CENTRO UNIVERSITÁRIO LUTERANO DE PALMAS

Recredenciado pela Portaria Ministerial nº 1.162, de 13/10/16, D.O.U. nº 198, de 14/10/2016
AELBRA EDUCAÇÃO SUPERIOR - GRADUAÇÃO E PÓS-GRADUAÇÃO S.A.

Natanna Rocha Santos

BLOCKCHAIN PARA A VALIDAÇÃO DOS CERTIFICADOS DA PLATAFORMA SIG EXTENSÃO DO CEULP/ULBRA

Palmas – TO

2022

Natanna Rocha Santos

BLOCKCHAIN PARA A VALIDAÇÃO DOS CERTIFICADOS DA PLATAFORMA SIG EXTENSÃO DO CEULP/ULBRA

Projeto de Pesquisa elaborado e apresentado como requisito parcial para aprovação na disciplina de Laboratório de Criação do curso de bacharel em Ciência da Computação pelo Centro Universitário Luterano de Palmas (CEULP/ULBRA).

Orientador: Prof. Esp. Fábio Castro Araújo.

Palmas – TO

2022

Natanna Rocha Santos

BLOCKCHAIN PARA A VALIDAÇÃO DOS CERTIFICADOS DA PLATAFORMA SIG EXTENSÃO DO CEULP/ULBRA

Projeto de Pesquisa elaborado e apresentado como requisito parcial para aprovação na disciplina de Laboratório de Criação do curso de bacharel em Ciência da Computação pelo Centro Universitário Luterano de Palmas (CEULP/ULBRA).

Orientador: Prof. Esp. Fábio Castro Araújo.

Aprovado em: _____/_____/_____

BANCA EXAMINADORA

_____

Prof Esp. Fábio Castro Araújo.

Orientador

Centro Universitário Luterano de Palmas – CEULP

_____

Prof. M.e Madianita Bogo Mariotti

Centro Universitário Luterano de Palmas – CEULP

_____

Prof. Esp. Fernanda Pereira Gomes

Centro Universitário Luterano de Palmas – CEULP

Palmas – TO

2022

RESUMO

SANTOS, Natanna Rocha. **BLOCKCHAIN PARA A VALIDAÇÃO DOS CERTIFICADOS DA PLATAFORMA SIG EXTENSÃO DO CEULP/ULBRA**. 2021. 28 f. Projeto Tecnológico (Graduação) – Curso de Ciência da Computação, Centro Universitário Luterano de Palmas, Palmas/TO, 2021[1].

O presente trabalho apresenta um sistema *Blockchain* para ajudar na validação dos certificados da plataforma SIG Extensão. A plataforma SIG Extensão é responsável por auxiliar na administração das atividades e eventos de extensão, além de emitir os certificados da instituição de ensino CEULP. O trabalho foi pensado levando em consideração que a plataforma SIG Extensão, que disponibiliza os certificados emitidos pela instituição, possui um sistema de validação inadequado para garantir a veracidade dos dados presentes nos documentos. Diante disso, esta monografia concentra-se no desenvolvimento de um sistema utilizando *Blockchain* Privada e *Smart Contract* para garantir a veracidade dos certificados existentes na plataforma SIG Extensão usando as assinaturas das coordenações de Curso, Extensão e Reitoria para aplicar o consenso entre as partes responsáveis por realizar a validação dos blocos inseridos na cadeia *Blockchain*.

Palavras-chave: Certificados, Validação, *Blockchain*.

1

LISTA DE FIGURAS

LISTA DE ABREVIATURAS E SIGLAS

CEULP	Centro Universitário Luterano de Palmas
EVM	*Ethereum Virtual Machine*
IoT	*Internet of Things*
P2P	*Peer-to-peer*
PDF	*Portable Document Format*
ULBRA	Universidade Luterana do Brasil

SUMÁRIO

1 INTRODUÇÃO

"Um sistema distribuído é aquele no qual os componentes localizados em computadores interligados em rede se comunicam e coordenam suas ações apenas passando mensagens" (COULOURIS et al., 2013, p. 9). A utilização desse tipo de sistema favorece a descentralização e o compartilhamento de recursos para que possam ser aproveitados pelos serviços em que forem necessários. Segundo Tanenbaum e Steen (2007, p. 6), "um sistema distribuído é um conjunto de computadores independentes que se apresenta a seus usuários como um sistema único e coerente". Desse modo, os sistemas distribuídos podem ser constituídos de diferentes sistemas operacionais e redes, e ainda assim manter sua consistência e organização aparentando ser construído de um único elemento computacional.

Os sistemas distribuídos vêm ganhando espaço e novas tecnologias estão sendo aplicadas em seu contexto, como o recurso computacional denominado *Blockchain.* De acordo com Carvalho (2019, p. 7), "a razão para o interesse na *Blockchain* é sua capacidade de fornecer segurança, anonimato e integridade de dados sem qualquer terceiro no controle das transações", sendo essas características necessárias em sistemas que objetivam garantir a autenticidade e a segurança de suas informações. Nesse contexto, Vidal (2020, p. 37) argumenta que

> "A *Blockchain* incorporou o princípio da distribuição e acrescentou a criptografia. Como resultado ela conseguiu obter uma maneira de compartilhar informações de uma forma realmente segura, a prova de adulteração, ganhando reconhecimento ao garantir propriedades como transparência e fiabilidade".

A tecnologia *Blockchain* garante que os dados que são armazenados na sua estrutura de blocos não sejam alterados, proporcionando uma imutabilidade e segurança das informações. Pois a *Blockchain* pode ser caracterizada como uma cadeia de blocos, de modo que cada bloco adicionado na rede se conecta com o bloco anterior. Explicando de maneira mais técnica

> "Um *Blockchain* pode ser referido como um banco de dados distribuído, que armazena cronologicamente uma cadeia de dados empacotada em blocos selados de maneira segura e imutável. A cadeia de blocos, também chamada de *ledger*, está em constante crescimento, portanto, novos blocos estão sendo anexados ao final do *ledger*, em que cada novo bloco contém uma referência (mais precisamente um valor hash) para o conteúdo do bloco anterior" (TURKOVIC et al., 2018, p. 3).

As cadeias de blocos, também chamadas de nós, que armazenam as informações, são o que compõem a arquitetura do sistema da *Blockchain*, onde todos os blocos existentes nessa cadeia contém um valor de referência único criados por um sistema matemático de criptografia denominado *hash.* Esse valor *hash* é a base de construção da cadeia de blocos atualmente utilizada na *Blockchain*, pois o valor existente no bloco anterior será a referência para o bloco que virá a ser criado, construindo um encadeamento entre os nós.

Segundo Pereira (2011, p. 48), "funções do tipo *hash* são responsáveis por garantir a integridade de mensagens, calculando um código único e identificador para cada mensagem". Caso alguma informação seja alterada, a chave *hash* será modificada, garantindo a visualização de mudanças no conteúdo das informações enviadas. De acordo com Pierro (2017, p. 2), a "única maneira de adulterar os dados enquanto preservar o *hash* seria encontrar uma colisão entre os dados e isso é computacionalmente impossível. Seria exigido tanto poder de computação que é praticamente antieconômico".

Portanto, com a aplicação dos fundamentos de sistemas distribuídos, como a descentralização e o compartilhamento de recursos que estão presentes no *Blockchain*, é possível garantir a autenticidade e confiabilidade dos dados, de modo que, o presente trabalho visa oferecer uma alternativa menos burocrática que ajude a validar os certificados emitidos pelas instituições de ensino, podendo ser capaz de confirmar a autenticidade das informações ou a origem dos documentos armazenados em sistemas que disponibilizam os certificados de forma digital.

A plataforma SIG Extensão da instituição de ensino CEULP é um sistema que disponibiliza os certificados de forma digital, de modo que é possível confirmar a autenticidade de suas informações ou sua origem utilizando um código de registro em *hash*, que é único para cada documento, não sendo a melhor maneira de garantir as propriedades básicas da segurança das informações. E, obviamente, sistemas que se propõem a disponibilizar certificados *online* em instituições precisam de um sistema de validação adequado para garantir a veracidade e a validação dos certificados emitidos. Desta forma, a utilização de *Blockchain* em um sistema de validação que auxiliaria na veracidade das informações dos certificados seria uma maneira eficaz de evitar fraudes na sua autenticidade bem como na garantia da sua validade. Desse modo, seria interessante para as instituições a utilização de sistema de validação de certificados com a tecnologia *Blockchain,* pois estas aplicações podem facilitar o acesso às certificações e evitar o acúmulo excessivo de papel.

Com o intuito de garantir a veracidade dos certificados existentes na plataforma SIG Extensão usando *Blockchain* para a validação, o presente trabalho tem como objetivo

desenvolver uma *Blockchain* e *Smart Contract* para garantir a veracidade dos certificados emitidos na instituição de ensino CEULP, de modo que a validação é executada a partir de uma sistema simples, não sendo aplicado a plataforma do SIG Extensão. Desse modo, foi preciso analisar a atual forma de emissão e validação dos certificados na plataforma SIG Extensão, definir a estrutura das informações que farão parte da cadeia de blocos de armazenamento da *Blockchain,* definir algoritmo de criptografia para a implantação do *Blockchain* e implementar o *Blockchain* e o *Smart Contract*. Nesse sentido, utilizando linguagem *TypeScript* foi implementado uma *Blockchain* Privada e um Smart *Contract* que permite os setores de coordenação de Curso, Extensão e Reitoria realizarem a validação das informações dos certificados.

2 REFERENCIAL TEÓRICO

Esta seção apresenta uma visão geral dos tipos de certificados (subseção 2.1): certificados físicos (subseção 2.1.1), e certificados eletrônicos (subseção 2.1.2), compreende sobre os tipo de seguranças (2.2) e mais profundamente sobre criptografia (2.2.1) e os recursos utilizando para criptografar, o *hash* (2.2.1.1), aborda também sobre Certificados Digitais (2.3) e a tecnologia Blockchain (2.4), com suas mais variáveis aplicações (2.4.1), além disso, abrange sobre os sistemas de certificados eletrônicos (2.5) e mais profundamente sobre a plataforma SIG Extensão do Centro Universitário Luterano de Palmas (2.5.1) e concluí argumentando sobre contratos inteligentes (2.4).

2.1 Certificados

Os certificados são meios de comprovação da participação em um determinado curso, conclusão de ensino superior ou até mesmo de um evento. Essa importância atribuída aos certificados se deve às informações contidas em sua estrutura, como: carga horária; tipo de curso ou evento; e ministrante; evidenciando a realização da atividade e comprovando sua conclusão.

Segundo Araújo e Vieira (2014, p. 4), "nossa sociedade vive desde a metade do século passado uma mudança na forma de criar, armazenar, circular e validar os documentos produzidos". Com essas mudanças vêm surgindo novas maneiras de emitir certificados e o mais conhecido atualmente é o formato digital.

2.1.1 Certificados Físicos

De acordo com Souza, Carneiro e Coutinho (2021, p. 1), os certificados são categorizados como documentos arquivísticos, que são concedidos por uma instituição de ensino, que confirma que o portador cumpriu as exigências necessárias para obter um título e

que possui as adequações para exercer determinada atividade. Por seu valor, vários indivíduos optam por utilizar a versão física do documento, fazendo com que o modelo em papel seja uma maneira de aquisição do certificado, visto que, culturalmente, o papel em mãos representa segurança.

Vidal (2020, p. 2) afirma que o modelo em papel ainda é frequentemente utilizado, e que só pode ser destruído por quem o possui fisicamente, de modo que o documento impresso não oferece liberdade para as entidades emissoras cancelarem os registros que apresentarem erros. Outros problemas que os certificados físicos podem apresentar é o acúmulo de papel com o passar do tempo podendo encadear na ocupação de espaço, além da possibilidade de haver a perda do documento que, ainda, pode apresentar sinais de maus cuidados, entre outros problemas relacionados ao uso de papel. Entretanto, com o "documento em papel tem-se acesso direto ao conteúdo sem auxílio de equipamentos" (ARAÚJO; VIEIRA, 2012, p. 293).

De acordo com Gandini, Salomão e Jacob, (2017, p. 4), "as pesquisas demonstram que os documentos impressos estão sendo gradualmente substituídos por arquivos eletrônicos, mesmo que por mais de vários anos todos os conhecimentos humanos e as informações foram armazenados em documentos de papel". A sociedade vem adquirindo novos meios de armazenar as informações de modo digital, como uma maneira de facilitar a busca e a preservação das informações.

2.1.2. Certificados Eletrônicos

A criação e edição de documentos em computadores proporcionou uma grande facilidade e agilidade, tanto ao mundo corporativo quanto ao ambiente acadêmico e também na gestão universitária (PAVANATI et al., 2007, p. 3). Com a adoção do uso do computador e de novas tecnologias, surgiram novas maneiras de produção e de armazenamento de certificados. Segundo Boesing et al. (2015, p. 8), sistemas de emissão de certificados *online* possuem o intuito de ajudar na comunicação institucional e mercadológica que vem surgindo na sociedade da informação, com o objetivo de reduzir o tempo na emissão dos certificados e facilitar a entrega aos participantes.

Explicando de maneira mais técnica,

> "O Documento eletrônico apresenta características específicas que não estão presentes no documento tradicional em papel. No Documento em papel tem-se acesso direto ao conteúdo sem auxílio de equipamentos. Os eletrônicos, por sua vez, estão armazenados na forma de um conjunto de bits formatada segundo algum padrão

> de representação para um formato mais apropriado a compreensão humana" (ARAÚJO; VIEIRA, 2014, p. 4).

A facilidade que os certificados digitais proporcionam é evidente, fazendo com que esse novo formato de armazenamento de documentos seja cada vez mais aderido pelas instituições de ensino no geral. Pois acessar e obter o certificado para comprovação de documentos, acaba se tornando mais acessível e explícito a partir do uso de plataformas de certificados eletrônicos.

2.2. Segurança

De acordo com Gandini, Salomão e Jacob (2002),

> "A segurança da nova modalidade de transmissão de informações tem sido questionada, visto que é ela quem garante o sucesso das transações; no entanto, mecanismos informáticos foram desenvolvidos com finalidade de nos fazer acreditar na total segurança dos arquivos digitais, enviados por correio eletrônico".

Com o intuito de manter seguros os dados que transitam pela rede de informações, surgiram maneiras de garantir a segurança das informações. Ohtoshi (2013, p. 48) afirma que, "uma das formas de se evitar o acesso indevido a informações confidenciais é codificar ou cifrar a informação de forma que somente as pessoas às quais a informação se destina sejam capazes de compreendê-las. Essa técnica de codificação é conhecida como criptografia". A criptografia é capaz de proporcionar os recursos de segurança da informação fundamentais dos principais da segurança, que são a confiabilidade, integridade e disponibilidade.

2.2.1 Criptografia

Segundo Alecrim (2005), a criptografia trata-se de um conjunto de bits baseado em um determinado algoritmo capaz de codificar e decodificar informações, como ilustrado na Figura 2. Se o receptor da mensagem usar uma chave incompatível com a chave do emissor, não conseguirá extrair a informação. A partir dessas características, a criptografia foi ganhando espaço como meio de garantir a seguridade dos dados que navegam através da internet.

Figura 1. Criptografia (ALECRIM, 2009).

De acordo com Araújo e Vieira (2012, p. 5), criptografia dispõe de determinados recursos para garantir:

- Autenticação: garante a origem da informação;
- Integridade: assegurar a veracidade e a integridade da informação recebida;
- Confidencialidade: acesso às informações somente pelas pessoas autorizadas;
- Irretratabilidade: a origem da mensagem não poderá negar que foi o criador da mensagem enviada.

Portanto, a criptografia é uma maneira de esconder informações de uma terceira parte, dificultando e tornando o trabalho para acessar a informação das operações impossíveis quando não se tem permissão para tal (SILVA, 2013, p. 12). Essa forma de ocultar as informações só é possível por causa dos cálculos de funções *hash*. Algoritmo criado para criptografia, no qual é possível transformar dados em um conjunto de combinações entre letras do alfabeto e números, possuindo um comprimento padrão.

2.2.1.1 Hash

"O uso da função *hash* é bastante utilizado em assinaturas digitais, já que para uma sequência de bits analisado de uma informação é gerado um valor único e caso essa informação seja modificada um novo valor será gerado" (SILVA, 2013, p. 14). Por apresentar esse funcionamento, o *hash* é mais utilizado para realizar comparações entre os valores gerados na função matemática. Segundo Vale (2020),

> "O algoritmo *Hash* é conhecida como uma função matemática criptográfica, na qual você possui dados de entrada e, após passar pela criptografia, eles apresentam valores de saída "padronizados", ou seja, as saídas devem possuir o mesmo tamanho (geralmente entre 128 e 512 bits) e o mesmo número de caracteres alfanuméricos. "

Ferreira et al. explica que o, "*Hash* é um algoritmo que transforma dados grandes e de tamanho variável para pequenos dados de tamanho fixo através de operações e cálculos matemáticos, onde uma mesma entrada gera sempre o mesmo resultado e entradas diferentes, um bit que seja, geram resultados bem diferentes". Com a aplicação de *hash* é possível prevenir fraudes e garantir a autenticidade das informações. Pois os valores de *hashes* são únicos, e qualquer mudança mudaria completamente sua estrutura e valores do *hash*. Por isso as funções matemáticas *hash* são muito aplicadas como código de comparação e também como chaves para a busca de dados coerentes.

Abreu (2020, p. 33), aponta que a função *hash* possui características importantes como:

- A resistência à colisão presente na função *hash*, isso acontece por causa de seu funcionamento, onde duas entradas diferentes não produzem a mesma saída;
- A ocultação de informações, de modo que é dada apenas a saída, ninguém pode inferir o valor da entrada;
- A facilidade de quebra-cabeça, onde a função *hash* pode ser apresentada na forma de um quebra-cabeça matemático, de modo que é possível tentar diferentes entradas para a função *hash* para obter uma saída com o valor da entrada.

2.3 Certificado Digital

Os sistemas de certificados digitais surgiram como um meio de garantir a segurança das informações dos documentos emitidos pelas instituições do setor financeiro e educacional. Segundo Pavanati *et al.* (2017, p. 4),

> "A certificação digital foi desenvolvida para garantir autenticidade especificamente no ambiente virtual. Trata-se de um sistema de criptografia assimétrica que codifica a mensagem através de uma chave criptográfica (chave privada) e, em seu destino, é decodificada com outra chave criptográfica (chave pública)".

De acordo com Freitas e Veronese (2007, p. 11), poucas pessoas sabem que o sistema de certificação digital procura garantir a autenticidade das informações que são enviadas para a internet, informando e identificando ao receptor quem é o emissor daquelas informações. Possibilita também o fluxo de mensagens criptografadas (não visualizáveis facilmente), que garantem o sigilo na comunicação. Desse modo, os sistemas de certificação digitais têm como prioridade garantir a autenticidade e confiabilidade das informações contidas nos documentos que são emitidos pelas instituições. "Os certificados digitais, também chamados de identidade digital, é um arquivo de computador capaz de identificar dados de um indivíduo ou entidade, possuindo chaves para fazer a certificação" (ARAÚJO; VIEIRA, 2012, p. 5).

Segundo Friedrich e Medina (2007, p. 3),

> "A utilização deste tipo de certificado atualmente vem ganhando importância para garantir a segurança das informações que transitam na grande rede principalmente pelos inúmeros benefícios trazidos, como por exemplo, para realização de transações comerciais eletrônicas que envolvem informações críticas como senhas e números de cartões de crédito. "

A partir disso, os certificados digitais surgem como garantia de manter as informações das certificações seguras e bem armazenadas. De acordo com Jimene e Blum (2010, p. 1), o certificado digital nada mais é que um documento armazenado de maneira tecnológica, possuindo um suporte digital.

Menke (2003, p. 2) afirma que, "as assinaturas e os certificados digitais servem para agregar os valores confiança e segurança às comunicações e negócios veiculados em ambiente virtual, especialmente na internet". Desse modo, os certificados digitais facilitam o acesso aos documentos, permitindo às solicitações dos registros a partir de qualquer lugar, além de proporcionar a redução de custo e a perda de dados importantes. Os certificados digitais são assinados pelas instituições de ensino que os oferecem, confirmando os dados do documento a partir da assinatura digital, identificando que o documento é original e que possui validade pela instituição. A validação a partir de certificados digitais pode ser desenvolvida utilizando os mais diversos tipos de tecnologias da atualidade, como por exemplo a *Blockchain*.

2.4 *BLOCKCHAIN*

"*Blockchain* é uma tecnologia emergente que oferece suporte distribuído confiável e seguro para realização de transações entre participantes que não necessariamente têm confiança entre si e que estão dispersos em larga escala numa rede P2P" (GREVE, 2018, p. 2). Por possuir uma estrutura baseada em sistemas distribuídos qualquer aplicação ou computadores, que não possui dependência entre si, pode se comunicar dentro do sistema da *Blockchain*. Essa comunicação ocorre através da troca de mensagens entre os sistemas operacionais.

Segundo Zheng *et al.* (2016),

> "O *Blockchain* tem inúmeros benefícios, como descentralização, persistência, anonimato e capacidade de auditoria. Há um amplo espectro de aplicativos de *Blockchain* que variam de criptomoeda, serviços financeiros, gerenciamento de risco, Internet das coisas (IoT) a serviços públicos e sociais."

A *Blockchain* surgiu como proposta para criptomoeda *Bitcoin*, porém pode ser aplicada em vários contexto diferentes. Sua funcionalidade não se limita apenas ao uso de transferências no meio financeiro. Com isso, sua finalidade se expande para demais áreas, desde gerenciamento de informações no contexto educacional aos serviços públicos e sociais.

Como representado na Figura 1, um *Blockchain* consiste em conjuntos de dados compostos por uma cadeia de pacotes de dados (blocos), onde um bloco compreende

múltiplas transações (NOFER *et al*., 2017, p. 1). Sendo composta de encadeamento de blocos, que contribui com a função de guarda e ampara o compartilhamento de informações.

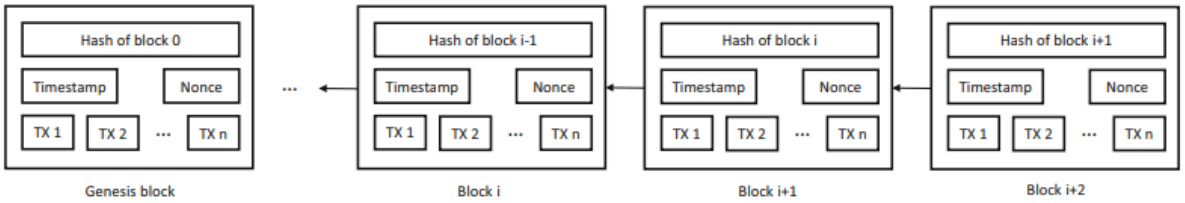

Figura 2. Sistematização da cadeia de blocos na *Blockchain* (ZHENG *et al*. 2016).

A *Blockchain* é baseada num algoritmo matemático que, através de uma corrente de blocos, identifica uma transação realizada virtualmente. A cadeia de blocos formada após a operação fica registrada e replicada em diversos servidores responsáveis por validar, por consenso, o registro (MOURA; BRAUNER; MUNIZ, 2020). Cada bloco é capaz de armazenar as informações de modo que não podem ser deletadas depois, garantindo a imutabilidade dos dados. Essa característica da *Blockchain* é evidenciada em Mougayar (2017), onde o autor afirma que a tecnologia contida na *Blockchain* é capaz de gravar permanentemente transações que não podem ser deletadas depois, mas podem ser atualizadas mantendo o histórico das transações.

Swan (2015, p.12), falar que a *Blockchain* é o livro-razão público, que contém todas as transações *Bitcoin* que já foram executadas. A cadeia de blocos cresce constantemente à medida que são adicionados novos blocos a ela, servindo para registrar as ações mais recentes. Os blocos são adicionados ao *Blockchain* em uma linha e em ordem cronológica.

2.4.1 Aplicações do *Blockchain*

O uso da tecnologia *Blockchain* não se limita apenas às criptomoedas. O *Blockchain* surgiu como a tecnologia chave por trás do *Bitcoin*, porém o conceito da tecnologia foi generalizado para a de um livro razão distribuído capaz de verificar e armazenar uma grande variedade de transações, incluindo as que não tem associação com criptomoeda (XU et al., 2019, p. 84). Várias outras áreas do conhecimento buscam utilizar suas características e praticidades para o armazenamento de dados e também em diversas aplicações.

De acordo com Swan (2015, p. 11), a tecnologia *Blockchain* é dividida em três categorias:

- *Blockchain* 1.0: a criptomoeda em aplicativos relacionados a movimentação de dinheiro e transações no geral.

- *Blockchain* 2.0: uso em contratos de todos os tipos, sendo sua aplicação mais extensas do que simples transações envolvendo dinheiro, como: ações, títulos, hipotecas e contratos inteligentes.
- *Blockchain* 3.0: são aplicações que vão além das utilidades apresentadas nas categorias anteriores. Suas aplicações são voltadas especialmente nas áreas do governo, saúde, ciência, alfabetização, cultura e arte.

Desse modo, a partir do *Blockchain* 3.0 a tecnologia apresenta outras finalidades, como o armazenamento de dados pessoais de usuários e empresas. O funcionamento da tecnologia apresenta um sistema seguro e praticamente inquebrável, garantindo a proteção dos dados sigilosos. Segundo Lucena e Henriques (2016, p. 1), é possível perceber que o conceito de conjuntos com os elementos encadeados do *Blockchain* é fortemente protegido contra adulterações sendo interessantes em outras aplicações em que as alterações dos dados não são desejadas e também em aplicações em que há a necessidade de garantia contra erros e modificações.

Com essas características presente na estrutura do *Blockchain* é possível usufruí-la de outras maneiras além da área financeira. Gates (2017, p. 36) demonstra em seu trabalho que a tecnologia *Blockchain* pode desempenhar papéis além das áreas voltadas para a economia, como em:

- Gerenciamento de identidade e identidades digitais;
- Votação Digital;
- Registros médicos e de saúde;
- Certificados Acadêmicos;
- Música;
- Armazenamento na nuvem;
- Indústria de viagens.

Para Greve (2018, p. 2), a tecnologia *Blockchain* apresenta um potencial de transformação imenso, podendo ser aplicada em inúmeros setores: finanças, saúde, artes, governo, além da própria computação: protocolos de redes, nuvem, IoT etc. Isso tudo se deve a capacidade da *Blockchain* de manter a integridade de dados e de seu mecanismo de criptografia responsável por garantir a segurança das operações realizadas. Levando em consideração todas as características da Blockchain, é possível criar sistemas que ajudem na validação das informações dos certificados eletrônicos que são emitidos pelas instituições de ensino a partir de sistema de certificação eletrônica.

2.5 Sistemas de Certificados Eletrônicos

Atualmente existem no mercado diversos sistemas para gerenciar eventos com emissão de certificados e declarações, que vão desde versões *free*, até versões pagas (LIMA, 2019, p. 4). Sobre essa premissa, a instituição de ensino do CEULP/ULBRA utiliza um sistema denominado SIG Extensão, que oferece várias funcionalidades aos alunos e principalmente a de certificação eletrônica. Desse modo, é possível gerar e disponibilizar os documentos de modo virtual.

2.5.1 SIG Extensão

Grande parte das universidades adotaram sistemas de certificação eletrônico com o intuito de tornar a parte administrativa mais eficiente, de fácil acesso e visualização para os acadêmicos. O Centro Universitário Luterano de Palma - CEULP/ULBRA possui uma plataforma para auxiliar na administração das atividades da instituição de ensino, denominado SIG Extensão. Em entrevistas com o especialista do domínio Fabio Castro Araújo, professor especialista no Centro Universitário Luterano de Palmas, foi possível obter informações e características sobre o funcionamento e estrutura da plataforma SIG Extensão.

De acordo com o especialista do domínio, o SIG Extensão foi criado para administrar e organizar eventos de extensão da instituição. De modo que permita aos usuários realizar inscrições, obter certificados, controlar os pagamentos etc. Com a utilização da plataforma, atividades relacionadas a eventos se tornaram mais fáceis e acessíveis, evitando a sobrecarga dos setores da instituição no momento da realização de eventos. Além da administração de eventos, o SIG Extensão possui uma funcionalidade que proporciona bastante praticidade, a geração e visualização de certificados digitais.

O sistema do SIG Extensão foi desenvolvido usando um modelo de plataforma *WEB*, utilizando a linguagem *Python*, juntamente com o *framework Django*. Sua arquitetura é baseada em camadas apresentando uma arquitetura MTV (*Model, Template e View*). Todos os dados das ações realizadas dentro da plataforma são armazenados em SQL Server, um tipo de banco de banco de dados, sendo ele próprio da instituição.

O SIG possui uma diversidade de funcionalidades, sendo divididas em funções voltadas para alunos e para os setores da instituição CEULP. A divisão dessas funcionalidades consiste em alunos sendo capaz de:

- Realizar inscrições em eventos;
- Visualizar os certificados de cada evento que participou;
- Visualizar eventos que estão ocorrendo na instituição;
- Validar certificados.

Enquanto os professores são capazes de:

- Gerenciar informações dos eventos;
- Gerenciar e confirmar as inscrições;
- Gerenciar os minicursos;
- Gerenciar os certificados.

As certificações são emitidas digitalmente pelo sistema SIG, por conseguinte o sistema verifica a participação dos usuários nos eventos ou cursos. Após a análise das informações é gerado um PDF do certificado baseado nas informações das atividades que o(a) acadêmico(a) realizou durante o evento em que se inscreveu. Além disso, cada certificado possui um código de registro para ajudar na pesquisa do documento na plataforma.

Segundo o especialista do domínio, para realizar a visualização dos certificados na plataforma SIG Extensão, basta o usuário entrar no sistema e acessar a área chamada de "Meus Certificados". Nessa área será exibida uma lista de todos os eventos que o acadêmico já participou, de modo que ao lado de cada evento é apresentado um botão de *download*. Ao selecionar o botão *download* de determinado evento, o aluno poderá baixar e visualizar seus certificados de participação.

A plataforma SIG Extensão dispõe de um sistema para realizar a validação dos certificados, de modo que cada certificado emitido pela instituição possui um código de registro. Esse código é criado a partir de um *hash* tornando possível a verificação da validade do certificado. Cada *hash* é único no sistema, tornando possível verificar se o documento foi emitido pela coordenação de extensão do CEULP. Ao utilizar o *hash* criptográfico como código para validar as certificações é possível garantir a origem dos certificados emitidos pela instituição.

De acordo com o especialista do domínio, cada certificação emitida apresenta um *hash* que funciona como um código que garante que as informações foram emitidas pelo CEULP. Desse modo, os acadêmicos que obtiverem os certificados podem utilizar o código em *hash* para verificar a autenticidade do documento por meio da funcionalidade de validação do SIG Extensão. Esse formato de validação utilizando *hash* é aplicado em diversos sistemas e tecnologias, uma delas é a tecnologia *Blockchain*.

2.6 Contratos inteligentes

"Um contrato inteligente é uma aplicação autônoma com entradas e saídas pré-definidas que podem ser executadas por um minerador de maneira determinística" (ABREU, 2020, p. 42). Os contratos inteligentes têm como finalidade satisfazer as condições elaboradas durante sua criação, evitando a necessidade de intermediários em suas operações.

Martinelli (2019, p. 150), afirma que,

> “Os contratos inteligentes (*Smart Contracts*) possuem grande similaridade com os tradicionais contratos legais os quais são firmados entre partes, mas em âmbito digital e são informações ou linhas de códigos aplicadas nas transações ocorrentes no *Blockchain*. Tais contratos não podem ser adulterados, o que garante a seguridade da transação “.

Desse modo, os contratos físicos podem ser substituídos pelo meio digital sem perder a sua natureza e a garantia de segurança. Os contratos inteligentes prometem ser cada vez mais aplicados em vários contextos, visto que podem proporcionar mais agilidade e segurança nas operações de transação.

Divino (2018, p.1) apresenta a definição de que os *Smart Contracts* são contratos digitais elaborados em um código de computador e armazenados na *Blockchain.* Dessa forma, podem ser autoexecutáveis, de caráter descentralizado, e que buscam pela praticidade, redução de custos e pelo anonimato. Já Schuettel (2017), afirma que “um contrato inteligente é um contrato online de acordo com base em aplicativos que executam exatamente como programado sem qualquer possibilidade de indisponibilidade, censura, fraude ou interferência de terceiros”. Reafirmando que os *Smart Contracts* garantir a segurança das informações contidas nas aplicações em que forem adotadas.

De acordo com Buterin (2014), os contratos inteligentes são "caixas" criptográficas que contêm valor e só os desbloqueiam se certas condições forem atendidas. Portanto, os *Smart Contracts* ou contratos inteligentes são geralmente executados automaticamente, apresentando ações predefinidas na sua estrutura de código. De modo, esses comandos de códigos só são executados quando cumprem as condições estabelecidas pelas partes.

3 METODOLOGIA

3.1 Materiais

As tecnologias e os materiais que foram utilizados no desenvolvimento do trabalho proposto serão apresentados e descritos a seguir.

3.1.1 Linguagem *TypeScript*

Segundo Bierman, Abadi e Torgersen (2014), “TypeScript é uma extensão do JavaScript destinada a facilitar o desenvolvimento de aplicativos JavaScript em larga escala. Embora todo programa *JavaScript* seja um programa *TypeScript*, o *TypeScript* oferece um sistema de módulos, classes, interfaces e um rico sistema de tipos graduais**“.** A linguagem *TypeScript* surge como uma maneira de auxiliar os desenvolvedores de *Javascript* a

programarem em POO (Programação Orientada a Objeto). Pois ao aplicar as duas juntamente é possível melhorar o suporte a POO, que não é possível utilizando somente a linguagem *Javascript* pura (NOLETO, 2020). Com a utilização dessa linguagem de programação será possível criar a estrutura de funcionamento da *Blockchain* e dos *Smart Contract* do projeto desenvolvido neste trabalho.

3.1.2 Certificados da Plataforma SIG Extensão

Documento emitido pelo sistema SIG Extensão, como comprovante de participação das atividades realizadas na instituição de ensino CEULP/ULBRA. Os certificados atualmente são gerados usando o apoio do SIG e de um sistema de mala direta para preencher as informações e dados que compõem o documento.

Figura 3. Certificados do CEULP/ULBRA.

As Figuras 3 e 4 exemplificam o formato dos certificados emitidos pelo sistema SIG Extensão. Na Figura 4 é possível visualizar o código de registro dos certificados emitidos pela instituição de ensino ULBRA.

Figura 4. Segunda parte dos certificados com o código de registro em *hash*.

O código é gerado em *hash,* servindo como validador para o documento emitido. Os certificados serão utilizados no projeto como material fundamental no processo para confirmar a validação da plataforma desenvolvida no trabalho.

3.2 MÉTODOS

Para o planejamento e desenvolvimento do presente trabalho, os processos realizados foram divididos em etapas. A estruturação da metodologia de desenvolvimento do trabalho é apresentada na Figura 5.

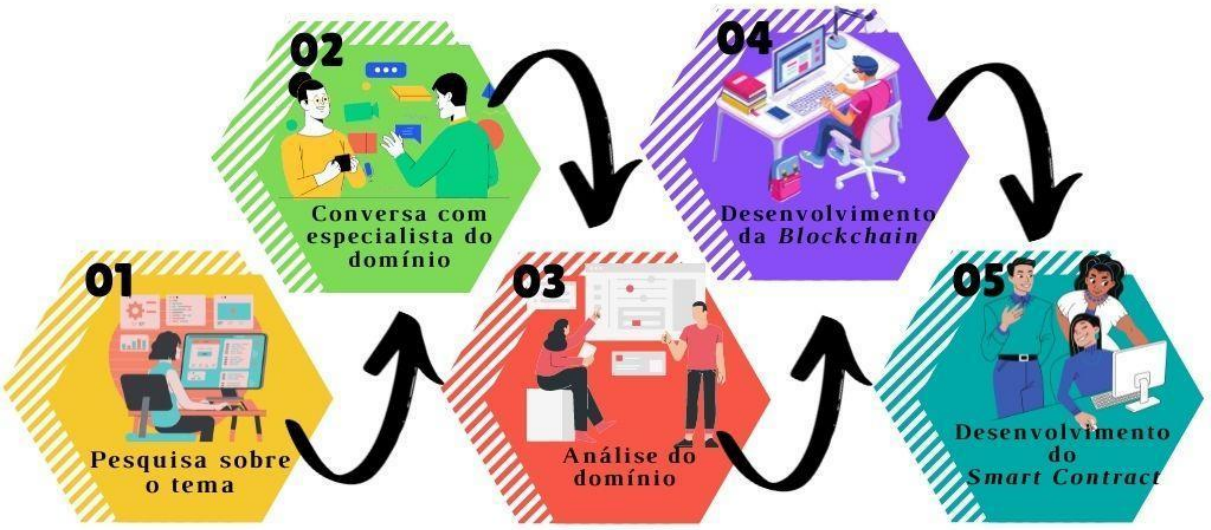

Figura 5. Metodologia aplicada no projeto.

A metodologia apresentada na Figura 5 representa as etapas que foram seguidas para a conclusão do objetivo proposto. Na primeira etapa foi realizada uma pesquisa sobre o tema e as tecnologias que seriam utilizadas no decorrer da proposta. Essa etapa foi iniciada logo após obter o tema proposto, que foi definido junto ao orientador e professor especialista Fábio Castro Araújo. A partir da definição da proposta do trabalho, que é focado em uma plataforma que utiliza sistema distribuído com *Blockchain* para auxiliar na validação de certificados da plataforma SIG Extensão, foi possível identificar e pesquisar os fundamentos das tecnologias que serão aplicadas.

Na etapa 2 foram realizadas algumas reuniões com o especialista do domínio para obter mais informações sobre a plataforma que seria estudada. A conversa foi realizada com o professor especialista Fábio Castro Araújo, sobre a plataforma SIG Extensão utilizada na instituição de ensino do CEULP/ULBRA. A conversa foi necessária para entendimento das funcionalidades, da estrutura, das tecnologias que foram utilizadas para a criação da plataforma e também para o esclarecimento das aplicações envolvendo a criação dos certificados na plataforma SIG Extensão.

Após a conversa com a especialista do domínio foi realizada a etapa 3, que consistiu na observação e utilização da plataforma SIG Extensão. A utilização da plataforma foi realizada com a finalidade de identificar as funcionalidades e composição do SIG Extensão, além de observar como a plataforma gera os certificados dos eventos cadastrados no sistema, as informações presentes nos certificados e o seu modo de validação dos documentos de certificação.

A etapa 4 consistiu na elaboração da estrutura e no desenvolvimento da *Blockchain* privada que será utilizada pelos setores da instituição ULBRA para a validação dos certificados eletrônicos que são emitidos. A *Blockchain* privada desenvolvida tem como foco não realizar a mineração de dados/blocos como as *Blockchains* públicas, mas apenas validar as informações com base no consenso entre os principais participantes, como as coordenações responsáveis por aprovar os certificados da instituição.

Após a criação da *Blockchain*, a etapa 5 consistiu na elaboração da estrutura dos componentes do *Smart Contract* e a forma de verificação aplicada para as assinaturas. A estrutura foi elaborada com base nos dados existentes nos certificados e nas 3 partes que compõem a aprovação dos certificados: Reitoria, Coordenação de Extensão e a Coordenação do respectivo curso da atividade elaborada. Dessa forma, o sistema valida as informações dos certificados a partir da confirmação do contrato entre as 3 partes existentes da instituição.

4 RESULTADOS

O presente trabalho trata-se da criação de uma *Blockchain* e *Smart Contract* para garantir a veracidade dos certificados emitidos na instituição de ensino CEULP, de modo que a validação é executada a partir de uma sistema simples, não sendo aplicado a plataforma do SIG Extensão e sem interface visual. O trabalho desenvolvido nessa seção, mostra que a *Blockchain* juntamente com o *Smart Contact* são tecnologias capazes de proporcionar segurança das informações contidas nos certificados, garantindo a confiabilidade dos dados armazenados em sua cadeia de blocos. A imagem 6 abaixo demonstra o funcionamento e fluxo de como funciona a validação através do consenso entre os setores.

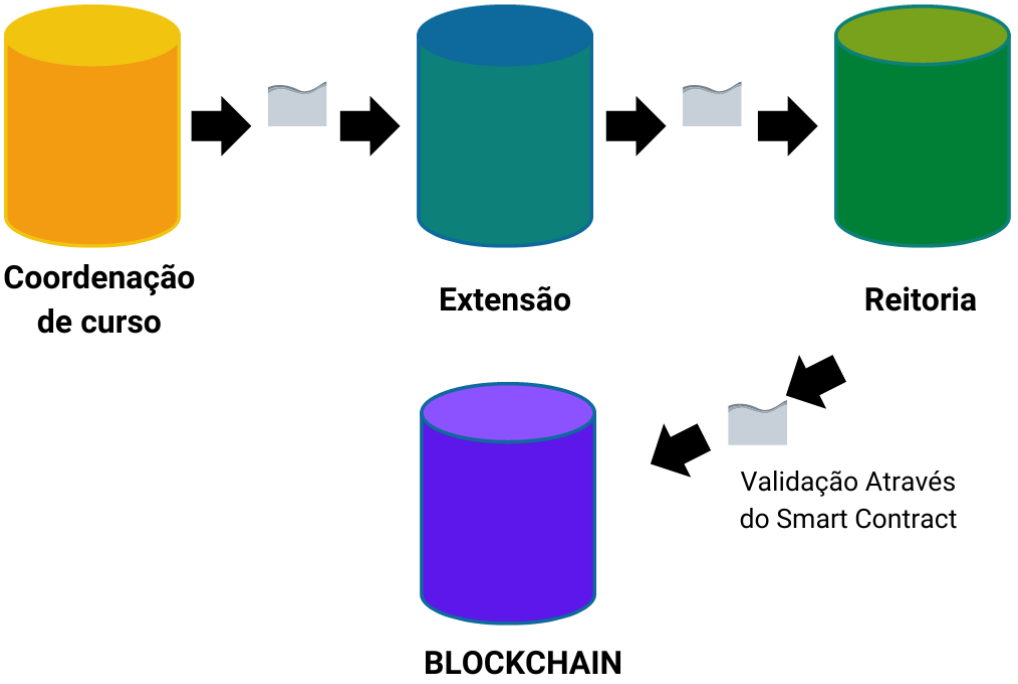

Figura 6. Funcionamento da estrutura do projeto.

O funcionamento do sistema consistem em as informações do certificado serem inseridas pelo setor de coordenação de curso e logo depois ser assinada por este setor. Em seguida a informações do certificado é passada para os demais setores que ainda não realizaram a validação para que possa ser autenticado e assinado por eles, depois de todos setores realizarem a assinatura e verificação das informações é realizado o consenso entre as partes através do *Smart Contract* para verificar se todos os setores confirmaram a veracidade das informações. Ao final, com o consenso aplicado em os 3 setores: Coordenação de Curso, Extensão e Reitoria, as informações do certificado são adicionadas dentro da Blockchain, gerando um novo bloco validado na cadeia, onde o novo bloco possui as informações do certificado e as assinaturas dos 3 setores.

As etapas para o desenvolvimento do sistema foram divididas em seções que consistiram na implementação da estrutura da *Blockchain*, seguindo da criação do *Smart Contract* onde foi criada a função para gerar as chaves para assinatura e a função para verificação dos blocos.

4.1 Implementação da *Blockchain* Privada

Para a implementação do projeto foi preciso pesquisar sobre a tecnologia *Blockchain*. A partir da pesquisa realizada, foi possível identificar diferentes tipos da tecnologia *Blockchain* e qual seria mais adequada para a aplicação no desenvolvimento do trabalho. Para o presente trabalho foi identificado que a *Blockchain* Privada seria melhor aplicada, pois os únicos a fazer parte do processo de validação dos certificados dentro da *Blockchain* serão os três setores responsáveis da Instituição de Ensino CEULP: Reitoria, Extensão e as Coordenações dos respectivos Cursos. Outro motivo para escolher a utilização de uma *Blockchain* Privada, é que o trabalho visa apenas a validação e autenticação dos certificados por meio da tecnologia de cadeia de blocos que a *Blockchain* fornece, sem a necessidade de uso de mineração de incentivos para os participantes a cada novo bloco inserido na cadeia de validações.

A implementação do projeto para validação dos certificados começou com a escolha da linguagem com a qual seria implementada a estrutura da *Blockchain.* A linguagem escolhida foi a *Typescript*, ideal para a criação de *scripts* e, também, por possuir apoio à programação orientada a objetos. Logo após a escolha da linguagem, foi definido como seria a estrutura e os dados que iriam fazer parte da composição dos blocos que seriam adicionados na *Blockchain*. As informações escolhidas são apresentadas na Figura 6.

```
6  interface InfoBloco{
7      header: {
8          nonce?: number;
9          hashBloco: string;
10     }
11     payload: {
12         indexBloco: number;
13         timestamp: number;
14         dado: any;
15         hashBlocoAnterior: string;
16     }
17 }
```

Figura 7. Estrutura do Bloco

Com base nos estudos realizados sobre a criação de blocos da *Blockchain*, foi criada uma interface denominada `InfoBloco`, que auxilia na criação de um objeto dentro do código *TypeScript.* Ela possui os métodos e propriedades dos blocos que serão adicionados na *Blockchain*. Na Figura 6 é possível visualizar a estrutura dentro da interface do `InfoBloco`, que possui um *`header`* e o *`payload`*. O *`header`* criado na linha 7 (sete) possui os parâmetros *`nonce`* e o `hashBloco`, o *`nonce`* é o número único para informar a quantidade de vezes que esse bloco teve que ser executado para que seja validado, e o `hashBloco` é o parâmetro que irá receber o *hash* que foi criado e validado para a criação do bloco na rede *Blockchain*. O *`payload,`* da linha 11 (onze), possui os parâmetros com as informações que serão armazenadas dentro do bloco, como:

- `indexBloco`: Que indica o número do bloco que acabou de ser gerado dentro da cadeia;
- `timestamp`: Responsável por informar a quantidade de tempo que levou para a geração do bloco;
- `dado`: Que receber o *hash* de todas as informações que serão armazenados no bloco, como as informações dos certificados e as assinaturas dos setores;
- `hashBlocoAnterior`: Variável que armazena o *hash* do bloco que foi criado e adicionado por último na cadeia de blocos.

Logo após a criação da interface, foi codificada a classe da *Blockchain* Privada, que possui todas as funções necessárias para a criação e inserção de blocos na cadeia *Blockchain,* Esses blocos, criados a partir desta classe, serviram como prova de validação dos certificados.

Dentro da classe Blockchain foi criada uma lista, linha 21 da Figura 7, denominada `#cadeiaBlock`, responsável por armazenar todos os blocos que forem adicionados na rede, e também foi criado uma constante chamada de `prefixo`, que servirá como parâmetro de identificação para todo bloco que foi adicionado no sistema (Figura 7). Esse parâmetro `prefixo` será transformado em *hash* e será concatenado ao início de todos os *hash* gerados das informações dos blocos que forem inseridos na *Blockchain,* servindo como parâmetro de identificação para os blocos gerado neste projeto, pois caso o *hash* de um bloco qualquer não possua o início igual ao *hash* gerado pela constante definida no prefixo, esse bloco não foi gerado pela *Blockchain* criada nesse projeto.

```
20  export class Blockchain{
21      #cadeiaBlock: InfoBloco[] = [];
22      private prefixo: string = "BlockTeste";
```

Figura 8. Lista de Blocos e prefixo.

Logo após a criação desses parâmetros iniciais, foi criada a função responsável por iniciar a cadeia de blocos, a função `BlocoGenesis()` na linha 28 da Figura 8. Essa função cria o primeiro bloco a ser inserido na *Blockchain*, por isso leva o nome de Gênesis, por ser o bloco responsável por começar a rede, sendo a peça inicial da construção de todo o sistema *Blockchain*.

```
28      private BlocoGenesis(): InfoBloco{
29          const payload: InfoBloco["payload"] = {
30              indexBloco: 0,
31              timestamp: +new Date(),
32              dado: "Bloco Genesis da cadeia de blocos de validação de certificados",
33              hashBlocoAnterior: ''
34          }
35
36          return{
37              header: {
38                  nonce: 0,
39                  hashBloco: gerarHash(this.prefixo) + gerarHash(JSON.stringify(payload))
40              },
41              payload
42          }
43      }
```

Figura 9. Bloco Gênesis.

A função `BlocoGenesis()` recebe como parâmetro a interface InfoBloco, pois ela que guarda a estrutura que todo novo bloco que for adicionado a rede deve possuir. Por ser o bloco inicial, seu parâmetro `indexBloco` recebe o valor 0 (zero), indicando que é o primeiro bloco a ser adicionado, além disso o seu parâmetro `hashBlocoAnterior` recebe uma *string* vazia indicando que não possui nenhum bloco anterior a ele na cadeia *Blockchain* (Figura 8).

```
53  criarBloco(dados: any): InfoBloco["payload"]{
54      const NovoBloco: InfoBloco["payload"] = {
55          indexBloco: this.ultimoBloco.payload.indexBloco + 1,
56          timestamp: +new Date(),
57          dado: dados,
58          hashBlocoAnterior: this.hashUltimoBloco()
59      }
60      console.log("\n" + "Bloco " + NovoBloco.indexBloco + " criado: " + JSON.stringify(NovoBloco) + "\n")
61      return NovoBloco
62  }
```

Figura 10. Função para criar blocos.

Após a inicialização da cadeia de blocos com a função `BlocoGenesis()`, é possível adicionar outros blocos que iriam interligar-se entre si através do *hash* do bloco, pois cada novo bloco irá receber o *hash* do bloco que foi adicionado anteriormente na cadeia, por esse motivo a *Blockchain* precisa de uma função para criar o bloco inicial, a `BlocoGenesis()`. Logo após criação da `BlocoGenesis()` é implementada a função `criarBloco()` na linha 53 apresentada na Figura 9, que é responsável por construir os novos blocos que viram a serem adicionados na cadeia, recebendo como parâmetro a estrutura do *payload* da interface InfoBloco.

A função `criarBloco()` fica responsável por organizar o *payload* do novo bloco da rede, de modo que o *index* do novo bloco na linha 55 (cinquenta e cinco) irá identificar qual é o *index* do último bloco da rede através da função `ultimoBloco()`, função incubida de sempre retorna o último bloco da rede *Blockchain* (Figura 10). Além disso, a função `criarBloco()` possui os parâmetros de `timestamp` e `dado`, para receber o valor do tempo de duração que o bloco foi criado e as informações contidas no bloco respectivamente. Também apresenta na linha 58 o parâmetro de `hashBlocoAnterior` que recebe a função que sempre retorna o *hash* do último bloco da cadeia (Figura 10).

```
45      private get ultimoBloco(): InfoBloco{
46          return this.#cadeiaBlock.at(-1) as InfoBloco;
47      }
48
49      private hashUltimoBloco(): string{
50          return this.ultimoBloco.header.hashBloco;
51      }
```

Figura 11. Função ultimoBloco e hashUltimoBloco

Logo depois de criar todo o *payload* do novo bloco, é feita a função para validar e gerar as informações do *header* do bloco, pois nela é armazenada a informação *hash* do bloco que será gerado e adicionado na cadeia. Para isso, foi implementada a função `validarPreBloco()`, que possui, na linha 65 (sessenta e cinco), a variável *nonce* com o valor de 0 (zero) e, na linha 66 (sessenta e seis), a variável `início` com a função de `Data()`.

```
64      validarPreBloco(bloco: InfoBloco["payload"]): {} {
65          let nonce: number = 0;
66          let inicio: number = +new Date();
67
68          while (true) {
69              const hashBloco: string = gerarHash(this.prefixo) + gerarHash(JSON.stringify(bloco));
70
71              if (hashValidado({ hash: hashBloco, dificuldade: 1, prefixo: this.prefixo })) {
72                  const final: number = +new Date();
73                  const HashReduzido: string = hashBloco;
74                  const tempoGerando: number = (final - inicio) / 1000
75                  console.log("Bloco " + bloco.indexBloco + " minerado em " +
76                              tempoGerando + ". Hash: " + HashReduzido + ". Nonce: " + nonce);
77
78                  return {
79                      blocoMinerado: {
80                          payload: { ...bloco },
81                          header: {
82                              nonce,
83                              hashBloco
84                          }
85                      }
86                  }
87              }
88          }
89      }
```

Figura 12. Função que valida o hash do bloco gerado

Na figura 11 é possível visualizar a estrutura completa da função `validarPreBloco()`, na linha 69 é definida uma constante denominada `hashBloco` que recebe a concatenação do *hash* do prefixo com o *hash* das informações que foram armazenadas no bloco, essas informações são transformadas em *hash* a partir da função `gerarHash()`, contida no arquivo infra.ts. Esse arquivo funciona como suporte para os *scripts main* do projeto, contendo funções de apoio.

infra.ts

src > infra.ts > Assinaturas

```
1   import {BinaryLike, createHash } from "crypto";
2   const fs = require('fs');
3   const path = require('path');
4   const crypto = require("crypto");
5   const chavePublicaRetoria = fs.readFileSync(path.join(__dirname, './Chaves/ChavePublicaReitoria.pem'), 'utf8');
6   const chavePublicaCurso = fs.readFileSync(path.join(__dirname, './Chaves/ChavePublicaCurso.pem'), 'utf8');
7   const chavePublicaExtensao = fs.readFileSync(path.join(__dirname, './Chaves/ChavePublicaExtensao.pem'), 'utf8');
8
9   export function gerarHash(dadosPayLoad: BinaryLike): string{
10      return createHash('sha256').update(dadosPayLoad).digest("hex")
11  }
12
13  export function hashValidado({hash, dificuldade= 1 , prefixo}: {hash: string,
14  dificuldade: number, prefixo: string}): boolean{
15      const check: string = prefixo.repeat(dificuldade);
16      const inicio: string =  gerarHash(check)
17      return hash.startsWith(inicio)
18  }
```

Figura 13. Arquivo infra.ts

A função `validarPreBloco()` possui um condicional na linha 71 da Figura 11, que verifica se a função `hashValidado()` retorna verdadeiro ou falso para o *hash* criado para o novo bloco. A função `hashValidado()` (Figura 12) recebe como parâmetros o *hash* das informações do bloco, a dificuldade padrão sendo 1 e o prefixo definido para a *Blockchain,* com isso é indicado uma variável `check` que repete o prefixo na quantidade de vez da dificuldade e depois é gerado o *hash* dessa variável (check).

Para realizar a verificação é chamada a função padrão do *TypeScript* `startsWith()`, que verifica se o inicial de algum parâmetro (lista ou *string*) é igual ao parâmetro passado para a função. Esse recurso do `startsWith()` é aplicado na função `hashValidado()`, para verificar se o *hash* do bloco criado e concatenado anteriormente, possui o mesmo prefixo padrão da *Blockchain* Privada. Caso seja igual, o bloco é validado e aceito para ser inserido na cadeia, de modo que a função `validarPreBloco()` retorna com o *header* e o *payload* completo do novo bloco (Figura 11).

```
91    verificarBloco(bloco: InfoBloco): boolean {
92        if(bloco.payload.hashBlocoAnterior != this.hashUltimoBloco()){
93            console.log("Bloco" + bloco.payload + "invalido");
94            return false;
95        }
96        const hashTeste: string = gerarHash(this.prefixo) + gerarHash(JSON.stringify(bloco.payload))
97        if(!hashValidado({hash: hashTeste, dificuldade: this.dificuldade, prefixo: this.prefixo})){
98            console.log("'Bloco " + bloco.payload + "invalido");
99            return false;
100       }
101       return true;
102   }
103
104   enviarBloco(bloco: InfoBloco): InfoBloco[]{
105       if(this.verificarBloco(bloco)){
106           this.#cadeiaBlock.push(bloco)
107           console.log("\n"+'Bloco ' + bloco.payload.indexBloco + ' foi adcionado a blockchain' + "\n");
108
109       }
110       return this.#cadeiaBlock;
111   }
```

Figura 14. Função de verificação do bloco

Logo após a geração do InfoBloco por completo, é feita uma função de verificação do bloco para examinar se está cumprindo com as regras padrão de uma *Blockchain*. A função `verificarBloco()` possui dois condicionais, um na linha 92 e outro na linha 97, de modo que, o primeiro verifica se o bloco que irá ser inserido na cadeia *Blockchain* realmente possui o `hashUltimoBloco` idêntico ao *hash* do atual último bloco da cadeia e o segundo condicional verifica se o `hashTeste` com as mesmas informações passada para o bloco que foi adicionado, retorna uma informação diferente do novo bloco. Caso o novo bloco não entre

em nenhum dos dois condicionais da função `verificarBloco()`, o novo bloco pode ser enviado e inserido na cadeia da *Blockchain* Privada. O processo de envio é feito a partir da função `enviarBloco()` na linha 104, que chama a lista #cadeiaBlock inserido o novo bloco e informando que o bloco foi adicionado (Figura 13).

4.2 Implementação do smart contract

Com a finalização do *script* da *Blockchain* Privada, contendo todas as funções necessárias para a criação e adição de blocos na cadeia seguindo toda a estruturação e validação de dados presente em uma *Blockchain*, foi iniciada a implementação do *Smart Contract*. A parte do *Smart Contract* no projeto é a responsável por criar um contrato que exige que determinada condição seja cumprida para que os novos blocos sejam adicionados à cadeia.

Essa condição se baseia nas assinaturas entre as partes responsáveis, de modo que o contrato verifica se as partes envolvidas no processo de validação realmente realizaram a assinatura para o novo bloco, caso todas as partes envolvidas assinem o contrato, é confirmado que os dados dos certificados que estão sendo adicionados na *Blockchain* estão validados.

4.2.1 Gerar Chaves

Para que a validação dos certificados inseridos na *Blockchain* aconteça através de assinatura no *Smart Contract* é preciso que os setores responsáveis por validar os novos blocos possuam suas respectivas chaves públicas e privadas. Com esse pensamento, foi criada uma função dentro do contrato para a geração das chaves dos setores que serão responsáveis por realizar as assinaturas, a função `gerarChaves()` (Figura 14).

```
25  gerarChaves(ChavePrivada: string , ChavePublica: string ) {
26      const { privateKey = ChavePrivada, publicKey = ChavePublica} = crypto.generateKeyPairSync('rsa', {
27          modulusLength: 2048,
28          publicKeyEncoding: {
29              type: 'pkcs1',
30              format: 'pem',
31            },
32            privateKeyEncoding: {
33              type: 'pkcs1',
34              format: 'pem',
35            },
36
37
38      })
39      writeFileSync('./src/Chaves/ChavePrivada.pem', privateKey)
40      writeFileSync('./src/Chaves/ChavePublica.pem', publicKey)
41  }
```

Figura 15. Função para gerar chaves pública e privada

A função `gerarChaves()` (Figura 14) é responsável por gerar arquivos contendo as chaves tanto pública como privada, para os setores. Essas chaves são criptografadas utilizando o algoritmo de RSA, responsável por realizar a criptografia assimétrica, onde o usuário utiliza a chave pública para criptografar os dados e a chave privada para descriptografar. Dentro da função `gerarChaves()` possui duas constantes, a `privateKey` e `publicKey`, elas recebem respectivamente as chaves privada e pública informadas pelo usuário. Essas constantes recebem a biblioteca *crypto* responsável por realizar a criptografia das chaves, dentre as funções da biblioteca *crypto* existe a função `generateKeyPairSync()` que auxilia na criação das chaves públicas e privadas. Nessa função é preciso especificar o tipo de criptografia, o tamanho que as chaves irão possuir e o seu tipo de formatação, como é possível visualizar na Figura 14.

4.2.2 Assinatura assimétrica e Verificação

Com a criação da função que irá gerar as chaves das respectivas coordenações é possível implementar as funções que serão responsáveis por realizar a assinatura. A função responsável por esse processo no Smart Contract é a `Assinar()`, que recebe como parâmetros de entrada o tipo de coordenação que está assinando, as informações que vão ser armazenadas no bloco e a chave pública utilizada na assinatura (Figura 15).

```
43    Assinar(tipoCoordenacao: string, info: any, chaveCoordenacao: any){
44        if (tipoCoordenacao === "Retoria"){
45            this.retoria = crypto.sign("sha256", Buffer.from(JSON.stringify(info)), {
46                key: chaveCoordenacao,
47                padding: crypto.constants.RSA_PKCS1_PSS_PADDING,
48            })
49
50            console.log("\n" + "Assinado por reitoria: " + this.retoria.toString("base64") + "\n");
51        }
52        else if (tipoCoordenacao === "Curso"){
53            this.curso = crypto.sign("sha256", Buffer.from(JSON.stringify(info)), {
54                key: chaveCoordenacao,
55                padding: crypto.constants.RSA_PKCS1_PSS_PADDING,
56            })
57
58            console.log("Assinado pela coordenação de cursos: " + this.curso.toString("base64") + "\n");
59        }
60        else {
61            this.extensao = crypto.sign("sha256", Buffer.from(JSON.stringify(info)), {
62                key: chaveCoordenacao,
63                padding: crypto.constants.RSA_PKCS1_PSS_PADDING,
64            })
65
66            console.log("Assinado pela coordenação de extensão: " +this.extensao.toString("base64") + "\n");
67        }
68    }
```

Figura 16. Função de assinatura das coordenações

Nessa função é feito um condicional para identificar qual o tipo de coordenação está assinando no momento, a reitoria, o curso ou a extensão. Após verificar qual é o tipo de

coordenação que está assinando é chamado os parâmetros da respectivas coordenação que foram definidos inicialmente na criação do *Smart Contact* (Figura 16). Parâmetros que só poderão ser acessados diretamente dentro do Smart Contract pois foram privados para que não sejam alterados fora dessa classe.

```
12  export class ValidacaoCoordenacoes{
13      private retoria: any;
14      private curso: any;
15      private extensao: any;
16      private informacoes: any;
17
18      constructor( retoria: any, curso: any, extensao: any, info: any){
19          this.retoria = retoria;
20          this.curso = curso;
21          this.extensao = extensao;
22          this.informacoes = info;
23      }
```

Figura 17. Parâmetros iniciais do Smart Contract

Esses parâmetros recebem a função `sign()` da biblioteca crypto visível na linha 45 da Figura 15, responsável por facilitar o procedimento de gerar a assinatura passando apenas o tipo do hash, as informações que serão assinadas e a respectiva chave privada da coordenação assinante. Depois que a coordenação realiza a assinatura a função imprime a informação na tela, informando que determinada coordenação realizou a assinatura e informa a assinatura gerada. Com a função de Assinatura criada foi preciso elaborar o processo de validação para verificar se a assinatura informada para o contrato realmente condiz com as assinaturas dos três setores da instituição de ensino CEULP.

```
70      verificacaoReitoria(info: any){
71          if (this.retoria!=null){
72              const verificarReitoria = crypto.verify(
73                  "sha256",
74                  Buffer.from(JSON.stringify(info)),
75                  {
76                    key: chavePublicaRetoria,
77                    padding: crypto.constants.RSA_PKCS1_PSS_PADDING,
78                  },
79                  this.retoria
80              );
81              console.log("signature verified: ", verificarReitoria);
82              return verificarReitoria;
83          }
84          else{
85              Assinaturas({retoria: this.retoria , curso: this.curso, extensao: this.extensao})
86          }
87      }
```

Figura 18. Função de verificação de assinatura da reitoria

Para isso foram criadas funções de verificação separadas para cada coordenação existente no contrato. Essas funções de verificação possuem a mesma estrutura mudando somente as variáveis e constantes para a coordenação correspondente no momento da verificação. Como é possível observar na Figura 17, a função de verificação corresponde a Reitoria, sendo chamada de `verificacaoReitoria()` que recebe como parâmetro de entrada as informações que serão adicionadas no bloco.

A função `verificarcaoReitoria()` possui um condicional na linha 71 para verificar se a reitoria já realizou a assinatura dentro do Smart Contract, se caso a reitoria já tenha realizado a assinatura ela passa pelo condicional e entra na constante `verificarReitoria` na linha 72, que recebe a função `verify()` da biblioteca crypto, auxiliando no processo de verificação apenas informando o tipo do *hash*, as informações que foram assinadas, a chave pública da coordenação de Reitoria e a sua chave privada que foi armazenada na variável reitoria. Com essas informações a função `verify()` irá se certificar de que as informações assinadas realmente vieram da coordenação de Reitoria por quem a chave pública foi emitida, retornando que a informação é verdadeira.

```
verificacaoCurso(info: any){
    if (this.curso!= null){
        const verificarCurso = crypto.verify(
            "sha256",
            Buffer.from(info),
            {
              key: chavePublicaCurso,
              padding: crypto.constants.RSA_PKCS1_PSS_PADDING,
            },
            this.curso
        );
        console.log("signature verified: ", verificarCurso);
        return verificarCurso;
    }
    else{
        Assinaturas({retoria: this.retoria , curso: this.curso, extensao: this.extensao})
    }
}

verificacaoExtensao(info: any){
    if (this.extensao!= null){
        const verificarExtesao = crypto.verify(
            "sha256",
            Buffer.from(info),
            {
              key: chavePublicaExtensao,
              padding: crypto.constants.RSA_PKCS1_PSS_PADDING,
            },
            this.extensao

        );
        console.log("signature verified: ", verificarExtesao);
        return verificarExtesao;
    }
    else{
        Assinaturas({retoria: this.retoria , curso: this.curso, extensao: this.extensao})
    }
```

Figura 19. Função de verificação de assinatura da coordenação de curso e extensão

As outras funções de verificação para a coordenação de curso e de extensão, possuem a mesma estrutura da função `verificacaoReitoria()`. Caso algumas dos setores ainda não tenham realizado a assinatura, as funções de assinatura irão entrar no condicional que não possui assinatura, e imprimir para o usuários quais setores não realizaram as assinatura dentro do *Smart Contract* (Figura 18). Desse modo, a validação dos certificados na *Blockchain* fica em estado de espera até que as demais participantes do grupo de setores realizem a validação.

4.2.3 Adição de blocos a partir do Consenso entre as Setores

Com as funções de assinatura e de verificação implementadas dentro do *Smart Contract* foi preciso criar a função responsável por executar o consenso do contrato entre as partes para que o novo bloco seja de fato adicionado na *Blockchain*. Para isso foi codificada a função `adicionarNovoBloco()`, na linha 129 há um condicional que chama todas as 3 funções de verificação dos respectivos setores, caso todas as funções de verificação retornem *true*, significa que todos os setores assinaram e as assinaturas são verídicas. Essa confirmação advinda das 3 assinaturas da coordenação é que comprova que as partes envolvidas no processo de adição de blocos na *Blockchain,* chegaram a um consenso de que as informações contidas nele são válidas.

```
128     adicionarNovoBloco(info: any): void {
129         if (this.verificacaoReitoria(info) && this.verificacaoCurso(info) && this.verificacaoExtensao(info)) {
130             let cadeiaBlock = blockchain.cadeiaBlock;
131
132             this.informacoes= JSON.stringify(info) + " Assinatura da Reitoria: " + this.retoria.toString("base64")
133             ", Assinatura do Curso: " + this.curso.toString("base64") + ", Assinatura da Extensão:"
134             + this.extensao.toString("base64")
135
136             const bloco = blockchain.criarBloco(this.informacoes);
137             const mineInfo = blockchain.validarPreBloco(bloco);
138             cadeiaBlock = blockchain.enviarBloco(mineInfo.blocoMinerado);
139
140             console.log("_____ BLOCK_____");
141             console.log(cadeiaBlock)
142         }
143         else{
144             console.log("Não é possivel adicionar novo bloco, pois as informações de assinatura não são validas");
145         }
```

Figura 20. Consenso entre os setores

Após passar pelo condicional, a função que adiciona um novo bloco cria variáveis chamando os parâmetros e funções existentes dentro do script da *Blockchain.* De modo que a variável `cadeiaBlock` na linha 130 recebe a lista existente dentro da *Blockchain,* a variável `bloco` na linha 136 recebe a função `criarBloco()` com o parâmetro inicial do *Smart Contract*, denominado `informacoes`. O parâmetro `informacoes` na linha 132 consiste em juntar todas as informações que serão guardadas dentro do novo bloco da *Blockchain*, com a estrutura e informações dos certificados que serão validados e as 3 assinaturas dos respectivos

setores da instituição CEULP, sendo que cada setor e coordenação de curso irá possuir a sua própria chave de validação. A imagem abaixo (Figura 20) demonstra as etapas de funcionamento da *Blockchain* juntamente com o *Smart Contract*.

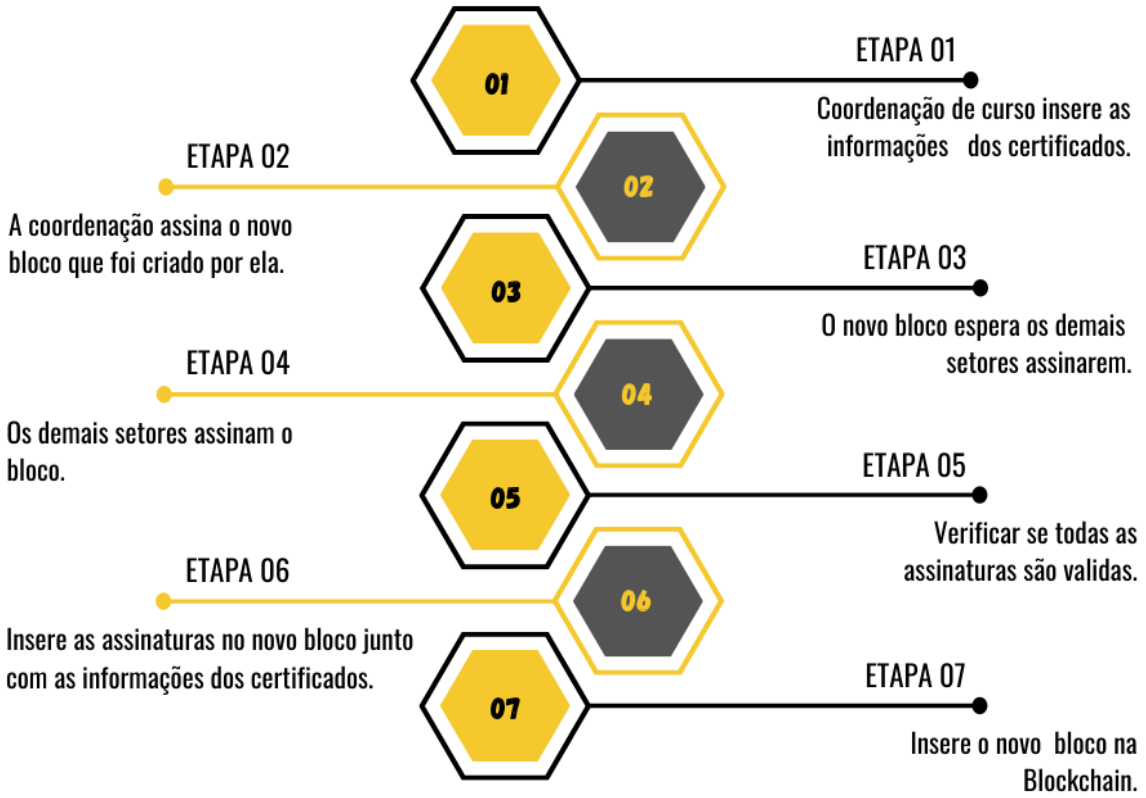

Figura 21. Etapas do funcionamento dos smart contracts

A Figura 20 demonstra as etapas de funcionamento de todo o *Smart Contract* e o consenso entre os setores, desde da etapa 1 que consiste na inserção dos dados dos certificados pela coordenação de curso, até a etapa final com o bloco já adicionado na cadeia *Blockchain*. Ao aplicar essas etapas nos certificados emitidos através da plataforma SIG Extensão, é possível garantir a validação e veracidade das informações contidas nos certificados. Para demonstrar o funcionamento do projeto completo é preciso criar um arquivo index.ts chamando todos os arquivos necessários para execução do *Smart Contract* (Figura 21).

```
1  import { ValidacaoCoordenacoes } from "./smartContract";
2  const fs = require('fs');
3  const path = require('path');
4  const privateKey = fs.readFileSync(path.join(__dirname, './Chaves/ChavePrivadaReitoria.pem'), 'utf8');
5  const privateCurso = fs.readFileSync(path.join(__dirname, './Chaves/ChavePrivadaCurso.pem'), 'utf8');
6  const privateExte = fs.readFileSync(path.join(__dirname, './Chaves/ChavePrivadaExtensao.pem'), 'utf8');
7  const SmartContract = new ValidacaoCoordenacoes();
8  SmartContract.gerarChaves("ReitoriaPrivada", "ReitoriaPublica")
9  SmartContract.gerarChaves("CursoPrivada", "CursoPublica")
10 SmartContract.gerarChaves("ExtensãoPrivada", "ExtensãoPublica")
11
12 const informacaoCertificados1 = {"Nome do Participante": "Natanna Rocha Santos",
13 "Nome do Evento": "22º ENCOINFO - CONGRESSO DE COMPUTAÇÃO E TECNOLOGIA DA INFORMAÇÃO",
14 "Curso Responsaveis": "Ciência da Computação, Engenharia de Software e Sistema de Informação",
15 "Data": "03 e 06 de novembro de 2020",
16 "Carga Horaria": "8.0 Horas",
17 "Tipo de Participação": "Participante",
18 "Atividade": "Análise de Dados com o Opendistro for Elastisearch",
19 "Codigo de Registro": "840fbebe-c87a-469d-bca0-f93ce17f8d09"
20 }
21 const assinaturaRetoria = SmartContract.Assinar("Reitoria", informacaoCertificados1, privateKey)
22 const assinaturaCurso = SmartContract.Assinar("Curso", informacaoCertificados1, privateCurso)
23 const assinaturaExtensao = SmartContract.Assinar("Extensao", informacaoCertificados1, privateExte)
24 SmartContract.adicionarNovoBloco(informacaoCertificados1)
```

Figura 22. Criando novo bloco

A Figura 21 mostra os dados e informações necessárias para executar o *Smart Contract*, para isso é preciso importar a classe que contém as funções do contrato inteligente, na linha 1 a classe `ValidacaoCoordenacoes` é chamada do arquivo smartContrast.ts. No arquivo index é criado na linha 7 (sete) a constante SmartContrast, que irá receber sempre um novo objeto da classe `Validaçãocoordenaçoes,` gerando assim sempre novos parâmetros para cada bloco que for criado. Ao realizar a assinatura da Reitoria na linhas 21 e chamando a função `Assassina(),` que tem como parâmetro definido como padrão a string “Reitoria” para indicar o tipo de coordenação que está assinando, os demais parâmetros que a função recebe é o dicionário com as informações do certificado e a chave privada da coordenação que está assinando.

```
    header: {
      nonce: 0,
      hashBloco: 'bd2c3dca84435dea1d240448f5ebb9503657162f43e009288e472f3c149f684aaab9fae4b89612bf6e3db3c8562e839c41e0e0091304a22b2fef6b63e3
1a3131'
    },
    payload: {
      indexBloco: 0,
      timestamp: 1654829814584,
      dado: 'Bloco Genesis da cadeia de blocos de validação de certificados',
      hashBlocoAnterior: ''
    }
  },
  {
    payload: {
      indexBloco: 1,
      timestamp: 1654829815499,
      dado: '{"Nome do Participante":"Natanna Rocha Santos","Nome do Evento":"22º ENCOINFO - CONGRESSO DE COMPUTAÇÃO E TECNOLOGIA DA INFORMA
ÇÃO","Curso Responsaveis":"Ciência da Computação, Engenharia de Software e Sistema de Informação","Data":"03 e 06 de novembro de 2020","Carg
a Horaria":"8.0 Horas","Tipo de Participação":"Participante","Atividade":"Análise de Dados com o Opendistro for Elastisearch","Codigo de Reg
istro":"840fbebe-c87a-469d-bca0-f93ce17f8d09"} Assinatura da Reitoria: cq5avyeLz/MRiBu9KXz63Dh9IYG+ftvnI3bq0x/RHGG+yQuJSz8l9uOD+PGIfPYbNilfe
MYFHVdO6Hh0SaQtvaj3+wxzzeNZ7ew3pSObSRSjfar8oMKhukIH1RHA5Tpt4tsdTMlxKuAq22P9qCm184WrmUzPUur0KjDC8tgNN7EMip2Rk3S7p6yXue50uIJ/b8zWo6y0irdOgbiam
zMA8p++zgjmSjkesZP3Bu+2754RtiIpXqIj6N6PLWhMqKc2pxqVoPPvQPPsufQ1UBwRMQ8PreP7GPUN99PdfGZPd3sZ6jnoS1SzZKaCx0vp8Xagx2CEzmPtv0kvpS2fZsqNCw==, Ass
inatura do Curso: ufAIZOhfsXFS5ogvvWdfoQ2xTfLT837xY01GiE9BpZDI5O9RSLYYh+j+U8R3GtKa45z9lyarJu0TQCk4bHZXE9ucztlMT3ZBkRSDtXz5GPyaQtiB4ThTHU/lzK
8DNwhqwjb2xtqaRJyH+i1LKdr7rn/RW6Bz/dpJYjs5MSiF411R2a+YcGuQggiUhMfLL1i0AAjA4DoLZmWGU3ZOMHcLazngbop879iJxTr0PElQmOrvdGW0yA72hPkkMzZXkIUqUogWFP
YRYVpqfl5ORj+ApIrFM5JKIDzMaeWb5JCZFEI6e/XWvBsPvkUaEiwP6U9RQ9wN5FXaqSgHDCPmMhkzBA==, Assinatura da Extensão:w1jhjGa656VGb+3hlAG08c6lU0/z49FbV
Vjd37Ico8/KLAvOwhXClQ/0ouOAlPQDihAD2vvDvHvEnFbkKEhiMhPKqpT9grIHuGmdpFlbz8oAwueav9Mfk2HGva0nBR+5vapHH0QMHBjyFa001lQTchbBDHydVfyYfCcu8Q0lm3HD6
aWyUZ1j7gy8OhLs1fg2cVidoPdPH4upPSeTrNzSgE74MioVupTFre0Vo1T+8XOiunujGlNWYADdbeZMTBDchhSrOU8/fkJlarfIRdhdhPTCMo8NkLicz25q+Y2EsFY+f1rgm/VvrwtJW
puOY6OiovkQa4vkbYR3h8rakkZvQA==',
      hashBlocoAnterior: 'bd2c3dca84435dea1d240448f5ebb9503657162f43e009288e472f3c149f684aaab9fae4b89612bf6e3db3c8562e839c41e0e0091304a22b2f
ef6b63e31a3131'
    },
    header: {
      nonce: 0,
      hashBloco: 'bd2c3dca84435dea1d240448f5ebb9503657162f43e009288e472f3c149f684a33eb99db9a7b330aaf76dfd485aa0d36de6cb059dbc8e1ff5b03dfc8c5
```


Figura 23. Blockchain em execução

Ao executar o projeto é possível observar no terminal todos os blocos criados como teste no index (Figura 22), de modo que o primeiro bloco da cadeia é o bloco gênesis criado a partir da função `BlocoGenesis()` existente no *script* do blockchain.ts. O segundo bloco que foi inserido na cadeia possui a informação dos certificados das Figuras 3 e 4, essas informações que foram retiradas do certificados e foram armazenadas no bloco consistem no nome do participante, o evento, cursos responsáveis, data de acontecimento, a carga horária, o tipo de participação, a atividade executada e o código de registro do certificado na plataforma SIG extensão.

```
Assinado por reitoria: Ox3W13TI1rBZirJhWZQoMlAhU1fvaD4ybF/0tFsus7KPUAnTYBe0+MQApzQyiKh70iWOnE5TcaoapgHBV4OHDeQjk8PmIW0EUiBkdNXFrKe3B86AeY3eN
fccxsldS0iHQAJEhoMeyI7V0BD3a/a+SDTagCDb/NvySrAA3ZrB6d3BGahSYNpvKeNvjWd8R/EC41VqnRNRd6GFhSxmkFke3ufqC0Na0YYZS55Akyn/BHjJDwKjovW7/5J5M6n69Rah2
P+/dk9Oa4jQa1KWvXPBWlwW823fUtxqBlPqyZyNV7/OI1+kFpQKUarJ5RRFiAYLqIpIp75KF0fwV/u8ssboaQ==

Assinado pela coordenação de cursos: XAAbbVCVbuGK0espeOxTfjiEDFgYidvhVPqKqwZKSV/WfCBL3Ld0rShto74B7kc5xipC29MSDHeQggwXB0yP7TvJVfGxWd2mkS8q9Uo
jv3M/zOlnNPzfCX2UGMLXzTM+9LVdecAIMUXWMKcChlfd97bgiSIHinRS4iY2E4HMQkznfqcxVgugkZu50/Pyujs51n/CVjYakv/aJBNna+C2ToG5wQBNbRBPkdX6NZ6N1mv3r0Copoa
HTqkeuxTgY7B6Y7N7Joi7pnTDMn7ZgXJZn7bPfNY47J7q/uh3ElZeKLBJBsCaXeD7ecWatP8efGLgfCEsZj2BQHDzHBtjucML+w==

Assinado pela coordenação de extensão: O6hR4asaoMHReeImq2fgpDr6jHWwZhV6ERoyYIXUzhQh1BwPPclwN8LoYoN/vUqJVmpx+A5sbySKX8BY7X168rRiYDAavyOM2gbHx
FiJMwPVFFzZrxk0eS0NbB0NZbhdpidqqM3nxLtc/Hh2hoIRhuZDXiOT6x7RsivDNigzAPcrqkeFR59JjATNddKl5lt/mC6nYt3V1cV7qxZuSvCmTyiv77s3o42l4yJ4RCHD58aeSjvus
iBJYiKdAFPoEN1Or1YA9QgfyUz/dkiYHWrwmuLUPZnHh5SLt24lE0H752Is7/ReEmRVi/8bN+4VuNk2z8WlgKBoHu9cbTOsuMJRoA==

signature verified:  true
signature verified:  true
signature verified:  true

Bloco 3 criado: {"indexBloco":3,"timestamp":1654829815668,"dado":"{\"Nome do Participante\":\"Natanna Rocha Santos\",\"Nome do Evento\":\"FL
ASHCLIP\",\"Curso Responsaveis\":\"Ciência da Computação, Engenharia de Software e Sistema de Informação\",\"Data\":\"\",\"Carga Horaria\":\
"4.0 Horas\",\"Tipo de Participação\":\"AUTOR DE ARTIGO\",\"Atividade\":\"MINICURSO SMALLTALK\",\"Codigo de Registro\":\"e6e9fe14-0c4a-41ff-
b76e-4a90f8c3946b\"} Assinatura da Reitoria: Ox3W13TI1rBZirJhWZQoMlAhU1fvaD4ybF/0tFsus7KPUAnTYBe0+MQApzQyiKh70iWOnE5TcaoapgHBV4OHDeQjk8PmIW0
EUiBkdNXFrKe3B86AeY3eNfccxsldS0iHQAJEhoMeyI7V0BD3a/a+SDTagCDb/NvySrAA3ZrB6d3BGahSYNpvKeNvjWd8R/EC41VqnRNRd6GFhSxmkFke3ufqC0Na0YYZS55Akyn/BHj
JDwKjovW7/5J5M6n69Rah2P+/dk9Oa4jQa1KWvXPBWlwW823fUtxqBlPqyZyNV7/OI1+kFpQKUarJ5RRFiAYLqIpIp75KF0fwV/u8ssboaQ==, Assinatura do Curso: XAAbbVCV
buGK0espeOxTfjiEDFgYidvhVPqKqwZKSV/WfCBL3Ld0rShto74B7kc5xipC29MSDHeQggwXB0yP7TvJVfGxWd2mkS8q9Uojv3M/zOlnNPzfCX2UGMLXzTM+9LVdecAIMUXWMKcChlfd
97bgiSIHinRS4iY2E4HMQkznfqcxVgugkZu50/Pyujs51n/CVjYakv/aJBNna+C2ToG5wQBNbRBPkdX6NZ6N1mv3r0CopoaHTqkeuxTgY7B6Y7N7Joi7pnTDMn7ZgXJZn7bPfNY47J7q
/uh3ElZeKLBJBsCaXeD7ecWatP8efGLgfCEsZj2BQHDzHBtjucML+w==, Assinatura da Extensão:O6hR4asaoMHReeImq2fgpDr6jHWwZhV6ERoyYIXUzhQh1BwPPclwN8LoYoN
/vUqJVmpx+A5sbySKX8BY7X168rRiYDAavyOM2gbHxFiJMwPVFFzZrxk0eS0NbB0NZbhdpidqqM3nxLtc/Hh2hoIRhuZDXiOT6x7RsivDNigzAPcrqkeFR59JjATNddKl5lt/mC6nYt3
V1cV7qxZuSvCmTyiv77s3o42l4yJ4RCHD58aeSjvusiBJYiKdAFPoEN1Or1YA9QgfyUz/dkiYHWrwmuLUPZnHh5SLt24lE0H752Is7/ReEmRVi/8bN+4VuNk2z8WlgKBoHu9cbTOsuMJ
RoA==","hashBlocoAnterior":"bd2c3dca84435dea1d240448f5ebb9503657162f43e009288e472f3c149f684a494b75989b92544682eda9973b57cf0b4663dd3ac787e5aa
5e1e1c10e2c3cea2"}
```


Figura 24. Assinaturas e verificação do bloco no terminal

A cada novo bloco inserido é mostrada uma mensagem no terminal, informando qual coordenação já assinou e sua assinatura. Além disso, também informa se a assinatura dos setores é válida, em seguida indica que o novo bloco foi criado e está pronto para ser adicionado na *Blockchain*.

5 CONSIDERAÇÕES FINAIS

O presente trabalho teve como resultado o desenvolvimento de uma *Blockchain* e de um *Smart Contract* para garantir a validação e veracidade das informações dos certificados disponíveis na plataforma SIG Extensão da instituição de ensino CEULP. A partir do desenvolvimento da *Blockchain* para o presente trabalho foi possível analisar como era feita a atual forma de emissão e validação dos certificados da plataforma SIG Extensão e identificar quais informações seriam necessárias para serem armazenadas e validadas no momento de adicionar os blocos na cadeia *Blockchain*. Com essas informações foi possível alcançar os objetivos de definir a estrutura das informações que fariam parte dos blocos que seriam armazenados na cadeia e o tipo de criptografia que seria aplicada no *Smart Contract* para garantir a segurança e confiabilidade das informações dos certificados. Com esses passos realizados foi possível concluir a implementação da *Blockchain* e do *Smart Contract*, garantido desse modo que é possível validar os certificados do SIG Extensão utilizando *Blockchain*.

Porém, foram encontradas algumas dificuldades no decorrer da implementação do trabalho, pois algumas tecnologias como *Hyper Ledger Sawtooth* que são sugeridas para a criação de uma *Blockchain* Privada com *Smarts Contracts*, não possuíam versões da linguagem node.js atualizadas, apresentando suporte apenas para a versão node 8.0 que não possui mais suporte por ser um versão muito antiga. Com isso foi preciso pensar em outras maneiras de realizar a implementação do projeto optando para a utilização da linguagem *TypeScript*, sendo necessário aprender seus comandos, sintaxes e formas de aplicação, para que pudesse ser possível iniciar a construção das etapas de implementação da *Blockchain* e do *Smart Contract*.

Para a realização do presente trabalho foi preciso pesquisar e aprender sobre os diferentes tipos de *Blockchain* e qual se encaixava melhor no contexto do projeto, como a diferenças entre as *Blockchain,* onde a Pública permite que qualquer pessoa possa participar sendo ativa na rede e também oferecer como troca por essa participação um tipo de recurso gerado através da mineração e a *Blockchain* Privada, que é preciso obter a permissão da rede para participar ativamente na cadeia *Blockchain*, além de não possui um tipo de mineração

para gerar o tipo de recurso de troca por participação. Também foi preciso compreender os tipos de consenso existentes e qual se aplicaria melhor para realizar a validação entre os setores na *Blockchain* Privada, sendo aplicado um consenso com a mesma ideia do *Proof of Authority* (Prova de Autoridade), pois é possível definir um conjunto específico de validadores para a rede de blocos, sendo esse conjunto de autoridades os setores responsáveis, além disso é preciso a aprovação de todas a autoridades para que uma ação dentro da rede seja validada.

Para Trabalhos futuros, é sugerido que a *Blockchain* para validação dos certificados da instituição de ensino CEULP possua uma interface visual e funcional, com uma aba que possibilite os setores responsáveis (Reitoria, Curso e Extensão) a gerarem as chaves para realizar a assinatura e validação dos certificados, além de uma página com um formulário para auxiliar na adição das informações dos certificados a serem inseridas em um novo bloco para validação e uma pagina que possa listar e pesquisar todos os blocos existente na cadeia *Blockchain*. Também seria interessante uma página que possibilite os setores verificarem os status dos blocos a serem validados, facilitando a visualização de quais blocos estão esperando para serem assinados e que o sistema possua o conceito de sistemas distribuídos, possibilitando a comunicação entre os setores, trocando informações em tempo real, auxiliando no momento de realizar o consenso entre os setores responsáveis.

REFERÊNCIAS

ALECRIM, Emerson. **CRIPTOGRAFIA**. Disponível em: <http://www.infowester.com/criptografia.php>. Acesso em: 12 out. 2021.

ALECRIM, Emerson. **CERTIFICAÇÃO DIGITAL: O QUE É E PARA QUE SERVE?**. Disponível em: <https://www.infowester.com/assincertdigital.php>. Acesso em: 07 out. 2021.

ABREU, Antônio Welligton dos Santos. **UMA ABORDAGEM BASEADA EM BLOCKCHAIN PARA ARMAZENAMENTO E CONTROLE DE ACESSO AOS DADOS DE CERTIFICADOS DE ALUNOS DO ENSINO SUPERIOR**. 2020. 146 f. Dissertação (mestrado) – Universidade Federal do Ceará, Campus de Quixadá, Programa de Pós-Graduação em Computação, Quixadá, 2020.

ARAÚJO, Wagner Junqueira de; VIEIRA, Renato Melo. ASSINATURA DE DOCUMENTOS ELETRÔNICOS UTILIZANDO CERTIFICADOS DIGITAIS. **Biblionline**, João Pessoa, v. 8, p. 290-302, 2012. Disponível em: <https://periodicos.ufpb.br/ojs/index.php/biblio/article/view/14204>. Acesso em: 29, out. 2021.

ARAÚJO, W. J. de; VIEIRA, R. M. Assinatura de documentos eletrônicos utilizando certificados digitais: estudo de caso de assinaturas digitais aplicadas em atas de reuniões. **Múltiplos Olhares em Ciência da Informação**, *[S. l.]*, v. 3, n. 2, 2014. Disponível em: <https://periodicos.ufmg.br/index.php/moci/article/view/17435>. Acesso em: 8 abr. 2022.

BIERMAN, Gavin; ABADI, Martín; TORGERSEN, Mads. Understanding TypeScript. **Ecoop 2014 – Object-Oriented Programming**, [S.L.], p. 257-281, 2014. Springer Berlin Heidelberg. http://dx.doi.org/10.1007/978-3-662-44202-9_11.

BOESING, J.; HEINECK, T.; CROTTI DA ROSA, A. M.; ROSSI, L. L. EMISSÃO DE CERTIFICADOS ELETRÔNICOS NOS EVENTOS DO INSTITUTO FEDERAL CATARINENSE CAMPUS VIDEIRA. Extensão Tecnológica: **Revista de Extensão do Instituto Federal Catarinense**, Blumenau, n. 3, p. 7–12, 2015. Disponível em: <https://publicacoes.ifc.edu.br/index.php/RevExt/article/view/88>. Acesso em: 8 abr. 2022.

BUTERIN, V.. **A NEXT-GENERATION SMART CONTRACT AND DECENTRALIZED APPLICATION PLATFORM**. Github, 2019. Disponível em: <https://github.com/ethereum/wiki/wiki/White-Paper>. Acesso em: 28 de mar. 2019.

CARVALHO, Rogério Atem de. **CARTEIRA DE CURSOS BASEADA EM GREVETECNOLOGIA BLOCKCHAIN.** Brasil: Escola Nacional de Administração Pública (Enap), 2019.

CHENG, Jiin-Chiou; LEE, Narn-Yih; CHI, Chien; CHEN, Yi-Hua. BLOCKCHAIN AND SMART CONTRACT FOR DIGITAL CERTIFICATE. **IEEE International Conference on Applied System Invention (ICASI)**, p. 1046-1051, 2018, doi: 10.1109/ICASI.2018.8394455.

COULOURIS, George; DOLLIMORE, Jean; KINDBERG, Tim; BLAIR, Gordon. **SISTEMAS DISTRIBUÍDOS: CONCEITOS E PROJETOS.** 5. ed. Porto Alegre: Bookman Editora Ltda, 2013. 1055 p. ISBN 978-85-8260-053-5.

DIVINO, Sthéfano Bruno Santos. SMART CONTRACTS: CONCEITOS, LIMITAÇÕES, APLICABILIDADE E DESAFIOS. **Revista Jurídica Luso-Brasileira**. 2018. Disponível em: <https://blook.pt/publications/journal/b0c0c138247e/#main>. Acesso em: 28 nov. 2021. p.2776.

FERREIRA, D.; CAMARGO, H.; ALMEIDA, R.. SISTEMA PARA REGISTRO E AUTENTICAÇÃO DE CERTIFICADOS EM BLOCKCHAIN. **Simpósio de Pesquisa e Inovação**, Brasil, ago. 2019. Disponível em: <http://comunica.barbacena.ifsudestemg.edu.br/index.php/spi/XSPI/paper/view/467/222>. Data de acesso: 01 Dez. 2021.

FREITAS, Christiana Soares de ; VERONESE, Alexandre. SEGREDO E DEMOCRACIA: CERTIFICAÇÃO DIGITAL E SOFTWARE LIVRE. **Revista Informática Pública**. Belo Horizonte, v. 8, n. 02, p. 9-26, 2007.

FRIEDRICH, Diego Mostardeiro. MEDINA, Roseclea Duarte. **CERTIFICAÇÃO DIGITAL ACADÊMICA**: IMPLANTAÇÃO DO SISTEMA DE GERENCIAMENTO DE CERTIFICADOS DIGITAIS ICPEDU NA UFSM. Disponível em: <http://www.cinted.ufrgs.br/ciclo10/artigos/2iDiego.pdf>. Acesso em: 08 nov. 2021.

GANDINI, João Agnaldo Donizeti; SALOMÃO, Diana Paola da Silva; JACOB, Cristiane. **A VALIDADE JURÍDICA DOS DOCUMENTOS DIGITAIS. Revista dos Tribunais**, São Paulo, v.91, n.805, p. 83-98, nov. 2002. Disponível em: <https://ambitojuridico.com.br/edicoes/revista-9/a-validade-juridica-dos-documentos-digitais/>. Acesso em: 02 out. 2021.

GANDINI, João Agnaldo Donizeti; SALOMÃO, Diana Paola da Silva; JACOB, Cristiane. **A segurança dos documentos digitais**. Disponível em: <https://jus.com.br/artigos/2677/a-seguranca-dos-documentos-digitais>. Acesso em: 08 mar. 2022.

GATES, Mark. **Blockchain**: Ultimate Guide to Understanding Blockchain, Bitcoin, Cryptocurrencies, Smart Contracts and the Future of Money. Breinigsville, Pensilvânia: Createspace Independent Publishing Platform. 2017. 126 p.

GREVE, F.; SAMPAIO, L.; ABIJAUDE, J.; COUTINHO, A. A.; BRITO, I.; QUEIROZ, S. BLOCKCHAIN E A REVOLUÇÃO DO CONSENSO SOB DEMANDA. In: Minicursos do Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos. Porto Alegre, RS: Sociedade Brasileira de Computação, 2018.

JIMENE, Camilla do Vale; BLUM, Renato Müller da Silva Opice. **O VALOR PROBATÓRIO DO DOCUMENTO ELETRÔNICO**. São Paulo, 2010.

LIMA, Anderson Ferreira de. **Sistema de gerenciamento de certificados e declarações para UTFPR-GP**. 2019. 38 f. Trabalho de Conclusão de Curso (Graduação) – Universidade Tecnológica Federal do Paraná, Guarapuava, 2019.

LUCENA, Antônio Unias de. HENRIQUES, Marco Aurélio Amaral. ESTUDO DE ARQUITETURAS DOS BLOCKCHAINS DE BITCOIN E ETHEREUM. In: **IX Encontro de Alunos e Docentes do DCA/FEEC/UNICAMP (EADCA)**. Ed. FEEC, 2016. Disponível em:

<http://www.fee.unicamp.br/sites/default/files/departamentos/dca/eadca/eadcaix/artigos/lucena_henriques .pdf> . Acesso em: 07 set. 2017.

MARTINELLI, T.; PINTO, G. S. BLOCKCHAIN: comparação evolutiva utilizando Bitcoin e Ethereum. **Revista Interface Tecnológica**, *[S. l.]*, v. 16, n. 1, p. 146-157, 2019. Disponível em: <https://revista.fatectq.edu.br/index.php/interfacetecnologica/article/view/570>. Acesso em: 27 out. 2021.

MENKE, Fabiano. Assinaturas Digitais, certificados digitais, infra-estrutura de chaves públicas brasileira e a ICP alemã. **Revista de Direito do Consumidor**, v. 12, n. 48, p. 17, 2003.

MORENO, Douglas Aquino; SANTOS, Natanna Rocha; SILVA, Stefan Lucas Aquino; ARAUJO, Fábio Castro. PROPOSTA DE UMA CARTEIRA DE VACINAÇÃO DIGITAL COM BLOCKCHAIN. In: CONGRESSO DE COMPUTAÇÃO E TECNOLOGIAS DA INFORMAÇÃO, 23., 2021, Palmas. **Anais [...]** . Palmas: Ceulp/Ulbra, 2021. p. 169-180.

MOUGAYAR, William. **BLOCKCHAIN PARA NEGÓCIOS: PROMESSA**, PRÁTICA E APLICAÇÕES DA NOVA TECNOLOGIA DA INTERNET. Rio de Janeiro: Alta Books, 2017. 224 p.

MOURA, Luzia Menegotto Frick de; BRAUNER, Daniela Francisco; JANISSEK-MUNIZ, Raquel. BLOCKCHAIN E A PERSPECTIVA TECNOLÓGICA PARA A ADMINISTRAÇÃO PÚBLICA: UMA REVISÃO SISTEMÁTICA. **Revista de Administração Contemporânea**, [S.L.], v. 24, n. 3, p. 259-274, jun. 2020. FapUNIFESP (SciELO). http://dx.doi.org/10.1590/1982-7849rac2020190171.

NOFER, Michael; GOMBER, Peter; HINZ, Oliver; SCHIERECK, Dirk. BLOCKCHAIN. **BUSINESS & INFORMATION SYSTEMS ENGINEERING**, [S.L.], v. 59, n. 3, p. 183-187, 20 mar. 2017. Springer Science and Business Media LLC. http://dx.doi.org/10.1007/s12599-017-0467-3.

NOLETO, Cairo.Typescript: o que é, principais conceitos e porquê usar!, **betrybe**, 2020. Disponível em: <https://blog.betrybe.com/desenvolvimento-web/typescript/>. Acesso em: 07, maio. 2022.

OHTOSHI, Paulo Hideo. **O comportamento informacional: estudo com especialistas em segurança da informação e criptografia integrantes da RENASIC/COMSIC**. 2013. 156 f., il. Dissertação (Mestrado em Ciência da Informação)—Universidade de Brasília, Brasília, 2013.

OTTONI, Márcia Benedicto. **CERTIFICAÇÃO DIGITAL E SEGURANÇA.** São Paulo: Certisign, 2005.

PAVANATI, Andre; SOUZA, Jacqueline Maria Bastos de; NUNES, Heliete; COELHO, Adriano. DOCUMENTOS DIGITAIS NA GESTÃO UNIVERSITÁRIA: O CERTIFICADO DIGITAL COMO GARANTIA DE SEGURANÇA, ORIGEM E INTEGRIDADE. **XVII Colóquio Internacional de Gestão Universitária**, Mar del Plata, p. 1-10, 23 nov. 2017. Disponível em: <https://repositorio.ufsc.br/xmlui/bitstream/handle/123456789/181080/101_00241.pdf?sequence=1>. Acesso em: 23 nov. 2021.

PEREIRA, WINICIUS. **PROTOCOLO PARA EMISSÃO DE ASSINATURA DIGITAL UTILIZANDO COMPARTILHAMENTO DE SEGREDO**. 01/08/2011 109 f. Mestrado em CIÊNCIA DA COMPUTAÇÃO Instituição de Ensino: UNIVERSIDADE FEDERAL DE UBERLÂNDIA, UBERLÂNDIA Biblioteca Depositária: Biblioteca da Universidade Federal de Uberlândia.

PIERRO, Massimo di. **WHAT IS THE BLOCKCHAIN? COMPUTING IN SCIENCE & ENGINEERING,** [S.L.], v. 19, n. 5, p. 92-95, 2017. Institute of Electrical and Electronics Engineers (IEEE). http://dx.doi.org/10.1109/mcse.2017.3421554.

SCHUETTEL, Patrick. **The Concise Fintech Compendium**. Fribourg: School of Management Fribourg/Switzerland, 2017.

SILVA JÚNIOR, Sérgio Mota da. **CERTIFICAÇÃO DIGITAL A IMPORTÂNCIA PARA ÓRGÃO PÚBLICO**. 2013. 37 f. TCC (Graduação) - Curso de Lato Sensu em Rede de Computadores, O Centro Universitário de Brasília (Uniceub/Icpd), Brasília, 2013.

SOUZA, Emerson de Brito; CARNEIRO, Elisângela; COUTINHO, Antonio. GERAÇÃO E VALIDAÇÃO DE DIPLOMAS E CERTIFICADOS UTILIZANDO BLOCKCHAIN PÚBLICA. *In*: WORKSHOP EM BLOCKCHAIN: TEORIA, TECNOLOGIAS E APLICAÇÕES (WBLOCKCHAIN), 4. , 2021, Evento Online. **Anais** [...]. Porto Alegre: Sociedade Brasileira de Computação, 2021. p. 54-59.

SWAN, Melanie. BLOCKCHAIN: BLUEPRINT FOR A NEW ECONOMY. Sebastopol, California: **O'Reilly Media Inc**., 2015. 149 p.

TANENBAUM, Andrew S.; VAN STEEN, Maarten. **SISTEMAS DISTRIBUÍDOS: princípios e paradigmas.** 2. ed. São Paulo: Pearson Prentice Hall, 2007. 416 p.

TURKANOVIC, Muhamed; HOLBL, Marko; KOSIC, Kristjan; HERICKO, Marjan; KAMISALIC, Aida. EduCTX**: A BLOCKCHAIN-BASED HIGHER EDUCATION CREDIT PLATFORM.** Ieee Access, [S.L.], v. 6, p. 5112-5127, 2018. Institute of Electrical and Electronics Engineers (IEEE). http://dx.doi.org/10.1109/access.2018.2789929.

VALE, Sávio. **SOLIDITY: A LINGUAGEM DE PROGRAMAÇÃO PARA CRIAR OS SMART CONTRACTS NA ETHEREUM**. Disponível em: <https://www.voitto.com.br/blog/artigo/linguagem-de-programacao-solidity>. Acesso em: 03 dez. 2021.

VIDAL, Fernando Richter. **ANÁLISE E APLICAÇÃO DA TECNOLOGIA BLOCKCHAIN NA GESTÃO DE DIPLOMAS DO ENSINO SUPERIOR.** 2020. 139 f., Faculdade de Ciência e Tecnologia, Fernando Pessoa, 2020.

XU, X.; WEBER, I.; STAPLES, M. BLOCKCHAIN IN SOFTWARE ARCHITECTURE. In: . **Architecture for Blockchain Applications**. Cham: Springer International Publishing, 2019. p. 83–92. ISBN 978-3-030-03035-3. Disponível em: < https://doi.org/10.1007/978-3-030-03035-3_5>. Acesso em: 20 nov. 2021.

ZHENG, Zibin et al. Blockchain challenges and opportunities: A survey. **International Journal of Web and Grid Services**, v. 14, n. 4, p. 352-375, 2018.