PROBLEM STATEMENT

Develop a Blockchain based application for health related medical records.

AIM AND OBJECTIVE

The main objective of this project is to build a web application using blockchain technology which establishes that a patient's medical record is safe, consistent and accessible across all healthcare providers as determined by the patient.

SYSTEM REQUIREMENTS

THEORY

Functional Requirements

Functional Requirement defines a function of software system and how the system must behave when presented with specific inputs or conditions. These may include calculations, data manipulation and processing and other specific functionality. The functional requirements of the project are one of the most important aspects in terms of entire mechanism of modules.

The functional requirements here are:

- **Maintaining user:** Interface responsiveness: If the application needs to perform a time-consuming task, multiple threads can be used to prevent user interface from becoming unresponsive while the task is in progress. If the program is downloading information from the Internet, this will keep theuser- interface running at nearly full-speed while the download is in progress.

- **Simple Multitasking:** Multitasking allows to execute multiple instances of a process quit easily. The downloading routine just mentioned can be extended so that the program can transfer multiple files simultaneously and still keep the user interface well behaved. All that is needed is to create another thread for each file to download.

- **Building Multi-user Applications:** Multithreading is often used when building server applications. Server applications wait for request to arrive and then establish conversations with the requester.

- **Multiprocessing:** Many operating systems support machines with multiple processors. Most of these systems are unable to break a single thread of execution for execution on different processors. By breaking an application into different Threads, it is possible to make the best use of processing power.

Non-functional requirement

- **Reliability**
  The framework ought to be dependable and solid in giving the functionalities. When a client has rolled out a few improvements, the progressions must be made unmistakable by the framework. The progressions made by the Programmer ought to be unmistakable both to the Project pioneer and in addition the Test designer.

- **Security**
  Aside from bug following the framework must give important security and must secure the entire procedure from smashing. As innovation started to develop in quick rate the security turned into the significant concern of an association. A great many dollars are put resources into giving security. Bug following conveys the greatest security accessible at the most noteworthy execution rate conceivable, guaranteeing that unapproved clients can't get to imperative issue data without consent. Bug following framework issues diverse validated clients their mystery passwords so there are limited functionalities for all the clients.

- **Maintainability**
  The framework observing and upkeep ought to be basic and target in its approach. There should not be an excess of occupations running on diverse machines such that it gets hard to screen whether the employments are running without lapses.

- **Performance**
  The framework will be utilized by numerous representatives all the while. Since the framework will be facilitated on a solitary web server with a solitary database server out of sight, execution turns into a noteworthy concern. The framework ought not succumb when numerous clients would be utilizing it all the while. It ought to permit quick availability to every last bit of its clients. For instance, if two test specialists are all the while attempting to report the vicinity of a bug, then there ought not to be any irregularity at the same time.

HARDWARE REQUIREMENTS

- Processor type : Intel core i3 and above
- Processor speed : Minimum 2.00 GHz and above
- RAM : 4 GB and above
- HARD DISK : 400 GB or more
- Monitor : 800x600 or higher resolution

SOFTWARE REQUIREMENTS

- Operating System : Windows 7 (32 bit and 64 bit) and Above
- Development Environment : Python Programming, Web Development(CSS, HTML,JAVA SCRIPT)
- Scripting Language : Python, Html, CSS
- Browser : Google Chrome, Microsoft Edge
- Software : Visual Studio or Similar IDE

THEORY

Most of us remember going to the doctors and seeing them pull out a dreary-looking folder with our name written on the front. If you had found yourself going to the doctors quite frequently, your folder will probably have been loaded with different pieces of paper making it look scarily full. It is very important to maintain all the record in secure manner and also confidentiality is very important. Because the medial reports are more sensitive.

Blockchain is a new emerging technology of distributed databases, which guarantees the integrity, security and incorruptibility of data by means of the cryptography. Such features are suitable for secure and reliable data storage.

Moreover, the blockchain technique accelerates the medical record or information exchange such that the cost of human resource is significant reduced. All patients can manage their individual medical records and information easily in the different hospitals and clinics. They also have the privilege to deal with and authorize personal medical records in the proposed management framework. Blockchain has a widerange of applications and uses in healthcare. Distributed ledger technology facilitates the secure transfer of patient medical records, manages the medicine supply chain and helps healthcare researchers unlock genetic code.

Need for EMR:

Since blockchain technology relies on a distributed network there is no one point of failure. This means that it is not possible for hackers to simply find an electronic health record security flaw and then gain access to the data in this way. Should a hacker target any one node on the blockchain network and attempt to make an unauthorized change then the other nodes will prevent it from happening. Blockchain technology would prevent the majority of the 477separate breaches that took place in the last years from ever happening. Any attack that exploited the single point of failure weakness of the traditional client-server model has would be useless against a blockchain database.

As each participant on the network has a complete copy of the entire blockchain ledger, they are able to independently verify any new block being added and identify any attempt to alter any previous block. As every new data block that is added would also contain details of the doctor who added it, blockchain EMRs would offer full accountability for the data that they contain. In the case of incorrect diagnoses, for example, patients could be confident that there is no way that records could be altered should the doctor or healthcare provider wish to deny accountability.

UNIVERSAL ACCESS:

The fact that it is possible to encrypt data stored on a blockchain means that it is possible to keep sensitive patient data on these distributed networks without the risk of unauthorized access.

Creating a non-centralized EMR blockchain would mean that it is possible for any authorized healthcare provider to access patient information.

Since no one healthcare provider would be liable for any data breach when using such a database, there need not be any resistance to making records universally available. As long as a provider has software that is compatible with the data to the blockchain then they will be able to access it.

All that would be required is for a patient to authorize access to their protected health information, possible through a secure blockchain-based ID verification solution like Sovrin. After this, the doctor would have full access to their records and would, therefore, be able to treat the patient.

Aside from allowing lifesaving care by emergency services to be improved, such universal access to personal health information also promises to save healthcare providers huge sums of money from lower administration costs relating to the upkeep and transfer of records as well as from reduced liability from data breaches, etc.

Keeping our important medical data safe and secure is the most popular blockchain healthcare application at the moment, which isn't surprising. Security is a major issue in the healthcare industry. There were 692 large healthcare data breaches reported between July 2021 and June2022. The perpetrators stole credit card and banking information, as well as health and genomic testing records.

Blockchain's ability to keep an incorruptible, decentralized and transparent log of all patient data makes it a technology rife for security applications. Additionally, while blockchain is transparent, it is also private, concealing the identity of any individual with complex and secure codes that can protect the sensitivity of medical data. The decentralized nature of thetechnology also allows patients, doctors and healthcare providers to share the same information quickly and safely.

CODE

app.py

```
from flask import Flask,render_template,request,url_for,session,redirect,jsonify
import json,sqlite3
from datetime
import date
import datetime
from time import time
from hashlib import sha256
import datetime,time,pymongo
from passlib.context import CryptContext
import requests, ipfshttpclient
import os,webbrowser
from werkzeug.utils import secure_filename

states=["Andhra Pradesh", "Arunachal
Pradesh","Assam","Bihar","Chhattisgarh","Goa","Gujarat","Haryana","Himachal
Pradesh", "Jammu and
Kashmir","Jharkhand","Karnataka","Kerala","MadhyaPradesh","Maharashtra","
Manipur","Meghalaya","Mizoram","Nagaland","Odisha","Punjab","Rajasthan","
Sikkim","Tamil
Nadu","Telangana","Tripura","UttarPradesh","Uttarakhand","WestBengal","And
aman and Nicobar Islands", "Chandigarh", "Dadra and Nagar Haveli", "Daman
and Diu", "Lakshadweep", "NCT", "Puducherry"]

#session['user']='Genesis'

'''Blockchain=[{

        'index':'0',zzz
        'patientid':'00000',
        'first': '','
        last': '',
        'doctor id': '',
        'Dor': '12-13-2019',
        'Age': '',
        'haemo':'',
        'blood':'',
    }]’’’

login_status=0
app= Flask(_name_)
app.secret_key = 'PATREC Authentication'
#Password encryption scheme
pwd_context = CryptContext(
    schemes=["pbkdf2_sha256"],
    default="pbkdf2_sha256",
    pbkdf2_sha256_default_rounds=30000
)
```

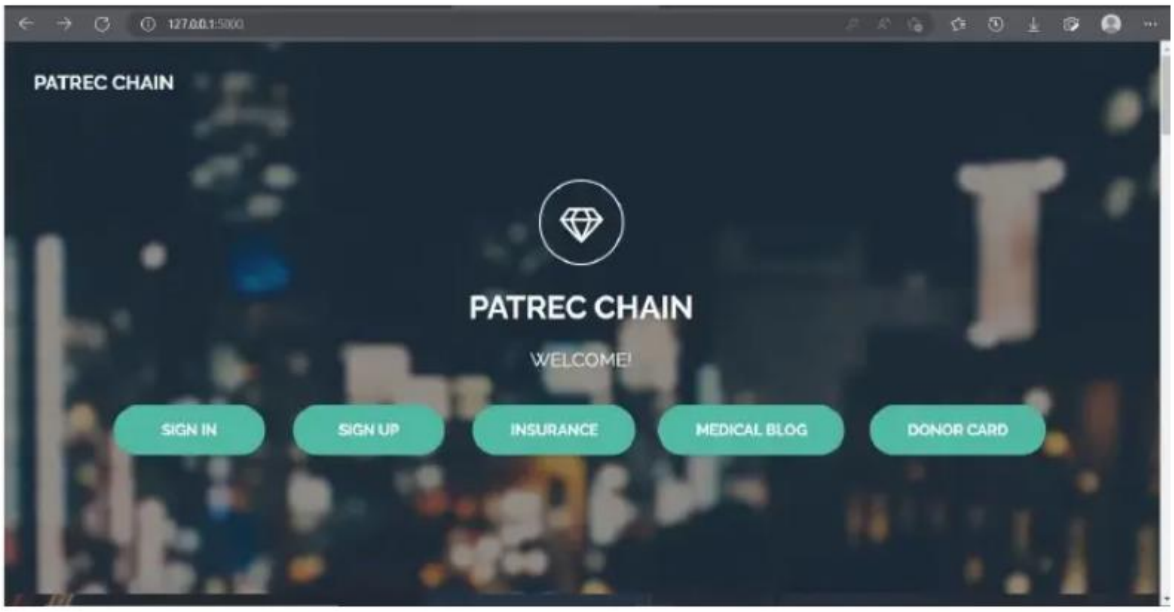
```
#Session variables
'''class sessionlog:
        def init (self):
             self.username=''
             self.id=''
sess=sessionlog()'''



def encrypt_password(password):
      return pwd_context.hash(password)
def check_encrypted_password(password, hashed):
      return pwd_context.verify(password, hashed)
'''hashset=[]
ind=-1
patdoc= mycol.find()
for x in patdoc:
      hashset.append(x['hash'])
      ind=ind+1
class index:
      index=ind
      def getindex(self):
            self.index=int(self.index+1)
            return self.index
idv=index()''
'@app.route('/')
@app.route('/home')
def home():
       return render_template('index.html')
@app.route('/learnmore')
def learnmore():
       return render_template('generic.html')'''
def oldhome():
       return render_template('welcome.html')'''
@app.route('/addguard')def addguard():
       return render_template('addguard.html')
@app.route('/addguardian',methods=['post'])
defaddguardian():
        con=mydb['Guardian_contract']
         contract={
               'guardian':request.form['guardian'],
               'owner':request.form['owner']
}
@app.route('/linkedacc')
deflinkedacc():
acc=session['user']
con=mydb['Guardian_contract']
contract={'guardian':acc}
myval=con.find(contract)
mycontro=[]
for i in myval:
      mycontro.append(i)
contract={'owner':acc}
myval=con.find(contract)
outcontro=[]
for i in myval:
      outcontro.append(i)
    return render_template('linkedacc.html',mycontrol=mycontro,outcontrol=outcontro)
'''
#MEDREC form creation
@app.route('/create', methods=['POST'])
def createblock():
pid=request.form['pid']
doc= request.form['doc']
blood= request.form['blood']
serum= request.form['serum']
thdl=request.form['thdl']
ldl=request.form['ldl']
rbc=request.form['rbc']
pulse=request.form['pulse']
patdoc= myrow.find()
ind=-1
for x in patdoc:
      prev=x['hash']
     ind=ind+1
ts=time.time()
st = datetime.datetime.fromtimestamp(ts).strftime('%Y-%m-%d %H:%M:%S') block={
'_id':pid+'REC'+str(ind+1),
'owner':pid,
'doc':doc,
'gluc':pp,
'glucf':fast,
'serum':serum,
'blood':blood,
'chol': tot,
'thdl':thdl,
'ldl':ldl,
'rbc':rbc,
'pulse':pulse,
'prev': prev,
'timestamp':st
}
block_string = json.dumps(block, sort_keys=True)hashval=sha256(block_string.encode()).hexdigest() block={
'_id': pid+'REC'+str(ind+1),
'owner':pid,
'doc':doc,
'gluc':pp,
'glucf':fast,
'serum':serum,
'blood':blood,
'chol': tot,
'thdl':thdl,
'ldl':ldl,
'rbc':rbc,
'pulse':pulse,
'prev': prev,
'hash':hashval,
'timestamp':st
}
myrow.insert_one(block)
return render_template('newrec.html',post=block)
```

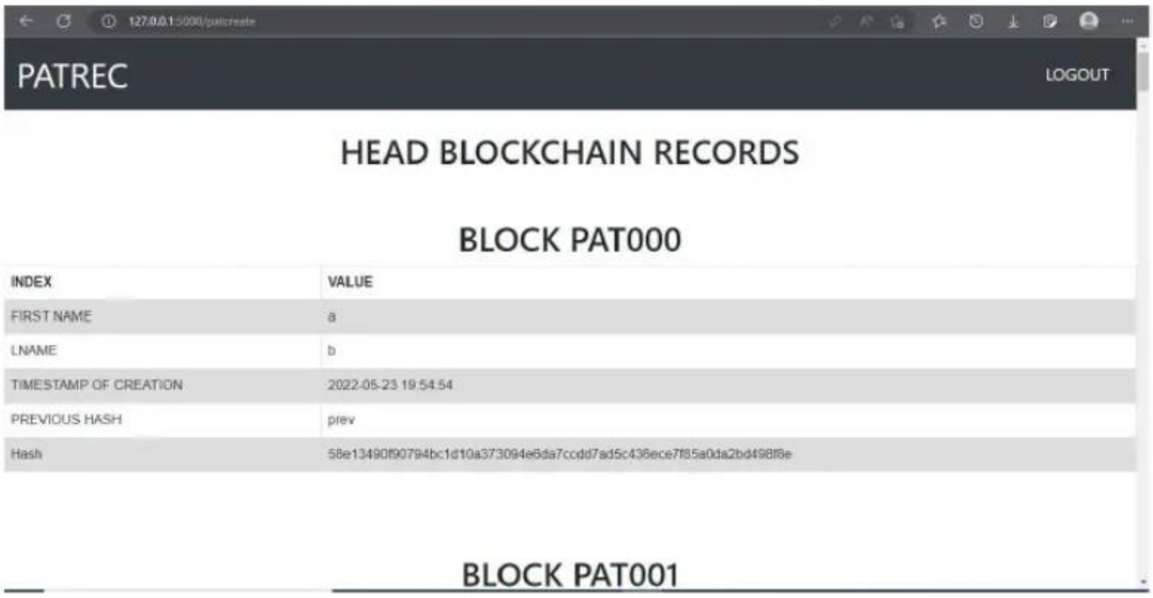
Main.css

```
@charset "UTF-8";

@import url(font-awesome.min.css);

@import url("https://fonts.googleapis.com/css?family=Raleway:400,500,700");

html, body, div, span, applet, object, iframe, h1, h2, h3, h4, h5, h6, p, blockquote, pre,a, abbr, acronym,
address, big, cite, code, del, dfn, em, img, ins, kbd, q, s, samp, small,strike, strong, sub, sup, tt, var, b, u, i,
center, dl, dt, dd, ol, ul, li, fieldset, form, label,legend, table, caption, tbody, tfoot, thead, tr, th, td,
article, aside, canvas, details, embed,figure, figcaption, footer, header, hgroup, menu, nav, output, ruby,
section, summary,time, mark, audio, video {

        margin: 0;
        padding: 0;
        border: 0;
        font-size: 100%;
        font: inherit;
        vertical-align: baseline;
}


article, aside, details, figcaption, figure, footer, header, hgroup, menu, nav, section{ display: block;}
body {line-height: 1;}
ol, ul {list-style: none;}
blockquote, q {quotes: none;}
blockquote:before,
blockquote:after,
q:before, q:after{ content: ''; content: none }
table { border-collapse:collapse; border-spacing:0;}
body { -webkit-text-size-adjust: none; }
/* Box Model */*,
*:before, *:after {
-moz-box-sizing: border-box;
-webkit-box-sizing: border-box;
box-sizing: border-box;}
.container {margin-left: auto;margin-right: auto;}
```

```
.container.\31 25\25 {width: 100%;max-width: 100em;min-width: 80em;}
.container.\37 5\25 {width: 60em;}
.container.\35 0\25 {width: 40em;}
.container.\32 5\25 {width: 20em;}
.container {width: 80em;}
```

OUTPUT



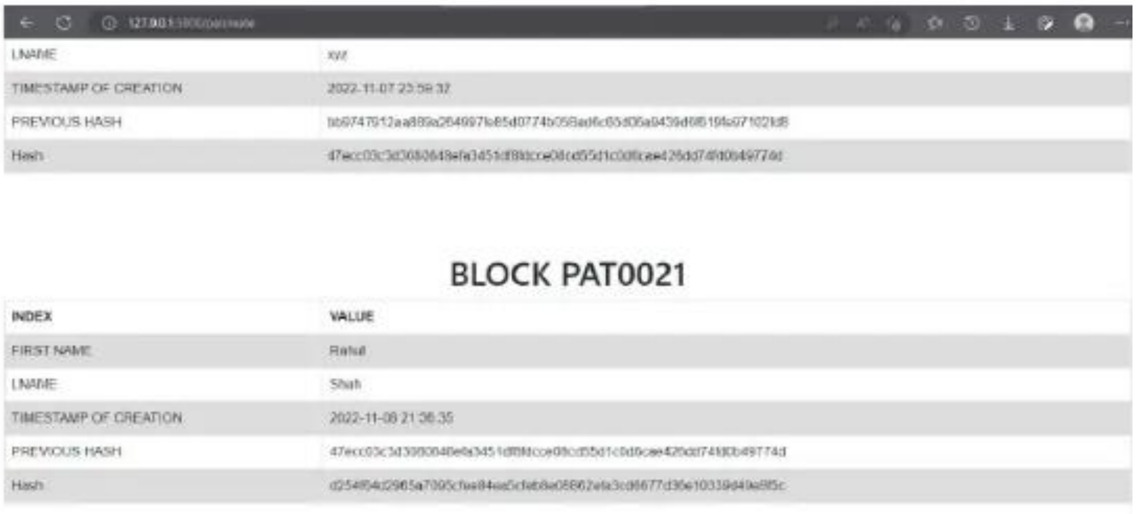


HEAD BLOCKCHAIN RECORDS

BLOCK PAT000

INDEX	VALUE
FIRST NAME	a
LNAME	b
TIMESTAMP OF CREATION	2022-05-23 19:54:54
PREVIOUS HASH	prev
Hash	58e13490f90794bc1d10a373094e6da7ccdd7ad5c436ece7f85a0da2bd498f8e

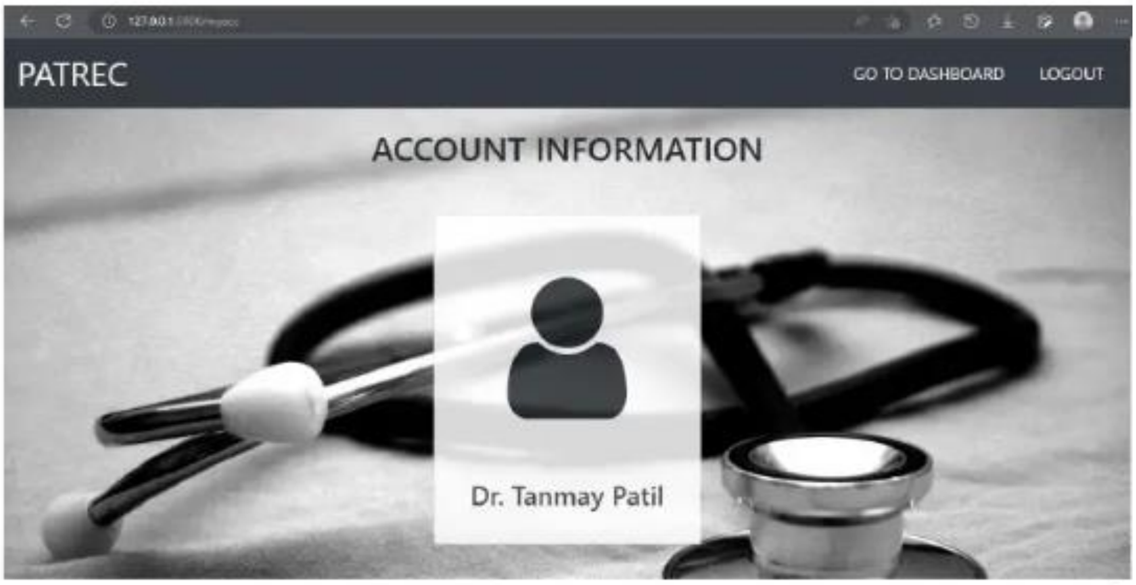
BLOCK PAT001



LNAME	xyz
TIMESTAMP OF CREATION	2022-11-07 23:59:37
PREVIOUS HASH	bb6747612aa889a264997b85d0774b058ad6c65d06a0439d6619fa071021d8
Hash	47ecc03c3d3080648efa3451df8dcce08cd55d1c0d6cae426dd74fd0b49774d

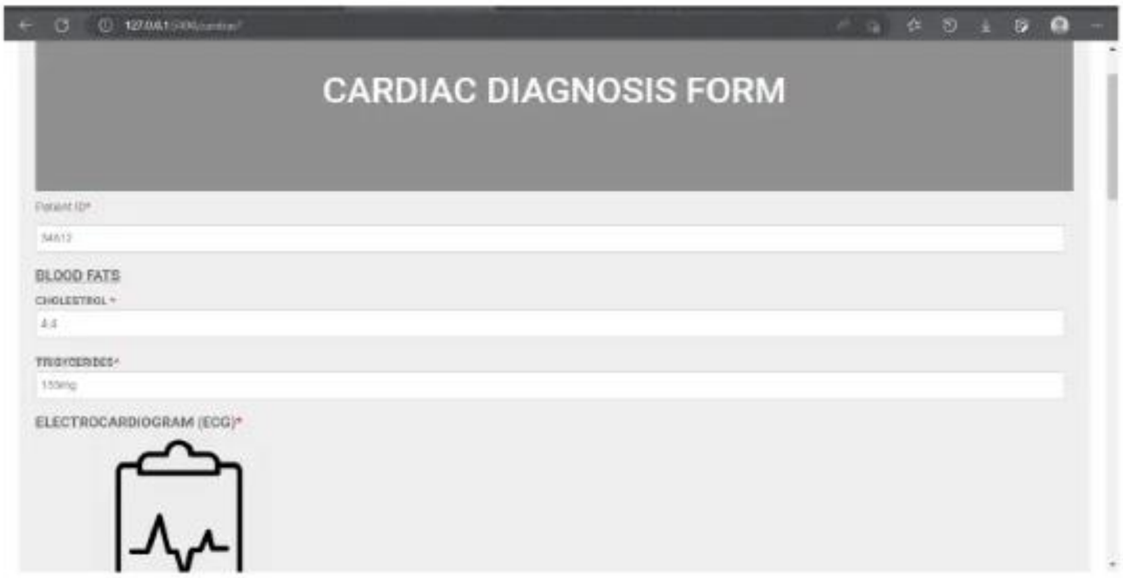
BLOCK PAT0021

INDEX	VALUE
FIRST NAME	Rahul
LNAME	Shah
TIMESTAMP OF CREATION	2022-11-08 21:36:35
PREVIOUS HASH	47ecc03c3d3080648efa3451df8dcce08cd55d1c0d6cae426dd74fd0b49774d
Hash	d254f64d2965a7095cfae84ea5cfeb8e05862efa3cd6677d36e10339d49e85c



ACCOUNT INFORMATION

Dr. Tanmay Patil



CARDIAC DIAGNOSIS FORM

Patient ID*

54612

BLOOD FATS

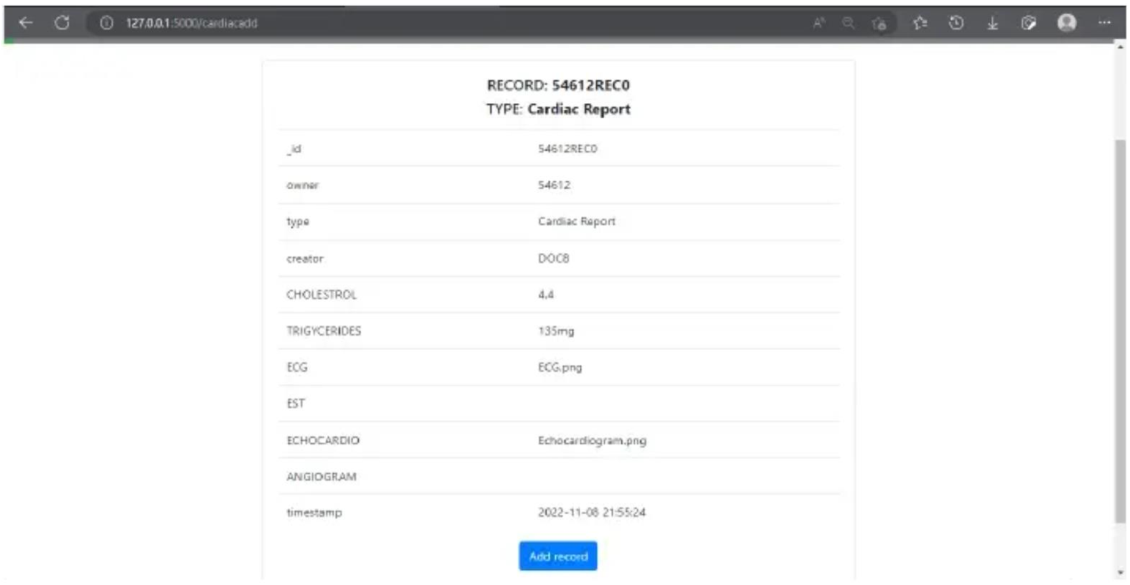
CHOLESTROL *

4.4

TRIGYCERIDES*

150mg

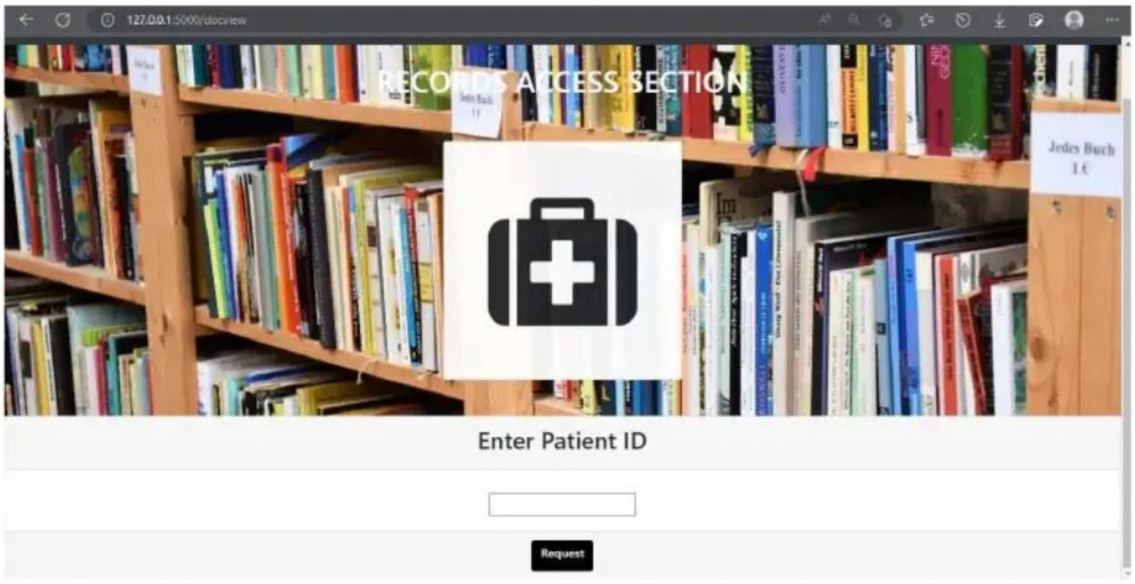
ELECTROCARDIOGRAM (ECG)*



127.0.0.1:5000/cardiacadd

RECORD: **54612REC0**
TYPE: **Cardiac Report**

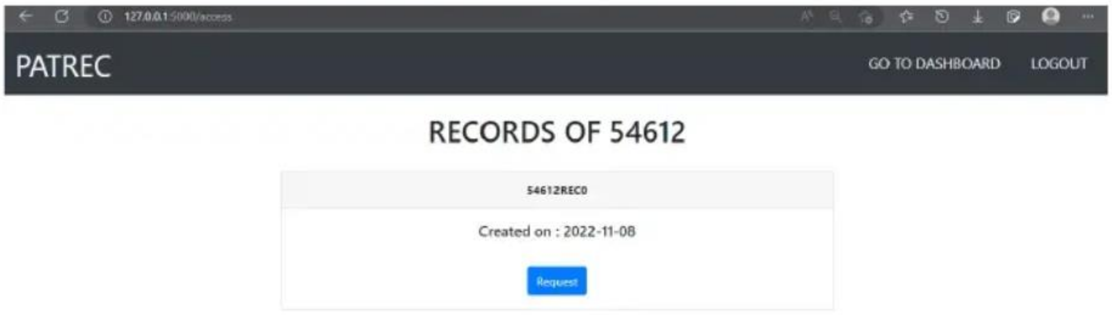
_id	54612REC0
owner	54612
type	Cardiac Report
creator	DOC8
CHOLESTROL	4.4
TRIGYCERIDES	135mg
ECG	ECG.png
EST	
ECHOCARDIO	Echocardiogram.png
ANGIOGRAM	
timestamp	2022-11-08 21:55:24



127.0.0.1:5000/docview

RECORDS ACCESS SECTION

Enter Patient ID

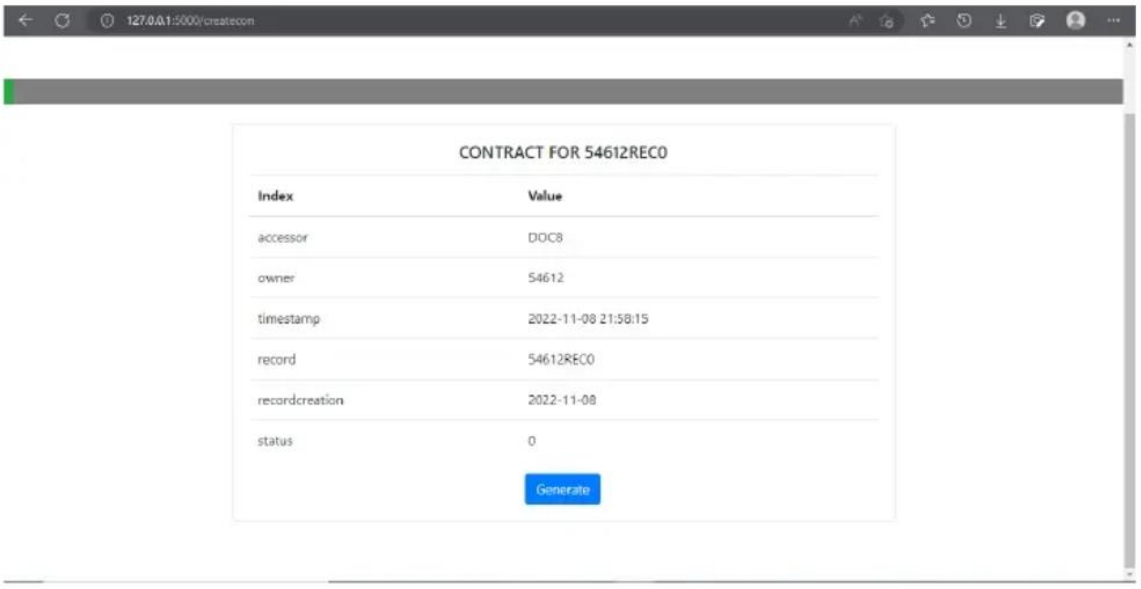


127.0.0.1:5000/access

RECORDS OF 54612

54612REC0

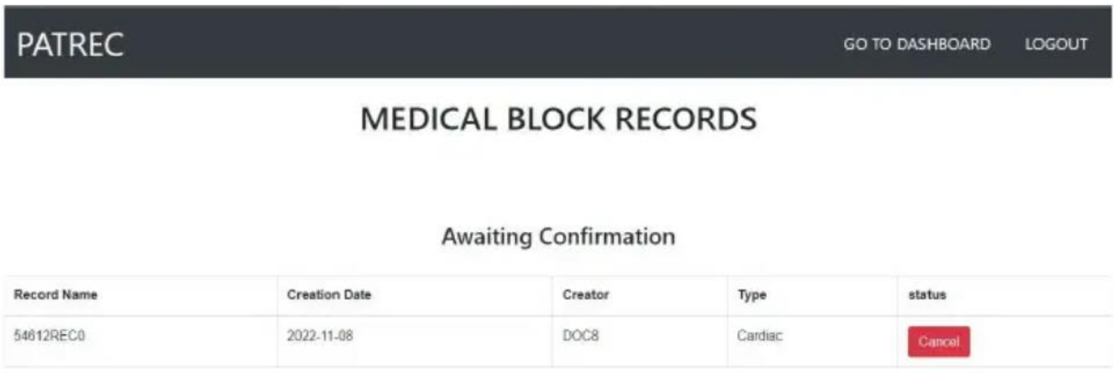
Created on : 2022-11-08



127.0.0.1:5000/createcon

CONTRACT FOR 54612REC0

Index	Value
accessor	DOC8
owner	54612
timestamp	2022-11-08 21:58:15
record	54612REC0
recordcreation	2022-11-08
status	0



PATREC

MEDICAL BLOCK RECORDS

Awaiting Confirmation

Record Name	Creation Date	Creator	Type	status
54612REC0	2022-11-08	DOC8	Cardiac	Cancel

CONCLUSION

Applying blockchain technology to the secure electronic health records industry is set to bring about some exciting new changes to the availability and use of our medical records.
While it remains in the early days, blockchain will overcome the difficulties of technical safeguards that caused such disappointment in past attempts to create electronic medical records. The main challenge that blockchain is sure to face comes from storing such large amounts of data on the chain. Blockchain bloat is a fundamental problem that pertains to all blockchains. The amount of medical data required for the 100‘s of millions of medical records currently stored by healthcare providers worldwide will take a vast network of nodes to maintain and process. With the leading smart contract platform, Ethereum‘s, network already running at 100%capacity, blockchain can truly store tamper proof medical records