DEPARTMENT OF
COMPUTER SCIENCE

NOVA SCHOOL OF
SCIENCE & TECHNOLOGY

DIOGO SARAIVA CAVALHEIRO NEVES PEREIRA
Bachelor in Computer Science and Engineering

AN AUTOMATED SYSTEM FOR MONITORING AND CONTROL CLASSIC CARS' RESTORATIONS

AN IOT-BASED APPROACH

MASTER IN COMPUTER SCIENCE

NOVA University Lisbon
November, 2021

DEPARTMENT OF
COMPUTER SCIENCE

AN AUTOMATED SYSTEM FOR MONITORING AND CONTROL CLASSIC CARS' RESTORATIONS

AN IOT-BASED APPROACH

DIOGO SARAIVA CAVALHEIRO NEVES PEREIRA
Bachelor in Computer Science and Engineering

Adviser: Vasco Miguel Moreira do Amaral
Associate Professor, FCT, NOVA University of Lisbon

Co-adviser: Fernando Brito e Abreu
Associate Professor, ISTAR-Iscte, University Institute of Lisbon

Examination Committee:

Chair: Pedro Abílio Duarte de Medeiros
Associate Professor, FCT, NOVA University of Lisbon

Rapporteur: Paulo Jorge Fernandes Carreira
Associate Professor, IST, University of Lisbon

Adviser: Vasco Miguel Moreira do Amaral
Associate Professor, FCT, NOVA University of Lisbon

MASTER IN COMPUTER SCIENCE

NOVA University Lisbon
November, 2021

An automated system for monitoring and control classic cars' restorations

Acknowledgements

Without the support of both my supervisors, Prof. Vasco Amaral and Prof. Fernando Brito e Abreu it wouldn't have been possible to produce this work. Their support and constant availability allowed me to make progress tackling any problem that appeared along the way. I want therefore to thank them both for that and for, unknowingly, motivating me to do more and better every day during the development of this work.

The facilities, the professors and the whole atmosphere at FCT-NOVA made it possible to achieve this goal in my life. I want to thank the NOVA University, especially the Computer Science Department and all the professors I met because everything and everyone contributed to my growth as a person.

I would also like to thank my colleague Diogo Lívio, whose dissertation, together with mine, addresses solutions to the problems presented by the professors.

The constant availability of all the workers in Raimundo Branco's workshop, including himself and all the collaborators, allowed the data collection process to run smoothly over the months, for which I owe all of them a profound thank you.

I would also like to thank João Sousa of Vitruvius FabLab, located in ISCTE-IUL, for the development of the 3D model presented in this work.

I want to leave a special thanks to my family for supporting me in the most challenging moments and for always being supportive. Without their support, I would not have been able to reach this milestone. I would also like to thank my grandmother Amélia for all the support, for being the greatest grandmother and for always pushing me to do better and go further.

During my progress at FCT-NOVA, I had the opportunity to meet people who always motivated me in the most difficult times. With them, this arduous path became less distressing and more productive.

I am lucky to have friends that I have known since seventh grade, I also owe a big thank you to them for being there for me and for keeping the friendship alive, it is one of the greatest prides I have.

Abstract

The lack of information during car restorations is incompatible with the digital transformation currently taking place worldwide, Industry 4.0. In the context of classic vehicles, when dealing with values in the order of hundreds of thousands, or even millions, it is important to ensure the authenticity of the vehicle after restoration and to maintain a constant connection with the owners, especially in situations where vehicles are sent abroad to be restored.

The authenticity of classic cars is done through a certification that ensures the quality and legitimacy of the restoration process, which can be time-consuming and tedious for both the certifying authority and the workshop workers. Our goal is then to facilitate this process by automatically identifying the different restorations performed on classic cars and, at the same time, to share this information with the owners, keeping them aware of any intervention made on their cars.

In this work, we present a prototype that allows the identification of restoration processes, an internal location system for the workshop, a system in charge of transforming data from the prototype into restoration events and a web application that serves as an interface for the users of the system, all supported by an Internet of Things (IoT) system designed to meet its requirements.

The knowledge obtained through bibliographic research, as well as the structuring of the development plan for this solution, allowed for greater organisation and better decision making, crucial steps for the success of this project that aims, in essence, to digitally transform a successful car restoration workshop.

Keywords: Internet of Things, Sensors, Industrial Internet of Things, Data processing, Classic Cars Restoration, Industry 4.0

Resumo

A falta de informação existente durante o restauro de automóveis clássicos é incompatível com a transformação digital atualmente em curso em todo o mundo, a Indústria 4.0. No contexto dos veículos clássicos, quando se trata de valores da ordem das centenas de milhares, ou mesmo milhões, é importante assegurar a autenticidade do veículo após o restauro e manter uma ligação constante com os proprietários, especialmente nas situações em que os veículos são enviados para o estrangeiro para serem restaurados.

A autenticidade dos automóveis clássicos é feita através de uma certificação que assegura a qualidade e legitimidade do restauro, um processo que pode ser demorado e enfadonho tanto para a autoridade certificadora como para os trabalhadores da oficina. O nosso objectivo é então, facilitar este processo através da identificação automática das diferentes reparações efectuadas em automóveis clássicos e, ao mesmo tempo, partilhar esta informação com os proprietários, mantendo-os conscientes de qualquer intervenção feita nos seus automóveis.

Neste trabalho, apresentamos um protótipo que nos ajuda na identificação dos processos de restauro, um sistema de localização interna para a oficina, um sistema encarregue de converter dados do protótipo para eventos de restauro e uma aplicação web que serve de interface para os utilizadores do sistema, tudo isto apoiado por um sistema IoT concebido para satisfazer os seus requisitos.

Os conhecimentos obtidos através da investigação bibliográfica, bem como a estruturação do plano de desenvolvimento desta solução, permitiram uma maior organização e uma melhor tomada de decisões, passos cruciais para o sucesso deste projecto que visa, na sua essência, transformar digitalmente uma oficina de restauro de automóveis com sucesso.

Palavras-chave: Internet das Coisas, Sensores, Internet Industrial das Coisas, Processamento de dados, Restauro de automóveis clássicos, Indústria 4.0

Contents

List of Figures

List of Tables

List of Listings

Acronyms

ACM	Association for Computing Machinery xvii, 1, 3, 4, 93
ACP	Automóvel Club de Portugal 2
AI	Artificial Intelligence 18, 20
AP	Access Point 13
API	Application Programming Interface 26, 27, 41, 46, 47, 48, 51, 53, 78, 113
ASE	Automated Software Engineering 2
AWS	Amazon Web Services 5, 22, 25, 26, 40, 41, 42, 43, 47, 51, 52, 55, 103
BLE	Bluetooth Low Energy xix, 6, 12, 13, 34, 38, 39, 57, 77, 129, 130, 131
BPMN	Business Process Model and Notation 51
CCS	Computational Classification System 3
CRM	Customer Relationship Management 18
DFT	Discrete Fourier Transform 15
DSP	Digital Signal Processing 14
DSR	Design Science Research 5, 22
ERP	Enterprise Resource Planning 52
FFT	Fast Fourier Transform 15, 35, 56, 63, 69, 74, 75, 104
FIR	Finite Impulse Response 15
Gb	Gigabyte 101
GPS	Global Positioning System 13, 38, 40
HTTPS	Hypertext Transfer Protocol Secure 42, 43
IaaS	Infrastructure-as-a-Service 11, 21
IaC	Infrastructure as Code 43

IEEE	Institute of Electrical and Electronics Engineers 24, 92
IFML	Interaction Flow Modelling Language xviii, 48, 125
IIC	Inter-Integrated Circuit 33, 35
IIoT	Industrial Internet of Things 9, 10, 20
IIR	Infinite Impulse Response 15
IoT	Internet of Things ix, xi, 2, 3, 4, 5, 6, 9, 10, 11, 12, 13, 17, 18, 19, 21, 22, 23, 24, 25, 26, 27, 28, 29, 34, 40, 41, 42, 43, 47, 51, 52, 55, 82, 92, 93, 100, 103
IP	Internet Protocol 51
IT	Information Technology 10, 11, 21
LoRaWAN	Long Range Wide Area Network 42, 43
LTE	Long-Term Evolution 12
MES	Manufacturing Execution Systems 17
ML	Machine Learning 11, 20, 40, 44
MQTT	Message Queuing Telemetry Transport 12, 20, 42, 43
OMG	Object Management Group 48
PaaS	Platform-as-a-Service 11
PCB	Printed Circuit Board 28
PCS	Process Control System 4
RAM	Random-access memory 56, 101
RFID	Radio-frequency Identification 10, 12, 13
RSSI	Received Signal Strength Indicator 13, 39, 40, 105
SaaS	Software-as-a-Service 11
SAM	Serverless Application Model 43, 47
SDK	Software Development Kit 28, 42
SIG	Special Interest Groups 92, 93
SIGDOC	Special Interest Group on Design of Communication 93
SoCs	System on a Chip 13
SPI	Serial Peripheral Interface 35
SUS	System Usability Scale xviii, xix, 93, 94, 95, 96, 136, 137, 138
TLX	Task Load Index xix, 93, 94, 95, 139, 140
WSN	Wireless Sensor Network 20
WSS	WebSockets 42

CHAPTER 1

Introduction

This chapter introduces the context and description of the dissertation, the institutional context underlying it, explaining its challenges, the problem and the final objectives it aims to achieve. Then, we present the classification of this work following the standard computing classification system from ACM accompanied by the guideline used in this work's development, describing each step. Finally, we present the methodology followed while developing this work, list the expected results, and its structure.

1.1 Context and Description

In a world increasingly divided between petrol and electric cars, classic automobiles bring a breath of fresh air when it comes to admiration and recognition of a true work of art. For a few, it may just be a family car well kept over the years that brings pride but for others is an investment, nevertheless, for both these people, that own classic cars, it is important to know restoration workshops they can trust their cars with.

Over the years, several vintage classic cars have been sold in auctions for large amounts of money that got up to 70 million dollars (over 61.8 million euros)[2]. Price tags like these can decrease with slight scratches, defects in the paint job which can devalue the car in tens of thousands of dollars, if not more.

In this era of easily available information, it seemed unreasonable for car owners to be unable to follow their cars' restoration process, especially in cases when these are shipped overseas. This information would be useful to the owners, the workshop and institutions in charge of assessing the car's value after restoration. This is precisely what this work aims to achieve.

1.2 Institutional Context

The research depicted in this work is the result of a partnership between the Automated Software Engineering (ASE) group from NOVA-LINCS [1], ISCTE-IUL [2] and the workshop where this system is expected to be implemented, Raimundo Branco [3], our case study. There is also a collaboration with the largest automobile association in Portugal, Automóvel Club de Portugal (ACP) [4], and BASF [5], the largest chemical producer in the world, with a strong impact in the production of specific paints used in classic car restorations.

1.3 Challenges

Automating the restoration process of a car to give feedback to the owner and workshop workers can quickly become a very complex system simply because there are a lot of restoration processes involved. However, there are repairs common to all types of cars: structural or body restoration, sanding, painting, and polishing.

Even though these processes provide significant information, each have their own sub-processes that could provide more fine-grained detail about the restoration process. These sub-processes are detected by our system through a small portable box with sensors, that accompanies the car through all these different stages.

The correct choice of the sensors as well as the architecture to support all the data being collected are crucial and since this solution has to be applied in a real environment, the protection of all data is a priority.

The solution needs to support the reality of a workshop where several cars are under restoration at the same time which is why the amount of data generated can become overwhelming. This can easily become a challenge for both the network and the processing unit especially if the chosen solution processes the data locally rather than in the cloud.

Another common problem in IoT is power consumption, since our box will need to accompany the car through the workshop it has to be connected to a limited power supply. The magnitude of the workshop can cause intermittent connections and make the process of locating the boxes inside difficult, thus increasing the hurdles in implementing this system.

1.4 Problem Statement and Final Goals

With all the challenges described in section 1.3 this work aims to answer the following question:

[1] https://nova-lincs.di.fct.unl.pt Last accessed on: 27/11/2021
[2] https://www.iscte-iul.pt Last accessed on: 27/11/2021
[3] https://www.facebook.com/raimundo.joaquim.branco Last accessed on: 27/11/2021
[4] https://www.acp.pt/inicio Last accessed on: 27/11/2021
[5] https://www.basf.com/pt/pt.html Last accessed on: 27/11/2021

Is it possible to create a system to correctly identify the different stages of classic cars restorations and at the same time provide this information to multiple users for restoration, monitoring, documentation and certification purposes?

This research question can be broken down into several objectives that this dissertation aims to achieve:

- Design and development of a robust and reliable IoT device prototype responsible for data collection and pre-processing;
- Design and development of a scalable IoT system for the developed IoT prototypes;
- Implementation of the algorithm responsible for identifying restoration processes;
- Design and development of a web application to manage the IoT system and its devices;
- Plan and test the system using typical use case scenarios.

1.5 ACM Classification

The ACM [6] Computational Classification System (CCS) [7] was created to provide a standard hierarchical organisation system for different computing subjects. In this way, it becomes possible to properly index and categorise this work, helping readers to achieve a better understanding of all the components comprised in it. The implemented solution is composed of multiple subsystems that fall into different categories these being: cross-computing tools and techniques, sensor networks, communication hardware, interfaces and storage, signal processing systems, hardware validation, information systems applications and process control systems.

The cross-computing tools and techniques category is present since all its subcategories were used in the development of this work, such as verification and validation of the system as a whole discussed in chapter 5. Performance, reliability, and cost are also important topics when choosing an IoT platform, as is the experimentation done in preparation for this dissertation that helped us decide which IoT platform was best suited for our problem.

The system can be included in the embedded systems category because one of its components is the sensor box that can be considered as a sensor network since these subsystems consist of "spatially distributed devices that communicate via wireless radio and cooperate in the detection of physical or environmental conditions" while providing a "very complete degree of visibility of physical environmental processes" [9].

Although their main category is cyber-physical and cyber-physical systems, this work cannot be considered cyber-physical, since these require some kind of physical interaction

[6]https://www.acm.org Last accessed on: 27/11/2021
[7]https://dl.acm.org/ccs Last accessed on: 27/11/2021

with the environment, turning on or off an air conditioner, for instance. This kind of control does not happen in our system, the sensor box will sense the environment and will not change it in any way [29].

Hardware is one of the main components of IoT systems because it allows users to build their own detection devices, without which it would be impossible for the system to collect information. The sensor box can also be considered a signal processing system because it processes input signals from the environment and produces output signals in the form of values that the sensors are detecting [33].

As expected this type of system requires a special type of validation, most of the time there is only software to be validated, but for this system to be valid, its hardware must be validated as well, as explained in the evaluation chapter 5. “A Process Control System (PCS) includes a data collection and distribution system that collects and stores data from various data sources, each of which can use its way of acquiring or generating data in the first place“ [12], which is what this system is intended to do, fitting directly into the PCS category.

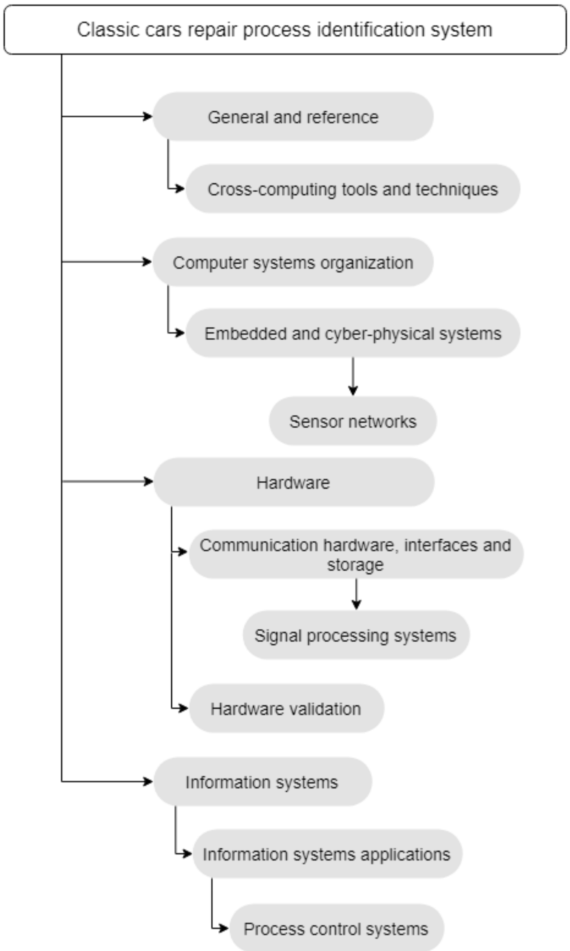


Figure 1.1: Restoration process identification system ACM classification.

1.6 Scientific Methodology

This work followed a scientific research methodology known as Design Science Research (DSR) that offers guidelines for the evaluation and iteration of research projects. This approach was used to help make a better-founded decision of which IoT platform to choose via the construction of artefacts, simplified versions of the sensor boxes. DSR methodology claims that the development of artefacts is also part of the research which enabled us to test and evaluate different IoT platforms greatly improving the knowledge extracted [37].

This particular research method focuses mainly on two topics: the creation of new knowledge and the analysis of the use or performance of an artefact. These artefacts are the result of a scientific research approach to the design of a problem and can range from constructs (the conceptual vocabulary of a problem/solution domain) to complete instantiations (the realisation of constructs in a given environment, which can be interpreted as a simplified version of the type of prototypes we intend to build). For this case, two simple IoT projects that sense temperatures in an environment have been developed and each of them is based on a specific IoT platform, these being Amazon Web Services (AWS) and Azure. Having a common denominator, the physical aspect (the same temperature sensor, board and internet connection), helped us evaluate the different features of each of these platforms and choose the most appropriate one.

Awareness of problem: This problem emerged due to a total reconstruction of the workshop. Classic cars are not only expensive but also special for their owners and to mitigate the lack of restoration information owners receive, a digital transformation of the workshop was initially presented. This idea ended up leading to two master thesis in which the main focus was to solve this problem by dividing it into two domains, one that focuses on data collection, identifying the stages of car restoration, and the other that focuses on the creation of a management system service to serve car owners and the country's automobile association. The integration of these two systems is further described in section 4.8.

Suggestion: There are multiple ways to execute a solution to the proposed problem and many ideas arrived and were discussed during the meetings, but it was clear from the beginning that the sensor boxes would be connected to the IoT platform to update the system and the status of the cars under restoration.

Development: Although this methodology brings great results through the construction of artefacts, another type of research was done as a complement, a theoretical approach that required reading the documentation of the IoT platforms to have better insights into each one's capabilities, the combination of these is our results from this methodology.

Evaluation: After development, the evaluation of these prototypes was mainly based on the features provided by the IoT platforms and how they were perceived in the prototypes, as well as the available support and documentation.

Conclusion: The conclusion we got from this methodology research is the most appropriate IoT platform based on our needs, using the theoretical comparison and the experience while developing the artefacts.

1.7 Expected Key Contributions

Throughout this work, the progress made allows us to expect the following contributions:

- Configurable prototypes used to record, pre-process and send data to an IoT platform;
- An indoor location system based on BLE beacons;
- An IoT platform to support the IoT devices;
- An algorithm capable of identifying the different restoration processes from the data recorded by the prototypes;
- A web application to serve as a monitoring and control station of the prototypes.

1.8 Document Structure

This document is organised in the following way:

- **Chapter 1 - Introduction:** this chapter is the presentation of the problem, its correct indexing and categorisation, and introduces the various components that together make up the solution.
- **Chapter 2 - Background:** a brief overview of the main concepts in IoT, its architectures and information about IoT platforms, indoor location techniques, cloud computing platforms and the Raspberry Pi.
- **Chapter 3 - Related Work:** In this chapter, we present some similar solutions implemented by some companies, as well as useful articles for the development of certain components of our solution.
- **Chapter 4 - Solution Overview:** this chapter presents the solution itself, the technical decisions taken during the development of the prototype, the IoT system and the web application. It includes the architectures for these three components and the reasoning behind both the architectures and the technical decisions.
- **Chapter 5 - Data Collection, Test and Validation:** In this chapter we introduce how the evaluation plan was carried out, present the data collection techniques and the results. It also contains the validation process of the web application.

- **Chapter 6 - Conclusions:** we present our thoughts and conclusions about this work summarising the achievements of this dissertation. This chapter also presents this work's future work.

CHAPTER 2

Background

This chapter gives an insight into essential topics such as Internet of Things (IoT) and its architecture, some of the biggest cloud providers and presents their comparison as well as key IoT technologies necessary for the development and understanding of this work. After we give an introduction about the topic of indoor location we present different possibilities for achieving this. Finally, we give some insights on digital signal processing and a data processing technique used during the development of this dissertation.

2.1 Internet of Things

The IoT can be defined as "the network of things, with clear element identification, embedded with software intelligence, sensors, and ubiquitous connectivity to the Internet" [30], creating a digital network composed of multiple components, each with its purpose. There has been an evolution in the IoT definition caused by the multiple technologies that were (and still are) converging with the IoT.

These things produce a lot of information and it is increasingly important to provide accessibility and infrastructure to process it, now more than ever because more and more companies are integrating IoT into their business models, commonly known as the Industrial Internet of Things (IIoT), which are tailor-made systems to help ensure flexibility and manufacturing in industrial processes. These help ensure future global competitiveness, increase revenues and obtain overall better productivity [16]. IIoT systems are considered to be an extension of IoT converting regular factories into smart factories connecting people and machines in real-time.

This work resonates more as an IIoT system than an IoT one since the real value of the IIoT is the "availability of ubiquitous information and consequently, the decisions that can be made from it", the restoration steps in our case [11]. IoT and IIoT are in the centre of the fourth industrial revolution, Industry 4.0 — a set of practices to enable machine-to-machine communication, self-monitoring, analysis and diagnosis resulting in smart factories. Throughout this work, the term IoT is used instead of IIoT since it is the foundation of any IIoT system.

2.1.1 Architectures

Like in IIoT there is no consensus on the choice of an IoT architecture but there are some unique aspects of these types of systems. One of the main reasons for the lack of architectures is the freedom that developers must have to build their solutions, the IoT "integrates multiple wired and wireless communication, control, and Information Technology (IT) technologies, which connect various terminals or subsystems" [39], finding a common architecture for all types of use cases in all industries can be challenging.

However, any architecture for the IoT needs to meet all the requirements mentioned below:

- Scalability
- Interoperability (to connect multiple different devices)
- Availability
- Reliability
- Flexibility (to respond to stakeholders changes)

Although there was no specific architecture, researchers have designed their own and even developed systems to test such architectures like Zhang [40] that focused on the basic components of an IoT system. Despite the scarce literature in the field, the most common architecture proposed is presented as layers to describe functionality [39] such as the Five-Layer Architectures (business, application, processing, transport and perception layers).

The Five-Layer Architecture provides a high level of detail and encompasses the:

- **Perception Layer** - corresponds to the physical layer that contains the sensors. Its main function is to sense the environment and other smart objects.
- **Transport Layer** - responsible for transferring data between the Perception and the Processing layers and vice versa via 3G, Bluetooth, Radio-frequency Identification (RFID) or Wi-Fi.
- **Processing Layer** - also known as the middleware layer. It stores, analyses and processes the data received by the Transport layer. Sometimes offers services to the layer below, secures the connectivity to the cloud and provides edge data analytics or fog computing.

- **Application Layer** - this layer is responsible for delivering application-specific services to the user.

- **Business Layer** - this final layer is responsible for managing the whole IoT system (devices, users, data). It can also build graphs, flow charts based on data received from the Application layer.

2.1.2 Cloud Providers

Cloud computing is part of a broader concept of virtualisation of information technology, meaning that it provides "useful function while hiding how their internals work" [38], however, for computing to be virtualised, it is necessary for processing, storage, data and software resources to be distributed.

One of the main purposes of cloud computing is to provide large amounts of computing power in a fully virtualised and sometimes even reactive manner, making consumers pay to providers based on the usage of such resources [38]. Over the years, cloud computing has been providing "a paradigm shift of business and IT, where computing power, data storage, and services are outsourced to third parties and made available as commodities to enterprises and customers" [1]. This paradigm shift enables some of the most disruptive advancements in recent technology like automation, Machine Learning (ML), IoT and Big Data.

Some of the best known commercial providers are AWS [1], Google Cloud [2] and Azure [3], each with their own strengths and defining characteristics, whilst providing on-demand cloud computing services like Infrastructure-as-a-Service (IaaS), Platform-as-a-Service (PaaS) and Software-as-a-Service (SaaS). Some options stand out as being open source, such as Cloud Foundry [4] and WSO2 [5], however, these vendors sometimes do not provide a broad set of tools to build more detailed solutions. Although, because they are open source, they allow integration with other solutions more extensively than most commercial vendors.

2.1.3 IoT and the Cloud

There are multiple cloud providers with key IoT technologies and concepts already embedded, while others focus on processing and extracting knowledge from data choosing to provide additional integrations to extend a general application to an IoT system. In this section, we present some key concepts used in the IoT that will later help explain the rationale behind some architectural and technological decisions.

[1] https://aws.amazon.com/ Last accessed on: 27/11/2021
[2] https://cloud.google.com/ Last accessed on: 27/11/2021
[3] https://azure.microsoft.com/ Last accessed on: 27/11/2021
[4] https://www.cloudfoundry.org/ Last accessed on: 27/11/2021
[5] https://wso2.com/ Last accessed on: 27/11/2021

Edge and Fog Computing - With over 75 billion IoT devices expected to be active by 2025 [32], the need to find a way to efficiently process all the data produced by these devices has quickly become a priority. Edge and fog computing have been put forward as possible solutions, moving processor and battery-intensive computations to the network layers above, which most often have access to an unlimited power supply and greater processing power.

The highest network layer in this context would be the cloud data centres that the IoT platform is relying on, although these centres have the ability to process data quickly, they are often far away, which leads to higher latencies, higher loads on the data centres and opens the door to possible network congestion, hence the need to bring some of this cloud computation to the edge of the network, closer to the user. This means that it has become possible to process, analyse and filter the data collected by IoT devices before it reaches the cloud, sometimes even avoiding the need to process it in the cloud. This mechanism is the interconnection of cloud and fog computing that allows IoT applications to achieve better performance.

This type of computing not only has latency and cost reduction benefits, but it also increases system security as data is now analysed locally and protected by the local network or a service provider's closed system, making solutions more reliable and providing faster scalability [8].

Communication Channels and Messaging Protocols - IoT devices need to transmit data they perceive from the environment to fog nodes or directly to the cloud and sometimes they even need to communicate with each other, an attribute only possible due to the existence of communication channels and messaging protocols. These channels and protocols have different characteristics and the most appropriate one should be chosen depending on the IoT device, its function and its users. Given the limited power supply that most devices have, some communication channels have lower bandwidth and range, while others prioritise range and sacrifice bandwidth.

Some of the most common are RFID, Long-Term Evolution (LTE), ethernet, Wi-Fi, Bluetooth, Zigbee or Bluetooth Low Energy (BLE). BLE are specific to limited power use cases sacrificing bandwidth to achieve better range than regular Bluetooth. The layer in charge of establishing a communication channel is the network layer presented in the aforementioned 5-Layer architecture subsection 2.1.1.

Messaging protocols also have different characteristics, since IoT devices have limited processing power and battery. One of the most common protocols is Message Queuing Telemetry Transport (MQTT) [6], it follows the publish-subscribe standard, is very light (in terms of communication overhead compared to other protocols) and is often used for low bandwidth connections.

[6] https://mqtt.org/ Last accessed on: 27/11/2021

2.1.4 Indoor Location

Indoor location technologies have been developed over the years, however, there is not yet a standard device or communication channel to achieve this, mainly because of the problem heterogeneity, forcing the industry to use technologies like BLE, Wi-Fi or RFID. BLE beacons are capable of solving indoor location problems given their high versatility, configuration options and range while providing good accuracy.

Indoor location using Wi-Fi means tracking the IoT through several Access Point (AP) when the device exits the range of one AP and enters in the range of another one. This approach requires very good Wi-Fi coverage, which can be difficult if the premises are medium to large. Even though IoT devices usually require an internet connection they can still perform without an active one, via Edge Computing and local storage, requiring the companies or factories to invest large amounts of money for a Wi-Fi coverage to build a solution that has lower accuracy than BLE beacons.

RFID however can perform well, despite it providing selective location and not area-wide location. Typical RFID solutions most of the time require a human in the loop to explicitly put the device in the RFID area given its low range. In these types of communication protocols, one can use the Received Signal Strength Indicator (RSSI) of a signal to resonate about the distance between a device and the source of the signal received. However, a study [10] showed that this type of indoor location is very volatile and can lead to poor insights for cases where high accuracy is required.

Another solution to indoor location is to use devices from companies like Estimote [7] which offers a broad variety of BLE proximity and location beacons and a Global Positioning System (GPS) beacon for outdoors accurate positioning. Kontakt [8] has an even broader offer of gateways, beacons (powered by BLE) and tags, all configurable via a web-based device management centre. Nordic Semiconductor [9] sells System on a Chip (SoCs) mainly for indoor localization using Bluetooth, BLE, Zigbee and Wi-Fi in the same chip offering multiple wireless protocols in a small form factor.

2.2 Signal Processing

A signal is observed data that represents a physical phenomenon as simple as pressure or temperature variation in a room. To study the physical phenomenon, it often needs to be translated by some kind of transducer into an electrical equivalent to be then displayed on an oscilloscope. Signals have 4 subsets: analog (or continuous), digital (or discrete), sampled (continuous in values and discrete in time) and quantized (discrete in values, continuous in time).

Moreover, signals can be deterministic or non-deterministic, the difference being the ability to predict the behaviour of such signals. Non-deterministic signals, often described

[7] https://estimote.com/ Last accessed on: 27/11/2021
[8] https://kontakt.io/ Last accessed on: 27/11/2021
[9] https://www.nordicsemi.com/ Last accessed on: 27/11/2021

as random signals or time series, usually require probabilistic and statistical methods to analyse them, a procedure called time series analysis.

The behaviour of deterministic signals, depending on the measurement, can be predicted because their waveform repeats exactly at regular time intervals, like a machine polishing a car at a constant speed for instance. Such measurements, after performing digital signal processing, result in a fundamental frequency at the lowest frequency and several harmonics.

All these types of signals only become relevant when it becomes possible to analyse and extract data from them, hence the existence of the field of signal processing [31].

Signal processing is “a branch of electrical engineering that models and analyzes data representations of physical events“ that enables us to use computers, mobile phones, smart devices and other electronic devices that underpin our digital lives [15].

As Shin et al. point out [31], signal processing encompasses multiple procedures used to reveal information from the data recorded. There are 7 categories to signal processing:

- Analogue signal processing
- Continuous-time signal processing
- Discrete-time signal processing
- Digital signal processing
- Nonlinear signal processing
- Statistical signal processing

Most devices today are digital and digital signal processing is the field that processes the discrete sampling signals from these devices.

2.2.1 Digital Signal Processing

Digital Signal Processing (DSP) is the field of analysis and data processing of a digital signal, however, if the signal is analogue there is a need to perform an analogue-to-digital conversion which is the first of three steps required to extract information from a signal. As mentioned above, almost all recent devices already produce digital signals, avoiding the need to perform the acquisition step, leaving only the processing and interpretation steps [31].

“The wide availability of software to carry out digital signal processing (DSP) with such ease now pervades all areas of science, engineering, medicine, and beyond“ [31] allowing to perform digital signal processing in small computers like the Raspberry Pi.

The field of DSP encompasses several areas and is composed of two main concepts: digital filtering and spectrum analysis. Digital filtering is considerably one of the most important aspects when performing DSP because signals often come with noise and this helps, through a set of mathematical operations on a sample of discrete-timed signal, reduce such noise by applying Finite Impulse Response (FIR) or Infinite Impulse Response (IIR) filters [28]. Spectrum analysis contribution is the "examination of the frequency content in a signal" which "is often useful when trying to understand what physical components are contributing to a signal" [36].

2.2.2 Fast Fourier Transform

The examination made via the spectrum analysis can be performed by a Discrete Fourier Transform (DFT) or by using statistical techniques when there is randomisation in the signals. A 1965 Cooley-Tukey paper [10] introduced a series of techniques later known as the Fast Fourier Transform (FFT), reducing "by one to two orders of magnitude" the DFT computing time [28], which works by computing the DFT over small subsets of the full sample and then combining the results. Several libraries like NumPy [11] perfected the FFT algorithm, decreasing the computation time using matrix operations to calculate the transforms simultaneously.

Despite the contribution from Cooley-Tukey, there are several variations of the same FFT algorithm widely available across multiple libraries. In its essence the FFT algorithm is composed of a set of algorithms developed to change the domain of a signal from time to frequency decomposing, for instance, a 10 second recorded signal from an electric sander into (one or more) fundamental frequencies and their harmonics.

[10] https://www.ams.org/journals/mcom/1965-19-090/S0025-5718-1965-0178586-1/S0025-5718-1965-0178586-1.pdf Last accessed on: 27/11/2021

[11] https://numpy.org/ Last accessed on: 27/11/2021

CHAPTER 3

Related Work

In this chapter, we will introduce systems and studies that have techniques or objectives in common with our solution accompanied by a comparison between them. The following is another comparison across various IoT platforms and the reasoning behind the platform chosen.

3.1 MES, Studies and Platforms Comparison

Manufacturing Execution Systems (MES) providers offer centralised solutions specifically for manufacturing companies to monitor, control and perform data analysis from their machines allowing them to achieve better overall productivity. We decided to exclude these systems from being possible contenders because of their high specificity (manufacturing companies profit the most with this type of systems, not the case of the workshop upon which the proposed solution is based) and the fact that they are centralised, making it impossible to build solutions tailored to our needs.

Syafrudin et al.[34] presented a real-time monitoring system for an automotive manufacturing assembly line. It uses various sensors and given the amount of data produced, a complete big data processing platform was also developed. Furthermore, a hybrid prediction model was used to extract outliers from the sensors data and to prevent faults during the manufacturing process. This study's solution, like many others found in the literature, was built on top of technologies like Apache Kafka [1], Apache Storm [2] and MongoDB [3]. Kafka is a scalable messaging queue system capable of dealing with high volumes of real-time data while Storm was developed to process high-velocity streams of data, fed by Kafka. MongoDB was used to store large amounts of data efficiently.

[1] https://kafka.apache.org/ Last accessed on: 27/11/2021

[2] https://storm.apache.org/ Last accessed on: 27/11/2021

[3] https://www.mongodb.com/ Last accessed on: 27/11/2021

This stack is widely used given its capability to process big data from a high number of sensors in almost real-time. In this case, machine learning was also used to prevent manufacturing losses during the process and improve overall performance.

Multiple cloud-based shop management software programs and platforms are available in today's market enabling customers (workshop managers and workers) to get a hold of their business. Some examples of such programs are: Shopmonkey [4], Fullbay [5] and Shop-Ware [6] and these provide advantages to their customers like built-in communication tools to contact and manage car owners via a Customer Relationship Management (CRM), workflow management for repair tracking, repair scheduling, repair documentation to facilitate inspections, integrated payments processing, team managing and tracking, inventory management, real-time dashboards and reports on profitability, fleet performance and technician efficiency. Fullbay stands out from the competition by offering multi-shop tools for easy aggregation and management of data across multiple locations.

Shop-ware also comes with a portal that creates a direct connection between technicians and car owners where the latter can submit repair requests, prioritise them, track present and past jobs made in the workshop, facilitate payments and invoices. All three have built-in notification tools to notify their customers. It is clear that these programs offer a lot to workshop managers and their customers, however, they lack an important feature, the core of our system, the automatic restoration processes identification.

These platforms have features such as inventory and employee management, integrated payment processing offering more than our proposed solution, however, this study aims is to facilitate restoration stages identification and customer interaction, heavily focusing on the former whilst providing these features at a cheaper cost than these companies.

3.2 Industrial Internet of Things Systems

The combination of technological advances of the internet with intelligent devices has allowed the advanced digitalisation of factories, in some cases, this new paradigm has completely changed industrial production. This change that has occurred in recent years is called Industry 4.0 and is the most recent industrial revolution after the rapid adoption of digitalisation in the industry. The new concept goes beyond digitisation and implies the optimisation of the manufacturing process through the use of technologies such as IoT, big data and Artificial Intelligence (AI). By improving the manufacturing process and keeping costs down, companies can achieve higher profit margins just by collecting and processing data from their manufacturing operation.

This term was initially introduced by the German government to optimise the industry in their country, passing the duty of research and investigation to the Ministry of

[4]https://www.shopmonkey.io Last accessed on: 27/11/2021
[5]https://www.fullbay.com Last accessed on: 27/11/2021
[6]https://www.shop-ware.com Last accessed on: 27/11/2021

Education and Research. In addition to the technologies that are regularly used in Industry 4.0 projects that were presented above, it is common to see increased mechanisation and automation, as well as even more pervasive digitisation by means of sensors and intelligent devices or even, in some cases, simulation[17].

These new technologies and approaches allowed companies to attain value never reached before and one of the most common case of the application of this new revolution is by developing cyber-physical systems, the merge of physical and digital levels. Preventive maintenance can save an entire operation, factories can save time by building a cyber-physical preventive maintenance system composed of sensors and "smart" devices alongside an artificial intelligence trained model to prevent breakdown by simply analysing the vibrations picked up by accelerometer sensors on these IoT devices[17].

A study from 2016 [25] showed that 37% of German companies implemented some Industry 4.0 technologies, when only a third of those made purchasing adjustments to sustain the new technologies meaning that, depending on the size of the company the value extracted by using these new technologies to analyse the manufacturing process can be quite rewarding and not require a vast investment upfront.

The successful use of these technologies propelled the industry and the research community to build innovative solutions. In Morocco, a University of Rabat research laboratory developed a low-cost urban air pollution monitoring system [13] that tracked several air pollutants like carbon dioxide and monoxide, sulphur dioxide and others in seven highly concentrated neighbourhoods reducing the life expectancy of its inhabitants and increasing the probability of future respiratory problems, mainly seen already on children. Several devices, called sensing nodes were built and spread near open street food vendors, thrift shops and traditional baths over these seven neighbourhoods as these activities are common and known to affect negatively the air quality.

The sensing nodes were composed by a Raspberry Pi as an easy-to-use, reliable computing unit, a GPS tracking device to estimate the node location, a USB 3G/4G modem to transmit data in real-time and a 20000 mAh (milliAmp hours) as well as a solar panel to power this device. Each node had a sensor unit as well with a temperature sensor, a humidity sensor and an air quality measurement sensor, used to get dust particles and smoke concentration in the air. Besides data collection the nodes were also responsible for some pre-processing and encapsulation of data, justifying the use of the Raspberry Pi, a more powerful and capable device than most microcontrollers.

The pre-processed data was sent over 3G/4G (by using SIM cards) to the IoT system that processed the data and stored it in a long term data archive. Once the system had an extensive dataset, the team started using machine learning to predict the future state of the pollution in those areas based on the data gathered by the sensors with also traffic and meteorological features that occurred in the same time frame.

The results showed high levels of pollutants in crowded areas in these neighbourhoods and the machine learning component allowed them to estimate when the peak pollution time is as well as its location taking into account the traffic data.

The next example has a more straightforward relationship with factories and companies since it was developed to mitigate the leakage of gas emissions in hazardous sites, these being mainly solvent or chemical factories and companies that buy these types of products [23]. The problem to solve was the atmosphere contamination caused by certain gases which may increase the risk to human health. There are several portable hand tools to detect these types of situations, however, they are costly and cannot detect some pollutants leakage because their odour threshold is higher than required, this happens mainly in oil and gas activities where the non-detection of gas leaks can lead to serious consequences. Therefore, the group implemented an end-to-end, distributed monitoring system using several detectors “capable of performing real-time detection of gas emissions at potentially hazardous sites at a per-minute data rate“ [23].

This solution also has a set of sensing nodes but is connected by a Wireless Sensor Network (WSN) infrastructure. Each node has weather-climatic sensors, particle detecting sensors and communicates directly with the remote server over mobile networks. This system was installed in a petrochemical plant near Montova, Italy and a similar one was installed in a refinery located in Sicily. For the latter, the system was slightly different and was composed of 11 nodes with temperature, humidity sensors and also an anemometer to measure the wind speed and direction. Each one of these nodes is connected to several smaller wireless nodes responsible for the detection of gas leaks. The communication between the two types of nodes is made wireless with a MQTT like routing communication layer. These two systems were successful while being cost-effective, flexible and capable of detecting abnormal gas values in the atmosphere [23].

Another interesting and useful application of IIoT is in the maintenance of renewable energy. Zhang et al. [40] proposed a monitoring system for wind farms, since its maintenance can be cumbersome and expensive, more efficiently maintained wind farms can produce more energy and reduce or even avoid expensive downtime which can affect the consumer electricity cost. Once again, “smart“ devices can be used to gather data from the spinning turbines and with the help of AI predict when it needs to be serviced or even when a specific bearing needs to be replaced.

This proposed analysis requires a set of acceleration, electrical, strain, temperature and oil particle counter sensors that combined can assess the overall status of a wind turbine. The data provided by some sensors needs to be pre-processed to convert from time-domain to frequency-domain, like the accelerometer data, then the frequency values can be joined with all other sensors data and displayed in a real-time web and mobile monitoring application.

The then processed data can be sent to a trained ML model to infer possible parts replacements or schedule maintenance visits. The same concept can be implemented in hydroelectric dams, power plants, solar panel farms and in regular homes with slight changes to the sensors required but with the same or a similar architecture to support it [40].

3.3 Internet of Things Platforms

In this section, we will give an overview of the most widely used IoT platforms in the market today. The aim of this work is not to make an extensive comparison between the various existing platforms, this is why we decided to focus on the ones developed by large IT companies like Google, Amazon or Microsoft. The rationale behind this choice is the Magic Quadrant for Cloud Infrastructure and Platform Services, an annual report by Gartner [7] that focuses on the gap between the best platforms on the market because IoT platforms are often supported by infrastructure resources (storage, computing and networks, for example) [3]. These basic IaaS cloud, computing, storage and network resources as a service offer useful capabilities for this work, hence the choice of these platforms and not a standalone IoT platform with no cloud infrastructure to support it.

Open source platforms and tools did not offer the same technology stack as these and despite the possibility to build solutions tailored to our needs that often requires more time, and in this case, some out the box solutions were already implemented, tested and highly scalable for a compelling price. For large companies with medium to large tech teams, this may be the best path, because it will allow them to use existing resources, build a custom solution and save costs for not using already implemented but priced services. However, for small businesses, there is a very interesting turn out for choosing either one of the big tech companies options. The pricing is more attractive since its made for solutions that require high throughput and in the case of IoT much fewer devices than large factories while being able to use the most recent services and technologies available, tested and developed to be highly scalable from the ground up.

In the end, small businesses that choose to use platforms backed by big IT companies will be able to build solutions technologically on par with others implemented by renowned enterprises using their own or open-source IoT systems but at a significantly lower cost and possibly even faster. The disadvantage of choosing an IoT platform from big IT companies is the dependency of using their services and servers, shifting an entire system to another cloud provider can become cumbersome, which can become a possibility if they decide to change their business plan or increase their prices.

However, small businesses, given the fact that they need far fewer requirements, will almost always feel a significant price increase much later than large enterprises, which may cause the latter to look for another cloud provider to host their stack or even switch to open-source, certainly something that is not in the best interest of the current provider while ultimately safeguarding the best interest of small businesses.

As we can see in Figure 3.1 the three main competitors come from large IT companies, but another particularly important aspect in this project that helps consolidate our choice is the longevity we want to provide. Another reason that led us to choose these platforms is the fact that they have more financial and human resources, which most of the time translates into more and better-implemented functionalities as well as innovation.

[7] https://www.gartner.com/en Last accessed on: 27/11/2021

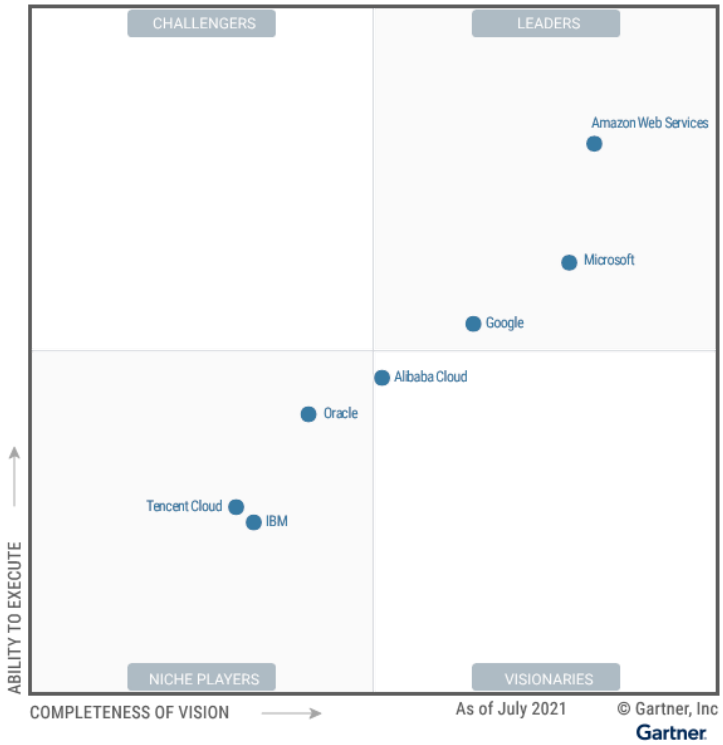


Figure 3.1: Magic Quadrant for Cloud Infrastructure and Platform Services as of July 2021. Taken from here.

After the selection of these three options, a theoretical comparison was made to reduce the selection to just two. Google Cloud [8] did not fit our requirements since one of its main offerings is the capability to deploy machine learning models to custom-built edge devices enabling users to build solutions with high performance inference in the same location where the data is being gathered, which provides lower latency and great privacy. However, their IoT service and features offering is less consolidated and with less support for new users than Azure and Amazon Web Services (AWS). These last properties, alongside the market longevity aspect, were particularly relevant for this use case and consolidated our choice.

The Design Science Research (DSR) approach ultimately helped us choose the platform to sustain our solution, and the choice of the AWS IoT platform was based on their vast offering of affordable services as well as useful features combined with consolidated learning space for new users of AWS resources, the availability of white-papers, blogs, straightforward documentation and numerous courses and workshops on their resources pages [9] [10].

[8] https://cloud.google.com/ Last accessed on: 27/11/2021
[9] https://www.aws.training/ Last accessed on: 27/11/2021
[10] https://explore.skillbuilder.aws/learn Last accessed on: 27/11/2021

CHAPTER 4

System Overview

This chapter aims to present the implemented system, its architecture and technologies used based on the requirements that are also presented. We then describe the components of the developed prototypes, the reasoning behind their choice and the layout of the devices. After introducing the reader to the components, we present the algorithms and the general functionality of the different components of the system, from the IoT prototypes to the algorithms used to process the data from the devices. We also give an explanation of the indoor location system developed for the workshop and our IoT application. We end by presenting the technologies and the development process of the web application.

4.1 Description

As introduced in chapter 1, the aim of this work is to build a system capable of monitoring and controlling classic cars restorations. Although it is stated in the title of this work, this system can be applied to any car, nonetheless, one of its main purposes is to help the classic car owners get more information, quicker and in a more systematic way than simply inquiring the workshop.

These types of cars most of the time are considered as investments by their owners, meaning that any restoration made should be according to code, trusting the workshop owner and workers can sometimes help with the business but a transparent procedure detailing the entire restoration process of a car helps confirm that it continues to hold its value.

The workshop used for the development of this work is the Raimundo Branco workshop [1], a classic car restoration shop in Portugal. There are several steps between the start and finish of a restoration process and these vary from car to car, however, these are the main goals of this project:

[1] https://www.facebook.com/raimundo.joaquim.branco/ Last accessed on: 27/11/2021

- Identification of restoration process steps.
- Control of the device responsible for the data collection.
- Monitoring of the entire system.

The stages of the restoration process we want to identify are as follows: mineral body blasting, spot weld dent remover, hammers, electric sanders, hand sanding, painting and polishing.

Hammers, dent removers, electric sanding machines and manual sanding are considered to be body shop restorations, usually the first type of repair a car needs after a crash. The most common use cases seen throughout the data collection stage was dent removals but occasionally, on rare classic cars, there is a need to build entire panels from scratch using special bending machines and hammers. Painting is one of the most important steps in the entire restoration process, especially when dealing with classic cars and finally, polishing usually ends these major steps in restoring a classic car. Later on, we will explain the process of collecting and data processing required to be able to identify some of these steps and which sensors are useful for each use case.

This work is combined with another dissertation [22], responsible for converting the processes identified by this work to information easy to understand for the car owner. After processing the data from the sensor boxes, this information is sent, to be confirmed by a workshop manager. After confirmation, the identified restoration tasks get assigned to the restoration project and are available for the car owner to access.

This chapter is branched into different sections corresponding to the three main components of this work: the IoT device named sensor box, the algorithm responsible for processing the sensor box data and a web application responsible for controlling, managing and monitoring the entire system.

The explanation of the integration between the two dissertations mentioned above can be found in section 4.8, but next we proceed with the detailed description of the solution architecture.

4.2 System Architecture

The Institute of Electrical and Electronics Engineers (IEEE) 1471 is an IEEE recommended practice for architectural description of systems approved in 2000 that helped define software architecture "as the fundamental organisation of a system embodied in its components, their relationships to each other and to the environment" [7]. Software architectures need to be designed with the technology speed of advancement in mind, ensuring that their interpretation remains the same over time.

Another necessary caution in designing a system architecture is the high cost resulting from any modification of it. However, it is clear that software architectures have a tendency to change due to the rapid advancement of technology, regardless of how carefully they are designed [4]. Keeping in mind the common changes that occur in system architectures and to minimise the probability of presenting an obsolete architecture, we decided to show two points of view on our system architecture:

- an abstract view of our architecture made up of components giving it a higher probability of surviving the constant technology advancement.
- a detailed architecture, with instances of the technologies chosen, an approach more likely to become obsolete.

Before presenting these two architectures, the system needs to be broken down into components so that it continues to meet its requirements. These components can be considered subsystems within the larger system, the workshop restorations identification system. These are its main components:

- **Sensor Boxes** - responsible for gathering and pre-processing information from the environment.
- **Processing algorithm** - responsible for converting sensor data into insightful information.
- **IoT platform** - responsible for data analysis and device management.
- **Web Application** - the main source of interaction between the IoT platform and the stakeholders.

The sensor boxes are our IoT devices, each one is handled by a workshop employee when restoring classic cars, gathering data and sending alarms when abnormal situations are detected. The data recorded by the IoT device is sent to the cloud, triggering the algorithm. The IoT platform connects the devices with the stakeholders via the web application, where they can perform a series of operations.

Below we present the two approaches mentioned, an abstract architecture less affected by the constant technology advancement and another that focuses precisely on the technologies chosen for our solution.

The architecture presented in Figure 4.1 does not commit to a single technology, it only expresses the need for a cloud service provider. In the left side of the diagram, inside the IoT subsystem, we have a component that represents a sensor box, although there is only one represented, this system is capable of handling multiple devices operating simultaneously.

The architecture diagram that follows, presented in Figure 4.2 is very similar to the previous one, but once we choose the technologies that will be used in our solution, we can simply swap the cloud service provider for an architecture that follows AWS standards, specifying all the services required by solution.

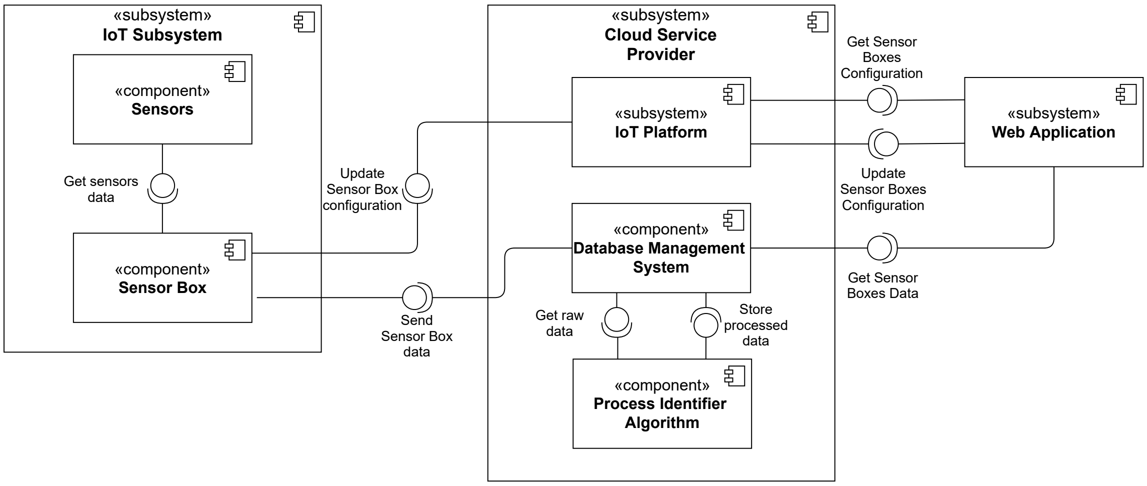


Figure 4.1: Component diagram of the system's architecture.

In Figure 4.2 we have the sensor boxes, with the sensors responsible for sampling data through its sensors. These devices communicate with *IoT Core* to receive and update their device shadows, passing their alarms to a *DynamoDB* table. The rule responsible for processing the sensor boxes alarms also sends an e-mail to the manager stating the error occurred. Periodically, these IoT devices send raw data and log files to an *S3* bucket, under different folders, however, for easier understanding, these are depicted as separate buckets. Each time a raw data file is sent to the *S3* bucket the Process Identifier Lambda function is activated. This function queries the beacons and tools data to get the most recent beacons layout, the active tools and uses this information to process the raw data. When finished stores it on another folder under the same *S3* bucket and notifies the subsystem developed in the scope of [22] that a new processed file has been created. The integration of both systems is presented in section 4.8.

All the data from the sensor boxes and the *DynamoDB* tables can be accessed through the *Amazon Application Programming Interface (API) Gateway* by an authenticated user (verified by *Amazon Cognito*), to be noted that the *API Gateway* service does not interact directly with the other AWS services, each API request triggers a *AWS Lambda* function that executes the back end code. By representing only one *AWS Lambda* function in Figure 4.2 we obtained a cleaner approach, not cluttered with the numerous back end functions developed in our solution.

We mentioned earlier in subsection 2.1.1 two commonly used layered architectures for IoT applications. Following the more detailed Five-Layer Architecture presented earlier, our solution can be divided in the following manner:

- **Perception Layer** - the sensor boxes on the left of the diagram above.
- **Transport Layer** - the sensor boxes communicate with the AWS services via Wi-Fi.
- **Processing Layer** - it is composed by the *IoT Core*, the *S3* buckets used for storing the data, including the Lambda function responsible for processing the data.

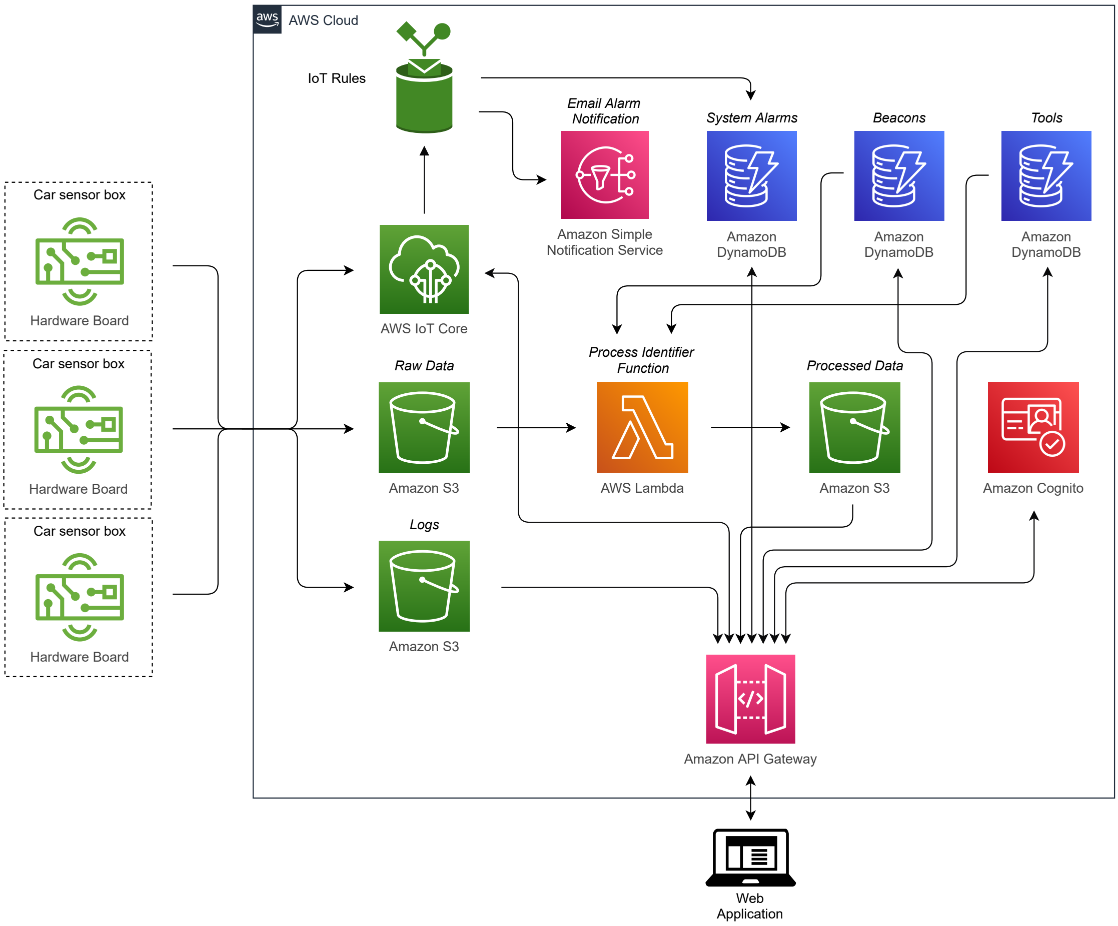


Figure 4.2: Architecture diagram according to the technologies chosen earlier.

- **Application Layer** - this layer enables the delivery of services and data to the user, the *API Gateway* duty.
- **Business Layer** - this final layer is responsible for managing the whole IoT system and is represented by the web application icon at the bottom.

4.3 Raspberry Pi

IoT has stimulated the development of microprocessors and portable computers capable of handling an impressive amount of computing load, given their size. The portability and low cost of these devices brought a new possibility to IoT devices, as it became easier to implement solutions that needed these features to become mobile.

A Raspberry Pi is a low cost, portable computer mainly designed for users with less programming experience, but its high versatility attracted more experienced users to build solutions as they are reliable, easy to replace and useful for projects that required more power than a cheap microcontroller can provide.

There are other useful computing units to power IoT devices like Arduinos or ESP8266 microcontrollers, but we decided to use a Raspberry Pi given the processing power and configuration available for most IoT platforms.

Like any electronic device, it is prone to fail, especially in a workshop where drops are regular, this ended up making our choice easier because the Raspberry Pis are available in any regular electronic shop, can be easily replaced and can be accessed remotely, which can be quite useful to solve problems without the technician needing to go to the sensor box location.

The user interface is simpler to use because it shares similarities with some operating systems, making troubleshooting and updating the system easier for people who have no knowledge in the area, the learning curve for dealing with this type of devices is less steep when compared with microcontrollers.

The use of a Raspberry Pi helped us avoid the need to use large breadboards for the development of this project because it allowed us to connect the sensors directly to the Raspberry Pi pins using jumper cables, which made it easier to replace sensors and the overall device maintenance. Avoiding the need to build a set of custom Printed Circuit Board (PCB) from scratch with specific sensor connectors accelerated the prototype development.

Some of the most common IoT platforms require the developer to install an Software Development Kit (SDK) on the device to give access to libraries and accelerate the development. The use of Raspberry Pi facilitated the development of a script capable of converting a regular Pi into a sensor box by installing required dependencies, downloading the code, setting up credentials and connecting to the IoT platform. Later in this chapter, we will explain how the device will perceive the environment and how this also helped us to choose the Raspberry as the computing unit for our sensor boxes.

Manual sandpaper and electric sanding machines produce small particles that can affect the performance of the Raspberry Pi over time, settling on connector ports that are not useful in our use case ultimately reducing the life expectancy of the device. This issue ended up motivating us to find a solution to not only keep the device clear from this dust but also to deal with the heat produced by it, one of its main disadvantages, that often comes with more powerful computing units when compared to microcontrollers.

Since regular heat dissipation fans would pull all the dust into the device and drain a lot of power from the battery the only option left was passive heat dissipation through a heat sink. The Coolipi [2] website sells sets of passive anodised aluminium heat sinks with 3D models, allowing anyone with access to a 3D printer to build its own passive heat dissipation case. However, these models needed to be upgraded to cover the Raspberry Pi's connection ports for dust protection and to accommodate the sensors, as well as the screen used to provide feedback to the user. The end result of a sensor box 3D model is presented in Figure 4.3.

[2] https://www.coolipi.com/ Last accessed on: 27/11/2021

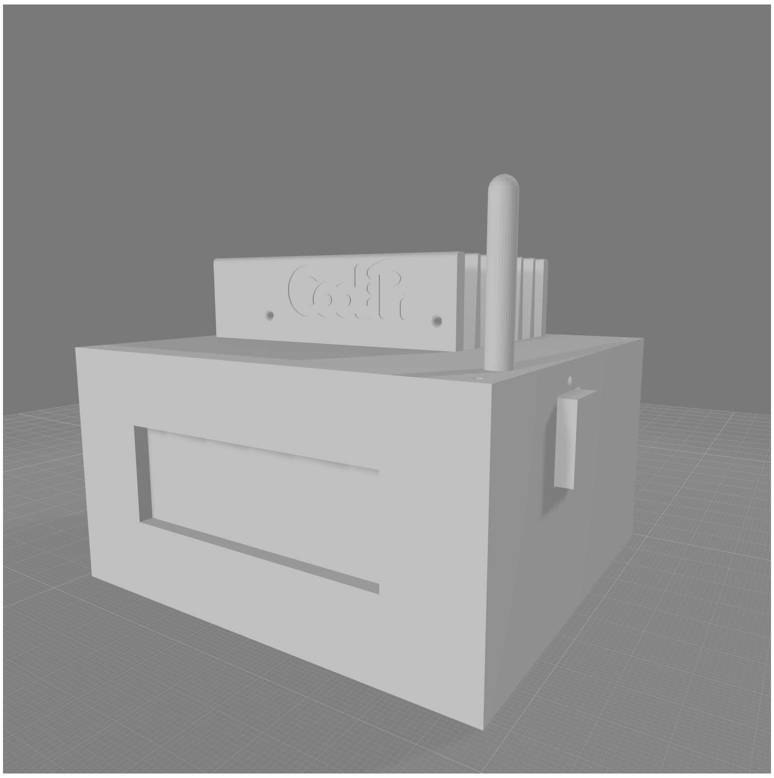


Figure 4.3: Sensor Box 3D model with custom case and Coolipi heat sink.

This passive heat dissipation (seen on top of the device) allowed us to satisfy the requirements needed, improving the sensor box life expectancy while maintaining its robustness and development cost-efficient. The updated model was built by João Sousa, a member of Vitruvius Fab Lab [3] at the technology and architecture campus of ISCTE-IUL [4] and the sensor placement was taken into account when modelling this model. As it can be seen in Figure 4.3 the temperature sensor (in front and similar to an antenna) is in the furthest possible location from the heat sink to minimise any case of it affecting the restoration processes identification.

4.4 Sensor Box

The sensor box represented in Figure 4.3 is the IoT device of our work and it is responsible for the recording, pre-processing and submission of data and or alarms to the IoT platform. Each worker has their own assigned sensor box and before starting a new restoration job, he needs to select the car to work on in the web application. This step will allow to cross information and assess the restoration job, the worker who did it and the car restored.

Since it is common for workers to be assigned to one car at a time from the start of the restoration to the end, this initial setup will not recur throughout the working

[3]https://vitruviusfablab.iscte-iul.pt/ Last accessed on: 27/11/2021
[4]https://www.iscte-iul.pt/ Last accessed on: 27/11/2021

day, only when a worker starts a new restoration project. This is because the sensor box configuration remains unchanged when the device is restarted.

The sensor box has four status:

- **On -** the device is on and ready to receive instructions.
- **Off -** the device is off and cannot be configured.
- **Stand by -** the device is **On**, it is not recording data but can be configured to do so.
- **Recording -** the device is **On**, actively recording data and logging, can be configured.

This sensor box control is made via the web application, and all boxes can be controlled the same way assuming they have an active internet connection. If a sensor box is temporarily without an active internet connection when it reconnects its local configuration is updated to the configuration requested by the user on the web application.

The configuration of any sensor box in our system is always available via the web application, regardless of its status and connectivity, enabling users to configure their boxes even when these are not active, knowing that these configurations will be passed on to the device when it turns on.

Each sensor box has a set of settings that can be tailored according to the worker or workshop needs:

- **Sleep time -** the frequency at which the device records data, this value can vary from 0 to 100 seconds between each recording.
- **Vibration frequency sample -** the frequency while sampling recording the accelerometer data, varies from 100 Hz to 3200 Hz (3200 samples recorded per second).
- **Vibration sample duration -** the amount of time the sensor box records data from the accelerometer, from 5 to 60 seconds.
- **Temperature thresholds -** the minimum and maximum temperature thresholds, for normal sensor box execution in ºC as the temperature differs significantly throughout the year.
- **Humidity thresholds -** the minimum and maximum humidity thresholds, for normal sensor box execution in % from 0% to 100% since the humidity, like the temperature, differs significantly throughout the year.
- **Box owner -** the worker responsible for the device, can be changed to any worker.
- **Car -** the car being restored, can be changed to any car with an active restoration project.

Temperature and humidity thresholds help manage the unpredictability of weather, seasons and even erroneous readings by allowing users to set minimum and maximum values, any temperature or humidity sensed outside these thresholds triggers an alarm, notifying the user of a possible sensor malfunction.

Since restoration processes we aim to identify can take hours, days or in some cases weeks, our solution does not send data in real-time as most messages would be similar resulting in wasted resources (both monetary, for writing data to the cloud, and energy-wise, because sending data regularly drains the battery faster). Therefore, the device stores the data locally and sends it to the cloud when it sees fit. With this implementation, the car owner does not receive real-time updates, yet there is still the possibility of being notified several times a day while keeping costs down.

The ability to manage data frequency recording, vibration sample duration and even device sleep time brings advantages to this work. Increasing the sleep time for tasks that are known to be time consuming can save a lot of resources and the combination of these three different settings will ultimately control the size of the data file when necessary, giving the system manager the ability to further optimise the system at the device level, which will translate into resource savings in other services.

In the next section, we clarify in detail each component present in a sensor box and describe its purpose.

4.4.1 Components

Each sensor box can be powered by any 3.0A power bank or battery, in section 5.2 we test the power consumption of our sensor box. The power bank enclosure of the sensor box can be seen in Figure 4.4 as well as the top and side covers.

As we mentioned earlier, the boxes have Raspberry Pis 4B as their processing unit, justifying the need for a 3.0A power source and multiple sensors.

Whilst the sensor box designing process, we decided to choose sensors based on the processes we wanted to identify. The combination of data from these sensors would, together with the location system inside the workshop presented in section 4.5, help us meet the goal of this thesis.

From the sensory perspective a sensor box has an accelerometer, a temperature and a humidity sensor, the last two can be seen Figure 4.3.

Since the accelerometer does not require special needs from the environment it is located inside the sensor box. The temperature sensor stands out vertically like an antenna and the humidity sensor is located on the lateral right side of the sensor box.

Although this combination was capable of identifying certain processes, it was unable to help up identify all tools and therefore all tasks presented in section 4.1. However, for some cases, the indoor location system could narrow the workers' restoration process, reducing the possibilities and helping the workshop manager verify the tasks.

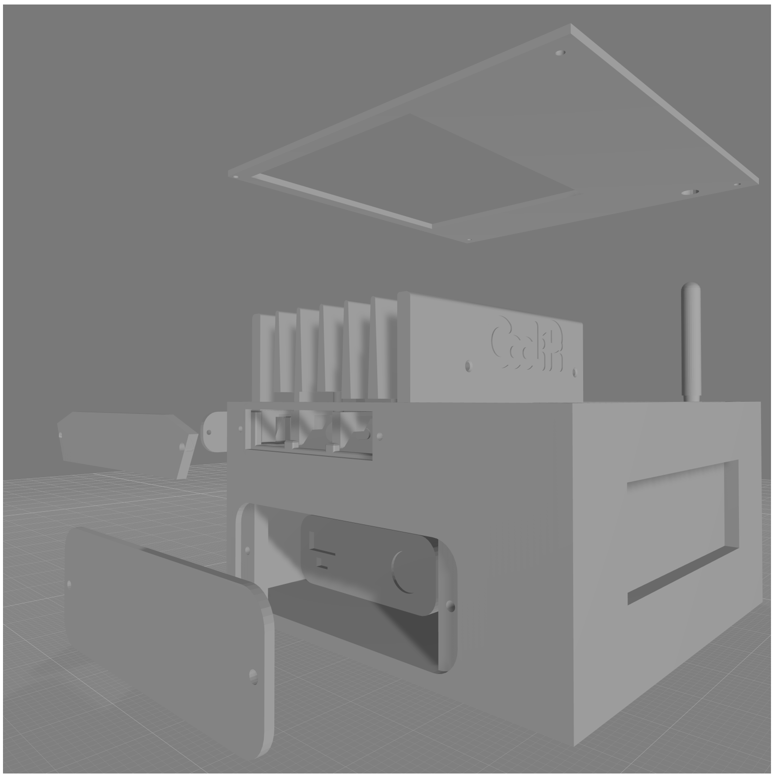


Figure 4.4: Sensor Box 3D open model showing the power bank enclosure.

For user feedback purposes, each box is provided with a screen (located on the right side of the sensor box from Figure 4.4) and a strong magnet located below the power bank (capable of holding 6.4 Kg, a maximum force of 62.8 N) to adhere to the car's metal sheet. There are some classic cars with an aluminium body panel rendering the magnet unusable, for these specific cases a clamp can be used to tightly secure the box to the body panel. The custom-built Coolipi cooled sensor box case protects the Raspberry Pi, the cables and helps with the sensor box overall heat dissipation. Below, we present the schematics of a sensor box.

The schematic in Figure 4.5 represents the Raspberry Pi and the sensors used in the following order: temperature, accelerometer and humidity. This particular choice of temperature sensor, as a probe, was made to reduce the temperature influence from the computing unit and the heat sink. The breadboard was required to share power for all three sensors from the 3.3V Raspberry Pi connector.

As this pin can supply a maximum of 16 mA (milliampere) it is important, for data integrity and not to reduce the life span of the sensors, to confirm that all sensors have sufficient power to operate without problems. The temperature sensor requires between 1.5 mA and 4 mA[5], the humidity sensor requires 1.5 mA at peak usage[6] and the accelerometer

[5]https://datasheets.maximintegrated.com/en/ds/DS18B20.pdf Last accessed on: 2021/10/28
[6]https://www.sparkfun.com/datasheets/Sensors/Temperature/DHT22.pdf Last accessed on: 2021/10/28

used features an ultra-low power measurement mode of 140 μA, or 0.14 mA[7].

The choice of LCD was motivated by the need to present the greatest amount of data to the user. This screen can also be seen in the front panel of the model from Figure 4.3 and it provides information about the box identification, the car currently assigned to the box, the worker responsible for it and the errors the box is experiencing. When integrating the LCD with the sensor box it was necessary to use an Inter-Integrated Circuit (IIC) library taken from here [8] that allowed us to turn on and off both its screen and brightness setting as well as send text to it.

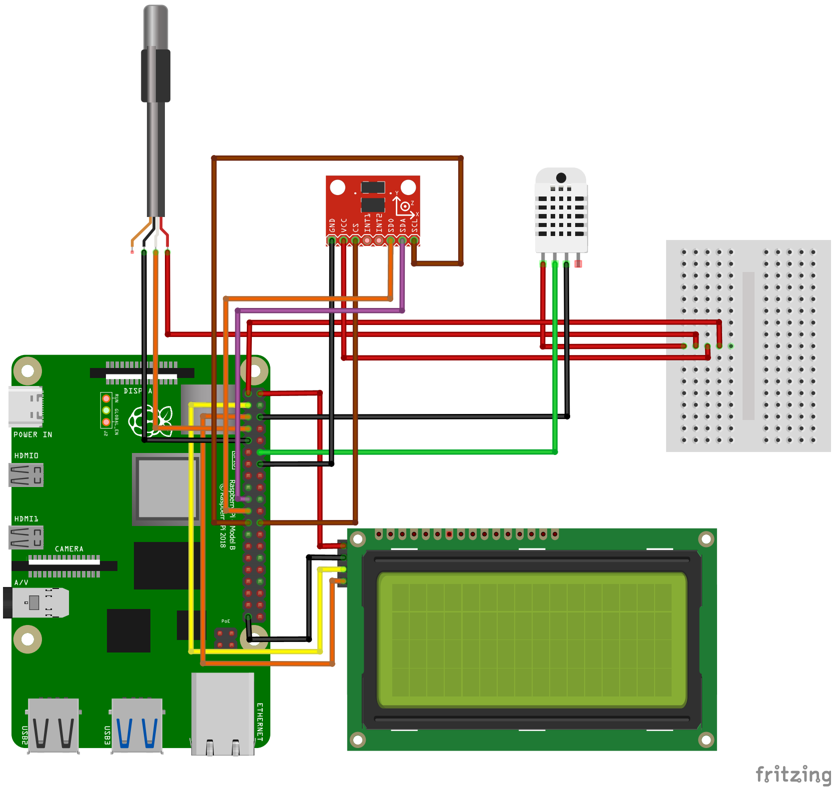


Figure 4.5: Sensor Box schematic.

To allow the inclusion of new tools in the workshop it was necessary to build another device, simpler and with fewer components than the sensor box presented above. It consists only of a Raspberry Pi, an accelerometer and a screen to give frequency information to the user. This box can also be used to update the operating frequency range of a tool, does not collect data and logs nor can it be controlled by the web application. The script for collecting, processing and displaying data from the accelerometer runs automatically as soon as the box is turned on until it is turned off.

[7]https://www.analog.com/media/en/technical-documentation/data-sheets/ADXL345.pdf Last accessed on: 27/11/2021

[8]https://github.com/dork3nergy/lcd_2004 Last accessed on: 27/11/2021

4.4.2 Alarms and Logs

Each sensor box, upon setup and after the dependencies installation, attaches a service to the device boot using systemd [9]. This service causes the device to start the main script automatically, allowing the user to switch on the device and use it immediately.

The device stores its logs locally and, when the user turns off the device, this file is sent automatically if the device has an active Internet connection, if not, the next time the user turns off the device, it sends the current log files and the previous ones that weren't sent. The same file management system is implemented for the raw data files, however, they are sent if their size is over 100Kb or if the file has been created more than 4 hours prior.

As we mentioned earlier, every electrical component in the sensor box can fail, meaning that the sensor box needs to be able not only to recover from errors but also to inform the worker or even the IoT platform manager of its errors.

When using sensors, errors are common and when an error is detected the sensor box retries the execution. If a certain component continues to fail, then an alarm is sent to the manager and displayed on the screen for the worker. This alarm is available in the web application with greater details. This feature allows us to build a device more capable of dealing with errors, avoiding crashes when any minimal error occurs.

4.4.3 Sensors Calibration

Despite the fact that most sensors are calibrated at the factory, for critical applications where accuracy is crucial, as our solution, it is highly recommended to calibrate the sensors to guarantee the devices best performance.

The BLE beacons have a temperature sensor enclosed in a silicone casing and since these are being used in our system they needed to be calibrated, the same way as our temperature and humidity sensors. Even though the procedure was repeated for all sensors, the Estimote [10] beacons are not able to detect sudden changes as well as the temperature sensor due to its casing.

For these three sensors, the calibration method used was the comparison method, a procedure where the measured values of the sensor being tested are compared to the values measured by a reference sensor, in the same environment during a certain period of time. We calibrated all the temperature and humidity sensors using a Velleman PMHY-GRO [11] as our reference sensor, an affordable digital hygrometer and thermometer with high resolution.

The accelerometer sensor was calibrated by comparing its output in both directions of each axis to a known stable reference force, gravity. Later on, a 3-axis G-Force Datalogger:

[9] https://systemd.io/ Last accessed on: 27/11/2021

[10] https://estimote.com/ Last accessed on: 27/11/2021

[11] https://www.velleman.eu/products/view/?id=435222 Last accessed on: 27/11/2021

VB300 [12] was used to help confirm the frequencies calculated using the accelerometer sensor.

4.4.4 Accelerometer Data Processing

One of the most challenging tasks of this work was to process the data from the accelerometer sensor, turning it into information we could use to identify different restoration tasks. The FFT enables us to convert the accelerometer data (originally in a time-domain) to a signal in the frequency-domain.

To take advantage of the Serial Peripheral Interface (SPI) bus interface found in Raspberry Pis and to avoid the development of a controller for the ADXL345 sensor, an interface from here [13] was used in our solution. This interface takes advantage of this bus and is responsible for programming the sensor sampling rate and data collection duration. Using this interface has simplified the SPI bus interface integration allowing the sampling of data from the sensor to occur at the maximum frequency of 3200 Hz, something that would not be possible with the IIC interface bus, which enables a broader range of tools to be detected.

The sensor interface writes the accelerometer data in a file, its size can vary depending on the sampling frequency and the measurement duration, both settings can be changed by the user in the web application. For a sampling frequency of 3200 Hz and a duration of 10 seconds, the user can expect a file with 32000 entries, each one with values for the 3 axes.

The sensor sampling frequency is directly related to the types of tools that can be identified due to the Nyquist frequency, which is always half of the sensors sampling frequency (also known as bandwidth) [18]. This means that if the sensor is set to record 3200 readings per second, the Nyquist frequency is 1600 Hz, the maximum frequency that it is possible to detect by this sensor. Reducing the sampling frequency helps in faster processing (if the recording duration remains constant) but loses some data and since the accelerometer data file is deleted after being processed, we recommend the user to set the accelerometer to record at a sampling frequency of 3200 Hz to identify future tools without any issue. Below is a plot of accelerometer data in a time domain, a 10-second recording of a powered sander operated by a workshop worker. The different colours refer to the data for each of the axes.

After storing the information in the file, the data is ready to be processed. The algorithm uses Numpy [14] implementation of the FFT algorithm on the data of each axes. After applying the FFT to the data we have a spectrum similar to the one presented in Figure 4.7 with frequencies on the x-axis and the acceleration magnitude on the y-axis. Note that the x-axis is limited to 1700 Hz, yet the data stops at 1600 Hz (half of 3200 Hz,

[12] http://www.extech.com/products/VB300 Last accessed on: 27/11/2021
[13] https://github.com/nagimov/adxl345spi Last accessed on: 27/11/2021
[14] https://numpy.org/ Last accessed on: 27/11/2021

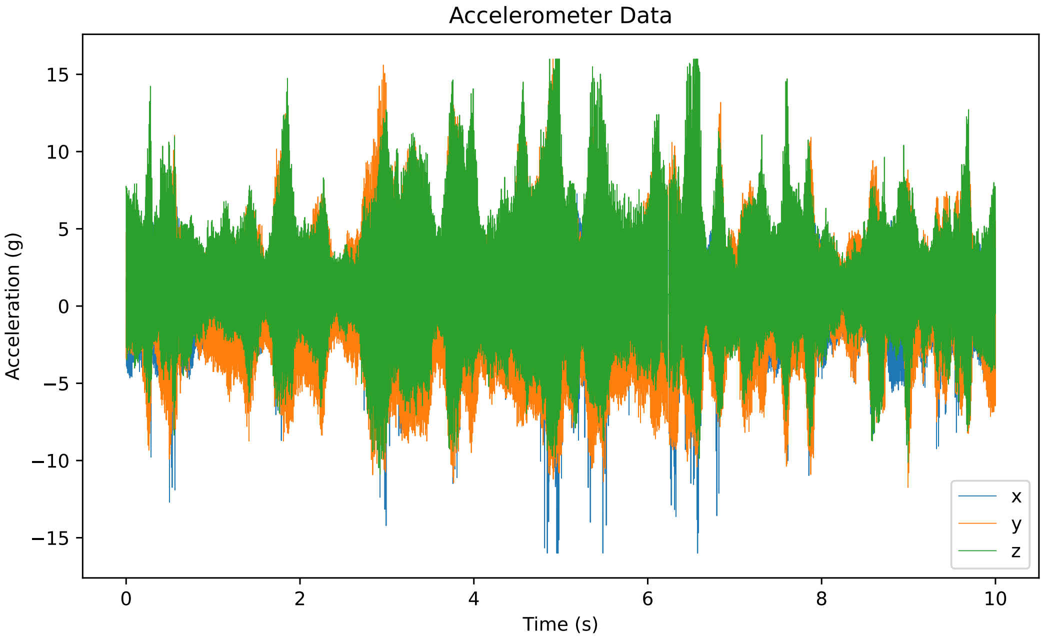


Figure 4.6: Accelerometer data in a time domain.

the frequency sampling used in this recording) result of the Nyquist frequency. As we can see in Figure 4.7, the identification of the peak values of the acceleration magnitudes can be used to access the corresponding frequencies to then identify the fundamental frequency, which for this data set is 130 Hz. A flow diagram of the algorithm responsible for identifying the frequencies from the accelerations detected by the sensor is available in the appendix.

When a hammer is used there is a spike in the acceleration data, however, this has not proved useful due to its poor consistency and high variability, workers can apply different blows to bodywork and each worker has a different stroke intensity, which could lead to wrong tool identifications. Despite the lack of consistency of hand tools (hand sanding and hammering), the power of the signal can be used to detect when the most intensive activity happened. If we can identify the peaks in the acceleration magnitude values, we can evaluate the frequencies by looking for the x-coordinates of these values which correspond to the frequencies of the signal perceived by the sensor with the highest intensity.

A similar technique consisting in setting a fixed minimum peak height to return all relevant peaks and discard noise was used by Ullrich et al. [35] to detect gait using an accelerometer and a gyroscope inside a small device clipped to a shoe. Since our solution is required to work for several different tools, we decided not to set a fixed value to filter out the noise and instead use the 98th percentile, following the same approach of Li et al. [21] responsible for the development of a vision-based cardiopulmonary monitoring system. This means that the percentile would get the acceleration magnitude value below which 98% of the power values would be. A 98% percentile allows us to get

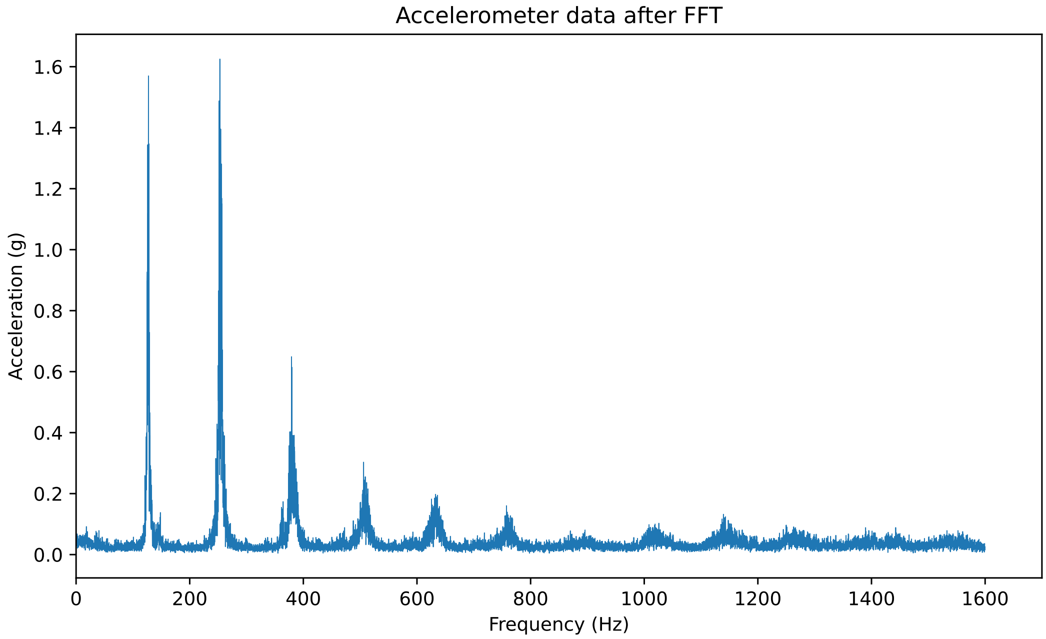


Figure 4.7: Accelerometer data in a frequency domain after the FFT.

the highest values of the acceleration magnitudes, filtering out the rest, without setting a fixed value. We made tests with several percentile values and due to the high amount of data, this ended up being the number that worked best for most of the data, sometimes even managing to identify frequencies of manual sanding.

After identifying the frequencies that correspond to the values with the highest acceleration magnitude, we identify the fundamental frequency, since it is expected for the signal to have harmonics associated. These fundamental frequencies are stored and later used by the algorithm responsible for identifying the tools.

During the development of this project, several visits were made to the workshop to collect data of tools that would need to be identified by the system, each time small adjustments were made to the algorithm to improve the identification of fundamental frequencies.

However, there were some difficulties for the algorithm to identify non-electric tools such as dent-dollys, hand sanders and hammers due to the high amount of noise, a problem that accompanied us during part of the development of this work. Several attempts of noise reduction were implemented, through the installation of PyPi [15] packages created for the effect but without success. Sometimes the algorithm was able to identify frequencies relative to hand sanders but not with the expected precision. In the future, someone with knowledge in cleaning and processing temporal data from sensors may provide assistance in the development of this part of the project.

Despite the less positive results explained above, the algorithm works perfectly in

[15]https://pypi.org/ Last accessed on: 27/11/2021

identifying frequencies of power tools, being able to identify changes in frequency when the speed of the machine is changed. This proves that the algorithm is able to identify frequencies however some fine-tuning is still needed to be able to identify tools that typically have a lower intensity on a car panel.

4.5 Indoor Location System

Identifying different restoration processes is challenging using only sensors as input data, so a new approach is needed that can help filter the different types of restoration processes. Although the location may be useful, since the devices will be inside the workshop, GPS performance is not consistent or reliable so it must be discarded.

The indoor location however, proved to be relevant given our use case because the layout of the workshop is static, like most workshops each worker has their own work booth and the sections are well established. As an example, the bodywork restoration area has the heavy tools needed for that type of repair, just as the painting area has the painting booths.

This static zone layout is used to our advantage by reducing the type of restoration tasks that may be happening in each zone and can be used in other places that also have static layouts, such as factories and other workshops.

To build our indoor location system we used Estimote BLE beacons. These have two modes available: Indoor Location and Estimote Monitoring, however, the decision was made to keep our solution as generic as possible, avoiding relying on the Estimote technology to allow future possible integrations with beacons from different companies. This company does not provide much confidence for users when it comes to long term support which can directly affect our solution. However, it is important to note that although we chose to avoid the technology developed by this company we are using their application to configure all the beacons with the same frequency and signal strength, a task that once completed does not need to be repeated.

As we mentioned above, an initial setup of the beacons is recommended for optimal performance through a mobile application available for Android and iOS. Each beacon, alongside the temperature and identifier, sends their battery percentage which allowed us to build a battery monitoring system, triggering an alarm for the manager if a given sensor box receives the signal from a beacon with a low battery. The user can also change this minimum battery value in the web application.

During the development of the project, Raimundo Branco's workshop underwent a total reconstruction, offering better conditions to workers, customers and optimising the restoration process. Alongside this physical transformation, the development of this work helped drive a potential digital transformation of the company. This new scenario motivated some adjustments during data collection and, in particular, in the tests carried out to ensure the reliability of the location algorithm inside the workshop presented in section 5.5. For the remaining part of this section, we will explain how the algorithm works and how it integrates with the rest of the system.

When recording data, the sensor box stores a new packet in the current data file with the sensors information, the current timestamp, the car assigned to the box, its identifier and the most recent beacons signals picked up by the device. Each message with data to be processed necessarily has a set of beacons, these beacons correspond to the most recent view the box has, given the BLE signals it receives.

The sensor box is constantly collecting data from the beacons and locally updating its view of them, with the help of timestamps to know at any time which are the most recent signals it has received, given the possible high mobility of the box, it is important to filter out the beacons that are associated with a possible restoration task. Since location plays a very important role in identifying the restoration tasks of a car, the best beacons that can be used for any task identification are the most recent ones, the ones that have been detected while performing the restoration process. After storing this data locally, the data file is eventually sent to the cloud for processing.

It is the processing algorithm hosted in the cloud that queries the beacons information from the system, which allows us to change these beacons status and location without directly affecting it. It is recommended to change beacons from one zone to another before or after the working day hours since a mid-day beacon replacement, although possible can decrease the possibilities of correct identification of zones by the algorithm.

Nonetheless, the algorithm has beacon redundancy, meaning it does not rely on a single beacon to identify a zone, it averages the beacons distances to the sensor box meaning more beacons per zone increases the probability of a correct inference by the algorithm. Since the algorithm can match a beacon identifier to its zone, it can infer the zone by choosing the lowest average distance of incoming beacons organised by zone. Beacon redundancy helps the algorithm to become less vulnerable to a performance decrease during a mid-day beacon replacement, which can sometimes happen.

Another possible course of action for when this happens is for the user to simply change the beacon status in the web application, which disables the use of this beacon during the algorithm's execution, while the other two beacons in that zone can still help determine the location of the sensor box within the workshop. Each time the processing algorithm is activated, it queries the most recent status of all system beacons to avoid the use of beacons marked as unusable by the workshop manager. The reasoning behind the availability of beacons battery information is to avoid this type of outcome, allowing the user to have more control over the entire system.

As we mentioned earlier, the algorithm uses the most recent information from the beacons as well as beacon redundancy for the sensor box location inference, but together with the beacons data sent by the box is the RSSI value associated with each signal sent by the beacon and constantly updated locally in the sensor box. This value is a measure of the strength of the signal received by the sensor housing and is used to determine the signal strength indicator present in all phones today. Despite its low accuracy when compared to GPS, as far as indoor location systems are concerned it is still one of the most common approaches and ultimately allows us to continue to fulfil our desire to build a generic system, as any beacon can be used.

For each beacon set in a message from a raw data file, the RSSI values of the beacons are converted to distances and these distances are grouped by zone. The average of all distances by zone is made and the lowest distance zone is selected. Later in this work, in section 5.5, we will present the tests done in more detail, showing the algorithm results.

4.6 Cloud Development

Our solution was built using AWS services, which allowed us to focus on how to solve the problem instead of focusing most of the time on code development to avoid problems on topics such as availability, server management and scalability while being able to keep operating costs low. Serverless architectures also have their drawbacks, despite the vast array of documentation and guides provided by AWS, it is still a constantly evolving topic that requires the developer to always be up to date to provide the best solution possible.

There is also a learning curve in building serverless applications as you need to learn how the provider has designed their services, decide which services to use, how to pass data or trigger actions between services and what the purpose of each service is. Without the experience and even with full documentation and guides, you often need to change one service for another over details. This can mean learning a new service, its possibilities, drawbacks and integrations with other services, delaying the development of the project.

Thorough knowledge of most of the services provided by AWS is recommended and for someone new to the platform can quickly become overwhelming to decide the most suitable architecture for solving the problem. Our cloud architecture changed a few times during development, which turned out to be beneficial for future reference, but ultimately delayed the development of this phase.

Our cloud architecture, presented in section 4.2 contains services shared by different components, the IoT platform, the process identification algorithm and the web application back end, each one of them is presented below.

4.6.1 IoT Platform

AWS provides more services than the regular *IoT Core*, the basis for any IoT based application built on this provider. The full IoT architecture provided by AWS is presented below:

The Data Services analyse data from the IoT fleet to relieve the burden of processing algorithms mainly when applications require the use of ML models where data must have the same structure. These services can also be used if the user desires to complement the device's data with external data, preparing it to be processed. Our solution does not use any of these services, however, we have built code to have the same functionality offered

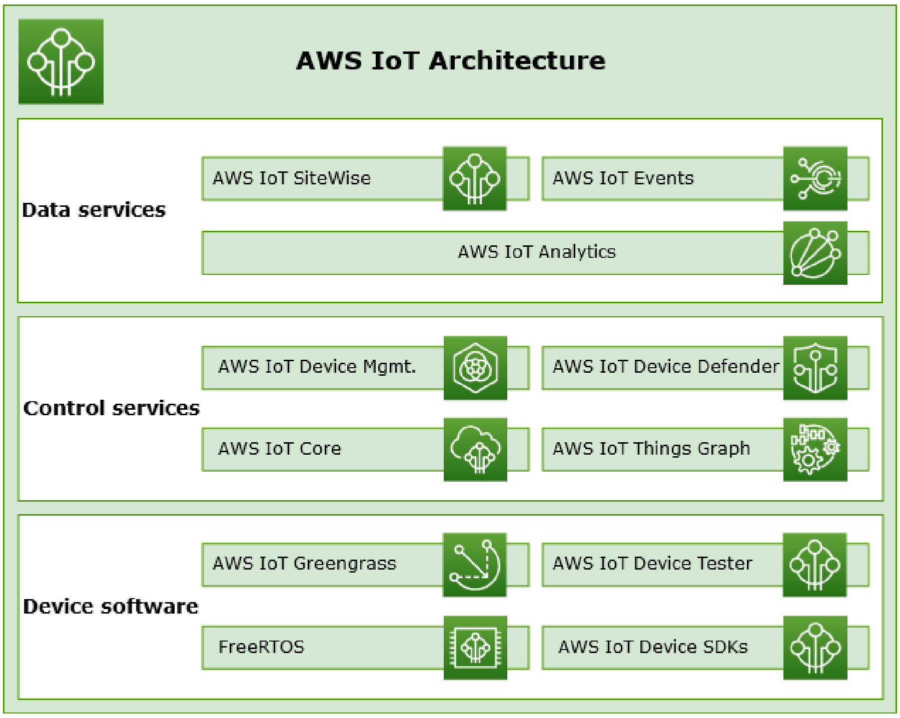


Figure 4.8: AWS IoT Architecture taken from here.

by some of these services. This allowed us to have a custom alarm system that essentially detects and responds to alarms from the fleet without requiring the use of *AWS IoT Events*. These types of situations can be seen as uncommon when building a solution solely by using services from a cloud provider in a serverless fashion, however, throughout the development of this work, it was possible to spend less monetary resources while achieving the same goals offered by other services.

The Control Services focus on controlling the entire IoT system and its devices. Our solution uses *IoT Core* [16] as the IoT platform. This service allows users to manage, connect several devices at scale, provide several communication protocols, secure connections with device certificates and grant developers the possibility of processing their devices' data how their solution requires it. This full circle is the basic setup to build an IoT application, whilst providing a high level of customisation when designing a solution.

During the development of this solution, the need to test the system with more than one device eventually arrived which made us develop a device setup script to convert a regular Raspberry Pi to a sensor box, ready to record data. This was possible because of the AWS control services APIs available, enabling regular devices to add themselves to the system, with proper credentials.

This ended up changing the architecture of our solution and required the need to build

[16] https://aws.amazon.com/iot-core/ Last accessed on: 27/11/2021

a system capable of dealing with a high tool and beacon placement variability, which is explained below in subsection 4.6.2. We decided instead to use the *Python* [17] *AWS IoT Device SDK* to optimise the sensor boxes connections with *AWS IoT Core*.

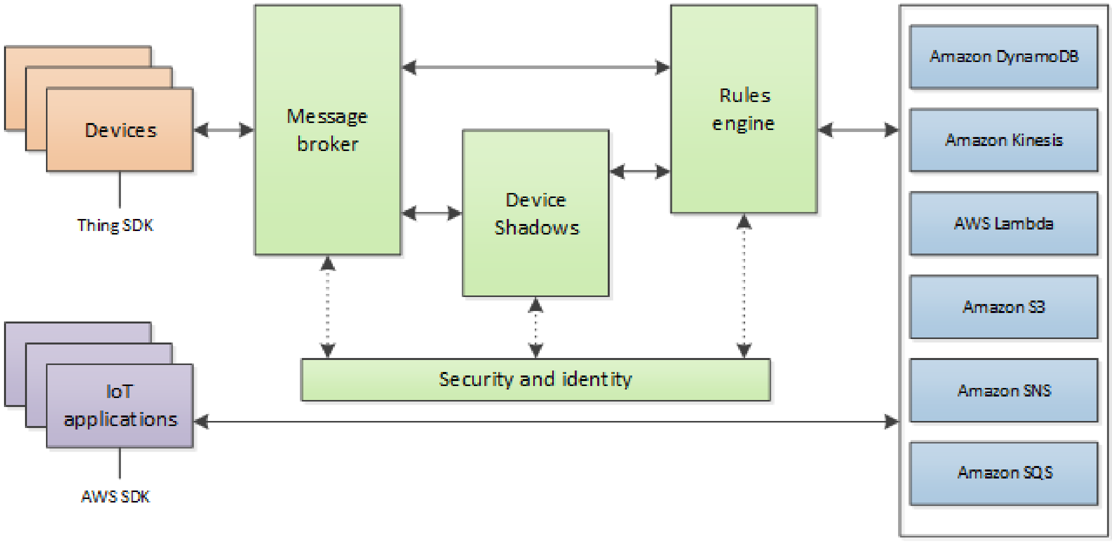


Figure 4.9: AWS IoT Core data services taken from here.

While building the IoT application we created a rule using the *Rules Engine* in Figure 4.9 to allow the IoT devices to send data to other services, for instance when an alarm occurs in a sensor box a message is sent to a rule that receives the message, stores it for later access via the web application and notifies the user by sending a custom e-mail. Below we present the *AWS IoT Core* services to better explain the *Rules engine* and the *Device Shadows* used in our solution.

Each rule is composed of a query to select what data to pass on to other services and a set of actions that are triggered as soon as this rule is activated. This feature provides the ability to store alarms in an efficient way without needing to write code to access other AWS services directly on the device, reducing the risk of failure. A downside of this implementation is it requires an active internet connection, if a device does not have one, the only feedback to the user is through the LCD screen.

An important feature of IoT Core is the wide range of communication protocols available, offering MQTT [18], Hypertext Transfer Protocol Secure (HTTPS), MQTT over WebSockets (WSS) and Long Range Wide Area Network (LoRaWAN) as communication protocols and device specific SDKs in multiple programming languages to help and more easily secure communications between the device and the cloud.

Early in the development stage, we studied the possibility of providing an active continuous connection between the sensor boxes and the web application to display the sensors data in real-time, providing updated monitoring, however, this would eventually drain the device battery faster than expected and easily stop working as soon as the

[17]https://www.python.org/ Last accessed on: 27/11/2021
[18]https://mqtt.org/ Last accessed on: 27/11/2021

device lost connection. These and other possible occurrences, along with the irrelevance of having the car owner receive real-time updates, changed our system architecture and communication protocol.

Since we did not require a long-range communication protocol or continuous data transferring which enabled us to rule out LoRaWAN and MQTT over WebSockets. HTTPS did not suit our solution because we designed our box to be fully configurable and to have the notion of shadow, which is designed for a publish/subscribe communication protocol so we decided to use MQTT, the standard for IoT messaging. This communication is commonly made through a message broker, however, AWS IoT has a cost-free alternative, *Basic Ingest* [19] that avoids the need to set up a Message Broker in their end. *Basic Ingest* is used in our alarms rule, however, the device shadows still use the regular broker because they need the publish/subscribe functionality and the ability to store the devices state.

The *Device Shadow* is one of the most relevant features from *IoT Core* used in our solution since it allows any user to configure any sensor box in a quick and efficient way while being always accessible even when the sensor box does not have an active internet connection or is turned off. This is the service within *IoT Core* that encouraged the choice of AWS as our cloud provider, even though Azure also provides this feature, digital twins. WSO2 [20], an open-source IoT platform studied while designing our solution does not have anything similar to this type of feature, directly affecting the solution we planned to build since the beginning.

Each device has a policy stating what the device is allowed to do, what services can access and where it can send data via IoT rules explained above. *IoT Core* also gives the developer the ability to create groups of devices and attach policies to groups instead of singular devices, easing the policy update process across a device fleet. This device hierarchy becomes useful if Raimundo Branco's workshop managers decide to add new IoT devices to complement the solution.

4.6.2 Process Identification Algorithm

The algorithm is written in *Python* [21] and was deployed as an *AWS Lambda* [22] function. As recommended when developing serverless applications, we used a framework designed to help write Infrastructure as Code (IaC). *Serverless Application Model (SAM)* [23] is an open-source, free framework that provides a syntax to help users with the development, testing, deployment and maintenance of serverless applications. In this case, our application is composed of a *Lambda* function, triggered each time a new unprocessed file was uploaded to an *S3* [24] bucket by a sensor box.

[19] https://docs.aws.amazon.com/iot/latest/developerguide/iot-basic-ingest.html Last accessed on: 27/11/2021

[20] https://wso2.com/ Last accessed on: 27/11/2021

[21] https://www.python.org/ Last accessed on: 27/11/2021

[22] https://aws.amazon.com/lambda/ Last accessed on: 27/11/2021

[23] https://aws.amazon.com/serverless/sam/ Last accessed on: 27/11/2021

[24] https://aws.amazon.com/s3/ Last accessed on: 27/11/2021

The restoration tasks or process identification algorithm is one of the most relevant components of this work. A flowchart of the algorithm is depicted in Figure A.1 in the appendix. As we mentioned earlier, the sensor boxes record data from the sensors in a sequence. For the accelerometer sensor it records and processes data from the boxes to identify the fundamental frequency (or frequencies) as explained in subsection 4.4.4. This data is stored in a file sent by the sensor box and each data entry in the file is composed by:

- **Device Identification** - the device which recorded the data.
- **Car Assigned** - the car associated with each data entry.
- **Timestamp** - the exact time of when the data was recorded, in Unix time.
- **Temperature** - the temperature value sensed.
- **Humidity** - the humidity value of the environment.
- **Beacons** - the information of the beacons from which the box received signals during data recording.
- **VibrationPeaks** - the set of frequencies identified by the algorithm presented in subsection 4.4.4.

The reasoning behind including the sensor box name and the car assigned within each data entry and not only at the beginning of the file is because the sensor boxes can switch cars, a common use case when there are several cars prepared and waiting to be painted. This allows the worker to use the same box on another car and continue working, with the only requirement being the need to update the sensor box configuration or device shadow using the web application. Having the corresponding car assigned within each data entry also allows the same file to contain data from multiple restorations, reducing operational costs.

The sensor boxes indoor location enables us to filter the types of restorations that could have been recorded, justifying the need to send the beacons information in each data entry. The variability of tools and beacons placement forced us to abandon a ML approach meaning that it was necessary to build a solution robust and capable of dealing with these types of occurrences.

By storing the data of all the tools and beacons in the system, we have been able to know at all times which tools are actively being used and the current placement of the beacons. The ability to query this data removed the problem of tool and beacon variability since now the algorithm knows the status of the information required to make inferences.

A ML trained model would be discarded and a new one created every time any change occurred regarding the tools (necessary to identify the tools used by the workers) or the beacons (required for indoor location). After querying for the active tools and the latest beacon placement inside the workshop, the algorithm processes each data entry sequentially, inferring the location of the box using the beacons data from the data entry.

The algorithm then analyses the sensors data depending on the zone inferred. As stated earlier, each zone corresponds to specific tasks or a set of specific tasks. Since the zones in a workshop are static, this particular topic is not open to variability, but for future reference, it might be interesting to assume that this can happen and change the algorithm accordingly, following the same approach used for the beacons and tools.

All processes we intended to identify fall into four zones within Raimundo Branco's workshop:

- **Mineral blast** - a specific zone where cars get sandblasted to remove rust or paint.
- **Bodywork** - the zone where custom body parts are build and fixed.
- **Paint** - where the paint booth is located.
- **Post-Paint** - zone used for final details and polishing.

This clear separation allows us to focus on the data from certain sensors, depending on the zone. For the mineral blast, since there was no opportunity to detect this task with the sensor box, it can only be inferred using the sensor box location. The paint zone can be identified with the location of the sensor box and the values of both the temperature and humidity sensors. Later in this document, we will present a test done inside the paint booth while a car was being painted and cured section 5.3.

The bodywork and post-painting areas require the use of both mechanical and non-mechanical tools, so the input vibration peak frequencies are used. Since the algorithm already has the working frequency range of the active tools in the workshop, it is a matter of comparison to identify them.

For all processes that require tools, it is common that sensors continue to store data despite not actively identifying tool frequencies, this is due to the sensor box not having information about its location. In a paint zone, there is no need to store accelerometer data, for example, and this could be an additional optimisation to drain even less battery. However, this requires all sensor boxes to have an active internet connection to be able to actively pull up the latest placement of the beacons or be notified as soon as any changes happen. If, for example, a sensor box is not able to query this information, it may discard relevant data, which directly affects the system's output. By intentionally making the sensor box not too “smart“, we are able to avoid problems like these.

So our sensor box continues to record data, despite not detecting any vibrations, as the worker may be having a break or changing to another type of sanding paper. As the box continues to record data, this means that between the correctly identified peaks, there can sometimes be messages that are stored without any identified vibrations. These entries are not accounted for by the algorithm and only messages that have frequencies in the same range as any tool in the workshop are stored as processed data. This allows us to filter out any erroneous values that have been stored in the sensor box due to a missed reading or pre-processing.

The workshop contains different sets of tools for each type of work, which means that there is a high likelihood of these tools to have a similar range of frequencies within each tool type, so it is reasonable to expect that one frequency of a powered sander may correspond to more than one tool. When this scenario happens, if the identified tools are of the same type only that type is stored, but if the tools have different types, both are returned for the manager to select which tool was actually used. This is why we decided to store all workshop tools and use them to identify different tasks. This allows us to access and manage a wide range of tools without affecting the identification of the restoration task performed.

In some cases, the beacons provided do not identify the zone in which the sensor box was located during data recording. This type of issue is not relevant for two of the four zones (bodywork and post-paint) as these have the same procedure of trying to identify frequencies, however, this can be a problem when zones that use different types of sensors as input data get mixed up. If a tool is being polished near the paint booth and the beacons picked up by the sensor box converge towards the paint zone, this means that a painting process would be incorrectly identified. This could be dismissed later by the workshop manager, but it can also be addressed by accessing the temperature data recorded by the sensor box. At the start of the execution, the algorithm queries the temperature around the workshop vicinity using OpenWeather [25] API. There was a need for an API usage since some restorations can be done outdoors and the beacons (which also have temperature sensors) are located inside the workshop. If we would use the beacons temperature in a car being fixed outdoors (while being identified in the correct zone) this could make the algorithm discard the restoration.

Before comparing the tool frequencies and identifying a painting process, the algorithm compares the sensor box temperature with the API temperature, if an error occurs when querying the API, the average temperature detected by the beacons associated with the specific entry being analysed is used instead. This helps avoid misidentifying a paint event (since the temperature inside a paint booth ranges from 25°C to 65°C) when the sensor box is outside the paint booth and prevents the algorithm from comparing peak vibration frequencies when the sensor box is actually inside the paint booth. As the reader may notice, there is still a chance of this happening while the paint booth is heating up (temperature between 25ºC and 40ºC), however, the occurrence of this type of phenomenon has been reduced and, as we mentioned before if a process is incorrectly identified, the workshop manager has always the ability to discard it.

Since the data is processed sequentially and some processes have a longer duration, some of the processes are expected to repeat, however, the algorithm always checks the last identified event and if the current event is the same then it just updates its final

[25]https://openweathermap.org/api Last accessed on: 27/11/2021

timestamp, reducing the overall size of the processed data file and giving an easier way for the workshop manager to understand when one event has ended and another has started.

After processing all the data in the input file the algorithm stores the new file in the *S3* bucket in the folder associated with the sensor box and notifies the workshop manager that there is a new processed data file. In section 5.6 we will present the testing procedure that validates this algorithm with different types of input files.

4.7 Web Application

The web application was developed with feedback from people who know the requirements, the workshop and what this solution should deliver. Through the recording of data of the car restoration process, and the attention to detail regarding the style of operation of the workers it was possible to build a web application with all the necessary requirements. These were identified at an early stage and complemented throughout the development of this work.

We were able to identify scenarios and build a solution capable of adapting to the existence of these scenarios that, had they not been contemplated, would have negatively affected this system. For instance, by observing the workers, we were able to identify the need for the data processing algorithm to take into account periods in which no activities were being performed in a particular car, simply because the worker might be taking a break. As well as these, other situations arose and ended up shaping our solution, increasing its robustness.

4.7.1 Serverless Web Application Back end

While describing the cloud architecture we informed the reader that in between the connections *API Gateway* and the services were several *Lambda* functions that enabled the user to browse and interact with the system and IoT devices. These *Lambda* functions, similarly to our Process Identification algorithm presented above in subsection 4.6.2, were developed with *SAM* using *YAML Ain't Markup Language™ (YAML)* [26] serialization language. The difference being the runtime of the *Lambda* function, for the back end we used *NodeJS* [27] as opposed to *Python*, used in the development of the Process Identification algorithm.

The *Lambda* functions were developed using *Typescript* [28] due to its type safety over standard *JavaScript*, which meant that it was necessary to compile the code into *JavaScript* before deployment due to the lack of support for *Typescript* on AWS. The *AWS Lambda* functions allow the programmer to define not only the execution time but also the size of

[26] https://yaml.org/ Last accessed on: 27/11/2021
[27] https://nodejs.org/en/ Last accessed on: 27/11/2021
[28] https://www.typescriptlang.org/ Last accessed on: 27/11/2021

the memory in which the code will be executed, setting its price accordingly. Memory size has a direct ratio relationship with processing power, however, for our use case, the lowest available memory size, 128 MB, proved sufficient due to the low processing required.

One of the downsides of developing a serverless back end is the time it takes to load a *Lambda* function. The first time a *Lambda* function is invoked takes more time to load because it needs to load the function code into the execution environment to then execute it, commonly denominated has a cold start. However for the next requests, since the *Lambda* function is active for the next 15 minutes (each time it is invoked) the user can have a smooth interaction while using the web application without experiencing cold starts.

Since we have an idea about the number of users that will be using our web application we decided to add a usage plan to *API Gateway*, limiting it at 5000 executions per day, and starting to throttle if the rate is above 50 requests per second or the burst of requests is above 100. If any of these limits are surpassed in a day the API fails the limit-exceeding requests and returns a “Too many requests“ to the client.

4.7.2 Web Application Front end

During the development of this work, because of the different types of input needed to build the web application, we decided to use a tool to better visualise its appearance. We used *Balsamiq* [29] to build mock-ups with the help of the stakeholders involved in the workshop. Below we can see the mock-up of the login page in Figure 4.10 as well as the initial page after the login in Figure 4.11 where you can find a list of all the sensor boxes in the system. The remaining mock-ups developed can be found in the appendix of this document at section B.3.

The development of these mock-ups helped validate the usefulness of the system however a downside of this means of communication with the client is the difficulty in capturing the flow of the application. To address this problem an Interaction Flow Modelling Language (IFML) diagram was also developed and can be accessed in section B.2. IFML is an Object Management Group (OMG) published specification, the purposes of IFML are to describe the flow of an application and also the front end view components [5] which help later on with the development of the web application code since *React* [30] relies heavily on components for code reusability and improved performance.

It was important to build a web application that was simple, efficient, memorable and with a low learning curve so that the workshop workers could use it on a daily basis, as this is the only point of contact with the whole system. These are usability attributes created by Jakob Nielsen in [24] which we aimed to follow. The most relevant usability attributes given our use case were efficiency of use and learnability because we needed the user to achieve a high level of productivity after learning how to use the system, this

[29]https://balsamiq.com/ Last accessed on: 27/11/2021
[30]https://reactjs.org/ Last accessed on: 27/11/2021

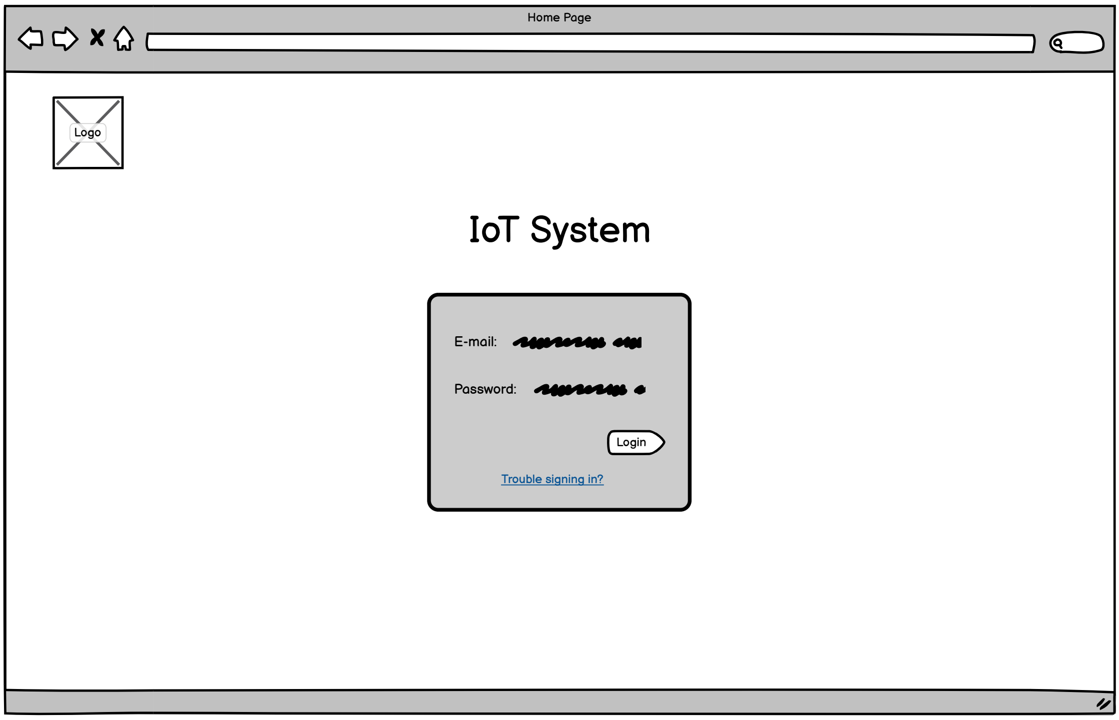


Figure 4.10: System login page mock-up developed using Balsamiq.

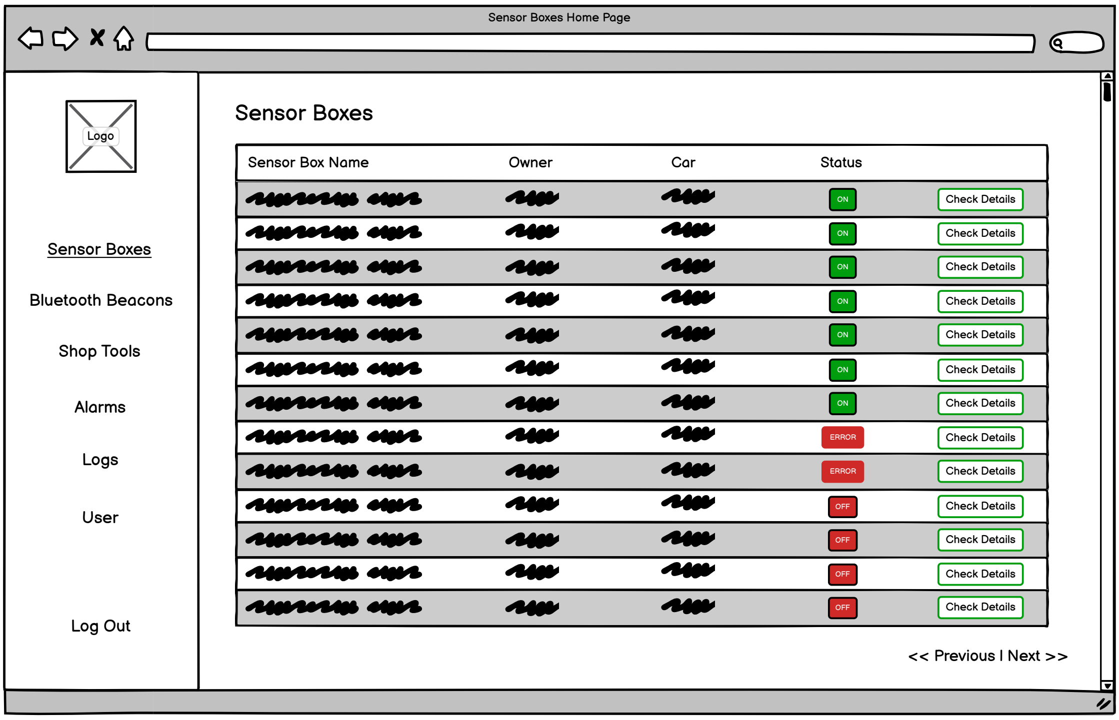


Figure 4.11: Sensor boxes main page mock-up developed using Balsamiq.

would improve the experience the workshop managers were having with Shopmonkey presented in section 3.1.

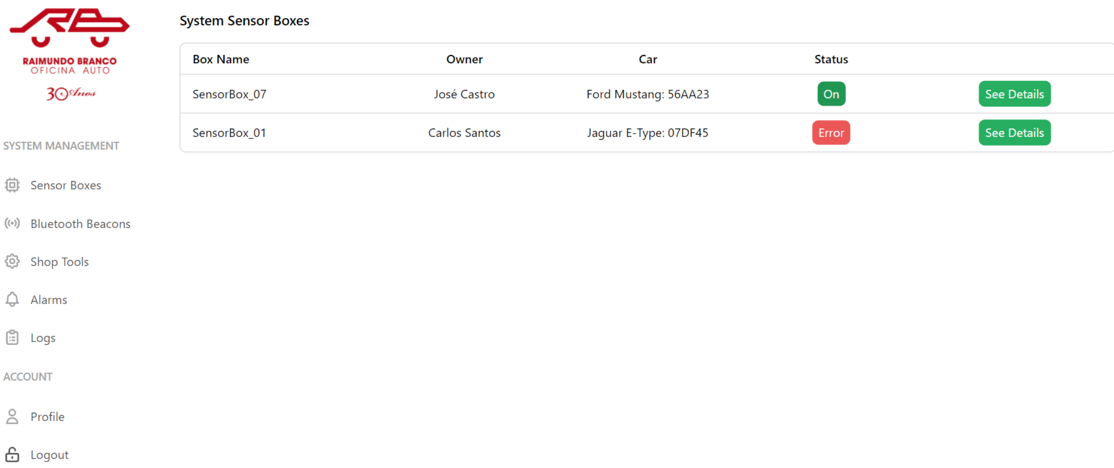


Figure 4.12: Sensor boxes web application main page.

These attributes influenced the way the web application was initially designed and then implemented, as the reader can see in Figure 4.12, the use of colour coding to help users understand the meaning behind the button and different symbols for each of the box states reduces occurrences of misinterpretation. These symbols association with a sensor box state makes them reason about it quicker leading to less confusion when compared to simple text messages. Simple changes like these help the user get used to the web application more quickly, as they feel familiar with another website or mobile application with a similar concept.

The web application front-end was developed using the *React* alongside *Typescript* and *TailwindCSS* [31].

4.8 Systems Integration

To provide a better comprehension of the system as a whole we will now present the integration between the dissertations described before: our project, which is described in depth in this document; and our colleague's counterpart thesis [22]. A car restoration process starts with the project creation in the ERPNext [32] subsystem, which triggers the creation of the project in the middleware application (Tasklist component) that interacts with the ERPNext subsystem and the Camunda [33] Workflow Engine.

[31] https://tailwindcss.com/ Last accessed on: 27/11/2021
[32] https://erpnext.com/ Last accessed on: 27/11/2021
[33] https://camunda.com/ Last accessed on: 27/11/2021

After the restoration project is created, the car can be found in the IoT monitoring and control web application (by calling the Tasklist API to retrieve the Projects list) where a worker can assign a sensor box to it. The sensor box accompanies the car restoration process and records data from temperature, humidity and accelerometer sensors. This data is then processed and different events are identified, such events are then passed onto the Tasklist component.

Since this initial restoration process identification is made based on sensor data and that can lead to wrong identifications, all data is mapped to a Business Process Model and Notation (BPMN) diagram of the project to be then confirmed by a workshop worker. Inside the process diagram, sensor identified tasks are coloured yellow to facilitate the distinction between regular unapproved tasks.

Upon the selection of the tasks by the automotive restoration shop worker, the group of tasks is sent to the server for approval, including the sensor predicted tasks or not. If those tasks are included in the selection, there is a confirmation from the subsystem that the prediction was accepted, and the prediction is promptly removed from the pool available for matching. Otherwise, it stays in the pool as it still could be an accurate prediction for a diagram executing simultaneously (e.g. two sub-processes executing in branches of an inclusive or parallel gateway).

In Figure 4.13 we present the application deployment diagram centred on the two dissertations' integration.

- **Camera Firmware** – The Internet Protocol (IP) camera firmware.
- **Workflow Editor** - The application used to model and deploy the process definitions to the remote server.
- **IoT System Web Application** - The web application responsible for monitoring and controlling the IoT system, presented in section 4.7.
- **Sensor Data Collector** - This is the edge computing component, based on a Raspberry Pi [34] running Raspbian [35] and is connected to several sensors (Humidity Sensor, Temperature Sensor and Accelerometer). It records, pre-processes, and sends data to the AWS IoT Core component for further cloud processing. This component is introduced in section 4.4.
- **Humidity Sensor Firmware** - This corresponds to the firmware of the humidity sensor.
- **Temperature Sensor Firmware** - This corresponds to the firmware of the temperature sensor.

[34]https://www.raspberrypi.com/ Last accessed on: 27/11/2021
[35]https://www.raspbian.org/ Last accessed on: 27/11/2021

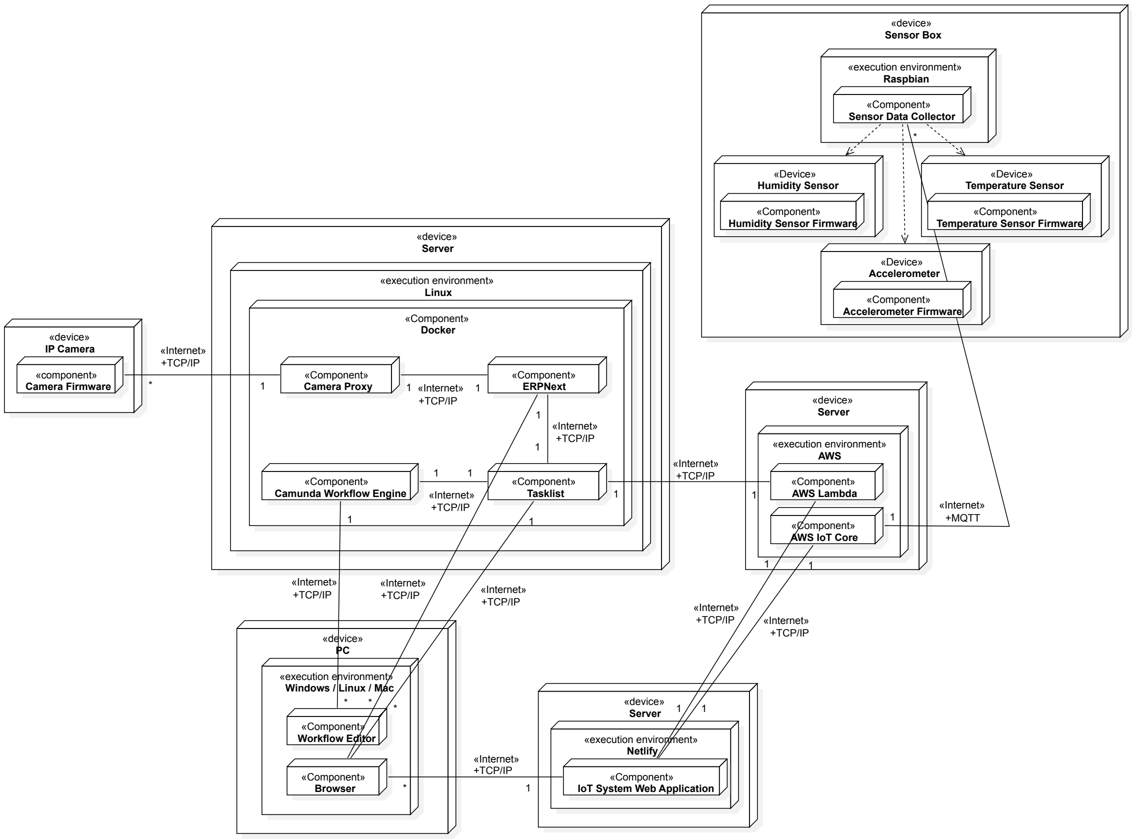


Figure 4.13: Deployment diagram of the entire system's architecture.

- **Accelerometer Sensor Firmware** – This corresponds to the firmware of the accelerometer sensor.
- **AWS Lambda** - This component runs code in response to events and automatically manages the required computing resources. Here it is used for retrieving the active car restoration projects. The full architecture of the system running on AWS is represented in section 4.2.
- **AWS IoT Core** - The AWS service that allows the communication with the sensor boxes, the IoT devices, also introduced in section 4.2.
- **Camunda Workflow Engine** - The Workflow Engine that runs the restoration process and saves the current state of every project.
- **ERPNext** - The Enterprise Resource Planning (ERP) system responsible for managing Customers and Projects, with the addition of providing a Client area where a Customer can visualize its vehicle's restoration information and camera live streams scheduled by the workshop manager.
- **Camera Proxy** - This component implements the authentication procedures for the IP Cameras and provides access to the ERPNext component.

- **Tasklist** - This component communicates with the Camunda and ERPNext components and allows the workshop floor manager to select and confirm restoration steps. It provides an API with the methods needed to retrieve the required information.

This section is based on a co-authored technical note [27] developed in the scope of the entire project and, as such, is reproduced both here and in [22].

CHAPTER 5

Data Collection, Test and Validation

This chapter provides an explanation about the evaluation of the various system components such as sensors and IoT prototypes as well as their integration with the web application and the cloud algorithm responsible for processing the IoT devices data. Throughout the validation of the different components, we will present the results of all the recorded data and the different test scenarios used to validate the system. We conclude this chapter by presenting the validity threats.

5.1 Description

The solution presented in the previous chapter is composed of different components and each one has its own validation method, so we decided to divide the validation of this system, starting with the sensor box, passing on to the indoor location system, then the algorithm in charge of processing the IoT devices data and ending with the web application validation. However, there are certain components that are more complicated to validate, as a result of having chosen a serverless architecture for the development of some components of this work, such as testing the AWS services scalability and performance assurance.

To ensure the success of this work it will be necessary to collect more data over a longer period to allow the reconstruction or adaptation of the algorithm in charge of identifying the frequencies through the accelerometer data. As we mentioned in the presentation of this algorithm, although it is able to identify electric sanders, we will also show some of the difficulties encountered during the identification of frequencies caused by other tools.

As previously mentioned, the reconstruction of the workshop occurred in parallel with the development of this solution so there was no opportunity to test the system as a whole, nevertheless, the aim of this chapter is to show that the different components work individually, that they have the capabilities to perform their task and also to test the integration of these components, even though it is not possible to test all of them simultaneously in the workshop for which this system was implemented.

In an industrial context, certain precautions are necessary when we are developing a device to be used by several workers. The premise of following the entire restoration process of a car implies the choice of components and reasoning to support it, however, it is also necessary to validate their ability to deal with this type of situation and that was also one of our focuses in the validation of the sensor box developed.

In the validation of the web application, as well as its integration with the sensor box we decided to use acceptance testing in which we used test scenarios to cover the most common situations we expect the system to be used for. The acceptance tests are concluded with surveys, to understand if the system meets the planned requirements. Conducting these tests and the feedback obtained from them became very useful as they were performed by participants with a background in Computer Science.

To finalize the validation we also performed integration tests since two dissertations were developed that are incorporated in the same system. In this type of validation, we performed tests in the two situations where both systems exchange information.

Next, we will start by validating certain components of the sensor box that are necessary due to the context in which this dissertation is inserted. Then we move on to the validation of the indoor location system, the box's data processing algorithm, its integration with the other dissertation and finally, we present the three test scenarios that will help validate the integration between the web application and the sensor box.

5.2 Sensor Box Power Consumption

To ensure that the sensor box would be able to meet our needs, it needs to be active and working, and for that, it is necessary to study its energy consumption. This becomes an important issue because we want to minimise scenarios in which the worker has to stop to change a sensor box, affecting its efficiency.

During the operation of the box, the Raspberry Pi is constantly connected and, therefore, consumes energy, as do other components such as sensors and the screen. The latter is not constantly on but gives information at each iteration of data collection from the box. A future optimisation would be to put a button to save power by turning on the screen only when necessary, but as we will demonstrate the nonexistence of this feature did not affect significantly the energy consumption when compared to the baseline test.

Another feature that we have to consider for the energy consumption is the preprocessing done by the Raspberry Pi, we believe that applying the FFT to the accelerometer data draws more energy. The sensors although they are in stand-by mode when they are not being used also draw energy.

This study was done with a Rasberry Pi 4B with 2GB of Random-access memory (RAM). We did tests with two different power banks, the ASUS ZenPower with a capacity of 10050 mAh and Tronsmart PB20 with 20000 mAh. The difference in capacities in each of the power banks gives us a broader range helping us decide which capacity is more suited taking into account the requirements described above.

The size of these power banks also accounts, as the sensor housing must be compact and small in size. As expected, the ASUS power bank is much smaller in size compared to the Tronsmart power bank and is even similar in size to the Raspberry Pi. The larger size of the Tronsmart power bank would imply a larger case which could be detrimental and even affect the data collected.

Other relevant features that justify the use of power banks are the feedback of their remaining capacity and the ease of replacement at the end of their lifetime.

Despite the recommended minimum amperage of 3A, according to the Raspberry Pi official website, a 2.5A power supply can be sufficient if less than 500 mA are used from peripherals [1]. Since we do not require the use of any peripherals, we decided to use the Asus power bank to power our Raspberry Pi during the test even though it has a maximum output of 2.4A. During the test there was no change in the operation, however, it is important to note that there is no guarantee that ensures the regular operation of the Raspberry Pi using this power bank given the lack 0.1A on its power output.

To extract the most amount of information from the power banks test we recorded the power bank capacity every 30 minutes for 9 hours, an unusual but common workday at the workshop. All power banks started the test with a full charge. The blue line in each of the graphs represents the baseline, a test where the sensor box had an active internet connection via Wi-Fi and Bluetooth activated, but not executing any scripts nor in stand-by mode. The orange line is the test performed in which the sensor box had also an active internet (via Wi-Fi) and Bluetooth connection, was actively using all sensors, recording data locally, logging, using the LCD screen for feedback and receiving signals from BLE beacons, what we expect to be a normal workshop use case.

In Figure 5.1 we can see the Tronsmart 20000 mAh power bank performance. After the active test, the power bank had 73% battery remaining meaning it used 5400 mAh for a day of work. In the baseline test, the power bank ended with a capacity of 74%, 5200 mAh. As expected, the sensor box uses more resources while being actively recording data.

Since the Asus power bank only provides a scale from 1 to 4 to indicate the battery status, we decided to use these values in the graph. In Figure 5.2 we see that the test started with the power bank at maximum capacity (value of 4) and it decreased during the 9 hours of testing. As expected, running the algorithm causes the capacity level of the power bank to drop faster but still manages to reach the end of a day's work with more than 25% of capacity.

This test proves that it is possible to use a power bank with only 10050 mAh despite the need for daily charging. The Tronsmart power bank has a longer life span but is larger in size, which limits the design of the sensor box case.

[1] https://www.raspberrypi.com/products/raspberry-pi-4-model-b/specifications/ Last accessed on: 27/11/2021

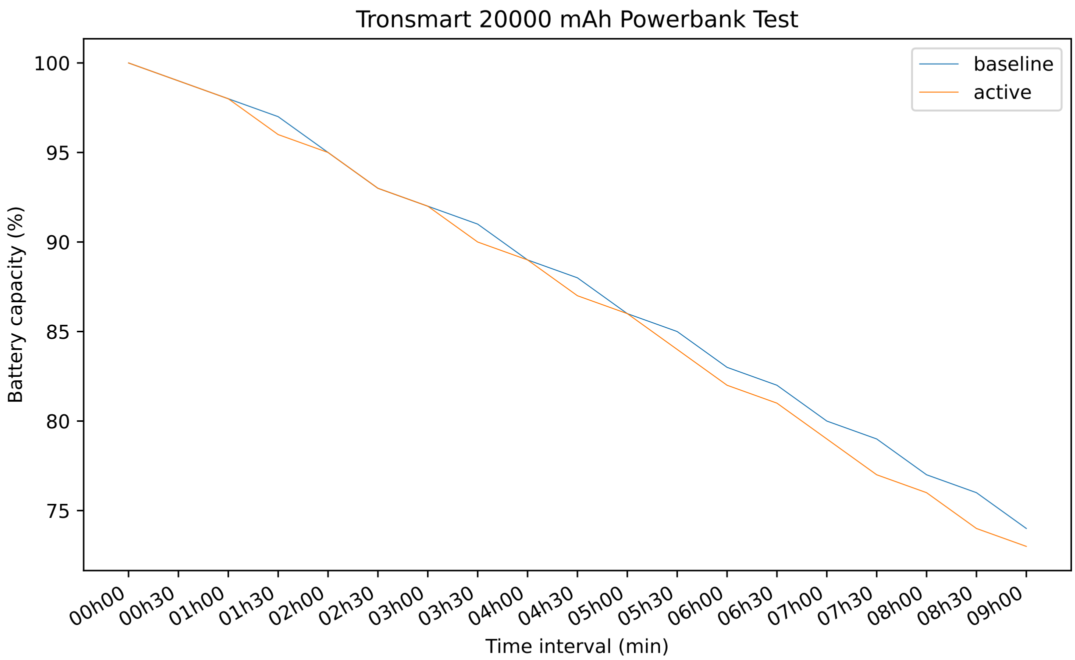


Figure 5.1: Tronsmart power bank capacity test.

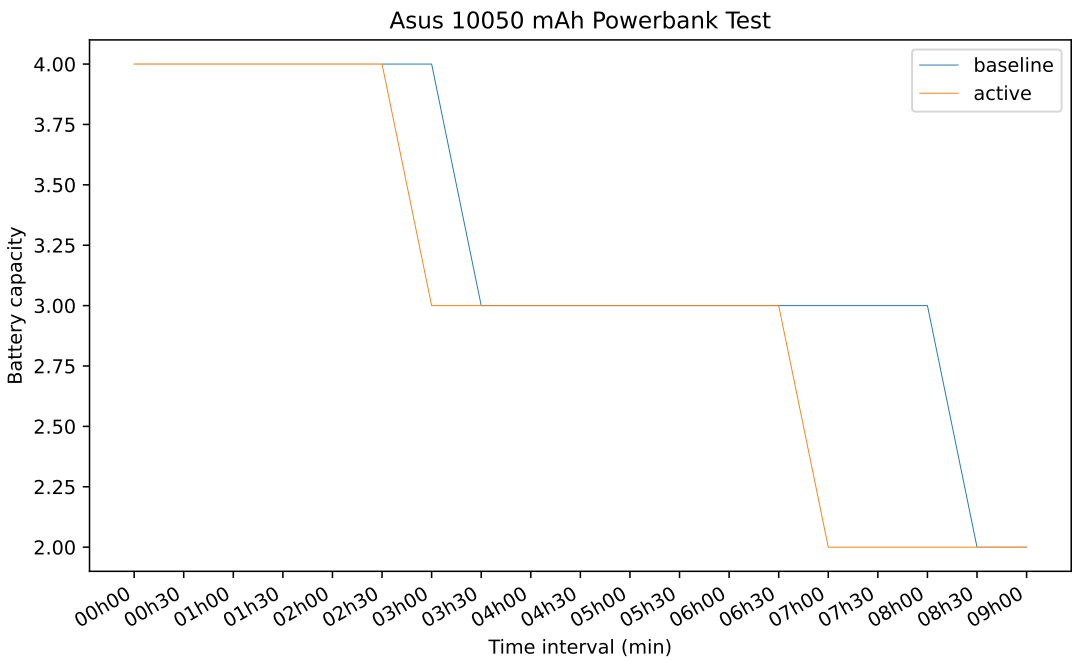


Figure 5.2: Asus power bank capacity test.

5.3 Sensor Box Temperature Stress Test

In addition to having to account for the energy consumption of the sensor box, it is important that the box is able to cope with the temperature that a paint booth can reach while baking a car.

Before starting with the temperature tests it was necessary to talk with the workers to understand in detail how the painting process occurs in order to identify situations that could damage the sensor box. It was then possible to identify the different steps of this process:

- The worker starts by preparing the car for painting, removing impurities on the body.
- The painting process begins where the paint and clear coat are applied at a stable temperature that may vary slightly but is usually around 20ºC to 25ºC.
- Finally the paint booth oven is turned on for a variable period between 20 and 30 minutes at a temperature chosen by the workers of 60ºC to finish the painting process.

During the design of this work, some doubts arose in relation to the collection of data in these conditions. Namely the fear of the paint booth acting as a Faraday cage blocking the Wi-Fi signals however this did not occur and the box managed to keep the Wi-Fi connection alive throughout the duration of the test.

It was suggested to place the sensor box on the exterior wall of the paint booth to avoid these high temperatures, however, the good insulation of these walls kept the heat inside making it impossible for the sensor box to detect any temperature difference that would become relevant. This scenario however could work, by identifying the zone only by the beacons, as it happens with the mineral blast, however since we already had the temperature and humidity sensors integrated into the system and knowing that all the components are able to handle the maximum temperature of the paint booth oven we decided to collect data inside it during the whole process of painting and baking.

All three types of sensors present, the humidity sensor, the accelerometer and the temperature sensor are capable of dealing with maximum temperatures of 80ºC [2], 85ºC [3] and 125ºC, [4] respectively. The LCD screen however has a maximum temperature of 60ºC [5], which despite being in the temperature limit referred by the workers, is a little bit lower than the maximum temperature detected during the test as we demonstrate next.

This temperature limitation of the LCD enables the possibility of the future existence of an exclusive box to be used in painting, properly insulated for heat and without the accelerometer and the screen. Through the web application, it would be necessary to associate this box to the car and the worker in charge of the painting.

[2]https://www.sparkfun.com/datasheets/Sensors/Temperature/DHT22.pdf Last accessed on: 27/11/2021

[3]https://www.analog.com/media/en/technical-documentation/data-sheets/ADXL345.pdf Last accessed on: 27/11/2021

[4]https://datasheets.maximintegrated.com/en/ds/DS18B20.pdf Last accessed on: 27/11/2021

[5]https://www.waveshare.com/datasheet/LCD_en_PDF/LCD1602.pdf Last accessed on: 27/11/2021

To ensure the safety of the most expensive component of the sensor box, the Raspberry Pi, research was done on the performance of the heat dissipation box presented in section 4.3.

Despite not having Coolipi's heat sink we managed to find a comparative study on various cases that offer heat dissipating capabilities for the Raspberry Pi. The test done, referred to as Stressberry[6], consists of software that performs stress tests on the Raspberry Pi 4B. Various users can then use their devices to test the heat dissipation capacity of their cases. The software also takes care of all the temperature sampling and production of the final graph that is shared with the community.

The author and the contributors repeated the same stress test with several heat dissipation cases of both types: active (with fans) and passive (only by contact and taking advantage of the aluminium properties).

We decided to use these results from the community to prove the Coolipi's choice, not relying on the data provided by Coolipi and since during the test performed at the workshop we didn't have access to Coolipi's heat sink. In the future, it will be necessary to test once again the heat dissipation capacity of the case but with all the components presented above.

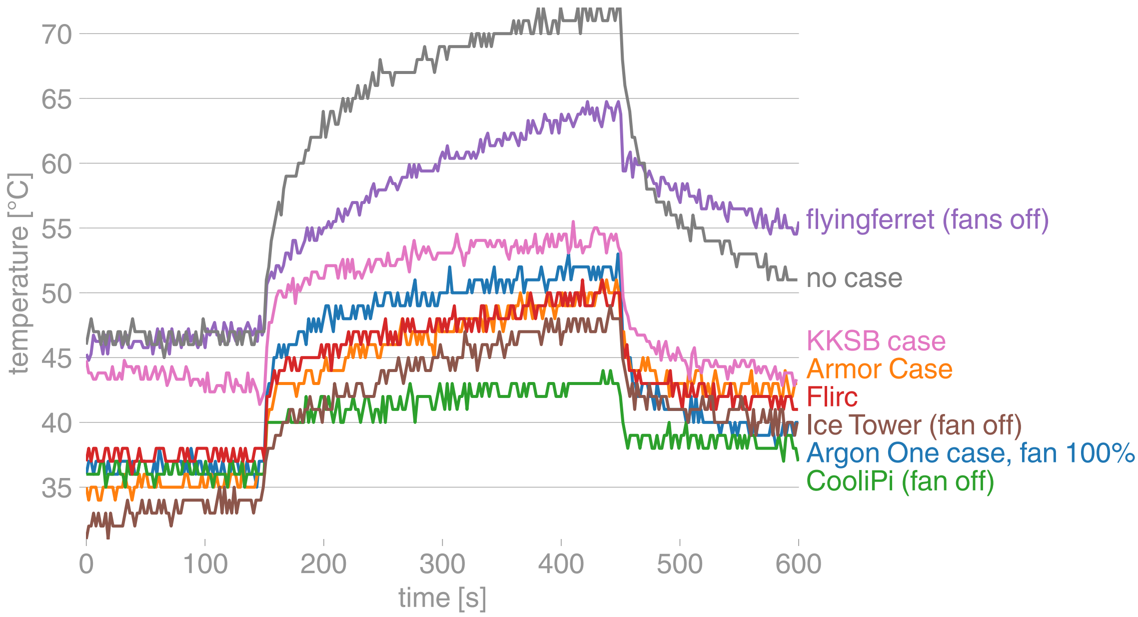


Figure 5.3: CPU temperature comparison of passive heat dissipation Raspberry Pi 4B cases cited in here.

In Figure 5.3 we can see the very positive results of the case (in green) when compared with others available in the market. Considering the need for the sensor box to support high temperatures, it becomes imperative to choose the best performing heat dissipation case available in the market, something that became possible with the help of this software

[6]https://github.com/nschloe/stressberry Last accessed on: 27/11/2021

and the community. In the following subsection, we will present the test made inside one of Raimundo Branco's paint booths and analyse the data gathered.

As mentioned above, the test performed at the workshop used a common case for the Raspberry Pi, without any heat dissipation properties but with plenty of openings for air circulation. Below in Figure 5.4 we present a photo of the device inside the paint booth, gathering data that is mapped onto the graph shown in Figure 5.5.

Figure 5.4: The sensor box recording temperature and humidity data inside the paint booth.

As shown in the graphic from Figure 5.5, it took the shop worker about 1 hour and 30 minutes to prepare the car, paint and apply the clear coat. During this period the temperature remained quite stable at 25ºC as well as the humidity, at 40%. As soon as the oven was turned on in that same paint booth we can see the temperature gradually rising to the maximum value of 65.4ºC, above the 60ºC set by the worker. At the same time, it is possible to see the humidity decrease as the temperature increases. There is a noticeable increase in humidity at the beginning of the baking process (at 15h55) which is suspected to be the humidity from the paint evaporating, but as soon as a temperature of 31ºC is reached inside the paint booth the humidity drops to the minimum value of 18.8%.

With this test, we were able to inform the workers of the temperature and humidity variation that occurs inside the paint booth and the fact that it reaches a maximum

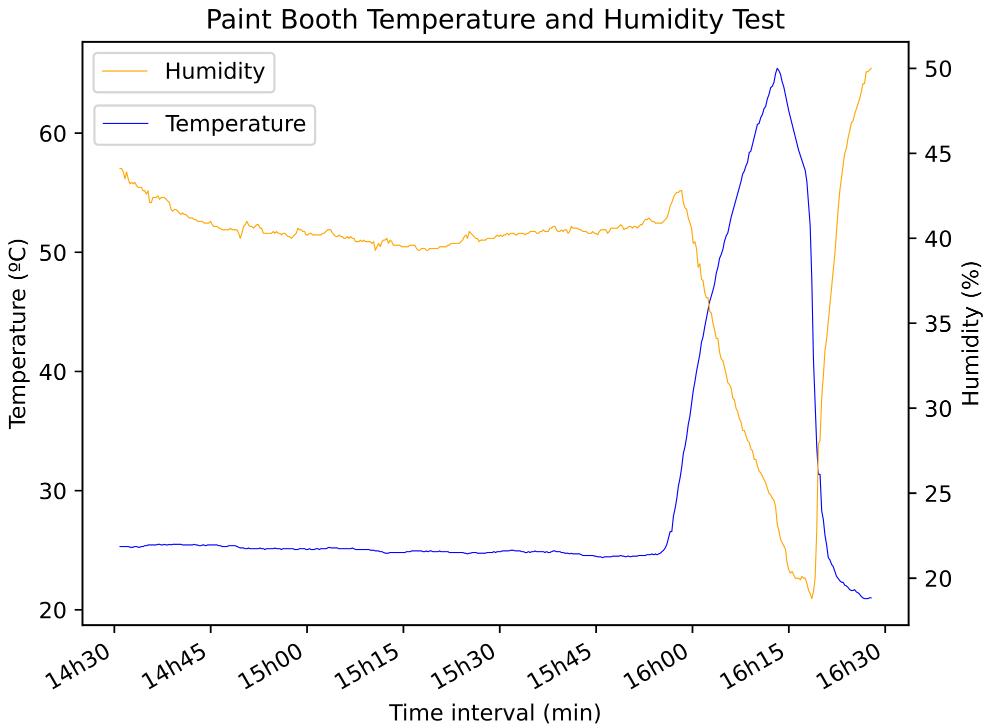


Figure 5.5: Variation of temperature and humidity values during a car painting process.

temperature higher than the one set by the worker on the paint booth.

The decreasing trend starts immediately after this temperature peak is reached and as we can see it is not necessary for the car to be baked at the maximum temperature for a certain period of time to achieve a perfect finish. From 16h30 onwards we can see the temperature sensor together with the humidity sensor returning to the values sensed before being placed in the paint booth.

5.4 Sensor Box Frequency Recording Test

The development of the frequency detection algorithm suffered some fine-tuning throughout the course of this project, caused by the difficulty in detecting the frequency of hand sanders and hammers. This difficulty is due to the amount of noise present in the data, making the algorithm less reliable than expected in a lower frequency range.

Before going to the workshop for data collection and testing, a data collection plan was developed to not only organise the process but also to understand how certain properties could affect the final results during data collection, such as:

- The car material and thickness.
- The distance between the restoration area and the sensor box.
- The machine velocities.

Therefore, in the subsection below we will separate our tests in these 3 categories and only then proceed with the validation of the algorithm in detecting frequencies of various tools. This algorithm had a continuous validation process, in the workshop environment and with machines that later would have to be identified by the system. The machines were always handled by workers from the workshop.

Over a period of a few months, more than 130 data files were collected for tools such as hammers, spot welder dent pullers, hand sanders, electric sanders and polishers. In the following section, after analysing the 3 properties mentioned above, we will show the most relevant results for each of these tools, the difficulties encountered and how they were overcome.

In all the graphs presented we show the result of the FFT algorithm having as input the acceleration data retrieved to facilitate the reader in identifying the fundamental frequency.

5.4.1 Car panel materials and thicknesses

Although it is not possible to test all existing car models and knowing in advance that each car has panels with different thicknesses, we decided to carry out this test to prove that the algorithm, even in the case of two different cars 13 years apart, is able to identify frequencies that do not affect its normal operation.

One of the cars being fixed is a 2015 BMW 425d that had a scratch along both doors on the driver's side and the other was a 2002 Mercedes E 270 that needed a bonnet restoration. In both cases, the same electric sander was used at speed 3.

One particularity that can be seen in Figure 5.6 is that the Mercedes' bonnet was separated from the car while the BMW's door repairs were done with these mounted on the vehicle, also visible in the same figure. This aspect helped us in the end because it is a common situation in the workshop and the system must be able to deal with it.

Below, in Figure 5.7 and Figure 5.8, it is shown that the frequency detected while the Mercedes was being repaired was 114 Hz and 164 Hz for the BMW. Despite there being a difference in the fundamental frequency detected, it won't affect the normal operation of the system and in both situations, the tool would be correctly identified.

This happens because when a tool is added to the system the worker runs tests to understand what the frequency of the minimum and maximum speed of the machine is on several different cars. This allows the worker to use the machine at any speed without affecting its identification, and as we can see, even on cars with different panels, we would still be able to correctly identify the tool used, whether the part being restored is on the vehicle or not.

Figure 5.6: Worker using an electric sander on a Mercedes E 270 bonnet with the BMW 425d in the background.

5.4.2 Restoration area and box distance

The second condition mentioned earlier that needed to be tested was how distance could affect the frequencies sensed by the sensor box. We knew it would be quite tricky to detect frequencies if the restoration area and the box were on opposite sides of the car or even on adjacent panels. For this reason, the focus of these tests is to understand to what extent distances can affect the identification of a tool's frequencies. One of the goals of this work is to keep the overhead low for the worker, so understanding the minimum distance between the restoration location and the box is certainly necessary.

To validate the algorithm in this kind of scenarios we collected and analysed data from tools such as electric and hand sanders at distances from 10 cm up to 30 cm with intervals of 5 cm for hand sanding and distances of 25 cm and 75 cm for electric sanders. We decided to collect more data and at closer intervals for hand sanding due to the high noise and little impact passed to the car panel by the worker. The car used for the hand sanding data collection was a Mercedes 190 SL.

In both Figure 5.9 and Figure 5.10 we plotted data from a hand sanding recorded 10 cm from the sensor box, only a few minutes apart. Several tests were performed in these conditions and most of the time the results were similar to Figure 5.9 however,

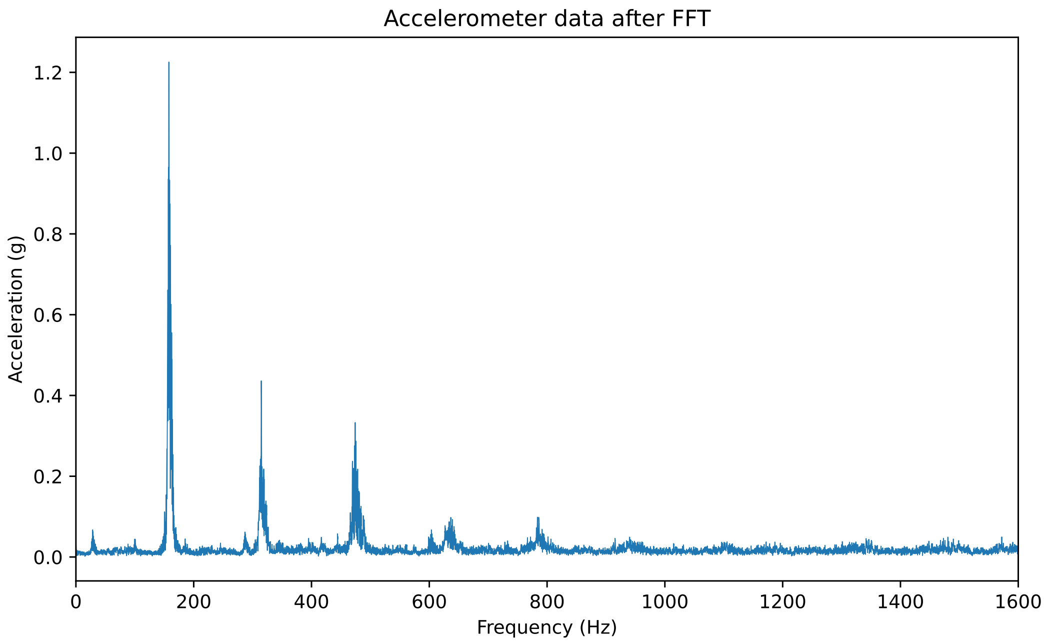


Figure 5.7: FFT data recorded on the BMW.

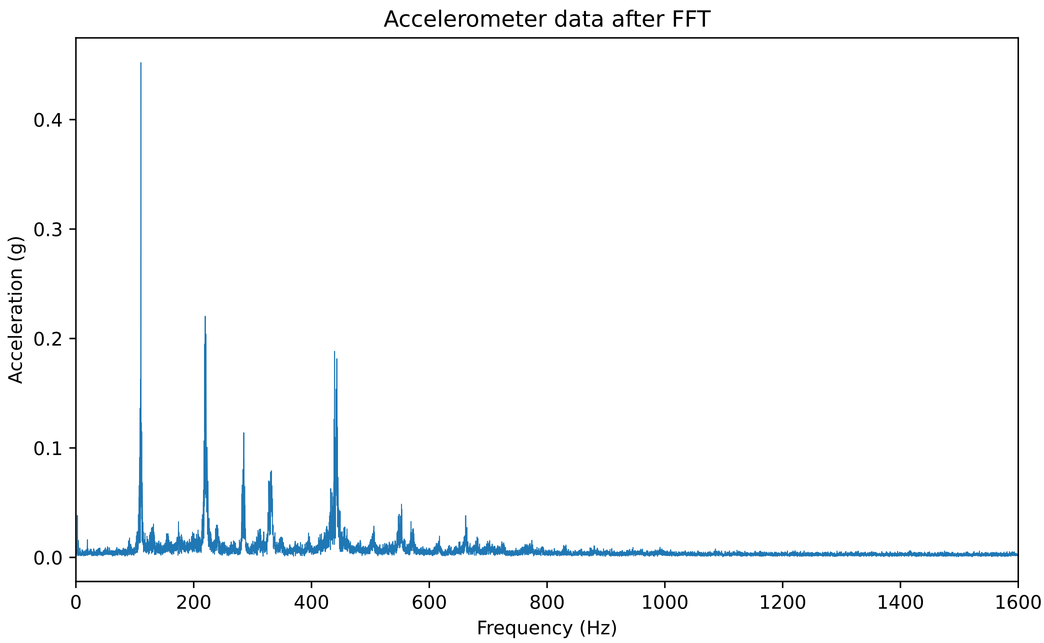


Figure 5.8: FFT data recorded on the Mercedes.

sometimes the algorithm was able to detect the low frequency of 3 and 4 Hz as we can see in Figure 5.10 (note the long blue peak near the y-axis). The latter is from a recording in the same conditions, minutes after the data from Figure 5.9 was recorded and has less noise enabling the algorithm to detect the frequency.

This issue remained throughout the development of this work in which the algorithm

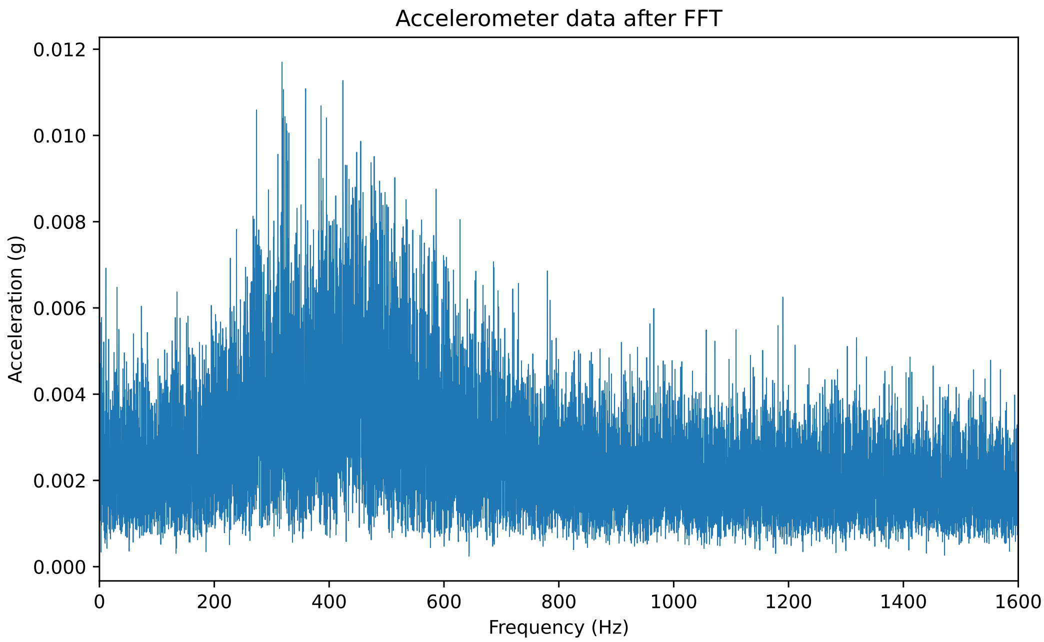


Figure 5.9: Noisy data from hand sanding 10 cm of the sensor box on a Mercedes 190 SL.

was sometimes able to detect the frequency but without the expected regularity, caused by the noise in conjunction with the distance from the box to the manual sandpaper.

As we can see in figures Figure 5.11 and Figure 5.12 the noise rises considerably with the increase of the distance between the box and the restoration area. Although the algorithm, even with low reliability, was able to identify frequencies of hand sanders this required a short distance that could in some cases affect the normal functioning of the worker.

For the electric sanders, the car used was the same BMW 425d from the previous test. Since the restoration area is generally larger and the tool is more powerful, its digital signature passes to the car panel with a greater intensity which allowed us to make the tests at a higher distance between the box and the contact point of the tool reducing the impact of its use by the worker.

We started by collecting data at a distance of 25 cm and as you can see from the graph in Figure 5.13, the algorithm is able to identify a fundamental frequency of 163 Hz. Then we increased the distance to 75 cm, allowing more space for the worker, and obtained the following data plotted in Figure 5.14. Repeating the test in the same conditions (75 cm) we obtained the results 164 Hz, 165 Hz.

Despite the good results in identifying the fundamental frequency, it should be noted that the sensor is able to identify the harmonics more easily at 75 cm than at 25 cm.

As we were successfully obtaining the fundamental frequencies with the box at 25 cm and 75 cm from the restoration area, we decided to test if the algorithm was able to detect the frequency being placed in a different (although adjacent) panel from the one being restored. The results can be seen in Figure 5.15.

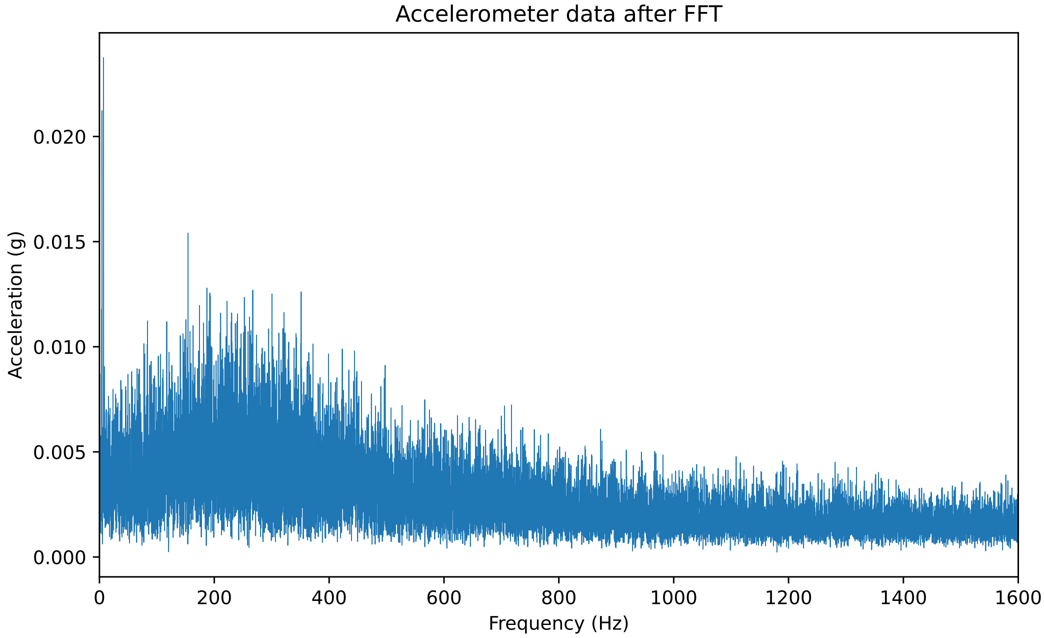


Figure 5.10: Data from hand sanding 10 cm of the sensor box on a Mercedes 190 SL.

In Figure 5.15 we can clearly identify the fundamental frequency but not so much the harmonics and as we can see there are fewer peaks when compared to the graphs with data collected in the same panel. This was expected since the car was assembled while the data was being recorded, and part of the vibrations of the electric sander ended up being absorbed by the dampers or rubbers located inside the car door.

These tests allowed us to realise the importance of the location of the box when dealing with non-electric tools, which makes it difficult to pass their digital signature through the car panel to the box. Despite some success in identifying frequencies in the case of hand sandpaper this algorithm proved not to be capable of reliable identification for this type of tool. However, we could also see that although the distance affects the identification of the fundamental frequency, if the machine is electric the algorithm is able to produce the expected results. Even though it is possible to find the operating frequency of the tool in different panels, this becomes more reliable if the box is in the same panel where the restoration is being made.

5.4.3 Tool velocities and frequencies

Another variant that could affect the identification of the fundamental frequency is the speed of the machine used in the repairs. During these worker restorations, it is not common to have the need to change the speed of the tool, in this case, the electric sanders are the only tools with different speeds. Even though it is a rare situation it can happen and to better understand this phenomenon we used a triangular tip electric sander at speeds 1 and 4.

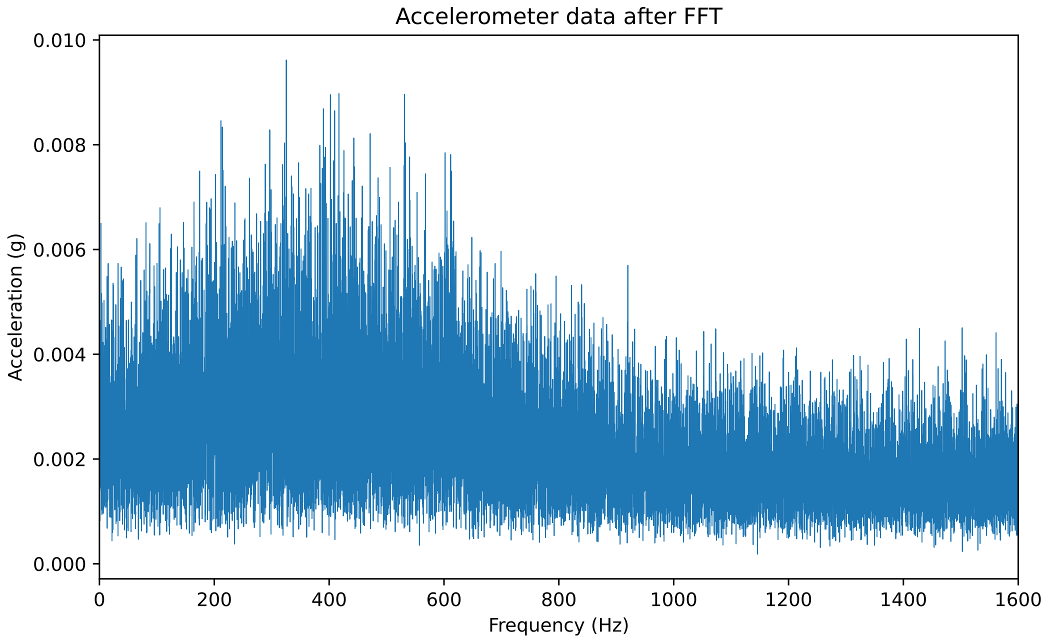


Figure 5.11: Data from hand sanding 15 cm of the sensor box on a Mercedes 190 SL.

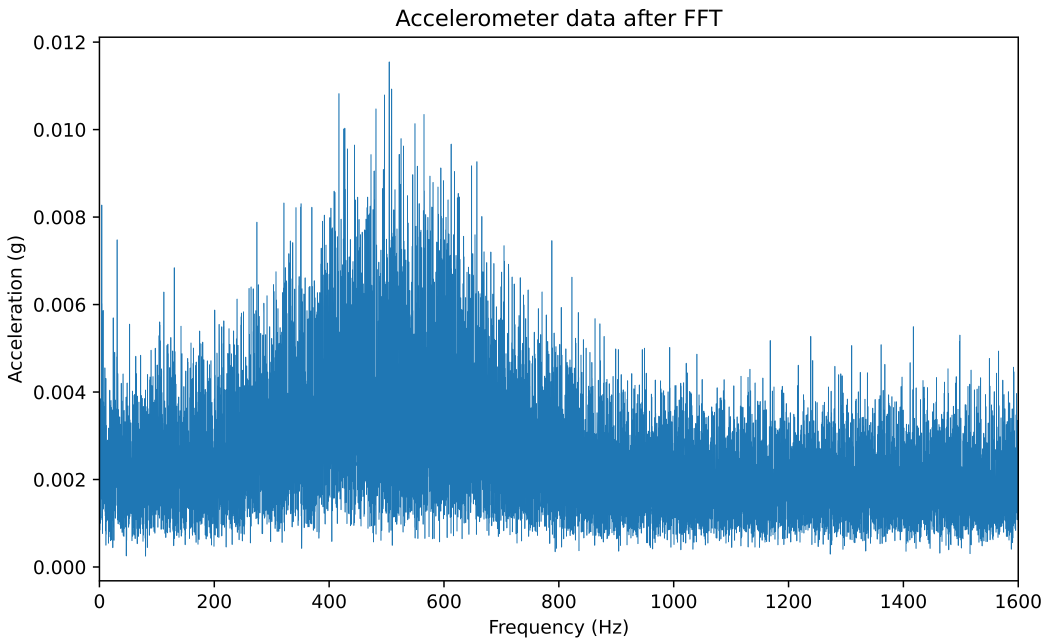


Figure 5.12: Data from hand sanding 30 cm of the sensor box on a Mercedes 190 SL.

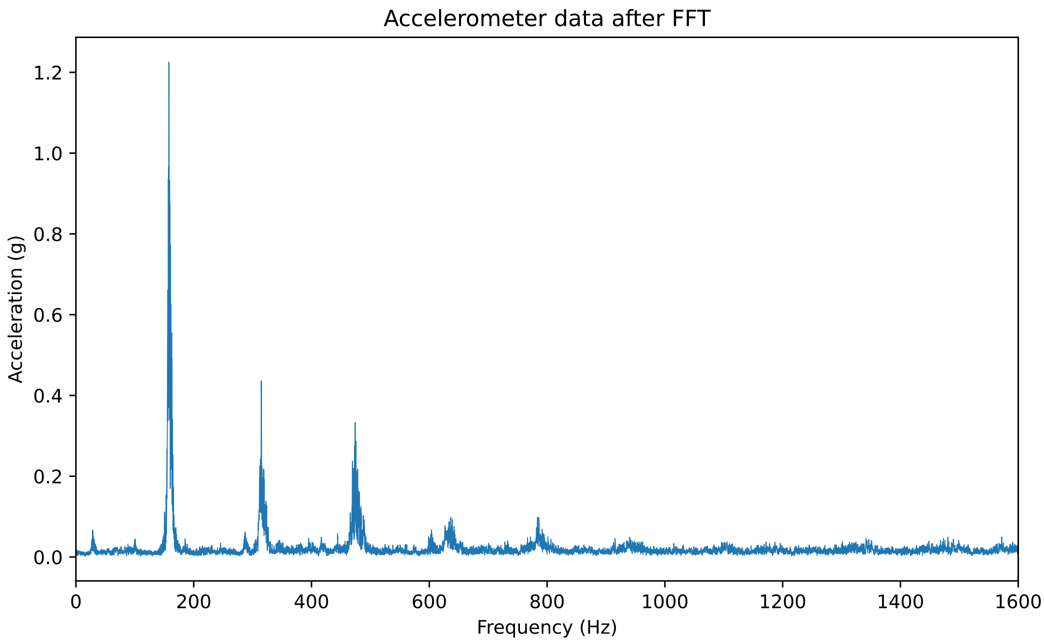


Figure 5.13: Data from an electric sander 25 cm of the sensor box on a BMW 425d.

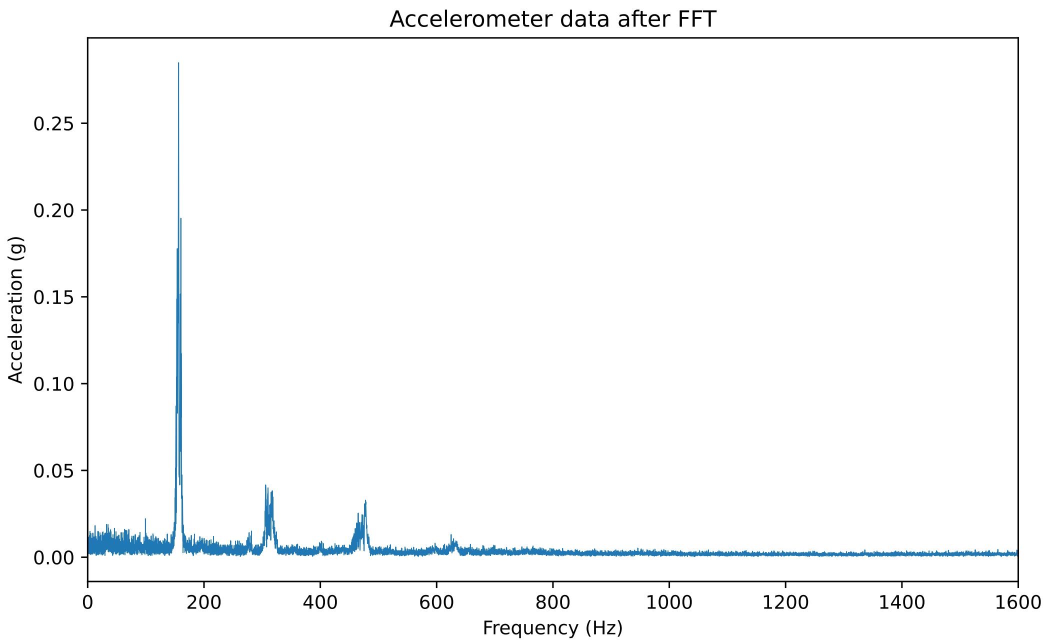


Figure 5.14: Data from an electric sander 75 cm of the sensor box on a BMW 425d.

After analysing the two graphs in Figure 5.16 and Figure 5.17 it is possible to see that the peaks, which correspond to the fundamental frequency and its harmonics, are located further to the right in Figure 5.17 that corresponds to the FFT applied to the data collected while the tool was working at speed 4.

This shift to the right proves that the frequency felt by the sensor box increases with

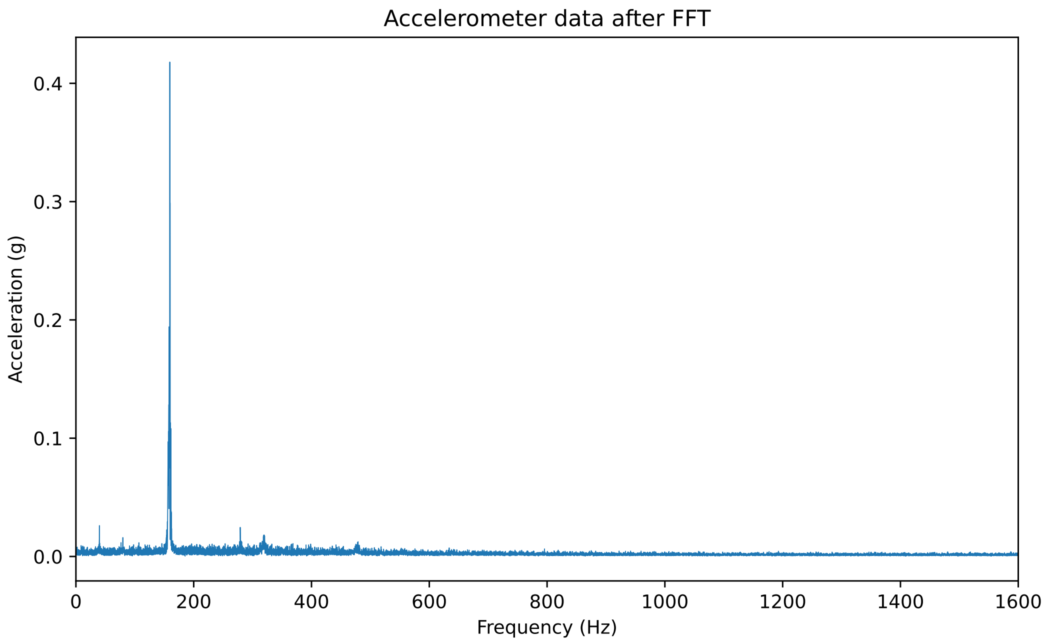


Figure 5.15: Data from an electric sander while the sensor box is located on an adjacent panel of the BMW 425d.

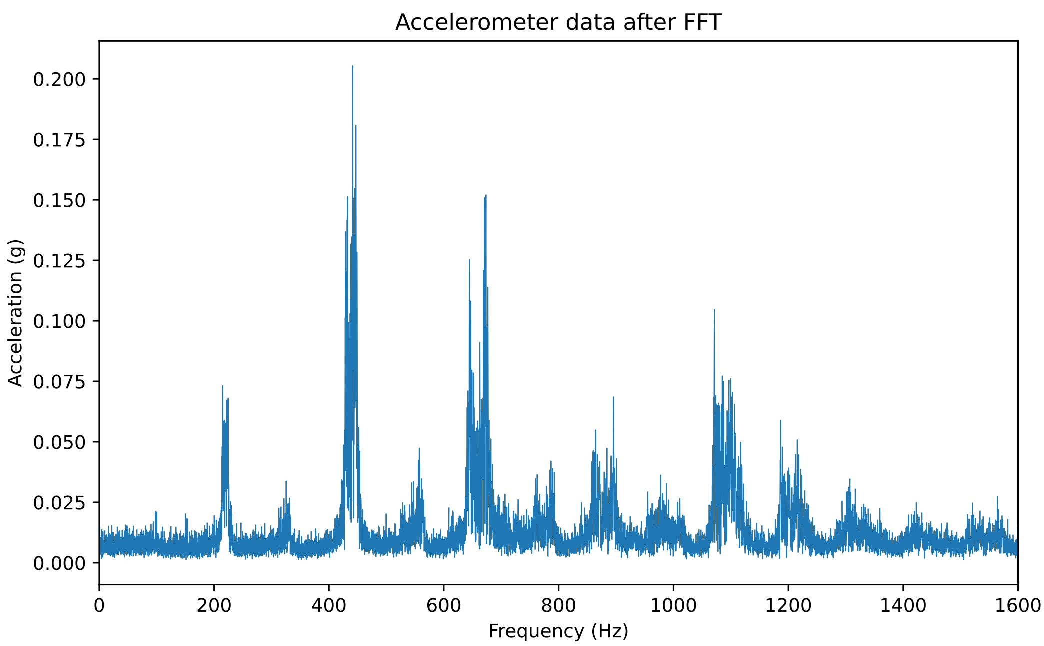


Figure 5.16: Data from an electric sander used at speed 1.

the speed of the machine, which justifies the need to identify the frequency when the machine is working at its lowest speed and at its highest speed whenever a new machine is added to the system so that it can be identified regardless of the speed at which it is being used.

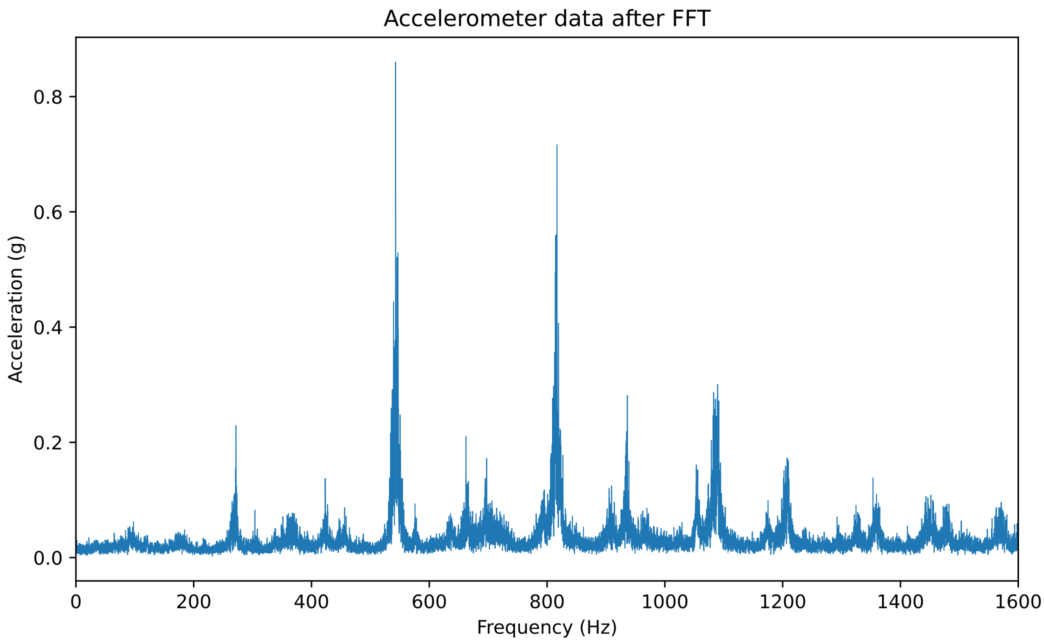


Figure 5.17: Data from an electric sander used at speed 4.

5.4.4 Workshop tools data collection

Now that we have analysed and validated the algorithm for the different scenarios that can occur during its execution we will present the results of the data collected in the workshop organised by non-electric and electric tools. The number of tests performed varies for each tool however, we will perform an overview of the algorithms successful identification of the fundamental frequency and also its difficulties in doing so.

Non-Electric Tools

As we mentioned before, the algorithm has difficulties in identifying the frequency of tools that are directly dependent on the intensity of its use by the worker. During the development of this work we studied the possibility of using the magnitude of the acceleration obtained by the FFT (y-axis of all graphs shown so far) however, these values depend on the task to be performed and on the worker himself.

In the case of the hammer, all the data was collected from a 1964 Ford Mustang, which was involved in a frontal collision that damaged the front panel on the driver's side. To straighten the panel the worker used a hammer and the results obtained after analysing the acceleration values and the results of applying FFT to that data are presented in Figure 5.18.

Due to the high intensity of the hits, it was necessary to place the box on the same panel but away from the point of impact. Despite the distance, it is possible to observe the peaks in the acceleration graph that correspond to the worker's strikes.

During the development of this project, it became evident the existence of a relationship between the magnitude obtained from the data and the task to be performed,

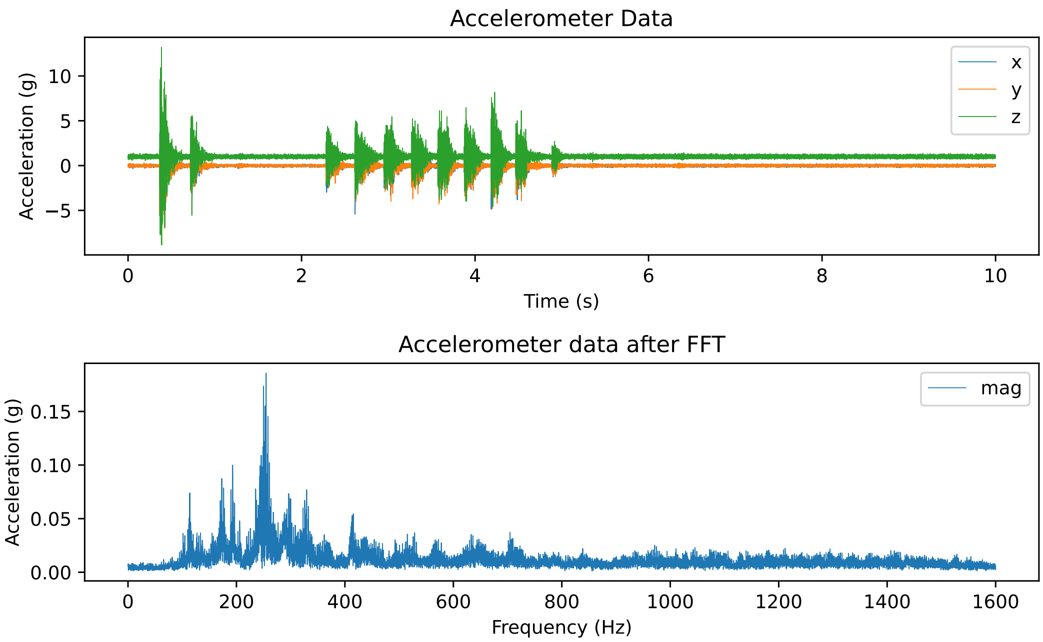


Figure 5.18: Acceleration data and the output of the FFT from a hammer.

however, and using the situation of the hammer as an example, we decided not to use the magnitude of the signal in the identification of the tool once it varies depending on the force exerted by the worker, which would lead to incorrect identification of the tasks.

As we show below in Figure 5.19 the magnitudes from these tests are similar to the magnitudes obtained in certain tests performed with spot welder dent puller, note the similarity of the maximum y-axis value, even though a different tool is used.

In this type of tools, we are not exactly looking for their frequency, however, as mentioned before, the magnitude ends up not being useful either due to the overlap of magnitudes with other tools caused by the variability of the force applied by the worker on the tool. The location of these boxes on the car also has a direct impact on the magnitudes obtained, adding yet another factor that can affect the tools' identification.

In figure Figure 5.20 we can see the worker using a spot welder dent puller while the box collects acceleration data. These tests were collected from a Volkswagen family van and such as the hammer tests we can see peaks that correspond to the worker pulling the tool in Figure 5.19.

As previously stated these two tools share the same magnitude of the signal picked up by the sensor, which causes uncertainty in their identification. Not only there is sharing of the magnitude values but it also depends on the style in which the worker uses the tool and mainly on the condition of the car before the restoration process.

During the development of this algorithm, hand sanding data from several cars was collected and although the magnitude value is much lower than the tools previously observed, as we can see in Figure 5.21, sometimes it is possible to detect frequencies caused by the movement made by the worker. However, the challenge remains due to the

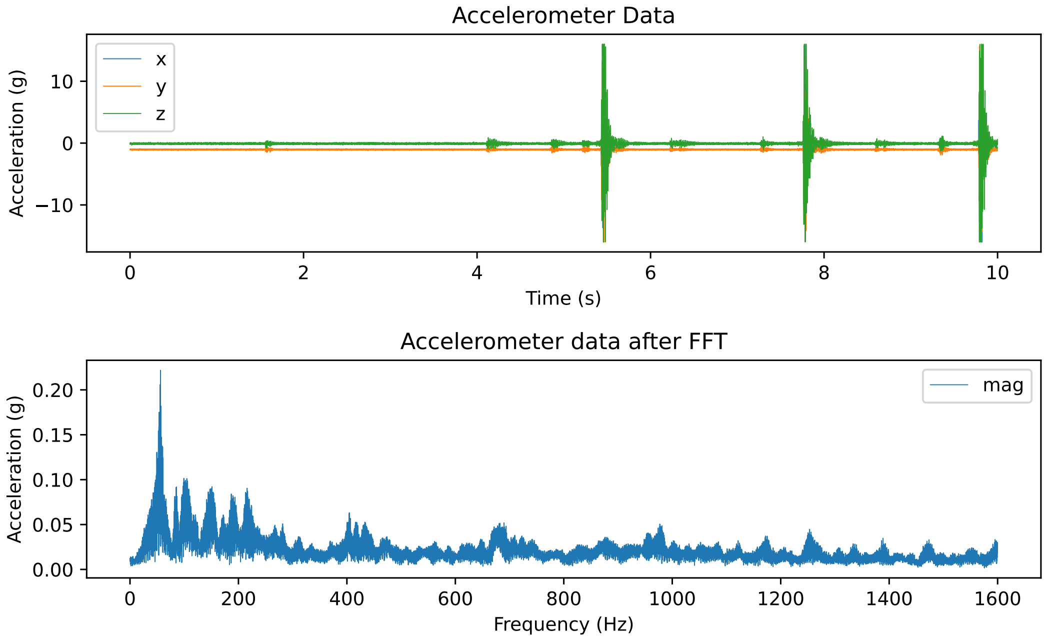


Figure 5.19: Acceleration data and the output of the FFT from a spot welder dent puller.

Figure 5.20: Worker using the spot welder dent puller.

poor ability of the algorithm to handle the variations that occur during data collection. The weak impact of sandpaper and worker on the dashboard makes it difficult to reliably and repeatedly identify its frequency.

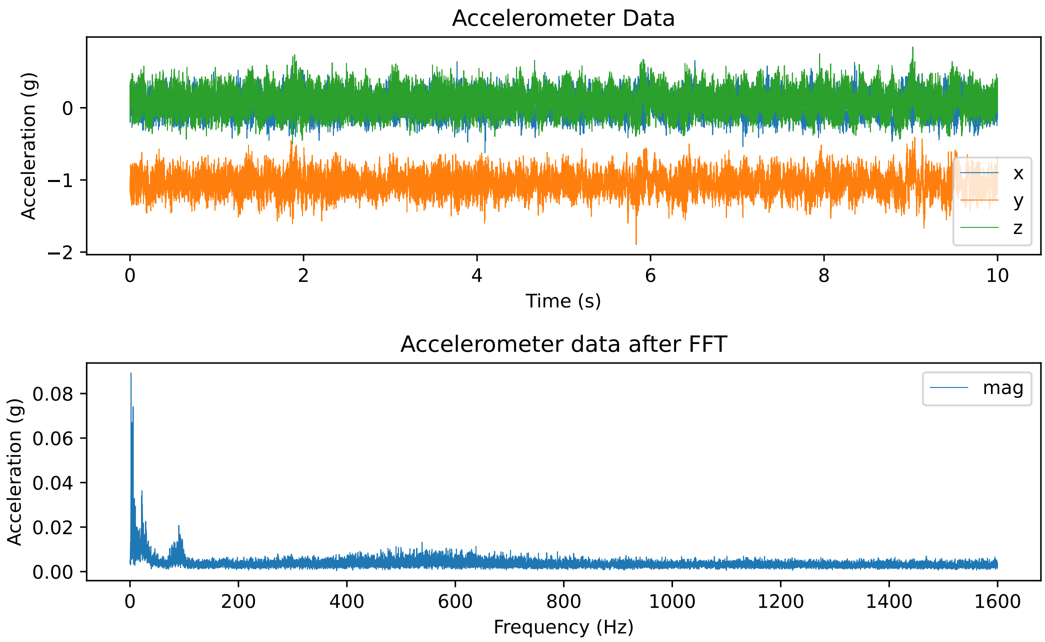


Figure 5.21: Acceleration data and the output of the FFT from a worker hand sanding.

In figure Figure 5.21 we can see the acceleration plot and the FFT result of data collected while one of the workshop workers was hand sanding a 1950s Messerschmitt KR200. As can be seen in the graph, the acceleration magnitude values are low and easily mistaken for noise. However, the algorithm is able to detect fundamental frequencies in the various measurements taken under these conditions, obtaining values between 2 Hz and 5 Hz.

Due to the inability of the algorithm to reliably identify these types of tools and to avoid erroneous detection of them, their identification was left out and reserved for future work where other techniques will have to be used for reliable and effective identification. We believe that the identification of hand sanding fundamental frequencies is possible however a better cleaning of the data needs to be done and the reliability of their detection by the algorithm needs to be improved. Future work regarding this issue will be addressed later in this paper where we propose techniques that could be used in the further development of this work.

Electric Tools

The electric tools tested in the workshop were the electric sander and the polisher. As with manual sanding, data from the electric sander was collected on several cars as seen in some of the results presented earlier. This allowed us to fine-tune and consolidate the good functioning and good results of the algorithm in identifying such tools. Data was collected from cars of the most diverse eras from the 50s to recent ones. Of these, we will show the tests performed on a 1950 Messerschmitt KR200, mentioned above, and a Range Rover. In both repairs, the same machine was used, at the same speed of 3, which made the results of both tests quite similar.

Figure 5.22: Worker using an electric sander to fix the body of a Messerschmitt KR200.

The panel of the first car was placed on top of easels, as can be seen in Figure 5.22. In all 7 data files collected under these conditions, the algorithm was always able to identify the fundamental frequencies between 132 Hz and 135 Hz. An example of the acceleration data and the result of the FFT using this data as input can be seen in picture Figure 5.23.

The same test was performed on a Range Rover's door, which was also not mounted on the car. One of the results of the test is represented in Figure 5.24 and 11 data files were taken in similar conditions and the algorithm could not identify the fundamental frequency in one of these files. In the others, it identified fundamental frequencies between 129 Hz and 133 Hz, values very similar to those detected in the Messerschmitt, which confirms the ability of the algorithm in identifying frequencies even when dealing with different cars. The differences between the two FFT graphs are most likely related to the material of the cars' panels, due to the 20 years difference between them.

Regarding the polisher, data was collected on the bonnet of an Alfa Romeo MiTo and the results were not as expected. As we can see from the graph on Figure 5.25 the polisher gives a very low FFT calculated acceleration magnitude, which is related to the lower operating frequency than that of an electric sander and the paste applied on the

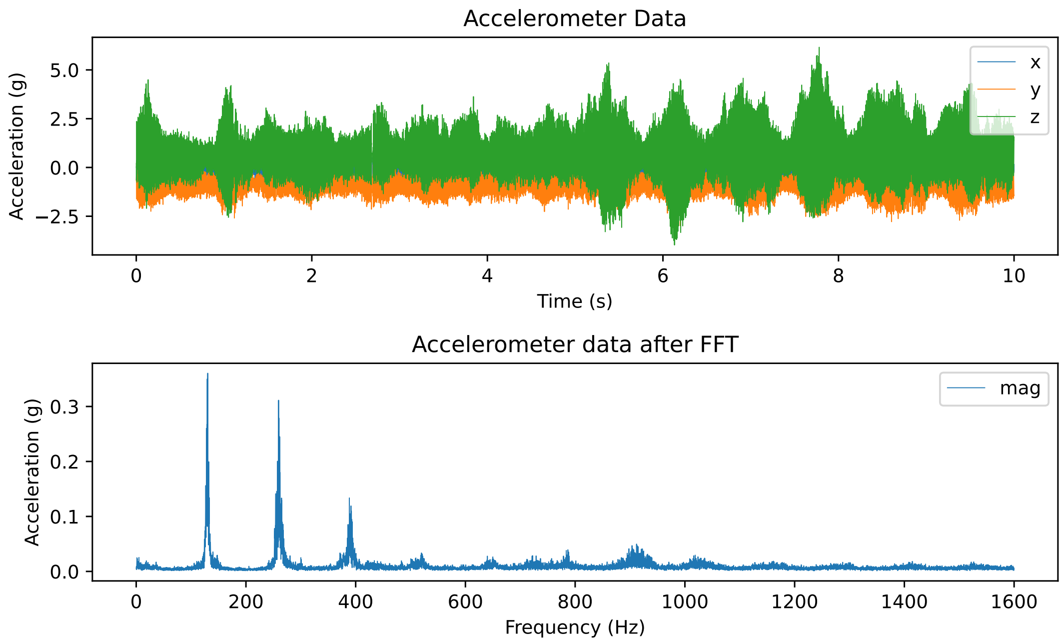


Figure 5.23: Acceleration data and the output of the FFT from a worker using an electric sander on a Messerschmitt KR200.

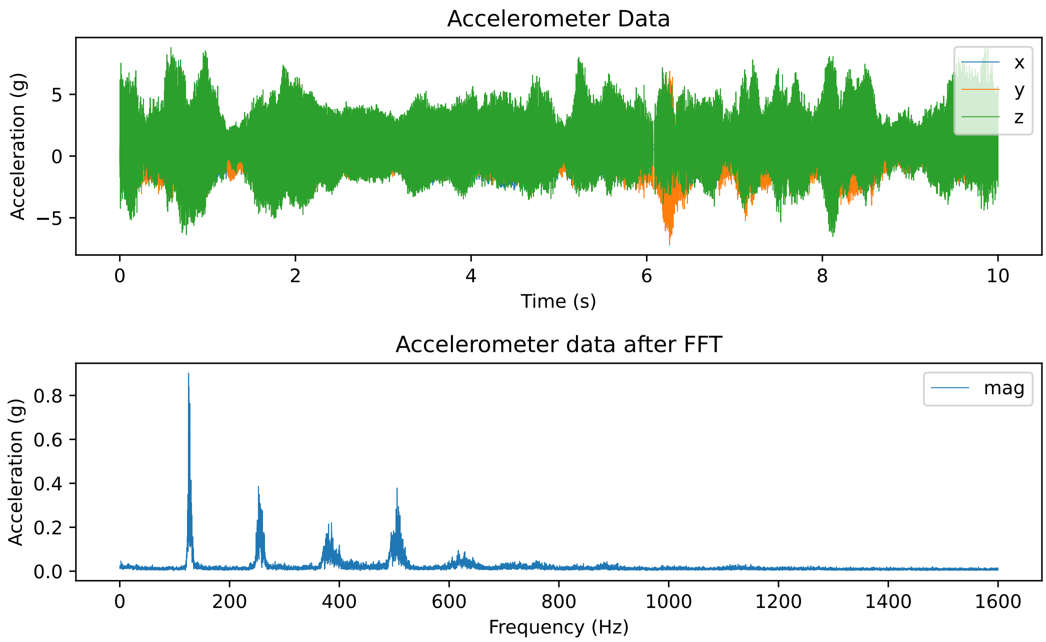


Figure 5.24: Acceleration data and the output of the FFT from a worker using an electric sander on a Range Rover door.

polisher head that will allow the polishing of the car. This paste alongside the polisher foam head makes it harder to detect its frequency. As with non-electric tools, the process of polishing a car will need another means of identification, a subject that is addressed

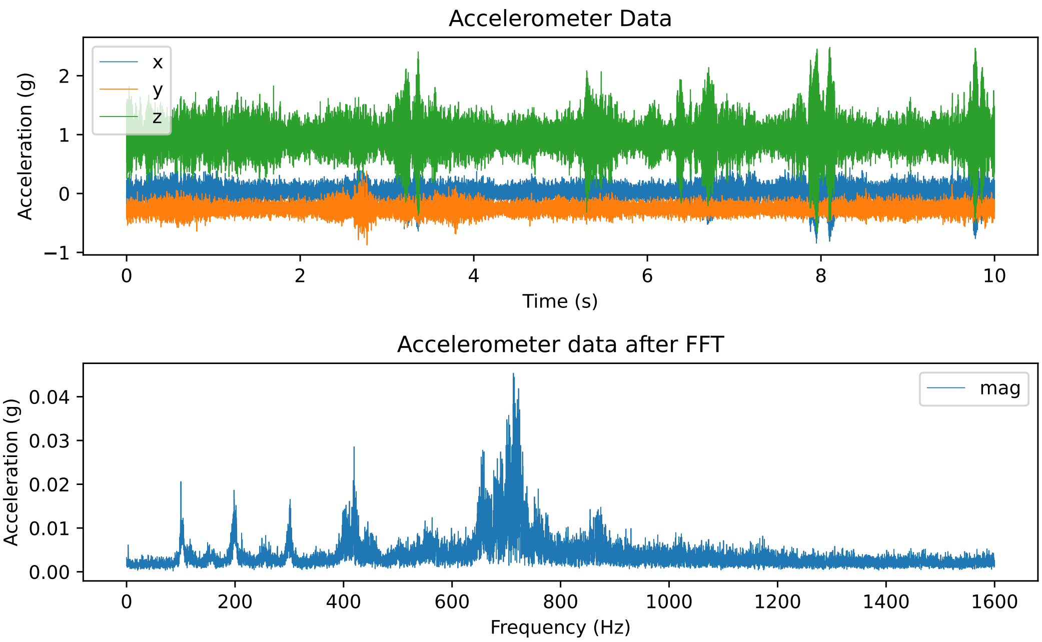


Figure 5.25: Acceleration data and the output of the FFT from a worker using a polisher on an Alfa Romeo bonnet.

later in this paper aiming at the future work of this project.

5.5 Indoor Location Test

As mentioned in section 4.5 where we presented this system, it was not possible to perform this test in the workshop due to its reconstruction, however, while the current building is being rebuilt, a second space was rented to place cars that needed to be restored and to allow work that was already active to continue. This new building was used to validate this indoor location system using BLE beacons.

Next, we show the layout of the workshop where this system will be implemented and the zones delimitation. As we can see in Figure 5.26 there are 3 zones that are relevant to our system, the bodywork restoration zone in red, the painting zone in green and the mineral blast zone in blue.

From the floor layout in Figure 5.26 we can see that the bodywork restoration zone in red occupies several sections on the left and right. In each of these left sections will be a worker restoring a car, it happens that all tasks done in these areas are tasks that can be identified by the accelerometer, while the temperature and humidity sensor alone do not allow us to detect which tool is being used. For this reason, although each section has its own zone with its own beacons, they will all correspond to a single zone, which the data processing algorithm running on the cloud already knows to look for tools that may have been used by the workers.

In Figure 5.26 it is also possible to observe the painting zone that corresponds to the

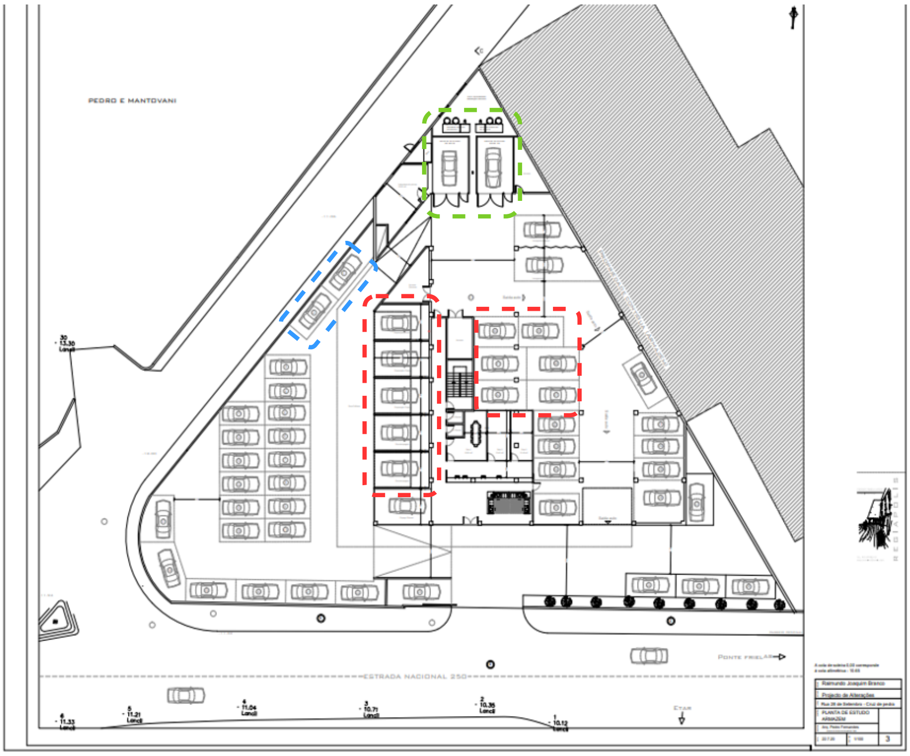


Figure 5.26: The layout of the workshop being rebuilt.

location of the paint booth in green and the mineral blasting zone in blue. The latter will be identified only by its location, whereas the painting zone will be identified through the location of the booth and the temperature and humidity values, which as we saw previously in the temperature tests allow the identification of this process.

The need to use the Open Weather API [7] has also already been mentioned. This is because, in each of the sections of the left red zone, the worker may choose to do their job in the open air and in these cases, as the beacons (which also have temperature sensors) are inside the workshop, they would not be useful in confirming that the sensor box is effectively in this zone due to the temperature difference that can be felt inside and outside of the workshop, hence the need for this API.

The temperature values coming from the beacons and/or the API serves as confirmation that the box is in the expected zone. This reduces misidentification of events even if there is some confusion in determining the sensor box location. The use of temperature values prevents situations where the box is in the red bodywork zone but the beacons picked up by the sensor box point to the paint zone in green, in which case the temperature will prevent the latter from being detected by mistake.

Another possible scenario is that the sensor box is in the green painting zone but the information from the beacons determines that its location is in another zone, in this case as we already know the temperature during the painting and baking of a car we can use

[7] https://openweathermap.org/ Last accessed on: 27/11/2021

that information to our advantage and avoid having the algorithm try to look for what tools are being used when the box is located inside the paint booth.

The use of temperature allows us to build a more robust solution and, with the use of the sensors present in the sensor box, to minimise the misidentification of events. However, although useful, it has the drawback of not being able to accurately identify the moment when the painting process started. Since any painting event is confirmed by the location of a sensor box, temperature and humidity values, to be able to identify without any error that a car has been painted, these temperature and humidity values must be high and low enough respectively so that they are not confused by temperature and humidity values outside the paint booth.

By performing the test inside the paint booth we know that the worker paints the car at 25°C, justifying the need to choose temperature and humidity values that cannot be detected outside the paint booth, which in turn reduces the duration of the painting event.

This makes it impossible to incorrectly identify the painting event, under the penalty of not being able to accurately identify the duration of the entire process.

To be able to detect the whole painting procedure from its very beginning, we would have to reduce the minimum temperature necessary for its detection to below 25ºC, a temperature easily reached in Portugal during spring and summer. This approach would increase the cases of mistaken identifications whenever a box incorrectly identifies a paint zone. In the case of no mistakes in zones detection, this approach would be able to identify the painting event and correctly obtain information of its beginning and end.

As the indoor location system inside the workshop may allow the incorrect identification of a zone, the implementation of this approach would result in rather low system performance and potential loss of information about the restoration process of a car. Overall, the loose notion of when the painting process effectively started compensates because it allows us to say with certainty that a car was painted and to avoid situations in which tools and processes are no longer identified because the information from the beacons states that the box is inside the paint booth when it is not, which negatively affects the goal of this work.

As we will show below, it is possible for the system to detect wrong zones despite the high redundancy of the beacons and the use of techniques such as averaging the distances between several beacons from each of the detected zones, which allows us to continue to have good results despite the failure of some beacons in a zone. As mentioned above, this system was validated in another building and to test it intensively, we established 7 zones instead of 3 in a smaller space when compared to the new workshop building. The goal of this test was to understand the algorithm's ability to correctly identify the different zones even when they were adjacent to each other.

In Figure 5.27 we can see the plan of the building, the layout of the beacons inside it as well as the corresponding zones inside the red rectangles. A total of 20 beacons, represented in blue, were used and each zone was identified with three with the exception

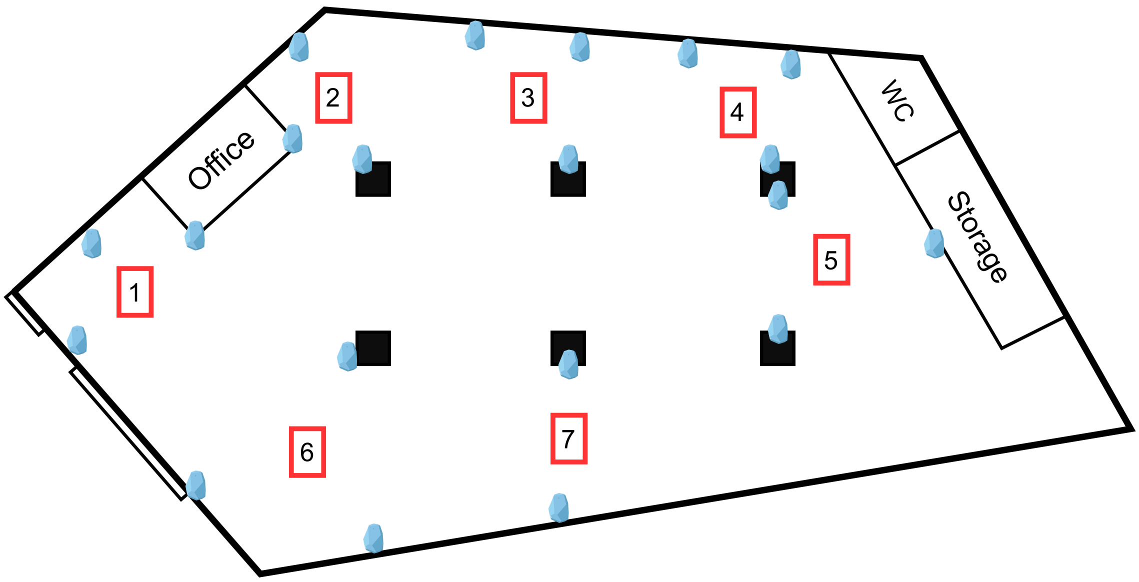


Figure 5.27: The layout of the rented workshop where the ongoing car restorations took place.

of zone 7, which only had two beacons, a factor that did not affect its identification.

The purpose of placing only 2 beacons in zone 7 was to test the identification of zones 6 and 1 and to understand if, despite the proximity, it would be possible to obtain positive results. After the beacons were distributed throughout the building two tests were performed in which the sensor box was constantly receiving signals from the beacons but would only return the location inferred through the signals received by it every 15 seconds. During the test, the sensor box remained in each zone between 1 minute and 1 minute and 30 seconds to obtain several inferences to test the reliability of the algorithm.

In the graph of figure Figure 5.28 we can see the results of the first test performed. In this test, the box circulated clockwise throughout the workshop, starting in zone 1 and finishing in zone 6, passing first through zone 7.

As shown in the graph of Figure 5.28, in this test the sensor box makes wrong location inferences for 3 times, one of them a little after 12h08 and the other two a little before 12h10. The first wrong inference occurred during the change from zone 5 to 7, the box passed through the central corridor of the workshop which made it receive signals from the beacons of zone 3 in good conditions. The next two wrong inferences occurred when zone 6 limits were being tested, which caused the box, for a moment, to make the decision of being in zone 1.

In the second test performed there are more incorrect inferences as shown in Figure 5.29 however these happen in the closest zones, zone 3 and 4. This type of situation is not possible to occur in the bodywork zone because even though each section of the worker contains beacons, these will all be from the same zone. However, as mentioned earlier it is necessary to perform a very similar test in the workshop where this system will be implemented.

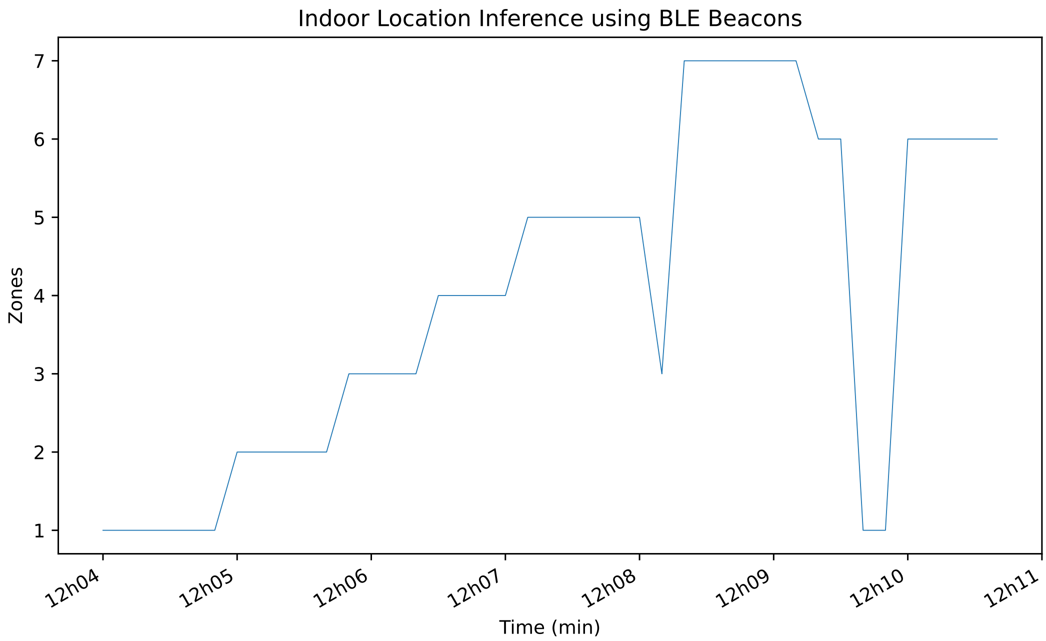


Figure 5.28: Zones inferred by the BLE signals received by the sensor box in the first test.

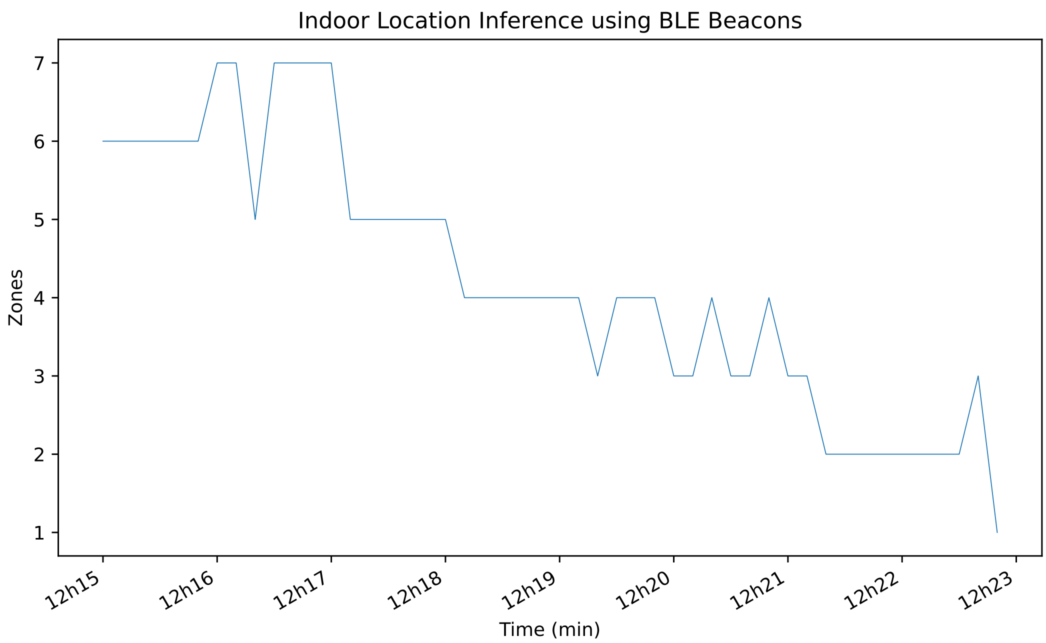


Figure 5.29: Zones inferred by the BLE signals received by the sensor box in the second test.

5.6 Process Identification Algorithm Test

One of the most important components of this work is the function that allows car restoration tasks to be identified based on sensor data and the location of a sensor box in the workshop.

From the beginning, we have always known the importance of workshop indoor location because it allows us to focus on sensor data knowing in advance what kind of work is done in each of the zones and taking advantage of the fact that the distribution of zones and the tasks done in them remain static for the lifetime of a workshop or until a redesign occurs. As we explained during the workshop indoor location system, it can be faulty, hence the need to deal with misidentification situations using temperature and humidity values.

In the appendix, we have Figure A.1 representing the algorithm's execution flowchart. As it was already possible to understand, this function was developed using a serverless methodology, which means that it doesn't need to be constantly active on a server, allowing it to be activated only when a new data file arrives at the *S3* bucket from any sensor box.

Features such as function scaling are more difficult to test precisely because of this methodology. Nevertheless, a test was made in which 10 input files were stored in the *S3* bucket at the same time and the *Lambda* function was able to scale automatically executing all of them in just 2 seconds. Such a situation is rare to happen, considering the small number of devices expected to be implemented in the workshop. For solutions that may require more sensor boxes running in parallel, further study of the scaling capability of *Amazon's Lambda* function is required.

As this *Lambda* function is activated whenever a new data file arrives at the *S3* bucket the best way found to validate this function was to create all the different possibilities of data files that can be produced by a sensor box. Below we'll start by presenting each of the test files used, the results obtained by the function and an explanation of its execution. For some longer test files, it was necessary to place them in the appendix of this document, the remaining are placed throughout the explanation for an easier understanding.

Empty file sent by the sensor box:

An empty test file is a scenario that although rare, has to be tested. As an empty file does not contain any relevant information that could be useful to the user, this file is not accounted for and therefore the algorithm does not produce any output.

Consecutive events of the same type:

One of the most important features when processing data coming from an IoT device is the ability to filter and condense the data as much as possible. Each identified event has a start and end timestamp and in this file we have several events of the same type to test the algorithm's ability to group all the similar events that happen successively into a single event, saving the start timestamp of the first event and the end timestamp of the last event of the same kind. This allows us to reduce the size of the processed data file and eases the reading of the different detected events.

In this test file, which due to its size is available in the appendix, we have several consecutive mineral blast events, which produces a single event in the processed data file

```
[
  {
    "DeviceId": "SensorBox_01",
    "Car": "9312AA",
    "Location": 3,
    "Event": ["Mineral Blast"],
    "StartTimestamp": 1636213943,
    "EndTimestamp": 1636215214
  }
]
```


Listing 1: Output file for testing the process identification algorithm with respect to event grouping.

with the timestamps duly updated, as we can see below in Listing 1.

Intermittent tool usage:

In addition to the consecutive identification of the same type of event, it is common for the workers in the workshop to alternate the tools used, in this case, what we want is the same result as the previous test but without losing the information of the event if they are new tools detected in between. This means that in the processed data file contents shown in Listing 2, each event starts when the previous one finished, sharing the same timestamp. Once again, given the size of the input file, this can be found in the appendix.

In the processed data file we can see a mineral blast event followed by a paint event. As it is possible to observe in this test file there are 5 entries stored by the sensor box. The first three are mineral blast events and therefore there is an aggregation of their timestamps, in between, there is an entry where it is not possible to identify an event and finally a painting event. As we can see as there is only one painting event the start and end timestamp is the same. It should be noted that the entries that do not translate into events are discarded and their timestamp does not affect the start and end time of the previous and following events.

In all the following test files we test the situations where vibrations are both detected and not detected in all three zones of the workshop. As in the previous tests, we used files with multiple entries and the grouping of events of the same type, the next files only have one entry, so in case output files are created, there is only one event identified resulting in the same start and end timestamp. In a normal situation like the previous tests, multiple events would be detected, each with its own timestamp.

```
[
  {
    "DeviceId": "SensorBox_01",
    "Car": "9312AA",
    "Location": 3,
    "Event": ["Mineral Blast"],
    "StartTimestamp": 1636213943,
    "EndTimestamp": 1636214462
  },
  {
    "DeviceId": "SensorBox_01",
    "Car": "9312AA",
    "Location": 2,
    "Event": ["Paint"],
    "StartTimestamp": 1636214961,
    "EndTimestamp": 1636214961
  }
]
```


Listing 2: Output file for testing the process identification algorithm with respect to intermittent tool usage.

No frequencies identified in the bodywork zone:

With the test file present in Listing 3 we validate the execution of the algorithm when the entry of the input file, despite having information from the beacons that would indicate the bodywork zone, does not have any frequency value that would allow the identification of a tool. This happens when the worker stops to check the restoration process or simply leaves the scene. In this scenario, the algorithm discards the entry because it cannot identify any tool and as the test file only had one entry and this was discarded, the algorithm does not produce an output file.

```
[
  1636727056,
  {
    "DeviceId": "SensorBox_01",
    "Car": "9312AA",
    "Timestamp": 1636727055,
    "Temperature": 16.375,
    "Humidity": 46.1,
    "VibrationPeaks": "[]",
    "Beacons": {
      "b4c8c0b52049e4de": {
        "timestamp": 1636727052,
        "battery": 36,
        "rssi": -77,
        "temperature": 19
      },
      "9fcccaab3f956f66": {
        "timestamp": 1636727053,
        "battery": 48,
        "rssi": -64,
        "temperature": 18.875
      },
      "a58db253434c6759": {
        "timestamp": 1636727055,
        "battery": 40,
        "rssi": -75,
        "temperature": 19.1875
      }
    }
  }
]
```


Listing 3: Input file for testing the process identification algorithm with respect to the lack of frequencies in a bodywork zone.

Frequencies identified in the bodywork zone:

A more common scenario is the existence of frequency values when the box is in the bodywork zone and it is precisely this situation that the test file in Listing 4 validates.

As we can see from the output file presented in Listing 5 the algorithm identifies two possible tools that may have been used by the worker. This type of situation is common because several tools can share the same range of operating frequencies. Although this situation is common and since workshops usually have several tools of the same type, it became imperative to return only the type of the tool. As we can see in this case, there are several electric sanders that work in the frequency of 206 Hz identified by the sensor box, however, only the tool type was stored as a possibly used tool.

```json
[
  1636727056,
  {
    "DeviceId": "SensorBox_01",
    "Car": "56AA24",
    "Timestamp": 1636727055,
    "Temperature": 16.375,
    "Humidity": 46.1,
    "VibrationPeaks": "[206]",
    "Beacons": {
      "b4c8c0b52049e4de": {
        "timestamp": 1636727052,
        "battery": 36,
        "rssi": -77,
        "temperature": 19
      },
      "9fcccaab3f956f66": {
        "timestamp": 1636727053,
        "battery": 48,
        "rssi": -64,
        "temperature": 18.875
      },
      "a58db253434c6759": {
        "timestamp": 1636727055,
        "battery": 40,
        "rssi": -75,
        "temperature": 19.1875
      }
    }
  }
]
```


Listing 4: Input file for testing the process identification algorithm with respect to the frequencies detected in the bodywork zone.


```json
[
  {
    "DeviceId": "SensorBox_01",
    "Car": "56AA24",
    "Location": 1,
    "Event": ["Electric Sander", "Polisher"],
    "StartTimestamp": 1636727055,
    "EndTimestamp": 1636727055
  }
]
```


Listing 5: Output file for testing the process identification algorithm with respect to the frequencies detected in the bodywork zone.

No frequencies identified in the mineral blast zone:

With the test file shown in Listing 6 we can validate the algorithm when the sensor box is in the mineral blast zone but no vibration is detected. As the detection of this event is based only on location we can see the event detected in the processed data file in Listing 7.


```json
[
  1636202314,
  {
    "DeviceId": "SensorBox_01",
    "Car": "9312AA",
    "Timestamp": 1636213943,
    "Temperature": 21.4375,
    "Humidity": 35.4,
    "VibrationPeaks": "[]",
    "Beacons": {
      "0d5c5b6ed855dbbc": {
        "timestamp": 1636213941,
        "battery": 59,
        "rssi": -73,
        "temperature": 24.75
      },
      "fc337481622c4dc8": {
        "timestamp": 1636213930,
        "battery": 99,
        "rssi": -76,
        "temperature": 24.875
      },
      "4b528a07c6810670": {
        "timestamp": 1636213933,
        "battery": 100,
        "rssi": -78,
        "temperature": 25.9375
      }
    }
  }
]
```


Listing 6: Input file for testing the process identification algorithm with respect to the lack of frequencies detected in the mineral blast zone.

```json
[
  {
    "DeviceId": "SensorBox_01",
    "Car": "9312AA",
    "Location": 3,
    "Event": ["Mineral Blast"],
    "StartTimestamp": 1636213943,
    "EndTimestamp": 1636213943
  }
]
```

Listing 7: Output file for testing the process identification algorithm with respect to the lack of frequencies detected in the mineral blast zone.

Frequencies identified in the mineral blast zone:

The test in Listing 8 serves to validate the algorithm when frequencies are detected in the mineral blast zone. The correct behaviour of the algorithm is to identify the mineral blast event by location, despite the detection of frequencies that could be used to identify a particular tool if another location was inferred. The contents of the output file can be found at Listing 9.

```
[
  1636202314,
  {
    "DeviceId": "SensorBox_01",
    "Car": "07DF45",
    "Timestamp": 1636213943,
    "Temperature": 22.4375,
    "Humidity": 35.4,
    "VibrationPeaks": "[213]",
    "Beacons": {
      "0d5c5b6ed855dbbc": {
        "timestamp": 1636213941,
        "battery": 59,
        "rssi": -73,
        "temperature": 24.75
      },
      "fc337481622c4dc8": {
        "timestamp": 1636213930,
        "battery": 99,
        "rssi": -76,
        "temperature": 24.875
      },
      "4b528a07c6810670": {
        "timestamp": 1636213933,
        "battery": 100,
        "rssi": -78,
        "temperature": 25.9375
      }
    }
  }
]
```


Listing 8: Input file for testing the process identification algorithm with respect to the frequencies detected in the mineral blast zone.


```
[
  {
    "DeviceId": "SensorBox_01",
    "Car": "07DF45",
    "Location": 3,
    "Event": ["Mineral Blast"],
    "StartTimestamp": 1636213943,
    "EndTimestamp": 1636213943
  }
]
```


Listing 9: Output file for testing the process identification algorithm with respect to the frequencies detected in the mineral blast zone.

No frequencies identified in the paint zone:

The purpose of the input test file presented in appendix is to validate that the algorithm proceeds with the identification of a car's paint job, even though no vibrations are detected by the sensor box. As we can see in the input file, the temperature and humidity values validate the occurrence of the event together with the information from the beacons that identify the painting zone. In this test, there are two entries and only the first entry is valid due to these same values, the second entry has a temperature value that despite being high, is still not enough to identify the painting process.

The second entry is an example of an input that can occur during the car painting process, in which the temperature inside the paint booth is constant at 25ºC, but to minimise the occurrence of wrong identifications, it was decided to raise the temperature and lower the minimum humidity for an entry to be considered as a painting event. In the processed data file in Listing 10, only the first entry is considered as an event despite the same information from the beacons in both entries.


```
[
  {
    "DeviceId": "SensorBox_01",
    "Car": "9312AA",
    "Location": 2,
    "Event": ["Paint"],
    "StartTimestamp": 1636214538,
    "EndTimestamp": 1636214538
  }
]
```


Listing 10: Output file for testing the process identification algorithm with respect to the lack of frequencies detected in the paint zone.

Frequencies identified in the paint zone:

With the test file in Listing 11 we validate the correct execution of the algorithm even when frequencies are identified in a paint zone. As the algorithm identifies the zone as being a painting zone it automatically focuses only on the temperature and humidity values to identify the event, as can be seen in the processed data file in Listing 12.


```json
[
  1636202314,
  {
    "DeviceId": "SensorBox_01",
    "Car": "07DF45",
    "Timestamp": 1636214538,
    "Temperature": 55.6875,
    "Humidity": 36.1,
    "VibrationPeaks": "[206]",
    "Beacons": {
      "0d5c5b6ed855dbbc": {
        "timestamp": 1636214521,
        "battery": 59,
        "rssi": -73,
        "temperature": 24.5625
      },
      "fc337481622c4dc8": {
        "timestamp": 1636214531,
        "battery": 99,
        "rssi": -69,
        "temperature": 24.75
      },
      "4b528a07c6810670": {
        "timestamp": 1636214533,
        "battery": 100,
        "rssi": -66,
        "temperature": 25.3125
      },
      "d5c87a75d12c7cbd": {
        "timestamp": 1636214528,
        "battery": 99,
        "rssi": -69,
        "temperature": 25.25
      }
    }
  }
]
```


Listing 11: Input file for testing the process identification algorithm with respect to the frequencies detected in the paint zone.

```
[
  {
    "DeviceId": "SensorBox_01",
    "Car": "07DF45",
    "Location": 2,
    "Event": ["Paint"],
    "StartTimestamp": 1636214538,
    "EndTimestamp": 1636214538
  }
]
```


Listing 12: Output file for testing the process identification algorithm with respect to the frequencies detected in the paint zone.

5.7 Web Application Tests

One of the main requirements of this work requires the validation of the web application to prove its capabilities to monitor and control the system including the sensor boxes.

The act of testing a system only achieves all its goals if extensive test scenarios are developed for all components of the application. Due to its size, it was not possible to test the backend functions, but by developing test scenarios, we tried to achieve the maximum number of user stories that were defined at the beginning of the development of this work and in this way test both the front end and the back end together, as well as the web application usability.

To prove that the application meets the customers' expectations, three test scenarios of the most common situations in a workshop were created. The user stories defined before the development of the web application (presented in section B.1) determined the functionalities of the system and facilitated its validation because it was possible to use them in the test scenarios.

Since the web application serves to monitor and control an IoT system and given that the workshop workers do not yet have the technical knowledge to be able to use this application, a pilot test was made with participants that have a Computer Science background to obtain feedback that is not affected by the natural complications of users who do not have this kind of technical understanding.

These tests were made to have an initial validation of the system, however, since the population is low and skilled it can be difficult to "identify a statistically significant difference if a difference truly exists" [6].

Nonetheless, it is hoped that future tests will be conducted with a broader type of participants to obtain more tangible results. The population of participants can be found in online groups and committees related with IoT and Industry 4.0, such as Special Interest Groups (SIG) from IEEE.

Examples of such SIG are the Cloud Networking group, the IoT group (as can be seen here) or the Industrial IoT Networks group. LinkedIn [8] groups of other SIG (such as these) related to these topics can also be interesting ways of finding participants for these tests.

Another interesting group specially created towards systems usability and design is the Special Interest Group on Design of Communication (SIGDOC) from Association for Computing Machinery (ACM) which includes members such as technical communication professionals, usability specialists, software engineers, system developers, web designers among others, despite their main focus on design, a combination of participants from several of these groups would improve the web application and overall interaction with the system.

Even though the population of the surveys was low, with these usability tests, it is possible to make an initial evaluation and possibly improve certain aspects of the web application before actually presenting it to the workshop workers who will use it. To better prepare the workers this system presentation will always need to be accompanied by a more technical explanation about the technologies used.

To achieve the maximum functionality in these three scenarios we decided to divide them into three sections: the addition of tools to the system, the monitoring of the system by the application and the control of sensor boxes. Before the scenarios were introduced, the participants were given 5 minutes to experiment with the web application.

At the end of each of the tests, a System Usability Scale (SUS) survey was completed, which allows us to measure the usability and learnability of a system. This survey initially presented by John Broke in 1986 consisted of 8 questions to evaluate the usability of a system and 2 questions to evaluate the user's ability to learn how to use a system. All answers to these 10 questions have a scale of 1 to 5, where the first stands for *Strongly disagree* and the last for *Strongly agree*.

After the answers, it is possible to get a result between 0 and 100 with increments of 2.5 [20]. A grading scale for SUS surveys is represented in Figure 5.30. Even though Lewis et al. defends that "it is becoming a common industrial goal to achieve a SUS of 80 as evidence of an above average user experience", we decided to consider an acceptable grade for SUS score values above 75 given the early stage of our web application. [19]

In addition to the survey presented above, participants were also requested to answer another survey to assess the impact of the task on the individual on 6 different fronts. NASA-Task Load Index (TLX) [9] is composed of 6 questions in which each one represents a dimension of the task requested: mental demand, physical demand, time pressure, perceived success with the task, overall effort level and frustration level.

For each of the questions the user must answer on a 21-point scale, from *Very Low* to *Very High* with exception for the Performance question whom answers vary from *Good* to *Poor* [14]. The purpose of this type of survey is to obtain the lowest possible values implying a low mental, physical and temporal effort as well as a reduced level of frustration

[8] https://www.linkedin.com/ Last accessed on: 27/11/2021

[9] https://humansystems.arc.nasa.gov/groups/TLX/index.php Last accessed on: 27/11/2021

Grade	SUS	Percentile range
A+	84.1 - 100	96 - 100
A	80.8 - 84.0	90 - 95
A-	78.9 - 80.7	85 - 89
B+	77.2 - 78.8	80 - 84
B	74.1 - 77.1	70 - 79
B-	72.6 - 74.0	65 - 69
C+	71.1 - 72.5	60 - 64
C	65.0 - 71.0	41 - 59
C-	62.7 - 64.9	35 - 40
D	51.7 - 62.6	15 - 34
F	0 - 51.6	0 - 14

Figure 5.30: SUS score grading table.

whilst carrying out the test scenarios presented below.

The results of both surveys are presented after the scenarios description and an example of both surveys can be found in subsection B.4.1. These results consist of the SUS scores average and the NASA-TLX response values average for each of the 6 questions from this survey.

5.7.1 First Test Scenario

John was given the task of buying a new power sander because the workers had warned him that it had stopped working. After deactivating this tool with the identifier OWDA294 2KDAW in the web application, he proceeded to purchase a new tool for the workshop.

Having the tool in his possession, John adds it to the system by entering the tool type, its model key, JAESD2332LADW, the Black & Decker brand and a brief description of the tool.

After adding the tool, John can now collect data from the tool on a vehicle under restoration, obtaining the frequency values of 95 Hz and 240 Hz for the minimum and maximum speeds. With these values, he can now save the digital signature on the tool previously added to the system.

5.7.2 Second Test Scenario

Throughout the workday, John is in charge of ensuring the system is working properly. This means that he has to perform a number of tasks, starting with checking the alarms on all the boxes in the system.

Then he must confirm that all the beacons are operational, immediately afterwards John checks the system alarms and sees which are the most recent ones. Finally, he was asked to download the log file of 30/08/2021 since a problem with the SensorBox_01 was detected on that day.

5.7.3 Third Test Scenario

After being told by a worker that he is going to commence work on another car, John needs to update the sensor box settings. The worker José Castro is using SensorBox_07 and was working on a Ford Mustang with license plate 56AA23, but is now starting to work on a Corvette C4 with license plate 9312AA.

John knows that José works at a very fast pace so he decides to reduce the time between measurements to 10 seconds.

5.7.4 Survey Results

In this section, we will present the results obtained in both surveys as histograms. As you will be able to see, all of them have a count of 15 since 5 participants were questioned and each one carried out the 3 test scenarios presented above.

The results for the NASA-TLX and SUS surveys are represented in Table 5.1 below. In this table we present the average (μ) and standard deviation (σ) for each of the metrics.

Table 5.1: SUS and NASA-TLX survey results average and standard deviation.

Survey	**Metric**	μ	σ
SUS	SUS Score	93.83	5.62
NASA-TLX	Mental Demand	2.47	1.82
	Physical Demand	1.07	0.25
	Temporal Demand	4.87	5.31
	Performance	19.8	1.72
	Effort	2.53	1.82
	Frustration	1.60	1.08

In what concerns the SUS scores, an average value of 93.83 is a positive result placing the usability and learnability of our web application in a comfortable position in the table presented in Figure 5.30. Through the histogram represented at Figure 5.31 it is possible to observe the relation between the SUS scores and the number of test scenario/participant pairs. As we can see the results between the 15 test scenario/participant pairs are relatively stable.

The survey question that obtained more consistent answers is related to inconsistencies when interacting with the web application. The participants were unanimous in answering the question “I thought there was too much inconsistency in this system“ with “Strongly Disagree“ for all scenarios, which reveals some stability in the web application.

As can be seen in the resulting histogram of the NASA-TLX survey, there was a discrepancy in the question measuring the participant’s performance during the execution

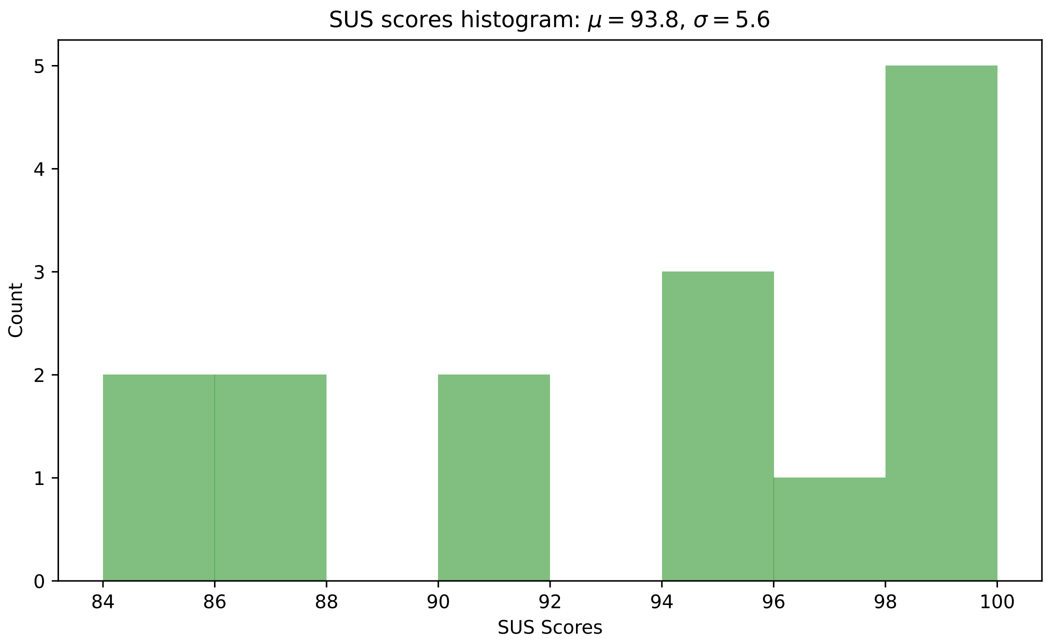


Figure 5.31: SUS scores histogram.

of the test scenarios. The error was to use inverse scales in the answer, that is, instead of using a scale where the value of 1 would mean "Good" and 21 "Poor" for this question, a scale was used where 1 means "Very Low" and 21 "Very High".

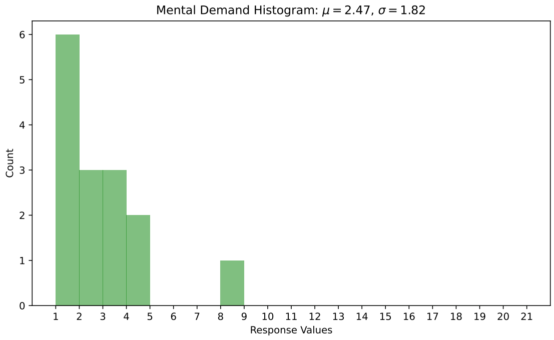
This eventually yielded reversed results for this particular question so for future reference we will assume high values in the performance histogram as positive values where participants think they have successfully achieved all scenario objectives.

As it is possible to observe from the Figure 5.32d histogram, abnormally high values are obtained, which actually mean that participants assumed that they were able to perform all the tasks requested in the different scenarios, results that are in line with the values obtained in the SUS survey.

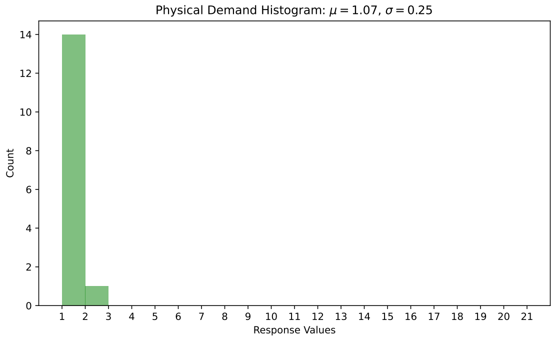
In addition to the performance values, in the Table 5.1 table it is possible to observe the average values of the responses from this survey, which implies an uncomplicated interaction with the application, moreover, the participants considered that they were able to execute the scenarios successfully and did not feel frustrated while doing so.

The most unanimous answer by the participants occurred in the physical demand question, a value that is usually low when web applications are evaluated. In the question related to temporal demand, which assesses how rushed was the pace of the task, we obtained abnormally high values three times which could suggest a poor understanding of the question by a participant when answering in all three surveys, resulting in the highest standard deviation of this survey.

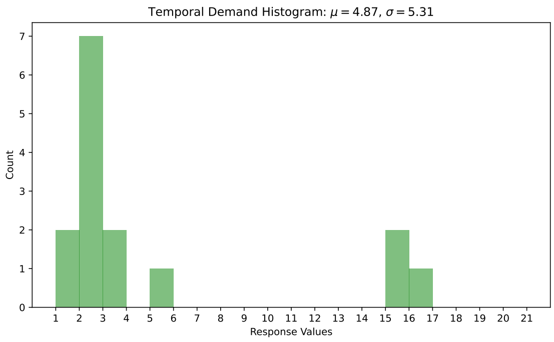
The results of both surveys are in agreement and even though the results are positive some suggestions were made by the participants since they have a background in Computer Science. The most pertinent suggestions are related to usability, such as the



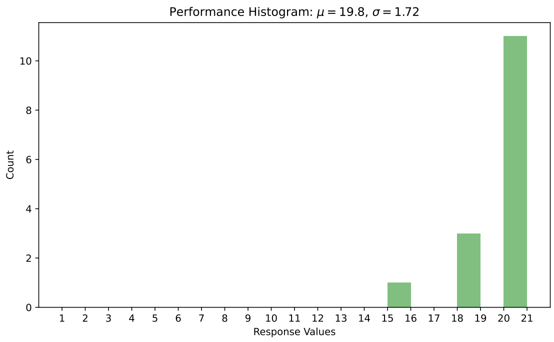
(a) Mental Demand Histogram.



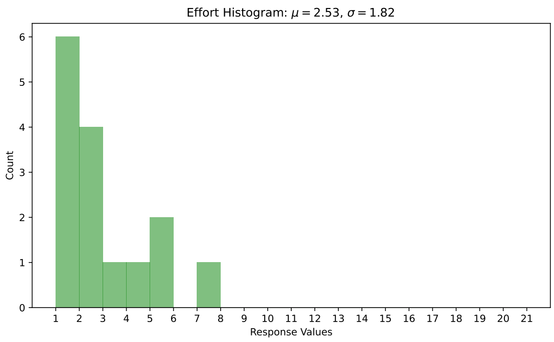
(b) Physical Demand Histogram.



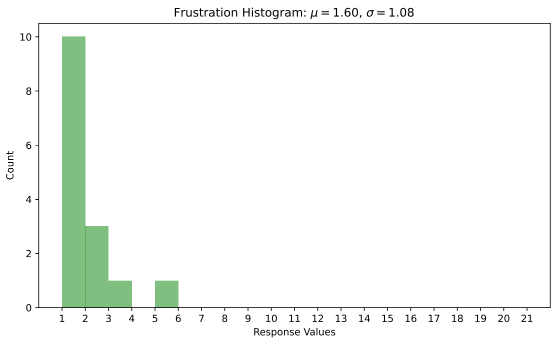
(c) Temporal Demand Histogram.



(d) Performance Histogram.



(e) Effort Histogram.



(f) Frustration Histogram.

Figure 5.32: NASA-TLX survey histograms.

possibility of changing the status of a beacon directly from its list and the use of input boxes instead of sliders in the sensor box configuration page.

During the second scenario, certain participants went first to the general system alarms instead of looking for the specific alarms for each box as requested, which points to the need for a better distinction between both.

5.8 Integration Tests

To test the connections between the two dissertations presented in section 4.8 it was necessary to create two integration tests. The focus of these tests is to show the capacity of the two subsystems to communicate with each other and pass the necessary information for both to function.

One of the most relevant aspects during the execution of this subsystem is the need to pass the processed information to the other subsystem developed in the scope of [22], to be mapped and confirmed by the plant shop manager.

For this reason, similar test files to the ones used to test the data processing function presented in section 5.6 were used as input to this particular test.

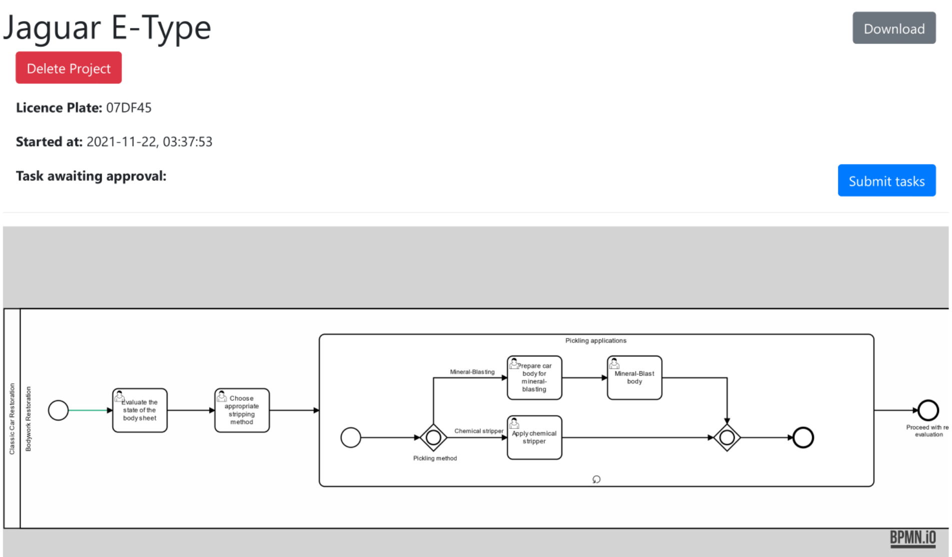


Figure 5.33: The colleague's Jaguar project details page before this subsystem correctly identifies a mineral blast event.

To spare the reader from repeated situations we will use as a demonstration example only the test file where the 'Mineral Blast' event is identified by the algorithm responsible for processing the data coming from the sensor boxes. As can be seen in Figure 5.33 the Jaguar in question has a mineral blast process that can be identified by the subsystem.

After running the data processing function presented in subsection 4.6.2 with an input file whose content is in Listing 8 it produces an output file with the following content (Listing 13), being able to identify the 'Mineral Blast' event. This output file is the input of the colleague's subsystem.

The proof that this file was correctly processed by the colleague's dissertation subsystem [22] is that in Figure 5.34 we can see that same activity in yellow, as it was described in section 4.8 where we presented the integration of the two subsystems.


```json
[
  {
    "DeviceId": "SensorBox_01",
    "Car": "07DF45",
    "Location": 3,
    "Event": ["Mineral Blast"],
    "StartTimestamp": 1636213943,
    "EndTimestamp": 1636213943
  }
]
```


Listing 13: Output file from the process identification algorithm.

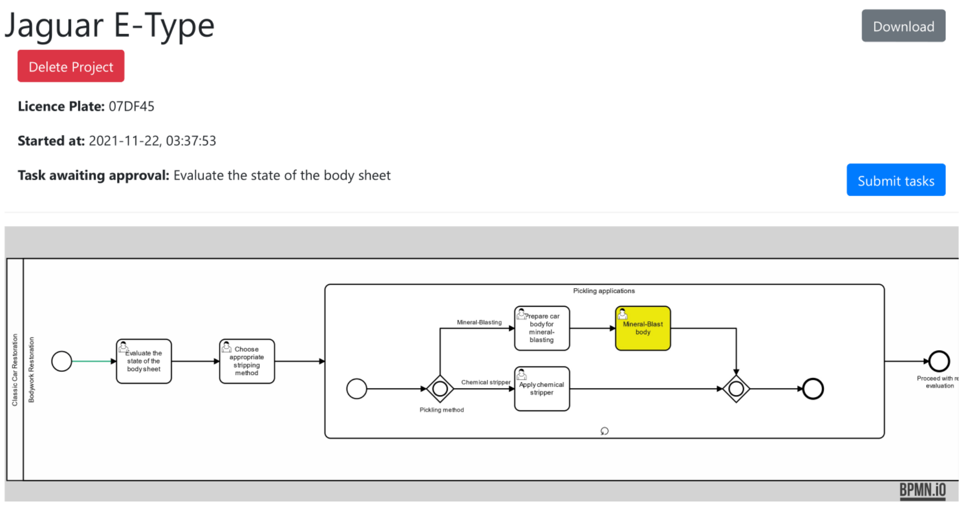


Figure 5.34: The colleague's Jaguar project details page after this dissertation system correctly identifies a mineral blast event.

This sequence confirms that if a restoration process event that is waiting to be detected is in fact detected by a sensor box, it is possible to pass that information to the subsystem developed in the scope of [22], to enrich the feedback provided to the car owner.

The second integration test aimed to validate the ability to obtain information of active restoration projects from the colleague's subsystem developed in the scope of [22] to give the workshop worker the ability to associate any sensor box to one of those projects.

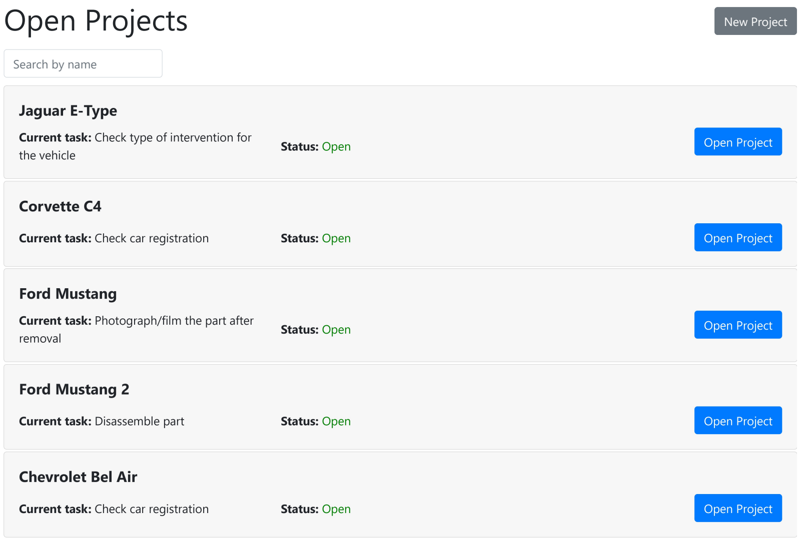


Figure 5.35: Colleague's Tasklist web application homepage responsible for creation and management of car restoration projects.

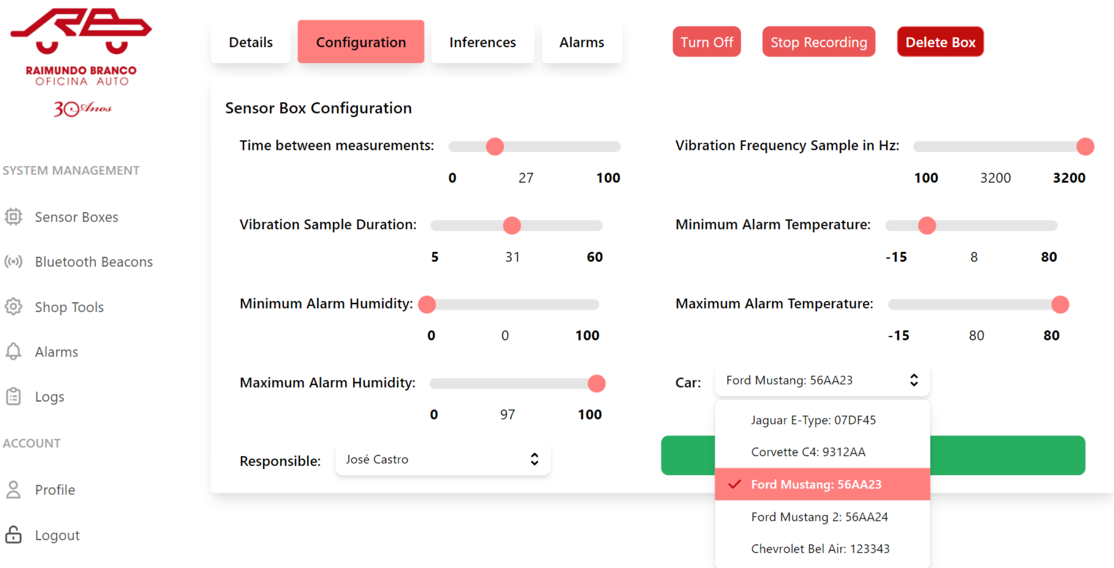


Figure 5.36: The monitoring and control web application page of this dissertation that shows the active restoration projects.

As it is possible to see by Figure 5.35, there are 5 active car restoration projects in the subsystem, originated from the colleague's dissertation [22]. In Figure 5.36 the list of these same projects can be seen in the web application in charge of monitoring and controlling the IoT subsystem. One detail relevant to point out is the incorporation of the car license plate next to the project name to avoid any confusion since these are unique for each vehicle.

Similarly to section 4.8 where we presented the systems integration, this section was also based on a co-authored software documentation article [26] developed in the scope of the entire project and, as such, is reproduced both here and in [22].

5.9 Validity Threats

During the development of this work, several sensor data files were collected in the workshop with the objective of later identifying the tools used by the workers. There are certain tasks in the workshop that are more recurrent than others, which meant more data files from tools such as electric and manual sanders and fewer files on tools such as polishers.

As mentioned during the validation of the workshop interior location system, it was not possible to perform this test in the building where the system is expected to be implemented, due to its reconstruction. Once the construction is finished, it will be necessary to implement this system in the building and repeat the test.

During the power bank capacity test, in the most intensive tests in which the sensor box was actively collecting and storing data, it remained stationary, meaning that there was no change from one beacon boundary zone to another. However, as can be expected in a normal operating situation, beacons were always active around the sensor box during the entire test. Due to the location where this test was carried out, the sensor box did not suffer interactions with any kind of tools, despite the continuous data collection, pre-processing and storage.

All these battery tests were developed using a Raspberry Pi with 2 Gigabyte (Gb) of RAM, the energy expenditure, as well as the ability to run the code on any other Raspberry Pi models, is unknown. Within the same model, Raspberry Pis with 4 Gb of RAM were also used, which did not affect the normal operation of the system running on these devices.

CHAPTER 6

Conclusions

This chapter ends this work with a summary of the solution developed and future directions planned to implement and further evaluate it.

6.1 Conclusions

The purpose of this work was to study the feasibility of creating a system capable of identifying different tasks and tools in the various stages of classic car restorations and to provide this information in a way that it could be catalogued and made available for documentation and certification purposes. In what concerns this question we believe it is possible to identify different restoration tasks and tools from a restoration process despite the need to improve the way the system detects them.

To achieve this we created an IoT application using various AWS services and devices that could identify such tasks and tools. We studied which sensors could be useful, after having identified the tools and processes we wanted to detect, and collected data over a few months. Despite positive results in identifying frequencies in some tools and materials by the sensor boxes, the same method proved ineffective for other tools. The limited knowledge in the field of temporal data processing and digital signal processing delayed the development of this component.

In the context of the workshop, we knew the relevance of indoor location to help identify the different tools, so it was also necessary to create a system to infer the location of the sensor boxes. It was necessary to instil some resilience to errors not only in the IoT devices but also in the indoor location system because each sensor box or beacon can fail and it should be possible to replace them if this occurs.

To be able to identify restoration tasks we also implemented an algorithm responsible for identifying tools and processes from raw data sent from the sensor boxes. As you might expect it was necessary to have a dataset of all the tools that are expected to be identified by the system, this way we could deal with the breakdown and purchase of tools. This possibility as well as the possible failure of beacons was the main motivation for creating the algorithm the way it was presented, this way we can always use the most recent information regarding the status of the system components to make the inferences.

Another key objective was the development of a web application capable of monitoring the whole system and controlling the sensor boxes. Through pilot tests, it was possible to obtain an initial evaluation of its performance, although it is necessary to test with a larger and more diverse population.

With the work developed in this dissertation, we believe to have managed to prove the viability of the idea through the implementation of several components properly tested. With this work, we hope to contribute to the development of new projects, related to the subject and continue to improve everything that was developed in this work to make the system more economical, viable and assertive in its inferences regarding the different tools.

6.2 Future Work

One of the most relevant aspects regarding the future work of this project is to try to create a more affordable sensor box, either through the use of microcontrollers or even other Raspberry Pi models, without forgetting the necessary requirements for the normal system execution, mainly the processing power required to execute the FFT algorithm.

Another aspect that will have to be implemented is the rotation of the sensor box certificates. Although the current ones will only reach the end of their useful life in 2049 and since our system is not used by unknown external entities, an automatic rotation of the certificates is always recommended.

Improving the restoration process identification algorithm and the algorithm responsible for locating the sensor boxes within the workshop requires a more thorough data gathering and testing process. The more data recorded the better the system manager's ability to correct some situations that may not have been caught during the testing done for this work. The continuous recording, processing and analysis of data will enable the fine-tuning of the algorithms, producing more reliable results.

In line with recording more data, better ways will have to be found to identify tools that have a lower digital signature that is ultimately dissipated in the noise. The use of smart power sockets positioned in strategic places coupled with information on the energy consumption of each tool can further help the algorithm identify the correct one.

The algorithm in charge of identifying restoration processes does not inform or try to re-execute if an error has occurred. Warning the system manager whenever any trouble processing data is detected is one of the most important features that should be implemented. Moreover, having a monitoring screen on successful and failed executions with the ability to order the re-execution of failed ones would give more control over an important component of this system.

The use of RSSI to identify the location of the sensor boxes inside the workshop may have errors and some events may not be detected. To avoid this kind of situation it could be interesting the use of tags in certain workshop areas and NFC readers in the sensor boxes to identify their location without any error.

The disadvantage of this take is the need for the workshop worker to not forget to pass the sensor box by the tag zone when he changes zones, otherwise, if he forgets to do so, everything that has been done by him may not be identified by the system, losing part or all the vehicle restoration process.

As for the web application, and since the use of tablets is common in Raimundo Branco's workshop, a pertinent improvement is for it to be able to resize depending on the size of the screen, as well as to change the language depending on the device's language. The creation of different types of users will allow us to secure the most confidential data, strategically assigning users limiting roles in the web application.

Bibliography

[1] N. Antonopoulos and L. Gillam. *Cloud Computing: Principles, Systems and Applications*. Accessed: 2021/8/20. Jan. 2010. ISBN: 978-1-84996-240-7. DOI: 10.1007/978-1-84996-241-4 (cit. on p. 11).

[2] Autocar. *The 13 most expensive cars ever sold*. Accessed: 2020/11/12. 2018. URL: https://www.autocar.co.uk/car-news/industry/12-most-expensive-cars-sold-auction-#:~:text=1.,car%5C%20parts%5C%20mogul%5C%20David%5C%20MacNeil (cit. on p. 1).

[3] R. Bala et al. "Magic Quadrant for Cloud Infrastructure and Platform Services". In: *Gartner* (2020). Accessed: 2021/10/24. URL: https://www.gartner.com/doc/reprints?id=1-271OE4VR&ct=210802&st=sb (cit. on p. 21).

[4] J. Bosch. *Software Architecture: The Next Step*. Ed. by F. Oquendo, B. C. Warboys, and R. Morrison. Accessed: 2021/10/24. Springer Berlin Heidelberg, 2004, pp. 194–199. ISBN: 978-3-540-24769-2 (cit. on p. 25).

[5] M. Brambilla and P. Fraternali. *Interaction flow modeling language: Model-driven UI engineering of web and mobile apps with IFML*. Morgan Kaufmann, 2014 (cit. on p. 48).

[6] E. Burmeister and L. M. Aitken. "Sample size: How many is enough?" In: *Australian Critical Care* 25.4 (2012), pp. 271–274. ISSN: 1036-7314. DOI: 10.1016/j.aucc.2012.07.002. URL: https://www.sciencedirect.com/science/article/pii/S1036731412000847 (cit. on p. 92).

[7] C/S2ESC - Software & Systems Engineering Standards Committee. *Recommended Practice for Architectural Description for Software-Intensive Systems*. en. Standard IEEE 1471-2000. Institute of Electric and Electronics Engineers, 2000. URL: https://standards.ieee.org/standard/1471-2000.html (cit. on p. 24).

[8] Cisco. *Fog Computing and the Internet of Things: Extend the Cloud to Where the Things Are*. Accessed: 2021/8/18. 2015. URL: https://www.cisco.com/c/dam/en_us/solutions/trends/iot/docs/computing-overview.pdf (cit. on p. 12).

[9] D. De et al. "Chapter 1 - Advances in Real-World Sensor Network System". In: *Connected Computing Environment*. Ed. by A. Hurson. Vol. 90. Advances in Computers. Accessed: 2020/11/12. Elsevier, 2013, pp. 1–3. DOI: 10.1016/B978-0-12-408091-1.00001-4 (cit. on p. 3).

[10] Q. Dong and W. Dargie. "Evaluation of the reliability of RSSI for indoor localization". In: *2012 International Conference on Wireless Communications in Underground and Confined Areas*. Accessed: 2021/6/12. 2012, pp. 1–6. DOI: 10.1109/ICWCUCA.2012.6402492 (cit. on p. 13).

[11] N.-C. Gaitan, V. G. Gaitan, and I. Ungurean. "An IoT Middleware Framework for Industrial Applications". In: *International Journal of Advanced Computer Science and Applications* 7 (2016). Accessed: 2021/1/12, pp. 31–41 (cit. on p. 9).

[12] D. Galar and U. Kumar. "Chapter 4 - Data and Information Fusion From Disparate Asset Management Sources". In: *eMaintenance*. Ed. by D. Galar and U. Kumar. Accessed: 2021/2/6. Academic Press, 2017, pp. 179–234. ISBN: 978-0-12-811153-6. DOI: 10.1016/B978-0-12-811153-6.00004-X (cit. on p. 4).

[13] I. Gryech et al. "MoreAir: A Low-Cost Urban Air Pollution Monitoring System". In: *Sensors* 20.4 (2020). Accessed: 2021/10/21. ISSN: 1424-8220. DOI: 10.3390/s20040998 (cit. on p. 19).

[14] S. G. Hart. "Nasa-Task Load Index (NASA-TLX); 20 Years Later". In: *Proceedings of the Human Factors and Ergonomics Social Annual Meeting* 50 (Oct. 2006), pp. 904–908. DOI: 10.1177/154193120605000909 (cit. on p. 93).

[15] IEEE. *Signal Processing 101*. Accessed: 2021/8/20. 2021. URL: https://signalprocessingsociety.org/our-story/signal-processing-101 (cit. on p. 14).

[16] D. Kiel et al. "The impact of the industrial internet of things on established business models". In: *Proceedings of the 25th international association for management of technology (IAMOT) conference*. Accessed: 2021/1/12. 2016, pp. 673–695 (cit. on p. 9).

[17] H. Lasi, F. Fettke Kemper, and Hoffmann. "Industry 4.0". In: *Business & Information Systems Engineering* 6 (2014). Accessed: 2021/4/21. ISSN: 1867-0202. DOI: 10.1007/s12599-014-0334-4 (cit. on p. 19).

[18] J. W. Leis. "Sampled Signals and Digital Processing". In: *Digital Signal Processing Using MATLAB for Students and Researchers*. John Wiley & Sons, Ltd, 2011. Chap. 3, pp. 45–101. ISBN: 9781118033623. DOI: 10.1002/9781118033623.ch3 (cit. on p. 35).

[19] J. Lewis and J. Sauro. "Item Benchmarks for the System Usability Scale". In: 13 (May 2018), pp. 158–167 (cit. on p. 93).

[20] J. R. Lewis and J. Sauro. “The Factor Structure of the System Usability Scale”. In: *Proceedings of the 1st International Conference on Human Centered Design: Held as Part of HCI International 2009*. HCD 09. San Diego, CA, 2009, pp. 94–103. ISBN: 9783642028052. DOI: 10.1007/978-3-642-02806-9_12 (cit. on p. 93).

[21] M. Li, A. Yadollahi, and B. Taati. “Non-Contact Vision-Based Cardiopulmonary Monitoring in Different Sleeping Positions”. In: *IEEE Journal of Biomedical and Health Informatics* 21 (May 2016), pp. 1–1. DOI: 10.1109/JBHI.2016.2567298 (cit. on p. 36).

[22] D. Lívio. “Process-based Monitoring in Industrial Context - The Case of Classic Cars Restoration”. In: () (cit. on pp. 24, 26, 50, 53, 98–101).

[23] G. Manes et al. “Realtime Gas Emission Monitoring at Hazardous Sites Using a Distributed Point-Source Sensing Infrastructure”. In: *Sensors* 16.1 (2016). Accessed: 2021/10/21. ISSN: 1424-8220. DOI: 10.3390/s16010121 (cit. on p. 20).

[24] J. Nielsen. “Chapter 2 - What Is Usability?” In: *Usability Engineering*. Ed. by J. Nielsen. San Diego: Morgan Kaufmann, 1993, pp. 23–48. ISBN: 978-0-12-518406-9. DOI: 10.1016/B978-0-08-052029-2.50005-X (cit. on p. 48).

[25] K. Pellengahr and D. A. Schulte. *Einkauf 4.0*. Accessed: 2021/10/22. 2016. URL: https://www.bme.de/fileadmin/_horusdam/4190-Vorstudie_Einkauf_40.pdf (cit. on p. 19).

[26] D. Pereira and D. Lívio. *Integration Testing of an IoT Process-Based Monitoring System for Classic Car Restoration Shops*. Nov. 2021. DOI: 10.5281/zenodo.5731608 (cit. on p. 101).

[27] D. Pereira et al. *Software Architecture of an IoT Process-Based Monitoring System for Classic Car Restoration Shops*. Nov. 2021. DOI: 10.5281/zenodo.5731602 (cit. on p. 53).

[28] L. Rabiner and B. Gold. *Theory and Application of Digital Signal Processing*. Prentice-Hall signal processing series. Prentice-Hall, 1975. ISBN: 9780139141010. URL: https://books.google.pt/books?id=iAxTAAAAMAAJ (cit. on p. 15).

[29] S. K. V. Ragavan and M. Shanmugavel. “Engineering cyber-physical systems — Mechatronics wine in new bottles?” In: *2016 IEEE International Conference on Computational Intelligence and Computing Research (ICCIC)*. Accessed: 2020/11/13. 2016, pp. 1–5. DOI: 10.1109/ICCIC.2016.7919516 (cit. on p. 4).

[30] A. Rayes and S. Salam. “Internet of Things (IoT) Overview”. In: *Internet of Things From Hype to Reality: The Road to Digitization*. Accessed: 2020/12/2. Springer International Publishing, 2019, pp. 1–35. ISBN: 978-3-319-99516-8. DOI: 10.1007/978-3-319-99516-8_1 (cit. on p. 9).

[31] K. Shin and J. Hammond. *Fundamentals of signal processing for sound and vibration engineers*. Accessed: 2021/8/20. Wiley, Feb. 2008. ISBN: 978-0-470-51188-6 (cit. on p. 14).

[32] Statista. *Internet of Things - number of connected devices worldwide 2015-2025*. Accessed: 2021/1/13. 2016. URL: https://www.statista.com/statistics/471264/iot-number-of-connected-devices-worldwide/ (cit. on p. 12).

[33] J. Y. Stein. "Digital Signal Processors". In: *Digital Signal Processing: A Computer Science Perspective*. Accessed: 2020/11/16. John Wiley & Sons, Ltd, 2000. Chap. 17, pp. 619–644. ISBN: 9780471200598. DOI: 10.1002/047120059X.ch17 (cit. on p. 4).

[34] M. Syafrudin et al. "Performance Analysis of IoT-Based Sensor, Big Data Processing, and Machine Learning Model for Real-Time Monitoring System in Automotive Manufacturing". In: *Sensors* 18.9 (2018). Accessed: 2021/2/18. ISSN: 1424-8220. DOI: 10.3390/s18092946 (cit. on p. 17).

[35] M. Ullrich et al. "Detection of Gait From Continuous Inertial Sensor Data Using Harmonic Frequencies". In: *IEEE Journal of Biomedical and Health Informatics* 24.7 (2020), pp. 1869–1878. DOI: 10.1109/JBHI.2020.2975361 (cit. on p. 36).

[36] P. University. *Spectrum Analysis*. Accessed: 2021/8/22. 2011. URL: https://engineering.purdue.edu/ME365/Textbook/chapter8.pdf (cit. on p. 15).

[37] V. K. Vaishnavi and W. L. Kuechler. "Design Science Research in Information Systems". In: *Ais* (2004). Accessed: 2021/8/15, pp. 1–45. DOI: 10.1007/978-1-4419-5653-8 (cit. on p. 5).

[38] W. Voorsluys, J. Broberg, and R. Buyya. "Introduction to Cloud Computing". In: *Cloud Computing*. John Wiley & Sons, Ltd, 2011. Chap. 1, pp. 1–41. ISBN: 9780470940105. DOI: 10.1002/9780470940105.ch1 (cit. on p. 11).

[39] P. Wlodarczak, M. Ally, and J. Soar. "Data Mining in IoT: Data Analysis for a New Paradigm on the Internet". In: *Proceedings of the International Conference on Web Intelligence*. WI '17. Accessed: 2021/2/16. Association for Computing Machinery, 2017, pp. 1100–1103. ISBN: 9781450349512. DOI: 10.1145/3106426.3115866 (cit. on p. 10).

[40] P. Zhang, Y. Wu, and H. Zhu. "Open ecosystem for future industrial Internet of things (IIoT): Architecture and application". In: *CSEE Journal of Power and Energy Systems* 6.1 (2020). Accessed: 2021/2/1, pp. 1–11. DOI: 10.17775/CSEEJPES.2019.01810 (cit. on pp. 10, 20).

APPENDIX A

Processing Identification Algorithm

A.1 Processing function flow diagram

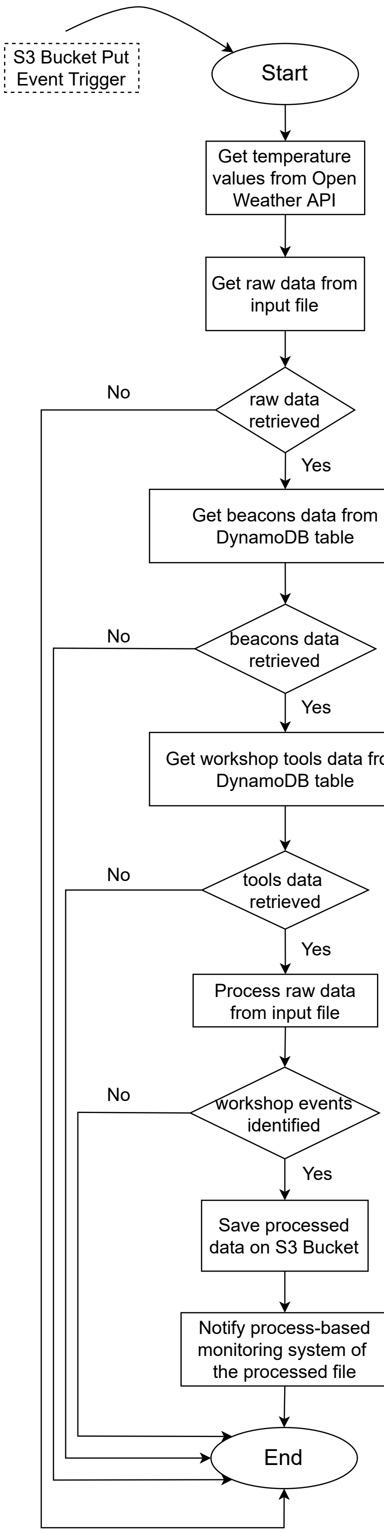


Figure A.1: Process Identification algorithm flowchart.

As can be seen in this high-level representation of the algorithm, it starts by using the

Open Weather API [1] to get the temperature data at the workshop location. Next, it fetches the contents of the raw data file sent by the box, the most recent data from the beacons and the active tools in the workshop. With all this data, it can start processing the raw data and identify the processes based on that data. Finally, if restoration processes have been identified, the algorithm saves these processes to a file in the cloud and notifies another system that a new raw data file has been processed.

[1] https://openweathermap.org/api

A.2 Processing function input files

Input file for testing the process identification algorithm with respect to event grouping:

```
[
  1636727056,
  {
    "DeviceId": "SensorBox_01",
    "Car": "9312AA",
    "Timestamp": 1636213943,
    "Temperature": 26.4375,
    "Humidity": 35.4,
    "VibrationPeaks": "[213]",
    "Beacons": {
      "0d5c5b6ed855dbbc": {
        "timestamp": 1636213941,
        "battery": 59,
        "rssi": -73,
        "temperature": 24.75
      },
      "fc337481622c4dc8": {
        "timestamp": 1636213930,
        "battery": 99,
        "rssi": -76,
        "temperature": 24.875
      },
      "4b528a07c6810670": {
        "timestamp": 1636213933,
        "battery": 100,
        "rssi": -78,
        "temperature": 25.9375
      }
    }
  },
  {
    "DeviceId": "SensorBox_01",
    "Car": "9312AA",
    "Timestamp": 1636214891,
    "Temperature": 26.4375,
    "Humidity": 35.4,
    "VibrationPeaks": "[213]",
```

```
    "Beacons": {
      "0d5c5b6ed855dbbc": {
        "timestamp": 1636213941,
        "battery": 59,
        "rssi": -73,
        "temperature": 24.75
      },
      "fc337481622c4dc8": {
        "timestamp": 1636213930,
        "battery": 99,
        "rssi": -76,
        "temperature": 24.875
      },
      "4b528a07c6810670": {
        "timestamp": 1636213933,
        "battery": 100,
        "rssi": -78,
        "temperature": 25.9375
      }
    }
  },
  {
    "DeviceId": "SensorBox_01",
    "Car": "9312AA",
    "Timestamp": 1636215174,
    "Temperature": 26.4375,
    "Humidity": 35.4,
    "VibrationPeaks": "[13]",
    "Beacons": {
      "0d5c5b6ed855dbbc": {
        "timestamp": 1636213941,
        "battery": 59,
        "rssi": -73,
        "temperature": 24.75
      },
      "fc337481622c4dc8": {
        "timestamp": 1636213930,
        "battery": 99,
        "rssi": -76,
        "temperature": 24.875
      },
```

```
      "4b528a07c6810670": {
        "timestamp": 1636213933,
        "battery": 100,
        "rssi": -78,
        "temperature": 25.9375
      }
    }
  },
  {
    "DeviceId": "SensorBox_01",
    "Car": "9312AA",
    "Timestamp": 1636215214,
    "Temperature": 26.4375,
    "Humidity": 35.4,
    "VibrationPeaks": "[13]",
    "Beacons": {
      "0d5c5b6ed855dbbc": {
        "timestamp": 1636213941,
        "battery": 59,
        "rssi": -73,
        "temperature": 24.75
      },
      "fc337481622c4dc8": {
        "timestamp": 1636213930,
        "battery": 99,
        "rssi": -76,
        "temperature": 24.875
      },
      "4b528a07c6810670": {
        "timestamp": 1636213933,
        "battery": 100,
        "rssi": -78,
        "temperature": 25.9375
      }
    }
  }
]
```


Listing 14: Input file for testing the process identification algorithm with respect to event grouping.

Input file for testing the process identification algorithm with respect to intermittent tool usage:


```
[
  1636727056,
  {
    "DeviceId": "SensorBox_01",
    "Car": "9312AA",
    "Timestamp": 1636213943,
    "Temperature": 26.4375,
    "Humidity": 35.4,
    "VibrationPeaks": "[213]",
    "Beacons": {
      "0d5c5b6ed855dbbc": {
        "timestamp": 1636213941,
        "battery": 59,
        "rssi": -73,
        "temperature": 24.75
      },
      "fc337481622c4dc8": {
        "timestamp": 1636213930,
        "battery": 99,
        "rssi": -76,
        "temperature": 24.875
      },
      "4b528a07c6810670": {
        "timestamp": 1636213933,
        "battery": 100,
        "rssi": -78,
        "temperature": 25.9375
      }
    }
  },
  {
    "DeviceId": "SensorBox_01",
    "Car": "9312AA",
    "Timestamp": 1636214213,
    "Temperature": 26.4375,
    "Humidity": 35.4,
    "VibrationPeaks": "[13]",
```

```
    "Beacons": {
      "0d5c5b6ed855dbbc": {
        "timestamp": 1636213941,
        "battery": 59,
        "rssi": -73,
        "temperature": 24.75
      },
      "fc337481622c4dc8": {
        "timestamp": 1636213930,
        "battery": 99,
        "rssi": -76,
        "temperature": 24.875
      },
      "4b528a07c6810670": {
        "timestamp": 1636213933,
        "battery": 100,
        "rssi": -78,
        "temperature": 25.9375
      }
    }
  },
  {
    "DeviceId": "SensorBox_01",
    "Car": "9312AA",
    "Timestamp": 1636214462,
    "Temperature": 26.4375,
    "Humidity": 35.4,
    "VibrationPeaks": "[213]",
    "Beacons": {
      "0d5c5b6ed855dbbc": {
        "timestamp": 1636213941,
        "battery": 59,
        "rssi": -73,
        "temperature": 24.75
      },
      "fc337481622c4dc8": {
        "timestamp": 1636213930,
        "battery": 99,
        "rssi": -76,
        "temperature": 24.875
      },
```

```
    "4b528a07c6810670": {
      "timestamp": 1636213933,
      "battery": 100,
      "rssi": -78,
      "temperature": 25.9375
    }
  }
},
{
  "DeviceId": "SensorBox_01",
  "Car": "9312AA",
  "Timestamp": 1636214583,
  "Temperature": 24.375,
  "Humidity": 46.1,
  "VibrationPeaks": "[206]",
  "Beacons": {
    "b4c8c0b52049e4de": {
      "timestamp": 1636727052,
      "battery": 36,
      "rssi": -77,
      "temperature": 19
    },
    "9fcccaab3f956f66": {
      "timestamp": 1636727053,
      "battery": 48,
      "rssi": -64,
      "temperature": 18.875
    },
    "a58db253434c6759": {
      "timestamp": 1636727055,
      "battery": 40,
      "rssi": -75,
      "temperature": 19.1875
    }
  }
},
{
  "DeviceId": "SensorBox_01",
  "Car": "9312AA",
  "Timestamp": 1636214961,
  "Temperature": 55.6875,
```

```
    "Humidity": 36.1,
    "VibrationPeaks": "[206]",
    "Beacons": {
      "0d5c5b6ed855dbbc": {
        "timestamp": 1636214521,
        "battery": 59,
        "rssi": -73,
        "temperature": 24.5625
      },
      "fc337481622c4dc8": {
        "timestamp": 1636214531,
        "battery": 99,
        "rssi": -69,
        "temperature": 24.75
      },
      "4b528a07c6810670": {
        "timestamp": 1636214533,
        "battery": 100,
        "rssi": -66,
        "temperature": 25.3125
      },
      "d5c87a75d12c7cbd": {
        "timestamp": 1636214528,
        "battery": 99,
        "rssi": -69,
        "temperature": 25.25
      }
    }
  }
]
```


Listing 15: Input file for testing the process identification algorithm with respect to intermittent tool usage.

Input file for testing the process identification algorithm with respect to the lack of frequencies detected in the paint zone.


```
[
  1636202314,
  {
    "DeviceId": "SensorBox_01",
    "Car": "9312AA",
    "Timestamp": 1636214538,
    "Temperature": 55.6875,
    "Humidity": 36.1,
    "VibrationPeaks": "[]",
    "Beacons": {
      "0d5c5b6ed855dbbc": {
        "timestamp": 1636214521,
        "battery": 59,
        "rssi": -73,
        "temperature": 24.5625
      },
      "fc337481622c4dc8": {
        "timestamp": 1636214531,
        "battery": 99,
        "rssi": -69,
        "temperature": 24.75
      },
      "4b528a07c6810670": {
        "timestamp": 1636214533,
        "battery": 100,
        "rssi": -66,
        "temperature": 25.3125
      },
      "d5c87a75d12c7cbd": {
        "timestamp": 1636214528,
        "battery": 99,
        "rssi": -69,
        "temperature": 25.25
      }
    }
  },
```

```
  {
    "DeviceId": "SensorBox_01",
    "Car": "9312AA",
    "Timestamp": 1636214623,
    "Temperature": 25.6875,
    "Humidity": 46.1,
    "VibrationPeaks": "[]",
    "Beacons": {
      "0d5c5b6ed855dbbc": {
        "timestamp": 1636214521,
        "battery": 59,
        "rssi": -73,
        "temperature": 24.5625
      },
      "fc337481622c4dc8": {
        "timestamp": 1636214531,
        "battery": 99,
        "rssi": -69,
        "temperature": 24.75
      },
      "4b528a07c6810670": {
        "timestamp": 1636214533,
        "battery": 100,
        "rssi": -66,
        "temperature": 25.3125
      },
      "d5c87a75d12c7cbd": {
        "timestamp": 1636214528,
        "battery": 99,
        "rssi": -69,
        "temperature": 25.25
      }
    }
  }
]
```


Listing 16: Input file for testing the process identification algorithm with respect to the lack of frequencies detected in the paint zone.

APPENDIX B

Web Application Files

B.1 User Stories

These are the web application user stories.

- As a shop floor manager, I want to remove a sensor box from the IoT system.
- As a shop floor manager, I want to see all the operational and non-operational sensor boxes.
- As a shop floor manager, I want to add a new tool to the system.
- As a shop floor manager, I want to update the status of a system tool.
- As a shop floor manager, I want to delete a tool from the system.
- As a shop floor manager, I want to update the digital signature of a tool.
- As a shop floor manager, I want to update the details of a tool.
- As a shop floor manager, I want to see all the Estimote beacons battery levels and information.
- As a shop floor manager, I want to change an Estimote beacon activity status.
- As a shop floor manager, I want to be able to set the minimum battery level.
- As a shop floor manager, I want to be notified whenever an Estimote beacon battery level goes below the minimum battery level.
- As a shop floor manager, I want to be able to replace a defective beacon.
- As a shop floor manager, I want to see the logs of the system.

- As a shop floor manager, I want to see the sensor box shadow (desired and reported state).
- As a shop floor manager, I want to update the sensor box shadow:
  - Turn off a sensor box.
  - Make the sensor box start gathering information.
  - Make the sensor box stop gathering information.
  - Associate a sensor box to a new car already in the system.
  - Disassociate a sensor box from a car.
  - Add / Edit / Remove the sensor box coordinator - worker responsible for it.
  - Edit the time between sensor box measurements.
  - Edit the temperature and humidity thresholds (minimum and maximum).
  - Edit the frequency used by the accelerometer to record data.
  - Edit the vibration sensing duration for the accelerometer sensor.
- As a shop floor manager, I want to see the alarms of a specific sensor box.
- As a shop floor manager, I want to be notified when an alarm is triggered by any sensor box.
- As a shop floor manager, I want to be notified when an alarm is triggered by any sensor box.
- As a shop floor manager, I want to see the latest decisions inferred from each sensor box data.
- As a shop floor manager, I want to be able to see the latest alarms triggered by all sensor boxes.
- As a shop floor manager, I want to be able to edit my password.

B.2 IFML Diagram

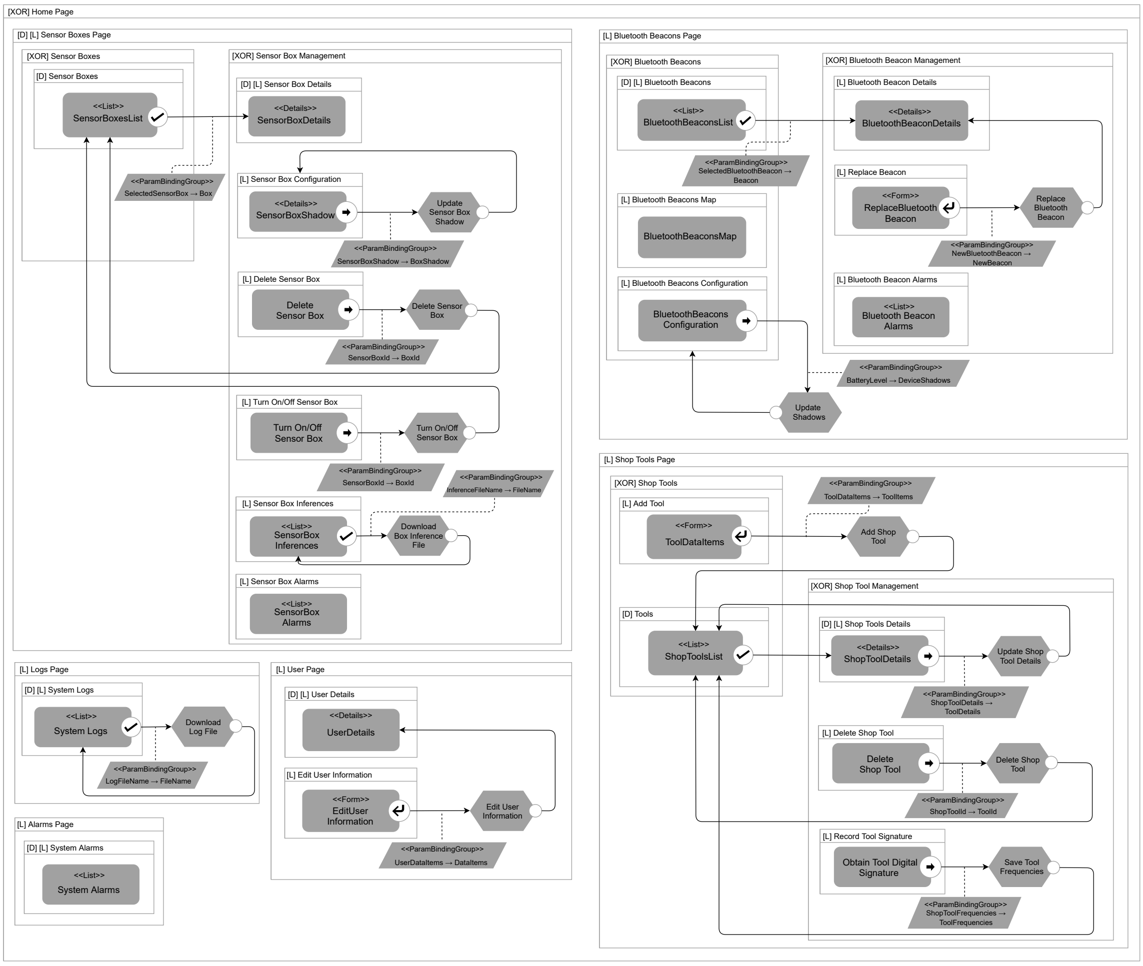


Figure B.1: IFML diagram of the web application.

B.3 Web application mock-ups

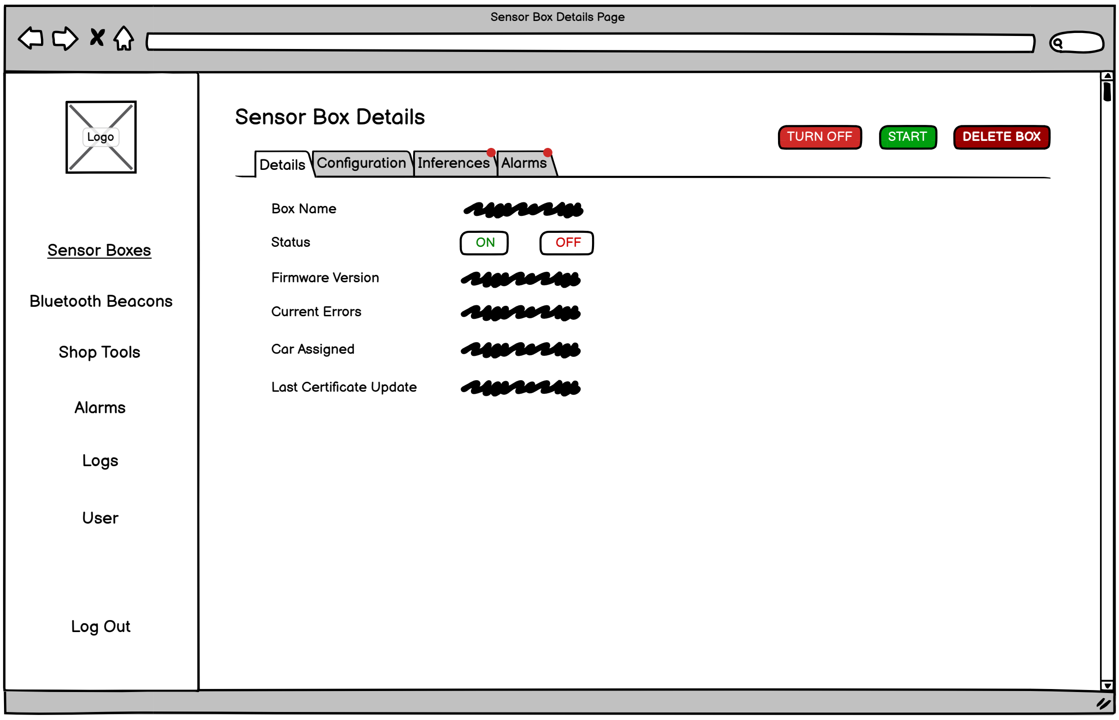


Figure B.2: Sensor box details page.

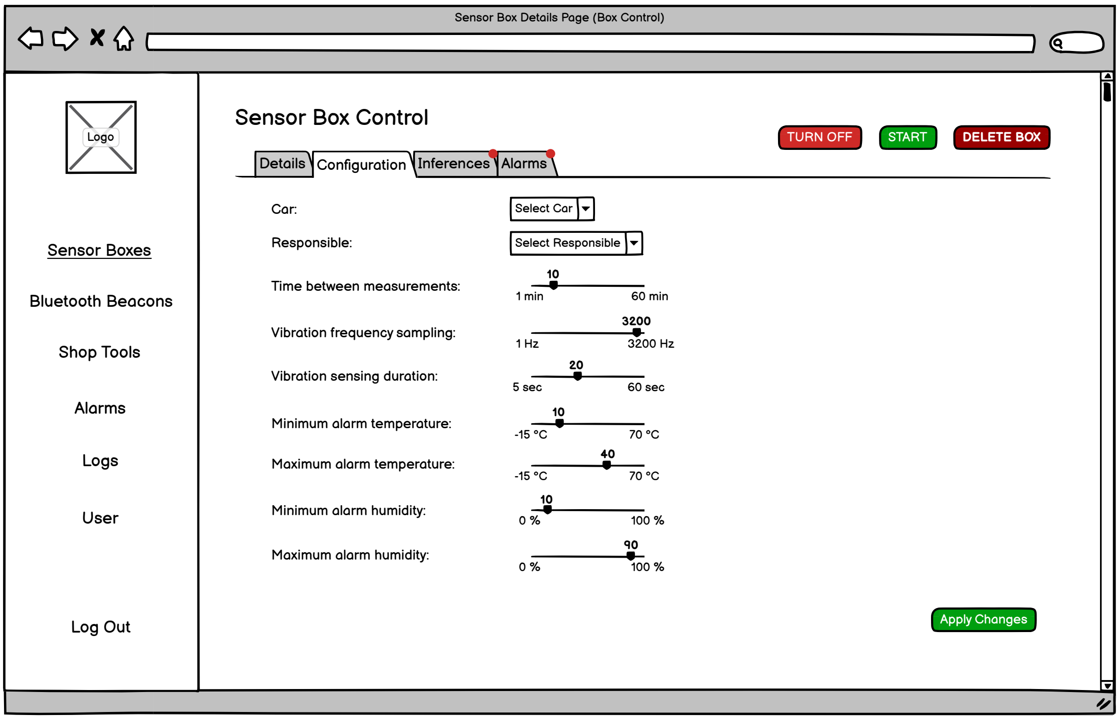


Figure B.3: Sensor box control page.

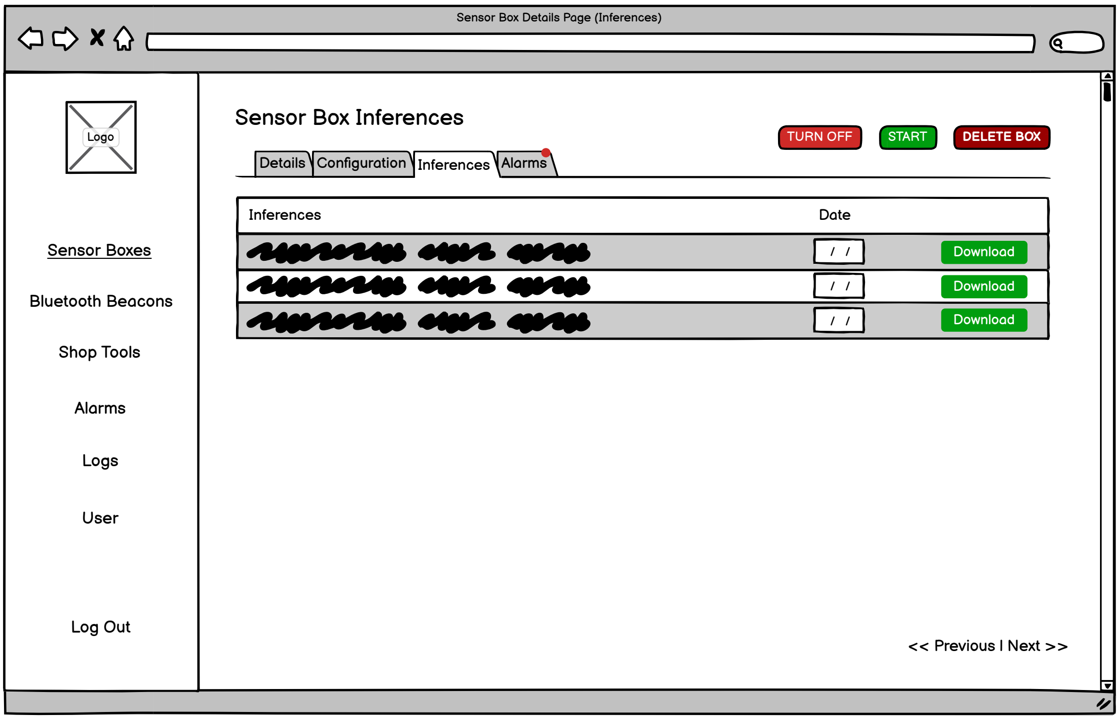


Figure B.4: Sensor box inference files page.

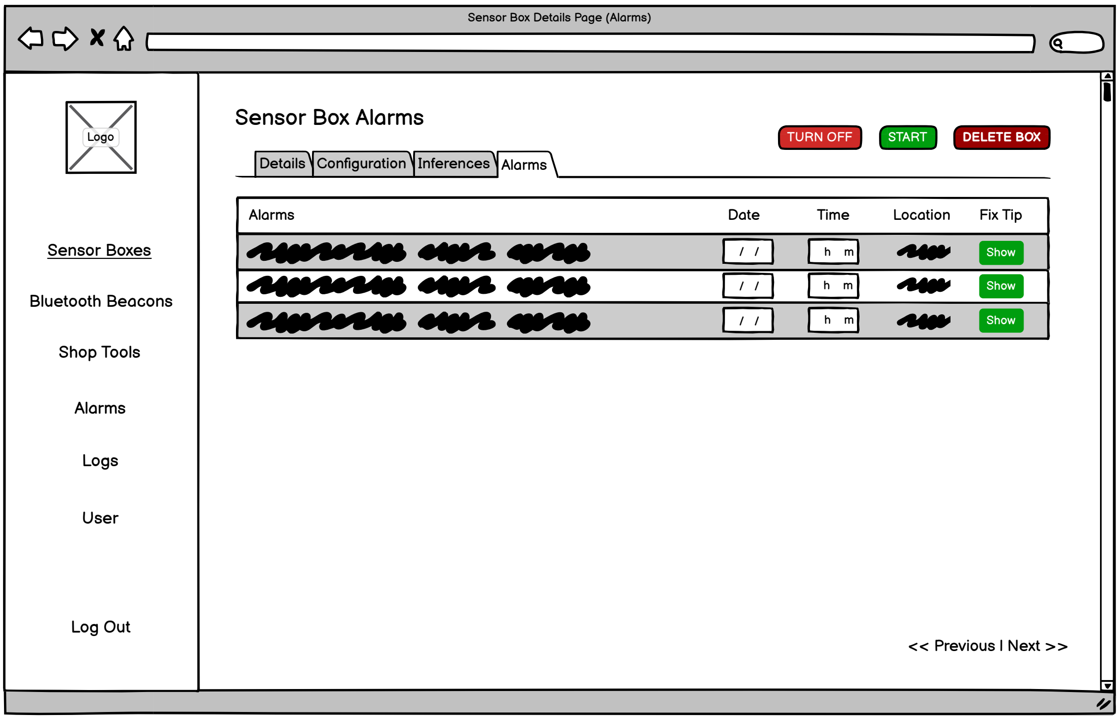


Figure B.5: Sensor box alarms page.

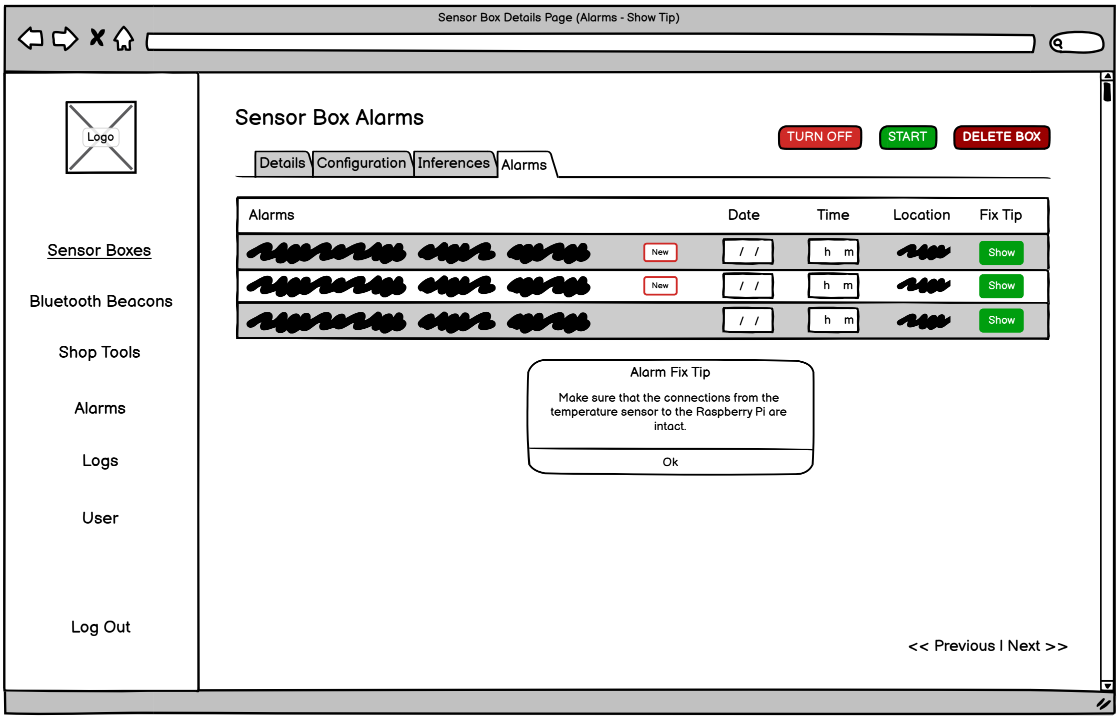


Figure B.6: Fix tip from a sensor box alarm.

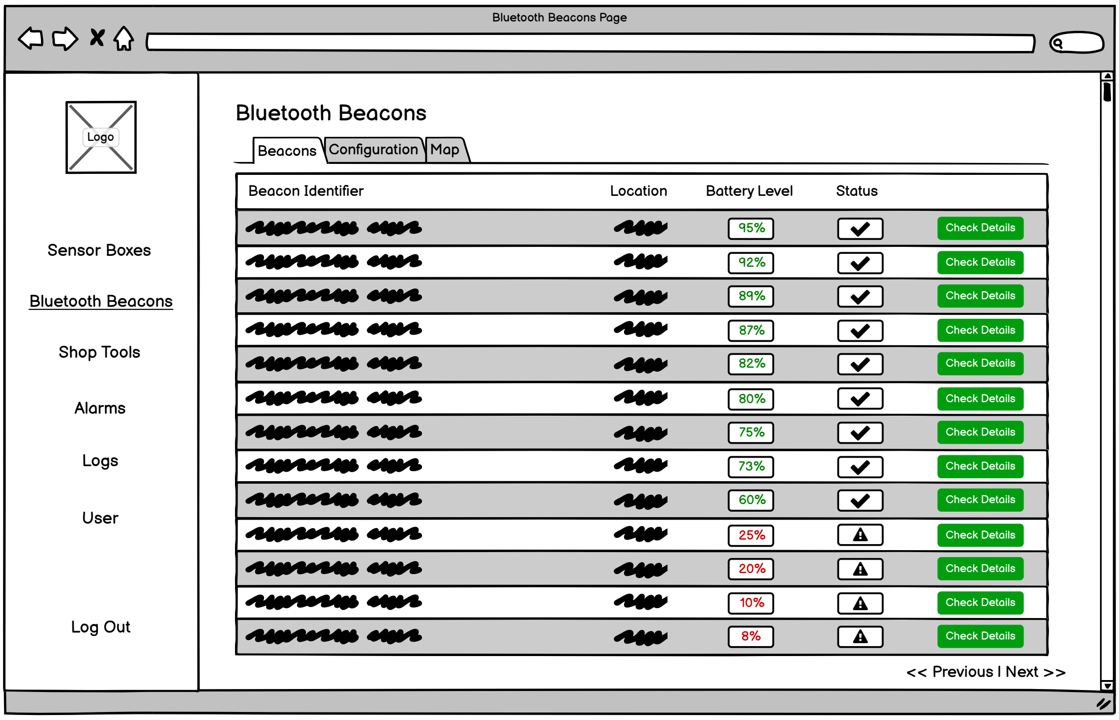


Figure B.7: BLE beacons main page.

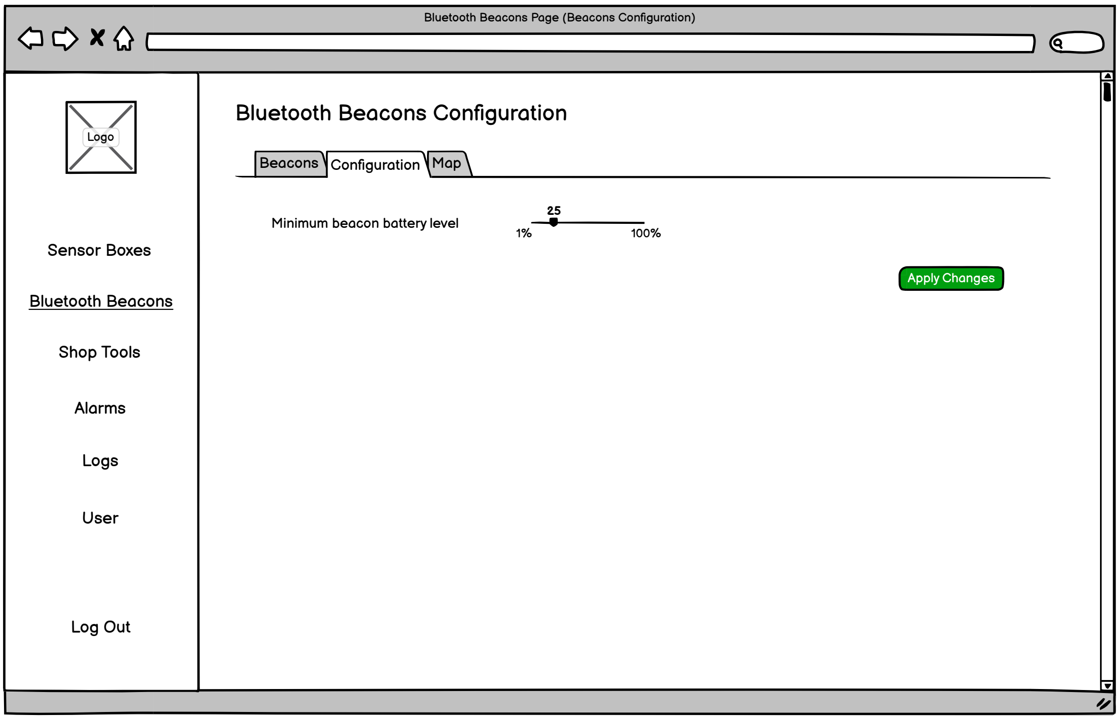


Figure B.8: BLE beacons minimum battery level configuration page.

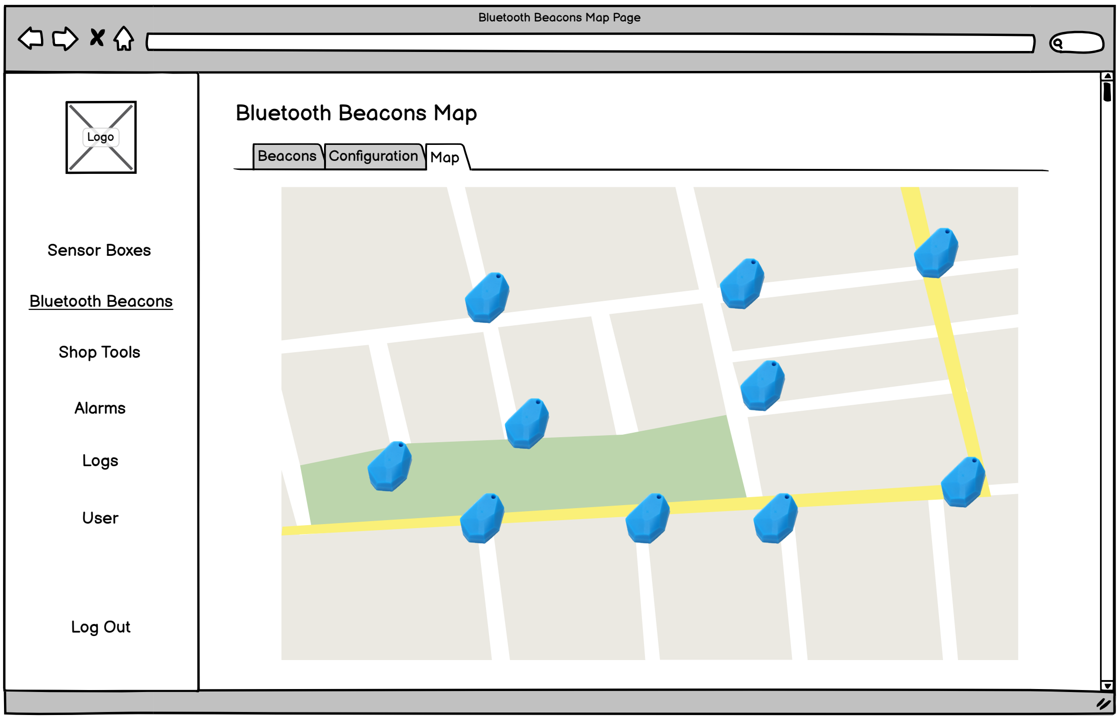


Figure B.9: BLE beacons distribution across the workshop.

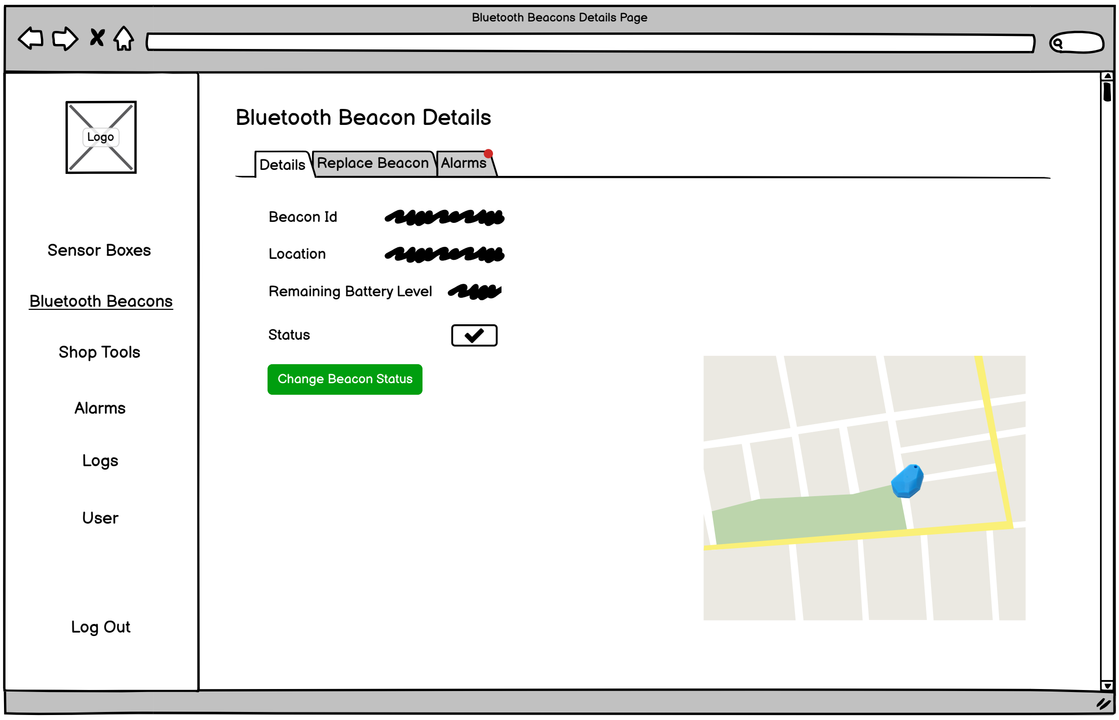


Figure B.10: BLE beacon details page.

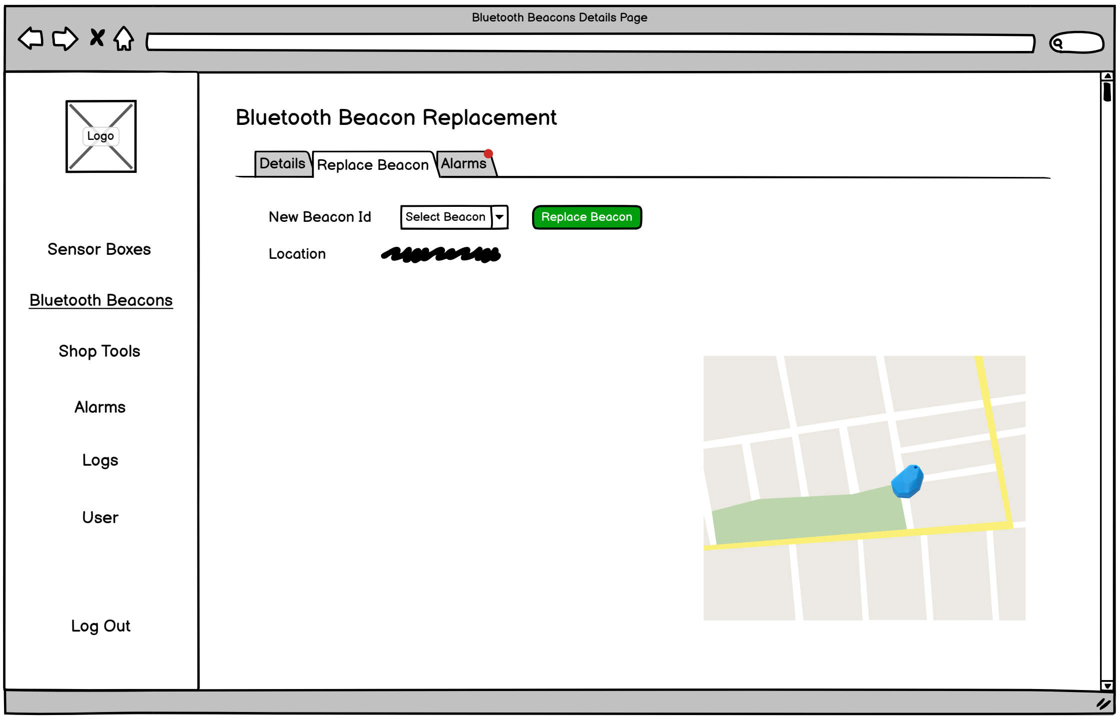


Figure B.11: BLE beacon replacement page.

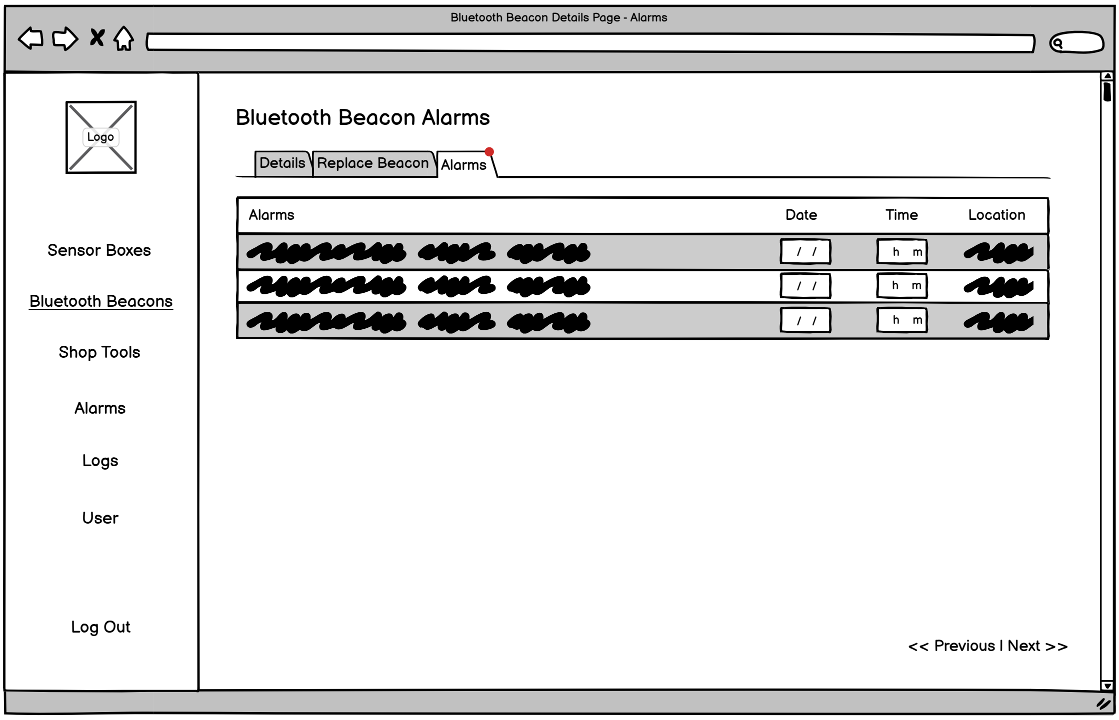


Figure B.12: BLE beacons alarms page.

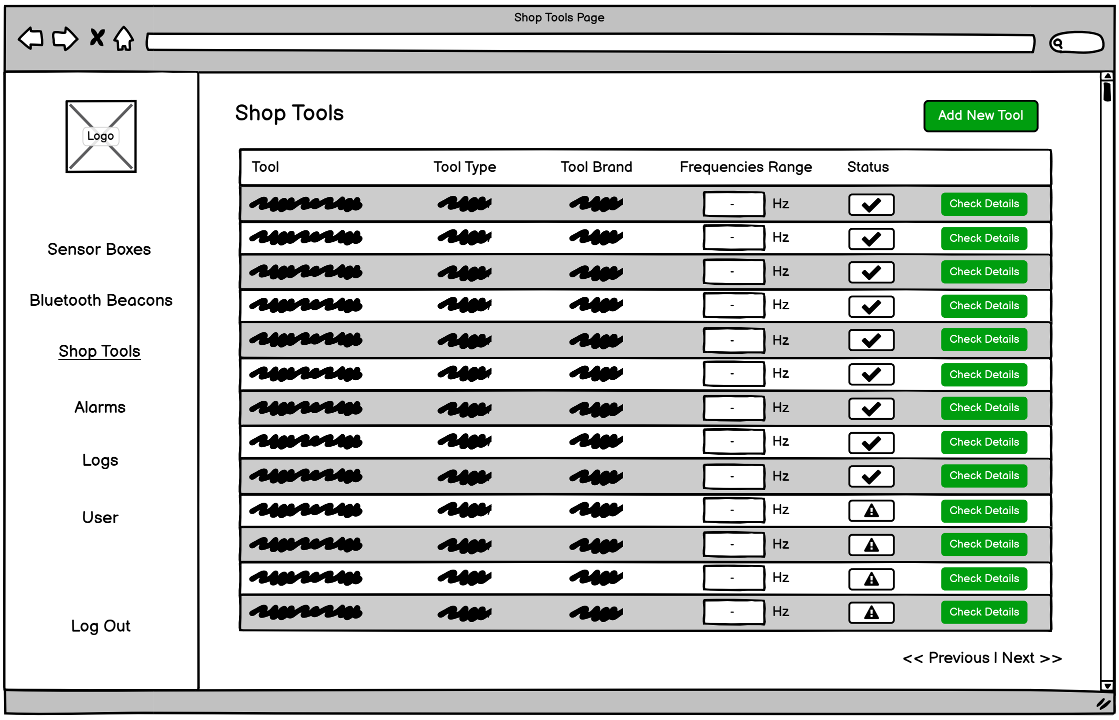


Figure B.13: Workshop tools page.

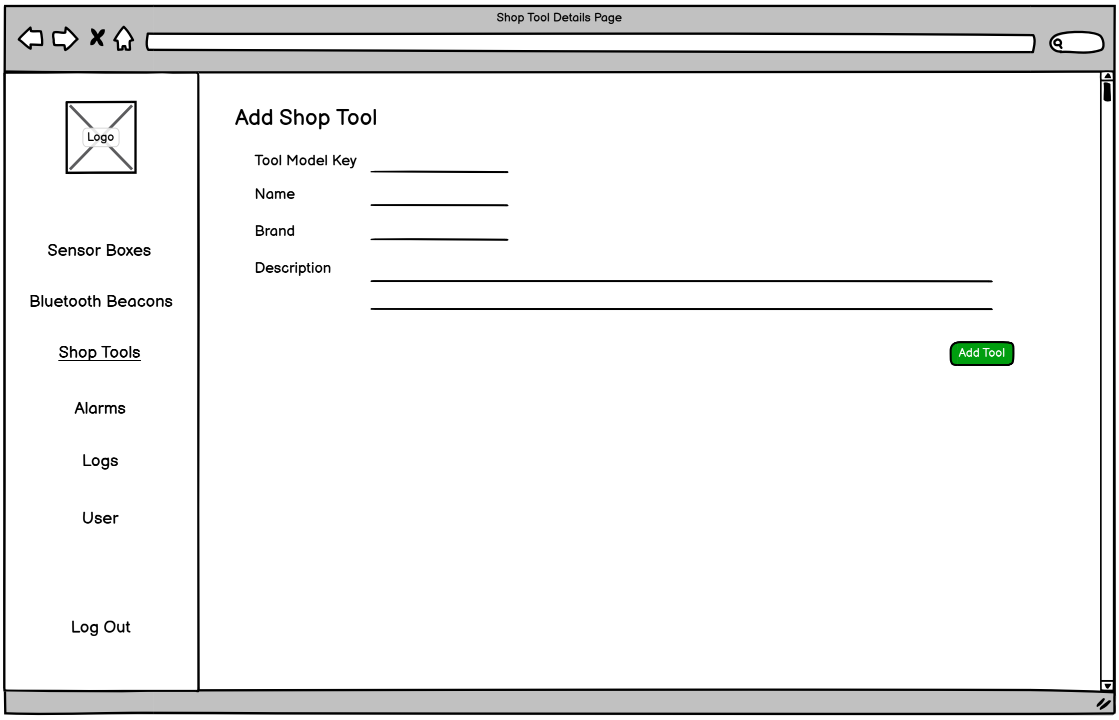


Figure B.14: Add workshop tool page.

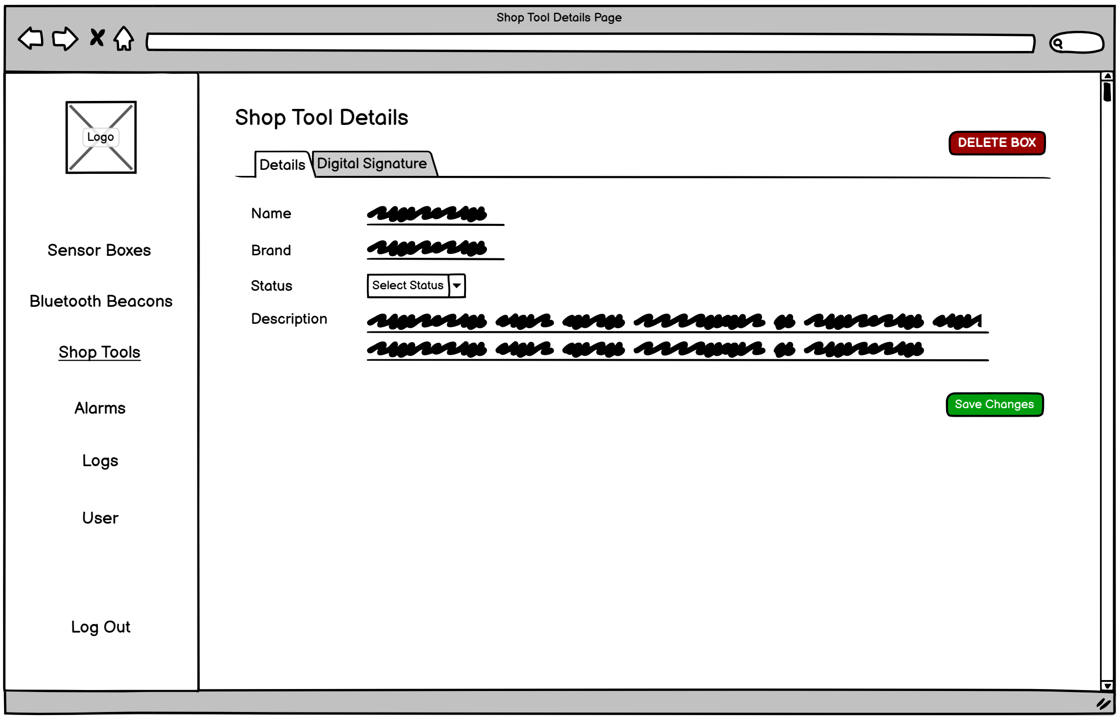


Figure B.15: Workshop tool details page.

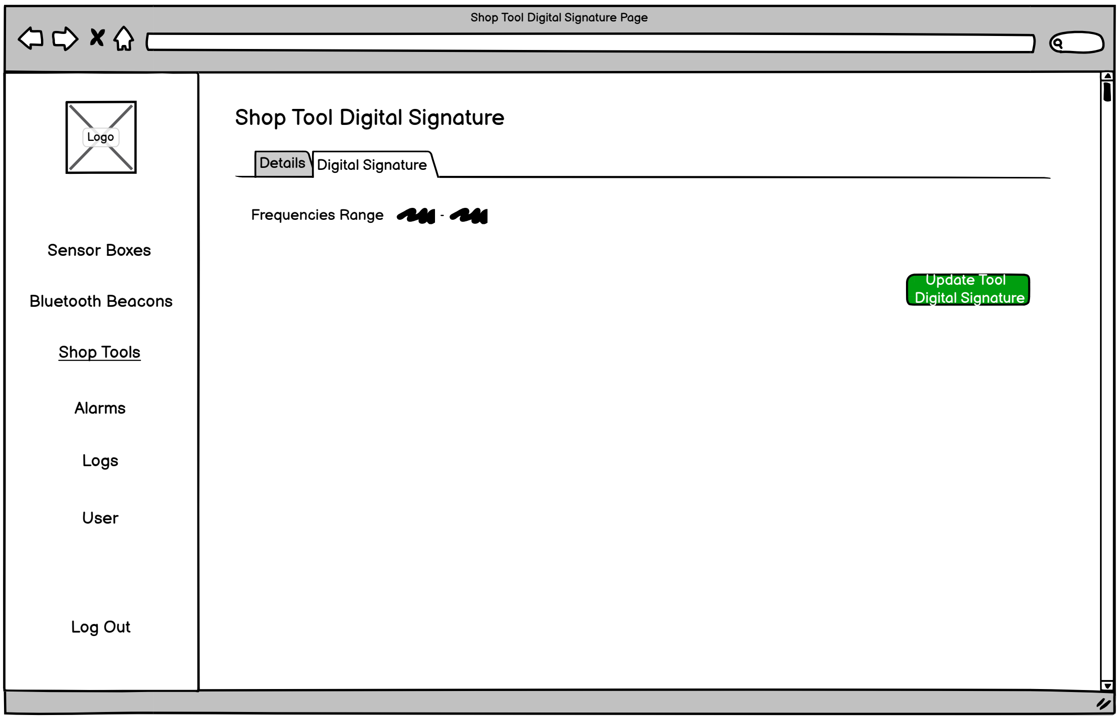


Figure B.16: Workshop tool frequency digital signature page.

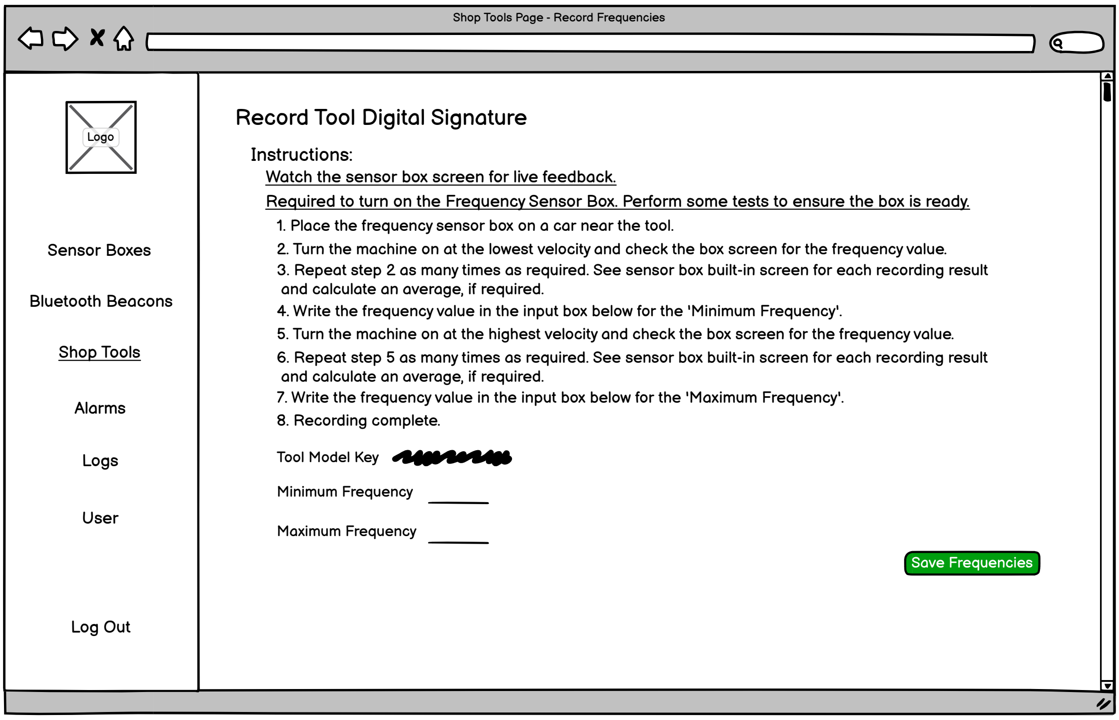


Figure B.17: Workshop tool digital signature update page.

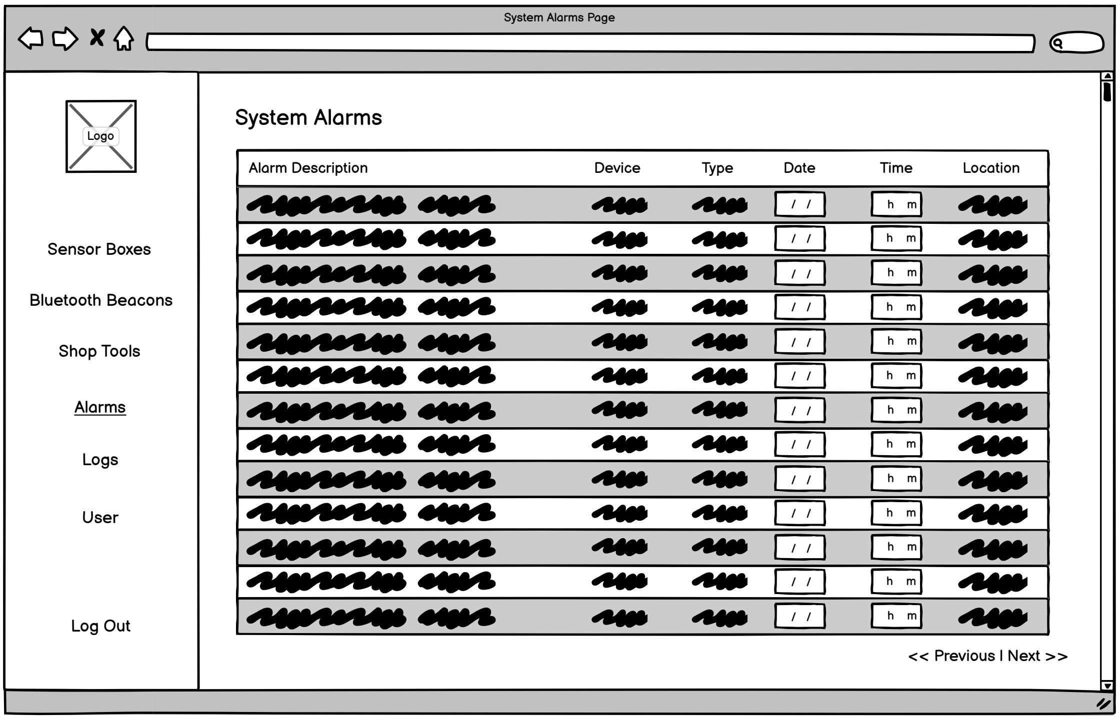


Figure B.18: System alarms page.

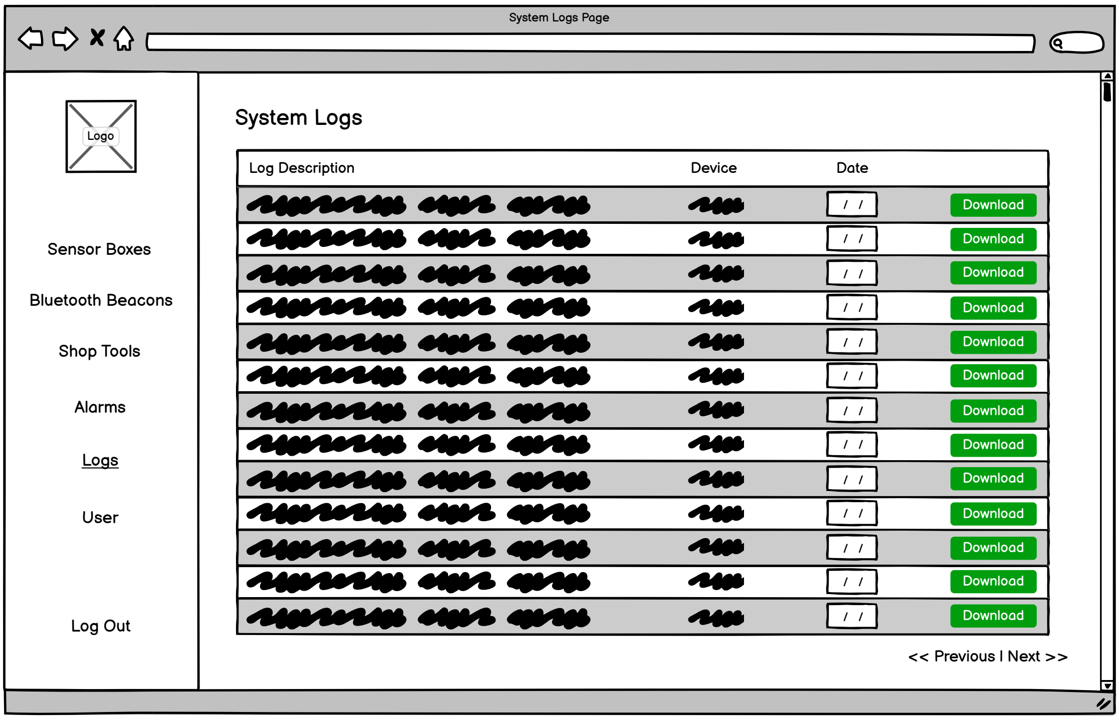


Figure B.19: Sensor boxes logs page.

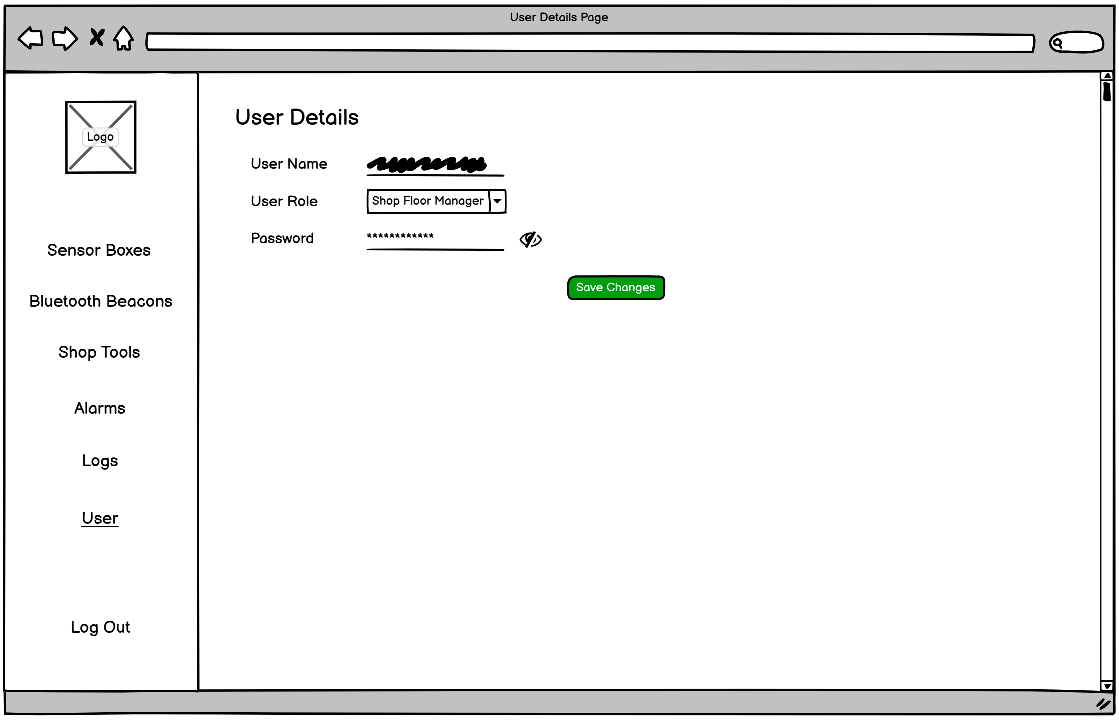
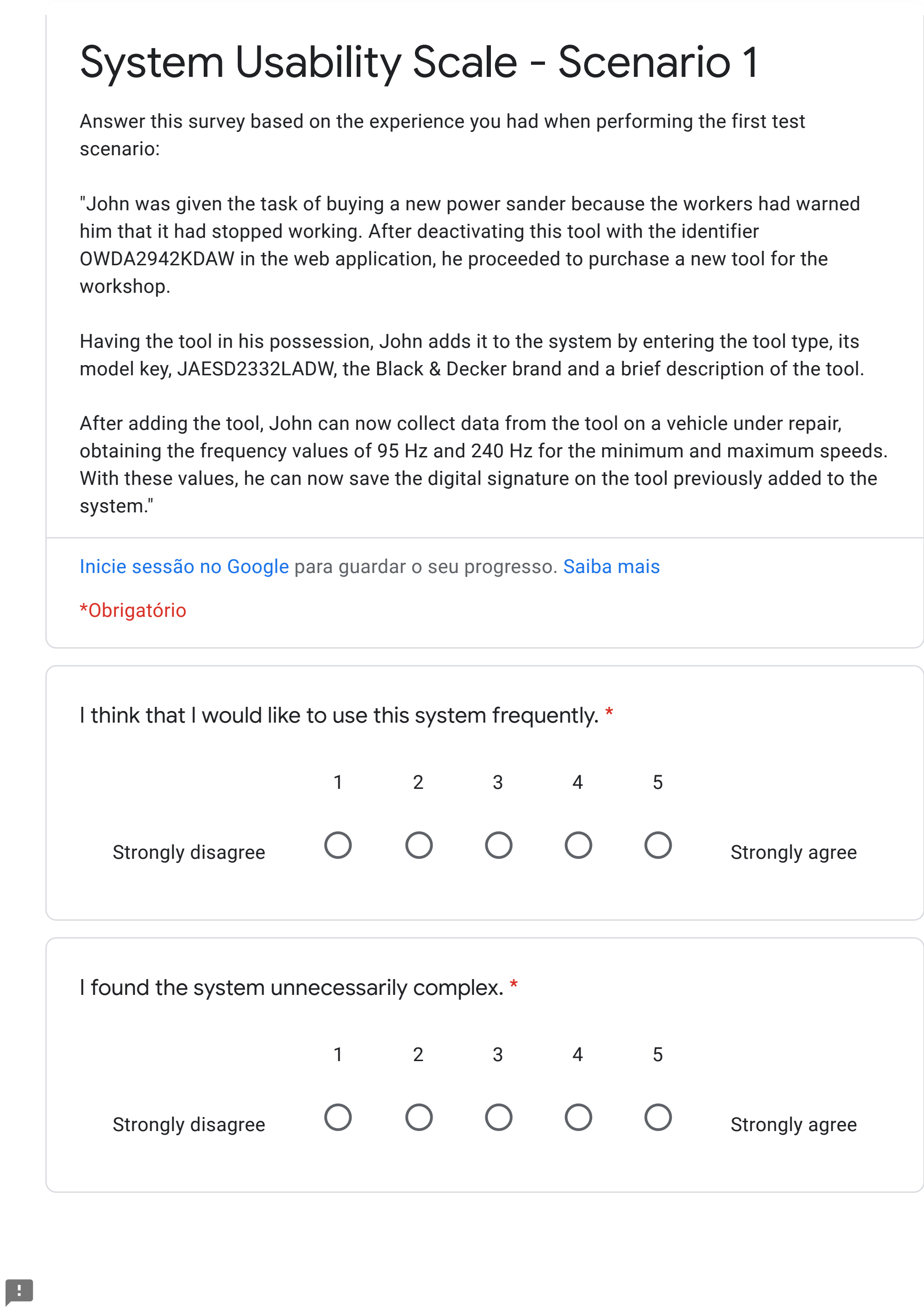


Figure B.20: User page.

B.4 Surveys

B.4.1 System Usability Scale Survey



System Usability Scale - Scenario 1

Answer this survey based on the experience you had when performing the first test scenario:

"John was given the task of buying a new power sander because the workers had warned him that it had stopped working. After deactivating this tool with the identifier OWDA2942KDAW in the web application, he proceeded to purchase a new tool for the workshop.

Having the tool in his possession, John adds it to the system by entering the tool type, its model key, JAESD2332LADW, the Black & Decker brand and a brief description of the tool.

After adding the tool, John can now collect data from the tool on a vehicle under repair, obtaining the frequency values of 95 Hz and 240 Hz for the minimum and maximum speeds. With these values, he can now save the digital signature on the tool previously added to the system."

Inicie sessão no Google para guardar o seu progresso. Saiba mais

*Obrigatório

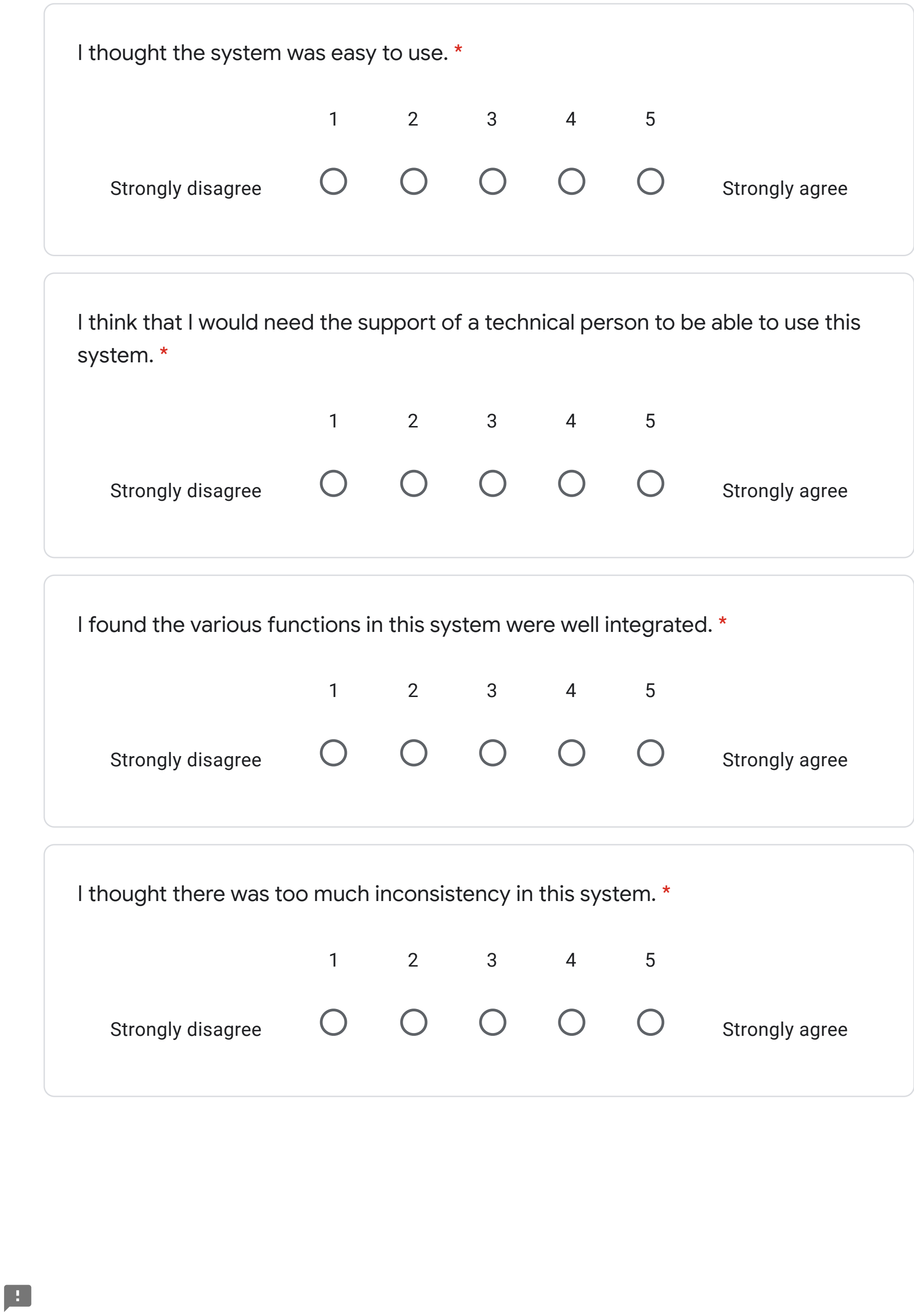
I think that I would like to use this system frequently. *

	1	2	3	4	5	
Strongly disagree	○	○	○	○	○	Strongly agree

I found the system unnecessarily complex. *

	1	2	3	4	5	
Strongly disagree	○	○	○	○	○	Strongly agree

Figure B.21: SUS survey shown after the first test scenario, page one.



I thought the system was easy to use. *

	1	2	3	4	5	
Strongly disagree	○	○	○	○	○	Strongly agree

I think that I would need the support of a technical person to be able to use this system. *

	1	2	3	4	5	
Strongly disagree	○	○	○	○	○	Strongly agree

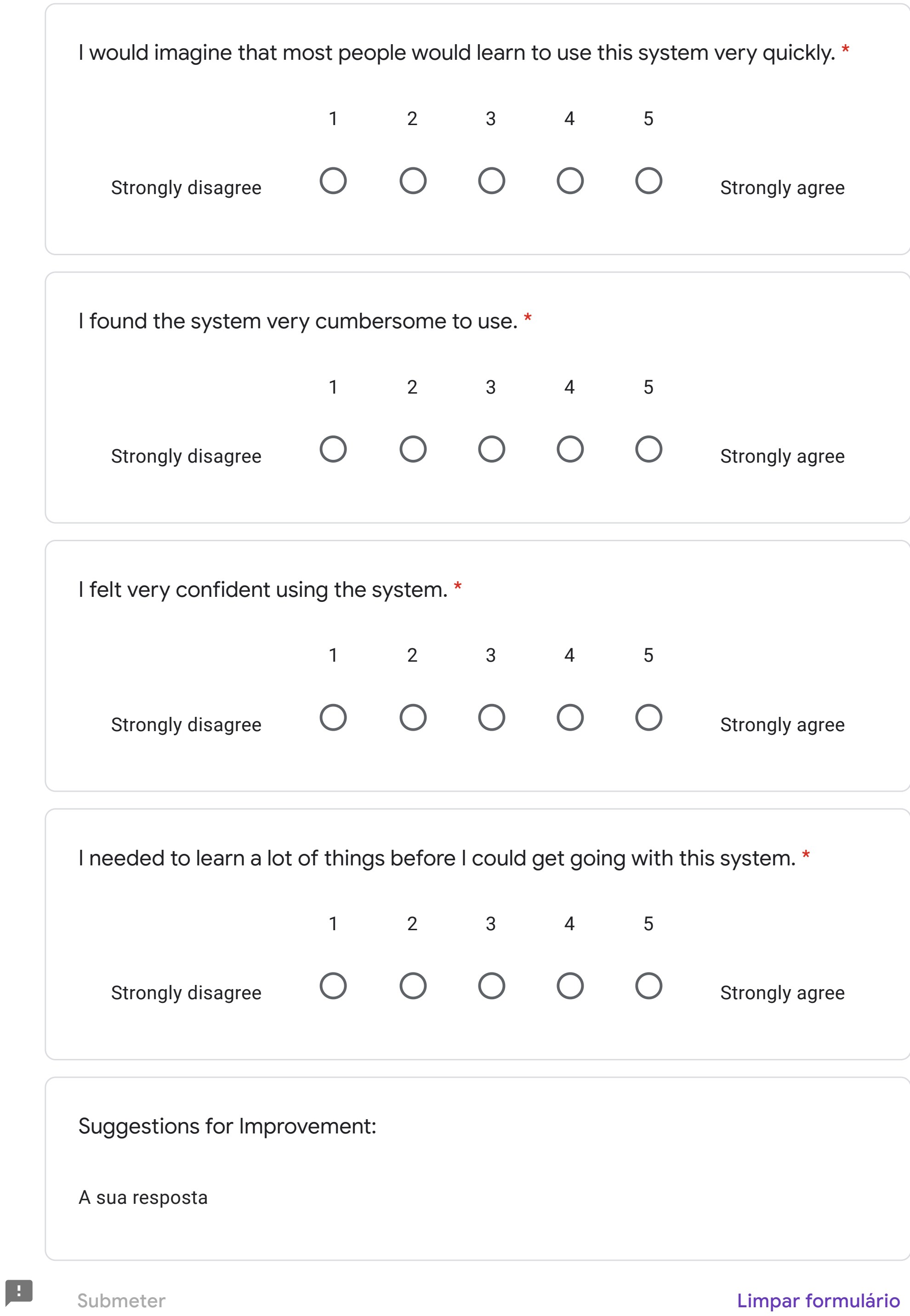
I found the various functions in this system were well integrated. *

	1	2	3	4	5	
Strongly disagree	○	○	○	○	○	Strongly agree

I thought there was too much inconsistency in this system. *

	1	2	3	4	5	
Strongly disagree	○	○	○	○	○	Strongly agree

Figure B.22: SUS survey shown after the first test scenario, page two.



I would imagine that most people would learn to use this system very quickly. *

	1	2	3	4	5	
Strongly disagree	○	○	○	○	○	Strongly agree

I found the system very cumbersome to use. *

	1	2	3	4	5	
Strongly disagree	○	○	○	○	○	Strongly agree

I felt very confident using the system. *

	1	2	3	4	5	
Strongly disagree	○	○	○	○	○	Strongly agree

I needed to learn a lot of things before I could get going with this system. *

	1	2	3	4	5	
Strongly disagree	○	○	○	○	○	Strongly agree

Suggestions for Improvement:

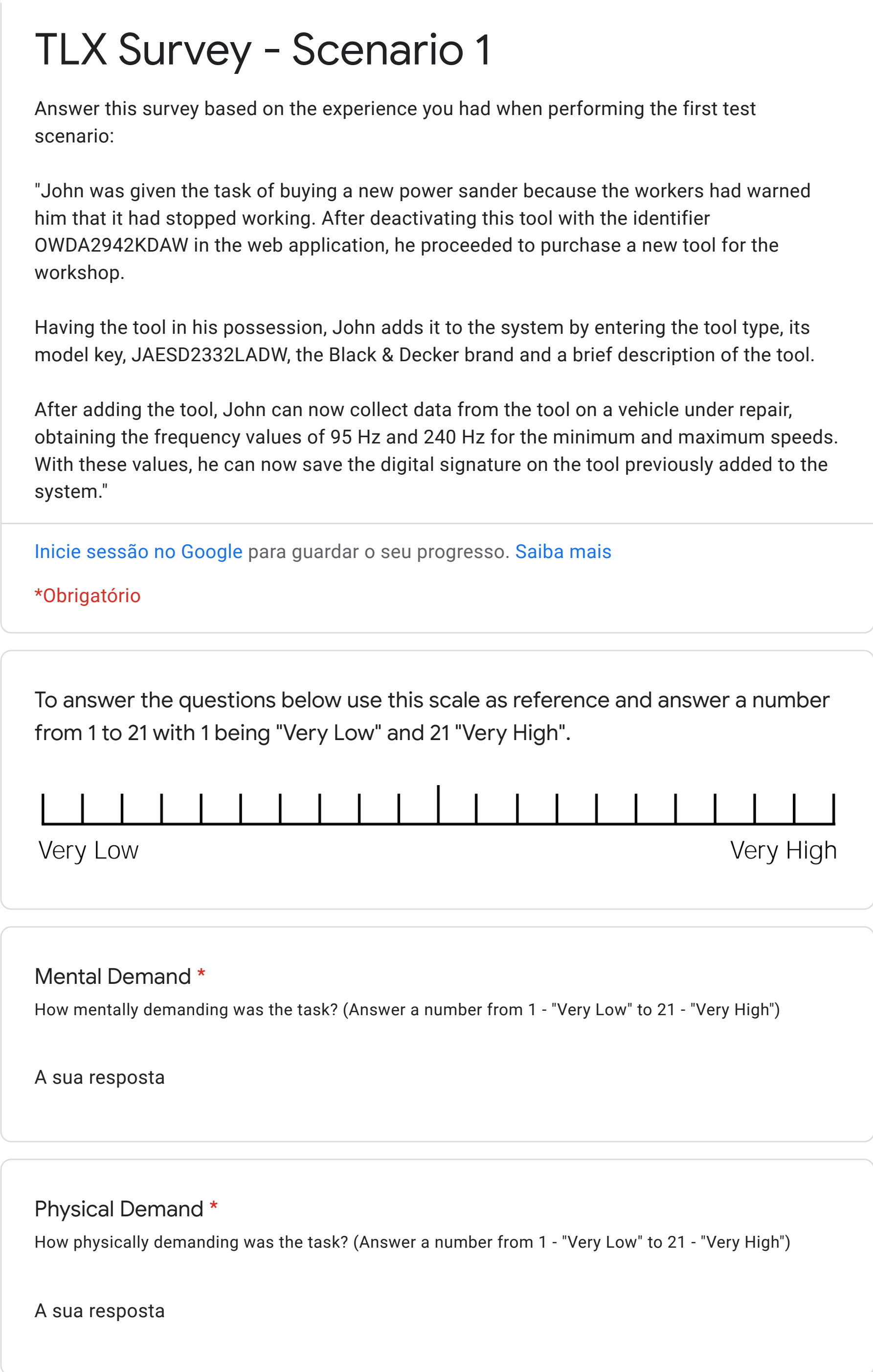
A sua resposta

Submeter

Limpar formulário

Figure B.23: SUS survey shown after the first test scenario, page three.

B.4.2 NASA-TLX Survey



TLX Survey - Scenario 1

Answer this survey based on the experience you had when performing the first test scenario:

"John was given the task of buying a new power sander because the workers had warned him that it had stopped working. After deactivating this tool with the identifier OWDA2942KDAW in the web application, he proceeded to purchase a new tool for the workshop.

Having the tool in his possession, John adds it to the system by entering the tool type, its model key, JAESD2332LADW, the Black & Decker brand and a brief description of the tool.

After adding the tool, John can now collect data from the tool on a vehicle under repair, obtaining the frequency values of 95 Hz and 240 Hz for the minimum and maximum speeds. With these values, he can now save the digital signature on the tool previously added to the system."

Inicie sessão no Google para guardar o seu progresso. Saiba mais

*Obrigatório

To answer the questions below use this scale as reference and answer a number from 1 to 21 with 1 being "Very Low" and 21 "Very High".

Very Low Very High

Mental Demand *

How mentally demanding was the task? (Answer a number from 1 - "Very Low" to 21 - "Very High")

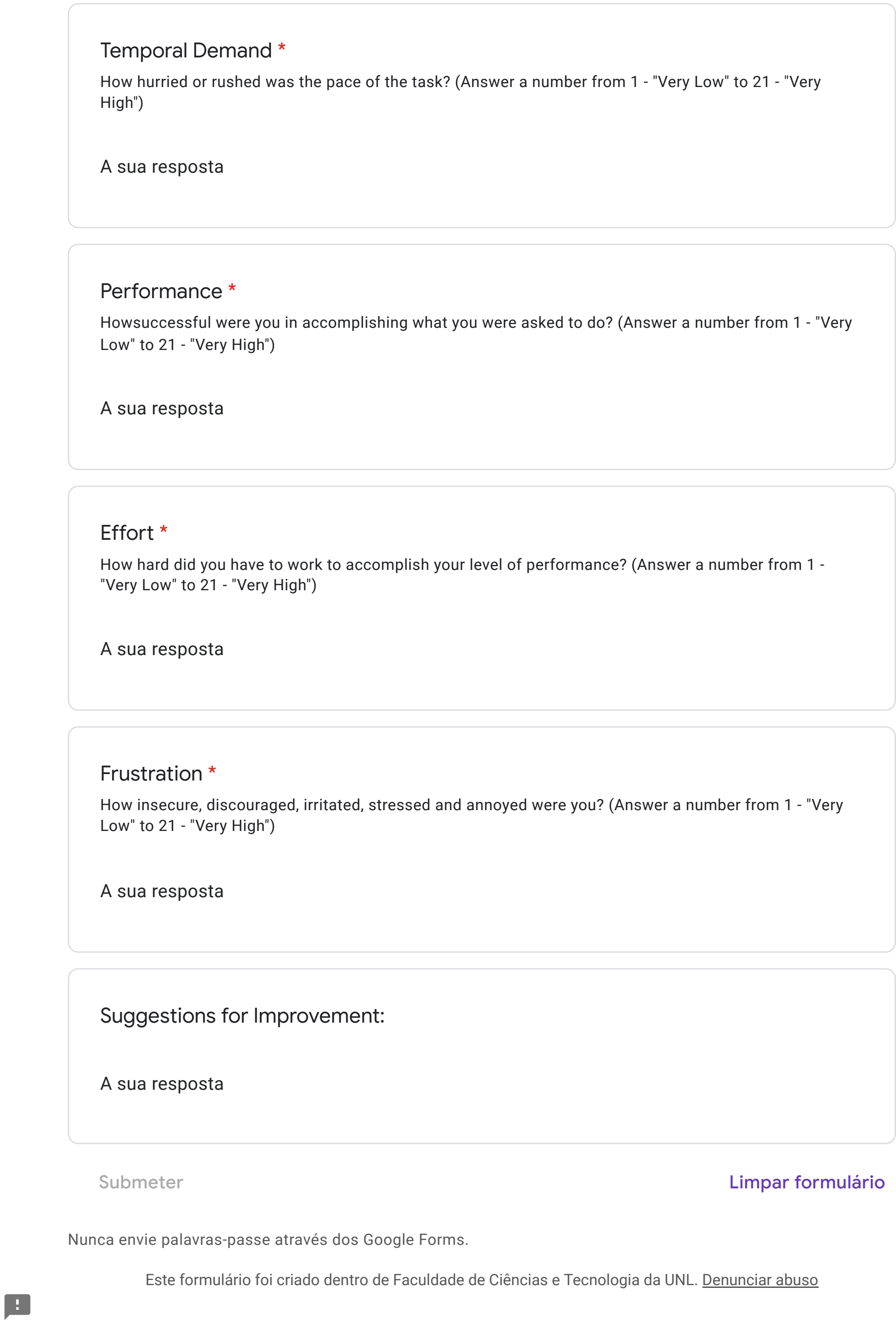
A sua resposta

Physical Demand *

How physically demanding was the task? (Answer a number from 1 - "Very Low" to 21 - "Very High")

A sua resposta

Figure B.24: TLX survey shown after the first test scenario, page one.



Temporal Demand *

How hurried or rushed was the pace of the task? (Answer a number from 1 - "Very Low" to 21 - "Very High")

A sua resposta

Performance *

Howsuccessful were you in accomplishing what you were asked to do? (Answer a number from 1 - "Very Low" to 21 - "Very High")

A sua resposta

Effort *

How hard did you have to work to accomplish your level of performance? (Answer a number from 1 - "Very Low" to 21 - "Very High")

A sua resposta

Frustration *

How insecure, discouraged, irritated, stressed and annoyed were you? (Answer a number from 1 - "Very Low" to 21 - "Very High")

A sua resposta

Suggestions for Improvement:

A sua resposta

Submeter   Limpar formulário

Nunca envie palavras-passe através dos Google Forms.

Este formulário foi criado dentro de Faculdade de Ciências e Tecnologia da UNL. Denunciar abuso

Figure B.25: TLX survey shown after the first test scenario, page two.