MINI PROJECT

(2021-22)

“eHospital using Blockchain”

Project Report

Institute of Engineering & Technology

Submitted By -

Yogesh Yadav(191500943)

Under the Supervision Of

Mrs. Harvinder Kaur

Technical Trainer

Department of Computer Engineering & Applications

Department of Computer Engineering and Applications
GLA University, 17 km. Stone NH#2, Mathura-Delhi Road,
Chaumuha, Mathura – 281406 U.P (India)

Declaration

I hereby declare that the work which is being presented in the Bachelor of technology. Project **“eHospital using Blockchain”**, in partial fulfillment of the requirements for the award of the ***Bachelor of Technology*** in Computer Science and Engineering and submitted to the Department of Computer Engineering and Applications of GLA University, Mathura, is an authentic record of my own work carried under the supervision of **Mrs. Harvinder Kaur, Technical Trainer, Dept. of CEA,GLA University.**

The contents of this project report, in full or in parts, have not been submitted to any other Institute or University for the award of any degree.

Sign: *YogeshYadav*

Name of Candidate: Yogesh Yadav

University Roll No.:191500943

Department of Computer Engineering and Applications
GLA University, 17 km. Stone NH#2, Mathura-Delhi Road,
Chaumuha, Mathura – 281406 U.P (India)

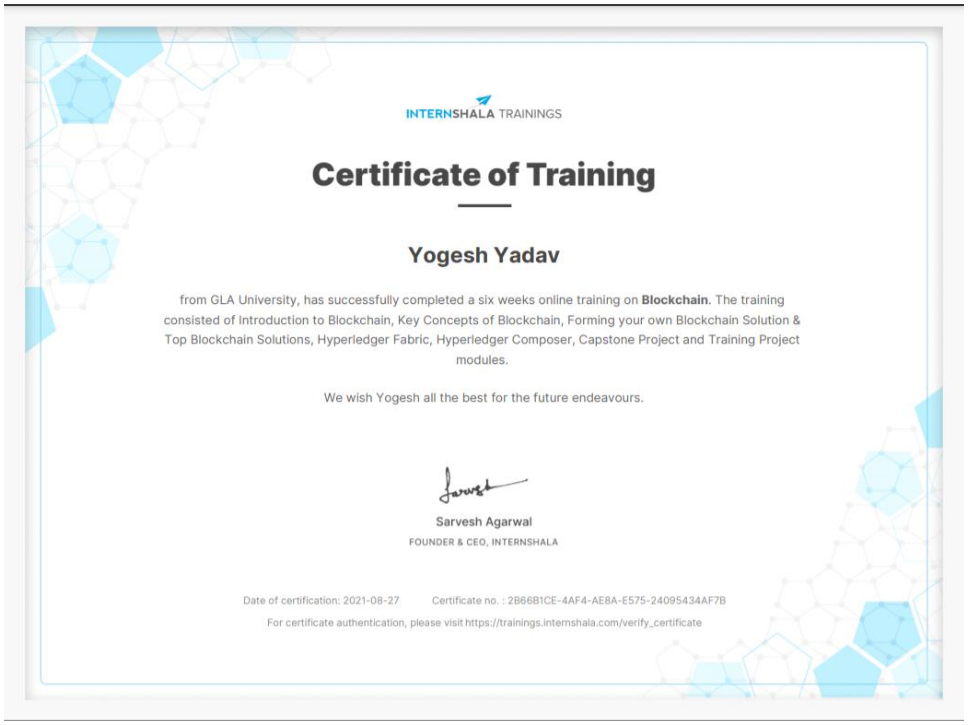
Certificate

This is to certify that the project entitled "eHospital using Blockchain", carried out in Mini Project – I Lab, is a Bonafide work by Yogesh Yadav and is submitted in partial fulfillment of the requirements for the award of the degree Bachelor of Technology (Computer Science & Engineering).

Signature of Supervisor:

Name of Supervisor: Mrs. Harvinder Kaur
Date:

Training Certificates



INTERNSHALA TRAININGS

Certificate of Training

Yogesh Yadav

from GLA University, has successfully completed a six weeks online training on **Blockchain**. The training consisted of Introduction to Blockchain, Key Concepts of Blockchain, Forming your own Blockchain Solution & Top Blockchain Solutions, Hyperledger Fabric, Hyperledger Composer, Capstone Project and Training Project modules.

We wish Yogesh all the best for the future endeavours.

Sarvesh Agarwal
FOUNDER & CEO, INTERNSHALA

Date of certification: 2021-08-27 Certificate no. : 2B66B1CE-4AF4-AE8A-E575-24095434AF7B

For certificate authentication, please visit https://trainings.internshala.com/verify_certificate

Department of Computer Engineering and Applications
GLA University, 17 km. Stone NH#2, Mathura-Delhi Road,
Chaumuha, Mathura – 281406 U.P (India)

ACKNOWLEDGEMENT

Presenting the ascribed project paper report in this very simple and official form, we would like to place my deep gratitude to GLA University for providing us the instructor Mr Mandeep Singh, our technical trainer and supervisor.

He has been helping us since Day 1 in this project. He provided us with the roadmap, the basic guidelines explaining on how to work on the project. He has been conducting regular meeting to check the progress of the project and providing us with the resources related to the project. Without his help, we wouldn't have been able to complete this project.

And at last but not the least we would like to thank our dear parents for helping us to grab this opportunity to get trained and also my colleagues who helped me find resources during the training.

Thanking You

Sign: *YogeshYadav*

Name of Candidate: Yogesh Yadav
University Roll No.:191500327

ABSTRACT

In this project, I am creating an Blockchain based system of the hospital, basically a Healthcare management system which we have named eHospital using blockchain. This management system will provide us a platform for patient, doctors and hospital admin to view and interact with various options to add a information in the it. I have provided a login module for admin, admin will able to login and can-do things like adding new doctor, updating record of previous faculty, uploading notices and pdfs of study material for students, admin can also add the images and videos of events and fests in the campus. We have used Kotlin and Android studio for creating this app. Further. The app also has a complete User Interface attached to the firebase a perfect login system with email id and password.

"Blockchain technology has long been associated with cryptocurrencies such as Bitcoin, but there is so much more that it has to offer, particularly in how public and private organizations secure, share and use data," comments Steve Davies, Global Leader, Blockchain and Partner, PwC UK.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER-1

INTRODUCTION

1.1 CONTEXT

This management system has been submitted in partial fulfilment of the requirements for the award of the degree of Bachelor of Technology in Computer Science and Engineering at GLA University, Mathura supervised by Mrs. Harvinder Kaur. This project has been completed approximately in 50 days and has been executed in modules, meetings have been organized to check the progress of the work and for instructions and guidelines.

1.2 MOTIVATION

Today's technology is offering are to improve security, user experience and other aspects of healthcare sector. These benefits were offered by Electronic Health Record (EHR) and Electronic Medical Record (EMR) systems. However, they still face some issues regarding the security of medical records, user ownership of data, data integrity etc. The solution to these issues could be the use of a novel technology, i.e., Blockchain. This technology offers to provide a secure, temper-proof platform for storing medical records and other healthcare related information.

1.3 OBJECTIVE

Blockchain is considered to have great potential is healthcare. Understanding the pertinence and importance of blockchain in healthcare, in 2016, the Office of the National Coordinator for Health Information Technology (ONC), composed an ideation challenge for requesting white papers on the potential utilization of blockchain in healthcare. This challenge brought about a few proposed healthcare applications for blockchain.

we focus on the most important studies classified by several use cases such as electronic medical records, remote patient monitoring, pharmaceutical supply chain and health insurance claims.

1.4 EXISTING SYSTEM

Healthcare sector used paper-based system to store the medical records, i.e., using handwritten mechanism. This paper-based medical record system was inefficient, insecure, unorganized and was not temper-proof. It also faced the issue of data- duplication and redundancy as all the institutions that patient visited had various copies of patient's medical records.

1.5 SOURCES

The source of our project (including all the project work,hjhyt documentations and presentations) will is available at the following link https://github.com/JanviPangoriya/Book_finder_app_mini_projecty

CHAPTER -2 SOFTWARE REQUIREMENT ANALYSIS

2.1 IMPACT OF THIS SYSTEM ON DAILY LIFE

Well Indian health care system and hospital are not much advance that’s why we all face so many problems in covid. So blockchain based health care system will help the government and hospital to identify the patient’s needs and trace the records of the patients. So, I have created this system because when I saw that hospital used old types of methods to store data like paper and centralized methods where data lost and stolen are normal things. Here, blockchain provide the security and mutability of the data. So, data is safe from everything.

2.2 PROBLEM STATEMENT

The decentralized nature of sensitive health information can bring about situations where timely information is unavailable, worsening health outcomes. Furthermore, as patient involvement in health care increases, there is a growing need for patients to access and control their data. Blockchain is a secure, decentralized online ledger that could be used to manage electronic health records (EHRs) efficiently, therefore with the potential to improve health outcomes by creating a conduit for interoperability. Blockchain possesses the potential to revolutionize the Indian healthcare industry by placing the patient at the center of the ecosystem, amplifying interoperability, privacy, and security of health data. The technology is set to equip the highly data-driven industry with a more advanced Health Information Exchange (HIE) model that could genuinely transform electronic medical records, making them significantly more secure, efficient, and disintermediated.

2.3 HARDWARE AND SOFTWARE REQUIREMENTS

Hardware Requirement

- Processor :intel i5
- Operating System :Linux Ubuntu
- RAM : 8 GB (or higher)
- Hard disk : 256GB

Software Requirement

- Software used: VS code , Remix , ganache.
- Language used: Solidity, JavaScript, HTML

2.4 MODULES AND FUNCTIONALITIES

Doctor Registration: - It is registration is done by hospital no other can do it. In it we provide details like doctor name, doctor specification, doctor phone number, doctor address.

Patient Registrations: - In this we give the patient details like name of the patient, age gender, height, weight, address, phone number, email id, date of visit.

Medical Records: - It is done by using ERC21 non fungible token. Here, we get the patient medical report by using its id along with date. It will show insurance details, present illness details, past illness details, provisional diagnosis details, treatment summary. It is only updated by doctor.

Patient examine details: - In this we get a detail of reports and tests and general examination by using patient id and date. We can update the information also here. Information like investigations of blood tests, urine tests, ECG, MRI scan, CT scan, Xray and other lab tests. In general examination we have Nourishment, eyes, tongue, pulse, blood pressure, temperature, respiratory rate. In systemic examination we get the information of the CVS, CNS, RS, abdomen system.

2.5 eHosspital using Blockchain

In this Management system doctor, patient and hospital keep the records of the patient. Well, Covid situation tells that currently management system of hospital is good and it should be changed because of it so many lives were died. Now this management system helps the hospital to maintain the records of the patient according to which patient who have serious condition can get treatment easily.

CHAPTER-3 TECHNOLOGY USED

3.1 Remix

Remix IDE is an open-source web and desktop application. It fosters a fast development cycle and has a rich set of plugins with intuitive GUIs. Remix is used for the entire journey of contract development as well as being a playground for learning and teaching Ethereum.

Remix IDE is part of the Remix Project which is a platform for development tools that use a plugin architecture. It encompasses sub-projects including Remix Plugin Engine, Remix Libs, and of course Remix-IDE.

Remix IDE is a powerful open-source tool that helps you write Solidity contracts straight from the browser.

It is written in JavaScript and supports both usage in the browser, in the browser but run locally and in a desktop version.

Remix IDE has modules for testing, debugging and deploying of smart contracts and much more.

Remix-IDE is available at remix.ethereum.org and more information can be found in these docs. Our IDE tool is available at our GitHub repository.

I created a smart contract on remix because it provides the testing functionality and auto compiling of the code.

3.2 VS CODE

Visual Studio Code is a streamlined code editor with support for development operations like debugging, task running, and version control. It aims to provide just the tools a developer needs for a quick code-build-debug cycle and leaves more complex workflows to fuller featured IDEs, such as Visual Studio IDE.

I used vs code for making interface using HTML, java script.

3.3 TOOLS AND LANGUAGES

Tools used to build are:-

- **Solidity**: **Solidity** is a brand-new programming language created by the **Ethereum** which is the second-largest market of cryptocurrency by capitalization, released in the year 2015 led by Christian Reit Wiessner. Some key features of solidity are listed below:
  - Solidity is a high-level programming language designed for implementing smart contracts.
  - It is statically-typed object-oriented(contract-oriented) language.
  - Solidity is highly influenced by Python, c++, and JavaScript which runs on the Ethereum Virtual Machine (EVM).
  - Solidity supports complex user-defined programming, libraries and inheritance.
  - Solidity is primary language for blockchains running platforms.
  - Solidity can be used to creating contracts like voting, blind auctions, crowdfunding, multi-signature wallets, etc.

.

- Ethereum Virtual Machine(EVM)

Ethereum Virtual Machine abbreviated as EVM is a runtime environment for executing smart contracts in Ethereum. It focuses widely on providing security and execution of untrusted code using an international network of public nodes. EVM is specialized to prevent Denial-of-service attack and confirms that the program does not have any access to each other's state, also ensures that the communication is established without any potential interference.

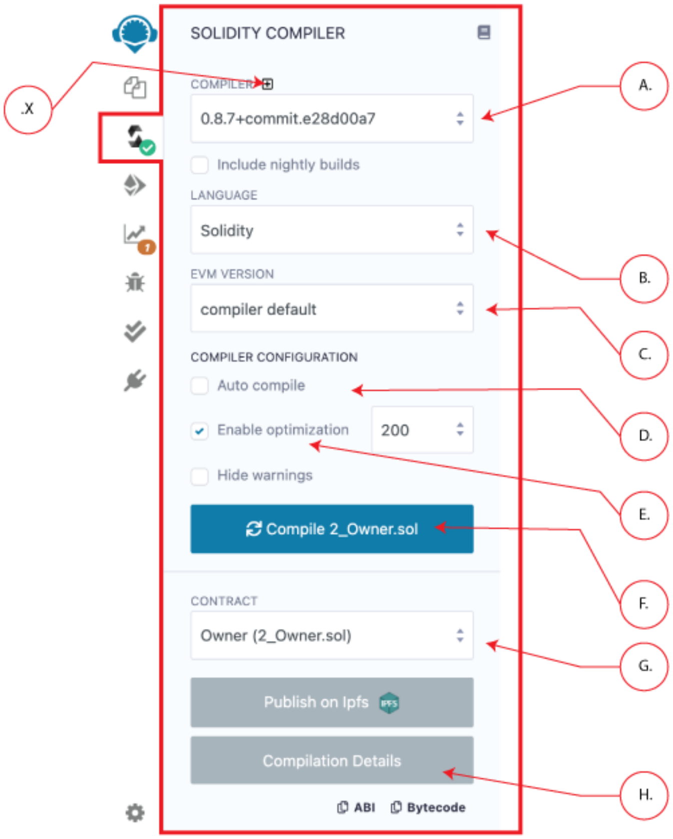
- **HTML: HTML** stands for Hypertext Markup Language. It is used to design web pages using a markup language. HTML is the combination of Hypertext and Markup language. Hypertext defines the link between the web pages.

- **JavaScript:** JavaScript is a cross-platform, object-oriented scripting language used to make webpages interactive (e.g., having complex animations, clickable buttons, popup menus, etc.). There are also more advanced server-side versions of JavaScript such as Node.js, which allow you to add more functionality to a website than downloading files (such as Realtime collaboration between multiple computers). Inside a host environment (for example, a web browser), JavaScript can be connected to the objects of its environment to provide programmatic control over them.

3.4 BASIC TERMINOLOGY

- **An Ethereum Fork:** The "fork selection" dropdown (**C. in image below**) list allows to compile code against a specific **Ethereum hard fork**. The compiler default corresponds to the default hard fork used by a specific version.

To see the name of the hard fork used in the current compilation, click the "Compilation Details" (**H. in image below) button** and in the Metadata section will be a sub-section called **settings**. Open the **settings** to see the hard fork's name.



SOLIDITY COMPILER

COMPILER

0.8.7+commit.e28d00a7

□ Include nightly builds

LANGUAGE

Solidity

EVM VERSION

compiler default

COMPILER CONFIGURATION

□ Auto compile

☑ Enable optimization 200

□ Hide warnings

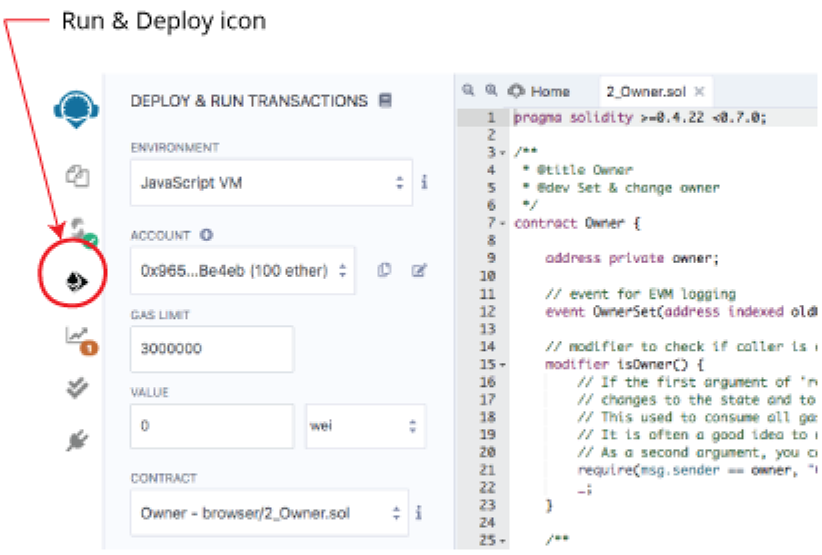
CONTRACT

Owner (2_Owner.sol)

- **Auto Compile**: If a contract has a lot of dependencies, it can take a while to compile - so you use auto compilation at your discretion.

- **Enable optimization :** According to the the Solidity Docs, “the optimizer tries to simplify complicated expressions, which reduces both code size and execution cost, i.e., it can reduce gas needed for contract deployment as well as for external calls made to the contract.”

- **Deploy and Run**: To use this module, you need to have a contract compiled. So, if there is a contract name in the **CONTRACT** select box (the select box is under the VALUE input field), you can use this module. If nothing is there or you do not see the contract you want, you need to select a contract in the editor to make it active, go to a compiler module and compile it, and then come back to Deploy & Run.



Run & Deploy icon

DEPLOY & RUN TRANSACTIONS

ENVIRONMENT: JavaScript VM

ACCOUNT: 0x965...Be4eb (100 ether)

GAS LIMIT: 3000000

VALUE: 0 wei

CONTRACT: Owner - browser/2_Owner.sol

```
pragma solidity >=0.4.22 <0.7.0;

/**
 * @title Owner
 * @dev Set & change owner
 */
contract Owner {

    address private owner;

    // event for EVM logging
    event OwnerSet(address indexed old

    // modifier to check if caller is
    modifier isOwner() {
        // If the first argument of '
        // changes to the state and to
        // This used to consume all ga
        // It is often a good idea to
        // As a second argument, you c
        require(msg.sender == owner, "
        _;
    }

    /**
```

- **Environment**:
- JavaScript VM: All the transactions will be executed in a sandbox blockchain in the browser. This means nothing will be persisted when you reload the page. The JsVM is its own blockchain and on each reload it will start a new blockchain, the old one will not be saved.

  - `Injected Provider`: Remix will connect to an injected web3 provider. `Metamask` is an example of a provider that inject web3.
  - `Web3 Provider`: Remix will connect to a remote node. You will need to provide the URL to the selected provider: geth, parity or any Ethereum client.
-

- **Ganache:** Ganache is used for setting up a personal Ethereum Blockchain for testing your Solidity contracts. It provides more features when compared to Remix. You will learn about the features when you work out with Ganache. Before you begin using Ganache, you must first download and install the Blockchain on your local machine.

- **Account:** Account: the list of accounts associated with the current environment (and their associated balances). On the JsVM, you have a choice of 5 accounts. If using Injected Web3 with MetaMask, you need to change the account in MetaMask.

- **Deploy and AtAddress:** In the image above, the select box is set to **Ballot**. This selects box will contain the list of compiled contracts.

  - **Deploy** sends a transaction that deploys the selected contract. When the transaction is mined, the newly created instance will be added (this might take several seconds).
  Note: If the contract's constructor function has parameters, you need to specify them.
  - **At Address** is used to access a contract that has already been deployed. Because the contract is already deployed, accessing a contract with **AtAddress** does not cost gas.

- **Using the ABI with AtAddress:** The **ABI** is a JSON array which describes the contract's interface.

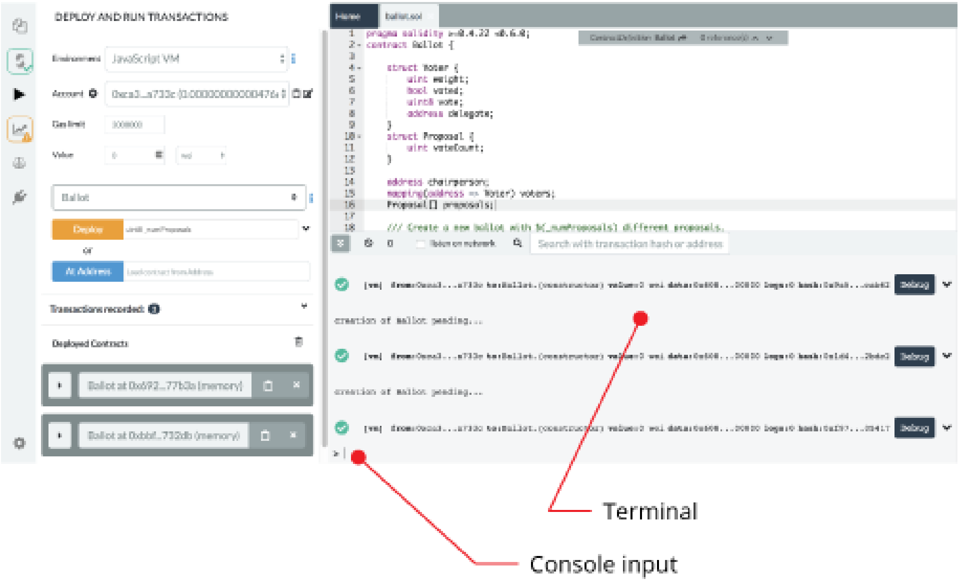
  - To interact with a contract using the ABI, create a new file in Remix with extension *__.abi__ and copy the ABI content to it.
  - Make sure this file is the active tab in the editor. Then, in the field next to `At Address`, input the contract's address and click on `At Address`. If successful, an instance of the contract will appear below - in the list of **Deployed Contracts**.

- **Terminal**:

  Features, available in the terminal:

  - It integrates a JavaScript interpreter and the `web3` object. It enables the execution of the JavaScript script which interacts with the current context. (note that `web3` is only available if the `web provider` or `injected provider` mode is selected).
  - It displays important actions made while interacting with the Remix IDE (i.e., sending a new transaction).

- It displays transactions that are mined in the current context. You can choose to display all transactions or only transactions that refers to the contracts Remix knows (e.g transaction created from the Remix IDE).
- It allows searching for the data and clearing the logs from the terminal.
- You can run scripts by inputting them at the bottom after the ›.



Environment: JavaScript VM
Account: 0xca3...a733c (0.000000000000476
Gas limit: 3000000
Value: 0 wei
Ballot
Deploy: uint8 _numProposals
At Address: Load contract from Address
Transactions recorded: 3
Deployed Contracts
Ballot at 0x692...77b3a (memory)
Ballot at 0xbbf...732db (memory)

```
pragma solidity >=0.4.22 <0.6.0;
contract Ballot {

    struct Voter {
        uint weight;
        bool voted;
        uint8 vote;
        address delegate;
    }
    struct Proposal {
        uint voteCount;
    }

    address chairperson;
    mapping(address => Voter) voters;
    Proposal[] proposals;

    /// Create a new ballot with $(_numProposals) different proposals.
```

0 listen on network
Search with transaction hash or address

```
[vm] from:0xca3...a733c to:Ballot.(constructor) value:0 wei data:0x608...30030 logs:0 hash:0x9a8...ca643
creation of Ballot pending...
[vm] from:0xca3...a733c to:Ballot.(constructor) value:0 wei data:0x608...30030 logs:0 hash:0x1d4...2bde3
creation of Ballot pending...
[vm] from:0xca3...a733c to:Ballot.(constructor) value:0 wei data:0x608...30030 logs:0 hash:0xf37...0541f
>
```

CHAPTER -4

IMPLEMENTATION AND USER INTERFACE

4.1 Implementation of the Bookopedia:

Implementation of eHospital using Blockchain is taken place in various phases. Firstly, I build the smart contracts then user interface i.e., detail of doctors and patients etc. Are shown by using interface. And finally, I use JavaScript to connect the smart contracts to the html (user interface) by using adding ABI in html file.

4.1.1 Step to be followed to develop the app:

Firstly, I created the smart contracts in remix compiler.

- First, I created the body examine smart contracts in which I added various testes.
- Then, I created doctor registration smart contracts in which doctors are registred by hospital.
- After other smart contracts were created of patients, patient’s details, etc.
- After creating all smart contract I created a user interface for each smart contract.

Flow Chart for the User is given below:

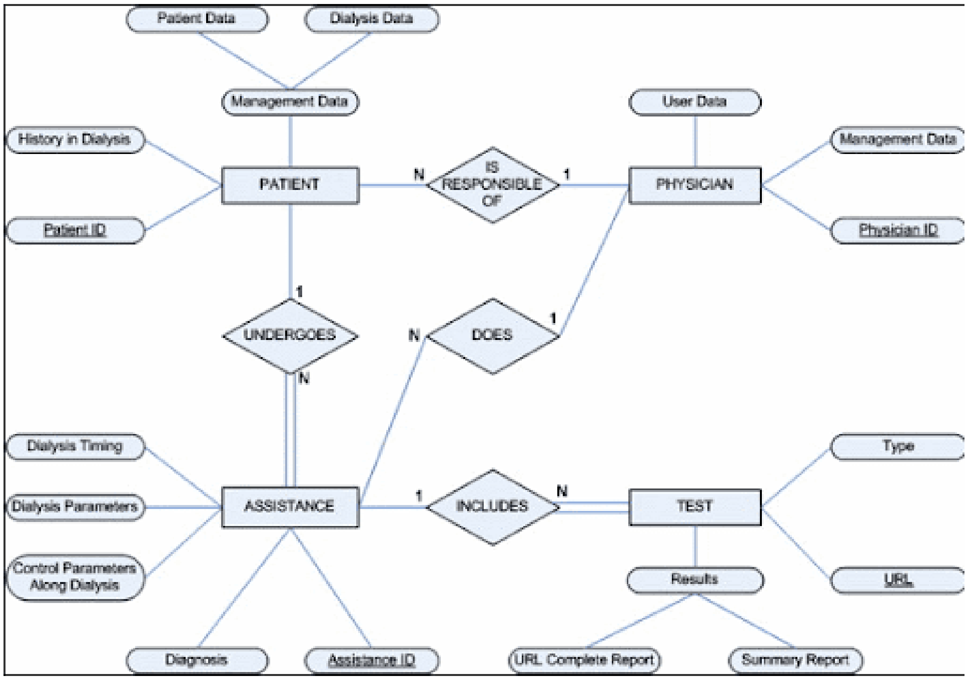


Figure-6: Flow Chart for User

CHAPTER - 5

TESTING

Once source code has been generated, software must be tested to uncover as many errors as possible before delivery. It is very important to work the system successfully and achieve high quality of software. Testing includes designing a series of test cases that have a high likelihood of finding errors by applying software-testing techniques.

System testing makes logical assumptions that if all the parts of the system are correct, the goal will be successfully achieved. The system should be checked logically. Validations and cross checks should be there. Avoid duplications of record that cause redundancy of data.

In other Words, Testing is the process of evaluating a system or its component(s) with the intent to find whether it satisfies the specified requirements or not. It is executing a system in order to identify any gaps, errors, or missing requirements in contrary to the actual requirements.

There are different types of testing some of them are listed below:

5.1 user interface Testing:

In this I tested that Hospital, doctor and patient are able to read and write the data on blockchain and patient detail are only uploaded by doctors and doctors' registration are only done by hospital.

5.2 Unit Testing

It focuses on smallest unit of system design. In this we test an individual unit or groups of inter related units. It is often done by programmer by using sample input and observing its corresponding outputs. In this testing technique we are primarily focuses on data .

Unit Testing of the system:

Test cases	**Description**	**Expected Outcome**	**Result**
1	Doctor Registration	It is only done by hospital. No other can upload the detail.	Pass

2	Patient registration	It is only done by doctor and if any other try to do it then smart contract will revoke.	Pass
3	View Patients details	Should display the details of patients	Pass

4	View medical records	Should display the medical records of the patients	Pass
5	View patient examine details	Should display all the tests and reports of the patients	Pass

Table 1: Unit Testing of eHospital using blockchain

5.3User Testing

User testing is the process through which the interface and functions of a website, app, product, or service are tested by real users who perform specific tasks in realistic conditions. The purpose of this process is to evaluate the usability of that website or app and to decide whether the product is ready to be launched for real users.

This app was tested by our team mates and friends who are using different mobile phones (and having different android version) also tested on different emulator to check its performance and it seems to be working fine and users of this system are satisfied with the facilities and performance of the it and like the way how the system is worked.

1. 5.4 Performance Testing

In this type of testing, we have checked the performances of our application under some peculiar conditions are checked. Those conditions include:

- Low memory in the device.
- The battery in extremely at a low level.
- Poor/Bad network reception.

Performance is basically tested from 2 ends, application end, and the application server end. Our app is also performing well in this phase of testing as well. And we are getting positive feedback from user of our app.

5.5 Combability Testing

This system was tested and used on different devices like Google Nexus 4, tablets, Linux, macOS. the system worked fine and is stable. On all types of testing (that we have performed above) our performing well on it.

CHAPTER -6

CONCLUSION

A country’s expenditure on healthcare as a proportion of GDP is considered as one of the important indicators for the overall population health. India’s current spending on healthcare is <4% of the total GDP, compared to the top 10 countries (of OECD), which spends on an average >10%. Electronic Health Record, which plays a crucial role in processing and analyzing patients' medical records to provide better quality care, in India, specifically, is practiced in an unstandardized manner. Healthcare providers face challenges in terms of maintaining data immutability and privacy, with respect to data sharing with relevant stakeholders. Different healthcare institutions use different models and their own set of codes, which also makes it hard to achieve interoperability. This issue is further escalated when patients change their service providers due to unforeseen reasons. Patients' are forced to redo their medical tests and treatment processes, which adds to the overhead costs and also to their frustration. This can be addressed to a major extent through the use of Blockchain technology.

REFERENCES

1. **InternShala: https://trainings.internshala.com/**

2. **Ethereum developer Guide:**

   https://ethereum.org/en/developers/

3. **For rectifying the error :**

   https://stackoverflow.com/