Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 263 (2025) 522–529

Procedia Computer Science

www.elsevier.com/locate/procedia

International Conference on Industry Sciences and Computer Science Innovation

Integrating Unity into IoT solutions

Manuel Casimiro[a], Carlos Coutinho[a,b]

[a]*Instituto Universitário de Lisboa (ISCTE-IUL), Lisboa, Portugal*
[b]*ISTAR, Lisboa, Portugal*

Abstract

This paper investigates the integration of Unity with Internet of Things (IoT) technologies to enhance data visualization and interaction capabilities. Unity, renowned for its capabilities in game development, presents untapped potential within the IoT landscape, particularly in 3D modeling, augmented reality (AR), and virtual reality (VR), which can revolutionize end-user applications. This paper begins with a comprehensive literature review exploring Unity's integration with sensors within the recent years. It proceeds with a detailed technology stack and architecture proposal tailored specifically to a Unity-powered mobile application. The findings underscore the feasibility and effectiveness of Unity's integration into IoT frameworks, promising enhanced user experiences. Future research directions are also explored and look to further explore and optimize this integration, aiming to bridge the gap between cost-effective IoT solutions and groundbreaking data visualization techniques using sensor data.

Keywords: IoT; Unity; Internet of Things;

1. Introduction

Energy sustainability is an ever-pressing issue in today's world. As societal needs arise, inefficiencies in energy consumption lead to significant environmental problems such as contributing to climate change and environmental degradation and destabilization, while also carrying notable economic consequences [1]. The urgency in mitigating energy waste cannot be overstated such that the mission to find plausible and innovative solutions have transcended both the scientific and the technological panorama and have become a moral and global imperative.

The main objective of this paper is to present a comprehensive Internet of Things (IoT) architecture that integrates seamlessly with Unity, aiming to revolutionize how we visualize and interact with IoT data. This paper also serves as the initial phase of a thesis project focused on developing a general-public application designed to reduce household energy waste. This is achieved through the collection, processing, and display of sensor data, encouraging behavioral

10.1016/j.procs.2025.07.063

changes in users. Additionally, this project lays the groundwork for future developments aimed at fully harnessing the synergy between Unity and IoT.

The structure of this paper is as follows: Section 2 provides a comprehensive review of pertinent literature in the Unity-IoT landscape, accompanied by insights from our Systematic Literature Review. Section 3 presents the research methodology employed in this study. In Section 4, we conduct an extensive examination of our proposed Technology Stack, which is subsequently applied to a functioning IoT ecosystem in Section 5. Section 6 presents our results and analysis, while Section 7 offers our conclusions and outlines future directions for ongoing research efforts and for stakeholders interested in advancing Unity's integration in IoT applications.

2. Related Work

2.1. Conducting a Systematic Literature Review

Given that this paper focuses on the integration of Unity within IoT, an environment where sensors play a pivotal role, this subsection explores existing literature that examines the utilization of Unity and sensors to develop impactful and practical solutions. In order to aggregate and select the appropriate papers for this subsection, we led a systematic literature review (SLR) using Scopus [2], having built the following query:

- *TITLE-ABS ((unity) AND ("Internet of Things" OR iot) AND sensor AND NOT ("machine learning"))*

As a result, we obtained a total of 36 papers that satisfied the conditions proposed by the query, a number we considered too large for an in-depth analysis of each paper. Table I describes the subsequent filters applied to the query and the number of papers that remained at each step of the SLR.

Table 1. SLR's filtering conditions and subsequent number of papers.

Filter	Remaining papers
Initial Query	36
Release Date: Post-2016	33
Subject Areas: Engineering, Energy or Environmental Science	16

Having obtained 16 papers, we consider this an acceptable number to conduct an individual review of each of these thus concluding our SLR.

2.2. Existing IoT-adjacent solutions using Unity

With the completion of our SLR, this subsection will now explore some of our findings on the current developments within the IoT-Unity landscape.

The solution proposed by Shah et al. [3] presents an innovative way to monitor potted plants, reduce waterlogging, mud cracks and enhance plant growth. This solution merges a DHT11 sensor with Augmented Reality (AR), powered by the Unity platform. The researchers developed a mobile application that showcases a given plant's temperature and humidity in real-time, within the plant's surroundings. The researchers also highlight the cost-effectiveness of the solution as well its wireless nature, which allows for facilitated handling and installation.

Given the cost-effective facet of the solution, we consider it acceptable for the solution to offer the modest list of features described above. However, the solution contains two vulnerabilities the researchers may not have accounted for. Firstly, the DHT11 sensor is significantly inferior to its DHT22 counterpart despite both sensors carrying a similar price point [4]. Secondly, in the proposed solution, the sensor must be placed within the soil of the potted plant in question, making it liable to water-damage, leading to an affordable although unreliable solution. However, the data transmission displayed here, between the sensor and the Unity platform, is worthy of note and will be taken into consideration in the proposed architecture presented in this paper.

A similar solution was proposed by Tisza et al. [5], also employing IoT sensors and communication to a mobile application to ensure and improve plant growth. In addition to temperature and humidity receptors, this solution improves on the previous paper by adding carbon dioxide measurements, furthering the system's depth and utility. Communication between the application layer and the perception layer is entrusted to MQTT, ensuring a stable and lightweight way to communicate sensor data in real time. The researchers opted to have a Quick Response (QR) code placed within the user's environment upon where the 3D model, accompanied by the corresponding sensor readings, is instantiated in Augmented Reality, something we consider was executed better on the previous solution due to it avoiding added costs and the necessity for physical objects to function.

Another solution employing Unity's AR capabilities alongside sensors was developed by Dash et al. [6] and compiled the use of different sensors, such as temperature, humidity and pressure sensors, with 3D visualization in real time displaying the gathered data alongside the appropriate 3D model within a mobile application e.g. if pressure sensors were installed on the wheels of a motorized vehicle, the application would display a 3D model of an automobile with the corresponding sensor data over the appropriate wheel.

As detailed above, the solution is focused on the dynamic display of data, lending the user an intuitive and simple way to interpret and monitor said data but does not provide ways to analyze it, something we consider to be a considerable flaw in the design. Nonetheless, similarly to the paper explored in [3], the data transmission structure between the IoT layer and the Unity platform is deployed effectively and will be taken into account under our solution.

The article developed by Neves et al. presents a novel solution to physical rehabilitation, combining a gamified environment, powered by Unity's Virtual Reality (VR) capabilities, with the capture and analysis of captured data from a Smart Glove device. The application, aimed at patients who sustained injuries to their upper extremities, utilizes a VR game to obtain data regarding variables such as applied force per finger, finger flexion and acceleration and rotation of the upper limbs. The obtained data is then aggregated and stored in an online database, allowing the researchers to analyze and visualize it. Finally, the researchers compared data from a patient who sustained injuries to their upper extremities to that of a healthy patient and determined that the proposed solution could effectively determine the state of a patient's recovery as well as help the patient's physiotherapist's decision-making process.

The solution developed by my colleagues presents an insightful and beneficial approach to the physical therapy-IoT landscape and showcases how IoT can deviate from traditional sensors to achieve truly advantageous solutions. However, during our analysis of the solution we encountered a possible detriment in the solution's architecture: the glove itself. While the glove is suitable for most injuries in the upper extremities, for the cases of injuries that cause increased sensitivity in the patient's hands (such as serious burns), less intrusive devices should have been explored further as these injuries could prevent innumerable patients from taking advantage of the solution.

Concluding this subsection, the paper researched by Vera-Coca et al. [7] represents a similar project to our own: a Unity-powered mobile application aimed at reducing energy through human-led decision-making. The researchers opted to focus on voltage anomalies namely 'voltage sags', a temporary drop in voltage levels in an electrical power system which can severely impact electrical equipment and machinery, potentially precipitating industrial malfunctions or even consequential damage to the equipment. The application allows users to not only consult voltage anomalies by the date of the occurrence but also, real-time data collected by the sensor in question: a Power Quality (PQ) sensor, allowing the solution to correctly identify the voltage anomalies in question.

Despite providing a solution with a similar end goal to our own, the aforementioned approach is more suitable to an industrial setting, due to the fact that voltage anomalies are not a relevant metric for the general-public since they do not present the user with an accessible and/or straightforward way of improving and/or fixing the problem. In addition, we believe the interface of the developed application is quite unpleasant and something that could have been explored further, given the capabilities of the Unity software.

In conclusion, while Unity has expanded beyond its origins in game development, the solutions discussed here underscore its potential for creating innovative data visualization methods and highlight positive advancements within the field. However, despite these advancements, Unity in this context remains relatively underutilized, a gap we aim to address through the investigations detailed in the following chapters as well as in future research endeavors.

3. Research Methodology

The Design Science Research (DSR) methodology [8] was chosen for conducting our research. In our exploration of IoT smartphone application development, the choice of the Design Science Research (DSR) methodology stands out due to its ability to prioritize the creation of innovative, practical solutions. Unlike purely theoretical or quantitative methods, the DSR's adaptability and responsiveness make it an ideal choice for addressing the dynamic nature of IoT technology, ultimately bridging the gap between theory and practice in the development of cutting-edge applications.

The DSR Methodology features four possible entry points with this paper using the Problem-Centred Initiation as a specific problem has been identified, as detailed in the paper's Introduction section.

4. Technology Stack

The main objective of this research project is to design and develop an intuitive smartphone application featuring a robust notification and alert system, all while maintaining a low cost. Such a system would allow the user to receive notifications when, for example, a light was on in a vacant room or if an air-conditioner device was on inside a room whose temperature is adequate. Given this, the requirements for our solution are:

- Sensor array: A collection of sensors that, altogether, can capture different types of data.
- Integrated Development Environment (IDE): For programming and uploading code to the microcontroller.
- Microcontroller: Able to handle data collection from the aforementioned array as well as communicating that data wirelessly.
- Message Broker: An intermediary messaging agent for reliably ensuring that the data is successfully sent beyond the perception layer.
- Database: Able to receive sensor data, store it as well as enable access to it upon request from an authorized source.
- Mobile Application: The end-user interface from which the user interacts with the proposed ecosystem.

Upon extensive research and consideration, the following technologies were chosen for each of the aforementioned requirements.

- Sensor array: DHT22 sensor, PIR motion sensor & Grove light sensor.

As mentioned previously, affordability is a key feature of this solution, Therefore, the widely adopted sensors DHT22, PIR motion sensor and the Grove light sensor were selected for their cost-effectiveness, reliability and ease of integration. Although alternative sensors are available, the DHT22, the PIR motion sensor, and the Grove light sensor were employed in this project due to their immediate accessibility during our research period. However, considering that the case study presented here is theoretical in nature, other sensors, such as the Adafruit AHT20 temperature and humidity sensor, the RCWL-0516 human presence detector, or the Adafruit TSL2561 light sensor, could readily substitute any of the sensors utilized in this project.

- Integrated Development Environment (IDE): Arduino IDE.

The Arduino IDE was selected for its user-friendly interface, extensive community support, and compatibility with a wide range of microcontrollers, including the one chosen for our solution. Its widespread adoption and robust feature set provide an effective platform for both efficient coding and debugging. One alternative we considered was the PlatformIO IDE, a more powerful and feature-rich development environment that supports a wider range of

microcontroller platforms and more advanced features, allowing for more complex projects at the cost of a steeper learning curve [9]. Ultimately, we chose Arduino IDE due to our solution's perception layer being relatively simple, making Arduino IDE well-suited and capable of handling our requirements.

- Microcontroller: ESP-wroom-32.

The ESP-wroom-32 was chosen for its robust processing power, integrated Wi-Fi and Bluetooth connectivity, and low power consumption. This versatile module is well-suited for handling the multiple sensors in our IoT system. Moreover, its popularity and extensive community support provide valuable resources for development and troubleshooting. During the selection process, we evaluated another microcontroller: the Raspberry Pi. The Raspberry Pi microcontroller offers significantly higher processing power (of up to 1.6 GHz) but comes with increased size, higher financial cost and energy requirements [10]. As our solution aims to be a low-cost, energy-conscious solution, we have dictated the ESP-wroom32 to be the ideal microcontroller for our solution.

- Message Broker: mosquitto & Node-RED.

For our message brokers, we opted for a combination of mosquitto and Node-RED. mosquitto was chosen for its efficient implementation of the MQTT protocol, providing reliable and lightweight communication between IoT devices. Node-RED serves as a versatile platform for testing communication between different layers and enables future expansion of our solution by facilitating access to various APIs and external services. An alternative to mosquitto that we examined was RabbitMQ. RabbitMQ, similarly to mosquitto, supports MQTT and message brokering but also provides additional protocol support, enhanced scalability, and more robust messaging features at the cost of being a more resource-intensive message broker. Given that our solution’s data flow is relatively lightweight and does not transmit sensitive information through the message broker, mosquitto was deemed sufficient for our requirements.

- Database: Firebase Realtime Database.

We chose Firebase Realtime Database due to its ability to provide real-time data synchronization, ensuring that all connected devices receive updates instantly. It offers a straightforward and scalable NoSQL cloud database, simplifying data storage and retrieval. Additionally, its seamless integration with other Firebase services and extensive documentation solidifies this as the ideal choice. One notable alternative identified in our research was the widely adopted MongoDB, which excels in overall performance, offering faster and seamless handling of vast amounts of data, making it suitable for large-scale data management and fast querying responses [11]. However, our application does not require large-scale data management or complex database queries. Therefore, our choice was heavily influenced by the ease of integration between the selected database and our application. Ultimately, Firebase Realtime Database was chosen due to its ready-made Unity packages, significantly reducing development time and meeting our specific requirements.

- Mobile Application: Unity.

As previously mentioned, Unity was selected as our application development platform due to its potential in IoT solutions and data visualization. Unity offers cross-platform support, powerful 3D visualization, and robust real-time interaction features, making it an ideal choice for creating visually rich and responsive IoT applications. Additionally, its flexibility allows for future expansions, such as integrating augmented reality dioramas or virtual reality features. As it was always our intention to explore innovative IoT applications using Unity, alternatives were not considered.

5. Solution Architecture

Following the assembling of our technology stack, we now present the architecture of our IoT solution, which integrates the chosen components detailed above into a cohesive system as seen in Fig. 1.

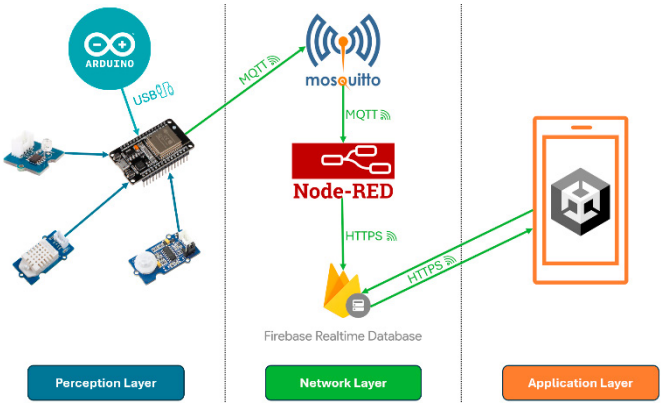


Fig. 1. Proposed solution architecture

5.1. Perception Layer

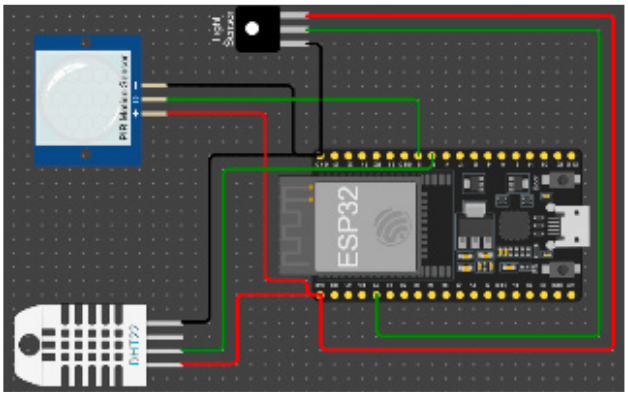
At the perception layer, our Arduino IDE code is uploaded to the ESP-wroom-32 microcontroller, instructing it on how to handle data received from the DHT22 sensor, the PIR motion sensor, and the Grove light sensor. The microcontroller then prompts the sensors to gather data periodically, which is subsequently transmitted through mosquitto to the network layer, the intermediate layer of the proposed architecture. To preserve bandwidth and maintain energy efficiency, the collected data are sent as a single message. Additionally, after the code has been uploaded, the solution has the capability to operate wirelessly, as a physical connection to the Arduino IDE platform is only required during the upload step. A diagram of our perception layer is presented in Fig. 2. (a).

5.2. Network Layer

Within this layer, mosquitto redirects the data to a Node-RED instance, which enriches the data with the current time and date and formats it into JSON before forwarding it to Firebase's Realtime Database as can be seen in Fig. 3. Additionally, two separate flows were developed to test the communication between the perception and network layers by producing synthetic data, thus removing the dependency on the sensor array to continuously produce and communicate data during the development period.

The database, which features a NoSQL schema i.e. does not conform to a traditional relational database model, can be accessed through HTPPS requests, allowing for efficient retrieval and management of the data. The injected timestamp described previously serves as the identifier for the database nodes, Fig. 2. (b), ensuring each data entry node is unique and stored in real-time.

a



b

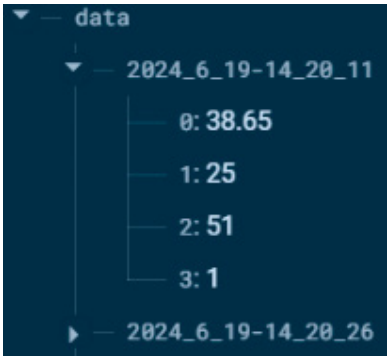


Fig. 2. (a) perception layer diagram; (b) database node

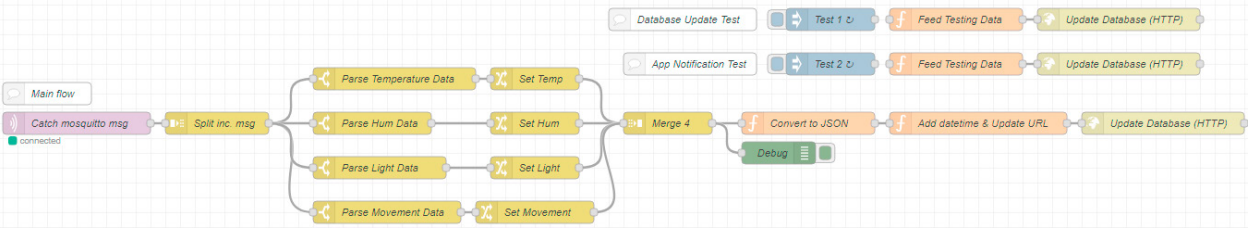


Fig. 3. Node-RED flow

5.3. Application Layer

Finally, our Unity application periodically exchanges HTTPS requests with our database, retrieving the latest data and updating the application interface with the most recent values. To better demonstrate the evolution of the data, whenever the data is updated in the application's interface, an arrow graphic beside the numeric value indicates its trend: increase, decrease, or no change. A screenshot of the application's interface is presented in Fig. 4.

6. Results Discussion

As stated previously, this paper had two main objectives: first, to present a comprehensive IoT architecture that integrates seamlessly with Unity to enhance how we visualize and interact with IoT data; and second, to establish the groundwork for a general-public application aimed at reducing household energy waste through the collection, processing, and display of sensor data to encourage behavioural changes in users.

To address the first objective, the proposed IoT architecture not only integrates with Unity but does so in a robust and highly responsive manner. The architecture fulfils all the requirements demanded by the application, ensuring seamless communication between sensors, the network layer, and Unity. This allows for real-time, accurate data visualization that is critical for interpreting sensor data effectively. The system's scalability and reliability make it well-suited for expanding the application's capabilities, as the architecture can accommodate more complex sensor arrays and larger datasets without compromising performance.

Regarding the second objective, while the initial development of the application has been successful, this represents only a first step in the broader thesis project. The current implementation demonstrates the viability of the concept by providing users with real-time, actionable feedback that encourages more energy-conscious behaviour. However, further exploration and iteration are required to fully harness the potential of Unity and IoT synergy. Future phases of the project will delve deeper into optimizing the system, expanding its features, and assessing its long-term impact on reducing household energy consumption.

7. Conclusion and Future Work

In this paper, we have explored the integration of Unity within IoT to create innovative end-user applications, leveraging its extensive capabilities in areas often lacking in widely used platforms, such as 3D modeling, augmented reality (AR), and virtual reality (VR). While we are not yet prepared to showcase how our own system will use these technologies nor make definitive statements about the system's long-term effectiveness in reducing household energy waste, preliminary assessments by our team indicate that our solution holds promising potential in both fronts. Future work will involve ongoing iterations and monitoring of the developed ecosystem to enhance the application's performance and ultimately empower individuals to contribute to energy conservation efforts through behavioral changes.

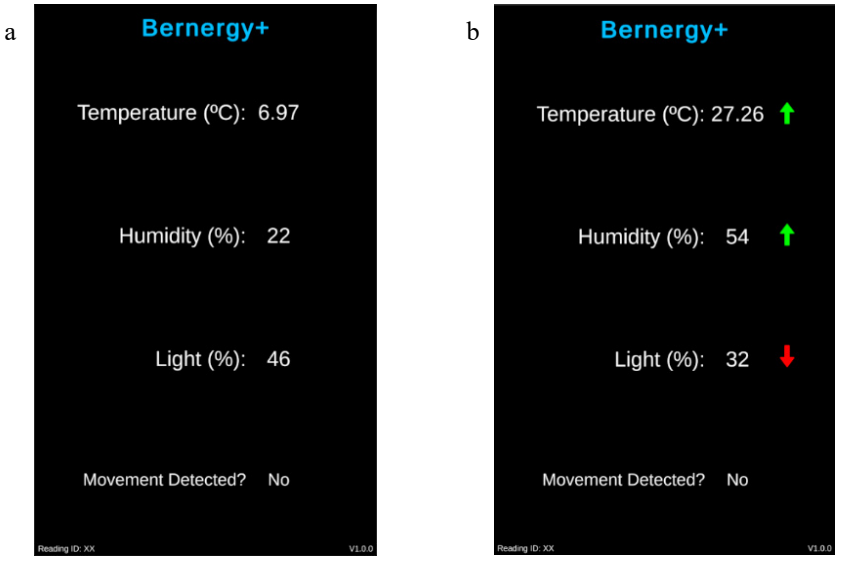


Fig. 4. (a) application interface upon start; (b) application interface after update

Acknowledgements

This work was supported by Fundação para a Ciência e a Tecnologia, I.P. (FCT) [ISTAR Projects: UIDB/04466/2020 and UIDP/04466/2020].

References

[1] P. and G. A. National Research Council *et al.*, *Hidden Costs of Energy: Unpriced Consequences of Energy Production and Use*. National Academies Press, 2010.
[2] 'Scopus preview - Scopus - Welcome to Scopus'. Accessed: Jun. 20, 2024. [Online]. Available: https://www.scopus.com/home.uri
[3] H. Shah, J. Gurnani, and S. Gajjar, 'Design and Development of AR-PlaSys: Augmented Reality Based Plant Monitoring System', in *2020 IEEE 17th India Council International Conference (INDICON)*, Dec. 2020, pp. 1–5. doi: 10.1109/INDICON49873.2020.9342166.
[4] RayMing, 'Temperature and Humidity Sensor DHT11 vs DHT22 Which one is better', RAYPCB. Accessed: Mar. 25, 2024. [Online]. Available: https://www.raypcb.com/dht11-vs-dht22/
[5] J. Tisza and D. Ortega, 'Architecture proposal for real-time sensor monitoring using IoT and Augmented Reality', in *2022 IEEE Engineering International Research Conference (EIRCON)*, Oct. 2022, pp. 1–4. doi: 10.1109/EIRCON56026.2022.9934808.
[6] V. E. Kolisetty Aditya and A. K. Dash, 'Real-life Applications of integration of Augmented Reality and Internet of Things', in *2022 Second International Conference on Artificial Intelligence and Smart Energy (ICAIS)*, Feb. 2022, pp. 1268–1273. doi: 10.1109/ICAIS53314.2022.9742851.
[7] F. Vera-Coca, A. Gil-de-Castro, R. Medina-Gracia, J. Garrido-Zafra, R. Savariego-Fernández, and A. Moreno-Munoz, 'Interactive visualization of IoT power quality data on mobile devices', in *2021 IEEE International Conference on Environment and Electrical Engineering and 2021 IEEE Industrial and Commercial Power Systems Europe (EEEIC / I&CPS Europe)*, Sep. 2021, pp. 1–6. doi: 10.1109/EEEIC/ICPSEurope51590.2021.9584771.
[8] J. vom Brocke, A. Hevner, and A. Maedche, 'Introduction to Design Science Research', 2020, pp. 1–13. doi: 10.1007/978-3-030-46781-4_1.
[9] 'Arduino IDE vs PlatformIO IDE — PlatformIO latest documentation'. Accessed: Jun. 14, 2024. [Online]. Available: https://docs.platformio.org/en/latest/faq/arduino-vs-platformio.html
[10] 'Comparing Microcontrollers: What Brain Should I Go With?', DigiKey. Accessed: Jun. 18, 2024. [Online]. Available: https://www.digikey.com/en/maker/projects/02d2dcb1a0d441f5a11fc9956559b226
[11] 'Firebase VS MongoDB: What To Choose?', Blog | TechMagic. Accessed: Jun. 19, 2024. [Online]. Available: https://www.techmagic.co/blog/firebase-vs-mongodb/