university of groningen

faculty of science and engineering

Environmental Sustainability Calculator Service for Business Processes

Bogdan-Petrut Popescu

university of groningen

faculty of science and engineering

University of Groningen

Environmental Sustainability Calculator Service for Business Processes

Master's Thesis
To fulfill the requirements for the degree of
Master of Science in Computing Science
at the University of Groningen under the supervision of
D. (Dimka) Karastoyanova, Prof. (Bernoulli Institute, University of Groningen)
and
V. (Vasilios) Andrikopoulos, Prof. Dr. (Bernoulli Institute, University of Groningen)

Bogdan-Petrut Popescu (S5358809)

August 17, 2024

Contents

Acknowledgments

First and foremost, I extend my thanks to my thesis coordinator, Dimka Karastoyanova, for their guidance, encouragement, and insightful feedback throughout this research. I thank you for cooperating with me on this topic. Your expertise, advice and support have been essential in the completion of this work.

I owe my deepest gratitude to my partner and family for their support and belief in me. Their encouragement, love and support have been the foundation upon I was able to finish this project.

I would also like to extend a general thank you to all those who have supported me in various ways along the way. Your contributions are deeply appreciated.

Abstract

With the increasing pressure on businesses to adopt sustainable practices and reduce their environmental impact, there is a growing need for tools that enable accurate and real-time sustainability assessments in business process management (BPM) and in the corresponding Process Aware Information Systems (PAIS) for process automation. The complexity of managing environmental impacts in BPM systems necessitates solutions to ensure effective and efficient sustainability measurements and reporting. The current state-of-the-art does not provide fully-fledged general solutions with broad compatibility and automation. This research addresses this need by developing an advanced and interactive solution designed to assess the environmental footprint of business operations, more specifically, an environmental sustainability calculator designed to integrate with existing BPM systems, e.g. Camunda 7 and JBPM. The tool features modular architecture and real-time data processing to continuously monitor and calculate CO2 GHG emissions thus providing up-to-date and precise assessments. The research is evaluated using a proof-of-concept implementation for a use case that requires calculations of CO2 emissions based on fuel emission factors and user input activity data, enabling its regulatory compliance and flexibility. Designed with extensibility in mind, this project will be available for future enhancements. It aims not only to provide a powerful tool for immediate environmental impact analysis but also to serve as a starting point for subsequent research and development efforts.

List of abbreviations

- **BPM** - Business Process Management
- **BPMN** - Business Process Model and Notation
- **GHG** - Greenhouse Gas
- **FR** - Functional Requirement
- **NFR** - Non-Functional Requirement
- **CO_2** - Carbon Dioxide
- **HTTP** - HyperText Transfer Protocol
- **KJar** - Knowledge JAR (Java ARchive)
- **Stomp** - Streaming Text Oriented Messaging Protocol
- **JBPM** - Java Business Process Model
- **API** - Application Programming Interface
- **DD** - Design Decision
- **ISO** - International Organization for Standardization
- **IBM** - International Business Machines
- **BIM** - Building Information Modeling
- **LEED** - Leadership in Energy and Environmental Design
- **SQL** - Structured Query Language
- **IDE** - Integrated development environment

List of Tables

List of Figures

1 Introduction

The concept of sustainability has evolved from being a simple buzzword to becoming a very important component of contemporary business strategies. In the quest to address environmental concerns and ensure long-term viability, organizations are increasingly integrating sustainability into their core operations. One significant approach to achieving this integration is through the use of sustainability calculators as a service, supported by Green Business Process Management (Green BPM). This introduction outlines the fundamental principles of such sustainability calculators, their application in business, and the role of Green BPM in allowing sustainable practices.

Sustainable development, defined as meeting the needs of the present without compromising the ability of future generations to meet their own needs, has gained significant traction in the business world. Companies are increasingly recognizing the importance of incorporating sustainability into their core operations to mitigate environmental impact and promote long-term viability [1]. In addition, the urgency of addressing climate change and resource depletion has prompted businesses to adopt more sustainable practices, thereby integrating ecological considerations into their process management strategies [2].

Green BPM is a specialized area within Business Process Management (BPM) that emphasizes the ecological impact of business processes. It involves the systematic integration of environmental goals into BPM life cycle stages, aiming to reduce resource consumption, greenhouse gas emissions, and waste generation [3]. The concept of Green BPM extends traditional BPM by incorporating sustainability metrics and performance indicators, thus enabling organizations to evaluate and improve their ecological footprint [4].

Green BPM supports the creation of sustainable business processes through the application of various techniques and methods designed to enhance resource efficiency. These methods include process redesign, optimization, and continuous improvement, all aimed at achieving higher levels of environmental performance [5]. By leveraging Green BPM, companies can transform their operations to align with sustainability objectives, thereby contributing to broader environmental goals [1].

Sustainability calculators are tools designed to assess the environmental impact of business activities and guide decision-making towards more sustainable practices. These calculators consider multiple factors, including energy consumption, emissions, and resource use, to provide a comprehensive analysis of a process's ecological footprint [6]. The application of sustainability calculators in Green BPM allows businesses to quantify their environmental impact accurately and identify areas for improvement [7].

The integration of sustainability calculators into business processes enables organizations to make informed decisions that align with their sustainability goals. These tools facilitate the monitoring and reporting of environmental performance, thus promoting transparency and accountability in corporate sustainability initiatives [3]. By utilizing sustainability calculators, companies can enhance their strategic planning and operational efficiency, leading to more sustainable business practices [8].

The adoption of Green BPM and sustainability calculators offers several benefits to organizations. Firstly, these tools help businesses reduce their environmental impact by optimizing resource use and minimizing waste [9]. Secondly, they support regulatory compliance by providing accurate data for

environmental reporting and ensuring adherence to sustainability standards [2]. Additionally, Green BPM and sustainability calculators can enhance a company's reputation by demonstrating a commitment to environmental stewardship and social responsibility [10]. Furthermore, these tools can drive innovation by encouraging the development of new, sustainable business models and processes. The continuous improvement framework inherent in Green BPM fosters a culture of sustainability within organizations, leading to long-term ecological benefits and competitive advantage [11]. Overall, the integration of Green BPM and sustainability calculators into business operations represents a strategic approach to achieving sustainability and addressing global environmental challenges [2].

1.1 Motivation

This thesis is motivated by the pressing need to address climate change and reduce carbon footprints across industries. The development of a sustainability calculator aims to bridge the gap between business process management and environmental guarding. By creating a tool that accurately assesses the environmental footprint of business operations, this thesis seeks to contribute significantly to the field of Green Business Process Management. The calculator will aid in precise GHG emissions calculations, providing businesses with actionable insights to enhance their sustainability efforts. The potential impact of such a tool is immense, as it can drive more informed decision-making, promote sustainable practices, and ultimately contribute to global efforts to combat climate change[12].

My personal desire to help combat climate change and reduce carbon consumption is in addition, a driving factor behind this project. Witnessing the effects of climate change on our planet has fueled my commitment to finding solutions that mitigate environmental damage. I believe that through the application of technology and innovative approaches, we can create significant positive change. Developing a tool that helps businesses to monitor and manage their environmental impact aligns perfectly with my will for participating in sustainability and my aspiration to contribute meaningfully to the global struggle against climate change.

1.2 Research Questions

The research questions for this thesis focus on creating, using, and integrating a sustainability calculator for Green BPM. We want to identify the necessary components of the system, ensure the tool is accurate and reliable, and understand the challenges and opportunities during integration. The questions are:

- Q1: What is the state-of-the-art in the field of BPM sustainability calculators and what are the key components and metrics that should be included to effectively measure and manage the environmental impact of business processes?

  This question looks at what needs to be included in a good sustainability calculator. It covers:

  - Important Metrics: Figuring out which environmental metrics are essential for measuring sustainability.
  - Necessary Parts: Identifying the technological and methodological parts needed, like data collection tools, calculation methods, and reporting features.
  - Stakeholder Needs: Considering what different stakeholders want from the calculator.

- Q2: How can a sustainability calculator for Green BPM be designed to ensure accuracy and reliability in sustainability assessments across different industries and various scales?

  This question focuses on making sure the sustainability calculator gives consistent and reliable results. It includes:

  - Industry Adaptable: Seeing how the calculator can be adapted to different industries without losing accuracy.
  - Scalability: Finding ways to keep the tool reliable whether it's used by small businesses or large companies.
  - Accuracy Checks: Developing ways to check and confirm the accuracy of the sustainability assessments, like unit testing.

- Q3: What are the challenges and opportunities in integrating a sustainability calculator with existing BPM systems and workflows in organizations aiming to enhance their sustainability performance?

  This question examines the practical aspects of implementing the sustainability calculator within current business systems. It includes:

  - Integration Challenges: Identifying technical issues that may arise in integrating such project in a company's production environment.
  - Impact on Organizations: Evaluating how adding a sustainability calculator affects business operations, decision-making, and overall sustainability goals.

1.3 Thesis Outline

The thesis is structured to explore the development, implementation, and evaluation of a sustainability calculator for business processes in the context of business process management (BPM). The following outline provides an overview of each chapter and its primary focus:

- Introduction: This section highlights the importance of sustainable business practices and the role of a sustainability calculator. It continues with explaining the motivation behind the project. The chapter also outlines the main research questions and gives a brief overview of the thesis structure.

- Background Literature: It covers Green BPM and its role in integrating sustainability into business operations. The chapter also discusses greenhouse gas emissions, emission scopes, and emission factors, providing the necessary context for understanding the importance of accurate sustainability assessments. Real-time GHG emissions monitoring tools and techniques are also explored.

- Literature Review: This section goes into a detailed analysis of previous studies related to sustainability calculators and BPM. It describes the methods used to collect and analyze relevant literature, including database searches and snowballing techniques. The chapter compares different studies based on specific criteria such as emissions measured, technologies used, and BPM integration, identifying key findings and gaps that the current research aims to address.

- Approach: The approach section outlines the strategy for developing the sustainability calculator. It identifies the key stakeholders and their needs, and details both functional and non-functional requirements. This chapter also discusses all design decisions of the system, outlining the key properties and explores the use cases and the interaction design from a user perspective.

- Solution: provides a comprehensive overview of the system architecture. It describes the components of the sustainability calculator, including BPM engines, integration modules, the main application, and the front end. This chapter details the technologies used through software decisions, and how these components interact, supported by diagrams and descriptions of process flows. The implementation of key features and the use of various design patterns are explained. It also provides a step-by-step walk-through of the setup of the project, to facilitate easy deployment.

- Discussion: This section analyzes the results of the research questions. It evaluates the effectiveness of the sustainability calculator in measuring and managing the environmental impact of business processes. The chapter discusses how the system meets the functional and non-functional requirements, reflecting on the design decisions and their implications. This section also acknowledges the constraints and limitations encountered during the development and implementation of the sustainability calculator. It discusses potential areas where the system could be improved and the challenges that were faced, providing a balanced view of the research outcomes. This chapter, in addition, outlines potential directions for further research and development. It suggests enhancements to the sustainability calculator, such as incorporating additional emission factors, improving real-time data processing capabilities, and expanding the system's applicability to more industries and scales.

- Conclusion: It summarizes the main contributions of the thesis. It recaps the key findings and their significance in the context of Green BPM and sustainability. The chapter reflects on the research questions and discusses the broader impact of the sustainability calculator on business practices and environmental management.

2 Background Literature

2.1 Green BPM

Green Business Process Management (Green BPM) is an evolving discipline that integrates sustainability into traditional BPM practices, aiming to minimize the environmental impact of business operations. This integration considers ecological goals alongside traditional business objectives such as efficiency, cost, and quality. According to [13], the incorporation of green aspects into the BPM life cycle stages is critical, and the proposed concept of "Process Greenability" highlights how these green characteristics can be measured and managed effectively.

A systematic mapping study by [3] categorizes Green BPM research into five attributes: scope, disciplines, accountability, researchers, and quality control. This study emphasizes the need for a collaborative effort between academics and practitioners to advance Green BPM as a business-oriented discipline. Similarly, [14] identifies that while Green BPM follows a development path similar to conventional BPM, it requires more comprehensive research to go beyond the traditional business process lifecycle.

Empirical studies show that the adoption of Green BPM is influenced by various factors such as the organization's size, sector, and competitiveness. For instance, larger and more competitive organizations are more likely to adopt Green BPM practices, which in turn positively impacts their society-related performance [15]. Additionally, [1] demonstrates that BPM techniques can support the sustainability of business activities, contributing to a reduction in the environmental footprint through process optimization and energy-efficient IT infrastructures.

The necessity to integrate environmental considerations into business process management is underlined by [16], who propose a multidimensional framework for Green BPM. This framework combines insights from Green IS and BPM to close existing research gaps and foster sustainable practices within organizations. Additionally, [2] discuss the potential of Green BPM to enhance resource efficiency and sustainability by leveraging innovative BPM techniques. They stress the importance of developing new methods and approaches tailored specifically to the sustainability context, thereby enabling businesses to significantly reduce their environmental impact.

2.2 Greenhouse gas emissions

Greenhouse gas (GHG) emission factors quantify emissions per unit of activity, such as energy production or transportation. They are crucial for assessing human induced climate change and formulating mitigation strategies. Derived from empirical data and modelling, these factors are compiled in databases, aiding global emission tracking and policy development.

GHG emission factors are integral to developing emission reduction strategies. They enable the computation of emissions from specific activities, such as energy production or vehicle use, providing a foundation for policy development and business strategies that are aimed at reducing carbon footprints. Accurate emission factors are vital for tracking progress towards climate goals and for the implementation of green BPM practices [17].

For calculating greenhouse gas (GHG) emissions on a municipal level [18] follows a standard for

GHG emission inventories called the Global Protocol for Community-Scale Greenhouse Gas Emission Inventories (GPC), using 43 emission types. it focuses on the Carbon Track and Trace (CTT) project, to explore how BPM can support the city's efforts in managing GHG emission data. In the study of [19] it is discussed the importance of optimizing existing operations to reduce emissions and environmental impact. They highlight that while significant research has been dedicated to developing alternative energy sources, the optimization of existing processes remains fundamental. The authors propose using BPM technology to model and improve not only traditional business processes but also manufacturing and other "physical" processes, thus contributing significantly to the reduction of carbon footprints.

2.3 Emission Scopes

Green BPM emphasizes the importance of managing Scope 1, Scope 2, and Scope 3 emissions to comprehensively address a company's carbon footprint.

- Scope 1 Emissions refer to direct GHG emissions from sources that are owned or controlled by the company. These include emissions from company-owned vehicles, on-site fuel combustion, and industrial processes. Effective management of Scope 1 emissions involves optimizing processes to reduce fuel consumption and switching to low-carbon alternatives.

- Scope 2 Emissions encompass indirect GHG emissions from the consumption of purchased electricity, steam, heating, and cooling. Managing Scope 2 emissions involves improving energy efficiency and sourcing renewable energy. By investing in energy-efficient technologies and purchasing electricity from renewable sources, companies can significantly reduce their Scope 2 emissions.

- Scope 3 Emissions are all other indirect emissions that occur in a company's value chain. These include emissions from business travel, employee commuting, waste disposal, and the production and transportation of purchased goods and services.

According to [19], understanding and optimizing existing operations to reduce their emissions impact is fundamental to Green BPM. This includes addressing all three scopes of emissions. The effective process management and optimization provided by Green BPM are essential for achieving sustainability goals. Accurate measurement and reporting of Scope 1, Scope 2, and Scope 3 emissions are critical for developing effective carbon management strategies and meeting regulatory requirements.

2.4 Emission factors

Emission factors are coefficients that provide the average emissions of specific greenhouse gases per unit of activity, energy consumed, or quantity of material used. They are crucial in calculating the environmental impact of various processes, activities, or systems, particularly in estimating GHG emissions. Emission factors are essential in environmental science, policy-making, and business process management, especially when direct emissions measurements are not feasible or cost-effective. [20] uses emission factors to determine GHG emissions attributable to cities or city regions. [21] reviews emission factor standards worldwide for estimating emissions from construction equipment and develops a selection process for the Australian context. [12] developed a model that uses GHG emission factors to estimate and control the environmental performance of projects. This approach integrates emission factors into earned value management to monitor and predict GHG emissions

throughout the project lifecycle. [22] evaluated the use of Bayesian Networks to manage GHG emissions in the British agricultural sector. Their model incorporates emission factors to understand the impact of various farming activities on GHG emissions, providing a more accurate representation of environmental impacts. [23] applied the DNDC biogeochemistry model to estimate GHG emissions from Italian agricultural areas. This study used emission factors to simulate carbon and nitrogen cycles, demonstrating the model's ability to accurately reflect GHG emissions based on local agricultural practices. [24] compared tools for quantifying the environmental performance of urban territories, including emission factors in their analysis. Their study highlighted the importance of accurate emission factors for reliable environmental assessments at the municipal level.

2.5 Real-Time GHG Emissions Monitoring

The importance of real-time greenhouse gas (GHG) emissions monitoring has gained significant attention as businesses try to enhance their environmental sustainability. Real-time monitoring tools provide continuous, up-to-date data on emissions, enabling organizations to identify and address inefficiencies faster, reducing their overall carbon footprint. Several studies and tools have been developed to facilitate real-time GHG emissions monitoring, demonstrating its effectiveness in various sectors.

The research conducted by [25] establishes a thorough and efficient tool for continuously monitoring greenhouse gas emissions (GHG) and energy efficiency over the course of 34 operated sites in the exploration and production sector. This system of monitoring has tracked emissions from the most significant sources such as energy use, flaring, venting, and the burning of hydrocarbon fuel. Its real-time data has provided a comprehensive picture of emissions at various levels from parses such as the equipment individual to the entire sites and regions. The implementation of the tool has enhanced operational efficiency and decision-making for emissions reduction aspects significantly. In [26] the authors developed a GHG module within an existing real-time advanced analytic tool specifically for well operations. This tool collects high-frequency sensor data from drilling activities to track CO2 emissions associated with fuel consumption. The integration of this tool allows for continuous monitoring and provides insights to improve energy efficiency and reduce emissions during well construction. [12] introduced a model that integrates GHG emissions monitoring with earned value management (EVM) to assess the environmental performance of projects. This model utilizes emission factors to estimate and control GHG emissions throughout the project lifecycle, enabling continuous monitoring and prediction of emissions based on real-time data. The approach provides a structured method to track and manage the environmental impact of projects effectively. [27] developed Preserve, a low-cost, sensor-based solution for real-time emissions management in small and medium farms. This tool provides real-time measurements and predictive models to help farmers monitor and manage their GHG emissions effectively. The user-friendly cloud interface of Preserve makes it accessible and practical for a wide range of agricultural users, addressing the need for scalable and affordable emissions management solutions in the sector.

3 Literature review

3.1 Introduction

The literature review section provides an analysis of the existing research, tools, and methodologies relevant to the development of asustainability calculator for business process management. This section outlines the process of literature collection, the databases used, and a comparative analysis of the identified papers based on key factors.

3.2 Literature collection

The papers read for acquiring relevant knowledge were extracted using forward and backward snowballing techniques, and direct searches on a number of paper databases. This methodology was chosen because it aligns with the review's goal of identifying key factors efficiently. Additionally, since the thesis focuses more on developing a tool rather than extensive research, and considering the project's time constraints, a concise review was preferred over a more comprehensive one.

3.2.1 Snowballing

Snowballing [28] is an iterative method of literature review where the researcher begins with a set of primary papers and then explores the references cited in these papers to identify additional relevant studies. This method ensures a comprehensive and interconnected understanding of the research topic. The research process started from an initial set of systematic literature reviews on various Green BPM techniques, and followed forward and backward snowballing to ensure all significant relevant studies were considered.

3.2.2 Database searches

In addition to snowballing, direct searches were conducted in several academic databases to ensure coverage of the relevant literature. The databases used include:

- **IEEE Xplore**: For technical papers and conference proceedings on sustainability and BPM technologies.
- **Google Scholar**: For a broader search across multiple disciplines and to leverage its citation tracking features.
- **Smartcat**: An academic database that aggregates scholarly articles and research papers across various disciplines, providing comprehensive access to literature on sustainability and business process management
- **Consensus**: A specialized search engine for scientific literature that uses AI to provide insights and synthesize consensus from multiple studies.

3.2.3 Search Queries Used

To identify relevant literature, the following search queries were employed across the databases:

- Sustainability Calculators: ”sustainability calculator”, ”carbon footprint calculator”, ”environmental impact assessment tool”

- Green BPM: ”Green Business Process Management”, ”sustainable business processes”, ”BPM and sustainability”

- Emission Calculation Models: ”GHG emission models”, ”carbon emission calculation”, ”environmental impact modeling”

- Real-Time Data Processing: ”real-time data sustainability”, ”live data processing environmental”, ”real-time carbon footprint”

- Integration with BPM Systems: ”BPM integration sustainability”, ”sustainability tools business process management”, ”Green BPM integration”

3.2.4 Filtering and Selection of Papers

After conducting the initial searches, a filtering process was applied to ensure that only the most relevant papers were included. The following criteria and steps were used to filter the papers:

- First round of selection
  - Title relevance: The paper titles were reviewed to be related to the topic and to contain good topic’s keywords.
  - Publication Date: Preference was given to recent publications (last 15 years) to ensure up-to-date information.
  - Duplicates: Duplicate papers found across multiple databases were removed.

- Second round of selection
  - Content relevance: Abstracts and conclusions were reviewed to determine the relevance to sustainability calculators and Green BPM.
  - Methodological Correctness: Papers with robust and well-documented methodologies were prioritized.

- Third round of selection
  - Full-Text Review: The remaining papers were reviewed in full to assess their methodology, findings, and relevance to the research objectives.

The paper filtering process is shown in the figure below:

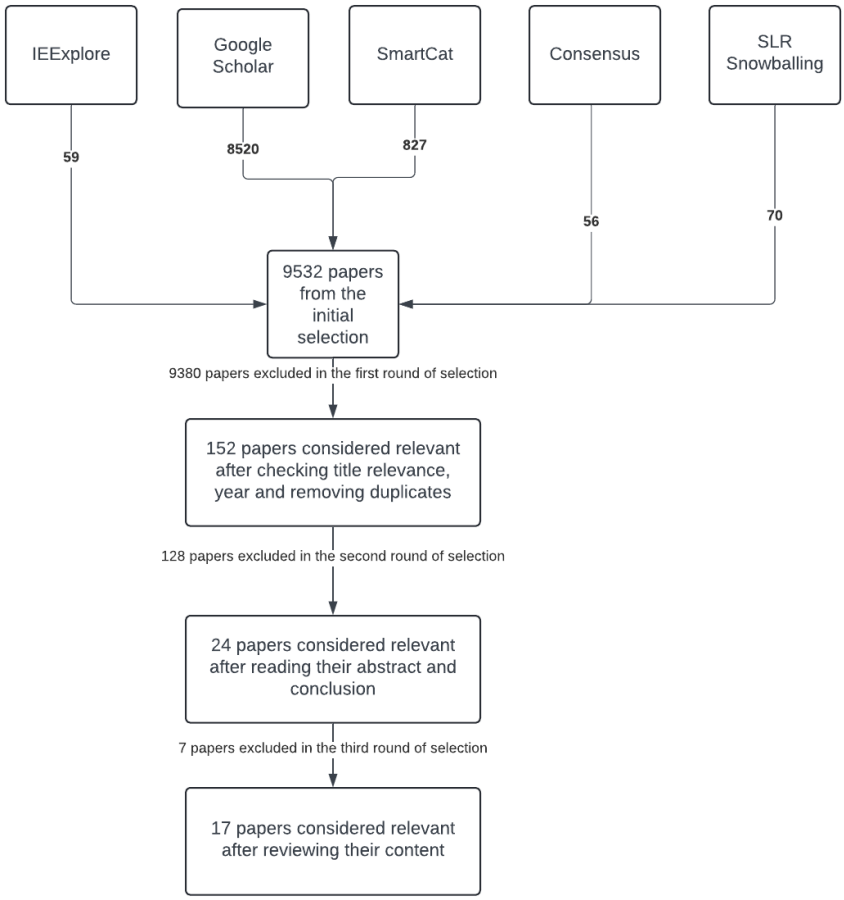


Figure 1: Paper filtering process

3.3 Comparative Analysis of Identified Papers

The identified papers were compared based on several key factors to assess their relevance and contributions to the field of sustainability calculators. The factors of comparison were the following:

- Specific Emissions Measured: This factor identifies what specific emissions are being measured, such as greenhouse gases (GHG) including CO2, methane, and other substances. It may also include broader environmental impact metrics.
- Technologies Used: This factor looks at the specific technologies, tools, or algorithms used in the emissions calculation and analysis process.
- Real-Time Monitoring: This factor indicates whether the research or software provides capabilities for real-time monitoring of emissions.
- BPM Integration: This factor assesses the extent to which Business Process Management (BPM) is integrated into the study or software
- Presents Software System: This factor indicates whether the study includes the development or use of a software system designed to assess and manage emissions. It highlights the practical application of research through software tools that facilitate sustainability assessments.

- Presents Software System Using BPM: This factor specifies whether the software system presented in the study utilizes a BPM engine.

Table 1 shows a comparison of a number of selected papers based on the factors described in the list above:

Table 1: Comparison of Studies on Sustainability Calculators Using BPMN

Paper Title	Specific Emissions Measured	Technologies/Tools Used	Real-Time Monitoring	BPM Integration	Presents Software System	Presents Software System Using BPM
Green business process management: A research agenda [19]	General GHG	Green BPM methodologies	No	Full BPM integration	No	No
Towards Green BPM - Sustainability and Resource Efficiency through Business Process Management [2]	CO2, general GHG	Green BPM methodologies	No	Full BPM integration	No	No
Building performance modelling for sustainable building design [8]	CO2, resource consumption	Building Information Modelling (BIM), Ecotect	No	BPM for construction processes	Yes	No
Advancing business process technology for humanity: Opportunities and challenges of green BPM for sustainable business activities [1]	CO2, general GHG	Green BPM methodologies	No	Full BPM integration	No	No
Approaching green BPM characterisation [13]	CO2, resource consumption, waste generation	PDCA (plan-do-check-act) model, Process Greenability characteristics	No	BPM lifecycle stages	No	No
Sustainability Calculator: A Tool to Assess Sustainability in Cosmetic Products [6]	GHG emissions, resource consumption	Microsoft Excel tool	No	Integration with product lifecycle management	Yes	No
A systematic review of green business process management [14]	CO2, methane, nitrous oxide	Systematic mapping study	No	Green BPM as a business-oriented discipline	No	No
Supporting Municipal Greenhouse Gas (GHG) emission inventories using business process modeling: a case study of Trondheim Municipality [18]	CO2, methane, nitrous oxide	BPM for GHG emission inventories	Yes	Integration with municipal BPM	No	No
BPMS-Game: Tool for Business Process Gamification [7]	General environmental impacts	BPMS-Game, gamification tools	No	BPM for promoting sustainability	Yes	Yes
SUSTAINABLE DEVELOPMENT AND BUSINESS PROCESS MANAGEMENT [9]	Energy consumption, pollution	Green BPM methodologies	No	Integration of environmental objectives within BPM	No	No
Environmental sustainability through green business process management [5]	CO2, general GHG	Green BPM, environmental performance indicators	Yes	Integration of environmental performance indicators	No	No
How to Incorporate Sustainability into Business Process Management Lifecycle [29]	Environmental, economic, social impacts	Comprehensive BPM approaches	No	Integration of sustainability in BPM lifecycle	No	No
Optimising building sustainability assessment using BIM [30]	CO2, resource consumption	BIM, SBToolPT-H	No	BPM for building sustainability assessments	Yes	No
Enterprise Integrated Business Process Management and Business Intelligence Framework for Business Process Sustainability [31]	Energy consumption, GHG emissions	BPM and Business Intelligence (BI)	No	Integrated BPM and BI framework	No	No
An empirical study on Green BPM adoption: Contextual factors and performance [15]	CO2, methane, nitrous oxide	BPM capabilities, contextual factors	No	Adoption of Green BPM in various organizational contexts	No	No
Integrated BIM-LEED application to automate sustainable design assessment [32]	CO2, resource consumption	BIM, LEED certification, K Nearest Neighbour (KNN)	Yes	Integration of BIM with green building certification systems	Yes	No
Modeling and analyzing the carbon footprint of business processes [33]	CO2	BPM methodologies	No	Full BPM integration	Yes	Yes

3.4 Literature review conclusion

There are a number of key insights we can draw from this comparison:

- **Focus on GHG emissions**: Most studies focus on measuring greenhouse gas (GHG) emissions, particularly CO2. This is consistent with the broader goal of reducing the carbon footprint of business processes.
- **Lack of Real-Time Monitoring**: Only a few studies incorporate real-time monitoring capabilities. Moreover, real-time data computation and monitoring are identified as critical for ongoing assessment and adjustment of emissions.
- **Lack of BPM software integration**: Several studies present software systems for sustainability assessment, but fewer integrate these systems with BPM engines. The integration of BPM engines is crucial for automating and managing business processes effectively.
- **Comprehensive BPM usage**: Some studies demonstrate comprehensive integration of BPM methodologies with sustainability goals.
- **Diverse Technologies and Tools**: The studies utilize a range of technologies and tools, including Building Information Modelling (BIM)[1], Ecotect [2], and Microsoft Excel[3].

These conclusions may help in making design decisions for the sustainability calculator developed in this thesis:

- **Real-Time Monitoring Integration**: Incorporating real-time monitoring capabilities will facilitate more dynamic and responsive management of emissions, allowing for real-time adjustments and improvements in sustainability practices.
- **Software Development with BPM Integration**: There is a clear need for a software system that integrates BPM engines. This integration will enable the full automation and management of sustainable business processes, driving more efficient and effective sustainability initiatives.
- **Enable Multi Platform Compatibility**: To accommodate the diverse range of technologies and tools used in sustainability assessments, the developed software system should be compatible with multiple technology stacks. This will enhance its applicability and adoption across various platforms and industries.

[1] https://en.wikipedia.org/wiki/Building_information_modeling

[2] https://www.autodesk.com/support/technical/article/caas/sfdcarticles/sfdcarticles/Ecotect-Analysis-and-Green-Building-Studio.html

[3] https://ro.wikipedia.org/wiki/Microsoft_Excel

4 Approach

4.1 Introduction

This section introduces the BPM sustainability calculator software system, outlining the design decisions (4.4), system requirements (4.3), and key stakeholders (4.2) involved.

4.2 Stakeholders

This section describes the various stakeholders of the system, and lists out their key concerns ordered from most significant to the least using the ISO/IEC 25010:2011 quality standard as a reference [34].

- **Enterprise Customer**: They are the customers of this system, willing to integrate it in their company's technology stack for better monitoring of their sustainability. Their concerns are:
  - Compatibility: The sustainability calculator needs to be compatible with the company's technology stack in order to be feasible for incorporation.
  - Scalability: The system must ensure similar performance when dealing with a variable amount of stress
  - Maintainability: The system must be modular, easily modifiable and testable.
- **User**: They are the people responsible for monitoring the sustainability of the company's process, and will directly interact with the application:
  - Usability: The system must be user-friendly to interface with, easy to learn and operate, and must provide some safety from user errors.
  - Adaptability: The system must be adequately generalized and modular to fit various edge cases that might appear during the regular usage.
  - Reliability: The system must have built-in fault tolerance and recovery mechanisms during regular operation.
- **Operations team**: They are the people responsible for keeping the company's products running, and they will be directly impacted by adding a new service:
  - Reliability: The system must have low fault rates and down-times to minimize resource utilization on maintenance.
  - Maintainability: The system must be modular, easily modifiable and testable.
  - Compatibility: The system needs to be compatible with the company's technology stack for easier maintenance and debugging in case of failure.

4.3 Requirements

In this subsection, the system requirements will be outlined. The comparative analysis of existing research and tools will be taken into consideration, in addition to other needs of the system, in order to have a relevant functionality and based on the gaps identified in the literature overview:

4.3.1 Functional requirements

FR-1	BPM integration
The system should communicate with a BPM engine	

FR-2	Real-time monitoring
The system should make and show real-time updates on sustainability metrics	

FR-3	User interface
The system should provide a dashboard that allows the users to receive information	

FR-4	Data Management
The software needs to include a robust database system for data storage	

FR-5	Emission Calculation
The system must accurately calculate GHG emissions per process instance and on the level of activities	

FR-6	Multi-Platform Compatibility
The system will be designed to support multiple BPM platforms	

FR-7	Communication
The system needs to ensure communication between modules	

FR-8	Automated Reporting
The system should generate automated sustainability reports based on predefined templates.	

4.3.2 Non-functional requirements

NFR-1	Scalability
The system should support horizontal scaling (adding more machines) and vertical scaling (upgrading existing machines).	

NFR-2	Usability
The interface should be intuitive and accessible, ensuring a good user experience	

NFR-3	Maintainability
The system must require no more than 1 hour of maintenance per year	

NFR-4	Adaptability
The system must be adequately generalized and modular to fit various edge cases that might appear	

NFR-5	Reliability
The system should have an uptime of 99.9%, ensuring high availability	

NFR-6	Interoperability
The system should be able to integrate with existing BPM tools and other relevant external systems.	

NFR-7	Portability
The system should be able to run on various operating systems	

NFR-8	Extensibility
The system should be designed to allow easy addition of new features and functionalities	

NFR-9	Data integrity
The system must ensure data accuracy and consistency	

4.4 Design decisions and alternatives

This section will describe major design decisions made for the project, together with why these design decisions were made. On top of that, this section will also describe alternatives to the decision, and the reason they were not chosen.

Code	DD-1
Name	Application running environment
Problem	The system needs to be running in a specific way.
Decision	Containers
Benefits	• Consistency: Provides a consistent environment across different stages of development, testing, and production, reducing the ”works on my machine” problem. • Isolation: Containers encapsulate the application and its dependencies, providing isolation. • Rapid Deployment: Containers can be started and stopped quickly, for rapid deployment.
Drawbacks	• Managing containers can become complex with increasing scale. • There is some performance overhead due to the additional abstraction layer.
Alternatives	• Virtual machines – VMs are more resource-intensive than containers because they include a full OS. – VMs take longer to start and stop compared to containers. – Requires more resources and effort to manage compared to containers. • JAR Files – Environment Consistency: Differences in OS and JVM configurations can cause inconsistencies. – Isolation: Lacks the process and file system isolation that containers provide, which can lead to resource conflicts. • Serverless Computing – Vendor Lock-In: Heavily dependent on the cloud provider’s ecosystem, leading to potential vendor lock-in. – : Execution Limits: Has execution time limits, which may not be suitable for long-running tasks.

Table 2: Design decision 1: Docker

Code	DD-2
Name	Data communication protocol
Problem	The system needs a protocol to communicate data over the networks.
Decision	TCP/IP.
Benefits	• Guarantees that data arrives (if the network is up), requiring less frequent sending of data. • Traffic management, allowing for better usage of the network's bandwidth. • Guaranteed correct sequence of data.
Drawbacks	• Overhead and Complexity, because of the three-way handshake • Handshake delays cause latency
Alternatives	• UDP. – Data is not guaranteed to arrive, so data needs to be sent more frequently • SCTP. – SCTP contains many features that TCP does not, which is not necessary for this project, unnecessarily increasing project complexity. • DCCP. – DCCP is highly optimised for timing-constrained workloads, which is not actually relevant to a sustainability calculator system. Therefore it is not worth the slight reduction in reliability compared to TCP.

Table 3: Design decision 2: TCP/IP communication

Code	DD-3
Name	Communication paradigm
Problem	The system needs a way to exchange data.
Decision	Mixed communication(syncronous/asyncronous).
Benefits	• Flexibility: Having the possibility to select between the 2 types of communication • Enhanced User Experience: Users benefit from immediate responses where necessary, while less critical operations can be handled in the background
Drawbacks	• Debugging and Maintenance: Troubleshooting issues can be more complicated due to the different nature of synchronous and asynchronous operations. • Complexity: Integrating both synchronous and asynchronous communication methods adds complexity to the system design and implementation.
Alternatives	• Fully Synchronous Communication. – Less scalable due to the blocking nature of synchronous operations. • Fully Asynchronous Communication. – Latency: Delays in processing can affect real-time responsiveness, making it unsuitable for tasks requiring real-time feedback.

Table 4: Design decision 3: Mixed communication

Code	DD-4
Name	BPM Engine integrations
Problem	The system needs to communicate with a BPM engine to extract information on the process execution
Decision	Multiple BPM engine integrations
Benefits	• Diverse Business Needs: Different systems may have varying process management requirements that are best met by different BPM engines. • Flexibility: Offers the ability to have process management solutions for the specific needs of various business units, enhancing overall efficiency.
Drawbacks	• Complexity: Building a sustainability calculator with multiple BPM engine compatibilities is complex, needing extra parsing and abstractisation.
Alternatives	• Single BPM Engine compatibility – Limited Flexibility: May not meet all specific needs of different business units or legacy systems.

Table 5: Design decision 4: Multiple BPM integrations

Code	DD-5
Name	Types of GHG emissions used in calculation
Problem	Which types of GHG emissions should be calculated by the system
Decision	CO2 emissions
Benefits	• Primary GHG: CO2 is the most significant GHG in terms of volume and impact on global warming, making it a critical target for reduction efforts. • Regulatory Compliance: Many regulations and reporting frameworks prioritize CO2 emissions[35], ensuring compliance is simpler. • Easier Data Collection: Data for CO2 emissions is often more readily available and easier to collect compared to other greenhouse gases (GHGs).
Drawbacks	• Other GHGs Ignored: Focusing only on CO2 emissions overlooks other significant GHGs like methane (CH4), nitrous oxide (N2O), and fluorinated gases, which can have a much higher global warming potential (GWP). • Misleading Results: A reduction in CO2 emissions might cause increases in other GHGs, leading to an incomplete assessment of the overall environmental impact.
Alternatives	• Calculating All GHG Emissions – Data Collection Challenges: Gathering accurate data for all GHGs can be more complex and resource-intensive. – Increased cost: Higher costs associated with comprehensive monitoring, data collection, and reporting.

Table 6: Design decision 5: CO2 emission calculation

Code	DD-6
Name	Emission calculation approach
Problem	The system needs to decide the approach for finding CO2 emitters.
Decision	Bottom-up approach: using task specific information to determine the amount of co2 emissions
Benefits	• Detailed data: The bottom-up approach involves collecting data at a granular level, as individual activities or equipment usage. This results in accurate emission calculations. • Compatibility with business processes: Business processes consist of a number of tasks that match very well with the bottom-up approach, as by gathering information on all task, you can determine conclusions on the whole process.
Drawbacks	• Data collection: Gathering detailed data for multiple sources can be time-consuming and require significant resources. • Reduced functionality: By only looking from a bottom-up perspective, some emitors can be overlooked.
Alternatives	• Top-down approach: it eliminates the work to determine all single emitors, instead dealing more with average aggregate data. – Lower Accuracy: Less precise as it relies on aggregate data. – Reduced functionality: By only looking from a top-down perspective, the single devices as emission emitors are overlooked. • Hybrid approach – Complexity: Using both approaches has a significant increase in complexity of the system.

Table 7: Design decision 6: Bottom-up approach

Code	DD-7
Name	Emission calculation
Problem	What is the method for calculation of the CO2 emissions
Decision	Multiplying fuel quantity by an emission factor. The process is the following: 1. Determine the Fuel Quantity 2. Identify the Emission Factor 3. Calculate the CO2 Emissions $$CO2Emissions = FuelQuantity * EmissionFactor \quad (1)$$
Benefits	• Simplicity: This straightforward method ensures accurate and reliable estimation of CO2 emissions based on the specific characteristics of the fuel and the combustion process. • Popular: Several cited papers [36] [37] [20] [21] use this method of calculation
Drawbacks	• More effort on the client's side: The imported annotations will contain a predetermined consumption amount. For some resources, this amount is not straightforward and the person responsible with the annotations may have to to additional computations.
Alternatives	• Expanding the equation with other variables (power(hp), load factor etc.) – Increased complexity: More difficult to be compatible with the large number of emission factors.

Table 8: Design decision 7: Co2 Emission calculation

Code	DD-8
Name	Emission sources
Problem	What emission sources will be available for calculation
Decision	Any source, as long as they are imported before the process started
Benefits	• Compatibility: Any CO2 emitor can be used in calculation, expanding the compatibility with a wide variety of business processes.
Drawbacks	• Manual Effort: Additional manual effort might be required to import and correlate the emissions file with the BPMN model, leading to potential errors. However, a number of popular emission sources will be already implemented in the system. • Synchronization Issues: Ensuring that the fuel emissions file is correctly synchronized with the BPMN model can be challenging.
Alternatives	• Using only a set of defined sources – Limited flexibility: May not meet all specific needs of different business processes

Table 9: Design decision 8: Co2 emission sources

Code	DD-9
Name	Map emission information
Problem	The emission information needs to be mapped for the business process
Decision	Separate annotations file, imported before the process started
Benefits	• Flexibility: Allows the environmental impact data to be updated or modified without altering the business process model. • Separation of Concerns: Keeping carbon emissions data separate from the main BPMN file maintains a clear separation of the business logic and the data. • No Learning Curve: Users can continue using BPMN as they currently do without needing to learn new notations or extensions.
Drawbacks	• Synchronization Issues: Ensuring that the annotations file is correctly synchronized with the BPMN model can be challenging. • Manual Effort: Additional manual effort might be required to import and correlate the annotations file with the BPMN model, leading to potential errors. • Multiple Files: Users need to work with and maintain both the BPMN file and the annotations file, which can be cumbersome.
Alternatives	• Extending BPMN Notations – Learning Curve: Users need to learn and adapt to the new notations, which can be a barrier to adoption. – Compatibility Issues: Extended notations might not be compatible with all existing BPMN tools, leading to potential compatibility issues. • Embedded Scripts or Code Annotations – Increased Complexity: Embedding scripts or code increases the complexity of the BPMN models, making them harder to read, understand, and maintain. – Execution Overhead: Running scripts during process execution can introduce performance overhead, potentially slowing down the process execution, especially with complex calculations.

Table 10: Design decision 9: Upload annotations

4.5 User's perspective

The application use cases will be shown in figure 2. The more important use cases will be elaborated in the following tables.

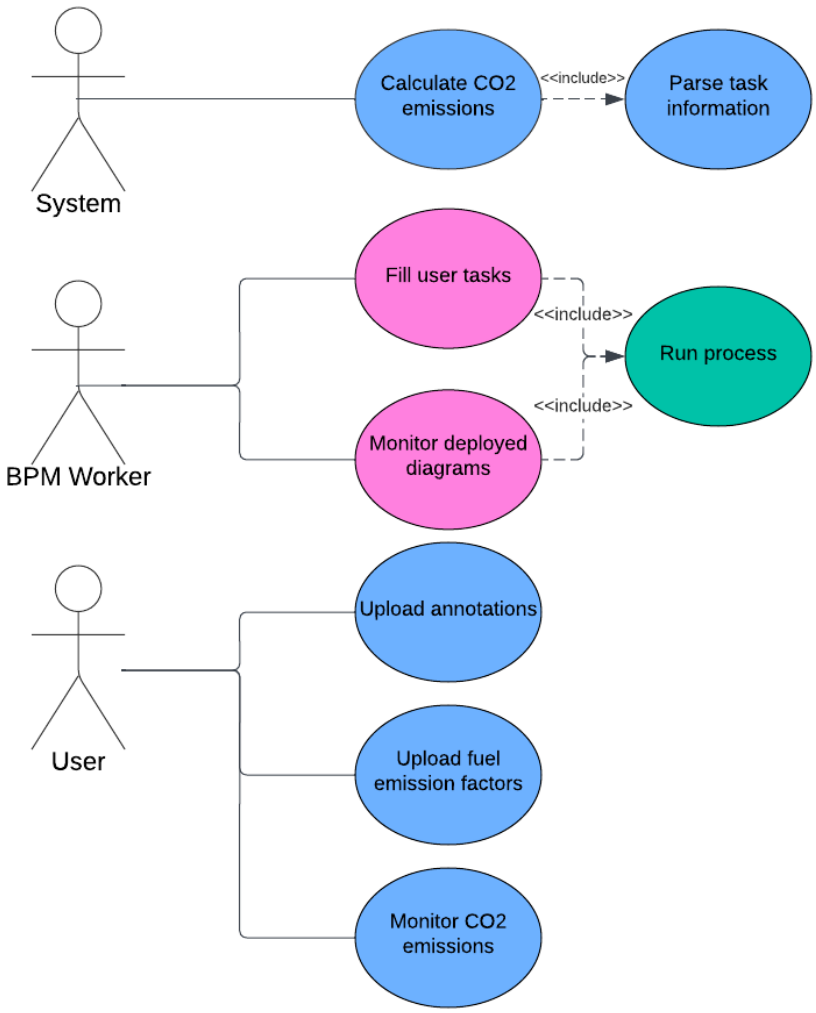


Figure 2: Use case diagram

Code	**UC-1**
Use Case	**Calculate CO2 emissions**
Actor	System
Result	The system computes carbon emissions after receiving information from the business process event listener and interpreting the input
Preconditions	• Running business process instance(es) • System is running • Stable and correct communication
Main Success Scenario	1. Event listener fires event 2. Engine integration receives the data and sends it to system 3. System receives data 4. System parses the data 5. System stores the data 6. System computes CO2 emissions 7. System stores CO2 emissions 8. System sends the data for visualization
Extensions	3a, 4a: Data not received: 1. Data is ignored, system waits until next payload 5a: Data does not have the expected format: 1. Error message is generated 2. System waits until next payload 7a: Task annotations not present: 1. Data is ignored, system waits until next payload 7b: Data values (e.g. measurement unit) do not correlate with annotation values 1. Error message is generated 2. System waits until next payload
Postconditions	• The activity data was parsed and stored • The activity's CO2 impact was calculated, stored, and delivered for visualization.

Code	**UC-2**
Use Case	**Upload annotations**
Actor	User
Result	The activity information is ready to be analysed
Preconditions	• System running
Main Success Scenario	1. User uploads annotations 2. System parses annotations 3. System stores annotations
Extensions	2a: Error uploading annotations 1. Error message is generated 3a: Bad annotation format 1. Error message is generated 2. Payload is ignored
Postconditions	• The annotations have been stored

Code	**UC-3**
Use Case	**Upload fuel emission factors**
Actor	User
Result	The CO2 emissions are ready to be calculated
Preconditions	• System running
Main Success Scenario	1. User uploads factors 2. System parses factors 3. System stores factors
Extensions	2a: Error uploading factors 1. Error message is generated 3a: Bad factor format 1. Error message is generated 2. Payload is ignored
Postconditions	• The factors have been stored

5 Solution

This section will outline the software solution developed for this master's thesis. The following subsections will provide an architectural overview, software decisions and implementation details.

5.1 Architectural overview

The system is structured within a containerized environment to ensure modularity and scalability. The architecture includes the following key components:

- **Business Process Management (BPM) Engines**: This component is a running BPMN execution engine that is used for executing the business processes that need to be verified. As the system is compatible with multiple BPM engines, a corresponding integration is used for communication between the BPM engine and the other modules. The engine itself utilizes event listeners to notify the system of process events such as process start, process end, activity start, and activity end

- **Integration Modules**: For each BPM engine, there is a dedicated integration module that acts as an intermediary between the main application and the BPM engines. These modules facilitate seamless communication and data exchange.

- **Main Application**: This component contains the core business logic of the system. It performs the majority of computational tasks, including emission calculations, and handles data management by using a database to persist data. Additionally, it ensures real-time communication with the front end for letting users visualize the carbon emissions.

- **Front End**: The front end manages the user dashboard, serving as the interface for system users. It allows users to upload fuel annotations files and fuel emission values, and provides a real-time view of the carbon emissions data.

- **Database**: The database is used by the main application to store and manage all relevant data, including process information and emissions calculations.

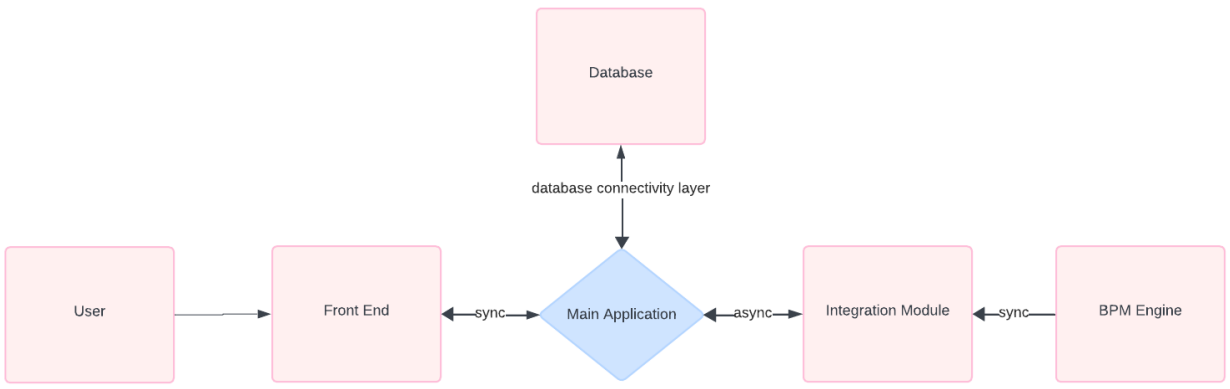


Figure 3: System Components Diagram

This structured approach ensures that the system is modular, scalable, and capable of providing accurate real-time information on carbon emissions, facilitating better environmental impact management for business processes.

5.2 Software decisions

In the development of the system, several design decisions were made regarding the technologies used for each component. Below is a detailed analysis of the advantages and disadvantages of each technology, along with alternatives and their respective disadvantages.

Code	SD-1
Name	Business Process Management (BPM) Engines
Problem	The system needs to be compatible with 2 business process engines
Decision	Camunda 7 & JBPM
Benefits	• Open Source: Both engines are freely available. • Popularity: Both Camunda and JBPM are among the most popular open source BPM engines (at the moment).
Drawbacks	• Complexity: JBPM has some comprehensive features that make it complex to set up and use. • Documentation: JBPM lacks extensive documentation and community support compared to some alternatives. • Deprecation: Camunda 7 is soon to be deprecated in favor of Camunda 8.
Alternatives	• Camunda 8 – Learning Curve: The behaviour of event listeners is different to the 2 selected engines, and there were issues in setting it up • Activiti – Community Support: Smaller community compared to Camunda and jBPM, which might lead to less documentation and plugins.

Table 11: Software decision 1: Camunda/JBPM

Code	SD-2
Name	Module framework
Problem	The application modules need to be implemented using a technology framework
Decision	Spring
Benefits	• Rapid Development: Easy to set up and run a project • Dependency Injection: Encourages decoupling of components through dependency injection, making the codebase more modular and easier to manage. • Integration: Seamlessly integrates with various other technologies and frameworks, such as Hibernate, JPA, RabbitMQ, etc.
Drawbacks	• Steep Learning Curve: Spring has a steep learning curve, especially for developers new to the framework. Understanding its extensive ecosystem and features can take considerable time.
Alternatives	• Django – Less Modularity: Django follows a more monolithic approach compared to Spring's highly modular structure – Built-In Features: While Django comes with many built-in features that simplify development, these can also lead to less flexibility and higher overhead if not all features are needed. Spring's modular approach allows for more selective inclusion of only necessary features. • Node.Js & Express – Less Mature Ecosystem: Node.js and Express, while powerful, are less mature in providing comprehensive enterprise-level features out of the box compared to Spring, which includes extensive support for transaction management, security, and dependency injection.

Table 12: Software decision 2: Spring

Code	SD-3
Name	Message broker
Problem	The system needs a message broker
Decision	RabbitMQ
Benefits	• Open Source: RabbitMQ is an open-source project • Spring AMQP Integration: RabbitMQ integrates seamlessly with Spring via the Spring AMQP project. This provides a simplified, consistent API for messaging, reducing boilerplate code and streamlining development. • Management Plugin: Offers a powerful web-based management interface for monitoring, managing, and debugging RabbitMQ.
Drawbacks	• Latency: May introduce latency in message processing compared to some other messaging systems designed for ultra-low latency.
Alternatives	• Apache Kafka – Setup and Management: Requires a more complex setup and ongoing management compared to RabbitMQ. – Learning Curve: Steeper learning curve due to its distributed nature and configuration options. • ActiveMQ – Community Size: Smaller community and ecosystem compared to more popular systems like Kafka and RabbitMQ

Table 13: Software decision 3: RabbitMQ

Code	SD-4
Name	Database technology
Problem	A database instance is needed for storing various data of the system
Decision	PostgreSQL
Benefits	• ACID Compliance: Full ACID (Atomicity, Consistency, Isolation, Durability) compliance ensures reliable transactions. • Open Source: Free to use and supported by a large community, reducing licensing costs. • Community Support: Strong community support with extensive documentation, plugins, and third-party tools.
Drawbacks	• Write-Intensive Workloads: May not perform as well as some NoSQL databases for extremely write-intensive workloads.
Alternatives	• MySQL – ACID Compliance: Historically, less strict ACID compliance in some configurations, although this has improved in recent versions. • MongoDB – Data Consistency: Uses eventual consistency, which can be challenging for applications requiring strict transactional consistency.

Table 14: Software decision 4: PostgreSQL

Code	SD-5
Name	Communication protocol
Problem	The system needs a communication protocol between the frontend and backend
Decision	HTTP & WebSocket
Benefits	• Low Latency: WebSockets provide a persistent connection, enabling low-latency, real-time communication between the client and server. • Simplicity: HTTP requests are straightforward to implement and understand. They are widely used and supported across various platforms and frameworks. • Spring MVC: Spring provides robust support for creating RESTful APIs using Spring MVC. It offers powerful annotations (@RestController, @RequestMapping, etc.) to define request mappings, handle responses, and manage HTTP methods (GET, POST, PUT, DELETE).
Drawbacks	• Complexity: Implementing and managing both WebSocket and HTTP connections increases complexity
Alternatives	• Long Polling – Inefficient compared to WebSockets due to increased latency and overhead from frequently opening and closing connections.

Table 15: Software decision 5: Http/Websocket

The software decisions defined above project the following architecture diagram, that illustrates the technologies used to build the components and the interactions between these components within the containerized environment. The diagram shows the flow of data and communication paths, highlighting how the system components work together to achieve the overall functionality.

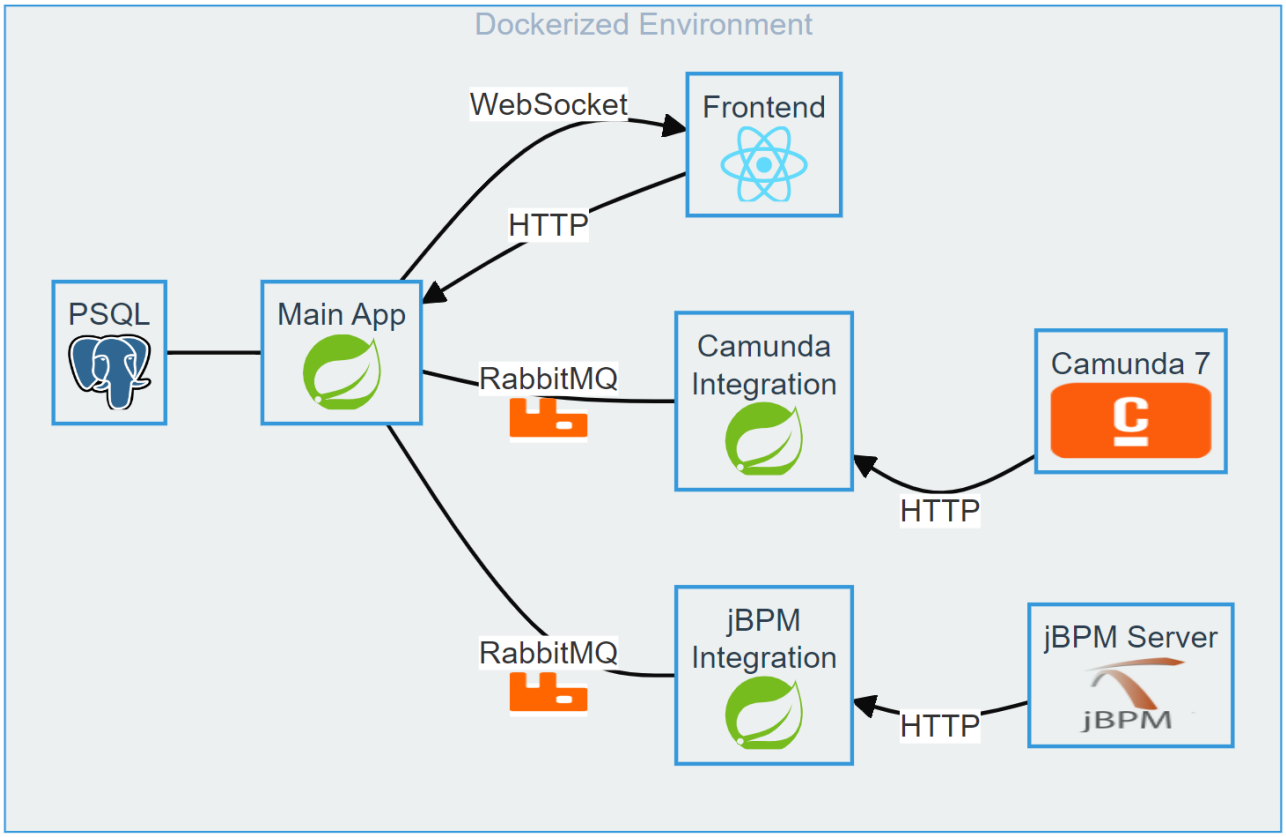


Figure 4: System Architecture Diagram

5.3 Implementation Design

This section will go into more detail about each component of the system. A detailed description of the modules' functionalities and their logical views is provided.

5.3.1 BPM Engines

Both compatible BPM engines are used to execute BPM processes. The product that wants to integrate the sustainability calculator system uses one of these engines to run their process instances, and the system will just connect to the running BPM technology stack. For both Camunda 7 and JBPM, the system needs an event listener implemented to listen to several events used in the emission calculation and data management. The type of the event listeners is up to the stakeholders of the system, depending on their deployment format and preferences. For this thesis, the event listeners were mocked:

- Camunda 7 uses 4 execution listeners with Java delegate implementation.
- JBPM uses a process event listener with 4 overriden methods.

Both types of listeners have the same logic of sending a payload through HTTP. The payload differs depending on the event type, with the following possible fields:

- **processName** with the name of the process
- **processKey** with the id of the process instance

- **platform**
- **taskName** with the name of the current activity that started/ended
- **status** flag to set when a process starts or ends
- **startTime** for when a process or activity starts
- **endTime** for when a process or activity ends

All events contain processName, processKey and platform, and the rest of the attributes are sent as following:

- An event fires when the process starts. It sends a payload containing status start flag (=*1*) and the start time of the process
- An event fires when the activity starts. It sends the task name and start time of the task
- An event fires when the activity ends. It sends the task name and end time of the task
- An event fires when the process ends. It sends a payload containing status end flag (=*2*) and the end time of the process

```
public class StartEventExecutionListener implements ExecutionListener {
    @Override
    public void notify(DelegateExecution delegateExecution) {}
}
public class StartExecutionListener implements ExecutionListener {
    @Override
    public void notify(DelegateExecution delegateExecution) {}
}
public class EndExecutionListener implements ExecutionListener {
    @Override
    public void notify(DelegateExecution delegateExecution) {}
}
public class EndEventExecutionListener implements ExecutionListener {
    @Override
    public void notify(DelegateExecution delegateExecution) {}
}
```

Figure 5: Camunda 7 listener code

```
public class MyProcessEventListener extends DefaultProcessEventListener {
    @Override
    public void beforeNodeLeft(ProcessNodeLeftEvent event) {}
    @Override
    public void beforeNodeTriggered(ProcessNodeTriggeredEvent event) {}
    @Override
    public void beforeProcessStarted(ProcessStartedEvent event) {}
    @Override
    public void afterProcessCompleted(ProcessCompletedEvent event) {}
}
```

Figure 6: JBPM listener code

Besides the event listeners, another important aspect is the engine deployment. In a production environment, the system connects to a deployed engine stack with the necessary listeners. For this project, both engines are deployed on singular Docker containers.

- For Camunda, a Spring Boot project with the Camunda 7 Spring Boot dependency was chosen for its easy setup and listener management. The process is deployed and run through the Camunda Modeler that is connected to the Camunda Engine project. The information on the process and the user task management is then done from the Camunda Engine project.
- The JBPM architecture is significantly more difficult to set up locally for process execution. The way it works is that a project containing the BPMN diagram installed inside a KJar needs to be accessible to create a JBPM Container, and to execute the process. The way it was set-up for this project is that a Maven project containing the Event Listener implementation

and the diagram was installed as a KJar and added in a docker volume at the JBPM Server Docker container start-up. The JBPM container is created through an HTTP request(can be made through the Swagger API contained in the JBPM Server), as well as the process instance start. The information on the process and the user task management is then done from the JBPM Server.

5.3.2 Engine integration modules

The integration components' objective is to ensure communication between the main application and the BPM engines. It listens to requests from the process event listeners from the BPM engine and perpetuates them through the asynchronous message broker to the main app. In addition, it is tasked with managing the communication message channel and its properties.

```
1     return QueueBuilder.durable("camundaReturnChannel")
2                 .withArgument("x-dead-letter-exchange", "
                      dlxExchange")
3                 .withArgument("x-dead-letter-routing-key", "camunda
                      ")
4                 .withArgument("x-message-ttl", 1000)
5                 .build();
```

Listing 1: RabbitMQ properties

5.3.3 Main application

It contains the core business logic of the system. It has multiple purposes:

- It parses the received annotations data, to map the fuel consumption to the business tasks. It also receives fuel emission factors, to validate the resources mentioned in the annotations. The annotations and fuel emission factors are received as HTTP payloads from the frontend. The annotations and fuel emissions are more thoroughly explained in section 5.5

```
1     @Transactional
2     public void saveAnnotations(String json) throws
          JsonProcessingException {}
3
4     @Transactional
5     public void saveFactors(String json) throws
          JsonProcessingException {}
```

Listing 2: Annotations and fuel emissions parsing

- It receives task execution information through message channels. It uses this information to map each process instance domain object

```
1     @RabbitListener(queues = {"camundaReturnChannel","
          jbpmReturnChannel"})
2     public void receiveMessage(String message) throws
          JsonProcessingException{}
3
```

```
    @Transactional
    public void parseReceivedMessage(String message){}
```

Listing 3: Message channel usage

- Using the received payloads, it calculates CO2 emissions per activity and per process instance, detailed in section 5.4.3

```
    @Transactional
    public List<IntermediateProcessConsumption>
       calculateConsumption(Map map, TaskAnnotation
       taskAnnotation) {}

    @Transactional
    public double calculateEmissions(List<
       IntermediateProcessConsumption> list){}

    @Transactional
    public double calculateTotalEmissions(Process process){}
```

Listing 4: CO2 emission calculation

- Performs real-time communication with the frontend, ensuring the user visualizes CO2 emissions per task.

```
    @Transactional
    public void sendPayloadToFrontend(Map payload) throws
       JsonProcessingException{}

    public void sendAllAnnotationJsonToFrontend(){}

    public void sendAllFuelEmissionFactorToFrontend(){}
```

Listing 5: Frontend communication

Multiple design patterns are used to ensure that the code is modular, maintainable, and abides by best practices.

- Singleton Pattern: It ensures that a class has only one instance and provides a global point of access to it. This pattern is useful when exactly one object is needed to coordinate actions across the system. This is achieved using the @Configuration and @Bean annotations in Spring.

```
        @Configuration
        public class AppConfig {
            @Bean
            public RestTemplate restTemplate() {
                return new RestTemplate();
            }
        }
```

Listing 6: Singleton pattern in code

- Controller Pattern: The Controller Pattern defines a controller that handles incoming HTTP requests, processes them (often involving calls to service layers), and returns a response. It helps in separating the concerns of user interface logic from business logic. Annotated with @RestController and @RequestMapping.

```
    @PostMapping("/upload-fuel-emission-factors")
    public ResponseEntity<String> uploadFuelEmissionFactors(
        @RequestParam("factors") String factors) throws
        IOException, InterruptedException {
         try {
             appService.saveFactors(factors);
             return ResponseEntity.status(HttpStatus.CREATED).
                 body("Fuel emission factors upload request
                 finished!");
         } catch (Exception e) {
             e.printStackTrace();
             return ResponseEntity.status(HttpStatus.
                 INTERNAL_SERVER_ERROR).body("Failed to save fuel
                  emission factors: " + e.getMessage());

         }
    }

}
```

Listing 7: Controller pattern in code

- Dependency Injection Pattern: Dependency Injection (DI) is a design pattern used to implement IoC (Inversion of Control), allowing the creation of dependent objects outside of a class and providing those objects to a class through different ways. This pattern helps in achieving loose coupling. Achieved using @Autowired.

```
    @Autowired
    private AppService appService;
```

Listing 8: DI pattern in code

- Observer Pattern: The Observer Pattern defines a one-to-many dependency between objects, such that when one object changes state, all its dependents are notified and updated automatically. Implemented Using RabbitListeners and WebSocket connections.

```
    @Component
    public class WebSocketEventListener implements
        ApplicationListener<SessionSubscribeEvent> {
         @Autowired
         private SimpMessagingTemplate messagingTemplate;
         @Autowired
         private FrontendDataRepository frontendDataRepository;
         @Autowired
```

```
8           private AnnotationJsonRepository
                annotationJsonRepository;
9           @Autowired
10          private FuelEmissionFactorRepository
                fuelEmissionFactorRepository;
11
12          @Override
13          public void onApplicationEvent(SessionSubscribeEvent
                event) {
14              List<FrontendData> frontendDataList =
                    frontendDataRepository.findAll();
15              List<AnnotationJson> annotationJsonList =
                    annotationJsonRepository.findAll();
16              List<FuelEmissionFactor> fuelEmissionFactorList =
                    fuelEmissionFactorRepository.findAll();
17              messagingTemplate.convertAndSend("/topic/emissions"
                    , frontendDataList);
18              messagingTemplate.convertAndSend("/topic/
                    annotations", annotationJsonList);
19              messagingTemplate.convertAndSend("/topic/
                    emissionFactors", fuelEmissionFactorList);
20          }
21      }
```

Listing 9: Observer pattern in code

- Repository Pattern: The Repository Pattern mediates between the domain and data mapping layers using a collection-like interface for accessing domain objects. This pattern abstracts data access, making the code easier to maintain and test. Extending JpaRepository provides CRUD operations for entities.

```
1       @Repository
2       public interface ProcessRepository extends JpaRepository<
            Process, Long> {
3           Process findByProcessNameAndProcessKey(String
                processName, String processKey);
4       }
```

Listing 10: Repository pattern in code

- Service Pattern: The Service Pattern defines a service layer that encapsulates the business logic of an application. It serves as an intermediary between controllers and repositories, ensuring that business rules and logic are applied consistently. Implemented using classes annotated with @Service

```
1       @Service
2       public class RabbitService {
3
4           @Autowired
```

```
5         private RabbitTemplate rabbitTemplate;
6
7         public void sendMessage(String channel, String message)
              {
8             System.out.println(channel + message);
9             rabbitTemplate.convertAndSend(channel, message);
10        }
11    }
```

Listing 11: Service pattern in code

- Handler Pattern (Controller Advice): The Handler Pattern provides a way to handle concerns such as error handling or logging in a centralized manner. @ControllerAdvice annotations help classes intercept and handle exceptions thrown by controller methods across the application.

```
1     @ControllerAdvice
2     public class GlobalExceptionHandler {
3
4         @ExceptionHandler(IOException.class)
5         public ResponseEntity<?> handleIOException(IOException
              ex, WebRequest request) {
6             return new ResponseEntity<>("I/O error: " + ex.
                  getMessage(), HttpStatus.INTERNAL_SERVER_ERROR);
7         }
8
9         @ExceptionHandler(IllegalArgumentException.class)
10        public ResponseEntity<?> handleIllegalArgumentException
              (IllegalArgumentException ex, WebRequest request) {
11            return new ResponseEntity<>("Invalid argument: " +
                  ex.getMessage(), HttpStatus.BAD_REQUEST);
12        }
13
14        @ExceptionHandler(Exception.class)
15        public ResponseEntity<?> handleGlobalException(
              Exception ex, WebRequest request) {
16            return new ResponseEntity<>("An error occurred: " +
                  ex.getMessage(), HttpStatus.
                  INTERNAL_SERVER_ERROR);
17        }
18    }
```

Listing 12: Handler pattern in code

To ensure the accuracy and reliability of the main application, a suite of unit tests has been developed. These tests cover critical functionalities and services within the application. Each unit test is designed to validate the behavior of individual components in isolation, using Spring annotations, ensuring that they perform as expected under various conditions. They handle core logic for annotations and process data, calculations on fuel and emission consumption, real-time communication and asynchronous messaging functionality.

```
public class AppServiceTest {
    @Test
    public void testCalculateTotalEmissions() {}
    @Test
    public void testSaveAnnotations() throws
        JsonProcessingException {}
    ...
}
public class EmissionFactorServiceTest {
    @Test
    public void testCalculateTotalEmission() throws
        JsonProcessingException {}
    @Test
    public void testCalculateConsumption() {}
    ...
}
public class WebSocketServiceTest{
    @Test
    public void testSendPayloadToFrontend() throws
        JsonProcessingException {}
    @Test
    public void testSendAllFuelEmissionFactorToFrontend() {}
    ...
}
public class RabbitServiceTest {
    @Test
    public void testSendMessage() {}
}
```

Listing 13: Unit tests

5.3.4 Front-end

This module presents a real-time overview of carbon emissions of a business process during execution. In addition, it manages process annotations and emission factors:

- It allows users to upload annotation files or write the annotation payload themselves. The same goes for emission factors
- It allows editing saved annotations and emission factors
- It presents carbon emissions per business process activity and per entire process instance, together with the business process diagram. These informations can be downloaded in the form of a report

It uses HTTP to communicate with the main application for uploading elements to the db.

```
const handleFileUpload= (event) => {};
```

```
    const handleFileSubmit = async (annotations) => {};
    const handleEmissionFactorSubmit = async (emissionFactors) =>
       {};
```

Listing 14: Upload handlers

The front-end does not have a database connection, for better decoupling. In order to receive stored data, it uses Stomp WebSockets with a connection to the main application

```
    stompClient.subscribe('/topic/messages', (message) => {};
    stompClient.subscribe('/topic/emissions', (message) => {};
    stompClient.subscribe('/topic/annotations', (message) => {};
    stompClient.subscribe('/topic/emissionFactors', (message) =>
       {};
```

Listing 15: Stomp channels

It allows users to add and edit annotations through a modal interface. There are React state hooks used to manage the modal's open/close state and the selected annotation.

```
    const [modalIsOpen, setModalIsOpen] = useState(false);
    const [selectedAnnotation, setSelectedAnnotation] = useState(
       null);

    const openModal = (annotation) => {
        setSelectedAnnotation(annotation);
        setModalIsOpen(true);
    };

    const closeModal = () => {
        setModalIsOpen(false);
        setSelectedAnnotation(null);
    };
```

Listing 16: Modal

The dashboard also visualizes business process models using the bpmn-js library, providing a graphical representation of the processes. It initializes the BPMN viewer and loads the diagram XML when received on the websocket channel from the main application

```
    bpmnViewer.current = new BpmnViewer({});
    bpmnViewer.current.importXML(diagramXML).then(() => {});
```

Listing 17: BPM viewer

The CO2 emission data for each process instance can be downloaded in a report with the PDF format. The pdf contains each task's CO2 emissions and the total amount per process instance, as well as the business process diagram.

```
    const handleDownload = async () => {};
    await html2canvas(hiddenViewerRef.current).then((canvas) => {};
    hiddenBpmnViewer.current.importXML(diagramXML).then(() => {});
```

Listing 18: Report component

5.4 Process view

The process diagram below presents an overview of the system's processes and the interaction between the different modules. It focuses on the system's run-time behaviour and the possible use cases present on each step.

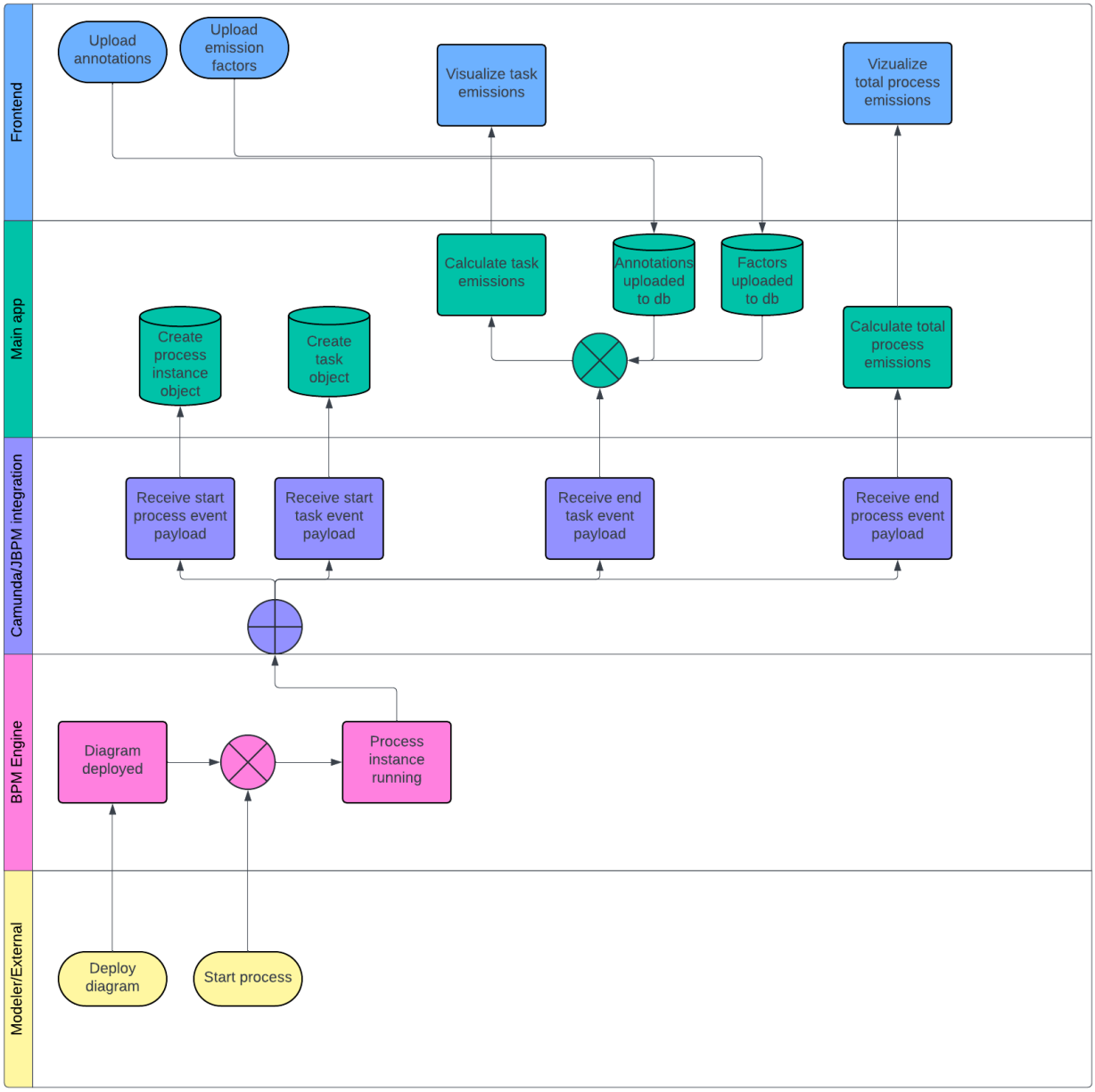


Figure 7: Process diagram

5.4.1 Upload annotation sequence

Annotation upload is done every time the user wants to import new business process annotations or update the already existing ones. It involves submitting them though the frontend, their parsing and their persistence. The figure below presents the sequence diagram, along with the more relevant function calls throughout the system.

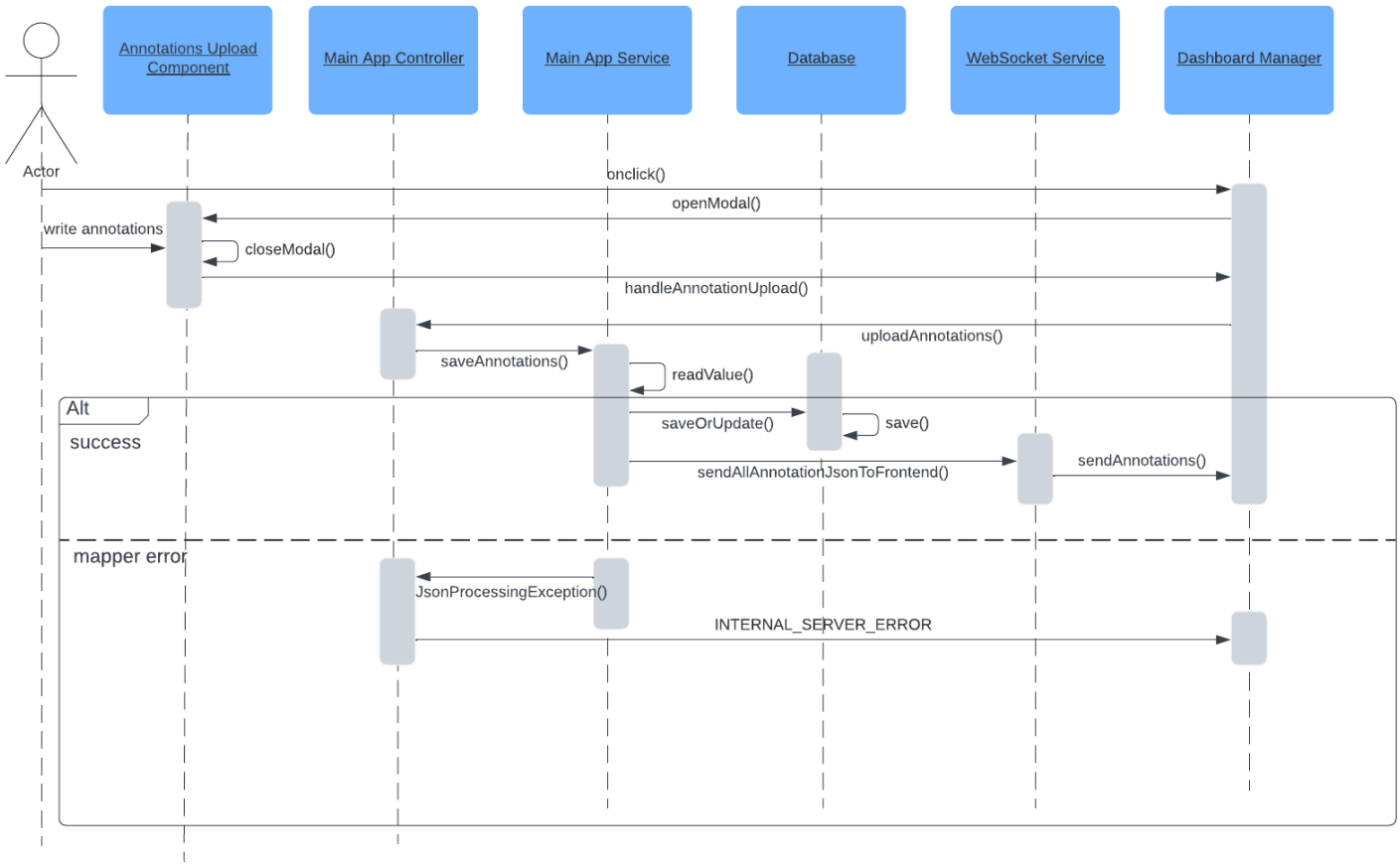


Figure 8: Annotations sequence diagram

The fuel emission factors upload sequence follows a similar procedure. Therefore, this sequence diagram effectively represents both the annotation and fuel emission upload sequences.

5.4.2 Process instance sequence

This diagram presents the behaviour of the business process event listeners that interact with the system, and how all the event types are handled.

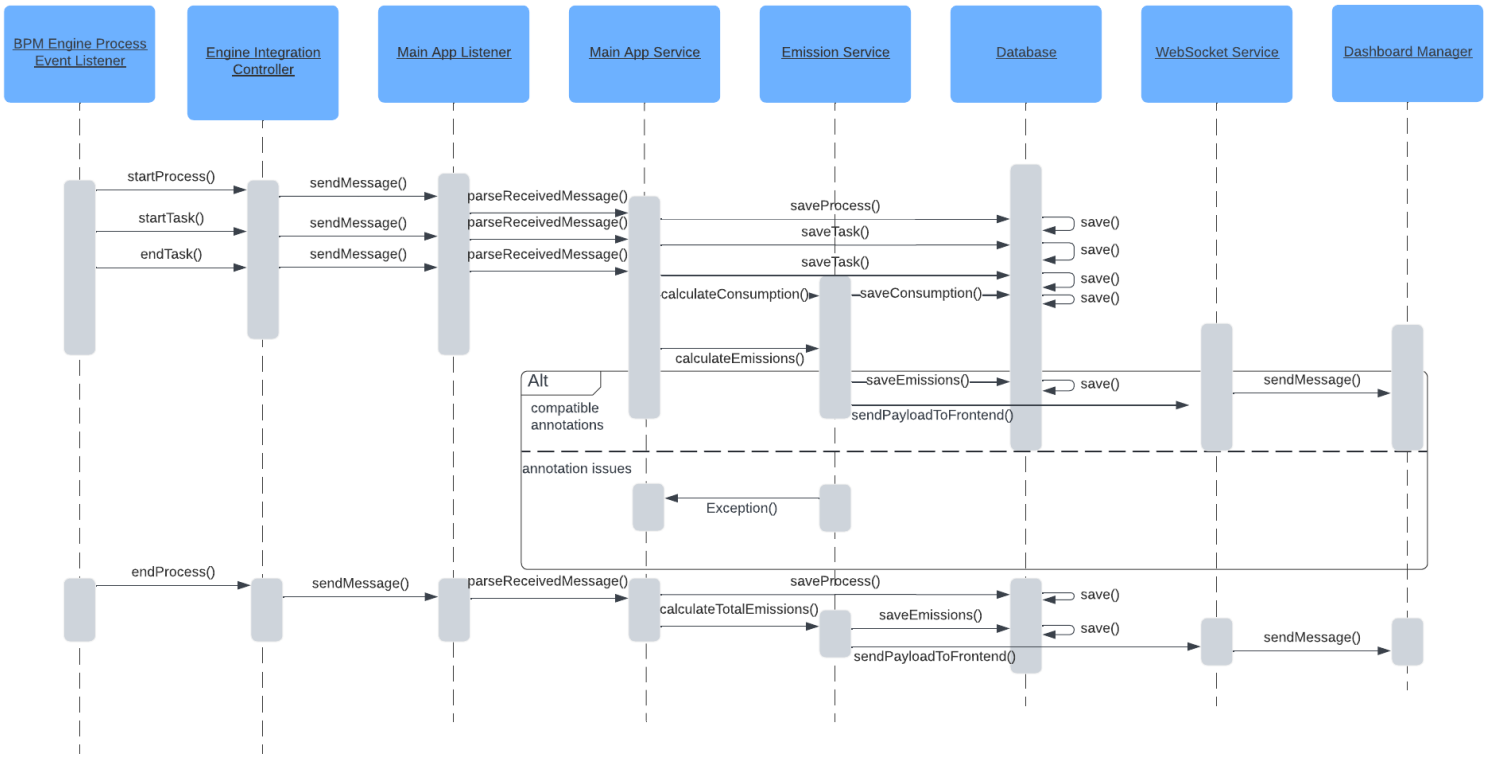


Figure 9: Process instance sequence diagram

5.4.3 Emission calculation sequence

The emission calculation procedure follows the flow of the messages received by the main application from the engine integration through RabbitMQ channels. This sequence was summarized in the previous sequence diagram, but it will be presented in more detail below.

- When a start process message is received, a process instance object is created, with the given process name, instance name, and start time
- When a start task message is received, a task object is created, with the given name, process instance and start time.
- When an end task message is received, the task object is updated with the given end time. In addition, the procedure to calculate the task emissions begins. The annotations needed for the task are extracted from the database. Next, the amount of fuel consumed by the task itself is calculated using either the amount of time written in the annotations, or the duration between the start and end task events. The end result for this calculation is a list of fuel quantities with varying units. This list is then parsed element by element, to extract the fuel emission factor for the specific fuel type, do unit conversion(e.g. tonnes to kilograms) and multiply the emission factor with the resulted fuel amount. The sum of emissions that results from all the resources consumed by the task is then the CO2 emission value of the task, and it is saved in the database and sent to the front-end dashboard.
- When an end process message is received, the process instance object is updated with the given end time, and the total amount of emissions from the process is calculated, by summing the CO2

emission values of all the tasks belonging to the process instance. It is saved in the database and sent to the front-end dashboard

5.5 Domain view

In this section, the more important domain classes of the system will be presented.

The state of the process instances that are monitored by the system needs to be stored persistently. The system reacts to the process events and updates the domain objects' timestamps, and also imports the XML diagram.

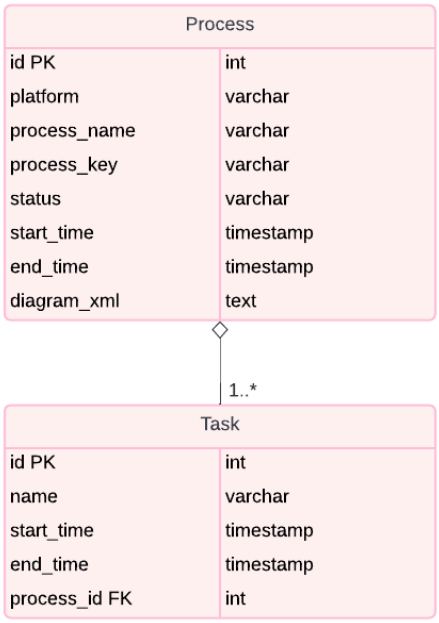


Figure 10: Process and task domain models

Annotations are imported before the business process instance starts and are used to map the processes' tasks to a number of resources used, in order to calculate the CO2 output. The process of importing annotations involves their preparation in a structured JSON format. This format encapsulates the entire process, including its tasks and associated resources. The following is an example JSON that encapsulates the entire process, tasks, and resources:

```
{
    "name": "Process_1y80xt6",
    "frequencyPerMonth": 5,
    "tasks": [
        {
            "name": "task1",
            "duration": "30",
            "timeUnit": "minutes",
            "resourcesUsed": [
```

```
                {"resourceName": "Welder", "timeUsed": "30", "unit
                   ": "minutes"}
            ]
        },
        {
            "name": "task2",
            "duration": "45",
            "timeUnit": "minutes",
            "resourcesUsed": [
                {"resourceName": "Paint Sprayer"}
            ]
        }
    ],
    "resources": [
        {"name": "Welder", "type": "atomic", "fuelPerUse": "5", "
           fuelType": "Diesel", "fuelUnit": "l", "timeUnit": "hour
           "},
        {"name": "Paint Sprayer", "type": "atomic", "fuelPerUse":
           "3", "fuelType": "Diesel", "fuelUnit": "l", "timeUnit":
           "hour"}
    ]
}
```

Parsing this file creates 4 different types of objects, **ProcessAnnotation, TaskAnnotation, ResourceAnnotation, ResourceUsage**. By importing all annotations at once, the consistency of the data is maintained. All related components (processes, tasks, resources) are imported together, ensuring that there are no missing parts. In addition, it minimizes the need for repetitive data entry. Users can prepare the data once and import them, avoiding redundant operations.

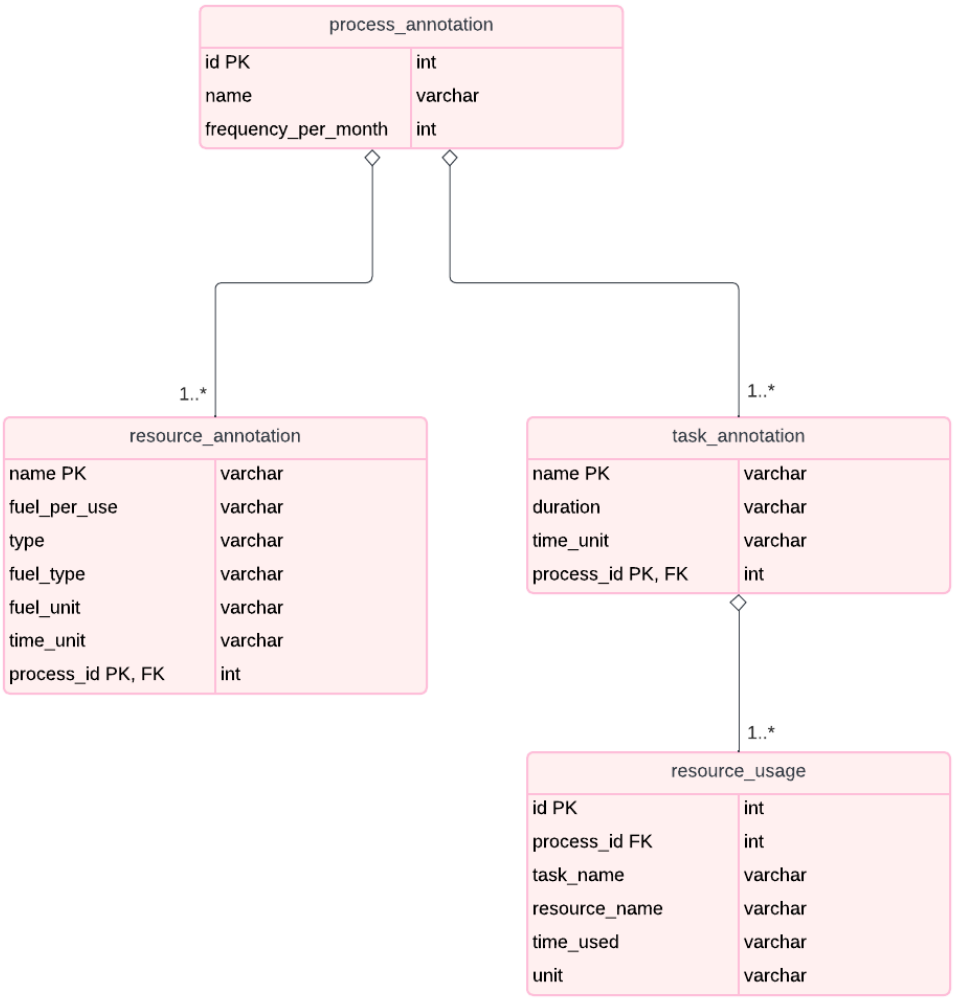


Figure 11: Annotations domain

5.6 Project setup

This subsection will provide step-by-step instructions to manage running the sustainability calculator project. A more detailed documentation can be found in the public repository.

- Clone the project from the public Github [4] repository
- Open the project in your chosen IDE compatible with Spring
- Build all Gradle [5] dependencies
- In order to run in a containerized environment, make sure Docker is running, before running the **docker-build.sh** script. The script has 1 argument, representing the docker compose profile wanted. There are 2 profiles: camunda and jbpm.

[4] https://github.com/
[5] https://gradle.org/

- If the script ran without any issues the frontend dashboard located at **localhost:3000** should look like this:

Figure 12: Empty dashboard

- For uploading/editing annotations and fuel emission factors, use the buttons present in the dashboard. Modals will pop up for either uploading a file or writing the json.

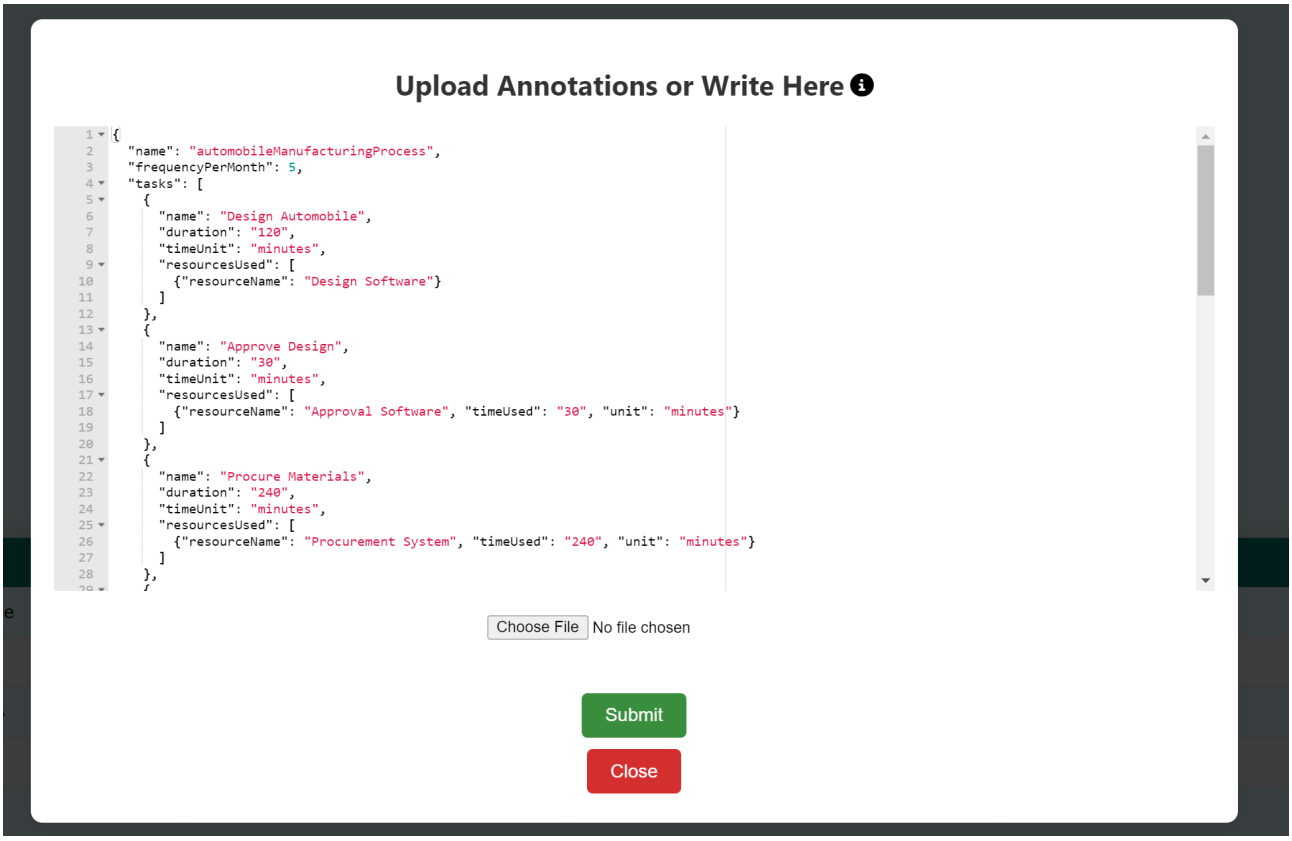


Figure 13: Annotations upload

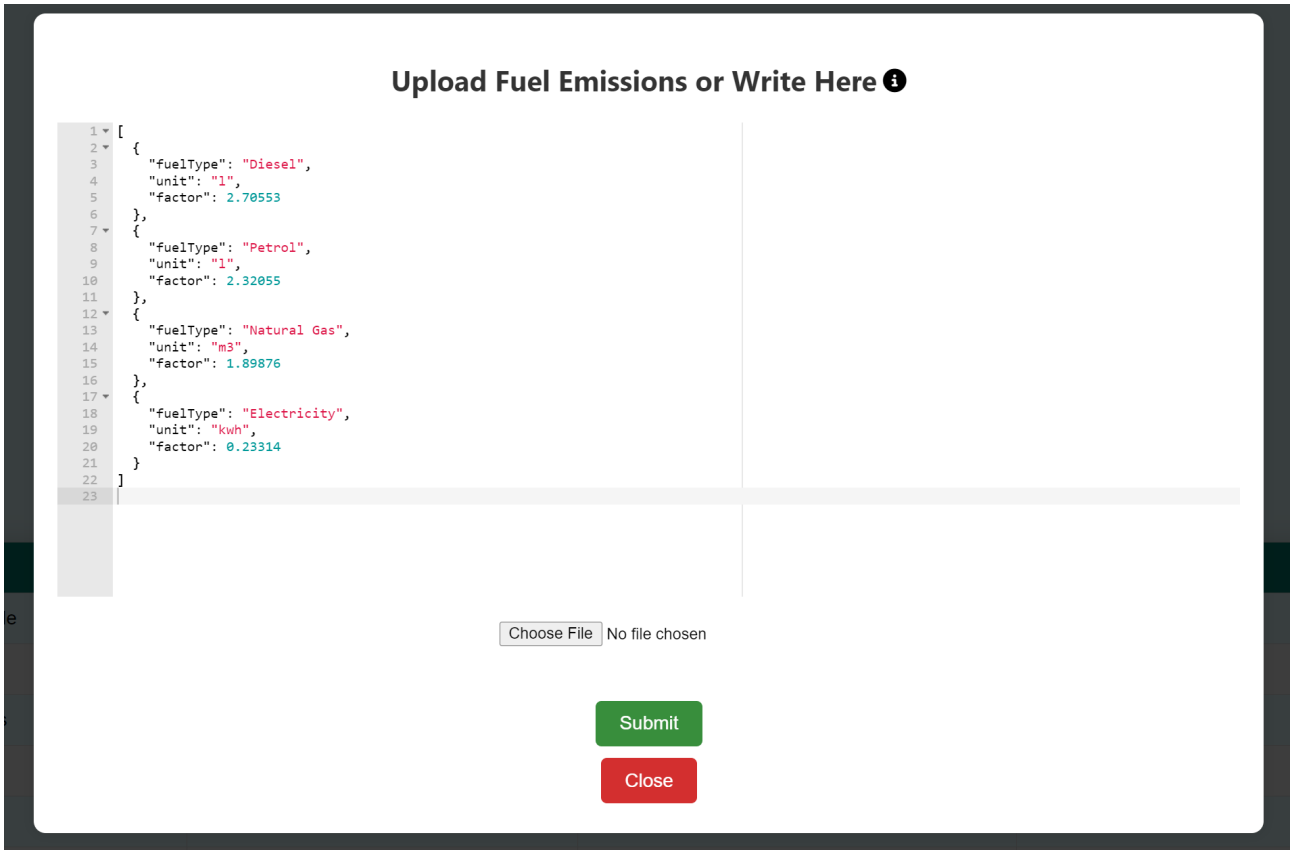


Figure 14: Emissions upload

Figure 15: Annotations edit

- For integrating Camunda, you can use a locally running Camunda Modeler to build the BPMN model. For the system to monitor the emissions correctly, make sure to add start and end event listeners on each activity, start event listener on the start event, and end event listener on the end event. Each service task should also have an implementation. When done with setting up the model, you can deploy it on **http://localhost:8080/engine-rest**, along with any forms, and then run it from the modeler. Any user task can be filled at **http://localhost:8080/camunda/app/tasklist/default**.

- For JBPM, the docker compose script uploads a KJAR on the JBPM server. The KJAR contains the BPMN diagram and the linked forms. After the containers are running, use Swagger located at http://localhost:8080/kie-server/docs to create the container. use the following payload: ”container-id” : $container-name, ”release-id” : ”group-id” : ”org.tests.processes”, ”artifact-id” : ”jbpm-processeventlistener”, ”version” : ”0.1.0-SNAPSHOT” . Then you can either create a process instance through the jbpm server console at http://localhost:8080/business-central (wbadmin:wbadmin) or through swagger, using your process id.

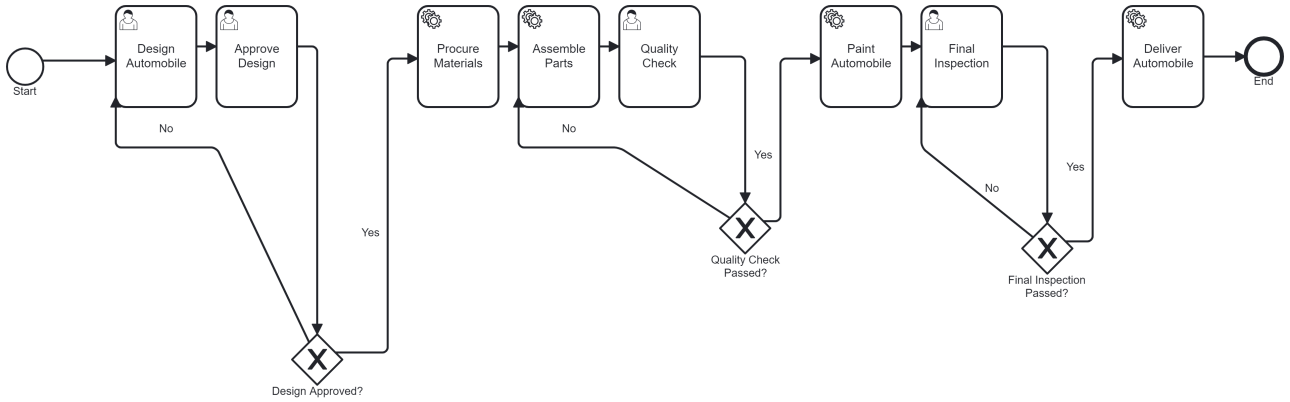


Figure 16: Example BPMN diagram

- Run your process instance from the engine. After the first task has been executed, new buttons for the current process and the running process instance will appear, along with the process diagram. The dashboard should look like this:

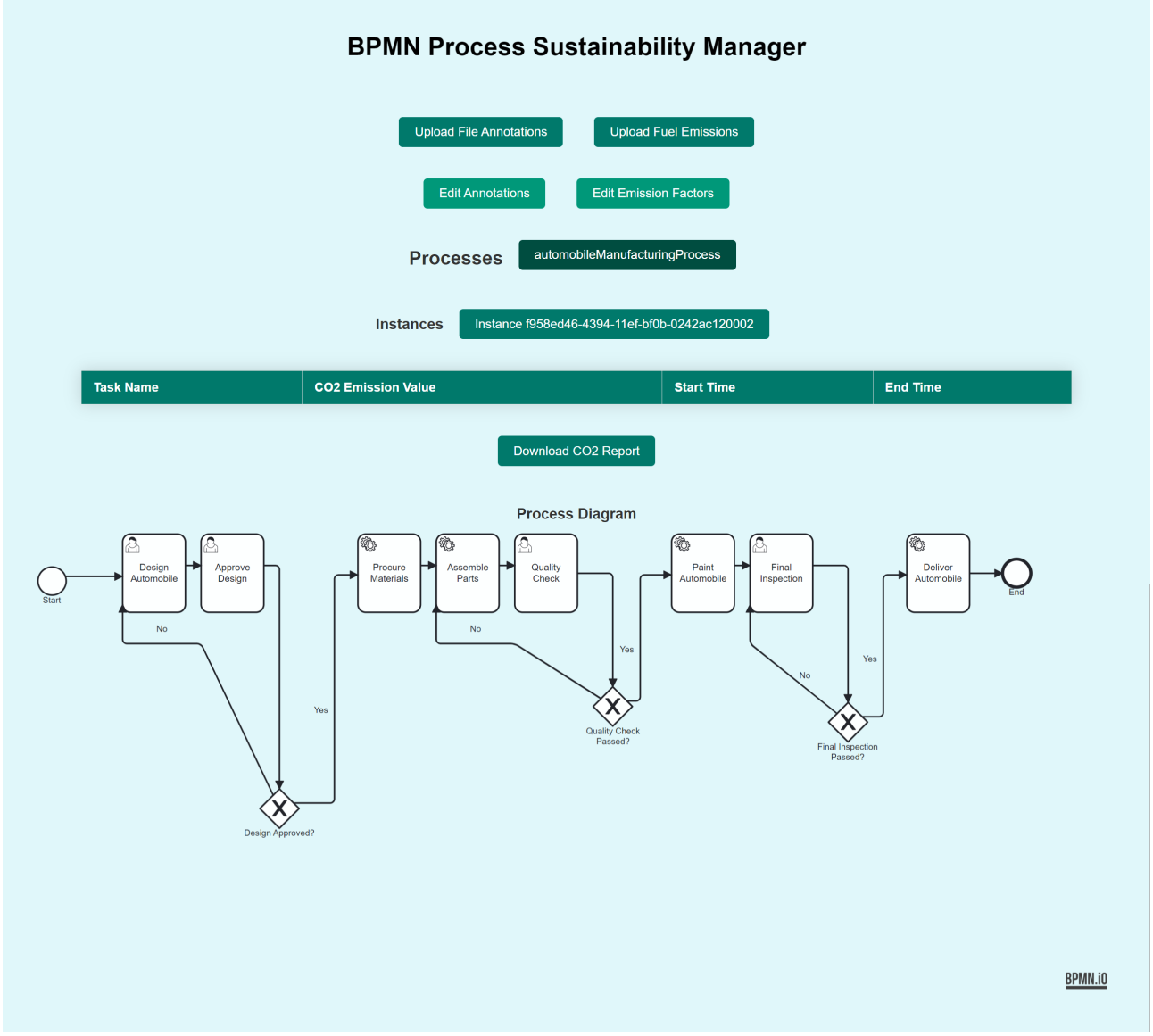
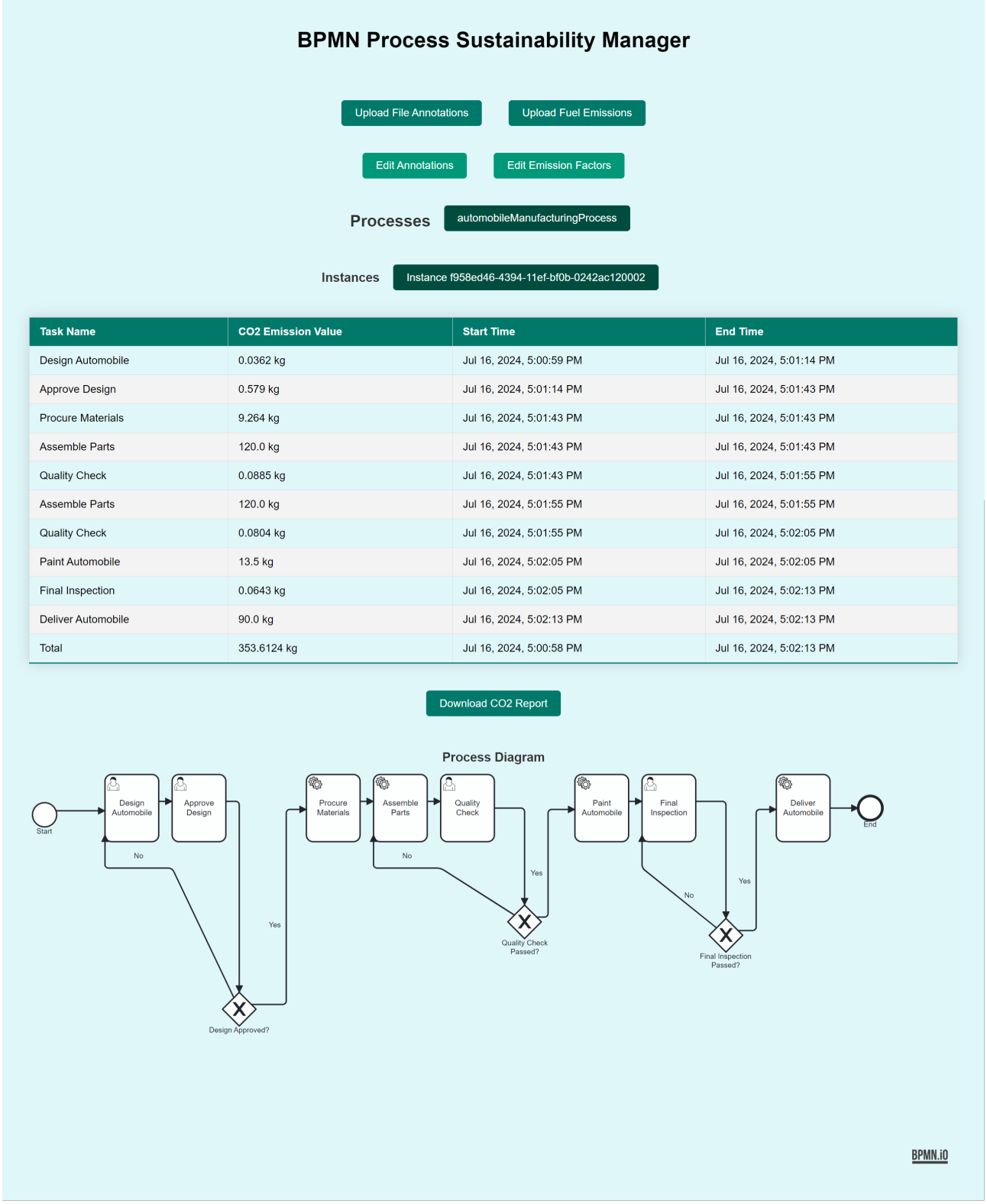


Figure 17: Dashboard during process execution

- Select the desired process instance and see each executed activity's Co2 emissions. The emissions per activity appear after the activity has been executed end the activity end event fired. The emissions for user activities are shown after the form was claimed, filled and submitted. After the process ended, the total co2 amount is shown.



Task Name	CO2 Emission Value	Start Time	End Time
Design Automobile	0.0362 kg	Jul 16, 2024, 5:00:59 PM	Jul 16, 2024, 5:01:14 PM
Approve Design	0.579 kg	Jul 16, 2024, 5:01:14 PM	Jul 16, 2024, 5:01:43 PM
Procure Materials	9.264 kg	Jul 16, 2024, 5:01:43 PM	Jul 16, 2024, 5:01:43 PM
Assemble Parts	120.0 kg	Jul 16, 2024, 5:01:43 PM	Jul 16, 2024, 5:01:43 PM
Quality Check	0.0885 kg	Jul 16, 2024, 5:01:43 PM	Jul 16, 2024, 5:01:55 PM
Assemble Parts	120.0 kg	Jul 16, 2024, 5:01:55 PM	Jul 16, 2024, 5:01:55 PM
Quality Check	0.0804 kg	Jul 16, 2024, 5:01:55 PM	Jul 16, 2024, 5:02:05 PM
Paint Automobile	13.5 kg	Jul 16, 2024, 5:02:05 PM	Jul 16, 2024, 5:02:05 PM
Final Inspection	0.0643 kg	Jul 16, 2024, 5:02:05 PM	Jul 16, 2024, 5:02:13 PM
Deliver Automobile	90.0 kg	Jul 16, 2024, 5:02:13 PM	Jul 16, 2024, 5:02:13 PM
Total	353.6124 kg	Jul 16, 2024, 5:00:58 PM	Jul 16, 2024, 5:02:13 PM

Figure 18: Dashboard with emissions

- You can download the report using the download report button

CO2 Emission Report

Process: automobileManufacturingProcess
Instance: f958ed46-4394-11ef-bf0b-0242ac120002

Task Name	CO2 Emission Value	Start Time	End Time
Design Automobile	0.0362 kg	Jul 16, 2024, 5:00:59 PM	Jul 16, 2024, 5:01:14 PM
Approve Design	0.579 kg	Jul 16, 2024, 5:01:14 PM	Jul 16, 2024, 5:01:43 PM
Procure Materials	9.264 kg	Jul 16, 2024, 5:01:43 PM	Jul 16, 2024, 5:01:43 PM
Assemble Parts	120.0 kg	Jul 16, 2024, 5:01:43 PM	Jul 16, 2024, 5:01:43 PM
Quality Check	0.0885 kg	Jul 16, 2024, 5:01:43 PM	Jul 16, 2024, 5:01:55 PM
Assemble Parts	120.0 kg	Jul 16, 2024, 5:01:55 PM	Jul 16, 2024, 5:01:55 PM
Quality Check	0.0804 kg	Jul 16, 2024, 5:01:55 PM	Jul 16, 2024, 5:02:05 PM
Paint Automobile	13.5 kg	Jul 16, 2024, 5:02:05 PM	Jul 16, 2024, 5:02:05 PM
Final Inspection	0.0643 kg	Jul 16, 2024, 5:02:05 PM	Jul 16, 2024, 5:02:13 PM
Deliver Automobile	90.0 kg	Jul 16, 2024, 5:02:13 PM	Jul 16, 2024, 5:02:13 PM

Total CO2 Emissions: **353.6124 kg**

BPMN Diagram:

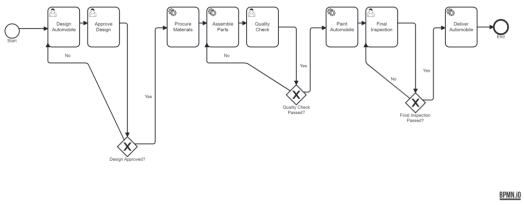


Figure 19: Emissions report

6 Discussion

This section will emphasize on how the presented system answered the research questions and fulfilled the drawn requirements. The design and software decisions will be taken into consideration, as well as the overall implementation of the system. The limitations that this project has with also be acknowledged. Building on the current capabilities and addressing the limitations of the sustainability calculator, several areas of future work can enhance the system, presented also in this section.

6.1 RQ1

The main focus of the sustainability calculator was to calculate CO2 emissions of the business process instance. To address the environmental metrics gathered, as described in design decision 6, CO2 emissions are the most significant GHG emissions, and are the most popular emissions needed for regulations. These CO2 emissions are calculated using emission factors that can be imported by the application user, ensuring compatibility with the company's needs. While providing a standardised method, it also ensures consistency and accuracy in emission calculations across various activities and resources. The calculation is being done by a computing service inside the main application module. The other modules were presented in the previous section, together with software decisions for choosing the best technological options.

Annotations play a significant role in data collection and calculation, as described in the domain view, by providing a detailed mapping of process tasks to the resources used. The annotations use a structured JSON format that allows for a useful assessment of each task's environmental impact, while capturing all relevant data.

Additionally, the front-end dashboard allows users to upload, edit, and visualize annotations and emission factors, making the system accessible and effective for use. It enables the tracking of emissions, ensuring that any changes in the process flow are immediately reflected in the emission data presented.

6.2 RQ2

To ensure accuracy and reliability in sustainability assessments, the design of the calculator incorporates the use of emission factors, as demonstrated in the decision to multiply fuel quantity by an emission factor (DD 8). It provides a consistent and reliable method for emission calculations. This approach, supported by multiple cited papers, offers a straightforward and validated technique for estimating CO2 emissions, ensuring the system's calculations are dependable and repeatable across different scenarios.

The system's modular architecture supports both scalability and adaptability, which are crucial for handling varying loads and extending the system to include new features. By using containers and decoupled components (DD 2), the system can be easily scaled horizontally or vertically, ensuring the stakeholders' scalability concerns.

The data validation and error handling mechanisms present in the project (e.g. Listing 11) ensure the integrity and accuracy of the data used for calculations. The accuracy of the system is further acknowledged by a number of unit tests for the main application services, testing the more important

parts of the system.

The combination of synchronous and asynchronous communication (DD 4) allows the system to process data efficiently and provide real-time updates. This capability is essential for maintaining the accuracy and relevance of the sustainability assessments. The decision to support multiple BPM engines (DD 5) ensures multi-platform compatibility enhancing the system's versatility. This flexibility allows the sustainability calculator to be used across different industries, maintaining accuracy and reliability regardless of the specific business context.

6.3 RQ 3

Integrating a sustainability calculator with existing BPM systems presents several challenges and opportunities. One of the main challenges is ensuring data synchronization between the annotations and BPMN models (DD 10). Any discrepancies can lead to inaccurate emission calculations, which undermines the reliability of the sustainability assessments. To address this, the system requires a careful check of the annotations and the business process diagram, to ensure that all data is consistent and up-to-date.

Another challenge is the complexity involved in integrating the system with multiple BPM engines (DD 5). Each engine has different APIs and event listeners, requiring custom integration modules to facilitate communication and data exchange. This complexity can increase the development time and resources required, but it also ensures that the system is versatile and can be used with a wide range of BPM platforms.

With regards to the organizational impact that the system introduces in a company's production environment, this study did not address such issues. This is addressed in the limitations section below.

6.4 Requirements

The sustainability calculator was developed with a focus on addressing both functional and non-functional requirements that were outlined in the approach section 4. Integrating the system with multiple BPM engines such as Camunda 7 and JBPM (FR-6) was essential for facilitating communication between the main application and the BPM engines, as this integration allows for real-time monitoring and updates (FR-2), providing continuous tracking of sustainability metrics. The system also benefits of the modular architecture, that supports scalability (NFR-1), enabling the system to handle increased loads and making sure it can integrate new features easier, thus maintaining high performance levels.

A significant emphasis was placed on real-time data processing and user interaction (FR-3). The use of Stomp WebSockets (FR-7) for real-time data communication ensures that any changes in the business process instance flow are immediately reflected on the user dashboard. This real-time capability is essential for fast decision-making and monitoring. The user-friendly dashboard, developed using React, provides an intuitive interface, easy to use for users to upload and edit annotations and visualize carbon emissions data. This interface ensures that users can easily interact with the system, thus enhancing usability and accessibility (NFR-2).

Data management (FR-4) was addressed through the use of PostgreSQL, ensuring that all relevant data, including process details, annotations, and emission factors, among other data tables used throughout the implementation, are stored and managed efficiently. This database system supports reliable data retrieval and updates. The emission calculation method (FR-5), which involves multiplying fuel quantity by predefined emission factors, ensures accurate and consistent emission estimates.

Communication between system modules (FR-7) is facilitated by RabbitMQ, which ensures reliable and efficient asynchronous communication. Other communication protocols like HTTP and WebSocket further ensure that the system can integrate with existing BPM tools and other external systems, enhancing interoperability (NFR-6).

The system's design prioritizes maintainability (NFR-3) and adaptability (NFR-4). By using the best practices such as dependency injection and service patterns, the system remains readable, modular and easy to maintain. This modular design also supports extensibility (NFR-8), allowing new features and functionalities to be added. Additionally, the use of containerization using docker containers ensures the system can be deployed across various operating systems, enhancing portability (NFR-7) and adaptability(NFR-4).

Reliability (NFR-5) and data integrity (NFR-9) are maintained through validation and error handling mechanisms. These features ensure that all data used in sustainability calculations is accurate and consistent, and any discrepancies are promptly identified and addressed.

6.5 Limitations

- **Emissions data**: The way to import the annotations file to be able to receive emissions data and integrate them in the business process may not be the best way for emission calculation. The issue this brings is that the data required for the annotations needs to be precise and correct in order to get good sustainability results. Incomplete or inaccurate data can lead to unreliable calculations and assessments. Moreover, it can be difficult to extract some of the data (e.g. resource consumption values) without making assumptions that may decline the quality of the calculation. However, this was considered to be the best way for mapping the emissions to the business tasks, with the best balance between flexibility, compatibility and complexity.

- **Lack of shared resources**: The project takes into consideration only atomic resources [19], that are all relevant devices consuming a considerable amount of fuel, for which the average consumption is known, or worth determining. The bottom up approach can also include shared resources, that are groups of consuming devices used by multiple actors (e.g. lightning system inside a warehouse housing multiple activities at the same time). By not including the shared type of resources, the calculator loses a significant amount of functionality. This limitation is addressed in the future work section 6.6

- **Only compatible with 2 BPM engines**: Although Camunda 7 and JBPM are some of the most popular business engines used, there are other open-source and enterprise business process execution engines that could have been implemented. This limitation is addressed in section 6.6

- **Scalability challenges**: Although the system is designed to support scalability, handling large volumes of data and extensive business processes can pose challenges. If the number of pro-

cesses and the volume of data is large, the system may require additional computational resources and data management strategies to maintain performance and accuracy.

- **Lack of practical testing**: The system was not integrated into any company's environment in order to get an accurate view of the impact on an organization. This limitation is understandable, as the system is not in a production phase at the moment, but by adding several new features like the ones presented in the future section below, it can become a fully fledged product usable throughout the industry

6.6 Future Work

- **Introducing shared resources**: The emission calculation can be enhanced by adding a new type of computed resources, presented in the previous section. An overhaul of the imported annotations is required, with new data needed for the process, task and resources. The annotations should include a total amount of fuel consumed, and mappings of atomic and shared resources. The shared resources should also be allocated using drivers [19] for a way to quantify the weight of each task.

- **Add compatibility with other BPM engines**: There are other business process execution engines(e.g. Activity, IBM Business Process Manager, Bizagi) that can be integrated with the system. This can increase the user base with the customers that use these engines in their process execution. Another engine that is worth to integrate is Camunda 8 with Zeebee, as it is the successor of Camunda 7, and will replace it in the near future. Camunda 7 was chosen for this project because the event driven microservice architecture of Camunda 8 posed issues for setting up the event listeners. In the future, Camunda 8 can be integrated by setting up the Camunda Exporter (that replaced the event listeners), or by using the zeebee worker environment to replicate the event listener inside the task execution step. For the latter step, the functionality is already implemented, but it is not in use, as the scope of the project was to implement a decoupled monitoring tool and modifications inside the task execution process were not wanted.

- **Add new use case for process execution**: At this moment the system listens to the process engine event listeners for process execution. This use case is very useful looking from a monitoring perspective, but from a testing perspective, when the user wants to find out the CO2 consumption of the process with a specific configuration, another use case can be starting the process from the system dashboard. This functionality increases the scope of the system and provides additional usages to it. For this to work, the process forms and diagram need to be imported to the engine integration module, and the module needs to be connected to the engine to be able to use method calls or HTTP requests to deploy the files and execute processes. Most of this functionality is already implemented and part of the code-base, but it is not in function. It involves file transfers by a shared docker volume for Camunda, and runtime deployed updated KJar for JBPM. Majority of the functionality is inside the engine integration modules, being built with this specific scope in mind. Altough at the moment their only functionality is channel communication and they could have been replaced by just communication between the engine and main application module

- **Introduce runtime process flow manipulation**: A possible new functionality of the system is using variables to alter the flow of the process during execution. Depending on the already existing CO2 emissions of the previous activities, if a set limit is reached, some activities could

be replaced with other less emitting counterparts (e.g. using an electric tool instead of a diesel tool, or transport by train instead of car/truck).

- **Extend the emissions pool**: Other GHG emissions can be calculated (e.g methane, nitrous oxide etc.) for more detailed sustainability calculations. The emission factors would need to be extended to also include the other emission types.

7 Conclusion

This thesis has focused on developing and integrating a sustainability calculator to assess and manage the environmental footprint of business operations effectively. The primary aim was to design a tool that not only calculates greenhouse gas emissions accurately but also supports ongoing sustainability efforts across various industries through real-time monitoring of the carbon footprint of business processes.

7.1 Summary of the thesis

The thesis began with an introduction 1 to the concept of sustainability and its importance in business strategies. It highlighted the need for tools like sustainability calculators to measure and manage the environmental impact of business processes accurately. The research questions focused on identifying key components and metrics for the calculator, ensuring its accuracy and reliability, and addressing the challenges and opportunities in integrating such a tool with existing BPM systems.

The background literature 2 section provided insights into Green BPM, GHG emissions, emission scopes, emission factors, and real-time GHG emissions monitoring. This review helped establish the theoretical foundation for the sustainability calculator

The literature review 3 presented a brief analysis of filtered existing literature on the topic and drew a number of important conclusions used as decisions throughout the project, such as measurement of CO2, real-time monitoring and BPM integration.

The approach 4 section detailed the stakeholders, requirements, design decisions, and user perspectives. Functional and non-functional requirements were outlined, and key design decisions were made regarding the use of containers, communication protocols, and the choice of BPM engines. The user perspective focused on the needs of enterprise customers, system users, and operations teams.

The solution section 5 provided an architectural overview, software decisions, and implementation details. The system architecture was designed for modularity and scalability, using containerization and real-time data processing. Detailed descriptions of the BPM engines, integration modules, main application, and front-end components were provided.

The discussion section 6 analyzed how the system addressed the research questions and met the outlined requirements. It highlighted the system's accuracy in emission calculations, modular architecture, the importance of data validation and efficient data processing and real-time updates. In additon, it acknowledged several limitations, including the reliance on accurate data for annotations, the system's current support for only atomic resources, and compatibility with only two BPM engines. Furthermore, it included possible future areas of development of the project, such as introducing shared resources for more comprehensive emission calculations, expanding compatibility to additional BPM engines like Camunda 8 and IBM BPM, enabling process execution from the dashboard for enhanced testing, implementing runtime process flow manipulation, and extending the emissions pool to include other GHG emission.

7.2 Results of the system

The sustainability calculator achieved several significant results:

- **Integration with BPM Engines**: The system successfully integrated with two popular BPM engines, Camunda 7 and JBPM. This integration enables the monitoring of business processes and the real-time calculation of CO2 emissions. The system's flexibility to work with multiple BPM engines increases its applicability across different organizations.
- **Real-Time Emission Tracking**: The implementation of real-time data processing using Stomp WebSockets and RabbitMQ allows the system to provide immediate feedback on emission data. This capability is important for businesses to have a real-time view of their current business processes carbon emissions
- **User Interaction and Data Management**: The front-end dashboard, developed using React, offers a user-friendly interface for uploading, editing, and visualizing annotations and emission factors. It also offers the possibility to receive a report of the current CO2 emissions of the business process instance. These features ensure that users can easily interact with the system, enhancing usability and accessibility. The use of PostgreSQL for data management ensures efficient storage and retrieval of process details, annotations, emission factors, and other data objects.
- **Accurate Emission Calculations**: The system uses a reliable method of multiplying fuel quantity by predefined emission factors to calculate CO2 emissions. This approach, validated by existing research, ensures that the emission calculations are accurate and consistent across different scenarios. The accuracy of the system is further supported by unit tests for the main application services, ensuring the reliability of the key functionalities.

7.3 Final thoughts

The development of this sustainability calculator marks a significant advancement in integrating sustainability into business process management. By providing a tool that enables real-time monitoring and accurate calculation of CO2 emissions, this project contributes to the broader goal of promoting sustainable business practices and addressing global climate change. The system's robust architecture, real-time capabilities, and user-friendly interface ensure that it can effectively support sustainability efforts across various industries, paving the way for more environmentally conscious business operations. This project can be considered as a starting point in BPM sustainability calculations, with a significant number of possible future areas of development, enhancing the project with multiple new features.

Bibliography

[1] C. Houy, M. Reiter, P. Fettke, P. Loos, K. Hoesch-Klohe, and A. Ghose, "Advancing business process technology for humanity: Opportunities and challenges of green bpm for sustainable business activities," *Green Business Process Management: Towards the Sustainable Enterprise*, pp. 75–92, 2012.

[2] C. Houy, M. Reiter, P. Fettke, and P. Loos, "Towards green bpm - sustainability and resource efficiency through business process management," pp. 501–510, 2010.

[3] D. Couckuyt and A. Van Looy, "Green bpm as a business-oriented discipline: a systematic mapping study and research agenda," *Sustainability*, vol. 11, no. 15, p. 4200, 2019.

[4] S. Seidel, J. Recker, J. Brocke, Lopes, S. Oliveira, Australia, J. Recker, and J. vom Brocke, "Green business process management," pp. 3–13, 2012.

[5] S. R. Gohar, M. Indulska, *et al.*, "Environmental sustainability through green business process management," *Australasian Journal of Information Systems*, vol. 24, 2020.

[6] S. Bom, H. Ribeiro, and J. Marto, "Sustainability calculator: A tool to assess sustainability in cosmetic products," *Sustainability*, 2020.

[7] J. Mancebo, F. García, O. Pedreira, and M. A. Moraga, "Bpms-game: Tool for business process gamification," pp. 127–140, 2017.

[8] O. Oduyemi and M. Okoroh, "Building performance modelling for sustainable building design," *International journal of sustainable built environment*, vol. 5, pp. 461–469, 2016.

[9] A. Lisovsky, "Sustainable development and business process management," *Strategic decisions and risk management*, 2019.

[10] D. Couckuyt and A. V. Looy, "An exploration of green business process maturity based on ecolabels," *Bus. Process. Manag. J.*, vol. 27, pp. 1999–2020, 2021.

[11] T. Rozman, A. Draghici, and A. Riel, "Achieving sustainable development by integrating it into the business process management system," pp. 247–259, 2015.

[12] A. Abdi, S. Taghipour, and H. Khamooshi, "A model to control environmental performance of project execution process based on greenhouse gas emissions using earned value management," *International Journal of Project Management*, vol. 36, pp. 397–413, 2018.

[13] A. Hernandez Gonzalez, C. Calero, D. Perez Parra, and J. Mancebo, "Approaching green bpm characterisation," *Journal of Software: Evolution and Process*, vol. 31, no. 2, p. e2145, 2019.

[14] D. Couckuyt and A. Van Looy, "A systematic review of green business process management," *Business Process Management Journal*, vol. 26, no. 2, pp. 421–446, 2019.

[15] D. Couckuyt and A. Van Looy, "An empirical study on green bpm adoption: Contextual factors and performance," *Journal of Software: Evolution and Process*, vol. 33, no. 3, p. e2299, 2021.

[16] N. Opitz, H. Krüp, and L. Kolbe, "Green business process management – a definition and research framework," *2014 47th Hawaii International Conference on System Sciences*, pp. 3808–3817, 2014.

[17] I. Arto and E. Dietzenbacher, "Drivers of the growth in global greenhouse gas emissions.," *Environmental science & technology*, vol. 48 10, pp. 5388–94, 2014.

[18] D. Ahlers, J. Krogstie, P. Driscoll, H.-E. Lundli, S.-J. Loveland, C. Rothballer, and A. Wyckmans, "Supporting municipal greenhouse gas (ghg) emission inventories using business process modeling: a case study of trondheim municipality," in *Business Process Management Workshops: BPM 2016 International Workshops, Rio de Janeiro, Brazil, September 19, 2016, Revised Papers 14*, pp. 416–427, Springer, 2017.

[19] A. Ghose, K. Hoesch-Klohe, L. Hinsche, L.-S. Le, *et al.*, "Green business process management: A research agenda," *Australasian Journal of Information Systems*, vol. 16, no. 2, 2010.

[20] C. Kennedy, J. Steinberger, B. Gasson, Y. Hansen, T. Hillman, M. Havránek, D. Pataki, A. Phdungsilp, A. Ramaswami, and G. Méndez, "Methodology for inventorying greenhouse gas emissions from global cities," *Energy Policy*, vol. 38, pp. 4828–4837, 2010.

[21] G. Zhang, M. Sandanayake, S. Setunge, C. qing Li, and J. Fang, "Selection of emission factor standards for estimating emissions from diesel construction equipment in building construction in the australian context.," *Journal of environmental management*, vol. 187, pp. 527–536, 2017.

[22] E. Perez-Minana, P. Krause, and J. Thornton, "Bayesian networks for the management of greenhouse gas emissions in the british agricultural sector," *Environmental Modelling & Software*, vol. 35, pp. 132–148, 2012.

[23] E. Lugato, M. Zuliani, G. Alberti, G. Vedove, B. Gioli, F. Miglietta, and A. Peressotti, "Application of dndc biogeochemistry model to estimate greenhouse gas emissions from italian agricultural areas at high spatial resolution.," *Agriculture, Ecosystems and Environment*, vol. 139, pp. 546–556, 2010.

[24] A. Dias, D. Lemos, X. Gabarrell, and L. Arroja, "Comparison of tools for quantifying the environmental performance of an urban territory," *Journal of Industrial Ecology*, vol. 22, 2018.

[25] J. S. Guzman, D. Fuenmayor-Gonzalez, B. Turrel, and L. Dauce, "Leveraging real time operational data to reduce greenhouse gas emissions," *Day 2 Tue, October 03, 2023*, 2023.

[26] V. Iorio, D. Farina, S. Racca, L. D. Forno, and F. Concina, "Advanced analytic tool development for ghg emissions monitoring in well operations," *Day 1 Mon, May 01, 2023*, 2023.

[27] S. Mahesh and M. Hibbett, "Preserve: Sensor based low-cost emissions management for small and medium farms," *2023 IEEE Green Energy and Smart Systems Conference (IGESSC)*, pp. 1–7, 2023.

[28] C. Wohlin, "Guidelines for snowballing in systematic literature studies and a replication in software engineering," in *Proceedings of the 18th international conference on evaluation and assessment in software engineering*, pp. 1–10, 2014.

[29] A. Magdaleno, L. Duboc, and S. Betz, "How to incorporate sustainability into business process management lifecycle?," pp. 440–443, 2016.

[30] J. P. Carvalho, L. Bragança, and R. Mateus, "Optimising building sustainability assessment using bim," *Automation in Construction*, 2019.

[31] G. Kabra, V. Ghosh, and A. Ramesh, "Enterprise integrated business process management and business intelligence framework for business process sustainability," pp. 228–238, 2018.

[32] F. Jalaei, F. Jalaei, and S. Mohammadi, "An integrated bim-leed application to automate sustainable design assessment framework at the conceptual stage of building projects," *Sustainable Cities and Society*, vol. 53, p. 101979, 2020.

[33] J. Recker, M. Rosemann, A. Hjalmarsson, and M. Lind, "Modeling and analyzing the carbon footprint of business processes," in *Green business process management: Towards the sustainable enterprise*, pp. 93–109, Springer, 2012.

[34] "Iso 25010," 3 2019.

[35] Y. Majanne, T. Korpela, and T. Uotila, "Eu emission trading related co2 monitoring in power plants," *IFAC Proceedings Volumes*, vol. 47, pp. 1361–1366, 2014.

[36] Y. Nassar, M. Salem, K. R. Iessa, I. AlShareef, K. Ali, and M. A. Fakher, "Estimation of co2 emission factor for the energy industry sector in libya: a case study," *Environment, Development and Sustainability*, vol. 23, pp. 13998 – 14026, 2021.

[37] J. A. de Carvalho, A. D. de Castro, G. H. Brasil, P. A. de Souza, and A. Z. Mendiburu, "Co2 emission factors and carbon losses for off-road mining trucks," *Energies*, 2022.