The European Master in Building Information Modelling is a joint initiative of:

Univerza *v Ljubljani*

Universidade do Minho

POLITECNICO MILANO 1863

BIM as a Smart Facility Management Tool of University Facilities

AbdurRahman Wassim Abou Yassin

BIM A+

UMinho | 2021

Universidade do Minho
Escola de Engenharia

AbdurRahman Wassim Abou Yassin

BIM as a Smart Facility Management Tool of University Facilities

BIM A+ European Master in Building Information Modelling

Co-funded by the Erasmus+ Programme of the European Union

December 2021

Universidade do Minho
Escola de Engenharia

AbdurRahman Wassim Abou Yassin

BIM as a Smart Facility Management Tool of University Facilities

European Master in
Building Information Modelling

Master Dissertation
European Master in Building Information Modelling

Work conducted under supervision of:
Bruno Acácio Ferreira Figueiredo
Miguel Ângelo Dias Azenha
Ricardo Lobão (Tutor in Institution)

Co-funded by the
Erasmus+ Programme
of the European Union

December, 2021

AUTHORSHIP RIGHTS AND CONDITIONS OF USE OF THE WORK BY THIRD PARTIES

ACKNOWLEDGEMENTS

Praise is to Allah the Almighty, on whom we depend for guidance, for giving me strength and good health to successfully complete this dissertation.

I would like to give my utmost respect and appreciation to my supervisor Prof. Bruno Figueiredo and co-supervisor Prof. Miguel Azenha, for the completion of this dissertation could not have been possible without their extensive knowledge in the field and their constant assistance and guidance.

I would also like to thank Eng. Ricardo Lobão, the tutor at the University of Minho, for his follow up on the project and meaningful insight that helped with the facility management workflow of the study.

I am especially grateful for Mario Coelho, who contributed in the major part of the programming of the microcontroller devices, and always gave useful advice and valuable information.

Gratitude is also expressed to the support and collaboration with the I&D Project "Cognitive CMMS Cognitive Computerized Maintenance Management System", with reference POCI-01-0247-FEDER-033574, co-financed by the European Regional Development Fund (ERDF), through the Competitiveness and Internationalization Operational Program (POCI).

To all the friends and colleagues who encouraged me and supported me, to my family in all parts of the world, to my father and my mother who looked out for me every step of the way.

STATEMENT OF INTEGRITY

I hereby declare having conducted this academic work with integrity. I confirm that I have not used plagiarism or any form of undue use of information or falsification of results along the process leading to its elaboration.

I further declare that I have fully acknowledged the Code of Ethical Conduct of the University of Minho.

AbdurRahman W. Wassim Abou Yassin

RESUMO

Título: BIM como ferramenta de Gestão Inteligente de Instalações Universitárias

A Qualidade Ambiental Interior (IEQ) das instalações de construção desempenha um papel significativo no bem-estar dos ocupantes e na eficiência global. Devido ao potencial de expansão das aplicações de Modelação de Informação de Edifícios (BIM) nas indústrias de Arquitectura, Engenharia, Construção e Operação (AECO), o número de estudos para o desenvolvimento de novas tecnologias que contribuiriam ainda mais para o seu sucesso multiplicou-se recentemente, particularmente em Sistemas de Gestão de Instalações (FMS).

Uma destas tecnologias proeminentes é a integração da Internet das Coisas (IoT) e da tecnologia BIM na fase de Gestão de Instalações (FM), uma vez que as aplicações BIM têm sido implementadas principalmente dentro das fases de concepção e construção. A investigação para a implementação da Internet das Coisas centrada em BIM em FMS tem vindo a surgir nos últimos anos com tecnologias de Redes de Sensores Sem Fios (WSN) que sentem e comunicam as condições ambientais através da Internet das Coisas (IoT). Esta dissertação visa integrar modelos BIM e dispositivos IoT especializados de microcontroladores Arduino numa estrutura que permite um fluxo de trabalho coeso para a monitorização, análise, interpretação e visualização de informação de origem IoT num ambiente BIM. Estes dispositivos IoT monitorizariam aspectos IEQ das instalações universitárias durante a fase de Operações e Manutenção (O&M) e visualizariam virtualmente a condição ambiental do edifício.

A estrutura concebida procura integrar as plataformas BIM (por exemplo, Autodesk Revit) e Serviços de Bases de Dados Online (ODS) directamente numa plataforma de visualização (por exemplo, Microsoft Power BI) sem ter de trocar previamente informações entre o modelo BIM e o ODS. Os dados fornecidos por ambas as fontes colaboram para criar painéis informativos e interactivos que exibem parâmetros ambientais em tempo real. Os parâmetros medidos pelos sensores são enviados para um ODS, tal como o MySQL, onde os dados são armazenados. O sistema desenvolvido regula parâmetros tais como temperatura, humidade, iluminação e qualidade do ar de acordo com limiares pré-definidos de comodidade de acordo com normas universais.

Espera-se que desenvolva e implemente um Sistema de Monitorização Ambiental para melhorar o desempenho do edifício e controlar o consumo de energia da Biblioteca do Campus de Azurém, Universidade do Minho. Espera-se que o sistema desenvolvido forneça uma ferramenta prática para a recolha e visualização de dados. Isto contribui para uma tomada de decisão mais rápida e precisa, mantendo assim uma qualidade ambiental equilibrada e ocupantes mais saudáveis. O sistema pode ser melhorado através da introdução de um processo de ensino-aprendizagem que compreende, prevê, e optimiza as variáveis de conforto. Tais sistemas podem ser alcançados com algoritmos de Machine Learning (ML) e a utilização de Inteligência Artificial (AI) e Big Data para criar, através da tecnologia BIM, um ambiente Digital Twin (DT) que controla o edifício físico de forma autónoma.

Palavras chave: Arduino, Modelação de informação de edifícios (BIM), Qualidade ambiental interior (IEQ), Serviços de bases de dados em linha (ODS), Sistemas de gestão de instalações (FMS).

ABSTRACT

Indoor Environmental Quality (IEQ) of building facilities plays a significant role in the occupants' well-being and overall efficiency. Because of the expanding potential of Building Information Modelling (BIM) applications in Architecture, Engineering, Construction, and Operation (AECO) industries, the number of studies for the development of new technologies that would further contribute to its success has recently multiplied, particularly in Facility Management Systems (FMS).

One of these prominent technologies is the integration of the Internet of Things (IoT) and BIM technology into the Facility Management (FM) phase, as BIM applications have been mainly implemented within the design and construction phases. Research for BIM-centered IoT deployment in FMS has come to light over the last few years with Wireless Sensor Networks (WSN) technologies that sense and communicate environmental conditions through IoT. This dissertation aims to integrate BIM models and specialized IoT devices from Arduino microcontrollers into a framework that allows for a cohesive workflow for IoT- sourced information to be monitored, analyzed, interpreted, and visualized in a BIM environment. These IoT devices would monitor IEQ aspects of university facilities during the Operations and Maintenance (O&M) stage and visualize the building's environmental condition virtually.

The designed framework looks into integrating BIM platforms (e.g., Autodesk Revit) and Online Database Services (ODS) directly into a visualization platform (e.g., Microsoft Power BI) without having to exchange information between the BIM model and the ODS beforehand. The data provided by both sources collaborate to create informative and interactive dashboards that display environmental parameters in real-time. The parameters measured by the sensors are sent to an ODS, such as MySQL, where the data is stored. The developed system regulates parameters such as temperature, humidity, lighting, and air quality according to pre-defined thresholds of comfortability according to universal standards.

It is expected to develop and implement an Environmental Monitoring System to improve the building performance and control energy consumption of the Library of the Azurém Campus, University of Minho. The developed system is expected to provide a practical tool for data collection and visualization. This contributes to faster and more accurate decision-making, thus maintaining a balanced environmental quality and healthier occupants. The system can be improved by introducing a teaching-learning process that understands, predicts, and optimizes comfort variables. Such systems can be achieved with Machine Learning (ML) algorithms and the use of Artificial Intelligence (AI) and Big Data to create, through BIM technology, a Digital Twin (DT) environment that controls the physical building autonomously.

Keywords: Arduino, Building Information Modelling (BIM), Facility Management Systems (FMS), Indoor Environmental Quality (IEQ), Online Database Services (ODS).

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

This page is intentionally left blank

1. INTRODUCTION

The recent expansion of applied technologies such as electronic microcontrollers, Internet of Things (IoT), and Building Information Modeling (BIM) has fundamentally changed the way we interact with the built environment, especially in the field of Architecture, Engineering, Construction, and Operation (AECO). Despite the low digitization level of the construction industry compared to other sectors (Hagsheno *et al.*, 2019), these technologies have the potential to transform how we interact with the built environment and improve the experience for the end users and service providers (Dave *et al.*, 2018). However, improvements in methodologies and technologies are under development to better manage AECO processes. The implementation of IoT devices and sensors is increasingly being applied in the industrial and educational built environment. The number of sensor installation is increasing at an exponential rate and some estimates suggest that there will be around 30 billion connected devices by 2022 (IoT Middleware Market, 2021).

The breakthrough for improving the efficiency and effectiveness of Facility Management (FM) is the integration with IoT technologies, such as building environmental data (Dave *et al.*, 2018) and the use of wireless sensors to collect data from building systems (Krishnamurthy *et al.*, 2008). Several technologies are introduced such as comfort monitoring through the integration of BIM tools with IoT-based Wireless Sensor Networks (WSN) (Zaballos *et al.*, 2020), or visualizing BIM data with the help of sensor communication protocols (Arduino, Raspberry Pi) to improve energy-saving management (Trabulci, 2020), or the deployment of Digital Twins (DT) and placing them in a Common Data Environment (CDE) that influences projects using Big Data, Machine Learning (ML), and predictive programming (Malheiro *et al.*, 2020).

The object of this study mainly focuses on an environmental monitoring system aimed to maintain positive Indoor Environmental Quality (IEQ) for selected university building indoor spaces. Recent studies have suggested that comfort in educational environments is a critical parameter for the success of learning and the evolution of society (Zomorodian *et al.*, 2016). According to (Charles *et al.*, 2005), maintaining moderate IEQ is the most critical factor contributing to worker productivity, satisfaction, and well-being.

Environmental parameters such as air temperature, relative humidity, atmospheric pressure, light intensity, and VOC gas emissions are measured, monitored, and regulated according to the World Health Organization (WHO) standards and other sources. For example, the acceptable indoor temperature ranges between 21.1 and 24.4 °C (WHO, 1990); the preferred range of relative humidity limits for the thermal comfort is set between 30% to 65% (WHO, 1990); the recommended lighting exposure for comfortable living conditions is around 200 lux (Falkenberg *et al.*, 2019). Certain measurement optimization values might vary with respect to other environmental factors. To illustrate, at 20 °C, the acceptable range is from 30% to 80% Relative Humidity (RH) for respiratory comfort and avoidance of sultriness, while, at 28 °C the acceptable range is from 18% to 50% RH (WHO, 1990). Several studies show that illumination has direct effect on mental and physical health (Osibona *et al.*, 2021).

Studies on BIM and WSN integration for indoor environmental monitoring are common when the issue at hand is air quality (Marzouk and Abdelaty, 2014a; Zhong *et al.*, 2018; Lin and Cheung, 2020), and other studies where the issue at hand is thermal comfort (Marzouk and Abdelaty, 2014b; Natephra *et al.*, 2017) are also common. However, fewer studies where both issued are explored simultaneously (Zaballos *et al.*, 2020). In addition, the increasing number of studies that involve BIM and IoT is substantially higher than studies dealing with BIM and IoT for FM (Mannino *et al.*, 2021). Other studies deal with the subject on a larger scale (Valinejadshoubi, Moselhi and Bagchi, 2021), or use live-sensor data visualization for the gaming industry using immersive and interactive Virtual Reality (VR) environment (Natephra and Motamedi, 2019).

This research proposes to implement IoT technology as a FM tool with the help of BIM tools. The focus of the study is on upgrading two aspects: occupants' comfort and energy efficiency. Monitoring, managing, and optimizing these two aspects is done by the FM team. The proposed framework includes a WSN that consists of multiple environmental sensors integrated with Arduino microcontrollers, an open-source electronic boards company, to create IoT devices and populate them in a university facility building. IoT devices are responsible for detecting comfort levels according to IEQ parameters and energy efficiency through regulating room temperatures in a building. The WSN aims to coordinate the connectivity between the physical and the 3D virtual model through a connected network where the WSN must be: (1) reliable, (2) accurate, (3) flexible in nature, (4) cost effective, and (5) easy to install.

After acquiring the necessary data from the 3D BIM model, the collected data is stored on an Online Database Service (ODS) with an SQL format. When this collected sensory data is combined with the necessary data from the 3D BIM model, they create interactive visualization dashboards in Power BI that display all collected environmental parameters in real-time. The proposed system sends readings, alerts, and all the essential information such as the room ID, room name and location, and current environmental parameter values. The feedback exhibited in the dashboards helps the facility manager with decision-making protocols, resulting in a significantly improved learning performance of occupants and an overall more sustainable and energy efficient facility.

The dissertation mainly aims to:

- Deploy the potential of BIM-centered IoT in the Operations and Maintenance (O&M) phase of the Facility Management industry to create an Environmental Monitoring System (EMS).
- Develop a methodology that can be used as a reference to the implementation of FM ecosystem to other academic facilities.
- Equip a university facility with different environmental sensors that store data in an online MySQL database.
- Use the collected data to control energy consumption, improve the building performance, and increase occupants' comfort level through assisting the facility manager with decision-making protocols.

The knowledge gathered was primarily sourced from journal articles that were published recently (between 2017 and 2021) in order to portray the state-of-the-art information and keep the data relevant, while filtering out the outdated material. Extra journal articles were surveyed from the citations of the original journals, which had further valuable material on the research subject. A selection of E-books, datasheets, previous dissertations, standards manuals, and scientific websites were also consulted as secondary sources, while keeping the most trustworthy to ensure the validity and reliability of the presented information. The bibliographic research has been mainly carried out using Scopus database for the primary sources. The secondary sources have either been cited by the primary sources or are consistent with them.

Having said that, after the introductory chapter, the second chapter explains the different applications of BIM for FM through listing several uses of BIM and giving a brief overview of FM disciplines. In addition, the chapter covers the current state of BIM-FM adoption by referencing previous dissertations and research studies. The discussion shifts to the subject of IoT integration for BIM, technologies used, methodologies for implementation, the present and the future of IoT for BIM. After that the databases used for FM purposes are reviewed, while briefly going over different aspects of a database structure. Finally, the chapter considers several interactive dashboards for business analytics while mentioning their advantages and disadvantages for the workflow.

The third chapter explains the criteria of IoT implementation in a facility building. It also goes over the working process for sensor network. The chapter discusses different kinds of sensors (MKR ENV and BME680) and microcontrollers (MKR WiFi 1010 and Nano 33 IoT) and how they were selected, what alternatives could have been chosen, and explaining in detail how the IoT devices were assembled and wired. Lastly, it explains the Arduino programming of the microcontrollers that serve the monitoring system for environmental data collection for IEQ. The programming part has been developed with the help of Mario Coelho, a postdoctoral researcher in the School of Engineering at the University of Minho.

In the fourth chapter, SQL databases are discussed as well as relational database properties. Then local and online servers are evaluated, and the benefits of an ODS. Moreover, the optimal dashboard layout design is presented and how it serves best the purposes of a Facility Management System (FMS), and how it was set up for the facility manager. The chapter also talks about the implementation of the visualization tool (Power BI) after collecting environmental data from the IoT devices and the necessary parameter data from the 3D BIM model. In the end, the global workflow of the process is examined and reviewed.

The fifth chapter presents a case study for the framework application. The physical building, which is a university library, is described in detail. The previously existing BIM model of the building is then described as well and how it was modified to fit the purposes of the study, where different additional parameters were included, such has *"Room ID"* and *"Thermal Comfort Level"*. Modelling of the IoT devices in Autodesk Revit 2021 is described as well, and what specific properties are assigned to best serve the application.

The conclusion highlights the results and conclusions of the study, while giving a brief summary of the dissertation, and suggests future developments and proposals for later works to be achieved subsequently.

This page is intentionally left blank

2. BIM AND FACILITY MANAGEMENT

FM is stated as an integrated approach to operating, maintaining, improving and adapting the buildings and infrastructure of an organization in order to create an environment that supports the primary objectives of that organization (Laird, 1994). According to International Facility Management Association (IFMA), there are 11 core competencies in FM (IFMA, 2018): Occupancy and Human Factors, O&M, Sustainability, Facility Information and Technology Management, Risk Management, Communication, Performance and Quality, Leadership and Strategy, Real Estate, Project Management, Finance and Business.

FM is generally divided into two categories, which are soft FM services and hard FM services (IFMA, 2018). It is important to distinguish between the two kinds of FM since the way both are handled differs. Soft FM services include space management, cleaning services, security, waste disposal, recycling, plant maintenance…etc. Hard FM services include Heating, Ventilation, and Air Conditioning (HVAC), mechanical services, plumbing and drainage, elevators and escalators…etc. From this data, it can be established that FM is a multidisciplinary topic that requires the collaboration and coordination of different professionals (Chung *et al.*, 2018).

Therefore, hard FM services are related to maintenance activities of the physical and structural building, while soft FM services are responsible for taking care of the environmental aspects in a building's operational phase. This means that soft FM services control occupants' comfort and health qualities. Given that there are clear differences between soft FM and hard FM, these two services are typically overseen by two separate organizations. As a result, the activities accomplished by the two organizations need to be regularly coordinated in the O&M phase to maximize building performance for users.

Presently, a large number of buildings do not have optimal facility management (Omar *et al.*, 2018), since procedures are non-operational and cause there a major lack of data for proper maintenance. While in some other cases, the information collected is not exercised to its full potential even though sensors/IoT devices and databases are in use (Cheng *et al.*, 2016). Many FM daily activities are also manually handled on paper, which consumes more time, increases the likelihood of inaccuracies, and generally causes slower and less efficient coordination and collaboration between different FM services (Xiao *et al.*, 2019). For this reason, the is an essential need for the improvement of FM tools and processes for organizations and facilities in the O&M phase.

2.1. Application of BIM for Facility Management

BIM has been recently increasingly adopted in the AECO industry for information management, as projects grow larger in size, and tasks and responsibilities becoming more complicated for manual supervision. BIM models allow integrated management of information throughout the building's entire life cycle, hence improving FM (Hilal *et al.*, 2019).

In the design and construction phases, BIM instigates work efficiency through the development of 3D models that run clash detection tests before starting the construction, thus ensuring a well-planned and cost-effective project. BIM models also have the property of acquiring a good amount of data during the several phases of the building lifecycle. This helps facility managers during operations management, maintenance activities, environmental analysis, and energy performance simulations (Mannino *et al.*, 2021).

Integrating BIM with FM practices can be outlined in various ways. According to Eastman et al, owners or developers can realize significant benefits on projects by using BIM processes and tools to streamline the delivery of higher quality and better performing buildings. BIM facilitates collaboration between project participants, reducing errors and field changes and leading to a more efficient and reliable delivery process that reduces project time and cost (Eastman *et al.*, 2011). As can be seen in Figure 1, while the design and construction phases only require 1-5 years, the operation phase covers 20 plus years, which is a significantly larger part of the cost and lifecycle of a facility. This means that investing in BIM adoption into FM significantly increases the value of facility documentation when compared to the old manual method. In spite of the additional costs in the design and construction phases, investing the BIM model in the O&M phase has the potential of reducing overall costs (Becerik-Gerber *et al.*, 2012). More recent studies on BIM still confirm its practical impact on FM applications (e.g., (Bortoluzzi *et al.*, 2019; Hilal *et al.*, 2019)).

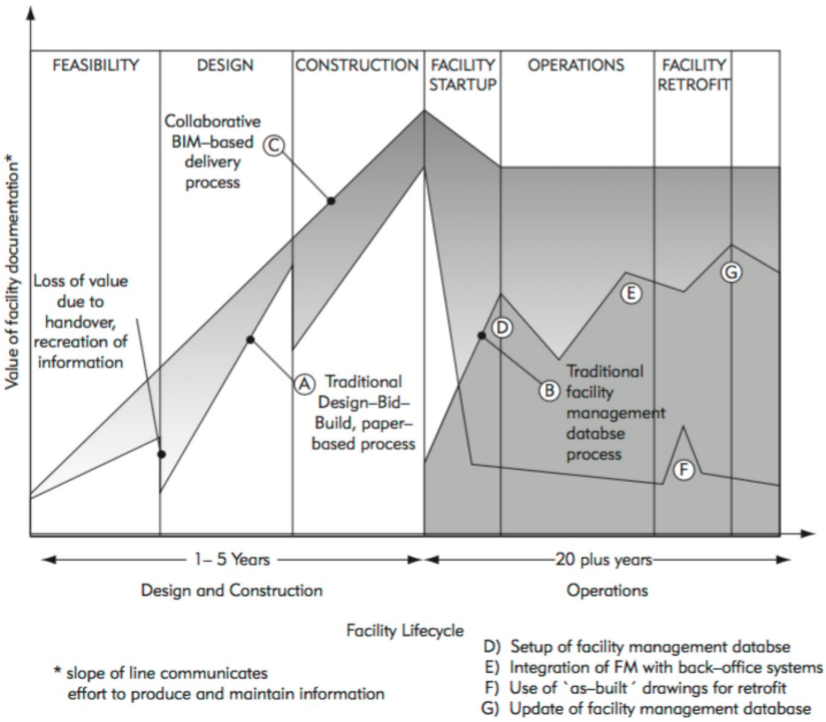


Figure 1 – Value change of facility when adopting BIM with FM (Nguyen, 2016)

Eastman *et al.* (2011) also expect an increased building performance through energy saving, a reduced financial risk through earlier and more accurate financial assessment, better scheduling, more reliable cost-estimations through automatization, and an optimized facility management and maintenance by updated information of the building and its lifecycle added to the BIM-system (Nguyen, 2016).

O&M of a facility is one of the main responsibilities in FM. However, the old paper-based approaches fall behind in most aspects whether it was data management, scheduling, cost efficiency, energy saving, or decision-making for maintenance. This can only be accomplished properly and efficiently by having enough information about the building systems, interior and exterior, and the human factor while also complying to universal or regional standards and regulations. One of the first real-time data connection between BIM and a sensor network was made in 2013 (Rio *et al.*, 2013), where the sensor data in this research were stored within the BIM model. This method showed to be basic and unfavorable since the overflow of data from multiple types of sensors may overwhelm the model or cause the BIM tool to stop responding.

Other studies (Zhang and Bai, 2015; Boddupalli *et al.*, 2019; Valinejadshoubi *et al.*, 2019) propose a framework where an external database is integrated to facilitate storage, sharing, and deployment of the data collected from the sensor network. The authors of these studies also propose the implementation of a visualization platform which includes the BIM model and the external database, enabling the continuous updating of the performance parameters and potentially allowing long-term management. The main purpose of implementing such tools is to expedite decision-making for FM and risk management by avoiding manual human error.

In another study, the concept of Cyber-Physical System (CPS) was introduced (Fitz, Theiler and Smarsly, 2019). The CPS was applied for a structural health monitoring system where its communication-related properties and behavior was described. Then the metamodel that defines the CPS is outlined, where the information is stored, documented, and exchanged on the formal basis of Industry Foundation Classes (IFC). This ultimately facilitates design, optimization, and documentation of the CPS.

One notable research (Cheng *et al.*, 2020) develops a data-driven predictive maintenance approach based on BIM and IoT. The models are developed to predict future conditions of Mechanical, Electrical, and Plumbing (MEP) components based on Machine Learning (ML) algorithms, and the data integration among BIM, IoT networks, and FMS is achieved. This framework has achieved the following modules:

1) Condition monitoring and fault alarming module
2) Condition assessment module
3) Condition predictive module
4) Maintenance planning module

In other studies, the topic was centered around the human factor, with issues such as protecting the environment and the people using the facility and decreasing the risk factor which helps all the stakeholders involved.

This topic focuses mainly on IEQ monitoring. Four research articles address the issue of air quality (Marzouk and Abdelaty, 2014a; Zhong *et al.*, 2018; Ma, Liu and Shang, 2019; Lin and Cheung, 2020), while two articles consider the issue of thermal comfort (Marzouk and Abdelaty, 2014b; Natephra *et al.*, 2017). One article follows a procedure of integrating the captures sensor data from a hot and humid environment with BIM in order to provide health and safety planning solutions for buildings (Arslan *et al.*, 2014).

The follow Table 1 displays additional previous researches on BIM-based sensor integration solutions for IEQ monitoring and their limitations.

Table 1 – Previous researches on a BIM-based sensor-integrated system

Authors	**Sensor types used**	**Objective**	**Issue**	**Limitations**
(Wu and Liu, 2020)	Temperature, humidity, CO_2	Integrating IoT into BIM	Air quality Thermal comfort	Data is stored in a local database (not automated)
(Natephra and Motamedi, 2019)	Temperature, humidity, light	Integrating environmental sensors with BIM	Thermal comfort Lighting	Lack of thermal data retrieval system
(Wehbe and Shahrour, 2019)	Temperature, humidity, light	Integrating IoT into BIM	Thermal comfort Lighting	Data is stored in a local database (not automated)
(Natephra *et al.*, 2017)	Environmental sensor	Integrating BIM geometry data and environmental sensor data	Thermal comfort	Lack of direct integration of sensor data and BIM tool; Humidity didn't influence thermal comfort assessment
(Kensek, 2014)	Light, humidity, CO_2	Link from a Revit model to physical model	Air quality	Used only in the design stage
(Cahill, Menzel and Flynn, 2012)	Temperature, humidity, light, CO_2, occupancy	Optimization of building operations	Air quality Thermal comfort Occupancy	Conceptual only; No digital implementation
(O'Flynn *et al.*, 2010)	Temperature, humidity, light, motion and occupancy sensors	IoT for building energy management application	Air quality Thermal comfort Occupancy	Mostly focused on the hardware; Conceptual in the BIM stage
(Yin, 2010)	Temperature, humidity, CO_2	Monitoring a building's operation and energy performance	Air quality	Conceptual; Lack of visualization
(Katranuschkov *et al.*, 2010)	Temperature, humidity	BIM-based generation of multi-model views	Thermal comfort	No link between BIM and IoT

While BIM is primarily aimed to be a developing technology for managing data related to a building project, there still exist many challenges in the management of data and existing specifications. This includes the size of data sets, Level of Development (LOD), interoperability with currently existing formats (Gerrish *et al.*, 2015), and Level of Information Need (LOIN) as described in the EN ISO 19650-1 and EN 17412-1:2020 international standards as well. Challenges from using existing standards and specifications such as IFC in modelling monitoring systems include the lack specific entities and attributes for modelling, lack of directives for data management and visualization, and lack of guidelines for connections with external sources of data (Davila Delgado *et al.*, 2018). When a BIM model is transformed into an IFC model, it loses fundamental proprietary information. This has constantly been an issue with exporting into IFC. From the previous researches mentioned, most used the IFC standards in their framework, which still has issues, as mentioned above, that haven't been solved.

As can be concluded from in Table 1, IoT technology in the context of BIM has not been exploited to its full potential, even though it has been an increasingly used aspect in the AECO industry. Some researchers studied the integration of BIM into IoT systems, mostly aimed at building conceptual framework. Other researchers developed an automated BIM-based alarm system, primarily for thermal comfort monitoring purposes. One study (Wu and Liu, 2020) developed a BIM-based visual energy conversion system, where they developed a system to integrate BIM into IoT for IEQ monitoring. However, their system lacked autonomy in the visualization aspect since a local database (Excel) was utilized, which needs manual update for the collected data.

In another study (Natephra and Motamedi, 2019), a method for an automated live sensor data visualization was introduced. This method visualizes IEQ conditions based on environmental sensors and the use of BIM. In this study, the authors used an Arduino microcontroller and Dynamo to record, collect, and transfer sensor data from the physical building into the BIM model. This method lacked a retrieval module for previous thermal condition data of building spaces.

BIM is also used in another study to support decision-making for comfort levels in a building (Wehbe and Shahrour, 2019). The authors managed to link IoT with BIM, but the collected sensor data was stored in a local database, slowing the real-time updates of thermal comfort parameters, which ultimately reduces the collaborative capabilities of using BIM.

One of the research studies that managed to implement a BIM-based thermal comfort monitoring tool, where BIM geometry data and environmental sensor data are integrated for evaluating the indoor thermal comfort level (Natephra *et al.*, 2017). In this study, although the relative humidity data was collected, it was not taken into consideration when assessing thermal comfort in an air-conditioned building. The study also shows no direct integration of sensor data into the BIM model. This is not necessarily a disadvantage in the framework, since the data collected from the sensor and the BIM model geometry data can be both linked at the visualization level, thus preventing data redundancy.

Some aspects that have not been adequately adopted in previous studies include: (1) the instant synchronization between the physical model and the 3D BIM model; (2) the operative connection of sensor data into an external online database through IoT technology; (3) the application of a visualization monitoring system based on a 3D BIM model for an EMS.

2.2. IoT integration for Facility Management

The Internet of Things (IoT) is a system characterized by the ability of devices to communicate and coordinate with each other through the Internet to improve the quality of life. It allows autonomous decisions to connected devices through algorithms and machine learning, and can properly inform users to make the best decisions (Rizal and Hikmatyar, 2019). IoT integration with data networks is an advanced approach for the optimization FM activities, including document management, historical data cataloging, logistics and material tracking, building component lifecycle monitoring, and building energy controls (Wong *et al.*, 2018).

The Internet of Things aims to create and improve operations in different fields by providing business models that contribute to raising business efficiency, reducing costs, promoting innovation, creating new business opportunities, increasing profitability and offering new visions for organizations in their operations. These organizations contribute in turn to providing advanced and innovative solutions for the general public in many vital sectors such as industrial, energy, transportation, health, education, tourism, agriculture, environment, security, entertainment...etc. All this is accomplished to promote the improvement of the quality of life.

According to the report issued by the GSM Association, IoT reached 12 billion connected devices in 2019 and is expected to reach 25.1 billion connected devices by 2025. According to Ericsson forecasts, it is estimated that there would be over 30 billion connected devices in use by 2022, of which around 18 billion would be used for a diverse range of IoT applications (IoT Middleware Market, 2021).

Technologies for IoT include Radio Frequency Identification (RFID) and sensor systems. RFID is a wireless communication technology. The main components of an RFID are: (1) a tag that contains electronically stored information; (2) and a reader that is a device which has one or more antennas that emit radio waves and receive signals through electromagnetic fields.

A sensor, which is the contemporary term for 'sensor system' or 'sensor element' (Figure 2) (National Academies Press, 1995), is a device that detects environmental parameters and converts this physical parameter into a measurable voltage (analog or digital) that can be displayed, transmitted, or analyzed for further purposes.

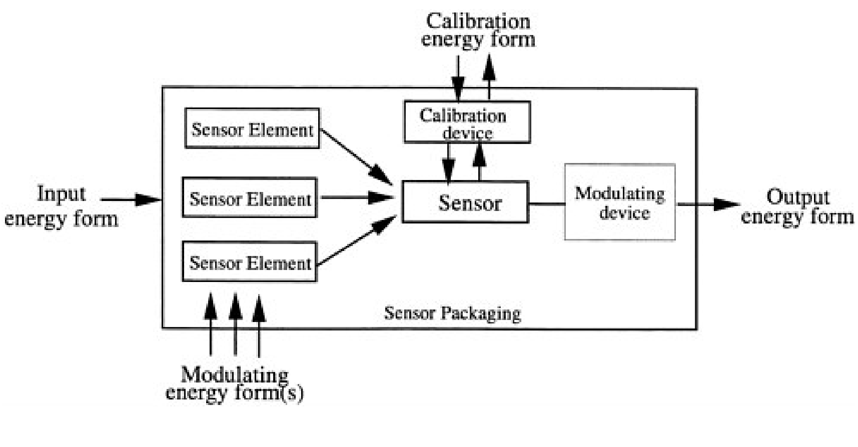


Figure 2 – Anatomy of a sensor system (National Academies Press, 1995)

Both RFID and sensor systems have great potential for FM activities and processes, such as building security and document management, building component lifecycle tracking, building energy control, and alert monitoring (Gubbi *et al.*, 2013). However, this dissertation will go in-depth only into sensors and sensor networks. Traditional sensor networks relied on USB wired connection sensors. Modern sensor networks switched to wireless technology through a WSN that offers flexibility and cost-efficiency. The potential future of WSN is equivalently related to the expanding use of BIM throughout a building's lifecycle (Underwood and Isikdag, 2011).

WSN is considered to be a more advanced and enhanced technology when compared to wired connection. This is shown on the social level through the widespread adoption of WSNs in the AECO industry, especially in the FM disciplines. On a more methodological level, the predominance of WSN exists due to: (1) the expendability of a fixed infrastructure when setting up network; (2) the cost-efficiency in the deployment; (3) the uncomplicated configuration without the need for cables; (4) the flexible change of location for sensors making their installment and removal easier; (5) the shorter installation time. WSN still embodies some drawbacks (Kaur and Monga, 2014) such as the slightly lower speed, the weaker security, and the limited power supply. However, innovative solutions constantly being developed for these disadvantages, and the benefits outweigh the problems.

Environmental sensors such as temperature, humidity, and atmospheric pressure are used to capture real-time values for related comfort parameters. The sensor data acquired is stored in an online database server. Afterwards, the data is exported from the online server and uploaded into a visualization dashboard tool that allows the user (e.g., facility manager) to observe the changes in the environmental values in real-time.

When measuring IEQ, there are many parameters to be considered. This dissertation will carry out an Environmental Monitoring System (EMS) process for monitoring the quality of the environment. The parameters that are taken into account for this process are temperature, humidity, pressure, light intensity, and Volatile Organic Compound (VOC) gas.

A study by Valinejadshoubi *et al.* (2021), who based their study on the ASHRAE Standard 55-2017 (standard widely used in North America) for occupants' thermal comfort in a building, shows the acceptable temperature ranges in summer and winter as per the Relative Humidity (RH) (Table 2). For the purpose of this study, no critical changes were made in the ASHARE Standard 55-2020 version.

Table 2 – Acceptable operative temperature ranges (ASHRAE, 2017)

Season	Relative Humidity (RH) (%)	Acceptable Operating Temperature (°C)
Summer	30=<RH<60	24.5 – 28
	RH=60	23 – 25.5
Winter	30=<RH<60	20.5 – 25.5
	RH=60	20 – 24

The quality of lighting, to find or provide well-lit places, primarily determines the human satisfaction and in a further sense well-being with the natural and artificial lighting systems (WHO, 1990). The lighting of an indoor space depends on several factors such as the natural sunlight, the artificial lighting, and the daylight factor (DF) – which is the outdoor-indoor illumination factor mainly influenced by the size of the window and the room proportions.

Air quality is influenced by different indoor pollutants. The final indoor health impacts indoors are the resultant combination of indoor and outdoor air quality factors acting together within the interior (WHO, 1990). However, it is usually the added indoor pollutants, which cause the greatest deterioration in indoor air quality (WHO, 1990). Indoor air pollutants can be classified into 4 types:

1- Gaseous matter, such as CO, NOx, VOC.
2- Liquid matter, such as droplets produced by humans (sneezing or coughing)
3- Particulate matter, such as smoke, dust, or bacteria.

This study will focus solely on the VOC gas pollutants in the EMS. To settle on a suitable sensor type, several factors are taken into consideration:

- Measurement range;
- Accuracy;
- Response time;
- Power consumption;
- Technology used;
- Cost.

A literature on the state-of-the-art of sensors and environmental monitoring technologies in buildings (Hayat *et al.*, 2019) provides an insight into various sensing and environmental monitoring technologies commonly deployed in buildings by surveying different sensor technologies, and the key selection parameters for optimal sensor placement. The results are shown in Table 3 below.

Table 3– Summary of key characteristics of several environmental sensors (Hayat *et al.*, 2019)

Parameter.	Sensor	Measurement Range	Accuracy	Response Time	Power Consumption	Applications/Technology	Cost
Temperature	Thermocouples	−100–500 °C	±1–4 °C	5–80 s	Low–High (0.5 µA–30 mA)	BMS, HVAC/Wired, Portable	$6–50
	RTDs	−50–250 °C	±0.2–1 °C	1–8 min	High during measurement (1.5–100 mA)	BMS, HVAC and Visualisation/Wired, Wireless	$30–100
	Thermistors	−50–130 °C	±0.05–0.5 °C	0.2–10 s	High during measurement (1–80 mA)	BMS, HVAC and Visualization/Wired, Wireless	$20–70
	IC sensors	−40–150 °C	±0.5–1 °C	0.5–100 s	Low (0.5–100 µA)	BMS, HVAC and Visualisation/Wired, Wireless	$1–15
Carbon emissions	NDIR (CO_2)	0–10,000 ppm	±30–200 ppm	30–100 s	Low–High (20–200 mA)	Airflow Control, Monitoring/Wired, Wireless	$100–600
	MOSFET (CO and VOC)	400–20,000 ppm	±30–100 ppm	50–60 s	High (typ. >50 mA)	Airflow Control, Monitoring/Wired, Wireless	$25–250
	Electrochem. (CO and VOC)	0–1000 ppm	±0–30 ppm	10–60 s	Low (30 µA–10 mA)	Airflow Control, Monitoring/Wired, Wireless	$100–650
Humidity	Capacitive sensors	0%–100% RH	±0%–5%	15–90 s	Low (2 µA–4 mA)	BMS, HVAC and Visualisation/Wired, Wireless	$40–200
	Resistive sensors	5%–90% RH	±1%–10%	10–60 s	Low (0.5–5 mA)	BMS, HVAC and Visualisation/Wired, Wireless	$25–170
Light	Photores. Photodiode	–	±5%–10% of reading ±5%–10% of reading	5–20 s 1–10 s	Low–High (10–60 mA) Low (100 µA–5 mA)	BMS, HVAC and Visualisation/Wired, Wireless BMS, HVAC and Visualisation/Wired, Wireless	$50–200 $15–65

Table 3 displays the state-of-the-art commercially available sensors and monitoring devices in terms of the factors mentioned earlier, along with their key characteristics. Most sensor types can be used for this study as most of them use wireless technology, have the required measurement ranges, response time, and accuracy, and the power consumption is relatively low. All these characteristics are favorable for implementation target of the sensor network.

2.3. Databases used for FM purposes

FM activities typically take information from multiple sources, and integrate these data to perform different tasks and objectives. These sources are an organized collection of structured information called a database. This information used to be manually stored on paper in the past, but at the present time it is generally stored electronically in a computer or a server (local or online). The facility manager controls the database through a Database Management System (DBMS) where the data can be accessed, modified, updated, and organized either locally or remotely.

A BIM model can be used to populate a FM database in order to increase efficiency, accuracy, and coordination between different FM tasks. FM databases are also developed to allow for automatic and real-time maintenance information updates from the BIM database (Liu and Issa, 2012). The BIM data can be used for space management and environmental monitoring (soft FM services) or for maintenance activities of the physical and structural building (hard FM services).

Because of the large number of various types of information collected and recorded by the EMS, a database is necessary to manage this large amount of input. This database requires a database model to determine its structure. A database model affects how the data can be stored, organized, and manipulated. Hierarchical and network data models are earlier developments of data models than the relational model and are often replaced by relational models (Gray *et al.*, 2017).

For the purpose of EMS data storage, relational data models are largely used. This type of database models uses tables to represent both data and the relationships between data. Each table has a number of columns that specify the different categories of attributes. Entries are added as rows to represent the data sets. Relationships between tables is achieved by relating columns in multiple tables.

Relational models for database management were invented in 1970 by the computer scientist E. F. Codd. Two earlier examples of successful implementation of relational databases used to monitor environmental data are discussed. In the first example (Wösten *et al.*, 1999), a relational database was developed in order to store various soil properties from different sources in a central area for ease of use. A relational data model was selected in order to provide flexibility in data extraction with the diverse range of data being collected. The other example (Peacock *et al.*, 2007) documents a database development for the storage of large amounts of data relating to trees in the Amazon forest. A user interface is developed for visualization, modification, and addition of data to the system. An Entity Relationship Diagram (ERD) of the relations between the tables of the database is shown in Figure 3, with a list of their respective fields.

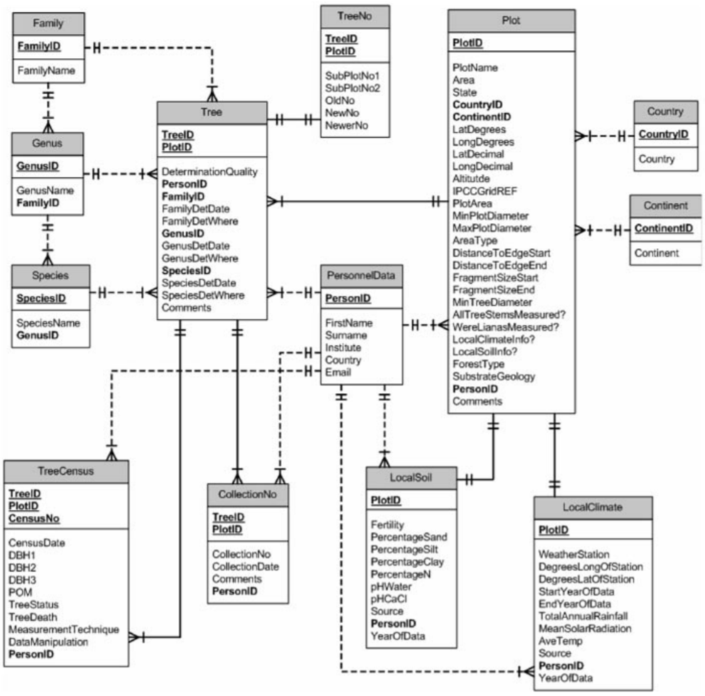


Figure 3 – ERD of the relational database model (Peacock *et al.*, 2007)

As can be seen in Figure 3, primary keys are underlined, and items in bold are required. Solid lines represent links between primary keys. The relationships between tables can be one to one (e.g., *'TreeNo'* and *'Tree'*, *'LocalSoil'* and *'Plot'*) or one to many (e.g., *'Family'* and *'Tree'*, *'Country'* and *'Plot'*). The type of relationship is shown by symbols of one or two stripes through the ends of the connecting link.

A more recent example for relational database implementation (Valinejadshoubi *et al.*, 2021) that developed a database model for recording and collecting temperature and humidity parameters from sensors installed in an indoor environment (Table 4). To insert the sensed data into a database (MySQL database in this study), a schema and tables and all essential parameters are defined. The tables store the sensor data received from the EMS network.

Table 4 – MySQL database model specifications (Valinejadshoubi *et al.*, 2021)

Tables	**Parameters**
waspmote_temperature_sensor	'Record_ID', 'Sensor_ID', 'Sensor_Value', 'Recorded_AT'
waspmote_humidity_sensor	'Record_ID', 'Sensor_ID', 'Sensor_Value', 'Recorded_AT'
temperature_measurement_history	'Record_ID', 'Sensor_ID', 'Sensor_Value', 'Recorded_AT'
humidity_measurement_history	'Record_ID', 'Sensor_ID', 'Sensor_Value', 'Recorded_AT'
room	'Room_ID', 'Room_Name', 'Occupancy', 'Thermal_Condition', 'Latest_DateTime'
room_info	'Room_ID', 'Room_Name', 'Thermal_Condition', 'Latest_DateTime'

To construct a comprehensive database model for the whole sensory system, the information on the corresponding rooms and their conditions must be determined. Figure 4 shows the Entity-Relationship Diagram (ERD) of the proposed database model.

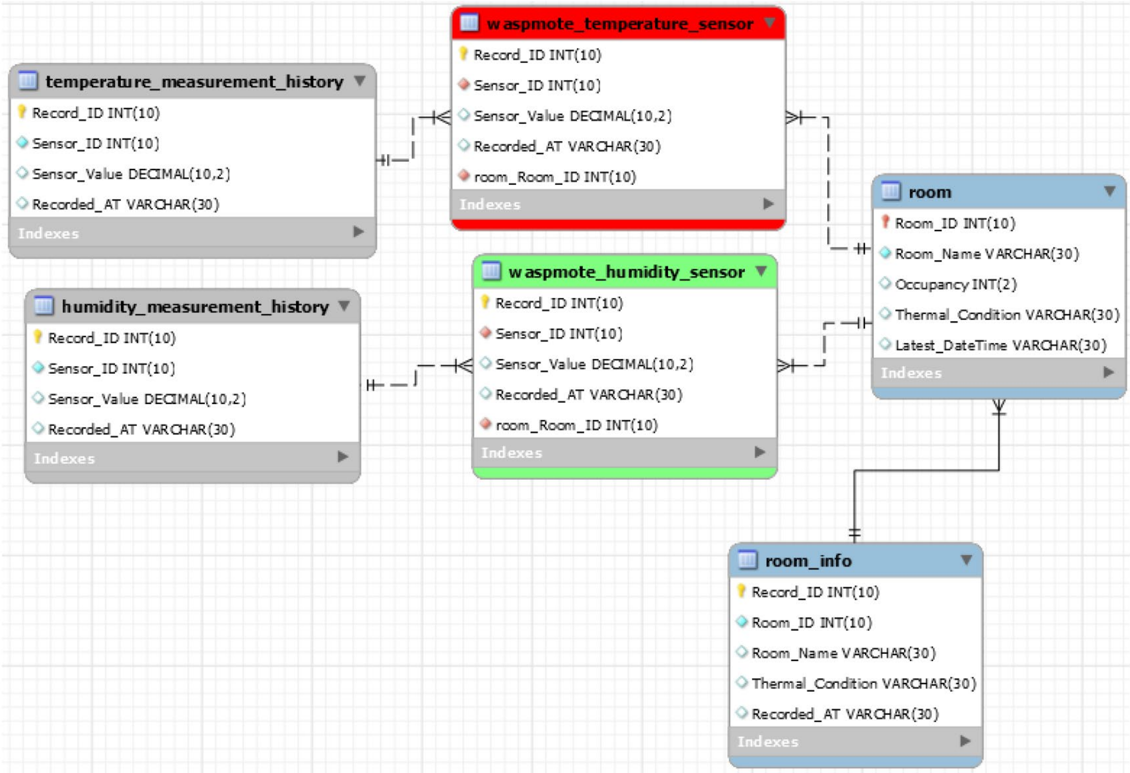


Figure 4 – ERD of the proposed monitoring system database (Valinejadshoubi *et al.*, 2021)

The database model for Valinejadshoubi *et al.* was created to support the developed system utilizing modules in Dynamo, which defines the workflow for integrating BIM into thermal comfort monitoring.

Structured Query Language (SQL) is a programming language used to communicate with data stored in a Relational Database Management System (RDBMS) (Codeacademy, 2021). SQL was initially developed at the International Business Machines Corporation (IBM) by Donald Chamberlin and Raymond Boyce after learning about the relational model from E. F. Codd (Chamberlin, 2012) in 1974 (Chamberlin and Boyce, 1974). To settle on an appropriate SQL platform, it has to comply with the published SQL standard. The database platform must support relational tables, relational processing of sets, and Atomicity, Consistency, Isolation, and Durability (ACID) transactions (PerformanceDBA, 2019). This automatically excludes many open-source platforms that don't follow the standard principles. If scalability or performance are required, then the Server Architecture is an important factor. According to PerformanceDBA (2019), SQL platforms can be categorized into four types:

1) High-end, Commercial, SQL Compliant, such as *Sybase ASE* and *IBM DB2*.
2) Commercial, SQL Compliant, such as *MS SQL Server*, a platform with good architecture.
3) Commercial, SQL Non-Compliant, such as *Oracle*.
4) Non-Commercial, SQL Non-Compliant, such as *PostgreSQL* and *MySQL*.

2.4. Interactive dashboards for business analytics

For the purpose of this dissertation, the usual IoT-BIM integration workflow can be briefly summarized by: starting with installing IoT sensors in building facilities; building the hardware and networking of the system; setting up IoT sensor node dashboards; and integrating IoT data with the BIM model data and visualize it on dashboards. Therefore, following the recording and collecting of the environmental parameters for the EMS is the implementation of a dashboard for visualization of the stored data from the sensors. The facility manager has access to the data provided by the sensors and to the local and cloud-based visualization dashboards.

Sending real-time notifications to building supervisors and the facility manager through the wireless connected devices, such as smartphones or computers, is critical for taking necessary actions if the operating temperature data does not fall within the predefined acceptable range (Valinejadshoubi *et al.*, 2021). For this purpose, a workflow must be outlined to develop a system that sends the room data from the BIM model along with the sensor data retrieved from ODS, and combine these databases on the level of the visualization platform.

A number of applications compatible with BIM-IoT integration were considered for the study objectives for sensor data collection and real-time data visualization and analysis. One of these applications is ‘usBIM.IoT’ (Figure 5) which integrates the BIM model with IoT systems and sensors, views sensor data on the model in real-time, and obtains a dashboard with information synthesis and sensor controls. This application is still in its development phase, and is not suitable for production, professional use, or even testing during the time of the study.

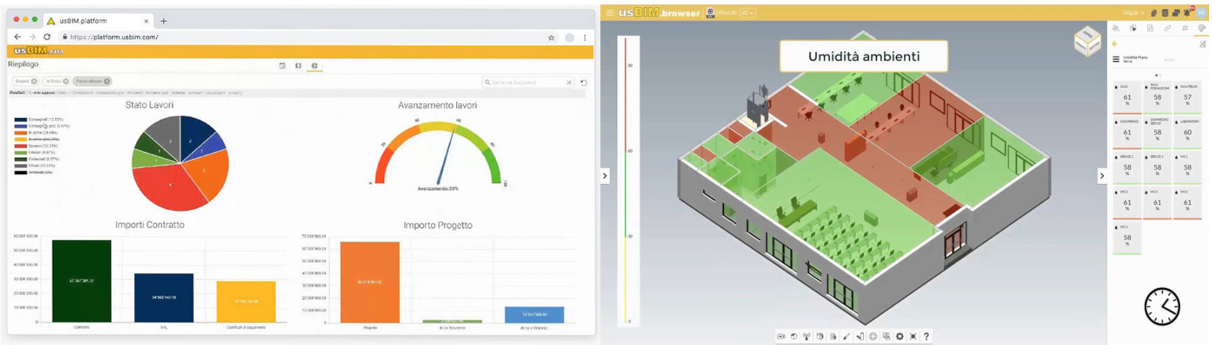


Figure 5 – usBIM.IoT platform data visualization (left) and 3D model visualization (right) (accasoftware.com, 2021)

An alternative tool is ‘Azure IoT Hub’ and an Azure web app (Figure 6) that provides two-way communication between IoT application and the devices it manages, analyzes and manages incoming data, and sends commands and notifications as needed. This tool may be more expensive and less flexible for startups and smaller low-budget facilities and companies with low-cost devices in the field. Therefore, it is not used in this workflow for assessment.

Another option developed by the software company Nomitech is ‘CellBIM’ (Figure 7) that imbeds the BIM model inside of Microsoft Excel and creates interactive spreadsheets for visualization. ‘CellBIM’ can provide graphs, charts, and views that display different types of data, to create a FM tool for monitoring sensor data in real-time. However, this option is more expensive than the options mentioned earlier and the subscription is viable for one person only.

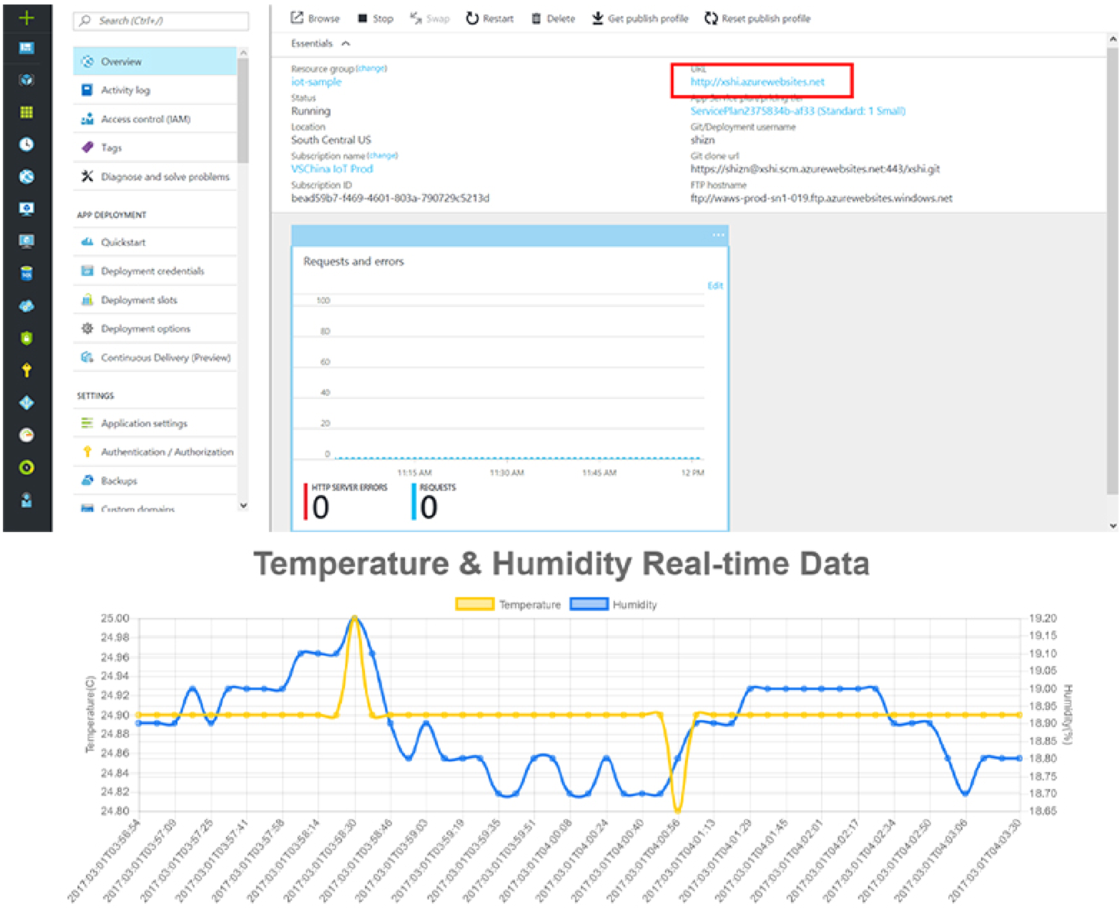


Figure 6 – Azure IoT Hub platform (top) with a web app visualization (bottom)

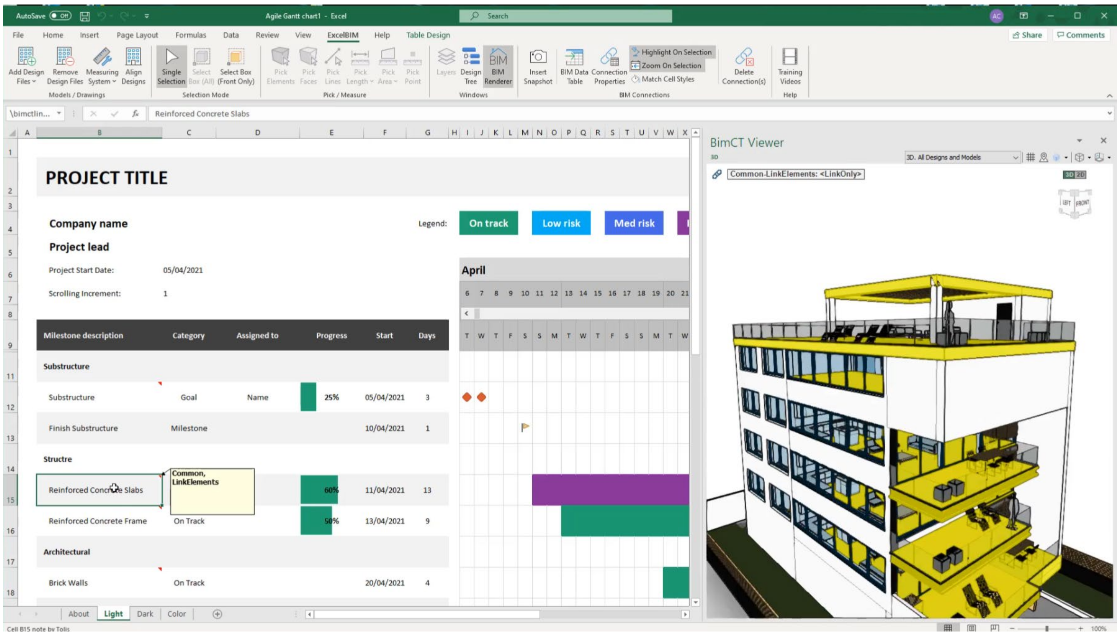


Figure 7 – CellBIM spreadsheet in Excel with the 3D model representation

One further solution is the use of Microsoft Power BI which is a software that enables unifying different data sources into cohesive and interactive dashboards or reports to aid with FM tasks and responsibilities. Power BI has a user-friendly visualization interface that allows beginners to master the use of its different tools and design customized views for dashboard to meet their required purpose. The aim of utilizing Power BI in this study is to visualize the data collected through the EMS combined with the BIM model data to facilitate building FM tasks, and to create a real-time notification system that connects to one or more devices which alerts the facility manager of the proper actions to take in a short period time.

Power BI allows to monitor and analyze the collected sensor data throughout a timeline or a specified interval of time. Multiple factors such as temperature, humidity, and light exposure can be monitored and analyzed simultaneously through different gadgets to ensure a satisfactory atmospheric IEQ (Figure 8). Other gadgets can be used to display the average temperature and other parameters, as well as the minimum or maximum limits during a defined time frame.

When introducing 'Tracer', a toolkit for creating interactive 2D and 3D diagrams of BIM models, Power BI operational FM capacities expand tremendously. Tracer includes an Autodesk Revit add-in that allows users to export diagrammatic geometry and data for BIM objects including rooms, areas, spaces, and family instance locations (ProvingGround, 2020). In addition, Tracer allows to harvest Revit element data into an open relational database, connect Revit data to a Power BI report, and visualize the Revit geometry and distinguish between rooms, areas, and other elements with rendering.

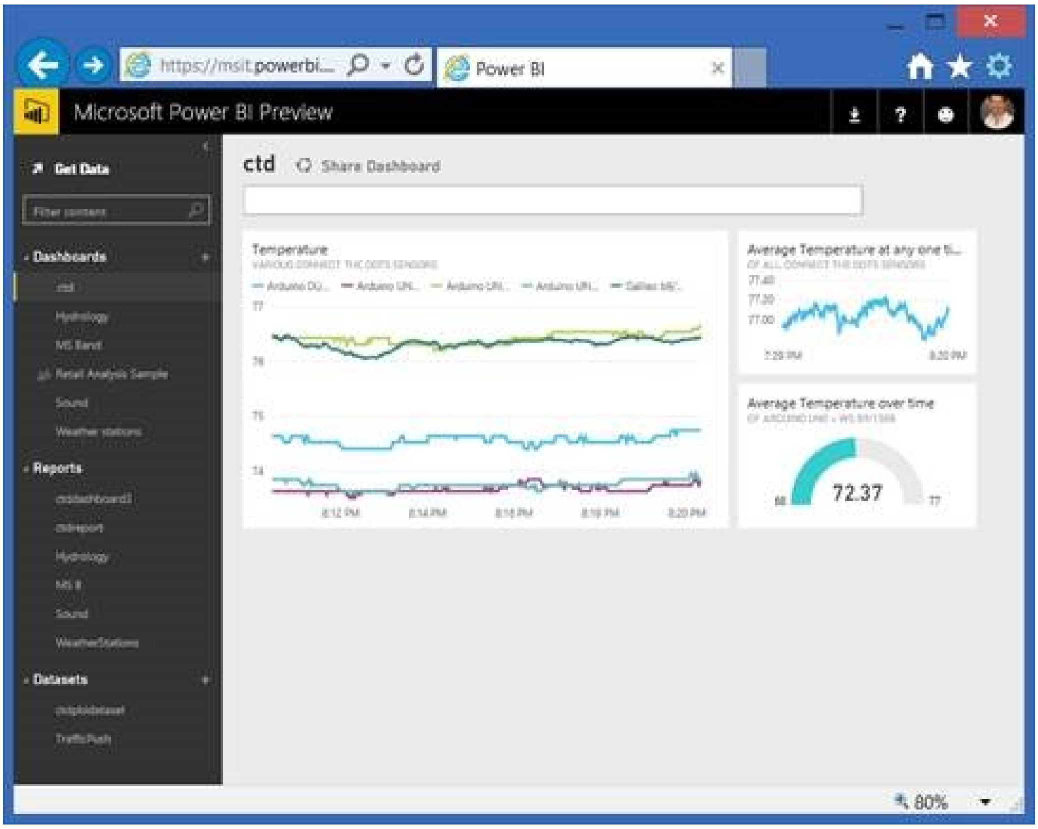


Figure 8 – Power BI dashboard displaying multiple gadgets and parameters

Tracer has a 15-day trial application version that includes Revit toolkits and is able to harvest 2D and 3D data from a Revit document. An IFC toolkit was recently introduced that collects 3D data from exported IFC models from BIM software including ArchiCAD, Revit, Tekla, and others. Both toolkits produce dynamic 3D Power BI reports that can be used for FM. The trial version was the basis of assessment in this study, which showed the basic functional principles of the platform.

This shows that Power BI makes it possible for an automated integration of data from the Revit model and an SQL database, into the visualization platform. This gives facility managers the opportunity to access the dashboard locally or remotely to add, edit, or analyze the data, all while receiving up-to-date information from the database and the BIM model that assists FM decisions for maintenance.

One open-source IoT platform is 'ThingsBoard' (Figure 9) which can collect, process, visualize and manage data. It provides device connectivity through industry standard IoT protocols and supports both cloud and local deployments (Desogus *et al.*, 2021). The platform also allows data monitoring through built-in or customized widgets and flexible dashboards. For an automated data exchange between the IoT sensors and the BIM model, Revit must be integrated with the Dynamo visual programming platform with a specific Application Programming Interface (API) that can manage sensor information and position in a space.

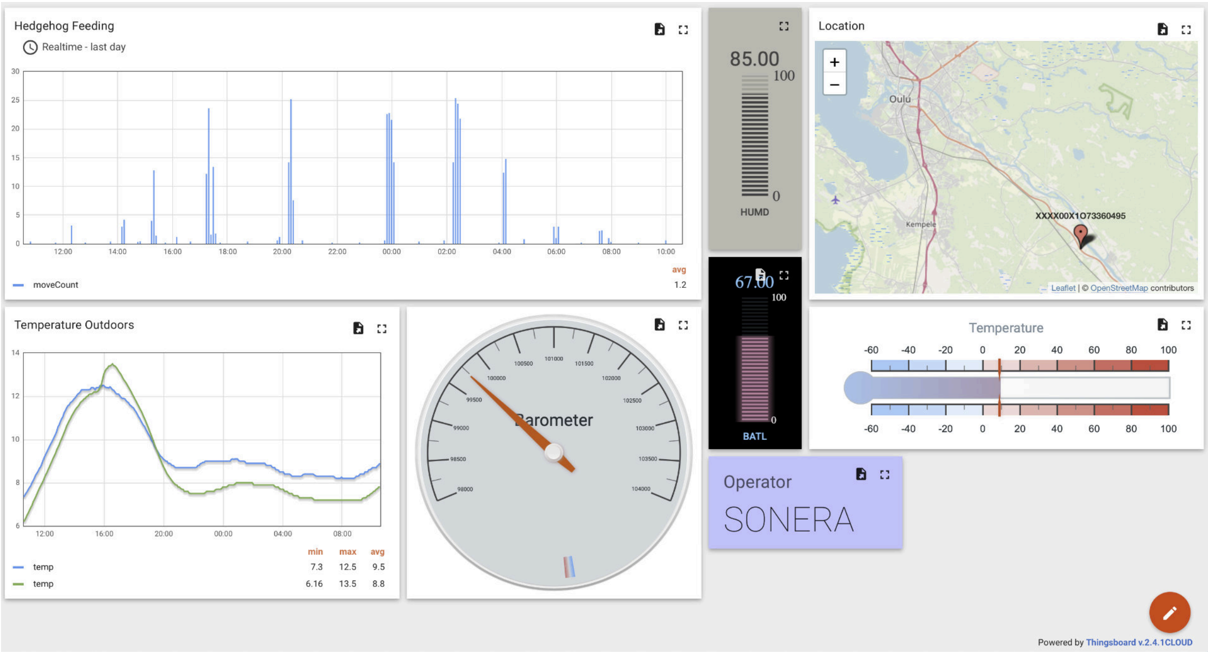


Figure 9 – ThingsBoard visualization dashboard (Haltian, 2021)

This page is intentionally left blank

3. IMPLEMENTATION OF IOT DEVICES

This chapter explains the criteria through which the implementation of the IoT devices took place in a private indoor location as a prototype for university facility buildings that needs to be verified at the library building of the Campus of Azurém of the University of Minho. The public building consists of different spaces that serve as meeting rooms, offices, lounge areas, and an open common area on each level. For the framework, one IoT device in placed in some space of the library to measure, monitor, and record different environmental parameters such as temperature, humidity, pressure…etc. Each IoT device is composed of an environmental sensor/shield that collects environmental data, and a microcontroller that processes the data and sends it to the MySQL database.

3.1. Sensor networks

This section highlights the purpose of a wireless sensor network in IoT technology to build a complete system architecture that can be used as a fully functioning facility management tool.

In order to have a continuous flow of the latest environmental information on a dashboard report, all devices and sensors must be uninterruptedly connected to a network. A Wireless Sensor Network (WSN) is a system that incorporates a wireless network of spatially distributed autonomous devices or sensors (often called nodes) to provide real-time monitoring of physical or environmental conditions (such as temperature, sound and pressure) within built environment systems (Riaz *et al.*, 2017). The use of a WSN eliminates concerns related to scalability and dynamic reconfiguration because it allows remote monitoring of the IoT devices through sensor node communication.

A sensor network is a group of sensors that are being used to perform a specific task like monitoring the environmental data of a specific space or location. There are three steps involved in the working process of sensor networks as shown in Figure 10. First, the sensors are to be installed in the area where the environmental condition are to be monitored. In this step, the sensors available in the market are selected based on their different types, applications, and capabilities. There are different sensors to monitor various parameters like temperature, humidity, occupancy, noise…etc. These quantities measured by the sensors will be converted into digital values by some extra modules like ADC (Analog-to-digital converter). After that, sensor collected data is finally transmitted to be visualized.

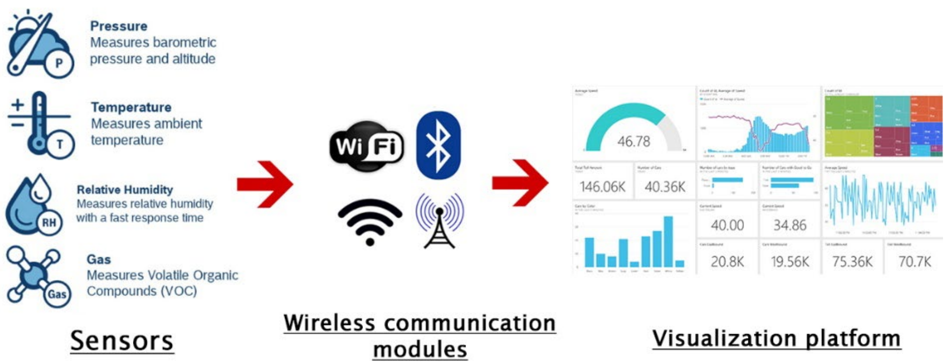


Figure 10 – Working process for sensor network

In between this end destination, there are wireless communication modules that are able to perform this procedure wirelessly. From Figure 10, the middle process uses a wireless communication technology to transfer the information from one device to another. There are different types of wireless communication modules, but only WiFi will be used in the WSN to transfer the sensory data to a system which is the third step that is involved in the working process of sensor network for the purpose of this research.

The third step in the working process involved in the sensor network in the visualization application. There are several applications that allow to visualize this collected information and present it for further analysis or for specific actions such as controlling devices through actuators based on the values of the collected environmental data. This step is further discussed in Chapter 4.

3.2. Microcontrollers

This section focuses on the microcontrollers used for the IoT implementation and the criteria adopted in the selection process.

With no previous experience with electronic microcontroller software and hardware, the most suitable choice when settling on an electronic board was to pick a well-known platform – Arduino, that has a large user base that is able to provide through online forum discussions answers for obstacles encountered during the project assembly and programming phase. Arduino provides this possibility as it is an open-source platform that provides pre-assembled microcontroller boards with some additional components to perform required functions.

Due to the wide spread of WiFi technology and to ease the integration on an existent infrastructure that is also supplied by a fully integrated wireless network of communication, all microcontrollers to be taken into consideration must be able to connect to a WiFi network. Not only does this ensure a wider range for the monitoring system, but it gives more flexibility to the microcontroller mobility as well. The two boards selected for the purposes of FM IoT applications are Arduino MKR WiFi 1010 and Arduino Nano 33 IoT, which were ultimately selected due to their WiFi connectivity feature, low energy consumption, and low price point. The use of an Arduino simplifies the process of working with microcontrollers and offers some advantages to the users over other systems due to its simple, clear programming environment, open-source and extensible software and hardware. Arduino platform has good specifications e.g., cheap, easy to use and wide varieties of shields that have been emerged with many different purposes (Villa *et al.*, 2021).

3.2.1. MKR WiFi 1010

Arduino MKR WiFi 1010 (Figure 11) is a proper place to start when dealing with IoT projects. It allows users with no previous experience in networking applications to design practical and cost-effective projects that require direct WiFi connectivity. This will ensure that results from the monitoring system are observed in real-time. Bluetooth is also supported by the board.

The MKR WiFi 1010 comes with pre-assembled headers and easy-to-read labels for the pins on the sides and the bottom of the board. On the side can be seen the Lipo battery connector that charges the battery as long as it’s connected to a USB power source, which will automatically switch off to the battery power when the USB is unplugged. It is designed to work with the IoT cloud which is a system that allows the user to build their own dashboards as well as managing their IoT devices from a centralized location.

Figure 11 – Arduino MKR WiFi 1010 microcontroller board

3.2.2. Nano 33 IoT

Arduino Nano 33 IoT (Figure 12) is a member of the Nano 33 family that uses the SAMD21G 32-bit microcontroller. This board has the smallest size when it comes to getting started with Internet of Things. It is a microcontroller board that offers a practical and cost-effective solution for IoT projects especially since it is integrated with WiFi and Bluetooth connectivity due to the ublox NINA-W102 module which is a multiradio Microcontroller Unit (MCU) and a radio responsible for wireless connection (U-blox AG, 2021). However, Bluetooth connectivity will not be part of the use case of the monitoring system since all devices will be connected to a wireless network. The Nano 33 IoT is also compatible with the Arduino IoT Cloud, which facilitates the development of IoT projects by connecting them to a supported cloud service dedicated for such boards.

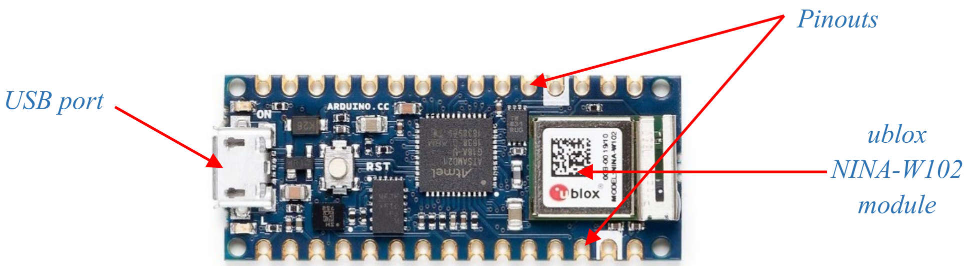


Figure 12 – Arduino Nano 33 IoT microcontroller board

The microcontroller on the Arduino Nano 33 IoT runs at 3.3V, which means that you must never apply more than 3.3V to its Digital and Analog pins. It should be mentioned that care must be taken when connecting sensors and actuators to assure that this limit of 3.3V is never exceeded (*ABX00032*, 2021).

This board is a smaller form factor than the MKR WiFi 1010. In exchange for the reduced size, the battery connector is sacrificed in the Nano 33 IoT board. Otherwise, both boards operate more or less the same.

3.2.3. Alternative boards/devices

Raspberry Pi is a single board computer that works on a modified Linux OS version depending on the use case. It has its own storage space via a microSD card to run the operating system. What makes this device versatile and sets it apart from normal computers is its GPIO (General-purpose input/output) ports. These ports are physical pins on the board itself that are programmable, and they can either send voltage to whatever is connected or even receive data through electric impulses which can be utilized in checking sensor data for an environmental monitoring system.

The use of this technology was considered in this study. However, upon closer inspection, with requirements such as a very low energy footprint for sending data from a sensor while running on a battery or a power source for a longer period of time, the Raspberry Pi was most likely not the best option. In this case, smaller Arduino or ESP boards would fulfil the user's needs more suitably. This means that the Raspberry Pi is able to perform the task successfully but it would not be using its full capabilities. Some of the Raspberry Pi's projects include:

- Storing sensor data and controlling actuators for a smart home
- Setting up home security surveillance
- Creating retro video games
- Building visualizer that can be connected to a monitor or a TV

This only shows that with the Raspberry Pi's larger size, added cost, greater energy consumption, and additional practical features and customization, using it for simply monitoring sensing devices in this study would be inefficient on virtually every aspect. Other boards were considered in the initial stages like the MKR WiFi 1000 and the Portenta H7 but the MKR WiFi 1010 and Nano 33 IoT were ultimately selected because of their lower price range, upgraded features, or higher efficiency.

3.3. Sensors

This section aims to determine the sensors which are most suitable for the purposes of this study and which are compatible with the microcontrollers selected in the previous section 3.2.

Since the environmental monitoring will be carried on in an indoor setting for measuring thermal comfort and air quality, the measurement range does not need to be marginal. The minimum and maximum of the sensor limits must be at the most within the acceptable human condition. If the sensors are chosen especially because they exceed this limit, the added cost and capabilities of the sensor will be unworkable. The price is a crucial factor in the selection process since the aim to be as cost efficient as possible. The implementation is aimed to be targeted to an entire facility which necessitates cost reduction and energy efficiency, which is all in line with the fundamental purpose of BIM implementation to the FMS.

All sensors taken into account must include indoor environmental monitoring capabilities that ensure air quality and thermal comfort. The response time and power consumption were relatively short and low respectively for most sensors. Therefore, they did not have a huge impact on the decision making when choosing the suitable sensor, apart from for a few exceptions that did not fit the choosing criteria in the first place.

3.3.1. Alternative sensors

The research for alternative sensors for the IoT devices was not deeply delved into, due to the fact that the microcontroller boards were selected in advance, and the sensors were specifically picked to be compatible with their respective microcontroller board and hold all the necessary features and sensing capabilities.

3.3.2. MKR ENV Shield

The 61mm x 25mm MKR ENV Shield (Figure 13) allows the Arduino MKR WiFi 1010 to capture environmental data collected by a range of sensors. These sensors are of the latest generation and measure atmospheric pressure, temperature, humidity, UVA intensity (Ultraviolet A-rays), UVB intensity (Ultraviolet B-rays), UV index and light intensity (in lux). This shield also contains a microSD slot which gives an extra option in case data needs to be stored locally. All the sensors are placed on one side of the shield which must always be facing upwards and to be placed on top of any stack of shields. This will make sure that all the measurement values of the shield that are light-related are taken correctly.

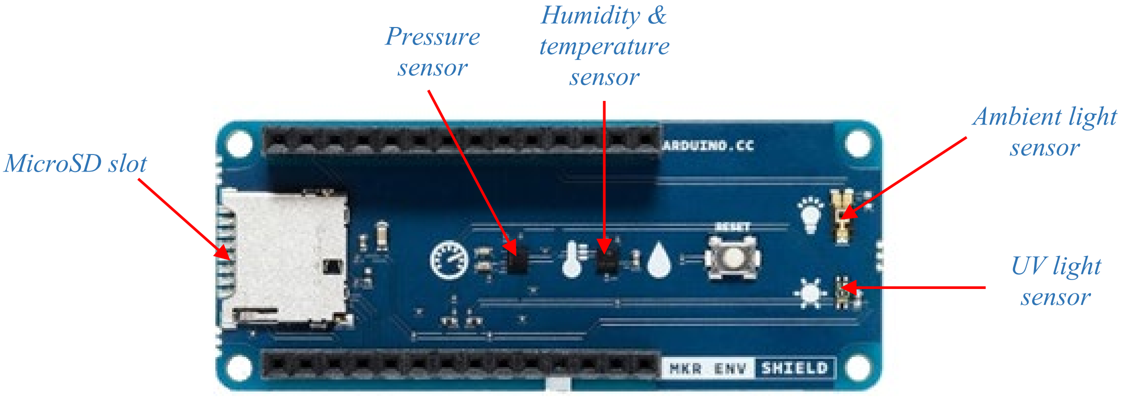


Figure 13 – Arduino MKR ENV Shield

The first sensor from the left is the LPS22HB, which is an ultra-compact piezoresistive absolute pressure sensor which functions as a digital output barometer. It operates over a temperature range extending from -40 °C to +85 °C (*MEMS nano pressure sensor: 260-1260 hPa absolute digital output barometer*, 2017). Next to the right is the HTS221 which measures relative humidity and temperature. It operates over a temperature range from -40 °C to +120 °C (*Capacitive digital sensor for relative humidity and temperature*, 2016). To the right is the reset button which allows to reset the MKR WiFi 1010 board in Figure 13 above. The top sensor on the right of the board is the TEMT6000 which is an ambient light sensor (Lux of the ambient). It is sensitive to visible light much like the human eye and has peak sensitivity at 570 nm (*Ambient Light Sensor*, 2011). Below the Lux is the VEML6075 that senses UVA and UVB light. When the UV sensor is applied, it is able to detect UVA and UVB intensity to provide a measure of the signal strength as well as allowing for UVI measurement (*UVA* *and UVB Light Sensor with I2C Interface*, 2019). The VEML6075 is also capable of producing the UV index by some calculations done by the MKRENV library. To the far left of the board is the metallic square which is the microSD slot.

3.3.3. Adafruit BME680

The BME680 (Figure 14) is a 18mm x 12mm sensor that contains temperature, humidity, pressure and VOC gas sensing capabilities. This precision sensor can measure humidity with ±3% accuracy, barometric pressure with ±1 hPa absolute accuracy, and temperature with ±1.0°C accuracy (lady ada, 2021). The sensor is also equipped with a small MOX sensor. The heated metal oxide changes resistance based on the volatile organic compounds (VOC) in the air, so it can be used to detect gasses & alcohols such as Ethanol, Alcohol and Carbon Monoxide and perform air quality measurements (lady ada, 2021). However, the VOC sensor cannot differentiate between gasses and alcohols.

Figure 14 – Adafruit BME680 sensor

3.4. Assembly and wiring

In this section, we will go through the assembly process of the Arduino boards with the sensors and the wiring required to create the IoT devices.

3.4.1. Assembly of the MKR WiFi 1010

As mentioned in section 3.2.1, the MKR WiFi 1010 is a suitable place to start with IoT devices because of how simple it is to set up a project, especially when it goes together with the MKR ENV shield. The microcontroller board and the sensor shield are designed to fit on top of each other without any additional tools, parts, or wires. The ENV shield fits on top of the MKR WiFi 1010 board, so that it will be able to properly measure values related to light as shown in Figure 15 below.

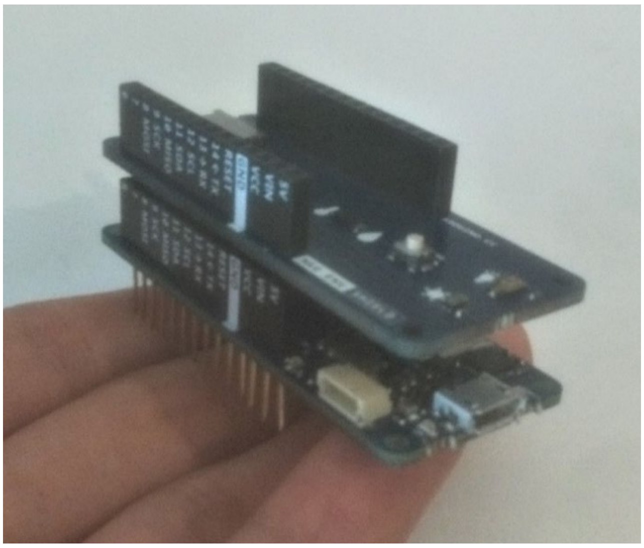


Figure 15 – Assembled MKR WiFi 1010 and MKR ENV shield

The only additional element for this project is a USB cable to connect it to a device for configuration or as a power source. In other cases, a Lipo battery charger would be required if the power source is a Lipo battery or a Li-ion battery. This depends on the power intensity needed and the price budget.

3.4.2. Assembly and wiring of the Nano 33 IoT

For the purpose of initial testing of the IoT device workability, the microcontroller is placed on an electronic breadboard (Figure 16). The final product for IoT device will be placed in a 3D printed sensor case modeled and printed during the time of the study specifically to fit the Nano 33 IoT dimensions (Figure 17).

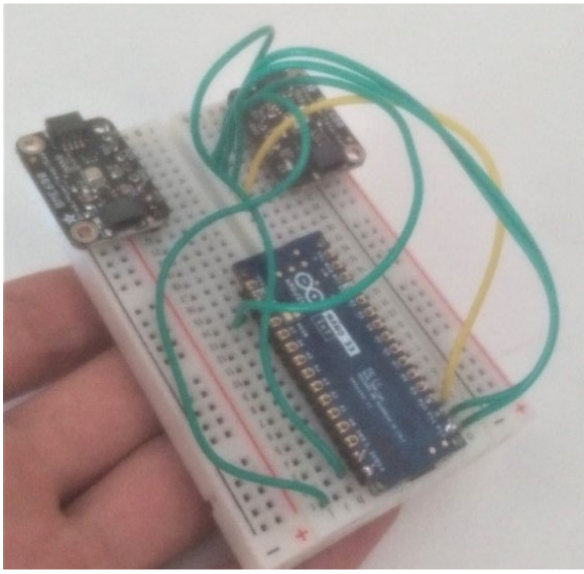


Figure 16 – Assembled and wired Nano 33 IoT with the BME680 sensor

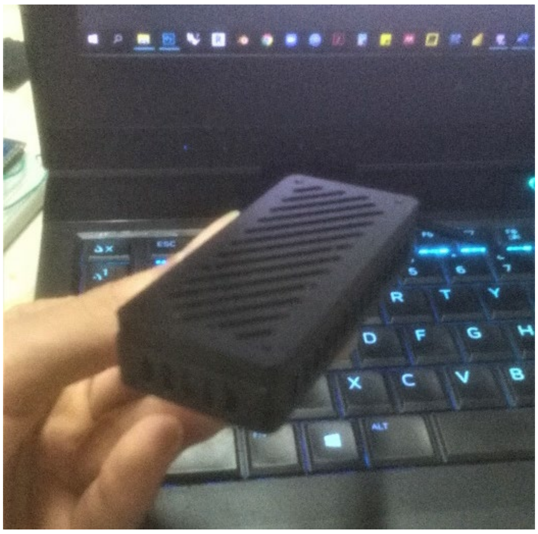


Figure 17 – 3D printed sensor case

Modelling of this sensor case took place in Autodesk Revit (Figure 18), where the design focused on keeping the case breathable and open to exterior stimulus and environmental changes. from Figure 18, the upper rectangular space is designed to fit the BME680 sensor, whereas the lower part is reserved for the Nano 33 IoT board. After the design was completed, the model was sent to be fabricated in the Couros Campus of the University of Minho in Guimarães.

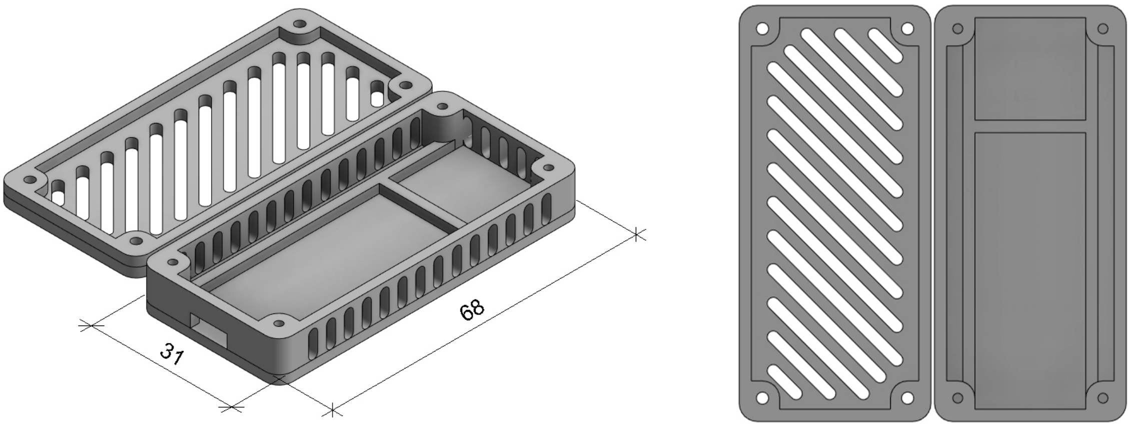


Figure 18 – 3D Revit model of sensor case

Furthermore, while the wiring and assembly of the Nano 33 IoT is a little more complicated than placing 2 boards on top of each other, it is still rather simple. The tools needed for building this project are:

- Microcontroller board (Nano 33 IoT)
- BME680 sensor
- Header strips
- Breadboard or protoboard
- A soldering iron with a small tip
- Thin solder wire
- Connecting wires

For the assembly, the first step is to cut the header strip to the necessary length making it easier to solder when inserted into the breadboard (long pins directed downward). The BME680 is then placed over the pins so that the short pins poke through the BME680 board holes. After that, all pins are soldered to ensure a reliable electrical contact. The microcontroller is also inserted into the breadboard which allows us to move on to the wiring process.

For the wiring process, the following pins are used and displayed more clearly in an organized manner in Figure 19 below.

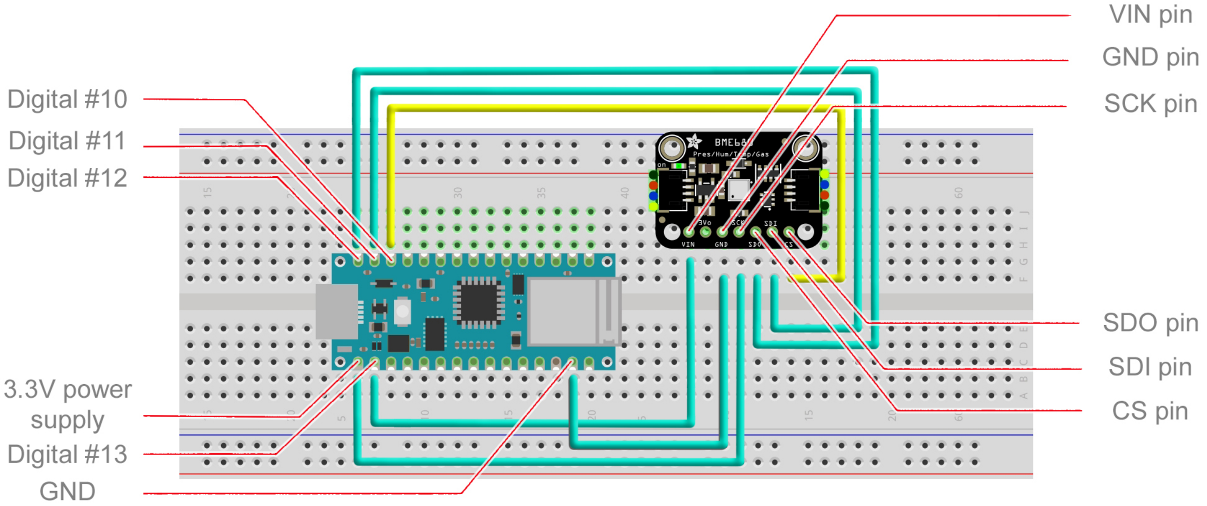


Figure 19 – Labeled Arduino Nano 33 IoT and BME680 wiring diagram

From Figure 19:

- The Vin BME680 pin (power pin is connected to the 3.3V power supply Nano 33 IoT pin
- The GND (common ground) BME680 pin is connected to the GND Nano 33 IoT pin
- The SCK BME680 pin is connected to Digital #13 on the Nano 33 IoT
- The SDO BME680 pin is connected to Digital #12 on the Nano 33 IoT
- The SDI BME680 pin is connected to Digital #11 on the Nano 33 IoT
- The CS BME680 pin is connected to Digital #10 on the Nano 33 IoT

Regarding the power supply pin, it should be mentioned that the Arduino Nano 33 IoT only supports 3.3V for input-output (IO pins) and it is not 5V tolerant like most of the other Arduino boards. Connecting more than 3.3V on IO pins will damage the board.

3.5. Programming the microcontrollers

The following step after the assembly and wiring is setting up the Arduino boards. The microcontroller is built to perform specific tasks through Arduino programming using a simplified version of C/C++ languages. This will enable the WiFi connectivity between the IoT devices and the end-user devices to visualize the collected data as graphs and diagrams. The data acquired is a collection of values of environmental aspects that ensure indoor thermal comfort and air quality such as temperature, humidity, light, VOC gases…etc. The developed system aims to obtain data from the environmental parameters in uniform intervals (every 5 seconds in this study).

Arduino Integrated Development Environment (IDE) is an open-source software that enables the communication with the Arduino hardware through uploading code which consists of programs and specific libraries for communication. From the official website download page, Arduino IDE for Windows 10 version is installed on the device, where the latest stable release is Arduino IDE 1.8.16 during the writing of the study. Once the microcontroller board is connected to the computer with a USB cable, the Arduino IDE is ready for programming.

Before starting with the coding, it is important to mention that many of the advanced features of the boards will require installing additional libraries and packages in order for the computer to recognize the microcontroller device and function according to the project requirement. The libraries that should be installed are:

- *'WiFiNINA'* which is the main library for using MKR 1010 to enable the network connection
- *'MySQL Connector Arduino'* which is required to connect to MySQL server directly without using an intermediate computer or a web/cloud-based service.

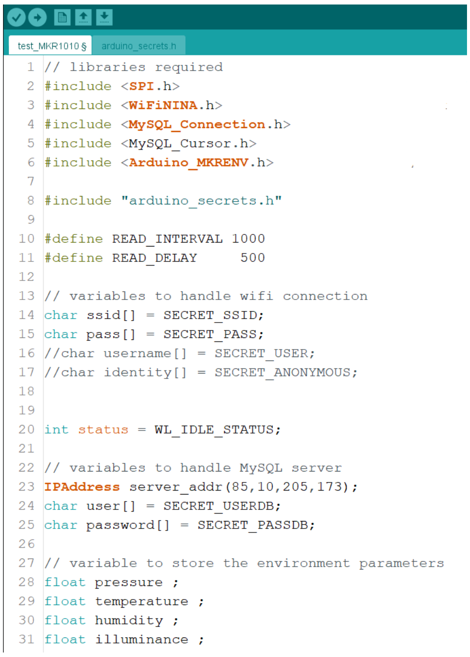
The libraries mentioned above are required for both boards (MKR WiFi 1010 and Nano 33 IoT), but for the sensors 2 different libraries must be included as well. The '*Arduino MKRENV'* that allows to read the environmental parameters provided by the MKR ENV Shield. The other library is the '*Adafruit BME680 library'* that is required to use the BME680 sensor.

Other complimentary libraries are also installed automatically that will be mentioned in the following steps. Once all the libraries are installed, the coding is now ready to start.

3.5.1. Programming of the MKR WiFi 1010

First, the required libraries previously installed must be included in the code (see Figure 20). The included libraries are *'<WiFiNINA.h>'* which enables the connection to the main MKR WiFi 1010, *'<SPI.h>'* which is required for WiFiNINA, *'<MySQL_Connection.h>'* which required to connect to MySQL, *'<MySQL_Cursor.h>'* which is automatically installed with the MySQL connector and is required to run MySQL queries, and *'<Arduino_MKRENV.h>'* which is required to use the MKR ENV Shield.

A separate file must be created to store all the sensitive data. This file will be called *'arduino_secrets'*. For the network configuration, the SSID and password are defined and stored in the separate file along with the database username and its password.



```
// libraries required
#include <SPI.h>
#include <WiFiNINA.h>
#include <MySQL_Connection.h>
#include <MySQL_Cursor.h>
#include <Arduino_MKRENV.h>

#include "arduino_secrets.h"

#define READ_INTERVAL 1000
#define READ_DELAY     500

// variables to handle wifi connection
char ssid[] = SECRET_SSID;
char pass[] = SECRET_PASS;
//char username[] = SECRET_USER;
//char identity[] = SECRET_ANONYMOUS;


int status = WL_IDLE_STATUS;

// variables to handle MySQL server
IPAddress server_addr(85,10,205,173);
char user[] = SECRET_USERDB;
char password[] = SECRET_PASSDB;

// variable to store the environment parameters
float pressure ;
float temperature ;
float humidity ;
float illuminance ;
```

Figure 20 – Arduino IDE sketch showing the first 34 lines of code for the MKR WiFi 1010 board (see Appendix 1 for full code)

For the MKR ENV shield, the variables defined for environmental parameters are: pressure, temperature, humidity, illuminance, UVA, UVB, and UV Index. These variables will define the SQL query template. The loop function responsible for continuously running in a loop showing the environmental parameters is as follows:

- void loop() {

//the loop function runs continuously until it is manually stopped.

- Serial.println("\n\nReading data from MKR ENV...");

//reads the data from the sensor

- readENV();

- printENV();

//prints the sensor data on the serial monitor of the Arduino IDE

- delay(READ_DELAY);

//small delay to make sure data was read

- Serial.println("\nSending data to MySQL server...");

//function that sends data to MySQL server

- sendData();

- delay(READ_INTERVAL);

//function to get data from MySQL server

- getData();

}

The full code with all the other functions in detail is shown in the Appendixes.

3.5.2. Programming of the Nano 33 IoT

As was done for the MKR board, the Nano 33 IoT goes through a similar procedure (see Figure 21). The libraries to be included are the same except for the ‘<Arduino_MKRENV.h>’ which would be replaced by ‘<Adafruit_Sensor.h>’ since the Nano board requires the BME680 sensor instead of the MKR ENV Shield.


```
// libraries required
#include <SPI.h>
#include <WiFiNINA.h>
#include <MySQL_Connection.h>
#include <MySQL_Cursor.h>

//parameters for BME680
#include <Adafruit_Sensor.h>
#include "Adafruit_BME680.h"
#define BME_SCK 13
#define BME_MISO 12
#define BME_MOSI 11
#define BME_CS 10
#define SEALEVELPRESSURE_HPA (1013.25)

#include "arduino_secrets.h"

#define READ_INTERVAL 1000
#define READ_DELAY     500

// variables to handle wifi connection
char ssid[] = SECRET_SSID;
char pass[] = SECRET_PASS;
//char username[] = SECRET_USER;
int status = WL_IDLE_STATUS;

// variables to handle MySQL server
IPAddress server_addr(85,10,205,173);
char user[] = SECRET_USERDB;
char password[] = SECRET_PASSDB;

// variable to store the environment parameters
float pressure ;
float temperature ;
float humidity ;
//float illuminance ;
//float uva ;
//float uvb ;
```

Figure 21 – Arduino IDE sketch showing the first 41 lines of code for the Nano 33 IoT board (see Appendix 2 for full code)

A separate file for sensitive data is created as well.

For the BME680, the variables defined for environmental parameters are: pressure, temperature, humidity, gas resistance, and altitude. The loop function responsible for continuously running in a loop showing the environmental parameters is as follows:

- void loop() {

  - Serial.println("\nReading data from BME680...");

  - readENV();

- delay(10000);

- printENV();

- delay(READ_DELAY);

//small delay to make sure data was read

- Serial.println("\nSending data to MySQL server...");

- sendData();

//function to send data to MySQL server

- delay(READ_INTERVAL);

- //getData();

}

The full code with all the other functions in detail is shown in the Appendixes.

3.5.3. Assessment of the code performance

Once the code is finished, it must be verified which helps with debugging. The microcontroller should be first connected to the computer through a USB cable. the green power LED will illuminate on the board indicating that the board is being powered properly (Figure 22). The code is then uploaded by clicking the Upload button in the toolbar. Another LED begins to rapidly blinking after proper communication has been established between the computer and the boards. The serial monitor receives data from the Arduino and sends them to the computer, and as a result displays the resulting environmental values.

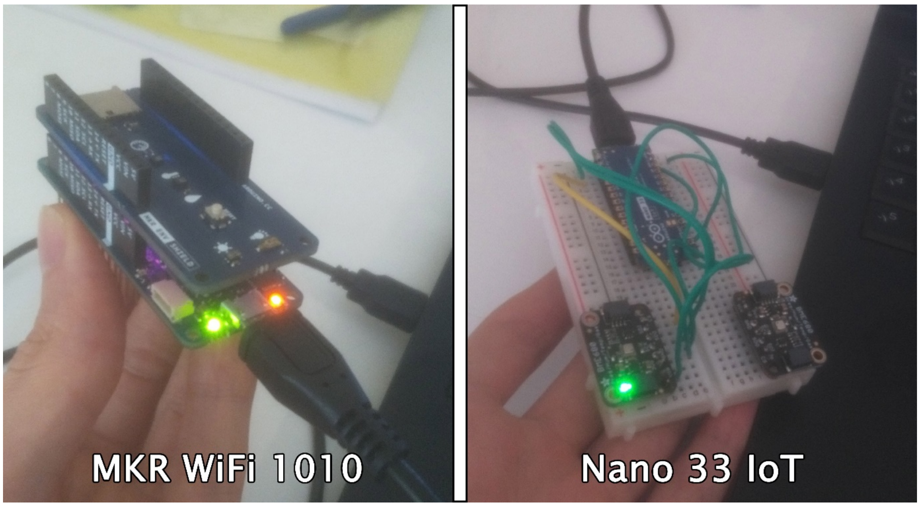


Figure 22 – Both IoT devices connected to the computer and working properly after upload

After that, the collected data populates the dashboard report for environmental monitoring. The visualization platform continuously displays the latest data results by refreshing the report. This is further discussed in Chapter 4.

This page is intentionally left blank

4. DATABASES, DASHBOARD SETUP, AND GLOBAL WORKFLOW

In a BIM-based EMS, data is classified into two categories: monitoring data and BIM data (Kang, Lin and Zhang, 2018). The two have different requirements on storage system. This chapter discusses the process of the integration of a MySQL database which is critical to the development of the BIM-FM workflow and the case study. A relational model of data is established to maintain a straightforward link between tables, minimize redundancy, and retain the consistency of relations. Furthermore, the performance differences between local and online database servers are explained and a decision is made based on the presented results and conclusions.

4.1. Relational database properties

This section considers the most proficient and adaptable SQL database design for the purposes of the research, as well as the approach to successfully implement database in a project for FM duties in the O&M phase of the project lifecycle.

This study features two types of data: monitoring data and BIM data. The monitoring data are the continuous data collected by the WSN. Sensors in the building collect environmental parameters, transform this data into readable values for the facility manager, and upload it to an online server. Each record contains specific identification for the sensors such as ID, type, location, and several other monitoring information. This data type has a simple data structure, since each recording is recognized by its ID and has its own timestamp and monitoring results. The data query of the monitoring result is not complicated as well (Kang, Lin and Zhang, 2018). It is previously handled in the programming phase of the framework. The facility manager only needs to update the results when it's necessary to display the latest environmental values.

The BIM model data consists of a variety of information of different components in the BIM model. The BIM model contains: (1) the information of structural components such as walls, floors, windows and doors; (2) the information of many kinds of facilities such as pipes, cables, and airducts; (3) and also has the information of other types of data such as spaces, rooms, and levels. BIM model data also features complex structures. There are lots of correlations between the elements. For example, the wall should be linked with the floor and the four sides of the room will be closely related to the location and shape of the wall (Kang *et al.*, 2018).

There are different ways to begin designing a database model for a project. An Entity-Relationship Diagram (ERD) is a model that shows relationships of entity sets (people, places, objects…etc.) stored in a database. ERDs use different symbols and shapes to distinguish between model roles, where rectangles represent entities (blue), ovals represent attributes (white), and rhombuses represent relationships (yellow). Figure 23 shows the ERD model designed for the purpose of this study. The implementation is aimed at university facility buildings and other commercial buildings, where the floor plans are typically designed with spaces of modular sizes. Multiple factors are taken into consideration when designing an ERD that is later translated into the database structure. The building spaces (rooms) are distinguished by their level and function. Each room consists of its own designated shield that has a unique ID to measure and record environmental parameters. This shield has identifying information as well as other alphanumerical properties (name, thresholds, range…etc.) that must be stored for later use. In addition, each shield has multiple sensors that measure different environmental parameters. This is where the sensor types need to classified. These sensors record reading values and alerts.

If the number ‘1’ and letter ‘n’ are on both sides of the yellow rhombus, this represents a one-to-many relationship between entities. In this ERD this is the only type of relationship. However, other relationships may occur such as one-to-one (‘1’ and ‘1’) or many-to-many (‘n’ and ‘n’) relationships.

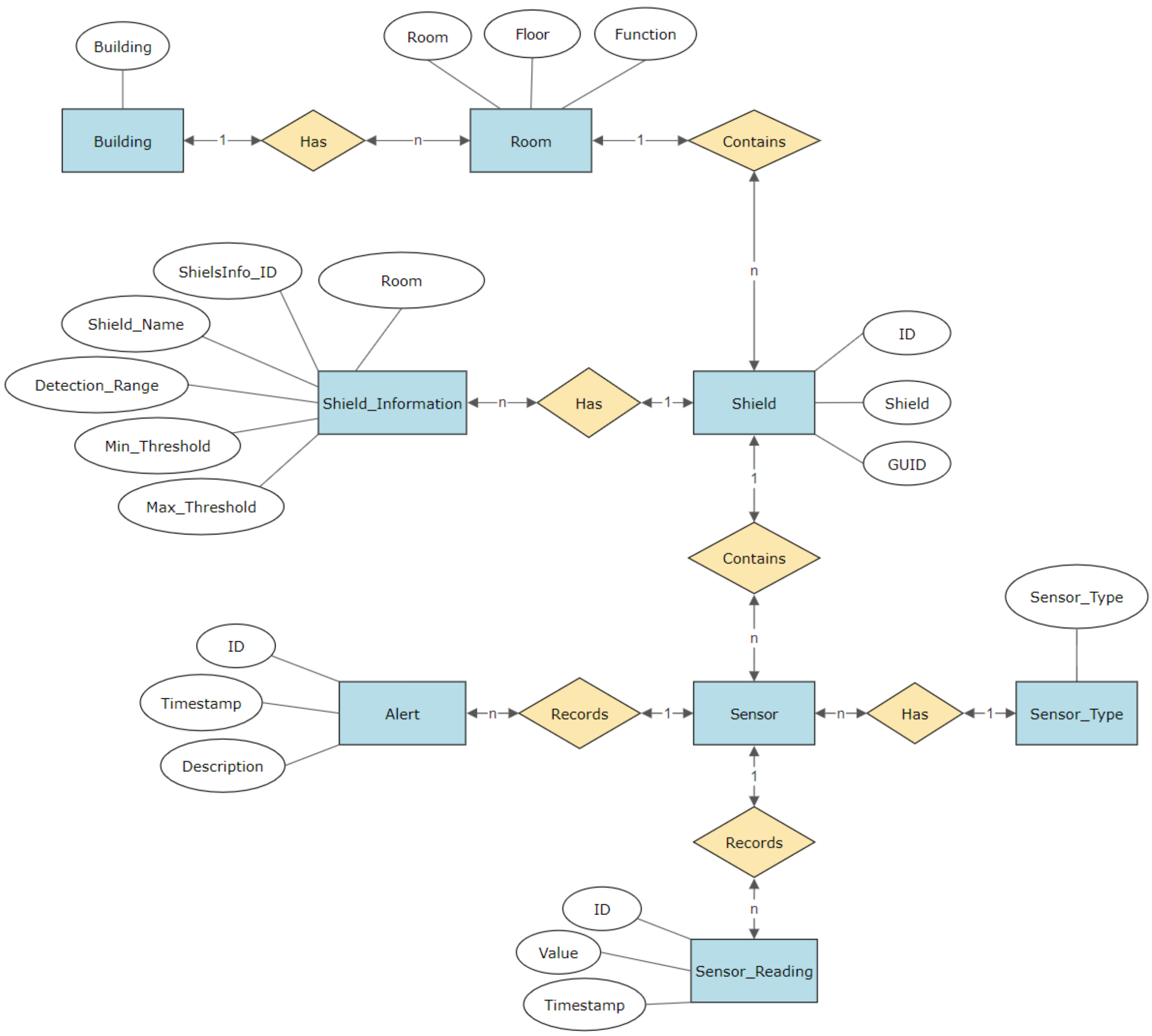


Figure 23 – Primary ERD for an indoor environmental sensor monitoring system

The primary ERD is then developed into a more workable diagram that serves the monitoring system better, with the help of a database system software. MySQL Workbench software is a visual tool for developing databases design, as well as executing, optimizing, and administrating SQL queries in a single environment that can store data and can be connected securely to, either locally or remotely.

The data required to design the database is input into Workbench, with the appropriate datatypes and keys identified. The result is shown in Figure 24 below.

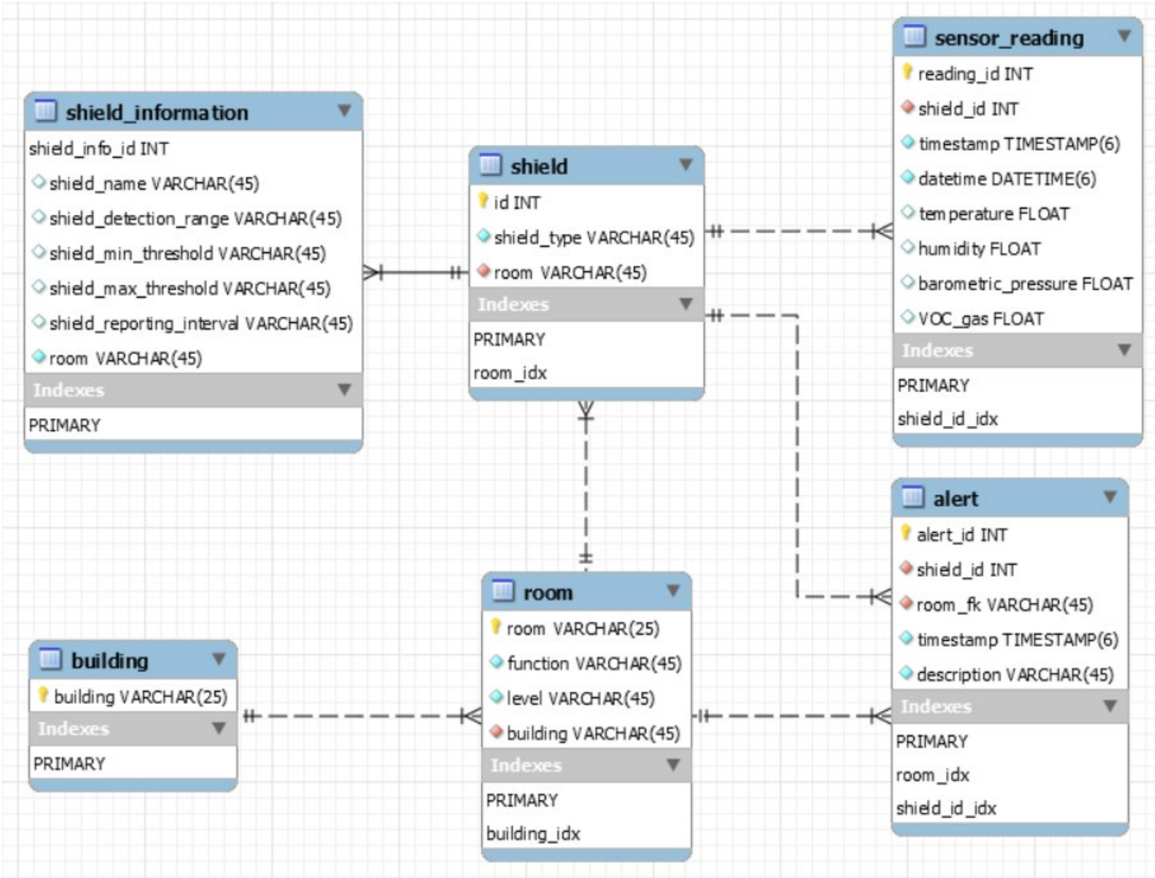


Figure 24 – ERD design for the sensor monitoring system created in MySQL Workbench

In the latter design displayed in Figure 24, the data model is more systematized and illustrated according to relational database design standards. Each rectangle from Figure 23 is translated to a table title in Figure 24. The ovals branching from the rectangles represent table attributes. Each table represents one entity that has different attributes with specific column names. Each 'Column Name' has an associated 'Datatype' that determines what kind of data is expected to be populated in that cell. Primary Keys (PK) and Foreign Keys (FK) are chosen to the necessary attributes to signify the main identifying attribute (relational identifier) for the entity (PK) or a linked entity (FK). Such databases that are easy to change and extend, as well as easy to use, by users and future applications. The result is a genuine Relational database, which allows expansion, rather than replacement due to the limitations otherwise (Asirvadem, 2013).

In an attempt to better understand the ERD, Table 5 simplifies the database layout and structures the data in a way that presents only what is required to be viewed by the end user or the facility manager. Entities are plainly presented on the 'Tables' column, with either single or multiple attributes shown next to them in the 'Datatypes' column, each attribute separated by a 'comma'.

Table 5 – Database design with the entities (Tables) and their attributes (Datatypes)

Tables	Datatypes
building	'building'
room	'room', 'function', 'level', 'building'
shield	'id', 'shield_type', 'room'
shield_information	'shield_info_id', 'shield_name', 'shield_detection range', 'shield_min_threshold', 'shield_max_threshold', 'shield_reporting_interval', 'room'
sensor_reading	'reading_id', 'shield_id', 'timestamp', 'temperature', 'humidity', 'barometric_pressure', 'VOC_gas'
alert	'alert_id', 'shield_id', 'room_fk', 'timestamp', 'description'

This database design manages the data that comes from the 3D model and the sensor network and stores it for use by the facility manager.

4.2. Local and online servers

This section explains the procedure for implementing a local or online database to the environmental monitoring system. The IoT devices collect and store data from the indoor environment through the WSN, which in turn sends this data to a database for FM purposes.

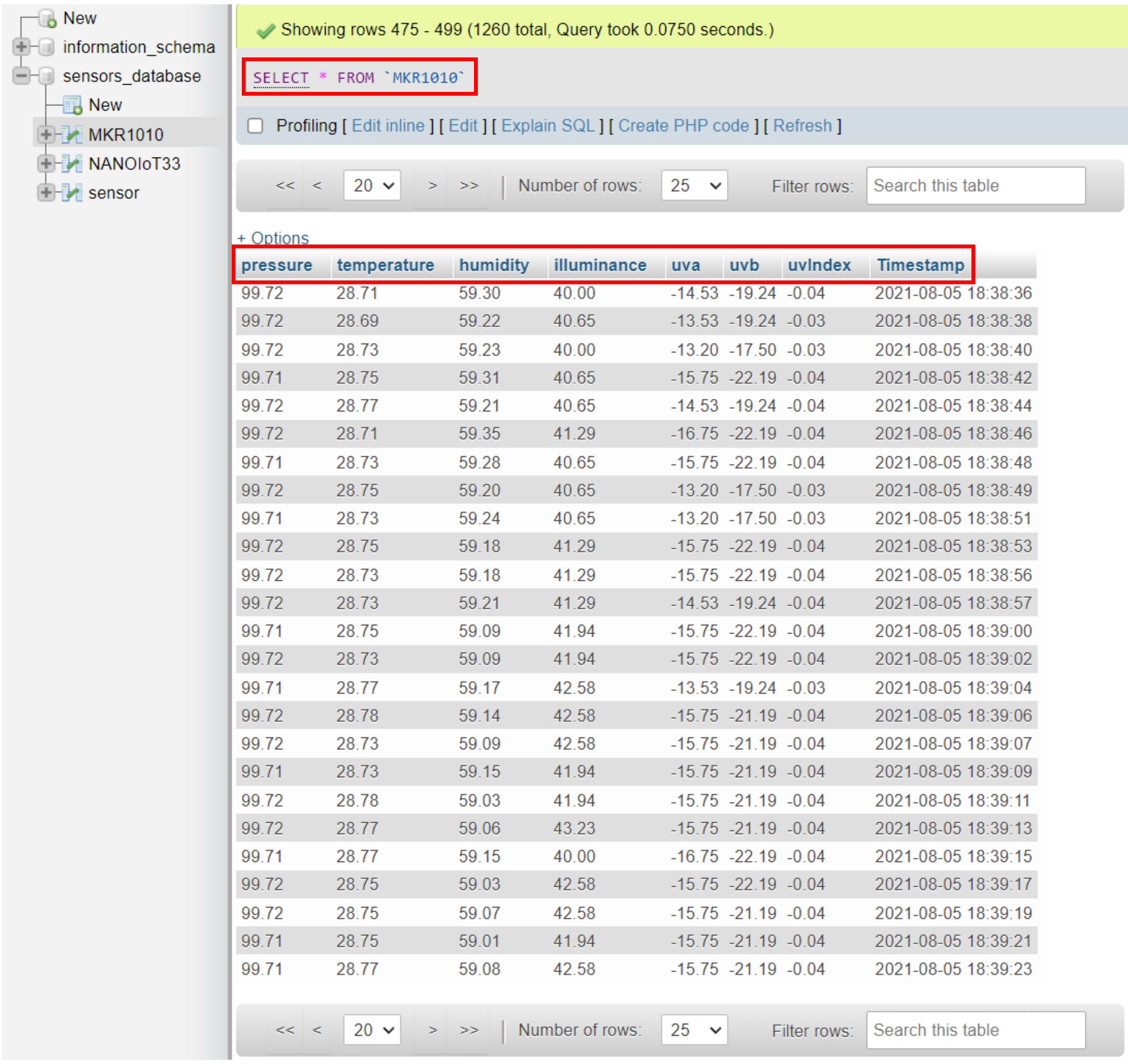
4.2.1. Local server

When planning the project workflow, the initial objective was to store all the data from the 3D model and the sensor network on a local database server, since the only party involved that needed direct access for the database was the facility manager. Therefore, it was plausible to store all the data locally without having to create an Online Database Service (ODS) accessible by another party. As the project progressed, introducing an online server when managing the database connection through MySQL Workbench, the software didn't allow connect to the internet in order to access the Arduino devices and the localhost server at the same time. The Workbench failed to differentiate between the IP address of the local network and the WiFi network responsible for the WSN. This procedure manages to eliminate the confusion from the software and both the WSN and the 3D model are successfully connected.

4.2.2. Online server

When considering to use a certain ODS, there were several options to take into account. That said, one of the leading services for this kind of task is phpMyAdmin. This is a free and open-source administration tool for MySQL that gives the ability to interact with the database. In addition, it is a portable web application that can be accessed from any place with a network connection and the necessary login credentials.

One of the services that provide an up-to-date version of phpMyAdmin is he website db4free.net. This is a free testing service that performed adequately for the purpose of this study, but might not be suitable for production and practical implementation, and may not meet the standards of a professional data hosting provider. This is due to possible outages, data loss, data constraints limitations, data transfer constraints, and potential security problems that are not expected to be found in a professional service infrastructure for storing and managing access to data. Figure 25 below shows the results from the MKR1010 sensor for all changing environmental parameters with the timestamp being recorded approximately every 2 seconds.



pressure	temperature	humidity	illuminance	uva	uvb	uvIndex	Timestamp
99.72	28.71	59.30	40.00	-14.53	-19.24	-0.04	2021-08-05 18:38:36
99.72	28.69	59.22	40.65	-13.53	-19.24	-0.03	2021-08-05 18:38:38
99.72	28.73	59.23	40.00	-13.20	-17.50	-0.03	2021-08-05 18:38:40
99.71	28.75	59.31	40.65	-15.75	-22.19	-0.04	2021-08-05 18:38:42
99.72	28.77	59.21	40.65	-14.53	-19.24	-0.04	2021-08-05 18:38:44
99.72	28.71	59.35	41.29	-16.75	-22.19	-0.04	2021-08-05 18:38:46
99.71	28.73	59.28	40.65	-15.75	-22.19	-0.04	2021-08-05 18:38:48
99.72	28.75	59.20	40.65	-13.20	-17.50	-0.03	2021-08-05 18:38:49
99.71	28.73	59.24	40.65	-13.20	-17.50	-0.03	2021-08-05 18:38:51
99.72	28.75	59.18	41.29	-15.75	-22.19	-0.04	2021-08-05 18:38:53
99.72	28.73	59.18	41.29	-15.75	-22.19	-0.04	2021-08-05 18:38:56
99.72	28.73	59.21	41.29	-14.53	-19.24	-0.04	2021-08-05 18:38:57
99.71	28.75	59.09	41.94	-15.75	-22.19	-0.04	2021-08-05 18:39:00
99.72	28.73	59.09	41.94	-15.75	-22.19	-0.04	2021-08-05 18:39:02
99.71	28.77	59.17	42.58	-13.53	-19.24	-0.03	2021-08-05 18:39:04
99.72	28.78	59.14	42.58	-15.75	-21.19	-0.04	2021-08-05 18:39:06
99.72	28.73	59.09	42.58	-15.75	-21.19	-0.04	2021-08-05 18:39:07
99.71	28.73	59.15	41.94	-15.75	-21.19	-0.04	2021-08-05 18:39:09
99.72	28.78	59.03	41.94	-15.75	-21.19	-0.04	2021-08-05 18:39:11
99.72	28.77	59.06	43.23	-15.75	-21.19	-0.04	2021-08-05 18:39:13
99.71	28.77	59.15	40.00	-16.75	-22.19	-0.04	2021-08-05 18:39:15
99.72	28.75	59.03	42.58	-15.75	-22.19	-0.04	2021-08-05 18:39:17
99.72	28.75	59.07	42.58	-15.75	-21.19	-0.04	2021-08-05 18:39:19
99.71	28.75	59.01	41.94	-15.75	-21.19	-0.04	2021-08-05 18:39:21
99.71	28.77	59.08	42.58	-15.75	-21.19	-0.04	2021-08-05 18:39:23

Figure 25 – Query result for collected sensor data for pressure, temperature, humidity, illuminance, UVa, UVb, and UVindex from the db4free.net ODS

In Figure 25, the online database for the sensor readings shows the pressure, temperature, humidity, illuminance, UVA, UVB, UV index, and timestamp. This is the only data that requires to be stored remotely. The remaining data is necessary for the facility manager and is stored in a local database server. The data in Figure 26 follows the method to specify the LOIN in accordance with EN 17412-1:2020 standards (EN 17412-1, 2020). It contains the geometric information, alphanumeric information, and documentation for the object. The different proprietary information about the sensors taken from the product datasheets are shown in Figure 27. This list of information was carefully selected for FM purposes and reviewed by the facility manager overseeing the development of the workflow.

Purpose:	**Facility Management**
Actor	*Facility Manager*
• Object:	Sensors
• Geometric Information:	
• Detail	Simplified volume representation
• Dimensionality	3D
• Location	Relative
• Appearance	Not requested
• Parametric Behavior	Not requested
• Alphanumeric Information:	
• Element ID	(Ex: 357106)
• Asset Type	Movable
• Volume	Not requested
• Material	Not requested
• Uniclass2015Code	Pr_75_50_76_02
• Documentation:	
• Sensor Datasheet Webpage	https://web.archive.org/web/20190228002601/https://cdn-learn.adafruit.com/downloads/pdf/adafruit-bme680-humidity-temperature-barometic-pressure-voc-gas.pdf
• Board Information Webpage	https://web.archive.org/web/20210616144832/https://static.rapidonline.com/pdf/73-4863_v1.pdf

Figure 26 – Example for 3D model object parameters LOIN as described in EN 17412-1:2020 required for FM purposes (see Appendix 3 for the editable table)

Temperature	
Application	(Ex: Frost protection)
Range	-10°C to 40°C
Accuracy	±0.5°C
Relative Humidity	
Sensor type	(Ex: Capacitive sensor)
Application	Relative humidity (RH)
Range	10% to 90% RH
Accuracy	±5% RH
Barometric Pressure	
Sensor type	(Ex: Capacitive)
Application	Atmospheric pressure
Accuracy	±2% measured value
Air Quality (VOC)	
Sensor type	VOC sensor

Figure 27 – Example for sensor data from the product datasheets required for FM purposes

4.3. Dashboard layout design

This section goes into several visualization platforms available for the management of FMS, as well as the platform selected for this study and its dashboard layout proposal.

Facility management includes multidisciplinary roles that overlook the building maintenance process. One of these roles is data visualization that can provide the end users with clear information displayed on an interactive dashboard. This dashboard must be designed to up to the facility manager's standards and suitable for their uses. Data visualization can be achieved through a WSN data collection system that stores information in a database and imports it into the visualization platform. This in turn allows to analyze, visualize, and interact with this displayed data accurately and in real-time to aid in the O&M phase of the project lifecycle.

4.3.1. Layout proposal description

The layout proposal for the dashboard is presented in Figure 28 below. The design was completed through multiple gadgets originally available in Power BI, the gadgets found in the Tracer toolkit that are responsible for holding the BIM model data, and other gadgets individually installed for broader and enhanced visualization preferences. The data obtained and shown in Figure 28 are the results of data accumulation of one room in an apartment building for experimentation and analysis purposes. On the left of the dashboard are the 'Cards' that display the average environmental parameters separately. The information on the cards is the data collected by the IoT devices and is obtained from the ODS. When these cards are clicked, the location of the displayed data is shown, as well as the time frame through which the average parameter is calculated.

On the top center is the 'Slicer' that is responsible for toggling between different levels of the building when displaying environmental parameters. The results shown on the cards are affected by the change in the level. This ensures the dynamic nature of the FM dashboard for faster responses and a more intelligent workflow.

The 'Dial Gauge' that displays the temperature parameter is positioned at the center of the dashboard. This gadget has an important role on the dashboard since temperature can be considered as the most influential environmental parameter according to the WHO (1990). The Dial Gauge allows to define various ranges in the dial along with the pointer value. Different colors indicate the change in the temperature level, as well as the comfort level of the inhabitants. A minimum and a maximum are defined by the facility manager (e.g., 0 °C to 31 °C) and the pointer shows the current temperature value. When the gadget is clicked, parameters such as the date, the location, and the temperature level range is displayed (e.g., temperature is between 18 °C and 23 °C).

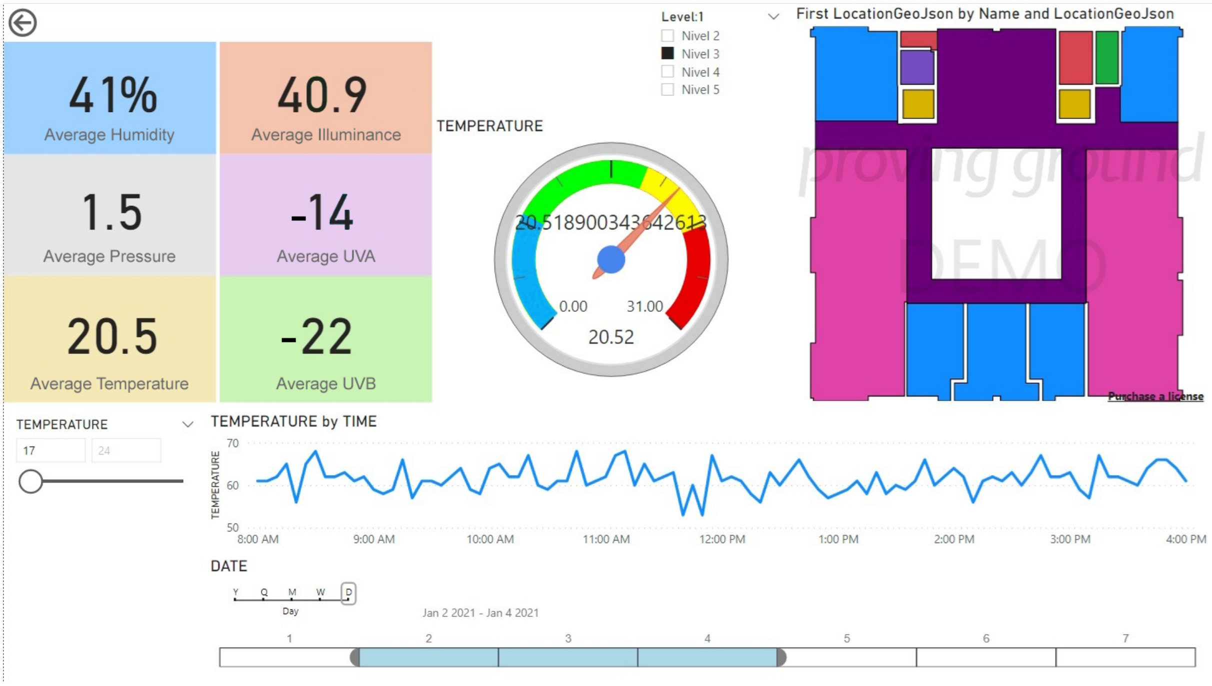


Figure 28 – Power BI dashboard layout proposal for FM purposes

Under the Dial Gauge is a linear chart that displays the change in temperature (or any other environmental parameter) with respect to time. This helps the facility manager observe and analyze the variations that are necessary for decision making keep a regulated and comfortable IEQ. The gadget under the linear chart is a timeline that customizes the time frame through which the linear chart or the average environmental parameter is to be displayed. This gadget has the option to toggle between different time units such as years, quarters, months, weeks, and days. This gives the opportunity to make decisions regarding the future when enough environmental data is collected by the WSN.

To the left of the linear chart is a second slicer that acts as a temperature filter. This means that when the facility manager needs to recognize the behavior of environmental parameters' variation under a certain temperature value, or between two given temperature values, they only have to define a range or a limit to obtain the results required.

At the top right of the dashboard is the gadget from Tracer that displays the 2D plan extracted from the Revit model. This gadget demonstrates various valuable information for the facility manager that the integrated Power BI capabilities cannot normally achieve. The 2D plan view is linked directly to the Revit BIM model and dynamically changes in shape or parameters in case there are modifications in the 3D model.

The gadget contains multiple fields that can be filled with information from the BIM model that is imported through the Revit toolkit. The imported data can include information such as elements, rooms, spaces, levels, sensors, as well as environmental comfort boundaries. In the same way as the standard Power BI gadgets, the Tracer plan view is able to show different assigned parameters such as the name of space, location, temperature, and others to be displayed when the desktop pointer hovers over the gadget, in order for the facility manager to get the information needed instantaneously.

A 3D model view is also available for the Power BI dashboard. However, it was not added to the dashboard layout since it was not essential for the workflow in this study and served only as a complementing function. The research deals with rooms and spaces, so a 2D view is sufficient to show all the necessary elements that are in use. When the FM responsibilities include tasks that require going more into smaller details such as the MEP or structural systems of a BIM project, a 3D model view for the dashboard may perhaps be a necessity.

4.4. Implementation and workflow

This section proposes a methodology for BIM-FM-IoT processes implementation in a real-world setting. The methodology process goes through five layers of the workflow that starts with the model layer (M), through the sensing layer (S), the data exchange layer (E), the service layer (D), and ends with the interface layer (V). A workflow process diagram is included in Figure 29 for further demonstration.

1) The process starts with the Model Layer from M1 to M3, where the 3D BIM model must be created for the building. If a 3D model of the building already exists, then it must be checked for as-built conditions. If the model requires any elements to be modified, updated, or added, then it must be performed by the BIM specialist in accordance with the project requirements. After the elements are added, the final model quality checking is performed to confirm its usability for application.

2) The process then moves forward to the Data Exchange Layer. If the quality check shows a negative result, then the additional model elements must be corrected or remade. If the result shows to be positive, and no issues were found, then the process moves forward to the Data Exchange layer.

   After setting up a local connection using MySQL Workbench tool for databases, BIM model parameters are selected according to the FM needs and then exported to be stored on a relational database. The model parameters are exported using the Tracer Revit exporter which is a utility that builds an SQL (*.db) file from the open Revit document. The tool constructs a file containing data from the Revit document, elements, and parameters in an SQL relational structure (ProvingGround, 2021). The exported data that contains geometric, alphanumeric, and documentation information about the model elements, as well as the sensors' required parameters is stored in the database until it is transferred to the visualization platform to be used in the Interface Layer of the workflow.

3) In the Sensing Layer of the workflow process, the first step required is to build IoT devices from environmental sensors such as temperature, humidity, pressure, and light intensity sensors. Since all these parameters must be measured and monitored simultaneously, an environmental shield containing multiple sensors combined in a single board. Those shields are assembled with Arduino hardware boards that are responsible for carrying the Arduino IDE code for data acquisition to create IoT devices ready for installment in an indoor space environment. The next step is establishing a network connection with these IoT devices by running the previously created Arduino IDE sketch.

4) The Service Layer of the workflow starts by setting up an ODS such as phpMyAdmin which is an open-source tool for the administration of MySQL databases over the web. The necessary tables are created for storing the environmental data on the ODS, while giving the tables' columns appropriate names indicating the environmental parameters. From the IoT devices to the network, the environmental data accumulated is stored on the ODS to be used later on for visualization.

   If the environmental data is being collected accurately by the sensors and this data is being correctly stored in the ODS relational database, the workflow moves forward to the Interface Layer that is the final layer for the process. Otherwise, if there is an issue with either the data collecting or storing, process rewinds back to the Sensing Layer where the Arduino code is reviewed and modified accordingly by the specialized programmer, and ran again for assessment to verify that it's working properly. This activity can be performed multiple times before a satisfactory result is reached.

5) Following the Service Layer is the Interface Layer that deals with the platform and tools for visualization and data analysis. To begin with, Power BI must be installed on the device along with the Tracer toolkit. With the available Power BI visuals and the additional gadgets from the Tracer toolkit that gives the capability of integrating 2D and 3D representations of the BIM model on the visualization platform, a dashboard report layout is created for environmental data visualization purposes. This activity is completed by the collaboration between the BIM specialist and the facility manager who is consulted to design a dashboard according to their needs for optimum efficiency.

After the dashboard layout is finalized, the BIM specialist shares a web portal link of the dashboard report with the facility manager, which allows the latter to manage environmental data from a remote location using any device as long as they have the link. It also allows them to modify the default dashboard if needed to better suit their requirements.

The dashboard report is available for the facility manager only when the BIM data previously stored on a MySQL Workbench relational database and the environmental data stored on the phpMyAdmin ODS are imported into Power BI to populate the dashboard. The facility manager receives BIM data and environmental data automatically, and to receive the latest data they only have to click the refresh button on the report in order to keep the information updated from the linked data files. The facility manager and O&M team is now prepared for analysis of indoor environmental parameters and decision making to improve FM related activities for more efficient asset management.

The above approach involves a simple cost-efficient BIM, FM, and IoT integration in a single workflow process that can be implemented in a facility building.

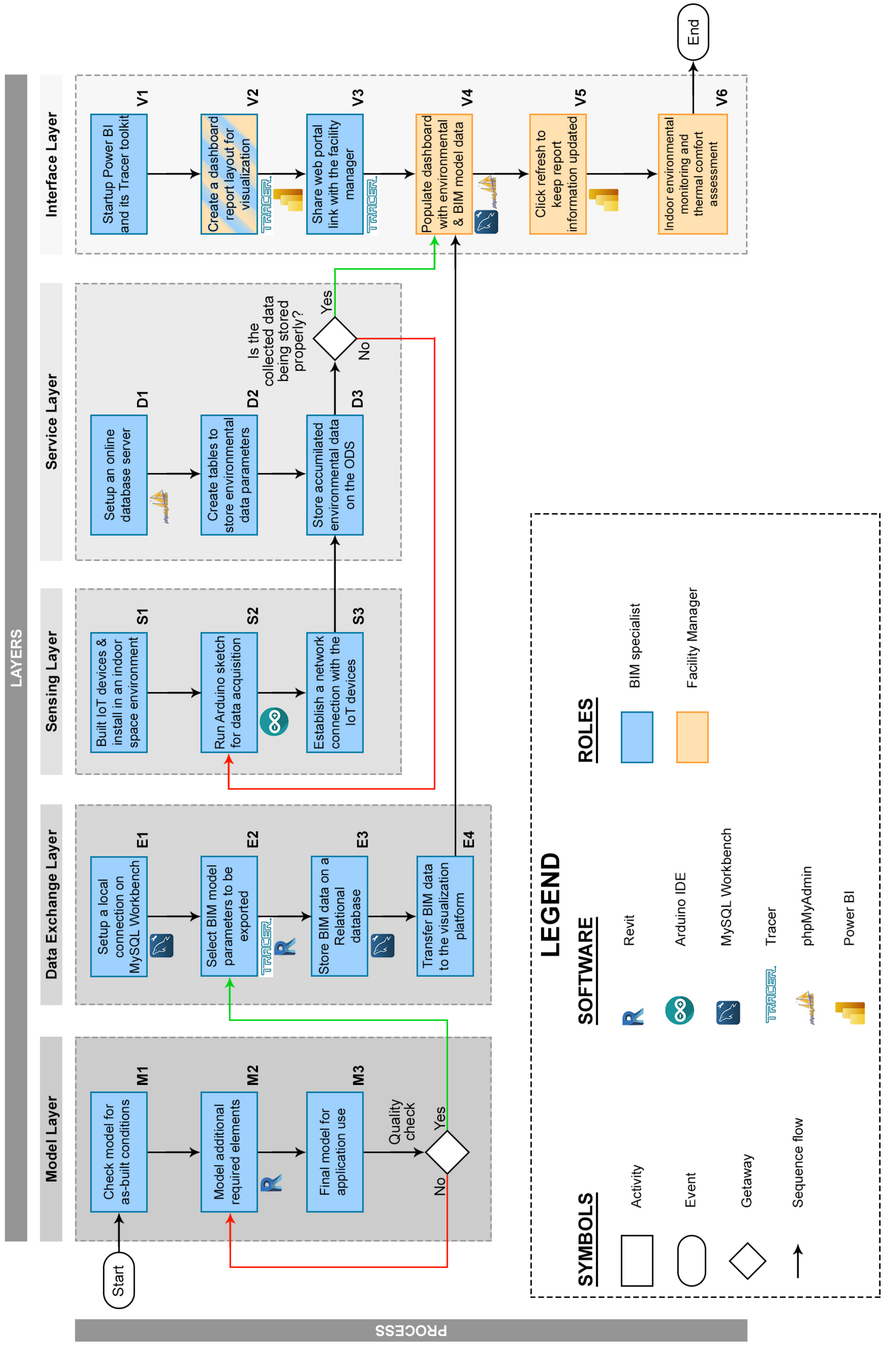


Figure 29 – BIM-FM-IoT workflow process for project implementation

This page is intentionally left blank

5. CASE STUDY APPLICATION: LIBRARY

The library building considered for the implementation of the case study is located in the Campus of Azurém of the University of Minho, Guimarães (building #13) (Figure 30). The aim is to propose a framework of a smart facility management (FM) system for the library building and develop an approach that can be used as a reference to the implementation of FM ecosystem to other academic facilities. One of the reasons this building was chosen is the fact that the building already has an IoT architecture consisting of a motion sensor system and protocols installed which control the lighting fixtures according to the space occupancy. This facilitates the process of implementation of a FM monitoring system for the building to ensure thermal comfort of occupants and indoor air quality.

For the purpose of application, a key factor is the use of a wireless sensor network (WSN) that enables an environmental monitoring system across a large facility to be established. A system of this kind would significantly help reducing energy waste and controlling energy consumption of the buildings and monitoring the comfort and safety of the users. Other applications may include increasing the lifecycle of the buildings by supervising the development of constructive pathologies and assisting the maintenance of the building infrastructures.

In this sense, the physical building develops aspects of a smart facility building with the help of an alternate 3D BIM model connected to it through sensors, microcontrollers, and different IoT devices to provide valuable input to the facility manager. The facility manager in turn monitors the changes and supervises the monitoring system in case of any notifications or warnings and ready to take the proper regulatory actions.

Figure 30 – Map of Azurém campus with a close-up of the library building #13 (campi.uminho.pt, 2021)

5.1. Description of the building

Located in the Azurém Campus (Figure 31) of the University of Minho, the General Library was designed by the architects Alexandre Alves Costa and Sérgio Fernandes of the architectural studio Atelier 15. It accommodates about 400 study spaces, 60,000 books, multimedia areas, and other services. It is made up of seven floors, with a gross area exceeding three thousand square meters. It is a multifunctional space that combines traditional library services in addition to other innovative services and resources, as well as spaces available for individual or group study, courses and classes, small events and recreational spaces.

Figure 31 – UMinho library building in Guimarães (Mais Guimarães, 2018)

The seven floors of gross area 2800 m^2 functions are as follows:

- Ground floor which is the publications deposit.
- First floor that has two 24x7 meters study rooms split into four smaller rooms by a glass partition.
- Second floor that includes the building's main entrance, services, a lounge area and a multifunctional space for events or for training.
- The third floor through the fifth floor consist of study rooms, group study/work cabinets, and a common living area (Figure 32).
- Sixth floor which includes the management and technical rooms, individual study areas, a study room, and a multimedia room.

These functions varied between study rooms, group study/work, and offices. On top of that, the library provides more than 120 seats in lounge areas and multifunctional areas.

Figure 32– 3rd floor of UMinho library building with labeled functions

5.2. The library BIM model

The BIM model of the seven-story building was developed in Revit software by a group of students from the same University Masters course in Civil Engineering (Trabulci, 2020). The BIM files relevant to this study are the architectural and structural Revit files.

The initial BIM model contained basic geometrical information such as dimensions, constraints, appearance, and floor level. After developing the model, additional geometric and alphanumeric parameters were included to the architectural model for the visualization platform to identify separate spaces and rooms and their locations. Virtual environmental sensors were embedded in the model as well. A set of parameters were defined during the modeling process:

- For the MKR ENV Shield virtual sensor: *'Sensor_ID'*, *'Pressure'*, *'Temperature'*, *'Humidity'*, *'Illuminance'*, *'UVA'*, *'UVB'*, *'UV Index'*, and *'Timestamp'*.
- For the BME680 virtual sensor: *'Sensor_ID'*, *'Pressure'*, *'Temperature'*, *'Humidity'*, *'Gas Resistance'*, *'Altitude'*, and *'Timestamp'*.
- For the Rooms in the BIM model: *'Room_ID'* and *'Thermal Comfort Level'*.

The environmental parameters for the sensors were created to measure the values of the latest time recorded or during a specified interval of time. The *'Timestamp'* parameter was created to store the running environmental measurement's date and time each 5 seconds. The *'Sensor_ID'* parameter was used to link the physical sensors to the virtual sensors in the BIM model. The virtual sensor in the BIM model must be assigned an element unique identifier which corresponds to the physical sensors (Figure 33). The *'Thermal Comfort Level'* parameter was created to monitor the thermal conditions of a room in real-time.

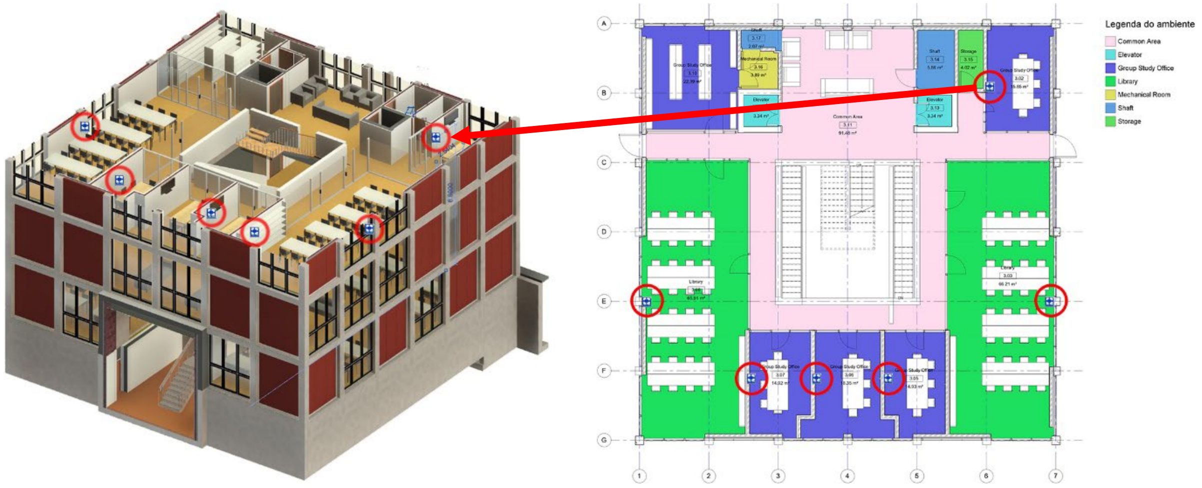


Figure 33 – 3rd floor isometric view and plan view of BIM model showing sensor locations

In the initial stages of the study, the goal was to connect the BIM model to the online database server which contains all the essential parameters about the model and the environmental data related to the model that is being updated in real-time.

5.3. Modelling of the IoT devices

To support the case study development, virtual environmental sensors were modeled in Revit and embedded into the 3D BIM model of the library facility building. These virtual sensors have the parameters of a real sensor and simulate its functions (Figure 35). They contain:

1) geometric information such as the dimensions, location, and geometric details (simplified or detailed);

Since the geometrical details of the sensors were not relevant to the study, only a simplified representation was required. The location of the sensor was aimed to be relative to the room it is located in. Other information such as appearance and the parametric behavior was not requested.

2) alphanumeric information such as the element ID, asset type, classification code…etc.;

Only the essential information of the sensors was needed such as identification information to locate the sensor in later stages, classification code as a universal reference or standardized system for the object's nature and function, and the asset type to determine the quantity needed for carrying out the study assessment of the workflow.

3) documentation such as the sensor datasheet weblink and board information weblink.

This contains the full product datasheets of the sensors and electronic boards which can be referred to during the O&M stage of the project.

All this information is stored inside the sensor element in Revit model as well as on an external relational database in order for the facility manager to retrieve the information easily when needed.

In this case study, two types of sensors were modelled for use which correspond to the MKR ENV Shield and the BME680 sensor. These two types then populate the BIM model and are positioned in the same place where the IoT devices are located in the physical building. To model the virtual sensors on Revit, a number of steps need to be followed to ensure that the modelling process abides by BIM standards, whether local or international. These steps include (1) searching for similar family types on an online BIM library; (2) checking parameters and information from the manufacturing company through attached datasheets; (3) placing elements in the BIM model that correspond to their location on the physical model; (4) adding or modifying missing or incorrect parameter data in the Revit element according to the LOIN standards; (5) assessing the completion of the required information for FM purposes. Figure 34 shows some of the similar family types of sensors that were downloaded from the BIMobject website library.

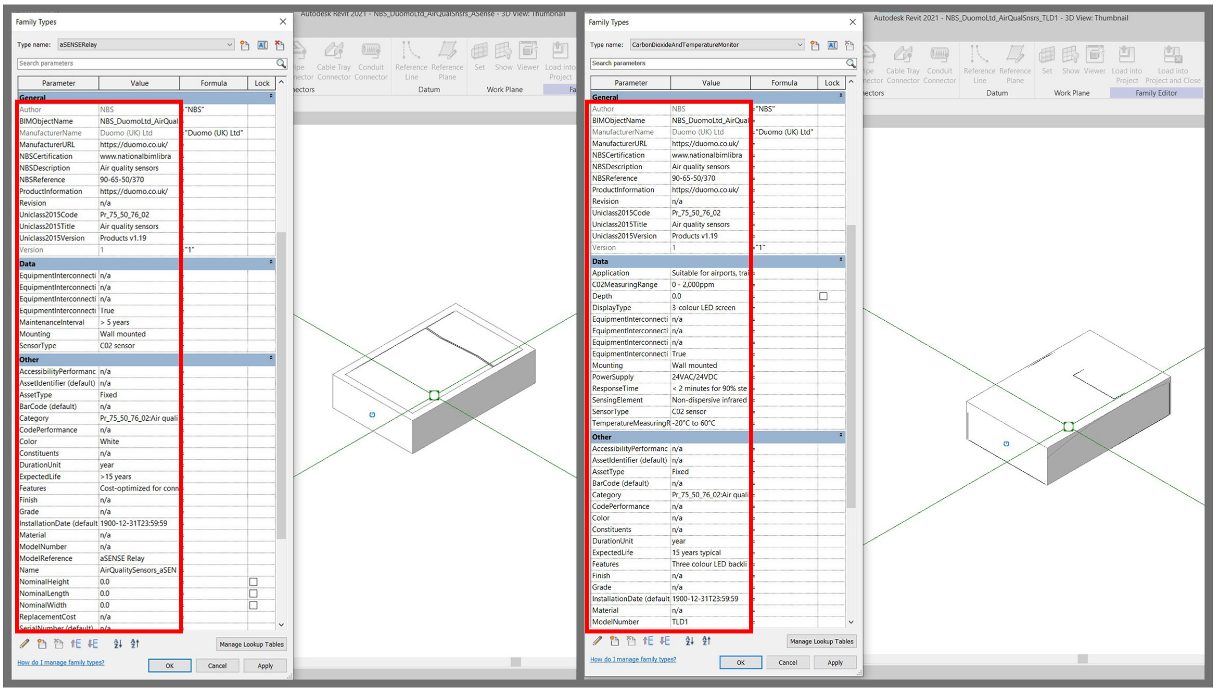


Figure 34 – Examples of different sensors downloaded from BIMobject website library

The second step is checking the necessary parameters for MKR ENV shield from 'Distrelec Group' and for BME680 from 'Adafruit'. The datasheets of both sensors were retrieved, reviewed, and used to manage the information on the BIM model. The third step was discussed earlier and shown in Figure 33 where virtual sensors' placement in the BIM model is determined location of the physical IoT devices. The results of step four are shown in Figure 35 below where the necessary geometric, alphanumeric, and documentation information is inserted for application uses.

After the modelling process of Revit sensor elements is completed, an evaluation of the model is required to confirm the completeness of the given information. This final step ensures that all property sets are filled with the necessary information and follow the same naming convention.

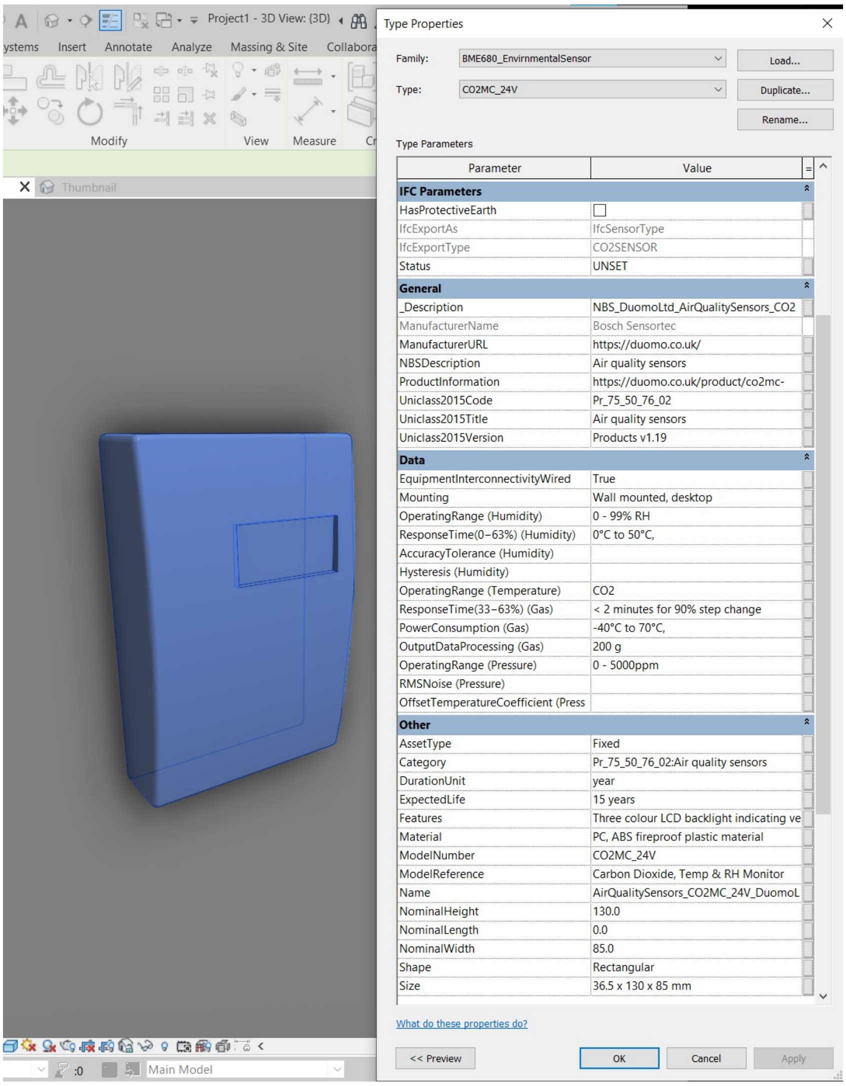


Figure 35 – BIM model information of the BME680 environmental sensor

5.4. Pilot application

Following the explanation of the proposed workflow process for implementation, the project development moves forward towards real-life application.

After modelling the additionally required objects in Revit and giving them the appropriate parameters and input data, these objects' BIM data is exported using Tracer toolkit for Revit in order to store them in an external relational database on MySQL Workbench. The relational database is formerly designed to properly maintain the BIM data after importing it. The tables of the 'Rooms' and 'Sensors' are shown in Figure 36 and Figure 37 respectively. In Figure 36, the *'room'* column stores rooms' names according to the naming convention appointed by the University of Minho website (campi.uminho.pt). The rest of the columns also follow the university website's naming convention as well.

room	function	floor	building
CA-13-03-38-02	Group Study Office	4	CA-28
CA-13-03-38-03	Library	4	CA-28
CA-13-03-38-05	Group Study Office	4	CA-28
CA-13-03-38-06	Group Study Office	4	CA-28
CA-13-03-38-07	Group Study Office	4	CA-28
CA-13-03-38-08	Library	4	CA-28
CA-13-03-38-10	Group Study Office	4	CA-28
CA-13-04-38-02	Group Study Office	5	CA-28
CA-13-04-38-03	Library	5	CA-28
CA-13-04-38-05	Group Study Office	5	CA-28
CA-13-04-38-06	Group Study Office	5	CA-28
CA-13-04-38-07	Group Study Office	5	CA-28
CA-13-04-38-08	Library	5	CA-28
CA-13-04-38-10	Group Study Office	5	CA-28
CA-13-05-38-04	Library	6	CA-28
CA-13-05-38-06	Group Study Office	6	CA-28
CA-13-05-38-07	Group Study Office	6	CA-28
CA-13-05-38-08	Group Study Office	6	CA-28
CA-13-05-38-09	Library	6	CA-28
CA-13-05-38-11	Group Study Office	6	CA-28
NULL	NULL	NULL	NULL

Figure 36 – MySQL Workbench database 'Rooms' table

In Figure 37, the *'SensorId'* column holds the ID that identifies a particular sensor. This is different from the Revit element ID that is assigned by Revit as a default which can be shown in Figure 38. The *'Shield'* column shows the different sensors that are included in the BME680 which are temperature, relative humidity, barometric pressure, and VOC gas emissions.

SensorId	Name	Shield	Room
001	BME680	ENV_T/RH/P/VOC	CA-E13-1.01
002	BME680	ENV_T/RH/P/VOC	CA-E13-1.03
003	BME680	ENV_T/RH/P/VOC	CA-E13-1.04
004	BME680	ENV_T/RH/P/VOC	CA-E13-1.06
005	BME680	ENV_T/RH/P/VOC	CA-E13-2.04
006	BME680	ENV_T/RH/P/VOC	CA-E13-2.04
007	BME680	ENV_T/RH/P/VOC	CA-E13-2.06
008	BME680	ENV_T/RH/P/VOC	CA-E13-3.02
009	BME680	ENV_T/RH/P/VOC	CA-E13-3.03
010	BME680	ENV_T/RH/P/VOC	CA-E13-3.05
011	BME680	ENV_T/RH/P/VOC	CA-E13-3.06
012	BME680	ENV_T/RH/P/VOC	CA-E13-3.07
013	BME680	ENV_T/RH/P/VOC	CA-E13-3.08
014	BME680	ENV_T/RH/P/VOC	CA-E13-3.10
015	BME680	ENV_T/RH/P/VOC	CA-E13-4.02
016	BME680	ENV_T/RH/P/VOC	CA-E13-4.03
017	BME680	ENV_T/RH/P/VOC	CA-E13-4.05
018	BME680	ENV_T/RH/P/VOC	CA-E13-4.06
NULL	NULL	NULL	NULL

Figure 37 – MySQL Workbench database 'Sensors' table

The building process of the IoT devices and setting up the connection with a network is explained in sections 3.4 and 3.5 of chapter 3. Arduino electronics are used for assembling the hardware, and the Arduino IDE manages the programming of the electronic boards. The results of the different IoT devices are placed in the library building, and connected to the library network to accumulate environmental data.

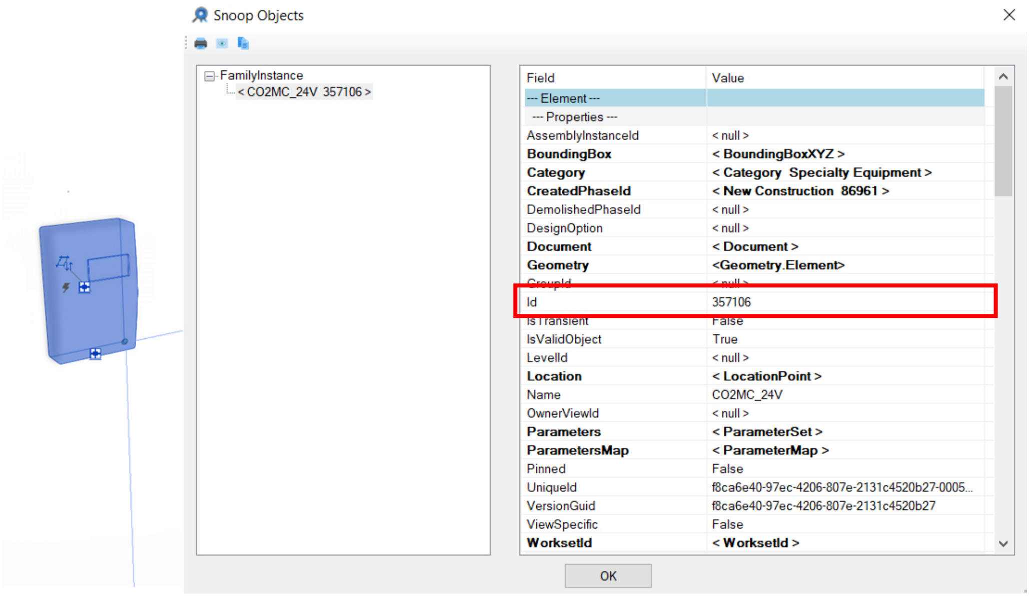


Figure 38 – RevitLookup Add-on sensor object data

After storing BIM data on a database, the environmental parameters data need to be stored on an ODS. The website www.db4free.net manages this level in the application process. The results of the data accumulation and the table structure for the MKR ENV shield parameters are shown in Figure 38. Once the network connection is established and the data is being accurately collected, it is stored on the ODS tables shown in Figure 39.

pressure	temperature	humidity	illuminance	uva	uvb	uvIndex	Timestamp
99.72	28.71	59.30	40.00	-14.53	-19.24	-0.04	2021-08-05 18:38:36
99.72	28.69	59.22	40.65	-13.53	-19.24	-0.03	2021-08-05 18:38:38
99.72	28.73	59.23	40.00	-13.20	-17.50	-0.03	2021-08-05 18:38:40
99.71	28.75	59.31	40.65	-15.75	-22.19	-0.04	2021-08-05 18:38:42
99.72	28.77	59.21	40.65	-14.53	-19.24	-0.04	2021-08-05 18:38:44
99.72	28.71	59.35	41.29	-16.75	-22.19	-0.04	2021-08-05 18:38:46
99.71	28.73	59.28	40.65	-15.75	-22.19	-0.04	2021-08-05 18:38:48
99.72	28.75	59.20	40.65	-13.20	-17.50	-0.03	2021-08-05 18:38:49
99.71	28.73	59.24	40.65	-13.20	-17.50	-0.03	2021-08-05 18:38:51
99.72	28.75	59.18	41.29	-15.75	-22.19	-0.04	2021-08-05 18:38:53
99.72	28.73	59.18	41.29	-15.75	-22.19	-0.04	2021-08-05 18:38:56
99.72	28.73	59.21	41.29	-14.53	-19.24	-0.04	2021-08-05 18:38:57
99.71	28.75	59.09	41.94	-15.75	-22.19	-0.04	2021-08-05 18:39:00
99.72	28.73	59.09	41.94	-15.75	-22.19	-0.04	2021-08-05 18:39:02
99.71	28.77	59.17	42.58	-13.53	-19.24	-0.03	2021-08-05 18:39:04
99.72	28.78	59.14	42.58	-15.75	-21.19	-0.04	2021-08-05 18:39:06
99.72	28.73	59.09	42.58	-15.75	-21.19	-0.04	2021-08-05 18:39:07
99.71	28.73	59.15	41.94	-15.75	-21.19	-0.04	2021-08-05 18:39:09
99.72	28.78	59.03	41.94	-15.75	-21.19	-0.04	2021-08-05 18:39:11
99.72	28.77	59.06	43.23	-15.75	-21.19	-0.04	2021-08-05 18:39:13
99.71	28.77	59.15	40.00	-16.75	-22.19	-0.04	2021-08-05 18:39:15
99.72	28.75	59.03	42.58	-15.75	-22.19	-0.04	2021-08-05 18:39:17
99.72	28.75	59.07	42.58	-15.75	-21.19	-0.04	2021-08-05 18:39:19
99.71	28.75	59.01	41.94	-15.75	-21.19	-0.04	2021-08-05 18:39:21

Figure 39 – Populated environmental data on the phpMyAdmin ODS

Following the data collection from the EMS and the BIM model, the next step is proceeding to the interface layer to visualize the given data and analyze it for FM. The visualization platform used is Power BI which provides a connection between MySQL databases and the platform. With the addition of the Tracer toolkit, Power BI is also able to connect Revit BIM data as well as Revit 2D and 3D views to the created interactive dashboards. This method allows for data-driven diagrams in a report that track environmental quality, and provide views of interior spaces connected directly to the BIM model, with an option to display the latest information only by clicking the 'Refresh' button on the visualization tool.

After the data is properly imported to the Power BI platform, the visualization gadgets used in the report must be supplied with data in order to function properly. Figure 40 shows the data fields of the Tracer gadget for displaying a 2D plan view of the BIM model, shown earlier in Figure 28, and some of the BIM model data embedded to the gadget. For the Tracer gadget, [LocationGeoJson] data from the 'Element' table can be transferred to the 'Geometry' field with a simple drag-and-drop action. This data is responsible for defining the geometrical outline of the plan view to be in accordance with the BIM model.

Another example is the [Name] data from the 'RoomElements' table that is added to the 'Category' field of the Tracer gadget, which is responsible for determining the classifying parameters on which the visual will operate. Since the name parameter is matched with the 'Category' field, different colors will be shown for spaces with different names and functions. From the visual, 'Library' spaces are in the color pink, 'Group Study Office' spaces are in the color blue, 'Common Area' spaces are in the color purple, and so on.

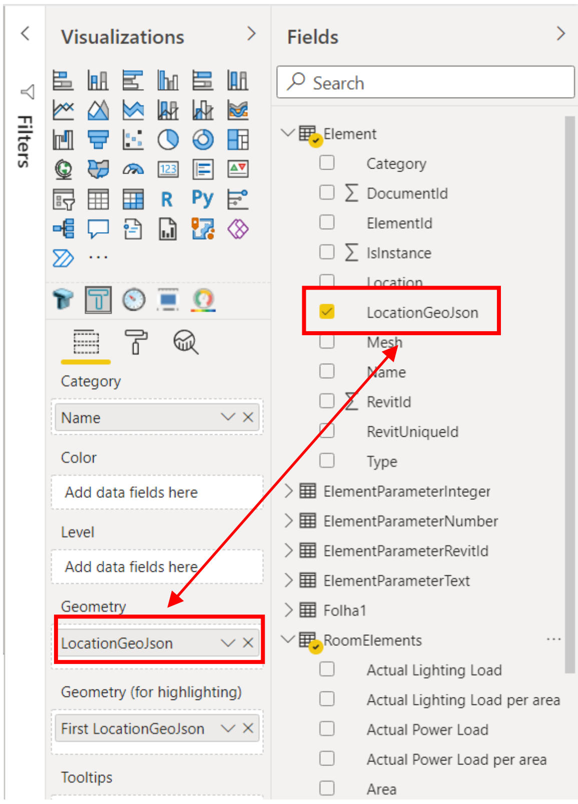


Figure 40 – Element 'Category' field matched with [LocationGeoJson] from data tables

Figure 41 presents the visualization data fields for the Dial Gauge on the left, and the Revit BIM data tables on the right. The visualization fields are matched with the appropriate BIM data which results in a data-driven visual for monitoring latest average temperature assessment. The 'Pointer Value' field determines the parameter that needs to be evaluated. This value is the temperature in this study. From the sheets table, values such as [Σ minimum] and [Σ maximum] are used to determine the average temperature extremities, and other values such as [Σ comfortable] specify the required average temperature range.

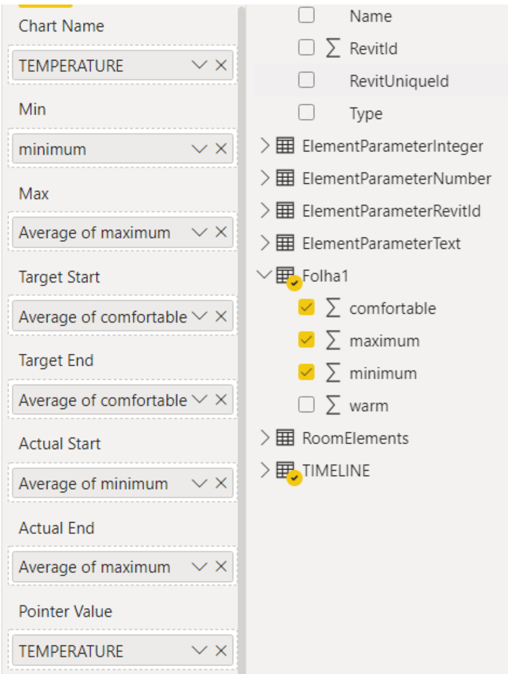


Figure 41 – Dial Gauge visualization fields filled with BIM data from the Revit tables

These values were measured and monitored over the course of a short period of time, since the free trial version of the Tracer toolkit only offers fifteen days of using the tool. However, this time gave sufficient knowledge to verify the validity of the tool for an operational workflow process from Revit and the WSN, to the databases, and finally to communicate this data with the visualization platform Power BI and the Tracer toolkit.

This workflow can be further developed to involve a two-way communication automated feedback process that does not rely on the facility manager's manual intervention, but depends on actuators that help manipulate the environmental conditions. Furthermore, the data collected over a long period of time can be put to use through (1) Machine Learning and Deep Learning techniques for the mechanical control of HVAC systems; (2) future prediction of usage intensity; (3) and longevity of IoT devices and systems devices, to boost performance and create more accurate decision-making strategies for energy conservation.

The aforementioned workflow of this study is conducted as a prototype for university facility buildings, through an experimental phase that needs to be verified, tuned, and tested with the above stated tools and platforms, or other fitting tools. For this reason, there were certain limitations regarding the budget or purchasing the software licenses specialized for FM support.

Once this workflow is implemented on a commercial-scale project, these budgetary constraints may become less of a concern for a large company, whether it is at the Service layer when purchasing a professional ODS, or at the Interface Layer when purchasing the unlimited Tracer license. The visualization platform can also be substituted by another platform in case the facility manager or the O&M team have a preferred tool that supports larger scale projects. The only requirement is that the platform is compatible with MySQL relational database structures, or whichever service is employed.

Moreover, two factors that ensure the expandability of the workflow processes are (1) that the relational databases discussed in section 2.3 are designed in a manner that allows for indefinite expansion of tables and datatypes; (2) the use of a WSN that depends on IoT network that supports long-range communication capabilities and sending data from numerous low-power devices and sensors.

This page is intentionally left blank

6. CONCLUSIONS

During the initial stages of the dissertation development, there were several topics related to FM and AECO industries and technologies that were delved into, such as BIM, sensor networks, IoT devices, monitoring systems, Indoor Environmental Quality (IEQ), RDBMS, and analytical visualization platforms. All these topics contributed to the growth of the project and the accomplishment of the case study application.

Throughout the dissertation, a BIM-centered IoT integration was able to be achieved to create an Environmental Monitoring System (EMS) that collects data to control energy consumption, resulting in an improved building performance and an increase in the occupants' comfort level. This methodology is practiced during the O&M phase of the building FM, and can be also used as a reference for implementation in other academic, commercial, or industrial facilities. Since the processes scale is expandable, this can be successfully applied to a varied number of project types, and help the facility manager adopt more effective decision-making protocols.

During the course of the dissertation, several conclusions have been reached about the workflow and how to improve some of its aspects. These conclusions can be summarized as:

- MySQL performance as a RDBMS is notable. However, there are other servers that are worth exploring that unlike MySQL, are commercial, but are high-end SQL compliant servers such as SAP ASE and IBM Db2. These two have good server architecture and high performance and scalability.

- Arduino hardware and software are fitting for such a project application, since it is an open-source platform and enables beginners to create interactive electronics. They have a wide range of high-quality microcontrollers, and their prices are relatively inexpensive, which gives them an advantage in the market.

- The combination of the Tracer toolkit and Power BI creates data-driven visual dashboard of environmental parameter values, and includes 2D or 3D views of the BIM model. Although this tool is able to function properly, it must be mentioned there are other platforms that behave in a similar way, which are open-source IoT platforms such as 'ThingsBoard' that can be integrated with BIM tools such as Revit.

Suggestions for future developments may include:

- The complete automation of the workflow process through the addition of actuators to the WSN. These actuators, such as servos and motors, create a two-way communication process through collecting environmental information and communicating actions wirelessly to the physical elements and systems without human intervention.

- The introduction of a teaching-learning process that understands, predicts, and optimizes the comfort variables. This process can be achieved with Big Data that use ML and Deep

Learning algorithms that learn from a certain behavior to create, through BIM technology, a DT environment that operates and manages the building autonomously, and insights predictive analysis that assists FM duties.

- Developing IFC based web-tools that depend on open-source packages for BIM model visualization and platforms for the database services. Packages such as ‘xeokit’ or ‘IFC.js’ can transform a web browser into an IFC viewer with high-detail, full-precision 2D/3D BIM and AEC models. An open-source BIM platform is also endorsed, but at the time of this dissertation, there are no viable alternatives that can compete with the market usage and popularity of Revit.
- The extension of the capabilities of the RDBMS over FM responsibilities to incorporate additional O&M processes such as material costs and maintenance, possible damage to furniture, fees and expenses, surveillance, and utilities.

REFERENCES

ABX00032 (2021). Available at: https://docs.arduino.cc/static/7a6ee82345412ad541c7ffee57dfabfd/ABX00032 (with headers)-datasheet.pdf (Accessed: 2 September 2021).

Ambient Light Sensor (2011). Available at: https://www.vishay.com/docs/81579/temt6000.pdf (Accessed: 28 August 2021).

Arslan, M. *et al.* (2014) 'Real-time environmental monitoring, visualization and notification system for construction H&S management', *Journal of Information Technology in Construction*, 19(September 2013), pp. 72–91. doi: 10.5840/agstm201454111.

ASHRAE (2017) *Thermal environmental conditions for human occupancy*. Atlanta: ANSI/ASHRAE Standard-55-2017.

Asirvadem, D. (2013) 'IDEF1X Introduction', p. 6. Available at: https://www.softwaregems.com.au/Documents/Documentary Examples/IDEF1X Introduction.pdf.

Becerik-Gerber, B. *et al.* (2012) 'Application Areas and Data Requirements for BIM-Enabled Facilities Management', *Journal of Construction Engineering and Management*, 138(3), pp. 431–442. doi: 10.1061/(ASCE)CO.1943-7862.0000433.

Boddupalli, C. *et al.* (2019) 'Improved visualization of infrastructure monitoring data using building information modeling', *Structure and Infrastructure Engineering*, 15(9), pp. 1247–1263. doi: 10.1080/15732479.2019.1602150.

Bortoluzzi, B. *et al.* (2019) 'Automating the creation of building information models for existing buildings', *Automation in Construction*, 105, p. 102838. doi: 10.1016/j.autcon.2019.102838.

Cahill, B., Menzel, K. and Flynn, D. (2012) 'BIM as a center piece for optimized building operation', *Taylor & Francis Group, London.*

Capacitive digital sensor for relative humidity and temperature (2016). Available at: https://www.st.com/resource/en/datasheet/hts221.pdf (Accessed: 28 August 2021).

Chamberlin, D. D. (2012) 'Early History of SQL', *IEEE Annals of the History of Computing*, 34(4), pp. 78–82. doi: 10.1109/MAHC.2012.61.

Chamberlin, D. D. and Boyce, R. F. (1974) 'SEQUEL: A STRUCTURED ENGLISH QUERY LANGUAGE', *IBM Research Laboratory*, pp. 249–264. Available at: https://web.archive.org/web/20070926212100/http://www.almaden.ibm.com/cs/people/chamberlin/sequel-1974.pdf.

Charles, K. *et al.* (2005) 'Indoor air quality and thermal comfort in open-plan offices', *Institute for Research in Construction Construction Technology Update*, p. 64.

Cheng, J. *et al.* (2016) 'A BIM-based decision support system framework for predictive maintenance management of building facilities', *See.Eng.Osaka-U.Ac.Jp*, (4), pp. 711–718. Available at: http://www.see.eng.osaka-u.ac.jp/seeit/icccbe2016/Proceedings/Full_Papers/090-102.pdf.

Cheng, J. C. P. *et al.* (2020) 'Data-driven predictive maintenance planning framework for MEP components based on BIM and IoT using machine learning algorithms', *Automation in Construction*, 112, p. 103087. doi: 10.1016/j.autcon.2020.103087.

Chung, S. *et al.* (2018) 'Smart Facility Management Systems Utilizing Open BIM and Augmented/Virtual Reality', in. doi: 10.22260/ISARC2018/0118.

Codeacademy (2021) *What is a Relational Database Management System?* Available at: https://www.codecademy.com/articles/what-is-rdbms-sql#:~:text=SQL (Structured Query Language) is,write%2C read%2C and interpret. (Accessed: 30 August 2021).

Dave, B. *et al.* (2018) 'A framework for integrating BIM and IoT through open standards', *Automation in Construction*, 95(August), pp. 35–45. doi: 10.1016/j.autcon.2018.07.022.

Davila Delgado, J. M. *et al.* (2018) 'Structural Performance Monitoring Using a Dynamic Data-Driven BIM Environment', *Journal of Computing in Civil Engineering*, 32(3), p. 04018009. doi: 10.1061/(ASCE)CP.1943-5487.0000749.

Desogus, G. *et al.* (2021) 'Bim and iot sensors integration: A framework for consumption and indoor conditions data monitoring of existing buildings', *Sustainability (Switzerland)*, 13(8). doi: 10.3390/su13084496.

Eastman, C. *et al.* (2011) *BIM Handbook, a Guide to Building Information Modelling 2nd ed.*, *John Wiley & Sons, Inc, Hoboken*.

EN 17412-1 (2020) 'Building Information Modelling. Level of Information Need. Concepts and principles', p. 28.

Falkenberg, H. K. *et al.* (2019) 'Improved indoor lighting improved healthy aging at home – an intervention study in 77-year-old Norwegians', *Journal of Multidisciplinary Healthcare*, Volume 12, pp. 315–324. doi: 10.2147/JMDH.S198763.

Fitz, T., Theiler, M. and Smarsly, K. (2019) 'A metamodel for cyber-physical systems', *Advanced Engineering Informatics*, 41, p. 100930. doi: 10.1016/j.aei.2019.100930.

Gerrish, T. *et al.* (2015) 'Attributing in-use building performance data to an as-built building information model for lifecycle building performance management', *Proceedings of the 32nd CIB W78 Conference*, pp. 1–11. Available at: https://dspace.lboro.ac.uk/dspace-jspui/handle/2134/19745.

Gray, J. *et al.* (2017) 'Wireless data management system for environmental monitoring in livestock buildings', *Information Processing in Agriculture*, 4(1), pp. 1–17. doi: 10.1016/j.inpa.2016.12.001.

Gubbi, J. *et al.* (2013) 'Internet of Things (IoT): A vision, architectural elements, and future directions', *Future Generation Computer Systems*, 29(7), pp. 1645–1660. doi: 10.1016/j.future.2013.01.010.

Hagsheno, S. *et al.* (2019) 'Digitale Technologien und deren Wertschöpfungspotenziale für die Bauwirtschaft/Identification and description of relevant digital technologies for the construction industry', *Bauingenieur*, 94(02), pp. 45–55. doi: 10.37544/0005-6650-2019-02-31.

Haltian (2021) *Thingsboard*. Available at: https://support.haltian.com/knowledgebase/thingsboard/ (Accessed: 26 October 2021).

Hayat, H. *et al.* (2019) 'The state-of-the-art of sensors and environmental monitoring technologies in buildings', *Sensors (Switzerland)*, 19(17). doi: 10.3390/s19173648.

Hilal, M. *et al.* (2019) 'A hybrid conceptual model for BIM in FM', *Construction Innovation*, 19(4), pp. 531–549. doi: 10.1108/CI-05-2018-0043.

IFMA (2018) *What is Facility Management*. Available at: https://www.ifma.org/about/what-is-facility-management (Accessed: 22 September 2021).

IoT Middleware Market (2021) *IoT Middleware Market - Growth, Trends, COVID-19 Impact, and Forecasts (2021 - 2026)*. Global. Available at: https://www.researchandmarkets.com/reports/4591776/iot-middleware-market-growth-trends-covid-19?utm_source=GNOM&utm_medium=PressRelease&utm_code=hsm7q8&utm_campaign=1482270+-+Worldwide+Industry+for+IoT+Middleware+to+2025+-+Manufacturing+Expected+to+Have+H (Accessed: 28 August 2021).

Kang, K. *et al.* (2018) 'Monitoring Framework for Utility Tunnels based on BIM and IoT Technology', *Conference: 17th International Conference on Computing in Civil and Building Engineering*, (December), pp. 1–9. Available at: https://www.researchgate.net/publication/329555180_Monitoring_Framework_for_Utility_Tunnels_based_on_BIM_and_IoT_Technology.

Kang, K., Lin, J. and Zhang, J. (2018) 'BIM- and IoT-based monitoring framework for building performance management', *Journal of Structural Integrity and Maintenance*, 3(4), pp. 254–261. doi: 10.1080/24705314.2018.1536318.

Katranuschkov, P. *et al.* (2010) 'BIM-BASED GENERATION OF MULTI-MODEL VIEWS', (1), pp. 16–18.

Kaur, N. and Monga, S. (2014) 'Comparsions of Wired and Wireless Networks: A Review', *International Journal of Advanced Engineering Technology*, V(II), pp. 34–35.

Kensek, K. M. (2014) 'Integration of Environmental Sensors with BIM: case studies using Arduino, Dynamo, and the Revit API', *Informes de la Construcción*, 66(536), p. e044. doi: 10.3989/ic.13.151.

Krishnamurthy, S. *et al.* (2008) 'Automation of facility management processes using machine-to-machine technologies', *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 4952 LNCS, pp. 68–86. doi: 10.1007/978-3-540-78731-0_5.

lady ada (2021) *Adafruit BME680*. Available at: https://cdn-learn.adafruit.com/downloads/pdf/adafruit-bme680-humidity-temperature-barometic-pressure-voc-gas.pdf (Accessed: 8 August 2021).

Laird, S. (1994) 'Total Facilities Management', *Facilities*, 12(13), pp. 25–26. doi: 10.1108/02632779410795396.

Lin, Y.-C. and Cheung, W.-F. (2020) 'Developing WSN/BIM-Based Environmental Monitoring Management System for Parking Garages in Smart Cities', *Journal of Management in Engineering*, 36(3), p. 04020012. doi: 10.1061/(ASCE)ME.1943-5479.0000760.

Liu, R. and Issa, R. R. A. (2012) '3D Visualization of Sub-Surface Pipelines in Connection with the Building Utilities: Integrating GIS and BIM for Facility Management', in *Computing in Civil Engineering (2012)*. Reston, VA: American Society of Civil Engineers, pp. 341–348. doi: 10.1061/9780784412343.0043.

Ma, Liu and Shang (2019) 'A Building Information Model (BIM) and Artificial Neural Network (ANN) Based System for Personal Thermal Comfort Evaluation and Energy Efficient Design of Interior Space', *Sustainability*, 11(18), p. 4972. doi: 10.3390/su11184972.

Malheiro, A. *et al.* (2020) *POLITECNICO DI MILANO Master in Building Information Modelling Data flow from BIM to Digital Twins*.

Mannino, A. *et al.* (2021) 'Building information modelling and internet of things integration for facility management-literature review and future needs', *Applied Sciences (Switzerland)*. MDPI AG. doi: 10.3390/app11073062.

Marzouk, M. and Abdelaty, A. (2014a) 'BIM-based framework for managing performance of subway stations', *Automation in Construction*, 41, pp. 70–77. doi: 10.1016/j.autcon.2014.02.004.

Marzouk, M. and Abdelaty, A. (2014b) 'Monitoring thermal comfort in subways using building information modeling', *Energy and Buildings*, 84, pp. 252–257. doi: 10.1016/j.enbuild.2014.08.006.

MEMS nano pressure sensor: 260-1260 hPa absolute digital output barometer (2017). Available at: https://www.st.com/resource/en/datasheet/dm00140895.pdf (Accessed: 28 August 2021).

Natephra, W. *et al.* (2017) 'Integrating 4D thermal information with BIM for building envelope thermal performance analysis and thermal comfort evaluation in naturally ventilated environments', *Building and Environment*, 124, pp. 194–208. doi: 10.1016/j.buildenv.2017.08.004.

Natephra, W. and Motamedi, A. (2019) *BIM-BASED LIVE SENSOR DATA VISUALIZATION USING VIRTUAL REALITY FOR MONITORING INDOOR CONDITIONS*.

National Academies Press (1995) *Expanding the Vision of Sensor Materials*. Washington, D.C.: National Academies Press. doi: 10.17226/4782.

Nguyen, H. T. (2016) 'Integration of BIM and IoT to improve the building performance for occupants' perspective', (June), p. 84. Available at: https://www.researchgate.net/publication/310481201.

O'Flynn, B. *et al.* (2010) 'Development of miniaturized wireless sensor nodes suitable for building energy management and modelling', in *eWork and eBusiness in Architecture, Engineering and Construction*. CRC Press, pp. 253–258. doi: 10.1201/b10527-42.

Omar, N. S. *et al.* (2018) 'Developing of building maintenance management by using bim', *International Journal of Civil Engineering and Technology*, 9(11), pp. 1371–1383.

Osibona, O. *et al.* (2021) 'Lighting in the Home and Health: A Systematic Review', *International Journal of Environmental Research and Public Health*, 18(2), p. 609. doi: 10.3390/ijerph18020609.

Peacock, J. *et al.* (2007) 'The RAINFOR database: monitoring forest biomass and dynamics', *Journal of Vegetation Science*, 18(4), pp. 535–542. doi: 10.1111/j.1654-1103.2007.tb02568.x.

PerformanceDBA (2019) *Relational Database schema design for metric storage*. Available at: https://stackoverflow.com/questions/54823674/relational-database-schema-design-for-metric-storage (Accessed: 30 August 2021).

ProvingGround (2020) *Tracer*. Available at: https://provingground.io/tools/tracer/ (Accessed: 25 September 2021).

ProvingGround (2021) *Tracer Revit Addin Overview*. Available at: https://apps.provingground.io/docs/tracer-v1-0-documentation/tracer-revit-export-addin/overview/ (Accessed: 3 October 2021).

Riaz, Z. *et al.* (2017) 'BIM and sensor-based data management system for construction safety monitoring', *Journal of Engineering, Design and Technology*, 15(6), pp. 738–753. doi: 10.1108/JEDT-03-2017-0017.

Rio, J. *et al.* (2013) 'Expansion of IFC model with structural sensors', *Informes de la Construcción*, 65(530), pp. 219–228. doi: 10.3989/ic.12.043.

Rizal, R. and Hikmatyar, M. (2019) 'Investigation Internet of Things (IoT) Device using Integrated Digital Forensics Investigation Framework (IDFIF)', *Journal of Physics: Conference Series*, 1179, p. 012140. doi: 10.1088/1742-6596/1179/1/012140.

Trabulci, M. B. (2020) *Universidade do Minho Escola de Engenharia Interoperability framework for BIM-FM based on a relational database*.

U-blox AG (2021) *NINA-W10 series*. Available at: https://www.u-blox.com/sites/default/files/NINA-W10_ProductSummary_UBX-17051775.pdf (Accessed: 10 September 2021).

Underwood, J. and Isikdag, U. (2011) 'Emerging technologies for BIM 2.0', *Construction Innovation*, 11(3), pp. 252–258. doi: 10.1108/14714171111148990.

UVA and UVB Light Sensor with I2C Interface (2019). Available at: https://web.archive.org/web/20190614090419/http://www.vishay.com/docs/84304/veml6075.pdf (Accessed: 28 August 2021).

Valinejadshoubi, M. *et al.* (2019) 'Development of a BIM-Based Data Management System for Structural Health Monitoring with Application to Modular Buildings: Case Study', *Journal of Computing in Civil Engineering*, 33(3), p. 05019003. doi: 10.1061/(ASCE)CP.1943-5487.0000826.

Valinejadshoubi, M. *et al.* (2021) 'Development of an IoT and BIM-based automated alert system for thermal comfort monitoring in buildings', *Sustainable Cities and Society*, 66. doi: 10.1016/j.scs.2020.102602.

Valinejadshoubi, M., Moselhi, O. and Bagchi, A. (2021) 'Integrating BIM into sensor-based facilities management operations', *Journal of Facilities Management*, (November). doi: 10.1108/JFM-08-2020-0055.

Villa, V. *et al.* (2021) 'Iot open-source architecture for the maintenance of building facilities', *Applied Sciences (Switzerland)*, 11(12). doi: 10.3390/app11125374.

Wehbe, R. and Shahrour, I. (2019) 'Use of BIM and Smart Monitoring for buildings' Indoor Comfort Control', *MATEC Web of Conferences*. Edited by I. Shahrour, X.-Y. Xie, and H. Bian, 295, p. 02010. doi: 10.1051/matecconf/201929502010.

WHO (1990) 'Indoor environment: Health aspects of air quality, thermal environment, light and noise', (November 1985), p. 128.

Wong, J. K. W. *et al.* (2018) 'Digitisation in facilities management: A literature review and future research directions', *Automation in Construction*, 92, pp. 312–326. doi: 10.1016/j.autcon.2018.04.006.

Wösten, J. H. . *et al.* (1999) 'Development and use of a database of hydraulic properties of European soils', *Geoderma*, 90(3–4), pp. 169–185. doi: 10.1016/S0016-7061(98)00132-3.

Wu, I.-C. and Liu, C.-C. (2020) 'A Visual and Persuasive Energy Conservation System Based on BIM and IoT Technology', *Sensors*, 20(1), p. 139. doi: 10.3390/s20010139.

Xiao, Y.-Q. *et al.* (2019) 'Automatically Generating a MEP Logic Chain from Building Information Models with Identification Rules', *Applied Sciences*, 9(11), p. 2204. doi: 10.3390/app9112204.

Yin, H. (2010) 'Building Management System to support building renovation', *The Boolean: Snapshots of Doctoral Research at University College Cork*. Edited by A. Kelly et al., (2010), pp.

164–169. doi: 10.33178/boolean.2010.37.

Zaballos, A. *et al.* (2020) 'A smart campus' digital twin for sustainable comfort monitoring', *Sustainability (Switzerland)*, 12(21), pp. 1–33. doi: 10.3390/su12219196.

Zhang, Y. and Bai, L. (2015) 'Rapid structural condition assessment using radio frequency identification (RFID) based wireless strain sensor', *Automation in Construction*, 54, pp. 1–11. doi: 10.1016/j.autcon.2015.02.013.

Zhong, B. *et al.* (2018) 'Ontology-based framework for building environmental monitoring and compliance checking under BIM environment', *Building and Environment*, 141, pp. 127–142. doi: 10.1016/j.buildenv.2018.05.046.

Zomorodian, Z. S. *et al.* (2016) 'Thermal comfort in educational buildings: A review article', *Renewable and Sustainable Energy Reviews*, 59, pp. 895–906. doi: 10.1016/j.rser.2016.01.033.

LIST OF ACRONYMS AND ABBREVIATIONS

ACID	Atomicity, Consistency, Isolation, and Durability
ADC	Analog-to-digital converter
AECO	Architecture, Engineering, Construction, and Operation
AI	Artificial Intelligence
API	Application Programming Interface
BIM	Building Information Modelling
CDE	Common Data Environment
CS	Chip Select
CPS	Cyber-Physical System
DBMS	Database Management System
DF	Daylight Factor
DT	Digital Twin
EMS	Environmental Monitoring System
ERD	Entity-Relationship Diagram
FK	Foreign Key
FM	Facility Management
FMS	Facility Management Systems
GND	Ground
GPIO	General-Purpose Input/Output
HVAC	Heating, Ventilation, and Air Conditioning
IDE	Integrated Development Environment
IEQ	Indoor Environmental Quality
IFC	Industry Foundation Classes
IO	Input-output
IoT	Internet of Things
LED	Light Emitting Diode
Li-ion	Lithium Ion
Lipo	Lithium Polymer
LOD	Level of Development
LOIN	Level of Information Need
MCU	Microcontroller Unit
MEP	Mechanical, Electrical, and Plumbing
MicroSD	Micro Secure Digital
ML	Machine Learning
MOX	Metal Oxide
O&M	Operations and Maintenance
ODS	Online Database Service
OS	Operating System
PK	Primary Key
RDBMS	Relational Database Management System

RFID	Radio Frequency Identification
RH	Relative Humidity
SCK	SPI Clock
SDI	Serial Data In
SDO	Serial Data Out
SPI	Serial Peripheral Interface
SQL	Structured Query Language
SSID	Service Set Identifier
USB	Universal Serial Bus
UV	Ultraviolet
Vin	Input Voltage
VOC	Volatile Organic Compounds
WHO	World Health Organization
WiFi	Wireless Fidelity
WSN	Wireless Sensor Network

APPENDICES

APPENDIX 1: ARDUINO MKR 1010 WIFI SRCIPT

This appendix comprises of the Arduino script used for this research study case to run the Arduino MKR 1010 WiFi microcontroller in order to collect environmental parameter values with the MKR ENV Shield for the monitoring system.

- **ARDUINO MKR 1010 WIFI – MAIN CODE**

```
// libraries required
#include <SPI.h>                              //required for WiFiNINA
#include <WiFiNINA.h>                         //main library to use mkr1010
wifi board
#include <MySQL_Connection.h>                   //required to connect to sql
#include <MySQL_Cursor.h>                              //required to run sql
queries
#include <Arduino_MKRENV.h>                   //required to use mkr env shield

#include "arduino_secrets.h"                   //stores all the sensitive data

#define READ_INTERVAL 1000                   //interval between data readings
#define READ_DELAY     500                   //delay to make sure data is read

// variables to handle wifi connection
char ssid[] = SECRET_SSID;                     //taken from arduino_secrets.h
char pass[] = SECRET_PASS;                     //taken from arduino_secrets.h
//char username[] = SECRET_USER;
//char identity[] = SECRET_ANONYMOUS;


int status = WL_IDLE_STATUS;                     //taken from WiFiNINA.h this
is a typical value used for initialization

// variables to handle MySQL server
IPAddress server_addr(85,10,205,173);        //IP of
https://www.db4free.net/ "ping www.db4free.net" on the Command Prompt
char user[] = SECRET_USERDB;                   //taken from arduino_secrets.h
char password[] = SECRET_PASSDB;               //taken from arduino_secrets.h

// variable to store the environment parameters
float pressure ;
float temperature ;
float humidity ;
float illuminance ;
float uva ;
float uvb ;
float uvIndex ;

// variables to store the sql query template and its final value (after
replacing the environmental values for the current ones)
```

```
char SELECT_SQL[] = "SELECT * FROM sensors_database.MKR1010";
char INSERT_SQL[] = "INSERT INTO sensors_database.MKR1010
(pressure,temperature,humidity,illuminance,uva,uvb,uvIndex) VALUES (%s, %s,
%s, %s, %s, %s, %s)";
char query[300];
char szpressure[10] ;
char sztemperature[10] ;
char szhumidity[10] ;
char szilluminance[10] ;
char szuva[10] ;
char szuvb[10] ;
char szuvIndex[10] ;

// instanciate the objects for wifi and sql connections
WiFiClient client;
MySQL_Connection conn((Client *)&client);

// declaration of some auxiliary functions
void conectWifi();
void conectSQL();
void readENV();
void printENV();
void sendData();
void getData();
char* dtostrf (double, signed char, unsigned char, char*);

// **************** SETUP function only runs once when the sketch loads
*************************
void setup() {
  Serial.begin(115200);                                                  //
opens serial port for communication between the Arduino board and a
computer or other devices
                                                                                   //

  pinMode(LED_BUILTIN, OUTPUT);                                   // sets
board's built in led to work as output
  digitalWrite(LED_BUILTIN, LOW);                         // turns built in
led off (in case it was on form some reason)

  while (!Serial);                                                       //
program will not move forward until it cannot communicate with arduino
board

  Serial.println("Connecting to wifi:");
  conectWifi();                                                          //
function to deal with the wifi connection process

  Serial.println("\nConnecting to MySQL server:");
  conectSQL();                                                           //
function to deal with the SQL connection process

  digitalWrite(LED_BUILTIN, HIGH);                    // turns built in led
on since we were able to successfully connect to wifi and SQL

  Serial.println("\nConnecting to ENV shield:");
  conectENV();
}

// **************** LOOP function runs continously until we stop it
*************************
void loop() {
  Serial.println("\n\nReading data from MKR ENV...");
```

```
  readENV();
  //printENV();

  delay(READ_DELAY);                                    //small delay to make
sure data was read

  Serial.println("\nSending data to MySQL server...");
  sendData();                                           //function to send data
to MySQL server

  delay(READ_INTERVAL);

  //getData();                                          //function to get data
from MySQL server
}

// ***************** AUXILIARY functions **************************

void conectWifi() {
  // check the communication with the integrated WiFi module
  if (WiFi.status() == WL_NO_MODULE) {
    Serial.println("Communication with WiFi module failed!");

    while (true);                                                   //
don't continue and blocks program at this point
  }

  // check if firmware is updated
  String fv = WiFi.firmwareVersion();
  if (fv < WIFI_FIRMWARE_LATEST_VERSION) {
    Serial.println("Please upgrade the firmware");
  }

  // attempt to connect to Wifi network:
  while (status != WL_CONNECTED) {
    Serial.print("Attempting to connect to SSID: ");
    Serial.println(ssid);


    if (WiFi.status() != WL_IDLE_STATUS) {
      WiFi.end(); //disconnect from wifi to set new wifi connection
    }
    delay(2000);
    status = WiFi.begin(ssid, pass);
                           // Connect to WPA/WPA2 network (use this at
home)
    //status = WiFi.beginEnterprise(ssid, user, pass,identity, ca);  //
Connect to eduroam (not working!)

    //printStatus();

    delay(5000);
                           // wait 10000 miliseconds (10 seconds) for
connection
  }
  Serial.println("Connected to wifi");

  // print your board's IP address:
  IPAddress ip = WiFi.localIP();
  Serial.print("IP Address: ");
  Serial.println(ip);
```

```
  // print the received signal strength:
  long rssi = WiFi.RSSI();
  Serial.print("signal strength (RSSI):");
  Serial.print(rssi);
  Serial.println(" dBm");
}

void printStatus(){
  switch(status){
    case WL_CONNECTED:
      Serial.println("WL_CONNECTED");
      break;
    case WL_NO_SHIELD:
      Serial.println("WL_NO_SHIELD");
      break;
    case WL_IDLE_STATUS:
      Serial.println("WL_IDLE_STATUS");
      break;
    case WL_NO_SSID_AVAIL:
      Serial.println("WL_NO_SSID_AVAIL");
      break;
    case WL_SCAN_COMPLETED:
      Serial.println("WL_SCAN_COMPLETED");
      break;
    case WL_CONNECT_FAILED:
      Serial.println("WL_CONNECT_FAILED");
      break;
    case WL_CONNECTION_LOST:
      Serial.println("WL_CONNECTION_LOST");
      break;
    case WL_DISCONNECTED:
      Serial.println("WL_DISCONNECTED");
      break;
    default:
      break;
  }
}

void conectSQL(){
  while (!conn.connect(server_addr, 3306, user, password)) {
    Serial.println("Unable to connect to MySQL server.");
    conn.close();                                                    // close
this connection attempt before start a new one
    delay(1000);                                                     // wait
for 1 second before retry connection
    Serial.println("Trying again to connect to MySQL server.");
  }
  Serial.println("Connected to MySQL server!");
}

void conectENV(){
  while (!ENV.begin()) {
  Serial.println("Failed to initialize MKR ENV shield! Retrying...");
 }
 Serial.println("MKR ENV shield successfully initialized!");
}

void readENV(){
  // read all the sensor values
  pressure    = ENV.readPressure();
```

```
  temperature = ENV.readTemperature();
  humidity    = ENV.readHumidity();
  illuminance = ENV.readIlluminance();
  uva         = ENV.readUVA();
  uvb         = ENV.readUVB();
  uvIndex     = ENV.readUVIndex();
}

void printENV(){
  // print each of the sensor values
  Serial.print("Temperature = ");
  Serial.print(temperature);
  Serial.println(" C");

  Serial.print("Humidity    = ");
  Serial.print(humidity);
  Serial.println(" %");

  Serial.print("Pressure    = ");
  Serial.print(pressure);
  Serial.println(" kPa");

  Serial.print("Illuminance = ");
  Serial.print(illuminance);
  Serial.println(" lx");

  Serial.print("UVA         = ");
  Serial.println(uva);

  Serial.print("UVB         = ");
  Serial.println(uvb);

  Serial.print("UV Index    = ");
  Serial.println(uvIndex);
}

void sendData() {
  // convert all read float values to string
  dtostrf(pressure,9,2,szpressure);
  dtostrf(temperature,9,2,sztemperature);
  dtostrf(humidity,9,2,szhumidity);
  dtostrf(illuminance,9,2,szilluminance);
  dtostrf(uva,9,2,szuva);
  dtostrf(uvb,9,2,szuvb);
  dtostrf(uvIndex,9,2,szuvIndex);

  //update the query with the new values
  sprintf(query, INSERT_SQL, szpressure, sztemperature, szhumidity,
szilluminance, szuva, szuvb, szuvIndex);

  // Initiate the query class instance
  MySQL_Cursor *cur_mem = new MySQL_Cursor(&conn);

  // Execute the query
  boolean res = cur_mem->execute(query);

  // Note: since there are no results, we do not need to read any data
  // Deleting the cursor also frees up memory used
  delete cur_mem;

  if(!res) Serial.println("Data successfuly sent!");
```

```
}

void getData(){
  row_values *row = NULL;
  long value = 0;
  // Initiate the query class instance
  MySQL_Cursor *cur_mem = new MySQL_Cursor(&conn);
  // Execute the query
  boolean res = cur_mem->execute(SELECT_SQL);
  Serial.println(res);

  // Fetch the columns (required) but we don't use them.
  column_names *columns = cur_mem->get_columns();

  // Read the rows
  do {
    row = cur_mem->get_next_row();
    if (row != NULL) {
      value = atol(row->values[0]);
      //Serial.println(value);
    }
  } while (row != NULL);
  // Deleting the cursor also frees up memory used
  delete cur_mem;
  Serial.println(value);
}

char *dtostrf (double val, signed char width, unsigned char prec, char
*sout) {
  char fmt[20];
  sprintf(fmt, "%%%d.%df", width, prec);
  sprintf(sout, fmt, val);
  return sout;
}
```

- **ARDUINO MKR 1010 WIFI - SENSITIVE DATA**

```
// variables to handle wifi connection
#define SECRET_SSID "network12345"                        // wifi network
SSID (name)
#define SECRET_PASS "<REDACTED>"                          // wifi network
password

// variables to handle MySQL server
#define SECRET_USERDB "username"                         // MySQL user login
username
#define SECRET_PASSDB "<REDACTED>"          // MySQL user login password
```

APPENDIX 2: ARDUINO NANO 33 IOT SCRIPT

This appendix comprises of the Arduino script used for this research study case to run the Arduino Nano 33 IoT microcontroller in order to collect environmental parameter values with the BME680 sensor for the monitoring system.

- **ARDUINO NANO 33 IOT – MAIN CODE**

```
// libraries required
#include <SPI.h>                                              //required for
WiFiNINA
#include <WiFiNINA.h>                                   //main library to use
nano 33 IoT board
#include <MySQL_Connection.h>                      //required to connect to sql
#include <MySQL_Cursor.h>                               //required to run sql
queries

//parameters for BME680
#include <Adafruit_Sensor.h>
#include "Adafruit_BME680.h"
#define BME_SCK 13
#define BME_MISO 12
#define BME_MOSI 11
#define BME_CS 10
#define SEALEVELPRESSURE_HPA (1013.25)

#include "arduino_secrets.h"           //stores all the sensitive data

#define READ_INTERVAL 1000            //interval between data readings
#define READ_DELAY     500            //delay to make sure data is read

// variables to handle wifi connection
char ssid[] = SECRET_SSID;             //taken from arduino_secrets.h
char pass[] = SECRET_PASS;             //taken from arduino_secrets.h
//char username[] = SECRET_USER;       //taken from arduino_secrets.h
int status = WL_IDLE_STATUS;           //taken from WiFiNINA.h this is a
tipical value used for initialization

// variables to handle MySQL server
IPAddress server_addr(85,10,205,173); //IP of https://www.db4free.net/
char user[] = SECRET_USERDB;        //taken from arduino_secrets.h
char password[] = SECRET_PASSDB;    //taken from arduino_secrets.h

// variable to store the environment parameters
float pressure ;
float temperature ;
float humidity ;
//float illuminance ;
//float uva ;
//float uvb ;
//float uvIndex ;
float gasResistance ;
float altitud ;
```

```
Adafruit_BME680 bme(BME_CS); // hardware SPI

// variables to store the sql query template and its final value (after
replacing the environmental values for the current ones)
char SELECT_SQL[] = "SELECT * FROM sensors_database.NANOIoT33";
char INSERT_SQL[] = "INSERT INTO sensors_database.NANOIoT33
(pressure,temperature,humidity,gasResistance,altitud) VALUES (%s, %s, %s,
%s, %s)";
char query[300];
char szpressure[10] ;
char sztemperature[10] ;
char szhumidity[10] ;
char szilluminance[10] ;
char szuva[10] ;
char szuvb[10] ;
char szuvIndex[10] ;
char szgasResistance[10];
char szaltitud[10];

// instanciate the objects for wifi and sql connections
WiFiClient client;
MySQL_Connection conn((Client *)&client);

// declaration of some auxiliary functions
void conectWifi();
void conectSQL();
void readENV();
void printENV();
void sendData();
void getData();
char* dtostrf (double, signed char, unsigned char, char*);

// ***************** SETUP function only runs once when the sketch loads
*************************
void setup() {
  Serial.begin(115200);                                          // opens
serial port for communication between the Arduino board and a computer or
other devices

  pinMode(LED_BUILTIN, OUTPUT);                         // sets board's built
in led to work as output
  digitalWrite(LED_BUILTIN, LOW);               // turns built in led off (in
case it was on form some reason)

  while (!Serial);                                                     //
program will not move forward until it cannot communicate with arduino
board

  Serial.println("Connecting to wifi:");
  conectWifi();                                                          //
function to deal with the wifi connection process

  Serial.println("\nConnecting to MySQL server:");
  conectSQL();                                                           //
function to deal with the SQL connection process

  digitalWrite(LED_BUILTIN, HIGH);     // turns built in led on since we
were able to successfully connect to wifi and SQL

  Serial.println("\nConnecting to BME680:");
  conectENV();
```

```
}

// ********* LOOP function runs continously until we stop it ************
void loop() {

  Serial.println("\nReading data from BME680...");
  readENV();
  delay(10000);
  printENV();

  delay(READ_DELAY);                    //small delay to make sure data was
read

  Serial.println("\nSending data to MySQL server...");
  sendData();                           //function to send data to MySQL server

  delay(READ_INTERVAL);

  //getData();
}

// ***************** AUXILIARY functions **************************

void conectWifi() {
  // check the communication with the integrated WiFi module
  if (WiFi.status() == WL_NO_MODULE) {
    Serial.println("Communication with WiFi module failed!");

    while (true);                                                   // don't
continue and blocks program at this point
  }

  // check if firmware is updated
  String fv = WiFi.firmwareVersion();
  if (fv < WIFI_FIRMWARE_LATEST_VERSION) {
    Serial.println("Please upgrade the firmware");
  }

    Serial.print("Number of available WiFi networks discovered:");
    Serial.println(WiFi.scanNetworks());

  // attempt to connect to Wifi network:
  while (status != WL_CONNECTED) {
    Serial.print("Attempting to connect to SSID: ");
    Serial.println(ssid);

     status = WiFi.begin(ssid, pass);          // Connect to WPA/WPA2
network. Change this line if using open or WEP network

    delay(10000);
         // wait 10000 miliseconds (10 seconds) for connection
  }
  Serial.println("Connected to wifi");

  // print your board's IP address:
  IPAddress ip = WiFi.localIP();
  Serial.print("IP Address: ");
  Serial.println(ip);

  // print the received signal strength:
```

```
  long rssi = WiFi.RSSI();
  Serial.print("signal strength (RSSI):");
  Serial.print(rssi);
  Serial.println(" dBm");
}

void conectSQL(){
  while (!conn.connect(server_addr, 3306, user, password)) {
    Serial.println("Unable to connect to MySQL server.");
    conn.close();                                                   // close
this connection attempt before start a new one
    delay(1000);                                                    // wait
for 1 second before retry connection
    Serial.println("Trying again to connect to MySQL server.");
  }
  Serial.println("Connected to MySQL server!");
}

void conectENV(){

  if (!bme.begin()) {
    Serial.println("Could not find a valid BME680 sensor, check wiring!");
    while (1);
  }else{
    Serial.println("BME680 successfully initialized!");
  }

  // Set up oversampling and filter initialization
  bme.setTemperatureOversampling(BME680_OS_8X);
  bme.setHumidityOversampling(BME680_OS_2X);
  bme.setPressureOversampling(BME680_OS_4X);
  bme.setIIRFilterSize(BME680_FILTER_SIZE_3);
  bme.setGasHeater(320, 150); // 320*C for 150 ms
}

void readENV(){
  // read all the sensor values
  if (! bme.performReading()) {
    Serial.println("Failed to perform reading :(");
    return;
  }
  // stores read values in temporary
  pressure    = bme.pressure / 100.0;
  temperature = bme.temperature;
  humidity    = bme.humidity;
  //illuminance = ENV.readIlluminance();
  //uva         = ENV.readUVA();
  //uvb         = ENV.readUVB();
  //uvIndex     = ENV.readUVIndex();
  gasResistance = bme.gas_resistance / 1000.0;
  altitud = bme.readAltitude(SEALEVELPRESSURE_HPA);
}

void printENV(){
  // print each of the sensor values
  Serial.print("Temperature = ");
  Serial.print(temperature);
  Serial.println(" *C");

  Serial.print("Pressure = ");
  Serial.print(pressure);
```

```
  Serial.println(" hPa");

  Serial.print("Humidity = ");
  Serial.print(humidity);
  Serial.println(" %");

  Serial.print("Gas = ");
  Serial.print(gasResistance);
  Serial.println(" KOhms");

  Serial.print("Approx. Altitude = ");
  Serial.print(altitud);
  Serial.println(" m");
}

void sendData() {
  // convert all read float values to string
  dtostrf(pressure,9,2,szpressure);
  dtostrf(temperature,9,2,sztemperature);
  dtostrf(humidity,9,2,szhumidity);
  //dtostrf(illuminance,9,2,szilluminance);
  //dtostrf(uva,9,2,szuva);
  //dtostrf(uvb,9,2,szuvb);
  //dtostrf(uvIndex,9,2,szuvIndex);
  dtostrf(gasResistance,9,2,szgasResistance);
  dtostrf(altitud,9,2,szaltitud);

  //update the query with the new values

  //TODO adapt this line according to the existing SQL table
  sprintf(query, INSERT_SQL, szpressure, sztemperature, szhumidity,
szgasResistance, szaltitud);

  // Initiate the query class instance
  MySQL_Cursor *cur_mem = new MySQL_Cursor(&conn);

  // Execute the query
  boolean res = cur_mem->execute(query);

  // Note: since there are no results, we do not need to read any data
  // Deleting the cursor also frees up memory used
  delete cur_mem;

  if(!res) Serial.println("Data successfuly sent!");
}

void getData(){
  row_values *row = NULL;
  long value = 0;
  // Initiate the query class instance
  MySQL_Cursor *cur_mem = new MySQL_Cursor(&conn);
  // Execute the query
  boolean res = cur_mem->execute(SELECT_SQL);
  Serial.println(res);

  // Fetch the columns (required) but we don't use them.
  column_names *columns = cur_mem->get_columns();

  // Read the rows
  do {
    row = cur_mem->get_next_row();
```

```
    if (row != NULL) {
      value = atol(row->values[0]);
      //Serial.println(value);
    }
  } while (row != NULL);
  // Deleting the cursor also frees up memory used
  delete cur_mem;
  Serial.println(value);
}

char *dtostrf (double val, signed char width, unsigned char prec, char
*sout) {
  char fmt[20];
  sprintf(fmt, "%%%d.%df", width, prec);
  sprintf(sout, fmt, val);
  return sout;
}
```

- **ARDUINO NANO 33 IOT - SENSITIVE DATA**

```
// variables to handle wifi connection
#define SECRET_SSID "network12345"                        // wifi network
SSID (name)
#define SECRET_PASS "<REDACTED>"                         // wifi network
password

// variables to handle MySQL server
#define SECRET_USERDB "username"                     // MySQL user login
username
#define SECRET_PASSDB "<REDACTED>"          // MySQL user login password
```

APPENDIX 3: REVIT COMPONENT PARAMETERS FOR FM

Data Type	Example
Purpose:	Facility Management
Actor	Facility Manager
• Object:	Sensors
• Geometric Information:	
• Detail	Simplified volume representation
• Dimensionality	3D
• Location	Relative
• Appearance	Not requested
• Parametric Behavior	Not requested
• Alphanumeric Information:	
• Element ID	(Ex: 257106)
• Asset Type	Movable
• Volume	Not requested
• Material	Not requested
• Uniclass2015Code	Pr_75_50_76_02
• Documentation:	
• Sensor Datasheet Webpage	https://web.archive.org/web/20190228002601/https://cdn-learn.adafruit.com/downloads/pdf/adafruit-bme680-humidity-temperature-barometic-pressure-voc-gas.pdf
• Board Information Webpage	https://web.archive.org/web/20210616144832/https://static.rapidonline.com/pdf/73-4863_v1.pdf