1 2 9 0

UNIVERSIDADE Đ
COIMBRA

Hermínio Manuel Simões da Silva Tão Espírito Santo

DIGITAL TWINS APPLIED TO SUSTAINABILITY AS A SERVICE

Dissertation in the context of the Master in Informatics Engineering, specialization in Software Engineering, advised by Professor Pedro Furtado and presented to the Department of Informatics Engineering of the Faculty of Sciences and Technology of the
University of Coimbra.

July 2022

1 2 9 0

FACULDADE DE
CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE Đ
COIMBRA

DEPARTAMENTO DE ENGENHARIA INFORMÁTICA

Hermínio Manuel Simões da Silva Tão Espírito Santo

DIGITAL TWINS APPLIED TO SUSTAINABILITY AS A SERVICE

Dissertation in the context of the Master in Informatics Engineering, specialization in Software Engineering, advised by Professor Pedro Furtado and presented to the Department of Informatics Engineering of the Faculty of Sciences and Technology of the
University of Coimbra.

July 2022

This page is intentionally left blank

Acknowledgements

I would like to express my sincere thanks to my supervisor, Professor Pedro Furtado from the Department of Informatics Engineering, André Duarte, and Stefan Loureiro from Ubiwhere, whose expertise was invaluable in formulating the research questions and methodology.

I would also like to express my gratitude to my friends and girlfriend for their collaboration. Finally, I would like to thank my parents for their wise counsel and sympathetic ear.

This page is intentionally left blank

Abstract

Associated with the rapid industrial growth that is being witnessed in this century, companies have started to adopt new measures so that they do not fall behind. Most companies initiated the process of digitizing all their processes, with the goal of reducing costs, saving space, reducing the environmental footprint. As a result, the Digital Twin technology emerged as one of the best solutions for companies.

This project has the objective of creating a Digital Twin capable of replicating a given textile supply chain, by digitizing all the physical processes inside the given model. The following document presents a Digital Twin solution for the problem posed; it proposes an architecture for the system as well as all the requirements needed. This Digital Twin will follow a textile supply chain that utilizes cotton to manufacture cloth as the end product. The main feature of this Digital Twin is the ability to measure the levels of water usage that is consumed in each instance of the supply chain, furthermore it will be possible to calculate the amount of electrical energy that is spent inside of each process inside the supply chain. With access to this information, it will be possible to adjust the processes with the goal of decreasing the electrical energy consumption alongside the water usage, to promote a sustainable future for a given organization. The solution proposed for this work is a Proof of Concept that will serve as basis for a future product which will be developed by Ubiwhere.

Keywords

Digital Twin, Supply Chain, Sustainability, Industry 4.0

This page is intentionally left blank

Resumo

Associado ao rápido crescimento industrial que está a ocorrer neste século, as empresas começaram a adotar novas medidas com o objetivo de não ficarem para trás. A maioria das empresas iniciaram o processo de digitalização de todos os seus processos com o intuito de reduzir custos, poupar espaço e diminuir a pegada ambiental. Como resultado, a tecnologia Digital Twin surge como uma das melhores soluções para as empresas.

Este projeto tem o objetivo de criar um Digital Twin capaz de replicar uma cadeia de abastecimento da indústria têxtil digitalizando todos os seus processos físicos dentro do modelo dado. O documento apresenta uma solução Digital Twin para o problema apresentado; uma arquitetura para o sistema e os seus requisitos são propostos neste documento. Este Digital Twin segue uma cadeia de abastecimento do setor têxtil, onde o algodão é o produto que entra no sistema e uma peça de roupa é o produto final. A principal funcionalidade deste Digital Twin é o cálculo da quantidade de energia elétrica e de água gasta por cada produto dentro do sistema, cada processo vai ter um valor associado de gastos. Com acesso a esta informação, vai ser possível visualizar o custo que uma quantidade qualquer de algodões que se usem para a produção vai ter no sistema todo e a partir daí encontrar alternativas para reduzir o consumo destes dois bens, isto tudo tem o objetivo de promover um futuro sustentável para uma dada empresa. A solução proposta é uma prova de conceito que vai servir como base para um produto futuro que vai ser desenvolvido pela Ubiwhere.

Palavras-Chave

Digital Twin, Supply Chain, Sustainability, Industry 4.0

This page is intentionally left blank

Table of Contents

This page is intentionally left blank

Acronyms

AI Artificial Intelligence

ACID Atomicity, Consistency, Isolation, Durability

API Application Programming Interface

CSS Cascading Style Sheets

DT Digital Twin

GE General Electric

HTML HyperText Markup Language

HTTP Hypertext Transfer Protocol

IoT Internet of Things

JSON JavaScript Object Notation

MVT Model-View-Template

ORM Object Relational Mapper

PoC Proof of Concept

PHP Hypertext Preprocessor

REST Representational State Transfer

RDBMS Relational Database Management System

SDTC Swedish Digital Twin Consortium

SQL Structured Query Language

TBL Triple Bottom Line

XML Extensible Markup Language

YAML Yet Another Markup Language

This page is intentionally left blank

List of Figures

This page is intentionally left blank

List of Tables

This page is intentionally left blank

Chapter 1
Introduction

The following report documents the work that was performed during the second year of the master's degree in Informatics Engineering specifically in the branch of Software Engineering. The internship was supervised by Engineer André Duarte of Ubiwhere and by Professor Pedro Furtado. The internship is taking place at Ubiwhere, which is a company that focuses on R&I for Smart Cities, Telecommunications, and the Internet of Things. The first part of the internship was entirely remote, and the second part was hybrid.

The internship goal is to develop a Digital Twin system PoC based on a supply chain with special focus on sustainability of all processes involved.

1.1 Context and Motivation

This work focuses on digitalization of a supply chain and the processes associated with.

This internship has a duration of one year and it is being performed at Ubiwhere. which was founded in 2007 and it has offices in Aveiro and Coimbra.

In today's world, with the digital era having arrived in full flow, every company is starting to adopt digital techniques with the intent of improving the adaptability, response time, and performance of their company. Digital Twin is a virtual replication of a certain physical model or system that absorbs all physical processes from the real system.

The proof of concept developed during this internship serves as a baseline for a new line of product at Ubiwhere. Hence, it was developed with the mindset that different companies have given Ubiwhere along the years. In this sense, and to better improve the understanding of the Proof of Concept (PoC) a sample use case is implemented and demonstrated. A PoC is evidence obtained from an initial project, which is implemented to show that a product is feasible. For these reasons, no client was involved in drawing the requirements since Ubiwhere has acted as the sole stakeholder of the project.

When talking about Digital Twins it is vital to mention Industry 4.0 which is defined as the digitization of industrial processes, which incorporates the digitization of data as well as physical attributes (Abideen et al., 2021). It is a tool that enhances the technological maturity level of any organizational system that allows for the adoption of digitalization, integration, and automation in the production and supply chain network (Abideen et al., 2021).

It is vital to mention that the Digital Twin will be based on a supply chain. A supply chain is a system inside organizations that represents all stages to get the product or service from its original form to the customer. The main processes inside the supply chain are sourcing of the supplies, inventory management of the supplies and products, the production phase of those supplies that will turn into the final products and transportation to the customers.

The Digital Twin serves the purpose of digitizing the supply chain because the main feature of the Digital Twin is the absorption of physical processes, in this case it will absorb the supply chain stages. By digitizing all the supply chain processes, a company will be able to run simulations, see what processes need to be improved, and what other technologies can be implemented to enhance the performance of the company.

This internship intends to measure sustainability using the Digital Twin, which promotes the development of the new system with the goal of gaining environmental awareness in terms of the products and supplies that are used in the physical model. In practice, every time a product or supply leave the system, it will have an associated sustainability measure that shows if that product is sustainable in future iterations of the system. This functionality creates an opportunity to reduce the environmental footprint by changing the supplies or the way that the products are manufactured presently.

By joining these two different concepts together, it will be possible to implement a Digital Twin system which can have a huge impact in many companies that use a supply chain and want to promote a sustainable future.

1.2 Objectives

The main objective for this internship is the creation of a Digital Twin that will absorb all the physical processes, entities, and means of communication from the physical model of a supply chain. The platform will follow the stages of a supply chain, starting from inventory management to production. This Digital Twin will serve as a PoC for the company to develop a product of their own, the whole system will be brand new as the requirements will be defined by the intern.

By having this platform implemented, it will be possible to run simulations to optimize values inside the physical model and test new solutions before using them on the real model.

It will also be possible to measure the sustainability level inside each module, for example if a car is produced, it is possible to check the environmental footprint that the production of the car left behind. By having access to this information, it is possible to use it to reduce the environmental cost associated with the production of the car.

List of objectives:

- Implement the backend software to create the digital representation of the supply chain;
- Create an architecture for the Digital Twin;
- Functionality to measure sustainability inside the Digital Twin;

1.3 Document Structure

The document is structured as follows:

- Chapter 2 reviews topics related to this work. The topics reviewed include Digital Twins, Internet of Things, Artificial Intelligence, Virtual and Augmented Reality, Cloud Computing, APIs and Open Standards, Supply Chain and Digital Twins applied to sustainability
- Chapter 3 discusses the methodology and planning followed in the internship. It also includes success criteria, planning the two semesters, and risk management.
- Chapter 4 discusses application requirements, where both the scope, stakeholders, functional requirements, use cases and quality attributes are presented.
- Chapter 5 introduces the architecture of the solution, and the technologies used for the development.
- Chapter 6 presents the implementation stage of the thesis, where the logic behind the system was developed and displays all the stages of the developed Digital Twin.
- Chapter 7 presents the testing phase, which shows what tests were used for this stage along with the results
- Chapter 8 shows how the developed solution can be applied to other areas.
- Chapter 9 gives a summary of the experience that the intern had, as well as some hints about the future work of the developed solution.
- Appendix A shows the data models for the Digital Twin
- Appendix B displays the tables with the endpoints developed.
- Appendix C presents all the unit tests that were performed.

Chapter 2 State of the Art

This Chapter explores all the associated themes related to Digital Twins to create a base of knowledge regarding the context of this thesis. This Chapter is divided into five sections.

The first section introduces the concept of Digital Twin, its use for sustainability and provides examples of Digital Twins.

The second part refers to all the related technology that is needed to implement a functional Digital Twin.

The third part discusses how supply chains benefit from having a Digital Twin.

The fourth section discusses the benefits of the usage of Digital Twins regarding sustainability.

The fifth section will provide an overview of the software's developed concerning Digital Twins.

The final section summarizes the important aspects of the Digital Twin technology and details what main components will be used during the other stages of the project.

2.1 Digital Twins

A Digital Twin is a virtual model that reflects a physical model in a precise way, for example, a car production system. It includes the properties, condition, and behavior of the real-life object through models and data. The Digital Twin is developed alongside its physical twin and remains its virtual counterpart across the entire product lifecycle (Haag & Anderl, 2018). There are three main components in the DT system, the physical space, the digital space, and the information processing layer that connects two spaces (Zheng et al., 2019).

Sensors are used in the physical model to store and send the data to the Digital Twin.

A Digital Twin possesses the following traits: being a virtual model of a real thing; simulating both the physical state and behaviour of the real asset; being unique, associated with a single, specific instance of the thing; being connected to the real thing, updating itself in response to a known change to the physical model state, condition, or context; providing value through visualization, analysis, prediction, or optimization (Dohrmann et al., 2019).

One interesting aspect of the digital twin is that it can exist before the real thing is created and it can also exist after the real thing has reached the end of its life. A single thing can have more than one twin, with different users and use cases, such as what-if scenario planning or predicting the behaviour of the thing under future operating conditions (Dohrmann et al., 2019).

Digital Twins can add value to a product in many ways and these ways fall into the following categories: descriptive value, predictive value, analytic value, and diagnostic value (Dohrmann et al., 2019)

Descriptive Value - The ability to immediately visualize the status of an asset via its Digital Twin is valuable when those assets are remote or dangerous. Digital twins make information more accessible and easier to interpret from a distance.

Analytical Value - Digital twins that incorporate simulation technologies can provide data that is impossible to measure directly on the physical object. This can be used as a troubleshooting tool for existing models and can help to optimize the performance of subsequent product generations.

Diagnostic Value - They can include diagnostic systems that use measured or derived data to suggest the most probable root causes of specific states or behaviors. These systems can be implemented in the form of explicit rules on company know-how, or they may leverage analytics and machine learning approaches to derive relationships based on historical data.

Predictive Value - The likely future state of the physical model can be predicted using a Digital Twin model. The most sophisticated Digital Twins do more than merely predict the issue that may occur; they also propose the corresponding solution.

There have been several reported benefits from early adopters of Digital Twins according to (Dohrmann et al., 2019), the advantages discussed are data-driven decision making and collaboration, streamlined business processes, and new business models.

Figure 1: Digital Twin

In **Figure 1**, there is an example of a Digital Twin, where the physical and the virtual models of the same object are displayed.

Digital Twins Examples

The concept of Digital Twin is widespread in several working areas, from healthcare to manufacturing, smart cities, and others. Three examples will be presented in this section. The first one refers to the healthcare area, where the Swedish Digital Twin Consortium (SDTC) has the goal of developing a strategy for personalized medicine. The SDTC strategy is involved around: constructing unlimited copies of network models of all molecular, phenotypic, and environmental factors relevant to disease mechanisms in individual patients; computationally treating those digital twins with thousands of drugs to identify the best performing drug and treating the patient with this drug (Björnsson et al., 2019).

Moving onto the next example, Siemens offers various software solutions individually for each company: from real-time data analysis to a complex Digital Twin concept that optimizes the entire comprehensive production process. Maserati is using Siemens software to help support their product development as well as production process from product design to production planning, engineering, production execution, and services (I͡Uzhno-Uralʹskiĭ, 2020). It was also important for the development of the Maserati Ghibli vehicle: with the digital copy, the company was able to create a virtual twin in parallel to the physical development of the car. In the development process, the Maserati developers used data from the real and the virtual model at the same time. By incorporating the Digital Twin in their development process, they were able to cut the costs and the time required to develop by 30% (Digitalization in the Automotive Industry | Industry | Siemens Global, n.d). Another important aspect to mention is that the period between idea to market readiness decreased by sixteen months.

The last example illustrates how SAP's digital twins' technology impacts the renewable energy field. Traditional condition monitoring solutions for wind power systems are developed to report the operational state to understand changes that may occur over time. The main goal behind the monitoring is to prevent damage to the system and to ensure the efficient operation of the asset. SAP's Digital Twin for Wind Power monitoring enables operators to implement adaptive control strategies as well as improved predictive maintenance tactics based on the physical condition of the system at any time, using a digital representation of the real asset. Already implemented in a pilot project with Arctic Wind, this wind turbine digital twin solution supports maintenance operations and structural capability utilization (Erikstad & Ove, 2017).

2.2 Enabling Technologies

The development of a Digital Twin is possible because of the advances made inside Industry 4.0. The enabling technologies for the development of a Digital Twin are Internet of Things, Artificial Intelligence, Augment and Virtual Reality, Cloud Computing and APIs.

2.2.1 Internet of Things

The Internet of Things refers to all objects, often fitted with pervasive knowledge, that are connected to a network. A wide distribution network of communicative devices is generated by combining many objects for interaction through embedded systems, enabling contact between human devices and devices.

New standards of visibility and adaptability could be allowed to implement IoT, and efficiency could be increased in different fields, from smart homes to supply chains. In a supply chain, data obtained from various sensors in different locations can be processed and eventually informed by early alerts to human operators about any possible issues.

The number of IoT devices recorded year on year shows the considerable growth of this technology. In 2018 the number was over 17 billion (K. L. Lueth, 2018). By the year 2025, (Statista, n.d.) predicts that there will be over 75 billion devices with the industry predicted to be worth over $5 trillion.

The proliferation of IoT devices is universally beneficial, impacting the core of daily life, the communication sector, healthcare, building and transport, smart cities, and manufacturing (Fuller et al., 2020).

IoT play an important role in Digital Twin application enabling sensed data collection through smart devices for more precise and quick insights about the physical model status. Through wireless communication like Zigbee, Wi-Fi, and Bluetooth, IoT technologies provide Ad-hoc, mobile, safe, and self-configuring networks (Souza et al., 2019). Digital twins can often optimize an IoT deployment for maximum efficiency, as well as help designers figure out where things should go or how they operate before they are physically deployed (Shaw & Fruhlinger, 2019).

The IoT sensors come in all shapes and sizes. They are important for the Digital Twins idea as they provide the data the Digital Twin will use to process the information and access the actual state of the physical twin.

IoT developers and researchers use the data to create new schemas and logic and test it upon the Digital Twin. Once vetted, the working code is updated into the IoT device through over the air updates (Dave, 2020).

2.2.2 Artificial Intelligence

Artificial Intelligence is a branch of science and technology that creates intelligent machines and computer programs to perform various tasks which requires human intelligence (Ting et al., 2021).

AI is the science and engineering of making intelligent machines, especially intelligent computer programs. It is related to the similar task of using computers to understand human intelligence, but AI does not have to confine itself to methods that are biologically observable (Mccarthy, 2007). It is a system that mimics human actions and what humans can do. AI uses data like big data with the goal of completing a given task with excellent performance.

A subsection of AI, machine learning is the creation of algorithms that can give the computer the ability to learn and act for the user without being directly programmed to do so (Fuller et al., 2020).

There are three forms of performing machine learning: supervised learning, unsupervised learning, and reinforcement learning.

Supervised learning - The algorithms use large amounts of labelled data to analyse and learn. The algorithm is tasked with learning and analysing the labelled data to identify a given task correctly; image classification is one example (Fuller et al., 2020).

Unsupervised learning - Unsupervised learning algorithms learn using its own methods in categorizing and highlighting patterns within data instead of relying on user feedback (Fuller et al., 2020)

Reinforcement learning - Reinforcement learning is the closest machine learning type to how humans learn. The algorithm or agent used learns by interacting with its environment and getting a positive or negative reward.

Great improvements regarding usability and computation power transformed how companies extract information from big and complex sets of data. It incorporates components from both computer science and statistics and can be represented as a computer system that automatically enhances its efficiency over experience.

Machine learning is now being applied progressively in several sectors, including healthcare, education, engineering, marketing, customer service, and logistics (Moshood et al., 2021). Machine learning frameworks are enabling the development of systems that can make decisions autonomously as well as predictions about future conditions based on historical and real-time data (Dohrmann et al., 2019).

Supervised learning can be particularly useful when wanting to embed existing human knowledge in the Digital Twin, as it is based on modelling pairs of input data and labels. The result of training a supervised learning model is that the model can then infer labels for new input data, entirely automatically.

2.2.3 Augmented and Virtual Reality

In terms of increasing user experience, both augmented and virtual reality are becoming increasingly popular tools. Virtual technology is a technology that emulates the real world by creating a virtual world and how the user virtually experiences it.

This virtual world could be anything from creating a real-world high fidelity to the simulation of a particular portion of the user experience. In contrast, augmented reality, rather than creating a whole new virtual world, adds a layer of information to the real world. In a Digital twin context, both augmented and virtual reality can be useful tools to view and inspect the Digital Twins either on a screen (2D) or in a physical space (3D).

The technologies such as IoT, Cloud Computing, APIs, and machine learning all provide and process the necessary data and infrastructure to create and visualize Digital Twins in either augmented or virtual reality (Moshood et al., 2021).

To date, most Digital Twins have been rendered in two-dimensional space, as the conventional computing norms of today limit us to display on monitors, laptops, and other screens. But increasingly, augmented reality is enabling us to display digital content in 3D. In addition, mixed reality allows us to interact with digital content in our existing physical environment. And virtual reality allows us to create entirely new environments to render Digital Twins in a highly immersive way, creating the richest consumption of interaction with the information (Dohrmann et al., 2019).

Augmented, mixed, and virtual reality are the tools for visualizing Digital Twins and making them real to the user.

2.2.4 Cloud Computing

Cloud computing is a paradigm for allowing access to computing services from a public network. It is also the on-demand availability of computer system resources, especially data storage and computing power, without direct active management by the user. The main benefit of using a cloud is that there is no need for the customer to purchase and own costly hardware or own any computing space. The customer only pays for the cloud services that he uses, with the option to scale up or always scale down their desired services (Moshood et al., 2021).

Cloud Services are divided into three different categories:

- **Information as a Service (IaaS) -** Information as a Service is an emerging cloud business model in which a company shares or sells relevant information to another company or individuals to perform their business (*Information as a Service | Times of Cloud*, n.d.).

- **Platform as a Service (PaaS) -** Platform as a Service is a complete development and deployment in the cloud, with resources that enable you to deliver everything from simple cloud-based apps to complex, cloud-enabled enterprise applications. The resources are purchased from a cloud service provider on a pay-as-you-go basis and access over a secure Internet connection (Microsoft, 2020).

- **Software as a Service (SaaS) -** Software as a Service allows users to connect and use cloud-based apps over the Internet. Common examples are email, calendaring, and office tools. SaaS provides a complete software solution that is purchased on a pay-as-you-go from a cloud service provider. The use of the app is rented for an organization, and the users from the organization connect it over the Internet, usually from a web browser (Microsoft, 2021).

Developing, maintaining, and using Digital Twins is a compute and storage-intensive endeavour. Thanks to the continually falling cost of processing power and storage, large data center networks with access provided via SaaS solutions now enable companies to acquire exactly the computer resources they need, when they need them, while keeping costs under control (Dohrmann et al., 2019).

2.2.5 APIs

An Application Programming Interface (API) enables companies to open their application's data and functionality to external third-party developers, business partners, and internal departments within their companies. This allows services and products to communicate with each other and leverage each other's data and functionality through a documented interface.

Developers do not need to know how an API is implemented, they simply use the interface to communicate with other products and services. API use has surged over the past decade, to the point that many of the most popular web applications today would not be possible without APIs (*What Is an Application Programming Interface (API) | IBM*, n.d.).

A simple example of this is the Google Maps API, which can be easily combined with third-party content to view local points of interest on a Google Map. The third-party would not have to schedule a completely new map in this scenario, which saves both time and money. API's help to transfer data easily from clouds, computers, and other systems, as many modern enterprises deal with cloud computing (Moshood et al., 2021).

The physical model uses the API to communicate with the Digital Twin and vice-versa. So, it is important to ensure that the API is reliable and performs well to guarantee the success of the Digital Twin (Scheibmeir & Malaiya, 2019).

2.3 Supply Chain

At its highest level, a supply chain consists of two basic, integrated processes: the Production Planning and Inventory Control Process, and the Distribution and Logistics Process. These processes provide the basic framework for the conversion and movement of raw materials into final products (Beamon, 1998).

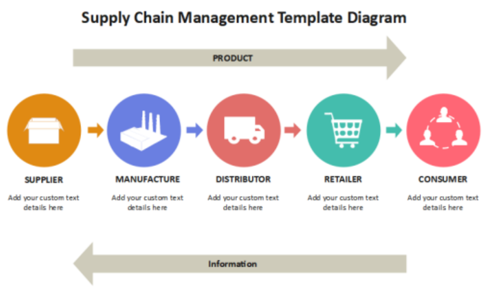

Supply Chain Management Template Diagram
PRODUCT
SUPPLIER
Add your custom text details here
MANUFACTURE
Add your custom text details here
DISTRIBUTOR
Add your custom text details here
RETAILER
Add your custom text details here
CONSUMER
Add your custom text details here
Information

Figure 2: Supply Chain Example

In **Figure 2** a typical supply chain is displayed along with all the stages associated with it.

The Production Planning and Inventory Control Process encompasses the manufacturing and storage sub-processes, and their interface(s). Production planning describes the design and management of the entire manufacturing process. Inventory management describes the design and management of the storage policies and procedures for raw materials, work in process inventories, and final products. The Distribution and Logistics Process determines how products are retrieved and transported from the warehouse to retailers (Beamon, 1998).

Any company in which the supply chain has a critical function can quickly become a concern since lackluster accountability and exposure of the supply chain can throw clouds on outdated processes that may otherwise be rectified (Moshood et al., 2021).

Chapter 2
State of the Art

Nowadays one of the most important aspects of the supply chain is visibility, which means the ability of the focal company to access the significant, reliable, and meaningful information owned by its Supply Chain partners, neglecting additional information that is not useful in the decision-making process (Caridi et al., 2014).

Ensuring proper supply chain visibility within a company can have a myriad of positive effects improving forecasting, planning, scheduling and execution of orders to only name a few. Organizations benefit from proper supply chain visibility by obtaining the ability to reconfigure the supply chain quickly and efficiently, a skill that is becoming increasingly important when it comes to generating competitive advantage in rapidly evolving business environments (Blomkvist & Loenbom, 2020).

There are four core processes inside the visibility of the supply chain which are visibility for sensing, learning, coordinating, and integrating (Wei & Wang, 2010).

- **Visibility for sensing -** This metric indicates the extent of which the organization can quickly get real-time information regarding internal and external processes and react to a changing business environment
- **Visibility for learning -** This indicates the extent of which the organization can gather and learn from new information and knowledge from both internal and external processes.
- **Visibility for coordinating -** This represents how adept the organization is at coordinating different areas of their supply chain making decisions with overarching consequences for many different actors within the system.
- **Visibility for integrating -** This represents how adaptable the organization is when it comes to adopting and integrating new methods and technologies to develop a strategic business advantage.

To increase the visibility of the supply chain, it soon becomes evident that providing functioning Digital Twins integrated within a company's logistical supply chain will prove very useful (Moshood et al., 2021).

A Digital Twin will dramatically improve the metrics of visibility for sensing (quickly gathering knowledge about internal and external processes) and visibility for learning (how quickly the information gathered can be processed), and this aggregated network of information can then be used to organize in a better way both internal and external processes (Moshood et al., 2021).

A supply chain using Digital Twin technology can be used for:

- Understanding supply chain dynamics and behaviour
- Bottleneck discovery
- Testing supply chain design changes and development
- Monitoring risk and testing contingencies
- Transportation planning
- Inventory optimization

- Cash to serve and cost to serve analysis
- Forecasting and testing operations over the coming days and weeks

2.4 Sustainability

Sustainability focuses on meeting the needs of the present without compromising the ability of future generations to meet their needs. The concept of sustainability is associated with three words: economic, environmental, and social. Sustainability encourages businesses to frame decisions in terms of environmental, social, and human impact for the long term (Grant, 2016).

Nowadays measuring sustainability is a very important aspect regarding the future of a given company, using sustainability as a metric generally means an expansion of the traditional business reporting framework to consider social and environmental performance in addition to financial performance.

Importantly for sustainability, twinning can also help to predict water network flow, balance energy grids, and build resilience in the face of the impacts of climate change. To maximize the benefits, digital twins need to be elevated from focusing on individual assets only, to encompass whole processes or even entire enterprises or cities (Brink, n.d.)

2.4.1 Triple Bottom Line

The triple bottom line (TBL) concept hopes to transform the current financial accounting-focused business system to take on a more comprehensive approach in measuring impact and success (A Simple Explanation of the Triple Bottom Line | University of Wisconsin, 2019). The triple bottom line approach has the goal of helping organizations to move towards a regenerative and more sustainable future.

TBL provides the necessary tools for the organizations to measure, set goals, improve, and eventually evolve toward more sustainable systems and models (A Simple Explanation of the Triple Bottom Line | University of Wisconsin, 2019). It is important to note that TBL has three categories that make up this framework: people, planet, and prosperity.

There is no universal standard method for calculating the TBL. This can be viewed as something positive because each organization will provide the data that suits the framework in the best way. Different companies with distinct goals will provide different data according to their main objectives and that is something very positive for the future of the organization (Slaper, n.d.).

Economic Measures

Economic measures are variables that deal with the bottom line and the flow of money. It could look at income or expenditures, taxes, business climate factors, employment, and business diversity factors. Some examples include:

- Personal Income
- Cost of underemployment
- Job Growth
- Revenue by sector contributing to gross state product

Environmental Measures

Environmental variables should represent measurements of natural resources and reflect potential influences on their viability. It could incorporate air and water quality, energy consumption, natural resources, solid and toxic waste, and land/use cover. Some examples include:

- Sulphur dioxide concentration
- The concentration of nitrogen oxides
- Selected priority pollutants
- Fossil fuel consumption
- Solid waste management
- Electricity consumption

Social Measures

Social variables refer to social dimensions of a community or region and could include measurements of education, equity and access to social resources, health and well-being, quality of life, and social life. Some examples include:

- Unemployment rate
- Female labour force participation rate
- Relative poverty
- Violent crimes per capita
- Health-adjusted life expectancy

2.4.2 Sustainability Examples with Digital Twins

Destination Earth

Destination Earth aims to develop a high precision digital model of the Earth to model, monitor and simulate natural phenomena and related human activities. As part of the European Commission's Green Deal and the Digital Strategy, Destination Earth (DestinE) will contribute to achieving the objectives of the twin transition, green and digital (European Commission, 2021).

DestinE will unlock the potential of digital modeling of the Earth system. It will focus on the effects of climate change, water, and marine environments, polar areas, cryosphere, biodiversity, or extreme weather events, together with possible adaptation and mitigation strategies. It will help to predict major environmental degradation and disasters with unprecedented fidelity and reliability (European Commission, 2021).

Kongsberg Digital

Kongsberg Digital, the digital branch of global technology company Kongsberg, recently joined in a strategic partnership with FutureOn, a global energy software company, to bring together their digital twin platforms for energy projects (Carey, 2021).

For oil and gas enterprises, Digital Twins are vital to unlock information that will improve environmental performance and productivity of assets by incorporating sustainability elements when optimizing production, minimizing energy use, and reducing emissions. This delivers accountable data for environmental, social, and governance (ESG) financial analysis and allows communication of a credible plan toward meeting net-zero targets and timelines (Carey, 2021).

Integrated Environmental Solutions

British company Integrated Environmental Solutions has developed the Intelligent Communities Lifecycle (Hurtado, 2020). This environmental digital twin technology reduces carbon emissions of buildings and cities worldwide.

It connects the physical and real world with the intent to enable energy-efficient design and continuous optimization by using real world operational data, which keeps Digital Twin accurate. This way, buildings can detect anomalies that would not be detected previously (Hurtado, 2020).

Digitalization provides an increasing number of technologies that offer a different perspective on sustainable production. This reflects how the Digital Twins can positively impact sustainability.

2.5 Digital Twin Software's

Inside this section the existing Digital Twins software's developed by Microsoft, Siemens and General Electric are presented.

2.5.1 Azure Digital Twins

Azure Digital Twins is an Internet of Things platform that enables the user to create a digital representation of real-world things, places, business processes, and people. Gain insights that help you drive better products, optimize operations and costs, and create breakthrough experiences.

It is very simple to create digital representations of connected environments with an open modelling language. Model buildings, factories, farms, energy networks, railways, stadiums, and even entire cities. It is possible to build dynamic business logic and data processing on a robust event system. Another feature is the possibility of using powerful query APIs and integrating them with Azure data, analytics, and AI Services.

Using Azure Digital Twins, it is possible to model any physical environment that is important to a certain business. Furthermore, it connects inputs from IoT devices that are using Azure IoT Hub or from any business system to establish a single live integration layer that delivers insights from across the entire environment (*Digital Twins – Modeling and Simulations | Microsoft Azure*, n.d.).

2.5.2 Siemens MindSphere

MindSphere is a cloud based, open IoT operating system that enables the user to connect machines, products, plants, and systems. With MindSphere, you can complete the closed-loop Digital Twin, permanently connecting the physical assets in the factories to their digital copies.

This connection facilitates an open exchange of information that gives you advantages throughout the lifecycle- from product design, production planning, and engineering to commissioning, operation, servicing, and the modernization of systems and plants. Using MindSphere also makes it easy to develop, test, and operate sophisticated, targeted IoT applications- both for internal use and selling to customers.

To help manufacturers streamline the adoption of MindSphere, Siemens has built a collection of packaged solutions for customers in different phases of their digitalization journey.

The Digitalize and Transform packaged solution is primarily designed to help manufacturers that are already realizing many of the benefits of IoT to take the next step in their digital transformation journey by building powerful, targeted applications. These applications are key to developing new business models and products, such as asset-as-a-service offerings (*Build Powerful, Targeted IoT Applications with a Packaged Solution*, 2019).

2.5.3 GE Digital Twin

GE has created the most advanced and functional Digital Twin that integrates analytic models for components of the power plant that measure asset health, wear, and performance with business objectives.

The Digital Twin runs on an industrial platform, Predix™, designed to ingest massive volumes of machine sensor data, to manage and execute analytic models, to run a high-speed business rules engine, and to manage industrial data at scale. Further, this environment is integrated with business applications designed to allow plant executives, plant managers, and Users to interact with the Digital Twin in real-time.

Business applications tied to the Digital Twin provide a window of interaction to act on insights, to manage the power plant and generation fleet functions to a greater level of control, and to be able to react to changing market, fuel price, and weather conditions in rapid fashion. These business applications are designed to increase asset performance, enhance operations, and improve energy trading decisions to create additional revenue and cost reduction opportunities (Power Digital Solutions, 2016).

2.6 Final Considerations

Through this state of the art, it is possible to see the importance of the Digital Twin technology in today's world not only because of the positive impact that it can have inside companies by reducing costs, increasing performance and efficiency.

It was important to identify the several technologies that enable the Digital Twin development and well-functioning. Thanks to the rapid growth of the Industry 4.0, the technologies that enable the Digital Twin are much more capable of feeding real time data to the Digital Twin in large amounts, using artificial intelligence to process those large chunks of data and propose solutions to the potential problems that the company that is using the Digital Twin has.

The supply chain will benefit from the implementation of the Digital Twin technology because the visibility of the supply chain will increase dramatically the metrics of visibility for sensing and learning. Besides that, it can be used for inventory optimization, transportation planning, testing supply chain design changes and development and monitoring risk and testing contingencies.

Regarding sustainability the advantages of using this technology are immense, by looking at the examples given inside this state of the art it is possible to see that it will help to predict major environmental degradation and disasters with unprecedented fidelity and reliability; for oil companies the Digital Twins are vital to unlock information that will improve environmental performance and productivity of assets by incorporating sustainability elements when optimizing production, minimizing energy use, and reducing emissions.

For this internship, a Digital Twin solution will be developed and based on a physical supply chain. The processes inside this supply chain will be integrated inside the Digital Twin, to mimic the physical system. The API technology that is mentioned along this Chapter will be used during the development of the system. There will be a special focus on the sustainability aspect of the Digital Twin because one of the reasons behind the implementation of this technology is to reduce the environmental footprint behind the supply chain and gain awareness of the amount of pollution the processes inside the supply chain are causing.

Chapter 3
Planning and Methodology

In this Chapter, the methodology for the project is presented along with the success criteria defined for the internship. Besides that, the tools that were chosen are shown, as well as the planning for the first and second semester and the risks identified in this internship are displayed with the mitigation plan for each risk.

3.1 Success Criteria

The success of the internship is directly related to the completion of the goals that were defined at the beginning of the internship. The main goal behind the internship is the development of a Digital Twin capable of replicating a supply chain by absorbing all the physical processes from the supply chain. The other goal is to implement a functionality that calculates the level of sustainability for each absorbed process.

- The requirements and architecture proposed by the intern must be approved by Ubiwhere;
- The developed architecture and project must ensure all the quality attributes requirements defined in this document;
- The internship and all the predefined goals have to be completed within the stipulated time;

3.2 Process Management

An agile method was chosen to develop the project, in this case the SCRUM methodology. The agile methodology is a way to manage a project by breaking it up into several phases.

Instead of launching the product just at the end of development, the product is continuously developed in increments, which means they are functional without the product being finished. Requirements, plans, and results are evaluated continuously because they can be changed during the development of the project; this way teams possess a natural mechanism for responding to change quickly (Atlassian, 2020).

The development process is iterative, which means that the project is executed in short iterations. Furthermore, every iteration has its testing phase, which allows implementing regression testing every time new functions or logic are released. Additionally, when an iteration ends the new features that were developed are sent to the customer ready to be shipped (Thomas Hamilton, 2019).

It is important to mention that there is a product backlog that contains all the functionalities needed to be developed until the end of the project. Each sprint retrieves a set of functionalities from the backlog, and they must be developed until the end of the sprint. There are three important roles that must be defined inside Scrum. They are the Scrum Master which acts as the leader of the team, Product Owner which is responsible for connecting the business side to the technical aspect of the project, Dev Team responsible for developing the functionalities listed on the backlog.

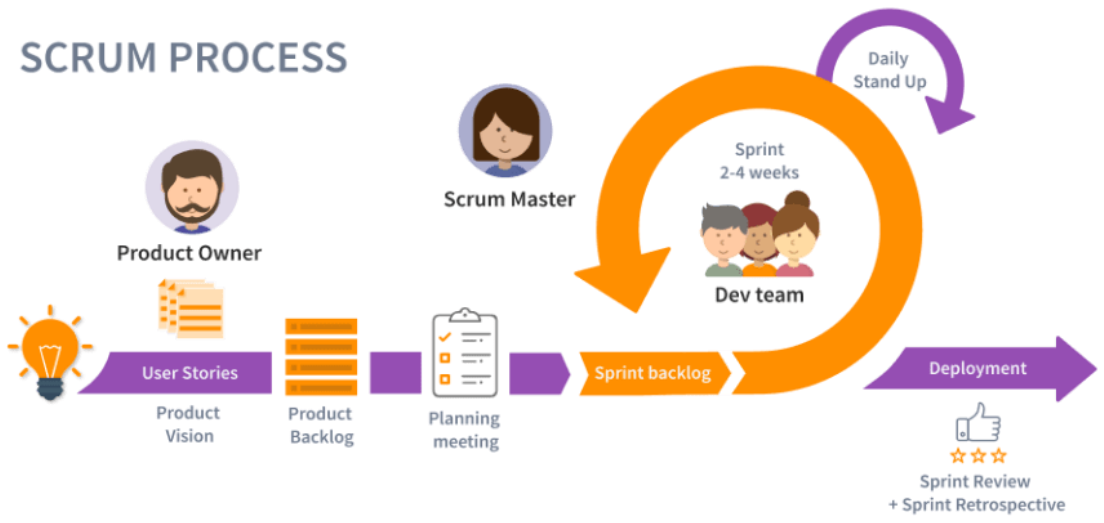
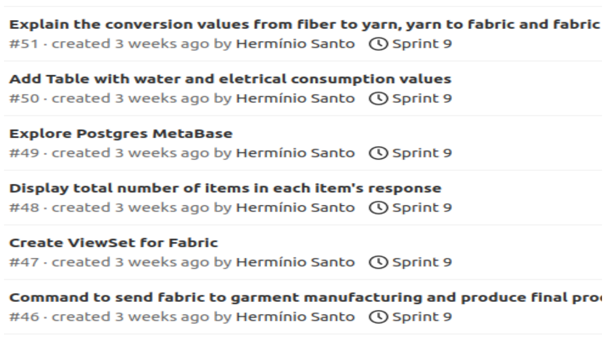


Figure 3: Scrum Process

In this specific case, the intern would have weekly meetings instead of the daily stand up with his supervisors, where weekly sprints were defined as well as goals to present the following week. The supervisors would give feedback regarding the work that had been done during the week and would set up the following goals for the upcoming sprint. The intern is the sole member of the development team. For this internship Use Cases were used instead of User Stories like is shown on **Figure 3**.

If there was a problem during the sprint, the intern could contact the supervisors with the intent of solving the issue that caused the delay.



Explain the conversion values from fiber to yarn, yarn to fabric and fabric
#51 · created 3 weeks ago by Hermínio Santo Sprint 9

Add Table with water and eletrical consumption values
#50 · created 3 weeks ago by Hermínio Santo Sprint 9

Explore Postgres MetaBase
#49 · created 3 weeks ago by Hermínio Santo Sprint 9

Display total number of items in each item's response
#48 · created 3 weeks ago by Hermínio Santo Sprint 9

Create ViewSet for Fabric
#47 · created 3 weeks ago by Hermínio Santo Sprint 9

Command to send fabric to garment manufacturing and produce final pro
#46 · created 3 weeks ago by Hermínio Santo Sprint 9

Figure 4: Sprint Planning

In **Figure 4**, one of the sprints for the internship is displayed, the spring planning was done inside the Gitlab platform where the intern defined the issues for each sprint with the approval of the supervisor.

3.3 Tools

During the internship the following tools were used to promote the development of the application:

- **Slack -** Slack is an instant messaging platform used as the main way to communicate inside Ubiwhere, where the workers have access to multiple channels where they can communicate with the other workers. Specifically, in this internship, the intern had a channel with his supervisors to facilitate communication between them.

- **Easy Redmine -** It is a high-end project management application. It is used by companies from around the world to handle projects, tasks, schedules, resource utilization, budgets, attendance monitoring, support management, and much more (*Easy Redmine vs Redmine | What Are the Differences?*, n.d.).

- **Google Meets -** Google is making enterprise-grade video conferencing available to everyone (Google, 2021). This tool is used during the weekly meetings between the intern and the Ubiwhere supervisors.

- **Google Calendar -** With Google Calendar, you can quickly schedule meetings and events and get reminders about upcoming activities, so you always know what's next (*What Can You Do with Calendar? - Google Workspace Learning Center*, n.d.). This is used to calendarize all the events related to Ubiwhere, from the weekly meetings to morning coffees.

- **GitLab -** Is an open-source Git-repository manager used in Ubiwhere with many features. Such features include Version control and repository management based on Git, Issue management, bug tracking and boards, Code Review functionality and Review Apps tool.

- **Visual Studio Code –** Is a source-code editor made by Microsoft for Windows, Linux and macOS. The main features are support for debugging, syntax highlighting, code completion, snippets, code refactoring and Git integration.

- **TeamGantt -** Is a cloud-based Gantt chart and project planning solution for small, midsize, and large enterprises. It was used for the planning of the first and second semester.

- **Miro –** It is an online collaborative whiteboard platform that enables distributed teams to work effectively together (Miro, 2020).

- **Postman –** Postman is an application used for API testing; it is a HTTP client that tests HTTP requests (Romero, 2021). It was used during the implementations stage to test the user’s requests.

3.4 Planning

In this section, the planning of what was done during the first and second semesters will be explained. What decisions were made and the reasoning behind every decision, all the stages that the internship has been through from start to finish are all documented in this section.

3.4.1 First Semester

The first semester was mainly focused on the area of researching about the subject Digital Twin, the applications where a Digital Twin is used, and which big companies have produced software embedded with this technology. All this research had the goal of writing a state of art about this subject, which can be found in Chapter 2.

After finishing the state of art, the next step was to start preparing the requirements needed to develop this project, beginning by stating the use cases that will be implemented later in the development stage; the requirements can be found in Chapter 4.

Along with the requirements, the development of the architecture was also a point of interest during this semester; the architecture can be found in Chapter 5.

During the first semester the following tasks were expected to be completed by the end of the semester:

- **State of the Art Research -** The first semester was mainly focused on the area of researching the subject Digital Twin, the applications where a Digital Twin is used, and which big companies have produced software embedded with this technology. All of this research had the goal of writing a state of the art about this subject, which can be found in Chapter 2;

- **Introduction and Context -** These tasks embedded the context in which the application for this internship will be developed, the main goals of the internship, and finally the structure of this report. All of this information can be found in Chapter 1;

- **Requirements Specification -** After finishing the state of art, the next step was to start preparing the requirements needed to develop this project: defining the scope, stakeholders, constraints, and lastly all the requirements which can be found in Chapter 4;

- **Proposed Architecture -** The definition of the architecture of the app that will be developed is presented. Along with that, the choice of technologies that will be used is also displayed. This can be found in Chapter 5;

- **Planning and Methodologies -** For this task, the process and tools used are described. Additionally, the planning is displayed, besides that, the risk assessment is also presented. This is shown in Chapter 3;

- **Intermediate Report -** During the first semester, the report has been written in parallel with the other tasks.

To illustrate how the first semester was planned and executed the following high-level Gantt diagram (**Figure 5**) is displayed so that it is easier to see how much time the intern spent on each task.

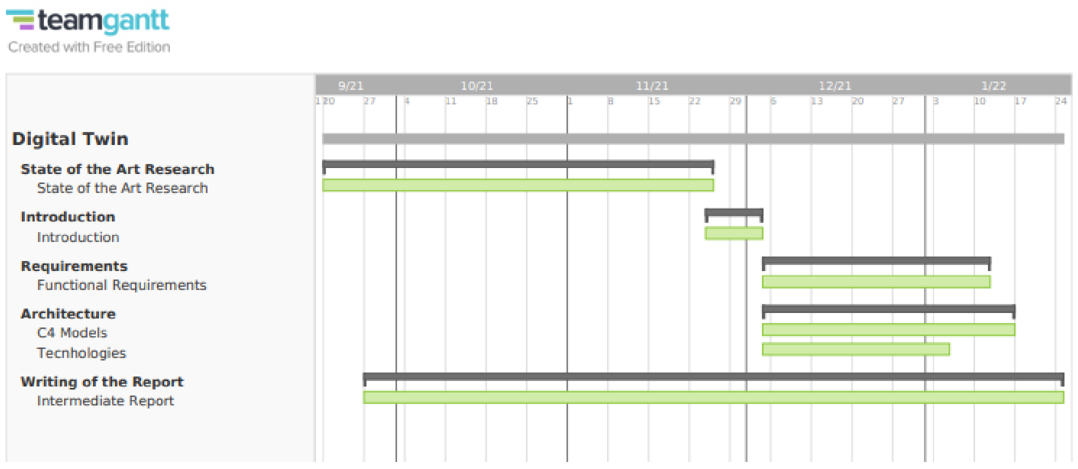


Figure 5: First Semester Planning

3.4.2 Second Semester

During the second semester the following tasks were expected to be completed by the end of the semester, the schedule is shown in **Figure 6**:

- **Intermediate Report Correction -** During the first weeks of the second semester, the report must be corrected according to the notes from the intermediate defense and the quality attributes of the project must be defined.

- **Case Study Research –** In order to instantiate the problem defined by this internship, the intern must research supply chain cases to apply to the project. Then after having a significant number of options, the intern must choose one to apply and define the data models that will be used during the implementation stage.

- **Implementation** – The intern will start by learning the technologies that will be used for the project. After that, the work environment for the project will be prepared to start coding. When the work environment is ready, the intern will start the development stage by implementing the requirements defined during the first semester. Firstly, the users application will be implemented, then the processes and items applications will be developed simultaneously. After finishing the development of the three applications the implementation stage is concluded.

- **Testing** – The quality attributes and the functional requirements defined previously will be tested during this stage. Unit Testing will be performed to test the endpoints functionality.

- **Live Demo** – At the end of the semester, a live demo will be prepared to show at the final presentation of this thesis.

- **Final Written Report** – During all the semester, the report will be continuously written.

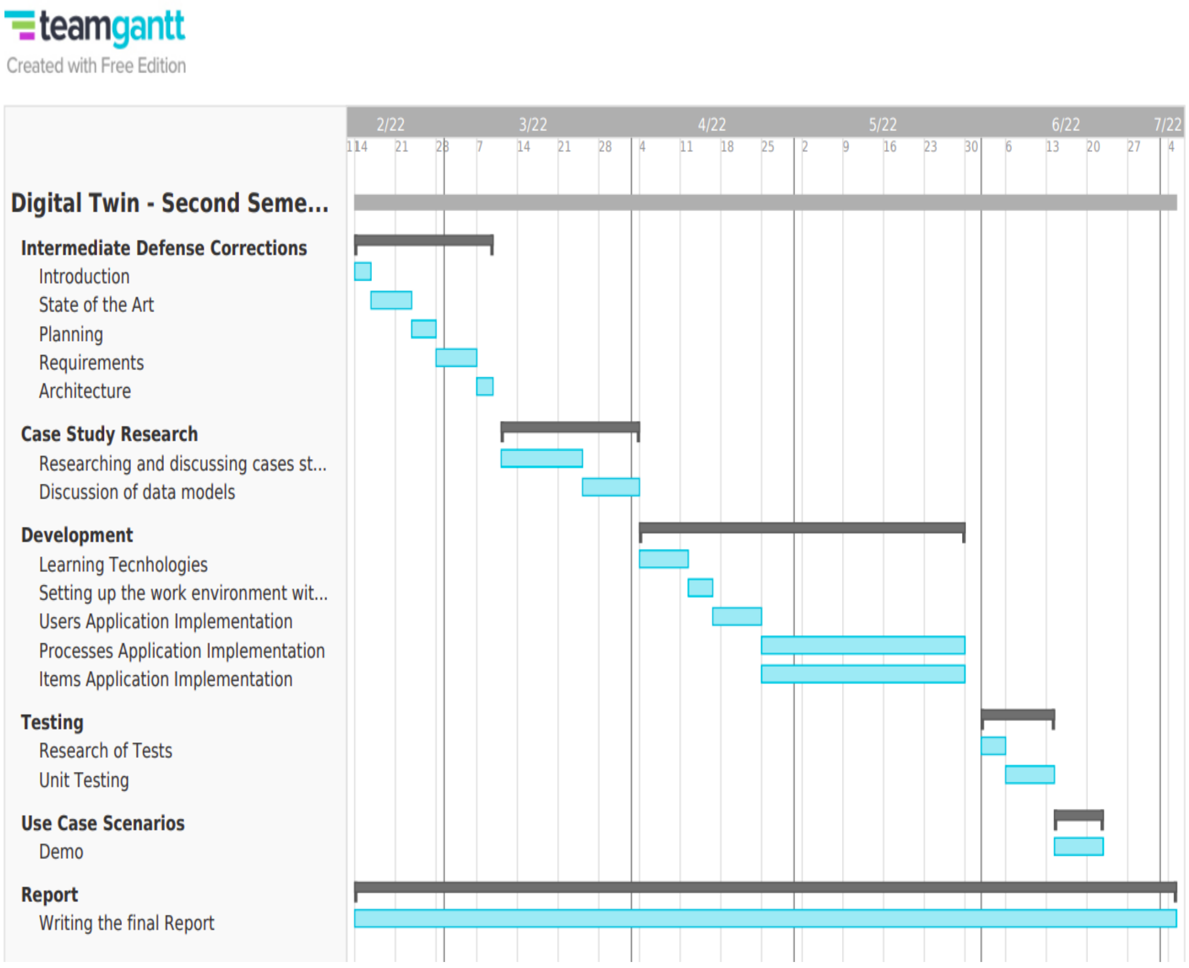


Figure 6: Second Semester Planning

3.5 Risk Management

Software risk management is a vital point during the development of specific software because the possible risks to the project are identified beforehand, which leads to some degree of prevention and mitigation when those risks materialize (Stodder, 2016).

It is also important to mention that this is a continuous process, so it evolves during the development of the software, to be updated with the latest changes of the project.

This risk management includes the identification and classification of technical, programmatic, and process risks, which becomes a part of a plan that links to a mitigation strategy (Stodder, 2016). If one or more risks turns into a threat, there is always a contingency plan to minimize the threat, that is the main reason behind the existence of risk management.

To classify a risk, the impact and probability can be measured. The impact refers to the consequences that would happen if the risk became real, the probability like the name refers to the odds of the risk happening.

3.5.1 Impact Classification

To measure the impact of a risk in a project, the risk management assessment scale that classifies the risks from Severe, Significant, Moderate, Minor to Minimal is used (The MITRE Corporation, 2014). Impact is measured through the effect that a risk has on the success criteria if the risk happens.

- **Severe -** If this risk occurs, it will have a huge impact on the result of the project, because the goal of the internship will not be reached;
- **Significant -** If this risk occurs, it will have a significant impact on the project, because to reach the goal of this internship, a lot of cost and effort will be needed to suppress this risk;
- **Moderate -** If this risk occurs, the end goal will still be reached but it will require some effort and cost to suppress the risk;
- **Minor -** If this risk occurs, the impact will be minimum, it will be suppressed with minimum effort and cost;
- **Minimal -** If this risk occurs, the impact will be invisible, so it will not be necessary for any additional effort and cost to suppress it.

3.5.2 Probability Classification

To measure the probability of a risk, the following scale will be used:

- **High -** The probability of a risk to appear on the project is very high;
- **Medium -** The probability of a risk to appear on the project is medium;
- **Low -** The probability of a risk appearing on the project is minimal.

3.5.3 Risk Analysis and Mitigation

The following tables show the risks that were identified, each one of the risks is classified using the scales mentioned in the subsections above. Each risk has a mitigation plan associated

Table 1: Risk 1 The project is too general and abstract to instantiate

ID	R1
Condition	The project is too general, and it is not possible to instantiate the problem defined for this internship
Consequence	The project cannot move forward because it is impossible for the intern to extract requirements and propose an architecture from the definition of the internship
Impact	Severe
Probability	Medium
Mitigation Plan	The intern will research case studies associated to the theme of the internship and choose one along with the company, in order to apply it into the project.

Table 2: Risk 2 supervisor's pandemic situation

ID	R2
Condition	Given the current pandemic situation, the intern supervisors may be infected
Consequence	The supervisors will not be able to attend the remote weekly meetings and the productivity of the intern may decrease
Impact	Minor
Probability	Medium
Mitigation Plan	The meetings between intern and supervisors will be online to minimize the risk of getting infected

Table 3: Risk 3 Scope of the project

ID	R3
Condition	The project scope too broad for the intern to handle
Consequence	The intern will not be able to meet the desired product
Impact	Severe
Probability	Low
Mitigation Plan	The intern will discuss with his supervisors to establish a scope that is agreeable for both parts, in order to complete the project by the end of the internship

Table 4: Risk 4 Intern's lack of experience

ID	R4
Condition	The intern does not have experience dealing with Digital Twins
Consequence	The project may take more time to develop than was anticipated
Impact	Moderate
Probability	High
Mitigation Plan	The intern will take some time to learn about Digital Twins, how they work, how they are implemented so that this risk is minimized

Table 5: Risk 5 Projected tasks take more time than expected

ID	R5
Condition	The project tasks may take more time to do than what was planned
Consequence	The project will take more time to develop than was anticipated
Impact	Moderate
Probability	High
Mitigation Plan	The intern will talk to his supervisors about the delays and whether to focus on other tasks that might be more important to the development of the product

Table 6: Risk 6 Intern lacks experience with the chosen technologies

ID	R6
Condition	The intern lacks experience in the technologies that will be used for the development of the project
Consequence	The project will take more time to develop if the intern is not familiar with the technology that is going to be used
Impact	Significant
Probability	Medium
Mitigation Plan	The intern will spend the appropriate time learning the designated technologies chosen for this internship

Table 7: Risk 7 intern's pandemic situation

ID	R7
Condition	Given the current pandemic situation, the intern may be infected
Consequence	The productivity of the intern will decrease and some of the defined goals will not be reached
Impact	Minor
Probability	Medium
Mitigation Plan	The intern will be remote during the majority of the internship to minimize the risk of getting infected

3.5.4 Materialized Risks

During the duration of the internship, the first risk **R1** occurred, which meant that the intern had to use the mitigation plan to reduce the effects of it. That risk delayed the development of the project because the intern had to research a case study to use for this project and the case study had to be one that made sense integrating into the internship.

The risk was surpassed successfully, as the intern managed to choose an appropriate case study for the Digital Twin and could finish all the objectives that were defined at the beginning of the internship.

Chapter 4 Requirements

This Chapter presents the requirements that were raised for the intended system. In this Chapter, the requirements of this internship will be presented. The scope, stakeholders and actors of the project are discussed first. Functional requirements are then presented as use cases. These raised requirements will be the basis for architecture and development. Along with the functional requirements, the quality attributes that the system will have are presented in this Chapter.

4.1 Scope

The scope of this internship revolves around a Digital Twin system that replicates a textile supply chain. The system will receive inputs, specifically cotton and that input will go through all the processes of the supply chain until it reaches its final form, which is a finished product.

The product will not be delivered to a direct client, as it was mentioned in the first Chapter, it is a PoC for Ubiwhere.

It was defined by the company that the inputs for the system should be generated inside the system and that the input for the Digital Twin is cotton, because it is one of the most used materials in the textile industry. Following the textile supply chain of cotton production, it is possible to see that it is transformed into yarn and then into fabric and finally into the finished product (Textile Learner, 2020). The Digital Twin will follow the stages from the textile supply chain that uses cotton as an input.

4.2 Stakeholders

The sole stakeholder of this project is Ubiwhere. The result of this work will create a relevant product line for Ubiwhere, and the company is providing constant feedback to the intern about the project.

4.3 Functional Requirements

A functional requirement is a component or the software system itself. For this part of the project, to better understand the functional requirements needed for this project, a flowchart was created to better display which requirements would be needed to develop the project. The flowchart represents all the textile supply chain stages, starting from the sourcing of the materials and ending when the product is finished (Martin, 2021).

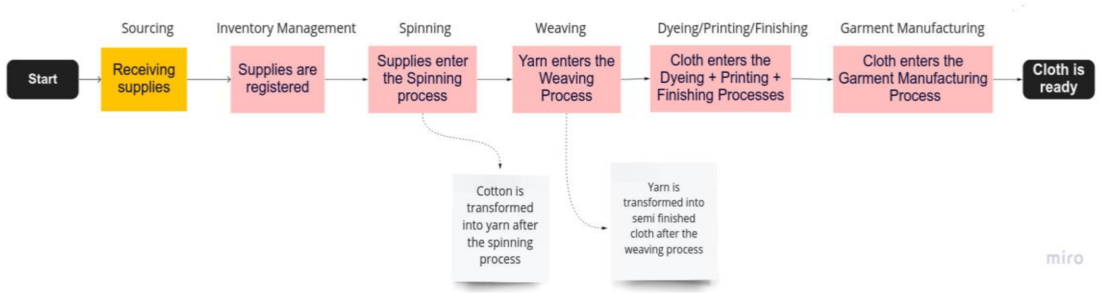

Sourcing
Inventory Management
Spinning
Weaving
Dyeing/Printing/Finishing
Garment Manufacturing
Start
Receiving supplies
Supplies are registered
Supplies enter the Spinning process
Yarn enters the Weaving Process
Cloth enters the Dyeing + Printing + Finishing Processes
Cloth enters the Garment Manufacturing Process
Cloth is ready
Cotton is transformed into yarn after the spinning process
Yarn is transformed into semi finished cloth after the weaving process
miro

Figure 7: Example flowchart from a Textile Supply Chain

Figure 7 represents all the active stages of a textile supply chain except the transportation and sale to the retailer's stage, every process inside the flowchart belongs to a functional module. Each module represents a supply chain stage, starting from sourcing the supplies and ending when the product is finished. The flowchart will serve as the basis for the system implementation and requirements. The supply considered for this textile supply chain is the cotton fiber.

Textile Supply Chain Behavior

- To start the whole system, the supply chain receives cotton from a given supplier, then the following supplies are registered;
- After being registered, the supplies are ready to enter into production which consists of six processes: spinning, weaving, dyeing, printing, finishing and garment manufacturing;
- The supply is transformed into yarn after passing the spinning process, it enters the weaving process where the yarn is transformed into fabric;
- Then, the fabric enters the dyeing process, followed by the printing and finishing processes and it is transformed into cloth;
- Finally, the product is almost finished and enters the garment manufacturing process where it is transformed into finished cloth.

To better understand the main processes behind the supply chain, the intern gathered functional modules. These modules are the main processes that guide all the operation from start to finish of the supply chain. The following modules were collected:

- **Sourcing:** Sourcing is the process of selecting all the necessary supplies based on the supply chain criteria.

- **Inventory Management:** Inventory management is the process where all the items inside the system are registered inside the application.

- **Production:** This production module encapsulates all the productions processes and transformations derived from **Figure 7** and it includes spinning, weaving, dyeing, printing, finishing and garment manufacturing.

4.3.1 Actors

In this section, the actors that are involved with the project are identified along with their goal, what they need from the system and how the actors will use the system.

Normal User

The Normal User will be able to read all the information inside each module of the Digital Twin and access the statistics from each component, but it will not be able to alter the data inside a given module.

Admin User

Admin user and will be responsible for adding, editing, and deleting supplies or products from a given module inside the system. It will also be responsible to check the sustainability levels of each module in the system. The admin will have access to all the data inside each module.

System

The system corresponds to the Digital Twin and will be responsible to receive and transfer data from one module to another.

4.3.2 Processes

In this section, the production processes inside the cotton supply chain will be presented, to give a better overview on how each process works.

Spinning

Spinning is a procedure of converting fiber materials into yarn (Textile Learner, 2020). On an initial phase, the fibers go through the blow room where the size of amount of cotton becomes smaller with the help of machinery followed by carding. The process continues by drawing which includes attenuating in spinning mills.

The silver produced by drawing is then processed for combining where consistent size of cloth is attained. It is then stepped further for roving for purpose to prepare input package. This roving is attenuated by rollers and then spun around the rotating spindle (Textile Learner, 2020).

Weaving

The yarn from the spinning section is sent further for doubling and twisting. Then it is processed for shifting of yarn in convenient form of package containing sufficient yarn length. The exhausted packages are replaced with new ones which is followed by wrapping. The wrap yarn is provided a protective coating to mitigate the breakage of yarn which is called sizing. The yarn is then processed for winding on weaver’s beam supported by the final stage of weaving (Textile Learner, 2020).

After finishing this process, the yarn is transformed into a fabric and it is important to know the conversion rate between them, what is the amount of fabric that one unit of yarn produces.

Dyeing + Printing + Finishing

Dyeing as well as printing are carried out before the application of other finishes to the product in dyeing mills. It provides color to the fabric and improves its appearance. Then, the product is converted from woven to knitted cloth known as finishing. Finishing is applied to give a specific look to the product (Textile Learner, 2020).

Garment Manufacturing

This process converts semi-finished cloth into the finished product. There are various sub processes included in this step such as Designing, Sampling, Costing, Maker Making Cutting, Sewing Washing and much more (Textile Learner, 2020). It is up to the manufacturing companies to know which sub processes to include in their supply chain.

4.3.3 Conversion Rates

In this section, the conversion rates between each item of the system will be displayed, to know the quantity of yarn that one unit of cotton produces; the quantity of fabric that one unit of yarn generates; the quantity of the final product that one unit of fabric develops. With this information, each process will know how much it should produce during its execution.

Table 8: Conversion Rates

Input	Process	Output	Conversion Rate
Cotton	Spinning	Yarn	0.82
Yarn	Weaving	Fabric	0.88
Fabric	Finishing	Final Product	0.82

Inside **Table 8** it is possible to see the input that enters a given process then the output that comes from that process along with the conversion rate from that process. The conversion rate data was taken from Better Cotton Initiative (Better Cotton Initiative, 2020), where it was possible to find the rates from each one of the items.

From the information in **Table 8** it is visible that 1 unit of cotton fiber generates 0.82 units of yarn; 1 unit of yarn produces 0.88 units of fabric, and 1 unit of fabric develops 0.82 units of the final product.

4.3.4 Water and Electrical Consumption

In this section the water usage along with the electrical consumption of each process will be discussed so that the intern knows how much each process consumes from one of the two given resources.

Table 9: Water and Electrical Consumption Values per Process

Process	Electrical Consumption	Water Usage	Minimum Consumption	Maximum Consumption
Spinning	X		3.23 kWh/kg	3.76 kWh/kg
Weaving	X		1.7 kWh/kg	4.2 kWh/kg
Dyeing-Printing-Finishing		X	70L	150L
Garment Manufacturing	X		0.065 kWh/kg	0.195 kWh/kg

From observing **Table 9** it is possible to see that the spinning process has a minimum electrical consumption of 3.23 kWh/kg and a maximum consumption of 3.76 kWh/kg this information was taken from (Kaplan & Koç, 2010); the weaving process has a minimum electrical consumption of 1.7kWh/kg and a maximum one of 4.2kWh/kg according to (Koç & Çinçik, 2010); the dyeing, printing and finishing processes have a minimum water usage of 70L per kg of fabric and maximum usage of 150L per kg of fabric as is shown by (Palamutcu, 2010); finally the garment manufacturing process has a minimum electrical consumption of 0.065kWh/kg and a maximum consumption of 0.195kWh/kg as told by (Palamutcu, 2010).

This information will be very beneficial once the development stage begins because the intern will know the consumption values connected to each process and incorporate it on the implementation stage.

4.3.5 Use Cases

A Use Case is a methodology used in system analysis to identify, clarify, and organize system requirements (*What Is a Use Case?*, n.d.) .

The Use Cases represent the possible actions that the system will have after the development stage. Each Use Case identifies a primary actor that will be responsible to perform that specific Use Case, it also sets the preconditions for that Use Case, which means that there are certain conditions that are needed to be completed before starting the Use Case, there is also the guarantee of success statement which determines if the desired outcome of the Use Case was achieved.

The Use Cases were divided into three parts: the first one represents the System Use Cases, the second one relates to the Normal User Use Cases, and the third represents the Admin User Use Cases. These Use Cases are the functional requirements considered for this internship.

Chapter 4
Requirements

The elicitation process of the functional requirements began when Ubiwhere suggested that the project for this thesis would be to implement a Digital Twin that would replicate a general textile supply chain, where the input of the system would be cotton. From this starting point, the intern started to search how textile supply chains worked, after knowing how they were programmed, the intern began to separate each process from the supply chain and transform it into a specific Use Case. The flowchart developed by the intern proved to be crucial because it allowed the intern to visualize the different processes that belong inside the supply chain.

The intern selected which Use Cases were more appropriate for the Normal User, Admin User and for the System, so the different Use Cases were separated into three groups. The main difference between a typical supply chain and this one is that each process from the production module gives insight into the amount of water that is being consumed in each process, as well as the amount of electrical energy that is consumed. So, for that, Use Cases were developed to accommodate this feature. It was stipulated by the intern and in accordance with Ubiwhere that the Normal Users would only have access to information inside the system and could not change it. Unlike Normal Users, it was defined that the Admin Users could have access to all the data inside the system and change the data accordingly.

System Use Cases

Use Case 1: The cotton fiber data is generated

Table 10: Use Case 1

ID	1
Primary Actor	System
Preconditions	N/A
Guarantee of success	The system generates the supply data
Main Success Scenario	1. The data is generated successfully.
Alternative Paths	1.a. The system cannot generate the data. 1.a.1 The system sends a message reporting the problem.

Use Case 2: Digital Twin register the cotton

Table 11: Use Case 2

ID	2
Primary Actor	System
Preconditions	The system has generated the supply data
Guarantee of success	The Digital Twin registers the cotton into the system.
Main Success Scenario	1. The cotton is registered inside the database of the system.
Alternative Paths	1.a. The system does not receive the supplies and products. 1.a.1 The system sends a message reporting the problem.

Use Case 3: Cotton goes through the spinning process

Table 12: Use Case 3

ID	3
Primary Actor	System
Preconditions	The system has generated the supply data. The system has registered the data inside the database.
Guarantee of success	Cotton is successfully transformed into yarn
Main Success Scenario	1. Cotton goes through the spinning process
Alternative Paths	1.a. The amount of cotton available is not enough to perform the operation. 1.a.1 Cotton does not go through the spinning process and the system sends a message reporting the situation. 2.a. The spinning process has reached maximum occupation and cannot accommodate more cotton at this moment. 2.a.1 Cotton does not go through the spinning process and the system sends a message reporting the situation.

Use Case 4: Yarn enters the weaving process

Table 13: Use Case 4

ID	4
Primary Actor	System
Preconditions	The system has generated the supply data. The system has registered the data inside the database. Cotton has been transformed into yarn.
Guarantee of success	Yarn is successfully transformed into Fabric
Main Success Scenario	1. Yarn goes through the weaving process
Alternative Paths	1.a. The amount of yarn available is not enough to perform the operation. 1.a.1 Yarn does not go through the weaving process and the system sends a message reporting the situation. 2.a. The weaving process has reached maximum occupation and cannot accommodate more yarn at this moment. 2.a.1 Yarn does not go through the weaving process and the system sends a message reporting the situation.

Use Case 5: Fabric enters the dyeing, printing, and finishing processes

Table 14: Use Case 5

ID	5
Primary Actor	System
Preconditions	The system has generated the supply data. The system has registered the data inside the database. Cotton has passed the spinning process and has been transformed into yarn. Yarn has been transformed into fabric and has passed the weaving process.
Guarantee of success	Fabric is transformed into cloth
Main Success Scenario	1. Fabric goes through the dyeing process. 2. Fabric passes the printing process. 3. Fabric enters and successfully leaves the finishing process.
Alternative Paths	1.a. The amount of fabric available is not enough to perform the operation. 1.a.1 Fabric does not go through the dyeing, printing and finishing processes and the system sends a message reporting the situation. 2.a. The dyeing, printing, and finishing processes have reached maximum occupation and cannot accommodate more fabric at this moment. 2.a.1 Fabric does not go through the dyeing, printing and finishing processes and the system sends a message reporting the situation.

Use Case 6: Cloth enters the garment manufacturing process

Table 15: Use Case 6

ID	6
Primary Actor	System
Preconditions	The system has generated the supply data. The system has registered the data inside the database. Cotton has passed the spinning process and has been transformed into yarn. Yarn has been transformed into fabric and has passed the weaving process. Fabric has generated cloth and has passed dyeing, printing and finishing processes.
Guarantee of success	Cloth becomes finished cloth.
Main Success Scenario	1. Cloth passes the garment manufacturing process
Alternative Paths	1.a. The amount of cloth available is not enough to perform the operation. 1.a.1 Cloth does not go through the garment manufacturing process and the system sends a message reporting the situation. 2.a. The garment manufacturing process has reached maximum occupation and cannot accommodate more yarn at this moment. 2.a.1 Cloth does not go through the weaving process and the system sends a message reporting the situation.

User Use Cases

Use Case 7: User wants to check what items are registered inside the system

Table 16: Use Case 7

ID	7
Primary Actor	User
Preconditions	The User has access to the Internet. The User is logged in.
Guarantee of success	The User can see all the items in the system.
Main Success Scenario	1. The User chooses which item to see. 2. The User accesses all items inside the system.
Alternative Paths	1.a. The User is not registered and does not have access to that information. 1.a.1. The System delivers a message to the User saying that he does not have permission to see that information.

Use Case 8: User wants to access the statistics inside the spinning process

Table 17: Use Case 8

ID	8
Primary Actor	User
Preconditions	The User has access to the Internet. The User is logged in.
Guarantee of success	The User can see all the statistics inside the spinning process.
Main Success Scenario	1. The User chooses the spinning process. 2. The User accesses all information regarding the spinning process
Alternative Paths	1.a. The User is not registered and does not have access to that information. 1.a.1. The System delivers a message to the User saying that he does not have permission to see that information.

Use Case 9: User wants to access the statistics inside the weaving process

Table 18: Use Case 9

ID	9
Primary Actor	User
Preconditions	The User has access to the Internet. The User is logged in.
Guarantee of success	The User can see all the statistics inside the weaving process.
Main Success Scenario	1. The User chooses the weaving process. 2. The User accesses all information regarding the weaving process.
Alternative Paths	1.a. The User is not registered and does not have access to that information. 1.a.1. The System delivers a message to the User saying that he does not have permission to see that information.

Use Case 10: User wants to access the statistics inside the dyeing, printing, and finishing processes

Table 19: Use Case 10

ID	10
Primary Actor	User
Preconditions	The User has access to the Internet. The User is logged in.
Guarantee of success	The User can see all the statistics inside the dyeing, printing, and finishing processes.
Main Success Scenario	1. The User chooses the dyeing, printing, and finishing processes. 2. The User accesses all information regarding the dyeing, printing, and finishing processes.
Alternative Paths	1.a. The User is not registered and does not have access to that information. 1.a.1. The System delivers a message to the User saying that he does not have permission to see that information.

Use Case 11: User wants to access the statistics inside the garment manufacturing process

Table 20: Use Case 11

ID	11
Primary Actor	User
Preconditions	The User has access to the Internet. The User is logged in.
Guarantee of success	The User can see all the statistics inside the garment manufacturing process.
Main Success Scenario	1. The User chooses the garment manufacturing process. 2. The User accesses all information regarding the garment manufacturing process.
Alternative Paths	1.a. The User is not registered and does not have access to that information. 1.a.1. The System delivers a message to the User saying that he does not have permission to see that information.

Admin Use Cases

Use Case 12: Admin wants to create, view, delete or update a process and its values from the system.

Table 21: Use Case 12

ID	12
Primary Actor	Admin
Preconditions	The Admin has access to the Internet. The Admin is logged in. The Admin has admin permissions
Guarantee of success	The Admin successfully creates, views, deletes or updates processes and its values.
Main Success Scenario	1. The Admin chooses which process to perform the operation. 2. The Admin chooses the operation that wishes to be performed. 3. The Admin performs the operation on the given process.
Alternative Paths	1.a. The Admin does not have permission to perform operation on a process 1.a.1. The System delivers a message to the Admin saying that he does not possess permission to perform the operation.

Use Case 13: Admin wants to create, update, view or delete processes and its values from the system.

Table 22: Use Case 13

ID	13
Primary Actor	Admin
Preconditions	The Admin has access to the Internet. The Admin is logged in. The Admin has admin permissions
Guarantee of success	The Admin successfully creates, views, deletes or updates one item.
Main Success Scenario	1. The Admin chooses which item to perform the operation. 2. The Admin chooses the operation that wishes to be performed. 3. The Admin performs the operation on the given item.
Alternative Paths	1.a. The Admin does not have permission to perform operation on an item 1.a.1. The System delivers a message to the Admin saying that he does not possess permission to perform the operation.

Use Case 14: Admin wants to create, update, view or delete a user and its data from the system.

Table 23: Use Case 14

ID	14
Primary Actor	Admin
Preconditions	The Admin has access to the Internet. The Admin is logged in. The Admin has admin permissions.
Guarantee of success	The Admin successfully creates, updates, views or deletes a user and its data from the system.
Main Success Scenario	1. The Admin chooses which operation to perform. 2. The Admin chooses the User to perform the operation. 3. The Admin performs the operation.
Alternative Paths	1.a. The Admin does not have permission to perform operation on an item 1.a.1. The System delivers a message to the Admin saying that he does not possess permission to perform the operation.

4.4 Quality Attributes

Quality attributes are characteristics that the system must have in addition to functionality. In another words, a quality attribute is a measurable or testable property of a system that is used to indicate how well the system satisfies the needs of its stakeholders (Silva, 2017).

For this internship modularity, security and portability were considered as the quality attributes for the project to develop.

4.4.1 Modularity

In software engineering, modularity refers to the ability of the system to be divided in smaller components. Modularization is the process of separating the functionality of a program into independent, interchangeable modules, in a way that each module contains everything necessary to execute only part of the desired functionality (Jee, 2021).

A proper software application generally uses modularity as a guideline for development, and for this case it was no exception. It was used to provide re usability of the code base, to divide the project into small parts to use and test them separately.

For this quality attribute, the intern did not create a scenario because, it was considered that developing the software into small pieces was sufficient to guarantee this attribute.

4.4.2 Security

Security is responsible for the ability of the system to reduce the probability of malicious or accidental actions as well as the possibility of theft or loss of data (Syndicode, 2020). The focus of the company was to control the access of the endpoints for non-registered users.

Scenario - The system must prevent non-registered users to access information only available to registered users.

Table 24: Security Scenario

Source of Stimulus	Unregistered Users
Stimulus	An unregistered user wants to access information that is only available to registered users
Environment	Normal Conditions
Artifact	System
Response	The user receives an error message saying that he does not have permission to access that information
Response Measure	The system rejects the request from unregistered users

4.4.3 Portability

Portability is a measure of how easily an application can be transferred from one computer environment to another. It refers to the ability of an application to move across environments (Techopedia, 2011).

It is an important quality attribute to have for this project because the Digital Twin is a PoC, so it is vital that it can be run on other environments.

Scenario - The project must be easily run-on different environments.

Table 25: Portability Scenario

Source of Stimulus	Developer
Stimulus	Developer from Ubiwhere wants to run the project
Environment	Development
Artifact	Project
Response	The project is started inside the new environment
Response Measure	The process to run this project on a different environment takes less than 30 minutes

Chapter 5 Architecture

In this section, the architecture for this project is presented along with the technologies that will be used for the development of the Digital Twin. The architecture is based off the requirements and quality attributes defined in Chapter 4 of the Requirements.

5.1 Architectural Model

Software architecture is the organization of a system, how all the components interact with each other, and all the relationships between them (*What Is Software Architecture - Examples, Tools, & Design | CAST*, n.d.). Another purpose of architecture is that any person can read the architecture and understand it easily, even if that person does not possess any background knowledge on software development. With that in mind, the C4 model was the obvious choice for the representation of the architecture.

The C4 model is an approach to diagramming software architecture, based upon abstractions that reflect how software architects and developers think about and build software (Brown, 2020). This model has four levels, even though the use of them all is not necessary (Brown, 2020).

For the architecture, the following C4 diagrams were considered for the project: context diagram, container diagram, and component diagram.

The context diagram describes what the system does, who will use it, and what other systems will interact with it (*What Is a C4 Model? How to Make C4 Software Architecture Diagrams | Gliffy by Perforce*, n.d.). This diagram presents the scope of the system and what problems the system will solve s

The container diagram is more specific than the context one because it goes into detail about the data components that will be used, all the APIs that will be used, and all the systems that integrate the Digital Twin.

The component diagram specifies all the components that will be implemented during the development stage.

5.2 Technologies

To determine the best technology to use during this internship several technologies were considered for the development of the Digital Twin. Ruby on Rails, Flask, Laravel and Django were studied for this internship as they are frameworks with a great degree of popularity, according to **Figure 8** and easy to work with.

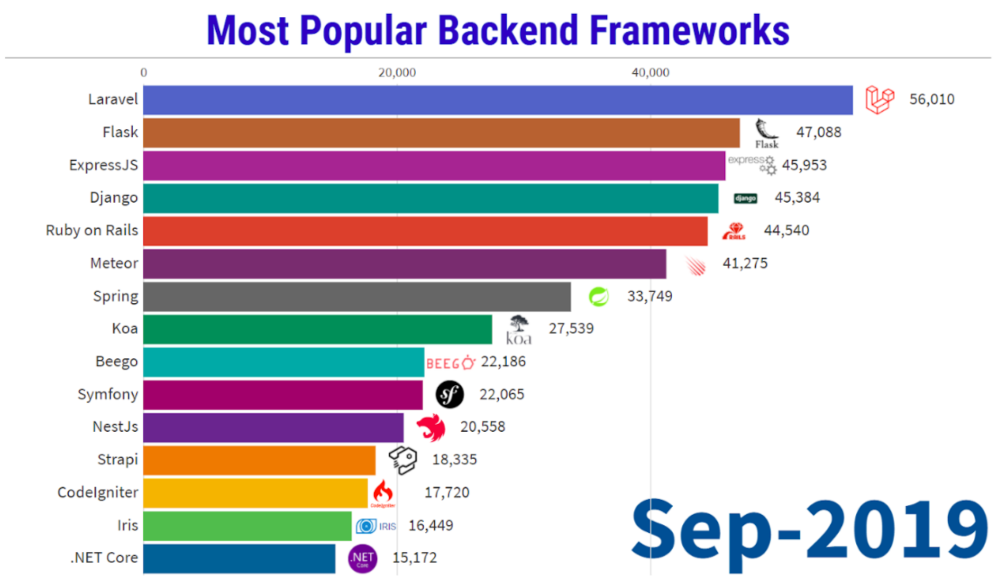


Figure 8: Popular Backend Frameworks

5.2.1 Backend Technologies

Ruby on Rails

Rails is a web application development framework written in the Ruby programming language (Rails Guides Team, 2015).

Rails is a model–view–controller framework, providing default structures for a database, a web service, and web pages. It encourages and facilitates the use of web standards such as JSON or XML for data transfer and HTML, CSS, and JavaScript for user interfacing (Holzner, 2007).

Flask

Flask is a web framework, it's a Python module that lets you develop web applications easily. It has a small and easy-to-extend core: it's a microframework that doesn't include an ORM (Object Relational Manager) or such features (*What Is Flask Python - Python Tutorial*, 2010).

Laravel

Laravel is a free and open-source PHP framework that provides a set of tools and resources to build modern PHP applications. Laravel provides powerful database tools including an ORM (Object Relational Mapper) called Eloquent, and built-in mechanisms for creating database migrations and seeders. With the command-line tool Artisan, developers can bootstrap new models, controllers, and other application components, which speeds up the overall application development (*What Is Laravel? | DigitalOcean*, n.d.).

Django

Django is a high-level web Python framework that encourages rapid development and clean, pragmatic design (Django Software Foundation, 2018).

Comparisons

Table 26: Backend Technologies

	Ruby on Rails	Flask	Laravel	Django
Release Date	2004	2010	2011	2005
Language	Ruby	Python	PHP	Python
Advantages	Time Efficiency Huge and active community	High scalability for simple applications Database integration is easy	Open Source and Wide Community Eloquent ORM	Highly Scalable Open Source and Wide Community
Disadvantages	Shortage of Flexibility Performance Time	Higher maintenance costs for more complex systems Lack of database and ORM	Lack of Inbuilt Support Problematic with certain upgrades	Django is Monolithic Not ideal for smaller projects
Performance	Ruby on Rails is faster than Django	Flask is faster than Django, though the difference is negligible.	Laravel is slower than Django	Django is faster than Laravel
Popularity	High	High	High	High
Ubiwhere support	Low	Low	Low	High

5.2.2 Database Technologies

For the database technology for this project PostgreSQL was considered along with MySQL as they are two of the most used databases nowadays.

PostgreSQL

PostgreSQL is a free and open-source relational database management system (RDBMS) emphasizing extensibility and SQL compliance. PostgreSQL features transactions with Atomicity, Consistency, Isolation, Durability (ACID) properties, automatically updatable views, materialized views, triggers, foreign keys, and stored procedures (Momjian, 2001).

MySQL

MySQL is an open-source relational database management system (RDBMS) (MySQL, 2020). It is a lightweight database which can be installed and used on production application servers with big multi-tier applications as well as on a desktop by developers. It can be installed on all platforms like Windows, Linux, and Mac. It is secure and is not vulnerable to any security vulnerabilities (Smallcombe, 2020).

Comparisons

Table 27: Database Technologies

	PostgreSQL	MySQL
Release Date	2002	1995
Advantages	Excellent for complex queries Supports NoSQL and a large variety of data types ACID compliance	Highly flexible and scalable A Focus on speed and reliability Easy to use and popular
Disadvantages	Comparatively low reading speed Expandable documentation only available in English	MySQL does not support a very large database size as efficiently It suffers from poor performance scaling Transactions are not handled very efficiently
Performance	PostgreSQL is known to be faster while handling massive data sets, complicated queries, and read-write operations	MySQL is known to be faster with read-only commands
Popularity	High	High

5.2.3 Chosen Technologies

The technology chosen for the backend development was Django and the reasons behind this choice are it is one of the primary technologies used by Ubiwhere, which means it will be much easier for them to help the intern during the development of the application; another important aspect is the familiarity that the intern already possesses with Django; this framework provides Object Relational Mapper(ORM) which is a library that automatically transfers data stored in databases into objects commonly used in application code (11 Advantages of Django: Why You Should Use It – Pythonista Planet , n.d.), this way the developer does not have to write SQL code that would be time-consuming; besides that, the Django documentation is very well written and it is relatively easy for a developer to solve a certain problem by searching in the documentation; another advantage of using Django is that it possesses an admin panel, which facilitates the work done by the developer.

For the database, PostgreSQL was the choice made because it is highly scalable, reliable, secure, and stable. Because it is a PoC, the row reading speed of PostgreSQL is not relevant for the protject. It is very easy to integrate it with Django, so it was a natural option, another reason was the previous experience that the intern had with PostgreSQL.

5.3 Architecture

In this section, the C4 context, container and component diagrams are displayed along with a description of each diagram and how they connect to the functional requirements.

5.3.1 Context Diagram

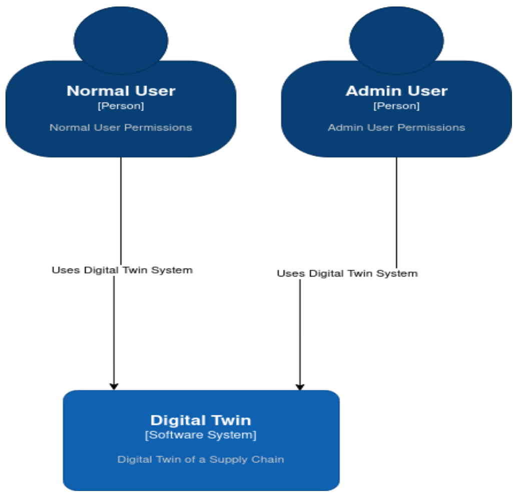


Figure 9: Context Diagram

In **Figure 9** the users of the application are identified, specifically, the Normal User, and the Admin User. These two groups of users interact directly with the Digital Twin System.

There are two types of software architecture methodologies to implement a given system, which are the monolithic and the distributed approach.

Monolithic architecture means that a single code base contains all the required functionalities for a given system, whereas a distributed approach has multiple deployment units connected through remote access protocols. The most known distributed architecture is the microservices one.

The advantages of using the monolithic approach are it is simpler to develop comparing to the microservices approach; easier to deploy as a single jar/ward file is deployed; problems of network latency and security are relatively less in comparison to microservices architecture (GeeksforGeeks, 2020); it fits better a small working team.

The microservices approach also has its own benefits starting with its ease to manage; if there is any update in one of the microservices, then only that microservice must be redeployed; microservices support horizontal scaling; each microservice can use different technologies and it is independent from the others if one of them crashes that does not affect the others (GeeksforGeeks, 2020).

For this internship the monolithic approach was chosen because the development team only consists of the intern; the intern does not have experience dealing with a microservices architecture and a certain degree of expertise is necessary for that approach to pay off; the project that is being developed is a PoC, so the monolithic approach is perfect for rapid production iteration; the scalability and complexity of the Digital Twin is manageable (Blockhuys, 2019). It is important to mention that for a real system in production, a microservices architecture is the ideal choice for this project.

5.3.2 Container Diagram

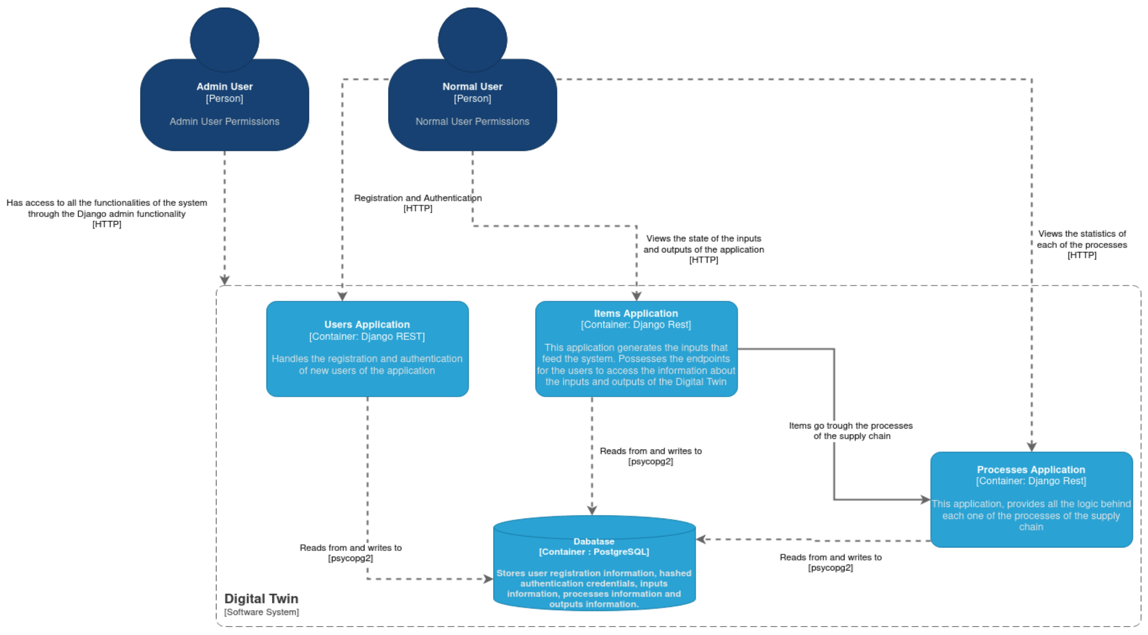

Admin User
[Person]
Admin User Permissions
Normal User
[Person]
Normal User Permissions
Has access to all the functionalities of the system through the Django admin functionality
[HTTP]
Registration and Authentication
[HTTP]
Views the state of the inputs and outputs of the application
[HTTP]
Views the statistics of each of the processes
[HTTP]
Users Application
[Container: Django REST]
Handles the registration and authentication of new users of the application
Items Application
[Container: Django Rest]
This application generates the inputs that feed the system. Possesses the endpoints for the users to access the information about the inputs and outputs of the Digital Twin
Items go trough the processes of the supply chain
Reads from and writes to
[psycopg2]
Processes Application
[Container: Django Rest]
This application, provides all the logic behind each one of the processes of the supply chain
Dabatase
[Container : PostgreSQL]
Stores user registration information, hashed authentication credentials, inputs information, processes information and outputs information.
Reads from and writes to
[psycopg2]
Reads from and writes to
[psycopg2]
Digital Twin
[Software System]

Figure 10: Container Diagram

In **Figure 10** it is possible to see that each one of these Django applications represents a functional module that was defined during the requirements stage. In this container diagram it is possible to observe that there are three main applications behind the Digital Twin system which are users, items, and processes application.

The users application handles the registration and authentication of users of the system, items application generates the inputs that feed the system and has the endpoints that give information to the users about the inputs and outputs of the system. Finally, the processes application has all the information about the processes of the supply chain, and the users can access the endpoints to obtain the data about water usage and gas emissions.

It is important to mention that all the applications communicate directly with the PostgreSQL database that is responsible to store all data related to the system including registration, authentication, inputs, outputs, and processes information.

5.3.3 Component Diagram – Users Application

In this subsection the component diagram for the users application is displayed along with all the connections that the components make with different elements from the system.

By looking at **Figure 11** it is possible to see the component diagram of the users application. In this application there are two important components for the application the register controller where the logic for the registration is developed alongside the authentication controller which controls who has access to the endpoints of the application. The two controllers from the application communicate directly with the database.

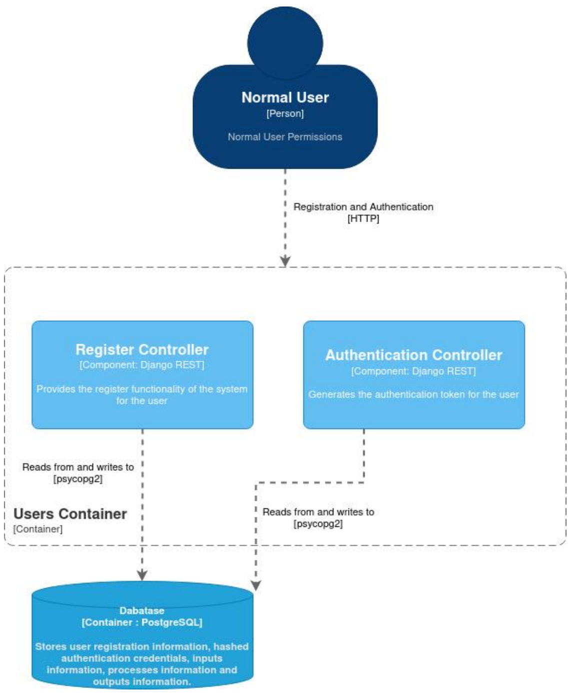

Normal User
[Person]
Normal User Permissions
Registration and Authentication
[HTTP]
Register Controller
[Component: Django REST]
Provides the register functionality of the system for the user
Authentication Controller
[Component: Django REST]
Generates the authentication token for the user
Reads from and writes to
[psycopg2]
Users Container
[Container]
Reads from and writes to
[psycopg2]
Dabatase
[Container : PostgreSQL]
Stores user registration information, hashed authentication credentials, inputs information, processes information and outputs information.

Figure 11: Component Diagram - Users Application

5.3.4 Component Diagram – Items Application

In this subsection the component diagram from the items application is presented highlighting the different connections that every element from it makes with the different components.

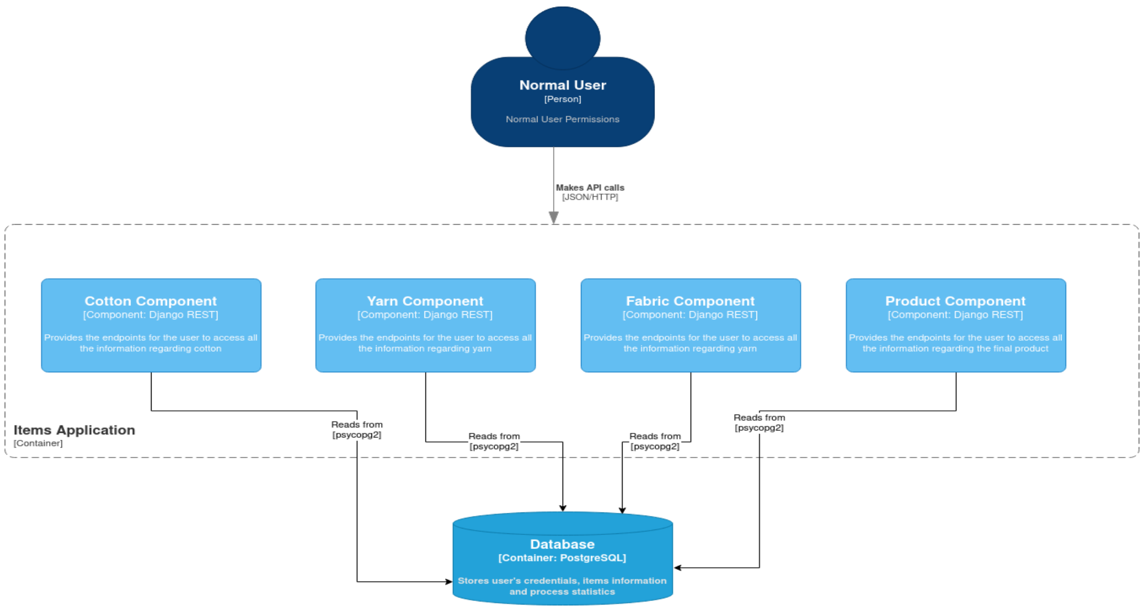

Normal User
[Person]
Normal User Permissions
Makes API calls
[JSON/HTTP]
Cotton Component
[Component: Django REST]
Provides the endpoints for the user to access all the information regarding cotton
Yarn Component
[Component: Django REST]
Provides the endpoints for the user to access all the information regarding yarn
Fabric Component
[Component: Django REST]
Provides the endpoints for the user to access all the information regarding yarn
Product Component
[Component: Django REST]
Provides the endpoints for the user to access all the information regarding the final product
Items Application
[Container]
Reads from
[psycopg2]
Reads from
[psycopg2]
Reads from
[psycopg2]
Reads from
[psycopg2]
Database
[Container: PostgreSQL]
Stores user's credentials, items information and process statistics

Figure 12: Component Diagram - Items application

In **Figure 12** it is possible to observe the four different components that together integrate the items application. Inside each component, there are endpoints available to the user that let him access information regarding each element from the system starting with cotton, which is the input of the system, and ending with the final product component where it is possible to access all information regarding the product.

All the different components interact directly with the database, and they only have permission to read from it because the normal user does not possess authorization to write into the databases.

The flow of execution of this diagram is the following, the user makes a request to the API of the items application, the application processes the request and chooses which component should handle it, after choosing it the component retrieves the desired information from the database and returns it to the user.

5.3.5 Component Diagram – Processes Application

In this subsection, the component diagram for the processes application is displayed along with connection between each component and the database.

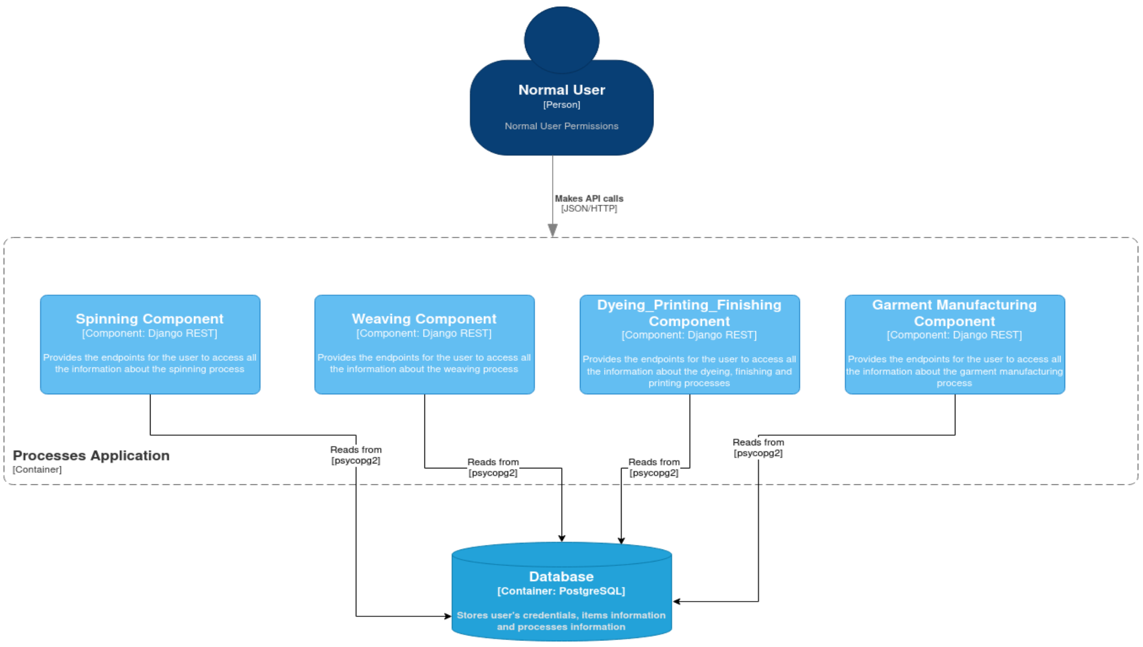

Normal User
[Person]
Normal User Permissions
Makes API calls
[JSON/HTTP]
Spinning Component
[Component: Django REST]
Provides the endpoints for the user to access all the information about the spinning process
Weaving Component
[Component: Django REST]
Provides the endpoints for the user to access all the information about the weaving process
Dyeing_Printing_Finishing Component
[Component: Django REST]
Provides the endpoints for the user to access all the information about the dyeing, finishing and printing processes
Garment Manufacturing Component
[Component: Django REST]
Provides the endpoints for the user to access all the information about the garment manufacturing process
Processes Application
[Container]
Reads from
[psycopg2]
Reads from
[psycopg2]
Reads from
[psycopg2]
Reads from
[psycopg2]
Database
[Container: PostgreSQL]
Stores user's credentials, items information and processes information

Figure 13: Component Diagram - Processes Application

In **Figure 13**, the component diagram for the process's application is displayed, there it is visible that there four different components, each one of them represents one of the processes of the textile supply chain explained in Chapter 4 of the Requirements.

It was important to separate each process into different components because when a given item enters one process it will only go through that unique process, for example cotton only enters the spinning process.

Inside each component, there will be endpoints available to the user that will return information about the sustainability of a given process as well as the number of items processed by each process. Each component interacts directly with the database by performing read-only operations from it.

Chapter 6 Implementation

The following Chapter presents all the stages that occurred during the implementation stage of the Digital Twin. First, the environment used for the development will be explained, after that the data models will be presented. Sequentially, each process regarding the supply chain will be properly described along with the authentication method chosen for this internship. Finally, an explanation of the functioning of the whole system will be given.

6.1 Environment

The following section describes the development environment used for the implementation stage along with the decisions behind the choice of a given tool.

6.1.1 Environment Choice

Before the development stage began, the intern had to make a choice between installing all necessary tools and technologies directly into the intern's computer or use a virtualization tool. With the goal of saving time and space, the intern decided that the best option would be to use a virtualization tool for this project. Another added benefit of this option is that it is easy to run the project in another machine because virtualization provides that flexibility, instead of having to install all the technologies and dependencies into another computer. For this project, Docker and Virtual Machines were considered as options to be the virtualization tool.

Virtualization

Virtualization is the process of creating a simulated computing environment that is abstracted from the physical computing hardware. It enables the user to create multiple, virtual computing instances from the hardware and software components of a single machine.

Docker

Docker is a visualization software that helps its users in developing, deploying, monitoring, and running applications in a Docker container with all their dependencies. A container is a standard unit of software that packages up code and all its dependencies, so the application runs quickly and reliably from one computing environment to another.

Docker container image is a lightweight, standalone, executable package of software that includes everything needed to run an application.

Virtual Machine

A virtual machine is no different from a physical computer. It has the same properties such as CPU, memory, disks to store files and can connect to the Internet. Instead of having physical hardware like a physical computer, they are defined as virtual computers within physical servers, existing only as code. The virtual machine is separated from the rest of the system, which means that the software inside the virtual machine does not affect with the computer's primary operating system.

Comparison

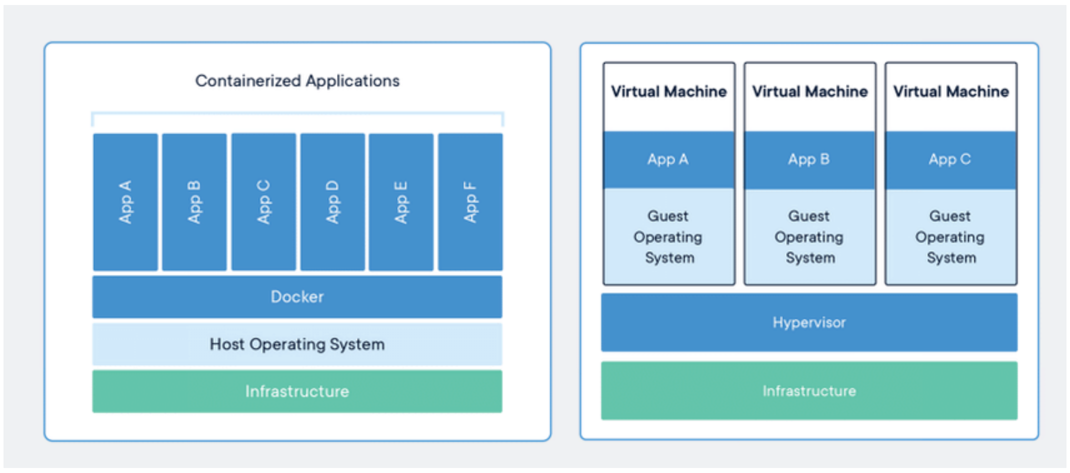


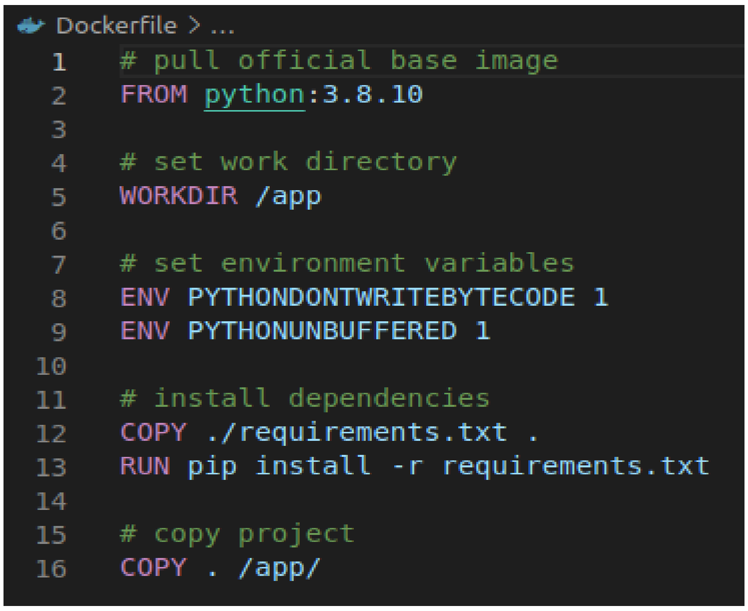
Figure 14: Docker vs Virtual Machine

By observing **Figure 14**, it is possible to see that Docker containers do not require a hardware hypervisor, which translates into far fewer resources consumed compared to the resources used by a Virtual Machine. Besides this, Docker is much faster than a Virtual Machine, which can take a few minutes to boot and be dev-ready, Docker only takes a few milliseconds to a few seconds to initiate.

Docker containers also offers the possibility of portability between different machines, which means that the applications inside the containers can be run in any machine, this benefit is not present inside the Virtual Machine. For these reasons, Docker was the natural choice for the environment of this project.

6.1.2 Environment Setup

Docker can build images automatically by reading the instructions from a Dockerfile. A Dockerfile is a text document that contains all the commands a user could call on the command line to assemble an image (*Dockerfile Reference | Docker Documentation*, n.d.). In **Figure 14** it is possible to see the Dockerfile of the Django application.



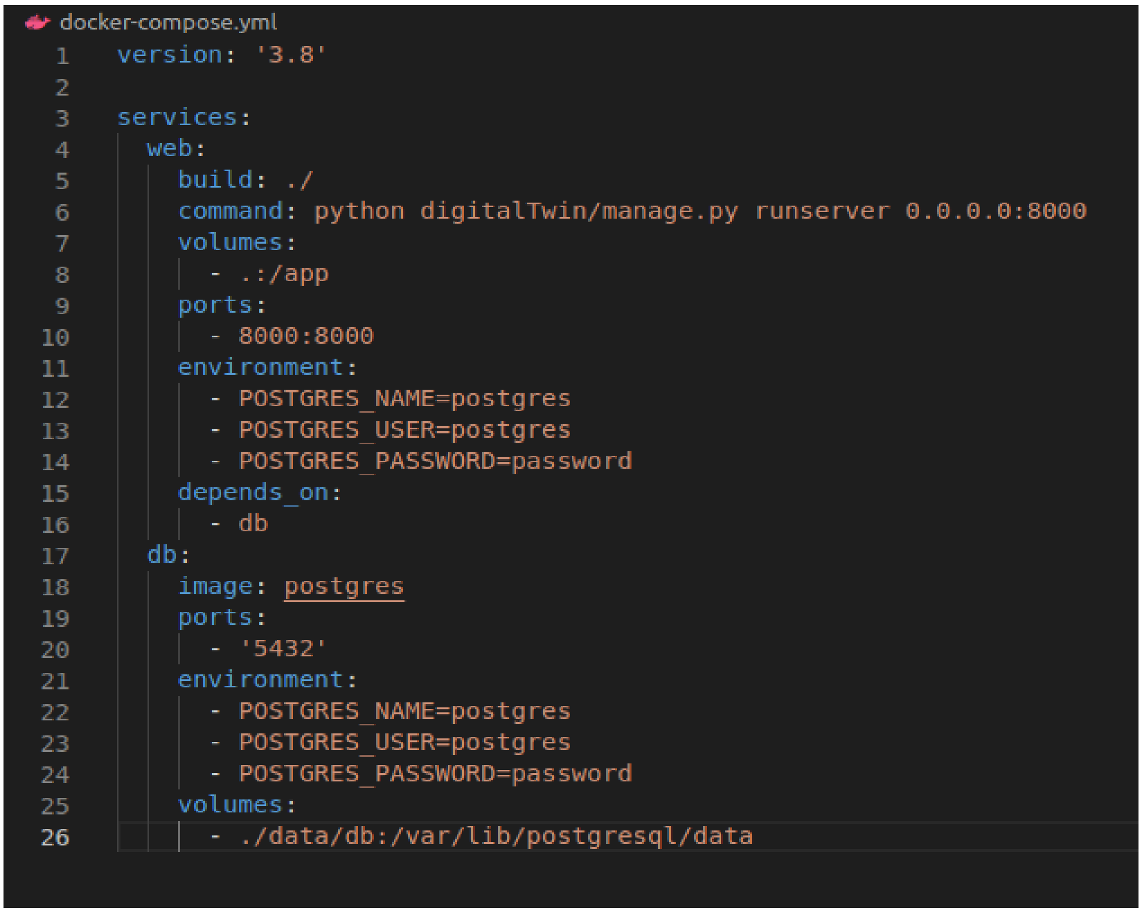
Dockerfile > ...

```
# pull official base image
FROM python:3.8.10

# set work directory
WORKDIR /app

# set environment variables
ENV PYTHONDONTWRITEBYTECODE 1
ENV PYTHONUNBUFFERED 1

# install dependencies
COPY ./requirements.txt .
RUN pip install -r requirements.txt

# copy project
COPY . /app/
```

Figure 15: Django Application Dockerfile

In **Figure 15** it is visible the process begins by pulling the official base image of Python 3.8.10. It is important to note that all the requirements needed for this project are inside the file requirements.txt, which is copied during the Dockerfile execution as well as their installation.

To deploy these Dockerfiles docker compose was used. Docker Compose is a tool that was developed to help define and share multi-container applications. A YAML file is used to configure the project services, then with a single command, the services of the configuration are created and started (*Overview of Docker Compose | Docker Documentation*, n.d.). YAML files are human-readable and easy to understand, and they are usually used for configuration files. **Figure 16** displays the docker-compose.yml file that was used to configure this project.



```yaml
docker-compose.yml
version: '3.8'

services:
  web:
    build: ./
    command: python digitalTwin/manage.py runserver 0.0.0.0:8000
    volumes:
      - .:/app
    ports:
      - 8000:8000
    environment:
      - POSTGRES_NAME=postgres
      - POSTGRES_USER=postgres
      - POSTGRES_PASSWORD=password
    depends_on:
      - db
  db:
    image: postgres
    ports:
      - '5432'
    environment:
      - POSTGRES_NAME=postgres
      - POSTGRES_USER=postgres
      - POSTGRES_PASSWORD=password
    volumes:
      - ./data/db:/var/lib/postgresql/data
```

Figure 16: Docker-Compose Configuration

In **Figure 16** it is possible to see the commands that are necessary to run the application, the port in which the application will be executed. In this case there are two services one for the backend application "web" and another for the database "db". The "web" service depends on the "db" service, because all the data that the backend application uses is stored inside the database. Volumes are used to persist the generated data, which eradicates data loss when the container is turned off.

6.2 Data Models

The data models defined for the implementation were based off from the processes detailed initially in the Requirements Chapter and on the inputs and outputs of the system also specified in the same Chapter.

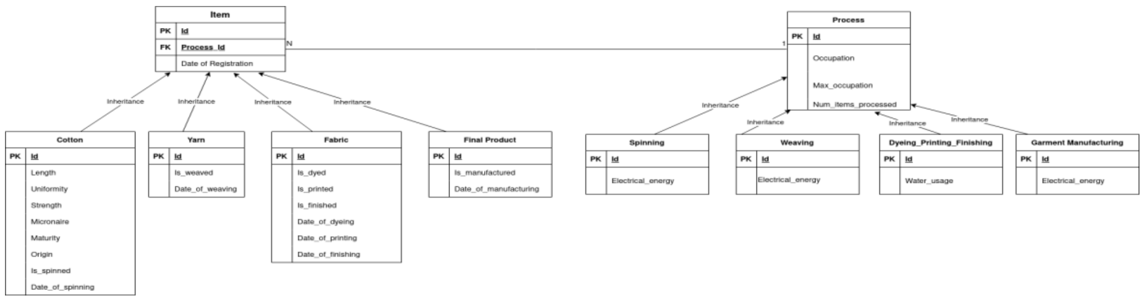


Figure 17: Digital Twin Data Models

In **Figure 17**, the data models designed for the implementation are displayed. Ten models were defined for this, it is important to mention that the user and admin models are not represented in **Figure 17**. The full image can be seen inside Appendix A.

The item model serves as parent class for the cotton, yarn, fabric, and final product models. The item model has a relation of **many to one** with the process model, which means that one process can handle many items, this model has id as the primary key, *Process_id* has a foreign key and a date field to know the date of registration of the item.

The cotton model has id as the primary key, cotton attributes as the other fields, *Is_spinned* as field to know if the cotton has gone through the spinning process and *Date_of_spinning* as field to capture the date of when the unit of cotton has been spun. The yarn model has *Is_weaved* as a field to know if the unit of yarn has passed the weaving process and *Date_of_weaving* as a field to perceive when it has passed that process. The fabric model has three fields to know if it has passed the dyeing, printing, and finishing processes and three more fields to know the dates of when that fabric has passed them. At last, the final product model has a field to know if the product has been manufactured and a date field associated to see the date of manufacturing.

The process model serves as the parent class for the spinning, weaving, *dyeing_printing_finishing* and garment manufacturing processes. They inherit the occupation field that gives the information about the current usage of the process; *Max_occupation* field that tells what the maximum capacity of that process is and *Num_items_processed* which gives information about the number of items processed by a given process.

The spinning, weaving and garment manufacturing processes have *Electrical_energy* as field to know the amount of electrical energy that has been consumed until now, whereas the dyeing_printing_finishing process has *Water_usage* as an attribute to see the quantity of water used.

6.3 Django Applications

Firstly, it is important to mention that Django follows a MVT structure, where the **Model** will act as the interface of the data that will persist inside the project. It is represented by a database, in this case by a PostgreSQL one. The **View** accepts HTTP requests from the browser and returns HTTP responses. It makes the connection between the **Model** and the **Template**, rendering the template according to the user input. In this case, because the project itself is an API there is no need to mention the **Template**.

Each Django application possesses a **models.py** where the models that will be saved inside the database are defined; **views.py** is the file where view functions are coded, each function takes a web request and returns a web response (Django, 2021b); there is also a **migrations** folder that represents the changes that are made to the models into the database schema (Django, 2021a). Because the intern is using Django REST for the development of the project there is a file that is created in each Django application and it is called **serializers.py**, in this file **serializers** of the models are defined and they allow complex data such as querysets and model instances to be converted to native Python datatypes that then can be rendered into JSON, XML or other content types (Keith-Magee, n.d.). **Serializers** also provide deserialization which allows parsed data to be converted back into complex types, after first validating the incoming data (Keith-Magee, n.d.).

For the development of the Digital Twin three Django applications were created to help differentiate the various elements of the project. There is a Django application for the users, that covers user creation and their authentication; another Django application that handles all the inputs and outputs during the execution of the program; there is also a Django application that takes care of all the processes that exist inside the Digital Twin which includes spinning, weaving, dyeing, printing, finishing and garment manufacturing.

6.3.1 Users Application

In this application, the User models are defined, in this case there is a class called **UserManager** that contains the methods to create a normal user and a super user which represents the administrator of the project. There is a **User** class that inherits the properties of the **AbstractBaseUser**, which provides the core implementation of a user model, including hashed passwords and tokenized password resets (Django, n.d.); this class also inherits the properties of the abstract class **PermissionsMixin**, that gives all the methods and database fields necessary to support Django’s permission model (Django, n.d.).

Registration and Authentication

For a normal user to register into the application, the user has to access the endpoint ‘***api/register/***’ and provide a valid username, email and password. The system checks if the data is valid and generates an authentication token associated to the user. The personal data and the token are stored inside the database in separate tables. The response from this request can be positive where the authentication token is sent back to the user or negative when the data provided by the user is not valid.

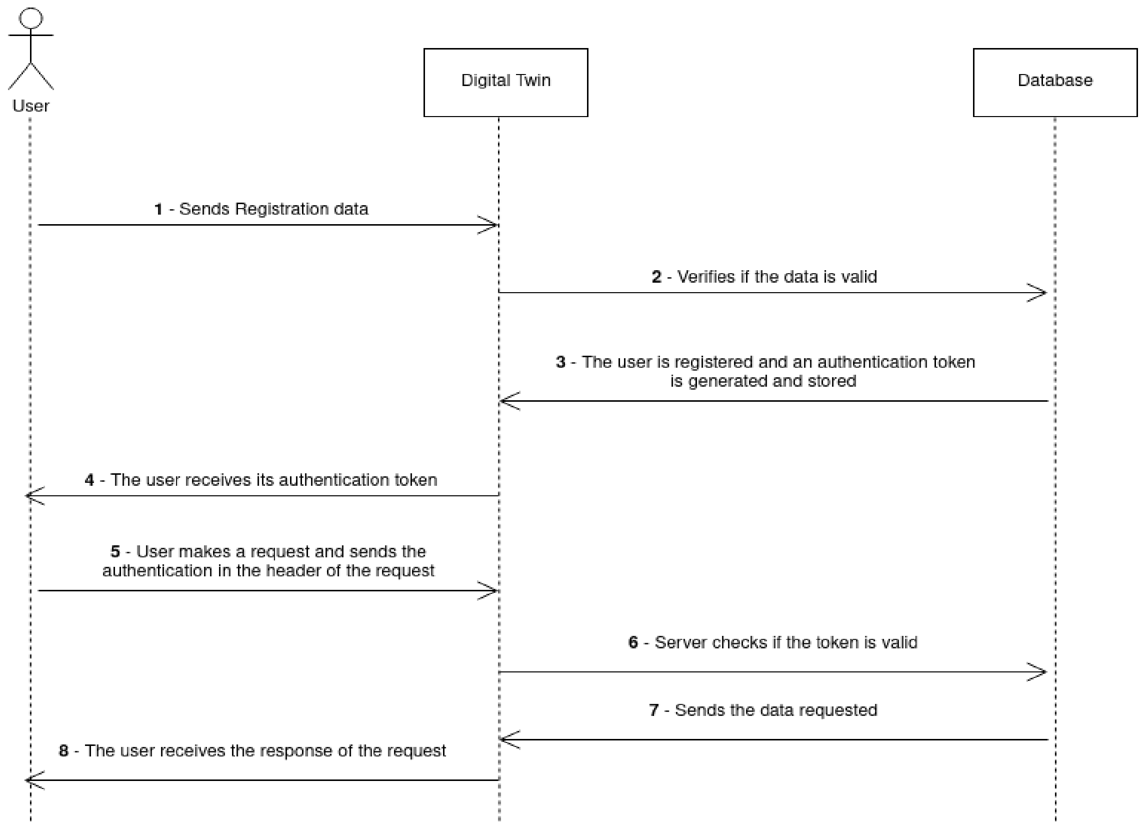

User
Digital Twin
Database
1 - Sends Registration data
2 - Verifies if the data is valid
3 - The user is registered and an authentication token is generated and stored
4 - The user receives its authentication token
5 - User makes a request and sends the authentication in the header of the request
6 - Server checks if the token is valid
7 - Sends the data requested
8 - The user receives the response of the request

Figure 18: Registration and Token Authentication

Following the flow from **Figure 18**, every time the user wants to make a request to access info from the Digital Twin, the user must send the authentication token, inside the header of the request, that was generated during the registration process. The server checks the validity of the token and responds accordingly. If the user needs to access the authentication token, the user can use the following endpoint to get it '**api/token/auth/'**, where the user provides username, email, and password to match credentials on the database and the server sends back the token.

The endpoints created inside this application are presented in the following **Table 18**, as well as their purpose inside the application.

Table 28: User Application Endpoints

Routes	Purpose
POST /api/register/	This endpoint is used to register new users to the application, where the user submits the data necessary to enter the register endpoint. The server responses with the authentication token if the request was successful or an error message;
POST /api/token/auth/	This endpoint receives credentials from a user and returns the authentication token to the user if the request was successful or an error message
GET /api/users/	This endpoint returns a list of all the users that registered inside the application. It is important to mention that this endpoint is protected, only authenticated users can access it

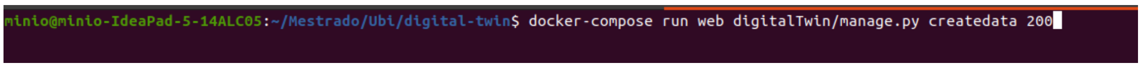
6.3.2 Items Application

Inside this application, the models for the inputs and outputs for the systems are implemented. There is an Items model that serves as the basis for inputs and outputs of the program. In the case of this Digital Twin, the input of the system is cotton, the middle products are yarn and fabric, and the final product is the finished product. There are four models for this and each one of them inherits the Item model properties and attributes, so they can be accessed at any instance of these models.

Input Generation

The generation of the data to feed the Digital Twin is done inside this application using custom django-admin commands, which gives the ability to the developers to implement their own actions using the terminal (Django Software Foundation, 2018). Inside the **createdata.py** file there is a method called "handle" that has all the logic behind the generation of the cotton for the system.

It is important to mention that the data generated is based off real values of cotton production (Truent, 2018). The attributes from the Cotton Model were also defined by taking into consideration the measurements that are made with cotton. For this project the length, uniformity, strength, maturity, elongation, and country of origin of the cotton are considered as the attributes for the cotton. Following the US Cotton Fiber Chart (US Cotton Fiber Chart, 2021), it was possible to generate real values of the cotton attributes, to make the simulation even more realistic.



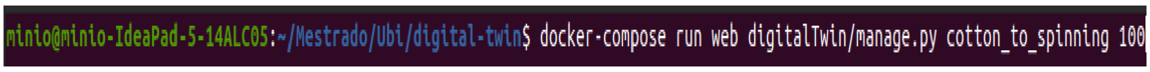
```
minio@minio-IdeaPad-5-14ALC05:~/Mestrado/Ubi/digital-twin$ docker-compose run web digitalTwin/manage.py createdata 200
```

Figure 19: Data Generation

In **Figure 19**, the command that generates the necessary data that feeds the Digital Twin. By using this command 200 units of cotton are generated and directly registered inside the database.

Allocating cotton to the spinning process

To allocate cotton to the spinning process, a django-admin command is also used, where the number of units of cotton to send to that specific process are defined inside the terminal. The system verifies if in fact the number of units to allocate are indeed available in the system, then it checks if the process occupation is greater or equal to the number of cotton units to be spun. If it passes all these requirements, then each unit of cotton enters the spinning process and starts being transformed into yarn.



```
minio@minio-IdeaPad-5-14ALC05:~/Mestrado/Ubi/digital-twin$ docker-compose run web digitalTwin/manage.py cotton_to_spinning 100
```

Figure 20: Command to send cotton to the spinning process

In **Figure 20**, the command that sends the cotton registered inside the system, that has not been spun, to the spinning process is displayed, where 100 units of cotton are sent to that process.

Each passage from one process to another is made using a django-admin command and it follows the same logic as the previous explained commands. The remaining commands are: the ***yarn_to_weaving*** command that sends units of yarn to the weaving process and transforms it into fabric; ***fabric_to_dyeing*** one where the units of fabric are sent to the dyeing process; ***fabric_to_printing*** where the units of fabric are sent to the printing process; ***fabric_to_finishing*** where the units of fabric are sent to the finishing process; the last one is ***fabric_to_garment_manufacturing*** where the fabric passes through the garment manufacturing process and transforms the fabric into the final product. These transformations follow the conversion rates that were defined inside Chapter 4 of the Requirements. In a real system these commands would correspond to a sensor or an external service that would perform the actions automatically. The commands described have the goal of controlling the flow of how the current Digital Twin solution works.

Endpoints

The logic developed for the endpoints of this application was coded inside the **views.py** file. Each class inside this file uses **ViewSets,** which simplifies and reduces the lines of code needed to implement a group of endpoints belonging to the same Model (ViewSets - Django REST Framework, n.d.), inside each class there is a property that does the authentication verification, which verifies if the user making the request to the application is authenticated or not.

Table 29: Items application endpoints

Routes	Purpose
GET /api/cotton/	This endpoint is used for a registered user to access all cotton stored in the database;
GET /api/cotton/<int:pk>	This endpoint is used for a registered user to access information about a specific cotton stored in the database;
GET /api/cotton/cotton_spun/	This endpoint is used for a registered user to access all cotton that has passed the process of spinning, which is stored inside the database;
GET /api/cotton/cotton_not_spun/	This endpoint is used for a registered user to access all cotton that has not passed the process of spinning, which is stored inside the database;

In **Table 29**, some of the endpoints that were developed during the implementation stage, that belong to the items application are displayed. There, the routes for each endpoint are displayed as well as the type of request given for each route. All the endpoints are shown Appendix B.

6.3.3 Processes Application

Inside this application, the processes of the textile supply chain are defined: spinning, weaving, dyeing, printing, finishing and garment manufacturing. Each one of these processes has a model associated with it, where the occupation, energy consumed, and water usage are attributes of the given model. Each item from the items application will go through all of the processes that belong to this application, in order to have a completed product by the end of supply chain cycle.

Table 30: Processes application endpoints

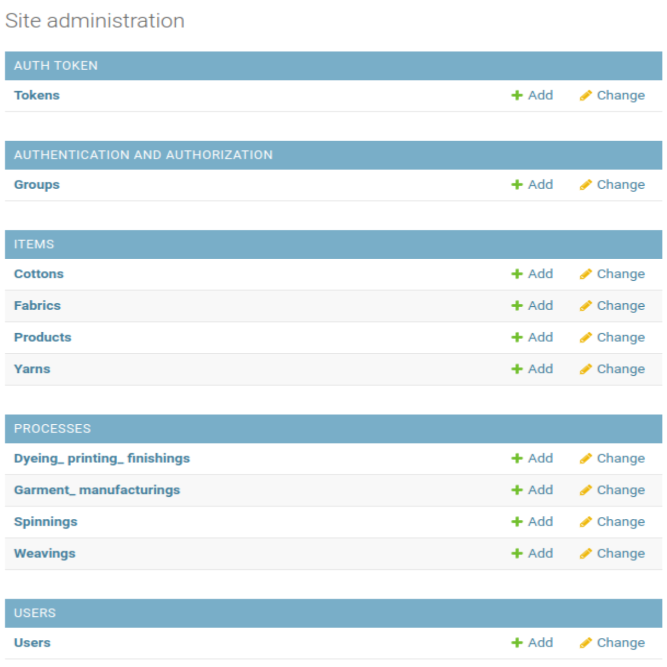
Routes	Purpose
GET /api/weaving/	This endpoint is used for a registered user to access all the information inside the weaving process including the total number of weaved products, electrical energy consumed, current occupation and maximum occupation;
GET /api/spinning/	This endpoint is used for a registered user to access all the information inside the spinning process including the total number of spinned products, electrical energy consumed, current occupation and maximum occupation;
GET /api/dyeing/	This endpoint is used for a registered user to access all the information inside the dyeing, finishing, and printing processes including the total number of dyed, printed and finished products, water usage, current occupation, and maximum occupation;
GET /api/garment_manufacturing/	This endpoint is used for a registered user to access all the information inside the garment manufacturing process including the total number of manufactured products, electrical energy consumed, current occupation and maximum occupation;

In **Table 30**, all the endpoints of the processes application are presented along with a given route and its type of request associated with that route, besides that there is a column that describes the functionality of each route.

6.3.4 Django Admin

The Django admin application is one of the main features of the Django framework, it is used to automatically build a site where it can be used in conjunction with the models defined previously to create, view, update, and delete records making it easy and time saving to test the models created during the development stage (Admin, n.d.).

The manager of this application is the Admin User, which was defined previously at the Requirements Chapter 4. This user is responsible to oversee all stages of the supply chain and the state of the products inside the system.



Site administration

AUTH TOKEN		
Tokens	+ Add	Change

AUTHENTICATION AND AUTHORIZATION		
Groups	+ Add	Change

ITEMS		
Cottons	+ Add	Change
Fabrics	+ Add	Change
Products	+ Add	Change
Yarns	+ Add	Change

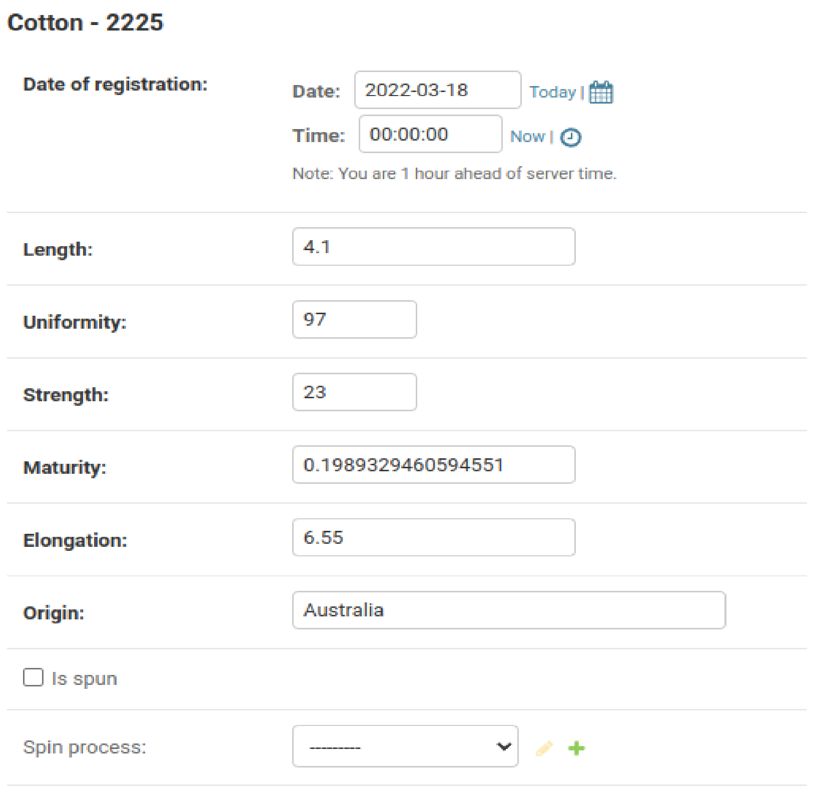
PROCESSES		
Dyeing_ printing_ finishings	+ Add	Change
Garment_ manufacturings	+ Add	Change
Spinnings	+ Add	Change
Weavings	+ Add	Change

USERS		
Users	+ Add	Change

Figure 21: Django Admin Homepage

In **Figure 21**, the django admin panel is displayed and all the models defined during this stage are presented inside it. This application was mainly used to monitor the values from the items and processes application, if any value from this application was not correct the admin would intervene and correct that imprecision.

Besides that, this panel was used for testing purposes to check if the passages of the items from one process to another was being performed as expected. It was also used to see if the values inside each application were being updated accordingly with the models that were defined.



Cotton - 2225

Date of registration:	Date: 2022-03-18 Today | Time: 00:00:00 Now | Note: You are 1 hour ahead of server time.
Length:	4.1
Uniformity:	97
Strength:	23
Maturity:	0.1989329460594551
Elongation:	6.55
Origin:	Australia
☐ Is spun	
Spin process:	---------

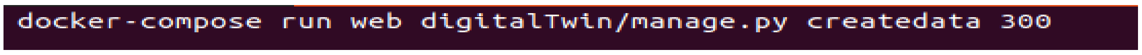
Figure 22: Django Admin Cotton

In **Figure 22**, a unit of cotton is displayed, that had been generated inside the system. Here, the admin user can alter any of the attributes present in the object. This is also applied to all models present inside the django admin panel.

6.4 System Setup

In this section, a simulation of the project developed is shown along with all the products inside the system, the values from each stage of the supply chain and the electrical and water consumption.

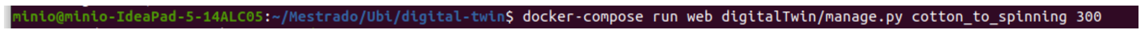
To start off, 300 units of cotton will be generated to feed the Digital Twin. All the commands inside the Digital Twin are shown in this part of the Chapter.



```
docker-compose run web digitalTwin/manage.py createdata 300
```

Figure 23: Generation of 300 units of cotton

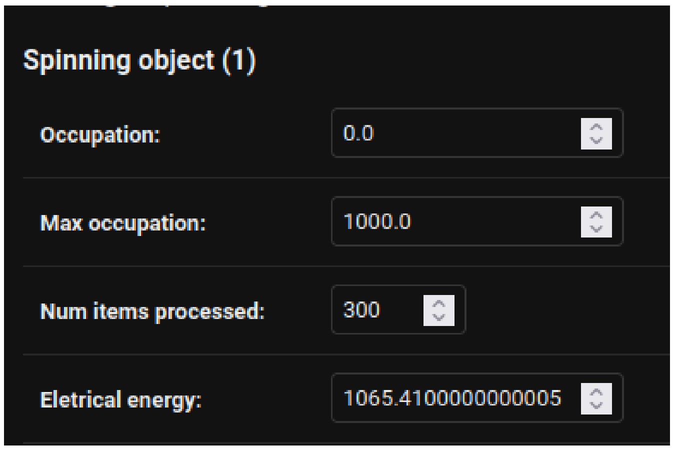
In **Figure 23**, the command generates 300 units of cotton and stores them inside the system. Then, 300 units of cotton will be sent to the spinning process, to transform the cotton into yarn.



```
minio@minio-IdeaPad-5-14ALC05:~/Mestrado/Ubi/digital-twin$ docker-compose run web digitalTwin/manage.py cotton_to_spinning 300
```

Figure 24: Sending 300 units of cotton to the spinning process

In **Figure 24**, the command sends 300 units of cotton to the spinning process and generates 246 yarn units.



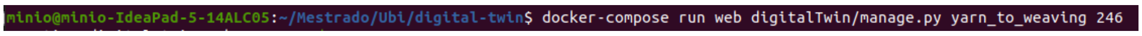
Spinning object (1)

Occupation:	0.0
Max occupation:	1000.0
Num items processed:	300
Eletrical energy:	1065.4100000000005

Figure 25: Spinning Process

In **Figure 25**, it is visible that 300 units of cotton have been processed and transformed into yarn. Besides that, the electrical consumption of the processing is shown, and the value is **1065.41 kWh/kg**.

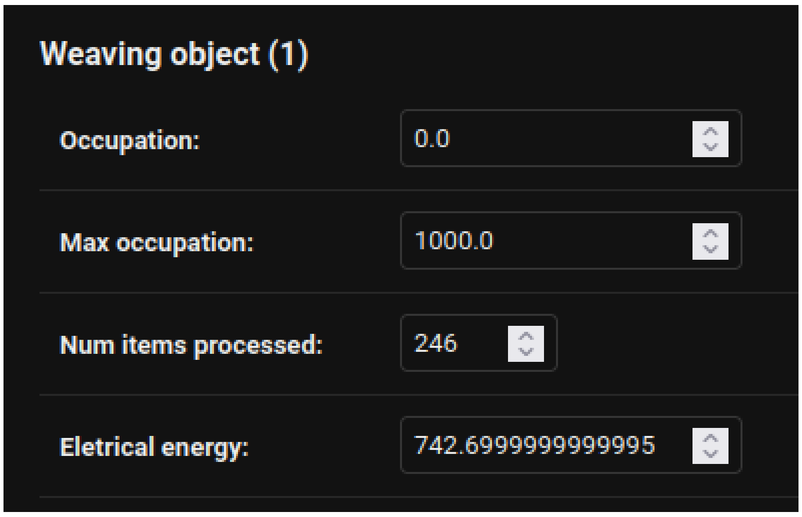
Then, 246 units of yarn are passed to the weaving process and transformed into fabric.



```
minio@minio-IdeaPad-5-14ALC05:~/Mestrado/Ubi/digital-twin$ docker-compose run web digitalTwin/manage.py yarn_to_weaving 246
```

Figure 26: Sending 246 units of yarn to the weaving process

In **Figure 26**, the command to send 246 units of yarn to weaving process is displayed. After performing this command, 217 units of fabric are generated from the yarn.



Weaving object (1)

Occupation: 0.0

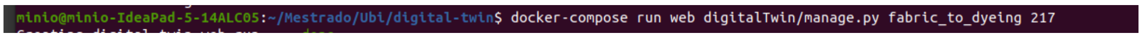
Max occupation: 1000.0

Num items processed: 246

Eletrical energy: 742.6999999999995

Figure 27: Weaving process

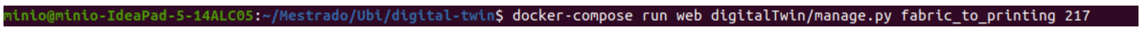
In **Figure 27**, the weaving process is shown where it is possible to see the number of yarns that have been processed by it. Besides that, the electrical energy consumed is also displayed.



```
minio@minio-IdeaPad-5-14ALC05:~/Mestrado/Ubi/digital-twin$ docker-compose run web digitalTwin/manage.py fabric_to_dyeing 217
```

Figure 28: Sending 217 units of fabric to the dyeing process

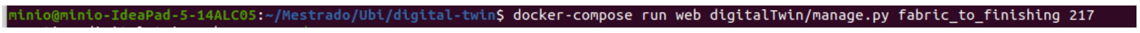
In **Figure 28**, the command to send units of fabric to the dyeing process is illustrated, in this case 217 units of fabric are sent to the dyeing one.



```
minio@minio-IdeaPad-5-14ALC05:~/Mestrado/Ubi/digital-twin$ docker-compose run web digitalTwin/manage.py fabric_to_printing 217
```

Figure 29: Sending 217 units of dyed fabric to the printing process

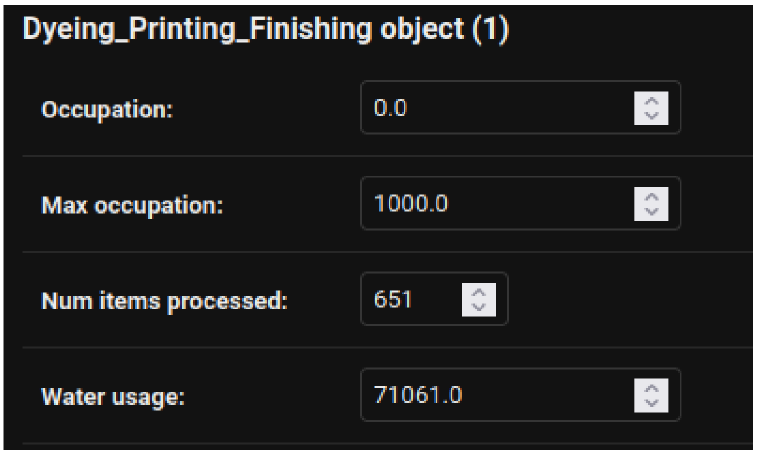
In **Figure 29**, the command to send units of dyed fabric to the printing process is displayed, in this case 217 units of fabric are sent to the printing one.



```
minio@minio-IdeaPad-5-14ALC05:~/Mestrado/Ubi/digital-twin$ docker-compose run web digitalTwin/manage.py fabric_to_finishing 217
```

Figure 30: Sending 217 units of dyed and printed fabric to the finishing process

In **Figure 30**, the command to send units of dyed and printed fabric to the finishing process is shown, in this case 217 units of fabric are sent to the process. Then, 188 of the final products are produced.

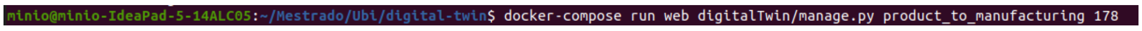


Dyeing_Printing_Finishing object (1)

Occupation:	0.0
Max occupation:	1000.0
Num items processed:	651
Water usage:	71061.0

Figure 31: Dyeing, Printing and Finishing Processes

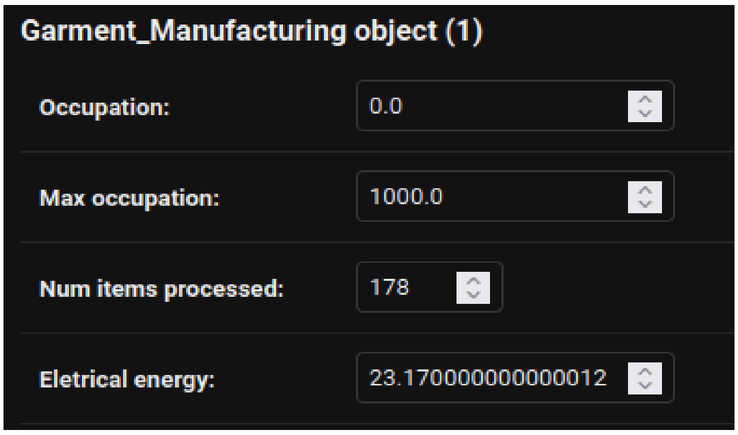
In **Figure 31**, the processes of dyeing, printing, and finishing processes are displayed. The water consumption is shown along the number of items processed by these processes.



```
minio@minio-IdeaPad-5-14ALC05:~/Mestrado/Ubi/digital-twin$ docker-compose run web digitalTwin/manage.py product_to_manufacturing 178
```

Figure 32: Command that sends 178 units of product to the garment manufacturing process

In **Figure 32**, the command that sends units 178 units of the final product to the garment manufacturing is illustrated.


Garment_Manufacturing object (1)

Occupation:	0.0
Max occupation:	1000.0
Num items processed:	178
Eletrical energy:	23.170000000000012

Figure 33: Garment Manufacturing Process

In **Figure 33**, the processes of garment manufacturing are displayed. The electrical consumption is shown along the number of items processed by this process.

With all the processes completed, the supply chain ends. By having access to the information inside each process, an organization can see how much units of the final product are produced by starting with 300 units of cotton.

The other benefit is that the electrical consumption and water usage associated with each process is visible to that organization, which in turn can help make strategies to reduce these two values.

By having access to this kind of information, an organization can use this Digital Twin solution to better prepare for the future regarding sustainability, by reducing the values of the electrical energy and the water usage.

The ideal scenario would be to have access information to real data of a company and simulate the data given by it. Then, two simulations could be done, one with the original data of this project that was taken from literature, the second one with the data from the company.

In the end, a direct comparison can be made between the simulations and the company could see if it was performing according to the values inside each process, if it was producing yarn, fabric and the final product below the levels defined in this project. If this was the case, the organization knows which processes are not being maximized in terms of performance, energy, and water consumption.

Chapter 7 Testing

Software testing is a method to check whether the actual software product matches expected requirements and to ensure that software product is defect free. The main goal of software testing is to identify errors, gaps or missing requirements in contrast to actual requirements, besides that if there any errors in the software, it can be identified early and can be solved before delivery of the software product (Hamilton Thomas, 2021).

For this project unit tests were performed into the three different applications that compose the system. For the user application, the registration and the token retrieval functionalities were tested, for the items application the authentication was tested as well as the responses returned when making a request, the same type of tests were applied to the processes application.

All the tests that were performed by the intern are displayed inside Appendix C.

7.1 Unit Testing

Unit testing is a type of software testing where individual units or components of a software are tested. The objective is to ensure that each unit of the software code works as expected. Unit tests help to fix bugs early in the development cycle and save costs; it helps the developers to understand the testing code base and enables them to make changes quickly; they can be used as project documentation (Hamilton Thomas, 2022).

For this part of the internship the intern used Pytest to perform the unit tests. Pytest is a testing framework that allows users to write codes using Python programming language. It helps writing simple and scalable test cases for databases, APIs. It is mainly used to write tests for APIs (Campbell, 2022).

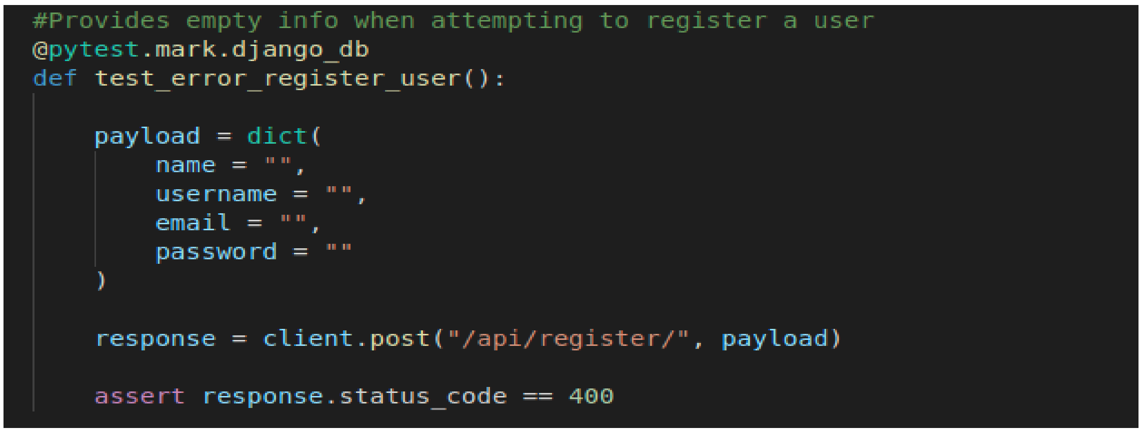
This framework was chosen because it is easy to learn and to use, provides functionalities to test the endpoints that were developed and uses the same language as the one used during the development stage.

The tests were divided into three separate folders, where each one of them represents one of the Django applications.

7.1.1 Users application tests

For this application, the tests created were meant to test the registration process and the retrieval of the authentication token using username and password. The registration process was tested with the following criteria:

- Right credentials filled in;
- Empty credentials filled in;
- Trying to register an already registered user



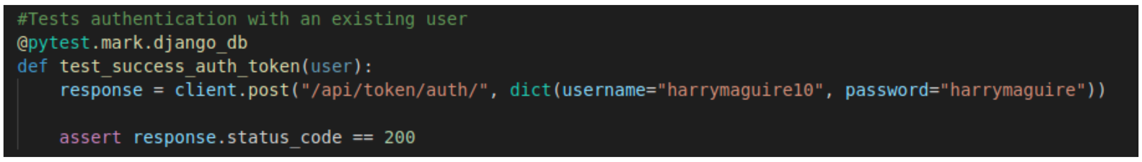
```
#Provides empty info when attempting to register a user
@pytest.mark.django_db
def test_error_register_user():

    payload = dict(
        name = "",
        username = "",
        email = "",
        password = ""
    )

    response = client.post("/api/register/", payload)

    assert response.status_code == 400
```

Figure 34: Registration test with empty fields

In **Figure 34**, the registration test with empty fields is displayed, the request is made to the "api/register" endpoint and the application should a return a 400 status code response to pass the test. It is important to mention that the data that is sent to the database is removed from it when the test is finished.

Then, the next step was testing the retrieval of the authentication token that is returned when the user performs the registration or when the "api/auth/token" endpoint is accessed. This last endpoint was tested with:

- Right credentials;
- Wrong credentials;
- Right username but wrong password;



```
#Tests authentication with an existing user
@pytest.mark.django_db
def test_success_auth_token(user):
    response = client.post("/api/token/auth/", dict(username="harrymaguire10", password="harrymaguire"))

    assert response.status_code == 200
```

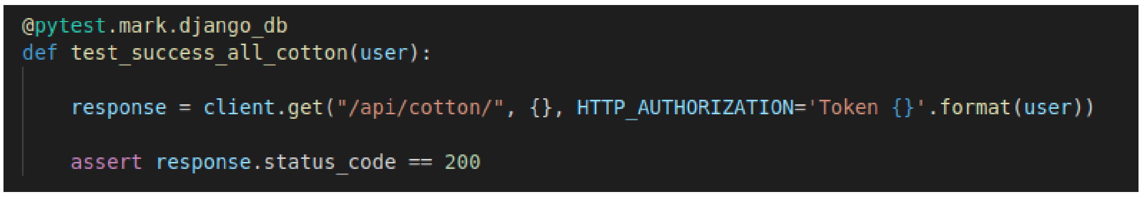
Figure 35: Authentication token retrieval with the right credentials

In **Figure 35**, the authentication token retrieval test is presented, where the client sends a request to the "api/token/auth" endpoint with the credentials of an already registered user, the test is expecting to receive a 200 status code response from the application.

7.1.2 Items application tests

The tests for this application were divided in two, one part of the tests refers to the authentication process of each endpoint and the other part is about validating the content that is returned from it.

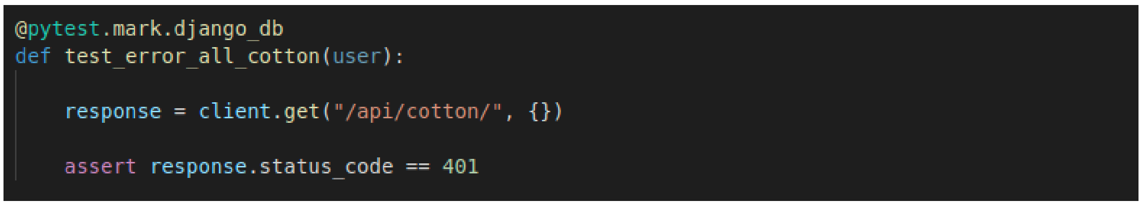
The first part of tests consisted in making requests from all the endpoints of the application with a valid authentication token inside the header of the request and no token. The goal was to receive a 200 status code response when the token sent was valid and a 401 response when there was no token to ensure that the authentication was working correctly.



```
@pytest.mark.django_db
def test_success_all_cotton(user):

    response = client.get("/api/cotton/", {}, HTTP_AUTHORIZATION='Token {}'.format(user))

    assert response.status_code == 200
```

Figure 36: Item authentication test with valid token

In **Figure 36**, the "api/cotton" endpoint was accessed, and the token sent inside the header of the request is valid, so the expected status code is 200 which represents a successful request.

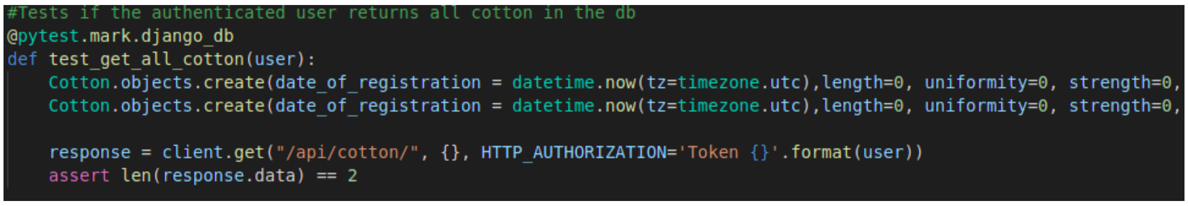


```python
@pytest.mark.django_db
def test_error_all_cotton(user):

    response = client.get("/api/cotton/", {})

    assert response.status_code == 401
```

Figure 37: Cotton authentication test with no token

In **Figure 37**, the request made to the “api/cotton” endpoint does not have an authentication token inside it, so the expected response code is 401, which represents Unauthorized access because the client did not provide the right credentials, in this case the authentication token.

For the second part of the tests, all the endpoints of the items application were used to validate their responses content. Each request made had a valid authentication token inside it, for it to be valid.



```python
#Tests if the authenticated user returns all cotton in the db
@pytest.mark.django_db
def test_get_all_cotton(user):
    Cotton.objects.create(date_of_registration = datetime.now(tz=timezone.utc),length=0, uniformity=0, strength=0,
    Cotton.objects.create(date_of_registration = datetime.now(tz=timezone.utc),length=0, uniformity=0, strength=0,

    response = client.get("/api/cotton/", {}, HTTP_AUTHORIZATION='Token {}'.format(user))
    assert len(response.data) == 2
```

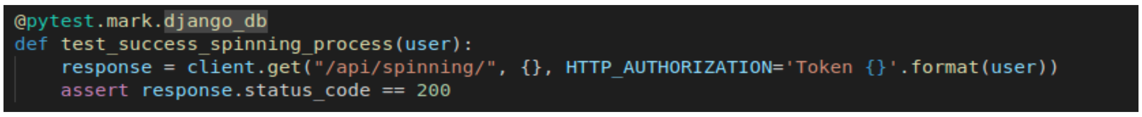
Figure 38: Cotton content validation

In **Figure 38**, two cotton units are created to fill the database and then the request is made to the “api/cotton/” endpoint with the goal of receiving a list of the cotton units registered inside the database. It is expected that the length of the list is two because they were created before the request was made, to pass the test.

7.1.3 Processes application tests

Like the items application tests, they were divided into two parts, one for the authentication of each endpoint and another to validate the content that is returned from the application.

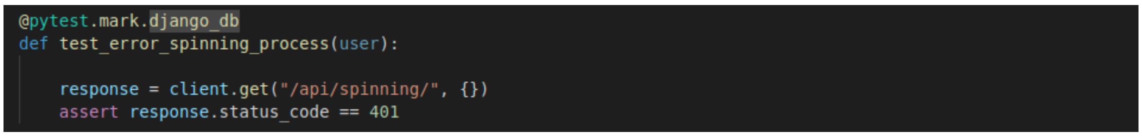
The first part of testing was performed by making requests to each endpoint one request with a valid authentication token and another with no authentication token. The goal was to observe a 200 status code response when using the valid token and a 401 response when using no token inside the request.



```
@pytest.mark.django_db
def test_success_spinning_process(user):
    response = client.get("/api/spinning/", {}, HTTP_AUTHORIZATION='Token {}'.format(user))
    assert response.status_code == 200
```

Figure 39: Spinning authentication with valid token

In **Figure 39**, the request is made to the "api/spinning/" endpoint with a valid authentication token in the request. Because a valid token is used the test is expecting a response of 200.

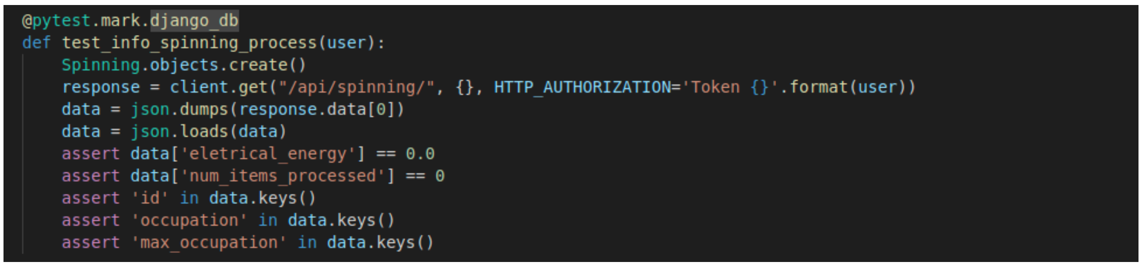


```
@pytest.mark.django_db
def test_error_spinning_process(user):

    response = client.get("/api/spinning/", {})
    assert response.status_code == 401
```

Figure 40: Spinning authentication with no token

In **Figure 40**, the request is made to the "api/spinning/" endpoint with no token on the header of the request. Following the logic, it is expected that the application returns a 401 status code response, that represents unauthorized access to it.

The second part of tests is also split into two parts, the first one accesses a given process that has not been initiated, which that the values are the default ones. Instead, the second part consists of performing operations inside a given process, to change the default values. Both parts use a valid token to make the request valid.

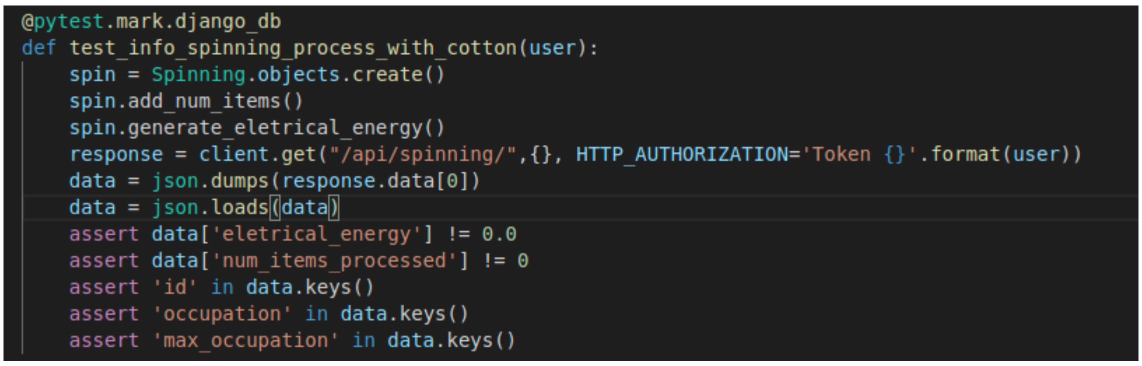


```
@pytest.mark.django_db
def test_info_spinning_process(user):
    Spinning.objects.create()
    response = client.get("/api/spinning/", {}, HTTP_AUTHORIZATION='Token {}'.format(user))
    data = json.dumps(response.data[0])
    data = json.loads(data)
    assert data['eletrical_energy'] == 0.0
    assert data['num_items_processed'] == 0
    assert 'id' in data.keys()
    assert 'occupation' in data.keys()
    assert 'max_occupation' in data.keys()
```

Figure 41: Empty spinning process content validation

In **Figure 41**, a Spinning object is created and stored in the database, to fill it. Then the request is made to "api/spinning/" endpoint and the response expected is the

Spinning process that was created before making the request. To test if the content is valid, five assertions are made to assure that the content inside the process is correct.



```python
@pytest.mark.django_db
def test_info_spinning_process_with_cotton(user):
    spin = Spinning.objects.create()
    spin.add_num_items()
    spin.generate_eletrical_energy()
    response = client.get("/api/spinning/",{}, HTTP_AUTHORIZATION='Token {}'.format(user))
    data = json.dumps(response.data[0])
    data = json.loads(data)
    assert data['eletrical_energy'] != 0.0
    assert data['num_items_processed'] != 0
    assert 'id' in data.keys()
    assert 'occupation' in data.keys()
    assert 'max_occupation' in data.keys()
```

Figure 42: Initiated spinning process content validation

In **Figure 42**, the request made is the same as the one in **Figure 41**, but with one difference, the values of 'eletrical_energy' and 'num_items_processed' have been altered after creating the Spinning process. It is expected that the default values of 'eletrical_energy' and 'num_items_processed' are different than zero, because they have been initiated before the making of the request. The other processes are tested following the same logic, and all the tests can be found inside the Appendix C.

7.2 Results and Final Considerations

In total sixty-one tests were performed divided by the three applications and all of them passed. After performing all tests, the code coverage of them was calculated.

Code coverage is a metric that can help understand how much of the source code is tested. Most coverage reports have information about the actual parts of the code that were not covered by tests and then use that to identify critical parts of the application that need to be tested (Pittet, 2002).

Having a high percentage of coverage does not guarantee that the code is flawless. Using Coverage.py, which is a tool for measuring code coverage of Python programs, the value calculated was of 94%. The coverage percentage should not be taken as an absolute figure because it only means that almost all the source code was tested, but it does not assure about the quality of the tests executed.

For this internship, the goal was to validate all the endpoints developed, to meet the requirements defined previously. After the testing was over, it was assured that all functionalities defined in the Requirements Chapter were sufficiently tested and that they could perform all their actions correctly.

Chapter 8
Adding Use Case Scenarios

In this Chapter, it will be displayed how the current Digital Twin implementation could be applied to other areas of business. The principle is that if a given field has a supply incorporated, then this Digital Twin can be adapted to fit that specific supply chain.

8.1 Why is it Possible?

It is possible to adapt the current Digital Twin solution to any business that makes use of a supply chain because all supply chains follow the same logic independently of the product or service produced. It will always have inventory management, production, and transportation modules.

To facilitate this process of adaptation, the data models used by the intern use inheritance for the items and process, this way when adjusting to a different business model, the developers only must change this data models attributes and adapt the ones that are inherited from them. In terms of development, as existing code is reused, it leads to less implementation and maintenance costs, the base code will be already tested and debugged.

Another thing that must be changed when dealing with a new supply chain is the energy values that are spent along with the water usage and other forms of energy that are used in each process from the new supply chain. The conversion rates if they exist must be also changed to fit the supply chain products.

8.2 Bread Supply Chain

In this section, an example of a bread supply chain is presented with the goal of showing how in reality it is possible to transform a Digital Twin that handles a cotton supply chain into one that simulates a bread supply chain.

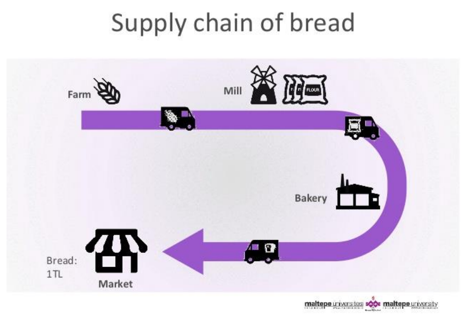


Figure 43: Bread Supply Chain

In **Figure 43**, the bread supply chain is displayed, there the processes, inputs and outputs of this specific supply chain are shown. Relatively to processes of production there is the process that transforms wheat, which is the input of the system, into flour, then that flour is transported to the bakery plant, where it bakes the flour into bread, which corresponds to the final product. Excluding the transportation components of the supply chain, just by having this information it is possible to transfer it to the current Digital Twin model.

To have the Digital Twin work to its fully capacity, the information about the conversion rates must be known, for example one kilogram of wheat generates how many kilograms of flour, and finally one kilogram of flour bakes how many kilograms of bread. Then the information about the energy and water consumed within each process must be known too, to have the sustainability functionality working.

With this information, all the benefits of the Digital Twin would still be present such as the simulation of the whole supply chain, the costs associated with the production of a given number of breads, the number of breads produced by having a given number of initial wheats.

8.3 Furniture Supply Chain

In this subsection, it will be explained how the furniture supply can be applied to the current Digital Twin solution.

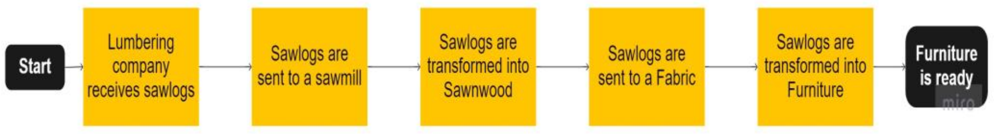

Start
Lumbering company receives sawlogs
Sawlogs are sent to a sawmill
Sawlogs are transformed into Sawnwood
Sawlogs are sent to a Fabric
Sawlogs are transformed into Furniture
Furniture is ready

Figure 44: Furniture Supply Chain

In **Figure 44**, the furniture supply chain is displayed, and it is visible that the input of the whole chain is saw logs. These inputs are transported into a Sawmill, where they are transformed into sawn wood, then they are moved to the factory, where they are transformed into furniture, which matches the final product of the supply chain.

In this case, there are three items that would be registered into the data model of the Digital Twin and two processes that would also be registered. Then, it must be known the conversion rates between saw logs and sawn wood, and sawn wood and furniture.

Finally, the energy and water consumed of each process must established previously. With all this information, it will be achievable to alter the current solution to match the furniture supply chain.

8.4 Modifications to the Digital Twin Core

In this section, the necessary steps to adapt any supply chain to this Digital Twin solution are presented.

- Change the items model to fit any type of product or input of the new supply chain;
- Change the processes model to adapt them to the new supply chain ones;
- Change the conversion rates;
- Change the consumption of water and electrical energy.

By following these steps, it is possible to migrate the current Digital Twin solution to fit any supply chain that matches the logic implemented in this one.

Chapter 9 Conclusion

This Chapter presents the conclusions drawn by the intern about the developed project and the whole internship. Additionally, some insight is given about the future of this project along with some features that would link well with what was developed until now.

9.1 Work Done

Considering, the main objectives behind this internship, the stakeholder considers it a success. All the defined requirements were implemented successfully, there is a backend architecture defined for the Digital Twin, a project ready to be used and a functionality that allows the calculation of energy and water consumption inside a given process.

Using the knowledge acquired during the writing of the state of art and with the guidance of the supervisor's from Ubiwhere the process of defining the requirements was natural. It started off by understanding how a real supply chain worked, with that information in mind the functional modules for the project were set up. The requirements were written in a way that follows the logic of a real supply chain.

While the requirements were being set up, the process of defining the architecture also began. The technologies for this internship were also chosen along with the architecture, taking into consideration that the main framework used by Ubiwhere was Django.

The development phase started by learning the technologies that were involved in this stage. After that, the intern already knew that the system would be divided into three different Django applications that represent the core of it. The first application developed was the Users one, where the registration and authentication components of the users were developed. The next stage was creating the items and processes applications, subsequently the Django models that have a direct connection to the PostgreSQL database were created.

Afterwards, the intern started to implement the actions to create the inputs for the system and the ones to send a given item to a specific process. After finishing all the actions, the entire Digital Twin was complete, and it was possible to start with a cotton fiber and end with a finalized product. The django admin application was used as a tool to see if the values generated by the command actions matched with what was supposed to occur and to manage the Digital Twin.

Regarding the testing part, the intern used unit testing as way to make sure that the developed code was performing as expected. The intern divided the tests into two parts, one for the authentication, to ensure that the quality attribute of security was present in this system, and other part to verify the content of the responses from the endpoints. In the end, all the sixty-one tests performed were passed and mostly of the developed code was tested by them. Regarding the quality attributes of the system all of them are ensured, the system is divided into three different applications which means that it is modular; it is portable because the environment choice is docker, which installs all the dependencies of the project automatically and runs it inside the docker container; the security attribute was verified with the unit tests performed.

The main challenge behind the internship was the lack of instantiation of the project, which meant that the intern had to research a specific supply chain to apply to this project, to draw the requirements and the architecture from that model.

The final product is a PoC of what can be achieved by the current Digital Twin solution, but it is important to mention that this solution can be adapted to any supply chain that follows the same logic as the one described in this work.

9.2 Future Work

The work performed at the internship was considered a success because all the functionalities that the intern planned to implement were completed, but there are some things that can be integrated with the current solution.

The current PoC can be tested with new clients to get feedback that can be pivotal for the project. The whole monolithic architecture can be transferred to a microservices one, where each Django application can have their own container. Then, the Digital Twin could be connected to a physical supply chain to have real data flow.

Another technology that can be utilized with this solution is Artificial Intelligence, with the goal of creating a functionality that allows to make predictions according to the data that is entering the system.

Additionally, a frontend application can be connected to this backend, so that the normal user has a better experience navigating through all the functionalities of the system. Finally, this solution can be integrated into other types of supply chain that involve the same logic as the one developed in this internship.

9.3 Lessons Learned

This internship proved to be a great experience for the intern, it allowed the learning of new technologies and solidify existing skills. The intern gained the knowledge and experience of what was like to develop a product from scratch, defining the requirements and the architecture that would fit best into the problem defined by the internship. Furthermore, the intern had the opportunity to work with tests, which was another area that he had no experience with.

The intern had full autonomy during all the internship and Ubiwhere always tried to point him towards the right direction, whenever the intern faced a problem.

The biggest takeaway from the internship is that planning the work before doing it makes a huge difference, having each sprint defined really helped to move the project forward. Another important aspect of the internship is that continuous communication with the company is vital to ensure that the quality of project developed is following a certain degree of quality.

Ubiwhere provided the right tools for the intern to progress during the whole internship, by always forcing him to think outside of the box, by being consistently ready to sort any doubt that the intern had.

References

11 Advantages of Django: Why You Should Use It – Pythonista Planet. (n.d.). Retrieved December 29, 2021, from https://pythonistaplanet.com/advantages-of-django/

Abideen, A. Z., Pandiyan, V., Sundram, K., Pyeman, J., Othman, A. K., & Sorooshian, S. (2021). Digital Twin Integrated Reinforced Learning in Supply Chain and Logistics. *Logistics 2021, Vol. 5, Page 84*, *5*(4), 84. https://doi.org/10.3390/LOGISTICS5040084

Atlassian. (2020). *What is Agile? | Atlassian*. Atlassian. https://www.atlassian.com/agile

Digital Twins – Modeling and Simulations | Microsoft Azure, (testimony of Azure Digital Twins). Retrieved November 15, 2021, from https://azure.microsoft.com/en-us/services/digital-twins/#features

Beamon, B. M. (1998). Supply chain design and analysis:: Models and methods. *International Journal of Production Economics*, *55*(3), 281–294. https://doi.org/10.1016/S0925-5273(98)00079-6

Better Cotton Initiative. (2020). *Measuring Cotton Consumption: BCI Conversion Factors and Multipliers Better Cotton Initiative*. *October*, 1–24.

Björnsson, B., Borrebaeck, C., Elander, N., Gasslander, T., Gawel, D. R., Gustafsson, M., Jörnsten, R., Lee, E. J., Li, X., Lilja, S., Martínez-Enguita, D., Matussek, A., Sandström, P., Schäfer, S., Stenmarker, M., Sun, X. F., Sysoev, O., Zhang, H., & Benson, M. (2019). Digital twins to personalize medicine. In *Genome Medicine* (Vol. 12, Issue 1). BioMed Central Ltd. https://doi.org/10.1186/s13073-019-0701-3

Blockhuys, M. (2019). *Monolith vs Microservices: Choosing the Right Architecture for Your Organization*. SQLI Digital Experience. https://www.sqli.nl/en/blog/monolith-vs-microservices

Blomkvist, Y., & Loenbom, L. U. (2020). *Improving supply chain visibility within logistics by implementing a Digital Twin A case study at Scania Logistics*.

Brink, B. (n.d.). *Digital twins and sustainability challenges*. Retrieved February 25, 2022, from https://global.royalhaskoningdhv.com/digital/resources/blogs/overcoming-sustainability-challenges-with-digital-twins

Brown, S. (2020). *The C4 model for visualising software architecture*. Infoq.Com. https://c4model.com/

Build powerful, targeted IoT applications with a packaged solution. (2019). www.siemens.com/mindsphere

Campbell, S. (2022). *PyTest Tutorial: What is, How to Install, Framework, Assertions*. https://www.guru99.com/pytest-tutorial.html

Carey, B. (2021). *Digital Twins Drive Sustainability in Oil and Gas Operations*. Journal of Petroleum Technology. https://jpt.spe.org/digital-twins-drive-sustainability-in-oil-and-gas-operations

Caridi, M., Moretto, A., Perego, A., & Tumino, A. (2014). The benefits of supply chain visibility: A value assessment model. *International Journal of Production Economics, 151*, 1–19. https://doi.org/10.1016/j.ijpe.2013.12.025

Dave, A. (2020). *How Digital Twins Accelerate the Growth of IoT*. https://www.iotforall.com/how-digital-twins-accelerate-the-growth-of-iot

Digitalization in the automotive industry | Industry | Siemens Global. (n.d.). Retrieved November 15, 2021, from https://new.siemens.com/global/en/company/stories/industry/getting-to-market-quickly.html

Django. (n.d.). *Customizing authentication in Django | Django documentation*. Retrieved April 21, 2022, from https://docs.djangoproject.com/en/4.0/topics/auth/customizing/

Django. (2021a). *Migrations | Django documentation | Django*. https://docs.djangoproject.com/en/4.0/topics/migrations/

Django. (2021b). *Writing views | Django documentation | Django*. https://docs.djangoproject.com/en/4.0/topics/http/views/

Django Software Foundation. (2018). The Web framework for perfectionists with deadlines | Django. In *Django Software Foundation*. https://www.djangoproject.com/

Dockerfile reference | Docker Documentation. (n.d.). Retrieved April 4, 2022, from https://docs.docker.com/engine/reference/builder/

Dohrmann, K., Gesing, B., & Ward, J. (2019). *Digital Twins in Logistics*. 1–39. https://www.logistics.dhl/content/dam/dhl/global/core/documents/pdf/glo-core-digital-twins-in-logistics.pdf?j=330813&sfmc_sub=108058070&l=59_HTML&u=20126700&mid=7275327&jb=69

Easy Redmine vs Redmine | What are the differences? (n.d.). Retrieved December 27, 2021, from https://stackshare.io/stackups/easy-redmine-vs-redmine

Erikstad, S. O., & Ove, S. (2017). *High-Performance Marine Vehicles*. https://www.predix.io.

European Commission. (2021). Destination Earth. Shaping Europe’s digital future. *Https://Digital-Strategy.Ec.Europa.Eu/En/Policies/Destination-Earth*. https://digital-strategy.ec.europa.eu/en/policies/destination-earth

Fuller, A., Fan, Z., Day, C., & Barlow, C. (2020). Digital Twin: Enabling Technologies, Challenges and Open Research. *IEEE Access*, *8*, 108952–108971. https://doi.org/10.1109/ACCESS.2020.2998358

GeeksforGeeks. (2020). Monolithic vs Microservices architecture - GeeksforGeeks. *Geeksforgeeks*. https://www.geeksforgeeks.org/monolithic-vs-microservices-architecture/

Google. (2021). *How to Use Google Meet Video Conferencing | Google Meet*. Google. https://apps.google.com/intl/en/meet/how-it-works

Grant, M. (2016). *Sustainability Definition*. Defintion Sustainability. https://www.investopedia.com/terms/s/sustainability.asp%0Ahttps://www.investopedia.com/terms/s/sustainability.asp%0Ahttps://www.investopedia.com/terms/s/sustainability.asp%0Ahttp://www.dictionary.com/browse/sustainability

Haag, S., & Anderl, R. (2018). Digital twin – Proof of concept. *Manufacturing Letters*, *15*, 64–66. https://doi.org/10.1016/J.MFGLET.2018.02.006

Hamilton Thomas. (n.d.). *What is Software Testing? Definition, Basics & Types in Software Engineering*. 2021. Retrieved June 15, 2022, from https://www.guru99.com/software-testing-introduction-importance.html

Hamilton Thomas. (2022). *Unit Testing Tutorial: What is, Types, Tools & Test EXAMPLE*. Guru99. https://www.guru99.com/unit-testing-guide.html

Holzner, S. (2007). *Beginning Ruby on Rails*. 380.

Hurtado, J. (2020). *How digital twins are helping repurpose sustainability values across different sectors - PreScouter - Custom Intelligence from a Global Network of Experts*. PreScouter. https://www.prescouter.com/2020/07/how-digital-twins-are-helping-repurpose-sustainability-values/

Information as a Service | Times of Cloud. (n.d.). Retrieved November 15, 2021, from https://timesofcloud.com/cloud-tutorial/information-as-a-service/

I͡Uzhno-Uralʹskiĭ gosudarstvennyĭ universitet, & Institute of Electrical and Electronics Engineers. (2020). *Proceedings, 2020 Global Smart Industry Conference (GloSIC) : South Ural State University (national research university), Chelyabinsk, Russian Federation, November 17-19, 2020.*

Jee, C. (2021). *Modularization in Software Engineering | by Caitlin Jee | Medium*. https://medium.com/@caitlinjeespn/modularization-in-software-engineering-1af52807ceed

K. L. Lueth. (2018). State of the iot 2018: Number of iot devices now at 7b Ð market accelerating. *IOT Analytics*. https://iot-analytics.com/state-of-the-iot-update-q1-q2-2018-number-of-iot-devices-now-7b/

Kaplan, E., & Koç, E. (2010). Investigation of energy consumption in yarn production with special reference to open-end rotor spinning. *Fibres and Textiles in Eastern Europe*, *79*(2), 7–13.

Keith-Magee, R. (n.d.). *Serializers - Django REST framework*. Retrieved April 21, 2022, from https://www.django-rest-framework.org/api-guide/serializers/

Koç, E., & Çinçik, E. (2010). Analysis of energy consumption in woven fabric production. *Fibres and Textiles in Eastern Europe*, *79*(2), 14–20.

Martin, M. (2021). *What is a Functional Requirement in Software Engineering? Specification, Types, Examples*. Guru99. https://www.guru99.com/functional-requirement-specification-example.html

Mccarthy, J. (2007). *WHAT IS ARTIFICIAL INTELLIGENCE?* http://www-formal.stanford.edu/jmc/

Microsoft. (2020). *What is PaaS? Platform as a Service | Microsoft Azure*. Microsoft Azure. https://azure.microsoft.com/en-us/overview/what-is-paas/

Microsoft. (2021). *What is SaaS? Software as a Service | Microsoft Azure*. Microsoft. https://azure.microsoft.com/en-in/overview/what-is-saas/#overview

Miro. (2020). *What Is Miro? – Miro Support & Help Center*. Miro/Healp Center. https://help.miro.com/hc/en-us/articles/360017730533-What-is-Miro-

Momjian, B. (2001). *PostgreSQL : introduction and concepts*. 461.

Moshood, T. D., Nawanir, G., Sorooshian, S., & Okfalisa, O. (2021). Digital twins driven supply chain visibility within logistics: A new paradigm for future logistics. In *Applied System Innovation* (Vol. 4, Issue 2). MDPI AG. https://doi.org/10.3390/asi4020029

MySQL. (2020). History of MySQL. In *MySQL 5.5 Reference Manual*. https://dev.mysql.com/doc/refman/8.0/en/history.html

Overview of Docker Compose | Docker Documentation. (n.d.). Retrieved April 4, 2022, from https://docs.docker.com/compose/

Palamutcu, S. (2010). Electric energy consumption in the cotton textile processing stages. *Energy*, *35*(7), 2945–2952. https://doi.org/10.1016/j.energy.2010.03.029

Pittet, S. (2002). *Introduction to code verification* (pp. 1–5). https://doi.org/10.1201/9781420035421.ch1

Power Digital Solutions, G. (2016). *GE Power Digital Solutions GE Digital Twin*.

Rails Guides Team. (2015). *Getting Started with Rails — Ruby on Rails Guides*. https://guides.rubyonrails.org/getting_started.html

Romero, G. (2021). *What is Postman API Test*. Encora. https://www.encora.com/insights/what-is-postman-api-test

Scheibmeir, J., & Malaiya, Y. (2019). An API Development Model for Digital Twins. *Proceedings - Companion of the 19th IEEE International Conference on Software Quality, Reliability and Security, QRS-C 2019*, 518–519. https://doi.org/10.1109/QRS-C.2019.00103

Shaw, K., & Fruhlinger, J. (2019). *What is a digital twin and why it's important to IoT | Network World*. https://www.networkworld.com/article/3280225/what-is-digital-twin-technology-and-why-it-matters.html

Silva, A. (2017). *How to Write Meaningful Quality Attributes for Software Development*. https://www.codementor.io/@antoniopfesilva/how-to-write-meaningful-quality-attributes-for-software-development-ez8y90wyo

Slaper, T. F. (n.d.). *The Triple Bottom Line: What Is It and How Does It Work?*

Smallcombe, M. (2020). *PostgreSQL vs MySQL: The Critical Differences*. Xplenty. https://www.integrate.io/blog/postgresql-vs-mysql-which-one-is-better-for-your-use-case/

Souza, V., Cruz, R., Silva, W., Lins, S., & Lucena, V. (2019). A Digital Twin Architecture Based on the Industrial Internet of Things Technologies. *2019 IEEE International Conference on Consumer Electronics, ICCE 2019*. https://doi.org/10.1109/ICCE.2019.8662081

Statista. (n.d.). • *Number of IoT devices 2015-2025 | Statista*. Statista. Retrieved February 25, 2022, from https://www.statista.com/statistics/471264/iot-number-of-connected-devices-worldwide/

Stodder, D. (2016). *Software Development Risk Management Plan with Examples*. CAST. http://www.castsoftware.com/research-labs/software-development-risk-management-plan-with-examples

Syndicode. (2020). *12 software architecture quality attributes and their types*. https://syndicode.com/blog/12-software-architecture-quality-attributes/

Techopedia. (2011). *What is Portability? - Definition from Techopedia*. https://www.techopedia.com/definition/8921/portability

Textile Learner. (2020). *Textile Manufacturing Process*. https://www.textileinfomedia.com/blog/textile-manufacturing-process-with-flow-chart/

The MITRE Corporation. (2014). Risk impact assessment and prioritization. In *Systems engineering guide*. https://www.mitre.org/publications/systems-engineering-guide/acquisition-systems-engineering/risk-management/risk-impact-assessment-and-prioritization%0Awww.mitre.org

Thomas Hamilton. (2019). *Agile Methodology: What is Agile Software Development Model & Process in Testing?* Guru99. https://www.guru99.com/agile-scrum-extreme-testing.html

Ting, S.-H., Ng, Y.-J., & Neelam, M. (2021). *Neelam MahaLakshmi (2021) Aspects of Artificial Intelligence In*. https://www.researchgate.net/publication/358119068

Truent, S. Ll. (2018). *Cotton Fibers and its Properties - Textile School*. 1–7. https://www.textileschool.com/164/cotton-fibers-and-its-properties/

A Simple Explanation of the Triple Bottom Line | University of Wisconsin, Www.Sustain.Wisconsin.Edu (2019). https://sustain.wisconsin.edu/sustainability/triple-bottom-line/

US Cotton Fiber Chart. (2021). *Comparison of Fiber Properties*. *423*, 37615. https://www.cottoninc.com/cotton-production/quality/us-cotton-fiber-chart/ratings-of-fiber-properties/

ViewSets - Django REST framework. (n.d.). Retrieved May 10, 2022, from https://www.django-rest-framework.org/api-guide/viewsets/

Wei, H. L., & Wang, E. T. G. (2010). The strategic value of supply chain visibility: Increasing the ability to reconfigure. *European Journal of Information Systems*, *19*(2), 238–249. https://doi.org/10.1057/EJIS.2010.10

What can you do with Calendar? - Google Workspace Learning Center. (n.d.). Retrieved January 22, 2022, from https://support.google.com/a/users/answer/9302892?hl=en

What is a C4 Model? How to Make C4 Software Architecture Diagrams | Gliffy by Perforce. (n.d.). Retrieved January 13, 2022, from https://www.gliffy.com/blog/c4-model

What is a Use Case? (n.d.). Retrieved January 13, 2022, from https://searchsoftwarequality.techtarget.com/definition/use-case

What is an Application Programming Interface (API) | IBM. (n.d.). Retrieved November 15, 2021, from https://www.ibm.com/cloud/learn/api

What is Flask Python - Python Tutorial. (2010). https://pythonbasics.org/what-is-flask-python/

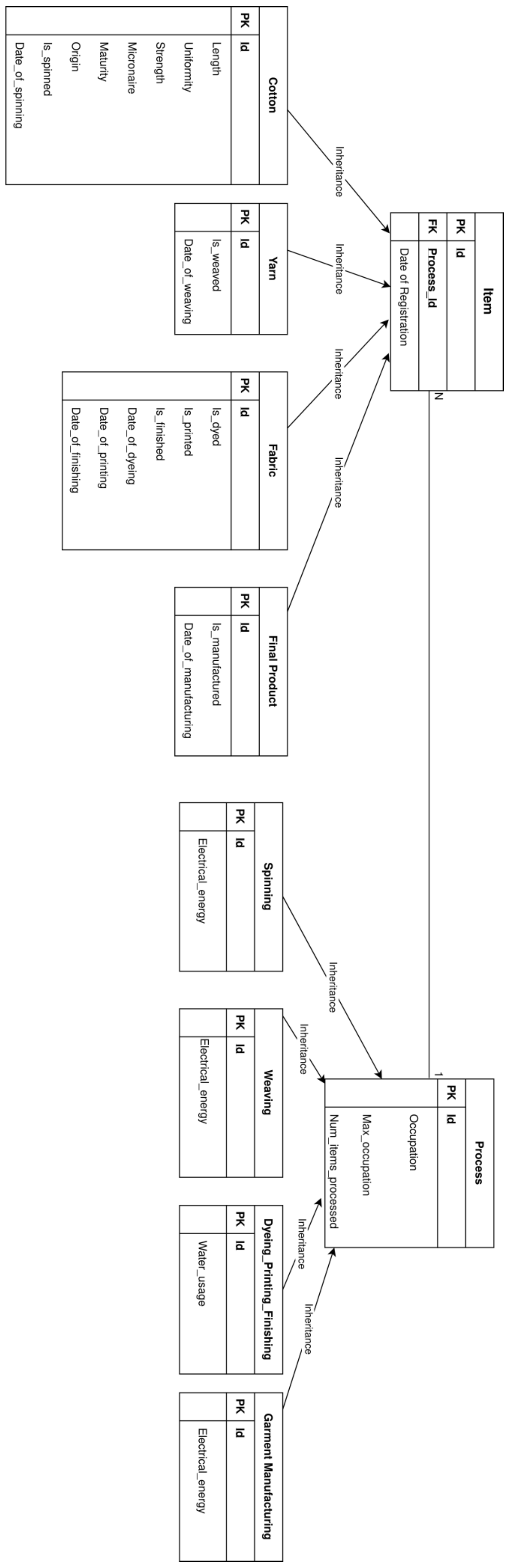
What is Laravel? | DigitalOcean. (n.d.). Retrieved February 25, 2022, from https://www.digitalocean.com/community/tutorials/what-is-laravel

What Is Software Architecture - Examples, Tools, & Design | CAST. (n.d.). Retrieved December 29, 2021, from https://www.castsoftware.com/glossary/what-is-software-architecture-tools-design-definition-explanation-best

Zheng, Y., Yang, S., & Cheng, H. (2019). An application framework of digital twin and its case study. *Journal of Ambient Intelligence and Humanized Computing*, *10*(3), 1141–1153. https://doi.org/10.1007/S12652-018-0911-3

Appendix A

Data Models


Cotton
PK Id
Length
Uniformity
Strength
Micronaire
Maturity
Origin
Is_spinned
Date_of_spinning
Yarn
PK Id
Is_weaved
Date_of_weaving
Fabric
PK Id
Is_dyed
Is_printed
Is_finished
Date_of_dyeing
Date_of_printing
Date_of_finishing
Final Product
PK Id
Is_manufactured
Date_of_manufacturing
Item
PK Id
FK Process_Id
Date of Registration
N
1
Inheritance
Spinning
PK Id
Electrical_energy
Weaving
PK Id
Electrical_energy
Dyeing_Printing_Finishing
PK Id
Water_usage
Garment Manufacturing
PK Id
Electrical_energy
Process
PK Id
Occupation
Max_occupation
Num_items_processed

Appendix B

Items Endpoints

Routes	**Purpose**
GET /api/cotton/	This endpoint is used for a registered user to access all cotton stored in the database;
GET /api/cotton/<int:pk>	This endpoint is used for a registered user to access information about a specific cotton stored in the database;
GET /api/cotton/cotton_spun/	This endpoint is used for a registered user to access all cotton that has passed the process of spinning, which is stored inside the database;
GET /api/cotton/cotton_not_spun/	This endpoint is used for a registered user to access all cotton that has not passed the process of spinning, which is stored inside the database;
GET /api/yarn/	This endpoint is used for a registered user to access all yarn stored inside the database;
GET /api/yarn/<int:pk>	This endpoint is used for a registered user to access information about a specific yarn stored in the database;
GET /api/yarn/yarn_weaved/	This endpoint is used for a registered user to access all yarn that has passed the process of weaving, which is stored inside the database;
GET /api/yarn/yarn_not_weaved/	This endpoint is used for a registered user to access all yarn that has not passed the process of weaving, which is stored inside the database;
GET /api/fabric/	This endpoint is used for a registered user to access all fabric stored in the database;

GET /api/fabric/<int:pk>	This endpoint is used for a registered user to access information about a specific fabric stored in the database;
GET /api/fabric/fabric_dyed/	This endpoint is used for the user to access all the fabrics that have been dyed;
GET /api/fabric/fabric_finished/	This endpoint is used for the user to access all the fabrics that have been finished;
GET /api/fabric/fabric_printed/	This endpoint is used for the user to access all the fabrics that have been printed;
GET /api/fabric/fabric_not_dyed/	This endpoint is used for the user to access all the fabrics that have not been dyed;
GET /api/product/	This endpoint is used for the user to access all the products;
GET /api/product/<int:pk>	This endpoint is used for the user to access a specific product;
GET /api/product/product_manufactured/	This endpoint is used for the user to access all products that have gone through all the stages of the supply chain;
GET /api/product/product_not_manufactured/	This endpoint is used for the user to access all products that have gone through all the stages of the supply chain except garment manufacturing;

Appendix C

Unit Testing

Table 31: Users application tests

Test Name	Purpose	Expected Output	Result
test_success_register_user()	Request to register a new user into the application with correct credentials	The new user is registered successfully	Pass
test_error_register_user()	Request to register a new user into the application with wrong credentials	The application returns a 400 status code	Pass
test_error_user_exists_register(user)	Request to register an already registered user	The application returns a 400 status code	Pass
test_success_auth_token(user)	Request to retrieve the authentication token by logging in with a registered user credentials	The application returns the authentication token	Pass
test_wrong_credentials_auth_token()	Request to retrieve the authentication token by logging in with a non-existing user credentials	The application returns a 400 status code	Pass
test_wrong_password_auth_token(user)	Request to retrieve the authentication token by logging in with the right username but wrong password	The application returns a 400 status code	Pass

Table 32: Items application authorizations tests

Test Name	Purpose	Expected Output	Result
test_success_all_cotton(user)	Request to test the authentication mechanism given a valid authentication token in the "/api/cotton/" endpoint	The application returns a 200 status code	Pass
test_error_all_cotton(user)	Request to test the authentication mechanism given an invalid authentication token in the "/api/cotton/" endpoint	The application returns a 401 status code	Pass
test_success_cotton_spun(user)	Request to test the authentication mechanism given a valid authentication token in the “api/cotton/cotton_spun/” endpoint	The application returns a 200 status code	Pass
test_error_cotton_spun(user)	Request to test the authentication mechanism given an invalid authentication token in the “api/cotton/cotton_spun” endpoint	The application returns a 401 status code	Pass
test_success_cotton_not_spun(user)	Request to test the authentication mechanism given a valid authentication token in the “api/cotton/cotton_not_spun/” endpoint	The application returns a 200 status code	Pass
test_error_cotton_not_spun(user)	Request to test the authentication mechanism given an invalid authentication token in the “api/cotton/cotton_not_spun” endpoint	The application returns a 401 status code	Pass
test_success_all_yarn(user)	Request to test the authentication mechanism given a valid authentication token in the “api/yarn/” endpoint	The application returns a 200 status code	Pass
test_error_all_yarn(user)	Request to test the authentication mechanism given an invalid authentication token in the “api/yarn/” endpoint	The application returns a 401 status code	Pass

test_success_yarn_weaved(user)	Request to test the authentication mechanism given a valid authentication token	The application returns a 200 status code	Pass
test_error_yarn_weaved(user)	Request to test the authentication mechanism given an invalid authentication token in the “api/yarn/yarn_weaved” endpoint	The application returns a 401 status code	Pass
test_success_yarn_not_weaved(user)	Request to test the authentication mechanism given a valid authentication token in the “api/yarn/yarn_not_weaved” endpoint	The application returns a 200 status code	Pass
test_error_yarn_not_weaved(user)	Request to test the authentication mechanism given an invalid authentication token in the “api/yarn/yarn_not_weaved” endpoint	The application returns a 401 status code	Pass
test_success_all_fabric(user)	Request to test the authentication mechanism given a valid authentication token in the “api/fabric/” endpoint	The application returns a 200 status code	Pass
test_error_all_fabric(user)	Request to test the authentication mechanism given an invalid authentication token in the “api/fabric/” endpoint	The application returns a 401 status code	Pass
test_success_fabric_dyed(user)	Request to test the authentication mechanism given a valid authentication token in the “api/fabric/fabric_dyed/” endpoint	The application returns a 200 status code	Pass
test_error_fabric_dyed(user)	Request to test the authentication mechanism given an invalid authentication token in the “api/fabric/fabric_dyed/” endpoint	The application returns a 401 status code	Pass

test_success_fabric_printed(user)	Request to test the authentication mechanism given a valid authentication token in the “api/fabric/fabric_printed/” endpoint	The application returns a 200 status code	Pass
test_error_fabric_printed(user)	Request to test the authentication mechanism given an invalid authentication token in the “api/fabric/fabric_printed” endpoint	The application returns a 401 status code	Pass
test_success_fabric_finished(user)	Request to test the authentication mechanism given a valid authentication token in the “api/fabric/fabric_finished” endpoint	The application returns a 200 status code	Pass
test_error_fabric_finished(user)	Request to test the authentication mechanism given an invalid authentication token in the “api/fabric/fabric_finished” endpoint	The application returns a 401 status code	Pass
test_success_fabric_not_dyed(user)	Request to test the authentication mechanism given a valid authentication token in the “api/fabric/fabric_not_dyed” endpoint	The application returns a 200 status code	Pass
test_error_fabric_not_dyed(user)	Request to test the authentication mechanism given an invalid authentication token in the “api/fabric/fabric_not_dyed” endpoint	The application returns a 401 status code	Pass
test_success_all_products(user)	Request to test the authentication mechanism given a valid authentication token in the “api/product/” endpoint	The application returns a 200 status code	Pass
test_error_all_products(user)	Request to test the authentication mechanism given an invalid authentication token in the “api/product” endpoint	The application returns a 401 status code	Pass
test_success_product_manufactured(user)	Request to test the authentication mechanism given a valid authentication token in the “api/product/product_manufactured” endpoint	The application returns a 200 status code	Pass

test_error_product_manufactured(user)	Request to test the authentication mechanism given an invalid authentication token in the "api/product/product_manufactured" endpoint	The application returns a 401 status code	Pass
test_success_product_not_manufactured(user)	Request to test the authentication mechanism given a valid authentication token in the "api/product/product_not_manufactured" endpoint	The application returns a 200 status code	Pass
test_error_product_not_manufactured(user)	Request to test the authentication mechanism given an invalid authentication token in the "api/product/product_not_manufactured" endpoint	The application returns a 401 status code	Pass

Table 33: Items application content tests

Test Name	Purpose	Expected Output	Result
test_get_all_cotton(user)	Request to retrieve all cotton registered in the database with a valid authentication token	A list of cotton is displayed	Pass
test_get_all_spun_cotton(user)	Request to retrieve all spun cotton inside the database with a valid authentication token	A list of spun cotton is shown	Pass
test_get_all_not_spun_cotton(user)	Request to retrieve all cotton that was not spun inside the database with a valid authentication token	A list of not spun cotton is shown	Pass
test_get_all_yarn(user)	Request to retrieve all yarn registered in the database with a valid authentication token	A list of yarn is displayed	Pass

test_get_all_weaved_yarn(user)	Request to retrieve all weaved yarn inside the database with a valid authentication token	A list of weaved yarn is shown	Pass
test_get_all_non_weaved_yarn(user)	Request to retrieve all yarn that was not weaved inside the database with a valid authentication token	A list of non weaved yarn is shown	Pass
test_get_all_fabric(user)	Request to retrieve all fabric registered inside the database with a valid authentication token	A list of fabric is displayed	Pass
test_get_all_dyed_fabric(user)	Request to retrieve all fabric that has been dyed, with a valid authentication token	A list of dyed fabric is presented	Pass

test_get_all_printed_fabric(user)	Request to retrieve all fabric that has been printed, with a valid authentication token	A list of printed fabric is presented	Pass
test_get_all_finished_fabric(user)	Request to retrieve all fabric that has been finished, with a valid authentication token	A list of finished fabric is presented	Pass
test_get_all_non_dyed_fabric(user)	Request to retrieve all fabric that has not been dyed, with a valid authentication token	A list of non dyed fabric is shown	Pass
test_get_all_product(user)	Request to retrieve all products, with a valid authentication token	A list of products is shown	Pass
test_get_all_manufactured_product(user)	Request to retrieve all manufactured products, with a valid authentication token	A list of manufactured products is displayed	Pass

test_get_all_non_manufactured_product(user)	Request to retrieve all non manufactured products, with a valid authentication token	A list of non manufactured products is presented	Pass

Table 34: Processes application authorization tests

Test Name	Purpose	Expected Output	Result
test_success_spinning_process(user)	Request to test the authentication mechanism given a valid authentication token in the “api/spinning” endpoint	The application returns a 200 status code	Pass
test_error_spinning_process(user)	Request to test the authentication mechanism given an invalid authentication token in the “api/spinning” endpoint	The application returns a 401 status code	Pass
test_success_weaving_process(user)	Request to test the authentication mechanism given a valid authentication token in the “api/weaving” endpoint	The application returns a 200 status code	Pass
test_error_weaving_process(user)	Request to test the authentication mechanism given an invalid authentication token in the “api/weaving” endpoint	The application returns a 401 status code	Pass

test_sucess_dyeing_printing_finishing_process(user)	Request to test the authentication mechanism given a valid authentication token in the “api/dyeing” endpoint	The application returns a 200 status code	Pass
test_error_dyeing_printing_finishing_process(user)	Request to test the authentication mechanism given an invalid authentication token in the “api/dyeing” endpoint	The application returns a 401 status code	Pass
test_sucess_garment_manufacturing_process(user)	Request to test the authentication mechanism given a valid authentication token in the “api/garment_manufacturing” endpoint	The application returns a 200 status code	Pass
test_error_garment_manufacturing_process(user)	Request to test the authentication mechanism given an invalid authentication token in the “api/garment_manufacturing” endpoint	The application returns a 401 status code	Pass

Table 35: Processes application content tests

Test Name	**Purpose**	**Expected Output**	**Result**
test_info_spinning_process(user)	Request to access empty Spinning process with a valid authentication token	Empty Spinning process	Pass
test_info_spinning_process_with_cotton(user)	Request to access an initiated Spinning process with a valid authentication token	Initiated Spinning process	Pass
test_info_weaving_process(user)	Request to access empty Weaving process with a valid authentication token	Empty Weaving process	Pass
test_info_weaving_process_with_weaving(user)	Request to access an initiated Weaving process with a valid authentication token	Initiated Weaving process	Pass
test_info_dyeing_finishing_printing_process(user)	Request to access empty Dyeing_Printing_Finishing processes with a valid authentication token	Empty Dyeing_Printing_Finishing processes	Pass

test_info_dyeing_finishing_printing_process_with_fabric(user)	Request to access empty Dyeing_Printing_Finishing processes with a valid authentication token	Initiated Dyeing_Printing_Finishing processes	Pass
test_info_garment_manufacturing_process(user)	Request to access empty Garment Manufacturing process with a valid authentication token	Empty Garment Manufacturing process	Pass
test_info_garment_manufacturing_process_with_product(user)	Request to access an initiated Garment Manufacturing process with a valid authentication token	Initiated Garment Manufacturing process	Pass