometer
Adafruit_MPU6050 mpu;

// MP3 Player
DFRobotDFPlayerMini myDFPlayer;

// System Variables
int currentSong = 0;
unsigned long lastSongChange = 0;
int displayScreen = 0;
unsigned long lastScreenChange = 0;
float acceleration = 0;
String activity = "Ready";
int simulatedHR = 75;
unsigned long lastHRUpdate = 0;
bool mp3Ready = false;
bool mpuReady = false;

void setup() {
Serial.begin(115200);
Serial.println("=== MUSIC TO FITNESS SYSTEM ===");

// Initialize OLED
display.begin(SSD1306_SWITCHCAPVCC, 0x3C);
display.setTextSize(1);
display.setTextColor(WHITE);
Serial.println("OLED: READY");

// Initialize MPU6050
if(mpu.begin()) {
mpu.setAccelerometerRange(MPU6050_RANGE_8_G);
mpu.setGyroRange(MPU6050_RANGE_500_DEG);
mpu.setFilterBandwidth(MPU6050_BAND_21_HZ);
mpuReady = true;
Serial.println("MPU6050: READY");
} else {
Serial.println("MPU6050: NOT FOUND");
}

// Initialize MP3 Player
Serial2.begin(9600, SERIAL_8N1, 16, 17);
if(myDFPlayer.begin(Serial2)) {
```

```
myDFPlayer.volume(22); // 0-30
myDFPlayer.play(1); // Start with calm music
currentSong = 1;
mp3Ready = true;
Serial.println("MP3: READY - Playing Song 1");
} else {
Serial.println("MP3: NOT CONNECTED");
}

// Show startup sequence
startupDisplay();
}

void startupDisplay() {
display.clearDisplay();
display.setCursor(0,0);
display.println("MUSIC TO FITNESS");
display.println("================");
display.println("Initializing...");
display.display();
delay(1000);

display.clearDisplay();
display.setCursor(0,0);
display.println("SYSTEM CHECK");
display.println("============");
display.print("OLED: ");
display.println("OK ✓");
display.print("MP3: ");
display.println(mp3Ready ? "OK ✓" : "FAIL ✗");
display.print("MPU6050: ");
display.println(mpuReady ? "OK ✓" : "FAIL ✗");
display.println("============");
display.println("Starting in 2s...");
display.display();
delay(2000);
}

String detectActivity() {
if(!mpuReady) return "DEMO MODE";

sensors_event_t a, g, temp;
mpu.getEvent(&a, &g, &temp);

acceleration = sqrt(a.acceleration.x*a.acceleration.x +
a.acceleration.y*a.acceleration.y +
a.acceleration.z*a.acceleration.z);

if(acceleration > 15.0) return "RUNNING";
else if(acceleration > 11.0) return "WALKING";
else if(acceleration > 9.8) return "MOVING";
else return "STANDING";
}

void updateSimulatedHR() {
if(millis() - lastHRUpdate > 7000) {
// Simulate different HR based on activity
```

```
if(activity == "RUNNING") simulatedHR = 95 + random(0, 15);
else if(activity == "WALKING") simulatedHR = 78 + random(0, 10);
else if(activity == "MOVING") simulatedHR = 75 + random(0, 8);
else simulatedHR = 68 + random(0, 7);

lastHRUpdate = millis();

// Update music based on simulated HR
int newSong = 0;
if(simulatedHR < 75) newSong = 1; // Calm
else if(simulatedHR < 90) newSong = 2; // Normal
else newSong = 3; // High intensity

if(newSong != currentSong && mp3Ready) {
myDFPlayer.play(newSong);
currentSong = newSong;
lastSongChange = millis();
}
}
}

String getSongName(int songNum) {
switch(songNum) {
case 1: return "Calm Music";
case 2: return "Normal Beats";
case 3: return "High Intensity";
default: return "None";
}
}

void loop() {
// Update activity detection
activity = detectActivity();

// Update simulated heart rate
updateSimulatedHR();

// Change display screen every 4 seconds
if(millis() - lastScreenChange > 4000) {
displayScreen = (displayScreen + 1) % 4;
lastScreenChange = millis();
}

// Update display
display.clearDisplay();
display.setCursor(0,0);

switch(displayScreen) {
case 0: // Main Dashboard
display.println("MUSIC TO FITNESS");
display.println("================");
display.print("Activity: ");
display.println(activity);
display.print("HR: ");
display.print(simulatedHR);
display.println(" BPM");
display.print("Music: ");
```

```
display.println(getSongName(currentSong));
display.println("================");
display.print("Accel: ");
display.print(acceleration, 1);
display.println(" m/s²");
break;

case 1: // Music Player Screen
display.println("MUSIC PLAYER");
display.println("============");
display.println("Now Playing:");
display.println("");
if(currentSong == 1) {
display.println(" Song 1: Calm");
display.println(" Relaxing tempo");
display.println(" For: Resting");
} else if(currentSong == 2) {
display.println(" Song 2: Normal");
display.println(" Moderate beats");
display.println(" For: Daily activity");
} else {
display.println(" Song 3: High");
display.println(" Fast tempo");
display.println(" For: Exercise");
}
break;

case 2: // Fitness Data
display.println("FITNESS DATA");
display.println("============");
display.print("Heart Rate: ");
display.print(simulatedHR);
display.println(" BPM");
display.print("Zone: ");
if(simulatedHR < 70) display.println("Resting");
else if(simulatedHR < 85) display.println("Fat Burn");
else if(simulatedHR < 100) display.println("Cardio");
else display.println("Peak");
display.println("");
display.print("Activity Level: ");
display.println(activity);
display.print("Acceleration: ");
display.print(acceleration, 1);
break;

case 3: // System Status
display.println("SYSTEM STATUS");
display.println("=============");
display.print("OLED: ");
display.println("OK ✓");
display.print("MP3: ");
display.println(mp3Ready ? "OK ✓" : "OFF ✗");
display.print("Accelerometer: ");
display.println(mpuReady ? "OK ✓" : "OFF ✗");
display.println("");
display.print("Uptime: ");
```

```
display.print(millis() / 1000);
display.println("s");
display.print("Next update: ");
display.print((4000 - (millis() - lastScreenChange)) / 1000);
display.println("s");
break;
}

display.display();

// Serial output for monitoring
static unsigned long lastSerial = 0;
if(millis() - lastSerial > 2000) {
Serial.print("Activity: ");
Serial.print(activity);
Serial.print(" | HR: ");
Serial.print(simulatedHR);
Serial.print(" BPM | Music: ");
Serial.println(getSongName(currentSong));
lastSerial = millis();
}

delay(100);
}
```