UNIVERSIDADE
BEIRA INTERIOR

# Automated Career Stagnation Detection System

**Filipe Lourenço Catarino**

Dissertação para obtenção do Grau de Mestre em
**Engenharia Informática**
(2º ciclo de estudos)

Orientador: Prof. Doutor Paulo André Pais Fazendeiro

**Junho de 2024**

# Declaração de Integridade

Eu, Filipe Lourenço Catarino, que abaixo assino, estudante com o número de inscrição M12494 do 2º Ciclo de Engenharia Informática da Faculdade de Engenharias, declaro ter desenvolvido o presente trabalho e elaborado o presente texto em total consonância com o Código de Integridades da Universidade da Beira Interior.

Mais concretamente afirmo não ter incorrido em qualquer das variedades de Fraude Académica, e que aqui declaro conhecer, que em particular atendi à exigida referenciação de frases, extratos, imagens e outras formas de trabalho intelectual, e assumindo assim na íntegra as responsabilidades da autoria.

Universidade da Beira Interior, Covilhã 11/06/2024

Assinado por: **Filipe Lourenço Catarino**
Num. de Identificação: 30735363
Data: 2024.06.11 14:57:51+01'00'

# Acknowledgements

With the completion of this internship project report and the conclusion of my second cycle of studies in Computer Science and Engineering, I am deeply grateful to all those who have supported me throughout this journey.

Firstly, I want to express my heartfelt appreciation to my entire family, particularly my parents and siblings, for their unwavering support and belief in me. Their constant encouragement during challenging times and their invaluable advice have been the foundation of my success and accomplishments.

I extend my profound thanks to my supervisor, Professor Doctor Paulo André Pais Fazendeiro, for his exceptional guidance and mentorship. His patience, trust, and unwavering support throughout the realization of this project have been invaluable.

I am genuinely thankful to Softinsa, the company where the internship took place, and to all the team members who welcomed me there. I truly value the opportunity they provided, their acknowledgment of my ideas, and their role in making me feel embraced in a completely new environment.

Additionally, I appreciate the Instituto de Telecomunicações (Covilhã Delegation) for welcoming me during the internship preparatory phase.

I am also deeply grateful to my colleagues and friends for their continuous support and encouragement. Our discussions, shared knowledge, and mutual motivation have significantly contributed to both my academic and personal growth.

In conclusion, I extend my heartfelt gratitude to everyone who directly or indirectly contributed to the realization of this work. Their support and dedicated efforts have made this achievement possible. As we look ahead, I am confident that many more accomplishments lie ahead. All that remains is to express, from the depths of my heart, a profound thank you.

# Resumo

O presente documento descreve todo o trabalho realizado durante um estágio curricular, com a duração de 107 dias, efetuado na empresa Softinsa. Este projeto tem um papel fundamental no cumprimento dos requisitos para a conclusão do mestrado em Engenharia Informática na Universidade da Beira Interior (UBI).
O objetivo principal é criar um sistema capaz de detetar automaticamente o estado de estagnação numa carreira. Trata-se do desenvolvimento de um modelo preditivo capaz de obter e analisar informação para identificar, de forma autónoma, sinais de estagnação nos colaboradores.
No contexto empresarial, este sistema visa otimizar os processos de gestão de recursos humanos, permitindo a identificação precoce de possíveis falhas e a implementação de medidas corretivas. Adicionalmente, tem como propósito promover o aumento do desempenho e bem-estar dos trabalhadores, contribuindo assim para um ambiente de trabalho mais produtivo e satisfatório.
A arquitetura do projeto desenvolvido é composta por três componentes principais, sendo estas a base de dados, o servidor e a interface do utilizador.
Inicialmente foi efetuada uma planificação cuidada, incluindo uma pesquisa extensiva em artigos científicos para identificar os principais problemas e obter informações sobre possíveis soluções e abordagens existentes. Em seguida, foi estabelecida uma calendarização detalhada para gerir e acompanhar eficazmente a progressão das tarefas durante o estágio.
Após a fase de planeamento, o projeto avançou para a sua implementação. Nesta fase, todos os passos efetuados para a construção adequada da aplicação foram delineados e fundamentados de modo a cumprir os requisitos propostos pela empresa. Por fim, o resultado foi exibido, efetuando testes para assegurar a integridade e eficácia do projeto.
No final do documento, são apresentadas as principais conclusões derivadas deste estágio curricular. Isto inclui uma exposição da experiência pessoal, opiniões gerais, desafios enfrentados e uma análise dos objetivos alcançados.

# Palavras-chave

Desenvolvimento de *Software*, Estágio Curricular, Modelo Preditivo, *Microsoft Structured Query Language (MSSQL)*, *Node.js*, *React*.

# Resumo alargado

O presente documento descreve todo o trabalho realizado durante um estágio curricular, com a duração de 107 dias, efetuado na empresa Softinsa, sendo esta especializada em serviços de consultoria, gestão e desenvolvimento de aplicações. Esta organização emprega mais de 700 profissionais localizados em vários pontos do país e a sua principal missão consiste em contribuir para um mundo mais sustentável através da inovação tecnológica, apostando em investimentos persistentes no desenvolvimento regional para ampliar os talentos e competências da população local.

A realização deste projeto tem um papel fundamental no cumprimento dos requisitos para a conclusão do mestrado em Engenharia Informática na Universidade da Beira Interior e foi efetuado no âmbito da cadeira de Dissertação ou Estágio em Engenharia Informática pertencente ao segundo semestre do ano letivo de 2023/2024.

O objetivo principal é criar um sistema capaz de detetar automaticamente o estado de estagnação numa carreira. Trata-se do desenvolvimento de um modelo preditivo capaz de obter e analisar informação para identificar, de forma autónoma, sinais de estagnação nos colaboradores.

No contexto empresarial, este sistema visa otimizar os processos de gestão de recursos humanos, permitindo a identificação precoce de possíveis falhas e a implementação de medidas corretivas. Adicionalmente, tem como propósito promover o aumento do desempenho e bem-estar dos trabalhadores, contribuindo assim para um ambiente de trabalho mais produtivo e satisfatório.

A arquitetura do projeto desenvolvido é composta por três componentes principais, sendo estas a base de dados, o servidor e a interface do utilizador. Para a base de dados, utilizou-se um servidor MSSQL com o objetivo de configurar e inserir os dados essenciais da aplicação, assegurando a conformidade do esquema com os requisitos especificados. Para o servidor, optou-se pela aplicação *Node.js*, que permite a execução de consultas e manipulação de dados entre a base de dados e o servidor. Por último, para a aplicação do lado do cliente, utilizou-se o *React*, que proporciona uma interface interativa e responsiva.

Esta estrutura permitiu criar um ambiente consistente e isolado para cada componente, onde o servidor e a base de dados tratam da lógica da aplicação e do processamento de dados respetivamente, enquanto o lado do cliente proporciona a interação com o utilizador.

Para o completo desenvolvimento desde sistema, os passos iniciais envolvem uma planificação cuidada, incluindo uma pesquisa extensiva em artigos científicos para identificar os principais problemas e obter informações sobre soluções existentes. O primeiro artigo apresentado descreve um sistema desenvolvido para avaliar o desempenho dos colaboradores numa agência de segurança pública, cuja arquitetura se assemelhava ao trabalho a ser desenvolvido, proporcionando uma compreensão mais clara da estrutura em geral. Em seguida, discutiu-se um artigo científico apresentado em conferência, que tratava da automação do processo de promoção de colaboradores, detetando automaticamente critérios indicativos de progressão na carreira e assim fornecendo estratégias via sistemas de aprendizagem automática. Os dados apresentados nesse estudo também compartilhavam similaridades com

aqueles descritos no presente projeto. Posteriormente, abordou-se outro artigo científico, também apresentado em conferência, que destacava a importância da utilização de boas visualizações gráficas para uma compreensão e análise mais profunda da informação obtida pelo utilizador.
Neste documento também foi estabelecida uma calendarização pormenorizada, servindo como guia para gerir e acompanhar eficazmente a progressão das tarefas, cada uma descrita com precisão para garantir uma abordagem sistemática e organizada ao longo do projeto.
Após a fase de planeamento, o projeto avançou para a sua implementação. Nesta fase, todos os passos efetuados para a construção adequada da aplicação foram delineados e fundamentados de modo a cumprir os requisitos propostos pela empresa. Por fim, o resultado foi exibido, efetuando testes para assegurar a integridade e eficácia do projeto.
No final do documento, são apresentadas as principais conclusões derivadas deste estágio curricular incluindo uma exposição da minha experiência pessoal, opiniões gerais, desafios enfrentados e uma análise dos objetivos alcançados.

# Abstract

This document describes all the work carried out during a curricular internship, lasting 107 days, at Softinsa, a distinguished company specializing in consulting, management, and application development services. It plays a crucial role in meeting the requirements for the completion of the master's degree in Computer Science and Engineering at the University of *Beira Interior* (UBI).

The primary objective of this project is to establish a system that is capable to automatically detect career stagnation. This involves the development of a predictive model capable of retrieving and analyzing information to autonomously identify signs of career stagnation among employees.

In the business context, this system aims to optimize human resources management processes, enabling early identification of possible failures and the implementation of corrective measures. In addition, its purpose is to promote increased employee performance and well-being, thus contributing to a more productive and satisfying work environment.

The architecture of the project is made up of three main components, being this the database, the server, and the user interface.

Initially, careful planning was carried out, including extensive research into scientific articles to identify the main problems and obtain information on possible solutions and existing approaches. A detailed timetable was then established to effectively manage and monitor the progress of tasks during the internship.

After the planning phase, the project moved on to implementation. At this stage, all the steps taken to properly build the application were outlined and substantiated in order to fulfill the requirements proposed by the company. Finally, the result was displayed, carrying out tests to ensure the integrity and effectiveness of the project.

At the end of the document, the main conclusions derived from this curricular internship are presented. This includes an account of my personal experience, general opinions, challenges faced, and an analysis of the objectives achieved.

# Keywords

Curricular Internship, MSSQL, *Node.js*, Predictive model, *React*, Software Development.

# Contents

# List of Figures

# List of Tables

# Acronyms

**ACSDS** Automated Career Stagnation Detection System

**ANN** Artificial Neural Network

**API** Application Programming Interface

**CORS** Cross-Origin Resource Sharing

**CSS** Cascading Style Sheets

**DBMS** Database Management System

**ERD** Entity-Relationship Diagram

**FR** Functional Requirements

**GUI** Graphical User Interface

**HR** Human Resources

**HTML** HyperText Markup Language

**HTTP** Hypertext Transfer Protocol

**IBM** International Business Machines Corporation

**ID** Identification

**IoT** Internet of Things

**IPP** Polytechnic Institute of *Portalegre*

**IPT** Polytechnic Institute of *Tomar*

**JSON** JavaScript Object Notation

**KPIs** Key Performance Indicators

**MSSQL** Microsoft Structured Query Language

**NFR** Non-Functional Requirements

**RF** Random Forest

**ROS** Random Oversampling

**SA** system administrator

**SMOTE** Synthetic Minority Over-sampling Technique

**SQL** Structured Query Language

**SSMS** SQL Server Management Studio

**SVM** Support Vector Machine

**TDS** Tabular Data Stream

**UBI** University of *Beira Interior*

**UI** User Interface

**URL** Uniform Resource Locator

**UTAD** University of *Trás-os-Montes and Alto Douro*

**VM** Virtual Machine

**VPN** Virtual Private Network

# Chapter 1

# Introduction

This report was written in the context of the curricular unit "*Dissertação ou Estágio em Engenharia Informática*", with the purpose of concluding the Master's Degree in Computer Science and Engineering at University of *Beira Interior* (UBI) [1].
The entirety of this project was carried out as part of an internship at the company "*Softinsa*" [2], details of which will be thoroughly discussed in this document.
The current chapter is divided into four sections, beginning with the identification and description of the problem, which outlines the specific challenges that led to the realization of this work. The subsequent section explores the motivation behind its creation, mentioning the reasons and factors that highlight its importance and relevance in this context. Following that, the next section outlines the specific objectives to be achieved during this internship, delivering a clear understanding of the intended goals. Finally, this chapter concludes with a comprehensive overview of the document's organization, providing readers with a pathway to navigate through the subsequent chapters, each one focused on exploring and detailing various aspects of the subject matter.

## 1.1 Identification of the Problem

We begin our careers with some sense of who we are, where we hope to go, and who we want to become. Along the way, most of us inevitably find ourselves coping with the feeling we're not headed in the right direction. Sometimes, we might even feel like we're going nowhere [3]. These existential questions are not arbitrary, they are often an indicator that people believe their lives may have come to a halt in terms of progress or advancement. Stagnation is the state of not moving forward and for many employees, particularly high performers, it's an uncomfortable place to be.
**Career stagnation** occurs when individuals experience a lack of growth or advancement in their professional trajectories, often leading to a diminished sense of job satisfaction and engagement. This not only affects the individual's well-being but also reverberates within the company, influencing the overall organizational dynamics. The causes are diverse, and it becomes crucial to identify, recognize and address these indicators.

## 1.2 Motivation

The inspiration propelling the execution of this project is driven by the opportunity to make a meaningful impact on the professional landscape by addressing and finding solutions to this problem. The creation of a tool with the purpose of helping companies identify and resolve carrier stagnation among employees is a strategic initiative aligned with the goal of increasing workplace satisfaction, and overall well-being. This initiative aims to achieve a positive change, with the objective to improve the overall employee experience and contribute to the healthy development of both the workers and the organizations.
By completing this project within the scope of an internship, it is also important to mention that it serves as an opportunity to gain new experiences and further develop myself, aligning with my dedication to **continuous growth** and **learning**. This work offers a chance to deepen theoretical and practical concepts, while applying them to real-world scenarios, thereby enhancing both personal and professional evolution.

## 1.3 Objectives

The main goal of this internship is to develop a Automated Career Stagnation Detection System (ACSDS) with a focus on providing a proactive solution to identify and address this issue within organizations. The system aims to autonomously analyze information, make accurate predictions, and offer valuable insights for effective management practices. The successful completion of this project involves conducting initial research on pertinent information and industry best practices, preceding the design and development phase of the system, while adhering to the company's work standards and security protocols. Under those circumstances, the primary objectives of this internship can be summarized as follows:

1. Build a predictive model capable of retrieving and analyzing relevant information to autonomously identify signs of career stagnation among employees;
2. Develop both the frontend and backend components of the system, ensuring a seamless and intuitive web interface for Human Resources (HR) professionals and management;
3. Create a robust database to store all pertinent information for analysis. This database will serve as a centralized repository for the data needed for the system;
4. Design the system architecture with a focus on security, scalability, and responsiveness. The system must efficiently handle vast datasets and real-time interactions while ensuring the protection of sensitive information;
5. Demonstrate punctuality and regular attendance, meet project deadlines, apply concepts learned during the internship, and develop an understanding of work ethics;
6. Gain experience collaborating with colleagues within the organization, adhere to the company's software development processes, acquire proficiency in necessary tools, languages, and frameworks, and construct an application in accordance with directives from superiors.

## 1.4 Document Organization

In order to reflect the work that has been done, this subsection provides readers with a clear pathway to navigate through the subsequent chapters. Each chapter is dedicated to the in-depth exploration and detailed examination of various aspects related to this project. In that context, the document is structured as follows:

- The Chapter 1 presents the **Introduction** of the project, including the detailed identification of the underlying problem, a thorough exploration of the motivation behind its selection, explicit objectives, and a comprehensive overview of the entire document's organizational structure;
- The Chapter 2 shows the **Internship Location**, delving into the historical background and foundational aspects of the company. The chapter also provides in-depth information about the institution, including its core values, a comprehensive overview of its services, insights into available careers and opportunities, an exploration of the tools and resources offered, and concludes with a summary of the key findings;
- The Chapter 3 describes the **State Of The Art**, which provides the knowledge of the most important theoretical issues and concepts in the scope of this project and refers some existing works related to the theme;
- The Chapter 4 introduces the **Internship Planning**, that outlines the proposed work and presents a comprehensive overview of all tasks planned, providing detailed descriptions and allocated time for each chore. Additionally, it discusses the tools and technologies chosen for its implementation;
- The Chapter 5 presents the **System Conception**, focusing on the overall design and implementation of the project. It outlines requirements, exhibits the system architecture, and details implementation steps, with a particular emphasis on coding, showcasing the final product, and testing;
- The Chapter 6 presents the **Conclusion**, which defines all the conclusions drawn after carrying out this project, offers some personal opinions, and suggests directions for future work.

# Chapter 2

# Internship location

The upcoming chapter presents a comprehensive exploration of *Softinsa* [2], the company that played an essential role in the realization of this report by giving me the opportunity to pursue this internship. It aims to provide a deep understanding of the organization's background, structure, and operations.
The first section offers an in-depth examination of the historical roots and milestones that have shaped the company over time, delivering a retrospective analysis that contributes to a more detailed understanding of its trajectory. Moving forward, the second section delves into the fundamental principles of *Softinsa* and defines the company's purpose. It also examines the mission statement and its core values.
The third section explores the company's offerings, highlighting a comprehensive range of services that deliver innovative solutions across various domains. Following that, the next section discusses the opportunities and careers that the organization offers. This serves as a gateway for individuals to align their aspirations with *Softinsa's* dynamic work environment and their personal goals.
An additional, important fifth section describes all the tools offered by the company to their employees to facilitate the execution of their projects. Finally, the chapter concludes with a summary of the key findings.

## 2.1 History and Foundations

In the year **1998**, *Softinsa* took a major step forward by opening an office in Lisbon. This initiative was led by *Viewnext* [4], a Spanish company, founded in 1991, under the control of the International Business Machines Corporation (IBM) [5].
After 9 years since its inception, in **2007**, the establishment of *Softinsa* as an independent entity was marked. Initially, it was a branch of *Viewnext*, but it evolved into the instituion as we know it today.
In **2013**, a protocol involving the Municipal Council of *Tomar*, the Polytechnic Institute of *Tomar* (IPT), and IBM was made, laying the foundation for the creation of an Innovation Center. The subsequent year, **2014**, witnessed its official inauguration, boasting around 100 collaborators. Simultaneously, a dedicated center for clients in the banking sector was also established at IPT.
Continuing the expansion, *Softinsa* inaugurated the another Innovation Center in *Viseu* in the year of **2016**, graced by the presence of the President of the Portuguese Republic, *Marcelo Rebelo de Sousa* [6].
In **2019**, a significant protocol was signed with the Municipality of *Fundão*, paving the way for a specialized lab in this city, being the current location where the mentioned internship

in this report is hosted.

In the next few years we witnessed another two crucial agreements, one with the Municipality of *Portalegre* and the Polytechnic Institute of *Portalegre* (IPP) for a specialized lab (**2020**) and another with the Municipality of *Vila Real*, *Regia Douro Park*, and the University of *Trás-os-Montes and Alto Douro* (UTAD) for the establishment of a center for innovation and technology in this region (**2022**).

As of **2023**, *Softinsa* celebrated 25 years since its foundation, marking a significant milestone in its journey of growth, innovation, and contributions to multiple fields. To enhance comprehension of its impact, the following image illustrates all the current locations and regions in which the company currently operates while presenting their address:

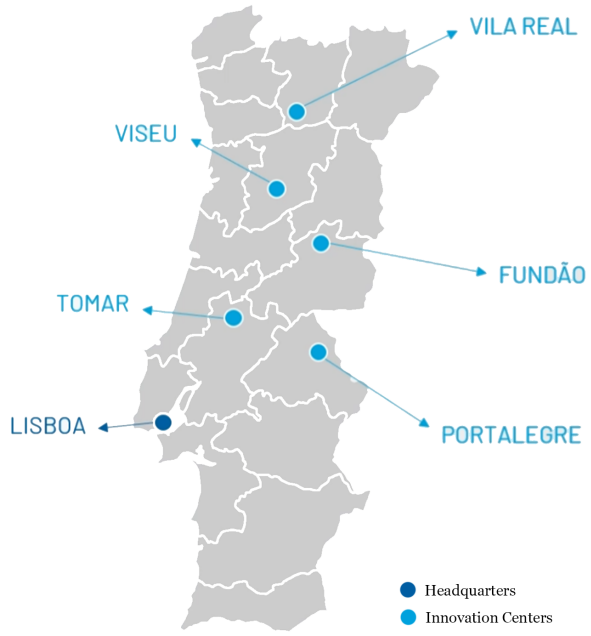



Figure 2.1: Current innovation centers and headquarters of *Softinsa* and their respective address.

***Lisboa**, 1990-138 — Edificio Office Oriente, Rua do Mar da China nº3 — B6, Parque das Nações;*

***Tomar**, 2300-305 — Avenida Doutor Aurélio Ribeiro nº1;*

***Viseu**, 3500-618 — Zona Industrial de Coimbrões 16;*

***Fundão**, 6230-378 — Rua da Cale nº54;*

***Portalegre**, 7300-555 — BioBIP - Bioenergy and Business Incubator of Portalegre, Campus Politécnico, 10;*

***Vila Real** 5000-033 — Regia-Douro Park, Parque de Ciência e Tecnologia.*

## 2.2 About *Softinsa*

Currently, *Softinsa* is a subsidiary of IBM, specializing in consulting, management, and application development services. The company employs over 700 professionals located in various places, as mentioned in the previous section.

The organization's **main mission** involves contributing to a more sustainable world through technological innovation, with a focus on persistent investments in regional development aimed at developing the talents and skills of the local population. It also aspires to maintain its status as a benchmark for innovation and excellence in the market, relying on sustainable solutions and services that contribute to the growth of its people, the success of its clients, and the development of society.

*Softinsa* is also guided by a set of core values that serve as the foundation for achieving its main objectives, which will be highlighted in the list below:

1. **People** — The organization places a high priority on creating a supportive environment for the workers, ensuring their success and facilitating continuous personal growth;
2. **Integrity** — Maintaining a code of ethical conduct is at the core of the organization's work and relationships. There is a dedicated effort to safeguard the integrity of clients while contributing positively to society and various stakeholders;
3. **Quality** — The company's ambition also revolves around elevating the overall quality of work, aspiring to set new benchmarks within the market. This commitment to quality is a cumulative result of collaborative teamwork and individual performance;
4. **Innovation** — The organization aspires to have the capacity to proactively anticipate the evolving requests of its clients. This approach involves presenting forward-thinking and inventive proposals, managing any service or initiative with agility and dedication;
5. **Clients** — The institution is committed to actively addressing and adding substantial value to the needs of clients, with a dedicated focus on delivering an optimal service experience.

## 2.3 Services Overview

In alignment with its core values, the company provides a comprehensive range of services designed to meet the diverse requests of its clients. These services reflect a strategic approach to addressing the dynamic challenges of the modern business landscape. Further elaboration on these services will be provided in the subsequent paragraphs according to the information available on the organization's website [7].

To begin, the digital business districts service aims to promote the digitalization of the economy through the adoption of technologies and the automation of business models. It is a measure focused on the trade and services sectors open to the consumer, while being also a catalyst for economic growth, promoting online commerce and integrating digital supply and sales chains.

The company also facilitates business by offering application management services, enhancing flexibility, speed, and efficiency in developing and supporting software, while utilizing the best methodologies and tools for reduced intervention times and higher service quality rigor.

Another key service worth highlighting is the IBM maximo asset management, an integrated solution for the strategic control of corporate assets and services. It facilitates a sustained decision-making process based on technical information, work and safety protocols, historical data, and statistical analysis of expenses.

Nowadays, data stands as one of the most valuable assets for companies, and *Softinsa*, in collaboration with IBM, is dedicated to developing methodologies, approaches, and technologies accordingly. Addressing these challenges, the business intelligence and analytics service

is designed to facilitate the consolidation, understanding, and extraction of value from data. This service engages in information related activities such as data modeling, data architecture, data integration, data quality, data governance, and reporting.

Another significant data-related asset provided by the company is the cognitive service, with the aim of assisting customers in evolving their systems, optimizing and reducing operating costs, exploring more agile working methods, and discovering innovative ways to expand business opportunities. This service is designed to transform structured and unstructured data, including images, text, audio, and video files, into essential and meaningful information, as well as automating the process.

The human capital solutions service, which has accompanied *Softinsa* since the start of its activity in Portugal, is currently made up of a vast team of functional and technical consultants exclusively dedicated to the HR area and committed to offering its clients a basis for innovation, through application solutions, implementation services and HR consultancy.

The company also provides mobile solutions, a service that develops web applications and efficient corporate mobility resolutions according to the client business requirements. These solutions make it possible to extend organizational processes to mobile devices in a simple and secure way, anywhere and at any time.

Lastly, the smart cities and Internet of Things (IoT) service aims to achieve greater quality and efficiency in urban management, supporting its citizens by providing them access to relevant information for their interaction with the surroundings. The incorporation of the right tools and big data technologies has led to the development of a centralized, intelligent, and automated platform for managing and controlling critical city services.

According to all the information provided in this section, a consensus can exist that *Softinsa* offers a diverse service portfolio, including various sectors and comprehensive solutions to address the multifaceted needs of its clients.

## 2.4 Careers and Opportunities

In addition to serving clients, *Softinsa* places great emphasis on offering diverse career paths for its employees. The company's approach revolves around crafting individualized career plans based on skills and values, allowing each employee to acquire knowledge at their preferred pace.

Within *Softinsa's* comprehensive framework for professional development, the extensive list of career plans spans across a spectrum of diverse areas, including roles such as consulting, digital applications' development, cloud and managed services, sales, project management, and corporate functions. This comprehensive and multifaceted program is designed to create an environment that not only supports, but actively encourages professional growth. It is adapted to address the distinct expectations, motivations, and experiences of each employee, providing a personalized path for their career journey within the organization.

## 2.5 Environment and Tools Offered

The company extends a valuable benefit to its employees by providing complimentary access to all innovation centers. These well-equipped spaces contain multiple rooms designed to provide for various needs, including recreational areas for leisure. The organization also ensures free access to important work equipments, facilitating an environment where each employee has access to tools that enhance their productivity and contribute to a conducive working atmosphere.
By actively participating in this internship, I was fortunate to be granted the opportunity to engage in remote work, enabling me to contribute effectively from the comfort of my own space. Additionally, *Softinsa* has grated to me exclusive access to the company's Virtual Private Network (VPN), facilitating access entry into private cloud files for collaborations and information retrieval. Moreover, as part of this program, I have been given a dedicated personal laptop to work on, specifically a Lenovo ThinkPad L460 [8], enabling me to carry out my tasks with efficiency and convenience.

## 2.6 Concluding Remarks

In summary, this chapter provides a comprehensive exploration of *Softinsa*, delving into its historical roots, organizational principles, service offerings, career paths, and the tools provided to employees. The company's commitment to innovation, sustainability, and fostering individual growth emerges as prominent themes.
This chapter starts by presenting the company's journey from its establishment in 1998 to its current status as a IBM subsidiary. It also outlines its mission, core values, and diverse services. A consensus can exist that *Softinsa* places a strong emphasis on employee development, offering individualized career plans while providing a supportive work environment, including access to innovation centers, remote work opportunities, and essential tools.

# Chapter 3

# State Of The Art

This chapter serves as a crucial foundation for the comprehensive understanding and development of the project, providing an in-depth exploration of the theoretical components and concepts essential to its framework.
It begins by presenting an elaborate definition of career stagnation, including some causes and indicators that might identify this problem, laying a foundation for a thorough comprehension of the phenomenon. Following this, the chapter delves into a meticulous analysis of existing research, identifying connections between both projects. This exploration not only enhances comprehension of the current landscape but also reveals gaps in knowledge, offering insights and potential ideas for further exploration.
In conclusion, the last section succinctly summarizes the content covered in this chapter, offering a cohesive overview of the theoretical base and research landscape relevant to the project.

## 3.1 Career Stagnation

### 3.1.1 Definition

Career stagnation is «a complex and widespread phenomenon marked by a profound lack of engagement with one's work or career trajectory [9]. This stagnation becomes apparent when individuals perceive a lack of positive changes in their professional journey or express concerns about the potential erosion of their valuable professional skills. It manifests as a sense of hitting a metaphorical invisible ceiling in the career, accompanied by a noticeable absence of enthusiasm, engagement, and motivation.
This feeling of being stuck often arises from a perception that the current job fails to provide the necessary challenges for personal and professional growth, or from the belief that one has reached a dead end in terms of advancing to the next step in their career. Importantly, career stagnation is not confined to a specific stage, it can also afflict individuals at various points in their professional journey, including those in the early stages.

### 3.1.2 Causes and Signs

Stagnation can manifest through numerous diverse factors, and, regrettably, **it often goes unnoticed**. This subtle and pervasive phenomenon becomes particularly apparent when individuals fail to recognize them. In an effort to enhance awareness, the following list is presented to elucidate and identify all potential cause [10]:

1. **No Rise in Salary** — Despite making significant contributions and putting in dedicated efforts, the lack of salary adjustment and unchanged titles can lead to feelings of

stagnation;

2. **Lacking Skills** — Recognizing the need for additional skills to achieve improved performance is crucial. Skill gaps can impede effectiveness, and identifying them is essential for addressing certain limitations;

3. **No Opportunities** — Over an extended period, there may be a lack of new learning experiences within the organization. The current job can fail to provide avenues for advancement, contributing to a pervasive feeling of stagnation;

4. **Workplace Decline** — If the current organization is experiencing losses, it can significantly impact the worker career trajectory, indicating a need to explore new opportunities outside the declining business;

5. **Uncertainty** — If doubt and instability persist regarding both personal and organizational futures, it can add an extra layer of stress while contributing to a sense of stagnation in career planning;

6. **No Scope for Skills Implementation** — Despite acquiring skills, the workplace does not provide opportunities to utilize them to their fullest potential, leading to frustration;

7. **Boredom at Work** — Persistent boredom and frequent contemplation of quitting characterize this situation. There can be consistent feelings of being unchallenged in the current role, as the job lacks complexity and fails to engage the necessary skills;

8. **Overloaded with Work** — Despite being a capable employee, feeling overloaded and unrecognized contributes to a sense of stagnation. Efforts going unnoticed and overwhelming tasks add to this feeling;

9. **Workplace Dynamics** — Strained relationships in the professional environment contributes to a sense of stagnation in professional growth. A lack of mutual respect and appreciation inhibits the overall sense of well-being at work.

In response to these challenges, **the development of an automated system emerges as a strategic response to proactively manage and mitigate career stagnation**.

## 3.2 Literature Review

### 3.2.1 MetricsVis: A Visual Analytics System for Evaluating Employee Performance in Public Safety Agencies

An article authored by Jieqiong Zhao, Morteza Karimzadeh, Luke S. Snyder, Chittayong Surakitbanharn, Zhenyu Cheryl Qian, and David S. Ebert, published in 2020 [11] presents a critical exploration into the **impact of career stagnation on employee motivation and productivity**. This research has presented significant insights into the interaction between career stagnation and key factors influencing employee engagement and effectiveness.

The authors emphasize the importance of understanding the correlation between quantitative measurements of employee achievements and supervisor expectations. Furthermore, the research seeks to identify the primary drivers of good performance and explores methodologies to integrate complex and flexible performance evaluation metrics.
To aid in this multifaceted evaluation process, the authors introduce **MetricsVis**, a visual analytics system designed to dynamically assess and compare individual, team, and organizational performance within public safety organizations. The system's functionality is visually depicted in the following image, illustrating the interconnected operation of its various components.

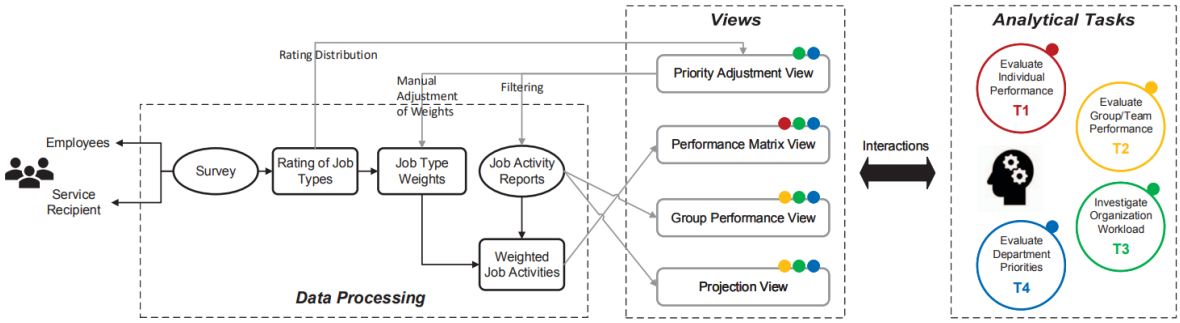



Figure 3.1: Visualization of the MetricsVis system, featuring three modules highlighted, presented in [11].

Most systems follow a common initial step, which involves retrieving data from all relevant parties. This information can be acquired through **surveys** conducted with personnel or by analyzing **job activity reports**, with a specific focus on assigning weights to the various job types. This crucial step is highlighted in the "Data Processing" area depicted in the Figure 3.1.
The system comprises four principal visual components, each serving a distinct purpose in assisting performance evaluation. These components include a priority adjustment view for direct manipulation of evaluation metrics, a reorder able performance matrix view offering detailed insights into individual employee performance, a group performance view highlighting aggregate performance and individual contributions, and a projection view illustrating employees with similar specialties to streamline shift assignments and training. This is the focal point where all the retrieved data is visualized and analyzed, serving as a comprehensive framework for assisting performance evaluation. The forthcoming list describes the different analytical tasks that users can perform within the system, as illustrated in the Figure 3.1, providing a comprehensive overview of the actions and engagements available:

- Evaluate Individual Employee Performance (T1) — In this procedure, activity reports are gathered and modified to gauge the performance of subordinates. It takes into account factors such as the frequency, difficulty, and effort associated with different job types, providing the option to include weighted values. Supervisors are required to analyze the performance of multiple employees, identifying patterns among both low and high-performing individuals;
- Evaluate Group and Team Performance (T2) — Recognizing the pivotal elements contributing to the success of teams is essential. Factors such as location, manager, shift

time, personnel proficiency, and time spent working play a significant role in influencing team effectiveness. Assigning officers with expertise in specific areas to calls within that region can improve patrolling effectiveness. Analyzing these factors becomes instrumental in optimizing personnel allocation strategies;

- Investigate Organizational Workload (T3) — Managers aim to gain insights into strategies for resource and personnel allocation, as well as to understand patterns of change in services and their effects on workload balance. Exploring grouping factors such as locations, time periods, and servicing patterns facilitates an understanding of whether resource expenditure aligns with organizational goals. This exploration also helps identify unexpected drains on resources and pinpoint areas of excess personnel capacity;
- Evaluate Department Priorities (T4) — In instances where organizational priorities endure shifts over time, managers can adapt to these changes by adjusting the weights of evaluation metrics. Given that stakeholders and managers may hold varying opinions regarding the importance of job types or activities, a robust performance evaluation system becomes essential. Such a system enables administrators and managers to thoroughly investigate the impacts of applying different evaluation criteria, fostering adaptability and alignment with evolving organizational goals.

The usability of the MetricsVis framework is demonstrated through two case studies conducted in medium-sized law enforcement agencies. In the first scenario, the chief of a law enforcement agency leverages MetricsVis to build specialized anti-drug teams. By examining historical workforce performance and reviewing incident records, the chief identifies officers who have demonstrated commendable performance in handling drug abuse incidents. This process aids in the identification of potential candidates for the anti-drug team.
The second use case involves the chief seeking to enhance the efficiency of officer performance evaluations. By comparing data-driven officer metrics with subjective evaluations, the chief discovers that this system allows for a more effective and efficient assessment of officer performance. The tools offered enables a more detailed exploration, such as differentiating between day and night shifts, assessing workload distribution, and guiding effective policing strategies. A lieutenant from the highway patrol acknowledges MetricsVis's capability to provide a comprehensive assessment of officers and teams, emphasizing its impact on evaluating performance at both team and organizational levels.
In summary, MetricsVis offers a robust framework for evaluating employee performance. Through **visual analytics and dynamic metrics**, the system addresses the complexities of performance assessment in real-world scenarios. The interconnected visual components facilitate a comprehensive approach to understanding individual, team, and organizational effectiveness.
Drawing parallels to the system that will be developed during the course of this internship, both frameworks share a common goal of addressing dynamic aspects of **employee performance**. The insights gained from this article enhance the effectiveness of career development and performance management, ensuring organizations **optimize their workforce with insight and agility**. In general, both system architectures are similar, providing a

foundation for future references.

### 3.2.2 Employee Promotion Prediction by using Machine Learning Algorithms for Imbalanced Dataset

A conference paper published in the year 2022 [12], written by Kevser Şahinbaş, **explores the significance of promotion processes as crucial components in human resources management**. A fair and well-organized promotion process serves as a managerial tool, motivating employees and contributing to the overall business continuity. For many employees, promotion acts as a significant motivation, fostering engagement, commitment to the organization, and sustaining current performance levels. Additionally, it serves as a vital rewarding mechanism and performance control tool for the organization. This study delves into a prediction methodology based on various criteria, employing machine learning algorithms such as Support Vector Machine (SVM) [13], Artificial Neural Network (ANN) [14], and Random Forest (RF) [15].
In the initial phase, the author employs a dataset sourced from Kaggle [16], a platform renowned for hosting publicly accessible information and promotes various competitions. The table below outlines key attributes crucial for the analysis:

| Attributes | Explanation |
|---|---|
| employee_id | Unique identification number assigned to each employee. |
| department | Department in which the employee is working. |
| region | Geographical region of the employee. |
| education | Highest level of education attained by the employee. |
| gender | Gender of the employee. |
| recruitment_channel | Channel through which the employee was recruited, indicating the number of other trainings completed in the previous year on soft skills, technical skills, etc. |
| age | Age of the employee. |
| previous_year_rating | Employee's rating for the previous year. |
| length_of_service | Length of service in years for the employee. |
| awards_won | Indicates whether the employee won any awards during the previous year. |
| avg_training_score | Average score in current training evaluations for the employee. |
| is_promoted | Target variable indicating whether the employee is recommended for promotion. |

Table 3.1: Attributes analyzed in the dataset by the author.

The writer also addresses the challenge of learning from imbalanced datasets and its impact on the performance of machine learning algorithms. Imbalanced datasets can lead to issues with prediction accuracy, and both internal and external imbalances pose problems.
Hybrid approaches, such as Synthetic Minority Over-sampling Technique (SMOTE) and Random Oversampling (ROS), are introduced to address this issue. SMOTE involves creating synthetic instances for the minority class, focusing on feature space interpolation. On the other hand, ROS randomly duplicates minority class samples to balance class distribution, approximating the large class label. The goal is to achieve equilibrium in sample counts between the minority and majority classes.

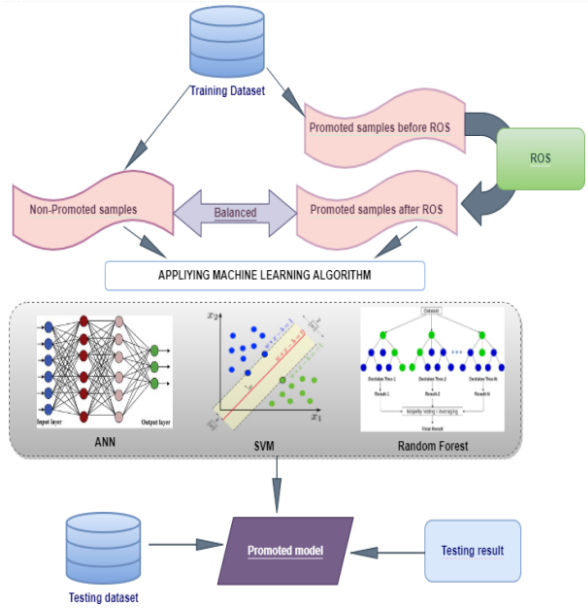



Figure 3.2: Schematic representation of the employee prediction mechanism shown in [12].

The Figure 3.2 provides an overview of the system. The initial phase of the study focuses on balancing the dataset using techniques like SMOTE and ROS, followed by applying machine learning algorithms (SVM, ANN, and RF) to the data. The model's output distinguishes non-promoted individuals as "0" and promoted individuals as "1", presenting comprehensive findings through key performance metrics such as **accuracy** (overall correctness), **precision** (the ability to correctly identify positive cases), **recall** (the ability to capture all relevant positive cases) and **F1-Score** (a combination of precision and recall) [17]. The performance metrics, detailed in Table 3.2, highlight the significance of considering recall and F1-Score alongside accuracy for accurate model selection in imbalanced datasets. Notably, the random forest algorithm exhibited the highest performance, achieving 98% accuracy, 96% precision, 1.0% recall, and 98% f1-score values with the ROS approach.

| Algorithms | Accuracy | Precision | Recall | F1-Score |
|---|---|---|---|---|
| SVM | 0.92 | 0.867 | 0.09 | 0.17 |
| SVM_ROS | 0.726 | 0.756 | 0.667 | 0.70 |
| SVM_SMOTE | 0.756 | 0.766 | 0.736 | 0.751 |
| ANN | 0.9234 | 0.8266 | 0.15 | 0.25 |
| ANN_ROS | 0.729 | 0.788 | 0.627 | 0.698 |
| ANN_SMOTE | 0.77 | 0.795 | 0.728 | 0.76092 |
| RF | 0.9314 | 0.84 | 0.247 | 0.38 |
| RF_ROS | 0.9842 | 0.96 | 1.0 | 0.98 |
| RF_SMOTE | 0.9157 | 0.9118 | 0.92 | 0.915 |

Table 3.2: Performance metrics for different machine learning algorithms.

The findings of this study provide valuable insights for HR professionals and managers to **predict the probability of promotion**, aiding in the identification of the right parameters for **employee advancement**.

The visualization of the metrics within the dataset, along with the strategies to automate this process by employing machine learning algorithms and imbalanced dataset techniques, can establish a connection to the project that will be developed during this internship. Both studies focus on **prediction** and **automation**, and the **similarity in the data** from the datasets and **information manipulation approaches** enhances the alignment between the two projects.

### 3.2.3 Performance Dashboard: Cutting-edge Business Intelligence and Data Visualization

In the years 2017, S. M. Kumar and Meena Belwal wrote a paper for a conference about **business intelligence** and **data visualization** [18].
Business Intelligence is the set of strategies, processes, applications, data, products, technologies and technical architectures which are used to support the collection, analysis, presentation, and dissemination of business information [19]. It enables a wide range of business decisions, from operational to strategic, and it is most effective when combining internal company data with external market data. Visualization technologies help interpret data efficiently, enhancing the understanding of business dynamics.
Visualization is the graphical representation of an information. The major goal of data visualization is to provide the user with a qualitative and easy understanding of the information contents [20]. It is the process of transforming objects, numbers, and concepts into a form that can be easily interpreted by the human eyes. Generally, "information", can be data, relations, perceptions, or processes.
In this paper, the authors critiques existing methods for measuring company growth, pointing out their limitations in providing comprehensive information and analyzing business behavior. It asserts that these methods often fail to offer complete assurance to business owners in decision-making. To address these challenges, the writers propose the development of a **performance dashboard**. This proposed solution integrates business intelligence technologies, data mining, and data visualization techniques to comprehensively analyze business trends, growth, profit, employee performance, customer satisfaction, and areas for improvement.
The performance dashboard is positioned as an information management tool that traces the business behavior from the organization's inception. It serves to track metrics, Key Performance Indicators (KPIs), and additional factors relevant to the business or specific processes.
The following image illustrates the systems technical architecture:

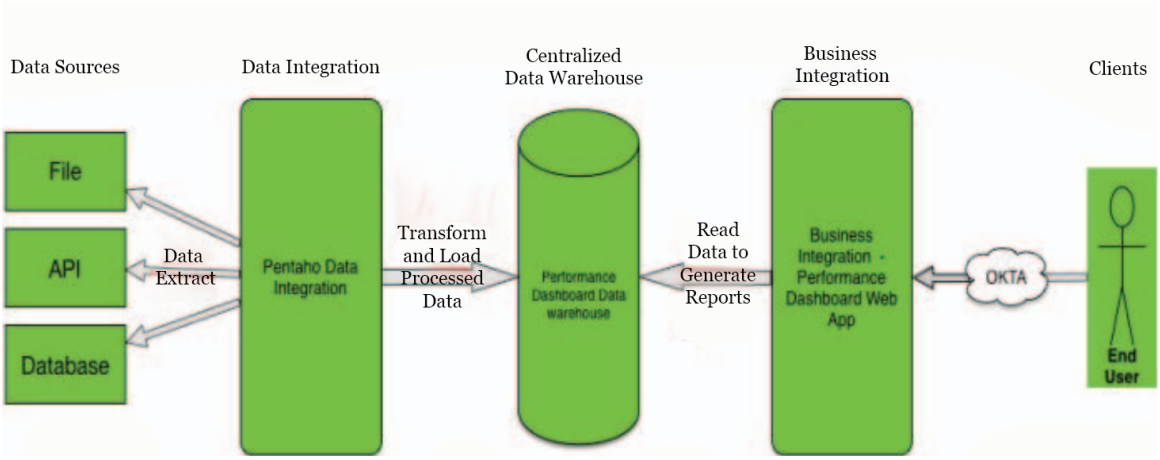



Figure 3.3: Technical architecture of the system presented in [18].

As depicted in Figure 3.3, this system comprises five components, which will be detailed in the list below:

- **Data Sources** — Within an organization, various data sources offer raw input data in diverse formats, such as structured, semi-structured, or unstructured data. These sources may exist on-premise or in the cloud. Common data formats include file-based, API for data extraction, and read-only database schema exposure. Consequently, the dashboard architecture should be versatile enough to handle all types of data;
- **Data Integration** — It facilitates the extraction of data from diverse sources, transforming the extracted data to the designated format (in the author's case, SQL tables), and subsequently loading the transformed data into the data warehouse. Upon identifying data sources, a connection is established to extract data from the source to the application. The extracted raw data may contain noise, necessitating an extract, transform, and load process based on the data type. This process addresses noise reduction in the information before analyzing it. This encompasses providing default values for missing fields based on relations with adjacent data, and eliminating unwanted columns or data not useful for analytical computations;
- **Centralized Data Warehouse** — Enhances data security, reliability, and infrastructure cost reduction, offering additional advantages in the dashboard development process, including easy integration with data mining and machine learning technologies. The dashboard simplifies complex datasets, providing users with a quick understanding of present performance and enabling the tracking of departmental capabilities to achieve service level targets;
- **Business Integration** — It is a key component of the dashboard. While certain software companies offer business integration tools designed for direct integration into the organization, these tools often come with drawbacks. Issues may include exorbitant pricing or a lack of comprehensive functionalities necessary for building robust models. In contrast, the proposed model presents an ideal solution by exclusively leveraging open-source software. This choice offers flexibility, allowing for seamless alterations and integration tailored to the specific requirements of each organization;
- **Clients** — End users can access the dashboard web application through a browser on their computer, laptop, or mobile phone. The design that adapts to different devices makes it easy for clients to use.

After explaining how the system works, the author presents illustrations of the various reports generated, each tailored for different user classes, showcasing relevant information for the company, as depicted in the following images:


(a) Report by average resolution time.

(b) Report by employee performance.

(c) Report by yearly performance.

(d) Report by daily sales.

(e) Report by quarterly sales.

(f) System health report.

Figure 3.4: The results and analysis presented through the dashboard developed in [18].

The proposed performance dashboard offers several advantages, including improved decision-making, consolidation of key performance indicators from various sources, dynamic data handling, implementation of diverse views within a single business intelligence model, role-based views and platform independence. It enables effective insights into organizational performance through various reports, such as daily sales, employee and team performance, yearly or quarterly reports, average resolution time, and system health. The dashboard serves as a real-time user interface, enhancing executive decision-making and improving overall business performance. The system's flexibility allows easy implementation of new metrics, avoiding data redundancy, and ensuring scalability.

The conclusion of this paper highlights the significant benefits achieved by identifying business trends, monitoring growth, analyzing profits, and enhancing employee and customer satisfaction **through data visualization**.

In the context of the project that will be developed during this internship, a key connection lies in the emphasis on information visualization. This conference paper introduces various techniques for presenting data to users in a comprehensible and dynamic manner, utilizing different types of diagrams and emphasizing their significance, which is crucial to this work.

## 3.3 Concluding Remarks

To finalize, this chapter provides a comprehensive overview of the theoretical foundations and existing research relevant to the project. It starts by presenting a detailed definition of career stagnation, exploring its causes and signs, while introduces the concept of an automated career stagnation detection system as a strategic response. The literature review highlights some studies that collectively contribute valuable insights and methodologies that inform and create a fundamental support the development of the proposed system. The synthesis of these diverse perspectives sets the stage for the practical implementation of this project that will be mentioned in the next chapter.

# Chapter 4

# Internship Planning

This chapter serves as the foundation for the internship, commencing with a delineation of the proposed work. Following the first section, it provides a comprehensive overview of all the planned tasks, presenting detailed descriptions and allocated time for each assign respectively. The discussion then transitions to the tools and technologies chosen for its implementation, and it concludes with a concise summary containing all the relevant information mentioned.

## 4.1 Proposed Work

After applying for the internship, the company proposed the development of a project focused on creating a system that automatically detects career stagnation. This involves creating **a predictive model** capable of **retrieving and analyzing information** to autonomously identify signs of career stagnation among employees. The goal is to efficiently modify management practices within organizations.
This project involves the design and implementation of a robust prevision system. Additionally, it necessitates the development of both the frontend and backend, including the creation of a database to store all the relevant information that will be analyzed. The frontend will present a modern and intuitive web interface, providing a dynamic platform for HR professionals and management to easily interpret insights and recommendations, while the backend establishes an infrastructure to handle, process, and access data.
The system's architecture must ensure **security**, **scalability**, and **responsiveness** in handling vast datasets and real-time interactions.
This internship is schedule for a duration of 107 days, commencing on December 20, 2023, and concluding on April 5, 2024, respecting the minimum duration requirement of 15 weeks stipulated by the specific regulations for the second cycle of studies in computer science and engineering at the University of Beira Interior [21].

## 4.2 Overview of Project Tasks

This project will be developed with a total duration of **640 hours**, divided into five distinctive segments, which will be carefully described in this subsection.
To begin, the project's planning and setup will consume **40 hours**. During this initial phase, a meticulous strategy will be formulated to establish the foundational framework for the entire work. A comprehensive breakdown is outlined below:

1. **Define project scope and objectives** (6 hours) — Definition of specific goals and boundaries of the project. This involves establishing clear objectives that align with the

company's vision and requirements, ensuring a comprehensive understanding of the project's scope and purpose;

2. **Research and technology selection** (10 hours) — Thoroughly conduct research on technologies and tools that hold relevance to the project's demands. Implementation of a comprehensive evaluation process to discern and select the most suitable technologies for both backend and frontend development. This involves a meticulous assessment of the latest advancements to ensure optimal alignment with the project's objectives and future scalability;

3. **Database design and setup** (10 hours) — Strategically plan and design the database structure, emphasizing efficiency in data storage and retrieval processes. This implies developing a comprehensive blueprint for the database architecture, ensuring organization and accessibility of data.

4. **Project Documentation and Planning** (14 hours) — Creation of the project documentation that delineates its workflow, key milestones, and assigned responsibilities. This involves making a detailed and well-structured project plan that encompasses timelines and specific marks. The objective is to provide a comprehensive guide that not only outlines the project's direction but also ensures clarity in roles and responsibilities, facilitating effective coordination and project management.

After the realization of a careful project planning, the subsequent step involves the development of the backend, which will encompass a total of **280 hours**. The backend development phase is dedicated to constructing the fundamental functionalities and infrastructure essential for the system. The following list outlines the tasks associated with this phase:

1. **Application Programming Interface (API) development for data input** (60 hours) — Design and implementation of a robust API to facilitate the flow of data input into the system. It emphasizes the importance of secure and efficient data transfer between the frontend and backend components, ensuring the integrity and confidentiality of information throughout the entire process;

2. **Database Integration and Schema Implementation** (50 hours) — Efficiently integrate the database into the backend infrastructure, ensuring seamless connectivity. Implementation of the designed database schema, optimizing it for enhanced performance and responsiveness while prioritizing the establishment of a robust connection between the backend and the database to facilitate smooth data interactions within the system;

3. **Historical Data Storage and Tracking** (50 hours) — Implementation of sophisticated mechanisms within the backend to store and track historical data for thorough analysis. The system needs to guarantee that the backend is proficient in efficiently managing and retrieving historical records;

4. **Forecasting Module Development** (120 hours) — Leverage advanced algorithms and machine learning techniques to construct a robust forecasting module. Integration of these capabilities into the system to predict future trends and outcomes with precision and accuracy, ensuring a forward-looking approach to decision-making.

Following the completion of the backend development, the subsequent step involves dedicating a total of **180 hours** to the frontend development phase. This crucial stage concentrates on crafting a user-friendly and intuitive interface, ensuring a consistent and engaging experience. The following list will thoroughly and carefully explain all the processes involved:

1. **User authentication and authorization** (30 hours) — Creation and deployment of a robust and advanced secure user authentication and authorization mechanisms to fortify the protection of sensitive information. Implementation of a comprehensive access control system that guarantees the highest level of security, prioritizing the confidentiality and integrity of user data;

2. **Employee data display and filtering** (50 hours) — Incorporation of sophisticated features for the correct display of data, while integrating user-friendly filtering options that enhance the overall user experience. The interface needs to be dynamic and not only showcases employee information with clarity but also allows users to have the ability to effortlessly navigate and filter data according to their specific needs;

3. **Historical data visualization** (40 hours) — Deployment of visualization tools to transform historical data into a comprehensible and visually engaging format. This elevates the user experience by incorporating interactive charts and graphs, providing them with a dynamic and intuitive means of interpreting complex data. This implementation not only makes historical data more accessible, but also facilitates insightful analysis.

4. **Alerting mechanism and notification integration** (60 hours) — Incorporation of a robust alerting mechanism designed to promptly notify users of critical events, ensuring a proactive response to time-sensitive situations. This integrated alerting solution serves as a proactive tool, keeping users informed, allowing them to respond swiftly to important events.

After completing the entire system, including both backend and frontend components, the subsequent phase involves testing and quality assurance, demanding a total of **90 hours**. This phase centers on guaranteeing the reliability, security, and performance of the developed system. The following list details all the tasks associated with this essential stage:

1. **Unit testing for backend and frontend components** (40 hours) — Performing a comprehensive unit testing for each individual backend and frontend component. Scrutinize and rectify any identified bugs or issues to ensure the seamless functionality of each module.

2. **Integration testing** (25 hours) — Conduct rigorous integration testing for the coordination of backend and frontend components. Validate data flow and maintain consistency to ensure effective communication between different modules of the system.

3. **Performance Testing and Optimization** (15 hours) — Perform comprehensive performance testing to identify and address potential problems. Optimization of the system for enhanced responsiveness and efficiency, ensuring optimal performance under varying conditions and workloads.

4. **Security testing and vulnerability assessment** (10 hours) — Conduct thorough security testing to identify and mitigate potential vulnerabilities. Ensures that the system aligns with industry-standard security best practices, providing a robust defense against potential threats and safeguarding sensitive information.

The concluding phase, named buffer and iterations, encompasses a total of **50 hours**. During this phase, time will be allocated to address unforeseen challenges and unexpected issues, while also refining and iterating on system components based on testing feedback and user reviews.

To gain a clear visualization of the task durations, the subsequent image showcases a Gantt chart, serving as a visual representation that effectively communicates project timelines and progress. This chart provides a comprehensive overview of the project schedule, facilitating stakeholders in understanding the timeline of each task. In essence, Gantt charts visually represent tasks, durations, and dependencies of a project timeline, utilizing different colors or bars to denote the various phases.

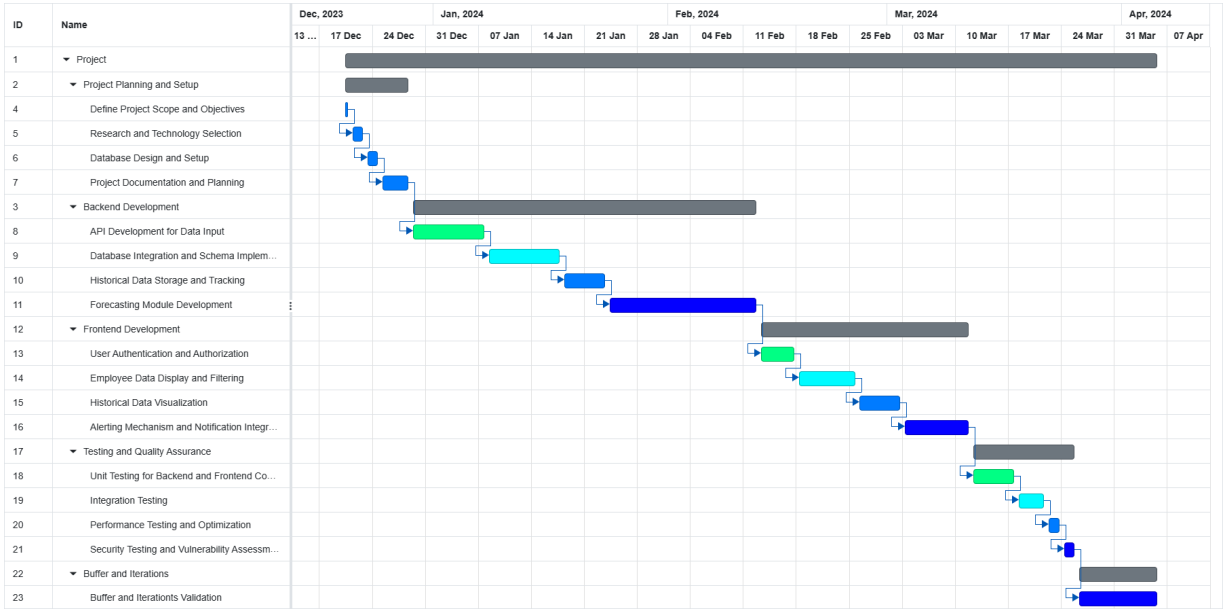



Figure 4.1: Gantt chart illustrating the timeline for all tasks within the project.

## 4.3 Selection of Tools and Technologies

To ensure the consistent development of this project, an in-depth and systematic approach to task planning is essential. As highlighted in the preceding section, an initial goal involves the selection of tools for having an effective implementation. The choice of these technologies may be influenced by company requirements or determined independently based on my discretion and expertise.

### 4.3.1 *Docker*

*Docker* is an open source platform for automating the deployment, scaling, and management of containerized applications. Containers are a way of packaging and isolating programs and their dependencies so that they can run consistently on any system, without being affected by the underlying environment. This technology makes it easier for developers to build, test and deploy applications by providing a consistent environment for them to run. This helps streamline the development and deployment process, making it faster and more reliable [22]. The following image illustrates how *Docker* will containerize this project, isolating distinct components being this the database, the server, and the client, while presenting their relationships.

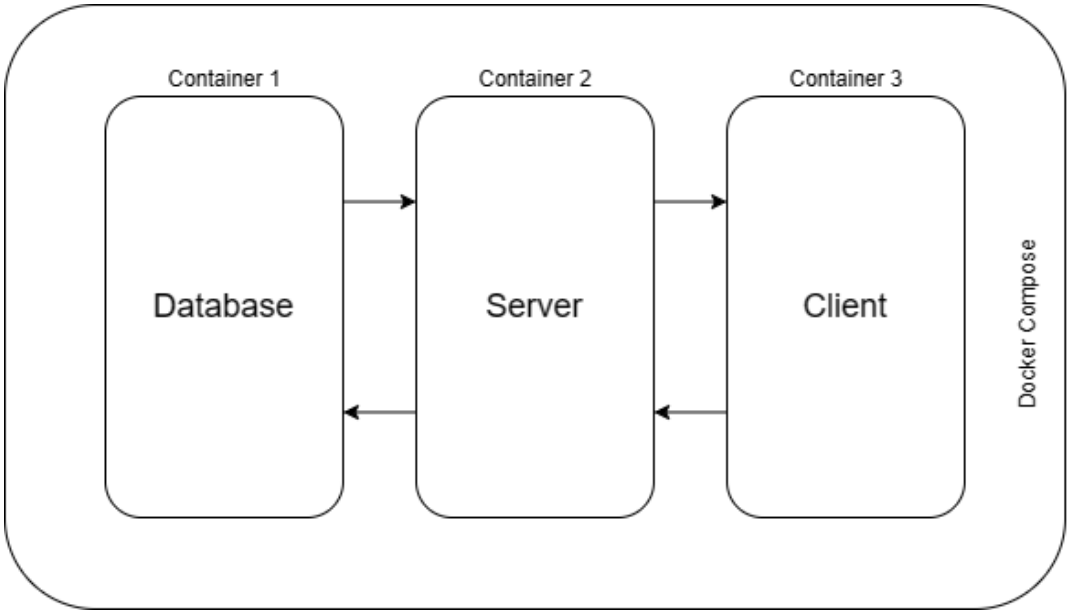



Figure 4.2: Architecture of the containerization of this project.

### 4.3.2 *GitHub*

*GitHub* is a platform for hosting source code and files with version control using *Git* [23]. It allows programmers, utilities or any user registered on the platform to contribute to private and/or open source projects from anywhere in the world. It is widely used by programmers

to publicize their work or for other programmers to contribute to the project, as well as promoting easy communication through resources that report problems or mix remote repositories (issues, pull request) [24]. In this project, this technology will be utilized to upload completed work and facilitate collaboration within the company.

### 4.3.3 Microsoft SQL Server Management Studio

SQL Server Management Studio (SSMS) is an integrated environment for managing any SQL infrastructure. It is used to access, configure, manage, administer, and develop all components of SQL Server, Azure SQL Database, Azure SQL Managed Instance, SQL Server on Azure Virtual Machine (VM), and Azure Synapse Analytics. This technology provides a single comprehensive utility that combines a broad group of graphical tools with many rich script editors to provide access to SQL Servers for developers and database administrators of all skill levels [25]. This tool is used to control the database of the system.

### 4.3.4 *Node.js*

*Node.js* is an open-source, cross-platform JavaScript runtime environment that executes JavaScript code outside the web browser. It is designed to build scalable network applications and is commonly used for server-side programming. This technology is built on the V8 JavaScript runtime engine, which is developed by Google for use in the Chrome web browser [26]. It will be responsible for developing the server-side of this project, facilitating communication between the server, the database, and the client.

### 4.3.5 *React*

*React* is an open-source JavaScript library used for building user interfaces or User Interface (UI) components, particularly for single-page applications where user interaction is key. This technology allows developers to create reusable components and manage the state of an application efficiently following a declarative approach to programming, making it easier to understand and debug [27]. It will play an essential role in crafting the entire frontend experience for this project, serving as the foundation for the user interface and interactive elements.

## 4.4 Concluding Remarks

In summary, this chapter serves as the foundation of the internship, providing an overview of the proposed work and a detailed breakdown of all planned tasks, each accompanied by comprehensive descriptions and allocated time. The project unfolds through five distinctive phases: planning, backend development, frontend development, testing, and a buffer for unforeseen challenges, amounting to a total duration of 640 hours. A Gantt chart is also included to visually convey the project timeline.
The technologies that will be used for the future project's realization include *Docker* for containerization, *GitHub* for collaboration, Microsoft SQL Server Management Studio for

database control, *Node.js* for server-side development, and *React* for frontend crafting. Each technology is crucial in ensuring the success of this work.

# Chapter 5

# System Conception

To ensure the successful realization of this project, the next step, as outlined in the **previously provided project planning**, involves presenting a comprehensive conceptualization, design, and setup for its execution. This chapter delves into the principles of software engineering guiding the formulation of system requirements and architectural design, while also thoroughly addressing the implementation phase. It is structured into two main sections, that ensures a comprehensive exploration of each aspect.
The first section, **software engineering**, focuses on the systematic approach to defining the system's requirements, presenting documentation of all functional and non-functional specifications. It also includes multiple diagrams that illustrate the system's functionality, aiming to create a reliable system architecture.
The second section, **project development**, transitions from planning to action. It provides a comprehensive overview of the development phase, detailing the steps taken to implement the system according to the defined requirements. This section highlights the iterative process of coding, testing, and refinement, ensuring that the final product meets the project's objectives.

## 5.1 Software Engineering

In the ninth edition of a textbook released in 2011 [28], the author delineates software engineering as the engineering discipline concerned with all aspects of software production. In simpler terms, it involves the design, development, testing, maintenance, and various other facets of software application development.
This section is crucial for the successful realization of any project for several reasons [29]:

1. **Time Efficiency**: Utilizing software engineering procedures can significantly reduce development time. By following systematic methods, developers can avoid redundant efforts and streamline the coding process, resulting in faster delivery of functional software;

2. **Reduction of Complexity**: Handling large software projects is inherently complex and challenging. Software engineering provides systematic approaches and methodologies to break down complex problems into manageable components, simplifying the overall process;

3. **Minimization of Software Costs**: This practice can also help optimize the development process by identifying and eliminating unnecessary components, thereby, reducing costs and making the final product more affordable without compromising quality;

4. **Management of Large Projects**: Large-scale projects can be difficult to plan and coordinate without a structured approach. Software engineering can also offer frameworks and tools that facilitate effective project management, maximizing resource utilization;

5. **Reliability**: Reliability is a key attribute of any well-engineered software, ensuring that the final product is robust and dependable. It is essential to deliver software that is secure, performs reliably, and meets the agreed-upon requirements throughout its lifecycle.

In summary, **the principles and practices of software engineering are fundamental to the successful completion of this internship project**, as they provide the essential foundation and methodologies for effectively addressing challenges and ensuring efficient, and accurate development processes.

### 5.1.1 Functional Requirements (FRs)

In a conference paper published in 2022, the author attempts to identify functional and non-functional software requirements [30]. These FRs outline the expected behavior of the software under development, detailing activities, responses to inputs, and the state of entities before and after each activity. According to the paper, functional requirements should address key aspects such as the software's intended actions, potential modes of operation, necessary computations or data transformations, and appropriate responses to various stimuli.
The system to be developed pertains to an automated mechanism designed to detect career stagnation among employees. The functional requirements anticipated by the system are detailed in the following table:

Table 5.1: Functional requirements of the system.

| ID | Requirement name | Description |
|---|---|---|
| FR01 | Employee information access | Display employee information on the dashboard including Identification (ID), name, citizen card, status, type, and a career stagnation detection parameter. |
| FR02 | Employee filtering | Allow users to filter the employee list by name or ID for easy access and navigation. |
| FR03 | Project information access | Provide options to access detailed project information for each employee, including project ID, project name, date, and hours spent. |
| FR04 | Email notification | Enable users to send email warning notifications directly from the dashboard to the employee. |
| FR05 | Career stagnation parameter | Allow users to adjust filters for evaluating career stagnation based on preferences, including fields for maximum, minimum, and offset values for each filter, as well as a weight field to assign importance. |
| FR06 | Visualization of filter impact | Visually illustrate how different inputs affect the career stagnation search results. |
| FR07 | Algorithmic calculation in real-time | Implement an algorithm to calculate the combined impact of all filters on the career stagnation evaluation. Perform the calculation in real-time and provide accurate results based on the current filter settings. |

### 5.1.2 Non-Functional Requirements (NFRs)

In the same conference paper mentioned in the subsection above, the author also presents a description of non-functional requirements. A non-functional requirement specifies the quality attributes that the software solution must possess. In other words, a NFR describes the characteristics and qualities of how our system or application should perform. According to the paper, these include performance, security, reliability and availability, maintainability, usability, and human factors. For this project, the following NFRs have been identified and specified:

Table 5.2: Non-Functional Requirements of the system.

| ID | Requirement name | Description |
|---|---|---|
| NFR01 | Performance | Ensure that the system can handle a large volume of employee data efficiently, with minimal response times for data retrieval and processing. |
| NFR02 | Usability | Design a user-friendly interface that is intuitive and easy to navigate, catering to users with simple levels of technical expertise. Provide clear instructions and guidance to help users effectively utilize the system features. |
| NFR03 | Reliability | Develop a robust system architecture that ensure uninterrupted access to critical functionalities. |
| NFR04 | Security | Implement stringent security measures to safeguard sensitive employee information and prevent unauthorized access. |
| NFR05 | Scalability | Design the system to scale seamlessly with growing user and data volumes, supporting future expansion and additional features without compromising performance or stability. |
| NFR06 | Compatibility | Ensure compatibility with a wide range of devices and platforms, including desktop computers, laptops, tablets, and mobile devices. Support multiple operating systems and web browsers to maximize accessibility and usability. |
| NFR07 | Maintainability | Establish clear documentation and coding standards to facilitate system maintenance and updates. Design modular and well-structured codebase to simplify troubleshooting and debugging processes. |

The previously mentioned requirements, defined by the company where the internship is taking place, **will be revisited in detail during the system's implementation phase**.

### 5.1.3 Database Architecture

The architecture of a database system encompasses its fundamental structural design and methodology, serving as the cornerstone of any Database Management System (DBMS). This framework dictates the organization, storage, and access of data, exerting a significant impact on the efficiency and effectiveness of data management [31].

To visualize the database structure and relationships, we employ a Entity-Relationship Diagram (ERD), which serve as visual representations of the interconnectedness among various entities or tables within the database. ERDs utilize symbols such as rectangles for entities and connecting lines to illustrate the associations. These diagrams operate as flowcharts, elucidating how entities, whether individuals, objects, or concepts, interact within the system, facilitating the design and debugging of relational databases[32].

The following chart illustrates the framework of the database system that will be developed, providing a comprehensive overview of its structure, data elements, and relationships.

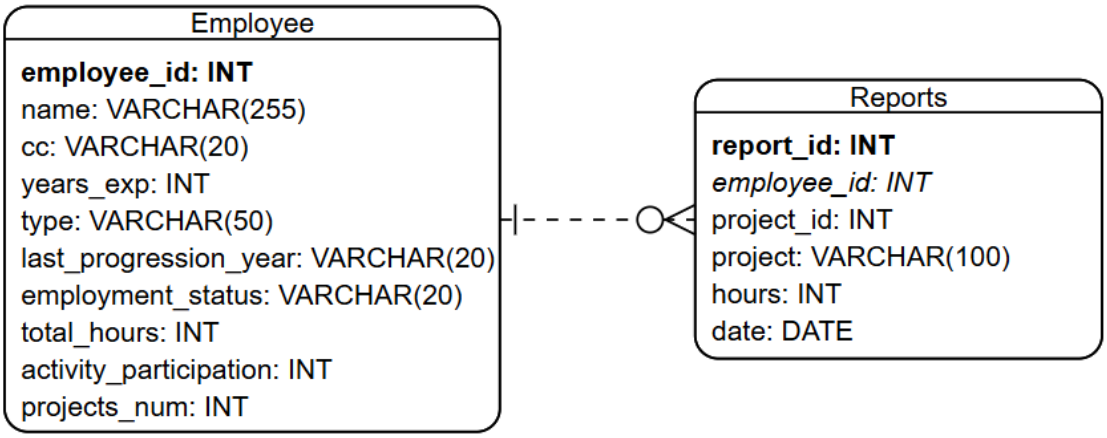



Figure 5.1: Entity-Relationship Diagram illustrating the schema of the database for this project.

During the initial phase of this project, discussions within the company focused on identifying the essential information required for successful project execution. It was collectively agreed to develop a simple and straightforward database consisting of two tables, one for storing employee information and another for housing data related to the reports generated by employees. The following tables present detailed information about all attributes, including their names, data types, a brief description and an example for better comprehension:

Table 5.3: Employee table attributes

| Attribute | Type | Description | Example Data |
|---|---|---|---|
| employee_id | Integer | Unique identifier for each employee. It is the primary key of this table. | 4534 |
| name | VARCHAR | Name of the employee. | John Doe |
| cc | VARCHAR | Character code associated with each employee. | CC123456 |
| years_exp | Integer | Number of years of experience of each employee. | 5 |
| type | VARCHAR | Type of employee. | Full-time/Part-time/Contractor |
| last_progression_year | Integer | Date of the employee's last promotion. | 2022 |
| employment_status | VARCHAR | Status of the employee as per HR records. | Active/Inactive |
| total_hours | Integer | Sum of the time spent by the employee on projects. | 350 |
| activity_participation | Integer | Number of activities that the employee participated within the company. | 8 |
| projects_num | Integer | Total number of different projects that the employee has participated in. | 16 |

Table 5.4: Reports table attributes

| Attribute | Type | Description | Example Data |
|---|---|---|---|
| report_id | Integer | Unique identifier for each report. It is the primary key of this table. | 9876 |
| employee_id | Integer | Indicates the employee to whom the report belongs. It is a foreign key from the table employee. | 4534 |
| project_id | Integer | Identifies the project associated with the report. | 15 |
| project | VARCHAR | Name of the project. | Project alpha |
| hours | Integer | Number of hours spent on the project. | 25 |
| date | DATE | Completion date of the project. | 2024-06-30 |

The relationship between the two tables **is represented by a one-to-many connection**. This means that an employee can generate multiple reports, but a report can only belong to a specific employee. The attribute "employee_id" serves as a foreign key in the "reports" table, linking each report to its respective employee in the other table. This relationship allows for the retrieval of project-related data for each employee, including the total hours spent and the number of projects participated in, which facilitates comprehensive employee performance analysis.

### 5.1.4 System Architecture

An article published in 2022 describes a software architecture diagram as a graphical representation illustrating how the various elements of a system interact within a broader context [33]. These diagrams serve to provide an overview and context of the system's structure and functionalities, depicting the relationships between different components, such as servers, databases, and client applications.

By visualizing the system's architecture, developers can better understand its design and make informed decisions regarding **development, maintenance, and scalability**.

The architecture of the system under development delineates three core components, the database, the server, and the client-side application. These components will be crafted utilizing the selected tools and technologies mentioned in the project planning chapter.

The forthcoming image will provide an overview of the system architecture for this work:

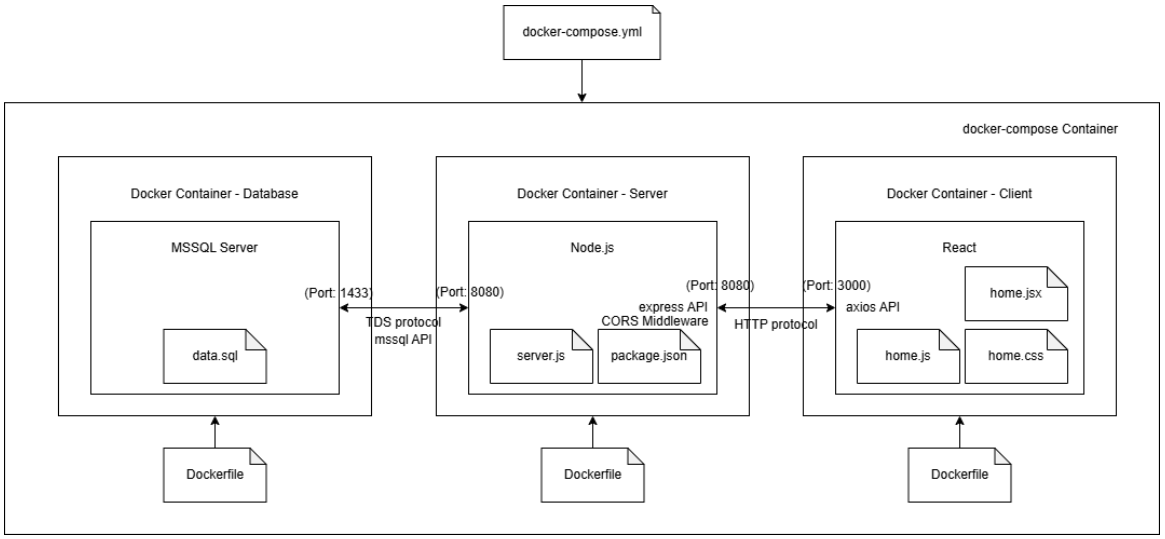



Figure 5.2: Project system architecture overview.

The following text provides a detailed explanation of the system architecture diagram for this project:

First, the docker-compose file orchestrates the creation and integration of all necessary containers. This file defines three distinct containers, each corresponding to a key component of the system, such as **the database, the server, and the client-side application**. Each container is initialized using its specific dockerfile, ensuring that all dependencies and configurations are correctly set up for the application to function as intended.

The initialization process begins with the database container. This container is configured to run in a MSSQL server, using initialization scripts such as “data.sql” to set up the database structure and populate it with essential data. These scripts ensure that the database schema adheres to the requirements specified in the database architecture section. The database container exposes **port 1433**, making it accessible for incoming connections. Communication with the database are managed using the **Tabular Data Stream (TDS) protocol**, and the connection between the database and the server are facilitated through the **MSSQL API**, specifically the *Node.js* mssql module.

Following the database setup, the server container is initialized. The server application, built using the Express.js framework, starts up by running the “server.js” file. This server listens for incoming Hypertext Transfer Protocol (HTTP) requests on **port 8080**. The server connects to the database using the MSSQL API, enabling it to execute SQL queries and retrieve or manipulate data as required. Additionally, the server employs Cross-Origin Resource Sharing (CORS) middleware to manage and secure cross-origin requests, ensuring that the client-side application can communicate seamlessly with the server despite being hosted on a different port or domain.

Lastly, the client-side application container is initialized. This application is developed using *React*, a popular JavaScript library for building user interfaces. The main page of the client-side application is served on **port 3000**, providing an interactive and responsive interface for users. The client-side code includes various files which define the components and styles of the application. To facilitate communication with the server, the client-side application

uses Axios, a promise-based HTTP client for the browser and *Node.js*. Axios simplifies making asynchronous HTTP requests to the server, allowing the client to send data to the server and receive responses effectively.
In this architecture, *Docker* is utilized to ensure that each component operates in a consistent and isolated environment. Overall, the database container handles data storage and retrieval, the server container manages application logic and data processing, and the client-side container provides the user interface.
To facilitate comprehension, the following list represents a detailed explanation of all the components involved in the system, including their **APIs, ports, and roles**:

- **Database (MSSQL)**:
  - **API:** MSSQL API: Connects to the server module (Node.js);
  - **Port:** 1433;
  - **Role:** Manages data storage and retrieval.
- **Server (*Node.js*):**
  - **APIs:**
    * Express API: Manages routes, middleware, and HTTP requests;
    * MSSQL API: Connects to the MSSQL database module;
    * CORS Middleware: Handles cross-origin requests.
  - **Port:** 8080;
  - **Role:** Processes incoming requests, interacts with the database, and serves data to the client-side application.
- **Client-Side Application (*React*):**
  - **API:** Axios: Facilitates HTTP requests to the server;
  - **Port:** 3000;
  - **Role:** Provides the user interface and manages client-side interactions.

**The detailed content of the files and the comprehensive setup of the system will be thoroughly discussed in the project development section**.

### 5.1.5 Sequence Diagram

After gaining a comprehensive understanding of the database operation and the entire system, the next step is to comprehend the tasks users can perform within the system. Various diagrams, such as use case, activity diagrams, or sequence diagrams, aid in this endeavor.
Sequence Diagrams **are interaction diagrams that detail how operations are carried out**. They capture the interaction between objects in the context of a collaboration. They are time focus, and they show the order of the interaction visually by using the vertical

axis of the diagram to represent time what messages are sent and when [34]. It also illustrates the sequence of messages exchanged between these components or actors in response to various stimuli or events.

In the context of this project, a sequence diagram can elucidate how different system components communicate with each other to fulfill user requests or execute specific functionalities. The following image illustrates a sequence diagram to demonstrate the main operations the user can perform and how the system responds.

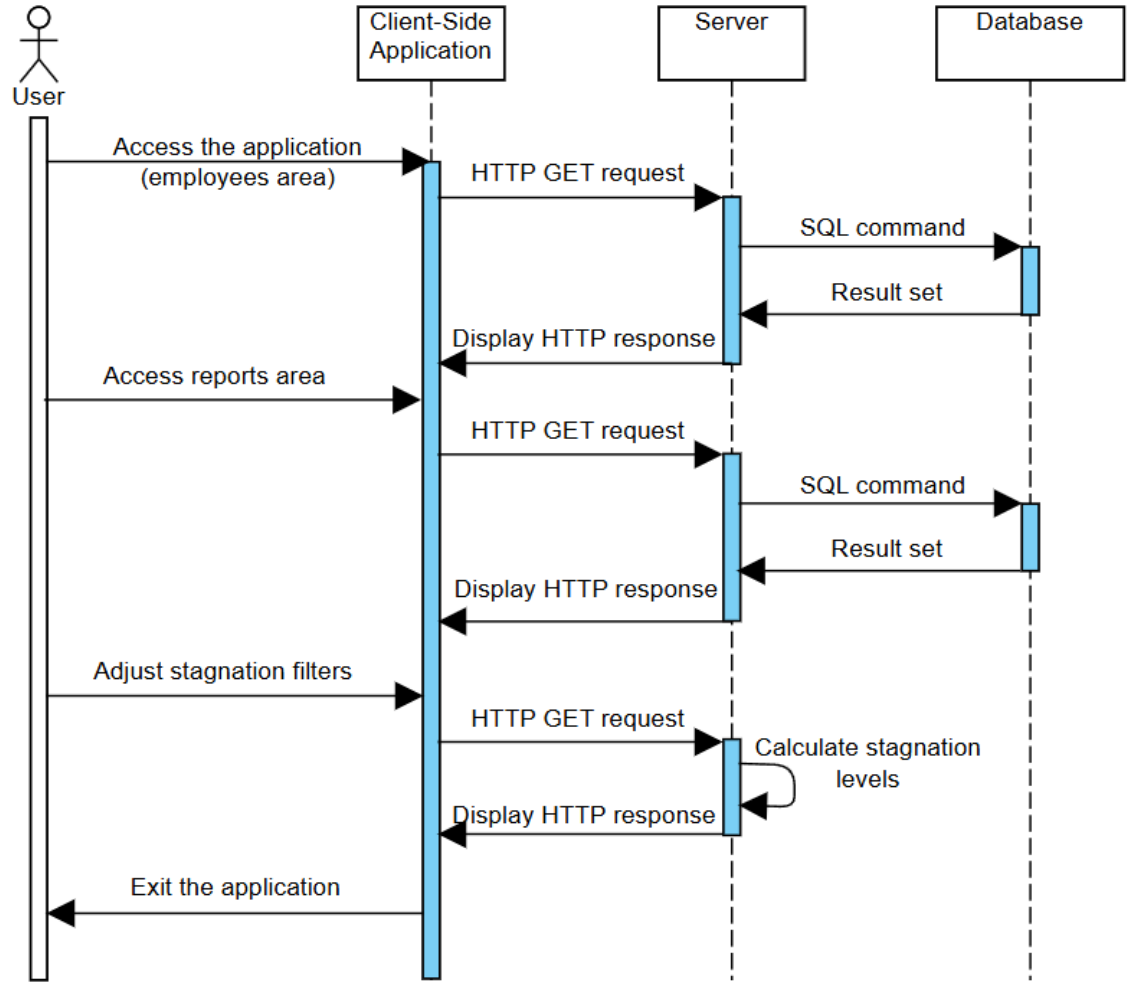



Figure 5.3: Sequence diagram of the main tasks the user can perform in the system.

In the depicted sequence diagram, three primary actions are initiated by the user. Firstly, when the user accesses the application, it triggers the loading of all employee information from the database, initializing a process to fetch the required data. Initially, an HTTP GET request is dispatched to the server, prompting the server to execute an SQL command querying the database for the necessary data. Subsequently, the database retrieves the requested data and returns the result set to the server. Upon receiving the data, the server generates an HTTP response, delivering the information to the interface, where it is displayed for the user.

Similarly, when users navigate to the report area, a similar process unfolds. Upon accessing the report section, an HTTP GET request is dispatched to the server, initiating an SQL

query to retrieve the relevant project information from the database. The database then processes the request, retrieves the specified project data, and transmits it back to the server. The server, upon receiving the data, generates a HTTP response, rendering the project information on the interface for the user to view and analyze.
Another user action depicted in the sequence diagram involves adjusting the stagnation filter settings. When users modify the stagnation filter parameters, such as maximum, minimum, or offset values, an HTTP GET request is sent to the server, containing the updated filter criteria. Upon receiving the request, the server processes the filter settings, calculating the corresponding filter levels based on the provided parameters. Subsequently, the server generates an HTTP response, delivering the updated filter information to the user interface, where it is displayed for the user's reference and analysis.
During this internship, there was a specific requirement to minimize access to the database. Consequently, data is fetched only when accessing different pages within the application. Once the initial data load is complete, subsequent manipulations, such as filtering and search queries, are performed solely on the data stored on the server and not retrieved from the database again. While this approach enhances system performance by reducing database queries, there is a potential drawback, if the page is not reloaded regularly, the displayed values may become outdated.
In conclusion to this section, the exploration of software engineering principles **has been highly beneficial in providing us with invaluable insights into system development methodologies**. By examining functional and non-functional requirements, analyzing the database and system architectures, and exploring multiple diagrams, we have gained a thorough understanding of the diverse facets of this work. This knowledge will provide us with a clearer vision of the work ahead, ensuring efficient and effective project development. With this foundation, **we are ready to proceed to the next phase of the project and initiate the construction of the system**.

## 5.2 Project Development

The transition from planning to implementation marks a significant phase in the project lifecycle. In this section, we detail the steps taken to develop the system in accordance with the defined requirements outlined in the previous chapter.
While the initial plan provides a roadmap for development, **it's essential to note that certain tasks may evolve or undergo adjustments during the development phase**. Factors such as personnel decisions, company directives, or time constraints, given the four-month duration of the internship, may influence the execution of the planned tasks. Nonetheless, this section elucidates the progress made in transforming conceptual plans into tangible software solutions.

### 5.2.1 *Docker* Configuration and Containers Setup

The initial phase of the project involved configuring *Docker*. As a newcomer to this technology, the learning process required some time. As a result, the *Docker* Desktop version was

installed to streamline the process with its graphical user interface and simplified console commands, raising efficiency [35].

This version provides a straightforward Graphical User Interface (GUI) that can manage containers, applications, and images. It also reduces the time spent on complex setups. It takes care of port mappings, file system concerns, and other complex default settings while providing visual insights into the created containers, aiding in monitoring and management.

To set up the containers, each component required its **Dockerfile**. This file is a text document that contains instructions for building a *Docker* image. It defines the environment and configuration needed for an application to run inside a *Docker* container. Dockerfiles typically include commands to install dependencies, copy files into the container, configurations about the environment, and specifies the entry points for the applications.

When a Dockerfile is used to build an image, it executes the instructions in the file step-by-step, creating layers that form the image. These layers represent incremental changes to the file system and settings of the container. Once built, the *Docker* image can be used to create and run containerized instances of the application.

For instance, the Dockerfile for the database container includes the following content, as illustrated by the following code excerpt:

Listing 5.1: Dockerfile for the database container.

```
# Use the official SQL Server image
FROM mcr.microsoft.com/mssql/server

# Set environment variables like the password and accept
# the End User License Agreement for SQL Server
ENV SA_PASSWORD=P@ssw0rd!2022
ENV ACCEPT_EULA=Y

# Create a directory for storing database scripts
WORKDIR /database

# Copy SQL Server initialization scripts
COPY ./database/entrypoint.sh .
COPY ./database/data.sql .

# Set the user to root for executing privileged commands
USER root
RUN chmod +x ./entrypoint.sh

# Expose SQL Server port
EXPOSE 1433

# Run the entrypoint script on container startup
CMD /bin/bash ./entrypoint.sh & /opt/mssql/bin/sqlservr >
/dev/null 2>&1
```

This Dockerfile is designed to configure a *Docker* container for hosting a SQL Server database. The first line specifies the base image to use for the container, pulling the official SQL Server image from the Microsoft Container Registry. The following lines set environment variables to specific password for the SQL Server system administrator (SA) account, and to indicate acceptance of the End User License Agreement for the SQL Server.
Afterward, the working directory inside the container is set to "database", where the database initialization scripts will be copied. Two scripts, entrypoint.sh and data.sql, are then duplicated from the host machine to the current directory in the container. The entry point script contains the setup and the initialization logic, while the data.sql includes SQL commands to initialize and populate the database.
The user context is switched to root within the container, allowing privileged commands to be executed. The permissions of entrypoint.sh are then modified to make it executable. The port 1433, is also exposed, allowing external connections to the SQL Server instance running inside the container.
Finally, the command to be executed when the container starts are specified. This command runs entrypoint.sh in the background and starts the SQL Server process. Any output from this process is redirected to /dev/null to suppress it because the console would be full of information about the process overwise.
In summary, this Dockerfile sets up a *Docker* container with SQL Server installed, initializes it with the provided scripts, and exposes it on port 1433.
After this initial setup, a docker-compose file was formulated to facilitate the orchestration and management of multiple containers as a cohesive system. This file serves as a central configuration hub, defining the services, networks, and volumes required for the entire application stack. For this work, the following docker-compose.yml was created, as shown in the next code snippet.

Listing 5.2: Part of the Docker compose configuration file of this project.

```
version: '3'  # Docker Compose file version
services:
  database:  # Configuration for the database service
    build:  # Specifies how to build the service
      context: ./ACSDS_Database  # Directory containing the Dockerfile
    container_name: ACSDS_Database  # Assigns a name to the container
    working_dir: /database  # Sets the working directory
    tty: true  # Allocates a pseudo-TTY for interaction
    ports:  # Maps host ports to container ports
      - "1433:1433"  # Maps port 1433 on the host to port 1433
    networks:  # Connects the container to networks
      - backend  # Connects to the backend network
  backend: # Configuration for the backend service (not in this snippet)
    ....
```

This file sets the initial setup of the project. The version number three specifies the version of *Docker* Compose syntax being utilized.

Within this *Docker* Compose file, configurations for three distinct services are delineated, the database, the backend, and the frontend. However, the snippet of code presented exclusively details the configuration of the database service for the sake of simplicity. Under the services section, distinct components of the system are defined. For instance, the database service is configured to build from a Dockerfile located in the "ACSDS_Database" directory. The working_dir directive establishes the working directory within the container as /database, indicating where commands should be executed.
The line, "tty: true", only ensures that the container remains interactive. The ports section facilitates port mapping, allowing external access to the SQL Server instance by mapping port 1433 on the host to port 1433 on the container.
Lastly, the networks directive connects the database service to the backend network, enabling communication between the database and other services in the system.
Similar configurations are applied to other services like the backend, the server side and the frontend, the client side, customizing their setup to their specific roles and requirements within the system architecture.

### 5.2.2 Database Implementation

After the initial container configuration, **we can execute a single command to initiate the database, server, and client-side applications simultaneously**. Additionally, we have the flexibility to edit each container independently without affecting the others.
Next, the predefined values from the software engineering section will be introduced by populating the data.sql file within the database container. This file contains table definitions and the insertion of random values, generated internally for security reasons as requested by the company. Some example of these values are illustrated in the following tables:

Table 5.5: Employee Information

| employee_id | name | cc | years_exp | type | last_progression_year | employment_status | total_hours | activity_participation | projects_num |
|---|---|---|---|---|---|---|---|---|---|
| 1 | John Doe | CC123123 | 5 | Full-time | 2 | Active | 40 | 1 | 3 |
| 2 | Jane Smith | CC456321 | 3 | Part-time | 1 | Inactive | 20 | 4 | 4 |
| 3 | Bob Johnson | CC789423 | 4 | Full-time | 3 | Active | 35 | 3 | 5 |
| 4 | Alice Williams | CC321432 | 2 | Contractor | 4 | Active | 25 | 2 | 3 |
| 5 | Charlie Davis | CC654342 | 6 | Full-time | 5 | Active | 30 | 5 | 3 |
| ... | ... | ... | ... | ... | ... | ... | ... | ... | ... |

Table 5.6: Reports Information Example

| report_id | project_id | employee_id | project | hours | date |
|---|---|---|---|---|---|
| 18 | 1 | 1 | Project Alpha | 4 | 2023-01-15 |
| 19 | 2 | 2 | Project Beta | 3 | 2023-02-28 |
| 20 | 3 | 3 | Project Gamma | 1 | 2023-03-15 |
| 21 | 4 | 4 | Project Delta | 8 | 2023-04-01 |
| 22 | 5 | 2 | Project Epsilon | 2 | 2023-05-01 |
| ... | ... | ... | ... | ... | ... |

**The type of each attribute and its description were already addressed in the previous section, during the discussion of the database architecture**.
After populating and correctly configuring the database, the next step is to set up the server. This involves configuring the server settings and establishing a connection between the server and the database to enable data retrieval and ensure seamless communication between the two components.

### 5.2.3 Server Configuration

In this subsection, we delve into the server set-up, focusing on the various components, protocols, and logic behind the code. It will be discussed the tools and technologies used, the setup process, and the functionality provided by the server.
The server is built using *Node.js*, a powerful JavaScript runtime that allows for building fast and scalable network applications. For database interaction, the mssql module is utilized. This module provides a straightforward interface to connect, query, and manage Microsoft SQL Server databases.
The first step in setting up the server is configuring the database connection. The mssql module requires a configuration object that includes the database credentials and connection details. This includes the username, password, server address, database name, and an option to trust the server certificate. The configuration is crucial as it ensures that the server can authenticate and establish a secure connection to the SQL Server database. **This information needs to match the one that was defined when creating the database, including the values from the respective Dockerfile**, as shown in the following code block:

Listing 5.3: Database connection configuration from the server side.

```
1  const sql = require('mssql'); // Importing the 'mssql' module.
2  // Database configuration
3  const config = {
4      user: 'sa', // SQL Server username.
5      password: 'P@ssw0rd!2022', // SQL Server password.
6      server: 'database', // SQL Server hostname.
7      database: 'employerDB', // Database name.
8      trustServerCertificate: true, // Trust server certificate.
9  };
```

The connection function, “connectToDatabase”, is an asynchronous function that attempts to establish a connection to the SQL Server using the provided configuration. In case of an error, it catches the error and logs a message indicating that it will retry the connection. This ensures that temporary connection issues do not prevent the server from functioning. This function will be presented below:

Listing 5.4: Function that establish the connection to the SQL Server.

```
async function connectToDatabase() { // Function that makes the connection.
  try {
    // Establishing a connection with the database.
    await sql.connect(config);
  } catch (err) {
    // Log error if connection fails.
    console.log('Connecting to the database. Retrying...');
  }}
```

To ensure the server is fully functional before handling requests, it first establishes a connection to the database. This is managed by the Connection function, which calls “connectToDatabase” and, upon successful connection, starts the server. The server listens for incoming HTTP requests on port 8080, as specified in the following function.

Listing 5.5: Function that starts the server.

```
function startServer() { // Function to start the Express server.
  // Instructing the Express application to listen for incoming HTTP requests
      on port 8080.
  app.listen(8080, () => {
    // Logging a message to the console indicating that the server is running
       and listening on port 8080.
    console.log('Server listening on port 8080');
  });}
// Calling the 'Connection' function to establish the connection to the
   database and start the server.
Connection();
```

The server uses *Express*, a minimal and flexible *Node.js* web application framework. It provides a versatile set of features for web and mobile applications, making it an ideal choice for handling HTTP requests. The *Express* application is set up to use CORS middleware. CORS is a security feature implemented by web browsers to control how web pages in one domain can request and interact with resources hosted on another domain. By integrating it, the server can handle requests from different origins, raising its interoperability and flexibility. After initializing the server and establishing the connection with the database, the code it’s just lacking the definition of endpoint handlers to manage various types of requests. These handlers are responsible for interacting with the database and returning the requested data to the client. There are multiple handlers, and for simplicity, only one will be shown in full detail.

One of the route handlers is intended to fetch project data for a designated employee. It operates by accepting an employee_id as a parameter and then querying the projects table for records corresponding to that employee ID. This functionality allows users to dynamically request comprehensive project details for individual employees. This handler specifically serves when users navigate to the project page of a particular employee. Below, it's presented a code snippet elucidating the construction of such route handlers:

Listing 5.6: Route handler for employee project data retrieval.

```
// Route handler to fetch project data for a specific employee
app.get('/projects/:employeeId', async (req, res) => {
  // Extract the employeeId from the request parameters
  const { employeeId } = req.params;
  try {
    // Connect to the database
    await connectToDatabase();
    // Query the projects table for records corresponding to the employeeId
    const result = await sql.query`SELECT * FROM projects WHERE n_empregado =
        ${employeeId}`;
    // Send the retrieved data as a JSON response
    res.json(result.recordset);
  } catch (error) {
    // Log any errors encountered while fetching data
    console.error('Error fetching projects data:', error);
    // Send an internal server error response
    res.status(500).json({ error: 'Internal Server Error' });
  }
});
```

This code snippet is an Express.js route handler designed to fetch project data for a specific employee. When a client requests data from the '/projects/' endpoint, this handler is triggered. It first extracts the employee_id parameter from the request Uniform Resource Locator (URL). Then, it attempts to connect to the database and executes a SQL query to select relevant project records. If successful, the handler sends the retrieved data back to the client as a JavaScript Object Notation (JSON) response. However, if any database operation fails, it logs the error and returns a 500 Internal Server Error response to the client.
After addressing the server-side configuration, including integrating *Node.js* with Microsoft SQL Server using the mssql module, setting up the *Express* framework, and defining route handlers for database operations, the next step involves developing the client-side or frontend of the application. This frontend development phase will focus on creating user interfaces, implementing interactive components, and integrating them with the backend services provided by the server.

### 5.2.4 Frontend Development

The frontend of this application is developed using *React*, a popular JavaScript library for building user interfaces. Typically, in web applications, the client side comprises HyperText Markup Language (HTML), Cascading Style Sheets (CSS), and JavaScript, often utilizing frontend frameworks such as *React* or *Angular*.
In 2023, a conference paper was published conducting a comprehensive comparison and analysis of prevalent frontend frameworks and libraries [36]. This scholarly work accentuates the critical role of selecting the appropriate framework and ensuring the quality of its development. By looking into the characteristics of various frontend tools, the paper underscores the profound impact that these choices wield on the efficacy and sophistication of web development endeavors.
The frontend serves as the initial point of contact for users upon visiting a website, setting the stage for their entire experience. Its functionality, appearance, and usability collectively shape the impression of users, and it can be positive or negative. A well-crafted frontend not only captivates and sustains user interest but also motivates engagement and retention. In this project, it can influence the **perception and efficiency** of HR professionals who are using the application to address career stagnation among employees.
The initial phase of interface development involves creating mockups of multiple pages, adhering to the functional and non-functional requirements outlined previously. The following images provide an initial glimpse of a basic mockup of the application, accompanied by legends explaining the placement of key functionalities:

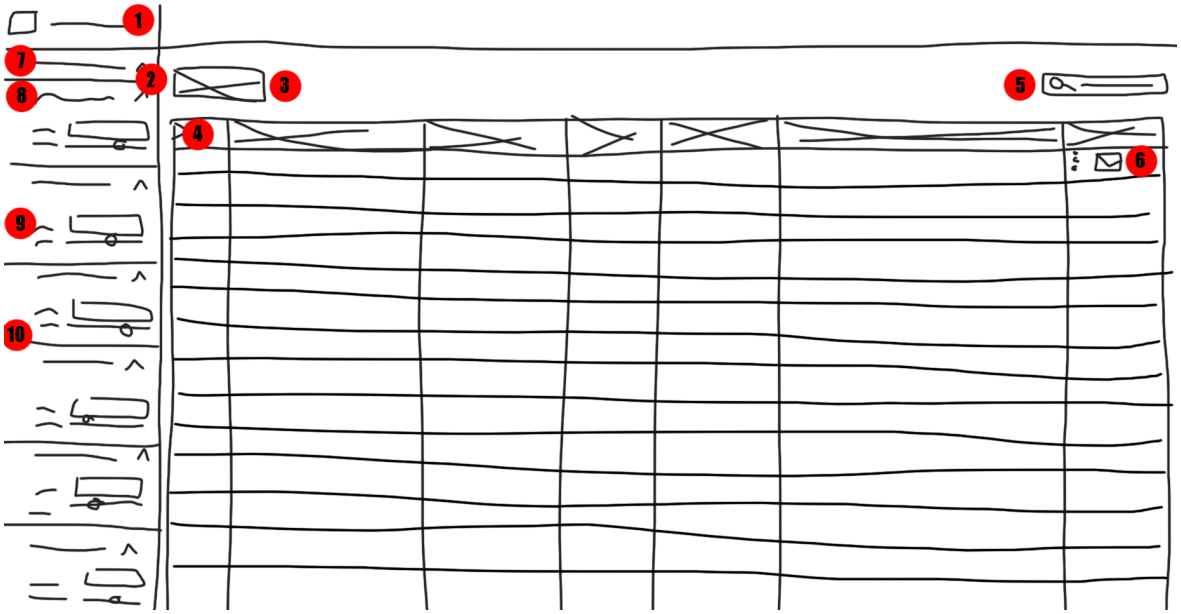



Figure 5.4: Initial mockup of the page with the information of the employees.

1. **Logo and branding:** Positioned prominently in the top-left corner for brand recognition;

2. **Sidebar with multiple filters:** A functional sidebar on the left-hand side providing various filter options for refining employee search results;

3. **Name of the table displayed:** Clearly labeled to indicate the type of data being presented;
4. **Location of the table displayed:** Indicates where the table showcasing employee information is located within the interface;
5. **Search field:** Allows users to input employee names or IDs for quick search functionality;
6. **Options field:** Offers additional functionalities such as sending notifications to employees or accessing project reports;
7. **Career stagnation filter selection:** Enables users to open the area where they can change the stagnation criteria;
8. **Career stagnation filter name:** Clearly labeled to indicate the specific parameters that are being changed;
9. **Career stagnation filter metrics:** Displays metrics such as maximum, minimum, and offset values used to calculate the career stagnation level;
10. **Career stagnation filter weight sliders:** Interactive sliders allowing users to adjust the weightage of different career stagnation metrics for refined filtering.

After refining and coding in the foundational frontend languages, the culmination of these efforts will be showcased in the following image:

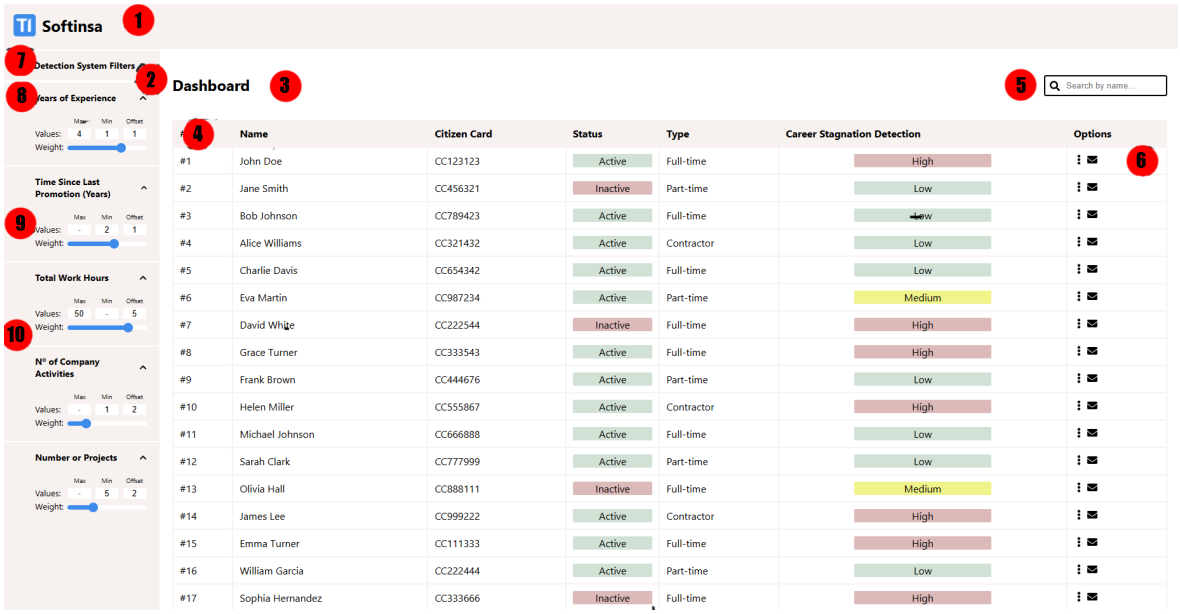



Figure 5.5: Final mockup of the page with the information of the employees.

A significant addition to the application, respecting the requirements stipulated, includes a dedicated page where all reports associated with a single user are prominently displayed. Accessible through the options settings, users can conveniently navigate to this section to

gain comprehensive insights into individual performance and progress. The following image illustrates this page:


Figure 5.6: Final mockup of the page with the information of the reports of a specific employee.

The evaluation and description of user actions, interface usability, performance, and utility will be intricately detailed within the testing and quality assurance section, guaranteeing a comprehensive assessment of the application's functionality and user experience. But before delving into the testing fase, it's imperative to address the foundational code that defines the frontend interface and it's relationships with the server.
This interaction, where users transmit request data to the server, is an important step in the application's functionality and data flow. To facilitate this communication, several steps are involved:

1. **Client-side preparation**: Before sending a request to the server, the client-side code must be prepared to gather the necessary data. This may involve user input from forms, interaction with UI elements, or other triggers that prompt the need for data transmission;

2. **Request initialization**: Once the client has collected the required data, a request must be initialized. This typically involves creating an HTTP request object with the appropriate method like, GET, POST, PUT, DELETE;

3. **Data transmission**: With the request initialized, the client sends the request data to the server over the network.

4. **Server-side handling**: Upon receiving the request, the server-side code processes the incoming data. This may involve parsing the request payload, validating input, performing business logic, querying databases, or executing other operations necessary to fulfill the request;

5. **Response generation**: After processing the request, the server generates a response to send back to the client;

6. **Client-side response handling**: Finally, the client-side code receives the server's response and handles it accordingly. This may involve extracting data from the response, updating the UI to reflect the result, or handling any errors that occurred during the request.

These essential steps of client-server interaction can be seen in the following segment of the code, which effectively demonstrates the process of fetching and handling data from a server:

Listing 5.7: Obtaining data from the client side perspective separated in different steps.

```
//Step 1 - Client-side preparation.
const [tableData, setTableData] = useState([]); // State to store table data.
--------------------------------------------------------------------------------

//Step 2 - Request Initialization.
const fetchData = useCallback(async () => {
try {
    //Step 3 - Data transmission - The request is sent to the server:
    const response = await axios.get('http://localhost:8080/data');
    //Step 4/5 - Server-side handling/Response generation. Update the data.
    setTableData(response.data);
} catch (error) {
     console.error('Error fetching data:', error);
}
}, []);
// useEffect hook to fetch data when the component mounts.
useEffect(() => {
    fetchData(); // Fetch data when the component mounts.
}, [fetchData]);
--------------------------------------------------------------------------------

//Step 6 - Updates the UI accordingly
return (
<div>
    {/* Render the table data */}
    {tableData.map((item, index) => (
    <div key={index}>
        {item.name} {/* Display the name of each item in the tableData array
            */}
    </div>
    ))}
</div>
);
```

The provided code exemplifies the process of client-server interaction within a *React* component. The process starts with the client-side preparation, where the "useState" hook initializes a state variable, "tableData", to store the data that will be fetched from the server. This is the first step, ensuring the component is ready to manage and display the incoming data. Next, the request initialization occurs within the "fetchData" function, defined using the "useCallback" hook to prevent unnecessary re-renders. This function prepares to send a request to the server. The actual data transmission is managed by an axios.get request, which targets a specified endpoint ('http://localhost:8080/data'). This request sends the necessary information from the client to the server.

Upon receiving this request, the server processes the incoming data, performs the required operations, and generates a response. This response is then sent back to the client. In the

context of the code, once the server responds, the response data is used to update the "tableData" state. This step encompasses both the server-side handling and the response generation.
Finally, the client-side response updates the user interface accordingly. The component renders the "tableData" by mapping over each item in the array and displaying its name property. This concludes the cycle, where the client receives the processed data and updates the UI to reflect the current state of the data fetched from the server.
In summary, all the connections between the client-side and the server are effectuated in this manner. By understanding each step of this interaction, developers can effectively design, implement, and test the communication between the frontend interface and the server, thereby ensuring the correct flow of data and the optimal functioning of the application. This comprehensive understanding not only aids in debugging and enhancing performance but also provides a solid foundation for scaling and maintaining the application over time.
The final aspect to cover is the explanation of the career stagnation detection system algorithm and the functionality of its filters.

### 5.2.5 Filters Functionality and Stagnation Detection Algorithm

To meet the company's requirements, it was essential to design the system to be as simple and user-friendly as possible. **The user must be able to change the input parameters according to their preferences**.
The initial idea was to utilize a machine learning algorithm due to its typically high accuracy and potential for full automation. However, several reasons led to the decision not to pursue this approach:

1. **Data acquisition:** Gathering relevant information to train a model proved to be challenging. Obtaining employee related data required conducting extensive data collection within the company, which posed logistical difficulties;

2. **Time constraints:** Retrieving and organizing the necessary information demanded significant time investment. Given the project's tight schedule, allocating time for data collection was impractical since it was not in the original planning;

3. **Complexity:** Developing a prediction model with accurate parameters is not only extremely time-consuming but also a difficult process. There is the need to have a deep understanding of the parameters involved and that would require additional learning and resources;

4. **Dynamic nature:** One of the system's requirments is to adapt to changing user parameters dynamically. Employing a machine learning approach would necessitate frequent retraining of the model to accommodate these changes.

Therefore, the focus shifted towards a more straightforward data analysis approach rather than a complex machine learning model. This method allows for real-time adjustments and ensures the system remains intuitive and adaptable to completely different user inputs.

The filter functionality enables users to dynamically adjust the thresholds and weights for each performance metric, directly influencing the career stagnation detection algorithm. Users can input specific values for the maximum, minimum parameters and define an offset in the input fields corresponding to each performance metric.
The following image will provide a clearer illustration of how user inputs affect the system's output, categorizing them as **high, medium, or low** for each parameter.

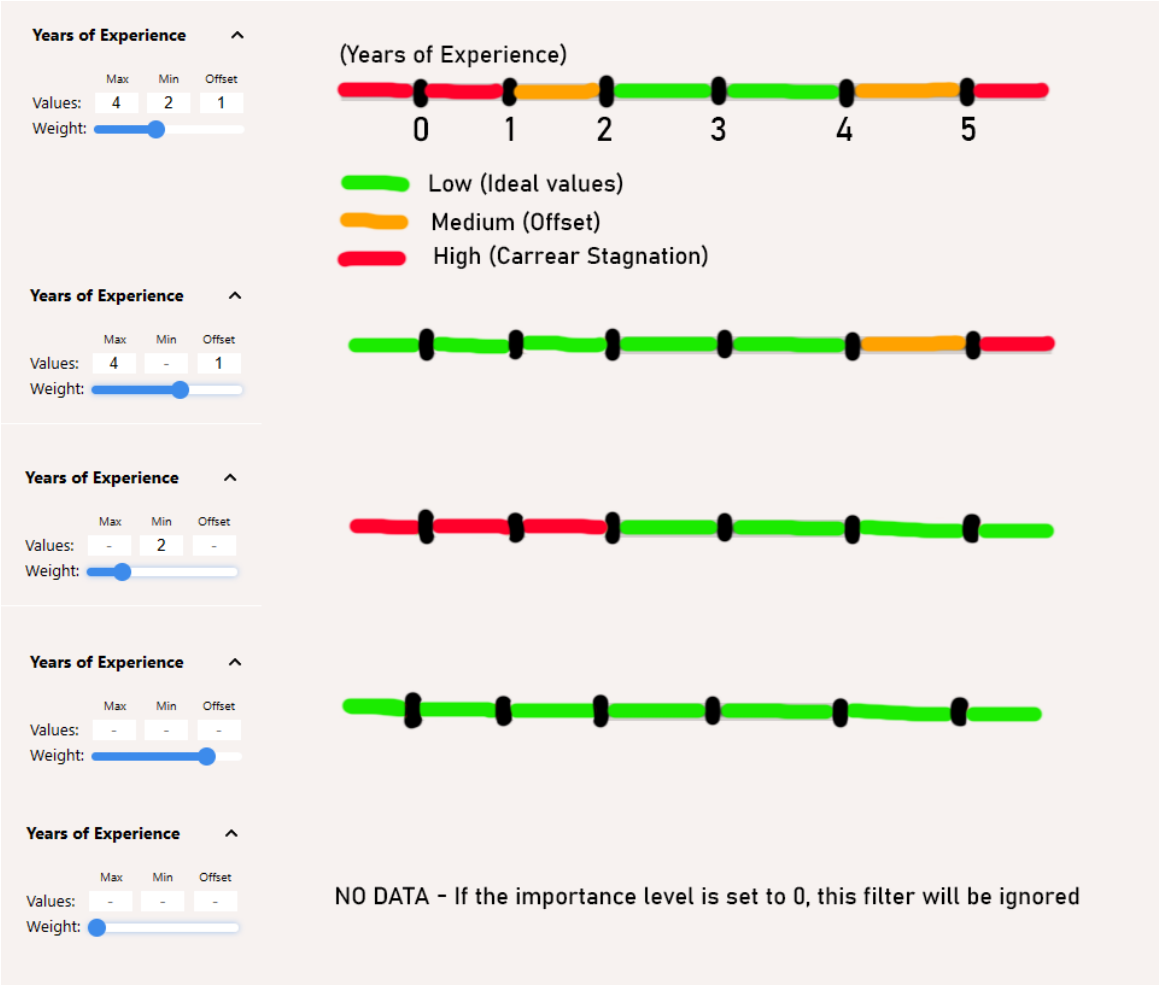



Figure 5.7: Illustration of how the filters work depending on the user input.

In the image, there are five examples counting from top to bottom, symbolized by the user parameters on the left and by the visual metric on the right side. **Each output is symbolized by a color**, red represents high, amber represents medium, and green represents low stagnation levels.
For instance, utilizing the "years of expertise" metric, being this the first example, the user set the maximum value to 4, the minimum to 2, and the offset to 1. This configuration implies that if an employee has between 2 and 4 years of expertise, their stagnation level will be **low**. However, if the experience falls between 1 and 2 or 4 and 5 years, it's categorized as **medium**

due to the offset. Values outside these ranges, falling into the red area, indicate a **high level of stagnation**.
It's worth mentioning that none of the parameters are mandatory. Leaving a parameter blank results in infinite values, as demonstrated in the second example, where the minimum level wasn't set. Thus, any value below the maximum (4 in this case) is considered a low level of stagnation. Similarly, leaving the offset blank restricts the output to only high or low, as depicted in the third example. In the fourth, all parameters were left empty, meaning there are no limits, resulting in a perpetually low stagnation level.
Each parameter can have varying levels of importance, which is why a weight parameter was implemented. The more the slider is adjusted, the more significant that parameter's impact on the final output, once all filters are combined.
Moreover, this system is designed to be highly scalable. If the company decides to add another parameter for evaluating stagnation, it simply needs to be added to the database. This new parameter will then appear as a filter and contribute to the final assessment.
Now that the functionality of the filters has been elucidated, the next step is to correlate all parameters, taking into account the importance level assigned to each one. The objective is to aggregate these diverse metrics into a single value that effectively describes the career stagnation level of an employee. This aggregation process involves weighting each parameter based on its significance, combining their individual impacts into a comprehensive assessment. The first approach was the creation of an equation like the following:

$$\text{Final Value} = \sum_{i=1}^{n} (\text{Atributte}_i \times \text{Weight}_i)$$

In this equation, $n$ represents the total number of parameters, $\text{Attribute}_i$ represents the value of attributes such as high, medium or low, and $\text{Weight}_i$ represents the weight assigned to this attribute. The summation symbol $\sum$ denotes that we sum up the products of each parameter value and its corresponding weight from $i = 1$ to $n$.
The challenge with this method lies in the non-numeric nature of the attribute values. To incorporate them into formulas, we must convert descriptors like high, medium, and low into their corresponding numerical representations.
This task becomes simpler when dealing with only two values, such as high and low. For instance, we could assign numerical values like 1 and 2. By iteratively adding or subtracting until convergence, we can determine the closest value. However, with three parameters, this approach becomes tricky as it may lead to inaccuracies.

Another idea would be to isolate each different tribute and make an average calculation. In this case, the highest among these values determines the classification. To provide further detail, let's illustrate with real values. There will be defined five different parameters, already

processed by the filter system, with their respective weights and classifications in the table below:

| Attribute Name | Type | Weight |
|---|---|---|
| Years of experience | High | 0.4 |
| Total work hours | Low | 0.6 |
| Nº of company activities | Medium | 0.3 |
| Number of projects | Low | 0.5 |
| Time since last promotion | High | 0.2 |

Listing 5.8: Example values to illustrate how the algorithm works.

In this method, we separate the diferent types and consequently, the attributes would be always the same values, so they can be replaced by 1 in the formula. the calculations would be something like shown:

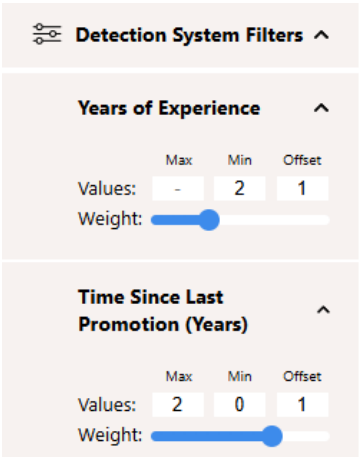
- High = $1 \times 0.4 + 1 \times 0.2 = 0.6$;
- Medium = $1 \times 0.3 = 0.3$;
- Low = $1 \times 0.6 + 1 \times 0.5 = 1.1$.

In this scenario, the resulting value would be categorized as **low**. Nonetheless, this method might lead to inaccuracies as it treats each word as distinct levels, overlooking the nuanced interpretation of medium as values between the other two extremes. Despite this limitation, it was deemed the most reliable approach and thus was selected.
To demonstrate how different input values may affect the career stagnation results, here is the data of five employees followed by three images where the input values change as follows:

| Employee ID | Name | ID Number | HR Status | Employment Type | Years Since Last Promotion | Years of Experience |
|---|---|---|---|---|---|---|
| 1 | John Doe | CC123123 | Active | Full-time | 2 | 5 |
| 2 | Jane Smith | CC456321 | Inactive | Part-time | 1 | 3 |
| 3 | Bob Johnson | CC789423 | Active | Full-time | 3 | 4 |
| 4 | Alice Williams | CC321432 | Active | Contractor | 4 | 2 |
| 5 | Charlie Davis | CC654342 | Active | Full-time | 5 | 6 |

Table 5.7: Example of employee data.

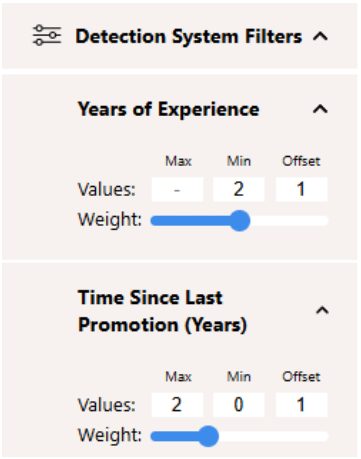
Based on this data, the system's output for career stagnation can vary significantly between individuals. The first image illustrates how the input values for the years of experience and years since the last promotion fields impact the final result. The minimum value for years of experience is set to 2, with no specified maximum, and an offset of 1. For the time since the last promotion parameter, the minimum is set to 0, the maximum to 2, and the offset to 1. Additionally, the weight assigned to the years since the last promotion is slightly higher than that the years of experience.




| #ID | Name | Citizen Card | Status | Type | Career Stagnation Detection |
|---|---|---|---|---|---|
| #1 | John Doe | CC123123 | Active | Full-time | Medium |
| #2 | Jane Smith | CC456321 | Inactive | Part-time | Low |
| #3 | Bob Johnson | CC789423 | Active | Full-time | High |
| #4 | Alice Williams | CC321432 | Active | Contractor | High |
| #5 | Charlie Davis | CC654342 | Active | Full-time | High |

Figure 5.8: Demonstration of the career stagnation detection parameter variation depending on the input.

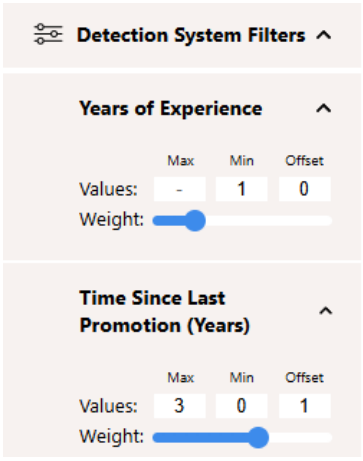
The second image has the same parameters set as the first one, but there is a slight change in the weight field. As we can observe, the final values are completely altered.




| #ID | Name | Citizen Card | Status | Type | Career Stagnation Detection |
|---|---|---|---|---|---|
| #1 | John Doe | CC123123 | Active | Full-time | Low |
| #2 | Jane Smith | CC456321 | Inactive | Part-time | Low |
| #3 | Bob Johnson | CC789423 | Active | Full-time | Low |
| #4 | Alice Williams | CC321432 | Active | Contractor | Medium |
| #5 | Charlie Davis | CC654342 | Active | Full-time | Low |

Figure 5.9: Demonstration of the career stagnation detection parameter variation depending on the input with a change in the weight fields.

The third and last image is used to compare the final values by maintains the same weight as the first image, but there is a change in the input values. In the years of experience field, the minimum has been set to 1 and the offset set to 0, while in the field of the time since last promotion, only the maximum has been changed from 2 to 3.




| #ID | Name | Citizen Card | Status | Type | Career Stagnation Detection |
|---|---|---|---|---|---|
| #1 | John Doe | CC123123 | Active | Full-time | Low |
| #2 | Jane Smith | CC456321 | Inactive | Part-time | Low |
| #3 | Bob Johnson | CC789423 | Active | Full-time | Medium |
| #4 | Alice Williams | CC321432 | Active | Contractor | High |
| #5 | Charlie Davis | CC654342 | Active | Full-time | High |

Figure 5.10: Demonstration of the career stagnation detection parameter variation depending on the input with a change in the maximum, minimum and offset fields.

In summary, the system demonstrates flexibility in calibrating career stagnation results across a defined range of input parameters. It's noteworthy that as more detection system filters are integrated into the project, the transition in the final result becomes less abrupt with slight adjustments in the input parameters. Following the planning and development of the entire system, including its logical components such as the previously mentioned algorithms, **the final step involves conducting tests to ensure that the system functions as intended**.

### 5.2.6 Results, Testing and Quality Assurance

In the final stages of this internship, comprehensive tests were conducted to ensure the project met all specified requirements. The primary aim was to develop a system capable of automatically detecting career stagnation.
Despite careful planning, some challenges proved difficult. Not all steps outlined in the initial project plan were completed. This was partly due to the amount of time required to complete certain tasks, and partly due to new instructions received from the company side. Additionally, some tasks began to lose relevance during the implementation phase. These adjustments impacted the original timeline and scope.
The following table provides a detailed overview of the tasks planned at the beginning of the project. It indicates whether each goal was achieved and, if not, explains the reasons behind any deviations.

| Task | Completed | Challenges and explanations |
|---|---|---|
| Define project scope and objectives. | ✓ | The project main goals were successfully defined at the beginning of the internship, as outlined in the introduction of this document. |

| | | |
|---|---|---|
| Research and technology selection. | ✓ | All the technologies selected during the planning phase of this project were utilized correctly as intended. However, during the development phase, some additional technologies could have been considered to enhance efficiency and accuracy of the project implementation. |
| Database design and setup. | ✓ | The full setup of the database was completed in a later phase than previously planned. However, it was executed successfully according to all requirements. |
| Project Documentation and Planning. | ✓ | The objective was to create comprehensive project documentation and a well-structured plan. However, this task proved to be much more time-consuming than originally planned, significantly impacting the project's progress. It included the full development of this document, which was only completed after the internship concluded. |
| API development for data input. | ✓ | This task was successfully completed within the planned timeframe. The implementation of the MSSQL API was straightforward and efficient. |
| Database Integration and Schema Implementation. | ✓ | The integration was successfully executed, although there were some challenges during the implementation of the *Docker* container due to limited experience with this technology, which resulted in some delays. |
| Historical Data Storage and Tracking. | × | During the implementation phase, this task was discussed with the company superiors and deemed unnecessary due to its reduced importance. This decision allowed for more time to be allocated to the development of other critical tasks. |
| Forecasting Module Development. | ✓ | Successfully developed and implemented. The decision to not utilize machine learning, as outlined in the document, diverged from the initial planning. However, the final result meets the requirements. |

| | | |
|---|---|---|
| User authentication and authorization. | × | This task was not completed within the allocated time frame due to its significant time requirements and relatively lower priority in this project, given that it was not the main objective. Additionally, it required the implementation of new tools, requirements, and specifications, which added complexity and extended the development timeline. The decision to discard this task from the project was made early on, following discussions and considerations of project priorities. |
| Employee data display and filtering. | ✓ | This task was completed successfully and even exceeded expectations. The final product had a high visual appeal and functioned as planned. It was vital, as it enabled users to utilize the application to its full efficiency, enhancing overall usability and experience. |
| Historical data visualization. | × | It was determined during the implementation phase that this task did not align with the project's objectives. Therefore, it was deemed unnecessary to allocate resources to this task, allowing for a more focused approach on other essential aspects of the project. |
| Alerting mechanism and notification integration. | × | Despite being included in the system requirements and appearing as an option in the interface, the logic part of its implementation was ultimately omitted due to time constraints. |
| Unit testing for backend and frontend components. | ✓ | Each component was separately tested during the implementation phase to ensure proper functionality and again during the final phase. |
| Integration testing. | ✓ | All testing was conducted correctly by verifying the successful transmission of data between components depending on user actions. |
| Performance Testing and Optimization. | × | While specific performance testing was not conducted, efforts were made during the implementation phase to optimize the code and ensure the system operated as efficiently and quickly as possible. |

| | | |
|---|---|---|
| Security testing and vulnerability assessment. | × | Similar to the previous point, while specific security testing was not conducted, safety measures were implemented throughout the development process. The code was developed following the best practices and guidelines provided by the documentation of the different technologies used, ensuring the final product's security. |

Table 5.8: Overview of planned tasks and their completion status, along with descriptions of challenges and justifications.

Overall, the majority of planned tasks were completed successfully, particularly the most significant ones essential to the project's functionality. The primary objectives were met despite some challenges and time constraints.
Proceeding to the discussion of the testing phase, the initial expectation was to use real data for the final testing and to evaluate the project's performance according to company standards. However, due to confidentiality concerns, this was not possible. Instead, a file was created to simulate company data, respecting the database format presented in the software engineering subchapter. The information used for the testing was random and entirely fictional, so some values may be unrealistic.
After manually inserting the data into the database, multiple tests were conducted to ensure compliance with the project's objectives and adherence to all functional and non-functional requirements. The following table presents the actions performed by users and their corresponding results, providing a better understanding of the system.

| **Test Description** | **Result** |
|---|---|
| Employee information display. | All requested employee details, including ID, name, citizen card, status, type, and the career stagnation detection parameter, are accurately displayed on the dashboard, indicating successful data transmission between the three components of the system. |
| Employee list filtering. | The filtering functionality accurately manipulates the employee list by name or ID. |
| Project information access. | Upon accessing the area where project information for each employee is displayed, including project ID, project name, date, and hours spent, no issues were encountered. |
| Email notification. | When attempting to send a notification, a warning message appears indicating that this feature is not fully implemented as specified in the table 5.8, but it correctly simulates the expected behavior of the system. |

| | |
|---|---|
| Customization of career stagnation filters. | The system responds accurately as users make changes to parameters like adjusting the maximum, minimum, offset, and the weight slider, ensuring intended functionality across all input fields. |
| Visual representation of filter impact. | Changes to filters result in accurate calculations and display correct values on the interface, providing users with valuable insights. |

Table 5.9: Tests conducted and description of outcomes.

The system demonstrated strong performance, meeting the established requirements and expectations. It serves as a valuable tool for identifying career stagnation within the organization, effectively fulfilling its objectives. With the conclusion of the testing phase and the completion of the project, the internship comes to an end.
To comprehensively document the progress achieved and tasks undertaken throughout the internship, the Appendix A provides a weekly log of the main activities conducted. There is a notable difference from the initial plan, as presented in the Gantt chart illustrated in the Figure 4.1.

## 5.3 Concluding Remarks

In summary, this chapter is regarded as the most significant section of this document, as it provides insight into the system's implementation in detail. It begins by explaining the system's behavior, presenting its requirements, architectural design, user interactions, and then outlining its intended functionalities. This analysis integrates an initial exploration of the system's three main components, which include the database, the server, and the client side.
After understanding the system's structure, the chapter proceeds to the discussion of its implementation, utilizing the tools previously selected. It offers a comprehensive overview of the development phase, specifying the steps taken to realize the system in accordance with all the requirements. This section focuses on the process of coding, execution, and testing. Furthermore, it describes the algorithm that constitute the career stagnation detection system, explaining its functionality and development process. Ultimately, the chapter concludes by presenting the results and comparing them with the initially planned tasks, offering a thorough assessment of the project's progress and outcomes.

# Chapter 6

# Conclusions

## 6.1 Main Conclusions

To draw a conclusion, this internship at *Softinsa* has been a valuable and enriching experience, providing a comprehensive overview of the company's operations. The project, that was made during this internship, focused on creating a system capable of identifying career stagnation among employees. Initially, a detailed description of this matter was provided, identifying the causes and emphasizing the importance of developing this system.
Following that, the exploration of scientific literature and previous works in related fields has laid a strong foundation, drawing insights from successful implementations and making use of best practices. It was also created a comprehensive plan outlining the activities intended, although there were certain challenges encountered that did not go according to it. Nonetheless, it served as a guiding framework facilitating effective project management.
The tools and technologies used included *Docker* for containerization, *GitHub* for collaboration and sharing of information and files, Microsoft SQL Server Management Studio for the database control, *Node.js* for the development of the server side, and *React* for the frontend crafting.
During the implementation phase, the main requirements were defined, along with the development of a system architecture and database schema that adhered to them. A straightforward algorithm was created to automatically detect career stagnation based on specific user inputs, fulfilling the project's primary objective. Subsequently, testing was conducted to ensure the system's effectiveness.
Although certain tasks required more time than initially anticipated, and others were removed or redefined during the main development phase, the final outcome corresponded to the expectations, resulting in a successful final product.

### 6.1.1 Personal Opinion

This subsection serves as my personal reflection on the internship experience, highlighting both positive and negative aspects based on my own observations.
Entering an internship was a completely new experience for me, as it meant putting myself for the first time in a professional work environment, adding to both my anticipation and

excitement. One of the aspects I appreciated the most was the freedom provided by the company. Unlike traditional work environments with strict daily schedules, there was no specific timeframe imposed on our work. Instead, we were given tasks with time limits, allowing me to approach the project in a more comfortable and relaxed manner without having to adjust to a completely new schedule.
Additionally, the opportunity to participate in real world projects granted me a profound insight into this industry as a whole. I acquired essential knowledge and built up new skills that complemented my academic background, potentially benefiting my future endeavors. It was also the first time I could apply theories and concepts that I acquired from my academic studies, be it from my bachelor's or master's degree, in practical scenarios.
However, it's important to acknowledge that not everything is flawless. Keeping up with schedules proved to be difficult at times, and the numerous changes during the project development phase often deviated from the initial plan. Nevertheless, such occurrences are entirely common and unavoidable, and the final results remained unaffected.
I also found it troublesome to work with outdated equipment, as it significantly interfered with the development process due to its slow and inefficient performance. However, I understand the necessity for such measures, as they contribute to the safety and control of activities. Additionally, as this was a curricular internship, I sometimes felt that my project lacked significance to the company, serving merely as an evaluation of my skills. Yet, upon reflection, it makes sense, as it would have been imprudent to grant access to sensitive information or assign critical tasks without having proper skills in this scenario.
Despite all the challenges, this internship provided to be an exceptional learning opportunity that expanded my knowledge and deepened my appreciation for the field. I'm immensely grateful for this experience, and I hope to apply this newfound knowledge as I move forward.

## 6.2 Future Work

Although the project has been fully developed and the internship has concluded, there are still areas for improvement to make this project applicable in a real world scenario.
This includes the realization of tasks that were not performed during its implementation but were documented in the planning phase. For instance, establishing a comprehensive system for authentication and authorization would be essential, providing a secure login area for HR professionals to access the application.
Additionally, the system should have the capability to receive files from employees' reports to further expand the information available, thereby significantly improving the accuracy of the career stagnation detection system. To achieve this, it would also be necessary to ensure proper storage by implementing a historical data storage system.
It is also essential to conduct security tests, since the provided information will be confidential.
Lastly, the most crucial aspect to be done is the optimization of the algorithm used to detect career stagnation. For instance, by using the data from reports and applying it to machine learning algorithms, we could create a fully automatic and efficient system that could signif-

icantly enhance the accuracy of the final product.

# Bibliography

[1] University of beira interior. Last access at 5th June 2024. [Online]. Available: https://www.ubi.pt/ 1

[2] Softinsa, advanced software engineering, ltd. Last access at 5th June 2024. [Online]. Available: https://www.softinsa.pt/pt/ 1, 5

[3] Anne Chow. (2024) What to do when your career feels like it's stagnating. [Online]. Available: https://fortune.com/2023/02/23/what-to-do-when-your-career-feels-stagnating-jobs-mental-health-anne-chow/ 1

[4] Viewnext. Last access at 5th June 2024. [Online]. Available: https://www.viewnext.com/ 5

[5] International business machines corporation. Last access at 5th June 2024. [Online]. Available: https://www.ibm.com/ 5

[6] President of the portuguese republic biography. Last access at 5th June 2024. [Online]. Available: https://www.presidencia.pt/en/president-of-the-republic/the-president/ 5

[7] Softinsa's services. Last access at 5th June 2024. [Online]. Available: https://www.softinsa.pt/pt/servicos/ 7

[8] Lenovo thinkpad l460. Last access at 5th June 2024. [Online]. Available: https://www.lenovo.com/pt/pt/p/laptops/thinkpad/thinkpadl/thinkpad-l460/22tp2tbl460 9

[9] Mrugank Patel. (2023) Breaking free: Overcoming stagnation and achieving career growth. Last access at 5th June 2024. [Online]. Available: https://www.linkedin.com/pulse/breaking-free-overcoming-stagnation-achieving-career-growth-patel 11

[10] Joanna Zambas. (2023) 10 signs of career stagnation and how to overcome it. [Online]. Available: https://www.careeraddict.com/career-stagnation 11

[11] J. Zhao, M. Karimzadeh, L. S. Snyder, C. Surakitbanharn, Z. C. Qian, and D. S. Ebert, "Metricsvis: A visual analytics system for evaluating employee performance in public safety agencies," *IEEE Transactions on Visualization and Computer Graphics*, vol. 26, no. 1, pp. 1193–1203, 2020. 12

[12] K. Şahinbaş, "Employee promotion prediction by using machine learning algorithms for imbalanced dataset," in *2022 2nd International Conference on Computing and Machine Intelligence (ICMI)*, 2022, pp. 1–5. 15

[13] Support vector machines. Last access at 6th June 2024. [Online]. Available: https://scikit-learn.org/stable/modules/svm.html 15

[14] Neural networks. Last access at 6th June 2024. [Online]. Available: https://www.ibm.com/topics/neural-networks 15

[15] Random forest. Last access at 6th June 2024. [Online]. Available: https://www.ibm.com/topics/random-forest 15

[16] Kaggle. Last access at 6th June 2024. [Online]. Available: https://www.kaggle.com/ 15

[17] C. Goutte and E. Gaussier, "A probabilistic interpretation of precision, recall and f-score, with implication for evaluation," vol. 3408, 04 2005, pp. 345–359. 16

[18] S. M. Kumar and M. Belwal, "Performance dashboard: Cutting-edge business intelligence and data visualization," in *2017 International Conference On Smart Technologies For Smart Nation (SmartTechCon)*, 2017, pp. 1201–1207. 17

[19] L. Wu, G. Barash, and C. Bartolini, "A service-oriented architecture for business intelligence," in *IEEE International Conference on Service-Oriented Computing and Applications (SOCA '07)*, 2007, pp. 279–285. 17

[20] S. Ashraf and S. A. Khan, "Visualizations-based analysis of telco data for business intelligence," in *2015 6th IEEE International Conference on Software Engineering and Service Science (ICSESS)*, 2015, pp. 242–246. 17

[21] Specific regulations for the second cycle of studies in computer science and engineering at the university of beira interior. Last access at 6th June 2024. [Online]. Available: https://www.di.ubi.pt/Cursos/mei 21

[22] Docker overview. Last access at 6th June 2024. [Online]. Available: https://docs.docker.com/get-started/overview/ 25

[23] Git. Last access at 6th June 2024. [Online]. Available: https://git-scm.com/ 25

[24] Github. Last access at 6th June 2024. [Online]. Available: https://github.com/ 26

[25] What is sql server management studio. Last access at 6th June 2024. [Online]. Available: https://learn.microsoft.com/en-us/sql/ssms/sql-server-management-studio-ssms?view=sql-server-ver16 26

[26] About node.js. Last access at 6th June 2024. [Online]. Available: https://nodejs.org/en/about 26

[27] React - a javascript library for building user interfaces. Last access at 6th June 2024. [Online]. Available: https://legacy.reactjs.org/ 26

[28] I. Sommerville, *Software engineering*. Department, 501 Boylston Street, Suite 900, Boston, Massachusetts 02116: Pearson Education, 2011. 29

[29] Importance of software engineering: Key reasons. Last access at 7th June 2024. [Online]. Available: https://www.knowledgehut.com/blog/web-development/importance-of-software-engineering 29

[30] D. Dave and V. Anu, "Identifying functional and non-functional software requirements from user app reviews," in *2022 IEEE International IOT, Electronics and Mechatronics Conference (IEMTRONICS)*, 2022, pp. 1–6. 30

[31] What is database architecture? Last access at 7th June 2024. [Online]. Available: https://www.mongodb.com/resources/basics/databases/database-architecture 31

[32] What is an er diagram? Last access at 7th June 2024. [Online]. Available: https://www.lucidchart.com/pages/er-diagrams 31

[33] J. Nicacio and F. Petrillo, "An approach to build consistent software architecture diagrams using devops system descriptors," in *Proceedings of the 25th International Conference on Model Driven Engineering Languages and Systems: Companion Proceedings*, ser. MODELS '22. ACM, Oct. 2022. [Online]. Available: http://dx.doi.org/10.1145/3550356.3561567 33

[34] What is sequence diagram? Last access at 7th June 2024. [Online]. Available: https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-sequence-diagram/ 36

[35] Overview of docker desktop. Last access at 7th June 2024. [Online]. Available: https://docs.docker.com/desktop/ 38

[36] G. Kaur and R. G. Tiwari, "Comparison and analysis of popular frontend frameworks and libraries: An evaluation of parameters for frontend web development," in *2023 4th International Conference on Electronics and Sustainable Communication Systems (ICESC)*, 2023, pp. 1067–1073. 44

# Appendix A

# Weekly Log of Main Activities Conducted During the Internship

| Time Interval | Activities |
|---|---|
| Dec 20, 2023 - Dec 24, 2023 | Initial visit to the establishment including introductions to colleagues, acquisition of essential equipment, and finalization of preparatory documentation. |
| Dec 25, 2023 - Dec 31, 2023 | Short Christmas break. |
| Jan 1, 2024 - Jan 7, 2024 | Definition of the project goals and objectives, acquisition of insight into company requirements, and establishment of a GitHub repository. |
| Jan 8, 2024 - Jan 14, 2024 | Research on relevant information such as career stagnation detection methods, initial technology stack selection, and creation of the document that will present all the work that will be carried out during the internship. |
| Jan 15, 2024 - Jan 21, 2024 | Complementation of the research by studying multiple articles and summarizing their significance. |
| Jan 22, 2024 - Jan 28, 2024 | Setup of the development environment. Requirements gathering and creation of the initial project plan. |
| Jan 29, 2024 - Feb 4, 2024 | Design of database schema. Development of database schema, setting up database server, and creation of initial API endpoints. |
| Feb 5, 2024 - Feb 11, 2024 | Initial setup of the server environment involving configuration and deployment of the necessary components, followed by establishment of connections to the database. |
| Feb 12, 2024 - Feb 18, 2024 | Initiation of the design process for the user interface, implying the creation of mockups. |
| Feb 19, 2024 - Feb 25, 2024 | Translation of the mockups into a user interface and development of its connection to the server. Testing of the data flow between the three components, which include the database, the server, and the client side. |
| Feb 26, 2024 - Mar 3, 2024 | Development of the employee private project information display menu, testing data integration between the database and user interface, and optimization of backend performance, including bug fixes. |
| Mar 4, 2024 - Mar 10, 2024 | Development of the user management features, including information filtering. |

| | |
|---|---|
| Mar 11, 2024 - Mar 17, 2024 | Initiation of the development process for the career stagnation detection algorithm. |
| Mar 18, 2024 - Mar 24, 2024 | Creation of multiple adjustable filters for the user, facilitating the input of data into the developed algorithm. |
| Mar 25, 2024 - Mar 31, 2024 | Integration of the career stagnation detection system with the user interface, testing the algorithm, testing the final product, comparing results with initial requirements, and preparing the final project report. |
| Apr 1, 2024 - Apr 5, 2024 | Bug fixes, finalization of the project, return of used equipment to the company, and conclusion of the internship. |

Table A.1: Weekly log of the internship activities from December 20, 2023, to April 5, 2024