REPORT 3 : SERIAL AND USB INTERFACING WITH MICROCONTROLLER AND COMPUTER BASED SYSTEM (2): SENSORS AND ACTUATORS)

GROUP 4

MCTA 3203

SEMESTER 2 2024/2025

MECHATRONICS SYSTEM INTEGRATION

DATE OF SUBMISSION: 23 MARCH 2025

NO	NAME	MATRIC NUMBER
1.	IRDINA NABIHAH BINTI MOHD NAZRI	2214772
2.	IZZAH ZAHIRA BINTI NORAZLEE	2217696
3.	NOR MAISARAH BINTI ISMAIL	2213080

TABLE OF CONTENTS

INTRODUCTION

In this week's experiment, it solely focuses on implementing serial communication between a microcontroller (Arduino) and a computer-based system to interface with sensors and actuators. Two key components are involved: the MPU6050 Inertial Measurement Unit (IMU) and an RFID card reader. Meanwhile, the MPU6050 sensor, which integrates an accelerometer and a gyroscope, enables the detection of motion and orientation data. The RFID reader is used for card-based authentication. By connecting these components via USB and using Arduino and Python programming, the experiment demonstrates how real-time data can be transmitted, processed, and utilized effectively.

To be more exact, the objectives of this experiment are to create a basic hand gesture recognition system using motion data from the MPU6050 sensor and to control a servo motor based on RFID card authentication. The Arduino is programmed to read data from the IMU and transmit it to the computer via serial communication, where a Python script interprets the data and recognizes specific gestures. In the RFID setup, the Python script identifies the scanned card and determines whether access should be granted. But based on this decision, control signals are sent to the servo motor and corresponding LEDs to reflect the outcome.

Moreover, it is expected that this setup will accurately recognize predefined gestures and distinguish between authorized and unauthorized RFID cards. When a recognized gesture is performed or a valid card is detected, appropriate responses such as printing data, rotating a servo motor, or illuminating LEDs are anticipated. This experiment highlights the practical application of serial communication in integrating sensors, actuators, and software to build interactive and responsive systems.

ABSTRACT

This experiment explores the use of serial communication to interface a microcontroller with sensors and actuators for real-time data acquisition and control. The primary objectives were to implement a hand gesture recognition system using the MPU6050 Inertial Measurement Unit (IMU) and to develop an RFID-based authentication system to control a servo motor. The MPU6050 sensor was connected to an Arduino to collect accelerometer and gyroscope data, which was then transmitted to a computer via USB. A Python script processed this data

to identify specific gestures. In the second setup, an RFID reader was used to scan card IDs, and Python was employed to determine access authorization, which then controlled the servo motor and LED indicators accordingly.

Furthermore, the experiment that we did successfully demonstrated reliable communication between hardware and software components. The system was able to accurately detect predefined hand gestures and differentiate between authorized and unauthorized RFID cards. These results confirm the effectiveness of using serial communication for integrating sensor inputs and actuator outputs in interactive applications. The project highlights the potential of combining microcontrollers, sensors, and software for building responsive embedded systems.

EXPERIMENT 4A

MATERIALS AND EQUIPMENTS

- Arduino Board
- MPU6050 Sensor Module
- USB Cable
- Jumper Wires
- LEDs
- Arduino IDE
- Python with PySerial Library

EXPERIMENTAL AND SETUP

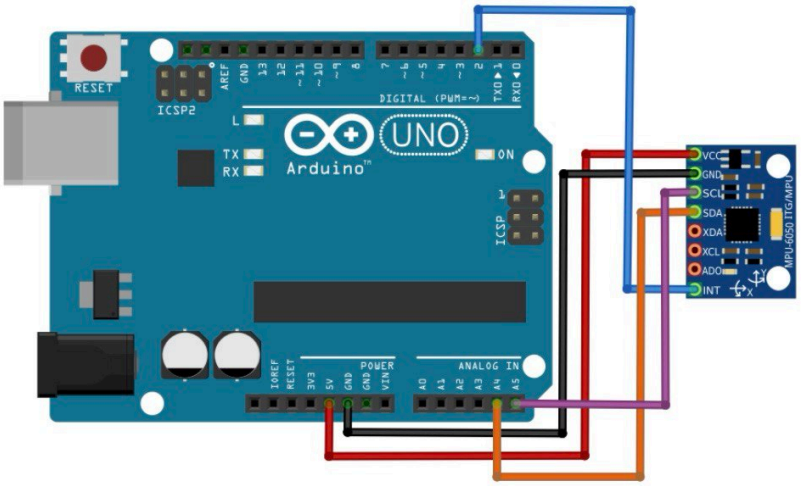


Fig. 1: Arduino-MPU6050 Connections

1. Connect the MPU6050 sensor to the Arduino board using the appropriate pins. The MPU6050 typically uses I2C communication, so connect the SDA and SCL pins of the MPU6050 to the corresponding pins on the Arduino (usually A4 and A5 for most Arduino boards).

2. Connect the power supply and ground of the MPU6050 to the Arduino's 5V and GND pins.

3. Ensure that the Arduino board is connected to your PC via USB.

METHODOLOGY

The main idea about this experiment was to establish serial communication between a microcontroller and a computer system using the MPU6050 IMU sensor for gesture recognition. The following steps and procedures were followed:

1. Materials Preparation

   The following components and software were prepared before starting the experiment:

   • Arduino Uno board
   • MPU6050 sensor module
   • Jumper wires and breadboard
   • USB cable for Arduino-PC connection
   • LEDs (for optional visual output)
   • Computer with Arduino IDE and Python installed
   • Python libraries: pyserial

2. Hardware Setup

   • The MPU6050 sensor was connected to the Arduino via I2C interface:
   • SDA to Arduino A4, SCL to A5
   • VCC to 5V, GND to GND
   • The Arduino was connected to the PC using a USB cable for programming and data transmission.

3. Arduino Programming

   • An Arduino sketch was written to initialize and read data from the MPU6050 using the MPU6050.h library.
   • The sketch read accelerometer and gyroscope values using getMotion6() and transmitted them via the serial port at 9600 baud.
   • An enhanced version of the sketch included gesture recognition logic, where sensor thresholds were used to identify specific hand gestures and transmit corresponding messages like "Detected Gesture: Gesture 1".

4. Python Programming on PC

• A Python script was developed using the pyserial library to establish serial communication with the Arduino.

• The script read and decoded the incoming data in real-time.

• When gesture messages were received, specific actions (e.g., print statements or triggering logic) were executed based on the detected gesture.

5. Testing and Visualisation

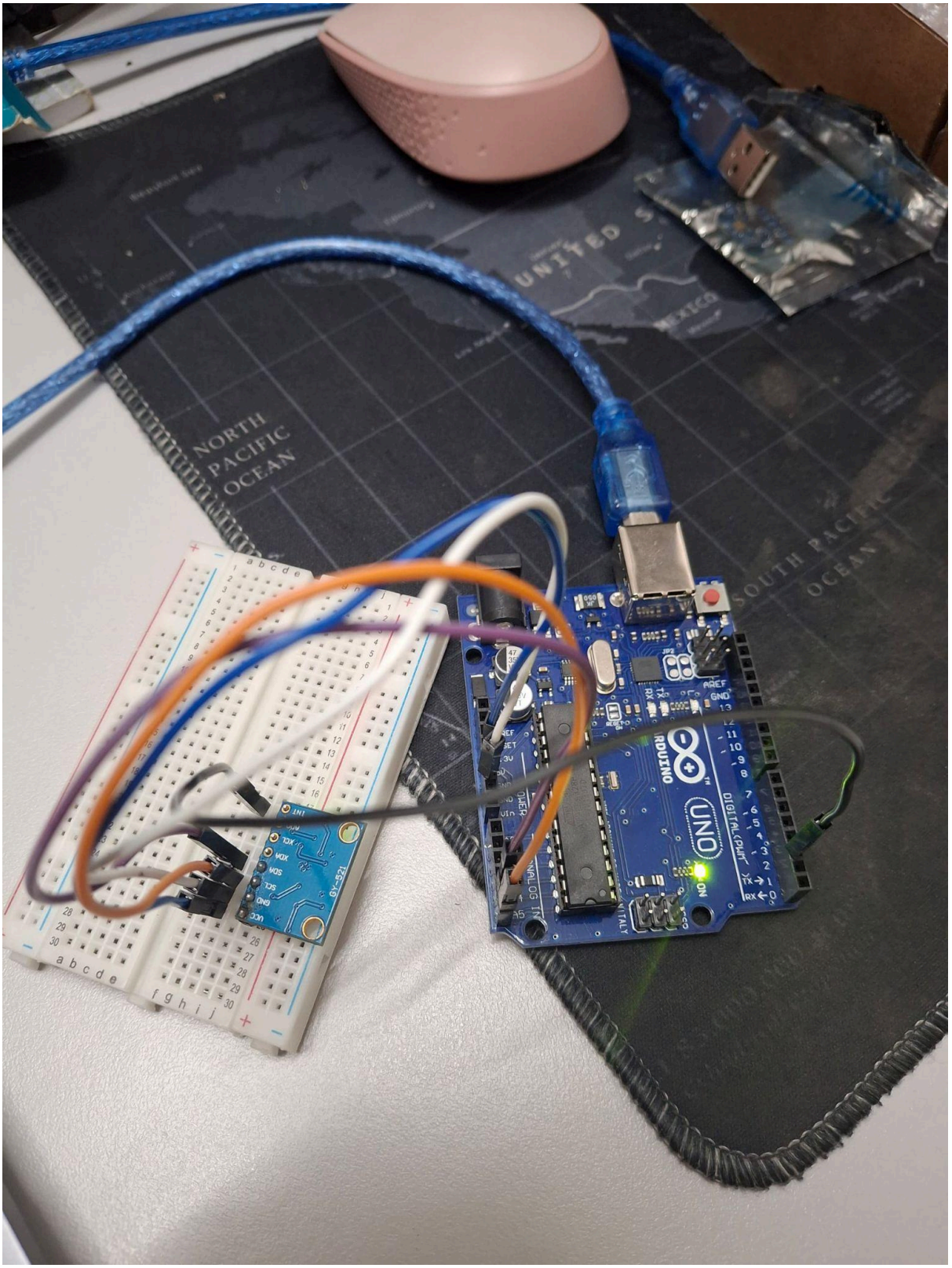
• The Arduino code was uploaded via Arduino IDE.

• The Python script was run on the PC, and the serial output was observed.

• Gestures were performed by moving the MPU6050 sensor, and their detection was verified through the Python console.

• Optionally, motion paths could be visualized using the x and y components of the accelerometer data.

RESULTS



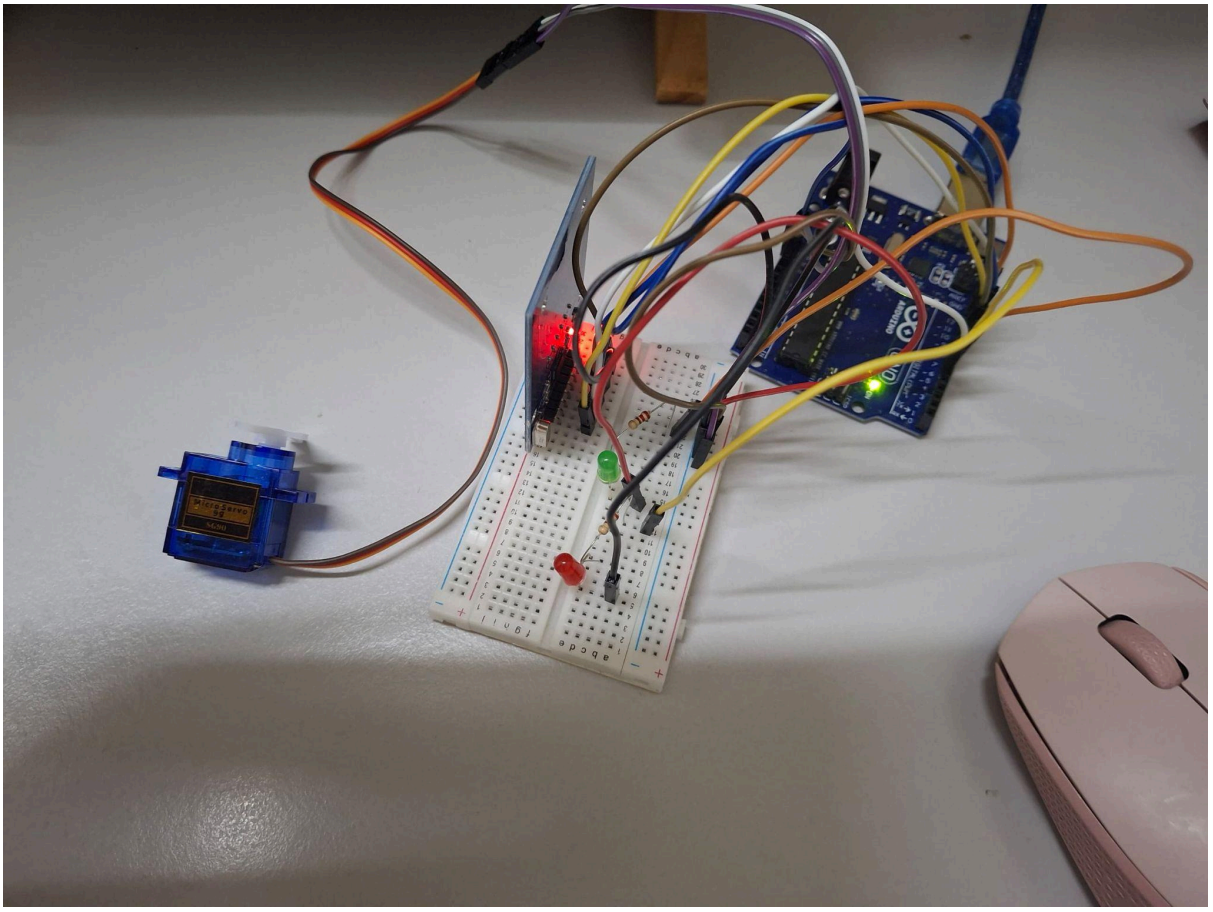
Video link: https://github.com/irdinazri/MSI_G4/blob/main/Week%204/4A/4A.mp4

EXPERIMENT 4B

MATERIALS AND EQUIPMENTS

- Arduino Board
- RFID Card Reader (USB connectivity)
- RFID Tags or Cards
- Servo Motor
- Jumper Wires
- Breadboard
- LEDs (Various Colours)
- USB Cables
- Computer with Software Installed
  - • Arduino IDE
  - • Python with pyusb library
- Datasheets and Manuals
- Power Supply (Optional)
- Mounting Hardware for Servo

EXPERIMENTAL SETUP



METHODOLOGY

1. Hardware Setup:

   The setup began with the wiring of a servo motor to the Arduino. The servo motor's power (red) and ground (brown or black) wires were connected to the Arduino's 5V and GND pins respectively, and the signal wire (usually yellow or orange) was connected to one of the Arduino's PWM digital pins (e.g., D9). An RFID card reader with USB connectivity was connected directly to the computer, which provided both power and a communication interface.

   LEDs were also integrated into the setup, with a green LED used to indicate successful card authentication (access granted) and a red LED to indicate failed authentication (access denied). These LEDs were connected through digital output pins on the Arduino.

2. Arduino Programming:

The Arduino was programmed to receive commands from the computer via the serial port. Based on the command received, the Arduino controlled the servo motor and activated the appropriate LED. Two commands were defined: one for granting access (which moved the servo to a certain angle and lit the green LED) and one for denying access (which reset the servo and lit the red LED).

3. Python-Based Control:
   A Python script was developed to run on the computer, which handled communication with the USB-connected RFID reader. Using a library for USB HID communication (such as pyusb), the script continuously read card IDs scanned by the RFID reader. A list of authorized card IDs was defined in the script.

   When a card was scanned, its ID was compared against the authorized list. If the ID matched, the script sent a command (e.g., 'A') to the Arduino via serial communication to activate the servo motor and green LED. If the card was not recognized, a different command (e.g., 'D') was sent to deactivate the servo and light the red LED instead.

4. Control Algorithm Logic:
   Access Granted: If the card ID matches the authorized list, the servo motor rotates to a specific angle (e.g., 180°) to represent unlocking, and the green LED is turned on.

   Access Denied: If the card ID is not in the list, the servo motor returns to the default position (e.g., 90°) to represent locked state, and the red LED is activated.

5. Optional Enhancements: Additional features included integrating JSON data structure for card management, allowing dynamic updates to the list of authorized cards. The system could also be extended to let users set the servo position through the Python interface.

RESULTS

```
#include <SPI.h>
#include <MFRC522.h>
#include <Servo.h>

#define SS_PIN 10
#define RST_PIN 9
#define GREEN_LED 7
#define RED_LED 6
#define SERVO_PIN 5

MFRC522 rfid(SS_PIN, RST_PIN);
Servo servo;

String authorizedUIDs[] = {"123456789", "987654321"};  // Replace with actual RFID UIDs
bool isAuthorized = false;

void setup() {
    Serial.begin(9600);
    SPI.begin();
    rfid.PCD_Init();
    pinMode(GREEN_LED, OUTPUT);
    pinMode(RED_LED, OUTPUT);
    servo.attach(SERVO_PIN);
    servo.write(90);  // Default position
}
```

```
void loop() {
    if (!rfid.PICC_IsNewCardPresent() || !rfid.PICC_ReadCardSerial()) {
        return;
    }

    String uid = "";
    for (byte i = 0; i < rfid.uid.size; i++) {
        uid += String(rfid.uid.uidByte[i], DEC);
    }

    Serial.println(uid);  // Send UID to Python
    isAuthorized = false;

    for (String authUID : authorizedUIDs) {
        if (uid == authUID) {
            isAuthorized = true;
            break;
        }
    }
```

```
    if (isAuthorized) {
        digitalWrite(GREEN_LED, HIGH);
        digitalWrite(RED_LED, LOW);
    } else {
        digitalWrite(RED_LED, HIGH);
        digitalWrite(GREEN_LED, LOW);
    }

    delay(2000);
    digitalWrite(GREEN_LED, LOW);
    digitalWrite(RED_LED, LOW);

    rfid.PICC_HaltA();
    rfid.PCD_StopCrypto1();
}
```

An RFID reader and a servo motor are used to control access based on authorization on the card. An MFRC522 library manages RFID functionality, and a Servo library manages the motor as a gate or door lock. The system verifies the UID of the card against a pre-programmed list of accepted UIDs..

In the `setup()`, RFID reader, servo motor, and LED pins (green for granted access, red for denied) are initialized. In the `loop()`, upon finding a card, the `isAuthorizedCard()` function checks if the UID is any of those in the authorizedCards list.

DISCUSSION

This experiment highlighted two main applications: access control using RFID-based with the MFRC522 module and a servo motor, and gesture detection via the MPU6050 sensor. The MPU6050 with its accelerometer and gyroscope was able to detect forward and backward tilt gestures by monitoring acceleration thresholds. Video analysis confirmed that the system could easily accomplish simple gesture-controlled functionality as the gestures were consistently detected and reflected on the serial monitor. These systems have the potential to be applied in robotics and other fields where simple motion-based commands can ease interaction.

The RFID access control system could distinguish between authorized and unauthorized cards consistently. Upon detection of an authorized card, the servo motor would open, while an unauthorized attempt would be indicated by a red LED. This serves as evidence of the capability of the system for secure access control in restricted areas like laboratories or offices. There were, however, differences between expected and actual performance. There were occasional false negatives or positives, usually as a result of minor card misalignments or environmental vibrations. These findings underscore the importance of MPU6050 alignment and stabilization for greater gesture accuracy. Likewise, variable RFID reads due to probable card orientation or environmental interference point toward improvements.

The gesture detection accuracy was also affected by power noise and power fluctuations, and the fixed threshold sensitivity may not suit all environments. MPU6050 calibration and the setting of the thresholds based on specific application conditions would improve the performance. Intermittent failure of the servos caused by power instability occurred with the RFID system. Enhancements such as the addition of a dynamic database for the processing of approved cards and position feedback for the servo would add scalability and responsiveness.

In summary, both systems worked but were restricted. Gesture recognition was restricted to two gestures and was prone to false positives from interference from the environment. The RFID system was unscalable since it contained a static list of permitted cards and had a fixed servo operation time. Setting aside these restrictions, the experiment was able to demonstrate elementary concepts of secure access control and gesture recognition, both of which hold potential for broader usage in robotics and security industries with further development.

CONCLUSION

The experiments showed that serial communication between an Arduino microcontroller and a computer system to communicate with several sensors and actuators could be implemented successfully. The first experiment used the MPU6050 IMU sensor to record gyroscope and accelerometer data, allowing for the identification of simple hand gestures. In the second experiment, which concentrated on RFID-based authentication, a servo motor was controlled by authorised RFID cards using signals that were analysed by Python. The results in both instances validated the original theories by demonstrating that motion and identification information could be efficiently recorded, sent, and utilised to elicit physical reactions in real time.

These results demonstrate the usefulness of sensor integration in embedded systems, especially for access control and human-machine interaction. Applications in automation, robotics, assistive technology, and smart environments become possible when low-cost, widely accessible components can be used to recognise hand gestures or authenticate users. These studies provide a solid basis for more complex implementations that include wireless connectivity, data visualisation, and AI or machine learning-based intelligent decision-making.

RECOMMENDATION

The performance and the scalability of this experiment can be improved by calibrating the MPU6050 IMU sensor to increase the accuracy of gesture recognition by fine tuning the sensitivity thresholds to better adapt to different environmental conditions and reduce false positives. Without the calibration, the system might misinterpret minor intended movements as valid gestures, leading to false positives or missed detections. The experiment also can be improved by using a stable power supply. It is used to minimize power fluctuations since it can affect servo motor behavior and sensor readings.

For the RFID system, improvement can be made by employing a higher-sensitivity RFID reader. This can help the reader to address occasional errors caused by card misalignment or environmental interference since the RFID are sensitive to the alignment and distance between the card and the reader. These recommendations aim to enhance the system’s reliability, usability and applicability in real world scenarios.

ACKNOWLEDGEMENTS

A special thanks goes out to Dr. Wahju Sediono and Dr. Zulkifli Bin Zainal Abidin, my teaching assistant, and my peers for their invaluable help and support in finishing this report. Their advice, feedback, and experience have greatly influenced the level of quality and understanding of this work. Their time, patience, and commitment to supporting my academic success are greatly appreciated.

STUDENT'S DECLARATION

Certificate of Originality and Authenticity

This is to certify that we are **responsible** for the work submitted in this report, that the original work is our own except as specified in the references and acknowledgement, and that the original work contained herein have not been untaken or done by unspecified sources or persons.

We hereby certify that this report has **not been done by only one individual** and **all of us have contributed to the report**. The length of contribution to the reports by each individual is noted within this certificate.

We also hereby certify that we have **read** and **understand** the content of the total report and no further improvement on the reports is needed from any of the individual's contributors to the report.

We therefore, agreed unanimously that this report shall be submitted for **marking** and this **final printed report** has been **verified by us**.

Signature: 

Name: IRDINA NABIHAH BINTI MOHD NAZRI

Matric Number: 2214772

Read ☑
Understand ☑
Agree ☑

Signature: 

Name: IZZAH ZAHIRA BINTI NORAZLEE

Matric Number: 2217696

Read ☑
Understand ☑
Agree ☑

Signature: 

Name: NOR MAISARAH BINTI ISMAIL

Matric Number: 2213080

Read ☑

Understand ☑

Agree ☑