



## **Plataforma de voto eletrónico com tecnologia Blockchain**

### **Laboratório em Engenharia Informática**

Mestrado Integrado em Engenharia Informática  
Universidade do Minho

2º Semestre  
2017-2018  
Grupo 48

Afonso Fontes



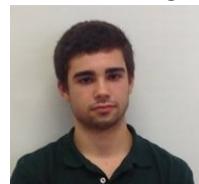
pg35389

Bruno Dantas



a74207

Daniel Rodrigues



a75655

27 de Junho de 2018

# Conteúdo

<b>1</b>	<b>Introdução</b>	<b>3</b>
1.1	Descrição do projeto . . . . .	3
1.2	Propósito do projeto . . . . .	3
1.2.1	Motivações . . . . .	3
1.2.2	Objetivos . . . . .	3
1.3	Âmbito do trabalho . . . . .	3
1.3.1	Contextualização . . . . .	3
1.3.2	Divisão de etapas . . . . .	3
1.4	Definições e convenções . . . . .	4
1.4.1	Termos chave . . . . .	4
1.4.2	HyperLedger Fabric . . . . .	4
<b>2</b>	<b>Propriedades de segurança</b>	<b>5</b>
2.1	Autenticação . . . . .	5
2.2	Unicidade do voto . . . . .	5
2.3	Anonimato . . . . .	5
2.4	Integridade . . . . .	6
2.5	Auditabilidade . . . . .	6
2.6	Mobilidade . . . . .	6
2.7	Confidencialidade do voto . . . . .	6
<b>3</b>	<b>Implementação</b>	<b>7</b>
3.1	Arquitetura . . . . .	7
3.1.1	Componentes . . . . .	7
3.1.2	Diagrama geral . . . . .	8
3.2	Blockchain . . . . .	9
3.2.1	Docker . . . . .	9
3.2.2	Chaincode . . . . .	9
3.3	SDK . . . . .	9
3.4	Web . . . . .	10
3.4.1	Servidor . . . . .	10
3.4.2	Interface . . . . .	10
<b>4</b>	<b>Referências</b>	<b>12</b>

# Listas de Figuras

3.1	Simples arquitetura inicial . . . . .	8
3.2	Página de login . . . . .	11
3.3	Selecção do voto . . . . .	11

# Capítulo 1

## Introdução

### 1.1 Descrição do projeto

O projeto consiste no desenvolvimento um protótipo de um sistema de voto eletrónico com auxilio das tecnologias de *Blockchain*.

### 1.2 Propósito do projeto

#### 1.2.1 Motivações

O principal motivo que nos incentivou a desenvolver este projeto foi a exploração da tecnologia *Blockchain* e das suas possíveis aplicações. Dadas as propriedades desta tecnologia, como a imutabilidade das transações inseridas, transparência, acessibilidade e descentralização, a sua aplicação num sistema de voto eletrónico faz todo o sentido. Especialmente devido à imutabilidade e acessibilidade dos dados, uma vez que é crucial que o sistema seja consistente e que permita a verificação dos votos inseridos. Por outro lado, já existem vários projetos desenvolvidos nesta tecnologia e são abundantes os exemplos de sistemas de voto eletrónico que usam a tecnologia *Blockchain*. No entanto, o *Hyperledger* apresenta diferenças significativas perante outras tecnologias *Blockchain*, possibilitando uma melhor estruturação das entidades envolvidas no processo de voto que não encontramos noutras projetos publicados.

#### 1.2.2 Objetivos

O principal objetivo é o desenvolvimento de um protótipo de uma plataforma de voto eletrónico que garanta os requisitos mínimos de segurança relacionados com a sua arquitetura. O sistema deve ser desenvolvido de modo a aproveitar as propriedades da tecnologia *Blockchain*.

### 1.3 Âmbito do trabalho

#### 1.3.1 Contextualização

Com o avanço da tecnologia e o desenvolvimento da Internet, a substituição do sistema de voto tradicional por sistemas de votos eletrónicos é uma questão que já existe a alguns anos e que já foi adotada com sucesso em alguns países. As vantagens que motivam esta transição são principalmente a redução dos custos associados, a acessibilidade que o sistema pode disponibilizar, uma maior rapidez na contagem dos votos e a facilidade com que é possível verificar a correta contagem dos mesmos. No entanto, um sistema de voto eletrónico requer técnicas de segurança muito rigorosas, o que torna o processo de desenvolvimento e de verificação dessas mesma técnicas muito complexo. A tecnologia *Blockchain* vem deste modo colmatar alguns dos requisitos de segurança, tornando-se a base do sistema, uma vez que esta tecnologia garante por si só a imutabilidade dos dados inseridos, auditabilidade e não é necessário confiar nas entidades envolvidas no processo de inserção dos dados.

#### 1.3.2 Divisão de etapas

O projeto foi dividido em três etapas, a primeira etapa consiste na pesquisa sobre as tecnologias *Blockchain* disponíveis e as suas propriedades. A segunda etapa consiste no desenvolvimento da

arquitetura do sistema, de modo a garantir os requisitos mínimos de segurança. A terceira etapa consiste no desenvolvimento de um protótipo de voto eletrónico.

## 1.4 Definições e convenções

### 1.4.1 Termos chave

*Blockchain, Hyperledger Fabric, Chaincode, Web, Docker, Golang, Node.js*

### 1.4.2 HyperLedger Fabric

O Hyperledger foi fundado em 2015 pela Linux para promover as tecnologias *Blockchain* entre empresas. O Hyperledger Fabric é um dos projetos do Hyperledger que possui características distintas das mais comuns tecnologias de *Blockchain*. Este sistema é *permissioned*, em vez de um sistema aberto sem necessidade de qualquer autenticação, este sistema só é acessível a um determinado conjunto de utilizadores. A característica anterior permite adotar um sistema de inserção das transações sem necessitar de protocolos semelhantes ao *Proof of Work*, em vez disso as transações são inseridas quando as entidades presentes no sistema chegarem a um **consenso**, ou seja, se a resposta das entidades a uma determinada transação for igual, ou pelo menos uma determinada percentagem destas. O sistema permite também a criação de **canais** que isola as transações efetuadas entre os elementos presentes nesse canal do resto dos elementos do sistema, assim sendo, apenas os elementos desse canal podem ver as transações que ocorre neste, no entanto cada entidade do sistema pode estar associada a vários canais. O Hyperledger fabric usa *Smart Contract* para interagir com a *Blockchain*.

## Capítulo 2

# Propriedades de segurança

Dada a importância na garantia de certas propriedades de segurança num sistema de votos eletrónico e visto que o projeto se enquadra no perfil de *Criptografia e Segurança da Informação*, é dedicado um capítulo para abordar individualmente cada uma dessas propriedades. Para cada uma delas começamos pela definição, seguindo-se a explicação dos motivos de garantia da propriedade e finalmente como esta é garantida no nosso sistema.

### 2.1 Autenticação

A primeira propriedade que abordamos é a autenticidade, que consiste em verificar a validade de pelo menos uma forma de identificação. De forma mais genérica, esta propriedade visa confirmar qualquer forma de identificação. No nosso caso, o sistema deve verificar e controlar o acesso dos utilizadores de forma a impedir que certas entidades possam exercer o direito de voto. Dito de outro modo, só os votantes relacionados/escolhidos para a eleição podem votar no sistema.

A autenticação do votante é garantida pela verificação das suas credenciais quando este solicita o acesso ao sistema de votos. Para o efeito, o serviço web mantém uma lista de utilizadores que permite identificar quais possuem o direito de voto nas eleições disponíveis, assim como identificar se esse direito foi exercido.

### 2.2 Unicidade do voto

Esta propriedade prende-se com a capacidade de o sistema garantir que não é possível efetuar duas ou mais ações (idênticas ou não) quando apenas uma seria esperada. Os exemplos mais clássicos estão relacionados com transações bancárias, em que se deve prevenir que a unicidade de uma transação.

Obviamente, cada votante tem direito a efetuar apenas um voto em cada eleição (apesar de em alguns sistemas os votos possuírem diferentes pesos em função das entidades que votam). Assim que este é submetido, o votante correspondente não pode voltar votar. Esta propriedade é verificada após o processo de autenticação e estão relacionadas. Para isso, o serviço web possui os registos de voto associados a cada utilizador, indicando se o votante já exerceu o seu voto. Mais especificamente, mantém um estado de entre três possíveis que permite: i) identificar se o votante já votou; ii) identificar se o processo de votação está a decorrer (durante o processamento do sistema) ; iii) se ainda não votou.

### 2.3 Anonimato

O anonimato consiste em, para determinadas ações e/ou informações, esconder a entidade associada a estas. O interesse em garantir esta propriedade num sistema de voto eletrónico relaciona-se com o facto de nenhuma entidade (para além do votante) conseguir associar um voto efetuado ao votante. Além disso, ninguém deve conseguir saber se o votante exerceu o seu direito de voto.

O sistema foi desenhado de modo a não permitir associar os votos às entidades que o efetuaram.

## 2.4 Integridade

Uma das propriedades de segurança fundamental em praticamente todo o tipo de aplicações é a integridade. Toda a informação gerada e/ou transmitida deve permanecer inalterada durante o seu tempo de vida, seja por ações maliciosas (ataques ativos) ou erros de processamento e/ou transmissão. Neste caso, os votos não podem ser alterados ou perdidos no processo do sistema. Note-se que efetuar apenas a cifragem colocaria a possibilidade de existirem ataques de bit escolhido, pelo que também são necessárias assinaturas digitais e *hashes*. Este mecanismos são utilizados desde a criação do voto até à sua inserção na *Blockchain*, sendo verificadas pelas várias entidades presentes no sistema.

## 2.5 Auditabilidade

O processo de auditabilidade permite verificar a contagem e verificar a validade dos votos inseridos no sistema. Facilmente se entende que este requisito é essencial para que o sistema seja considerado de confiança. É possível verificar os votos e se estes foram contados corretamente. Este requisito é garantido pela *Blockchain* uma vez que esta se torna pública no fim da votação e tendo em conta as suas propriedades de acessibilidade e de imutabilidade dos dados.

## 2.6 Mobilidade

A mobilidade do sistema consiste na facilidade do acesso ao sistema, como o acesso é efetuado por intermédio da Internet o sistema permite uma grande mobilidade. Este requisito é um dos aspectos fundamentais dos sistemas de voto eletrónico e uma das vantagens que torna estes sistemas mais atraentes que o sistema de voto tradicional.

## 2.7 Confidencialidade do voto

Juntamente com a integridade, esta propriedade costuma ser das mais desejáveis em sistemas de informação. Dito de forma muito simples, esconde a informação a transmitir/armazenar através da cifragem (ou qualquer outro método de ofuscação/codificação) da mesma. Claro que, num sistema de voto eletrónico, o conteúdo do voto deve ser mantido em segredo até que a votação termine. Esta propriedade é garantida porque o voto escolhido pelo utilizador é cifrado com a chave pública da eleição (a chave privada para decifragem está dividida por diferentes entidades através da partilha de segredos de *Shamir*). Nestas condições, seria possível descobrir os votos efetuados uma vez que o espaço de mensagens cifradas estaria limitado ao número de diferentes opções de voto. Por este motivo, é introduzido também um número aleatório para inclusão no processo de cifragem, evitando o caso descrito anteriormente.

# Capítulo 3

# Implementação

## 3.1 Arquitetura

### 3.1.1 Componentes

#### Cliente

O cliente é o utilizador que se espera que efetue o voto através da interface disponibilizada pelo serviço web. Cada utilizador é identificado apenas pelas suas credenciais. Após a escolha do voto, gera um número aleatório e realiza a cifragem de ambos com a chave pública recebida pelo servidor e a assina o resultado com a sua chave privada. Finalmente, envia estes dados juntamente com o *token* para o *SDK*.

#### Web

Este serviço funciona como um serviço web tradicional. Oferece uma interface intuitiva aos utilizadores, disponibilizando a lista de eleições disponíveis para cada um. Simultaneamente, trata dos mecanismos de autenticação e é a entidade que fornece a chave pública associada a cada eleição, usada para cifrar o voto escolhido pelo utilizador.

#### SDK

Esta aplicação assegura a comunicação com a *Blockchain* e é esta entidade que controla maioritariamente o sistema. Note-se que é a única das quatro que comunica simultaneamente com todas as outras. Além de ajudar a garantir a maioria das propriedades de segurança, assegura que os votos cifrados são inseridos no *ledger* de todos os *peers*.

#### Blockchain

A *Blockchain*, no caso de um sistema deste género, é uma rede bastante reduzida. O número de entidades não se afasta muito do número de listas candidatas (cada uma é um *peer* na rede), uma vez que para além destas apenas existem os *Orderers* (devem ser considerados dois para não existir um único ponto de falha) e a comissão eleitoral (*peer*).

### 3.1.2 Diagrama geral

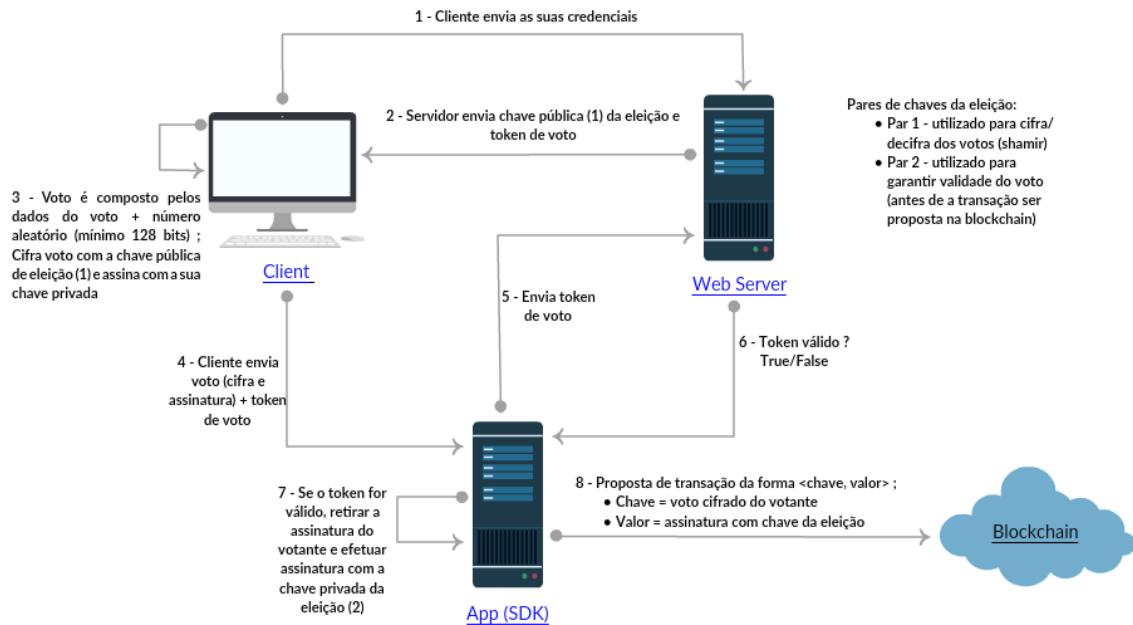


Figura 3.1: Simples arquitetura inicial

**1** – As credenciais do cliente são enviadas em conjunto com a sua chave pública, por um canal seguro, ao servidor *Web*. O servidor verifica a validade das credenciais do cliente, assim como a sua chave pública. Se os parâmetros passarem no processo de validação, o servidor atribui um *token* ao cliente. O *token* de cada cliente deve ser único e gerado de forma aleatória, e é usado de modo a verificar a autenticidade do votante de forma indireta (serve para autenticar os votos anônimos recebidos pela *App* de modo a garantir o anonimato do voto).

**2** – Depois do cliente se autenticar perante o servidor *Web*, o servidor envia por um canal seguro a chave pública de eleição e o *token* gerado. A chave pública de eleição é usada no processo de cifragem/decifragem do voto gerado pelo votante autenticado. Note-se que as chaves da eleição são geradas de acordo o algoritmo de *Shamir Secret Sharing* (divisão de segredo de Shamir).

**3** – Neste ponto o votante escolhe a opção de voto que pretende submeter, seguindo-se o processo de cifra do voto com a chave pública de eleição, anteriormente referida. Neste processo é necessário adicionar um número aleatório (com pelo menos 128bits) aos dados do voto, antes deste ser cifrado, de modo a aumentar a entropia dos dados. O voto é assinado com a chave privada do votante, para posteriores verificações de autenticidade.

**4** – O cliente envia por um canal seguro o voto cifrado e assinado em conjunto com o *token* que lhe foi atribuído pelo servidor *Web* no passo 1. É de salientar que os únicos dados, presentes nesta mensagem, que podem revelar a identidade do votante no sistema são o seu *token* e a sua assinatura. No entanto, para associar o voto ao votante são necessárias informações adicionais, como a sua chave pública e uma associação direta entre o votante e o *token*. Deste modo apenas o serviço *Web* pode determinar a quem pertence determinado voto.

**5** – Depois de receber o voto e o *token* do votante, a *App* envia o *token*, através de um canal seguro, para o servidor *Web* de modo a verificar a autenticidade do voto e a sua unicidade. A autenticidade é garantida pela associação do *token* com os dados do votante e a unicidade é garantida por um contador associado ao votante. Neste passo o anonimato do voto não é comprometido uma vez que o servidor *Web* não possui dados sobre o voto.

**6** – O resultado da verificação do ponto anterior é enviado, por um canal seguro, para a *App*.

**7** – Se os resultados da verificação do passo 5 forem positivos, a *App* procede à substituição da assinatura do voto, à assinatura do votante, pela assinatura com a chave privada da *App*. A assinatura da *App* indica que o voto foi aceite pelo sistema.

**8** – Neste passo é criada uma transação, constituída pelo par (Chave - voto cifrado, valor - assinatura da *App*), que é enviada para o sistema que gera a *Blokchain*, esta transação ainda deve ser verificada pelos *peers* de modo a ser inserida corretamente na *Blockchain*, os *peers* vão verificar a assinatura presente na transação para determinar se o voto é válido ou não.

**Nota** Todas as comunicações entre os diversos componentes são feitas sobre TLS.

## 3.2 Blockchain

### 3.2.1 Docker

De modo a facilitar a simulação dos vários *peers* na rede do sistema, foi utilizado o programa *Docker* que permite criar virtualizações (*containers*) na mesma máquina. Por outras palavras, cada *container* criado pelo *Docker* executa de forma independente. Além disso, oferece mecanismos para interagir com os *containers* (por exemplo, para verificar *logs* ou para inspecionar os conteúdos dessa máquina virtual).

Para criar os diferentes *containers* (*peers*, *orderers*, *channels*, etc), é executada uma *script* que consome um ficheiro de configuração **.yaml**. Dentro desse ficheiro são definidas as características de cada *container*. O *Hyperledger Fabric* fornece vários exemplos desses mesmos ficheiros para diferentes casos de utilização, pelo que não se construiu um de raiz. Os detalhes desse ficheiro de configuração não são aqui abordados. As imagens do *Docker* que são executadas pelos *containers* também são disponibilizadas na página oficial do *Hyperledger Fabric*.

### 3.2.2 Chaincode

O *chaincode* da aplicação (normalmente referido como *smart contract* noutras redes *Blockchain*) foi desenvolvido em linguagem **Golang**, ou simplesmente **Go**. Possuí funções muito simples para inserir o voto cifrado no *ledger* (após a correta verificação da assinatura efetuada pelo *SDK*) ou fazer a consulta de todos os votos do *ledger*. Esta última função só é executada pela aplicação *SDK* no final da eleição, ou seja, quando for atingido o limite de tempo em que é possível submeter votos na rede.

Este ficheiro é instalado em cada *peer* e qualquer alteração no seu conteúdo resulta em diferentes resultados de execução nos diferentes *peers*. Uma vez que foi definido que o *endorsement* tem que ser efetuado por todas as entidades (lista candidatas e comissão eleitoral), essas alterações são detetadas e não é possível realizar qualquer operação sem o consenso das restantes partes.

É importante manter apenas as funcionalidades essenciais neste ficheiro, porque o mesmo código é executado em diferentes máquinas, em diferentes alturas e em diferentes condições. Se toda a lógica do programa for colocada nesta componente, os erros na interação com a rede aumentam. Primeiro porque a probabilidade de obter diferentes resultados aumenta, e segundo porque o tempo das execuções dificultaria o processo de sincronização dos vários *ledgers* distribuídos. As invocações às funções disponibilizadas pelo *chaincode* só são efetuadas a partir da aplicação *SDK*. Note-se que cabe ao *SDK*, após receber os resultados de execução do *chaincode*, verificar se todos são iguais.

## 3.3 SDK

Esta componente está associada a praticamente todas as ações efetuadas no sistema. Começando na verificação do *token* de voto - prova de que o utilizador pode efetuar a votação - esta aplicação controla o estado de execução desde essa fase até à inserção final do voto na *Blockchain*. Assim como o serviço web, foi desenvolvido através da linguagem **Node.js**.

Após a receção do *token* pelo cliente, o *SDK* reencaminha-o para o servidor web para verificar se o utilizador tem direito a votar na eleição. Admitindo que o voto é legal, a aplicação prossegue com todos os mecanismos de segurança (descritos na secção 3.1) e tenta inserir o voto na *Blockchain*. A validade na verificação das assinaturas (nomeadamente a que controla a inserção no *ledger*) não garante que a submissão do voto seja bem sucedida, uma vez que se trata de um sistema distribuído. Para garantir isso, é preciso manter a informação do voto nesta componente até que uma tentativa de submissão tenha sucesso. Assim que isto aconteça, o *SDK* comunica com o servidor web para que este altere o estado associado ao utilizador que realizou o voto (durante todo o processo descrito o servidor web altera temporariamente o seu estado para prevenir ataques de *double spending* e garantir a unicidade dos votos).

## 3.4 Web

### 3.4.1 Servidor

A aplicação contém um serviço web (denominado Web Server na figura 3.1), que não só assegura a autenticação do votante como desempenha um papel fundamental na segurança do sistema, ao assegurar a unicidade e anonimato do voto, através da emissão e controlo de tokens de voto.

O desenvolvimento deste componente foi feito com recurso a **Node.js** e o desenvolvimento das duas APIs (uma para autenticação do votante e uma para verificação de tokens de voto), foi feita com recurso ao popular módulo **express**.

Este componente verifica as credenciais do utilizador, fornece-lhe um boletim e token de voto e, numa fase posterior, comunica com a App (SDK) para verificação do token de voto. Um boletim de voto foi implementado simplesmente como uma lista de candidatos que são apresentados ao votante, dos quais deve escolher um (e apenas um), no entanto, o sistema poderia ser adaptado a eleições com diferentes formatos. Os tokens de voto tratam-se de números inteiros de 128 bits, gerados aleatoriamente e com validade de 2 minutos (ou seja, desde a sua emissão até à inserção do voto na blockchain, não podem passar mais de 2 minutos, ou o voto é recusado).

### 3.4.2 Interface

A interface com o utilizador (votante) é inteiramente Web e relativamente minimalista. Foi assumido que um votante já tem conhecimento prévio das suas credenciais, pelo que não é necessário ou desejável ter uma funcionalidade de registo. Assim sendo, a interface permite fazer login e votar.

O desenvolvimento foi feito com as ferramentas tradicionais de desenvolvimento web, nomeadamente, **HTML**, **CSS** e **Javascript**, sendo que o único módulo externo utilizado foi o **jQuery**.

A aplicação web é responsável por comunicar com os 2 serviços disponíveis. Numa primeira fase, autentica-se no Web Server e recebe um token de voto e um boletim, se a autenticação ocorrer com sucesso. Numa segunda fase, gera o número aleatório (128 bits) para concatenar com o voto e envia o seu voto (com o número aleatório) e token para a App(SDK) que é responsável por inserir o voto na blockchain.

Exemplos do funcionamento da aplicação podem ser vistos nas figuras 3.2 e 3.3

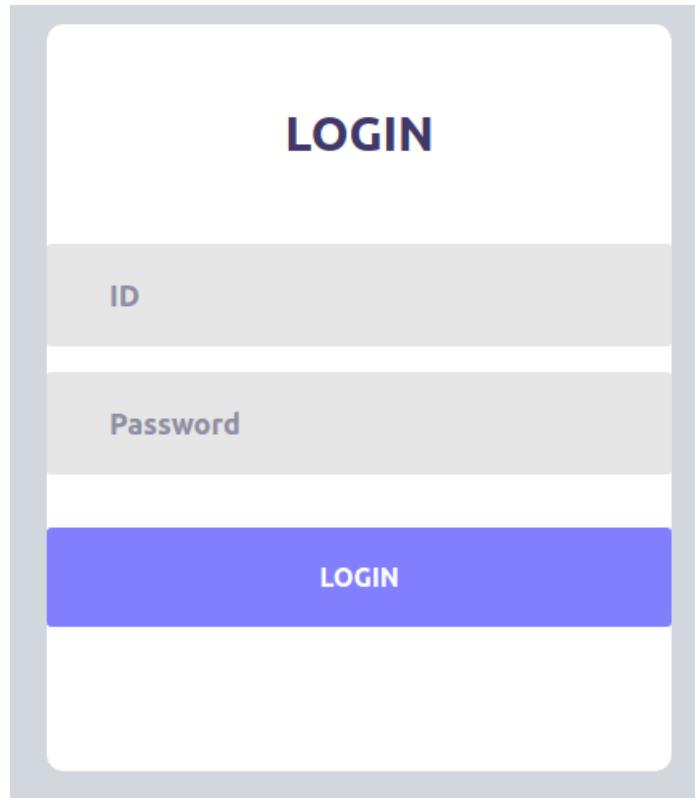


Figura 3.2: Página de login

A screenshot of a mobile application's voting screen. The title "Opções" is at the top in bold black capital letters. Below it are two rows, each consisting of a label and a radio button. The first row has the label "Lista A" and the second row has the label "Lista B", both in black capital letters. At the bottom is a large blue button with the word "SUBMETER" in white capital letters.

Figura 3.3: Selecção do voto

# **Capítulo 4**

# **Referências**

[1] - Electronic Voting 2006 2nd International Workshop Co-organized by Council of Europe.