Department of Computer Science & Engineering

A Project Report on

Development of Blockchain-based Medical Records Management System

Submitted in partial fulfillment of the requirements for the award of the degree of

Bachelor of Engineering in Computer Science & Engineering

By

Bhavya Jain	1MS15CS035
Koushik A S	1MS15CS058
Divyansh Lohia	1MS15IS032
Nikita Menon K P	1MS15IS069

Under the guidance of

Dr. Shilpa S. Chaudhari
Associate Professor

M S RAMAIAH INSTITUTE OF TECHNOLOGY
(Autonomous Institute, Affiliated to VTU)
BANGALORE-560054
www.msrit.edu
May 2019

RAMAIAH Institute of Technology

Department of Computer Science & Engineering

CERTIFICATE

Certified that the project work entitled "Development of Blockchain-based Medical Records Management System" carried out by Bhavya Jain(1MS15CS032), Koushik AS(1MS15CS058) bonafide students of M.S.Ramaiah Institute of Technology, Bengaluru in partial fulfillment for the award of Bachelor of Engineering in Computer Science and Engineering of the Visvesvaraya Technological University, Belgavi during the year 2018-19. It is certified that all corrections/suggestions indicated for Internal Assessment have been incorporated in the report deposited in the department library.

The project report has been approved as it satisfies the academic requirements in respect of Project work prescribed for the said Degree.

Project Guide	**Head of the Department**
DR. SHILPA SHASHIKANT CHAUDHARI	**DR.ANITA KANAVALLI**

External Examiners

Name of the Examiners:	**Signature with Date**
1.	
2.	

RAMAIAH Institute of Technology

Department of Computer Science & Engineering

DECLARATION

We, hereby, declare that the entire work embodied in this project report has been carried out by us at M.S.Ramaiah Institute of Technology, Bengaluru, under the supervision of **Dr. Shilpa S. Chaudhari, Associate Professor,** Dept of CSE. This report has not been submitted in part or full for the award of any diploma or degree of this or to any other university.

Signature
BHAVYA JAIN
1MS15CS032

Signature
KOUSHIK A S
1MS15CS058

ACKNOWLEDGEMENT

We take this opportunity to express our gratitude to the people who have been instrumental in the successful completion of this project. We would like to express our profound gratitude to the Management and **Dr. N.V.R Naidu** Principal, M.S.R.I.T, Bengaluru for providing us with the opportunity to explore our potential.

We extend our heartful gratitude to our beloved **Dr. Anita Kanavalli**, HOD, Computer Science and Engineering, for having kind concern and consideration regarding the collaboration with ISE Dept. We are highly indebted to her for constant support and guidance.

We whole heartedly thank our project guide **Dr. Shilpa S. Chaudhari,** for providing us with the confidence and strength to overcome every obstacle at each step of the project and inspiring us to the best of our potential. We also thank her for her constant guidance, direction and insight during the project.

We would like to thank **Dr. Vijaya Kumar B.P**, Head of Department, ISE Dept. Project guide, ISE Dept for motivating and mentoring us all throughout the Project from the initial Ideation Stage to the Final Demo. We are indebted to his for her vision and helping us get around the Project work.

We would like to thank our team-mates **Divyansh Lohia** and **Nikita Menon K P** from the Information Science and Engg. Dept. for collaborating with us in this extensive project and making ends meet at all times. It is with their passion, openness, patience and dedication that we could deliver the project along with a Research Paper and a Work Paper.

This work would not have been possible without the guidance and help of several individuals who in one way or another contributed their valuable assistance in preparation and completion of this study.

Finally, we would like to express sincere gratitude to all the teaching and non-teaching faculty of CSE Department, our beloved parents, seniors and my dear friends for their constant support during the course of work.

BHAVYA JAIN KOUSHIK A S

Abstract

Digital Platforms for storing of medical history has become widely popular as they prove to be a helpful tool in storing all medical information pertaining to an individual such as x-ray's, prescriptions, lab results, etc. A blockchain network, using Hyperledger Fabric, collects data of each patient interaction with the medical professionals. Once a patient visits a doctor, the prescription along with necessary observation is updated on the user's log. The data pertains to health issues faced along with medical solutions and/or therapies proposed by the doctor. Thus, when a patient visits a doctor again, he/she can access the data readily, by either parties.

The reason to port such an application to blockchain is to ensure that this data is immutable. This is due to the reason that during critical times the highly sensitive nature of the patient's medical history can lead to doctors taking better and effective decisions.

LIST OF FIGURES

FIG NO.	CAPTION	PAGE NO.
2.1	Software Process Models	5
4.1a, 4.1b	Gantt Chart	21, 22
5.1	Use Case Diagram	26
6.1	System Design	29
6.2	Class Diagram	31
6.3	Sequence Diagram	32
6.4	Flow of Activities	33
9.1	Signin/Signup Pages	45
9.2	Doctor page	46
9.3	Patient page	47

LIST OF TABLES

TABLE NO	CAPTION	PAGE NO.
1.1	Project Deliverables	3
2.1	Roles and Responsibility	6-7
3.1	Comparative study of papers	18-20
4.1	Risk Identification 1	22
4.2	Risk Identification 2	23
4.3	Risk Identification 3	23
4.4	Risk Identification 4	23
5.1	Functional Modelling 1	27
5.2	Functional Modelling 2	27-28
5.3	Functional Modelling 3	28
8.1	Test Cases	42-44
9.1	Performance Analysis	48

TABLE OF CONTENTS

Chapter No.	Title	Page No.
Declaration		*i*
Acknowledgements		*ii*
Abstract		*iii*
List of Figures		*iv*
List of Tables		*v*
1	**INTRODUCTION**	
1.1	General Introduction	1
1.2	Problem Statement	1
1.3	Objectives of the project	2
1.4	Project deliverables	2
1.5	Current Scope	3
1.6	Future Scope	4
2	**PROJECT ORGANIZATION**	
2.1	Software Process Models	5
2.2	Roles and Responsibilities	6
3	**LITERATURE SURVEY**	
3.1	Introduction	8
3.2	Related Works with the citation of the References	8
3.3	Conclusion of Survey	17
4	**PROJECT MANAGEMENT PLAN**	
4.1	Schedule of the Project	21
4.2	Risk Identification	22
5	**SOFTWARE REQUIREMENT SPECIFICATIONS**	
5.1	Product Overview	24
5.2	External Interface Requirements	24
	5.2.1 User Interfaces	24
	5.2.2 Hardware Interfaces	25
	5.2.3 Software Interfaces	25
	5.2.4 Communication Interfaces	25

5.3 Functional Requirements 26
5.3.1 Functional Requirement 1.1 27
5.3.2 Functional Requirement 1.2 27
5.3.n Functional Requirement 1.3 28

6 **DESIGN**
6.1 Introduction 29
6.2 Architecture Design 29
6.3 Graphical User Interface 30
6.4 Class Diagram and Classes 31
6.5 Sequence Diagram 32
6.6 Data flow diagram 33
6.7 Conclusion 33

7 **IMPLEMENTATION**
7.1 Tools Introduction 34
7.2 Technology Introduction 35
7.3 Overall view of the project in terms of implementation 36
7.4 Explanation of Algorithm and how it is been implemented 37
7.5 Information about the implementation of Modules 38
7.6 Conclusion 40

8 **TESTING**
8.1 Introduction 41
8.2 Testing Tools and Environment 41
8.3 Test cases 42

9 **RESULTS & PERFORMANCE ANALYSIS**
9.1 Result Snapshots 45
9.2 Performance analysis 47

10 **CONCLUSION & SCOPE FOR FUTURE WORK** 49
11 **REFERENCES** 50

I. INTRODUCTION

1.1 General Introduction

Digital Platforms for storing of medical history has become widely popular as they prove to be a helpful tool in storing all medical information pertaining to an individual such as x-ray's, prescriptions, lab results, etc. The solution has proven to be beneficial in solving the problem of storage but has been unable to address the protection of data from the risk of tampering and unauthorized access. Another drawback of this solution is that it is not patient-centric, which means that does not allow the user to control access of their medical records. The need for protection from tampering comes from the consultation as well as the insurance field of work where every record is important and is required in its true form, any false information/tampered data can have a negative impact on the client.

It is these issues that have led us to build an app to store all medical data history of the user on a blockchain. Blockchain, in a nutshell, allows us to have transactions that cannot be tampered and is a secured, decentralized form of storing data.

1.2 Problem Statement

In a research conducted by John Hopkins University where they found that Medical Error is "Third" Leading Causes for death (251,454) in the U.S.A. These errors could have been avoided by sharing medical data of patients to the doctors at crucial times. The problems also include the data is often tampered by patients while submitting to insurance companies. This increases the process of insurance settlements. However, there are constraints to building a digitized solution, such as:

1. Since medical records are highly sensitive and private, there is a threat of malicious users. Centralized databases provide a single point of access to malicious users.

2. Presently patients can't share these documents/records easily with doctors as this data is stored in some silos which takes one-two weeks to find. The patient should be able to control their medical records.

1.3 Objectives of the project

The objectives of this project are to build an app with 4 main goals in mind:

1. A Patient-centric app where patients have full control over their medical record to be built. This includes patients to give permission to view/update their data to medical professionals and revoke permission whenever needed.
2. The proposed solution should provide a secure way of storing medical records such that it is not accessible to malicious users.
3. The proposed solution should provide access to medical history patients that is important for doctors to diagnose a patient's condition during crucial times. This would increase well-informed decisions taken by the doctors.
4. This project should also make it easy for the verification of insurance by providing data that is authentic and tamper proof. Thus, reducing insurance frauds.

1.4 Project deliverables

We propose a blockchain model in which patients and doctors can view the patient's data with ease. The patient is in control of their data thus in such a situation the patient grants permission to the hospital to view their data. The model with be decentralized, thus avoiding a single point of failure during a malicious attack. Data is stored repeatedly on every peer ensuring the authenticity of data on the blockchain. We propose to build a web app that creates an user interface for everyday users can understand and operate easily.

Sl. No.	Deliverable
1	Project Plan
2	System Design
3	Sequence Diagram
4	Blockchain Model
5	Web Application
6	IEEE research paper

Table 1.1. Project Deliverables

1.5 Current Scope

The model with help manages the healthcare records of patients, allowing doctors to view records based on the patient's will. A serious issue associated with electronic health records is the fact that it is specific to a particular institution. The model we propose can be accessed by professionals across institutions provided the patient has offered access. Often while managing hospital records there can be a malicious attack of a patient trying to manipulate their records to claim insurance. This scenario can be avoided by using a decentralized system. This system stores data on every node and therefore any update needs to logged on every system making it harder to make modifications to the data. An additional feature to add to the security of the data is the SHA256 encryption algorithm used to encrypt every block added to the blockchain.

1.6 Future Scope

The model aims to further develop into the insurance industry and validation of doctor's expertise. The entire application can be made scalable making it market-ready. Insurance company requires the entire history of a patient, regarding every hospital visit which is already stored on the blockchain. Also, the insurance money claims can be backed on these records. Furthermore, without exposing patient's records to third party, different doctors' claims to a number of surgeries can be validated using these records, with patients' consent to use their data.

II. PROJECT ORGANISATION

2.1 Software Process Models

We have followed the Waterfall software process model to finish the project. Literature survey and System design, before working on implementation, was completed. The figure describes the process of the waterfall model.

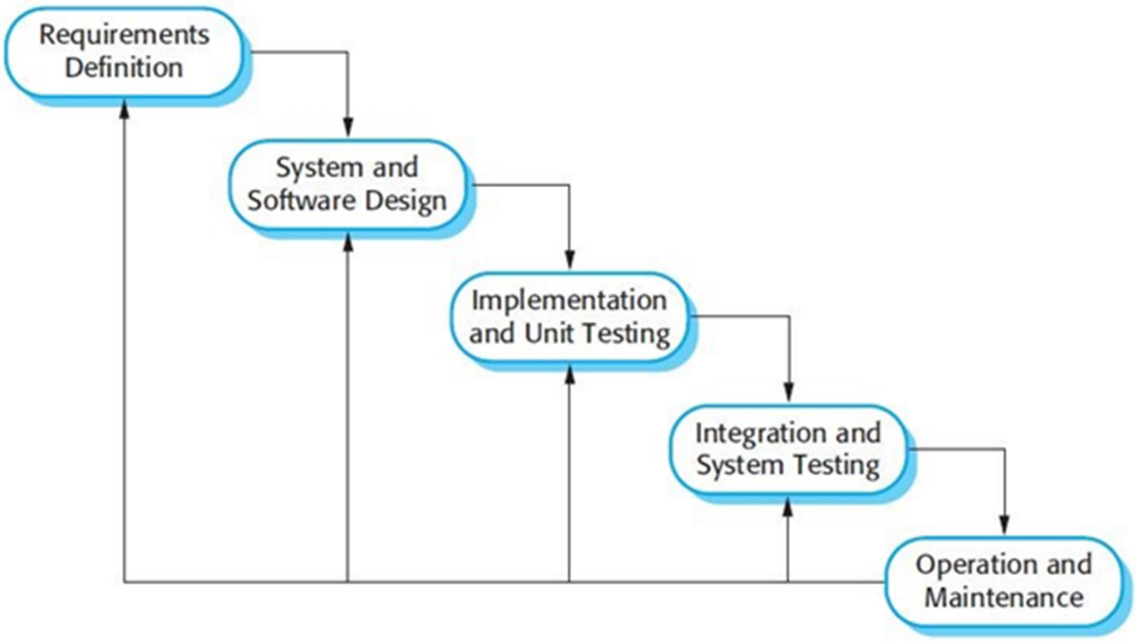

Figure 2.1: Software Process Models

2.2 Roles and Responsibilities

Roles	Description	Names
Technical Head	In charge of leading the development of modules of the project.	Bhavya Jain, Koushik A S, Nikita Menon
Documentation Head	Heads the entire documentation of the project Edits and compiles all the necessary documents.	Divyansh Lohia, Bhavya Jain, Nikita Menon
Literature Survey and Survey paper	Reads past research papers filter out relevant ones and assess the paper content.	Divyansh Lohia, Bhavya Jain, Koushik AS, Nikita Menon
Development of Blockchain Network	Understanding and outlining the implementation of the algorithmic model.	Koushik AS, Nikita Menon
Development of UI	Understanding and outlining the implementation of the algorithmic model.	Bhavya Jain, Koushik AS
Coding Team	Responsible for writing codes of the project.	Bhavya Jain, Koushik AS

Roles	Description	Names
Caliper Testing	Responsible for testing the performance of blockchain Network.	Bhavya Jain, Koushik AS
Assessment Team	Check if the project is meeting the requirements and if any new feature is to be added.	Divyansh Lohia, Bhavya Jain, Koushik AS, Nikita Menon
Documentation Team	Write all the necessary documents recording the milestones and relevant information of the project	Divyansh Lohia, Bhavya Jain, Koushik AS, Nikita Menon

Table 2.1: Roles and Responsibility

III. LITERATURE SURVEY

3.1 Introduction

Distributed electronic database verified by cryptography in which it could demonstrate the time, integrity and identity for individuals and machines without depending on customary frameworks. Blockchain technology differs from traditional databases as the latter utilizes a concentrated framework for validation. While using traditional systems it is troublesome for multiple users to use the database simultaneously because of fixed capacity and fixed a number of credentials provided by this centralized authentication. In Blockchain like a distributed ledger, singular transactions are encrypted into blocks by the applicable encryption, added to the ledger and never deleted. The information in Blockchain is verified fundamentally by a linked list of encoded exchanges that utilizes a hash. Hash function generates a hash by encrypting the information fed in Blockchain.

The World Health Care/Medical Science has advanced greatly since the paper/physical era. With applications like Practo, we have seen the gradual shift of the medical history storage to a digital platform. The need for such a platform arises from the cumbersome search for individual records, relying mainly on Unique ID's and record storage mechanisms.

3.2 Related Works with the citation of the References

A. Patient-centric Model:

Paper[1] focuses on the IT framework which encourages sharing of data and advances cooperation between various restorative partners, for example, patients, social insurance suppliers, controllers or insurance agencies. Be that as it may, there are not many challenges to share the information:

- Business rationale: Each supplier has their own work processes with respect to reading, editing, and updating of healthcare records.

- Trust: Providers need to confide in one another to protect a legitimate and up-to-date view records
- Security: What information can be imparted to whom, and how ailments considered delicate by the patient (for example addictions) can be dealt with
- Database becomes a single point of failure and can be focused by aggressor prompting ransomware assaults or refusal of administrations.

Therefore blockchain-based application can shape the foundation of a decentralized medicinal services platform shared by the patients and suppliers, acting as an interface to the patient's health record. Researchers have proposed a framework in which:

- Patients must have access to their medical record
- Patients must be allowed to selectively share their medical records
- Patients must give approval before any healthcare professional views their medical history.
- Healthcare professionals must declare a reason before requesting to access a patient's records.
- Patients must be allowed to see an entire list of people who have accessed their records
- The regulatory bodies (public health and social insurance organizations), make sure a registry of enrolled beneficiaries is made.
- Blockchain transactions could be approved by a miner network created by the authority, healthcare providers, authorized medical stakeholders (e.g. insurance companies) and other regulatory bodies.
- Should follow either LOINC (for lab exams) or the European Union's "European Patient Smart Open Service" (epSOS) (for drug prescriptions) standards.
- A smart contract can be written between patient and provider based on the EPR Access Agreement to calibrate authentication and notification patterns. When patient terminates from provider according to his smart-contract his data might not be accessible to provider.
- They have also given scope for a patient-provider Agreement for epr access according to rules and laws in the jurisdiction of patient.

The focal point of the paper[2] was to talk about the essential plan, underpins the buyer non-repudiation, model execution in the Ethereum blockchain foundation, its cost examination and a lightweight wrapper, applying legally approbation administration on 2 biomedical databases, to be specific, CARRE hazard factor reference vault and the PubMed MEDLINE database. The said model showed 3 computational layers, specifically, consumer frontend - sending questions to the database utilizing APIs, interface connecting with the biomedical database interface and contract engine which gathers the inquiry and the outcomes together, creates and plans exchanges and oversees contracts and metadata. The basic scheme proposed to generate and deploy a new contract for every query submitted to the blockchain infrastructure. Signing off the query with a public key ensures non-repudiation. The API client then retrieves results against this query and then hashed and added to a smart contract that is pushed on to the blockchain. The engine that generates the contacts then returns the results and its metadata to the data consumer. The entry made is added to the local contract database containing information of the certification to verify the identity of the database and ensure non-repudiation. The versioning scheme takes care of the core requirements and notarization. It limits the number of smart contracts generated to only 1.

This paper [4] proposes Ancile was as a blockchain Electronic Health Records(EHR) system that stores the hash of data references while sending the information of the query link over HTTPS. It overcomes the lack of techniques like proxy re-encryption in JP Morgan's Quorum to streamline secure transactions. It allows to keep keys along with short encrypted records on the blockchain, thus, eliminating the need to store the keys locally. Proxy re-encryption brings and end to the problem relating to the transfer of encrypted records in between nodes in the absence of symmetric keys by using a proxy. The blockchain supports three different types of nodes: full nodes, light nodes, and archive nodes. The proposed Ancile uses 6 different types of smart contracts for operation: Consensus, Classification, Service History, Ownership, Permissions, and Re-encryption. This reduces patient interaction with every contract by increasing utility and reducing privacy threats. It keeps up cryptographic hashes of put away records and query joins, affirming the trustworthiness of EHR Databases. Moreover, by sending the query joins for the records safely off of the

blockchain, Ancile guarantees that the three things required to get to an EHR, the encrypted record, the query link, and the symmetric key, are in various areas.

The paper[5] proposed adding search indices to the blockchain database rather than using a single keyword index while being stored on a public cloud in an encrypted form. The model was built using smart contracts on the Ethereum platform. The 3 entities of the system model: data owner, user, and blockchain. The data owner, be an individual or organisation, creates Electronic Health Records (EHRs). It's then sent to the blockchain and thereby encrypted using a symmetric key and stored on a cloud server. The owner gives the user permission to search the index and thus the acquired results don't require further validation by the user. According to the proposed scheme, every single query is a complex expression and every query is independent of the other The search algorithm returns every identifier that satisfies the query, and it is assumed that these identifiers are transferred through a secure channel to the user. Above this, we can encrypt these identifiers. MongoDB is used in this model where every EHR is seen as a document having a unique document identifier. The plaintext documents are encrypted by the data owner. The encrypted data gathered can be given to any decentralized file storage network, such as the InterPlanetary File System (IPFS). At the point when the user wishes to look through a few EHRs, (s)he first validates himself/herself to the data owner and obtains the search token after authorization. Smart contracts are later used with this token to search the blockchain and get the correct result. This way malicious operation can be easily identified and s dishonest user cannot obtain the data. The user can be assured a reliable and correct result due to the consensus characteristic of blockchain. Changes on the search results can be identified by every node on the Ethereum network. The absence of an update operation proves the confidentiality for the query expressions. The extraction of the document IDs from the EHRs and the transactions on smart contract led to many overheads in Ethereum.

The paper[6] proposes to use a Discrete Wavelet Transform for generating a unique hash decrypted key. There are 6 main parts: network node, cryptographic hash generator, request queuing, GA operation, database, and blockchain structure. Cryptographic Hash Generator (CHG) generates a user key, shared equally over the network, on fetching a new record.

When a user requests to join a permission cluster the network node is tasked and decides to give permission to the user requesting to join the permission cluster. The CHG approves when the generation of the key is correct and sends a verification key to the user along with the required parameters. Employing DWT for encryption enhances the security level. CHG employs an encrypted hash generator, MD5. MD5 results are converted followed by an operation to the numeric version, DWT is applied to make use of the wavelet coefficients. This result is then passed to the implementing system in the form of a resultant key for employing it inside the presented system. While verifying, the private key is obtained from the CHG, this is utilized to create requests sent by the user. This operator uses the remote key for generation of the block request, signing the request via the private key of the transaction and sending it to request queue. Requests from the queue are supplied to the GA process to find the best request retrieved by the consensus node to start the validation process. Immutability is assured by the block header. In the scenario where a block header is changed, the attacker will need to change every block header from the start of the genesis block to falsely access the record of a block. This provision ensures security of blocks on the network. The block header contains version number which shows the rules of validation. The hash of the previous block is used in making the header, which is an MD5 hash and makes sure that no previous block hash is changed without changing the header of the current block. Once the user's request is accepted, all access is given to the user relating to the data that is requested, and this verified request is considered ready for broadcast. This is recorded via the consensus time for sharing with a record that has a purpose and the user apart for just creating this block. A signature is recorded by verified processing proof. The block’s layer format is completed by attaching to a structure number of the created format by using the user key to detect the block mentioned current position.

This paper[8] tested a model on both private platforms of Ethereum and IBM’s Hyperledger fabric’s playground online. The raw data about a patient is collected through IoT devices, master smart device, namely, Oracle, and stored on EHR’s storage system and the transactions are effectively stored as smart contracts, ie, no confidential data is stored on blockchain network. The smart contracts then analyze the provided data and issues alerts to both patients and their health consultant. The blockchain transactions are mapped to the EHR

to provide authentication to the medical records. Its based on a private and consortium-based blockchain, thus reducing excess exposure to information. The true block needs to contain digital signatures from a least number of members. Its frontend is maintained on a Decentralised Application(DApp) for smart devices and communicates with smart contracts and manages user profiles. Doctors will be able to make special administrator actions on the patients' details which will include changing threshold values to be continuously checked. The input data from the sensors will be collected and formatted in the back-end of the DAPP and pushed to the smart contracts.

In Ethereum, a transaction is a signature packet of messages sent from an external account[10]. It contains the message destination, sender's signature Ethereum account balance and data to be sent. Ethereum account balance and data are called STARTGAS and GASPRICE. Transactions require an upper ceiling on the number of computational steps performed by every transaction including initial message and all messages during execution to prevent large scale bursts. The limit is referred to as STARTGAS whereas the cost to be paid to the miner for each step is known as GASPRICE. While comparing Bitcoin blockchain with Ethereum blockchain we see Ethereum has three major improvements. The confirmation time is shorter, confirmation time of the block is approximately 14 s which is extremely quick in comparison to 10 min required for Bitcoin, smaller blocks are present in Ethereum. In Bitcoin, the longest block 1M, whereas the block size in Ethereum depends on the complexity of smart contracts, called the Gas limit. Ethereum can record more data. Two strategies have been distinguished for preservation. The principal method submits unpreserved data when the user needs to secure the new data. The second method distinguishes the crudeness of the data, which is isolated into two fundamental classes: the first is to see the data that has been saved, and the second is to present the data that should be confirmed and check whether they are steady with the content of the security. Every system includes an Ethereum blockchain activity, and in this way every client has a wallet application working on PCs or advanced mobile phones.

B. Research-centric BlockChain:

The paper[3] proposed a decentralized application, DApp, on FHIRChain blockchain-based design that meets the ONC prerequisites for shared clinical basic leadership. The DApp is written in Javascript with 3 solid smart contracts as well as a private testnet of Ethereum. DApp encourages sharing and review of patient malignant growth information for the remote tumor board to make treatment plans for disease patients together and implements a notification service broadcasting to its users alerting them of the new data.

Clinical data silos are normalised with FHIR standards to ensure identical structure of data that is shared. Public and private key cryptography creates digital signatures and encrypts data. Sharing open keys generally enables clients to encode information and confirm computerized marks while private keys are stayed discreet decoding the substance and creating digital signatures at the respective user. The former are users' digital health identities stored in blockchain for identification thus being authenticated by FIHRChain against its private key. This facilitates clinicians efficient data sharing. This platform also made data storage easy and scalable by exchanging encrypted metadata referenced protected data with a short expiration time period. FIHRChain implements "sign then encrypt" policy, ie, encrypt content using users' digital health identities allowing only users holding private keys that are correct to decrypt the same. FIHRChain architecture applies the MVC pattern: model forms immutable blockchain component storing smart contracts, view gives the UI accepting and presenting data, controller facilitating communication between the first 2 and a data-connector validating the FIHR standards and responsible for creating the reference pointers upon requests from the server.

It contains Registry smart-contract keeping up the advanced wellbeing characters of suppliers and mapping their email addresses(or telephone numbers) to both open and private keys, created at the season of enlistment. The information is encoded utilizing the open-source HapiFHIR open test server. Approval is performed through standard articulation parsing of ways against FHIR APIs Bender and Sartipi. Access smart-contract logs all client associations and demands on the entries. It is intended to delineate computerized wellbeing characters to approvals to exclusively named access tokens. The entry advances these

solicitations to the server segment where the complex rationale is embodied. This FIHR increases modularity, enables scalable data integrity, fine-grained access control and enhances trust.

A mechanism to safely store and share medical data using blockchain technology[9]. They discuss a block structure and functioning of the blockchain along with the different permissions given to different bodies accessing the blockchain 2) We present an administration structure for sharing medicinal records to portray the procedure of individual restorative information the executives in certain applications. A blockchain-based individual restorative information application can give a patient medical data administration without disregarding security approaches. 3) The qualities of the medical blockchain are depicted, and a correlation with conventional frameworks from various angles is broke down. The restorative blockchain does not rely upon a confided in outsider, it is the foundation of a framework in which patients have their own total individual medical information, and it can accomplish secure capacity, protection insurance and sealing.

1. PARTIES INVOLVED- Medical foundations, patients and outsider offices, (for example, medicinal data administration stage, medical insurance agency, and so on.) are three primary kinds of exchange bodies in the medical blockchain. Medicinal foundations are in charge of the determination and treatment of patients and producing their therapeutic records. Patients can visit a specialist in various medical organizations and have possession and power over their own medicinal information. The outsider organizations can give a few administrations, for example, medical establishment proposal and arrangement enlistment. Diverse kinds of exchange body have distinctive permissions. The medical blockchain is in charge of producing creation squares. The recently created squares by system hubs are first approved and after that added to the fundamental chain to shape perpetual safeguarding of the exchange information. The timestamp is utilized to guarantee the blocks pursue the planning join in the medicinal blockchain. The information in the blockchain have not been altered through the hash capacity, and identity authentication can be accomplished with a public key encryption. The blends of these advancements guarantee the medical blockchain security.

2. ENCRYPTION- Medical record is generated whenever patient visits doctor. The record generated is hashed and posted in blockchain after signing in with the issuer's (i.e. physicians

and doctors) private key. Simultaneously medical record is encrypted with symmetric key and encryption key of the medical data encrypted with the patient's public key. This keys are sent to patient's. Now patient can verify the signature of the issuer on their medical record and use their own private key to decrypt the medical data encryption key, The original medical data and the signature. At this point another encryption key is produced to store the medicinal data and its signature in cloud storage. The use privileges of the medical record are totally constrained by the patients, and patients can approve the third party agency to get to a portion of his or her medicinal information through the access control mechanism and can pull back approval whenever they want. The location, utilization rights and expiration date of shared records in the cloud storage and the decryption key of the third party organization will be set in the access control policy.

Numerous blockchain-based information sharing applications which share information, for example, Electronic Health Records (EHRs) among various Care Delivery Organizations (CDOs), require privacy protection mechanism with double capabilities.User has to realize that EHR and other related information is without a doubt from real source. On one hand, the clients need to check the legitimacy of EHR information just as the character of the underwriter. Then again, the endorser needs to keep his genuine character private with the end goal that others cannot trace and infer his identity information. These challenges were addressed in Paper[11]: first, a decentralizing attribute-based signature (called DABS) scheme was proposed for providing privacy preserving verification services. The scheme had two salient features: (1) It can effectively confirm the characteristics of the signer without uncovering the identity of the signer; (2) The decentralized attribute based signing property makes it suitable for the distributed blockchain system. Besides, a blockchain-based EHR data storage system is stored through an effective on-chain and off-chain collaboration storage model. The location of each EHR record is stored in a transaction on the blockchain, and the EHR data is stored in off-chain storage. This makes it simpler for users to find each EHR data while also evading the storage constraint of the blockchain.

C. Others:

Ihe paper[7] proposes a 3-tier model containing constrained and unconstrained nodes and patients, N authorities, and EHRs cloud providers. This framework for IoT in the EHR system based on 'Pseudonym Based Encryption with Different Authorities (PBE-DA)'[7] uses Elliptic Curve Cryptography(ECC). DIfferent tiers are as follows:

- First tier**:** Sensors complete starting enrollment to finish the block characteristics. Later, it generates the patient's private key. While sending data, the sender's authenticates itself with the patient framework and distributes the mark of a pseudonym to the patient.
- Second tier: It about accessing or adding to the blockchain network, a new block. The authority, visited by the patient, sends data to the cloud provider to store. Patients restrict the abuse of EHRs and authorities offer services.
- Third tier: Public blockchain technique will be proposed as future work to finish the consistence issue between various EHR cloud suppliers. PBE-DA used the 'Elliptic Curve Diffie-Hellman(ECDH)' key understanding convention, 'Menezes-Vanstone Protocol' and 'Elliptic Curve Digital Signature Algorithm (ECDSA').

3.3 Conclusion of Survey

Research carried out on BlockChain Medical Records Management was surveyed among 11 different research papers, highlighting different approaches and implementation used by researchers. The survey of different papers was done by comparing it with different criteria (like Type of Framework, Type of Blockchain , Patient-centric e.t.c). Different approaches were taken by researchers to address different concerns related to medical records.

To begin the discussion about different approaches, researchers have implemented the blockchain network either in Ethereum or Hyperledger framework with different kinds of encryption added to provide extra layer of safety. Both approaches have their own

advantages, like Ethereum framework is used for building permissionless network which is suited for sharing records among different participants in network whereas Hyperledger is used for building permissioned network which is more suited for B2C businesses.

In respect of use-case, some work was research-centric while some are patient-centric. Patient-centric blockchain network gives the control to patient to share their medical record to different participants, addressing privacy issues concerned with medical records. Whereas Research-centric makes it easier for researchers to access the patients data easily so that they can conduct their research.

Some papers have also implemented external data storage to address the growing concerns of increased redundant data storage. This is done by storing bulk of the information in external database and mapping this data from the blockchain network.

As seen from above discussion, there are a number of different possible approaches for BlockChain Medical Records Management. It is hoped that the present paper has presented different approaches for building blockchain network with the advantages and disadvantages of different approaches.

Paper no.	Type of framework	Type of Blockchain	External Database	Type of Encryption	Patient-centric	Other Remarks
1)	Ethereum	Private (permissioned)	-	-	Yes	1. Conceptual Paper 2. LOINC (for lab exams) or the European Union's "European Patient Smart Open Service" (epSOS) 3. Used smart contract
2)	Ethereum	-	Yes	Uses public key to hash	Yes	1.Versioning scheme-notarization; Cost-effective 2. Used smart contract

Paper no.	Type of framework	Type of Blockchain	External Database	Type of Encryption	Patient -centric	Other Remarks
3)	Ethereum	private	-	Digital signatures are created by using both public and private keys	No	1.FIHRChain architecture; uses HapiFHIR public test server 2. Scalable 3. Used smart contract
4)	Ethereum	-	Yes	Proxy re-encryption using symmetric keys	Yes	Used smart contract
5)	Ethereum	public	Yes	Uses symmetric key	Yes	1. Decentralized file system: InterPlanetary File System 2. Used smart contract
6)	-	-	-	Discrete Wavelet Transform generates a unique hash key; uses MD5; works on private & public keys	-	Mainly talks about the implementation
7)	-	Public	Yes	Pseudonym Based Encryption with Different Authorities (PBE-DA) uses Elliptic Curve Cryptography	Yes	-
8)	Ethereum; Hyperledger fabric's online playground	private	Yes	-	No	1. HIPAA compliant 2. Used smart contract

Paper no.	Type of framework	Type of Blockchain	External Database	Type of Encryption	Patient -centric	Other Remarks
9)	-	-	No	Medical data encrypted using symmetric key & then Public key encryption	Yes	-
10)	Ethereum	-	-	-	-	Talks about benefits of using Ethereum framework.
11)	-	-	Yes (they use on-chain and off-chain collaboration to store data)	Attributes and private keys from multiple authorities, ie, belonging to different organisations are issued to the users. Globally verifiable identifier binds keys issued by the different authority, to the same user	-	Unforgeability, Security under Collusion attack, anonymity of (user' real identity), Non-Repudiation

Table 3.1: Comparative study of papers

IV. PROJECT MANAGEMENT PLAN

4.1 Schedule of the Project

	Name	Duration	Start	Finish	Predecessors	Resourc...
1	**Problem Description**	**13 days?**	**3/12/18 8:00 AM**	**19/12/18 5:00 PM**		
2	BrainStorming	3 days?	3/12/18 8:00 AM	5/12/18 5:00 PM		
3	Assessment of problem scope	5 days?	6/12/18 8:00 AM	12/12/18 5:00 PM		
4	Abstract and synopsis	8 days?	10/12/18 8:00 AM	19/12/18 5:00 PM		
5	**Project Plann with Gantt chart**	**4 days?**	**29/1/19 8:00 AM**	**1/2/19 5:00 PM**	**1**	
6	Ideafinialization	2 days?	29/1/19 8:00 AM	30/1/19 5:00 PM		
7	Preparing Gantt chart	2 days?	31/1/19 8:00 AM	1/2/19 5:00 PM		
8	**Literature Survey & Survey paper**	**10 days?**	**4/2/19 8:00 AM**	**15/2/19 5:00 PM**	**1;5**	
9	Study of Related works	7 days?	4/2/19 8:00 AM	12/2/19 5:00 PM		
10	Compare feature of existing products	4 days?	12/2/19 8:00 AM	15/2/19 5:00 PM		
11	**System Design**	**6 days?**	**18/2/19 8:00 AM**	**25/2/19 5:00 PM**	**1;5;8**	
12	Overview of tools and framework	6 days?	18/2/19 8:00 AM	25/2/19 5:00 PM		
13	Analysing potential constraints	6 days?	18/2/19 8:00 AM	25/2/19 5:00 PM		
14	**Development and Coding**	**14 days?**	**26/2/19 8:00 AM**	**15/3/19 5:00 PM**	**11**	
15	**Building the Blockchain Network**	**6 days?**	**26/2/19 8:00 AM**	**5/3/19 5:00 PM**		
16	Setting up the coding Environment	2 days?	26/2/19 8:00 AM	27/2/19 5:00 PM		
17	Configuring additional tools and framework	3 days?	26/2/19 8:00 AM	28/2/19 5:00 PM		
18	Application prototype	4 days?	28/2/19 8:00 AM	5/3/19 5:00 PM		
19	**Setting up the frontend**	**7 days?**	**27/2/19 8:00 AM**	**7/3/19 5:00 PM**		
20	Setting up the coding environment	2 days?	27/2/19 8:00 AM	28/2/19 5:00 PM		
21	Configuring additional tools and framework	6 days?	28/2/19 8:00 AM	7/3/19 5:00 PM		
22	Application Protoype	5 days?	1/3/19 8:00 AM	7/3/19 5:00 PM		
23	**Integration**	**7 days?**	**7/3/19 8:00 AM**	**15/3/19 5:00 PM**		
24	Deploying essential cloud service	3 days?	7/3/19 8:00 AM	11/3/19 5:00 PM		
25	Combining modules	0 days?	12/3/19 8:00 AM	12/3/19 8:00 AM		
26	Configuring front end with back end	4 days?	12/3/19 8:00 AM	15/3/19 5:00 PM		
27	**Testing**	**5 days?**	**18/3/19 8:00 AM**	**22/3/19 5:00 PM**	**23**	
28	Running Unit Testing	3 days?	18/3/19 8:00 AM	20/3/19 5:00 PM		
29	Real world Testing	4 days?	19/3/19 8:00 AM	22/3/19 5:00 PM		
30	Writing Report and Research paper	5 days?	22/4/19 8:00 AM	26/4/19 5:00 PM	1;5;8;11;14;27	
31	Pradarshana	0 days?	4/5/19 8:00 AM	6/5/19 5:00 PM	14	
32	FinalSubmission	2 days?	9/5/19 8:00 AM	10/5/19 5:00 PM	30	

Figure 4.1a: Gantt Chart

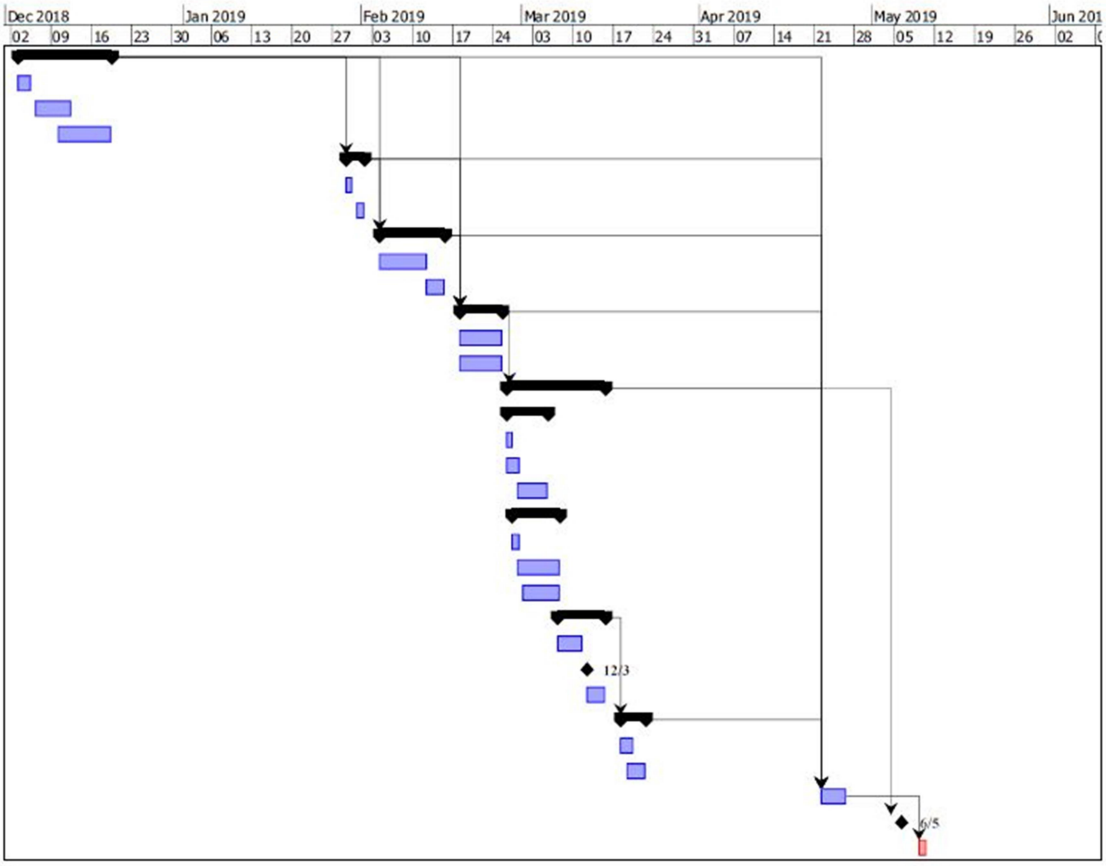

Figure 4.1b: Gantt Chart

4.2 Risk Identification

4.2.1 Risk Identification 1

Dependency on a technology that is still under development lengthens the schedule.

Probability	High
Impact	High
Mitigation	Research on use case of different Blockchain frameworks and choosing the appropriate framework to fit the problem case.

Table 4.1: Risk Identification 1

4.2.2 Risk Identification 2

Selected technology is a poor match to the problem

Probability	Medium
Impact	High
Mitigation	Research on use case of Blockchain in healthcare application.

Table 4.2: Risk Identification 2

4.2.3 Risk Identification 3

Personnel need extra time to learn unfamiliar software tools and programming language.

Probability	Medium
Impact	Medium
Mitigation	Learning Blockchain technology and its concepts before starting the project.

Table 4.3: Risk Identification 3

4.2.4 Risk Identification 4

Critical development work is being performed by one developer

Probability	Low
Impact	Low
Mitigation	Dividing critical works with 2 more team members.

Table 4.4: Risk Identification 4

V. SOFTWARE REQUIREMENT SPECIFICATIONS

5.1 Product Overview

The product (web application) developed through this project focuses on patients medical record management using blockchain technology. The blockchain based network for medical records system collects medical data about each visit of a patient from a medical professional and updates the prescription along with necessary observation on the patient log. This can be accessed by the patient or doctor readily during critical times to take better and effective decisions. The control over who can view and update the medical record is with the patient. The patient can give/revoke permission to any doctor at any point in time. The reason to port such an application to the blockchain is to ensure immutability.

5.2 External Interface Requirements

5.2.1 User Interface:

Web application: The Web application is a bridge between the user and the blockchain network. This application will be used by users to interact with the server which has the blockchain network model through the HTTP request and response. The patients should be able to view their medical record. They should also give and revoke permissions to doctors over their medical record. This application should also be accessible by doctors to see the patient current medications and observations of previous visits of the patient. This application should provide a good interface to update observations made during the current patient visit.

5.2.2 Hardware Interfaces

Devices connected to the network: Web browser is medium through which the user can launch the web application. Therefore the user should have a device which supports web browser such as (Firefox latest version, Chrome latest version, Edger 13-14 version, IE 9-11 version, Safari 7-10 version, Ios 7-10 version, Android 4.1 - 6.0 version).

Server (AWS Server): Server is needed for the project as Blockchain network is installed and deployed there The user can communicate to the server through the HTTP request and response.

5.2.3 Software Interfaces

Blockchain network (Hyperledger Composer): The most important requirement is to store the medical records in a decentralized and secure way. This could be done by building a blockchain network using hyperledger composer. Hyperledger is chosen as permissioned blockchain network to addressing privacy issues concerned with medical records.

Composer REST Server: Even after building a blockchain network, the application requires a responsive and reliable REST server to respond to the query and update requests sent by the application. This server defines the REST APIs and gives the requested services.

Docker: Docker presents the application with an isolated environment, containers, to run freely and independently from the host system.

5.2.4 Communication Interfaces

HttpClient: The Hypertext Transfer Protocol (HTTP) is an application protocol for distributed, collaborative, and hypermedia information systems. HTTP is the foundation of data communication for the World Wide Web. HTTP functions as a request-response

protocol in the client-server computing model. Purpose of HTTPClient is to transmit and receive HTTP messages. All HTTP requests have a request line consisting a method name, a request URI and an HTTP protocol version. The client side of HTTP is concerned with sending requests via GET or POST methods to the server. The HTTP response is a message sent by the server back to the client after having received and interpreted a request message. The obtained response is processed at the client side for application specific purposes.

5.3 Functional Requirements

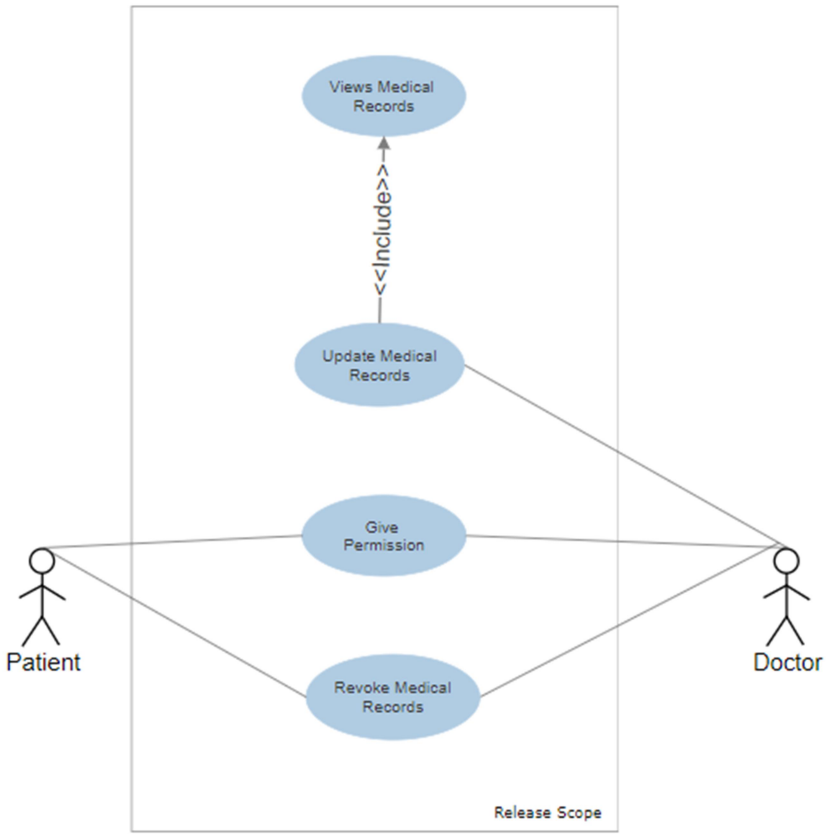

Figure 5.1: Use Case Diagram

5.3.1 Functional Requirement 1.1

Case name	Give Permission to view and view the medical history.
Actors	Patient, Medical Record
Precondition	• Actor should be patient • Patient should have a Medical Record
Main Success Scenario	• Patient gives the permission using Doctor id. • The peer ledgers should be updated with doctors id.
Exception	• Network problem. • Doctor does not have access to medical record.

Table 5.1: Functional Modelling 1

5.3.2 Functional Requirement 1.2

Case name	Updating the medical record.
Actors	User, Medical Record
Precondition	• The actor should be a doctor. • Doctor should have read/write permission of patients medical record.
Main Success Scenario	• A doctor should successfully create a new transaction using medical id. • New medication and Visit data should be stored in the peer ledgers

Exception	• The doctor does not have the read/write permission for patients medical record. • patient does not have access to medical record.

Table 5.2: Functional Modelling 2

5.3.3 Functional Requirement 1.3

Case name	Revoking the permissions of medical Record.
Actors	Patient and Medical record.
Precondition	• The actor should be a patient. • Patient should have a Medical Record • Doctor/Researcher should have permission to access medical record.
Main Success Scenario	• Patients should be able to revoke permissions granted earlier using Doctor id. • The peer ledgers should be updated with doctors id
Exception	• Network problem. • Patient does not have access to medical record. • Doctor/Researchers do not have the permission to the medical Record initially.

Table 5.3: Functional Modelling 3

VI. DESIGN

6.1 Introduction

The system design can be fairly abstracted to be consisting of two divisions namely the application and server side. This way of division helps to understand the interactions between various components of the system and their interdependencies.

The application block represents the interaction between the user and the application. This block sends any query to either read past medical records or updates the ledger. The server block represents the interaction between the application and the server. It is responsible for addressing the query from the application and return the processed information to the application which in turn renders the information to the user.

6.2 Architecture Design

The architecture design can be visualized to be composed of components interacting with each other exchanging data. The flow can be best represented by a diagram.

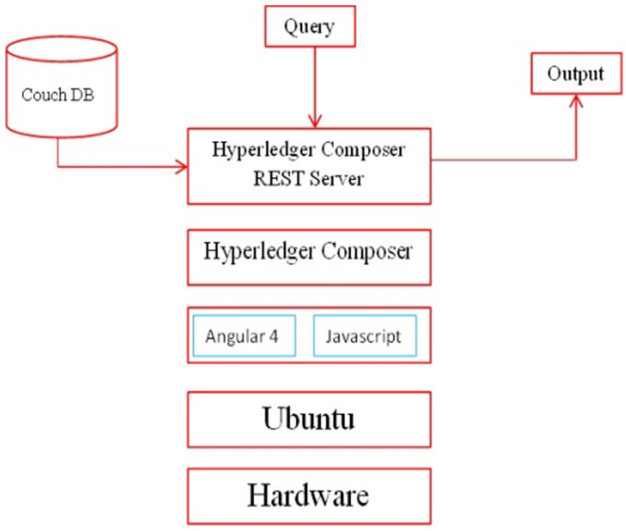

Figure 6.1: System Design

The model used the following hardware:

1. Linux-based Operating System (supporting docker), ie, Ubuntu
2. 4GB RAM
3. 1TB Hard Disk
4. Dual-Core Intel i5

The model uses Hyperledger Composer to build a network on Hyperledger Fabric and Hyperledger Composer REST Server to query and send response via Javascript and Angular 4 based web application. The database is maintained in Couch DB.

6.3 Graphical User Interface

User interface forms a vital part of any application. Web application seems apt for the proposed model.

The participants, doctor and the patient, has their respective home pages designed. The doctor, let's say, can view his/her profile and the patients' past medical records with a responsive grid. When updation of the visit is needed, the page navigates to another page asking for the needed details.

Similarly, the patient can view his/her profile and medical records. Additionally, the patient may revoke the permission to view the information from any doctor at any point of time and while giving permission to a doctor navigates to another page to enter in respective information.

6.4 Class Diagram and Classes

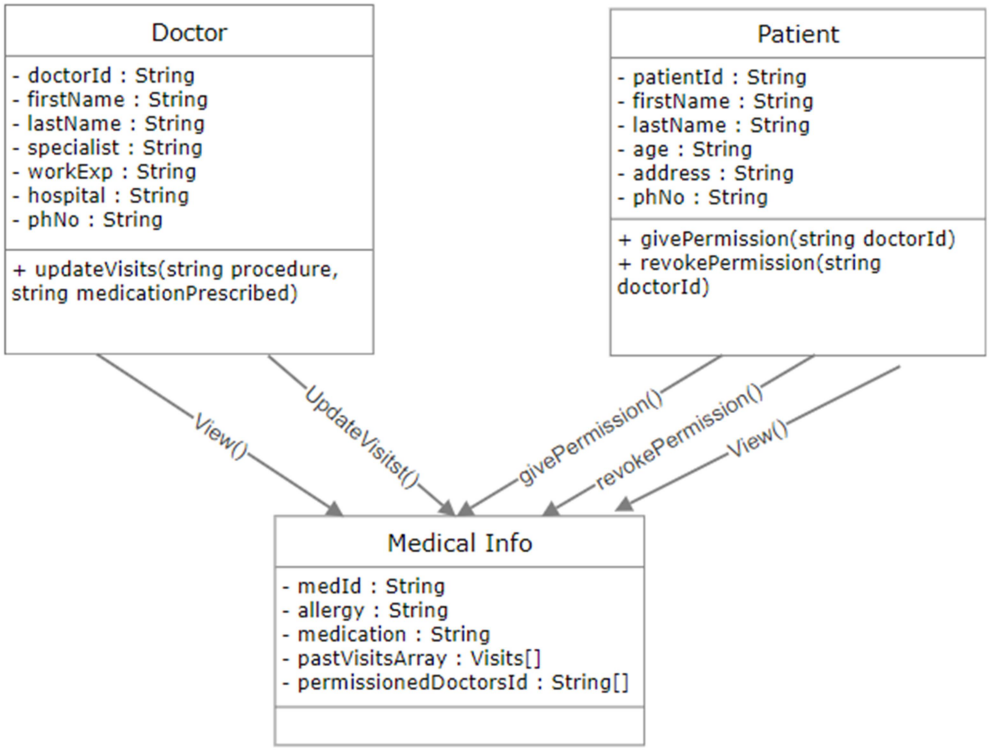

Figure 6.2: Class Diagram

The model has the following classes:

Doctor: This class stores different properties of the class such as name, specialisation, hospital currently working at, contact details and work experience. The method is updateVisit().

Visits: This class stores the details related to a visit of a patient to a medical professional.

MedicalInfo: The class stores the current medication prescribed and the past medical records of the patient.

Patient: This class stores the details of the patient such as name, age and contact details. The methods are givePermission() and revokePermission().

The fig 6.2 illustrates the class dependencies and different API calls, the doctor and the patient makes wrt to the medicalInfo.

6.5 Sequence Diagram

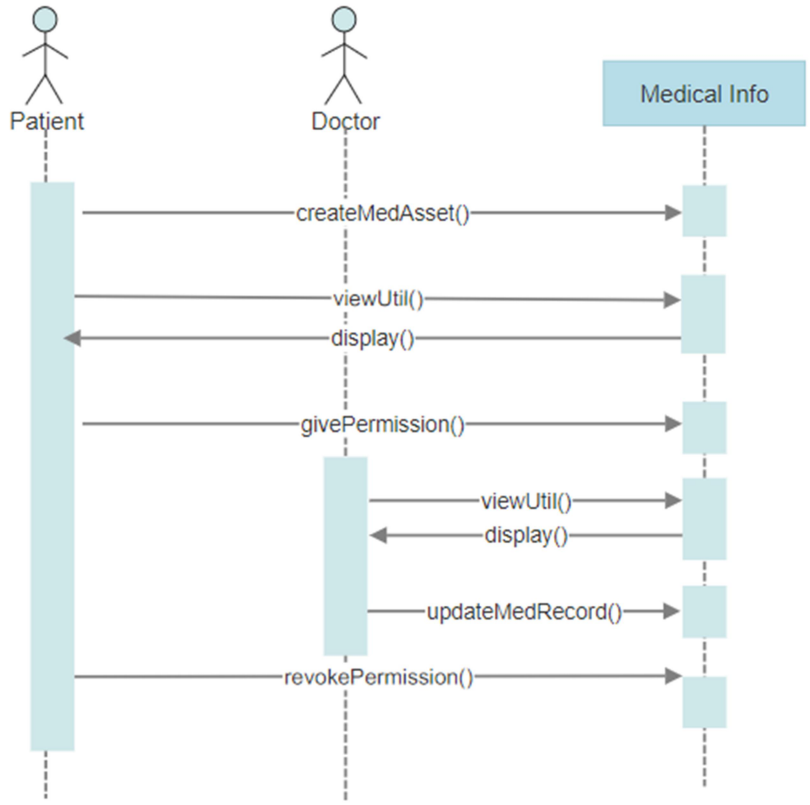

Figure 6.3: Sequence Diagram

The fig 6.3 illustrates the sequence diagram where a doctor and the patient makes respective calls to the medicalInfo and also the direction of flow of API calls.

6.6 Flow of Activities

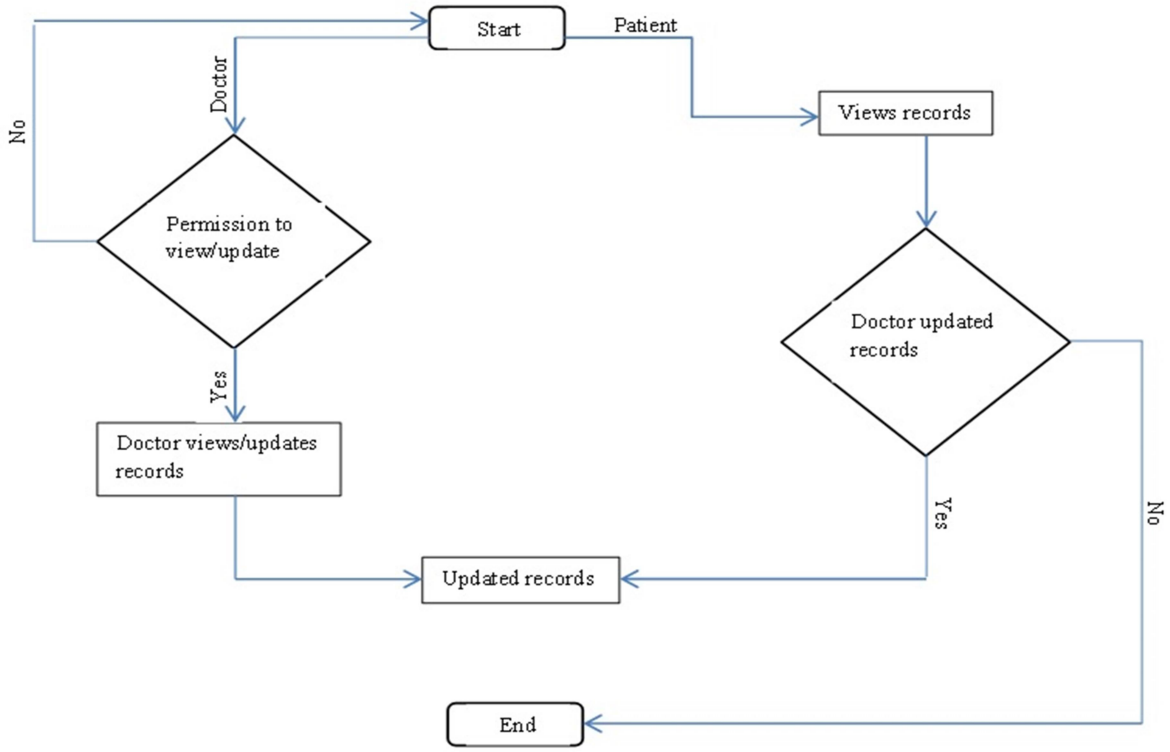

Figure 6.4: Flow of Activities

6.7 Conclusion

The overall view of the system has been provided an abstraction of how the components are interacting with each other can be ascertained. The architecture diagram, sequence diagram and the data flow diagram show the flow of data and interactions among the components which form the vital part for the coding process. A proper understanding of the components helps in the construction of modules.

VII. IMPLEMENTATION

7.1 Tools Introduction

7.1.1 Hyperledger Composer

Hyperledger Composer is an extensive, open development toolset and framework to make developing blockchain applications easier. One can use Composer to rapidly develop use cases and deploy a blockchain solution in weeks rather than months. Composer allows to model the business network and integrate existing systems and data with the blockchain applications. Hyperledger Composer supports the existing Hyperledger Fabric blockchain infrastructure and runtime, which supports pluggable blockchain consensus protocols to ensure that transactions are validated according to policy by the designated business network participants.

7.1.2 Docker

Docker is an open platform for developing, shipping, and running applications. Docker enables to separate the applications from one's infrastructure so the software can be delivered quickly. By taking advantage of Docker's methodologies for shipping, testing, and deploying code quickly, significantly reduce the delay between writing code and running it in production. Docker provides the ability to package and run an application in a loosely isolated environment called a container. The isolation and security allow you to run many containers simultaneously on a given host. Containers are lightweight because they don't need the extra load of a hypervisor, but run directly within the host machine's kernel.

7.1.3 Hyperledger Composer REST server

Hyperledger Composer can be integrated with existing systems by using a Loopback API. Integrating existing systems allows you to pull data from existing business systems and convert it to assets or participants in a Composer business network. Hyperledger Composer includes a standalone Node.js process that exposes a business network as a REST API. The LoopBack framework is used to generate an Open API, described by a Swagger document. The REST server can be configured to subscribe to events emitted from a deployed business network, and publish those events to client applications. The REST server can be configured to multiple user mode too. Multiple user mode permits clients of the REST server to provide their own Blockchain identities for digitally signing transactions. This enables the business network to differentiate between different clients of the REST server.

7.1.4 Angular 4

Angular is a Javascript framework to build more interactive web apps. It is designed for both Web, Desktop and Mobile platforms. While, we create apps using HTML, CSS and Javascript, Angular requires us to know Typescript, kind of stricter version of Javascript provided with OOPS features.

7.2 Technology Introduction

7.2.1 Blockchain

A blockchain is a time-stamped series of an immutable record of data that is managed by a cluster of computers not owned by any single entity. Each of these blocks of data (i.e. block) are secured and bound to each other using cryptographic principles (i.e. chain). The blockchain network has no central authority. Since it is a shared and immutable ledger, the information in it is open for anyone and everyone to see. Hence, anything that is built on the

blockchain is by its very nature transparent and everyone involved is accountable for their actions.

7.2.2 Cloud Computing - AWS

Amazon Web Services(AWS) is a cloud service from Amazon, which provides services in the form of building blocks, these building blocks can be used to create and deploy any type of application in the cloud. These services or building blocks are designed to work with each other, and result in applications which are sophisticated and highly scalable.

7.2.3 Javascript & Typescript

JavaScript (JS) is a lightweight interpreted or just-in-time compiled programming language with first-class functions. While it is most well-known as the scripting language for Web pages, many non-browser environments also use it, such as Node.js, Apache CouchDB and Adobe Acrobat. JavaScript is a prototype-based, multi-paradigm, dynamic language, supporting object-oriented, imperative, and declarative (e.g. functional programming) styles.

JavaScript was introduced as a language for the client side. The development of Node.js has marked JavaScript as an emerging server-side technology too. However, as JavaScript code grows, it tends to get messier, making it difficult to maintain and reuse the code. Moreover, its failure to embrace the features of Object Orientation, strong type checking and compile-time error checks prevents JavaScript from succeeding at the enterprise level as a full-fledged server-side technology. TypeScript was presented to bridge this gap.

7.3 Overall view of the project in terms of implementation

7.3.1 Application

This block represents the interaction between the user and the application. It is responsible for interacting with the participants of the network, namely, doctor and patient as follows:

Patient: The patient can track down his/her medical records ranging from regular checkup to medical procedures for any health related issues. The patient is in control of who gets the access to his medical records and can revoke the permissions at will.

Doctor: The doctor may view a patient's past medical records and update his/her visit if any. The doctor is able to view and update provided the respective patient has given the permission to.

7.3.2 Server

This block represents the interaction between the application and the Composer REST Server and the application. The application sends read and write requests to the server via REST APIs defined by the Composer REST Server. Couch DB is used the blockchain network to store the data generated on every visit of the patient to a medical professional.

7.4 Explanation of Algorithm and how it is been implemented

Blockchain-based medical records of 2 participants doctor and patient. The medical record is the main asset in the blockchain network. This medical record asset is owned by a patient, therefore every medical record will have one patient but patients can have multiple or no medical record. The medical record has to view-only access to the patient who owns the medical record, view and update access to a doctor who is given permissions and for others, no permission is given There are three basic transactions defined in the medical record system:

1. *Give Permission*-Permission transaction grants the permissions to doctor to view and update the patient's medical record. This transaction is created patient on his medical record.
   Code:

```
void givePermission(transaction) {
    /* `transaction` contains the `medicalAsset_id` and the `doctor_id`*/
```

```
        Update the `medicalAsset` on blockchain network by appending the
        `doctor_id` to `permissionedDoctorsId`
    }
```

2. *Revoke Permission*-Permission transactions revokes permission from a doctor to view and update the patient's medical record. This transaction is created patient on his medical record.

   Code:

```
    void revokePermission(transaction) {
        /* `transaction` contains the `medicalAsset_id` and the `doctor_id`*/
        Update the `medicalAsset` on blockchain network by removing the
        `doctor_id` from `permissionedDoctorsId`
    }
```

3. *Update Visit*-Visit transaction is used by doctors to update the observations made during the patients into the medical records which they access to. This transaction is created doctor on the medical records he has access to.

   Code:

```
    void updateVisit(transaction) {
        /* `transaction` contains the medicalAsset_id and the details of the
        new visit*/
        Update the `medicalAsset` on blockchain network by adding new visit
        details to `pastVisitsArray`
        Update the current medication of the patient
    }
```

7.5 Information about the implementation of Modules

A typical system model for Blockchain based healthcare records management consists of three modules:

1. BlockChain Network

2. Rest Server
3. Front End.

7.5.1 BlockChain Network

The Blockchain network was built using Hyperledger composer playground. Developing in Hyperledger Composer to digitize business networks. consists of multiple participants (like doctors, patients). We defined the data model for different participants and transactions among them. Here in this blockchain network, we have 3 transactions

1. Give-Permission transaction grants the permissions to doctor to view and update the patient's medical record.
2. Revoke-Permission transactions revoke permission from a doctor to view and update the patient's medical record.
3. The update-visit transaction is used by doctors to update the observations made during the patients the medical records which they have access to.

7.5.2 REST Server

REST is an interface between UI and Blockchain using HTTP to obtain data. The Hyperledger Composer REST server is used to generate a REST interface to the blockchain business network developed. We have created two - One server running on admin mode is used to validate and update users during login and signup process. Another one is used as a server which can be used by the users to query for data and create new transactions.

7.5.3 Front End

Angular 4 was used to build web applications. Introduction, Authentication (Signin/Signup), Doctor and Patient homepages were developed. eg- Grid, Responsive Form, Cards, etc were used to develop the basic structure of the UI.

7.6 Conclusion

The application has been carefully subdivided into modules such that isolation between them is ensured. Changes made to one module do not significantly affect the functioning of the other modules. The application has been designed in a manner that it is and does not burden the user. The blockchain network runs on the Composer REST Server giving a hasslefree experience. Overall the user is provided with a fluid application with relatively low response times.

VIII. TESTING

8.1 Introduction

Software testing is an activity to check the performance of the software model. and to ensure that the software system is defect free. It involves the execution of a software component or system component to evaluate one or more properties of interest. Application testing is a process through which the functionality, usability, and consistency of the entire application are tested. Another major component of software testing is performance testing. This subdomain encompasses various aspects like latency, response times and optimizations.

8.2 Testing tools and environment

8.2.1 Blockchain network testing

Blockchain network testing encompasses the performance testing of the network. There are various factors to see the performance indicators, such as throughput, transaction latency, resource utilisation, etc. This gives an estimate of the performance of the network and applicability of blockchain network.

HyperLedger Caliper

Hyperledger Caliper is a blockchain benchmark tool and one of the Hyperledger projects hosted by The Linux Foundation. Hyperledger Caliper allows users to measure the performance of a specific blockchain implementation with a set of predefined use cases. Hyperledger Caliper will produce reports containing a number of performance indicators, such as TPS (Transactions Per Second), transaction latency, resource utilisation etc. The intent is for Caliper results to be used by other Hyperledger projects as they build out their frameworks and as a reference in supporting the choice of a blockchain implementation suitable for a user's specific needs. Hyperledger Caliper was initially contributed by

developers from Huawei, Hyperchain, Oracle, Bitwise, Soramitsu, IBM and the Budapest University of Technology and Economics.

8.3 Test Cases

Test cases have been implemented for 5 functionalities in the application developed:

1. Login: login is used to login registered users to either doctor home page or patient page.
2. Register: register is used to new users to signup to doctor or patient and give their respective pages.
3. Give-Permission: givePermission () gives access to a given doctor, identified by doctorId, a number of times.
4. Revoke-Permission: *revokePermission* () revokes access from a given doctor, identified by doctorId.
5. Update-Visit: A doctor updates the visit using *updateVisit*() for a given patient, identifies by patientId, a number of times.

SL. No	Test Case Description	Test Data	Expected Result	Actual Result	Pass/Fail
1	Check Login for Patient	100 (valid User Id)	Successfully Login and Land to Patient Home	Successfully Logged in and Landed to Patient Home	Pass
2	Check Login for Patient	105 (Invalid User Id)	Unsuccessfully Login	Error: User Id does not Exists	Pass
3	Check Login for Doctor	1000 (valid User Id)	Successfully Login and	Successfully Logged in	Pass

			Land to Doctor Home	and Landed to Doctor Home	
4	Check Login for patient	1005 (valid User Id)	Unsuccessfully Login	Error: User Id does not Exists	Pass
5	Check Registration of Patient	Id 101 First Name: Doc Last Name: Raj Ph No: 1234567890 Age: 12 Address: Bengaluru (valid new user)	Successfully Register and Land to Patient Home	Successfully Registered and Landed to Patient Home	Pass
6	Check Registration of Patient	Id 100 First Name: Doc Last Name: Krish Ph No: 1234567890 Age: 12 Address: Bengaluru (user already Exists)	Unsuccessful Register	Error: User Id already exists cannot create card.	Pass
7	Check Registration of Doctor	Id 1001 First Name: Nikhil Last Name: Kumar Ph No: 1234567890 Specialist in: cardio Work Experience: 5 Hospital: Ramaiah (valid new user)	Successfully Register and Land to Doctor Home	Successfully Registered and Landed to Doctor Home	Pass
8	Check Registration of Patient	Id 1000 First Name: Virat Last Name: Gowda	Unsuccessful Register	Error: User Id already exists cannot	Pass

		Ph No: 1234567890 Specialist in: cardio Work Experience: 5 Hospital: Ramaiah (user already Exists)		create card.	
9	Check Patient Give Permission to Doctor Id	Id 1000 (valid Id)	Successfully give Permission	Successfully give Permission	Yes
10	Check Patient Give Permission to Doctor Id	Id 1005 (Invalid Id)	Unsuccessful Transaction	Error: Asset doesn't exist	Yes
11	Check Patient Revoke Permission to Doctor Id	Id 1000 (valid Id)	Successfully revoke Permission	Successfully revoke Permission	Yes
12	Check Update Visit	Id 100 Procedure: Xray Medication: Dolo (valid Entries)	Successfully update in chain code	Successfully updated in chain code	Yes
13	Check Update Visit	Id 100 Procedure: Xray Medication: Dolo (Doctor does not have permission to update)	Should not allow to update in chain code	Error: Authorization failure	Yes
14	Check Update Visit	Id 105 Procedure: Xray Medication: Dolo (invalid medical id)	Unsuccessful update in chain code	Error: Asset doesn't exist	Yes

Table 8.1: Test Cases

IX. RESULTS & PERFORMANCE ANALYSIS

9.1 Result Snapshots

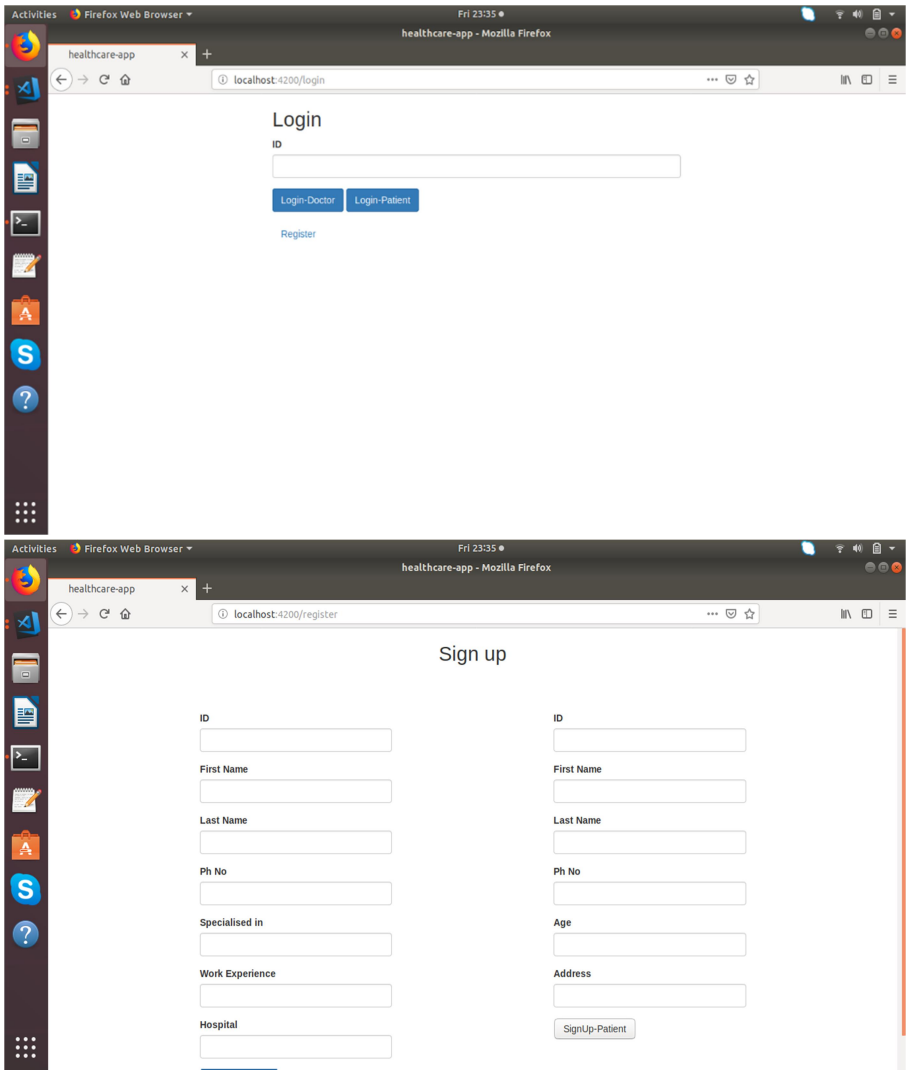

Figure 9.1: Signin/Signup Pages

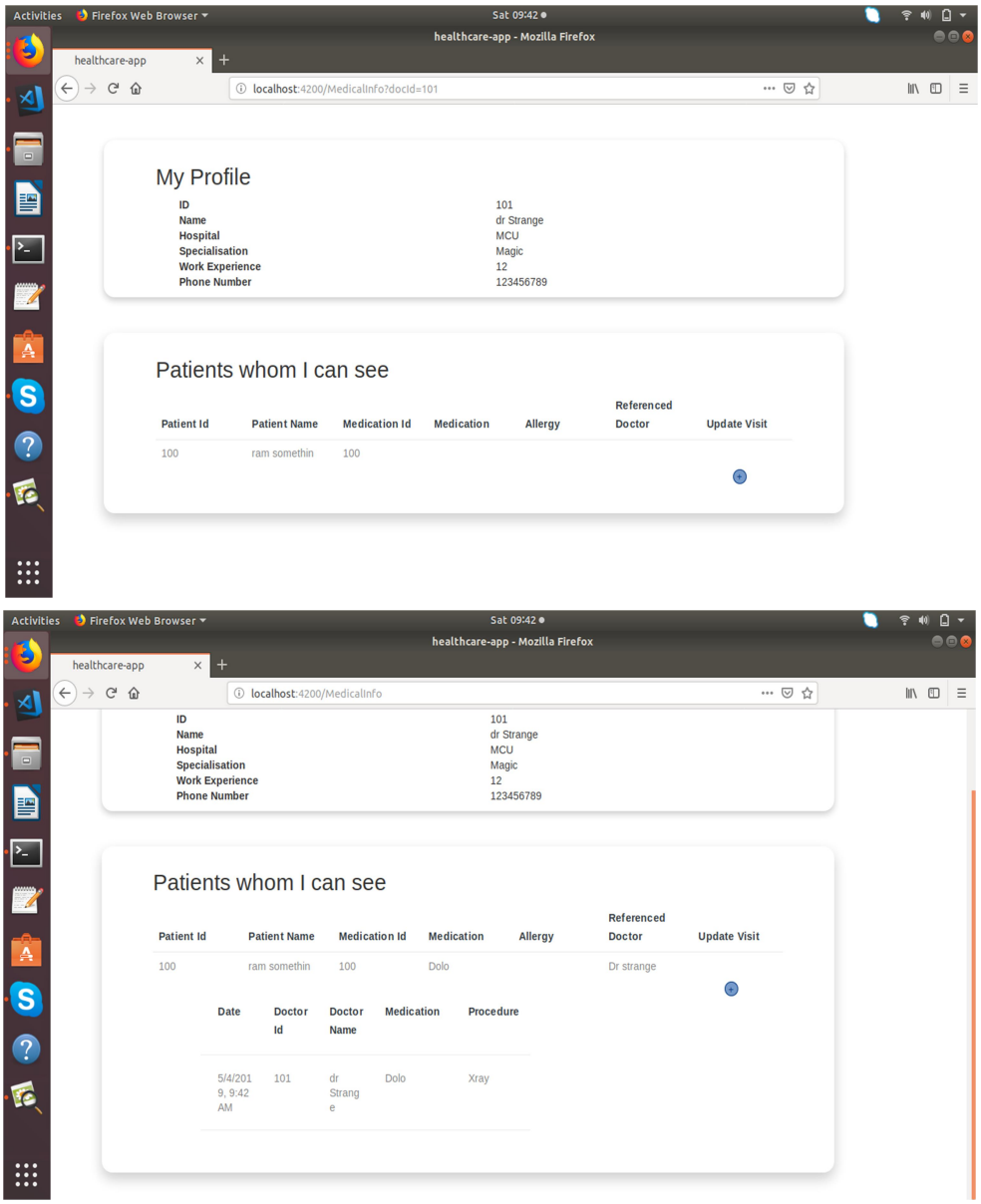

Figure 9.2: Doctors page

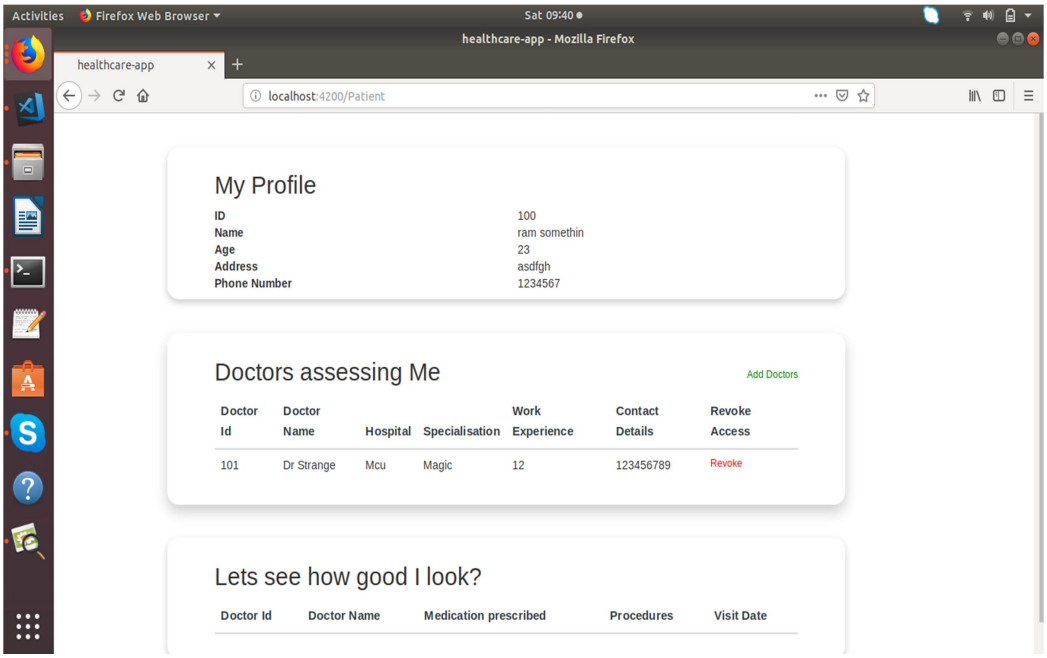

Figure 9.3: Patient page

9.2 Performance analysis

Using hyperledger Caliper benchmark testing could be done for the hyperledger network developed and know the performance in different fabric architecture. The parameters used for the testing the network:

- **Transaction throughput**: transaction throughput is the rate commits valid transaction in the defined period of time.
- **Transaction latency**: transaction latency is the time taken between when the transaction is submitted and when the transaction is confirmed committed across the network.
- **Send Rate**: send rate is no of transactions sent per second.

S. NO	Type of Hyperledger Fabric architecture	No of Txn	Send Rate	Max Latency	Min Latency	Avg Latency	Throughput
1	2 org 1 peer	10	11.1 tps	1.56s	0.87s	1.30s	4.2 tps
2	2 org 1 peer	50	51.0 tps	5.03s	2.02s	3.75s	8.7 tps
3	2 org 1 peer	100	10.1 tps	8.72s	1.13s	5.07s	5.5 tps
4	2 org 2 peer	10	11.1 tps	3.03s	1.43s	2.20	2.8 tps
5	2 org 2 peer	50	50.9 tps	9.98 s	3.58s	7.49 s	4.8 tps
6	2 org 2 peer	100	10.1 tps	18.21s	1.38s	9.21s	3.6 tps
7	3 org 1 peer	10	11.1tps	2.24 s	1.03s	1.59s	3.3 tps
8	3 org 1 peer	50	10.2 tps	7.67s	1.19s	4.64s	4.2tps
9	3 org 1 peer	100	10.1 tps	16.73s	0.97s	10.29s	3.9 tps

Table 9.1: Performance Analysis

X. CONCLUSION & FUTURE WORK

The main objective of the blockchain based healthcare records management system is to provide limited access to the patient's medical logs which is achieved via Hyperledger Composer, tool and framework built over Hyperledger Fabric thus, building a permissioned blockchain network. The patient decides on who gets access to his/her medical records via doctorId and may revoke access permission for the same. On the other hand, the doctor may access the patient's medical history to read and analyse the patient's situation better, given all medical history logs are updated timely on the blockchain network. The doctor may update the current visit with prescriptions and medical procedures, if any, on blockchain.

This project has a scope of being scalable and help the medical industry to work more efficiently. The patients can visualise and analyse the validation of doctors' expertise in the given field better by analysing the patients' records, with consent, and proceed accordingly.

Also, the insurance industry may take advantage of this immutable ledger while taking in more clients by having a look at this ledger to analyse the current medical condition of the client. The clients on the other hand, may use these records to back up their claim to insurance money and the company may validate using the same.

XI. REFERENCES

[1] Alexaki, Sofia, George Alexandris, Vasilis Katos, and E. Nikolaos Petroulakis. "*Blockchain-based Electronic Patient Records for Regulated Circular Healthcare Jurisdictions*." In 2018 IEEE 23rd International Workshop on Computer Aided Modeling and Design of Communication Links and Networks (CAMAD), pp. 1-6. IEEE, 2018.

[2] Kleinaki, Athina-Styliani, Petros Mytis-Gkometh, George Drosatos, Pavlos S. Efraimidis, and Eleni Kaldoudi. "A blockchain-based notarization service for biomedical knowledge retrieval." *Computational and structural biotechnology journal* 16 (2018): 288-297.

[3] Zhang, Peng, Jules White, Douglas C. Schmidt, Gunther Lenz, and S. Trent Rosenbloom. "Fhirchain: applying blockchain to securely and scalably share clinical data." *Computational and structural biotechnology journal* 16 (2018): 267-278.

[4] Dagher, Gaby G., Jordan Mohler, Matea Milojkovic, and Praneeth Babu Marella. "Ancile: Privacy-preserving framework for access control and interoperability of electronic health records using blockchain technology." *Sustainable cities and society* 39 (2018): 283-297.

[5] Chen, Lanxiang, Wai-Kong Lee, Chin-Chen Chang, Kim-Kwang Raymond Choo, and Nan Zhang. "Blockchain based searchable encryption for electronic health record sharing." *Future Generation Computer Systems* (2019).

[6] Hussein, Ahmed F., N. ArunKumar, Gustavo Ramirez-Gonzalez, Enas Abdulhay, João Manuel RS Tavares, and Victor Hugo C. de Albuquerque. "A medical records managing and securing blockchain based system supported by a genetic algorithm and discrete wavelet transform." *Cognitive Systems Research* 52 (2018): 1-11.

[7] Badr, Shaimaa, Ibrahim Gomaa, and Emad Abd-Elrahman. "Multi-tier Blockchain framework for IoT-EHRs systems." *Procedia Computer Science* 141 (2018): 159-166.

[8] Griggs, Kristen N., Olya Ossipova, Christopher P. Kohlios, Alessandro N. Baccarini, Emily A. Howson, and Thaier Hayajneh. "Healthcare blockchain system using smart contracts for secure automated remote patient monitoring." *Journal of medical systems* 42, no. 7 (2018): 130.

[9] Chen, Yi, Shuai Ding, Zheng Xu, Handong Zheng, and Shanlin Yang. "Blockchain-based medical records secure storage and medical service framework." *Journal of medical systems* 43, no. 1 (2018): 5.

[10] Li, Hongyu, Liehuang Zhu, Meng Shen, Feng Gao, Xiaoling Tao, and Sheng Liu. "Blockchain-based data preservation system for medical data." *Journal of medical systems* 42, no. 8 (2018): 141.

[11] Sun, You, Rui Zhang, Xin Wang, Kaiqiang Gao, and Ling Liu. "A Decentralizing Attribute-Based Signature for Healthcare Blockchain." In *2018 27th International Conference on Computer Communication and Networks (ICCCN)*, pp. 1-9. IEEE, 2018.

[12] https://hyperledger-fabric.readthedocs.io/en/release-1.4/

[13] https://hyperledger.github.io/composer/v0.19/introduction/introduction.html

[14] https://hyperledger.github.io/composer/v0.19/integrating/getting-started-rest-api

[15]https://hyperledger.github.io/caliper/docs/1_Getting_Started.html

[16] https://medium.com/coinmonks/getting-started-with-hyperledger-composer-34cb7228d44c

[17] https://medium.com/@CazChurchUk/developing-multi-user-application-using-the-hyperledger-composer-rest-server-b3b88e857ccc

[18] https://medium.com/coinmonks/building-a-blockchain-application-using-hyperledger-fabric-with-angular-frontend-part-2-22ef7c77f53