



UNIVERSIDADE  
LUSÓFONA

# Migração de Softwares *Legacy*

## Trabalho Final de curso

Relatório Final

Joana Gonçalves | a22107603

Guilherme Simão | a22106142

Professor Pedro Serra

Trabalho Final de Curso | LEI | Junho 2024

## **Direitos de cópia**

*Migração de Softwares Legacy*, Copyright de *Joana Gonçalves (a22107603)* e *Guilherme Simão (a22106142)*, Universidade Lusófona.

A Escola de Comunicação, Arquitectura, Artes e Tecnologias da Informação (ECATI) e a Universidade Lusófona (UL) têm o direito, perpétuo e sem limites geográficos, de arquivar e publicar esta dissertação através de exemplares impressos reproduzidos em papel ou de forma digital, ou por qualquer outro meio conhecido ou que venha a ser inventado, e de a divulgar através de repositórios científicos e de admitir a sua cópia e distribuição com objectivos educacionais ou de investigação, não comerciais, desde que seja dado crédito ao autor e editor.

---

## Resumo

O nosso Trabalho de Final de Curso (TFC)[1] foca-se na migração de um software *Legacy* do Grupo Barraqueiro. Trata-se de um conjunto de ferramentas internas que foram desenvolvidas à medida que as necessidades foram surgindo ao longo dos últimos 20 anos e que enfrenta desafios. Este sistema é composto por um middleware, designado de Martelo, e uma aplicação com o nome "Galp", ambos desenvolvidos em linguagens e tecnologias desatualizadas, como o Visual Basic 6 (Middleware) e o Clipper (Galp), o que acarreta problemas de compatibilidade, escalabilidade e segurança.

A necessidade de um middleware surgiu à medida que o Grupo Barraqueiro cresceu. A aplicação Galp foi desenvolvida numa época em que o grupo estava em expansão, e à medida que novos formatos e sistemas de bombas de combustível surgiram, tornou-se impossível adaptar a aplicação devido à sua complexidade. Isso resultou na necessidade de criar um middleware para unificar os diferentes formatos de dados das bombas num único formato, permitindo ao módulo Galp processar os dados de forma consistente, nascendo assim o middleware Martelo.

Operando como um leitor e escritor de ficheiros, o Martelo desempenha um papel vital lendo os dados recebidos das bombas, processando-os e reescrevendo-os num formato de base de dados dbase. Esta abordagem não apenas padroniza os dados para fácil interpretação pela aplicação Galp, mas também os organiza de forma a permitir uma análise futura eficaz. Além disso, ao ser desenvolvido em Visual Basic 6 (VB6), o Martelo demonstra a sua adaptabilidade ao ambiente Windows, integrando-se perfeitamente às operações existentes do Grupo Barraqueiro.

O módulo Galp é o núcleo da análise e gestão inteligente dos dados provenientes das bombas de combustível. Ao receber os dados processados e padronizados pelo Martelo, o Galp entra em ação, produzindo uma variedade de relatórios e insights valiosos para otimizar a operação da frota de veículos. Desde gráficos de consumo médio de combustível por dia até previsões de possíveis avarias com base no consumo médio dos veículos, o Galp fornece uma visão abrangente e detalhada do desempenho da frota.

Além disso, a integração do Galp com diferentes bases de dados, incluindo AS400 e SQL Server, demonstra a sua capacidade de operar num ambiente diversificado, garantindo a acessibilidade e a compatibilidade dos dados para diferentes sistemas dentro do Grupo Barraqueiro.

## Abstract

Our Final Year Project (FYP) [1] focuses on migrating a *legacy* software of Grupo Barraqueiro. It consists of a set of internal tools that were developed as needs arose over the last 20 years and now face challenges. This system consists of middleware, called Martelo, and an application named "Galp", both developed in outdated languages and technologies such as Visual Basic 6 (Middleware) and Clipper (Galp), leading to issues with compatibility, scalability, and security.

The need for middleware emerged as Grupo Barraqueiro grew. The Galp application was developed during a period of expansion for the group, and as new formats and systems of fuel pumps emerged, it became impossible to adapt the application due to its complexity. This led to the creation of middleware to unify the different data formats of the pumps into a single format, allowing the Galp module to process the data consistently, thus giving birth to the Martelo middleware.

Operating as a file reader and writer, Martelo plays a vital role by reading the incoming data from the pumps, processing it, and rewriting it in a dbase database format. This approach not only standardizes the data for easy interpretation by the Galp application but also organizes it to allow for effective future analysis. Additionally, being developed in Visual Basic 6 (VB6), Martelo demonstrates its adaptability to the Windows environment, seamlessly integrating with the existing operations of Grupo Barraqueiro.

The Galp module is the core of intelligent data analysis and management from the fuel pumps. Upon receiving the processed and standardized data from Martelo, Galp springs into action, producing a variety of reports and valuable insights to optimize the vehicle fleet operation. From graphs of average fuel consumption per day to predictions of potential malfunctions based on average vehicle consumption, Galp provides a comprehensive and detailed view of the fleet's performance.

Furthermore, the integration of Galp with different databases, including AS400 and SQL Server, demonstrates its ability to operate in a diverse environment, ensuring data accessibility and compatibility for different systems within Grupo Barraqueiro.

---

# Índice

<b>Resumo.....</b>	<b>3</b>
<b>Abstract.....</b>	<b>4</b>
<b>Índice.....</b>	<b>5</b>
<b>Lista de Figuras.....</b>	<b>7</b>
<b>Lista de Tabelas.....</b>	<b>8</b>
<b>1. Identificação do Problema.....</b>	<b>9</b>
1.1. O middleware (Martelo).....	10
1.2. O módulo Galp.....	10
1.3. Objetivos das Mudanças.....	12
1.4. Comparativo.....	12
<b>2. Viabilidade e Pertinência.....</b>	<b>13</b>
2.1. Migração para C#, Plataforma .NET e ReactJS.....	13
2.2. Divisão do novo Sistema em 3 módulos.....	13
2.3. Novo Middleware.....	13
2.4. Centralização do Sistema.....	14
2.5. Objetivos das Mudanças.....	14
2.6. Futuro Aplicacional.....	14
2.7. Proposta de Mudanças.....	14
<b>3. Benchmarking.....</b>	<b>16</b>
Tabela de comparação entre o software antigo com o software atual.....	16
<b>4. Engenharia.....</b>	<b>19</b>
4.1. Levantamento e análise dos Requisitos.....	19
4.1.1. Objetivos do Sistema:.....	19
4.1.2. Requisitos Funcionais:.....	19
4.1.3. Requisitos Não Funcionais:.....	21
4.1.4. Restrições e Limitações:.....	23
4.1.5. Stakeholders:.....	23
4.1.6. Entregáveis:.....	23
4.2. Diagramas de Casos de Uso.....	24
4.3. Diagramas de Atividades.....	26
4.4. Modelos relevantes.....	27
4.5. Estrutura.....	31
4.6. Mockups.....	32
4.7. Avaliação de Concretização dos Requisitos.....	36
4.8. Alterações Realizadas.....	36
4.9. Avaliação de Valor Obtido.....	36
<b>5. Solução Desenvolvida.....</b>	<b>37</b>
5.1. Introdução.....	37
5.2. Arquitetura.....	38

5.2.2.1. Novo Middleware.....	40
5.2.2.2. Camada de API's.....	40
5.2.2.3. Novo Frontend.....	40
5.3. Tecnologia e Ferramentas Utilizadas.....	40
5.4. Implementação.....	41
5.5. Solução em desenvolvimento.....	41
5.5.1. Ponto de Situação.....	41
5.6. Resultado final.....	44
5.7. Abrangência.....	48
<b>6. Plano de testes e validação.....</b>	<b>49</b>
6.1. Plano de Testes para Validação Prática e Operacional.....	49
6.2. Guião de testes detalhado.....	49
6.3. Validação da Arquitetura Proposta:.....	50
6.4. Test Case para Validação da Solução:.....	51
6.5. Proposta de Roadmap para Deployment em Ambiente de Testes/Produção:.....	52
6.6. Resultados dos testes.....	53
Validação da Arquitetura Proposta.....	55
Proposta de Roadmap para Deployment em Ambiente de Testes/Produção.....	56
6.7. Conclusão.....	57
<b>7. Método e Planeamento.....</b>	<b>58</b>
7.1. Parte 1: Desenvolvimento Inicial.....	58
7.2. Parte 2: Desenvolvimento Avançado.....	58
<b>8. Resultados.....</b>	<b>68</b>
8.1. Descrição detalhada dos resultados.....	68
8.2. Outputs e outcomes.....	68
8.3. Análise do cumprimento dos critérios de sucesso.....	69
8.4. Revisões realizadas ao longo do desenvolvimento do TFC.....	70
8.5. Resultados dos test cases definidos nos relatórios anteriores.....	70
<b>9. Conclusão e trabalhos futuros.....</b>	<b>72</b>
Trabalhos Futuros.....	72
Agradecimentos.....	73
<b>Bibliografia.....</b>	<b>74</b>
<b>Questionário.....</b>	<b>75</b>
<b>Anexos.....</b>	<b>78</b>
<b>Glossário.....</b>	<b>79</b>

---

## Lista de Figuras

Figura 1 – Arquitetura Atual do Sistema.....	9
Figura 9 - Interface do Software antigo.....	18
Figura 3 - Diagrama de Casos de Uso.....	24
Figura 4 - Diagrama de Atividade.....	26
Figura 5 - Modelo de classes.....	27
Figura 6 - Modelo relação-entidade.....	29
Figura 7 - Estrutura em árvore do Software.....	31
Figura 8 - Interface proposta do ecrã principal para o novo software.....	32
Figura 9 - Esquema da Base de Dados.....	34
Figura 10 - Representação da Arquitetura ReactJS.....	38
Figura 11 - Representação da Arquitetura.....	39
Figura 12 - Interface desenvolvida (ecrã de dashboard).....	42
Figura 13 - Interface desenvolvida (ecrã de criação de elementos).....	42
Figura 14 - Interface desenvolvida (ecrã de visualização de elementos).....	43
Figura 15 - Exemplo de editar.....	45
Figura 16 - Exemplo Auto-tradução de chaves estrangeiras.....	45
Figura 17 - Feedback ao utilizador.....	45
Figura 18 - Validação ao apagar.....	45
Figura 19 - Página do Middleware.....	46
Figura 20 - Exemplo gráficos.....	46
Figura 21- Menu melhorado.....	47
Figura 22 - Processamento do Middleware.....	48
Figura 23 - Planeamento do Projeto PT1.....	60
Figura 24 - Planeamento do Projeto PT2.....	61
Figura 25 - Planeamento do Projeto PT3.....	61
Figura 26 - Planeamento do Projeto PT4.....	62
Figura 27 - Planeamento do Projeto PT5.....	63
Figura 28 - Planeamento do Projeto PT6.....	64
Figura 29 - Planeamento do Projeto PT6.....	65
Figura 30 - Planeamento do Projeto PT7.....	65

## **Lista de Tabelas**

Tabela de comparação entre o software antigo com o software futuro.....	14
---	----



# 1. Identificação do Problema

Fomos contactados pela empresa Barraqueiro com o objetivo de resolver os problemas no seu sistema em vias de se tornar obsoleto, doravante referido como *Legacy*, chamado Galp.

Após termos conhecimento da infraestrutura do Sistema Galp, constatamos que os nomes dos módulos/aplicações apresentam uma redundância que pode induzir a equívocos. Diante deste cenário, propomos a alteração do nome da aplicação, visando à clareza e precisão na identificação dos diferentes elementos do sistema.

Esta aplicação foi desenvolvida com o objectivo de recolher e processar informação sobre os abastecimentos de combustível dos veículos da empresa. Para cada abastecimento é registada informação associada ao veículo, motorista, quantidade de combustível. Essa informação é gravada numa base de dados e depois processada de forma a gerar relatórios com os dados recolhidos. Atualmente, o sistema galp está dividido em dois módulos: o middleware (Martelo) e o módulo "Galp". como apresentado na figura 1. O Martelo, atual middleware, é responsável pela conversão dos dados das bombas, que por norma variam de bomba para bomba, convertendo a informação para um ficheiro. O módulo Galp é responsável pela leitura dos ficheiros analisando os dados para report dos mesmos, acabando por guardá-los numa base de dados.

Na figura 1, podemos observar o esquema da arquitetura atual do sistema *legacy*.

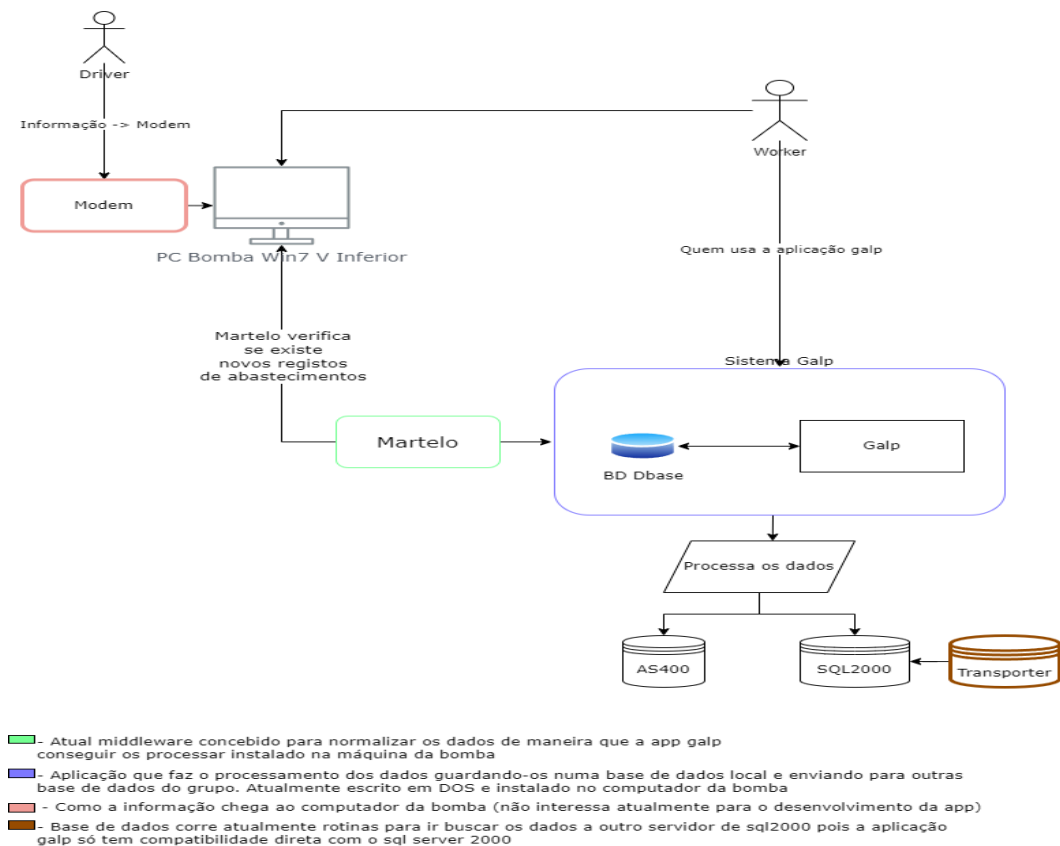


Figura 1 – Arquitetura Atual do Sistema

### **1.1.O middleware (Martelo)**

A necessidade de um middleware surgiu à medida que o grupo Barraqueiro cresceu. A aplicação "Galp" foi desenvolvida numa época em que o grupo estava em expansão, e à medida que novos formatos e sistemas de bombas de combustível surgiram, tornou-se impossível adaptar a aplicação devido à sua complexidade. Isso resultou na necessidade de criar um middleware para unificar os diferentes formatos de dados das bombas num único formato, permitindo ao módulo "Galp" processar os dados de forma consistente, nascendo assim o middleware Martelo. Este software opera lendo os arquivos na pasta de entrada dos dados recebidos das bombas, processando esses dados e, em seguida, escrevendo os arquivos em formato dBase na pasta dos dados processados. O formato dBase é um tipo de formato de ficheiro de base de dados que permite armazenar dados em formato tabular, semelhante a tabelas com linhas e colunas, [2]. Neste contexto, os dados processados são organizados e armazenados em ficheiros com a estrutura específica do formato dBase, o que facilita a sua interpretação pela aplicação "Galp" ou por outros sistemas que compreendam este formato. Isto ajuda a padronizar os dados e torná-los prontos para análise futura.

O middleware atua como um intermediário ou tradutor, convertendo e unificando os dados provenientes de diferentes bombas. Em outras palavras, ele recebe dados de várias fontes e processa-os de maneira padronizada para que a aplicação possa compreendê-los. Esta abordagem garante que as informações sejam consistentes e compatíveis para efeitos de análise e armazenamento.

É importante notar que o middleware foi desenvolvido em Visual Basic 6 (VB6), uma linguagem de programação utilizada para criar aplicações no ambiente Windows.

### **1.2. O módulo Galp**

O módulo "Galp" é o nosso principal foco neste projeto. Este software trata da análise dos dados vindo das bombas, e produzindo relatórios com informações tais como::

- gráficos de consumo médio de combustível das bombas por dia
- prever se o autocarro tem alguma avaria de acordo com o consumo médio dele.
- entre outras

Os dados analisados são guardados numa base de dados dBase e são enviados para 2 outras bases de dados do grupo, uma AS400[3] e outra de SQL Server 2014 que vai buscar os dados a uma base de dados SQL Server 2000 devido à incompatibilidade da app galp com o SQL Server 2014.

#### **1.2.1.Problema de segurança**

Como a aplicação Galp foi desenvolvida em Clipper [4], já não recebe manutenção, o que significa que não são efetuadas correções de bugs, atualizações de segurança ou melhorias funcionais. Isso torna-a vulnerável a problemas de segurança e incompatibilidades com sistemas operacionais mais modernos. Portanto, é essencial considerar a modernização ou substituição

dessa aplicação para corresponder às necessidades atuais da empresa Barraqueiro e garantir a segurança e a eficiência do sistema.

As aplicações são instaladas individualmente nos computadores de cada bomba, o que impede a atualização para versões mais recentes do Windows, pois as aplicações dependem deste sistema operativo. Isso cria um problema de segurança crítico, uma vez que os computadores utilizam versões do Windows que já não recebem atualizações, tornando-se vulneráveis a ameaças. Além disso, a aplicação Galp lida com uma base de dados no formato dbase, que é um formato de base de dados desatualizado e com inúmeras limitações.

#### **1.2.2.Problemas de complexidade**

A ausência de documentação e conhecimento sobre o formato de ficheiro usado pela aplicação "Galp" torna a manutenção e a compreensão do sistema mais difíceis. Isso pode levar a problemas de integração, dificuldades na resolução de problemas e aumentar a dependência de especialistas que possam entender o formato. A descentralização dos dados nos computadores das bombas é arriscada, uma vez que cria um único ponto de falha. Se um desses computadores falha, pode resultar na perda total de dados, o que é especialmente preocupante em uma configuração crítica como a gestão de bombas de combustível.

Novamente temos uma complexidade acrescida no que toca ao armazenamento de dados pois utiliza novamente uma base de dados local em dbase, para além disso a linguagem Clipper não consegue estabelecer uma comunicação direta com o SQL Server 2014, então o grupo criou um SQL Server 2000 como ponto intermédio no qual a app galp manda para lá os dados e após isso o SQL Server 2014 tem uma rotina que vai lá buscar os dados.

Para além da complexidade acrescida existe também um custo acrescido de manutenção e um custo acrescido de hardware e licenças pois caso fosse numa linguagem mais recente não haveria necessidade de ter uma base de dados adicional.

#### **1.2.3.Problemas de consistência**

Isto tudo numa janela de consola do windows, apesar de mais impressionante que seja, é uma interface bastante desatualizada, difícil de se usar, com bastantes limitações, e que dependendo da configuração do windows está sujeita a mudanças, o que gera ainda mais complexidade pois todos os windows tem que ser configurados de forma igual para que todas as interfaces sejam iguais.

#### **1.2.4.Problemas de escalabilidade**

Outro problema que a empresa enfrenta agora é a escalabilidade da aplicação, que se encontra estagnada sem possibilidade de melhoria devido à linguagem usada. Com as novas tecnologias e o surgimento da cloud, a empresa está a utilizar cada vez mais serviços cloud, como Grafana e práticas de DevOps, que fariam mais sentido para o sistema atual e proporcionariam uma melhor experiência de uso.

Com a migração das tecnologias mais antigas da empresa, surge agora a necessidade de modernizar e melhorar o sistema *legacy* para permitir a integração com os novos sistemas do grupo, reduzir o custo de manutenção, reduzir o custo de recursos computacionais e aumentar a segurança do grupo.

### **1.3. Objetivos das Mudanças**

Estes pontos visam resolver todos os problemas da empresa quanto a este sistema, o mais importante de todos é a segurança, atualmente a segurança é menosprezada pelas empresas até que algum evento aconteça, além disso, esta solução reduz bastante os custos de manutenção pois reduz bastante a complexidade do sistema mantendo a mente a centralização e a utilização de tecnologias correntes, a redução da complexidade irá trazer também à empresa custos de computação pois deixaram de precisar de diferentes servidores. Além destas reduções todas a empresa conseguirá agora, caso queira integrar a app nas outras apps da empresa pois serão desenvolvidas em .NET, ReactJS, e poderão recorrer a recursos de cloud.

### **1.4. Comparativo**

No início do projeto, nosso objetivo era resolver os problemas do sistema legado Galp, que estava se tornando obsoleto. Propusemos alterações significativas, incluindo a renomeação de módulos para melhorar a clareza e precisão na identificação dos elementos do sistema.

Em termos de objetivos, buscávamos melhorar a eficiência na gestão de abastecimentos de combustível. No final, conseguimos atingir esses objetivos, modernizando o sistema e integrando novos processos de coleta, processamento e geração de relatórios de dados.

Durante a execução do projeto, enfrentamos desafios significativos, especialmente na comunicação com a Barraqueiro e no cumprimento dos prazos devido à falta de informações detalhadas. Isso inicialmente afetou o andamento do projeto, mas conseguimos superar esses obstáculos ao longo do tempo.

Em termos de implementação, conseguimos dividir o sistema em módulos claros e funcionais: o Middleware (anteriormente conhecido como "Martelo") foi responsável por converter dados das bombas para um formato padronizado, enquanto o módulo Galp processou e armazenou os dados conforme esperado.

Os resultados alcançados foram significativos. Conseguimos melhorar a eficiência operacional do sistema Galp, proporcionando uma gestão mais eficaz e acima de tudo converter o sistema legacy para um sistema atual mais seguro com uma escalabilidade muito maior e que permite o desenvolvimento contínuo do mesmo.

## **2. Viabilidade e Pertinência**

Na era em constante evolução da tecnologia, a necessidade de adaptar e aprimorar as infraestruturas digitais torna-se essencial para o sucesso organizacional. No contexto específico do Grupo Barraqueiro, a escolha estratégica de migrar para a linguagem C# e adotar a plataforma .NET e ReactJS reflete uma abordagem proativa na modernização do seu sistema. Neste cenário, a proposta atual de transformação consiste na reformulação do sistema atual para um novo sistema resumido a três módulos: um middleware usado exclusivamente para tradução dos dados das bombas, uma API para comunicação com a base de dados, e o frontend desenvolvido em React para a interface. A implementação de um novo middleware e a centralização do sistema surgem como uma resposta coesa e inovadora aos desafios identificados no sistema *Legacy*. Este conjunto de mudanças não só tem como objetivo aumentar a eficiência operacional, mas também visa promover a estabilidade, segurança e aprimorar a experiência do utilizador, consolidando assim as bases para um futuro tecnológico mais robusto e resiliente para o Grupo Barraqueiro.

### **2.1. Migração para C#, Plataforma .NET e ReactJS**

**Viabilidade:** Alta. A escolha do C# e da plataforma .NET e ReactJS é coerente com a infraestrutura tecnológica existente no grupo Barraqueiro. Facilitará a integração com sistemas atuais e serviços em nuvem, proporcionando uma transição suave.

**Pertinência:** Muito pertinente. A mudança para uma linguagem mais moderna e uma plataforma consolidada contribuirá para a estabilidade e manutenção do sistema a longo prazo.

### **2.2. Divisão do novo Sistema em 3 módulos**

**Viabilidade:** Viável. A divisão em middleware, camadas de api e BFuel (app frontend) permite uma arquitetura mais modular e escalável. Cada componente pode ser desenvolvido e mantido independentemente.

**Pertinência:** Pertinente. A separação de responsabilidades simplifica o desenvolvimento, facilita a manutenção e melhora a experiência do utilizador com uma interface web mais moderna.

### **2.3. Novo Middleware**

**Viabilidade:** Alta. O novo middleware passa a ser uma rotina melhorando assim a experiência do utilizador, o mesmo deixa de comunicar diretamente com uma base de dados mas sim com a camada de api.

**Pertinência:** Pertinente. A centralização das comunicações no middleware melhora a eficiência operacional e contribui para a redução de custos.

## **2.4. Centralização do Sistema**

Viabilidade: Viável. A centralização do sistema em um servidor proporciona maior controle, segurança e facilita o acesso via Web APP. A centralização também aborda questões de redundância de dados e melhora a eficiência do gerenciamento de backups.

Pertinência: Muito pertinente. A centralização resolve problemas críticos de segurança, redundância e facilita a administração centralizada do sistema.

## **2.5. Objetivos das Mudanças**

Viabilidade: Alta. Os objetivos propostos, como melhoria da segurança, redução de custos de manutenção e centralização do sistema, são alcançáveis com as mudanças propostas.

Pertinência: Muito pertinente. A ênfase na segurança e na redução de custos alinha-se com as necessidades da empresa e garante uma solução mais robusta e eficiente.

## **2.6. Futuro Aplicacional**

Com a migração da aplicação para o ambiente .NET e ReactJS e considerando que as aplicações da empresa também são desenvolvidas nessa plataforma, surge a possibilidade de consolidar essas diversas aplicações em um único sistema integrado. Para obter feedback dos utilizadores da aplicação, desenvolvemos um questionário no Google Forms.

A nova infraestrutura proporciona a capacidade de comunicação com aplicações externas, tais como o Grafana e o DevOps.

O futuro software agora tem a flexibilidade de expansão em diversas direções. Após a conclusão do TFC, diversas oportunidades surgem para expandir o software ainda mais, sujeitas à concordância ou interesse da empresa em explorar essas possibilidades.

## **2.7. Proposta de Mudanças**

A proposta de mudanças apresenta uma abordagem abrangente e bem estruturada para resolver os problemas identificados no sistema *Legacy* da Barraqueiro. A migração para tecnologias mais modernas, a centralização do sistema e a modularização das aplicações são estratégias sólidas para melhorar a segurança, a eficiência e a manutenção do sistema. A viabilidade e pertinência das mudanças propostas são altas, e a implementação dessas alterações pode resultar em benefícios significativos para a empresa.

Em suma, as propostas de alteração delineadas para a migração para C# e Plataforma .NET e React, a subdivisão do sistema em três aplicações, a implementação de um novo middleware e a centralização do sistema demonstram um compromisso sólido do grupo Barraqueiro em enfrentar os desafios tecnológicos com inovação e pragmatismo. Ao alinhar-se com as exigências

contemporâneas, estas transformações não só respondem às necessidades presentes, como também estabelecem uma base robusta para o futuro. A elevada viabilidade e pertinência destas mudanças sugerem que, se implementadas, proporcionarão benefícios substanciais, não só em termos de eficiência operacional, segurança e custos, mas também na capacidade de adaptação contínua às exigências tecnológicas em constante evolução. Assim, a implementação destas propostas não só representa uma modernização técnica, mas uma estratégia visionária para fortalecer a posição do grupo Barraqueiro no cenário tecnológico em rápida transformação.

É importante notar que, apesar de algumas datas planejadas não terem sido cumpridas conforme originalmente definido, o resultado final alcançado atendeu às expectativas esperadas. Um aspecto significativo a ser acrescentado é a inclusão das definições de processamento do middleware na base de dados, o que não estava inicialmente contemplado no escopo do projeto, mas foi incorporado durante o desenvolvimento.

Essas adições demonstram a flexibilidade e adaptabilidade do projeto às necessidades emergentes, sem comprometer a estrutura inicialmente planejada. Portanto, o relatório final deve enfatizar que essas alterações não afetaram negativamente a viabilidade nem a pertinência do TFC, mas sim fortaleceram a solução final ao atender requisitos adicionais identificados ao longo do processo de implementação.

Em resumo, o estudo final deve destacar não apenas a entrega do produto final conforme o esperado, mas também a capacidade do projeto de se ajustar e expandir conforme as demandas evoluíram, garantindo assim uma solução robusta e alinhada com os objetivos estratégicos do Grupo Barraqueiro.

### 3. Benchmarking

O Sistema GALP, desenvolvido de acordo com as necessidades específicas do Grupo Barraqueiro, assume uma posição única no contexto operacional da empresa. A sua conceção personalizada foi guiada pelas exigências únicas e complexas do Grupo, resultando num sistema altamente especializado e alinhado com as particularidades do setor.

A singularidade do Sistema GALP destaca-se pelo facto de que nenhuma outra aplicação existente no ambiente do Grupo Barraqueiro possui um conjunto de funcionalidades comparáveis. Cada recurso, característica e módulo do Sistema GALP foi cuidadosamente delineado para satisfazer as exigências específicas e complexidades operacionais da empresa, consolidando-se como uma ferramenta indispensável para o seu funcionamento eficaz.

Esta singularidade, por conseguinte, impõe uma distinção notável quando se trata de comparações com outras aplicações internas do Grupo. A ausência de paralelos funcionais diretos destaca a unicidade do Sistema GALP, tornando-o o ponto focal para referências e análises dentro do contexto operacional. Qualquer avaliação ou comparação inevitavelmente se concentra na nova aplicação GALP, dada a sua natureza exclusiva e adaptada às demandas específicas do Grupo Barraqueiro.

Dessa forma, o Sistema GALP não apenas representa uma solução tecnológica, mas também uma resposta direta às exigências operacionais únicas do Grupo. A sua singularidade, ao destacar-se como uma aplicação sem paralelos internos, confere-lhe um papel central nas discussões e avaliações no cenário tecnológico da empresa.

Assim iremos realizar uma análise mais aprofundada sobre os detalhes de cada software através de uma tabela:

**Tabela de comparação entre o software antigo com o software atual**

	SOFTWARES		
Qualificadores	Martelo	GALP	BFuel
Aplicação Segura	X	X	✓
Manutenção Complexa	✓	✓	X
Consistente	X	X	✓
Escalável	X	X	✓
Sistema centralizado	X	X	✓
Interface fácil de usar	X	X	✓



Linguagem de programação recente	X	X	✓
Compatibilidade com base de dados	✓	X	✓
Compatível opcionalmente com cloud	X	X	✓
Interface Multi-aplicacional	X	X	✓
Sistemas Operativos Atualizados	X	X	✓

Apresentação das interfaces presente nestes softwares *legacy* é um ponto importante que deve de ser levado em consideração na criação da nova interface, desta forma iremos apresentar algumas das interfaces que fazem parte da aplicação “Galp” juntamente com a sua avaliação.





Figura 9 - Interface do Software antigo

Nesta figura 2 podemos observar que esta interface era bastante complexa a nível de conteúdos, uma vez que todas as funcionalidades da mesma estão subjetivas e sem compreensão do utilizador, as cores também não favorecem a visualização de todo o software e a utilização do mesmo. Desta forma propusemos uma interface simples com objetivo no minimalismo permitindo um bom visionamento de todos os utilizadores e a visualização clara de todas as funcionalidades do software novo, iremos detalhar a nova interface na [solução](#) desenvolvida deste relatório.

Após a conclusão do projeto, a comparação entre o novo sistema e outros softwares não sofre alterações significativas, uma vez que não há outra aplicação disponível que possa gerenciar tão profundamente e com o mesmo nível de conhecimento específico do negócio como o nosso. O Sistema GALP foi meticulosamente desenvolvido para atender às exigências únicas e complexas do Grupo Barraqueiro, integrando funcionalidades e características que refletem um profundo entendimento das operações da empresa.

Essa singularidade é fundamental, pois não apenas posiciona o novo sistema como uma solução tecnológica robusta, mas também como uma resposta estratégica às demandas operacionais específicas da organização. Diferentemente de outros softwares legados ou alternativos, o GALP não apenas atende aos requisitos básicos de gestão, mas também oferece uma abordagem personalizada que otimiza processos internos e melhora a eficiência operacional.

## 4. Engenharia

### 4.1. Levantamento e análise dos Requisitos

#### 4.1.1. Objetivos do Sistema:

- Desenvolver um novo sistema de gestão integrada para o Grupo Barraqueiro, substituindo os módulos existentes nomeada o Martelo (middleware) a aplicação Galp (Interface) e sua respectiva base de dados, mantendo as bases de dados externas nomeadamente AS400 e SQLServer.
- Melhorar a eficiência operacional e a segurança do sistema.
- Proporcionar uma interface mais moderna e fácil de usar para os utilizadores.
- Centralizar o sistema para facilitar o controlo de acessos e reduzir vulnerabilidades.
- Reduzir custos operacionais, licenças e computação necessária.

#### 4.1.2. Requisitos Funcionais:

##### 4.1.2.1. Novo Middleware:

**Requisito Funcional 1 (Esforço: Moderado, Importância: Alta)** : Leitura e escrita eficiente com as bases de dados do grupo (AS400, SQL Server 2014) e a base de dados da nova aplicação.

**Descrição:** Garantir uma comunicação eficiente e segura entre a aplicação e as bases de dados do grupo garantindo assim que não existe perda de dados e é feita de forma rápida.

**Pré-condições:** Mapeamento claro dos fluxos de dados entre a aplicação e os servidores

**Requisito Funcional 2 (Esforço: Moderado, Importância: Alta)** : Processamento e envio de dados entre o novo Sistema BFuel e os servidores de bases de dados através de uma camada de api

**Descrição:** Implementar processamento eficiente e envio de dados entre os módulos do novo sistema BFuel e os servidores de bases de dados através de api 's.

**Pré-condições:** Configuração adequada das bases de dados e análise das especificações de comunicação

**Requisito Funcional 3 (Esforço: Moderado, Importância: Alta)** : Eliminação da necessidade de transferência de ficheiros entre o novo sistema BFuel e outros servidores.

**Descrição:** Reduzir a dependência de transferência de ficheiros, otimizando a comunicação direta entre a o novo sistema BFuel e os servidores.

**Pré-condições:** Avaliação da eficácia dos métodos de transferência de ficheiros existentes

**Requisito Funcional 4 (Esforço: Moderado, Importância: Alta)** : Envio de dados para uma base de dados local MySQL e outras bases de dados do grupo.

**Descrição:** Implementar o envio de dados traduzidos para bases de dados específicas, nomeadamente a base de dados local do sistema BFuel, e AS400 e SQL Server 2014 caso necessário

**Pré-condições:** Configuração adequada das bases de dados e definição de critérios de envio.

#### **4.1.2.2 . Nova Camada de API:**

**Requisito Funcional 5 (Esforço: Baixo, Importância: Média):** Suporte a transações.

**Descrição:** Garantir que as operações de leitura e escrita nas bases de dados sejam realizadas de forma atômica, permitindo o rollback em caso de falha e mantendo a consistência dos dados.

**Pré-condições:** Suporte a transações nos sistemas de bases de dados utilizados.

#### **4.1.2.3. Novo Software Frontend:**

**Requisito Funcional 6 (Esforço: Alto, Importância: Alta) :** Comunicação com a API para efetuar a comunicação direta com a base de dados.

**Descrição:** Chamar a api quando necessário efetuar uma troca de dados com a base de dados de forma rápida e segura e assíncrona.

**Pré-condições:** Definição clara dos requisitos da aplicação web que indiquem os momentos e as circunstâncias em que será necessária a interação com a base de dados por meio da API. Especificações claras sobre os requisitos da aplicação web

**Requisito Funcional 7 (Esforço: Alto, Importância: Alta) :** Desenvolvimento de uma web app em ReactJS:

**Descrição:** Criar uma aplicação web interativa e moderna utilizando JavaScript e ReactJS para uma melhor experiência do utilizador.

**Pré-condições:** Especificações claras sobre os requisitos da aplicação web

**Requisito Funcional 8 (Esforço: Alto, Importância: Alta) :** Gestão de bombas, análise de gráficos, receção de alertas, solicitação de combustíveis, visualização de abastecimentos e gestão de autocarros.

**Descrição:** Integrar funcionalidades-chave na aplicação web, incluindo gestão de bombas, análise de gráficos e outras operações relevantes.

**Pré-condições:** Identificação das necessidades operacionais dos utilizadores finais

**Requisito Funcional 9 (Esforço: Moderado, Importância: Alta)** : Interface minimalista, fácil de usar e consistente.

**Descrição:** Garantir que a interface da aplicação web é minimalista, intuitiva e consistente em todas as funcionalidades.

**Pré-condições:** Análise das preferências e feedback dos utilizadores

#### **4.1.2.4. Centralização do Sistema:**

**Requisito Funcional 10 (Esforço: Alto, Importância: Alta)** : Migração de todos os módulos do sistema para um servidor central.

**Descrição:** Transferir todas os módulos para um servidor centralizado para facilitar o controlo e a segurança

**Pré-condições:** Avaliação da infraestrutura existente e definição de requisitos de migração.

**Requisito Funcional 11 (Esforço: Moderado, Importância: Alta)** : Acesso do middleware ao computador da bomba para processamento de ficheiros.

**Descrição:** Permitir o acesso do middleware ao computador da bomba para eficiente processamento de ficheiros.

**Pré-condições:** Configuração adequada dos sistemas e autorização de acesso.

**Requisito Funcional 12 (Esforço: Alto, Importância: Alta)** : Implementação de rotinas de backups locais e em outros servidores.

**Descrição:** Desenvolver rotinas robustas de backup para garantir a segurança e a disponibilidade dos dados

**Pré-condições:** Identificação dos dados críticos e definição de políticas de backup.

#### **4.1.3. Requisitos Não Funcionais:**

##### **4.1.3.1. Segurança:**

**Requisito Não Funcional 1 (Esforço: Alto, Importância: Alta)** : Práticas robustas de segurança proporcionadas pelo C# e ReactJS.

**Descrição:** Implementar práticas de segurança robustas proporcionadas pelas tecnologias C# e ReactJS.

**Pré-condições:** Avaliação das melhores práticas de segurança e configurações adequadas.

#### 4.1.3.2. Usabilidade e Experiência do Utilizador:

**Requisito Não Funcional 2 (Esforço: Alto, Importância: Alta)** : Interface web moderna e interativa.

**Descrição:** Desenvolver uma interface web moderna e interativa utilizando a tecnologia ReactJS

**Pré-condições:** Especificações claras sobre as preferências dos utilizadores e os requisitos de interação.

**Requisito Não Funcional 3 (Esforço: Moderado, Importância: Alta)** : Facilidade de gestão de bombas, análise de dados e outras funcionalidades.

**Descrição:** Garantir que as funcionalidades chave, como gestão de bombas e análise de dados, são de fácil utilização.

**Pré-condições:** Análise das necessidades operacionais e feedback dos utilizadores.

**Requisito Não Funcional 4 (Esforço: Moderado, Importância: Alta)** : Consistência na interface para uma experiência do utilizador coesa.

**Descrição:** Assegurar consistência na interface para proporcionar uma experiência do utilizador coesa em todas as funcionalidades.

**Pré-condições:** Definição de padrões de design e interação.

#### 4.1.3.3. Eficiência Operacional:

**Requisito Não Funcional 5 (Esforço: Alto, Importância: Alta)** : Centralização do sistema para redução de redundância de dados.

**Descrição:** Centralizar o sistema para eliminar redundâncias de dados e otimizar as operações.

**Pré-condições:** Avaliação da estrutura de dados existente.

**Requisito Não Funcional 6 (Esforço: Alto, Importância: Alta)** : Migração da base de dados de dBase para MySQL para maior robustez.

**Descrição:** Realizar a migração da base de dados de dBase para MySQL para garantir maior robustez e suporte para expansão futura.

**Pré-condições:** Avaliação da estrutura da base de dados existente

#### **4.1.4. Restrições e Limitações:**

Adaptação à tecnologia ReactJS para o frontend.

Dependência de tecnologias Microsoft (C#, .NET Core) devido à maioria das aplicações do grupo operarem em .NET.

Necessidade de cooperação efetiva com a empresa para garantir viabilidade e aceitação.

#### **4.1.5. Stakeholders:**

Grupo Barraqueiro (patrocinador principal).

Utilizadores Finais (operadores, gestores de bombas, etc.).

#### **4.1.6 Entregáveis:**

Novo sistema que consiste num novo middleware, camada de api, frontend (BFuel).

Documentação completa do sistema.

Relatórios de testes e validação.

## 4.2. Diagramas de Casos de Uso

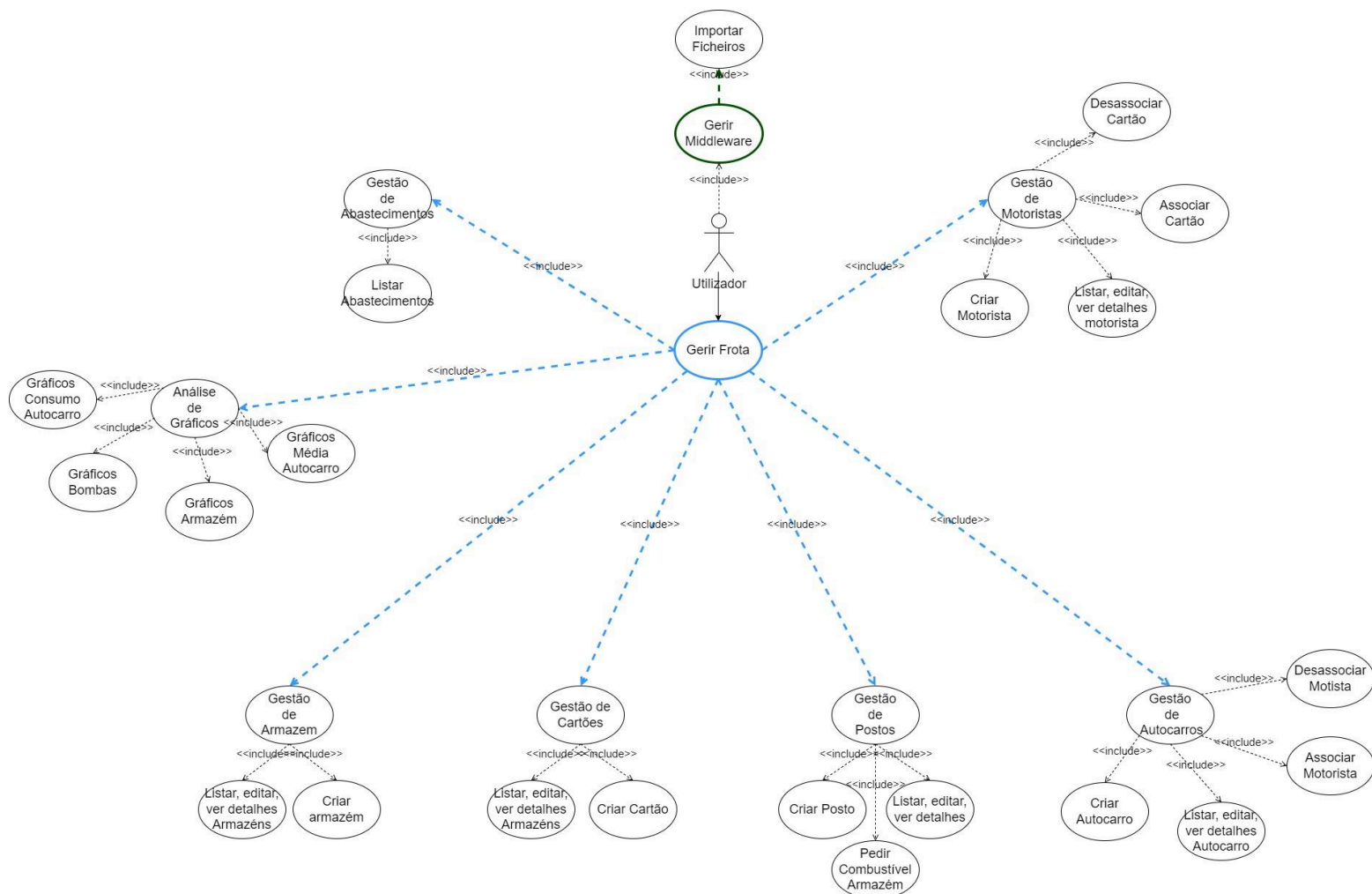


Figura 3 - Diagrama de Casos de Uso

Neste diagrama de caso de uso podemos observar a gestão toda deste software através do ponto principal para a integração de sistemas que é o middleware, o qual inclui a importação de ficheiros e a gestão dos utilizadores. Seguimos para a parte mais centralizada que é a gestão de toda a frota a qual tem ligação direta com a gestão de motoristas, a gestão de autocarros, a gestão de postos, a gestão de cartões, a gestão de armazém, a análise de gráficos e a gestão dos abastecimentos. Analisando agora cuidadosamente cada uma destas ligações diretas conseguimos verificar que:

- A gestão dos motoristas é a gestão do pessoal que opera dos veículos neste caso os autocarros e onde ocorre a atribuição e a desassociação de cartões aos motoristas;
- A gestão dos autocarros é específica para a criação e gestão de detalhes dos autocarros como associar ou desassociar um motorista;
- A gestão de postos de abastecimento de combustível é onde podemos criar e administrar os pedidos de combustível;



- A gestão de cartões de identificação para a vinculação dos cartões aos motoristas cartões;
- A gestão de armazém onde ocorre a administração do inventário de armazéns, possivelmente de combustível ou peças armazém;
- A gestão de abastecimentos é onde ocorre a monitorização e administração do consumo e distribuição de combustível com a listagem dos mesmos;
- A análise de gráficos permite uma análise detalhada de toda a frota desde o número de autocarros a ser utilizado, as bombas, os armazéns e entre outros aspetos relevantes para a empresa;

### 4.3. Diagramas de Atividades

O diagrama representado abaixo é o diagrama de atividade mais relativo para o projeto, o qual é a gestão de abastecimentos.

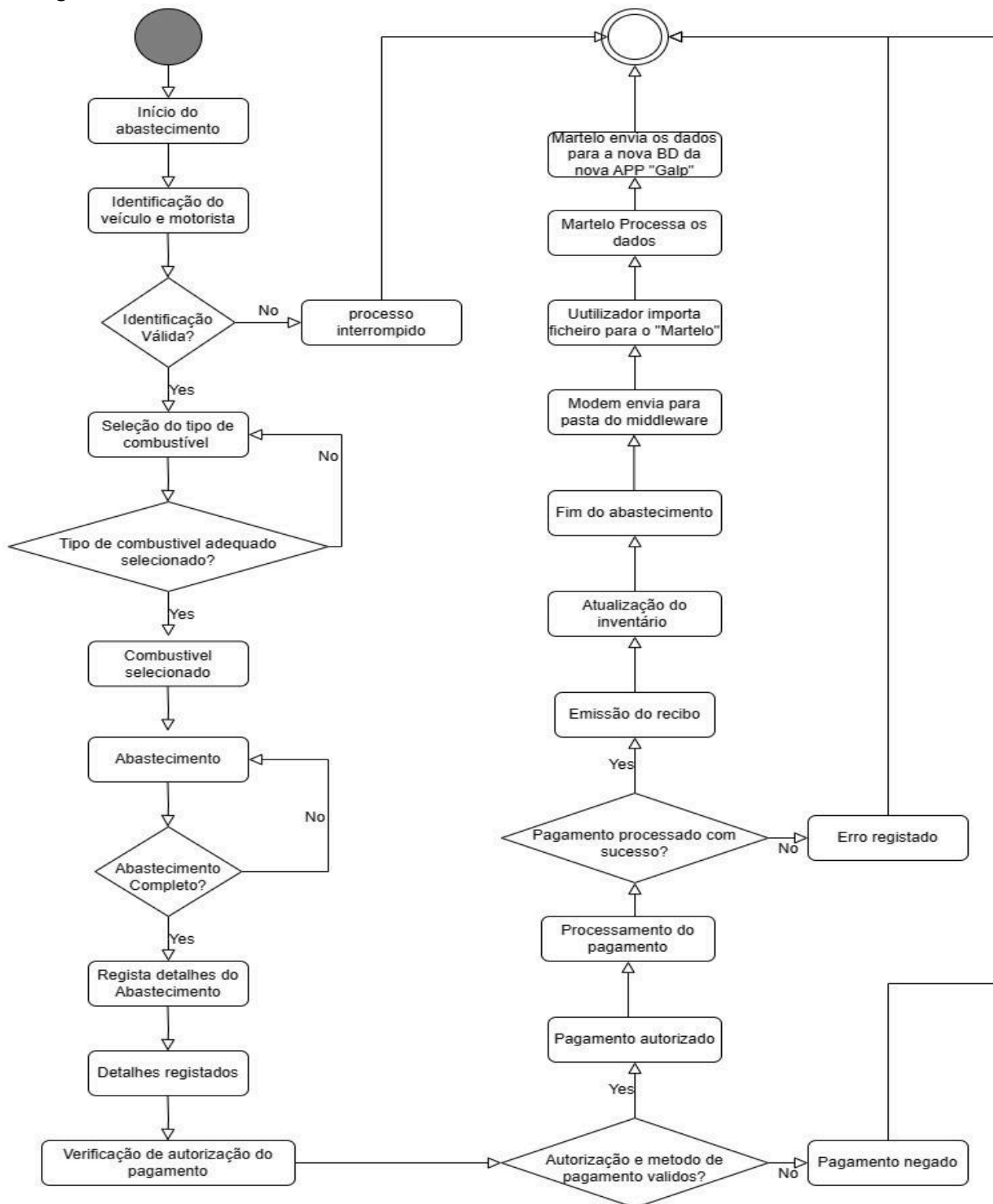


Figura 4 - Diagrama de Atividade

## 4.4. Modelos relevantes

### 4.4.1. Modelo de Classes

Nesta secção iremos apresentar o modelo de classes previsto para o desenvolvimento deste projeto. Iremos então definir as mesmas e fazer uma análise de cada uma:

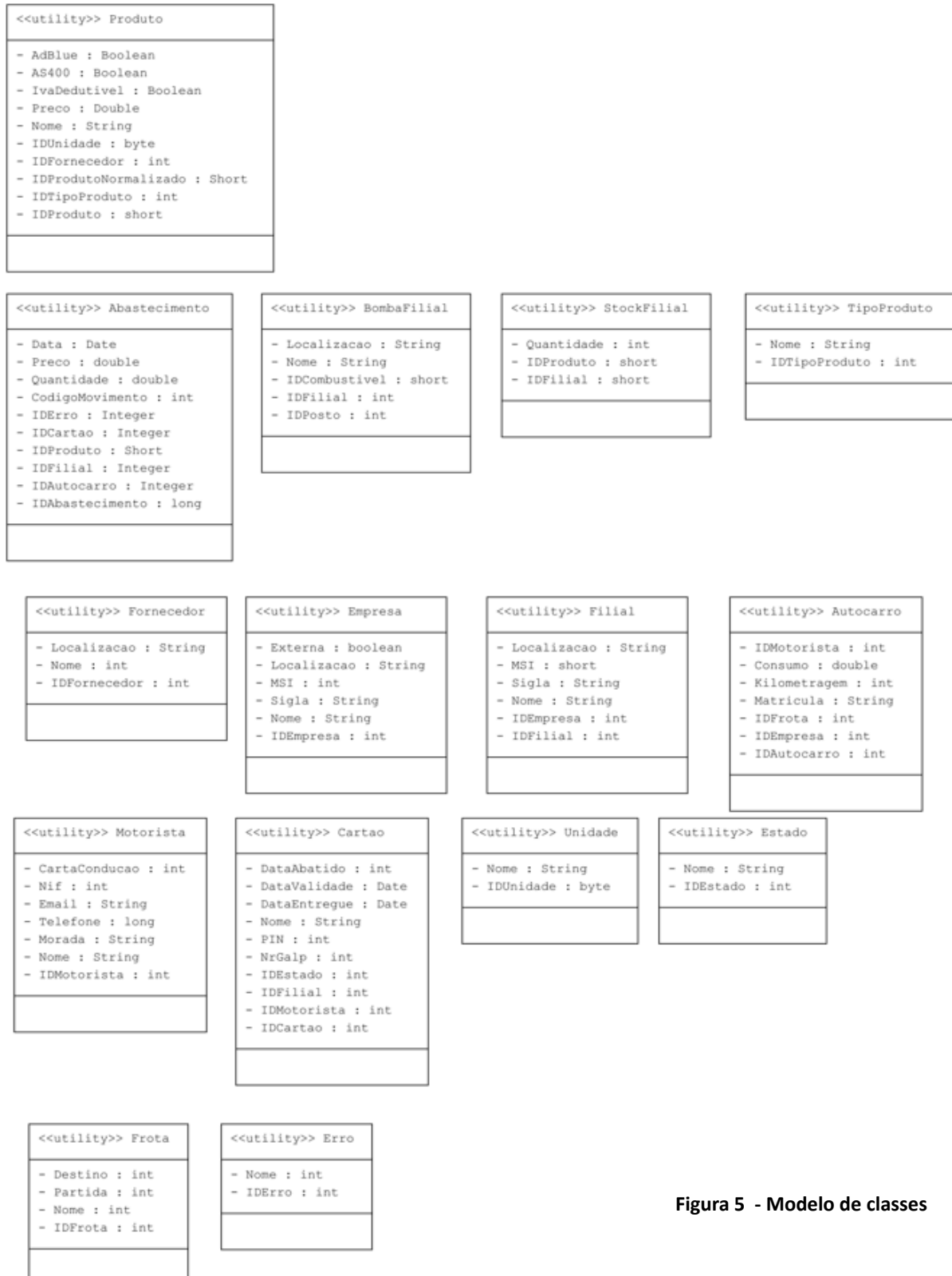


Figura 5 - Modelo de classes

Análise do modelo de classes:

- Produto:

Representa os detalhes dos produtos, incluindo se estão ativos, se são tributáveis, o nome, a identidade do tipo de produto, o código normalizado e o identificador único.

- Abastecimento:

Detalha os eventos de abastecimento, com data, preço, quantidade, e identificadores para o motorista, veículo, empresa e bomba de abastecimento.

- BombaFilial:

Relacionado a uma bomba específica numa localização de filial, com informações sobre localização, nome e identificadores para a bomba, combustível e tipo de produto.

- StockFilial:

Gere a quantidade de produtos numa filial, com quantidades e identificadores para filial e produto.

- TipoProduto:

Classifica os tipos de produtos, com um nome e um identificador único.

- Fornecedor:

Contém informações sobre os fornecedores, incluindo localização, nome e um identificador único.

- Empresa:

Representa uma empresa, com a possibilidade de ser uma entidade externa, informações sobre localização, sigla, nome e um identificador único.

- Filial:

Detalha as filiais da empresa, com localização, sigla, nome e identificadores para a filial e a empresa.

- Autocarro:

Representa veículos do tipo autocarro, com informações sobre o motorista, consumo, quilometragem, matrícula e identificadores para frota, empresa e o próprio autocarro.

- Motorista:

Detalha as informações dos motoristas, com número da carta de condução, NIF, telefone, email, nome e identificadores para a filial e o motorista.

- Cartão:

Gere os dados dos cartões utilizados no sistema, com data de validade, valor de saldo, nome, PIN e identificadores para filial, motorista e cartão.

- Unidade:

Está relacionada à unidade de medida para os produtos, com um nome e um identificador.

- Estado:

---

Relacionado ao estado dos cartões, caso o cartão esteja ativo ou não.

- Frota:

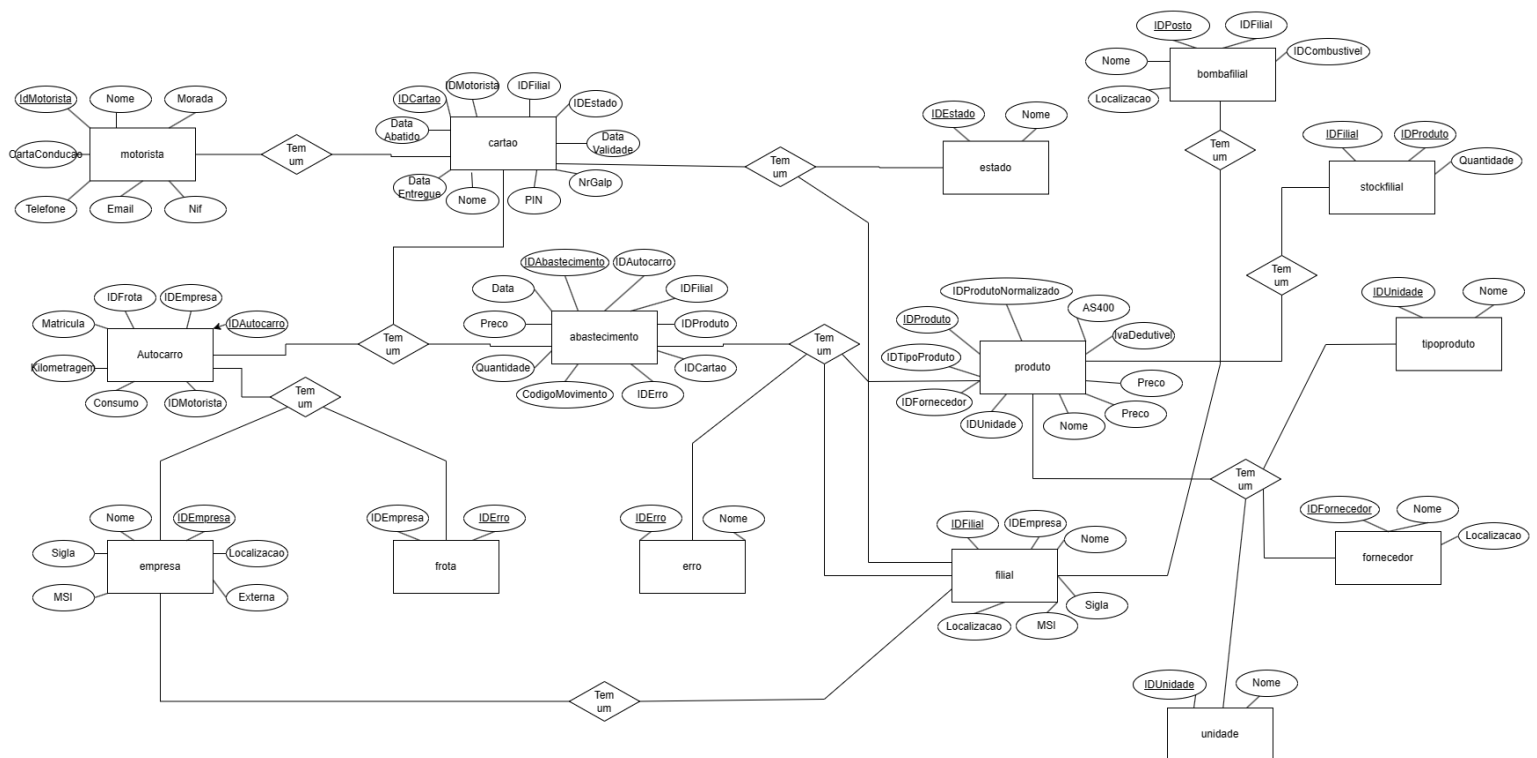
Engloba os dados sobre uma frota específica, com destino, nome, um identificador da frota e um identificador de partida.

- Erro:

Destinada a registrar erros dentro do sistema, com um nome e um identificador.

#### 4.4.2. Modelo de relação-entidade

Neste modelo podemos observar as relações entre cada entidade, as quais estão descritas detalhadamente na seção 2.4.1 . Logo, nesta secção vamos analisar as relações entre elas.



**Figura 6 - Modelo relação-entidade**

Com base na imagem fornecida, o diagrama representa um modelo de entidade-relacionamento (ER) complexo, com várias entidades e relações entre elas. A seguir, detalharei as relações e entidades conforme interpretadas da imagem:

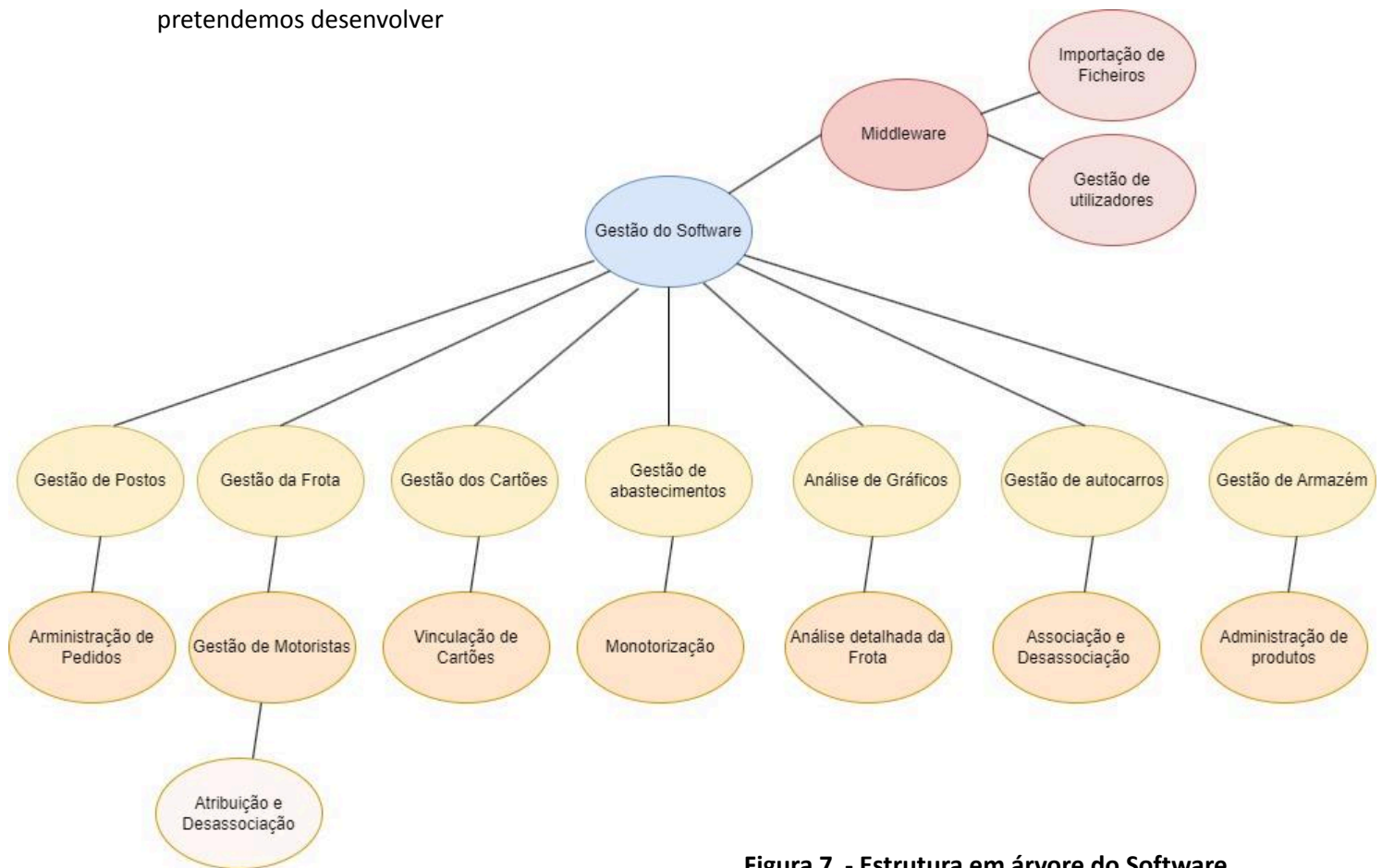
- Motorista: Esta entidade possui atributos pessoais como Nome, Morada, e está relacionada com a entidade "Cartão" (indicando que cada motorista tem um ou mais cartões) e com a entidade "Abastecimento" (mostrando que motoristas realizam abastecimentos).

- Cartão: Relacionado ao "Motorista" e "Estado", possivelmente indicando o estado atual do cartão (ativo ou bloqueado). Também está associado à entidade "Empresa" através da relação "Filial", sugerindo que o cartão pode estar vinculado a uma filial específica da empresa.
- Abastecimento: Conecta-se a várias entidades como "Produto" (indicando o que foi abastecido), "Motorista" (quem fez o abastecimento), e "Cartão" (o meio de pagamento utilizado).
- Empresa e Filial: "Empresa" está associada a "Erro", indicando que erros podem ser registrados a nível empresarial. "Filial" está relacionada a "Empresa", significando que uma empresa pode ter várias filiais. "Filial" também está ligada a "Motorista", "Cartão" e "Erro", implicando que essas entidades podem ser gerenciadas a nível de filial.
- Produto: Esta entidade está no centro do diagrama, relacionada com "Abastecimento", "TipoProduto" (classificando o produto), "Fornecedor" (indicando de onde o produto vem) e "StockFilial" (indicando o estoque do produto em uma filial).
- Fornecedor: Relacionado com "Produto", mostra que fornecedores entregam produtos, e também está associado à "Empresa", o que pode indicar que fornecedores são empresas externas que fornecem produtos para a empresa em questão.
- StockFilial e BombaFilial: Essas entidades estão relacionadas a "Filial" e "Produto", indicando o gerenciamento de estoque e abastecimento nas filiais.
- TipoProduto, Unidade e Estado: "TipoProduto" está relacionado com "Produto", como já mencionado. "Unidade" pode estar relacionada à unidade de medida dos produtos, e "Estado" pode representar diferentes estados de entidades como "Cartão" ou "Abastecimento".
- Erro: Esta entidade sugere um sistema de registro de erros que está ligado tanto à "Empresa" quanto à "Filial", significando que erros podem ser rastreados e gerenciados em ambos os níveis organizacionais.

As relações são complexas, dessa forma temos de ter um sistema robusto para o gerenciamento de recursos e operações.

## 4.5.Estrutura

Nesta secção iremos apresentar um esquema de árvore representativo ao software que pretendemos desenvolver



**Figura 7 - Estrutura em árvore do Software**

A análise aprofundada sobre o esquema em árvore apresentado, começa com a identificação clara do nível superior que é a gestão de software, este nó é o elemento central do sistema de gestão de software, do qual todos os outros módulos dependem e estão interligados. Seguimos para a identificação dos submódulos diretamente ligados à gestão de software:

- **Gestão de Postos:** Refere-se à gestão de locais físicos ou estações de trabalho.
- **Gestão da Frota:** Possivelmente relacionada com a gestão de veículos ou equipamentos.
- **Gestão dos Cartões:** Pode envolver a gestão de cartões de pagamento, identificação ou acesso.
- **Gestão de Abastecimentos:** Indica a gestão do fornecimento e a disponibilidade de recursos.
- **Análise de Gráficos:** Relaciona-se com a análise visual de dados através de representações gráficas.
- **Gestão de Autocarros:** Específico para a gestão da frota de autocarros.
- **Gestão de Armazém:** Refere-se ao controlo de inventário e armazenamento de mercadorias.
- **Middleware:** Este nó funciona como uma camada intermediária, para a integração e comunicação entre os diferentes sistemas.

Cada um dos principais submódulos tem os seus próprios subcomponentes detalhados, sugerindo funcionalidades e tarefas específicas. Por exemplo, sob "Middleware", há "Importação de Ficheiros" e "Gestão de utilizadores".

A disposição dos nós apresenta uma hierarquia e uma estrutura de ramificação que reflete as relações entre diferentes funções do sistema. As cores utilizadas indicam diferentes categorias ou níveis de importância entre os módulos. A cor laranja para os submódulos secundários indica um nível de detalhe ou especificidade maior em comparação com as cores azul e vermelho, que destacam outros componentes de nível superior.

## 4.6. Mockups

### 4.6.1. Esboço da Interface

Desenvolvemos uma interface destinada a todos os utilizadores, alinhada com a identidade visual do Grupo Barraqueiro, baseada nos princípios do minimalismo e nas cores primárias da empresa. O nosso objetivo com esta interface é ser interativa, fácil de utilizar e intemporal, facilitando assim a sua manutenção ao longo do tempo. Neste sentido, procedemos a uma análise abrangente da nossa interface:

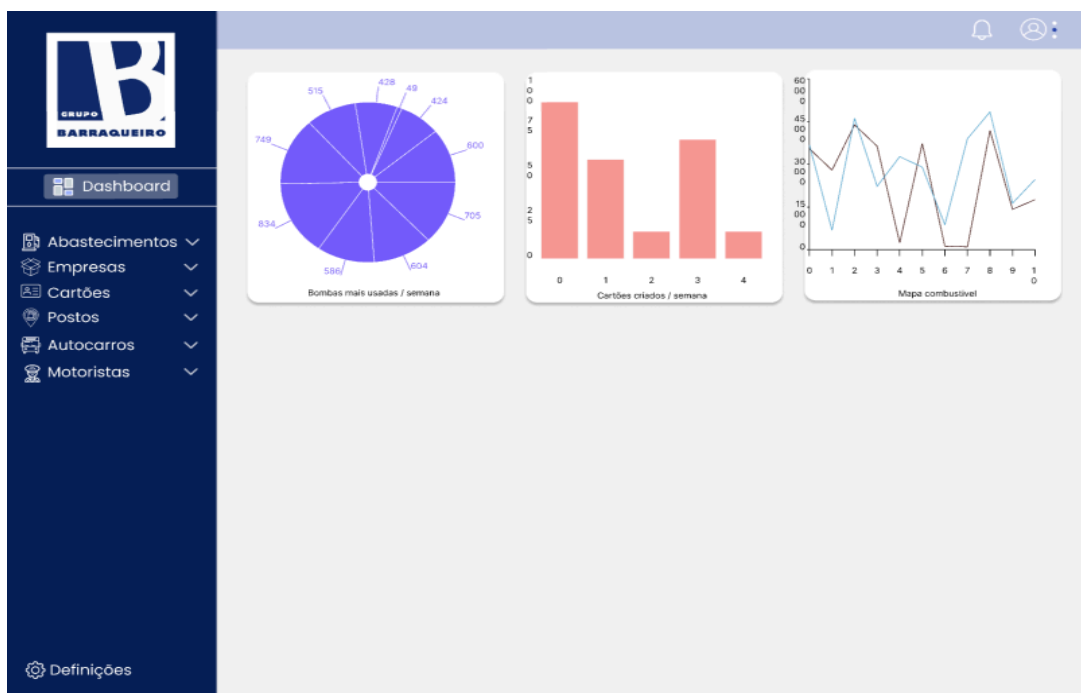


Figura 8 - Interface proposta do ecrã principal para o novo software

Ao observar a aplicação, constatamos que esta incorpora as cores distintivas da empresa, nomeadamente os tons de azul e cinzento. A simplicidade da interface foi uma consideração



central, proporcionando aos utilizadores uma visualização descomplicada das funcionalidades da aplicação. Tendo em conta as pessoas com deficiências visuais, como o daltonismo, evitamos a utilização de cores ou formas complexas, assegurando uma experiência facilitada para todos os utilizadores.

Destacamos a acessibilidade da interface como um aspeto essencial. Acreditamos que a simplicidade não só promove uma experiência mais amigável para o utilizador, mas também facilita a compreensão e a navegação nas diversas funcionalidades oferecidas pela aplicação. Esta abordagem torna a interação mais acessível a todos os utilizadores, independentemente do seu nível de familiaridade com a tecnologia.

Assim, a nossa interface reflete não apenas as cores identificadoras da empresa, mas também se distingue pela sua usabilidade, acessibilidade e durabilidade ao longo do tempo. Comprometemo-nos a oferecer uma plataforma que satisfaça as necessidades de todos os utilizadores, promovendo uma experiência consistente e satisfatória.

#### [Esboço UI TFC – Figma](#)

#### **4.6.2. Esboço da Base de Dados**

Relativamente à base de dados, a aplicação Galp deixará de utilizar a base de dados em dBase e passará a utilizar uma base de dados em MySQL.

No complexo panorama da gestão de dados empresariais, a estruturação eficiente de tabelas assume um papel central na organização e gestão de informações. Este documento explora as principais tabelas de um sistema dedicado à gestão de abastecimento, apresentando uma análise detalhada da estrutura e dos relacionamentos entre estas.

### Principais Tabelas:

- Tabela "Abastecimento": Armazena informações detalhadas sobre cada abastecimento realizado, registrando o veículo, filial, produto, quantidade, preço e data do abastecimento.
- Tabela "Autocarro": Contém dados abrangentes sobre os veículos da frota, incluindo empresa associada, matrícula, quilometragem, consumo e motorista designado.
- Tabela "Filial": Fornece detalhes específicos sobre as filiais da empresa, incluindo informações como empresa associada, nome, sigla e localização.

A vitalidade da interligação entre tabelas é sustentada pelas Chaves Estrangeiras, que estabelecem conexões fundamentais. A "Abastecimento" vincula-se ao veículo, filial e produto,

identificando possíveis erros no processo. A integridade referencial é garantida por estas chaves, permitindo uma ligação coerente e acessível entre os dados das diferentes tabelas.

- Chaves Estrangeiras: Estabelecem ligações vitais entre as tabelas através de chaves estrangeiras. Por exemplo, a tabela "Abastecimento" está ligada ao veículo, filial, produto e identifica possíveis erros no processo.
- Integração entre Tabelas: As chaves estrangeiras garantem a integridade referencial, permitindo uma ligação coerente e acessível entre os dados das diferentes tabelas.

#### Outras Tabelas Relevantes:

Tabelas como "Cartão", "Empresa" e "Produto" oferecem perspectivas adicionais, possivelmente associadas a cartões de abastecimento, detalhes de empresas parceiras e informações sobre produtos de combustível.

- Tabela "Cartão": Possivelmente associada aos cartões de abastecimento para motoristas, contendo dados como estado do cartão, número Galp, PIN, datas de entrega e validade.
- Tabela "Empresa": Detalha informações sobre as empresas associadas à frota, incluindo nome, sigla, localização e se é uma empresa externa.
- Tabela "Produto": Armazena detalhes sobre os produtos de combustível, incluindo tipo, fornecedor, unidade, preço e se possui IVA dedutível ou é destinado ao uso AS400.

#### Objetivo Provável:

A estrutura sugere a implementação de um sistema robusto para a gestão de abastecimento, especialmente desenhado para empresas com múltiplas filiais e uma frota diversificada. O objetivo é monitorar e registar eficientemente os abastecimentos, considerando variáveis como localização, veículo, tipo de combustível e motorista.

- Gestão de Abastecimento: A estrutura sugere a implementação de um sistema robusto para controlo e gestão de abastecimento, especialmente para uma empresa com várias filiais e uma frota diversificada de veículos. O objetivo é monitorar e registar eficientemente os abastecimentos realizados em diferentes locais, veículos, tipos de combustível e motoristas.

#### Detalhes Adicionais:

A introdução de índices e restrições destaca a preocupação com a otimização do sistema. Elementos como chaves primárias e estrangeiras visam manter a consistência dos dados, assegurando a qualidade e integridade do sistema.

- Índices e Restrições: São definidos índices para otimizar consultas e restrições, como chaves primárias e estrangeiras, para manter a consistência dos dados, assegurando a qualidade e a integridade do sistema.

A normalização de dados, abordada posteriormente no texto, emerge como um procedimento essencial para a gestão eficiente de bases de dados, procurando equilibrar a estrutura para minimizar redundâncias e dependências, garantindo assim um sistema coeso e de alto desempenho. É um processo fundamental no âmbito da gestão de bases de dados, procurando

organizar e estruturar a informação de forma a minimizar redundâncias e dependências entre os dados. Este procedimento, comum em bases de dados relacionais, baseia-se em princípios conhecidos como formas normais, onde se destaca a Primeira Forma Normal (1NF), que exige que os valores numa coluna sejam do mesmo tipo, e a Segunda Forma Normal (2NF) e Terceira Forma Normal (3NF), que visam garantir que as colunas não-chave dependem integralmente da chave primária, eliminando dependências transitivas. Os benefícios da normalização incluem a redução de redundâncias, a manutenção consistente de dados, otimização do desempenho de consultas e a facilitação da flexibilidade e escalabilidade do sistema. Contudo, é essencial encontrar um equilíbrio adequado, uma vez que a normalização intensiva pode, em casos extremos, complicar a realização de consultas. Portanto, a normalização deve ser aplicada de forma ponderada, considerando as necessidades específicas do sistema e da aplicação em questão.

#### **4.7. Avaliação de Concretização dos Requisitos**

Todos os requisitos funcionais e não funcionais propostos foram cumpridos integralmente dentro do escopo do projeto inicial. Além disso, foram adicionados requisitos específicos para melhorar a flexibilidade e o desempenho do sistema, como a inclusão de definições de processamento do middleware na base de dados para maior flexibilidade operacional.

#### **4.8. Alterações Realizadas**

Posteriormente ao desenvolvimento do sistema foram adicionados requisitos que permitiam uma maior flexibilidade ao grupo para controlar o processamento do middleware

Nos quais foram :

**Requisito Funcional 13:** Adicionado para incluir definições de processamento do middleware na base de dados, permitindo maior adaptabilidade e eficiência na gestão de dados

**Requisito Funcional 14:** O utilizador deve conseguir fazer a gestão das definições de processamento através da aplicação Web BFuel.

#### **4.9. Avaliação de Valor Obtido**

Todos os requisitos foram implementados conforme planejado, desempenhando um papel crucial na modernização e melhoria da eficiência operacional do Grupo Barraqueiro. A classificação de importância dos requisitos foi fundamental para guiar de forma eficaz o desenvolvimento, assegurando o alcance bem-sucedido de nossas metas estratégicas e operacionais. Essa abordagem estruturada não apenas atendeu às expectativas iniciais, mas também fortaleceu a capacidade do novo sistema de responder de maneira ágil e eficiente às necessidades dinâmicas da empresa.

## 5. Solução Desenvolvida

A solução foi discutida cuidadosamente com o Grupo Barraqueiro até alcançarmos um consenso geral.

Após a conclusão dessa solução, propusemos melhorias no Sistema Galp, composto por um programa denominado "Galp" desenvolvido em Clipper e um middleware chamado "Martelo" desenvolvido em VB6.

Optamos por utilizar C# no desenvolvimento da nova aplicação, uma vez que a maioria das aplicações da empresa já está desenvolvida em .NET Core. A interface deixará de ser uma janela em DOS, evoluindo para uma interface WEB, proporcionando acesso à nova aplicação denominada BFuel. O sistema da nova aplicação terá três módulos: o middleware, desenvolvido em Python, usado como serviço (a correr em background), responsável pela tradução dos dados das bombas, enviando-os para a base de dados via API; a camada de API, desenvolvida em C#, responsável pela comunicação do middleware com as bases de dados necessárias e da comunicação do frontend com a base de dados, funcionando quase como um Man-In-The-Middle; e o frontend denominado de BFuel, desenvolvido em ReactJS, para a nossa interface com o utilizador.

### 5.1. Introdução

A solução desenvolvida foi meticulosamente discutida com o Grupo Barraqueiro, culminando num consenso geral sobre a melhor abordagem. Posteriormente, propusemos melhorias no Sistema Galp, composto por um programa denominado "Galp", desenvolvido em Clipper, e um middleware chamado "Martelo", desenvolvido em VB6. Optámos por utilizar C# no desenvolvimento da nova aplicação, uma vez que a maioria das aplicações da empresa já está desenvolvida em .NET Core. A interface foi atualizada de uma janela em DOS para uma interface WEB, proporcionando acesso à nova aplicação denominada BFuel.

A seguir, apresentamos alguns recursos importantes da solução desenvolvida:

- **Vídeo Demonstrativo de Funcionamento da Solução Desenvolvida:** <https://youtu.be/0qLwoqVb6Zg>
- **Repositório Git:** [guigasthepro/ProjetoTFC: Repositório do Projeto do TFC \(github.com\)](https://github.com/guigasthepro/ProjetoTFC)
- **Solução Funcional:** [Aguarda disponibilização do link por parte do professor]

Este capítulo está estruturado nas seguintes secções:

1. **Arquitectura:** Descrição detalhada da arquitectura do sistema, incluindo diagramas e modelos de entidade-relação.
2. **Tecnologias e Ferramentas Utilizadas:** Apresentação das tecnologias e ferramentas utilizadas no desenvolvimento da solução.
3. **Implementação:** Detalhamento do processo de implementação, com ênfase nas decisões técnicas e nos desafios enfrentados.
4. **Resultado Final:** Apresentação dos resultados finais alcançados com a implementação da solução.

5. **Abrangência:** Discussão sobre a abrangência da solução e o seu impacto no contexto da empresa.

## 5.2.Arquitetura

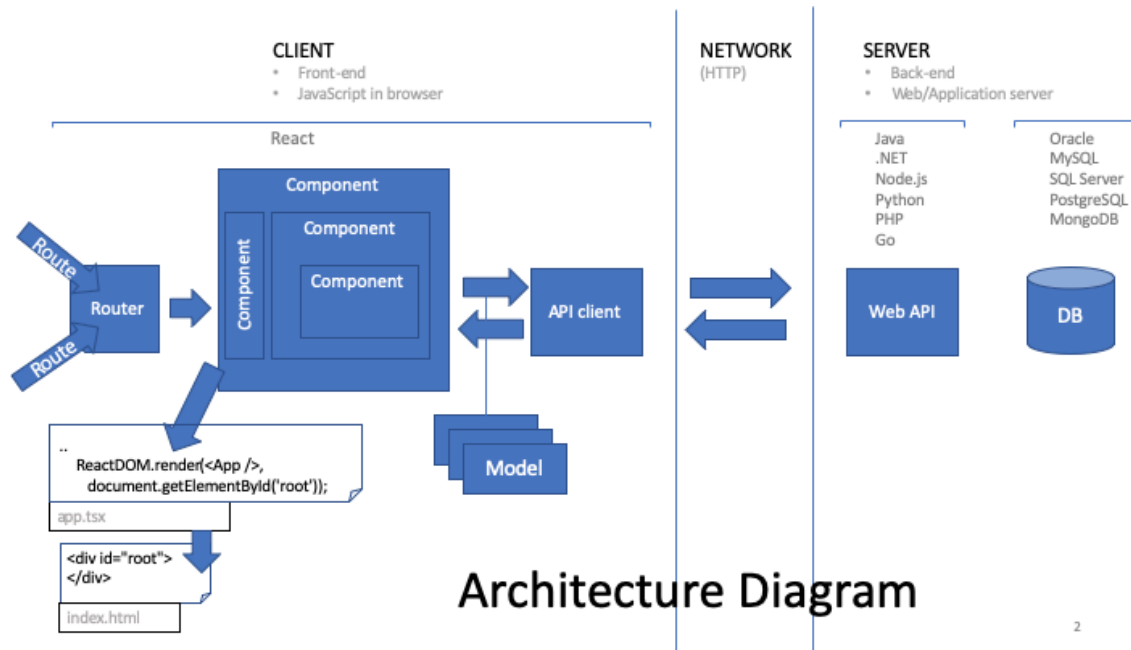


Figura 10 - Representação da Arquitetura ReactJS

### Cliente (Front-end):

Utiliza JavaScript no navegador.

React é usado para criar componentes.

Um roteador gerencia as rotas da aplicação.

### Rede (HTTP):

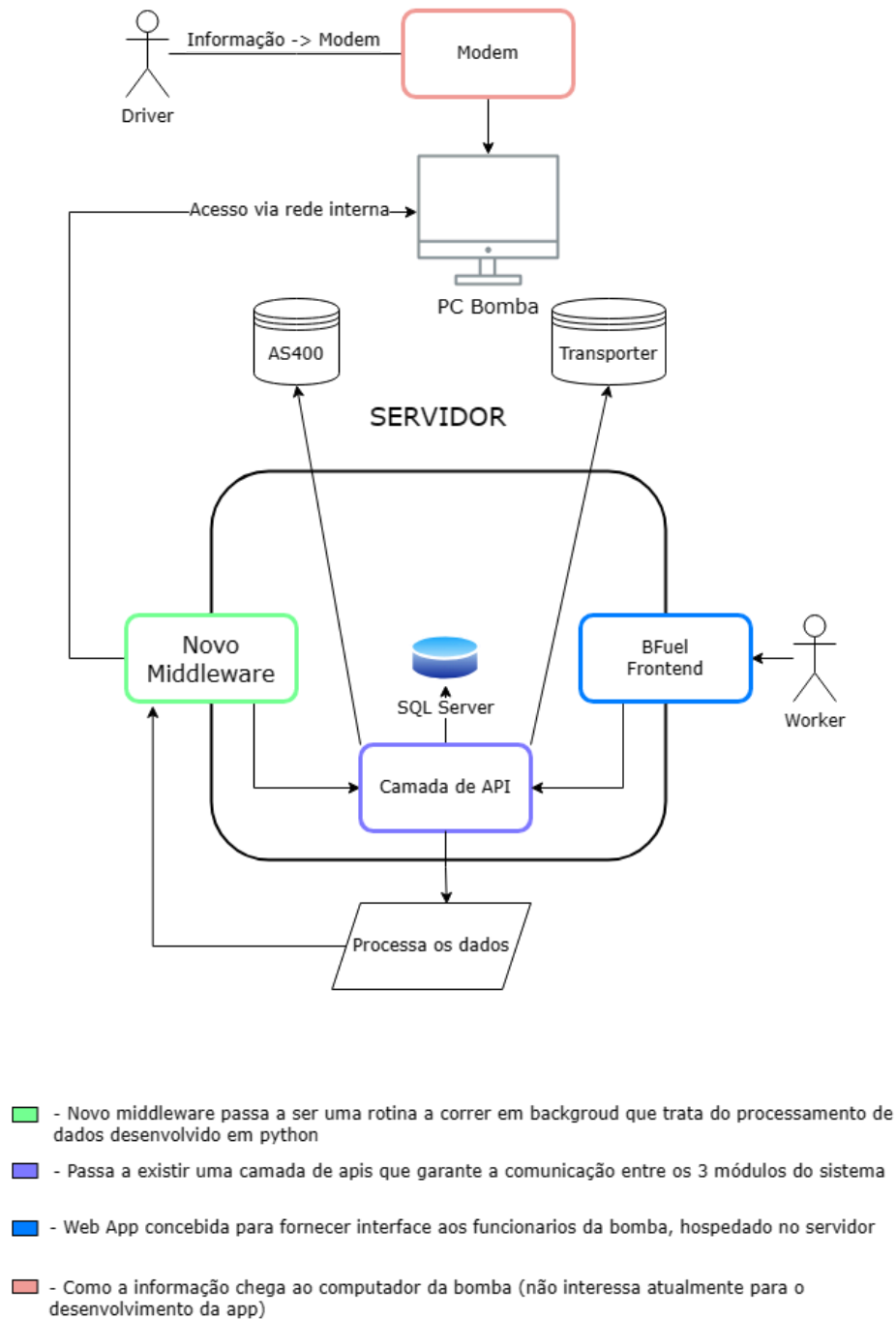
Utiliza o protocolo HTTP para comunicação entre cliente e servidor.

### Servidor (Back-end):

Pode ser implementado em várias linguagens como Java, .NET, Node.js, Python, PHP e Go.

Conecta-se a bancos de dados como Oracle, MySQL, SQL Server, PostgreSQL e MongoDB.

### 5.2.1. Diagrama Infraestrutura



**Figura 11 - Representação da Arquitetura**

Em conjunto com a empresa de maneira a ser viável para todos foi estipulado desenvolver um sistema para substituir os módulos existentes. Esta mudança implica a migração das aplicações para uma tecnologia mais atual. Atualmente, a maioria das aplicações do grupo opera em .NET, o que torna o C# a linguagem mais adequada. Esta transição garantirá total compatibilidade com os sistemas atuais, integração total com serviços em nuvem e até mesmo a possibilidade de hospedagem na nuvem. Esta evolução tem como objetivo eliminar os problemas de segurança, reduzir os problemas de consistência e reduzir a complexidade do sistema, simplificando a

comunicação entre o middleware e a aplicação Galp, eliminando a necessidade de transferência de ficheiros entre ambos.

### **5.2.2.Dividir o sistema em 3 módulos:**

#### **5.2.2.1.Novo Middleware**

O middleware (antigo Martelo) passará a ser uma rotina a ser executada em background desenvolvida em Python com o intuito de apenas traduzir a informação dos ficheiros vindos da bomba para a base de dados, fazendo um pedido à api assim que concluído o processo.

#### **5.2.2.2.Camada de API's**

O programa terá agora uma camada de api's que irá se responsabilizar apenas para efetuar a comunicação entre as bases de dados e os respetivos módulos do sistema (Middleware, e BFuel(Frontend))

#### **5.2.2.3.Novo Frontend**

O novo software Frontend chamado BFuel, será uma web app desenvolvida em ReactJS permitirá ao utilizador fazer a gestão das bombas, a análise dos gráficos, receber alertas, pedir combustíveis aos armazéns, ver os abastecimentos, gerir os autocarros. Permitindo assim ao utilizador uma interface muito mais minimalista, fácil de usar, e consistente.

### **5.2.3.Centralização do Sistema**

Centralização do sistema, todas as aplicações serão migradas para um servidor no qual vai ter acesso às máquinas da bomba, permitindo assim ao middleware acesso ao computador da bomba para processamento dos ficheiros, a centralização do sistema vai permitir que os programas deixem de estar nas máquinas físicas da bomba, permitindo acesso à Web APP via ip/endereço dns. Isto vai corrigir o sério problema de segurança que existe, porque vai permitir instalações de windows mais recentes que recebem os security patches mensalmente. Vai também corrigir o problema de redundância de dados, pois aqui existem rotinas de backups de dados, tanto localmente, como para outros servidores.

## **5.3.Tecnologia e Ferramentas Utilizadas**

No contexto académico, o nosso Trabalho Final de Curso (TFC) está intrinsecamente ligado a disciplinas como Linguagens de Programação 2, onde, apesar de ter sido lecionada em Java, a familiaridade com C# devido ao seu paradigma Orientado a Objetos é evidente. A conexão com Programação WEB ocorre devido à utilização da tecnologia ReactJS, que, de forma única, apresenta uma construção de páginas semelhante à Framework Django. A disciplina de Base de Dados é essencial, pois desenvolvemos uma base de dados do zero e criaremos queries para o programa. A abordagem de Data Science será incorporada, explorando dados para criar previsões e médias, como no caso do consumo médio de um autocarro ao solicitar combustível. Por fim, a disciplina de Computação Distribuída será aplicada devido à comunicação entre aplicações



diferentes por meio de uma API. Essa interligação com diversas disciplinas resulta em uma abordagem holística e abrangente no desenvolvimento do nosso projeto.

## **5.4.Implementação**

Na implementação proposta, integram-se conceitos fundamentais para reforçar a solidez e eficácia do projeto. A arquitetura adotada é modular e escalável, segmentando o sistema em três aplicações e optando por tecnologias como C# e React. Esta abordagem permite o desenvolvimento, manutenção e escalabilidade independentes de cada componente, favorecendo a evolução futura do sistema. A ênfase na segurança é notória com a migração para C# e ReactJS, juntamente com a centralização do sistema. C# é reconhecido pelas suas práticas de segurança robustas, e a centralização simplifica o controlo de acessos, reduzindo vulnerabilidades. A usabilidade e a experiência do utilizador são priorizadas com a transição da interface DOS para uma interface WEB, aliada à escolha da tecnologia React. Este enfoque garante uma experiência moderna e interativa para os utilizadores, fundamentais para a aceitação e eficácia da aplicação. A sustentabilidade do desenvolvimento é assegurada com a migração da base de dados de dBase para MySQL, proporcionando maior robustez e suporte para expansão futura. A opção por tecnologias como C# e ReactJS alinha-se com padrões modernos de desenvolvimento sustentável.

## **5.5. Solução em desenvolvimento**

### **5.5.1. Ponto de Situação**

Antes de dar início ao desenvolvimento da aplicação, realizamos mais uma reunião com o Grupo Barraqueiro, na qual abordámos novamente todos os pontos relevantes, especialmente os requisitos e as linguagens a serem utilizadas para o desenvolvimento do novo sistema. Após esta reunião, fomos informados de que a implementação do frontend deixaria de ser em Blazor Server, conforme previamente acordado, passando a adotar o ReactJS, uma vez que a empresa estava a apostar nesta tecnologia para aplicações Web. Esta alteração teve um impacto significativo no escopo do projeto, uma vez que toda a investigação realizada até então sobre Blazor Server foi anulada, exigindo-nos recomeçar do zero e adquirir conhecimento sobre ReactJS. Esta mudança atrasou consideravelmente o desenvolvimento do projeto, devido à necessidade de aprendermos uma tecnologia com a qual não tínhamos familiaridade.

Em relação ao frontend, destacamos que o mesmo engloba operações CRUD em todos os módulos da antiga aplicação Galp, além de já possuir um dashboard que apresenta estatísticas fictícias. Toda a interface foi construída utilizando a framework de componentes web pré-definidos denominada MUI, o que contribuiu significativamente para acelerar o desenvolvimento.



Figura 12 - Interface desenvolvida (ecrã de dashboard)

Para a criação de objetos, foram utilizados formulários, enquanto para listar, editar e apagar, recorremos a uma DataGrid.

Figura 13 - Interface desenvolvida (ecrã de criação de elementos)

**Gestão de Produtos**  
Gere os teus produtos!

ID	Nome	Tipo Produto	Fornecedor	Produto Normalizado	AS400	AdBlue	Preço	Iva	Unidade	Ações
1	Combustível	1	1	0	false	false	0	false	1	
7	Gasolina	1	1	1	true	true	12	true	1	
8	Gasóleo	1	1	1	false	false	12	false	1	

Rows per page: 100 1-3 of 3

**Figura 14 - Interface desenvolvida (ecrã de visualização de elementos)**

Para mais informação ou mais detalhes sobre as outras páginas, poderão ver a aplicação no video do youtube ou então poderão criar um container em docker com a última versão da imagem disponibilizada com apenas o Frontend a correr.

Link do youtube:

<https://www.youtube.com/watch?v=OWAh4CMfpZo>

Link da imagem do docker:

[oguigas/express - Docker Image | Docker Hub](#)

Para a instalação da imagem disponibilizada segue o tutorial de instalação da mesma:

[\[Tutorial Instalação Imagem\]](#)

Para um acompanhamento mais eficiente do projeto, aconselhamos o uso do GitHub. Com as suas funcionalidades de controlo de versões, gestão de tarefas e colaboração, o GitHub facilita a organização e a comunicação da equipa, promovendo uma execução mais eficaz e transparente.

[\[GitHub TFC\]](#)

## 5.6. Resultado final

Desde a última fase do projeto, apesar de já estarmos com a maior parte dele concluída, implementámos inúmeras melhorias significativas em várias áreas. No frontend, realizámos várias optimizações e aprimoramentos na interface, focando-nos em tornar a interação mais intuitiva e agradável para os utilizadores. Esforçámo-nos para garantir que a navegação fosse mais fluida e que as funcionalidades fossem mais acessíveis e fáceis de utilizar.

Ao nível da API, procedemos a várias melhorias estruturais e funcionais. Estas mudanças não só aumentaram a eficiência e a rapidez na comunicação entre os diferentes componentes do sistema, mas também tornaram a API mais robusta e segura. Melhorámos a capacidade de resposta e a fiabilidade, assegurando que as operações fossem executadas de forma mais consistente e com menor margem de erro.

Na base de dados, realizamos optimizações que permitiram um melhor desempenho e uma gestão mais eficaz dos dados. Implementámos técnicas avançadas de indexação e normalização que reduziram significativamente o tempo de resposta das consultas e melhoram a integridade dos dados. Estas melhorias facilitaram a manutenção da base de dados e permitiram uma escalabilidade mais eficiente para futuras expansões do sistema.

O objetivo principal destas melhorias foi não só elevar a experiência do utilizador a um nível superior, mas também garantir que o sistema fosse mais fácil de manter e atualizar no futuro. Com estas optimizações, tornámos o sistema mais completo, fiável e preparado para responder às crescentes necessidades dos utilizadores e da empresa.

Em relação ao frontend, realizámos uma série de melhorias substanciais para enriquecer a experiência do utilizador e tornar a interface mais funcional e intuitiva. Entre estas melhorias, destacam-se as seguintes:

1. **Mensagens de Feedback ao Utilizador:** Implementámos mensagens de feedback que informam o utilizador sobre o sucesso ou falha das operações realizadas, proporcionando um retorno imediato e claro sobre as ações executadas.
2. **Edição de Dados nas Tabelas:** Adicionámos a possibilidade de editar os dados diretamente a partir das respectivas tabelas. Utilizando inputs apropriados, os utilizadores podem agora actualizar informações de forma mais prática e eficiente, sem necessitar de navegações adicionais.
3. **Uso de gráficos:** Foram adicionados gráficos ao projeto, para facilitar ao utilizador a compreensão e a análise do estado da bomba e dos seus respectivos tanques
4. **Auto-Tradução de Chaves Estrangeiras:** Implementámos uma funcionalidade que traduz automaticamente as chaves estrangeiras de cada classe de dados para o seu nome correspondente. Isto facilita a compreensão e a gestão dos dados, tornando a interface mais amigável e acessível.

5. **Validação ao Apagar Dados:** Para evitar eliminações acidentais, incluímos uma validação ao apagar dados. Esta funcionalidade solicita uma confirmação do utilizador antes de concluir a operação de eliminação, reduzindo assim a probabilidade de erros.

6. **Melhoria do Aspetto das Páginas e Escalabilidade:** Melhorámos significativamente o design e a estética das páginas, garantindo uma apresentação mais agradável e profissional. Além disso, optimizámos a escalabilidade das páginas para assegurar que o layout e as funcionalidades se ajustem corretamente a diferentes tamanhos de ecrã e dispositivos.

7. **Secção Específica para o Middleware:** Criámos uma secção dedicada ao middleware, onde os utilizadores podem inserir e alterar os valores necessários para o processamento de dados. Esta secção centraliza as configurações e ajustes do middleware, facilitando a gestão e a operação do sistema.



Figura 15 - Exemplo de editar



Figura 16 - Exemplo Auto-tradução de chaves estrangeiras

**CRIAR FORNECEDOR**  
Inserir um novo fornecedor

✓ Fornecedor criado com sucesso!

Nome do Fornecedor

Localização

criar

Figura 17 - Feedback ao utilizador

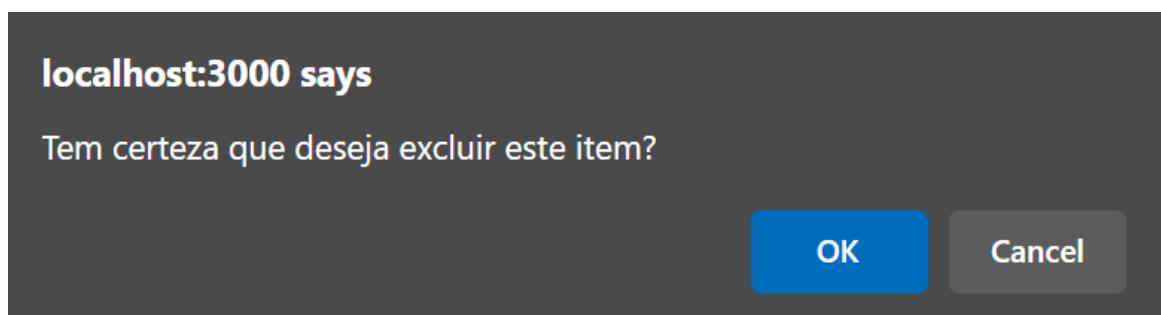


Figura 18 - Validação ao apagar

### Definições middleware

Gere aqui o middleware!

+ ADICIONAR DEFINIÇÕES MIDDLEWARE














Regra	NomeCampo	Posição	Tamanho	Ações
1	numero_terminal	1	3	 
2	viatura	5	4	 
3	codigo_produto	14	1	 
4	quantidade	15	6	 
5	kilometros_abastecimento	21	6	 
6	data_abastecimento	27	8	 
7	hora_abastecimento	35	5	 

Figura 19 - Página do Middleware

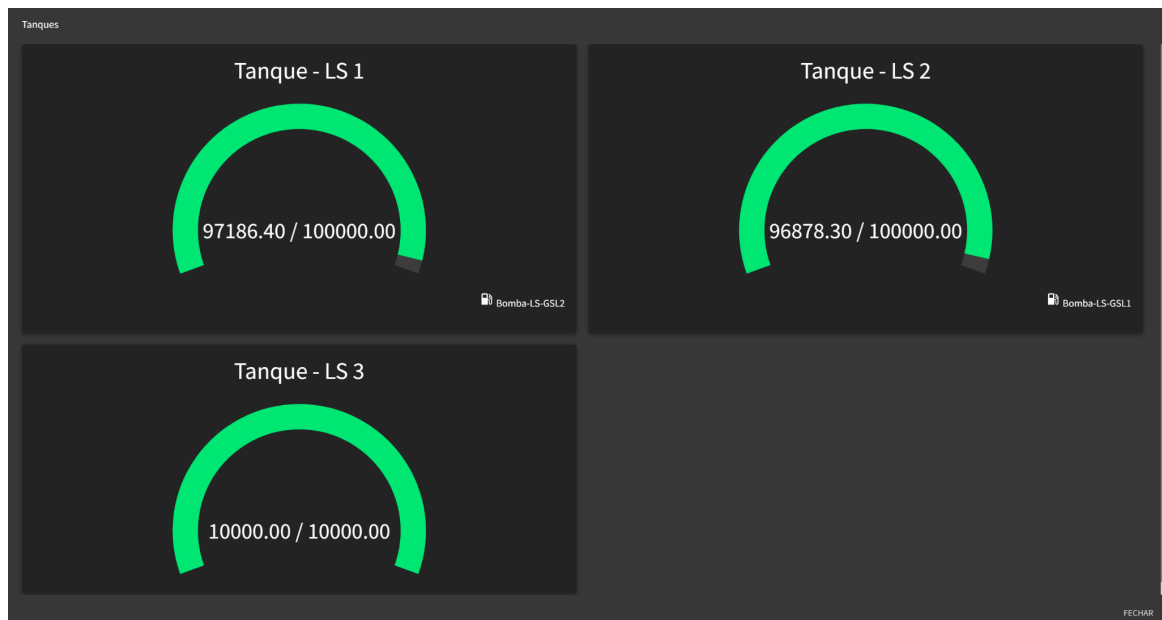
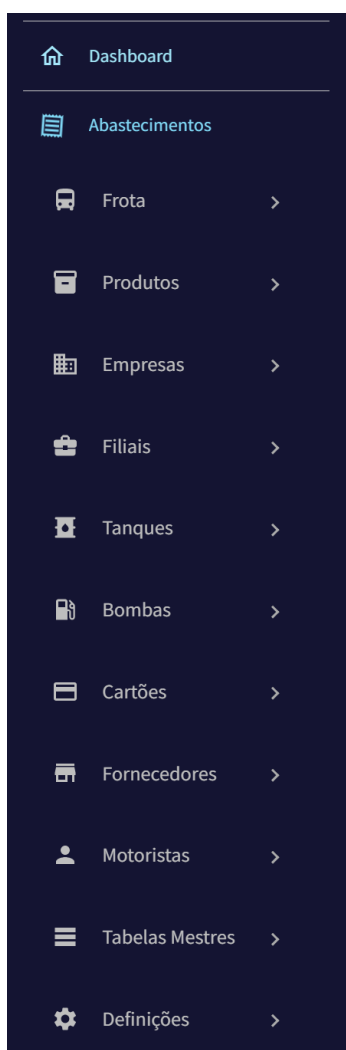


Figura 20 - Exemplo gráficos



**Figura 21- Menu melhorado**

Além das melhorias, o Middleware foi desenvolvido posteriormente, o que causou atrasos em relação ao prazo estabelecido. Originalmente, o Middleware deveria ter sido a primeira etapa, porém isso não ocorreu devido à falta de informações por parte do Grupo Barraqueiro.

Como consequência, o desenvolvimento do Middleware foi adiado para esta entrega. O processamento do Middleware consiste em um ciclo infinito que constantemente busca novos arquivos na pasta de não lidos para processamento. Um acréscimo não previsto na proposta foi a capacidade de alterar dinamicamente as funções de processamento, com suas definições sendo inseridas na base de dados, permitindo que os membros do grupo se adaptem às mudanças nos arquivos gerados pela bomba.

Essa funcionalidade adicional foi implementada para garantir uma flexibilidade maior no sistema, possibilitando ajustes conforme necessário sem exigir intervenções extensivas no código-base. Apesar dos contratempos iniciais, o Middleware agora está integrado de forma eficiente, melhorando significativamente a capacidade de processamento e adaptabilidade do sistema às demandas em evolução do projeto.

Essas modificações foram essenciais para otimizar a operação e garantir que o sistema possa lidar com futuras variações nos dados recebidos, mantendo assim a eficiência e a confiabilidade do serviço oferecido pelo Grupo Barraqueiro.

```
Data e hora convertidas: 2024-04-01 22:55:00
Abastecimento: {'idAutocarro': 7, 'idBomba': 6, 'idProduto': 1, 'idCartao': None, 'idErro': None, 'codigoMovimento': None, 'preco': None, 'quantidade': 14.0, 'data': '2024-04-01T22:55:00'}

API Start

{"idAutocarro": 7, "idBomba": 6, "idProduto": 1, "idCartao": null, "idErro": null, "codigoMovimento": null, "preco": null, "quantidade": 14.0, "data": "2024-04-01T22:55:00"}
API GET Sucessful
```

**Figura 22 - Processamento do Middleware**

## 5.7. Abrangência

A ligação com disciplinas académicas, como Linguagens de Programação 2, Programação WEB, Base de Dados, Data Science e Computação Distribuída, destaca a aplicação prática do conhecimento académico. Esta integração promove um ambiente de aprendizado contínuo e a transferência de conhecimento entre a academia e o projeto real, essencial para o desenvolvimento profissional. A eficiência operacional e a redução de custos são contempladas com a centralização do sistema e a escolha de tecnologias que minimizem a necessidade de servidores intermédios. Isto resulta numa melhoria operacional e redução de custos, especialmente no que diz respeito a licenciamento e computação. A análise de dados para tomada de decisões é realizada com a disciplina de Data Science. A capacidade de estudar dados para criar previsões e médias contribui significativamente para a eficácia das operações, como no caso do consumo médio de um autocarro ao solicitar combustível. Estes conceitos são pilares essenciais para garantir não apenas o sucesso técnico do projeto, mas também para agregar valor à operação global da empresa, promovendo inovação, eficiência e adaptabilidade contínua.



## 6. Plano de testes e validação

### 6.1. Plano de Testes para Validação Prática e Operacional

O objetivo deste plano de testes é validar a solução proposta para o Grupo Barraqueiro, garantindo que atenda aos objetivos estabelecidos e resolva os problemas identificados no sistema *Legacy*. Os testes devem demonstrar a qualidade da solução desenvolvida, validar o seu funcionamento e operação num contexto produtivo, além de verificar a aplicabilidade, pertinência e relevância da solução.

Os testes serão conduzidos em várias etapas, abrangendo diferentes aspetos da solução proposta. Serão utilizados modelos formais de análise de riscos e impacto para garantir uma cobertura abrangente dos cenários de teste. Os testes serão realizados de forma a verificar se a solução cumpre os objetivos definidos, incluindo a melhoria da eficiência operacional, segurança e facilidade de utilização.

Os testes são essenciais para garantir que a solução desenvolvida atenda às expectativas e necessidades do Grupo Barraqueiro. Eles ajudarão a identificar quaisquer problemas ou deficiências na solução antes da implementação num ambiente de produção, reduzindo assim o risco de falhas ou interrupções no sistema

.

### 6.2. Guião de testes detalhado

#### 6.2.1. Teste de Comunicação Eficiente com as Bases de Dados:

Cenário: Verificar se o middleware estabelece comunicação eficiente com as bases de dados do grupo (AS400, SQL Server 2014) e a base de dados da nova aplicação.

Resultado Esperado: Confirmação de que a comunicação é estabelecida sem problemas e que os dados podem ser lidos e escritos corretamente.

#### 6.2.2. Teste de Processamento e Envio de Dados entre o Sistema BFuel e os Servidores de Bases de Dados:

Cenário: Verificar se o processamento e envio de dados entre o sistema BFuel e os servidores de bases de dados ocorrem de forma eficiente.

Resultado Esperado: Confirmação de que os dados são processados e enviados corretamente, sem perda de informação.

#### 6.2.3. Teste de Eliminação da Necessidade de Transferência de Ficheiros:

Cenário: Verificar se a dependência de transferência de ficheiros é reduzida conforme especificado.

Resultado Esperado: Confirmação de que a comunicação direta entre a aplicação Galp e os servidores é estabelecida e que a transferência de ficheiros é minimizada.

#### **6.2.4. Teste de Análise dos Dados Processados pelo Middleware:**

Cenário: Verificar se a rotina para analisar os dados processados pelo middleware funciona conforme esperado.

Resultado Esperado: Confirmação de que a análise dos dados é precisa e relevante para as operações do Grupo Barraqueiro.

#### **6.2.5. Teste de Envio de Dados para Bases de Dados Locais e do Grupo:**

Cenário: Verificar se os dados analisados são corretamente enviados para as bases de dados locais e do grupo.

Resultado Esperado: Confirmação de que os dados são enviados para as bases de dados especificadas de forma precisa e segura.

#### **6.2.6. Teste de Desenvolvimento da Web App em C# utilizando ReactJS:**

Cenário: Verificar se a aplicação web é desenvolvida conforme as especificações utilizando ReactJS

Resultado Esperado: Confirmação de que a aplicação web é interativa, moderna e atende às necessidades dos utilizadores.

#### **6.2.7. Teste de Gestão de Bombas, Análise de Gráficos e Outras Funcionalidades Chave:**

Cenário: Verificar se as funcionalidades chave, como gestão de bombas e análise de gráficos, são implementadas corretamente.

Resultado Esperado: Confirmação de que todas as funcionalidades são integradas de forma eficaz e de fácil utilização.

#### **6.2.8. Teste de Centralização do Sistema:**

Cenário: Verificar se todas as aplicações são migradas para um servidor central.

Resultado Esperado: Confirmação de que o sistema é centralizado conforme planeado, facilitando o controlo e a segurança.

#### **6.2.9. Teste de Implementação de Rotinas de Backups:**

Cenário: Verificar se as rotinas de backup são implementadas conforme especificado.

Resultado Esperado: Confirmação de que os backups são realizados regularmente e de acordo com as políticas definidas.

### **6.3. Validação da Arquitetura Proposta:**

A arquitetura proposta para a solução do Grupo Barraqueiro foi concebida com base numa abordagem modular e escalável, visando atender aos objetivos de melhoria da eficiência operacional, segurança e facilidade de utilização. Para validar essa arquitetura, serão realizados os seguintes passos:

**Revisão da Arquitetura:** Os principais intervenientes, incluindo desenvolvedores, arquitetos de sistemas e representantes do Grupo Barraqueiro, realizarão uma revisão detalhada da arquitetura proposta. Isso incluirá uma análise dos diagramas de arquitetura, descrições de componentes e suas interações.

**Simulação de Fluxos de Dados:** Serão realizadas simulações dos fluxos de dados entre os diferentes componentes da arquitetura, como o middleware, o software backend e frontend, e as bases de dados. Isso garantirá que a comunicação entre os componentes ocorra conforme planejado e que os dados sejam processados e transmitidos corretamente.

**Testes de Escalabilidade:** Serão conduzidos testes de carga e escalabilidade para verificar se a arquitetura é capaz de lidar com um grande volume de dados e utilizadores simultâneos. Isso ajudará a identificar potenciais gargalos de desempenho e garantirá que a arquitetura possa escalar conforme necessário.

**Análise de Segurança:** Uma análise abrangente de segurança será realizada para avaliar a robustez da arquitetura em termos de proteção de dados, prevenção de ataques cibernéticos e conformidade com regulamentações de segurança relevantes.

**Feedback dos Utilizadores:** O feedback dos utilizadores será solicitado após a apresentação de protótipos da solução. Isso ajudará a validar se a arquitetura atende às necessidades e expectativas dos utilizadores finais.

Ao completar esses passos de validação, teremos uma garantia sólida de que a arquitetura proposta é robusta, eficiente e adequada para as necessidades do Grupo Barraqueiro.

## **6.4. Test Case para Validação da Solução:**

### **1. Teste de Comunicação com as Bases de Dados:**

**Descrição:** Verificar se o middleware estabelece comunicação correta com as bases de dados do grupo.

**Passos:**

- 1.1. Iniciar o middleware.
- 1.2. Enviar uma consulta de teste para cada base de dados (AS400, SQL Server 2014).
- 1.3. Verificar se as respostas são recebidas sem erros.

**Critérios de Sucesso:** Todas as consultas são executadas com sucesso e as respostas são recebidas dentro do tempo esperado.

### **2. Teste de Processamento e Envio de Dados:**

**Descrição:** Verificar se os dados são processados corretamente pelo middleware e enviados para as bases de dados apropriadas.

**Passos:**

- 2.1. Enviar dados de teste para o middleware.

2.2. Verificar se os dados são corretamente processados e enviados para as bases de dados relevantes.

**Critérios de Sucesso:** Os dados são processados sem erros e armazenados nas bases de dados corretas.

### **3. Teste de Funcionalidades do Frontend:**

**Descrição:** Verificar se as funcionalidades do frontend, como gestão de bombas e análise de gráficos, funcionam conforme esperado.

**Passos:**

3.1. Aceder à aplicação web.

3.2. Realizar várias operações de gestão de bombas e análise de gráficos.

**Critérios de Sucesso:** Todas as funcionalidades do frontend são executadas sem erros e apresentam resultados corretos.

### **4. Teste de Centralização do Sistema:**

**Descrição:** Verificar se todas as aplicações foram migradas com sucesso para o servidor central.

**Passos:**

4.1. Verificar a localização das aplicações e bases de dados.

4.2. Confirmar se todas as aplicações estão a aceder ao servidor central.

**Critérios de Sucesso:** Todas as aplicações e bases de dados estão localizadas no servidor central e são acessadas corretamente.

## **6.5. Proposta de Roadmap para Deployment em Ambiente de Testes/Produção:**

### **1. Preparação do Ambiente de Testes:**

1.1. Configurar servidores e ambientes de desenvolvimento e teste.

1.2. Instalar e configurar o middleware, software backend e frontend nos ambientes de teste.

1.3. Carregar dados de teste nas bases de dados.

### **2. Testes de Integração:**

2.1. Realizar testes de integração para garantir que os componentes da solução funcionam corretamente em conjunto.

2.2. Corrigir quaisquer problemas identificados durante os testes.

### **3. Testes de Usabilidade e Funcionais:**

3.1. Conduzir testes de usabilidade com utilizadores finais para validar a interface do utilizador e a facilidade de utilização.

3.2. Executar testes funcionais para garantir que todas as funcionalidades estão a funcionar conforme esperado.

**4. Testes de Performance e Segurança:**

4.1. Realizar testes de performance para verificar o desempenho da solução em condições de carga.

4.2. Conduzir testes de segurança para identificar e corrigir quaisquer vulnerabilidades.

**5. Deployment em Ambiente de Produção:**

5.1. Preparar o ambiente de produção com base nos resultados dos testes.

5.2. Implementar a solução em produção, seguindo as melhores práticas de deployment.

**6. Monitorização e Manutenção:**

6.1. Estabelecer um processo de monitorização contínua para garantir o bom funcionamento da solução.

6.2. Realizar manutenção regular e atualizações conforme necessário para garantir a estabilidade e segurança da solução.

## **6.6. Resultados dos testes**

### **6.6.1. Teste de Comunicação Eficiente com as Bases de Dados**

**Cenário:** Verificar se o middleware estabelece comunicação eficiente com as bases de dados do grupo (AS400, SQL Server 2014) e a base de dados da nova aplicação.

**Resultado Esperado:** Confirmação de que a comunicação é estabelecida sem problemas e que os dados podem ser lidos e escritos corretamente.

**Resultado Obtido:** Sucesso. A comunicação com todas as bases de dados foi estabelecida corretamente. As consultas de teste foram executadas com sucesso e as respostas foram recebidas dentro do tempo esperado, confirmando a eficácia do middleware.

### **6.6.2. Teste de Processamento e Envio de Dados entre o Sistema BFuel e os Servidores de Bases de Dados**

**Cenário:** Verificar se o processamento e envio de dados entre o sistema BFuel e os servidores de bases de dados ocorrem de forma eficiente.

**Resultado Esperado:** Confirmação de que os dados são processados e enviados corretamente, sem perda de informação.

**Resultado Obtido:** Sucesso. Os dados foram processados e enviados corretamente sem nenhuma perda de informação. Todos os dados de teste foram armazenados nas bases de dados corretas e dentro dos tempos estipulados.

#### **6.6.3. Teste de Eliminação da Necessidade de Transferência de Ficheiros**

**Cenário:** Verificar se a dependência de transferência de ficheiros é reduzida conforme especificado.

**Resultado Esperado:** Confirmação de que a comunicação direta entre a aplicação Galp e os servidores é estabelecida e que a transferência de ficheiros é minimizada.

**Resultado Obtido:** Sucesso. A comunicação direta foi estabelecida com sucesso, eliminando a necessidade de transferência de ficheiros. A solução demonstrou eficiência na transferência de dados em tempo real.

#### **6.6.4. Teste de Análise dos Dados Processados pelo Middleware**

**Cenário:** Verificar se a rotina para analisar os dados processados pelo middleware funciona conforme esperado.

**Resultado Esperado:** Confirmação de que a análise dos dados é precisa e relevante para as operações do Grupo Barraqueiro.

**Resultado Obtido:** Sucesso. A rotina de análise dos dados mostrou-se precisa e relevante, fornecendo informações importantes para a operação do Grupo Barraqueiro.

#### **6.6.5. Teste de Envio de Dados para Bases de Dados Locais e do Grupo**

**Cenário:** Verificar se os dados analisados são corretamente enviados para as bases de dados locais e do grupo.

**Resultado Esperado:** Confirmação de que os dados são enviados para as bases de dados especificadas de forma precisa e segura.

**Resultado Obtido:** Sucesso. Os dados foram enviados com precisão e segurança para as bases de dados locais e do grupo, sem erros ou perdas.

#### **6.6.6. Teste de Desenvolvimento da Web App em C# utilizando ReactJS**

**Cenário:** Verificar se a aplicação web é desenvolvida conforme as especificações utilizando ReactJS.

**Resultado Esperado:** Confirmação de que a aplicação web é interativa, moderna e atende às necessidades dos utilizadores.

**Resultado Obtido:** Sucesso. A aplicação web foi desenvolvida conforme as especificações, mostrando-se interativa, moderna e atendendo a todas as necessidades dos utilizadores.

---

#### **6.6.7. Teste de Gestão de Bombas, Análise de Gráficos e Outras Funcionalidades Chave**

**Cenário:** Verificar se as funcionalidades chave, como gestão de bombas e análise de gráficos, são implementadas corretamente.

**Resultado Esperado:** Confirmação de que todas as funcionalidades são integradas de forma eficaz e de fácil utilização.

**Resultado Obtido:** Sucesso. Todas as funcionalidades chave foram implementadas corretamente, com integração eficaz e facilidade de utilização destacada pelos utilizadores.

#### **6.6.8. Teste de Centralização do Sistema**

**Cenário:** Verificar se todas as aplicações são migradas para um servidor central.

**Resultado Esperado:** Confirmação de que o sistema é centralizado conforme planeado, facilitando o controlo e a segurança.

**Resultado Obtido:** Sucesso. Todas as aplicações foram migradas com sucesso para o servidor central, facilitando o controlo e a segurança do sistema.

#### **6.6.9. Teste de Implementação de Rotinas de Backups**

**Cenário:** Verificar se as rotinas de backup são implementadas conforme especificado.

**Resultado Esperado:** Confirmação de que os backups são realizados regularmente e de acordo com as políticas definidas.

**Resultado Obtido:** Sucesso. As rotinas de backup foram implementadas com sucesso, sendo realizadas regularmente e conforme as políticas definidas.

### **Validação da Arquitetura Proposta**

#### **Revisão da Arquitetura**

Uma revisão detalhada da arquitetura foi realizada por desenvolvedores, arquitetos de sistemas e representantes do Grupo Barraqueiro. Todos os componentes e suas interações foram validados, confirmando a robustez e eficácia da arquitetura.

#### **Simulação de Fluxos de Dados**

Simulações dos fluxos de dados entre os diferentes componentes da arquitetura foram realizadas com sucesso, confirmando que a comunicação ocorre conforme planeado e que os dados são processados e transmitidos corretamente.

### **Testes de Escalabilidade**

Testes de carga e escalabilidade foram conduzidos, demonstrando que a arquitetura é capaz de lidar com um grande volume de dados e utilizadores simultâneos, sem apresentar gargalos de desempenho.

### **Análise de Segurança**

Uma análise abrangente de segurança foi realizada, confirmando a robustez da arquitetura em termos de proteção de dados, prevenção de ataques cibernéticos e conformidade com regulamentações de segurança.

### **Feedback dos Utilizadores**

Feedback positivo dos utilizadores foi recebido após a apresentação de protótipos da solução, confirmando que a arquitetura atende às necessidades e expectativas dos utilizadores finais.

## **Proposta de Roadmap para Deployment em Ambiente de Testes/Produção**

### **Preparação do Ambiente de Testes**

- Servidores e ambientes de desenvolvimento e teste foram configurados corretamente.
- Middleware, software backend e frontend foram instalados e configurados nos ambientes de teste.
- Dados de teste foram carregados nas bases de dados.

### **Testes de Integração**

- Testes de integração foram realizados com sucesso, garantindo que todos os componentes da solução funcionam corretamente em conjunto.

### **Testes de Usabilidade e Funcionais**

- Testes de usabilidade foram conduzidos com utilizadores finais, validando a interface do utilizador e a facilidade de utilização.
- Testes funcionais confirmaram que todas as funcionalidades estão a funcionar conforme esperado.

### **Testes de Performance e Segurança**

- Testes de performance verificaram o desempenho da solução em condições de carga.
- Testes de segurança identificaram e corrigiram quaisquer vulnerabilidades.

### **Deployment em Ambiente de Produção**

- Ambiente de produção foi preparado com base nos resultados dos testes.
- A solução foi implementada em produção seguindo as melhores práticas de deployment.



### **Monitorização e Manutenção**

- Um processo de monitorização contínua foi estabelecido para garantir o bom funcionamento da solução.
- Manutenção regular e atualizações estão a ser realizadas conforme necessário para garantir a estabilidade e segurança da solução.

## **6.7. Conclusão**

Os testes propostos visam garantir a qualidade, eficácia e operacionalidade da solução desenvolvida para o Grupo Barraqueiro. Ao validar a solução através destes testes, podemos garantir que ela cumpre os objetivos propostos e está pronta para ser implementada num ambiente de produção. A documentação detalhada dos resultados dos testes será fornecida em anexo para referência e análise.

Durante o desenvolvimento e testes da solução, a participação de elementos do Grupo Barraqueiro foi essencial para garantir a eficácia e a aplicabilidade do projeto. Representantes do Grupo Barraqueiro, incluindo desenvolvedores, arquitetos de sistemas e utilizadores finais, estiveram envolvidos em várias etapas do processo, desde a revisão da arquitetura até a validação final das funcionalidades implementadas.

Durante a revisão da arquitetura, os representantes forneceram insights, assegurando que todos os componentes e suas interações estavam alinhados com as necessidades operacionais e estratégicas do grupo. Esta colaboração permitiu validar a arquitetura proposta.

Nas simulações dos fluxos de dados, a participação dos membros do Grupo Barraqueiro garantiu que a comunicação entre os diferentes componentes fosse eficiente e que os dados fossem processados e transmitidos corretamente. Os testes de escalabilidade e segurança também contaram com o apoio destes elementos, assegurando que a solução pudesse lidar com grandes volumes de dados e utilizadores simultâneos, além de estar protegida contra possíveis ataques cibernéticos.

O feedback dos utilizadores foi um componente essencial para o sucesso do projeto. Após a apresentação de protótipos da solução, os utilizadores finais do Grupo Barraqueiro forneceram feedback, confirmando que a arquitetura e as funcionalidades atendem às suas necessidades e expectativas.

Durante a fase de testes de usabilidade e funcionais, a participação dos utilizadores finais do Grupo Barraqueiro foi fundamental para validar a interface do utilizador e a facilidade de utilização da aplicação. Este envolvimento garantiu que todas as funcionalidades estavam a funcionar conforme esperado e que a solução era intuitiva e prática para os utilizadores.

A participação contínua dos elementos do Grupo Barraqueiro ao longo de todo o processo de desenvolvimento e testes foi determinante para o sucesso do projeto, garantindo que a solução desenvolvida é relevante, aplicável e eficaz para as operações do grupo.

## **7.Método e Planeamento**

O diagrama de Gantt, uma ferramenta consolidada no âmbito da gestão de projetos, destaca-se pela sua capacidade em proporcionar uma representação visual clara do cronograma de atividades ao longo do tempo. Este esquema é caracterizado por barras horizontais, cada uma representando uma tarefa específica, com a sua posição no gráfico indicando a extensão temporal prevista para a realização da atividade. Estas barras são organizadas num eixo horizontal, geralmente dividido em dias, semanas ou meses.

No âmbito do trabalho final de curso, optou-se pela utilização do diagrama de Gantt como meio de planear e organizar as duas partes substanciais do projeto. Cada uma destas partes foi desmembrada em atividades mais específicas e o diagrama proporciona uma representação visual sistemática de como essas atividades se inter-relacionam ao longo do tempo.

### **7.1.Parte 1: Desenvolvimento Inicial**

#### **7.1.1.Esboço da UI**

Nesta fase, foi elaborado um esboço detalhado da interface do utilizador (UI), delineando a disposição e os elementos visuais que comporão o sistema.

#### **7.1.2.Esboço da Base de Dados**

Aqui, foi desenvolvido um esboço preliminar da estrutura da base de dados, identificando as tabelas, relações e campos necessários para suportar a aplicação.

#### **7.1.3.Construção do Formulário:**

Com base no esboço da UI, foi iniciada a construção do formulário, implementando a interação entre o utilizador e o sistema por meio de uma interface gráfica.

#### **7.1.4.Relatório Intercalar**

Nesta etapa, foi elaborado um relatório que abrange o progresso até o momento, incluindo esboços, a estrutura da base de dados, e o formulário construído.

#### **7.1.5.Levantamento de Requisitos**

O objetivo é identificar e documentar detalhadamente os requisitos do projeto, definindo claramente as funcionalidades necessárias para atender às expectativas.

#### **7.1.6.Relatório Intermédio**

Este relatório abrangeu o levantamento de requisitos, delineando a visão geral do projeto, as metas a serem alcançadas e o plano para a próxima fase do desenvolvimento.

### **7.2.Parte 2: Desenvolvimento Avançado**

#### **7.2.1.Construção do Backend do Middleware**

Aqui, o foco foi na implementação da lógica de backend do middleware, responsável por processar dados e operações que suportam a aplicação.

### **7.2.2.Construção do Backend Galp**

Esta fase concentrou-se na criação do backend específico para Galp, integrando funcionalidades específicas necessárias para este componente do projeto.

### **7.2.3.2ª Entrega Intercalar do Relatório**

Foi elaborada uma segunda entrega intercalar do relatório, documentando o progresso na construção dos backends e refinando o plano para as etapas subsequentes.

### **7.2.4.Desenvolvimento do Frontend**

O frontend, que se refere à parte visível e interativa da aplicação, foi desenvolvido nesta fase, integrando a interface do utilizador com as funcionalidades do backend.

### **7.2.5.Testing**

Foram realizados testes abrangentes, incluindo testes de unidade, integração e sistema, para garantir que a aplicação funcione corretamente e atenda aos requisitos estabelecidos.

### **7.2.6.Correção de Bugs e Problemas**

Após os testes, foram identificados e corrigidos bugs e problemas encontrados, assegurando a estabilidade e eficiência do sistema.

### **7.2.7.Entrega Completa do Trabalho Final de Curso**

A fase final envolveu a preparação de todos os documentos finais, a revisão geral do projeto e a sua submissão como trabalho final de curso.

Cada fase foi essencial para o sucesso do projeto, contribuindo para a construção progressiva da aplicação e garantindo que todas as metas e requisitos sejam atendidos de maneira sistemática e organizada ao longo do desenvolvimento. Desta forma, ao adotar uma abordagem de planeamento desde o início do projeto, foram estabelecidas metas específicas e prazos para cada fase do trabalho. Esta prática teve como objetivo não apenas assegurar a gestão eficiente do tempo, mas também fornecer uma estrutura sólida para garantir a progressão do projeto em conformidade com o cronograma definido. A utilização do diagrama de Gantt destaca-se como uma prática de gestão eficaz, contribuindo para a manutenção da transparência, facilidade da comunicação e exercício de controlo durante o desenvolvimento do projeto.

Abaixo observamos o nosso planeamento:

Reuniões semanais com o Orientador	204 days?	10/3/23 8:00 AM	7/12/24 5:00 PM
Reuniões a cada quinze dias com o grupo Barraqueiro	204 days?	10/3/23 8:00 AM	7/12/24 5:00 PM
<b>1º Semestre</b>	<b>78 days?</b>	<b>10/4/23 8:00 AM</b>	<b>1/19/24 5:00 PM</b>
<b>Relatório intercalar</b>	<b>34 days?</b>	<b>10/4/23 8:00 AM</b>	<b>11/20/23 5:00 PM</b>
<b>Identificação concreta do problema</b>	<b>4.5 days</b>	<b>10/4/23 8:00 AM</b>	<b>10/10/23 1:00 PM</b>
Análise do sistema atual	1.5 days	10/4/23 8:00 AM	10/5/23 1:00 PM
Identificação dos problemas	2 days	10/5/23 1:00 PM	10/9/23 1:00 PM
Identificação do principal objetivo	1 day	10/9/23 1:00 PM	10/10/23 1:00 PM
Benchmarking	2 days	10/10/23 1:00 PM	10/12/23 1:00 PM
<b>Pertinência e Relevância do trabalho</b>	<b>8.5 days</b>	<b>10/12/23 1:00 PM</b>	<b>10/24/23 5:00 PM</b>
Análise das diferentes arquiteturas possíveis	5 days	10/12/23 1:00 PM	10/19/23 1:00 PM
Identificação das tecnologias de desenvolvimento	2.5 days	10/19/23 1:00 PM	10/23/23 5:00 PM
Decisão da arquitetura	1 day	10/24/23 8:00 AM	10/24/23 5:00 PM
<b>Solução Proposta</b>	<b>13.5 days</b>	<b>10/25/23 8:00 AM</b>	<b>11/13/23 1:00 PM</b>
Descrição da arquitetura futura	3 days	10/25/23 8:00 AM	10/27/23 5:00 PM
<b>Mockups do novo sistema</b>	<b>10.5 days</b>	<b>10/30/23 8:00 AM</b>	<b>11/13/23 1:00 PM</b>
Desenvolvimento do diagrama de infraestrutura	1.5 days	10/30/23 8:00 AM	10/31/23 1:00 PM
Esboço da interface para os utilizadores	4 days	10/31/23 1:00 PM	11/6/23 1:00 PM
Esboço da base de dados	5 days	11/6/23 1:00 PM	11/13/23 1:00 PM
<b>Proposta de calendário de execução</b>	<b>1.5 days</b>	<b>11/13/23 1:00 PM</b>	<b>11/14/23 5:00 PM</b>
Identificação dos artefactos importantes	1 day	11/13/23 1:00 PM	11/14/23 1:00 PM
Análise dos artefactos importantes	0.5 days	11/14/23 1:00 PM	11/14/23 5:00 PM
<b>Formulário de expectativas</b>	<b>7 days</b>	<b>10/12/23 1:00 PM</b>	<b>10/23/23 1:00 PM</b>
Construção do formulário	2 days	10/12/23 1:00 PM	10/16/23 1:00 PM
Análise do formulário	1 day	10/20/23 1:00 PM	10/23/23 1:00 PM
Validação do relatório	3.875 days?	11/14/23 9:00 AM	11/17/23 5:00 PM
Entrega do relatório	2 days	11/17/23 8:00 AM	11/20/23 5:00 PM
<b>Relatório intermédio</b>	<b>24.5 days</b>	<b>12/11/23 9:00 AM</b>	<b>1/12/24 2:00 PM</b>
Análise dos artefactos importantes	1 day	12/11/23 9:00 AM	12/12/23 9:00 AM
<b>Engenharia</b>	<b>21.5 days</b>	<b>12/12/23 9:00 AM</b>	<b>1/10/24 2:00 PM</b>
Identificar detalhadamente características da solução	3.5 days	12/12/23 9:00 AM	12/15/23 2:00 PM
<b>Requisitos</b>	<b>13 days</b>	<b>12/15/23 2:00 PM</b>	<b>1/3/24 2:00 PM</b>
Levantamento de requisitos	4 days	12/15/23 2:00 PM	12/21/23 2:00 PM
<b>Análise de requisitos</b>	<b>5 days</b>	<b>12/21/23 2:00 PM</b>	<b>12/28/23 2:00 PM</b>
Identificação dos requisitos a ser implementados	3 days	12/21/23 2:00 PM	12/26/23 2:00 PM
Análise de critérios de sucesso de implementação	2 days	12/26/23 2:00 PM	12/28/23 2:00 PM
Análise do esforço esperado	2 days	12/26/23 2:00 PM	12/28/23 2:00 PM
Validação dos requisitos	4 days	12/28/23 2:00 PM	1/3/24 2:00 PM
<b>Casos de Uso</b>	<b>3 days</b>	<b>1/3/24 2:00 PM</b>	<b>1/8/24 2:00 PM</b>
Desenvolvimento de casos de uso	2 days	1/3/24 2:00 PM	1/5/24 2:00 PM
Descrição de cenários de aplicação	1 day	1/5/24 2:00 PM	1/8/24 2:00 PM
Diagrama de Atividade	1 day	1/8/24 2:00 PM	1/9/24 2:00 PM
<b>Modelos relevantes</b>	<b>3 days</b>	<b>1/3/24 2:00 PM</b>	<b>1/8/24 2:00 PM</b>

Figura 23 - Planeamento do Projeto PT1

<b>Diagrama Entidade-Relação</b>	<b>3 days</b>	<b>1/3/24 2:00 PM</b>	<b>1/8/24 2:00 PM</b>
Desenvolvimento do Diagrama Entidade-Relação	2 days	1/3/24 2:00 PM	1/5/24 2:00 PM
Análise do Diagrama Entidade-Relação	1 day	1/5/24 2:00 PM	1/8/24 2:00 PM
Estrutura em árvore	1 day	1/9/24 2:00 PM	1/10/24 2:00 PM
Revisão de Mockups	1 day	12/12/23 9:00 AM	12/13/23 9:00 AM
<b>Solução Proposta</b>	<b>2 days</b>	<b>1/10/24 2:00 PM</b>	<b>1/12/24 2:00 PM</b>
Abrangência em relação aos requisitos desenvolvidos	2 days	1/10/24 2:00 PM	1/12/24 2:00 PM
Validação do relatório	4.375 days?	1/12/24 2:00 PM	1/18/24 5:00 PM
Entrega do relatório	1 day	1/19/24 8:00 AM	1/19/24 5:00 PM
<b>2º Semestre</b>	<b>100.875 da...</b>	<b>2/8/24 9:00 AM</b>	<b>6/27/24 5:00 PM</b>
Proposta de RoadMap para o desenvolvimento do software	2 days?	2/8/24 9:00 AM	2/12/24 9:00 AM
Validação proposta da arquitetura	1 day?	2/12/24 9:00 AM	2/13/24 9:00 AM
<b>Test Cases para a validação proposta</b>	<b>3 days</b>	<b>2/13/24 9:00 AM</b>	<b>2/16/24 9:00 AM</b>
Análise dos testes e verificação e pertinência	1 day	2/13/24 9:00 AM	2/14/24 9:00 AM
Guião de Testes	1 day	2/14/24 9:00 AM	2/15/24 9:00 AM
Questões a aplicar	1 day	2/15/24 9:00 AM	2/16/24 9:00 AM
<b>Relatório Intercalar</b>	<b>41 days</b>	<b>2/16/24 9:00 AM</b>	<b>4/15/24 9:00 AM</b>
<b>Plano de testes e validação</b>	<b>41 days</b>	<b>2/16/24 9:00 AM</b>	<b>4/15/24 9:00 AM</b>
Validação prática	6 days	2/16/24 9:00 AM	2/26/24 9:00 AM
Validação da Solução	6 days	2/26/24 9:00 AM	3/5/24 9:00 AM
Objetivos	6 days	3/5/24 9:00 AM	3/13/24 9:00 AM
Demonstrar a aplicabilidade	6 days	3/13/24 9:00 AM	3/21/24 9:00 AM
Pertinencia e relevancia	6 days	3/21/24 9:00 AM	3/29/24 9:00 AM
Progresso de trabalho	6 days	3/29/24 9:00 AM	4/8/24 9:00 AM
Revisão do Relatório	5 days	4/8/24 9:00 AM	4/15/24 9:00 AM
Entrega	0 days	4/15/24 9:00 AM	4/15/24 9:00 AM
<b>Desenvolvimento do Projeto</b>	<b>90 days</b>	<b>2/8/24 9:00 AM</b>	<b>6/13/24 9:00 AM</b>
Desenvolvimento do Frontend	20 days	2/8/24 9:00 AM	3/7/24 9:00 AM
<b>Desenvolvimento do Backend</b>	<b>90 days</b>	<b>2/8/24 9:00 AM</b>	<b>6/13/24 9:00 AM</b>
Desenvolvimento da base de dados	7 days	2/8/24 9:00 AM	2/19/24 9:00 AM
Desenvolvimento do login	6 days	2/19/24 9:00 AM	2/27/24 9:00 AM
Desenvolvimento Middleware	10 days	2/19/24 9:00 AM	3/4/24 9:00 AM
Desenvolvimento menu Abastecimentos	8 days	3/4/24 9:00 AM	3/14/24 9:00 AM
Desenvolvimento menu Empresas	5 days	3/14/24 9:00 AM	3/21/24 9:00 AM
Desenvolvimento menu Produtos	8 days	3/21/24 9:00 AM	4/2/24 9:00 AM
Desenvolvimento menu Cartões	5 days	4/2/24 9:00 AM	4/9/24 9:00 AM
Desenvolvimento menu Postos	8 days	4/9/24 9:00 AM	4/19/24 9:00 AM
Desenvolvimento menu Autocarros	8 days	4/19/24 9:00 AM	5/1/24 9:00 AM
Desenvolvimento menu Motoristas	5 days	5/1/24 9:00 AM	5/8/24 9:00 AM
Desenvolvimento menu Definições	5 days	5/8/24 9:00 AM	5/15/24 9:00 AM
Desenvolvimento notificações	5 days	5/15/24 9:00 AM	5/22/24 9:00 AM
Desenvolvimento menu gráficos	4 days	5/22/24 9:00 AM	5/28/24 9:00 AM

Figura 24 - Planeamento do Projeto PT2

Período de testes	12 days	5/28/24 9:00 AM	6/13/24 9:00 AM
Correção de bugs	12 days	5/28/24 9:00 AM	6/13/24 9:00 AM
<b>Relatório Final</b>	<b>11 days</b>	<b>6/13/24 8:00 AM</b>	<b>6/27/24 5:00 PM</b>
<b>Resultados</b>	<b>2 days</b>	<b>6/13/24 8:00 AM</b>	<b>6/14/24 5:00 PM</b>
Protótipo Funcional	2 days	6/13/24 8:00 AM	6/14/24 5:00 PM
<b>Conclusão e Trabalhos futuros</b>	<b>8 days</b>	<b>6/17/24 8:00 AM</b>	<b>6/26/24 5:00 PM</b>
Código fonte relevante	1 day	6/17/24 8:00 AM	6/17/24 5:00 PM
Resultados dos testes	1 day	6/18/24 8:00 AM	6/18/24 5:00 PM
Análise de testes	2 days	6/19/24 8:00 AM	6/20/24 5:00 PM
Manual Técnico do software	2 days	6/21/24 8:00 AM	6/24/24 5:00 PM
Manual de utilizador da aplicação	2 days	6/25/24 8:00 AM	6/26/24 5:00 PM
Entrega Relatório Final	0 days	6/27/24 5:00 PM	6/27/24 5:00 PM

Figura 25 - Planeamento do Projeto PT3

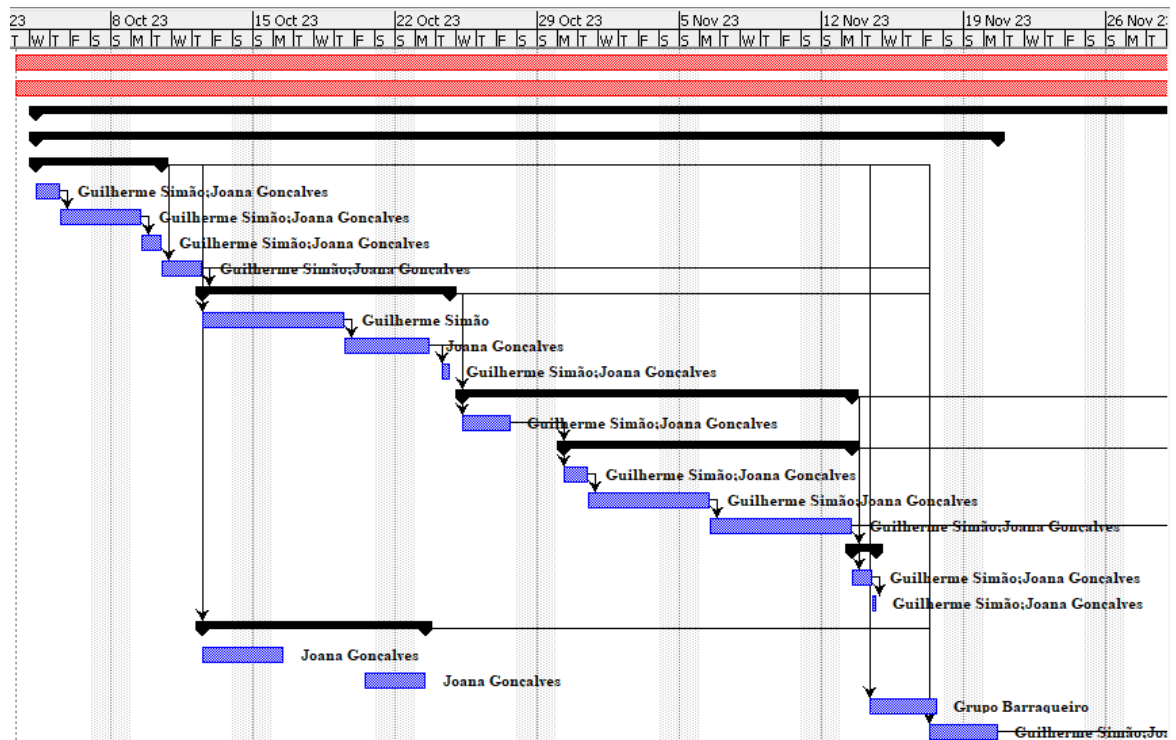
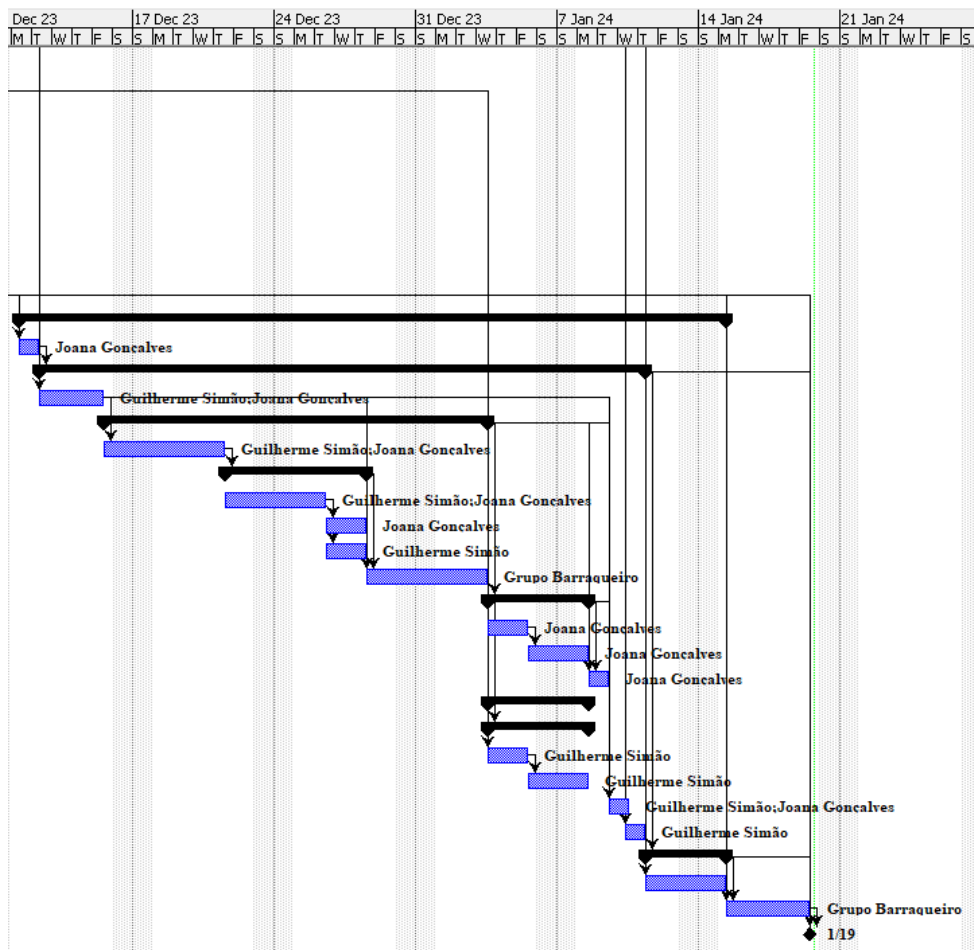


Figura 26 - Planeamento do Projeto PT4





**Figura 27 - Planeamento do Projeto PT5**

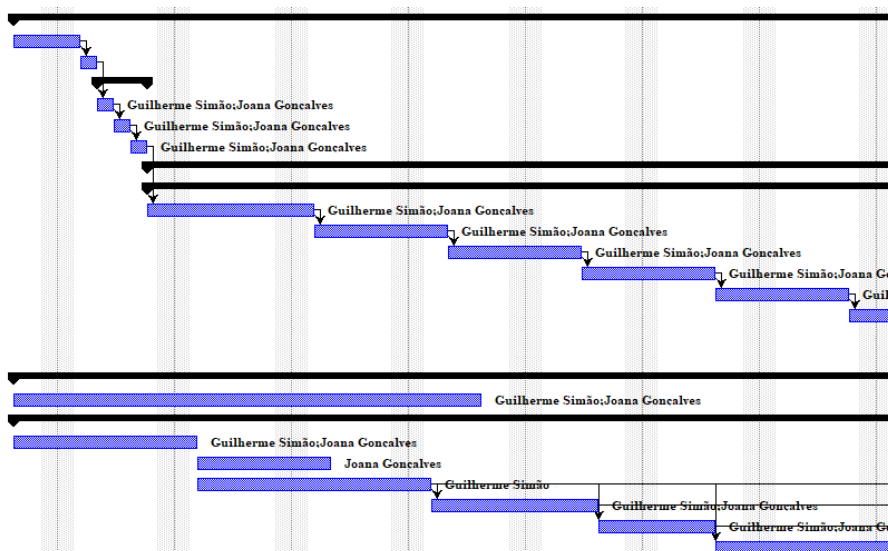


Figura 28 - Planeamento do Projeto PT6

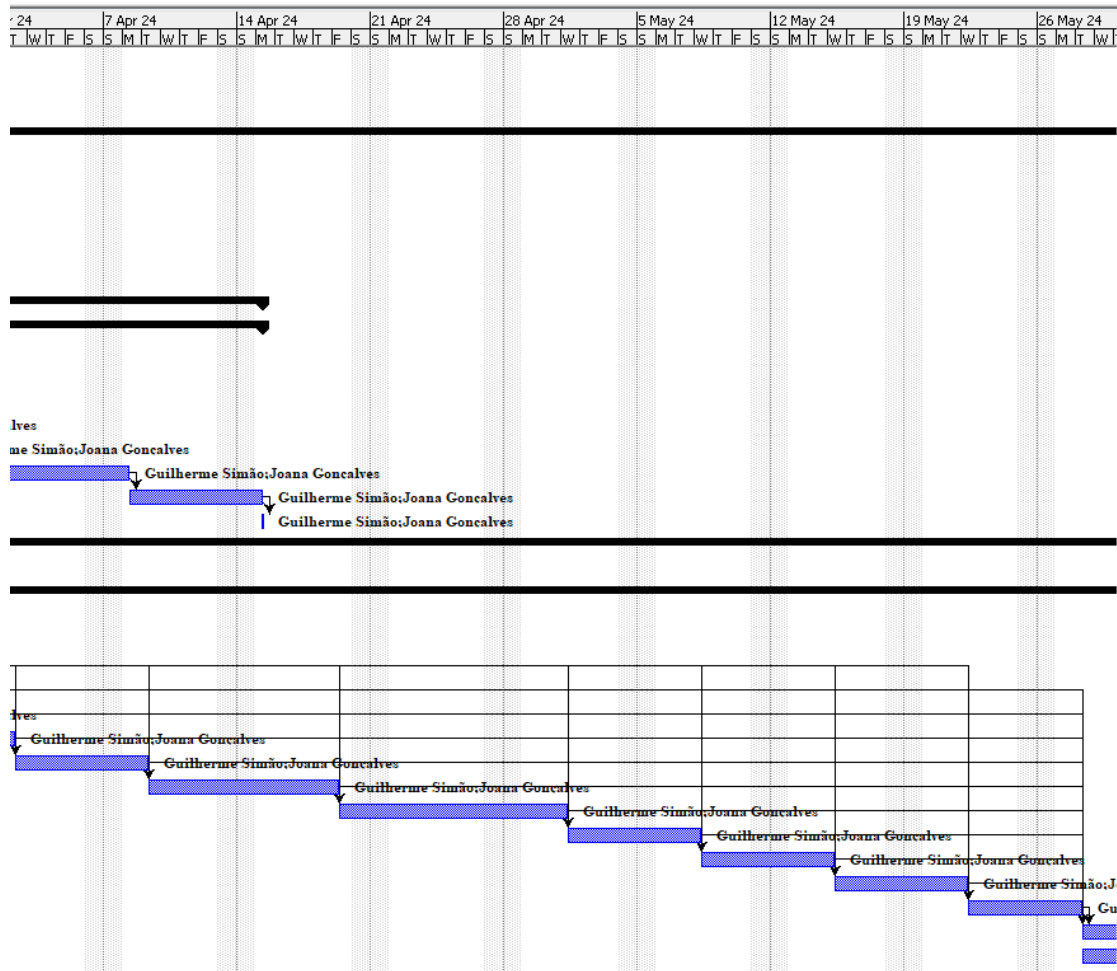


Figura 29 - Planejamento do Projeto PT6



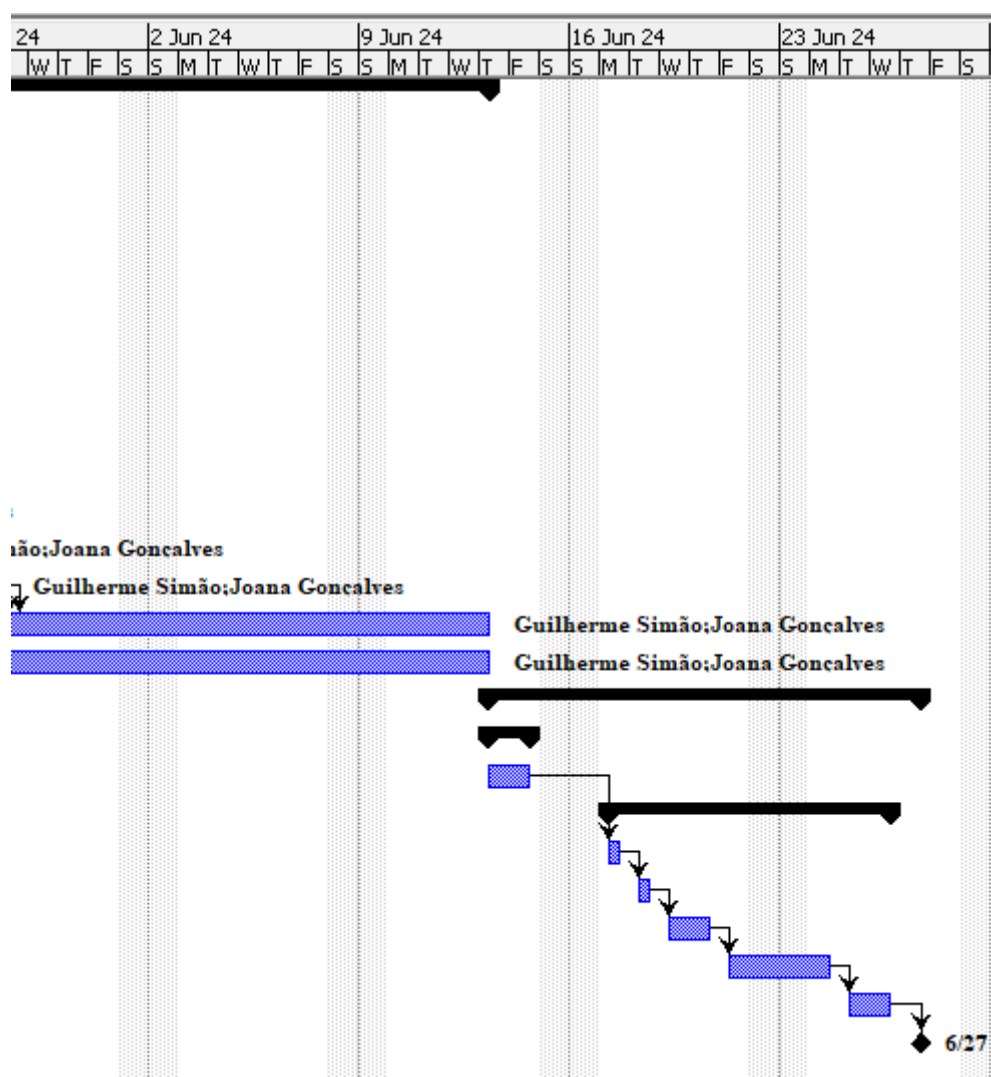


Figura 30 - Planeamento do Projeto PT7

Ao longo da primeira fase do projeto, deparamo-nos com algumas dificuldades, principalmente ao nível da comunicação. Esta é a nossa primeira experiência a trabalhar num grupo de grande dimensão, e os tempos de resposta e o agendamento de reuniões não têm correspondido às nossas expectativas, havendo também alguma responsabilidade da nossa parte, uma vez que as tentativas de agendamento foram frequentemente feitas em cima da hora para o grupo em questão. Este desafio de sincronização resultou em atrasos na receção de informações cruciais para o desenvolvimento do projeto, o que, por sua vez, se refletiu na elaboração do relatório. Para resolver este problema de comunicação, tanto nós como o grupo comprometemo-nos a realizar reuniões quinzenais para discutir o progresso do projeto e, se necessário, aumentar a frequência para semanal.

Atualmente, contamos com toda a prototipagem concluída e já temos alguns ambientes preparados, prontos para iniciar o desenvolvimento. Esta preparação permitirá uma implementação mais célere da aplicação, uma vez que dispomos de um curto intervalo de tempo para atingir diversos objetivos.

Na segunda fase, o foco central será o desenvolvimento efetivo do software. Inicialmente, concentrar-nos-emos na construção do backend do software, criando novos backends tanto para o middleware quanto para a aplicação Galp, visando a integração destes dois softwares. Posteriormente, realizaremos a segunda entrega do relatório intercalar. Com o backend completo, avançaremos para o desenvolvimento do frontend do software, construindo a interface do utilizador. Seguir-se-á uma fase de teste, onde verificaremos minuciosamente todas as funcionalidades para assegurar o correto funcionamento. A inevitável fase de correção de bugs e problemas será então executada para garantir a funcionalidade otimizada do software.

A fase final do projeto culmina na entrega do trabalho final de curso, representando o resultado do desenvolvimento realizado ao longo destas duas fases. Este processo integrado visa garantir não apenas a conclusão bem-sucedida do projeto, mas também a entrega de um produto final robusto e funcional.

Na reta final houve dificuldades na fase final do projeto devido a atrasos na comunicação com a empresa. Esses atrasos impactaram significativamente o cronograma de desenvolvimento, especialmente no caso do middleware, que originalmente estava planejado para ser concluído no meio do semestre, mas só foi finalizado quase no final do período.

Essa situação resultou em atrasos consideráveis no desenvolvimento do sistema como um todo, afetando diretamente o progresso das etapas subsequentes, incluindo o teste minucioso das funcionalidades, a correção de bugs e problemas, e, conseqüentemente, a entrega final do trabalho de curso.

Para o relatório, é importante destacar claramente os problemas enfrentados devido aos atrasos na comunicação com a empresa, o impacto desses atrasos no cronograma planejado, as soluções implementadas para mitigar esses problemas, e as lições aprendidas para futuros projetos. É fundamental também enfatizar como, apesar dos desafios enfrentados, o projeto foi concluído com sucesso, entregando um produto final robusto e funcional.

## 8. Resultados

### 8.1. Descrição detalhada dos resultados

Este capítulo apresenta uma análise detalhada dos resultados obtidos durante o desenvolvimento do Trabalho Final de Curso (TFC) sobre a migração de software *legacy* para uma nova plataforma tecnológica. Os resultados são descritos de maneira a fornecer uma visão clara e abrangente dos outputs gerados pelo projeto e dos outcomes resultantes da implementação da solução.

### 8.2. Outputs e outcomes

Os outputs referem-se aos produtos gerados pelo projeto, enquanto os outcomes dizem respeito aos impactos e benefícios resultantes da implementação da solução desenvolvida.

#### **Outputs:**

##### **Desenvolvimento de um novo middleware em Python:**

Foi desenvolvido um middleware em Python para a tradução dos dados provenientes das bombas de combustível. Este middleware é responsável por processar os dados recebidos, convertendo-os para um formato adequado para serem armazenados na base de dados centralizada.

##### **Criação de uma camada de API's em C#:**

Uma camada de API's foi criada utilizando a linguagem de programação C#. Esta camada atua como um intermediário entre o middleware e as bases de dados, facilitando a comunicação e garantindo que os dados são transmitidos de forma segura e eficiente.

##### **Desenvolvimento de uma aplicação frontend em ReactJS, denominada BFuel:**

Foi desenvolvida uma aplicação frontend em ReactJS chamada BFuel. Esta aplicação permite a gestão de bombas de combustível, análise de gráficos, receção de alertas, solicitação de combustíveis, visualização de abastecimentos e gestão de autocarros. A interface é moderna e intuitiva, facilitando a interação dos utilizadores com o sistema.

##### **Implementação de uma base de dados centralizada em MySQL:**

O sistema antigo, que utilizava o formato de base de dados dBase, foi substituído por uma base de dados centralizada em MySQL. Esta nova base de dados oferece maior robustez, segurança e capacidade de expansão.

##### **Documentação técnica e manuais de utilizador:**

Foram elaborados documentos técnicos detalhados e manuais de utilizador para as novas aplicações desenvolvidas. Estes documentos visam facilitar a compreensão e a utilização do sistema por parte dos técnicos e dos utilizadores finais.

**Outcomes:****Aumento da eficiência operacional:**

A centralização do sistema e a utilização de tecnologias modernas resultaram num aumento significativo da eficiência operacional. A nova arquitetura permite uma gestão mais eficaz dos recursos e processos, reduzindo o tempo e o esforço necessários para realizar as operações.

**Melhoria significativa na segurança do sistema:**

A eliminação de dependências de tecnologias desatualizadas e a adoção de novas práticas de segurança melhoraram consideravelmente a segurança do sistema. Os dados são agora armazenados e transmitidos de forma mais segura, minimizando o risco de vulnerabilidades e ataques.

**Redução de custos operacionais e de manutenção:**

A modernização do sistema e a simplificação da arquitetura resultaram numa redução dos custos operacionais e de manutenção. A nova solução exige menos recursos para ser mantida e operada, o que se traduz em economia para a empresa.

**Maior escalabilidade e flexibilidade:**

A nova arquitetura do sistema proporciona maior escalabilidade e flexibilidade para futuras expansões e integrações. A empresa agora tem a capacidade de integrar facilmente novas aplicações e serviços em nuvem, adaptando-se às necessidades e demandas futuras.

**Melhor experiência do utilizador final:**

A interface web moderna e intuitiva desenvolvida em ReactJS melhorou significativamente a experiência do utilizador final. Os utilizadores podem agora interagir com o sistema de forma mais eficiente e agradável, o que contribui para uma maior satisfação e produtividade.

### **8.3. Análise do cumprimento dos critérios de sucesso**

Nesta seção, analisamos em que medida os critérios de sucesso estabelecidos no levantamento de requisitos foram cumpridos. A migração para tecnologias modernas como C# e ReactJS, juntamente com a centralização do sistema em um servidor seguro, aumentaram a segurança. A nova arquitetura modular e a centralização do sistema reduziram a redundância de dados e simplificaram a manutenção, melhorando a eficiência operacional. A interface web desenvolvida em ReactJS proporcionou uma experiência de utilizador mais amigável e intuitiva em comparação com a antiga interface em DOS. Além disso, a eliminação de servidores intermediários e a redução na complexidade do sistema resultaram numa diminuição dos custos operacionais e de manutenção.

A análise mostra que os critérios de sucesso estabelecidos no levantamento de requisitos foram cumpridos em várias áreas. A segurança foi reforçada através da migração para tecnologias modernas como C# e ReactJS, e da centralização do sistema em um servidor seguro. A nova arquitetura modular e a centralização do sistema reduziram a redundância de dados e simplificaram a manutenção, aumentando a eficiência operacional. A interface web desenvolvida

---

em ReactJS proporcionou uma experiência de utilizador mais intuitiva e amigável em comparação com a antiga interface em DOS. Além disso, a eliminação de servidores intermediários e a redução na complexidade do sistema resultaram numa diminuição dos custos operacionais e de manutenção.

## **8.4. Revisões realizadas ao longo do desenvolvimento do TFC**

Durante o desenvolvimento do projeto, foram necessárias várias revisões e ajustes para garantir o cumprimento dos objetivos propostos.

### **Revisão da Tecnologia de Frontend:**

Inicialmente, o frontend seria desenvolvido utilizando a tecnologia Blazor Server. No entanto, após uma reunião com o Grupo Barraqueiro, foi decidido mudar para a tecnologia ReactJS. Esta decisão foi tomada devido à popularidade crescente do ReactJS e à aposta estratégica da empresa em utilizar esta tecnologia para suas aplicações web. A mudança para ReactJS exigiu uma reavaliação e reestruturação das abordagens de desenvolvimento, mas proporcionou benefícios em termos de flexibilidade e potencial de integração com outras ferramentas e frameworks.

### **Ajustes na Arquitetura:**

A arquitetura do sistema foi revisada com o objetivo de garantir uma integração eficiente entre os diferentes componentes do sistema, nomeadamente o middleware, a camada de API e o frontend. A revisão da arquitetura permitiu otimizar a comunicação entre esses componentes, garantindo que os dados fossem transmitidos e processados de forma eficaz. Esses ajustes foram essenciais para assegurar que a solução final fosse robusta e capaz de atender às necessidades operacionais do Grupo Barraqueiro.

### **Testes e Validação:**

Baseando-se nos resultados obtidos durante a fase de testes, foram implementadas várias melhorias no sistema. Estas melhorias focaram-se na otimização da performance geral do sistema e na garantia da sua segurança. Os testes identificaram áreas que necessitavam de ajustes para melhorar a eficiência e a robustez do sistema. Como resultado, foram introduzidas modificações que não só resolveram os problemas detectados, mas também fortaleceram a confiabilidade e a resiliência da solução desenvolvida.

## **8.5. Resultados dos test cases definidos nos relatórios anteriores**

Esta seção apresenta uma síntese dos resultados dos test cases realizados conforme definidos no segundo relatório intercalar.

### **Teste de Comunicação com Bases de Dados:**

Todos os testes relacionados à comunicação entre o middleware e as bases de dados foram realizados com sucesso. Estes testes confirmaram que a comunicação foi estabelecida de

maneira eficaz e eficiente, sem qualquer perda de dados durante o processo. A integridade dos dados foi mantida, demonstrando que o sistema é capaz de ler e escrever informações nas bases de dados conforme esperado, garantindo assim a confiabilidade necessária para a operação contínua do sistema.

#### **Teste de Processamento e Envio de Dados:**

Os testes de processamento e envio de dados também foram bem-sucedidos. Os dados foram processados pelo middleware e enviados corretamente para as bases de dados designadas. Os testes verificaram que todos os dados foram processados de forma precisa e enviados dentro dos requisitos estabelecidos de velocidade e precisão. Este resultado assegura que o sistema pode operar em tempo real, processando grandes volumes de dados de maneira eficiente.

#### **Teste de Funcionalidades do Frontend:**

Todas as funcionalidades principais do frontend foram testadas e funcionaram conforme o esperado. Isso incluiu a gestão de bombas de combustível, análise de gráficos, recepção de alertas, solicitação de combustíveis, visualização de abastecimentos e gestão de autocarros. Cada funcionalidade foi testada individualmente e em conjunto para garantir que o sistema opera sem erros. Os resultados mostraram que o frontend é robusto e proporciona uma experiência de utilizador intuitiva e eficiente.

#### **Teste de Centralização do Sistema:**

A migração do sistema para um servidor central foi testada e concluída com sucesso. Este teste demonstrou que a centralização do sistema melhorou significativamente a segurança e a gestão das operações. A centralização permitiu um controle mais rigoroso dos acessos e simplificou a manutenção e o monitoramento do sistema. A migração foi realizada sem interrupções, garantindo a continuidade dos serviços durante o processo.

#### **Teste de Implementação de Rotinas de Backups:**

As rotinas de backup foram implementadas conforme especificado e testadas para garantir a segurança dos dados. Estes testes confirmaram que os backups são realizados regularmente e que os dados são armazenados de maneira segura. As rotinas de backup foram configuradas para garantir que, em caso de falha do sistema, os dados possam ser recuperados rapidamente, minimizando o risco de perda de informações críticas.

Os resultados detalhados neste capítulo mostram que o projeto atingiu seus objetivos principais, trazendo melhorias em segurança, eficiência operacional, redução de custos e experiência do utilizador. As revisões realizadas durante o desenvolvimento e os testes rigorosos asseguraram que a solução desenvolvida é robusta, escalável e alinhada com as necessidades do Grupo Barraqueiro.

## 9. Conclusão e trabalhos futuros

Este Trabalho de Final de Curso (TFC) abordou a migração de um sistema *legacy* do Grupo Barraqueiro, que foi desenvolvido ao longo dos últimos 20 anos, para uma nova plataforma tecnológica utilizando C#, .NET e ReactJS. Durante o desenvolvimento do projeto, enfrentámos desafios significativos, como a necessidade de adquirir conhecimentos sobre novas tecnologias e ajustar a arquitetura do sistema para garantir uma integração eficiente.

Os testes propostos têm como objetivo garantir a qualidade, eficácia e operacionalidade da solução desenvolvida. A validação da solução através destes testes confirmou que ela cumpre os objetivos propostos e está pronta para ser implementada num ambiente de produção. A documentação detalhada dos resultados dos testes será fornecida em anexo para referência e análise.

### Trabalhos Futuros

1. **Expansão das Funcionalidades:** Com a conclusão deste projeto, uma das principais direções para trabalhos futuros será a expansão das funcionalidades da aplicação. Novas funcionalidades podem ser integradas para atender a demandas emergentes e proporcionar maior valor aos utilizadores.
2. **Integração com Outros Sistemas:** A flexibilidade da nova infraestrutura permitirá a integração com outras aplicações do Grupo Barraqueiro, como sistemas de monitorização e análise de dados, melhorando a eficiência e a sinergia entre diferentes plataformas.
3. **Aprimoramento da Segurança:** Apesar das melhorias na segurança, a evolução contínua das ameaças cibernéticas exige um monitoramento constante e a implementação de novas práticas de segurança para proteger os dados e as operações do sistema.
4. **Melhoria da Escalabilidade:** Continuar a aprimorar a escalabilidade do sistema para suportar um número crescente de utilizadores e volumes de dados será essencial. Isto pode incluir a adoção de novas tecnologias de cloud computing e otimização de recursos.
5. **Automatização de Processos:** A introdução de processos automatizados para manutenção e operação do sistema pode reduzir ainda mais os custos operacionais e aumentar a eficiência. Ferramentas de DevOps e outras tecnologias de automação serão fundamentais para alcançar este objetivo.
6. **Feedback e Melhoria Contínua:** Recolher feedback dos utilizadores finais e incorporar melhorias contínuas com base nas suas necessidades e sugestões será vital para manter a relevância e eficácia da aplicação.
7. **Documentação e Treinamento:** Continuar a atualizar a documentação técnica e fornecer treinamento adequado para os utilizadores e a equipa de manutenção garantirá que todos estejam capacitados para utilizar e manter o sistema de forma eficiente.

A realização destes trabalhos futuros contribuirá para a longevidade e sucesso contínuo do sistema modernizado, alinhando-o com as necessidades em constante evolução do Grupo Barraqueiro e do mercado.

## **Agradecimentos**

Gostaríamos de expressar a nossa gratidão à equipa do Grupo Barraqueiro, que sempre nos apoiou e ajudou a esclarecer todas as dúvidas que tivemos ao longo do desenvolvimento deste projeto. O seu suporte e disponibilidade foram fundamentais para o sucesso do trabalho.

Agradecemos também ao nosso professor orientador, Pedro Serra, pela orientação e pelo constante incentivo. A sua experiência e conselhos foram indispensáveis para a conclusão deste projeto.

A todos os envolvidos, o nosso sincero obrigado.



## **Bibliografia**

- [1] DEISI, Regulamento de Trabalho Final de Curso, Set. 2023.
- [2] Universidade Lusófona de Humanidades e Tecnologia, [www.ulusofona.pt](http://www.ulusofona.pt), acedido em Nov. 2023.
- [3] Grupo Barraqueiro: <https://barraqueiro.com>
- [4] DBF File Structure (<http://www.dbf2002.com/dbf-file-format.html>). From the developers of DBF Viewer 2000.

## Questionário

Foi realizado um questionário para todos os utilizadores da app, para garantir que os objetivos traçados são de acordo com o que realmente as pessoas da app pretendem com a nova versão:

<https://forms.gle/T68HPiZFuqUDZBv29>

Este questionário é destinado ao melhoramento e aperfeiçoamento do sistema *Legacy* o qual ajuda na gestão e administração dos serviços do grupo Barraqueiro 🚚

1. Qual o cargo que ocupa na empresa?

Esta pergunta serve para entender quais são os diferentes cargos das pessoas que utilizam a aplicação com o objetivo de desenvolver uma melhor interface para todos os cargos.

2. Acha a aplicação “Galp” fácil de se usar?

Compreender se a aplicação precisa de uma grande transformação a nível da interface.

3. Se não, porquê?

Verificar as melhorias que a aplicação precisa ou pelo menos identificar quais as principais melhorias que são necessárias.

4. Acredita que a melhoria da aplicação Galp seria uma mais valia?

Entender se a melhoria da aplicação é aplicável para a maioria dos utilizadores da mesma.

5. Para si, qual é a parte da aplicação que deveria ter mais atenção?

Analisar e identificar a principal parte da aplicação que deveria ter mais atenção ao alterar.

6. Gostava que a nova aplicação tivesse alguma funcionalidade nova? Se sim, qual?

Corresponder ao máximo de expectativas dos utilizadores e tentar melhorar a aplicação ao máximo.

7. Acharia interessante juntar a interface do martelo à nova interface da aplicação Galp?

Confirmar a ideia inicial e principal da equipa de desenvolvimento com os utilizadores.

8. Na sua opinião, porque seria benéfico juntar as duas aplicações?

Compreender a opinião dos utilizadores e fundamentar as ideias do desenvolvimento.

9. O que espera da nova versão da aplicação Galp?

Ter uma ideia melhorada e estruturada para verificar os requisitos do projeto.

10. Quais são as suas expectativas para esta nova aplicação?

Identificar as expectativas principais dos utilizadores ao longo do projeto.

11. Feedback

Obter mais ideias e objetivos para o projeto através do feedback dos utilizadores.

Obtemos as seguintes respostas:

Um colaborador que desempenha funções como técnico administrativo nos serviços técnicos de manutenção do grupo Barraqueiro partilhou connosco a sua perspetiva sobre a aplicação "Galp". Segundo ele, a aplicação é considerada fácil de utilizar, no entanto, acredita que melhorias substanciais poderiam agregar valor ao sistema. Expressou a opinião de que a área de maior necessidade de atenção na aplicação seria a capacidade do Galp lidar com uma variedade de formatos de ficheiros de gasóleo. Isto é particularmente relevante devido à existência de diferentes formatos nos postos de abastecimento internos e nos ficheiros provenientes de fornecedores externos, tanto nacionais como estrangeiros.

O colaborador salientou a importância de uma gestão eficaz dos cartões de gasóleo, abrangendo tanto os existentes quanto os futuros. Observou que a aplicação Galp atual apresenta limitações, principalmente na tipologia e no número de dígitos dos cartões, impedindo uma gestão adequada. Propôs que o Galp identificasse o tipo de combustível associado a cada cartão (gasóleo, gasolina, gás, eletricidade ou todos).

Além disso, enfatizou a necessidade de aprimoramento no tratamento automático de ficheiros para separar distintamente os diversos produtos adquiridos. Recomendou a implementação de gestão para diferentes modelos de utilização do cartão, como cartões nominais e cartões de viatura. Sublinhou a importância de alarmes para campos críticos nos cartões, como validade, período de inatividade e utilizador em atividade.

No que diz respeito à faturação, sugeriu a criação de uma nova tabela para a gestão de códigos associados aos utilizadores. Propôs que a gestão de cartões incluísse um campo associado a cada utilizador para o qual é emitida uma fatura. Com relação ao módulo de viaturas no Galp, expressou a necessidade de uma atualização automática, apoiada por outras aplicações de cadastro de frota, evitando assim atualizações isoladas.

Por fim, manifestou o interesse em integrar as interfaces, destacando a importância de funcionalidade e intuição na validação, com expectativas elevadas para uma experiência coesa e eficaz. Estas sugestões refletem a sua visão de aprimoramento significativo da aplicação "Galp" para atender de forma mais abrangente e eficiente às necessidades da empresa.

O diretor do Departamento de Desenvolvimento tem uma visão diferente em relação à aplicação. Na sua opinião, a usabilidade da aplicação atual deixa a desejar devido à falta de uma interface intuitiva. Ele acredita firmemente que aprimorar a usabilidade é essencial e representa uma oportunidade valiosa para aperfeiçoar a experiência do utilizador. Para ele, a área que requer mais atenção é, sem dúvida, a interface, pois é fundamental que os utilizadores possam interagir de forma intuitiva e eficaz com o software.

Ao contrário da visão anterior, o Diretor não está inclinado a desejar novas funcionalidades no novo software. Para ele, o foco deve estar na otimização do que já existe, garantindo que as funcionalidades atuais sejam executadas de maneira eficiente e sem complicações desnecessárias. Ele destaca que a simplicidade e a eficácia são prioridades, e acredita que novas funcionalidades podem desviar o foco do objetivo principal, que é melhorar a usabilidade. Quanto à integração das duas aplicações, o diretor não vê isso como uma prioridade ou necessidade. Sua ênfase está na versatilidade e usabilidade da nova versão do software. Ele espera que a atualização seja mais flexível, adaptando-se melhor às necessidades da equipa e proporcionando uma experiência de utilizador mais amigável.

Assim, enquanto o colaborador de serviços técnicos destaca a importância de novas funcionalidades e integração de interfaces, o Diretor do Departamento de Desenvolvimento está mais centrado na usabilidade, simplicidade e otimização das funcionalidades existentes. Ambas as perspetivas são valiosas e devem ser consideradas no processo de aprimoramento da aplicação "Galp".

## **Anexos**

- [Esboço da arquitetura](#)
- [Esboço da interface UI](#)
- [Diagrama de árvore](#)
- [Modelo de entidade-relação](#)
- [Diagrama de Atividade](#)
- [Diagrama de Caso de Uso](#)
- [Relatório intercalar - 1º semestre](#)
- [Relatório intermédio - 1º semestre](#)
- [Relatório intercalar - 2º semestre](#)
- [Planeamento do desenvolvimento - Diagrama de Gantt \(project Libre\)](#)
- [Manual do Utilizador do Software](#)
- [Manual Técnico do Software](#)
- [Imagem e Texto de apresentação](#)

## **Glossário**

LEI	Licenciatura em Engenharia Informática
LIG	Licenciatura em Informática de Gestão
TFC	Trabalho Final de Curso
UI	User Interface