



# Instituto Superior de Engenharia

Politécnico de Coimbra

DEPARTAMENTO DE INFORMÁTICA E SISTEMAS

## Block Explorer para a Tecnologia DLT Hashgraph

Relatório de Licenciatura em Engenharia Informática

Autor

**Carlos Manuel Pascoal Pinto**

Orientador

**João Durães**

Supervisor na empresa

CO<sub>2</sub>Offset

**Pedro Cipriano**

Coimbra, mês e ano



INSTITUTO POLITÉCNICO  
DE COIMBRA

INSTITUTO SUPERIOR  
DE ENGENHARIA  
DE COIMBRA



## **AGRADECIMENTOS**

Gostaria de expressar os meus sinceros agradecimentos a todas as pessoas e entidades que contribuíram para o desenvolvimento e conclusão deste estágio.

Primeiramente, desejo agradecer à empresa CO<sub>2</sub>Offset pela oportunidade de realizar este estágio e pelo apoio contínuo ao longo deste período. Agradeço especialmente aos colegas de trabalho pelo ambiente colaborativo e pelo constante incentivo ao crescimento profissional.

Um agradecimento especial ao meu orientador, João Durães, pela orientação, apoio e valiosas sugestões ao longo do projeto. A sua experiência e conhecimento foram fundamentais para o sucesso deste trabalho.

Também gostaria de expressar a minha gratidão ao meu supervisor, Pedro Cipriano, pela disponibilidade, pela partilha de conhecimento e pela orientação técnica ao longo do estágio. O seu apoio foi fundamental para enfrentar os desafios e alcançar os objetivos estabelecidos.

Por fim, um agradecimento especial à minha família, especialmente à minha mãe e amigos pelo apoio incondicional ao longo deste percurso.

Obrigado a todos pela vossa contribuição e pelo papel crucial que desempenharam durante o meu estágio.



## **ABSTRACT**

In the global carbon credit market, the need for transparency and traceability of transactions has become crucial in the fight against climate change. While blockchain technology, widely used for this purpose, faces criticism due to its high energy consumption, promising alternatives like Hedera Hashgraph have emerged. This technology offers a more efficient and sustainable consensus mechanism while maintaining the security and integrity of transactions.

With these challenges in mind, CO<sub>2</sub>Offset proposed developing a web application that would allow users to transparently and immutably monitor carbon credit transactions certified by the company. The strategic choice of Hedera Hashgraph to ensure verifiability and reliability of information was essential, meeting the rigorous transparency requirements of the market. This initiative not only enhances the trust of investors and stakeholders but also strengthens the commitment to sustainable and responsible practices.

This project aims not only to improve transparency and efficiency in the carbon credit market but also demonstrates a clear commitment to environmentally responsible practices. The implementation of Hedera Hashgraph not only optimizes transaction security and reliability but also contributes to a more sustainable and resilient future.

**Keywords:** Carbon credits, Web Application, Hedera Hashgraph, Sustainability, Blockchain



## RESUMO

No mercado global dos créditos de carbono, a necessidade de transparência e rastreabilidade das transações tem ganho destaque como uma ferramenta crucial na luta contra as mudanças climáticas. Enquanto a tecnologia blockchain, amplamente utilizada para este fim, enfrenta críticas devido ao seu elevado consumo de energia, surgem alternativas promissoras como a *Hedera Hashgraph*. Esta tecnologia oferece um mecanismo de consenso mais eficiente e sustentável, mantendo a segurança e a integridade das transações.

Com estes desafios em mente, a CO<sub>2</sub>Offset propôs o desenvolvimento de uma aplicação web que permitisse aos utilizadores monitorar de forma transparente e imutável as transações de créditos de carbono certificadas pela empresa. A escolha estratégica da *Hedera Hashgraph* para garantir a verificabilidade e a confiabilidade das informações foi essencial, respondendo às exigências rigorosas de transparência do mercado. Esta iniciativa não apenas fortalece a confiança dos investidores e stakeholders, mas também reforça o compromisso com práticas sustentáveis e responsáveis.

Este projeto não apenas visa melhorar a transparência e a eficiência no mercado de créditos de carbono, mas também demonstra um compromisso claro com práticas ambientalmente responsáveis. A implementação da *Hedera Hashgraph* não só otimiza a segurança e a confiabilidade das transações, mas também contribui para um futuro mais sustentável e resiliente.

**Palavras-chave:** Créditos de carbono, Aplicação Web, Hedera Hashgraph, Sustentabilidade, Blockchain





## ÍNDICE

Agradecimentos .....	i
Abstract.....	iii
Resumo .....	v
Índice de Figuras .....	ix
Índice de siglas e acrónimos.....	xi
1 Introdução .....	1
1.1 Instituto Superior de Engenharia de Coimbra .....	1
1.2 CO <sub>2</sub> Offset.....	2
1.2.1 Modelo de Negócio da CO <sub>2</sub> Offset.....	3
1.3 Estrutura do relatório .....	4
2 Metodologia de trabalho .....	7
2.1 YouTrack.....	8
2.2 Figma .....	10
2.3 Docker.....	11
2.4 GitHub e Branching .....	13
2.4.1 GitHub.....	13
2.4.2 Modelo de Branching.....	14
2.5 Postman.....	16
3 Conceitos .....	19
3.1 Distributed Ledger Technologies (DLTs) .....	20
3.1.1 Blockchain .....	22
3.1.2 Hashgraph .....	30
3.1.3 Porquê a Hashgraph.....	35
3.2 Créditos de carbono.....	38
3.3 Vite + React.....	41
4 Planeamento do projeto.....	45
4.1 Objetivos do Projeto .....	45
4.2 Análise de Requisitos .....	46
4.2.1 Requisitos Funcionais .....	46
4.2.2 Requisitos Não Funcionais.....	47
5 Desenvolvimento do block explorer.....	49

5.1	Design.....	49
5.2	Construção do Website: Do Design à Realidade .....	60
5.2.1	Setup inicial .....	61
5.2.2	Desenvolvimento do código .....	63
5.2.3	Resultados finais .....	73
6	Conclusões e apreciações finais .....	79
7	Referências.....	81
8	Anexos.....	85
	Anexo A.....	85
	Anexo B.....	85
	Anexo C.....	86
	Anexo D .....	87

## ÍNDICE DE FIGURAS

Figura 1 – Esquema de uma DLT descentralizada P2P .....	21
Figura 2 - Hashgraph no estado inicial .....	31
Figura 3 - Hashgraph após uma iteração .....	31
Figura 4 - Conteúdo de um evento .....	31
Figura 5 - Hashgraph após 3 iterações.....	32
Figura 6 - Rondas e Eventos testemunhas da Hashgraph.....	32
Figura 7 - Tabela comparativa entre redes DLT' .....	37
Figura 8 -Primeiro design da página inicial .....	51
Figura 9 - Design final da página inicial.....	52
Figura 10 - Design da página "Transactions".....	53
Figura 11 - Design da página referente aos detalhes de uma transação .....	54
Figura 12 - Design da página pools.....	55
Figura 13 - Design da página referente aos detalhes de uma pool.....	56
Figura 14 - Design do Dashboard.....	57
Figura 15 - Design da página Stats .....	58
Figura 16 - Design da página About .....	59
Figura 17 - Resultado final do Header.....	73
Figura 18 - Resultado final do Footer.....	73
Figura 19 - Resultado final da Página Inicial.....	74
Figura 20 - Resultado final da página transactions .....	74
Figura 21 - Resultado final da página pools .....	75
Figura 22 - Resultado final da página details?Transactions .....	75
Figura 23 - Resultado final da página details?Pool .....	76
Figura 24 - Resultado final da página Sobre.....	76
Figura 25 - Resultado final de PageNotFound .....	77



## **ÍNDICE DE SIGLAS E ACRÓNIMOS**

**API:** Application Programming Interface

**CO<sub>2</sub>:** Dióxido de carbono

**DLT:** Distributed Ledger Technology (Tecnologia de Registo Distribuído)

**EVM:** Ethereum Virtual Machine

**GEE:** Gases de Efeito Estufa

**HBAR:** Token nativo da rede Hashgraph

**HCS:** Hedera Consensus Service

**HMR:** Hot Module Replacement

**HTS:** Hedera Token Service

**IC:** Implementação Conjunta

**ISEC:** Instituto Superior de Engenharia de Coimbra

**MDL:** Mecanismo de Desenvolvimento Limpo

**PoS:** Proof of Stake

**PoW:** Proof of Work

**P2P:** Peer-to-Peer

**PNG:** Portable Network Graphics

**REST API:** Representational State Transfer Application Programming Interface

**TCP/IP:** Transmission Control Protocol/Internet Protocol



# 1 INTRODUÇÃO

Este documento tem como objetivo apresentar o trabalho realizado durante o estágio na empresa CO<sub>2</sub>Offset, focando-se na criação de um website que torna públicas todas as transações da empresa, desde a criação de um crédito até à sua venda. O estágio teve como objetivo a criação de uma plataforma que permita aos clientes e parceiros monitorizar e verificar todas as transações de forma transparente.

A transparência é fundamental para garantir a confiança e a integridade das operações da CO<sub>2</sub>Offset. Para alcançar este objetivo, foi utilizada a tecnologia *Hedera Hashgraph*, uma alternativa inovadora às *blockchains* tradicionais. A Hedera Hashgraph não só garante a segurança e a rastreabilidade dos dados, mas também consome menos energia, alinhando-se com os princípios de sustentabilidade da empresa.

O desenvolvimento do website envolveu várias etapas, desde a conceção inicial até à implementação final, enfrentando e superando diversos desafios técnicos. Este relatório detalha todo o processo, explicando as tecnologias utilizadas e os benefícios da nova plataforma para a CO<sub>2</sub>Offset.

Ao tornar as transações transparentes e acessíveis, o website reforça a confiança dos clientes e promove a fidelidade à empresa. Este projeto demonstra como é possível combinar transparência e sustentabilidade através da adoção de tecnologias avançadas, proporcionando uma solução eficiente e confiável para a gestão de créditos de carbono.

## 1.1 Instituto Superior de Engenharia de Coimbra

O Instituto Superior de Engenharia de Coimbra (ISEC) é uma instituição de ensino superior pública portuguesa com uma longa e rica tradição no ensino da engenharia e tecnologia. Fundado oficialmente em 1974, o ISEC tem as suas raízes no antigo Instituto Industrial e Comercial de Coimbra, estabelecido em 1921, o que lhe confere mais de um século de experiência no ensino técnico. Em 1988, o ISEC foi integrado no Instituto Politécnico de Coimbra (IPC), tornando-se uma das suas unidades orgânicas mais proeminentes.

A missão principal do ISEC é alcançar o reconhecimento nacional e internacional pela excelência na qualidade do ensino. Para isso, a instituição investe fortemente na valorização do seu corpo docente e estabelece ligações sólidas com o mundo empresarial. Um exemplo notável desta abordagem é a cooperação com o programa

"Trilhos", que visa enriquecer os currículos académicos dos alunos e facilitar a sua integração no mercado de trabalho.

O ISEC oferece uma vasta gama de programas educativos, incluindo 12 licenciaturas, 9 mestrados e 18 cursos técnicos superiores profissionais, abrangendo diversas áreas da engenharia como bioengenharia, engenharia civil, informática, eletrotécnica e mecânica. A instituição está organizada em 6 departamentos e conta com uma comunidade estudantil de 3357 alunos.

Um dos pontos fortes do ISEC é a sua notável taxa de empregabilidade. Estatísticas internas revelam que 70% dos alunos conseguem entrar no mercado de trabalho antes de concluírem o curso, e 98% obtêm emprego nos primeiros 6 meses após a graduação. Esta performance impressionante torna o ISEC uma referência nacional e uma escolha preferencial para estudantes de engenharia.

O prestígio e a qualidade do ensino do ISEC refletem-se na elevada procura por parte dos candidatos ao ensino superior. Nos últimos quatro anos, as vagas oferecidas pelo instituto têm sido consistentemente preenchidas na primeira fase de candidaturas, demonstrando a atratividade dos seus programas.

Em suma, o Instituto Superior de Engenharia de Coimbra destaca-se como uma instituição de ensino superior de referência na área da engenharia em Portugal. Com a sua história centenária, oferta formativa diversificada e forte compromisso com a qualidade do ensino e a empregabilidade dos seus graduados, o ISEC continua a desempenhar um papel crucial na formação de engenheiros qualificados e na promoção do desenvolvimento tecnológico e económico do país.[19]

## **1.2 CO<sub>2</sub>Offset**

A CO<sub>2</sub>Offset é uma startup inovadora fundada em 2021 em Ansião, Leiria, que se destaca no mercado voluntário de créditos de carbono. Esta jovem empresa diferencia-se pela utilização de tecnologias de ponta, como as tecnologias DLT e a inteligência artificial, para melhorar, agilizar e automatizar os seus processos.

Através destas tecnologias avançadas, a CO<sub>2</sub>Offset desenvolveu uma ferramenta que permite calcular, com um alto grau de precisão, a quantidade de dióxido de carbono sequestrado numa determinada área florestal. Este cálculo rigoroso dá origem a créditos de carbono, que podem ser vendidos a empresas, entidades ou até mesmo a particulares interessados em ser recompensados por compensar a sua pegada de carbono.

Em março de 2023, a CO<sub>2</sub>Offset recebeu um significativo investimento da GED Ventures, uma sociedade de capital de risco. Esta injeção de 1,5 milhões de euros foi realizada através do fundo GED Tech Seed Fund, destinado a empresas de tecnologia, em troca de 23% da empresa. Na ocasião, a GED Ventures divulgou ao Jornal de Negócios, uma subdivisão do Jornal de Notícias, que "numa altura em que o mundo ambiciona atingir as metas definidas pelo



Acordo de Paris e pela Agenda 2030 das Nações Unidas, acreditamos que a solução pioneira desenvolvida pela CO<sub>2</sub>Offset pode dar um importante contributo para alcançarmos este ambicioso objetivo".

A CO<sub>2</sub>Offset, com o seu espírito inovador e comprometimento com a sustentabilidade, está posicionada para se tornar uma líder no mercado de créditos de carbono, contribuindo significativamente para a luta contra as mudanças climáticas e para a concretização das metas ambientais globais. [20]  
[21]

### **1.2.1 Modelo de Negócio da CO<sub>2</sub>Offset**

A CO<sub>2</sub>Offset opera como um SaaS (Software as a Service), disponibilizando uma plataforma online que permite a interação de dois tipos de clientes: provedores de serviço e proprietários de terrenos (land owners).

Os proprietários de terrenos, sejam particulares ou empresas, podem submeter os seus terrenos e a documentação necessária na plataforma. Uma vez aprovados, estes proprietários ficam habilitados a receber compensações monetárias mediante a realização de certas ações nos seus terrenos. Estas ações, inicialmente sugeridas pela CO<sub>2</sub>Offset, serão no futuro recomendadas de forma semi ou totalmente automática através de ferramentas que utilizam inteligência artificial. Entre as ações recomendadas, podem estar limpezas florestais, adiamento de corte, etc. (Nota: Completar com as restantes ações).

Estas ações têm de ser realizadas pelos provedores de serviço, que pagam para executar estas tarefas. Os proprietários dos terrenos onde os serviços são realizados recebem uma compensação, caso deem autorização para a execução das ações. Posteriormente, estas ações traduzem-se em créditos de carbono que ficam na posse da CO<sub>2</sub>Offset, que procederá à venda destes créditos.

Todo o percurso de cada crédito de carbono, desde a sua criação, é certificado e imortalizado utilizando tecnologias DLT (Distributed Ledger Technology), garantindo transparência, um aspeto crucial neste mercado. [20]

## 1.3 Estrutura do relatório

O presente relatório de estágio está organizado em cinco capítulos principais, cada um abordando aspectos cruciais do trabalho desenvolvido na CO<sub>2</sub>Offset e das tecnologias envolvidas. A estrutura foi concebida para proporcionar uma compreensão abrangente do projeto, desde as metodologias e ferramentas utilizadas até às conclusões e reflexões finais.

- **Metodologia de Trabalho e Ferramentas**

O primeiro capítulo debruça-se sobre a metodologia de trabalho adotada durante o estágio na CO<sub>2</sub>Offset. Aqui, serão detalhadas as diversas ferramentas empregues, explicando a sua finalidade e como foram integradas no ecossistema de trabalho da empresa. Esta secção visa fornecer uma visão clara do ambiente de desenvolvimento e dos processos utilizados ao longo do projeto.

- **Conceitos Fundamentais**

O segundo capítulo é dedicado à exploração dos conceitos fundamentais que sustentam o projeto. Inicia-se com uma explicação pormenorizada do funcionamento das tecnologias de *Distributed Ledger Technology* (DLT), justificando a sua utilização para alcançar a transparência necessária no contexto do projeto.

Em seguida, será apresentada uma explicação detalhada sobre o funcionamento da *Hashgraph*. Esta secção abordará os princípios básicos desta tecnologia, incluindo o seu mecanismo de consenso e como ela proporciona maior velocidade e eficiência energética.

Posteriormente, será realizada uma análise comparativa sucinta de diferentes tecnologias DLT, como Bitcoin e Ethereum, em relação à Hashgraph. Esta comparação culminará na justificação do uso da Hashgraph para o projeto, destacando as suas vantagens no contexto específico da CO<sub>2</sub>Offset.

Adicionalmente, este capítulo abordará o conceito de créditos de carbono e a importância crucial da transparência neste mercado, estabelecendo a ligação entre a tecnologia escolhida e a sua aplicação prática no projeto.

Por fim, será apresentada uma explicação sobre o funcionamento do Vite + React-TS, evidenciando como esta combinação melhora a experiência de desenvolvimento neste tipo de projeto e como alcança a performance prometida.

- **Planeamento do Projeto**

O terceiro capítulo foca-se no planeamento detalhado do projeto. Aborda os objetivos específicos, incluindo o desenvolvimento da aplicação web e a apresentação dos dados de créditos de carbono. Detalha as restrições tecnológicas e a liberdade criativa permitida. Por fim, apresenta uma análise dos requisitos funcionais e não funcionais.

- **Processo de Desenvolvimento do Website**

O quarto capítulo é dedicado à descrição detalhada do processo de desenvolvimento do website. Começará por abordar a fase de design, incluindo a criação de protótipos no Figma. Em seguida, será explicado o processo de transformação desses designs em código funcional.

Esta secção também incluirá uma apresentação do resultado final das páginas, permitindo uma visão clara do produto desenvolvido.

- **Conclusões e Apreciações**

O capítulo final do relatório é reservado para as conclusões e apreciações do trabalho realizado. Aqui, serão apresentadas reflexões sobre a experiência do estágio e sobre o processo de elaboração do relatório. Serão destacadas as principais aprendizagens adquiridas durante este período, tanto a nível técnico como profissional.

Adicionalmente, este capítulo incluirá o feedback recebido por parte da CO<sub>2</sub>Offset. Serão também partilhadas reflexões pessoais sobre o projeto, o seu impacto e possíveis desenvolvimentos futuros.

Esta estrutura foi desenhada para proporcionar uma narrativa coerente e abrangente do trabalho desenvolvido durante o estágio, desde os fundamentos teóricos até à implementação prática e às reflexões finais. Cada capítulo foi pensado para complementar os anteriores, oferecendo uma visão holística do projeto e da experiência de estágio na CO<sub>2</sub>Offset.



## **2 METODOLOGIA DE TRABALHO**

A metodologia de trabalho é um dos pilares fundamentais para o sucesso de qualquer projeto, especialmente na área de desenvolvimento web. A adoção de uma abordagem estruturada e eficiente não só otimiza o processo de criação, como também assegura a qualidade e a sustentabilidade do produto final. No contexto da criação de um website, uma boa metodologia de trabalho permite que a equipa de desenvolvimento colabore de forma eficaz, minimize erros e responda rapidamente a mudanças e imprevistos.

Para este projeto, utilizámos um conjunto de ferramentas que se complementam e garantem um fluxo de trabalho coeso e eficiente. Estas ferramentas são:

- **YouTrack:** Ferramenta de gestão de projetos que permite organizar, acompanhar e gerir tarefas de forma eficiente, facilitando a colaboração e a monitorização do progresso.
- **Figma:** Utilizado para a criação do design inicial do website, permitindo a obtenção de feedback e aprovação antes de prosseguir para o desenvolvimento.
- **Docker:** Utilizado para criar, testar e implantar aplicações de forma rápida e consistente, garantindo que todos os ambientes de desenvolvimento, teste e produção são idênticos.
- **GitHub:** Plataforma de gestão de código fonte que suportou o nosso modelo de branching, permitindo um desenvolvimento controlado e estruturado através das fases de development, staging e production.
- **Slack:** Rede social empresarial que facilitou a comunicação dentro da equipa, permitindo conversas em tempo real e partilha de ficheiros.
- **Postman:** Aplicação essencial para organizar e realizar pedidos a APIs de diversos métodos, sendo fundamental para testar endpoints durante o desenvolvimento.

O YouTrack organiza e monitoriza tarefas, garantindo que os objetivos do projeto sejam cumpridos. O Figma permitiu criar e melhorar o design do website com base em feedback recebido. O Docker proporcionou consistência e portabilidade ao ambiente de desenvolvimento, enquanto o GitHub facilitou a gestão do código, assegurando a integridade nas diferentes fases de desenvolvimento. O Slack promoveu comunicação constante e fluida entre os membros da equipe, promovendo colaboração eficaz. O Postman, por sua vez, foi crucial para testar e validar as comunicações com as APIs, garantindo que todas as integrações funcionem corretamente.

A seguir, detalharei como cada uma dessas ferramentas contribuiu para o sucesso deste e de qualquer projeto.

## **2.1 YouTrack**

YouTrack é uma ferramenta de gestão de projetos que permite organizar, acompanhar e gerir tarefas de forma eficiente. No contexto deste projeto, a utilização do YouTrack foi essencial para garantir a organização e o acompanhamento contínuo de todas as atividades necessárias para o desenvolvimento do website. Abaixo, exploramos as funcionalidades do YouTrack que foram fundamentais para a realização deste projeto:

- **Planeamento e Organização de Tarefas**

**Issues:** As "Issues" no YouTrack representam as tarefas a serem realizadas. Elas podem ser criadas por qualquer membro do projeto e são utilizadas para descrever atividades específicas, como a resolução de um bug ou o desenvolvimento de uma nova funcionalidade.

**Prioridades e Prazos:** Cada issue pode ser categorizada e priorizada, permitindo que a equipa saiba quais tarefas são mais urgentes e quais podem ser agendadas para um momento posterior.

**Atribuição de Tarefas:** As tarefas podem ser atribuídas a membros específicos da equipa, garantindo que todos saibam as suas responsabilidades e que não haja sobreposição de esforços.

- **Acompanhamento de Progresso**

**Estados das Tarefas:** As tarefas podem passar por vários estados, facilitando o acompanhamento do progresso:

1. **Aberto:** Quando uma tarefa é criada e ainda não foi iniciada.
2. **Em Andamento:** Quando um membro da equipa inicia a tarefa, alterando o seu estado para evitar conflitos e duplicação de esforços.
3. **A Ser Discutido:** Após a conclusão de uma tarefa, ela é marcada como "A Ser Discutido", indicando que aguarda aprovação ou verificação. Isto assegura que o trabalho realizado é revisto para identificar possíveis problemas ou melhorias.
4. **Fechado:** Uma vez aprovada, a tarefa é marcada como "Fechado", indicando que foi concluída com sucesso.

- **Colaboração e Comunicação**

**Comentários e Feedback:** Cada tarefa permite a adição de comentários, facilitando a comunicação entre os membros da equipa. Isto é útil para discutir detalhes, fornecer feedback ou esclarecer dúvidas diretamente na tarefa.

**Notificações:** Os membros da equipa recebem notificações sobre atualizações em tarefas relevantes, mantendo todos informados sobre o progresso e mudanças.

- **Relatórios e Análise**

**Tempo de Trabalho:** Cada tarefa é marcada com o tempo que demorou para ser concluída. Esta funcionalidade é crucial para a gestão do tempo, permitindo a análise do esforço despendido em diferentes atividades e ajudando a melhorar a estimativa de prazos para futuras tarefas.

**Relatórios:** YouTrack permite a geração de relatórios detalhados sobre o progresso do projeto, estado das tarefas e tempo gasto em cada atividade. Estes relatórios podem ser partilhados com toda a equipa, proporcionando uma visão clara do andamento do projeto e facilitando a tomada de decisões informadas.

- **Gestão de Recursos**

**Capacidade da Equipa:** Ao rastrear o tempo gasto em cada tarefa, YouTrack ajuda a monitorizar a capacidade da equipa, garantindo que a carga de trabalho está distribuída de forma equilibrada e que os recursos são utilizados de maneira eficiente.

**Identificação de Bloqueios:** Através do acompanhamento do progresso e do estado das tarefas, é possível identificar rapidamente bloqueios ou atrasos, permitindo que a equipa tome medidas proativas para resolvê-los.

## 2.2 Figma

Figma é uma ferramenta de design colaborativa que permite criar protótipos de alta fidelidade de websites e aplicações. No contexto deste projeto, o Figma foi utilizado para:

**Desenvolvimento do Design:** Criar o *layout* inicial do website, incluindo a estrutura das páginas, elementos visuais e interação do utilizador.

**Colaboração:** Permitir que todos os membros da equipa e partes interessadas visualizem e comentem o design em tempo real, facilitando a comunicação e a incorporação de feedback.

**Iteração e Aprovação:** Ajustar o design com base no feedback recebido e obter aprovação final antes de proceder ao desenvolvimento, garantindo que o resultado final alinha-se com as expectativas dos stakeholders.

Uma das funcionalidades mais poderosas do Figma é a capacidade de simular o funcionamento de um website. Graças a recursos como:

- **Mudança de Página:** Permite criar links entre diferentes páginas do protótipo, simulando a navegação do utilizador através do website.
- **Cores e Estilos:** Facilita a aplicação de paletas de cores e estilos consistentes em todos os elementos do design.
- **Componentes interativos:** Permite simular cliques ou outras ações de utilizadores.

Além disso, durante o processo de design, foi utilizado o conceito de componentes reutilizáveis. Estes componentes são elementos de design padronizados que podem ser reutilizados em diferentes partes do projeto, assegurando a consistência visual e funcional em todo o website. Qualquer alteração feita num componente é automaticamente refletida em todas as suas instâncias, facilitando a manutenção e atualização do design.

A utilização do Figma assegurou que o design do website foi cuidadosamente planeado e validado antes de qualquer código ser escrito, minimizando correções posteriores e melhorando a qualidade do produto final. Os componentes reutilizáveis e as simulações interativas contribuíram significativamente para a consistência do design e para a eficiência do processo de desenvolvimento.

No **Anexo A** encontra-se um esquema com todo o design realizado neste projeto, onde é possível interagir com o mesmo.



## 2.3 Docker

Docker é uma plataforma de containerização que permite criar, testar e implantar aplicações de forma rápida e consistente. No contexto do desenvolvimento de websites, o Docker proporciona um ambiente isolado e replicável, o que é fundamental para garantir a uniformidade e eficiência em todas as fases do projeto.

### Funcionamento do Docker

Docker utiliza containers para executar software de forma isolada. Estes containers são ambientes de execução leves que contêm todas as dependências necessárias para executar uma aplicação, incluindo bibliotecas, código e configurações do sistema. Isso garante que a aplicação funcione da mesma maneira em qualquer ambiente, seja no computador do desenvolvedor, num servidor de testes ou num ambiente de produção.

### Principais componentes do Docker:

1. **Docker Engine:** O motor que permite a criação e execução dos containers.
2. **Dockerfile:** Um script que contém as instruções para criar um container. Define o ambiente e as dependências necessárias para a aplicação.
3. **Docker Image:** Uma imagem é uma versão estática do container que pode ser executada. As imagens são criadas a partir do Dockerfile.
4. **Docker Container:** Uma instância em execução de uma imagem. É um ambiente isolado onde a aplicação é executada.

### Importância do Docker para o Projeto

1. **Consistência:** Docker assegura que a aplicação funciona de maneira idêntica em diferentes ambientes. Isso elimina problemas de compatibilidade e "funciona na minha máquina", pois todos os desenvolvedores e servidores utilizam o mesmo ambiente containerizado.
2. **Isolamento:** Cada container é isolado do sistema host e de outros containers, o que significa que problemas em um container não afetarão outros containers ou o sistema host. Isso aumenta a segurança e a estabilidade do desenvolvimento.
3. **Eficiência de Recursos:** Containers são leves e utilizam menos recursos do que máquinas virtuais. Eles compartilham o kernel do sistema operativo do host, o que reduz a sobrecarga de desempenho.
4. **Portabilidade:** Uma vez criada, uma imagem Docker pode ser executada em qualquer lugar que suporte Docker. Isso facilita a migração entre diferentes

ambientes, como do desenvolvimento para o teste e, finalmente, para a produção.

5. **Automatização e CI/CD:** Docker integra-se facilmente com ferramentas de integração contínua (CI) e entrega contínua (CD). É possível automatizar a construção e teste de containers, o que acelera o ciclo de desenvolvimento e aumenta a eficiência.
6. **Escalabilidade:** Docker facilita a escalabilidade horizontal, permitindo a execução de múltiplas instâncias de um container para lidar com aumentos na carga de trabalho. Isso é essencial para aplicações que necessitam de alta disponibilidade e performance.

### **Aplicação do Docker no Projeto**

No desenvolvimento deste website, o Docker foi utilizado para criar ambientes de desenvolvimento, teste e produção consistentes. A decisão de usar Docker foi fortemente influenciada pelo fato de que o servidor de produção está hospedado na cloud da AWS e utiliza um sistema operativo Linux. Aqui estão alguns detalhes de como o Docker foi aplicado:

1. **Criação do Dockerfile:** Um Dockerfile foi escrito para definir todas as dependências e configurações necessárias para executar a aplicação web. Isso incluiu a especificação da base de imagem, instalação de bibliotecas e configuração do ambiente, de forma a compatibilizar com o ambiente Linux da AWS.
2. **Construção de Imagens:** A partir do Dockerfile, foram criadas imagens Docker que encapsulam a aplicação e todas as suas dependências. Estas imagens foram armazenadas num repositório de imagens, permitindo que qualquer membro da equipa as utilizasse para criar containers. Isso garantiu que o ambiente de desenvolvimento local fosse consistente com o ambiente de produção Linux na AWS.
3. **Execução de Containers:** Containers foram executados a partir das imagens Docker em ambientes de desenvolvimento e teste. Isso assegurou que a aplicação era executada em condições idênticas às que seriam encontradas no ambiente de produção Linux, minimizando problemas de compatibilidade.
4. **Testes Automatizados:** Utilizando ferramentas de CI/CD, os containers Docker foram automaticamente construídos e testados em cada alteração do código. Isso garantiu que problemas eram identificados e resolvidos rapidamente, mantendo a qualidade do código e a compatibilidade com o ambiente Linux da AWS.
5. **Desempenho e Escalabilidade:** Em produção, Docker permitiu a execução de múltiplas instâncias da aplicação para lidar com aumentos na carga de

trabalho, assegurando alta disponibilidade e desempenho. A integração com a AWS facilitou a escalabilidade horizontal e o gerenciamento eficiente dos recursos.

A utilização do Docker foi essencial para garantir a consistência, eficiência e portabilidade no desenvolvimento deste website. Docker simplificou a gestão de ambientes, eliminou problemas de compatibilidade e acelerou o ciclo de desenvolvimento através da automação e integração contínua. A capacidade de escalar a aplicação de forma eficiente e a compatibilidade com o ambiente Linux na AWS tornaram possível responder a aumentos de carga e assegurar um desempenho estável. Em resumo, Docker foi uma ferramenta fundamental para o sucesso deste projeto.

## **2.4 GitHub e Branching**

O GitHub é uma plataforma baseada em nuvem para hospedagem e colaboração em projetos de desenvolvimento de software que utilizam o sistema de controlo de versão Git. Ele facilita a gestão de código-fonte, permitindo que desenvolvedores colaborem de forma eficiente, façam commits, criem branches e gerenciem versões de forma organizada. No contexto deste projeto, o código-fonte está alojado num repositório GitHub, permitindo que os colaboradores contribuam com alterações e integrem as suas modificações de forma controlada e segura.

### **2.4.1 GitHub**

O GitHub é uma plataforma amplamente utilizada que oferece uma série de funcionalidades para facilitar o desenvolvimento colaborativo. Entre as principais características do GitHub estão:

- **Repositórios:** Espaços onde o código-fonte e o histórico de versões do projeto são armazenados. Cada repositório pode ser acedido por múltiplos colaboradores.
- **Commits:** Registos de alterações feitas no código-fonte. Cada commit inclui uma mensagem descritiva sobre as alterações realizadas.
- **Branches:** Permitem a criação de ramificações independentes no projeto, facilitando o desenvolvimento paralelo de novas funcionalidades e correções.
- **Pull Requests:** Solicitações para integrar alterações de uma branch para outra. Permitem a revisão de código e discussão antes da fusão.
- **Issues:** Ferramentas para rastrear tarefas, bugs e melhorias, possibilitando a gestão e priorização de trabalho.

O repositório deste projeto no GitHub serve como o centro de controlo para todas as alterações de código. Os colaboradores podem fazer commits para as suas próprias branches, criar pull requests para revisão e integração de alterações, e usar issues para rastrear problemas e melhorias.

## **2.4.2 Modelo de Branching**

O modelo de branching no GitHub é uma prática essencial para a gestão eficiente de projetos de desenvolvimento de software. Adotamos o modelo de branching descrito no artigo "A successful Git branching model" por Vincent Driessen [13], com algumas adaptações específicas para o nosso processo de desenvolvimento. Este modelo ajuda a organizar o fluxo de trabalho, facilitando o desenvolvimento paralelo e a integração de novas funcionalidades e correções.

### **Branches Principais**

No nosso fluxo de trabalho, utilizamos três branches principais:

1. **Development:** Esta branch contém o código mais recente em desenvolvimento ativo. Todas as novas funcionalidades e melhorias são inicialmente integradas aqui. É um ambiente colaborativo onde os desenvolvedores testam e ajustam novas alterações antes de serem consideradas estáveis.
2. **Staging:** Esta branch serve como um ambiente de pré-produção. As alterações que são consideradas estáveis na branch development são promovidas para staging, onde passam por testes mais rigorosos e preparativos finais para produção. Funciona como um passo intermédio para garantir que o código esteja pronto para ser lançado.
3. **Production:** Esta branch contém o código que está atualmente em produção. Apenas alterações amplamente testadas e aprovadas são integradas aqui. É a versão do código que os clientes finais utilizam.

### **Branches de Suporte**

Além das branches principais, utilizamos várias branches de suporte para facilitar o desenvolvimento paralelo e a gestão de alterações:

1. **Feature Branches:** Usadas para desenvolver novas funcionalidades. Estas branches são criadas a partir da branch development e, uma vez concluídas, são integradas de volta a development.

**\$ git checkout -b feature/feature1 development**

Este comando cria uma branch denominada feature/feature1 a partir da branch development.

2. **Fix Branches:** Utilizadas para corrigir bugs. Podem ser criadas a partir de qualquer branch principal (development, staging ou production), dependendo de onde o bug foi encontrado. Estas branches são integradas de volta à branch de origem após a correção do bug.

```
$ git checkout -b fix/bug1 production
```

Este comando cria uma branch fix/bug1 a partir da branch production. Também é possível criar fix branches para development ou staging, conforme necessário.

## **Integrar uma Feature Branch**

Quando uma nova funcionalidade estiver pronta para ser integrada na branch de desenvolvimento (development), seguimos estes passos:

1. **Mudar para a branch development:**

```
$ git checkout development
```

Este comando muda o contexto do repositório para a branch development.

2. **Fazer o merge da feature branch, preservando o histórico da funcionalidade:**

```
$ git merge --no-ff feature/feature1
```

Este comando integra a branch feature/feature1 na branch development, mantendo o histórico da funcionalidade.

3. **Opcionalmente, apagar a feature branch, uma vez integrada:**

```
$ git branch -d feature/feature1
```

Este comando apaga a branch feature/feature1.

4. **Enviar as alterações para o repositório remoto:**

```
$ git push origin development
```

Este comando atualiza a branch development no repositório remoto.

## **Promover uma Versão**

Quando a versão na branch development está pronta para ser promovida para staging ou a versão staging está pronta para production, seguimos um processo semelhante de merge. Por exemplo, para promover da branch development para staging:

1. **Mudar para a branch staging:**

```
$ git checkout staging
```

2. Fazer o merge da branch development na branch staging:

```
$ git merge --no-ff development
```

3. Enviar as alterações para o repositório remoto:

```
$ git push origin staging
```

Este modelo de branching garante que o desenvolvimento e a integração das funcionalidades sejam realizados de forma organizada e controlada, minimizando o risco de conflitos e problemas na produção.

## 2.5 Postman

Durante o desenvolvimento da aplicação web, foi necessário interagir com três APIs diferentes, cada uma com suas características específicas:

1. **Directus API:** API utilizada pela CO<sub>2</sub>Offset para gestão de dados e conteúdos. O Directus é um sistema de gestão de conteúdos (CMS) que oferece uma API robusta e flexível, permitindo a manipulação e consulta de dados armazenados de forma estruturada.
2. **API desenvolvida por uma empresa externa:** Esta API foi concebida para a CO<sub>2</sub>Offset de modo a facilitar a escrita de dados na Hedera Hashgraph, requer um certificado e chaves autorizados para acesso e suporta apenas pedidos POST e PUT. A segurança e a autenticidade dos pedidos são cruciais para a interação com esta API.
3. **Testnet Mirror Node REST API da Hedera Hashgraph:** API utilizada para interagir com a rede de ledger distribuído da Hedera Hashgraph, oferecendo endpoints para consultas de dados armazenados na rede de forma eficiente e segura.

Com APIs tão diversas, cada uma com seus próprios métodos de autenticação, tipos de pedidos e estruturas de dados, a utilização de uma ferramenta como o Postman foi fundamental. O Postman permitiu-nos interagir com todas essas APIs de maneira consistente e eficiente, facilitando o desenvolvimento e a integração no nosso projeto. A seguir, detalharemos como o Postman contribuiu para o sucesso deste processo:

- **Teste de Pedidos às APIs:** Utilizamos o Postman para enviar pedidos a todas as três APIs, verificando a autenticidade e funcionalidade dos endpoints. Esta capacidade de teste foi essencial para garantir que os pedidos

fossem corretamente formatados e autenticados, especialmente para a API externa que requer certificados e chaves autorizadas.

- **Visualização e Estruturação das Respostas:** O Postman permitiu-nos visualizar as respostas das APIs de forma clara e estruturada, independentemente das diferenças entre elas. Esta visualização facilitou o planeamento de como os dados seriam processados e integrados na aplicação web, garantindo uma integração fluida e alinhada com os requisitos do projeto.
- **Depuração e Ajuste de Pedidos:** Durante o desenvolvimento, o Postman foi uma ferramenta vital para identificar e corrigir problemas nos pedidos, como parâmetros incorretos ou erros de autenticação. Esta capacidade de depuração assegurou que a comunicação entre a nossa aplicação e as APIs fosse correta e eficiente.
- **Scripts Pre-request e Post-response:** O Postman permite a utilização de scripts pre-request e post-response, o que nos ajudou a manipular e tratar os dados recebidos nos pedidos. Esta funcionalidade foi especialmente útil para preparar os pedidos antes de serem enviados e para processar as respostas das APIs, garantindo que os dados fossem apresentados e utilizados conforme necessário no desenvolvimento da aplicação.

A integração do Postman no nosso projeto foi essencial para garantir uma interação eficaz com as APIs da Directus, da empresa externa e da Hedera Hashgraph. Através desta ferramenta, conseguimos testar, visualizar e planificar a implementação dos dados de forma estruturada e organizada. O Postman não só facilitou o desenvolvimento como também assegurou a qualidade e a fiabilidade da nossa aplicação web, demonstrando ser uma ferramenta indispensável no nosso fluxo de trabalho.





### 3 CONCEITOS

No desenvolvimento e análise de tecnologias emergentes, é fundamental compreender os conceitos que as sustentam. Neste capítulo, serão exploradas várias tecnologias e terminologias que desempenham um papel crucial no ecossistema das tecnologias distribuídas, bem como o conceito de créditos de carbono e a importância da transparência nesse mercado. Este capítulo tem como objetivo proporcionar uma base sólida de conhecimento, essencial para a compreensão mais aprofundada dos tópicos subsequentes discutidos no relatório.

Serão abordadas as DLTs (*Distributed Ledger Technologies*), que permitem a partilha de dados de forma segura e transparente entre múltiplas partes. A partir deste conceito geral, avançaremos para tecnologias específicas que implementam DLTs, como a *Blockchain* e a *Hashgraph*, explorando as suas características únicas, funcionamento e vantagens.

Além disso, serão discutidas as diferenças entre estas tecnologias, destacando as suas aplicações e casos de uso específicos. No contexto das implementações de *blockchain*, serão analisadas as características do *Bitcoin* e do *Ethereum*, duas das mais conhecidas e amplamente utilizadas plataformas de blockchain, explicando como cada uma utiliza esta tecnologia. A comparação será feita com a *Hedera Hashgraph*, examinando as vantagens e desvantagens de cada uma destas tecnologias em termos de desempenho, segurança, escalabilidade e eficiência. Esta análise permitirá perceber os motivos que levaram à escolha da *Hedera Hashgraph* para este projeto.

Também será explicado o que é um crédito de carbono, a sua importância no combate às alterações climáticas e a necessidade de transparência no mercado de créditos de carbono. Entender a interligação entre estas tecnologias emergentes e os créditos de carbono é crucial para perceber como as DLTs podem contribuir para uma maior transparência e eficiência neste mercado vital para a sustentabilidade ambiental.

### **3.1 Distributed Ledger Technologies (DLTs)**

As Tecnologias de Registo Distribuído (DLTs) representam uma inovação disruptiva na gestão de transações digitais, introduzindo um modelo descentralizado que elimina a necessidade de uma autoridade central para validar e registar transações. No cerne das DLTs está o conceito de um registo digital distribuído entre os participantes da rede, onde cada um mantém uma cópia idêntica e sincronizada do registo.

Embora uma DLT, elimina a necessidade de uma autoridade central, esse nem sempre é o caso, para se atingir uma descentralização completa é necessário uma rede Peer-to-Peer (P2P) que é uma infraestrutura subjacente das DLTs, onde os nós (dispositivos) participantes interagem diretamente uns com os outros sem a necessidade de um servidor centralizado. Cada nó na rede P2P é tanto um cliente quanto um servidor, capaz de enviar e receber dados para e de outros nós. Isso permite a descentralização completa das DLTs, pois não há um ponto único de controlo ou falha na rede. A comunicação entre os nós é geralmente facilitada por protocolos específicos, como o TCP/IP, que garantem a transferência confiável de dados através da internet.

Os nós numa rede P2P têm várias funções importantes:

- **Armazenamento de Dados:** Cada nó mantém uma cópia completa ou parcial do registo digital, assegurando que os dados sejam distribuídos e redundantes.
- **Validação de Transações:** Os nós verificam a validade das transações que recebem antes de as retransmitirem para outros nós.
- **Propagação de Informação:** Os nós propagam transações e blocos novos por toda a rede, assegurando que todos os participantes estejam sincronizados.

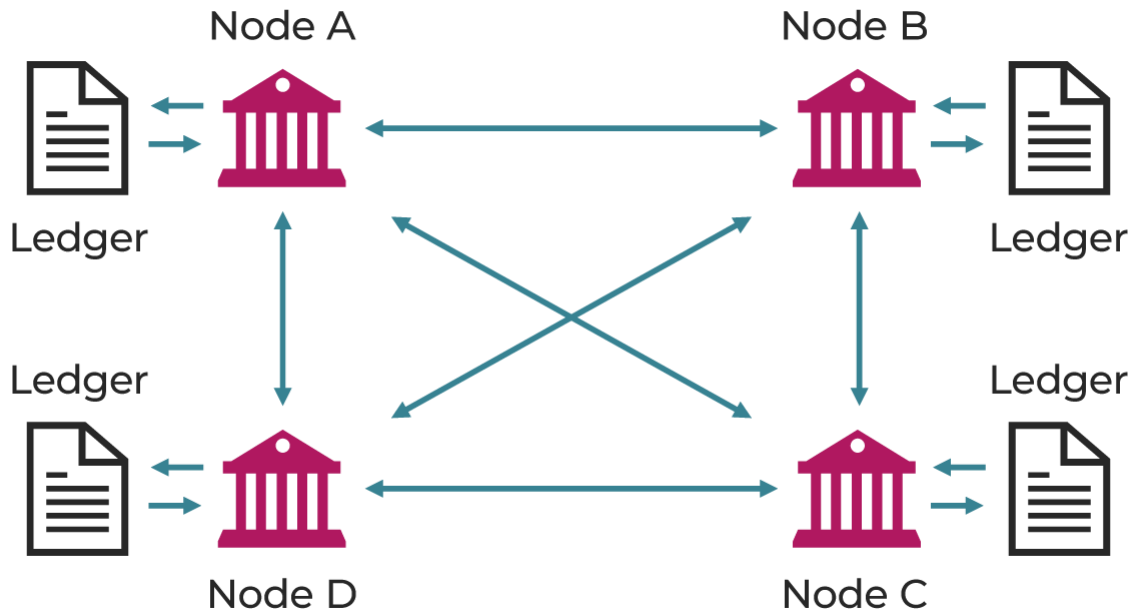


Figura 1 – Esquema de uma DLT descentralizada P2P

Numa rede descentralizada, a presença de nós com intenções maliciosas pode comprometer a segurança e a integridade do registo digital ao tentarem manipular transações, alterar blocos existentes ou criar ramificações na cadeia de blocos. Esta ameaça pode prejudicar a confiança no sistema e potencialmente corromper dados essenciais.

Para enfrentar este desafio, foram desenvolvidos os algoritmos de consenso, essenciais para o funcionamento das DLTs. Estes algoritmos permitem que os nós na rede cheguem a um consenso sobre o estado atual do registo, garantindo que todos concordem com a versão correta, apesar das possíveis discrepâncias causadas por atrasos na disseminação de informações ou tentativas maliciosas de manipulação. Exemplos de algoritmos de consenso incluem a Prova de Trabalho (*Proof of Work* - PoW), Prova de Participação (*Proof of Stake* - PoS), entre outros, cada um projetado para assegurar a integridade e a segurança do sistema descentralizado.

Para uma compreensão aprofundada do potencial e das aplicações das DLTs, os próximos capítulos dedicar-se-ão à exploração detalhada de dois tipos distintos: *blockchain* e *Hashgraph*. [9]

### 3.1.1 Blockchain

Um exemplo de uma DLT é a *Blockchain*, inicialmente concebida como a espinha dorsal das criptomoedas, evoluiu significativamente ao longo dos anos, encontrando aplicação em várias áreas além das finanças. Este capítulo apresenta uma visão abrangente da tecnologia Blockchain, abordando os seus conceitos fundamentais e detalhes técnicos, sem focar em blockchains específicas. Nos capítulos seguintes, discutiremos em detalhe as duas blockchains mais conhecidas: Bitcoin e Ethereum.

A *Blockchain* é, em essência, uma DLT imutável que permite o registo de transações de forma segura, transparente e eficiente. É composto por uma cadeia de blocos, onde cada bloco contém um conjunto de transações. Cada novo bloco é ligado ao anterior através de um processo criptográfico, formando uma cadeia contínua – daí o nome "*blockchain*". A estrutura de uma blockchain é composta por uma série de blocos ligados entre si, formando uma cadeia. Esta estrutura é essencial para garantir a integridade e segurança dos dados armazenados. Cada bloco na cadeia tem uma ligação criptográfica ao bloco anterior, criando uma sequência linear de blocos desde o bloco génese (o primeiro bloco da cadeia) até ao bloco mais recente. A estrutura de uma blockchain inclui os seguintes componentes:

- **Blocos:** Unidades básicas que compõem a blockchain. Cada bloco contém um conjunto de transações e um cabeçalho que inclui informações essenciais sobre o bloco.
- **Nó (Node):** Dispositivo que participa na rede blockchain, armazenando uma cópia completa ou parcial da blockchain e participando no processo de validação e transmissão de transações.
- **Rede P2P (Peer-to-Peer):** Estrutura descentralizada onde todos os nós da rede comunicam directamente uns com os outros, sem necessidade de uma autoridade central.
- **Mecanismo de Consenso:** Método pelo qual a rede blockchain concorda sobre o estado actual da blockchain e valida novos blocos a serem adicionados.

#### Estrutura de um Bloco

Um bloco na Blockchain contém vários elementos essenciais:

- **Cabeçalho do Bloco (Block Header):** Inclui metadados importantes, como o hash do bloco anterior, o hash do bloco actual, o carimbo de tempo (timestamp), e outros dados.

- **Hash do Bloco Anterior:** Este campo garante a ligação entre os blocos, criando a cadeia. Qualquer alteração num bloco anterior resultaria numa alteração no seu hash, tornando todos os blocos subsequentes inválidos.
- **Merkle Tree Root:** É um hash que representa todas as transações no bloco. A Merkle Tree (ou árvore de Merkle) permite verificar rapidamente a presença de uma transação no bloco.
- **Carimbo de Tempo (Timestamp):** Regista quando o bloco foi criado.
- **Nonce:** Um valor numérico utilizado no processo de mineração para encontrar um hash que atenda a um critério específico.
- **Lista de Transações:** Contém todas as transações validadas e incluídas no bloco.

## Processo de Mineração

A mineração é o processo pelo qual novos blocos são adicionados à *blockchain*. Este processo envolve a validação de transações e a criação de novos blocos que são adicionados à cadeia. A mineração serve dois propósitos principais:

1. **Validação de Transações:** Assegura que as transações incluídas no bloco são válidas.
2. **Segurança da Rede:** Dificulta a alteração dos dados já registados na *blockchain*.

## Consenso Distribuído

Para que a *blockchain* funcione de forma eficaz, é necessário um mecanismo de consenso distribuído, garantindo que todos os nós da rede concordem com o estado actual da *blockchain*. Independentemente do mecanismo específico utilizado, o objectivo principal é assegurar que todos os participantes da rede estejam de acordo com a versão actual da *blockchain*.

## Segurança e Imutabilidade

A segurança da *blockchain* é garantida por várias camadas de protecção:

- **Criptografia:** A utilização de algoritmos criptográficos avançados para proteger os dados e as transações.
- **Descentralização:** A ausência de uma autoridade central dificulta ataques coordenados.

- **Imutabilidade:** Uma vez que um bloco é adicionado à blockchain, alterar as informações contidas nele exigiria a re-mineração de todos os blocos subsequentes, o que é computacionalmente inviável.

## Aplicações da Blockchain

Embora originalmente concebida para criptomoedas, a tecnologia blockchain tem aplicações em diversas áreas, incluindo:

- **Contratos Inteligentes (Smart Contracts):** Programas autoexecutáveis que correm na blockchain.
- **Registos Descentralizados:** Gestão de activos, registos de propriedade e cadeias de abastecimento.
- **Votação Electrónica:** Sistemas de votação mais seguros e transparentes.
- **Identidade Digital:** Gestão de identidades digitais de forma descentralizada.

Concluindo, a tecnologia *blockchain* representa uma revolução na forma como armazenamos, gerimos e transferimos informações. Com a sua capacidade de fornecer um registo imutável e distribuído de transações, a *blockchain* tem o potencial de transformar diversas indústrias.

Nos capítulos seguintes, exploraremos em detalhe as duas *blockchains* mais conhecidas: *Bitcoin* e *Ethereum*, analisando as suas especificidades, vantagens e desvantagens, bem como as suas aplicações práticas no mundo real.

### 3.1.1.1 Bitcoin

Bitcoin é uma criptomoeda que opera numa rede descentralizada baseada na tecnologia *blockchain*. Esta criptomoeda foi criada por uma entidade (ou grupo) sob o pseudónimo de Satoshi Nakamoto, sendo lançada em 2009 como uma resposta à necessidade de um sistema monetário independente de entidades centralizadas, como bancos e governos. Neste capítulo, vamos explorar os aspectos técnicos específicos da *Bitcoin*.

Exceto o algoritmo de consenso e processo de mineração, a rede *Bitcoin* segue a lógica fundamental da tecnologia *blockchain*, que foi abordada no capítulo anterior.

O algoritmo de consenso é responsável pela criação de novos blocos na rede, ou seja, o processo de mineração é assegurado pelo seu algoritmo de consenso, o que faz com que os nós sejam também os mineradores, este algoritmo é o *Proof Of Work*.

## Proof of Work

### 1. Objetivo do PoW:

O *Proof of Work* é um mecanismo de consenso que visa garantir a segurança e integridade da rede *Bitcoin*, evitando fraudes como a duplicação de transações (o problema do "gasto duplo") e assegurando que todos os participantes da rede concordem sobre o estado atual da *blockchain*.

### 2. Processo de Mineração:

- **Criação de Blocos:** Os mineradores agrupam transações não confirmadas num bloco candidato.
- **Hashing:** Para adicionar este bloco à blockchain, os mineradores precisam resolver um problema criptográfico. Este problema envolve encontrar um valor chamado "nonce" (número único) que, quando combinado com os dados do bloco e passado por uma função de hash (SHA-256 no caso do Bitcoin), resulta num hash que é menor que um alvo predefinido. Este alvo é ajustado periodicamente para manter o tempo médio de criação de blocos em cerca de 10 minutos.
- **Prova de Trabalho:** O hash resultante serve como a prova de que uma quantidade significativa de trabalho computacional foi realizada. Este processo de tentativa e erro requer uma quantidade considerável de poder de processamento e energia.

### 3. Competição entre Mineradores:

- Todos os mineradores na rede competem para encontrar a solução para o problema de hashing. O primeiro minerador a encontrar uma solução válida (um hash que atende aos requisitos do alvo) transmite o bloco para a rede.
- Outros nós verificam a validade do bloco e do nonce. Se a maioria da rede concorda que a solução é válida, o bloco é adicionado à blockchain, e o minerador recebe uma recompensa na forma de novos bitcoins (recompensa por bloco) e taxas de transação incluídas no bloco.

## Segurança e Descentralização

### 1. Dificuldade Ajustável:

- O nível de dificuldade do problema de hashing é ajustado aproximadamente a cada 2016 blocos (cerca de duas semanas) para garantir que novos blocos sejam adicionados a um ritmo constante, independentemente das flutuações no poder computacional total da rede. Isto impede que os blocos sejam minerados muito rapidamente ou muito lentamente.

### 2. Descentralização:

- A rede Bitcoin é descentralizada porque não há uma entidade central que controla a criação de blocos ou a validação de transações. Qualquer pessoa com o hardware e software adequados pode participar como minerador.
- Os nós da rede espalham-se globalmente, contribuindo para a resistência da rede contra censura e ataques. Cada nó mantém uma cópia completa da blockchain, garantindo a transparência e imutabilidade das transações.

### 3. Resistência a Ataques:

- Para manipular a blockchain (por exemplo, realizar um ataque de 51%, onde um atacante controla a maioria do poder de hash), um atacante precisaria controlar mais da metade do total do poder de computação da rede. Isso exigiria recursos extraordinários, tornando tal ataque impraticável e extremamente caro.
- Além disso, como o PoW requer uma quantidade significativa de energia e recursos computacionais, a segurança da rede aumenta proporcionalmente ao aumento do poder de hashing, tornando mais difícil e caro tentar fraudar o sistema.

A Bitcoin tira pleno proveito da tecnologia blockchain para criar um sistema monetário descentralizado, seguro e resistente à censura. Através do algoritmo de consenso *Proof of Work*, a rede assegura que a criação de novos blocos e a validação de transações sejam realizadas de forma distribuída, sem depender de uma autoridade central. A combinação do hashing criptográfico e da competição entre mineradores garante a integridade e segurança das transações.



A descentralização da rede *Bitcoin* é um dos seus principais trunfos, permitindo que qualquer pessoa com os recursos necessários possa participar no processo de mineração e validação de transações. Esta característica aumenta a resistência da rede contra ataques e censura, uma vez que não existe um único ponto de falha ou controlo. Em resumo, a *Bitcoin* utiliza a tecnologia blockchain para criar uma moeda digital que opera de forma independente de bancos e governos, proporcionando um meio de troca seguro, transparente e globalmente acessível.

Com outra abordagem, surgiram outras redes blockchain como o Ethereum, que expandem as capacidades da tecnologia blockchain além das transações monetárias, permitindo a criação de contratos inteligentes e aplicações descentralizadas. [4][8][18]

### 3.1.1.2 Ethereum

*Ethereum* é uma plataforma descentralizada baseada na tecnologia blockchain, introduzida por Vitalik Buterin em 2015. Ao contrário da *Bitcoin*, que se foca principalmente em ser uma moeda digital, o Ethereum expande as capacidades da *blockchain* para permitir a execução de contratos inteligentes (smart contracts) e o desenvolvimento de aplicações descentralizadas (DApps). Neste capítulo, vamos explorar os aspectos técnicos específicos do Ethereum e como ele utiliza a tecnologia blockchain.

## Contratos Inteligentes e a Máquina Virtual Ethereum (EVM)

### 1. Contratos Inteligentes:

- **Definição:** Contratos inteligentes são scripts de código que são automaticamente executados quando certas condições predefinidas são atendidas. Estes contratos são armazenados e replicados na blockchain do Ethereum, garantindo transparência e imutabilidade.
- **Funcionalidade:** Permitem automatizar processos complexos e realizar transações seguras e confiáveis sem intermediários. Por exemplo, um contrato inteligente pode ser programado para libertar fundos automaticamente quando um produto é entregue.

### 2. Máquina Virtual Ethereum (EVM):

- **Papel:** A EVM é o ambiente de execução para os contratos inteligentes no Ethereum. Cada nó na rede Ethereum executa a EVM, garantindo que os contratos inteligentes funcionem da mesma forma em todos os lugares.
- **Execução de Código:** Quando um contrato inteligente é chamado, a EVM executa o seu código, que é escrito em linguagens específicas como

Solidity. A execução é feita de maneira distribuída e simultânea por todos os nós da rede.

## Algoritmo de Consenso e Mineração

### 1. Proof of Stake (PoS):

- **Transição do PoW para o PoS:** Originalmente, o Ethereum usava o algoritmo de consenso Proof of Work (PoW), semelhante ao Bitcoin. No entanto, a 15 de setembro de 2022, o Ethereum completou a transição para Proof of Stake (PoS) através de um evento conhecido como "The Merge".
- **O que foi o Merge:** O Merge foi a junção da camada de execução original do Ethereum (Mainnet) com a sua nova camada de consenso Proof of Stake, a Beacon Chain. Este evento eliminou a necessidade de mineração intensiva em energia, permitindo que a rede fosse assegurada utilizando ETH em staking. Esta mudança aumentou a escalabilidade, segurança e sustentabilidade da rede.
- **Funcionamento do PoS:** No PoS, os nós validadores são escolhidos para criar novos blocos com base na quantidade de ETH que possuem e estão dispostos a "apostar" como garantia. Isto reduz a necessidade de poder computacional intensivo e promove a sustentabilidade da rede.

### 2. Processo de Validação:

- **Criação de Blocos:** Os nós validadores selecionados agrupam transações e criam novos blocos. Eles são incentivados a agir de forma honesta através de recompensas e penalizações.
- **Finalidade e Segurança:** A utilização do PoS melhora a resistência da rede contra ataques, pois para comprometer a rede, um atacante precisaria controlar uma grande quantidade de ETH, o que é economicamente inviável. Além disso, após a transição para o PoS, o supply de Ethereum tornou-se deflacionário, aumentando o valor a longo prazo.

## Aplicações Descentralizadas (DApps)

- **Definição:** Aplicações descentralizadas são programas que funcionam na rede Ethereum e utilizam contratos inteligentes para a sua lógica de backend.

- **Exemplos:** Existem DApps para finanças descentralizadas (DeFi), jogos, redes sociais, entre outros. Estas aplicações beneficiam da segurança e descentralização da rede Ethereum.

## **Escalabilidade e Futuro**

### **1. Desafios de Escalabilidade:**

- **Limitações Atuais:** A rede Ethereum enfrenta desafios em termos de escalabilidade, com limitações no número de transações que pode processar por segundo. Teoricamente, Ethereum pode processar até 119 transações por segundo, embora o máximo registado até agora tenha sido ligeiramente superior a 62 transações por segundo. As taxas de transação (gas fees) podem ser elevadas em períodos de alta procura, tornando a utilização da rede cara para alguns utilizadores.

### **2. Planos Futuros:**

- **Proto-Danksharding:** Ethereum tem planos de implementar o Proto-Danksharding, um tipo de sharding que permitirá à rede processar até 100 mil transações por segundo. Este desenvolvimento aumentará significativamente a capacidade da rede, reduzindo as taxas de transação e melhorando a eficiência geral.

Ethereum representa uma evolução significativa da tecnologia blockchain, expandindo as suas capacidades além das transações monetárias para incluir contratos inteligentes e aplicações descentralizadas. Com a transição para Proof of Stake e os planos para implementar Proto-Danksharding, o Ethereum está bem posicionado para continuar a liderar a inovação no espaço das criptomoedas e blockchain. A sua flexibilidade e capacidade de execução de código complexo tornam-no uma plataforma robusta para uma vasta gama de aplicações descentralizadas, transformando diversos sectores da economia digital.

**[5]**

### **3.1.2 Hashgraph**

Como alternativa à tecnologia blockchain, surgiram algumas opções inovadoras. Uma dessas alternativas é a Hashgraph, uma rede descentralizada. A Hashgraph promete oferecer maior eficiência, segurança e velocidade em comparação com as redes tradicionais de blockchain. Desenvolvida pela Hedera, esta tecnologia utiliza um novo algoritmo de consenso que permite transações rápidas e seguras, mantendo a descentralização e a resistência a ataques. Neste capítulo, vamos explorar os detalhes técnicos da Hashgraph e entender como ela se distingue da blockchain, bem como as suas capacidades e funcionalidades.

#### **Diferença entre Hashgraph e Blockchain/Bitcoin/Ethereum**

Embora a Hashgraph não seja uma blockchain, ela é uma tecnologia de registro distribuído (DLT) tal como ela.

Ao invés de usar blocos encadeados como na blockchain, a Hashgraph utiliza um grafo acíclico direcionado (DAG). Este método permite a confirmação de transações através de um algoritmo de consenso mais rápido e eficiente.

Uma grande diferença entre Hashgraph e a concorrência é a localização dos nós, em Bitcoin e Ethereum qualquer pessoa pode correr o seu próprio nó, havendo um grande número e distribuição de nós nessas redes que previne ataques maliciosos, por outro lado a Hashgraph adotou uma estratégia controversa no espaço descentralizado, tendo uma limitada seleção de nós (31 à data de escrita deste documento) escolhidos pela própria Hedera, mas apesar das críticas a este método, a Hedera promete descentralização e segurança na rede, ela atinge esse objetivo da seguinte forma:

**Segurança:** A seleção dos nós da rede pela Hedera foi feita cuidadosamente e são hospedadas por grandes empresas de confiança, entre elas a LG, Google, IBM, Ubisoft, entre outras.

**Descentralização:** A descentralização é assegurada pelo algoritmo de consenso que usa 2 técnicas, “Gossip about Gossip” e “Virtual Voting”

Para entender como este algoritmo de consenso é aplicado na prática passo a detalhar o funcionamento da rede:

## Funcionamento da rede

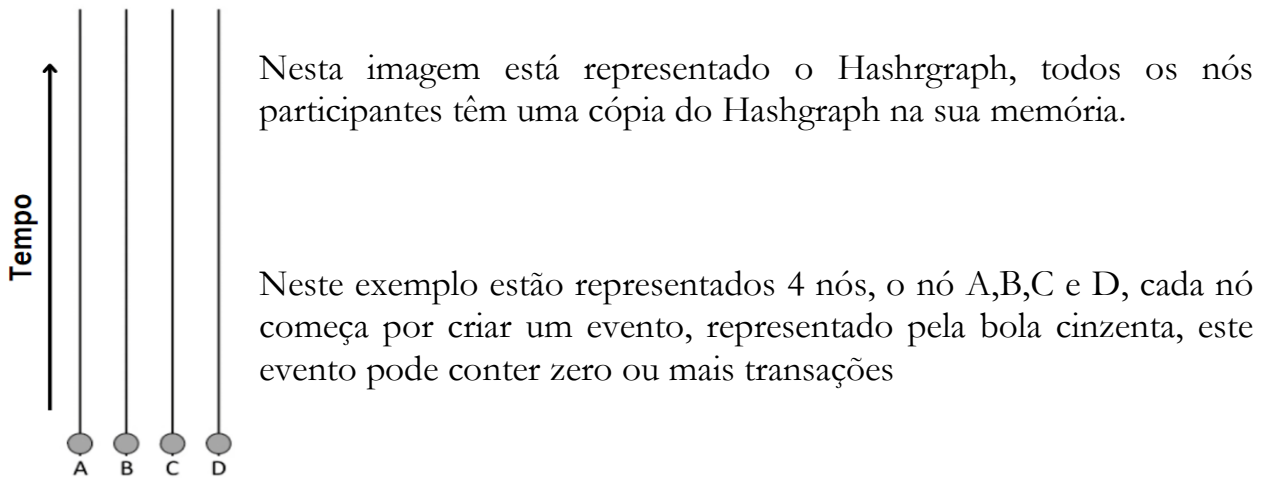


Figura 2 - Hashgraph no estado inicial

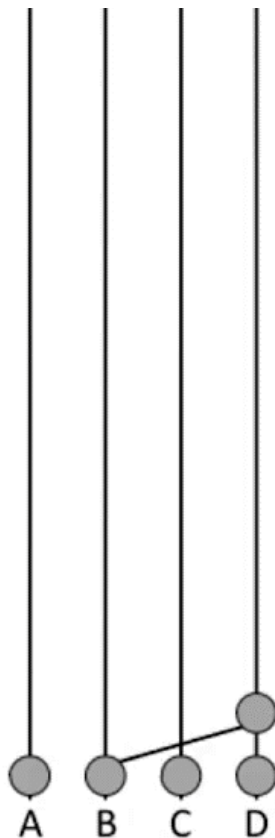


Figura 3 - Hashgraph após uma iteração

Neste momento o nó B escolheu aleatoriamente o nó D para enviar todos os eventos de que tem conhecimento que o nó D não tenha conhecimento (neste caso apenas 1, o que foi criado inicialmente pelo nó B), após receber esse evento o nó D cria um novo evento que vai ter os eventos que foram recebidos pelo nó B e os que o nó D já teria conhecimento.

Este novo evento passa a ter 2 conexões que apontam para 2 eventos, estes são chamados os “parent events”, este novo evento vai ter o hash de cada um dos seus parentes, bem como a timestamp, que indica quando foi criado o evento, e o conteúdo de todas as transações, e é assinado digitalmente pelo seu criador, que neste caso é o nó D.

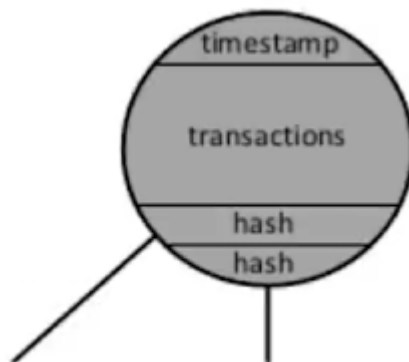
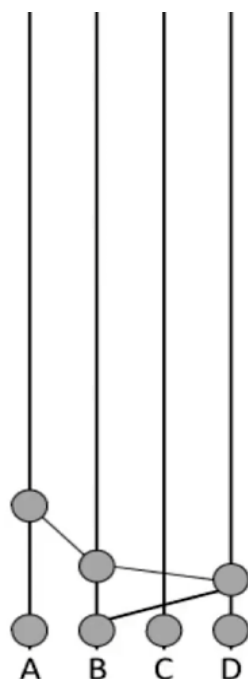


Figura 4 - Conteúdo de um evento



Avançando um pouco no tempo, podemos ver que neste momento, o nó D escolheu aleatoriamente o nó B para enviar todos os eventos que sabe e de seguida o nó B escolheu aleatoriamente o nó A para enviar todos os 4 eventos que conhece, o nó A cria então um novo evento para registar o ocorrido.

Este processo repete-se infinitamente desde que a rede esteja operacional.

Figura 5 - Hashgraph após 3 iterações

Para alcançar consenso sobre a ordem das transações é aplicado o Virtual Voting que segue 3 passos:

1. Dividir Rondas
2. Decidir a fama
3. Encontrar a ordem

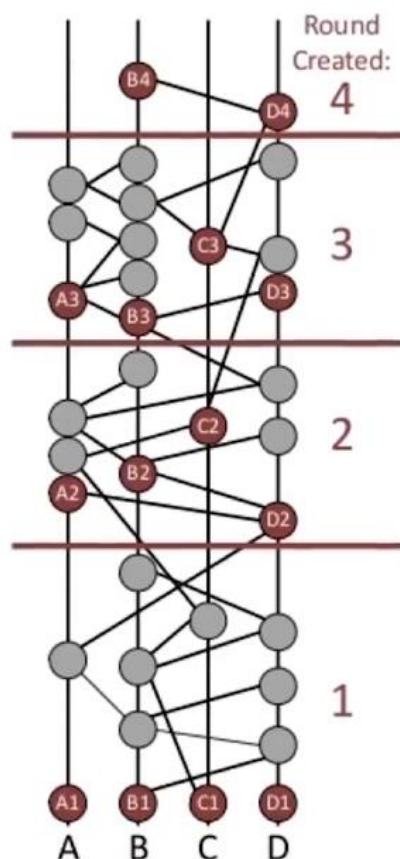


Figura 6 - Rondas e Eventos testemunhas da Hashgraph

O primeiro evento de cada nó em cada ronda vai ser considerada o evento testemunha, que na imagem estão a vermelho

Em cada ronda, os eventos testemunhas são analisados para determinar a sua "fama". A fama de um evento é uma medida do quão amplamente o evento é conhecido e aceite por outros nós.

Os nós usam o algoritmo de votação virtual para determinar a fama dos eventos testemunhas. Um evento é considerado famoso se for conhecido por mais de dois terços dos nós na rede.

Uma vez que a fama dos eventos testemunhas é determinada, a ronda é considerada finalizada. Os eventos famosos são utilizados para decidir a ordem consensual das transações.

A ordem das transações é então definida com base nos eventos famosos, garantindo que todos os nós tenham a mesma visão da sequência de transações.

## **Tipos de Transações na Hashgraph**

A hashgraph suporta os seguintes tipos de transações:

### **1. Smart Contract Calls**

Tal como no Ethereum, a Hashgraph suporta a execução de contratos inteligentes, permitindo automatizar e executar acordos complexos sem a necessidade de intermediários.

Devido a uma grande percentagem de todos os contratos inteligentes no espaço da descentralização serem desenvolvidos para serem executados na Ethereum Virtual Machine escritos na linguagem Solidity, a Hedera decidiu também usar a EVM nos nós da sua rede para os executar, isto permite que facilmente se faça a migração desses contratos da rede Ethereum para a rede Hashgraph.

Estas razões são favoráveis a desenvolvedores que queiram hospedar as suas aplicações descentralizadas na rede Hashgraph, devido à grande documentação e ferramentas existentes para desenvolver em solidity

### **2. HCS Messages**

As mensagens do Hedera Consensus Service (HCS) permitem a publicação de dados dentro da rede Hashgraph.

Estas transações permitem que os utilizadores publiquem mensagens em tópicos, tópicos esses que podem ser criados pelos próprios utilizadores aos quais é atribuído uma identificação única.

As mensagens podem ser quaisquer dados que os utilizadores desejem transmitir de forma confiável e imutável, e iram ser codificadas em Base 64 e guardadas na rede.

### **3. Crypto Transfers**

Transações que envolvem a transferência de criptomoedas entre contas na rede Hashgraph, quer seja a token nativa da rede, HBAR, quer seja outra token

Qualquer conta na Hashgraph pode, através da Hedera Token Service (HTS), facilmente criar a sua própria token, de forma bastante simples e barata.

## **Redes e Ambientes de Teste**

Em prol dos clientes que queiram usar a rede desenvolvida pela Hedera, tal como a CO<sub>2</sub>Offset, existem 3 diferentes redes na Hashgraph:

- **Mainnet**

A Mainnet é a rede principal da Hedera Hashgraph onde as transações reais ocorrem. É a rede utilizada pelos utilizadores para transações oficiais, e imutáveis.

A Mainnet oferece todas as funcionalidades da Hashgraph com a garantia de segurança e desempenho.

- **Testnet**

A Testnet é um ambiente de teste público onde os desenvolvedores podem experimentar e testar novas funcionalidades sem impactar a rede principal.

Ideal para desenvolvimento e testes preliminares de aplicações antes da implementação na Mainnet, pois a testnet segue exatamente a mesma lógica que a mainnet.

Esta rede recebe um “reset” uma vez por trimestre, em que todos os seus dados são apagados, sendo que os dados estão disponíveis para recuperação até duas semanas depois do evento.

- **Previewnet**

A Previewnet é uma rede que está sob desenvolvimento pela Hedera, onde novas funcionalidades são testadas antes de serem lançadas na Testnet e na Mainnet.

Permite a validação de novas características e melhorias num ambiente controlado.

## **Mirror Node REST API**

Para permitir obter os dados que estão presentes nas redes a Hedera dispõe de uma API, a Mirror Node REST API, as Mirror Nodes permitem aos utilizadores aceder e interagir com os dados históricos da rede Hashgraph.

A API REST facilita a consulta de transações passadas, estados de conta e outros dados relevantes de maneira eficiente e programática, estas são os URL base de cada rede:



**Mainnet:** <https://mainnet.mirrornode.hedera.com/>

**Testnet:** <https://testnet.mirrornode.hedera.com/>

**Previewnet:** <https://previewnet.mirrornode.hedera.com/>

A Hedera Hashgraph oferece uma alternativa inovadora à tecnologia blockchain tradicional, utilizando um algoritmo de consenso eficiente e uma estrutura de registo distribuído que proporciona maior velocidade, segurança e sustentabilidade. Com funcionalidades avançadas como contratos inteligentes, mensagens através do Hedera Consensus Service e transferências de criptomoedas, além de um ambiente robusto de testes e desenvolvimento, a Hashgraph posiciona-se como uma solução poderosa para aplicações descentralizadas. A sua abordagem única e as ferramentas oferecidas tornam-na uma plataforma atraente para desenvolvedores e utilizadores que procuram eficiência e inovação no espaço das tecnologias de registo distribuído.

[3][7][10]

### **3.1.3 Porquê a Hashgraph**

A CO<sub>2</sub>Offset demonstrou interesse na transparência, que é um dos grandes problemas no mercado dos créditos de carbono, a solução a que chegaram foi registar informações sobre os créditos de carbono numa tecnologia DLT que garantisse imutabilidade.

Agora que percebemos como funcionam algumas destas tecnologias vamos analisar alguns dados que permitam fazer uma comparação das tecnologias DLT disponíveis no mercado, especialmente a rede Bitcoin e Ethereum, com a tecnologia Hashgraph que foi a escolhida pela CO<sub>2</sub>Offset, este capítulo deverá sustentar o que levou a optar pela Hashgraph.

Tendo em conta os valores da CO<sub>2</sub>Offset relativamente a questões ambientais e por ser uma empresa que atua no mercado dos créditos de carbono, vamos primeiramente analisar os impactos ambientais.

#### **Bitcoin**

A rede Bitcoin foi imediatamente considerada inviável para os objetivos da CO<sub>2</sub>Offset, cujo intuito é guardar informações na rede de forma eficiente e sustentável. A principal razão para esta inviabilidade está no seu impacto ambiental e nas suas limitações técnicas.

A Bitcoin utiliza um mecanismo de consenso conhecido como "Proof of Work" (PoW), que exige uma quantidade massiva de energia para validar transações e manter a segurança da rede. Este consumo energético não é compatível com os

valores ambientais da CO<sub>2</sub>Offset, uma empresa que atua no mercado dos créditos de carbono e procura minimizar a pegada ecológica.

Além disso, a Bitcoin foi concebida principalmente como uma moeda digital e não como uma plataforma para registrar informações. As suas capacidades para armazenar dados são limitadas e pouco práticas para a aplicação específica que a CO<sub>2</sub>Offset procurava. A estrutura da rede e as suas características técnicas não oferecem a flexibilidade necessária para uma implementação eficiente e sustentável do registo de créditos de carbono.

Para se ter noção do impacto ambiental da Bitcoin, a pegada de apenas uma transação é de aproximadamente 430 kgCO<sub>2</sub>, ou seja por cada transação são emitidos 430 quilogramas de dióxido de carbono, o que seria equivalente a quase 1 milhão de transações processadas pela VISA, isto ocorre devido ao custo computacional exigido pelo seu algoritmo de consenso.

## **Ethereum**

Por outro lado, a rede Ethereum permite armazenar dados na sua blockchain e, após o Merge, com a transição do algoritmo Proof of Work para o algoritmo Proof of Stake, houve uma redução de cerca de 99,95% no consumo de energia e, consequentemente, uma grande diminuição da sua pegada de carbono. Isto torna o Ethereum uma opção viável para os objetivos da CO<sub>2</sub>Offset. Por essa razão, é necessário comparar mais aspetos para entender a escolha da Hashgraph em detrimento de outras redes.

## **Comparação**

Tanto Ethereum como a Hashgraph têm capacidades para suportar as necessidades da CO<sub>2</sub>Offset, portanto iremos comparar mais aspetos para além do impacto ambiental.

Segundo a Chainspect, à data de escrita deste documento, a Hedera Hashgraph aparece em primeiro lugar no número em tempo real de transações por segundo, enquanto Ethereum aparece em 11º lugar e Bitcoin em 12º, a única rede que se aproxima da velocidade da Hashgraph é Solana com menos de metade da sua velocidade, no entanto, no que conta ao máximo de transações por segundo já registados 3 redes Blockchain superaram a velocidade da Hashgraph, uma delas a solana.

#	NAME	REAL-TIME TPS	MAX REC. TPS	MAX THEOR. TPS	BLOCK TIME	FINALITY	GOVERNANCE	LAUNCH DATE
1	Hedera	1,844 tx/s	3,287 tx/s	10,000 tx/s	2s	7s	Council	Sep 16, 2019
2	Solana	834 tx/s	7,229 tx/s	65,000 tx/s	0.47s	12.8s	Off-chain	Mar 16, 2020
3	Stellar	155 tx/s	176 tx/s	183 tx/s	5.93s	6s	On-chain	Jul 31, 2014
4	Sei	49.43 tx/s	256 tx/s	12,500 tx/s	0.4s	0.38s	On-chain	May 22, 2023
5	Base	44.85 tx/s	293 tx/s	1,429 tx/s	2s	16m	Off-chain	Aug 9, 2023
6	Tron	37.41 tx/s	236 tx/s	2,516 tx/s	3s	57s	On-chain	Jun 25, 2018
7	Polygon	36.19 tx/s	429 tx/s	649 tx/s	2.14s	4m 16s	Off-chain	May 30, 2020
8	opBNB	34.2 tx/s	548 tx/s	4,762 tx/s	1s	16m	Off-chain	Aug 11, 2023
9	BNB Chain	25.45 tx/s	1,731 tx/s	2,222 tx/s	3s	7.5s	On-chain	Sep 1, 2020
10	Arbitrum	22.82 tx/s	648 tx/s	40,000 tx/s	0.25s	16m	On-chain	Aug 31, 2021
11	Ethereum	12.6 tx/s	62.34 tx/s	119 tx/s	12s	16m	Off-chain	Jul 30, 2015
12	Bitcoin	12.05 tx/s	12.36 tx/s	7 tx/s	5m 22s	1h	Off-chain	Jan 3, 2009

Figura 7 - Tabela comparativa entre redes DLT

Para além de velocidade, Hashgraph também lidera na sustentabilidade, visto que, segundo a própria Hedera, cada transação emite 0.0000205494552 kgCO<sub>2</sub> para a atmosfera, para se ter melhor noção, assumindo que uma transação na rede Bitcoin emite 430 kgCO<sub>2</sub> (outras fontes dizem que este número pode ser pelo menos 2x superior), isto significa que para emitir este valor na rede Hashgraph seriam necessárias mais de 20 milhões de transações.

Agora vamos comparar a Hashgraph com redes mais sustentáveis como Ethereum e Solana:

## 1 transação...

- ... na rede Ethereum antes do merge emitia entre 64 a 110 kgCO<sub>2</sub>
- ... na rede Ethereum depois do merge emite cerca de 0.01 kgCO<sub>2</sub>
- ... na rede Solana emite cerca de 0.0002428 kgCO<sub>2</sub>
- ... na rede Hashgraph emite cerca de 0.00002054 kgCO<sub>2</sub>

Como podemos verificar por estes dados a rede Hashgraph para além de ser a mais rápida é também a mais amiga do ambiente, como se não bastasse é também a opção que mais facilita a armazenagem de dados na rede, do ponto de vista prático e também a mais barata financeiramente, pois têm o tipo de transação “HCS Submit Message”

que é uma transação que permite armazenar uma mensagem arbitrária na rede, sendo que estes têm de estar submetidos num “Topic”, esse topic custa 0,01 dólares para ser criado e cada mensagem tem o custo de 0,0001 dólares, este processo irá ser detalhado no capítulo que engloba o desenvolvimento da aplicação.

Havendo uma rede como a Hashgraph que é a opção mais sustentável, com uma grande velocidade e com um baixíssimo custo, tornou a decisão da rede DLT a usar para este projeto bastante simples.

Uma comparação mais detalhada entre Hashgraph e Bitcoin/Ethereum/Solana, no que toca principalmente a velocidade pode ser encontrada nos **Anexos B,C e D**.

[1][2][7][4][15]

## **3.2 Créditos de carbono**

Para entender a necessidade de transparência no mercado dos créditos de carbono, é essencial perceber o que são créditos de carbono.

### **O que é um crédito de carbono**

Um crédito de carbono é uma unidade de medida que representa a redução ou remoção de uma tonelada de dióxido de carbono (CO<sub>2</sub>) ou de gases com efeito de estufa (GEE) equivalente da atmosfera. Estes créditos são gerados através de projectos que evitam, reduzem ou removem emissões de GEE, como por exemplo, iniciativas de reflorestação, energias renováveis, melhoria da eficiência energética ou gestão de resíduos.

O conceito de crédito de carbono surgiu no contexto do Protocolo de Quioto, um tratado internacional assinado em 1997 e que entrou em vigor em 2005, com o objectivo de combater as alterações climáticas. O protocolo estabeleceu metas vinculativas de redução de emissões para os países desenvolvidos e criou mecanismos de mercado, como o Mecanismo de Desenvolvimento Limpo (MDL) e a Implementação Conjunta (IC), que permitiam a criação e troca de créditos de carbono.

Empresas compram créditos de carbono para compensar as suas próprias emissões de GEE. A compensação é necessária quando uma empresa não consegue reduzir as suas emissões internamente a um nível aceitável devido a limitações tecnológicas, operacionais ou financeiras. Comprar créditos de carbono permite que as empresas assumam responsabilidade pelas suas emissões, investindo em projectos que reduzem ou removem GEE noutros locais. Este processo é frequentemente motivado por regulamentos governamentais que estabelecem limites de emissões, bem como por pressão de consumidores, investidores e outras partes interessadas que exigem práticas empresariais mais sustentáveis.

Para ilustrar o processo de criação e comercialização de um crédito de carbono, vejamos um exemplo prático:

1. **Criação do projeto:** Uma empresa de energia renovável decide instalar uma nova fazenda de painéis solares numa região com alta incidência solar. Este projecto visa substituir a electricidade gerada por centrais a carvão, que têm uma elevada emissão de CO<sub>2</sub>.
2. **Cálculo das reduções de emissões:** Antes do início do projecto, é calculado o potencial de redução de emissões. Suponha-se que a fazenda solar irá reduzir as emissões em 10.000 toneladas de CO<sub>2</sub> por ano, comparado com a electricidade gerada por carvão.
3. **Validação e certificação:** O projecto é submetido a uma entidade de certificação, que valida o cálculo das reduções de emissões. Após a verificação, a entidade certifica que o projecto irá gerar 10.000 créditos de carbono por ano.
4. **Monitorização contínua:** A empresa monitora continuamente a produção de energia e as reduções de emissões alcançadas. Relatórios regulares são enviados à entidade certificadora para verificar que as reduções estão a ser efectivamente alcançadas.
5. **Emissão dos créditos de carbono:** Com base nos relatórios de monitorização, a entidade certificadora emite os créditos de carbono correspondentes às reduções de emissões verificadas. No primeiro ano, são emitidos 10.000 créditos de carbono.
6. **Comercialização dos créditos:** A empresa pode agora vender os créditos de carbono no mercado. Uma multinacional que deseja compensar as suas próprias emissões compra 5.000 desses créditos, pagando um valor acordado por cada crédito.

## **Importância da transparência**

A transparência é fundamental no mercado de créditos de carbono por diversas razões. Primeiramente, garante a integridade ambiental dos créditos comercializados.

Sem transparência, há o risco de créditos duvidosos ou falsos serem introduzidos no mercado, comprometendo os esforços globais de mitigação das alterações climáticas. Além disso, a transparência assegura que os projectos de geração de créditos de carbono são verdadeiramente adicionais, ou seja, que não teriam ocorrido na ausência do incentivo financeiro proporcionado pela venda dos créditos.

Em segundo lugar, a transparência fomenta a confiança dos investidores e das partes interessadas no mercado de créditos de carbono. Quando as informações sobre a origem, verificação e monitorização dos créditos são facilmente acessíveis e verificáveis, aumenta-se a credibilidade do mercado e atrai-se mais investimento. Este investimento é crucial para financiar projectos de redução de emissões que, de outra forma, poderiam não ser economicamente viáveis.

Terceiramente, a transparência facilita a responsabilização e a monitorização contínua dos projectos. As partes interessadas, incluindo governos, organizações não-governamentais e o público em geral, podem acompanhar o desempenho dos projectos e assegurar que estes cumprem com as suas promessas ambientais. A verificação independente e a divulgação pública de relatórios são práticas que reforçam este aspecto.

Por fim, a transparência contribui para a equidade e justiça no mercado de créditos de carbono. Ela garante que os benefícios dos projectos de mitigação de emissões sejam distribuídos de forma justa, incluindo às comunidades locais que muitas vezes estão na linha da frente das iniciativas ambientais. Sem mecanismos transparentes, existe o risco de marginalização ou exploração dessas comunidades.

Em resumo, a transparência no mercado de créditos de carbono é essencial para garantir a sua integridade, credibilidade e eficácia. Sem ela, o potencial dos créditos de carbono para contribuir significativamente para a redução das emissões globais de GEE e para o combate às alterações climáticas seria severamente comprometido. Uma das soluções para assegurar esta transparência pode estar nas redes DLT, onde a imutabilidade e a rastreabilidade se tornam aliados na luta contra as alterações climáticas. Portanto, é crucial que os reguladores, empresas e todas as partes envolvidas promovam e implementem práticas transparentes em todas as fases da criação e comercialização dos créditos de carbono.

[14]

### 3.3 Vite + React

#### O que é o Vite?

O Vite é uma ferramenta de construção moderna para o desenvolvimento web, criada por Evan You, o mesmo criador do Vue.js. O nome "Vite" vem do francês e significa "rápido", o que reflete o seu principal objetivo: proporcionar um ambiente de desenvolvimento extremamente rápido e eficiente.

O Vite tira partido das mais recentes funcionalidades nativas dos navegadores modernos, como os módulos ES, para oferecer uma experiência de desenvolvimento quase instantânea, independentemente do tamanho da aplicação. Esta rapidez é conseguida através de uma abordagem de "não-bundle" durante o desenvolvimento, o que significa que o código é servido diretamente, sem ser agrupado.

#### Vite com React e TypeScript

No projeto em questão, o Vite foi utilizado em conjunto com React e TypeScript (react-ts). Esta combinação oferece várias vantagens significativas:

1. **Configuração rápida:** O Vite fornece modelos pré-configurados para React com TypeScript, permitindo iniciar um projeto rapidamente e sem complicações.
2. **Tipagem estática:** O TypeScript adiciona tipagem estática ao JavaScript, melhorando significativamente a deteção de erros e a autocompleção no ambiente de desenvolvimento integrado (IDE).
3. **Desempenho otimizado:** O Vite é otimizado para React, oferecendo um excelente desempenho tanto no ambiente de desenvolvimento como na produção.
4. **Hot Module Replacement (HMR)**

O Hot Module Replacement (HMR) é uma funcionalidade crucial do Vite que merece uma explicação técnica detalhada.

##### Como funciona o HMR no Vite:

- a. **Deteção de alterações:** O Vite observa constantemente os ficheiros do projeto. Quando uma alteração é detetada em qualquer ficheiro, o Vite inicia imediatamente o processo de HMR.
- b. **Análise de dependências:** O Vite analisa o módulo alterado e determina quais outros módulos podem ser afetados por esta alteração, criando uma árvore de dependências.
- c. **Atualização do módulo:** O módulo alterado é recarregado no navegador sem atualizar toda a página, mantendo o estado atual da aplicação.

- d. **Propagação das alterações:** As alterações são propagadas através da árvore de dependências, atualizando apenas os componentes afetados, o que garante uma atualização eficiente e focada.
- e. **Preservação do estado:** O estado da aplicação é preservado sempre que possível, permitindo que se veja imediatamente o efeito das alterações sem perder o contexto atual da aplicação.
- f. **Comunicação via WebSocket:** O Vite utiliza uma conexão WebSocket para comunicar as alterações ao cliente, permitindo atualizações quase instantâneas na interface do utilizador.

No contexto do React, o HMR do Vite é particularmente eficiente. Ele pode atualizar componentes React individuais sem perder o seu estado, proporcionando uma experiência de desenvolvimento extremamente fluida e responsiva.

## **Desempenho Rápido no Desenvolvimento**

O Vite mantém um desempenho impressionante durante o desenvolvimento, mesmo com muitas dependências, graças a várias estratégias avançadas:

1. **Non-bundling em desenvolvimento:** O Vite serve os ficheiros diretamente ao navegador usando módulos ES nativos. Isto significa que não há necessidade de um passo de bundling durante o desenvolvimento, o que acelera significativamente o processo de inicialização.
2. **Carregamento sob demanda:** Apenas os módulos necessários para a renderização inicial são carregados, com os restantes sendo carregados conforme necessário, o que reduz o tempo de carregamento inicial.
3. **Pré-bundling de dependências:** O Vite pré-bundle as dependências usando esbuild, que é extremamente rápido. Isto é feito apenas uma vez e os resultados são colocados em cache para uso futuro.
4. **Caching agressivo:** O Vite utiliza caching agressivo para evitar trabalho desnecessário. As dependências pré-bundled e os resultados de transformação são cuidadosamente armazenados em cache, reduzindo o tempo de processamento em carregamentos subsequentes.

## **Tratamento do TypeScript**

O Vite lida com o TypeScript de forma particularmente eficiente:

1. **Transpilação rápida:** O Vite usa o esbuild para transpilar TypeScript para JavaScript, o que é cerca de 20-30 vezes mais rápido que o compilador TypeScript padrão. Esta velocidade é crucial para manter um ambiente de desenvolvimento ágil.



2. **Skipping da verificação de tipos:** Durante o desenvolvimento, o Vite "salta" a verificação de tipos para manter a velocidade. Isto significa que erros de tipo não são reportados em tempo real durante o desenvolvimento, priorizando a rapidez de feedback.
3. **Verificação de tipos separada:** Para manter a segurança de tipos, é recomendado executar a verificação de tipos num processo separado ou usar um plugin de IDE. Isto permite que os desenvolvedores obtenham feedback sobre erros de tipo sem afetar a velocidade de recarga do Vite.

## 5. Rollup no Vite

O Rollup é um bundler de módulos JavaScript que o Vite utiliza para a construção de produção. Enquanto o Vite utiliza uma abordagem de não-bundle durante o desenvolvimento, ele recorre ao Rollup para criar bundles otimizados para produção.

## 6. Como o Rollup funciona no Vite:

- a. **Tree-shaking:** O Rollup analisa o código e elimina código morto, incluindo apenas o que é realmente utilizado na aplicação final, reduzindo significativamente o tamanho do bundle.
- b. **Code-splitting:** O Rollup divide o código em chunks menores, permitindo o carregamento lazy de partes da aplicação conforme necessário, melhorando o desempenho inicial.
- c. **Minificação e otimização:** O código é minificado e otimizado para reduzir o tamanho dos ficheiros finais, melhorando os tempos de carregamento.
- d. **Gestão de dependências:** O Rollup gere eficientemente as dependências externas, incluindo-as no bundle final de forma otimizada.
- e. **Plugins:** O Vite aproveita o ecossistema de plugins do Rollup para estender as suas capacidades, como o processamento de diferentes tipos de ficheiros.
- f. **Configuração automática:** O Vite configura automaticamente o Rollup para produção, otimizando as definições para o melhor desempenho possível.

## 7. Comandos de Build e Code-Splitting

Para criar uma build de produção com o Vite, utiliza-se o seguinte comando:

**npm run build**

Durante o processo de build, o Vite realiza várias otimizações:

- a. **Bundling com Rollup:** O Vite usa o Rollup para criar bundles otimizados para produção, garantindo o melhor desempenho possível.

- b. **Code-splitting automático:** O Vite implementa code-splitting automático baseado em:
  - Importações dinâmicas: Qualquer importação dinâmica (`import()`) resulta num chunk separado.
  - CSS splitting: O CSS é extraído para ficheiros separados, permitindo o carregamento paralelo de JS e CSS.
  - Chunks de biblioteca: Dependências comuns são agrupadas em chunks separados para melhor caching.
- c. **Lazy-loading:** Os chunks gerados pelo code-splitting são carregados de forma lazy, apenas quando necessários, melhorando o desempenho inicial da aplicação.
- d. **Tree-shaking:** O Rollup realiza tree-shaking eficiente, removendo código não utilizado e reduzindo o tamanho final do bundle.
- e. **Minificação:** O código é minificado para reduzir ainda mais o tamanho dos ficheiros finais, otimizando o tempo de carregamento.
- f. **Geração de source maps:** Source maps são gerados para facilitar o debugging em produção, permitindo aos desenvolvedores localizar problemas no código original mesmo após a minificação.

Para personalizar o processo de build, pode-se configurar opções adicionais no ficheiro `vite.config.js`, como definir o diretório de saída, configurar a base path, ou ajustar as estratégias de code-splitting.

O code-splitting no Vite é altamente otimizado e, na maioria dos casos, não requer configuração manual. No entanto, para casos específicos, é possível ajustar o comportamento através de comentários especiais ou configurações no `vite.config.js`.

Estas características combinadas fazem do Vite uma ferramenta extremamente eficiente tanto para desenvolvimento quanto para produção, oferecendo uma experiência de desenvolvimento rápida sem comprometer a qualidade e o desempenho do produto final.

[11][12]

## 4 PLANEAMENTO DO PROJETO

### 4.1 Objetivos do Projeto

1. **Desenvolver uma aplicação web** que extraia dados da Hashgraph através da Testnet Mirror Node API da Hedera.
2. **Apresentar de forma clara e detalhada:**
  - **Ações que geraram créditos de carbono:** Mostrar todas as ações que resultaram na geração de créditos de carbono, incluindo dados como data, descrição da ação e quantidade de créditos gerados.
  - **Transações de venda de créditos de carbono:** Listar todas as transações de venda de créditos de carbono, com informações como data, valor dos créditos e partes envolvidas.
3. **Permitir aos utilizadores:**
  - **Pesquisar:** Facilitar a localização rápida de ações e transações através de uma barra de pesquisa.
  - **Filtrar:** Oferecer opções de filtragem avançadas com base em critérios como data, valor, tipo de ação ou transação, e identificadores únicos.
  - **Visualizar:** Fornecer detalhes completos sobre as ações e transações selecionadas.

### Restrições e Liberdades

- **Tecnologia:** Utilização da Testnet Mirror Node API da Hedera Hashgraph para obter informações.
- **Desenvolvimento:** Front-end desenvolvido com Vite e React.
- **Liberdade Criativa:** Ampla liberdade no design da interface e na implementação de funcionalidades adicionais para melhorar a experiência do utilizador.
- **Idiomas:** A interface estará disponível em português, inglês e espanhol, atendendo à demografia dos clientes da CO<sub>2</sub>Offset.

### Metodologia de Desenvolvimento

O desenvolvimento seguirá uma abordagem iterativa com ciclos contínuos de feedback e refinamento. O processo incluirá:

1. **Prototipagem:** Desenvolvimento do design inicial utilizando Figma.

2. **Feedback e Aprovação:** Submissão do design para feedback e iterações até obter a aprovação final.
3. **Desenvolvimento:** Início do desenvolvimento utilizando as ferramentas e tecnologias.
4. **Testes e Refinamento:** Realização de ciclos contínuos de testes e melhorias com base no feedback e resultados obtidos.

## **Ferramentas e Ambiente de Desenvolvimento**

- **Controlo de Versões:** Utilização do GitHub para gestão do código e colaboração, de acordo com o modelo de branching.
- **Ambiente de Desenvolvimento:** Docker para garantir consistência entre ambientes e deployment para servidor em cloud.
- **IDE:** Visual Studio Code (VSC) com plugins relevantes.
- **Design:** Figma para prototipagem e design inicial.

## **4.2 Análise de Requisitos**

### **4.2.1 Requisitos Funcionais**

1. **Listagem de Transações e Ações:**
  - Apresentar uma lista de todas as transações de venda de créditos de carbono ou de ações que geraram créditos de carbono.
  - Incluir informações básicas como data, valor dos créditos e partes envolvidas.
2. **Detalhes das Transações e Ações:**
  - Permitir que os utilizadores selecionem uma transação/ação específica para visualização detalhada.
  - Mostrar detalhes adicionais como identificador da transação, quantidade de créditos, preço unitário e total, que não apareceriam na lista.
3. **Pesquisa e Filtros:**
  - Fornecer uma barra de pesquisa para localização rápida de ações e transações.

- Implementar opções de filtros avançados baseados em critérios como data, valor, tipo de ação ou transação, e identificadores únicos.

## **4.2.2 Requisitos Não Funcionais**

### **1. Desempenho:**

- Garantir tempos de resposta rápidos, especialmente para operações de pesquisa e filtragem.
- Assegurar a escalabilidade para suportar o aumento no volume de transações e utilizadores.

### **2. Transparência e Acessibilidade**

- Disponibilizar todos os dados a todos os utilizadores sem necessidade de autenticação.
- Garantir acesso livre e irrestrito às informações sobre transações e ações.

### **3. Usabilidade:**

- Desenvolver uma interface intuitiva e fácil de navegar.
- Criar uma experiência de utilizador agradável e eficiente.
- Fornecer documentação clara e concisa para auxiliar na utilização do sistema.

### **4. Interface:**

- Disponibilizar a interface em português, inglês e espanhol.
- Criar um design coerente com a identidade visual da CO<sub>2</sub>Offset.
- Implementar uma página inicial com um layout minimalista e funcional.



## **5 DESENVOLVIMENTO DO BLOCK EXPLORER**

Neste capítulo, será detalhado todo o processo envolvido no desenvolvimento da aplicação web, incluindo a cronologia do seu desenvolvimento, a utilização das ferramentas abordadas no capítulo 2, "Metodologia de Trabalho", e os problemas e desafios encontrados ao longo do projeto. Também será abordado o processo de design da aplicação, destacando como as decisões de design influenciaram a interface e a experiência do utilizador, e como essas decisões foram integradas com o desenvolvimento. Além disso, serão discutidas as dificuldades enfrentadas e as soluções encontradas durante todas as fases do projeto.

Todo o processo relatado neste capítulo foi conduzido por uma única pessoa, o autor deste relatório, com o auxílio e feedback de outras pessoas.

### **5.1 Design**

Para iniciar o processo de design, é necessário determinar os requisitos que o website deve cumprir. Estes requisitos são:

1. Mostrar todas as transações referentes à venda de créditos de carbono e todas as transações referentes à criação de créditos de carbono.
2. Permitir a pesquisa de transações por parâmetros específicos.

Foram também estabelecidas algumas condições e preferências de design:

1. Utilizar as cores da empresa.
2. Manter a página inicial simplificada.

Antes de iniciar o processo de design, é fundamental conhecer a estrutura dos dados de cada transação. Infelizmente, a informação fornecida sobre esta estrutura não era definitiva. Caso a estrutura dos dados fosse significativamente diferente da prevista inicialmente, poderiam surgir constrangimentos e haver necessidade de ajustes no design.

O que foi facultado na fase pré-design em relação aos dados a serem mostrados ao utilizador foi o seguinte: existiriam dois tipos de dados, as "Transactions" e as "Credit Pools". As "Transactions" referem-se à venda de créditos de carbono a uma empresa, enquanto as "Credits Pool" referem-se a uma ação que gerou créditos de carbono. Aqui está a estrutura prevista dos dados:

#### **Transactions:**

- buyer: String – Nome do comprador

- buyerNIF: String – NIF do comprador
- buyerWallet: Srtring – Carteira do comprador
- typeOfCredit: String – Tipo de crédito
- startDate: Date – Data de Inicio
- endDate: Date – Data Final
- Reference: String - id de outra transação (facultativo)
- Motive: String – motivo da compra (facultativo)
- Composition – Array com a quantidade de créditos vindos de uma respetiva pool
  - quantityOfCredits: Int – Quantidade de créditos de uma determinada pool presentes
  - pool: Id – id da pool
  - quantityOfCredits
  - pool: id da pool
  - quantityOfCredits
  - ...

#### **Credits Pool:**

- Action\_date: Date – Data da ação
- Location: Polígono – coordenadas de pontos do poligono {12.346567655, 3.486858595905...}
- Action: String –Designação da ação
- Species: String – Espécies mais abundantes (pinheiro, eucalipto)
- Credits: Float – Número de créditos gerados pela ação
- Co2stock Tracked: Float – Número de créditos rastreados
- Co2stock Additionality: Float – Número de créditos gerados para além dos rastreados
- Co2stock Additionality ML Bias: Float – de 0 a 1, mede a diferença entre as medições previstas por um modelo de Machine Learning e as medições reais.
- Co2stock Additionality ML R2: Float - Mede a precisão do modelo de Machine learning, um valor próximo de 1 indica que o modelo é muito preciso.
- Certification: URL direto para o documento que certifica a ação.



- Reference: String - Id de outra pool (facultativo)
- Motive: string – motivo da pool (facultativo)

Neste momento temos informações suficientes para iniciar o processo de design.

O design foi realizado na plataforma Figma. Os botões eram funcionais e levavam a outras páginas, simulando de forma realista o comportamento do website final. Foram utilizados componentes reutilizáveis para garantir a consistência e a eficiência no processo de design. Além disso, até estilos de hover foram implementados para proporcionar uma experiência de utilizador mais interativa, todos os dados que irão ser mostrados são fictícios e apenas para fins de visualização e teste.

Vamos começar com a página inicial, as únicas restrições no design desta página era que a página deveria de ser minimalista, para além de usar as cores da empresa, esta foi a primeira versão do design desta página.

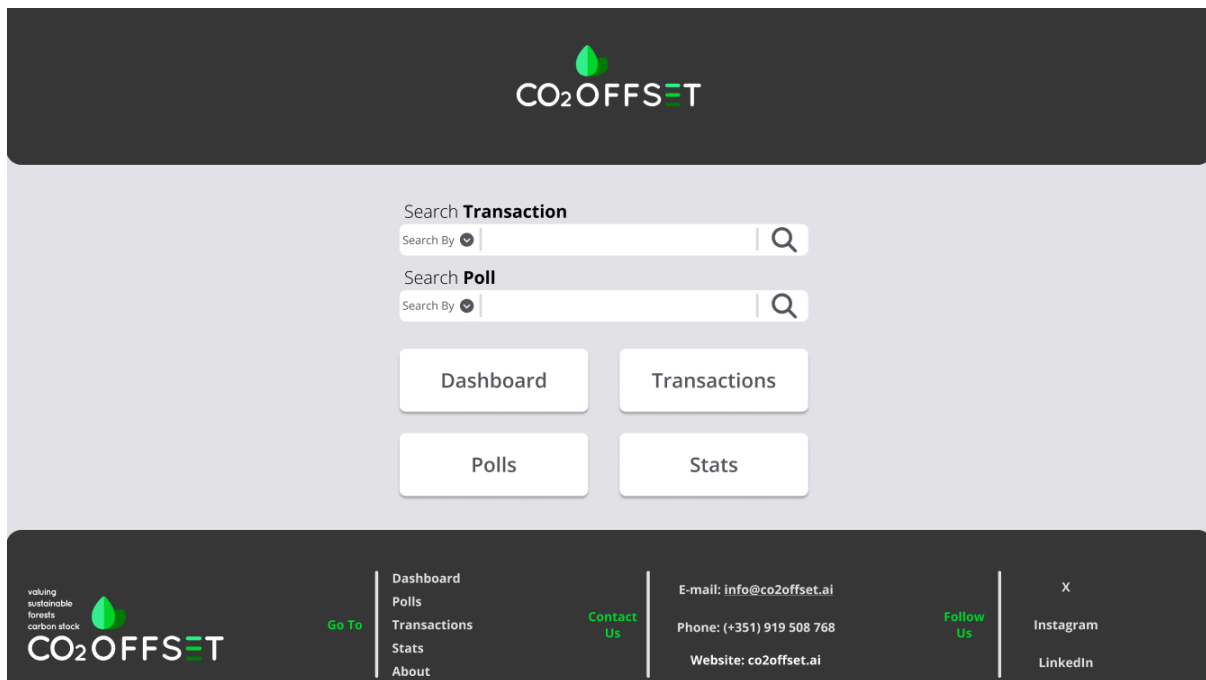


Figura 8 -Primeiro design da página inicial

Esta página permitiria pesquisar Pools (Devido a um erro de semântica, as 'pools' eram anteriormente denominadas 'polls') e transações por diferentes parâmetros, bem como botões para navegar para diferentes páginas, no entanto após receber algum feedback foram recebidas intruções para alterar este design para um design ainda mais simples, bem como outras mudanças, este foi o design final da página inicial.

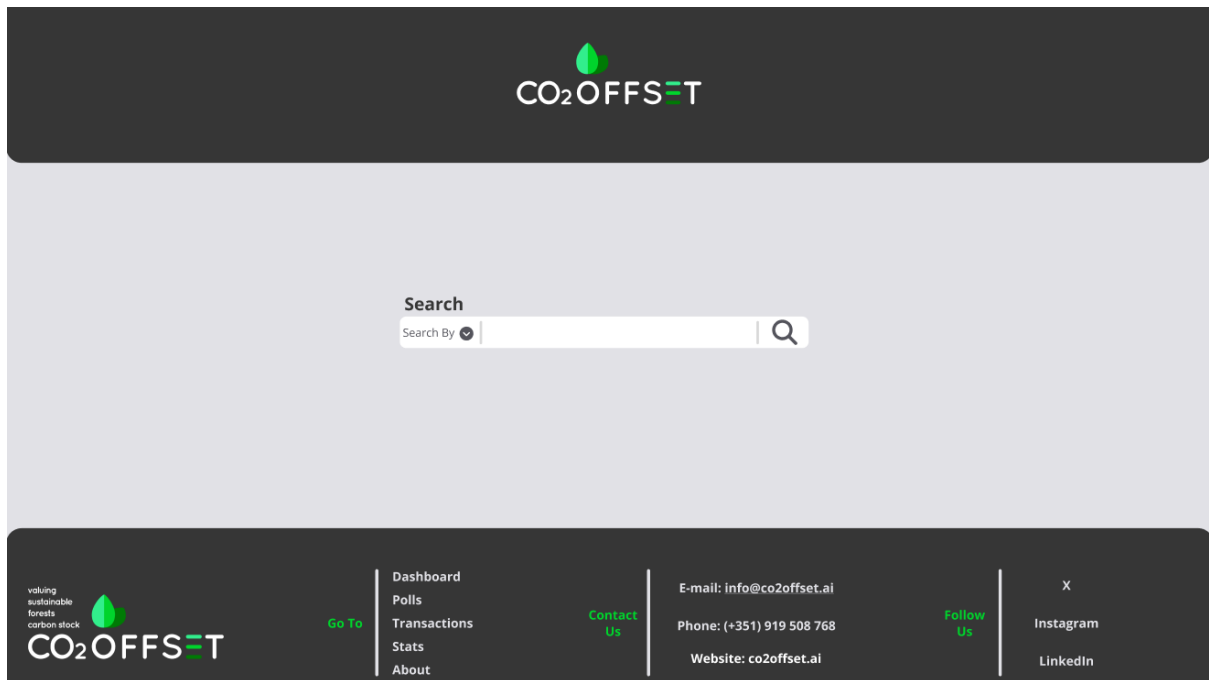


Figura 9 - Design final da página inicial

Nesta versão existe apenas uma barra de pesquisa para pesquisar apenas “Transactions” ou seja era impossível pesquisar pools, esta versão da página inicial teve um feedback positivo e por isso manteve-se.

Após a pesquisa o utilizador deverá ser levado para ver a lista das respetivas transações, caso o utilizador não escreva nada na barra de pesquisa irá ver a lista de todas as transações, este foi o design elaborado para esta página.

Token	Buyer	Buyer Wallet	Type Of Credit	Date	Details
Smart Contract	CO2Offset	0.0.858580	Sequoia Rise	7th February 2024	<a href="#">Details</a>
Smart Contract	CO2Offset	0.0.858580	Sequoia Rise	7th February 2024	<a href="#">Details</a>
Smart Contract	CO2Offset	0.0.858580	Sequoia Rise	7th February 2024	<a href="#">Details</a>
Smart Contract	CO2Offset	0.0.858580	Sequoia Rise	7th February 2024	<a href="#">Details</a>
Smart Contract	CO2Offset	0.0.858580	Sequoia Rise	7th February 2024	<a href="#">Details</a>
Smart Contract	CO2Offset	0.0.858580	Sequoia Rise	7th February 2024	<a href="#">Details</a>
Smart Contract	CO2Offset	0.0.858580	Sequoia Rise	7th February 2024	<a href="#">Details</a>
Smart Contract	CO2Offset	0.0.858580	Sequoia Rise	7th February 2024	<a href="#">Details</a>
Smart Contract	CO2Offset	0.0.858580	Sequoia Rise	7th February 2024	<a href="#">Details</a>
Smart Contract	CO2Offset	0.0.858580	Sequoia Rise	7th February 2024	<a href="#">Details</a>
Smart Contract	CO2Offset	0.0.858580	Sequoia Rise	7th February 2024	<a href="#">Details</a>
Smart Contract	CO2Offset	0.0.858580	Sequoia Rise	7th February 2024	<a href="#">Details</a>

Figura 10 - Design da página "Transactions"

Nesta imagem podemos ver o design da página onde o utilizador pode ver as transações que pesquisou (ou todas), e que permite efetuar novas pesquisas, filtrar e ordenar os dados, e é também possível verificar os detalhes de uma transação específica, também podemos ver um header onde se pode navegar entre as páginas.

Caso o utilizador clique para ver os detalhes de uma transação é levado para uma página onde deve ser possível ver atributos da transação adicionais que não se encontram na tabela, este foi o design implementado para essa página:

**Transaction**  
1707306625.408908810

**Smart Contract**

<b>Buyer</b> CO2Offset, SA	<b>Total Credits</b> 16.42 from 15 pools
<b>Buyer Wallet</b> 0.0.858580	<b>Start Date</b> 1st December 2023
<b>Type Of Credit</b> Sequoia Rise	<b>End Date</b> 24th January 2024

**Transactions**

Buyer	Type Of Credit	Date	Details
CO2Offset	Sequoia Rise	19th February 24	<a href="#">Details</a>
CO2Offset	Sequoia Rise	19th February 24	<a href="#">Details</a>
CO2Offset	Sequoia Rise	19th February 24	<a href="#">Details</a>
CO2Offset	Sequoia Rise	19th February 24	<a href="#">Details</a>
CO2Offset	Sequoia Rise	19th February 24	<a href="#">Details</a>
CO2Offset	Sequoia Rise	19th February 24	<a href="#">Details</a>

**Polls**

Action	Credits	Species	Date	Details
Thinning	17.53	Pinus Pinaster	1st December 2023	<a href="#">Details</a>
Thinning	17.53	Pinus Pinaster	1st December 2023	<a href="#">Details</a>
Thinning	17.53	Pinus Pinaster	1st December 2023	<a href="#">Details</a>
Thinning	17.53	Pinus Pinaster	1st December 2023	<a href="#">Details</a>
Thinning	17.53	Pinus Pinaster	1st December 2023	<a href="#">Details</a>
Thinning	17.53	Pinus Pinaster	1st December 2023	<a href="#">Details</a>

**CO2OFFSET**

Dashboard  
Polls  
Transactions  
Stats  
About

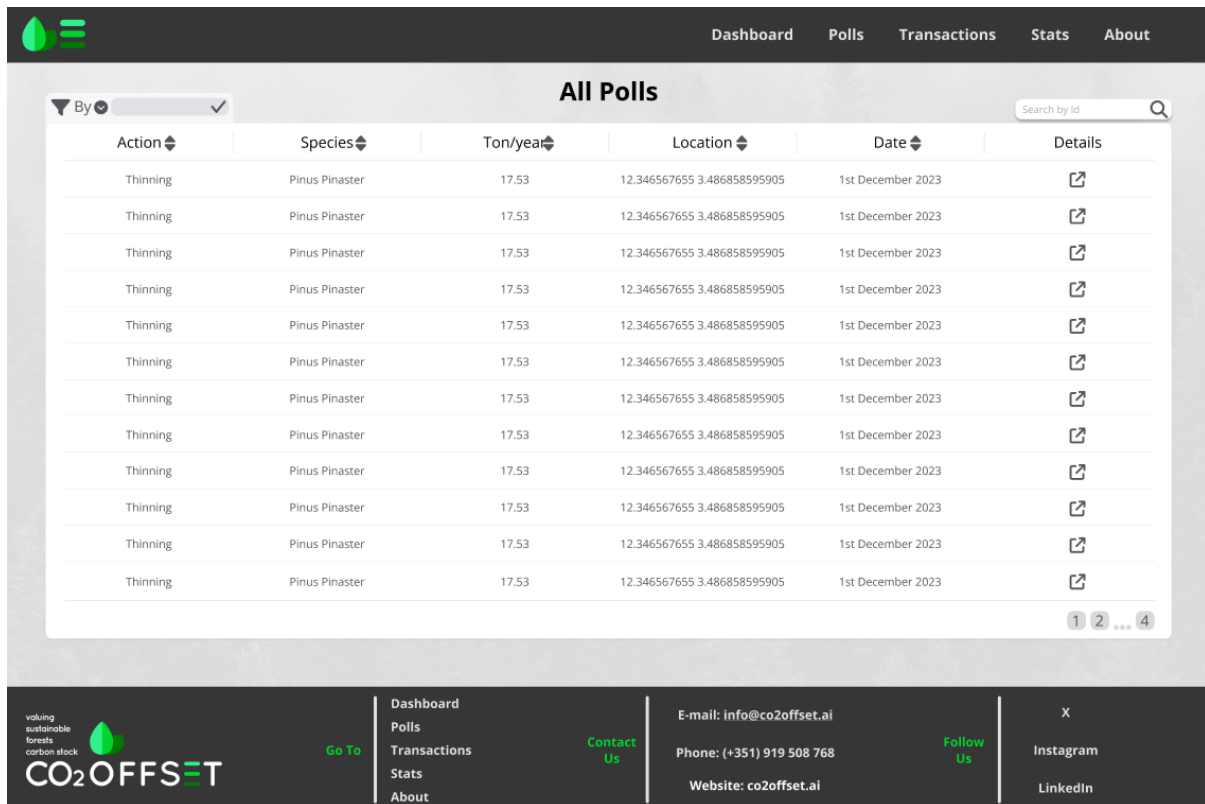
E-mail: [info@co2offset.ai](mailto:info@co2offset.ai)  
Phone: (+351) 919 508 768  
Website: [co2offset.ai](https://co2offset.ai)

X  
Instagram  
LinkedIn

Figura 11 - Design da página referente aos detalhes de uma transação

Como é possível ver o Header e Footer mantêm-se e é possível ver os detalhes todos de uma transação específica, bem como outras transações e as pools que foram usadas na composição desta transação.

No Header o utilizador pode navegar para a página “Polls”, posteriormente denominada Pools, esta segue a mesma lógica das transações, havendo uma página com as mesmas funcionalidades porém adaptadas para o seu tipo de dados para listar todas as pools e também uma página para ver os seus detalhes:



Action	Species	Ton/year	Location	Date	Details
Thinning	Pinus Pinaster	17.53	12.346567655 3.486858595905	1st December 2023	
Thinning	Pinus Pinaster	17.53	12.346567655 3.486858595905	1st December 2023	
Thinning	Pinus Pinaster	17.53	12.346567655 3.486858595905	1st December 2023	
Thinning	Pinus Pinaster	17.53	12.346567655 3.486858595905	1st December 2023	
Thinning	Pinus Pinaster	17.53	12.346567655 3.486858595905	1st December 2023	
Thinning	Pinus Pinaster	17.53	12.346567655 3.486858595905	1st December 2023	
Thinning	Pinus Pinaster	17.53	12.346567655 3.486858595905	1st December 2023	
Thinning	Pinus Pinaster	17.53	12.346567655 3.486858595905	1st December 2023	
Thinning	Pinus Pinaster	17.53	12.346567655 3.486858595905	1st December 2023	
Thinning	Pinus Pinaster	17.53	12.346567655 3.486858595905	1st December 2023	
Thinning	Pinus Pinaster	17.53	12.346567655 3.486858595905	1st December 2023	
Thinning	Pinus Pinaster	17.53	12.346567655 3.486858595905	1st December 2023	

Figura 12 - Design da página pools




[Dashboard](#)
[Polls](#)
[Transactions](#)
[Stats](#)
[About](#)


[<Back](#)

## Poll Details


Poll 1231

<b>Action</b> Thinning	<b>CO2 Stock tracked</b> 5.6	<b>Additionality ML R2</b> 0.36
<b>Action Date</b> 1st December 2023	<b>Additionality</b> 10.0	<b>Ton/year</b> 17.53 (1.11 left)
<b>Location</b> 12.3465676 3.4868585959	<b>Additionality ML Bias</b> 0.265	<b>Certification</b> 
<b>Species</b> Pinus Pinaster	<b>Reference</b> Poll 1321 	<b>Hazard Event</b> Fire

### Transactions

Buyer	Type Of Credit	Date	Details
CO2Offset	Sequoia Rise	19th February 24	
CO2Offset	Sequoia Rise	19th February 24	
CO2Offset	Sequoia Rise	19th February 24	
CO2Offset	Sequoia Rise	19th February 24	
CO2Offset	Sequoia Rise	19th February 24	
CO2Offset	Sequoia Rise	19th February 24	

### Polls

Action	Credits	Species	Date	Details
Thinning	17.53	Pinus Pinaster	1st December 2023	
Thinning	17.53	Pinus Pinaster	1st December 2023	
Thinning	17.53	Pinus Pinaster	1st December 2023	
Thinning	17.53	Pinus Pinaster	1st December 2023	
Thinning	17.53	Pinus Pinaster	1st December 2023	
Thinning	17.53	Pinus Pinaster	1st December 2023	



[Go To](#)

[Dashboard](#)
[Polls](#)
[Transactions](#)
[Stats](#)
[About](#)

[Contact Us](#)

E-mail: [info@co2offset.ai](mailto:info@co2offset.ai)  
Phone: (+351) 919 508 768  
Website: [co2offset.ai](https://co2offset.ai)

[Follow Us](#)

[X](#)  
[Instagram](#)  
[LinkedIn](#)

Figura 13 - Design da página referente aos detalhes de uma pool

De seguida temos a página Dashboard, que serviria para ver as últimas transações, as pools com créditos disponíveis e a lista com as empresas que adquiriram mais créditos, porém foi decidido posteriormente que esta página não iria ser

implementada pelo menos nesta fase, por falta de condições necessárias, ainda assim fica aqui como ficou o seu design:

**Latest Transactions**

Buyer	Wallet	Type Of Credit	Age	Details
CO2Offset	0.0.858580	Sequoia Rise	8 sec ago	<a href="#">Details</a>
CO2Offset	0.0.858580	Sequoia Rise	5 min ago	<a href="#">Details</a>
CO2Offset	0.0.858580	Sequoia Rise	4 hr ago	<a href="#">Details</a>
CO2Offset	0.0.858580	Sequoia Rise	20 hr ago	<a href="#">Details</a>
CO2Offset	0.0.858580	Sequoia Rise	2 d ago	<a href="#">Details</a>
CO2Offset	0.0.858580	Sequoia Rise	3 d ago	<a href="#">Details</a>

**Active Polls**

Action	Credits	Species	Date	Details
Thinning	17.53	Pinus Pinaster	1st December 2023	<a href="#">Details</a>
Thinning	17.53	Pinus Pinaster	1st December 2023	<a href="#">Details</a>
Thinning	17.53	Pinus Pinaster	1st December 2023	<a href="#">Details</a>
Thinning	17.53	Pinus Pinaster	1st December 2023	<a href="#">Details</a>
Thinning	17.53	Pinus Pinaster	1st December 2023	<a href="#">Details</a>
Thinning	17.53	Pinus Pinaster	1st December 2023	<a href="#">Details</a>

**Most Nature-Friendly**

Buyer	Credits
CO2Offset	16.42
Apple	0
Altri	0
Alphabet	0
Berkshire Hathaway	0
Pascoal e Pinto	0
Mota-Engil	0
EDP España	0
REN	0
EDP Renováveis	0
GREENVOLT	0
Iberdrola	0
Benfica	0
CORTICEIRA AMORIM	0
Unilever	0
Tesla	0

**Footer:**

valuing sustainable forests carbon stock **CO2OFFSET** [Go To](#)

[Dashboard](#)  
[Polls](#)  
[Transactions](#)  
[Stats](#)  
[About](#)

[Contact Us](#)

E-mail: [info@co2offset.ai](mailto:info@co2offset.ai)  
Phone: (+351) 919 508 768  
Website: [co2offset.ai](https://co2offset.ai)

[Follow Us](#)

X  
Instagram  
LinkedIn

Figura 14 - Design do Dashboard

Outra página que é possível ver no header é a página “Stats” que por razões semelhantes não foi mantida para desenvolvimento, esta página seria para ver estatísticas interessantes como o número de compradores, pools e créditos disponíveis e o número total de créditos vendidos, transações e pools, num determinado espaço de tempo.

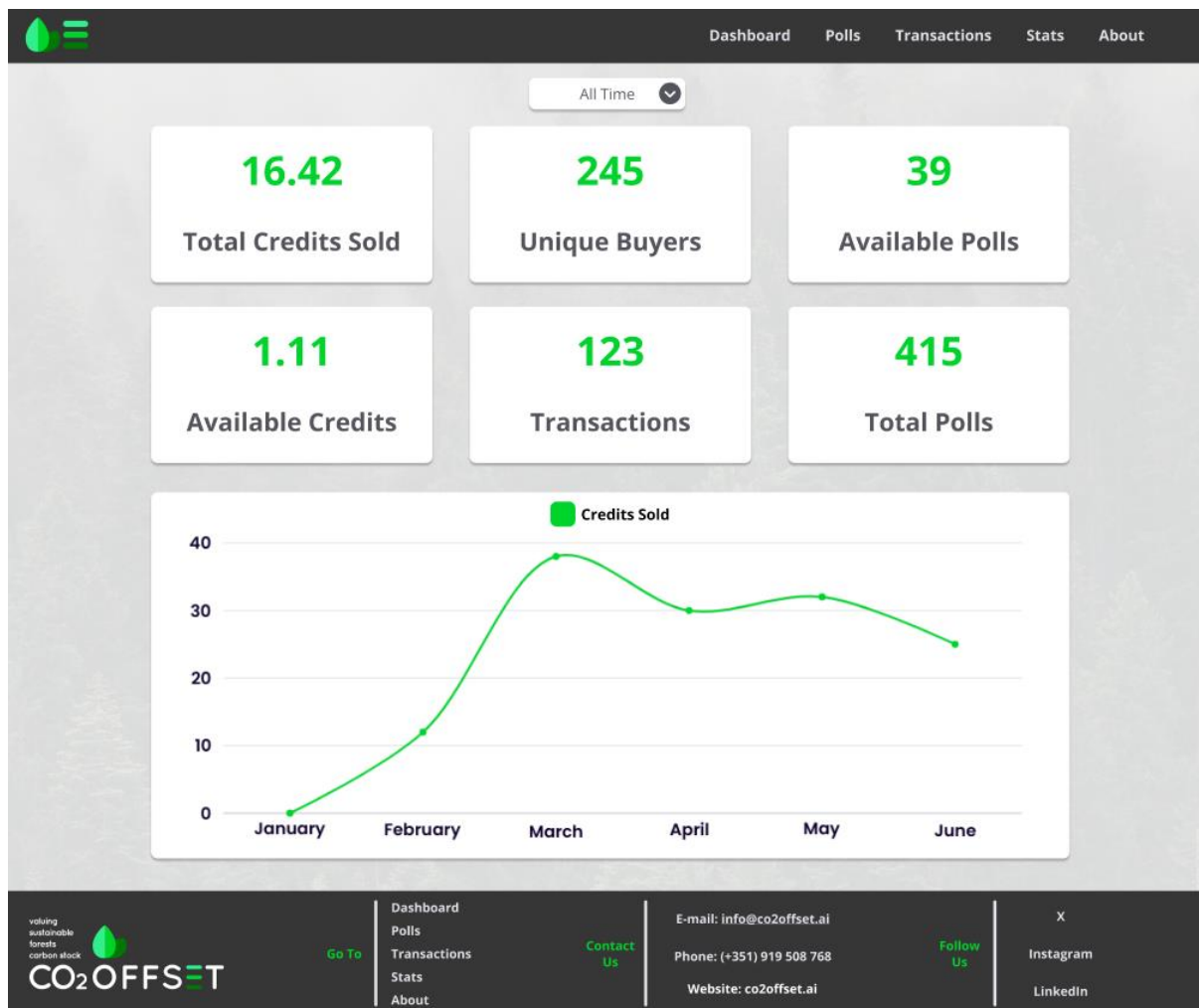


Figura 15 - Design da página Stats

Por último temos a página “About” onde estão algumas respostas a questões que um utilizador pode querer ver esclarecidas, sem necessidade de pesquisa ou contacto.



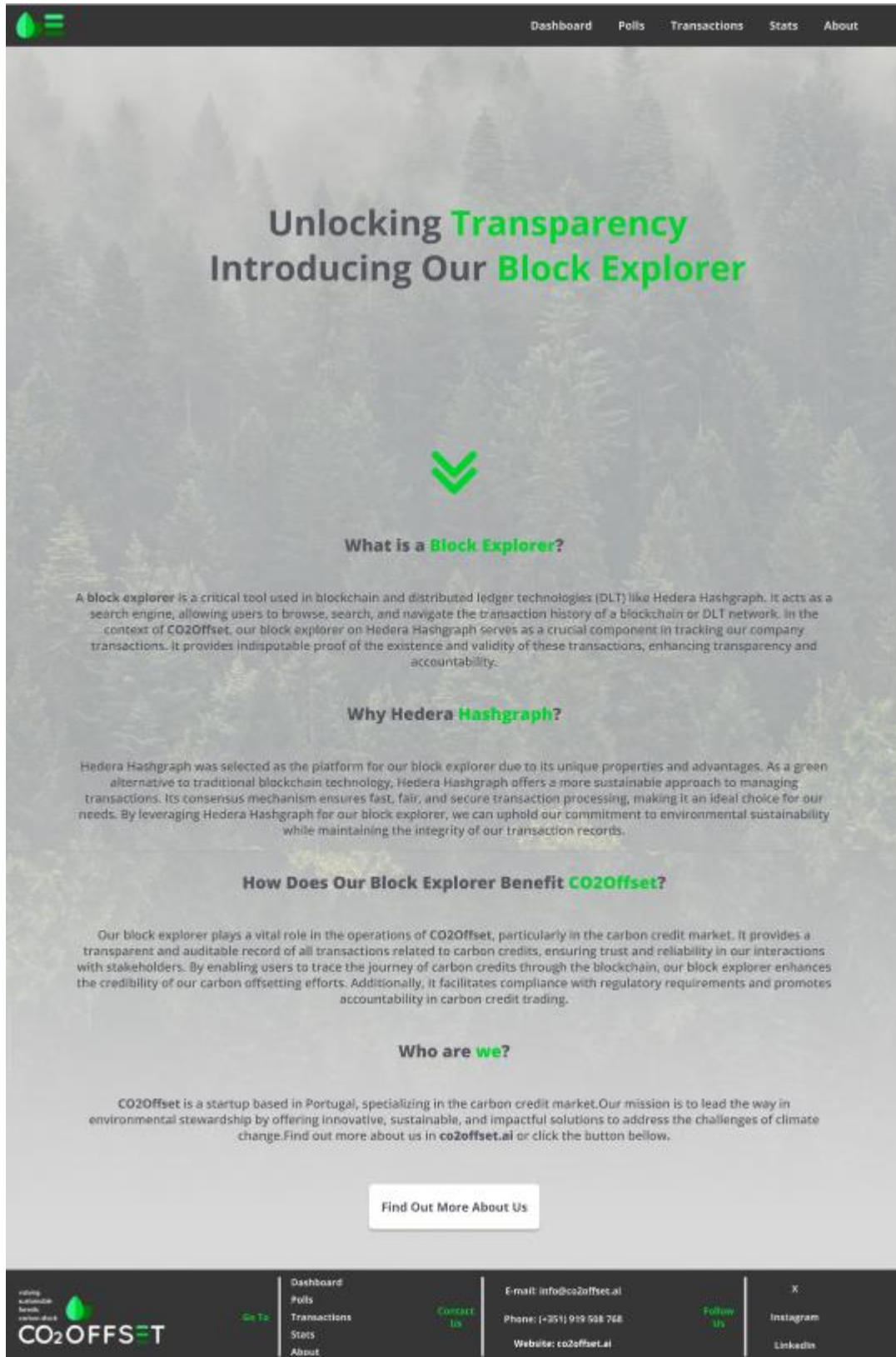


Figura 16 - Design da página About

É possível interagir com todas as páginas apresentadas pelo seguinte link:

- [Interagir com o design](#)

É também possível ver o esquema completo de todos os componentes e páginas implementados no **Anexo A**.

Concluindo este capítulo, o design do website foi cuidadosamente desenvolvido e refinado na plataforma Figma, respeitando os requisitos estabelecidos e as preferências de design. As páginas foram criadas com componentes reutilizáveis e estilos interativos, como efeitos de hover, para proporcionar uma experiência semelhante à real. A simplificação da página inicial e a reavaliação de algumas funcionalidades, baseadas no feedback recebido, garantiram um design final mais eficiente e alinhado com os objetivos do projeto.

Com o design aprovado, o próximo passo é a implementação técnica. No próximo capítulo, exploraremos como o design foi transformado em código funcional, detalhando o processo de desenvolvimento e a integração dos componentes projetados.

## **5.2 Construção do Website: Do Design à Realidade**

Com o design do website aprovado, o próximo passo foi transformar essas ideias visuais em código funcional. Para a implementação técnica, foi utilizada a framework React-TS em conjunto com o plugin Vite, que facilita um ambiente de desenvolvimento rápido e eficiente. Esta escolha foi feita devido à familiaridade dos desenvolvedores da CO<sub>2</sub>Offset com estas ferramentas, o que permite acelerar o processo de desenvolvimento posterior.

O React-TS (React com TypeScript) foi escolhido por oferecer uma robusta tipagem estática, que ajuda a evitar erros e a melhorar a qualidade do código, enquanto que o Vite proporciona um ambiente de desenvolvimento extremamente rápido, com tempos de construção rápidos e uma experiência de desenvolvimento mais fluida. Estas características são cruciais para um projeto que requer a visualização e interação constante com grandes conjuntos de dados, como é o caso do block explorer.

Além disso, o Chakra UI foi utilizado como biblioteca de componentes para garantir um design consistente e responsivo, facilitando a implementação dos modelos desenhados no Figma. O IDE escolhido para o desenvolvimento foi o Visual Studio Code (VSCode), que oferece um ambiente de desenvolvimento eficiente e uma vasta gama de extensões que aumentam a produtividade.

Neste capítulo, detalharemos o processo de implementação, desde a configuração inicial do projeto até a construção dos componentes utilizando Chakra UI, passando

pelos desafios encontrados e as soluções adotadas para superá-los. Este capítulo abordará também as melhores práticas seguidas durante o desenvolvimento e as ferramentas adicionais utilizadas para garantir a qualidade e performance do website.

### 5.2.1 Setup inicial

Para se começar a desenvolver o website é necessário preparar tudo para tal, portanto o primeiro passo será fazer a instalação de todas as ferramentas necessárias, foram seguidos os seguintes passos:

1. **Instalar o VSCode:** Download standard para Windows 11, [website](#).
2. **Postman:** Download standard para Windows, [website](#).
3. **Criar o projeto React-ts + Vite:** Abrir um terminal no VSCode e executar o comando: `npm create vite@latest omeuwebsite -- --template react-ts`, este comando cria um projeto padrão Vite + React-ts com um contador de cliques implementado.
4. **Criar o ficheiro install.sh:** Este é um script de instalação para preparar um servidor Linux (hospedado na cloud a partir dos AWS) para executar uma aplicação frontend. Ele realiza as seguintes tarefas principais:
  - Atualiza o software do sistema
  - Instala e configura o Docker e o Docker Compose
  - Cria uma pasta para o projeto
  - Instala o Git e configura-o para puxar um repositório específico
  - Instala e configura o Nginx
  - Instala o Certbot para certificados SSL
  - Executa um script de build (build.sh)
5. **Criar o ficheiro build.sh:** Este script é responsável por construir e executar a aplicação num container Docker. Este ficheiro faz o seguinte:
  - Solicita ao utilizador uma nova versão para a aplicação
  - Atualiza a versão no ficheiro package.json
  - Constrói uma nova imagem Docker com a versão especificada
  - Remove qualquer container existente com o mesmo nome
  - Inicia um novo container Docker com a imagem recém-construída

6. **Criar o ficheiro Dockerfile:** Este ficheiro define como a imagem Docker para a aplicação deve ser construída. Ele:
  - Copia os ficheiros do projeto para o container
  - Instala as dependências do projeto
  - Constrói a aplicação para produção
  - Define o comando para iniciar a aplicação
7. **Criar o ficheiro .dockerignore:** Este ficheiro especifica quais arquivos e diretorias devem ser ignorados ao construir a imagem Docker. Neste caso, ele ignora:
  - A pasta `node_modules` (que contém as dependências instaladas localmente)
  - A pasta `build` (que contém os arquivos de build gerados localmente)
8. **Criar o ficheiro .gitignore:** Este ficheiro é usado para especificar quais pastas e diretorias devem ser ignorados pelo Git, ou seja, não adicionados ao repositório.
  - Evita que arquivos desnecessários ou específicos do ambiente local sejam adicionados ao repositório Git.
  - Previne a partilha de informações sensíveis (como em arquivos `.env`).
  - Reduz o tamanho do repositório, excluindo arquivos grandes ou temporários (como `node_modules` e pastas de build).
  - Mantém o repositório limpo e focado apenas no código-fonte e arquivos essenciais do projeto.
9. **Criar o ficheiro .env:** Utilizado para armazenar variáveis de ambiente, As suas principais funções são:
  - Guardar informações sensíveis (chaves de API, palavras-passe) fora do código-fonte.
  - Não é versionado, sendo incluído no `.gitignore`.
  - Facilita a configuração do projeto para novos programadores.
  - É carregado automaticamente por muitas ferramentas de desenvolvimento.

Todos estes passos foram simples, mesmo a criação dos ficheiros, pois foram facultados ficheiros semelhantes relativos a outros projetos, sendo necessário apenas modificar o que lhes era específico e adaptar ao novo projeto.

Agora está tudo pronto para se começar a desenvolver o código da aplicação web, quando é necessário atualizar a versão development basta apenas executar o

comando: **npm run build** e o Vite irá preparar a aplicação web para ser publicada, se o comando for executado com sucesso, basta executar o **build.sh**.

## 5.2.2 Desenvolvimento do código

Este capítulo vai abordar como foram desenvolvidas as funcionalidades que o Block Explorer terá, estas funcionalidades e páginas são:

1. Estilização com componentes Chakra UI
2. Sistema de navegação
3. Sistema de tradução
4. Sistema de comunicação com a API

### Estilização com Chakra UI

Para utilizar o Chakra UI no nosso projeto, é necessário realizar algumas configurações iniciais. Primeiramente, devemos instalar o Chakra UI e as suas dependências através do npm. Para tal, é preciso executar o seguinte comando no terminal:

```
npm i @chakra-ui/react @emotion/react @emotion/styled framer-motion
```

Após a instalação, é crucial fazer o "wrap" da nossa aplicação com o ChakraProvider. Isto permite que todos os componentes do Chakra UI funcionem corretamente no projeto. Para tal, importamos o ChakraProvider no nosso ficheiro principal (geralmente o App.tsx ou main.tsx) e envolvemos a nossa aplicação com ele:

```
import { ChakraProvider } from '@chakra-ui/react'  
  
ReactDOM.createRoot(document.getElementById('root'))  
.render(  
  <React.StrictMode>  
    <ChakraProvider theme={theme} >  
      <App />  
    </ChakraProvider>  
  </React.StrictMode>,  
)
```

Com esta configuração, podemos agora utilizar todos os componentes e funcionalidades do Chakra UI ao longo do nosso projeto, para fazer o uso de um

componente como um Botão, numa determinada página, era necessário fazer o seguinte:

```
import { Button } from '@chakra-ui/react'

function Componente() {
  return (
    <Button colorScheme="green">Clique Aqui</Button>
  )
}
```

Este processo é semelhante para todos os outros componentes do Chakra UI, todos os componentes disponíveis podem ser vistos [aqui](#).

## Sistema de navegação

Para implementar o sistema de navegação entre as páginas no nosso Block Explorer, utilizou-se o React Router DOM. Primeiro, é necessário instalar esta biblioteca:

**npm install react-router-dom**

Após a instalação, importamos os componentes necessários do React Router DOM no nosso ficheiro principal, main.tsx:

```
import { BrowserRouter, createRoutesFromElements,
Route, RouterProvider } from 'react-router-dom';

const router = BrowserRouter(
  createRoutesFromElements(
    <Route>
      <Route path="/" element={<Home />} />
      <Route path="/transactions"
element={<Transactions Type="txn" />} />
      <Route path="/pools" element={<Pools />} />
      <Route path="/about" element={<About />} />
      <Route path="/Details" element={<Details />} />
      <Route path="*" element={<NotFoundPage />} />
    </Route>
  )
);
```

```
function App() {  
  return (  
    <ChakraProvider>  
      <RouterProvider router={router} />  
    </ChakraProvider>  
  );  
}
```

Aqui foram definidas as rotas, ou seja, as páginas que estão no escopo da aplicação, o “path” refere-se ao url para além do url base, se o url base fosse **www.url.com**, o **www.url.com/transactions** mostraria uma página cujo conteúdo estaria no componente “Transactions”, com passagem da variável “Type” com o valor “txn” para esse mesmo componente, o mesmo acontece para os outros caminhos (exceto a passagem de variáveis), sendo que se o utilizador colocar um url que não corresponda a nenhum dos caminhos definidos, ele verá a página cujo conteúdo se encontra no Componente “NotFoundPage”.

## Sistema de Tradução

Para implementar o sistema de tradução no nosso Block Explorer, utilizamos a biblioteca i18next. Começamos por instalar as dependências necessárias:

```
npm install i18next react-i18next i18next-browser-languagedetector
```

Após a instalação é necessário criar um ficheiro de configuração para o i18next, geralmente chamado i18n.ts, neste ficheiro é definido as linguagem suportadas pela aplicação web, neste caso será inglês, português e espanhol.

```
import i18next from "i18next";  
  
import LanguageDetector from "i18next-browser-  
languagedetector";  
  
import { initReactI18next } from "react-i18next";  
  
import translationEn from  
"./public/locales/en/translation.json";  
  
import translationEs from  
"./public/locales/es/translation.json";  
  
import translationPt from  
"./public/locales/pt/translation.json";
```

```
i18next
```

```
.use(LanguageDetector)
.use(initReactI18next)
.init({
  fallbackLng: "en",
  ns: ["default"],
  defaultNS: "default",
  supportedLngs: ["en", "es", "pt"],
  resources: {
    en: {
      default: translationEn,
    },
    es: {
      default: translationEs,
    },
    pt: {
      default: translationPt,
    },
  },
});
```

```
export default i18next;
```

Nas importações iniciais é possível verificar que estão a ser importados alguns ficheiros JSON, é nestes ficheiros que são guardados as keys e os respetivos valores, por exemplo para o ficheiro de tradução para português:

```
{
  "hello": "Olá Mundo"
}
```

Para que a aplicação se aperceba que tem de ir buscar esta informação ao ficheiro JSON é necessário usar o hook `useTranslation`, aqui está um exemplo que traduziria “hello” para “Olá Mundo” com recurso a esse ficheiro:



```
import { useTranslation } from 'react-i18next';

function MyComponent() {
  const [t] = useTranslation();

  return (
    <div>
      <h1>{t('hello')}</h1>
    </div>
  );
}
```

Neste exemplo, caso a linguagem estivesse definida para “pt” o texto apresentado seria “Olá Mundo”, para completar a funcionalidade de tradução vamos dar poder ao utilizador para alterar a linguagem de acordo com a sua preferência.

```
import { useTranslation } from 'react-i18next';
import { Select } from '@chakra-ui/react';

function LanguageSelector() {
  const { i18n } = useTranslation();

  const languages = [
    { value: "en", text: "English" },
    { value: "pt", text: "Portuguese" },
    { value: "es", text: "Spanish" },
  ];

  const handleChange = (e) => {
    i18n.changeLanguage(e.target.value);
  };

  return (
    <Select onChange={handleChange}>
```

```
{languages.map((lang) => (  
  <option key={lang.value} value={lang.value}>  
    {lang.text}  
  </option>  
))}  
</Select>  
);  
}
```

Neste exemplo é mostrado ao utilizador uma lista de opções entre inglês, português e espanhol, quando o utilizador seleciona a opção é efetuada a mudança da linguagem de toda a aplicação com recurso à função `changeLanguage` do Hook `useTranslation`, esta irá receber o valor (en,pt ou es) respetivo à escolha do utilizador, com esse conhecimento a `library i18n` sabe também em que ficheiro JSON se encontram os dados a usar para efetuar a tradução.

## Sistema de comunicação com a API

Para a comunicação com a API, utilizamos a biblioteca `Axios`. Primeiro, instalamos o `Axios`:

**npm install axios**

Em seguida, criamos uma instância do `Axios` com a URL base da nossa API, neste exemplo ela está a fazer a comunicação com a `Testnet Mirror Node API` da `Hedera`:

```
import axios from 'axios';  
  
const api = axios.create({  
  baseURL:  
  "https://testnet.mirrornode.hedera.com/api/v1",  
});
```

Apesar de neste exemplo estar a ser usada a API da `Hashgraph testnet`, no início do desenvolvimento deste projeto não foi usada esta API, foi usada uma API REST interna proporcionada pelo [Directus](#), por serem ambas REST API a lógica é a mesma, no entanto seria outro “baseURL”, sendo que a API do `Directus` retornava sempre dados em JSON e a da `Hedera` retorna o que quer que seja o conteúdo submetido na transação.

O uso inicial da API `Directus` e a passagem posterior para a API da `Hashgraph` gerou alguns problemas, a `CO2Offset` contratou uma empresa para desenvolver uma API que permitisse com um simples pedido HTTP registar uma transação na `Hashgraph`, este processo demorou mais do que esperado e o uso da `Hashgraph` era um passo

importante na resolução deste projeto, por estes motivos a troca de API foi dos últimos passos deste projeto, por isso houve a prevenção de problemas e foi tudo preparado para que a troca da API fosse tão simples como a troca do “BaseURL” do Directus para a Testnet Mirror Node, mas esta simplicidade esteve longe da realidade.

Após muito tempo de espera a empresa externa finalmente apresentou a sua solução uma API (e a sua documentação) protegida por um certificado digital que permite fazer pedidos POST e PUT, e, a partir desses pedidos registar automaticamente o conteúdo na Hashgraph, para que não bastasse o longo tempo de demora, o certificado que foi facultado estava protegido por uma private key encriptada, essa chave estava também protegida por uma palavra-passe, esta não nos foi facultado, pelo que foi necessário contactar novamente a empresa para que resolvesse a situação, depois de insistir bastante finalmente obtivemos uma resposta, que afirmava “o meu colega que trata disso está de férias”, com um certificado não funcional não seria possível integrar a Hashgraph no projeto, isto exigiu a que tivéssemos de insistir mais com esta empresa até que finalmente nos forneceram um certificado funcional.

Com o auxílio do Postman foram feitos alguns pedidos de acordo com a documentação facultada pela empresa sobre o uso da API, no entanto estes pedidos não tiveram sucesso, portanto isto significava que para além dos outros constrangimentos a documentação parece estar incorreta, felizmente não foi necessário contactar a empresa para resolver o problema, e foi possível identificar e resolver o problema, tratava-se de alguns campos com a denominação incorreta, na documentação constava “reference” porém a denominação correta era “referenceTransactionId”.

Depois de resolver estes problemas foi finalmente possível fazer um pedido à API com sucesso, basta apenas que este se traduza numa transação visível na rede Testnet da Hedera, o que se confirma, agora que temos transações registadas na rede é necessário obter esses dados para que possamos mostrar ao utilizador no Block Explorer.

Para se entender mais facilmente como obter os dados, é necessário perceber onde eles estão e o que eles significam, a API desenvolvida pela empresa permite:

- Criar um Tópico, que é onde são registadas mensagens na rede, irá ser criado um tópico para cada empresa, e as mensagens nesse tópico corresponderão a transações efetuadas por essa empresa.
- Criar uma transação num determinado tópico
- Criar uma pool, todas as pools ficam registadas também como mensagens todas sob o mesmo tópico

Sendo que precisamos apenas de obter as transações e as pools, a obtenção dos dados foi feita da seguinte forma, a CO<sub>2</sub>Offset tem uma conta na rede, todas as

contas têm um identificador único, portanto para obter as transações podemos obter todos os tópicos criados por uma conta específica, e depois obter todas as mensagens de cada um desses tópicos, para as pools, como estão todas sob o mesmo tópico basta só obter as mensagens desse tópico específico, serão necessários então 3 pedidos/funções diferentes:

1. Obter as mensagens de um tópico específico:

```
private async getMessagesFromSingleTopic(topicId: string): Promise<any[]> {  
    const response = await  
    this.api.get(`/topics/${topicId}/messages`);  
}
```

Esta função recebe o id de um tópico e devolve todas as mensagens sob esse tópico

2. Obter as pools

```
getMessagesFromSingleTopic("id do tópico das pools")
```

Aqui é usada a função referida anteriormente, no entanto tem de ser passado o id do tópico correto onde se encontram as transações referentes às pools.

3. Obter tópicos de uma payer account

```
const response = await  
this.api.get("/transactions", {  
    params: {  
        "account.id": "id da conta",  
        transactiontype:  
        "ConsensusCreateTopic",  
    },  
})
```

Este pedido devolve todos os tópicos criados por uma conta específica em formato de Array, usando novamente a função `getMessagesFromSingleTopic()` são recebidas todas as mensagens de cada um desses tópicos, essas mensagens são agregadas e devolvidas no formato desejado.

Aqui está um exemplo de os dados devolvidos por um pedido para obter as mensagens das pools (esta resposta é semelhante à obtenção das transações com exceção do conteúdo da mensagem):

```
"messages": [
```

```

{
  "chunk_info": null,
  "consensus_timestamp":
"1717433682.000635003",
  "message":
"VGvYcmFpbiBQcm9wZXJ0aWVzIC0gJ0FjdGlvbiBEYXRlJzogJzIw
MjItMDMtMTBUMTY6MTU6NTBaJzsgJ0FjdGlvbic6ICdOL0QnOyAnT
GF0aXR1ZGUsIExvbmdpdHVkZSc6ICg5MC4wMDAwMDAsMTgwLjAwMD
AwMCK7ICdTcGVjaWVzJzogJ0phcGFuZXNlIGtub3R3ZWVkJzsgJ1R
vdGFsIENyZWVpdCc6IDMyLjAwMDAwMDsgJ0NPMiBTdG9jayBUcmFj
a2VkZzogMS4xMDAwMDA7ICdDTzIgU3RvY2sgQWRkaXRpb25hbGx5J
zogMi4zMDAwMDA7ICdDTzIgU3RvY2sgQWRkaXRpb25hbGx5IE1MIE
JpYXM6IDAuODAwMDAwOyAnQ08yIEFN0b2NrIEFkZGl0aW9uYWxseSB
NTFIyJzogMC4xMDAwMDA7ICdDZXJ0aWZpY2F0aW9uJzogJ04vQSc7
ICdSZWZlcmVuY2UnOiAxOyAnTW90aXZlOiAnTi9EJw==",
  "payer_account_id": "0.0.4276983",
  "running_hash":
"w+4S9mjKcMSH+iB8cei4WZf2rzSvkhYSFYFkgMU9n5Ted07dfgJ8
4QeJYnpd+lTM",
  "running_hash_version": 3,
  "sequence_number": 1,
  "topic_id": "0.0.4351110"
},
...
]

```

Como é possível verificar, um humano não é capaz de extrair informações sobre a pool a partir da resposta da API, isto deve-se ao facto do conteúdo das mensagens na rede Hashgraph serem codificadas em Base64 e UTF-8, para decodificar a mensagem basta apenas usar a função **atob()** do JavaScript (e no typescript) e passando a string da mensagem por argumento.

Ao realizar o processo de decodificação da mensagem obtemos a resposta seguinte para cada uma das funções:

1. **Obter tópicos:** O pedido que permite obter os tópicos de uma determinada conta da hashgraph devolve um Array de tópicos (no formato standard da Hedera Hashgraph)
2. **Obter pools:** Ao decodificar a mensagem retornada, obtemos, por exemplo o seguinte (os dados são apenas para fins de visualização):

```
Terrain Properties - 'Action Date': '2022-03-10T16:15:50Z'; 'Action': 'N/D'; 'Latitude, Longitude': (90.000000,180.000000); 'Species': 'Japanese knotweed'; 'Total Credit': 32.000000; 'CO2 Stock Tracked': 1.100000; 'CO2 Stock Additionally': 2.300000; 'CO2 Stock Additionally ML Bias': 0.800000; 'CO2 Stock Additionally MLR2': 0.100000; 'Certification': 'N/A'; 'Reference': 1; 'Motive': 'N/D'
```

Esta mensagem não está em formato JSON, porém aparenta estar noutro formato normalizado em que excluindo “Terrain Properties – “ parece estar a seguir a lógica: **‘chave’:‘valor’**; ,mas observando mais atentamente é possível verificar a falta de formatação consistente, o último campo não acaba em “;”, e **“CO2 Stock Additionally ML Bias”** não está corretamente encapsulado por “”, isto é um problema causado pela lógica de backend da API desenvolvida pela empresa externa, esta falta de uma formatação correta e consistente torna a automatização e utilização dos dados um problema.

Para resolver este problema existem 2 soluções óbvias:

- a. Formatar os dados para um formato desejado JSON de forma automatizada após receber os dados do pedido.
- b. Contactar a empresa que desenvolveu a API para que use uma formatação consistente, de preferência JSON.

A solução adotada foi a 1., apesar da solução 2. ser favorável.

1. **Obter transações:** Segue a mesma lógica de obtenção das pools, a formatação desta também apresenta erros de formatação e erros, no entanto são erros diferentes que a anterior, portanto a formatação desta mensagem para JSON não segue a mesma lógica que a outra.

Com estes problemas foi necessário desenvolver as funções:

**poolMessageFormatter(mensagem)**

**txnMessageFormatter(mensagem)**

Ambas recebem a mensagem decodificada como argumento e devolvem a mensagem formatada em JSON, idealmente a API que envia as mensagens para a Hashgraph será atualizada no futuro para enviar as mensagens formatadas em JSON, caso isso aconteça estas funções deixam de ser necessárias.

Após concluir todos estes passos e resolver estes problemas, temos uma comunicação com a API que nos devolve o conteúdo desejado para cada transação decodificado e formatado em JSON, isto permite usar os dados de forma simples para que possam ser mostrados ao utilizador.

### 5.2.3 Resultados finais

Todas as páginas foram desenvolvidas com componentes Chakra UI e em react-ts. Aqui serão mostrados os resultados finais das páginas. Antes de mostrar cada página, iremos explicar a sua funcionalidade.

#### 1. Componente Header

O componente Header permite navegar entre as páginas /transactions, /pools e /about. Ele também permite alterar a linguagem do website e navegar para a página inicial clicando no logotipo, nesta imagem encontramos na página /transactions.



Figura 17 - Resultado final do Header

#### 2. Componente Footer

O componente Footer permite navegar para as páginas /transactions, /pools e /about. Ele contém informações de contacto da CO2Offset, incluindo email, telemóvel, website, e um aviso legal.

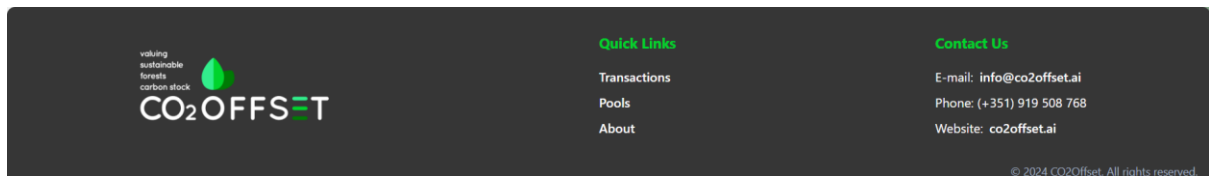


Figura 18 - Resultado final do Footer

#### 3. Página Inicial (/)

A página inicial permite realizar pesquisas de transações por determinados parâmetros ou "pesquisar" todas as transações. A pesquisa pode ser feita também por data através de um calendário interativo. Caso a pesquisa tenha resultados, o utilizador é levado para a página de transações (/transactions), é possível também alterar a linguagem.



Figura 19 - Resultado final da Página Inicial

#### 4. Página de Transações (/transactions)

Esta página permite visualizar numa tabela as transações de acordo com os critérios definidos na página inicial ou mostrar todas as transações se acedida diretamente. Permite filtrar os resultados, remover filtros e fazer novas pesquisas. As transações são divididas por páginas de acordo com o tamanho da janela, permitindo alternar entre páginas e visualizar os detalhes, levando o utilizador para a página /Details?Transaction&Id=”id da transação”. Além disso, ao escrever na barra de filtragem e na barra de pesquisa, o utilizador recebe sugestões de acordo com o parâmetro selecionado, os dados disponíveis e a sua query.

ORGANIZAÇÃO	TIPO DE CRÉDITO	TON/ANO	DATA DE INÍCIO	DATA DE FIM	DETALHES
CO2Offset	CO2Offset Credit	82	10/03/2024	10/03/2025	<a href="#">[Link]</a>
CO2Offset	Sequoia Rise	82	10/03/2024	10/03/2025	<a href="#">[Link]</a>
CO2Offset	CO2Offset Credit	82	10/03/2024	10/03/2025	<a href="#">[Link]</a>
CO2Offset	CO2Offset Credit	82	10/03/2024	10/03/2025	<a href="#">[Link]</a>
VOID Software	CO2Offset	33	10/03/2024	10/03/2025	<a href="#">[Link]</a>

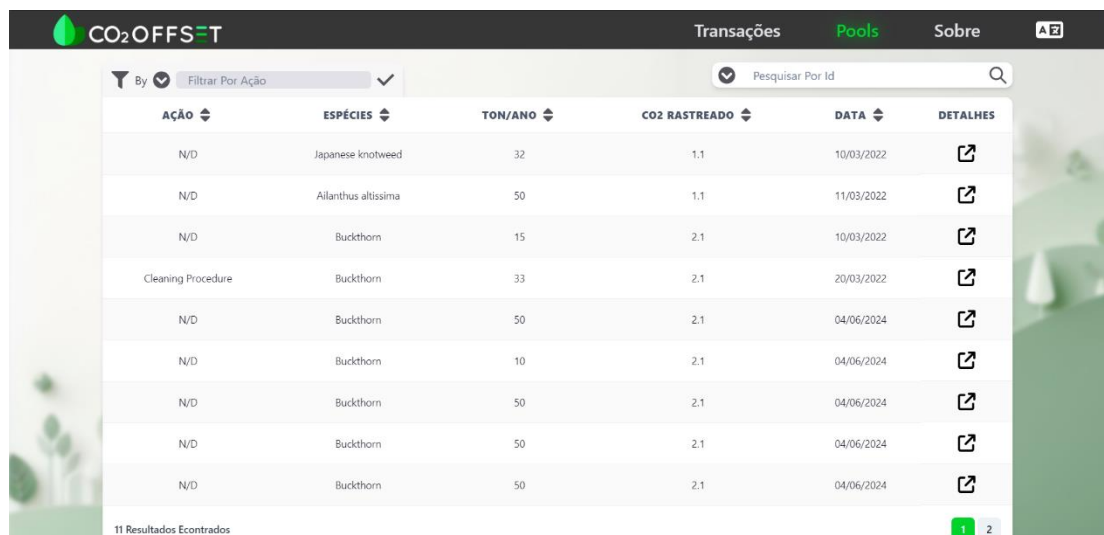
5 Resultados Encontrados

Figura 20 - Resultado final da página transactions

#### 5. Página de Pools (/pools)



Esta página utiliza o mesmo componente que a página de transações, adaptado conforme o link, seja /pools ou /transactions. Portanto, permite as mesmas funcionalidades: pesquisar, filtrar, etc., sendo que os detalhes levam para a página /Details?Pool&Id=id da respetiva pool. Assim como na página de transações, ao escrever na barra de filtragem e na barra de pesquisa, o utilizador recebe sugestões.



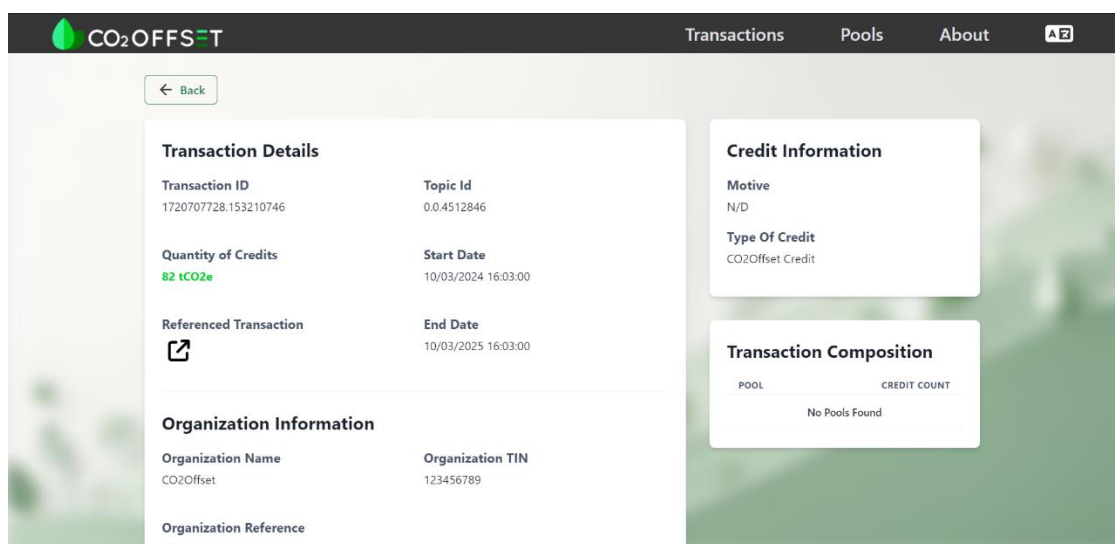
The screenshot shows the CO2 OFFSET website interface. At the top, there is a navigation bar with the logo and tabs for 'Transações', 'Pools', and 'Sobre'. Below the navigation bar, there is a search bar with the placeholder 'Pesquisar Por Id' and a filter dropdown set to 'Filtrar Por Ação'. The main content area displays a table with the following columns: 'AÇÃO', 'ESPÉCIES', 'TON/ANO', 'CO2 RASTREADO', 'DATA', and 'DETALHES'. The table contains 11 rows of data, including entries for 'Japanese knotweed', 'Ailanthus altissima', 'Buckthorn', and 'Cleaning Procedure'. At the bottom of the table, it indicates '11 Resultados Encontrados'.

AÇÃO	ESPÉCIES	TON/ANO	CO2 RASTREADO	DATA	DETALHES
N/D	Japanese knotweed	32	1.1	10/03/2022	
N/D	Ailanthus altissima	50	1.1	11/03/2022	
N/D	Buckthorn	15	2.1	10/03/2022	
Cleaning Procedure	Buckthorn	33	2.1	20/03/2022	
N/D	Buckthorn	50	2.1	04/06/2024	
N/D	Buckthorn	10	2.1	04/06/2024	
N/D	Buckthorn	50	2.1	04/06/2024	
N/D	Buckthorn	50	2.1	04/06/2024	
N/D	Buckthorn	50	2.1	04/06/2024	
N/D	Buckthorn	50	2.1	04/06/2024	

Figura 21 - Resultado final da página pools

## 6. Página de Detalhes de Transação (/Details?Transaction&Id=)

Esta página permite ver todos os detalhes da transação, pois muitos não eram visíveis na tabela. Permite também ver de quais pools são oriundos os créditos dessa transação e navegar para os detalhes dessas pools.



The screenshot shows the 'Transaction Details' page on the CO2 OFFSET website. The page is divided into several sections: 'Transaction Details', 'Credit Information', 'Transaction Composition', and 'Organization Information'. The 'Transaction Details' section includes fields for 'Transaction ID', 'Topic Id', 'Quantity of Credits', 'Start Date', 'Referenced Transaction', and 'End Date'. The 'Credit Information' section includes 'Motive' and 'Type Of Credit'. The 'Transaction Composition' section shows a table with 'POOL' and 'CREDIT COUNT' columns, indicating 'No Pools Found'. The 'Organization Information' section includes 'Organization Name', 'Organization TIN', and 'Organization Reference'.

Transaction Details	
Transaction ID	1720707728.153210746
Topic Id	0.0.4512846
Quantity of Credits	82 tCO2e
Start Date	10/03/2024 16:03:00
Referenced Transaction	
End Date	10/03/2025 16:03:00

Credit Information	
Motive	N/D
Type Of Credit	CO2Offset Credit

Transaction Composition	
POOL	CREDIT COUNT
No Pools Found	

Organization Information	
Organization Name	CO2Offset
Organization TIN	123456789
Organization Reference	CO2 123

Figura 22 - Resultado final da página details?Transactions

## 7. Página de Detalhes de Pool

## Block Explorer para a Tecnologia DLT Hashgraph

Esta página tem uma finalidade semelhante à de detalhes de transação. Permite ver todos os detalhes da pool, bem como uma lista de transações realizadas com recurso a essa pool. Também se pode navegar para os detalhes de cada transação e visualizar a certificação dessa pool.

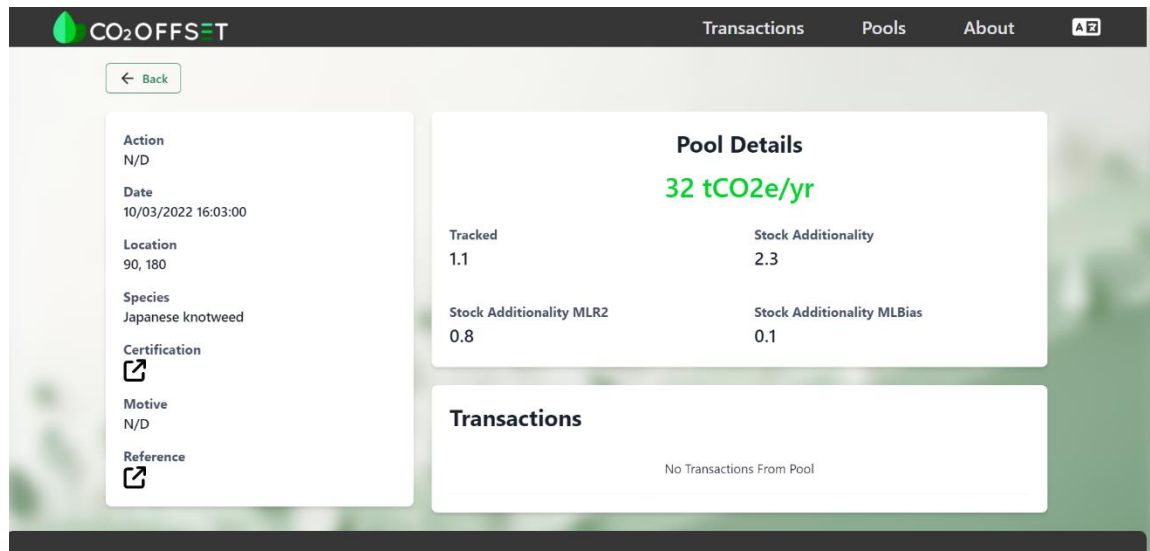


Figura 23 - Resultado final da página details?Pool

## 8. Página Sobre

Nesta página, são respondidas algumas questões que possam surgir ao utilizador:

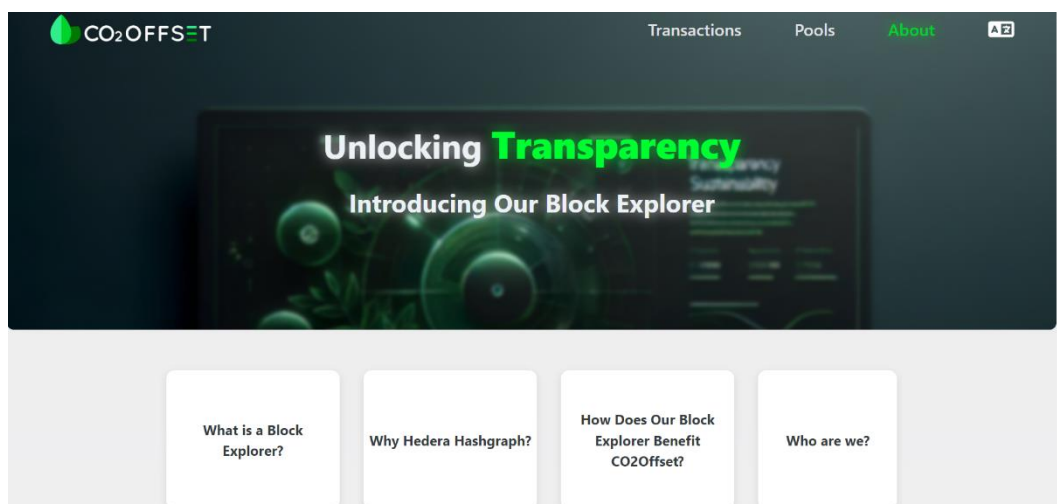


Figura 24 - Resultado final da página Sobre

## 9. Página NotFound

Se o utilizador tentar aceder a um caminho/página com um url não válido, desde que o domínio seja o correto, irá ver a seguinte página:



Figura 25 - Resultado final de PageNotFound



## **6 CONCLUSÕES E APRECIÇÕES FINAIS**

Concluindo o meu estágio na CO<sub>2</sub>Offset, sinto-me profundamente realizado com as experiências e aprendizagens adquiridas. Este período revelou-se uma oportunidade valiosa para desenvolver e aprofundar competências técnicas e pessoais, especialmente no que diz respeito ao funcionamento da Hashgraph, uma tecnologia inovadora e sustentável que se mostrou uma excelente alternativa às tradicionais blockchains.

Além disso, melhorei significativamente as minhas habilidades no desenvolvimento front-end, tendo a oportunidade de trabalhar com Vite e React-TS e explorar o design de interfaces através do Figma. A experiência de enfrentar problemas reais e trabalhar num ambiente empresarial foi enriquecedora e essencial para aliviar a ansiedade relativa ao mundo do trabalho.

Uma surpresa positiva foi ter de me envolver também no trabalho de design. Embora não esperasse desempenhar esse papel, foi uma experiência gratificante que ampliou a minha visão e competências.

Estou particularmente satisfeito com o resultado final do projeto, que superou as expectativas iniciais. O meu supervisor garantiu-me que o trabalho realizado será aproveitado e que, embora o estágio tenha sido apenas o primeiro passo, o projeto continuará a evoluir, com ou sem a minha presença. Fico feliz e curioso em saber como o que comecei se irá desenvolver no futuro, com a certeza de que o impacto do meu trabalho perdurará, quer seja visível ou não.

Esta experiência foi uma excelente preparação para a minha carreira, demonstrando a importância dos estágios curriculares na transição para o mercado de trabalho e na aplicação prática dos conhecimentos adquiridos. Agradeço pela oportunidade e estou entusiasmado para ver o futuro do projeto que iniciei.



## 7 REFERÊNCIAS

- [1] "Dashboard do Chainspect", *Chainspect*. [Online]. Disponível em: <https://chainspect.app/dashboard>. [Acesso em: 10 Jul. 2024].
- [2] "Consumo de Energia do Bitcoin", *Digiconomist*. [Online]. Disponível em: <https://digiconomist.net/bitcoin-energy-consumption>. [Acesso em: 13 Jul. 2024].
- [3] "Documentação da Hedera", *Hedera*. [Online]. Disponível em: <https://docs.hedera.com/hedera>. [Acesso em: 19 Jul. 2024].
- [4] "Página Inicial do Bitcoin", *Bitcoin*. [Online]. Disponível em: <https://bitcoin.org/en/>. [Acesso em: 11 Jul. 2024].
- [5] "Página Inicial do Ethereum", *Ethereum*. [Online]. Disponível em: <https://ethereum.org/pt/>. [Acesso em: 15 Jul. 2024].
- [6] "Documentação do Directus", *Directus*. [Online]. Disponível em: <https://docs.directus.io/>. [Acesso em: 14 Jul. 2024].
- [7] "Whitepaper da Hedera", *Crebaco*. [Online]. Disponível em: <https://crebaco.com/planner/admin/uploads/whitepapers/hedera-whitepaper.pdf>. [Acesso em: 16 Jul. 2024].
- [8] "O Que é Bit Gold?", *Investopedia*. [Online]. Disponível em: <https://www.investopedia.com/terms/b/bit-gold.asp>. [Acesso em: 12 Jul. 2024].
- [9] M. Rauchs et al., "Distributed Ledger Technology Systems: A Conceptual Framework," *Cambridge Judge Business School*. [Online]. Disponível em: <https://www.jbs.cam.ac.uk/wp-content/uploads/2020/08/2018-10-26-conceptualising-dlt-systems.pdf>. [Acesso em: 17 Jul. 2024].
- [10] "Hashgraph Consensus Algorithm Explained | Dr. Leemon Baird", *YouTube*. [Online]. Disponível em: <https://www.youtube.com/watch?v=cje1vuVKhwY>. [Acesso em: 19 Jul. 2024].
- [11] "Documentação do Vite", *Vite*. [Online]. Disponível em: <https://vitejs.dev/>. [Acesso em: 12 Jul. 2024].
- [12] "Introdução ao React", *React*. [Online]. Disponível em: <https://react.dev/>. [Acesso em: 15 Jul. 2024].
- [13] "Modelo de Branching no Git", *nvie*. [Online]. Disponível em: <https://nvie.com/posts/a-successful-git-branching-model/>. [Acesso em: 13 Jul. 2024].

- [14] "Guia Definitivo para Compreender Créditos de Carbono", *Carbon Credits*. [Online]. Disponível em: <https://carboncredits.com/the-ultimate-guide-to-understanding-carbon-credits/>. [Acesso em: 11 Jul. 2024].
- [15] "Relatório de Uso de Energia da Solana - Dezembro 2023", *Solana*. [Online]. Disponível em: <https://solana.com/pt/news/solana-energy-use-report-december-2023>. [Acesso em: 16 Jul. 2024].
- [16] "Introdução ao Chakra UI", *Chakra UI*. [Online]. Disponível em: <https://v2.chakra-ui.com/getting-started>. [Acesso em: 14 Jul. 2024].
- [17] "Página Inicial do npm", *npm*. [Online]. Disponível em: <https://www.npmjs.com/>. [Acesso em: 10 Jul. 2024].
- [18] "Whitepaper do Bitcoin", *Bitcoin*. [Online]. Disponível em: <https://bitcoin.org/bitcoin.pdf>. [Acesso em: 14 Jul. 2024].
- [19] "Página Principal do ISEC", *Instituto Superior de Engenharia de Coimbra (ISEC)*. [Online]. Disponível em: <https://www.isec.pt/PT/Default.aspx>. [Acesso em: 18 Jul. 2024].
- [20] "Página Inicial da CO<sub>2</sub>Offset", *CO<sub>2</sub>Offset*. [Online]. Disponível em: <https://co2offset.ai/>. [Acesso em: 17 Jul. 2024].
- [21] "GED alarga carteira em Portugal com compra de 23% da CO<sub>2</sub>Offset e 18% da Assetflood", *Jornal de Negócios*. [Online]. Disponível em: <https://www.jornaldenegocios.pt/empresas/detalhe/ged-alarga-carreira-em-portugal-com-compra-de-23-da-co2offset-e-18-da-assetflood>. [Acesso em: 19 Jul. 2024].
- [22] "Distributed Ledgers", *Fintech Latvia*. [Online]. Disponível em: [https://fintechlatvia.eu/wp-content/uploads/2023/11/Distributed-Ledgers\\_en.png](https://fintechlatvia.eu/wp-content/uploads/2023/11/Distributed-Ledgers_en.png). [Acesso em: 14 Jul. 2024].



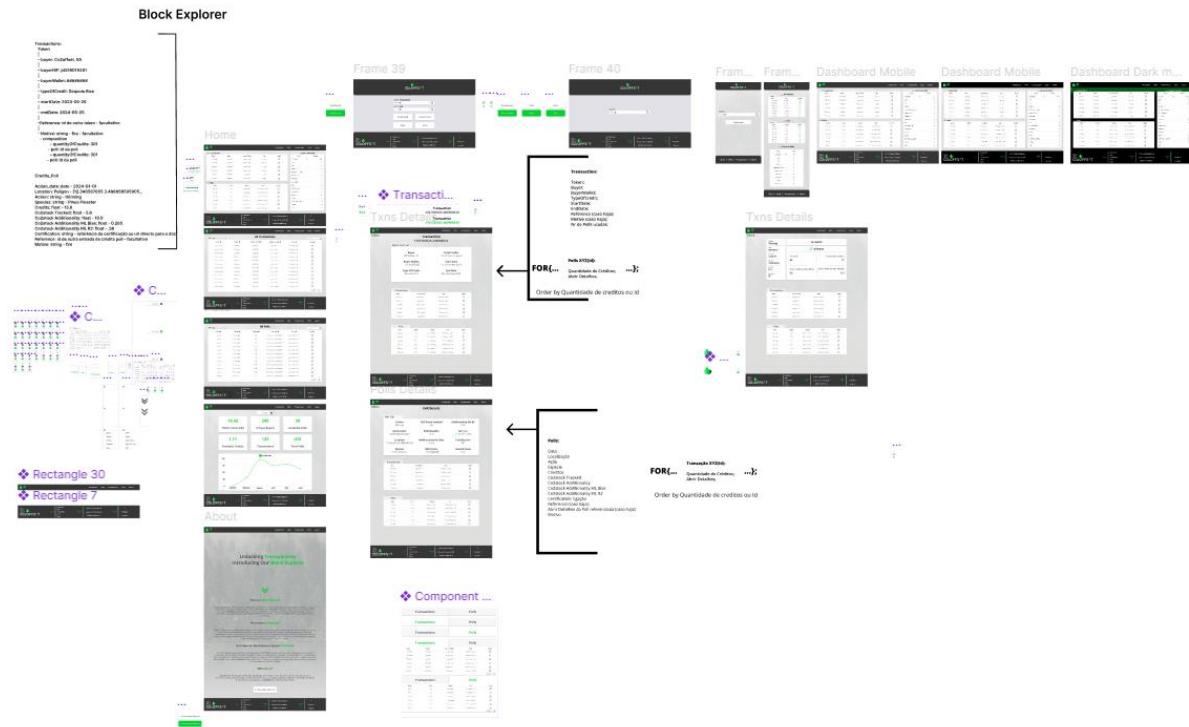




## 8 ANEXOS

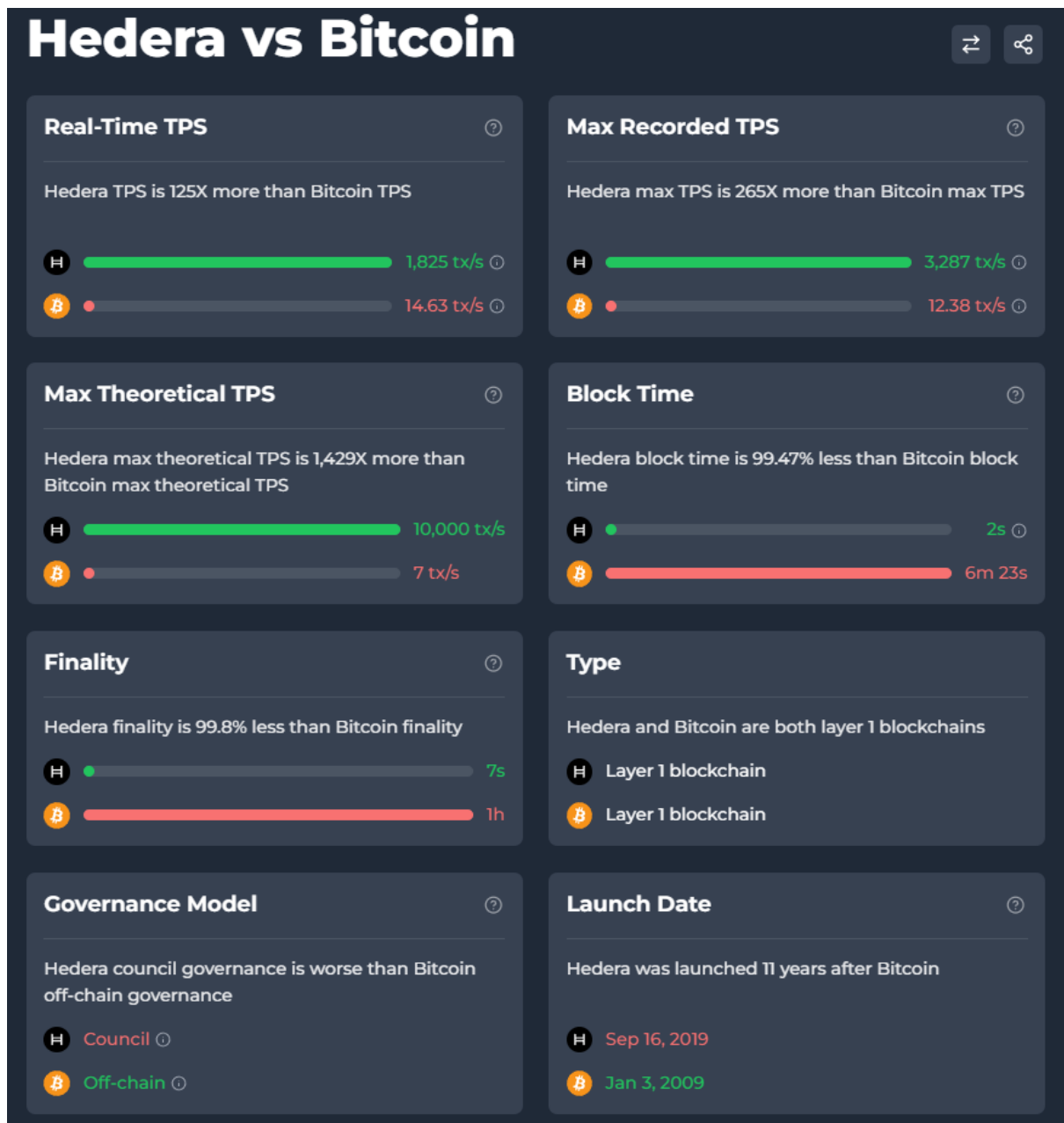
### Anexo A

Neste anexo, está uma visão geral de todo o design realizado no Figma, é possível interagir com ele aqui.



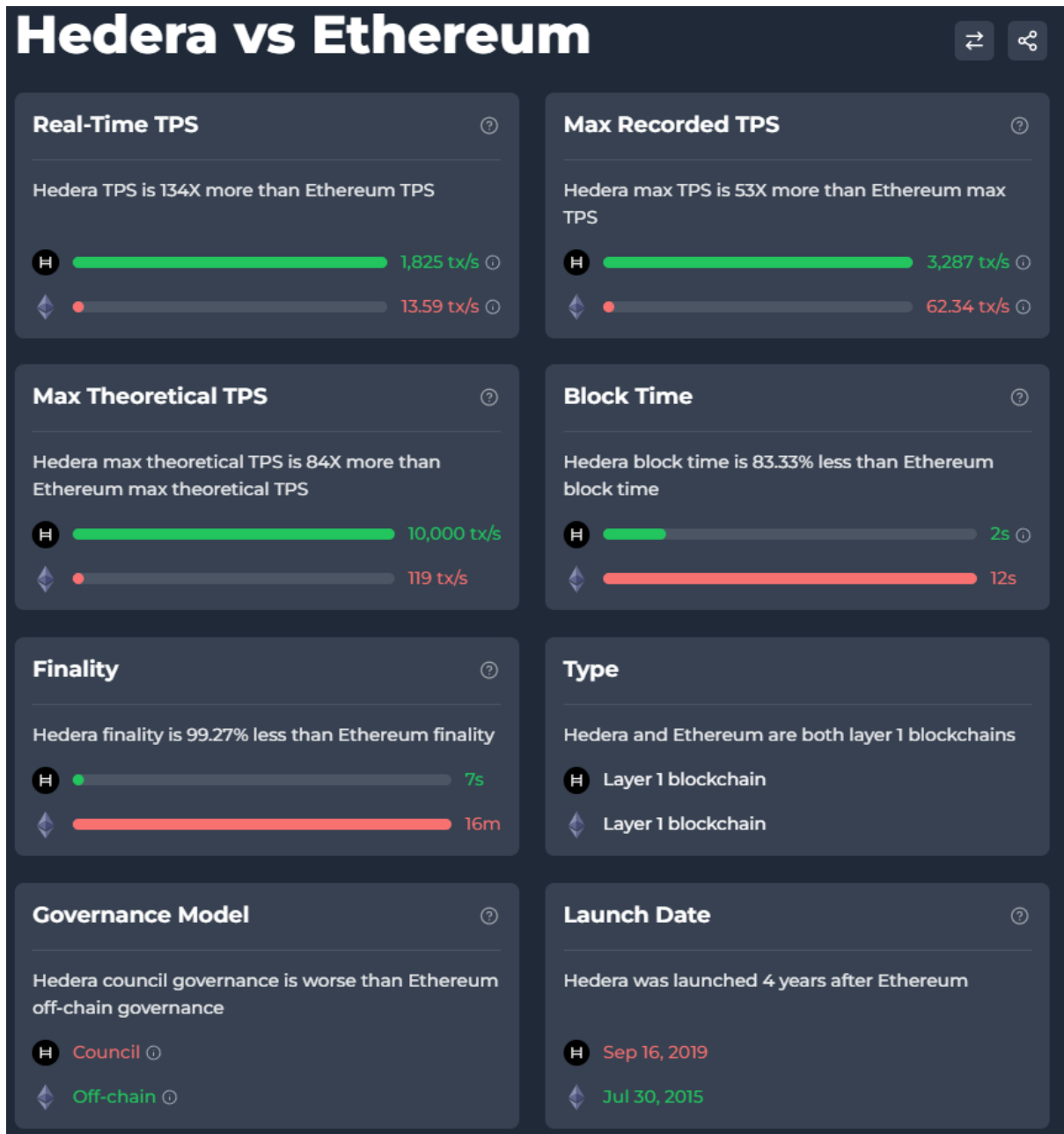
### Anexo B

Comparação entre a rede Hedera Hashgraph e Bitcoin.



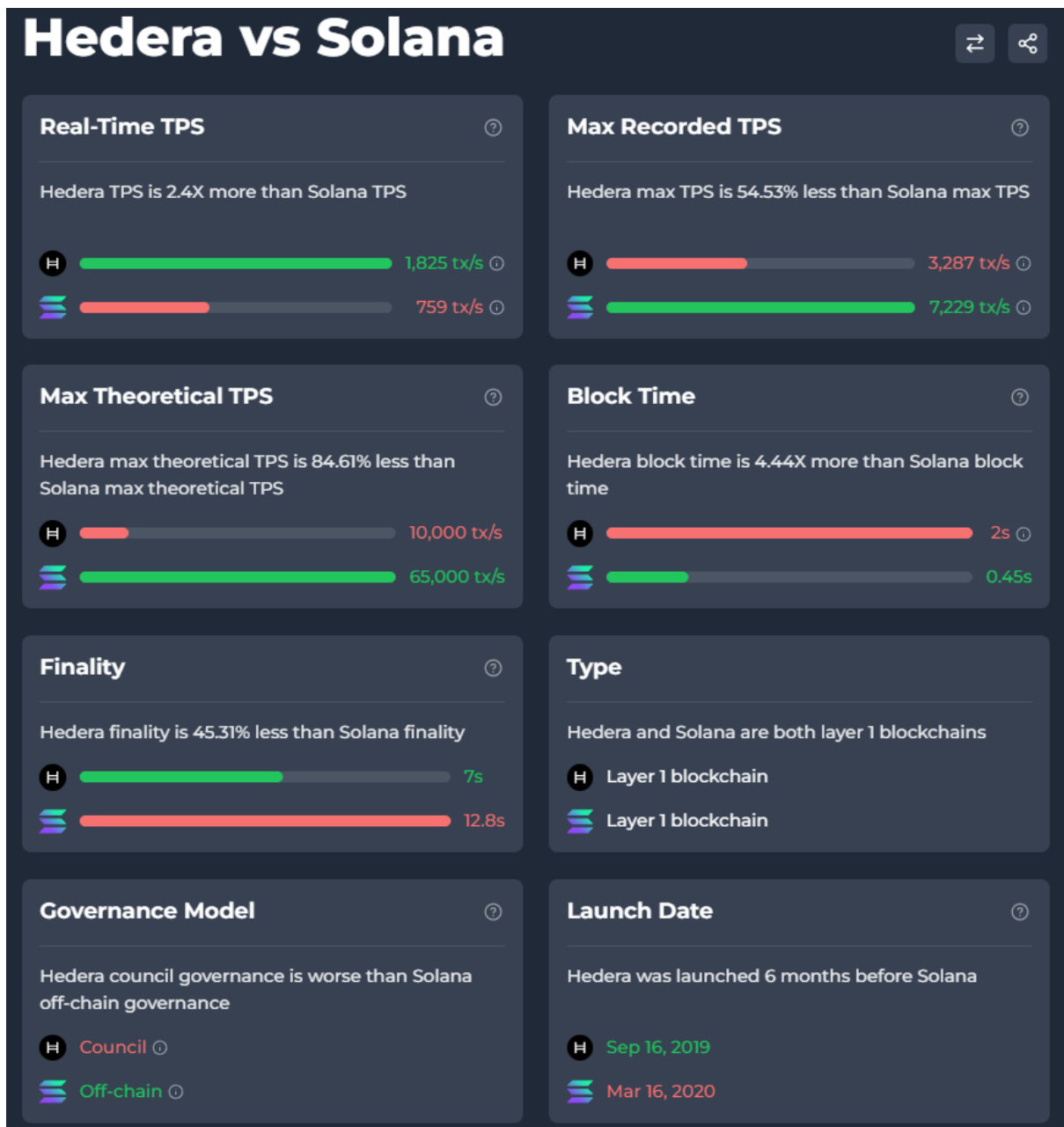
## Anexo C

Comparação entre a rede Hedera Hashgraph e Ethereum.



## Anexo D

Comparação entre a rede Hedera Hashgraph e Solana.





**Instituto Superior  
de Engenharia**

Politécnico de Coimbra