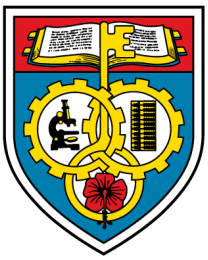


TARUMT

TUNKU ABDUL RAHMAN UNIVERSITY OF MANAGEMENT AND TECHNOLOGY

FACULTY OF COMPUTING AND INFORMATION TECHNOLOGY

Assignment

BAIT2123 Internet of Things
202305

Student's name/ ID Number	:	Sit Yie Sian 21WMR03693
Student's name/ ID Number	:	Yong Xian Ye 21WMR09967
Student's name/ ID Number	:	Teng Kai Deng 21WMR02977
Student's name/ ID Number	:	Ng Wen Jie 21WMR09807
Programme	:	RDS3
Tutorial Group	:	G2
Date of Submission to Tutor	:	1/10/2023

Table of Contents

Table of Contents 2
Smart Manufacturing 3
1.0 Description of System 3
2.0 System Design 5
2.1 Quality Control Module (Ng Wen Jie) 5
2.1.1 IoT Solution Design / Architecture 5
2.1.2 Modules Descriptions 7
2.1.3 Module Key Features 7
2.1.4 Software architecture 8
2.1.5 Hardware architecture 9
2.1.6 Snapshot of Quality Control prototype 11
2.2 Inventory Management Module (Sit Yie Sian) 12
2.2.1 IoT Solution Design / Architecture 12
2.2.2 Functional / Modules Descriptions 13
2.2.3 Hardware / Software architecture: 14
2.2.4 Snapshot / Picture of prototype design 16
2.3 Security (Yong Sian Ye) 17
2.3.1 IoT Solution Design / Architecture 17
2.3.2 Functional / Modules Descriptions 19
2.3.3 Hardware / Software architecture 20
2.3.4 Snapshot / Picture of prototype design 24
2.4 Safety (Teng Kai Deng) 25
2.4.1 IoT Solution Design / Architecture 25
2.4.2 Functional/Modules Description 26
2.4.3 Hardware Architecture 27
2.4.4 Software Architecture 30
2.4.5 Snapshot / Picture of prototype design 32
3.0 Task Allocations 33
4.0 Lesson that we learned / Problem that we faced 34
4.1 Venturing into Hardware 34
4.2 Coding for Embedded Systems 35
4.3 Integration of Data Science and IoT 35
4.4 Collaboration and Problem Solving 36
4.5 The Value of Persistence 36
5.0 Conclusion 37

Smart Manufacturing

1.0 Description of System

* Objective, Business Value / Advantages, Targeted Users.
* Why does everyone need our system?

Objective
The primary objective of our system is to create a comprehensive and versatile smart manufacturing solution that optimizes production processes across various industries. Leveraging state-of-the-art Internet of Things (IoT) technologies, the system aims to revolutionize traditional manufacturing methods, making them more efficient, data-driven, and responsive to the dynamic needs of businesses.

Business Value

- Enhanced Safety: The fire alarm detection module ensures the safety of employees, machinery, and assets by swiftly identifying potential fire hazards and triggering immediate responses, preventing catastrophic losses.
- Secure Authorization: Access control provides a robust layer of security, restricting unauthorized access to manufacturing facilities. This not only safeguards valuable assets but also protects sensitive data and intellectual property.
- Quality Control: Quality control ensures that products meet stringent standards. By identifying defects and inconsistencies in real-time, it minimizes the risk of product recalls, maintains customer satisfaction, and safeguards brand reputation.
- Efficient Inventory Management: Leveraging RFID technology, the system offers a real-time overview of inventory. This results in precise control over stock levels, reduction of carrying costs, and minimization of stockouts.
- Streamlined Operations: The integration of these modules optimizes processes, reduces manual labor, minimizes downtime, and enhances overall operational efficiency.
- Data-Driven Decision Making: Real-time data capture and analysis empower businesses to make informed decisions. It provides actionable insights into manufacturing processes, which are crucial for long-term efficiency.
- Cost Savings: Efficiency improvements, inventory control, and reduced fire risks lead to significant cost savings across the board.

Targeted Users

- Manufacturers: Traditional manufacturers looking to modernize their operations and maximize productivity while minimizing operational costs.
- Plant Managers: Professionals responsible for overseeing day-to-day operations, ensuring

optimal resource utilization, and maintaining a competitive edge.

- Quality Control Personnel: Those in charge of maintaining rigorous quality standards through real-time monitoring and data analytics.
- Supply Chain Managers: Individuals responsible for optimizing the flow of materials and products, reducing inventory costs, and minimizing lead times.
- Business Executives: Decision-makers who rely on real-time data and analytics to make informed choices that affect the entire organization, leading to increased competitiveness.
- Maintenance Teams: Personnel responsible for machinery and equipment upkeep, ensuring minimal downtime and optimal operational efficiency.
- Data Analysts: Experts who analyze the data generated by the system to gain valuable insights for continuous improvement and innovation.

Why Does Everyone Need Our System?

- Safety and Compliance: Ensuring a safe working environment is not just a regulatory requirement but also a moral obligation. Our fire detection and access control modules guarantee a secure workplace.
- Quality Assurance: In an age where quality and brand reputation are paramount, our quality control module becomes indispensable. It ensures products meet the highest standards consistently.
- Operational Efficiency: The integration of inventory management streamlines operations. It prevents understock or overstock scenarios, reducing costs and increasing efficiency.
- Data-Driven Decision-Making: Informed decisions are the foundation of success. Our system provides the real-time data necessary to make these decisions in manufacturing and supply chain management.

2.0 System Design

2.1 Quality Control Module (Ng Wen Jie)

2.1.1 IoT Solution Design / Architecture

The Quality Control module in our IoT system is essential for ensuring the production of top-notch goods by continuously overseeing and evaluating different quality factors.

Sensor Nodes:

- Sensor nodes strategically positioned across the manufacturing process are central to our quality control system. These nodes are equipped with various sensors, including those for temperature and humidity, to collect crucial data points.

Data Gathering:

- Sensor nodes gather real-time data concerning temperature and humidity levels, critical elements affecting product quality. Data is collected at regular intervals for precision monitoring.

Connectivity Layer:

- The data collected by sensor nodes is transmitted wirelessly to a central hub through communication protocols such as Wi-Fi or Bluetooth. This layer ensures a smooth data flow from the factory floor to the central processing unit.

Central Processing Unit (CPU):

- The CPU is the core of our quality control system, responsible for data consolidation, processing, and analysis. Here's its role:
    - Data Consolidation: The CPU compiles data from all sensor nodes, organizing it into an organized format for analysis.
    - Data Processing: Data preprocessing is performed to eliminate anomalies and guarantee data quality.
    - Analysis: Advanced data analysis techniques assess quality parameters, including temperature and humidity thresholds.

Real-Time Monitoring and Notifications:

- The CPU continually monitors data for any deviations from preset quality thresholds. If thresholds are surpassed, or unusual patterns emerge, the system generates instant notifications to inform relevant personnel. These notifications can be transmitted through various communication channels like email, SMS, or dashboard alerts.

Historical Data Repository:

- All raw or processed data is securely stored in a database for historical analysis and record maintenance. This historical data offers insights into long-term quality patterns and proves invaluable for quality enhancement efforts.

Dashboard and Data Visualization:

- Our IoT system boasts a user-friendly dashboard accessible to quality control managers and authorized users. This dashboard provides real-time visual representations of quality parameters, historical data trends, and alert logs. It empowers users to make informed decisions promptly.

Threshold Configuration:

- Quality control managers have the flexibility to set quality thresholds via the dashboard, enabling them to tailor the system to distinct production processes and product specifications.

Integration with Production Systems:

- Our IoT solution can be integrated with existing production systems and machinery to ensure seamless quality control. This integration enables automatic adjustments to the production process based on real-time quality data.

Scalability and Backup Measures:

- Our architecture is designed with scalability in mind, allowing for the effortless addition of new sensor nodes and data processing capabilities as the manufacturing facility expands. Furthermore, it incorporates backup measures to ensure system reliability and data integrity.

In summary, the IoT solution design and structure of the Quality Control module are meticulously crafted to offer real-time monitoring, analysis, and management of crucial quality parameters within the manufacturing process. By harnessing sensor data, connectivity, and advanced analytics, we empower organizations to uphold high-quality standards, minimize defects, and enhance overall product quality. This architecture is adaptable, extensible, and suitable for diverse manufacturing settings.

2.1.2 Modules Descriptions

The Quality Control module is an integral part of our Smart Manufacturer Dashboard, designed to ensure the highest standards of product quality and equipment safety. In today's competitive manufacturing landscape, maintaining quality is paramount, and this module provides the tools and insights needed to achieve that goal.

2.1.3 Module Key Features

1. Real-Time Monitoring: Gain real-time insights into temperature and humidity data, helping you identify deviations that could affect product quality.

2. Threshold Settings: Set and adjust thresholds for temperature and humidity to define acceptable quality ranges. Receive alerts when conditions fall outside these ranges.

3. Historical Data Analysis: Access historical data records, allowing you to track quality trends over time and make data-driven decisions.

4. MAC Address Tracking: Keep track of individual devices or machines (identified by unique MAC addresses) to pinpoint the source of quality issues.

5. Graphical Visualization: Visualize temperature and humidity data through interactive graphs and charts for deeper analysis.

6. Alerts and Notifications: Receive immediate notifications when quality thresholds are breached, enabling rapid response and corrective actions.

2.1.4 Software architecture

The software architecture within the Quality Control module is built upon a robust and flexible framework designed to handle data collection, analysis, and real-time monitoring seamlessly.

Data Collection Layer:

- At the foundation of our software architecture is the data collection layer. This component manages the reception of real-time data from sensor nodes. It employs efficient data retrieval mechanisms to ensure timely acquisition.

Real-Time Monitoring Component:

- To enable real-time quality control, we've integrated a monitoring component that continuously observes the data stream. If deviations from established quality parameters are detected, instant alerts and notifications are generated to keep stakeholders informed.

Dashboard and User Interface:

- The software architecture includes an intuitive dashboard and user interface accessible via web or mobile platforms. This interface allows authorized users to monitor quality metrics, set thresholds, and receive alerts. It provides graphical representations of quality data for straightforward interpretation.

Integration Layer:

- An integration layer is incorporated into the software architecture to ensure seamless integration with existing systems. This layer facilitates communication with production machinery, enabling automated adjustments based on real-time quality feedback.

Scalability and Extensibility:

- Our software architecture is designed with scalability and extensibility in mind. It can adapt to changing manufacturing requirements and accommodate the addition of new features or modules.

In summary, the software architecture of the Quality Control module is engineered to provide a comprehensive and responsive solution for quality assurance. It processes real-time data efficiently, applies advanced analytics, and gives users actionable insights through an intuitive interface.

2.1.5 Hardware architecture

In the Quality Control module, the hardware architecture serves as the physical foundation for data collection, sensor management, and actuation.

Sensor Nodes:

- The core components of our hardware architecture are the sensor nodes strategically placed throughout the manufacturing environment. These nodes are equipped with various sensors, including Grove DHT sensors for temperature and humidity monitoring. These sensors provide crucial quality-related data.

Microcontrollers:

- Sensor nodes are controlled by microcontrollers, which are responsible for data acquisition, processing, and communication. These microcontrollers ensure precise data collection and reliability. They interface with Grove DHT sensors for temperature and humidity measurements.

Connectivity Module:

- Data collected by sensor nodes, including temperature and humidity data from Grove DHT sensors, is transmitted wirelessly through a connectivity module. This module supports communication protocols Wi-Fi, ensuring seamless data transfer to the central processing unit.

Central Processing Unit (CPU):

- The CPU is at the heart of the hardware architecture, which acts as the central hub for data aggregation, analysis, and actuation. This high-performance unit is equipped to handle real-time data from multiple sensor nodes simultaneously. It also manages the actuation components.

Actuation Components:

- Our hardware architecture includes actuation components to respond to quality-related events. These components include:
    - Grove Buzzer: The Grove Buzzer serves as an audible alert system, activated when quality thresholds are breached.
    - Grove LED: Grove LEDs provide visual indicators in response to quality events, aiding in real-time monitoring.
    - Fan: The fan represents a cooling mechanism in the factory. It can be activated based on temperature data from Grove DHT sensors to regulate the environment.
    - Servo Motor: The servo motor is used to represent the production line and can be controlled as needed based on quality assessments.

Power Management:

- The hardware architecture incorporates power management solutions to ensure the longevity of sensor nodes and actuation components. Battery-powered nodes are optimized for energy efficiency, and backup power sources are available to prevent data loss during power disruptions.

Data Storage and Backup:

- All data collected by the hardware architecture, including sensor data and actuation logs, is securely stored in Firebase. Backup mechanisms are in place to safeguard against data loss, providing redundancy and reliability.

User Interfaces:

The user interfaces (UI) integrated into the hardware architecture serve multiple purposes, enhancing the usability and functionality of the system. These interfaces are designed to:

1. Real-time Data Display: The primary function of the user interface is to display real-time data related to temperature and humidity. Users can readily monitor these essential metrics to ensure quality control.

2. Threshold Temperature Adjustment: Users are provided with intuitive controls to modify the desired threshold temperature values. This feature empowers users to customize temperature limits based on specific quality requirements.

3. Historical Data Access: The user interface allows users to access historical data according to specific MAC addresses. It provides a user-friendly way to retrieve past quality-related information for analysis and reference.

4. Graphical Representation: To facilitate data analysis, the UI incorporates interactive graphs. These visual representations offer insights into temperature and humidity trends over time, aiding in decision-making and quality assessments.

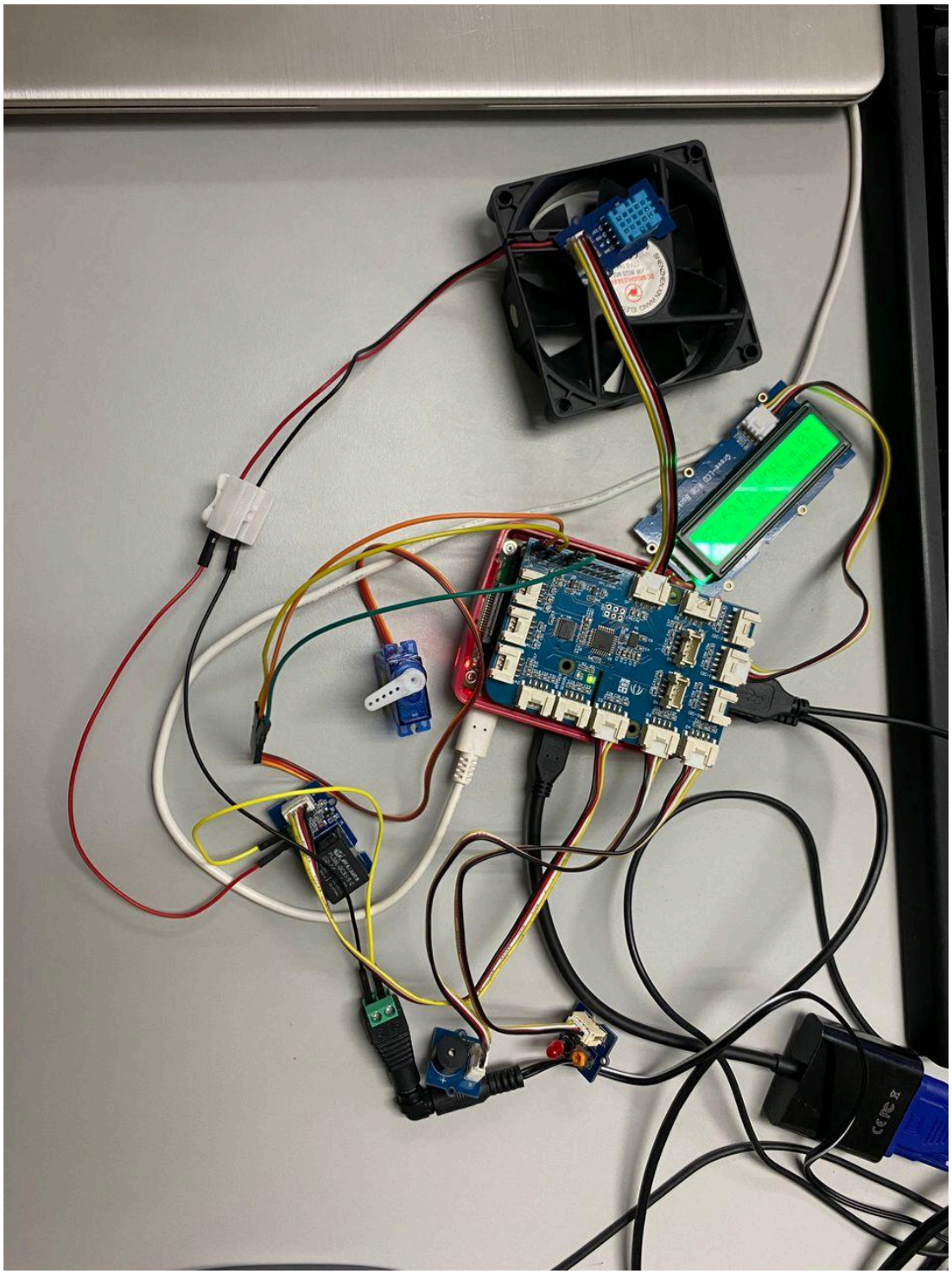
By encompassing these functionalities, the user interfaces provide real-time insights and enable users to actively manage and analyze data, contributing to effective quality control in manufacturing environments.

Scalability and Redundancy:

- Designed with scalability and redundancy in mind, the hardware architecture allows for the addition of new sensor nodes and actuation components as the manufacturing process expands. Redundant components are integrated to ensure system resilience.

In summary, the hardware architecture of the Quality Control module provides the essential infrastructure for reliable and accurate data collection, sensor management, and actuation. It encompasses sensor nodes equipped with Grove DHT sensors, microcontrollers, communication modules, actuation components, and a central processing unit. This architecture is engineered for durability, scalability, and seamless integration with existing systems while incorporating specific sensors and actuators tailored for quality control in manufacturing environments.

2.1.6 Snapshot of Quality Control prototype



2.2 Inventory Management Module (Sit Yie Sian)

2.2.1 IoT Solution Design / Architecture

1. IoT Devices and Sensors:
   - RFID Reader: RFID readers are used to scan RFID tags on products. Each product is tagged with a unique RFID tag, allowing for accurate tracking and identification.
   - Ultrasonic Sensor: Ultrasonic sensor to act as a parking system for the manufactured products to monitor the distance between products on storage shelves, optimizing space utilization within the warehouse.
   - Temperature and Humidity Sensor: These sensors monitor the environmental conditions in the warehouse. Temperature and humidity data help prevent spoilage of temperature-sensitive products.

2. Raspberry Pi (Edge Device):
   - The Raspberry Pi serves as the edge computing device that interfaces with the sensors. It runs the Python script to manage RFID, ultrasonic sensor, and environmental sensors.
   - The Raspberry Pi interacts with the inventory database, reads RFID data, and updates the inventory information based on incoming or outgoing products.

3. RFID Tags:
   - Each product is tagged with an RFID tag that contains a unique identifier. These tags are scanned by the RFID reader for product identification.

4. Cloud Backend:
   - Firebase Cloud Database: The cloud backend, powered by Firebase, stores and manages the inventory data. It provides real-time synchronization, which allows data to be accessed and updated from multiple devices and locations simultaneously.

5. Mobile Notifications:
   - Twilio Integration: Twilio is used to send SMS notifications in cases where the warehouse is full. It integrates with the cloud backend and sends SMS messages to designated staff members.

6. User Interface:
   - OLED Display: An OLED display is integrated with the Raspberry Pi to provide a real-time view of RFID tag information, product details, and the current quantity in stock.

7. Data Flow and Workflow:
   - RFID Scanning: When a product with an RFID tag is scanned, the Raspberry Pi reads the RFID data and retrieves product information from the hardcoded product_info or Firebase database.
   - Inventory Management: The Raspberry Pi updates the inventory data in the Firebase Cloud Database based on whether the status is "IN" or "OUT." This information includes the product name, description, cost price, selling price, and current quantity.
   - Environmental Monitoring: The temperature and humidity sensors continuously monitor

the environmental conditions in the warehouse. If the conditions exceed the defined limits, staff members are notified through SMS.
- Space Utilization: The ultrasonic sensor monitors the distance between products on storage shelves to optimize space utilization.
- Mobile Notifications: When the warehouse is full and cannot store additional products, Twilio sends SMS notifications to staff members.

8. Firebase Cloud Database Structure:
   - The Firebase database is structured to store product information, current quantities, and inventory activity logs by product name.

9. Security:
   - Access Control: Access to the Raspberry Pi and the cloud database is restricted through security measures such as passwords and user authentication.

This IoT inventory management solution enables real-time tracking, monitoring, and management of product inventory. It ensures efficient space utilization, prevents environmental factors from affecting product quality, and sends alerts when the warehouse is full. It provides valuable insights for inventory control and enhances overall warehouse management.

2.2.2 Functional / Modules Descriptions

1. RFID Tag Scanning Module:
   - Function: This module is responsible for scanning RFID tags attached to products as they are moved into or out of the warehouse. The scanned data is used to identify the products and update their inventory status.
   - Description: When a product with an RFID tag is scanned, the module retrieves the unique identifier, looks up product information, and updates the inventory data in the Firebase Cloud Database.

2. Inventory Database Management Module:
   - Function: This module manages the storage, retrieval, and update of inventory data. It includes product details, current quantities, and activity logs.
   - Description: Product information, such as name, description, cost price, selling price, and current quantity, is stored in the Firebase Cloud Database. The module updates quantities based on the "IN" or "OUT" status and logs all activity.

3. Environmental Monitoring Module:
   - Function: This module continuously monitors the temperature and humidity in the warehouse to prevent spoilage of temperature-sensitive products.
   - Description: Temperature and humidity sensors collect data and trigger alerts if the conditions exceed defined limits. This ensures the quality of stored products.

4. Parking System:
   - Function: This module employs an ultrasonic sensor to act as a parking system for the manufactured products. It monitors the distance between products on storage shelves,

optimizing space utilization within the warehouse.

- Description: The ultrasonic sensor constantly measures the distance between products on the shelves. When a product is moved in or out, the sensor ensures that products are placed with optimal spacing to maximize the use of available space. If the distance between products is within the defined range, the system indicates that the space is available for parking, ensuring efficient utilization of storage capacity. This functionality prevents overcrowding of products, reducing the risk of damage and facilitating easy retrieval.

5. Mobile Notifications Module:
   - Function: This module integrates with Twilio to send SMS notifications to staff members when the warehouse is full.
   - Description: When the warehouse reaches its maximum capacity, the system sends SMS notifications to staff, ensuring they are informed of the space limitations.

6. User Interface Module:
   - Function: The OLED display module provides a real-time view of RFID tag information, product details, and current quantity in stock.
   - Description: The OLED display presents product information and quantities, offering real-time insights for users.

7. Data Retrieval via Website:
   - Function: A website is designed to retrieve data from the Firebase Cloud Database, providing access to inventory counts, activity logs, and warehouse full status.
   - Description: Users can access data such as product inventory quantities, historical activity logs, and receive notifications about warehouse capacity via the website.

8. Mac Address Identification:
   - Function: The system uses the Raspberry Pi's MAC address to uniquely identify the device and, by extension, the system.
   - Description: The MAC address serves as an identifier for the system, ensuring secure access and tracking of the Raspberry Pi device.

The Inventory Management Module seamlessly combines IoT technologies, RFID scanning, environmental monitoring, and space utilization optimization to efficiently manage product inventory. It also provides real-time access to data through a user-friendly website, ensuring that users have up-to-date information about inventory status, activity logs, and warehouse capacity.

2.2.3 Hardware / Software architecture:

Hardware Architecture

The Inventory Management Module relies on a combination of hardware components to interact with the physical world and ensure effective inventory management. Here's an overview of the key hardware elements:

1. Raspberry Pi:
   - Central controller and processing unit.
   - Manages data flow between hardware components and software.
   - Connects to the internet for remote access and data storage.

2. RFID Reader (MFRC522):
   - Reads RFID tags attached to inventory items.
   - Identifies products and updates inventory data in real-time.

3. Ultrasonic Sensor:
   - Monitors the distance between stored items.
   - Acts as a parking system for products to optimize space utilization in the warehouse.

4. DHT Sensor (Temperature and Humidity Sensor):
   - Measures the environmental conditions in the warehouse.
   - Ensures that products are stored under suitable temperature and humidity levels.

5. LEDs and Buzzer:
   - Visual and auditory indicators for system alerts, status updates, and warnings.

6.OLED Display (SH1106):
   - Provides real-time information on inventory items, such as product name and quantity.
   - Displays alerts and warnings, such as temperature and humidity thresholds.

Software Architecture

The software architecture is designed to process data, interface with hardware, and enable user interaction and remote access. Here's an overview of the key software components:

1. Python Scripts:
   - RFID Tag Reader Script: Reads RFID tags, identifies products, and updates inventory data.
   - Ultrasonic Sensor Script: Monitors product spacing for efficient storage.
   - DHT Sensor Script: Measures temperature and humidity, triggers alerts if conditions are unfavorable.
   - Firebase Integration: Interacts with the Firebase database to retrieve and update inventory data.

2. Firebase Database:
   - Serves as the central repository for inventory data.
   - Stores product information, quantities, and activity logs.
   - Captures temperature and humidity data.
   - Enables real-time data access from web interfaces and mobile apps.

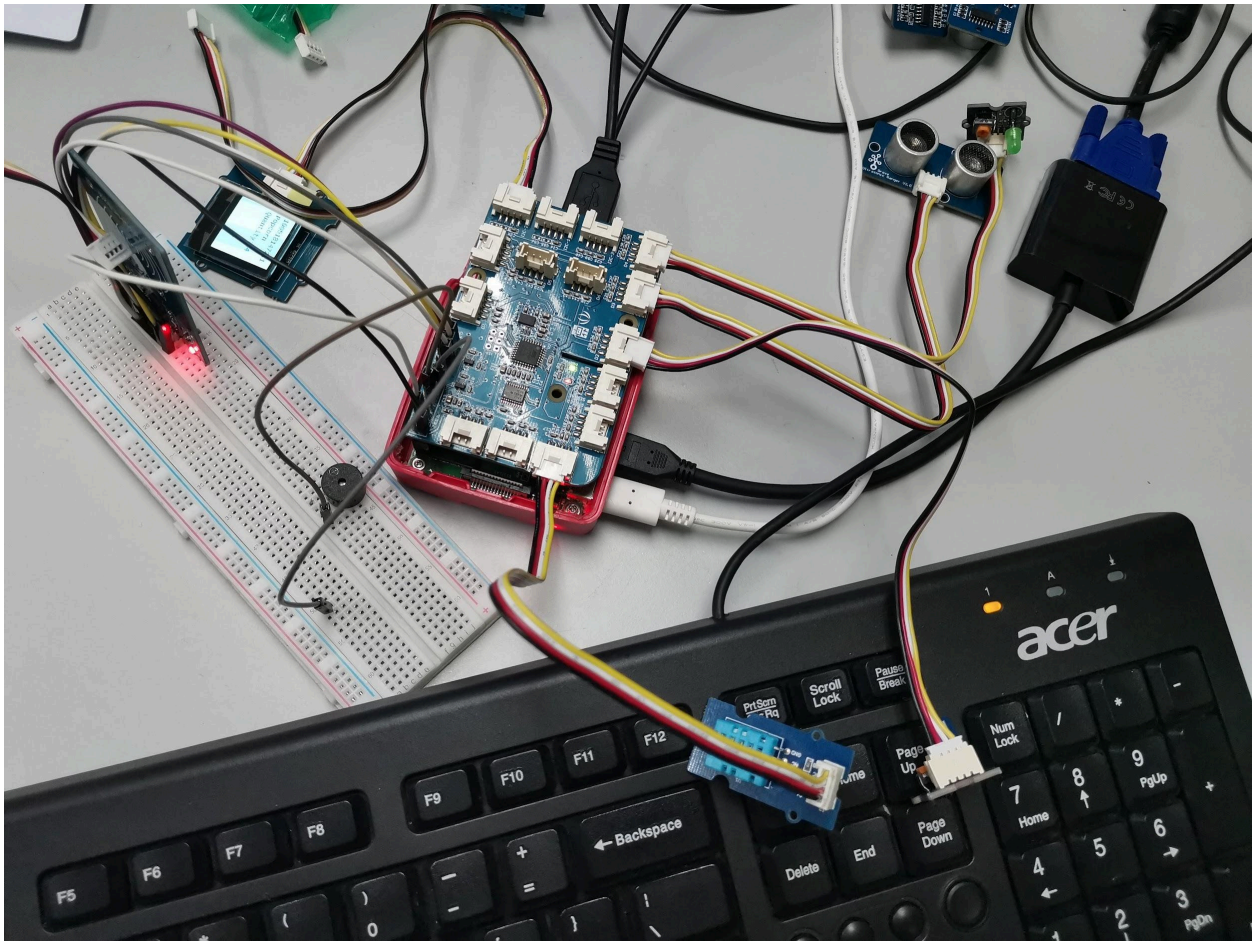
3. User Interface:
   - Web-Based Dashboard: Allows users to access and visualize inventory counts, activity logs, and warehouse status.

4. Alerting and Notification:
   - Twilio Integration: Sends SMS alerts to designated personnel when the warehouse reaches full capacity or when environmental conditions exceed defined thresholds.

5. Cloud Connectivity:
   - Enables remote access to inventory data and status from anywhere with an internet connection.
   - Facilitates data analysis and reporting.

The combination of hardware and software architecture ensures efficient inventory management by providing real-time tracking, monitoring, and alerting while optimizing warehouse space utilization and maintaining suitable storage conditions for products.

2.2.4 Snapshot / Picture of prototype design



2.3 Security (Teng Kai Deng)

2.3.1 IoT Solution Design / Architecture

Sensor Nodes

- In the security module, the RFID readers and tags are used for access control systems.
- Sensors to monitor environmental conditions, such as temperature and humidity for fire and hazard detection.

Data Gathering and Cloud Storage (Firebase Cloud Database)

- Continuous real-time data is gathered from security sensors, environmental sensors, and other sources.
- This data is seamlessly transmitted and securely stored in a Firebase Cloud Database, enabling real-time synchronisation and access from multiple devices and locations simultaneously. This unified approach ensures that data is readily available for security analysis and historical reporting while maintaining real-time accessibility.

Connectivity Layer

- Wired and wireless communication protocols for data transmission within the manufacturing facility.

Central Processing Unit (CPU)

- Centralised security management system that serves as the core for data consolidation and analysis.

Real-Time Monitoring and Alerts

- Real-time monitoring of sensor data for security breaches.
- Alerts are sent through security consoles, facility control systems, and to security personnel.

Historical Data Repository

- Securely stored in on-premises or cloud-based databases for historical analysis and compliance reporting.

User Interface and Dashboard (JavaScript)

- Update logs for authorised users are displayed.
- HTML and JavaScript are used as user interfaces to display update logs and real time data.

Access Control Integration

- Integration with existing access control systems and biometric authentication.
- Monitoring of entry/exit data to detect unauthorised access.

Raspberry Pi/Edge Device Integration

- Raspberry Pi serves as a local computing unit that collects data from various sensors within the manufacturing facility. For example, it receives data from surveillance cameras, motion detectors, access control systems, environmental sensors, and RFID readers.
- Raspberry Pi can run security algorithms and rules locally to detect security breaches and anomalies in real-time. For example, it can monitor the access logs from RFID readers, and trigger alarms when unauthorised access is detected.

Threshold Configuration

- Configurable security policies and alert thresholds to tailor the system to specific manufacturing needs.

Integration with Production Systems

- Seamless integration with production machinery and automation systems for immediate security-related actions or automatic adjustments.

Scalability and Redundancy

- Designed for scalability, accommodating additional sensors, cameras, and other security devices.
- Incorporates redundancy measures for system reliability and data integrity, including backup power systems

2.3.2 Functional / Modules Descriptions

RFID Sensor Integration and Access Control

- Function: This module is responsible for interfacing with RFID sensors and access control systems
- Description: Used to monitor and control product movements and personnel access within the manufacturing facility, ensuring security and access management.

Cloud Integration

- Function: This module ensures real-time data synchronisation and remote access to security information, enabling timely response and analysis.
- Description: Used to facilitate secure communication with the Firebase Cloud Database.

Responsive User Interface (UI)

- Function: The UI module employs HTML, CSS, and JavaScript to provide a user-friendly and interactive web-based dashboard
- Description: Allows security personnel to access real-time data, adjust security settings, and respond to security incidents efficiently.

Centralised Database

- Function: The central database stores records of each module in this assignment, all are stored within the Firebase Cloud Database as a centralised database.
- Description: Data such as security incidents, access control logs, and environmental data are stored together with other module's data, serving as the core repository for centralised information analysis, and reporting.

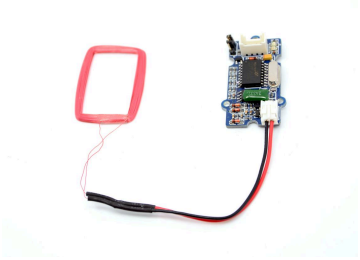
Serial Number Identification

- Function: The system uses the RFID access card or tag's serial number to uniquely identify the specific permission to access the system.
- Description: The RFID reader is programmed to a system that employs a unique serial number of RFID access cards or tags to grant specific permissions for system access, ensuring secure and individualised authentication.

2.3.3 Hardware / Software architecture

Hardware architecture

1) Groove RFID Reader

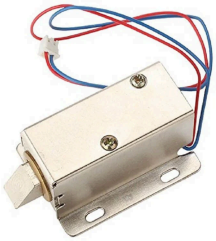


- Used to scan and authorise users into a specific facility depending on their access card's authority.
- In this module, the RFID reader is used to detect and approve access to authorised users into a specific facility. The reader will read and grant access to the users depending on their access card's serial number according to the system.

2) RFID Card and Tag Button

- Used to grant access to specific facilities depending on the serial number and the permission given.
- In the security module, the RFID card and tag is used as a mediator between the RFID reader and the solenoid lock. The RFID reader will read the serial number of the access card and decide whether it has permission to access a specific facility according to the system.

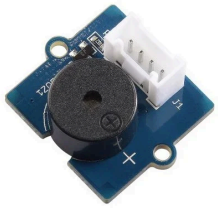
3) Solenoid Lock



- Used to lock or unlock the access door into the facilities.

- In this module, the solenoid lock acts as a security measure by unlocking when a verified serial number of an access card is detected. The solenoid lock will not activate when an anonymous access card is being scanned into the RFID reader.

4) Grove Buzzer



- Auditory alerts when the system is activated and deactivated.
- In the security module, the buzzer acts as an indication whether the user is permitted or prevented from entering the facility.

5) Grove RGB LCD Backlight

- Used to display permission and greet the users entering the facility.
- In this module, the LCD backlight acts as an interaction whether the user is permitted to enter the facility. It will display either "Access Granted" and "Access Denied" depending on the authority of the user's access card.

6) Grove LED

- Visual indication when the system is activated and deactivated.
- In the security module, the led acts as an indication whether the accessed door is locked or unlocked. It will light up when the door is unlocked and vice versa.

7) Grove Relay

- The Grove Relay acts as a switch for electrical power.
- When activated, it can open or close the electrical circuit, allowing power to flow to the solenoid lock or cutting off power.

Software architecture

Python Scripts

- RFID Sensor Integration Script: Manages the interaction with RFID sensors, reading RFID data to monitor product movements and access control.
- Sensor Data Analysis Script: Processes data from various security sensors, analysing it for security breaches and anomalies.
- Environmental Monitoring Script: Monitors data from environmental sensors, like temperature and humidity, and triggers alerts in case of unfavourable conditions.
- Firebase Integration Script: Interacts with Firebase for real-time data access, ensuring that security information is readily available.

Firebase Database

- Serves as the centralised repository for security data, storing records of security breaches, access logs, and environmental conditions.
- Captures RFID sensor data, including product and asset movements.
- Facilitates real-time data access from web interfaces and mobile apps, ensuring timely response to security incidents.

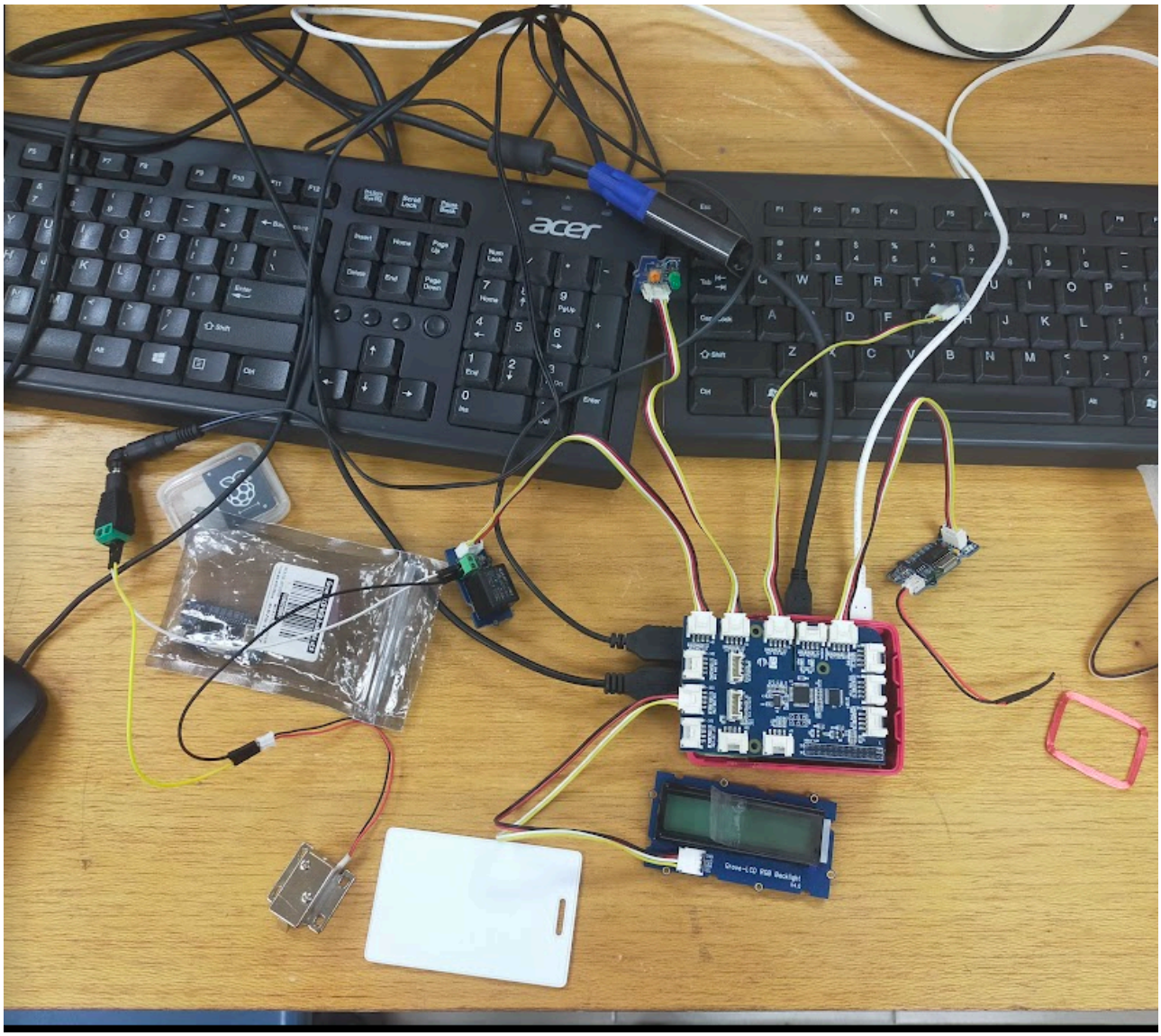
User Interface:

- Web-Based Security Dashboard: Offers a user-friendly interface that provides security personnel and authorised users with access to continuous security data. This includes incident logs, access records, and environmental conditions, enabling informed decision-making and rapid response.
- The user interface of the Smart Manufacturing Security System, built with HTML, CSS, and JavaScript, provides real-time data updates, interactive alerts, and dynamic visualisations, enabling security personnel to monitor and respond to security incidents with efficiency.

Cloud Connectivity:

- Enables remote access to security data and real-time status monitoring from anywhere with an internet connection.
- Facilitates data analysis and reporting, ensuring that security managers can effectively respond to security breaches and access incidents.

2.3.4 Snapshot / Picture of prototype design



2.4 Safety (Yong Xian Ye)

* Reference System (just a short description if you referred to some example)
* Your overall IoT Solution Design / Architecture
* Functional / Modules Descriptions:
* Hardware / Software architecture:
* Snapshot / Picture of prototype design.

2.4.1 IoT Solution Design / Architecture

Sensor Nodes:

The IoT solution begins with sensor nodes, which are the physical devices responsible for sensing and collecting data from the environment. These nodes support various sensors such as temperature, humidity, motion detectors and IR flame sensor. Key features include low power consumption for extended device life, basic data preprocessing capabilities, and the ability to efficiently integrate diverse sensors.

Data Gathering:

Following data collection from sensor nodes, the data gathering module aggregates and preprocesses the information before transmission. This stage involves filtering out noise, combining data from multiple nodes for efficient transmission, and ensuring accurate timestamping. The objective is to optimise the quality and relevance of data before it moves further through the system.

Connectivity Layer:

The connectivity layer facilitates communication between sensor nodes, edge devices like Raspberry Pi, and the cloud. It employs wireless protocols such as Bluetooth, Zigbee, or Wi-Fi, depending on the application requirements. Security measures, including data encryption, are crucial to ensure secure communication, and low latency is essential for real-time data transmission.

Raspberry Pi/Edge Device Integration:

The integration of Raspberry Pi or similar edge computing devices serves as a local processing hub. This layer handles edge analytics, local storage for temporary data, and implements security measures. Processing data closer to the source reduces latency and ensures quick responses to critical events, contributing to the efficiency of the overall IoT solution.

Real-Time Monitoring and Notifications:

Real-time monitoring and notifications are critical components of the IoT architecture. This layer includes a dashboard for live visualisation of sensor data, an alerting system that triggers notifications based on predefined rules, and integration with notification services such as SMS, email, or push notifications. For example, Twilio integration allows for seamless delivery of SMS notifications, enriching the alerting system within the real-time monitoring layer. This feature ensures that critical information reaches stakeholders in a timely manner, enhancing the responsiveness of the IoT solution.

Historical Data Repository:

The historical data repository layer focuses on storing and managing data for analysis and reporting. It utilises a relational or time-series database, implements data retention policies, and incorporates query and analysis tools. Efficient storage and retrieval of historical data contribute to the system's ability to provide insights into long-term trends and patterns.

2.4.2 Functional/Modules Description

Sensor Nodes Module:

The Sensor Nodes module serves as the foundation, collecting data from the environment through physical sensors. It supports diverse sensors, ensuring low power consumption for extended device life, basic data preprocessing, and integration of various sensing capabilities.

Data Gathering Module:

Responsible for post-data collection processes, the Data Gathering module aggregates and preprocesses information before transmission. This involves data filtering, aggregation from multiple nodes, and accurate timestamping to optimize the quality of transmitted data.

Connectivity Layer Module:

The Connectivity Layer module facilitates seamless communication among sensor nodes, edge devices (like Raspberry Pi), and the cloud. It employs wireless protocols (Bluetooth, Zigbee, Wi-Fi) and implements security measures, including encryption, to ensure secure, low-latency communication.

Raspberry Pi/Edge Device Integration Module:

Integrating Raspberry Pi or similar edge devices, this module enables local processing. It handles edge analytics, temporary data storage, and security measures, allowing for efficient data processing closer to the source and ensuring quick responses to critical events.

Real-Time Monitoring and Notification Module:

The Real-Time Monitoring and Notifications module continually monitors data and triggers notifications based on predefined criteria. It features a real-time dashboard for visualisation, an alerting system, and integration with Twilio for SMS notifications, ensuring timely communication of critical events. Enhancing communication capabilities, the Mobile Notification module integrates Twilio for SMS notifications. Leveraging Twilio's APIs, it seamlessly delivers critical information to stakeholders on their mobile devices, enriching the alerting system within the real-time monitoring layer.

Historical Data Repository Module:

Focused on storing and managing historical data for analysis, the Historical Data Repository module utilises a relational or time-series database. It enforces data retention policies and integrates query and analysis tools, facilitating efficient storage, retrieval, and analysis of historical data.

2.4.3 Hardware Architecture

- **Sensor Nodes Hardware**
    - **Microcontrollers**

      Power-efficient processors responsible for handling data processing tasks on the sensor nodes. These microcontrollers interpret signals from sensors and manage the communication with other components.

    - **Sensors and Actuators**

      Devices responsible for collecting data from the sensors and initiating actions based on actuators. Sensors that include IR flame sensor and Actuator that include buzzer, LED, fan, relay, RGB LCD Backlight, stepper motor.

- **Edge Device Hardware (Raspberry Pi)**
    - **Raspberry Pi Board**

      A versatile computing platform capable of running various applications and handling edge analytics. It serves as a local processing unit for data received from sensor nodes.

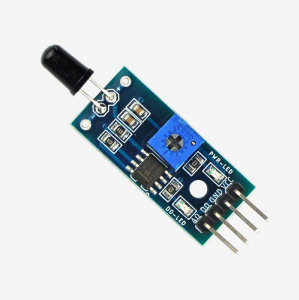
    - **Local Storage Device**

      An SD card is used for temporary storage of data before it is transmitted to the cloud. This ensures data availability for local processing and backup.

- **Communication Interfaces**
  Wi-Fi and Ethernet interfaces enable connectivity with the cloud and other devices in the network.

Sensor and Actuator

- **IR flame sensor**

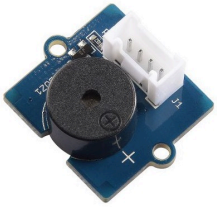


The IR Flame Sensor detects the presence of flames or infrared radiation emitted by a flame. It is commonly used for fire detection in various applications.

- **Grove RGB LCD Backlight**

The Grove RGB LCD Backlight is typically a display component that can show characters and colors. It often acts as a user interface or information display.

- **Grove Buzzer**



The Grove Buzzer is an audible alarm or notification device. It produces sound when an electrical signal is applied.

- **Grove LED**

The Grove LED (Light Emitting Diode) is a visual indicator that emits light when a voltage is applied.

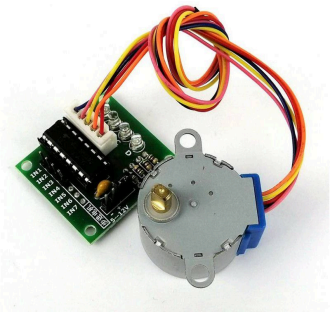
- **Grove Relay**

The Grove Relay is an electromechanical switch that controls the flow of electricity to a connected device.

- **12V Fan**

The 12V Fan is an actuator that generates airflow when powered.

- **Stepper motor**



The Stepper Motor is a type of motor that moves in discrete steps, usually in a circular motion.In this project I used a stepper motor to represent the water pump.

2.4.4 Software Architecture

- **Architectural Patterns:**
  Architectural patterns provide foundational templates for solving recurring design challenges. Common patterns include MVC, which separates applications into model, view, and controller components, and Microservices, which structures systems as a collection of independently deployable services. The choice of pattern depends on factors like scalability, maintainability, and specific project requirements.

- **Components and Modules:**
  Components and modules serve as the fundamental building blocks of a software system, encapsulating specific functionalities. These elements are organized hierarchically, contributing to the modularity and maintainability of the codebase. In a web application, for instance, modules might encompass user authentication, database access, and front-end views.

- **Communication Protocols:**
  Communication protocols define the rules for data exchange between software components, ensuring effective interaction. Examples include HTTP/HTTPS for web systems and MQTT for IoT communication. The choice of protocol depends on factors such as data transfer requirements, latency, and the nature of the communication.

- **Data Management:**
  Data management addresses how data is stored, retrieved, and managed within the system. It involves decisions about using relational databases such as firebase's realtime database based on factors like data structure, scalability needs, and transactional requirements.

- **Design Patterns:**
  Design patterns offer reusable solutions to common problems in software design. These patterns, such as the Singleton pattern for ensuring a class has only one instance, and the Observer pattern for distributed event handling, enhance code maintainability, readability, and scalability.

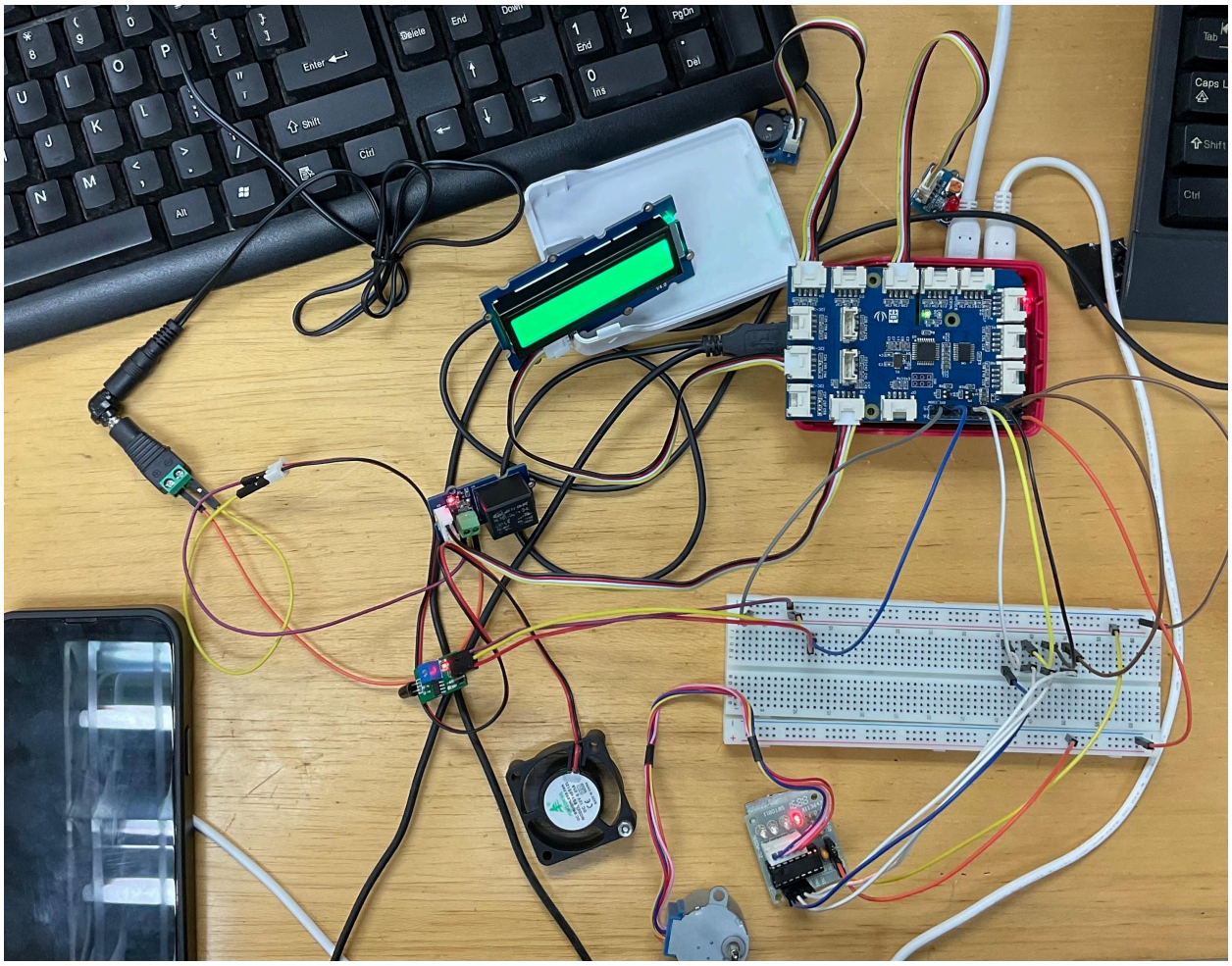
- **Deployment Architecture:**
  Deployment architecture describes how a software system is distributed across different hardware components or servers. Choices between monolithic and microservices architectures, containerization, and serverless computing impact scalability, maintenance, and resource utilization.

- **Scalability and Performance:**
  Scalability and performance considerations focus on how the system handles increased loads and optimizes for responsiveness. Strategies may include load balancing, caching mechanisms, and asynchronous processing to ensure optimal performance under varying conditions

- **User Interface Design:**
  User interface design is concerned with the structure and layout of visual components. Considerations include adherence to user experience (UX) principles, responsive design for various devices, and providing an intuitive interface for end-users.

- **Monitoring and Analytics:**
  Monitoring and analytics involve incorporating tools and strategies for tracking the performance and usage of the software system. Integration with monitoring tools, log analysis, and user analytics facilitates proactive issue identification, performance optimization, and data-driven decision-making.

2.4.5 Snapshot / Picture of prototype design



3.0 Task Allocations

Module Name	Member
Quality Control	Ng Wen Jie
Inventory Management	Sit Yie Sian
Safety	Yong Xian Ye
Security	Teng Kai Deng

4.0 Lesson that we learned / Problem that we faced

As data science students embarking on this project, our journey into the realm of hardware and code presented us with a unique set of lessons and challenges. Our prior experience in data analysis and machine learning provided a strong foundation, but working with hardware and writing code for embedded systems was a new and enlightening experience.

4.1 Venturing into Hardware

- Learning Opportunity: Exploring hardware components, such as sensors and microcontrollers, allowed us to gain hands-on experience with physical data collection. This experience was a departure from our usual data analysis tasks, helping us appreciate the intricacies of sensor technologies and the physics behind data measurement.
- Challenges:
    - Steep Learning Curve: The world of hardware presented a steep learning curve. We had to become familiar with datasheets, which are often complex technical documents, in order to understand how to interface with sensors effectively. Learning about electronic circuits and hardware interfaces required dedication and study.
    - Hardware Debugging: Troubleshooting hardware issues was a novel challenge. Whether it was calibrating sensors for accurate measurements or addressing connectivity problems, it demanded a systematic approach. We had to develop patience in resolving hardware-related glitches.

4.2 Coding for Embedded Systems

- Learning Opportunity: Writing code for embedded systems introduced us to the realm of firmware development. We had to adapt our programming skills to the constraints and peculiarities of these systems, which was a valuable learning experience.
- Challenges:
    - Resource Constraints: Embedded systems often operate in resource-constrained environments. They might have limited memory and processing power. This necessitated efficient coding practices and optimization techniques to ensure the code ran smoothly on these platforms.
    - Real-Time Considerations: Ensuring our code ran reliably in real-time was a significant challenge. Timing became crucial, and we had to handle interrupt-driven programming and maintain responsiveness. Meeting real-time requirements while managing sensor data added complexity to our coding tasks.

4.3 Integration of Data Science and IoT

- Learning Opportunity: Integrating data science principles with the Internet of Things (IoT) allowed us to bridge the gap between theory and practical application.
- Challenges:
    - Data Variability: Environmental factors can make real-world data unpredictable and noisy. We had to adapt our data analysis techniques to handle this variability effectively. This meant exploring data filtering and outlier detection techniques to ensure data quality.
    - Data Streaming: Managing data streaming from multiple sensors and devices requires careful synchronization and data storage strategies. Dealing with data streams demanded a robust data pipeline and strategies to handle large volumes of data in real time.

4.4 Collaboration and Problem Solving

- Learning Opportunity: Collaboration was at the heart of our success, and it provided us with valuable insights.
- Challenges:
    - Interdisciplinary Communication: Effective communication within a multidisciplinary team was essential. We had to balance technical discussions with team members from different backgrounds, including hardware engineering and data science. This required clarity and patience to ensure everyone was on the same page.
    - Problem Solving: Tackling hardware and software issues as a team required strong problem-solving skills. We learned to approach problems systematically, drawing from each other's expertise. The willingness to learn from one another was crucial in overcoming complex challenges.

4.5 The Value of Persistence

- Learning Opportunity: This project underscored the importance of persistence and continuous learning.
- Challenges:
    - Overcoming Frustration: Frustration is a natural part of the learning process, especially when dealing with new and complex technologies. Overcoming setbacks and persisting through technical hurdles taught us resilience and the value of tenacity.

Our journey from data science students to IoT developers was marked by valuable lessons and challenging moments. These experiences expanded our skill set, broadened our horizons, and prepared us for the ever-evolving field of data science and IoT. It reinforced the notion that embracing challenges and embracing a growth mindset can lead to tremendous personal and professional growth.

5.0 Conclusion

In conclusion, our IoT solution for smart manufacturing represents a holistic approach to revolutionising the manufacturing environment. By combining Quality Control, Inventory Management, Safety, and Security modules, we have created a comprehensive system that enhances efficiency, product quality, workplace safety, and facility security.

Our design draws inspiration from successful smart manufacturing systems, benefiting from their lessons and innovations. The hardware and software architecture we've developed is both robust and flexible, ensuring scalability and adaptability in a rapidly evolving industry.

Through real-time monitoring and data analytics, our Quality Control module empowers manufacturers to detect defects, optimise processes, and reduce waste. The Inventory Management module ensures a smooth flow of materials and goods, preventing production delays and streamlining operations.

Safety is a top priority, and our Safety module leverages IoT sensors and wearable devices to create a safer working environment. It not only detects hazards but also provides immediate alerts and real-time tracking for workers, reducing accidents and improving emergency response.

In terms of Security, our system protects the manufacturing facility against unauthorised access and potential threats. It integrates surveillance, access control, and intrusion detection systems, ensuring the safety of assets and personnel.

Ultimately, our IoT solution for smart manufacturing represents a leap forward in the Industry 4.0 era, where data-driven decision-making, automation, and enhanced safety measures are paramount. With this comprehensive solution, we aim to empower manufacturers to excel in an increasingly competitive and technology-driven landscape, fostering growth, innovation, and sustainability.