MECHATRONICS SYSTEM INTEGRATION

MCTA 3203

WEEK 4:

SERIAL INTERFACING WITH MICROCONTROLLER

SECTION 1

SEMESTER 1, 2025/20256

INSTRUCTOR:

ASSOC. PROF. EUR. ING. IR. TS. GS. INV. DR. ZULKIFLI BIN ZAINAL ABIDIN

DR. WAHJU SEDIONO

DATE OF SUBMISSION: 4TH NOVEMBER 2025

GROUP 8

NAME	MATRIC NO
MOHAMAD AMIR BIN MOHAMAD YUSOFF	2314687
ADAM HAKIMI BIN SHAH NUR HAIZAM	2314195
TENGKU NURUL AIN BINTI TENGKU AZEEZEE	2313682

Table Of Contents

Abstract

This experiment focuses on the development of a real-time motion tracking and authentication system using an MPU6050 motion sensor and an RFID module interfaced with an Arduino Uno. The objective is to acquire, process, and analyze accelerometer and gyroscope data to detect simple motion patterns such as circular motion. The MPU6050 provides six degrees of freedom for measuring acceleration and angular velocity, while the RFID module enables secure user identification. Data collected from the Arduino are transmitted to a computer and processed in Python, which visualizes sensor readings and enables bidirectional communication for control commands. A servo motor is integrated as an actuator, responding to both motion data and RFID verification for access control. This experiment demonstrates the integration of sensor interfacing, data communication, and control mechanisms, reflecting the practical application of mechatronic principles in intelligent access and motion detection systems.

Introduction

The integration of sensors and microcontrollers forms the foundation of modern mechatronic systems. Among various sensing technologies, inertial measurement units (IMUs) such as the MPU6050 play a significant role in motion detection, gesture recognition, and orientation tracking. The MPU6050 combines a 3-axis accelerometer and a 3-axis gyroscope, allowing it to capture linear acceleration and angular velocity in real time. In this experiment, the MPU6050 is interfaced with an Arduino Uno to measure motion data, while an RFID module is incorporated to enable user authentication.

The Arduino serves as the central controller, managing data acquisition and communication with the computer. Using Python, the real-time accelerometer and gyroscope data are processed, analyzed, and visualized, enabling the identification of basic motion patterns such as circular movements. Additionally, the system integrates a servo motor that operates based on both motion detection and RFID verification, forming a simple prototype of a secure access control mechanism.

Through this experiment, students gain hands-on experience in sensor integration, serial communication, Python-based data processing, and hardware control logic. The project emphasizes the importance of combining sensing, computation, and actuation to develop intelligent mechatronic systems capable of responding dynamically to environmental inputs.

TASK 1

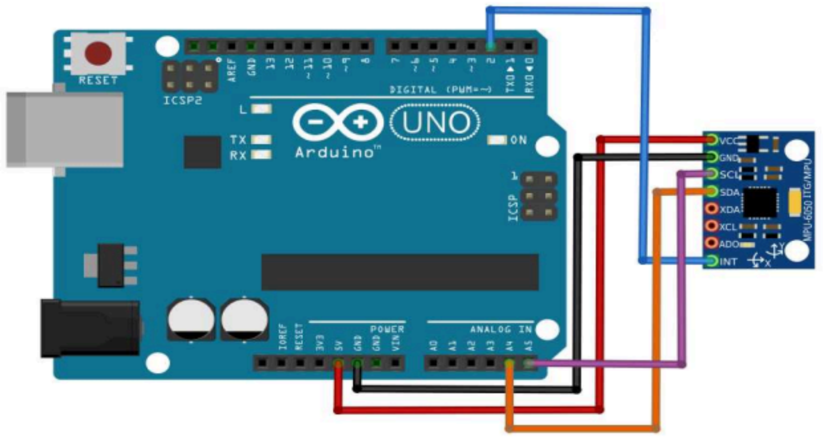
Materials and Equipment

1. Arduino board
2. MPU6050 sensor
3. Jumper wires/breadboard
4. USB cable

Experimental Setup

1. The experimental setup involved interfacing the MPU6050 motion sensor with an Arduino Uno microcontroller to measure and analyze real-time motion data. The MPU6050 allows the detection of both linear acceleration and angular velocity.
2. The VCC pin of the MPU6050 was connected to the Arduino's 5V supply, while the GND pin was connected to the GND terminal to provide a common electrical reference.
3. The SCL and SDA pins were connected to the Arduino's A5 and A4 pins, respectively, for clock and data transmission.
4. The INT pin of the MPU6050 was connected to digital pin 2 on the Arduino to enable interrupt-based data updates.
5. The Arduino Uno was connected to a computer via a USB cable, which supplied power and enabled serial communication.
6. Once the program was uploaded, the Arduino initialized the sensor, established I^2C communication, and continuously read motion data from the MPU6050's internal registers.

7. All components shared a common ground to ensure signal stability and prevent voltage inconsistencies.
8. After verifying the wiring and code functionality, the MPU6050 was gently moved and tilted to observe corresponding variations in acceleration and angular velocity readings, confirming the correct operation of the real-time motion tracking system.



Methodology

1. Overview

   This experiment involved using an MPU6050 sensor to capture real-time motion data, including acceleration and angular velocity, and sending the readings to a computer via serial communication. The Arduino Uno acted as the main controller, establishing an I²C connection with the MPU6050 to read sensor outputs continuously. The acquired data were displayed on the Arduino Serial Monitor for observation and analysis.

2. Hardware Setup

1. Arduino Uno	Serves as the microcontroller to read sensor data and communicate with the computer.
2. MPU6050 Sensor Module	Combines a 3-axis accelerometer and a 3-axis gyroscope to measure motion and orientation.
3. Jumper Wires	Connect the sensor to the Arduino's pins for power and data transfer.
4. Breadboard	Provides a base for connecting and organizing the components.
5. USB Cable	Supplies power to the Arduino and enables serial data communication with the computer.

3. Circuit Design

The VCC pin of the MPU6050 was connected to the 5V output of the Arduino, and the GND pin was connected to GND to complete the circuit. The SCL and SDA pins of the MPU6050 were connected to the Arduino's A5 and A4 pins, respectively, for I^{2}C communication. The INT pin was connected to digital pin 2 to allow interrupt-based data acquisition.The Arduino Uno was powered through the USB cable, which also established serial communication with the computer for data transmission and monitoring.

4. Program Development

The program consisted of two integrated parts: the Arduino code responsible for data acquisition and transmission, and the Python script responsible for data visualization and gesture detection.

The process flow was as follows:

1. The Arduino continuously reads accelerometer data (X and Y axes) from the MPU6050 sensor. The readings were then transmitted to the computer via serial communication.
2. On the computer side, a Python script received this serial data and plotted it dynamically using the *matplotlib* library, allowing real-time visualization of the sensor's motion pattern.
3. The Python program continuously analyzed the incoming accelerometer values to detect circular motion gestures. By evaluating the changes in X and Y coordinates over time, the program determined whether the movement formed a circular trajectory. If a valid circular motion was detected, the system highlighted the detection event within the plot and optionally displayed a notification message.

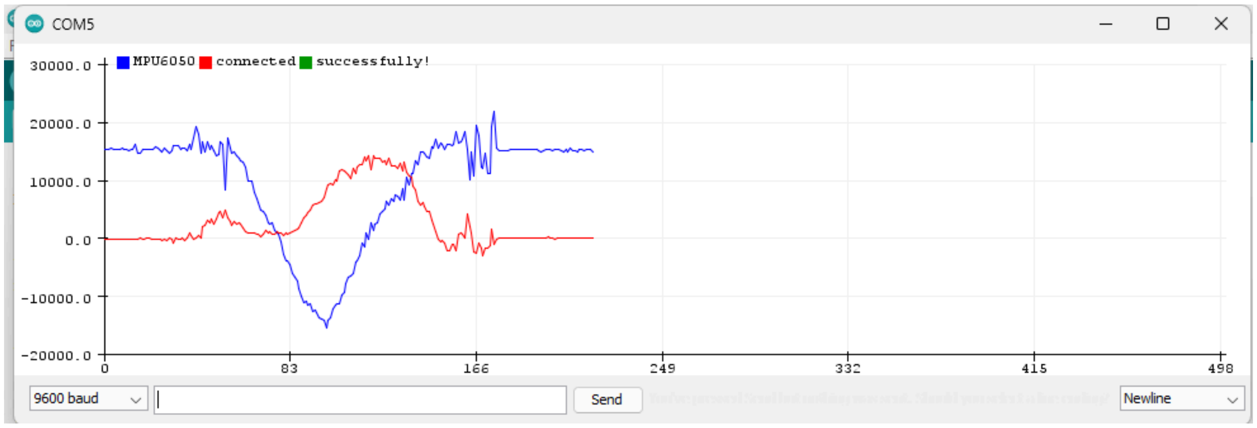
This integrated program combines embedded sensing with real-time data visualization and pattern recognition. It demonstrates how sensor data can be used not only for motion tracking but also for gesture-based control and intelligent movement analysis in mechatronic systems.

Data Collection.

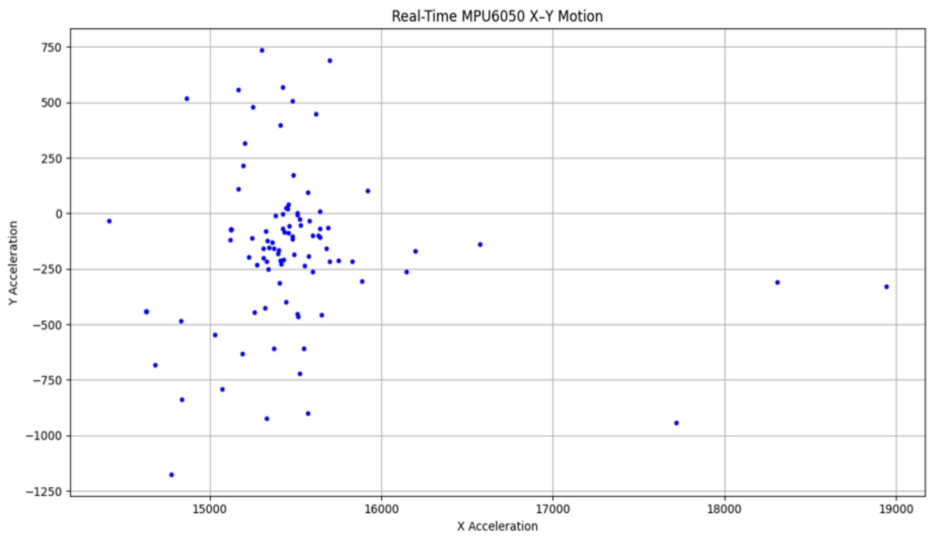
In this task, the MPU6050 was interfaced with an Arduino Uno to capture real-time motion data. The sensor outputs six parameters three for acceleration (Ax, Ay, Az) and three for angular velocity (Gx, Gy, Gz). The Arduino program continuously reads these values and sends them through a serial connection at 9600 baud.

Data was transmitted via the USB serial port to a Python script that used the pyserial library for communication. Only the X and Y acceleration data were extracted and plotted in real time using matplotlib to visualize motion patterns.

Graph produced by serial plotter in Arduino :



Graph produced by matplotlib in PyCharm:



Data Analysis

The collected data confirms that:

- The MPU6050 sensor accurately captured real-time accelerometer readings along the X and Y axes, showing consistent response to hand movements.
- The plotted data in Python (matplotlib) clearly reflected the user's motion pattern, allowing easy visualization of direction and shape.
- Circular motion gestures were distinguishable from linear or random movements through the smooth and repetitive pattern observed in the X–Y graph.
- The real-time data stream between Arduino and Python was stable, with minimal delay or missing data points, ensuring reliable dynamic plotting.

Analysis:

The results verify that the MPU6050 sensor was successfully interfaced with Arduino and Python for continuous motion tracking. The clear graphical representation confirmed accurate data transmission and effective visualization. The system demonstrated the ability to differentiate circular gestures from other motion types, fulfilling the experiment's main objectives. Overall, the collected data shows that the real-time motion tracking setup operated effectively and provided a solid foundation for gesture-based control applications.

Results

The experimental results confirm that the real-time motion tracking system using the MPU6050 sensor operated successfully and met its objectives. The sensor captured continuous accelerometer readings along the X and Y axes and transmitted the data to Python via serial communication for dynamic plotting.

- The plotted graph in *matplotlib* clearly reflected hand movements in real time. When the sensor was moved in a circular path, the X–Y plot displayed a smooth, rounded trajectory, accurately representing the circular motion gesture. In contrast, linear or irregular hand movements produced distinctly different graph patterns, confirming that gesture types could be visually distinguished.
- The serial communication between Arduino and Python remained stable throughout testing, with no significant delay or data loss. The live plot updated continuously, ensuring smooth motion visualization and responsive data capture.

Overall, the system achieved reliable real-time motion tracking and accurate graphical representation of accelerometer data. The successful detection of circular gestures demonstrates that the MPU6050 sensor and Python interface functioned effectively together, fulfilling the experiment’s objectives and validating the feasibility of using motion sensing for gesture-based applications.

TASK 2

Materials and Equipment

1. Arduino board
2. RFID reader (MFRC522) + RFID tags/cards
3. Servo motor
4. Red & Green LEDs
5. Resistors
6. MPU6050 sensor
7. Jumper wires and breadboard

Experimental Setup

1. The RFID module (MFRC522) was connected to the Arduino via the SPI interface, with the SS pin connected to digital pin 10, and the RST pin to digital pin 8.
2. The MPU6050 sensor was interfaced using the I^{2}C protocol, where the SCL and SDA pins were connected to A5 and A4, respectively.
3. The servo signal wire was connected to digital pin 9
4. The green and red LEDs were connected to pins D4 and D3 through appropriate current-limiting resistors.
5. All components shared a common GND and 5V supply from the Arduino.
6. The Arduino Uno was powered and programmed via a USB connection to a computer, which also enabled serial communication with a Python program for real-time motion analysis.

7. When an RFID tag was scanned, the Arduino compared the UID to an authorized list. If the tag was valid, the system prompted the user to perform a circular motion, which was detected and analyzed by the MPU6050 sensor. Based on the motion data, the Arduino received either an ‘A’ (Access Granted) or ‘D’ (Denied) signal from Python.
8. When both RFID authentication and correct motion were detected, the servo motor unlocked, and the green LED illuminated. If the RFID tag was invalid or the motion pattern incorrect, access was denied, and the red LED turned on.

Methodology

1. Overview

   This experiment aimed to design and implement a smart access control system that combines RFID-based authentication and motion detection. The Arduino Uno served as the central controller, interfacing with the MFRC522 RFID reader, MPU6050 motion sensor, servo motor, and LED indicators. Data was processed and analyzed in Python, enabling the system to make real-time decisions based on user input and motion patterns.
2. Hardware Setup

1. Arduino Uno	Acts as the main controller to manage sensor inputs, process RFID data, and control outputs.
2. RFID Reader (MFRC522)	Detects and identifies RFID tags or cards for authentication
3. MPU6050 Sensor	Captures acceleration and gyroscopic data to identify motion patterns.
4. Servo Motor	Represents a lock mechanism that rotates to grant or deny access.
5. Red and Green LEDs	Provide visual feedback : green for access

	granted and red for access denied.
6. Resistors	Limit current through the LEDs to prevent damage.
7. Jumper Wires	Connect the sensor to the Arduino's pins for power and data transfer.
8. Breadboard	Provides a base for connecting and organizing the components.
9. USB Cable	Supplies power to the Arduino and enables serial communication with the computer.

3. Circuit Design

   The circuit was constructed on a breadboard following the connections summarized in Table 1. The RFID reader was connected to the Arduino using SPI pins, with SS connected to D10 and RST connected to D8. The servo motor signal wire was connected to D9, while the green and red LEDs were connected to D4 and D3, respectively, each in series with a 220 Ω resistor. The MPU6050 used A4 (SDA) and A5 (SCL) for I^2C communication. All components shared the Arduino's 5V and GND rails for power distribution. This circuit configuration allowed seamless communication between the Arduino, RFID reader, and motion sensor, while the servo and LEDs provided output feedback based on authentication results.

4. Program Development

   The Arduino program was designed to handle RFID scanning, motion verification, and actuator control. It communicated with the Python script running on a computer for real-time motion analysis.

   The process flow was as follows:

1. The RFID module continuously scanned for nearby tags.
2. When a tag was detected, the Arduino checked whether its UID matched the authorized list.
3. If authorized, the Arduino prompted the Python script to start analyzing MPU6050 data.
4. The Python script processed accelerometer readings to detect circular motion.
5. If valid motion was detected, Python sent the character ‘A’ to the Arduino; otherwise, it sent ‘D’.
6. Upon receiving ‘A’, the Arduino activated the servo motor to unlock and turned on the green LED. For ‘D’ or invalid RFID, the servo remained locked, and the red LED illuminated.

This program integrates sensing, authentication, and actuation to form a multi-layered security system that only grants access upon both valid RFID verification and correct motion pattern detection.

Data Collection.

A real-time serial connection between Arduino and Python was established to manage RFID authentication and motion verification.

1. When an RFID card was tapped, its UID appeared in serial monitor and was transmitted to Python for validation.
2. The servo motor, LEDs, and motion sensor (MPU6050) responded to the Python commands with minimal delay, confirming stable two-way communication.

System reactions were observed for three main conditions as shown below.

RFID Authorization	Motion Detected	Servo Position	LED Status
Authorized	Circular motion	180° (Unlocked)	Green on
Authorized	No motion	90° (Locked)	Red on
Unauthorized	N/A	90° (Locked)	Red on

<u>Data Analysis</u>

The collected data confirms that:

1. The RFID module accurately identified valid and invalid UIDs, demonstrating effective digital signal communication.
2. The MPU6050 provided consistent motion readings, allowing clear differentiation between circular and non-circular gestures.
3. The servo motor responded precisely to Python commands ('A' or 'D'), rotating to the expected angles.
4. LED indicators correctly displayed system status, verifying output control functionality.
5. Serial communication between Arduino and Python showed minimal delay and no data loss.

Analysis:

1. The system successfully integrated multiple sensors and actuators, confirming correct serial interfacing between Python and Arduino.

2. The coordination of RFID authentication and motion detection proved reliable for multi-factor access control.
3. The servo and LEDs functioned smoothly, showing accurate execution of commands without signal interference.
4. Overall, the collected data demonstrates that the system operated as designed, ensuring secure and responsive performance.

Results

The experimental results confirm that the integrated smart access system functioned successfully and met its objectives.The system read RFID tags, validated authorised UIDs, verified motion gestures, and controlled the servo lock and LED indicators accordingly.

1. When a valid RFID tag was scanned and a correct circular motion was performed, the servo rotated to 180° (unlocked position), and the green LED turned ON, indicating access granted.
2. When a valid RFID tag was scanned but no motion or an incorrect gesture was performed, the servo remained at 90° (locked position), and the red LED turned ON, indicating access denied.
3. When an unauthorized RFID tag was scanned, the system immediately denied access, keeping the servo locked and turning the red LED ON.
4. The serial communication between Arduino and Python was stable throughout the test, with immediate response to both RFID scans and motion verification results.
5. The servo motor and LED indicators provided clear, observable feedback for every access condition.

The system achieved reliable integration between the RFID reader, MPU6050 motion sensor, servo motor, and LED indicators. Access was granted only when both verification conditions (valid RFID + correct motion) were satisfied, confirming that the system provides secure and efficient access control through combined hardware and software communication.

Discussion

In Task 1, the system successfully captured real-time accelerometer data from the MPU6050 sensor and plotted the X–Y motion using Python's *matplotlib*. The live graph clearly represented hand movement patterns and allowed circular motion gestures to be identified. This shows that the MPU6050 can effectively measure and transmit motion data for simple gesture recognition, proving suitable for interactive mechatronic systems.

In Task 2, the integrated smart access system worked as intended. The RFID module identified authorized tags, while the MPU6050 verified circular motion. When both were correct, the servo unlocked and the green LED lit up; otherwise, access was denied with the red LED. These results confirm the successful integration of motion sensing, RFID authentication, and serial communication between Arduino and Python.

Some limitations were observed. Sensor noise, motion inconsistency, and threshold sensitivity affected motion detection accuracy. Communication latency between Arduino and Python also caused occasional delays in servo response. In addition, minor discrepancies occurred when circular motion was not consistently detected due to differences in motion speed or hand orientation. Despite these issues, the overall results met the experiment's objectives and demonstrated a functional, two-factor smart access system.

The findings imply that combining RFID and motion sensing offers enhanced security compared to single-factor systems. This experiment demonstrates how sensor fusion and real-time data processing can be applied to access control, robotics, and other intelligent mechatronic systems.

<u>Conclusion</u>

The experiments successfully met their objectives by interfacing the MPU6050 and RFID module with Arduino, processing real-time accelerometer data in Python, and integrating both into a functional smart access system. The system accurately detected circular motion and combined it with RFID verification to control a servo and LED indicators, demonstrating effective two-factor authentication. The hypothesis that accelerometer data can enhance RFID-based access control was supported. The experiment proved that motion sensing and serial communication can be integrated smoothly for reliable system operation. Overall, the results highlight the potential of combining motion detection with RFID for improved security and interactivity in smart systems. This approach can be further developed for real-world applications such as automated access, smart locks, and gesture-based control systems.

Recommendations

While the system performed well, several improvements can be made to enhance reliability and usability in real-world applications:

1. Improve Motion Recognition:

   Use more advanced algorithms (e.g., combining accelerometer and gyroscope data or applying Kalman filtering) to detect motion patterns more accurately and reduce false positives.

2. Optimize LDR Sensitivity:

   Calibrate the LDR threshold dynamically according to ambient lighting to ensure consistent motion detection in different environments.

3. Add Data Logging:

   Implement a feature to log RFID scans, timestamps, and motion results for security tracking and debugging.

4. Enhance Power Stability:

   Provide a dedicated power supply for the servo motor to prevent voltage drops affecting other components.

5. Improve User Interface:

   Develop a GUI in Python to display motion tracking visuals, authentication status, and system logs for easier monitoring.

References

[1] https://lastminuteengineers.com/how-rfid-works-rc522-arduino-tutorial/ What is RFID? How It Works? Interface RC522 RFID Module with Arduino

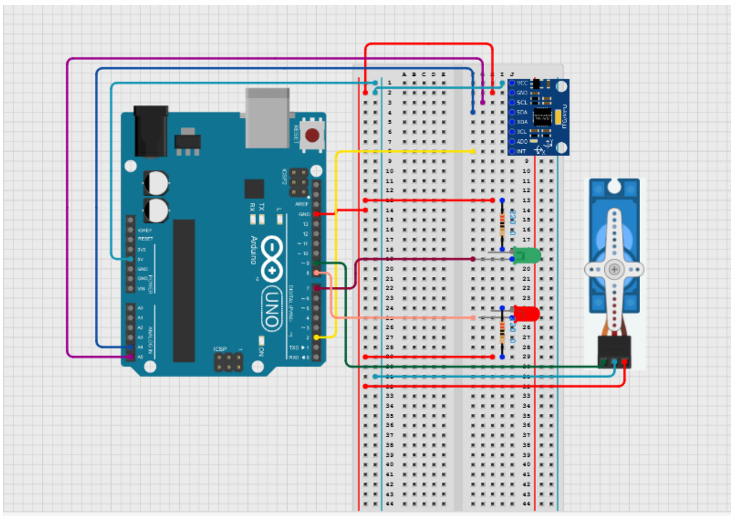
[2] https://randomnerdtutorials.com/security-access-using-mfrc522-rfid-reader-with-arduino/ Security Access using MFRC522 RFID Reader with Arduino

[3] https://randomnerdtutorials.com/arduino-Ɵme-aΣendance-system-with-rfid/ Arduino Time Attendance System with RFID

[4] https://www.instructables.com/Arduino-MFRC522-RFID-READER/ Arduino + MFRC522 RFID READER

Appendices

CirKit Design for Task 2 :



Acknowledgments

Alhamdulillah, all praise and gratitude be to Allah SWT for His blessings, guidance, and mercy that enabled us to successfully complete this project. Throughout the development process, we faced several challenges, but with His will and guidance, we managed to overcome them and complete the project successfully.

We would also like to express our heartfelt appreciation to our lecturer for their continuous guidance, support, and valuable feedback throughout this project. Their advice has greatly helped us in understanding both the theoretical and practical aspects of microcontroller systems and digital display design.

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: Amir	Read [/]
Name: Mohamad Amir Bin Mohamad Yusoff	Understand [/]
Matric Number: 2314687	Agree [/]
Signature: Adam	Read [/]
Name: Adam Hakimi Bin Shah Nur Haizam	Understand [/]
Matric Number: 2314195	Agree [/]
Signature: Ain	Read [/]
Name: Tengku Nurul Ain Binti Tengku Azeezee	Understand [/]
Matric Number: 2313682	Agree [/]