UNIVERSIDADE DO SUL DE SANTA CATARINA

IAN KRESSIN GUIMARÃES

DOLLAR-COST AVERAGING NA BLOCKCHAIN:

UMA ESTRATÉGIA DE INVESTIMENTO DESCENTRALIZADA

Florianópolis

2023

IAN KRESSIN GUIMARÃES

DOLLAR-COST AVERAGING NA BLOCKCHAIN:
UMA ESTRATÉGIA DE INVESTIMENTO DESCENTRALIZADA

Trabalho de Conclusão de Curso apresentado ao Curso de Sistemas de Informação da Universidade do Sul de Santa Catarina como requisito parcial à obtenção do título de Bacharel em Sistemas de Informação.

Orientador: Prof. Flávio Ceci, Dr.

Florianópolis

2023

IAN KRESSIN GUIMARÃES

DOLLAR-COST AVERAGING NA BLOCKCHAIN:

UMA ESTRATÉGIA DE INVESTIMENTO DESCENTRALIZADA

Este Trabalho de Conclusão de Curso foi julgado adequado à obtenção do título de Bacharel em Sistema de Informação e aprovado em sua forma final pelo Curso de Sistemas de Informação da Universidade do Sul de Santa Catarina.

Florianópolis, 13 de junho de 2023.

_____

Professor e orientador Flávio Ceci, Dr.
Universidade do Sul de Santa Catarina

_____

Prof. Aran Bey Tcholakian Morales, Dr.
Universidade do Sul de Santa Catarina

_____

Prof. Saulo Popov Zambiasi, Dr./Ms.
Universidade do Sul de Santa Catarina

Este trabalho é todo dedicado aos meus pais, pois é graças ao seu esforço que hoje posso concluir o meu curso.

AGRADECIMENTOS

Começo agradecendo àqueles que são os pilares da minha vida, minha família, Roberto, Andrea e Sabrina, que investiu nos meus estudos e depositou confiança em mim durante toda a minha caminhada acadêmica.

Agradeço a minha namorada e melhor amiga, Ana, sem suas palavras de incentivo seria bem mais difícil chegar ao fim deste trabalho.

Por último, mas não menos importante, gostaria de agradecer a todos os professores que contribuíram para a minha formação até então. Em especial, gostaria de agradecer ao professor Flávio, meu orientador, que compartilhou de forma honrável seus conhecimentos e seu tempo para a construção desse trabalho.

“A tecnologia só é tecnologia para quem nasceu antes dela ter sido inventada.” (ALAN KAY).

RESUMO

Este trabalho tem como objetivo desenvolver uma solução de *Dollar-cost averaging* para tokens ERC-20 em *blockchains* EVM, aproveitando a crescente adoção das tecnologias *blockchain* pelos investidores. A falta de ferramentas automatizadas para estratégias de investimento nesse contexto motivou a realização de pesquisas e a aplicação de conhecimentos teóricos para a modelagem e implementação de uma aplicação inicial. O sistema foi avaliado por meio de um questionário aplicado a potenciais usuários, demonstrando que a solução desenvolvida atendeu aos objetivos principais e despertou interesse entre os participantes. O trabalho também aborda conceitos fundamentais de *blockchain* e destaca a importância da descentralização na infraestrutura utilizada, proporcionando uma nova forma de realizar estratégias de *Dollar-cost averaging*.

Palavras-chave: Blockchain. Investimentos. Descentralização.

ABSTRACT

This work aims to develop a Dollar-cost averaging solution for ERC-20 tokens on EVM blockchains, taking advantage of the growing adoption of blockchain technologies by investors. The lack of automated tools for investment strategies in this context has prompted research and the application of theoretical knowledge to model and implement an initial application. The system was evaluated through a questionnaire administered to potential users, demonstrating that the developed solution achieved its main objectives and generated interest among participants. The study also addresses fundamental concepts of blockchain and emphasizes the importance of decentralization in the underlying infrastructure, providing a new way to execute Dollar-cost averaging strategies.

Keywords: Blockchain. Investments. Decentralization.

LISTA DE FIGURAS

SUMÁRIO

1 INTRODUÇÃO

A criação do Bitcoin foi o estopim para a revolução da economia digital. A criptomoeda foi notadamente a primeira a obter êxito na criação de um ativo primariamente digital, distribuído e com valor intrínseco, mas ainda segundo Buterin (2014) há também outra parte igualmente importante no experimento de Satoshi: o conceito de blockchain proof-of-work com base em trabalho para permitir consenso público na ordem das transações.

O Bitcoin segundo Alharby e Moorsel (2017) "[…] usa uma linguagem script de bytecode baseada em pilha. A habilidade de criar smart-contracts complexos usando a linguagem de script do Bitcoin é muito limitada". E ainda segundo Buterin (2014) "[…] enquanto existe um largo intervalo de cálculos que a linguagem de script do Bitcoin suporta, não está perto de suportar tudo. A principal categoria faltante são os loops", e por não possuir loops não é considerada Turing-complete por autores como Buterin, Alharby e Moorsel.

A falta de uma linguagem Turing-complete pode dificultar a criação de novas aplicações, dado que desenvolvedores de aplicação e engenheiros de software não têm à sua disposição uma ferramenta que possa descrever cálculos ilimitados sobre valores ilimitados.
Em 2014 um grupo de pesquisadores e desenvolvedores propôs um novo paradigma de blockchain, que teria em seu cerne uma máquina virtual capaz de interpretar o bytecode gerado por uma linguagem Turing-complete, assim como guardaria os estados das interpretações do bytecode on-chain, sob a segurança da prova de trabalho da rede. Isso permitiu que desenvolvedores pudessem lançar suas aplicações customizadas na rede, chamadas smart-contracts, e assim ficaria disponível para todos os usuários. No dia 30 de julho de 2015, o primeiro bloco dessa rede foi minerado, e o mundo passou a conhecer a Ethereum.

Em novembro do mesmo ano, foi lançado o *Ethereum Improvement Proposal 20* (EIP-20) que mais tarde viraria o *Ethereum Request for Comments 20* (ERC-20), que segundo Vogelsteller e Buterin (2015) é "uma interface padrão que permite qualquer token na Ethereum ser reutilizado por outras aplicações: de carteiras a corretoras descentralizadas", ou seja, ao utilizar este padrão, desenvolvedores não apenas estariam lançando um *smart-contract* na blockchain, mas sim se conectando a um ecossistema de outros tokens, smart-contracts e usuários. A Figura 1 apresenta a transação do primeiro bloco minerado.

Este ecossistema hoje é conhecido como Decentralized Finance (DeFi), um mercado que segundo Pineiro-Chousa et al. (2022) "promove o uso da tecnologia de blockchain para criar e emitir todos as categorias de produtos e serviços financeiros". Segundo o índice DeFi

Pulse, hoje o mercado de DeFi tem um *total value locked* (TVL) de U$ 75.31B (maio de 2022, DeFi pulse), e em um período de dois anos (maio 2020, maio 2022) cresceu cerca de 700%.

Os tokens que compõem o mercado de DeFi, exceto os que têm seu lastro em alguma moeda fiduciária, apresentam grande volatilidade, assim como outras criptomoedas e ações da bolsa de valores. Com a finalidade de mitigar os riscos ao investir nessa classe de ativos, alguns investidores tendem a adotar a estratégia chamada de *Dollar-cost Averaging* (DCA).

Segundo Leggio e Lien (2003, p. 4):

> Dollar-cost averaging é um método de investimento popular onde um investidor com uma soma de dinheiro para investir não investe a totalidade da soma imediatamente; ao invés disso, ele investe uma proporção fixa do investimento em dólares em incrementos regulares temporais. Este método visa garantir que o investidor não invista o total nos períodos de alta do mercado e assim se arrependa de suas decisões de investimento posteriormente.

Este trabalho não conseguiu constatar, a existência de uma solução para ecossistema DeFi, que possibilite aos seus usuários investir utilizando a estratégia de DCA, utilizando apenas smart-contracts, sendo a garantia da descentralização da infraestrutura do código que compõe a plataforma. Portanto o objetivo deste trabalho é assim fazê-lo de forma que usuários possam depositar qualquer token que siga o padrão ERC-20, definir um intervalo fixo onde deseja-se trocar o token depositado por outro token escolhido e delimitar o valor do token inicial a ser investido neste intervalo de tempo.

1.1 PROBLEMÁTICA

Quando levado em consideração estratégias de automatização de DCA voltada ao mercado de criptomoedas, e como citado anteriormente, através de pesquisa previa, foram constatados serviços sendo ofertados por corretoras centralizadas, como Bipa, Crypto.com e Binance, mas até o presente momento, nenhum serviço foi encontrado por este trabalho que fizesse o uso de tecnologias decentralizadas, como a blockchain, para criar tal serviço de automatização.

Por diversas vezes, dentro do mercado de criptomoedas, foram observadas as consequências da centralização dos ativos dos usuários na mão de corretoras. Um dos mais recente destes eventos foi a falência da corretora FTX, que segundo Jalan e Matkovskyy (2023, p. 2):

> Uma das maiores exchanges de criptomoedas do mundo, avaliada em US$ 32 bilhões, desmoronou da noite para o dia devido a problemas de governança. O Wall Street

> Journal estima que o déficit de caixa da FTX seja de cerca de US$ 8 bilhões. A suspensão das retiradas de investidores causou falências de outras empresas de negociação com investimentos na FTX (como a BlockFi e Voyager), sem mencionar grandes baixas contábeis de fundos de capital de risco como BlackRock e Sequoia. Com a perda de cerca de 20% da capitalização de mercado de criptomoedas em novembro de 2022, o colapso da FTX é um grande abalo na confiança dos investidores.

Chohan (2023) da ênfase nos contrastes entre os eventos associados a falências de grandes corretoras, que são comparadas pelo autor a oligarquias poderosas pela grande concentração de capital, e os ideias apresentados pelos fundadores do movimento criptoanarquista, de onde saíram as primeiras ideias de criptomodas, que visavam a descentralização e anonimidade. Para o autor, o ecossistema *crypto* foi e ainda é usurpado por atores como FTX e seus similares, que traem a confiança do grande publico assim como os ideia fundamentais do mercado de criptomoedas.

Diversas outras corretoras centralizadas sofreram com golpes, *hacks* e falências no decorrer da história das criptomoedas. Outros exemplos citados por Charoenwong e Bernardi (2021) incluem *Mt. Gox, QudrigaCX, Bitfinex* e *Coincheck,* onde os números apresentados pelos autores, indicam que apenas nestes casos citados os prejuízos somam por volta de U$ 1.189.000.000.

São diversos os motivos que investidores tem para não confiar em infraestruturas centralizadas quando o assunto e o mercado de criptomoedas, por este motivo, este trabalha tentar resgatar os conceitos fundamentais dos criptoanarquistas quando se trata de anonimidade e descentralização da infraestrutura.

1.2 OBJETIVOS

1.2.1 Objetivo geral

O objetivo geral deste consiste em: desenvolver uma solução aplicada a qualquer blockchain que utilize a *Ethereum Virtual Machine* (EVM) que permita investidores se beneficiarem da estratégia de *Dollar-cost Averaging* no ecossistema *DeFi*.

1.2.2 Objetivos específicos

- Desenvolver os contratos que serão responsáveis pela compra e venda dos tokens
- Implementar persistência de dados dos usuários de forma descentralizada e anônima

- Construir a interface gráfica que será responsável por coletar as informações dos usuários e distribuir entre os outros componentes do sistema
- Automatizar compras dos tokens de todos os usuários que utilizam a plataforma
- Utilizar tecnologias descentralizadas onde forem possíveis suas aplicações

1.3 JUSTIFICATIVA

Seguindo a visão criptoanarquista criada por May (1988), que previa a criação de tecnologias que seriam capazes de fornecer a indivíduos e grupos formas de se comunicar e interagir de forma anônima, onde pessoas poderiam trocar mensagens e realizar negócios sem revelar sua identidade real e ainda e utilizando o conceito de cooperação definido por Dai (1998) "uma comunidade é definida pela cooperação de seus participantes e a cooperação eficiente requer um meio de troca (dinheiro) e uma maneira de cumprir contratos.", este projeto visa a criação de uma aplicação onde seus usuários possam se valer da estratégia de investimento DCA ao passo que também se beneficiam das qualidades providas pelas tecnologias descentralizadas como a redução dos pontos únicos de falha e auditabilidade das regras impostas pelo código que rege a plataforma. E ainda que permita aos usuários não revelarem sua identidade, caso assim seja da sua vontade.

Da mesma forma que criar uma solução descentralizada, anônima e auditável, pretende-se fornecer uma solução nativa do ecossistema de *DeFi*, de forma que usuários não tenham de recorrer a plataformas centralizadas para automatizar a sua compra e venda de tokens em intervalos fixos.

Ao combinar a estratégia de *Dollar-cost Averaging* com a descentralização proporcionada pela blockchain e o ecossistema *DeFi*, este trabalho contribui para o avanço do conhecimento e para o desenvolvimento de soluções inovadoras na área financeira. A análise e avaliação dos resultados obtidos com essa implementação permitirão uma melhor compreensão dos benefícios e desafios da utilização de estratégias de investimento descentralizadas, contribuindo para a evolução do campo de estudos relacionados à blockchain e ao *DeFi*.

2 REVISÃO TEÓRICA

Neste capítulo serão abordados os principais conceitos que permeiam as áreas de criptomoedas, *DeFi* e ainda o conceito de *Dollar Cost-Averaging*, temas estes que serão fundamentais para solução desenvolvida por este trabalho.

2.1 CRIPTOMOEDAS

Segundo a definição de Mukhopadhyay et al. (2016), uma criptomoeda é um sistema de trocas digitais onde a criptografia é usada para gerar e distribuir unidades da moeda.

Um dos primeiros registros de uma moeda digital, segundo Chuen (2015) data de 1990, ano de fundação da empresa DigiCash, Inc. e de seu sistema de pagamento digital baseado em criptografia, eCash pelo acadêmico e empresário David Chaum. O eCash permitia que fossem realizados pagamentos online e offline e se valia de protocolos criptográficos para garantir a integridade da operação de transferência de valor entre os usuários, problema conhecido como gasto duplo, e para garantir a anonimidade de quem utilizava o sistema. Como cita Abrar (2014), ao contrário do estado da arte das criptomoedas, eCash era um sistema de pagamento centralizado, já que um dos requisitos para o funcionamento do sistema era que ambos, comprador e vendedor, compartilhassem a mesma instituição financeira. A empresa DigiCash, declarou falência no ano de 1998, com isso dando fim ao desenvolvimento do eCash.

Segundo Chuen (2015), no decorrer dos anos pós DigiCash, diversas tentativas de criar uma moeda digital baseada em esbarraram em impeditivos legais, como cumprir com as diretrizes de know-your-customer (KYC) e relatórios de transações suspeitas. A iniciativa com maior sucesso, e-gold, antes de fechar as portas por problemas jurídicos, processava mais de U $2 bilhões anualmente.

A história mostrava um padrão: criptomoedas falham quando o sistema possui um ponto centralizador com poder o suficiente para derrubá-lo, seja por más decisões de administração, como no caso da DigiCash, ou por problemas legais como com a e-gold. Em ambos os exemplos, os respectivos sistemas foram descontinuados uma vez que a empresa deixou de existir.

Ao passo que esforços para a criação de uma moeda digital começavam a emergir nos meios empresariais, acadêmicos também começavam a publicar suas primeiras ideias sobre o assunto.

Inspirado pelas visões de cripto-anarquistas Timothy May, o cientista da computação Wei Dai (1998), descreveu um sistema de moeda descentralizado, apresentando várias ideias que viriam a ser importantes para o futuro das criptomoedas, chamado de b-money. O sistema seria uma rede conectada de usuários, onde cada usuário manteria um banco de dados com os saldos de cada participante desse sistema, e os participantes não seriam identificados por seus nomes, mas sim por uma chave pública, o que os tornaria anônimos para qualquer outro usuário do sistema. A criação das moedas seria feita através de desafios computacionais, onde o primeiro usuário que transmitisse a resposta correta para o problema teria o direito de receber novas moedas. Daí ainda cita que o maior desafio do seu protocolo era a criação de novas unidades da moeda, porque isso requereu que todas as partes do sistema concordassem no valor computacional gasto para resolver o desafio.

Outro cientista da computação que propôs um modelo teórico de uma moeda digital e descentralizada foi Nick Szabo (2005), que em seu artigo denominado Bit Gold, Szabo justifica a criação de uma moeda digital sem intermediários pelos episódios de hiperinflação ocorridos no século XX. Szabo usa as propriedades dos metais preciosos que os tornam valiosos como a escassez e a dificuldade de produção de novas unidades como pano de fundo para a estruturação da sua moeda digital teórica. Assim como *b-money*, *Bit Gold* também se valia de uma prova computacional de trabalho, onde usuários deveriam executar uma função de *benchmark* em suas máquinas e submeter tal prova para a rede. Tal prova se juntaria a uma cadeia de provas de trabalho, denominada por Szabo de registro de título de propriedade distribuída, onde através dela seria possível validar os saldos de *Bit Gold* de todos os usuários do sistema.

Trabalhos de mesma relevância para a história das criptomoedas foram desenvolvidos por Adam Back e posteriormente Hal Finney. Back (2002) criou uma contra resposta aos ataques de negação de serviço e abuso de recursos na internet, usando um mecanismo de prova de trabalho, onde para participar de uma rede um cliente teria de computar um token que deveria ser fácil de verificar sua integridade mas difícil de produzi-lo. Hal Finney (2004) estendeu o protocolo de Back de forma a tornar os tokens reutilizáveis, de forma que um servidor que manteria os registros de todos os tokens assinados por uma chave pública de seus respectivos donos, e tais donos poderiam realizar transferências desses tokens para novas chaves públicas, assinando uma transação para o servidor com sua chave privada e pedindo a transferência de seus tokens, o servidor então, verificando a assinatura privada do usuário com sua chave pública, atualizar seus registros.

Em 2008 um usuário anônimo sob o pseudônimo de Satoshi Nakamoto, propagou em uma lista de e-mail o que seria o trabalho mais importante no âmbito das criptomoedas: o

whitepaper do Bitcoin. Nele, Nakamoto (2008) descreveu o que chamou de “um sistema de dinheiro eletrônico de ponto a ponto” ao que deu o nome de Bitcoin. No seu trabalho, Nakamoto cita trabalhos mencionados anteriormente como o de Adam Back e Wei Dai, porém a maior contribuição do Bitcoin foi a solução do problema de gasto duplo em um sistema puramente *peer-to-peer*.

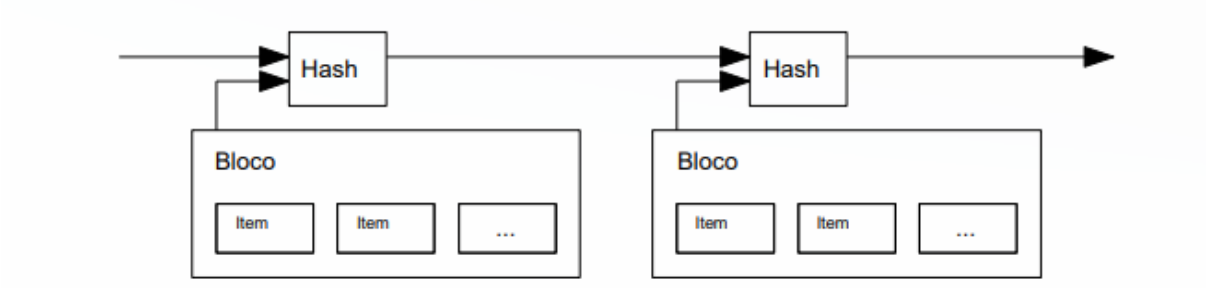
Gasto duplo, segundo Chohan (2021) é:

> [...] é um potencial falha na criptomoeda ou dinheiro digital onde o mesmo token digital pode ser gasto mais de uma vez, e isso é possível porque um token digital consiste em um arquivo digital que pode ser duplicado ou falsificado.

Para resolver este problema Nakamoto (2008) propôs “um servidor de horas distribuído *peer-to-peer* para gerar prova computacional da ordem cronológica das operações” para isto, segundo Nakamoto (op. cit.) a condição para que o sistema fosse seguro era que “nós honestos controlem coletivamente mais poder de CPU do que qualquer grupo cooperado de nós atacantes”. Ainda segundo o autor citado, para que haja uma forma de verificar que moedas não foram gastas duas vezes pelos seus donos, o sistema deve conhecer e concordar sobre todo o histórico e ordem das transações no sistema, e “sacador precisa da prova que, no momento de cada transação, a maioria dos nós concorda que ela está sendo recebida pela primeira vez”.

O servidor *timestamp* proposto por Nakamoto (op. cit.) funciona de forma a criar um hash de uma coleção de itens e publicá-los amplamente na rede, ao passo que o hash da próxima coleção de itens deve conter o timestamp da coleção anterior, formando assim uma cadeia, com cada *timestamp* reforçando os seus antecessores. Tal conceito ficou conhecido como blockchain.

Figura 1 - Fluxograma de Nakamoto para representar o servidor de timestamp.



Fonte: Nakamoto, Satoshi. 2008

Segundo Chohan (2022, p. 9) **"**Depois da disseminação do artigo, a plataforma para transação de Bitcoins passou a existir através do lançamento do primeiro cliente de Bitcoin e da emissão concomitante de Bitcoins**".** Após o lançamento e a proliferação das ideias do Bitcoin, diversos projetos de criptomoedas foram lançados. Segundo o site CoinMarketCap existem hoje mais de 21.735 ativos digitais entre criptomoedas e tokens em circulação.

2.2 BLOCKCHAINS DE PROPOSITO GERAL

Como visto anteriormente, Nakamoto (2008) descreveu um servidor de *timestamp* que fazia o uso de uma estrutura de dados de blocos encadeados por criptografia responsável por manter os registros de transações de uma rede descentralizada, denominada blockchain.

Apesar de revolucionário, a blockchain de Satoshi possui a limitação do uso da rede *peer-to-peer* estritamente para a transferência de Bitcoin entre seus participantes. Novas aplicações não podem ser construídas se valendo da segurança criada pelo mecanismo de consenso desta blockchain. A única forma de automatizar ações dentro da rede do Bitcoin é utilizando o chamado Bitcoin script, que apenas permite a criação de lógicas que permitem múltiplas assinaturas em uma transação.

Em 2014, Vitalik Buterin lançou o *whitepaper* da Ethereum, que tinha a proposta de ser uma blockchain, de propósito geral. Neste trabalho, Buterin (2014) discute a dificuldade de novas aplicações criarem e manterem cadeias de blocos longas o suficiente para serem consideradas seguras onde segundo ele "cada implementação individual precisa inicializar uma cadeia de blocos independente, bem como a construção e o teste de toda a transição de estado necessária e código da rede" e ainda afirma que "o conjunto de aplicações para tecnologia de consenso descentralizada seguirá uma lei de distribuição de poder onde a grande maioria dessas aplicações seria muito pequena para garantir sua própria cadeia de blocos".

O modelo sugerido por Buterin utiliza a mesma ideia fundamental do trabalho de Nakamoto, de uma rede peer-to-peer onde os seus nos concordam no estado atual do sistema através de uma cadeia de blocos protegidos pelo prova de trabalho, porém a Ethereum apresentou a ideia de uma blockchain de propósito geral, ou seja, que permitia a criação e execução de aplicações arbitrárias através de uma linguagem de programação *Turing-complete,* oferecendo a desenvolvedores e usuários a segurança e descentralização tais quais presentes no modelo do Bitcoin.

A Ethereum definiu uma série de conceitos que mais tarde seriam implementados por outras blockchains de propósito geral.

O primeiro desses conceitos é o que Buterin (2014) chamou de contas Ethereum, que compõem o estado da rede. Cada uma dessas contas possui um *nonce,* um contador incrementado a cada transação, o balanço em Ether, o código do contrato, em caso de a conta representar um *smart-contract,* e o armazenamento da conta, vazio por padrão.

Ainda segundo Buterin (op. cit), existem dois tipos de contas: contas externas, as quais são controladas por chaves privadas e contas de contratos, controladas pelo código de seus contratos.

Assim como o Bitcoin, a Ethereum também possui seu token nativo, o mencionado anteriormente Ether. Para cada transação que altere o estado da blockchain, segundo Buterin (op. cit.) "é necessário a definição de um limite de quantos operações computacionais de código podem ser usadas" e tal mecanismo tem o intuito de "prevenir *loops* infinitos acidentais ou maliciosos ou ainda outros tipos de desperdícios computacionais", o autor batiza a unidade de operação computacional de *gas*. Cada transação define duas variáveis em seu escopo para atingir tal objetivo citado anteriormente: STARTGAS e GASPRICE. A primeira é o limite de operações computacionais a serem executadas pelo contrato, enquanto a segunda é o preço de uma unidade de *gas,* denominada em Gwei, uma fração do token Ether.

Buterin (op. cit.) determina que o preço a ser pago aos mineradores da rede é calculado usando $STARTGAS * GASPRICE$.

Os *smart-contracts* que rodam na Ethereum são formados por uma cadeia de bytes, chamados de *bytecodes*. O *bytecode* é composto por todas as instruções, que o criador do contrato deseja executar na blockchain, tais instruções são denominadas *opcodes*. Desenvolvedor não necessitam escrever tal cadeia de *bytes* manualmente para seus *smart-contracts,* existem linguagens de programação, tendo Solidity e Vyper entre as mais populares, que permitem que seja usada sintaxe de alto nível, similares a linguagens como JavaScript e Python para que sejam definidas as regras de execução do programa. Uma vez finalizada a lógica de negócio a ser inserida na blockchain, tais linguagens, usando seus respectivos compiladores, irão criar o *bytecode* do contrato que será executado posteriormente pela blockchain.

A Ethereum, assim como Bitcoin, foi pioneira dentro das tecnologias de blockchain, lançando o conceito de uma blockchain de propósito geral e segue até os dias atuais entregando atualizações ao seu software de cliente principal, *geth.*

2.3 TOKENS E O PADRÃO ERC-20

De acordo com Angelo e Salzer (2020, p. 1) "*Um token criptográfico é um ativo digital em cima de uma criptomoeda ou blockchain, geralmente como um ativo programável gerenciado por um smart-contract [...]*".

Como mencionado anteriormente, a Ethereum foi o primeiro projeto que trouxe a ideia de blockchain de propósito geral, que permitia a execução de código arbitrário por parte de seus usuários, portando criação de tokens com as características da definição anterior, e mais que isso, permitiu a interação entre contratos, criando assim um ecossistema interconectado de *smart-contracts*. Logo surgiu a necessidade da padronização de casos de uso comuns, para que usuários e aplicações pudessem interagir entre si com mais facilidade, como cita Buterin (2015).

No ecossistema Ethereum, tais padronizações acontecem através de *Ethereum Improvement Proposals* (EIP), que segundo a Becze e Jameson (2015), são documentos provendo informações para a comunidade Ethereum ou descrevendo novas funcionalidades para Ethereum ou para seus processos e ecossistema. Dentre os três tipos de EIPs existentes, para o contexto deste trabalho, destaca-se a *Standards Track EIP,* que segundo Becze e Jameson (op. cit.) descrevem qualquer mudança que afeta a maioria ou todas as implementações da Ethereum. Os autores ainda citam quatro categorias de *Standards Track EIP, que* novamente para o contexto deste trabalho, vamos destacar a *Ethereum Request for Comments* (ERC), que são padronizações e convenções a nível de aplicação.

A primeira EIP relacionada a padronização de interface de *smart-contract*, ou seja, uma ERC, foi a ERC-20, que segundo Vogelsteller e Buterin (2015) permite a implementação de uma API para tokens dentro de *smart-contracts,* e ainda segundo os autores a motivação para tal padronização é a possibilidade de qualquer token na Ethereum ser reutilizado por outras aplicações, de carteiras a corretoras descentralizadas.

A interface proposta na ERC-20 apresenta uma lista de nove métodos, que tem como os mais relevantes para o desenvolvimento deste trabalho os seguintes:

- *transfer:* deve implementar a transferência de uma quantidade arbitrária do presente token entre duas contas Ethereum.
- *transferFrom:* deve implementar a transferência de uma quantidade arbitrária de tokens em nome de uma conta Ethereum. Tal transferência só deve acontecer mediante aprovação explícita da conta detentora dos tokens.

- *approve:* utilizada na execução da função *transferFrom*, dá permissão para que outra conta Ethereum faça transferência em nome da conta detentora dos tokens

Segundo o site Etherscan, existem 699.183 contratos de tokens ERC-20 circulando na rede Ethereum atualmente.

2.4 FINANÇAS DESCENTRALIZADAS, DESCENTRALIZED EXCHANGES E AUTOMATED MARKET MARKERS

Finanças descentralizadas, do inglês *Decentralized Finance*, coloquialmente conhecida por *DeFi*, segundo Jensen et al. (2021) são um novo conjunto de aplicações financeiras abertas implementadas em blockchains públicas e desprovidas de permissões. Ainda para os autores, o termo aplicação *DeFi* denota um arranjo de *smart-contracts* voltados ao consumidor, executando lógicas de negócio pré-estabelecidas dentro do ambiente computacional transparente e determinístico proporcionado pela tecnologia publica das blockchains. De acordo com Qin et al. (2021) *DeFi* suporta a maioria dos produtos disponíveis no mercado de finanças tradicionais: troca entre ativos, empréstimos, trocas alavancadas, votos de governança decentralizados e *stablecoin.*

Um dos produtos mais importantes do mercado *DeFi* e conhecido como *Decentralized Exchange* (DEX), uma plataforma de compra e venda de tokens de maneira pública, distribuída e desprovida de permissões. Tokens recém-lançados na blockchain podem usar uma DEX para oferecer a seus usuários uma forma de negociar o ativo digital em troca de outros, sem ter a necessidade de serem listados em uma *Centralized Exchange* (CEX), o que demandaria interesse econômico por parte da corretora centralizada em custodiar operações e oferecer tal token para seus usuários.

Para Aspris et al. (2021, p. 3):

> O termo DEX geralmente se refere a protocolos e aplicativos de contabilidade distribuída que permitem aos usuários transacionar criptomoedas na ausência de uma contraparte centralizada. Diferentemente das corretoras centralizadas de criptomoedas, que são serviços de custódia que guardam os tokens de seus participantes, as DEXs são confiáveis, o que significa que todas as transações ocorrem por meio de carteiras de participantes usando contratos inteligentes, com liquidação de cada transação terceirizada diretamente para o blockchain.

Uma DEX pode implementar tais trocas entre tokens de três maneiras de acordo com Pourpouneh et al. (2020):

- **Livro de ofertas com dados na blockchain:** um *smart-contract* é responsável por encontrar pares de compradores e vendedores de um determinado ativo, de forma que ambos concordem com o preço definido de um ativo, e uma vez que seja identificada uma oferta e uma demanda, a troca entre tokens é realizada sob os moldes de negociações *peer-to-peer*. Todos os registros de demanda, por parte dos compradores, e oferta, por parte dos vendedores, ficam armazenados na blockchain.
- **Livro de ofertas com dados fora da blockchain:** similar a um livro de oferta com dados na blockchain, este método também visa encontrar pares de vendedores e compradores, porém os dados de oferta e demanda são armazenados fora da *blockchain*, e apenas a transação entre as duas partes acontece utilizando *smart-contracts*, novamente sob o modelo *peer-to-peer*.
- **Automated Market Makers (AMM):** são *smart-contracts* com o proveem liquidez automática entre dois tokens, usando o conceito de *pools* de liquidez onde um algoritmo define o preço dos tokens de forma determinística através da demanda e oferta, e com isso a troca entre os tokens por parte de um usuário aconteça utilizando o modelo *peer-to-pool*, onde nenhum intermediário é necessário para que a operação aconteça.

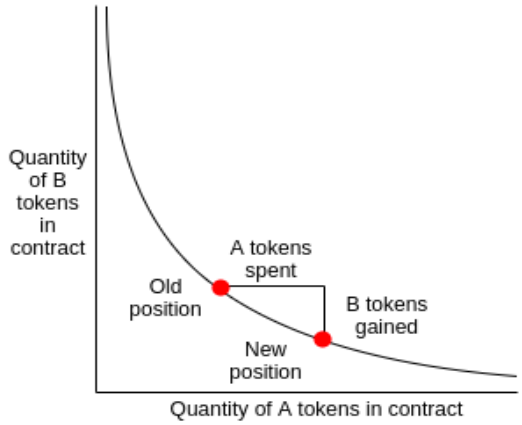
Os AMMs passaram a ser o estado da arte para DEXs por serem mais eficientes e apresentarem custos menores por transação, segundo Pourpouneh et al. (2020). As maiores DEXs em *total value locked* (TVL), Curve (U$ 3.79B), Uniswap (U$ 3.77B) e Pancake Swap (U$ 2.48B) segundo o site DeFiLlama, utilizam de algoritmos de AMM para construir seus produtos.

Como citado anteriormente, AMMs utilizam *pools* de liquidez para suportar a troca entre dois tokens. Para a finalidade deste trabalho, é apresentado o algoritmo de pools de liquidez do protocolo utilizado pela Uniswap e Uniswap v2, uma iteração em cima de sua forma inicial, que integra a solução proposta ao problema apresentado.

Segundo Adams (2018) a Uniswap utiliza a fórmula de produto constante para definir o preço entre dois ativos, onde x e y representam as reservas de dois tokens ERC-20 diferentes depositados no *smart-contract* e k é uma constante, produto da invariância entre x e y, como pode ser vista na fórmula: $x * y = k$

Segundo Buterin (2018) tal fórmula permite que qualquer pessoa possa comprar e vender moedas mudando a posição do *market maker* na curva da equação apresentada acima.

Gráfico 1 - Gráfico do produto constante para tokens B em função de tokens A.



Usuários da Uniswap, depositam dois tokens no contrato do protocolo, a chamada pool de liquidez, e estes são usados como reservas, x e y, na fórmula do produto constante. O primeiro depósito de um usuário no contrato da pool define a constante k, a partir do produto das quantidades depositadas de cada token. A cada troca entre dois tokens, o equilíbrio entre as reservas é alterado, atualizando o preço de cada token de forma inversa.

Os provedores da liquidez de cada pool, como são chamados os usuários que depositam seus tokens no contrato da pool, recebem um terceiro token indicando sua posição no momento do depósito, que podem posteriormente ser usados para remover a liquidez do contrato, recebendo as quantidades de cada token conforme o novo equilíbrio entre as quantidades das reservas da pool. Tais provedores de liquidez também recebem taxas das operações das pool, como incentivo para tal ação. Cada troca entre dois tokens de determinada pool pagam 0,3%, e tal taxa é distribuída entre os provedores de liquidez proporcionalmente às reservas depositadas no contrato.

2.5 DOLLAR-COST AVERAGING

Segundo Leggio e Lien (2003, p. 221):

> Dollar-cost averaging é um método de investimento popular onde um investidor com quantia para investir não investe a totalidade imediatamente; ao invés disso, ele investe uma proporção fixa em dólares em incrementos regulares temporal. Este método visa garantir que o investidor não invista a soma nas altas do mercado e assim se arrependa de suas decisões de investimento posteriormente.

O termo *Dollar-cost averaging* foi cunhado em 1949 pelo economista americano Benjamin Graham em seu livro *The Intelligent Investor*, e desde então diversos estudos acadêmicos foram conduzidos para avaliar os resultados deste método de forma a determinar o seu grau de otimização quando comparados a outros tipos de investimentos como o *lump-sum*, que consiste em investir a soma total do montante disponível em uma única operação.

Williams e Bacon (1993) traçam um comparativo entre *lump-sum (LS),* que consiste em investir a soma total do montante disponível em uma única operação e DCA, usando dados históricos do índice S&P 500 e dos títulos do tesouro americano, entre os períodos de 1926 a 1991. A estratégia analisada de DCA consistia na alocação do montante total de investimento em títulos do tesouro, com transferências mensais de valor entre os títulos de tesouro e o índice S & P 500 por 12 meses. Ao passo que a estratégia de LS consistia em alocar o montante de uma só vez no índice S & P 500. Ao final foram comparados os resultados anuais de todos os períodos citados acima e chegou-se à conclusão de que a estratégia de LS apresentou resultados melhores em dois terços das vezes.

Knight e Mendell (1993) discutem a performance do DCA, quando comparados a *Optimally Balanced* e *Buy and Hold.* Usando simulação de Monte Carlo e parâmetros extraídos da bolsa de valores de Nova Iorque entre os anos de 1962 e 1992, o estudo mostrou que apesar de apresentar uma melhor performance teórica e em simulações numéricas, as diferenças empíricas entre os três métodos não são significativas.

O método de DCA também é criticado por Constantinides (1979) que afirma ser irônico que investidores não levaram em consideração informações que se tornaram conhecidas durante o seu processo de investimento, continuando a investir os mesmos montantes em intervalos fixos.

Apesar das críticas acadêmicas e de seus resultados discutíveis em estudos de performance, o DCA ainda continua sendo um método de investimento popular entre investidores que buscam fugir da volatilidade dos ativos que desejam ter em seu portfólio.

Statman (1995) mostra que o sucesso do DCA pode ser explicado pela economia comportamental, já que tal estratégia ajuda investidores a investirem em um ritmo constante com uma quantidade de opções limitadas no curto prazo, o que reduz o arrependimento, a caça as altas e baixas do mercado e o embasamento em extrapolações infundadas das variações de preços atuais.

Truite (2017) apresenta uma análise de performance entre três métodos de investimento, *Dollar Cost Average, Lump Sum* e *Value Average,* comparando o valor do índice de Sharpe calculado ao final do período de análise para cada um dos investimentos. Foram utilizados

dados do mercado financeiro brasileiro do período entre janeiro de 2010 a dezembro de 2016 sendo eles o índice Ibovespa, dólar comercial e taxa de juros Selic. O portfólio a ser analisado foi composto por 70% de renda variável (Ibovespa) e 30% por dólar comercial, enquanto a taxa Selic foi utilizada como ativo livre de risco. Ao final da análise foi possível constatar que o DCA foi o investimento que teve o maior índice de Sharpe, ou seja, a melhor razão entre retorno e risco entre os demais métodos.

Conforme os resultados apresentados acima, podemos ver que existe discordância entre acadêmicos sobre a performance do DCA como estratégia de investimento, porém este trabalho visa equipar investidores com uma ferramenta para que estes possam se valer de tal estratégia no momento que sua análise do mercado os diga que é o melhor a ser feito.

Não foi encontrado nenhum estudo que comparasse diferentes métodos de investimento, incluindo DCA, utilizando criptomoedas.

2.6 INCENTIVOS E DESIGN DE MECANISMOS

Um software que pretende oferecer a seus usuários uma solução automatizada de investimentos, depende fundamentalmente da execução de trechos de seu código fonte em intervalos programados de tempo. Ao olhar para o estado da arte da computação, pode-se constatar que esse é um problema já muito bem resolvido. Usa-se como exemplo o utilitário de linha de comando do Unix, *Cron*, com exemplo. Ele permite que scripts *bash* sejam executados em intervalos fixos previamente definidos pelo usuário.

O utilitário citado anteriormente faz parte de um subconjunto de softwares chamados de *job scheduler,* que segundo a Wikipedia é um aplicativo de computador para controlar a execução autônoma de programas em segundo plano.

Um *job scheduler* permitiria a invocação de um script hipotético *trocar-tokens.sh,* que faria a troca dos tokens de cada um dos usuários da aplicação, vendendo um token hipotético X para comprar um token hipotético Y utilizando um AMM, como a Uniswap.

Porém, *job schedulers* não integram a especificação de clientes da *Ethereum* nem de outras blockchains baseadas em sua tecnologia, por isso não existe nenhuma forma nativa de criar atividades programadas para rodar na blockchain em determinados intervalos de tempo. Tendo em vista tal limitação técnica, foram criadas algumas práticas para permitir a automatização de tarefas dentro de uma blockchain que rode através de uma *EVM.*

A primeira e mais simples e a criação de um *script* que interaja com a blockchain, criando uma transação que invoque uma das funções do smart-contract e a envie a rede, atrelado

a um *job scheduler,* como o *Cron* citado anteriormente, para que esta transação seja repetida dentro do intervalo de tempo pré-definido. Porém, este método apresenta um ponto único de falha, onde o sistema não possui redundâncias para possíveis erros que possam ocorrer em um determinado componente do sistema, causando um interrompimento do tempo de disponibilidade. Por ser um script rodando em uma única máquina, se tal equipamento apresentar algum problema de hardware ou software, a execução da transação não acontecera, o que acarreta mal funcionamento do serviço sendo oferecido aos usuários, onde a troca de um token por outro não será feita, e continua não sendo feita até que o administrador do serviço restabeleça o funcionamento do sistema. Sendo o contrato desenvolvido por este trabalho sensível ao tempo, já que há variações nos preços dos ativos a todo instante dentro da blockchain, uma falha no sistema pode acarretar uma oportunidade de comprar um ativo por um preço inferior ao que se praticava anteriormente.

Uma segunda opção para automatização de código na blockchain é o uso de serviços de empresas como Chainlink e OpenZeppelin, com seus respectivos produtos *Chainlink Automation e OpenZeppelin Autotasks.* Ambos fornecem aos seus usuários maneiras mais sofisticadas de agendar transações na blockchain, porém o cerne destes produtos se assemelha a primeira ideia apresentada de um *job scheduler* executando um *script* em intervalos de tempo pré-definidos. *OpenZeppelin Autotasks* utiliza serviços centralizados por baixo dos panos como *AWS Key Management Service* (OpenZeppelin, 2022), o que gera novamente pontos únicos de falha, já que segundo o acordo de nível de serviço da Amazon a disponibilidade deste serviço pode variar entre 95% e 99%. Apesar disto este serviço é oferecido de forma gratuita aos usuários com até 120 transações por hora.

Chainlink Automation se vale de uma rede descentralizada de nodos para a execução das automações dos usuários (Chainlink, 2022), o que remove o ponto único de falha citado anteriormente sobre o serviço concorrente, porém este é um serviço pré-pago, onde usuários devem depositar tokens da empresa Chainlink, LINK, em suas contas e tais tokens serão consumidos toda vez que uma automação for executada. Com isso, vê-se a inserção de um novo ponto único de falha, já que requer que o operador das automações mantenha sempre a conta com fundos para que o contrato seja executado, caso contrário o serviço de automação é interrompido até que a conta receba novos tokens LINK.

Outra forma de analisar os desafios da execução de código na blockchain de forma autônoma seria enxergar como um problema de coordenação e incentivos e não automatização de código. Ou seja, as partes integrantes do sistema, os usuários, executariam funções conforme as regras definidas pelo sistema de forma a receberem incentivos para tomar tal ação.

Segundo Tan (2020) pessoas naturalmente tendem a não cooperar a não ser que sejam incentivadas a tal. Segundo a autora, tokens podem ser usados para incentivar uma estratégia em particular e ainda cita que cada ecossistema tem um objetivo, e um token incentiva participantes em direção a este objetivo em comum.

Tais estratégias para incentivar os usuários a tomarem certas ações dentro de um sistema, de acordo com Tan (op. cit.), são criadas através do *design* de mecanismos, que define as regras do meio que existem para governar as ações dos participantes. E a mesma explica que no processo de criar os mecanismos do sistema através do design de mecanismos, deve-se olhar para o design de instituições, mercados e ecossistemas para entender como estes afetam os resultados das interações das instituições para poder atingir os objetivos.

Quando olha-se para o problema da execução de código dentro da blockchain, pode-se enxergar uma oportunidade de implementar um sistema onde participantes serão incentivados a tomar determinados tipos de ações (executar transações), ao passo que receberão incentivos econômicos, em forma de tokens por estas ações.

3 METODOLOGIA

O objetivo deste capítulo é apresentar os procedimentos metodológicos utilizados para guiar este trabalho.

3.1 CLASSIFICAÇÃO DA PESQUISA (CITAR REFERÊNCIAS BIBLIOGRÁFICAS)

A pesquisa desenvolvida por este trabalho pode ser classificada como bibliográfica, qualitativa e aplicada.

Uma pesquisa bibliográfica tem como objetivo analisar trabalhos publicados com o objetivo de reunir informações e dados que serão explorados no restante da pesquisa para fundamentar as soluções e conclusões propostas. Este trabalho se vale da pesquisa bibliográfica para analisar conceitos do ecossistema criado ao redor das *blockchains*, assim como fundamentar a solução baseada na estratégia de investimento de Dollar Cost-Averaging.

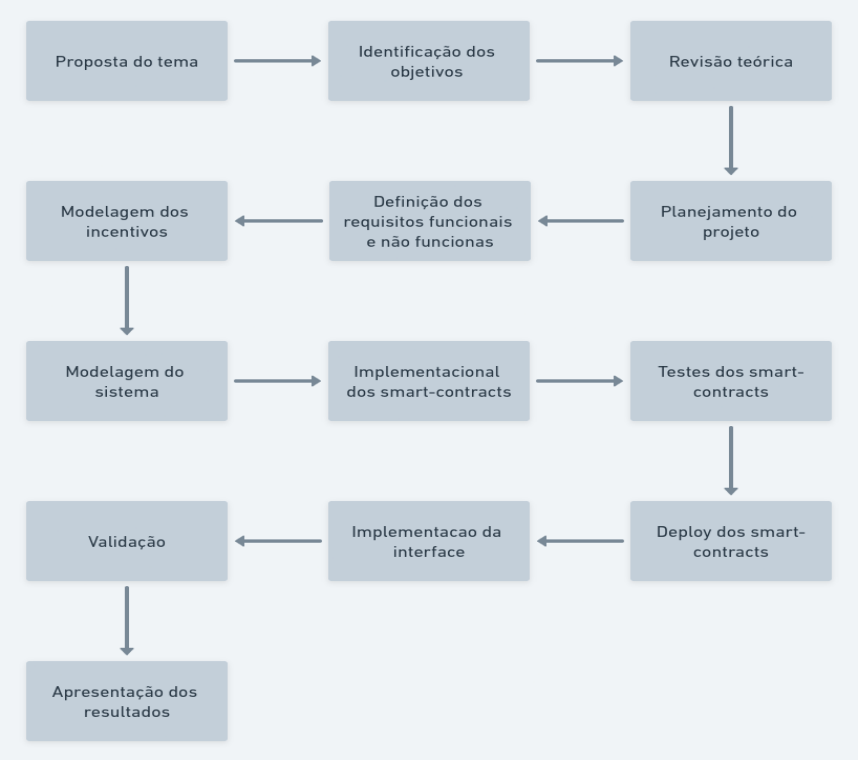
Na realização de uma pesquisa qualitativa busca-se fazer a análise de dados que não podem ser analisados numericamente, a fim de interpretar comportamentos e fenômenos. A presente pesquisa se vale das ferramentas da abordagem qualitativa com a finalidade de compreender o mercado de criptomoedas e as estratégias de investimento usadas por investidores.

A pesquisa aplicada visa a aplicação prática dos conceitos teóricos que delimitam o tema do trabalho. Esta pesquisa, após a apresentação das definições teóricas, visa concretizar tais conceitos através da solução de aplicação proposta pelo escopo desta exploração.

3.2 ETAPAS METODOLÓGICAS

Com o objetivo de guiar os passos deste trabalho, foi elaborado o fluxograma a seguir com todas as atividades a serem desenvolvidas e entregues.

Figura 2 - Fluxograma contendo as atividades



Fonte: Autoria própria, 2023

- **Proposta do tema:** apresenta os assuntos que serão abordados no decorrer da pesquisa, assim como delinear a solução a ser implementada.
- **Identificação dos objetivos:** define o escopo da pesquisa aplicada através de objetivos a serem alcançados por ela.
- **Revisão teórica:** a partir dos temas elencados no primeiro passo, realiza-se uma pesquisa bibliográfica a título de se obter um referencial teórico para a solução desenvolvida, apresentado o estado da arte de cada conceito.
- **Planejamento do projeto:** utilizando o referencial teórico criado no passo anterior, são planejadas as etapas a serem cumpridas durante o desenvolvimento do trabalho a título de guiar as entregas que serão feitas.
- **Definição de requisitos funcionais e não funcionais:** utilizando o tradicional modelo de levantamento de requisitos do campo da engenharia de *software*, define-se todas as regras que estão presentes no sistema e também delimitar as funcionalidades que o sistema não deve apresentar.

- **Modelagem dos incentivos:** fundamentado nos conceitos apresentados de design de incentivos e mecanismos, elabora o modelo de incentivos que estimula os usuários do sistema a tomarem as ações necessárias para o bom funcionamento deste.
- **Modelagem do sistema:** cria uma visão geral de como os componentes do sistema estarão interligados, definindo todas as possíveis linhas de interação entre eles, de forma a guiar a implementação.
- **Implementação dos smart-contracts:** utilizando a linguagem Solidity e a rede de testes da Ethereum, implementar os *smart-contracts* que serão responsáveis pelas regras de negócios do sistema
- **Testes dos smart-contracts:** a partir das funções implementadas no smart-contract no passo anterior, criar testes unitários e de integração para a integridade das regras de negócio implementadas
- **Deploy dos smart-contacts:** uma vez garantida a correta execução dos contratos, cria-se os scripts que são responsáveis pelo deploy dos smart-contracts em qualquer blockchain compatível com a *Ethereum Virtual Machine*.
- **Implementação da interface:** cria uma interface web que atua como uma ponte entre o usuário e os contratos de forma a facilitar a interação entre ambas as partes.
- **Validação:** testa o sistema de forma a garantir que todos os seus requisitos funcionais e não funcionais, definidos anteriormente, foram atendidos.
- **Apresentação de resultados:** analisa todos os passos anteriores e apresentaa as dificuldades encontradas, possibilidades de melhoria para uma futura iteração do projeto e identificação de oportunidades de pesquisa para trabalhos futuros.

3.3 DELIMITAÇÕES

A seguir serão apresentadas as delimitações impostas a este trabalho:

- Apesar da solução apresentada ter a intenção de auxiliar investidores com suas estratégias de investimento, não são analisados os impactos do uso da aplicação no portfólio do investidor.

- Muitos projetos de DeFi apresentam extensos relatórios sobre a segurança de seus *smart-contracts*. Entende-se que isto é uma parte importante de um projeto, mas não é compreendido no escopo do corrente projeto, principalmente pela complexidade extra que seria introduzida para sua implementação.

- Não são analisados os resultados utilizando uma *blockchain* de produção, ao invés disso serão usadas *blockchains* chamadas de *testnets*, que simulam o comportamento das *blockchains* de produção, porém seu uso não depende de recursos financeiros.

- Como o principal objetivo deste trabalho é a criação de *smart-contracts* que implementem estratégia DCA de investimento, não é feita uma análise aprofundada das tecnologias utilizadas para a criação da interface de usuário do projeto.

4 PROPOSTA DE SOLUÇÃO

O desenvolvimento de um sistema de software requer a utilização de diversas ferramentas e técnicas para a modelagem e construção do sistema. Neste capítulo, são abordados alguns dos principais artefatos UML utilizados no processo de desenvolvimento do sistema, como o Diagrama de Classes, Diagrama de Componentes e Casos de Uso. Além disso, é apresentado um protótipo de telas do sistema, que tem como objetivo fornecer uma visão geral da interface com o usuário.

4.1 UML

De acordo com Booch et al. (2005), a Unified Modeling Language (UML) é uma linguagem gráfica padronizada amplamente utilizada para modelagem de sistemas de software orientados a objetos. A UML permite que os desenvolvedores representem de forma clara e concisa a estrutura e o comportamento de sistemas complexos, auxiliando no processo de desenvolvimento e na comunicação entre os envolvidos.

No contexto deste trabalho, o uso da UML é importante para a modelagem e representação do Smart Contract que implementa a estratégia de dollar cost averaging. Como destacam Guedes e Silva (2016), a utilização dos diagramas UML permite representar graficamente a estrutura do sistema, identificar as classes e suas relações, bem como as interações entre os objetos envolvidos.

4.2 REQUISITOS

Os requisitos são uma parte fundamental no processo de desenvolvimento de software, pois eles definem o que o sistema deve fazer e como ele deve se comportar. Os requisitos podem ser divididos em diferentes categorias, como requisitos funcionais, não funcionais e regras de negócio. De acordo com Pressman (2016), os requisitos funcionais descrevem o comportamento do sistema em relação às funcionalidades que ele deve oferecer, enquanto os requisitos não funcionais descrevem as restrições e características não relacionadas diretamente às funcionalidades, mas que são importantes para o sistema, como a usabilidade, a segurança e o desempenho.

Além dos requisitos funcionais e não funcionais, as regras de negócio também são importantes para a definição do comportamento do sistema. Conforme destacam Barbosa e

Silva (2014), as regras de negócio representam as políticas, práticas e procedimentos que orientam a operação de uma organização e, portanto, devem ser levadas em consideração no processo de desenvolvimento de software.

Nessa seção, são apresentados os conceitos de requisitos funcionais, não funcionais e regras de negócio, e como eles se relacionam na especificação e modelagem do sistema.

4.2.1 Requisitos funcionais

Os requisitos funcionais descrevem o comportamento do sistema em relação às funcionalidades que ele deve oferecer. Eles representam as ações que o sistema deve ser capaz de realizar para atender às necessidades dos usuários e cumprir os objetivos do negócio. De acordo com Sommerville (2015), os requisitos funcionais são críticos para o sucesso do projeto de software, pois eles definem o que o sistema deve fazer.

Para este trabalho foram definidos os seguintes requisitos funcionais:

Quadro 1 - Requisitos funcionais

Identificação	Requisito
RF001	O sistema deve listar todas as alternativas de investimento
RF002	O sistema deve permitir que o usuário escolha a quantidade de tokens que devem ser negociados
RF003	O sistema deve permitir que o usuário escolha por quantas iterações a estratégia de DCA deve ser executada
RF004	O sistema deve permitir que o usuário saque a quantidade de tokens já negociados pelo smart-contract em seu nome
RF005	O sistema deve permitir que o usuário termine a sua posição de investimento a qualquer momento, recebendo a quantidade integral dos tokens negociados e não negociados
RF006	O sistema deve listar todas as posições em vigor do usuário
RF007	O sistema deve apenas listar posições e investimentos disponíveis na blockchain na qual a carteira do usuário está conectada
RF008	Para cada investimento disponível, o sistema deve mostrar o nome e preço dos tokens e a razão entre o token de entrada (depositado pelo usuário, e o de saída (negociado pelo contrato em nome do usuário)
RF009	A interface do sistema não deve listar posições que já tenham sido concluídas

RF010	O sistema deve possibilitar o uso da principal carteira de criptomoedas para blockchains compatíveis com EVM, Metamask
RF011	O sistema deve oferecer a administradores uma forma de criar uma nova opção de investimento sem que os mesmos devam interagir diretamente com o smart-contract
RF012	O sistema deve permitir que apenas usuários com carteiras conectadas a aplicação enviem transações a blockchain

Fonte: Autoria própria, 2023

4.2.2 Requisitos não funcionais

Os requisitos não funcionais são aqueles que definem atributos de qualidade do sistema, tais como desempenho, segurança, confiabilidade e usabilidade. Eles descrevem como o sistema deve se comportar em relação a esses atributos e, muitas vezes, são tão importantes quanto os requisitos funcionais. Segundo Sommerville (2015), os requisitos não funcionais representam os critérios pelos quais o sistema será avaliado e são fundamentais para a aceitação do sistema pelos usuários.

Para este trabalho, foram definidos os seguintes requisitos não funcionais:

Quadro 2 - Requisitos não funcionais

Identificação	**Requisito**
RNF001	A camada de smart-contract do sistema deve ser compatível com qualquer blockchain que rode sobre uma EVM
RNF002	O tempo de disponibilidade dos sistemas deve ser o mesmo da blockchain onde os smart-contracts foram lançados
RNF003	O sistema deve permitir que novas carteira de criptomoedas sejam integradas a qualquer momento
RNF004	O sistema não teve ter nenhuma de suas funcionalidades afetas caso seja acessado usando o aplicativo para dispositivos móveis da carteira Metamask

Fonte: Autoria própria, 2023

4.2.3 Regras de negócio

As regras de negócio são requisitos importantes para a especificação do comportamento do sistema, uma vez que definem como o sistema deve se comportar em determinadas situações. Conforme destaca Pressman (2016), "as regras de negócio são declarações precisas de restrições, condições ou comportamentos de negócios que o sistema deve ser capaz de suportar". Essas regras podem ser definidas por várias partes interessadas, incluindo clientes, usuários finais, analistas de negócios e especialistas do domínio.

Para este trabalho, foram definidas as seguintes regras de negócio:

Quadro 3 - Regras de negócio

Identificação	Requisito
RN001	O sistema deve permitir que apenas administradores do sistema criem pools de investimento entre dois tokens
RN002	O sistema não deve permitir a alteração de uma posição em vigor
RN003	O sistema não deve permitir saques parciais, tanto de tokens negociados e não negociados
RN004	O sistema não deve permitir que posições sejam manejadas por outros usuários, se não o criador da posição
RN005	Não devem ser permitidas chamadas para a função de negociação de tokens, caso não haja token a serem negociados
RN006	Não devem ser permitidas chamadas para a função de negociação de tokens, caso a diferença entre o *timestamp* da última compra e o timestamp atual seja menor que o tempo mínimo entre execuções estipulado na criação da pool
RN007	Qualquer usuário deve poder chamar a função de negociação de tokens e ganhar uma taxa de 1% do valor total da troca

Fonte: Autoria própria, 2023

4.3 PROTÓTIPOS DE TELA

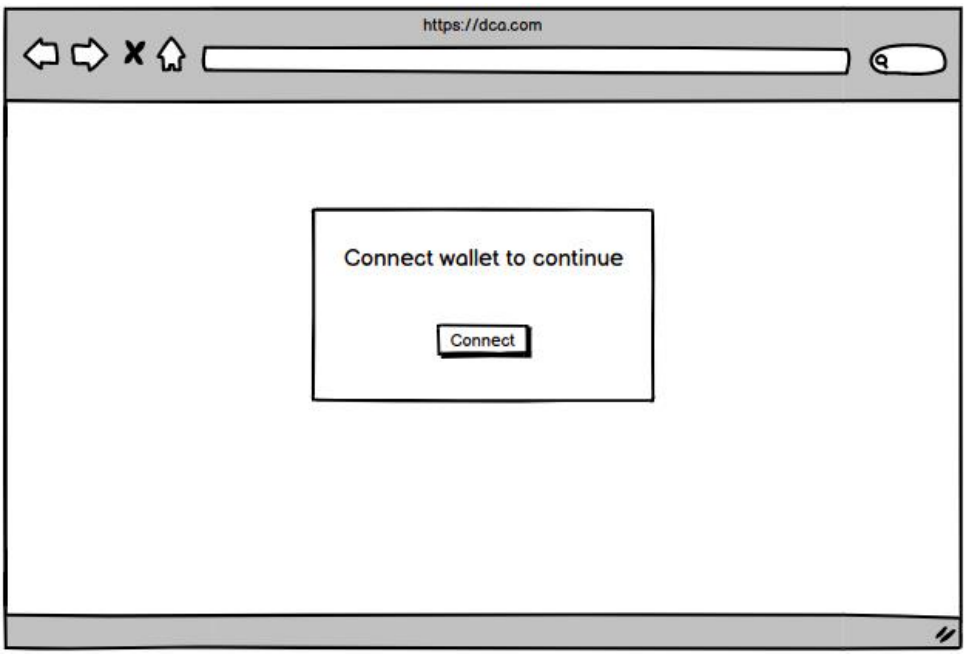
Os protótipos de tela são uma etapa importante do processo de desenvolvimento de software, permitindo que sejam criados modelos visuais das interfaces que serão utilizadas pelos usuários. Nesta seção, são apresentados os protótipos de tela desenvolvidos para o sistema *Dollar-Cost Averaging*. Esses protótipos são fundamentais para validar as decisões de design e

garantir a usabilidade do sistema. Conforme destacam Constantine e Lockwood (2002), protótipos de tela ajudam a antecipar possíveis problemas e aprimorar a experiência do usuário, reduzindo custos e tempo de desenvolvimento.

Neste capítulo são apresentados os protótipos de tela criados para guiar o desenvolvimento da interface gráfica.

Inicialmente, o usuário deverá conectar sua carteira de criptomoedas que seja compatível com *blockchains* que implementem a EVM, tal tela é representada pela Figura 3.

Figura 3 - Tela de conexão com a carteira



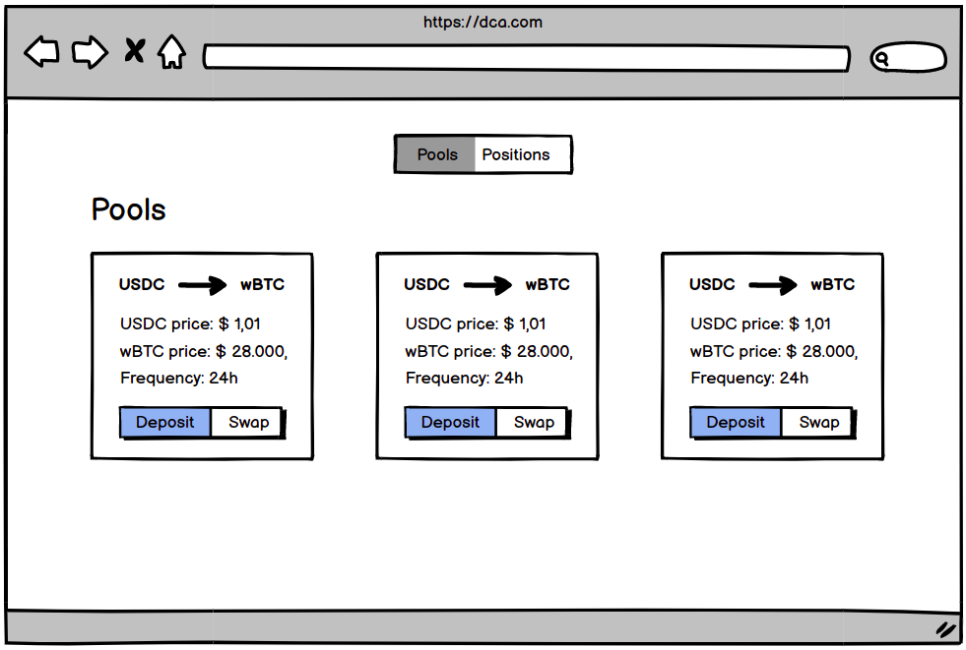
Fonte: Autoria própria, 2023

A Figura 4 representa a tela inicial do sistema, onde o usuário poderá ver todas as opções de estratégias disponíveis para investimento.

Nesta tela, para cada estratégia de investimento oferecida, tem-se duas possíveis ações, *deposit* e *swap.* Ao clicar em "*Deposit*" um modal com as informações necessárias para realizar a ação deve ser apresentado ao usuário. E ao clicar em "*Swap*", a interface deve chamar a função de troca de tokens para todas as posições da pool em questão e recompensar o usuário por ter tomado tal ação

O botão *"Swap"* deverá permanecer desabilitado conforme a condição descrita na regra de negócio RN006.

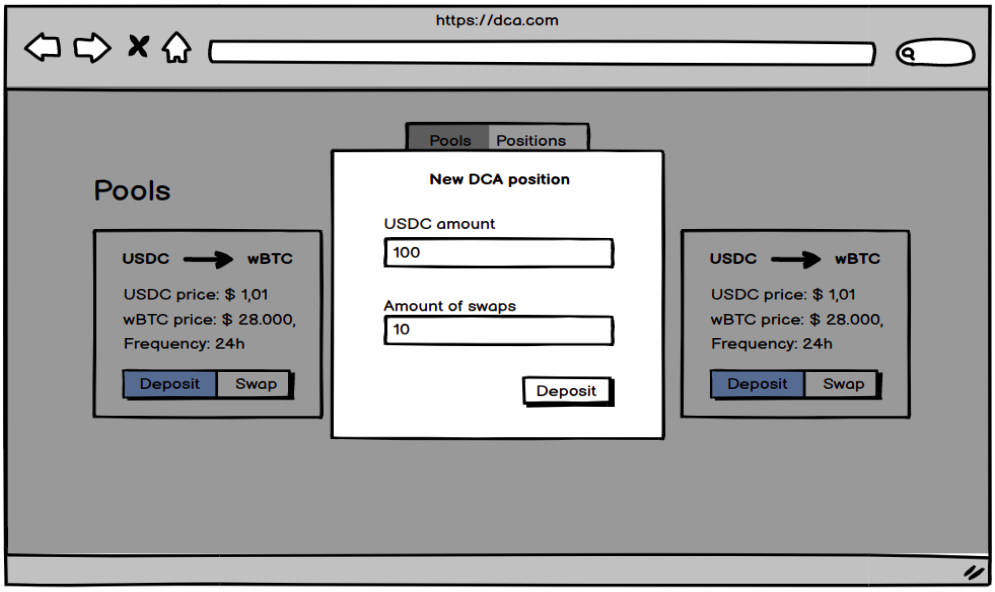
Figura 4 - Protótipo da listagem de pools



Fonte: Autoria própria, 2023

Como descrito anteriormente e representado na Figura 5, ao clicar em "*Deposit*", um modal de depósito deve ser apresentado ao usuário, para que ele possa inserir os detalhes da posição de investimento que gostaria de abrir. O usuário pode escolher a quantidade do token de entrada que será depositado e em quantas negociações de token a quantidade escolhida deverá ser negociada.

Figura 5 - Protótipo de criação de nova posição de investimento

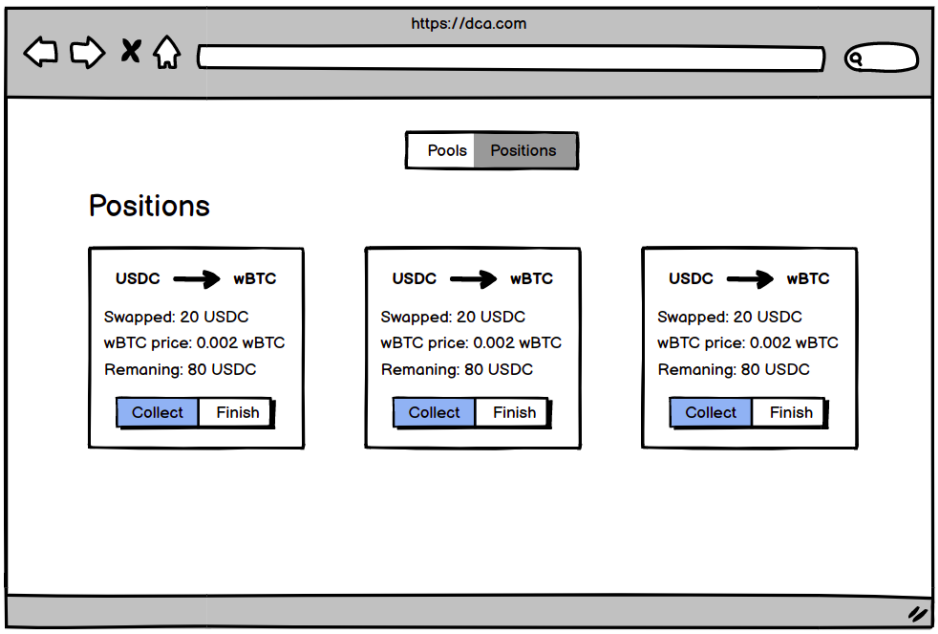


Fonte: Autoria própria, 2023

A Figura 6 representa a prototipação do requisito funcional RF006, que permite ao usuário visualizar todas as posições de investimento em execução.

Para cada posição, duas ações são disponibilizadas aos usuários, *Collect* e *Finish*, que são as implementações dos requisitos funcionais RF004 e RF005, respectivamente.

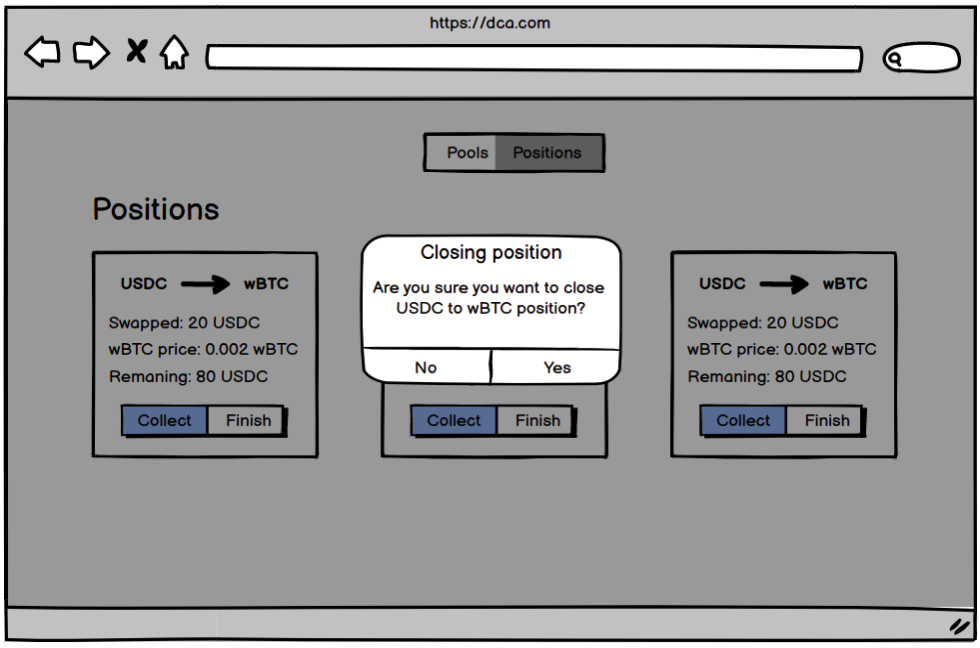
Figura 6 - Protótipo da listagem de posições do usuário



Fonte: Autoria própria, 2023

Ao clicar em "*Finish*", um modal de confirmação é apresentado ao usuário, tendo em vista que esta é uma ação de saída do sistema, como é apresentado pela Figura 7.

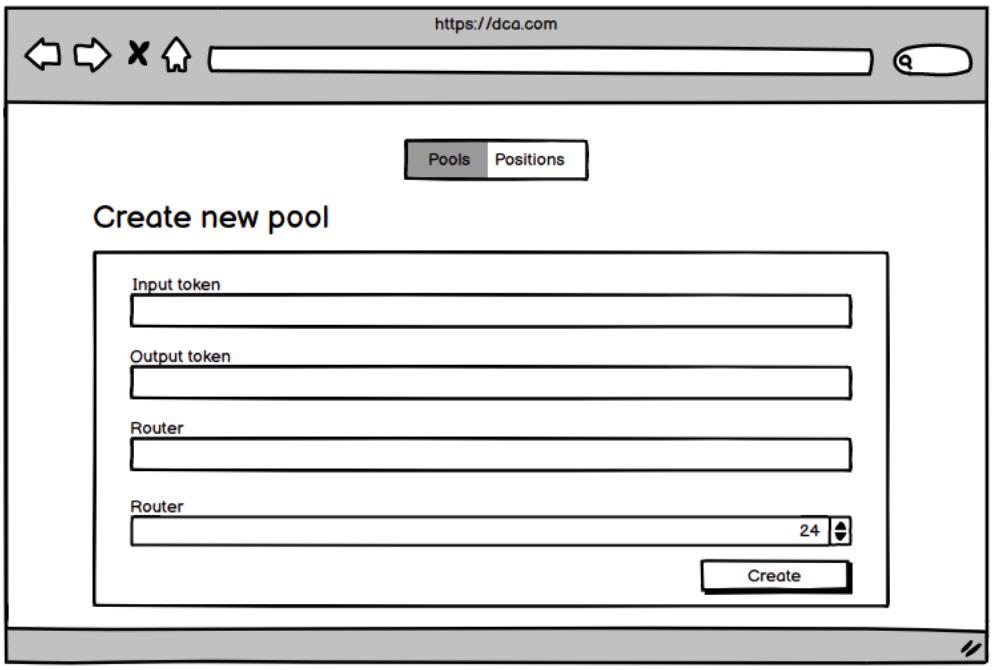
Figura 7 - Protótipo de confirmação de encerramento de posição



Fonte: Autoria própria, 2023

A Figura 8 representa a regra de negócio RN001 e o requisito funcional RF011, que exigem que apenas administradores do sistema possam criar novas *pools* de investimento

Figura 8 - Protótipo de criação de novas pools de investimento por administradores



Fonte: Autoria própria, 2023

4.4 CASOS DE USO

Os casos de uso são uma técnica amplamente utilizada na elicitação e especificação de requisitos de software. Segundo Cockburn (2000), um caso de uso representa uma interação entre o usuário e o sistema que produz um resultado observável e valioso para o usuário. Essa técnica é amplamente utilizada em projetos de desenvolvimento de software para capturar e comunicar as necessidades e expectativas dos usuários em relação ao sistema.

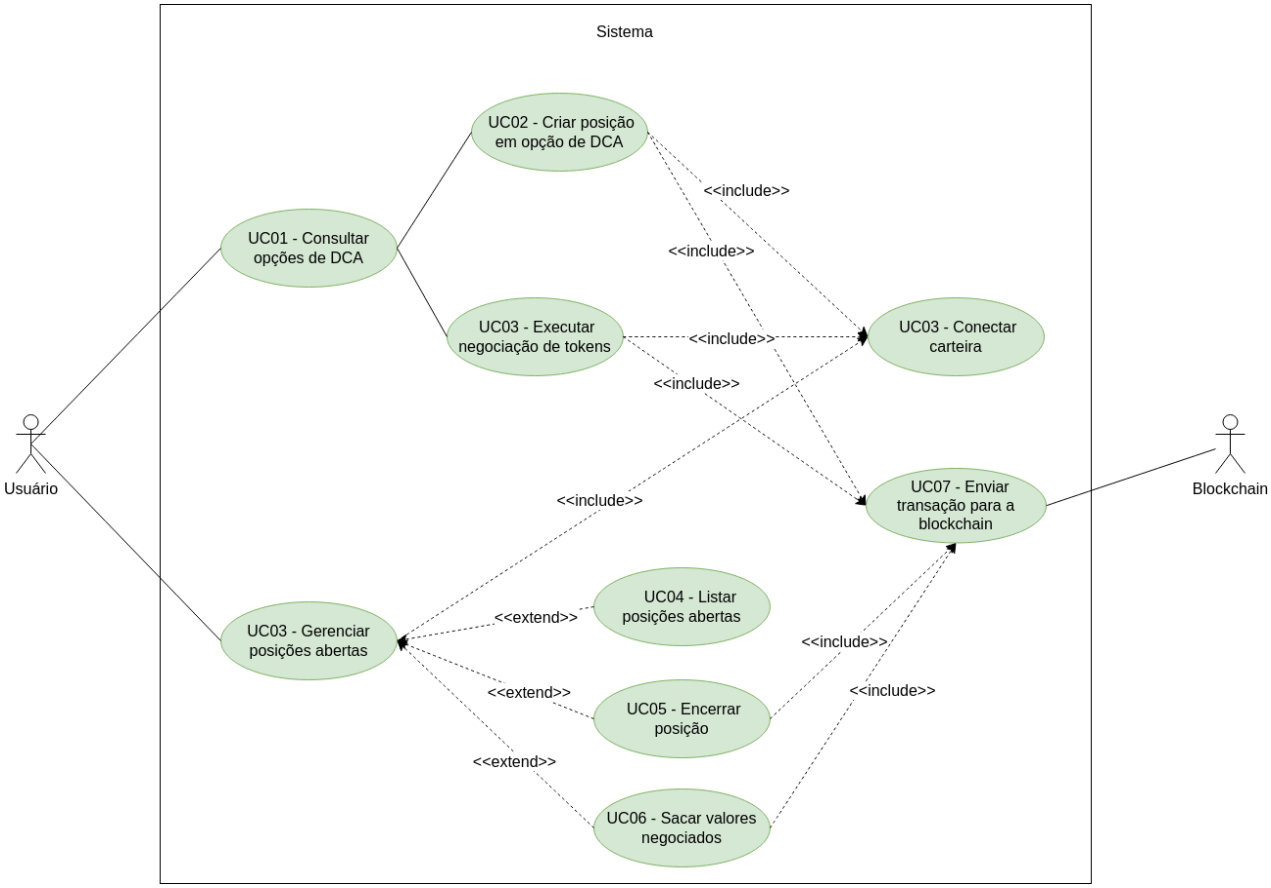
No contexto deste trabalho, os casos de uso são importantes para a identificação dos principais fluxos de interação entre o usuário, interface e *smart-contracts* que implementam a estratégia de *Dollar-coat Averaging*. Como destacam Larman e Basili (2003), os casos de uso permitem descrever as principais funcionalidades do sistema de forma clara e objetiva, definindo os principais requisitos de usuário e identificando as principais interações com o sistema. A partir dos casos de uso é possível também definir cenários de teste, que são essenciais para garantir a qualidade e a confiabilidade do sistema.

Assim, a subseção de casos de uso deste trabalho apresenta os principais casos de uso identificados para o *smart-contract* de *Dollar-coat averaging*, descrevendo as principais funcionalidades do sistema e as principais interações com o usuário.

O primeiro caso de uso é representado pela Figura 9, que ilustra os casos de uso de um usuário não administrador do sistema.

No quadro a seguir são apresentados os casos de uso para a listagem de opções de investimentos, identificado no diagrama de casos de uso como UC01.

Figura 9 - Caso de uso usuário não administrador



Fonte: Autoria própria, 2023

Quadro 4 - Caso de uso UC01

UC01 - Consultar opções de DCA
Descrição: Ao acessar o sistema, ator Usuário deve ser apresentado a todas as opções de investimentos disponíveis naquele momento, sendo visíveis o preço dos tokens, e a razão dos preços entre o token de entrada e de saída.
Pré-condições: --
Pós-condições: Poder interagir com as opções de investimentos disponíveis e ter as informações dos tokens
Requisitos funcionais: RF001, RF008, RF007
Fluxo principal: **Passo 1:** O ator usuário acessa a tela inicial do sistema. **Passo 2:** O ator recebe a listagem de opções de investimento disponíveis. **Passo 3**: O ator escolhe a ação desejada. **Passo 4:** O sistema realiza a ação desejada.
Fluxo alternativo A: **Passo 1:** O ator usuário escolhe uma opção de investimento de DCA para fazer um aporte e

clica em “*Deposit*”. **Passo 2:** O sistema apresenta ao usuário um modal onde este deve preencher a quantidade de tokens a serem depositados e por quantas iterações o montante total a ser depositado deve ser negociado. **Passo 3:** O ator usuário assina a transação que lhe é apresentada e a aplicação encaminha a transação para a blockchain. **Passo 4:** A transação é bem-sucedida e o usuário é redirecionado para a página de listagem de posições.
Fluxo alternativo B: **Passo 1:** O ator usuário escolhe uma opção de investimento de DCA para fazer um aporte e clica em “*Deposit*”. **Passo 2:** O sistema apresenta ao usuário um modal onde este deve preencher a quantidade de tokens a serem depositados e por quantas iterações o montante total a ser depositado deve ser negociado. **Passo 3:** O ator usuário assina a transação que lhe é apresentada e a aplicação encaminha a transação para a blockchain. **Passo 4:** A transação é malsucedida e é mostrado um aviso ao usuário sobre o erro.
Fluxo alternativo C: **Passo 1:** O ator usuário escolhe uma das opções de investimento a qual ele gostaria de realizar a negociação dos tokens para receber as taxas da transação e clica em “*Swap*” **Passo 2:** O ator assina a transação apresentada a ele e a aplicação envia a transação para a blockchain. **Passo 3:** A transação é bem-sucedida e o ator usuário recebe 1% do valor da negociação dos tokens.
Fluxo alternativo D: **Passo 1:** O ator usuário escolhe uma das opções de investimento a qual ele gostaria de realizar a negociação dos tokens para receber as taxas da transação e clica em “*Swap*” **Passo 2:** O ator assina a transação apresentada a ele e a aplicação envia a transação para a blockchain. **Passo 3:** A transação é malsucedida e é mostrado um aviso ao usuário sobre o erro.

Fonte: Autoria própria, 2023

No próximo quadro, Quadro 5, são listados os casos de uso UC03, que trata do gerenciamento das posições abertas pelo usuário.

Quadro 5 - Caso de uso UC04

UC03 - Gerenciar posições abertas
Descrição: Com a carteira de criptomoedas conectada a aplicação deve ser disponibilizado ao usuário uma listagem das suas posições de investimento abertas e permitir o gerenciamento delas, excluindo posições finalizadas ou encerradas manualmente pelo usuário.
Pré-condições: Estar com a carteira de criptomoedas conectada com a aplicação e ter

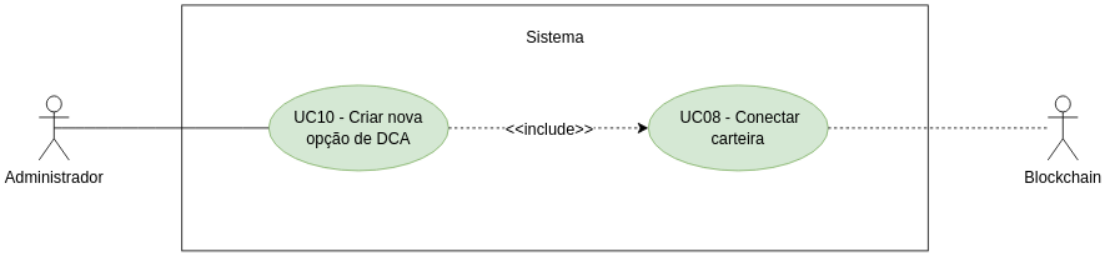
carregado a listagem de tokens.
Pós-condições: O ator usuário pode interagir e gerenciar suas posições de investimento abertas
Requisitos funcionais: RF004, RF005, RF006, RF007, RF009
Fluxo principal: **Passo 1:** O ator usuário acessa a tela de gerenciamento de posições. **Passo 2:** O ator recebe a listagem de posições abertas. **Passo 3**: O ator escolhe a ação desejada. **Passo 4:** O sistema realiza a ação desejada.
Fluxo alternativo A **Passo 1:** O ator usuário escolhe uma posição que gostaria de sacar valores negociados e clica em "*Collect*". **Passo 2:** O ator usuário assina a transação apresentada a ele e a aplicação envia a transação para a blockchain. **Passo 3:** A transação foi bem-sucedida e a listagem das posições abertas e atualizada com o novo valor disponível para ser coletado pelo usuário.
Fluxo alternativo B **Passo 1:** O ator usuário escolhe uma posição que gostaria de sacar valores negociados e clica em "Collect". **Passo 2:** O ator assina a transação apresentada a ele e a aplicação envia a transação para a blockchain. **Passo 3:** A transação é malsucedida e é mostrado um aviso ao usuário sobre o erro
Fluxo alternativo C **Passo 1:** O ator usuário escolhe uma posição que deseja encerrar e clica em finish **Passo 2:** O ator usuário assina a transação que lhe é apresentada e a aplicação encaminha a transação para a blockchain **Passo 3:** A transação e validada pela blockchain e a listagem de posições é atualizada, com apenas as posições em vigor do usuário
Fluxo alternativo D **Passo 1:** O ator usuário escolhe uma posição que deseja encerrar e clica em finish **Passo 2:** O ator usuário assina a transação que lhe é apresentada e a aplicação encaminha a transação para a blockchain **Passo 3:** A transação é malsucedida e é mostrado um aviso ao usuário sobre o erro

Fonte: Autoria própria, 2023

O segundo caso de uso é representado pela Figura 10, que ilustra a possível ação de um usuário administrador do sistema.

O Quadro 6 apresenta os casos de uso para um usuário administrador da aplicação representando o caso de uso UC10.

Figura 10 - Caso de uso usuário não administrador



Fonte: Autoria própria, 2023

Quadro 6 - Caso de uso UC10

UC10 - Criar nova opção de DCA
Descrição: Para atores administradores, o sistema deve disponibilizar um formulário de criação de novas pools de investimento de DCA
Pré-condições: Estar com a carteira conectado com a aplicação e ser administrador da aplicação
Pós-condições: --
Requisitos funcionais: RF011
Fluxo principal: **Passo 1:** O administrador acessa a tela de criação de opções de investimento. **Passo 2:** O administrador deve preencher os campos *input token*, *output token*, *router*, *path* e *interval* **Passo 3**: O administrador assina a transação que lhe é apresentada e a aplicação encaminha a transação para a blockchain **Passo 4:** A transação é validada pela blockchain e o ator é redirecionado para a listagem inicial do sistema, onde ele poderá comprovar a criação da nova opção de investimento.
Fluxo alternativo A: **Passo 1:** O administrador acessa a tela de criação de opções de investimento. **Passo 2:** O administrador deve preencher os campos *input token*, *output token*, *router*, *path* e *interval* **Passo 3**: O administrador assina a transação que lhe é apresentada e a aplicação encaminha a transação para a blockchain **Passo 4:** A transação é mal-sucedida e o ator recebe um aviso sobre o seu resultado.

Fonte: Autoria própria, 2023

4.5 DIAGRAMA DE CLASSES

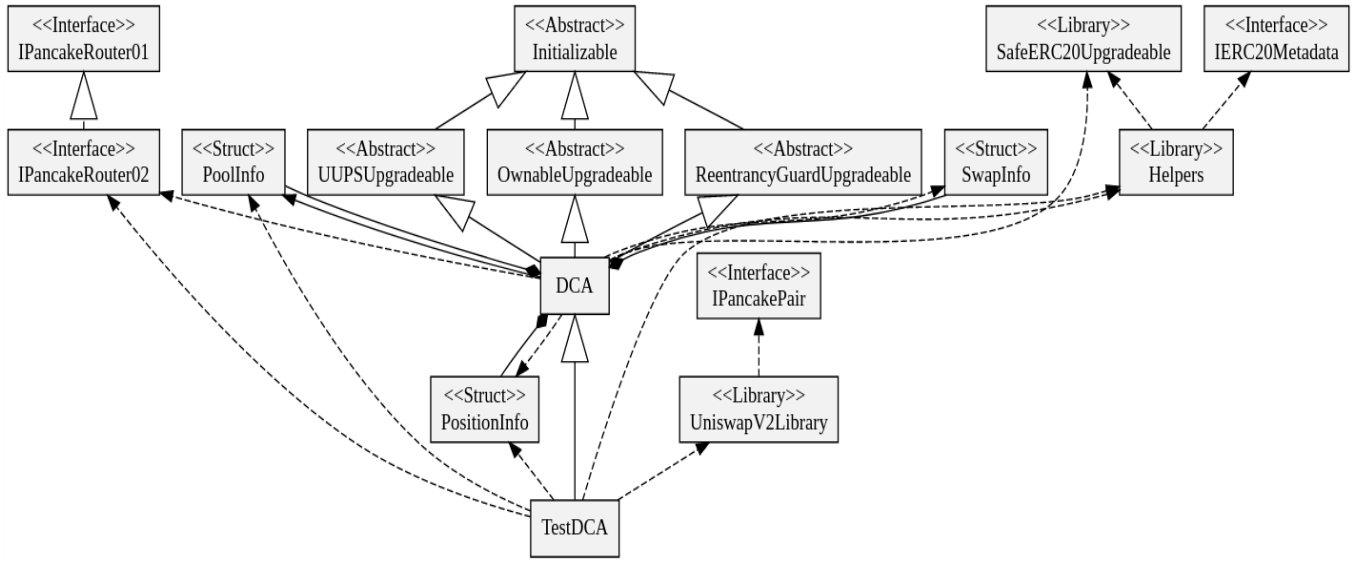
De acordo com Ambler (2002), o Diagrama de Classes é um dos diagramas mais importantes da UML, pois fornece uma visão geral da estrutura do sistema e é usado como base

para a criação de outros diagramas da UML. Além disso, o Diagrama de Classes é fundamental para a comunicação entre os membros da equipe de desenvolvimento, pois permite que todos tenham uma compreensão clara da estrutura do sistema.

No contexto deste trabalho, o Diagrama de Classes é utilizado para representar a estrutura do *smart-contract* que implementa a estratégia de *Dollar-cost averaging*. Através desse diagrama, é possível visualizar as classes e seus relacionamentos, o que é essencial para o desenvolvimento do software.

A Figura 11 apresenta o diagrama de classes para o contrato de DCA.

Figura 11 - Diagrama de Classes



Fonte: Autoria própria, 2023

4.6 DIAGRAMA DE COMPONENTES

Conforme destacado por Bass et al. (2003), o diagrama de componentes permite identificar os componentes individuais do sistema, suas interfaces e as dependências entre eles. Essa representação gráfica auxilia no entendimento da estrutura e na visualização das relações entre os componentes, facilitando o processo de design e desenvolvimento do sistema.

Na Figura 12 temos o diagrama de componentes desenvolvido por esse trabalho, nele podemos identificar dois principais pilares do sistema, a web e a blockchain.

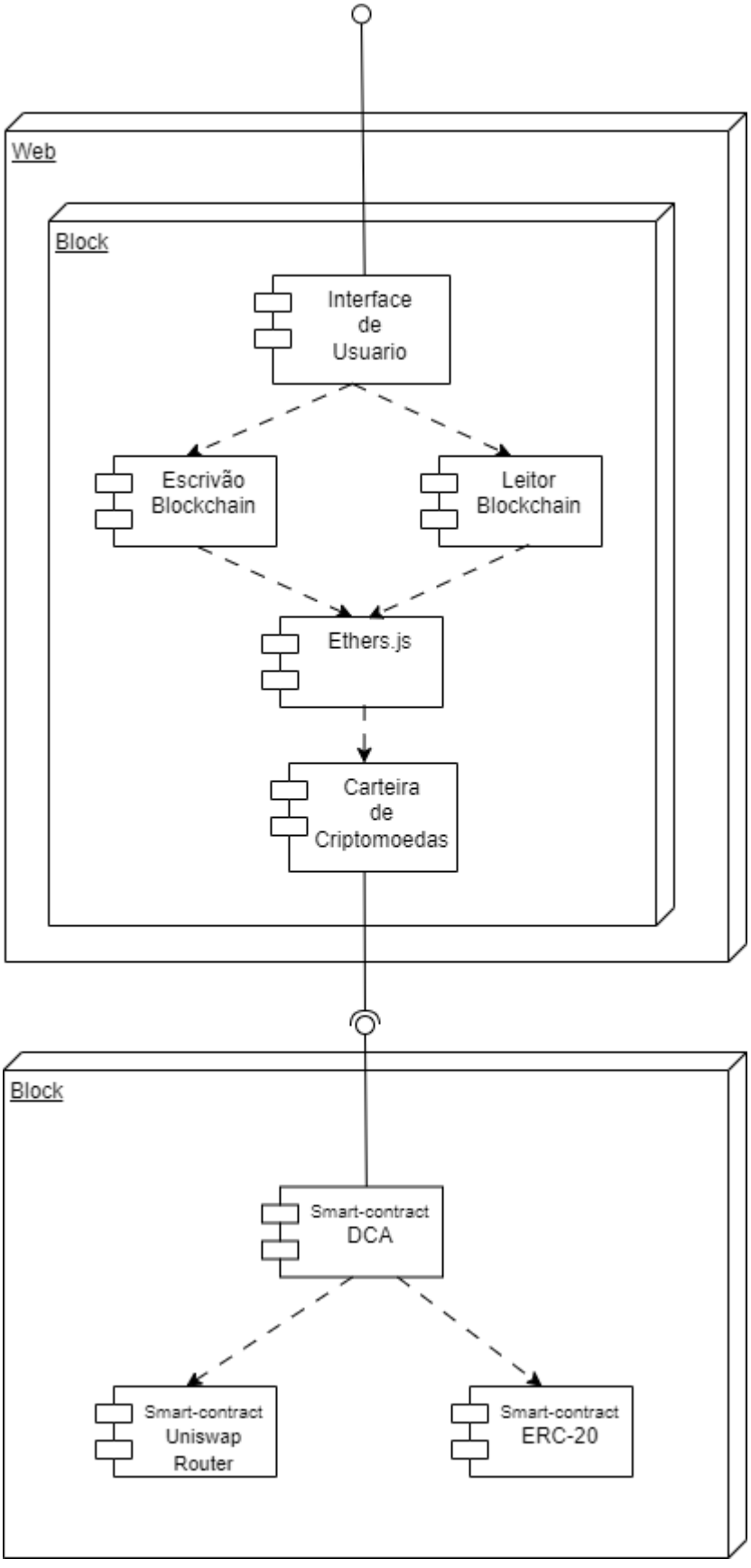
A seção da aplicação que será executada na web, junto de um navegador tem os seguintes componentes:

- **Interface de usuário:** camada da aplicação que será responsável por colher e apresentar todas as informações aos usuários de forma gráfica.
- **Leitor blockchain:** camada da aplicação responsável por requerer as informações necessárias da blockchain sobre as estratégias de investimento, posições e tokens
- **Escrivão blockchain**: camada da aplicação responsável por enviar requisições de escrita a blockchain.
- **Ethers.js:** biblioteca Javascript encarregada de abstrair as chamadas a blockchain em uma API alto nível para simplificar o desenvolvimento da aplicação
- **Carteira de Criptomoedas:** responsável por interagir com os nodos da blockchain para enviar e requisitar informações da cadeia de blocos.

Já seção da aplicação que é executada dentro dos limites da blockchain, conta com os seguintes componentes:

- **Smart-contract DCA:** o *smart-contract* responsável por guardar todas as informações de pools de DCA, de posições de usuários, e executar as ações de saque, término de posição, deposito e criação de novas opções de investimento
- **Smart-contract ERC-20:** são os contratos que guardam informações dos tokens ERC-20 que serão utilizadas para registros dentro da *smart-contract* DCA.
- **Smart-contract Uniswap Router:** contratos baseados na tecnologia de rotas criada pela Uniswap, que executam as trocam entre quaisquer dois tokens que sigam o padrão ERC-20.

Figura 12 - Diagrama de componentes



Fonte: Autoria própria, 2023

4.7 MODELAGEM DE INCENTIVOS

A modelagem de incentivos é uma abordagem comumente utilizada para o design de sistemas baseados em *blockchain* que busca alinhar os interesses dos participantes e incentivar comportamentos desejáveis por meio de recompensas e punições. Como destacam Swan (2015) e Antonopoulos e Wood (2018), a modelagem de incentivos é essencial para a segurança e o bom funcionamento de sistemas descentralizados, já que incentiva os participantes a seguir as regras do sistema e a contribuir para sua operação.

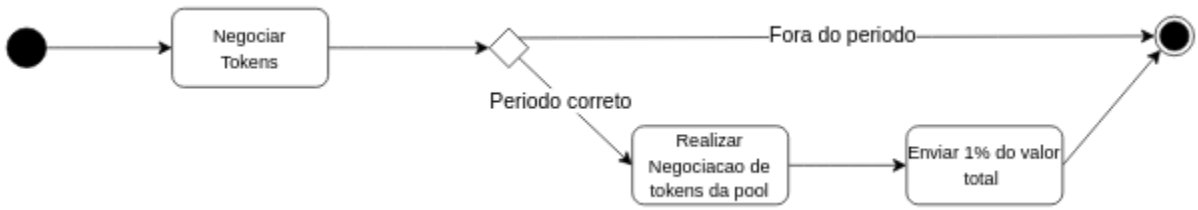
Para o contexto deste trabalho, a modelagem de incentivos é uma peça fundamental para o bom funcionamento da aplicação desenvolvida, uma vez que depende que usuários interajam com o *smart-contract*, chamando a função de negociação de tokens, para que as estratégias de DCA sejam executadas.

O primeiro incentivo a ser introduzido aos usuários é o fator de poder realizar suas transações de negociação de dois tokens sem que precisem pagar as taxas de transação de rede, tendo em vista que essa negociação acontece uma única vez para todos os usuários que fizeram deposito em uma pool de DCA.

O segundo incentivo diz sobre para a execução das negociações de tokens no período estipulado pela pool de investimento. Ao ser criada uma pool de DCA, o administrador do sistema pode definir o intervalo de tempo que a função de negociação de tokens poderá ser invocada. Para que a aplicação não dependa apenas de alternativas centralizadas de automação de código em *blockchain*, apresentadas anteriormente no corpo deste trabalho, toda vez que um usuário invocar a função de negociação através da interface web ou pelo próprio *smart-contract*, após o intervalo estipulado na criação da pool, o sistema o recompensa com 1% do valor negociado pela pool.

A seguir, a Figura 13 representa um diagrama de atividade que ilustra a recompensa dada aos usuários

Figura 13 - Diagrama de atividades



Fonte: Autoria própria, 2023

Tal incentivo cresce ao passo que o tamanho das pools do sistema crescem, uma vez que os usuários recebem, uma porcentagem sobre um valor maior de tokens a ser negociado. Uma vez que é esperado que o aumento de incentivo gere o aumento do esforço por parte dos usuários para invocar a função de negociação de tokens para cada pool no momento correto, criando assim uma aplicação mais segura e confiável.

Da mesma forma que são definidos incentivos, punições por comportamentos indesejados também devem ser criadas. A única punição em prática no sistema será para os usuários que chamem a função de negociação de tokens no momento fora do período estipulado pelo, onde estes terão que pagar a taxa de transação de *blockchain*, sem receber nenhuma recompensa, tendo assim uma perda de capital.

5 DESENVOLVIMENTO

Neste capítulo são apresentadas as ferramentas e tecnologias utilizadas para o desenvolvimento da aplicação, as etapas aplicadas na construção do artefato de saída e a apresentação da aplicação a partir de sua interface gráfica. Por fim, ainda neste capítulo é explorada a avaliação qualitativa do software em questão, de forma a validar as premissas e ideias abordadas na fundamentação teórica.

5.1 FERRAMENTAS E TECNOLOGIAS

Para o desenvolvimento deste trabalho foram escolhidas ferramentas e tecnologias com que o autor estivesse familiarizado, de forma a aumentar a qualidade de entrega do artefato final, uma vez que era de conhecimento prévio deste mesmo autor uma série de técnicas e padrões que podem ser aplicados utilizando tais artefatos.

A seguir são apresentadas as ferramentas e tecnologias:

5.1.1 Solidity

Solidity é uma linguagem de programação de alto nível, orientada a contratos, com a sintaxe parecida com JavaScript e desenhada para ser executada na Máquina Virtual Ethereum (EVM). Solidity é estatisticamente tipada, suporta herança, bibliotecas e tipos complexos definidos pelo usuário entre outras características**.** (SOLIDITY, 2023).

Apesar de existirem outras linguagens em que se pode escrever smart-contract, como Vyper, Solidity se destaca pela grande quantidade de bibliotecas e recursos quando comparada a seus competidores, desta forma, sendo a escolhida para a execução deste trabalho.

5.1.2 Hardhat

O Hardhat é um ambiente de desenvolvimento para software Ethereum. Ele consiste em diferentes componentes para edição, compilação, depuração e implantação de seus contratos inteligentes e dApps, todos os que trabalham juntos para criar um ambiente de desenvolvimento completo. (HARDHAT, 2023).

Para o contexto deste trabalho, o Hardhat foi utilizado para escrever um conjunto de testes unitários para o *smart-contract* desenvolvido, assim como os scripts de implantação, responsáveis por lançar o contrato na *blockchain* escolhida.

Dentre as demais opções de ferramentas que disponibilizam funcionalidades parecidas, na avaliação do autor, Hardhat possui o melhor suporte nativo a TypeScript, integração com bibliotecas populares e personalização do ambiente de desenvolvimento.

5.1.3 Vue

Vue é um framework progressivo para a construção de interfaces de usuário. Ao contrário de outros frameworks monolíticos, Vue foi projetado desde sua concepção para ser adotável incrementalmente. A biblioteca principal é focada exclusivamente na camada visual (view layer), sendo fácil adotar e integrar com outras bibliotecas ou projetos existentes. Por outro lado, Vue também é perfeitamente capaz de dar poder a sofisticadas Single-Page Applications quando usado em conjunto com ferramentas modernas e bibliotecas de apoio. (VUE, 2023).

O framework foi escolhido pela simplicidade em desenvolver interfaces robustas. Enquanto frameworks como React e Angular, na avaliação do autor, se distanciam das tecnologias nativas da web, criando suas próprias sintaxes.

5.1.4 Ethers.js

A biblioteca ethers.js tem como objetivo ser uma biblioteca completa e compacta para interagir com a *blockchain* Ethereum e seu ecossistema. Ela foi originalmente projetada para uso com ethers.io e desde então se expandiu para se tornar uma biblioteca de propósito mais geral. (ETHERS, 2023).

No contexto do software desenvolvido, Ethers.js é usado em conjunto com Hardhat, tanto para executar os testes dos *smart-contracts* que serão rodados em uma *blockchain* emulada localmente, quanto para interagir com uma *blockchain* de produção para implantação dos contratos. Ele também é usado dentro da interface de usuário para se comunicar com a extensão da carteira de criptomoedas do usuário para que as transações possam ser enviadas à *blockchain*.

5.1.5 Typescript

TypeScript é uma linguagem de programação de código aberto desenvolvida pela Microsoft. É um superconjunto sintático estrito de JavaScript e adiciona tipagem estática opcional à linguagem. Tipos fornecem uma maneira de descrever a forma de um objeto,

fornecendo melhor documentação e permitindo que o TypeScript valide se seu código está funcionando corretamente. (WIKIPEDIA, 2022).

O TypeScript é parte em integrante na maioria dos artefatos de saída deste trabalho, sendo utilizado para os testes e implantação do *smart-contract*, bem como para a interface de usuário, e foi escolhido para compor esse trabalho pela segurança que traz ao desenvolvedor através da checagem estática de tipos, fundamental para evitar erros em tempo de execução.

5.1.6 Polygon Mumbai

A Mumbai é o testnet da rede Polygon, que replica a mainnet da Polygon. Ele permite que os desenvolvedores implementem, testem e executem seus dApps no ambiente *blockchain* sem riscos e sem custos. Assim como a Polygon, que foi lançada em 2017, o Mumbai também utiliza o mecanismo de consenso proof-of-stake (PoS) para concordar com o estado do *blockchain*. (ALCHEMY, 2022).

A aplicação foi desenvolvida utilizando o ecossistema da rede Polygon, porém esta decisão é arbitrária, e unicamente escolhida pela familiaridade do autor com tal *blockchain*. Assim como foi escolhida a rede Polygon, poderia ter sido escolhida qualquer outra *blockchain* EVM, que todas as instâncias da aplicação estariam preparadas para tal alteração.

5.1.7 MetaMask

As carteiras de criptomoedas ajudam a gerenciar permissões sobre com quem você compartilha seus dados, armazenam e enviam criptomoedas, NFTs e muito mais. Sua carteira de criptomoedas contém uma chave privada que identifica e avalia os ativos que são seus. Dessa forma, você pode pensar em uma carteira como o MetaMask como um sistema de gerenciamento de identidade digital. (METAMASK, 2023).

MetaMask foi escolhida como a carteira de criptomoedas a qual a aplicação desenvolvida suportaria primariamente, por ser na visão do autor a carteira com a maior parcela de mercado.

5.1.8 Balsamiq Wireframes

Balsamiq Wireframes é uma ferramenta de criação de wireframes de IU (Interface do Usuário) rápida e de baixa fidelidade que reproduz a experiência de fazer um esboço em um bloco de notas ou quadro branco, mas usando um computador. Ela realmente faz com que você

se concentre na estrutura e no conteúdo, evitando discussões longas sobre cores e detalhes que devem ser abordados posteriormente no processo. (BALSAMIQ, 2023).

Como a própria descrição criada pela própria empresa, Balsamiq Wireframes foi escolhido pela praticidade e simplicidade que a ferramenta oferece para a criação de wireframes. Outras ferramentas similares foram consideradas, porém todas ofereciam wireframes muito detalhados, o que fugia do propósito deste trabalho.

5.2 HISTÓRICO DO DESENVOLVIMENTO

O processo de desenvolvimento da aplicação aconteceu dentro de três principais passos para que se pudesse alcançar o objetivo proposto por este trabalho: Modelagem do sistema, desenvolvimento dos *smart-contracts* e desenvolvimento da interface.

A seguir serão detalhados tais processos.

- **Modelagem do sistema**

O processo de modelagem do sistema incluiu os subprocessos de desenvolvimento dos protótipos de tela, requisitos funcionais, requisitos não funcionais, regras de negócio, identificação dos casos de uso e modelagem dos incentivos.

Os requisitos e regras de negócio foram elaborados através do conhecimento prévio do autor sobre sistemas web que interagem com *blockchains*, assim como através de pesquisa aprofundada nas limitações intrínsecas de aplicações desenvolvidas a partir de blockchains.

Os protótipos de tela foram desenvolvidos através do reconhecimento de práticas comuns em aplicações que interagem com a blockchain, como Uniswap, PancakeSwap, Beefy, Aave, Curve entre outras. Destaca-se entre estas práticas, o modelo de conexão de carteira, que permite que os usuários interajam com a aplicação normalmente, até que a conexão da carteira seja imprescindível para dar continuidade a ação do usuário.

A identificação dos casos de uso foi feita através da intersecção das regras de negócio, requisitos e protótipos de telas. Com estas informações dispostas, foi feita a análise e reconhecimento dos possíveis fluxos tomados pelos usuários dentro da aplicação.

Dentro da modelagem de incentivos foram definidas as estratégias que incentivem usuários a invocar as funções de negociação de tokens, criando assim um sistema automatizado através de ações tomadas por usuários interessados nas recompensas geradas por tais ações. O incentivo principal para o sistema funcionar adequadamente, o pagamento de 1% do valor total da pool para o usuário que invocar a função de swap, pode não atrair usuários para pools que

tenham montantes baixos de tokens depositados, uma vez que o usuário só terá lucros uma vez que a porcentagem paga por seus serviços ultrapasse a taxa de transação, logo, constatou-se inicialmente a necessidade de subsídio das transações por parte dos administradores do sistema.

- **Desenvolvimento dos smart-contracts**

Como citado anteriormente, blockchains baseadas na tecnologia do ecossistema Ethereum, possui um limite de opcodes, as instruções para a máquina virtual da Ethereum, que podem ser executadas por bloco minerado. Tal limitação veio a se tornar o maior percalço no desenvolvimento dos smart-contracts e da aplicação como um todo.

Solidity mune desenvolvedores com estruturas de dados nativas como *structs*, *mapping* (análogo à *hashmaps*), e *arrays* dinâmicos. Tais estruturas fornecem o ferramental suficiente para produzir as coleções de controle e implementação. A Figura 14 representa uma possível implementação didática e simplificada do contrato de DCA.

Figura 14 - Implementação simplificada de um contrato de DCA

```
contract DCA {
  struct Pool {
    uint256 id;
    address inputToken;
    address outputToken;
  }

  struct UserPosition {
    uint256 inputTokenBalance;
    uint256 outputTokenBalance;
  }

  mapping(address => mapping(uint => UserPosition)) public userPositions;
  Pool[] public pools;

  function createPosition(uint poolId, uint inputTokenAmount) external {   infinite gas
    userPositions[msg.sender][poolId] = UserPosition ({
      inputTokenBalance: inputTokenAmount,
      outputTokenBalance: 0
    });
  }

  function executeTransactions(address[] calldata users, uint256 poolId) external {   772 gas
    for (uint256 i = 0; i > users.length; i++) {
      UserPosition memory position = userPositions[users[i]][poolId];
      Pool memory pool = pools[poolId];

      swap(position, pool);
    }
  }
```

Fonte: Autoria própria, 2023

A maior falha desta implementação reside no método *executeTransactions*, que neste exemplo é responsável por coordenar a troca de tokens. A função recebe um array de endereços de usuários, itera através destes e invoca a função *swap*, que trocara os tokens em nome dos usuários.

Porém, conforme a quantidade de posições de usuários cresce, também cresce o número de iterações, logo o número de instruções executadas pela máquina virtual da Ethereum. Mas como visto anteriormente, esta solução esbarra no limite de *gas* por bloco, que a depender da quantidade de posições existentes, podem ser necessárias múltiplas transações, o que eventualmente pode se mostrar financeiramente inviável pelos custos intrínsecos de uma operação na blockchain. Logo, nota-se que se trata de uma solução não escalável para o problema proposto.

Através de pesquisas em contratos inteligentes implantados em diversas redes e algoritmos aplicados a smart-contracts, chegou-se a uma solução que permite todas as trocas de uma determinada *pool* de DCA serem realizadas em uma única transação, através do acúmulo de todo o valor a ser negociado por aquela pool em uma única variável de controle. A informação de controle guarda o montante do token de entrada a ser negociado em cada das iterações da pool e faz parte do conjunto de informações de uma pool, que. Uma iteração é caracterizada pela execução da função do contrato de negociação de tokens.

Ao invocar a função de criação de posição, o usuário deve enviar como parâmetro o identificador da pool a qual se deseja fazer parte, a quantidade de tokens a ser depositada inicialmente e a quantidade de iterações pelas quais se deseja trocar os tokens.

Como descrito anteriormente e representado na Figura 14, deve-se acrescer o valor sendo depositado à variável de controle da pool, portanto é aferido o valor a ser negociado em nome do usuário em cada uma das iterações, através da razão entre a quantidade depositada e o número de iterações, como observado na definição da variável *rate*. A variável da pool, *nextSwapAmount*, portanto é incrementada com o valor de *rate*.

É necessário manter também um registro do *rate* de posições que se encerram na presente iteração, para que desta forma, ao ser invocada a função de negociação de tokens, o valor das próximas negociações seja atualizado de forma a remover os valores do montante total de tokens de entrada. Com isso foi criada a variável *_poolDelta*, responsável por registrar a quantidade total a ser subtraída em cada uma das iterações. Logo, é preciso que seja somado o *rate* calculado deste depósito ao montante subtraído do valor de *nextSwapAmount* na iteração *finalSwap*, que consiste no número atual de operações de troca executadas somado a quantidade estabelecida pelo usuário de *swaps* o qual gostaria de participar.

Figura 15 - Implementação da função de criação de posição

```
function createPosition(uint _pid, uint _amount, uint _swaps) external {
    require(_pid < poolLength(), "DCA:: invalid pid");
    require(_amount > 0, "DCA:: invalid amount");
    require(_swaps > 0, "DCA:: invalid swaps");

    PoolInfo storage pool = poolInfo[_pid];

    IERC20Upgradeable(pool.inputToken).safeTransferFrom(msg.sender, address(this), _amount);

    uint rate = _amount / _swaps;
    uint finalSwap = pool.performedSwaps + _swaps;

    pool.nextSwapAmount += rate;
    _poolDelta[_pid][finalSwap + 1] += rate;

    positionInfo[msg.sender].push(
        PositionInfo({
            amount: _amount,
            swaps: _swaps,
            rate: rate,
            pid: _pid,
            finalSwap: finalSwap,
            firstSwap: pool.performedSwaps,
            lastUpdatedAt: pool.performedSwaps
        })
    );

    emit CreatePosition(msg.sender, _pid, _swaps, rate, finalSwap, pool.performedSwaps);
}
```

Fonte: Autoria própria, 2023

- **Desenvolvimento da interface de usuário**

O desenvolvimento da interface de usuário se deu sem maiores desafios, tendo em vista a familiaridade do autor com tal tipo de tarefa. Contudo, algumas informações idealizadas anteriormente no processo de criação dos protótipos de tela foram retrabalhadas para entregar mais contexto aos usuários sobre suas posições de investimento e sobre as *pools* disponíveis.

Conforme apresentado anteriormente a interface foi codificada usando Vue.js, se valendo das abstrações oferecidas pela biblioteca Ethers.js para comunicação entre agentes externos, nesse caso a interface, e a *blockchain*.

A arquitetura apresentada anteriormente para os serviços de leitura e escrita da *blockchain*, apresentada anteriormente, foi substituída por uma única camada de leitura e escrita sob o mesmo serviço. Assim foi decidido pelo autor, por este entender que tanto as camadas de leitura e escrita utilizam o mesmo canal de comunicação com a *blockchain*.

Foi utilizada uma abordagem para o desenvolvimento da interface que visa facilitar a integração de qualquer outra *blockchain* que rode sob os padrões definidos pela EVM. Portanto

se existe uma necessidade futura de integração com demais *blockchains*, o impacto a nível de código seria mínimo.

Figura 16 - Implementação da função swap

```
function swap(SwapInfo[] calldata swapInfo) external nonReentrant {
    for (uint32 i = 0; i < swapInfo.length; i++) {
        uint pid = swapInfo[i].pid;
        PoolInfo storage pool = poolInfo[pid];

        require(block.timestamp >= pool.lastSwapAt + pool.interval, "DCA:: too early");
        require(pool.nextSwapAmount > 0, "DCA:: nextSwapAmount must be greater than zero");

        Helpers.approveIfNeeded(pool.inputToken, pool.router, type(uint).max);

        uint[] memory amountsOut = IPancakeRouter02(pool.router).swapExactTokensForTokens(
            pool.nextSwapAmount,
            swapInfo[i].minOutputAmount,
            pool.path,
            address(this),
            block.timestamp
        );

        uint outputTokenAmount = amountsOut[amountsOut.length - 1];

        uint fee = _calculateFee(outputTokenAmount);

        IERC20Upgradeable(pool.outputToken).transfer(msg.sender, fee);

        uint amountOut = amountsOut[amountsOut.length - 1];

        emit Swap(pid, pool.nextSwapAmount, amountOut, fee);

        uint ratio = _calculateRatio(pool.nextSwapAmount, outputTokenAmount);

        pool.performedSwaps += 1;
        pool.nextSwapAmount -= _poolDelta[pid][pool.performedSwaps + 1];
        pool.lastSwapAt = block.timestamp;

        _accumulatedRatio[pid][pool.performedSwaps] = _accumulatedRatio[pid][pool.performedSwaps - 1] + ratio;
    }
}
```

Fonte: Autoria própria, 2023

5.3 APRESENTAÇÃO DA APLICAÇÃO

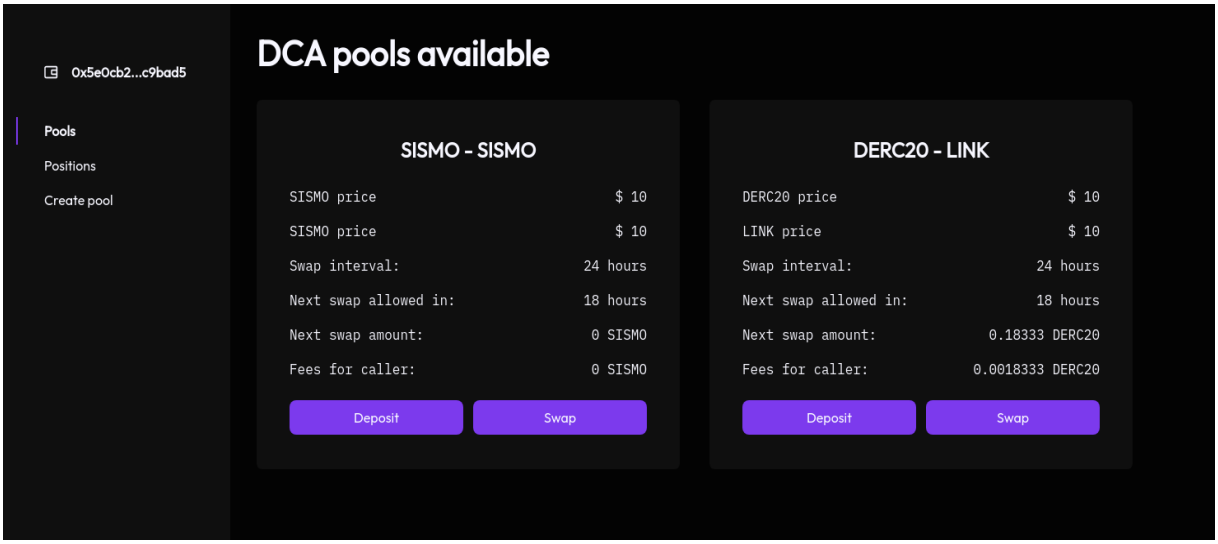
Nesta seção é apresentada a aplicação a partir da sua interface gráfica. São detalhadas as funcionalidades presentes em cada uma das telas e onde foi necessário a implementação divergir dos protótipos de tela criados.

A tela inicial do sistema consiste na listagem das opções de investimento, ou seja, todas as *pools* de DCA criadas pelo administrador do sistema. Cada uma das *pools* é representada por um card, contendo as informações de preços de cada um todos tokens, o intervalo mínimo de espera entre cada negociação de tokens, o tempo restante até que a função de negociação possa ser invocada novamente, o volume de tokens a serem negociados na próxima iteração, e a quantidade estimada de taxas a serem recebidas para o usuário que invocar a função *swap* para aquela opção de investimento.

Além das informações exibidas, cada *card* conta com os botões de chamada para ação de depósito e negociação de tokens. Ao clicar em "*Swap*", a função de negociação de tokens, uma transação é enviada diretamente a blockchain, já a função de depósito é descrita a seguir

A tela inicial do sistema é apresentada pela Figura 17.

Figura 17 - Tela inicial do sistema: listagem de opções de investimento

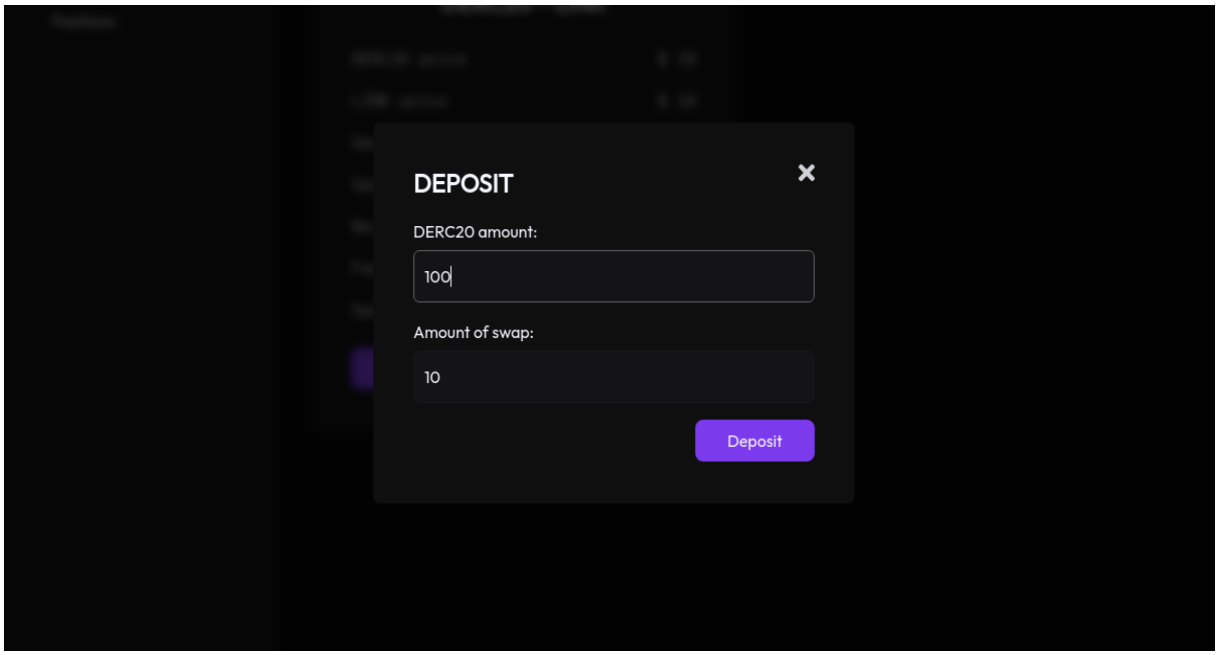


Fonte: Autoria própria, 2023

Para realizar um depósito, o usuário deve clicar no botão "*Deposit*", no canto inferior esquerdo do *card* que representa a opção de investimento escolhida. Tal ação abrirá um modal de depósito, onde o usuário poderá especificar a quantidade de tokens a ser utilizada na sua nova posição, assim como a quantidade de iterações pelas quais aqueles tokens deverão ser negociados.

A Figura 18 apresenta o resultado da ação descrita acima.

Figura 18 - Modal de depósito



Fonte: Autoria própria, 2023

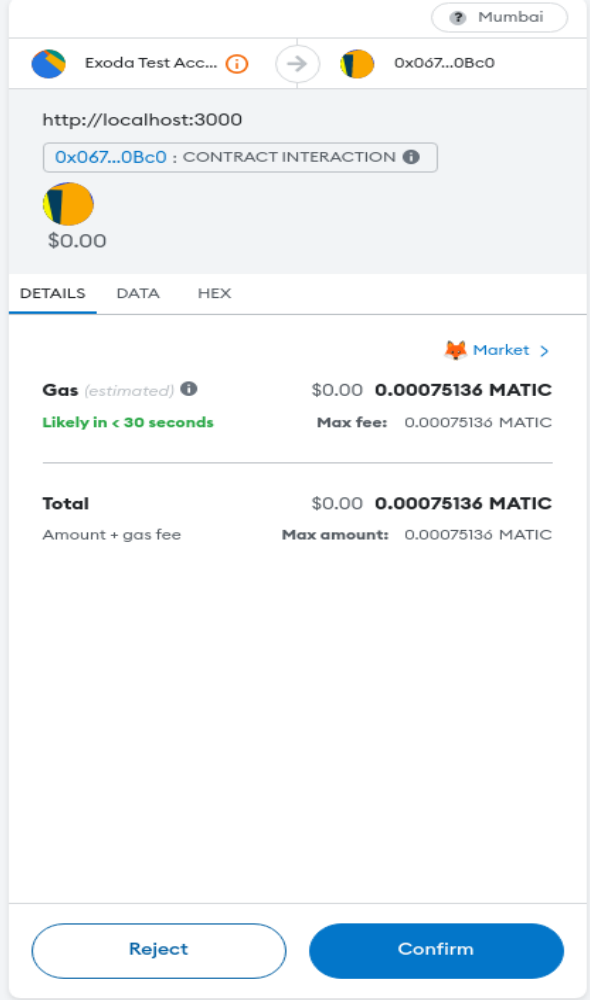
Uma vez que o usuário confirme as informações de depósito, um botão de confirmação da ação fica disponível no canto inferior direito do modal. Ao clicar neste botão, uma aplicação envia as informações da transação que se pretende enviar a *blockchain*, para que o usuário confirme a transação dentro da extensão da carteira de criptomoedas, neste caso, como descrito anteriormente está sendo usada a carteira MetaMask.

Dentre as informações que são apresentadas pela carteira ao usuário estão os custos da transação, o endereço do *smart-contract* sendo invocado, a função do contrato a qual se deseja interagir com e os parâmetros passados para esta função. De tal maneira, o usuário pode confirmar que todas as informações passadas pela interface para o contrato, condizem com os dados informados pelo usuário.

Uma vez que a transação seja enviada e confirmada pela *blockchain*, ela ficará disponível para consulta dentro da carteira do usuário.

A Figura 19 apresenta a requisição de confirmação de transação exibida ao usuário para que a transação possa ser enviada ao *smart-contract* na *blockchain*.

Figura 19 - Confirmação de transação através da carteira MetaMask



Fonte: Autoria própria, 2023

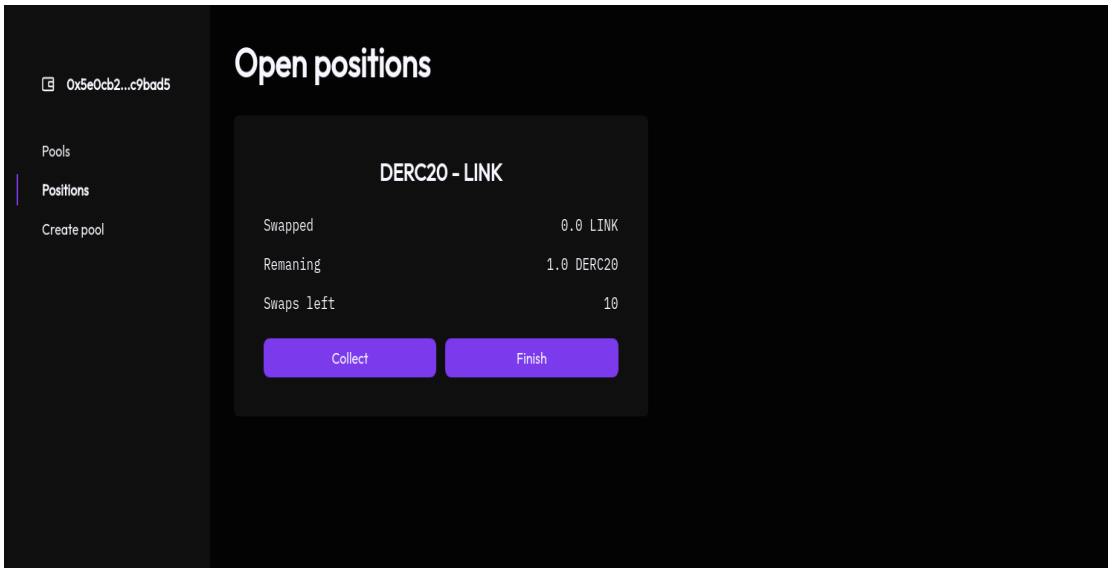
Uma vez que a transação for confirmada pela *blockchain*, uma confirmação visual é dada pela interface gráfica ao usuário, portanto este poderá gerenciar sua posição através da tela de gerenciamento de posições.

A Figura 21 mostra a implementação da página de gerenciamento de posições, onde o usuário pode consultar o estado de cada uma de suas posições de investimento em vigor. São apresentadas ao usuário as seguintes informações: quantidade já negociada (*Swapped*), quantidade que ainda resta a ser negociada (*Remaining*) e quantidade de iterações restantes (*Swaps left*).

Além das informações disponibilizadas, cada card de posição oferece duas ações aos usuários. A primeira é a ação de coletar os tokens já negociados, ou seja, sacar da sua posição os tokens de saída (*Collect*). A segunda é a ação de encerramento de posição, onde o usuário saca tanto os tokens já negociados quanto os tokens ainda não negociados e posição é

encerrada. Para cada uma dessas ações uma transação é enviada para a carteira do usuário a exemplo do descrito sobre a Figura 20.

Figura 20 - Página de gerenciamento de posições



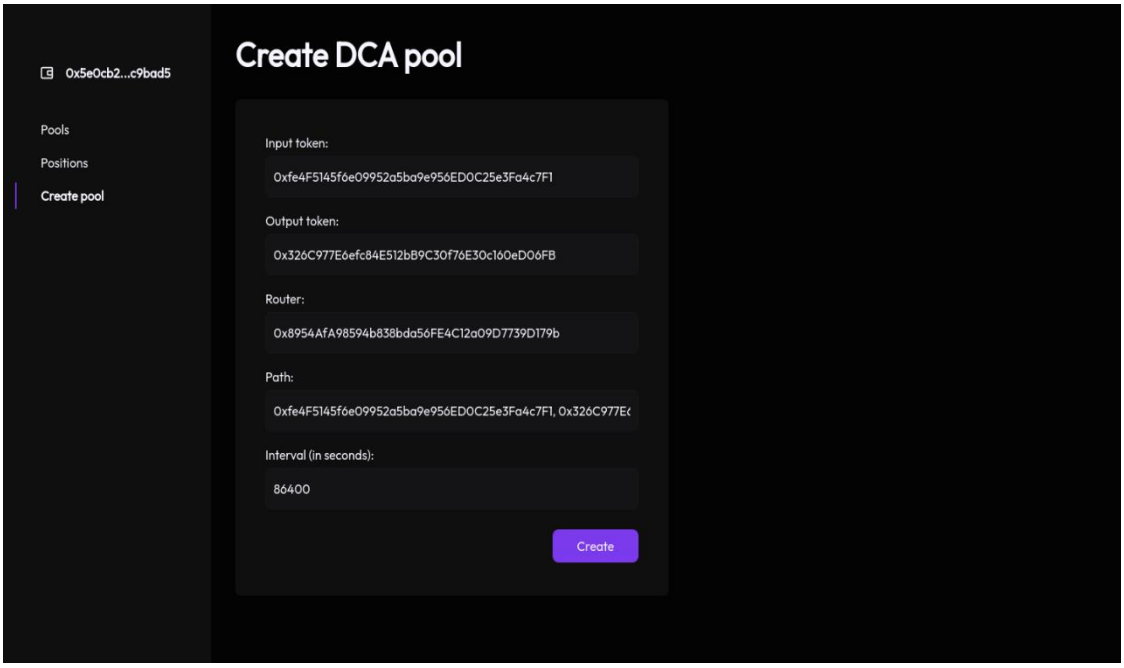
Fonte: Autoria própria, 2023

Por fim, a última tela a ser apresentada e a criação de pools de DCA, tela esta que está apenas disponível para administradores do sistema, por tanto será listada no menu lateral somente quando o sistema verificar junto ao *smart-contract* na blockchain que a carteira conectada pertence a um administrador. Nesta tela, como se vê na Figura 20, apresenta um formulário com todas as informações necessárias para que uma nova *pool* seja criada: endereço do token de entrada, endereço do token de saída, endereço do *router*, caminho que o *router* usara entre o swap dos dois tokens (path), e o intervalo em segundos entre as negociações poderão ser invocadas. O caminho entre os tokens a ser utilizado pelo *router* deve ser informado com uma vírgula separando estes endereços.

Novamente, uma vez que o administrador do sistema confirme as informações passadas para o formulário e clique em "*Create*", uma confirmação de transação será enviada à carteira para que o usuário possa conferir os detalhes da transação e confirmá-la ou não.

Ao final do processo de criação de *pool*, uma confirmação visual será enviada ao usuário por parte da interface e este poderá verificar a criação da nova opção de investimento na listagem presente na tela inicial do sistema.

Figura 21 - Formulário de criação de opção de investimento



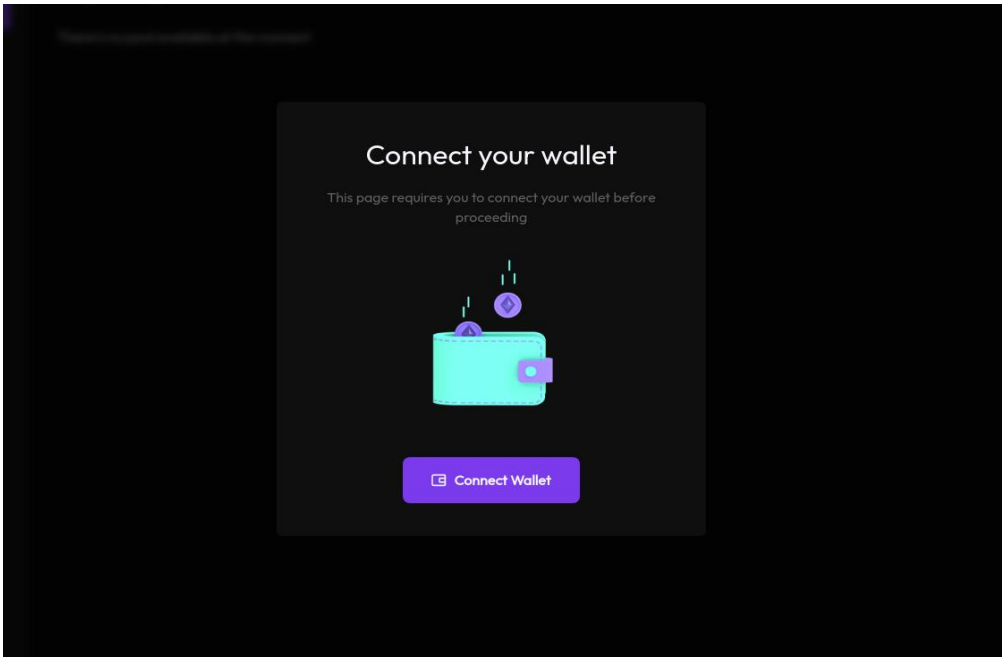
Fonte: Autoria própria, 2023

Antes de cada uma das telas do sistema que exigem conexão com a carteira de criptomoedas do usuário (gerenciamento de posições e criação de *pool*), verifica-se se o usuário está com a carteira conectada a aplicação, de forma que se a conexão não existir, um modal requisitando a conexão é exibido, de forma que ele não poderá ser dispensado pelo usuário, apenas será fechado quando a carteira estiver devidamente conectada a aplicação.

Uma vez que o usuário clicar em "*Connect Wallet*", uma requisição de conexão é enviada a carteira do usuário, que prontamente apresenta o pedido ao mesmo e este terá que confirmar ou negar a conexão da carteira com a aplicação. De forma que se o pedido for negado o sistema manterá o modal de conexão de carteira aberto e o usuário não poderá usar o recurso oferecido pela página que ele estiver tentando acessar.

A Figura 22 apresenta o modal de conexão com a carteira.

Figura 22 – Modal de conexão com a carteira



Fonte: Autoria própria, 2023

5.4 AVALIAÇÃO DA APLICAÇÃO

Nesta seção, é apresentada a avaliação do sistema. Essa avaliação foi conduzida por meio de um questionário aplicado a potenciais usuários, com o objetivo de verificar se o sistema desenvolvido atende às necessidades de investidores de criptomoedas.

O objetivo da aplicação do questionário é obter dados numéricos e descritivos sobre o protótipo desenvolvido, com o intuito de validar a aceitação do sistema e coletar sugestões para aprimoramentos.

Nas seções seguintes, são detalhadas as etapas necessárias para a avaliação do protótipo.

5.4.1 Cenário de avaliação

O sistema foi idealizado para investidores que atuam no mercado de criptomoedas e veem na estratégia de *Dollar-cost Average* como uma saída para combater a volatilidade nos preços dos ativos os quais estão sendo comprados. Desta forma, todos os participantes do questionário alegam ter ou já tiveram no passado exposição a criptomoedas em seu portfólio. O corpo de respondentes foi composto por 7 participantes.

O sistema foi apresentado através de um vídeo explicativo e demonstrativo de todas as funcionalidades desenvolvidas, desde a criação de *pools*, até o encerramento de uma posição.

5.4.2 Elaboração do questionário

A avaliação dos resultados de um trabalho é uma etapa crucial para determinar a eficácia e a satisfação dos usuários em relação ao sistema desenvolvido. Uma abordagem comumente utilizada para coletar informações sobre a experiência dos usuários é por meio de questionários. Nesta subseção, serão apresentados os questionários como uma ferramenta de avaliação e os benefícios que eles podem proporcionar.

Ao elaborar um questionário, é importante considerar aspectos como a formulação adequada das perguntas, a ordem das questões, a seleção do tipo de resposta (aberta, fechada, escala de Likert etc.) e a aplicação do questionário em si. Essas etapas devem ser cuidadosamente planejadas para garantir a qualidade e a confiabilidade dos resultados (Bryman, 2016).

Neste trabalho, o questionário foi utilizado como uma ferramenta para avaliar a experiência dos usuários em relação ao sistema que implementa a estratégia de DCA. Através das respostas dos participantes, foi possível obter informações sobre a usabilidade, a eficácia e a satisfação com o sistema, auxiliando na identificação de pontos fortes e áreas de melhoria.

As perguntas desenvolvidas para este questionário foram escolhidas de que pudessem ser avaliados os seguintes tópicos:

- Objetivos do trabalho
- Problemática do trabalho
- Usabilidade da aplicação
- Incentivos de uso
- Modelagem do projeto
- Melhorias futuras

Para elaborar as perguntas e a estrutura do questionário, foram levadas em consideração as ideias de Alreck e Settle (2004) que dizem ser essencial formular perguntas claras e objetivas, evitando ambiguidades e dupla negação. Além disso, é importante considerar a ordem das questões, começando com perguntas introdutórias fáceis e progressivamente avançando para questões mais complexas. Essa abordagem sequencial ajuda a criar um fluxo natural e a manter o interesse dos participantes. Quanto ao formato das respostas, Dillman et al. (2014) sugerem a utilização de escalas de Likert para medir atitudes e opiniões, bem como a inclusão de perguntas abertas para permitir que os participantes expressem suas ideias de forma mais livre.

Para as perguntas fechadas, onde os usuários respondem a cada uma das questões a partir de um universo finito de opções, foram escolhidas as seguintes possíveis respostas:

concordo totalmente, concordo parcialmente, indiferente, discordo parcialmente e discordo totalmente.

Também foram utilizadas duas perguntas fechadas, com respostas possíveis sim e não, para entender um pouco mais sobre a familiaridade de usuários com DCA e plataformas que oferecem esse serviço.

Uma pergunta de carácter semiaberto foi feita aos participantes, a qual questiona sobre a extensão de carteira de criptomoedas usada por estes, para validar a escolha da carteira escolhida para incorporar inicialmente a aplicação. Dentre as possíveis escolhas pré-definidas estavam: Metamask, Coinbase Wallet, Safepal, Wallet Connect, Trust Wallet. Além destas opções foi fornecido um campo "Outra", para que usuários registrem outras carteiras utilizadas que não estivessem presentes nessa lista.

Também foi utilizada uma pergunta de caráter aberto, onde o respondente pode expressar livremente sua opinião sobre os pontos fortes e fracos do sistema, de forma a que se perceba melhor onde estão os valores que o sistema está entregando ao usuário, o que ainda precisa ser trabalhado na plataforma.

O Quadro 7 apresenta as perguntas escolhidas para serem aplicadas aos respondentes da avaliação.

Quadro 7 - Perguntas selecionadas para o questionário

Pergunta	**Tipo de pergunta**
A estratégia de Dollar-cost averaging faz parte das suas ferramentas de investimento?	Fechada
O sistema fornece todas as informações necessárias para que um investidor tome a melhor decisão de investimento?	Fechada
Você estaria disposto a criar mecanismos para lembrá-lo de invocar a função de swap dentro do intervalo estabelecido pela pool	Fechada
Você estaria disposto a implementar mecanismos para acionar essa função de forma automática, caso as recompensas do sistema superassem o valor da transação?	Fechada
A aplicação pode contribuir para o crescimento do seu patrimônio.	Fechada
A descentralização é importante para você em aplicações envolvendo criptomoedas	Fechada
As taxas de 1% exercidas sobre o depósito do usuário seriam um impeditivo para o seu uso da plataforma?	Fechada

Você já utilizou uma plataforma de automatização de dollar-cost averaging para investimentos em geral	Fechada
Você já utilizou uma plataforma de automatização de dollar-cost averaging voltada para o mercado de criptomoedas totalmente on-chain?	Fechada
A aplicação possui uma interface de uso simples	Fechada
Qual carteira de criptomoedas você usa?	Semi-aberta
Você considera fazer o uso desta aplicação uma vez que ela venha a ser lançada em uma blockchain de produção.	Fechada
O processo de escolher uma pool e depositar o token de entrada se dá de maneira clara	Fechada
O processo de sacar os tokens já negociados é claro e simples?	Fechada
O processo de encerrar a posição é claro e simples?	Fechada
O processo de invocar a função de swap de uma pool é claro e simples?	Fechada
Na sua opinião, quais os pontos fortes e fracos da aplicação apresentada?	Aberta

Fonte: Autoria própria, 2023

5.4.3 Aplicação do questionário

A coleta de dados por meio do questionário proporcionou uma análise dos resultados que permitiu avaliar se o sistema desenvolvido alcançou seus objetivos iniciais e atendeu às expectativas estabelecidas. Para fornecer uma visão abrangente do sistema aos avaliadores, a aplicação do questionário seguiu uma série de etapas, incluindo:

- Demonstração da aplicação e suas funcionalidades.
- Realização do questionário.

Foram fornecidos dois vídeos aos participantes da avaliação. O primeiro visava apresentar a proposta de solução do trabalho e apresentar o conceito *de Dollar-cost Averaging* para usuários que não estivessem familiarizados com a estratégia de investimento. O segundo vídeo disponibilizado foi a apresentação do sistema, suas telas e funcionalidades.

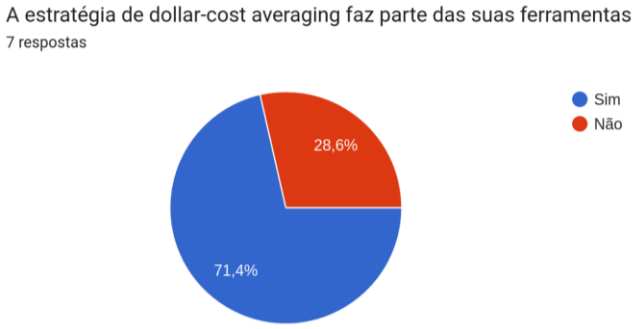
A aplicação do questionário foi feita utilizando a ferramenta *Google Forms*, pela praticidade oferecida pela plataforma tanto de implantação quanto de análise dos resultados.

5.4.4 Análise dos resultados

Nesta seção são apresentados os resultados da aplicação do questionário avaliativo, assim como uma breve comentários sobre dados produzidos pelas respostas dos questionários, para que se possa fazer a interpretação total da avaliação dos resultados desta pesquisa no capítulo seguinte.

A seguir são apresentadas os obtidos para cada umas das perguntas do questionário

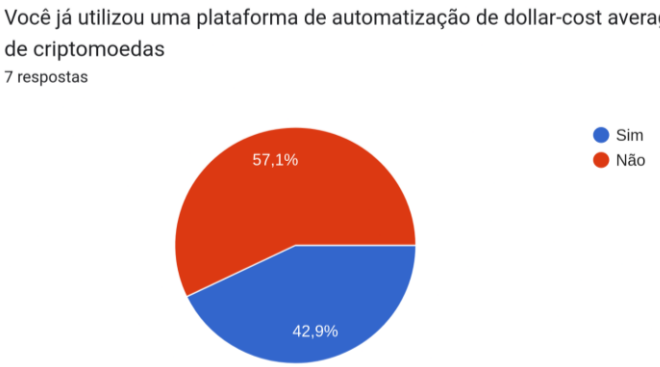
Figura 23 - Questionário avaliativo: resultados pergunta 1



Fonte: Autoria própria, 2023

A Figura 23 mostra os resultados da primeira pergunta do questionário que revela que grande parte dos entrevistados utiliza a técnica de DCA para investir.

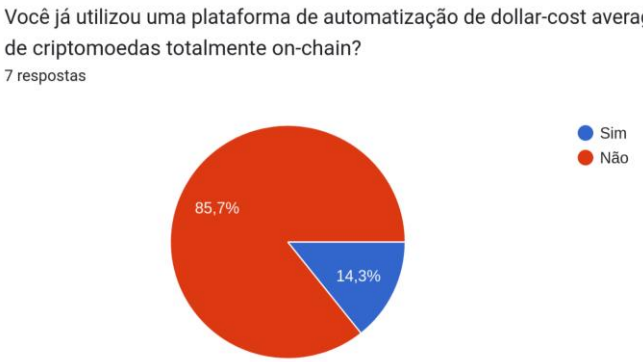
Figura 24 - Questionário avaliativo: resultados pergunta 2



Fonte: Autoria própria, 2023

A Figura 24 apresenta os resultados da segunda pergunta do questionário, indicando um equilíbrio entre usuários que já utilizaram ferramentas de automação para estratégias de DCA de compra de criptomoedas e usuários que as executam manualmente.

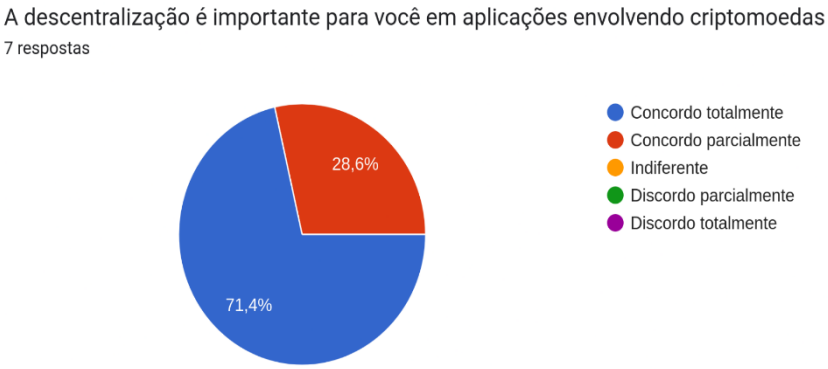
Figura 25 - Questionário avaliativo: resultados pergunta 3



Fonte: Autoria própria, 2023

A Figura 25 exibe o gráfico correspondente à pergunta 3. Onde apenas um respondente indica que já utilizou uma ferramenta de automatização de DCA com código apenas sendo executado na *blockchain*, sem depender de serviços centralizados. Uma falha identificada durante a avaliação das respostas foi a não existência de um campo aberto para o respondente indicar qual a plataforma utilizada.

Figura 26 - Questionário avaliativo: resultados pergunta 4



Fonte: Autoria própria, 2023

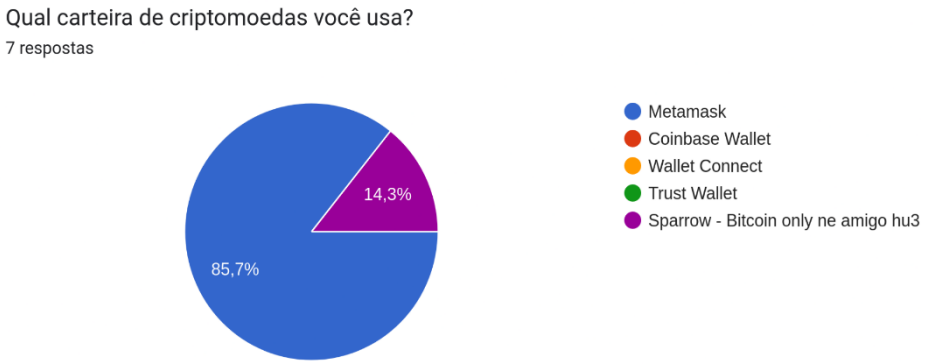
Na Figura 26 temos o resultado das da pergunta 4, que indica que para todos os participantes a descentralização da execução da aplicação é de alguma maneira importante para aplicações envolvendo criptomoedas.

Figura 27 - Questionário avaliativo: resultados pergunta 5

Fonte: Autoria própria, 2023

A Figura 27 mostra que todos os participantes do questionário acreditam que de alguma forma a aplicação desenvolvida pode contribuir para o crescimento do patrimônio deles.

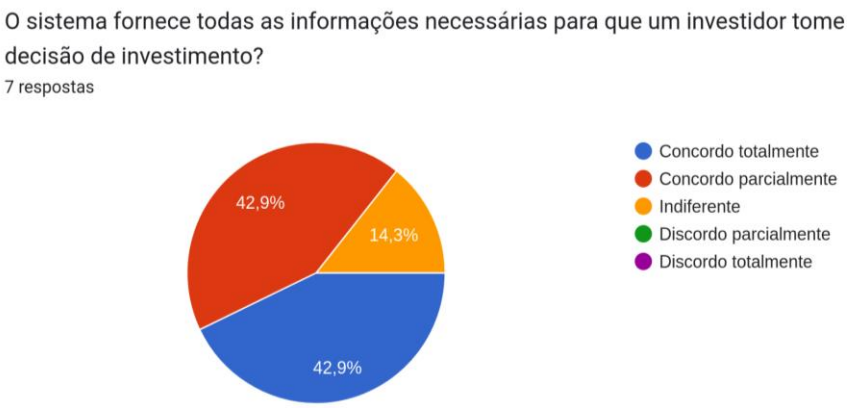
Figura 28 - Questionário avaliativo: resultados pergunta 6



Fonte: Autoria própria, 2023

Como podemos constatar pela Figura 28, a carteira mais utilizada entre os entrevistados é a MetaMask, carteira escolhida pelo autor para prover suporte inicial. Como podemos notar, apenas um respondente utilizou o campo “Outros” para descrever a sua carteira, Sparrow, que se trata de uma carteira de criptomoedas destinada apenas para a rede Bitcoin, que não faz parte do corpo deste trabalho. Apesar de apenas uma das respostas fugir do tema do trabalho, foi constatada a necessidade da especificação de carteira de criptomoedas apenas para redes compatíveis com a *Ethereum Virtual Machine*.

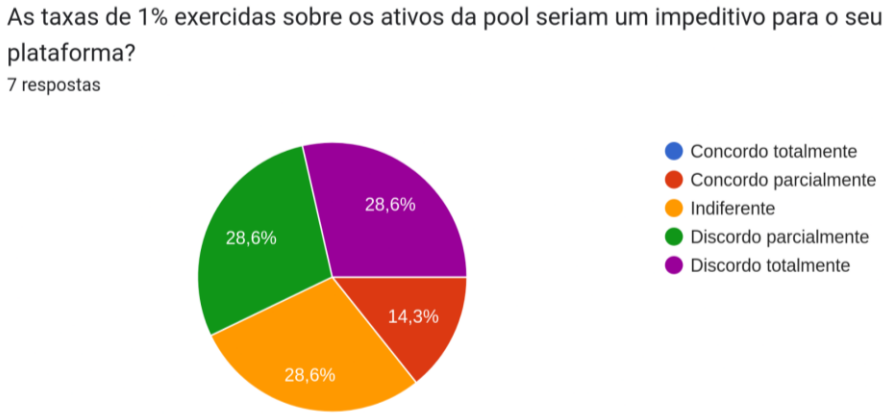
Figura 29 - Questionário avaliativo: resultados pergunta 7



Fonte: Autoria própria, 2023

A Figura 29 nos apresenta um cenário onde para a maioria dos usuários as informações apresentadas pelo sistema são suficientes para que estes possam tomar suas decisões de investimento. Apenas um participante se diz indiferente às informações disponibilizadas pela plataforma.

Figura 30 - Questionário avaliativo: resultados pergunta 8



Fonte: Autoria própria, 2023

Os dados apresentados pela Figura 30 mostram que a maior parte participantes estariam dispostos a utilizar a plataforma mesmo com as taxas cobradas pelas operações das *pools*, uma vez que tivemos dois participantes concordam parcialmente, dois concordando totalmente e dois sendo indiferentes as taxas. Apenas um respondente demonstrou um sentimento negativo pelas taxas praticadas pela plataforma.

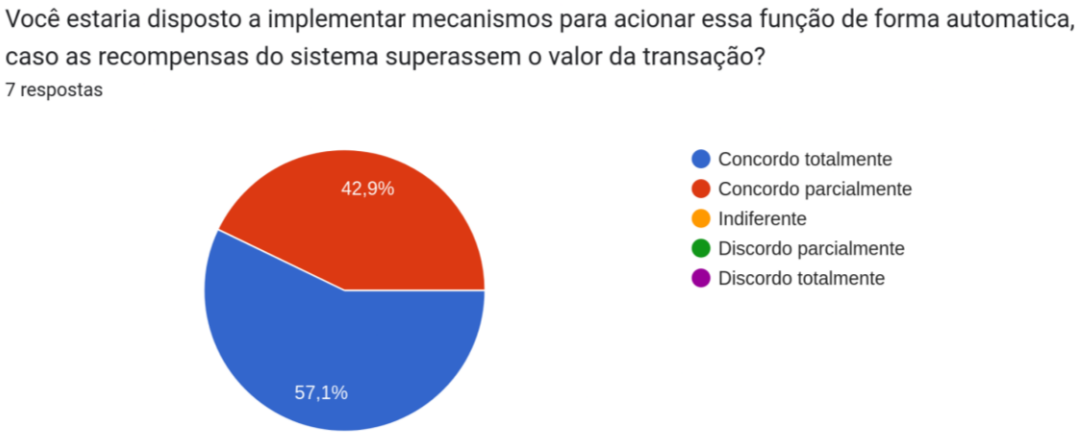
A Figura 31 apresenta as respostas dos participantes quando perguntados se estes estariam dispostos a criar mecanismos para lembrá-los de invocar a função de negociação de tokens dentro do intervalo estabelecido na criação da pool.

Figura 31- Questionário avaliativo: resultados pergunta 9

Fonte: Autoria própria, 2023

A Figura 32 mostra que todos os participantes indicaram interesse em participar do processo de automatização das negociações de tokens nas pools de DCA, caso haja alguma recompensa financeira por isso, ou seja, as taxas superem os custos da transação.

Figura 32 - Questionário avaliativo: resultados pergunta 10



Fonte: Autoria própria, 2023

A Figura 33 mostra o resultado da pergunta 10, que questiona os usuários sobre a possível utilização desta aplicação uma vez que ela fosse implantada em uma blockchain de produção, e não apenas em uma rede de testes. Apenas um participante não consideraria a

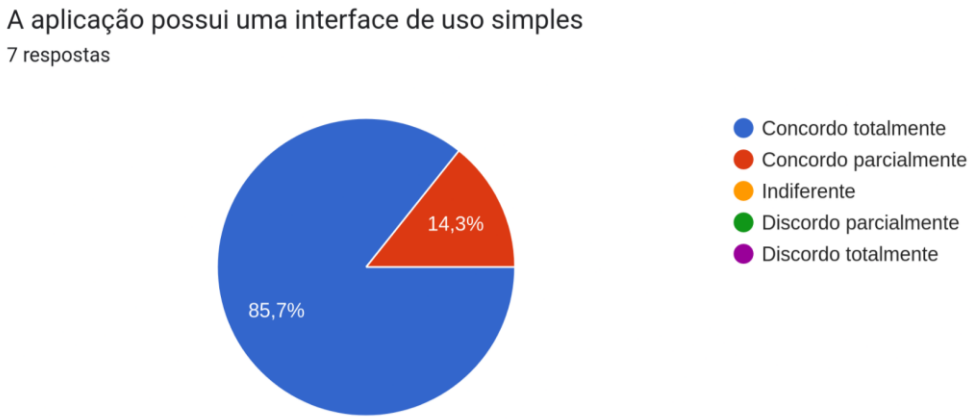
utilização da aplicação no cenário apresentado, todos os outros consideram de alguma forma a interagir com sistema uma vez implantado em um cenário real.

Figura 33 - Questionário avaliativo: resultados pergunta 11

Fonte: Autoria própria, 2023

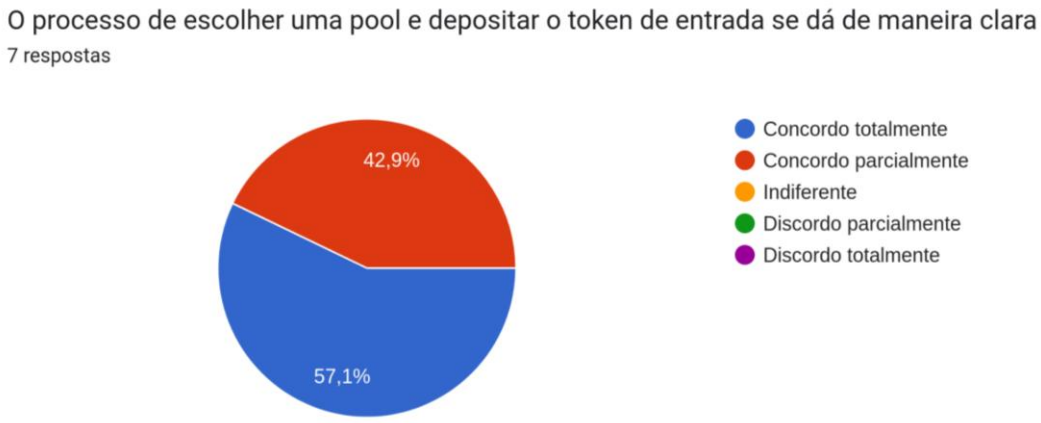
A Figuras 34, 35, 36 e 37 dizem respeito a avaliação dos usuários da interface que lhes foi apresentada no vídeo de demonstração da plataforma. Em todas elas os respondentes foram questionados em relação à clareza e simplicidade de cada um dos processos e telas apresentados. De maneira geral, em cada um dos gráficos nota-se que os usuários consideram as interfaces claras e simples para uso.

Figura 34 - Questionário avaliativo: resultados pergunta 12



Fonte: Autoria própria, 2023

Figura 35 - Questionário avaliativo: resultados pergunta 13

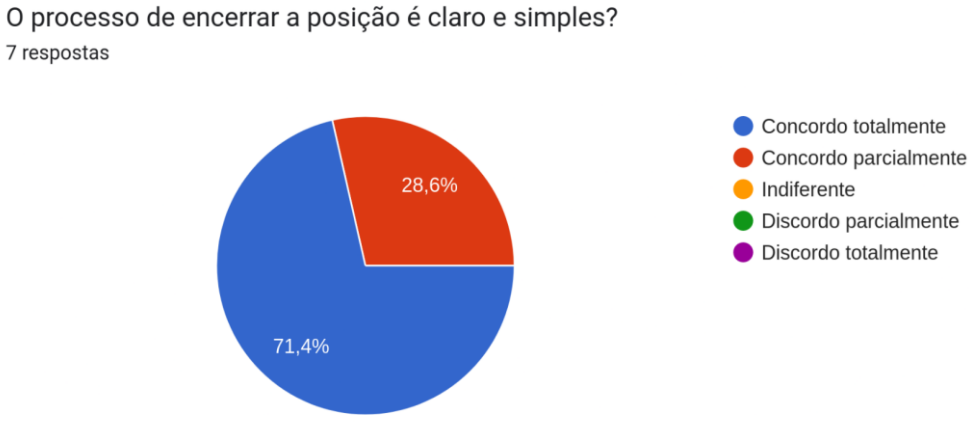


Fonte: Autoria própria, 2023

Figura 36 - Questionário avaliativo: resultados pergunta 14

Fonte: Autoria própria, 2023

Figura 37 - Questionário avaliativo: resultados pergunta 14



Fonte: Autoria própria, 2023

5.4.5 Conclusão da avaliação

A partir das análises dos resultados do questionário aplicado, pode-se constatar que a maioria dos participantes já faz o uso de estratégias de *Dollar-cost Averaging* para gerenciar o preço médio dos seus investimentos. Contudo, a maioria destes usuários nunca fez o uso de plataformas de DCA voltadas a criptomoedas e apenas um usuário utilizou uma plataforma com automatização utilizando a blockchain como infraestrutura principal. Para todos estes usuários a descentralização é um fator importante em algum grau, quando se trata de aplicações envolvendo criptomoedas. Fazendo uma análise desde subgrupo de informações, podemos concluir que o tema deste trabalho é relevante para o contexto de investimentos na blockchain, e identifica-se um nicho de mercado que pode ser explorado mais a fundo em trabalhos futuros.

Em relação a aplicação desenvolvida a maioria dos usuários se mostraram satisfeitos com a interface e os processos que por ela podem ser executados. Porém, em alguns cenários, usuários se revelaram não totalmente satisfeitos com a clareza das informações e dos processos apresentados, o que abre margem para que tais elementos possam vir a ser retrabalhados em novas iterações de desenvolvimento.

Reunindo as informações obtidas a partir dos dados reunidos durante a avaliação dos usuários, foi considerado que os objetivos propostos por este trabalho foram atingidos. Contudo, estes mesmos dados indicam que melhorias ainda podem ser feitas na plataforma para que ela atinja um estágio de maturidade até que seja considerada pronta para o mercado.

6 CONCLUSÃO

Com a prática de investimentos do autor e as pesquisas iniciais deste trabalho, constatou-se a falta de uma solução de *Dollar-cost averaging*, voltada para tokens ERC-20 em blockchains EVM. A adoção crescente de tecnologias blockchain por investidores, faz crescer também a necessidade de instrumentos de automatização de estratégias de investimento.

O conhecimento obtido durante a referência teórica deste trabalho permitiu que fossem mais bem compreendidas todas as variáveis que viriam a estar presente na modelagem e desenvolvimento da aplicação. Neste capítulo também foram apresentados ao leitor conceitos básicos de *blockchain*, para que este pudesse ter uma melhor compreensão dos assuntos abordados, assim como entender o funcionamento do ecossistema onde o serviço viria a ser desenvolvido.

O processo de modelagem do sistema foi iniciado com as definições de requisitos funcionais, não funcionais e regras de negócio, definições estas que foram de grande importância para a criação dos protótipos de tela que foram criados em sequência. O *wireframe* da aplicação foi construído com o objetivo de guiar a implementação da interface da aplicação, porém também foi útil para a validação visual dos requisitos criados no passo anterior. No processo de modelagem ainda foram criados os diagramas de casos de uso, que foram fundamentais para definir todos os possíveis cenários e ações de um usuário dentro da plataforma. Por fim, foram abordados ainda neste capítulo a criação do diagrama de classes do *smart-contract* que veio a ser desenvolvido e o diagrama de componentes do sistema.

Concluída a modelagem do sistema, o trabalho procedeu para a fase de implementação, onde todas as ideias referenciadas e modelos criados foram aplicadas na prática, resultando assim em uma aplicação de estágio inicial que ao final desta etapa foi submetida a avaliações de possíveis usuários. Foi considerada pelo autor a fase mais desafiadora de todo o processo por conta da pesquisa, tentativa e erro para chegar em um contrato inteligente que fosse capaz de atender os objetivos criados por este trabalho. Para a avaliação do objeto de saída deste trabalho, foi aplicado um questionário a possíveis usuários da plataforma com o intuito de reconhecer se a aplicação entrega ou não uma solução útil para investidores da classe de ativos alvo. Ao avaliar as respostas obtidas, foi constatado que o software estudado e desenvolvido ao longo deste projeto atende os seus objetivos principais e pode-se notar que existe uma demanda pelos serviços desenvolvidos dentre os usuários que participaram do questionário.

Por fim, considera-se atingido o objetivo geral e específico, porém com constatado durante a avaliação, com espaços para melhorias futuras. Mas ainda assim o sistema provê uma

forma de realizar estratégias de DCA utilizando a infraestrutura descentralizada da blockchain através do alinhamento de incentivos entre investidores buscando automatizar estratégias de *Dollar-cost Averaging*, usuários em busca de recompensas e a aplicação buscando por serviços prestados.

6.1 TRABALHOS FUTUROS

Os trabalhos futuros foram definidos de acordo com o campo do questionário destinado a avaliação aberta dos respondentes sobre o sistema, assim como na percepção do autor sobre possíveis novas funcionalidades que poderiam integrar o corpo de ferramentas oferecidas pela plataforma, porém não foram inicialmente incluídas com o objetivo de manter o corpo do projeto conciso. Também foram elencadas possíveis pesquisas de perguntas não respondidas por este trabalho por não constarem dentro do seu escopo inicial.

- **Histórico de preços dos ativos:** Para cada uma das *pools* criadas pelo administrador do sistema, seriam mostrados aos usuários o histórico de preços de ambos os ativos envolvidos naquela estratégia de investimento. Essa funcionalidade tem como objetivo ajudar na tomada de decisão do usuário sobre as *pools* que este escolherá para investir.

- **Melhorias no suporte de usuários que invocam função de *swap:*** Os usuários que invocam a função de negociação de tokens são peça fundamental para o bom funcionamento do sistema, apesar disso a interface gráfica não possui uma área dedicada para estes acompanharem suas recompensas pelas ações realizadas na plataforma. Crê-se, que construir tal suporte internamente na plataforma pode aumentar o nível de engajamento, logo a qualidade do serviço prestado pelos mesmos.

- **Melhorias na interface de criação de *pools*:** O processo de criação de *pools* exige que o administrador do sistema inclua diversos endereços de *smart-contracts* manualmente, que serão encaminhados para o contrato inteligente como parâmetros da função de criação, porém endereços da rede Ethereum são longos e não amigáveis para leitura humana. Desta forma, se faz necessária a criação de artifícios na interface gráfica que apresentem tais endereços de forma a melhorar a experiência do usuário. Uma possível solução para o problema é utilizar listagem de tokens, onde serão apresentados no formulário apenas o

logotipo e o nome ativo, ao passo que internamente, a informação sobre o endereço do token selecionado seria encaminhada para o *smart-contract.*

- **Aumentar corpo de informações da posição em *pool*:** O contexto dado ao usuário de cada uma de suas posições de DCA, e considerado mínimo pelo autor e por alguns respondentes do questionário, por tanto, essa melhoria na interface visa munir usuário com mais informações sobre suas posições, como a data de criação, quando acontecerá a próxima negociação de tokens e o histórico de valor da sua posição e moeda fiduciária, para que este possa julgar se sua tomada de decisão resultou em lucros ou perdas.

- **Avaliação da figura de administrador do sistema:** Na sua versão atual o sistema conta com um administrador responsável pela criação das opções de investimento, contudo, este trabalho não avalia a fundo a necessidade ou não da figura de um administrador e quais seriam os impactos para o funcionamento do sistema e seu mecanismo de recompensa, uma vez que usuário pudessem criar livremente *pools* de investimento.

- **Implementação para blockchains não-EVM:** Um dos objetivos deste trabalho era a implantação dos contratos inteligentes em uma blockchain que tivesse em seu cerne a Ethereum Virtual Machine. Porém outras blockchains de propósito geral que implementam suas próprias máquinas virtuais existem no mercado e a criação e implantação de soluções para DCA nessas redes pode aumentar o público alcançado pela solução desenvolvida.

REFERÊNCIAS

ALCHEMY. **Mumbai Testnet**. Disponível em: <https://www.alchemy.com/overviews/mumbai-testnet/>. Acesso em: 01 maio 2023.

ALHARBY, Maher; MOORSEL, Aad van. **Blockchain-based Smart Contracts: A systematic Mapping Study**. 2017.

AMBLER, Scott W.. **Agile modeling: Effective practices for eXtreme programming and the Unified Process**. John Wiley & Sons. 2002.

ANGELO, Monika di; SALZER, Gernot.. **Tokens, Types, and Standards: Identification and Utilization in Ethereum**. 2020 IEEE International Conference on Decentralized Applications and Infrastructures (DAPPS). 2020.

ASPRIS, A.; FOLEY, S.; SVEC, J.; WANG, L. **Decentralized exchanges: The "wild west" of cryptocurrency trading**. International Review of Financial Analysis, 77, 101845, 2021. doi:10.1016/j.irfa.2021.101845. 2021.

BARBOSA, S. D. J., & SILVA, B. P.. **Engenharia de requisitos: software orientado ao negócio**. Rio de Janeiro: Elsevier. 2014.

BASS, L., CLEMENTS, P., & KAZMAN, R. **Software architecture in practice (2nd ed.).** Addison-Wesley. 2003.

BECZE, Martin; JAMESON, Hudson.**"EIP-1: EIP Purpose and Guidelines," Ethereum Improvement Proposals". Disponível** em: <https://eips.ethereum.org/EIPS/eip-1>. Acesso em 20 de julho de 2022.

BRYMAN, Alan. **Social research methods**. Oxford University Press, 2016.

BUTERIN, Vitalik, **Standardized Contract APIs**. Diponível em: <https://github.com/ethereum/wiki/wiki/Standardized_Contract_APIs/499c882f3ec123537fc2fccd57eaa29e6032fe4a>. Acesso em 17 de novembro de 2022

BUTERIN, Vitalik. **Ethereum: A Next-Generation Smart Contract and Decentralized Application Platform**. Disponível em: <https://ethereum.org/669c9e2e2027310b6b3cdce6e1c52962/Ethereum_Whitepaper_-_Buterin_2014.pdf>. Acesso em 17 de novembro de 2022.

CHOHAN, Usman W. **A History of Bitcoin**. Discussion Paper Series: Notes on the 21st Century. 2022

CHOHAN, Usman W. **FTX, Cryptocurrencies, and Anarchism Ignored**. 2023. Disponivel em: <https://ssrn.com/abstract=4350999 or http://dx.doi.org/10.2139/ssrn.4350999>. Acesso em 19 de junho de 2023

CHUEN, D.L.K. **Handbook of Digital Currency**. 1. ed. Singapore: Elsevier. 2015

DAI, Wei. **B-money**. Disponível em: <http://www.weidai.com/bmoney.txt>. Acesso em 17 de abril de 2022.

ETHERS. **Documentação**. Disponível em: <https://docs.ethers.org/v5/>. Acesso em: 01 maio 2023.

FARELL**,** Ryan**. An Analysis of the Cryptocurrency Industry**. Disponível em: <https://repository.upenn.edu/cgi/viewcontent.cgi?article=1133&context=wharton_research_scholars>. Acesso em 30 de novembro de 2022.

HARDHAT. **Documentação**. Disponível em: <https://hardhat.org/docs>. Acesso em: 01 maio 2023.

JALAN, Akanksha; MATKOVSKYY, Roma. **Systemic Risks in the Cryptocurrency Market: Evidence from the FTX Collapse**. 2023

JENSEN, Johannes Rude; VON WACHTER, Victor; ROSS, Omri. **An Introduction to Decentralized Finance (DeFi)**. 2020 IEEE International Conference on Decentralized Applications and Infrastructures (DAPPS), 2020.

LEGGIO, Karyl B.; LIEN, Donald. **An Empirical Examination of the Effectiveness of Dollar-Cost Averaging Using Downside Risk Performance Measures**. JOURNAL OF ECONOMICS AND FINANCE Volume27 Number 2. Summer 2003. 2003.

METAMASK. **Glossary**. Disponível em: <https://learn.metamask.io/glossary/>. Acesso em: 01 maio 2023.

MUKHOPADHYAY, Ujan; SKJELLUM, Anthony; HAMBOLU, Oluwakemi; OAKLEY, Jon; YU, Lu; BROOKS, Richard**. A Brief Survey of Cryptocurrency Systems**, **Bitcoin and Cryptocurrency Technologies**: **A Comprehensive Introduction**. 2016

PIÑEIRO-CHOUSA, Juan; LÓPEZ-CABARCOS, M. Ángeles; SEVIC, Aleksanda; GONZÁLEZ-LÓPEZ, Isaac. **A preliminary assessment of the performance of DeFi cryptocurrencies in relation to other financial assets, volatility, and user-generated content**. Technological Forecasting and Social Change. Volume 181, 2022.

PRESSMAN, Roger S.; MAXIM, Bruce R. **Engenharia de Software: Uma Abordagem Profissional.** 8ª edição. McGraw-Hill, 2016.

POURPOUNEH, Mohsen; NIELSEN, Kurt; ROSS, Omri. **Automated Market Makers**. DK 1958 Frederiksberg DENMARK. 2020.

QIN, Kaihua; ZHOU, Liyi; AFONIN, Yaroslav; LAZZARETTI, Ludovico; GERVAIS, Arthur. **CeFi vs. DeFi — Comparing Centralized to Decentralized Finance**. Disponível em: <https://arxiv.org/pdf/2106.08157.pdf>. Acesso em: 10 out. 2022.

TAN, Lisa JY. **Economics and Math of Token Engineering and DeFi: Fundamentals of Token Economics (English Edition)**. 2020.

SOLIDITY. **Introdução**. Disponível em: <https://docs.soliditylang.org/en/v0.8.20/>. Acesso em: 01 maio 2023.

VOGELSTELLER, Fabian; BUTERIN, Vitalik "**EIP-20: Token Standard," Ethereum Improvement Proposals n° 20**. Disponível em: <https://eips.ethereum.org/EIPS/eip-20>. Acesso em 15 de novembro de 2022.

VUE. **Introdução**. Disponível em: <https://br.vuejs.org/v2/guide/index.html>. Acesso em: 01 maio 2023.

WIKIPEDIA. **TypeScript**. Disponível em: <https://pt.wikipedia.org/wiki/TypeScript>. Acesso em: 01 maio 2023.