

Bachelor of Science in Computer Science & Engineering



Developing Online Healthcare System for CUET Medical Center

by

Jahedul Alam

ID: 1504004

Department of Computer Science & Engineering
Chittagong University of Engineering & Technology (CUET)
Chattogram-4349, Bangladesh.

May, 2021

Developing Online Healthcare System for CUET Medical Center



Submitted in partial fulfilment of the requirements for
Degree of Bachelor of Science
in Computer Science & Engineering

by

Jahedul Alam

ID: 1504004

Supervised by

Dr. Muhammad Ibrahim Khan

Professor

Department of Computer Science & Engineering

Chittagong University of Engineering & Technology (CUET)

Chattogram-4349, Bangladesh.

The thesis titled ‘**Developing Online Healthcare System for CUET Medical Center** ’ submitted by ID: 1504004, Session 2019-2020 has been accepted as satisfactory in fulfilment of the requirement for the degree of Bachelor of Science in Computer Science & Engineering to be awarded by the Chittagong University of Engineering & Technology (CUET).

Board of Examiners

Chairman

Dr. Muhammad Ibrahim Khan

Professor

Department of Computer Science & Engineering

Chittagong University of Engineering & Technology (CUET)

Member (Ex-Officio)

Dr. Md. Mokammel Haque

Professor & Head

Department of Computer Science & Engineering

Chittagong University of Engineering & Technology (CUET)

Member (External)

Dr. Kaushik Deb

Professor

Department of Computer Science & Engineering

Chittagong University of Engineering & Technology (CUET)

Declaration of Originality

This is to certify that I am the sole author of this thesis and that neither any part of this thesis nor the whole of the thesis has been submitted for a degree to any other institution.

I certify that, to the best of my knowledge, my thesis does not infringe upon anyone's copyright nor violate any proprietary rights and that any ideas, techniques, quotations, or any other material from the work of other people included in my thesis, published or otherwise, are fully acknowledged in accordance with the standard referencing practices. I am also aware that if any infringement of anyone's copyright is found, whether intentional or otherwise, I may be subject to legal and disciplinary action determined by Dept. of CSE, CUET.

I hereby assign every rights in the copyright of this thesis work to Dept. of CSE, CUET, who shall be the owner of the copyright of this work and any reproduction or use in any form or by any means whatsoever is prohibited without the consent of Dept. of CSE, CUET.

Signature of the candidate

Date:

Acknowledgements

First I would like to thank almighty Allah for giving me the strength to finish this work. I owe thanks to a number of people without whom this work would not have been possible. I am grateful to my honourable project supervisor Professor Dr. Muhammad Ibrahim Khan , Dept. of CSE, CUET, for the guidance, inspiration and suggestion which were helpful in the preparation of this project. I would also like to thank to my defense committee members, for providing me with their valuable advices and feedback. I am incredibly grateful to Professor Dr. Md. Mokammel Haque, Head, Dept. of CSE, CUET for his outstanding support throughout my undergraduate education. I would also like to extend my gratitude to all of my teachers. Finally I would like to thank my friends, senior, junior and the other people who helped me in this whole journey.

Abstract

In today's world Information and Communication Technology (ICT) plays very important role in every aspect of life. ICT is used heavily in healthcare sector also. The digitalization of healthcare assets is the effective answer to many problems of healthcare industry. While digitizing healthcare assets, security, integrity and privacy must be ensured because health data is too personal and sensitive . In this case blockchain can be used. Blockchain is a system of recording information in which it is difficult to cheat the system. Patient's medical files could be stored in blockchain to provide security features. As storing data in blockchain is expensive, so it is not feasible to store whole document on-chain. The solution to the problem is to store a document's hash on-chain and keep the document elsewhere because hash values are very smaller than whole documents. In this way we can save space and cost. Moreover, modification of files can be detected because change in input results in a new hash value. In our work we have developed an online healthcare system which provides features to manage medical files in an organized way and validate the integrity of the files using blockchain. We hope that if the system is implemented practically it will help to provide better healthcare facilities.

Keywords : Healthcare systems, Electronic healthcare records, Data validation, Data integrity, Blockchain, Hash

Table of Contents

Acknowledgements	iii
Abstract	iv
List of Figures	viii
List of Tables	ix
1 Introduction	1
1.1 Introduction	1
1.2 Framework/Design Overview	1
1.3 Difficulties	2
1.4 Applications	2
1.5 Motivation	3
1.6 Contribution of the thesis	3
1.7 Thesis Organization	4
1.8 Conclusion	4
2 Literature Review	5
2.1 Introduction	5
2.2 Django	5
2.3 HTML CSS Javascript	5
2.4 SQL	6
2.5 Blockchain	6
2.6 Hyperledger Sawtooth	7
2.7 Docker	9
2.8 Related Work	11
2.9 Implementation Challenges	11
2.10 Conclusion	12
3 Methodology	13
3.1 Introduction	13
3.2 Diagram/Overview of Framework	13
3.3 Detailed Explanation	14
3.3.1 Implementation	20

3.3.1.1	Home Page	20
3.3.1.2	Functionality of Admin	20
3.3.1.3	Functionality of Patient	21
3.3.1.4	Functionality of Doctor	27
3.4	Conclusion	31
4	Results and Discussions	32
4.1	Introduction	32
4.2	Dataset Description	32
4.3	Impact Analysis	32
4.3.1	Social and Environmental Impact	32
4.3.2	Ethical Impact	33
4.4	Evaluation of Framework	33
4.4.1	Evaluation of the Client Application	33
4.4.2	Evaluation of the Blockchain Application	33
4.5	Evaluation of Performance	34
4.5.1	Performance of the Client Application	34
4.5.2	Performance of the Blockchain Application	34
4.6	Conclusion	35
5	Conclusion	36
5.1	Conclusion	36
5.2	Future Work	36

List of Figures

1.1	Framework Overview	2
2.1	Blocks of Blockchain	7
2.2	Docker Container	10
2.3	Docker Workflow	11
3.1	Architecture of Client Application	14
3.2	Architecture of Blockchain Application	14
3.3	Patient Module	15
3.4	Doctor Module	16
3.5	Admin Module	17
3.6	Sawtooth Application Architecture	18
3.7	Flowchart for validating files	19
3.8	Home Page	20
3.9	Registering Doctor	21
3.10	Create Schedule	21
3.11	Registering Patient	22
3.12	Blockchain state before Registering Patient	23
3.13	Blockchain state after registering patient	23
3.14	Data saved in blockchain after registering patient	24
3.15	File upload option	25
3.16	Blockchain state after uploading file	25
3.17	Data saved in blockchain after uploading file	26
3.18	Data saved in blockchain after registering patient	26
3.19	Booking Appointment	27
3.20	Doctor's action after booking appointment	27
3.21	After approving appointment	27
3.22	Patient's action after approving appointment	28
3.23	After giving file access	28
3.24	Doctor viewing Medical File	28
3.25	Creating Prescription	29
3.26	Blockchain state after uploading prscription	29
3.27	Data saved in blockchain after uploading prscription	30

4.1	Components of blockchain application in running state using docker	
	33
4.2	Test Hash without file modification	34
4.3	Test Hash with file modification	35

List of Tables

4.1	Testing Features	34
-----	----------------------------	----

Chapter 1

Introduction

1.1 Introduction

The medical world is constantly changing. Technology now plays a big role in the medical domain. Nowadays online healthcare system is a necessity to provide better treatment. To develop an online healthcare system we need to include many functional features along with many security features. In our work we shall try to include the general features such as patient-doctor registration, appointment management, electronic health record management etc. We shall also try to implement the security features using blockchain.

In this chapter we shall discuss the overview of the framework, objective and motivation of the project, challenges we may face while implementing it and application of the project.

1.2 Framework/Design Overview

The framework developed for the work consist of 3 components. These components are-

- **Client Application :** This part contains the general features of an on-line healthcare system. These are- patient registration, doctor registration, appointment booking, uploading medical files, uploading prescriptions, calculating hash etc.
- **Blockchain Application :** This part contains the blockchain related features needed for the project such as storing file hash in blockchain, creating state in blockchain for patient, validating files etc.

- **Rest Api** : This is used for sending data from client application to block-chain application.

Figure 1.1 shows the overview of the framework

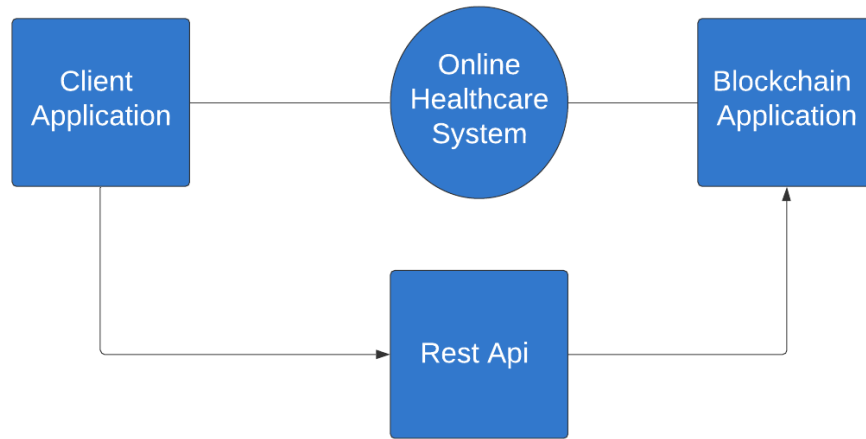


Figure 1.1: Framework Overview

1.3 Difficulties

One of the main challenges in developing online healthcare system is ensuring data privacy because it uses and stores personal and sensitive data about patient.

Converting current paper-based patient medical files into digital record is also a major challenge.

Another challenge is to build interoperable healthcare system that enables the transfer of information among multiple providers.

Developing a solution using blockchain is also a challenge as it is comparatively newer technology.

1.4 Applications

Applications of our work are listed below-

- This system could be used for managing patient medical data in an organized way.

- By using the system doctor will get easy access to patient's medical data which will help them to provide quality treatment.
- Using the system doctor can provide online prescription
- This system could also be used for sharing data among many providers
- This system also provides the feature to check the integrity of patient medical files using Blockchain.

1.5 Motivation

Online Healthcare System is beneficial both to the patient and doctor. Patients get the ability to check medical reports, to book appointment, and to check doctor's schedule online. In this way patient's time and energy is saved because they no longer need to stand in a queue for prescriptions or appointment. Doctors get the ability to check the test reports, to check disease history and to add prescriptions. Access to patient's disease history helps doctors to treat patients more accurately. Doctors would need to depend on the patient's memory without medical records. Due to forgetfulness and complex drug names this can lead to inaccurate disease history. All these things motivated us to do the work.

1.6 Contribution of the thesis

The objectives and possible outcomes of this work are

- To develop an online healthcare system
- To manage medical files in an organized way
- To save time of patients and doctors
- To identify modification of data using blockchain
- To ensure better healthcare facilities

1.7 Thesis Organization

The entire report is represented in five different chapters. We can outline the report structure as follows:

- Chapter 1 gives an overview of the work and explains challenges, motivation and application of the work.
- Chapter 2 describes technologies used in this work and gives an comparison with some related works.
- Chapter 3 explains the methodology and shows the implementation of the system.
- Chapter 4 represents experimental results and evaluation.
- Chapter 5 presents the conclusion and gives recommendation for future work related to this work.

1.8 Conclusion

In this chapter we discussed about design overview, motivation behind the work, difficulties regarding the work , application of the work and the contribution of the work.

Chapter 2

Literature Review

2.1 Introduction

In this chapter first we shall discuss about general terms and technologies used in work. Then we shall discuss about related works.

2.2 Django

Django is a Python Web framework. It helps in clean design and rapid development. Many important development functionalities are readily available in Django. So one can focus on writing app without needing to reinvent the wheel. Some features of Django are

- Rapid Development
- Secure
- Open Source
- Scalable
- Vast and Supported Community

2.3 HTML CSS Javascript

Hyper Text Markup Language (HTML), Cascading Style Sheets (CSS), and JavaScript are the technologies which are used for web design. They are designed for very specific tasks. These are

- HTML is for giving structure to a web page and its content.

- CSS is for adding style to the web page .
- JavaScript is for making the web page interactive.

So HTML is the text and images behind a web page, CSS is the page that actually gets displayed, and JavaScript is the behaviors that can manipulate both HTML and CSS.

2.4 SQL

SQL is Structured Query Language. It is mainly used for manipulating, storing and retrieving data stored in a relational database. SQL is the standard language for Relational Database System. SQL is used in SQL Server, MySQL, Oracle etc.

2.5 Blockchain

Blockchain is an information recording system. It is difficult to change or hack this system [1]. It as an append-only database of transactions. It is distributed to all participants in a blockchain network. Change to the state of the blockchain database is called a transaction. For example:

- Transfer of an asset, such as spending cryptocurrency
- Editing user data, such as patient data in electronic medical record

Blockchain contains log of transactions or a series of state changes. Each block on the blockchain contains transaction data and a header. Header contains signer information, hash value and timestamp. The hash value is a cryptographic signature. It is used to identify the data in a block uniquely. It also helps to identify the block's position in the blockchain because each block's hash includes the hash value of the previous block. So a malicious actor cannot change a block easily. The first block of blockchain is called a genesis block.

Figure 2.1 shows blocks of blockchain-

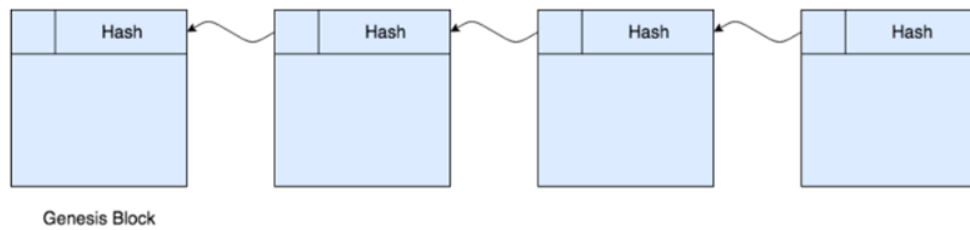


Figure 2.1: Blocks of Blockchain

Blockchain networks have different types of permissions [2]. Such as:-

In a public blockchain [3] there is no restrictions on participation. Anyone can join and submit transactions. Each submitter is identified by a signing mechanism (public/private keys). Bitcoin and Ethereum are public blockchains.

A consortium blockchain [4], is partly decentralized. Joining in the network, participating in consensus is controlled here. But any participant can submit signed transactions. It supports policy-based transaction permissions. Quorum is an Ethereum-powered consortium blockchain. It is used by financial and banking industries.

A private blockchain [5] is "permissioned". It has many access control features. There are separate controls to restrict who can join, participate in consensus and submit transactions. Moreover, in a private blockchain:

- The type of transactions that "transactors" can sign is specified
- "Address space access" is restricted to a limited set of "transactors"
- Policy-based transaction permissioning is supported.

Examples of private blockchain platforms include Hyperledger Sawtooth and Hyperledger Fabric.

2.6 Hyperledger Sawtooth

Hyperledger Sawtooth is an enterprise solution for building, running and deploying distributed ledgers. A modular platform is provided by sawtooth which helps

to implement transaction-based updates to shared state between many participants [6].

In Hyperledger Sawtooth , the core system is separated from the application domain. Application developers have the opportunity to build the business application using the language of their choice. They need not to have the knowledge of the core system. Applications can choose the transaction rules, consensus algorithms and permissioning according to their business needs in sawtooth architecture.

Some terminologies used in hyperledger sawtooth-

- **State:** State is represented as a log of all changes that have occurred since the genesis block. There is a copy of state in a local database of each validator node. Each validator updates its state database to reflect the current blockchain status when new blocks are published.
- **Transaction:** Transaction is a single change in the state of the blockchain. There is unique transaction ID and identification of signer in transaction header of sawtooth transaction.
- **Batch:** Batch is the atomic unit of state change. A transaction is always wrapped inside of a Sawtooth batch. All transactions within a batch are either committed to state together or are not committed at all. A Sawtooth batch has a batch header. In the batch header there is transaction family name , version and the public key of the client who created the batch. Transactions from different applications can be combined in a batch. For example, transactions relating to on-chain settings could be batched with application-specific transactions.
- **Block:** Block consists of many Sawtooth batches. A block has a header. In header there is a timestamp, a signer, and a hash.
- **Validator:** Sawtooth validator is the most important component that performs the following tasks:
 - Validating batches of transactions

- Combining batches into blocks
- Coordinating communication between other components such as other validator nodes, clients and transaction processors
- Maintaining consensus with the Sawtooth network for adding candidate blocks.

2.7 Docker

Docker is a containerization system which helps to run an application successfully irrespective of underlying OS [7]. While developing, testing and deploying an application we face many problems regarding the environment. An application is developed in development environment but it is deployed in production environment. So all the time it does not run successfully. Docker was created to solve this problem.

Docker helps to run every component of a system in separate container. Containers are isolated environments which can have their own network interface, file system, process etc [8]. But the containers share the same OS kernel. A part of an application can run in a docker container with its own libraries and dependencies. So compatibility issue is removed. Figure 2.2 shows this.

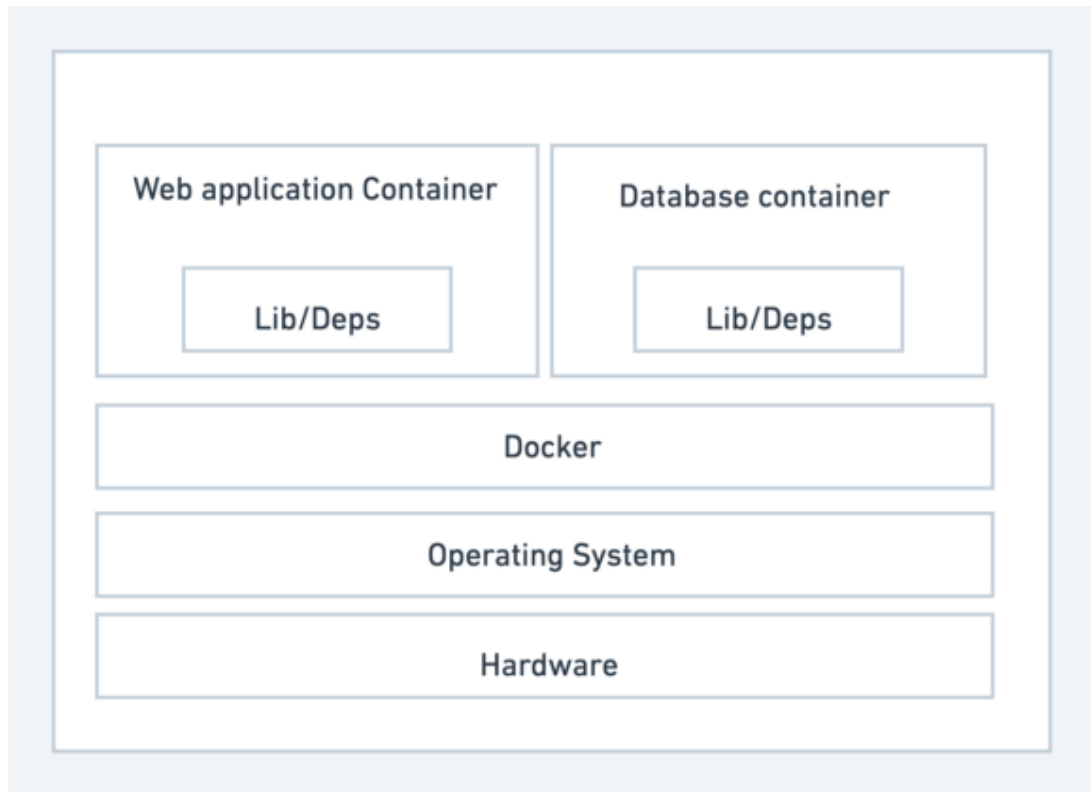


Figure 2.2: Docker Container

Workflow of docker is given below-

1. Developers write code to fulfil the application requirements. Then a Docker File is written which contains project code and commands to create environment to run the application.
2. Docker File produces Docker Image. This image contains dependencies required for a particular application. Docker Containers are runtime instance of Docker Image.
3. Docker Images are then uploaded onto the Docker Hub.
4. Then various teams pull that image when they need it from Docker Hub.

Figure 2.3 shows the workflow of docker.

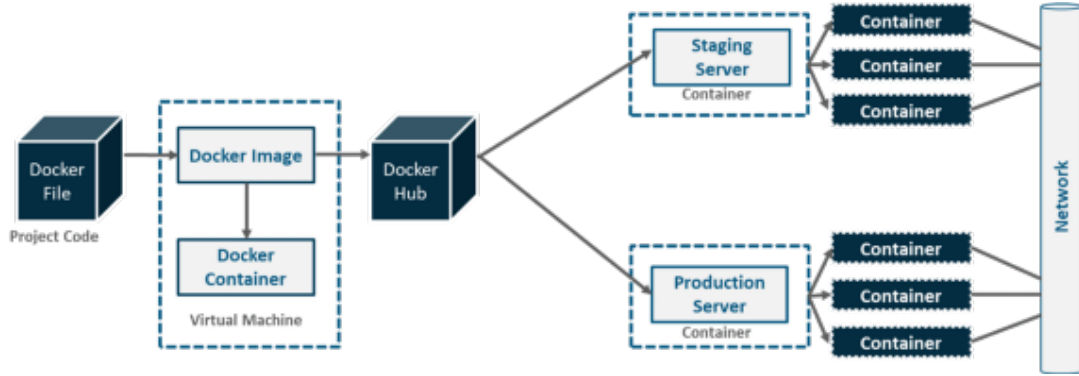


Figure 2.3: Docker Workflow

2.8 Related Work

Many studies have been done to improve online healthcare facilities. Some researcher used artificial intelligence to solve one category of problem in this field. On the other hand some other researcher used blockchain technology to solve another category of problem in online healthcare.

In our work we used blockchain technology to solve data integrity problem in online healthcare. In 2018 a work [9] has been done to validate data integrity with blockchain. Here file hash is stored in Ethereum blockchain and the file itself is stored in database. In our work we used Hyperledger Sawtooth as blockchain to store file hash.

In 2020 another work [10] has been done to develop blockchain-based electronic healthcare record system for healthcare application. In this work Hyperledger Fabric is used as blockchain.

2.9 Implementation Challenges

To implement the whole system we need a client web application, a blockchain application and a rest api.

Client application includes most of features of the project such as patient-doctor registering, appointment booking, file uploading etc. As there are a lot of features

to implement so this part will take a lot of time.

The blockchain application is built to provide blockchain functionality such as storing hash of files, validating files etc. This part is built using hyperledger sawtooth sdk. As hyperledger sawtooth is a newer technology there is only a few resource available to use as reference. This is the challenge of this part.

Rest api will be responsible for sending data from one application to another. It is a big challenge to implement the api with it's expected feature.

2.10 Conclusion

In this chapter we have introduced the term and technologies which are used in our work. Then we have discussed about some related work. After that implementation challenges were explained.

Chapter 3

Methodology

3.1 Introduction

In this chapter we shall discuss about the methodology of the work. At first the overview of the system will be given. Then we shall gradually explain it in detail. After that the implementation of the project will be shown.

3.2 Diagram/Overview of Framework

In the system there are 3 components. These are

- Client Application
- Blockchain Application
- Rest Api

Client Application: Patient registration, doctor registration, appointment booking, uploading medical files, uploading prescriptions, calculating hash etc features are implemented in this part.

The architecture of client application is shown in Figure 3.1

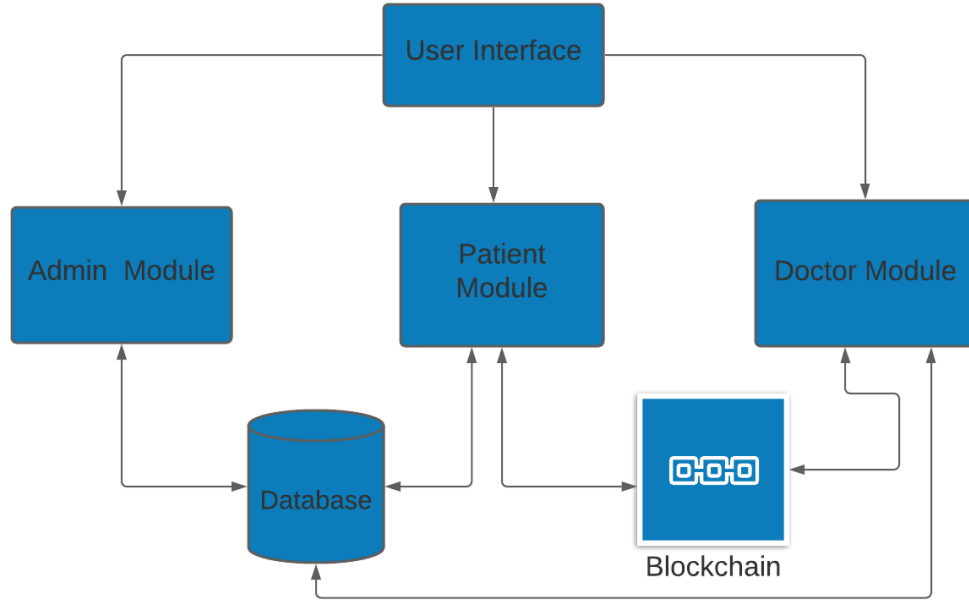


Figure 3.1: Architecture of Client Application

Blockchain Application: Storing file hash in blockchain, creating state in blockchain for patient, validating files etc features are implemented in this part.

The architecture of blockchain application is shown in Figure 3.2

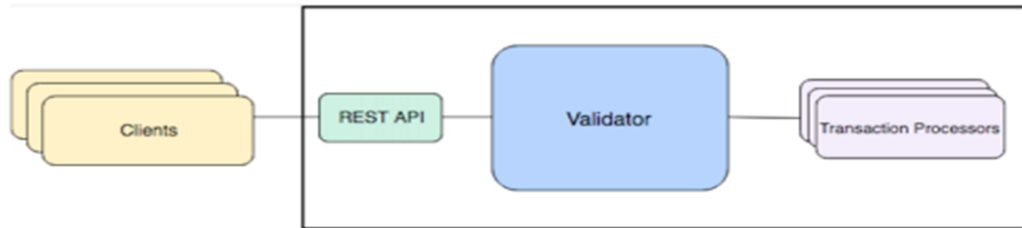


Figure 3.2: Architecture of Blockchain Application

3.3 Detailed Explanation

In this section we shall discuss about every part of the system in details.

Client Application: This part is composed of 3 main modules. From the home page the user can Signup/Login as Patient/Doctor/Admin. After signing in to the system the user can perform predefined tasks according to the respective module. These tasks are given below-

Patients:

- Signup/Login
- Upload personal medical files
- Book appointment at the flexible time and date
- View Prescriptions
- View history of old appointments
- Test Hash of the medical files and prescriptions using blockchain

Figure 3.3 summarizes the tasks included in patient module.

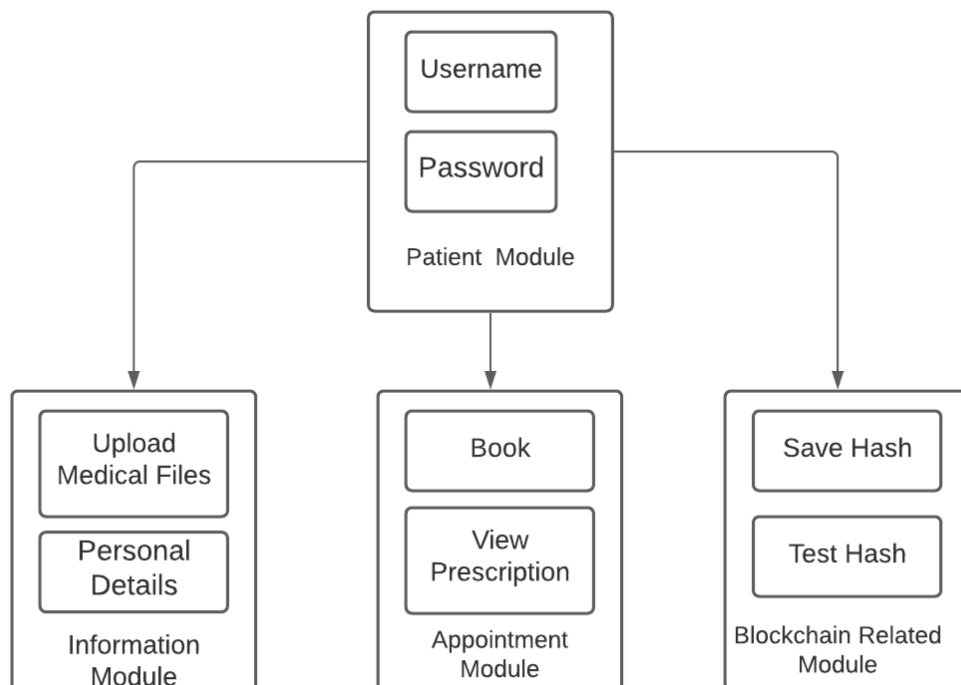


Figure 3.3: Patient Module

Doctors:

- Signup/Login
- Adjust their schedule
- Can confirm or decline an appointment
- View medical history of patients
- Add prescription

- Test Hash of the medical files and prescriptions using blockchain

Figure 3.4 summarizes the tasks included in doctor module.

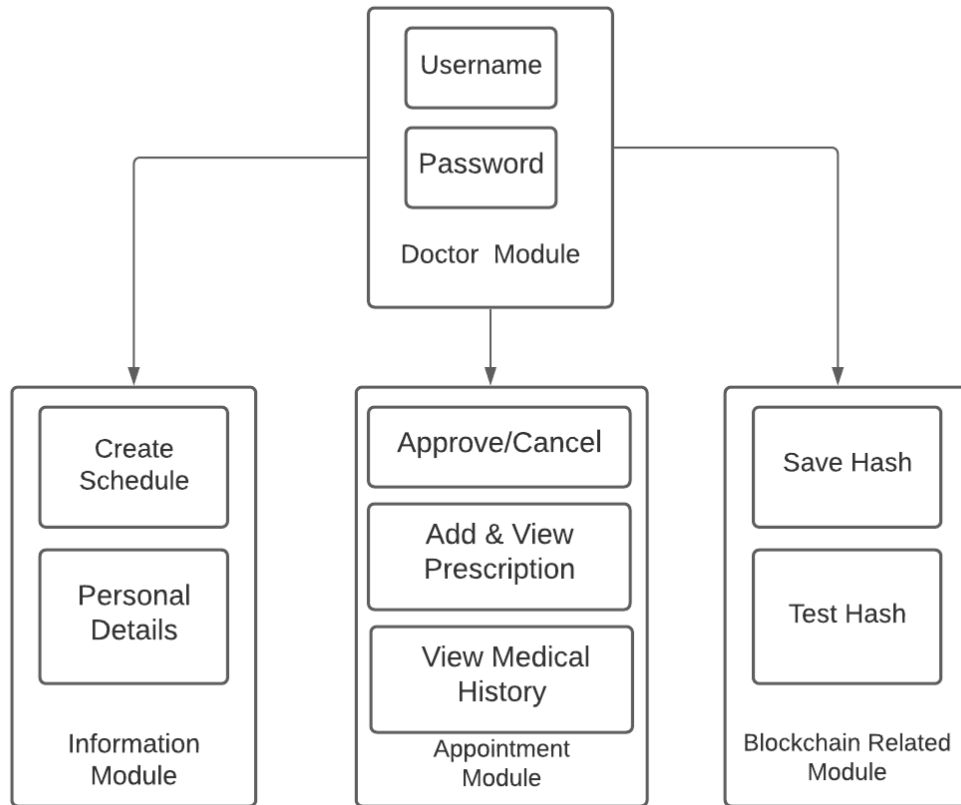


Figure 3.4: Doctor Module

Admin:

- Signup/Login
- Monitor System
- Make necessary changes in system
- Add doctors

Figure 3.5 summarizes the tasks included in admin module.

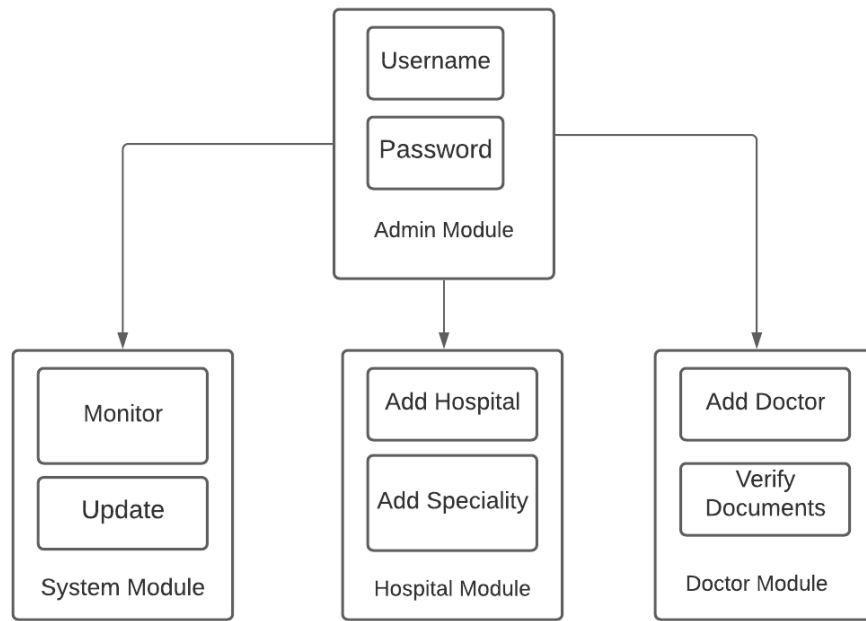


Figure 3.5: Admin Module

Blockchain Application: Patient registration, uploading medical files by patient and uploading prescriptions by doctor, each of these activities adds new block in blockchain. In this work we used hyperledger sawtooth for the purpose of using blockchain. So first we should discuss about a general sawtooth application.

A sawtooth application contains many components which are [11] -

- A data model which helps to provide definition of valid operations and specification of the transaction payload.
- A transaction processor which helps to provide definition of the business logic of any application. Transaction processors validate batches of transactions. It also updates state.
- A client handles application's client logic. It generates and sends transactions to the validator. Moreover it also displays data.
- A REST API which helps to communicate between the the client and transaction processor. Sawtooth provides REST API. We can also use custom REST API.

Figure 3.6 shows architecture of a sawtooth application.

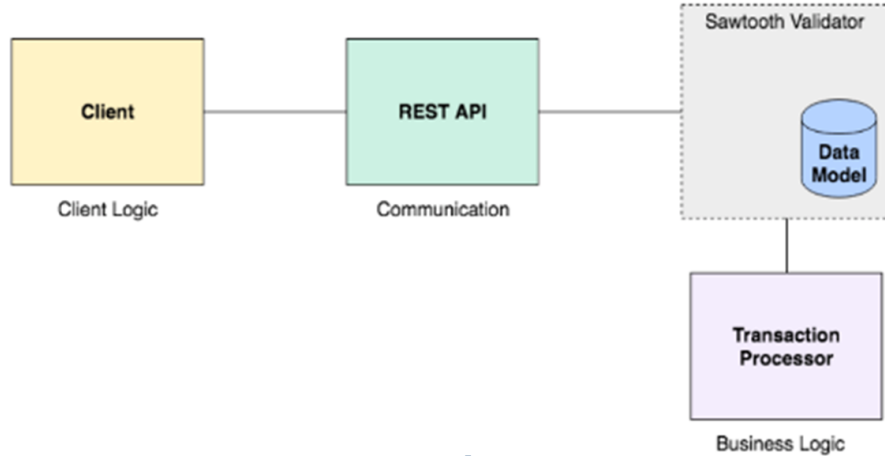


Figure 3.6: Sawtooth Application Architecture

The workflow in Hyperledger Sawtooth application is given below:

1. An action is performed by an user, such as initialization of a value (for example, “set B to 5”)
2. That action is called transaction and it is picked by the client. Then the following tasks are performed by the client:
 - (a) Encoding (serializing) it in a payload,(for example, {“B”: 5})
 - (b) Wrapping that payload in a signed transaction, then in a signed batch which contains one or more transactions
 - (c) Submitting the batch to the validator
3. Then the validation of the batch and transaction are confirmed by validator
4. Then transaction is received by transaction processor and:
 - (a) The signer is verified
 - (b) The transaction is unwrapped and the payload is decoded
 - (c) Validation of the action is verified (for example, ensuring that B can be set to 5)
 - (d) State is modified in a way that satisfies the action (for example, address ...000000b becomes 5)

5. Then the client read that state address and decode it for display to the user.

The main tasks performed in blockchain application are storing hash and validating files. The below flowchart shows the steps needed to implement this features.

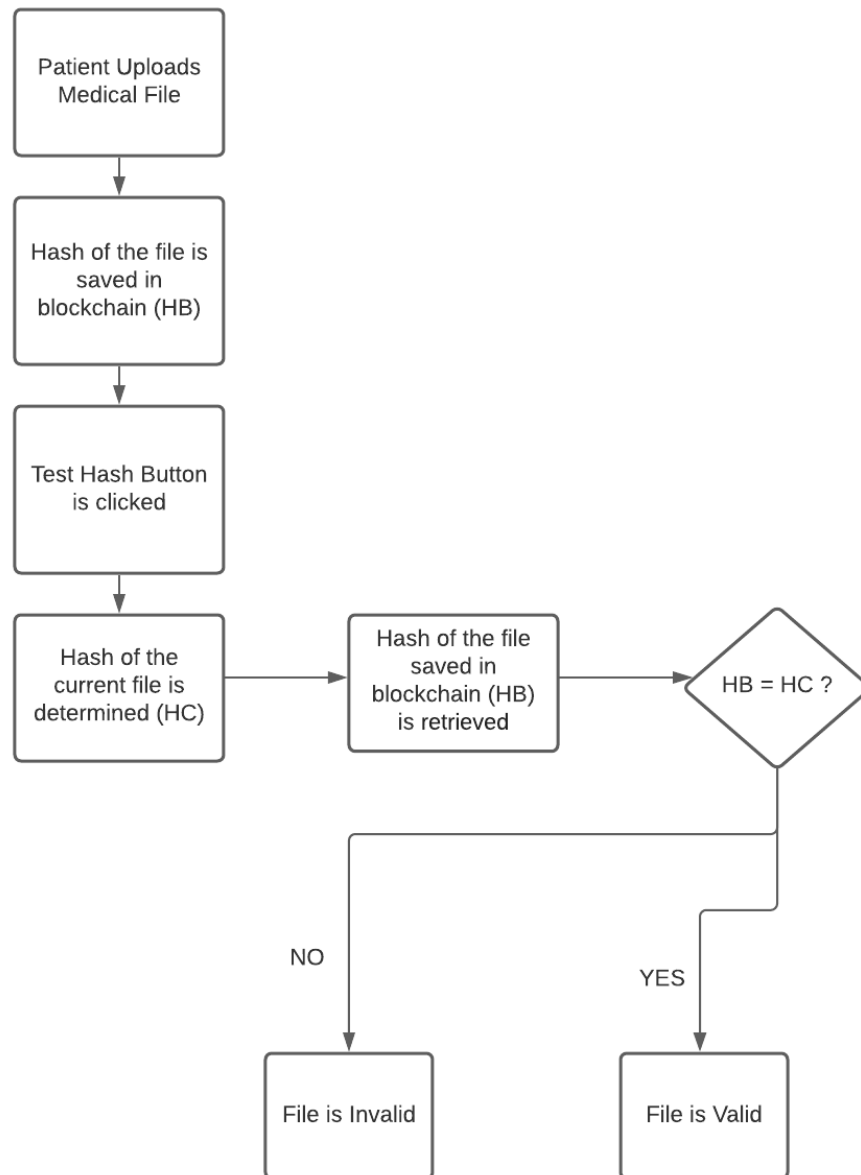


Figure 3.7: Flowchart for validating files

3.3.1 Implementation

The implementation of the system consists of a client web application developed by using Django web framework which will handle the client logic, a blockchain application developed by using hyperledger sawtooth sdk which will provide blockchain functionality and a rest api developed by using Django Rest Framework which will provide communication between the blockchain and client application.

3.3.1.1 Home Page

When we enter into the system the home page is visited first. From the home page user can register as a patient or login as a doctor/patient. Doctor is added by admin of the system.



Figure 3.8: Home Page

3.3.1.2 Functionality of Admin

The admin has the option to add doctor and hospital.

The screenshot shows a web browser window with the URL `localhost:8000/doctor/register`. The page title is "HealthCare". On the right, there is a user profile icon with the text "Hello, useradmin". The main content area features a "Register Doctor" form with the following fields: "Username:" (text input with "doctor1"), "Password:" (password input with masked characters), "Name:" (text input with "doctor1"), "Email:" (text input with "doctor1@gmail.com"), "Hospital:" (dropdown menu with "cmc" selected), "Designation:" (text input with "Medical Officer"), and "Specialty:" (dropdown menu with "Medicine" selected). A green "Register" button is located at the bottom of the form.

Figure 3.9: Registering Doctor

Then when the doctor will login he will get the option to create schedule.

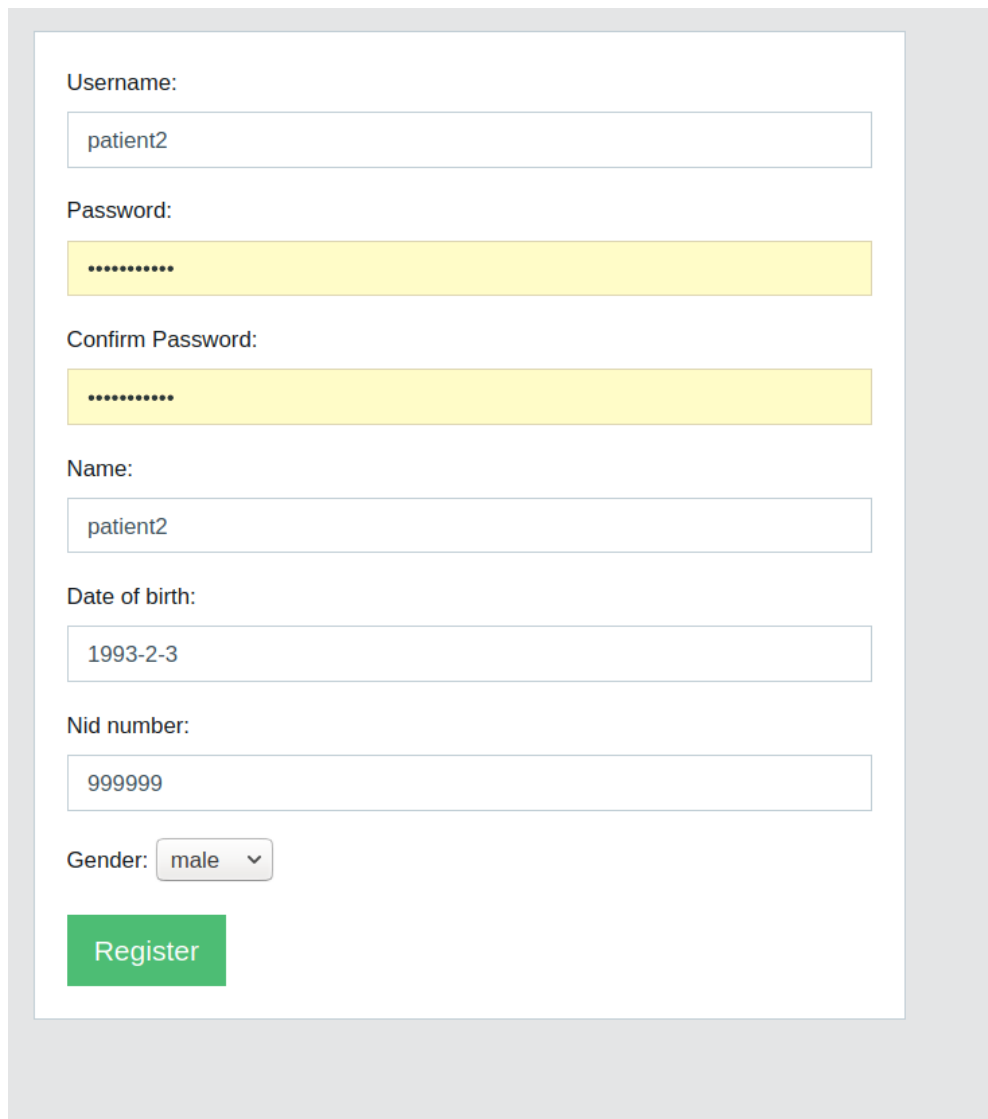
The screenshot shows a "Create Schedule" form. It contains the following fields: "Week day:" (dropdown menu with "Sunday" selected), "Hospital:" (dropdown menu with "cmc" selected), "Start time:" (text input with "09:00"), and "End time:" (text input with "17:00"). A green "Submit" button is positioned at the bottom left of the form.

Figure 3.10: Create Schedule

Now the registered patient can get the appointment of the doctor.

3.3.1.3 Functionality of Patient

A patient can perform various tasks in the system. At first from homepage user can register as a patient.



A patient registration form with the following fields and values:

- Username:** patient2
- Password:** (masked with 8 dots)
- Confirm Password:** (masked with 8 dots)
- Name:** patient2
- Date of birth:** 1993-2-3
- Nid number:** 999999
- Gender:** male (dropdown menu)
- Register** (green button)

Figure 3.11: Registering Patient

This data will be saved in blockchain and database. Before this register action the last block in the blockchain was block 63.

```

{
  "data": [
    {
      "batches": [
        {
          "header": {
            "signer_public_key": "0391a559fa1a6db50829a848f86699a4e5a3b06848b26b58cc4ec87fe8d297d6",
            "transaction_ids": [
              "f39f8ea236d0e8b7305f2053cb5336049ba38a561190405307fba3378ce3a73e5ac7f86b1fbac04d781d0f8fd7c3d248886fe53f59459184ff0594d5e31e"
            ]
          },
          "header_signature": "a1b1ca086c7c570803f5ede6f6142224e29aa715e1100025c4ceab7aceb7e042521017f1b0472f065082dc70724b9397d996f18c980cb85cb8ccdbac262401",
          "trace": false,
          "transactions": [
            {
              "header": {
                "batcher_public_key": "0391a559fa1a6db50829a848f86699a4e5a3b06848b26b58cc4ec87fe8d297d6",
                "dependencies": [],
                "family_name": "health_subscriber",
                "family_version": "1.0",
                "inputs": [
                  "4aa33908ce3329d6ebc2a4d9df5e8a76d4f3b9ca823f8773bb167f5fe7ca57708788f0"
                ],
                "nonce": "0x1.f089373b1a624p-2",
                "outputs": [
                  "4aa33908ce3329d6ebc2a4d9df5e8a76d4f3b9ca823f8773bb167f5fe7ca57708788f0"
                ],
                "payload_sha512": "963621c56f185572bf3b28185ef661aa19e97ca7e6725d3132ce2938a82976cd222cb11059f7920ea65ecc29537c31c2bc7bb08c714cd03efdc5d226fc597",
                "signer_public_key": "0391a559fa1a6db50829a848f86699a4e5a3b06848b26b58cc4ec87fe8d297d6"
              },
              "header_signature": "f39f8ea236d0e8b7305f2053cb5336049ba38a561190405307fba3378ce3a73e5ac7f86b1fbac04d781d0f8fd7c3d248886fe53f59459184ff0594d5e31e",
              "payload": "CghwYXp2bW5089IEtWfzZ0aWNTK3NybmU0aWlyeSJMHTAwMTAwgSDUkVBUVEVUEFUSUUV0Vw=="
            }
          ]
        },
        {
          "header": {
            "batch_ids": [
              "a1b1ca086c7c570803f5ede6f6142224e29aa715e1100025c4ceab7aceb7e042521017f1b0472f065082dc70724b9397d996f18c980cb85cb8ccdbac262401"
            ],
            "block_num": "63",
            "consensus": "R0v2D0W9A2c27TEf8d08bW0k5Ijct0bNk4wLd/Yu0T0WCV0cc",
            "previous_block_id": "8ee910147a317f8a79033de0df4a81cc0933c99c381cd8f8e464ac6018b05663ca5e744725f0323b4a2318ecf43434ac6f08be5ab4b2a61654a98130b3a9804",
            "signer_public_key": "030295f5ab2006f1fe082563ea36dddcabe717c7d8f58dac835ff2d515ee080a57",
            "state_root_hash": "396d346c9f91c367b1e83aa9c7d129ca086d633ea7f6dc08099ec146a7a3c55"
          },
          "header_signature": "54507fa5d25286fbc9463fe87a3753748f8b34087b9c2c48166f18099a161a494c5201d321ee08e4070006f904814ee859636ef1615a36a89472abdf7777fe"
        }
      ]
    }
  ],
  "header": {
    "batch_ids": [
      "a1b1ca086c7c570803f5ede6f6142224e29aa715e1100025c4ceab7aceb7e042521017f1b0472f065082dc70724b9397d996f18c980cb85cb8ccdbac262401"
    ],
    "block_num": "63",
    "consensus": "R0v2D0W9A2c27TEf8d08bW0k5Ijct0bNk4wLd/Yu0T0WCV0cc",
    "previous_block_id": "8ee910147a317f8a79033de0df4a81cc0933c99c381cd8f8e464ac6018b05663ca5e744725f0323b4a2318ecf43434ac6f08be5ab4b2a61654a98130b3a9804",
    "signer_public_key": "030295f5ab2006f1fe082563ea36dddcabe717c7d8f58dac835ff2d515ee080a57",
    "state_root_hash": "396d346c9f91c367b1e83aa9c7d129ca086d633ea7f6dc08099ec146a7a3c55"
  },
  "header_signature": "54507fa5d25286fbc9463fe87a3753748f8b34087b9c2c48166f18099a161a494c5201d321ee08e4070006f904814ee859636ef1615a36a89472abdf7777fe"
}

```

Figure 3.12: Blockchain state before Registering Patient

After registering the last block in the blockchain is block 64.

```

{
  "data": [
    {
      "batches": [
        {
          "header": {
            "signer_public_key": "0391a559fa1a6db50829a848f86699a4e5a3b06848b26b58cc4ec87fe8d297d6",
            "transaction_ids": [
              "bdf6a85028cb299c2983bae367345089dc035da7da71e0236fadb0ae4f5f2ad18cbf5da3f19c350871b2c128a256705c8aba9cc324c4a0204345292e81a1b9"
            ]
          },
          "header_signature": "f99088ec1e652f99f6756ce17ae7519f6405a9172ae4cea3909657cd22e0f21e08002299c02cb1feb1a3f9cc139e194e784de1c8b5b35a7843f39ac37ce98f2de",
          "trace": false,
          "transactions": [
            {
              "header": {
                "batcher_public_key": "0391a559fa1a6db50829a848f86699a4e5a3b06848b26b58cc4ec87fe8d297d6",
                "dependencies": [],
                "family_name": "health_subscriber",
                "family_version": "1.0",
                "inputs": [
                  "4aa33908ce3329d6ebc2a4d9df5e8a76d4f3b9ca823f8773bb167f5fe7ca57708788f0"
                ],
                "nonce": "0x1.60c70ed15c7cap-2",
                "outputs": [
                  "4aa33908ce3329d6ebc2a4d9df5e8a76d4f3b9ca823f8773bb167f5fe7ca57708788f0"
                ],
                "payload_sha512": "c591a360be5bba19fbc31479b705230ab640b1c4905461c2187e07b71f33134637c0591e0d227040f27974a703106450b1741f067608540114097022cd",
                "signer_public_key": "0391a559fa1a6db50829a848f86699a4e5a3b06848b26b58cc4ec87fe8d297d6"
              },
              "header_signature": "bdf6a85028cb299c2983bae367345089dc035da7da71e0236fadb0ae4f5f2ad18cbf5da3f19c350871b2c128a256705c8aba9cc324c4a0204345292e81a1b9",
              "payload": "CghwYXp2bW5089IEtWfzZ0aWNTK3NybmU0aWlyeSJMHTAwMTAwgSDUkVBUVEVUEFUSUUV0Vw=="
            }
          ]
        },
        {
          "header": {
            "batch_ids": [
              "f99088ec1e652f99f6756ce17ae7519f6405a9172ae4cea3909657cd22e0f21e08002299c02cb1feb1a3f9cc139e194e784de1c8b5b35a7843f39ac37ce98f2de"
            ],
            "block_num": "64",
            "consensus": "R0v2D0W9A2c27TEf8d08bW0k5Ijct0bNk4wLd/Yu0T0WCV0cc",
            "previous_block_id": "54507fa5d25286fbc9463fe87a3753748f8b34087b9c2c48166f18099a161a494c5201d321ee08e4070006f904814ee859636ef1615a36a89472abdf7777fe",
            "signer_public_key": "030295f5ab2006f1fe082563ea36dddcabe717c7d8f58dac835ff2d515ee080a57",
            "state_root_hash": "3005f4bc0f86dc3a0e477380570a52779cfed3a3089608373720e0fc30b"
          },
          "header_signature": "a0e590e0b0debae21fab2c128e0eb4381bed64160426558f6714ce94f9764e95f6d2aa183b9cf5cd8513b7dc8eff336dc0892077742ca71752f08590ff4"
        }
      ]
    }
  ],
  "header": {
    "batch_ids": [
      "f99088ec1e652f99f6756ce17ae7519f6405a9172ae4cea3909657cd22e0f21e08002299c02cb1feb1a3f9cc139e194e784de1c8b5b35a7843f39ac37ce98f2de"
    ],
    "block_num": "64",
    "consensus": "R0v2D0W9A2c27TEf8d08bW0k5Ijct0bNk4wLd/Yu0T0WCV0cc",
    "previous_block_id": "54507fa5d25286fbc9463fe87a3753748f8b34087b9c2c48166f18099a161a494c5201d321ee08e4070006f904814ee859636ef1615a36a89472abdf7777fe",
    "signer_public_key": "030295f5ab2006f1fe082563ea36dddcabe717c7d8f58dac835ff2d515ee080a57",
    "state_root_hash": "3005f4bc0f86dc3a0e477380570a52779cfed3a3089608373720e0fc30b"
  },
  "header_signature": "a0e590e0b0debae21fab2c128e0eb4381bed64160426558f6714ce94f9764e95f6d2aa183b9cf5cd8513b7dc8eff336dc0892077742ca71752f08590ff4"
}

```

Figure 3.13: Blockchain state after registering patient

The data is saved in blockchain in this format.

Check Patient

```
GET /api/accounts/patient/check/999999
```

```
HTTP 200 OK
```

```
Allow: GET, HEAD, OPTIONS
```

```
Content-Type: application/json
```

```
Vary: Accept
```

```
{  
  "name": "patient2",  
  "gender": "Male",  
  "date_of_birth": "1993-02-03",  
  "nid_number": "999999",  
  "action": "CREATE_PATIENT"  
}
```

Figure 3.14: Data saved in blockchain after registering patient

When patient login into the system then he get the option to upload medical files.

Upload Medical Files

File name:

Description:

Cholesterol

File: report1.gif

Figure 3.15: File upload option

This time also data is saved in blockchain and database. After this action the last block in the blockchain is block 65.

```

    },
    "header": {
      "batch_ids": [
        "99100e6b89187c88a551b424f33e124f90a32c7872e04b926c431b67a5d73ffc3cd0d68c6f47b0206fc66f005723e8c580f57259a24ad7e3b175141853fd9fa9"
      ],
      "block_num": "65",
      "consensus": "R0V2bW9kZFRx/0wq1TQ0WUqA0rGIoy650LJ1kP7559H55/8CWJqH",
      "previous_block_id": "a5e598e0b0bde9ee21fab2c128e6e6c301db6e41d0b285558fd714ce994f97d46e95f6d2aa183b9cf5cd8513b7dc0eff364dc8092677742c2a71752fd0599ff4",
      "signer_public_key": "038295f5ab2086f1fe082563aa36dddcabe717c71d0f50dac635ff2d515e088a57",
      "state_root_hash": "b404c01759db539d32a08a1c19a16eb0410695bd774fcbba27247562ce97a596"
    },
    "header_signature": "187d77e5152e583dc6fab61db5d300cb3296351d9e01fb0a96e5fa25ecef83886d305b60b2b7c99283954319e0e021b3824406356a634d3261c0227a621bfa96"
  },
  "batches": [

```

Figure 3.16: Blockchain state after uploading file

The data is saved in blockchain in this format.

Check Patient

```
GET /api/accounts/patient/check/999999

HTTP 200 OK
Allow: GET, HEAD, OPTIONS
Content-Type: application/json
Vary: Accept

{
  "name": "patient2",
  "gender": "Male",
  "date_of_birth": "1993-02-03",
  "nid_number": "999999",
  "medicalfiles": [
    {
      "name": "report1",
      "date": "2020-09-19",
      "url": "/media/medical_files/report1.gif",
      "file_hash": "225baa45a3ae3924d40a6a3ce0765d27",
      "pk": 14
    }
  ],
  "action": "CREATE_PATIENT"
}
```

Figure 3.17: Data saved in blockchain after uploading file

Similarly a patient can upload more medical file.

```
{
  "name": "patient2",
  "gender": "Male",
  "date_of_birth": "1993-02-03",
  "nid_number": "999999",
  "medicalfiles": [
    {
      "name": "report1",
      "date": "2020-09-19",
      "url": "/media/medical_files/report1.gif",
      "file_hash": "225baa45a3ae3924d40a6a3ce0765d27",
      "pk": 14
    },
    {
      "name": "report2",
      "date": "2020-09-19",
      "url": "/media/medical_files/reprt2.GIF",
      "file_hash": "2d992f8c75c182805b41f457af5c0115",
      "pk": 15
    }
  ],
  "action": "CREATE_PATIENT"
}
```

Figure 3.18: Data saved in blockchain after registering patient

A patient can also book appointment.

Appointment

Doctor: doctor1 ▾

Schedule:

cmc-Sunday-09:00:00-17:00:00 ▾

Date:

2020-09-19

Register

Figure 3.19: Booking Appointment

3.3.1.4 Functionality of Doctor

When a patient book an appointment then doctor get 2 options.

My Appointment					
Hospital	Day	Date	Patient Name	Status	Action
cmc	Sunday	Sept. 19, 2020	patient2	pending	<div>Approve</div> <div>Decline</div>

Figure 3.20: Doctor's action after booking appointment

If doctor approves, then this view comes-

My Appointment					
Hospital	Day	Date	Patient Name	Status	Action
cmc	Sunday	Sept. 19, 2020	patient2	approved	<div>Terminate</div> <div>Create Prescription</div>

Figure 3.21: After approving appointment

After approval of doctor patient get this option-

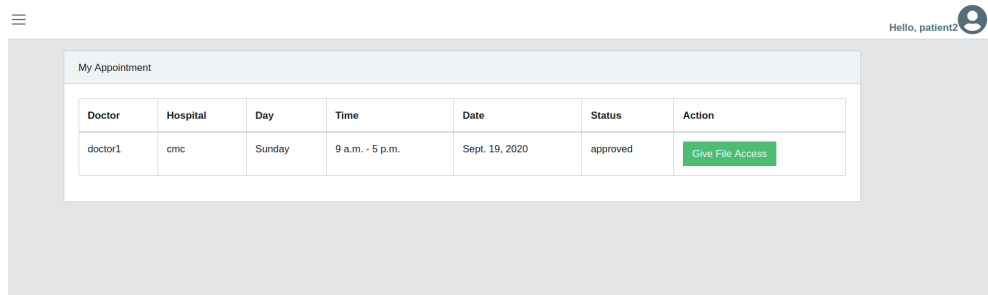


Figure 3.22: Patient's action after approving appointment

When patient gives the file access , doctor gets this view-

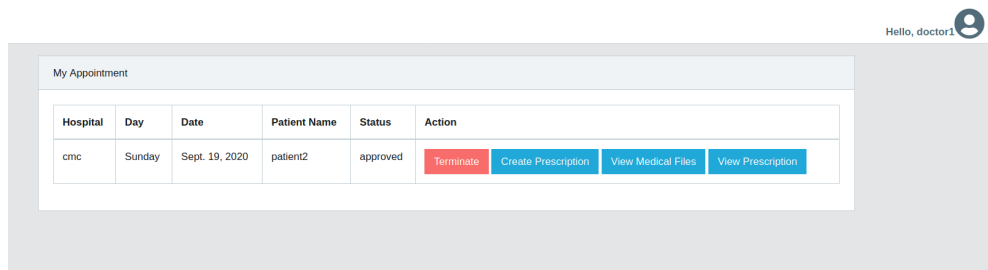


Figure 3.23: After giving file access

When doctor click “View Medical Files” button then doctor can see the medical files of patient.

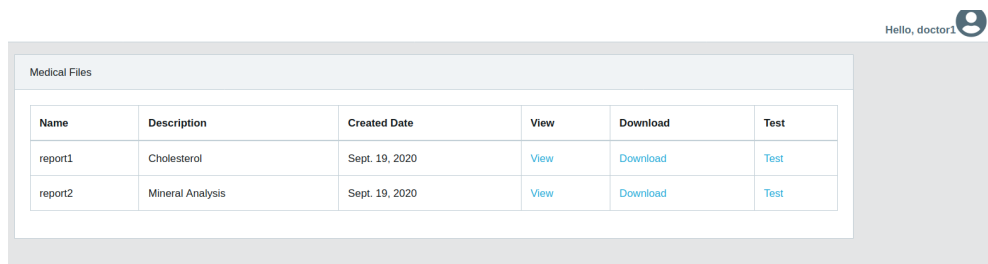


Figure 3.24: Doctor viewing Medical File

Doctor also has option for creating prescription.

Figure 3.25: Creating Prescription

This action adds another block.

```

"header": {
  "batch_ids": [
    "e4fbc9981caa70233f1e7c111cc519b5cb28194f7fb3a04e5a8e4f67f33250563b8a97cf500684ee387325c2adeb0cc0eb30822f388d89dc99f691a2d7da0b02"
  ],
  "block_num": "67",
  "consensus": "R0V2bW9kZZsrJ7LWBHBL/G8/B0K9I7RUP90ZZCTqIdwYIKcYyJzF",
  "previous_block_id": "d808060754f7122bdf4eeb764b3d8254fc240c8062e27e62de5d0918325bd6b05a108eb6da1ed7955abb889c88ca84600d91acbed85e0e154182e32930e1c0a9",
  "signer_public_key": "038295f5ab2006f1fe082563ea36ddcca0e717c71d8f50dac635ff2d515e008a57",
  "state_root_hash": "773b063cdb08843e938c523a6b6fb341767e3b8da4898d5b164c3f17e08cd9f3"
},
"header_signature": "7b22eaf4c4360de833f9b79bc7b1bb6f53dc3a3463bba4ded7de32fa5d217e8a6dcbadfd36467dd2e81cddc685a8d2b898453083013969109b9a92d8b9b34cb6"

```

Figure 3.26: Blockchain state after uploading prscription

The data is saved in blockchain in this format.


```

[
  {
    "name": "patient2",
    "gender": "Male",
    "date_of_birth": "1993-02-03",
    "nid_number": "999999",
    "medicalfiles": [
      {
        "name": "report1",
        "date": "2020-09-19",
        "url": "/media/medical_files/report1.gif",
        "file_hash": "225baa45a3ae3924d40a6a3ce0765d27",
        "pk": 14
      },
      {
        "name": "report2",
        "date": "2020-09-19",
        "url": "/media/medical_files/reprt2.GIF",
        "file_hash": "2d992f8c75c182805b41f457af5c0115",
        "pk": 15
      }
    ],
    "prescriptions": [
      {
        "file_name": "p1",
        "prescribe_by": "doctor1",
        "date": "2020-09-19",
        "url": "/media/prescriptions/prescription1.jpg",
        "file_hash": "de8d8752838f6bbfd4f61c0d8ae84c77",
        "pk": 21
      }
    ],
    "action": "CREATE_PATIENT"
  }
]

```

Figure 3.27: Data saved in blockchain after uploading prscription

Doctor also has the option for testing the hash of medical files. This feature is shown while evaluating the system in the next chapter.

3.4 Conclusion

In this chapter we have explained the methodology used in our work. Then the implementation of the project was shown.

Chapter 4

Results and Discussions

4.1 Introduction

In this chapter we shall discuss about the dataset used in the project. Then we will explain the impact of our project. At last we will evaluate our work.

4.2 Dataset Description

In this project we used patient medical data to evaluate the system. This was collected from some patient through google form. Information of doctor was collected from CUET Medical Center.

4.3 Impact Analysis

Impact of a project is how a project affects the matters which it comes in contact with. A project impact explains the effects which the project is expected to produce upon environment, organization, community, people etc. Here we will discuss the social, environmental and ethical impact of our project.

4.3.1 Social and Environmental Impact

Our work provides many facilities regarding online healthcare. Checking doctor's availability, booking appointment, managing medical files - all these activities can be done online by using the system. These save patient's time. The doctors also get easy access to patient's medical data which helps them to treat the patient properly. So this system has great social and environmental impact.

4.3.2 Ethical Impact

In our work we used hyperledger sawtooth which is a private blockchain. Here only allowed parties can join the network. Moreover patient has control in sharing their medical files with others. This features provide privacy and security of data. There is also a feature for validating data integrity by storing the hash of medical file in blockchain. So all these feature have great ethical impact.

4.4 Evaluation of Framework

To evaluate the system we need to test all the components of the system. We have to run all the components parallely in the same time.

4.4.1 Evaluation of the Client Application

In the system user only interacts with the client application. The client application interacts with blockchain application with the help of rest api. We can do black box testing to test the client application. Black box testing method focuses only on input and output. In this method we need not to know the internal code structure of the software. We only check the output for specific input. The client application of our system has admin, patient and doctor functionality. We have to test all these features.

4.4.2 Evaluation of the Blockchain Application

Evaluating this part of the system is one of the most critical tasks because this part itself contains many parts. This part is implemented using hyperledger sawtooth sdk and docker. Docker containers helps to run all the components of the blockchain application simultaneously. Figure 4.1 shows the application of docker.

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
b56ead9b210a	sawtooth-health-tp	"bash -c '\n bin/hea..."	3 seconds ago	Up 1 second		health-tp
5c0121d79e3d	hyperledger/sawtooth-shell:chline	"bash -c '\n tall -f..."	7 seconds ago	Up 3 seconds	4004/tcp, 8008/tcp	health-shell
713f80e4e0e0	health-client	"bash -c '\n echo 'l..."	11 seconds ago	Up 5 seconds	0.0.0.0:5000->5000/tcp	health-client
46494eb72a8d	hyperledger/sawtooth-settings-tp:1.2	"settings-tp -vv -c ..."	10 months ago	Up 9 seconds	4004/tcp	health-settings-tp
f8c9a0bf5b99	hyperledger/sawtooth-rest-api:1.2	"sawtooth-rest-api ..."	10 months ago	Up 6 seconds	4004/tcp, 0.0.0.0:8008->8008/tcp	health-rest-api
2a0ff703b035f	hyperledger/sawtooth-demode-engine-rust:1.2	"demode-engine-rust..."	10 months ago	Up 7 seconds	0.0.0.0:5050->5050/tcp	health-demode-engine-rust-default
88c3fd5d16d9	hyperledger/sawtooth-validator:1.2	"bash -c '\n lf [! ..."	10 months ago	Up 11 seconds	0.0.0.0:4004->4004/tcp	health-validator

Figure 4.1: Components of blockchain application in running state using docker

Finally the environment for running the components of blockchain application is created. So we can now test the features of blockchain application.

4.5 Evaluation of Performance

In this section we shall explain how our project has performed in real environment.

4.5.1 Performance of the Client Application

We have done black box testing to test the client application. Table 4.1 shows the test result.

Table 4.1: Testing Features

No	Description	Expected Result	Status
1	Application Execution	Application runs properly	Yes
2	Interface Visibility	All interfaces working properly	Yes
3	Admin Module	Admin can perform all tasks properly	Yes
4	Patient Module	Patient can perform all tasks properly	Yes
5	Doctor Module	Doctor can perform all tasks properly	Yes
6	Blockchain Functionality	Communication with blockchain is working	Yes

4.5.2 Performance of the Blockchain Application

The core feature in the blockchain application is validating the integrity of the medical files. So we need to test this feature. When a doctor in the system views the medical files of the patients, he also get the option to test the integrity of the files. If the medical file is unchanged then clicking “Test” button gives “File is valid” result.

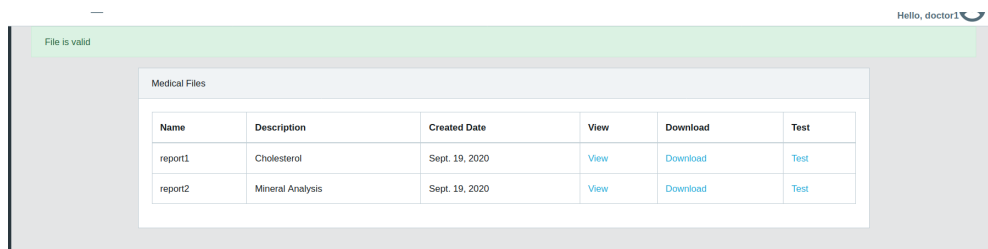


Figure 4.2: Test Hash without file modification

If the medical file is changed then clicking “Test” button gives “File is invalid” result.

Medical Files					
Name	Description	Created Date	View	Download	Test
report1	Cholesterol	Sept. 19, 2020	View	Download	Test
report2	Mineral Analysis	Sept. 19, 2020	View	Download	Test

Figure 4.3: Test Hash with file modification

Doctor can test the hash of prescription by same manner. Moreover patient can test the hash of prescription and medical files by same manner.

4.6 Conclusion

In this chapter we have discussed about dataset of the work. Then while explaining the impact of the work we discussed social, environmental and ethical impact. At last evaluation of the project was done.

Chapter 5

Conclusion

5.1 Conclusion

In our work we have tried to develop a system which will be beneficial to both doctor and patient. Patients can get online appointment, store their medical files in an organised way , share the files with doctors and validate the integrity of the files . Doctors can view patient's previous medical data, validate the data and provide online prescription. To validate the integrity of medical files we used blockchain-based hash validation technique. This technique can only detect modification of data, but it cannot prevent modification. However we think that our system can make a positive change in healthcare industry.

5.2 Future Work

There a lot of scope to work in online healthcare field. We have worked on some of the features of online healthcare system. Some more features can be included for the improvement of the project.

Video chatting service could be included in the system as it is not present now. Our system only detects data tampering but it cannot prevent tampering. It could be an interesting research to investigate the way of preventing data tampering by storing the actual data on the blockchain.

Storing large amount of data on the blockchain is very expensive. So the use of InterPlanetary File System(IPFS) for data storage could be another solution. IPFS [12] is a distributed file system that is resistant to tampering because of its distributed nature and checksum verification. IPFS with blockchain technology can detect and prevent data modification.

References

- [1] M. Nofer, P. Gomber, O. Hinz and D. Schiereck, ‘Blockchain,’ *Business & Information Systems Engineering*, vol. 59, no. 3, pp. 183–187, 2017 (cit. on p. 6).
- [2] C. Mohan, ‘State of public and private blockchains: Myths and reality,’ in *Proceedings of the 2019 International Conference on Management of Data*, 2019, pp. 404–411 (cit. on p. 7).
- [3] *What is a public blockchain? beginner’s guide / 101 blockchains*, <https://101blockchains.com/what-is-a-public-blockchain/>, (Accessed on 05/17/2021) (cit. on p. 7).
- [4] *What are consortium blockchains, and what purpose do they serve? / open-ledger insights*, <https://openledger.info/insights/consortium-blockchains/>, (Accessed on 05/17/2021) (cit. on p. 7).
- [5] *Over blockchain l advantages of a private blockchain*, <https://www.northchain.tech/en/about-blockchain/>, (Accessed on 05/17/2021) (cit. on p. 7).
- [6] K. Olson, M. Bowman, J. Mitchell, S. Amundson, D. Middleton and C. Montgomery, ‘Sawtooth: An introduction,’ *The Linux Foundation*, 2018 (cit. on p. 8).
- [7] B. B. Rad, H. J. Bhatti and M. Ahmadi, ‘An introduction to docker and analysis of its performance,’ *International Journal of Computer Science and Network Security (IJCSNS)*, vol. 17, no. 3, p. 228, 2017 (cit. on p. 9).
- [8] X. Wan, X. Guan, T. Wang, G. Bai and B.-Y. Choi, ‘Application deployment using microservice and docker containers: Framework and optimization,’ *Journal of Network and Computer Applications*, vol. 119, pp. 97–109, 2018 (cit. on p. 9).
- [9] R. Kalis and A. Belloum, ‘Validating data integrity with blockchain,’ in *2018 IEEE International Conference on Cloud Computing Technology and Science (CloudCom)*, IEEE, 2018, pp. 272–277 (cit. on p. 11).
- [10] S. Tanwar, K. Parekh and R. Evans, ‘Blockchain-based electronic healthcare record system for healthcare 4.0 applications,’ *Journal of Information Security and Applications*, vol. 50, p. 102 407, 2020 (cit. on p. 11).
- [11] *Course / hyperledger sawtooth for application developers / edx*, <https://learning.edx.org/course/course-v1:LinuxFoundationX+LFS174x+3T2019/home>, (Accessed on 03/28/2021) (cit. on p. 17).

- [12] J. Benet, ‘Ipfs-content addressed, versioned, p2p file system,’ *arXiv preprint arXiv:1407.3561*, 2014 (cit. on p. 36).