UNIVERSIDADE FEDERAL FLUMINENSE

Levantamento de requisitos para desenvolvimento de Software: Protótipo de Scanner de Criptomoedas

Niterói

2018

Bruno Magalhães da Silva

Levantamento de requisitos e para desenvolvimento de Software: Protótipo de Scanner de Criptomoedas

Trabalho de Conclusão de Curso submetido ao Curso de Tecnologia em Sistemas de Computação da Universidade Federal Fluminense como requisito parcial para obtenção do título de Tecnólogo em Sistemas de Computação.

Orientadora:
Julliany Sales Brandão

NITERÓI
2018

Ficha catalográfica automática - SDC/BEE

S5861 Silva, Bruno Magalhães da
Levantamento de requisitos para desenvolvimento de Software: Protótipo de Scanner de Criptomoedas / Bruno Magalhães da Silva ; Julliany Sales Brandão, orientadora. Niterói, 2018.
45 f.

Trabalho de Conclusão de Curso (Graduação em Tecnologia de Sistemas de Computação)-Universidade Federal Fluminense, Escola de Engenharia, Niterói, 2018.

1. Moeda. 2. Prototipagem. 3. Engenharia de requisitos. 4. Scanner. 5. Produção intelectual. I. Título II. Brandão,Julliany Sales , orientadora. III. Universidade Federal Fluminense. Escola de Engenharia. Departamento de Ciência da Computação.

CDD -

Bibliotecária responsável: Fabiana Menezes Santos da Silva - CRB7/5274

Bruno Magalhães da Silva

Levantamento de requisitos para desenvolvimento de Software: Protótipo de Scanner de Criptomoedas

Trabalho de Conclusão de Curso submetido ao Curso de Tecnologia em Sistemas de Computação da Universidade Federal Fluminense como requisito parcial para obtenção do título de Tecnólogo em Sistemas de Computação.

Niterói, ___ de _____ de 2018.

Banca Examinadora:

_____

Profª. Julliany Sales Brandão, DSc. – Orientadora

Cefet/RJ – Centro Federal de Educação Tecnológica Celso Suckow da Fonseca.

_____

Prof. Felipe Pereira do Carmo, MSc. - Avaliador

IFF – Instituto Federal Fluminense

AGRADECIMENTOS

À orientadora, Profa. Julliany Sales Brandão, por toda contribuição e dedicação na elaboração deste trabalho científico. Aqui expresso meu sincero agradecimento por ter, gentilmente, disponibilizado seu tempo, atenção e paciência.

Agradeço aos meus professores, que contribuíram para a minha formação acadêmica.

Agradeço a meus familiares, que sempre me apoiaram e sem os quais nunca teria tido a oportunidade de concluir esse curso de formação, em especial à minha mãe, minha irmã e namorada que sempre me ajudaram nos momentos mais difíceis.

Agradeço aos meus colegas de faculdade dos quais sempre lembrarei com saudade os dias que passamos juntos.

A todos aqueles que de alguma maneira contribuíram para essa monografia e para minha formação acadêmica.

RESUMO

Pretende-se nesse trabalho realizar o desenvolvimento de toda a etapa de levantamento de requisitos além da criação de um protótipo de um sistema capaz de monitorar os preços das principais criptomoedas hoje comercializadas no mercado, de maneira a alertar o usuário sobre variações recentes nos preços desses ativos.
Palavras-chaves: moeda, prototipagem, engenharia de requisitos, scanner.

ABSTRACT

It is intended in this work to develop the entire stage of software requirements and create a prototype of a system capable of monitoring the prices of the main crypto-currencies in the market, in order to alert the user of recent variations in prices of these assets.

Key words: currency, prototyping, requirements engineering, scanner.

SUMÁRIO

1 INTRODUÇÃO

Nos últimos anos tem-se falado cada vez mais em criptomoedas em especial do bitcoin, o bitcoin é uma moeda digital descentralizada que teve sua ideia apresentada ao mundo em 2008 na lista de discussão *The Cryptography Mailing List* [1] onde foi descrito o protocolo bitcoin pela primeira vez. Em 2009 começam as emissões pela primeira vez de bitcoins e de lá pra cá o mercado de criptomoedas tem se tornado cada vez mais popular.

Uma criptomoeda pode ser definida como dinheiro virtual ou digital, que assume a forma de “moedas” ou “tokens”. As criptomoedas de uma maneira geral possuem um algoritmo que permite elas serem geradas, armazenadas e transferidas de maneira segura e tipicamente anônima.

Tendo em vista essa crescente difusão do mercado de criptomoedas o presente trabalho destina-se a realizar a análise de requisitos necessários para a criação de um software capaz de monitorar de maneira customizável as flutuações nos preços das principais criptomoedas existentes no mercado.

Figura 1 – Capitalização total do mercado de criptomoedas [2]

Hoje em dia existem uma série de corretoras (*Exchanges*) tanto brasileiras como internacionais que realizam a intermediações desses tipos de ativos, no Brasil a exemplo temos as corretoras *Foxbit*, *Mercado Bitcoin, Bitcointrade,* dentre outras e já as corretoras internacionais podemos ressaltar a *HitBTC, Cryptopia, Poloniex, Binance*, dentre diversas outras. Como normalmente as corretoras brasileiras negociam em geral com volume apenas as principais criptomoedas como *Bitcoin* e *Ether*, e as corretoras internacionais possuem maior volume de movimentação além de maior oferta de moedas de menor difusão (*altcoins*) o presente trabalho terá como base a utilização de uma corretora internacional no caso a *HitBTC*.

No presente trabalho serão utilizadas algumas ferramentas de modelagem de sistemas como diagramas de casos de uso, diagramas de classes e de sequência para poder avaliar os requisitos que serão necessários no sistema.

1.1 Motivação

Apesar de existirem ferramentas capazes de parametrizar alertas sobre variações nos preços das moedas como a plataforma do Coinigy (coinigy.com) por exemplo, essas ferramentas em geral não são gratuitas além de exigirem uma quantidade razoável de tempo para parametrizar cada moeda com um alerta, nesse sentido criar uma ferramenta capaz de realizar essa parametrização de forma geral se torna extremamente produtivo uma vez que os alertas seriam disparados para todas as moedas que atendessem aquela condição, o que permite um melhor monitoramento das possibilidades de ganhos nesse mercado.

1.2 Objetivo

Esse trabalho tem o intuito de fazer o levantamento e análise de requisitos necessários para a criação do software de monitoramento de criptomoedas, porém não será o intuito desse trabalho implementar ainda o software em si, sendo apenas realizada toda a etapa de levantamento de requisitos e a criação de um protótipo para tal solução.

1.3 Estrutura do Trabalho

O trabalho foi organizado em seis capítulos, sendo o primeiro capítulo a introdução onde é apresentado a ideia geral do porquê do tema escolhido e a organização do trabalho; já no segundo capitulo é introduzido ao leitor as características do protótipo e como as API’s da HitBTC funcionam; em seguida no terceiro capítulo é realizada toda análise de requisito da aplicação passando por identificação e atuação dos Stakeholders, requisitos funcionais e não funcionais do sistema, além da criação dos diagramas de casos de uso, de classes e de sequência; no quarto capítulo será apresentada a interface do programa e como ocorre o funcionamento dessa interface; no quinto capítulo fala-se sobre as tecnologias envolvidas na aplicação e sobre a sua estrutura de desenvolvimento; por fim no sexto capítulo a conclusão do trabalho é realizada.

2 CARACTERISTICAS DO PROTÓTIPO

A ideia do protótipo consiste em um programa que utilizará uma API para comunicação com uma determinada corretora de criptomoeda como a HitBTC por exemplo, essa comunicação ocorrerá com a utilização da chamada da API dentro de um loop que ficaria ativo durante toda a execução do programa.

A periodicidade de leitura da API será fixada em trinta em trinta segundos para atualizar os valores das cotações em uma tabela interna do programa.

A variação do valor, o volume mínimo e o código das moedas que não serão exibidos deverão ser parametrizados dentro da interface do programa antes de sua execução.

O protótipo deverá ser exibido em uma página HTML que após ser acessada apresentará os parâmetros no início da página.

2.1 API da HitBTC

Para o programa em questão será utilizado apenas as informações pertinentes a seção de *Market Data* da API da HitBTC [3] uma vez que não serão disparadas ordens de compra ou de venda pelo aplicativo. Na sessão *Market data* da plataforma pode-se verificar todas as fontes de informações sobre movimentações referentes a todas as moedas disponíveis na corretora [4] ou apenas para a moeda que foi selecionada.

Na aplicação existirá um campo destinado a exclusão de moedas desta forma a aplicação buscará todas as moedas disponíveis na API da HitBTC, em seguida após a seleção de todas as moedas da corretora serão excluídas da matriz da tabela resultante os códigos das moedas que foram especificados para não serem utilizados, em um primeiro momento pode parecer estranho excluir dos alertas variações de determinadas moedas, entretanto, existem moedas que possuem valores

muito baixos como variações de U$ 0,01 para U$ 0,02 representando desta maneira uma variação de 100% o que faria com que o scanner alertasse constantemente sobre essas variações que apesar de em um primeiro momento parecerem vantajosas representam em sua grande maioria variações de compras e ou vendas fora da curva e com baixíssimo volume de vendas.

Para a aplicação primeiramente será necessário resgatar todas as moedas que são listadas pela corretora, para isso será utilizada API de *currency* [5], que exibe todas as moedas existentes na corretora e através do parâmetro *delisted* e informa se a moeda está listada ou não, abaixo são apresentados os parâmetros retornados pela API.

Name	Type	Description
id	String	Currency identifier. In the future, the description will simply use the `currency`
fullName	String	Currency full name
crypto	Boolean	Is currency belongs to blockchain (false for ICO and fiat, like EUR)
payinEnabled	Boolean	Is allowed for deposit (false for ICO)
payinPaymentId	Boolean	Is required to provide additional information other than the address for deposit
payinConfirmations	Number	Blocks confirmations count for deposit
payoutEnabled	Boolean	Is allowed for withdraw (false for ICO)
payoutIsPaymentId	Boolean	Is allowed to provide additional information for withdraw
transferEnabled	Boolean	Is allowed to transfer between trading and account (may be disabled on maintain)
delisted	Boolean	True if currency delisted (stopped deposit and trading)
payoutFee	Number	Default withdraw fee

Figura 2 - Estrutura da API de *Currency* HitBTC. [5]

Figura 3 - Formato da API de *Currency* da HitBTC - [5]

- Id: Identificador da moeda
- fullName: Nome completo da moeda.
- Crypto: identifica se a moeda pertence ao *blockchain*.
- PayinEnabled: identifica a permissão de depósito.
- payinPaymentId: identificador de informação adicional de deposito.
- payinConfirmations: confirmação do bloco para depósito.
- payoutEnabled: identificador de permissão de retirada.
- payoutIsPaymentId identificador de informação adicional de retirada.
- transferEnabled: transferência entre contas permitidas.
- Delisted: identificador de moeda não listada.
- payoutFee: taxa de retirada.

Outra API que será utilizada consiste na API referente aos *candlesticks (ou candles)* de determinada moeda, para isso primeiramente é necessário destacar como os *candlestick* são formados.

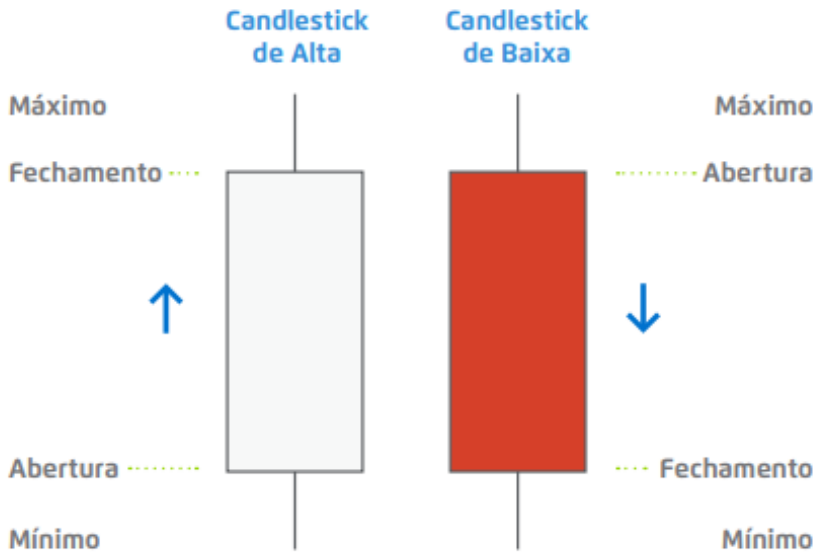

Figura 4 – Estrutura de um *Candlesticks* [6]

Pode-se verificar na Figura 4 acima que os *candles* são compostos por uma parte fina do corpo que representa o valor máximo e o valor mínimo negociado durante um determinado intervalo de tempo, já a parte mais grossa do corpo representa o valor de abertura e fechamento do *candle* sendo o “vermelho” representando que o ativo fechou em queda, logo a parte de baixo será o fechamento e a parte de cima será a abertura, analogamente caso o *candle* seja “branco” (ou verde na Figura abaixo) representa que o candle fechou em alta, logo a parte de cima representa o fechamento e a parte de baixo a abertura.

Figura 5 - Gráfico ETH/BTC da HitBTC [7]

> “No gráfico *Candlestick*, esses retângulos são alinhados lado a lado, com traços (ou sombras) que informam os mínimos e os máximos do período. Além disso, os *candles* possuem cores que revelam se o ativo fechou acima ou abaixo do preço de abertura, facilitando a identificação à primeira vista: se seu preenchimento for branco ou transparente, é sinal de que os preços fecharam em alta; caso seja colorido, significa que os ativos fecharam em baixa.” [6]

Agora que foi definido como os *candles* funcionam pode-se demonstrar como funciona a API referente aos *candles* da HitBTC [8]. Essa API deverá ser chamada para cada moeda que o sistema verificará a cotação, diferente da primeira API essa agora necessita de alguns parâmetros adicionais para definir a estrutura de retorno dos *candles*.

Name	Type	Description
period	String	One of: M1 (one minute), M3, M5, M15, M30, H1, H4, D1, D7, 1M (one month). Default is M30 (30 minutes).
sort	String	Default ASC
from	Datetime	
till	Datetime	
limit	Number	Limit of candles, default 100.
offset	Number	

Figura 6 - Parâmetros da API de *candles* da HitBTC [8]

- period: Periodicidade do *candle*.
- Sort: formato de ordenação (padrão ascendente).
- From: data/hora inicial.
- Till: Data/hora Final.
- Limit: limite máximo de *candles* selecionados.
- Offset: número de *candles* iniciais ignorados.

Como pode-se verificar acima os parâmetros são simples, eles basicamente solicitam o número de *candles* máximos a serem selecionados e um intervalo definido, vale ressaltar que o parâmetro mais importante nessa chamada trata-se do parâmetro *period*, esse parâmetro define basicamente se os *candles* que serão selecionados em minutos, horas, dias ou meses o que será a base da análise do programa.

O retorno da API “/api/2/public/*candles*/{symbol}” é mostrado abaixo:

Name	Type	Description
timestamp	Datetime	
open	Number	Open price
close	Number	Close price
min	Number	Min price
max	Number	Max price
volume	Number	Volume in base currency
volumeQuote	Number	Volume in quote currency

Figura 7 – Estrutura da API de *Candles* da HitBTC [8]

- timestamp: registro data/hora do registro.
- open: preço de abertura (do *candle*).
- close: preço de fechamento (do *candle*).
- min: preço mínimo que o *candle* atingiu.
- max: preço máximo que o *candle* atingiu.
- volume: volume na moeda base.
- volumeQuote: volume na moeda de cotação.

Os parâmetros acima citados são fornecidos através de um arquivo tipo “JSON” que retorna uma matriz de vetores em uma “string” com os valores atualizados.

Figura 8 – Formato da API de *Candles* da HitBTC [8]

Por fim a última API que será utilizada é a API referentes aos *Tickers* [9], esses podem ser definidos como um dispositivo que faz as marcações atualizadas das cotações das moedas e retorna para os usuários da aplicação. Essa API pode ser chamada para cada moeda que quisermos verificar a cotação ou caso o parâmetro de moeda não seja passado ela retornará à cotação de todas as moedas no instante.

Name	Type	Description
ask	Number	Best ask price
bid	Number	Best bid price
last	Number	Last trade price
open	Number	Last trade price 24 hours ago
low	Number	Lowest trade price within 24 hours
high	Number	Highest trade price within 24 hours
volume	Number	Total trading amount within 24 hours in base currency
volumeQuote	Number	Total trading amount within 24 hours in quote currency
timestamp	Datetime	Last update or refresh ticker timestamp
symbol	String	

Figura 9 – Estrutura da API de Tickers da HitBTC [9]

O formato da linha de resposta é composto por atributos que serão informados para cada moeda selecionada, são eles:

- *Ask:* Representa o preço de compra, nesse campo da linha conterá a informação do maior preço de compra ofertado.
- *Bid:* De maneira análoga representa o preço de venda, nesse campo da linha conterá a informação do menor preço de venda ofertado.
- *Last*: Representa o ultimo preço comercializado na data corrente.
- *Open:* Representa o ultimo preço comercializado no dia anterior.
- *Low:* Menor preço comercializado nas últimas 24 horas.
- *High:* Maior preço comercializado nas últimas 24 horas.
- *Volume:* Volume total de comercialização nas últimas 24 horas na respectiva moeda de base.

- *VolumeQuote*: Volume total de comercialização nas últimas 24 horas na respectiva moeda de referência da cotação.
- *Timestamp:* Registro de data e hora da última atualização fornecida.
- *Symbol*: Símbolo da moeda base concatenado com o símbolo da moeda de referência.

Figura 10 - Formato da API de Tickers da HitBTC [9]

2.2 Parâmetros de alerta da aplicação

Essa etapa destina-se a descrever os parâmetros que o usuário poderá preencher a fim de filtrar quais corretoras e criptomoedas devem ser monitoradas.

- *Período de monitoramento*

Esse parâmetro destina-se a estabelecer o intervalo entre os dois tempos que serão monitorados. Esse parâmetro deve admitir os valores 1, 5, 15 e 30 que

representará o intervalo de minutos da janela de comparação. Possui valor *default* de 5 (minutos).

- *Percentual de variação*

Parâmetro define o percentual de variação no preço da moeda que deve ser atingido para que o alerta de preço seja disparado. Esse parâmetro admite tanto entradas negativas como positivas. Possui valor *default* de 7 (porcento).

- *Volume mínimo*

Parâmetro define o volume mínimo em BTC (bitcoins) necessário para que o alerta seja disparado, variações que atinjam o percentual sem ter o volume mínimo alcançado não serão disparadas pelo programa. Possui valor *default* de 5 BTC (bitcoins).

- *Moeda base*

Nesse parâmetro são definidas quais moedas base são aceitas para as nossas verificações, as moedas bases possíveis para a maioria das corretoras são, BTC (bitcoin), ETH (etherium), USD (Dollar) e USDT (Tether). Parâmetro do tipo “check button” com as quatro possibilidades de moedas. Possui valor *default* de marcado para todas as moedas.

- *Moedas de exclusão*

Esse parâmetro consiste de uma *string* que tenta interpretar os códigos de todas as moedas separados com o caractere “;” (ponto e vírgula), caso o programa não identifique o código entre “;” como um código válido, o programa simplesmente irá desconsiderar a parte não identificada da *string*.

Esses parâmetros ficaram amostra para o usuário em uma *pop-up* caso o usuário clique na opção de alterar os valores de monitoramento.

3 ANÁLISE DE REQUISITOS

A análise de requisitos corresponde ao processo de observação e levantamento dos elementos do ambiente onde será realizada a implementação de determinado software. Trata-se do início do processo de desenvolvimento onde todas as atividades envolvidas no sistema devem ser verificadas de maneira a identificar novas necessidades e mitigar possíveis impactos no sistema.

3.1 *Stakeholders*

No processo de desenvolvimento de software os *Stakeholders* são todas as partes interessadas no projeto, segundo o PMBOK “são indivíduos e organizações diretamente envolvidos no projeto, ou aqueles cujos interesses podem ser afetados, de forma positiva ou negativa, no decorrer do projeto ou mesmo após sua conclusão” [10]. Nesse sistema foram identificados os *Stakeholders* abaixo:

- Administrador do sistema

Responsável por gerenciar o sistema de uma maneira geral, tanto com relação a acessos como com relação a disponibilidade.

- Usuário do sistema

Qualquer pessoa envolvida que tenha permissão para utilizar o sistema.

3.2 Atuação dos *Stakeholders*

Essa seção é destinada a descrever como será a atuação dos *Stakeholders* no sistema em questão.

- Administrador do sistema

Será responsável por realizar as parametrizações necessárias que garantem a cada usuário o acesso ao sistema, além de garantir a disponibilidade do servidor em que a plataforma irá rodar.

- Usuário do sistema

Pessoa que irá usufruir das informações do sistema, será responsável por customizar as informações de alerta do sistema de maneira a atender ao seu interesse.

3.3 Requisitos funcionais

Os requisitos funcionais tratam o comportamento esperado do sistema, quais funções são esperadas serem atingidas com a utilização do sistema.

- Alertas

O sistema deverá constantemente realizar consultas sobre as variações nos preços das criptomoedas e reportar os casos em que as condições estabelecidas pelo usuário forem atingidas.

- Histórico

O sistema deverá manter os históricos dos alertas atingidos pelas parametrizações do usuário enquanto o usuário estiver com a seção aberta.

3.4 Requisitos não funcionais

Os requisitos não-funcionais especificam o comportamento do produto, são relacionados ao uso da aplicação em termos de desempenho, usabilidade, confiabilidade, segurança, disponibilidade, manutenção e tecnologias envolvidas.

- Desempenho

O sistema deve ser capaz de realizar a comparação de todas as criptomoedas (busca completa: cenário com o parâmetro “moedas de exclusão” vazio) para quaisquer parâmetros definidos com o menor tempo de resposta possível sem que ultrapasse o tempo de resposta de 10 segundos.

O sistema deve conseguir popular as tabelas internas com os preços das criptomoedas a cada iteração em menos 60 segundos.

- Operacional

O sistema deve ser desenvolvido em Python 3.

O sistema deve ser capaz de exibir seu leiaute de alertas nos principais navegadores do mercado, como Chrome, Firefox, Edge e Internet Explorer.

- Disponibilidade

O servidor que rodar a aplicação deve ficar disponível 24 horas por dia, 7 dias por semana.

3.5 Diagrama de caso de uso

Segundo Booch, Rumbaugh e Jacobson “Um caso de uso especifica um comportamento de um sistema ou parte de um sistema e é uma descrição de um conjunto de ações, incluindo variantes, realizadas pelo sistema para produzir um resultado observável do valor de um ator.” [11]

Casos de uso descrevem os requisitos externos do sistema, esses são usados na fase de análise de requisitos.

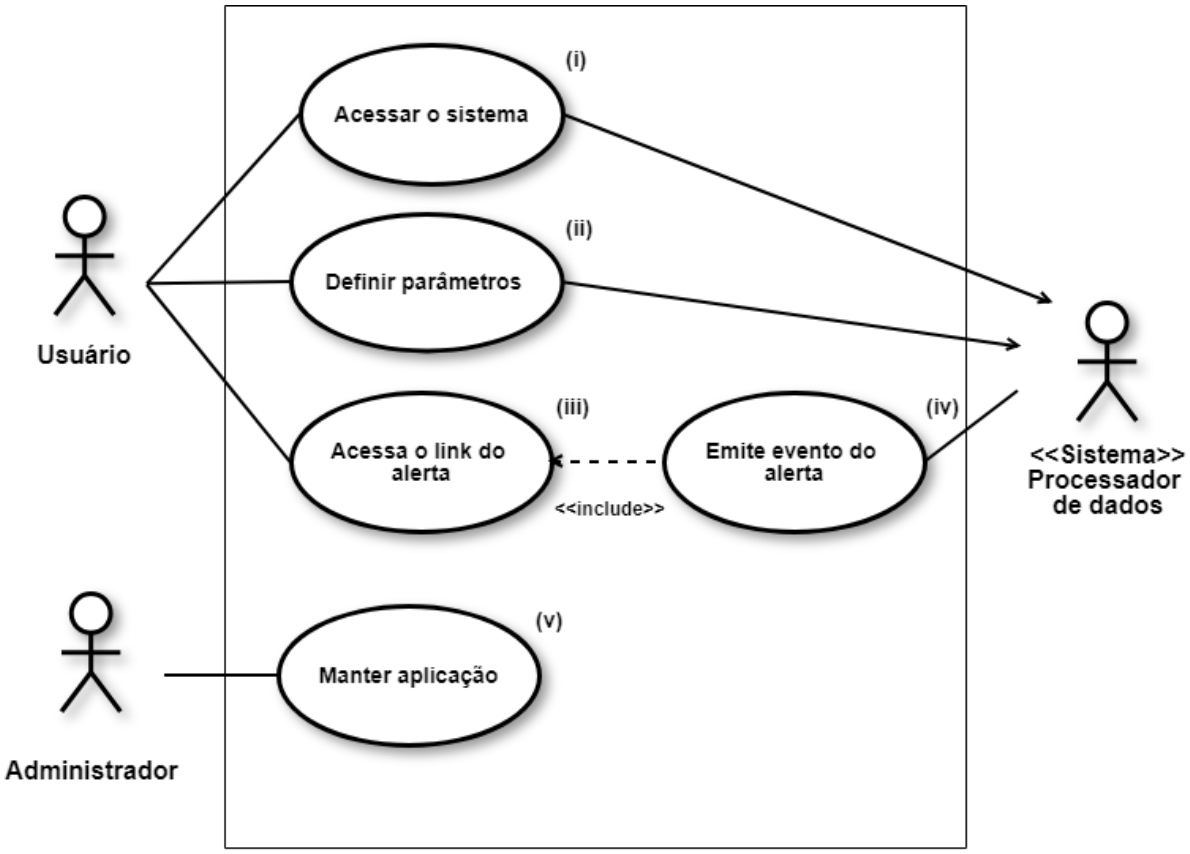

Figura 11 - Casos de uso do sistema

3.6 Descrição de caso de uso

I. DCU - Acessar sistema

Esse caso de uso é iniciado quando o usuário acessa via navegador o site em que a aplicação está exibindo o leiaute.

Ator: Usuário.

Fluxo básico:

1. Usuário acessa o site da aplicação.
2. Sistema verifica se existem parâmetros definidos anteriormente para a seção do usuário.
3. Sistema resgata parâmetros anteriormente salvos.
4. Sistema inicia a verificação de alertas para aquele usuário.

Fluxo Alternativo:

3.a. Sistema não identifica parâmetros anteriormente salvos.

4.a. Sistema não inicia verificação de alertas.

II. DCU - Definir parâmetros

Esse caso de uso ocorre quando o usuário clica no botão “redefinir parâmetros”.

Ator: Usuário.

Fluxo básico:

1. Usuário clica no botão “redefinir”.

2. Sistema abre uma janela com os parâmetros a serem preenchidos.

3. Usuário preenche os parâmetros que desejar.

4. Usuário clica no botão “salvar”.

5. Sistema inicia a verificação de alertas para aquele usuário.

Fluxo Alternativo:

4.b. Usuário clica no botão “cancelar”.

5.b. Sistema volta ao estado anterior da definição de parâmetros.

Pós-condição: Sistema passa a monitorar a partir dos novos parâmetros.

III. DCU - Acessar link do alerta

Esse caso de uso ocorre quando após a emissão do alerta o usuário acessa o link disponível no registro do alerta.

Ator: Usuário.

Pré-condição: Sistema deve emitir um alerta com o link disponível para acesso.

Fluxo básico:

1. Sistema emite alerta para usuário.
2. Usuário acessa o link disponível referente ao alerta.
3. Usuário é direcionado em uma nova janela para o site da HitBTC na sessão da moeda referente ao alerta.

IV. DCU - Emitir evento do alerta

Esse caso de uso ocorre quando o sistema identifica que os parâmetros preenchidos pelo usuário foram atingidos.

Ator: Processador de dados do sistema.

Fluxo básico:

1. Sistema processa dados com os parâmetros passados.
2. Registro com os parâmetros indicados é encontrado.
3. Sistema emite alerta para o usuário.

Pós-condição: Emitir um alerta com o link disponível para acesso.

V. DCU - Garantir disponibilidade

Esse caso de uso ocorre quando por algum motivo o sistema fica, indisponível.

Ator: Administrador.

Fluxo básico:

1. Administrador para a aplicação.
2. Administrador realiza ajustes na aplicação.
3. Administrador reinicia a aplicação.

3.7 Diagrama de classes

Um diagrama de classes trata-se de uma representação da estrutura e relações das classes que servem de modelo para objetos. [12]

O Diagrama de classes fornece uma visão estática do sistema, entretanto não é capaz de mostrar a natureza dinâmica das comunicações entre os objetos das classes.

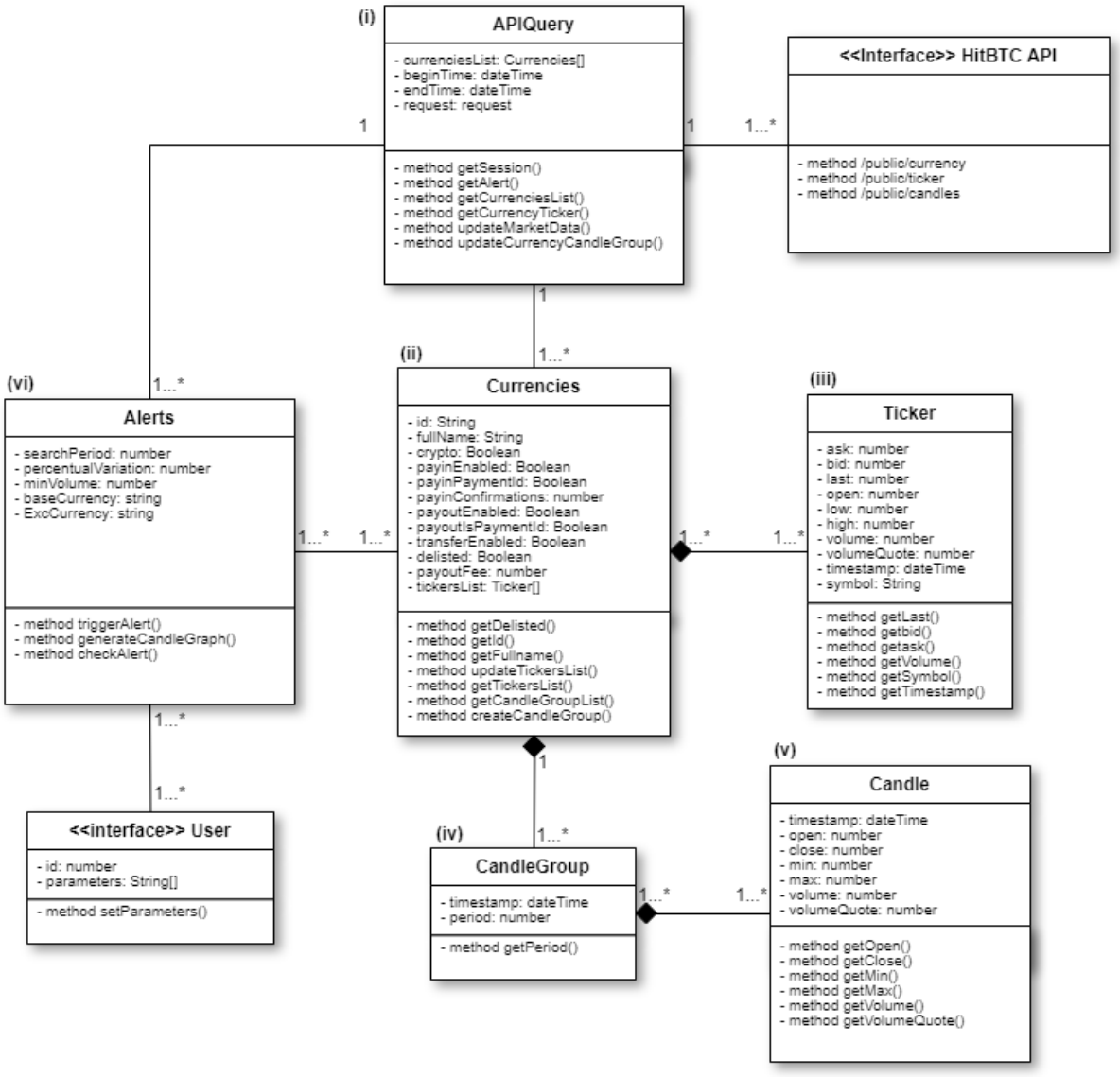

Figura 12 - Diagrama de classes do sistema

Na Figura 12 acima pode-se verificar a disposição das classes do sistema. As classes APIQuerry, Alerts, Currencies, Ticker, CandleGroup e Candle representam a totalidade das classes dispostas no sistema.

I. C - APIQuerry

classe responsável por estabelecer as conexões necessárias entre o sistema e a interface da HitBTC.

II. C - Currencies

Representa as informações referentes a cada moeda, ela armazena as demais classes que compõem os dados mais específicos de cada moeda.

III. C - Ticker

classe específica da classe clemencies que representa as informações das cotações atualizadas a cada iteração.

IV. C - CandleGroup

Classe é responsável por agrupar um conjunto de *candles* que formam a gráfico que é apresentado juntamente com o alerta da cotação.

V. C - Candle

Essa classe modela cada *candle* presente na classe *CandleGroup*.

VI. C - Alerts

Classe responsável por monitorar se as condições estabelecidas pelo usuário do sistema foram atingidas.

3.8 Diagrama de sequencia

O Diagrama de sequência é um diagrama utilizado na linguagem UML que representa uma sequência de processos existentes em um sistema.

Em contraste com os diagramas de classe e os de distribuição, que mostram a estrutura estática de um componente de software, o diagrama de sequência é utilizado para indicar as comunicações dinâmicas entre objetos durante a execução de uma tarefa. Ele mostra a ordem temporal na qual as mensagens são enviadas entre os objetos para executar aquela tarefa. Podemos usar o diagrama de sequência para mostrar as iterações em um caso de uso ou em um cenário de sistema de software. [13]

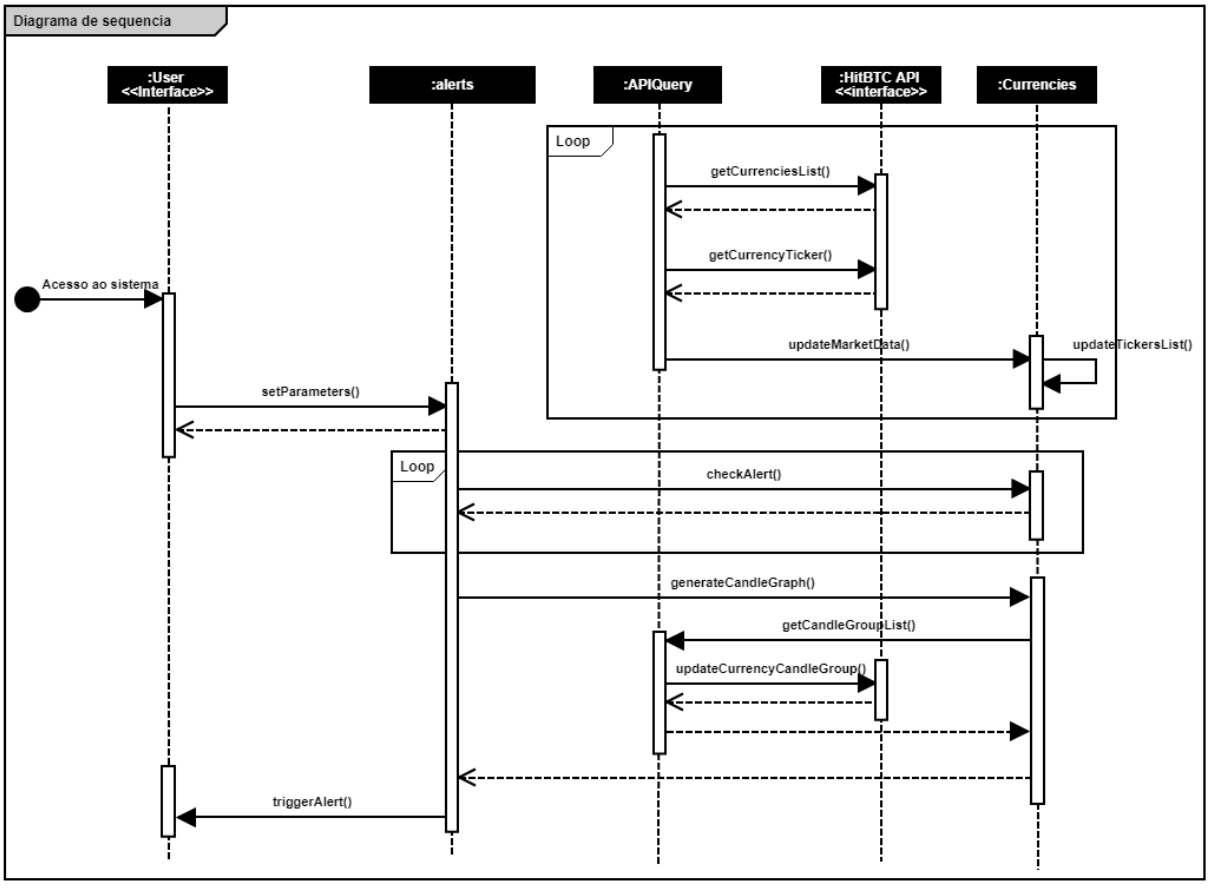

Figura 13 - Diagrama de sequência do sistema

No diagrama de sequência acima estão representados os principais métodos responsáveis pelas ações do software.

O primeiro processo constitui os métodos necessário para a constante atualização da tabela de preço das criptomoedas do sistema, nessa etapa os métodos *getCurrenciesList, getCrurrencyTicker, UpdateMarketData* e *updateTickersList* são chamadas em um loop infinito em intervalos regulares regatando informações da API da HitBTC e para popular a tabela.

O segundo processo se inicia quando o usuário acessa de fato a aplicação, nessa etapa o programa irá definir os parâmetros de busca do usuário através do método *setParameters* e a partir desse momento o programa irá entrar em um loop constante que verifica se os parâmetros estabelecidos pelo usuário foram atingidos através do método *checkAlert*.

Caso um alerta seja encontrado antes de disparar o alerta para o usuário o sistema irá buscar as informação dos *candles* daquela moeda para poder formar um pequeno gráfico de *candles* juntamente com o alerta, esse processo tem seu início com o método *generateCandleGraph* passando pelos métodos *getCandleGroupList* e *updateCurrrencyCandleGroup* que irão buscar as informações dos *candles* através da API da HitBTC novamente.

4 INTERFACE DO PROTÓTIPO DE SCANNER

Nesse capítulo serão tratadas as principais características referentes a interface do protótipo em questão.

4.1 Layout e funcionalidade do sistema

Ao acessar o site do sistema o usuário encontrará a tela da Figura 14, nessa tela será possível acompanhar os alertas que forem disparados bem como redefinir os parâmetros de busca de alerta da aplicação.

Mesmo que seja o primeiro acesso do usuário na aplicação o alerta ainda sim começará a funcionar com os valores *default*, conforme mostra a Figura 14.

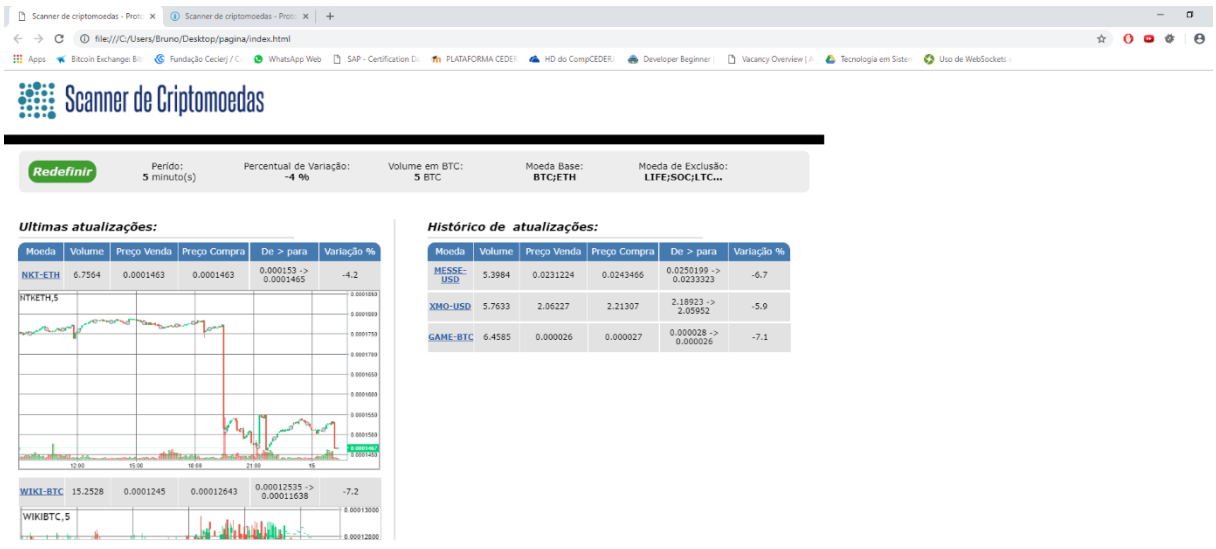

Figura 14 - Tela do Site do Scanner de Criptomoedas

Como pode-se verificar na Figura 14 a tela inicial é composta basicamente de três grandes grupos de elementos além de uma funcionalidade de

hyperlink presente em alguns desse grupos, sendo eles:

- Definição de parâmetros

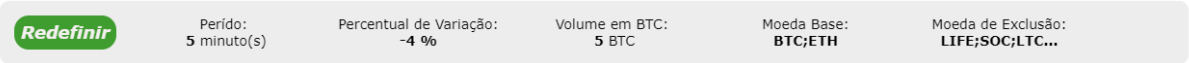

Figura 15 - Indicador de parâmetros do sistema

Na Figura 15 podemos verificar todos os parâmetros que foram selecionados pelo sistema ou que vieram automaticamente como *default* do sistema.

Caso o usuário deseje redefinir os parâmetros do sistema basta que clique no botão “redefinir”. Ao clicar no botão redefinir uma janela tipo *pop-up* irá aparecer na tela para que o usuário redefina os parâmetros conforme a Figura 16.

Figura 16 - Tela de parâmetros do sistema.

O *pop-up* que irá aparecer para o usuário é bem intuitivo, após preenchimento dos campos o usuário terá a opção de salvar ou de cancelar os parâmetros preenchidos de acordo com a opção selecionada nos respectivos botões.

Independente de qual dos dois botões o usuário optar ambas as opções o levarão para a tela inicial da aplicação, caso ele opte por cancelar os parâmetros antigos continuaram, caso ele opte por salvar os novos parâmetros serão utilizados nas comparações.

- Últimas atualizações

Nessa seção o usuário poderá visualizar os alertas mais recentes, cada alerta disparado irá criar uma entrada com os valores encontrados [Figura 17], gerar um gráfico [Figura 18] e disparar um efeito sonoro para o usuário. Nessa seção ficaram todos os alertas disparados nos últimos 10 minutos.

Moeda	Volume	Preço Venda	Preço Compra	De > para	Variação %
NKT-ETH	6.7564	0.0001463	0.0001463	0.000153 -> 0.0001465	-4.2

Figura 17 - Valores retornados pelo alerta.

Figura 18 - Gráfico retornado pelo alerta.

- Histórico de atualizações

Nessa seção ficaram todos os alertas antigos que o usuário recebeu enquanto a seção do usuário estiver aberta, quando um alerta passar de 10 minutos automaticamente ele passa do lado esquerdo do site referente as últimas atualizações para o lado direito referente ao histórico, como pode-se verificar na Figura 19.

Histórico de atualizações:

Moeda	Volume	Preço Venda	Preço Compra	De > para	Variação %
MESSE-USD	5.3984	0.0231224	0.0243466	0.0250199 -> 0.0233323	-6.7
XMO-USD	5.7633	2.06227	2.21307	2.18923 -> 2.05952	-5.9
GAME-BTC	6.4585	0.000026	0.000027	0.000028 -> 0.000026	-7.1

Figura 19 - Histórico de atualizações do usuário.

- Hyperlink com a HitBTC

O sistema dispõe também de uma funcionalidade de hyperlink para os alertas com o site da HitBTC de maneira que para cada alerta disparado pela aplicação no campo “Moeda” gerado o nome da moeda é um hyperlink conforme a Figura 20.

Figura 20 - Hyperlinks com a HitBTC

Ao clicar por exemplo no link “GAME-BTC” será aberta uma nova janela no site da HitBTC referente a moeda selecionada conforme demostrado na Figura 21.

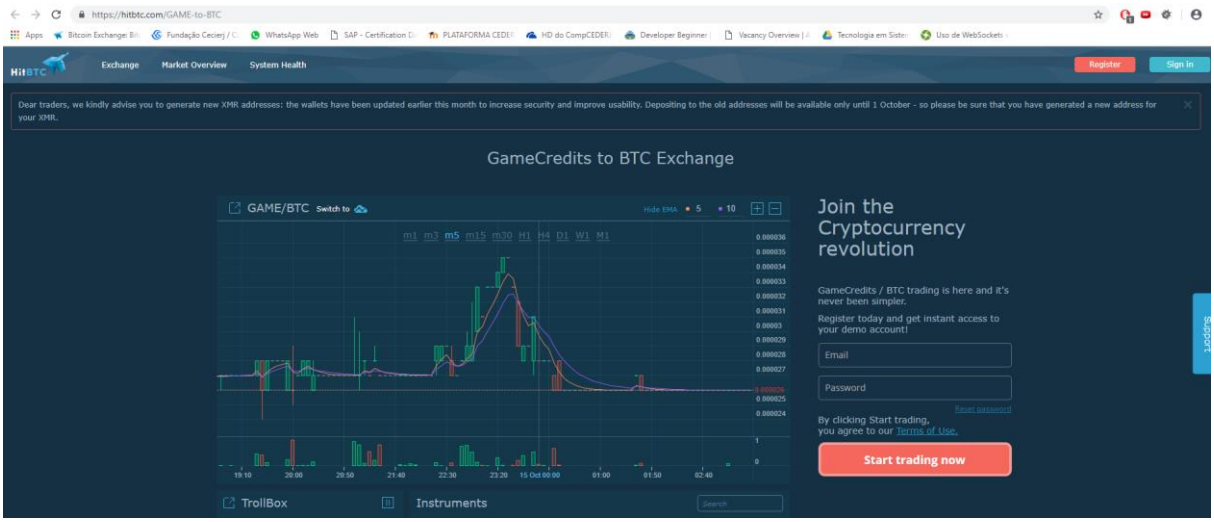

Figura 21 - Site HitBTC para a moeda GAME-BTC

5 TECNOLOGIAS UTILIZADAS NA APLICAÇÃO

O Desenvolvimento da aplicação como dito anteriormente se fará na linguagem Python3 isso se dará pois através dela foi possível colocar em prática parte do conteúdo abordado durante a extensão desse curso de tecnologia de sistemas de computação além de ser uma das linguagens capazes de utilizar as API's fornecidas pela HitBTC, corretora adotada como modelo para a criação desse protótipo.
É interessante ressaltar que não será utilizado nenhum banco de dados para o desenvolvimento da aplicação, uma vez que todos os dados necessários são processados em memória durante o tempo de execução. Como o programa não necessita armazenar nenhum tipo de informação (seja de usuários ou de aplicação) para futuros acessos não se faz necessário a criação de um banco de dados.

5.1 Estrutura da lógica de programação

Basicamente a lógica do programa consistirá primeiramente na importação da biblioteca "*requests*" que contém os métodos necessários para acessar a API, posteriormente o programa irá chamar o método "*get*" presente na API para poder solicitar as informações referentes às criptomoedas.

Para melhor separação todos esses métodos utilizados da biblioteca *requests* serão chamados dentro dos métodos da classe *APIQuerry* definida anteriormente no diagrama de classes, como a classe responsável por estabelecer a comunicação entre a HitBTC com a aplicação.

Posteriormente todas as moedas ativas da corretora serão armazenadas em uma lista e para cada uma dessas moedas (em um *loop*) será chamado um método para buscar os respectivos *tickers* para cada *currencie* (moeda), como a aplicação ao ser roda em um *loop* infinito constantemente monitorando os preços cada *currencie* terá vários *tickers* associados a ela, e através desses *tickers* quer serão

feitas as comparações. Cada iteração do loop com as buscas de *tickers* será programada para acontecer após 60 segundos do início da anterior, sendo assim é imprescindível que o sistema consiga realizar os cálculos necessários em menos de 60 segundos como dito anteriormente.

Ao término de cada iteração o sistema irá armazenar os dados obtidos em 4 tabelas distintas uma para cada periodicidade possível, ou seja, a cada um minuto um dado novo será inserido nessa tabela para cada moeda ativa da HitBTC, sendo assim a tabela de que armazenará os dados de um minuto sempre terá dois registros para cada moeda, enquanto que já na tabela de cinco minutos existirão cinco registros para cada moeda.

Após o preenchimento dessas tabelas o programa terá o subsídio necessário para realizar as comparações os parâmetros de alerta cadastrado pelo usuário presente em memória estarão sendo testados constantemente monitorando um possível disparo de alerta.

Quando o alerta em questão é acionado o programa irá acessar outra API da HitBTC dessa vez para resgatar as informações de *candles* necessárias para a formação do gráfico.

Por fim será exibido no navegador do usuário a criptomoeda que teve o alerta disparado informando valores de preços de compra e venda além de suas variações conforme mostrados na Figura 14.

6 CONCLUSÕES E TRABALHOS FUTUROS

Com o desenvolvimento de toda a etapa de levantamento de requisitos e do protótipo exibido nesse trabalho será possível futuramente colocar em prática o scanner de criptomoedas que servirá como uma aplicação de apoio nas decisões de comercialização de ativos dessa natureza.

Apesar do protótipo descrito no trabalho considerar apenas as informações da corretora HitBTC é possível a ampliação para as mais diversas corretoras criptomoedas no futuro como *Cryptopia, Poloniex, Binance* além de outras.

Com o desenvolvimento desse trabalho também foi possível aperfeiçoar alguns dos principais conceitos desenvolvidos ao longo desse curso de tecnologia em sistemas de computação ministrado pelo Cederj.

REFERÊNCIAS BIBLIOGRÁFICAS

[1] [Online]. Available: http://www.metzdowd.com/mailman/listinfo/cryptography.

[2] “coinmarketcap,” CoinMarketCap, 2018. [Online]. Available: https://coinmarketcap.com/charts/. [Acesso em 14 10 2018].

[3] “api.hitbtc,” hitBTC, [Online]. Available: https://api.hitbtc.com/#market-data. [Acesso em 02 10 2018].

[4] “api.hitbtc,” HitBTC, [Online]. Available: https://api.hitbtc.com/api/2/public/ticker. [Acesso em 10 2018].

[5] “api.hitbtc,” HitBTC, [Online]. Available: https://api.hitbtc.com/api/2/public/currency. [Acesso em 10 2018].

[6] “ativainvestimentos,” Ativa Investimentos, [Online]. Available: https://ativainvestimentos.com.br/comunicacao/outros/blog-ativa/05-primeiros-passos-analise-tecnica.pdf. [Acesso em 10 2018].

[7] “hitbtc.com,” HitBTC, 27 09 2018. [Online]. Available: https://hitbtc.com/exchange/ETH-to-BTC. [Acesso em 27 09 2018].

[8] “HitBTC,” HitBTC, [Online]. Available: https://api.hitbtc.com/api/2/public/candles/). [Acesso em 14 10 2018].

[9] “HitBTC,” HitBTC, [Online]. Available: https://api.hitbtc.com/api/2/public/ticker. [Acesso em 14 10 2018].

[10] . A. J. Soares e M. Tibo, “PMBOK - Project Management Body of Knowledge - PORTUGUÊS,” Project Management Institute Brazil Minas Gerais Chapter, 2000.

[11] G. Booch, J. Rumbaugh e I. Jacobson, Uml - Guia do Usuário, Elsevier Editora Ltda, 2012.

[12] D. Tybel, “devmedia,” 2018. [Online]. Available: https://www.devmedia.com.br/orientacoes-basicas-na-elaboracao-de-um-diagrama-de-classes/37224. [Acesso em 14 10 2018].

[13] R. S. Pressman, Engenharia de Software - Uma abordagem Profissional, AMGH editora ltda, 2011.

[14] ABOUT HitBTC API, “https://hitbtc.com,” 09 08 2018. [Online]. Available: https://api.hitbtc.com/.

[15] B. Prableen, “investopedia,” 5 10 2018. [Online]. Available: https://www.investopedia.com/tech/most-important-cryptocurrencies-other-than-bitcoin/. [Acesso em 10 2018].