



Pioneering Futures Since 1898

Module code: CN7051

SCHOOL OF ARCHITECTURE, COMPUTING & ENGINEERING

Blockchain and Cryptocurrency

Prof. Arish Siddiqui

University of East London

04 May 2022

By

Aekkaraj Kuplakatee – u2235955

Stevin Sam – u1802868

Webster Osakpamwan IMASUEN - u1423795

Abstract

In recent years, blockchain has developed into an innovative and secure platform for the secure exchange of data between organizations and individuals across many application areas including finance, supply chain management, food industry, energy industry, a wide range of internet-connected devices, and healthcare. An assessment of the existing literature both on the usage of blockchain technology in the healthcare system as well as the applications at hand is presented in this report.

The Ethereum Blockchain platform has been used to design and implement the medical workflow involving prescriptions between healthcare providers, pharmacies, and patients. In addition, medical data can be accessed and managed using this platform.

Keywords: blockchain technology; smart contracts; healthcare; data exchange; secure; distributed ledger technology

Table of Contents

Abstract.....	2
1. Introduction	4
2. Literature Review.....	5
2.1 The Blockchain Technology: A Background Concept	5
2.2 An overview of blockchain applications in healthcare	5
2.3 The potential of blockchain technology in the healthcare industry.....	7
3. Design.....	8
3.1 Technologies used	8
3.2 Functional Requirements:	9
3.3 Events and Procedures:	10
3.4 Methodology for estimating costs	11
3.5 Development of a blockchain-based system application.....	11
4. Theoretical Implementation	13
4.1 Smart Contract Implementation:	13
4.2 Testing	15
5. Conclusion	17
6. Individual reflection and discussion	17
6. Gantt Chart	18
7. Reference:	19
8. Appendix	20
8.1 Code:.....	20
8.2 Presentation:	23
8.3: Agreement of Participation – Group Assignment One CN7051	27

1. Introduction

Several research studies have identified the potential of blockchain technology in the healthcare ecosystem, making it a key technology in the digital revolution of the healthcare industry. As technology develops, it will lead to a complete transformation of how businesses and traditional medical systems engage in the healthcare industry. The establishment of an IT-based healthcare ecosystem is crucial to the decentralization and digitization of healthcare.

Blockchain technology represents one of the best alternatives to traditional methods for decentralizing healthcare services. It enables a modern and digitized healthcare ecosystem that benefits both patients and providers. Blockchain technology, as the chain of blocks, is seen as the future of internet data management. It provides utilities that will greatly enhance consumer and provider experience by identifying and eliminating errors with data management, providing record access and control, managing claims, payments, and controlling the overall security of IoT, as well as the verification and exchange of research data for auditing and transparency within the organization.

A decentralized blockchain ledger that is encrypted and encrypted in real-time is used for monitoring, controlling, and understanding medical data in these applications. In addition to this, this facilitates the restriction of sensitive information from being accessed by unauthorized persons.

In addition to the many processes involved in health care administration, there are also financial management, patient care, legal issues, transportation, inventory, and many other related activities. As a rule, medical workflows can be plotted as a sequence of conditional steps that can be fulfilled to perform repetitive tasks in relation to the patient's treatment. In hospitals and other healthcare facilities, these systems, which are usually called e-health or EMR systems, are designed to provide improved internal controls and productivity, as well as reduced risk, work cycles, and overhead. This paper presents a medical workflow application that is designed for one specific domain of healthcare management.

Our project presents a system for simplifying complex medical processes between a physician and a patient through a healthcare smart contract. Research on blockchain technology in healthcare was discussed, as well as technology based on Ethereum for healthcare management. Furthermore, the paper shows the challenges and potential directions blockchain research can take as it pertains to healthcare and illustrates the potential uses of blockchain. We included research on healthcare solutions, algorithms, methods, methodologies, or architectures that are specific to healthcare in this systematic review.

A brief background description of blockchain technology is given in Section 2, followed by a review of the related works in Section 3. In Section 4, blockchain technology is discussed in terms of what it can do. Section 5 of this paper discusses the design and developing the codes for the System. Section 6 of this paper summarizes the paper's overall discussion. It is concluded in Section 7.

2. Literature Review

2.1 The Blockchain Technology: A Background Concept

A brief overview of Blockchain Technology Blockchain is a technology for securing transactions and data that is based on peer-to-peer networks where each peer is responsible for maintaining their own ledger. Neither centralized management of data storage nor centralized administration is required with this technology. Replication and encryption keep the data quality high after it is spread across several nodes. Blockchain was born on 31st October 2008, when Satoshi Nakamoto wrote a white paper explaining the concept. To solve this problem, he devised a platform that would allow cryptocurrency transactions through peer-to-peer, without going through any third-party financial institutions, to send online payments directly from one to another.

In essence, his main idea was for a peer-to-peer distributed ledger to solve the double-spending problem using a mathematical model that could prove the chronological order in which transactions were carried out. In computing terms, blockchain refers to the sequential arrangement of blocks of data that consist of successive layers of information about a system's past, present, and future. Whenever a new block is added to the system to be a part of the chain, it plays an important role in connecting it with the previous block, and then connecting it with the following block. As a rule of thumb, each block has the responsibility of recording, validating, and distributing all transactions among its neighbouring blocks. As a result, it means a block in the chain cannot be removed or modified since this would cause the block next in the chain to be altered.

This is where a blockchain network comes into play; it is a decentralized system that stores an archive of all past transactions and enables it to function according to a predefined set of rules. Among these rules are those that define the directions of performing and validating the transactions, as well as how the network and the members function together as a whole. The data for a particular network is usually stored on each node of the individual network of the network in question, and as such is usually called a distributed registry.

Blockchain networks combine a group of transactions into blocks of transactions linked together with a hash of the previous block's records together with the hash of the next block's record connected to the previous block. It is therefore imperative that, because of this property of immutability, blockchain networks are enforced with a basic security mechanism. As a block (and the data included in it) gets older, it becomes more protected from changes.

As a result of differences in hash function algorithms, an attacker attempting to alter any key would result in the local register ceasing to function because the hash values stored inside the next block headers would be completely different.

2.2 An overview of blockchain applications in healthcare

Medical and healthcare systems that are based on legacy systems usually do not share resources outside the healthcare field. Despite the evidence in favour of the integration of these networks, interconnected healthcare providers and a better patient experience are still challenges. This calls for a closer collaboration between health information researchers. Interorganizational data exchange is a critical issue in this regard, as it requires healthcare providers' medical data to be accessed easily by institutions such as physicians or research institutions. Blockchain technology is

fast becoming a fundamental part of how healthcare data is processed and governed. A lot of this rests on its ability to adapt and segment medical data and services in a way that is unprecedented, to be secure and to be shared. Several recent advancements in healthcare are attributable to Blockchain technology, which has many innovative developments in that field. Using advances in e-health data, cloud storage, and patient data protection regulations, new possibilities are being created for managing health data, as well as improving patient convenience in accessing and sharing health data. Any organization that relies on data is aware of the importance of storing and transferring information securely and managing the smooth integration of information. Those in the healthcare sector are especially concerned with these issues, which blockchain technology may assist in addressing in a flexible and efficient manner.

As part of this study, we take a detailed look at blockchain-based applications related to data sharing, data management, data storing, and electronic health records. Blockchain based innovations are emerging in healthcare in recent years, and can be conceptually divided into several layers, depending on whom you ask, ranging from data sources to blockchain technology, healthcare applications, stakeholder groups, etc.

In Gordon and Catalina's review titled "Trustworthy Information in the Healthcare World - The Case for Blockchain," they discuss how Blockchain tech enables patients to manage their healthcare records without institutional interference, as opposed to institutionally controlled data sharing. The authors from the University of British Columbia investigated how blockchain technology can transform the healthcare sector by enabling digital access rights for patients, allowing them to identify themselves electronically across the network, as well as handling large amounts of healthcare data and ensuring data integrity.

To send medical records to the Hyperledger blockchain network, Daisuke et al used a blockchain platform based on Hyperledger fabric. Mobile devices were used to collect the medical information. By registering healthcare data to the Blockchain, they were aiming to make sure it is secure.

Using blockchain to manage health information efficiently was the subject of Anurag et al.'s study. In their study the authors cover various types of research, as well as many types of investigations and discussions. Nonetheless, most of their discussions were about blockchain technology in healthcare, both its potential benefits and limitations, without providing any proof or system evaluation. After discussing the potential uses of blockchain for the management of health care records on a cloud-based system, they came up with various ways that blockchain could better serve this function. At the same time, they could control security and privacy of patient data while using this platform.

In order to address limitations of permissioned versus permissionless blockchains, Rouhani et al. proposed an approach. Their patient-controlled healthcare data management system is based on the Hyperledger platform.

In a literature review, Wu and Tsai presented two algorithms to secure network traffic in healthcare management systems. As well as developing a distributed system to manage health data, they also suggested setting up regulations for how the health data should be used.

Using a peer-to-peer network and blockchain technology, Shen, et al., have presented a new way to share medical data by utilizing MedChain, a peer-to-peer network and blockchain technology. A

healthcare data system has been designed for the collection of patient data from various sources, from medical examinations and point of care sensors to mobile apps and IoT sensors.

2.3 The potential of blockchain technology in the healthcare industry

Blockchain technology can provide substantial benefits for researchers, health care providers, and individuals alike. Creating a central location for the storage of all health data, tracking individual data in real-time, and setting individual permissions to access data would benefit both research and personalized medicine.

The growing need for comprehensive data sets that researchers and health care professionals can use to advance understanding of diseases and disease progression, accelerate biomedical discoveries, understand the effects of new drugs more quickly, and design individual treatment plans based on genetics, life cycle, and environment is a growing trend. A shared data system of Blockchain would provide a wide range of data sets since it would cater to patients from different ethnic groups, socioeconomic backgrounds, and geographical locations. As a result, blockchain offers a significant amount of data for longitudinal studies since it collects a person's health history throughout his or her lifetime.

By using a blockchain, health care data can be collected from a wide range of individuals who either aren't traditionally involved in science or are underserved by the healthcare community.

Using blockchain technology is a great way to introduce "hard-to-reach" audiences to the power of the blockchain, resulting in a higher accuracy rating from the general public as a result of the shared data environment of the blockchain.

It has been suggested that new apps may emerge that circumvent medical research and tailor treatments to individual patients in light of the introduction of a health care blockchain. Patient and provider can discuss research-based treatment options in a cooperative, educated manner as opposed to relying only on intuition.

3. Design

3.1 Technologies used

- ❖ **Node:** Node Package Manager
- ❖ **Truffle:** Truffle is a development environment allows for compilation, linking, deployment and management. It allows us to test and deploy our smart contract to the blockchain. It also provides us an environment to develop a client-side decentralised application.
- ❖ **Ganache :** Ganache is a personal Ethereum Blockchain that is used to test and deploy contract, as well as develop applications without any cost.
- ❖ **MetaMask** is a software cryptocurrency wallet that is used to interact with the Ethereum blockchain
- ❖ **Solidity IDE – Visual Studio Code / Remix**

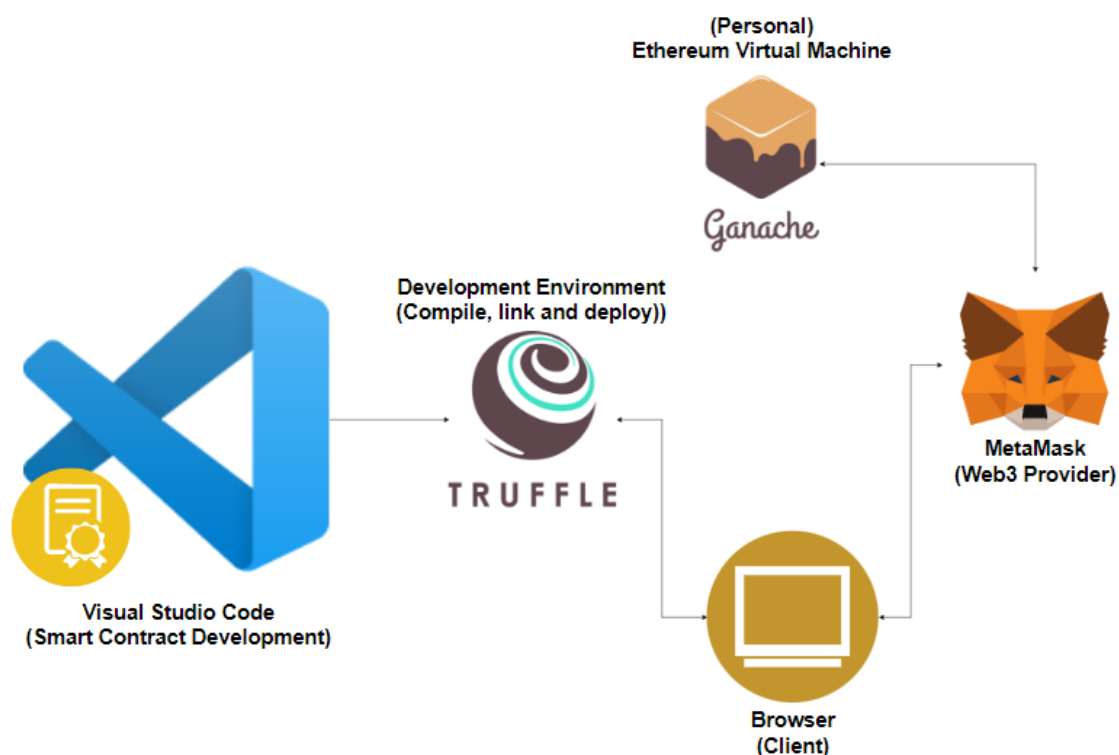


Figure 1: How the different technologies interact

The Solidity IDE (Visual Studio Code / Remix) would be used to write the smart contract and all the relevant scripts, then we would use the truffle development environment to test the compilation and deployment of the smart contract. Finally, we will use a combination of Ganache and MetaMask to simulate a blockchain and a crypto wallet to interpret transactions with the smart contract.

3.2 Functional Requirements:

Our blockchain smart contract system has been used to design and implement different medical workflows involving specific medical procedures. For example, basic medical prescriptions and their procedures have been created using our smart contracts. Medical smart contracts are designed to overcome administrative inefficiencies by facilitating communication between patients, doctors, and pharmacies. In addition to retrieving, analysing, and managing medical data and procedures, this system will also help manage complex healthcare data.

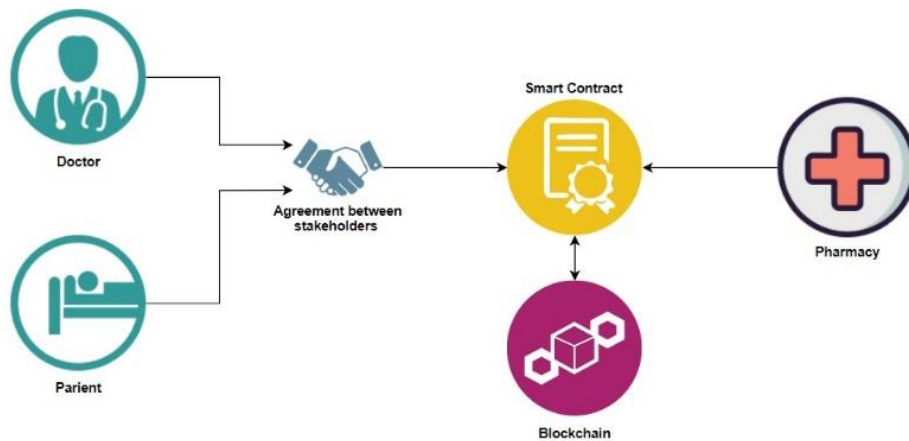


Figure 2: The basic design of the implementation

The main requirements are:

- ❖ Adding Prescriptions
- ❖ Removing Prescriptions
- ❖ Adding Doctors
- ❖ Removing Doctors
- ❖ Having a valid doctor

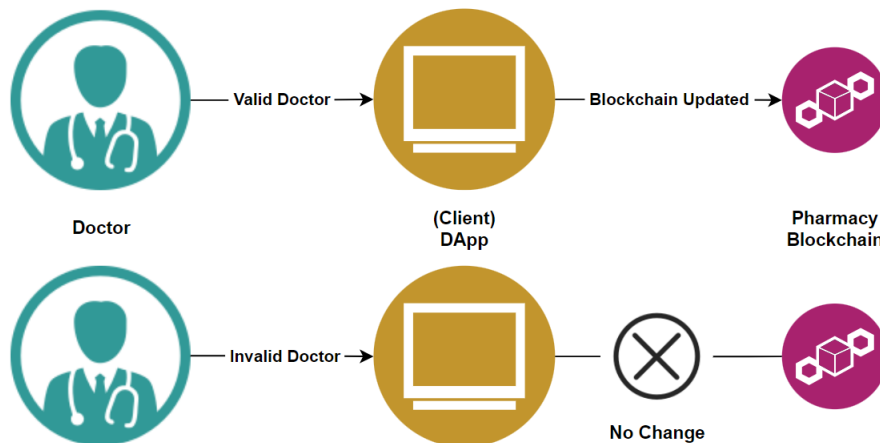


Figure 3: Doctor interaction validation

The outcomes that are shown above in Figure 3 highlights that without validation from the contract, a person cannot impersonate a doctor to prescribe medicine, this is a security feature that would prevent unlawful distribution of medicine. Therefore, only valid doctors have authority to issue a prescription.

3.3 Events and Procedures:

The system should reduce errors associated with physician misinterpretations by eliminating long waiting times, eliminating fraud, and reducing rates of physician misinterpretations. By using smart contracts, doctors can add prescriptions to the patient's healthcare record. Through the Ethereum blockchain, the pharmacy then gains access to the prescription through permissions granted by both the physician and patient. Upon receiving the prescription, the pharmacist dispenses the medication with a counter-signed prescription and with a smart contract, which then populates the patient's healthcare records with the medicine's expiration date and dosage. After this is done, the patient can pick up their medication.

As a result of smart contracts, doctors and pharmacies tend to share information regarding patient care. The average time that doctors spend with drug stores, and more specifically with the pharmacists after patients' visits, is four times less than they spend explaining medicines request. According to the diagram in Figure 4, prescribing a medicine is an agreement between doctor and patient, in which the smart contract is deployed with each users trust.

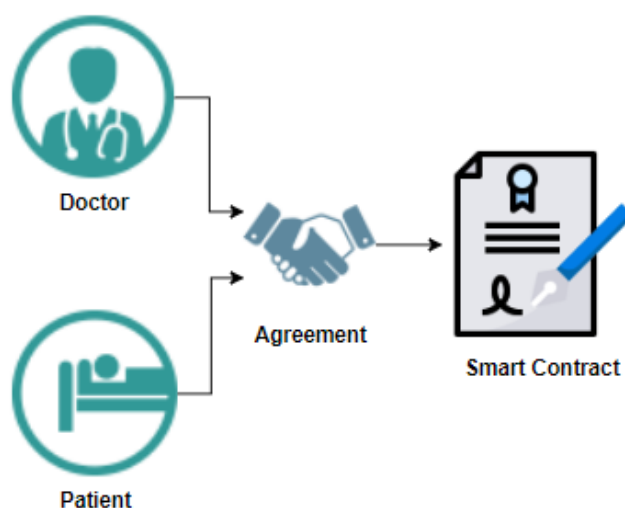


Figure 4. Issuing and submitting prescriptions through a smart contract.

Moreover, with the use of decentralised ledger technologies for a pharmacy-based smart contract, it would allow for greater transparency that can be easily examined to validate any changes that occur within the blockchain. And due to its decentralised nature of the blockchain, the integrity and availability of the data is guaranteed.

[We have also enclosed below the demonstration in bytecode of the Issuing and submitting prescriptions through a smart contract. (See the rest at the appendix section.)]

3.4 Methodology for estimating costs

A cost estimate for deploying smart contracts in the healthcare industry will be necessary before deploying the medical blockchain. By designing a blockchain-integrated medical health system, we aim to create a solution that is truly feasible. To avoid network abuse and overcome other computational problems on the Ethereum network, all programmable transactions cost some fees. To run every kind of transaction on Ethereum blockchain, a fee is set up as gas. For transactions or contracts executed on the Ethereum Blockchain platform, gas is the price value that is needed for a transaction to be successful. In other words, the network miners are able to determine the gas price, and they have the capability to reject transactions if the price of gas is out of step with their capacity limits.

3.5 Development of a blockchain-based system application

A decentralized application (DApp) will be implemented as a framework that supports a private blockchain network deployed by Ganache at the back end. Smart contract systems for healthcare have been implemented using Ethereum. With an active community and a repository of public DApps. Developers are planning to switch to a proof-of-stake consensus algorithm called PoS soon, but presently the platform uses a proof-of-work consensus procedure called Ethash.

The implementation of distributed applications should theoretically make use of a consensus algorithm such as the Delegated Proof-of-Stake (DPOS) or Practical Byzantine Fault Tolerance (PBFT). DApps can detect irregularities, unauthorized additions of data, and misplaced entities by comparing DFS content with ledger records. A timeline is provided for auditing each step.

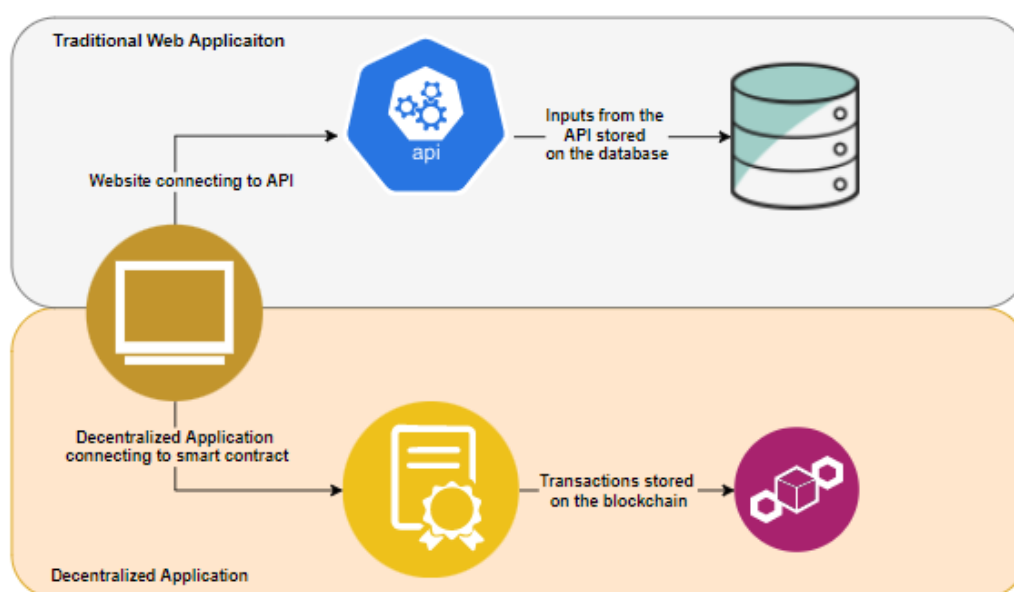


Figure 5. Blockchain-based application interaction compared to normal web applications

Solidity is a high-level programming language that is used to devise smart contracts, which include functions, events, state variables, and modifiers. This smart contract has been deployed on the Remix and Ganache test network, along with MetaMask for paying the transaction fees in the network.

The process of creating smart contracts involves three steps: writing, compiling, and then broadcasting the contract by using the Solidity language.

The Solidity runtime compiler produces bytecode for smart contracts. Ethereum Wallet such as MetaMask is used to announce smart contracts on the blockchain. As soon as the user submits a request or inquiry through the interface, the smart contract pulls the data they requested from the web, embeds it into a transaction, deploys it to the blockchain, then sends that transaction back to them. It is reported in a Block that the transaction status has been updated.

4. Theoretical Implementation

4.1 Smart Contract Implementation:

Figure 6 shows the beginning of the smart contract, and so it starts by defining the 'EPrescription' contract and inputting the various essential features of the smart contract that needs to be used for interaction. First we define the 'owner' and 'prescriptionId', these would be used as variables that can be easily accessible throughout the code. Then there is the strut, creating a 'PrescriptionForm' record, that can be to create a custom data type in the form of a structure with the associated properties inside. In our example, we defined the relevant assets to define a valid prescription.

Then we mapped the relevant data structure to a public getter function called 'prescriptions' that can be used to return the details of the 'PrescriptionForm'

```
1  pragma solidity >=0.8.0;
2  pragma experimental ABIEncoderV2;
3  // SPDX-License-Identifier: MIT
4
5
6  contract EPrescription {
7
8      address owner;
9      uint256 public prescriptionId;
10
11     struct PrescriptionForm {
12         address GP; //Address of doctor that issued this prescription
13         address Patient; //Address of Patient that receives this prescription
14         string PIPcode; //A unique seven-digit coding system used to ensure traceability of medication
15         string medicationName; //Scientific name of the medicine
16         uint256 dosage; //quantity per pill
17         string dosageUnit; //Unit of the dosage ex. mg, ml and etc
18         uint8 numPills; //quantity of pills to give in the prescription
19         uint8 repeat; // quantity of time to take the pill (ex. 2 means two times a day)
20         string spicificTime; // time to take the pill (ex. in the morning, afternoon, evening)
21         string BeforeOrAfter; // take the pill before or after meal
22         uint256 date; //Epoch time when the prescription was given
23         uint256 expirationTime; //Epoch expiration date (When is this prescription no longer valid)
24     }
25
26
27     mapping(uint256 => PrescriptionForm) public prescriptions; //Mapping to Prescription struct
28     mapping(address => bool) public GP; //Mapping the doctor address
29     mapping (address => bool) public chemists; //Mapping the chemist address
30
31     constructor() {
32         owner = msg.sender;
33     }
34 }
```

Figure 6. Smart Contract 'EPrescription' implementation (1)

Figure 7 details the function that is used to add a prescription that uses the various inputs that it receives from the mapped 'prescriptions' to return an output on the 'PrescriptionForm' structure.

```
function AddPrescription(
    address _Patient,
    string memory _PIPcode,
    string memory _medicationName,
    uint8 _dosage,
    string memory _dosageUnit,
    uint8 _numPills,
    uint8 _repeat,
    string memory _spicificTime,
    string memory _BeforeOrAfter,
    uint256 _date,
    uint256 _expirationTime) public GPisApproved(msg.sender){
    prescriptions[prescriptionId] = PrescriptionForm(
        msg.sender,
        _Patient,
        _PIPcode,
        _medicationName,
        _dosage,
        _dosageUnit,
        _numPills,
        _repeat,
        _spicificTime,
        _BeforeOrAfter,
        _date,
        _expirationTime);
}
```

Figure 7. Smart Contract 'EPrescription' implementation (2)

```

function approved(address _GPToApprove) public onlyOwner {
    GP[_GPToApprove] = true;
} //GPToApprove uint256 ID of doctor to approve for giving prescriptions

function remove(address _GPToRemove) public onlyOwner {
    GP[_GPToRemove] = false;
} //GPToRemove uint256 ID of doctor to remove for giving prescriptions

function permissionPharmacy (address _pharmacist) public view returns (bool) {
    return chemists[_pharmacist];
} //given permission to chemist

```

Figure 8. Smart Contract 'EPrescription' implementation (3)

Figure 8 is the various validation functions that make sure the prescription that is being transacted is done by a valid doctor (*Similar to Figure 3*). In which the wallet address of the doctor has been already approved before a transaction has been made, otherwise the contract will fail. Moreover, in order for the chemist to view the valid prescriptions that are available, their address also has to be validated.

```

function cancelPrescription() public view GPIsApproved(msg.sender){
    require(prescriptions[prescriptionId].GP == msg.sender);
}

function giveMedication(address _pharmacist) public view {
    require (permissionPharmacy(_pharmacist) == true);
}

```

Figure 9. Smart Contract 'EPrescription' implementation (4)

Figure 9 is the basic functionality of the 'cancelPrescription' function is to remove a valid prescription already within the blockchain. And the 'givenMedication' is a function called when the chemist interacts with the patient and validates that the prescription has been given and requires the chemist's address has already been approved.

```

modifier onlyOwner {
    require(msg.sender == owner);
    _;
}

modifier GPIsApproved(address _GP) {
    require(GP[_GP]);
    _; //Guarantees that address belongs to a valid and approved doctor
}

modifier hasNotExpired(uint256 _prescriptionId) {
    uint256 timeToExpiry = prescriptions[prescriptionId].expirationTime - block.timestamp;
    require(timeToExpiry > 0);
    _; //Guarantees prescription is not being used after its expiration date
}

modifier onlyPrescribedPatient(uint256 _prescriptionId) {
    require(prescriptions[prescriptionId].Patient == msg.sender);
    _; //Guarantees msg.sender is patient who was actually prescribed
}

```

Figure 10. Smart Contract 'EPrescription' implementation (5)

Figure 10 highlights the different modifier that change the behaviour of a function and is used to as a prerequisite to an existing function; acts like an if statement that validates a condition that has been executed.

4.2 Testing

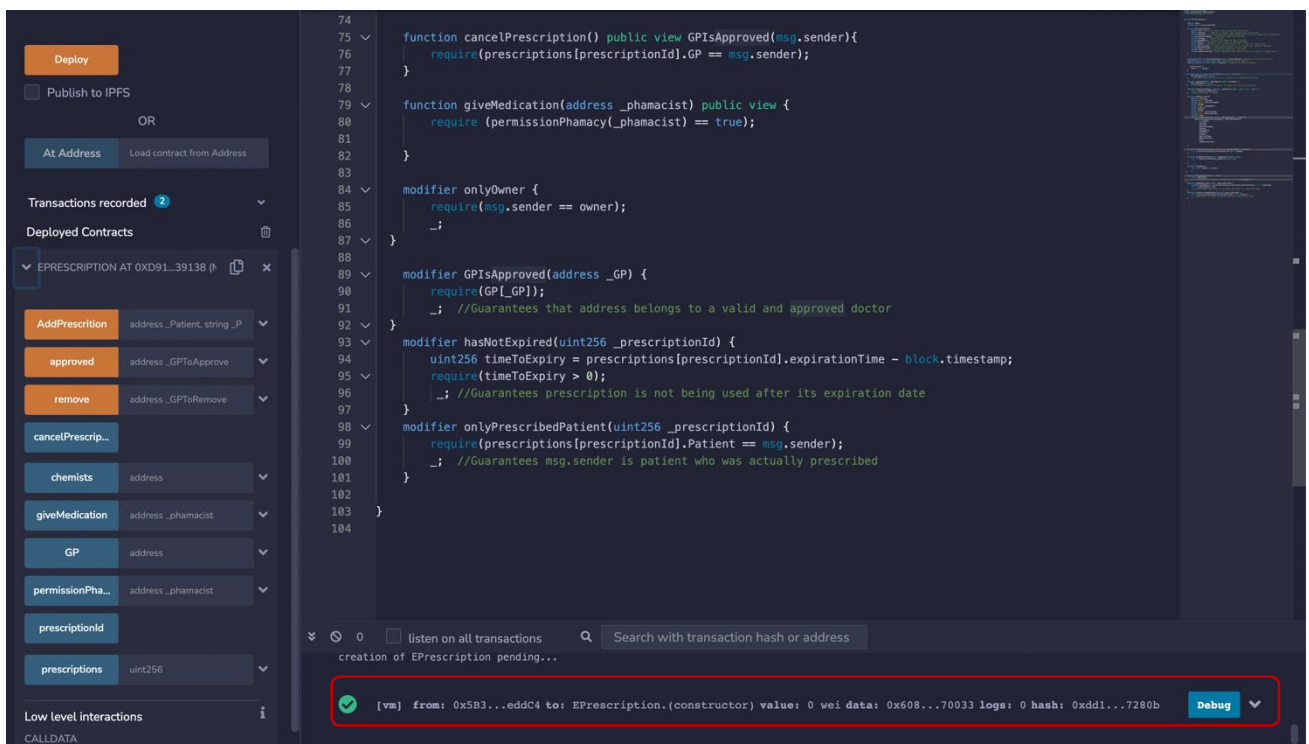


Figure 11. The smart contract is successfully deployed on the remix server

Figure 11 shows that the 'EPrescription' smart contract has been successfully deployed within the Remix IDE environment and can now interact with MetaMask to validate transactions. Then we tested to check if the wallet address of a user can be validated to be the doctor, which has been successfully deployed as shown in Figure 12. This can allow for a more secure validation function that is a core part of our smart contract, interacting with the 'PerscriptionForm'.



Figure 12. The wallet address is successfully approved to be the doctor

AddPrescription

_Patient:

0xcbf9c446deCD1AA6499B;

_PIPCode:

111110

_medicationName:

Paara

_dosage:

8

_dosageUnit:

G

_numPills:

8

_repeat:

2

_spicificTime:

MORNIG

_BeforeOrAfter:

BEFORE

_date:

1651670423

_expirationTime:

1651670423

transact

Contract Deployment

Status

Confirmed

View on block explorer

Copy Transaction ID

From

0xcbf...20ff

To

New Contract

Transaction

Nonce

3

Amount

-0 ETH

Gas Limit (Units)

72281

Gas Used (Units)

72281

Gas price

20

Total

0.00144562 ETH

£3.28 GBP

+ Activity log

+ Transaction data

At Inversebrah

Status

Confirmed

View on block explorer

Copy Transaction ID

From

0xcbf...20ff

To

New Contract

Transaction

Nonce

8

Amount

-0 ETH

Gas Limit (Units)

1399365

Gas Used (Units)

1399365

Gas price

20

Total

0.0279873 ETH

£63.41 GBP

+ Activity log

+ Transaction data

✓

[vm] from: 0x5B3...eddC4

to: EPrescription.AddPrescription(address,string,string,uint8,string,uint8,uint8,string,string,uint256,uint256)

0xDA0...42B53

value: 0 wei data: 0x0eb...00000 logs: 1 hash: 0x46c...703cb

status

true Transaction mined and execution succeed

Figure 13. Successfully deploying the 'AddPrescription' function

After the doctor has been validated, they are able to prescribe medication to pateints using the 'Addprescription' function. As shown in Figure 13, the function is deployed effectively and takes relevant inputs for prescription and then charges a gas fee to write to the blockchain.

5. Conclusion

Providers, laboratories, payers (insurance companies) and drug companies are all required to store data in a wide variety of formats to manage patient data in traditional healthcare systems; patient records are also maintained in multiple formats. As we see today in the discussion of health records, we have a lot of problems between the lack of data and the general disarray. Research into public health and drug discovery have also been hampered by insufficient data sharing. A common standard has been pushed throughout the ecosystem to tackle this problem. During this coursework, we looked at the current requirements in healthcare, examined the drawbacks of the existing system, and discussed Ethereum-based solutions for the administration of healthcare systems. The problem of implementing adapted medicine and how the system we have designed offers solutions are discussed. We provide an overview of the present state of personalized medicine.

Our study demonstrated that smart contracts could automate complicated medical procedures by utilizing blockchain technology, which is a decentralised method of managing medical data. With smart contracts, medical records are auditable, interoperable, and accessible. Flexible, granular control is maintained over the system. Medical researchers can also use the system to share patient data, as well as receive incentives to use it. This could be a potential use for blockchain in health care. A blockchain-based healthcare system can improve patient outcomes and processes. Through smart contracts, blockchain reduces transaction costs, reduces administrative burdens, and eliminates middlemen. Iterative, scalable, secure, accessible, and decentralized healthcare ecosystem can be created with blockchain technology. It eliminates the siloes of data, the inconsistencies of legacy networks, unstructured data collection difficulties, prohibitively high administrative costs, and privacy concerns of the current healthcare system.

6. Individual reflection and discussion

Webster Imasuen:

My involvement in the first section of this report - including the abstract and methodology - led me to contributing to other sections as well. To understand the different applications of the blockchain technology and how they can be utilized, it was necessary to investigate smart contract technologies built on the blockchain.

Involvement in this report gave me the opportunity to gain insight into the algorithm backend; how it operates, as well as the state-of-the-art techniques we use to classify different types of smart contracts. We hope to expand our research efforts in the near future by using a much larger data set and adding some additional functionalities to health care.

To determine whether it will be possible to achieve greater improvement in the future by incorporating new algorithms to perform this task than is currently possible by using the existing techniques used to perform this task. In addition to that, there may also be other ways to improve accuracy in the future.

Stevin Sam:

My participation in this project was mostly based on management of tasks and outcomes. In addition, I have also contributed my knowledge in both theoretical and practical aspects, to my colleagues. Moreover, I was involved in the development and implementation, in which, I worked along my teammates to write a smart contract and state any errors our codes had and came up with the solutions, to make it more robust.

This project has helped me expand my skills in terms of resolving problems and expanding my theoretical knowledge in a more unique programming languages, using Solidity and Java Script. Moreover, this has further deepened my understanding of blockchain and the underlying functions that it has in different concepts e.g., the use of smart contracts to prescribe medication. Future development of the project can allow for additional healthcare features that can make the use of the smart contract to be more widespread.

Aekkaraj Kuplakatee:

I have been involved in this project through the development and management in terms of coding and finding solutions for the team. To be more precise, in the back-end development, I wrote the smart contract with my colleague. To help us build the smart contract we employed many resources in our research such as textbooks, websites, and developer communities. Further to this, I was also able to help the team solve problems that we found along the way.

By doing this project, I learnt additional theory of blockchain technology, smart contracts as well as practical knowledge. In addition, all the skills I have gained from this work will enable me to be a more proficient developer in the future. We believe in future enhancements to this project to gain even better outcomes can be achieved by using big data. We can accomplish this by integrate the project with additional healthcare systems.

6. Gantt Chart

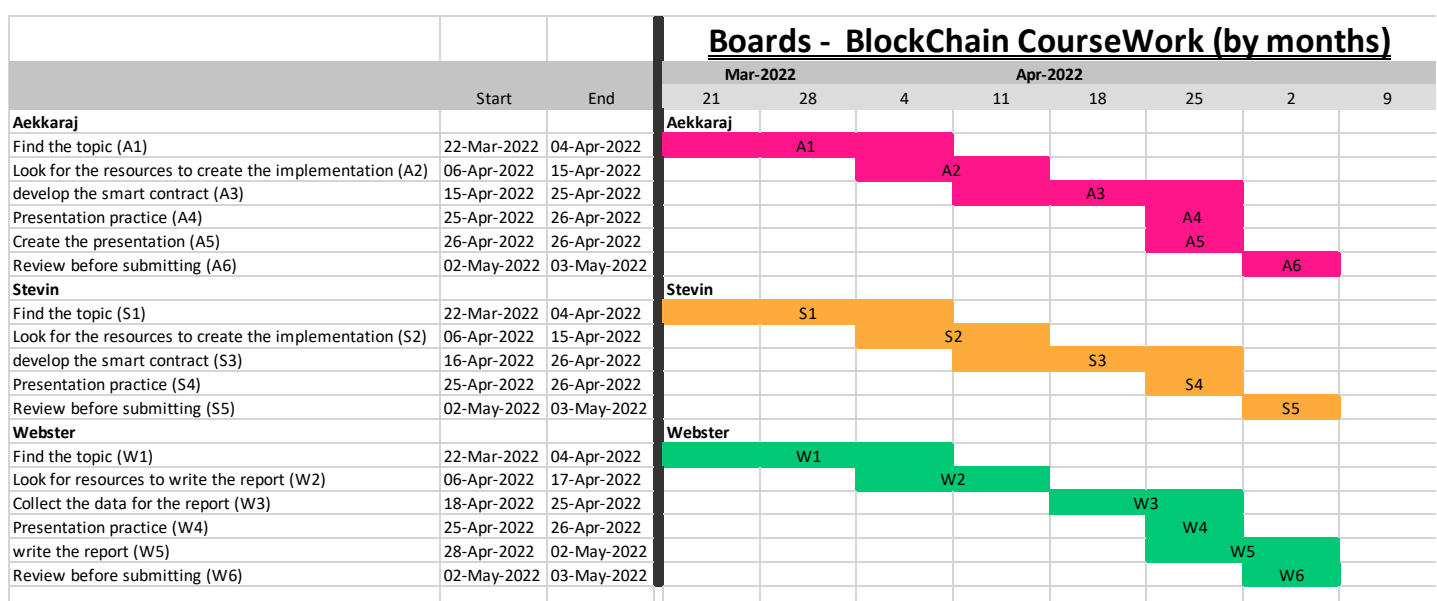


Figure 14. Gantt Chart

7. Reference:

- Nakamoto, S. Bitcoin: A Peer-to-Peer Electronic Cash System; 2008; Available online: www.bitcoin.org (accessed on 15 April 2022).
- Curran, B. What Are the Trustless Environments and How Cryptocurrencies Create Them? Blockonomi.com. 9 July 2018. Available online: <https://blockonomi.com/trustless-environments/> (accessed on 25 April 2022).
- Beck, R.; Avital, M.; Rossi, M.; Thatcher, J.B. Blockchain technology in business and information systems research. *Bus. Inf. Syst. Eng.* 2017, 59, 381–384.
- Lemieux, V.L. Trusting records: Is Blockchain technology the answer? *Rec. Manag. J.* 2016, 26, 110–139.
- Gordon, W.J.; Catalini, C. Blockchain technology for healthcare: Facilitating the transition to patient-driven interoperability. *Comput. Struct. Biotechnol. J.* 2018, 16, 224–230.
- Daisuke, I.; Kashiya, M.; Ueno, T. Tamper-resistant mobile health using blockchain technology. *JMIR Mhealth Uhealth* 2017,
- Wu, H.T.; Tsai, C.W. Toward blockchains for health-care systems: Applying the bilinear pairing technology to ensure privacy protection and accuracy in data sharing. *IEEE Consum. Electron. Mag.* 2018, 7, 65–71.
- Shen, B.; Guo, J.; Yang, Y. MedChain: Efficient Healthcare Data Sharing via Blockchain. *Appl. Sci.* 2019.
- Agbo, C.C.; Mahmoud, Q.H.; Eklund, J.M. Blockchain technology in healthcare: A systematic review. *Healthcare* 2019, 7, 56.
- Kumar, T.; Ramani, V.; Ahmad, I.; Braeken, A.; Harjula, E.; Ylianttila, M. Blockchain Utilization in Healthcare: Key Requirements and Challenges. In *Proceedings of the 2018 IEEE 20th International Conference on e-Health Networking, Applications and Services (Healthcom)*, Ostrava, Czech Republic, 17–20 September 2018.
- Radanović, I.; Likić, R. Opportunities for use of blockchain technology in medicine. *Appl. Health Econ. Health Policy* 2018, 16, 583–590.
- Heston, T. A case study in blockchain healthcare innovation. *Int. J. Curr. Res.* 2017, 20, 131–148.
- Kuo, T.T.; Ohno-Machado, L. Modelchain: Decentralized privacy-preserving healthcare predictive modelling framework on private blockchain networks. *arXiv* 2018, arXiv:1802.01746.
- Stawicki, S.P.; Firstenberg, M.S.; Papadimos, T.J. What's new in academic medicine? Blockchain technology in healthcare: Bigger, better, fairer, faster, and leaner. *Int. J. Acad. Med.* 2018, 4, 1–11.

8. Appendix

8.1 Code:

```
pragma solidity >=0.8.0;
pragma experimental ABIEncoderV2;
// SPDX-License-Identifier: MIT

import "https://github.com/bosszukung/openzeppelin-
contracts/blob/master/contracts/token/ERC721/ERC721.sol";
import "./SafeMath.sol";

contract EPrescription {

    address owner;
    uint256 public prescriptionId;

    struct PrescriptionForm {
        address GP; //Address of doctor that issued this prescription
        address Patient; //Address of Patient that receives this prescription
        string PIPcode; //A unique seven-digit coding system used to ensure
traceability of medication
        string medicationName; //Scientific name of the medicine
        uint256 dosage; //quity per pill
        string dosageUnit; //Unit of the dosage ex. mg, ml and etc
        uint8 numPills; //qulity of pills to give in the prescription
        uint8 repeat; // qutity of time to take the pill (ex. 2 means two times a
day)
        string spicificTime; // time to take the pill (ex. in the morning, afternon,
evening)
        string BeforeOrAfter; // take the pill before or after meal
        uint256 date; //Epoch time when the prescription was given
        uint256 expirationTime; //Epoch expiration date (When is this prescription
no longer valid)
    }

    mapping(uint256 => PrescriptionForm) public prescriptions; //Mapping to
Prescription struct
    mapping(address => bool) public GP; //Mapping the doctor address
    mapping (address => bool) public chemists; //Mapping the chenmist address

    constructor() {
        owner = msg.sender;
    }

    function approved(address _GPToApprove) public onlyOwner {
        GP[_GPToApprove] = false;
    } // _GPToApprove uint256 ID of doctor to approve for giving prescriptions
```

```

function permissionPharmacy (address _phamacist) public view returns (bool) {
    return chemists[_phamacist];
} //given permission to chemist

event PrescriptionAdded(
    address Patient, //Address of Patient that receives this prescription
    string PIPcode, //A unique seven-digit coding system used to ensure
traceability of medication
    string medicationName, //Scientific name of the medicine
    uint256 dosage, //qutity per pill
    string dosageUnit, //Unit of the dosage ex. mg, ml and etc
    uint8 numPills, //qulity of pills to give in the prescription
    uint8 repeat, // qutity of time to take the pill (ex. 2 means two times a
day)
    string spicificTime, // time to take the pill (ex. in the morning, afternon,
evening)
    string BeforeOrAfter, // take the pill before or after meal
    uint256 date, //Epoch time when the prescription was given
    uint256 expirationTime //Epoch expiration date (When is this prescription no
longer valid)
);

function AddPrescription(
    address _Patient,
    string memory _PIPcode,
    string memory _medicationName,
    uint8 _dosage,
    string memory _dosageUnit,
    uint8 _numPills,
    uint8 _repeat,
    string memory _spicificTime,
    string memory _BeforeOrAfter,
    uint256 _date,
    uint256 _expirationTime) public payable {
    prescriptionId++;
    prescriptions[prescriptionId] = PrescriptionForm(
        msg.sender,
        _Patient,
        _PIPcode,
        _medicationName,
        _dosage,
        _dosageUnit,
        _numPills,
        _repeat,
        _spicificTime,
        _BeforeOrAfter,
        _date,
        _expirationTime
    );
}

```

```

    );
    emit PrescriptionAdded(
        msg.sender,
        _PIPcode,
        _medicationName,
        _dosage,
        _dosageUnit,
        _numPills,
        _repeat,
        _specificTime,
        _BeforeOrAfter,
        _date,
        _expirationTime
    );
}

function cancelPrescription(uint256 _prescriptionId) public payable
GPIsApproved(msg.sender){
    PrescriptionForm memory p = prescriptions[_prescriptionId];
    require(p.GP == msg.sender);
}

function giveMedication(address _phamacist) public view {
    require (permissionPharmacy(_phamacist) == true);
}

modifier onlyOwner {
    require(msg.sender == owner);
    _;
}


modifier GPIsApproved(address _GP) {
    require(GP[_GP]);
    _; //Guarantees that address belongs to a valid and approved doctor
}

modifier hasNotExpired(uint256 _prescriptionId) {
    uint256 timeToExpiry = prescriptions[prescriptionId].expirationTime -
block.timestamp;
    require(timeToExpiry > 0);
    _; //Guarantees prescription is not being used after its expiration date
}

modifier onlyPrescribedPatient(uint256 _prescriptionId) {
    require(prescriptions[prescriptionId].Patient == msg.sender);
    _; //Guarantees msg.sender is patient who was actually prescribed
}
}

```


8.2 Presentation:



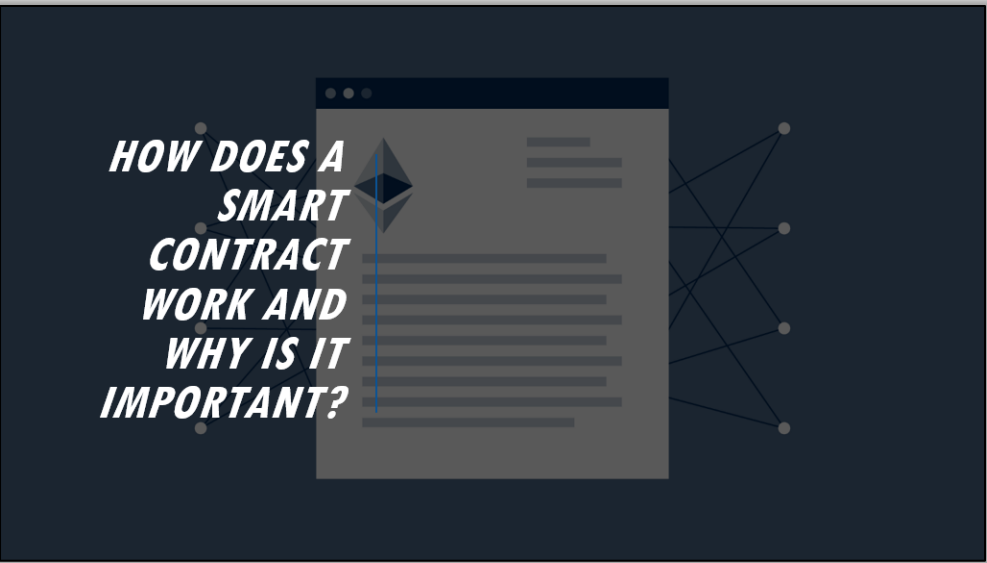
**DEVELOPMENT OF A BLOCKCHAIN-BASED
SMART CONTRACT FOR PRESCRIPTIONS**

Blockchain and Financial
Technologies

Stevin Sam, Aekkaraj Kuplakatee, Webster Osakpamwan IMASUEN



WHAT IS A SMART CONTRACT?



***HOW DOES A
SMART
CONTRACT
WORK AND
WHY IS IT
IMPORTANT?***



HOW A SMART CONTRACT EMPOWERS HEALTHCARE



SMART CONTRACT IN PRACTICAL USE (E-PRESCRIPTION)



OUR DESIGN (1)

```

1
2
3 pragma solidity >=0.8.0;
4 pragma experimental ABIEncoderV2;
5 // SPDX-License-Identifier: MIT
6
7 import "OpenZeppelin/contracts/token/ERC721/ERC721.sol";
8 import "SafeMath.sol";
9
10 contract EPrescription {
11     using SafeMath for uint256;
12     address owner;
13     uint256 public prescriptionId;
14
15     struct PrescriptionForm {
16         address GP; //Address of doctor that issued this prescription
17         address Patient; //Address of Patient that receives this prescription
18         string MFCODE; //A unique seven-digit coding system used to ensure traceability of medication
19         string medicationName; //Scientific name of the medicine
20         uint8 dosage; //Quantity per pill
21         string dosagelimit; //Unit of the dosage ex. mg, ml and etc
22         uint8 numPills; //Quantity of pills to give in the prescription
23         uint8 repeats; //Quantity of time to take the pill (ex. 2 means two times a day)
24         string specificTime; //Time to take the pill (ex. in the morning, afternoon, evening)
25         string BeforeOrAfter; //Take the pill before or after meal
26         uint256 date; //Epoch time when the prescription was given
27         uint256 expirationTime; //Epoch expiration date (when is this prescription no longer valid)
28     }
29
30     mapping(uint256 => PrescriptionForm) public prescriptions; //Mapping tokenId to Prescription struct
31     mapping(address => bool) public GP; //Mapping the doctor address
32     mapping(address => bool) public chemist; //Mapping the chemist address
33
34     constructor() {
35         owner = msg.sender;
36     }
37 }

```


OUR DESIGN (2)

```

38
39 ✓ function approve(address _GPToApprove) public payable onlyOwner {
40     GP[_GPToApprove] = true;
41 } //GPToApprove uint256 ID of doctor to approve for giving prescriptions
42
43 ✓ function remove(address _GPToRemove) public payable onlyOwner {
44     GP[_GPToRemove] = false;
45 } //GPToRemove uint256 ID of doctor to remove for giving prescriptions
46
47 ✓ function seeChemist (address _chemist) public view returns (bool) {
48     return chemists[_chemist];
49 } //given permission to chemist
50

```

OUR DESIGN (3)

```

51 ✓ function AddPrescription(
52     address _Patient,
53     string memory _PCode,
54     string memory _medicationName,
55     uint8 _dosage,
56     string memory _dosageUnit,
57     uint8 _numPills,
58     uint8 _repeat,
59     string memory _specificTime,
60     string memory _BeforeOrAfter,
61     uint256 _date,
62     uint256 _expirationTime) public GPisApproved(msg.sender){
63     prescriptions[prescriptionId] = PrescriptionForm(
64         msg.sender,
65         _Patient,
66         _PCode,
67         _medicationName,
68         _dosage,
69         _dosageUnit,
70         _numPills,
71         _repeat,
72         _specificTime,
73         _BeforeOrAfter,
74         _date,
75         _expirationTime);
76 }
77

```

```

78
79 ✓ function cancelPrescription(uint256 _prescriptionId) public payable GPisApproved(msg.sender){
80     require(prescriptions[prescriptionId].GP == msg.sender);
81 }
82
83 ✓ function giveMedication(uint256 _prescriptionId, address _chemist) public{
84     require(seeChemist(_chemist) == true);
85 }
86
87
88 ✓ modifier onlyOwner {
89     require(msg.sender == owner);
90 }
91
92
93 ✓ modifier GPisApproved(address _GP) {
94     require(GP[_GP]);
95     _; //Guarantees that address belongs to a valid and approved doctor
96 }
97
98 ✓ modifier hasNotExpired(uint256 _prescriptionId) {
99     uint256 timeToExpiry = prescriptions[prescriptionId].expirationTime - block.timestamp;
100     require(timeToExpiry > 0);
101     _; //Guarantees prescription is not being used after its expiration date
102 }
103
104 ✓ modifier onlyPrescribedPatient(uint256 _prescriptionId) {
105     require(prescriptions[prescriptionId].Patient == msg.sender);
106     _; //Guarantees msg.sender is patient who was actually prescribed
107 }
108

```

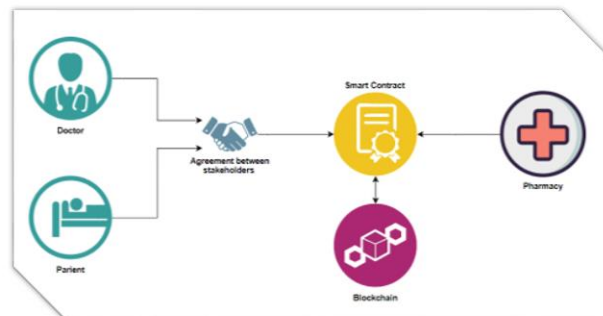
OUR DESIGN(4)

IMPLEMENTATION



SMART CONTRACT ARCHITECTURE

- ❖ Smart Contract Requirements
- ❖ Design
- ❖ Logic
- ❖ Execution
- ❖ Updating




CONCLUSION



8.3: Agreement of Participation – Group Assignment One CN7051

Please complete this agreement and keep a copy for each member of your group. The original of this agreement goes to your Tutor.

We agree to work as a group (**group of 2-3**) to complete the course work for CN7051 and understand that the grade awarded will be the grade allocated to us individually as a result of our group work.

Student ID	Name (block letters) and E-mail Address	Signature
u1423795	Webster Imasuen U1423795@uel.ac.uk	<i>Webster Imasuen</i>
u2235955	Aekkaraj KUPLAKATEE U2235955@uel.ac.uk	<i>Aekkaraj Kuplakatee</i>
u1802868	Stevin SAM U1802868@uel.ac.uk	

Note: Students should form their groups (group of 2-3) within the SAME Tutorial / Practical.

Tutorial / Practical Number: CN7051

Tutor's Name: Arish Siddiqui

Date of agreement: 20 February 2022