



Instituto Politécnico de Tomar

Computação Distribuída

Engenharia Informática

2022/2023

SafetyRate



SAFETYRATE

novembro de 2022

Índice



1	Grupo de Trabalho	1
2	Introdução	2
2.1	Estado da arte	3
2.1.1	Descrição da Blockchain	3
2.1.2	Blocos e transações	3
2.1.3	Rede P2P	4
2.1.4	Bases de dados distribuídas	4
2.1.5	Algoritmos de consenso	5
2.2	Vantagens	6
2.2.1	Imutabilidade	6
2.2.2	Transparência	6
2.2.3	Censura.....	6
2.2.4	Rastreabilidade	7
2.3	Desvantagens.....	7
2.3.1	Imutabilidade	7
2.3.2	Custo de implementação.....	7
2.3.3	Velocidade e performance	7
2.3.4	Custo energético	7
2.4	Aplicações e projetos baseados em blockchain	8
3	Descrição do projeto	9
4	Trabalho 1 - Blockchain.....	10
4.1	Arquitetura da aplicação	10
4.2	Bibliotecas	11
4.2.1	Serializable.....	11
4.2.2	FileInputStream e FileOutputStream	11
4.2.3	ObjectOutputStream e ObjectInputStream	11
4.2.4	Concurrent.CopyOnWriteArrayList	11
4.3	Classes.....	12
4.3.1	mainMenu	12
4.3.2	Blockchain.....	13
4.3.3	Block	14
4.3.4	Miner	15
	○ Esta classe contém todo o código das threads mineradoras, é assim crucial para o bom funcionamento da aplicação.	15
	○ Diagrama detalhado da classe	15
4.3.5	Hash.....	16
4.4	Aplicação	18
5	Trabalho 2 – Computação multitarefa	21
5.1	Arquitetura da aplicação.....	21
5.2	Bibliotecas	22
5.2.1	Serializable.....	22
5.2.2	FileInputStream e FileOutputStream	22
5.2.3	ObjectOutputStream e ObjectInputStream	22
5.2.4	Concurrent.CopyOnWriteArrayList	22
5.3	Classes.....	23

5.3.1	mainMenu	23
5.3.2	Blockchain.....	24
5.3.3	Block	25
5.3.4	Miner	26
○	Esta classe contém todo o código das threads mineradoras, é assim crucial para o bom funcionamento da aplicação.	26
○	Diagrama detalhado da classe	26
5.3.5	Hash.....	27
5.4	Aplicação.....	28
6	Trabalho 3 – Computação distribuída	31
6.1	Arquitetura da aplicação.....	31
6.2	Biblioteca 1	31
6.2.1	Classe A.....	31
6.2.2	Classe B.....	31
6.3	Aplicação.....	31
7	Limitações e Desenvolvimentos Futuros.....	33
8	Conclusão	34
9	Referencias	35
9.1	Livros.....	35
9.2	Documentos	35
9.3	Referência web.....	35

Figuras

Figura 1 - Alix Carmo	1
Figura 2 - André Almeida.....	1
Figura 3 – Imagem representativa da Blockchain	2

1 Grupo de Trabalho

 <i>Figura 1 - Alix Carmo</i>	Número : 23324 Nome : Alix António Pestana Carmo Curso : Engenharia Informática Turma : B Email : aluno23324@ipt.pt
 <i>Figura 2 - André Almeida</i>	Número : 23049 Nome : André Filipe Santos Almeida Curso : Engenharia Informática Turma : B Email : aluno23049@ipt.pt

Declaração:

Os alunos declaram sob compromisso de honra que o projeto descrito neste relatório é original e da sua autoria com exceção dos seguintes elementos:

Descrição	Fonte
Class Hash	https://doctrino.ipt.pt/mod/resource/view.php?id=63390
Classe User	Teams da UC
Classe Autentication	Teams da UC
Classe SecurityUtils	Teams da UC

Assinaturas:

Alix Carmo

André Almeida

2 Introdução

No âmbito da Unidade Curricular (UC) de Computação Distribuída do terceiro ano da Licenciatura em Engenharia Informática na Escola Superior de Tecnologias de Tomar, pertencente ao Instituto Politécnico de Tomar, foi apresentado, pelo docente António Manso, aos alunos desta UC o projeto a realizar.

Este projeto consiste em criar uma aplicação distribuída baseada em *blockchain*, implementando um serviço baseado em confiança sobre a tecnologia supracitada, tirando sempre partido dos conceitos estudados ao longo do semestre nesta UC.

A implementação desta aplicação divide-se em três partes:

- Numa primeira fase o objetivo é realizar uma investigação sobre o estado da arte na área de aplicações baseadas em *blockchain*, adquirir conhecimento sobre este recurso tecnológico e deste modo apresentar aplicações e projetos que o utilizem. De seguida e com base na pesquisa efetuada, deve ser criado, ou replicado um serviço existente, que implemente uma *blockchain* com recurso à linguagem de programação Java.
- Nesta componente o objetivo é efetuar um melhoramento nas interfaces para a exploração da aplicação, ou seja, as interfaces devem ser não bloqueantes e os serviços são executados de forma assíncrona e procurando explorar as capacidades de processamento paralelo sempre que tal se justifique.
- Por fim, pretende-se que a aplicação desenvolvida até então seja implementada numa rede de computadores para que o serviço seja distribuído numa rede *peer-to-peer*, ou rede ponto a ponto, (P2P) e devem ser implementados mecanismos de sincronização e consenso da *blockchain*.

Deste modo, os redatores deste relatório visam descrever todo o trabalho efetuado ao longo do projeto, assim como os triunfos e as nossas adversidades, sendo que este relatório abordará a primeira e a segunda fase do projeto.

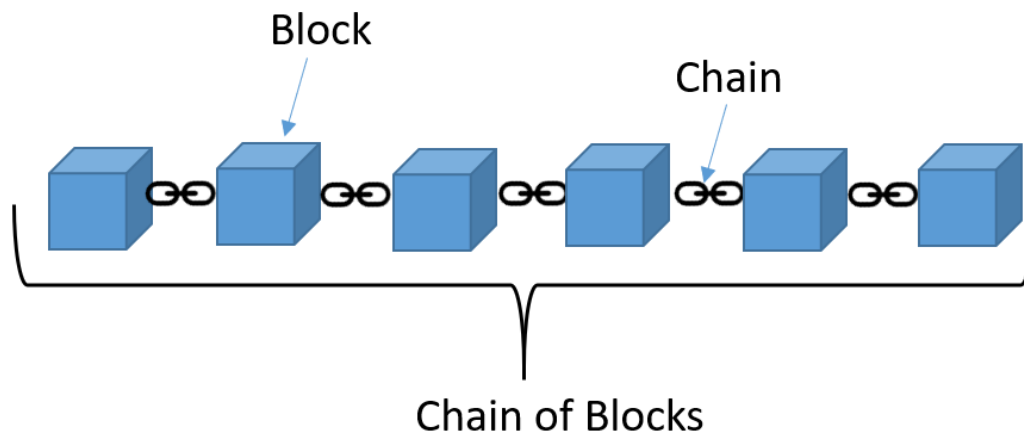


Figura 3 – Imagem representativa da Blockchain

2.1 Estado da arte

A *blockchain* trata-se de uma base de dados descentralizada e distribuída, onde guarda permanentemente um registo de informações de modo que a violação desta seja praticamente impossível [W1]. Segurança deste nível é utópico, sendo que ela é alcançada pela ligação de blocos de informação, entre si, que se validam mutuamente, para que quando seja efetuada uma alteração num dos blocos os restantes blocos verifiquem e aprovelem essa mesma alteração.

Inicialmente descrita em 1982 por Stuart Haber e W. Scott Stornetta [W4], a primeira *blockchain* implementada só aconteceu com a criação da bitcoin. Foi em 2008 que foi conceitualizado por Satoshi Nakamoto a primeira *blockchain* descentralizada e, no ano seguinte, surge a sua implementação na bitcoin e minerou os primeiros bitcoins. [W5]

Desde esse momento, devido ao seu sucesso nas moedas digitais, que a *blockchain* recebeu o interesse de empresas, bancos e organizações governamentais pela sua fiabilidade e versatilidade., surgindo assim alguns anos depois, por volta de 2014/2015, o conceito de *blockchain 2.0*, que se diferencia da *blockchain* original por abranger as suas capacidades para além da descentralização de dinheiro e pagamentos [W6]. Assim, esta nova versão da *blockchain* permitiu a facilitação da descentralização de mercados em geral, como se pode observar algumas das suas aplicações no tópico [2.2 Aplicações e projetos baseados em blockchain](#). Por vezes, as utilizações desta blockchain em aplicações podem necessitar de estar em conformidade com a legislação legal em cada país onde está a ser aplicada, pelo que pode surgir um impedimento na sua implementação.

2.1.1 Descrição da Blockchain

A principal tecnologia utilizada neste projeto é a *blockchain*. Trata-se de uma base de dados descentralizada e distribuída que funciona numa rede *peer-to-peer* (P2P) e que contem informações.

2.1.2 Blocos e transações

A *blockchain* pode ser resumida como sendo uma cadeia de blocos que contém informação, podendo ser comparados com um extrato bancário, pois os mesmo mantêm registo das transações validas efetuadas. Os blocos são encriptados e colocados numa *hash tree* ou *merkle tree*. [W7]

Os blocos são distintos entre si. Os blocos quando são finalizados geram um *hash*, que é o *hash* do block, mas o mesmo contém outro hash, o do bloco anterior. Deste modo, o bloco que acabou de ser completado esta ligado ao anterior, assim como o seguinte está ligado ao atual, tornando-se assim parte da corrente de blocos já existente. (ver imagem seguinte para melhor esclarecimento) Além do mais, um

bloco quando da sua finalização, a transação é verificada por toda a rede várias vezes e após conformação da mesma, esta é adicionada ao bloco e torna-se imutável, ou seja, os dados dos blocos anteriores e do atual não podem ser alterados e todo o sistema guarda a informação permanentemente e distribuí por toda a rede o registo de transações.

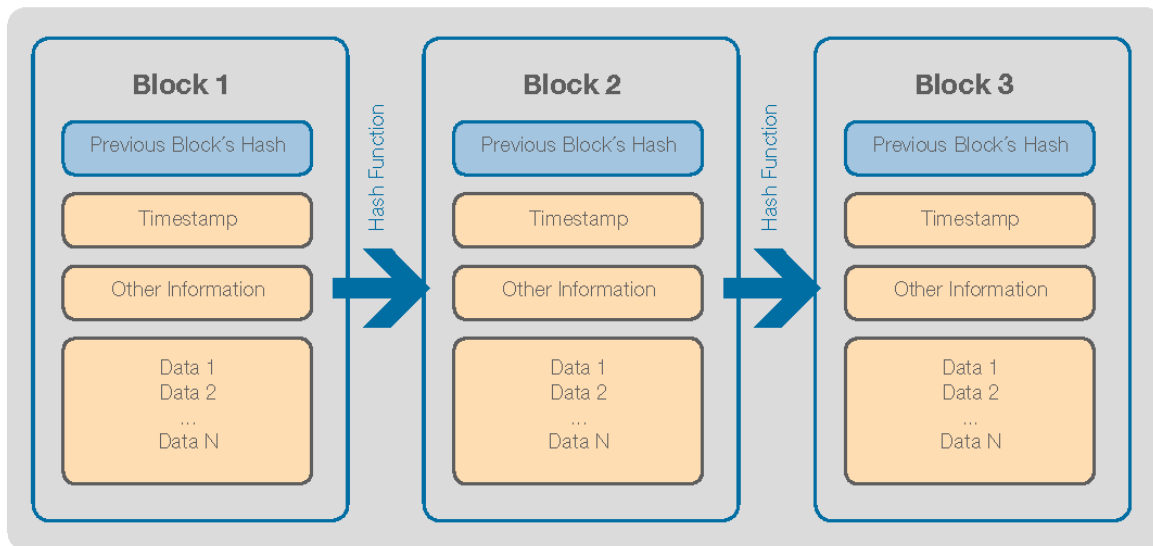


Figura 4 – Estrutura e funcionamento dos blocos da blockchain

2.1.3 Rede P2P

A Rede P2P é uma arquitetura de computadores ou redes que compartilham entre os peers tarefas, arquivos, ... entre si.

Os peers são parceiros na rede que possuem privilégios e influencia iguais no ambiente em que se encontram. Cada peer é chamada de nó na rede P2P, ou seja, uma rede P2P é uma rede de nós.

A rede P2P na blockchain consiste numa arquitetura de computadores e servidores onde cada um atua como um nó na rede. Quando uma nova mensagem entra na rede, esta propaga-se por todos os nós, sendo a mesma encriptada e privada, não havendo possibilidade de verificar quem a colocou na rede, mas apenas a sua validade.

Quando um nó abandona a rede, a mesma garante que não existe falhas devido ao facto de todos os outros nós possuírem uma copia exatamente igual para todos de toda a informação compartilhada.

2.1.4 Bases de dados distribuídas

Dado que a rede é descentralizada isto garante que não existe falhas no sistema. Do mesmo modo que quando um nó abandona um nó da rede esta não é abalada, como foi supracitado no ponto anterior, o inverso também se sucede, ou seja,

quando entra um nó na rede o mesmo recebe uma cópia exata da informação compartilhada até ao momento, passando a ser um nó da rede a partir desse momento.

2.1.5 Algoritmos de consenso

O algoritmo de consenso é uma parte importante do processo da formação e da adição de um novo bloco à rede. Este é utilizado para resolver os problemas de confiança que existem com a informação, ou seja, uma informação não pode ser apagada e deve ser confirmada e acreditada por toda a rede. Para tal, deve ser utilizado um algoritmo que permita a inclusão de novos dados.

Assim, um bloco só é adicionado à cadeia de blocos existente se seguir um algoritmo de consenso, sendo os mais comuns os seguintes: [W8]

2.1.5.1 PoW

Este algoritmo provoca uma competição entre os computadores ligados a blockchain para ver quem será o primeiro a encontrar um hash respectivo ao novo bloco considerando a dificuldade imposta, fazendo com que os computadores com mais poder computacional tenham maiores probabilidades de “minerar” um novo bloco

2.1.5.2 PoS

A mineração de novos blocos por este algoritmo ocorre pela votação dos participantes da rede que possuem “ativos digitais” (da Blockchain que o utiliza), onde as recompensas e o peso dos votos estão diretamente relacionados com a quantidade de ativos de cada participante.

2.1.5.3 DPoS

Este algoritmo é similar ao PoS, porém os participantes não votam diretamente se um bloco é válido ou não, ou seja, os participantes votam em Delegados e Testemunham que votarão por eles.

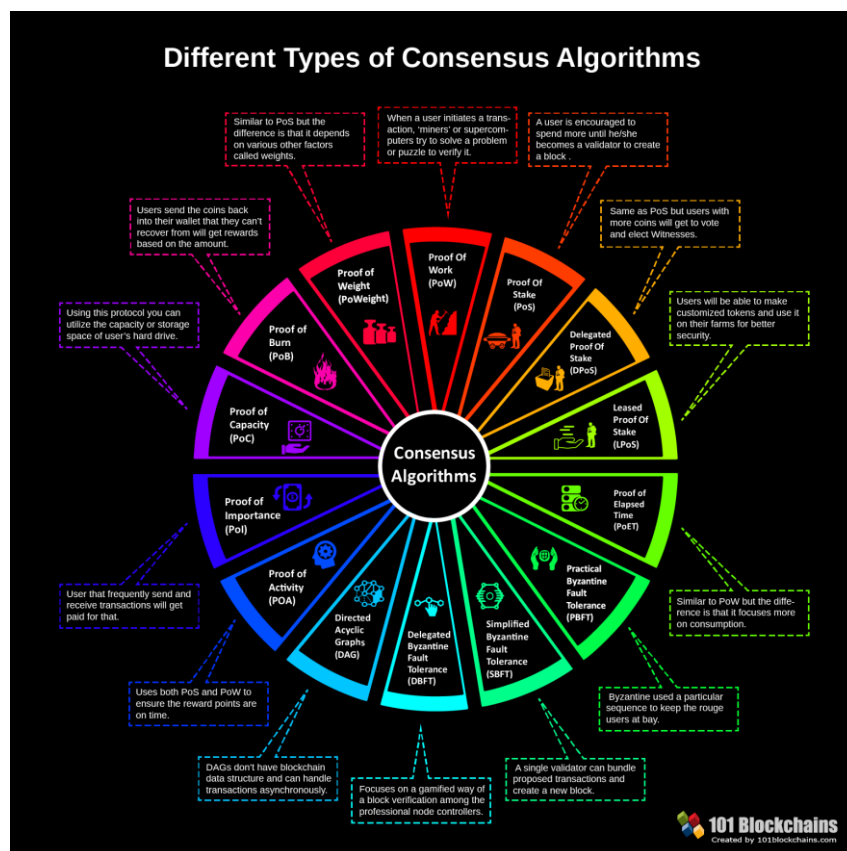


Figura 5 – Os vários tipos de algoritmos de consenso

2.2 Vantagens

2.2.1 Imutabilidade

A *blockchain* é imutável, ou seja, é impossível alterar ou apagar dados que já estejam gravados na mesma. Isto mitiga assim a adulteração de dados na rede e mostra resultados ao nível de segurança superior as bases de dados convencionais.

2.2.2 Transparência

Qualquer membro da rede pode verificar os dados da *blockchain*, permitindo assim uma maior confiança neste serviço por parte do público. Por outro lado, as bases de dados tradicionais não permitem a verificação dos dados por parte de qualquer utilizador, apenas podem verificar os dados que o administrador da base de dados permitir.

2.2.3 Censura

A *blockchain*, ao contrário de outras base de dados, é livre de censura dado que não existe controlo algum dos dados na mesma e ninguém nem nenhuma autoridade pode interromper operações na rede. As restantes bases de dados possuem, por norma, uma autoridade central a regular a rede, e podem remover ou alterar informação, ou seja, exercem assim a censura.

2.2.4 Rastreabilidade

A blockchain cria, através de *hash*, uma maneira simples e rápida de detetar alterações na rede.

2.3 Desvantagens

2.3.1 Imutabilidade

Dado a impossibilidade de efetuar alterações dos dados na rede, em caso de erro é necessário reescrever os blocos todos para poder efetuar uma alteração e passar pelo processo de verificação novamente, pelo que representa um alto consumo de tempo e dinheiro.

2.3.2 Custo de implementação

A *blockchain*, ao contrário da base de dados tradicional, não é de implementação direta, dado que é necessário efetuar um plano para ser possível integrar a *blockchain* no projeto em questão.

2.3.3 Velocidade e performance

Devido a redundância da rede onde cada nó tem de verificar e aprovar cada transação e alterações na rede e guardar a mesma para si e ao sistema ter de verificar as assinaturas dos utilizadores, a velocidade da blockchain não é a mais rápida, comparado a outras base de dados.

2.3.4 Custo energético

Devido principalmente aos pontos 2.3.2 e 2.3.3, a utilização da blockchain mostra-se como um problema para um ambiente dado que cada verificação no nó e cada peça de hardware representam mais um consumo energético, contribuindo assim indiretamente para o degradar do planeta. [W9]

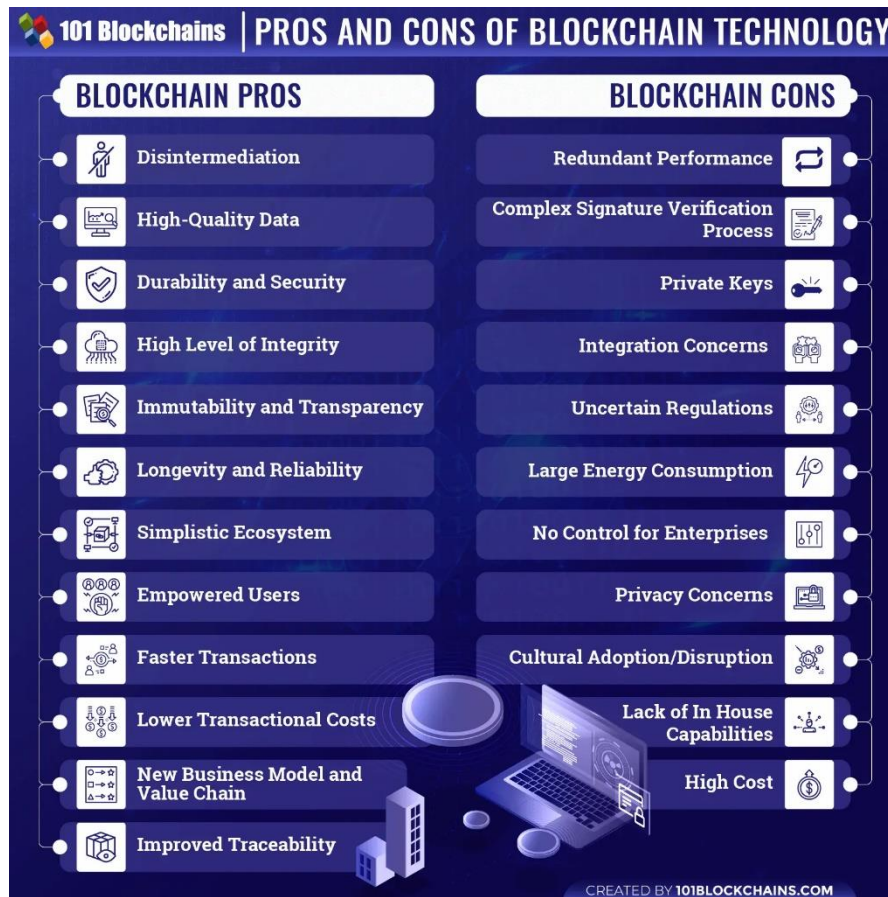


Imagem – Vantagens e desvantagens da blockchain

2.4 Aplicações e projetos baseados em blockchain

Como já foi referido anteriormente, a *blockchain* possui uma vasta capacidade de implementação em várias indústrias (ver figura 4), sendo algumas das aplicações as seguintes [W2][W3]:

- Moedas digitais;
- Sistemas de pagamentos
- Contratos Inteligentes
- Educação
- Armazenamento em *cloud*
- Saúde
- *Supply Chains* atualizadas
- Turismo
- Vendas em Marketplace
- Identificação dos utilizadores
- Aplicações governamentais
- Finanças.

3 Descrição do projeto

A SafetyRate consiste numa aplicação onde utilizadores poderão avaliar websites que visitem de forma a partilhar a sua experiência com outros utilizadores.

O Utilizador poderá avaliar o website com uma escala de avaliação que varia de 0 a 10, sendo 0 uma má experiência, e 10 uma boa experiência.

Para proceder á avaliação de um website serão recolhidas várias informações do utilizador para garantir que a sua avaliação não é repetida, isso é feito com o objetivo de garantir unicidade na avaliação para prevenir fraude ou manipulação de avaliações.

Cada vez mais a internet é utilizada como um meio malicioso de obter informações privadas, infetar dispositivos com trojans ou loggers, e outro tipo de ações que podem afetar negativamente um utilizador.

Como tal, a SafetyRate possibilita uma forma de verificar a credibilidade de um website, visto que cada avaliação feita a um website é única e muito dificilmente manipulada, todas as avaliações que sejam feitos através da SafetyRate poderão ser utilizadas de forma segura para decidir a utilização de um website.

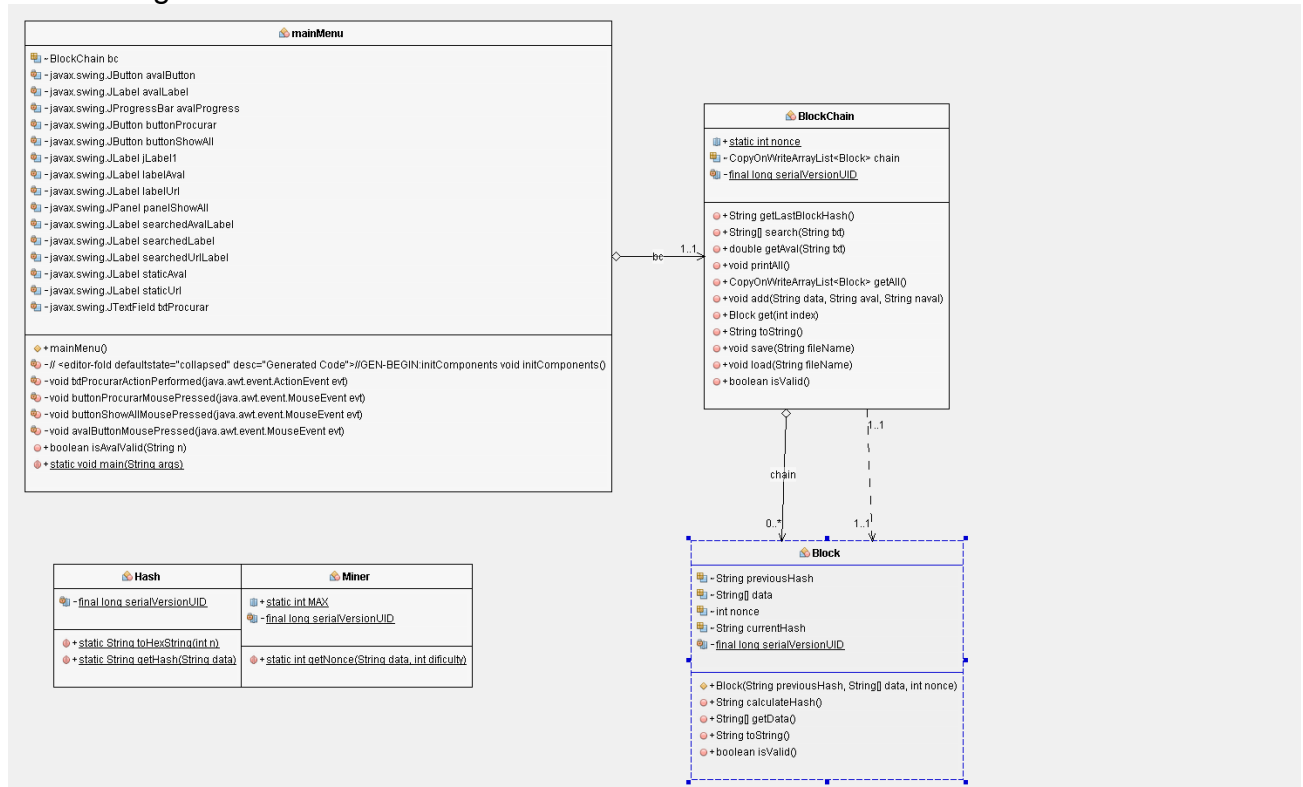
Existem vários projetos pertinentes que se assemelham à SafetyRate, alguns exemplos são:

- [TrustPilot](#) – Website onde utilizadores podem avaliar e descrever a sua experiência de um website ou empresa.
- [SiteJabber](#) – Website semelhante ao TrustPilot mas que também oferece a possibilidade de avaliar serviços individuais e não apenas empresas ou websites.
- [VirusTotal](#) – Website que confirma se outros websites ou ficheiros são seguros pela verificação da existência de *software* malicioso.

4 Trabalho 1 - Blockchain

4.1 Arquitetura da aplicação

- Diagrama de classes



- Como se encontra estruturado o código da aplicação:
 - Objetos e classes do sistema.
 - Classe mainMenu** – Classe principal da aplicação, é responsável pelo processamento de todas as interações com o GUI.
 - Classe Blockchain** – Classe responsável pela implementação de uma Blockchain, bem como a implementação de todos os métodos utilizados em volta da Blockchain.
 - Classe Block** – Classe responsável pela implementação dos blocos da Blockchain, implementa também vários métodos utilizados em cada bloco.
 - Classe Hash** – Classe que trata de calcular a hash de um número inteiro ou de uma string.
 - Classe Miner** – Classe responsável pelo processamento do “Nonce”.

- Descrição geral das bibliotecas
 - As bibliotecas utilizadas:
 - Serializable;
 - FileInputStream;
 - FileOutputStream;
 - ObjectInputStream;
 - ObjectOutputStream;
 - concurrent.CopyOnWriteArrayList.
 - recursos externos
 - Class Hash – Esta class é utilizada para obter uma string de um número em base 16.

4.2 Bibliotecas

4.2.1 Serializable

- Descrição geral da biblioteca

A biblioteca Serializable permite salvar o estado atual dos objetos em arquivos binários no computador, podendo ser recuperado posteriormente recriando o objeto em memória tal como o mesmo se encontrava no momento da sua serialização.

4.2.2 FileInputStream e FileOutputStream

- Descrição geral da biblioteca

Estas bibliotecas são semelhantes entre si, trabalham com stream de bytes em arquivos. O FileInputStream lê os bytes de um ficheiro no sistema enquanto o FileOutputStream escreve bytes e dados para um ficheiro.

4.2.3 ObjectOutputStream e ObjectInputStream

- Descrição geral da biblioteca

Esta biblioteca permite guardar e efetuar a leitura de objetos serializados num ficheiro.

4.2.4 Concurrent.CopyOnWriteArrayList

- Descrição geral da biblioteca

Esta biblioteca implementa um ArrayList, mas é realizada uma nova cópia do mesmo após ser efetuado uma alteração no ArrayList.

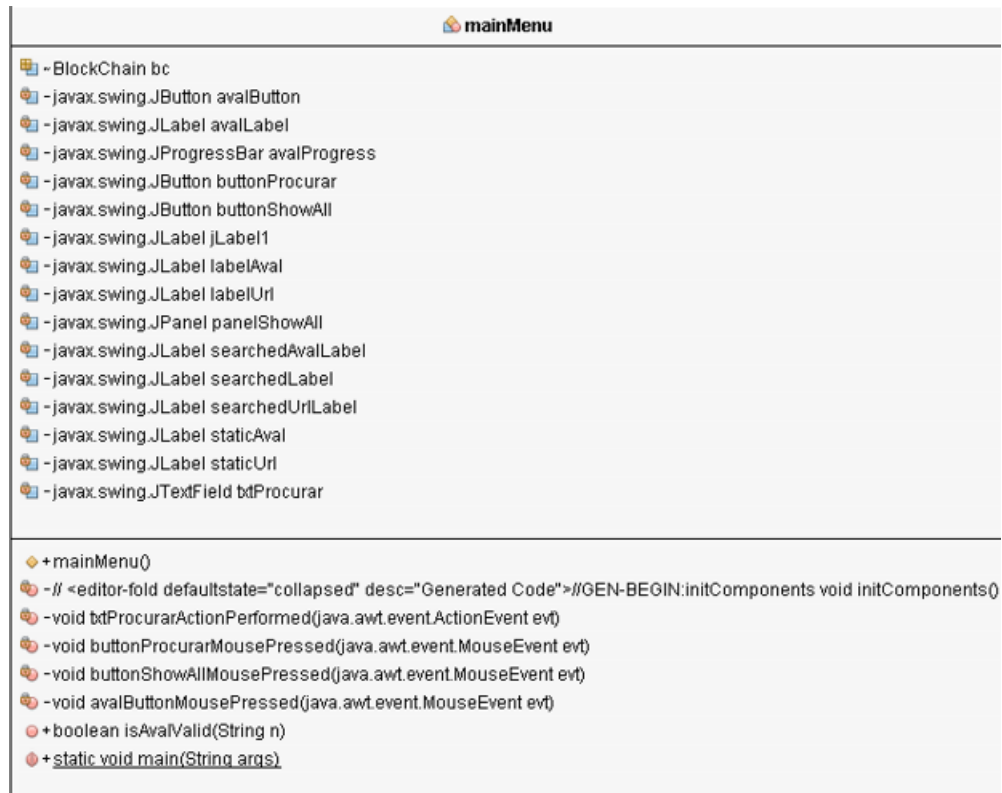
4.3 Classes

4.3.1 mainMenu

- Justificação da sua existência e a sua função no projeto.

Esta é a classe principal do projeto, sendo esta a responsável pela interface gráfica do mesmo.

- Diagrama detalhado da classe e as ligações a outras classes



- Classes que usa

Esta classe implementa e utiliza a classe Blockchain

- Descrição dos atributos

- Sendo uma interface gráfica, os atributos nela são todos os elementos necessários para a correta utilização da mesma. Logo os mesmo são praticamente botões, labels, etc.

- Descrição dos métodos

```
private void buttonProcurarMouseClicked(java.awt.event.MouseEvent evt) {
..}
```

- Este método procura na blockchain se o url inserido já existe.

```
private void avalButtonMouseClicked(java.awt.event.MouseEvent evt) {..}
```


- Este método permite avaliar um URL

```
public boolean isAvalValid(String n){..}
```

- Este método verifica se a String n está contida entre os valores 0.0 e 10.0

4.3.2 Blockchain

- Justificação da sua existência e a sua função no projeto.

Esta classe é crucial, pois a mesma define e a blockchain que irá ser utilizada ao longo do projeto

- Diagrama detalhado da classe e as ligações a outras classes



- Classes que usa

Esta classe utiliza a classe Block e implementa o Serializable.

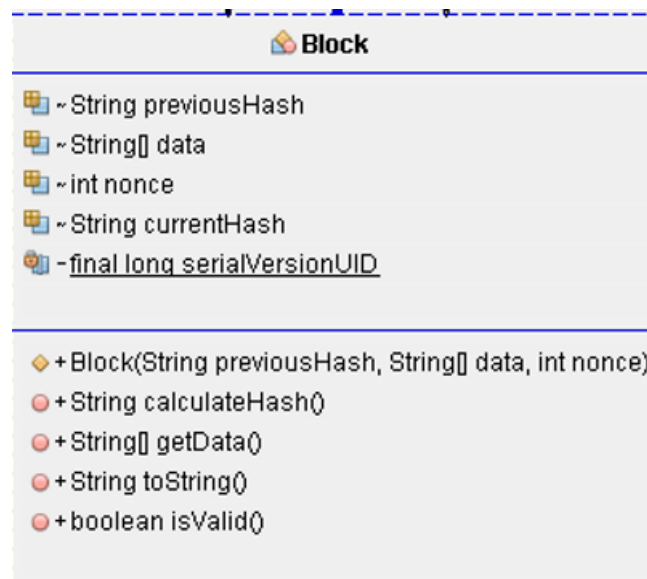
- Descrição dos métodos

- Retorno, nome, parâmetros, descrição
- `getLastBlockHash()` – este método retorna o hash do bloco anterior
- `search(String txt)` – tendo como parâmetro uma string txt, este método descobre se o txt já existe em algum bloco e retorna o URL

- `getAval(String txt)` – Tendo como parâmetro uma string txt, este método percorre a blockchain e procura em cada bloco se já existe o url e retorna a sua avaliação
- `printAll()` – Este método mostra o conteúdo dos blocos da blockchain
- `getAll()` – adiciona a um novo `arrayList` os urls presentes na blockchain
- `add(String data, String aval, String naval)` – Cria um novo bloco com o url, a avaliação e o número de avaliações e adiciona-o a blockchain
- `get(int index)` – Retorna o index de um bloco
- `save(string filename)` – guarda os objetos num ficheiro cujo nome é o atributo filename
- `load (string filename)` – carrega os objetos do ficheiro com o nome designado no atributo filename
- `isValid()` - valida os blocos da rede são válidos

4.3.3 Block

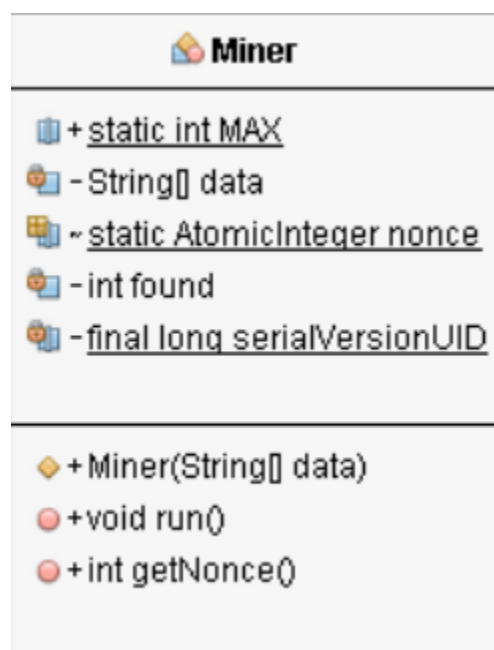
- Justificação da sua existência e a sua função no projeto.
 - Esta classe define um dos blocos da blockchain, é crucial para o funcionamento da mesma.
- Diagrama detalhado da classe e as ligações a outras classes



- Utiliza a classe Hash para calcular a hash do bloco
- Descrição dos atributos
 - String previousHash – Uma string onde está contida a hash do bloco anterior ao atual.
 - String currentHash – Uma string onde está contida a hash do bloco atual.
 - Int Nonce – nonce minerado do bloco atual
 - String[] data – array com os dados do bloco(URL e avaliação)
- Descrição dos métodos
 - Block(String previousHash, String[] data, int nonce) – Construtor dos blocos
 - String calculateHash() – calcula a hash do bloco atual
 - String[] getData() – retorna o array data do bloco atual
 - String toString() – retorna uma string que representa o bloco
 - Boolean isValid() – verifica se o bloco atual é válido através da sua hash

4.3.4 Miner

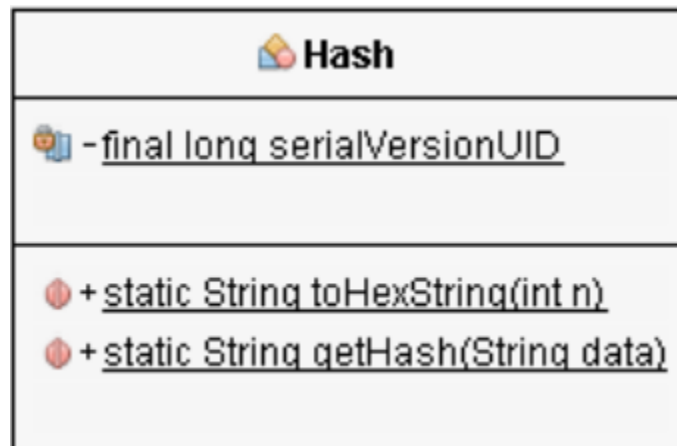
- Esta classe contém todo o código das threads mineradoras, é assim crucial para o bom funcionamento da aplicação.
- Diagrama detalhado da classe



- Descrição dos atributos
 - static int MAX – valor máximo do nonce
 - String[] data – array data do bloco a minerar
 - static AtomicInteger nonce – valor a encontrar que satisfaça o bloco atual.
 - int found – flag para controlo do nonce.
- Descrição dos métodos
 - Miner(String[] data) – cria um objeto miner com os dados de um bloco.
 - void run() – código a executar quando as threads mineradoras são iniciadas.
 - int getNonce() – retorna o último nonce minerado pelas threads.

4.3.5 Hash

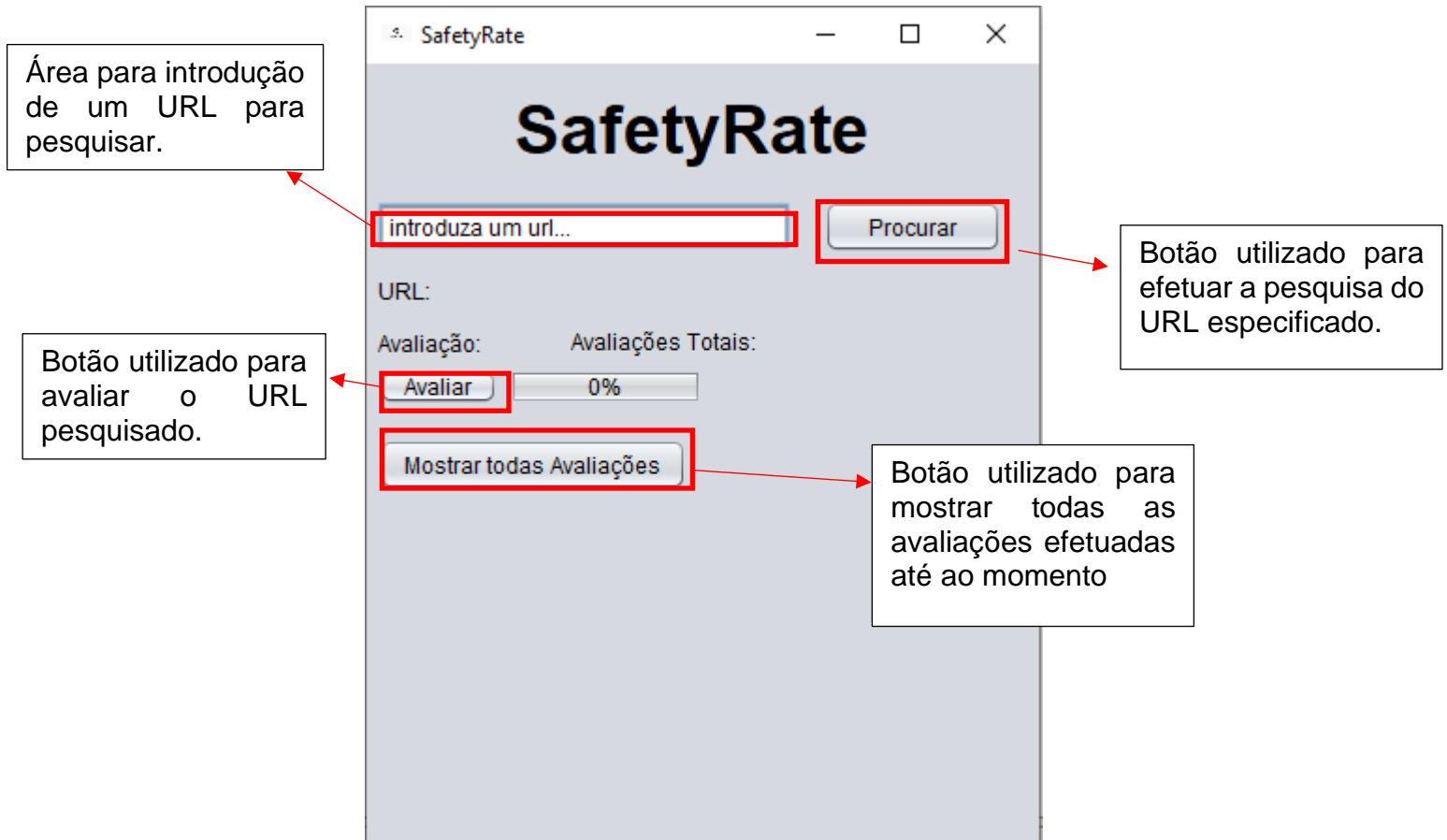
- Esta classe é utilizada para obter a hash de vários atributos e conjunto de atributos da aplicação
- Diagrama detalhado de classe



- Descrição dos métodos
 - static String toHexString(int n) – transforma um inteiro em uma string de base 16
 - static String getHash(String data) – obtém a hash de uma dada string

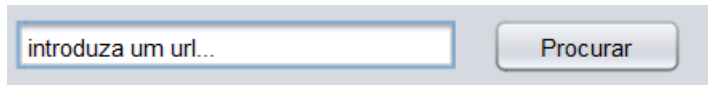
4.4 Aplicação

- Interfaces com o utilizador



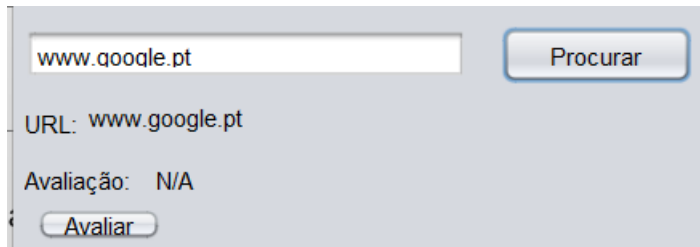
- Manual do utilizador

O utilizador começa por introduzir um URL no qual pretende efetuar uma pesquisa.

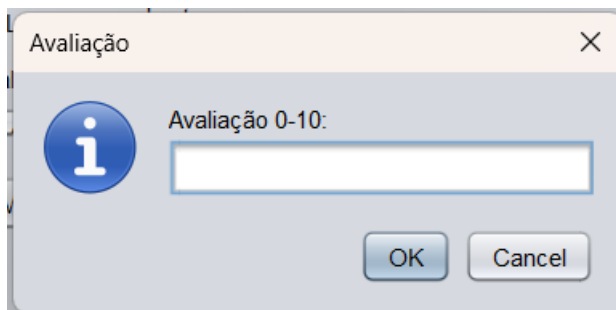


A screenshot of a web application interface. It features a text input field with the placeholder text "introduza um url..." and a button labeled "Procurar" to its right.

Após a pesquisa ser efetuada ele poderá ver a avaliação atual do mesmo e votar de 0.0 a 10.0 após clicar no botão "Avaliar".

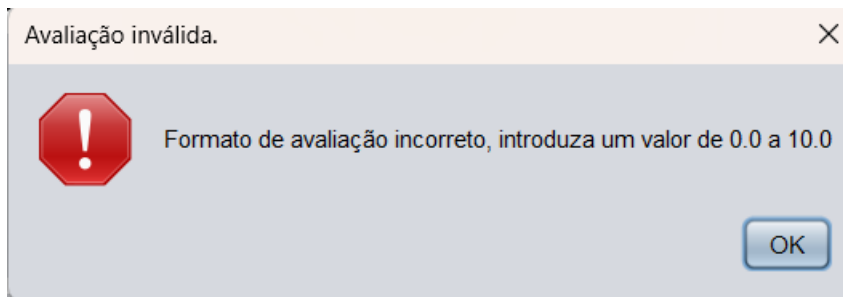


A screenshot of the application interface after a search. The text input field now contains "www.google.pt". Below the input field, it displays "URL: www.google.pt" and "Avaliação: N/A". A button labeled "Avaliar" is positioned at the bottom left, and the "Procurar" button remains at the top right.



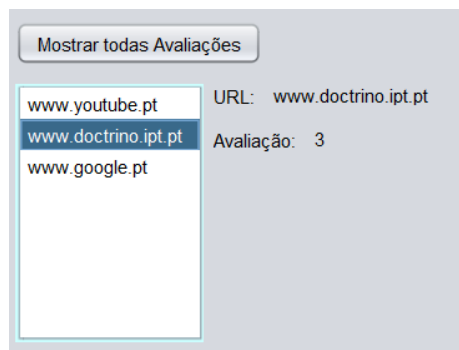
A screenshot of a dialog box titled "Avaliação". It contains an information icon (a blue circle with a white 'i') on the left. To the right of the icon, the text "Avaliação 0-10:" is displayed above a text input field. At the bottom right of the dialog, there are two buttons: "OK" and "Cancel".

Caso a avaliação não seja um valor neste intervalo ou um número, surge uma mensagem de erro.



A screenshot of an error dialog box titled "Avaliação inválida.". It features a red octagonal warning icon with a white exclamation mark on the left. To the right of the icon, the text "Formato de avaliação incorreto, introduza um valor de 0.0 a 10.0" is displayed. An "OK" button is located at the bottom right of the dialog.

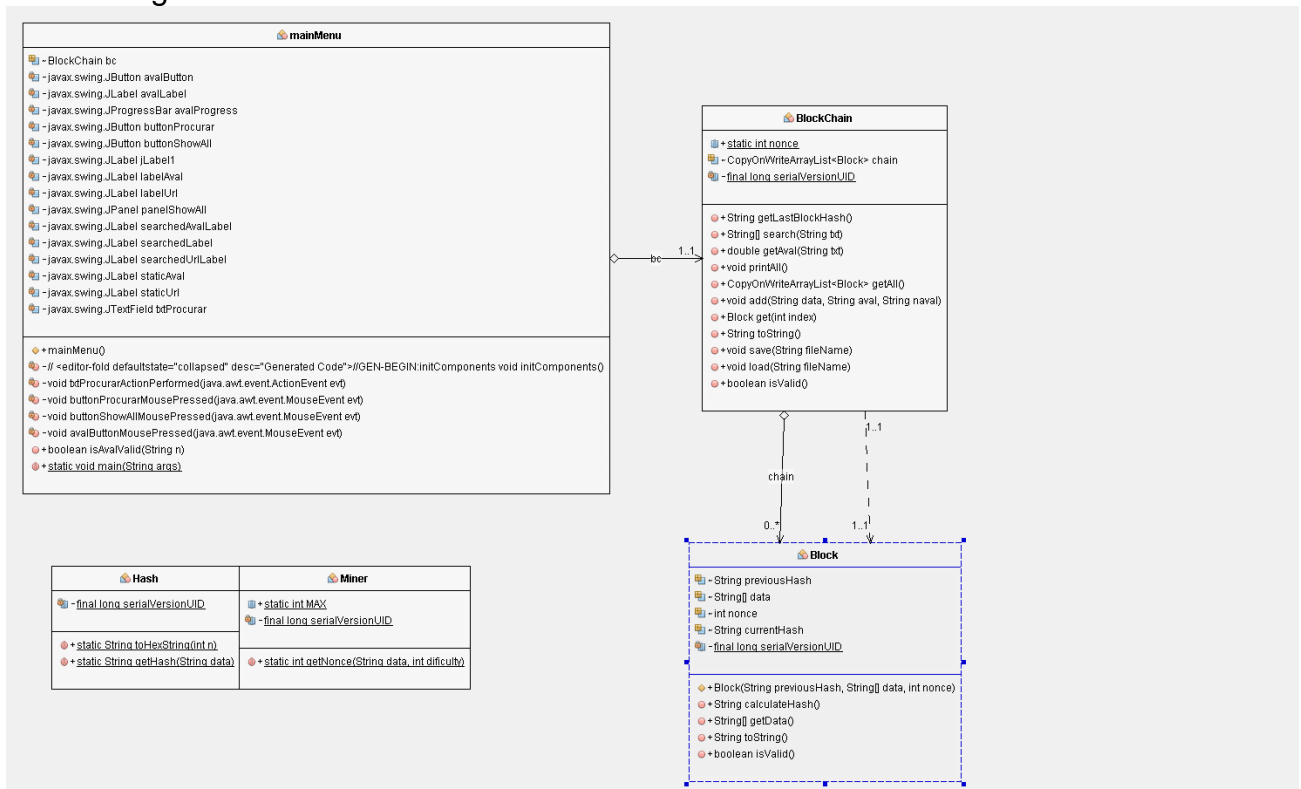
Ao carregar no botão “Mostrar todas avaliações” irá ser disponibilizado todos os URL’s pesquisados até ao momento e as respetivas avaliações.



5 Trabalho 2 – Computação multitarefa

5.1 Arquitetura da aplicação

- Diagrama de classes



- Como se encontra estruturado o código da aplicação:
 - Objetos e classes do sistema.
 - Classe mainMenu** – Classe principal da aplicação, é responsável pelo processamento de todas as interações com o GUI.
 - Classe Blockchain** – Classe responsável pela implementação de uma Blockchain, bem como a implementação de todos os métodos utilizados em volta da Blockchain.
 - Classe Block** – Classe responsável pela implementação dos blocos da Blockchain, implementa também vários métodos utilizados em cada bloco.
 - Classe Hash** – Classe que trata de calcular a hash de um número inteiro ou de uma string.
 - Classe Miner** – Classe responsável pelo processamento do “Nonce”.

- Descrição geral das bibliotecas
 - As bibliotecas utilizadas:
 - Serializable;
 - FileInputStream;
 - FileOutputStream;
 - ObjectInputStream;
 - ObjectOutputStream;
 - concurrent.CopyOnWriteArrayList.
 - recursos externos
 - Class Hash – Esta class é utilizada para obter uma string de um número em base 16.

5.2 Bibliotecas

5.2.1 Serializable

- Descrição geral da biblioteca

A biblioteca Serializable permite salvar o estado atual dos objetos em arquivos binários no computador, podendo ser recuperado posteriormente recriando o objeto em memória tal como o mesmo se encontrava no momento da sua serialização.

5.2.2 FileInputStream e FileOutputStream

- Descrição geral da biblioteca

Estas bibliotecas são semelhantes entre si, trabalham com stream de bytes em arquivos. O FileInputStream lê os bytes de um ficheiro no sistema enquanto o FileOutputStream escreve bytes e dados para um ficheiro.

5.2.3 ObjectOutputStream e ObjectInputStream

- Descrição geral da biblioteca

Esta biblioteca permite guardar e efetuar a leitura de objetos serializados num ficheiro.

5.2.4 Concurrent.CopyOnWriteArrayList

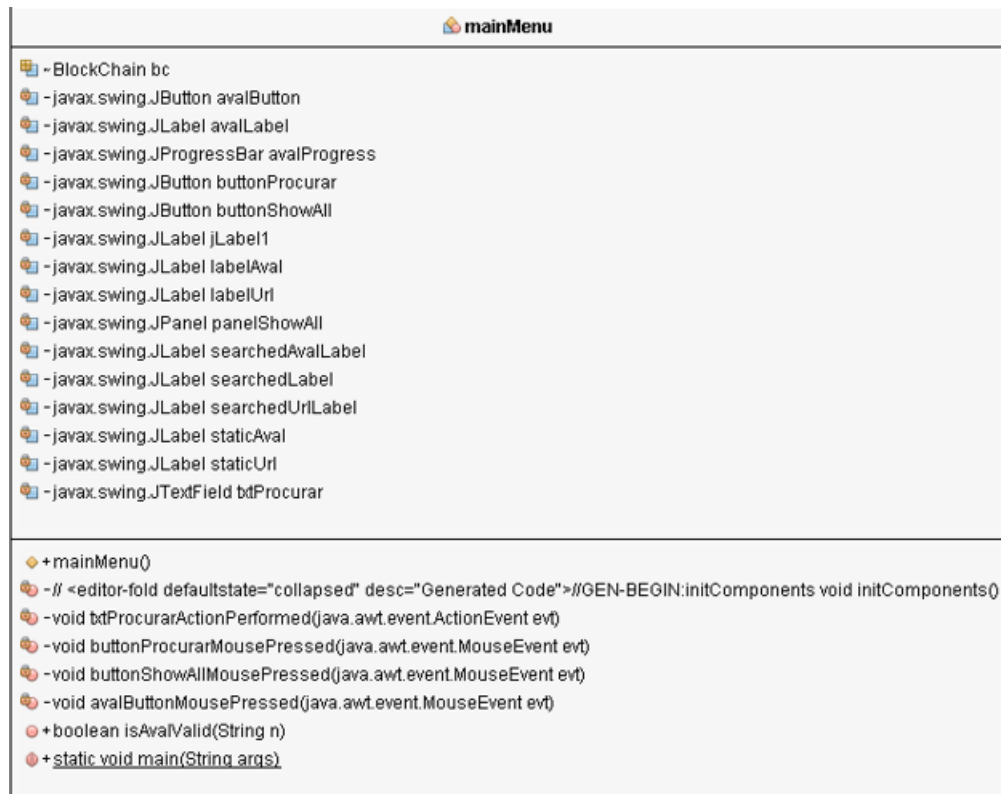
- Descrição geral da biblioteca

Esta biblioteca implementa um ArrayList, mas é realizada uma nova cópia do mesmo após ser efetuado uma alteração no ArrayList.

5.3 Classes

5.3.1 mainMenu

- Esta é a classe principal do projeto, sendo esta a responsável pela interface gráfica do mesmo.
- Diagrama detalhado da classe e as ligações a outras classes



- Classes que usa

Esta classe utiliza a classe Blockchain

- Descrição dos atributos
 - Sendo uma interface gráfica, os atributos nela são todos os elementos necessários para a correta utilização da mesma. Logo os mesmo são praticamente botões, labels, etc.
- Descrição dos métodos

```
private void buttonProcurarMousePressed(java.awt.event.MouseEvent evt) {
..}

```

- Este método procura na blockchain se o url inserido já existe.

```
private void avalButtonMousePressed(java.awt.event.MouseEvent evt) {..}

```

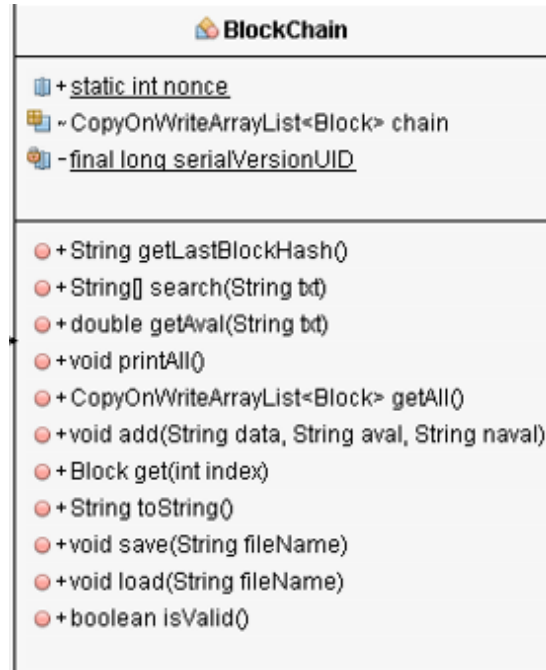
- Este método permite avaliar um URL

```
public boolean isAvalValid(String n){..}
```

- Este método verifica se uma avaliação é válida

5.3.2 Blockchain

- Esta classe é crucial, pois a mesma define e a blockchain que irá ser utilizada ao longo do projeto
- Diagrama detalhado da classe e as ligações a outras classes



- Classes que usa

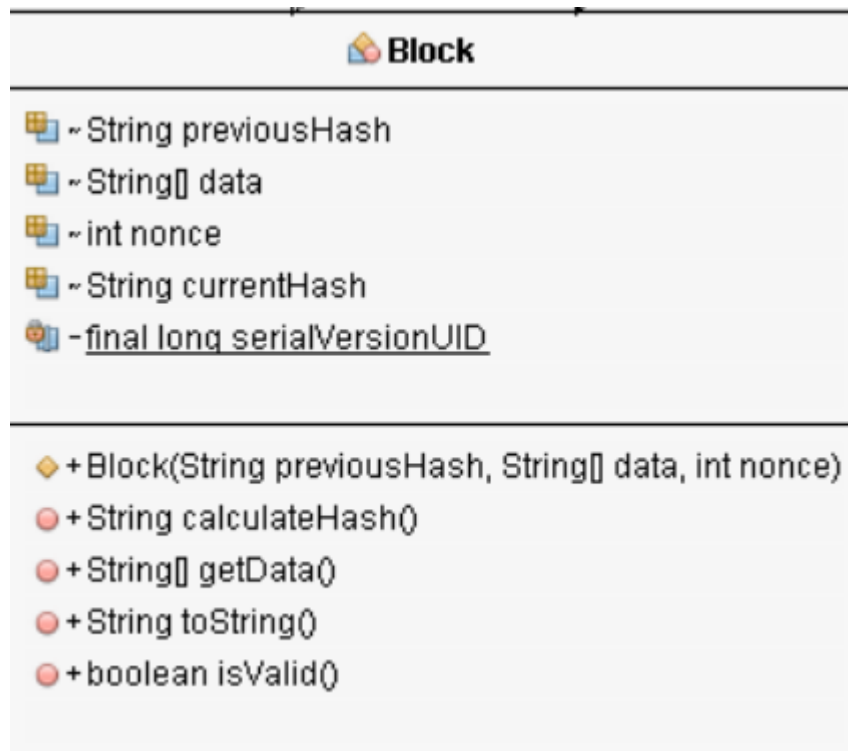
Esta classe utiliza a classe Block e implementa o Serializable.

- Descrição dos atributos
 - `static int nonce` – nonce a ser adicionado no bloco
 - `CopyOnWriteArrayList chain` – representação da blockchain
- Descrição dos métodos
 - Retorno, nome, parâmetros, descrição
 - `getLastBlockHash()` – este método retorna o hash do bloco anterior
 - `search(String txt)` – dado como parâmetro um url, procura-o na blockchain e retorna todos os seus dados caso exista.
 - `getAval(String txt)` – dado como parâmetro um url, procura-o na blockchain e retorna a média de todas as suas avaliações.

- `printAll()` – Este método mostra todos os urls avaliados da blockchain
- `getAll()` – retorna um `CopyOnWriteArrayList` com todos os blocos da blockchain.
- `add(String data, String aval)` – Cria um novo bloco com o url e a avaliação e adiciona-o á blockchain.
- `get(int index)` – Retorna o index de um bloco.
- `save(string filename)` – guarda a blockchain num ficheiro cujo nome é o atributo filename
- `load (string filename)` – carrega a blockchain do ficheiro com o nome designado no atributo filename
- `isValid()`- valida todos os blocos da blockchain

5.3.3 Block

- Esta classe define um dos blocos da blockchain, é crucial para o funcionamento da mesma.
- Diagrama detalhado da classe e as ligações a outras classes

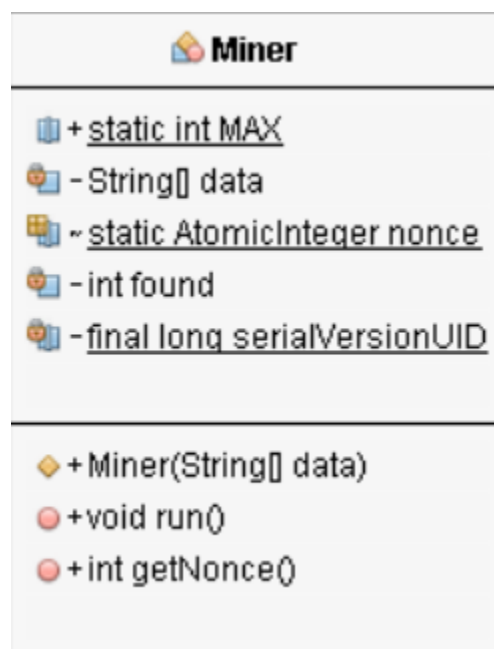


- Utiliza a classe Hash para calcular a hash do bloco
- Descrição dos atributos

- String previousHash – Uma string onde está contida a hash do bloco anterior ao atual.
- String currentHash – Uma string onde está contida a hash do bloco atual.
- Int Nonce – nonce minerado do bloco atual
- String[] data – array com os dados do bloco(URL e avaliação)
- Descrição dos métodos
 - Block(String previousHash, String[] data, int nonce) – Construtor dos blocos
 - String calculateHash() – calcula a hash do bloco atual
 - String[] getData() – retorna o array data do bloco atual
 - String toString() – retorna uma string que representa o bloco
 - Boolean isValid() – verifica se o bloco atual é válido através da sua hash.

5.3.4 Miner

- Esta classe contém todo o código das threads mineradoras, é assim crucial para o bom funcionamento da aplicação.
- Diagrama detalhado da classe

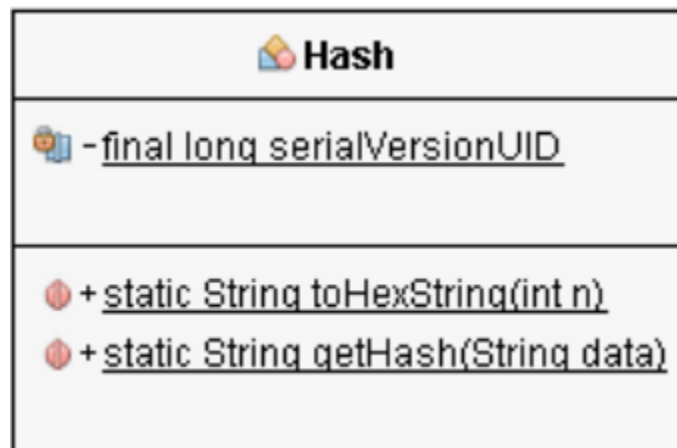


- Descrição dos atributos
 - static int MAX – valor máximo do nonce
 - String[] data – array data do bloco a minerar

- static AtomicInteger nonce – valor a encontrar que satisfaça o bloco atual.
- int found – flag para controlo do nonce.
- Descrição dos métodos
 - Miner(String[] data) – cria um objeto miner com os dados de um bloco.
 - void run() – código a executar quando as threads mineradoras são iniciadas.
 - int getNonce() – retorna o último nonce minerado pelas threads.

5.3.5 Hash

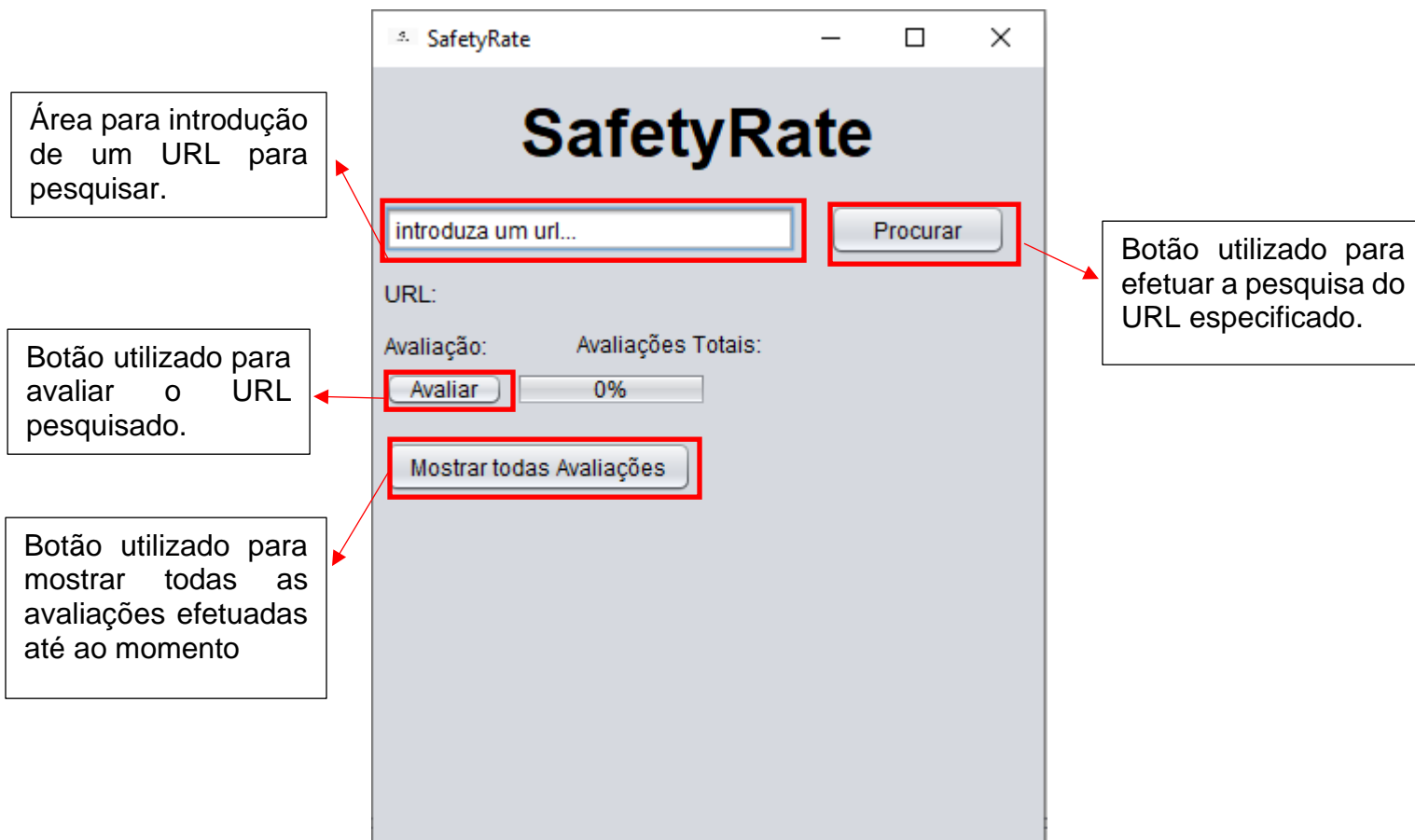
- Esta classe é utilizada para obter a hash de vários atributos e conjunto de atributos da aplicação
- Diagrama detalhado de classe



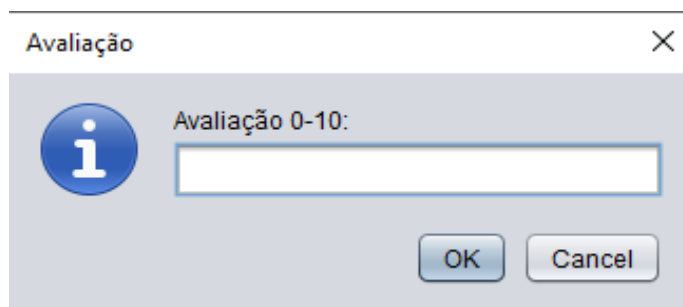
- Descrição dos métodos
 - static String toHexString(int n) – transforma um inteiro em uma string de base 16
 - static String getHash(String data) – obtém a hash de uma dada string

5.4 Aplicação

- Menu Inicial



- Menu de Avaliação

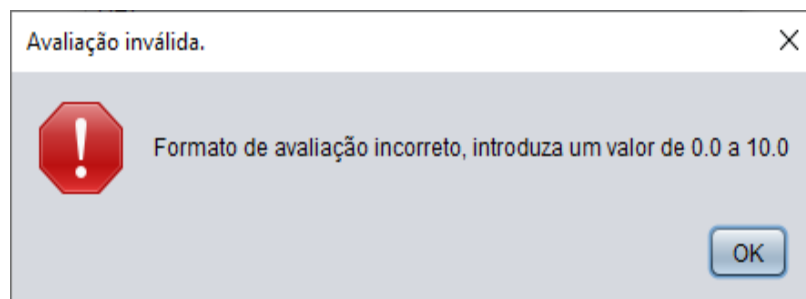


Avaliação

Avaliação 0-10:

OK Cancel

- Avaliação inválida introduzida



Avaliação inválida.

Formato de avaliação incorreto, introduza um valor de 0.0 a 10.0

OK

- Lista de todas as avaliações atuais



SafetyRate

SafetyRate

Avaliacao3 Procurar

URL: Avaliacao3

Avaliação: N/A Avaliações Totais: 0

Avaliar 100%

Mostrar todas Avaliações

Avaliacao1
Avaliacao2
Avaliacao3

URL:
Avaliação:

- Seleção de uma das avaliações da lista

The screenshot shows a web application window titled "SafetyRate". At the top, there is a search bar containing the text "Avaliacao3" and a "Procurar" button. Below the search bar, the text "URL: Avaliacao3" is displayed. Further down, it shows "Avaliação: N/A" and "Avaliações Totais: 0". There are two buttons: "Avaliar" and a progress bar labeled "100%". Below these is a button labeled "Mostrar todas Avaliações". At the bottom, there is a list of evaluations: "Avaliacao1", "Avaliacao2", and "Avaliacao3". The first item, "Avaliacao1", is highlighted. To the right of this list, the text "URL: Avaliacao1" and "Avaliação: 2" is shown.

○

6 Trabalho 3 – Computação distribuída

6.1 *Arquitetura da aplicação*

- Diagrama de classes
- Como se encontra estruturado o código da aplicação:
 - Objetos e classes do sistema.
 - Relações entre as classes dos objetos
- Descrição geral das bibliotecas
 - As minhas bibliotecas
 - recursos externos
 - justificação para a sua utilização
 - Inserir na bibliografia

6.2 *Biblioteca 1*

- Descrição geral da biblioteca

6.2.1 **Classe A**

- Justificação da sua existência e a sua função no projeto.
- Diagrama detalhado da classe e as ligações a outras classes
 - Herança ou composição
 - Interfaces implementadas
 - Classes que usa
- Descrição dos atributos
 - Tipo, nome, descrição
- Descrição dos métodos
 - Retorno, nome, parâmetros, descrição

6.2.2 **Classe B**

....

6.3 *Aplicação*

- Interfaces com o utilizador
- Manual do utilizador

7 Limitações e Desenvolvimentos Futuros

- Quais as limitações que a aplicação tem:
 - Limitação no 1º Trabalho:
 - O 1º trabalho possui a limitação de não utilizar várias threads no seu processamento, o que limita a velocidade da aplicação, no 2º trabalho já foi implementado MultiThreading o que aumentou significativamente a velocidade da aplicação.
- O que ficou por fazer.
 - Melhoramentos visuais.
- Desenvolvimento futuro
 - Novas funcionalidades
 - Justificação da funcionalidade
 - Mais-valia para o projeto

8 Conclusão

- Apreciação geral do trabalho desde a sua projeção até ao resultado.
- Nível de concretização do projeto
 - Justificação do nível.
- Apreciação do trabalho de projeto.
 - O que aprendeu a fazer por si autonomamente.
 - Situações em que precisou de ajuda e a sua resolução.
 - agradeça aqui a quem o ajudou.
- Conclusão final

9 Referencias

NOTA:

TODAS AREFERENCIAS DEVEM SER CITADAS NOS COMPITULOS ANTERIORES.

9.1 Livros

- [REF] Apelido, Nome Próprio , Título, local: editora, edição, páginas, , data de publicação

9.2 Documentos

- [REF] Apelido, Nome Próprio, "Título do Artigo", Nome da Publicação, local: editora, páginas, data de publicação
- [D1] Nakamoto, Satoshi. "*Bitcoin: A peer-to-peer electronic cash system.*" *Decentralized Business Review* (2008): 21260.

9.3 Referência web

- [REF] Título, endereço, autores, data de consulta
- [W1] O que é e como funciona o blockchain: além das criptomoedas, URL: <https://distrito.me/blog/blockchain-o-que-e-como-funciona/> , Amarílis Ferreira, Consultado em 20/09/2022
- [W2] Descubra as 7 principais aplicações do blockchain e seu funcionamento URL: <https://www.voitto.com.br/blog/artigo/aplicacoes-do-blockchain> João Henrique, Consultado em 28/09/2022
- [W3] Blockchain Applications: 62 Killer Ideas For You URL: <https://www.connectbit.com/blockchain-applications/> Cheefo, consultado em 12/10/2022
- [W4] Blockchain URL: <https://pt.wikipedia.org/wiki/Blockchain> Wikipédia, consultado em 29/09/2022
- [W5] Blockchain URL: <https://en.wikipedia.org/wiki/Blockchain> Wikipédia, consultado em 29/09/2022
- [W6] Blockchain 2.0 URL: <https://coinmarketcap.com/alexandria/glossary/blockchain-2-0> Alexandria, consultado em 29/09/2022
- [W7] How Many Blocks Are in a Blockchain? URL: <https://originstamp.com/blog/how-many-blocks-are-in-a-blockchain/#:~:text=In%20blockchain%20technology%2C%20a%20block,each%20other%20with%20specific%20characteristics.> originstamp, consultado em 29/09/2022

- **[W8]** O que é um Algoritmo de Consenso?
URL: <https://academy.binance.com/pt/articles/what-is-a-blockchain-consensus-algorithm#other-consensus-algorithms>
Binance Academy, consultado em 01/10/2022

- **[W9]** Os Prós e os Contras da Tecnologia Blockchain
URL: <https://kriptomat.io/pt/blockchain/os-pros-e-os-contras-da-tecnologia-blockchain/>
Kriptomat, consultado em 29/09/2022