

DOCSTONE

A Blockchain-based Architecture for a Customizable Document Registration Service

Pamella Soares¹, Raphael Saraiva¹, Iago Fernandes¹,
Jerffeson Souza¹, Ricardo Loiola²

¹ Universidade Estadual do Ceará

² Pontifícia Universidade Católica do Rio Grande do Sul

AGENDA

- 1. INTRODUÇÃO**
- 2. BLOCKCHAIN**
- 3. TRABALHOS RELACIONADOS**
- 4. VISÃO GERAL DA ARQUITETURA**
- 5. ESTUDO EMPÍRICO**
- 6. DISCUSSÕES E LIÇÕES APRENDIDAS**
- 7. AMEAÇAS À VALIDADE**
- 8. CONSIDERAÇÕES FINAIS**

1

Introdução

Um documento pode ser definido como um conjunto de informações capazes de serem utilizadas para consulta, prova e pesquisa , de forma que se possa comprovar fatos e pensamentos do homem em dado tempo.

INTRODUÇÃO

GESTÃO DOCUMENTAL E TECNOLOGIA DA INFORMAÇÃO

A **tecnologia da informação** pode **impactar na gestão documental** quanto aos seguintes pontos (Aziz et al. 2020):

- Gerenciamento de documentos,
- Redução e a otimização do espaço de armazenamento,
- Compartilhamento e a transferência de informações,
- Redução da perda de registros,
- Eliminação do papel e uso de “documentos digitais”.

INTRODUÇÃO

GESTÃO DOCUMENTAL E TECNOLOGIA DA INFORMAÇÃO

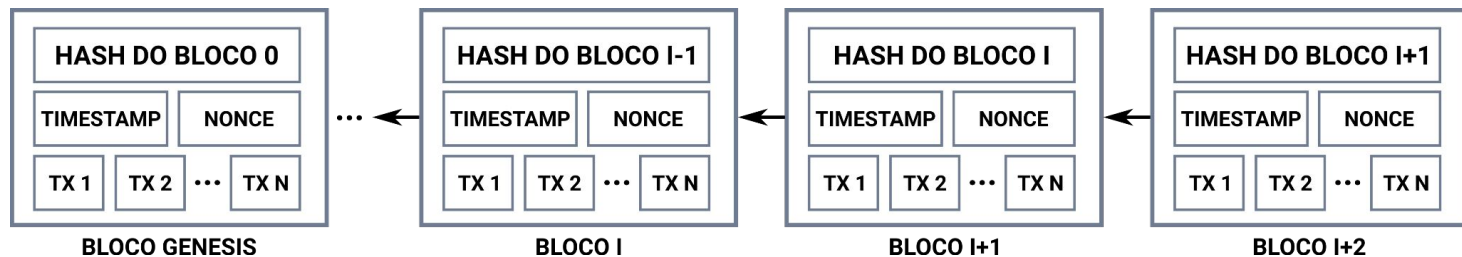
Ao realizar a gestão documental, as organizações **devem garantir**:

- **Privacidade e segurança** às informações para dados sensíveis e confidenciais,
- **Transparência** na disponibilização de informações públicas.

INTRODUÇÃO

BLOCKCHAIN

Blockchain é uma rede peer-to-peer (P2P) formada por um livro-razão distribuído que armazena transações por meio de consenso da rede e que, junto com uso de criptografia, Contratos Inteligentes e outros componentes, torna-se uma rede segura e confiável (Acharya et al. 2019).



INTRODUÇÃO

BARREIRAS NA ADOÇÃO DE BLOCKCHAIN

A blockchain é **essencialmente um sistema distribuído**, causando:

- Dificuldade de implantação;
- Alto custo de operação e manutenção da blockchain.

O **processo de implementação** de blockchain e aplicações descentralizadas (dApps) pode se **tornar complexo e propenso a equívocos** (Jie et al., 2021);

INTRODUÇÃO

BLOCKCHAIN AS A SERVICE

Ajuda os desenvolvedores a focar apenas na codificação das regras de negócios, já que o **BaaS** fornece serviços em nuvem, como implantação da infraestrutura e monitoramento da rede (Onik e Miraz, 2019).

PROBLEMÁTICAS

- Documentações extensas;
- Implantação com **elevados custos** para sistemas simples.

INTRODUÇÃO

ESPECIFICIDADES DE SOLUÇÕES ATUAIS

Existem **propostas encontradas na literatura** e que apresentam três principais **requisitos básicos em comum**: 1) o **registro**, 2) a **busca** e 3) a **validação** de informações documentais

Considerando o contexto no qual as propostas estão inseridas, tem-se encontrado uma certa **especificidade de domínios** e **regras de negócios rígidas**.

- Registro de certificados e diplomas;
- Registro de informações médicas, etc;

OBJETIVO

O presente trabalho propõe uma arquitetura para o **Serviço de Registro de Documentos Customizáveis baseado em Blockchain**, cuja solução permite a **configuração de parâmetros para armazenamento de informações**, desde a criação de modelos de documentos até a **escolha de uma ou mais blockchains públicas e uma blockchain privada** para registro, e validação de acordo com a demanda da aplicação cliente.

CONTRIBUIÇÕES

As principais contribuições deste estudo são:

- 1) a proposta de uma **arquitetura genérica** para atender uma **variedade de áreas de negócio** no registro e validação de informações em **múltiplas blockchains**,
- 2) uma **API developer-friendly** para sistemas novos ou pré-existentes **para melhoria na otimização de tempo, e redução de custos de desenvolvimento e refatorações**,
- 3) uma análise baseada em **decisões de design para aplicações descentralizadas**, e
- 4) uma **avaliação de desempenho** entre diferentes blockchains no contexto de armazenamento de informações.

2

Blockchain

BLOCKCHAIN

CONTRATOS INTELIGENTES

Novas gerações de blockchains permitiram o desenvolvimento de lógicas e modelos de negócio elaborados;

Contratos Inteligentes (CI), introduzidos pela rede Ethereum (Buterin et al., 2013), consistem em **códigos que auto executam a lógica de negócios** no momento em que **determinadas condições programadas são atendidas** (Bashir, 2017).

BLOCKCHAIN

TIPOS DE BLOCKCHAIN

Blockchain Pública

**Ethereum Virtual
Machine**

Proof-of-Work

Proof-of-Stake

Blockchain Privada

Hyperledger Fabric

Peers

Ordering Service

Chaincode

Livro-razão

***MSP (Membership
Services Provider)***

3

Trabalhos Relacionados

TRABALHOS RELACIONADOS

DMS-XT: A Blockchain-based Document Management System for Secure and Intelligent Archival.

Edlira Martiri et al., 2018

Lekana-Blockchain Based Archive Storage for Large-Scale Cloud Systems.

Bandara et al., 2020.

A blockchain-based integrated document management framework for construction applications.

Das et al., 2022

A decentralized document management system using blockchain and secret sharing.

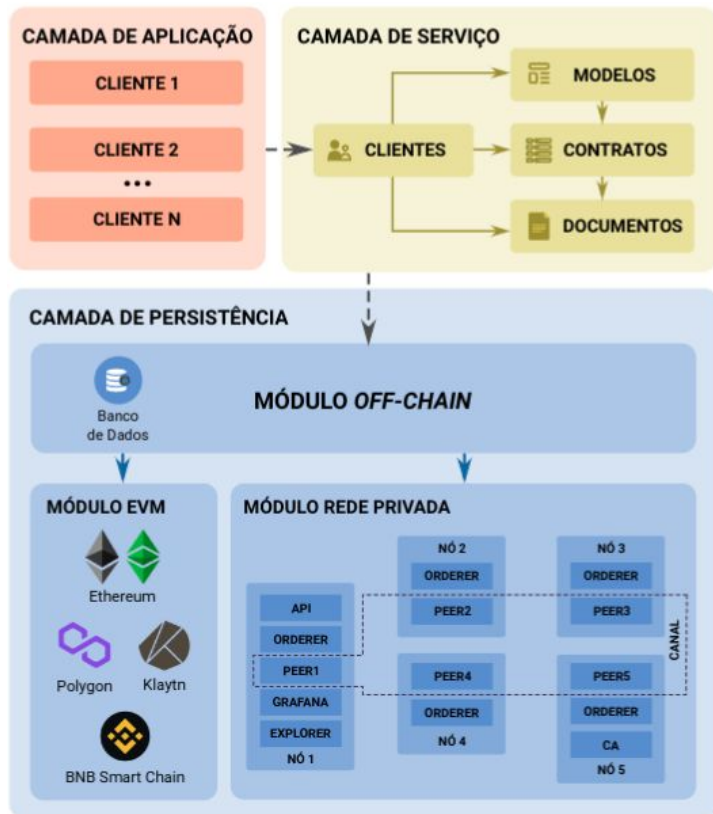
Han et al., 2018

4

Visão Geral da Arquitetura

DOCSTONE

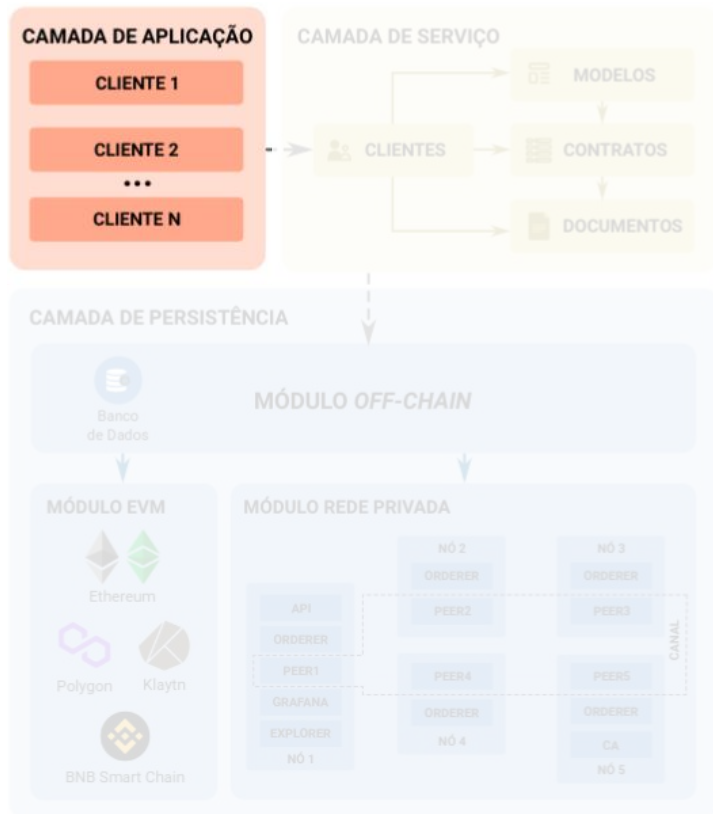
VISÃO GERAL DA ARQUITETURA



Uma **arquitetura para o serviço customizável** de registro e validação de documentos baseado em blockchain. O serviço contém funcionalidades pertinentes ao gerenciamento de documentos, podendo ser **facilmente integrado às plataformas clientes** para a criação de **modelos personalizáveis** de documentos que podem, então, ser armazenados em **múltiplas blockchains**.

DOCSTONE

VISÃO GERAL DA ARQUITETURA



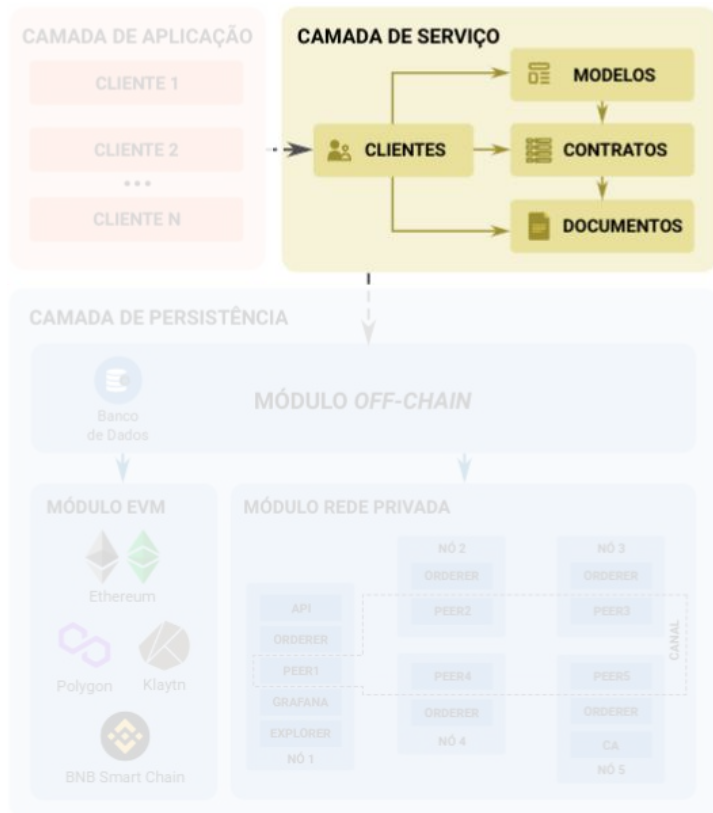
É uma camada composta por todas as aplicações clientes que se conectam e utilizam a solução.

Disponibiliza um subconjunto de rotas para as aplicações clientes, são elas:

- `POST \model` - Criação de modelos;
- `POST \contract` - Implantação de CI em blockchains específicas;
- `POST \document` - Registro de documentos;
- `GET \document` - Busca por documentos;
- `POST \validation` - Validação de documento.

DOCSTONE

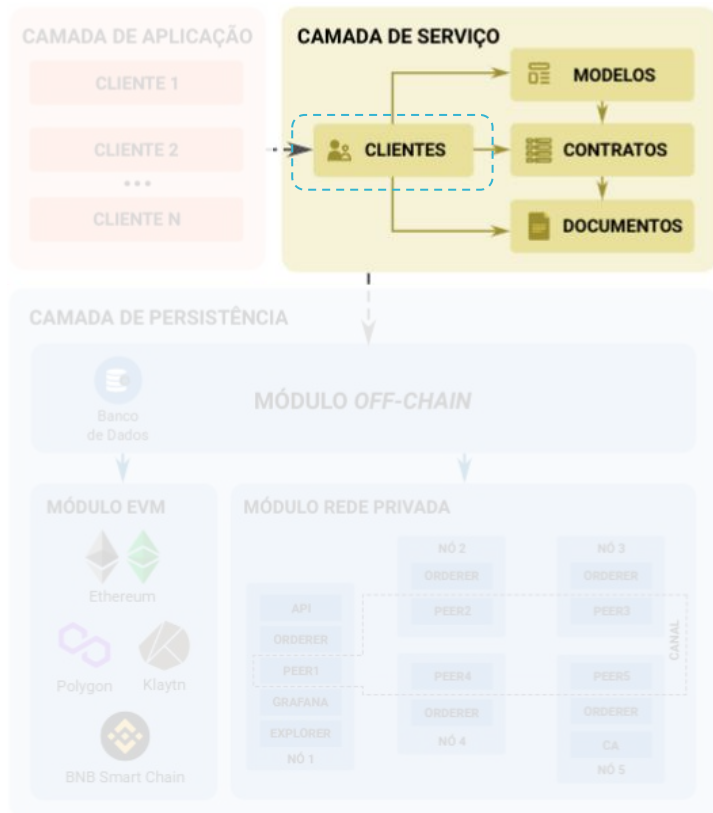
VISÃO GERAL DA ARQUITETURA



Esta camada lida com a **implementação e execução de todas as funcionalidades** referentes ao gerenciamento de documentos na API por meio de **quatro ativos principais**.

DOCSTONE

VISÃO GERAL DA ARQUITETURA

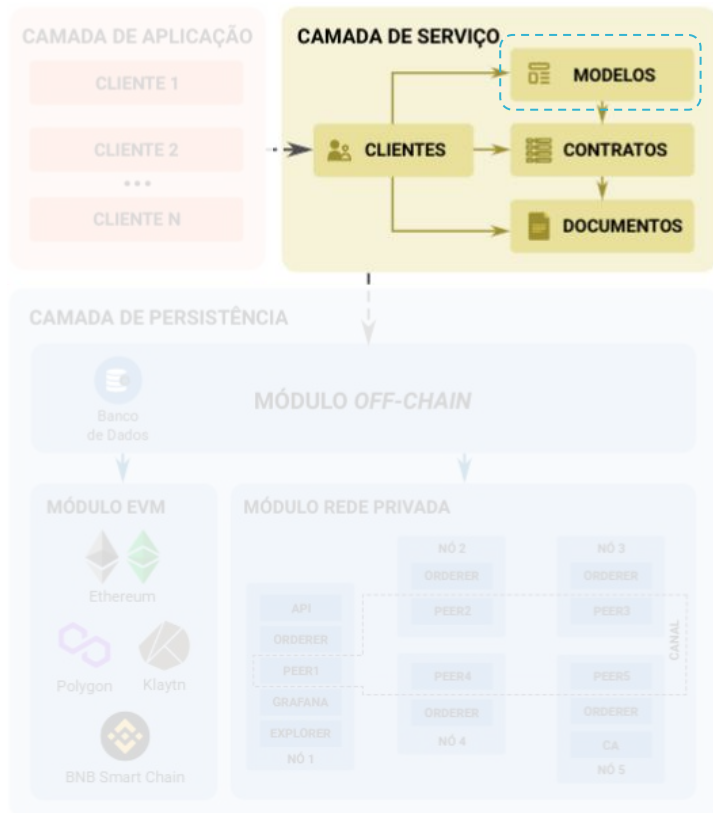


Esta camada lida com a **implementação e execução de todas as funcionalidades** referentes ao gerenciamento de documentos na API por meio de **quatro ativos principais**.

O **Cliente** representa cada aplicação cliente advinda da **Camada de Aplicação**.

DOCSTONE

VISÃO GERAL DA ARQUITETURA

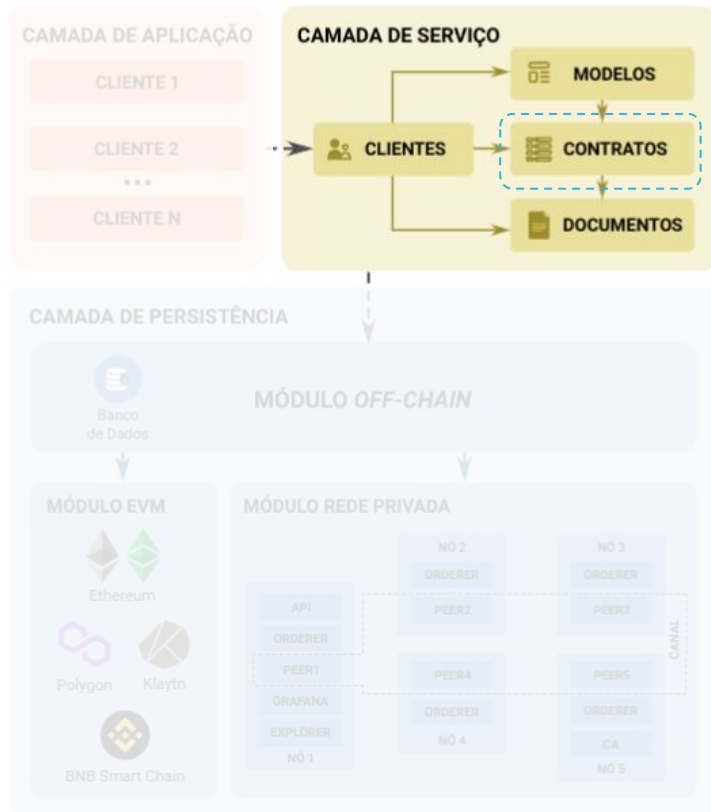


O **Modelo** representa o **formato dos tipos de documentos** que o Cliente registra em blockchain através da API.

Composto por atributos específicos pré-definidos conforme o documento em questão.

DOCSTONE

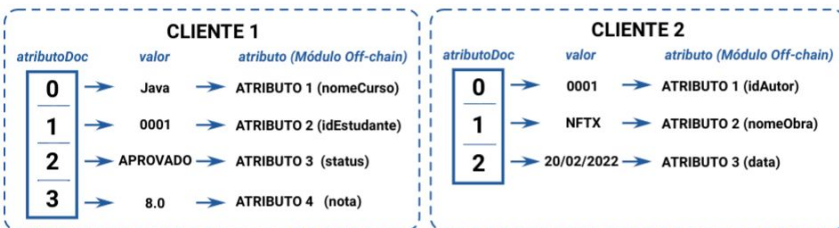
VISÃO GERAL DA ARQUITETURA



O ativo **Contrato Inteligente (CI)** refere-se ao **código implantado** responsável por executar as principais funções de registro e busca das informações do documento em blockchain.

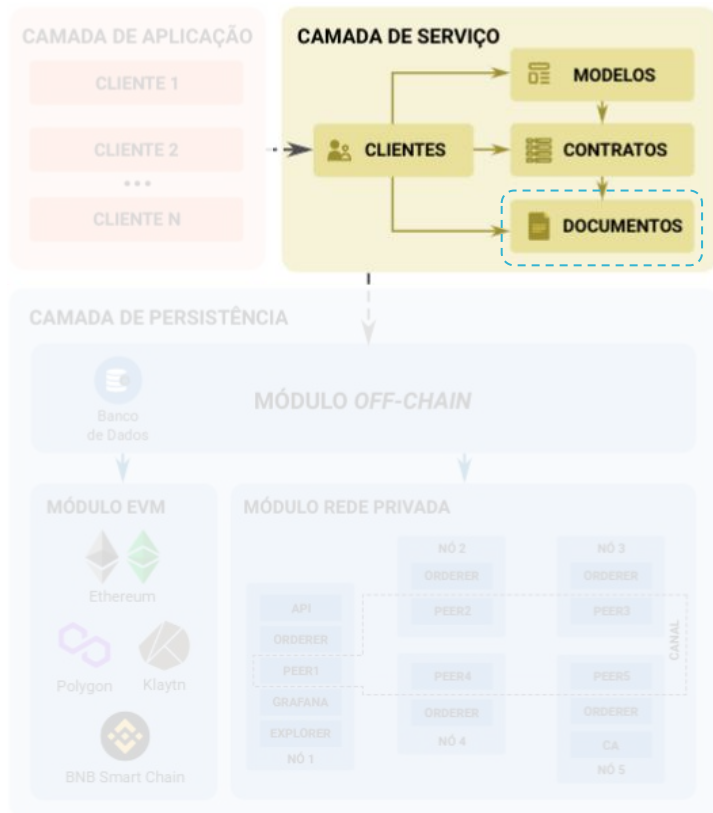
Composto pelas principais funções:

- *insereDocumento();*
- *buscaDocumento();*
- *atualizaDocumento().*



DOCSTONE

VISÃO GERAL DA ARQUITETURA

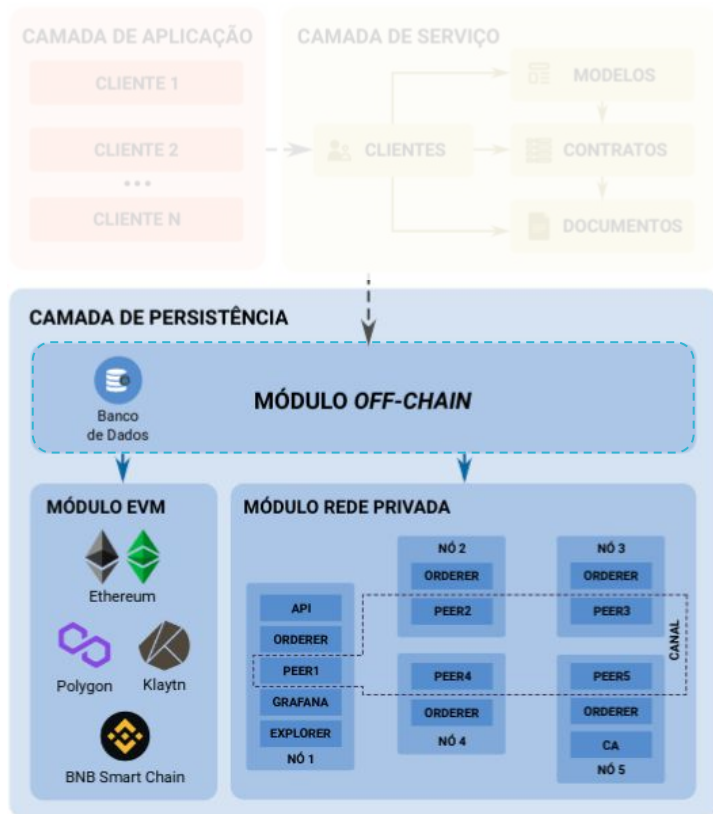


O ativo **Documento** consiste no conteúdo das informações de um documento propriamente dito.

Tais informações serão gerenciadas conforme o Modelo criado previamente.

DOCSTONE

VISÃO GERAL DA ARQUITETURA



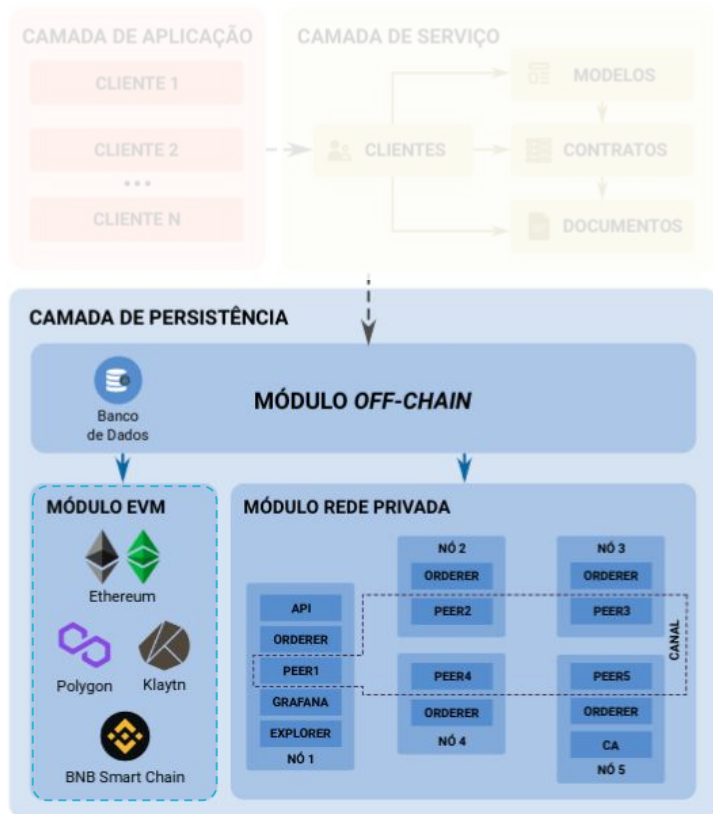
O **Módulo Off-chain** serve para cadastro e gerenciamento dos Clientes da aplicação:

- Armazenamento de credenciais,
- Tokens de acesso à API,
- Dados sensíveis.

O banco de dados também armazena informações privadas e os atributos de modelos criados para realizar o cruzamento de dados entre o banco de dados e a blockchain.

DOCSTONE

VISÃO GERAL DA ARQUITETURA

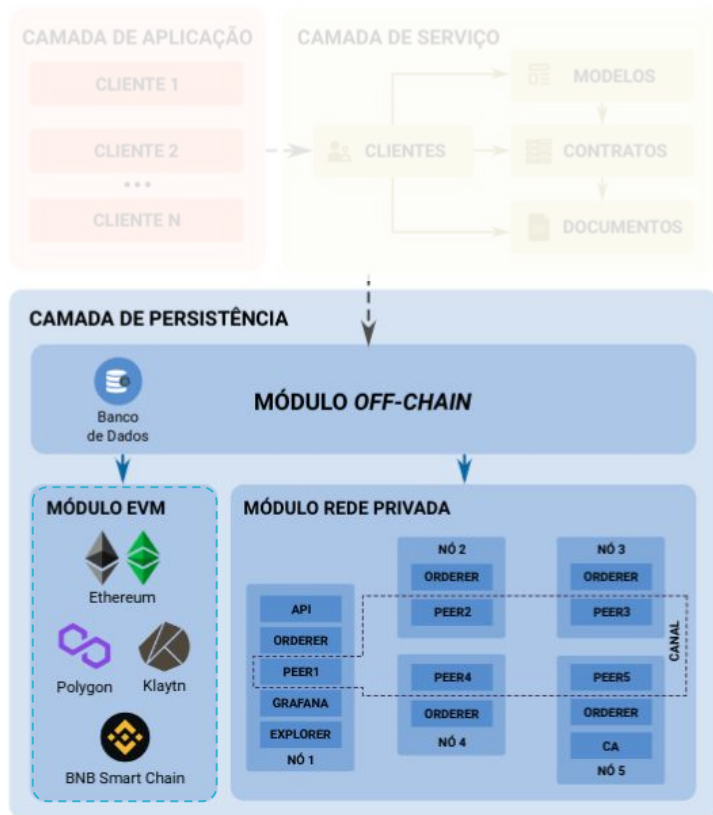


No **Módulo EVM**, as informações não sensíveis armazenadas a partir do Modelo criado são inseridas em blockchain.

A inserção de informações em uma ou mais blockchains para possibilitar o sistema cliente escolher a blockchain que mais se adéque às suas demandas.

DOCSTONE

VISÃO GERAL DA ARQUITETURA

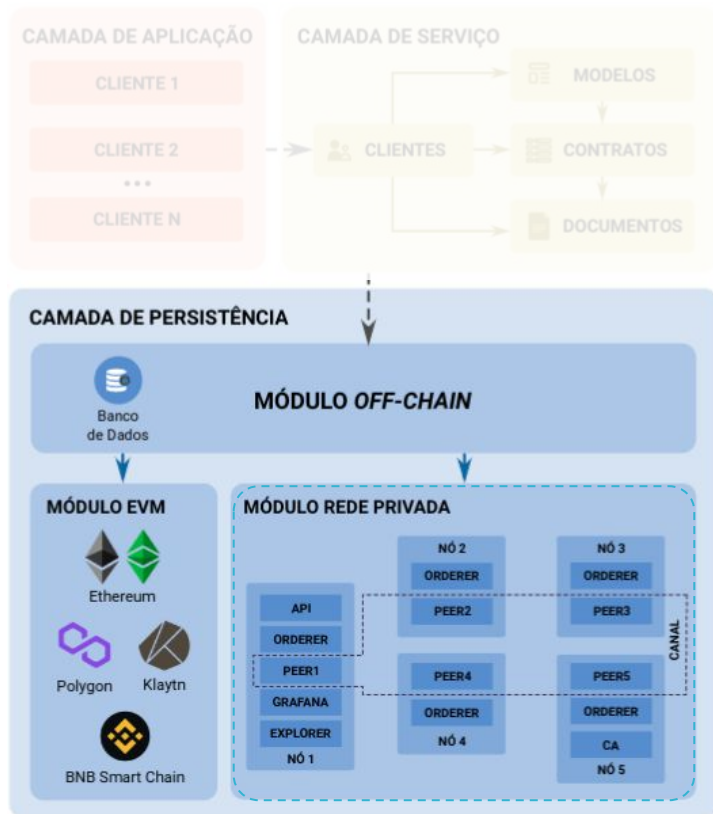


O cliente pode decidir em qual blockchain registrar informações considerando diferentes pontos, como:

- 1- o quanto está disposto a pagar pelas **taxas** da rede;
- 2- a **velocidade** de registro das informações, a qual pode ser influenciada pelo algoritmo de consenso e tecnologias utilizadas por cada blockchain;
- 3- a **reputação** da rede na comunidade e seu engajamento;
- 4- o **nível de descentralização**; dentre outros.

DOCSTONE

VISÃO GERAL DA ARQUITETURA



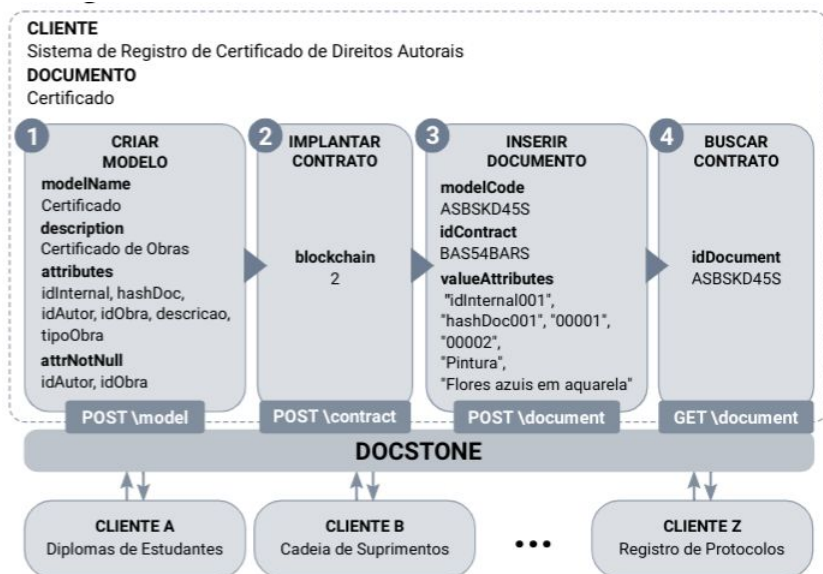
O **Módulo Rede Privada** Dentre as blockchains integradas à API, uma Rede Privada de organização única (*single organization*) baseada no Hyperledger Fabric foi implementada.

A rede é composta por cinco nós, cada um deles contendo, pelo menos, um peer e um orderer.

A rede usa a *Transport Layer Security* (TLS), o que exigiu a configuração de uma CA TLS e seu uso para gerar certificados TLS.

DOCSTONE

DEMONSTRAÇÃO



Passo 1: o cliente cria um modelo de certificados através da rota POST \model.

Passo 2: o Cliente faz a chamada do \POST \contract para implantar o CI em uma blockchain definida, indicando aquela na qual se deseja armazenar as informações dos certificados a serem inseridos.

Passo 3: a API recebe o *idContract* e o *modelCode* para identificar o contrato e o tipo de modelo no qual as informações serão armazenadas ao solicitar o POST \document.

Passo 4: o documento registrado pode ser consultado através da requisição GET \document.

5

Estudo Empírico

ESTUDO EMPÍRICO

CONFIGURAÇÕES DO EXPERIMENTO

Realizou-se um **experimento computacional** através de um conjunto de testes, uma métrica de desempenho e as configurações do processo de coleta.

A fim de **analisar o atributo de desempenho da solução** nas redes de testes das blockchains:

- Rinkeby (Ethereum),
- BNB Testnet (Binance),
- Baobab (Klaytn),
- HLF (Hyperledger).

ESTUDO EMPÍRICO

CONFIGURAÇÕES DO EXPERIMENTO

Neste experimento foi adotado o **JMeter 5.020** a fim de simular cargas e enviá-las para o DocStone.

Os testes de desempenho realizados foram aplicados na API, especificamente, sobre dois principais tipos de requisições:

- 1) POST \document
- 2) GET \document

Para representar um modelo de documento, os seguintes atributos foram definidos:

```
{  
  "description": "Modelo padrao para testes.",  
  "attributes": "idInternal;hashDoc;campo1",  
  "modelName": "padrao01",  
  "attrNotNull": "campo1"  
}
```

ESTUDO EMPÍRICO

CONFIGURAÇÕES DO EXPERIMENTO

Configurações do JMeter:

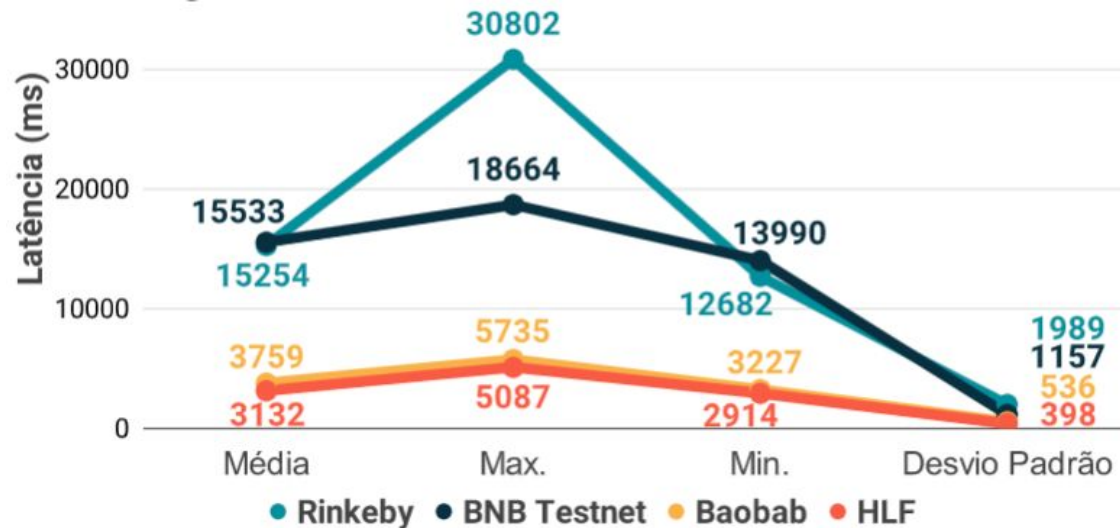
- 100 requisições por parte desse cliente para cada *blockchain*,
- As solicitações são realizadas a cada 5 segundos de **forma serializada**.

Configurações da máquina:

- Processador Intel Core i5-10400F,
- 16GB de memória RAM,
- Conexão de internet banda larga de 80 Mbps.

ESTUDO EMPÍRICO

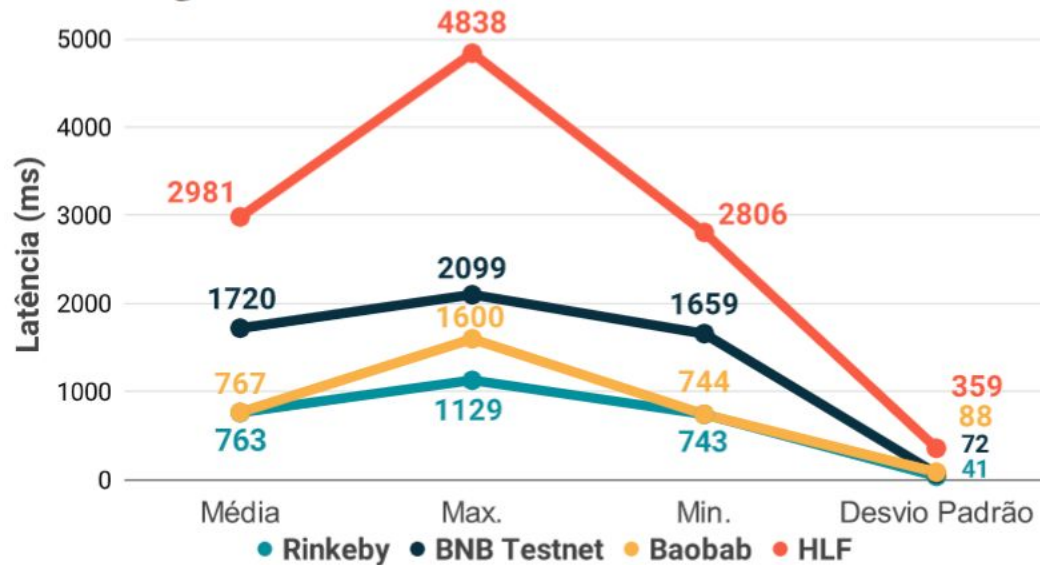
AVALIAÇÃO DE DESEMPENHO



Latência da requisição POST \document nas blockchains integradas ao DocStone

ESTUDO EMPÍRICO

AVALIAÇÃO DE DESEMPENHO



Latência da requisição GET \document nas blockchains integradas ao DocStone

ESTUDO EMPÍRICO

AVALIAÇÃO DE DESEMPENHO

Testes estatísticos:

- Wilcoxon (WC), que identifica a ocorrência de diferença estatística entre as amostras considerando um **nível de confiança de 95%** utilizando-se da correção de Bonferroni; e
- Vargha-Delaney, o qual retorna o número relativo de vezes que um tipo de caso produziu valores superiores ao outro.

Redes	POST \document						GET \document					
	BNB testnet		Rinkeby		HLF		BNB testnet		Rinkeby		HLF	
	WC	\hat{A}_{12}	WC	\hat{A}_{12}	WC	\hat{A}_{12}	WC	\hat{A}_{12}	WC	\hat{A}_{12}	WC	\hat{A}_{12}
Rinkeby	0.67	0.43	-	-	-	-	2e-16	0	-	-	-	-
HLF	2e-16	0	2e-16	0	-	-	2e-16	0	9e-07	0.27	-	-
Baobab	2e-16	1	2e-16	0	2e-16	0.10	2e-16	0	2e-16	1	2e-16	1

Valores estatísticos de Wilcoxon e Vargha-Delaney para todas as blockchains

6

Discussões e Lições Aprendidas

DISCUSSÕES E LIÇÕES APRENDIDAS

DESEMPENHO/ESCALABILIDADE

- Blockchains públicas **possuem escalabilidade mais crítica** que as privadas pois o **desempenho da rede** pode variar com uma série de fatores:
 - Algoritmos de consenso;
 - Taxa de processamento;
 - Latência da transmissão dos dados.
- Operações que demandam a **mudança de estado** do livro-razão tendem a **demorar mais**;

NÍVEL DE DESCENTRALIZAÇÃO

- Docstone apresenta-se com um nível de **descentralização parcial** (semi-descentralizada ou híbrida);

DISCUSSÕES E LIÇÕES APRENDIDAS

CONEXÃO COM A BLOCKCHAIN

- Realizada com **endpoints** (infura) de blockchain, pois não se fez necessária, inicialmente, a criação de nós próprios para as **blockchains públicas** integradas à API;

LÓGICA DE APLICAÇÃO E ARMAZENAMENTO

- Uso de estratégias **on-chain** através dos **contratos inteligentes** e técnicas **off-chains** foram utilizadas para **controle e autenticação de usuários** à API;
- Quanto ao **armazenamento**, buscou-se **preservar as informações sensíveis dos clientes**.

DISCUSSÕES E LIÇÕES APRENDIDAS

PROVISIONAMENTO DE IDENTIDADE, GERENCIAMENTO DE CHAVES E TRATAMENTO DE TRANSAÇÕES.

- O Docstone **gerência os pares de chaves** e encaminhar informações (transações) para a blockchain;
- O Docstone ainda não implementa o mapeamento dos usuários através de **endereços de suas próprias wallets**;
- Desafios quanto aos erros de nonce, **congestionamento de rede**, perda de peers, perda de transações devido a um aumento repentino de preço, etc.
 - Uso de **Gerenciadores de Transações para futuras versões** (como o **EthVigil**, por exemplo).

7

Ameaças à Validade

AMEAÇAS À VALIDADE

AMEAÇAS INTERNAS

- As redes de testes de blockchain utilizadas podem apresentar **alterações de latências** nos diferentes instantes de tempos avaliados;
- Diferentes comportamentos entre as **redes de testes** e as *mainnets*.

AMEAÇAS EXTERNAS

- Contrato inteligente **limitado** e **restrito**;
- A **rede privada não foi totalmente explorada** quanto aos seus recursos de controle de permissão entre **multi-organizações**.

AMEAÇAS À VALIDADE

AMEAÇA DE CONSTRUÇÃO

- **Outras métricas referentes às características internas das blockchains** para complementar a análise de desempenho poderiam ter sido adicionadas.

AMEAÇA DE CONCLUSÃO

- O **número de requisições** da avaliação pode **não retratar ambientes reais**, principalmente ao se tratar de uma API que deve suportar mais de um tipo de aplicação cliente.

8

Considerações Finais

CONSIDERAÇÕES FINAIS

- O presente trabalho apresentou uma **arquitetura de registro e validação de documentos customizáveis e modulares para diferentes contextos e blockchains**, o qual permite a personalização de parâmetros desde a criação de modelos de documentos até o uso de uma blockchain específica.
- Desenvolveu-se uma **arquitetura genérica** para atender diferentes áreas de mercado para o registro e validação de informações em múltiplas blockchains no formato de uma **API developer-friendly**.
- Os resultados mostraram que a **rede HLF** possui o melhor tempo de resposta na escrita entre as blockchains avaliadas, porém esse resultado diferiu na leitura, sendo a **rede Baobab** tendo o melhor desempenho 83% das vezes que a segunda melhor (Rinkeby).

CONSIDERAÇÕES FINAIS

Como trabalhos futuros, pretende-se:

- 1) Implementar CIs com **novas funcionalidades** para permitir que os clientes selecionem regras e lógicas mais específicas para além de **registro, busca e validação**;
- 2) Realizar **novos experimentos** para avaliar a solução em um **cenário de requisições paralelas**, advindas de organizações diferentes, a fim de testar a robustez da API quando exposta a requisições vindas de fontes diferentes;
- 3) Aplicar métodos específicos de **avaliação de arquiteturas de software**, como o **Architecture Trade-off Analysis Method (ATAM)** para determinar a extensão e as potencialidades do Docstone no atendimento de seus atributos de qualidade esperados em uma perspectiva arquitetural e de negócio.

Obrigado!

contato: raphael.saraiva@aluno.uece.br



UNIVERSIDADE
ESTADUAL DO CEARÁ



Pontifícia Universidade Católica
do Rio Grande do Sul



Dúvidas?

contato: raphael.saraiva@aluno.uece.br



UNIVERSIDADE
ESTADUAL DO CEARÁ



Pontifícia Universidade Católica
do Rio Grande do Sul



ESTUDO EMPÍRICO

Realizou-se um **experimento computacional** através de um conjunto de testes, uma métrica de desempenho e as configurações do processo de coleta.

Para fins de experimentação, este trabalho desenvolveu uma API no formato de uma **Prova de Conceito**.

FERRAMENTAS	DESCRIÇÃO
Solidity e Go	Linguagens utilizadas para implementação de contratos inteligentes baseados em EVM e HLF, respectivamente.
Web3.js e Caver.js	Coleção de bibliotecas que permitem a interação com um nó Ethereum local ou remoto. A Caver.js é usada na blockchain Klaytn.
Rinkeby Baobab BNB Testnet	Redes de teste das blockchains utilizadas. Normalmente é usada por desenvolvedores para executar testes sem utilizar a cripto real.
Node.js e Express.js	Node.js pode ser definido como um ambiente de execução Javascript server-side. Já Express.js é uma estrutura da web baseada no módulo Node.js HTTP principal e em componentes chamados middlewares.
Docker	Conjunto de produtos para virtualização para entrega de software em pacotes chamados contêineres, podendo ser orquestrados pela ferramenta Docker Swarm.

DISCUSSÕES E LIÇÕES APRENDIDAS

NÍVEL DE DESCENTRALIZAÇÃO

- Docstone apresenta-se com um nível de **descentralização parcial** (semi-descentralizada ou híbrida);
- O uso de **componentes centralizados** pode ser utilizado de forma a **complementar**, por exemplo, para o **armazenamento de informações sensíveis**.

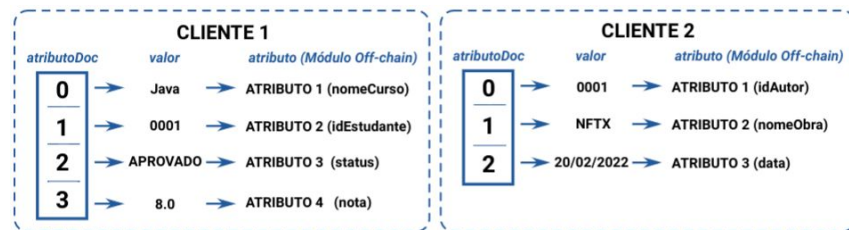
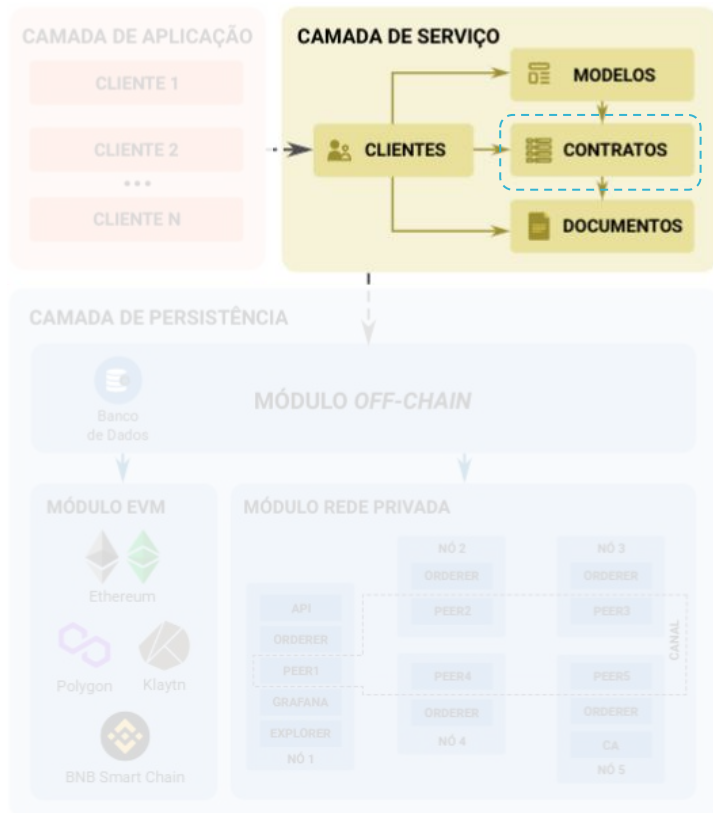
DISCUSSÕES E LIÇÕES APRENDIDAS

LÓGICA DE APLICAÇÃO E ARMAZENAMENTO

- Uso de estratégias **on-chain** através dos **contratos inteligentes**.
- As técnicas **off-chains** foram utilizadas para **controle e autenticação de usuários** à API.
- Quanto ao **armazenamento**, buscou-se **preservar as informações sensíveis dos clientes**, assim como a tentativa de terceirizar o local de armazenamento de dados brutos, como grande volume de dados e mídias dos documentos.

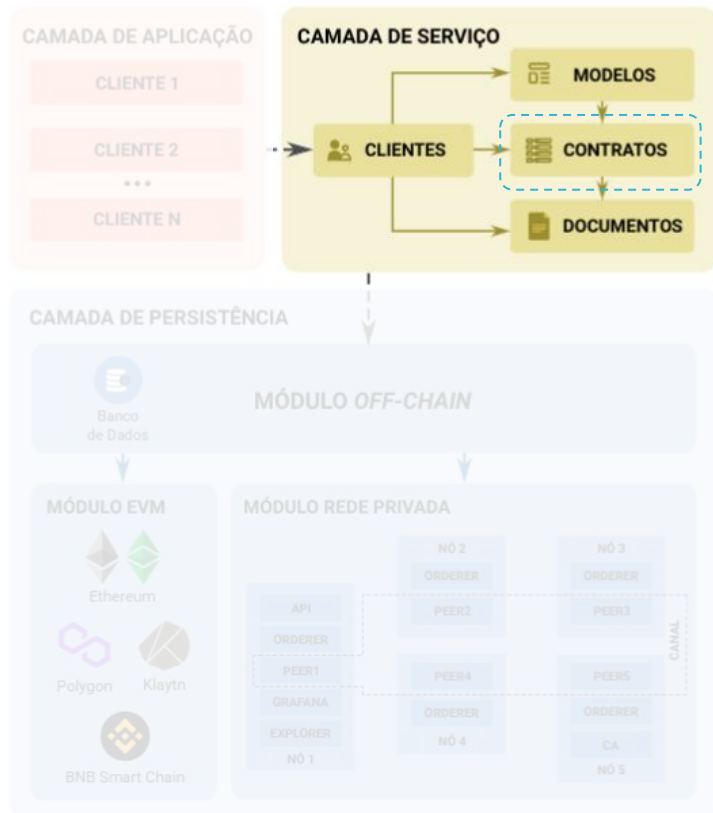
DOCSTONE

VISÃO GERAL DA ARQUITETURA



DOCSTONE

VISÃO GERAL DA ARQUITETURA



```

1 pragma solidity ^0.4.17;
2 pragma experimental ABIEncoderV2;
3 contract DocStone {
4     address public proprietario;
5     struct Documento {
6         string[] atributosDoc;
7     }
8     mapping (string => Documento) private Documentos;
9     function DocStone () public {
10         proprietario = msg.sender;
11     }
12
13     function insereDocumento(
14         string memory _idDocumento,
15         string[] memory _atributosDoc
16     ) public returns (string memory) {
17         if(msg.sender != proprietario){
18             return "ERRO: apenas o proprietario do CI
19                 pode inserir um Documento";
20         }
21         Documentos[_idDocumento] = Documento(
22             _atributosDoc);
23         return "SUCESSO: Documento registrado!";
24     }
25
26     function buscaDocumento(
27         string memory _idDocumento
28     ) public view returns ( string[] memory) {
29         return Documentos[_idDocumento].atributosDoc;
30     }
31
32     function atualizaDocumento(
33         string memory _idDocumento,
34         uint _posicao,
35         string memory _novoValor
36     ) public returns (string memory) {
37         if(msg.sender != proprietario){
38             return "ERRO: apenas o proprietario do
39                 contrato pode atualizar o Documento";
40         }
41         Documento doc = Documentos[_idDocumento];
42         doc.atributosDoc[_posicao] = _novoValor;
43         return "SUCESSO: Documento alterado!";
44     }
45 }

```

BLOCKCHAIN

BLOCKCHAINS COMPATÍVEIS À ETHEREUM VIRTUAL MACHINE (EVM)

A rede Ethereum pode ser pensada como uma **máquina de estado baseada em transações**, na qual os nós da rede P2P têm uma visão compartilhada de um estado global.

O usuário emite uma transação que representa uma transição de estado válida.

Então, os nós mineradores escolhem um conjunto de transações ainda não confirmadas da mempool para verificar sua validade, executam a **devida computação** e atualizam o estado (Tikhomirov, 2017).

BLOCKCHAIN

BLOCKCHAINS COMPATÍVEIS À ETHEREUM VIRTUAL MACHINE (EVM)

A rede Ethereum pode ser pensada como uma **máquina de estado baseada em transações**, na qual os nós da rede P2P têm uma visão compartilhada de um estado global.

O usuário emite uma transação que representa uma transição de estado válida.

Então, os nós mineradores escolhem um conjunto de transações ainda não confirmadas da mempool para verificar sua validade, executam a **devida computação** e atualizam o estado (Tikhomirov, 2017).



Algoritmo de Consenso
Proof-of-Work

BLOCKCHAIN

HYPERLEDGER

Hyperledger Fabric (HLF) é um framework open-source para desenvolvimento de soluções e aplicações distribuídas, à nível empresarial, baseadas em blockchain permissionada. Através do design modular e versátil, pode satisfazer uma variedade de casos de uso da indústria (Gaur et al. 2018).

BLOCKCHAIN

HYPERLEDGER

COMPONENTES DO HYPERLEDGER FABRIC

Peers

1. Endorser peer, que recebe a solicitação para validar a transação e executar o chaincode;
2. Anchor peer, os quais recebem mensagens e as enviam para outros peers da organização;
3. Orderer peer, que cria, ordena e anexa os blocos ao livro-razão.

Ordering Service

Gerencia o fluxo das transações entre os componentes da rede.

Chaincode

Lidam com a lógica de negócios da rede, sendo executados nos peers.

BLOCKCHAIN

HYPERLEDGER

COMPONENTES DO HYPERLEDGER FABRIC

Livro-razão

1. Estado mundial, que é um banco de dados contendo os valores atuais de um conjunto de estados do livro-razão, e
2. Blockchain, que é o log de transações que registra todas as mudanças que resultaram no estado mundial, sendo coletadas em blocos que são anexados à blockchain.

Membership Services Provider (MSP)

Utilizado para o gerenciamento do certificado digital, identificações de usuário e autenticação de todos os participantes na rede. Apenas os membros com identidades conhecidas podem executar transações.

DISCUSSÕES E LIÇÕES APRENDIDAS

LÓGICA DE APLICAÇÃO E ARMAZENAMENTO

- Utilização da biblioteca **Web3.js**, criando-se middlewares e fluxos únicos e modularizados para as diferentes blockchains.
- Uso de criptografia para o armazenamento de informações e geração de tokens de acesso às rotas.
- Boas práticas para o desenvolvimento de uma API.