

Universidade Paulista - UNIP

RENAN LUIS BIANCHINI

SMART CONTRACT PARA RECEITAS MÉDICAS CONTROLADAS

Limeira

2020

Universidade Paulista - UNIP

RENAN LUIS BIANCHINI

**SMART CONTRACT PARA OBTENÇÃO DE RECEITAS MÉDICAS
CONTROLADAS.**

Trabalho de conclusão de curso apresentado à banca examinadora da Faculdade UNIP, como requisito parcial à obtenção do Bacharelado em ciência da computação sob a orientação dos professores Me. Sérgio Eduardo Nunes, Me. Antonio Mateus Locci e Me. Marcos Vinicius Gialdi.

**Limeira
2020**

Universidade Paulista - UNIP

Smart Contract para obtenção de Receitas Médicas Controladas

Trabalho de conclusão de curso apresentado à banca examinadora da Faculdade UNIP, como requisito parcial à obtenção do Bacharelado em ciência da Computação sob a orientação dos professores Me. Sérgio Eduardo Nunes, Me. Antonio Mateus Locci e Me. Marcos Vinicius Gialdi.

Aprovada em 26 de novembro de 2020.

BANCA EXAMINADORA

DEDICATÓRIA

Dedico este trabalho aos meus colegas, amigos, familiares e professores, o qual estavam ao meu lado dando apoio durante esta trajetória acadêmica, para que eu crescesse como pessoa e profissional durante a graduação.

RESUMO

Este trabalho visa demonstrar a melhoria no fluxo de emissão do receituário médico, de maneira fácil e segura, de modo que o profissional da saúde consiga emitir receitas médicas digitais as quais serão disponibilizadas em uma plataforma de consulta para garantir que um certo paciente tenha acesso ao medicamento necessário, atualmente as receitas sofrem com uma certa burocracia, na qual é preciso da assinatura do médico, tipo do problema, número de protocolos para rastreio, entre outros dados. Ao garantir que uma plataforma consiga validar a assinatura digital do médico e disponibilizá-lo para consulta pelo mercado de fármacos, seria possível que um paciente consiga fazer a compra de seu medicamento online, além de garantir um monitoramento dos mesmos assim evitando fraudes, pois assim que a receita é gerada no blockchain a mesma não será alterada a menos que seja de maneira programada e compilada pelo desenvolvedor do smart contract, evitando que estes dados sejam manipulados e também facilitando auditorias pelos órgãos regulamentadores, pois o dados sendo persistido e validado dentro do blockchain no banco de dados descentralizado, isto torna mais acessível ao auditor.

Palavra-Chave: Blockchain; Smart Contract; Área da Saúde; Assinatura digital.

ABSTRACT

This work aims to improve the flow of issuing medical prescriptions, the easiest and safest way, the way the health professional repairs digital prescription prescriptions and which are made available on a consultation platform to ensure that a certain patient has access to the medication used, currently as prescriptions used with a certain bureaucracy, in which the doctor's signature is required, type of problem, number of screening protocols, among other data. By ensuring that a platform is able to validate a doctor's digital signature and make it available for consultation by the drug market, it is possible for a patient to consider making a purchase of his medicine online, in addition to ensuring monitoring of the same factors that can harm, as soon as the medical prescription is generated on the blockchain it will not be changed unless it is programmed and compiled by the developer of the smart contract, preventing this data from being manipulated and also facilitating audiences by the regulatory bodies, as the data being persisted and validated within the blockchain with a decentralized database, it becomes more accessible to the auditor.

Key Words: Blockchain; Smart Contract; Health Area; Digital Signature.

LISTA DE FIGURAS

Figura 1 – Grafo de um Sistema Distribuído	18
Figura 2 – Representação de chave pública e privada	20
Figura 3 – Representação do algoritmo de Hashing	21
Figura 4 – Exemplo de um Hashing de texto	22
Figura 5 – Demonstrativo da rede Ethereum	24
Figura 7 – Receituário médico branco especial	30
Figura 8 – Receituário médico azul	31
Figura 9 – Receituário médico amarelo.....	31
Figura 10 – Diagrama de caso de uso	34
Figura 11 – Diagrama da arquitetura do projeto.....	35
Figura 12 – Diagrama da arquitetura do projeto com tecnologias utilizadas.....	37
Figura 13 – Diagrama de serviço do medico.....	38
Figura 14 – Diagrama de serviço do paciente.....	39
Figura 15 – Diagrama de serviço da farmácia.....	40
Figura 16 – Diagrama de serviço da Organização	40
Figura 17 – Criação da rede blockchain local com Ganache	41
Figura 18 – Menu do Ganache.....	42
Figura 19 – Contas na rede Ehereum local.....	42
Figura 20 – Transações na rede Ethereum.....	43
Figura 21 – Transações na rede Ethereum.....	44
Figura 22 – Contrato implementado no Ganache	44
Figura 23 – Estrutura de pastas do Smart Contract.....	45
Figura 24 – Configuração da rede Blockchain local	46
Figura 25 – Estrutura do Smart Contract.....	47

Figura 26 – Mapeamento das Estruturas	48
Figura 27 – Métodos do Smart Contract	49
Figura 28 – Representação parcial do Bytecode	50
Figura 29 – Instrução para implementação do Smart Contract.....	51
Figura 30 – Arquitetura do Angular 2+	54
Figura 31 – Tela de Login.....	55
Figura 32 – Tela de Login do médico	55
Figura 33 – Tela de Login da loja do usuário farmácia	56
Figura 34 – Tela de Login do usuário farmácia	57
Figura 35 – Tela de Login do usuário paciente	57
Figura 36 – Tela de Login do usuário Admin.....	58
Figura 37 – Tela de prescrição da receita médica	59
Figura 38 – Tela de profile do médico.	59
Figura 39 – Tela de pedidos da farmácia	60
Figura 40 – Tela de listagem das farmácias.....	61
Figura 41 – Tela de compra da receita.....	62
Figura 42 – Tela de listagem de médicos.....	63
Figura 43 – Tela de listagem de receitas por médico.....	64
Figura 44 – Tela de detalhes da receita médica	65
Figura 45 – Diagrama de gateway do sistema distribuído	67
Figura 46 – Estrutura de pastas do sistema auxiliar	68
Figura 47 – Arquivos de criação das entidades de domínio.....	68
Figura 48 – Arquivos de controladores do sistema	69
Figura 49 – Listagem dos dados cadastrados no MongoDB.....	70
Figura 50 – Ciclo do sistema de emissão e compra.....	71

LISTA DE QUADROS

Quadro 1 - Tabela das categorias do receituário médico.....	27
---	----

LISTA DE ABREVIATURAS

ANVISA	Agencia Nacional de Vigilância Sanitária;
API	Application Programming Interface ou Interface de Programação de aplicativos;
CNPJ	Cadastro Nacional de Pessoa Juridica;
CPF	Cadastro de Pessoa Física;
CRM	Conselho Regional de Medicina;
CSS	Linguagem de estilo em cascata;
DAPP	Aplicativos Descentralizados;
DIVISA	Diretoria de Vigilância Sanitária e Ambiental;
ETH	Criptomoeda da rede Ethereum;
GAS	Recurso utilizado para o processamento dos nós;
HTML	Linguagem de marcação para a internet;
HTTP	Hypertext Transfer Protocol ou Protocolo de transferencia de hipertexto;
LGPD	Lei Geral de Proteção de Dados;
NPM	Node Package Manager ou Gerenciador de Pacotes Node;
ODM	Object Data Manager ou Gerenciados de Objetos de Dados;
REST	Representational State Transfer ou Transferencia de Estado Representacional;
RG	Registro Geral;
SCSS	Linguagem de estilos para sistemas WEB com o framework SASS;
SHA	Secure Hash Algorithm ou Algoritmo de Dispersão Seguro;

SPA	Aplicação de uma única página;
WEB	Sistema que opera através da internet;

SUMÁRIO

INTRODUÇÃO	14
1.1. Objetivo	15
1.2. Justificativa	15
1.3. Metodologia	16
2. Blockchain.....	18
2.1. Smart Contract	19
2.2. Assinatura Digital	19
2.2.1. Criptografia de chave publica.....	20
2.2.2. Hashing de texto	21
2.2.3. Criptografia SHA-2.....	23
2.3. Ethereum.....	23
2.3.1. Utilização prática da plataforma Ethereum	24
2.3.2. Confiança no Sistema	25
3. Emissão de Receitas	27
3.1. Tipos de Receitas.....	27
3.1.1. Receituário médico branco simples	29
3.1.2. Receituário médico branco especial	29
3.1.3. Receituário médico azul.....	30
3.1.4. Receituário médico amarelo	31
3.2. Dados sensíveis	32
3.2.1. Lei Geral de Proteção de Dados.....	32
3.2.2. LGPD na área da saúde	33
4. Desenvolvimento da Aplicação.....	34
4.1. Arquitetando o Sistema	35
4.1.1. Serviços da Entidade Médico.....	38
4.1.2. Serviços da Entidade Paciente	38
4.1.3. Serviços da Entidade Farmácia	39
4.1.4. Serviços da Entidade Organizacional	40
4.2. Desenvolvimento do Smart Contract.....	41
4.2.1. Criando a rede Ethereum Local	41
4.2.2. Desenvolvendo o Smart Contract	45
4.2.3. Compilando o código	49

4.2.4.	Implementação do código na rede blockchain.....	50
4.3.	Desenvolvimento da Interface Gráfica	51
4.3.1.	Layout minimalista	52
4.3.2.	Redução de erros com base no layout	52
4.3.3.	Criando o projeto em Angular	53
4.3.3.1.	Desenvolvimento do Login	55
4.3.3.2.	Desenvolvimento da tela do Médico	59
4.3.3.3.	Desenvolvimento da tela da farmácia.....	60
4.3.3.4.	Desenvolvimento da tela do paciente.....	61
4.3.3.5.	Desenvolvimento da tela do Auditor	63
4.4.	Desenvolvimento do sistema auxiliar	66
4.4.1.	Necessidade do sistema auxiliar no projeto	66
4.4.2.	Integração de sistemas distribuídos.....	67
4.4.3.	Criando o projeto com node.js	67
4.4.4.	Utilizando Banco de dados Não Relacional	69
4.5.	Sistema de pagamentos.....	70
4.5.1.	Necessidade do Sistema de pagamento	71
4.5.2.	Flexibilidade do Sistema de Pagamento.....	72
	CONCLUSÃO	73

INTRODUÇÃO

O atual cenário acerca de documentos médicos vem se provando cada vez mais inseguro com o decorrer do tempo, embora tais documentos tenham um protocolo a ser seguido, isto não significa que algumas etapas não possam ser burladas, tais como a assinatura do médico, o vínculo do paciente com a receita médica, podendo ocasionar em vendas de receitas médicas de medicamentos controlados, os quais não devem por hipótese alguma ser utilizada como automedicação, partindo deste contexto, podemos levantar uma pergunta, o que seria preciso para aumentar a segurança e autenticidade destes documentos?

Ao analisar o tema pode-se perceber que o processo está focado em números de protocolos, assinaturas, nome de medicamentos e descrições, partindo do ponto onde sabe-se que estes dados são centralizados em um órgão de controle.

Pode-se perceber que base de dados centralizados geralmente são mais fáceis de concentrar um ataque, um hacker ou até mesmo uma pessoa visando uma oportunidade de negócio ilícito pode facilmente arquitetar um plano de ataque para que consiga criar protocolos fantasmas, ou até mesmo conseguir protocolos em branco, o qual está em posse dos profissionais da saúde.

Com base no que foi levantado podemos chegar a um ponto chave, será que a segurança está na mão de uma pessoa que pode ser corruptível?

Levando como base o profissional da saúde que será o nosso homem do meio, que faz a emissão destas receitas, pode-se identificar que o elo fraco desta segurança talvez seja ele, pois um ataque direto à base de dados precisaria de recursos de difícil acesso à qualquer pessoa, desta maneira podemos tomar como certo a insegurança deste processo.

As receitas médicas controladas são emitidas para o controle de consumo dos medicamentos pela população, para esta função ele é composto de três tipos de receitas, branca, azul e amarela, sendo que as duas últimas são numeradas com um nº de protocolo, CRM do médico, nome do paciente e da medicação a ser usada e ainda a posologia que consiste como o paciente deve usar o medicamento. (ANVISA, 1998).

1.1. Objetivo

Este projeto tem como objetivo facilitar a emissão de receituários médicos para a população, assim permitindo que comprem os medicamentos online, sem a necessidade de uma prescrição médica protocolada, também evitando possíveis fraudes que possam vir a ocorrer durante o processo de geração e emissão dos receituários, facilitando também auditorias.

Para alcançar este objetivo o sistema desenvolvido será responsável por permitir a prescrição de medicamentos online, garantindo que esta receita seja emitida e visível em uma rede blockchain para que assim seja dificultado qualquer processo de fraude, pois uma vez o receituário sendo persistido no banco de dados descentralizado da rede blockchain, ele será sempre validado diante os vários nós de computadores da rede, assim garantido que qualquer alteração de dados seja apontada em uma dessas validações.

O sistema também permitirá a compra dos medicamentos online, listando por farmácias e gerando o pedido após pagamento, após esta etapa o sistema irá permitir que a farmácia dê baixa na receita para que a mesma seja invalidada no blockchain, assim evitando que ela seja utilizada novamente.

1.2. Justificativa

Com este projeto proposto será possível garantir que as pessoas tenham acesso aos seus medicamentos controlados dentro de sua residência, pois evita um processo burocrático, no qual consiste de vários protocolos a serem seguidos e várias regras a serem respeitadas para que assim o medicamento possa ser prescrito aos pacientes.

Além da facilidade ao acesso dos medicamentos também há uma atenção às pessoas com algum tipo de deficiência, pois elas muitas vezes utilizam de medicamentos controlados, esse sistema garante a facilidade de acesso para estas pessoas, pois também garante um histórico de retiradas e também facilita de pedir o medicamento do conforto de sua residência.

Outra grande justificativa para este sistema é a garantia da utilização de cada receita, pois o mesmo só pode ser retirado uma vez e tem os dados de ambas as partes desta transação, no caso do médico e paciente, assim também permitindo uma melhor auditoria do fluxo de medicamentos, pois os dados não podem ser alterado e estão visíveis na rede blockchain.

1.3. Metodologia

Inicialmente foi feito um estudo bibliográfico para entender o funcionamento de um blockchain e suas possíveis vantagens para o problema inicialmente proposto, onde foi descoberto uma nova vertente dentro do blockchain, voltado para as empresas, no caso esta tecnologia é o smart contract qual visa facilitar a transação entre duas partes e ainda garantir que os processos do contrato sejam cumpridos por ambas as partes por meio de programação.

Na segunda Etapa foi feito um levantamento de requisitos para fechar as regras de negócio do projeto, já visando as tecnologias do mercado que possam satisfazer todas as necessidades do projeto, para isso foi utilizado uma Engenharia de Software para fazer os casos de uso a fim de entender o papel de cada usuário da aplicação, além disso também foram criados diagramas de serviço para a consolidação dos processos que serão utilizados no sistema.

Em seguida foi desenvolvido uma Aplicação web, a fim de manter uma maior segurança e controle nas versões disponibilizadas para os consultórios e farmácias, o Sistema contém níveis de acesso, os quais são médico, paciente, organização e farmácia.

O usuário médico emite a receita digital durante a consulta através do sistema para que o paciente adquira o mesmo por e-mail e em seguida seleciona a farmácia mais próxima para que o remédio seja entregue, para a geração do receituário médico é necessário que alguns dados sejam criados, estes contém uma camada de segurança extra para evitar que qualquer um consiga utilizar o medicamento prescrito, para isso é gerado um hash único e uma senha, estes dois dados são enviados para o paciente via e-mail, então o hash da receita é

gravado com esta senha, o que impossibilita a retirada do sistema sem ter acesso a estes dados.

Após esta etapa de geração do receituário médico foi feito a integração de um método de pagamento pelo PayPal, o qual é uma biblioteca integrada ao projeto que faz todo o processo de pagamento e recebimento de valores online, após o paciente escolher a farmácia e inserir o hash da Receita junto à senha ele terá a opção de pagar pelo PayPal, após o pagamento um pedido será gerado e enviado para a farmácia receber e dar baixa nesta receita do Blockchain.

Por fim foi desenvolvido a funcionalidade de recebimento dos pedidos pelas farmácias, no qual o usuário precisa fazer o login no sistema com o CNPJ da farmácia e o login único vinculado a mesma, após a autenticação do usuário ser feita com sucesso, ele conseguirá visualizar todos os pedidos que foram feitos para a sua farmácia, tendo o pagamento já sido feito logo resta à farmácia separar o pedido para entrega e dar baixa no sistema.

Com todo o processo de emissão e retirada do receituário médico feito, agora foi possível desenvolver a visualização governamental para auditar as receitas médicas que foram emitidas por médico, este recurso do sistema lista todos os médicos e mostra o estado de cada receita, se foi apenas gerado, feito o pagamento ou dado baixa pela farmácia, além de mostrar qual farmácia foi feito este processo.

Para a validação destes dados, indicando uma possível fraude será preciso a utilização de uma tecnologia que está recebendo cada vez mais relevância no cenário tecnológico, o Blockchain baseado na metodologia Smart Contract, o qual visa garantir uma transação por cruzamento de dados criptografados verificados por um grid de computadores, os quais garantem que estes dados não sejam manipulados por nenhum nó do grid, pois uma transação fora do padrão pode acarretar o isolamento deste bloco e possível auditoria para entender o que houve durante o processo, esta etapa do projeto será escrito em uma linguagem própria para Smart Contract tal como o Solidity.

2. Blockchain

A tecnologia blockchain surgiu junto à criptomoeda bitcoin, embora estas tecnologias estejam fortemente atreladas, hoje em dia podemos utilizá-la para aumentar a segurança de processos, evitando possíveis fraudes.

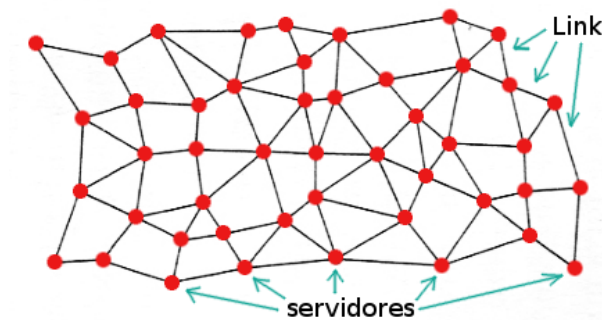
Esta tecnologia surgiu para ser um livro contábil público, o qual servia para armazenar o registro de uma transação de moedas virtuais, de maneira confiável e imutável. (FOXBIT, 2019).

A principal característica do blockchain é a sua capacidade de autenticar transações de maneira transparente, pois o mesmo é composto de uma rede distribuída, onde todos os nós estão conectados, onde estes são alimentados por uma base de dados as quais mantem um histórico de transações efetuadas e cada nó da rede precisa validar estas transações para que a mesma seja efetuada, se houver algum tipo de alteração de um nó para outro no mesmo bloco analisado estes dados serão invalidados e a transação não será permitida.

Para aumentar a segurança e confiabilidade dos blocos os dados são armazenados e analisados em forma de hash, o qual gera uma assinatura única para a transação e esta assinatura é conferida pelos nós da rede do blockchain sempre verificando as assinaturas anteriores deste bloco.

Dentro de uma rede distribuída temos vários nós de computadores sendo utilizados para dar andamento ao processo de validação do blockchain, como está sendo representado na figura 1, onde o poder de análise das transações é feito pela maioria, se todos os nós identificam que as transações dentro de um nó são válidas, então é feita a persistência dessa transação dentro do blockchain.

Figura 1 – Grafo de um Sistema Distribuído



Fonte: Nós Digitais. Disponível em:<[http://wiki.nosdigitais.teia.org.br/Modelos de Rede](http://wiki.nosdigitais.teia.org.br/Modelos_de_Ne)>.

Acesso em 4 out. 2019.

2.1. Smart Contract

O Smart Contract tem o mesmo conceito de um contrato, onde uma transação necessita do acordo de ambas as partes para serem firmados, atualmente com o avanço tecnológico do blockchain, podemos ter a mesma ideia feita de maneira digital.

Os Smart Contracts são contratos digitais autoexecutáveis, isso quer dizer que eles são códigos de programação que definem as regras estritas e as consequências, da mesma forma que um documento tradicional, estabelecendo obrigações, benefícios e penalidades devidas às partes em diferentes circunstâncias. (BITCOINTRADE, 2020).

Esta teoria pode ser explicada com uma simples máquina de venda de bebidas, a qual necessita que alguém coloque o valor da bebida para que assim consiga adquiri-la, a máquina passa a ser o contrato, o qual necessita que tenha a bebida e que o valor seja o correto para que ambas as partes concluam a transação, o cliente precisa da bebida em troca do dinheiro e a empresa da máquina precisa do dinheiro em troca da bebida.

2.2. Assinatura Digital

A assinatura digital é um meio de identificar algo através das técnicas de criptografia assimétrica, a qual gera um valor criptografado por meio de funções matemáticas avançadas para que seja gerada uma assinatura única, que deve ser o mesmo quando verificado.

Para garantir a autenticidade desta assinatura, ela é feita através do método de chaves públicas e privadas, onde a chave privada fica em posse do detentor da assinatura e a chave pública serve para verificar a veracidade do remetente.

2.2.1. Criptografia de chave publica

É qualquer criptografia que utiliza de pares de chaves, as chaves públicas nas quais podem ser vistas por qualquer um interessado e a chave privada que é única e exclusivamente do proprietário dela, como pode ser descrito pela citação direta a seguir:

“Na criptografia assimétrica, cada parte envolvida na troca de informações deve possuir seu próprio par de chaves $\{K_{pub}; K_{pri}\}$, A chave pública, como seu próprio nome indica, é de conhecimento geral, sendo livremente distribuída e divulgada. A chave privada, por sua vez, é conhecida apenas pelo seu proprietário.” (CARÍSSIMI, 2009, p. 349)

Esta criptografia exerce duas funções, a de autenticação, onde a chave pública verifica o envio de uma mensagem pelo portador da chave privada e encriptação, em que apenas o portador da chave privada pode decriptar uma mensagem criptografada, logo quando o remetente envia um texto comum para o destinatário, antes esta mensagem é cifrada para evitar a leitura de pessoas mal intencionadas, quando o destinatário recebe esta mensagem a sua chave privada é capaz de decriptar a mesma, assim a tornando legível novamente, como mostrado na figura 2.

Figura 2 – Representação de chave pública e privada



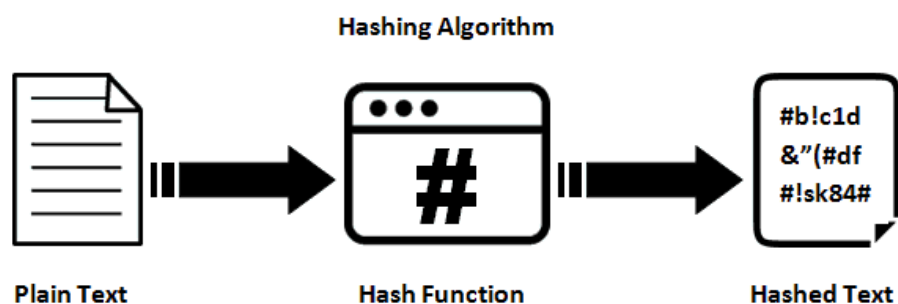
Fonte: Macoratti.net. Disponível em: <http://www.macoratti.net/Cursos/Cripto/net_cripto5.htm>. Acesso em 9 out. 2019.

Com este modelo de criptografia assimétrica podemos ter a segurança de que se alguém mal intencionado adquirir a mensagem criptografada pela rede, ele não será capaz de ler o conteúdo da mensagem pois seria necessário a chave privada.

2.2.2. Hashing de texto

O Hashing é a técnica de criptografia em que transforma uma mensagem em uma sequência de caracteres totalmente diferentes do original, tanto em conteúdo quanto em comprimento, a fim de evitar quaisquer alterações no conteúdo também, de maneira que com uma única alteração de bit na mensagem resulte em uma sequência de caracteres totalmente diferente, pode-se ver esta representação na figura 3.

Figura 3 – Representação do algoritmo de Hashing



Fonte: Network Encyclopedia. Disponível em: <<https://networkencyclopedia.com/hashing-algorithm/>>. Acesso em 07 out. 2019.

O Hashing embora seja utilizado como uma forma de criptografia, ele não é de fato uma, como pode-se ver na citação direta a seguir:

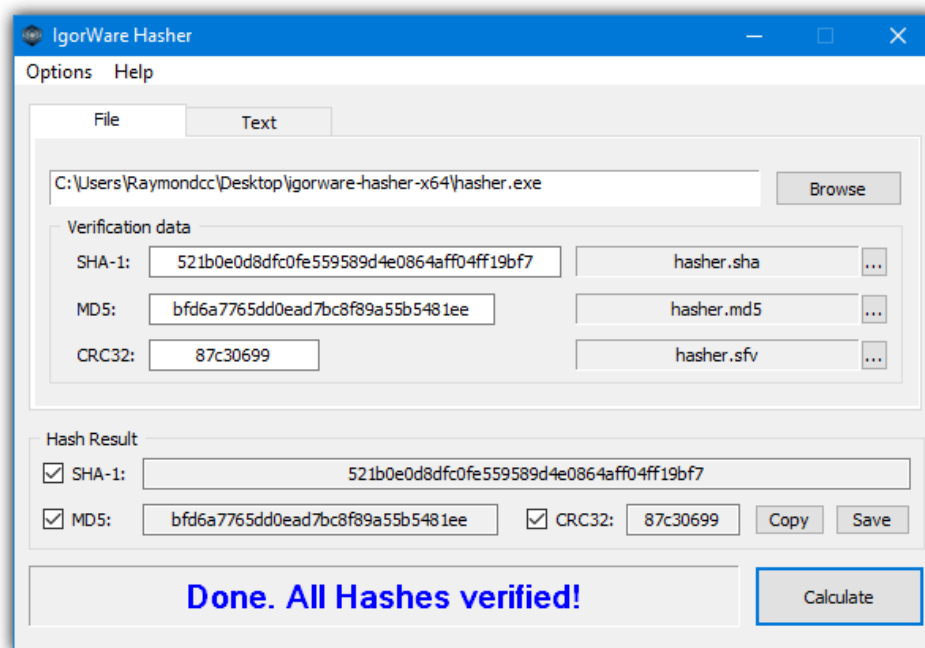
“Hashing não é criptografia, mas uma forma de criptografia. Lembre-se do Capítulo 22 que hashing é um processo matemático unidirecional usado para criar um texto cifrado. Mas, ao contrário da criptografia, depois que um hash é criado, você não pode transformá-lo de volta ao texto simples original.” (NEGUS, 2018, p. 684).

Dentro do método de Hashing podemos ter vários outros tipos de algoritmos os quais são utilizados para diferentes finalidades, os mais populares são MD2, MD4, MD5 e SHA1. O modelo de criptografia MD5 resulta em 128bits enquanto o SHA1 tem 160bits.

Quanto maior a quantidades de Bits que uma criptografia gera, maior será o tamanho do hash resultante, também evitando uma colisão, onde um caractere específico está na mesma posição do texto original.

O método de hashing abrange uma grande área de atuação, um bom exemplo prático da utilização de uma criptografia SHA é a garantia de integridade de um arquivo, em que você pode gerar um hash do arquivo original e compará-lo após a transferência, se ocorrer alguma alteração neste arquivo a chave de verificação (hash do arquivo) será diferente do original, a figura 4 demonstra o funcionamento de um software que realiza o hashing de um texto.

Figura 4 – Exemplo de um Hashing de texto



Fonte: Raymond.cc. Disponível em: <<https://www.raymond.cc/blog/7-tools-verify-file-integrity-using-md5-sha1-hashes/>>.

Acesso em 10 out. 2019.

2.2.3. Criptografia SHA-2

A criptografia SHA-2 é um conjunto de funções hash criptográficas desenvolvida pela Agência de Segurança Nacional dos Estados Unidos da América, dentro deste conjunto temos as funções hash SHA-224, SHA-256, SHA-384, SHA-512, SHA-512/224, SHA-512/256.

Quanto maior o número do SHA (secure hash algorithm) maior será os bits resultantes, consequentemente exigindo um processamento maior para que a mensagem seja criptografada. Para ter um hash SHA é preciso fazer um processo iterativo onde será executado uma função matemática específica em cada caractere de uma mensagem de texto.

2.3. Ethereum

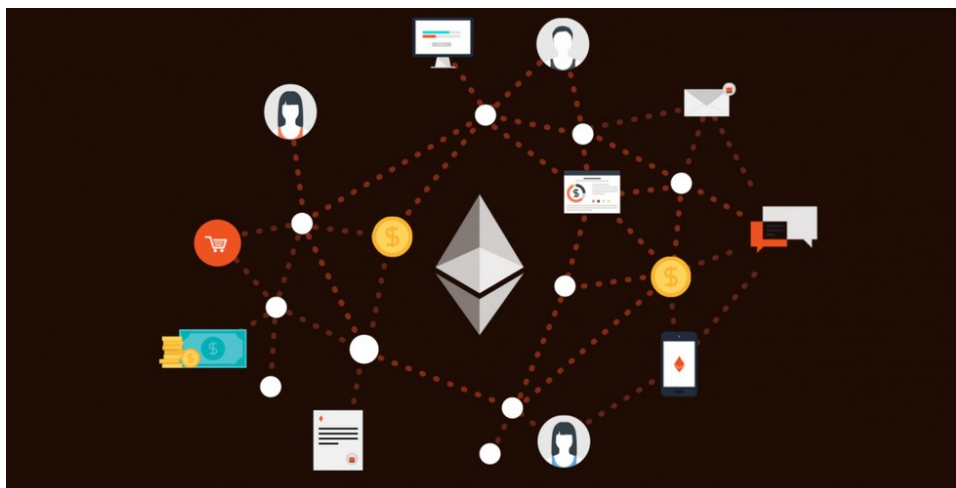
O Ethereum surgiu de uma divisão após um hacker roubar cerca de US\$ 50 milhões de Ether (criptomoeda da rede Ethereum) por meio de uma falha, após a incerteza diante do futuro da moeda, seu criador, Vitalik Buterin apoiou a

criação de uma nova rede, que permitiria inclusive, que estes ativos roubados fossem devolvidos. (INFOMONEY, 2019).

Diferente do Bitcoin o Ether não foi feito para se tornar uma moeda digital, mas sim um ativo para recompensar os desenvolvedores que usam a plataforma Ethereum para seus projetos, mesmo assim o Ethereum é uma das três moedas digitais mais negociadas do mundo. (INFOMONEY, 2019).

Ethereum é uma plataforma construída utilizando da tecnologia blockchain para a execução de contratos inteligentes e DAPPs(Aplicações Descentralizadas) em que se utiliza do blockchain para desenvolver programas sem qualquer tipo de fraude, censura ou interferência de terceiros, a figura 5 demonstra como estas aplicações descentralizadas podem ser independentes de um sistema centralizado.

Figura 5 – Demonstrativo da rede Ethereum



Fonte: Tech Tudo. Disponível em:<<https://www.techtudo.com.br/noticias/2018/08/o-que-sao-dapps-apps-descentralizados-podem-revolucionar-a-internet.ghtml>>.

Acesso em 15 out. 2019.

2.3.1. Utilização prática da plataforma Ethereum

A plataforma Ethereum vem crescendo muito com o passar dos anos pelo fato de possibilitar o uso de uma rede descentralizada para a criação de sistemas transparentes, que tem o foco em evitar fraudes, censuras e interferências de terceiros.

Agora, milhares de desenvolvedores em todo o mundo estão construindo aplicativos na Ethereum e inventando novos tipos de aplicativos. (ETHEREUM, 2019).

Como já citado o blockchain foi feito a fim de criar uma moeda digital desancorada de qualquer outra moeda, de maneira descentralizada assim impossibilitando a fraude, como o caso de alguma pessoa má intencionada adicionar mais moedas a sua carteira, o Ethereum utiliza dessa mesma ideia para o desenvolvimento de aplicações, o aplicativo pode utilizar de todo o recurso da rede descentralizada do Ethereum para impedir a fraude ou manipulação dos dados inseridos na rede.

Esta plataforma vem atraindo cada vez mais a atenção de órgãos governamentais, onde há uma grande taxa de fraude envolvendo licitações, contratos, etc. Com uma rede descentralizada, onde cada nó da rede está situado em um lugar diferente, validando cada transação, é possível criar um sistema de licitações para que nada seja feito de maneira a beneficiar uma pessoa.

Podemos ter também exemplos da área da Saúde, onde empresas já utilizam dos smart contracts para registrar dados de pacientes, os quais só e somente podem ser exibidos com o consentimento do paciente, provando que esta rede consegue também manter sigilo de informação, dependendo somente da regra de negócio do programa e das condições do contrato a ser seguido.

Outro ótimo exemplo a ser utilizado com um sistema baseado em smart contracts é uma aplicação de votação, pois pode ser feito de forma transparente, segura e a prova de fraudes.

2.3.2. Confiança no Sistema

Temos diversas razões para utilizar um smart contract ao invés de um contrato tradicional. Temos uma maior eficiência ao analisar os resultados de um smart contract, pois ele consegue ser executado de uma maneira bem mais rápida sem a necessidade de cartórios, advogados, transferências bancárias, etc. A

maior vantagem mesmo vem da segurança ganha em garantir que nenhum das partes faltem com as suas obrigações, deixando de cumprir o acordo ou caloteando outra das partes.

A principal vantagem de um blockchain é permitir a troca de informação entre pessoas que não se conhecem ou confiam umas nas outras, sem a necessidade de que haja uma entidade central que assume o papel de árbitro.

A maior vulnerabilidade do Ethereum está na falha humana, porque cada pessoa é responsável pelos contratos que participa ou submete, dessa maneira uma pessoa pode criar um contrato de investimentos que levanta interesse de pessoas que desejam participar dele, injetando dinheiro, a qualquer momento o contrato pode ser dado como concluído e o criado pode então sumir com o dinheiro de todos, essa vulnerabilidade é uma falha humana pois as partes que participam do contrato devem analisa-lo para descobrir possíveis brechas.

3. Emissão de Receitas

O principal ponto para o controle da emissão de receitas médicas é única e exclusivamente para o uso racional de medicamentos, em que é regulamentado pela Agência Nacional de Vigilância Sanitária (ANVISA) de maneira a garantir a fiscalização dos pontos de venda de medicamentos.

Dentro das receitas temos vários tipos que indicam qual tipo de medicamento esta receita se destinará, para um maior controle do fluxo dos medicamentos, principalmente os que podem induzir a algum tipo de sintomas com uma maior frequência, todos as receitas controladas são protocoladas e sujeitas a análise da ANVISA.

O profissional da saúde legalmente habilitado que efetuar a emissão de uma receita médica para um paciente deve fazer uma análise do quadro clínico do mesmo para justificar a receita que deve ser

3.1. Tipos de Receitas

Em torno as receitas médicas nós temos várias cores diferentes que indicam um tipo de medicamento que deve seguir, dessa maneira, a ANVISA estabeleceu esta diferenciação dos receituários médicos para controlar a prescrição e compra adequada do medicamento, a figura 6 demonstra cada classificação de receituário médico, com seus respectivos critérios. Portanto é de extrema importância que os profissionais da saúde legalmente habilitados a receitar medicamentos (médicos, veterinários, dentistas e farmacêuticos) saibam identificar o tipo de papel correto em que a receita médica deve ser indicada.

Quadro 1 - Tabela das categorias do receituário médico

RECEITUÁRIOS DE MEDICAMENTOS CONTENDO SUBSTÂNCIAS SUJEITAS A CONTROLE ESPECIAL							
Lista presente no anexo I da portaria 344	Tipo de substância	Tipo de receita	Cor	Anexo da portaria 344 em que o modelo está contido	Características do modelo de receita	Validade e alcance no território nacional do receita	Tempo de tratamento possível em cada notificação/receita
A1	Entorpecentes	Notificação de Receita A	Amarela	IX	De cor amarela, será impressa, as expensas da Autoridade Sanitária Estadual ou do Distrito Federal, deve conter 20 (vinte) folhas em cada talonário e será fornecida gratuitamente aos profissionais e instituições devidamente cadastrados. (art. 40)	Válida por 30 dias a contar da data de sua emissão em todo o território nacional. Se destinada à aquisição em outra Unidade da Federação também é necessário que seja acompanhada da receita médica com justificativa do uso (art. 41). As Notificações de receita "a" procedentes de outras unidades federativas devem ser obrigatoriamente apresentadas pelas farmácias e drogarias à autoridade sanitária local em até 72 horas, para averiguação e visto (parágrafo único).	A Notificação de Receita "A" poderá conter no máximo de 5 (cinco) ampolas e para as demais formas farmacêuticas de apresentação, poderá conter a quantidade correspondente no máximo a 30 (trinta) dias de tratamento. (art. 43)
A2	Entorpecentes						
A3	Psicotrópicas						
B1	Psicotrópicas	Notificação de Receita B	Azul	X	De cor azul, impressa às expensas do profissional ou da instituição, conforme modelos anexos da portaria SVS nº 344 / 1998 ou RDC nº 58/2007.	Válida por 30 dias a contar da data de sua emissão e somente dentro da Unidade Federativa que concedeu a numeração (art. 45).	A Notificação de Receita "B" poderá conter no máximo 5 (cinco) ampolas e, para as demais formas farmacêuticas, a quantidade para o tratamento correspondente no máximo a 60 (sessenta) dias. (art. 46)
B2	Psicotrópicas Anorexígenas	Notificação de Receita B2	Azul	I (RDC nº 58 / 2007)	De cor azul, impressa às expensas do profissional ou da instituição, conforme modelos anexos da portaria SVS nº 344 / 1998 ou RDC nº 58/2007.	Validade de 30 (trinta) dias contados a partir da sua emissão e somente dentro da Unidade Federativa que concedeu a numeração (art. 1º, § 2º da RDC nº 58/2007)	Cada Notificação de Receita "B2" deve ser utilizada para tratamento igual ou inferior a 30 (trinta) dias. Se o médico prescrever quantidade inferior, esta deverá ser mantida (de acordo com a RDC nº 58 / 2007). Deverá ser respeitada a dose diária recomendada (DDR) estabelecida na RDC nº 52 / 2011.
C1	Outras substâncias sujeitas a controle especial	Receita de controle especial em duas vias	---	XVII	Deverá estar escrita de forma legível, a quantidade de algarismos arábicos e por extenso, sem emenda ou rasura. Na receita C5 são obrigatórias a inclusão do CPF do profissional e do CID do paciente.	Válida por 30 dias contados a partir da data de sua emissão e em todo o território nacional (art. 52, § 1º).	C1 e C5: Limitada a 5 (cinco) ampolas e para as demais formas farmacêuticas, a quantidade para o tratamento correspondente a no máximo 60 (sessenta) dias. No caso de substâncias ou medicamentos antiparkinsonianos e anticonvulsivantes, a quantidade ficará limitada até 6 (seis) meses de tratamento.
C5	Anabolizantes						
C4	Antiretrovirais	Próprio do programa de DST/AIDS do Ministério da Saúde	---	---	Não há modelo ou cor definida.	Não há prazo de validade determinado e nem proibição de uso em várias unidades federativas.	Não há quantidade determinada pela legislação.
C2	Retinóides de Uso tópico	Receita comum (sem retenção)	---	---	---	---	---
C2	Retinóides de Uso sistêmico	Notificação de Receita especial	Branca	XII	Será impressa às expensas do médico prescritor ou pela instituição a qual esteja filiado	Válida por 30 dias contados a partir de sua emissão e somente dentro da Unidade Federativa que concedeu a numeração (art. 50).	Poderá conter no máximo 5 (cinco) ampolas, e, para as demais formas farmacêuticas, a quantidade para o tratamento correspondente no máximo a 30 (trinta) dias.
C3	Talidomida	Notificação de receita de talidomida	Branca	VI (RDC nº 11 / 2011)	Será impressa e distribuída gratuitamente pela autoridade sanitária competente somente aos profissionais médicos devidamente cadastrados.	Válida por 20 dias contados a partir da data de sua emissão e somente dentro da unidade federativa onde foi emitida.	A quantidade de Talidomida por prescrição, em cada Notificação de Receita, não poderá ser superior à necessária para o tratamento de 30 (trinta) dias. (art. 21, §3º da RDC nº 11 / 2011)

Fonte: Secretaria Municipal de Saúde. Disponível em: <<http://www.saude.campinas.sp.gov.br>>. Acesso em 20 nov. 2019.

Cada Tipo de receita tem um tipo de notificação, o qual indica o tipo de medicamento da receita os quais são Notificação de Receita "A", "B" e "C", onde cada notificação tem um sub tipo indicado por um número, indo de 1 até 5, cada um desses tipos indica um tipo de medicamento diferente, estas receitas são

apenas válidas pelo território nacional brasileiro, impossível de ser utilizado fora de nosso território nacional.

As receitas médicas contêm uma quantidade máxima por receita e período de tratamento específica de cada faixa de notificação também limitando o número de medicamentos a serem expedidos pela receita, além disso as receitas têm uma data de validade para que não seja possível usar a mesma mais de uma vez

3.1.1. Receituário médico branco simples

Este tipo de notificação é geralmente prescrito para medicamentos que não necessitam de receituário médico para a compra, mas que não devem ser adquiridos para automedicação, estes devem ser prescritos por um profissional da saúde.

Esta receita é emitida em um papel branco em somente uma via apenas para a indicação de uso do medicamento e a garantia de que não há automedicação.

Embora esta receita contenha medicamentos normais prescritos pelos profissionais da saúde, também pode ser prescrito para medicamentos mediante o acompanhamento do termo de responsabilidade de uso, contemplando as notificações C2 e C3 as quais se referem ao uso de retinóides de uso sistêmico e imunossupressores (Talidomida).

3.1.2. Receituário médico branco especial

Esta notificação é exclusivamente emitida para medicamentos de uso controlado, limitando a quantidade retirada e a receita especificando a posologia do mesmo, para que assim o paciente siga a orientação médica para o uso, a figura 7 demonstra o aspecto visual deste receituário.

Este tipo de notificação está dentro da classe C, contendo alguns adendos às notificações B e A, onde contempla os medicamentos de controle especial

(C1), anabolizantes (C5), adendos das Listas (A1; A2; B1), Antiparkinsonianos Anticonvulsivantes (C1; B1), como pode ser visto na figura 6.

Os itens contemplados por esta receita contêm um controle de 5 ampolas para os medicamentos de classe (C1; C5; A1; A2; B1) com tratamento para 60 dias e as classes (C1; B1) para 180 dias e 3 medicamentos ou substâncias por receita.

Figura 7 – Receituário médico branco especial

RECEITUÁRIO CONTROLE ESPECIAL																	
<table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <th colspan="2" style="text-align: center; padding: 5px;">IDENTIFICAÇÃO DO EMITENTE</th> </tr> <tr> <td colspan="2" style="padding: 5px;">Nome Completo: _____</td> </tr> <tr> <td style="padding: 5px;">CRM _____</td> <td style="padding: 5px;">UF _____ Nº _____</td> </tr> <tr> <td colspan="2" style="padding: 5px;">Endereço Completo e Telefone: _____</td> </tr> <tr> <td style="padding: 5px;">Cidade: _____</td> <td style="padding: 5px;">UF: _____</td> </tr> </table>	IDENTIFICAÇÃO DO EMITENTE		Nome Completo: _____		CRM _____	UF _____ Nº _____	Endereço Completo e Telefone: _____		Cidade: _____	UF: _____	<div style="text-align: center; margin-bottom: 10px;">1ª VIA FARMÁCIA</div> <div style="text-align: center;">2ª VIA PACIENTE</div>						
IDENTIFICAÇÃO DO EMITENTE																	
Nome Completo: _____																	
CRM _____	UF _____ Nº _____																
Endereço Completo e Telefone: _____																	
Cidade: _____	UF: _____																
<p>Paciente: _____</p> <p>Endereço: _____</p> <p>Prescrição: _____</p> <p>_____</p> <p>_____</p> <p>_____</p> <p>_____</p> <p>_____</p> <p>_____</p>																	
<table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <th colspan="2" style="text-align: center; padding: 5px;">IDENTIFICAÇÃO DO COMPRADOR</th> </tr> <tr> <td colspan="2" style="padding: 5px;">Nome: _____</td> </tr> <tr> <td style="padding: 5px;">Ident.: _____</td> <td style="padding: 5px;">Org. Emissor: _____</td> </tr> <tr> <td colspan="2" style="padding: 5px;">End.: _____</td> </tr> <tr> <td style="padding: 5px;">Cidade: _____</td> <td style="padding: 5px;">UF: _____</td> </tr> <tr> <td colspan="2" style="padding: 5px;">Telefone: _____</td> </tr> </table>	IDENTIFICAÇÃO DO COMPRADOR		Nome: _____		Ident.: _____	Org. Emissor: _____	End.: _____		Cidade: _____	UF: _____	Telefone: _____		<table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <th colspan="2" style="text-align: center; padding: 5px;">IDENTIFICAÇÃO DO FORNECEDOR</th> </tr> <tr> <td colspan="2" style="height: 150px; vertical-align: bottom; padding: 5px;"> <div style="display: flex; justify-content: space-between; margin-top: 10px;"> ASSINATURA DO FARMACÊUTICO _____ DATA: ____/____/____ </div> </td> </tr> </table>	IDENTIFICAÇÃO DO FORNECEDOR		<div style="display: flex; justify-content: space-between; margin-top: 10px;"> ASSINATURA DO FARMACÊUTICO _____ DATA: ____/____/____ </div>	
IDENTIFICAÇÃO DO COMPRADOR																	
Nome: _____																	
Ident.: _____	Org. Emissor: _____																
End.: _____																	
Cidade: _____	UF: _____																
Telefone: _____																	
IDENTIFICAÇÃO DO FORNECEDOR																	
<div style="display: flex; justify-content: space-between; margin-top: 10px;"> ASSINATURA DO FARMACÊUTICO _____ DATA: ____/____/____ </div>																	

Fonte: Secretaria Municipal de Saúde. Disponível em: <<http://www.saude.campinas.sp.gov.br>>. Acesso em 20 nov. 2019.

3.1.3. Receituário médico azul

Este tipo de notificação tem um controle maior do que as notificações de cor branca especial ou simples, ele permite apenas 1 substância por receita com

as validades fixadas em 30 dias, esta notificação tem como objetivo controlar o uso de psicotrópicos.

A numeração da receita precisa ser retirada com a Diretoria de Vigilância Sanitária e Ambiental (DIVISA), e então enviado a gráfica para a impressão do talão de receitas que será utilizado pelo profissional da saúde, cujo seu visual modelo pode ser visto na figura 8.

Figura 8 – Receituário médico azul

NOTIFICAÇÃO DE RECEITA B

UF: SP NÚMERO: 01 000.001

numeração: de 01.000.001 a 01.000.500

DATA: ____/____/____

PACIENTE: _____

MEDICAMENTO: _____

NOTIFICAÇÃO DE RECEITA

UF: SP NÚMERO: B

SÉRIE: ZZ

de ____ de ____

Paciente: _____

Assinatura do Emissor: _____

Endereço: _____

IDENTIFICAÇÃO DO COMPRADOR

Nome: _____

Endereço: _____

Telefone: _____

Identidade Nº: _____

Órgão Emissor: _____

IDENTIFICAÇÃO DO EMITENTE

Medicamento ou Substância: _____

Quantidade e Forma Farmacêutica: _____

Dose por Unidade Posológica: _____

Posologia: _____

CARIMBO DO FORNECEDOR

Nome do Vendedor: _____

Data: ____/____/____

Numeração desta Impressão: de 01.000.001 a 01.000.500

DADOS DA GRÁFICA: ENDEREÇO, TEL, CNPJ, INSC. ESTADUAL

Fonte: Secretaria Municipal de Saúde. Disponível em: <<http://www.saude.campinas.sp.gov.br>>. Acesso em 20 nov. 2019.

3.1.4. Receituário médico amarelo

Esta notificação se refere às receitas de entorpecentes e tem um controle bem rigoroso, pois este gênero de medicamentos pode causar dependência física e psicológica, seu modelo pode ser visto na figura 9.

Figura 9 – Receituário médico amarelo

NOTIFICAÇÃO DE RECEITA A

UF: SP NÚMERO: 01 000.001

numeração: de 01.000.001 a 01.000.500

DATA: ____/____/____

PACIENTE: _____

MEDICAMENTO: _____

NOTIFICAÇÃO DE RECEITA

UF: SP NÚMERO: A

Data: ____/____/20____

Assinatura do Emissor: _____

Paciente: _____

Endereço: _____

IDENTIFICAÇÃO DO COMPRADOR

Nome: _____

Endereço: _____

Identidade Nº: _____

Órgão Emissor: _____

TEL: _____

IDENTIFICAÇÃO DO EMITENTE

ESPECIALIDADE FARMACÊUTICA

Nome: _____

Quantidade e Apresentação: _____

Forma Farm / Concent / Unid. Posológica: _____

IDENTIFICAÇÃO DO FORNECEDOR

Nome do Vendedor: _____

Data: ____/____/20____

Numeração desta Impressão: de 01.000.001 a 01.000.500

DADOS DA GRÁFICA: ENDEREÇO, TEL, CNPJ, INSC. ESTADUAL

Print Healthy 3846-5756

Fonte: Secretaria Municipal de Saúde. Disponível em: <<http://www.saude.campinas.sp.gov.br>>. Acesso em 20 nov. 2019.

3.2. Dados sensíveis

Dentro da área da saúde, temos vários pontos a se considerar, um dos principais é sigilo médico paciente, o qual impede que a informação trocada entre o médico e seu paciente seja divulgada por algum meio, seja de maneira verbal ou digital.

Em 2018 foi criado uma lei para a proteção dos dados pessoais em diversas áreas empresárias, essa lei é a LGPD (Lei Geral de Proteção de Dados) a qual visa manter o sigilo dos dados pessoais dos cidadãos, de maneira com que apenas ele decida compartilhar este dado ou não. Na área da saúde temos dados protegidos pelo sigilo médico paciente, em que os dados compartilhados em uma consulta ficam a cargo do paciente compartilhá-lo ou não, com o avanço tecnológico dos sistemas hospitalares, o LGPD acaba sendo incorporado em diversas áreas, onde o vazamento destes dados pode gerar multas aos hospitais.

Na área da saúde temos diversos documentos contendo dados de pacientes, seja ele uma receita médica, atestado médico, entre outros.

3.2.1. Lei Geral de Proteção de Dados

A Lei LGPD (Lei Geral de Proteção de Dados) foi sancionada em agosto de 2018 e entrou em vigor em 2020. Esta lei tem por objetivo controlar as atividades em relação aos dados pessoais das pessoas, de maneira que estes dados não sejam compartilhados sem seus consentimentos.

Estamos vivendo um momento onde a moeda mais valiosa é a informação, logo cada indivíduo é responsável pelo o que faz com seus dados pessoais, esta lei garante o cumprimento de que as empresas não utilizem ou divulguem os dados das pessoas sem a autorização delas, esses dados podem ser Nome, CPF, endereço e muitos outros dados referentes ao seu histórico de saúde e até mesmo plano de saúde.

3.2.2. LGPD na área da saúde

A principal ideia do LGPD (Lei Geral de Proteção de Dados) na área da saúde é oferecer segurança e privacidade para os seus pacientes dentro dos consultórios e clínicas médicas.

A proteção dos dados pessoas vem da necessidade de garantir que um paciente não seja identificado fora da consulta médica, estes dados são Nome, documento de RG e CPF, número de telefone, e-mail ou endereço por exemplo.

Os dados sensíveis são aqueles que possam vir a originar discriminação e preconceito, estes dados dizem respeito aos valores e convicções de cada um, tal como sua etnia, orientação sexual, convicção religiosa, opinião política e informações de saúde.

Na área da saúde temos vários documentos que utilizam de informações pessoais e sensíveis do paciente, pois para ter a avaliação do quadro clínico é preciso identificar a pessoa, analisar seu histórico médico e identificar quaisquer problemas que esta pessoa possa estar ou vir a desenvolver, encima destes dados é feito um tratamento, o qual passa por um processo de armazenamento, compartilhamento, classificação, acesso, reprodução e avaliação, tendo sempre que ser feito com o consentimento do paciente, isso deve ser informado de maneira clara para que não ocorra equívocos sobre a finalidade do tratamento destes dados.

4. Desenvolvimento da Aplicação

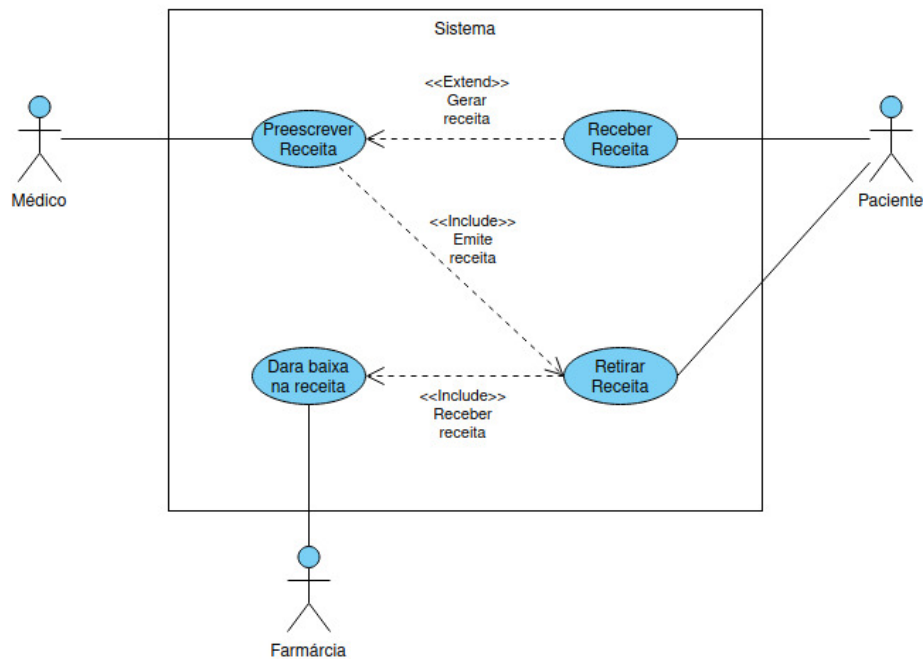
Para um maior entendimento das funcionalidades do sistema foi utilizado a metodologia caso de uso, a qual implica em definir quais usuários (atores) terão no sistema e também as fases de execução dos processos do sistema através de um grafo, o primeiro passo para a criação do caso de uso, como pode-se ver na figura 10 é necessário definir os atores.

O sistema irá funcionar no consultório médico e na farmácia logo, já podemos definir três atores, o médico que será responsável por fazer a consulta e identificar a necessidade de o paciente adquirir algum tipo de medicamento, logo ele é responsável pela prescrição da receita, em seguida tempos o paciente que participará da consulta podendo ter algum tipo de medicamento receitado pelo médico, além desses dois atores temos a farmácia, a qual receberá o pedido do paciente e irá emitir a receita (dar baixa) se a mesma for válida.

Depois de definir os atores podemos identificar o que cada um fará dentro do sistema, pensando no ator médico, podemos identificar que o papel dele no sistema será apenas emitir a receita médica que será gravada dentro de um bloco do blockchain, ele é uma das duas partes do contrato, a outra parte é o paciente que precisa da prescrição da receita, assim podemos já identificar um contrato, onde o médico emite a receita e o paciente recebe a mesma, com este contrato firmado dentro do blockchain, podemos agora pensar na retirada da receita médica, através da interação do paciente com a farmácia, utilizando uma gatilho no contrato feito entre paciente e cliente, onde o cliente tem o hash da receita e a senha única gerada pela mesma, para garantir que somente ele seja capaz de controlar a retirada da receita, o ator da farmácia apenas recebe os dados da receita do paciente e da baixa na mesma.

Neste processo temos apenas três atores, onde cada processo depende da interação dos atores, a prescrição da receita precisa de um cliente para receber a mesma, a retirada necessita que tenha uma receita gerada pela consulta médica e a retirada precisa que tenha uma receita feita para o ciclo da receita seja finalizada.

Figura 10 – Diagrama de caso de uso



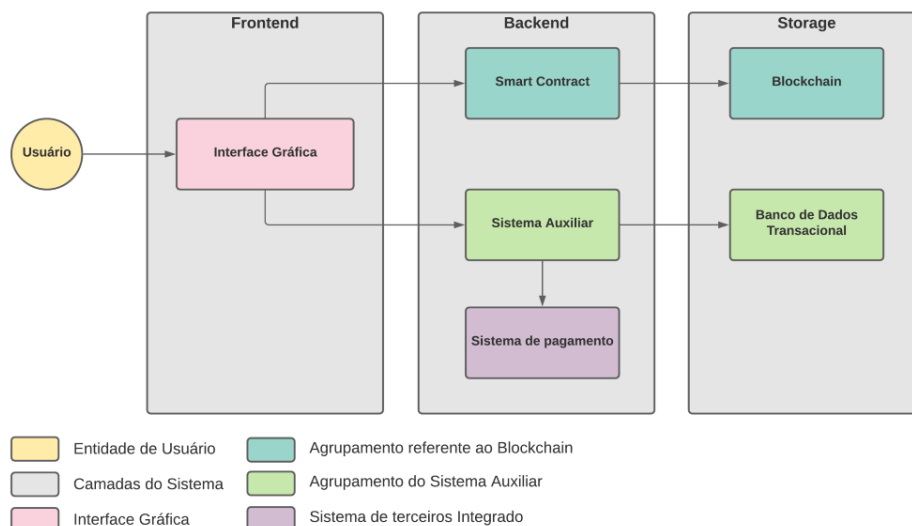
Fonte: Elaborado pelo autor.

Após a definição do diagrama de Caso de uso podemos ter uma base para iniciar o levantamento dos requisitos do sistema e as tecnologias a serem utilizadas.

4.1. Arquitetando o Sistema

Como primeiro passo antes de começar o desenvolvimento do sistema, foi preciso levantar os requisitos e as tecnologias que seriam necessárias para que o projeto tivesse sucesso, para isso foi preciso identificar as camadas que seriam utilizadas no projeto, como pode ser visto no diagrama representado na figura 11, para uma melhor ilustração.

Figura 11 – Diagrama da arquitetura do projeto



Fonte: Elaborado pelo autor.

De acordo com a imagem acima podemos identificar que o projeto foi separado em três camadas, o Frontend que contém a interface gráfica de acesso direto do usuário, em seguida há a camada de Backend, onde temos as regras de negócio, o qual irá executar o smart contract, associar os usuários aos contratos, gerar pedidos de medicamento, dar baixa do contrato no blockchain, etc. E por fim temos a camada de Storage, onde está situado a base de dados descentralizada do blockchain e um banco de dados transacional para que possamos persistir as ações do usuário no sistema e também fazer a integração de diversas bases de dados para termos os dados do paciente, médico e farmácia, de maneira a facilitar o middleware da aplicação.

Após a estruturação das camadas do sistema bem definidas, podemos então dar início ao levantamento de tecnologias a serem utilizadas no projeto. Iniciando pelo Blockchain temos uma linguagem já bem difundida no mercado de blockchain e em constante crescimento que é o Solidity, com ele podemos escrever smart contracts para a rede de blockchain Ethereum, com esta linguagem nós escrevemos o contrato, compilamos e então é feita a implementação para o blockchain, os dados só podem ser alterados por meio das regras do contrato, é impossível alterá-los de qualquer outra maneira. Com esta decisão já temos duas tecnologias a serem inseridas no diagrama, Solidity e o Ethereum.

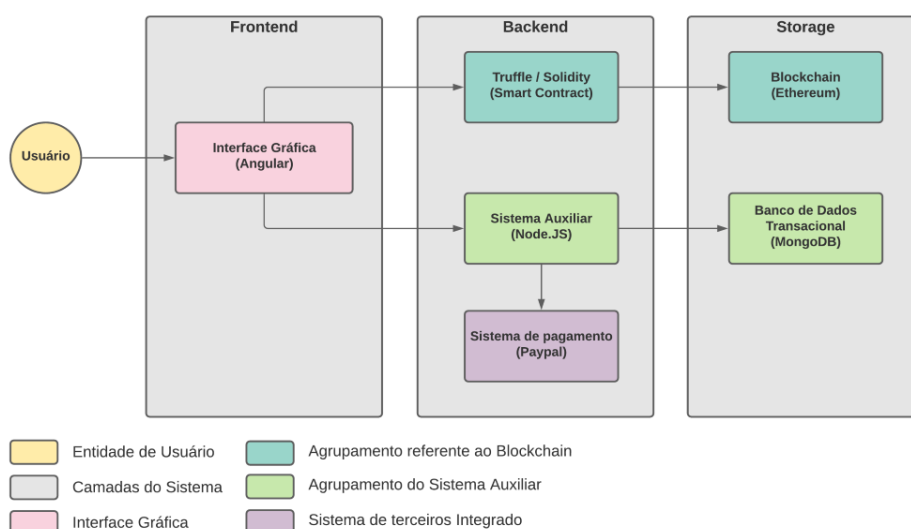
Para a interface gráfica o ideal seria um framework de javascript moderno, tal como Angular, React ou Vue. Foi escolhido o Angular para otimizar o tempo de desenvolvimento, logo que tenho uma melhor afinidade.

O sistema auxiliar é um requisito importante para termos uma sincronia com os dados persistido no blockchain sem que tenhamos dados sensíveis circulando na rede, então o solidity trata das regras de negócio em torno das transações dos contratos e o Backend trabalha em nomear os envolvidos nessas transações e também guardar os pedidos gerados pelo paciente às farmácias cadastradas, com isso para agilizar o desenvolvimento e ainda assim manter um sistema auxiliar forte foi escolhido o Node.JS com a framework Express.js que permite a criação de sistemas com o padrão REST (Representational State Transfer) o qual permite a transferência de dados por meio do protocolo HTTP. Pensando no pagamento das receitas de maneira online foi escolhido a API do Paypal por ter como efetuar testes de maneira gratuita e também ser de fácil utilização.

Já na camada de Storage para o sistema auxiliar foi escolhido o banco de dados não relacional MongoDB, pois teremos um grande fluxo de dados sem a necessidade de entidades relacionadas, logo foi a melhor escolha.

Após o levantamento das tecnologias a serem utilizadas no projeto, por fim temos o diagrama que podemos ver na figura 12.

Figura 12 – Diagrama da arquitetura do projeto com tecnologias utilizadas



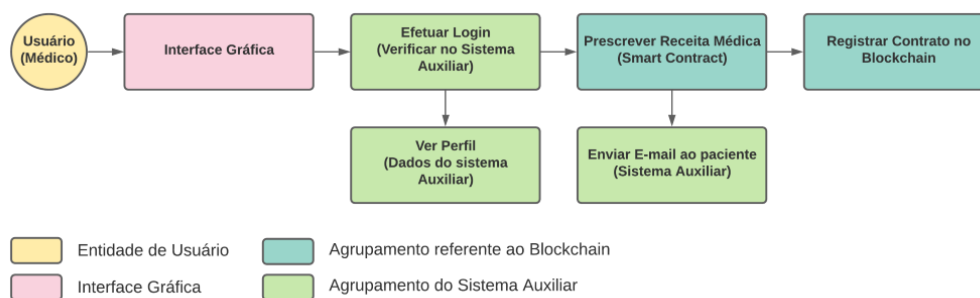
Fonte: Elaborado pelo autor.

4.1.1. Serviços da Entidade Médico

Esta entidade é responsável por prescrever a receita médico ao paciente, recuperando os dados necessários através de integrações com sistemas já existentes, por meio de um trabalho de middleware é possível ter os dados de qualquer paciente e medicamentos oriundos de sistemas distribuídos.

O Médico deverá efetuar o seu login no sistema, por meio de seu CRM e uma senha, após ter seu acesso validado, ele será capaz de ver seus dados na tela de perfil e prescrever uma receita médica na tela de prescrições, a obtenção destes dados será feito por meio do sistema auxiliar que será integrado com quaisquer sistemas que já existam estes dados, ele também será responsável por enviar um hash único e uma senha para o e-mail do paciente, de maneira com que apenas ele consiga retirar a receita médica da farmácia, após o envio do email a aplicação irá persistir os dados necessários no blockchain através do smart contract programado, este processo está sendo ilustrado na figura 13.

Figura 13 – Diagrama de serviço do medico



Fonte: Elaborado pelo autor.

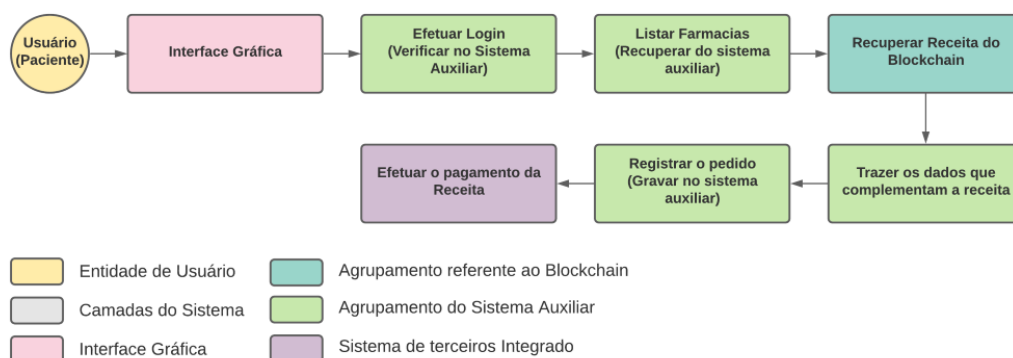
4.1.2. Serviços da Entidade Paciente

Este serviço foi desenvolvido para que a entidade Paciente consiga retirar o receituário médico através do próprio sistema, ele depende que o médico faça

a prescrição e gere est receituário no sistema para que seja persistida dentro do blockchain.

O Paciente irá efetuar seu login no sistema, através de seu e-mail e senha, após a validação de seu acesso, ele será capaz de visualizar todas as farmácias que serão populadas por uma API de um sistema distribuído ou pelo próprio sistema auxiliar, cada farmácia irá listar um ou mais endereços, ao selecionar o endereço desejado o usuário será direcionado para uma tela na qual ele deverá inserir o hash do receituário médico e a senha disponibilizada em seu e-mail, ao preencher estes campos e o receituário for válido estes dados serão mesclados em um novo hash para assim adquirir o verdadeiro hash da receita, assim sendo de acesso exclusivo do portador do e-mail, logo então os dados do receituário serão obtidos diretamente do blockchain, e seus dados extras ou sensíveis serão obtidos por meio do sistema auxiliar. Com o receituário já disposto na tela, se este não for inválido o usuário conseguirá efetuar o pagamento do mesmo através do Paypal, depois que a transação foi feita, este pedido será gerado no sistema auxiliar e assim a farmácia selecionada receberá para assim dar baixa no blockchain e enviar o medicamento ao endereço do paciente, como ilustrado na figura 14.

Figura 14 – Diagrama de serviço do paciente



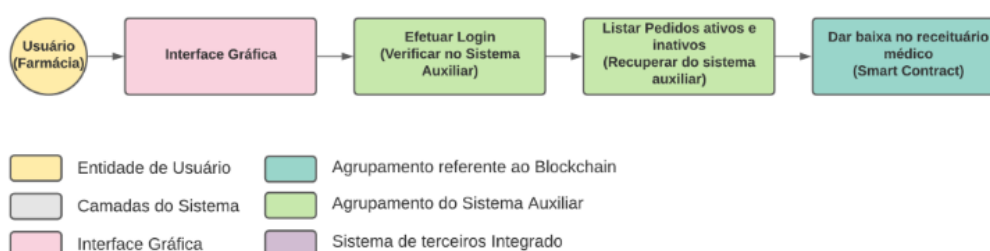
Fonte: Elaborado pelo autor.

4.1.3. Serviços da Entidade Farmácia

Esta Entidade representa todas as ações que o usuário farmácia exercerá no sistema, ele irá listar todos os pedidos já fechados (inativos) e os que ainda não foram fechados (ativos), permitindo ver todos os dados referentes a este pedido, que são eles o endereço do paciente, detalhes do medicamento prescrito e o total da compra, este processos podem ser vistos na figura 15.

O usuário farmácia fica apenas o cargo de receber um pedido e alterá-lo no blockchain através do smart contract para que o mesmo não possa ser utilizado novamente.

Figura 15 – Diagrama de serviço da farmácia

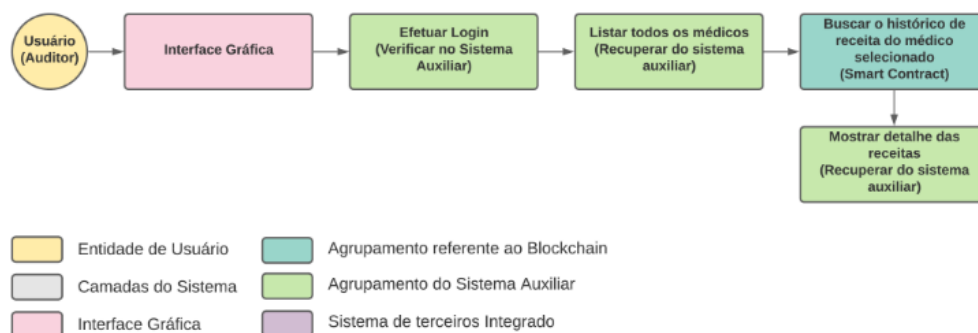


Fonte: Elaborado pelo autor.

4.1.4. Serviços da Entidade Organizacional

Esta entidade tem como atribuições apenas verificar a veracidade de cada contrato emitido por um profissional da saúde, assim o usuário será capaz de trazer um histórico de receitas emitidos diretamente do blockchain para verificar possíveis discrepâncias de maneira manual, além das validações que são feitas pelos nós da rede, impedindo qualquer manipulação de um dado específico, como ilustra o diagrama da figura 16.

Figura 16 – Diagrama de serviço da Organização



Fonte: Elaborado pelo autor.

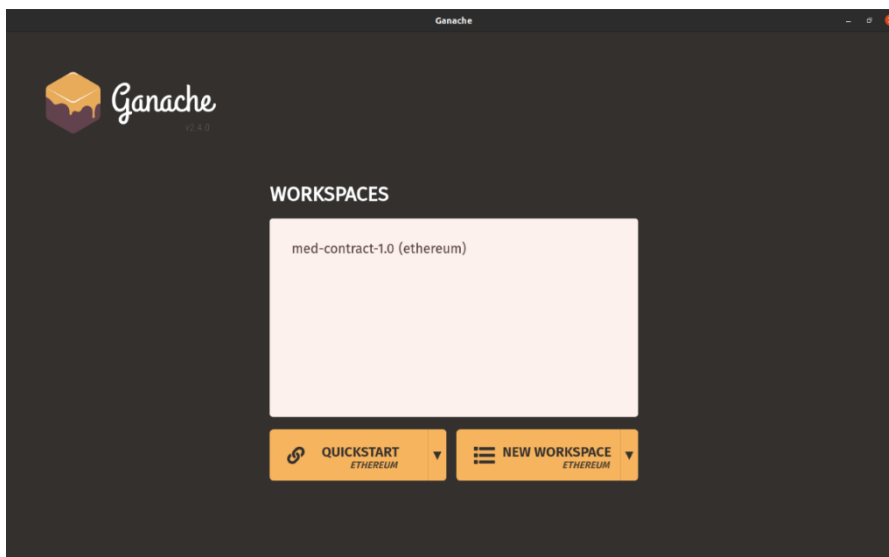
4.2. Desenvolvimento do Smart Contract

Com os requisitos do projeto já consolidados foi dado início na parte de criação do Smart Contract para que possamos definir a interação do usuário dentro do blockchain, tudo o que for programado aqui terá que ser feito para que a rede Ethereum valide a transação com sucesso, qualquer outro tipo de manipulação destes dados será invalidados dentro da rede, assim impossibilitando que o medicamento seja retirado da farmácia.

4.2.1. Criando a rede Ethereum Local

O primeiro passo para o desenvolvimento do Smart Contract é o setup da rede Ethereum a ser utilizada, para isso foi escolhido o Ganache, que pode ser visto na figura 17, oferecido pelo Truffle Suite, o qual disponibiliza uma gama de ferramentas para o desenvolvimento de Smart Contract, o Ganache em si é uma interface gráfica em que é possível ver todas as contas da rede, transações e blocos gerados, simulando uma rede Ethereum real.

Figura 17 – Criação da rede blockchain local com Ganache



Fonte: Elaborado pelo autor.

Ao Criar a rede local pode-se notar algumas opções que remetem ao conceito do blockchain, em que temos contas de usuários, geralmente chamadas de carteira, blocos que indicam uma nova transação que precisa ser minerada pelos computadores nós da rede, transações que são as interações das contas seguindo o modelo programado no Smart Contract, os contratos que foram implementados na rede, eventos disparados pelos contratos e os logs de tudo que for feito dentro da sessão aberta do ganache, a figura 18 ilustra a barra de ferramentas que podem ser utilizado no Ganache.






Figura 18 – Menu do Ganache



Fonte: Elaborado pelo autor.

Na aba de Accounts, representada na figura 19, temos todos os dados das contas utilizadas no sistema, o Ganache já cria um número padrão de contas para propósitos de desenvolvimento, cada conta é composta por um endereço, balanço da moeda ETH nesta conta, número de transações feitas, índice e a chave privada.

Figura 19 – Contas na rede Ethereum local

ADDRESS 0x77B6dfBB2310ab552619a2517897B7474217faFA	BALANCE 99.96 ETH	TX COUNT 15	INDEX 0	
ADDRESS 0xFB3DDE836a38f1a744BEE6C3b123cEdE88DB9774	BALANCE 100.00 ETH	TX COUNT 0	INDEX 1	
ADDRESS 0x6E79555a146A60DF36d0F83e43B46c84527a09a8	BALANCE 100.00 ETH	TX COUNT 0	INDEX 2	
ADDRESS 0x283CE91f9bDd81Cde62647cCd248A213166b2B9	BALANCE 100.00 ETH	TX COUNT 0	INDEX 3	
ADDRESS 0xD00890537DD78185dd9e80E6aEcbd681c2f9C684	BALANCE 100.00 ETH	TX COUNT 0	INDEX 4	

Fonte: Elaborado pelo autor.

Estas contas serão feitas para cada usuário que utilizara o sistema, para ter como guardar a transação por contas além dos dados do contrato, assim garante um bom rastreio de quem está emitindo e recebendo os receituários por meio de procura na própria rede do blockchain.

Além desta aba, outro importante é a que registra as transações que estão ocorrendo na rede, como pode ser visto na figura 20, por meio dessa podemos ver os parâmetros que estão sendo passados por cada contrato e também quais funções estão sendo utilizadas e quais endereços executaram esta transação.

Figura 20 – Transações na rede Ethereum

TX HASH 0xae4a9151442db14822e3dd50e9042bb7d5b9f5cbaa748f7843b7c8d27413b782	TO CONTRACT ADDRESS ReceitaMedica	GAS USED 42977	VALUE 0	CONTRACT CALL
FROM ADDRESS 0x77B6dfBB2310ab552619a2517897B7474217faFA				
TX HASH 0xca75f35634bbf9d5f07f238b2ca1f0e7e76e17ef68c1f01ed0c57edaad7705db	TO CONTRACT ADDRESS ReceitaMedica	GAS USED 139786	VALUE 0	CONTRACT CALL
FROM ADDRESS 0x77B6dfBB2310ab552619a2517897B7474217faFA				
TX HASH 0x620470a7cf0457030899d0bb178aa39d353cd5fc0da2f99753d0a0acb979f39a	TO CONTRACT ADDRESS ReceitaMedica	GAS USED 139786	VALUE 0	CONTRACT CALL
FROM ADDRESS 0x77B6dfBB2310ab552619a2517897B7474217faFA				
TX HASH 0xf3f62f6277368fd9ff7de883cf190720c48e2b02f0304813d802fa5af256f55e	TO CONTRACT ADDRESS ReceitaMedica	GAS USED 42977	VALUE 0	CONTRACT CALL
FROM ADDRESS 0x77B6dfBB2310ab552619a2517897B7474217faFA				
TX HASH 0x86afb89ee6226e4afd7f77f600db4af5c52ac9a581e01efe95a6e69a49e8884d	TO CONTRACT ADDRESS ReceitaMedica	GAS USED 139786	VALUE 0	CONTRACT CALL
FROM ADDRESS 0x77B6dfBB2310ab552619a2517897B7474217faFA				

Fonte: Elaborado pelo autor.

Pode-se também ver os detalhes de cada transação, demonstrado pela figura 21, tal como quais parâmetros foram passados pelo mesmo, e o hash final que será gerado para a validação das máquinas, o qual resulta em um endereço

mnemônico para rápido entendimento das máquinas, também é possível ter acesso ao GAS utilizado na transação, pois cada validação dentro da rede tem um custo de ETH para que a rede seja mantida.

Figura 21 – Transações na rede Ethereum

[illegible]

Fonte: Elaborado pelo autor.

Na aba de Contratos do Ganache pode-se ver tudo que for referente ao Smart Contract, desde o endereço deste contrato q será utilizado pelos nós para validar quaisquer transações que sejam feitas dentro da rede quanto os dados que podem ser recuperados por este contrato, além disso é possível ver um histórico de transações realizado com este contrato, estas informações podem ser vistas na figura 22.

Figura 22 – Contrato implementado no Ganache

The screenshot displays a web application for a smart contract named "ReceitaMedica". At the top, a dark header bar contains various network and contract parameters: CURRENT BLOCK 15, GAS PRICE 2000000000, GAS LIMIT 6721975, HARDFORK MURGELACHER, NETWORK ID 5777, RPC SERVER HTTP://127.0.0.1:7545, MINING STATUS AUTOMINING, WORKSPACE MED-CONTRACT-1.0, and buttons for SWITCH and settings. Below the header, the contract name "ReceitaMedica" is prominently displayed with a back button. The main content area is divided into three sections: 1. Contract Details: Shows the ADDRESS (0xc7f623c0632636e1f041020d5e713337124c7609) and BALANCE (0.00 ETH), and the CREATION TX (0x39206050f1364a0d86dFa4EC04b852376cf0Eb2C5f64eC60760B9FF28248dc32). 2. STORAGE: Displays a JSON object representing the contract's state: { 5 items, receitasCount: uint:0, receitaById: {} mapping @ items, receitaByHashId: {} mapping @ items, hashMap: {} mapping @ items, medicHistoryByHashId: {} mapping @ items }. 3. TRANSACTIONS: Lists two transactions. The first transaction has TX HASH 0x86afb89ee6226e4afd7f77f600db4af5c52ac9a581e01efe95a6e69a49e8884d, FROM ADDRESS 0x77B0dfBB2310ab552619a2517897B7474217FaFA, TO CONTRACT ADDRESS ReceitaMedica, GAS USED 139786, and VALUE 0. The second transaction has TX HASH 0xf3f62f6277368fd9ff7de883cf190720c48e2b02f0304813d802fa5af256f55e. Both transactions have a "CONTRACT CALL" button next to them.

Fonte: Elaborado pelo autor.

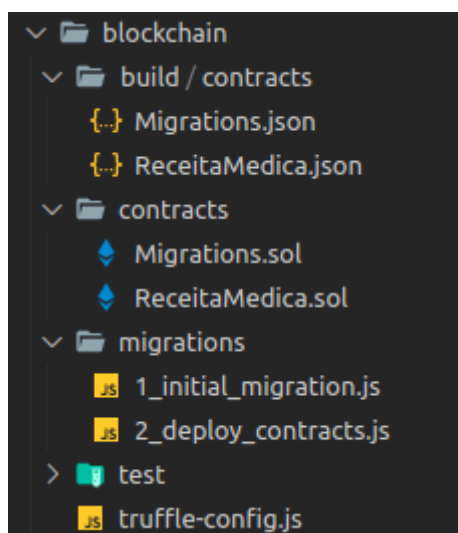
4.2.2. Desenvolvendo o Smart Contract

Para o desenvolvimento do Smart Contract foi escolhido a linguagem de programação Solidity, em que é uma linguagem de programação orientada a objetos para escrever Smart Contracts.

Esta linguagem é utilizada para o desenvolvimento do Smart contract em várias plataformas de blockchain, nesse projeto ele será escrito para o Ethereum.

Para o desenvolvimento dos Smart Contracts foi utilizado uma pasta para armazenar toda a lógica e configurações, como pode ser visto na figura 23.

Figura 23 – Estrutura de pastas do Smart Contract



Fonte: Elaborado pelo autor.

Neste diretório temos o arquivo de configuração na raiz, onde está o apontamento para a rede local blockchain que foi criado utilizando o Ganache fornecido pelo Truffle Suite, este arquivo é o truffle-config.js, demonstrado na figura 24.

Figura 24 – Configuração da rede Blockchain local

```

networks: {
  // Useful for testing. The `development` name is special - truffle uses it by
  // default
  // if it's defined here and no other network is specified at the command line
  // You should run a client (like ganache-
  // cli, geth or parity) in a separate terminal
  // tab if you use this network and you must also set the `host`, `port` and `
  // network_id`
  // options below to some value.
  //
  development: {
    host: "127.0.0.1",    // Localhost (default: none)
    port: 7545,          // Standard Ethereum port (default: none)
    network_id: "*",     // Any network (default: none)
  },
}

```

Fonte: Elaborado pelo autor.

Dentro do diretório contracts estão os arquivos solidity que contém toda a lógica de nossos contratos programados, todo arquivo precisa especificar a versão utilizada para que o compilador mantenha sempre esta versão, pois como é

uma linguagem recente ela pode ser submetida à várias alterações que podem impactar códigos escritos em versões mais antigas.

Após especificar a versão do solidity utilizada, então é necessário escrever o nome do contrato, pelo qual irá identificar todas as transações e validações na rede, dentro deste contrato vem toda a lógica necessária para o funcionamento do mesmo, assim como a estrutura de dados utilizada para armazenar os dados do contrato no blockchain, esta estrutura pode ser vista na figura 25.

A estrutura da Receita foi desenvolvida pensando na prescrição do médico, levando em consideração os principais dados necessários para garantir que ambas as partes estejam neste contrato, estes dados são hashId necessário para identificar o contrato, crm do médico, cpf do paciente, código do medicamento prescrito e o status da receita, outra estrutura a ser utilizada é a que garante o histórico de emissão de receitas médicas do médico, ela foi pensada para otimizar a busca deste histórico dentro do blockchain, pois cada transação dentro da rede tem um custo que pode ser maior quanto mais operações de iteração for feita, logo é recomendado guardar uma lista inteira ao invés de buscar os dados por uma query, esta estrutura armazena apenas o hashId do médico e a lista de hashes das receitas emitidas.

Figura 25 – Estrutura do Smart Contract

```
pragma solidity >=0.4.2;
pragma experimental ABIEncoderV2;

contract ReceitaMedica {

    struct Receita {
        uint id;
        bytes32 hashId;
        string crm;
        string cpf;
        string codMed;
        bool status;
    }

    struct MedicHistory {
        bytes32 hashId;
        bytes32[] receitas;
    }

    uint public receitasCount;
```

Fonte: Elaborado pelo autor.

As propriedades de mapping da receita servem para mapear a estrutura de Receita por seus atributos, como mostrado na figura 26, logo conseguimos definir que um hash de 32 bytes pode identificar uma receita dentro da estrutura especificada no blockchain.

Além destas estruturas temos um uint feito exclusivamente para executar uma contagem encima das receitas emitidas.

Figura 26 – Mapeamento das Estruturas

```
mapping(uint => Receita) receitaById;
mapping(bytes32 => Receita) receitaByHashId;
mapping(bytes32 => uint) hashMap;

mapping(bytes32 => MedicHistory) medicHistoryByHashId;
```

Fonte: Elaborado pelo autor.

Em seguida tem a seção das lógicas programadas do contrato, estas funções consistem do identificador function, nome da função, parametros a serem utilizados, visibilidade desta função dentro do ambiente e o que esta função retorna.

A lógica mais importante deste contrato é a prescreverReceita, que recebe os parâmetros da estrutura Receita e armazena os dados especificados nos parâmetros dentro do blockchain, quando utilizado o mapping dentro desta função é possível definir o atributo de acesso que será o hashId da Receita, logo armazenamos todos os parâmetros dentro do bloco com o identificador de hashId que depois poderá ser lido ao especificá-lo, nesta mesma lógica é escrito que o hashId da Receita será armazenado também na lista de prescrições do médico, garantindo um histórico de receituários médicos emitidos.

Para impedir que mesma receita seja utilizada mais de uma vez foi feito um método para apenas encontrar a receita e alterar o seu boolean de false para true, assim indicando que esta receita já está em uso, este método foi nomeado de retirarReceita.

Os seguintes métodos, que podem ser vistos na figura 27, foram desenvolvidos para acessar as receitas dentro dos blocos através de uma propriedade definida, estes métodos foram nomeados como `getPrescricaoByCrm`, `getPrescricaoById` e `getMedicHistory`.

Figura 27 – Métodos do Smart Contract

```
function prescreverReceita(bytes32 _hashId, bytes32 _medicHashId,
    string memory _codMed, string memory _medCrm, string memory _pacCpf) public
{
    receitaByHashId[_hashId] = Receita(receitasCount, _hashId, _medCrm, _pacCpf, _codMed, false);
    medicHistoryByHashId[_medicHashId].hashId = _medicHashId;
    medicHistoryByHashId[_medicHashId].receitas.push(_hashId);
}

function retirarReceita(bytes32 _hashId) public {
    receitaByHashId[_hashId].status = true;
}

function getPrescricaoByCrm(bytes32 _hashId) public view returns(Receita memory) {
    return receitaByHashId[_hashId];
}

function getPrescricaoById(uint _id) public view returns(Receita memory) {
    return receitaById[_id];
}

function getMedicHistory(bytes32 _medicHashId) public view returns(MedicHistory memory) {
    return medicHistoryByHashId[_medicHashId];
}
}
```

Fonte: Elaborado pelo autor.

4.2.3. Compilando o código

O código do smart contract necessita ser compilado para que a rede Ethereum entenda o que cada linha de código escrito com o solidity, com o Truffle Suite basta acessar o código pelo CLI (Interface de Linha de Comando) e rodar o comando para compilar, após o comando ser executado será criado um

arquivo dentro da pasta build com o mesmo nome dos contratos escritos em solidity, porém, com a extensão .json.

Este arquivo de build é utilizado para executar as transações no blockchain diretamente pela interface, pois ele trabalha como um artefato de código, sinalizando as operações e as estruturas que são permitidas no sistema.

O compilador do truffle transforma todas as instruções escritas em alto nível para bytecodes, um tipo de linguagem de baixo nível que as máquinas entendem, como demonstrado na figura 28.

Figura 28 – Representação parcial do Bytecode

```
"bytecode": "0x608060405234801561001057600080fd5b50610e09806100206000396000f3fe
608060405234801561001057600080fd5b50600436106100625760003560e01c80631628080
c1461006757806347927846146100975780634b07548e146100b5578063575e9960146100e5
578063925fed6d14610101578063f9f07c681461011d575b600080fd5b61008160048036036
1007c9190810190610948565b61014d565b60405161008e9190610c3a565b60405180910390
f35b61009f61038f565b6040516100ac9190610c5c565b60405180910390f35b6100cf60048
036036100ca9190810190610a30565b610395565b6040516100dc9190610c3a565b60405180
910390f35b6100ff60048036036100fa9190810190610948565b6105d7565b005b61011b600
48036036101169190810190610971565b610609565b005b6101376004803603610132919081..."
```

Fonte: Elaborado pelo autor.

Para que este código seja compilado o Truffle Suite tem um comando específico para isso, navegando até o diretório raiz do Smart Contract é possível executar o comando `truffle compile` para que todas as instruções do Smart Contract seja transformada em uma estrutura json com seus respectivos bytecodes.

4.2.4. Implementação do código na rede blockchain

Após a fase de desenvolvimento do Smart Contract é necessário que haja uma implementação do código já compilado para a rede blockchain, como a rede blockchain está rodando de maneira local para fins de desenvolvimento, o truffle suite há um comando para que o código compilado seja transferido para a rede.

O Truffle Suite tem um método chamado Migration (Migração) que foi desenvolvido para preparar o código para implementação na rede blockchain, representado na figura 29, isso foi pensado visando que o código sofrerá várias alterações de acordo com o tempo, então este processo pode ser feito sempre que for preciso atualizar o código na rede, sem que precise remover implementações antigas, para que o código seja migrado com sucesso é preciso escrever uma instrução na pasta do projeto.

Figura 29 – Instrução para implementação do Smart Contract

```
var ReceitaMedica = artifacts.require("./ReceitaMedica.sol")

module.exports = function(deployer) {
  deployer.deploy(ReceitaMedica);
};
```

Fonte: Elaborado pelo autor.

4.3. Desenvolvimento da Interface Gráfica

O principal ponto a se considerar no projeto é a interface gráfica, na qual deve ser de fácil entendimento e a sua navegação de maneira intuitiva, para que o usuário tenha uma melhor experiência na utilização do sistema.

Antes de iniciar o desenvolvimento da aplicação, foi pensando em como otimizar o tempo do profissional entre cada emissão de receitas médicas, imaginando que um médico pode emitir várias receitas dentro de um dia de trabalho, a interface de usuário deve ser de fácil acesso e utilização, logo foi pensando em uma alternativa WEB, para que o profissional apenas abra seu navegador, efetue o login e comece a utilizar.

Além de otimização de tempo a interface gráfica teve um cuidado maior com a experiência do usuário na aplicação, tendo as informações na tela de maneira clara e sem muitos textos que podem confundir o usuário durante a sua sessão.

Este projeto também teve um cuidado para com a acessibilidade de pessoas com idade avançada, com ícones grandes e textos em cores que chamam a atenção para a ação que ele executará.

4.3.1. Layout minimalista

O principal aspecto a ser considerado no desenvolvimento da aplicação foi seguir um layout minimalista para que o usuário tenha uma boa experiência de navegação, sem muitos detalhes na tela que possam vir a confundi-lo, esta confusão pode gerar possíveis erros, principalmente de entendimento, de acordo com a neurocientista cognitiva americana Maryanne Wolf (O Cérebro no Mundo Digital - Os desafios da leitura na nossa era; ed. Contexto) as pessoas estão cada vez mais apenas passando pelas palavras e tentando identificar palavras chaves para que elas compreendam o contexto e não o texto em si.

Logo textos mais curtos que indicam as ações no sistema facilitam o entendimento, pois caso tenham vários textos distribuídos pela tela, o usuário não irá levar o tempo total de leitura para entender o que está escrito na tela.

Então um Layout mais minimalista, rico em informação visual, sem textos com fontes pequenas é o ideal para que o usuário entenda o que cada ação do sistema faz, evitando com que ele cometa um equívoco sobre a ação que deseja fazer, pensando nisso.

4.3.2. Redução de erros com base no layout

O layout segue um conceito de centralizar o sistema no usuário, de maneira com que ele se sinta confortável ao utilizá-lo, logo temos padrões de fontes, com tamanhos específicos para garantir uma fácil leitura, disposição de elementos de fácil acesso.

- Fontes de texto de no mínimo 12 pixels, com este tamanho de fonte é possível garantir uma maior compreensão das ações que o usuário deve efetuar no sistema.

- Botões padronizados para fácil identificação, o usuário deve ver o botão e já entender que ele deve ser pressionado, caso a ação precise estar inativa até que algum campo seja preenchido, o botão deverá estar ofuscado e indicar que a ação não poderá ser executada.
- O menu é de fácil acesso para que o usuário sempre consiga navegar ou efetuar o logout da aplicação a qualquer momento, sem poluir a tela com um menu maior do que o necessário.

Ao seguir os padrões propostos, o sistema tem as suas funcionalidades mais acessíveis a qualquer momento, deixando explícito qual é o papel de cada ação.

Sistemas grandes com vários textos e campos de formulários podem estressar o usuário, pois quanto mais ações forem exigidas para atingir um objetivo, mais dados devem ser levantados pelo usuário, assim aumentando o estresse e podendo levar a algum erro de digitação o abandono da aplicação durante algum cadastro.

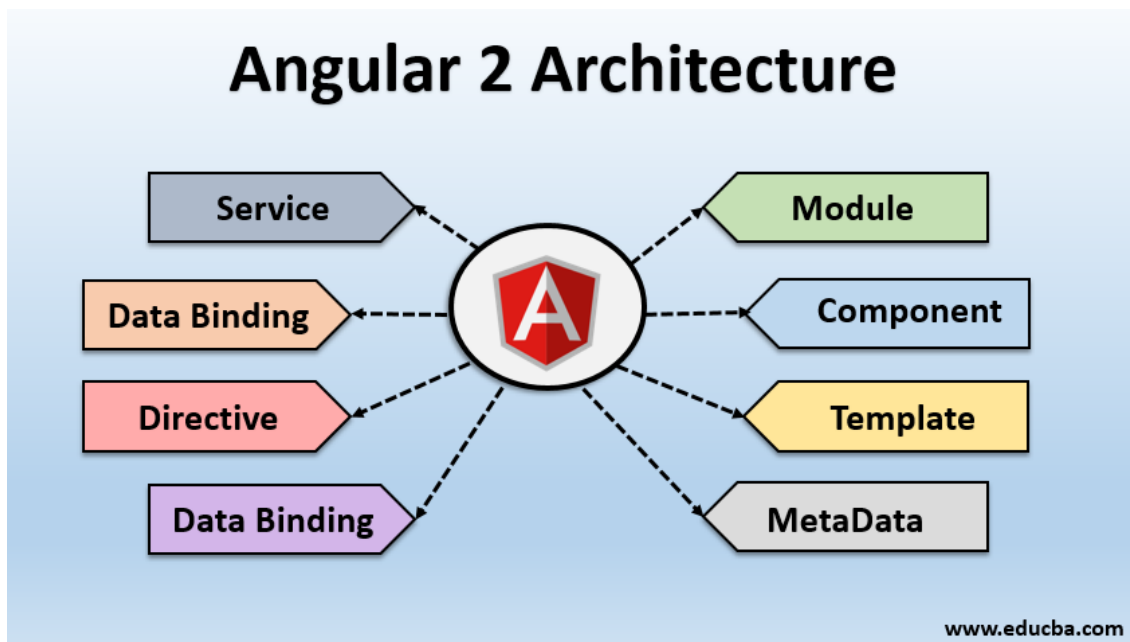
Ao se utilizar os conceitos apresentados acima, o sistema pode se beneficiar de várias maneiras, se o médico precisar apenas trazer o nome do paciente, medicamento a ser prescrito e a prescrição de uso, nós limitamos apenas a 3 campos de inserção, este método evita o estresse do profissional da saúde durante a prescrição, pois ele não precisa inserir a dosagem, endereço do paciente, documento, etc. Tendo o paciente já na tela sem precisar digitar mais campos, também garante com que ele consiga identificar o paciente mais rápido, neste caso o componente de typeahead, onde lista todos os pacientes conforme a digitação, facilita a procura do mesmo.

4.3.3. Criando o projeto em Angular

A framework de interface gráfica angular foi utilizada no projeto pela a sua baixa curva de aprendizado e diversos benefícios, que podem ser vistos na figura 30, já integrados com a framework, dentro destes benefícios já está a integração

do javascript, SCSS para estilo, linguagem de marcação HTML e diversos pacotes do núcleo angular, para requisições HTTP e navegação SPA (Single Page Application - aplicação de página única), evitando que várias páginas sejam carregadas a cada navegação no sistema, com isso apenas o conteúdo da página se altera.

Figura 30 – Arquitetura do Angular 2+



Fonte: www.educba.com.

O projeto em angular necessita da instalação do Node.js e NPM (Node Package Manager – Gerenciador de pacotes node), pois ele contém um CLI (Command Line Interface – Interface de Linha de Comando) para a sua instalação e utilização.

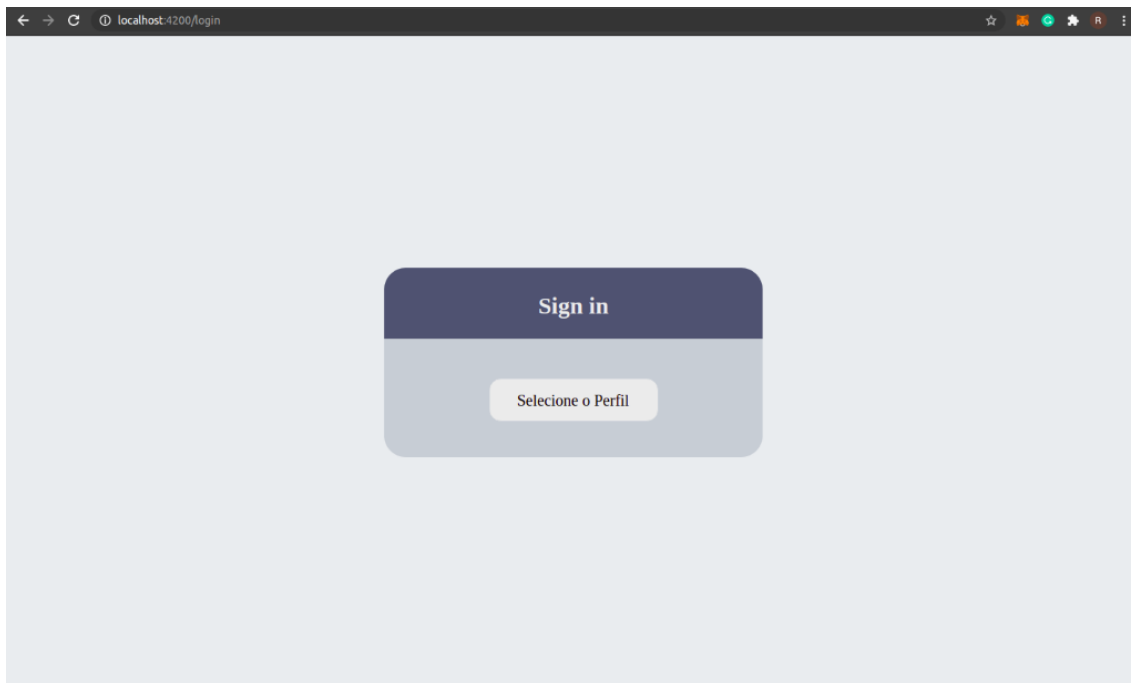
O angular utiliza o Typescript para seu desenvolvimento, o qual é uma linguagem parecida com o Javascript porém com uma tipagem mais forte e podendo ser facilmente trabalhado com o padrão Orientado a Objetos.

Após o desenvolvimento da aplicação todo o código escrito em HTML, SCSS e Typescript será compilado para HTML, CSS e javascript, sendo facilmente compreendido pelo navegador.

4.3.3.1. Desenvolvimento do Login

O login do sistema, demonstrado na figura 31, consiste de um layout padrão, onde se deve escolher o tipo de usuário e em seguida será exibido os dados necessários para que seja efetuado o Login no sistema.

Figura 31 – Tela de Login

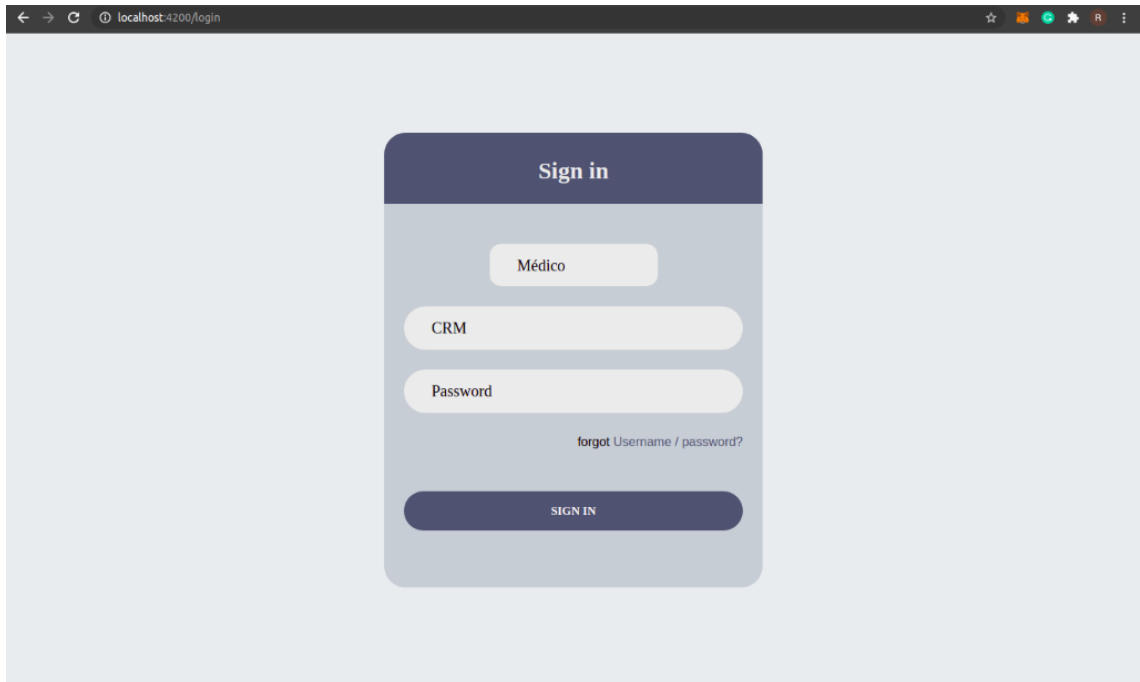


Fonte: Elaborado pelo autor.

O Sistema contém quatro tipos de usuários, logo será possível selecionar apenas uma opção dentro destas.

Ao selecionar o usuário Médico, irá aparecer dois campos para inserir os dados da CRM e password, além do botão de login e um link para caso o usuário esqueça a sua senha, como demonstrado na figura 32.

Figura 32 – Tela de Login do médico

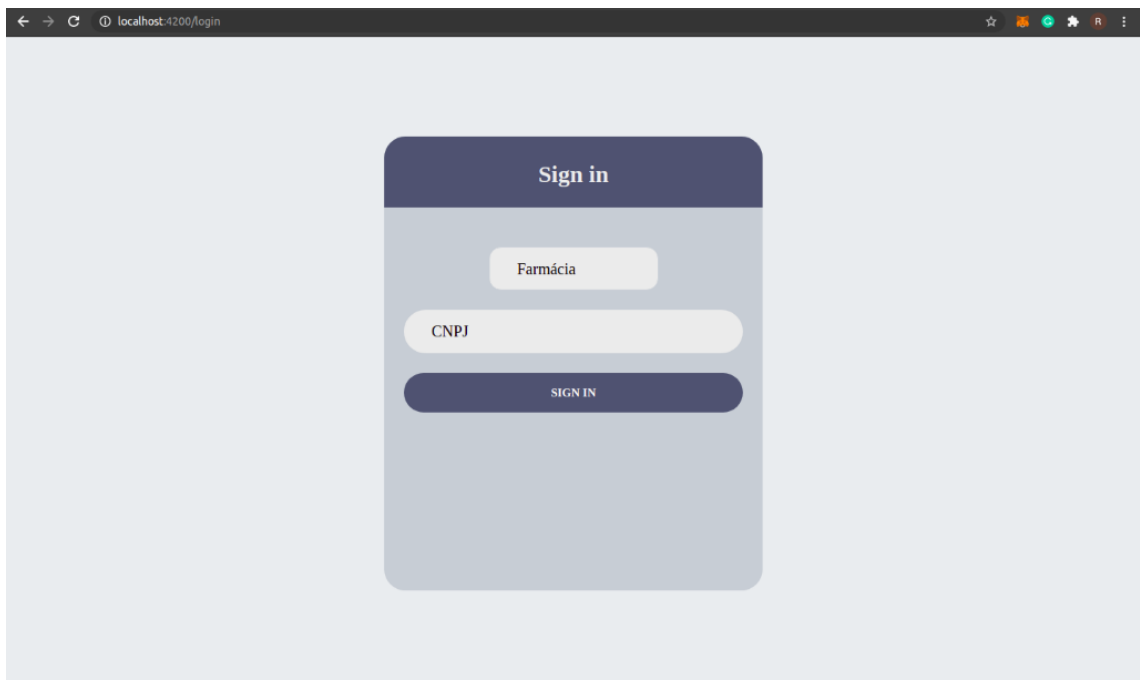


The screenshot shows a web browser window with the address bar displaying 'localhost:4200/login'. The main content is a 'Sign in' form with a dark blue header. The form contains three input fields: 'Médico', 'CRM', and 'Password'. Below the 'Password' field is a link that says 'forgot Username / password?'. At the bottom of the form is a dark blue button labeled 'SIGN IN'.

Fonte: Elaborado pelo autor.

O Usuário Farmácia contém um campo a mais para identificar o cnpj da loja, para que assim seja possível logar o usuário referenciando a sua loja, como pode ser visto na figura 33.

Figura 33 – Tela de Login da loja do usuário farmácia

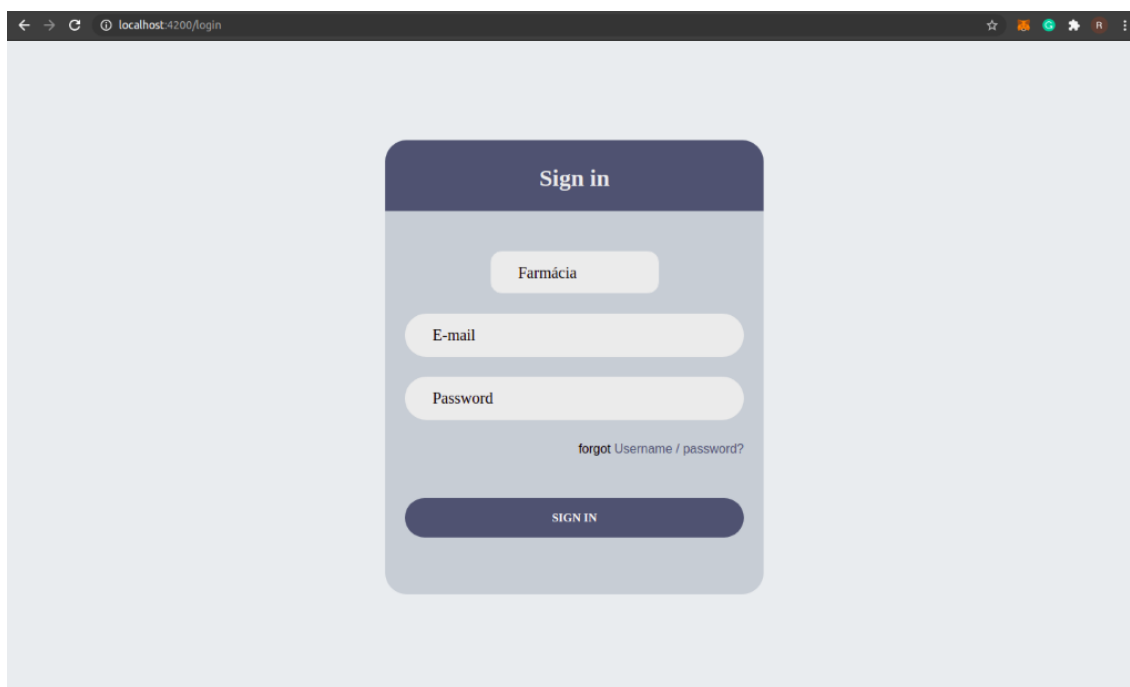


The screenshot shows a web browser window with the address bar displaying 'localhost:4200/login'. The main content is a 'Sign in' form with a dark blue header. The form contains two input fields: 'Farmácia' and 'CNPJ'. Below the 'CNPJ' field is a dark blue button labeled 'SIGN IN'.

Fonte: Elaborado pelo autor.

Após o usuário digitar o cnpj de sua loja, irá aparecer dois novos campos para serem preenchidos, o e-mail e senha do respectivo usuário vinculado a esta loja, como demonstrado na figura 34.

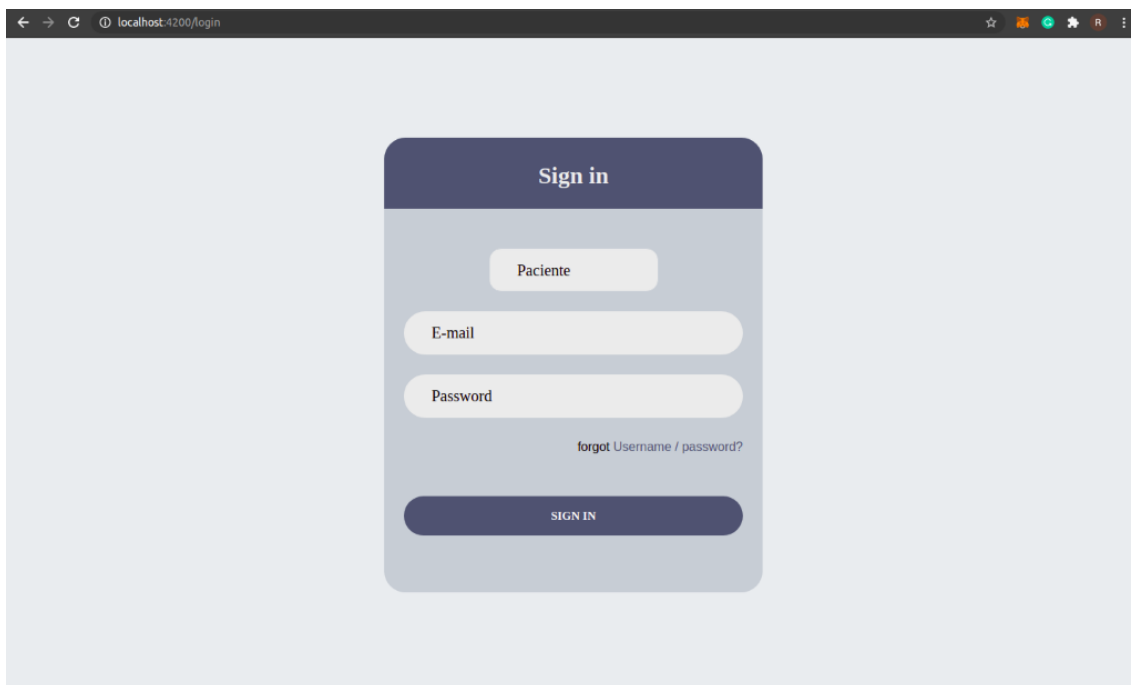
Figura 34 – Tela de Login do usuário farmácia

The image shows a web browser window with the address bar displaying 'localhost:4200/login'. The main content area features a 'Sign in' form. The form has a dark blue header with the text 'Sign in'. Below the header, there are three input fields: the first is labeled 'Farmácia', the second is labeled 'E-mail', and the third is labeled 'Password'. Below these fields, there is a link that says 'forgot Username / password?'. At the bottom of the form is a dark blue button with the text 'SIGN IN' in white capital letters.

Fonte: Elaborado pelo autor.

O Usuário Paciente contém dois campos para que consiga efetuar o seu login, esses campos são e-mail e senha que pode ser visto na figura 35.

Figura 35 – Tela de Login do usuário paciente



← → ↻ ⓘ localhost:4200/login ☆ 📱 🌐 ⚙️ R ⋮

Sign in

Paciente

E-mail

Password

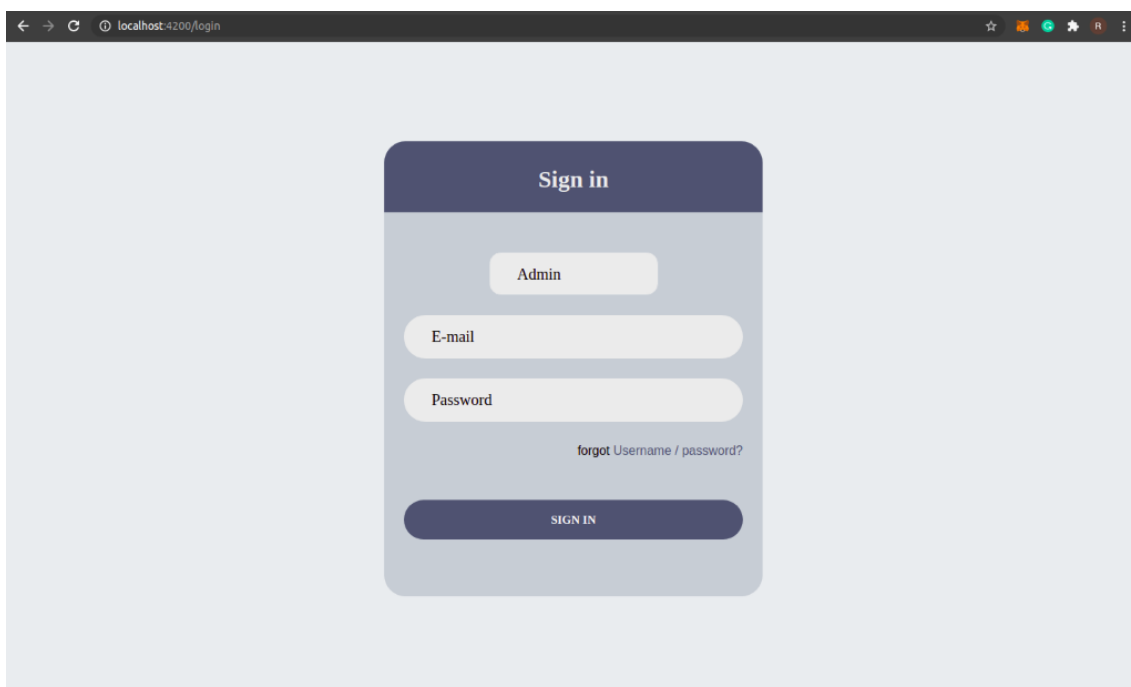
[forgot Username / password?](#)

SIGN IN

Fonte: Elaborado pelo autor.

O usuário Admin ou organizacional necessita do e-mail e senha para efetuar o seu login no sistema, como demonstrado na figura 36.

Figura 36 – Tela de Login do usuário Admin



← → ↻ ⓘ localhost:4200/login ☆ 📱 🌐 ⚙️ R ⋮

Sign in

Admin

E-mail

Password

[forgot Username / password?](#)

SIGN IN

Fonte: Elaborado pelo autor.

4.3.3.2. Desenvolvimento da tela do Médico

A tela do usuário médico foi desenvolvida para ser simples e de fácil acesso, ao efetuar o login no sistema o usuário poderá ver o formulário de prescrição médica, o qual contém todos os dados que serão utilizados para a emissão da receita médica.

Na tela de prescrição, demonstrada na figura 37, temos dois componentes typeahead, responsável por listar todos os pacientes e medicamentos de acordo com o que o médico for digitando, executando um filtro, após a seleção destes campos, temos o endereço do paciente, dosagem do medicamento e descrição, que serão preenchidos automaticamente, além destes campos há também a descrição do tratamento que será enviado no corpo do e-mail do paciente pelo sistema auxiliar.

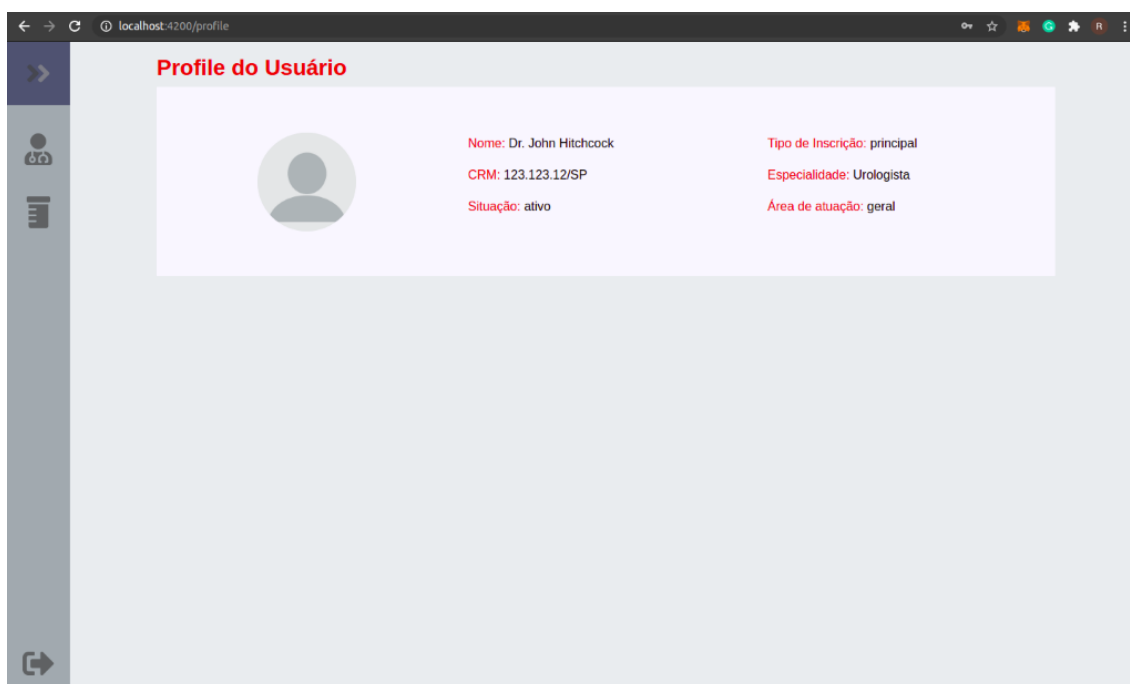
Figura 37 – Tela de prescrição da receita médica

A imagem mostra uma interface web para a emissão de receitas médicas. O formulário principal, com o título "Prescrição Médica", está centralizado e possui um fundo claro. Ele contém os seguintes campos de entrada: "Nome do Paciente", "Medicamento", "Endereço do paciente", "Quantidade" e "Descrição do Medicamento". Abaixo desses, há um campo maior para a "Descrição do tratamento:". Um botão escuro com o texto "Gerar Prescrição" está posicionado na base do formulário. À esquerda, uma barra lateral cinza contém ícones para navegação. O navegador no topo da janela indica o endereço "localhost:4200/prescription".

Fonte: Elaborado pelo autor

Além desta tela, o usuário médico contém uma representação dos detalhes de seu usuário, como visto na figura 38, listando seu nome, CRM, tipo de inscrição, especialidade, área de atuação e a situação de seu cadastro.

Figura 38 – Tela de profile do médico.



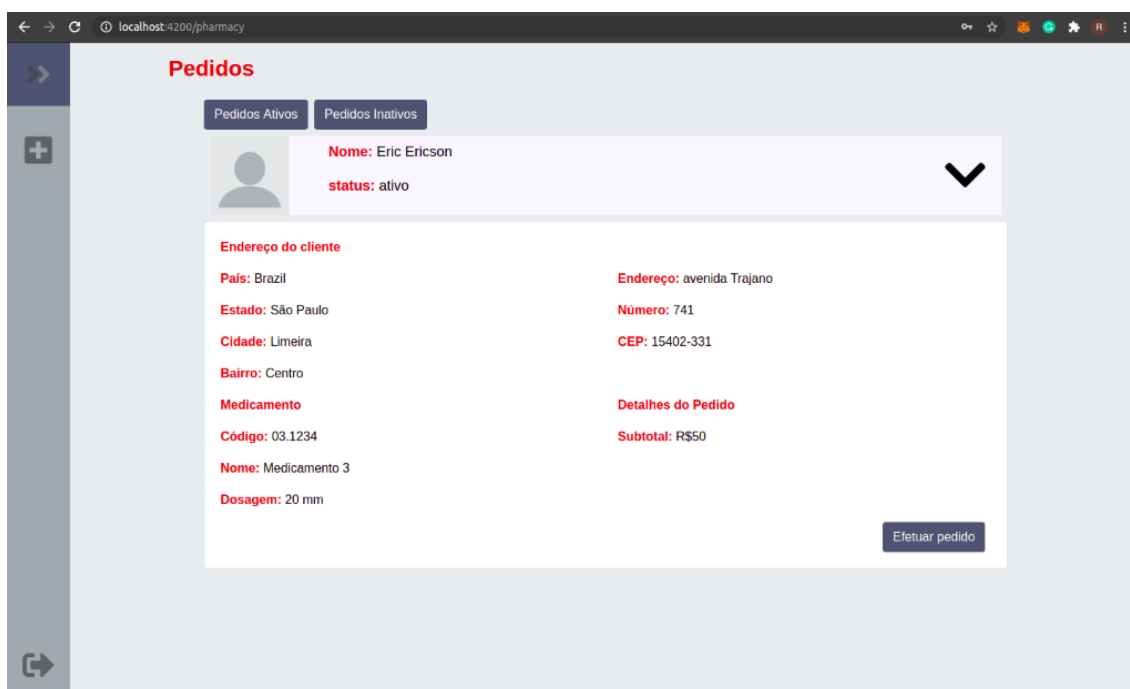
Fonte: Elaborado pelo autor.

4.3.3.3. Desenvolvimento da tela da farmácia

A tela do usuário farmácia é responsável pelo controle de pedidos gerados para o estabelecimento, ele apenas lista os novos pedidos e também todos que já foram finalizados.

Para cada paciente que faz o pedido de um medicamento é gerado uma linha com o nome e status do pedido, ao clicar na seta ao lado direito é então mostrado os detalhes deste pedido, como demonstrado na figura 39, com endereço, medicamento e o subtotal do pedido, seguido do botão que finaliza o mesmo.

Figura 39 – Tela de pedidos da farmácia



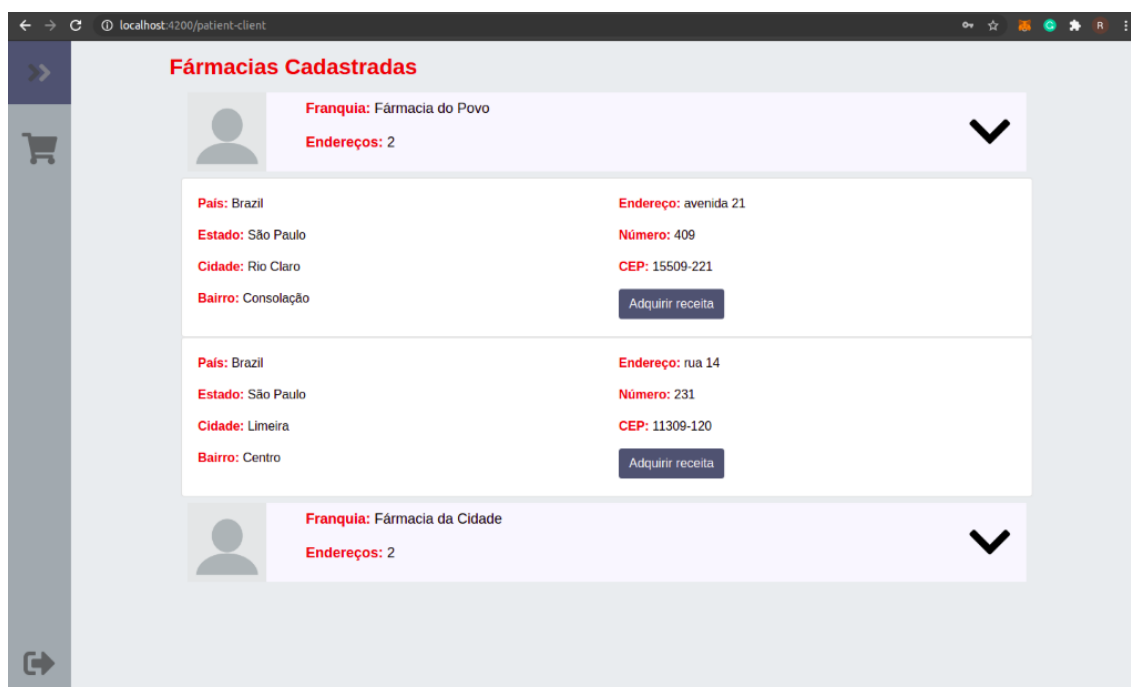
Fonte: Elaborado pelo autor.

4.3.3.4. Desenvolvimento da tela do paciente

Esta tela contém todas as farmácias no qual o paciente poderá efetuar a compra de seu medicamento, onde cada farmácia pode ter um ou mais endereços, fica a critério do paciente escolher a farmácia de sua preferência.

As farmácias são mostradas em formato de linhas, como mostrado na figura 40, onde cada uma mostra uma foto, nome da franquia e quantidade de endereços da mesma, ao expandir os detalhes clicando na seta ao lado direito, é possível visualizar os dados referentes a cada endereço desta farmácia e também adquirir a receita, caso você tenha gerado uma durante uma consulta médica.

Figura 40 – Tela de listagem das farmácias

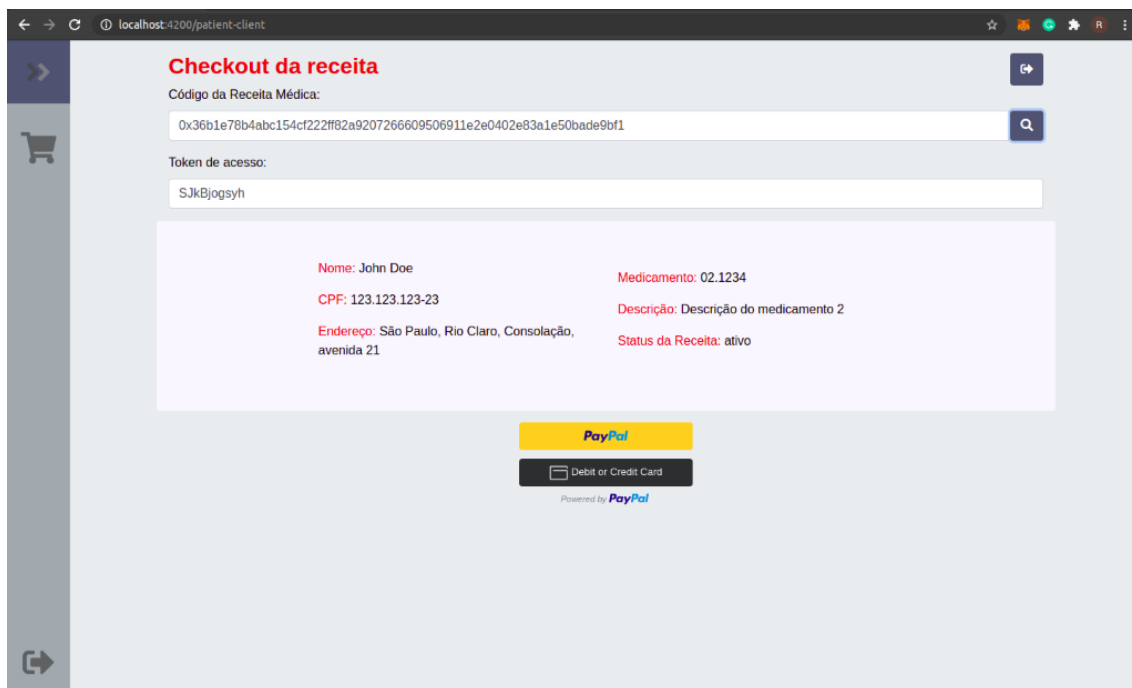


Fonte: Elaborado pelo autor.

Ao clicar no Adquirir receita, o paciente será redirecionado para a tela demonstrada na figura 41, onde pede que seja inserido o código e token da receita médica.

Para dar continuidade na compra do medicamento, o paciente deve clicar no botão do método de compra, neste caso foi utilizado o PayPal, mas pode ser facilmente trocado por qualquer outro sistema de pagamentos.

Figura 41 – Tela de compra da receita

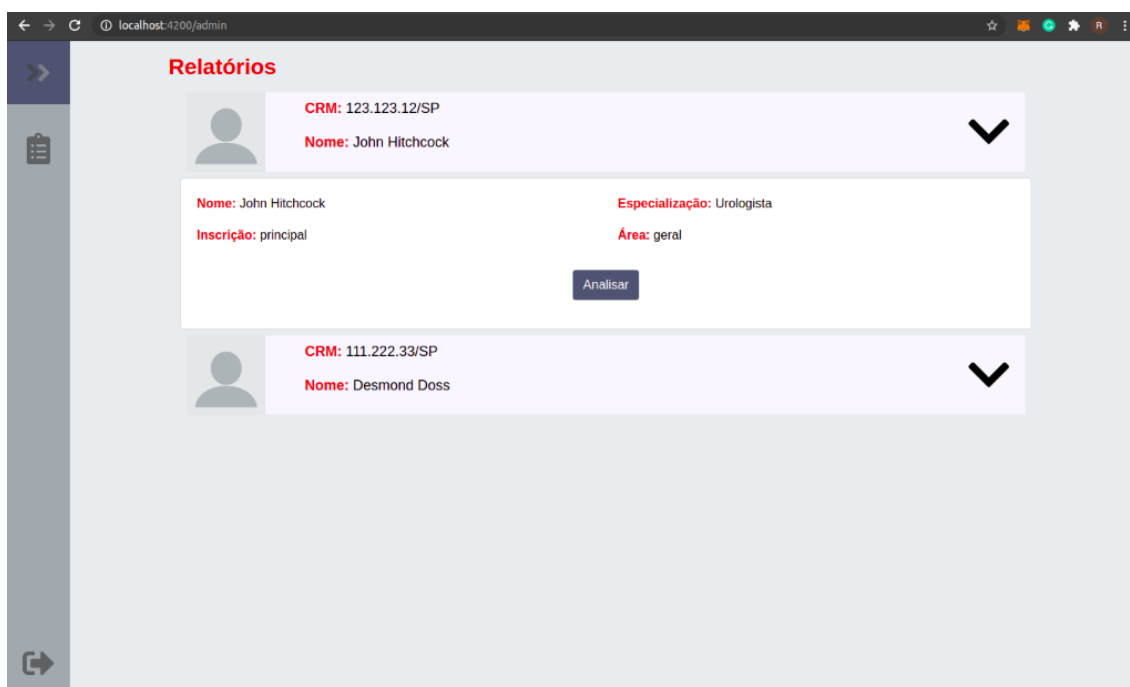


Fonte: Elaborado pelo autor.

4.3.3.5. Desenvolvimento da tela do Auditor

Esta tela tem como objetivo principal ajudar o auditor a ver o fluxo da emissão de receituários médicos de maneira mais concreta, o usuário consegue visualizar uma listagem de médicos, onde será mostrado seu CRM e nome, ao clicar na seta ao lado direito, poderá ser visto mais detalhes deste profissional e também com a ação do botão Analisar será possível ver todas as receitas que foram emitidas pelo mesmo, como demonstrado na figura 42.

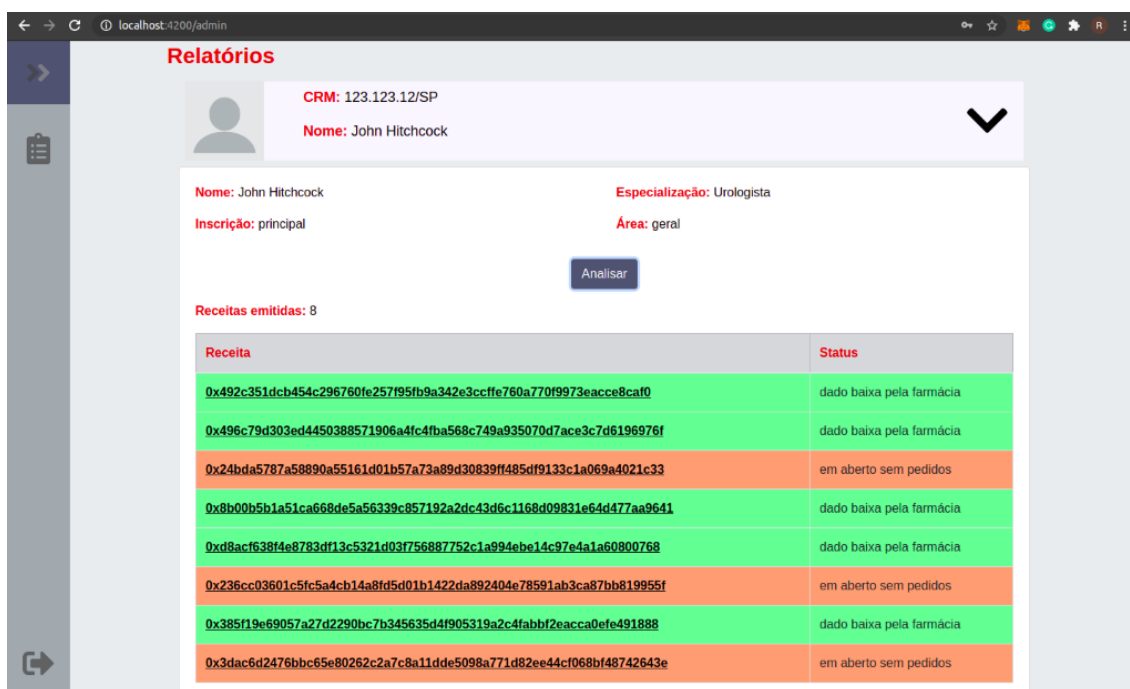
Figura 42 – Tela de listagem de médicos



Fonte: Elaborado pelo autor.

Ao clicar no botão Analisar, o auditor conseguirá ver todas as receitas que foram emitidas, das mais antigas para as mais novas, sendo exibidas dentro de uma tabela de duas colunas, uma contendo o código da receita no blockchain e outra o seu status, que consiste de três valores, em aberto sem pedidos, em aberto com pedidos e dado baixa pela farmácia, sendo mostradas com as suas respectivas cores, vermelho, amarelo e verde, indicando o estado da receita de uma maneira visual, como mostrado na figura 43.

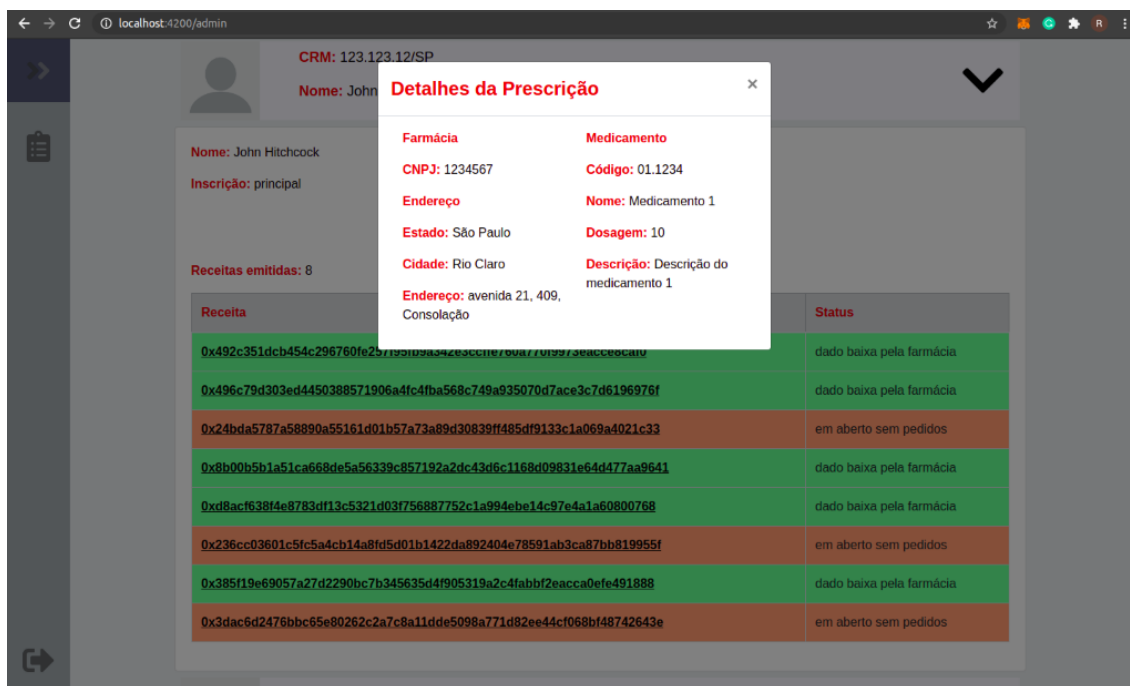
Figura 43 – Tela de listagem de receitas por médico



Fonte: Elaborado pelo autor.

Ao selecionar um dos códigos exibidos, será possível ver os detalhes das receitas, como demonstrado na figura 44, que já conterem pedidos, sendo mostrado dentro de um modal de exibição, com os dados da farmácia e do medicamento.

Figura 44 – Tela de detalhes da receita médica



Fonte: Elaborado pelo autor.

4.4. Desenvolvimento do sistema auxiliar

Este sistema tem como objetivo auxiliar todos os processos da aplicação, para que facilite a integração de outros sistemas e também otimize o custo das transações do Smart Contract, pois quanto mais dados de tamanhos indefinidos forem inseridos no blockchain, maior será o custo de gas para a transação.

O sistema será desenvolvido encima de uma arquitetura de domínios, em que consiste de um domínio representando o modelo da entidade do banco de dados, uma camada abstraída de repositório, usada para executar operações diretamente no banco de dados e uma camada de controladores, que enviamos requisições HTTP através da prática do REST.

4.4.1. Necessidade do sistema auxiliar no projeto

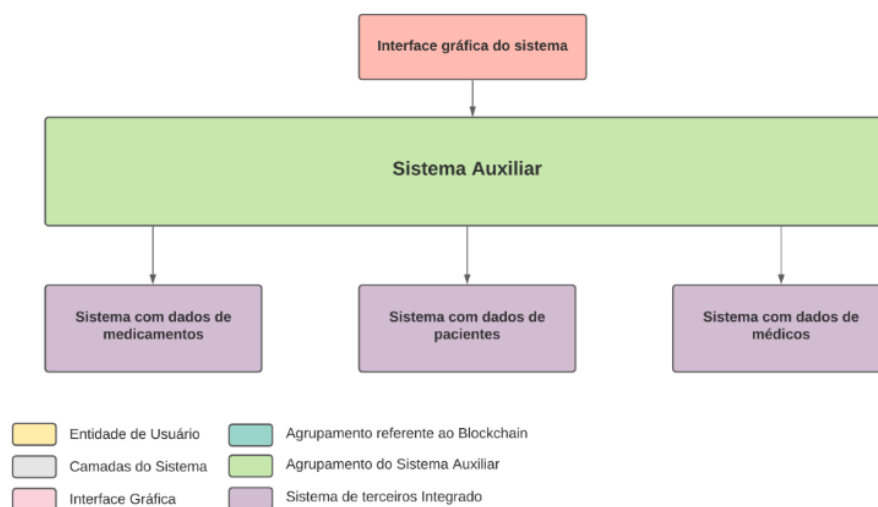
Este projeto teve a necessidade de um sistema auxiliar para executar algumas tarefas que são possíveis apenas por meio de um servidor, como armazenar todos os dados do pedido no banco de dados pode ser muito custoso, porque são muitos dados com tamanhos variáveis, o blockchain segue como base o valor da criptomoeda ETH e cada transação de smart contracts dentro da rede pode gerar custo de utilização de recursos dos computadores que validam os dados do smart contract.

Para a otimização de consumo dos recursos da rede Ethereum, foi criado este sistema para minimizar a quantidade de dados gravados nos blocos do blockchain, para minimizar este consumo, foi desenvolvido um processo de armazenamento em um banco de dados os detalhes da transação do smart contract, tal como preço, detalhes do medicamento, endereço do paciente entre outros, o smart contract armazena um dado chave para que possa ser utilizado como vínculo deste dado em um banco de dados.

4.4.2. Integração de sistemas distribuídos

Com os diversos sistemas que já existem com dados de pacientes, medicamentos, entre outros, houve a necessidade de criar uma espécie de gateway para facilitar a visualização destes dados, como pode ser visto na figura 45, pois a interface gráfica vai conseguir concentrar os seus acessos ao sistema auxiliar em vez de precisar enviar uma requisição para vários sistemas diferentes.

Figura 45 – Diagrama de gateway do sistema distribuído



Fonte: Elaborado pelo autor.

4.4.3. Criando o projeto com node.js

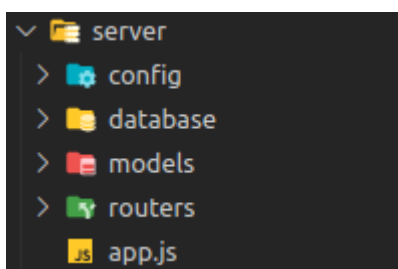
Para o desenvolvimento do sistema auxiliar foi escolhido a linguagem node.js com o framework RESTful express.js, para que seja possível receber e enviar requisições de maneira simples e rápida, seguindo o padrão de arquitetura por domínios.

Node.js é uma linguagem de programação que utiliza o javascript para programação em servidores, ele contém um gerenciador de dependências já integrado chamado NPM, o qual permite anexar todas as dependências necessárias para o funcionamento da aplicação, com ele é possível aumentar o reuso de código, evitando duplicidade nas funções do sistema.

O express.js é um framework já conhecida e bastante utilizada no mercado, ele fornece uma camada fina de recursos fundamentais para aplicativos da web, sem obscurecer os recursos do Node.js.

O projeto consiste de uma estrutura de pastas específicas do modelo orientado a domínios, como pode ser visto na figura 46, temos a pasta config responsável pela conexão com o banco de dados específico, database que contém os dados de configuração da ORM (Object Data Manager – Gerenciador de Objetos de Dados) utilizada para construir as representações das entidades de domínio do projeto, o qual já cria as entidades no banco de dados seguindo a programação do Node.js., o diretório models contém todos os domínios no qual o sistema irá trabalhar, a pasta routers são os controladores que executam as ações do sistema levando em consideração os métodos HTTP, GET, POST, PUT, PATCH e DELETE, além destes diretórios, há um arquivo chamado app.js, ele é responsável pela configuração do express.js e o ponto inicial da aplicação.

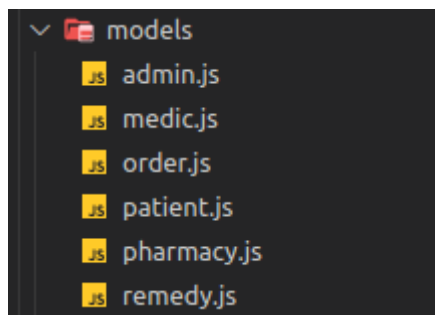
Figura 46 – Estrutura de pastas do sistema auxiliar



Fonte: Elaborado pelo autor.

O ponto inicial do desenvolvimento do sistema auxiliar foi a criação dos models, estes são as representações dos dados do admin (auditor), médico, paciente, farmácia, medicamento e pedidos, como listados na figura 47.

Figura 47 – Arquivos de criação das entidades de domínio



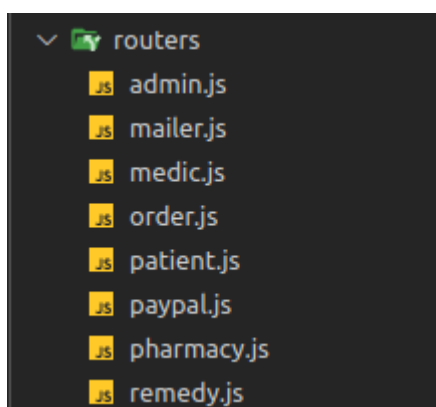
Fonte: Elaborado pelo autor.

Cada domínio tem o seu respectivo controlador, definido dentro do diretório routers, como pode ser visto na figura 48, cada controlador tem seu CRUD (Create, Read, Update e Delete) padrão, para que os dados sejam controlados dentro do banco de dados.

Além das operações de regras de negócios, o sistema também tem a funcionalidade de envio de e-mails, para que durante a emissão do smart contract do receituário médico, também seja enviado o código e token de acesso para que o paciente consiga adquirir o seu medicamento por meio de uma compra online.

Outro controlador que se pode ver é o PayPal, este é responsável por gerenciar a compra do medicamento no lado do servidor, para garantir as configurações da transação, tal como moeda, conta, entre outros detalhes.

Figura 48 – Arquivos de controladores do sistema



Fonte: Elaborado pelo autor.

4.4.4. Utilizando Banco de dados Não Relacional

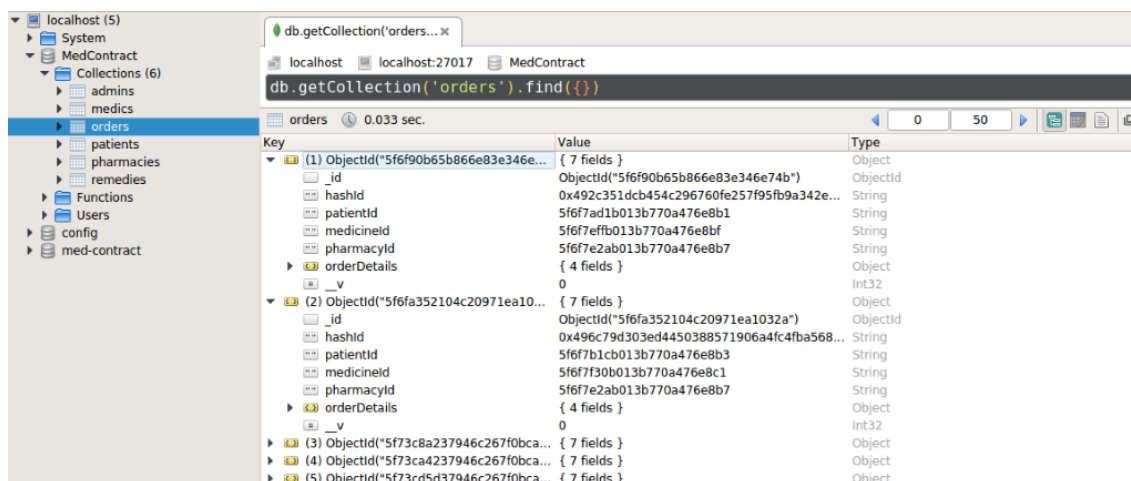
O Banco de dados não relacional foi escolhido para fazer parte do sistema auxiliar, pois não há a necessidade de relacionamento entre as telas, mesmo porque o sistema será um gateway de dados, consumindo serviços já existentes, logo este banco de dados irá trabalhar como um registro de detalhes do smart contract, gerando pedidos para as farmácias e criando dados dos em forma de cache para que tenha um registro de dados importantes para o funcionamento da aplicação, podendo ter também uma base de logs, o qual é utilizado para gravar passos do usuário durante a utilização do sistema.

O MongoDB foi a melhor escolha para o projeto, pois ele contém um ODM (Object Data Manager – Gerenciador de Objetos de Dados) já pronto para utilização no express.js, o qual se chama Mongoose.

O Mongoose é responsável por gerenciar todos os documentos inseridos na base de dados do sistema, ele tem a camada de repositório do sistema já abstraído para o uso do Node.js.

A figura 49 demonstra a listagem do MongoDB destes dados criados pelo sistema auxiliar, utilizando o mongoose como ODM.

Figura 49 – Listagem dos dados cadastrados no MongoDB



Fonte: Elaborado pelo autor.

4.5. Sistema de pagamentos

O projeto pode ter vários sistemas de pagamentos, mas para a prova de conceito foi utilizado o PayPal, que contém uma API de fácil acesso e gerenciamento, sendo executado tanto na interface gráfica quanto no sistema auxiliar, para garantir a execução dos pagamentos em um ambiente controlado.

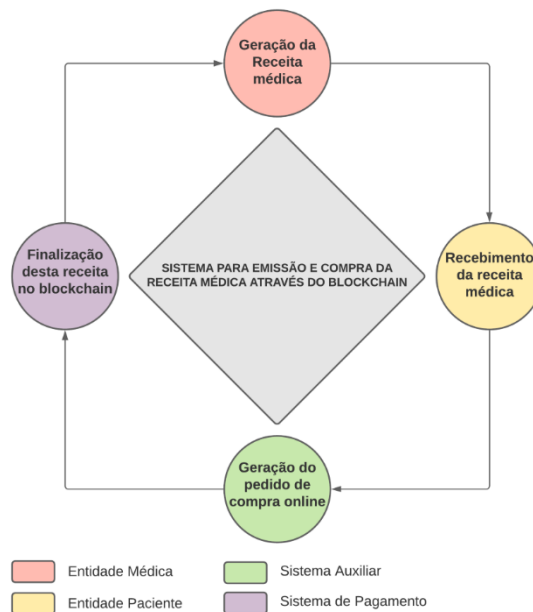
O PayPal oferece um ambiente de fácil acesso e controle para sistemas que utilizam pagamentos online, o sistema permite que o pagamento seja feito através do cartão de crédito ou saldo PayPal que pode ser inserido no sistema deles.

4.5.1. Necessidade do Sistema de pagamento

Para a comprovação de que o cliente consegue retirar o medicamento do conforto de sua residência, foi proposto o recurso de executar um pagamento dentro do sistema, com ele podemos garantir que o blockchain possa permitir apenas um único uso para cada receita emitida, alterando o seu estado.

Logo podemos analisar a necessidade deste sistema de pagamento dividindo o projeto em dois, a primeira parte é a médico paciente, no qual há uma consulta médica que gera um bloco de dados dentro da rede Ethereum, com isso já podemos comprovar a eficácia do blockchain para rastrear estas receitas, já a segunda parte é o ponto chave para que o sistema consiga contribuir para a sociedade, garantindo o fácil acesso a estes medicamentos, sem precisar sair de casa, com o sistema de pagamento, listando todas as farmácias e permitindo que todo o ciclo de emissão da receita médica seja fechada, como visto na figura 50.

Figura 50 – Ciclo do sistema de emissão e compra



Fonte: Elaborado pelo autor.

4.5.2. Flexibilidade do Sistema de Pagamento

Pode-se utilizar diversos outros meios de pagamentos para compras online, o PayPal foi utilizado para simplificar o processo de pagamento e demonstrar o uso e as vantagens deste meio para a farmácia receber um pedido e executar a entrega do mesmo.

O sistema ainda pode ter o benefício de rastrear todas as transações, logo também pode-se ter o benefício total do blockchain mesmo executando o processo de venda no estabelecimento, basta uma integração com o sistema de pagamento da farmácia, assim qualquer meio de pagamento pode ser utilizado para a liberação destas receitas, evitando que possam ser utilizadas novamente.

CONCLUSÃO

Conforme discussões anteriores, onde a proposta era garantir a segurança e rastreio das receitas médicas por meio blockchain, pois com o Smart Contract é possível garantir algumas regras dentro do blockchain, o qual tem a garantia de que se algo for gravado em um bloco, ele não será excluído, dessa maneira pode-se reduzir o tempo de apuração de algum problema, facilitar auditorias e ter uma grande visibilidade de quais receituários os médicos estão emitindo, garantindo uma grande ferramenta para evitar possíveis fraudes.

Além da segurança, também foi garantido o sucesso na compra destes medicamentos online, aumentando a acessibilidade destes medicamentos, pois o paciente consegue comprar remédios controlados sem a necessidade de estar presente no estabelecimento, portando um documento protocolado em que dá o direito de retirar um certo medicamento.

Em trabalhos futuros, há a possibilidade de implementar o gateway do sistema auxiliar por completo, garantindo que funcione sem precisar aposentar qualquer outro sistema de cadastro de pacientes, medicamentos e médicos já existentes, assim tendo a possibilidade de trazer preços de medicamentos mais precisamente de acordo com a loja da escolha do paciente, há também a possibilidade de implementação de uma listagem de dados mais robusto na interface gráfica, trazendo dados paginados, com um sistema de filtro e buscas mais avançado.

Por fim este projeto demonstrou o poder que uma ferramenta com base em blockchain pode trazer para o setor da saúde, principalmente na batalha contra fraudes e corrupção, facilitando auditorias e garantindo que nenhum dado registrado tenha sido alterado, assim aumentando a possibilidade de redução da burocracia em torno do comércio de fármacos controlados.

REFERÊNCIAS BIBLIOGRÁFICAS

ANTONPOULOS, A. M.; Wood, G. **Mastering Ethereum: Building Smart Contracts and Dapps** 1. ed. "O'Reilly Media, Inc.", 13 nov. de 2018.

BITCOINTRADE: smart contracts: entenda o que são e como funcionam, c2020. Smart contract. Disponível em:< <https://blog.bitcointrade.com.br/smart-contract/>>. Acesso em: 21 mai. 2020.

ETHEREUM: Ethereum.org, c2019. Descrição da plataforma. Disponível em:< <https://ethereum.org/pt-br/what-is-ethereum/>>. Acesso em: 20 set. 2019.

FOXBIT: foxbit.com.br, c2019. O que é blockchain. Disponível em:< <https://foxbit.com.br/o-que-e-blockchain/>>. Acesso em: 13 set. 2019.

GRIGGS, K. N.; Ossipova, O.; Kohlios, C. P.; Baccarini, A. N.; Howson, E. A.; Hayajneh T. **Healthcare Blockchain System Using Smart Contracts for Secure Automated Remote Patient Monitoring** Disponível em:<<https://link.springer.com/content/pdf/10.1007/s10916-018-0982-x.pdf>>. Acessado em 20 de maio de 2020.

INFOMONEY: infomoney.com.br, c1019. Sobre Ethereum. Disponível em:< <https://www.infomoney.com.br/cotacoes/ethereum-eth/>>. Acesso em: 11 set 2019.

MUKHOPADHYAY, M. **Ethereum Smart Contract Development** 1. ed. Packt Publishing Ltd, 23 de fev. de 2018.

NORMAN, A. T. **Blockchain Technology Explained: The Ultimate Beginner's Guide about Blockchain Wallet, Mining, Bitcoin, Ethereum, Litecoin, Zcash, Monero, Ripple, Dash, IOTA and Smart Contracts** 1. ed. CreateSpace Independent Publishing Platform, 11 de dez. de 2017.

WU. X.; Zou, Z.; Song, D. **Learn Ethereum: Build your own decentralized applications with Ethereum and smart contracts** 1. ed. Packt Publishing Ltd, 20 de set. de 2019.