Universidade Federal de Itajubá - UNIFEI

SmartLock:
Controle de acesso através de Smart Contracts e Smart Property

UNIFEI
Itajubá
2019

Universidade Federal de Itajubá - UNIFEI

SmartLock:
Controle de acesso através de Smart Contracts e Smart Property

Maurício Xavier Zaparoli

Universidade Federal de Itajubá – UNIFEI

Programa de Pós-Graduação em Ciências e Tecnologia da Computação

Orientador: Prof. Dr. Adler Diniz de Souza

UNIFEI
Itajubá
2019

Universidade Federal de Itajubá - UNIFEI
SmartLock:
Controle de acesso através de Smart Contracts e Smart Property/ Universidade Federal de Itajubá - UNIFEI. – UNIFEI
Itajubá, 2019-
96p. : il. (algumas color.) ; 30 cm.

Orientador: Prof. Dr. Adler Diniz de Souza

Dissertação de Mestrado – Universidade Federal de Itajubá – UNIFEI
Programa de Pós-Graduação em Ciências e Tecnologia da Computação, 2019.

1. *blockchain*. 2. Ethereum. 3. contratos inteligentes. 4. *smart property*. 5. ISO/IEC 25010. 6. viabilidade. I. Prof. Dr. Adler Diniz de Souza. II. Universidade Federal de Itajubá.

Universidade Federal de Itajubá - UNIFEI

SmartLock: Controle de acesso através de Smart Contracts e Smart Property

Maurício Xavier Zaparoli

Trabalho aprovado. UNIFEI
Itajubá, 06 de agosto de 2019:

Prof. Dr. Adler Diniz de Souza
Orientador

Prof. Dr. Rafael de Magalhães Dias Frinhani
Convidado 1

Dr. Carlos Eduardo de Andrade
Convidado 2

UNIFEI
Itajubá
2019

Este trabalho é dedicado aos amigos e família por todo apoio, carinho e compreensão nos momentos difíceis dessa jornada.

Agradecimentos

Os agradecimentos principais são direcionados ao orientador, Prof. Dr. Adler Diniz de Souza, que deu grande apoio, incentivo e boas conexões que foram fundamentais na realização deste trabalho e à Kamila Rocha Ribeiro de Bragança pelo apoio, carinho e compreensão nos momentos difíceis dessa jornada.

Agradecimentos especiais são direcionados aos integrantes da equipe Byron e do laborátorio da Prof.ª Melise Maria Veiga de Paula na participação dos testes do protótipo desenvolvido, ao aluno de graduação Cristiano Costa pela ajuda no desenvolvimento do circuito eletrônico utilizado nesse trabalho, aos servidores Nedson e Anderson pelo suporte para consolidar o circuito impresso e pela permissão do uso de equipamentos da UNIFEI.

O presente trabalho foi realizado com apoio da Coordenação de Aperfeiçõamento de Pessoal de Nível Superior - Brasil (CAPES) - Código de Financiamento 001.

"Não vos amoldeis às estruturas deste mundo,
mas transformai-vos pela renovação da mente,
a fim de distinguir qual é a vontade de Deus:
o que é bom, o que Lhe é agradável, o que é perfeito.
Carta de São Paulo aos Romanos, Capítulo 12, Versículo 2

Resumo

Blockchain é a tecnologia que permite a transação de dinheiro entre partes em uma forma *peer-to-peer* sem a necessidade de um intermediador de confiança como os bancos. Esse sistema é conhecido por suas atratentes características: (i) integridade dos dados e (ii) segurança. Para tomar vantagem dessas características, existem redes *blockchain* onde o foco não está em sua criptomoeda. Uma delas é a rede Ethereum, que é uma plataforma para aplicações chamadas contratos inteligentes que firmam acordo entre partes mas o fazem de maneira descentralizada. São algoritmos que são implantados no sistema e podem ser acessadas globalmente. Essas aplicações são executados sem a possibilidade de censura, fraude ou intervenção externa de terceiros. Uma de suas possíveis aplicações é o conceito *smart property* que é transacionar a posse ou controle de propriedades nos moldes da *blockchain*. Isso pode ser aplicado no contexto de alugueis de casas de veraneio e como o uso de serviços similares ao AirBnB tem se tornado cada vez mais popular devido sua atratividade financeira para os usuários há a necessidade de se desenvolver um sistema capaz de solucionar problemas de segurança ao mesmo tempo que traz conforto. Essa dissertação detalha o desenvolvimento de um projeto que estuda a viabilidade da aplicação do conceito de *smart property*. Foi desenvolvido uma plataforma de reservas onde o usuário faz sua reserva que quando aprovada é implantado um contrato inteligente na plataforma Ethereum. Para acessar a propriedade o usuário usa uma aplicação desenvolvida para o sistema Android que transfere as credenciais usando um protocolo de transferência de dados por som. A fechadura verifica as credenciais acessando o contrato inteligente e aciona o circuito da fechadura liberando o acesso a propriedade. Para verificar a viabilidade da solução, foram realizados testes com supostos usuários do sistema que responderam um questionário. Após análise do questionário, constatou-se que o projeto é viável, mais atrativo que os modelos de contrato tradicionais, funcional e proporcionou conforto.

Palavras-chave: *Blockchain*; Ethereum; Contratos inteligentes; *Smart property*; ISO/IEC 25010; Viabilidade.

Abstract

Blockchain is the technology that allows the transaction of money between parties in a peer-to-peer manner without the need of a trusted intermediary such as banks. This technology is known for its attractive characteristics: (i) the data integrity, and (ii) security. To take advantage of such characteristics, some blockchain networks are not focused on their cryptocurrency. One of those networks is the Ethereum network, which is a platform for smart contracts. Smart contracts are applications that represent an agreement between parties in a decentralized manner. These algorithms are deployed on the system and can be globally accessed. They are executed without the possibility of censorship, fraud, or external third party intervention. One of their possible applications is the smart property concept that is transacting property on the blockchain. This can be applied on the property rental context and the raising use of services such as AirBnB due to its financial attractiveness to its users, there is a need to develop a system capable of solving the security issues while providing comfort. This work will presents the development of a project that studies the viability of applying the concept of smart property. A reservation platform was developed where the user makes one reservation and, when approved, a smart contract is deployed to the Ethereum platform. To access the property, the user uses the application developed for Android which transfer the credentials using a data through sound protocol. The lock verifies these credentials accessing the smart contract and activates the lock's circuit allowing access to the property. To examine the viability of this project, tests were performed with supposed users of this system who also answered a survey. After the analysis of this survey, the project proved to be viable, more attractive than the traditional standard, functional, and provided comfort.

Keywords: Blockchain; Ethereum; Smart contracts; Smart property; ISO/IEC 25010; Viability.

Lista de ilustrações

Lista de tabelas

Lista de abreviaturas e siglas

PoW	*Proof-of-Work*
PoS	*Proof-of-Stake*
PoI	*Proof-of-Importance*
dPoS	*Delegated Proof-of-Stake*
SCP	*Stellar Consensus Protocol*
EVM	*Ethereum Virtual Machine*
NFC	*Near-field Communication*
MVP	*Minimum Viable Product*
CRUD	*Create-Read-Update-Delete*
MVC	*Model-View-Controller*
IDE	*Integrated Development Environment*
API	*Application Programming Interface*
SDK	*Software Development Kit*
ABI	*Application Binary Interface*
GPIO	*general-purpose input/output*
BJT	*Bipolar Junction Transistor*
EC2	*Elastic Compute Cloud*
RDS	*Relational Database Service*
ORM	*Object-Relational Mapping*
XML	*eXtensible Markup Language*
JSON	*JavaScript Object Notation*
HTTP	*Hypertext Transfer Protocol*
MitM	*Man-in-the-Middle*

SSL	*Secure Sockets Layer*
TLS	*Transport Layer Security*
SSH	*Secure Shell*

Sumário

1 Introdução

1.1 Contexto

Viajar ou alugar uma propriedade através de serviços como o AirBnB ou outros serviços similares podem trazer diversos incômodos para as partes envolvidas. O processo de troca de chaves entre hóspede e proprietário é problemático pois geralmente eles precisam se encontrar para isso. O horário para o encontro pode ser problemático. O proprietário pode estar no trabalho ou a chegada do hóspede na cidade pode ser durante a madrugada. O proprietário deve estar presente na cidade da propriedade alugada ou depender de outra pessoa de confiança para entrega das chaves. Além disso o hóspede ainda pode perder as chaves ou não devolvê-las em casos mais graves[1,2].

Os problemas apresentados anteriormente trazem riscos tanto para os futuros hóspedes quanto para o proprietário. Ambos podem ter seus bens furtados ou até mesmo se expor a riscos, principalmente em áreas de grande atividade turística (XU; PENNINGTON-GRAY et al., 2017). Existem diversos relatos de hóspedes que foram abordados ou assaltados dentro da propriedade alugada[3,4], ou ainda de proprietários que tiveram os seus bens roubados por hóspedes[5] que tiraram cópias indevidas das chaves do imóvel alugado. Outro problema é a necessidade de controlar a entrada de funcionários que prestam serviços, limpeza por exemplo, na propriedade. Eventualmente, esses funcionários podem estar mal intencionados e furtar os hóspedes[6,7,8]. Nesses casos específicos, ter o controle de quem entrou, quanto tempo ficou e quando saiu é essencial para minimizar os riscos apresentados.

---

[1] *Dealing with Airbnb for a Lost Key is Ridiculous*: <https://www.airbnbhell.com/dealing-airbnb-lost-key-ridiculous/> Acesso em: 13 de ago. de 2019

[2] *Elderly Airbnb Guest in Germany Kicks Cat, Steals Keys*: <https://www.airbnbhell.com/elderly-airbnb-guest-germany-kicks-cat-steals-keys/> Acesso em: 13 de ago. de 2019

[3] *Short-Term Rental Guests Robbed, Pistol Whipped in Hollywood*: <https://www.nbclosangeles.com/news/local/AirBnB-guests-robbed-beaten-in-Hollywood-487738771.html> Acesso em: 13 de ago. de 2019

[4] *British tourists flee New Orleans after Airbnb home invasion*: <https://www.fox8live.com/story/38355584/british-tourists-flee-new-orleans-after-airbnb-home-invasion/> Acesso em: 13 de ago. de 2019

[5] *EXCLUSIVE: Discovery Bay Airbnb host says her home was brutally ransacked by renters*: <https://news.yahoo.com/exclusive-discovery-bay-airbnb-host-234541811.html> Acesso em: 13 de ago. de 2019

[6] *Clothes Stolen by Host, Airbnb Does Nothing*: <https://www.airbnbhell.com/clothes-stolen-by-host-airbnb-does-nothing/> Acesso em: 13 de ago. de 2019

[7] *Dishonest Host Refuses to Admit Shoes have been Stolen*: <https://www.airbnbhell.com/dishonest-host-refuses-to-admit-shoes-have-been-stolen/> Acesso em: 13 de ago. de 2019

[8] *Guests Robbed in Salò Airbnb, Host Possessions Untouched*: <https://www.airbnbhell.com/guests-robbed-salo-airbnb-host-possessions-untouched/> Acesso em: 13 de ago. de 2019

Portanto, o problema de pesquisa tratado nesse artigo é: Como garantir segurança para o locador e o locatário no acesso a ambientes físicos compartilhados?

Diversas soluções eletrônicas e computacionais já estão disponíveis para tratar os problemas apresentados anteriormente como a *August Smart Lock Pro*[9] e a *Ultraloq UL3*[10]. No entanto grande parte delas podem ser facilmente burladas ou hackeadas por especialistas em computação[11,12]. Sendo assim, há a necessidade de se buscar uma solução que aumente a segurança desses tipos de sistemas de acesso.

Os últimos anos foram marcados pela popularização de sistemas baseados na tecnologia *blockchain*. *Blockchain* é uma tecnologia descentralizada de gerenciamento, base de qualquer criptomoeda, responsável pela emissão e transferência de dinheiro entre seus usuários (YLI-HUUMO et al., 2016). É o registro continuamente crescente das transações entre os usuários confirmadas pelos nós participantes de uma rede, disponível a todos e de posse e controle de ninguém (YLI-HUUMO et al., 2016; SWAN, 2015). A vantagem do blockchain é que esse registro não pode ser modificado nem deletado uma vez que os dados das transferências tenham sido aprovados por todos os nós participantes. Por isso o blockchain é conhecido por sua integridade dos dados e segurança, o que estende seu uso para outros serviços e aplicações (YLI-HUUMO et al., 2016) além da simples transferência de dinheiro entre usuários.

Assim, esse trabalho propõe o desenvolvimento de uma fechadura eletrônica controlada por contratos inteligentes registrados na plataforma *blockchain* da Ethereum. A solução abrange tanto *hardware* quanto *software* e é composta por três componentes: (i) uma fechadura eletrônica, (ii) um servidor hospedado na nuvem e (iii) aplicativo de celular. O hóspede se cadastra em uma plataforma de reservas, reserva a propriedade desejada e essa reserva quando aprovada pelo proprietário tem suas informações armazenadas em um contrato inteligente implantado na rede *blockchain* da Ethereum. Para acessar a propriedade, o hóspede utiliza a aplicação em seu celular para transmitir por som o código de credencial específico da reserva que é captado pela fechadura. A fechadura verifica as informações armazenadas no contrato inteligente e se forem válidas o acesso a propriedade é liberado. As seguintes características foram avaliadas: (i) comodidade e (ii) segurança tanto para o hóspede quanto para o proprietário trazidos ao utilizar *smartphones* para destravar fechaduras eletrônicas agregando a segurança proporcionada pela plataforma blockchain da Ethereum.

[9] August Smart Lock Pro | The Ultimate Smart Lock for Your Smart Home: <https://august.com/products/august-smart-lock-pro-connect> Acesso em: 21 de ago. de 2019.

[10] UL3 BT Bluetooth Enabled Fingerprint and Touchscreen Smart Lever Lock | U-Tec: <https://store.u-tec.com/pages/ul3-bt-bluetooth-enabled-fingerprint-and-touchscreen-smart-lever-lock> Acesso em: 21 de ago. de 2019.

[11] Here's what happened when someone hacked the August Smart Lock: <https://www.cnet.com/news/august-smart-lock-hacked/> Acesso em: 21 de ago. de 2019.

[12] The not so ultra lock | Pen Testers Partners: <https://www.pentestpartners.com/security-blog/the-not-so-ultra-lock/> Acesso em: 21 de ago. de 2019.

1.2 Questões de pesquisa e objetivos

O objetivo deste trabalho é desenvolver um protótipo de fechadura eletrônica controlada por aplicativos de celular e contratos inteligentes para acessar ambientes tornando-os *smart properties* e estudar a viabilidade de utilizá-lo.

Para tal, tem-se os seguintes objetivos específicos:

- Revisar a bibliografia sobre os temas: *blockchain*, contratos inteligentes e *smart property*;
- Pesquisar trabalhos correlatos;
- Definir a metodologia utilizada;
- Descrever o desenvolvimento do protótipo;
- Analisar a Qualidade de Software usando a norma ISO/IEC 25010.

E com isso responder as seguintes questões de pesquisa referentes a utilização de contratos inteligentes e o conceito de *smart property*:

- É viável a aplicação do conceito de *smart property* a propriedades físicas? Ou seja é possível controlar o acesso e a posse de uma propriedade usando contratos inteligentes?
- O protótipo desenvolvido atendeu aos requisitos na realização dessa tarefa?
- O uso da fechadura eletrônica desenvolvida é viável e atrativo para uma aplicação real?

1.3 Estrutura do trabalho

Esta dissertação está estruturada da seguinte forma: a Seção 2 contém a fundamentação teórica das tecnologias utilizadas e conceitos aplicados na pesquisa; a Seção 3 apresenta a metodologia de pesquisa incluindo: Revisão de literatura, Proposta de projeto e Implementação do projeto; a Seção 4 descreve a análise de vulnerabilidades do projeto desenvolvido, como elas podem ser utilizadas para explorar o sistema e propõe melhorias que podem ser implementadas para aprimorar sua segurança; a Seção 5 contém a Metodologia de Avaliação e os Resultados, onde se apresenta a metodologia utilizada para avaliar a aplicabilidade do projeto e por último a Seção 6 apresenta as conclusões e contribuições da pesquisa e trabalhos futuros.

2 Fundamentação teórica

2.1 *Blockchain*

Blockchain é um termo que vêm ganhando cada vez mais destaque desde a sua primeira aparição em 2008 com o Bitcoin (NAKAMOTO et al., 2008). Essa tecnologia permite o registro público, distribuído, criptografado e inalterável das transações de qualquer criptomoeda (BECK et al., 2017), e usa a técnica *proof-of-work* para manter o sistema em sincronia (BURNISKE; TATAR, 2017).

É público pois qualquer computador pode acessar a rede *blockchain* (BURNISKE; TATAR, 2017; TAPSCOTT; TAPSCOTT, 2016). É descentralizado pois todos os nós participantes da rede, os mineradores, possuem uma cópia local deste registro (YLI-HUUMO et al., 2016; SWAN, 2015; BURNISKE; TATAR, 2017; TAPSCOTT; TAPSCOTT, 2016). Não necessita de uma entidade intermediadora de confiança para gerenciá-lo (YLI-HUUMO et al., 2016; SWAN, 2015; BURNISKE; TATAR, 2017; TAPSCOTT; TAPSCOTT, 2016) o que permite que usuários transfiram dinheiro diretamente entre si. Então não há um banco de dados central com riscos de invasão e comprometimento da integridade dos dados (TAPSCOTT; TAPSCOTT, 2016).

É criptografado por dois motivos. Primeiramente, as transações devem ser verificadas para garantir que o usuário que envia uma criptomoeda é dono dela usando criptografia de chave pública (CRUZ, 2018; BURNISKE; TATAR, 2017). E segundo, na forma em que as transações são adicionadas a blocos que são encadeados pelo processo *proof-of-work* (YLI-HUUMO et al., 2016; BURNISKE; TATAR, 2017).

Para o usuário, os elementos importantes para transferir criptomoedas são: (i) um endereço de carteira de criptomoedas, (ii) uma chave-privada e (iii) o programa da carteira. O endereço de carteira é utilizado para enviar e receber criptomoedas, a chave-privada é usada para verificação de posse da cada unidade de criptomoeda e o programa de de carteira é utilizado para gerenciar criptomoedas (SWAN, 2015).

Toda transação possui um identificador único (*transaction hash*) que é um resumo dos seus dados obtido pelo uso da uma função de *hash*, que é um algoritmo que mapeia dados de comprimento variável para dados de comprimento fixo, do tipo SHA256 duas vezes[1].

De acordo com Antonopoulos (2014), de forma simplista as transações notificam a rede que o proprietario de uma quantidade de criptomoedas autorizou a transação de

[1] TXID: <https://learnmeabitcoin.com/glossary/txid> Acesso em: 27 de ago. de 2019.

algumas delas para outro usuário. O novo proprietario pode então gastá-las fazendo outra transação. Em uma abordagem mais didática, as transações em uma rede *blockchain* possuem entradas e saídas. As entradas são débitos de unidades de criptomoedas de algumas contas e as saídas são créditos em outras contas. As saídas das transações podem ser utilizadas como entradas em outras transações.

As transações também contém prova de posse dos montantes de criptomoedas na forma de uma assinatura digital do proprietário com a sua chave-privada. Para "gastar" criptomoedas, o proprietário assina a transação com a sua chave-privada (ANTONOPOULOS, 2014).

A Figura 1 ilustra o funcionamento das transações. Cada retângulo representa uma transação na rede *blockchain* Bitcoin. Considerando a segunda, Alice faz uma transferência de uma quantia de 0,0150 unidades de *bitcoin* para Bob. Como entrada dessa transação, tem-se os créditos da transação anterior que Alice recebeu de Joe que foram 0,1000 unidades de *bitcoin* e como saídas temos créditos de 0,0150 unidades de *bitcoin* que são associados ao endereço da carteira de criptomoedas de Bob por meio da sua chave-privada, 0,0845 unidades de *bitcoin* para Alice e 0,0005 unidade de *bitcoin* como tarifa da transação. Como os créditos da primeira transação estão atrelados ao endereço da carteira de criptomoedas de Alice por sua chave-privada, ela pôde usa-los na segunda transação assinando essa transação. A tarifa da transação é uma quantia que é paga aos nós mineradores pelos custos da mineração das transação.

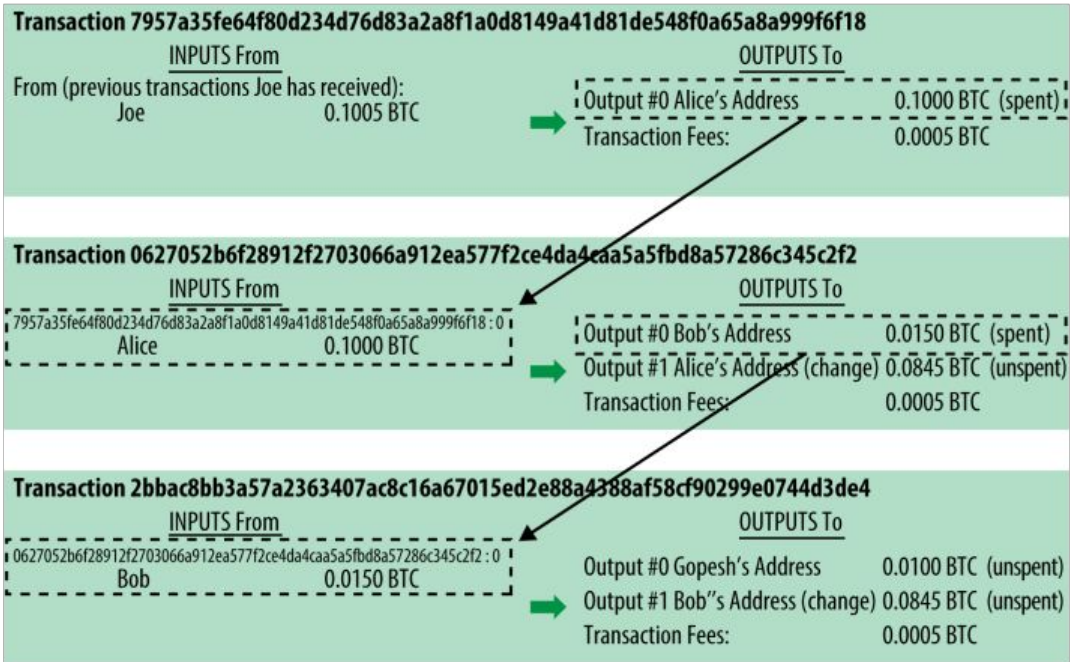

Figura 1 – Cadeia de transações.

Fonte: *Mastering Bitcoin: Unlocking Digital Cryptocurrencies* (ANTONOPOULOS, 2014)

A tecnologia *blockchain* resolve o problema do gasto duplo ou do inglês *double spending* onde uma das partes tenta gastar seus *tokens* de criptomoedas múltiplas vezes. Para isso, a transação mais antiga com um *token* específico é a válida e tentativas de gasto duplo da mesma são ignoradas. A única maneira de se obter esse resultado é tomando conhecimento de todas as transações, ou seja, todas elas devem ser publicamente anunciadas (NAKAMOTO et al., 2008). Como os nós participantes se sincronizam pelo processo de *proof-of-work* há um acordo em relação a ordem que elas acontecem.

Proof-of-work é o processo no qual os mineradores competem entre si para resolver um problema criptográfico em troca de pagamento em criptomoeda. Esse processo, que exige um grande poder computacional, envolve a combinação de quatro variáveis: horário e data (*timestamp*), uma assinatura (*hash*) das transações, a identidade do bloco anterior e um *nonce* (BURNISKE; TATAR, 2017). *Nonce* é um número aleatório que ao ser combinado com as outros três números deve produzir uma *hash* com um número específico de dígitos 0 no início, ou seja, uma sequência de caracteres menor que um alvo ou *target*. Esse *target* dita a dificuldade do processo. De acordo com Antonopoulos (2014), até o momento de publicação de seu livro o alvo era `000000000000004c296e6376db3a241271f43fd3f5de7ba18986e517a243baa7` e com ele levava-se em média 10 minutos para se encontrar o *nonce* capaz de produzir uma *hash*. Cada vez que se aumenta o número de *bits* iguais a zero no alvo, o espaço de possibilidades da *hash* diminui pela metade (ANTONOPOULOS, 2014).

Por meio desse mecanismo, os nós participantes entram em consenso sobre como o novo bloco será encadeado (NOFER et al., 2017). Como informações dos blocos anteriores são usadas, nenhum deles pode ser alterado sem alterar informações dos seguintes (YLI-HUUMO et al., 2016; BURNISKE; TATAR, 2017). Esta estratégia dificulta muito fraudes no sistema pois é considerada como válida a cadeia de blocos de maior extensão, ou seja, aquela que tem maior poder computacional investido nela (NAKAMOTO et al., 2008).

Se a maioria desse poder de processamento é controlada por nós honestos, a cadeia de blocos honesta vai aumentar mais rapidamente que as desonestas. Para efetivamente burlar esse sistema um nó ou conjunto deles teria que possuir poder computacional maior que o conjunto de nós honestos (NAKAMOTO et al., 2008).

Também existem outros processos para consenso entre os nós além do *Proof-of-Work* (PoW) mencionado. Dentre eles tem-se *Proof-of-Stake* (PoS), o qual a idade da moeda é levada em consideração na forma de "dias-moeda" ou do inglês *"coin-days"*. Possuir 10 moedas por 10 dias se iguala a 100 "dias-moeda". Ao gastar essas moedas, a idade é "consumida" e ajustada para zero. Diferentemente do sistema PoW, o qual a cadeia com mais trabalho computacional investido é vista como a principal, no PoS usa-se a cadeia com maior consumo de idade da moeda. Outros existentes são *Proof-of-Importance* (PoI),

Delegated Proof-of-Stake (dPoS), *Stellar Consensus Protocol* (SCP) e *Cardano's Ouroboros* (BACH; MIHALJEVIC; ZAGAR, 2018).

Devido a essas características, a tecnologia *blockchain* possui pontos fortes que são muito atrativos: (i) a integridade dos dados e (ii) a segurança que tornam a *blockchain* atrativa para outros serviços e aplicações além da transação de dinheiro (YLI-HUUMO et al., 2016).

Existem *blockchains* alternativas ao Bitcoin cujo foco não está na moeda. Elas implementam uma plataforma para contratos inteligentes, registros e outras aplicações (ANTONOPOULOS, 2014). Podem ser aplicados para registros públicos e privados, identidade digital, registro de ativos físicos e intangíveis (SWAN, 2015). Por exemplo, para proteger uma idéia, ao invés de patenteada ela pode ser codificada na *blockchain* servindo como meio de prova no futuro.

Zhao et. al (ZHAO; FAN; YAN, 2016) apontam outras aplicações da tecnologia *blockchain*, entre elas: (i) um sistema para proteção de dados pessoais, (ii) um para votação mais transparente e (iii) outro para rastreamento da origem de produtos na cadeia de suprimentos. Christidis e Devetsikiotis (2016) (CHRISTIDIS; DEVETSIKIOTIS, 2016) concluíram que a combinação de *blockchain* com IoT tem grande potencial e pode causar transformações em certas indústrias, pavimentando novos modelos de negócios e aplicações distribuídas inovadoras.

A funcionalidade do *blockchain* pode revolucionar vários campos como finanças, contabilidade, gerência, direito, política, governo e muitos mais (SWAN, 2015; ANTONOPOULOS, 2014; ZHAO; FAN; YAN, 2016).

Uma aplicação atual é a realizada pela rede de supermercados Walmart que está usando a tecnologia para tornar mais transparente a cadeia de suprimentos dos seus produtos[2]. Com essa solução é possível rastrear a origem dos alimentos para assim ter maior controle de qualidade além de maior transparência com seus consumidores.

2.2 Ethereum

Considerando a existência de outras criptomoedas, com propriedades mais dinâmicas que o Bitcoin, nesse trabalho utiliza o *token* da Ethereum[3]. Essa rede foi escolhida pelas seguintes razões:

- É a segunda maior rede *blockchain* e a que mais cresce (TAPSCOTT; TAPSCOTT, 2016).

[2] Walmart Case Study - Hyperledger: <https://www.hyperledger.org/resources/publications/walmart-case-study> Acesso em: 13 de ago. de 2019

[3] Home | Ethereum: <https://www.ethereum.org/> Acesso em: 13 de ago. de 2019

- Cada nó executa a Ethereum Virtual Machine (EVM), onde aplicações, chamadas de contratos inteligentes, são implantados e podem ser acessados mundialmente (BURNISKE; TATAR, 2017).

- Essas aplicações são executadas exatamente como foram programadas, sem censura, fraude ou interferência de terceiros (TAPSCOTT; TAPSCOTT, 2016).Uma das possíveis aplicações desses contratos permitiria a transferência eletrônica de bens, de maneira automática através da sua execução.

- A Ethereum tem sua própria criptomoeda chamada *ether* que é necessária para pagar a implantação e execução desses contratos (ANTONOPOULOS, 2014). Esses contratos são desenvolvidos na linguagem de programação Solidity[4] (DANNEN, 2017). A plataforma possui uma grande comunidade de desenvolvedores trabalhando no desenvolvimento de aplicações descentralizadas e na escalabilidade do sistema (D'ALIESSI, 2018).

- O método de consenso PoW que foi utilizado na Ethereum 1.0 pela rede de nós é o algoritmo Ethash. É a última versão do algoritmo chamado Dagger-Hashimoto, introduzida por Buterin e Dryja, com modificações (WOOD et al., 2014). Para a Ethereum 2.0, conhecida como "*Serenity*", o mecanismo de consenso entre os nós é chamado *Casper* e é um algoritmo PoW/PoS híbrido (BUTERIN; GRIFFITH, 2017).

- A Ethereum ainda possui o grupo Enterprise Ethereum Alliance que tem como objetivo especificar, promover e apoiar as melhores práticas, padrões e arquiteturas, referências de tecnologias baseadas na rede Ethereum, contando com o apoio de grandes empresas como Intel, Microsoft entre outras (POPPER, 2017).

Na rede Ethereum, existem dois tipos de contas: (i) conta de posse externa (*externally owned account*) e (ii) conta de contrato. A conta de posse externa é aquele de posse dos usuários qua a utilizam para fazer transações com outros usuários ou interagir com contratos. Essas contas possuem uma chave-privada que é usada para controlar aos fundos ou a contratos inteligentes. A conta de contrato é aquela que possui código de contrato e não possui chave-privada. Esse tipo de conta é controlado pelo seu próprio código (ANTONOPOULOS; WOOD, 2018).

Ambas possuem endereço na rede. Mas quando uma transação tem o endereço de um contrato como destino, ativa-se a sua execução na EVM usando os dados recebidos como entrada. Essas dados especificam que função no contrato deve ser executada e que parâmetros devem ser passados a ela (ANTONOPOULOS; WOOD, 2018).

[4] *Solidity - Solidity 0.5.9 documentation:* <https://solidity.readthedocs.io/en/v0.5.9/> Acesso em: 13 de ago. de 2019

Na Figura 2, é ilustrado a arquitetura simplificada da Ethereum com os usuários e dispositivos eletrônicos, como IoT (*Internet of Things*), interagindo com funções rede e com contratos inteligentes. A execução de um *script* na controladora faz uma chamada RPC (*Remote Procedure Call*) usando, por exemplo, a bliblioteca web3.py e interage com as funções do contrato inteligente que pode ter sido implantado na rede principal, de teste ou até em uma rede privada da Ethereum executando-o na EVM. O usuário interagindo com uma aplicação *web* pode ativar a execução de *scripts* que usam a biblioteca web3.js que geram transações para interagir com outros usuários ou contratos inteligentes e que também se comunicam com a rede através de chamadas RPC.

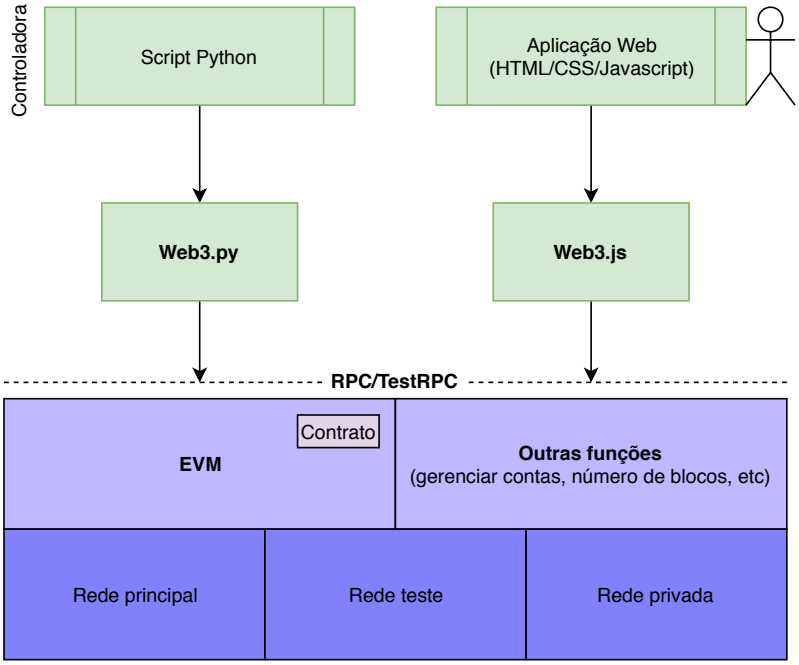

Figura 2 – Infraestrutura da rede Ethereum.

Fonte: Adaptado de steemit.com

2.3 Contrato Inteligente

Na sua forma tradicional, um contrato é um acordo entre duas pessoas ou organizações ou um documento legal que explica os detalhes desse acordo. Cada uma das partes adquire direitos e deveres relativos aos da outra parte (BOLTON; DEWATRIPONT et al., 2005).

Contratos inteligentes também firmam acordos entre duas ou mais partes da mesma forma que contratos tradicionais mas o fazem de forma automatizada, descentralizada, impondo a execução das cláusulas sem a possibilidade de fraude ou intervenção de terceiros (SWAN, 2015; TAPSCOTT; TAPSCOTT, 2016; CHRISTIDIS; DEVETSIKIOTIS, 2016). Tudo isso eliminando a necessidade das partes confiarem entre si (SWAN, 2015).

De acordo com (SZABO, 1996), os objetivos dos contratos inteligentes são satisfazer as condições contratuais, minimizar exceções acidentais ou maliciosas, eliminando a necessidade de intermediários de confiança, minimizando assim os custos. Fez-se uma revisão sistemática de literatura para identificar os principais trabalhos relacionadas à contratos inteligentes desde sua idealização mas esse conceito teve sua primeira aplicação real na plataforma *blockchain* da Ethereum que foi lançada em Julho de 2015[5].

Essencialmente o contrato inteligente é um algoritmo que é executado em todos os nós participantes de uma plataforma *blockchain* (SWAN, 2015; DANNEN, 2017; SZABO, 1996). São capazes de armazenar dados, receber e realizar pagamentos com criptomoeda assim como armazená-la (SWAN, 2015).

Como contratos inteligentes residem na *blockchain*, eles possuem um endereço único (PEE et al., 2019). Para acionar a execução de suas funções deve-se realizar uma transação para esse endereço. Ele então é executado independentemente e automaticamente de acordo com os dados passados pela transação (PEE et al., 2019; HILDENBRANDT et al., 2018).

2.4 *Smart Property*

Smart property é um termo introduzido por Nick Szabo (1996), que o define da seguinte forma: "É estender o conceito de contratos inteligentes à propriedades. *Smart Property* pode ser criada embutindo contratos inteligentes em objetos físicos".

O conceito geral é transacionar qualquer propriedade em modelos baseados em *blockchain* (SWAN, 2015; ZHANG; WEN, 2017). Uma propriedade codificada em *blockchain* pode ter sua posse ou acesso controlados por contratos inteligentes sujeitos a leis existentes (SWAN, 2015; NAKAMOTO et al., 2008; BUTERIN et al., 2014), de maneira eficiente, automática e descentralizada (BUTERIN et al., 2014). Isso aplica-se para qualquer tipo de ativo: físico (e.g.casa, veículos, bicicletas) ou intangível (e.g.idéias, votos, informações sobre saúde) (SWAN, 2015). As vantagens são a minimização de fraude e custos intermediários podendo também completar transações que não aconteceriam em cenários de baixa confiança (ZHANG; WEN, 2017).

Para exemplificar esse conceito, imagine um carro financiado que é controlado por contratos inteligentes. O controle do veículo pode ser retornado ao banco em caso de falta de compromisso com pagamentos de parcelas do financiamento. Assim como os carros, isso pode tornar acordos de aluguéis de apartamentos ou casas mais seguros (ACHESON, 2018).

[5] Ethereum Announces Official Launch Date | Cointelegraph: <https://cointelegraph.com/news/ethereum-announces-official-launch-date> Acesso em: 26 de ago. de 2019.

3 Metodologia

3.1 Proposta

3.1.1 Proposta de Aplicação e Modelo BPMN

A proposta deste trabalho é desenvolver uma fechadura eletrônica controlada por contratos inteligentes que ficam registrados na *blockchain* da *Ethereum*. O objetivo é proporcionar comodidade e segurança para os usuários de serviços como AirBnB ou até do setor de hotelaria.

Na Figura 4 é representado o processo de registro e aprovação das reservas no sistema. O hóspede deve fazer uma reserva do espaço físico que deseja alugar através de uma aplicação web. Feito isso, o servidor da aplicação web lança no seu banco de dados as informações referentes a reserva e com a aprovação do proprietário é implantado o contrato inteligente na rede *blockchain* da Ethereum através de uma transação feita pela sua carteira. Caso a reserva seja reprovada, o seu *status* passa a ser "Cancelada".

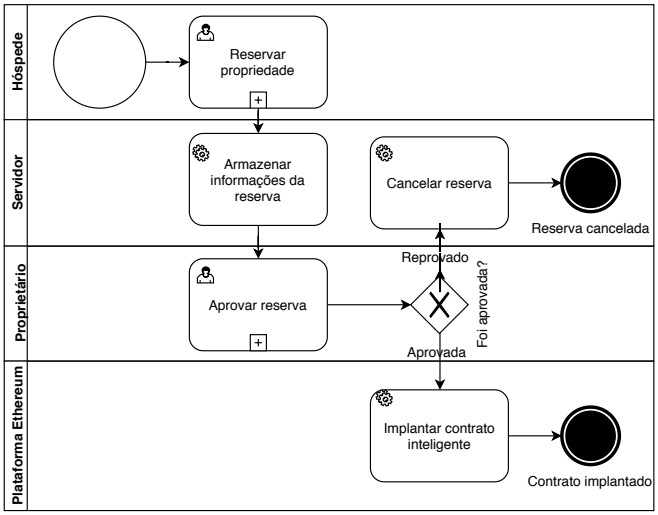

Figura 3 – Modelo BPMN do registro e aprovação das reservas.

Na Figura 3 é representado o processo de acesso da propriedade. Uma plataforma de prototipação (Raspberry Pi) controla cada uma das fechaduras, consulta o contrato inteligente para validar as credenciais do hóspede que são apresentadas utilizando um aplicativo em seu smartphone. Com isso o dono da propriedade garante que as suas chaves não serão copiadas, o hóspede garante que ele será o único a ter acesso a propriedade durante o tempo de sua reserva e elimina a necessidade de check-in ou, no caso de serviços como AirBnB, e do hóspede e proprietário se encontrarem para a entrega de chaves.

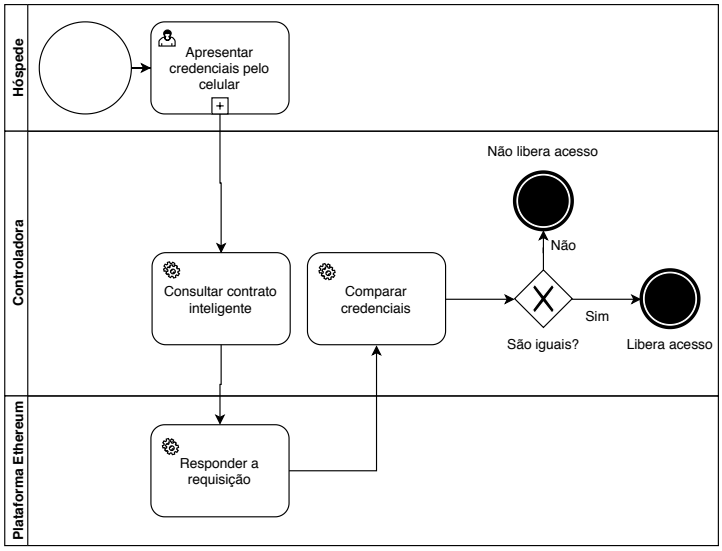

Figura 4 – Modelo BPMN do acesso à propriedade.

3.1.2 Prototipação

Para o protótipo desse projeto foram implementados um aplicativo nativo Android, um circuito para controle de uma fechadura solenóide, um script Python e um contrato inteligente que foi registrado manualmente na plataforma da Ethereum. O funcionamento do protótipo está ilustrado na Figura 5. O aplicativo, por meio de um protocolo de transferência de dados por som, chamado Chirp.io[1], transmite o código de uma credencial de teste fixada no código que é captado pelo microfone conectado ao controlador implementado usando a plataforma Raspberry Pi[2]. Escolheu-se trabalhar com este protocolo pois usar *Bluetooth* torna necessário o pareamento dos dispositivos, o que tornaria a interação com a fechadura um processo mais burocrático, e usar a tecnologia NFC (*Near-field communication*) permite atingir um número limitado de usuários dado que essa tecnologia não esta presente na maioria dos celulares, com uma espectativa de 2.2 bilhões de dispositivos com NFC em circulação no ano de 2020[3] e uma previsão de 6 bilhões de celulares em circulação para o mesmo ano[4]. Um script Python decodifica essa credencial e consulta um contrato inteligente para verificar se essa credencial está registrada. Após a verificação da credencial o circuito de controle da fechadura solenóide é acionado pelo controlador destravando a porta.

[1] Chirp | *Send data with sound*: <https://chirp.io/> Acesso em: 13 de ago. de 2019.
[2] Teach, Learn, and Make with Raspberry Pi - Raspberry Pi: <https://www.raspberrypi.org/> Acesso em: 13 de ago. de 2019
[3] *NFC-enabled Handset Shipments to Reach Three-Quarters of a Billion in 2015 - IHS Technology:* <https://technology.ihs.com/533599> Acesso em: 15 de ago. de 2019.
[4] 6 billion smartphones will be in circulation in 2020: Report: <https://www.statista.com/statistics/330695/number-of-smartphone-users-worldwide/> Acesso em: 15 de ago. de 2019

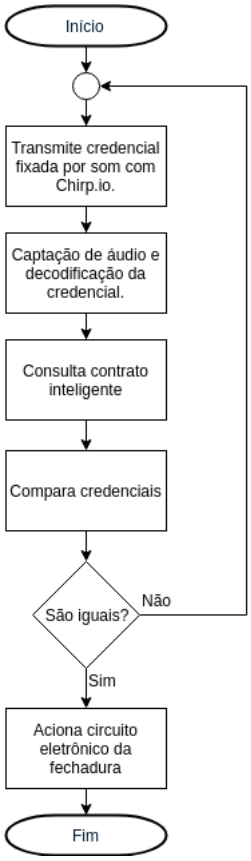

Figura 5 – Fluxograma do funcionamento do protótipo.

3.1.3 Validação Conceitual

Durante participação no Startup Weekend Maker 2018 Itajubá, evento de incentivo ao empreendedorismo e inovação, para validação do MVP (*Minimum Viable Product*) contatou-se a gerente de um hotel em Itajubá. Identificou-se a intenção da troca do sistema de fechaduras comuns para fechaduras eletrônicas. A motivação para isso veio de problemas internos para monitoramento de entrada e saída de funcionários dos quartos e de clientes que haviam reclamado de itens pessoais que haviam sumido. As necessidades do cliente evidenciaram a demanda de uma solução para atendê-las e ainda trazer outros benefícios como comodidade e automatização de processos.

3.1.4 Arquitetura

O site responsivo, plataforma para realizar reservas, foi desenvolvido usando o framework PHP Laravel[5] para acelerar o desenvolvimento dos CRUD's (*Create-Read-Update-Delete*) usando o modelo MVC (*Model-View-Controller*), Bootstrap para o *front-end* e MySQL para o banco de dados. A visão geral da arquitetura é mostrada na Figura 6.

[5] Framework PHP Laravel: <https://laravel.com/> Acesso em: 13 de ago. de 2019.

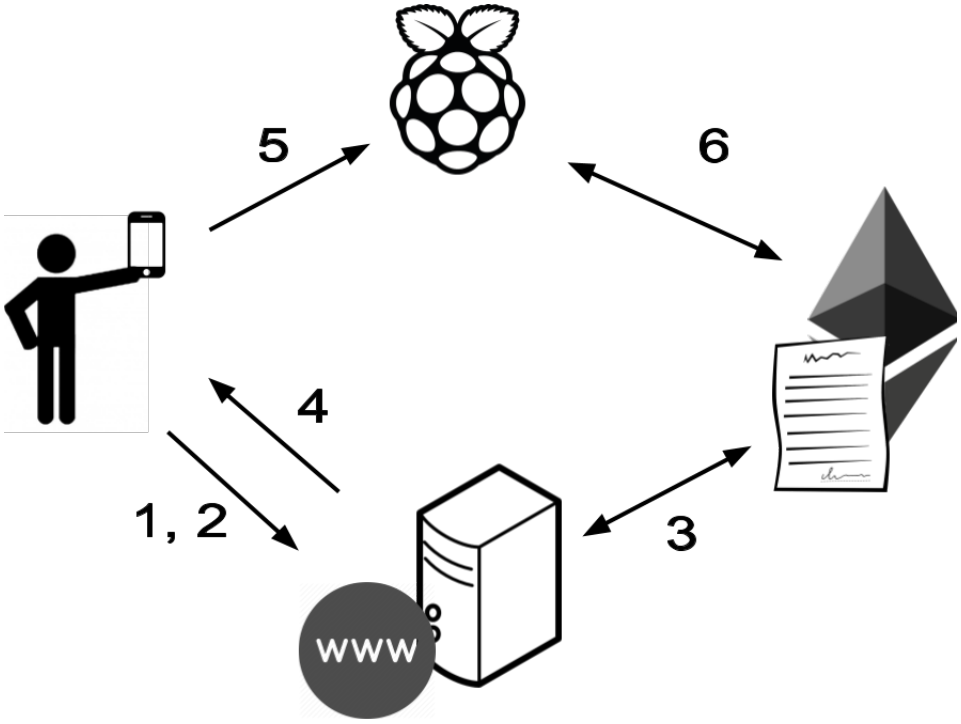

Figura 6 – Arquitetura da solução proposta.

1) Cadastro. A solução proposta neste trabalho começa com o hóspede se cadastrando no site do projeto com seus dados pessoais. Esse cadastro precisa ser ativado através do link que é enviado para o e-mail do usuário.

Após o cadastro do hóspede no site, o *login* também fica disponível no seu aplicativo de celular. Ele permite que o usuário apenas acesse suas reservas e transmita as informações necessárias para a fechadura validar a entrada no ambiente. As reservas devem ser feitas apenas pelo site. A aplicação de acesso às fechaduras foi desenvolvida nativamente em Android usando a IDE (*Integrated Development Environment*) *Android Studio* e utiliza um protocolo de transferência de dados por som chamado Chirp.io.

Para poder fazer reservas e interagir com as suas funções de cancelamento e edição o usuário necessita de uma carteira de criptomoedas da rede Ethereum. Esse trabalho utilizou a carteira Metamask[6] tanto na fase de desenvolvimento quanto na fase de testes. Essa carteira funciona como uma extensão de navegadores, disponível para Google Chrome e Mozilla Firefox. A carteira é necessária pois deve-se registrar o seu endereço no perfil do usuário. Essa informação é salva para depois determinar quem pode interagir com o contrato inteligente e alterar suas informações. Tanto o endereço da carteira do proprietário quanto do hóspede são usados no momento da implantação do contrato na rede *blockchain* com essa finalidade. Com essas informações registradas no contrato inteligente pode-se restringir o seu acesso.

[6] MetaMask: <https://metamask.io/>. Acesso em: 13 de ago. de 2019.

2) Reserva. No menu principal, que pode ser visualizado na Figura 7, são listados imóveis para o usuário. Como os testes do projeto foram feitos apenas na sala do professor Adler Diniz de Souza apenas o botão de reservar dela funcionava.

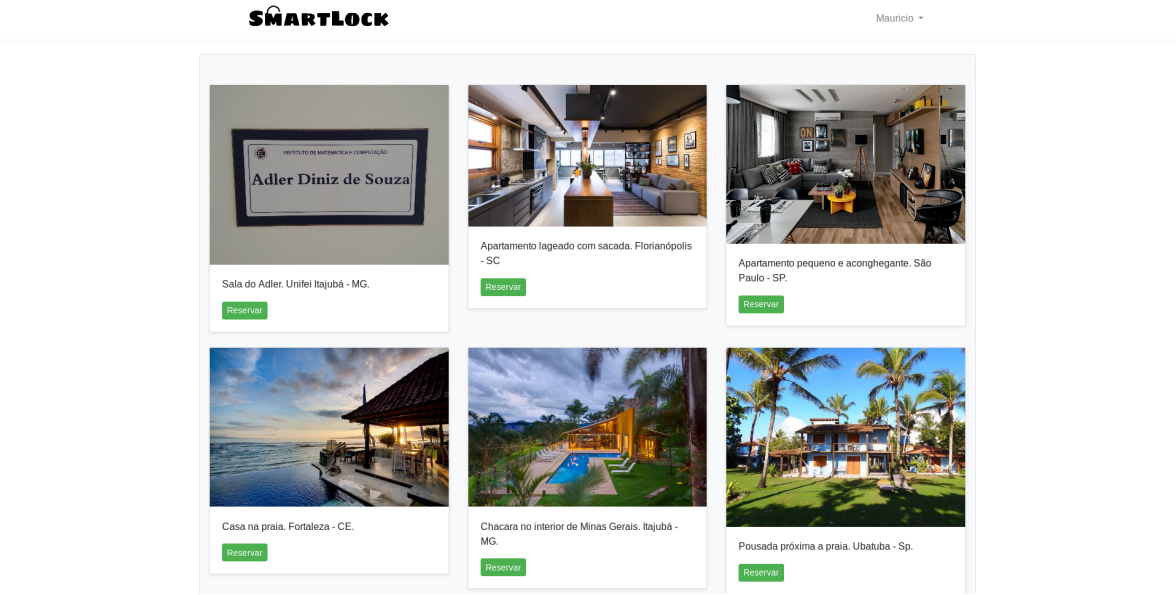

Figura 7 – Menu Principal.

Ao clicar o ambiente que deseja alugar o hóspede então é levado a outra página onde especifica o período da reserva. Como o modelo de negócio é baseado nos moldes do AirBnB e redes de hotéis, a diária vai de meio-dia a meio-dia. Então é necessário que as datas sejam diferentes para não haver problemas no momento de acessar o ambiente. Ao armazenar uma reserva gera-se a credencial que mais tarde será utilizada para implantar o contrato inteligente. Esta credencial é o valor *hash* da cadeia de caracteres formada pela concatenação da data inicial, data final, identificador do quarto e identificador do usuário. Esta *hash* é computada usando o algoritmo SHA256. A página para realizar reservas pode ser vista na Figura 8.

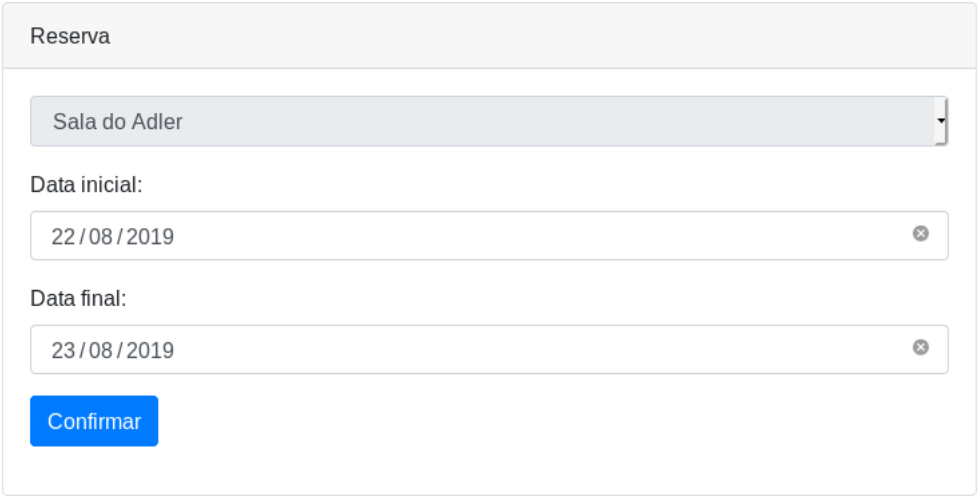

Figura 8 – Menu Reservar.

A reserva que acabou de ser feita permanece com o estado de pendente aguardando aprovação do administrador do sistema. Após suas confirmações as reservas podem ser visualizadas no menu de Reservas e estão divididas em três abas: Aprovadas, Pendentes e Canceladas. O menu para visualização de reservas pode ser visto na Figura 9.

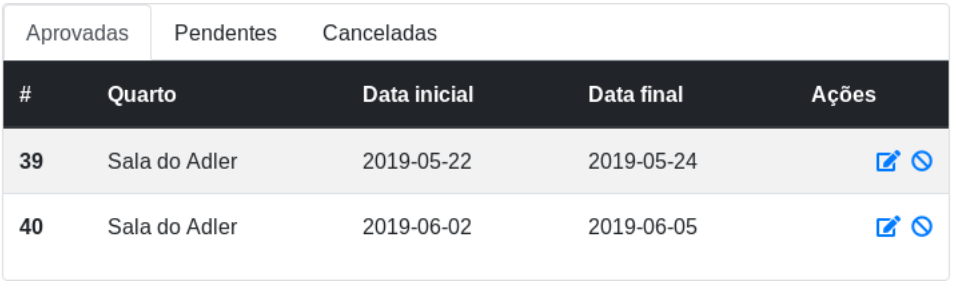

Figura 9 – Menu Reservas.

Quando cadastrada uma reserva é gerada uma senha que posteriormente será usada para acessar a propriedade. Essa senha é gerada usando uma função *hash*.

3) Implantação. Nesse ponto a reserva deve ser aprovada pelo administrador, que assume o papel do proprietário. Ao fazer o *login* e acessar o menu de gerenciamento de reservas, as reservas são listadas de maneira similar a Figura 9 com o diferencial que na aba Pendentes há a opção de aprovar reservas. Também são apresentadas informações de quem fez a reserva, as suas datas e identificador. O menu de gerenciamento de reservas pode ser visto na Figura 10.

Figura 10 – Menu Gerenciamento de Reservas.

Após a confirmação da reserva, o script desenvolvido usando a biblioteca web3.js[7], que é a API (*Application Programming Interface*) JavaScript da Ethereum, é responsável por implantar o contrato inteligente. O script utiliza a extensão MetaMask como intermediário de comunicação com a rede *blockchain*. Ele gera a transação necessária para a implantação do contrato inteligente que deve ser aprovada pelo administrador. A seguir a Figura 11 ilustra o transação que é criada.

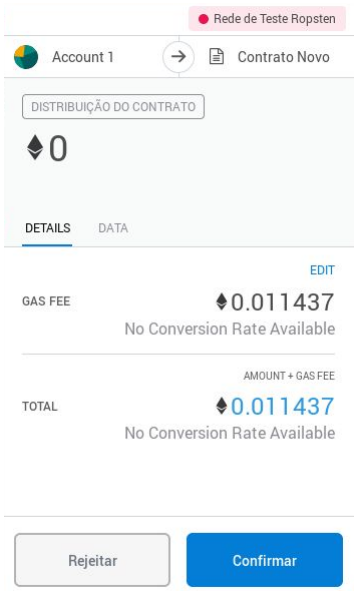

Figura 11 – Transação com a carteira MetaMask.

Na Figura 11, pode-se visualizar algumas informações relevantes. A primeira é "Contrato Novo", que indica que a transação tem o objetivo de criar um contrato inteligente novo. Abaixo da caixa com o texto "Distribuição do Contrato", temos o montante (*amount* em inglês) da transação que no caso é zero. Em seguida, temos "*GAS FEE*" que é a

[7] Ethereum Javascript API: <https://github.com/ethereum/web3.js/> Acesso em: 13 de ago. de 2019.

multiplicação dos valores de *Gas Price* e *Gas Limit* da transação. Por ultimo, temos "TOTAL", que é a soma dos valores de montante e de "*GAS FEE*".

4) Acesso às reservas. Acessando o aplicativo de celular, o usuário pode visualizar apenas suas reservas aprovadas, ou seja, aquelas com contrato inteligente implantados na *blockchain*. Ao tocar em uma delas o usuário transmite a credencial e o endereço do contrato inteligente, que são únicos para cada uma das reservas, usando o protocolo de Chirp.io.

5) Identificação. As informações citadas no item anterior são transmitidas através do auto-falante do celular, captados pelo microfone conectado ao controlador e decodificados pelo protocolo Chirp.io.

6) Consulta ao contrato. Com o endereço o contrato inteligente pode ser acessado, as credenciais e as datas da reserva podem ser verificadas. Após essa validação o circuito de acionamento da fechadura eletrônica é ativado e destrava a porta, caso haja uma reserva válida.

4 Implementação

4.1 Raspbery Pi

A tecnologia *Android*[1] foi escolhida para o aplicativo de *smartphone*, pois é amplamente difundido e utilizado no mundo. Em países de grande população como a Brasil, India e Rússia o sistema *Android* é dominante[2] e atinge 76,08% do mercado global[3]. Optou-se por trabalhar com a *Raspberry Pi* pois é um micro-computador muito versátil, simples de operar e permite implementação de vários tipos de projeto. Escolheu-se a linguagem de programação *Python*, pois tanto o protocolo Chirp.io quanto a biblioteca de comunicação *web3* possuem versão nessa linguagem, e sistema operacional *Raspbian Stretch Lite*[4], versão do sistema com visual minimalista apenas com a interface de comandos por questões de desempenho.

A comunicação do aplicativo de celular do usuário com a controladora foi feita utilizando um protocolo de transferência de dados através do som chamado Chirp.io. Este protocolo tem SDKs (*Software Development Kit*) em múltiplas plataformas, como Android e iOS[5]. Como uma *Raspberry Pi* foi utilizada para acionar o circuito, optou-se por usar a versão *Python* do protocolo. Este opera em vários modos dos quais foi escolhido o padrão (*Standard*) que permite a transferência de 32 bytes em 4,5 segundos[6].

A Figura 12 demonstra o procedimento para acessar uma propriedade. Através do aplicativo de celular, o usuário transmite um áudio com a senha referente a reserva e o endereço do contrato inteligente codificados usando o auto-falante do seu smartphone, a Raspberry Pi recebe o áudio através do microfone conectado a uma placa de som USB ligada a ela e por com o *script Python* acessa o contrato inteligente na rede *blockchain* da Ethereum para verificar a credencial. A Figura 13 mostra o microfone conectado a placa de som USB.

---

1 Android: <https://www.android.com/> Acesso em: 14 de ago. de 2019.
2 Android v iOS market share 2019 <https://deviceatlas.com/blog/android-v-ios-market-share> Acesso em: 13 de ago. de 2019
3 Mobile Operating System Market Share Worldwide | StatCounter Global Stats <https://gs.statcounter.com/os-market-share/mobile/worldwide> Acesso em: 14 de ago. de 2019.
4 Download Raspbian for Raspberry Pi: <https://www.raspberrypi.org/downloads/raspbian/> Acesso em: 14 de ago. de 2019.
5 Chirp | Developer friendly data over sound SDKs: <https://developers.chirp.io/> Acesso em: 14 de ago. de 2019.
6 Chirp | Data over sound SDKs: <https://developers.chirp.io/applications> Acesso em: 14 de ago. de 2019.

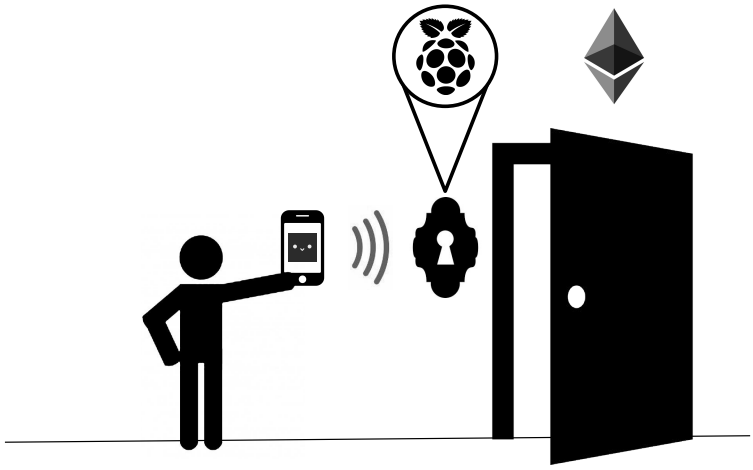

Figura 12 – Procedimento de acesso à propriedade.

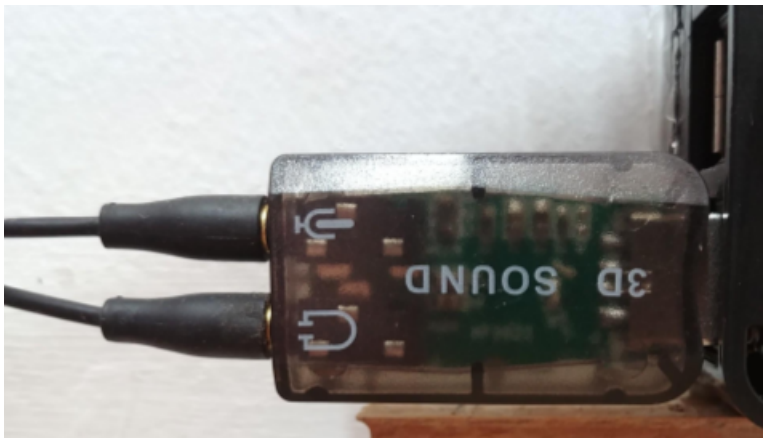

Figura 13 – Microfone conectado a placa de som USB.

A configuração do Chirp.io é feita criando-se um arquivo chamado ".chirprc"que deve ser criado no diretório "~/"do sistema Raspbian e ter o seguinte conteúdo:

```
[default]
app_key = ...
app_secret = ...
app_config = ...
```

Os valores das variáveis foram retirados para economia de espaço. As variáveis `app_key` e `app_secret` se referem a conta que é necessário ser criada na plataforma dos desenvolvedores e `app_config` configura o protocolo em si.

Na interação com o contrato inteligente registrado na *blockchain* da Ethereum, foi usada a biblioteca python *web3.py*. Sua API é derivada da API Javascript web3.js.

Para configurar essa comunicação basta definir o endereço do contrato inteligente na rede *blockchain*, um `HTTPProvider`, que é um adaptador de dados que capta as requisições JSON-RPC e retorna as respostas[7], que conecta a biblioteca *web3.py* com a API da Ethereum e a ABI (*Application Binary Interface*) do contrato inteligente. O `HTTPProvider` escolhido foi o disponibilizado pela INFURA[8] , que fornece acesso a rede *blockchain* da Ethereum sem a necessidade de instalar, configurar e manter uma infraestrutura Ethereum. A ABI define detalhes de como as funções do contrato são chamadas e em que formato binário a informação recebida deve ser interpretada. Após essa configuração é possível chamar as funções do contrato inteligente e conferir a credencial do usuário e datas da reserva. Parte do ABI pode ser visto a seguir:

```
[{
   "constant":true,
   "Inputs":[],
   "name":"status",
   "outputs":[{
      "name":"",
      "type":"uint8"
    }],
   "payable":false,
   "stateMutability":"view",
   "type":"function"
},
...
```

Para a verificação de data e configuração de fuso horário foram usadas os bibliotecas *datetime* e *pytz*. Da *pytz*, foi usada a *timezone* para configurar o fuso horário local que é necessário para fazer as verificações das datas das reservas de maneira correta. O fuso horário escolhido foi "America/São Paulo" pois os testes foram realizados em Itajubá - MG.

O registro de entrada e saída no ambiente foi feito usando-se a módulo *logging*. A configuração básica é feita especificando-se o nome do arquivo de log e nível de severidade das informações que serão registradas. Os níveis que se pode escolher em ordem decrescente de importância são `CRITICAL`, `ERROR`, `WARNING`, `INFO` e `DEBUG`. O nível que for escolhido indica que os tipos de registros podem ser classificados do seu nível e os acima. Como foi configurado para `INFO`, pode-se registrar eventos como `INFO`, `WARNING`, `ERROR` e `CRITICAL`.

[7] Providers — Web3.py 5.0.0 documentation: <https://web3py.readthedocs.io/en/stable/providers.html> Acesso em: 14 de ago. de 2019.

[8] Ethereum API | IPFS API & Gateway | ETH Nodes as a Service | Infura: <https://infura.io> Acesso em: 14 de ago. de 2019.

Os possíveis eventos na interação do usuário com o sistema estão presentes na Tabela 1. Todos os registros de informação são precedidos de data e hora atuais a ocorrência.

Tabela 1 – Quadro de eventos e resultados possíveis.

Evento	**Resultado**
Problemas da decodificação do áudio	Registro do tipo *ERROR* com a mensagem "*Decode failed!*"
Acesso bem sucedido	Registro do tipo *INFO* com a mensagem "Acesso bem sucedido com o seguinte endereço de contrato:" seguida do endereço de contrato inteligente acessado
Credencial diferente	Registro do tipo *ERROR* com a mensagem "Acesso mal sucedido com o endereço de contrato:" seguido do endereço do contrato e "ERRO: Senha inválida!"
Data fora do intervalo da reserva	Registro do tipo *ERROR* com a mensagem "Acesso mal sucedido com o endereço de contrato:" seguido do endereço do contrato e "ERRO: Data fora da reserva"

Para acionamento do circuito controlador da fechadura foram utilizados os pinos GPIO (*general-purpose input/output*) da Raspberry Pi. Para isso, foi utilizado o módulo `RPi.GPIO` que permite manipular o nível lógico dos pinos de acordo com o resultado da verificação do usuário. Os pinos utilizados foram os pinos 6, de aterramento, e o pino 8 (GPIO14) que foi ligado ao circuito de acionamento da fechadura. O diagrama dos pinos da Raspberry Pi 3 B+ podem ser visualizados na Figura 14.

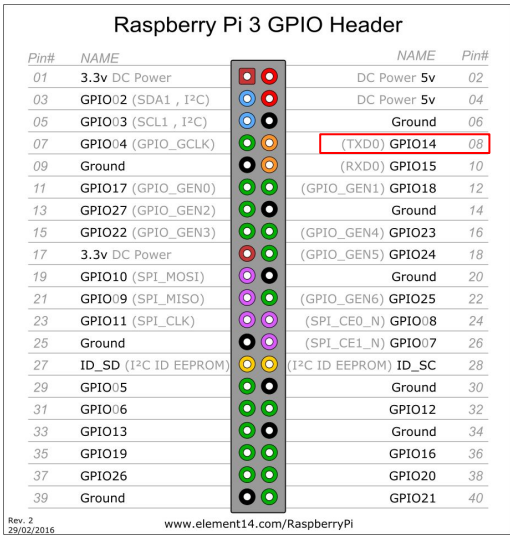

Figura 14 – Pinos GPIO da Raspberry Pi 3 B+.

Fonte: element14.com

4.2 Circuito de acionamento da fechadura

Para realizar o acionamento da fechadura eletrônica foi necessário o desenvolvimento de um circuito eletrônico. A Figura 15, mostra o projeto do esquema elétrico do circuito. Nele os elementos mais importantes são os dois transistores TIP120[9] componentes 1 e 2, o regulador de tensão L7805[10] componente 3 e a fechadura solenoide componente 4.

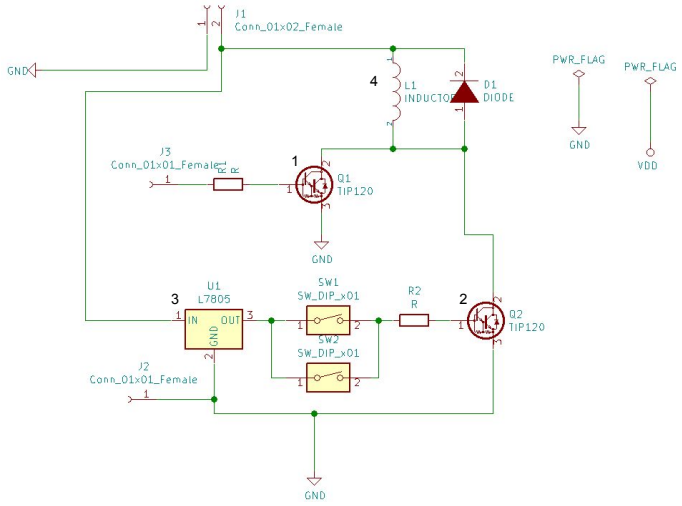

Figura 15 – Esquemático construido

A fechadura solenóide é uma fechadura eletromecânica controlada. Seu componente principal é a bobina elétrica com um núcleo ferromagnético móvel no centro, o êmbolo. A bobina ou indutor tem um comportamento peculiar pois é um componente magnético. Em posição de repouso, o trinco ligado ao êmbolo segura a porta. Quando uma corrente elétrica atravessa a bobina é criado um campo magnético alinhado com o eixo da bobina que faz com que se manifeste uma força no êmbolo (HOROWITZ; HILL, 2015).

O segundo elemento de importância é o transistor. A estrutura do transistor de junção bipolar (BJT - *Bipolar Junction Transistor*) determina as suas características de operação (FLOYD, 2012). O BJT é construído de três regiões de semicondutores dopados do tipo p, que apresentam lacunas como condutores de corrente, e do tipo n, que apresenta elétrons livres como condutores de corrente separadas por duas junções pn. As três regiões são chamadas base, coletor e emissor. O transistor TIP120 é do tipo npn que consiste de duas regiões n separadas por uma do tipo p. O termo bipolar se refere ao uso de tanto elétrons quanto lacunas como condutores de corrente na estrutura do transistor. A estrutura do transistor e sua representação como componente de circuito eletrônico podem ser vistos na Figura 16.

[9] Complementary power Darlington transistors: <https://www.st.com/resource/en/datasheet/tip120.pdf>

[10] Datasheet - L78 - Positive voltage regulators ICs: <https://www.st.com/resource/en/datasheet/l78.pdf>

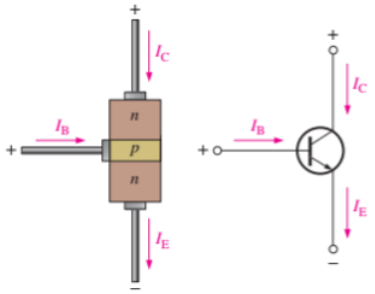

Figura 16 – Estrutura e representação do transistor.

Fonte: Electronic devices: conventional current version(FLOYD, 2012)

O transistor pode ser utilizado com dois objetivos (FLOYD, 2012): 1) para funcionar como amplificador de sinal, no processo de aumentar exponencialmente a amplitude de um sinal eletrônico; 2) como chave eletrônica, quando o transistor opera em *cutoff* ou em saturação. *Cutoff* é o estado em que o transistor não está conduzindo corrente do coletor para o emissor pois não há uma tensão aplicada na junção base-emissor corrente incidindo sobre a base do transistor. Saturação é o estado contrário ao *cutoff*, há tensão aplicada na junção e corrente incidindo na base e há corrente do coletor para o emissor.

A representação dos estados de saturação e de *cutoff* podem ser vistos na Figura 17. Vale ressaltar que a corrente do coletor é aumentada quando chega ao emissor como pode ser vista na Equação 4.1 e melhor visualizado na Figura 16:

$$I_E = I_C + I_B \tag{4.1}$$

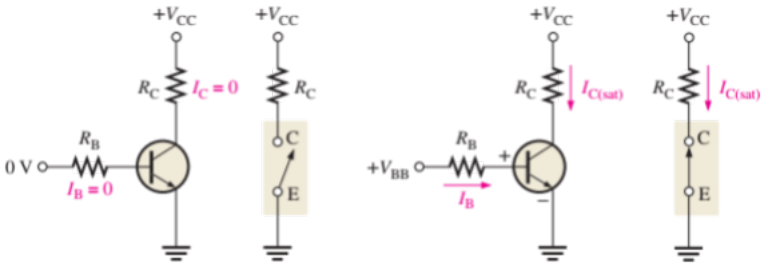

Figura 17 – Representaçao dos estados de *cutoff* e saturação do transistor.

Fonte: Electronic devices: conventional current version (FLOYD, 2012)

Um regulador de tensão provê uma tensão de saída constante que é essencialmente independente da tensão de entrada, corrente de carga de saída e temperatura (FLOYD, 2012). A maioria dos reguladores de tensão se enquadram em duas categorias: reguladores lineares e reguladores de chaveamento. O 7805 é do tipo linear.

A série de reguladores 78xx é representante de dispositivos de três terminais que provêm uma tensão positiva fixa de saída (FLOYD, 2012). Os três terminais são entrada, saída e terra como indicado na Figura 18.

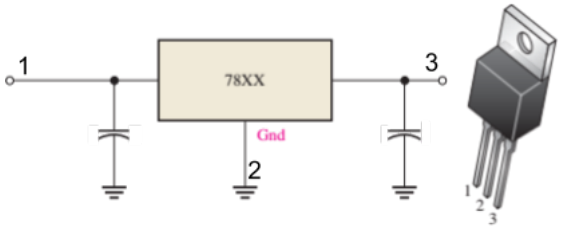

Figura 18 – Esquemático do transistor.

Fonte: Adaptado de Electronic devices: conventional current version(FLOYD, 2012)

Os dois últimos dígitos no número representam a tensão de saída. Por exemplo, o 7805 é um regulador de +5,0V. Para qualquer regulador, a tensão de saída pode variar 4% para mais e para menos da tensão nominal de saída. Então, o 7805 pode ter saída entre 4,8V e 5,2V mas vai permanecer constante nessa faixa. E a tensão de entrada deve ser aproximadamente 2,5V acima da tensão de saída para manter a regulação (FLOYD, 2012).

O circuito foi projetado para ser acionado por um dos pinos de GPIO de uma Raspberry Pi 3. Como pode ser observado na Figura 19, um pino é ligado a base de um dos transistores por meio de um resistor e o outro pino é o terra. O outro transistor, que também aciona a tranca solenoide, tem sua base conectada a dois botões em paralelo que foram adicionados ao circuito apenas para casos problemas inesperados no funcionamento da controladora Raspberry Pi ou na execução do *script* Python. Esses dois botões estão ligados ao regulador de tensão que fornece 5,0V provenientes de uma fonte de alimentação de 12,0V ligada ao circuito. Um botão foi colocado para fora da sala e o outro para dentro, para possibilitar o acionamento de ambos os lados da porta.

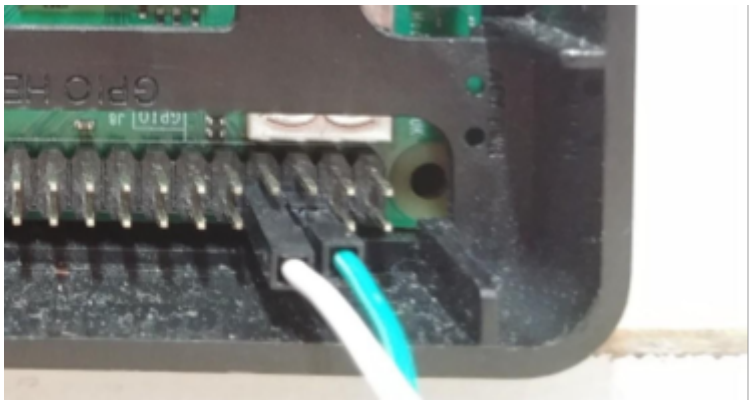

Figura 19 – Pinos GPIO utilizados da Raspberry Pi.

O esquema do circuito foi projetado no software Kicad[11] e, em seguida, com a mesma ferramenta projetou-se as trilhas para o circuito impresso. Nesta etapa, especifica-se a dimensão de cada componente para que possa ser gerado um circuito com o espaçamento necessário para o encaixe de seus pinos para depois passar pelo processo de soldagem. Especificados os componentes, estes devem ser rearranjados de forma a evitar que as trilhas que forem desenhadas não se cruzem para minimizar o número de camadas de trilhas necessárias para se montar o circuito. O resultado final dessa etapa pode ser visto na Figura 20.

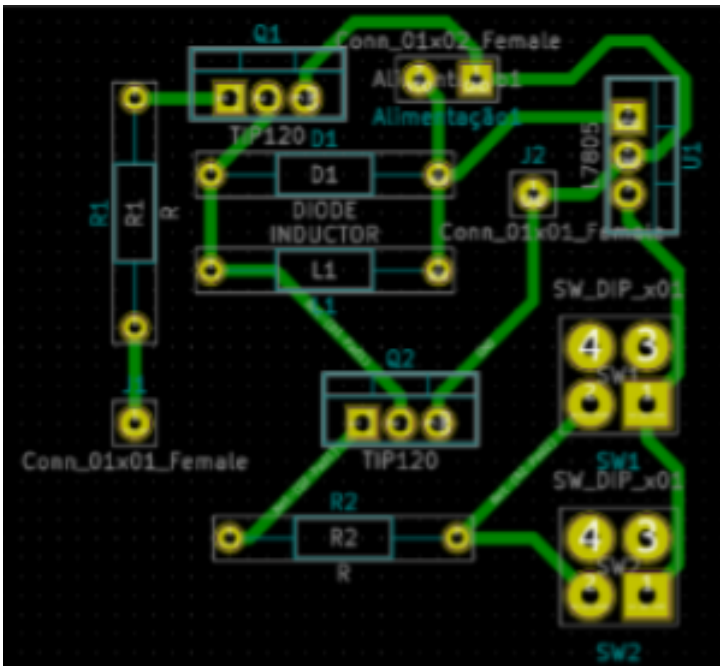

Figura 20 – Esquemático das trilhas do circuito.

Com o esquemático das trilhas concluído, foi necessário passá-lo para a placa de fenolite, que é um laminado plástico industrial utilizado como isolante elétrico com as faces laminadas de cobre. Optou-se pelo método de transferência térmica. Esse processo envolve imprimir o circuito em papel fotográfico com impressora a lazer e contato com uma fonte calor como um ferro de passar. Com esse processo a tinta do papel fotográfico passa para a placa de fenolite. O circuito impresso no papel fotográfico pode ser visualizado na Figura 21.

[11] KiCad EDA <http://www.kicad-pcb.org/> Acesso em: 15 de ago. de 2019.

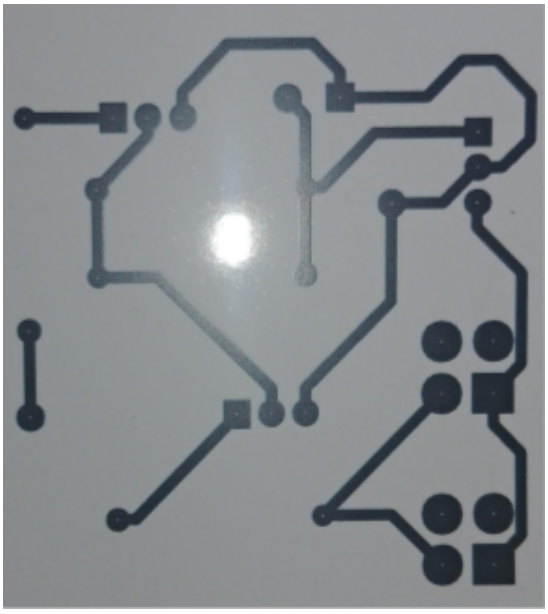

Figura 21 – Trilhas do circuito impressas em papel fotográfico.

Em seguida, a placa de fenolite e o circuito impresso foram mergulhadas em uma solução de percloreto de ferro dissolvido em água. Nessa solução apenas o cobre por debaixo da tinta do circuito impresso não é corroído deixando assim as trilhas, que podem ser visualizadas na Figura 22.

Com as trilhas impressas na placa de fenolite, foram feitos os furos para encaixe dos componentes eletrônicos. Foram utilizados uma furadeira de bancada e uma broca de 0.8 mm. Em seguida, os componentes foram soldados a placa e o resultado final pode ser visto na Figura 23.

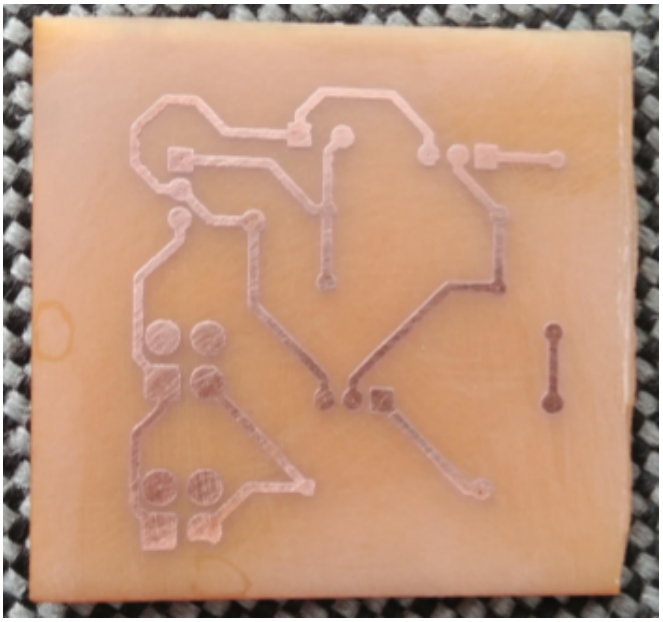

Figura 22 – Trilhas do circuito impresso.

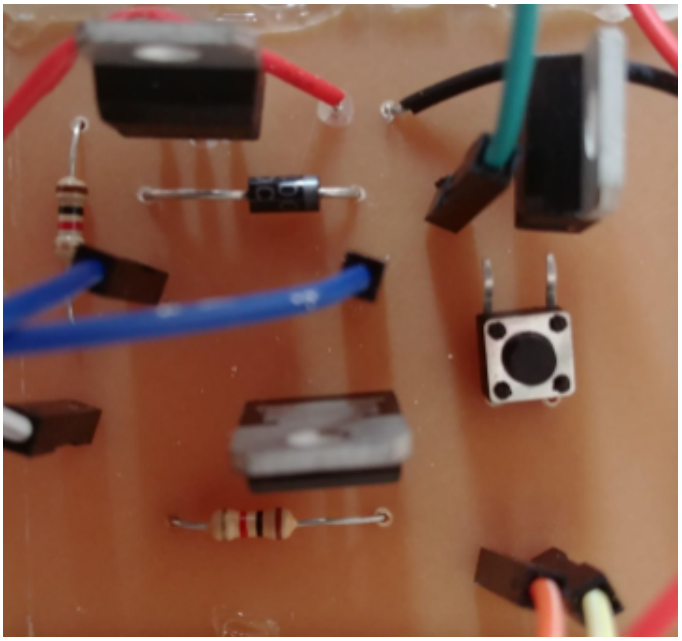

Figura 23 – Circuito soldado.

Na Figura 24, o protótipo da fechadura pode ser visualizado completamente. Esse protótipo foi fixado na parede ao lado do batente da porta próximo ao rodapé de azulejo. A placa de som é o componente 1, a Raspberry Pi 3 B+ é o 2, o circuito soldado é o 3 e a tranca solenoide é o 4.

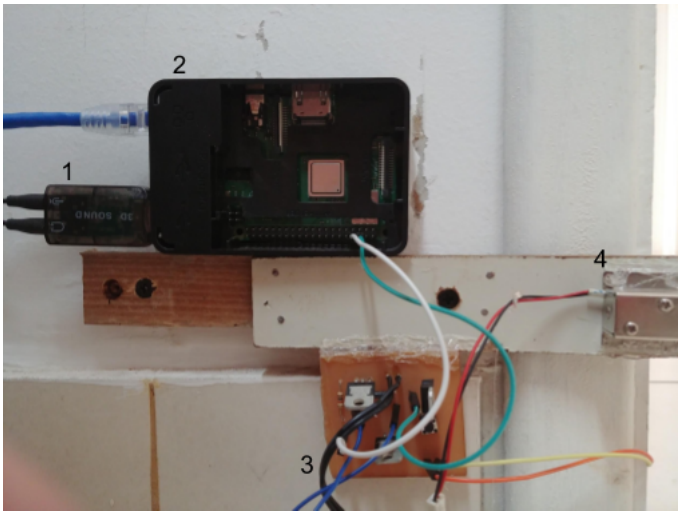

Figura 24 – Protótipo da fechadura.

4.3 Servidor

Para hospedar na nuvem componentes independentes do sistema que juntos são importantes para seu bom funcionamento foi configurado uma máquina como servidor. Os componentes são: um *website* onde os usuários fazem seus cadastros e reservas, o

banco de dados e uma Interface de Programação de Aplicações (da sigla em inglês API) de autenticação e acesso aos dados de reservas. A hospedagem desses sistemas foi feita utilizando-se de uma instância do serviço Amazon *Elastic Compute Cloud* (EC2) e uma instância do serviço Amazon *Relational Database Service* (Amazon RDS) do nível gratuito da AWS (Amazon Web Services[12]).

O Amazon Elastic Compute Cloud (Amazon EC2) é um web service que disponibiliza capacidade computacional segura e redimensionável na nuvem[13]. Ele foi projetado para facilitar a computação em nuvem na escala da web para os desenvolvedores. A configuração da máquina escolhida para hospedar o site e a API pode ser visualizada na Tabela 2.

Tabela 2 – Configuração da máquina de hospedagem do servidor.

Tipo de instância	**Procesador**	**Memória RAM (GiB)**	**Armazenamento (GiB)**
t2.micro	Intel(R) Xeon(R) CPU E5-2676 v3 @ 2.40GHz	1	20

O Amazon RDS facilita a configuração, a operação e a escalabilidade de bancos de dados relacionais na nuvem. O serviço oferece capacidade econômica e redimensionável, e automatiza tarefas demoradas de administração, como provisionamento de hardware, configuração de bancos de dados, aplicação de patches e backups[14]. O sistema de gerenciamento de banco de dados escolhido foi o MySQL[15], que utiliza a linguagem SQL como interface. A configuração da máquina para hospedar o banco de dados pode ser visualizada na Tabela 3. Das informações que são armazenadas no banco de dados as mais importantes para o projeto são as dos usuários e reservas que são pertinentes para implantar o contrato na *blockchain*.

Tabela 3 – Configuração da máquina de hospedagem do banco de dados.

Tipo de instância	**Procesador**	**Memória RAM (GiB)**	**Armazenamento (GiB)**
db.t2.micro	Intel(R) Xeon(R) CPU E5-2676 v3 @ 2.40GHz	1	20

O sistema de reservas é uma aplicação criada para armazenar os dados de usuários e reservas, sendo este sistema criado sobre o *framework* MVC (*Model-View-Controller*) PHP Laravel. *Model* ou modelo é usado para interagir com tabelas no banco de dados que

[12] Amazon Web Services <https://aws.amazon.com/pt/> Acesso em: 15 de ago. de 2019
[13] Elastic Compute Cloud - Amazon EC3 - AWS: <https://aws.amazon.com/pt/ec2/> Acesso em: 15 de ago. de 2019.
[14] AWS RDS (Relational Database Service) - Amazon Web Services: <https://aws.amazon.com/pt/rds/> Acesso em: 15 de ago. de 2019.
[15] MySQL: <https://www.mysql.com/> Acesso em: 15 de ago. de 2019.

o representa, *View* é referente a parte visual de cada página e *Controller* ou controlador específica funções para interagir com os objetos de cada uma das entidades.

O framework facilita a implementação dos CRUD's (*Create-Read-Update-Delete*) das entidades de maneira mais rápida, automatizada, simples e elegante. Isso tudo é feito com a ferramenta Eloquent ORM (*Object-Relational Mapping*)[16]. Sistemas ORM são construídos em cima de banco de de dados relacional tradicionais, que permite que o programador defina mapeamentos entre tuplas nas relações do banco de dados e objetos na linguagem de programação (SILBERSCHATZ et al., 2010).

Um objeto ou uma sequência deles pode ser obtida baseando-se na seleção condicional dos seus atributos. O programa pode opcionalmente atualizar tais objetos, criar novos, ou especificar que um objeto seja excluído, e então emite um comando para salvá-lo. O mapeamento dos objetos para as relações são usadas para correspondentemente para atualizar, inserir e excluir tuplas no banco de dados (SILBERSCHATZ et al., 2010).

Cada tabela no banco de dados tem um modelo correspondente que é usado para interagir com ela. Eles que permitem a consulta de dados nas tabelas, assim como inserir novos dados. Para cada modelo também especifica-se um controlador e uma migração. No controlador são especificados os métodos de cada uma das entidades e a migração funciona como um controle de versão do banco de dados e especifica os atributos da entidade. As principais entidades do sistema são "reserva" e "usuário".

O *framework* Laravel também facilita a implementação de autenticação. Apenas com um comando o framework configura o ambiente com views, rotas, controladores, modelos e migrações necessários para registrar usuários, realizar o *login* e *logout* e redefinir senhas.

Uma API de autenticação foi desenvolvida para permitir o *login* na aplicação e que as informações de reservas pudessem ser acessadas. O Laravel fornece para isso o Passport[17] que provê uma implementação completa do OAuth2. OAuth2 fornece fluxos de autorização específicos para aplicações web, aplicações desktop e celulares.

Para implementar a API é necessário especificar as possíveis rotas que uma requisição pode percorrer. A especificação das rotas da API encontra-se no Apêndice A.

A comunicação com a *blockchain* é feita usando a biblioteca JavaScript web3.js fornecida pela Ethereum para comunicação com a sua API. O objetivo desta camada é armazenar informações das reservas para em seguida implantar o contrato inteligente na *blockchain* que posteriormente serão utilizados pelo Raspberry Pi para validação dos acessos.

[16] Eloquent: Getting Started - Laravel - The PHP Framework for Web Artisans: <https://laravel.com/docs/5.8/eloquent> Acesso em: 15 de ago. de 2019.

[17] Laravel Passport - Laravel - The PHP Framework for Web Artisans: <https://laravel.com/docs/5.8/passport> Acesso em: 15 de ago. de 2019.

Para a biblioteca web3.js se comunicar com a API da Ethereum são necessárias algumas configurações. Detalhes da comunicação e da implantação do contrato inteligente encontram-se no Apêndice B.

4.4 Aplicativo de celular

A aplicação de *smartphone* permite que o usuário faça *login* no sistema, visualize as informações das reservas e transmita as credenciais referentes a elas. Ela foi desenvolvida nativamente para celulares da plataforma Android. O aplicativo é bem simples, possui apenas três telas. Uma de *login*, uma de perfil do usuário apenas com um botão para *logout* e outra onde as reservas são listadas.

Cada tela é atribuída a uma atividade (*Activity*) que é o componente que descreve o seu visual com o qual o usuário vai interagir. A especificação dos componentes visuais e como eles vão ser arranjados é definida em um arquivo XML (*eXtensible Markup Language*) para cada uma das telas.

A tela inicial apresentada para o usuário é a tela de *login* que pode ser visualizada na Figura 25. Nela existem dois campos do tipo *EditText*, que é um elemento de interface de usuário para inserir e modificar texto. Um deles é destinado para inserir o *e-mail* do usuário e o outro para inserir a sua senha. Logo abaixo deles há um botão com o texto *"Log In"*.

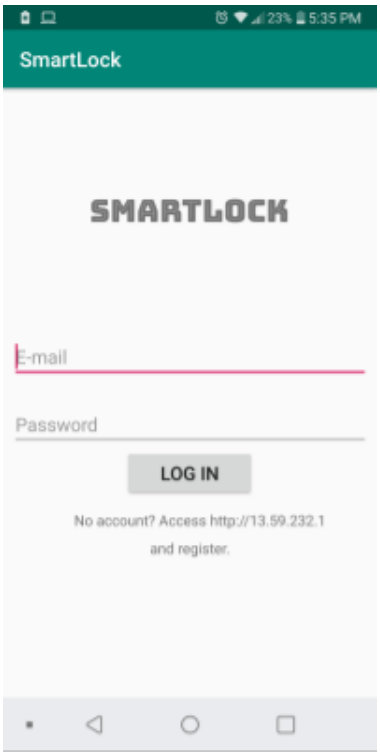

Figura 25 – Tela de *login* do aplicativo.

Quando o botão da tela inicial for selecionado, é chamada a execução da função *login*. Está função valida *e-mail* e a senha informados pelo usuário. Com o retorno dessa função é obtido o *token* de acesso que depois será usado em todas as outras requisições. A última operação realizada pela função *login* é iniciar outro componente do tipo atividade para carregar a tela do aplicativo com as abas "Perfil" e "Reservas" passando o *token* de acesso para esse componente.

Na tela de perfil, foi adicionado um botão para realizar o *logout* do sistema. Ela pode ser visualizada na Figura 26. Quando esse botão é selecionado é executada a função de *logout* e após a sua execução o *token* é revogado e não tem mais validade para fazer novas requisições.

Figura 26 – Tela de perfil do aplicativo.

Durante a criação da tela das reservas, que pode ser visualizada na Figura 27, três operações importantes são realizadas. Primeiro, é configurado o protocolo Chirp.io. Segundo, é instanciada um objeto que será usado para gerar a credencial de acesso com as informações obtidas de cada uma das reservas usando o algoritmo de *hashing* do tipo SHA256. Terceiro, uma uma função é executada para obter as informações das reservas e listada-las. As informações das reservas são recebidas no formato JSON (*JavaScript Object Notation*) e pode ser visualizadas no exemplo do Apêndice C.

Figura 27 – Tela de reservas do aplicativo.

Quando o usuário seleciona uma das reservas, são enviados sua credencial de acesso e seu endereço do contrato inteligente, que são únicos para cada uma delas. A credencial é formada pelos oito primeiros digitos da *hash* produzida e que é descrita na subseção 3.1.4. Essas informações são codificadas no formato de um áudio pelo protocolo Chirp.io que é emitido pelo auto-falante do celular.

5 Segurança

Nesta seção serão discutidas vulnerabilidades encontradas no sistema pelos participantes dos testes. As vulnerabilidades foram encontradas utilizando-se de ferramentas gratuitas de escaneamento dos componentes de sistema, por discussões com os envolvidos no projeto, avaliação manual de código fonte e busca na literatura. Para cada componente será evidenciado o meio de identificação das vulnerabilidades. Serão discutidas como essas vulnerabilidades poderiam ser utilizadas para prejudicar o sistema e seus usuários e como elas poderiam ser corrigidas em trabalhos futuros. Cabe deixar claro que a identificação e correção dessas vulnerabilidades não torna o sistema completamente seguro pois estas são referentes a pontos específicos do sistema e podem haver vulnerabilidades em outros pontos que não foram analisados.

5.1 Rede *blockchain* da Ethereum

A avaliação da segurança da rede Ethereum foi feita por meio da busca de trabalhos que abordaram o tema.

Em relação a linguagem Solidity, deve-se mencionar a propriedade *Reentrancy*, ou reentrada das funções, que possibilita que funções sejam re-invocadas antes do término da sua execução o que pode levar a comportamentos inesperados do contrato inteligente (ATZEI; BARTOLETTI; CIMOLI, 2017). Certas funções em alguns contratos possibilitam sacar a quantia de *ether* armazenada neles. Enquanto a chamada dessas funções era executada há a possibilidade de fazer uma nova o que permite sacar a mesma quantia mais de uma vez. Essa vulnerabilidade foi explorada no ataque mais reconhecido na rede chamado "*The DAO Attack*"[1] em 2016 onde foram roubados 3,3 dos 150 milhões de dolares arrecadados pela empresa em seu *crowd-funding*. Esse ataque pôde ser revertido com um *hard-fork*, que é uma atualização no protocolo utilizado pelos nós que transformam certos blocos inválidos em válidos[2], na rede. Esse tipo de vulnerabilidade pode ser resolvido usando-se das melhores práticas no desenvolvimento dos contratos inteligentes propostas pela própria comunidade da rede[3].

[1] Understanding The DAO Attack - CoinDesk: <https://www.coindesk.com/understanding-dao-hack-journalists> Acesso em: 28 de ago. de 2019.
[2] Hard Fork: <https://www.investopedia.com/terms/h/hard-fork.asp> Acesso em: 28 de ago. de 2019.
[3] Ethereum Smart Contract Best Practices: <https://consensys.github.io/smart-contract-best-practices/> Acesso em: 28 de ago. de 2019.

Outro método para contornar esse problema é usar outra rede *blockchain*. Com o *HyperLedger Fabric*[4], que permite implementar uma plataforma *blockchain* onde os nós precisam de permissão para participar da rede e que permita acesso somente aos usuários da plataforma de reservas.

5.2 Website

Para avaliar as possíveis vulnerabilidades do website, foi utilizado um escaneador chamado Arachni[5]. Essa ferramenta é um framework de alto desempenho e modular na linguagem Ruby. Para aprimorar os resultados do escaneamento foram fornecidas credenciais de acesso para que o framework pudesse avaliar a segurança de páginas que só estivessem disponíveis após o *login*.

5.2.1 Formulários de senha não encriptados

O servidor está configurado com o protocolo de comunicação HTTP, da sigla em inglês *Hypertext Transfer Protocol*, que trabalha diretamente com texto com as mensagens usando o padrão de código de caracteres alfa-numéricos como troca de informação (KUROSE; ROSS, 2007). Isso significa que os dados que são transmitidos via HTTP podem ser capturados e seu conteúdo visualizado.

Pessoas má intencionadas, conhecidas por *hackers*, costumam tentar comprometer credenciais passadas do cliente para o servidor usando HTTP. Isso pode ser realizado por vários ataques diferentes do tipo *Man-in-the-Middle* (MitM) ou por captura de pacotes. Nos ataques MitM, o *hacker* se posiciona entre as partes que se comunicam com ambas acreditando que se comunicam entre si mas na verdade estão se comunicando com o atacante (ERICKSON, 2008). O atacante pode então interceptar as mensagens e depois transmiti-las ou então substituí-las por outras mensagens.

Com esse tipo de ataques, é possível que o atacante obtenha posse das credenciais do sistema, utilize-se delas para cancelar ou alterar as informações das reservas dos usuários prejudicando a integridade de seus dados. Ou em casos mais graves, o atacante pode obter informações de localidade e data da reserva ou endereço de moradia dos usuários pondo assim em risco sua integridade física.

Para manter os dados privados e prevenir que eles sejam interceptados, as mensagens do protocolo HTTP podem ser enviadas através de *Secure Sockets Layer* (SSL) ou de *Transport Layer Security* (TLS). O TLS é sucessor do SSL, baseado na 3ª versão do

[4] Hyperledger Fabric - Hyperledger Fabric - Hyperledger Confluence: <https://wiki.hyperledger.org/display/fabric/Hyperledger+Fabric> Acesso em: 28 de ago. de 2019.

[5] Arachni - *Web Application Security Scanner Framework*: <https://www.arachni-scanner.com/>

SSL (KUROSE; ROSS, 2007; STEVENS, 2012). Quando um desses padrões é utilizado o protocolo HTTP é referenciado como HTTPS.

5.2.2 Cookie HTTP

O HTTP é um protocolo sem estado, o que significa que o servidor não é capaz de determinar que requisições são feitas por quais clientes, e que clientes são autenticados e não autenticados (KUROSE; ROSS, 2007).

O uso de *cookies* nos cabeçalhos das requisições permite que servidores identifiquem cada cliente e então diferenciar quais mantêm uma autenticação válida daqueles que não mantêm. Esses são chamados de *cookies* de sessão.

Quando um *cookie* é definido pelo servidor há vários marcadores, em inglês *flags*, que podem ser ajustados para ajustar propriedades do *cookie* e como ele deve ser manipulado pelo navegador.

A *flag HttpOnly* ajuda a prevenir que *scripts* do lado do cliente, como na linguagem JavaScript, acessem e usem o cookie. Esse tipo de ataque é chamado *Cross-site scripting*. Nele um atacante explora a confiança que o navegador dá a um servidor confiável e executa um *script* injetado no navegador com os privilégios do servidor (WASSERMANN; SU, 2008). Isso permite que os *cookies* que têm o *token* da sessão sejam acessados e que o atacante assuma a identidade de qualquer usuário podendo acessar as mesmas informações mencionadas na Seção 5.2.1.

Quando a *flag secure* não está marcada é possível que o navegador envie o *cookie* por um canal não encriptado possibilitando que um atacante o acesse. E com isso tenha acesso às informações mencionadas na Seção 5.2.1.

Para ajudar na prevenção desses ataques basta que essas *flags* sejam marcadas nos *cookies* das sessões.

5.3 Aplicativo

Em conversa com o professor orientador desta dissertação e com os alunos que participaram dos testes levantou-se o questionamento se seria possível gravar o áudio reproduzido pelo alto-falante do celular. Para testar essa hipótese usou-se uma ferramenta para gravar áudio disponível em um *smartphone* e constatou-se que essa simples técnica poderia ser utilizada por uma pessoa má intencionada para acessar uma propriedade com uma reserva vigente, bastando deixar um gravador próximo a fechadura e poderia então acessar a propriedade.

Uma contramedida para essa vulnerabilidade é empregar o uso de autenticação de múltiplos fatores. Esse tipo de autenticação envolve o uso de dois ou mais fatores

independentes na verificação de um usuário. São alcançadas combinando a tradicional forma de autenticação por texto, *email* e senha por exemplo, com outros fatores. Esses outros fatores podem ser dispositivos USB, celulares ou senhas de uso único (SABZEVAR; STAVROU, 2008).

Para aprimorar a segurança do sistema poderia-se utilizar da Autenticação de dois fatores no momento de acesso a propriedade. Uma ferramenta que implementa esse tipo de autenticação e que poderia ser facilmente integrada ao sistema é a Google Autenticator[6]. Existem implementações PHP abertas (*open source*) disponibilizadas pela comunidade no Github[7,8] para incorporá-lo ao servidor. Também é necessário que o usuário instale em seu celular o aplicativo do Google Autenticator e configure uma conexão com o sistema lendo um QRCode[9] que seria informado no site.

No momento de acesso, uma senha com validade temporária (30 segundos para cada uma) fica disponível para ser digitada em um painel digital ou ser transferida pelo próprio alto-falante do celular pelo protocolo Chirp.io. Com isso a cada acesso uma outra pessoa tem uma janela de tempo muito curta para fazer acesso com o áudio gravado.

Uma outra possibilidade para solucionar o problema de se gravar o áudio é utilizar a tecnologia NFC. É uma tecnologia que funciona com o uso de campos de indução magnética e tem uma distância máxima de operação de 20 centímetros. Dentre as principais aplicações para a tecnologia estão as de "toque para confirmar". Isso permite que dispositivos com NFC, como celulares mais modernos, ajam como chaves eletrônicas para acessar casas, escritórios ou carros por exemplo (WANT, 2011). Usar essa tecnologia permite substituir completamente o uso do protocolo Chirp.io para fazer acesso ao ambiente porém essa tecnologia não está presente na maioria dos celulares atualmente.

5.4 Raspberry Pi

A avaliação da segurança da fechadura em si foi realizada analisando as configurações de segurança, ponto que não recebeu a devida atenção durante o desenvolvimento do projeto. Ela é controlada por uma Raspberry Pi 3 B+ que está conectada a internet e qualquer interação com ela deve ser feita através SSH (*Secure Shell*). Por esses motivos ela está sujeita a ser acessada indevidamente devido a falhas nas configurações de segurança. Uma pessoa que conseguir acessar a Raspberry Pi pode tanto conseguir acesso a propriedade

[6] Google Authenticator - Apps on Google Play: <https://play.google.com/store/apps/details?id=com.google.android.apps.authenticator2&hl=en> Acesso em: 15 de ago. de 2019

[7] *Google Authenticator PHP class*: <https://github.com/PHPGangsta/GoogleAuthenticator> Acesso em: 15 de ago. de 2019.

[8] *Google Authenticator - by SonadaCI:* <https://github.com/sonata-project/GoogleAuthenticator> Acesso em: 15 de ago. de 2019.

[9] Simple questions: What are QR codes and why are they useful? | Digital Citizen: <https://www.digitalcitizen.life/simple-questions-what-are-qr-codes-and-why-are-they-useful> Acesso em: 15 de ago. de 2019

quanto danificar a fechadura. As mudanças de configuração que serão propostas aqui são recomendações indicadas pela documentação oficial[10].

O primeiro passo para segurar a Raspberry Pi é desabilitar a sua conta padrão `pi` no sistema porque ela utiliza "*raspberry*" como senha o que é um fato conhecido por toda a comunidade. Mas antes disso é necessário criar uma outra conta no sistema e configurar uma nova senha para ela. Em seguida, será necessário ajustar a senha do usuário *root* para algo mais seguro. E por fim, deve-se desabilitar a conta *pi*.

Detalhes de implementação de cada uma das recomendações estão presentes no Apêndice D

5.5 Contrato Inteligente

Constatou-se uma vulnerabilidade analisando o código-fonte do contrato inteligente. A causa dessa vulnerabilidade foi um erro de implementação do autor. Da forma que o contrato inteligente foi projetado, qualquer pessoa com o endereço do contrato inteligente poderia chamar sua função `getInfo` e obter informações sensíveis sobre a reserva, como credencial de acesso e datas.

Isso é possível pois não foram feitas restrições de quem pode interagir com essa função. Então um atacante com o endereço do contrato poderia obter as informações necessárias para acessar a propriedade e as datas inicial e final para fazê-lo.

Mas as partes que deveriam interagir com as informações fornecidas por essa função são o hóspede e a fechadura. Não é viável que o proprietário possa ter acesso a essas informações pois ele poderia acessar a propriedade durante a reserva e poderia acessar a propriedade. Para interagir com o contrato é necessário que uma das partes faça uma transação com o endereço do contrato na rede *blockchain* e por meio do endereço de quem a faz se verifica quem está fazendo essa ação. O hóspede a faz através da sua carteira de criptomoedas. Já a fechadura faz essa interação por meio do provedor Infura.io. Então pode-se usar essas informações para restringir a interação com a função `getInfo` e solucionar esse problema.

Para essa tarefa é possível usar a função `require` que exige que certas condições sejam atendidas e por ela especificar que endereços podem interagir com essa função. Para isso no momento de se implantar o contrato deve-se declarar uma nova variável que armazene o endereço que o provedor Infura.io utiliza e com o função `require` restringir os endereços. A linguagem Solidity permite obter o endereço de quem está fazendo uma transação para interagir com o contrato inteligente através da variável `msg.sender`. Com isso o problema está solucionado.

[10] *Securing your Raspberry Pi - Raspberry Pi Documentation*: <https://www.raspberrypi.org/documentation/configuration/security.md> Acesso em: 15 de ago. de 2019.

5.6 Segurança Física

Na avaliação e discussão sobre o protótipo desenvolvido indentificou-se vulnerabilidades no projeto em relação a fatores externos a fechadura. Esses fatores são: (i) corte de fornecimento de energia elétrica a fechadura e (ii) falha na conexão da fechadura com a internet.

Como a tranca solenoide utilizada retrai o trinco somente quando uma tensão específica é aplicada à ela, um corte no fornecimento de energia elétrica não dá acesso indevido a terceiros. Mas a falta de eletricidade na propriedade implicaria no desligamento da fechadura e, portanto, a impossibilidade de acessar o recinto. Para corrigir essa vulnerabilidade, é necessário desenvolver um sistema de alimentação auxiliar para a fechadura que é acionado durante quedas de energia e é carregado durante o funcionamento normal da rede elétricade.

As verificações de acesso dependem diretamente da conexão da fechadura com a internet e uma falha nessa conexão impede que os usuários acessem a propriedade sendo assim uma vulnerabilidade. Para contornar esse problema, pode-se utilizar o *cron*[11] que é uma ferramenta presente nos sistemas operacionais UNIX. Essa ferramenta permite agendar tarefas para serem executadas periódicamente. Com isso pode-se agendar uma tarefa que acesse o banco de dados do projeto, obtenha as informações das reservas válidas para aquele dia e as armazene localmente para serem utilizadas no momento de acesso do usuário. Essa tarefa deve ser agendada para se repetir diariamente, duas vezes ao dia ou da forma que melhor satisfizer as necessidades específicas de onde foi instalada a fechadura.

[11] cron(8) - Linux manual page: <http://man7.org/linux/man-pages/man8/cron.8.html> Acesso em: 24 de ago. de 2019

6 Metodologia de Avaliação e Resultados

O protótipo desenvolvido foi disponibilizado para uso e avaliação para um conjunto de alunos de graduação da Universidade Federal de Itajubá - *Campus* Itajubá. Após utilizarem a solução, ela foi avaliada através de um questionário quantitativo que mediu as seguintes características da solução: **Usabilidade**, quanto a sua operabilidade, design da interface de usuário e apreensibilidade; **Satisfação**, quanto ao seu conforto, confiança e utilidade; **Desempenho**, quanto ao comportamento em relação ao tempo; **Funcionalidade**, quanto a sua completude funcional; e **Confiabilidade**, quanto a sua disponibilidade. O questionário utilizado está disponível no **Apêndice E**, baseado no norma ISO/IEC 25010[1].

O quesito usabilidade é dividido em várias características das quais as que serão avaliadas estão descritas a seguir:

- **Operabilidade**: grau com que um produto ou sistema é fácil de operar, de se controlar e apropriado para usar;
- **Design da Interface de Usuário**: grau com que a interface de usuário possibilita a interação agradável e satisfatória ao usuário;
- **Apreensibilidade**: grau com que o produto ou sistema permite que o usuário aprenda a usá-lo com efetividade e eficiência.

O quesito satisfação possui várias características das quais as que serão avaliadas estão descritas a seguir:

- **Conforto**: grau com que o usuário está satisfeito com conforto físico;
- **Confiança**: grau com que o usuário tem confiança de que o produto ou sistema se comportará como esperado;
- **Utilidade**: grau com que o usuário está satisfeito com sua percepção em atingir objetivos pragmáticos, incluindo resultados de uso e as consequências do uso.

O quesito desempenho ramifica-se em diversas características das quais será avaliada o **comportamento em relação ao tempo** que trata do grau com que as respostas,

[1] ISO/IEC 25010:2011 *Systems and software engineering - Systems and Software Quality Requirements and Evaluation (SQuaRE) - System and software quality models:* <https://www.iso.org/standard/35733.html> Acesso em: 16 de ago. de 2019.

tempos de processamento, taxa de transferência do produto ou sistema, enquanto realiza suas funções, atende seus requisitos;

Do quesito funcionalidade também dividido em várias características será avaliada a sua **completude funcional** que trata do grau com que o conjunto de funções cobrem todas as tarefas especificadas e objetivos dos usuários.

Do quesito confiabilidade que é subdividido em categorias será avaliada a sua **disponibilidade** que trata do grau com que o produto ou sistema é operacional e acessível quando requerido seu uso.

Através do questionário avaliou-se a viabilidade de se utilizar da fechadura eletrônica controlada por aplicativo de celular para se fazer acesso em ambientes aplicando assim o conceito de transformar uma propriedade em uma *smart property*.

O questionário é composto de 19 perguntas com possibilidades de resposta de acordo com a escala *Likert*[2] em que as possibilidades de resposta variam de 1 a 5, sendo 1 - Discordo Totalmente, 2 - Discordo parcialmente, 3 - Indiferente, 4 - Concordo parcialmente e 5 - Concordo Totalmente. Também é apresentada uma questão extra onde o participante pudesse dar sua opinião em relação ao projeto como um todo.

No total foram nove pessoas que responderam ao questionário, sendo quatro do curso de Sistemas de Informação, um de Ciências da Computação, um de Engenharia da Computação, dois de Engenharia de Produção e um de Engenharia Eletrônica. Nenhum deles já havia usado sistemas que utilizassem a tecnologia *blockchain* e todos responderam como nível Iniciante em conhecimento da tecnologia dentre as opções Iniciante, Intermediário e Avançado. Nenhum participantes conheciam o autor desse trabalho o que garante que não há um viés para avaliar positivamente ou negativamente o projeto.

Os resultados serão apresentados separadamente em seções na seguinte ordem: usabilidade, aplicabilidade, motivação, segurança, eficiência e confiabilidade.

6.1 Usabilidade

O atributo usabilidade diz respeito a várias características das quais serão avaliadas a operabilidade, o *design* da interface de usuário do sistema e a apreensibilidade. As Questões de 1 a 9 se referem a ela e na Figura 28 é exibido um gráfico de velas (*boxplot*) de suas respostas que foi montado usando os dados da Tabela 4.

[2] Entenda o que é Escala Likert e como aplicá-la: <https://mindminers.com/blog/entenda-o-que-e-escala-likert/> Acesso em: 16 de ago. de 2019

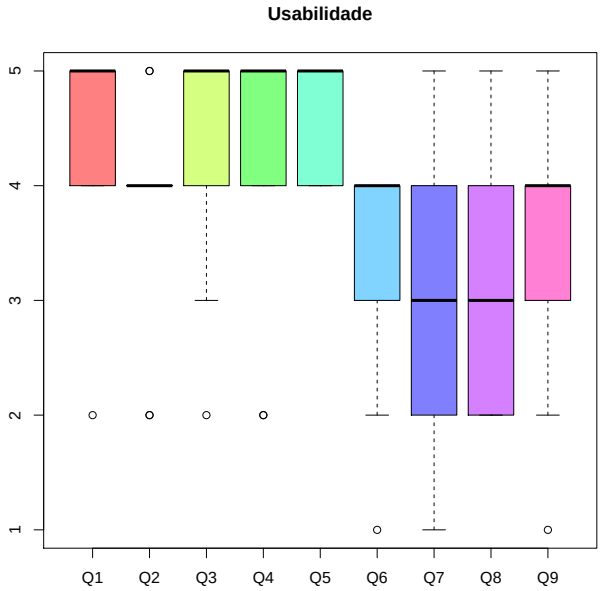

Figura 28 – *Boxplot* das questões de usabilidade.

Tabela 4 – Dados estatísticos de usabilidade.

	Q1	**Q2**	**Q3**	**Q4**	**Q5**	**Q6**	**Q7**	**Q8**	**Q9**
Mínimo	2	2	2	2	4	1	1	2	1
1º Quartil	4	4	4	4	4	3	2	2	3
Mediana	5	4	5	5	5	4	3	3	4
Média	4,44	3,78	4,22	4,11	4,67	3,33	3,11	3,33	3,44
3º Quartil	5	4	5	5	5	4	4	4	4
Máximo	5	5	5	5	5	4	5	5	5
Desvio Padrão	1,06	1,06	1,16	1,31	0,52	1,16	1,46	1,28	1,19
Variância	1,13	1,13	1,36	1,71	0,27	1,36	2,13	1,64	1,41

As Questões Q1 - “Registrar-se no sistema é fácil”, Q3 - “A navegação nos menus do aplicativo é de fácil entendimento e intuitiva” e Q4 - “A navegação nos menus do site é de fácil entendimento e intuitiva”, obtiveram boa avaliação com alto grau de concordância. Isso significa que a navegação tanto no site quanto no aplicativo de celular se encaixa no padrão que os usuários esperam e que já estão acostumados de um sistema desse tipo.

O resultado das questões Q2 - “Cadastrar o endereço da carteira de criptomoedas é fácil” e Q5 - “A interação com as ações referentes às reservas é de fácil entendimento e intuitiva” foi inesperado pois pelo perfil dos participantes do teste nenhum tinha experiência com a tecnologia *blockchain* ou criptomoedas. Isso pode ser resultado direto do treinamento e explicação dado aos participantes. Todas as etapas de interação com o sistema desenvolvido foram detalhadas em uma apresentação que foi fornecida aos participantes.

As questões Q6 - “O uso de uma carteira de criptomoedas para utilização do site na interação com as reservas foi de fácil entendimento e intuitivo” e Q9 - “A utilização de contratos inteligentes para representar informações das reservas ficou clara no sistema” apresentaram avaliações de boas a neutras. Isso indica que há uma necessidade tornar mais clara a interação com as funções das reservas por meio do uso da carteira de criptomoedas e suas transações. Também existe a necessidade de explicar melhor o papel do contrato inteligente ao representar as informações das reservas e verificar autenticidade dos usuários. Como o sistema desenvolvido representa uma mudança nos modelos tradicionais presentes no dia-a-dia é necessário que haja maior clareza nesses detalhes para que seja possível se vender tais produtos e serviços engajamento dos consumidores.

As questões Q7 - “O design do aplicativo é atraente” e Q8 - “O design do site é atraente” obtiveram resultados que variam desde a concordância parcial à discordância parcial o que conclui-se que as interfaces devem ser melhoradas pois isso é um fator importante na experiência do usuário e o fato do sistema possuir mais um caráter de teste influenciou nesse quesito.

6.2 Satisfação

O atributo satisfação se divide em várias características das quais serão avaliadas o conforto, a confiança e a utilidade. As questões de 10 a 16 se referem a esse atributo e na Figura 29 é exibido o seu *boxplot* de suas respostas que foi montado com os dados da Tabela 5.

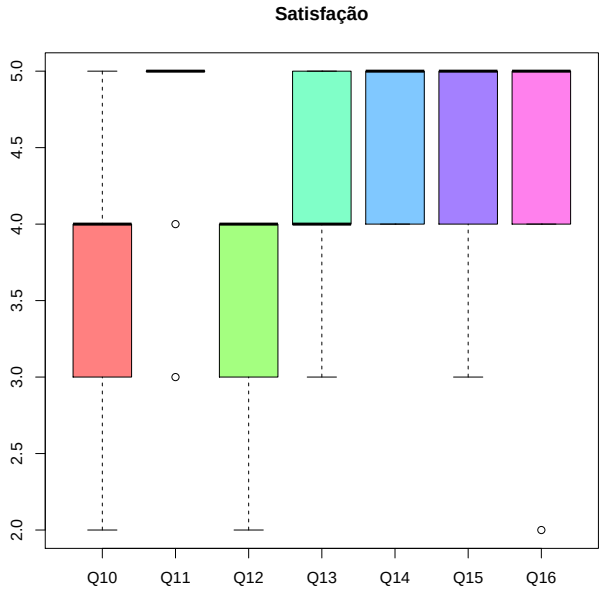

Figura 29 – *Boxplot* das questões de satisfação.

Tabela 5 – Dados estatísticos de satisfação.

	Q10	**Q11**	**Q12**	**Q13**	**Q14**	**Q15**	**Q16**
Mínimo	2	3	2	3	4	3	2
1º Quartil	3	5	3	4	4	4	4
Mediana	4	5	4	4	5	5	5
Média	3,78	4,67	3,44	4,22	4,67	4,33	4,44
3º Quartil	4	5	4	5	5	5	5
Máximo	5	5	4	5	5	5	5
Desvio Padrão	0,92	0,74	0,74	0,64	0,52	0,89	1,06
Variância	0,84	0,55	0,55	0,41	0,27	0,79	1,13

A Questões Q10 - "O uso de um aplicativo no celular para acessar um recinto é mais atrativo queo uso de chaves" obteve avaliação de boa a neutra o que pode ter sido reflexo de ter que desbloquear o celular, abrir a aplicação, navegar pelos menus e encontrar a reserva desejada. O que é um processo que envolve mais passos do que simplesmente pegar as chaves e abrir a porta. Isso poderia ser contornado por um sistema que usasse a tecnologia NFC como mencionado na Seção 5.3.

A Questão Q11 - "A utilização de um sistema em que o controle das informações das reservas não está apenas nas mãos do prestador de serviço é mais atrativo que o modelo tradicional" obteve avaliação muito boa com alto grau de concordância pois um sistema onde o controle das informações de reservas não centralizado dá mais segurança ao usuário pois qualquer alteração nelas é transparente e na resolução de conflitos acredita-se na veracidade das informações.

A Questão Q12 - "Controle de acesso por aplicativo no celular trouxe mais segurança" obteve avaliação de boa a neutra o que evidencia uma maior necessidade de explicar o funcionamento do sistema detalhando melhor o papel de cada componente e sua contribuição para a segurança principalmente do contrato inteligente e da *blockchain*.

A Questão Q13 - "Controle das informações das reservas por meio de contrato inteligente trouxe mais segurança" foi bem avaliada com alto grau de condordância pois para alterar as informações referentes a criação do contrato e as interações com as suas funções um agente externo tem que alterar todo o histórico de transações que levaram a esse resultado usando um ataque do tipo "*51% attack*"[3] o que é inviável economicamente.

As Questões Q14 - "Você usaria serviços (viagens, aluguel de sala para reunião, etc) que usassem esse tipo de fechadura eletrônica", Q15 - "O acesso ao espaço físico utilizado, usando esse sistema foi diferenciado e atrativo" e Q16 - "Você voltaria a usar essa solução novamente" obtiveram boas avaliações com alto grau de concordância nos quesitos avaliados. O sistema se mostrou mais atrativo pois além de mais transparente, o que traz segurança quanto as informações, e mais seguro, pelo uso da *blockchain*, também

[3] 51% attack: <https://www.investopedia.com/terms/1/51-attack.asp> Acesso em: 29 de ago. de 2019.

proporciona conforto e isso leva aos participantes declararem que o usariam. O usuário ao viajar ou alugar uma sala de reunião não quer problemas com acesso indevidos, se preocupar com a devolução de chaves e em último caso quer uma resolução de conflitos mais ágil e acertiva.

6.3 Desempenho

O atributo desempenho diz respeito a várias características das quais será avaliada apenas a comportamento com o tempo. A Questão 17 se refere a ela e a Figura 30 exibe o seu *boxplot* montado usando os dados da Tabela 6.

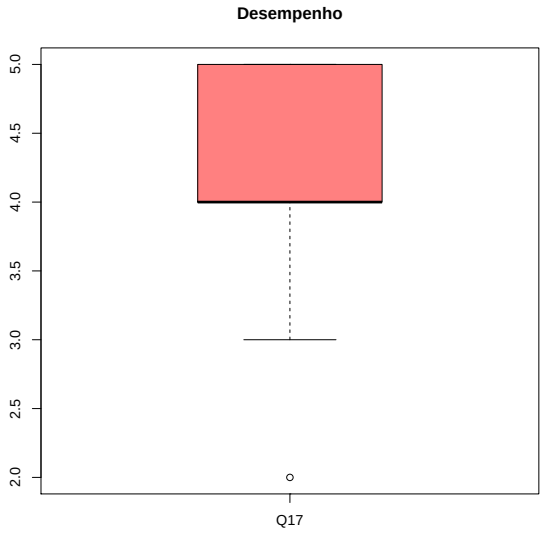

Figura 30 – *Boxplot* dos resultados de desempenho.

Tabela 6 – Dados estatísticos de desempenho.

	Q17
Mínimo	2
1º Quartil	4
Mediana	4
Média	4,11
3º Quartil	5
Máximo	5
Desvio Padrão	1,07
Variância	1,14

A Questão Q17 - “Cadastrar reservas e interagir com as mesmas são tarefas executadas rapida-mente e sem travamento” obteve boa avaliação com alto grau de concordância no quesito avaliado. Mesmo se usando os serviços *web* gratuitos da Amazon

obteve-se um bom desempenho para o sistema desenvolvido. Para um sistema comercial e completo será necessário um sistema com maior desempenho com mais recursos de armazenamento e processamento.

As interações com as funções do contrato inteligente podem ser um problema maior quando implantado em um cenário real em que a rede principal da *blockchain* da Ethereum for utilizada. Na rede principal, o intervalo de tempo entre um bloco e outro dura em média 12 segundos mas pode chegar a 30 segundos como ocorreu nos mêses de Setembro e Outubro de 2017[4].

Para contornar esse problema pode-se desenvolver um sistema onde a interação com os contratos inteligentes ocorra de maneira mais transparente ao usuário onde não seja necessário a espera da confirmação das transações nas interações *client-side* do navegador. Essa interação com o contrato inteligente pode ser feita no *backend* do servidor. Assim o sistema poderia apenas informar a confirmação das transações sem que o usuário fique esperando por isso.

6.4 Funcionalidade

O atributo funcionalidade trata de várias características das quais foi avaliada a completude funcional. A Questão 18 trata deste atributo e na Figura 31 é exibido o seu *boxplot* usando os dados da Tabela 7.

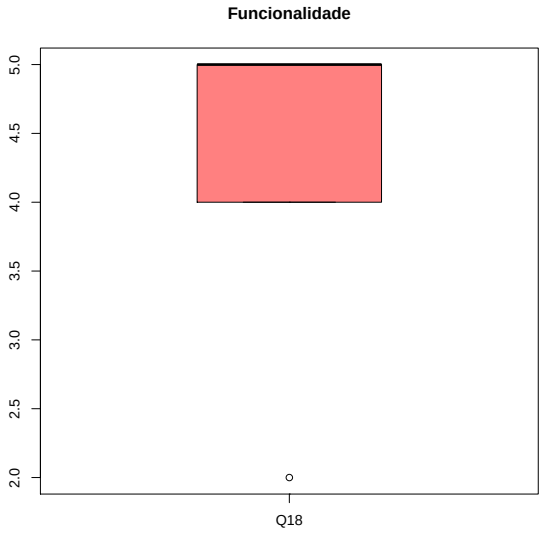

Figura 31 – *Boxplot* da questão de Funcionalidade.

[4] Ethereum Average Block Time Chart: <https://etherscan.io/chart/blocktime> Acesso em: 21 de ago. de 2019.

Tabela 7 – Dados estatísticos de funcionalidade.

	Q18
Mínimo	2
1º Quartil	4
Mediana	5
Média	4,44
3º Quartil	5
Máximo	5
Desvio Padrão	1,06
Variância	1,13

A Questão Q18 - "O sistema mostrou-se eficiente, cumprindo as funcionalidades esperadas" obteve boa avaliação com alto grau de concordância no quesito avaliado. Mesmo não sendo uma versão final voltada para o comércio o sistema desenvolvido obteve boa avaliação em relação ao que propõe de acordo com os avaliadores. Mas para um produto comercializável muitas funcionalidades a mais serão necessárias como, por exemplo, toda a parte de pagamento de reservas.

6.5 Confiabilidade

O atributo confiabilidade engloba muitas características e nessa pesquisa foi avaliada a disponibilidade. A Questão 19 aborda esse atributo.

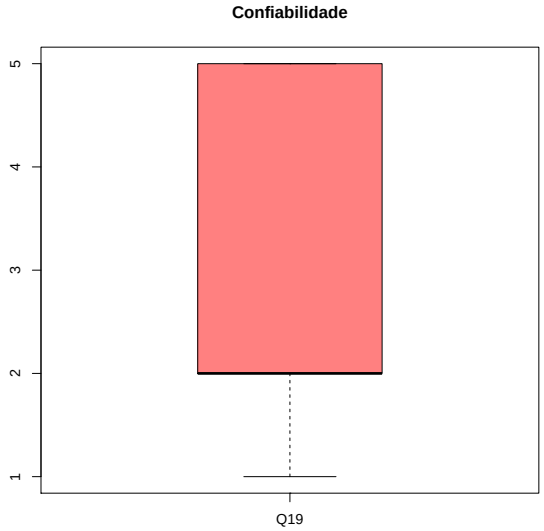

Figura 32 – *Boxplot* da questão de Confiabilidade.

Tabela 8 – Dados estatísticos de confiabilidade.

	Q19
Mínimo	1
1º Quartil	2
Mediana	2
Média	3,11
3º Quartil	5
Máximo	5
Desvio Padrão	1,69
Variância	2,86

A questão Q19 - "O sistema mostrou ter disponibilidade, pois não apresentou falhas ou interrup-ções durante o uso" obteve a pior avaliação de todas com maior nível de discordância do que concordância. Isso pode estar relacionado ao fato de que alguns dos participantes fizeram testes do sistema em momentos diferentes. Uma primeira parte dos alunos fez os teste num momento em que o sistema não apresentou falhas. Mas durante os testes o servidor não estava respondendo de maneira correta e a página da plataforma de reservas não estava carregando nos navegadores dos usuários e uma segunda parte dos envolvidos no teste estavam tentando acesso. Com isso a avaliação desse grupo pode ter sido afetada mas os verdadeiros motivos necessitam de melhor investigação para serem identificados. Isso evidencia que esse tipo de sistema deve estar sempre disponível e não apresentar falhas.

Na Tabela 9, os resultados da análise de qualidade de software estão sintetizados.

Tabela 9 – Quadro de atributos avaliados e resultados sintetizados.

Atributo	**Resultado sintetizado**
Usabilidade	A interação com o site e o aplicativo é fácil. É necessário esclarecer a interação das reservas com a carteira de criptomoedas e o papel dos componentes do projeto. As interfaces de usuário precisam ser melhoradas.
Satisfação	Um sistema mais transparente é mais atrativo para os usuários. O projeto foi considerado útil atendendo as necessidades do usuário. O processo de acessar a propriedade precisa se desburocratizado. E é necessário esclarecer a contribuição de cada componente do sistema para a segurança
Desempenho	O sistema apresentou bom desempenho mesmo sendo um protótipo e usando serviços grátuitos. Uma versão comercial deve ser mais eficiente principalmente na interação com a blockchain.
Funcionalidade	O projeto realiza as funcionalidades mínimas esperadas.
Confiabilidade	Um sistema de reservas e de controle de acesso deve estar sempre disponível e não pode apresentar falhas.

6.6 Validade do estudo

Dados os resultados desse estudo cabe uma discussão sobre a validade das conclusões apresentadas nele. Ao se tomar conclusões em uma pesquisa, sua validade pode ser dividida em 4 categorias (WOHLIN et al., 2012) e cada uma delas apresenta ameaças:

- **Validade da conclusão:** preocupa-se com a relação entre o tratamento realizado e o resultado. Deseja-se ter certeza de uma relação estatística;
- **Validade interna:** se é observado uma relação entre o tratamento e o resultado, deseja-se que ela seja causal e que a causa não seja de algo que não se possa controlar ou algo que não se mensurou;
- **Validade do constructo:** se preocupa com a relação entre a teoria e a observação;
- **Validade externa:** preocupa-se com a generalização das conclusões.

6.6.1 Ameaças à validade

A forma com que o experimento descrito no início do Capítulo 6 foi estruturado e o perfil participantes que foram convidados para os testes podem trazer ameaças a validade das conclusões apresentadas. Aqui serão discutidas sucintamente aquelas que foram identificadas pelo autor.

As ameaças à Validade de Conclusão se tratam de problemas que afetam a habilidade de se obter as conclusões corretas sobre a relação entre tratamento e resultados. As que devem ser mencionadas são o **Baixo poder estatístico** e o **Procura por um resultado específico**. Como o autor optou por análise de gráficos *boxplot* sem domínio aprofundado em estatística as conclusões podem ter sido afetadas pois isso é uma ameaça ao poder do teste estatístico em revelar padrões nos dados. E ao se procurar por um resultado específico (a viabilidade do projeto) o autor pode ter influenciado nos resultados.

Tratando-se da Validade Interna suas ameaças são influencias que podem afetar as variaveis independentes em relação a causalidade. As possíveis identificadas foram o **Momento de aplicação do experimento** e a **Instrumentação**. Os testes foram aplicados aos participantes, alunos de graduação da UNIFEI, em época de avaliações e isso pode ter afetado negativamente os resultados por conta de suas atividades acadêmicas que influenciam tanto no seu estado emocional quanto no tempo hábil disponível para os testes. Para coleta de dados da opnião dos participantes foi utilizado um questionário disponibilizado no Google Forms[5], e caso este tenha sido mal planejado e construido pode

[5] Formulários Google: crie e analise pesquisas gratuitamente: <https://www.google.com/forms/about/> Acesso em: 20 de ago. de 2019

afetar negativamente o experimento pois questões mal formuladas ou muito similares podem dificultar a compreensão dos participantes.

A Validade do Constructo preocupa-se com a generalização dos resultados ao conceito ou teoria por trás do experimento. As possíveis ameaças identificadas foram **Explicação pré-operacional dos constructos inadequada** e a **Expectativa do participante**. Como ficou claro nos resultados obtidos nas questões Q6 - "O uso de uma carteira de criptomoedas para utilização do site na interação comas reservas foi de fácil entendimento e intuitivo" e Q9 - "A utilização de contratos inteligentes para representar informações das reservas ficou clara no sistema" não deixar mais claro para os participantes o motivo do uso da carteira de criptomoedas e como operá-la; e descrever melhor o papel do contrato inteligente em representar as informações das reservas, como a fechadura o acessa para verificar autencidade dos acessos e como seu uso torna as informações do sistema mais transparentes aos usuários pode ter afetado negativamente a avaliação desses pontos no projeto. Participantes com expectativas prévias aos experimentos também podem influenciar negativamente suas respostas. Um participante que não tiver suas expectativas atendidas pode avaliar os pontos estudados de maneira negativa.

Ameaças a Validade Externa são condições que limitam a habilidade de generalizar os resultados do experimento a pratica industrial. As possíveis ameaças identificadas são a **Interação da seleção com o tratamento**, que é o efeito de se ter uma população de participantes não representante daquela que se quer fazer uma generalização; a **Interação do contexto com tratamento**, que é o efeito de um contexto ou matéria do experimento não representativo de um contexto real; e a **Interação do histórico com o tratamento**, que é o efeito de se conduzir o experimento em um dia que afete os resultados.

A Interação da seleção com o tratamento apresenta-se no perfil dos participantes pois não é um que representa em sua totalidade aquele que se espera de possíveis consumidores desse tipo de serviço ou produto. Todos eram jovens estudantes de vários cursos de engenharia, de ciências da computação e sistemas da informação na Universidade Federal de Itajubá. São pessoas que têm mais contato com tecnologia e inovação o que pode ter se expressado em maior aceitação do projeto desenvolvido. Quando selecionado uma população mais heterogenea os resultados seriam outros evidenciando de melhor maneira o que pode ser melhorado, adaptado, retirado ou adicionado ao sistema. A influência da Interação do contexto com o tratamento evidencia-se no fato que os participantes estavam apenas testando a fechadura sem acessar a sala do Prof. Dr. Adler com o mesmo objetivo similar ao contexto de uma aplicação real, que é acessar uma propriedade alugada e isso pode afetar os resultados obtidos. Um teste em um ambiente com o mesmo objetivo traria resultados mais precisos podendo evidenciar fatos que não foram identificados aqui nesse trabalho. A influência da Interação do histórico com o tratamento fica evidenciada no fato de um segundo grupo ter feito testes no momento que o sistema apresentou falhas.

Isso obviamente influenciou negativamente os resultados o que foi evidenciado na questão Q19 - “O sistema mostrou ter disponibilidade, pois não apresentou falhas ou interrupções durante o uso”.

7 Conclusão

Para combinar o uso de tecnologias atuais com uma contribuição acadêmica significativa optou-se por estudar a viabilidade de aplicar o conceito de *smart property* utilizando contratos inteligentes implantados na rede *blockchain* da Ethereum para controle de acesso em propriedades nos moldes dessa tecnologia.

O sistema desenvolvido foi resultado da combinação de várias tecnologias diferentes, desde aplicativo de celular a infra-estrutura de servidores, que necessitavam se intercomunicar para seu bom funcionamento. De todas a mais disruptiva foi a *blockchain* que na sua aplicação mais básica realiza a transferência de valor entre usuários mas o faz fora dos moldes tradicionais sem intermediadores de confiança no processo. E devido às suas atrativas características possibilita muitas outras aplicações das quais foi utilizado o contrato inteligente.

Para analisar a viabilidade do projeto desenvolveu-se um questionário que necessitou ser refinado e reorganizado para que as questões representassem os características corretamente. O engajamento dos alunos participantes tanto em testar a fechadura quanto em respondê-lo foi problemático pois coincidiu com o período de provas mas o grupo atingiu um número suficiente para análise realizada.

Analisando os resultados e as respostas ao questionário com ferramentas estatísticas pode-se concluir que o conceito de *smart property* é viável. Os contratos inteligentes mostraram-se eficientes em representar as informações das reservas e mostrar transparência do sistema aos usuários. Analisando as questões de pesquisa o sistema desenvolvido é viável apresentando mais atratividade que os moldes tradicionais fechadura, atendeu as funcionalidades esperadas e proporcionou conforto para aqueles que o utilizaram.

Para trabalhos futuros pode-se estudar o uso do sistema para aluguel de propriedades fora do âmbito universitário, como casas, quartos de hotel ou salas de reunião, podendo eliminar assim o viés dos participantes. Isso levando em consideração o que foi discutido na Seção 6.6.1 para melhorar os resultados coletados e diminuir as ameaças as conclusões que serão obtidas. Sobre o sistema em si há a possibilidade de aprimorar sua segurança usando-se de outros métodos de autenticação do usuário diante da fechadura. Poderia-se utilizar da autenticação de duas etapas juntamente com o protocolo para eliminar o problema com a gravação do áudio emitido pelo celular dos usuários ou optar pelo usa da tecnologia NFC dificultando que outros dispositivos acessem pois é difícil clonar aquele que é autorizado. Alinhada a essa ultima, propoe-se a análise de viabilidade economica da utilização de tecnologias como o NFC comparadas ao uso do protocolo de utilizado nessa dissertação, o Chirp.io. Sobre a segurança física do projeto, estudar a influência do um

campo eletromagnético intenso ao lado da tranca solenoide observando se é possível abrí-la. Avaliar também a possibilidade de acoplar os contratos inteligentes e a controladora à fechaduras já existentes no mercado e que são conhecidas por sua robustez na sua implementação como um mecanismo de segurança adicional. Para obter mais engajamento dos usuários, tornar o sistema blockchain mais transparente sem a necessidade de instalar uma carteira, registrá-la e autorizar cada uma das transações passando toda essa logística para o *back-end* pode ser algo a ser desenvolvido no futuro uma vez que nem todos os que testaram o sistema concordaram que essas tarefas eram fáceis e claras. Pode-se desenvolver adicionar ao sistema desenvolvido nessa dissertação uma funcionalidade que gere um contrato digital, além do contrato inteligente, que defina as cláusulas que regem a reserva de uma maneira mais similar ao tradicional, que deva se assinado digitalmente, onde cada parte fique com uma cópia e avaliar o nível de engajamento dos usuários em um sistema ao qual estão mais habituados. Além desses pontos o estudo da viabilidade do uso de contratos inteligentes na venda transferindo a posse da propriedade pode vir a ser tema de pesquisas futuras englobando uma pesquisa em conjunto com a área de direito.

7.1 Contribuições

A pesquisa descrita nessa dissertação também obteve outras contribuição:

- Produziu-se um artigo que foi aceito para apresentação na conferência *16th International Conference on Information Technology : New Generations* - ITNG 2019[1] com classificação B1 na Qualis CAPES e cujo artigo foi publicado como capítulo de livro nos anais da conferência.
- Gerou-se seu Registro de Software desenvolvido no trabalho através do INPI[2] com o código fonte das principais linguagens utilizadas nele.
- Pediu-se a patente pelo projeto desenvolvido.

[1] INTG 2019: <http://www.itng.info/OLD/2019Web/index.php>
[2] INPI - Instituto Nacional de Propriedade Industrial: <http://www.inpi.gov.br/>

Referências

ACHESON, N. *What is smart property, and what can it be used for?* 2018. Disponível em: <http://www.fintechblue.com/2015/12/smart-property-what-does-that-mean-for-the-blockchain/>. Acesso em: 04.10.2018. Citado na página 33.

ANTONOPOULOS, A. M. *Mastering Bitcoin: Unlocking Digital Crypto-Currencies*. 1st. ed. [S.l.]: O'Reilly Media, Inc., 2014. ISBN 1449374042, 9781449374044. Citado 4 vezes nas páginas 28, 29, 30 e 31.

ANTONOPOULOS, A. M.; WOOD, G. *Mastering ethereum: building smart contracts and dapps*. [S.l.]: O'Reilly Media, 2018. Citado na página 31.

ATZEI, N.; BARTOLETTI, M.; CIMOLI, T. A survey of attacks on ethereum smart contracts (sok). In: SPRINGER. *International Conference on Principles of Security and Trust*. [S.l.], 2017. p. 164–186. Citado na página 59.

BACH, L.; MIHALJEVIC, B.; ZAGAR, M. Comparative analysis of blockchain consensus algorithms. In: IEEE. *2018 41st International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*. [S.l.], 2018. p. 1545–1550. Citado na página 30.

BECK, R. et al. Blockchain technology in business and information systems research. *Business & Information Systems Engineering*, v. 59, n. 6, p. 381–384, Dec 2017. ISSN 1867-0202. Disponível em: <https://doi.org/10.1007/s12599-017-0505-1>. Citado na página 27.

BOLTON, P.; DEWATRIPONT, M. et al. *Contract theory*. [S.l.]: MIT press, 2005. Citado na página 32.

BURNISKE, C.; TATAR, J. *Cryptoassets: The Innovative Investor's Guide to Bitcoin and Beyond*. [S.l.]: McGraw Hill Professional, 2017. Citado 3 vezes nas páginas 27, 29 e 31.

BUTERIN, V.; GRIFFITH, V. Casper the friendly finality gadget. *arXiv preprint arXiv:1710.09437*, 2017. Citado na página 31.

BUTERIN, V. et al. A next-generation smart contract and decentralized application platform. *white paper*, 2014. Citado na página 33.

CHRISTIDIS, K.; DEVETSIKIOTIS, M. Blockchains and smart contracts for the internet of things. *Ieee Access*, Ieee, v. 4, p. 2292–2303, 2016. Citado 2 vezes nas páginas 30 e 32.

CRUZ, A. S. *Direito Empresarial*. 8th. ed. [S.l.]: Editora Forense LTDA, 2018. Citado na página 27.

D'ALIESSI, M. *How Does Ethereum Work?* Medium, 2018. Disponível em: <https://medium.com/@micheledaliessi/how-does-ethereum-work-8244b6f55297>. Citado na página 31.

DANNEN, C. *Introducing Ethereum and Solidity*. [S.l.]: Springer, 2017. Citado 2 vezes nas páginas 31 e 33.

ERICKSON, J. *Hacking: the art of exploitation*. [S.l.]: No starch press, 2008. Citado na página 60.

FLOYD, T. L. *Electronic devices: conventional current version*. [S.l.]: Pearson, 2012. Citado 3 vezes nas páginas 47, 48 e 49.

HILDENBRANDT, E. et al. Kevm: A complete formal semantics of the ethereum virtual machine. In: *2018 IEEE 31st Computer Security Foundations Symposium (CSF)*. [S.l.: s.n.], 2018. p. 204–217. Citado na página 33.

HOROWITZ, P.; HILL, W. *The art of electronics*. 3rd. ed. [S.l.]: Cambridge University Press, 2015. Citado na página 47.

KUROSE, J. F.; ROSS, K. W. *Computer networking: a top-down approach*. [S.l.]: Addison Wesley, 2007. Citado 2 vezes nas páginas 60 e 61.

NAKAMOTO, S. et al. Bitcoin: A peer-to-peer electronic cash system. Working Paper, 2008. Citado 3 vezes nas páginas 27, 29 e 33.

NOFER, M. et al. Blockchain. *Business & Information Systems Engineering*, Springer, v. 59, n. 3, p. 183–187, 2017. Citado na página 29.

PEE, S. J. et al. Blockchain based smart energy trading platform using smart contract. In: *2019 International Conference on Artificial Intelligence in Information and Communication (ICAIIC)*. [S.l.: s.n.], 2019. p. 322–325. Citado na página 33.

POPPER, N. *Business Giants to Announce Creation of a Computing System Based on Ethereum*. The New York Times, 2017. Disponível em: <https://www.nytimes.com/2017/02/27/business/dealbook/ethereum-alliance-business-banking-security.html>. Citado na página 31.

SABZEVAR, A. P.; STAVROU, A. Universal multi-factor authentication using graphical passwords. In: IEEE. *2008 IEEE International Conference on Signal Image Technology and Internet Based Systems*. [S.l.], 2008. p. 625–632. Citado na página 62.

SILBERSCHATZ, A. et al. *Database system concepts*. 6th. ed. [S.l.]: McGraw-Hill New York, 2010. Citado na página 54.

STEVENS, W. R. *TCP/IP illustrated vol. I: the protocols*. [S.l.]: Pearson Education, Inc., 2012. Citado na página 61.

SWAN, M. *Blockchain: Blueprint for a New Economy*. 1st. ed. [S.l.]: O'Reilly Media, Inc., 2015. ISBN 1491920491, 9781491920497. Citado 5 vezes nas páginas 24, 27, 30, 32 e 33.

SZABO, N. Smart contracts: building blocks for digital markets. *EXTROPY: The Journal of Transhumanist Thought,(16)*, v. 18, 1996. Citado na página 33.

TAPSCOTT, D.; TAPSCOTT, A. *Blockchain Revolution: How the Technology Behind Bitcoin Is Changing Money, Business, and the World*. [S.l.]: Portfolio, 2016. ISBN 1101980133, 9781101980132. Citado 4 vezes nas páginas 27, 30, 31 e 32.

WANT, R. Near field communication. *IEEE Pervasive Computing*, IEEE, n. 3, p. 4–7, 2011. Citado na página 62.

WASSERMANN, G.; SU, Z. Static detection of cross-site scripting vulnerabilities. In: IEEE. *2008 ACM/IEEE 30th International Conference on Software Engineering*. [S.l.], 2008. p. 171–180. Citado na página 61.

WOHLIN, C. et al. *Experimentation in software engineering*. [S.l.]: Springer Science & Business Media, 2012. Citado na página 74.

WOOD, G. et al. Ethereum: A secure decentralised generalised transaction ledger. *Ethereum project yellow paper*, v. 151, p. 1–32, 2014. Citado na página 31.

XU, Y.-H.; PENNINGTON-GRAY, L. et al. Explore the spatial relationship between airbnb rental and crime. 2017. Citado na página 23.

YLI-HUUMO, J. et al. Where is current research on blockchain technology?—a systematic review. *PLOS ONE*, Public Library of Science, v. 11, n. 10, p. 1–27, 10 2016. Disponível em: <https://doi.org/10.1371/journal.pone.0163477>. Citado 4 vezes nas páginas 24, 27, 29 e 30.

ZHANG, Y.; WEN, J. The iot electric business model: Using blockchain technology for the internet of things. *Peer-to-Peer Networking and Applications*, Springer, v. 10, n. 4, p. 983–994, 2017. Citado na página 33.

ZHAO, J. L.; FAN, S.; YAN, J. Overview of business innovations and research opportunities in blockchain and introduction to the special issue. *Financial Innovation*, v. 2, n. 1, p. 28, Dec 2016. ISSN 2199-4730. Disponível em: <https://doi.org/10.1186/s40854-016-0049-2>. Citado na página 30.

Apêndices

APÊNDICE A – Especificação de rotas da API

Para *login*, definiu-se que uma requisição do tipo POST deve ser feita para o endereço http://13.59.232.1/api/login. Na requisição, devem constar email e password do usuário. Essa requisição chama a execução da função *login* do controlador `AuthController`. Nela são validados o email e password e caso corresponda com algum usuário cadastrado é enviado um token de acesso do tipo `Bearer`. Esse token que deve ser anexado em qualquer uma das outras requisições disponíveis pois ele é usado para identificar a identidade de cada um dos usuários. Isso se deve ao fato de o acesso por API de autenticação não funcionar como o acesso ao website pelo navegador que é feito por sessões.

O logout é feito com uma requisição do tipo *GET* para o endereço http://13.59.232.1/api/logout. Essa requisição chama a execução da função logout. Nela o token anexado a requisição é revogado e é enviada a mensagem *"You have been successfully logged out"* como resposta.

A aquisição das informações das reservas foi efetuada com uma requisição do tipo get para o endereço http://13.59.232.1/api/reservations. Ela executa a função `getReservations`. Para obter as informações das reservas o Laravel oferece uma camada de transformação que fica entre os modelos do Eloquent e as respostas JSON que são enviadas para o aplicativo dos usuários. Isso é realizado com classes do tipo resource que permitem transformar modelos e coleções de modelos em JSON. Com esse recurso a função envia uma coleção de reservas na forma de JSON. De cada uma das reservas são enviados seu `id, user_id, data_ini, data_fin` e `ctr_addr`.

APÊNDICE B – Configuração da comunicação com a *blockchain* e implantação do contrato inteligente

Para configurar a comunicação com a rede *blockchain* um Http Provider para intermediar a comunicação precisa ser especificado e o escolhido foi o fornecido pela extensão MetaMask. Em seguida é necessário informar o ABI descrito como uma estrutura JSON e o EVM (*Ethereum Virtual Machine*) *bytecode* em hex do contrato inteligente. É criada uma classe proxy de contrato usando o comando eth.contract e a ABI. Após isso uma transação é criada usando a classe proxy de contrato e o EVM bytecode em hex seguido dos argumentos do construtor e ela é assinada usando a private key do próprio usuário.

Quando o comando `eth.contract` é executado, se não houver nenhum erro o objeto `contract` está instanciado. Portanto, terá o seu endereço disponível e para salvá-lo no banco de dados no servidor usa-se a função *JavaScript ajax*[1] que faz uma requisição HTTP assincrona para o endereço http://13.59.232.1/reservation/rid/contract-address. Assim, o sistema armazena o endereço do contrato inteligente que representa a reserva que acabou de ser aprovada.

[1] jQuery.ajax() | jQuery API documentation: <https://api.jquery.com/jquery.ajax/>

APÊNDICE C – JSON exemplo das informações das reservas


```
{"data":[{
  "id":39,"user_id":1,"room_id":1,"data_ini":"2019-05-22",
  "data_fin":"2019-05-24",
  "ctr_addr":"0xdd67059eaa3f9395da7d04cfc0060625e4f27539"},
{
  "id":40,"user_id":1,"room_id":1,"data_ini":"2019-06-02",
  "data_fin":"2019-06-05",
  "ctr_addr":"0x271b209e20e48ceca2ef905594a07889368d58c8"}
]}
```


O resultado da requisição precisa ser processado para listar as reservas. Chama-se uma função que transforma esse resultado em uma *ArrayList* de objetos do tipo Reserva. Com essa *ArrayList* de objetos Reserva fica mais simples listar as informações.

APÊNDICE D – Configurações mínimas de segurança da controladora Raspberry Pi

Antes de desabilitar a conta padrão `pi` deve-se criar uma outra primeiro especificando um novo diretório `home` com o seguinte comando:

```
sudo useradd --groups sudo -m newUserName
```

O marcador -m faz com que o diretório *home* seja criado e –groups sudo adiciona a conta ao grupo *sudo* para que o usuário use o comando `sudo` para executar operações que necessitam de permissões especiais. Uma vez criada a conta é necessário configurar uma nova senha para ela. Para isso usamos a próxima linha de comando:

```
sudo passwd newUserName
```

Em seguida, será necessário ajustar a senha do usuário *root* para algo mais seguro. Para isso o seguinte comando:

```
sudo passwd root
```

E por fim, deve-se desabilitar a conta *pi* com a seguinte linha de comando:

```
sudo passwd --lock pi
```

Com isso garante-se que uma pessoa má intencionada não consiga ter acesso à fechadura usando métodos muito simples por falta de configurações mínimas de segurança.

Colocar *sudo* na frente de um comando o executa como *superuser*, e por padrão isso não necessita de senha. Entretanto se alguém não autorizado acessar a Raspberry Pi poderá executar comandos com os privilégios desse perfil se não for necessário informar senha quando se usa *sudo*.

Para forçar o *sudo* a requerer senha pode-/se usar o seguinte comando mudar a registro do usuário criado para *newUserName ALL=(ALL) PASSWD: ALL*.

```
sudo nano /etc/sudoers.d/010\_pi-nopasswd
```

O próximo passo é bloquear o acesso indevido ao SSH (*Secure Shell*). Para isso pode-se desabilitar a habilidade de que um usuário se conecte como *root* e configurar o SSH para que apenas máquinas com uma chave SSH possam se conectar. Para isso deve-se alterar o conteúdo do arquivo *sshd_config* usando qualquer editor de texto. O conteúdo do arquivo deve ser o seguinte:

```
# Authentication:
LoginGraceTime 120
PermitRootLogin no
StrictModes yes

RSAAuthentication yes
PubkeyAuthentication yes
AuthorizedKeysFile      %h/.ssh/authorized\_keys

# To enable empty passwords, change to yes (NOT RECOMMENDED)
PermitEmptyPasswords no

# Change to yes to enable challenge-response passwords (beware issues with
# some PAM modules and threads)
ChallengeResponseAuthentication no

# Change to no to disable tunnelled clear text passwords
PasswordAuthentication no

UsePAM no
```

A última linha é muito importante já que desabilita o uso de *Pluggable Authentication Modules* (PAM), ou a autenticação Linux nativa e apenas permitir conexão com uma chave. Agora é necessário gerar uma chave SSH. Para isso pode-se usar o comando *ssh-keygen*. Esse comando precisa de um arquivo para salvar as informações da chave. Então pode-se criar um novo diretório no *home* com o nome *.ssh* e nele criar este arquivo com o nome *authorized_keys*. Quando o comando *ssh-heygen* é necessário informar o arquivo criado. Em seguida é necessário reiniciar o SSH para assegurar que as mudanças sejam aplicadas usando a seguinte linha de comando:

```
systemctl restart ssh
```

Uma medida adicional que pode ser tomada também é configurar uma tarefa para ser executada diariamente para atualizar o servidor SSH e assegurar que a Raspberry Pi

terá os últimos ajustes de segurança do SSH prontamente. Isso pode ser feito através do agendador de tarefas chamado Cron. O próximo comando pode ser configurado para ser executado diariamente através dessa ferramenta:

```
sudo install openssh-server
```

Uma vez bloqueado o acesso indevido do SSH pode-se assegurar que o *firewall iptables* está instalado e sendo executando. Para boas medidas adicionais pode-se configurar que o *firewall* registre mensagens sempre quando uma regra sua é ativada e uma conexão é bloqueada. Para instalá-lo usa-se a linha de comando a seguir:

```
sudo apt-get install iptables iptables-persistent
```

Para que se possa alterar as regras o *firewall* pode-se usar a próxima linha de comando e salvá-las no arquivo *rules.v4* que é utilizado pelo sistema quando se inicia ou reinicia para garantir que o *firewall* ainda está executando.

```
sudo /sbin/iptables-save > /etc/iptables/rules.v4
```

O conteúdo do arquivo *rules.v4* deve ser algo similar ao próximo bloco de código.

```
:INPUT ACCEPT [0:0]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [0:0]

# Allows all loopback (lo0) traffic and drop all traffic to 127/8 that
doesn't use lo0
-A INPUT -i lo -j ACCEPT
-A INPUT ! -i lo -d 127.0.0.0/8 -j REJECT

# Accepts all established inbound connections
-A INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT

# Allows all outbound traffic
# You could modify this to only allow certain traffic
-A OUTPUT -j ACCEPT

# Allows SSH connections
# The --dport number is the same as in /etc/ssh/sshd\_config
```

```
-A INPUT -p tcp -m state --state NEW --dport 22 -j ACCEPT

# log iptables denied calls (access via 'dmesg' command)
-A INPUT -m limit --limit 5/min -j LOG --log-prefix "iptables denied:"
--log-level 7

# Reject all other inbound - default deny unless explicitly allowed
policy:
-A INPUT -j REJECT
-A FORWARD -j REJECT

COMMIT
```

APÊNDICE E – Questionário

E.1 Usabilidade

E.1.1 Operabilidade

- Q1) Registrar-se no sistema é fácil.
- Q2) Cadastrar o endereço da carteira de criptomoedas é fácil.
- Q3) A navegação nos menus do aplicativo é de fácil entendimento e intuitiva.
- Q4) A navegação nos menus do site é de fácil entendimento e intuitiva.
- Q5) A interação com as ações referentes às reservas é de fácil entendimento e intuitiva.
- Q6) O uso de uma carteira de criptomoedas para utilização do site na interação com as reservas foi de fácil entendimento e intuitivo.

E.1.2 Design da interface de usuário

- Q7) O design do aplicativo é atraente.
- Q8) O design do site é atraente.

E.1.3 Apreensibilidade

- Q9) A utilização de contratos inteligentes para representar informações das reservas ficou clara no sistema

E.2 Satisfação

E.2.1 Conforto

- Q10) O uso de um aplicativo no celular para acessar um recinto é mais atrativo que o uso de chaves.

E.2.2 Confiança

- Q11) A utilização de um sistema em que o controle das informações das reservas não está apenas nas mãos do prestador de serviço é mais atrativo que o modelo tradicional.
- Q12) Controle de acesso por aplicativo no celular trouxe mais segurança.
- Q13) Controle das informações das reservas por meio de contrato inteligente trouxe mais segurança.

E.2.3 Utilidade

- Q14) Você usaria serviços (viagens, aluguel de sala para reunião, etc) que usassem esse tipo de fechadura eletrônica.
- Q15) O acesso ao espaço físico utilizado, usando esse sistema foi diferenciado e atrativo.
- Q16) Você voltaria a usar essa solução novamente.

E.3 Desempenho

E.3.1 Comportamento com o tempo

- Q17) Cadastrar reservas e interagir com as mesmas são tarefas executadas rapidamente e sem travamento.

E.4 Funcionalidade

E.4.1 Completude Functional

- Q18) O sistema mostrou-se eficiente, cumprindo as funcionalidades esperadas.

E.5 Confiabilidade

E.5.1 Disponibilidade

- Q19) O sistema mostrou ter disponibilidade, pois não apresentou falhas ou interrupções durante o uso.