VIETNAM NATIONAL UNIVERSITY HO CHI MINH CITY
HO CHI MINH CITY UNIVERSITY OF TECHNOLOGY
FACULTY OF COMPUTER SCIENCE AND ENGINEERING

CAPSTONE PROJECT

PROGRAMMING DIGITAL TWINS FOR PRODUCT LIFECYCLE MANAGEMENT AND SUPPLY CHAIN MANAGEMENT

Major: Computer Science

THESIS COMMITTEE: COMPUTER SCIENCE-3

SUPERVISORS: **Dr. LÊ LAM SƠN**
Assoc Prof. LÊ HỒNG TRANG

REVIEWER: **Dr. PHAN TRỌNG NHÂN**

—o0o—

STUDENT: Ủ MINH QUÂN (1911940)

HO CHI MINH CITY, 10/2023

TRƯỜNG ĐẠI HỌC QUỐC GIA TP.HCM
TRƯỜNG ĐẠI HỌC BÁCH KHOA

CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM
Độc lập – Tự do – Hạnh phúc

KHOA:KH & KT Máy tính
BỘ MÔN:KHMT

NHIỆM VỤ LUẬN VĂN/ ĐỒ ÁN TỐT NGHIỆP

Chú ý: Sinh viên phải dán tờ này vào trang nhất của bản thuyết trình

HỌ VÀ TÊN: **Ủ MINH QUÂN** MSSV: **1911940**
NGÀNH: **KHOA HỌC MÁY TÍNH** LỚP: **CC19KHM1**

1. Đầu đề luận văn/ đồ án tốt nghiệp:: Programming digital twins for product lifecycle management and supply chain management.

2. Nhiệm vụ (yêu cầu về nội dung và số liệu ban đầu):

Companies in different industries have already started applied digital twin technology to product development, manufacturing, and through-life support. It is the goal of this thesis to investigate the concept of twinning business processes for the sake of process monitoring and process analysis. Domain of choice could be supply chain management.

Nhiệm vụ (yêu cầu về nội dung và số liệu ban đầu):

- Get acquainted to digital twin technology.
- Perform data conditioning for the datasets in your domain of choice.
- Get started with coding and prototyping a digital twin.

3. Ngày giao nhiệm vụ: 03/06/2023

4. Ngày hoàn thành nhiệm vụ: 31/07/2023

5. Họ tên giảng viên hướng dẫn:

1. PSG. TS. **LÊ HỒNG TRANG**
2. TS. **LÊ LAM SƠN**

Nội dung và yêu cầu LVTN/ ĐATN đã được thông qua Bộ môn.

Ngày .... Tháng........ Năm ....

CHỦ NHIỆM BỘ MÔN	**GIẢNG VIÊN HƯỚNG DẪN CHÍNH**
(Kí và ghi rõ họ tên)	*(Kí và ghi rõ họ tên)*

PHẦN DÀNH CHO KHOA, BỘ MÔN:

Người duyệt (chấm sơ bộ):

Đơn vị:

Ngày bảo vệ:

Điểm tổng kết:

Nơi lưu trữ LVTN/ĐATN:

TRƯỜNG ĐẠI HỌC BÁCH KHOA
KHOA KH & KT MÁY TÍNH

CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM
Độc lập - Tự do - Hạnh phúc

Ngày tháng năm

PHIẾU ĐÁNH GIÁ LUẬN VĂN/ ĐỒ ÁN TỐT NGHIỆP

(Dành cho người hướng dẫn)

1. Họ và tên SV: Ủ MINH QUÂN
MSSV: 1911940 Ngành (chuyên ngành): Khoa học máy tính

2. Đề tài: Programming digital twins for product lifecycle management and supply chain management.

3. Họ tên người hướng dẫn/phản biện:
1. PGS. TS. **LÊ HỒNG TRANG**
2. TS. **LÊ LAM SƠN**

4. Tổng quát về bản thuyết minh:

Số trang:	Số chương:
Số bảng số liệu	Số hình vẽ:
Số tài liệu tham khảo:	Phần mềm tính toán:
Hiện vật (sản phẩm)	

5. Những ưu điểm chính của LV/ ĐATN:
The student demonstrates his understanding of the theory, principles and practices of digital twins in general, and the digital replicas of a business process in particular. The prototype he implemented enables the monitoring of a business process by displaying its spatio temporal information while at the same offering a conceptual representation of the said process in BPMN. This would open the door to advanced analytics being done on the respective process: process estimation, bottle neck analysis, etc.

6. Những thiếu sót chính của LV/ĐATN:
It is unclear how to obtain the realtime execution log of a business process to feed the prototype implemented by the student.

7. Đề nghị: Được bảo vệ □ Bổ sung thêm để bảo vệ □ Không được bảo vệ □

8. Các câu hỏi SV phải trả lời trước Hội đồng:

9. Đánh giá chung (bằng chữ: Xuất sắc, Giỏi, Khá, TB): Điểm : /10

Ký tên (ghi rõ họ tên)

TRƯỜNG ĐẠI HỌC BÁCH KHOA
KHOA KH & KT MÁY TÍNH

CỘNG HÒA XÃ HỘI CHỦ NGHĨA VIỆT NAM
Độc lập - Tự do - Hạnh phúc

Ngày 25 tháng 09 năm 2023

PHIẾU CHẤM BẢO VỆ LVTN

(Dành cho người phản biện)

1. Họ và tên SV: Ủ Minh Quân
MSSV: 1911940 Ngành (chuyên ngành): Khoa học Máy tính
2. Đề tài: Programming digital twins for product lifecycle management and supply chain management
3. Họ tên người phản biện: TS. Phan Trọng Nhân
4. Tổng quát về bản thuyết minh:
Số trang: Số chương:
Số bảng số liệu Số hình vẽ:
Số tài liệu tham khảo: Phần mềm tính toán:
Hiện vật (sản phẩm)
5. Tổng quát về các bản vẽ:
- Số bản vẽ: Bản A1: Bản A2: Khổ khác:
- Số bản vẽ vẽ tay Số bản vẽ trên máy tính:
6. Những ưu điểm chính của LVTN:
- The student took effort to understand and investigate the concepts of supply chain management and apply digital twins for it. In general, the main contributions include: (1) setting up near real-time IoT devices for simulation; (2) building a web application to show alerts and support decision making. The business scope is from supply stores to bakeries with trucks.

7. Những thiếu sót chính của LVTN:
- The system processing flow is hard-code for the demo flow (i.e., if we want to see alerts from truck, we have to modify the code), so it is hard for the system to scale and adapt to other models of supply chain.
- The evaluation is only for UI, not for the efficiency of product life cycle management and supply chain management.
- The information presentation on UIs is not well-organized (e.g., information shown in the form of rows rather than graphical objects for managers).
- The report should be double-checked with references and citations.

8. Đề nghị: Được bảo vệ ☑ Bổ sung thêm để bảo vệ □ Không được bảo vệ □
9. 3 câu hỏi SV phải trả lời trước Hội đồng:

10. Đánh giá chung (bằng chữ: giỏi, khá, TB): Pretty Good Điểm: 7.3/10

Ký tên (ghi rõ họ tên)

Phan

Phan Trọng Nhân

STATEMENT OF ORIGINALITY

This is to certify that to the best of my knowledge, the content of this thesis is my own work at the Faculty of Computer Science and Engineering (CSE), Ho Chi Minh University of Technology (HCMUT) under the guidance of Doctor Le Lam Son and Associate Professor Le Hong Trang. This thesis has not been submitted for any degree or other purposes.

I certify that the intellectual content of this thesis is the product of my own work and that all the assistance received in preparing this thesis and sources have been acknowledged.

ACKNOWLEDGEMENT

Firstly, I would like to express my sincere gratitude to my advisor, Doctor Le Lam Son and Associate Professor Le Hong Trang for their support and help throughout every step in this process. Besides my advisor, I would like to thank all my teachers at the Faculty of Computer Science & Engineering, Ho Chi Minh University of Technology for providing me with knowledge necessary for the completion of my capstone project.

Finally, I would like to thank my family for their unconditional support and encouragement throughout this course. Without their help, this accomplishment would have been impossible.

ABSTRACT

The thesis “Programming digital twins for product lifecycle management and supply chain management” was research and conducted by student U Minh Quan under the guidance of Doctor Le Lam Son and Associate Professor Le Hong Trang during the course CO4337 Capstone Project in Ho Chi Minh University of Technology.

Companies in different industries have already started applying digital twins to product development, manufacturing, and through-life support. It is the goal of this thesis to investigate how a supply chain digital twin can help in improving product lifecycle management (PLM) and supply chain management (SCM) by creating a prototype digital twin application.

Incorporating research and gathering data from relevant supply chains, this thesis aims to create a digital twin that can represent a real-world supply chain. Using a case study of a bakery supply chain, the results show that the supply chain digital twin developed can maintain a real-time connection with the participants in its real-world counterpart using internet-of-things (IoT) devices, as well as provide insight on the current supply chain. The result twin can also detect failure and give warning to all the users in the organization making it highly valuable in maintaining supply chain resilience.

TABLE OF CONTENT

LIST OF FIGURES

1. INTRODUCTION

1.1 Motivation

1.1.1 Modern supply chain

One of the most significant paradigms shifts of modern business management is that individual businesses no longer compete solely as autonomous entities, but rather as supply chains (Lambert, 2000). We are now entering the era of internetwork competition for business management. It is now suppliers-brand-store versus suppliers-brand-store, or supply chain versus supply chain, as opposed to brand versus brand or store versus store. The final success of the single business will depend on management's capacity to integrate the complex web of commercial ties within this newly competitive market. Thus, the rise for a management philosophy that not only impacts a single company but all its suppliers, customers and all the firms that participate in its supply chain. One such philosophy is supply chain management (SMC).

However, as business grows and the supply chain extend, it has become increasingly difficult for manager to manage the whole supply chain. Firstly, due to the connected nature of a supply chain, a disrupt at any size or level can cause progressively larger fluctuations to other members of the supply chain (bullwhip effect) (Yanfeng et al., 2009). Therefore, managers must not only keep track of their company supply chain but also oversee all members in supply chain. Secondly, because of COVID-19, many supply chains have been massively disrupted due to economic slowdown and social issues. According to surveys by using data from Current Population Surveys (CPS), during the initial stages of the pandemic and the forced lockdown's adoption in the middle of March 2020, small businesses that had been running successfully for a while experienced a sharp fall in activity. During the month of February, there were 15 million active enterprises; this figure sharply decreased to 11 million during the pandemic's early stages (Kalogiannidis, 2020). The pandemic requires modern supply chain to be able to quickly adapt and respond to changes. Thirdly, as supply chain extend, the cost of changing a policy or implement a new tactic become increasingly costly. This requires the manager to plan extensively to avoid changes as much as possible.

1.1.2 Industry 4.0

With these problems, managing a modern supply chain has become an extensively difficult challenge. However, the arrival of industry 4.0 give rise to many new and exciting technologies, one of which are digital twin, based on a concept that emerged more than 40 years ago. In simple terms, digital twin is a digital replica of a real word entity (e.g., car, bridge or house). The technology has been used for a long time in manufacturing or construction to provide insight and perform analysis on a virtual environment. Ford Motor company have been using digital twin technology to accurately detect energy losses, pinpoint areas where energy can be conserved, and improve the overall performance of production lines (better buildings).

Although the concept of digital twin has been around for more than three decades, not until the early 2000s that there has been a sharp rise in interest in digital twin technology. And with that emerges the concept of a digital twin for a process (e.g., supply chain, business process). According to Microsoft, new technologies, such as low-cost connectivity, mixed reality, and artificial intelligence enabled the digital twin to be more affordable and thus help it increase in popularity. Moreover, with the occurrence of the COVID-19 pandemic, which make companies realize that the tradition process of managing a supply chain is not effective enough, all contribute to the rise of digital twin in recent years.

Consider all the above arguments, this thesis aims to investigate how a digital twin can help in making supply chain resilient and how it can provide insight as well as help companies dealt with unforeseeable scenarios.

1.2 Goal and scope of the project

1.2.1 Goal

This thesis focusses on exploring the concept of a supply chain digital twin as well as the implement of the sample digital twin. The result of this thesis will be a generalize idea for what can a digital twin help to improve in supply chain management as well as a sample digital twin application. The result digital twin must have the following requirements:

- The result application can run on popular internet browsers (e.g., Chrome, Microsoft Edge, and Mozilla Firefox).
- The digital twin can represent a real-world supply chain and its participants.
- The digital twin must be connected to its physical counterpart by a series of Internet-of-things (IoT) devices.
- The application can alert user if something malfunctions and track a problem.

1.2.2 Scope

Because this thesis is made in the area of software engineering, the main scope of this thesis will be to develop and understand the supply chain digital twin. Supply chain managements is a large topic with many theories about how to set up the chain and managing it. The concept of supply chain and supply chain management will be introduced as catalyst to help in defining a supply chain digital twin, therefore subject about how to improve supply chain or the advanced concept supply chain management will not be focused on.

A modern supply chain is very complex and not every supply chain structure is the same. The modern basic structure of supply chain is therefore hard to define and remains a large topic for another major. Because of that developing a digital twin for a general use case of every supply chain is not possible at this current state. For that reason, in this thesis, the result digital twin will build upon a company that specialized in making cake and bread, which own a number of bakeries located in France. The supply chain structure of the company is based on **Figure 1**. Bakery supply chain participants.

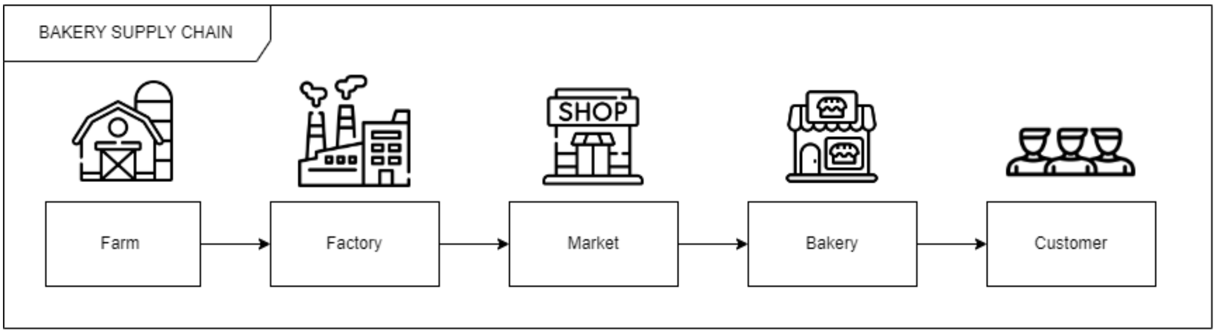


Figure 1. Bakery supply chain participants

During this thesis, all use-case and test-case will be made based on this company supply chain which will be explained in section 2.1.3.

1.3 Thesis structure

The thesis included 7 chapters. Chapter 1 give a brief summary of what the thesis is about and provide the thesis structure. Chapter 2 will focus investigate the concept of supply chain management and digital twin, provided with their concept and use case. Chapter 2 will also give literature review about digital twin research made over the years. Chapter 3 will define the overall structure of the result digital twin and its related technologies. Chapter 4 will be the presentation of the developed digital twin as well as its performance in provided use cases. Chapter 5 will be focus on testing and analyze the performance of the result twin. Chapter 6 will give a general conclusion based on conducted studies and proposed ideas for future research.

1.4 Limitations

The time-series data used in this thesis is incomplete as some of the features are hidden because of its confidential nature, because of that the time-series data used in this thesis is composed of data in other dataset that have been transformed to fit the used data. Although it has been transformed, the data have been tested and proved to be able to give enough insight on how a supply chain digital twin works.

With limited access to funding and IoT devices shops in the area, all the IoT devices in this thesis will be simulated based on an implemented IoT hub device simulator.

Because of the author’s background in Computer Science major, some of the research about supply chain or supply chain management may be incorrect. However, the bakery supply chain use case used to develop the digital twin is correct and proved usable as a simple supply chain model so it is enough to clarify that mistake made in the literature review step of supply chain or supply chain management will not affect the result of the thesis.

2. BACKGROUND AND RELATED WORK

2.1 Supply chain management

2.1.1 Definition

“An army marches on its stomach.” This is a remark made by Napoleon Bonaparte, a French military commander and political leader who rose to prominence during the French Revolution and led successful campaigns during the Revolutionary Wars. This remark shows that he was a master strategist as unlike other generals at the time, he understood what we now would call an efficient supply chain. People can discuss all sorts of grand strategies and dashing maneuvers but none of that will be possible without first figuring out how to meet the day-to-day demands of providing an army with fuel, spare parts, food, shelter, and ammunition. In today's world, supply chain is defined as the connected series of activities which is concerned with planning, coordinating, and controlling material, parts and finished goods from suppliers to the customer (Stevens, 1989). r. It is concerned with two distinct flows through the organization: material and information. The scope of the supply chain begins with the source of supply and ends at the point of consumption.

Although a supply chain may contain many companies, a company is not limited to a single supply chain. In today's world where global sourcing has risen, many companies work together to deliver the end product (Mentzer et al., 2001). Furthermore, companies in particular and supply chains in general compete more today on the basis of time and quality. Getting a defect-free product to the customer faster and more reliably than the competition is no longer seen as a competitive advantage, but simply a requirement to be in the market. Customers are demanding products consistently delivered faster, exactly on time, and with no damage. Each of these necessitates closer coordination with suppliers and distributors. This global orientation and increased performance-based competition, combined with rapidly changing technology and economic conditions, all contribute to marketplace uncertainty. This uncertainty requires greater flexibility on the part of individual companies and supply chains, which in turn demands more flexibility in supply chain relationships.

SCM has risen to popularity in the past ten years (Mentzer et al., 2001) and although despite its popularity, its definition, both in academia and practice, still varied between authors. (Monczka et al., 1998) defined SCM as a concept whose “primary objective is to integrate and manage the sourcing, flow, and control of materials using a total systems perspective across multiple functions and multiple tiers of suppliers.”, while (Cooper et al. 1997) view it as a management philosophy. In this study we will view SCM as a philosophy, a paradigm, a coordination of production, inventory, location and transportation among the participants in the supply chain to achieve the best mix of responsiveness and efficiency for the market being served (Hugos, 2011).

2.1.2 SCM in the 4.0 industry

A fourth industrial revolution which is the computerization of manufacturing systems. It has cyber-physical systems which are the combination of software and production assets. It includes automation, the industrial Internet of Things, data sharing and cloud computing. It has mainly the following characteristics: interoperability, transparency, technical guidance and independent choices (Jayaram, 2016).

The essential components that offer the typical quality for the digital connectivity and communication of the physical and digital elements in the SCs, enabling real-time data storage, analysis, and sharing, might be regarded as the Industry 4.0 enablers and features in the digitalized supply chain. (Ben-Daya et al., 2017). Impact of the industry 4.0 in supply chains are such as increased flexibility, quality standards, efficiency and productivity. These will result in mass customization, allowing companies to meet customers' demands, creating value through constantly introducing new products and services to the market. Additionally, the interaction between machines and people may have a social impact on how future workers live their lives, particularly in relation to the improvement of decision-making.

2.1.3 Business process

A supply chain is very large and complex which comprises of many companies and activities inside of the chain. To keep track of the action inside of a participant of a supply chain, their activities can be defined as a business process. Therefore, we can say that a business process is a collection of linked tasks that find their end in the delivery of a service or product to a client. A business process has also been defined as a set of activities and tasks that, once completed, will accomplish an organizational goal. The process must involve clearly defined inputs and a single output. These inputs are made up of all of the factors that contribute (either directly or indirectly) to the added value of a service or product. These factors can be categorized into management processes, operational processes, and supporting business processes.

2.1.4 Case study

For the development of the digital twin and this thesis, a supply chain use case is proposed. The supply chain is a model of a bakery supply chain, which is the process of making and delivering baked products of a company. The company in the use case is a company located in French and has the supply chain structure as in **Figure 2**. Bakery supply chain participants.

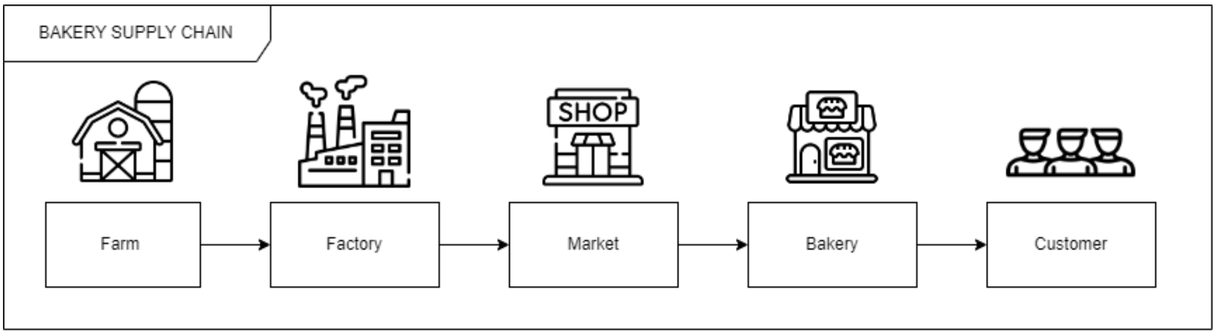


***Figure 2**. Bakery supply chain participants*

The supply chain's main goals will be to deliver dairy products from the start of the supply chain, which is the farm, to the end customer. Each participant in the chain will perform a different process and all participants belong to the bakery company described. The business process will be presented using Business Process Model and Notation (BPMN).

BPMN: It is a graphical notation that depicts all the steps from the start such as gathering raw material to the final step such as the delivering the product to the end customer in a business process. The notation has been specifically designed to coordinate the sequence of processes and the messages that flow between different process participants in a related set of activities. **Figure 3**. BPMN node presented the symbols that will be used in modeling the supply chain.

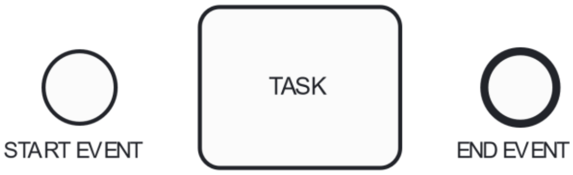


***Figure 3**. BPMN node*

The thesis will use the following symbols in **Figure 3**. BPMN node to depict a BPMN:

- **Start event:** events are triggers that fired once under certain circumstances, and the start event is the signal for the beginning of the business process. All business process starts from the start event.
- **Task:** are activities that are performed in the process by human or automaton (e.g., selling milk, pack product)
- **End event:** is the event that signal the end of a business process.

Farm:

- **Business process:**

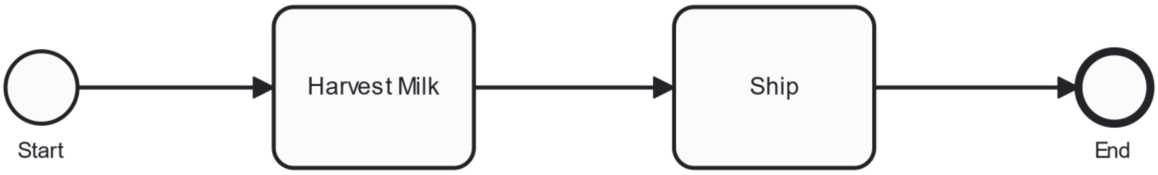


***Figure 4**. Farm BPMN*

- **Function:** The farm act as the start of the supply chain, its function is to harvest milk from cow and then ship the product to the factory to begin the process of making dairy product.

Factory:

- **Business process:**

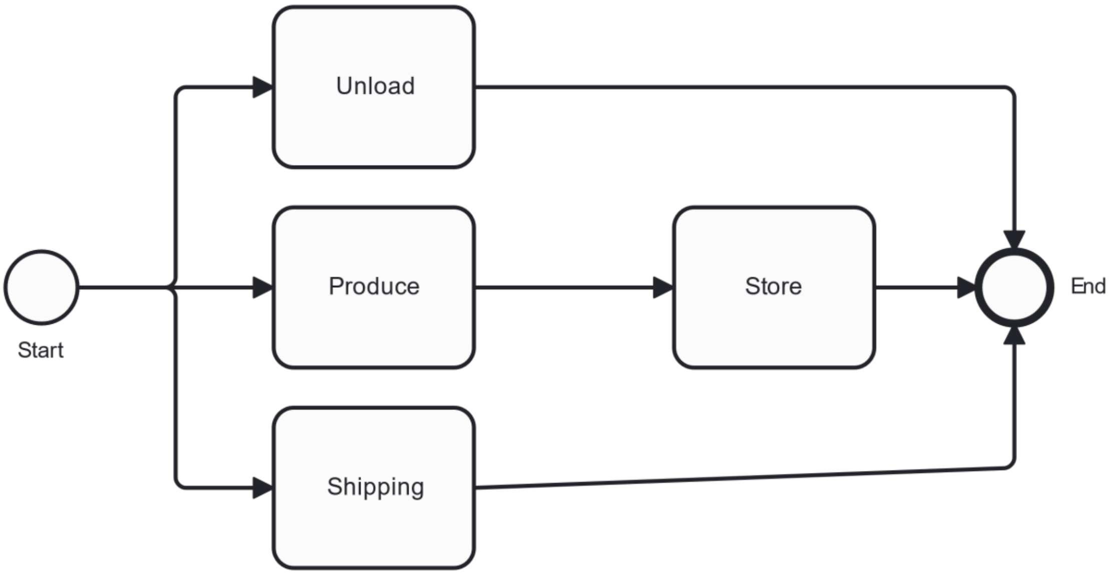


***Figure 5**. Factory BPMN*

- **Function:** The factory is the place where dairy product is made. When it receives raw milk from the farm, it will first unload the cargo, and this process happen concurrently with the process of producing dairy product, all of which will then be stored in the warehouse waiting to be shipped. If the shipping process happened, made product will be sent to the area distribution center.

Market:

- **Business process:**

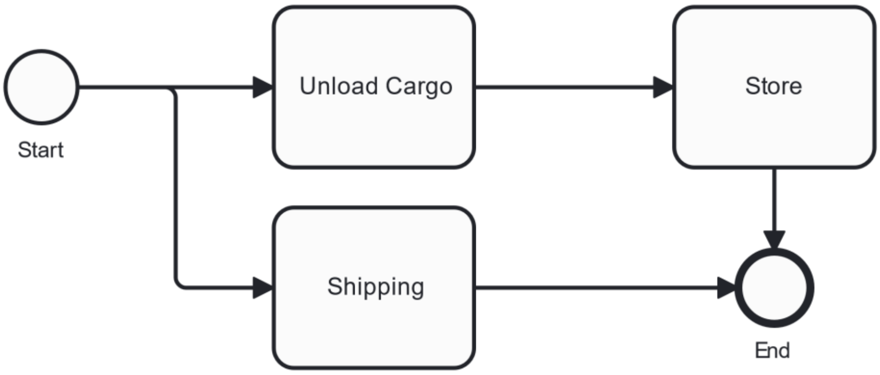


***Figure 6**. Distribution center BPMN*

- **Function:** The market will be the place the product is sold. The market can be an outdoor market or a supermarket depends on the area the product is shipped to or the targeted customers. However the business process will be the same, which includes of unloading the cargo and storing the products inside of a warehouse.

Bakery:

- **Business process:**

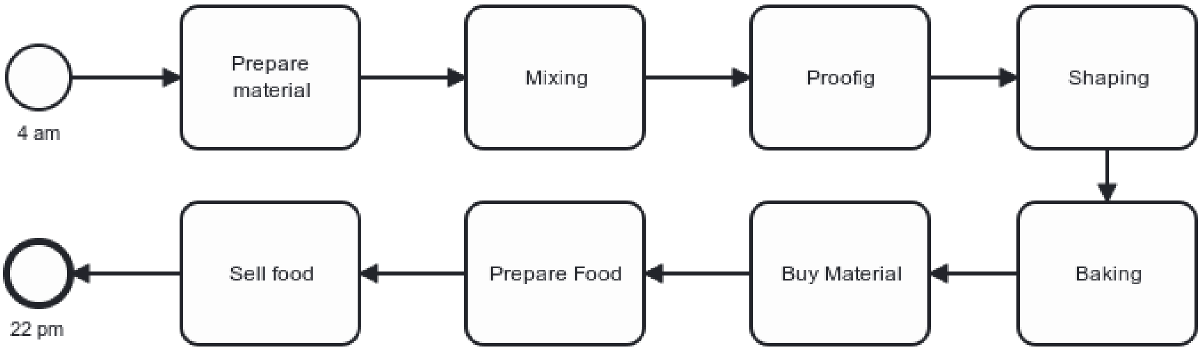


***Figure 7**. Bakery BPMN*

- **Function:** The bakery is the end of the supply chain and will be responsible for making the end product which is the baked goods which will then be sold directly to the end customer. The business process will start with preparing the material for the baking process such as flour, sugar, butter, eggs, and flavorings. Then a commercial baking machine will be used to baked the end products. The process of making cake start with mixing the material together, then proofing will allow the dough to rest and rise before baking. Shaping is an important step in bread-making that comes after the dough has gone through the proofing or fermentation process, refers to creating the desired form or structure of the bread before it is baked. It helps to create a good texture, proper rise, and aesthetic appearance of the final loaf. Once the shaped bread has completed its final rise, it is ready to be baked according to the specific recipe instructions. After that the store will decide if material is needed. If needed, a truck will go to the market and get the required material for tomorrow. After that, the food is then prepared and sold to the end customer.

All participants in the supply chain are as important and relied on each other. In this use-case, the detail supply chain will be as in **Figure 8**. Detail supply chain, which will detail how many participants will be in the chain and how products are shipped to other participants. The supply chain used in this thesis will be based on a bakery company in France. As in **Figure 8**. Detail supply chain, the company consist of 2 bakery which will take their supply from 2 store which is the box and parchment store and the bread material store using truck delivery.

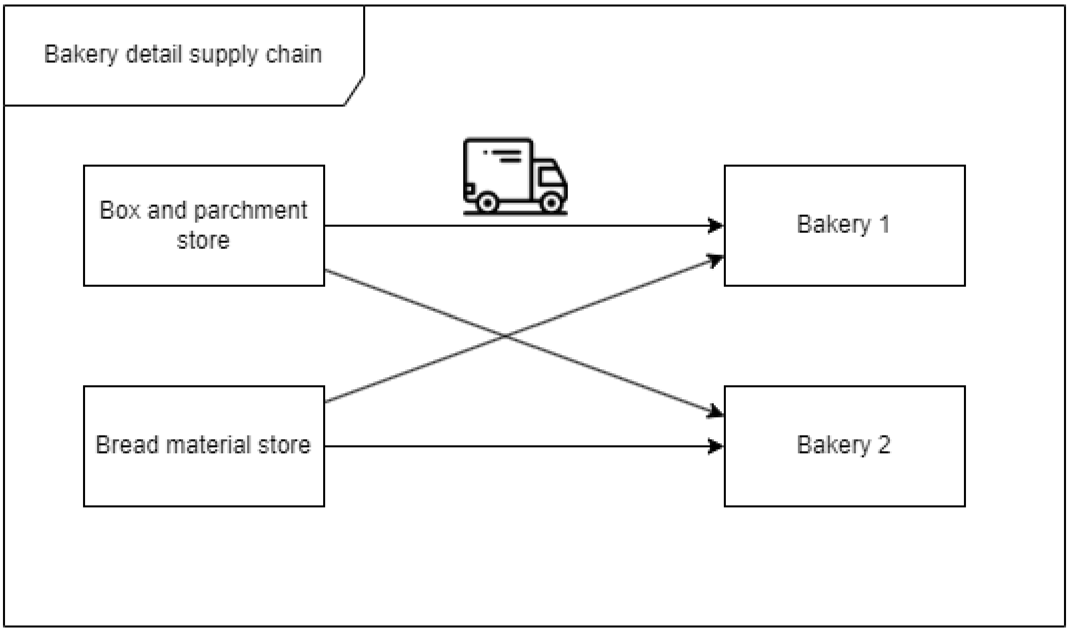


Figure 8. Detail supply chain

2.2 Digital twin

The 'twin' concept was first employed by NASA's Apollo space program. In order for the space vehicle on Earth to reflect, imitate, and forecast the conditions of the other space vehicle in space, the program created two identical space vehicles. The vehicle that completed the mission in space was identical to the one that stayed on Earth.

The term "digital twin" was originally used in the work of Hernández and Hernández. Urban road network design iterative changes were made using a digital twin. However, it is widely acknowledged that the terminology was first introduced as "digital equivalent to a physical product" by Michael Grieves at University of Michigan in 2003. We can define digital twin as follow: ''A Digital Twin is a virtual instance of a physical system (twin) that is continually updated with the latter's performance, maintenance, and health status data throughout the physical system's life cycle.'' (Fuller, 2020)

The basic idea of digital twin is simple, that is linking physical object and digital object in an accurate and real-time manner. Digital twin is not a specific technology, but is an idea that can be implemented by any technology. However, all digital twin must share the 3 core technologies:

- **Data related technologies:** Data is the basis of digital twin. Sensors, gauges, RFID tags and readers, cameras, scanners, etc. should be chosen and integrated to collect total-element data for digital twin. Data then should be transmitted in a real-time or near real-time manner.
- **High-fidelity modeling:** Model is the core of digital twin. Models of digital twin comprise semantic data models and physical models. Semantic data models are trained by known inputs and outputs, using artificial intelligence methods. Physical models require comprehensive understanding of the physical properties and their mutual interaction. Thus, multi-physics modeling is essential for high-fidelity modeling of digital twin.

- **Model based simulations:** Simulation is an important aspect of digital twin. Digital twin simulation enables virtual models to interact with physical entities bi-directionally in real-time.

The use of digital twin varied from many fields such as aerospace, construction and most notable is economy. The ability to give real-time data, predict results and create an environment for testing makes digital twin a valuable asset in the 4.0 industry. Thus within the new age of digital economy emerged a new concept of supply chain digital twin.

2.3 Supply chain digital twin

2.3.1 Overview

Since the coming of the 4.0 industry, digitalization has emerged as a new phenomenon and affects all aspect of life all over the world. More than 90% of internet users have already made online purchases and about 40% of companies has used sophisticated tools for big data analytics. Moreover, in 2020, there are already 26 billion ‘things’ enabled with the ‘Internet of things’.

Digitalization has also changed the way how supply chain structured. The shift to digital supply chain has proved to bring more benefits and create more revenue for the company than the traditional supply chain (Ageron, 2011). And one of the interesting breakthroughs in the field of digital supply chain is incorporating digital twin technology to create a digital supply chain, thus create a concept of supply chain digital twin.

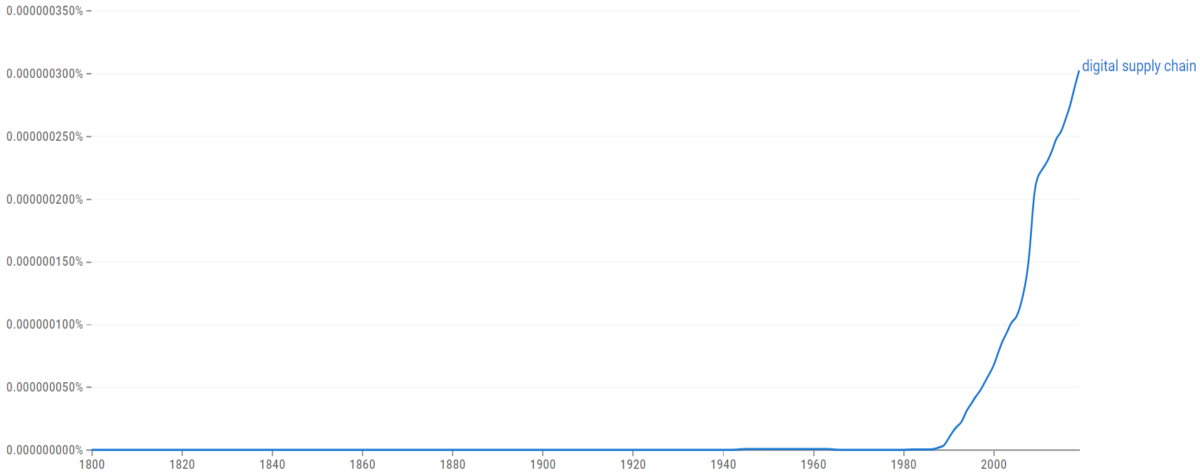
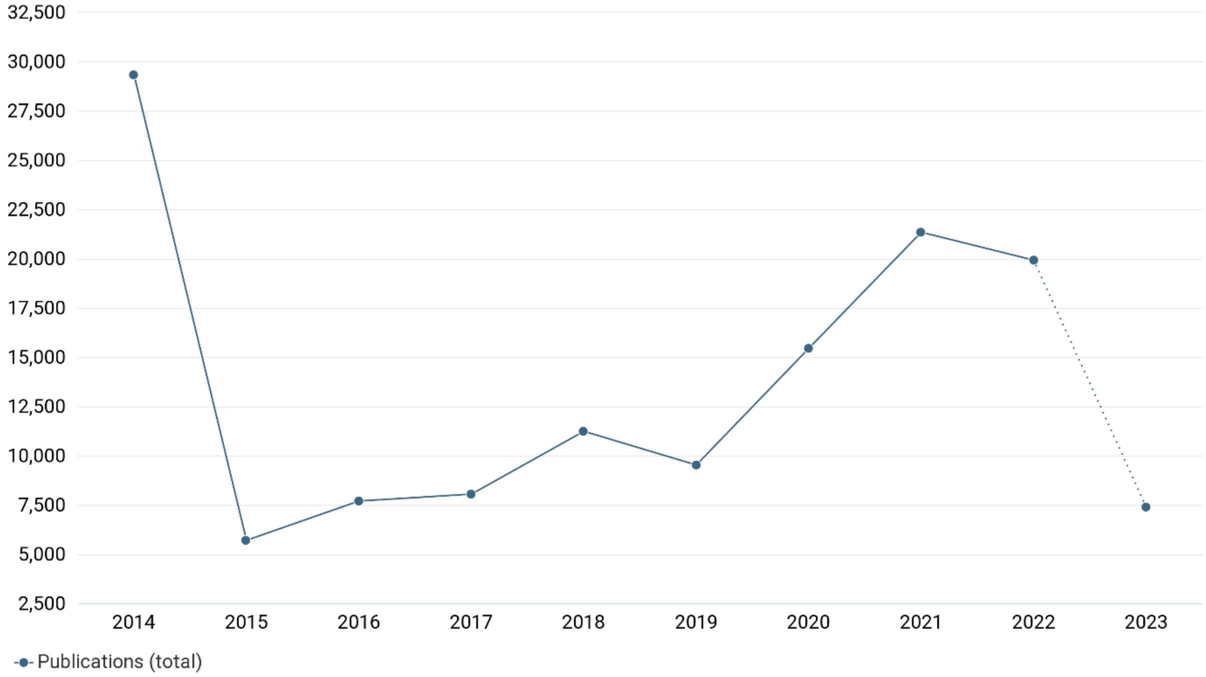


***Figure 9**. Trend of digital supply chain over the year*

Publications in each year. (Criteria: see below)



Source: https://app.dimensions.ai
Exported: July 13, 2023
Criteria: 'supply chain digital twin' in full data.

***Figure 10**. Publications of supply chain digital twin over the year*

According to Google Ngram viewer (**Figure 9**), after the year 1990 there has been an overwhelming in increase in interest in the digital supply chain. The evolution of technologies helped in making the digitalization of supply chain more affordable and able to boost its effectiveness. The concept of supply chain digital twin has emerge in recent years and currently it is not possible to pinpoint the exact time when the concept was born. However, researchers and developers have gradually realized the benefits and power of supply chain digital twin since many corporate like Microsoft Azure, Google or Amazon Web Service have and are developing their own digital twin solution for the supply chain. According to Dimension, which is the most comprehensive research grants database which links grants to millions of resulting publications, clinical trials and patents, from 2014 to 2023 there have been 252,606 publications which is about the supply chain and digital twin (**Figure 10**).

Although there are many publications throughout the year, the definition of a supply chain remain unchanged: it is a detailed simulation model of an actual supply chain which uses real-time data and snapshots to forecast supply chain dynamics and has the following characteristics:

- Be thorough enough to study the interactions of the supply chain, from large-scale changes in demand to what happens inside facilities. It ought to be able to do tasks like demand fluctuation detection, financial, stock keeping unit forecasting, and scenario testing.
- Offer programmable alerts, alarms, or notifications to notify you of unusual events, such as service levels.

- Utilize real-time information feeds to assess the current state of the supply chain and offer up-to-date forecasts, such as incoming shipment schedules, vehicle locations, and inventory levels.
- Allow you to create action plans to assist you in dealing with odd situations, then test those plans to make sure they work.
- Be part of a “bigger thing”: integrate with the surrounding IT environment of databases and business intelligence tools.

2.3.2 Core technologies

Digital twin is a concept, not a specific technology but a combination of many. Therefore, through published research the author’s idea of core technologies differs from each other. Ivanov et al. (2019) define the core technologies for supply chain digital twin are: simulations, optimization, and data analytics. Essakly et al. (2019) added IoT, block chain and cyber security. Unity technologies considered data analytics and business intelligent in implementing a supply chain digital twin. However, most authors agree on simulation, IoT and data analytics are core technologies of a supply chain digital twin (Barykin et al,. (2020)).

2.3.3 Benefits

Supply chain digital twin provide benefits such as:

- **Risk management:** Since digital twin can accurately recreate the production process, they can assist users in thoroughly checking each step of the process as well as assist businesses in evaluating products before they are released to the market. As a result, it is able to spot errors and prepare for unforeseen circumstances.
- **Improved short to mid-term forecast:** One of a supply chain digital twin's most prominent benefits is its improved ability to predict short- and medium-term outcomes. Simulating a few weeks to months of operations with the applied supply chain management tactics can help predict incoming errors that would be too costly or critical to operations in real life.
- **Remote supervision:** Digital twin make it possible for users to remotely monitor and manage the system. This will help managers to oversee the whole supply chain and enable better decision making.
- **Savings on costs:** With digital twin, users can rehearse and recreate the process in a more affordable setting than in real life. Managers can test different strategies and policies that would be too costly and hard to change in the future.

3. PROPOSED SYSTEM

3.1 System overview

To understand how a supply chain digital twin works and its effect on a business, a supply chain digital twin will be developed during this thesis. The result digital twin is a web-based application, connected to its physical instance using IoT devices. The model that the digital twin simulate will be of the supply chain model defined in section 2.1.3. The digital twin will be able to connect to the supply chain participants (e.g.

Factory, Distribution center or Retailer) by IoT devices and receive telemetry sent by devices in real-time. Sent data will be stored in a database in order to analyze its performance.

Current supply chain digital twin solution in the market:

In order to understand more about the digital twin implementation, 2 solutions already released in the market are analyzed:

- **Azure digital twin:** It is a platform provided by Microsoft Azure that enables the creation, deployment, and management of digital representations of physical environments. Azure digital twin has advantages over other digital twin application is that it is well documented, use digital twin definition language (DTDL) to model entities and since it is a part of Azure Microsoft Cloud service, it can be integrated with more than 200 products provided by Microsoft. It only disadvantages were that it is not possible to integrate with products that is not made by Microsoft.
- **Anylogistix:** It is a software combines supply chain optimization and simulation for logistics network design, inventory, and production capacity planning. One of its advantages is of the simulation component. Since it is built on Anylogic, a multimethod simulation modeling tool that have been around for more than 20 years, it is flexible and can simulate any model. The use of state machine and state chart in creating model makes it easy to use. However, one of it's disadvantages is that it needs many third parties program such as an IoT hub and a real-time database in order for it to perform well.

3.2 Requirement analysis

The main goal of the supply chain digital twin is to enable organizations to create a virtual model of their supply chain operations and processes, using real-time data from physical supply chain activities. This virtual model can be used to monitor real-time change and allow stakeholders like suppliers or retailers to take action as fast as possible.

Functional requirements:

- **Data Integration**: The ability to collect and connect data from various sources, including suppliers, manufacturers, warehouses, and logistics providers, to create a unified view of the entire supply chain.
- **Physical to digital link using IoT devices**: The ability to transfer telemetry collected from physical IoT devices to the digital twin to simulate the real-time aspect.
- **Simulation and Optimization**: The ability to simulate and optimize supply chain operations based on different scenarios, such as changes in demand, disruptions in supply, transportation delays, and production issues.
- **Real-Time Monitoring**: The ability to monitor and track real-time data on key performance indicators (KPIs), such as current participants in the supply chains, sales per day, product delivery progress and current truck location.
- **Scalability and Flexibility**: The ability to scale and adapt to changing business needs, such as changes in the number of warehouses, retailers or products involved in the business process.

Non-functional requirements:

- **Performance**: Because in the real world, a supply chain is very complex and comprises of many participants and elements. Given that the system must be capable of processing a large amount of data in real-time, such as more than 20 IoT devices at the same time in order to simulate a model of a supply chain and maintaining real-time connect.
- **Security:** The system must have robust security measures in place to protect sensitive supply chain data from security breaks by using methods such as using MQTT protocol to encrypt messages and secure communication between the broker and the client.
- **Reliability**: The system must be able to send and receive a minimum of 20 messages sent from the devices and update it to the database in no less than a second to ensure real-time monitoring experience.
- **Usability**: The system must be easy to use and navigate, with intuitive interfaces that enable supply chain managers to quickly access and interpret data.
- **Flexibility**: The system must be flexible and adaptable to support different supply chain operations and processes such as adding or removing new retailers or products, as well as changing business needs and market conditions.

3.3 Architecture of the system

To implement the system, the layered architecture is applied. Layered architecture or n-tier architecture is a software architecture pattern and is the most widely known architecture for architects, designers, and developers. The layered architecture divides the system into many layers, all of which perform a distinct function within the application and each layered is independent of each other.

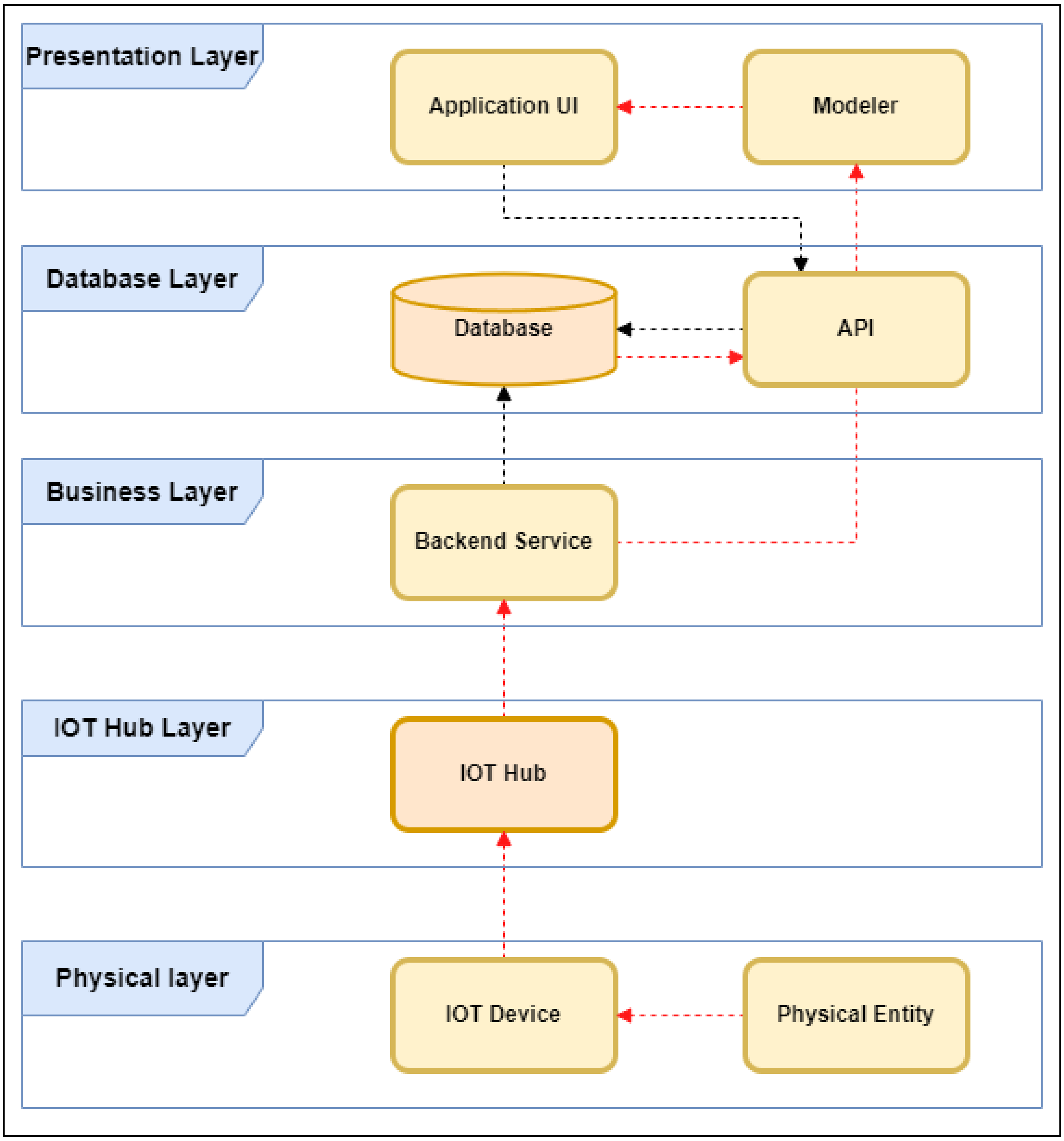


Figure 11. System architecture diagram

- **The presentation layer:** In this layer, the UI is revealed to the user and the modeler is used to represent the supply chain model, all of which help make render the supply chain for the user to interact with. To render the model, the modeler get the data which from the database through the API in database layer.
- **The database layer:** The database layer contain the database which will be used to store all information and an API which allow for the database to communicate with the client. The client can perform operation such as INSERT, UPDATE, DELETE, SELECT through the API.
- **The business layer:** This layer contain the backend service component which contain the API for communication between the database layer and the physical layer. This layer is responsible for receiving the messages sent from IoT hub. When message is received, the API will upload the received data to the database.

- **The IoT Hub layer:** This layer contains the IoT hub, which is used to manage all incoming telemetry sent by IoT devices, then it will take all changes and sent it to the database layer for the API to perform its function.
- **The physical layer:** This layer contains the physical entities and their corresponding IoT devices. The devices have a sensor that can sense physical phenomenon and will send the record data to the IoT hub after a time interval.

3.4 Technology

This section will describe all technologies used in developing the supply chain digital twin.

Node.js

Definition: Node.js is an open-source and cross-platform JavaScript runtime environment. It allows developers to write both front-end and back-end applications using JavaScript. Because the digital twin will be running on the web environment, Node.js is chosen to write both the front-end and back-end of the application.

Benefits:

- **Open source:** since node.js is open source, all of its features can be used without any charge. For
- **Cross platform:** Node.js can run on any operating system such as Mac, Linux, or Windows. And the built application can operate on most modern browsers like Google Chrome, Microsoft Edge, or Opera.
- **Great community support:** Since its first appearance in 2009, Node.js has gradually become one of the most popular options for web development. And since it was open-source, many developers have created many packages that help improve the library. Now the node.js library contains a package for almost every problem, making it a great choice for small teams with limited budget.

Front end

React.js

Definition: React is an open source JavaScript front-end library for building user interface. It can be used to create single-paged, mobile or server-rendered application and is one of the most popular front-end library.

Benefits:

- **Great community support:** because of its popularity and open-source nature, React receive a lot of community support. Many react libraries for almost every problem has been released throughout the year, and there are many tutorials provided by expert on the internet. Further more, encountered problem can be found easily on the internet since on many question-and-answer for programmers website like stack overflow, React.js problems was some of the most have been asked.
- **Components usage:** React use components to render user interface. Each component render a different features like a modal, sidebar, or menu. It’s like using class in other programming language like C++ so it is easy to use and extend.

- **Better code stability:** Because it use a downward data structure to maintain its components and because of that child component can not make changes to the parents component. This help in creating a hierarchy like structure and help prevent mistake related to overriding function.
- **Work directly with DOM:** The Document Object Model (DOM) in HTML is the interface that render the user interface. Because React.js can work directly with the DOM, it can help in modify user interface and help developer creating what they want.

Ant design: In order to improve the basic React.js, another library is used: Ant design. It is a library made for React.js and provide many pre-made components that can help with building a beautiful and elegant user interface. Not only that the components can be extended and customized base for own usage. Using it will help boost development time and user interface quality.

React-redux.js

Definition: Another library that can be used to help make React better is redux, which helps manage the state of the application. State is the data or variable that represents data sent to the user. As applications grow larger, that state can be hard to manage.

Benefits:

- **Predictable:** Since redux state are immutable, which means that the state are unchangeable after creation. This allow the code to be more readable and predictable after each render since the data will stay the same.
- **Optimize performance:** Normally, when a state is changed, react re-render all of its component inside the component tree. Redux make it so that react only render the changed data component when a changed is detected, making the application faster.

Bpmn.js

Definition: In order to show the current process inside a participant (e.g., retailer, supplier), a BPMN render library is used. The chosen library is Bpmn.js which is written in JavaScript, embeds BPMN 2.0 diagrams into modern browsers and requires no server backend

Benefits:

- **Framework-agnostic:** Since Bpmn.js is written in JavaScript, it can be easily integrated with many framework which is written using JavaScript.
- **Easy to use:** The Bpmn.js has many documents and tutorials to guide the user through its set up and usage. Further more the library can read directly from the BPMN XML file and render it on the view, making it easy to work directly with BPMN file.

Mapboxgl.js

Definition: Mapbox gl js is a library for rendering map on front end. The map data is taken from open data sources such as OpenStreetMap or WikiData. It help create custom map and has a navigation API for route finding. In order to digitalize a supply chain, a map is needed in order to show the exact location of its participants.

Benefits:

- **Support up to 25000 user for free:** The mapbox js has a free tier that allow for limited use of its API. Compare to the popular choices like Google Map, which charge users monthly with the Google Developer Account, the mapbox API has better price.
- **Read GeoJSON data:** The mapbox API also comes with a function to read GeoJSON, an open standard format designed for representing geographical features based on JSON. This is useful for rendering route and road on the map since most transport dataset now use GeoJSON data to represent location.
- **Free direction API:** The mapbox API also comes with a free direction API, which can be used to find the shortest route between the 2 locations on the map. It is a useful feature and can speed up the developing time of the process.

Backend

Express.js

Definition: Express is an open source back end framework for developing RESTful API with Node JS. It can be used to manage HTTP request receive from client and perform corresponding function.

Benefits:

- **Simplicity:** Express.js has a minimalistic and unopinionated design, making it easy to learn and use. It provides a straightforward and flexible approach to building web applications, allowing developers to focus on their specific requirements.
- **Middleware:** Express.js offers a middleware architecture that enables developers to add functionalities and handle requests and responses efficiently. Middleware functions can be used to perform tasks like logging, authentication, error handling, and more. This modular approach allows easy extensibility and enhances the overall development process.
- **Routing:** Express.js provides a simple but powerful routing mechanism. Developers can define routes easily and handle HTTP requests using different methods such as GET, POST, DELETE, etc. This makes it convenient to create RESTful APIs and handle various endpoints within an application.

Database

Supabase:

*Definition:*In order to maintain the connection between the digital twin and its physical counterpart, a realtime database is used to reflect changes. Not only that the database must be able to handle high volume of incoming data, it must be able to perform consistenly. For this reason, the realtime database Supabase is chosen. It is a cloud-based database which is built upon PostgreSQL. It provide users with a wide array of functions such as authentication, real-time, and storage. It also provide an endpoint so that the client can connect directly to the database.

Benefits:

- **Real-time function:** Supabase allow developers to access to database to store and sync data across multiple devices in real-time. This is useful in building a digital twin since it is crucial for the digital twin to be always up to date with the physical counterpart.
- **Built in endpoint:** The Supabase provided a built in endpoint that can allow client to perform create, read, delete and update (CRUD) directly on the database without the need for an API.
- **PostgreSQL:** Supabase is built on PostgreSQL, a powerful relational database management system. Because of that, it can perform any function that PostgreSQL can do such as write query using SQL syntax or create function.

Azure IoT Hub

Azure cloud service: Microsoft Azure is a complete development and deployment environment in the cloud. It provides a wide range of cloud-based services and solutions that help businesses and organizations build, deploy, and manage various applications and services. Azure offers a vast array of services covering computer, storage, networking, databases, artificial intelligence, machine learning, analytics, and more. During the development process for the supply chain digital twin, IoT Hub service from the Azure cloud service is used as the digital twin IoT service.

Definition: The Internet of Things (IoT) is a network of physical devices that connect to and exchange data with other devices and services over the Internet or other network. Anything that can be embedded with the necessary sensors and software can be connected over the internet. Azure IoT Hub is a managed service hosted in the cloud that acts as a central message hub for communication between an IoT application and its attached devices. Azure IoT Hub can receive messages sent from the physical devices and record changes on the Event Hub. It also provides an endpoint to connect with client and other API.

Benefits:

- **Secure Device Connectivity**: Azure IoT Hub provides secure and reliable connectivity between IoT devices and the cloud. It supports various protocols and encryption mechanisms, ensuring that data is transmitted securely over the internet.
- **Scalability:** Azure IoT Hub is built to handle large-scale deployments of IoT devices. It can handle millions of devices simultaneously, allowing organizations to easily scale their IoT solutions as the number of connected devices grows.
- **Device Management Capabilities**: IoT Hub offers robust device management features, including device provisioning, configuration, and monitoring. It allows organizations to remotely manage and update devices, track their health status, and perform firmware updates in a centralized and efficient manner.
- **Integration with Azure Services:** IoT Hub integrates smoothly with other Azure services, such as Azure Functions, Azure Logic Apps, Azure Time Series Insights, and Azure Machine Learning. This allows organizations to leverage advanced analytics, AI, and visualization capabilities offered by these services to gain valuable insights from IoT data.

- **Global Reach and Availability:** Azure IoT Hub is available in multiple regions globally, allowing organizations to deploy IoT solutions closer to their target markets, reducing latency and ensuring high availability.

Other tool:

GitHub

Definition: GitHub is a website and cloud-based service that helps developers store and manage their code, as well as track and control changes to their code.

Benefits:

- **Testing:** GitHub help creating a separate environment to test new features by branching using Git version control. Then if the changes are good git can help merge the testing branch with the main branch.
- **Code control:** GitHub can help managing code easier for developers. Changes to the code can be tracked by Git and developers can see the history of the code at any time.

Draw.io

Definition: Draw.io is a free and open-source cross-platform graph drawing software developed in HTML5 and JavaScript. It’s help in designing ERD diagram and other visualization in this thesis.

Benefits:

- **User friendly:** The application has a simple interface which helps make it easy for people to use. It also has a drag and drop functionalities which also help in making diagram more quickly and easier.
- **Versatile:** The application has many shapes and symbols thus making it capable of visualizing any diagrams. It also has many templates of many diagrams to help in visualizing a diagram.

Netlify

Definition: Netlify is a cloud-based platform that offers web hosting and serverless backend services for static websites and web applications. It simplifies the process of deploying, managing, and scaling websites by providing build automation, continuous deployment, CDN (Content Delivery Network), and DNS (Domain Name System) management. Netlify is used to deploy the digital twin to the web so that anyone can use.

Benefits:

- **Easy Deployment:** Netlify makes the deployment process simple and streamlined. It supports various methods of deploying your projects, including Git-based deployments, continuous deployment pipelines, and drag-and-drop uploads.
- **Scalability:** Netlify leverages a global CDN (Content Delivery Network) to distribute your website's content across multiple servers worldwide. This helps improve performance and ensures your site can handle high traffic loads without slowing down.

- **Serverless** Backend: Netlify offers serverless functions that allow you to run server-side code without managing infrastructure. This enables you to build dynamic and interactive web experiences without the need for a traditional backend server.
- **Git Integration:** Netlify seamlessly integrates with Git repositories, allowing you to trigger builds and deployments automatically whenever you push changes to your repository. This makes it easy to maintain version control and collaborate with team members.

3.5 How the system works

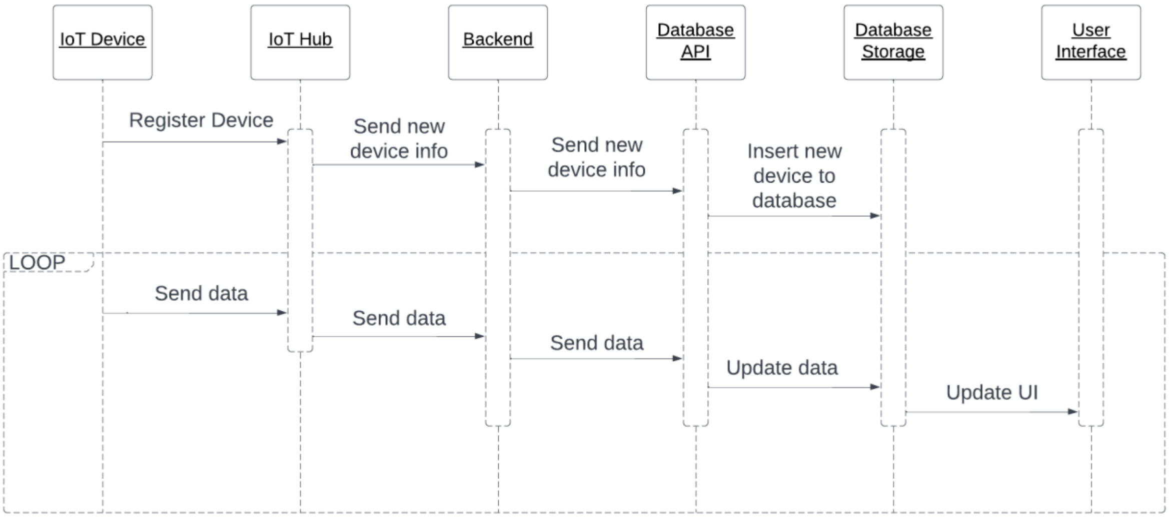


Figure 12*. Sequence diagram of the system*

How telemetry are sent from IoT device to IoT Hub

MQTT protocol:

Figure 13 demonstrate how the system get data from the IoT device and update it on the user interface. In order to sent data log from device to IoT Hub, Azure use a messaging protocol known as Message Queueing Telemetry Transport (MQTT). It is an extremely simple and lightweight messaging protocol (subscribe and publish) designed for limited devices and networks with high latency, low bandwidth or unreliable networks. Its design principles are designed to reduce the network bandwidth and resource requirements of devices and ensure security of supply.

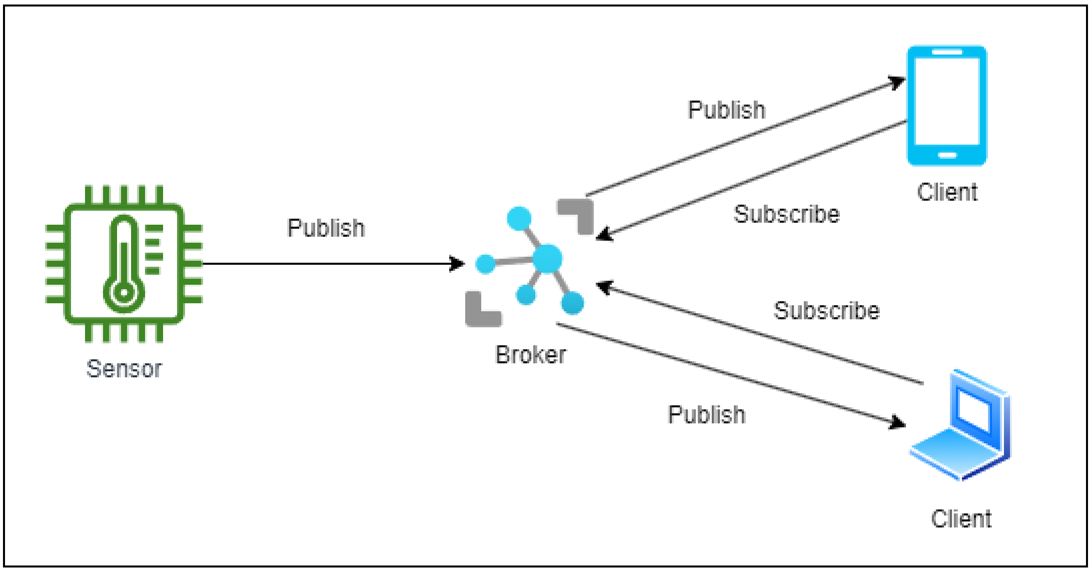


Figure 13. IoT data transport

With MQTT, an IoT device can send a or publish information of a specific topic to a server that act as a MQTT message broker. The broker act as a central hub for receiving, filtering messages as well as managing which client to sent the message to. A client can subscribe to a topic to get all the changes publish by the broker. When a topic is sent to the broker, the broker will get the list of client that have subscribed to that topic and publish the information to the client. In this thesis, the sensors are IoT devices attach at entities inside of a supply chain participants (e.g., cashier, truck), the broker is the third party service Azure IoT Hub and the client are the web application for the supply chain digital twin.

In order to connect IoT devices to Azure IoT Hub, the device must be registered on the hub. After registering, the hub will act as a broker and listen to all messages publish by the registered device. In order to subscribe to the hub and get receive the messages, a backend API is implemented. It use the Azure MQTT IoT Device library to create a subscription the IoT Hub. Then when a subscribed topic is published, the API can get the messages publish by the hub.

How the UI are rendered in real time

To render the user interface in real time, a real time database is used, and in this project the Supabase database is used. Supabase use websocket to broadcast changes made to the database.

WebSocket is a full-duplex bidirectional protocol which can establish a connection between the client and the server and can keep it alive until terminated. The connection is called a handshake and can help the server and client communicate with each other bidirectionally. After the client initiates a handshake protocol with Supabase, the client can determine how data is updated. In implementing the digital twin, the server communicates via PostgreSQL changes, which will send messages to all subscribed client when there are changes in the database. The client will then render the UI based on the received information.

3.6 Database design

This section describes the database schema which is used for storing data. In **Figure 14**, there are 14 tables across the database to save and use incoming data from IoT Hub.

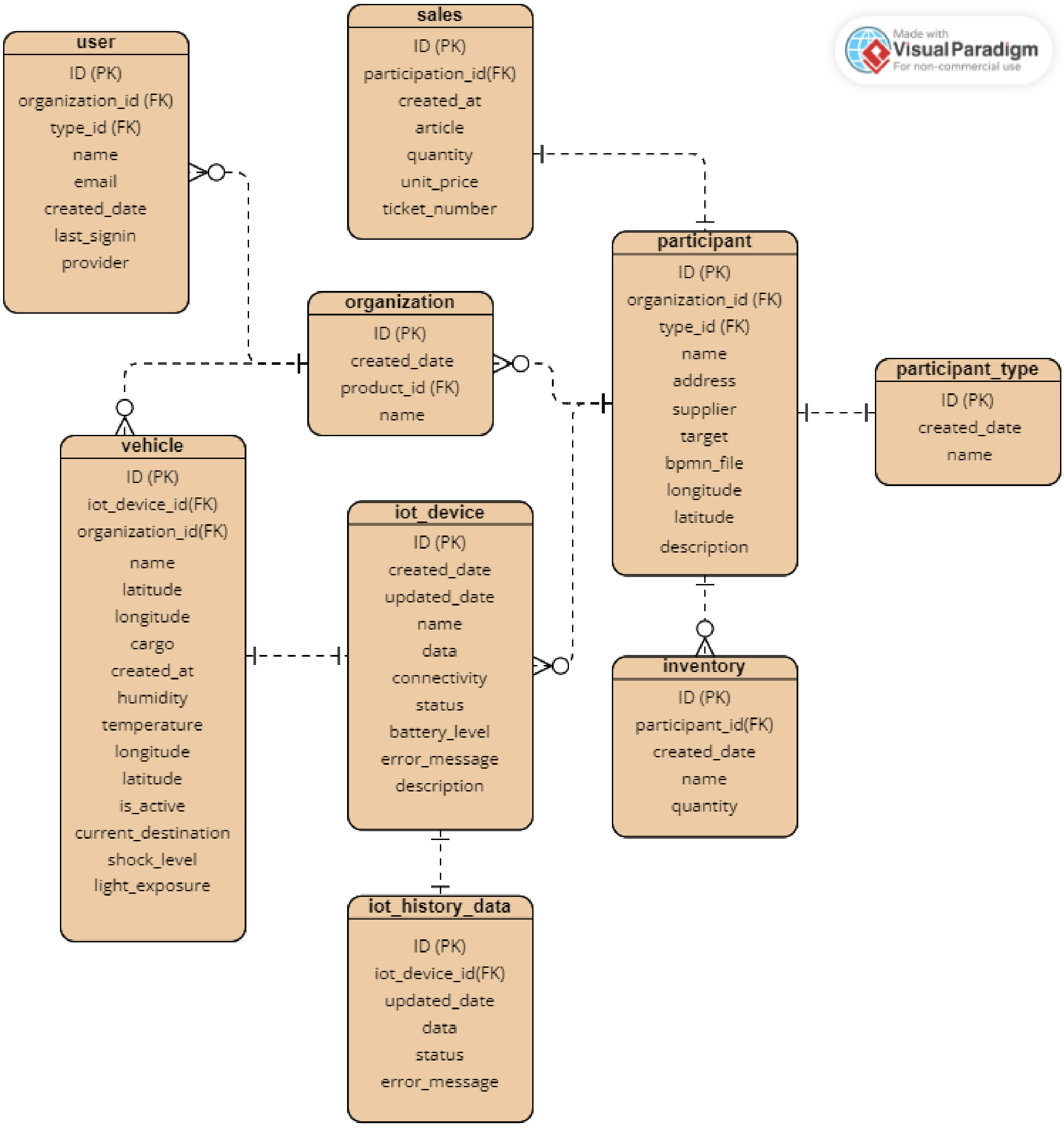


***Figure 14**. ERD diagram of database*

In this database:

- “user” represents the real-world user of the system, each user comes with a unique identifier and is created via the register function.

- “organization” represent the organization or the company that the user belongs to, each organization can have many employees and each employee in the user can inspect the data inside of the organization which they belong to.
- Each “organization” can have many “participant”, which are considered supplier or retailer in the supply chain.
- Each “participant” will have an “inventory” which show how many products are stored in the participant warehouse and inventory.
- Each “participant” will have their own sales data which will be record in “sales” table.
- Each “participant” will have its own IoT devices which will be used to store data sent from devices in that location. All historical IoT data will be stored in the table “iot_history_data”.
- When a row in “iot_device” changes, that change will trigger the function “on_update”, which will then insert the changes to “iot_historical_data”. The data can be used for various purpose such as analysis.
- Each “organization” will have one or many “vehicle” which carry cargo for the company. Each “vehicle” will have its own “iot_device” to track the cargo or the inside of a vehicle.

Table organization

Name	**Type**	**Null**	**Default Value**	**Description**
Id	uuid	no	Gen_random_uuid()	Primary key: unique id of the organization
Created_at	timestamptz	no	Now()	Created date of the record
Name	text	no	None	Name of the record

Table user

Name	**Type**	**Null**	**Default Value**	**Description**
Id	uuid	no	Gen_random_uuid()	Primary key: unique id of user
Created_date	timestamptz	no	None	Created date of the record

Last_sign_in	timestamp	no	None	The last sign in time of the user
email	text	no	None	Registered email of the user
phone	text	no	None	Registered phone number of the user
provider	text	no	None	Provide in case user use third party login such as Gmail or Facebook
Organization _id	uuid	no	None	Foreign key of table "organization"

Table participant

Name	**Type**	**Null**	**Default Value**	**Description**
Id	uuid	no	Gen_random_uuid()	Primary key: unique id of user
name	Text	no	None	Name of the participant
Address	Text	no	None	Address of the pariticipant
Longitude	Float8	no	None	Longitude of the participant
latitude	Float8	no	None	Latitude of the participant
type	text	no	None	Foreign key of table participant type

supplier	text	no	None	The name of the supplier for this participant, which is the place from which the participant receive its product
target	text	no	None	The name of the delivery target for this participant. The participant will deliver to this target
Organizaion_ id	uuid	no	None	Foreign key of table “organization”, show which organization the participant belong to
bpmn	text	no	None	Contain the bpmn schema for the participant which will show what business is in progress
description	text	no	None	Participant description

Table participant_type

Name	**Type**	**Null**	**Default Value**	**Description**
Id	Text	no	None	Primary key: unique id of participant_type

type	Text	no	None	Type of the participant for rendering, can be: dtmi:dtdl:Supplier, dtmi:dtdl:Distributor, dtmi:dtdl:Customer and dtmi:dtdl:Farm

Table inventory

Name	**Type**	**Null**	**Default Value**	**Description**
Id	uuid	no	Gen_random_uuid()	Primary key: unique id of inventory
Created_at	timestamptz	no	Now()	Created date of the record
Last_updated	timestamp	no	Now()	Last updated date of the record
Participant_id	text	no	None	Foreign key of table "participant"
name	text	no	None	Product name
quantity	Int4	no	None	Quantity of product in the storage

Table iot_device

Name	**Type**	**Null**	**Default Value**	**Description**
Id	varchar	no	None	Primary key: unique id of user

Created_date	timestamptz	no	Now()	Created date of the record
Updated_date	timestamp	no	Now()	Last updated date of the record
name	Text	no	None	Name of the IoT device
data	jsonb	no	None	Used to store telemetry sent by IoT hub in the form of JSON
connectivity	Boolean	no	False	Check if the IoT device is working or not by checking the connectivity with the server
status	text	no	“OK”	Status can be “OK” or “ERROR” which is used to check if data sent is good or not.
Battery_level	float	no	None	Battery level of the device
Error_message	text	no	None	If an error is occurred, a message is sent back to the server.
Description	text	no	None	Text description of the device

Table iot_history_data

Name	**Type**	**Null**	**Default Value**	**Description**
Id	uuid	no	Gen_random_uuid()	Primary key: unique id of user
Created_date	timestamptz	no	Now()	Created date of the record
Updated_date	timestamptz	no		updated date of the record
Iot_device_id	varchar	no	None	Foreign key of table “iot_device”
data	text	no	None	Data sent by IoT device in the form of JSON
status	text	no	“OK”	Status can be “OK” or “ERROR” which is used to check if data sent is good or not.
Error_message	text	no	None	If an error is occurred, a message is sent back to the server.

Table vehicle

Name	**Type**	**Null**	**Default Value**	**Description**

Id	uuid	no	Gen_random_uuid()	Primary key: unique id of user
Created_date	timestamptz	no	None	Created date of the record
name	Text	no	None	Name of the vehicle
velocity	Float4	no	None	Current velocity of the vehicle
weight	Float4	no	None	Vehicle weight
Cargo_id	uuid	no	None	Foreign key of table “cargo_data”
Max_cargo	Float4	no	None	Maximum cargo the vehicle can transport
type	text	no	None	Type of the vehicle
Is_active	bool	no	false	Check if the vehicle is active or not
longitude	Float8	no	None	Current longitude of the vehicle
latitude	Float8	no	None	Current latitude of the vehicle
Current_destination	text	no	None	Foreign key of table “participant”

Iot_device_id	Varchar	no	None	Foreign key of table “iot_device”
humidity	Float4	no	None	Humidity inside of cargo
temperature	Float4	no	None	Temperature inside of cargo
Cargo_weight	Float4	no	None	Cargo weight

Triggers: A trigger is a function invoked automatically when an event occurs. A few triggers are implemented in order to help manage the database easier.

- Trigger: on_telemetry_updated

```
CREATE OR REPLACE FUNCTION insert_iot_history()
  RETURNS TRIGGER
  LANGUAGE PLPGSQL
  AS
$$
BEGIN
  insert into public.iot_history_data(iot_device_id, data) values
(NEW.id,NEW.data);
  RETURN NEW;
END;
$$

-- trigger the function every time a telemetry is added
create trigger on_telemetry_updated
  after update on public.iot_devices
  for each row execute procedure public.insert_iot_history();
```

The trigger on_telemetry_updated is triggered when a UPDATE action has been performed on the table public.iot_devices. Which will trigger the function insert_iot_history() which will insert the new data into the table iot_history_data.

- Trigger: on_auth_user_created

```
-- inserts a row into public.profiles
create or replace function public.handle_new_user()
returns trigger
language plpgsql
security definer set search_path = public
as $$
begin
  insert into public.users (id, email)
  values (new.id, new.email);
  return new;
end;
$$;

-- trigger the function every time a user is created
create trigger on_auth_user_created
  after insert on auth.users
  for each row execute procedure public.handle_new_user();
```

The trigger on_auth_user_created will insert a new user data into the table public.users when a new user is created. This method is used to because the login and register function use Supabase authentication function, which will handle the check for user credentials. This is a very useful function since it help save time and effort to implement a checker function for user input. The user data is stored in a table called auth.users. And this table is limited by Supabase to read only and the system being implemented must be able to change user information if needed. Therefore it is better to create a duplicate table of the auth.users which is the table user and manager users on it.

4. IMPLEMENTATION

4.1 Technology version

Source code: The source code can be found on the link: https://github.com/FasterThanLigh1/Supply-Chain-Digital-Twin-React .

Technology version: The technology version will list some of the important dependencies.

Technology	**Version**
Node.js	v17.9.1
Redux.js	v1.9.5
React	v18.2.0
Recharts.js	v2.6.2
Azure-iothub	v1.16.3

Bpmn-js	v13.0.4
Mapboxgl.js	v2.14.1
Antd	v5.4.6
Azure-iot-device-mqtt	v1.16.2
Express.js	v4.16.1

4.2 IoT design

This section will discussed the implementation of IoT device in a supply chain. In order to establish real time connection between a supply chain and its digital twin, IoT devices must be installed in order to send messages back and forth. The supply chain used for the implementation will be the model use case defined in section 2.1.3, the supply chain of a bakery (**Figure 15**).

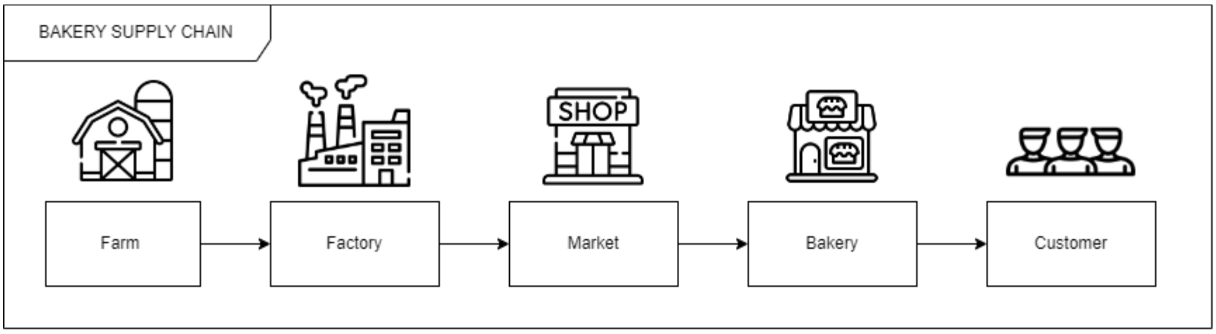


***Figure 15**. Bakery supply chain participants*

In order to create a real time connection between the real-world entities and the digital twin, IoT device must be implemented at 4 participants: the farm, the factory, the distribution center, the retailer and finally inside of all the truck that will be used to transport cargo. Each IoT device will work as in the flowchart (**Figure 16**), which shows that the data will be sent continuously to the cloud if not terminated. Inside of a participant there are many processes and in order to create a detailed enough digital twin, each process must be installed with its own IoT device.

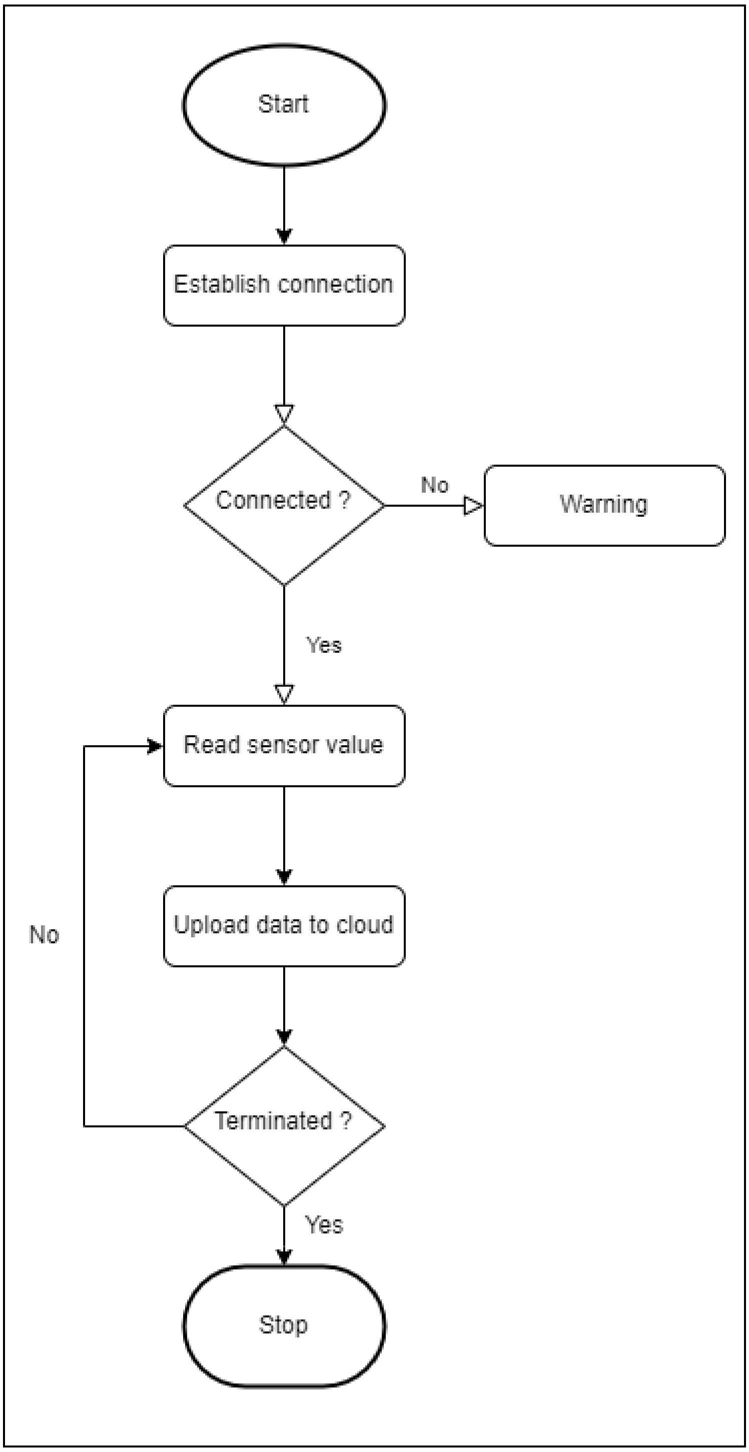


Figure 16. *IoT device flow chart*

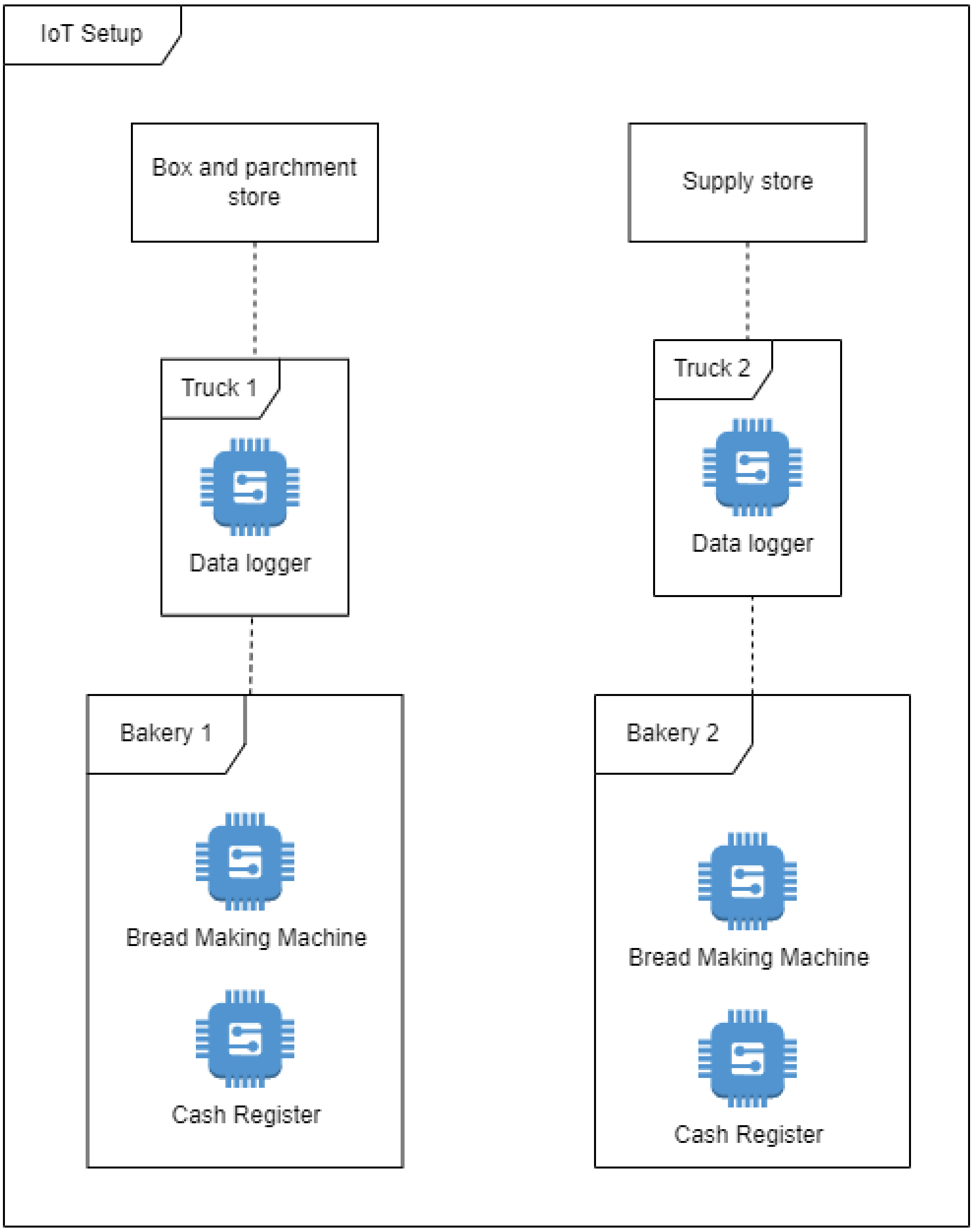


Figure 17. *IoT Setup*

According to **Figure 17**, the IoT devices are installed at 2 locations Bakery 1 and Bakery 2, each has 2 devices installed in the commercial bread making machine to track the progress of making bread and cash register to collect sales data. Each truck will have its own truck and on each truck will be installed a data logger to track transportation data. The telemetry to be sent back to server will be formatted as follow:

Bread making machine telemetry:

```
id: 'machine_1',

data: {
    id: 0,
    date_time: '2021-01-02 04:00:00',
    state: 'Mixing',
    oven_t: 350,
    proofing_t: 85,
    ambient_t: 26.5,
    motor_speed: 1200,
    power_consumption: 500,
    dough_weight: 50,
    dough_volume: 2.5,
    proof_time: 5,
    baking_time: 45,
    baking_temperature: 200,
    quantity_produced: 0
}
```

Cash register telemetry:

```
id: 'sales_record',

data: {
    id: 20,
    date_time: '2021-01-02 09:48:00',
    ticket_number: 150052,
    article: 'COUPE',
    quantity: 1,
    unit_price: 0.15
}
```

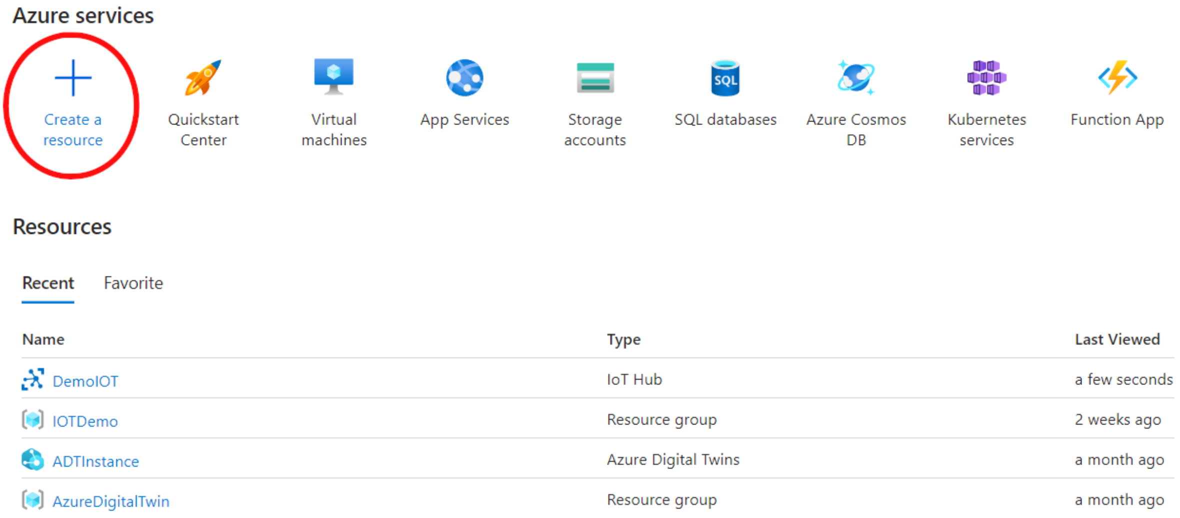
Truck telemetry:

```
id: 'truck_1',

data: {
    id: 20,
    timestamp: '2023-08-31 07:35:00',
    cargo_id: 101,
    latitude: 2.333367,
    longitude: 48.853766,
    temperature: 18.96,
    humidity: 60.49,
    shock_level: 0.13,
    light_exposure: 149.45,
    door_status: 1,
    battery_level: 78,
    connectivity: 1,
    destination: 'dtmi:dtdl:Bakery1;1'
}
```

4.3 How devices are connected to Azure IoT Hub

Azure IoT Hub is one of the most popular IoT Hub out there. One reason contributed to its popularity is because of its user friendly design. Anyone who have an Azur account can easily established connection with an IoT device by following these steps:

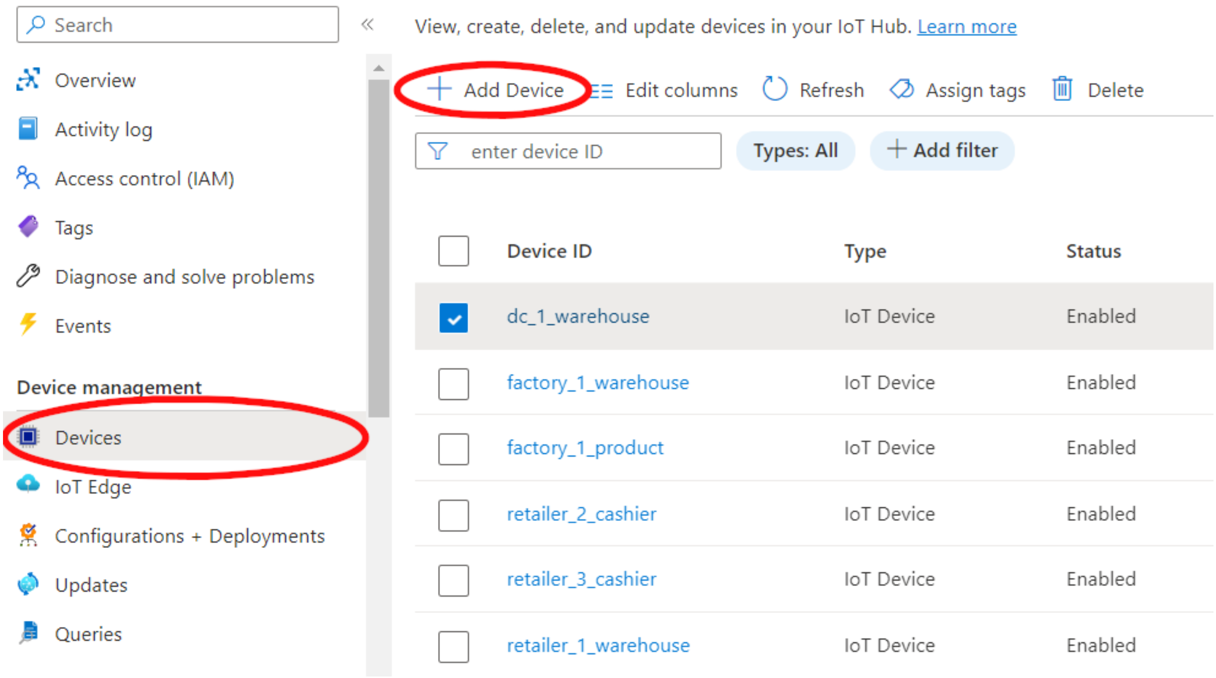
- First, create an IoT hub instance in the Azur portal



Name	Type	Last Viewed
DemoIOT	IoT Hub	a few seconds
IOTDemo	Resource group	2 weeks ago
ADTInstance	Azure Digital Twins	a month ago
AzureDigitalTwin	Resource group	a month ago

***Figure 18**. Azure IoT Hub main page*

- Second, after finishing creating the IoT hub, the devices option can be selected. Then a new device can be added via the option to add new device.

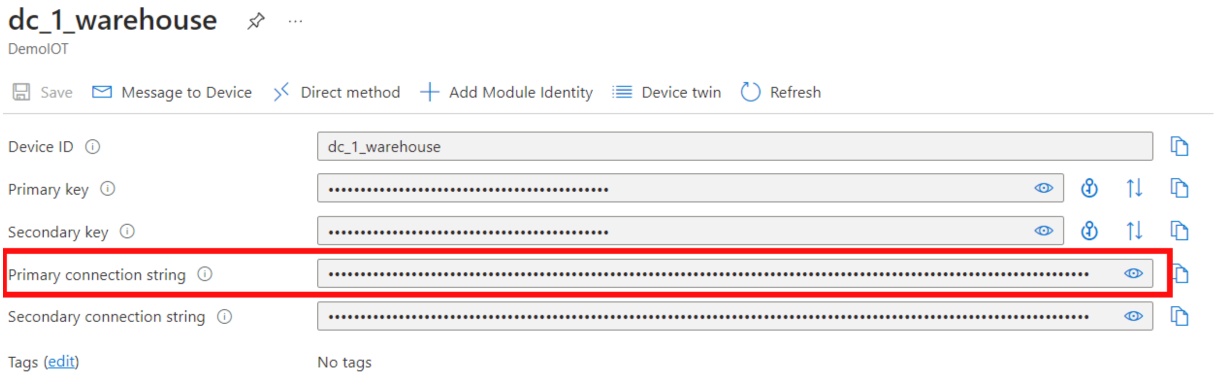


View, create, delete, and update devices in your IoT Hub. Learn more

□	Device ID	Type	Status
☑	dc_1_warehouse	IoT Device	Enabled
□	factory_1_warehouse	IoT Device	Enabled
□	factory_1_product	IoT Device	Enabled
□	retailer_2_cashier	IoT Device	Enabled
□	retailer_3_cashier	IoT Device	Enabled
□	retailer_1_warehouse	IoT Device	Enabled

Figure 19*. IoT page*

- After the device have been created, it can be connected to a device via a connection string. This connections string is act as a bridge between the physical device and the device created on Azur portal. The physical device can get the connection string and help it identify which address on the cloud should the data be sent to.



dc_1_warehouse

DemoIOT

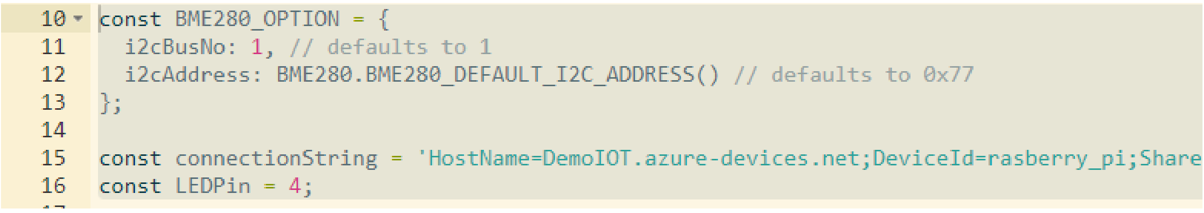
Device ID	dc_1_warehouse
Primary key	••
Secondary key	••
Primary connection string	••
Secondary connection string	••
Tags (edit)	No tags

Figure 20*. IoT device page*

4.4 Backend

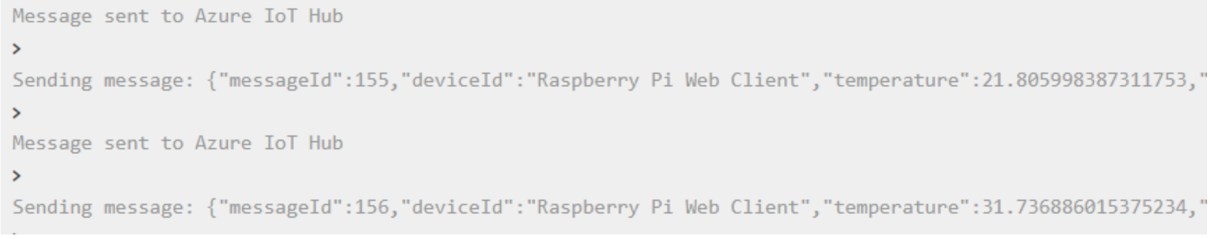
How IoT hub communicate with the backend API

Once connection with the device have been connected, communication can be made between the device and the IoT hub. In order to test this connection, a quick test can be made through the website https://azure-samples.github.io/raspberry-pi-web-simulator/ . This website create a simulator of a IoT device specifically the Raspberry Pi BME 280 in order to test the connection between the hub and the device.



```
const BME280_OPTION = {
  i2cBusNo: 1, // defaults to 1
  i2cAddress: BME280.BME280_DEFAULT_I2C_ADDRESS() // defaults to 0x77
};

const connectionString = 'HostName=DemoIOT.azure-devices.net;DeviceId=rasberry_pi;Share
const LEDPin = 4;
```

After setting up the connection string by using the connection string given when creating a virtual device in the Azure IoT Hub, the device can then be run to send MQTT messages to the Azure server. The Azure broker port is on TCP port 8883.



```
Message sent to Azure IoT Hub
>
Sending message: {"messageId":155,"deviceId":"Raspberry Pi Web Client","temperature":21.805998387311753,"
>
Message sent to Azure IoT Hub
>
Sending message: {"messageId":156,"deviceId":"Raspberry Pi Web Client","temperature":31.736886015375234,"
```

In order to subscribe to the Azure IoT hub to receive the messages broadcast by the server, a back end API is implemented. The API is developed using Node.js and express.js, and subscribe to the Azure IoT hub using the “azure-iot-device” node library.

```
const { Client } = require("azure-iot-device");
const Mqtt = require("azure-iot-device-mqtt").Mqtt;
const { Message } = require("azure-iot-device");
const { EventHubConsumerClient } = require("@azure/event-hubs");
const { supabase } = require("./config/supabaseClient.js");

const eventHubConnectionString = `Endpoint=sb://ihsuprodsgres025dednamespace.servicebus.windows.net/;SharedAccessKeyName=iothubowner;SharedAccessKey=MVMEf3cd4a8kQFzWh6MmT+DKsq1F78ZDXbpDBvec2Ss=;EntityPath=iothub-ehub-demoiot-25131092-2791705379`;

const client = Client.fromConnectionString(connectionString, Mqtt);
```

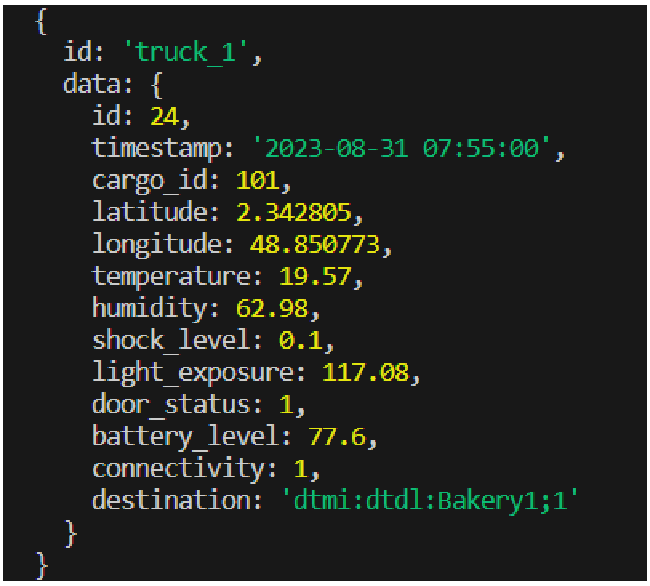
The eventHubConnectionString is the connection string of the IoT Hub service that the user is trying to subsribed to. It is in the form of “Endpoint=…;ShardAccessKey=…;EntityPath=…”. A client can then be created using the connection string, which will open the connection between the client and the IoT Hub.

```
// Open a connection to the IoT Hub
client.open((err) => {
  if (err) {
    console.error("Error opening IoT Hub connection:",
err);
  } else {
    console.log("IoT Hub connection opened");
    // Listen for incoming messages from the device
    client.on("message", (msg) => {
      console.log("Received message from device:",
msg.getData().toString());
    });
  }
});
```

One final step in order to subscribe to changes is to subscribe to Azure event hub. Azure Event Hubs is a modern big data streaming platform and event ingestion service that can seamlessly integrate with other Azure and Microsoft services, and in this case the IoT Hub service. Data from the IoT devices are not transferred directly to the IoT Hub, however, it is sent to the Azure event hub and this is where the message can be broadcast to subscribed devices.

```
// Create the client to connect to the default consumer
group of the Event Hub
  const consumerClient = new EventHubConsumerClient(
    "$Default",
    eventHubConnectionString,
    clientOptions
  );

  // Subscribe to messages from all partitions as below
  // To subscribe to messages from a single partition, use
the overload of the same method.
  consumerClient.subscribe({
    processEvents: printMessages,
    processError: printError,
  });
```

After setting up the API, it can be run in order to listen to changes made by an IoT device. The data received will be in the form of json object and can be restructured for various purpose (**Figure 21**).



```
{
  id: 'truck_1',
  data: {
    id: 24,
    timestamp: '2023-08-31 07:55:00',
    cargo_id: 101,
    latitude: 2.342805,
    longitude: 48.850773,
    temperature: 19.57,
    humidity: 62.98,
    shock_level: 0.1,
    light_exposure: 117.08,
    door_status: 1,
    battery_level: 77.6,
    connectivity: 1,
    destination: 'dtmi:dtdl:Bakery1;1'
  }
}
```

Figure 21. *Received IoT JSON data*

How the API connect to the database

Besides listening to incoming messages from the IoT hub, the API will also perform Create, Read, Update, Delete (CRUD) operation on the database using the Supabase API.

```
const { createClient } = require("@supabase/supabase-js");

const REACT_APP_SUPABASE_URL =
"https://yfglzchwttkotbtjhmnl.supabase.co";
const REACT_APP_ANON_KEY =

"eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpc3MiOiJzdXBhYmFzZ
SIsInJlZiI6InlmZ2x6Y2h3dHRrb3RidGpobW5sIiwicm9sZSI6ImFub24i
LCJpYXQiOjE2ODc0MTQxMjcsImV4cCI6MjAwMjk5MDEyN30.K0EXVA8ele2
aDAH5U5dUI8nw2UCkMLooAg7I4L4LlSg";

const supabaseUrl = REACT_APP_SUPABASE_URL;
const supabaseKey = REACT_APP_ANON_KEY;

const supabase = createClient(supabaseUrl, supabaseKey);

module.exports = {
  supabase,
};
```

REACT_APP_SUPABASE_URL and REACT_APP_ANON_KEY are the user's Supabase address and Supabase database identifier, it let the client know which database should the data be sent to. As a result, a connection with the database is made and data can sent to the database by using query.

4.5 Front-end

How data is updated on the user interface

In order for data to be updated in real time, a supabase client must also be created in the front end application.

```
const { createClient } = require("@supabase/supabase-js");

const REACT_APP_SUPABASE_URL =
"https://yfglzchwttkotbtjhmnl.supabase.co";
const REACT_APP_ANON_KEY =

"eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJpc3MiOiJzdXBhYmFzZ
SIsInJlZiI6InlmZ2x6Y2h3dHRrb3RidGpobW5sIiwicm9sZSI6ImFub24i
LCJpYXQiOjE2ODc0MTQxMjcsImV4cCI6MjAwMjk5MDEyN30.K0EXVA8ele2
aDAH5U5dUI8nw2UCkMLooAg7I4L4LlSg";

const supabaseUrl = REACT_APP_SUPABASE_URL;
const supabaseKey = REACT_APP_ANON_KEY;

const supabase = createClient(supabaseUrl, supabaseKey);

module.exports = {
  supabase,
};
```

However, another step must be taken in order to listen to changes to database is to subscribe to database changes.

```
supabase
.channel("any")
.on("postgres_changes", {
    event: "*",
    schema: "*"
}, (payload) => {
    console.log("Change received!", payload);
})
.subscribe();
```

Postgres changes is one of supabase broadcasting rules. The rules determine when should a message should be sent to all subscribed client and in case of “postgres changes”, a payload is sent to the client when there are changes in the database (e.g., CREATE, UPDATE, READ, and DELETE). The event ‘*” means that it will listen on all event (e.g., CREATE, UPDATE, READ, and DELETE) ad the schema “*” means that this is applied in all tables in the database. After this, when running the client can listen to all changes made by the database and update the UI correspondingly, thus create a real-time feel to the application.

User interface

- **Login Page:** This page handle the login and register function of the digital twin page, which help in protecting private supply chain information (**Figure 22**).

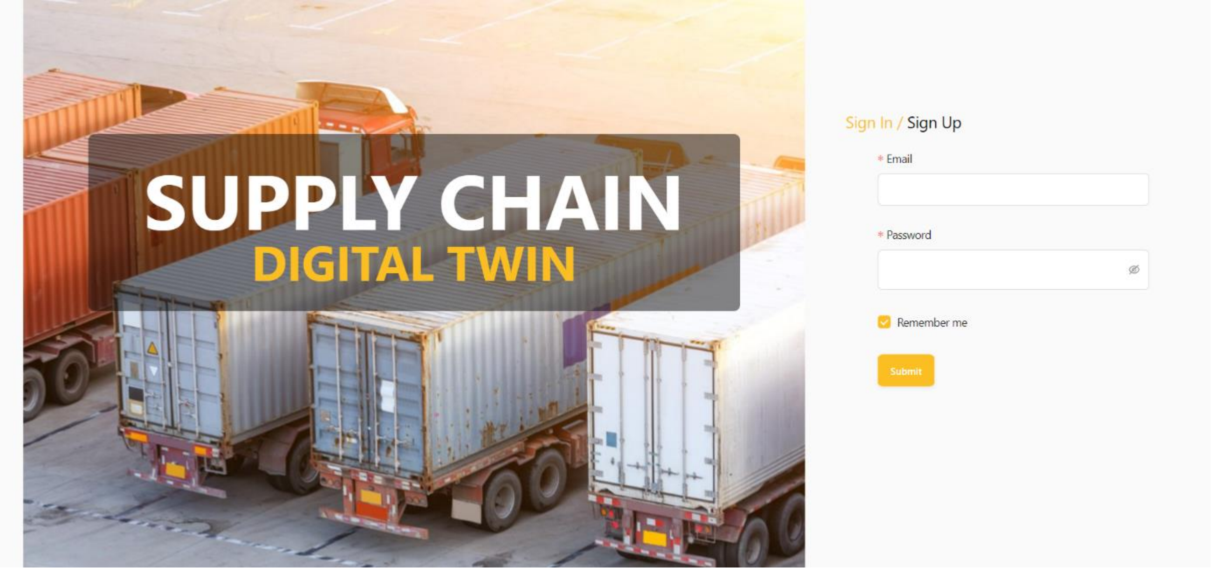
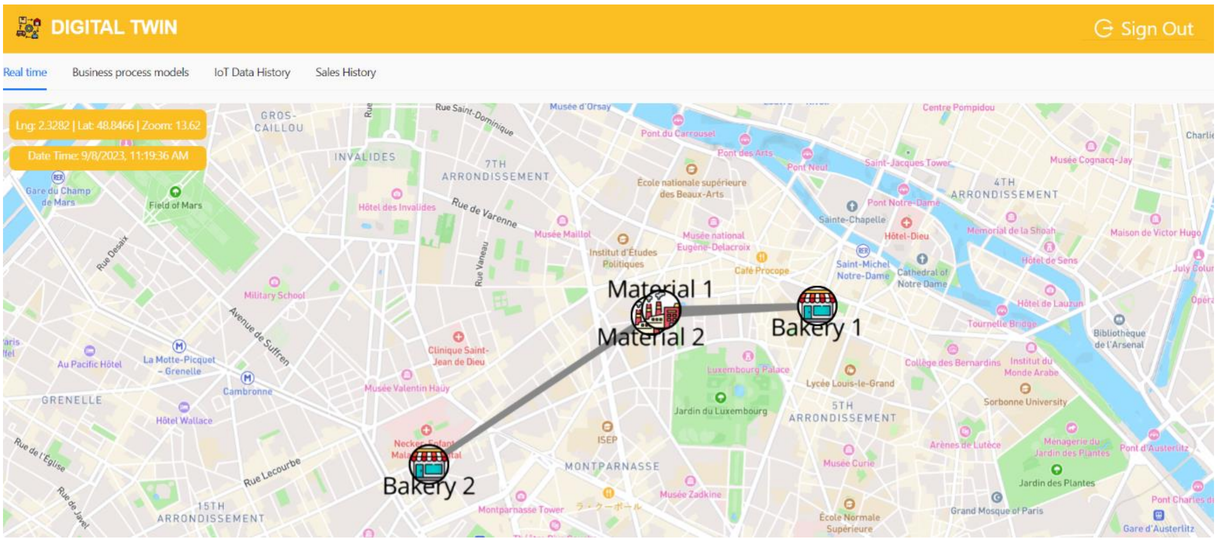


Figure 22. Login Page

- **Main Page:** This page display the whole model of a supply chain from start to finish. As in *Figure 23* The whole supply chain of the bakery is presented, which show the bakery 1 and 2 as well as the 2 suppliers.

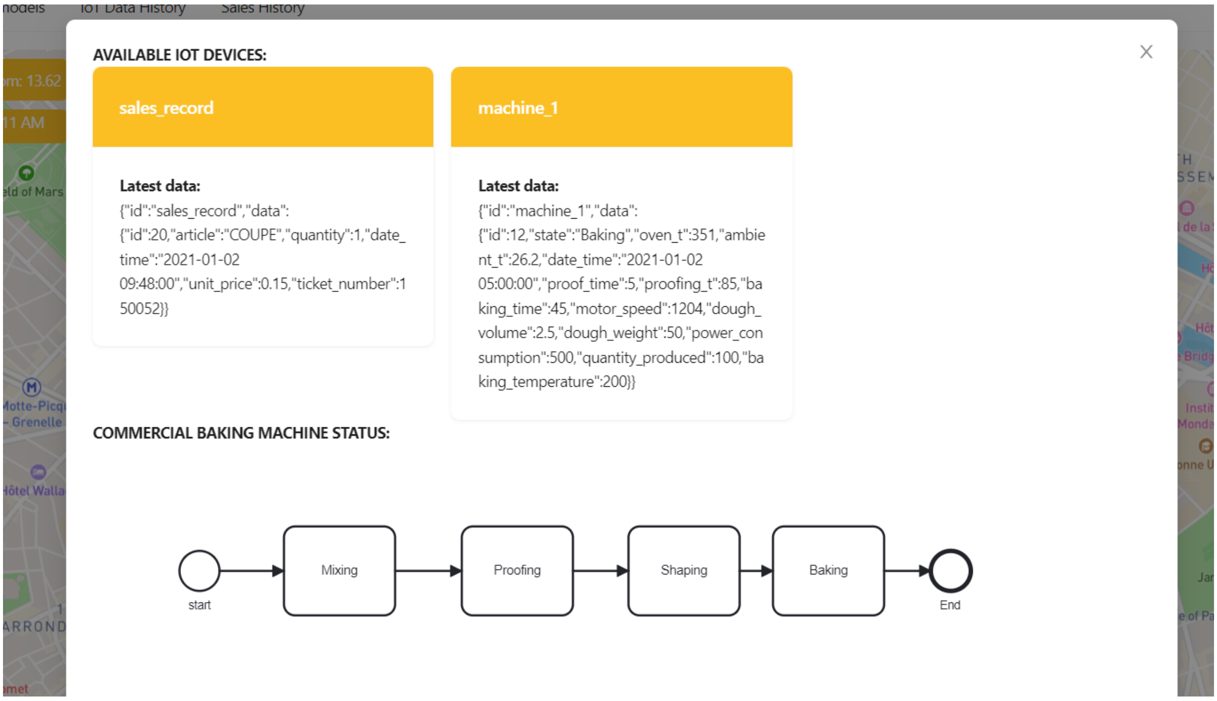


Lng: 2.3282 | Lat: 48.8466 | Zoom: 13.62

Date Time: 9/8/2023, 11:19:36 AM

Figure 23. Main page upper half

- **Participant detail:** Each icon on the map can be clicked to view its detail. Such as in Figure 26, when clicked on the bakery 1, it shows the available IoT devices at the location and its latest data. It also has a Bpmn of the process in the place to make it easy to track what is happening at the time.



AVAILABLE IOT DEVICES:

Latest data:

{"id":"sales_record","data":{"id":20,"article":"COUPE","quantity":1,"date_time":"2021-01-02 09:48:00","unit_price":0.15,"ticket_number":150052}}

Latest data:

{"id":"machine_1","data":{"id":12,"state":"Baking","oven_t":351,"ambient_t":26.2,"date_time":"2021-01-02 05:00:00","proof_time":5,"proofing_t":85,"baking_time":45,"motor_speed":1204,"dough_volume":2.5,"dough_weight":50,"power_consumption":500,"quantity_produced":100,"baking_temperature":200}}

COMMERCIAL BAKING MACHINE STATUS:

Figure 24. Detail modal component

- **BPMN:** The BPMN page show the business process inside of a supply chain, which help in tracking the current process inside of the chain.

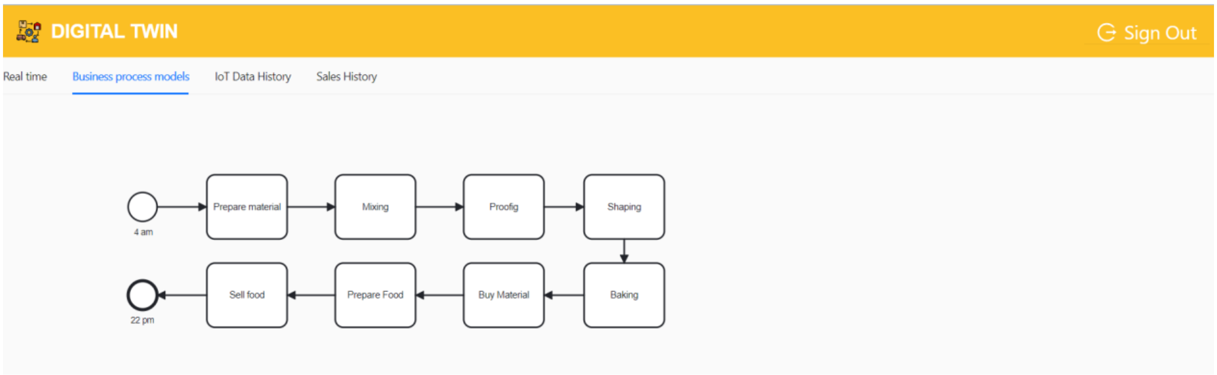
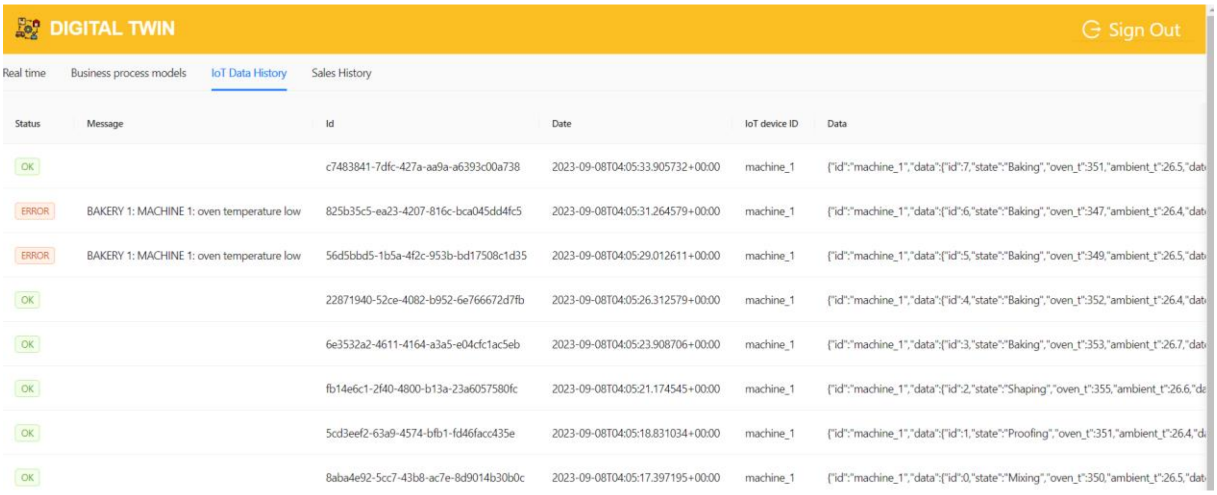


Figure 25. BPMN component

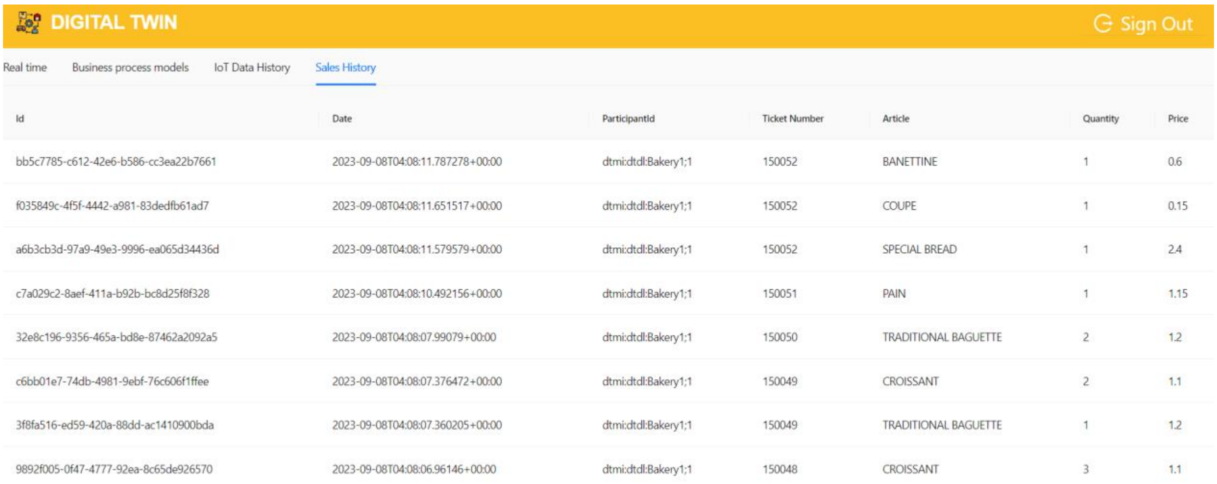
- **IoT data history page:** This page contain the IoT data collected from all registered IoT devices. The user can check if there are problems with the data or if something goes wrong with the physical devices.



Status	Message	Id	Date	IoT device ID	Data
OK		c7483841-7dfc-427a-aa9a-a6393c00a738	2023-09-08T04:05:33.905732+00:00	machine_1	{"id":"machine_1","data":{"id":7,"state":"Baking","oven_t":351,"ambient_t":26.5,"dat
ERROR	BAKERY 1: MACHINE 1: oven temperature low	825b35c5-ea23-4207-816c-bca045dd4fc5	2023-09-08T04:05:31.264579+00:00	machine_1	{"id":"machine_1","data":{"id":6,"state":"Baking","oven_t":347,"ambient_t":26.4,"dat
ERROR	BAKERY 1: MACHINE 1: oven temperature low	56d5bbd5-1b5a-4f2c-953b-bd17508c1d35	2023-09-08T04:05:29.012611+00:00	machine_1	{"id":"machine_1","data":{"id":5,"state":"Baking","oven_t":349,"ambient_t":26.5,"dat
OK		22871940-52ce-4082-b952-6e766672d7fb	2023-09-08T04:05:26.312579+00:00	machine_1	{"id":"machine_1","data":{"id":4,"state":"Baking","oven_t":352,"ambient_t":26.4,"dat
OK		6e3532a2-4611-4164-a3a5-e04cfc1ac5eb	2023-09-08T04:05:23.908706+00:00	machine_1	{"id":"machine_1","data":{"id":3,"state":"Baking","oven_t":353,"ambient_t":26.7,"dat
OK		fb14e6c1-2f40-4800-b13a-23a6057580fc	2023-09-08T04:05:21.174545+00:00	machine_1	{"id":"machine_1","data":{"id":2,"state":"Shaping","oven_t":355,"ambient_t":26.6,"da
OK		5cd3eef2-63a9-4574-bfb1-fd46facc435e	2023-09-08T04:05:18.831034+00:00	machine_1	{"id":"machine_1","data":{"id":1,"state":"Proofing","oven_t":351,"ambient_t":26.4,"d
OK		8aba4e92-5cc7-43b8-ac7e-8d9014b30b0c	2023-09-08T04:05:17.397195+00:00	machine_1	{"id":"machine_1","data":{"id":0,"state":"Mixing","oven_t":350,"ambient_t":26.5,"dat

__Figure 26.__ IoT data history page

- **Sales history page:** This page contain the sales data collected from all store. The user can use to track sales across the chain and manage the supply chain accordingly.



Id	Date	ParticipantId	Ticket Number	Article	Quantity	Price
bb5c7785-c612-42e6-b586-cc3ea22b7661	2023-09-08T04:08:11.787278+00:00	dtmi:dtdl:Bakery1;1	150052	BANETTINE	1	0.6
f035849c-4f5f-4442-a981-83dedfb61ad7	2023-09-08T04:08:11.651517+00:00	dtmi:dtdl:Bakery1;1	150052	COUPE	1	0.15
a6b3cb3d-97a9-49e3-9996-ea065d34436d	2023-09-08T04:08:11.579579+00:00	dtmi:dtdl:Bakery1;1	150052	SPECIAL BREAD	1	2.4
c7a029c2-8aef-411a-b92b-bc8d25f8f328	2023-09-08T04:08:10.492156+00:00	dtmi:dtdl:Bakery1;1	150051	PAIN	1	1.15
32e8c196-9356-465a-bd8e-87462a2092a5	2023-09-08T04:08:07.99079+00:00	dtmi:dtdl:Bakery1;1	150050	TRADITIONAL BAGUETTE	2	1.2
c6bb01e7-74db-4981-9ebf-76c606f1ffee	2023-09-08T04:08:07.376472+00:00	dtmi:dtdl:Bakery1;1	150049	CROISSANT	2	1.1
3f8fa516-ed59-420a-88dd-ac1410900bda	2023-09-08T04:08:07.360205+00:00	dtmi:dtdl:Bakery1;1	150049	TRADITIONAL BAGUETTE	1	1.2
9892f005-0f47-4777-92ea-8c65de926570	2023-09-08T04:08:06.96146+00:00	dtmi:dtdl:Bakery1;1	150048	CROISSANT	3	1.1

__Figure 27.__ Sales history page

4.6 Result

The implemented digital twin is available at the website: https://digitalsupplychain.netlify.app/ and the source code can be accessed at https://github.com/FasterThanLigh1/Supply-Chain-Digital-Twin-React.git . The user must register and then they can start using the digital twin to monitor the bakery supply chain. However, the system at the moment only supports the bakery supply chain, and if the user wants to support a new supply chain a modification in the source code must be made. Improvement for the system in the future could be to make it adapt to many supplies chain.

5. SYSTEM EVALUATION

5.1 Testing

This section will a simple test cases in order to test the function of the digital twin application. In order to test the digital twin, IoT devices are required to be installed at real world location in order to get data. However, because of the lack of access to real world company device in a supply chain, a simulator is used to simulate the IoT devices. The simulator will only simulated the data send back to IoT Hub in order to show that the application can work with real IoT device. The simulator is written in Python and use azure-iot-device library to simulate a device and send mock telemetry to the IoT hub. Although the message sent to IoT hub is simulated, it is guaranteed that the application is still able to work with real IoT device since only the message sending part is simulated.

5.1.1 Dataset description

In order to test the digital twin application, 3 datasets are used as input for simulated devices. The first dataset is French bakery daily sales (Gimbert, 2022) provided the daily sales of a bakery from 01/01/2021 to 30/09/2022. The dataset is 13.9 MB, consists of 6 column and 234005 rows. This data is used as input for the cash register IoT devices which is used to simulated sales data. The second dataset is the commercial bakery machine data which provided the data for the daily usage of a commercial baking machine from 01/01/2021 to 30/09/2022. Because of not having enough data to for the simulation, this dataset is created manually in order to simulate the devices. The third dataset is the truck dataset which provided the data of daily truck freight for the bakery. Because of the lack in dataset provided publicly, the truck dataset is created manually in order to test the digital twin, and is based on the Freight Transport Data (Nanni, 2021). This data is used to simulate the data sent by the data logger inside of the truck cargo.

Sales dataset

Column	**Description**
Date	Date of the recorded data
Time	Time of the recorded data
Ticket_number	Unique identifier
Article	Name of the sold product
Quantity	Sold quantity
Unit_price	Price of the sold product, count in Euro

Machine dataset

Column	**Description**

Date_time	Date time of the recorded data
State	State of the machine at the time, which can be: “Mixing”, “Proofing”, “Shaping” and “Baking”
Oven_t	Temperature of the oven (℃)
Proofing_t	Proofing temperature (℃)
Ambient_t	Ambient temperature of the machine (℃)
Motor_speed	Motor speed of the machine
Power_consumption	Power consumption of the machine
dough_weight	Weight of the dough currently used
dough_volume	Volume of dough currently used
proof_time	Proofing time
baking_time	Baking time
baking_temperature	Baking temperature
quantity_produced	Quantity of bread produced

Truck dataset

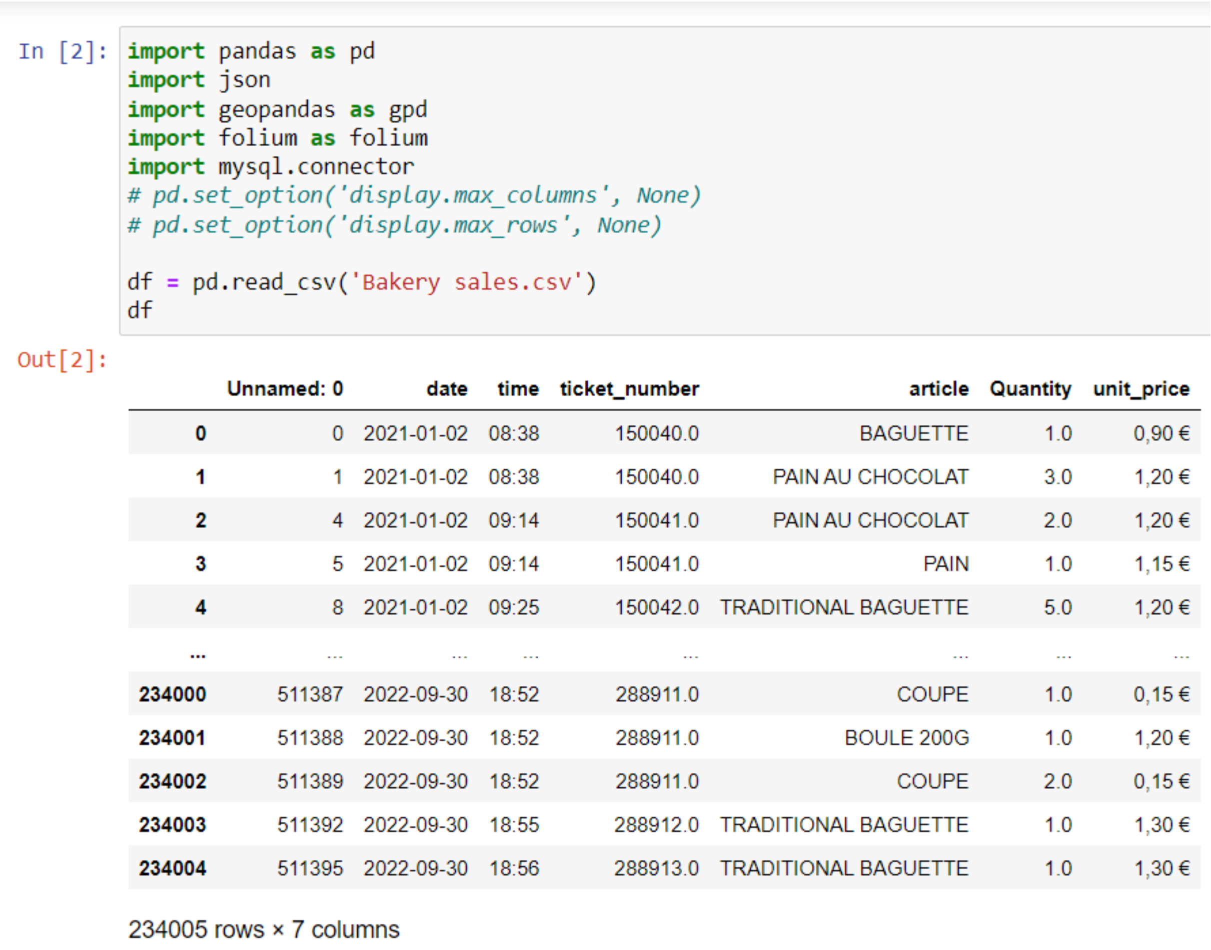
Column	**Description**
Cargo_id	Unique identifier of the cargo
Timestamp	Date time of the recorded data
Latitude	Current latitude of the truck
Longitude	Current longitude of the truck
Temperature	Current temperature inside of the cargo
Humidity	Current humidity inside of the cargo
Shock_level	Current shock level of the cargo
Light_exposure	Current light exposure of the cargo

5.1.2 Data preprocessing

Before loading data to the database, the dataset French bakery daily sales (Gimbert, 2022) must first be processed since it is a public dataset. The other 2 datasets which are machine and truck does not need to be processed since it is generated manually. The dataset consists of 6 column and 234005 rows. Pandas is used as the library for preprocessing and loading the data into the database.

Python Pandas: As an open-source software library built on top of Python specifically for data manipulation and analysis, Pandas offers data structure and operations for powerful, flexible, and easy-to-use data analysis and manipulation. Pandas strengthens Python by giving the popular programming language the capability to work with spreadsheet-like data enabling fast loading, aligning, manipulating, and merging, in addition to other key functions. Pandas is prized for providing highly optimized performance when back-end source code is written in C or Python.

- First the data can be loaded into python using pandas

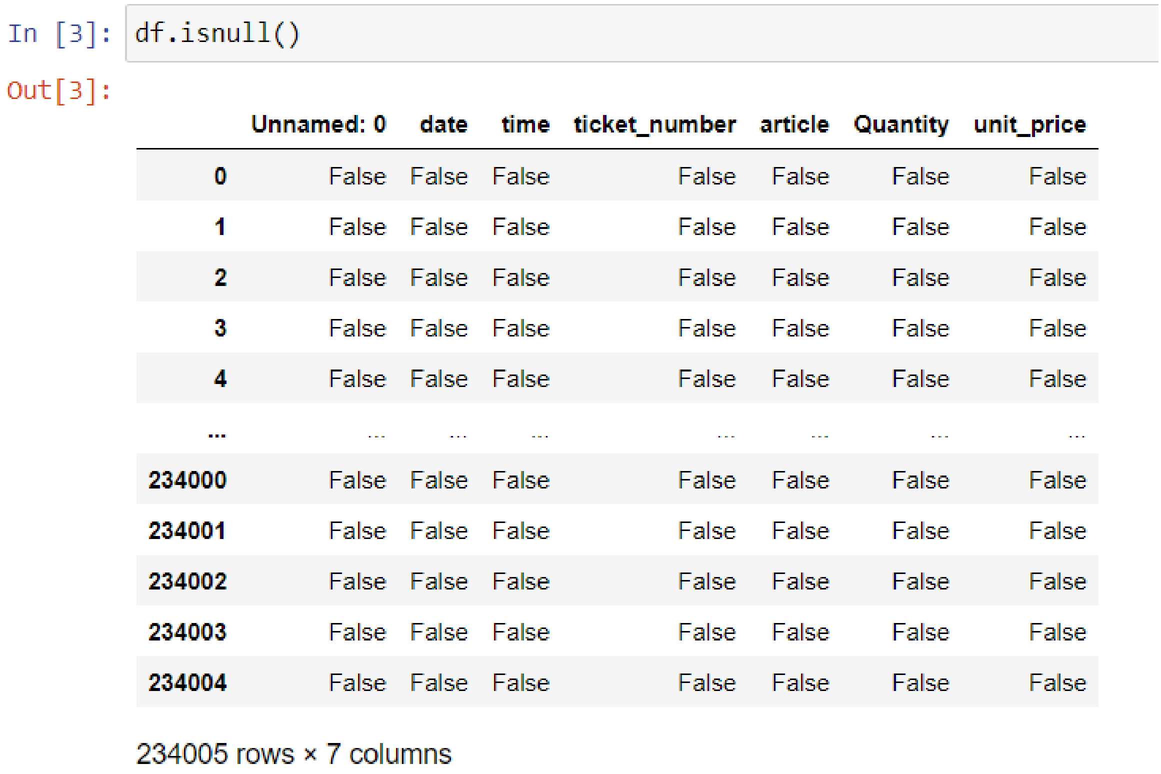


```
In [2]: import pandas as pd
        import json
        import geopandas as gpd
        import folium as folium
        import mysql.connector
        # pd.set_option('display.max_columns', None)
        # pd.set_option('display.max_rows', None)

        df = pd.read_csv('Bakery sales.csv')
        df
```

```
Out[2]:
        Unnamed: 0        date   time  ticket_number               article  Quantity  unit_price
      0          0  2021-01-02  08:38       150040.0              BAGUETTE       1.0      0,90 €
      1          1  2021-01-02  08:38       150040.0     PAIN AU CHOCOLAT       3.0      1,20 €
      2          4  2021-01-02  09:14       150041.0     PAIN AU CHOCOLAT       2.0      1,20 €
      3          5  2021-01-02  09:14       150041.0                  PAIN       1.0      1,15 €
      4          8  2021-01-02  09:25       150042.0  TRADITIONAL BAGUETTE       5.0      1,20 €
    ...        ...         ...    ...            ...                   ...       ...         ...
 234000     511387  2022-09-30  18:52       288911.0                 COUPE       1.0      0,15 €
 234001     511388  2022-09-30  18:52       288911.0            BOULE 200G       1.0      1,20 €
 234002     511389  2022-09-30  18:52       288911.0                 COUPE       2.0      0,15 €
 234003     511392  2022-09-30  18:55       288912.0  TRADITIONAL BAGUETTE       1.0      1,30 €
 234004     511395  2022-09-30  18:56       288913.0  TRADITIONAL BAGUETTE       1.0      1,30 €

234005 rows × 7 columns
```

***Figure 28.** Loading data*

- Then a null check can be given to see if the data contain corrupted or NaN data:



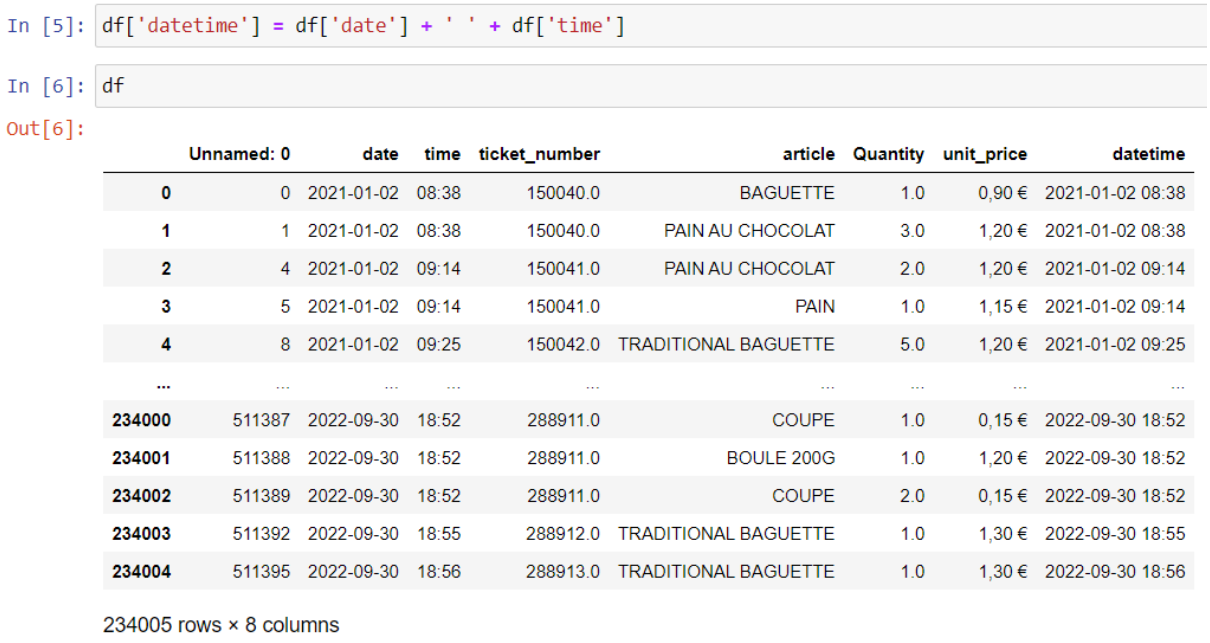
```
In [3]: df.isnull()
Out[3]:
```

	Unnamed: 0	date	time	ticket_number	article	Quantity	unit_price
0	False	False	False	False	False	False	False
1	False	False	False	False	False	False	False
2	False	False	False	False	False	False	False
3	False	False	False	False	False	False	False
4	False	False	False	False	False	False	False
...	...	...	...	...	...	...	...
234000	False	False	False	False	False	False	False
234001	False	False	False	False	False	False	False
234002	False	False	False	False	False	False	False
234003	False	False	False	False	False	False	False
234004	False	False	False	False	False	False	False

234005 rows × 7 columns

Figure 29. *Check for null*

- If there is no null data, the next step is to transform data type in each column. The date and time column can be combined into a single column for easier reading by the system.



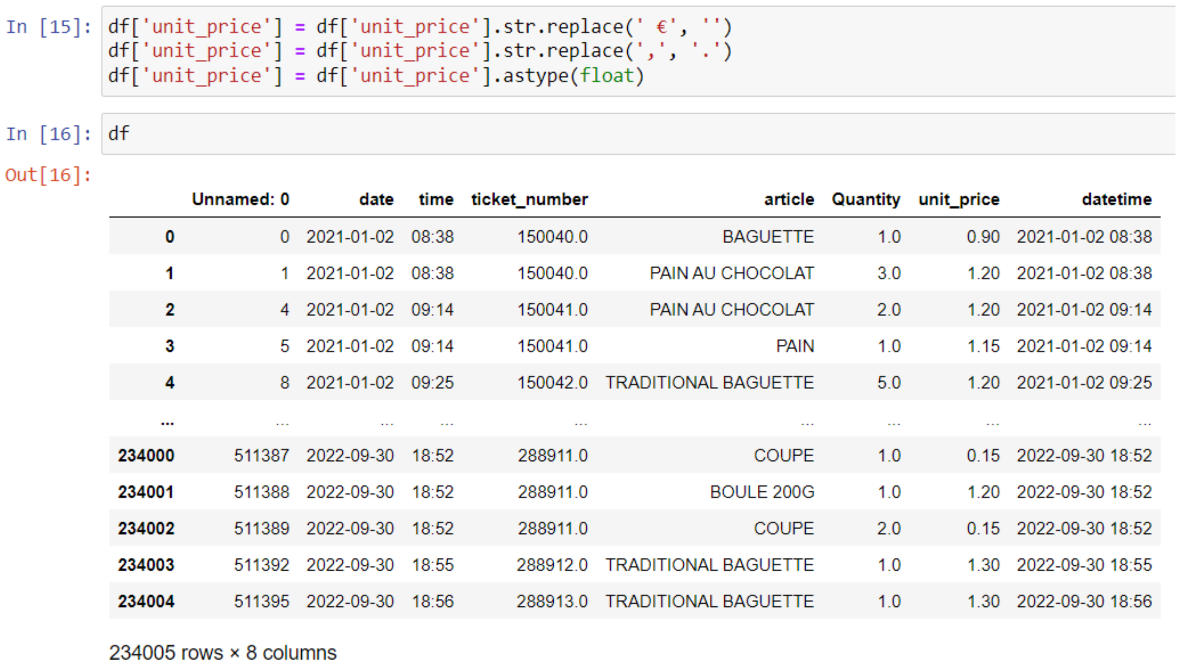
```
In [5]: df['datetime'] = df['date'] + ' ' + df['time']

In [6]: df
Out[6]:
```

	Unnamed: 0	date	time	ticket_number	article	Quantity	unit_price	datetime
0	0	2021-01-02	08:38	150040.0	BAGUETTE	1.0	0,90 €	2021-01-02 08:38
1	1	2021-01-02	08:38	150040.0	PAIN AU CHOCOLAT	3.0	1,20 €	2021-01-02 08:38
2	4	2021-01-02	09:14	150041.0	PAIN AU CHOCOLAT	2.0	1,20 €	2021-01-02 09:14
3	5	2021-01-02	09:14	150041.0	PAIN	1.0	1,15 €	2021-01-02 09:14
4	8	2021-01-02	09:25	150042.0	TRADITIONAL BAGUETTE	5.0	1,20 €	2021-01-02 09:25
...	...	...	...	...	...	...	...	...
234000	511387	2022-09-30	18:52	288911.0	COUPE	1.0	0,15 €	2022-09-30 18:52
234001	511388	2022-09-30	18:52	288911.0	BOULE 200G	1.0	1,20 €	2022-09-30 18:52
234002	511389	2022-09-30	18:52	288911.0	COUPE	2.0	0,15 €	2022-09-30 18:52
234003	511392	2022-09-30	18:55	288912.0	TRADITIONAL BAGUETTE	1.0	1,30 €	2022-09-30 18:55
234004	511395	2022-09-30	18:56	288913.0	TRADITIONAL BAGUETTE	1.0	1,30 €	2022-09-30 18:56

234005 rows × 8 columns

Figure 30. *Combine row*

- Next the unit_price column is transformed into float for better analysis.

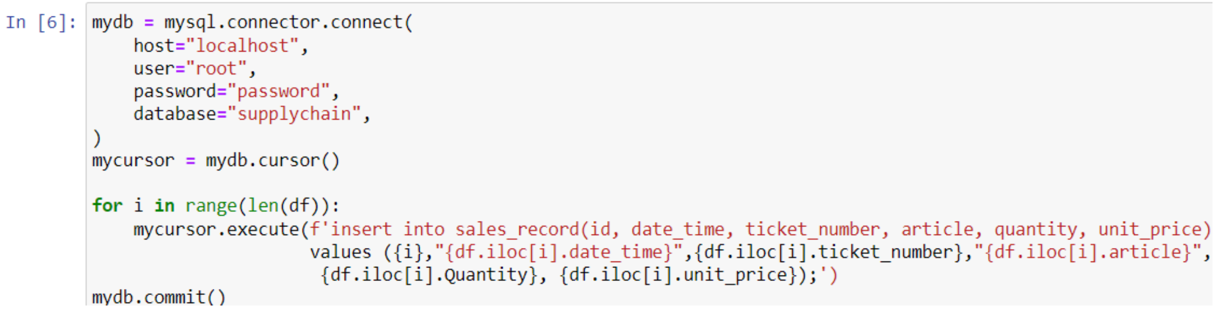


```
In [15]: df['unit_price'] = df['unit_price'].str.replace(' €', '')
         df['unit_price'] = df['unit_price'].str.replace(',', '.')
         df['unit_price'] = df['unit_price'].astype(float)

In [16]: df

Out[16]:
          Unnamed: 0        date   time  ticket_number               article  Quantity  unit_price          datetime
       0           0  2021-01-02  08:38       150040.0              BAGUETTE       1.0        0.90  2021-01-02 08:38
       1           1  2021-01-02  08:38       150040.0      PAIN AU CHOCOLAT       3.0        1.20  2021-01-02 08:38
       2           4  2021-01-02  09:14       150041.0      PAIN AU CHOCOLAT       2.0        1.20  2021-01-02 09:14
       3           5  2021-01-02  09:14       150041.0                  PAIN       1.0        1.15  2021-01-02 09:14
       4           8  2021-01-02  09:25       150042.0  TRADITIONAL BAGUETTE       5.0        1.20  2021-01-02 09:25
     ...         ...         ...    ...            ...                   ...       ...         ...               ...
  234000      511387  2022-09-30  18:52       288911.0                 COUPE       1.0        0.15  2022-09-30 18:52
  234001      511388  2022-09-30  18:52       288911.0             BOULE 200G       1.0        1.20  2022-09-30 18:52
  234002      511389  2022-09-30  18:52       288911.0                 COUPE       2.0        0.15  2022-09-30 18:52
  234003      511392  2022-09-30  18:55       288912.0  TRADITIONAL BAGUETTE       1.0        1.30  2022-09-30 18:55
  234004      511395  2022-09-30  18:56       288913.0  TRADITIONAL BAGUETTE       1.0        1.30  2022-09-30 18:56

234005 rows × 8 columns
```

Figure 31. *Transform unit_price column*

After processing, the data can be safely loaded into the database of choices. During this implementation, the database MySQL will be used to store the mock data.

MySQL: is an open-source relational database management system (RDBMS). A relational database arranges data into one or more tables where it is possible for the data to be connected to one another. The relational database's data can be created, modified, and extracted using the SQL programming language, which is also used to manage user access to the database. An RDBMS, such as MySQL, works with an operating system to implement a relational database in a computer's storage system, manages users, permits network access, and makes it easier to evaluate database integrity and create backups.



```
In [6]: mydb = mysql.connector.connect(
            host="localhost",
            user="root",
            password="password",
            database="supplychain",
        )
        mycursor = mydb.cursor()

        for i in range(len(df)):
            mycursor.execute(f'insert into sales_record(id, date_time, ticket_number, article, quantity, unit_price)
                                 values ({i},"{df.iloc[i].date_time}",{df.iloc[i].ticket_number},"{df.iloc[i].article}",
                                  {df.iloc[i].Quantity}, {df.iloc[i].unit_price});')
        mydb.commit()
```

Figure 32. *Loading data to MySQL*

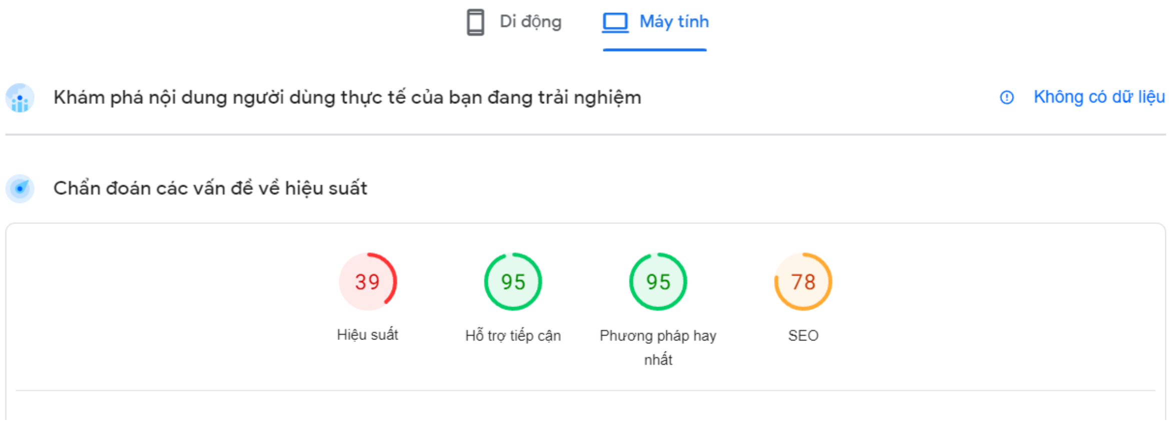
How mock data is loaded into simulated device

Threading: In order to simulated different devices running at the same time, Python Threading library is used to make use of multithreading. A thread is a separate flow of execution and in this way multithreading is to execute multiple, concurrent threads.

```
import mysql.connector

import threading

#CONNECT TO MYSQL SERVER
mydb = mysql.connector.connect(
  host="localhost",
  user="root",
  password="password",
    database="supplychain"
)
mycursor = mydb.cursor()

#CREATE DEVICE WITH CONNECTION STRING
connection_string = "HostName=DemoIOT.azure-devices.net;DeviceId=retailer_1_cashier;SharedAccessKey=cOF/JtQT753ODbo4u6+ywBzveUU7o2phhnZXccfbO0Y="
iot_device = IoTHubDeviceClient.create_from_connection_string(connection_string)

#FUNCTION TO SEND MOCK PAYLOAD TO IOT HUB
def send_monitor_telemetry(device_client):
    try:
        # Instantiate the simulated temperature sensor
        while True:
            #Fetch continuous row from dataset
            myresult = mycursor.fetchone()
            #Transform the fetch data to payload JSON format
            payload = TransformPayload(myresult)
            payload = payload
            message = Message(payload)
            # Add custom properties to the message if desired
            # Send the telemetry message
            print(f"Sending telemetry: {payload}")
            device_client.send_message(message)
            time.sleep(1)   # Delay between sending data

#USE THREADING TO START THE RUNNING THE DEVICES SIMULTANEOUSLY
thread = threading.Thread(target=send_monitor_telemetry, args=(iot_device))
thread.start() thread.join()
```

The dataset is first stored in MySQL server on a local machine. First, a connection to the MySQL server is made in order get the dataset. Then a device is created based on the connection string which is from the device created on the IoT Hub. Then threading is used to execute multiple thread of the function *send_monitor_telemetry* in order to simulated multiple devices running at one. The payload is then sent to the IoT Hub which will be read by Azure Event Hub and notify the changes to all subscribed clients.

5.1.3 User interface



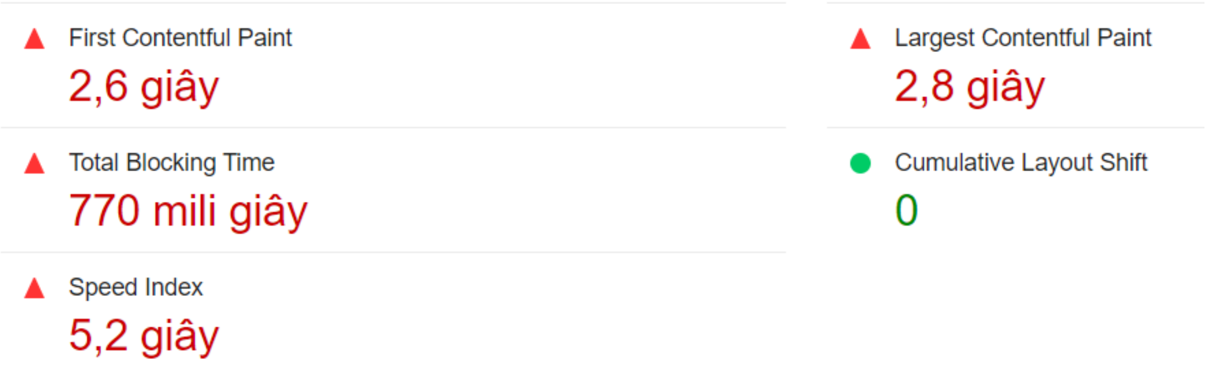
Di động | Máy tính

Khám phá nội dung người dùng thực tế của bạn đang trải nghiệm — Không có dữ liệu

Chẩn đoán các vấn đề về hiệu suất

39	95	95	78
Hiệu suất	Hỗ trợ tiếp cận	Phương pháp hay nhất	SEO

Figure 33. *Audit score*



▲ First Contentful Paint
2,6 giây

▲ Largest Contentful Paint
2,8 giây

▲ Total Blocking Time
770 mili giây

● Cumulative Layout Shift
0

▲ Speed Index
5,2 giây

Figure 34. *Performance attributes*

The UI will be tested using Googe Lighthouse, which is an open-source software that can measure a page for performance, accessibility, SEO, and more. According to the audit score (Figure 33), a 39 points in performance means that the website load very slow. Further inspection (Figure 34) shows that all of the important attributes which is: First Contentful Paint (Measures the time at which the first text or image becomes visible to users), Largest Contentful Paint (Calculates the time a page takes to load its largest element for users), Total Blocking Time (Measures the amount of time that a page is blocked from reacting to user input, like a mouse click), Cumulative Layout Shift (Measures the layout shifts that occur as users access a page) and Speed Index (Shows how quickly the content of a page is loaded) all load very slowly. Further improvement needed to be done in this section to improve performance.

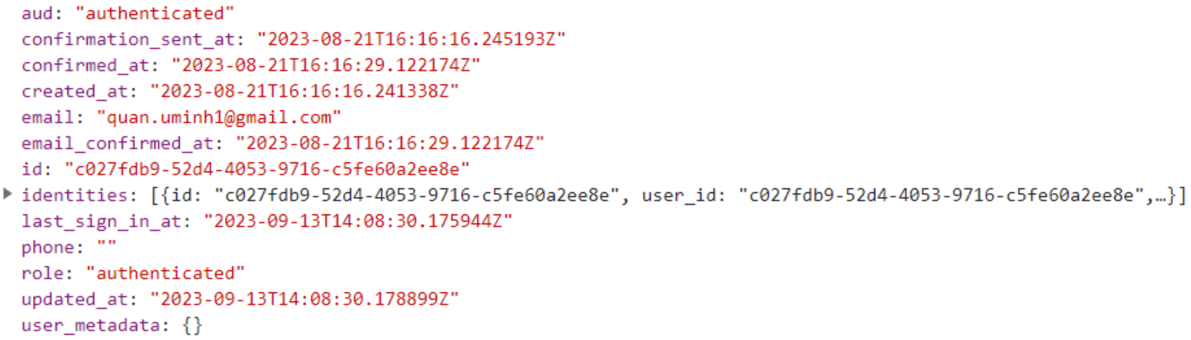
A score of 95 in accessibility show that people with visual, auditory, motor, or cognitive impairments can use the pages. Also a score of 95 in best practices indicates that the website has adopted methods and guidelines that are considered effective and efficient. However a score of 78 in search engine optimization shows that the website need to be more optimized in order to improve their visibility and rankings in search engine results pages (SERPs).

5.1.4 Test case definition

Test scenario: User login

Test case 1:

- **Test description:** User successfully login using registered credentials.
- **Pre-Conditions: User at main page**
- **Test Steps:**
    1. User enter username
    2. User enter password
    3. User press submit
- **Test Data:**
    - Username: quan.uminh1@gmail.com
    - Password: 123456
- **Expected Result:** User should be able to successfully login and the server return user information.
- **Actual Result**



```
  aud: "authenticated"
  confirmation_sent_at: "2023-08-21T16:16:16.245193Z"
  confirmed_at: "2023-08-21T16:16:29.122174Z"
  created_at: "2023-08-21T16:16:16.241338Z"
  email: "quan.uminh1@gmail.com"
  email_confirmed_at: "2023-08-21T16:16:29.122174Z"
  id: "c027fdb9-52d4-4053-9716-c5fe60a2ee8e"
▶ identities: [{id: "c027fdb9-52d4-4053-9716-c5fe60a2ee8e", user_id: "c027fdb9-52d4-4053-9716-c5fe60a2ee8e",…}]
  last_sign_in_at: "2023-09-13T14:08:30.175944Z"
  phone: ""
  role: "authenticated"
  updated_at: "2023-09-13T14:08:30.178899Z"
  user_metadata: {}
```

***Figure 35.** System response*

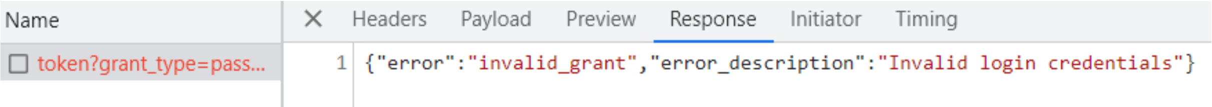
- **Status:** Pass

Test case 2:

- **Test description:** User will not be able to login without the correct credentials.
- **Pre-Conditions: User at main page**
- **Test Steps:**
    1. User enter username
    2. User enter password

3. User press submit

- **Test Data:**
  - Username: quan.uminh1111@gmail.com
  - Password: 12345678327831783
- **Expected Result:** User should be able to successfully login and the server return user information.
- **Actual Result**



```
{"error":"invalid_grant","error_description":"Invalid login credentials"}
```

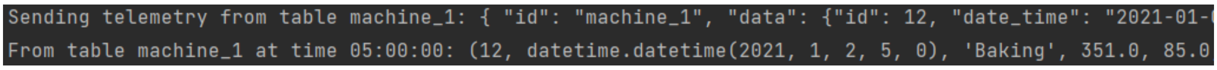
Figure 36. *System response*

- **Status:** Pass

Test scenarios: Monitoring the supply chain

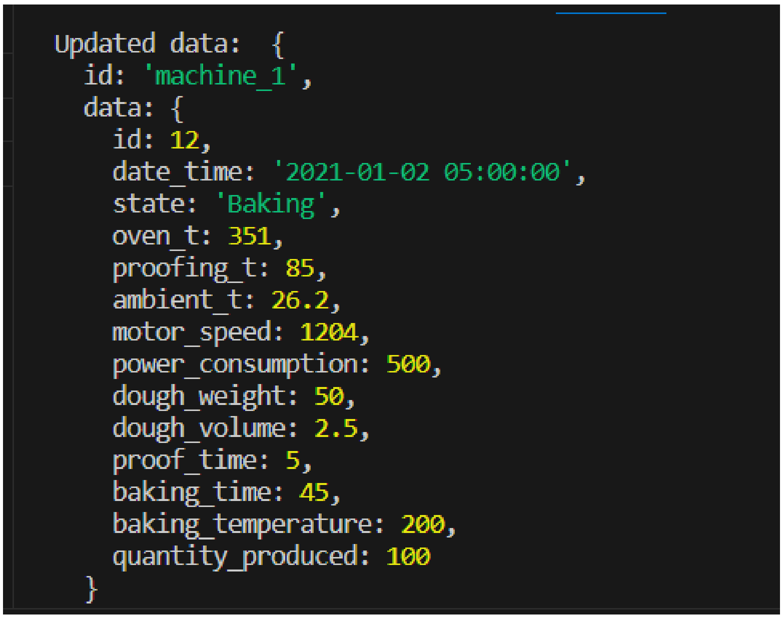
Test case 3:

- **Test description:** The system must be able to get changes made to the database and made changes with the UI in respond.
- **Pre-Conditions:** User has successfully login and is at the main page. The backend API must also be started in order to listen to data sent by the IoT devices
- **Test Steps:**
  1. Start the backend.
  2. Start the simulation program.
  3. Watch for changes in the main page.
- **Test Data:** The test data used in this test case will be the 3 dataset described in part 5.1.1.
- **Expected Result:** The backend is able to get all data sent by the IoT devices and the UI changes corresponding to the changes in data.
- **Actual Result**



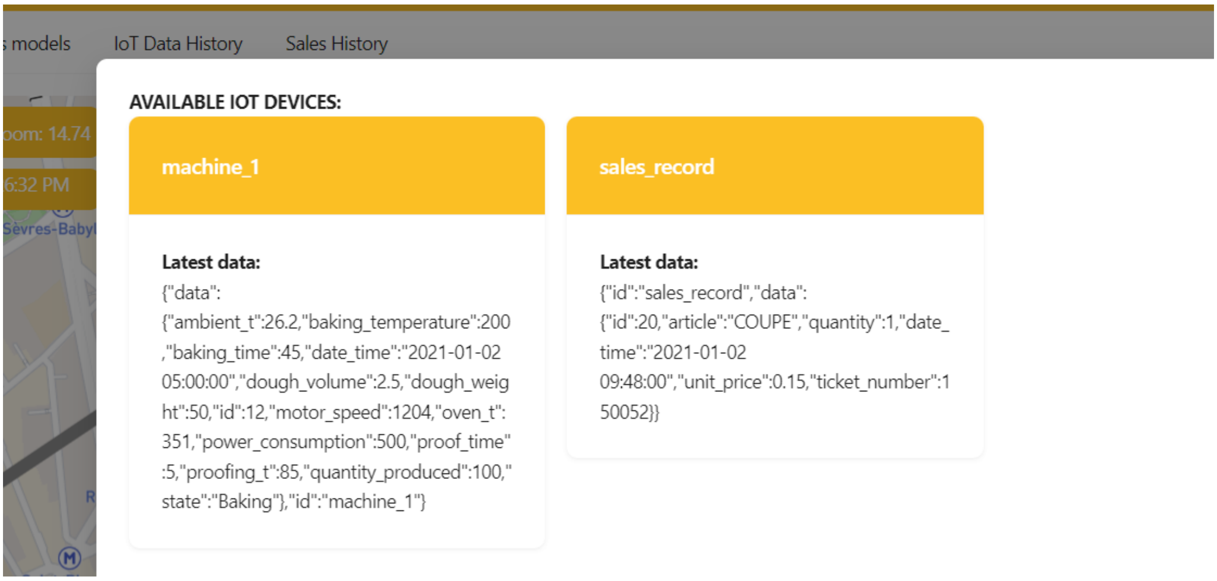
```
Sending telemetry from table machine_1: { "id": "machine_1", "data": {"id": 12, "date_time": "2021-01-
From table machine_1 at time 05:00:00: (12, datetime.datetime(2021, 1, 2, 5, 0), 'Baking', 351.0, 85.0
```

Figure 37. *Data sent from simulation*



```
Updated data:  {
  id: 'machine_1',
  data: {
    id: 12,
    date_time: '2021-01-02 05:00:00',
    state: 'Baking',
    oven_t: 351,
    proofing_t: 85,
    ambient_t: 26.2,
    motor_speed: 1204,
    power_consumption: 500,
    dough_weight: 50,
    dough_volume: 2.5,
    proof_time: 5,
    baking_time: 45,
    baking_temperature: 200,
    quantity_produced: 100
  }
```

Figure 38. *Data received by the backend*



AVAILABLE IOT DEVICES:

machine_1

Latest data:

{"data":{"ambient_t":26.2,"baking_temperature":200,"baking_time":45,"date_time":"2021-01-02 05:00:00","dough_volume":2.5,"dough_weight":50,"id":12,"motor_speed":1204,"oven_t":351,"power_consumption":500,"proof_time":5,"proofing_t":85,"quantity_produced":100,"state":"Baking"},"id":"machine_1"}

sales_record

Latest data:

{"id":"sales_record","data":{"id":20,"article":"COUPE","quantity":1,"date_time":"2021-01-02 09:48:00","unit_price":0.15,"ticket_number":150052}}

Figure 39. *Changes in UI*

- **Status:** Pass

5.2 Evaluation

Features: Features that the have been implemented in the digital twin

- **Data Integration:** The digital twin can connect with third-party database applications like Supabase and can collect data from any IoT devices through the use of Azure IoT Hub.
- **Physical to digital link using IoT devices**: The digital twin makes use of Azure IoT Hub and creates a connection with IoT devices to create a real-time link between the physical entities and its twins.

- **Simulation and Optimization**: The digital twin can simulate a supply chain including all its participant and inside process. The twin can also track the order and freight truck which is currently active using IoT devices.
- **Real-Time Monitoring**: The application provides a real-time mapping between the physical entities and its twin, which can help user track the current statistics of the supply chain in real-time.
- **Scalability and Flexibility**: The digital twin is able to add new warehouses, vehicles or businesses depending on the organization size. The digital twin is capable of adding many data through the use of real-time database like Azure IoT Hub.
- **Performance**: The digital twin is able to handle up to 6 IoT devices sending data at the same time.
- **Security:** The application use login credential to protect organization's sensitive information**.**
- **Reliability**: The application can maintain a connection with the IoT hub as well as the database at all times using MQTT protocol.
- **Usability**: The system is easy to use and navigate since all the information is only in one page.
- **Flexibility**: The digital twin is able to modify the supply chain based on the organization needs.
- **Interoperability**: Since using Azure cloud, the digital twin can also integrate with other Azure services such as Azure Event Hub, making it flexible and customizable based on organization needs.
- **Scalability**: Since using third-party provider like Supabase as a database storage and Azure IoT Hub as a IoT service, the application can scale up or down by customizing third-party subscription. Supabase subscription can range from free to 599$/month and can increase the database storage of the user. Azure IoT Hub is made to scale because it can handle hundreds of telemetry at the same time.

Strength:

- **Low cost:** The digital twin is mostly built using open-source technologies which can be helpful for business that don't have a lot of funds.
- **Running on website:** Because of the application nature as a web application, the digital twin can run on multiple web browsers such as Google Chrome, Microsoft Edge or Mozilla Firefox, thus making it fast and easy to access. It also reduces the system requirements to use since no installation is needed.

Weakness:

- **Heavy setup require:** Although the digital twin can model a supply chain and provide a real-time connection with its physical entities using IoT devices, setting up a model requires a lot of time and steps. First IoT devices must be registered in the Azure IoT hub. Second, the database must also be updated to know what data to store. Third, the backend must be updated to handle received data from IoT devices and send it corresponding to the database. Forth and finally, the database and UI must be modified to reflect the correct supply chain model in order to update data correctly. Each step will become lengthier as the size of the organization increase. Improving the digital twin can be made in this area to reduce the complexity of modifying the digital model.

- **Model part of a supply chain:** The supply chain in reality is very complex even the simplest chain can contain many data. Therefore it is impossible to simulate every aspect of the supply chain. The supply chain model the result digital twin only reflect the primary aspect of the chain. Improvement in this area may involve making a supply chain as detail as possible.
- **Handle little incoming IoT device simultaneously:** Although the goal is to handle more than 20 devices at the same time, after the implementation, the digital twin UI can only handle at best 6 device at the same time. Handling more than 6 device will reduce the update time and cause a time lag in rendering the UI. Improvement can be made in this area to help the system handle more device without causing delay to the system.
- **Depend on third-party server:** Built on the Azure IoT Hub, its performance depend on the server status of Azure. Having a private IoT Hub can remove the dependencies on Azure.

6. CONCLUSION

6.1 Conclusion

The thesis provide an understanding into the new and exciting concept of supply chain digital twin. The application also provide understanding into how to build a digital twin and applying it in the supply chain field. The result digital twin can collect and connect data from various sources through the use of IoT devices, it can simulate a supply chain from the start to the end customer and provide real time monitoring on the supply chain. For this thesis, the digital twin is built using open source technology and although has all the core technology of a digital twin, it does not fully modeled a real world supply chain. The full modelling of a real world supply chain is a complex process therefore the result twin only model a part but not all of the supply chain. This thesis contribute to the building of a supply chain digital twin, which will continue to rise in the near future since many company have already developed their own digital twin and according to Gartner Hype Cycle (Gartner, 2019), it will continue to rise for the next 10 years. Company and researchers can use this thesis as a reference to gain a basic understanding and how to build a supply chain digital twin.

6.2 Future research

During this thesis, the author creates a web application for the supply chain digital twin and is capable of connecting with IoT device and create a real time connection between the twin and its physical counterpart. However, in order to deploy the application and provide a practical used for the user, many functions should be added such as machine learning or mobile compatible. Applying machine learning can help user predict the result and state of the supply chain and a exposing the API can help programmer create many scenarios for testing.

REFERENCES

Ageron, B., Gunasekaran, A., Spalanzani, A.. (2012). *Sustainable supply management: An empirical study*. International journal of production economics, Vol. 140, 2011. (Accessed: July 29, 2023).

AnyLogistix. *Supply chain digital twins, definition, the problems they solve, and how to develop them*. Access from: https://www.anylogistix.com/resources/white-papers/supply-chain-digital-twins/ . (Accessed: July 29, 2023).

Barykin, S.Y., Bochkarev, A.A., Kalinina, O.V. and Yadikin, V.K.. (2020). *Concept for a Supply Chain Digital Twin*. International Journal of Mathematical, Engineering and Management Sciences, Vol. 5, No. 6, 1498-1515, 2020. (Accessed: July 29, 2023).

Better buildings, *Ford Motor company: Dearborn campus uses a digital twin tool for energy plant management.* Available at: https://betterbuildingssolutioncenter.energy.gov/implementation-models/ford-motor-company-dearborn-campus-uses-a-digital-twin-tool-energy-plant . (Accessed: July 29, 2023).

Cooper, M.C., Lambert, D.M. and Pagh, J.D.. (1997). *Supply Chain Management: More Than a New Name for Logistics.* The International Journal of Logistics Management. (Accessed: July 29, 2023).

Douglas Insight, Digital Twin Market Is On The Rise In The Next 10 Years - Report Comparison on Douglas Insights. Available at: https://www.globenewswire.com/en/news-release/2022/11/11/2554108/0/en/Digital-Twin-Market-Is-On-The-Rise-In-The-Next-10-Years-Report-Comparison-on-Douglas-Insights.html . (Accessed: July 29, 2023).

Essakly, A., Wichmann, M., and Spengler, T.S. (2019). *A reference framework for the holistic evaluation of Industry 4.0 solutions for small-and medium-sized enterprises.* Institute of Automotive Management and Industrial Production, 38118 Braunschweig Germany. (Accessed: July 29, 2023).

Fuller, A., Fan, Z., Day, C. and Barlow, C.. (2020). *Digital Twin: Enabling Technologies, Challenges and Open Research*. IEEE Access, vol. 8, pp. 108952-108971, 2020, doi: 10.1109/ACCESS.2020.2998358. (Accessed: July 29, 2023).

Gartner. (2019), *Gartner Survey Reveals Digital Twins Are Entering Mainstream Use*. Available at: https://www.gartner.com/en/newsroom/press-releases/2019-02-20-gartner-survey-reveals-digital-twins-are-entering-mai?_its=JTdCJTIydmlkJTIyJTNBJTIyJTIyJTJDJTIyc3RhdGUlMjIlM0ElMjIlMjIlN0Q= . (Accessed: July 29, 2023).

Gimbert, M.. (2022), *French bakery daily sales.* Available at: https://www.kaggle.com/datasets/matthieugimbert/french-bakery-daily-sales. (Accessed: July 29, 2023).

Graham C. Stevens. (1989). *Integrating the Supply Chain*. International Journal of Physical Distribution & Materials Management. (Accessed: July 29, 2023).

Hugos, M.. (2011). *Essentials of Supply Chain Management.* Third Edition. John Wiley & Sons. (Accessed: July 29, 2023).

Ivanov, D., Dolgui, A.. (2019). *New disruption risk management perspectives in supply chains: digital twins, the ripple effect, and resileanness.* Berlin School of Economics and Law, Department of Business Administration Supply Chain Management, 10825 Berlin, Germany. (Accessed: July 29, 2023).

Ivanov, D. (2021). *Supply chain simulation and optimization with anyLogistix. 5th, updated edition*. Berlin School of Economics and Law. (Accessed: July 29, 2023).

Jayaram, A.. (2016). *Lean six sigma approach for global supply chain management using industry 4.0 and IIoT.* 2nd International Conference on Contemporary Computing and Informatics (IC3I), 2016, pp. 89-94, doi: 10.1109/IC3I.2016.7917940. (Accessed: July 29, 2023).

Kalogiannidis, S. (2020). *Covid impact on small businesses.* International Journal of Social Science and Economics Invention (December 2020). (Accessed: July 29, 2023).

Lambert, D., Martha, C.. (2000). *Issues in Supply Chain Management*. Issues in Supply Chain Management, Industrial Marketing Management, Volume 29, Issue 1, 2000, Pages 65-83, ISSN 0019-8501. (Accessed: July 29, 2023).

Mentzer, J.T., DeWitt, W., Keebler, J.S., Min, S., Nix, N.W., Smith, D.C., Zacharia, Z.G.. (2001). *Defining supply chain management*. Journal of business logic, Volume22, Issue 2, Autumn 2001, Pages 1-25. (Accessed: July 29, 2023).

Microsoft. *The Process Digital Twin: A step towards operational excellence*. Available at: https://info.microsoft.com/rs/157-GQE-382/images/Digital%20Twin%20Vision.pdf. (Accessed: July 29, 2023).

Monczka, R.M., Handfield, R.B., Giunipero, L.C., and Patterson, J.L.. (2009). *Purchasing and Supply Chain Management*. Fourth edition. Cincinnati, OH: South-Western College Publishing, Chapter 8. (Accessed: July 29, 2023).

Nanni. (2021). *Freight Transport Data.* Available at: https://www.kaggle.com/datasets/giobbu/belgium-obu. (Accessed: July 29, 2023).

Sri, J.S., Settani, E., Tsolakis, N. and Aulakh P.K.. (2019). *Supply Chain Digital Twins: Opportunities and Challenges Beyond the Hype.* University of Cambridge. (Accessed: July 29, 2023).