Blockchain Frontier Technology (B-Front)
Vol. 3 No. 1 July 2023

P-ISSN: 2808-0831
E-ISSN: 2808-0009

Smart Contracts on the Blockchain: Design, Use Cases, and Prospects

Mohamed Imran Zacky[1], Syahri Helmi[2], Isadora Della Cella[2]

Electrical and Electronics Engineering Department, PETRONAS University of Technology[1,2,3]
Malaysia
e-mail: mohamedimran334@gmail.com, syahri_helmi123@yahoo.com, isadora_dellac@gmail.com

Author Notification
03 June 2023
Final Revised
15 June 2023
Published
01 July 2023

Zacky, M. I., Helmi, S., & Della Cella, I. (2022). Smart Contracts on the Blockchain: Design, Use Cases, and Prospects. Blockchain Frontier Technology, 3(1).
DOI: https://doi.org/10.34306/bfront.v3i1.363

Abstract

Since the invention of Bitcoin, blockchain technology has expanded to include more than just digital money. Because it was quickly developed and widely adopted, the blockchain allows users to perform secure transactions in an unreliable environment. One of the most critical components of real-world blockchain applications is the smart contract. In addition to being integrated into well-known blockchain-based development platforms like Ethereum and Hyperledger, smart contracts have in the digital economy and in the intelligent industries, there are a variety of potential application situations, including, among others, management, healthcare, the Internet of Things, and financial services. This paper's main objective is to present a comprehensive analysis of the research on smart contracts, including information on their inner workings, basic architecture, use cases, challenges, most recent developments, and possible future paths. Though they are still in their infancy, smart contracts have significant technical difficulties like security and privacy concerns that require more investigation. Before proposing a study of a framework based on a for smart contracts revolutionary architecture with six layers, the technique first described the workings and popular platforms of blockchain-enabled smart contracts. Second, a list of the current state of the research is provided, together with the technical and legal difficulties. In the third place, we provided several typical application examples. We talked about the several directions that smart contracts could take at the end. The purpose of this document is to serve as a useful guide and source for future research projects.

Keywords: *Smart contracts, a six-layer architecture, a parallel blockchain, blockchain technology*

1. Introduction

Szabo, a computer scientist, and cryptographer, first used the term "smart contract" in the middle of the 1990s [1]. Szabo described it as a set of digitally encoded promises that also include rules that govern how the parties carry out their obligations [2]. In his well-known illustration, Szabo compared smart contracts to vending machines, which accept coins and disburse change and goods by the indicated price using a basic mechanism (such as a finite automaton) [3]. By urging the digital insertion of contracts into many types of properties, beyond the vending machine, smart contracts exist [4]. Smart contracts might be much more useful than their inert paper-based relatives, according to Szabo, who also anticipated that they would be through the use of clear logic, cryptographic protocol verification, and enforcement [5]. The concept of smart contracts, however, was not realized until the advent of blockchain technology, where the consensus process and public and append-only distributed

ledger technology (DLT) allow for the implementation of smart contracts in their truest sense [6].

In general, smart contracts are computer protocols that use blockchain to digitally facilitate, verify, and execute agreements formed between two or more parties [7]. Smart contracts have certain distinctive features because they are often implemented on and secured by blockchain [8]. A smart contract is made impenetrable by being first recorded and verified in its computer code on the blockchain [9]. Second, without centralized management and the collaboration of outside authorities, an anonymous, untrusted network of independent nodes enforces the execution of a smart contract [10]. Third, a smart contract can store cryptocurrency, acquire more digital assets, and move them when certain criteria are met, much like an intelligent agent [11]. As a result of its transactions only being validated if specific conditions are met, it is important to note that Bitcoin is widely regarded as the first cryptocurrency to support fundamental smart contracts [12]. Scripting only supported by Bitcoin a little simple reasoning and math, as well as cryptographic functions, making it impossible to construct smart contracts with complicated logic [13].

With the aid of a Turing-complete virtual machine dubbed the Ethereum virtual machine, Ethereum is the first open-source platform that uses blockchain and allows sophisticated as well as personalized smart contracts (EVM) [14]. EVM implementations are run on each node of the Ethereum network that carries out similar instructions [15]. EVM is the runtime environment for smart contracts. The code for an Ethereum smart contract can be written in several high-level programming languages, including Solidity and Serpent, and then it can be placed on the blockchain for execution. The code is then translated into EVM bytecode. Digital rights management, crowdfunding, gaming, and other types of decentralized applications (DApps) can all be created using Ethereum, the most well-liked smart contract development platform now available [16].

Despite significant advancements in recent years, smart contracts still have several difficulties to overcome. A well-known incident occurred in June 2016 when an attack using the serious smart contract problem known as "Recursive call" targeted The DAO, an Ethereum network-secured decentralized investor-directed venture capital fund. In a "child DAO" with the same organizational structure as The DAO, the attacker poured more than $50 million in ether. To reclaim the assets from the attacker, Ethereum finally underwent a hard fork. Because it goes against the code of law premise that is central to blockchain technology, this hard fork has generated some controversy [17]. Performance, privacy, legal concerns, etc. are a few other difficulties besides the security issue [18].

The primary goal of this paper is to provide a thorough review of smart contract research, covering topics like the workings, the fundamental architecture, application scenarios, difficulties, most recent advancements, and potential future directions. The remaining portions of this essay are structured as follows. A fundamental research framework using a six-layer architecture is proposed in Section II, which presents smart contracts in detail, including their workings and popular development environments. Section III lists the current difficulties smart contracts are facing as well as recent developments in research. The Internet of Things (IoT), finance, management, and energy are just a few of the common application scenarios for smart contracts that are presented in Section IV. Future development tendencies are discussed in Section V. This essay comes to an end in Section VI.

2. Research Method

Artificial societies plus computational experiments plus parallel execution (ACP approach) is the only formalized research framework in the field of parallel organizational/societal management. It makes use of artificial systems for modeling and representation, computational experiments for analysis and evaluation, and parallel executions for controlling and managing complex systems [19]. The guiding theory, research, and conceptual framework methods for parallel blockchain were put forth [20]. To achieve parallel

P-ISSN: 2808-0831
E-ISSN: 2808-0009

organizational/societal management powered by smart contracts, the ACP model and blockchain make sense together.

First, the blockchain's P2P network, distributed consensus, and incentive structure are the most natural ways to model each node in a distributed system that will eventually make up software-defined organizations and social systems that are similar to artificial societies [21]. Second, the simulated experimental WHAT-IF designs, calculated experiments are used to deduce experimental scenarios and evaluate the results of the trials are made possible by the programmable feature of smart contracts, allowing the agents to choose the best course of action in a given situation. The ability to connect the real world and virtual cyberspace is made feasible by blockchain and Internet of Things together, which can produce a wide range of intelligent assets [22]. The ideal A management strategy for organizations and societies attained through virtual-real contacts and physical and artificial organizations' simultaneous development (corresponding to parallel execution). The manifestation layer of the study framework we suggested correlates to this [23].

3. Literature Review

Smart Contracts

A summary of smart contracts will be provided in this section. Before presenting the workings Ethereum and Hyperledger Fabric, two well-known systems for smart contracts, we provide a brief introduction to blockchain [24]. The historical development of blockchain-based smart contracts is depicted in Figure 1 with some key turning points [25]. Smart contracts were created in 1994 when Nick Szabo first introduced the idea to the public [26]. One of the most significant developments in smart contract history is the creation of Ethereum [27]. Users were able to join in and install smart contract apps in the public blockchains thanks to the Ethereum blockchain that was made available to everyone [28]. At first, currency trading was the main use for Ethereum. Collaboration with the Linux Foundation led to the creation of the Hyperledger Fabric project [29]. Since Hyperledger was designed to be an enterprise blockchain, the trajectory of Fabric has diverged from that of Ethereum. Targeting the needs of businesses, many platforms are being developed [30].

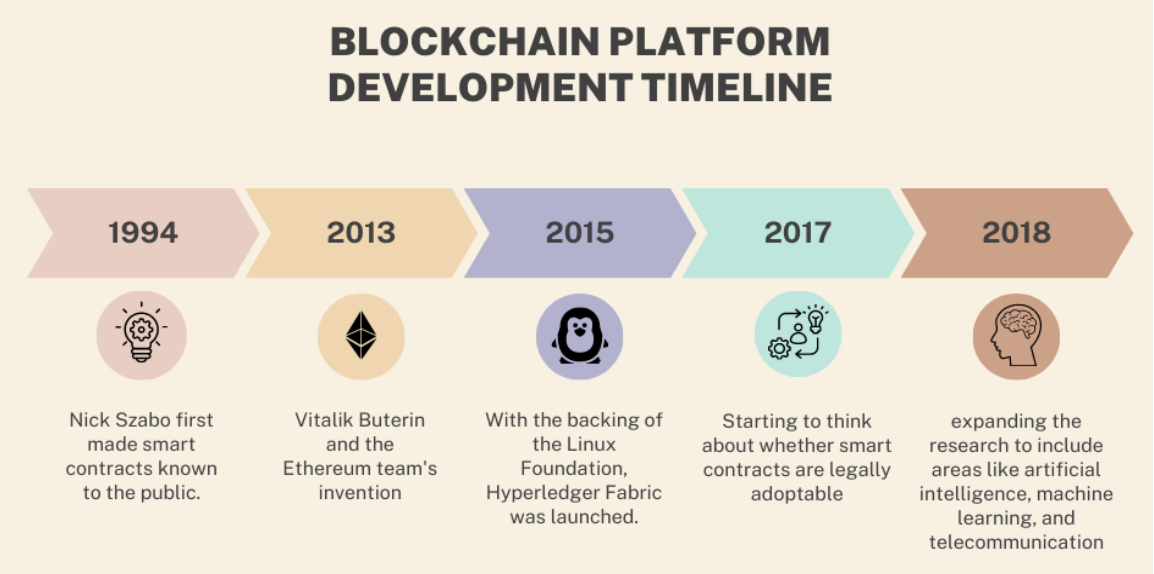

Figure 1. Shows the historical development of important blockchain platforms.

The study concentrated on the maturation of the smart contract context, which includes different platforms and a variety of use cases, and the legalization of smart contracts in that setting. The role of smart contracts in the new areas of computer science study is a major focus of the coming generation of studies.

A. Overview of Blockchain in a Few Words

Bitcoin, a cryptocurrency created in 2008 by an unidentified person or group of individuals under the pseudonym Nakamoto, served as the inspiration for the blockchain concept. Blockchain is a list of records that is constantly expanding and safeguarded using encryption. These records are connected and termed blocks. A single point of failure can be tolerated by the P2P protocol that Blockchain uses. The blockchain's integrity and consistency are guaranteed across geographically dispersed nodes thanks to the consensus mechanism, which also assures a uniform, clear ordering of transactions and blocks. Decentralization, honesty, and audibility are all qualities that blockchain was created with. Blockchain, in the opinion of Zen et al., can act as an original type of software interconnect and ought to be taken into consideration as a potential decentralized substitute for the current shared data storage in one location. Furthermore, blockchains can be categorized into three groups depending on varying degrees of access permission:

1) Private Blockchain.
2) Public Blockchain, including Ethereum as well as Bitcoin
3) Consortium Blockchain, including Ripple as well as Hyperledger

Smart contracts are housed and carried out on the blockchain as its underlying technology. By expressing business logic, circumstances, and triggers, smart contracts create programmable transactions that are complex. As computer programs, smart contracts are introduced. that execute throughout the network of blockchain. We will go through the specifics of how smart contracts work in more depth in the following section.

B. Smart Contracts' Operational Mechanism

Value and state are the two main aspects of smart contracts. Using activating conditional phrases, such as "If-Then" clauses, the contract terms' reaction actions and associated triggering situations are predetermined. All parties must agree to and sign a smart contract before it can be submitted to the blockchain network as a transaction. After that, the transaction is broadcast across the P2P network, miner verification is completed, and the transaction is then put in the relevant blockchain block. A contract can be invoked by users by submitting a transaction, and the contract creators get the returned parameters (for example, the contract address). Motivating component of the system encourages miners, and they will contribute their processing power to validate the transaction.

A contract is created or the contract code is executed in the local Sandboxed Execution Environment (SEE), for example, the EVM, by the miners after they receive the contract creation or invocation transaction. The agreement's terms specify whether the present scenario satisfies the triggering requirements based on the information from reliable data sources (also referred to as Oracles) as well as the system status. The response actions are strictly carried out if the prerequisites are met. A transaction is bundled into a new block once

it has been approved. Once the entire network agrees, the new block is chained onto the blockchain.

The operation of smart contracts is then explained using the examples of Ethereum and Hyperledger Fabric.

1. Ethereum

   A state machine that is transaction-based that starts through a genesis state as well as gradually conducts transactions to transform it into certain final states is Ethereum, currently the most popular platform for developing smart contracts. It is the ultimate state that we recognize as the canonical "version" in the Ethereum universe. Ethereum introduces the idea of accounts as opposed to Bitcoin's UTXO approach. Accounts can be classified as either contract accounts or externally owned accounts (EOAs). The first one is managed without using a private key any related code, whilst the latter is managed based on the contract code they had accompanying code. Only an EOA can be used by users to start a transaction. Binary data (payload) and ether are both permitted in the transaction. A smart contract is made if the recipient of a transaction is the zero account. Alternatively, if the receiver is a contract account, the contract account will be activated, and the local EVM will run any associated code. Following that, The blockchain network receives a broadcast of the transaction for validation by miners. All programmable computations in Ethereum (such as creating contracts, making message calls, using and accessing account storage, and performing operations in the virtual machine) are subject to fees as a reward for miners who contribute their computing resources to prevent issues of network abuse and to avoid the inevitable issues resulting from Turing completeness. Gas is the name of the measurement unit for the costs necessary for the computations.

1. Hyperledger Fabric

   One of The Linux Foundation's Hyperledger projects, Hyperledger Fabric is a blockchain framework implementation. Only a group of businesses-related organizations can join Hyperledger Fabric. Its network is made up of peers that are owned and contributed by those organizations and is made available through a membership service provider, as opposed to the public blockchain, like Bitcoin and Ethereum, where anyone can participate in the network. Peers serve as hosts for chain codes and ledgers (smart contracts). The sequential, impenetrable record of transactions and state changes is called a ledger. Chain code invocation causes the state to change (transaction). A collection of key-value pairs for each asset that is generated, changed, or removed as a result of a transaction is committed to the ledger. The Hyperledger Fabric transaction workflow consists of three phases.

1. Proposal: An implementation transmits a suggestion for a transaction to the endorsing peers of several organizations, also known as endorsers who verify transactions with endorsement guidelines and uphold the guidelines. Invoking a chain code function is what the proposal is asking for in order to read and/or write data to the ledger. The read set, write set, and response value are all included in the transaction results. As part of the transaction proposal response, the application receives the collection of these values as well as the endorsers' signatures.
2. Packaging: The program validates the consistency of the proposal responses and the endorsers' signatures. In order to update the ledger, the application then sends the

transaction to the ordering service (orderer). The order sorts the network transactions it has received and then gathers batches of them into a block that is prepared for distribution to all of its linked peers.

3. Validation: Every transaction in the block is validated by the peers related to the order to make sure that it has been consistently approved by the relevant organizations in accordance with the endorsement policy. It's important to remember that running chain code is only necessary during the proposal phase and is not required during this one. As soon as a block has been validated, each peer adds it to the chain, updating the ledger.

The following are some key areas where Ethereum and Hyperledger Fabric diverge. First, only a designated community of participants is allowed to join the network, whereas Hyperledger Fabric is a consortium blockchain infrastructure. This distinguishes Ethereum from Hyperledger Fabric as a public blockchain platform. In contrast, Hyperledger Fabric has a modular design with a distinction between the nodes' functions, such as endorsers and orderers, as well as customizable consensus and membership services. As a result, it has high levels of scalability, robustness, and confidentiality. Second, there are no built-in fuels like Ethereum's Ether and gas in Hyperledger Fabric, unlike other cryptocurrencies. Thirdly, the Hyperledger chain code merely defines a set of assets that are shown as key-value pairs and offers the functions for manipulating the assets and altering their states.

The transaction that is propagated contains the contract code over which any miner using the P2P network that gets This transaction has the ability to carry it out on a virtual machine they have locally. This brings us to the last point regarding contract code execution. The chain code, however, is hosted by peer nodes in Hyperledger Fabric. Specified peers are the only ones who can execute and sign transactions that the application creates. These endorsing peers independently execute each transaction after receiving it from the application by calling the transaction's referenced chain code. Chain code operates in an isolated environment called a container, such as Docker, for security reasons. It is important to note that Ethereum and Hyperledger are increasingly converging. For instance, the Hyperledger Fabric EVM chain code plugin now supports the execution of Ethereum smart contracts on Fabric by the Hyperledger Burrow project, which is powered by the Tendermint consensus engine.

C. Smart Contracts: Basic Research Framework

We divide the life cycle of a smart contract into five stages based on the operating mechanism of smart contracts: negotiation, development, deployment, maintenance, and learning and self-destruction. We provide a fundamental research paradigm for smart contracts based on this life cycle. The framework also refers to a variety of earlier works. To structure the conclusions of the current corpus of research on blockchain technology, Risius and Spohrer, for instance, provided a research framework. A taxonomy for categorizing and contrasting blockchains and blockchain-based systems was put forth by Prevy et al. The taxonomy encapsulates the key architectural elements of blockchains and the effects of various design choices, which aids in crucial architectural considerations regarding the functionality and standard of blockchain-based systems. The fabric layer and the application layer are the two coding levels that Glaser created as part of a comprehensive conceptual framework for blockchain systems.

The suggested research framework has a six-layer design, starting at the bottom with the infrastructures layer and working its way up through the contracts layer, operations layer, intelligence layer, manifestations layer, and applications layer. Following are the specifics.

1. Infrastructures Layer

   The trusted development trustworthy execution environments, trusted data feeds, and environments are all included in the infrastructures layer, which also includes all other infrastructures that support smart contracts and their applications. The design patterns and contract features of smart contracts will be influenced to some extent by the infrastructures chosen.

   a. Development Environments that are Trusted: IDEs (integrated development environments) and programming languages, development frameworks, clients, wallets, etc. are just a few of the development tools that are used during the development, deployment, and activation of smart contracts. As an illustration, the wallet typically performs tasks including acting as a boot node, deploying a contract, and executing a contract in addition to serving as a tool for managing digital assets.
   b. Environments for Trusted Execution: Smart contracts may be executed in a trusted environment thanks to blockchain technology. The consensus process, incentive system, and P2P network of the blockchain are all essential to the execution of smart contracts, and the outcomes of that execution a distributed ledger that is kept up by all nodes will contain a record of it. The design pattern, execution effectiveness, and security of smart contracts will all be impacted by various consensus algorithms and incentive structures. Attacks that cause a denial of service and excessive expenditures, the massive calling of dead code, opaque predicates, expensive operations in a loop, and other gas-costly operations, as well as the exception for running out of petrol brought on by the gas shortage, are just a few examples of how fuel consumption must be taken into account when developing and deploying smart contracts in Ethereum.
   c. Trusted Data Feeds: Smart contracts are typically run in SEE, such as a Docker container in Hyperledger Fabric or an EVM in Ethereum, which is not permitted for the importation of outside data. This is done to ensure the security of the blockchain network. Since anything a transaction hasn't produced in terms of information must be added in the form of transactional data in a secure and trusted manner in order to guarantee deterministic contract execution results, the smart contract requires trustworthy data sources that deliver transactional external states about the real world.

1. Contracts Layer

   The static contract information, such as the contract terms, scenario response guidelines, and interaction criteria, is contained in the contracts layer. As a result, these layers could be thought of as a smart contracts static database that contains all the rules for initiation, performance, and communication of contracts. The terms of the contract, which may include legal clauses, business logic, as well as agreements regarding intent, must first be negotiated and agreed upon by all parties when creating

a smart contract. Then, programmers programmatically convert the plain language descriptions of the contract terms, for example, a sequence rules for scenarios of the If-Then type, using software engineering techniques such as algorithm design and design patterns. Additionally, it is determined by the qualities of the development platforms as well as the intentions of the contractors, interaction criteria, such as access rights, the way in which you communicate, etc. For interactions between contracts and users, this layer should also be created.

2. Operations Layer

   The mechanism design, formal verification, security analysis, updates, and self-destruction are all dynamic actions on static contracts that are contained in the operations layer. The correct, secure, and effective execution of smart contracts depends on the maintenance layer since malevolent or insecure smart contracts can lead users to suffer significant financial losses. Before being implemented on the blockchain, mechanism design procedures use information and incentive theory to assist contracts to carry out their purpose effectively. This is seen from the perspective of the life-cycle of smart contracts, which spans from negotiation through self-destruction. To confirm the accuracy and security of contract codes as well as to guarantee that the codes will be executed in accordance with the programmers' actual semantics, formal verification and security analysis activities are used. When a smart contract's functionality is difficult to match user requests or the contract has patchable vulnerabilities after it has been published onto the blockchain, updates can be theoretically made, however all previous updates are recorded on the blockchain and cannot be altered. Self-destruction is carried out to ensure network security towards the end of the smart contracts' life cycle or whenever a high-risk vulnerability appears.

3. Intelligence Layer

   The sensing, reasoning, learning, decision-making, and socializing algorithms that make up the intelligence layer provide intelligence to the smart contracts created using the preceding three layers. It must be noted that smart contracts as they exist today lack significant intelligence. However, we think that future smart contracts should also contain "What-If"-type deduction, computation, and intelligent decision-making in unknowable scenarios in addition to being self-enforcing by the preset If-Then statements. As was already said, software agents that operate on behalf of their users can be referred to as smart contracts that are running on the blockchain network. Because of cognitive computing, reinforcement learning, and other advancements in artificial intelligence technology, agents will eventually possess some intelligence, including perception, reasoning, and learning. Because of this, those agents are not only autonomous in that they can choose which tasks to focus on first, prioritize them, and engage in goal-directed behavior (also known as belief-desire-intention behavior), but they are also sociable because they can interact with one another and cooperate and negotiate with one another. To optimize contract design and operation and ultimately create the really "smart" contract, the learning and cooperation results will also be transmitted back to the contracts layer and the operations layer.

4. Manifestations Layer

   DApps, decentralized autonomous organizations (DAOs), decentralized autonomous corporations (DACs), and decentralized autonomous societies are just a few examples of potential applications that could use the manifestation layer's many manifestation types of smart contracts (DASs). The application interfaces of blockchain are analogous to smart contracts, which can integrate various application scenarios by encapsulating complicated network node behaviors. Examples of DApps that can be created include those that incorporate intention agreements, business logic, and legal clauses into smart contracts. Additionally, different DAOs, DACs, and DAS will gradually develop from the multiagent systems constructed on the fourth layer. These advanced manifestation forms are anticipated to advance conventional commercial and management practices and provide the groundwork for a programmable society in the future. Consider the DAO as an example. DAOs are businesses that are powered by smart contracts and are managed by blockchains, with all of their operational procedures being documented there. DAOs have the potential to bring about a more decentralized alignment of stakeholder interests while lowering transaction costs. Since traditional management paradigms often have a top-down hierarchical structure, DAOs are anticipated to have a disruptive impact on them.

5. Applications Layer

   All application domains that are constructed on top of the manifestation layer are contained within the applications layer. For instance, the distributed autonomous art platform Plantoid was created in Ethereum based on the DAO protocol. It realized an economy that was genuinely beautiful, connecting creators, designers, works of art, a mutually beneficial relationship with audiences, freeing art from monopolistic as well as markets that are arranged in a hierarchy. All businesses, including finance, IoT, healthcare, supply chain, etc., could theoretically employ smart contracts.

It is important to remember that the suggested architecture, particularly for the intelligence layer, is simply an ideal framework. The fact that any activity in the blockchain must be initiated by a network node that is externally controlled, as Glaser also noted, is a functional limitation. As a result, smart contracts should use self-reliant techniques or intricate microservice interactions that, when combined, realize more complex service logic, such as an autonomous portfolio management service. Smart contracts of the future ought to be autonomous and intelligent. For academics and professionals, the suggested framework has some theoretical and practical relevance. On the one hand, the framework includes all of the essential components for the smart contract throughout its whole lifecycle. The framework, on the other hand, identifies the direction of the study and potential growth patterns.

5. Findings

Smart Contracts Application Scenarios

Smart contract applications are currently mushrooming. The application scenarios for smart contracts will be introduced in this part using examples from the fields of finance, management, IoT, and energy.

P-ISSN: 2808-0831
E-ISSN: 2808-0009

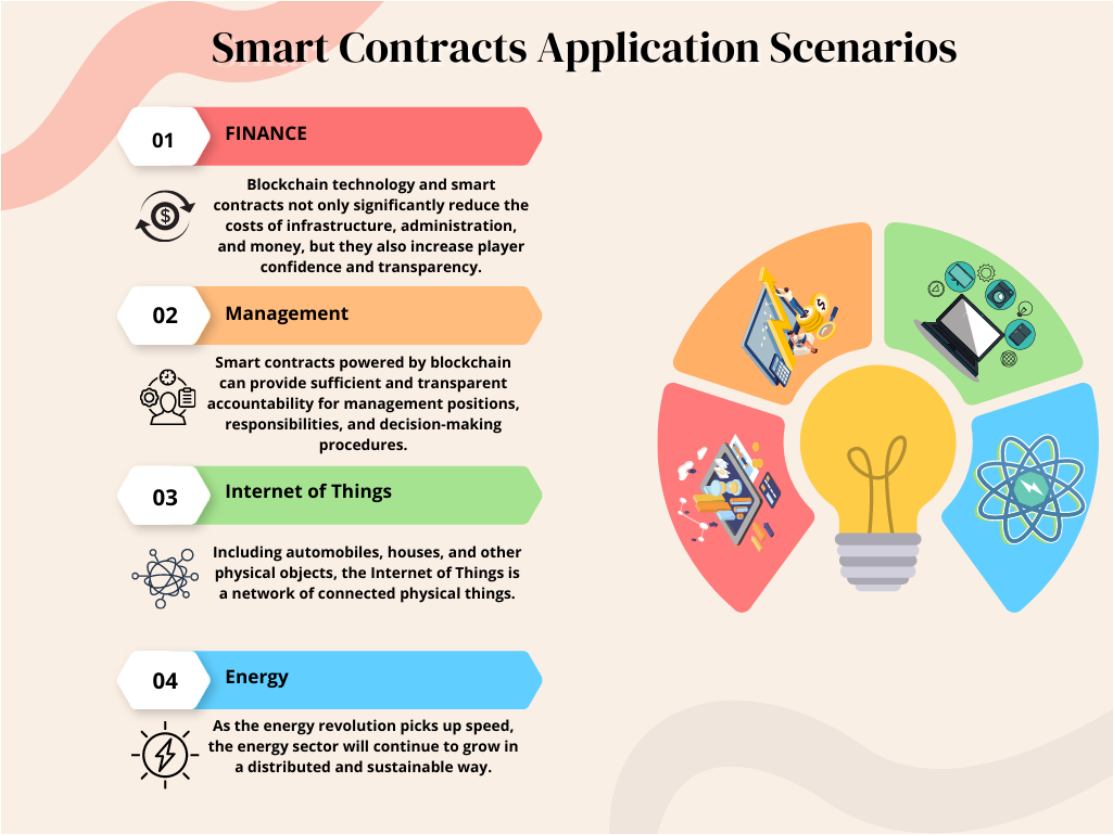

Fig. 4. The application scenarios for smart contracts.

A. Finance

In addition to providing significant financial, infrastructural, and administrative cost savings, blockchain technology and smart contracts provide enhanced transparency and trust among players. The following list includes a number of common uses for smart contracts in the financial sector.

1) Securities: Complex processes that are time-consuming, expensive, burdensome, and risky are used in the security business. The automatic payment of dividends, stock splits, and liability management can all be made possible by smart contracts, which can also lower operational risks by avoiding middlemen in the chain of custody for securities. The clearing and settlement of securities can also be made easier via smart contracts. Major markets in the U.S., Canada, and Japan continue to use a 3-day settlement cycle (T+3) that incorporates numerous institutions, including securities depositories and collateral management firms. Complex internal and external reconciliations and labor-intensive tasks go along with centralized clearing. Smart contracts can be used to execute bilateral peer-to-peer clearing business logic on the blockchain. For the purpose of replacing its stock settlement system, the Australian Securities Exchange is developing a post-trade platform based on DLT.
2) Insurance: The processing of claims costs the insurance industry tens of millions of dollars annually, and it also suffers millions of dollars in losses due to bogus claims.

Utilizing smart contracts can speed up claim processing while also reducing fraud and avoiding potential dangers. Smart contracts can be used to automate claim processing, verification, and payment. AXA, a French airline, is one example of a company using smart contracts to provide flight insurance. The compensation alternatives will be immediately communicated to travelers whose flights are more than two hours late. Due to their ability to store insurance terms, driving histories, and accident reports, smart contracts may also be employed in the auto insurance industry. This will enable IoT-enabled automobiles to process claims quickly following an accident.

3) Trade Finance: Currently, the trade finance sector is very inefficient and very susceptible to fraud. Also urgently needed is an improvement or replacement of the paper-based trade finance processes with digital ones. Smart contracts give businesses the ability to autonomously initiate business actions based on established criteria, which will increase productivity by streamlining procedures and lowering the expenses associated with fraud and compliance. Australia and Japan concluded a trade agreement in July 2017. The Hyperledger Fabric platform was used exclusively for this trade transaction, which lowered the amount of time it took to transfer documents as well as labor and other costs. This included everything from issuing a letter of credit to sending trade paperwork.

B. Management

In terms of management roles, duties, and decision-making processes, blockchain-enabled smart contracts can offer adequate and transparent accountability. Below are a few use cases.

1) Rights management and Digital Properties: Access and validation can be facilitated by storing certificates of properties or rights in encrypted form on the blockchain. Smart contracts were suggested to be used by Joshua et al. to vouch for the authenticity of intellectual property proofs of existence and authorship. Propy enables real estate brokers and owners to list their listings so that potential buyers can find them and conduct a sale negotiation. Together with the other party, both parties take part in the smart contracts, and procedures are taken at various points to guarantee fair and legal play. In digital rights management, smart contracts can also be used. When a musician's work is used for commercial gain, for instance, a DApp with the name of Ujo Music requires that royalties be paid.
2) Management within Organizations: These days, a board of directors, which now controls the majority of organizational decisions, manages and centers most firms. Organizational management will reportedly become more decentralized and flat in the future. Ineffective intermediaries that impose artificial limitations and needlessly complex regulations can be eliminated by smart contracts. As an illustration, the Aragon project, which uses Ethereum to power it, enables users worldwide to manage their organizations quickly and securely. It intends to remove the middleman from the process of building and maintaining organizational structures. In Aragon, tokens stand in for your ownership interest in the company. You may also hire new employees and use crowdfunding to generate money internationally. Voting also gives you more control over the outcome of elections.

3) E-Government: E-government can become more effective and powerful by streamlining bureaucratic procedures with smart contracts. For instance, Chancheng District in Foshan, China, created the first E-government service platform using blockchain and smart contracts technology to enhance the caliber of governmental services, create a personal credit system, increase the government's credibility, and encourage resource integration. Innovative payment methods for labor and pensions, bolstering international aid programs, electronic voting, and other uses for smart contracts are some other E-Government application areas.

C. Internet of Things

The Internet of Things (IoT) is a network of interconnected physical objects, including cars, homes, and other objects. IoT is reportedly used extensively in a variety of industries, including smart manufacturing, smart grids, smart housing, and intelligent transportation systems. The old centralized Internet system struggles to address the IoT's development needs, including the security of sensitive data and reliable communication between several devices. With smart contracts, the complex workflow can be automated, resource sharing is encouraged, prices are reduced, and efficiency and safety are ensured. As a result, the convergence of IoT and blockchain becomes inevitable. Using simulated studies, Karl et al. demonstrated that the suggested model may greatly lower the daily management expenses of IoT devices. The model is built on blockchain technology and smart contracts, and it discusses numerous interaction processes within the model. To accomplish distributed and dependable access control for IoT systems, Stephany et al. suggested a smart contract-based framework made up of numerous access control contracts, a judge contract, and a register contract. Iotex is a decentralized IoT network that uses blockchain technology and is privacy-focused. It supports a number of IoT ecosystems, including supply chain, smart homes, shared economies, and identity management.

D. Energy

The energy business will continue to flourish in a distributed and clean manner as the energy revolution gains momentum. To create decentralized energy trading markets, increase energy consumption efficiency, and lower grid running costs, distributed energy systems can be developed using blockchain technology, as well as smart contracts for energy supply and trade. The primary use cases for energy blockchain initiatives at the moment are distributed energy, electric vehicles, energy trade, carbon tracking, and registries. A blockchain technology developed by a collaboration called Exergy enables the creation of regional energy markets for the exchange of energy across current grid infrastructures. On the Exergy platform, prosumers that produce their renewable energy can transact energy on their own with customers in their neighborhood. The Sun Exchange is a blockchain-enabled marketplace that enables its members to buy solar cells and then lease them to communities, businesses, and schools in the sunniest regions of the world (primarily Africa). The amount of money that members make depends on how much electricity their solar cells have generated. The collecting and distribution of the monthly lease rentals will be handled by The Sun Exchange. A trustworthy, automated, and privacy-preserving method of choosing charging stations based on cost and proximity to electric vehicles was presented by Lerona et al. The suggested protocol was developed on a blockchain where charging stations place bids and electric vehicles express their demand. Healthcare, prediction markets, intelligent

transportation systems, and other areas are some of the other application possibilities for smart contracts.

3.1 Problem

Smart contracts confront numerous issues and difficulties at the moment because they are a young, evolving technology. This section will describe the problems with smart contracts and the most current developments in their study, which is based on the proposed research framework that uses a six-layer design.

A. Contract Vulnerabilities

According to the research paradigm we suggested, contract vulnerabilities mostly affect the contracts layer. They may be used for financial benefit by malevolent miners or users. Here are a few representative situations.

1) Depending on Transactions: Several transactions are included in each block, and the miner determines the order in which they are carried out. The term "DOT" refers to a situation in which a number of dependent transactions invoke a single contract and the miner can control the execution order.
2) Dependence on Timestamp: The timestamp for the block that the miners set was often established in accordance with the miner's local clock system. On condition that other miners agree to the block they suggested, the miner is permitted to change the timestamp by a few seconds. Because some smart contracts rely on timestamps as a trigger condition, such as those that include money transfers, this creates a vulnerability that allows for timestamp-dependent contracts to be manipulated by adversaries for their own purposes.
3) Awful Exception Handling: If a callee runs abnormally when a contract (the caller) calls it, the callee terminates and returns false. The caller may or may not receive this exception. In order to confirm that the call was successfully made, the caller must, in theory, explicitly inspect the callee's return value. However, possible dangers could arise if the caller does not thoroughly inspect the return result. The Ethereum King of the Ether Throne contract serves as a representative example.
4) Re-Entrancy Vulnerability: The present execution delays moving forward when one contract contacts another. Attackers may leverage the caller's intermediate state to make repeated calls because the fallback method enables them to re-enter the caller function. This can result in loops of invocations that retrieve several refunds and empty the balance. The DAO attack is the most well-known re-entrance vulnerability.
5) Call Stack Depth: The transaction's call stack increases by one frame for each contract that is invoked. Ethereum has a 1024 frame limit on the call stack. Any further invocation throws an exception after this cap is reached. In order to run the victim's function and cause it to produce an exception, the attacker first builds up a nearly full call stack. The adversary may be able to carry out their assault if the victim's contract fails to handle the exception properly.

An Ethereum smart contract security analyzer is called Security. Its study entails two steps: first, it uses a symbolic analysis of the dependency graph of the contract to extract exact semantic data from the code. The process then looks for patterns of compliance and violation that capture the necessary circumstances to demonstrate whether a property is valid

or not. Transaction reordering, recursive calls, unsafe coding practices, etc. are just a few examples of vulnerabilities that security can examine.

B. Blockchain's Limitations

One of the main obstacles to the development of the smart contract is the blockchain's inherent constraints. The infrastructure layer of the framework for smart contracts that we suggested corresponds to these restrictions. The following list includes several common restrictions.

1) Permanent Bugs: The smart contracts are finished and cannot be modified once they are deployed because of the blockchain's irreversible nature. In other words, if a smart contract has a bug, there is no easy method to correct it. Therefore, you must update a smart contract whenever a flaw is discovered. It is also highly laborious to manually initialize the new contract with the old data when you deploy a new version of an existing contract because data saved in the old contract is not immediately transferred.
2) Performance Issues: Smart contracts' performance is further hampered by performance difficulties in blockchain systems such restricted scalability, throughput bottlenecks, transaction latency, and storage limitations. As an illustration, consider how throughput works in the current blockchain systems. Miners and validators execute smart contracts in a serial fashion. Today's concurrent multicore and cluster architectures cannot be utilized by serial execution, which also limits the system throughput. Based on methods adopted from software transactional memory, Richalison et al. presented a novel mechanism to enable miners and validators to execute smart contracts concurrently.
3) Lack of Trusted Data Feeds: As previously established, trusted data feeds provide a link between the blockchain and the outside world, such as a Web API, by providing external data regarding actual states and events from sources outside the network. Smart contracts' development is frequently seen as being severely hampered by the absence of a robust ecosystem of reliable data sources. Reynald et al. developed a Town Crier (TC) solution to address this issue, acting as a trustworthy bridge between blockchain and websites that support HTTPS to deliver authenticated data feed for smart contracts.
4) Standards and regulations are lacking: The absence of standards and laws is one of the main hazards to and problems with blockchain security. Suly et al. introduced the idea of criminal smart contracts (CSCs) and presented several examples of common CSCs, including the theft of cryptographic keys, the release of private information, and other actual crimes (murder, arson, and terrorism). Because there is no reliable legal framework in place, it is challenging to monitor malicious actions that take place in smart contracts. Some regulatory organizations, such as the U.S. Securities and Exchange Commission, started to focus on the operational and regulatory problems brought on by these new technologies in response to the possible high-security concerns of blockchain and smart contracts.

C. Legal Concerns and Privacy

The infrastructure layer and contracts layer of the research framework we suggested is involved in the two categories of contract data privacy and trustworthy data feed privacy,

which are privacy concerns with respect to smart contracts. Currently, information pertaining to contracts, such as the bytecode, invocation parameters, and other related information, is available to the public (particularly for information on the public blockchain). Therefore, it poses a serious problem to use cryptography, prevent the disclosure of sensitive information, and keep important processes and procedures hidden. Hawk is a decentralized smart contract system that was proposed by Sency et al. Its compiler automatically creates an effective cryptographic protocol where contractual parties interact with the blockchain using cryptographic primitives like zero-knowledge proofs, allowing developers to create privacy-preserving smart contracts without the need to implement any cryptography. Smart contracts should be encrypted before being put into use on the blockchain, according to Hori et al. By employing the decryption keys, only parties to a contract are able to access its contents. For the security of access-controlled, off-chain data sources and the privacy of trustworthy data feeds, TC enabled private and custom data requests, enabling encrypted queries. The contracts layer serves as the primary vehicle for the legal aspects of smart contracts. Some academics contend that a smart contract is essentially a form of computer code that has the ability to self-enforce, self-verify, and self-constrain the implementation of its instructions, which may represent all, some, or none of the legally binding agreements under the laws in effect. Because of this, relational contract theories and smart contracts can be at odds. The "right to be forgotten" provision of European data privacy regulations, for instance, conflicts with the immutability of smart contracts that use blockchain technology. Besides those mentioned above, there are other legal difficulties as well.

1) Other than those mentioned above, what laws apply to the transactions occurring within the smart contract application?
2) What risks (such as data loss, business interruption, privacy violation, and/or performance failure) are associated with using the smart contract application alone?
3) What occurs if a smart contract produces results that are different from those required by law?

There are further difficulties besides the ones mentioned above on the contractual and infrastructure layers. For instance, bad smart contract mechanism design will result in higher contract execution costs and lower contract execution efficiency on the operations layer. To achieve incentive compatibility, designers must create a collection of incentive systems that balance individual interests with the overarching goals of the organization or society. Malicious intelligent agents may profit from their malign activity on the intelligence layer, etc.

3.2 Research Implementation

Formal Parallel smart contracts-driven verification, Layer 2, organizational/Management of society are the three areas from which We'll discuss how smart contracts will develop in the future in this part.

A. Formal Verification

Proof that the software operates by a specification is applied during formal verification. To specify the relationship between the input and output functions, a concrete specification language is typically utilized to do. A contract program must be shown to adhere to a formal definition of its behavior for smart contracts to be formalized. The research framework we

suggested, relates to the operations layer. The Lem language was used by Hirai to develop a formal model for the EVM. By using interactive theorem provers, the suggested model demonstrated the safety characteristics of a smart contract. By adding sound program logic at the bytecode level, Utoda et al. expanded an earlier EVM formalization in Isabelle/HOL. To provide a strong foundation for additional formal analyses, Brandon et al. presented KEVM, an executable formal specification of the K Framework-built EVM byte-code stack-based language. By converting the Functional programming language F* was created using the Solidity program and EVM bytecode intended for program verification, Sinyo et al. proposed a system to evaluate formalization confirming the accuracy of the functional specifications, runtime safety of the Ethereum smart contracts. The majority of these formal verification tools are still in the experimental stage and aren't very popular. As it offers the highest level of assurance on the proper Smart contracts' actions, formal verification will soon become a crucial study area.

B. Layer 2

As previously stated, smart contracts encounter a number of difficulties, including inadequate performance, an inability to manage sophisticated logic execution as well as high-throughput data, insufficient safeguards for privacy, and an inability to execute cross-chain. In the future, Layer 2 will be a practical answer. This layer in our research framework's infrastructures hierarchy corresponds to it. By separation between the smart contract execution and the blockchain's consensus process, Layer 2 establishes an environment for off-chain contract execution. This environment achieves a high level of performance and privacy by using the blockchain as the "consensus layer" to transition contract-related states and token payment. Off-Chain State Channels is one Layer 2 solution that establishes a bidirectional conduit connecting users or users and services to provide state maintenance services between various entities. With the full assurance that they can switch if required, back to the main chain, State Channels allow carrying the off-chain processing of transactions and other state updates. Furthermore, Plasma enables the establishment of "child" Ethereum blockchains that are connected to the main chain but rely on the underlying blockchain for security, enabling off-chain transactions. Through the use of smart contracts, Truebit21 enables the off-chain execution of computationally demanding operations.

C. Organizational and societal parallelism powered by smart contracts

The management style of contemporary enterprises and societies has undergone a significant change as a result of the Internet's quick development and close connection to the physical world. The transition systems that are both cyber-physical and cyber-physical-social (CPSSs), where social and individual variables consideration must be given, is necessary for companies' and societies' future development trends. The idea CPSS has shown a number of parallel societies just emerged, and their main traits are complexity owing to social complexity, diversity, and uncertainty. Because they offer efficient interactivity and decentralized data structures mechanisms with regard to distributed AI and social systems, the infrastructure for establishing parallel organizations and societies based on CPSS is blockchain and smart contracts. As was previously said, smart contract-running nodes can be thought of as software agents that comprehend and respond to their surroundings. Different nodes produce different DAOs/DACs/DASs as a result of autonomously deploying and executing contracts on behalf of the diverse interests of the many individuals represented by those nodes within an

organization or society. DAOs, DACs, and DASs can assist in resolving the principal-agent dilemma, which is the primary issue in the field of organizational management, in addition to the traditional organizations and societies that are structured in a hierarchical structure and top-down directives.

5.Conclusion

Emerging smart contracts have been a major research issue in academic and industry groups due to the growing popularity and expanded applications of blockchain technology. Smart contracts allow for the execution of contract terms between untrusted parties without the need for a central server or a trusted authority due to its decentralization, enforceability, and verifiability features. So many traditional industries, including finance, management, IoT, etc., are projected to undergo radical change as a result of smart contracts. The operational mechanism, popular platforms, and application scenarios of smart contracts are all covered in detail in this paper's overview. To be more precise, we provide a fundamental research framework for smart contracts built on a brand-new six-layer architecture.

From the multiagent systems built on the fourth layer, many DAOs, DACs, and DAS will gradually emerge. The development of conventional business and management methods as well as the foundation for a programmable society in the future are anticipated benefits of these enhanced manifestation forms. Future versions of smart contracts should be self-contained and intelligent. Future trends in business and society call for a shift from cyber-physical systems to cyber-physical-social systems (CPSSs), where social and individual factors must be taken into account.

The remaining difficulties for smart contracts are then discussed, as well as recent research developments. A discussion of potential development tendencies follows. This work focuses on doing a comprehensive evaluation of smart contracts and identifying key research gaps that should be filled in next investigations.

References

[1] Oganda, F. P., Hardini, M., & Ramadhan, T. (2021). Pengaruh Penggunaan kontrak cerdas pada Cyberpreneurship Sebagai Media Pemasaran dalam Dunia Bisnis. *ADI Bisnis Digital Interdisiplin Jurnal*, *2*(1 Juni), 55-64.

[2] Woebbeking, M. K. (2019). The impact of smart contracts on traditional concepts of contract law. *J. Intell. Prop. Info. Tech. & Elec. Com. L.*, *10*, 105.

[3] Low, K. F. (2020). Confronting Cryptomania. *Kelvin FK LOW,"Confronting Cryptomania: Can Equity Tame the Blockchain*, *14*.

[4] Rahardja, U., Sudaryono, S., Santoso, N. P. L., Faturahman, A., & Aini, Q. (2020). Covid-19: Digital Signature Impact on Higher Education Motivation Performance. *International Journal of Artificial Intelligence Research*, *4*(1), 65-74.

[5] Febriyanto, E., Rahardja, U., Faturahman, A., & Lutfiani, N. (2019). Sistem Verifikasi Sertifikat Menggunakan Qrcode pada Central Event Information. *Techno. Com*, *18*(1), 50-63.

[6] Hassanein, A. A., El-Tazi, N., & Mohy, N. N. (2022). Blockchain, smart contracts, and decentralized applications: an introduction. *Implementing and leveraging blockchain programming*, 97-114.

[7] Toorajipour, R., Oghazi, P., Sohrabpour, V., Patel, P. C., & Mostaghel, R. (2022). Block by block: A blockchain-based peer-to-peer business transaction for international trade. *Technological Forecasting and Social Change*, *180*, 121714.

[8] Mardisentosa, B., Rahardja, U., Zelina, K., Oganda, F. P., & Hardini, M. (2021, November). Sustainable learning micro-credential using blockchain for student achievement records. In *2021 Sixth International Conference on Informatics and Computing (ICIC)* (pp. 1-6). IEEE.

[9] Wahyuningsih, T., Oganda, F. P., & Anggraeni, M. (2021). Design and Implementation of Digital Education Resources Blockchain-Based Authentication System. *Blockchain Frontier Technology*, *1*(01), 74-86.

[10] Eenmaa-Dimitrieva, H., & Schmidt-Kessen, M. J. (2019). Creating markets in no-trust environments: The law and economics of smart contracts. *Computer law & security review*, *35*(1), 69-88.

[11] Hasan, M., & Starly, B. (2020). Decentralized cloud manufacturing-as-a-service (CMaaS) platform architecture with configurable digital assets. *Journal of manufacturing systems*, *56*, 157-174.

[12] Kirli, D., Couraud, B., Robu, V., Salgado-Bravo, M., Norbu, S., Andoni, M., ... & Kiprakis, A. (2022). Smart contracts in energy systems: A systematic review of fundamental approaches and implementations. *Renewable and Sustainable Energy Reviews*, *158*, 112013.

[13] Oganda, F. P., Rahardja, U., Aini, Q., Hardini, M., & Bist, A. S. (2020). Blockchain: Visualization of the Bitcoin Formula. *PalArch's Journal of Archaeology of Egypt/Egyptology*, *17*(6), 308-321.

[14] Lu, J., Wu, S., Cheng, H., & Xiang, Z. (2021). Smart contract for distributed energy trading in virtual power plants based on blockchain. *Computational Intelligence*, *37*(3), 1445-1455.

[15] Mohan, D., Alwin, L., Neeraja, P., Lawrence, K. D., & Pathari, V. (2022). A private ethereum blockchain implementation for secure data handling in internet of medical things. *Journal of Reliable Intelligent Environments*, *8*(4), 379-396.

[16] Oganda, F. P., Lutfiani, N., Aini, Q., Rahardja, U., & Faturahman, A. (2020, October). Blockchain education smart courses of massive online open course using business model canvas. In *2020 2nd International Conference on Cybernetics and Intelligent System (ICORIS)* (pp. 1-6). IEEE.

[17] Widayanti, R., Purnama Harahap, E., Lutfiani, N., Putri Oganda, F., & Manik, I. S. P. (2021). The Impact of Blockchain Technology in Higher Education Quality Improvement. *J. Ilm. Tek. Elektro Komput. Dan Inform*, *7*, 207-216.

[18] Ahmad, D., Lutfiani, N., Ahmad, A. D. A. R., Rahardja, U., & Aini, Q. (2021). Blockchain technology immutability framework design in e-government. *Jurnal Administrasi Publik (Public Administration Journal)*, *11*(1), 32-41.

[19] Manongga, D., Rahardja, U., Sembiring, I., Lutfiani, N., & Yadila, A. B. (2022). Dampak Kecerdasan Buatan Bagi Pendidikan. *ADI Bisnis Digital Interdisiplin Jurnal*, *3*(2), 41-55.

[20] Lynberg, L., & Deif, A. (2023). Network effects in blockchain and supply chain: a theoretical research synthesis. *Modern Supply Chain Research and Applications*, (ahead-of-print).

[21] Deepa, N., Pham, Q. V., Nguyen, D. C., Bhattacharya, S., Prabadevi, B., Gadekallu, T. R., ... & Pathirana, P. N. (2022). A survey on blockchain for big data: approaches, opportunities, and future directions. *Future Generation Computer Systems*.

[22] Manongga, D., Rahardja, U., Sembiring, I., Lutfiani, N., & Yadila, A. B. (2022). Pengabdian Masyarakat dalam Pemberdayaan UMKM dengan Melakukan Implementasi Website Menggunakan Plugin Elementor Sebagai Media Promosi. *ADI Pengabdian Kepada Masyarakat*, *3*(1), 44-53.

[23] Torkashvand, M., Neshat, A., Javadi, S., & Yousefi, H. (2021). DRASTIC framework improvement using stepwise weight assessment ratio analysis (SWARA) and combination of genetic algorithm and entropy. *Environmental Science and Pollution Research*, *28*, 46704-46724.

[24] Ding, M., Li, P., Li, S., & Zhang, H. (2021). Hfcontractfuzzer: Fuzzing hyperledger fabric smart contracts for vulnerability detection. In *Evaluation and Assessment in Software Engineering* (pp. 321-328).

[25] Hasan, H. R., & Salah, K. (2019). Combating deepfake videos using blockchain and smart contracts. *Ieee Access*, *7*, 41596-41606.

[26] Vigliotti, M. G. (2021). What do we mean by smart contracts? Open Challenges in Smart Contracts. *Frontiers in Blockchain*, *3*, 553671.

[27] Chen, J., Xia, X., Lo, D., Grundy, J., & Yang, X. (2021). Maintenance-related concerns for post-deployed Ethereum smart contract development: issues, techniques, and future challenges. *Empirical Software Engineering*, *26*(6), 117.

[28] Kumar, A., Abhishek, K., Nerurkar, P., Ghalib, M. R., Shankar, A., & Cheng, X. (2022). Secure smart contracts for cloud‐based manufacturing using Ethereum blockchain. *Transactions on Emerging Telecommunications Technologies*, *33*(4), e4129.

[29] Mohammed, A. H., Abdulateef, A. A., & Abdulateef, I. A. (2021, June). Hyperledger, Ethereum and blockchain technology: a short overview. In *2021 3rd International Congress on Human-Computer Interaction, Optimization and Robotic Applications (HORA)* (pp. 1-6). IEEE.

[30] Corradini, F., Marcelletti, A., Morichetta, A., Polini, A., Re, B., Scala, E., & Tiezzi, F. (2021). Model-driven engineering for multi-party business processes on multiple blockchains. *Blockchain: Research and Applications*, *2*(3), 100018.