Ibn Tofail University
Faculty of Sciences, Kenitra

End of Study Project Thesis

Artificial Intelligence and Virtual Reality Master

AI-Assisted Solutions for Dynamic Chart and Form Generation

Host establishment: E-Quality Engineering

Developed by: Mr. Abdessamie Nachi

Supervised by: Mrs. Khaoula Boukir(ENSC KENITRA (UIT))
Mr. Youness Laassouly (E-Quality Engineering)

19 November 2024, **before the jury composed of:**

- Mrs. Raja Touahni (FS Kenitra (UIT))
- Mr. Rochdi Messoussi (FS Kenitra (UIT))
- Mrs. Khaoula Boukir (ENSC KENITRA (UIT))
- Mr. Anass Nouri (FS Kenitra (UIT))
- Mrs. Souad Eddarouich (CRMEF Rabat)

Academic year 2023/2024

ACKNOWLEDGEMENT

It is with sincere gratitude that I dedicate this section to express my deep gratitude to all the people who were of indispensable support for the success of this project. On this special occasion, I would like to express my gratitude to the coordinator of the Master IARV, Mrs **RAJAA TOUAHNI**, for the quality of the training she gave us, as well as for her support and accompaniment throughout my journey.

My sincere thanks also go to my internal supervisor, Mrs. **KHAOULA BOUKIR**, for her valuable guidance throughout this work, Without her help and support throughout this project, it would not have been possible. I would also like to thank the members of the jury for agreeing to examine and evaluate my work.

I also express my gratitude to Mr. **YOUNESS LAASSOULI**, my external supervisor, for his warm welcome, his effective supervision and his support throughout this enriching professional experience.

With great respect, I express our great gratitude to the faculty of the Master artificial intelligence and virtual reality for having taken a keen interest in our training, and for having devoted most of their time, attention and energy to us in a framework pleasant complicity and respect.

I would like to express my deepest gratitude to my family. To my mother and father, for their unconditional love, constant support and sacrifices that allowed me to get where I am today. To my brothers and my sisters, for their encouragement, their complicity and their presence by my side throughout this journey. Your support has been an inexhaustible source of motivation and comfort. I would also like to express my gratitude to all those who have contributed directly or indirectly to the completion of this project. Their involvement and support were essential, and I warmly thank them for their valuable collaboration.

Abstract

This report summarizes my final-year internship at E-Quality Engineering in Casablanca, part of our master's degree in Artificial Intelligence and Virtual Reality (IARV) at the Faculty of Sciences, Ibn Tofail University (UIT) in Kenitra. The project aimed to develop a model capable of understanding natural language and accurately translating it into charts. Titled "Betterchart", and also the developement of an AI system capable of helping the user fill forms, by autcompleting some fields. Titled "AI Assistant", this project sought to automate the conversion of textual descriptions into chart configuration using a provided schema. The ability to automatically translate natural language into charts is crucial for solutions that place great importance on data visualization. My project at E-Quality Engineering is part of this technological innovation effort, aiming to improve the accessibility and efficiency of creating charts, while also keeping in mind issues such as inference time and data privacy. In addition to this main project, during my internship, I also worked on other tasks including Frontend and backend development, and server mangement. To achieve our objectives, I started with an in-depth analysis of the needs and functionality Specifications, followed by a data collection phase and executing fine-tuning methods for large language models (LLM).

Keywords: E-QUALITY ENGIENNERING, BETTERFLY, CONFIG, DATABASE, CHARTS, FORMS, QHSE, AUTOCOMPLETE, ISO.

Résumé

Ce rapport résume mon stage de fin d'études chez E-Quality Engineering à Casablanca, dans le cadre de notre master en Intelligence Artificielle et Réalité Virtuelle (IARV) à la Faculté des Sciences, Université Ibn Tofail (UIT) à Kénitra. Le projet visait à développer un modèle capable de comprendre le langage naturel et de le traduire avec précision en graphiques. Intitulé "Betterchart", et également le développement d'un système d'IA capable d'aider l'utilisateur à remplir des formulaires, en complétant automatiquement certains champs. Intitulé "AI Assistant", ce projet visait à automatiser la conversion de descriptions textuelles en configuration de graphiques à l'aide d'un schéma fourni. La capacité de traduire automatiquement le langage naturel en graphiques est cruciale pour les solutions qui accordent une grande importance à la visualisation des données. Mon projet chez E-Quality Engineering s'inscrit dans cet effort d'innovation technologique, visant à améliorer l'accessibilité et l'efficacité de la création de graphiques, tout en gardant à l'esprit des problèmes tels que le temps d'inférence et la confidentialité des données. En plus de ce projet principal, pendant mon stage, j'ai également travaillé sur d'autres tâches, notamment le développement front-end et back-end, et la gestion des serveurs. Pour atteindre nos objectifs, j'ai commencé par une analyse approfondie des besoins et des fonctionnalités des spécifications, suivie d'une phase de collecte de données et d'exécution de méthodes de finetune pour les LLMs.

Mots clés : E-QUALITY ENGIENNERING, BETTERFLY, CONFIG, DATABASE, CHARTS, FORMS, QHSE, AUTOCOMPLETE, ISO.

Table of Contents

List of Figures

List of Tables

List of abbreviations

Abréviation	**Signification**
NL	Natural language
LLM	Large language model
IA	Artificial Intelligence
ML	Machine Learning
DL	Deep learning
SQL	Structured Query Language
API	Application Programming Interface
ISO	International Organization for Standardization
QHSE	Quality, Health, Safety and Environment
Dataset	Collection of related sets of information
Schema	Instructions for creating a database

General Introduction

In an era where data-driven decision-making is paramount, the ability to efficiently visualize complex datasets has become an essential component of organizational success. As businesses increasingly rely on data analytics to inform strategies and operational improvements, the demand for advanced tools that can streamline this process has surged. E-QUALITY ENGINEERING, a leader in ISO certification, training, and consultancy services, recognizes this critical need within the realm of Quality, Health, Safety, and Environment (QHSE) management systems. Through its innovative digital platform, Betterfly, the company has been at the forefront of transforming traditional QHSE processes by integrating automation, real-time data processing, and comprehensive analytics.

Despite these advancements, a significant challenge persists in the domain of data visualization within QHSE management: the manual generation of optimized charts that accurately represent complex datasets. Existing solutions often require users to possess specialized technical skills or involve cumbersome processes that impede real-time data interpretation and decision-making. Moreover, while Large Language Models (LLMs) have shown promise in automating chart generation, they present limitations such as privacy concerns, reliance on static data inputs, and inefficiencies in processing time—particularly when handling sensitive or continuously updating datasets.

The structure of this document will be as follows:
The first chapter will present the general context of the project, including an introduction to E-Quality Engineering, The second chapter will be dedicated to the various technologies adopted during the project. The third chapter will describe the process and the various stages of the Betterchart project's realization, detailing the results obtained, the project's progress, and suggestions for improvements. Finally The fourth chapters are dedicated on my second project with E-Quality Engineering and the development of an Auto completion model

Chapter 1

Company introduction

1.1 Introduction

In this introductory chapter, we present E-QUALITY ENGINEERING, a leading provider of ISO certification, training, and consultancy services. The organization is committed to driving operational excellence and compliance across industries through the integration of innovative digital solutions. We explore the company's core values, services, and its flagship product, Betterfly, which revolutionizes traditional QHSE (Quality, Health, Safety, and Environment) management systems through automation and real-time data analytics.

The chapter further delves into the role of cutting-edge technologies, including AI and machine learning, in enhancing the capabilities of the Betterfly platform, particularly in the realm of automated data visualization. Through an examination of chart generation methodologies and the challenges associated with LLM (large language model) implementations, we outline the project's goal of creating an AI-driven solution to optimize chart creation. This sets the stage for understanding how E-QUALITY ENGINEERING and Betterfly aim to elevate QHSE standards, providing businesses with more effective and efficient tools for compliance and performance monitoring.

1.2 E-QUALITY ENGINEERING

1.2.1 Overview

E-QUALITY ENGINEERING is an expert provider of training, implementation, and certification services related to ISO standards, committed to propelling organizational excellence and compliance across multiple sectors. The consultancy integrates innovative digital solutions to modernize and enhance traditional QHSE management systems.

1.2.2 Services and Expertise

Offering a wide range of services from ISO certification assistance for standards like ISO 9001, ISO 14001, and ISO 45001 to bespoke training and consultancy, the firm adopts a

Figure 1.1: E-QUALITY ENGINEERING

digital-first approach. This method helps simplify the certification process, making ISO standards more accessible and less daunting for businesses.

1.2.3 Experience and Impact

The enterprise's proficiency is evidenced by its expansive portfolio, featuring successful projects that have notably boosted clients' operational standards and compliance across diverse industries, demonstrating significant international reach and impact.

1.3 Solution QHSE

1.3.1 Definition and Importance

QHSE systems are integral for businesses to manage quality, health, safety, and environmental performance effectively. These systems ensure compliance with legal and regulatory requirements while fostering a safe, efficient, and sustainable operational environment.

1.3.2 Betterfly's Role

Betterfly transforms QHSE management through its advanced digital platform, which employs automation, real-time data processing, and comprehensive analytics to streamline traditional QHSE processes, thereby reducing manual workloads and increasing accuracy.

1.4 Fundamental Values

1.4.1 Core Values

The operations of E-QUALITY ENGINEERING and Betterfly are rooted in core values of integrity, innovation, and customer focus. These principles ensure a consistent, ethical approach across all services and interactions.

1.4.2 Culture and Ethics

The commitment to ethical practices and continual improvement is central to the firm's operations, aiming to provide services that not only meet but exceed industry standards and enhance client satisfaction.

1.5 Betterfly

Figure 1.2: Betterfly

1.5.1 Company Overview

Conceived to tackle inefficiencies in QHSE management, Betterfly offers a technologically advanced solution that simplifies complex compliance processes and enhances organizational performance through digital automation.

1.5.2 Product and Services

The platform features automated process workflows, intuitive form management systems, and sophisticated data visualization tools designed to support organizations in achieving seamless compliance with ISO standards.

1.6 Data In Betterfly

1.6.1 Data Management

Betterfly prioritizes the security and integrity of data with rigorous controls and protocols to ensure data is handled securely and efficiently, protecting against unauthorized access and data breaches.

1.6.2 Use of Data

The platform leverages collected data to facilitate continuous improvements and strategic decision-making. Advanced analytics and customized reporting capabilities provide comprehensive insights that drive operational efficiency and strategic planning.

1.7 Technologies Used - Betterfly

1.7.1 Tech Stack

Utilizing cutting-edge technologies like Node.js, React, and Docker, Betterfly is built for scalability and robustness, accommodating the diverse and complex demands of QHSE management.

1.7.2 Integration and Scalability

Designed for high adaptability and scalability, Betterfly integrates seamlessly with existing systems and evolves with organizational growth, continually enhancing its capabilities and service offerings.

1.8 Charts In Betterfly

1.8.1 Chart Types and Functions

Betterfly utilizes a variety of dynamic charts, including customizable line and bar charts, to visualize complex QHSE data effectively. These visual tools are vital for monitoring compliance, assessing performance trends, and facilitating clear communication of results.

1.8.2 Customization and Interaction

The platform's charts offer extensive customization options and interactive features, allowing users to modify visual data presentations according to specific requirements and delve deeply into the data for meticulous analysis.

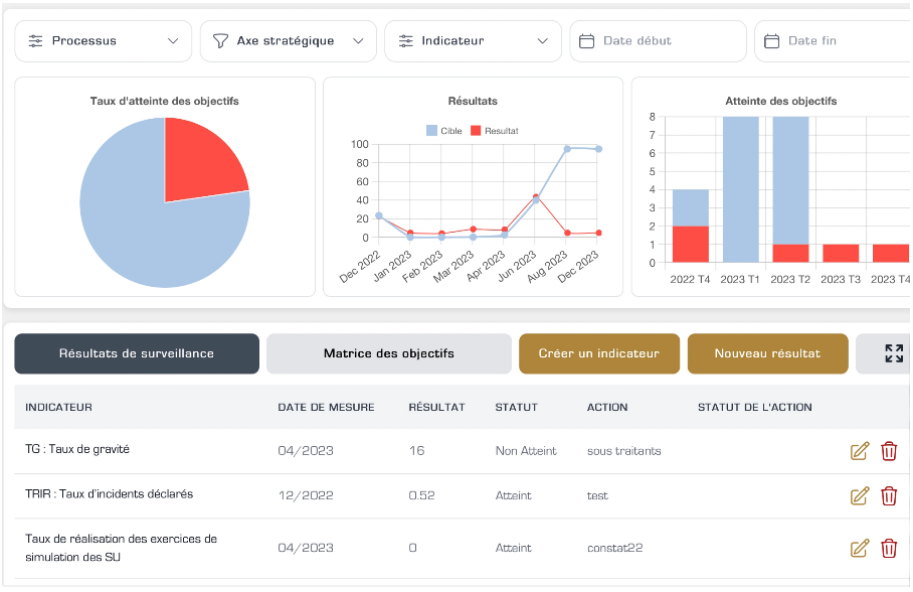

Figure 1.3: Charts In Betterfly

1.9 AI Assistants in Betterfly

1.9.1 Role of AI

AI assistants are integral to Betterfly, enhancing functionality by automating data analysis, chart generation, action planning, and custom report building. These AI systems use machine learning algorithms to analyze historical data, predict trends, and suggest optimal actions.

1.9.2 Adaptability and Customization

The AI-driven features of Betterfly ensure rapid adaptation to new automated processes and easy customization of charts and reports, catering to the specific needs of different QHSE domains. This adaptability is crucial for maintaining up-to-date and relevant tools for users.

1.9.3 Future Scope

The future expansion of Betterfly includes broader management domains such as energy management and ESG reporting. This progression aligns with global sustainability trends, equipping organizations to meet the emerging demands of environmental and social governance alongside traditional QHSE requirements.

1.10 Betterchart problem statement

As part of this final year project, the work required consisted of developing and implementing an AI-driven solution for automatically generating optimized charts based on data input. This mission included several key stages:

Data collection: Gather a diverse and representative dataset consisting of various types of data along with examples of optimal chart representations. This included researching existing datasets, as well as creating new, project-specific datasets to ensure the solution addresses a wide range of use cases and data complexities.

Data preparation: Perform data cleaning and preprocessing to ensure the collected data is suitable for training the AI model. This stage involved normalizing the data, handling missing or inconsistent information, and converting it into a format that can be used by machine learning algorithms.

Selection of the chart generation methodology: Evaluate different AI techniques and approaches for automating chart generation. This step included researching various machine learning models.

Model selection: Identify and select the most appropriate AI model for generating data-driven charts. This involved comparing different models based on criteria such as their ability to understand context, generate accurate visualizations, resource requirements, and processing speed, while taking into account the complexity and volume of the data.

Implementation and experimentation: Implement the selected AI model, training it using the prepared data. This stage also involved testing the model's performance, generating various types of charts, and conducting evaluations based on accuracy, relevance, and visual appeal. Iterative adjustments were made to fine-tune the model and optimize results.

Documentation and analysis of results: Document all stages of the project and analyze the effectiveness of the generated charts. This included comparing the AI-generated charts with manually created ones to assess quality, as well as formulating conclusions on the success of the project. Recommendations were also provided for future improvements and additional areas for research.

The work carried out in this project demonstrates the feasibility of using AI to automate the process of chart creation, offering a solution that saves time, improves chart accuracy, and enhances decision-making by providing high-quality visual data representations.

1.10.1 Similar Applications

Large language models (LLMs) are widely used to generate charts, making the process of data visualization easier and more efficient. Many advanced LLMs, like OpenAI's GPT and similar models, have the capability to understand natural language queries and produce charts based on data or even descriptions. These models can generate a variety of charts, like bar charts, line graphs, pie charts, and scatter plots. This approach simplifies chart creation, especially for non-technical users, as it eliminates the need for manual coding and understanding complex visualization libraries.

Several LLM-powered platforms, such as those integrated with B.I. tools, allow users to interact with data through simple queries and commands, and the system generates appropriate visualizations on the fly.

1.10.2 Limitations

While LLMs have impressive abilities when it comes to creating charts, there are some clear limitations. A key issue is that users must manually provide the necessary data each time they want a chart. This means the model doesn't have real-time access to live or automatically updated data, which can be a major downside in situations where data is constantly evolving. Whenever the dataset changes, the user has to manually submit the updated information to the model to produce a new chart.

Another drawback of using large language models (LLMs) for chart generation involves privacy concerns, particularly when users need to input sensitive or proprietary data. To generate charts, users must share their data with the LLM, which can raise potential data security risks.

Moreover, the time it takes for LLMs to process and generate charts is another limitation. LLMs are resource-intensive, so generating a chart from input data can take a

significant amount of time, especially with larger or more complex datasets. This issue becomes more noticeable when using LLMs locally on machines with limited computing power. As a result, real-time or near-real-time chart generation can become impractical, especially in cases where frequent updates or immediate insights are necessary. For users needing continuous, up-to-date visualizations, this delay can negatively impact the overall efficiency of the process.

In situations where real-time data visualization is crucial, this inefficiency can be particularly problematic. For instance, in business settings where key metrics are constantly being updated, users must continuously query the LLM with the latest data to ensure that the charts reflect the most current information. Additionally, since LLMs rely on static data inputs, they aren't designed to support ongoing data analysis without constant manual intervention. This limitation reduces their effectiveness in scenarios that require continuous monitoring or automatic chart updates. Overcoming this issue would require integrating LLMs with live data streams or automated refresh mechanisms, an area still in need of further development.

1.11 Solution

A potential solution to the limitations posed by LLM-based chart generation is to use the LLM to generate the configuration or template of the chart rather than directly producing the chart itself. In this approach, the LLM takes natural language input from the user, understands the data context, and outputs a detailed configuration file. This configuration can then be used by a separate chart-rendering engine to generate the final visual representation. By separating the chart design process from the data itself, this solution ensures that the user retains full control over the data and can update it independently of the LLM. Additionally, this approach minimizes privacy concerns, as the LLM only needs to process the table schema, not the raw data itself, offering a more secure method for handling sensitive information.

1.12 Approach

The approach for implementing this solution involves using the LLM to interpret the user's instructions and generate a chart configuration file that specifies the type of chart, axis labels, legends, color schemes, and other visual elements. The LLM would analyze the natural language query, determine the most appropriate chart type based on the user's intent, and then generate a code snippet or configuration template that a rendering engine, such as Plotly, Matplotlib, or a similar tool, could use to visualize the data. This approach also allows for greater flexibility: users can modify the configuration manually if needed and rerun the chart with updated data without having to re-query the LLM. By automating the chart configuration process while decoupling the LLM from direct data handling, this method addresses privacy concerns, reduces inference time, and allows for

more dynamic updates to charts.

1.13 Conclusion

In conclusion, this chapter introduces E-QUALITY ENGINEERING and its commitment to enhancing organizational excellence through innovative QHSE solutions. E-QUALITY ENGINEERING stands out as a leader in ISO certification, training, and consultancy services, leveraging cutting-edge digital technologies like the Betterfly platform to streamline and modernize QHSE management. With a focus on automation, real-time analytics, and data security, Betterfly enhances operational efficiency and compliance across industries.

The chapter also outlines the development of AI-driven solutions for automated chart generation, which addresses critical challenges in data visualization, such as efficiency, accuracy, and real-time updates. While current LLM-based models demonstrate significant potential for generating insightful charts, limitations like privacy concerns and real-time data access remain. However, proposed solutions, such as using LLMs to generate chart configurations rather than full visualizations, provide a promising path forward. This approach ensures better data control, improved privacy, and more dynamic, flexible visualizations, paving the way for the future of automated QHSE management and compliance reporting.

Chapter 2

Tools and Technologies used

2.1 Introduction

In this chapter, we will explore the various tools and technologies utilized in the development of this project, with a focus on how they contributed to achieving efficient workflows, powerful capabilities, and seamless collaboration. As the project involves complex machine learning models, dynamic web interfaces, and backend systems, a broad range of software and platforms was employed to address these needs.

We begin by examining Large Language Models (LLMs), the backbone of the AI-driven solutions within the project. LLMs allow for natural language processing, which simplifies user interactions with data and systems. The chapter will provide a detailed overview of LLMs, discussing their definitions, types, and applications, as well as the state-of-the-art models used in the project.

Next, we will delve into the programming languages and frameworks that played essential roles in the development process, such as Python, React, Node.js, and SQL. Each language and framework was selected for its unique strengths in handling tasks ranging from machine learning to web development. We will also cover platforms like Hugging Face, Google Colab, and cloud services such as Azure and AWS Amplify, which supported model training and deployment.

The chapter will also review critical industrialization tools like Docker, Git, and GitHub, which ensured that the project was efficiently managed, version-controlled, and deployed across different environments.

By the end of this chapter, you will gain a comprehensive understanding of the tools and technologies that powered the project, enabling its smooth execution and the development of an advanced AI-driven solution.

2.2 LLMs

In today's data-driven world, the ability to interact efficiently with complex systems is becoming increasingly vital across various sectors. However, for many users, the technical

complexity of these systems can act as a barrier. This is where large-scale language models play a pivotal role, offering a way to simplify interactions through natural language interfaces. These models provide users with the tools to engage with advanced systems without requiring deep technical knowledge, thus broadening accessibility and enhancing decision-making capabilities.

In this chapter, we will first provide a definition of LLMs, outlining their key characteristics and how they work. Then we will examine the different types of LLMs available, highlighting their unique features and applications. Next, we will discuss the state of the art in LLM development, drawing on the latest research and advancements in the field.

We will also explore the practical applications of LLMs, showing how these models are being applied in various industries. then focus on the current limitations and challenges associated with LLMs, identifying key areas that need improvement. Finally, we will present some benchmarks and comparisons to evaluate their performance, leading to the conclusion which summarizes key insights and future directions for this evolving technology.

2.2.1 Definition of LLMs

Large Language Models (LLMs) are advanced machine learning models designed to understand and generate human language with a high degree of fluency and contextual awareness. Built on neural networks, LLMs are trained on vast amounts of textual data, allowing them to capture and model the complex relationships between words, phrases, and concepts. The key innovation behind LLMs is the Transformer architecture, which enables the model to process language in a parallelized manner, leading to greater efficiency and scalability. This architecture allows LLMs to consider the context of words over long passages of text, which is crucial for generating coherent and contextually relevant responses. LLMs, such as OpenAI's GPT, have revolutionized natural language processing (NLP) by making it possible to automate tasks like text generation, summarization, translation, and even code creation, with near-human-level accuracy.

2.2.2 Types of LLMs

LLMs can be categorized based on their size, training objectives, and the tasks they are designed to perform. Some common types include general-purpose LLMs like GPT-3 or BERT, which are trained on a broad range of text and can be fine-tuned for specific tasks, and task-specific LLMs, which are optimized for particular applications such as question-answering or machine translation. Another distinction is between monolingual LLMs, which operate in a single language, and multilingual LLMs, which can handle multiple languages simultaneously. Additionally, there are decoder-only models (such as GPT) focused on generating text and encoder-decoder models (such as BERT and T5), which are better suited for understanding and transforming text, making them ideal for tasks like summarization or translation.

2.2.3 State of the art

The current state of the art in LLMs is characterized by models that are larger, more sophisticated, and capable of achieving remarkable results across a variety of natural language tasks. Some of the leading models include OpenAI's GPT-4, Google's PaLM, and Meta's LLaMA. These models often consist of billions or even trillions of parameters, allowing them to perform complex reasoning, generate high-quality text, and engage in multi-turn conversations. Recent advances have also seen improvements in fine-tuning and instruction-following capabilities, where models are trained not just on raw text but also on specific instructions to improve their utility in real-world applications. Additionally, fine-tuned variants like Codex (which specializes in code generation) have demonstrated how LLMs can be adapted to niche domains. Innovations such as few-shot learning and prompt engineering have further enhanced LLM performance, making them increasingly versatile and practical in both research and industry.

2.2.4 Applications of LLMs

LLMs have a wide range of applications across various industries. In business, LLMs are used to automate customer support through chatbots, generate content for marketing, and assist in decision-making by providing data insights from natural language queries. In healthcare, LLMs can summarize medical research, assist with diagnostics, and provide patient care recommendations based on large datasets. They are also heavily used in software development, where models like Codex can generate code snippets based on simple language prompts, allowing developers to speed up the coding process. Furthermore, LLMs are employed in fields like legal document analysis, translation services, and academic research, where they help process and synthesize vast amounts of text into actionable information.

2.2.5 Current limitations and challenges

Despite their advanced capabilities, LLMs face several significant limitations. One of the primary challenges is their requirement for vast computational resources during both training and inference, which makes deploying large models in real-time applications costly and resource-intensive. Moreover, LLMs lack real-time adaptability; they cannot access or update with live data, which means their output is based on the static data they were trained on. Privacy is another key concern, as many LLMs require access to sensitive data to generate relevant responses, raising security issues, especially when proprietary or personal information is involved. Additionally, hallucination remains a problem, where models generate factually incorrect or nonsensical text with confidence. Lastly, LLMs can sometimes struggle with understanding highly specialized or domain-specific information unless explicitly fine-tuned for that purpose.

2.2.6 Some benchmarks and comparisons

To measure the effectiveness and performance of LLMs, various benchmarks are used across different natural language tasks. For instance, models are tested on datasets like SuperGLUE, which evaluates understanding across a wide range of tasks such as question-answering, reading comprehension, and logical inference. SQuAD (Stanford Question Answering Dataset) is commonly used to benchmark models for question-answering accuracy. Additionally, coding models like OpenAI's Codex are compared using benchmarks such as HumanEval, which assesses the accuracy of generated code based on function descriptions. When comparing models, factors like parameter count, training dataset size, and inference speed are also critical. For example, GPT-4 has shown superior performance on multi-turn dialogue and creative text generation, while BERT models excel in tasks requiring deeper text understanding, such as text classification and sentiment analysis.

2.3 Languages

2.3.1 Python

Python is a versatile and widely-used programming language known for its simplicity and readability. It is commonly used in data science, machine learning, and AI due to its rich ecosystem of libraries like NumPy, Pandas, TensorFlow, and PyTorch. In this project, Python is essential for implementing machine learning models and managing data tasks.

2.3.2 React

React is a JavaScript library for building dynamic, interactive user interfaces, mainly for web applications. It allows developers to create reusable components and efficiently manage state, enabling responsive and user-friendly interfaces. In this project, React is used to build the front-end where users interact with data and visualize AI-generated charts. Its flexibility and speed ensure a seamless user experience, with charts updating and rendering smoothly based on user input.

2.3.3 NodeJs

Node.js is a server-side JavaScript runtime built on Chrome's V8 engine, commonly used for building scalable web applications. In this project, Node.js acts as the backend, handling server requests and facilitating interaction between the front-end (React) and the

AI-driven model on the server. It also manages API calls, processes user data, and passes configurations to the chart generation libraries.

2.3.4 SQL

SQL is the standard language for querying and managing relational databases. In this project, it enables efficient retrieval and manipulation of stored data. SQL is used to extract data from databases, which is then utilized by the AI model or for generating visualizations. Writing dynamic and complex SQL queries ensures that the necessary data is available for processing, analysis, and visualization.

2.3.5 GraphQL

GraphQL is a query language for APIs that allows clients to request specific data, reducing API calls and improving performance. In this project, GraphQL enables efficient data retrieval from the server, especially when dealing with large datasets. Its flexibility and ability to query nested relationships make it ideal for managing and structuring data exchanges between the front-end and backend, particularly for complex data visualizations.

2.4 Platforms

2.4.1 Hugginface

Hugging Face is a leading platform for NLP and machine learning models, offering pre-trained models and tools for fine-tuning. In this project, Hugging Face's model hub is utilized to access advanced language models like Llama The platform also provides tools to fine-tune models, ensuring they are optimized for the project's specific requirements.

2.4.2 Google Colab

Google Colab is a cloud-based platform that provides a free environment for running Jupyter notebooks, offering access to GPUs and TPUs without complex setups. In this project, Colab is used during the development and training of AI models, enabling efficient experimentation with various configurations and datasets. Its integration with popular machine learning libraries and ease of collaboration make it essential for rapid prototyping and testing.

2.4.3 Azure

Microsoft Azure is a cloud computing platform offering services such as virtual machines, databases, and machine learning environments. In this project, Azure is used to host back-end services, run AI models, and store datasets. Its scalability and robust infrastructure enable efficient large-scale data processing and model inference, ensuring reliable, secure, and high-performance service for end-users.

2.4.4 AWS Amplify

AWS Amplify is a suite of tools and services from Amazon Web Services that streamlines the development and deployment of full-stack applications. In this project, AWS Amplify is used to deploy the front-end web application and manage backend services like authentication and data storage. Its integration with other AWS services, such as S3 for storage and Lambda for serverless functions, ensures a smooth workflow for handling data and managing user interactions with the visualization tool.

2.5 Frameworks and Libraries

2.5.1 ChartJS

Chart.js is a popular open-source library used for creating responsive, interactive, and customizable charts and graphs in web applications. It supports various chart types such

as bar, line, pie, and scatter charts. In our project, Chart.js is utilized to render the charts based on the configurations generated by the model, providing users with visual insights from their data.

2.5.2 Datasets

The Datasets library is a versatile tool for accessing and managing large datasets in various formats, widely used in machine learning tasks. In this project, Datasets is used to organize and process datasets used for training and validating the AI model.

2.5.3 Transformers

Transformers is a library developed by Hugging Face that provides pre-trained models and utilities for natural language processing tasks. It enables efficient model fine-tuning and deployment. In our project, Transformers is used to power the model that interprets user inputs and generates corresponding chart configurations based on data schema.

2.5.4 Unsloth

Unsloth is a Python utility designed to optimize model loading times and manage large models more efficiently. In this project, Unsloth is used to streamline model loading and improve its performance.

2.5.5 Torch

Torch (PyTorch) is a deep learning framework widely used for building and training machine learning models. PyTorch's dynamic computation graph and flexibility make it ideal for research and production. In our project, Torch is used to implement and fine-tune the machine learning models.

2.5.6 Accelerate

Accelerate is a library from Hugging Face designed to simplify distributed training across multiple devices such as CPUs, GPUs, and TPUs. In our project, Accelerate was leveraged during the training phase to speed up model training on different hardware setups, reducing training time significantly.

2.5.7 Scipy

SciPy is an open-source Python library used for scientific and technical computing. It provides tools for optimization, integration, interpolation, and more. In this project, SciPy supports data processing tasks and assists in handling mathematical operations required during the model training process.

2.5.8 Xformers

Xformers is a library for memory-efficient Transformers, designed to improve performance and reduce memory usage when dealing with large models. In our project, Xformers was employed to optimize the Transformer model, ensuring that chart configurations were generated efficiently without overwhelming system resources.

2.5.9 Safetensors

Safetensors is a tool that provides a safer and more efficient format for storing and loading machine learning model weights. In this project, Safetensors ensures that the model weights are securely stored and quickly accessible during runtime, improving the overall safety and performance of the system.

2.5.10 peft

PEFT (Parameter-Efficient Fine-Tuning) is a library that enables fine-tuning models with fewer parameters, saving computation time and resources. In our project, PEFT was used to fine-tune the Transformer model responsible for generating chart configurations, making it more efficient and adaptable.

2.5.11 sklearn

Scikit-learn is a widely-used library for machine learning, providing simple and efficient tools for data analysis and model building. In our project, Scikit-learn was used for preprocessing data, evaluating the model's performance, and applying statistical techniques during the training process.

2.5.12 PyPDF2

PyPDF2 is a library for working with PDF files in Python. It enables PDF manipulation tasks such as splitting, merging, and extracting text. In our project, PyPDF2 was utilized to extract and organize data from PDF documents, such as ISO standards, which were later used to train the auto-complete models

2.6 Communications and meeting tools

2.6.1 Google Meet

Google Meet is a widely used video conferencing platform that supports remote team communication and collaboration with features like screen sharing, live captions, and meeting recording. In this project, Google Meet was utilized for team meetings, client interactions, and remote collaboration. Its integration with Google Workspace made scheduling and managing meetings efficient, ensuring seamless communication throughout the project.

2.6.2 Notion

Notion is an all-in-one productivity tool that integrates note-taking, task management, databases, and collaboration features. In this project, Notion served as the central hub for project management, documentation, and teamwork. The team used it to organize meeting notes, assign tasks, track progress, and maintain project documentation, while enabling the creation of a structured and collaborative workspace, enhancing communication and collaboration throughout the project.

2.7 Industrialization tools

2.7.1 Docker

Docker is an open-source platform that automates the deployment of applications in lightweight, portable containers, bundling the application and its dependencies to ensure consistent performance across different environments. In this project, Docker was used to containerize the application, ensuring a uniform configuration across all team members and deployment environments. This reduced compatibility issues and helped with the development, testing, and deployment processes.

2.7.2 Git

Git is a distributed version control system commonly used for tracking changes in source code during development. It enables multiple developers to collaborate by managing branches, handling merges, and maintaining a detailed history of changes. In this project, Git was used to manage the source code, track modifications, and facilitate team collaboration. Its versioning ensured smooth coordination and provided rollback capabilities when needed.

2.7.3 Github

GitHub is a web-based platform built on Git that hosts repositories and offers collaboration tools such as pull requests, code reviews, and issue tracking. In this project, GitHub was used to host the codebase and manage team contributions. It served as a central hub for storing code, tracking issues, and facilitating discussions via GitHub Issues and Pull Requests, ensuring a smooth and organized development workflow.

2.8 Conclusion

In conclusion, this project leveraged a comprehensive suite of tools, technologies, and languages to ensure successful execution and collaboration. The programming languages used, including Python, React, Node.js, SQL, and GraphQL, formed the backbone of the development process. Python's versatility in handling machine learning and data visualization, combined with React's capability to build dynamic front-end interfaces and Node.js managing the backend, enabled the team to create a robust and efficient system. SQL and GraphQL played pivotal roles in managing and querying data, ensuring seamless data communication between the application layers.

In terms of communication and collaboration, tools such as Google Meet and Notion facilitated consistent team interactions and effective project management. Google Meet allowed the team to hold regular meetings, ensuring that everyone stayed aligned on project goals and deadlines. Notion acted as the central repository for documentation, task management, and team coordination, improving transparency and workflow efficiency.

Industrialization tools like Docker, Git, and GitHub ensured that the development environment was stable and scalable. Docker was instrumental in containerizing the application, making deployment across different environments uniform and straightforward. Git and GitHub facilitated version control, allowing multiple team members to collaborate effectively, manage code changes, and ensure that all work was tracked and versioned.

Additionally, platforms such as Hugging Face, Google Colab, Azure, and AWS Amplify supported the model development and deployment processes. Hugging Face provided access to state-of-the-art models, while Google Colab offered a flexible environment for experimentation and model training. Azure and AWS Amplify were key in deploying the backend services and ensuring the application's scalability and security in a cloud environment.

The combined use of these tools, languages, and platforms ensured that the project was completed with high efficiency, collaboration, and innovation. Each tool played a crucial

role in addressing different challenges, from development and deployment to communication and coordination. Together, they contributed to the creation of a well-rounded, functional system capable of delivering the intended AI-driven chart generation solution.

Chapter 3

BetterChart

3.1 Introduction

BetterChart aims to revolutionize the way users generate data visualizations by harnessing the power of artificial intelligence. The goal of this project is to provide a solution that allows users to create optimized, relevant, and visually appealing charts with minimal effort. By leveraging AI to analyze the data and generate chart configurations, BetterChart simplifies the process, making data visualization more accessible to users with varying levels of technical expertise. The result is a tool that not only saves time but also enhances the quality of insights derived from data by generating the most appropriate chart type for the context.

3.2 Approach followed

The development of BetterChart was an extensive process that focused on two critical components to enhance the overall user experience and functionality.

The first key aspect involved implementing the DynamicChart system on the front-end, primarily utilizing React. This implementation was not only focused on providing a seamless and responsive user experience but also on ensuring that the system could efficiently connect to the back-end database. This connection was achieved using Node.js. The DynamicChart system is designed to dynamically receive outputs from a language learning model (LLM) and then translate that information into a clear, interactive visualization using Chart.js, a robust library for creating versatile and customizable charts. The entire process ensures that the users can easily visualize complex data through automatically generated charts, enhancing comprehension and decision-making processes.

The second essential component was focused on the training of the underlying model that powers BetterChart. This model was carefully designed and trained to understand and NL text inputs and then generate the corresponding chart configuration. This complex task involved fine-tuning the model to ensure it could accurately interpret varied user inputs, ranging from simple requests to more complex data-driven queries, and con-

vert them into structured formats that could be seamlessly integrated into the Chart.js visual output. This allows users to effortlessly translate natural language descriptions into precise, visually engaging data representations without needing extensive technical expertise.

Together, these two components – the front-end implementation using React and Node.js, and the sophisticated model training for NL to chart configuration generation – form the backbone of the BetterChart system, making it an innovative and user-friendly tool for data visualization.

3.2.1 System Architechture overview

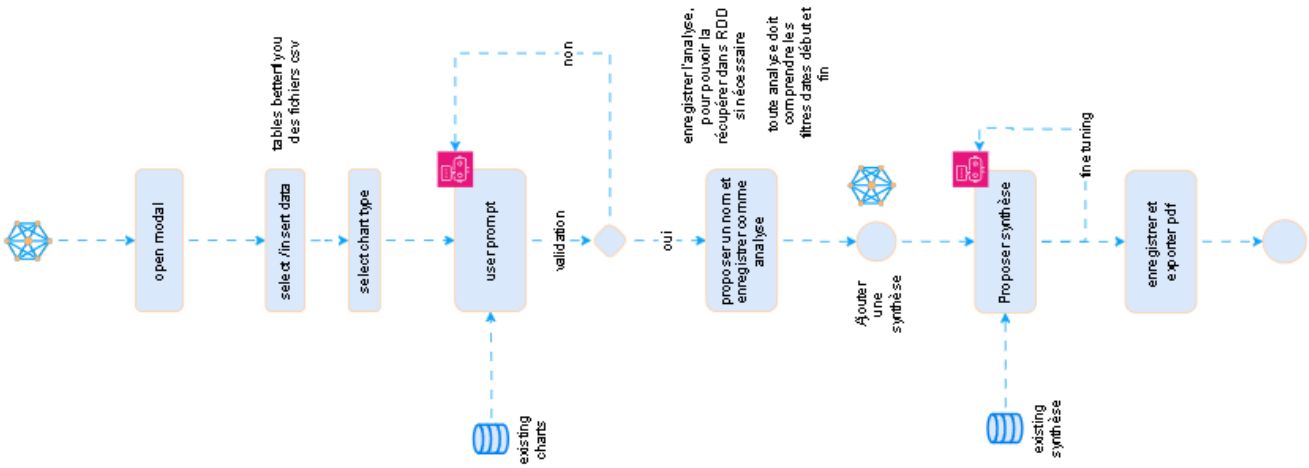

Figure 3.1: BetterChart Workflow

3.3 Setting up BetterChart - (DynamicChartJS)

Before diving into the AI phase of the project, we dedicated a significant portion of the internship duration to carefully designing and developing an improvement on the ChartJS Library capable of understanding and generating chart configurations. This deliberate approach allowed us to gain deep insights into the structure and types of configuration data that would need to be generated by the system.

As a result of this process, we successfully built a control panel that enables users to create charts by selecting options manually, providing them with full control over the customization process. The time invested in this system's development ensured that we thoroughly understood the configuration requirements, ultimately allowing us to refine and streamline the data generation process.

This groundwork has proven invaluable, as it not only prepared us for the AI phase of the project but also gave the users of the application to be able to create charts even without the use of AI. By establishing this solid foundation, we ensured that the system

is both user-friendly and robust, capable of handling complex chart customization needs with ease.

3.4 Training

The training phase played a pivotal role in transitioning our system from manual configuration to an AI-driven solution. During this phase, we focused on teaching the model to understand and replicate the logic behind chart configurations based on user inputs. By utilizing a carefully curated dataset of chart examples and configurations, we trained the model to generate accurate and customizable chart setups.

3.4.1 Data Collection

The data collection process involved manually gathering and generating the necessary configurations from previous chart examples and online sources. Additionally, a significant portion of the dataset was created manually to ensure diversity and comprehensiveness. The data was structured in a specific format to capture the relationship between user inputs, the corresponding table schema, and the resulting chart configurations.

For each entry in the dataset, the first input represented the user input, while the second input defined the table schema, outlining the relevant columns for the chart. The output consisted of the full chart configuration. Below is an example of how the data was formatted:

- **Input:** Context - "Distribution des utilisateurs par Login"
- **Table Schema:** "['id', 'Login', 'Prenom', 'Nom', 'Email', 'Langue', 'Processus', 'isAdmin', 'createdAt', 'updatedAt', 'userRoleId', 'version', 'lastChangedAt', 'deleted']"
- **Output:** X: "Login", chartType: "doughnut", isStacked: false, isFiltered: false, formula: none

This structured approach to data collection provided a strong foundation for training the model, ensuring it could handle a wide range of chart configurations based on various user inputs and table schemas.

3.4.2 Data Augmentation

To enhance the diversity and robustness of the dataset, we employed several data augmentation techniques. One key method was translating the user input into different languages, which allowed the model to handle a more diverse set of inputs and adapt to various linguistic contexts. Additionally, we applied reformulation techniques to rephrase user inputs while maintaining the same underlying intent.

By introducing these variations, we significantly increased the size and variety of the dataset, enabling the model to generalize better across different input formats and user

expressions. These augmentation techniques were crucial for ensuring that the system can process a wide range of user inputs and generate accurate chart configurations in different scenarios.

3.4.3 Data processing

Once the data was collected and augmented, the next crucial step was data processing. This involved cleaning and standardizing the dataset to ensure consistency and eliminate any irregularities. All user inputs, table schemas, and output configurations were carefully formatted to adhere to a uniform structure, facilitating efficient training of the model.

We also handled missing or incomplete data by applying preprocessing techniques such as filling missing values, filtering out irrelevant fields, and ensuring that all necessary components (inputs, schemas, and outputs) were present in each data entry. This step ensured that the model would be trained on high-quality, well-structured data, optimizing its performance and reducing errors during the configuration generation process.

Finally, the processed data was divided into training and validation sets, which enabled us to evaluate the model's performance and fine-tune it throughout the training phase.

3.4.4 Choice of model

LLaMA 3: This model is known for its efficiency and ability to handle tasks with minimal computational resources, making it a strong candidate our chart generation task. Its architecture is designed to be lightweight while still providing deep insights from a variety of contexts, especially in structured data environments.

GPT-3.5: With its extensive training on a broader dataset, GPT-3.5 excels at language understanding and can generate more contextually aware summaries. It can infer complex relationships within the data and articulate them in a natural, coherent manner, making it ideal for graph synthesis tasks.

3.5 Prompt Engineering

Prompt engineering is a technique used to guide and enhance the performance of large language models (LLMs) like GPT by carefully designing the input prompts. Rather than altering the underlying model, this method focuses on crafting precise instructions or questions that direct the model to generate the desired output more effectively.

In our project, prompt engineering was employed to ensure that the LLM consistently returned outputs in the specific format we required, allowing us to tailor the responses to fit the exact needs of our system. This approach optimized the model's ability to generate accurate chart configurations based on user input.

3.6 Finetuning

3.6.1 What is Fine-tuning?

Fine-tuning is a key technique in AI where a pre-trained model is adapted to perform a specific task. Instead of training a model from scratch, which requires a lot of data and time, fine-tuning allows us to take an existing model that has already learned general patterns and adjust it for a more focused application.

For example, a language model trained on a large general dataset can be fine-tuned to better understand legal or medical documents. This process saves time and resources while improving the model's performance on a specific task.

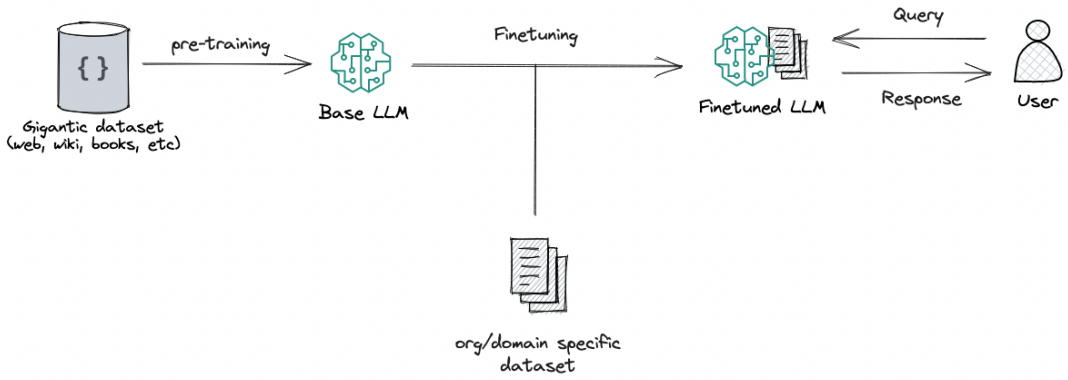

Figure 3.2: Finetuning example

3.7 Results

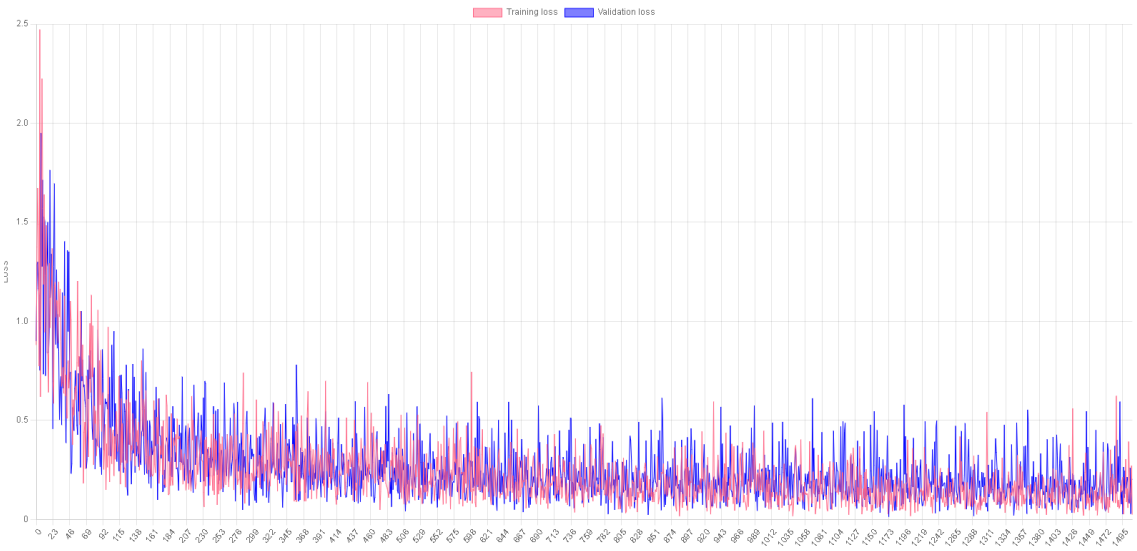

Figure 3.3: Gpt 3.5 Loss

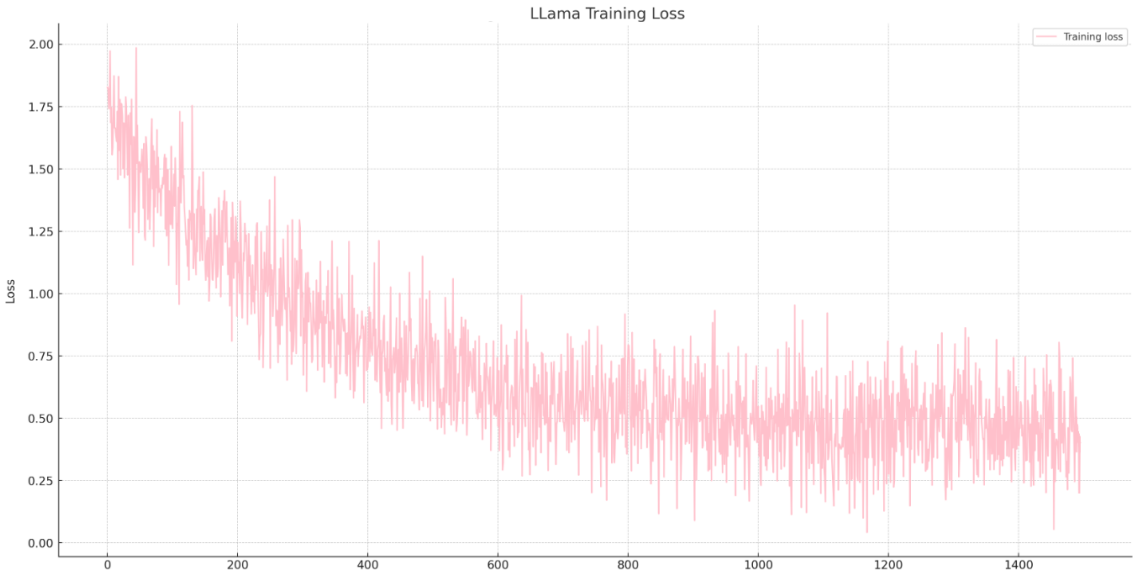

Figure 3.4: Llama3 Loss

3.8 Evaluation

To evaluate the performance of our fine-tuned models, we adopted a human feedback-based approach, which allowed us to obtain a qualitative and accurate assessment of the results. We established a rigorous evaluation process that consisted of the following steps:

Preparation of a varied set of database schemas
Preparation of a diverse set of Titles from already existing charts.
Generation of Chart configuration by each model in response to these questions.
Manual evaluation of the accuracy of the obtained results.
This method enabled us to effectively compare the performance of the different models.

Model	**Speed**	**Accuracy of chart generated**
LLaMA 3 7B finetuned	Fast	Average
GPT-3.5 finetuned	Moderate	High

Table 3.1: Comparison of Model Performance

This comparative analysis provided valuable insights into the strengths and weaknesses of each model, helping us identify the most suitable model for automating the generation of chart configurations.

3.9 Result interpretation

The model's output consists of complete chart configurations based on user inputs, such as chart type, axes labels, and additional properties like stacking or filtering. These configurations are generated to fit seamlessly into our charting library, ensuring that the charts reflect the user's preferences.

3.9.1 Chart Type

This refers to the type of chart generated by the AI model. Options can include various types such as pie charts, bar charts, doughnut charts, line charts, and others. The appropriate chart type is selected based on the nature of the data and how it is best represented visually.

3.9.2 X

This parameter defines the first label, which is used to split the data. It represents the primary axis or category by which the data is separated in the chart. For example, in a bar chart comparing sales across different regions, the "X" axis could be the regions.

3.9.3 Y

This parameter defines the second label used to split the data. It is typically used for numerical values that are plotted against the categories on the X-axis. In the sales example, the "Y" axis could represent the sales figures for each region.

3.9.4 X Group

This field represents how the first label (X) is grouped. It allows the AI model to aggregate data into larger categories or clusters, offering a clearer view of trends or patterns. For example, the data on the X-axis could be grouped by year or by trimester.

3.9.5 Y Group

Similar to X Group, this field groups the Y-axis data, allowing for aggregated calculations or summaries, such as total sales for each region or average ratings for a product category.

3.9.6 isStacked

This setting controls whether a bar chart should be stacked or not. A stacked bar chart layers data categories on top of one another in a single bar, which helps in comparing part-to-whole relationships.

3.9.7 isFiltered

This option controls whether the data should be filtered before being displayed. Filtering allows users to focus on specific subsets of the data, such as showing only data from a certain year or excluding outliers.

3.9.8 Filter Key

The filter key defines the specific attribute or field used for filtering the data. For instance, if the dataset includes multiple regions, a filter key like "Region" would allow the chart to display data from a particular region only.

3.9.9 Filter Value

This value determines the specific condition applied to the filter key. For example, if the filter key is "Region," the filter value could be "North America" to display data only from that region.

3.9.10 Filter Operator

The filter operator defines how the data is filtered. Common operators include "equal to," "less than," "greater than," or other comparison operations that narrow down the data set based on the filter key and value.

3.9.11 Formula

This field defines how the data is expressed within the chart. It can display percentages, raw values, or more complex mathematical expressions, such as sum, count, or averages.

This feature allows for more sophisticated data presentations, especially in cases where mathematical aggregation or manipulation is needed.

3.9.12 Colors

The AI model is also capable of automatically assigning colors to different labels within the chart. This ensures that each category or data point is easily distinguishable, enhancing the readability and visual appeal of the chart.

3.10 Similarity check

In some cases, the results generated by the AI model do not perfectly align with the expected data schema. To address this issue, I implemented a similarity check that returns a "fuzzy set" to help identify close matches between the generated results and the required schema.

The similarity check uses fuzzy matching techniques to compare the generated data with the predefined schema. Instead of requiring an exact match, the system calculates the degree of similarity between the two sets of data. This approach is particularly useful when there are minor discrepancies in formatting, labeling, or structure. By implementing this similarity check, the system can:
* Improve Flexibility: Handle minor differences in data schema and still return relevant results.
* Enhance Accuracy: Identify the closest possible matches, even when exact matches are not available.
* Optimize Workflow: Reduce the need for manual adjustments, as the fuzzy set provides suggestions that are nearly identical to the expected schema.
The similarity check is a valuable addition that ensures more robust and adaptable output, even when data mismatches occur.

3.11 Graph synthese

3.11.1 Introduction

In this section, we discuss the implementation of a base model, without fine-tuning, capable of generating syntheses based on the data returned from the charts. Leveraging powerful models like LLaMA 3 and GPT-4o, this approach uses pre-trained models to interpret data and provide concise summaries or insights without the need for additional training on task-specific data.

3.11.2 Models

Both LLaMA 3 and GPT-4o are large language models with strong capabilities in natural language understanding and generation. They have been pre-trained on diverse datasets,

enabling them to generate meaningful interpretations based on a wide variety of inputs, including the graphical data returned from the charts. By tapping into their vast general knowledge, these models can generate text syntheses that explain trends, highlight significant data points, and summarize key findings from complex visualizations.

3.11.3 Implementation Approach

The process starts by feeding the raw chart data into the model, along with a prompt requesting a synthesis. For example, the prompt might be structured as: "Generate a synthesis of the key insights from the pie chart with the following data ..."

Although the base models are not fine-tuned for specific datasets, their large-scale training makes them adaptable. They are capable of understanding numeric and categorical data, as well as making inferences about the relationships between them. For example, they can detect correlations, spot anomalies, or explain proportional changes, even when the chart data is novel or specific to a particular domain.

3.12 Integrating

The model is hosted in a cloud-based environment, providing scalability and availability for real-time interactions. To facilitate communication between the model and the frontend, an API was created, allowing the frontend to send user inputs and the associated data schema to the model.

The model processes the input and generates the appropriate chart configuration, which is then returned to the frontend. The frontend uses the dynamicChart library to render the chart based on the configuration, allowing users to instantly visualize the results tailored to their specifications.

Additionally, the generated chart configuration is saved in a dedicated table within a database. This ensures that each chart, along with its configuration, is stored for future reference. Since the configuration is dynamic, it is linked to live data, meaning that even if the underlying data changes, the saved configuration remains valid. The chart will automatically update to reflect new data without the need to re-engage the LLM, streamlining the process and keeping the charts up-to-date with minimal intervention.

3.13 The Model in Action in Betterfly

In this section, we showcase how the AI-powered chart generation model is integrated into the Betterfly application, providing users with an intuitive experience for creating dynamic charts. Below are some screenshots illustrating the user interface, the workflow of generating charts, and how the model interacts with the system.

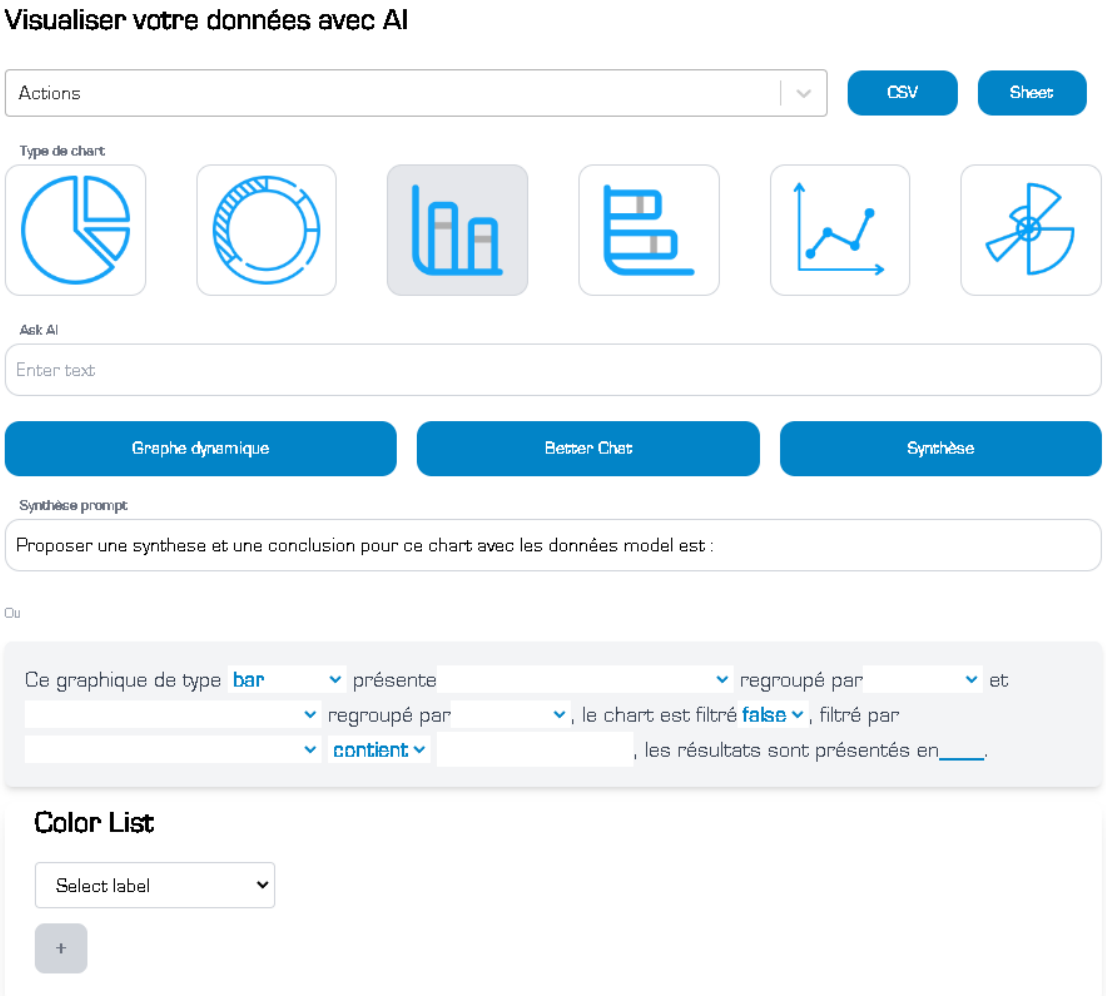

Figure 3.5: User Input Interface - The control panel where users can manually input their preferences for chart type, labels, and other properties.

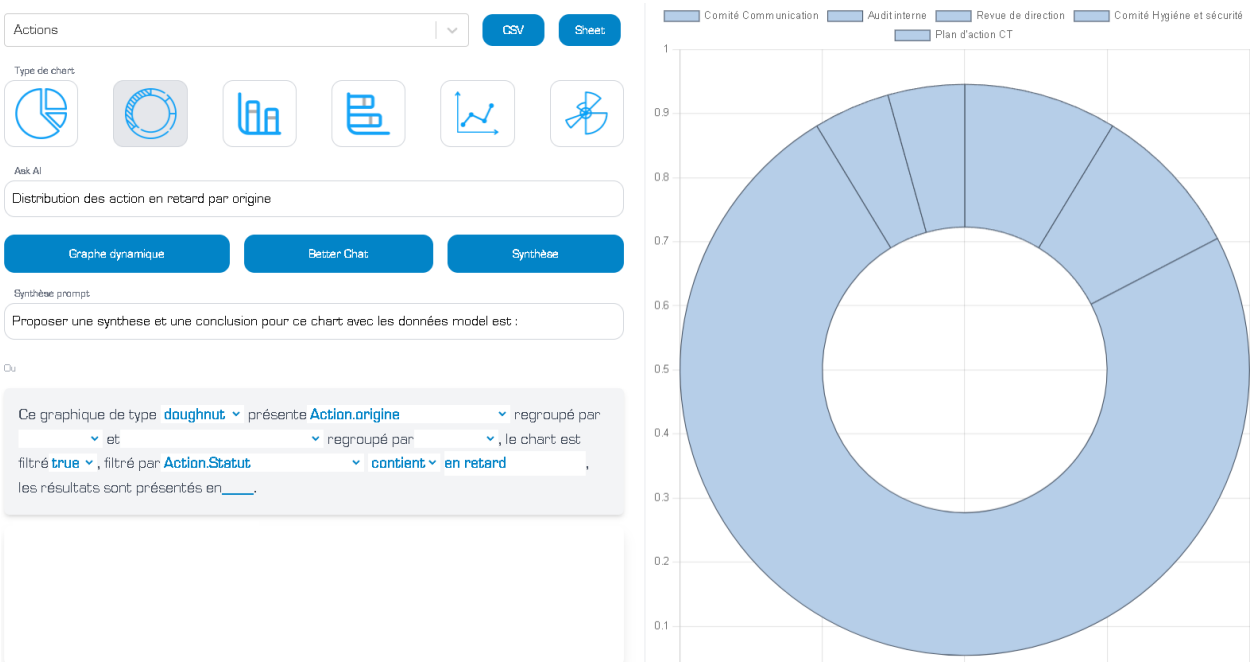

Figure 3.6: Chart Generation - A dynamically generated chart rendered in the frontend using the dynamicChart library, based on user preferences and linked data.

title character varying	code character varying	x character varying	y character varying	xgroup character varying	ygroup character varying	isstacked boolean	isfiltered boolean	filterkey character varying
Contexte		description	type			false	false	
Atteinte des objectifs par Axe stratégique		AxesStrategique.Axe	ResultatIndicateur.statut			false	false	
Atteinte des objectifs par processus		Processus.ProcessusTitle	ResultatIndicateur.statut			false	false	
SOR PAR CATÉGORIE		Catégories	Type SOR			false	true	Date de création
CATÉGORIES D'ENGINS PAR PROJET		projet	Catégories Engin			false	false	
JOURS PERDUS PAR TRIMESTRE		Date de création	calculated	Trimestre		false	false	
INDUCTIONS PAR PROJET PAR TRIMESTRE		Date de création	projet	Trimestre		false	false	
Nombre de modifications/enregistrement par semaine et par mod...		Logger.date	Logger.model	Semaine		false	false	
RÉSERVES OPR PAR PROJET PAR SEMAINE (2024)		Date de création	projet	Semaine		false	true	Date de création
INSPECTIONS PAR TYPE PAR SEMAINE (2024)		Date de création	Catégories	Semaine		false	true	Date de création
SORs par Projet par Semaine (2024)		Date de création	projet	Semaine		false	true	Date de création
JSA par Projet par semaine		Date de création	projet	Semaine		false	true	Date de création
NOMBRE DE PERMIS PAR TYPE PAR SEMAINE (2024)		Date de création	Types Permis	Semaine		false	true	Date de création

Figure 3.7: Configuration Saving - Example of the generated chart configuration saved in the database

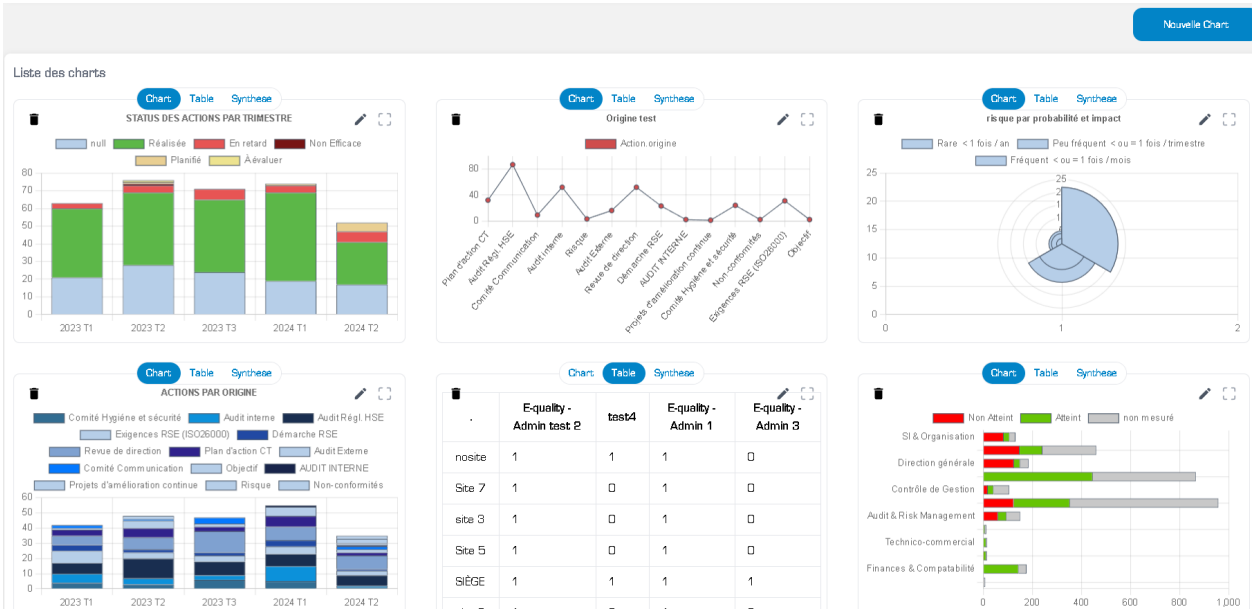

Figure 3.8: Configuration Saving - Example of the generated chart configuration saved in the web app

3.14 Conclusion

In conclusion, the BetterChart project represents a significant advancement in automated data visualization, combining AI-powered chart generation with a user-friendly interface. By utilizing advanced natural language processing (NLP) models, such as LLaMA 3 and GPT-3.5, BetterChart allows users to effortlessly generate optimized chart configurations from simple natural language inputs. The system's ability to interpret user inputs and automatically generate visually appealing and contextually relevant charts makes it an innovative tool for both technical and non-technical users.

The project's development approach, which included the implementation of a dynamic front-end using React and Chart.js, combined with sophisticated back-end support from Node.js, ensures seamless data visualization and efficient interaction with AI models. The training and fine-tuning of models for NL input to chart configuration generation further enhance the system's capability to interpret complex user requests and deliver accurate visual representations.

Furthermore, BetterChart addresses key challenges in data visualization, such as privacy concerns and real-time data handling, by decoupling the AI model from direct data manipulation. The integration of a similarity check system and data augmentation techniques strengthens the tool's flexibility, allowing it to handle diverse input formats and adapt to various contexts.

Ultimately, BetterChart not only streamlines the chart creation process but also enhances decision-making through high-quality data visualizations, providing an invaluable asset for businesses and individuals alike in a range of domains.

Chapter 4

AI Assistant

4.1 Introduction

The AI Assistant project is designed to enhance the user experience by intelligently generating form fields based on already typed-out inputs. In the context of Digitalisation QSE (Quality, Safety, Environment), the assistant aims to streamline the form completion process, reducing manual entry and improving efficiency. This system functions similarly to auto-complete, but it goes beyond simple text suggestions by dynamically predicting and generating entire form fields that align with user input and the specific context of the form.

Forms within the QSE domain are often complex, requiring extensive details on procedures, compliance, and safety measures. These forms may include interdependent fields where the completion of one section determines the content of subsequent sections. The AI Assistant tackles this challenge by learning from previously entered data, recognizing patterns, and generating relevant fields in real-time.

Moreover, since the primary purpose of the system is to improve the user experience, it also offers the ability to regenerate and reformulate already typed-out form fields. This feature allows users to update or adjust existing inputs effortlessly, ensuring flexibility and adaptability when new information is available or changes need to be made. By enabling both dynamic generation and reformulation of fields, the AI Assistant provides a seamless, user-centric approach to form completion.

The assistant is powered by a machine learning model trained to understand the relationships between different fields in a form, allowing it to generate accurate and contextually appropriate suggestions. By automating this process, the AI Assistant aims to reduce human error, save time, and provide a more intuitive experience when interacting with complex forms.

This chapter will explore the architecture and design of the AI Assistant, how it integrates with existing systems, and the techniques used to train and deploy the model effectively.

4.2 Similar Applications

The concept of generating form fields based on already typed content has been explored in several domains, with a variety of applications aimed at improving user experience and productivity. Here are a few notable examples:

4.2.1 Google Forms and Auto-complete Features

Google Forms offers basic auto-complete functionalities, such as auto-suggestions for answers in multiple-choice questions or drop-down fields. Although limited in its contextual understanding, it assists users by providing quick selections based on previous inputs. However, it doesn't dynamically generate new fields based on the context of previously filled data, highlighting a gap that our AI-driven solution seeks to address.

4.2.2 Intelligent CRMs

Customer Relationship Management (CRM) systems like Salesforce and HubSpot have started integrating AI-based tools to help auto-fill and suggest data in fields, such as customer details or project information. These tools often rely on existing databases and previous interactions to predict and auto-complete fields, enhancing the user's ability to manage clients efficiently.

4.2.3 Medical Record Systems

In healthcare, electronic medical record (EMR) systems have begun to utilize predictive text and auto-complete functionalities to assist healthcare professionals in filling out patient information. These systems analyze past data and current inputs to generate suggestions.

4.3 Limitations

Despite the advancements in similar applications, there are several limitations we must consider for our AI assistant project.

4.3.1 Domain-Specific Challenges

The existing models we discussed, such as those used in CRM, healthcare, and e-commerce, are tailored to specific domains, providing targeted solutions for their respective fields. While these models demonstrate the potential of auto-complete functionalities, they do not directly align with the needs of our domain. The unique requirements and structure of forms, which often involve compliance-related fields, safety checks, and detailed procedural documentation, make it necessary to develop a customized solution.

4.3.2 Lack of Training Data

One of the primary limitations we face is the lack of domain-specific training data. Existing datasets from other domains, like e-commerce or healthcare, are not directly applicable to the structure and terminology used in our processes. To train a robust model, we need comprehensive datasets that reflect real-world forms, user behaviors, and field dependencies. The scarcity of such data hinders the initial performance of the model.

4.4 Training

The training process for the AI Assistant is crucial to ensure that the model can accurately predict and generate form fields based on user input. The model is trained to understand the relationships between different fields in QHSE forms, as well as the context of each input. This enables the assistant to dynamically generate and reformulate fields, making form completion more intuitive and efficient. In this section, we will discuss the data used for training, the model architecture, and the strategies employed to optimize its performance.

4.4.1 Data Collection

Data collection for the AI Assistant primarily relies on datasets sourced from Betterfly's existing data, specifically from QHSE forms and related documentation. These datasets include real-world examples of form fields, user inputs, and their interdependencies, which are essential for training the model to understand the context and relationships between different form fields.

However, a significant limitation in the data collection process is the scarcity of domain-specific data. While Betterfly's datasets provide a valuable foundation, the lack of extensive, diverse examples within the Digitalisation QHSE domain poses a challenge for training a robust model.

For example in order to generate the "Analyse des causes," the dataset requires the "Description de Non-conformité" as input. This description provides the context necessary for the model to understand the source of the problem and produce the relevant root cause analysis using the 5M methodology.

Similarly, to generate the "Action corrective," the model uses the "Analyse des causes" as the input. The causes identified guide the model in proposing appropriate corrective actions to address the non-conformity.

The structured relationship between these fields is critical for training the model to produce contextually accurate outputs. Since the dataset follows this natural progression from identifying the problem, analyzing the root causes, and proposing corrective measures, it serves as an ideal foundation for training an AI system to automate the process of generating root cause analyses and corrective actions.

Description de Non-conformité	**Analyse des causes (Méthode des 5M)**	**Action corrective proposée**
Absence de documentation adéquate pour le suivi des inspections internes : Lors d'un audit interne, il est constaté que les rapports d'inspection ne sont pas systématiquement documentés, ce qui rend difficile la traçabilité des actions correctives.	**Matériel :** Absence de logiciels adaptés pour la gestion documentaire. **Méthode :** Absence de procédure claire pour la documentation des inspections. **Main-d'œuvre :** Manque de formation des employés sur l'importance de la documentation. **Milieu :** Environnement de travail sous pression, peu de temps pour la documentation. **Management :** Manque de surveillance de la part des responsables sur la documentation des inspections.	Mettre en place un logiciel de gestion documentaire et former les employés à son utilisation pour garantir que les inspections soient systématiquement documentées.

Table 4.1: Example of the available Dataset

4.4.2 Data processing

The initial data used for training was sourced from the Betterfly database, containing raw data from various QHSE forms. This data was then transformed into a structured format suitable for model training. Specifically, we focused on extracting the input and output fields from the columns that were most relevant to form completion tasks.

In this process, relationships between the columns were identified and mapped to create a coherent dataset where columns are connected, similar to how one would construct a graph. These connections reflect the dependencies between different form fields, allowing the model to learn the underlying structure of the forms and how changes in one field can influence the content of another.

Once this data was organized, further preprocessing steps included cleaning, normalizing, and formatting the inputs to ensure consistency across the dataset. This structured and processed data serves as the foundation for training the AI Assistant to understand form field generation and inter-field relationships.

4.4.3 Choice of Model

For this project, we are focused on exploring both paid, high-accuracy models and open-source alternatives to strike a balance between performance and cost-effectiveness.

One of the primary options we are considering is GPT, a paid, high-accuracy model known for its exceptional ability to generate coherent and contextually accurate text. GPT's robust performance makes it ideal for complex tasks like form field generation,

where precision and contextual understanding are crucial. This model offers superior accuracy and adaptability, making it a strong candidate for handling the nuances of QHSE forms.

On the other hand, we are also exploring open-source alternatives like LLaMA3, a powerful and flexible model that, while not as polished as GPT, offers significant potential for customization and scalability. LLaMA3's open-source nature provides the opportunity to fine-tune the model for our specific domain, offering a cost-effective solution without compromising too much on accuracy.

By evaluating both GPT and LLaMA3, we aim to determine the best fit for our AI Assistant, considering factors such as accuracy, cost, scalability, and adaptability to the Digitalisation QSE domain. This dual approach ensures that we can leverage the strengths of each model type depending on the specific needs and constraints of the project.

4.5 Prompt Engineering

One of the key components of our AI Assistant is its ability to generate contextually accurate form fields based on the relationships between other fields. In the Digitalisation QHSE domain, this process is enhanced by leveraging ISO standards, such as ISO 9001, ISO 14001, and ISO 45001, to inform and guide the generation of form fields. These standards provide structured frameworks that define specific requirements, processes, and data flows.

4.5.1 Extracting Guidelines from ISO Standards

ISO standards provide detailed guidelines on how different fields and processes relate to each other within the Quality, Health, Safety, and Environmental management systems. These guidelines are critical for ensuring compliance and consistency in form completion. For example, ISO 9001 specifies the relationship between quality management processes and documentation requirements, while ISO 45001 focuses on occupational health and safety data.

To enable the AI Assistant to generate form fields based on these relationships, we extracted key principles and rules from the ISO standards and incorporated them into the prompt engineering process. This involved identifying dependencies between fields as outlined by the ISO guidelines, such as how risk assessments (ISO 45001) may impact safety measures, or how environmental performance data (ISO 14001) is tied to specific compliance indicators.

4.5.2 Guiding the Model with ISO-Based Prompts

Using the extracted guidelines, we crafted specific prompts that direct the model to generate form fields in alignment with ISO standards. These prompts serve as structured inputs, ensuring that the model adheres to the logical relationships and regulatory requirements

defined in the system. For example, when a user inputs data related to quality control, the AI Assistant is prompted to suggest fields related to corrective actions and preventive measures, as required by ISO 9001.

This prompt engineering process is crucial for maintaining the integrity and relevance of the generated fields, ensuring that they not only reflect user inputs but also comply with international standards. By embedding ISO guidelines into the prompt structure, we improve the accuracy of field generation and ensure that the resulting forms meet the regulatory and procedural requirements of the QHSE domain.

4.5.3 Improving Data Generation Through ISO Compliance

The use of ISO guidelines as a foundation for prompt engineering not only improves the AI Assistant's ability to generate contextually appropriate fields but also enhances data quality and consistency. By following established standards, the model ensures that the generated data fields align with best practices, which is essential for auditability, compliance, and operational efficiency.

This approach also simplifies the form completion process for users, as they are guided by ISO-compliant prompts that automatically generate relevant fields based on their inputs. This minimizes the risk of errors and omissions, ensuring that all necessary information is captured according to regulatory standards.

4.5.4 Limitation

While this method has shown promising results, there are several limitations that need to be addressed.

One major limitation is the need for a separate model, each with its own fine-tuned prompts and domain-specific data, for various aspects of QHSE. For instance, when generating the "Analyse des causes," the ISO-extracted guidline 5M method (Matériel, Méthode, Main-d'œuvre, Milieu, Management) is a highly effective tool in root cause analysis. However, this approach is not universally applicable, especially when moving to the next step of generating "Action corrective."

The 5M method focuses primarily on diagnosing problems rather than proposing solutions. As a result, it lacks the depth needed to guide the generation of actionable and effective corrective measures. This gap highlights the need for different models or extensions of the existing model tailored to various aspects of the QHSE process. For example, generating corrective actions may require additional ISO guidelines or methodologies, such as PDCA (Plan-Do-Check-Act), which is more suited for formulating corrective strategies.

This limitation underscores the complexity of automating QHSE processes and the necessity of designing multi-layered models that can understand and address domain-specific needs at each step of the process. Without these specialized models, the results may lack precision and fail to meet the compliance requirements of diverse industries within the QHSE framework.

While the extraction of guidelines is currently done manually, there are potential ways to automate this process, which could improve efficiency and scalability. One approach explored was the use of Retrieval-Augmented Generation (RAG) systems to extract guidelines directly from relevant ISO standards and documentation. However, RAG systems proved to be ineffective in this context because they primarily focus on retrieving text similar to the input, rather than understanding and extracting complex guidelines that are not directly correlated to the input data.

RAG systems are proficient at finding text passages that match the input query, but they struggle to grasp the deeper, more structured relationships between different fields and guidelines. Guidelines in the QHSE context often depend on intricate frameworks like the 5M method or other compliance-specific methodologies, which require a deeper understanding of the relationships between non-conformities and corrective actions.

4.6 Finetuning

Given the relatively small dataset available for training, while the initial results from both GPT and LLaMA3 models were promising, there remains significant room for improvement. To enhance the model's performance, fine-tuning is necessary to tailor it more closely to the specific needs of the Digitalisation QHSE domain.

Fine-tuning involves adjusting the pre-trained model to better understand and generate form fields based on the unique structure and requirements of QHSE forms. By exposing the model to domain-specific data and gradually refining it, we can improve the accuracy and contextual relevance of its predictions.

During this process, we focus on optimizing key aspects such as the relationships between fields, handling complex interdependencies, and ensuring that the generated fields are both accurate and logically connected. This fine-tuning process allows the model to learn from the limited dataset more effectively, compensating for the small data size by increasing its domain-specific proficiency.

The goal of fine-tuning is to strike a balance between generalization and specialization, ensuring that the model performs well not only on the training data but also on new, unseen forms, ultimately leading to a more reliable and user-friendly AI Assistant.

4.7 Few-Shot Learning

To address the challenges posed by the small dataset, we are exploring the use of few-shot learning as a solution. Few-shot learning allows a model to generalize and make accurate predictions even with a limited amount of training data by leveraging pre-existing knowledge from large, pre-trained models.

In this approach, instead of requiring extensive domain-specific data, the model is provided with a few examples (known as "shots") of how form fields are structured and related to one another. By incorporating these examples into the learning process, the

model can infer patterns and generate accurate form field predictions without the need for a large dataset. This is particularly useful in domains like Digitalisation QSE, where comprehensive datasets are difficult to obtain.

Few-shot learning reduces the dependency on massive datasets by utilizing the inherent capabilities of advanced language models like GPT and LLaMA3, which are already trained on a vast range of text data. The few examples provided help the model adapt to the specific task of generating form fields based on limited input. This technique significantly enhances the performance of the AI Assistant, making it more effective despite the data constraints.

By leveraging few-shot learning, we can mitigate the impact of the small dataset and still achieve high accuracy and relevance in the generated form fields, ensuring the AI Assistant remains a valuable tool for form completion.

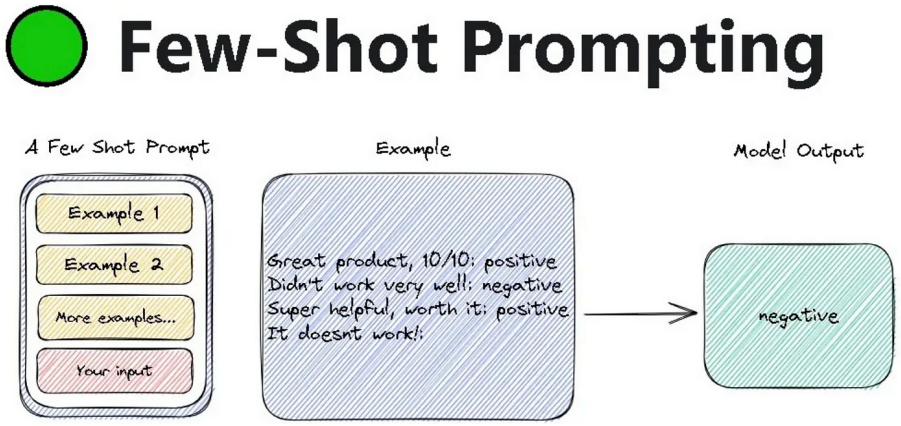

Figure 4.1: Fewshot Learning Example

4.8 Evaluation

In this section, we evaluate the performance of the models using both fine-tuning and few-shot learning approaches for LLaMA3 and GPT-3.5. The evaluation focuses on the accuracy, efficiency of the AI Assistant in the Digitalisation QSE context. The following table summarizes the results of the four model configurations: LLaMA3 Fine-Tuned, LLaMA3 Few-Shot Learning, GPT-3.5 Fine-Tuned, and GPT-3.5 Few-Shot Learning.

Model	**Accuracy**	**Inference Time**
LLaMA3 Fine-Tuned	Low	Medium
LLaMA3 Few-Shot Learning	High	Slow
GPT-3.5 Fine-Tuned	Low	Fast
GPT-3.5 Few-Shot Learning	Best	Slowest

Table 4.2: Model Evaluation: Accuracy and Inference Time

4.8.1 Performance

From the results, GPT-3.5 Few-Shot Learning emerges as the best-performing model in terms of accuracy, despite having a slow inference time. making it ideal for tasks where precision is critical, though less suited for real-time applications requiring fast responses.

4.8.2 Slow Inference Time in Few-Shot Learning

The slow inference times observed in few-shot learning models stem from the additional computational overhead required to process the small set of examples (or "shots") provided with each request. Unlike fine-tuned models, few-shot learning requires the model to continually reference these examples to generate accurate predictions, which increases the complexity of the computation.

While few-shot learning offers significant flexibility, especially with small datasets, its slower inference time can be a disadvantage for applications requiring rapid responses. This is particularly relevant for real-time form generation tasks, where the user expects immediate feedback. Despite this, few-shot learning's strength in handling limited data makes it a powerful option for our project.

4.9 Integrating

To facilitate the use of the AI Assistant in generating form fields, we have hosted the model on a cloud-based server and created an API. This API allows seamless communication between the front end and the model, enabling the automatic generation of fields based on user input and ISO guidelines.

The API is designed to handle requests from the front-end application, which sends relevant information such as:

- **Form context**: The specific type of form being used (e.g., "Non-conformité", "Exigence Légale").
- **Guidelines**: Which ISO guidelines or methodologies (e.g., 5M method, ISO 9001, ISO 45001) should be applied.
- **Field requirements**: What fields need to be generated or populated.
- **Used fields**: What fields are filled, and their values.

By hosting the model on a server and setting up a robust API, we ensure that the front-end can efficiently request and retrieve generated form fields without any interruption to the user experience. This integration not only automates a significant portion of the QHSE form completion process but also ensures compliance with ISO standards by embedding these guidelines directly into the generation pipeline.

4.10 The model in action in Betterfly

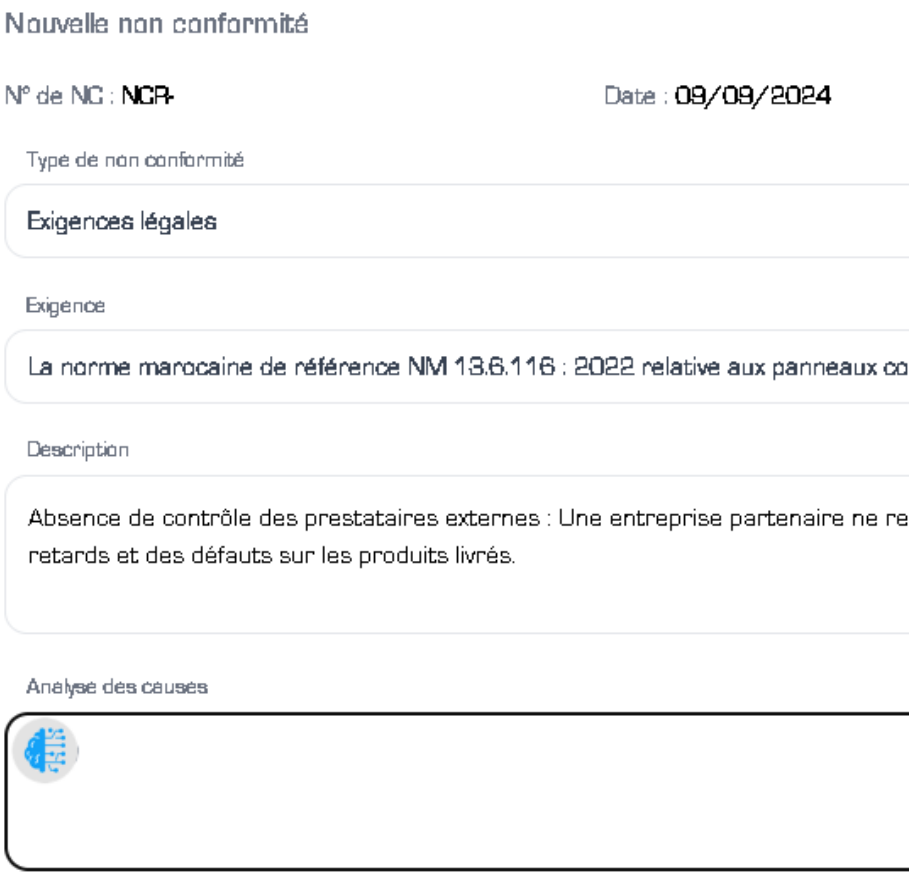

Figure 4.2: Assistant icon

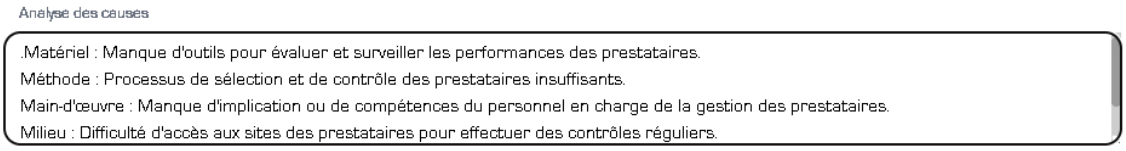

Figure 4.3: Data generated

4.11 Conclusion

In this project, we developed an AI Assistant aimed at automating the generation of form fields, specifically within the Digitalisation QHSE domain. using domain-specific guidelines and ISO standards, the AI system enhances the efficiency and accuracy of form completion, while reducing the time and effort required for professionals to complete their documentation.

Despite some limitations, such as the need for domain-specific models and the challenges posed by small datasets, we have explored solutions like few-shot learning and fine-tuning models. The integration of ISO guidelines into prompt engineering further ensures that the AI-generated outputs align with established standards and compliance requirements.

Through the development of a cloud-hosted model and API, we successfully integrated the AI Assistant into our front-end interface. This project demonstrates the potential of AI in enhancing productivity and compliance in the QHSE sector by automating routine yet critical tasks.

Looking ahead, future work will focus on improving model accuracy, refining domain-specific guidelines, and exploring more advanced techniques for automating the extraction and application of ISO standards in the generation process. With these improvements, the AI Assistant can become an even more valuable tool in the digital transformation of QSE practices.

Conclusion and Perspectives

This thesis presents two projects focused on using AI to enhance automation and efficiency in two distinct domains: chart configuration generation and form field generation in the QHSE domain. In the first project, we developed a system capable of generating chart configurations based on user inputs, leveraging a model fine-tuned to understand and replicate Chart.js configurations. This significantly improved the user experience, enabling them to create complex charts more efficiently without requiring deep technical knowledge.

The second project focused on an AI Assistant for generating form fields using predefined guidelines and ISO standards. By automating this process, we addressed the challenges of form completion in the QHSE domain, making it more streamlined and compliant with regulatory standards.

Both projects demonstrated the utility of AI in automating tasks that traditionally required manual input and deep domain expertise, proving that AI models can greatly enhance user experience and productivity.

Limitations

Despite the success of both projects, there are notable limitations:

In the chart configuration project, while the system works well for predefined chart types, its flexibility in handling completely new or highly customized configurations is still somewhat limited without extensive fine-tuning. Although the dataset used for training was comprehensive, the model is still dependent on understanding specific patterns, and more advanced use cases, such as creating highly complex or novel chart types, could benefit from further development.

For the AI Assistant, one of the main challenges is the need for domain-specific models. The current system requires separate models for each guideline. The reliance on predefined guidelines also limits its ability to adapt to new regulatory frameworks without additional model training or updates. Furthermore, the small dataset posed challenges in achieving high accuracy, especially for generating highly specialized fields.

Another limitation of both projects is the reliance on manual extraction of guidelines and prompts, which, while effective, limits scalability. Automating this process remains a challenge, particularly in the QSE domain, where the guidelines are complex and diverse.

Future Work

To address the limitations identified, future work will focus on several key areas:

1. **Dataset Expansion and Model Improvement:** In both projects, expanding the dataset and improving the fine-tuning of the models will be crucial to enhancing accuracy and flexibility. This will involve collecting more domain-specific data and utilizing techniques like few-shot learning to optimize performance, even in cases of limited data.

2. **Utilizing Text2SQL Models for Chart Configuration:** A key improvement in the chart configuration project would be the incorporation of Text2SQL models. These models are highly accurate and can transform natural language descriptions into SQL queries that extract the required data from the database. By applying this technology, the system can generate charts dynamically from SQL data, further improving its ability to handle new and highly customized configurations without the need for extensive manual configuration.

3. **Automation of Guideline Extraction:** For the QSE project, automating the extraction and application of guidelines from ISO standards and other regulatory frameworks is a priority. Future developments could include the use of knowledge graphs, ontology-based systems, or more advanced machine learning techniques to extract and apply guidelines dynamically.

4. **Unified Models for QHSE:** A key area for improvement is the development of unified models capable of handling multiple guidelines and methodologies, reducing the need for separate models for each task. This will increase the scalability and adaptability of the AI Assistant, making it more applicable across various contexts.

5. **Real-Time Performance Optimization:** As both projects deal with real-time user interaction, optimizing inference times for few-shot learning and reducing model latency will be crucial.

By addressing these areas, future iterations of these projects will be better equipped to handle more complex use cases, provide greater flexibility, and further enhance the user experience through automation.

References

Category	Resource
Website	EQUALITY
Website	Betterfly
Website	Chart.js: Simple, clean and engaging HTML5 charts for developers
Research Paper	Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A. (2020). Language Models are Few-Shot Learners. Advances in Neural Information Processing Systems.
Research Paper	Examining Autocompletion as a Basic Concept for Interaction with Generative AI
Research Paper	Automating Chart Generation: A Generative AI Approach to Data Visualization
Research Paper	Generative AI for Visualization: State of the Art and Future Directions
Research Paper	Source code auto-completion using various deep learning models under limited computing resources
Research Paper	Auto-completion Using Deep Learning: Comparing and Discussing Current Language-Model-Related Approaches
Research Paper	Text2Chart: A Multi-Staged Chart Generator from Natural Language Text
Research Paper	Evaluating the Text-to-SQL Capabilities of Large Language Models
Research Paper	Enhancing Natural Language Query to SQL Query Generation Through Classification-Based Table Selection
Research Paper	The Application of Enterprise QHSE Management Performance Evaluation System Based on Maturity Model
Research Paper	Overview of quality health safety and environmental management systems implementation in Zimbabwe
Research Paper	Machine Learning for Synthetic Data Generation: A Review
Research Paper	A Systematic Survey of Prompt Engineering in Large Language Models: Techniques and Applications
Documentation	Microsoft Azure Machine Learning Documentation
Documentation	Hugging Face Transformers Documentation
ISO Standard	ISO 9001:2015 Quality Management Systems – Requirements. International Organization for Standardization.

This thesis, conducted as part of an end-of-studies internship for a Master's degree in Artificial Intelligence and Virtual Reality at the Faculty of Sciences - Ibn Tofail University presents two projects aimed at utilizing AI to enhance automation and efficiency in two distinct areas: chart configuration generation and form field generation within the QHSE domain. The first project developed a system that generates chart configurations based on user inputs, leveraging a model fine-tuned to understand and replicate Chart.js configurations. This significantly improves the user experience by enabling users to create complex charts more efficiently and dynamically, without requiring extensive technical knowledge. The second project focused on creating an AI Assistant for generating form fields based on predefined guidelines and ISO standards in the QHSE domain. This automation addresses challenges related to form completion, making the process more streamlined and compliant with regulatory standards. Both projects demonstrated how AI can automate tasks that traditionally required manual input and domain-specific expertise, proving that AI can greatly improve user experience and productivity.

Ce mémoire, réalisé dans le cadre d'un stage de fin d'études pour un Master en Intelligence Artificielle et Réalité Virtuelle à la Faculté des Sciences - Université Ibn Tofail présente deux projets visant à utiliser l'IA pour améliorer l'automatisation et l'efficacité dans deux domaines distincts : la génération de configurations de graphiques et la génération de champs de formulaires dans le domaine QHSE. Le premier projet a développé un système qui génère des configurations de graphiques en fonction des entrées utilisateur, en s'appuyant sur un modèle affiné pour comprendre et reproduire les configurations Chart.js. Cela améliore considérablement l'expérience utilisateur en permettant aux utilisateurs de créer des graphiques complexes de manière plus efficace et dynamique, sans nécessiter de connaissances techniques approfondies. Le deuxième projet s'est concentré sur la création d'un assistant IA pour générer des champs de formulaire en fonction de directives prédéfinies et de normes ISO dans le domaine QHSE. Cette automatisation répond aux défis liés à la saisie de formulaires, rendant le processus plus rationalisé et conforme aux normes réglementaires. Les deux projets ont démontré comment l'IA peut automatiser des tâches qui nécessitaient traditionnellement une saisie manuelle et une expertise spécifique au domaine, prouvant que l'IA peut grandement améliorer l'expérience utilisateur et la productivité.