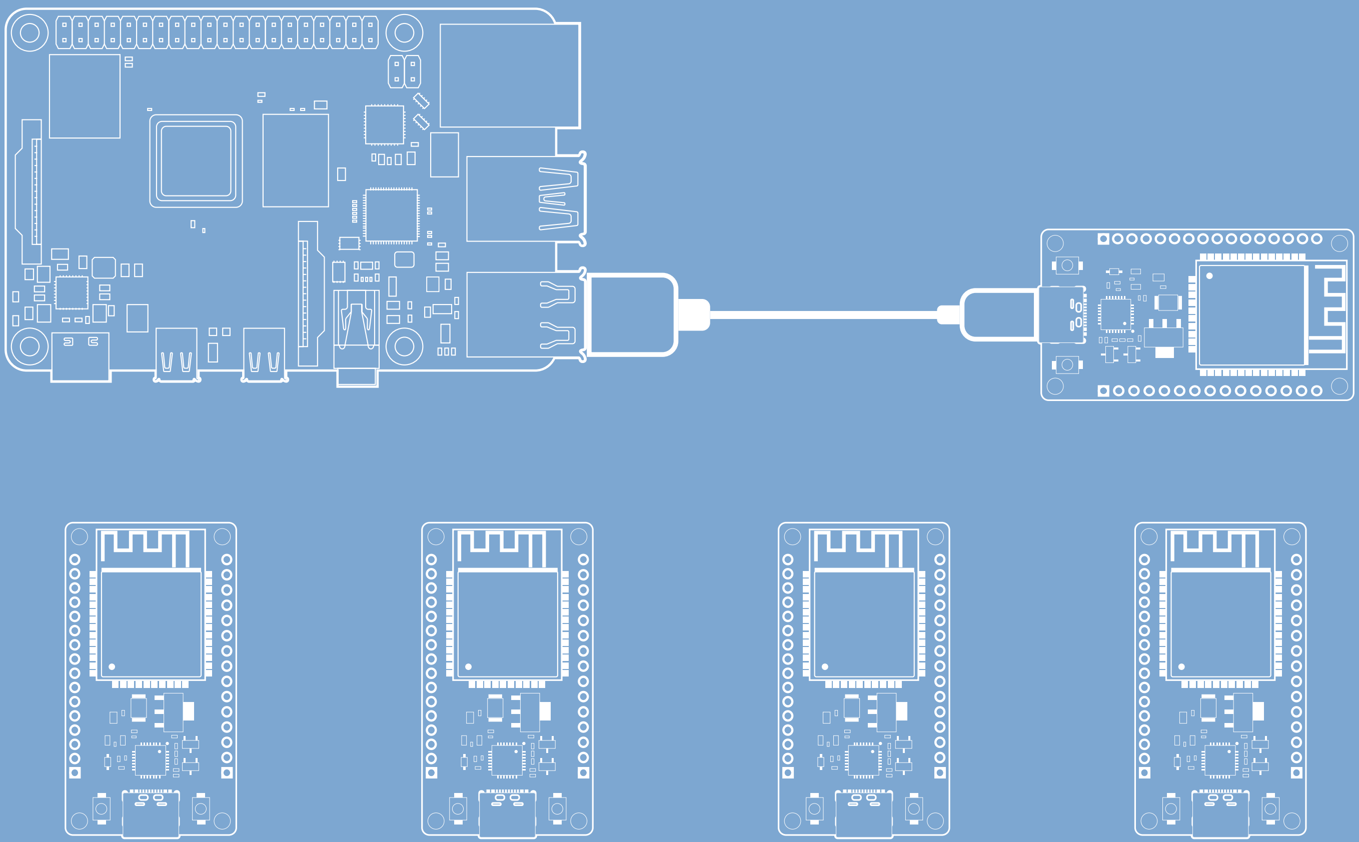
Privacy-oriented Wearable Data acquisition for MMLA

Wireless communication and data management

BAP 2024 Q1/Q2

Yasin Celem
Ryan Kolk
Darjan Stavrov



Privacy-oriented Wearable Data acquisition for MMLA

Wireless communication and data management

by

Yasin Çelem	5342538
Ryan Kolk	4931394
Darjan Stavrov	5380820

Date:	December 16, 2024
Project duration:	October 7, 2024 – December 20, 2024
Project supervisors:	Dr. B. Abdi,
	Dr.Ir. J. Dauwels
Faculty:	Faculty of Electrical Engineering,
	Mathematics and Computer Science,
	Delft University of Technology

TUDelft

Abstract

This thesis, conducted as part of the Bachelor Graduation Project at TU Delft, focusses on designing the communication and storage components of a data acquisition system for Multimodal Learning Analytics (MMLA). The goal is to create an adaptable system that improves learning outcomes in large, dynamic classrooms such as Tellegen Hall. Current systems often fail to address issues of privacy and integration, limiting their effectiveness in real-world applications.

The proposed design employs wearable dynamic nodes for data collection, and utilises static nodes and a root node for wireless communication. Wireless communication uses the ESP-NOW wireless protocol, and data is stored in a time-series database, InfluxDB, running on a Raspberry Pi 5. A Graphical User Interface(GUI) built in Grafana, allows monitoring and visualises the data. The system was tested for reliability and performance, showing a high success rate of 99.93% in data transmission. It enables a network consisting of, theoretically, up to 200 nodes.

Future improvements include real-time analytics and data security mechanisms.

Preface

This thesis is part of the Bachelor Graduation Project at TU Delft, to fulfil the requirements for the Bachelor of Science in Electrical Engineering. Proposed by Dr. Abdi and Dr. Dauwels, this project aims to develop an adaptable data acquisition system for MMLA research. This system needs to be adaptable to allow researchers to tailor it to their specific needs. The development of Machine Learning algorithms is beyond the project's scope, given its focus. The overarching goal is to enhance learning outcomes in large, dynamic settings such as Tellegen Hall.

We would like to extend our gratitude to Dr. Abdi and Dr. Dauwels for their supervision throughout the whole project, and for dedicating their time to provide essential support and guidance when needed.

We also acknowledge Mr. Lager for his effort in coordinating the project.

Yasin Celem
Ryan Kolk
Darjan Stavrov
Delft, January 2025

Contents

1 Introduction

Teachers often encounter difficulties in monitoring large, dynamic classrooms, particularly when students are mobile or when the number of students increases. The limited vision makes it difficult to maintain a broad overview of students and their behaviour, which is essential for providing personalised, targeted support. This issue mainly occurs in spaces such as Tellegen Hall at TU Delft (figure 1.1), as well as in other educational settings that involve practical tasks.

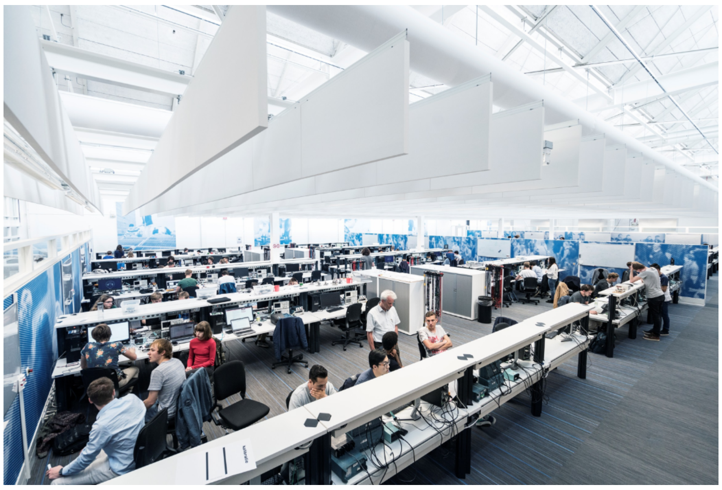


Figure 1.1: Overview of Tellegen Hall[1]

Despite the increase in educational tools, current technology has not yet been able to solve this problem. Interactive tools such as Kahoot! may improve engagement and learning experience, but they are too simplistic to automatically monitor students and accurately model their behaviour, as it is not their intended purpose.

Recent advancements in machine learning (ML) and sensing technologies have generated significant interest in their application within Multimodal Learning Analytics (MMLA). This domain involves the analysis of data collected from sensors using ML algorithms to monitor, understand and improve learning.

The field of MMLA introduces a new dimension to the educational setting by automating data collection and using ML for analysis. This approach proactively assists educators in identifying challenges, allowing for quicker and more accurate intervention. The goal is to enhance learning outcomes by identifying students who may require additional support, promoting higher level of engagement, and enhancing the overall educational experience.

These advancements present great potential, but they also raise important ethical and privacy concerns, related to the collection and use of sensitive data like video, audio, and physiological signals. Addressing these concerns is crucial for ensuring that the technology is both effective and responsible.

Therefore, **a privacy-oriented multimodal data acquisition system will be designed, tailored for large, dynamic learning environments**. The system will adhere to the ethical guidelines set by the Human Research Ethics Committee (HREC) at TU Delft, which is responsible for safeguarding participants' rights and privacy. This includes minimising the sensitivity of the collected data, in compliance with GDPR regulations.

The system does not currently employ Machine Learning. Instead, it functions as a foundational research product, designated to be adaptable for future modifications, allowing for adjustments or the integration of additional features to support research and development in this area.

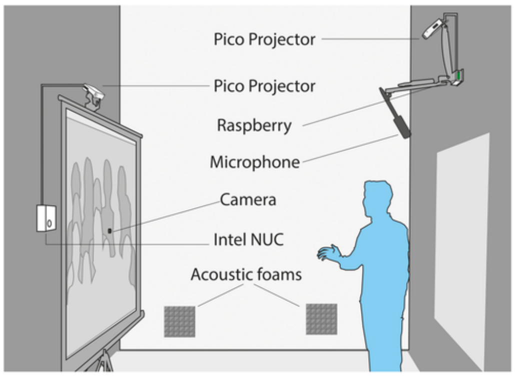
This chapter provides an overview of the project, beginning with a brief introduction to the field of MMLA (section 1.1). It addresses the current challenges and concerns in the field (section 1.2), followed by an analysis of existing systems (section 1.3), highlighting their shortcomings. This leads to the problem definition (section 1.4), followed by our proposed design as a solution to the problem(section 1.5). The final section (section 1.6) outlines the structure of the report.

1.1. Multimodal Learning Analytics

Learning analytics is an emerging field focused on understanding and improving education through analysing and visualising data. Traditional learning analytics tools have primarily concentrated on collecting data as a series of actions performed by an individual student, such as clicks or keystrokes on a computer [2]. This approach simplifies the presentation of data in charts or graphs, thereby assisting teachers in refining the learning process. These tools are usually integrated into existing digital systems, such as a computer. However, they are limited due to their reliance on unimodal data-data from a single source, such as video, audio, motion, or physiological signals [3]. As a result, these systems fail to capture the broader context and deeper interactions in classroom settings.

Conversely, Multimodal Learning Analytics (MMLA) addresses the shortcoming of traditional methods by integrating various complementary data sources. This integration provides a more robust foundation for identifying specific learning indicators and offers a thorough understanding of learning dynamics.

An example of MMLA in practice is the Automatic Presentation Feedback System (RAP), which illustrates how MMLA can improve oral presentation skills. In this scenario, participants present to a virtual audience while a camera, microphone, and slide-tracking software capture their performance. By extracting and analysing certain features, such as posture, vocal tone and slide text size, the system provides real-time feedback [4]. Similarly, another study [5] used motion sensors and video recordings to examine the impact of classroom surroundings on student collaboration. Features including participation levels, head movements, and interpersonal distances were extracted to analyse how table shapes affected group dynamics.



(a) Setup of the RAP system [4]

(b) Experimental setup of student collaboration study [5]

Figure 1.2: Examples of MMLA systems

To achieve its goal, an MMLA system needs to establish the learning indicator and its resulting behaviours. While indicators, such as student collaboration, are not measurable, related behaviours, such as speaking turns or eye contact are, which can be captured through data [6] Once these behaviours and their data traces are identified, the next step is to select modalities measure these behaviours effectively. For instance, audio can measure total speech duration [7], while video can capture gestures [8]. Selecting the right modalities ensures the collection accurate and meaningful data.

1.2. Challenges and concerns

Despite recent growth in MMLA research and advancements in sensing technologies and computational methods, MMLA has yet to reach its full potential in classroom settings. Two main challenges contribute to this gap.

- Firstly, teachers often lack experience with modern data collection and visualisation techniques, hindering their ability to use these tools in teaching [9],
- Secondly, existing MMLA systems may not align with teachers' instructional needs, as the educational field lacks an understanding on how to support teachers interpret and apply MMLA insights effectively [9]. As a result, systems are often designed without considering teachers' levels of technical knowledge, making them difficult to use for educators without technical training.

At present, not many MMLA systems effectively overcome these challenges by being both user-friendly and seamlessly integrable into classrooms without requiring technical expertise.

There is also added concern about the validity of results, as individuals change their behaviour when monitored. More obtrusive systems make natural behaviour less likely. Minimising obtrusiveness is desirable to obtain natural learning behaviour. However, the term ”unobtrusive” is interpreted in various ways across the literature, resulting in five distinct definitions [10]:

1. Interaction without main attention: Causes minimal disturbance to the user's routine.
2. Interaction using hard-to-notice hardware placement or design: Placing the hardware in subtle locations.
3. Interaction based on non-distracting data collection: Hardware might be visible, but data collection is non-distracting.
4. Interaction that is socially accepted: Designed to be socially comfortable and non-disruptive to others.
5. Interaction naturally integrated with the task: Does not alter original, natural interactions or experiences.

There is no universally accepted definition, leaving designers to prioritise aspects of unobtrusiveness. Restricting hardware design may limit the data collection. For example, subtle hardware placement might impede capturing full facial and body data. *Balancing unobtrusiveness with effective data-gathering is necessary.*

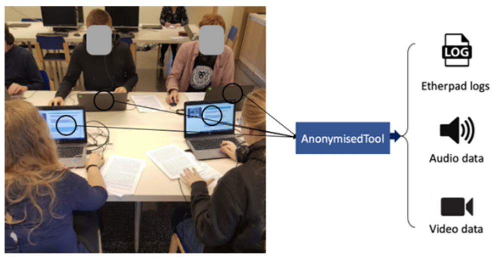
Last but not least, privacy concerns pose a significant challenge in MMLA research, particularly with data sensitivity and storage [11]. Researchers need ethics committees' approvals such as the HREC at TU Delft, to address potential risks and, comply with data protection laws such as GDPR. Despite this, there has been limited research on privacy and ethical considerations in MMLA [12], and few systematic frameworks exist to mitigate these risks [13].

While it may be challenging to create a universal approach to ethical concerns, existing MMLA systems have limited measures to minimise data sensitivity or prioritise anonymisation. *This gap indicates a need for systems that explicitly address these privacy risks, which could simplify the process of obtaining ethical approval and support broader adoption in educational settings.*

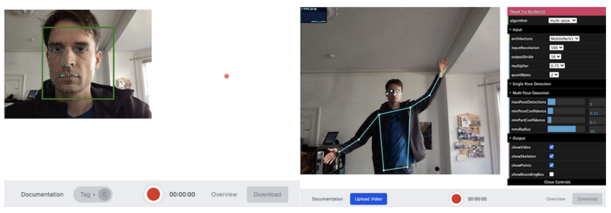
1.3. Available systems and tools

At present, there are several systems and research initiatives in the field of MMLA, though many of them are still in the research or prototype phase rather than being commercially available. These systems are still undergoing trials in various educational settings to assess their effectiveness. However, none of them fully address the issues highlighted in the previous section. By evaluating their design objectives and limitations, especially those encountered during data acquisition, we can gain useful insights. Since adaptability is an important requirement of the project, it will also be taken into account. The systems under consideration are as follows:

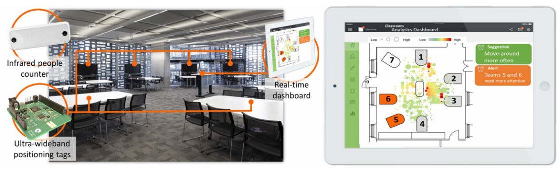
1. CoTrack - system or software that involves tracking or monitoring multiple entities and their interactions simultaneously through audio, video, and written logs. It features a graphical user interface for easy navigation. However, it lacks privacy considerations of the students during data collection [14];
2. EZ-MMLA toolkit – is a web-based application that uses video and audio recordings to generate data within the browser without transmitting sensitive information over the network. Data is collected and stored securely. While it has some privacy considerations, the collected data is sensitive. Additionally, integrating the toolkit into large, dynamic classrooms is complicated due to its reliance on computers for functionality[15];
3. Classroom prototype - system that provides a teacher with a heat map using localisation, proximity, and motion sensors. This allows the teacher to have an image of their interactions and shift attention to students who have not been visited recently. Even though privacy is not a primary concern, the collected data is less sensitive and excludes elements like video or audio. However, the system is not adaptable for broader research applications and operates similarly to an embedded system, dedicated to a single task [16];
4. Empatica E3 - a wearable data acquisition wristband that integrates 4 sensors (PPG, EDA, motion and temperature) to obtain valuable information regarding the individual's health. This device is lightweight and compact, making it minimally intrusive. However, the collected data is highly sensitive and may not be suitable for learning applications. Additionally, while learning applications might require different sensors, the system does not permit modifications [17].



(a) CoTrack [14]



(b) The EZ-MMLA toolkit [15]



(c) The classroom prototype with heat map [16]

(d) Empatica E3 [17]

Figure 1.3: Available tools and systems

The assessment of current MMLA systems highlights a need for a more efficient solution that addresses privacy concerns, adaptability, and integrability in large, dynamic classroom settings. Existing tools often rely heavily on static components like computers, collect sensitive data that may not comply with GDPR guidelines , or lack adaptability for diverse research purposes. The proposed system should address these limitations to ensure that the same data acquisition platform can evolve to meet various research needs while maintaining reliability and robustness.

1.4. Problem Definition

MMLA aims to improve teacher visibility in large, dynamic classrooms. An adaptable data acquisition system is needed that stores data for future use in MMLA applications. The system should be adaptable to allow for later adjustments such as the addition of extra sensors and users. Additionally, it should not limit itself to measuring specific learning indicators, but instead utilise sensors that can capture various data traces.

Furthermore, addressing the challenges and concerns associated with MMLA requires developing a system that focuses on privacy and integrates easily into large, dynamic classrooms. A privacy-oriented approach relates to mitigating privacy risks, mainly regarding anonymisation and collection of sensitive data. This aligns closely with HREC guidelines. For effective integration, the system should not solely rely on static components.

Therefore, the problem definition can be stated as follows:

How can a privacy-oriented data acquisition system for MMLA be designed that is adaptable and integrates easily into large, dynamic classrooms?

The system is not intended to provide a standardised framework but rather to offer an adaptable solution that researchers can adjust and build upon. It does not guarantee HREC approval, nor does it claim that the specific selection of sensors is universally suitable for all types of learning indicators. Such decisions should be made by educators and future researchers.

The main objective of the system is to provide reliable and robust data acquisition, with the capability to store data for future analysis. By focusing on privacy considerations, adaptability and integrability, the system can assist researchers in achieving their goals in large, dynamic classrooms, regardless of what those may be. As unobtrusiveness is an added concern, it will be defined here as ”data collection that occurs in a non-distracting way”, allowing some flexibility in data collection methods.

Considering the system’s adaptability, it is essential to provide clear performance specifications, such as data output rate, the total amount of data it can handle and the student support capacity. This will ensure that researchers are aware of its limitations and can tailor the system to meet their specific needs.

1.5. Design

The proposed design addresses all requirements outlined in section 1.4, with its structure illustrated in Figure 1.4. To effectively monitor students in large, dynamic classrooms, students and teachers will be equipped with small, wearable devices containing sensors that record data. These wearable devices are referred to as the system’s dynamic nodes. Since these devices handle all the data collection, it reduces the need for significant classroom modifications.

In large classrooms with many participants, all data will converge at a central point, known as the root node. The root node, which remains in a fixed location, communicates directly with a server to anonymously store the data for future analysis. Dynamic nodes transmit their data wirelessly to the root node when within range. To handle scenarios where direct communication is not possible, the classroom will also include relay devices called static nodes. These static nodes forward data from the dynamic nodes to the root node and record data to capture additional interactions. The placement of static and root nodes should be strategic provide wide network coverage and can be adjusted to the teacher’s needs.

This communication and storage infrastructure forms the system’s backbone, facilitating reliable data transmission and robust collection. The system must support the addition or removal of nodes without impacting its overall functionality, and maintaining ease of setup. The system requires a minimum configuration of 2 static and 2 dynamic nodes for demonstration purposes. A Graphical User Interface (GUI) will be developed to display system status information, including active nodes and incoming messages.

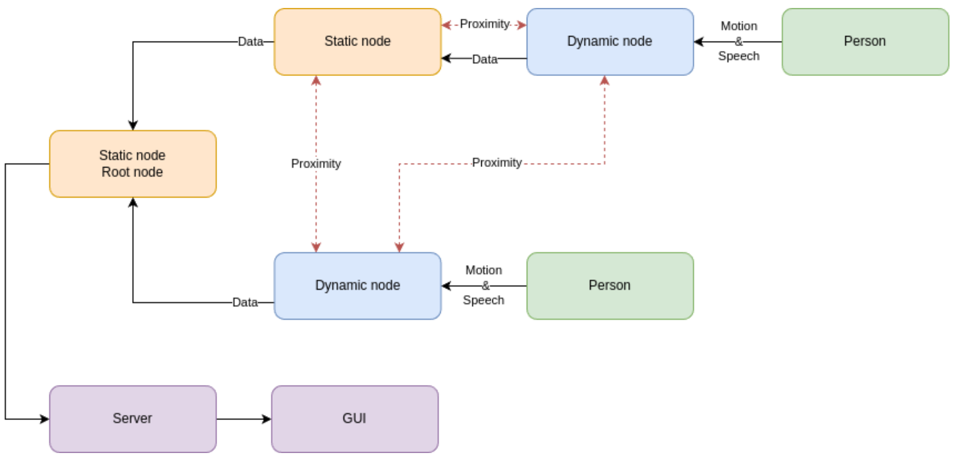


Figure 1.4: Black box diagram of the system

Each node in the system is essentially a microcontroller equipped with various sensors, capable of wireless communication and initial processing of sensor data. Once processed, the data is packaged into packets for transmission. Figure 3 illustrates this design in more detail. The selection of sensors should be in line with GDPR guidelines, avoiding collection of sensitive data. This includes pre-processed audio (where no identifiable information is stored), as well as non-intrusive sensors like proximity and motion sensors. A more detailed review of the sensor selection strategy will follow later. The static nodes only use proximity sensors, since they are not attached to talking individuals. To ensure adaptability, the system and data packages should accommodate the integration of additional sensors through allowing for a high data throughput.

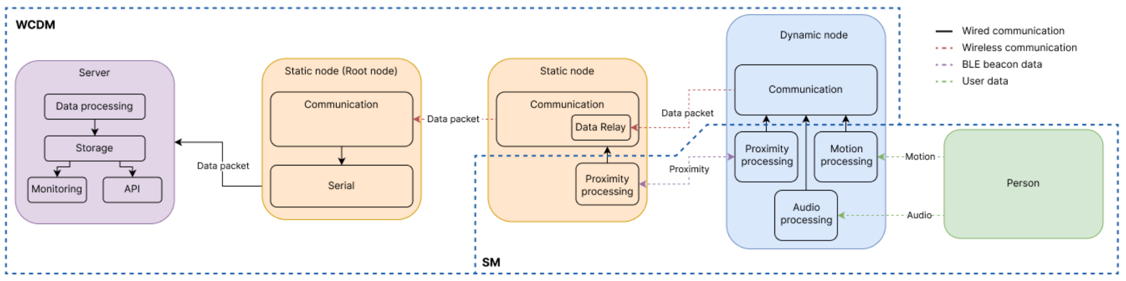


Figure 1.5: More detailed block diagram of system

The system is divided into two main modules: **Wireless Communication & Data Management (WCDM) and Sensors & Modalities (SM)**, each managed by a dedicated subgroup.

1.5.1. Wireless Communication & Data Management (WCDM)

The WCDM subgroup focuses on the communication between the nodes and the management of the data that is being transferred. The communication will be done wirelessly to keep the system unobtrusive. The subgroup must find suitable wireless technology that meets the requirements. At server level, the incoming data should be processed, stored, and monitored to allow for simple data extraction.

1.5.2. Sensors & Modalities

The SM subgroup is responsible for selecting sensors and modalities that can capture learning indicators. This selection must consider the privacy of the participants as it should not include sensitive data. The implementation of sensors and modalities should be kept as unobtrusive as possible. This will be a challenge for the dynamic nodes. The subgroup is also responsible for processing the sensor data and packaging it into data packets suitable for transmission.

1.6. Thesis overview

This report addresses the wireless communication and data management aspects of the project.

Chapter 1 provides an introduction to the project, including some background information, a state-of-the-art analysis, the problem definition and proposed solution.

The subsequent chapter outlines the specific requirements that the system must meet.

Chapters 3 and 4 detail the design choices made during the project: one focusses on the design of the wireless communication between nodes and the other addresses the server. This is followed by the implementation of the system and the results that validate its performance.

The discussion segment presents the significance of the findings and future work. The conclusion summarises the key points of the project.

2 Program of requirements

The specific requirements are established to develop a data acquisition system that measures student learning indicators according to the problem definition stated in chapter 1.

The Program of Requirements (PoR) uses labels for different requirements: **TA** for takeaways, **MR** for mandatory requirements, **TR** for trade-off requirements, and **BC** for boundary conditions.

2.1. Main PoR takeaways

- **TA1:** The primary objective of the system is to perform robust and reliable data acquisition and storage for MMLA purposes.
- **TA2:** The system should collect data in a non-distractive way.
- **TA3:** The system should store data ready to be further analysed.
- **TA4:** The system should be easily integrable for users in dynamic classroom settings.
- **TA5:** The system should implement a privacy-sensitive sensor selection strategy that minimises data sensitivity while obtaining useful data.
- **TA6:** The cost should be minimized when designing a data acquisition and storage system.
- **TA7:** The system should be designed as a foundational research product, intended to serve as a basis for further research.

2.2. General PoR

To meet the requirements of this product, the following guidelines should be followed. Both mandatory (hard) and negotiable (soft) are listed below. The TA-codes at the end of each line refer to the corresponding take-aways in section 2.1.

2.2.1. Mandatory requirements

The product must

- **MR1:** Extract data features from multiple sensors on the nodes. [TA1].
- **MR2:** Communicate the data features wirelessly between the nodes. [TA2]
- **MR3:** Receive and store the data features in the database. [TA1, TA3]
- **MR4:** Enable the extraction of the data in the database for use in MMLA. [TA1, TA7]
- **MR5:** Be extendible in terms of sensors and nodes. [TA7]
- **MR6:** Not have name-related user ID's. [TA5]
- **MR7:** Cost less than €500. [TA7]
- **MR8:** Operate shortly after being powered on, with no manual configuration required. [TA4]

- **MR9:** Should not require human intervention after setup. [TA4]
- **MR10:** Not collect sensitive data defined by the GDPR guidelines. [TA5]
- **MR11:** Provide a high processing performance to enable a high data throughput. [TA1, TA7]

2.2.2. Trade-off requirements

Also, the product should preferably

- **TR1:** Have little visibility when collecting data [TA2]
- **TR2:** Be light and small. [TA2, TA6]
- **TR3:** Be scalable in range. [TA7]
- **TR4:** Be usable for individuals without extensive technical knowledge. [TA4]
- **TR5:** Cost as low as possible. [TA6]
- **TR6:** Have low power consumption. [TA2, TA6, TA7]
- **TR7:** Take less than 20 minutes to setup. [TA4]
- **TR8:** Collect data in a non-distractive way. [TA2]

2.2.3. Boundary Conditions

The focus of this project is data acquisition and storage for MMLA purposes (**TA1**), which entails the following boundaries:

- **BC1:** The system will be tested using 2 static and 2 dynamic nodes. [TA1]
- **BC2:** Analysing data and reasoning using Machine Learning is out of scope. [TA3]
- **BC3:** Data security will be out of scope. [TA5]

2.3. Wireless Communication & Data Management PoR

In addition to the general PoR, there are specific requirements for the group Wireless Communication & Data Management group. These requirements are labelled by an extra **S**, with some of them being shared with the general requirements:

2.3.1. Mandatory requirements

The sub-product must

- **MRS1:** Collect and store data from nodes on server. [TA3, MR1]
- **MRS2:** Enable the extraction of the data in the database for use in MMLA. [TA1, TA7]
- **MRS3:** Have a server that is functional without internet. [MR1]
- **MRS4:** Not have name-related user ID's. [TA5, MR2]
- **MRS5:** Have wireless communication between the nodes. [TA2, MR5]
- **MRS6:** Support the addition or removal of devices without impacting overall functionality. [TA1, MR1]
- **MRS7:** Achieve a 99.9% success rate in communication between any two nodes. [TA1, MR1]
- **MRS8:** Be expandable in terms of quantity of nodes. [TA7, MR3]
- **MRS9:** Operate shortly after being powered on, with no manual configuration required. [TA4, MR8]
- **MRS10:** Should not require human intervention after setup. [TA4, MR9]
- **MRS11:** Provide a high processing performance to enable high data throughput. [TA1, MR11]

2.3.2. Trade-off requirements

Also, the sub-product should preferably

- **TRS1:** Have low power consumption. [T6]
- **TRS2:** Be low cost. [M6, T5]
- **TRS3:** Have light and small nodes. [TA2, TA6]
- **TRS4:** Be scalable in range [TA7, TR3]
- **TRS5:** Be usable for individuals without extensive technical knowledge. [TA4, TR4]
- **TRS6:** Take less than 20 minutes to set up. [TA4, MR7]

2.3.3. Boundary Conditions

- **BCS1:** The system will be tested using 2 static and 2 dynamic nodes. [TA1]
- **BCS2:** Analysing data and reasoning using Machine Learning is out of scope. [TA3]
- **BCS3:** Data security will be out of scope. [TA5]

3 Design: Wireless Communication

The complete design for data communication and management includes two main components that can be developed separately: the wireless communication system and the server. The wireless communication component is responsible for transmitting the data to a designated storage location for future analysis. This chapter outlines the design choices related to this aspect, including the selection of hardware, the communication protocol, and the implementation approach.

3.1. Design overview

Figure 3.1 provides an overview of the design, highlighting the communication architecture. The dynamic nodes wirelessly transmit data collected from sensors to the root node. Static nodes serve as relays, for out of range nodes. The root node establishes the connection with the server and uploads the data to the server. Microcontroller development boards handle all functionalities, so no other hardware is needed. The specific role of each microcontroller board, whether static, dynamic or root, is defined by its ability to either relay messages or communicate with a server or neither. As the server communication depends on the server's design, it will be discussed in chapter 4.

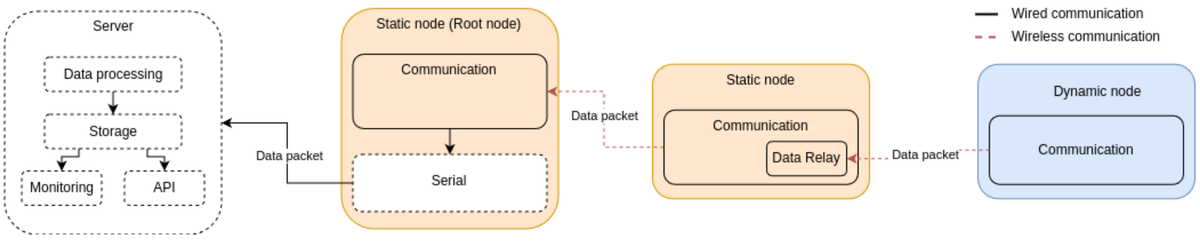


Figure 3.1: Diagram of communication system

3.2. Microcontroller development board

The microcontroller development board is a critical component of the design, required to perform multiple functions. Besides handling wireless communication, it also needs to perform the tasks assigned to the other subgroup. Specifically, it should be able to integrate sensors, process the collected data, and package it into data packets. Fast processing is essential to manage all these tasks simultaneously. To ensure compatibility, both subgroups will utilise the same type of board. The selection process will be guided by a shared set of requirements. Key criteria for this selection include cost, size, power and processing performance [18][19], defined as TRS1, TRS2, TRS3 and MRS11.

The cost, size, and weight of the development board typically refer to the entire board, which includes components such as voltage regulators, oscillators, connectors, and debugging interfaces. On the other hand, processing performance and power consumption are primarily dictated by the microcontroller (MCU) itself.

Processing performance
Processing performance is measured by the time required to complete a specific task [20]. Faster execution means better performance, influenced by the microcontroller's clock frequency; higher frequencies results in quicker task execution [21]. Thus, high-performance microcontrollers have high clock frequency to enable efficient data handling and rapid computations.

Power consumption
The power consumption of a microcontroller is influenced by its operating supply voltage and clock rate. Higher operating voltage and clock rate increase power consumption, but a lower clock rate negatively impacts performance, as tasks take longer to complete. Therefore, balancing between performance and power efficiency is essential. Since the microcontroller needs to handle multiple tasks simultaneously, selecting one that operates at a low voltage while maintaining a high clock rate is essential to ensure both power efficiency and sufficient processing capability [22].

Commonly used microcontroller development boards include the ESP32-DEV-38P [23], Arduino Nano 33 IoT [24], Raspberry Pi 5 [25] and STM32F103C8T6 Blue Pill [26][27]. These boards will be evaluated and compared based on the specified requirements presented in Table 3.1.

	ESP32-DEV-38P	**Arduino Nano 33 IoT**	**Raspberry Pi 5**	**STM32F103C8T6**
Size	1.98×5.99×1.80 cm	4.13×1.78 cm	8.50×5.60×1.26 cm	2.29×5.33 cm
Cost	€9.51	€28.07	€56.99	€10.95
Operating Supply Voltage	3.3 V	3.3 V	5.1 V	3.3 V
Processing Speed	240 MHz	48 MHz	2.4 GHz	72 MHz

Table 3.1: Comparison of microcontroller development boards

The Raspberry Pi 5 offers the best processing performance among the options but has higher power consumption, size, and cost, making it less suitable for this application. On the other hand, the ESP32-DEV-38P provides a solid 240 MHz clock speed, consumes less power, and is compact and affordable. This makes it the ideal choice, as it strikes the perfect balance between all requirements: TRS1, TRS2, TRS3 and MRS11.

Figure 3.2: Image of ESP32-DEV-38P [28]

3.3. Communication

To adhere to MRS5, nodes must communicate wirelessly. Various factors should be considered when selecting a communication method such as range, bandwidth, power consumption, interference with other components, latency, topology, and reliability to name a few. Each option involves a trade-off; for example, Bluetooth allows for low power consumption, but this comes at the cost of range and bandwidth. On the other side, a Wi-Fi based implementation allows for higher bandwidth and range, but this at the cost of power usage, and flexibility in topology. Therefore, it is important to evaluate multiple options and understand their strengths and weaknesses. The most important metrics for this system based on the requirements are low power (TRS1), high bandwidth(MRS11), and reliability (MRS6, MRS7).

3.3.1. Wi-Fi

Wi-Fi is widely used for high-bandwidth wireless communication, offering a transfer rate of up to 20 [29] Mbps on the ESP32, making it suitable for high throughput applications. Its support for acknowledgements ensures reliable communication, an important feature in environments where data integrity is important. However, this reliability comes with higher power consumption [30], making Wi-Fi less optimal for battery-powered devices.

3.3.2. BLE Advertising

Bluetooth Low Energy (BLE) [31] advertising allows for minimalistic and energy efficient wireless communication, making it suitable for certain IoT applications. This efficiency comes with constrains, as the packet size is limited to 31 bytes. The lack of acknowledgement support affects its reliability in scenarios that require strong data integrity. Therefore, BLE advertising is most appropriate for applications with minimal data transmission requirements.

3.3.3. BLE Mesh

BLE Mesh [32] provides both power consumption and reliable data transfer. With a bandwidth of 1 Mbps, it meets the requirements for many IoT applications requiring moderate data exchange. However, the maximum packet size of 11 bytes requires data fragmentation of payloads. Despite this, BLE Mesh remains an excellent option for applications where reliability and energy efficiency are prioritized.

3.3.4. ESP-NOW

ESP-NOW [33] is a protocol developed specifically for ESP devices designed to optimise efficiency, reliability, and speed. With a maximum packet size of 250 bytes, bandwidth of 1Mbps, and a peer-to-peer communication strategy, it offers flexibility for various network topologies such as mesh networks. Its low power consumption and optimized implementation makes it an optimal choice for applications requiring the exchange of moderately sized packets of real-time sensor data.

3.3.5. ZigBee

ZigBee[34] is a protocol designed for low-data-rate, low-power applications. With a bandwidth of 250 kbps, it prioritizes energy efficiency above bandwidth, making it a preferred choice for IoT systems like smart lighting and environmental sensors. Its maximum packet size of 100 bytes allows for small to moderate payloads, while support for acknowledgements and mesh networking improves reliability. However, its relatively low bandwidth may present an issue for real-time data transmission from many devices. ZigBee therefore excels in situations where energy efficiency and network scalability outweigh transmission speeds.

	Bandwidth	**Max packet size**	**Power usage**	**Acknowledgements**
Wi-Fi	20 Mbps	1500 Bytes	High	Yes
BLE advertising	-	31 Bytes	Low	No
BLE Mesh	1 Mbps	11 Bytes	Low	Yes
ESP-NOW	1 Mbps	250 Bytes	Low	Yes
ZigBee	250 kbps	100 Bytes	Low	Yes

Table 3.2: Comparison of wireless protocols

Based on Table 3.2, the decision was made to implement mesh functionality using the ESP-NOW protocol. The protocols peer-to-peer nature, ensures low power (TRS1) and low latency. The protocol has a relatively high data bandwitdh (MRS11). Additionally, a mesh implementation on top of the wireless protocol fulfils requirement MRS6 and trade-off requirement TRS4.

3.4. ESP-NOW

To implement the mesh logic in ESP-NOW the open source library, ZH Network [35] was selected, as it includes the required mesh logic, delivery confirmation and automatic resend logic. Although this library comes packed with features, changes are necessary to achieve the required functionality.

3.4.1. Routing

Each node maintains its own routing table. The routing table is updated by broadcasting a route request to a MAC address. When this occurs, nodes within range update their routing tables to include the route to the requesting node, as well as rebroadcasting the message if they are designated as relays (TRS4). Once the target device is reached, it responds with a unicast message back to the original MAC-address, and uses the routing table to select either the MAC of the original node or a relay MAC if necessary. By default ZH network designates all devices as relays, so by simply disabling the ability to respond to, and rebroadcast route requests for messages not directly addressed to the devices removes the possibility that devices will show up in routing tables as relays.

3.4.2. Topology

The proposed mesh network consists of three types of nodes:

- **Root node**: This node acts as the endpoint for receiving all messages intended for data collection and communicates with the ingest server.
- **Static nodes**: These devices serve as relays, transmitting for messages to the root node in order to extend the range of the network.
- **Dynamic nodes**: These devices, intended to be worn by the user, collect sensor data and pass these data to the root node.

3.4.3. Messages

The message structure consists of a header and a body. The header contains all data related to the sender, recipient, and message type and length. The body contains the actual data being transmitted. The message type defines the structure of this data.

3.4.4. Time synchronization

Given that many devices will be contributing to a central database, it is important to synchronize the clock of the nodes to satisfy MRS1. To achieve this, the network must have the capability to dispatch out some sort of synchronization signal. Two main implementations were considered: Network Time Protocol(NTP) and Flooding Time Synchronization Protocol (FTSP)

FTSP

Flooding Time Synchronization Protocol [36] works by broadcasting a timestamp from the root node, which sets the time for the other nodes. This method is simple, but has a drawback of propagation delays. The chosen network implementation includes processing times in the tens of millisecond range. although these processing times can be partially accounted for, there is a protocol that addressed both propagation and processing delay.

NTP

The Network Time Protocol [37] is the industry standard for synchronizing time across devices globally. Synchronization is achieved by the client sending out a synchronization request that includes the local time on the device. The server responds with the original time, the local time of the server when the request was received and the time at transmission. Upon receipt of the response, the client calculates the time offset using Equation 3.1, and adjusts its time accordingly. This method, as depicted in Figure 3.3, accounts for the round trip time of the packages. However, one limitation is the assumption that the transmission times are equal, which may introduct minor inaccuracies if this condition is not met.

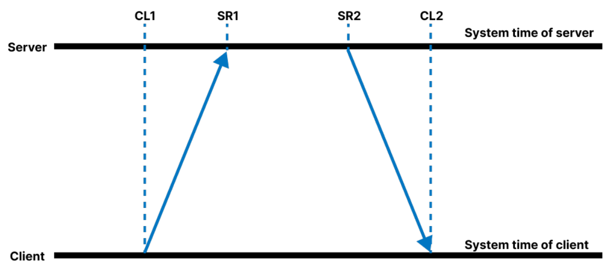


Figure 3.3: Network Time Protocol

$$offset = \frac{(sr1 - cl1) - (cl2 - sr2)}{2} \tag{3.1}$$

NTP was selected for this system due to its straightforward implementation, and its ability to adjust for transmission delays. The root node functions as the time authority, with all other nodes synchronizing their time by querying the root node.

4 Design: Server

The server is the core of the system, processing and aggregating network data for export. It includes data ingress, data storage, an API, and a GUI. This chapter explains the design choices and hardware tasks for these elements.

4.1. Design overview

Figure 4.1 provides an overview of the design, highlighting the server architecture. The root node collects data from the network and transfers it to the server. The server runs on a different microcontroller board than then the communication network, because it is based on the selected database. The transmission of data between the root node and server board is called the data ingress process and is the main challenge of the design. Since the incoming data is time-stamped, the system is in need of a database that can handle this. To communicate with this database, an Application Programming Interface is added. The Graphical User Interface offers a graphical display to monitor the status of the system.

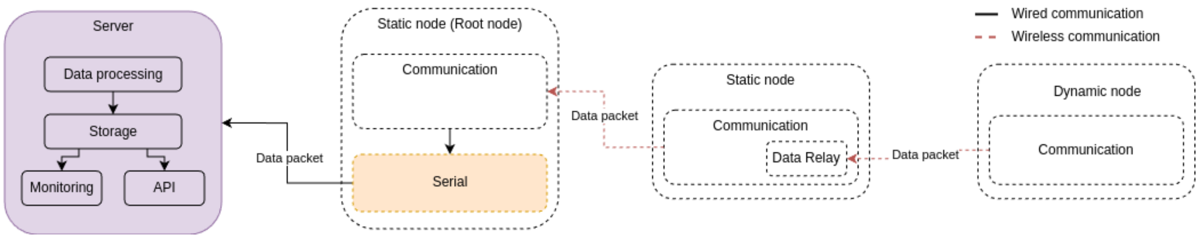


Figure 4.1: Diagram of Server part of the system

4.2. Data ingress

The data ingress process ensures that data collected from the wireless network is reliably transmitted from the root node to the server. To achieve this, a suitable programming language, serial communication protocol and data transfer method need to be selected. Additionally, implementing effective error handling through a checksum is essential to ensure data integrity. This section discusses the design decisions, the rationale behind each choice, and how these choices contribute to the reliability and efficiency of the system.

4.2.1. Programming Language

The server needs to be designed to act as an intermediary between the nodes and the place to store data. The server has to operate offline (MRS3), handle incoming data from additional nodes as the system scales(MRS6), have a high success rate in data storage (MRS7) and have a high processing performance to enable high data throughput (MRS11). Table 4.1 presents a comparison of several programming languages.

Language	Pros	Cons	Suitability
Go	Fast execution, built-in concurrency, easy deployment	Limited generics, relatively new	Great fit for handling multiple incoming data streams and efficient error handling
Python	Easy to write and read, rich library support	Slower execution, high memory usage	Not ideal due to its slow speed for real-time data handling
Java	Platform independence, robust libraries	Higher memory consumption, complex setup	Suitable, but more complex than Go for deployment purposes
C++	High performance, fine control over resources	Complex to code, manual memory management	Too complex and prone to memory issues for this use case

Table 4.1: Comparison of Programming Languages

Why GO?

Go is known for its simplicity, ease of deployment, and native support for concurrency through goroutines, which is useful for handling multiple data streams (MRS6), while also offering significantly higher speed compared to Python, which is essential for real-time data ingestion (MRS11). Also, the performance nearly matches that of C++ with reduced complexity and simplified but successful memory management (MRS7).

4.2.2. Communication Protocol

The decision to use a serial connection, such as I2C and UART, was determined by its simplicity and reliability, as indicated in Table 4.2.

Protocol	Pros	Cons	Suitability
UART	Simple implementation, longer range support	Lower speed compared to I2C	Suitable for longer distance communication between root node and server
I2C	High-speed communication, multiple devices	Short range, complex addressing scheme	Suitable for short-range, multi-device setups

Table 4.2: Comparison of Communication Protocols

Why UART?

UART communication is preferable due to its simplicity and capability to cover longer distances reliably without interference, which is beneficial in a distributed classroom setting (TRS4).

4.2.3. Data transfer method

Data can be transferred in a continuous data stream, with the root node transmitting data in real-time, or in data bundles, where data is aggregated into batches before being sent.

Method	Pros	Cons	Suitability
Continuous Streaming	Low latency, immediate data availability	High server load, increased risk of packet loss	Suitable for cases needing real-time reactions
Data Bundling	Lower overhead, reduced transmission errors	Slightly increased latency	Ideal for minimizing errors and efficient batch processing

Table 4.3: Comparison of Data Streaming vs. Data Bundling

Why Data Bundling? Data bundling provides a better balance between efficiency and reliability. By sending data in defined bundles, errors can be detected and corrected more easily, reducing the burden on the server and making the processing more predictable (MRS7).

4.2.4. Checksum Selection

To ensure for data integrity, a checksum has to be introduced. CRC-16 and XOR are both viable methods for checksum calculations, with Table 4.4) showing their pros and cons.

Method	Pros	Cons	Suitability
CRC-16	Strong error detection, detects burst errors	Computationally heavier than XOR	Ideal for ensuring data reliability in a noisy environment
XOR	Simple, fast computation	Weak error detection capabilities	Suitable for scenarios with lower reliability requirements

Table 4.4: Comparison of Checksum Methods

Why CRC-16? CRC-16 is significantly more effective at detecting burst errors compared to XOR, making it suitable for environments where data integrity is crucial (MRS7).

4.2.5. Design Considerations

Several parameters need consideration during the design of the data ingress process:

- **Command System**: The command system was designed to maintain reliable communication between the root node and the server. It is important for ensuring the reliability of data transmission, minimizing the risk of errors like misinterpretation due to bit flips, and supporting smooth and clear data flow.
- **Batch Processing**: Batch processing has to be considered to reduce the number of interactions with the database, providing a balance between system overhead and latency. By processing data in batches, we can optimize resource use and ensure timely data analysis without overwhelming the system.
- **Error Handling and Time Synchronization**: Error handling is essential for ensuring data reliability, allowing corrupted data to be retransmitted as needed. Time synchronization was also included to maintain consistent timestamps, which is critical for accurately correlating data from multiple nodes.
- **Parsing**: Data parsing is another key consideration, as raw data must be converted into a structured format that is usable for analysis and storage. Parsing ensures that the data can be effectively queried and stored in a database, supporting the overall goals of reliability and efficiency. To make the parsing successful, the size, number of messages, readability, and accessibility of the data have to be considered.

4.3. Data storage

Selecting a data storage method is a critical part of this system. Since the data consists of time-based measurements, a time-series database was ideal. These databases are specifically designed and optimized for storing, querying, and processing time-stamped data, making them perfectly suited for this application. Several options were evaluated, based on the requirement MRS11.

Prometheus
Prometheus[38] is widely used database for monitoring and alerting. Originally developed for server and system monitoring, Prometheus is a robust, scalable and employs its own query language PromQl real-time data ingestion.

InfluxDB
InfluxDB[39] is a purpose-built time-series database designed for high performant data ingestion. Its query language, Flux, allows users to easily extract specific data with a broad set of tools for downsampling, interpolation and analysis, among others. These tools enable the customisation of datasets to fit various use cases.

TimescaleDB
Built on top of PostgreSQL, TimescaleDB[40] uses an industry standard traditional database and transforms it into a powerful time-series database, offering SQL querying. One consideration is that TimescaleDB is built for large scale deployments, which may exceed the requirements of this project.

The Decision
InfluxDB was selected for its ease of integration, extensive documentation and examples, and high performance. Its capability to handle complex queries efficiently also made it the preferred database. Most importantly, InfluxDB meets requirement MRS11 with a higher performance with a small scale deployment than the other options.

4.4. Hardware

The root node in the system acts as the central hub, collecting data from the network. It communicates this data to a server running InfluxDB. The server board is responsible for managing the ingress of information and facilitates data storage. To be able to do this, the selected board should have sufficient memory and should balance processing performance with power consumption. Since data storage is the primary function of the server, memory capacity is the most critical requirement. Cost is not a concern because there is only one root node in the system.

Memory: To handle time-series sensor readings effectively, the microcontroller must have adequate RAM to store and manage the incoming data temporarily. An increase in data ingestion rate, requires additional RAM; otherwise, it may lead to significant delays or even lost data packages (MRS7).

A number of single board computers were compared, with a focus on cost to performance.

	BeagleBone Black	**ODROID-H4**	**Raspberry Pi 5**	**ASUS Tinker Board S R2.0**
RAM	512MB	8 GB	4 GB	2 GB
Cost	€44.97	€123.14	€56.99	€130.58
Processing Speed	1 GHz	3.6 GHz	2.4 GHz	1.8G GHz

Table 4.5: Comparison of single board computers

From table 4.5, the selection has been narrowed down to two options: the Raspberry Pi 5 and the BeagleBone Black. While the BeagleBone Black offers superior performance, the Raspberry Pi 5 provides a great cost to performance. Because of the superior cost to performance, Raspberry Pi 5 is identified as the more appropriate choice.

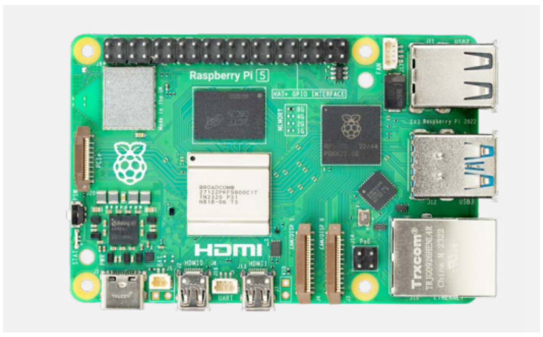
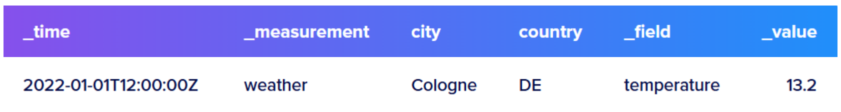


Figure 4.2: Image of Raspberry Pi 5 GB4 [7]

4.5. API

An Application Programming Interface (API) allows communication between two or more software systems, using a programming language to return a requested data set. This process is not visible to the end user, but only to the systems that are connected to it [41]. InfluxDB includes a built-in API using Flux, which simplifies sending requests without creating an API from scratch. Understanding the database structure allows for constructing queries in Flux to retrieve the desired data set.

InfluxDB organises time series data into a structured hierarchy of buckets, measurements, tags, fields, and timestamps. Buckets serve as storage locations that logically group various measurements, such as audio or proximity data. The tag key is used to identify the data source while the tag fields describe characteristics of a measurement whose values change over time. For example, audio can be described by attributes like tone or speech duration. Each measurement point contains a field value.



_time	_measurement	city	country	_field	_value
2022-01-01T12:00:00Z	weather	Cologne	DE	temperature	13.2

Figure 4.3: Database structure [42]

To retrieve data from the database, three main functions are commonly used: from, range and filter. The from function specifies the bucket, the range function defines a time interval and the filter function refines the query by filtering the specific tags and fields. By combining these 3 functions, any point in the database can be accessed [42].

4.6. GUI

The Graphical User Interface (GUI) offers a method to monitor the entire system. By accessing the database, it clearly visualises all data for the user. The GUI is essential for verifying the proper function of a data acquisition system, displaying real-time measurements of specific users and the number of connected nodes to the system.

InfluxDB offers various data visualization options, including its UI platform and external platforms like Grafana and Kapacitator [42]:

- **Influx DB UI:** simple tool for visualising InfluxDB data
- **Grafana:** advanced tool for visualising data from multiple databases
- **Kapacitator:** visualisation tool that focuses on real-time data processing

Kapacitator may be unnecessary for this project as most of the processing is done beforehand and advanced, complex visualisation is not required. Displaying the incoming measurements and number of connected nodes is sufficient to track the state of the system. However, to comply with TA7, future researchers may integrate new sensors and databases. In such cases, Grafana offers immediate support for these additions, making it the most suitable option.

5 Implementation and Validation

5.1. Wireless communication implementation

5.1.1. Architecture

There are three distinct roles within the mesh network: A root node, static nodes, and dynamic nodes. Each of these roles has specific functionality and sometimes overlapping functionalities divided into modules that can be enabled and disabled.

- **Communication**: Since each node is part of the network, all nodes must implement this logic, including communication, as well as time synchronization.
- **Relay functionality**: Static nodes should be capable of relaying messages between dynamic nodes and the root node. This is a submodule of the communication module.
- **Serial communication**: The root node has the added responsibility of communicating with the server, relaying all the relevant data over serial.
- **Sensor task**: Dynamic nodes should be capable of gathering data from their onboard sensors and send it o the communication stack. (This part is provided by the Sensors and Modalities group)

Given that a substantial amount of logic is shared across various modules, it was decided to integrate these modules within a single codebase. Feature flags are used to activate relevant features. This approach also facilitates the addition of test modules, allowing for easy testing and validation of specific modules.

5.1.2. Nodes

The node logic implementation was developed in C++ using PlatformIO[43], and embedded software toolkit that integrates into VS code[44]. Due to the significant amount of code involved, it will be made available on GitHub (https://github.com/ryankolk/BAP-2024) rather than included as an appendix.

Communication

As explained in chapter 3, data is transmitted using the ESP-NOW wireless protocol. ZH_network[35] supports unicast(one-to-one) and broadcast(one-to-many) communication. Most data will transmitted unicast to and from the root node, but routing tables are maintained through broadcasts. Appendix A illustrates the inner workings of this functionality.

Messaging

Each message sent through the network contains a message header and a body. The header includes the message type, node ID, and current timestamp. The message body is defined by the message type and contains the information relevant to the specified message type. The message types are as follows:

- **Sync request**: Initiating time synchronization.
- **Sync response**: Contains the response to a time synchronization.
- **Reset Time**: A message type that enables the root node to compel other nodes to synchronize their clocks. This message has no body and can be sent either as broadcast to the entire network or as unicast to a single node.
- **Data**: This main data package transmits collected sensor data for database storage. It is the only message type exits the mesh network and is forwarded to the server.

Time synchronisation
When the nodes initiate, their first task is to attempt synchronisation with the root node by sending the sync request message. Upon receiving this request, the root node will respond if its own clock is already synced, which is done through the communication channel with the server. However, if the root node's time has not yet been set, it will not respond to the synchronisation request. In such cases, the nodes will retry synchronisation every 5 seconds until a response is received. Once synchronisation is successful, subsequent attempts will occur once every hour to compensate for any potential drift.
The root node also has the ability to force a resynchronisation. This functionality is employed under two specific scenarios:

- If the message header of a node exhibits a drift exceeding 1 second, it indicates that the clock is likely out of sync, as there should be no instance where transmission takes longer than 1 second. This message will be transmitted directly to the affected node.
- When the root node's clock is updated, and there is a drift of more than 1 second compared to the server, the message is broadcast to all devices.

5.2. Server implementation

5.2.1. Data Ingress

The data ingress system is based on a serial communication protocol and is implemented with Go code. The ESP32 microcontroller functions as the root node, collecting data from various nodes using the serial protocol and transmitting it to the server. The server, also implemented in Go, manages the communication process, ensuring data validity, parsing, and storage. The Go program handles the serial communication by repeatedly requesting data from the ESP32, verifying its integrity using checksums, splitting it into readable components, and storing the processed data in a database for further analysis

Serial Connection Setup

The serial connection setup between the root node and server is fundamental for data transmission. The ESP32 is configured to collect data from various sensors, including both static and dynamic nodes. The serial communication between the ESP32 and the server is established at a baud rate of 500000, with a read timeout of 100 millisecond. This read timeout is automatically set due to a limitation of the library in GO. The baud rate of 500000 was selected after analysing various transmission speeds and error rates across different baud rates (e.g., 115200, 250000, 500000, and 1,000,000). It provided an optimal balance with fast data transfer to prevent bottlenecks during high-volume traffic and minimal error rate. Higher speeds like 1,000,000 caused message bundles loss, while the lower baud rate of 115200 was insufficient for real-time data transmission, causing significant latency in data collection. Testing demonstrated that at 500000, the ESP32 could reliably transmit data bundles of up to 256 bytes with an average error rate of less than 0.1%. The 100 milliseconds read timeout from the Go library, maintained steady data flow however it did introduce a higher latency, as the ESP32 average response time was considerably below this time threshold. These configurations ensured correct data collection and transmission, while the accuracy and system reliability were maintained, The limitation introduced by the GO library limited data throughput.

Data Request and Reception

The process of data collection starts when the server sends a start byte (0x25) to the ESP32, signaling a request for data. The ESP32 responds by sending data in chunks of up to 256 bytes, which are received by the server. The server continues reading the response until no more bytes are available to read, ensuring that the entire data payload is received.

The command bytes (e.g., 0x25 for data request, 0x07 for acknowledgment, 0x19 for resend) were selected to minimize the likelihood of bit-flip errors leading to incorrect commands. These distinct values make it less probable for corrupted data to be misinterpreted for another valid command. Upon receiving a complete response, the server extracts the expected data length and performs a checksum verification. If the checksum is correct, the data is validated, parsed, and an acknowledgment byte (0x07) is sent to the ESP32 to confirm successful data reception. In case of a mismatch or data error, the server sends a resend request byte (0x19) to the ESP32, requesting retransmission.

Message structure

The data bundles transmitted from the ESP32 include several components that maintain data integrity and ensure appropriate processing by the server. Each message consists of:

1. Header and Length Information: A header indicating the bundle type and a length field specifying the size of the data.
2. Sensor Data: The main payload containing sensor readings collected from various nodes.
3. Checksum: A checksum value calculated using a CRC-16 algorithm to verify the integrity of the data.

The server recalculates the checksum for the received data and compares it to the provided checksum to validate the bundle. If the calculated checksum matches the received checksum, the data is deemed valid. Furthermore, the checksum polynomial used for the CRC-16 calculation is carefully chosen for strong error-detection capabilities, making it highly effective at identifying errors during transmission. This ensures that even minor transmission errors are reliably detected.

Batch Processing and Data Management

The system is designed to aggregate valid data messages into a queue for batch processing. Up to 75 valid messages can be combined into a batch and subsequently stored in the database.

The batch size of 75 was chosen to ensure a timely transmission over UART. Larger batches would increase throughput, but would also be memory and processing intensive on the root node, whereas smaller batches would alleviate some of the memory overhead, but limit the throughput. This batching approach reduces the frequency of server-root node interactions and ensures that only complete and verified data is stored for further analysis.

Error handling and Time synchronization

The system includes a resend protocol to manage potential transmission errors. If data corruption is detected or no response is received following a data request, the ESP32 will resend the data up to three times. The server will acknowledge successfully received data or requests retransmission if necessary.

Time synchronization between the ESP32 and server is critical for ensuring consistent timestamps across the collected data. Periodically, the server sends a time synchronization command byte (0x11) to the ESP32, along with the current timestamp in seconds and microseconds. The ESP32 acknowledges with an acknowledgment byte (0x07) to confirm synchronization. This synchronization ensures that all collected data can be accurately analysed with respect to time.

Server Side Data Parsing and Storage

The server, developed in Go, is responsible for parsing incoming data bundles, verifying their integrity, and then storing the verified data in a time-series database. The parsing process involves extracting data length, recalculating and comparing the checksum, and ensuring that the data meets expected formats. Parsing is crucial for converting raw data into a structured format suitable for analysis and storage.

The server utilizes APIs to efficiently store and retrieve data from the database for future analysis. Batched writing reduces load and improve storage efficiency. The Go implementation ensures reliable data storage process, supports easy integration with other components like the Graphical User Interface (GUI), and is capable of handling large volumes of data.

5.2.2. Data storage

The setup of InfluxDB was quite straight forward. Deployment was done using the Docker [45] image, a platform designed to easily deploy applications in containers. After setting up credentials, The Access token was copied for use by other components.

5.2.3. API

The API implementation relies on the complete system application, including the Machine Learning algorithm. Future users of the system may chose real-time feedback for timely interventions and support by teachers. Alternatively, they may chose to review the feedback after the session has ended. These two scenarios can be constructed by a query in Flux only using the from, range and filter functions. This demonstrates that no single implementation that can cover all use cases, but aims to provide a solid foundation that is extendible. Appendix B contains two queries that cover the two mentioned scenarios, from which other scenario can be created.

5.2.4. GUI

To visualise collected data, Grafana uses dashboards consisting of panels, with each panel representing a graph or chart. The creation of a panel involves connecting to a data source, querying and transforming the data. Specifically, Grafana allows for connections with external sources such as Influx DB. When connected to Influx DB, it constructs queries in Flux to retrieve data from the database. if other sources are used, the programming language for queries will change accordingly. This means that the query in Grafana is constructed as described in the preceding section.

After retrieving data, Grafana enables data transformations to be able to visualise specific characteristics that are not immediately evident. In this case, that is not needed. Once all 3 steps are completed, data can be visualised within a panel. The optimal visualisation method depends on the specific nature of the data that is obtained. To accurately represent the system's state, it is important to display both incoming messages and the number of nodes. Therefore, the dashboard consists of 8 panels, one for each measurement and one for the connected nodes. This is shown in Figure 5.1.

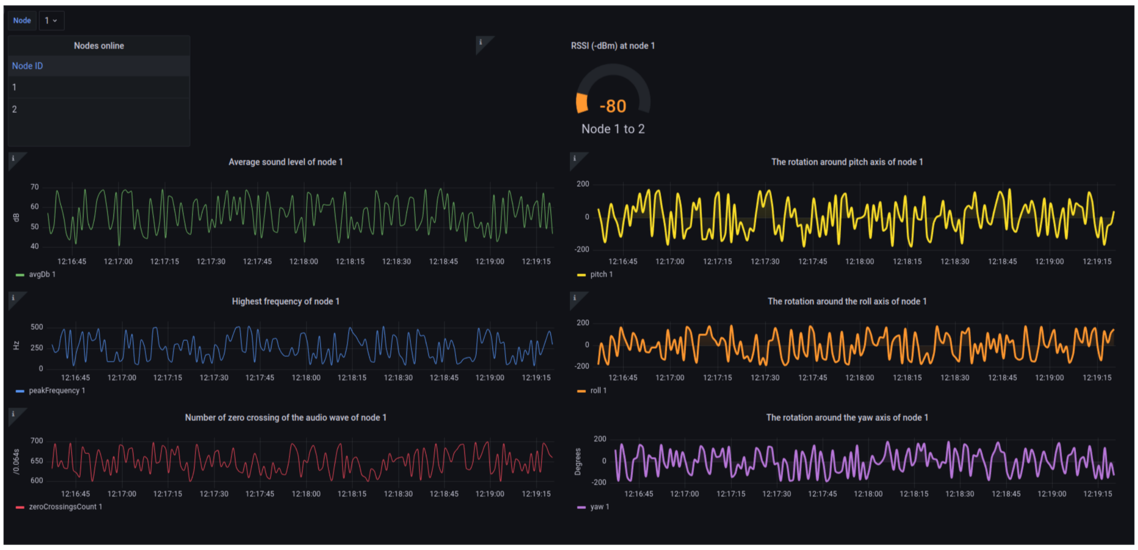

Node 1
Nodes online
Node ID
1
2
RSSI (-dBm) at node 1
-80
Node 1 to 2
Average sound level of node 1
avgDb 1
The rotation around pitch axis of node 1
pitch 1
Highest frequency of node 1
peakFrequency 1
The rotation around the roll axis of node 1
roll 1
Number of zero crossing of the audio wave of node 1
zeroCrossingsCount 1
The rotation around the yaw axis of node 1
yaw 1

Figure 5.1: Grafana dashboard with dummy data

5.3. Total overview

The final system's architecture, as shown in Figure 5.2, Illustrates the end-to-end integration of static and dynamic nodes, that all communicate to the root node. The root node interfaces with the server through serial for data storage.

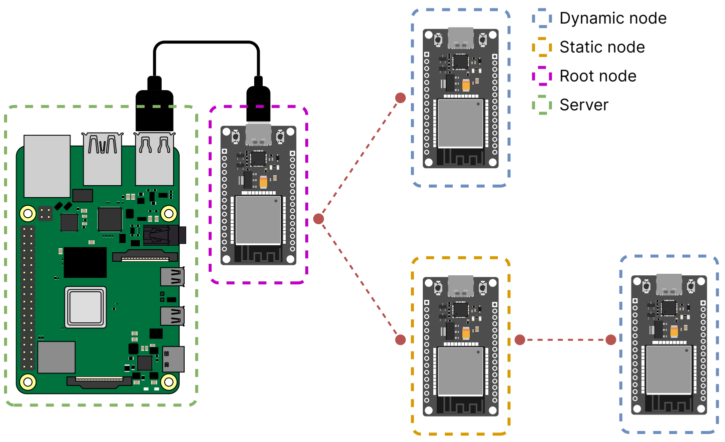


Figure 5.2: General overview of the design of the system

5.4. Validation of wireless communication

Tests were executed to verify the networks operation. Several scenarios were explored:

- Relay functionality was tested by limiting transmission power to simulate large range.
- Message transmission reliability was tested by simulating various traffic volumes and tracking lost packets.
- Message processing limits were tested using artificial loads created by sending over 500 messages per second from multiple nodes.
- Round trip time was measured for directly and relay-transmitted messages.

As of this time, the final size of the data package remains undetermined. To fully assess the capabilities of the network, some tests were conducted with varying message sizes, ranging from 50 bytes to a maximum of 250 bytes.

The relay function was verified, the node was able to successfully switch to the relay node once out of range of the root node without losing any messages, and was able to switch back to the root node once out of range of the relay. The average round trip time was increased when switching, as the message had to be relayed, going from a median of 6.1 ms to 11.6 ms. There was also a one time delay when updating the routing table, which had an average time of 23.4 ms.

The system's reliability was tested under both low-volume and high-volume conditions. the low-volume scenario involved a simulated load of 50 nodes transmitting a message every second, while high-volume scenario involved 200 nodes transmitting a message every second. Both tests included a total of 1 million messages, each with a size of 125 bytes. The outcomes of these tests are presented in Table Table 5.1.

	High-volume traffic	**Low-volume traffic**
Messages lost	637	432
Success rate	99.93%	99.95%

Table 5.1: Success rate of message transmission at 1 million total messages

Processing limits were tested at different message sizes. Two nodes simulated a stream of messages exceeding these limits to keep the message queue full. The received messages were tallied and measured every second for 100 seconds. The results are shown in Figure Figure 5.3.

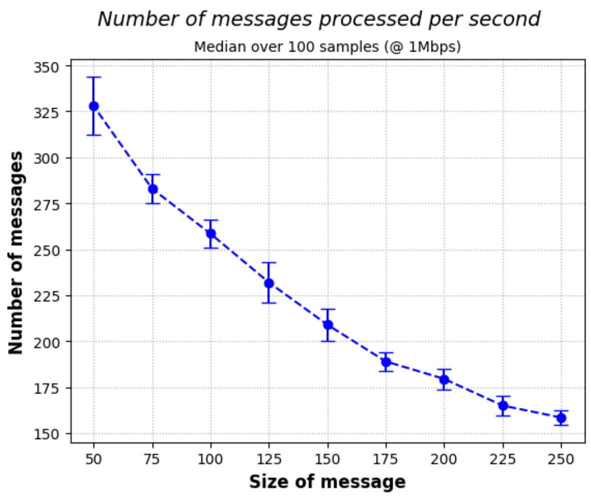


Figure 5.3: Messages processed per second at different message sizes

The round trip time was also measured for different message. Figure 5.4 shows that, after compensating for the theoretical transmission time, the end-to-end processing time remains consistent at 4.24 ms, indicating that the processing time does not scale with the message size.

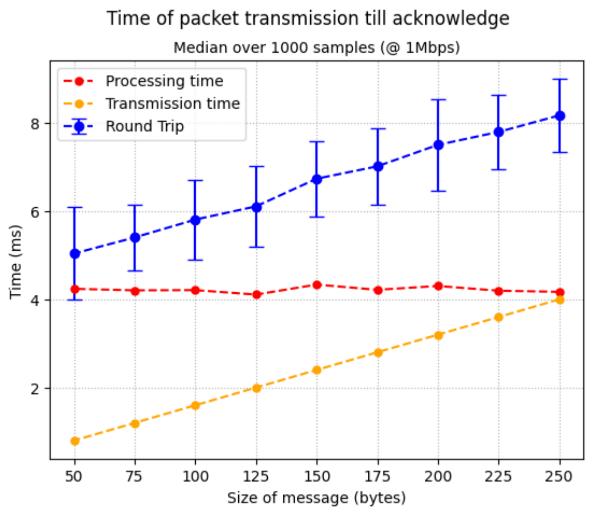


Figure 5.4: Round trip time of messages at different message sizes

5.5. Validation of server

The validation process of the data ingress system entailed multiple testing to ensure its reliability, efficiency, and accuracy. Initial assessment showed that a total of 21647 messages were processed, resulting in a 99.6073% success rate. Out of these, 85 messages failed during the initial transmission due to errors. These failed messages were successfully recovered on the first resend attempt, increasing the total number of messages processed to 21732, resulting in an overall success rate of 100% (see table: 5.2).The absence of the second and third resend attempts highlights the protocol robustness. The protocol's capacity for two additional resend attempts ensures that transmission errors are unlikely to remain unresolved, providing a strong safeguard against transmission failures. Error-handling capabilities were tested by simulating bundle losses and corrupted data. The ESP32 successfully retransmitted data up to three times as specified in the design, with the server's acknowledgment and retransmission requests functioning correctly. Time synchronization between the server and ESP32 was validated by comparing timestamps across multiple runs, ensuring that the synchronization was accurate to within a few microseconds.

Metric	Value
Total Messages Sent	21732
Initial Successful Messages	21562
Messages Recovered (1st Resend)	85
Messages Recovered (2nd Resend)	0
Messages Recovered (3rd Resend)	0
Success Rate Before Resend	99.6073%
Success Rate After Resend	100%

Table 5.2: Validation results showing successful data transmission rates.

The baud rate of 500000 was extensively tested for speed and reliability, while additional tests explored the system's performance at higher and lower baud rates. The 1 millisecond timeout configuration was validated for low latency, with further assessment for robustness up to 10 milliseconds.

To evaluate system performance, combination of batch size and baud rate were tested for throughput and efficiency. At a baud rate of 500,000, the system processed 375 messages per second with a batch size of 75 and 250 messages per second with a batch size of 50. For a baud rate of 115,200, the system processed 150 messages per second for both batch sizes of 75 and 50, see table 5.3. The batching mechanism was validated by processing message batches of various sizes, ranging from 5 to 20 messages. The optimal batch size of 10 was identified, minimizing database interaction overhead while maintaining low latency for real-time analysis. High-volume tests confirmed reliably server performance under continuous data transmission.

Maximum Batch Size	Messages Processed per Second	Baud Rate
75	375	500,000
50	250	500,000
75	150	115,200
50	150	115,200

Table 5.3: Batch performance metrics for different configurations of batch size and baud rate.

Data integrity was verified through rigorous checksum validation. The CRC-16 checksum of each received data bundle was recalculated and compared with the original to confirm data integrity. Tests were conducted with artificially introduced errors to validate the error-detection mechanisms. The robustness of command byte selection was assessed by simulating bit-flip scenarios to ensure that invalid commands could not corrupt the system.

The error-handling capabilities were evaluated under different traffic conditions. The system maintained a 100% success rate, demonstrating consistent reliability regardless of traffic volume. This analysis is presented below in table 5.4. These results validate the robustness of the data ingress system in handling varying data loads and communication speeds. The testing confirms the system's reliability and efficiency in diverse operating conditions.

Traffic Type	Messages Lost	Success Rate
High-volume traffic	0	100%
Low-volume traffic	0	100%

Table 5.4: Traffic success rate under varying conditions.

6
Discussion of the result

This project focused on designing a privacy-oriented, adaptable data acquisition system for Multimodal Learning Analytics (MMLA). This section evaluates and discusses the outcomes of the data acquisition system in the context of the requirements, takeaways, and boundary conditions as outlined in chapter 2.

6.1. Key achievements and results

The project successfully delivered a system, which met its primary goals:

Data Acquisition: The system achieved reliable data acquisition using a combination of an offline-capable server (MRS3) and a wireless communication network leveraging ESP-NOW (MRS5). The offline capability ensures independence from external networks, enabling secure and uninterrupted operation even in settings without internet access. Furthermore, the system was implemented with a time-series database (InfluxDB) and batch processing for data storage, significantly optimizing resource usage and data retrieval processes (MRS1, MRS2).

Privacy Orientation: The system adheres to GDPR guidelines by minimizing sensitive data collection and anonymizing user identifiers. (MRS4)

Robustness and Reliability: Designed a robust and reliable system using ESP-NOW, which was validated under various conditions achieving a communication success rate of higher than 99.9% (MRS7) even in high-traffic scenarios with multiple nodes (MRS6).

Scalability and Adaptability: Supported the addition or removal of nodes without affecting overall system performance while enabling the high data throughput, meeting the requirement for adaptability (MRS8, MRS11).

Easily integrable: The system was designed to operate shortly after being powered on, requiring no manual configuration (MRS9). Once set up, the system functioned without human intervention (MRS10), ensuring a seamless user experience.

6.2. Linking the Results to the Requirements

The Success of the project can be evaluated against the initial Program of Requirements (PoR) as seen in chapter 2 .

Mandatory requirements All mandatory requirements were met, particularly MRS1- MRS4, demonstrating the system's ability to collect, communicate, store, and extract data.

Trade-Off Requirements Trade-off requirements such as low power consumption (TRS1) and scalability (TRS4) were addressed effectively through ESP-NOW and modular hardware design.

Boundary Conditions The project adhered to its boundary conditions by demonstrating functionality with 2 static and 2 dynamic nodes (BC1) and focusing on data acquisition and storage (BC2), while staying in the cost limit.

6.3. Reflection on Project Success

The project achieved its goals of creating a functional, privacy-oriented data acquisition system. The results validate the system's ability to reliably collect, process, and store data while maintaining compliance with ethical guidelines. Key goals such as unobtrusive data collection, scalability, and robust communication were achieved. However, future iterations could focus on implementing encryption, refining real-time capabilities, and expanding deployment scenarios.

7 Conclusion, recommendations and future work

The aim of the project was to design a privacy-oriented data acquisition system that is adaptable and integrates easily into large, dynamic classrooms. This thesis focussed on achieving this by facilitating reliable wireless communication and effective data storage. It functions as a foundational research product and allows for the integration of additional features to support future research in MMLA.

The system was built around a network architecture consisting of a root node, small static nodes, and wearable dynamic nodes for data collection and communication. ESP32 microcontrollers were used to implement an ESP-NOW mesh protocol, enabling high data throughput and reliable wireless communication. A Raspberry Pi 5 was selected to run the server due to its strong processing power and memory capacity, ensuring efficient data storage and handling. The server communicates with the root node via a serial UART protocol, implemented in Go, to manage data ingress. The incoming data was stored anonymously on InfluxDB, which is designed for high performance data ingestion. Its built-in API enables simple queries in Flux to access the data. The external application Grafana was selected for data visualisation, providing a user-friendly interface to monitor the system's state.

Testing confirms that the system can handle high message volumes and achieve near-perfect data transmission rates. Testing confirmed that the wireless communication could handle high message volumes, achieving a communication success rate of 99.93% under high-traffic conditions with 200 messages per second. Data ingress was able to process up to 375 messages per second with a batch size of 75 messages with a success rate of 100%. These results demonstrate the reliability and robustness of the components' transmission capabilities.

Overall, the system delivers reliable data collection, communication and storage with a success rate over 99.93% under load. The final design ensures a data throughput of over 200 messages per second, reduces privacy risks by storing data anonymously and integrates easily into large, dynamic classrooms due to the simple configuration of the network. Additionally the system starts functioning immediately and nodes can be added or removed without impacting the overall performance.

7.1. Recommendations

While the project successfully developed the wireless communication and server modules, it faced difficulties in implementing the data ingress process. Namely, the decision to use Go for server-side operations highlighted a trade-off between simplicity and system efficiency. The language's ease of use and built-in concurrency support were advantageous for this project's scope, but the default 100-millisecond timeout for certain operations imposed limitations on latency-sensitive tasks. For a more demanding or scalable system, a language like C++ would be a more suitable choice due to its higher performance and finer control over system resources. Given that the system is intended as a foundational research product, it is recommended to select C++.

Additionally, more time should have been allocated to comparing different data transfer methods. The initial decision to proceed with continuous streaming without any evaluation, proved to be inefficient. This lead to unnecessary delays which would have been avoided by doing more extensive research.

Lastly, although the NTP-based time synchronisation achieves good accuracy, there is 1 consideration that needs to be kept in mind. Since time synchronisation is dependent on the root node, dynamic nodes that are out of range must rely on static nodes to relay time requests. This can introduce minor drifts that could accumulate and are currently unaccounted for. In classroom settings such as Tellegen Hall, no issues have been observed so far. However, to determine the limitations of this implementation, additional testing is advised.

7.2. Future work

Although the system meets its intended objective, there are opportunities for further development. These include integrating advanced features such as real-time analytics, implementing encryption mechanisms to enhance data security, and conducting additional testing in real-world educational environments. Addressing these areas would reduce privacy risks and improve effectiveness and integrability.

This project is an important advancement in using privacy-oriented data acquisition technologies to enhance learning in educational settings. It serves as a foundation for future work, providing an adaptable platform that can handle a high data throughput. This allows for adjustments and the integration of extra sensors to assist researchers in innovating the growing field of Multimodal Learning Analytics.

A
Mesh network

ZH network works using a message queue, responsible for processing incoming messages as well as transmitting messages. The way these messages are received and transmitted is illustrated by block diagrams in Figure A.1 and Figure A.2

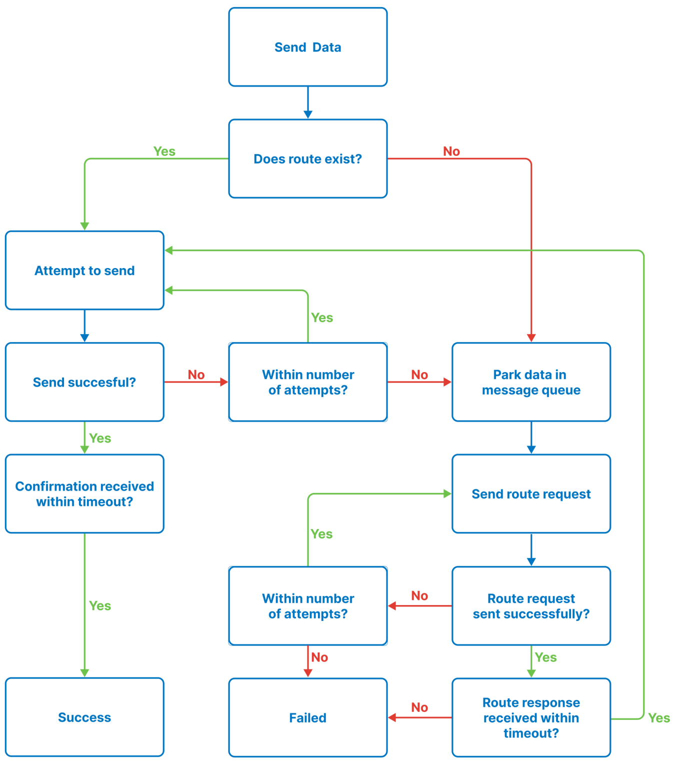

Send Data
Does route exist?
Yes
No
Attempt to send
Yes
Send succesful?
No
Within number of attempts?
No
Park data in message queue
Yes
Confirmation received within timeout?
Send route request
Yes
Yes
Within number of attempts?
No
Route request sent successfully?
No
Yes
Success
Failed
No
Route response received within timeout?
Yes

Figure A.1: Block diagram of sending a message

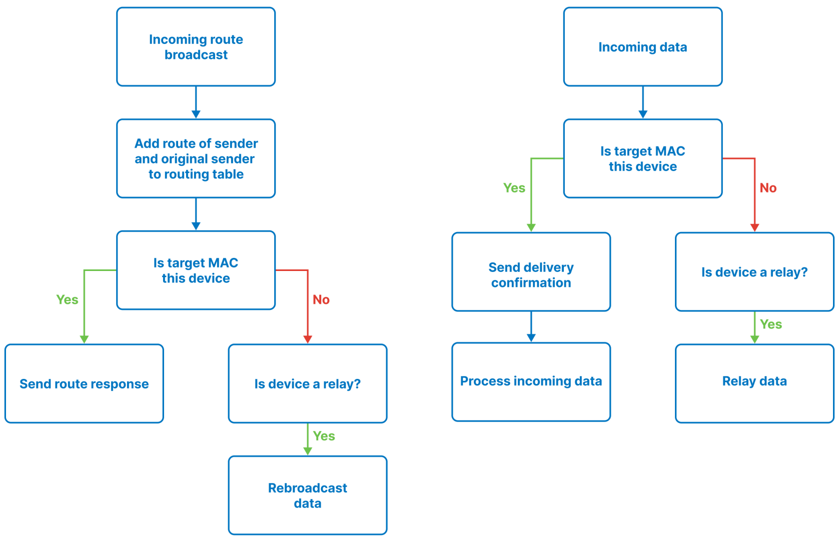


Figure A.2: Block diagram of receiving a message

B

API

```
// real-time, one individual student

from(bucket: "bap")
    |> range(start: -10m, stop: -5m)  // From 10 minutes ago to 5 minutes ago
    |> filter(fn: (r) => (r["_measurement"] == "BLEData" or r["_measurement"] == "measurement
        ") and r.id == "specific_node_id")
```

```
// post session of all students

from(bucket: "bap")
    |> range(start: -4h)  // last 4 hours
    |> filter(fn: (r) => (r["_measurement"] == "BLEData" or r["_measurement"] == "measurement
        "))
```

C
Code

The code is provided on GitHub: https://github.com/ryankolk/BAP-2024

Bibliography

[1] URL: https://microelectronics.tudelft.nl/eee/tellegen_hall/.

[2] Michail Giannakos et al. “Introduction to Multimodal Learning Analytics”. In: *The Multimodal Learning Analytics Handbook*. Ed. by Michail Giannakos et al. Cham: Springer International Publishing, 2022, pp. 3–28. ISBN: 978-3-031-08076-0. DOI: 10.1007/978-3-031-08076-0_1. URL: https://doi.org/10.1007/978-3-031-08076-0_1.

[3] Hamza Ouhaichi, Daniel Spikol, and Bahtijar Vogel. “Rethinking MMLA: Design Considerations for Multimodal Learning Analytics Systems”. In: *Proceedings of the Tenth ACM Conference on Learning @ Scale*. L@S ’23. Copenhagen, Denmark: Association for Computing Machinery, 2023, pp. 354–359. ISBN: 9798400700255. DOI: 10.1145/3573051.3596186. URL: https://doi.org/10.1145/3573051.3596186.

[4] Xavier Ochoa and Federico Dominguez. “Controlled evaluation of a multimodal system to improve oral presentation skills in a real learning setting”. In: *British Journal of Educational Technology* 51.5 (2020), pp. 1615–1630. ISSN: 0007-1013. DOI: 10.1111/bjet.12987. URL: https://berajournals.onlinelibrary.wiley.com/doi/10.1111/bjet.12987.

[5] Milica Vujovic et al. “Round or rectangular tables for collaborative problem solving? A multimodal learning analytics study”. In: *British Journal of Educational Technology* 51.5 (2020). © 2020 British Educational Research Association, pp. 1597–1614. DOI: 10.1111/bjet.12988.

[6] Xavier Ochoa. “Multimodal Learning Analytics – Rationale, Process, Examples, and Direction”. In: *Handbook of Learning Analytics*. Ed. by Charles Lang et al. 2nd. Vancouver, BC: SoLAR, 2022, pp. 54–65. DOI: 10.18608/hla22.

[7] Kshitij Sharma and Michail Giannakos. “Multimodal data capabilities for learning: What can multimodal data tell us about learning?” In: *British Journal of Educational Technology* 51.5 (2020), pp. 1450–1484. DOI: 10.1111/bjet.13013.

[8] Marcelo Worsley and Paulo Blikstein. “A Multimodal Analysis of Making”. In: *International Journal of Artificial Intelligence in Education* 28 (2018), pp. 385–419. DOI: 10.1007/s40593-017-0160-1. URL: https://doi.org/10.1007/s40593-017-0160-1.

[9] Rawad Hammad, Mohammed Bahja, and Mohammad Amin Kuhail. “Bridging the Gap Between Informal Learning Pedagogy and Multimodal Learning Analytics”. In: *The Multimodal Learning Analytics Handbook*. Ed. by Michail Giannakos et al. Cham: Springer International Publishing, 2022. ISBN: 978-3-031-08076-0. DOI: 10.1007/978-3-031-08076-0_7. URL: https://doi.org/10.1007/978-3-031-08076-0_7.

[10] Peter Kiefer Tiffany C.K. Kwok and Martin Raubal. “Unobtrusive interaction: a systematic literature review and expert survey”. In: *Human–Computer Interaction* 39 (2024). DOI: 10.1080/07370024.2022.2162404. eprint: https://doi.org/10.1080/07370024.2022.2162404. URL: https://doi.org/10.1080/07370024.2022.2162404.

[11] Haifa Alwahaby and Mutlu Cukurova. “Chapter 2 - Navigating the ethical landscape of multimodal learning analytics: a guiding framework”. In: *Ethics in Online AI-based Systems*. Ed. by Santi Caballé, Joan Casas-Roma, and Jordi Conesa. Intelligent Data-Centric Systems. Academic Press, 2024. ISBN: 978-0-443-18851-0. DOI: https://doi.org/10.1016/B978-0-443-18851-0.00014-7. URL: https://www.sciencedirect.com/science/article/pii/B9780443188510000147.

[12] Haifa Alwahaby et al. “The Evidence of Impact and Ethical Considerations of Multimodal Learning Analytics: A Systematic Literature Review”. In: *The Multimodal Learning Analytics Handbook*. Ed. by Michail Giannakos et al. Cham: Springer International Publishing, 2022. ISBN: 978-3-031-08076-0. DOI: 10.1007/978-3-031-08076-0_12. URL: https://doi.org/10.1007/978-3-031-08076-0_12.

[13] Mutlu Cukurova, Michail Giannakos, and Roberto Martinez-Maldonado. “The promise and challenges of multimodal learning analytics”. In: *British Journal of Educational Technology* 51.5 (2020). DOI: https://doi.org/10.1111/bjet.13015. eprint: https://bera-journals.onlinelibrary.wiley.com/doi/pdf/10.1111/bjet.13015. URL: https://bera-journals.onlinelibrary.wiley.com/doi/abs/10.1111/bjet.13015.

[14] Pankaj Chejara et al. “Bringing Collaborative Analytics using Multimodal Data to Masses: Evaluation and Design Guidelines for Developing a MMLA System for Research and Teaching Practices in CSCL”. In: *Proceedings of the 14th Learning Analytics and Knowledge Conference*. LAK ’24. Kyoto, Japan: Association for Computing Machinery, 2024. ISBN: 9798400716188. DOI: 10.1145/3636555.3636877. URL: https://doi.org/10.1145/3636555.3636877.

[15] Javaria Hassan, Jovin Leong, and Bertrand Schneider. “Multimodal Data Collection Made Easy: The EZ-MMLA Toolkit: A data collection website that provides educators and researchers with easy access to multimodal data streams.” In: *LAK21: 11th International Learning Analytics and Knowledge Conference*. LAK21. Irvine, CA, USA: Association for Computing Machinery, 2021. ISBN: 9781450389358. DOI: 10.1145/3448139.3448201. URL: https://doi.org/10.1145/3448139.3448201.

[16] Roberto Martinez-Maldonado et al. “Physical learning analytics: a multimodal perspective”. In: *Proceedings of the 8th International Conference on Learning Analytics and Knowledge*. LAK ’18. Sydney, New South Wales, Australia: Association for Computing Machinery, 2018. ISBN: 9781450364003. DOI: 10.1145/3170358.3170379. URL: https://doi.org/10.1145/3170358.3170379.

[17] Maurizio Garbarino et al. “Empatica E3 - A wearable wireless multi-sensor device for real-time computerized biofeedback and data acquisition”. In: *Empatica, Inc. and Massachusetts Institute of Technology*. Cambridge, MA, USA and Milan, Italy, 2020.

[18] Manas Kumar Parai, Banasree Das, and Gautam Das. “An Overview of Microcontroller Unit: from Proper Selection to Specific Application”. In: *International Journal of Soft Computing and Engineering (IJSCE)* 2.6 (Jan. 2013). ISSN: 2231-2307.

[19] Ashutosh M. Bhatt. “How to Select a Microcontroller for a Particular Application”. In: *International Journal of Engineering Research & Technology (IJERT)* 6.4 (Apr. 2017). This work is licensed under a Creative Commons Attribution 4.0 International License. ISSN: 2278-0181. DOI: IJERTV6IS040735. URL: http://www.ijert.org.

[20] David A. Patterson and John L. Hennessy. *Computer Organization and Design, Fifth Edition: The Hardware/Software Interface*. 5th. Cited section: Section 1.6 Performance, pages 28–40. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 2013. ISBN: 0124077269.

[21] Subhawat Jayasvasti and Don Isarakorn. “Study of Interaction Between Operational Clock-Frequency and Energy Consumption of Microcontrollers for Wireless Sensor Applications”. In: *MATEC Web of Conferences* 192 (2018). Published online 14 August 2018. Presented at the 4th International Conference on Engineering, Applied Sciences and Technology (ICEAST 2018), Track 2: Mechanical, Mechatronics and Civil Engineering., p. 02042. DOI: 10.1051/matecconf/201819202042. URL: https://doi.org/10.1051/matecconf/201819202042.

[22] STMicroelectronics. *Accurate Power Consumption Estimation for STM32L1 Series of Ultra-Low-Power Microcontrollers*. Tech. rep. TA0342. Rev. 2, May 2013. Geneva, Switzerland: STMicroelectronics, 2013. URL: http://www.st.com.

[23] Espressif Systems. *ESP32-DEV-30P-38P Datasheet*. Technical datasheet for ESP32 development kits. 2023. URL: https://www.espressif.com.

[24] Arduino S.r.l. *Arduino Nano 33 IoT Datasheet*. Product Reference Manual, SKU: ABX00027, Revision 4, modified 12 December 2024. 2024. URL: https://docs.arduino.cc.

[25] Raspberry Pi Ltd. *Raspberry Pi 5 Product Brief*. Technical specifications and overview of Raspberry Pi 5. Published by Raspberry Pi Ltd. Aug. 2024. URL: http://pip.raspberrypi.com.

[26] STM32 Base. *STM32F103C8T6 Blue Pill Board Overview*. 2024. URL: https://stm32-base.org/boards/STM32F103C8T6-Blue-Pill.html.

[27] STMicroelectronics. *STM32F103C8T6 Datasheet: Medium-Density Performance Line ARM®-Based 32-Bit MCU*. 2023. URL: `https://www.st.com/resource/en/datasheet/stm32f103c8.pdf`.

[28] RS Components. *ESP32 Microcontroller Development Board - Product 2863991*. 2024. URL: `https://nl.rs-online.com/web/p/microcontroller-development-tools/2863991`.

[29] Espressif. *ESP32 Wifi Driver*. URL: `https://docs.espressif.com/projects/esp-idf/en/stable/esp32/api-guides/wifi.html`.

[30] Espressif. *ESP32 series datasheet*. URL: `https://www.espressif.com/sites/default/files/documentation/esp32_datasheet_en.pdf`.

[31] Espressif. *ESP BLE API*. URL: `https://docs.espressif.com/projects/arduino-esp32/en/latest/api/ble.html`.

[32] Espressif. *ESP-BLE-MESH API*. URL: `https://docs.espressif.com/projects/esp-idf/en/stable/esp32/api-reference/bluetooth/esp-ble-mesh.html`.

[33] Espressif. *ESP-NOW API*. URL: `https://docs.espressif.com/projects/esp-idf/en/latest/esp32c3/api-reference/network/esp_now.html`.

[34] Zigbee alliance. *ZigBee Specification*. URL: `https://zigbeealliance.org/wp-content/uploads/2019/11/docs-05-3474-21-0csg-zigbee-specification.pdf`.

[35] aZholtikov. *Azholtikov/zh_network: Esp32 ESP-IDF and esp8266 RTOS SDK component (Arduino Library for ESP32 family) for ESP-now based Mesh Network.* URL: `https://github.com/aZholtikov/zh_network`.

[36] Miklós Maróti et al. "The flooding time synchronization protocol". In: *Proceedings of the 2nd International Conference on Embedded Networked Sensor Systems*. SenSys '04. Baltimore, MD, USA: Association for Computing Machinery, 2004, pp. 39–49. ISBN: 1581138792. DOI: `10.1145/1031495.1031501`. URL: `https://doi.org/10.1145/1031495.1031501`.

[37] D.L. Mills. "Internet time synchronization: the network time protocol". In: *IEEE Transactions on Communications* 39.10 (1991), pp. 1482–1493. DOI: `10.1109/26.103043`.

[38] Prometheus. *Prometheus Blog*. URL: `https://prometheus.io/docs/introduction/overview/`.

[39] Influx. *InfluxData*. Dec. 2024. URL: `https://www.influxdata.com/`.

[40] Timescale. *Timescale*. URL: `https://www.timescale.com/`.

[41] Matthias Biehl. *API Architecture: The Big Picture for Building APIs*. English edition. Introduction chapter, pages 15–25. Scotts Valley, CA, USA: CreateSpace Independent Publishing Platform, 2015, p. 190. ISBN: 978-1508676645.

[42] InfluxData. *InfluxDB v2 Documentation*. 2024. URL: `https://docs.influxdata.com/influxdb/v2/`.

[43] PlatformIO. *Your gateway to embedded software development excellence*. URL: `https://platformio.org/`.

[44] Microsoft. *Visual studio code - code editing. redefined*. Nov. 2021. URL: `https://code.visualstudio.com/`.

[45] Inc. Docker. *Docker*. URL: `https://www.docker.com/`.