BLOCKCHAIN BASED SMART CONTRACT FOR A BANKING APPLICATION USING SOLIDITY

DR PREETHI J	STAFF, CSE DEPARTMENT, AURCC, COIMBATORE, INDIA.
KAVIYA S R	STUDENT, EEE DEPARTMENT, AURCC, COIMBATORE, INDIA.
MANIKANDAN K	STUDENT, EEE DEPARTMENT, AURCC, COIMBATORE, INDIA.
TEJAL B	STUDENT, EEE DEPARTMENT, AURCC, COIMBATORE, INDIA.
THARUN A	STUDENT, EEE DEPARTMENT, AURCC, COIMBATORE, INDIA.

ABSTRACT:

A new crypto-economy has emerged in recent years as a result of the swift advancement of block chain technology and crypto currencies, which has had an impact on the financial sector. Then, thanks to the advent of smart contracts, which are computer protocols created to facilitate, verify, and enforce automatically the negotiation and agreement among numerous unreliable parties, next-generation decentralized apps without involving a trusted third party have developed. Despite the positive aspects of smart contracts, adoption is still hindered by several obstacles, including security risks, weaknesses, and legal concerns. In this paper, we present a comprehensive survey of block chain-enabled smart contracts for bank applications for Mint money into your account, Withdraw money from your account, Send money from your account to a smart contract address, and Check your balance.

KEYWORDS:

BLOCKCHAIN, CHECK BALANCE, ETHER, REMIX ETHEREUM, SMART CONTRACT, SOLIDITY, TRANSACTION.

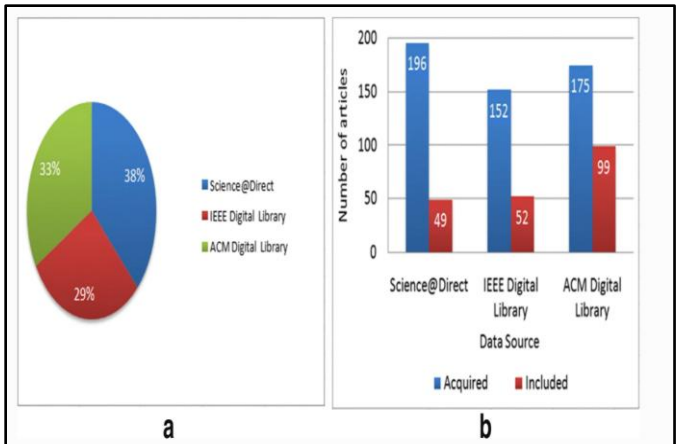
INTRODUCTION

The block chain has been around for more than ten years and is a system where a distributed database records every transaction that takes place in a peer-to-peer network. It is viewed as a distributed computing paradigm that effectively resolves the trust in a centralized party issue. As a result, multiple nodes work together in a block chain network to secure and maintain a set of shared transaction records in a distributed manner without depending on any third parties. The first proposed crypto currency that used the block chain as a distributed infrastructure technology was launched by Satoshi Nakamoto in 2008 with the release of Bit coin. Without the need for a centralized authority, it allowed users to securely transmit virtual money called "bit coins.". Aside from those, block chain-based platforms for crypto currency were also suggested, including Ethereum, NXT, and Hyper ledger Fabric. They can employ smart contracts, unlike Bit coin (SC). Because smart contracts automate the execution of contracts in a distributed environment when certain criteria are satisfied, block chain technology outperforms traditional contracts by adding the terms of agreements between two or more parties.

Without the assistance of a reliable third party, smart contracts enable, carry out, and enforce agreements between unreliable parties. They run on top of the block chain. Network automation and the capacity to convert paper contracts into digital ones were made possible by smart contracts. In contrast to traditional contracts, smart contracts provided automated transactions without the oversight of a central authority, enabling users to formalise their agreements and trust relationships. Smart contracts are copied to each node of the block chain network to prevent contract manipulation. Human error could be eliminated to prevent disputes involving such contracts by permitting the execution of operations by machines and using the capabilities offered by block chain platforms.

LITERATURE OVERVIEW

The adopted research approach, including the search strategy, the filtering procedure, and the inclusion and exclusion criteria, is described below. In addition, we provide the taxonomy used to classify the papers that were finally included. Utilizing the "smart contract" string term, we searched three already-existing databases: ScienceDirect, IEEEXplore, and the ACM Digital Library for pertinent articles. As indicated in Fig. 1a, which shows the percentage of research papers acquired from each digital database, and Fig. 1b, which shows the total number of preliminary studies collected from each digital database, we discovered 523 publications in the first phase.

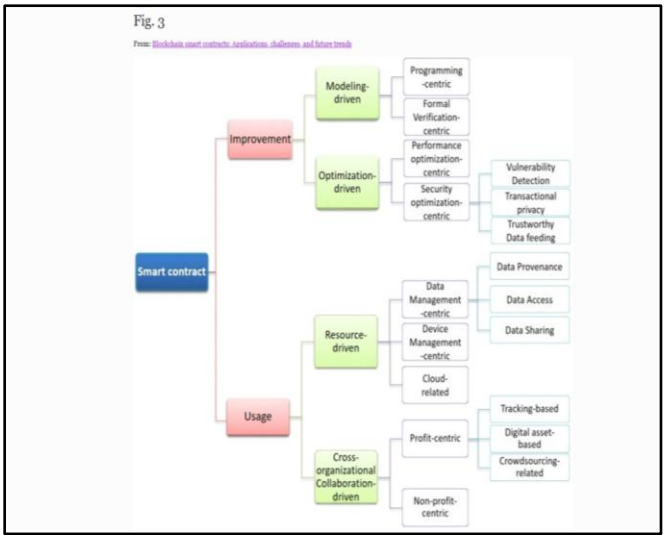


We screened the primary research retrieved from the databases in order to select the pertinent publications to be examined in our review. In order to achieve this, we created a list of inclusion and exclusion criteria, which are summarized in Table 2. We used a set of inclusion and exclusion criteria based on the findings of the first phase to eliminate articles that were deemed outside the purview of this review. As a result, we only considered studies that met all the criteria for inclusion. By selecting research based on the title, abstract, and keyword list, we were able to eliminate duplicate publications, surveys, and literature reviews. The filtering procedure resulted in the exclusion of 323 publications and the inclusion of 200 pertinent publications for this systematic review. The number of pertinent studies from each digital database that were used in this research is also shown in Figure 1b.

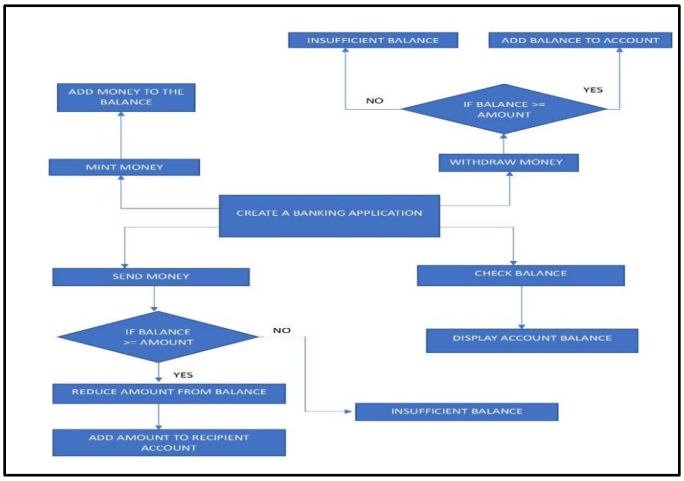
As a result, we only gave consideration to papers that satisfied all inclusion requirements. We were able to remove duplicate articles, surveys, and literature reviews by choosing research based on the title, abstract, and keyword list. 200 relevant publications for this systematic review were included after 323 publications were excluded as a result of the filtering process. Figure 1b also displays the number of pertinent studies from each digital database used in this study. Fig. 2 shows the number of included studies published each year from 2015 to September 2020 to assess the progress of the smart contract sector in terms of publication date. We see that there have been more papers published overall in the field under study during the last few years, which shows the subject's prominence. As a result, the field of smart contracts has been expanding quickly lately. A thorough taxonomy is created as a consequence of an in-depth study of the research contained in this review, which will help designers better understand the numerous factors they must take into account when creating a smart contract. This survey was conducted primarily for the purposes of identifying I the key publications on smart contracts, (ii) the present state of research in this area, and (iii) any potential gaps in the literature that might become research issues that the scientific community should address. Without involving a reliable third party, smart contracts have arisen as a novel and promising way to create fully decentralized apps. Despite the positive aspects of smart contracts, adoption is still hindered by a number of obstacles, including performance problems, security risks, and privacy concerns. The fact is that emerging smart contract applications have higher requirements for contract execution time, cost, security, and privacy..We hope to define the conceptual underpinnings of block chain-enabled smart contracts through this poll, but we also hope to highlight potential study topics for future studies. In fact, we divide the existing research on smart contracts into two main groups: smart contract enhancement and smart contract application. Studies addressing issues with smart contracts, such as performance, vulnerabilities, and a lack of reliable data feeding, are included in the first category. The latter includes research aimed at using smart contracts to solve problems particular to a given domain. The proposed taxonomy of block chain-enabled smart contracts is shown in this figure, which also includes the use of modeling-driven, resource-driven, optimization-driven, and cross-organizational collaboration-driven smart contracts.

REMIX ETHEREUM

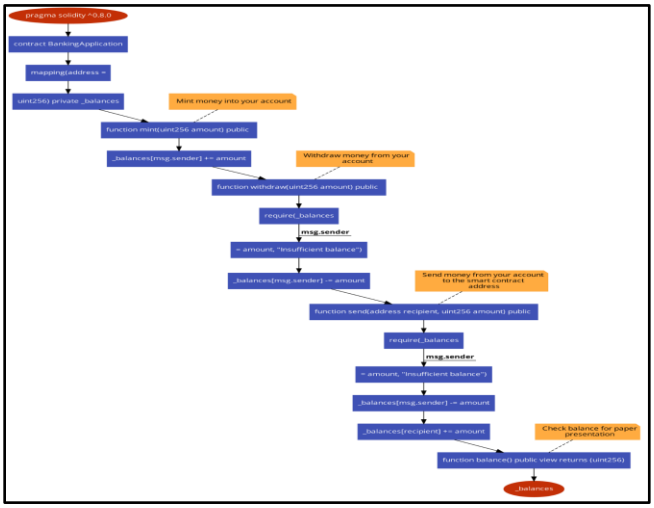
Remix Ethereum streamlines the development process by offering an intuitive interface that allows developers to write, test, and debug smart contracts. With its user-friendly environment, developers can efficiently navigate through various features and functionalities. Remix Ethereum provides a real-time feedback mechanism, highlighting potential coding errors and suggesting improvements, thus minimizing the risk of bugs or vulnerabilities in smart contracts. This simplification of the coding process enables both experienced developers and newcomers to create robust and secure block chain applications with ease.



BLOCK DIAGRAM:



CODE FLOWCHART:



SOLIDITY CODE:

```
// SPDX-License-Identifier: GPL-3.0
pragma solidity ^0.8.0;

contract Banking Application {

    mapping(address => uint256) private _balances;

    // Mint money into your account
    function mint(uint256 amount) public {
        _balances[msg.sender] += amount;
    }

    // Withdraw money from your account
    function withdraw(uint256 amount) public {
        require(_balances[msg.sender] >= amount, "Insufficient balance");
        _balances[msg.sender] -= amount;
    }

    // Send money from your account to smart contract address
    function send(address recipient, uint256 amount) public {
        require(_balances[msg.sender] >= amount, "Insufficient balance");
        _balances[msg.sender] -= amount;
        _balances[recipient] += amount;
    }

    // Check balance for paper presentation
    function balance() public view returns (uint256) {
        return _balances[msg.sender];
    }
}
```

PROCEDURE:

Step1: The first line of the code is the SPDX license Identifier. It is a unique address that every smart contractor who needs to work on solidity must get it. **Step2:** "pragma solidity ^0.8.0;" specifies the version of Solidity used in the contract. The "Bank" contract is defined to hold the mapping of each user's account balance.

Step3: The "mint" function allows a user to add funds to their account. When a user calls this function and sends Ether with it, their account balance is increased by the amount sent.

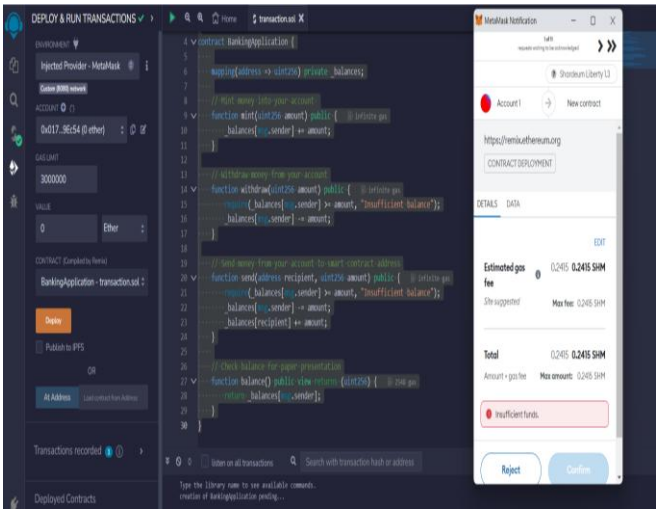
Step4: The "withdraw" function allows a user to withdraw funds from their account. It checks that the user has enough funds to withdraw, then transfers the funds to the user's address and updates the current status of the balance.

Step5: The "send" function allows a user to send funds to another address. It checks that the user has enough funds to send, then transfers the funds to the recipient's address.

Step6: The "getBalance" function returns the balance of the user's account.

HIERARCHY:

Go to https://remix.ethereum.org/. Create a new file and name that with the extension of "sol". Add MetaMask extension to Chrome. Login and sync the MetaMask account with your Remix. Write a smart contract code and RUN it. If it shows Green Tick on the Solidity compiler, it is an error-free code. Otherwise, the code contains an error. Please, rectify your code and run it. Once you have finished this process, you have created the code successfully. Next, go to deploy and run the transaction icon on the left side of the remix Ethereum. Change the Environment section to "Injected Provider-MetaMask". Set the default gas limit to 3000000. Change the value section to Ether with a minimum number. Once you finished the above steps "Deploy" the code. A new MetaMask tab will open on your screen and shows the transaction details. The below Output shows "Insufficient Funds" because we didn't purchase any ether. Suppose we had purchased the Ether and did a transaction. The MetaMask tab shows the transaction details.



CONCLUSION

Smart contracts' decentralisation, auto-enforcing capacity, and verifiability allow their encoded business rules to be carried out in a peer-to-peer network where each node is "equal" and none has any special authority without the need of a centralised server or trusted authority. Thus, it is anticipated that smart contracts would alter a number of established sectors, including finance, healthcare, energy, and others. In this paper, we presented a smart contract for a banking application using solidity in remix Ethereum. Thus, we introduced a banking application for creating mint money, viewing balance, sending money, and withdrawing money from the account and discussed the existing smart contract-based studies. Both smart contract difficulties and unresolved problems are noted for further research based on survey results. Finally, we talked about potential developments for smart contracts. This study offers stakeholders interested in the study of smart contracts informational support.

RESULT:

Thus it is provided Solidity smart contract that allows for the creation of a wallet with functionalities to mint tokens, send money, check balances, and deposit funds using remix Ethereum was done and executed successfully and the deployment of the code was done using the MetaMask extension and user's transaction details was displayed in the output.

REFERENCES

1. *Analytics TC Ripple xrp continue to revolutionize cross border payment systems. Available online at https://thecurrencyanalytics.com/11696/ripple-xrp-continue-to-revolutionize-cross-border-payment-systemshttps://thecurrencyanalytics.com/11696/ripple-xrp-continue-to-revolutionize-cross-border-payment-systems (2020). Last accessed: 2020-10-03*

2. *Ellul J, Pace GJ (2018) Alkylvm: A virtual machine for smart contract blockchain connected internet of things. In: 2018 9Th IFIP International Conference on New Technologies, Mobility and Security (NTMS), pp 1–4.*

3. *Angelo MD, Salzer G (2019) A survey of tools for analyzing ethereum smart contracts. In: IEEE International Conference on Decentralized Applications and Infrastructures, DAPPCON 2019, newark, CA, USA, April 4-9, 2019, IEEE, pp 69–78*

4. *Dickerson T, Gazzillo P, Herlihy M, Koskinen E (2019) Adding concurrency to smart contracts. Distrib Comput 33:1–17*