Relatório de Estágio

Tiago Filipe Teixeira Ribeiro - a39977

Relatório apresentado à Escola Superior de Tecnologia e de Gestão de Bragança para obtenção do Grau de Mestre em Informática.

Trabalho orientado por:

Prof. Paulo Jorge Matos

Leandro Ismael Pereira Alexandre

Este relatório final de estágio não inclui as críticas e sugestões feitas pelo Júri.

Bragança

2021-2022

Relatório de Estágio

Tiago Filipe Teixeira Ribeiro - a39977

Relatório final de estágio profissional apresentado à Escola Superior de Tecnologia e de Gestão de Bragança para obtenção do Grau de Mestre em Informática.

Trabalho orientado por:

Prof. Paulo Jorge Matos

Leandro Ismael Pereira Alexandre

Este relatório final de estágio não inclui as críticas e sugestões feitas pelo Júri.

Bragança

2021-2022

Dedicatória

Dedico este trabalho em primeiro lugar aos meus pais que fizeram sempre o que estava ao seu alcance para que eu conseguisse chegar aqui, à minha namorada, pois sem ela e sem o seu apoio não estava aqui a terminar e a entregar este trabalho.

Agradeço às minhas irmãs, por se preocuparem com o meu bem-estar e por estarem sempre ao meu lado. Dedico também este trabalho a toda a minha família que direta ou indiretamente todos eles contribuíram para eu chegar até aqui.

Por fim dedico este trabalho a dois colegas de curso, colegas estes que conheci na melhor cidade universitária do país, André Matos e João Castro.

Agradecimentos

Em primeiro lugar quero agradecer a toda a Equipa da TWA e Markdata, que sempre me ajudaram ao longo do estágio com todo o incentivo e ajuda que eu precisava, e de tornar a integração um processo simples. Um especial agradecimento ao Leandro Alexandre por todo o apoio e ajuda no desenvolvimento deste trabalho, a sua experiência revelou-se importante na sua orientação.

Agradecer também ao meu orientador Professor Paulo Jorge Matos, pela sua orientação sempre nos momentos em que foi necessária.

Resumo

O presente relatório visa detalhar o trabalho desenvolvido ao longo do estágio curricular na empresa TWA (Tagus World Analytics), no âmbito da unidade curricular de Estágio do ciclo de estudos do Mestrado em Informática.

O estágio focou-se no desenvolvimento de soluções *web* com recurso à *framework Vue.js* no *frontend* e o *backend* recorrendo a *framework Express.js* baseada em *Node.js*. Ao longo do relatório de estágio são apresentadas todas as tecnologias e ferramentas utilizadas, bem como a metodologia de desenvolvimento aplicada no decorrer do estágio.

As tarefas desenvolvidas ao longo de todo o estágio, passaram por dois projetos que já estavam a ser desenvolvidos pela empresa, nomeadamente o Telentry e Clipping. Ao longo deste relatório vão ser abordadas todas as tarefas desenvolvidas, bem como todos os desafios e problemas encontrados ao longo do estágio.

Por fim, abordar-se de uma maneira geral como foi o funcionamento de todo o estágio, nomeadamente, quais as capacidades desenvolvidas, quais os conhecimentos adquiridos e quais as mais-valias para o mercado de trabalho que este estágio trouxe.

Palavras-chave: *Vue.js*, TWA, *Node.js*

Abstract

This report demonstrates all the work developed during the curricular internship at the company TWA (Tagus World Analytics), within the scope of the Internship curricular unit of the cycle of studies of the Master of Informatics.

The internship focused on developing web solutions using the Vue.js framework on the frontend and, on the backend, using the Express.js framework, based on Node.js. Throughout the internship report, all the technologies and tools used are presented, as well as the development methodology applied during the internship.

The tasks, developed throughout the internship, went through two projects that were already being developed by the company, namely Telentry and Clipping. Throughout this report, all the tasks developed will be addressed, as well as all the challenges and problems encountered during the internship.

Finally, to discuss in a general way how the entire internship worked, namely, what skills were developed, what knowledge was acquired and what added value to the job market that this internship brought.

Keywords: *Vue.js*, TWA, *Node.js*

Conteúdo

Lista de Tabelas

Lista de Figuras

Capítulo 1

Introdução

Este relatório consistiu no desempenho da função de programador *fullstack developer* pela empresa *TWA(Tagus world Analytics)*. Recentemente, em novembro de 2021, a empresa percebeu a necessidade de aumentar a equipa de programadores com o principal objetivo de aumentar a produtividade e a integração em novos projetos de desenvolvimento *web*.

O principal objetivo deste relatório é retratar o dia a dia do desenvolvimento de um projeto *web*, sendo que se irá descrever o dia a dia do desenvolvimento de todas as tarefas propostas ao longo do estágio. Para o efeito, o presente relatório inicia com a caracterização da empresa, aspetos históricos, sua missão e visão, estrutura e cultura organizacionais.

Irão ser abordadas todas as ferramentas e tecnologias utilizadas no desenvolvimento das tarefas. A seguir, é elaborada uma descrição das funções realizadas durante o estágio, enquadrando à função de *fullstack developer*, por fim finaliza-se este relatório com algumas conclusões que derivam da reflexão, entretanto desenvolvida, discutindo lições aprendidas e melhorias para o futuro.

1.1 Enquadramento

O presente relatório de estágio retrata a experiência de estágio curricular decorrida no âmbito do 2.º ano do Mestrado de Informática, lecionado na Escola Superior de Tecnologia e Gestão ESTiG do Instituto Politécnico de Bragança Instituto Politécnico de Bragança

(IPB), este relatório final de estágio foi levado ao cabo na Empresa *TWA (Tagus world Analytics)*.

1.2 Objetivos

O presente relatório de estágio tem como principal objetivo demonstrar todo o percurso efetuado ao longo do período de estágio na empresa *TWA (Tagus world Analytics)*, ira-se retratar o dia a dia de um programador *fullstack*, e todos os desafios, tarefas que vão ser propostas, assim como retratar as ferramentas e tecnologias utilizadas ao longo de todo o estágio.

1.3 Estrutura do Documento

Este relatório final de estágio encontra-se dividido em 6 capítulos, estando estes ainda divididos em várias secções, de modo a estruturar da melhor forma o relatório de estágio.

- **Capítulo 1- Introdução**
  Neste capítulo é elaborada uma abordagem mais introdutória a todo o relatório. Desta forma, pretende-se abordar os objetivos e o enquadramento, assim como a estrutura deste documento.

- **Capítulo 2: Revisão de Literatura**
  Todo o enquadramento teórico e estado de arte são apresentados neste capítulo, dando especial destaque aos principais conceitos em que assentou o trabalho ao longo do período de estágio.

- **Capítulo 3: Caracterização da Organização**
  A descrição, estrutura e metodologia de trabalho da instituição de acolhimento são abordadas neste capítulo. Pretende-se dar a conhecer um pouco mais sobre a *TWA (Tagus world Analytics)* e os seus colaboradores.

- **Capítulo 4: Tecnologias e Ferramentas**

  É no 4.º capítulo que são apresentadas, detalhadamente, as principais tecnologias e ferramentas utilizadas ao longo do período de estágio.

- **Capítulo 5: Tarefas desenvolvidas**

  Os objetivos do estágio são apresentados neste capítulo, assim como a demonstração de trabalho efetuado e diários de trabalho.

- **Capítulo 6: Conclusões**

  O 6.º e último capítulo apresenta uma breve análise ao período de estágio. São ainda apresentadas algumas ideias em termos de trabalho futuro e uma breve reflexão de todo o trabalho desenvolvido ao longo do estágio.

Capítulo 2

Revisão da Literatura

2.1 Enquadramento Teórico

2.1.1 TWA (Tagus world Analytics)

A TWA (Tagus world Analytics) é uma empresa que trabalha com uma abundante quantidade de dados e informação. Atua principalmente no mercado de media e estudos de mercado, lidando com os diferentes tipos de dados (TV, Internet, Rádio, Digital, dados de pesquisas e muitos outros tipos de dados).

Tem também uma parceria estratégica que é a Markdata, uma empresa sediada em Lisboa com mais de 30 anos [1]. As principais funções da empresa passam pelo desenvolvimento de soluções *web* com o recurso à *framework Vue.js* no *frontend* utilizado para produzir e manipular informação, além disso, utilizam a *framework Express.js* baseada em *Node.js* para o *backend*, responsável por devolver a informação ao *frontend* e comunicação com a base de dados.

O uso destas tecnologias permite que seja mais fácil catalogar e armazenar toda a informação que os diferentes tipos de media disponibilizam.

Capítulo 3

Caracterização da Organização

3.1 História

A TWA (Tagus world Analytics) surgiu no ano de 2019 com a abertura de um polo na cidade de Bragança, onde o principal objetivo da empresa é desenvolvimento de *software* para lidar com os diferentes tipos de dados da media como TV, Internet, Rádio, Digital, dados provenientes de pesquisas e muitos outros tipos de dados. A TWA surgiu numa altura em que os dados, para serem úteis, têm de ser transformados em informação de modo a servir de base à tomada de decisão, ou seja, para que isto possa acontecer é necessário que a informação dos estudos seja processada, sintetizada, arrumada e é aqui que a TWA tem um papel importantíssimo, pois ajuda no desenvolvimento de *software* que ajudará no tratamento destes dados. Atualmente trabalham no polo de Bragança seis colaboradores.

A TWA é uma empresa recente no mercado, é uma empresa que pertence ao grupo Marktest. O grupo Marktest nasceu em 26 de junho de 1980, numa época em que os estudos de mercado iniciavam um ciclo de expansão. Centrados nos produtos de grande consumo, solicitados quase exclusivamente pelas multinacionais, os estudos de mercado apoiavam-se numa oferta diminuta. Os primeiros anos da Marktest, com uma estrutura ainda não muito consolidada, foram a procura de um espaço próprio num mercado em

expansão.

O grupo Marktest apresentava uma estratégia agressiva, centrada na preocupação de servir o cliente e de ajudar a resolver os seus problema e com esta estratégia permitiu uma implantação segura e progressiva junto dos principais agentes do mercado publicitário.

No início dos anos 80, a micro informática estava a surgir no mercado e o fundador da empresa, Luís Queirós, considera ainda que "o facto da Marktest ter, de forma pioneira, apostado nos computadores como instrumento de trabalho, ajudou a catapultar a empresa para o estádio mais avançado do sector".

Atualmente a grupo Marktest evoluiu e cresceu sendo empresa de *software* líder mundial, especializada em Pesquisa e planeamento de media. É único por ser independente de fontes de dados, portanto, capaz de lidar com diferentes tipos de dados e estudos de vários provedores. Presente em mais de 30 países e sendo uma das pioneiras mundiais neste mercado, a Marktest tem ajudado continuamente analistas e líderes de negócios a obter informações confiáveis há mais de 30 anos [2].

Figura 3.1: Marktest e TWA

3.1.1 Organigrama

Na figura 3.1, pode-se ver a organização do grupo ao qual pertence atualmente a TWA (Tagus world Analytics).

Figura 3.2: Grupo Marktest

3.1.2 Missão e objectivos

O Grupo Marktest, constituído por várias empresas especializadas na área de estudos de mercado e processamento de informação, tem crescido de uma forma contínua e sustentada desde a sua fundação, sendo hoje o Grupo português com maior projeção (inter)nacional na sua área de atuação.

A Atividade do Grupo abrange vários importantes segmentos como a medição de audiências de meios, monitorização de investimentos publicitários, estudos regulares (barómetros) nas áreas das telecomunicações, banca, seguros, painéis na área da Internet e estudos de *pricing* e auditoria de retalho [3].

Totalmente integrada com estas atividades, a TWA criou uma forte área de desenvolvimento de *software*, que permite o desenvolvimento da sua atividade baseado na qualidade e inovação e a afirmação do Grupo no mercado nacional e também numa sustentada expansão internacional. Desenvolve estudos para fornecer aos clientes informação com

qualidade, rigor e independência, prestando assim, um serviço de reconhecido valor acrescentado baseado na inovação e atualização tecnológica.

O grupo tem procurado ao longo destes anos junto das equipas delegar responsabilidades, dar autonomia, mas exigindo rigor, integridade e profissionalismo total para credibilizar de uma forma permanente a sua atividade. Isto é o que designam de Cultura da TWA e Marktest onde procuram desenvolver e recompensar os colaboradores.

O Grupo Marktest reconhece assim as suas responsabilidades sociais no âmbito da colaboração e parcerias que executado ao longo da sua existência com instituições de ensino e de solidariedade social, mesmo para a população em geral, através do acesso a fontes de conhecimento e informação propriedade das empresas do Grupo [4].

3.1.3 Estrutura Organizacional

Cada empresa do grupo Marktest tem os seus responsáveis, ou seja, cada uma das empresas pertencentes ao grupo tem um conselho que administra [5].

- Marktest:
  - Jorge Fonseca Ferreira
  - José Manuel Oliveira
  - Miguel Fontán
- Markdata:
  - Paulo Silva
  - Francisco Giménez
- MediaMonitor:
  - Ana Cristina Dias
  - Nathalie Costa
- Marktest Angola:

  - Ana Paula Pereira

- Medialog:
  - Carina Dias
- Tagus World Analytics
  - Paulo Silva

Capítulo 4

Tecnologias e Ferramentas Utilizadas

4.1 Arquitetura

4.1.1 Aplicação *web*

O que é uma aplicação *web*

Uma aplicação *web*, é uma aplicação de *software* utilizada na internet, em vez de funcionar localmente.

Uma aplicação *web* é um sistema com funcionalidades completas, programado a partir de requisitos e dos princípios da engenharia de *software* tal como uma aplicação *desktop* corrente. A principal diferença entre uma aplicação *web* e uma aplicação *desktop* é a informação onde o utilizador trabalha, é processada e armazenada num servidor na *internet* alojado fora do computador *desktop*, ou seja, o utilizador esteja a onde estiver tem sempre acesso aos dados e ficheiros com uma simples ligação a internet. Além disso, estas aplicações *web*, não precisam de estar instaladas localmente [6].

Como funcionam as aplicações *web*

Quando se usam as aplicações *web*, esteja o utilizador num computador ou num dispositivo móvel, a maioria do processamento é realizado mediante uma rede de servidores alojados

na internet em qualquer parte do mundo. Normalmente todos os pedidos que um utilizador faz ao servidor usa protocolos e métodos como, por exemplo, o HTTP e HTTPS, os mais comuns nas aplicações *web*, e estes protocolos são como a linguagem de comunicação entre o servidor a aplicação *web*.

A aplicação *web* deve permitir que os utilizadores consigam fazer um pedido ao servidor e receber algo como resposta, ou seja, os servidores precisam de ir buscar a resposta à pergunta do utilizador, devolvendo assim a resposta pretendida como resultado, por exemplo, se o utilizador pede para abrir uma fotografia, é preciso que o servidor devolva a fotografia, e não uma página *web* aleatória [7].

Em resumo, o servidor tem por função receber o pedido do utilizador e devolver uma resposta para a aplicação *web*.

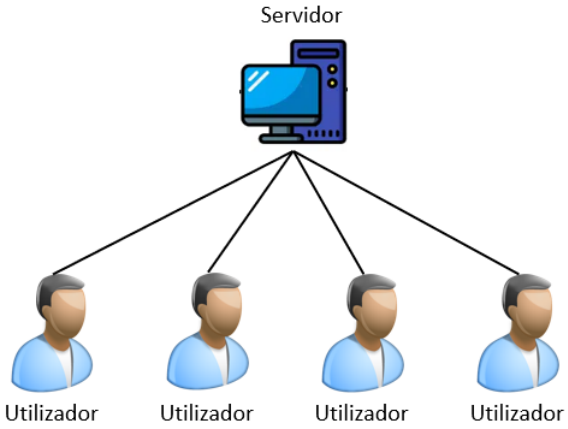

Figura 4.1: Comunicação

Quais as principais partes da estrutura de uma aplicação *web*

Uma aplicação *web* é composta essencialmente por duas partes que são elas:

- *frontend*: É toda a parte visual da aplicação *web*. Geralmente o *frontend* segue o *design* e o estilo predefinido nos *mockups*, definidos numa fase inicial do desenvolvimento. Através das linguagens de programação como, por exemplo, o *Vue.js*, o

frontend é desenvolvido, além disso, pode-se incluir no desenvolvimento outras tecnlogias como o CSS (*Cascading Style Sheet*) responsável por dar os estilos as páginas. Em suma, o *frontend* é o que é mostrado ao o utilizador e a onde este interagirá.

- *backend*: O *backend*, como o próprio nome já diz, é tudo aquilo que está por de trás de uma aplicação *web*. O *backend* pode ser desenvolvido com o recurso a linguagens de programação, como o *Java*, *Ruby*, *C Sharp*, *Python* e *Node.js*. É onde são construidos os métodos e funções responsáveis por interagir com o *frontend* e a base de dados.

  Além disso, é através do *backend* a onde ocorre a ligação com a base de dados.

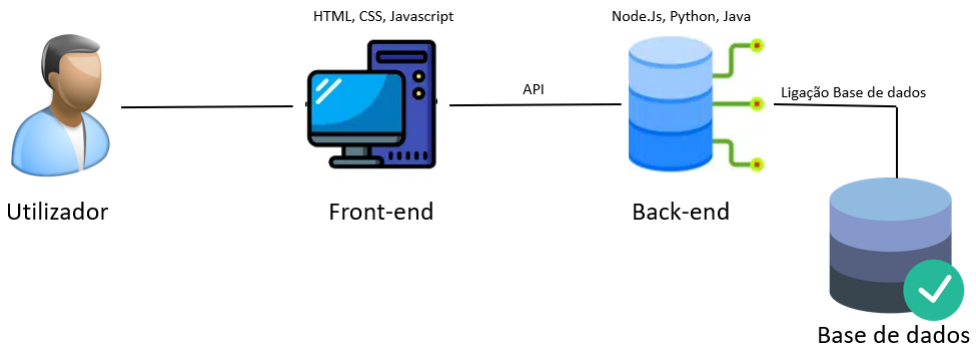

Figura 4.2: Comunicação numa aplicação *web*

Como desenvolver uma aplicação *web*

O desenvolvimento de aplicações *web*, pode ser feito de uma maneira estática, com uma metodologia específica. Contudo, existem alguns passos que as equipas seguem e que são essenciais no decorrer do desenvolvimento da aplicação *web*. As etapas são as seguintes:

- Levantamento de requisitos: na qual a equipa conversa com o cliente e analisa a ideia;

- Especificação de requisitos: especificação de todos os requisitos funcionais e não funcionais;

- Análise: criação de modelos e representações que ajudam a esquematizar e organizar o projeto;
- Implementação: programação de todos os requisitos e funcionalidades discutidas na parte de modelação;
- Verificação: validação da aplicação e descoberta de erros;
- Instalação e configuração: disponibilização da aplicação ao cliente;
- Validação: Observação do funcionamento da aplicação e discussão de melhorias.

Contudo, esta metodologia, é geralmente otimizada e diferente de equipa para equipa. Não há necessidade de seguir uma sequência rígida, por isso, todos os passos podem ser alterados dependendo da necessidade de cada equipa de desenvolvimento [8].

4.2 Tecnologias

4.2.1 *Vue.js*

História

Vue.js foi criado por Evan You, e o primeiro código-fonte foi lançado em julho de 2013 e o *Vue.js* foi lançado em fevereiro de 2014.

Evan You trabalhava na Google e desenvolvia projetos em *AngularJS*, mas quando ele precisava de desenvolver rapidamente uma *interface* para o cliente que cumprisse todos os objetivos do projeto, não existia nenhuma *framework* que o fizesse. E escrever um monte de código em *HTML* repetido consumia tempo e recursos, e foi por isso que Evan começou a procurar uma *framework* que já existisse no mercado para esse propósito.

Contudo, ele descobriu que não havia essa *framework* que pudesse encaixar exatamente nas suas necessidades. Naquela época, a *framework* Angular era bastante utilizada, e o *React.js* acabara de surgir no mercado, e as *frameworks* como o *Backbone.js* eram usados para aplicações de grande escala com arquitetura MVC *(Model-View-Controller)*. Para o

tipo de projeto que Evan You desenvolvia, ele precisava de algo realmente flexível e leve e nenhuma dessas *frameworks* eram adequadas.

Atualmente *Vue.js* é muito utilizado para criar aplicações *single page* e também para desenvolver vários tipos de *interfaces*, que necessitem de maior interação e experiência entre os utilizadores. Esta tecnologia foi usada como principal *framework* no *frontend* ao longo do estágio [9].

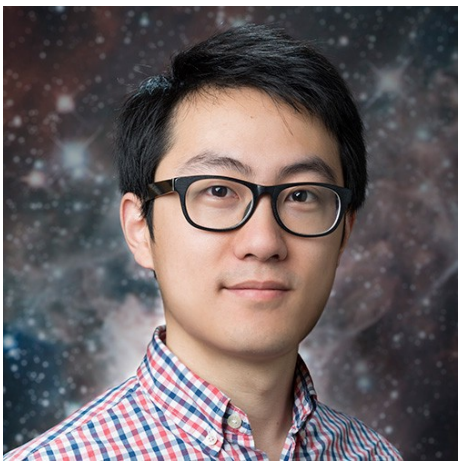

Figura 4.3: Even You criador do *Vue.js*

Principais funcionalidades e vantagens do uso do *Vue.js*

- Fácil aprendizagem e compreensão: A grande popularidade do *Vue.js* entre os programadores é conhecida pela sua simplicidade. Para utilizar e programar em *Vue.js*, os *developers* precisam de conhecer apenas os fundamentos do *JavaScript, HTML e CSS*. Além disso, o Vue consiste em documentos capazes de incluírem vários tipos de código *JavaScript, CSS, HTML*;

- Grande Comunidade: O *Vue.js* não é apoiado por nenhuma empresa gigante de tecnologia, mas é totalmente apoiado por doações de patrocinadores e a comunidade *open-source*, é o que torna o *Vue.js* em constantemente crescimento;

- Simplicidade: *Vue.js* tem uma arquitetura baseada em componentes com três principais vantagens:
  - Reutilização de componentes - pedaços de código que podem ser reutilizados como modelos;
  - Legibilidade do código - todos os componentes são armazenados em ficheiros separados, facilitando assim o acesso, a manutenção e a correção;
  - Excelente para testes de unidade - os testes de unidades funcionam muito bem, pois é fácil monitorizar o desempenho;
- Variedade de ferramentas - Apesar de ser uma *framework* relativamente nova, o *Vue.js* possui um poderoso conjunto de ferramentas que o tornam ainda mais competitivo e organizado, tais como:
  - Quasar;
  - Vuex;
  - Vuetify;
  - Nuxt;
- Alto Desempenho: O *Vue.js* é um dos *frameworks* mais rápidos. Em comparação com outros *frameworks* como o React e Angular, o *Vue.js* renderiza mais rápido a informação em dispositivos móveis, disponibilizando assim uma melhor *User Experience*;
- *Model-View-ViewModel* (MVVM) arquitetura: O *Vue.js* usa uma arquitetura MVVM que aprimora muito o *user interface* através da simplicidade da programação orientada a eventos da *user interface*. Concentrado principalmente na camada *ViewModel* da arquitetura MVVM, o *Vue.js* separa o *user interface* da lógica da aplicação;
- Capacidades de integração: Uma das características que todo o programador procura é a capacidade de integração com outras aplicações;

O *Vue.js* depende completamente do *JavaScript* para que os programadores possam escrever modelos com melhor flexibilidade e mudar para *React* ou *Angular*, por exemplo, se for necessário;

- Documentação: O *Vue.js* tem uma documentação bem escrita. A documentação está traduzida em oito idiomas com uma ampla variedade de exemplos detalhados. A comunidade *Vue.js* também oferece cursos em vídeo para diferentes níveis, sejam programadores iniciantes ou avançados [10];

Organização da estrutura *Vue.js*

Na figura 4.2 do lado esquerdo, esta representado a organização de uma página escrita utilizando a *framework Vue.js*. Como se pode ver tem-se primeiramente todo o código em HTML, depois todos os *imports* necessários. Depois na *"data"* tem-se todas as declarações de variáveis a utilizar.

No *"mounted"* é onde estão os métodos que são os primeiros a ser mostrados quando a página *web* é carregada para o utilizador. No *"computed"* é onde permite que o programador crie uma propriedade que pode ser usada para modificar, manipular e exibir dados nos componentes de maneira legível e eficiente. Já o *"watch"* é utilizado para estar a escuta e sempre que um componente é alterado o *"watch"* será manipulado e guarda essa informação.

Por fim, nos *"methods"* é onde vão ser criados todos os métodos de manipulação e de novas funcionalidades da página *web*. Estes foram os componentes maioritariamente usados ao longo do estágio, mas existem muitos mais, como se pode observar pela figura 4.4 do lado direito.

Na figura 4.5, pode-se observar um diagrama com o ciclo de vida de uma página construida em *Vue.js*. Com este diagrama é mais fácil perceber como funcionam os componentes em *Vue.js*.

```
You, 1 second ago | 4 authors (Leandro Alexandre and others)
<template>
  <div>···
  </div>
</template>
<script>
import ChannelService from '@/services/ChannelService.js'
import Sortable from 'sortablejs/modular/sortable.core.esm.js'

export default {
  data: () => ({···
  async mounted() {···
  },
  computed: {···
  },

  watch:{},

  methods: {···
  },
}
</script>
```

```
<script>
  // todos os imports que houver
  export default {
    props: {},
    components: {},
    mixins: [],
    directives: {},
    data () { return {} },
    computed: {},
    filters: {},
    beforeCreate () {},
    created () {},
    beforeMount () {},
    mounted () {},
    beforeUpdate () {},
    updated () {},
    beforeDestroy () {},
    destroyed () {},
    methods: {},
    watch: {}
  }
</script>
```

Figura 4.4: Componentes de uma página *Vue.js*

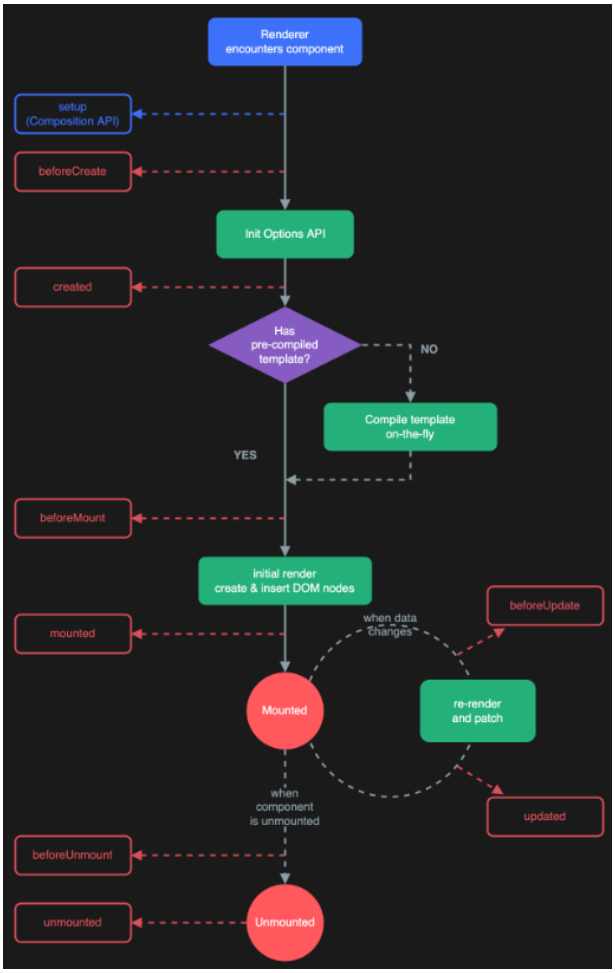

Figura 4.5: *Lifecycle Hooks* de uma página em *Vue.js* [11]

Evolução do *Vue.js*

Como se pode constatar na figura 4.6, o *Vue.js*, está em crescimento em comparação com outras *frameworks*. Sendo o *Vue.js* uma *framework* recente está com um bom ritmo de crescimento que, se continuar assim ao longo dos tempos, vai equiparar-se ou mesmo ultrapassar a *framework React*.

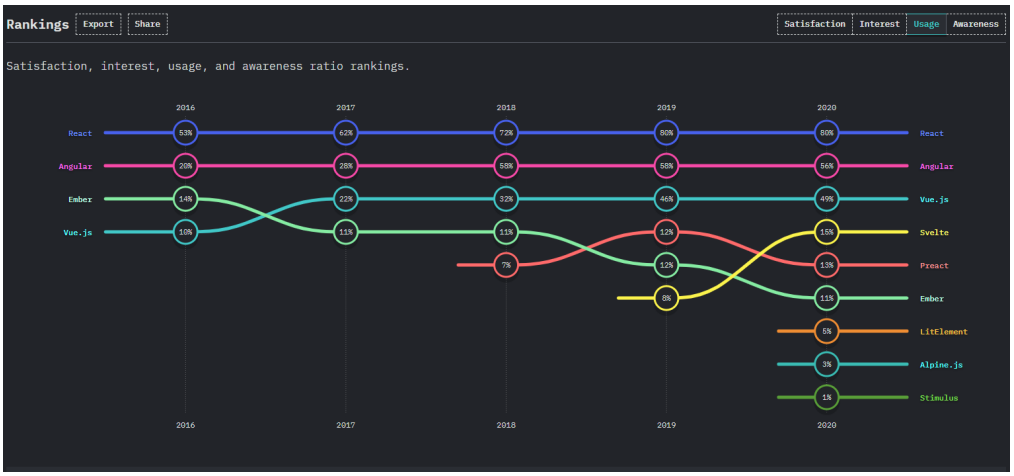

Figura 4.6: Evolução do *Vue.js* [12]

Na figura 4.7, esta representado a evolução do interesse no *Vue.js* ao longo dos anos, o que se pode verificar que o interesse cresce em relação ao desinteresse.

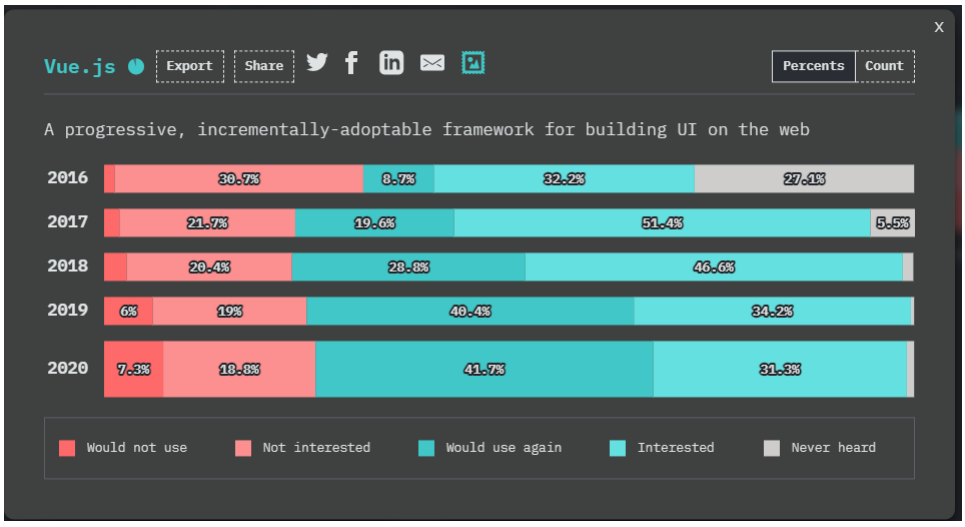

Figura 4.7: Interesse do *Vue.js* [12]

4.2.2 *Vuetify*

O *Vuetify* é uma estrutura de UI *(User Interface)* completa construída para usar com *Vue.js*.

O principal objetivo do *Veutify* é fornecer aos *developers* as ferramentas necessárias para enriquecer as experiências de utilização das páginas *web*. O *Vuetify* foi projetado desde o início para ser fácil de aprender e desta forma fácil de os *developers* dominarem centenas de componentes.

O *Vuetify* utiliza uma abordagem reativa, ou seja, que independentemente do dispositivo a onde o componente esteja a ser usados, ele funcionará imediatamente. Umas das vantagens em usar o *Vuetify* é que o programador não perde tempo em desenvolver componentes como, por exemplo, um calendário ou uma tabela [13].

O *Vuetify* é desenvolvido exatamente segundo as especificações do material design do *Vue.js*, com todos os componentes meticulosamente criados para serem modulares, responsivos e com um grande desempenho. Além disso, pode-se usar *Layouts* exclusivos e dinâmicos e o CSS *(Cascading Style Sheets)* dos componentes podem ser alterados usando variáveis SASS, ou seja, são variáveis que armazenam informações que podem ser reutilizadas posteriormente inúmeras vezes.

Figura 4.8: *Veutify*

4.2.3 *Node.js*

História

O *Node.js* surgiu no ano de 2009, e o seu criador foi Ryan Dahl.

Foi o próprio fundador Ryan Dahl que no ano de 2009 numa palestra a onde explicou que *Node.js* é baseado em eventos, e funciona em *callbacks* onde cada função de I/O, deve usar *callback*. No início o *Node.js* apenas estava disponível para Linux e só em julho de 2011 é que o *Node.js* ficou disponível para o *windows* [14].

O *Node.js* é uma plataforma de desenvolvimento *open-source* que executa código no lado do servidor. O *Node.js* é útil para desenvolver aplicações que exigem uma ligação persistente do *browser* ao servidor, é usado atualmente em muitas aplicações do dia a dia. O *Node.js* é suportado pelas várias plataformas existentes atualmente no mercado, como *windows* e *Linux*.

Com o *Node.js* é possível a criação de páginas com conteúdo dinâmico, é possível a criação, leitura, escrita e remoção de ficheiros no servidor, também se pode guardar informações de um formulário e, por fim, é possível modificar, eliminar ou criar informações na base de dados. Ao longo do estágio foi usado o *Node.js* com a *framework Express.js*, uma *framework* bastante usada no *Node.js*.

Figura 4.9: *Node.js*

Principais funcionalidades e vantagens do uso do *Node.js*

Uma das grandes vantagens do uso do *Node.js* é a execução do *Node single thread*, ou seja, apenas uma *thread* executa o código da aplicação. Desta maneira, menos recursos são exigidos, pois não é necessário criar uma *thread* para cada um dos pedidos recebidos. O *Node.js* utiliza apenas uma *thread*, chamada *Event Loop*, que cria eventos a cada pedido recebido [15].

- Flexibilidade - Isto faz do *Node.js* uma plataforma que pode ser utilizada em qualquer situação;

- Tamanho - Criar um ambiente *Node.js* e implementá-lo numa aplicação é uma tarefa que não exige muitos recursos computacionais em comparação com outras tecnologias mais tradicionais;

  O *Node.js* utilizado em conjunto com certas tecnologias tem o ganho na velocidade de desenvolvimento e em ambientes escaláveis isso significa menos custo e mais eficiência. Além disso, conta com suporte das principais empresas de produtos e serviços *Cloud* do mercado, como a AWS, Google *Cloud* e Microsoft Azure que oferecem na maioria dos seus produtos suporte nativo ao *Node.js*;

- Produtividade:

  - Simplicidade: O *Javascript* é uma linguagem padrão para desenvolvimento *web*. Empresas de desenvolvimento *web* contam com esta característica como um ponto de partida importante para iniciar o uso do *Node.js*. Além disso, esse fator pode representar ganhos de reutilização de código e criação de equipas multifuncionais, com melhor aproveitamento de recursos.

  - Ambiente de inovação: Possibilidade de iterações mais rápidas. Isto também permite a criação de soluções próprias e inovadoras.

Evolução do *Node.js*

Como se pode observar na figura 4.10, o *Node.js* é uma tecnologia bastante usada e apesar de estar agora em decréscimo não deixa de ser elevado o número de instalações. O *Node.js* foi uma das tecnologias usadas ao longo do estágio nos projetos desenvolvidos [16].

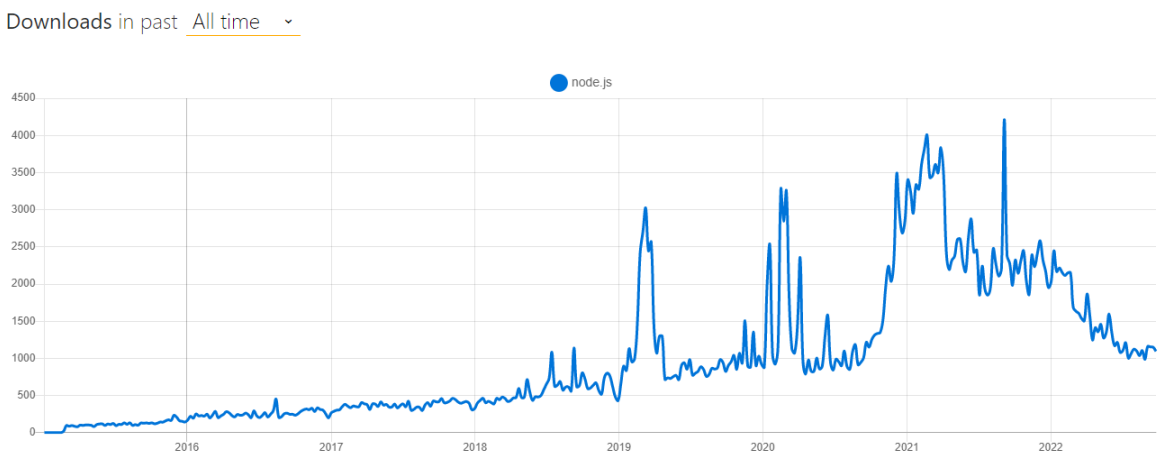

Figura 4.10: Utilização do *Node.js* [17]

4.2.4 *MySQL*

História

O *MySQL* é um dos sistemas de gestão de base de dados mais utilizados na atualidade. O seu nome é uma combinação de *"MY"*, o nome da filha do cofundador *Michael Widenius* e *"SQL"*, a abreviação de *Structured Query Language*.

Foi desenvolvido na década de 80 e apenas no ano de 1995 foi lançada a primeira versão do *MySQL*. Em janeiro de 2008, a *MySQL* foi adquirida pela Sun Microsystems, e no ano de 2009 foi adquirida pela Oracle.

Este sistema organiza os dados em tabelas, nas quais os dados podem estar relacionados entre si. O *MySQL* administra utilizadores, permite o acesso à rede e a criação de *backups*. O *MySQL* é utilizado por muitas aplicações da internet.

Figura 4.11: *MySQL*

Principais funcionalidades e vantagens do uso do *MySQL*

É uma tecnologia gratuita, tendo como principais características [18]:

- Facilidade de uso - O *MySQL* é fácil de configurar e de ajustar e por isso é muito fácil atingir grandes níveis de desempenho. Ferramentas de interface gráfica terceiras, como *MySQL* Workbench e dbForge Studio, tornam o *MySQL* ainda mais simples na manipulação de base de dados.

- Compatibilidade - Atualmente o *MySQL* oferece uma compatibilidade com a maioria das principais plataformas, como Linux, macOS, Microsoft Windows. Além disso, proporciona um grande desempenho para o armazenamento de dados em abundância.

- Suporte da comunidade - O apoio da comunidade é significativo para a melhoria de qualquer sistema de base de dados, neste ponto o *MySQL* apresenta uma comunidade muito ativa, que continuamente melhora os recursos existentes.

- Segurança - A segurança dos dados é garantida pelas funcionalidades do *Access Privilege System* e do *User Account Management*, para além de criptografia da palavra-passe do utilizador. Desta forma, torna, o *MySQL* é altamente seguro.

Utilização do *MySql*

Como se pode constatar pela figura 4.12, o uso do *MySql*, aparece em segundo lugar, o que faz com que seja um das linguagens de base de dados mais utilizadas. O *MySql* foi usado ao longo do estágio nos projetos desenvolvidos em âmbito empresarial.

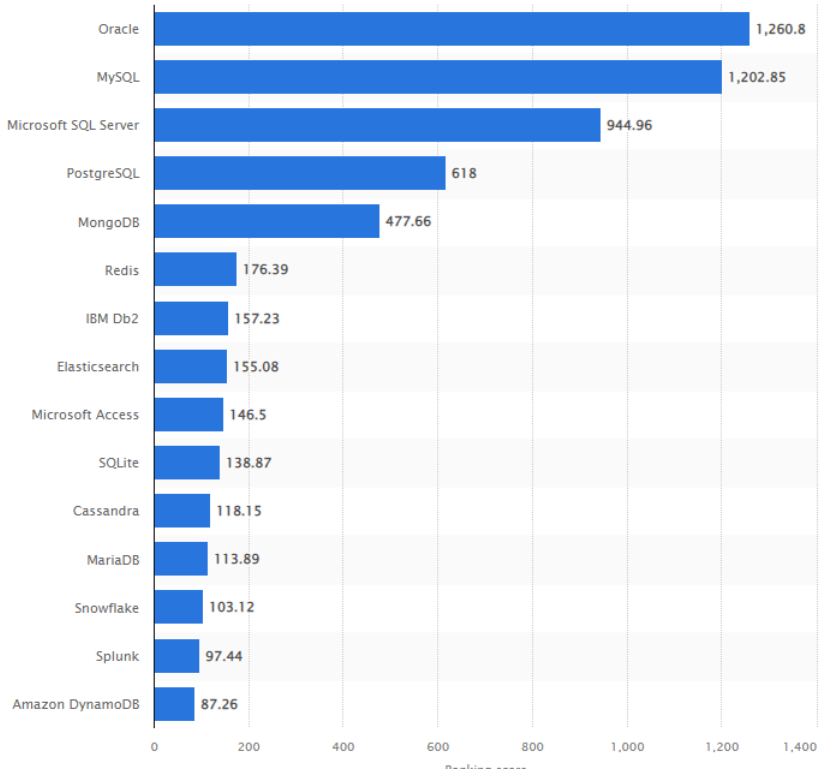

Figura 4.12: Utilização do *MySQL* [19]

4.2.5 *Microsoft SQL Server*

História

O *Microsoft SQL Server* é um dos principais sistemas de gestão de base de dados relacional e o seu lançamento ocorreu em 1989. Baseado na linguagem *Transact-SQL*, ou seja, é uma extensão da linguagem SQL usada principalmente no *Microsoft SQL Server*, o que significa que fornece algumas das funcionalidades do SQL, mas com algumas funcionalidades extra, próprias da Microsoft.

Um motivo pelo qual o *Microsoft SQL Server* é tão popular tem a ver com o seu criador, a Microsoft. Quando o *Microsoft SQL Server* estava numa fase de desenvolvimento, a Microsoft já era uma gigante da tecnologia. [20].

Figura 4.13: *Microsoft SQL Server*

Principais funcionalidades e vantagens do uso do *Microsoft SQL Server*

Existem várias vantagens e funcionalidades ao usar o *Microsoft SQL Server*, se bem que parte das funcionalidades são parecidas ao *MySql* [21]:

- O uso de triggers está bem desenvolvido e permite a sua utilização de uma maneira fácil;
- Tratamento de erros e exceção e controlo da transação;
- É usado principalmente para criar a lógica da aplicação;

Utilização do *Microsoft SQL Server*

Como se pode perceber pela figura 4.14, o uso do *Microsoft SQL Server*, aparece em terceiro lugar, o que faz com que seja uma das linguagens de base de dados mais utilizadas. O *Microsoft SQL Server* foi usado ao longo do estágio num dos projetos desenvolvidos em âmbito empresarial.

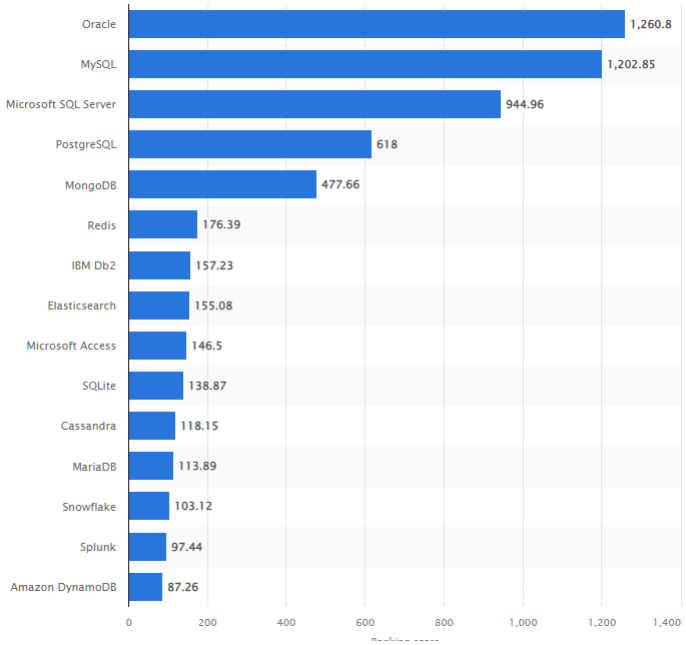

Figura 4.14: Utilização do *Microsoft SQL Server* [19]

4.2.6 *Sequelize*

O *Sequelize* é um ORM, ou seja, mapeamento objeto-relacional, e pode ser usado como, por exemplo, em Node.js e Postgres. O *Sequelize* é utilizado para comunicar com as bases de dados, sendo esta umas das grandes vantagens em usar *Sequelize*. Além disso, pode-se utilizar associações, transações de base de dados, reduzindo o tempo de desenvolvimento e evitando *SQL Injection*.

Pode-se alternar facilmente entre base de dados ajustando o ficheiro de configuração e o código desenvolvido permanecerá praticamente o mesmo.

O *Sequelize* foi usado no decorrer do estágio ao longo dos dois projetos desenvolvidos [22].

Figura 4.15: Sequelize

Principais funcionalidades e vantagens do uso do *Sequelize*

Existem várias vantagens e funcionalidades ao usar o *Sequelize* tais como [22]:

- Suporta muitos sistemas de base de dados;
- Suporte às transações;
- Sincronização de bases de dados;
- Migrações das bases de dados;
- Registo na consola de todas as consultas SQL que executa.

4.2.7 Redis

História

Redis, tem o significado de *Remote Dictionary Server*, é um armazenamento de dados de chave-valor rápido e de código aberto na cache. O projeto começou em 2009 na Itália,

quando Salvatore Sanfilippo, criador original do Redis, quis melhorar a escalabilidade da sua startup. A partir daí, desenvolveu o Redis, que agora é atualmente usado como armazenamento de dados, cache ou agente de mensagens.

Após encontrar problemas significativos no dimensionamento de alguns tipos de cargas de trabalho usando sistemas de bases de dados tradicionais, ele começou a desenvolver uma primeira versão do Redis. Redis foi escrito na linguagem ANSI C e funciona na maioria dos sistemas POSIX, como BSD, *Linux* sem dependências externas. O *Linux* é considerado o sistema operativo onde o Redis foi mais desenvolvido e testado e usado para implantar o mesmo.

O caso de uso comum por utilizadores experientes do Redis pode ser aplicado em tarefas como executar várias operações, como incrementar um valor de *hash*, anexar uma *string*, enviar um elemento para uma lista ou armazenar o membro com a maior classificação num conjunto ordenado. O Redis faz a gestão deste tipo de tarefas, o que resulta num excelente desempenho [23].

Figura 4.16: Redis

Principais funcionalidades e vantagens do uso do Redis

Existem várias vantagens e funcionalidades ao usar o Redis tais como [24]:

- Desempenho - Todos os dados do Redis residem na memória, permitindo assim o acesso a dados de baixa latência e alta taxa de transferência. Diferente dos armazenamentos de dados tradicionais, os armazenamentos de dados na memória não exigem uma visita ao disco, reduzindo a latência do mecanismo para microssegundos. Por causa disso, os armazenamentos de dados na memória podem suportar uma ordem de magnitude a mais de operações e tempos de resposta mais rápidos.

- Estruturas de dados flexíveis - Ao contrário de outros armazenamentos de dados de chave-valor que oferecem estruturas de dados limitadas, o Redis tem uma grande variedade de estruturas de dados. Os tipos de dados do Redis incluem:

  - Strings: dados em texto ou binários com tamanho de até 512 MB;
  - Listas: uma coleção de strings na ordem em que foram adicionadas;
  - Hashes: uma estrutura de dados para armazenar uma lista de campos e valores;
  - Conjuntos ordenados: conjuntos ordenados por um valor;

- Simplicidade e facilidade de utilização - O Redis permite a escrita de código tradicionalmente complexo com linhas mais simples e em menor quantidade. A diferença é que os programadores que usam o Redis usam uma estrutura de comandos simples, em oposição às linguagens de consulta das bases de dados tradicionais. Por exemplo, pode-se usar a estrutura de dados de hash do Redis para migrar dados para um armazenamento de dados com apenas uma linha de código. Uma tarefa similar, numa base de dados sem estruturas de dados de *hash*, exigiria muitas linhas de código para a conversão de um formato para o outro. O Redis é fornecido com estruturas de dados nativas e várias opções para manipular e interagir com dados. As linguagens compatíveis incluem *Java, Python, PHP, C, C++, C Sharp, JavaScript, Node.js, Ruby, R* entre outras.

- Repetição e persistência - O Redis emprega uma arquitetura de réplica principal e oferece suporte à replicação assíncrona, permitindo a replicação de dados para vários servidores. Esta arquitetura oferece maior desempenho de leitura (com a

distribuição dos pedidos entre vários servidores). Para proporcionar persistência, o Redis oferece um suporte a backups num ponto anterior no tempo.

- Alta escalabilidade - O Redis oferece uma arquitetura de réplica principal num único nó principal ou numa topologia de *clusters*. Dessa forma, pode-se criar soluções altamente disponíveis que oferecem um grande desempenho e fiabilidade constante.

- *Open Source* - Permitindo assim melhorias e novas funcionalidades constantes.

4.2.8 Socket.io

História

Socket.io foi criado em 2010 sendo desenvolvido com o intuito de facilitar a comunicação em tempo real, ainda que na época era um fenómeno novo.

O Socket.io permite a comunicação bidirecional entre cliente e servidor. As comunicações bidirecionais são ativadas quando um cliente tem Socket.IO no *browser* e o servidor também tem o Socket.io. Embora os dados possam ser enviados de várias formas, JSON é a forma mais simples. Para estabelecer a ligação e trocar dados entre cliente e servidor, o Socket.io usa o Engine.io. Engine.io é usado para a implementação do servidor e Engine.io-client é usado para o cliente [25].

Figura 4.17: Socket.io

Principais funcionalidades e vantagens do uso do Socket.io

Existem várias vantagens e funcionalidades ao usar o Socket.io tais como [26]:

- Desempenho - Geralmente, a ligação é estabelecida com o *WebSocket*, fornecendo assim um canal de comunicação de baixa sobrecarga entre o servidor e o cliente.

- Confiável - Caso a conexão ao *WebSocket* não seja possível, a conexão retornará para a HTTP *Long polling*. E se a conexão for perdida, o cliente tentara conectar-se novamente automaticamente.

- Escalável - Escalável para vários servidores e desta forma é possível comunicar com todos os clientes conectados.

Na figura 4.16, esta a explicação, do uso da tecnologia Socket.io e a tecnologia anteriormente falada, o Redis. Usou-se esta tecnologia ao longo do estágio na realização de algumas tarefas num dos projetos trabalhados. Com se pode observar pela figura, temos vários clientes em diferentes servidores, e com estas duas tecnologias juntas consegue-se comunicar de todas as maneiras, para que desta forma se consiga manter sempre os dados e informações atualizadas.

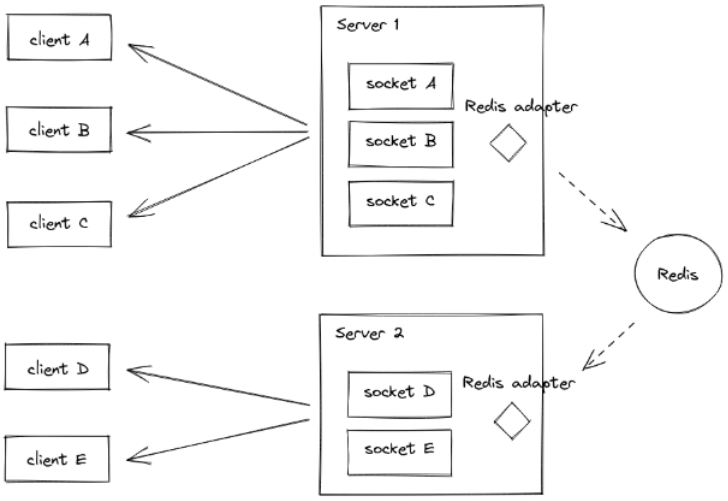

Figura 4.18: Explicação Socket.io

4.3 Ferramentas

4.3.1 MySQL Workbench

História

A primeira versão do *MySql Workbench* foi lançada em setembro de 2005, sendo o sucessor do *DB Designer 4 da fabforce.net* e substituiu o pacote de *software* anterior MySQL GUI *Tools Bundle*. *MySql Workbench* é uma ferramenta de construção de base de dados que integra o desenvolvimento, administração, construção da base de dados, criação e manutenção de SQL num único ambiente de desenvolvimento para o sistema de base de dados *MySQL*.

Este facilita a modificação da base de dados *MySQL* existentes, com funções de engenharia reversa. [27].

Figura 4.19: MySql Workbench

Principais funcionalidades e vantagens do uso do *MySql Workbench*

Existem várias vantagens e funcionalidades ao usar o *MySql Workbench* tais como [28]:

- O *MySql Workbench* é mundialmente conhecido por ser o sistema de gestão de bases de dados mais seguro e confiável usado em desenvolvimento de aplicações web.
- Disponível para várias plataformas como, Windows, Linux e OS X.
- A *interface* do utilizador é muito fácil de usar e bastante intuitiva.

- A formação e otimização de esquemas e consultas podem ser feitas usando ferramentas de visualização gráfica.
- Possibilidade de exportar em vários formatos como, PNG, PDF e SVG.

4.3.2 *SQL Server Management Studio*

O *SQL Server Management Studio* (SSMS) foi criado pela própria Microsoft e é um ambiente integrado para a gestão de qualquer infraestrutura SQL. O SSMS possui ferramentas para configurar, monitorizar e administrar instâncias do SQL Server.

Figura 4.20: SQL Server Management Studio

Principais funcionalidades e vantagens do uso do *SQL Server Management Studio*

Existem várias vantagens e funcionalidades ao usar o *SQL Server Management Studio* tais como:

- Consultar bases de dados;
- Criar, modificar e eliminar bases de dados e objetos de bases de dados;
- Criar e manter contas de utilizador e funções;
- Criar cópias de segurança das bases de dados (manuais ou programadas);
- Importar e exportar dados de/para outras bases de dados;
- Replicar bases de dados em múltiplas máquinas.

4.3.3 Visual Studio Code

História

Visual Studio Code é um editor de código-fonte (grátis) desenvolvido pela Microsoft para Windows, Linux e macOS. Este inclui suporte para depuração, controlos Git incorporado, realce de sintaxe e complementação inteligente de código. Tendo como objetivo fornecer aos programadores as ferramentas necessárias e deixar workflows mais complexos para IDEs que contenham mais recursos.

Figura 4.21: Visual Studio Code

Principais funcionalidades e vantagens do uso do Visual Studio Code

Existem várias vantagens e funcionalidades ao usar o Visual Studio Code tais como [29]:

- Suporta várias linguagens de programação. Com a utilização do Visual Studio Code os programadores não precisam de um editor diferente para diferentes linguagens de programação, pois o Visual Studio Code fornece suporte multilinguagem.
- O Visual Studio Code é multiplataforma. Portanto, pode funcionar nas plataformas mais comuns como Windows, Linux e OS X. Além disso, o código funciona nas três plataformas.
- Normalmente suporta todas as linguagens de programação, mas, se o programador quiser utilizar a linguagem de programação que não é suportada, então, facilmente

pode-se baixar uma extensão e utilizá-la. E em termos de desempenho, a extensão não diminui a velocidade do editor.

- Vem com suporte integrado para aplicações web. Assim, as aplicações web podem ser criados e suportados no Visual Studio Code.
- Vários projetos contendo vários ficheiros/pastas podem ser abertos simultaneamente. Esses ficheiros/pastas podem ou não estar relacionados entre si.
- Muitas vezes, o programador precisa de iniciar uma ação específica na raiz da pasta do projeto, e o terminal embutido fornece suporte ao programador para não alternar entre duas janelas para o mesmo.

4.3.4 Postman

História

Postman é uma plataforma de colaboração para desenvolvimento de *API (Application Programming Interface)* criada em 2012 pela Postdot Technologies e desde então o seu crescimento foi sempre aumentando. O *Postman* surgiu como uma extensão do Chrome e em 2014 foram lançadas as primeiras aplicações de cada plataforma, Windows, Linux e Mac. Os recursos do *Postman* simplificam cada etapa da construção de uma *API* e otimizam a colaboração para ser possível criar *API* com mais qualidade. Além de ser intuitivo, é gratuito e fácil de usar [30].

Figura 4.22: Postman

Principais funcionalidades e vantagens do uso Postman

Existem várias vantagens e funcionalidades ao usar o *Postman*, tais como [30]:

- *Client API* que envia solicitações REST, SOAP e GraphQL de maneira rápida e fácil diretamente no Postman.

- Teste Automatizado que automatiza os testes manuais e integra-os ao *pipeline* para garantir que nenhuma alteração de código interrompa a interface em produção.

- Monitores que mantém atualizado sobre o estado da API, verificando o desempenho e os tempos de resposta em intervalos programados.

- Espaços de trabalho que fornecem um espaço compartilhado para construir e consumir APIs e colabora em tempo real com controlo de versão integrado.

4.4 Ferramentas de Trabalho Colaborativo

4.4.1 Slack

História

A Slack *Technologies* é uma empresa fundada em 2009 por Stewart Butterfield. O Slack é uma aplicação de mensagens para empresas que conecta as pessoas às informações de que precisam. Ao reunir as pessoas para trabalhar como uma equipa unificada, o Slack transforma como as empresas comunicam.

A ferramenta Slack foi a ferramenta utilizada durante o decorrer do estágio para comunicar com o resto da empresa, sobre os projetos ou então com o suporte quando existia algum problema que não estava relacionado com os projetos [31].

Figura 4.23: Slack

Principais funcionalidades e vantagens do uso Slack

Existem várias vantagens e funcionalidades ao usar o Slack, tais como [31]:

- Possui muitas funcionalidades que incentivam o trabalho em equipa.
- Tem uma versão *lite* gratuita na qual é possível ter vários canais sem limite de participantes.
- Ter integração com outras ferramentas como Twitter, Dropbox, entre outros.
- Oferece suporte para um trabalho assíncrono. Quando o trabalho é organizado em canais, pode-se aceder as informações, independentemente da localização, fuso, horário ou função.
- É composto por vários atalhos no teclado que ao longo do dia acabam por ser bastante úteis.
- Permite a partilha de ficheiros e imagens.

4.4.2 *GitLab*

O *GitLab* foi lançado no ano de 2011 e é um serviço de controlo de versões que tem na sua base o *Git*. A principal funcionalidade do GitLab, é a possibilidade de controlar

todas as versões do projeto e a sua evolução, ou seja, é possível que o mesmo ficheiro seja modificado em simultâneo, por dois utilizadores diferentes, e que ambas as alterações sejam guardadas sem que nenhum código seja apagado. O sistema do *GitLab* ajuda também a acompanhar as mudanças feitas no código base, e, além disso, também regista quem efetuou a mudança e permite a restauração do código removido ou modificado. [32]. O *GitLab* foi utilizado durante todo o estágio, para o armazenamento e controlo dos projetos em desenvolvimento.

Figura 4.24: GitLab

Principais funcionalidades e vantagens do uso *GitLab*

Existem várias vantagens e funcionalidades ao usar o GitLab tais como [32]:

- O *GitLab* utiliza o conceito de *"branch"*, onde cada *"branch"* é uma linha do tempo que possui *commits*, e em cada *"branch"* os ficheiros podem ser alterados sem causar impacto noutras *"branch"*.
- Número de Repositórios ilimitados.
- Maior parte dos IDE dos dias de hoje já tem integração com o *GitLab*.
- Possibilidade de ter sempre acesso aos ficheiros que estão no repositório, ou seja, possibilidade de descarregar ou consultar o projeto em qualquer *desktop*.

Capítulo 5

Tarefas Desenvolvidas

5.1 Projetos

As tarefas desenvolvidas ao longo do estágio passaram por dois grandes projetos que já se encontravam em desenvolvimento na empresa onde decorreu o estágio, que foram eles o Telentry e o Clipping.

A principal função do estágio era desenvolver código e ser um *fullstack developer*, foram desenvolvidas várias tarefas ao longo de todo o estágio, como desenvolver código na *framework Vue.js*, anteriormente já explicada e demonstrada ou desenvolver em *Node.js*, já também explicada e demonstrada anteriormente. Outra das tarefas realizadas foram trabalhar com uma abundante quantidade de informação em *MySQL*, no *MySQL Workbench*. Além disso, ainda durante o estágio foi dado suporte num dos projetos desenvolvidos.

O projeto de Telentry já se encontrava em desenvolvimento à três anos, e encontrava-se numa fase de manutenção e de correção de erros. Com o intuito de melhorar ainda mais o Telentry surgiu um novo componente, este novo componente foi onde foram desenvolvidas todas as atividades durante o estágio correspondestes ao Telentry, atividades das quais irão ser explicadas mais à frente.

O projeto de Clipping já se encontrava em desenvolvimento há aproximadamente um ano e ainda faltavam implementar muitas das funções pretendidas, daí a maioria do tempo

do estágio ter passado por este projeto.

Como se pode verificar na figura 5.1, 69% do estágio foi passado no projeto de Clipping e apenas 31% no projeto de Telentry. A primeira semana do estágio foi aplicada à aprender o método de trabalho da empresa e, em simultâneo, aprender um pouco sobre as tecnologias que a empresa utilizava. Além disso, foram instaladas todas as ferramentas utilizadas pela equipa e obtidos os acessos aos projetos, como, por exemplo, acesso à base de dados dos projetos, acesso ao GitLab, para que desta forma se conseguisse o acesso a todo o trabalho já desenvolvido. Dai em diante o estágio decorreu normalmente com o desenvolvimento das tarefas que iam sendo propostas.

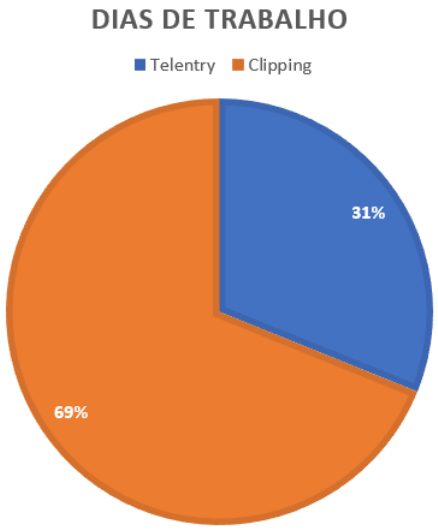

Figura 5.1: Tempo gasto nos projetos

5.2 Ferramentas e Tecnologias

5.2.1 Exemplos do uso das ferramentas e tecnlogias

Vue.js

Na figura 5.2, encontra-se um exemplo da construção de uma página na *framework Vue.js* de uma das tarefas desenvolvidas no estágio. Como se pode ver, a metodologia usada no *frontend* foi a utilização de um um "v-data-table" que esta presente numa das tecnologias usadas ao longo do estágio que foi o *Veutify*, e este "v-data-table" foi utilizado para organizar a informação que vinha do *backend*. O "data" é utilizado para fazer a declaração das variáveis, quer sejam arrays, strings ou numbers, que vão ser usados ao longo do código, após declaradas. Para chamar as variáveis utiliza-se a palavra reservada "this". No "mounted" foram chamados os métodos construidos previamente no *backend* para trazer a informação no momento inicial da abertura da página *web*, ou seja, no momento da renderização da página os métodos declarados no "mounted", também vão renderizar, pela ordem que são declarados.

No "computed" tratou-se todos os "headers" que vão ser chamados no "v-data-table". No "watch" construi-se um método responsável por reagir sempre que o utilizador tenta pesquisar sobre alguma coisa na tabela.

Já nos "methods" estão representados os restantes métodos necessários para tratar a informação e construção da página como, por exemplo, a navegação para outra página. Ao longo do estágio foram utilizadas várias propriedades da *framework Vue.js* como, por exemplo, as "props". As "props" são bastante úteis no *Vue.js* para transportar "objetos", "arrays", informações entre páginas. Além disso, também se usou, por exemplo, as "rules" definidas no "computed". As "rules" são métodos construidos com verificações que são mais tarde chamados nos componentes, ou seja, sempre que a tarefa era construir um formulário de criação de um Programa, ou Marca, haviam regras para a criação e é mesmo isso que as "rules" fazem, informam ou avisam os utilizadores quando estão, por exemplo, a preencher erradamente um filtro ou um campo de texto.

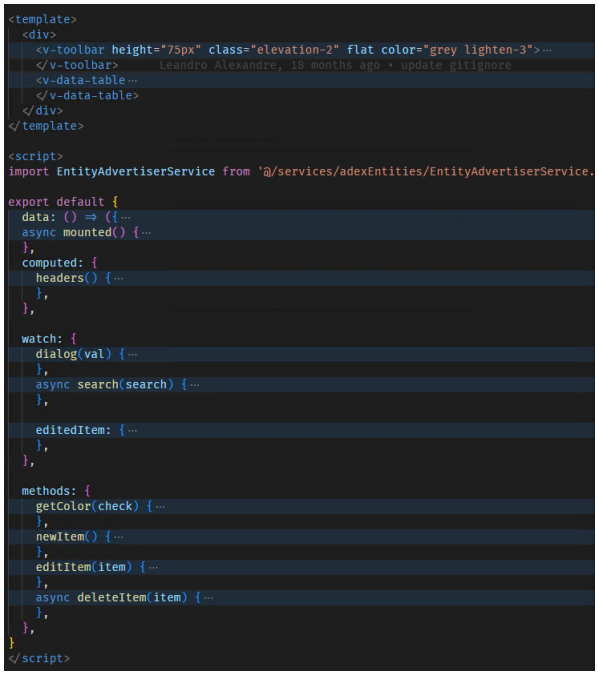

Figura 5.2: Exemplo de Código do *frontend* de uma tarefa

Node.js

Na figura 5.3, pode-se ver um exemplo da construção de uma página com o recurso à tecnologia *Node.js* e *Sequelize* utilizadas na construção do *backend* dos projetos desenvolvidos ao longo do estágio. Como se pode observar na figura estão representados os métodos mais utilizados na construção do *backend*, o "POST", "PUT", "DELETE" e os "GET's". Esta foi a metodologia usada na construção das tarefas do *backend*. Quando eram necessários métodos mais complexos, como um "GET" utilizando duas tabelas da base de dados, eram usadas "queries", previamente testadas no *MySQL Workbench*. Além disso, em algumas tarefas desenvolvidas recorreu-se a utilização destas ferramentas para à criação de migrações, ou seja, alteração ou criação de novos campos nas tabelas da base de dados dos projetos.

```
const { entity_advertiser } = require('../../models')
const db = require('../../models')

const Op = require('sequelize').Op

module.exports = {
  async index(req, res) {
    try {
      result = await entity_advertiser.findAll({})
      res.json(result)
    } catch (err) {
      res.status(500).send({
        error: 'an error has occured trying to fetch the entity_advertiser',
      })
    }
  },
```

```
async post(req, res) {
  let t = await db['adexEntities'].transaction()
  try {
    const result = await entity_advertiser.create(req.body, {
      transaction: t,
    })

    res.send(result)
    await t.commit()
  } catch (err) {
    res.status(500).send({
      error:
        'an error has occured trying to create the entity_advertiser' + err,
    })
    await t.rollback()
  }
},
```

Figura 5.3: Exemplo de Código do *backend* de uma tarefa

MySQL

Ao longo de todo o estágio o recurso a tecnologia de base de dados *MySQL* foi bastante, utilizado, na figura 5.4, pode-se ver um exemplo de uma das "queries" executadas na realização de uma das tarefas. O uso *MySQL* era utilizado diariamente, em vários cenários, por exemplo, quando havia inserções ou alterações da informação na base de dados.

```
select  '2022-09-30 09:20:00' as entrada, count(News.id) as result
          FROM News
          inner join Channels as channels on News.channel_id= channels.old_channel_id
          where News.user_id='11' and channels.id= '4'
            and News.createdAt between '2022-09-30 09:20:00' and '2022-09-30 12:25:00'
```

Figura 5.4: Exemplo de uma query

Microsoft SQL Server

Em algumas das tarefas realizadas num dos projetos a principal base de dados utilizava a tecnologia *Microsoft SQL Server*. Muito semelhante à tecnologia *MySQL*, sempre que era utilizado algum método responsável por alterar a informação na base de dados, esta tecnologia era utilizada. como se pode constatar na figura 5.5, esta representado um exemplo de uma "query" feita ao longo do estágio.

```
SELECT TOP (1000) [ID]
      ,[Data]
      ,[Canal]
      ,[Hora_Inicio]
      ,[Hora_Fim]
      ,[Sinopse]
      ,[Descricao]
      ,[Programa]
      ,[Alerta]
      ,[Valorizacao]
      ,[Valorizacao_30]
      ,[creation_date]
      ,[creation_time]
      ,[file_exist]
      ,[DataUpdate]
      ,[NeedsCut]
      ,[CopiedFrom]
  FROM [telenews_dataentry1].[dbo].[Noticia]
  where Data ='20220901'
```

Figura 5.5: Exemplo de uma *query*

Postman

O uso da tecnologia Postman foi bastante útil ao longo de todo estágio, na realização dos testes aos métodos construidos no *backend* das tarefas realizadas. Na figura 5.6, encontra-se um exemplo do uso do Postman. No método presente na figura utilizou-se um parâmetro de entrada para retornar todos os canais que estavam desativados.

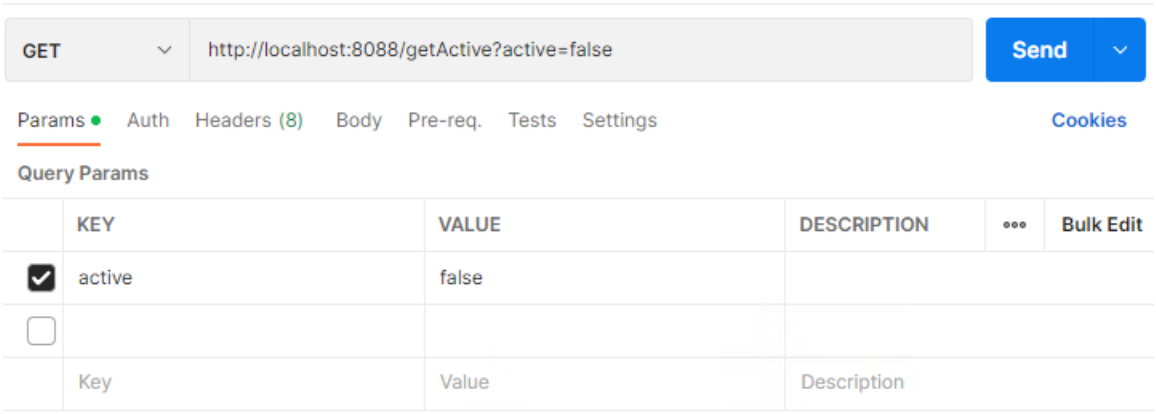

Figura 5.6: Exemplo de um pedido no Postman

Redis/Socket.io

O Redis é utilizado em ambos os projetos para fazer uma gestão das entradas e saídas dos utilizadores das aplicações. O Redis está implementado numa máquina Limux e tem sido usada por estas duas aplicações *web*. O Socket.io é utilizado em ambos os projetos.

Com o Redis e o Socket.io criou-se um ficheiro em *javaScript* capaz de guardar todas as entradas e saídas dos utilizadores em tempo real e saber em que rota se encontram os utilizadores na aplicação *web*.

Este método criado é bastante útil, pois consegue-se guardar essas informações na base e dados e, em simultâneo, criar tabelas e recolher informações sobre todo o trabalho realizado pelos utilizadores.

O exemplo da figura 5.7 explica como funciona o Redis juntamente com Socket.io num dos Projetos.

```
async init(io, socket, redis) {
  socket.on('updateRoute', (route) => {
    route = JSON.parse(route)
    if (socket.user) {
      redis.hget('users', socket.user.id, (err, u) => {
        if (err || !u) {
          console.log('\x1b[31m', err || 'User não encontrado', '\x1b[37m')
          return
        }

        u = JSON.parse(u)
        let oldRoute = u.routes.findIndex((r) => r.socket_id == socket.id)

        let settings = [
          {
            route: route.to,
            regex: gridRegex,
            type: 1,
            leave: false,
          },
        ]

        if (oldRoute != -1) {
          settings = settings.concat([
            {
              route: u.routes[oldRoute].route,
              regex: gridRegex,
              type: 1,
              leave: true,
            },
```

```
    ])

    u.routes[oldRoute].route = route.to
    redis.hset('users', socket.user.id, JSON.stringify(u))
  }

  for (const setting of settings) {
    let routeWorked = setting.regex.exec(setting.route)

    setting.regex.lastIndex = 0
    if (routeWorked) {
      console.log(routeWorked, 'routeWorked')
      let channel_id = parseInt(routeWorked[2]),
        date = new Date(routeWorked[1])
      console.log(date)
      workAccessControl(
        setting.leave,
        channel_id,
        date,
        setting.type,
        socket.user.id,
        socket.id
      )
    }
  }
```

Figura 5.7: Exemplo do uso do Redis com o Socket.io

5.3 Projeto Telentry

O Telentry é uma aplicação *web* onde o principal objetivo passa por catalogar e armazenar, todos os programas, spots (anúncios publicitários), quer para canais televisivos, quer para canais de rádio. As tarefas desenvolvidas no projeto Telentry ocuparam os dois primeiros meses do estágio e passaram pelo desenvolvimento de diversas tarefas que um programador *fullstack* desenvolve. Ao longo da primeira semana no projeto foi explicado quais eram os objetivos do novo componente e as tarefas que precisavam de ser desenvolvidas. Depois da explicação e de todas as demonstrações realizadas, começou-se com a implementação e desenvolvimento dos objetivos e tarefas propostas.

Este novo componente é um componente de gestão do Telentry, ou seja, com este componente é possível criar, alterar, eliminar, por exemplo, marcas, ou programas, utilizados pelos utilizadores para catalogar os programas ou spots que passam num determinado canal.

Além disso, as duas últimas semanas no projeto foram utilizadas a desenvolver melhorias visuais em algumas das páginas e realização de alguns *updates* em alguns componentes já existentes, tal como se irá mostrar mais a frente.

Figura 5.8: Página Login Telentry

5.3.1 Tarefas desenvolvidas

Página Anunciante

Nesta tarefa, foi pedido uma página onde fosse possível criar, editar ou eliminar Anunciantes. Primeiramente construiu-se todos os métodos no *backend*, posteriormente criaram-se os serviços para comunicar o *backend* com o *frontend*, finalizando com a construção da página no *frontend* com a *framework Vue.js*. Na figura 5.9, encontra-se a página e pode-se ver que se utilizou uma tabela, para mostrar toda a informação. Existe um botão no cimo da página que é o que faz navegar para o formulário de criação de um novo Anunciante. Além disso, se se observar a última coluna da tabela, existem dois *icons* um para eliminar o Anunciante e o outro para editar. Nesta página foram utilizados vários *icons*, responsáveis por tratar a informação vinda da tabela Anunciantes da base de dados.

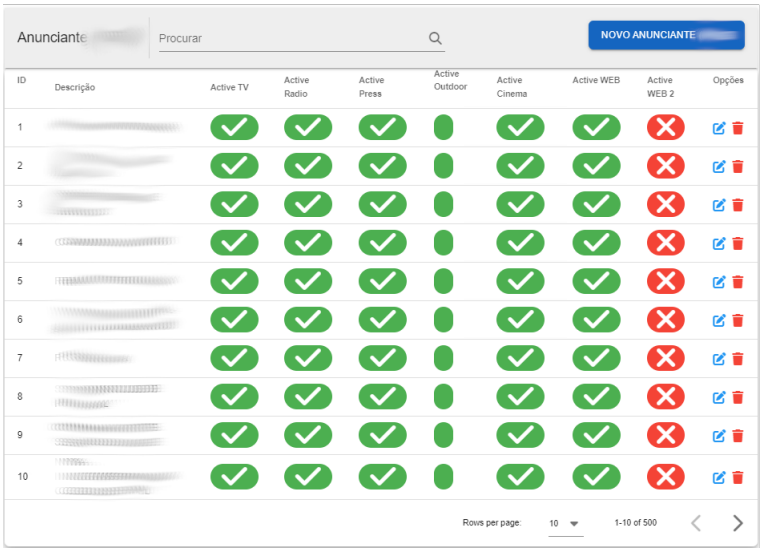

Figura 5.9: Página Anunciante

Na figura 5.10, esta representada a página da criação de um novo anunciante. Foi construido um formulário com os campos necessários para a criação. Quando preenchidos os campos o utilizador basta clicar no botão guardar, que a informação ficara registada na base de dados.

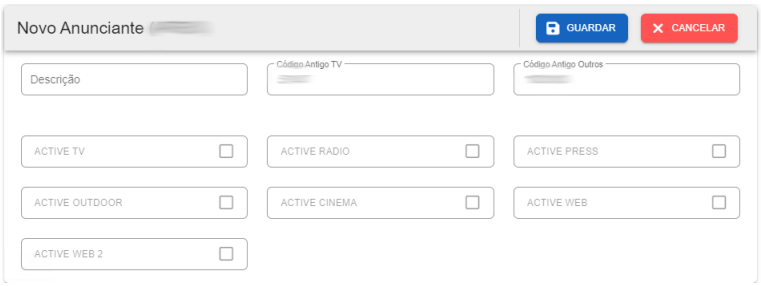

Figura 5.10: Página de criação de um novo Anunciante

Página Subclasse

A estratégia utilizada para o desenvolvimento desta tarefa foi idêntica a da página de Anunciante. Esta tarefa foi mais desafiadora, pois precisou de cruzar informações provenientes de várias tabelas da base de dados, ou seja, o tratamento da informação foi mais desafiador na construção dos métodos no *backend* e no tratamento da informação no *frontend*. Na figura 5.11, encontra-se a página de Subclasse. Pode-se ver que se utilizou novamente uma tabela para mostrar toda a informação existente na base de dados. Existe um botão no cimo da página que é o que faz navegar para o formulário de criação de uma nova Subclasse. Além disso, se se observar a última coluna da tabela, existem dois *icons* um para eliminar a Subclasse e o outro para editar.

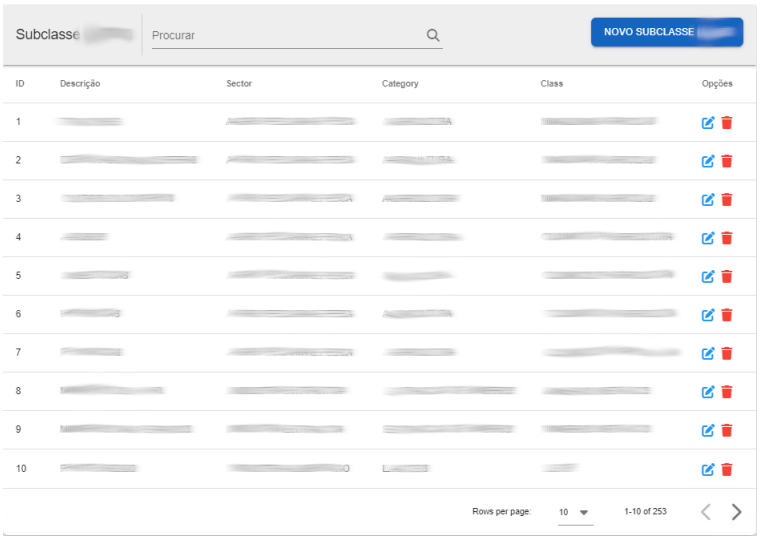

Figura 5.11: Página Subclasse

Na figura 5.12, esta representada a página de criação de uma nova Subclasse. No formulário apresentado estão os campos e filtros necessários para a criação de uma nova Subclasse. Este formulário em alguns dos campos utiliza listas de atributos correspondentes a outras páginas como é o exemplo dos Setores. Os Setores estão interligados a Categorias e as Categorias ligados a Classe. O utilizador ao preencher um Setor na lista de Categorias vão apenas aparecer as Categorias correspondentes aquele Setor e assim sucessivamente.

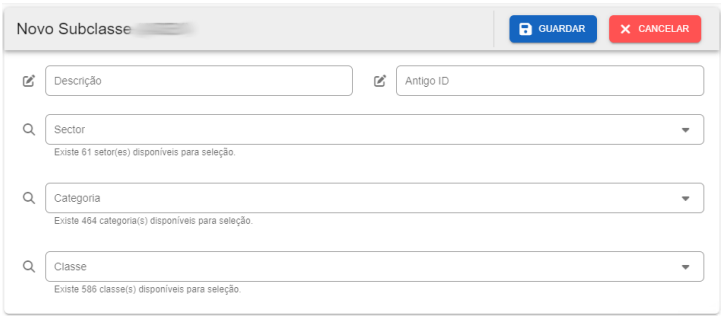

Figura 5.12: Página de criação de uma nova Subclasse

Página Marcas

Aqui a estratégia utilizada para o desenvolvimento desta tarefa foi idêntica a das páginas anteriores. Construiu-se novamente uma tabela a onde fosse possível, mostrar todas as marcas existentes na base de dados, de forma ao utilizador poder, alterar ou eliminar uma determinada marca, ou mesmo criar uma marca carregando no botão que fica no quanto superior direito. Os botões para editar ou eliminar uma marca estão representados na última coluna da tabela como se pode observar na figura 5.13.

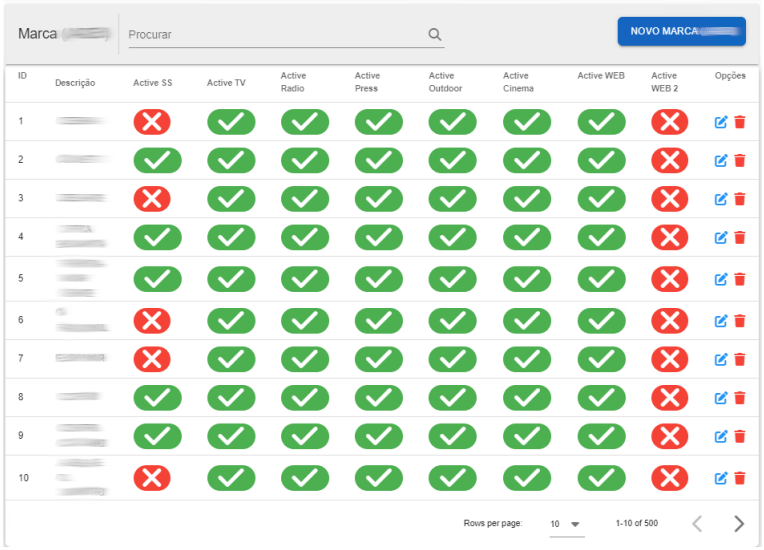

Figura 5.13: Página das Marcas

Outra tarefa também realizada foi a criação do formulário da criação de uma nova marca, como se pode observar na figura 5.14.

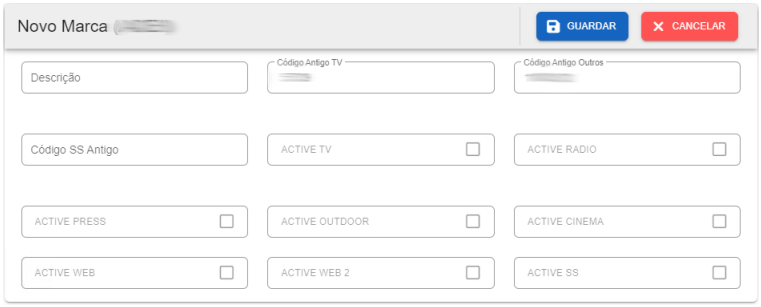

Figura 5.14: Página com o formulário de criação de uma nova Marca

Página Produtos

Outra das tarefas propostas foi a criação de uma página para mostrar os produtos. Utilizou-se mais uma vez uma tabela para mostrar todos os Produtos e os seus atributos, para que desta forma o utilizador consiga visualizar e procurar um determinado produto. Na tabela é possível editar e eliminar Produtos como também a pesquisa de um produto pelo nome. Além disso, na página existe um botão que leva o utilizador para o formulário de criação de um Produto. Pode-se observar a página produtos na figura

5.15. A construção do *backend* passou pelo desenvolvimento dos métodos comuns e a construção do *frontend* idêntico as das outras páginas já anteriormente representadas.

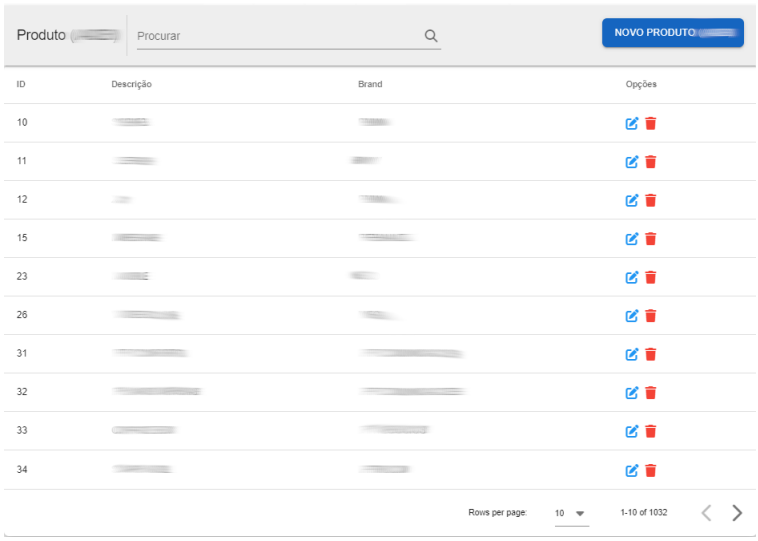

Figura 5.15: Página Produtos

A criação de um produto tem a ligação direta a uma marca, ou seja, o utilizador ao criar um produto tem que atribuir esse mesmo produto a uma determinada Marca. Ao preencher o filtro da Marca o utilizador apenas tem que escolher um valor de uma lista de marcas já previamente preparada para ser mostrada ao utilizador, ou seja, não é necessário o utilizador saber previamente qual a marca a que pertence o Produto. Observa-se o formulário de criação de um Produto na figura 5.16.

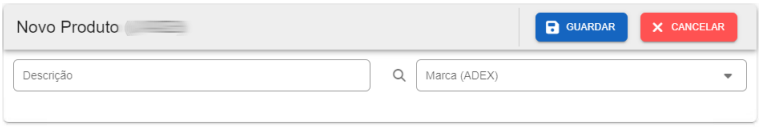

Figura 5.16: Página com o formulário de criação de um Produto

Página Sectores

Mais uma vez utilizando a mesma abordagem usada na página de Produtos, construiu-se uma tabela para todos os Setores e os seus atributos. Desta forma permite ao utilizador mais uma vez pesquisar, alterar, criar ou eliminar Setores. O botão no topo da página é responsável por carregar o formulário da criação de um Setor.

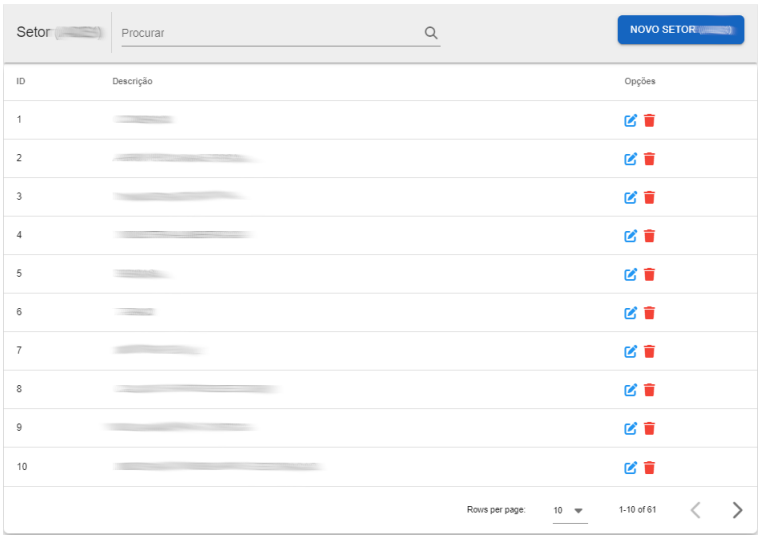

Figura 5.17: Página Sectores

Na figura 5.18, tem-se o formulário de criação de um Sector.

Figura 5.18: Página com o formulário de criação de um Sector

Página Tempos Visionamento

Na figura 5.13, está representada a funcionalidade de gerar "packs" de canais. Nesta página apenas foram adicionadas novas funcionalidades, com a construção de método no *backend* e desenvolvimento de uma página em *frontend*. Criou-se um método capaz de

dividir de igual forma um conjunto de canais. A divisão será feita conforme o número de "packs" que o utilizador escolher. Esta nova função foi criada para facilitar o trabalho dos utilizadores que manualmente distribuíam o trabalho, ou seja, os "packs" vão ser divididos baseados no tempo médio que cada utilizador demora a catalogar um canal. Após calculados os tempos médios de todos os canais, os mesmos são distribuídos pelo número de "packs" que o utilizador escolher. O número de dias utilizado para calcular o tempo médio foi de 30 dias.

Após o utilizador ter escolhido a quantidade "packs" que queria são construidos e organizados em grupos os "packs" com os canais, como já foi explicado anteriormente. Os "packs" vão ser formados com tempos de visualização idênticos entre si. Esta funcionalidade foi muito vantajosa na medida de distribuir de uma forma justa os canais pelos utilizadores responsáveis pela catalogação dos canais.

Esta página já estava criada e em utilização apenas foi acrescentada esta nova funcionalidade da criação dos "packs" e também foram feitas algumas melhorias visuais, com a utilização de componentes mais dinâmicos e utilização de CSS. Na figura 5.19, observa-se primeiramente a perguntar ao utilizador quantos "packs" quer gerar e depois os "packs" gerados.

Figura 5.19: Pop-up Gerar "packs" Canais

Na figura 5.20, observa-se a página dos "packs" criados. Na página aparecerá toda a informação relativa a cada um dos "packs" e dos canais que os compõem. Os "packs" vão aparecer já distribuídos por grupos para facilitar a leitura e organização da página. Outra funcionalidade implementada foi a possibilidade de os "packs" serem exportáveis para um ficheiro Excel. Esta função foi desenvolvida com o recurso a uma biblioteca do *Vue.js*.

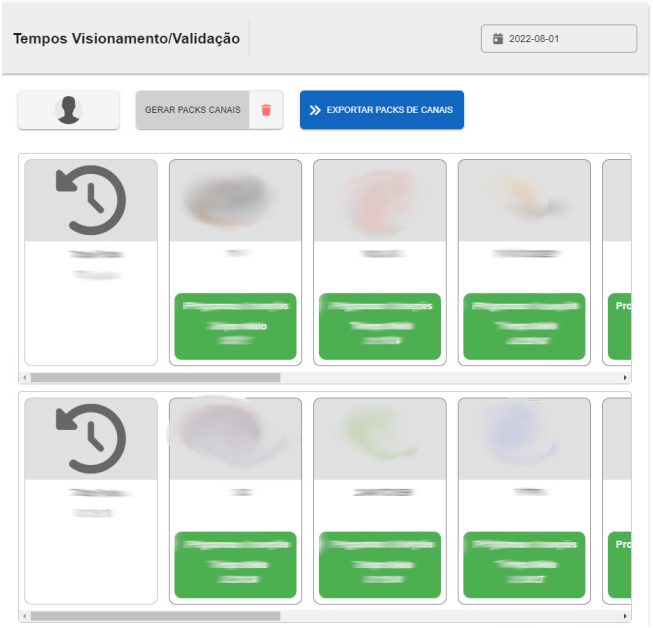

Figura 5.20: Página "packs" Canais

Página Perfis

Na figura 5.21, pode-se ver os tipos de perfis existentes no Telentry. Cada utilizador tem um perfil que determinará o que pode ou não ver dentro do Telentry. Nesta página apenas foram realizadas melhorias visuais com a organização da informação numa tabela, tendo sido também alterados os *icons* usados, para ficar mais percetível a leitura da página.

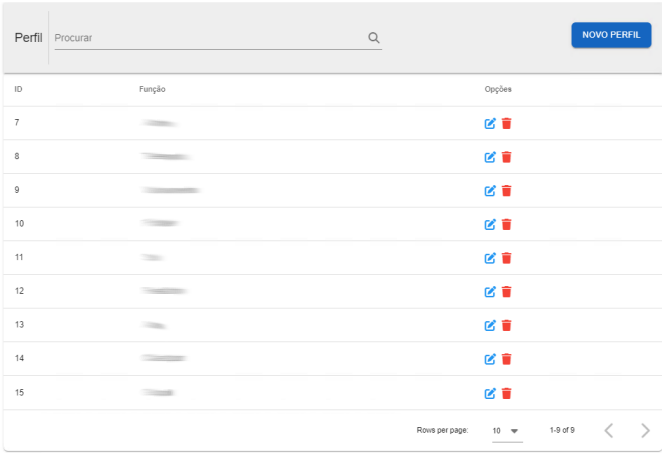

Figura 5.21: Página de Perfis

Página Associal Perfis

Na figura 5.22, pode-se ver a página de atribuição de perfis aos utilizadores. Aqui idêntico à página de Perfis, apenas foram feitas melhorias visuais, com a implementação de uma tabela e a atualização dos *icons* para uma melhor perceção da informação.

Nesta página foi adicionado a pesquisa por utilizador para ser mais rápido encontrar um determinado utilizador. Adicionou-se também um filtro capaz de organizar a tabela por Utilizador ou Perfil. Neste caso, como mostra a figura 5.22, a tabela estava organizada por utilizador, mas também podia ficar organizada por Perfil, e assim desta forma a tabela ficava organizada por Perfis.

Na segunda coluna da tabela está presente um botão que, navegava diretamente para a página representada na figura 5.23, que irá ser falada mais à frente.

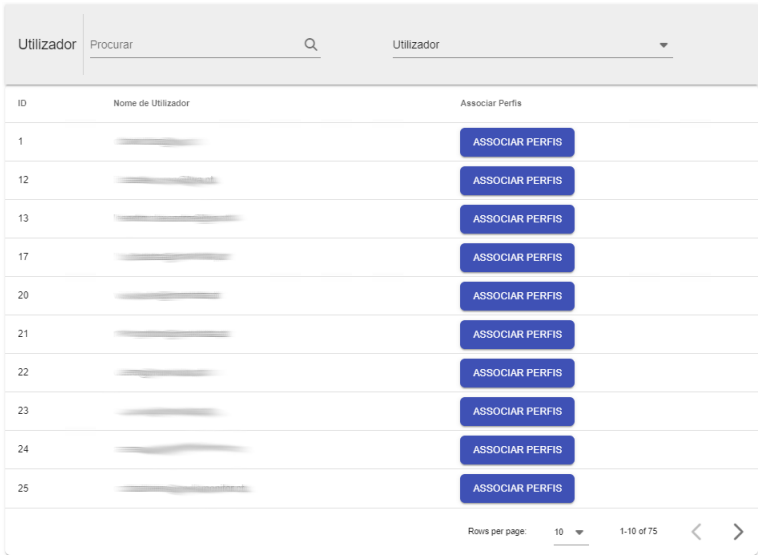

Figura 5.22: Gestão de Utilizadores e Perfis

Na figura 5.23, esta representado a forma como o utilizador atribuí um determinado perfil a um determinado utilizador. Nesta página, mais uma vez apenas foram realizadas melhorias visuais e correção de um erro no momento de guardar o perfil de um determinado utilizador.

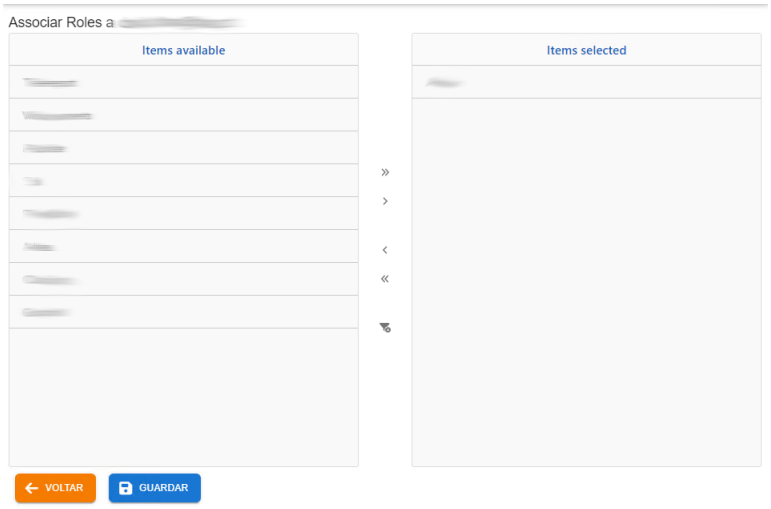

Figura 5.23: Gestão de Utilizadores e Perfis

5.4 Projeto Clipping

O Clipping é uma aplicação *web* onde o principal objetivo passa por catalogar e armazenar, todos os eventos, notícias e programas, quer para canais televisivos, quer para canais de rádio. O projeto era bastante semelhante ao projeto de Telentry, mas todas as informações produzidas eram para fins diferentes. O método de trabalho na aplicação *web* Clipping era totalmente diferente do método de trabalho de Telentry, o que levou ao desenvolvimento de funcionalidades totalmente diferentes.

As tarefas desenvolvidas no projeto Clipping ocuparam os últimos quatro meses e passaram pelo desenvolvimento de diversas funções que um programador *fullstack* desenvolve. Ao longo da primeira semana no projeto foi explicado quais eram os objetivos do Clipping, quais as tarefas que iam ser atribuídas e dai começou-se com a implementação e desenvolvimento dos objetivos e tarefas propostas.

Figura 5.24: Página Login Clipping

5.4.1 Tarefas desenvolvidas

Página Canais

Nesta tarefa, foi pedido uma página onde fosse possível editar Canais. Primeiramente construiu-se todos os métodos no *backend*, posteriormente criaram-se os serviços para o *backend* comunicar com o *frontend*, finalizando com a construção da página no *frontend* com a *framework Vue.js*. Na figura 5.25, esta representada essa página. Como se pode ver, foi utilizado uma tabela para mostrar todas as informações presentes na base de dados. Foi implementado um método capaz de atualizar o estado dos canais. Se o *icon* estiver representado com verde significa que o canal está ativado, caso o *icon* esteja a vermelho significa que o canal está desativado. Ao clicar no botão azul, representado na última coluna da tabela, ativará ou desativará o canal atualizando o seu estado na base de dados.

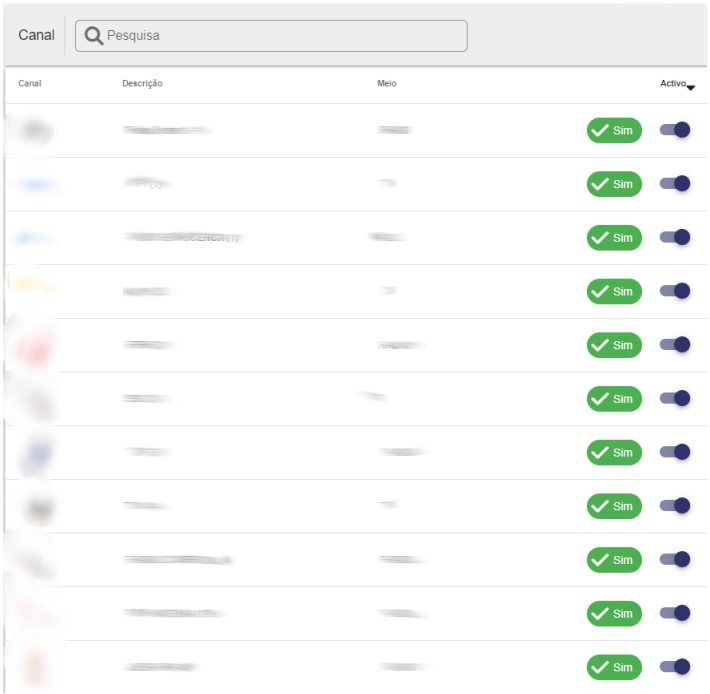

Figura 5.25: Página de Canais

Página Programas

Nesta tarefa foi pedido uma página capaz de criar, editar ou eliminar Programas. Primeiramente construiu-se todos os métodos no *backend*, posteriormente criaram-se os serviços para o *backend* comunicar com o *frontend*, finalizando com a construção da página no *frontend* com a *framework Vue.js*.

Na figura 5.26, pode-se ver a página de programas e a página de criação de um novo programa. Idêntico à página de Canais também foi usada uma tabela para mostrar todas as informações sobre os programas existentes na base de dados. Posteriormente foi criado um método capaz de filtrar os programas por canal. Quando se clica no botão para criar um programa, abre um "pop-up" com todas as informações necessárias para a criação de um novo programa.

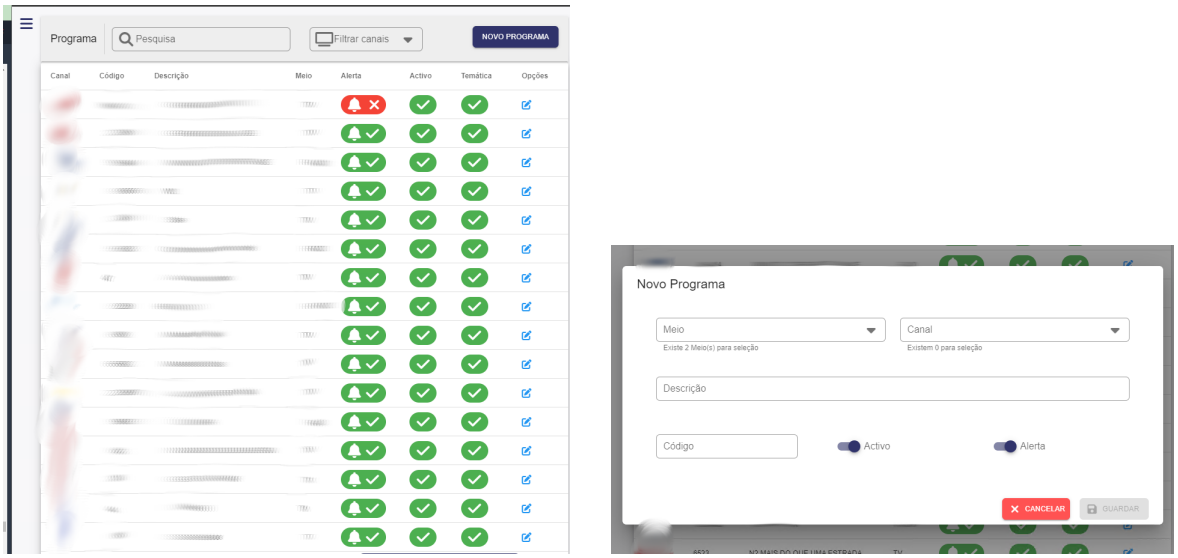

Figura 5.26: Página de Programas

Página Utilizadores

Muito semelhante à página de programas, esta representado na figura 5.27, a página de Utilizadores que foi construida com os mesmos princípios e métodos da página de Canais. A diferença é a encriptação dos dados pessoais no momento da criação de um novo Utilizador. Além disso, um utilizador administrador tem a possibilidade de eliminar um utilizador normal.

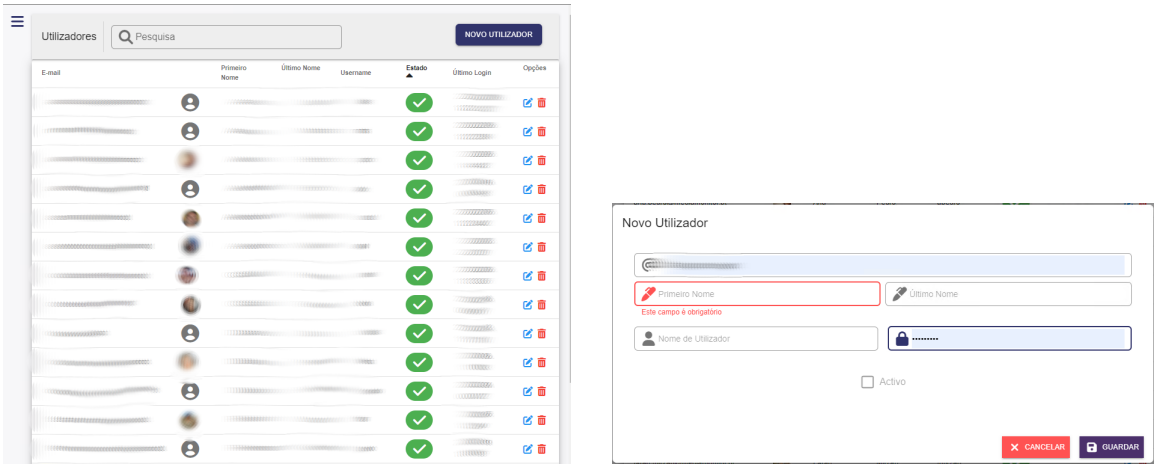

Figura 5.27: Página de Utilizadores

Página Logs Noticías

Os logs, numa aplicação *web*, são bastantes importantes e podem revelar e produzir dados relevantes em vários contextos. Neste caso foi construida uma página que mostra todos os logs relativos às notícias existentes no Clipping. Esta página é bastante importante em Clipping, pois permite aos administradores, descobrir problemas nas notícias catalogadas e, em simultâneo, como resolver esses problemas. Na construção desta página começou-se com a implementação dos métodos no *backend* capazes de filtrar os "POST", "PUT" e "DELETE" das notícias. A seguir, após testados os métodos na ferramenta Postman, construiu-se o *frontend* conforme os objetivos propostos no início da tarefa. Como se pode observar na figura 5.28, foi construida uma página com vários filtros, esses filtros podem ser preenchidos conforme as necessidades dos utilizadores.

A pesquisa pode ser feita por uma determinada data de apenas um dia, ou por um intervalo de dias. Além disso, o utilizador pode acrescentar ao filtro da data um determinado intervalo de tempo, este segundo filtro só estará disponível se o filtro da data estiver preenchido.

Para completar ainda mais a pesquisa de Logs de Notícias foi acrescentado um filtro para pesquisar por um determinado canal.

Os filtros não precisam de estar todos preenchidos, para mostrar dados na tabela, o utilizar se assim preferir pode pesquisar apenas por uma determinada data e utilizador, ou então, pesquisar por uma determinada data e um canal. Outra funcionalidade da página de Logs de Notícias é a pesquisa por um determinado "ID" de uma notícia. É possível pesquisar por um ou mais "IDS" em simultâneo. Por fim, no último filtro é possível pesquisar por uma palavra ou um conjunto de palavras que se encontrem na descrição da notícia.

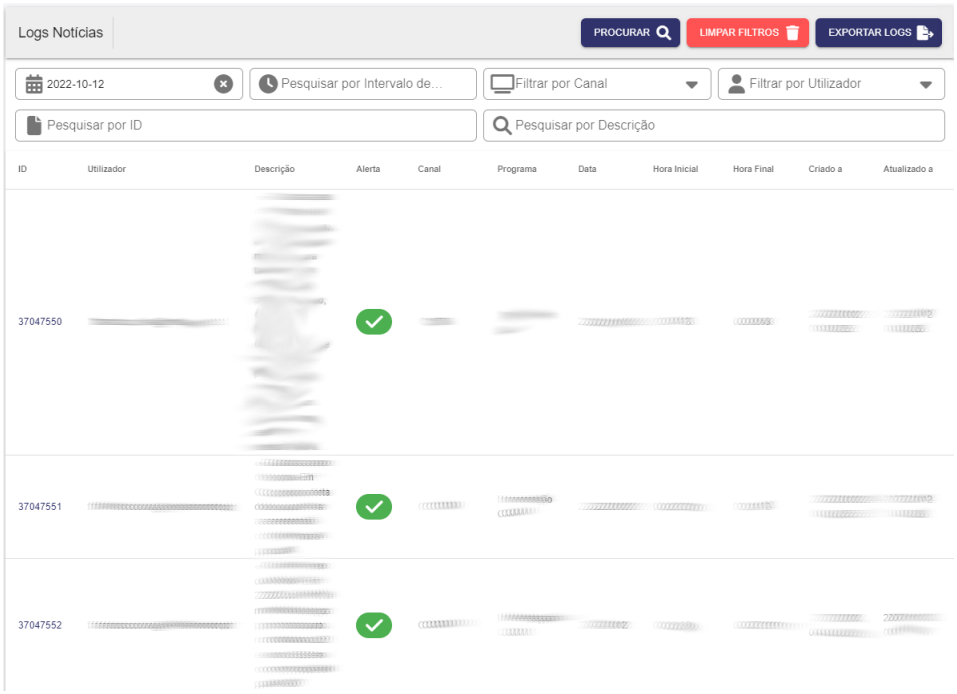

Figura 5.28: Página Logs de Notícias

Os botões que estão no lado superior direito, na figura 5.28, são responsáveis por controlar algumas funções da página como, por exemplo, a pesquisa quando os filtros

estão preenchidos, ou então limpar os filtros, este botão limpa todos os filtros que se encontram preenchidos. O último botão exporta toda a informação para um ficheiro em Excel que aparece na tabela conforme a pesquisa realizada nos filtros.

Outra funcionalidade presente na Página de Logs de Notícias é a visualização de todos os logs desde a criação de uma determinada notícia. Esta função esta disponível quando o utilizador carrega na primeira coluna na tabela em cima do campo "ID" de uma notícia. Esta função pode-se visualizar na figura 5.29.

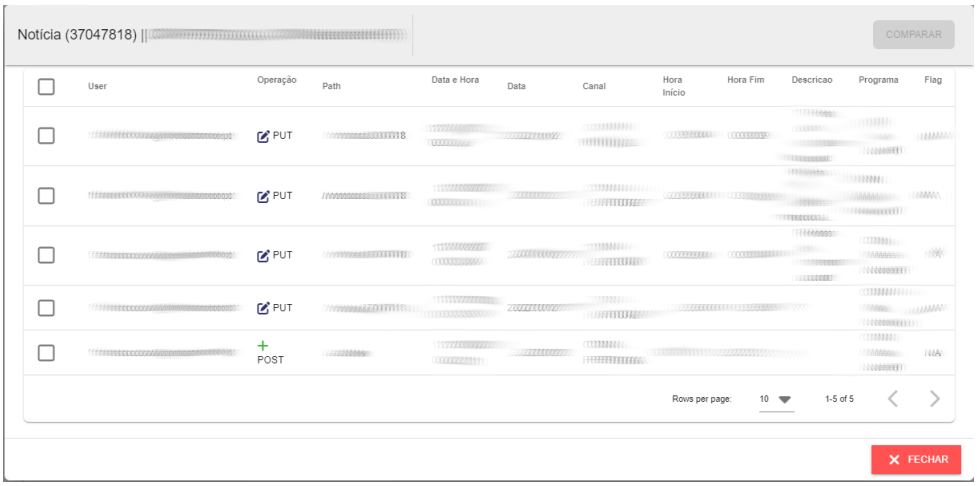

Figura 5.29: Pop-up com logs de uma notícia

Na figura 5.30, pode-se ver que ao selecionar dois logs e carregar no botão no canto superior direito abre um "pop-up" com a informação das alterações que a notícia sofreu, de um log para o outro. Permitindo assim uma melhor leitura das alterações efetuadas numa determinada notícia.

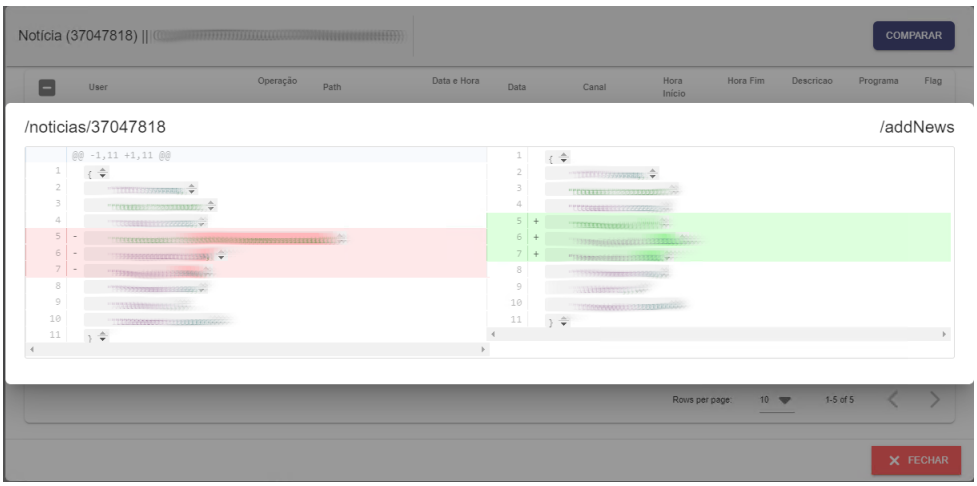

Figura 5.30: Pop-up com comparação de logs

Página Logs Globais

Os Logs Globais, são todos os métodos "POST", "PUT" ou "DELETE', ocorridos no Clipping. Para a construção desta página primeiramente implementou-se um método capaz de guardar numa tabela da base de dados todos os pedidos. De seguida construi-se todos os métodos no *backend* capazes de manipular a informação, por fim desenvolveu-se o *frontend* para mostrar essa informação. Pode-se observar a figura 5.31, página dos Logs Globais, e a onde o utilizador pode pesquisar um determinado log através de vários filtros implementados. Seja por uma data correspondente a um dia ou um intervalo de dias, seja o filtro por método, isto é, "POST", "PUT" ou "DELETE". Filtrar por utilizador, ou seja, pesquisar um log por um determinado utilizador. Esta pesquisa apenas devolve resultados se o filtro da data estiver preenchido. Como se pode ver, a pesquisa é apresentada numa tabela com toda a informação detalhada de cada log. Um dos botões no topo da página serve para pesquisar quando selecionados os filtros. Outro dos botões é para limpar os filtros e por fim o último botão serve para comparar dois logs. Este botão apenas está disponível se estiverem selecionados dois logs, os logs são selecionados na primeira coluna da tabela.

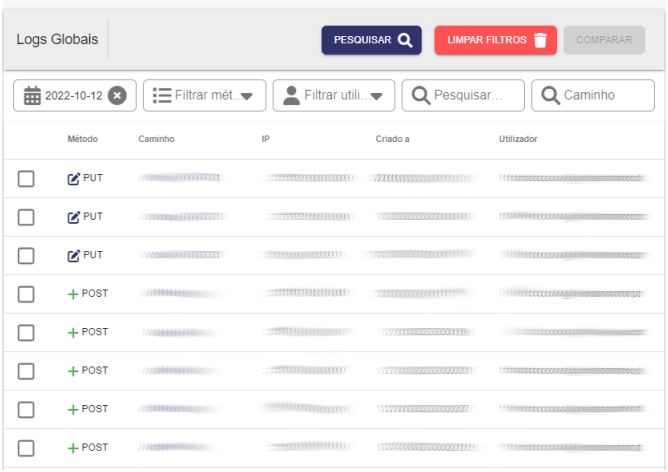

Figura 5.31: Página Logs Globais

Na figura 5.32, pode-se visualizar um pop-up com a comparação de dois logs. Esta função é bastante utilizada, pois o utilizador pode observar detalhadamente o decorrer de uma operação na aplicação *web*.

Figura 5.32: Pop-up comparação de Logs

Por vezes os utilizadores querem saber mais sobre um determinado log e com esta funcionalidade implementada, é possível. No filtro Caminho a pesquisa é feita relativamente a informações que estão na coluna da tabela "Caminho". Ao clicar em cima do caminho de um determinado log abre um pop-up com mais informações sobre aquele log.

Figura 5.33: Pop up detalhe caminho

Página Logs Eventos/Notícias

O principal objetivo do projeto de Clipping é de catalogar as notícias e eventos, por isso foi desenvolvida uma página em que estivessem visíveis todos os logs de eventos e notícias do Clipping. Idêntico à Página de Logs de Notícias, a metodologia utilizada para a construção desta página foi a mesma. A pesquisa por logs pode ser feita recorrendo a uma série de filtros implementados, apenas existe uma regra, o filtro da data tem que estar preenchido, com um dia ou intervalo de dias e o "path", ou seja, se o utilizador necessita de ver a criação, edição, eliminação de notícias ou eventos.

Outros filtros implementados foram a pesquisa por um determinado utilizador, ou seja, saber quem foi o utilizador que criou determinada notícia ou evento. Outro filtro é pelo método, ou seja, se o evento foi criado, editado ou eliminado. O filtro do canal, também pode ajudar na pesquisa dos logs, caso o utilizador saiba em que canal procurar o log, tudo isto pode ser visto na figura 5.34.

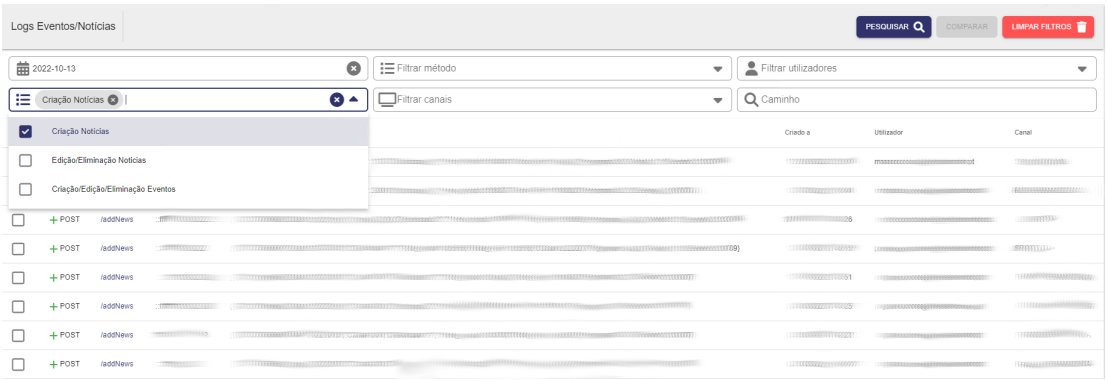

Figura 5.34: Página Logs Eventos/Notícias

O filtro de pesquisa por caminho só está disponível se existirem informações na tabela e só é visível se o utilizador clicar em cima do caminho de um determinado log. Ao clicar abre um pop-up com mais informações sobre aquele log, como se pode constatar na figura 5.35.

Figura 5.35: Pop up informação detalhada

Mais uma vez, à semelhança das outras páginas já construidas, foram criados três botões no canto superior esquerdo, a onde se encontra um botão de pesquisa, ou seja, sempre que um campo esteja preenchido. O botão de "Limpar Filtros", que limpa os filtros que estejam preenchidos, estes botões podem ser vistos na figura 5.34. Por fim, o

último botão compara dois logs. O botão "Comparar" apenas esta disponível se estiverem selecionados dois logs da tabela. Na figura 5.36, pode-se visualizar a informação de comparação de dois logs.

Figura 5.36: Comparação Logs

Página Controlo Acesso Grids

O projeto de Clipping é um projeto a onde os utilizadores trabalham por turnos durante as 24 horas, ou seja, trabalham muitas pessoas no Clipping em simultâneo, e, além disso, como todos sabemos, os últimos dois anos, foram praticamente passados com recurso ao teletrabalho, dai a necessidade de construir uma página capaz de controlar as saídas e entradas dos utilizadores e os seus movimentos dentro do Clipping para que desta forma as chefias pudessem controlar a qualidade e rendimentos dos utilizadores. Por esta razão o uso do Redis com o Socket.io foram bastante importantes no desenvolvimento desta página. Depois, através das migrações na base de dados foi criada uma migração com uma tabela nova, para guardar as informações de entrada e saída. A seguir construiram-se os métodos necessários para a manipulação da informação no *backend*. Por fim desenvolveu-se a página no *frontend*.

Na figura 5.37 pode-se ver essa página construida. A página funciona de uma maneira simples e organizada para ser de fácil interpretação. Para visualizar os acessos basta escolher uma data, sendo mostrado os dados correspondestes para essa data. Além disso, é possível também filtrar por canal e por estado. O estado do canal é explicado por dois estados, aberto ou fechado, ou seja, se um utilizador estiver naquele momento dentro

de algum canal a catalogar notícias, o estado do canal aparecerá fechado, se nenhum utilizador estiver no canal, aparece o estado de aberto. No detalhe é possível visualizar também o número de notícias que o utilizador inseriu num determinado intervalo de tempo, ou seja, durante o tempo em que esteve no canal.

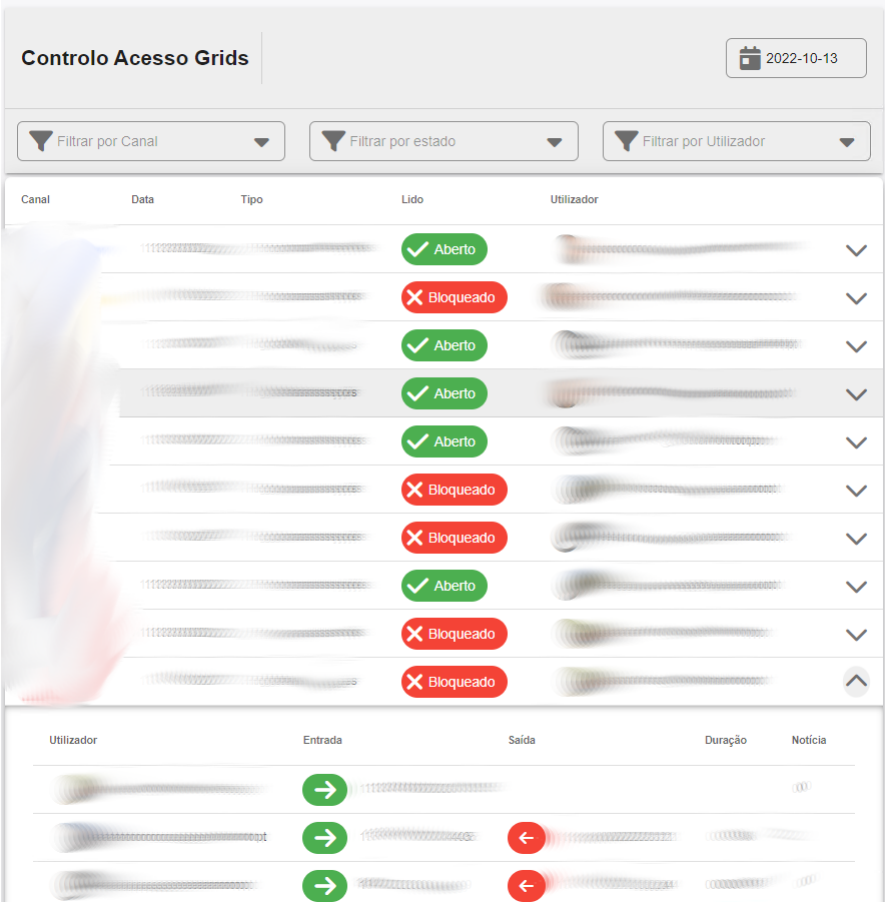

Figura 5.37: Página Controlo Acesso Grid

Página Controlo De Acessos Report

Na figura 5.38, pode-se ver o controlo de acessos, de uma forma mais detalhada e com uma disposição diferente dos elementos, ou seja, enquanto na página de cima o foco era os utilizadores, aqui nesta página o principal foco são os canais. Ao introduzir uma data são mostrados todos os canais e o tempo que os utilizadores estiveram nos canais.

Além disso, em cada um do canal é apresentado o tempo total do canal, ou seja, o tempo demorado a catalogar aquele determinado canal para aquela determinada data. Foi ainda acrescentado um filtro capaz de filtrar por utilizador. Este filtro só funciona se o filtro das datas estiver preenchido.

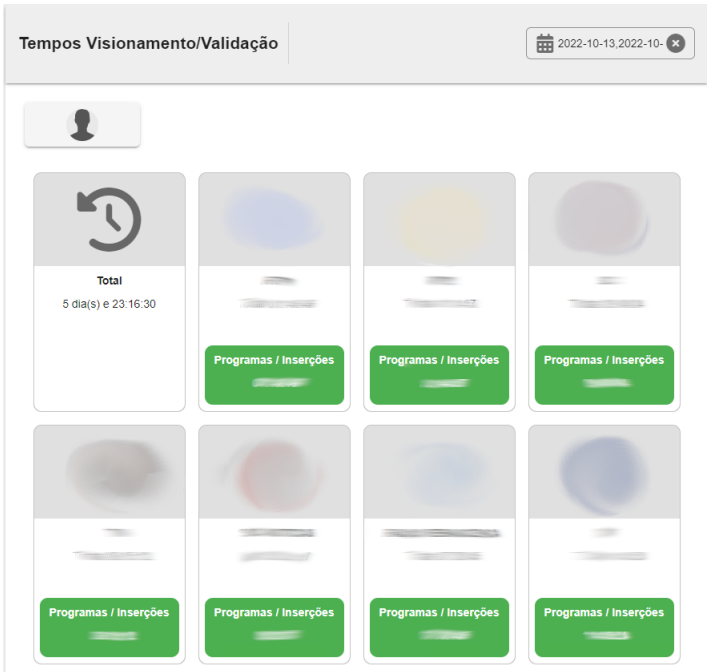

Figura 5.38: Página Controlo Report

Na figura 5.39, é apresentado o detalhe de cada canal a onde é apresentado os intervalos de tempo que cada um dos utilizadores perdeu no canal. Além disso, também são contadas as notícias efetuadas pelo utilizador naquele determinado intervalo de tempo.

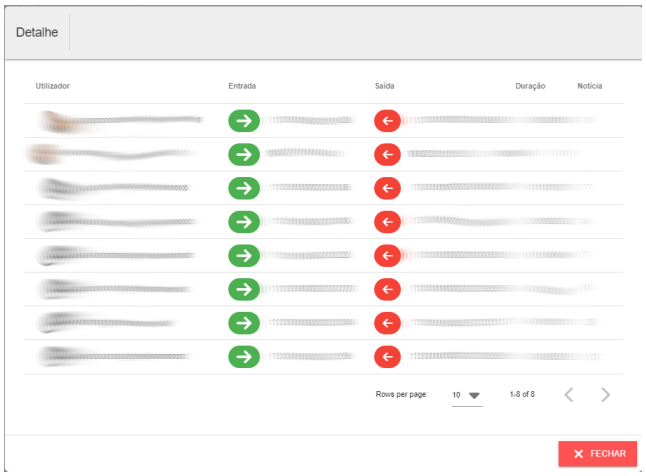

Figura 5.39: Detalhe da Página Controlo Report

Página de Rotas

As últimas tarefas do estágio passaram pelo desenvolvimento da construção das páginas para a gestão dos utilizadores e dos vários perfis. A metodologia utilizada na construção destas páginas foi idêntica a das outras tarefas. Primeiramente a construção do *backend* e depois do *frontend*. Como no Clipping trabalham diariamente vários utilizadores, iriam existir vários perfis. A existência de vários perfis, obriga a organização e distribuição das várias rotas pelos vários perfis.

A página de rotas presente na figura 5.40, apenas mostra as rotas existentes no projeto de Clipping. As rotas devem ser atualizadas regularmente enquanto o projeto se encontra em desenvolvimento, para que os utilizadores consigam ter acesso às páginas todas.

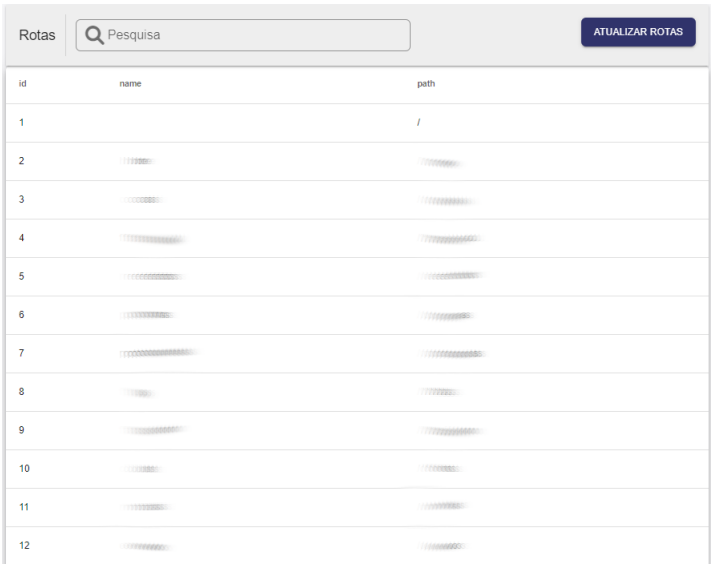

Figura 5.40: Página Rotas

Página Perfil

Tal como foi falado anteriormente, o Clipping é constituído por vários grupos de utilizadores e na figura 5.41 observam-se os perfis existentes, ou seja, cada utilizador do Clipping terá um determinado tipo de perfil. No cabeçalho da página existe o botão para a criação de um novo perfil. Já na última coluna da tabela pode-se ver dois *icons*, o azul para editar o perfil e o segundo para eliminar.

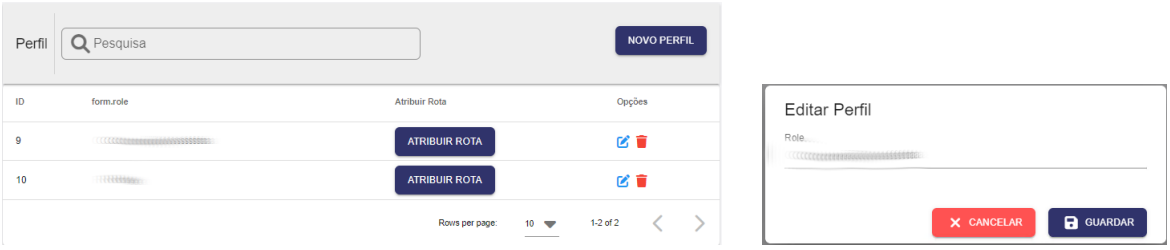

Figura 5.41: Perfis

Na figura 5.42, pode-se ver a atribuição de rotas a um determinado perfil. Conforme a figura 5.41, na tabela existe uma coluna com um botão, esse botão leva o utilizador exatamente para a página de atribuição de rotas. Pode-se selecionar apenas uma rota e mover do lado esquerdo para o direito ou então selecionar várias rotas e movê-las em simultâneo. É possível também eliminar uma determinada rota de um perfil. No lado esquerdo são as rotas existentes no Clipping e no lado direito as rotas que estão associadas aquele determinado perfil.

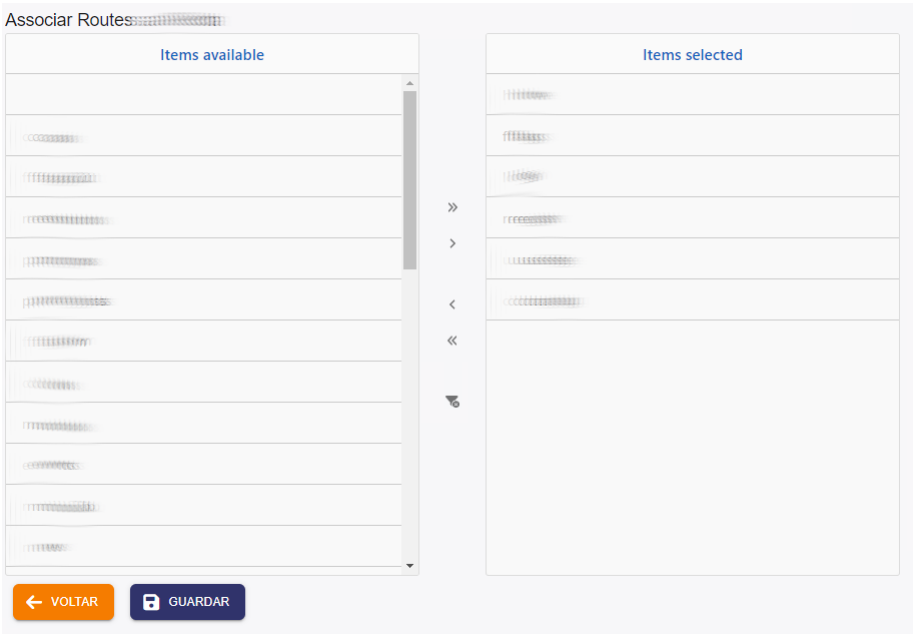

Figura 5.42: Associar Rota

Página Associar Perfis

Na figura 5.43, pode-se ver a página de associar perfis, ou seja, nesta página pode-se associar uma rota a um utilizador ou pode-se associar um perfil. Além disso, existe um filtro no cabeçalho da página a onde é possível alternar entre os utilizadores e perfis. O utilizador se selecionar "utilizador" a tabela mostra os utilizadores, mas se selecionar "perfil", a tabela mostra os perfis existentes no Clipping, tal como mostra a figura 5.44.

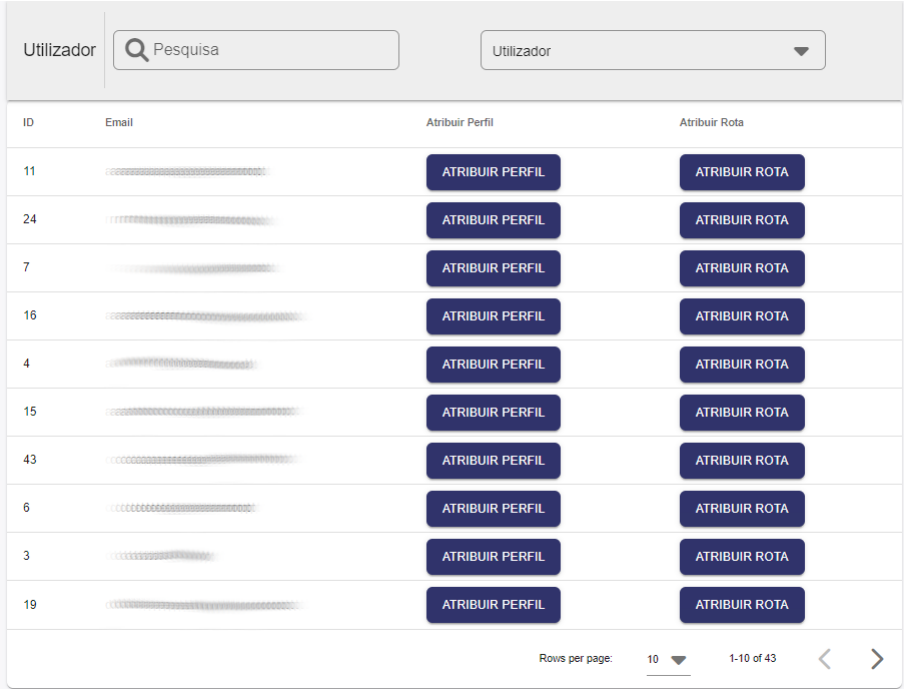

Figura 5.43: Página associar utilizador

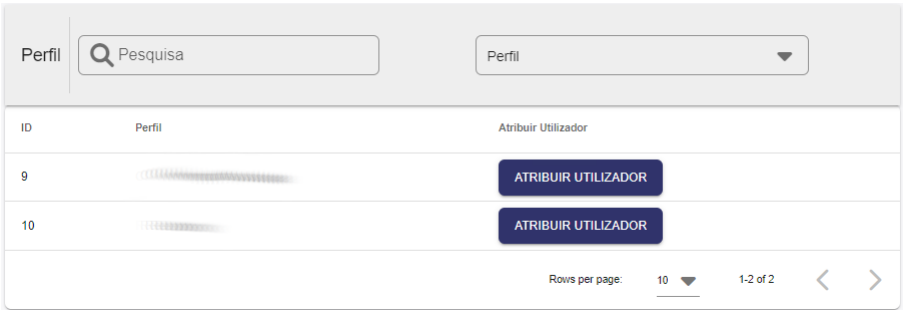

Figura 5.44: Página associar perfil

Na figura 5.45, pode-se ver a página de associar um determinado perfil a um utilizador. Para isso acontecer basta clicar na coluna da tabela atribuir perfil como esta na figura 5.43.

Para atribuir um perfil a um utilizador, basta selecionar o perfil desejado e através dos *icons* das setas mover para o lado direito. Além disso, também é possível apagar perfis.

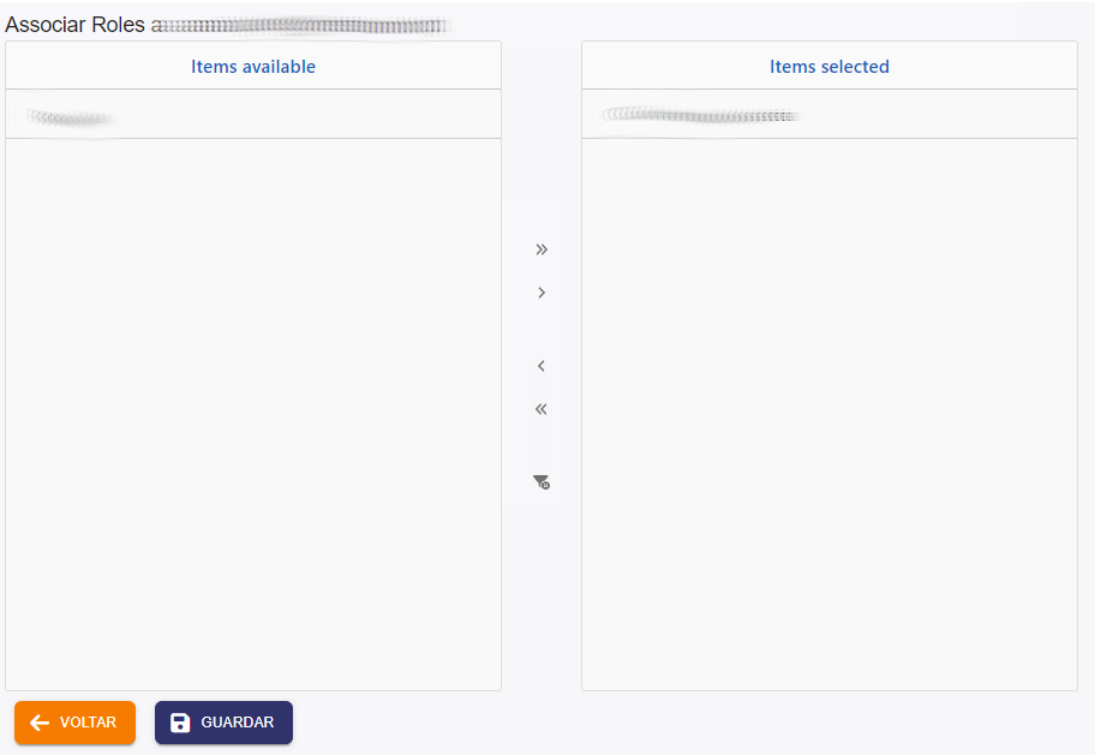

Figura 5.45: Página associar perfil a utilizador

Na figura 5.46, pode-se ver a página de associar apenas uma determinada rota a um utilizador. Esta página serve para adicionar uma rota que não se encontre no perfil onde o utilizador está inserido, ou seja, para além de o utilizador poder entrar em todas as rotas que pertencem ao perfil dele, pode também entrar na rota que se atribuir nesta página. Para associar uma determinada rota a um utilizador, basta clicar na coluna da tabela atribuir rota como esta na figura 5.43. Para atribuir uma rota basta selecionar a rota pretendida e mover para o lado direito, utilizando os *icons* com as setas existentes.

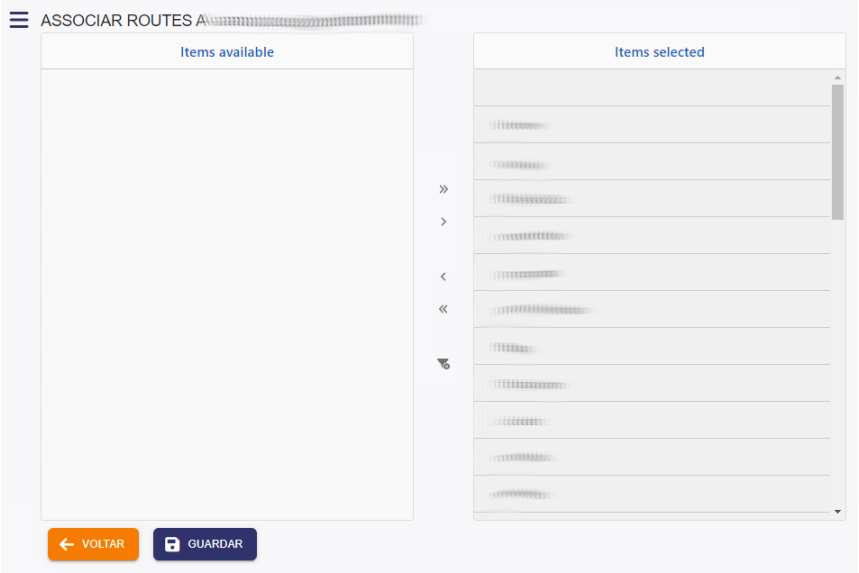

Figura 5.46: Página associar rota a utilizador

Na figura 5.47, pode-se ver a página de associar um utilizador a um determinado perfil. Esta função está disponível no botão da coluna da tabela da figura 5.44. Para atribuir um utilizador a um perfil basta selecionar o utilizador pretendido e mover para o lado direito, utilizando os *icons* com as setas existentes.

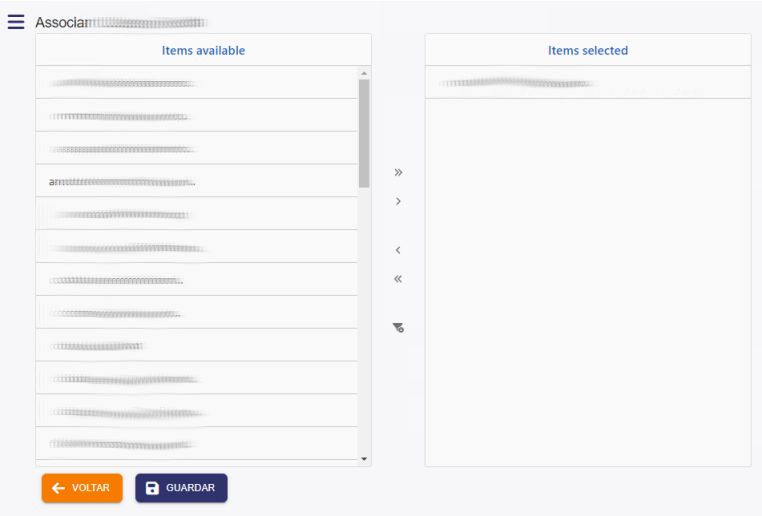

Figura 5.47: Página associar utilizador a perfil

Página Inicial

A página inicial de uma aplicação *web* é sempre bastante importante, é a primeira impressão que o utilizador tira da aplicação. Esta página foi construida e pensada para ser uma página útil e informativa. Como se pode ver, em cada um dos quadradinhos está representado um canal e o número de notícias e eventos introduzidos no dia.

Além disso, sempre que um utilizador está dentro no canal, aparece essa informação no quadrado correspondente. Esta página pode ser vista na figura 5.48.

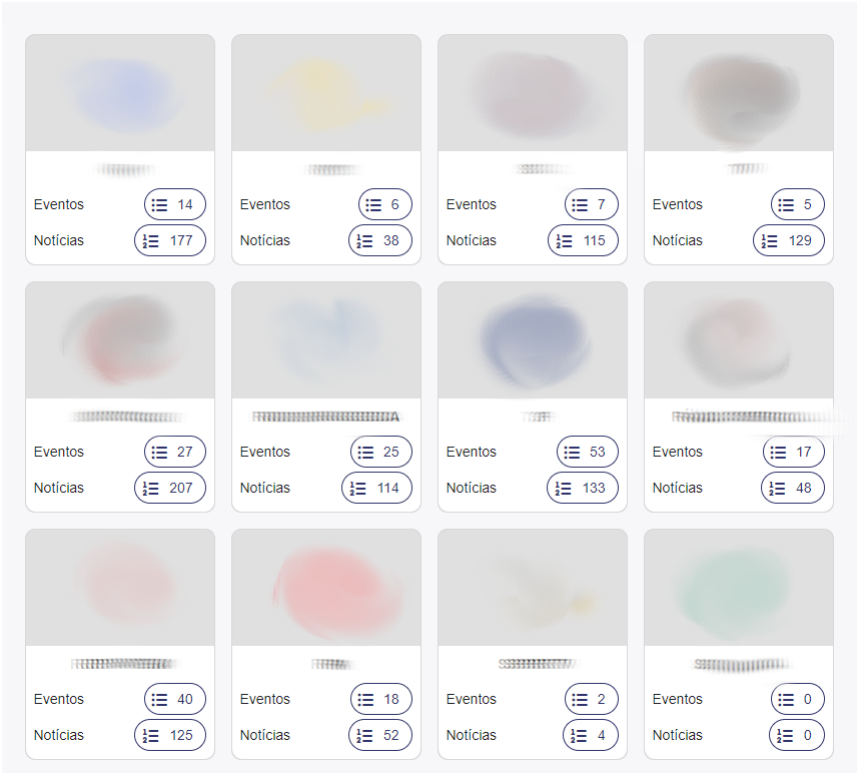

Figura 5.48: Página Inicial

Página Perfil de Utilizador

Na figura 5.49, observa-se a página de perfil do utilizar, aqui o utilizador consegue alterar as informações relativas à sua conta, como alterar o nome, a palava-passe ou a fotografia.

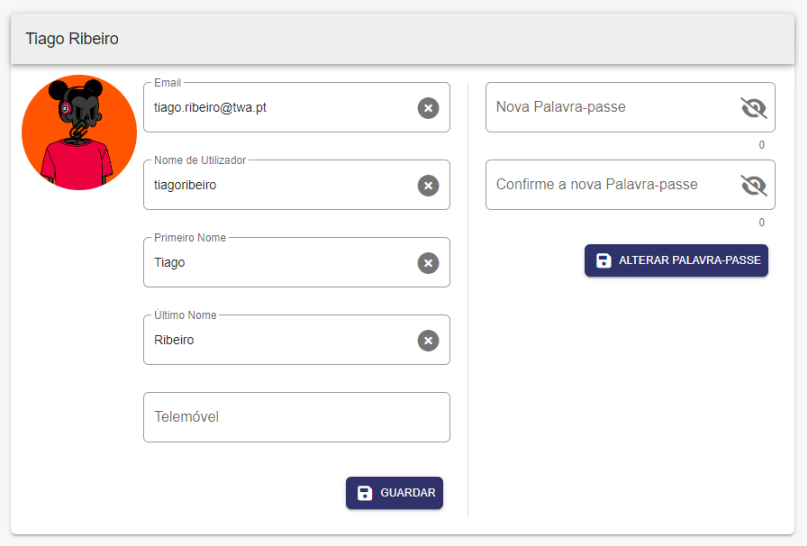

Figura 5.49: Perfil de utilizador

5.5 Diários de Trabalho das Tarefas

5.5.1 Projeto Telentry

O projeto Telentry, foi um projeto bastante desafiador e interessante. Foram realizadas tarefas e utilizadas tecnologias que nunca haviam sido usadas à data do estágio, o que tornou o desenvolvimento das primeiras tarefas um processo mais demorado. Primeiramente ainda tive que compreender e aprender as tecnologias para a realização das tarefas. As maiores dificuldades neste projeto passaram pelo desenvolvimento do *frontend*, nomeadamente no desenvolvimento da Página de Subclasse, pois envolvia a utilização e manipulação de informações vindas de outros componentes, o que precisei de algum tempo para compreender e desenvolver a página. Contudo tornou-se também a página mais desafiante e interessante de construir. Desenvolvi capacidades para a realização das tarefas seguintes. A outra tarefa a onde senti mais dificuldades foi a construção da página dos "packs" de canais, pois recorri à utilização de ciclos "for" diretamente no código HTML, o que no início se tornou confuso, mas com a ajuda da documentação e dos meus colegas de equipa consegui ultrapassar todas as minhas dificuldades. No fim consegui desenvolver todas as tarefas que me foram pedidas ao longo no estágio no projeto de Telentry.

Tabela 5.1: Tarefas Projeto Telentry

Tarefa	**Descrição da Tarefa**
Página Anunciante	• Criação de todos os métodos no *backend*; • Criação de todas as rotas; • Criação de todos os serviços para comunicação entre o *backend* e *frontend*; • Construção da página *frontend* com recurso a *framework Vue.js*;
Página Marca	• Criação de todos os métodos no *backend*; • Criação de todas as rotas; • Criação de todos os serviços para comunicação entre o *backend* e *frontend*; • Construção da página *frontend* recorrendo a *framework Vue.js*;
Página Programa	• Criação de todos os métodos no *backend*; • Criação de todas as rotas; • Criação de todos os serviços para comunicação entre o *backend* e *frontend*; • Construção da página *frontend* recorrendo a *framework Vue.js*;
Página Sector	• Criação de todos os métodos no *backend*; • Criação de todas as rotas; • Criação de todos os serviços para comunicação entre o *backend* e *frontend*; • Construção da página *frontend* recorrendo a *framework Vue.js*;

Página Subclasse	• Criação de todos os métodos no *backend*; • Criação de todas as rotas; • Criação de todos os serviços para comunicação entre o *backend* e *frontend*; • Construção da página *frontend* recorrendo a *framework Vue.js*;
Página Tempos de Visionamento/Validação	• Criação de alguns métodos *backend*; • Criação de algumas rotas; • Criação de alguns serviços para comunicação entre o *backend* e *frontend*; • Construção do método e página de gerar "packs" de canais no *frontend* recorrendo a *framework Vue.js* e melhorias visuais; • Construção da página dos "packs" de canais gerados no *frontend* recorrendo a *framework Vue.js*;
Página Perfil	• Correção de erros no *frontend*; • Verificação dos pedidos no *backend*; • Melhorias Visuais e estéticas no *frontend*;

5.5.2 Projeto Clipping

O projeto de Clipping, foi onde desenvolvi mais tarefas e a onde adquiri mais conhecimentos. Os conhecimentos adquiridos ao longo das tarefas realizadas no projeto de Telentry foram muito importantes, pois quando comecei a desenvolver as tarefas do projeto Clipping, foi só aplicar os conhecimentos adquiridos na realização de algumas tarefas, como é o exemplo da Página de Programas.

Mas mais uma vez, o ritmo de evolução no início do projeto foi menor, do que o ritmo para o fim do projeto. Foram desenvolvidas muitas tarefas com o recurso a manipulação de "arrays". A tarefa que achei mais difícil de desenvolver foi a Página de Controlo de Acessos, pois tive que implementar a parte da tecnologia Redis com o Socket.io para guardar todos os passos dos utilizadores na aplicação, que com a ajuda da documentação e dos meus colegas de equipa consegui ultrapassar todas as minhas dificuldades. Esta tarefa, para mim, foi a mais importante de todo o estágio, pois utilizei tecnologias bastantes interessantes e com muito potencial, para o desenvolvimento *web*. Outra tarefa que achei bastante interessante desenvolver foi a Página de Logs de Notícias, pois percebi que os logs numa aplicação *web*, tem bastante utilidade e pode-se produzir bastante informação importante, ou mesmo descobrir erros, recorrendo a padrões utilizados.

O suporte efetuado neste projeto também foi bastante importante, pois desenvolvi capacidades de comunicação e de trabalho em equipa. O suporte efetuado no Projeto de Clipping abrangeu tarefas desenvolvidas por mim e tarefas que já estavam desenvolvidas. Em suma, achei o projeto de Clipping bastante interessante, e com muito potencial para crescer.

Tabela 5.2: Tarefas Projeto Clipping

Tarefa	**Descrição da Tarefa**
Página Canais	• Criação de todos os métodos no *backend*; • Criação de todas as rotas; • Criação de todos os serviços para comunicação entre o *backend* e *frontend*; • Construção da página *frontend* recorrendo a *framework Vue.js*;
Página Programas	• Criação de todos os métodos no *backend*; • Criação de todas as rotas; • Criação de todos os serviços para comunicação entre o *backend* e *frontend*; • Construção da página *frontend* recorrendo a *framework Vue.js*; • Suporte e correção de erros;
Página Utilizadores	• Criação de todos os métodos no *backend*; • Criação de todas as rotas; • Criação de todos os serviços para comunicação entre o *backend* e *frontend*; • Construção da página *frontend* recorrendo a *framework Vue.js*;
Página Perfil Utilizador	• Criação de alguns métodos no *backend*; • Criação de algumas rotas; • Criação de alguns serviços para comunicação entre o *backend* e *frontend*; • Construção da página *frontend* recorrendo a *framework Vue.js*;

Tarefa	**Descrição da Tarefa**
Página Logs Globais	• Criação de um ficheiro *java Script* para guardar todos os Logs numa tabela na Base de dados; • Criação de todos os métodos no *backend* para tratar os Logs; • Criação de todas as rotas; • Criação de todos os serviços para comunicação entre o *backend* e *frontend*; • Construção da página *frontend* recorrendo a *framework Vue.js*;
Página Logs Notícias	• Criação de todos os métodos no *backend* para tratar a informação relativamente apenas aos Logs de Notícias; • Criação de todas as rotas; • Criação de todos os serviços para comunicação entre o *backend* e *frontend*; • Construção da página *frontend* recorrendo a *framework Vue.js*; • Suporte e correção de erros;
Página Logs Eventos/-Notícias	• Criação de todos os métodos no *backend* para tratar a informação relativamente apenas aos Logs de Notícias/Eventos; • Criação de todas as rotas; • Criação de todos os serviços para comunicação entre o *backend* e *frontend*; • Construção da página *frontend* recorrendo a *framework Vue.js*; • Suporte e correção de erros;

Tarefa	**Descrição da Tarefa**
Página Controlos de Acessos	• Criação de um ficheiro *JavaScript* para guardar a informação de acesso dos utilizadores, recorrendo à tecnologia do Socket.io e Redis; • Criação de todos os métodos no *backend* para a informação de acesso; • Criação de todas as rotas; • Criação de todos os serviços para comunicação entre o *backend* e *frontend*; • Construção da página *frontend* recorrendo a *framework Vue.js*
Página Controlo de Acesso Report	• Criação de todos os métodos no *backend* para tratar a informação relativamente ao controlo de acesso aos canais; • Criação de todas as rotas; • Criação de todos os serviços para comunicação entre o *backend* e *frontend*; • Construção da página *frontend* recorrendo a *framework Vue.js*;
Página Rotas	• Criação de uma migração para a criação da tabela de Rotas e desenvolvimento de todos os métodos no *backend* para tratar a informação; • Criação de todas as rotas; • Criação de todos os serviços para comunicação entre o *backend* e *frontend*; • Construção da página *frontend* recorrendo a *framework Vue.js*;

Tarefa	**Descrição da Tarefa**
Página Perfil	• Criação de uma migração para a criação da tabela de Perfis e desenvolvimento de todos os métodos no *backend* para tratar a informação; • Criação de todas as rotas; • Criação de todos os serviços para comunicação entre o *backend* e *frontend*; • Construção da página *frontend* recorrendo a *framework Vue.js*;
Página Associar Perfis	• Criação de uma migração para a criação da tabela de associar perfis e desenvolvimento de todos os métodos no *backend* para tratar a informação; • Criação de todas as rotas; • Criação de todos os serviços para comunicação entre o *backend* e *frontend*; • Construção da página *frontend* recorrendo a *framework Vue.js*;
Página Inicial	• Criação de alguns métodos *backend*; • Criação de algumas rotas; • Criação de alguns serviços para comunicação entre o *backend* e *frontend*; • Construção da página dos "packs" de canais gerados no *frontend* recorrendo a *framework Vue.js*;

Capítulo 6

Conclusões

6.1 Contributos do Estágio Curricular

O estágio realizado na TWA (Tagus World Analytics) foi um complemento adequado ao caminho de formação percorrido desde a licenciatura em Engenharia Informática até ao mestrado em Informática, pois foram usadas tecnologias importantes para o desenvolvimento *web*, permitindo também adquirir experiência ao nível da programação. O estágio na TWA, fez-me perceber ainda mais que atualmente os dados são bastante importantes e úteis, com os dados é possível encontrar padrões e realizar analogias, bastante importantes para o desenvolvimento do ser humano. Ao longo do meu estágio tive o privilégio de trabalhar com big data e transformar os dados em algo útil, dados estes que no início do meu estágio eu achava serem apenas mais dados, sem qualquer interesse. A meu ver, trabalhar na TWA foi uma mais-valia para mim e para o meu desenvolvimento como profissional na área da informática.

A utilização da tecnologia *Vue.js* permitiu ganhar competências bastante importantes na área do desenvolvimento *web*, o que acho que foi uma mais-valia para mim, pois a *framework Vue.js* esta a crescer de ano para ano. No início do estágio não tinha nenhum tipo de experiência na *framework Vue.js*, o que tornou o processo de aprendizagem e evolução mais lento, mas com dedicação e ajuda dos meus colegas de equipa, no final do

estágio já dominava as principais funções e métodos do *Vue.js*.

A utilização do *Node.js* no *backend* com a tecnologia Sequelize, foi bastante interessante e desafiadora, pois mais uma vez nunca tinha usado a tecnologia Sequelize, o que acho que é uma mais-valia para o desenvolvimento do *backend*, pois permite a simplificação da construção das "queries" e dos métodos, ajudando a avançar mais rápido um projeto.

Outra mais-valia deste estágio foi a utilização quase diariamente das tecnologias de base de dados, nomeadamente MySQL, mais uma vez permitiu-me o desenvolvimento de capacidades e de consolidação de conhecimentos.

Em suma, a experiência adquirida durante o estágio foi o desenvolvimento na capacidade de programação, pois a realidade no mercado de trabalho é bastante diferente do âmbito académico. Existem regras e processos já implementados que são obrigatórios seguir, permitindo ganhar a capacidade de adaptação a vários processos diferentes. Como a minha experiência profissional até à altura do estágio era nula, o estágio serviu para perceber o funcionamento de uma empresa, a responsabilidades que temos no cumprimento de prazos e as formações dadas sobre as plataformas. Mas mais importante que isso, este estágio ajudou-me a perceber se é mesmo isto que quero fazer ao longo da minha vida como Engenheiro Informático.

É muito importante ganhar experiencia a gerir um projeto e todo o seu desenvolvimento, além disso, desenvolver capacidades de trabalho em equipa, é uma mais-valia visto que empresas de desenvolvimento procuram muito esta qualidade nos seus novos colaboradores.

Em relação aos projetos a onde foram desenvolvidas as tarefas ao longo do estágio, agora o Telentry encontra-se em manutenção, ou seja, o projeto foi dado como concluído, apenas tem intervenções por parte do suporte. Já em relação ao projeto de Clipping, encontra-se em fase de teste e de finalização, o que faz com que os objetivos propostos no âmbito da elaboração do estágio curricular tenham sido cumpridos na íntegra.

6.2 Trabalho Futuro

Em relação ao trabalho futuro no Projeto de Telentry, deixo aqui as minhas sugestões: a primeira, passa pelo desenvolvimento de um 'bot' capaz de ajudar o utilizador sempre que este tenha uma dúvida na utilização da aplicação; a segunda passa pela melhoria de algumas páginas para simplificação de processos. Em relação ao trabalho futuro no Projeto de Clipping, a minha primeira ideia, passa pelo desenvolvimento de um método no *frontend* em que seja possível exportar todos os tipos de Logs para um ficheiro em Excel, para ser desta forma mais fácil demonstrar relatórios e informações para análise.

Bibliografia

[1] TWA. "História." (ago. de 2022), URL: https://www.linkedin.com/company/tagusworld-analytics/about/.

[2] MarkTest. "História." (ago. de 2022), URL: https://www.marktest.com/wap/grupo.aspx#destaques30anosmarktest.

[3] ——, "História." (ago. de 2022), URL: https://www.marktest.com/wap/a/q/id~c7.aspx.

[4] ——, "História." (ago. de 2022), URL: https://www.marktest.com/wap/a/q/id~c7.aspx.

[5] ——, "História." (ago. de 2022), URL: https://www.marktest.com/wap/.

[6] App. "História." (out. de 2022), URL: https://edu.gcfglobal.org/pt/informatica-basica/o-que-e-um-aplicativo-web/1/.

[7] ——, "História." (out. de 2022), URL: https://blog.betrybe.com/desenvolvimento-web/aplicacoes-web/.

[8] G. Sacramento. "Aplicação web." (out. de 2022), URL: https://rockcontent.com/br/talent-blog/aplicacao-web/.

[9] vue. "História." (ago. de 2022), URL: https://vuejs.org/.

[10] Inverita. "Vue.js." (out. de 2022), URL: https://inveritasoft.com/article-10-reasons-why-vue-js-is-so-popular.

[11] Vue.js. "Lifecycle." (out. de 2022), URL: https://vuejs.org/guide/essentials/lifecycle.html#lifecycle-diagram.

[12] Nivo. “Evolução Vue.js.” (out. de 2022), URL: https://2020.stateofjs.com/en-US/technologies/front-end-frameworks/.

[13] Veutify. “História.” (set. de 2022), URL: https://vuetifyjs.com/en/introduction/why-vuetify/#what-is-vuetify3f.

[14] H. S. T. R. Soomro. “História node.js.” (out. de 2022), URL: https://computerresearch.org/index.php/computer/article/view/1735/1719.

[15] E. TOTVS. “Vantagens node.js.” (out. de 2022), URL: https://www.totvs.com/blog/developers/node-js/.

[16] npmtrends. “Utilizações do node.js.” (out. de 2022), URL: https://npmtrends.com/node.js.

[17] NPM. “Trend Node.” (out. de 2022), URL: https://npmtrends.com/node.js.

[18] MySQl. “Características.” (out. de 2022), URL: https://dev.mysql.com/doc/refman/8.0/en/what-is-mysql.html.

[19] Statista. “Uso.” (out. de 2022), URL: https://www.statista.com/statistics/809750/worldwide-popularity-ranking-database-management-systems/.

[20] SQLSERVER. “História do MS SQL.” (out. de 2022), URL: https://www.sqlservertutorial.net/.

[21] geeksforgeeks. “Uso.” (out. de 2022), URL: https://www.geeksforgeeks.org/difference-between-structured-query-language-sql-and-transact-sql-t-sql/.

[22] Sequelize. “História do Sequelize.” (out. de 2022), URL: https://sequelize.org/.

[23] Redis. “História do Redis.” (out. de 2022), URL: https://redis.io/docs/about/.

[24] AWS. “Vantagens do Redis.” (out. de 2022), URL: https://aws.amazon.com/pt/redis/.

[25] G. Lewington. “História.” (out. de 2022), URL: https://ably.com/topic/socketio.

[26] socket.io. “Vantagens.” (out. de 2022), URL: https://socket.io/.

[27] M. Workbench. “MySQL.” (out. de 2022), URL: https://www.mysql.com/why-mysql/.

[28] P. Udhani. “MySQL.” (out. de 2022), URL: https://www.educba.com/mysql-workbench/.

[29] P. Pedamkar. “Visual Studio Code.” (out. de 2022), URL: https://www.educba.com/what-is-visual-studio-code/.

[30] Postman. “Postman.” (out. de 2022), URL: https://www.postman.com/company/about-postman/.

[31] Slack. “Slack.” (out. de 2022), URL: https://slack.com/intl/pt-pt/help/categories/200111606.

[32] GitLab. “GitLab.” (out. de 2022), URL: https://about.gitlab.com/.