UNIVERSIDADE FEDERAL DE SANTA MARIA
CENTRO DE TECNOLOGIA
ENGENHARIA DE COMPUTAÇÃO

Pedro Henrique Korb

COLLABRATE: UM PROTÓTIPO DE APLICATIVO DESCENTRALIZADO PARA AVALIAÇÃO DE DESEMPENHO USANDO A TECNOLOGIA BLOCKCHAIN

Santa Maria, RS
2018

Pedro Henrique Korb

COLLABRATE: UM PROTÓTIPO DE APLICATIVO DESCENTRALIZADO PARA AVALIAÇÃO DE DESEMPENHO USANDO A TECNOLOGIA BLOCKCHAIN

Trabalho de Graduação apresentado ao curso de Engenharia de Computação da Universidade Federal de Santa Maria (UFSM, RS), como requisito parcial para a obtenção do grau de **Bacharel em Engenharia de Computação.**

Orientador: Dr. João Carlos Damasceno Lima

Santa Maria, RS
2018

Pedro Henrique Korb

COLLABRATE: UM PROTÓTIPO DE APLICATIVO DESCENTRALIZADO PARA AVALIAÇÃO DE DESEMPENHO USANDO A TECNOLOGIA BLOCKCHAIN

Trabalho de Graduação apresentado ao curso de Engenharia de Computação da Universidade Federal de Santa Maria (UFSM, RS), como requisito parcial para a obtenção do grau de **Bacharel em Engenharia de Computação.**

Aprovado em 5 de dezembro de 2018:

_____

João Carlos Damasceno Lima, Dr. (DLSC - UFSM)
(Presidente/Orientador)

_____

José Eduardo Baggio, Dr. (DELC - UFSM)

_____

Fernando Pires Barbosa, M.e (PROPLAN - UFSM)

Santa Maria, RS
2018

AGRADECIMENTOS

Agradeço primeiramente aos meus pais, Ildo R. Korb e Roseli M. Korb, por terem me dado todo o amor, apoio (financeiro e emocional) e por sempre acreditarem no meu potencial. Sem eles, esse trabalho não estaria sendo lido.

À minha namorada, Karen Tatsch, por fazer parte dos últimos dois anos e meio da minha vida.

Aos membros da banca, Dr. José Eduardo Baggio e M.e Fernando Pires Barbosa, que disponibilizaram seu precioso tempo para ler, avaliar e dar sugestões para esse trabalho.

Ao amigo e engenheiro de computação Matheus M. Korb, que apesar de não sermos parentes, temos o mesmo sobrenome. Quase como um coorientador, me ajudou dando dicas preciosas na produção desse trabalho acadêmico. Que continuamos tendo essa bonita relação de amizade.

Aos meus amigos, que estão juntos nas horas boas e ruins, apoiando quando necessário.

À Lidiane Bertê, diretora da Nekto, empresa onde atualmente sou estagiário. Por me dar a oportunidade de trabalhar e ter contato com a área de gestão de pessoas.

À Universidade Federal de Santa Maria (UFSM), ao curso de Engenharia de Computação e a todos os professores que fizeram parte da longa caminhada até a conclusão do curso. Vocês foram muito importantes para minha formação pessoal e profissional.

Ao meu orientador João Carlos Damasceno Lima, por se mostrar prestativo e amigo, me orientando ao longo desse trabalho.

À Universidade Federal do Rio Grande (FURG) por também fazer parte da minha formação, como porta de entrada para a vida acadêmica.

A todos que de alguma forma contribuíram e não estão diretamente citados.

"Os computadores são incrivelmente rápidos, precisos e burros; os homens são incrivelmente lentos, imprecisos e brilhantes; juntos, seus poderes ultrapassam os limites da imaginação." – Albert Einstein.

RESUMO

Trabalho de Graduação
Engenharia de Computação
Universidade Federal de Santa Maria

COLLABRATE: UM PROTÓTIPO DE APLICATIVO DESCENTRALIZADO PARA AVALIAÇÃO DE DESEMPENHO USANDO A TECNOLOGIA BLOCKCHAIN

AUTOR: Pedro Henrique Korb
ORIENTADOR: João Carlos Damasceno Lima

Este trabalho apresenta o desenvolvimento de um protótipo de aplicativo móvel para o sistema operacional Android com a função de realizar avaliações do gestor sobre seus colaboradores, integrando os resultados das avaliações ao Blockchain. A avaliação de desempenho é uma ferramenta importante para o gestor acompanhar o desenvolvimento pessoal do seu quadro de funcionários. Essas avaliações são usadas para identificar os pontos fortes que devem ser enaltecidos e os pontos fracos que devem ser melhor desenvolvidos. Bonificações podem ser oferecidas aos colaboradores que se destacam positivamente.

Ao integrar os resultados das avaliações no Blockchain, adiciona-se confiabilidade e transparência ao processo avaliativo. Também, caso algum colaborador se desmembre do quadro de pessoal, será possível que o mesmo utilize os dados públicos para substituir as “cartas de recomendação” tradicionalmente oferecidas pelo gestor quando seu antigo colaborador está procurando outra empresa para contribuir com seus talentos.

No aplicativo, que é direcionado para o gestor, é possível cadastrar, alterar e excluir colaboradores do seu quadro pessoal. É possível criar um formulário de perguntas personalizado, com suas respostas obedecendo a escala Likert. Também podem ser vistos os resultados individuais das avaliações mensais de cada colaborador por meio de gráficos.

Palavras-chave: Blockchain. Avaliação de desempenho. Android.

ABSTRACT

COLLABRATE: A DECENTRALIZED APPLICATION PROTOTYPE FOR PERFORMANCE ASSESSMENT USING BLOCKCHAIN TECHNOLOGY

AUTHOR: Pedro Henrique Korb
ADVISOR: João Carlos Damasceno Lima

This work presents the development of a mobile application prototype for the Android operating system with the function of evaluating the manager about his collaborators, integrating the evaluation results to Blockchain. Performance assessment is an important tool for the manager to track the personal development of his or her staff. These assessments are used to identify the strengths that should be exalted and the weaknesses that should be best developed. Bonuses can be offered to employees who stand out positively.

By integrating the results of the evaluations into Blockchain, reliability and transparency are added to the evaluation process. Also, if a staff member disagrees with the staff, he or she may use the public data to replace the "letters of recommendation" traditionally offered by the manager when his former employee is looking for another company to contribute his or her talents.

In the application, which is directed to the manager, you can register, change and delete employees from your staff. You can create a custom question form with your answers following the Likert scale. Individual results of the monthly evaluations of each employee can also be viewed by means of graphs.

Keywords: Blockchain. Performance assessment. Android.

LISTA DE FIGURAS

LISTA DE SIGLAS E ABREVIATURAS

ABI	Application Binary Interface
API	Application Programming Interface
DApp	Distributed Application
DLT	Distributed Ledger Technology
EOA	Externally Owned Account
EVM	Ethereum Virtuam Machine
HTTP	Hypertext Transfer Protocol
IPC	Inter-Process Communication
JSON	JavaScript Object Notation
JVM	Java Virtual Machine
LLL	Low-Level Lisp-Like Language
RPC	Remote Procedure Call
SDK	Software Development Kit
UTXO	Unspent Transaction Output
XML	Extensible Markup Language

SUMÁRIO

1. INTRODUÇÃO

Avaliação de desempenho é um processo adotado por empresas para mensurar o desempenho e comportamento de seus colaboradores a partir de critérios comuns. Com essas avaliações é possível identificar os pontos fortes e bonificar o colaborador por isso, bem como capacitá-lo quando identificados pontos fracos. Para o gestor, a avaliação de desempenho abre a possibilidade de formar uma equipe mais qualificada e eficiente para suprir as necessidades da empresa. Portanto, a avaliação de desempenho pode ser uma ferramenta importante na hora de identificar problemas dentro da empresa e também valorizar seu capital humano (BOHLANDER; SNELL, 2016).

No entanto, a maioria das avaliações aplicadas são feitas de forma manual ou utilizando softwares que não são especializados para essa finalidade. Além disso, o surgimento de conflitos por colaboradores mal avaliados e a dependência da honestidade e da lealdade dos gestores são problemas em potencial para as empresas (DESSLER, 2012).

Muitas vezes os indicadores são confidenciais e o colaborador só toma conhecimento dos resultados quando é promovido, pois o processo é diretamente vinculado com a remuneração. A avaliação anual dificulta a mensuração da capacidade do colaborador, sendo um período inadequado que leva ao extremo as capacidades da mente humana em guardar recordações dos seus colaboradores (GRAMIGNA, 2012).

Uma forma de contornar esses problemas consiste na criação de um software para que o gestor possa realizar a avaliação de colaboradores mensalmente, permitindo os colaboradores verificarem as notas que lhes foram atribuídas. Para isso, esse modelo proposto se encaixa perfeitamente no conceito que o Blockchain propõe.

O Blockchain é uma rede descentralizada e pública, onde cada dispositivo presente nela tem uma cópia de todos os dados presentes nela. Os dados são armazenados em um bloco e a cada modificação um novo bloco é gerado e entrelaçado ao original, tornando o ambiente confiável e transparente (NAKAMOTO, 2008).

Tendo em vista que os modelos de aplicação de avaliação de desempenho estão, em sua maioria, sendo aplicados de maneira desatualizada, a elaboração de um aplicativo móvel para que o gestor possa fazer as avaliações de forma prática, transparente e segura são uma demanda atual para novos modelos de gestão.

Embora o planejamento anual seja importante, o acompanhamento em períodos menores é fundamental para alcançar as metas. Avaliações frequentes possibilitam atacar os pontos fracos com mais precisão, economizando assim, um recurso muito valioso: o tempo.

Para chamar a atenção dos colaboradores uma empresa precisa disponibilizar aos mesmos um ambiente de trabalho de confiança mútua. A transparência é uma peça fundamental para o funcionamento desse mecanismo (BUCKINGHAM; GOODALL, 2015).

1.1 OBJETIVOS

1.1.1 Objetivo Geral

O presente trabalho tem como objetivo principal a criação de uma ferramenta de auxílio para a aplicação de avaliação de desempenho. Essa ferramenta será produzida em forma de um software, mais especificamente um aplicativo móvel para o sistema operacional Android. A tecnologia Blockchain será integrada a essa ferramenta para que requisitos como confiabilidade e transparência sejam adicionados às avaliações.

1.1.2 Objetivos Específicos

- Estudar os conceitos e ferramentas relacionados ao desenvolvimento de aplicações descentralizadas, baseadas em Blockchain.
- Tornar a aplicação da avaliação de desempenho prática e rápida.
- Desenvolver uma plataforma de avaliação aplicada pelo gestor aos seus colaboradores, usando Blockchain.
- Evidenciar o funcionamento dos contratos inteligentes, mostrando a movimentação de transações entre contas distintas.

2. FUNDAMENTOS E REVISÃO DE LITERATURA

2.1 AVALIAÇÃO DE DESEMPENHO

Desde o nascimento, passando pela escola, posteriormente pela faculdade e ao longo da profissão, as pessoas são avaliadas. Em cada um desses momentos são necessárias diferentes avaliações. Dentro de uma organização empresarial as avaliações se classificam em três tipos que se diferem pelos momentos: avaliação diagnóstica, avaliação formativa e a avaliação de controle. A avaliação diagnóstica, acontece geralmente no início de uma atividade. A avaliação formativa surge durante o processo de trabalho para auxiliar no desempenho da atividade. Por fim, a avaliação de controle ocorre ao término do processo de trabalho para mensurar o desempenho, potencial ou resultados de uma atividade (MALHEIROS; ROCHA, 2014).

Antes da avaliação de desempenho, é importante conhecer e diferenciar os outros modelos de avaliações que existem nos padrões de gestão. Um desses modelos é a avaliação de resultados, que se assemelha bastante à avaliação de desempenho, sendo que seu foco principal são as metas que os colaboradores devem atingir. Outro modelo é a avaliação de competências, bastante conhecida e utilizada em grande escala, principalmente a partir dos anos 2000. Estes dois modelos são facilmente confundidos com a avaliação de desempenho, não sendo um impeditivo para sua aplicação de forma híbrida. Por fim, o modelo de avaliação de potencial, que é bastante distinta da avaliação de desempenho, pois enquanto uma trabalha com o futuro a outra trabalha com o presente. O que o colaborador pode vir a produzir está sendo medido, enquanto na avaliação de desempenho é mensurado o que ele já produziu (LEME; VESPA, 2012).

Ao analisar rapidamente os diferentes tipos de avaliação pode-se notar que a maioria é semelhante ou relacionada com a avaliação de desempenho. Portanto, reduzir a especificidade do software, de maneira que o gestor possa elaborar um formulário de acordo com a sua necessidade é uma vantagem significativa (BARBIERI, 2016).

Sempre que falamos em avaliação de desempenho, em se tratando de colaboradores, estamos antes de tudo tratando de pessoas. Essas pessoas têm vida fora do trabalho, sentimentos e expectativas. É importante que o gestor tenha paciência ao avaliar seus subordinados, caso contrário são geradas falsas expectativas quando o colaborador é exposto à um *feedback* positivo ou baixar sua autoestima e ânimo quando os resultados não são bons.

Avaliar o desempenho significa avaliar o desempenho atual e/ou passado de um funcionário em relação aos seus padrões de desempenho. É possível corrigir e treinar as

eventuais deficiências do colaborador e também evidenciar as qualidades que lhe são atribuídas. Os colaboradores que passam por esse tipo de avaliação podem trazer benefícios para a empresa ou para o gestor. Com ela é possível mensurar a remuneração e as promoções, e também perceber o enquadramento do colaborador aos objetivos que a empresa almeja. (DESSLER, 2014). Definir previamente os objetivos é parte fundamental da avaliação de desempenho e, para isso, as avaliações de performance podem ajudar a aferir quais itens os colaboradores devem focar para suprir as necessidades da empresa (BARBIERI, 2016).

O fator pessoal também interfere na aplicação da avaliação. Dois gestores podem avaliar diferentemente um mesmo colaborador. Enquanto um gestor pode ser pacífico e tender a dar resultados mais positivos, outro pode ser bastante rígido e não reconhecer os feitos de um bom colaborador. O gestor que aplica a avaliação deve estar ciente que os resultados atribuídos por ele vão impactar diretamente na vida dos colaboradores. Portanto, a avaliação é um processo de crítica bidirecional (RIBEIRO, 2013).

Nem sempre resultados ruins são atribuídos diretamente ao colaborador em questão. O trabalho em conjunto é fundamental dentro de organizações empresariais e problemas internos podem comprometer o funcionamento do sistema como um todo.

Vícios de avaliação são constantes nos meios empresariais. Esses vícios são erros cometidos principalmente pelo fator humano, e devem ser identificados e tratados. A implicância do gestor com algum colaborador, a falta de memória, a supervalorização e até mesmo a falta de técnica são os principais fatores que produzem vícios de avaliação.

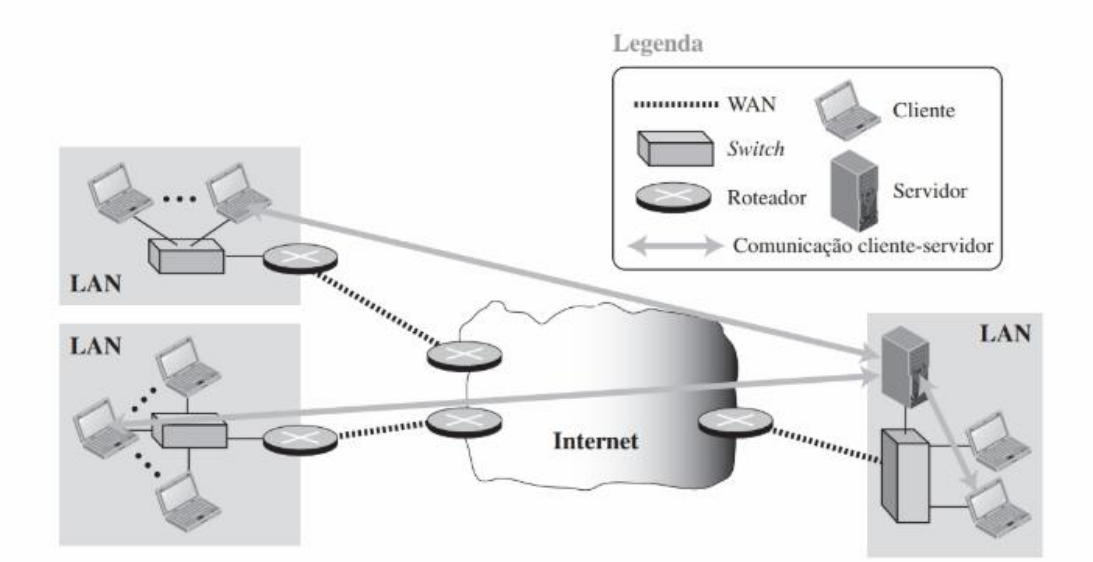
Os processos formais de avaliação são construídos para que a percepção do avaliador não interfira nos resultados finais. Por isso, é aumentado o peso das questões que observam e mensuram determinados aspectos dos colaboradores. Para realizar a mensuração do processo avaliativo é comum a utilização da escala Likert (LIKERT, 1932). Essa escala é usada habitualmente em questionários, sendo possível o avaliador especificar um nível de concordância com uma afirmação. Apesar de ser textual essa escala também pode ser demonstrada em forma numérica com o uso de níveis e faixas de valores, por exemplo: discordo totalmente – 0 a 2, discordo – 2 a 4, neutro – 4 a 6, concordo – 6 a 8 e concordo totalmente – 8 a 10 (MALHEIROS; ROCHA, 2014).

2.2 BLOCKCHAIN

Atualmente, a maior parte dos serviços oferecidos na internet são baseados no paradigma cliente-servidor, onde um dispositivo, geralmente mais robusto, armazena uma

aplicação e aguarda que outros dispositivos solicitem seu acesso, como podemos observar na Figura 1 (FOROUZAN; MOSHARRAF, 2013).

Figura 1- Paradigma cliente-servidor

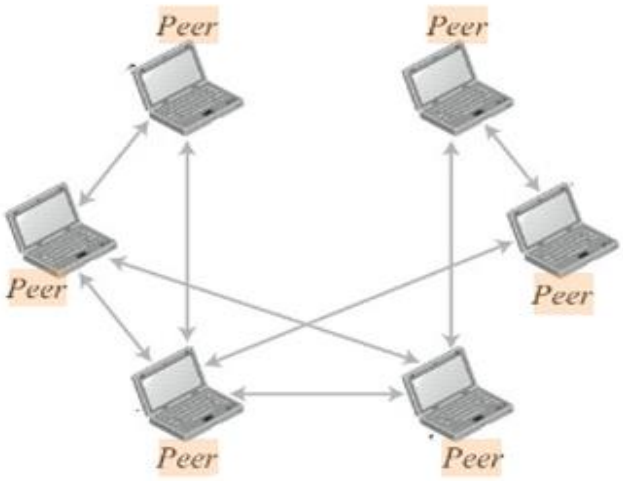


Fonte: (FOROUZAN; MOSHARRAF, 2013).

Um exemplo de aplicação do Blockchain é o Bitcoin[1]. O Bitcoin é um sistema de pagamentos, que tem como objetivo utilizar criptografia para eliminar a necessidade de organizações intermediárias nas transações de comércio eletrônico. Esse sistema resultou na criação de um mecanismo em que indivíduos participantes de uma rede *peer-to-peer* descentralizada, podem trocar criptomoedas diretamente entre si. Uma rede *peer-to-peer* descentralizada tem por característica principal não possuir um dispositivo central, como podemos observar na Figura 2.

[1] https://www.bitcoin.com/

Figura 2- Rede *peer-to-peer.*



Fonte: Adaptado de (FOROUZAN; MOSHARRAF, 2013).

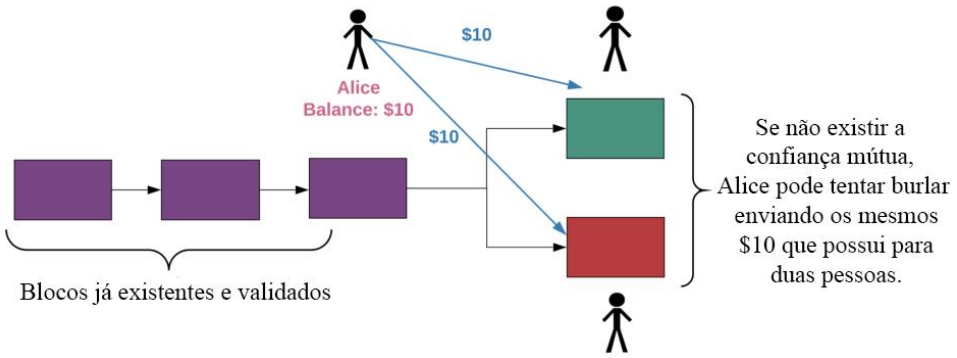
O Blockchain é uma criação de Nakamoto (NAKAMOTO, 2008), que se baseia na tecnologia de *log* seguro para implementar o livro-razão distribuído, utilizado pelo Bitcoin. É usado quando várias partes precisam compartilhar dados ou valores sem confiarem uns nos outros. Um serviço de *log* seguro tem o papel de assegurar a existência de determinado documento digital antes de um ponto específico no tempo (UNE, 2001).

Nessa tecnologia, os usuários não possuem moedas físicas, mas sim um histórico de transações efetuadas e recebidas, registradas em um livro-razão. Nesse sentido, se um usuário possui *bitcoins,* então ele possui transações endereçadas a ele, registradas neste livro.

Um dos desafios enfrentados na implementação desse sistema consiste no problema do gasto duplo da mesma criptomoeda, de forma que uma mesma transação pode ser utilizada mais de uma vez, por consistir em um arquivo digital que pode ser duplicado ou falsificado (CHOHAN, 2017).

A Figura 3 ilustra uma situação hipotética, em que uma usuária do Bitcoin possui transações endereçadas a ela no valor de 10 *bitcoins* e tenta transferir essa quantia para dois usuários diferentes, referenciando as mesmas transações mais de uma vez.

Figura 3- Situação hipotética demonstrando o problema do gasto duplo.



Fonte: Adaptado de (KASIREDDY, 2018).

Nesse caso, é necessário estabelecer um critério de desempate para que a mesma moeda não seja gasta mais de uma vez. A estratégia utilizada considera válida somente a primeira transação a ser registrada. Essa estratégia é conhecida como *first-to-file* (BUTERIN et al., 2014).

Para verificar se uma transação já foi utilizada, é necessário que todos os participantes da rede tenham conhecimento sobre as transações já registradas até o momento. Desta forma, em contraste com o modelo centralizado adotado pelos bancos, as transações que ocorrem no *Blockchain* são visíveis publicamente, como ilustra a Figura 4. Observando a imagem é possível identificar algumas estruturas indicadas por contornos coloridos, de forma que o contorno em azul indica as entradas de uma transação, o contorno em verde indica suas saídas e o contorno em vermelho indica seu identificador no sistema.

Figura 4- Elementos de uma transação no Blockchain.

Fonte: Adaptado de (BLOCKCHAIN, 2018).

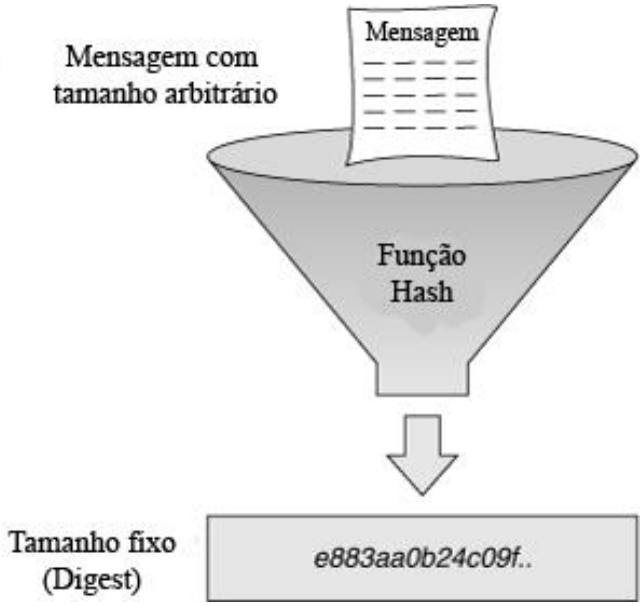
Ambas as entradas e saídas da transação estão representadas por pares endereço-quantia, denominados saídas não-gastas de transação ou UTXO (do inglês *Unspent Transaction Output*), de forma que as UTXO que entram são gastas e as UTXO que saem são geradas.

2.2.1 Funções de *hash*

O Bitcoin é constituído por uma cadeia de assinaturas digitais que envolvem o *hash* da transação anterior e a chave pública do beneficiário da transação corrente. Na matemática aplicada à criptografia, uma função de *hash* é definida como uma operação de via única e irrevogável, cuja aplicação resulta em uma saída de tamanho fixo, que só pode ser obtida por meio de uma entrada idêntica (CABRAL; CAPRINO, 2015).

A Figura 5 ilustra esse processo, onde uma mensagem de texto com tamanho arbitrário ao passar por um funil, que representa a função de *hash*, transforma-se em um código de tamanho único, denominado *digest.*

Figura 5- Processo de aplicação da função de *hash*.



Fonte: Adaptado de (MATETI, 2017).

O comprimento da saída gerada depende do algoritmo de *hash* escolhido. No caso do Bitcoin, é utilizado o algoritmo SHA-256 (BUTERIN et al., 2014), que produz uma saída de 256 bits.

Algoritmos de *hash* são construídos por meio de complexos arranjos de operações matemáticas encadeadas, projetadas de forma a oferecer uma baixa probabilidade de colisão entre os *hashes* de duas mensagens distintas.

2.2.2 Criptografia assimétrica

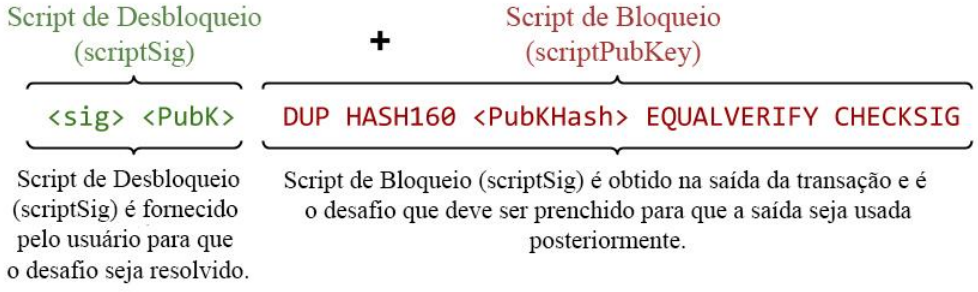
A criptografia assimétrica utiliza uma chave pública para cifrar e uma chave privada para decifrar uma mensagem. Quanto uma mensagem é cifrada utilizando a chave privada, está se aplicando uma assinatura digital, pois ela só poderá ser decifrada utilizando a chave pública correspondente (CANAVAN, 2001).

As UTXOs geradas no sistema Bitcoin são matematicamente vinculadas às chaves criptográficas de quem as possui. Dessa forma, assinaturas digitais podem ser usadas para provar a posse de uma chave privada sem que seja necessário revelá-la, possibilitando assim, o desbloqueio das UTXOs para seus respectivos donos em novas transações. (BARRERA, 2014).

Quando ocorre uma transação no Bitcoin, as UTXOs usadas na entrada precisam ser desbloqueadas e as UTXOs geradas na saída precisam ser bloqueadas. Dessa forma, quando um usuário transfere *bitcoins* a alguém, este deve primeiro provar que possui as UTXOs da entrada, fornecendo uma assinatura digital. Após essa comprovação, deve fornecer a chave pública do destinatário da transação para que um *script* seja gerado e bloqueie a nova UTXO, assim apenas o possuidor da respectiva chave privada possa desbloqueá-la.

A Figura 6 ilustra os scripts de bloqueio e desbloqueio de uma transação. O *script* de bloqueio recebe o *script* de desbloqueio como argumento e verifica se a assinatura fornecida corresponde à chave pública especificada.

Figura 6- Scripts de desbloqueio e bloqueio.



Fonte: Adaptado de (ANTONOPOULOS, 2017).

2.2.3 Prova de trabalho

Para resolver o problema do gasto-duplo no Bitcoin, as transações são organizadas em blocos e encadeadas cronologicamente no Blockchain. Com isso, é necessário um mecanismo

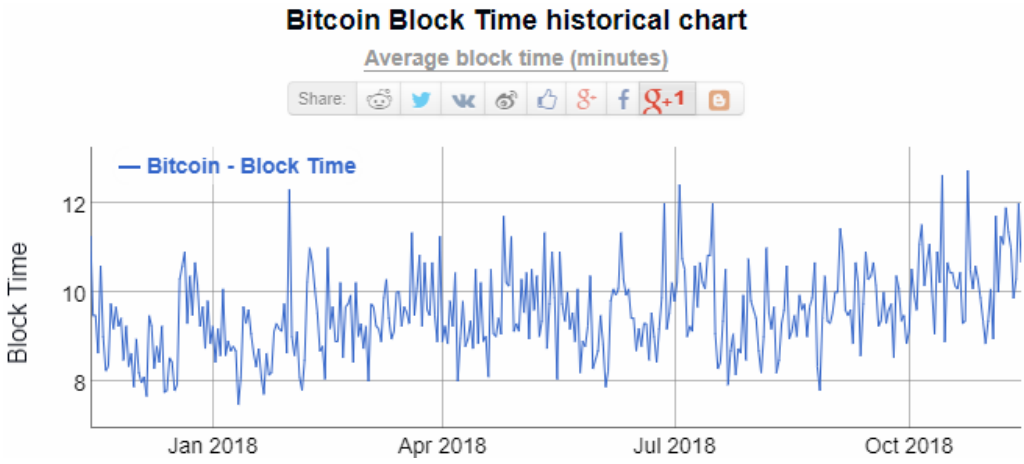
de consenso para que os participantes concordem em um histórico único, que contém a ordem em que os blocos foram recebidos.

O mecanismo de consenso empregado pelo Bitcoin é chamado de prova de trabalho (*proof-of-work*). Participantes da rede que desejam adicionar blocos de transação ao histórico compartilhado são denominados mineradores. Nesse mecanismo os mineradores competem entre si para resolver um desafio matemático e, por conseguinte, adicionar os blocos ao histórico de transações. Esse desafio envolve a obtenção de um *hash* criptográfico com uma determinada quantidade de zeros à esquerda.

O trabalho médio necessário para resolver o desafio é exponencialmente proporcional ao número de *bits* zero requeridos pela rede, porém o resultado pode ser verificado pela aplicação única de um algoritmo de *hash* (NAKAMOTO, 2008).

A Figura 7 apresenta o histórico de confirmação dos blocos no período de Novembro/2017 a Novembro/2018. Como pode ser observado, o tempo se mantém ao redor de 10 minutos. Isto se deve ao fato de que a cada 2016 blocos, aproximadamente duas semanas, o nível de dificuldade do desafio é reavaliado para que este tempo se mantenha (BITCOINWIKI, 2018).

Figura 7- Histórico do tempo de bloco.



Fonte: (BITINFOCHARTS, 2018).

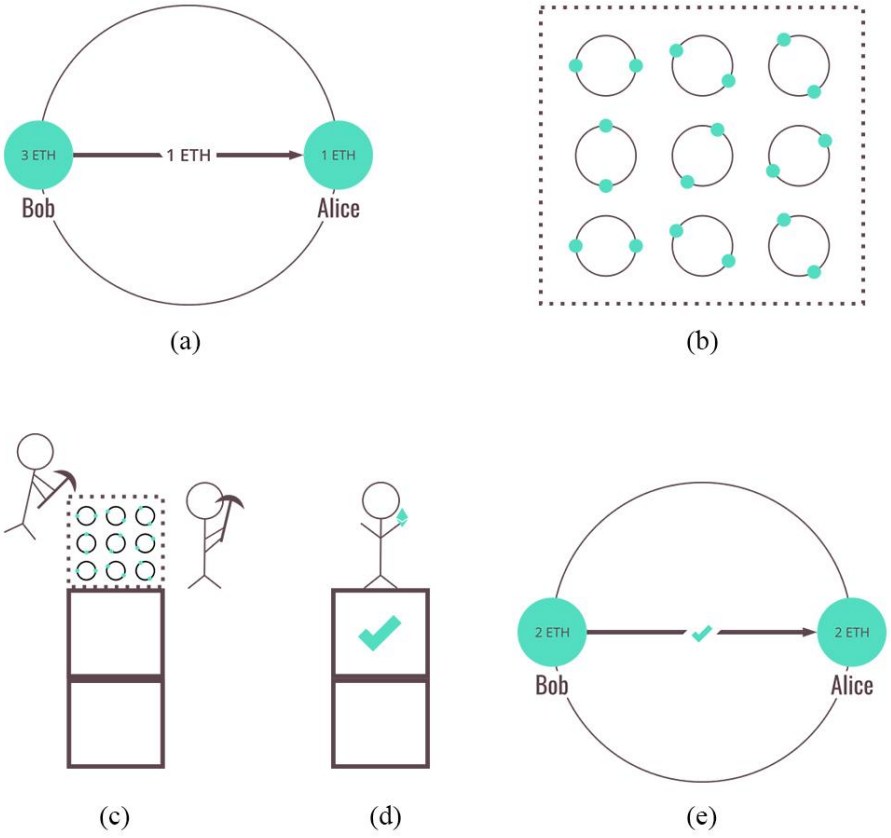
2.3 ETHEREUM

Ethereum é uma tecnologia de livro-razão distribuído (*Distributed Ledger Technology* - DLT), que permite a escrita de contratos inteligentes e aplicações descentralizadas por meio de uma máquina virtual distribuída (BUTERIN et al., 2014).

Diferentemente do Bitcoin, que é um sistema especializado para operar com transações financeiras, o Ethereum tem um escopo mais amplo e sua máquina virtual *Turing*-completa, a *Ethereum Virtual Machine* (EVM) é capaz de executar qualquer aplicação que possa ser definida em termos de um algoritmo.

O Ethereum mantém o estado da EVM no Blockchain e todos os nós mineradores processam e verificam a integridade dos contratos inteligentes.

Figura 8- Como funciona a mineração de um sistema Ethereum.



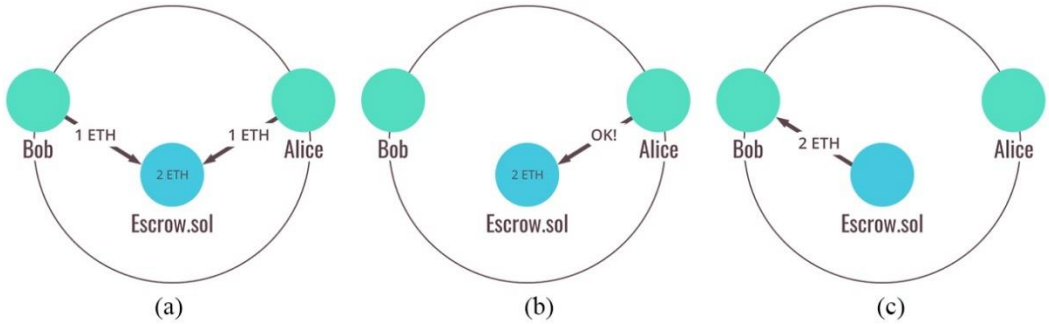
Em (a) Bob tenta enviar 1 ETH para Alice. (b) A transação de Bob e Alice é combinada com outras transações que ocorreram desde o último bloco. (c) Mineradores competem para validar o bloco com o novo conjunto de transações. (d) O minerador vitorioso cria um novo bloco e recebe uma recompensa. (e) Com a transação validada, Alice recebe 1 ETH.
Fonte: Adaptado de (TRUFFLE, 2018).

2.3.1 Contratos inteligentes

Contratos inteligentes são trechos de software associados às contas do Ethereum que desbloqueiam os valores nelas armazenadas mediante a satisfação de determinadas condições (BUTERIN et al., 2014).

A Figura 9 ilustra um exemplo de aplicação em que um contrato inteligente é utilizado como parte neutra para armazenar o pagamento e como garantia de uma negociação que envolve os atores Bob e Alice. Neste exemplo, Alice contratou Bob para construir um pátio e os dois concordaram em depositar 1 *ether* em um contrato de custódia (*escrow*). Caso Bob realize o serviço corretamente, ele poderá sacar o valor de 2 *ether.* No entanto, caso o compromisso não seja cumprido, Alice resgata o seu depósito e o de Bob.

Figura 9- Funcionamento de um contrato inteligente.



Em (a) Alice concorda em armazenar seu pagamento para o pátio dentro do contrato de custódia, e Bob concorda em depositar uma quantia igual. (b) Bob conclui o projeto do pátio e Alice dá permissão ao contrato inteligente para liberar os fundos. (c) Bob recebe o pagamento de Alice junto com sua garantia. Fonte: Adaptado de (TRUFFLE, 2018).

2.3.1.1 Linguagem Solidity

Muitas linguagens de programação foram criadas para o desenvolvimento de contratos inteligentes, tais como *Low-Level Lisp-Like Language* (LLL)[2], Serpent[3] e Mutan. No entanto, atualmente os contratos inteligentes são escritos majoritariamente na linguagem Solidity[4]. As linguagens *Serpent* e *Mutan* foram descontinuadas e a linguagem LLL ainda é usada para casos muito específicos (ORLICKI, 2018).

A LLL é uma linguagem de baixo nível semelhante Assembly. Destina-se a ser muito simples e minimalista, sendo essencialmente apenas uma pequena camada de abstração sobre codificação em EVM diretamente (ETHDOCS, 2018).

[2] https://lll-docs.readthedocs.io/en/latest/index.html
[3] https://github.com/ethereum/serpent
[4] https://github.com/ethereum/solidity

2.3.1.2 Gas

No Ethereum, a leitura de dados do Blockchain é uma operação gratuita, mas a escrita não. O custo de uma transação é denominado *gas*, em referência ao termo em inglês *gasoline,* e tem o propósito de limitar a quantidade de trabalho necessária para executar a transação e também de pagar por essa execução (SOLIDITY, 2018).

De certa forma, *gas* é uma medida de complexidade de uma transação. À medida que o número de operações realizadas por um contrato inteligente aumenta, o custo da transação também aumenta.

2.3.2 Contas

Uma conta no Ethereum pode ser de dois tipos: conta de propriedade externa (*Externally Owned Account* - EOA), ou conta de contrato.

Essas contas contêm quatro campos:

- *Nonce:* um contador usado para garantir que cada transação possa ser processada apenas uma vez;
- Saldo atual da conta;
- Código do contrato da conta, se presente;
- Armazenamento da conta (vazio por padrão).

Como se pode observar, o Ethereum apresenta importantes diferenças em relação ao Bitcoin. No Ethereum, as UTXOs dão lugar ao conceito de saldo e a verificação de gasto-duplo é feita por meio do campo *nonce*.

Contratos no Ethereum são considerados “cidadãos de primeira classe”. Toda vez que uma conta de contrato recebe uma mensagem seu código é ativado, o que permite a leitura e a escrita no armazenamento interno, o envio de novas mensagens e a criação de novos contratos (BUTERIN et al., 2014).

2.3.2.1 Redes Ethereum

Uma rede Ethereum pode ser pública ou privada. Dentre as redes públicas, existem redes de teste e existe a *Main Network*, que é a principal rede dessa tecnologia e trafega o maior volume diário de transações.

As redes públicas de teste são três:

- Ropsten: funciona de forma semelhante à *Main Network;*

- Kovan: utiliza um algoritmo de consenso chamado *proof-of-authority*[5]. As transações são validadas por membros selecionados e não é necessário realizar o procedimento de mineração.
- Rinkeby: também é baseada no algoritmo *proof-of-authority*, porem adota o algoritmo *Clique*[6] como mecanismo de consenso.

2.3.2.2 Carteiras

No ambiente Etherum, uma carteira consiste em um software que armazena chaves criptográficas relacionadas a EOA e realiza transações tanto em redes públicas como privadas.

As carteiras precisam manter cópias atualizadas do Blockchain para operar. Ao abrir o software é necessário aguardar o processo de sincronização.

Existem serviços na internet que fornecem acesso remoto a clientes Ethereum que rodam em infraestruturas de nuvem. Ao usufruir desses serviços, o usuário não precisa se preocupar com o armazenamento e nem a sincronização do Blockchain. Um exemplo de tal serviço é o Infura[7], oferecido pela empresa ConsenSys[8].

2.3.2.3 Aplicativos distribuídos (DApps)

Distributed Applications (DApps) são aplicativos distribuídos que usam contratos inteligentes. É a aplicação do conhecimento adquirido acerca do Ethereum Blockchain em inúmeras situações reais, como o que é proposto nesse trabalho (TRUFFLE, 2018).

[5] https://wiki.parity.io/Proof-of-Authority-Chains
[6] https://steemit.com/steemstem/@mareng/clique-algorithm-proof-of-authority-consensus
[7] https://infura.io/
[8] https://consensys.net/

3. DESENVOLVIMENTO DO APLICATIVO

3.1 FERRAMENTAS UTILIZADAS

O aplicativo foi desenvolvido através do Android Studio[9], ferramenta oficial de desenvolvimento Google[10], lançado oficialmente em Dezembro de 2014. O Android Studio permite o desenvolvimento integrado da parte gráfica e da parte funcional do aplicativo, além de fornecer um ambiente completo para testes em máquinas virtuais e dispositivos reais.

O banco de dados utilizado foi desenvolvido usando a base de dados nativa do Android, *SQLite*[11]. Nele foram criadas e manipuladas todas as tabelas e realizadas todas as consultas necessárias para o funcionamento do aplicativo proposto.

Para a criação dos Contratos Inteligentes, foi utilizado o *framework* Visual Studio Code[12], com o plugin de suporte da linguagem de programação Solidity. A integração dos Contratos Inteligentes com o aplicativo se dá por meio de um compilador chamado Solc[13] e a biblioteca de integração chamada Web3j[14], suportada na linguagem de programação Java[15]. Finalmente o gerenciamento e mineração de contratos foi feito por meio de uma rede local, por meio do *framework* Ganache[16], que faz toda a simulação necessária para o funcionamento do ambiente.

3.2 ESTRUTURA DO APLICATIVO

O aplicativo foi desenvolvido utilizando o conceito de navegação entre telas de atividades, ou *Activities*, onde cada atividade está ligada a um *layout* gráfico e a uma ou um conjunto de classes Java onde são designados o que cada componente da atividade vai realizar. As atividades estão dispostas no aplicativo da seguinte forma:

[9] https://developer.android.com/sdk/
[10] https://www.google.com/
[11] https://www.sqlite.org/index.html
[12] https://code.visualstudio.com/
[13] https://www.npmjs.com/package/solc
[14] https://web3j.io/
[15] https://www.oracle.com/br/java/
[16] https://trufflframework.com/ganache

Figura 10- Diagrama de navegação do aplicativo.

O diagrama mostra a navegação das telas, onde as folhas são os níveis mais profundos onde se pode chegar, e a raiz é a tela principal.

Fonte: Autoria própria.

3.3 CONEXÃO COM O ETHEREUM BLOCKCHAIN

Para interagir com contratos inteligentes, geralmente é necessário fazer o download de todo o Blockchain do Ethereum e manter um nó local sincronizado. Isso pode fazer sentido caso você esteja trabalhando em um computador de forma comercial, mas em um dispositivo móvel isso se torna inviável.

Para contornar essa situação, utilizaremos a biblioteca Web3j, em conjunto com o compilador da linguagem Solidity e um f*ramework* que simula uma rede privada, onde mineradores encontram blocos após transações. O Web3j fornece utilitários para gerar instancias de contratos inteligentes em Java e uma implementação completa da *Application Programming Interface* (API) do cliente JSON-RPC - protocolo de chamada de procedimento remoto, codificado em *JavaScript Object Notation* (JSON)[17] - que o Ethereum utiliza sobre *Hypertext Transfer Protocol* (HTTP) e *Inter-Process Communication* (IPC).

Para gerar uma classe Java do mesmo formato do contrato inteligente, primeiramente devemos usar o compilador Solidity Solc. Esse compilador gera um arquivo binário e uma *Application Binary Interface* (ABI) do contrato em questão. Posteriormente utilizaremos o Web3j para manipular esses arquivos gerados de forma que isso dê origem a todas as classes necessárias para o cliente se comunicar com o contrato inteligente que já se encontra na rede.

[17] https://www.json.org/

3.4 BIBLIOTECAS ADICIONAIS

Além das bibliotecas padrão do *Software Development Kit* (SDK), foram necessárias outras bibliotecas para implementação de recursos extras. Foram utilizadas: GraphView e Web3j.

3.4.1 HelloCharts

A biblioteca HelloCharts[18] é utilizada para a criação de gráficos de forma simples e intuitiva em aplicativos Android.

Cada avaliação é constituída por uma série de perguntas e suas respectivas respostas. Como as avaliações são realizadas mensalmente, cada gráfico exibe o desempenho mensal de cada pergunta do formulário em um gráfico de linhas. Nesse caso, são utilizados gráficos de linhas, mas a ferramenta permite utilizar outros tipos de gráficos como gráficos de barras, gráficos de pizza, etc.

Os dados inseridos nos gráficos são resultantes de uma consulta ao banco de dados que retorna uma lista de avaliações dado um colaborador. Dentro de cada avaliação as perguntas são filtradas pelo seu identificador e distribuídas de forma que o usuário consiga ver o desempenho individual de cada pergunta, comparando os meses que foram feitas as avaliações.

Dessa forma é fácil identificar onde o colaborador teve um aumento ou declínio de performance nos seus deveres mensais.

3.4.2 Web3j

A biblioteca Web3j possibilita gerar automaticamente *wrappers* para interagir com contratos inteligentes sem deixar de utilizar a *Java Virtual Machine* (JVM). Wrapper é uma classe empacotadora que torna possível a manipulação de informações como um único pacote. Tal biblioteca facilita o desenvolvimento da comunicação do aplicativo com os contratos inteligentes.

[18] https://github.com/lecho/hellocharts-android

3.5 PERMISSÕES NECESSÁRIAS

O gerenciamento de permissões é uma função muito importante em dispositivos com sistema Android. As permissões gerenciam quais recursos do aparelho estarão disponíveis para o acesso do aplicativo. Todas as funções, atividades e autorizações necessárias para a execução deste aplicativo estão especificadas no arquivo AndroidManifest.xml. São elas:

- WRITE_EXTERNAL_STORAGE: este é um requisito para que o banco de dados nativo do Android consiga escrever sua base em dispositivos externos de armazenamento, como um cartão de memória, por exemplo.
- READ_EXTERNAL_STORAGE: da mesma forma que o Android precisa escrever dados em sua base, é necessário ler os dados que nela estão escritos. É exatamente isso que essa permissão faz.

3.6 TELAS GERADAS

Esta seção aborda como cada tela do aplicativo foi desenvolvida, tanto no nível gráfico (arquivos XML), quanto a nível funcional (arquivos Java).

3.6.1 Tela inicial

Figura 11- Tela inicial do aplicativo.

Fonte: Autoria própria.

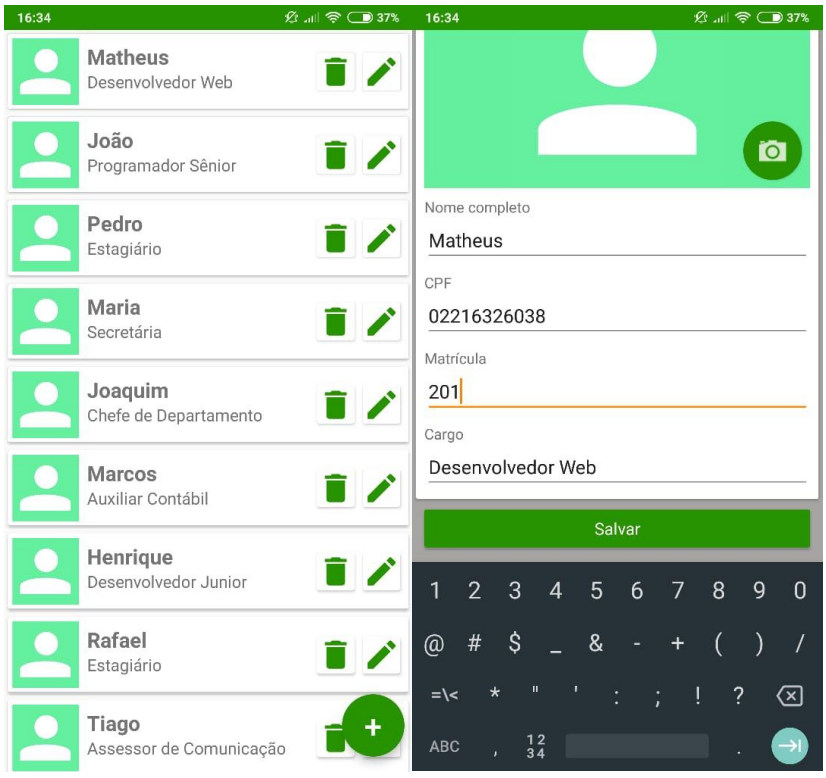
A tela inicial, ilustrada na Figura 11, possui um cabeçalho com o nome do gestor, usuário principal do aplicativo e seu cargo, com uma imagem de fundo e uma lista de menus. Esse arquivo gráfico tem o nome activity_menu.xml. A classe que gerencia essa tela tem o nome MenuActivity.java.

Cada menu é instanciado dentro da tela principal na forma de um item, contido no arquivo new_item_menu.xml. Esses itens contêm simplesmente um *CardView* com um fundo opaco e um texto, o qual será informado o título do menu.

Quando é feita a identificação de cada componente da tela, são definidos os títulos de cada menu e explicitadas suas intenções, conhecidas como *Intents* no Android. A distribuição dos menus ocorre como já citado na estrutura do aplicativo.

3.6.2 Tela de listagem dos colaboradores

Figura 12- Tela destinada aos colaboradores.



Fonte: Autoria própria.

Esta tela, com arquivo gráfico chamado activity_colaboradores_cadastro.xml e arquivo lógico chamado CadastroColaboradoresActivity.java, mostra todos os colaboradores cadastrados que fazem parte do plantel pessoal do gestor. A primeira ação, caso o gestor não possua colaboradores cadastrados, é clicar no botão de ação *Floating Action Button* que se

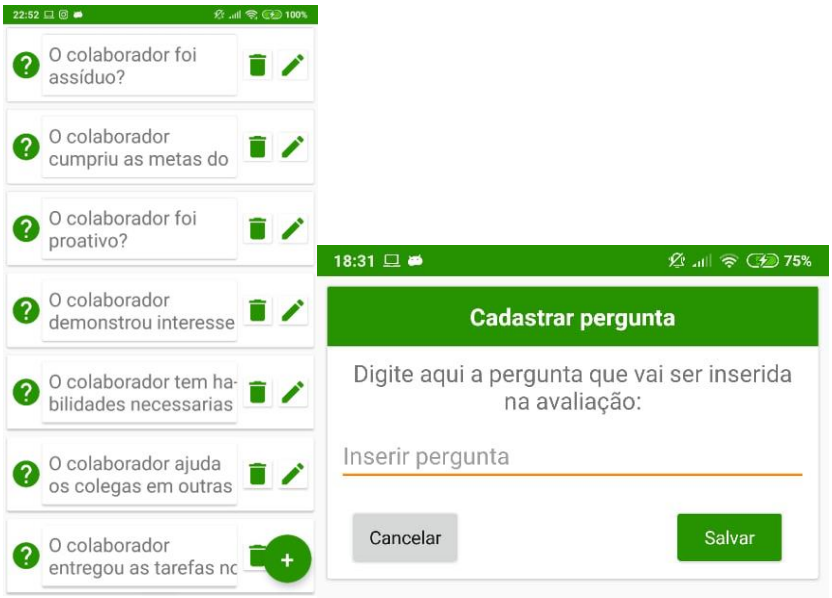
encontra na parte inferior da tela para adicionar um novo colaborador. Ao clicar nesse botão, o gestor é direcionado até a atividade Formulário Cadastro de Colaboradores, que será abordada posteriormente.

A medida que os colaboradores são cadastrados, consultas ao banco de dados são feitas e uma lista é criada. Para mostrar essas informações na tela é usada uma *RecyclerView*. Entretanto, essa visualização não é feita de maneira simples. É necessário criar uma classe Adapter para que as informações sejam disponibilizadas da forma correta. Uma classe Adapter trata individualmente cada elemento de uma lista para exibi-lo de forma personalizada em um item de uma lista. Isso é feito na classe ListaColaboradoresAdapter.java.

Caso o gestor tenha inserido alguma informação de forma equivocada, ele poderá editar o colaborador individualmente clicando no ícone de edição localizado ao lado de cada colaborador. Se a necessidade de excluir algum colaborador do quadro esteja presente, basta clicar no ícone que representa uma "lixeira" ao lado do colaborador. Uma mensagem será exibida em um *AlertDialog*, perguntando se tem certeza que deseja excluir o colaborador selecionado. Ao aceitar a mensagem o colaborador é removido da lista.

3.6.3 Tela Formulários

Figura 13- Tela destinada às perguntas do formulário.



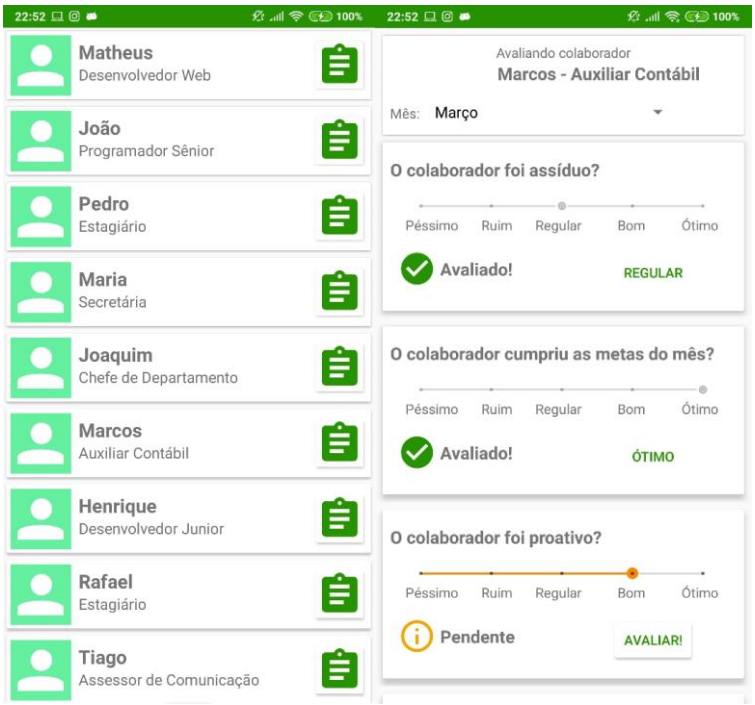
Fonte: Autoria própria.

Esta tela é muito semelhante a anterior. Seu arquivo de *layout* é activity_cadastrar_perguntas.xml e sua parte lógica é CadastrarPerguntas.java. Da mesma

forma que os colaboradores e as perguntas que compõe o formulário são apresentados em forma de lista. Também é possível adicionar novas perguntas ao formulário, editá-las ou excluí-las. Um adapter também é necessário nesse caso, para que a listagem das perguntas seja feita de forma correta em uma *RecyclerView*. O arquivo que corresponde ao *adapter* é chamado de ListaPerguntasAdapter.java.

3.6.4 Tela Realizar avaliação

Figura 14- Tela destinada à realização de avaliações.



Fonte: Autoria própria.

É nesta tela que o objetivo do aplicativo é executado. Nela são primeiramente listados todos os colaboradores que foram adicionados na tela de cadastro. Porém, a diferença da tela de cadastro é que o ícone ao lado de cada colaborador remete a um formulário, deixando intuitivo para o gestor que aquele ícone é exclusivo para realizar uma avaliação sobre aquele determinado colaborador.

Após escolher o colaborador para ser avaliado, o gestor seleciona o respectivo mês. Essa informação é exibida na forma de um *Spinner*, onde são mostrados os doze meses mais uma opção de verificação, que muda a visibilidade das perguntas do formulário e dos botões de ação.

Com o mês da avaliação selecionado, o gestor está apto a responder as perguntas que foram previamente cadastradas por ele, a respeito do colaborador selecionado. As perguntas são

exibidas na forma de uma lista, dentro de uma *RecyclerView*, com os respectivos layouts e arquivos lógicos: activity_responder_perguntas.xml, ResponderPerguntasActivity.java e PerguntasRealizarAdapter.java.

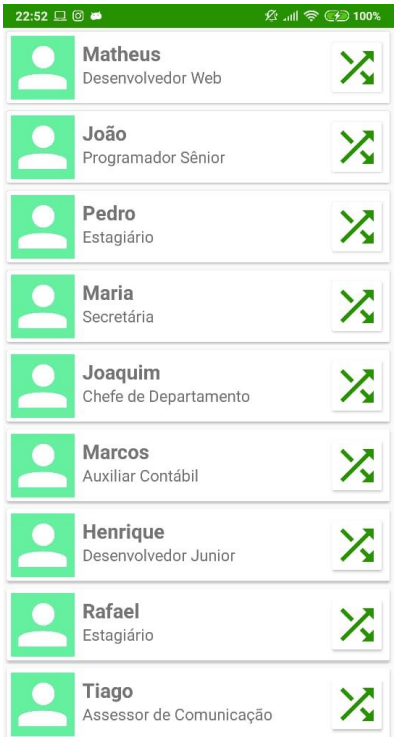
A medida que as perguntas vão sendo respondidas é necessário validar cada resposta inserida individualmente. Isso é feito com um botão que fica abaixo da escala escolhida, e o disparo de um alerta caso a resposta não tenha sido confirmada.

Tendo o gestor respondido todas as perguntas do questionário é hora de salvar toda a avaliação realizada. Nesse momento é instanciado um alerta para que o gestor verifique todas as respostas inseridas e tenha certeza que todas foram validadas de forma correta. Quando o gestor der sua resposta positiva à verificação das perguntas, o software vai instanciar um objeto da classe Avaliação, que contém a avaliação propriamente dita. Esse objeto vai se relacionar com um contrato inteligente já presente na rede Blockchain, validando assim a avaliação efetuada. Essa operação ocorre da forma que já foi abordado anteriormente, transformando o contrato em *bytecode*, posteriormente em formato JSON e, finalmente, em um objeto Java pelo compilador Solidity e a biblioteca Web3j.

É importante salientar que o contrato inteligente suprime algumas informações do colaborador, como o seu nome, CPF e cargo. Apenas o número da matrícula é publicado no Ethereum Blockchain. Isso ocorre por que as informações que são postadas no Blockchain são públicas e imutáveis, portanto, não seria interessante que todas as pessoas que acessam a rede tivessem essas informações. Para o controle de avaliações por colaborador apenas o número de matrícula ou assinatura digital é suficiente. O gestor pode consultar o banco de dados próprio ou o da empresa para identificar um colaborador específico.

3.6.5 Tela Resultados

Figura 15- Tela destinada aos resultados.



Fonte: Autoria própria.

A avaliação realizada se torna pouco importante se não for possível visualizar os resultados de todas as avaliações feitas sobre o colaborador em questão de forma simples e intuitiva. Pensando nisso, foi criada a tela de resultados onde o gestor pode visualizar, em forma de gráficos, os resultados mensais das perguntas presentes no formulário.

Cada pergunta contida no formulário ganha um gráfico exclusivo em que, no eixo das abcissas são distribuídos os meses que foram realizadas as avaliações e, no eixo da ordenadas, são distribuídas as notas atribuídas a cada mês.

Para gerar o gráfico é primeiramente feita uma consulta ao banco de dados, a qual retorna todas as avaliações realizadas daquele colaborador. Em seguida as perguntas são agrupadas e as notas são distribuídas conforme o mês da avaliação. Essa lógica é feita na classe GraficosColaborador.java.

3.7 RESULTADOS

Segundo Nielsen e Landauer (NIELSEN; LANDAUER, 1993) é necessário aplicar um teste de usabilidade em somente cinco usuários para obter resultados satisfatórios. O número

de informações que são adquiridas a partir de cinco usuários começam a se repetir e o coeficiente de crescimento da curva de novos problemas do software se torna pequeno.

Nesse sentido, foi aplicado o formulário contido no Apêndice A a cinco pessoas que se disponibilizaram a utilizar o aplicativo. Das cinco pessoas que responderam o questionário, a maioria (3 de 5) possui entre 26 e 35 anos com ensino superior completo em áreas de formação diferentes. Dentre os participantes, um se destaca pela formação em psicologia e sua área de atuação ser voltada para a gestão de pessoas, tema de enfoque para esse trabalho. Esse mesmo usuário foi o único que respondeu a questão número 9 de forma afirmativa tornando apto a responder a questão 10, na qual sinalizou aplicar suas avaliações anualmente.

A totalidade de usuários utiliza o sistema Android nos seus dispositivos móveis e a maioria (3 de 5) utiliza seus dispositivos entre 5 e 10 vezes por dia, durante o trabalho. Todos os participantes quando avaliaram o desempenho do aplicativo consideraram que o processo de avaliação foi ágil e prático, além disso, responderam que se não houvessem sido informados, não teriam percebido que o aplicativo utiliza a tecnologia Blockchain integrada.

Alguns usuários deixaram considerações finais, sendo eles relatados abaixo:

Usuário A, disse:

"O propósito de agregar o Blockchain à avaliação de desempenho é realmente inovador e acredito que tenha um grande potencial de crescimento. O aplicativo em si possui algumas inconsistências, mas como tu mesmo deixou bem claro, ainda é um protótipo. Acredito que esses detalhes sendo corrigidos e melhorados, o aplicativo tem um grande potencial!"

Usuário B, disse:

"Achei que o aplicativo faz o que é proposto, porém suponhamos que eu sou um gestor que administra dois grupos distintos de colaboradores, e quero aplicar duas avaliações com questões diferentes para os dois grupos. Eu não consigo fazer isso com o teu aplicativo."

Usuário C, disse:

"Gostei muito do aplicativo, nunca tinha visto algo assim. Acho que não teria alguma melhoria para fazer, só quem usaria mensalmente como realmente identificaria falhas. A única preocupação é com o gestor que pode avaliar como ele bem entender, sem que o colaborador possa contestá-lo."

Esses resultados foram satisfatórios pois o objetivo do trabalho é que o aplicativo produzido auxilie na aplicação da avaliação de desempenho, de forma que a utilização do Blockchain não atrapalhe em nenhum aspecto do processo.

4. CONCLUSÃO

Este trabalho teve como objetivo principal integrar a tecnologia *Blockchain* a um aplicativo de avaliação de desempenho. Tendo em vista os resultados encontrados durante o desenvolvimento, conclui-se que esse objetivo foi cumprido. Apesar desta tecnologia estar em crescimento exponencial, pode-se observar que ainda é difícil encontrar material explicativo a respeito à aplicação dela em aplicativos móveis. Em razão disso, este trabalho demandou muita pesquisa, sendo necessário fazer contato com pessoas de outros países em fóruns sobre o assunto e até adquirindo cursos que guiaram na conclusão de algumas etapas da implementação do aplicativo.

4.1 TRABALHOS FUTUROS

Apesar do sucesso da proposta do trabalho, muitos outros requisitos ainda podem ser implementados para que o sistema se torne estável, podendo aplicá-lo como produto final. Por exemplo, utilizar a rede principal do Blockchain para validar os contratos inteligentes. Também poderá ser implementado um sistema de *login*, onde cada gestor terá seu próprio plantel de colaboradores e também suas próprias perguntas para serem aplicadas nas avaliações. Para dar suporte a esse sistema individual, outro requisito será um serviço de *Back-end* juntamente com um banco de dados, no qual serão tratados e armazenados os dados individuais de cada usuário.

Da mesma maneira, o aplicativo poderá ser melhorado de forma com que se torne mais semelhante a realidade das avaliações de desempenho, como por exemplo, possibilitar a alteração da escala de avaliação para abranger outros domínios. Dar poder ao gestor para adicionar algum comentário ao final da avaliação também é uma função importante a ser implementada.

Mais além da melhoria do aplicativo já criado, seria a criação de outro aplicativo que se comunicasse com o produzido, de forma que o cruzamento das informações geraria um gráfico de compatibilidade, expondo a porcentagem de similaridade que possui a avaliação do gestor com a autoavaliação do colaborador.

APÊNDICE A

Formulário teste de usabilidade CollabRate

1- Qual a sua faixa etária?
( ) Abaixo de 15 anos ( ) 16 a 25 anos ( ) 26 a 35 anos ( ) 36 a 45 anos
( ) Acima de 46 anos

2- Qual o seu grau de escolaridade
( ) Ensino Fundamental Completo
( ) Ensino Médio Incompleto
( ) Ensino Médio Completo
( ) Ensino Superior Incompleto
( ) Ensino Superior Completo
( ) Outro: _____

3- Qual a sua formação?
_____

4- Qual a sua área de atuação?
_____

5- Qual a plataforma do seu celular?
( ) Android ( ) iOS ( ) Outro

6- Com que frequência você utiliza o celular durante o trabalho? (número de vezes ao dia)
( ) Menos de 5 ( ) de 5 a 10 ( ) de 10 a 15 ( ) mais de 15

7- Na sua opinião, o aplicativo foi útil quanto a agilidade da avaliação?
( ) Sim ( ) Não

8- Se você não soubesse que o aplicativo utiliza a tecnologia Blockchain, você teria notado?
( ) Sim ( ) Não

9- Você realiza avaliação de desempenho no seu trabalho?
( ) Sim ( ) Não

Caso tenha respondido “não” à pergunta acima, desconsidere a seguinte questão:

10- Com que frequência você realiza avaliação de desempenho sobre seus colaboradores?
( ) Semanal ( ) Mensal ( ) Bimestral ( ) Semestral ( ) Anual

Considerações adicionais:
_____

5. REFERÊNCIAS

ANTONOPOULOS, A. M. **Mastering Bitcoin: unlocking digital cruptocurrencies.** O'Reilly Media, Inc, 2014.

BARBIERI, U. F. **Gestão de Pessoas nas Organizações – Conceitos Básicos e Aplicações.** São Paulo, Atlas, 2016, p. 42.

BARRERA, A. **A Guide to Bitcoin (Part I): A look under the hood.** 2014. Disponível em: <http://tech.eu/features/808/bitcoin-part-one/>

BITCOINWIKI. **Difficulty – Bitcoin Wiki.** 2018. Disponível em: <https://en.bitcoin.it/wiki/Difficulty>

BITINFOCHARTS. **Bitcoin Block Time chart.** 2018. Disponível em: <https://bitinfocharts.com/comparison/bitcoin-confirmationtime.html>

BLOCKCHAIN. **Bitvoin Block 523452.** 2018. Disponível em: <https://www.blockchain.com/btc/block/000000000000000000208cbf4c8763c61d7ceacde416ad58635b367fb1d21fc0>

BOHLANDER, G. W.; SNELL, S. **Administração de Recursos Humanos**. São Paulo, Cengage Learning, 2016, p. 308.

BUCKINGHAM, M.; GOODALL, A. **Reinventing Performance Management.** Harvard Business Review, april 2015, p. 32.

BUTERIN, V. et al. **A next-generation smart contract and decentralized application platform.** White paper, 2014.

CABRAL, C.; CAPRINO, W. **Trilhas em Segurança da Informação: Caminhos e ideias para a proteção de dados.** Brasport, 2015.

CANAVAN, J. E. **Fundamentals of Network Secutiry.** Artech House, 2001.

CHOHAN, U. **The double spending problem and cryptocurrencies.** 2017.

DESSLER, G. **Administração de Recursos Humanos.** Rio de Janeiro, Pearson, 2012, p. 205.

ETHDOCS. **Contracts – Ethereum Homestead 0.1 documentation.** 2018. Disponível em: <http://www.ethdocs.org/en/latest/contracts-and-transactions/contracts.html#id4>

FOROUZAN, B. A.; MOSHARRAF, F. **Redes de Computadores: Uma Abordagem Top-Down.** São Paulo, AMGH, 2013.

GRAMIGNA, M. **Modelo de Competências e Gestão dos Talentos.** São Paulo, Pearson, 2012, p. 146.

KASIREDDY, P. **ELI5: What do we meann by “blockchains are trustless”?** 2018. Disponível em: <https://medium.com/@preethikasireddy/eli5-what-do-we-mean-by-blockchains-are-trustless-aa420635d5f6>

LEME, R.; VESPA, M. **Gestão do Desempenho Integrando Avaliação e Competências com o Balanced Scorecard.** São Paulo, Pearson, 2012, p. 50.

LIKERT, R. **A Technique for the Measurement of Attitudes.** Archives of Psychology. 1932.

MALHEIROS, B. T.; ROCHA, A. R. C. **Gestão de Pessoas – Avaliação e Gestão de Desempenho**. Rio de Janeiro, Nacional, 2014.

MATETI, P. **Hash Functions.** 2017. Disponível em: <http://cecs.wright.edu/~pmateti/Courses/3900/Lectures/Passwords/hash-functions.html>

NAKAMOTO, S. **Bitcoin: A peer-to-peer eletronic cash system.** 2008.

NIELSEN, J.; LANDAUER, T. K. **A mathematical model of the finding of usability problems.** 1993.

ORLICHI, J. I. **Programming languages intro.** 2018. Dispnível em: <https://github.com/ethereum/wiki/wiki/Programming-languages-intro>

RIBEIRO, A. **Gestão de Pessoas – 2ª edição.** São Paulo, Saraiva, 2013.

SOLIDITY. **Introduction to Smart Contracts.** 2018. Disponível em: <https://solidity.readthedocs.io/en/v0.4.21/introduction-to-smart-contracts.html>

SOLIDITY. **Security Considerations.** 2018. Disponível em: <https://solidity.readthedocs.io/en/v0.4.21/security-considerations.html>

TRUFFLE. **Ethereum Overview.** 2018. Disponível em: <https://truffleframework.com/tutorials/ethereum-overview>

UNE, M. **The security evaluation of time stamping schemes. The presente situation studies.** IMES discussion Papaers Series 2001-E-2018. 2001.