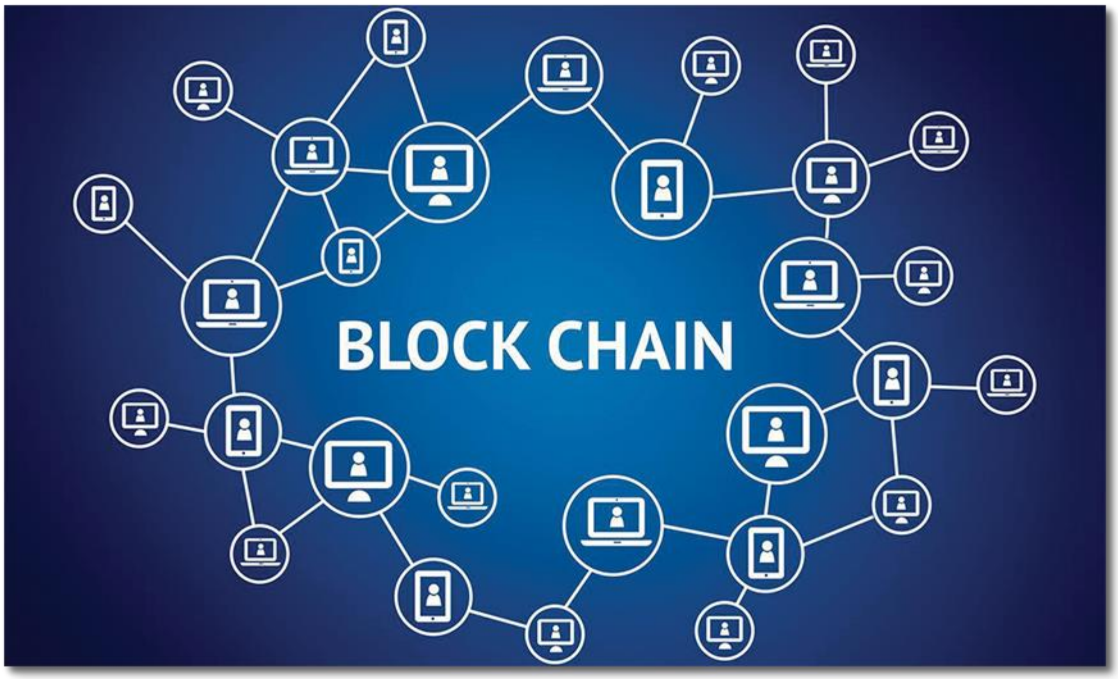
CHAPTER – 1

INTRODUCTION

Blockchain is a shared, immutable ledger that facilitates the process of recording transactions and tracking assets in a business network. Blockchains are tamper evident and tamper resistant digital ledgers implemented in a distributed fashion (i.e., without a central repository) and usually without a central authority (i.e., a bank, company or government). Blockchains are distributed digital ledgers of cryptographically signed transactions that are grouped into blocks. Each block is cryptographically linked to the previous one (making it tamper evident) after validation and undergoing a consensus decision. It provides immediate, shared and completely transparent information stored on an immutable ledger that can be accessed only by permissioned network members. Record-keeping systems can be vulnerable to fraud and cyberattacks. The whole point of using a blockchain is to let people — in particular, people who don't trust one another — share valuable data in a secure, tamperproof way. Blockchain can be used to securely store the medical records over the distributed network and these medical records can be accessed by network members to whom we have granted access.



Google Speech-Recognition API accurately converts speech into text with an API powered by the best of Google’s AI research and technology. Speech-to-Text has three main methods to perform speech recognition. These are listed below:

- **Synchronous Recognition (REST and gRPC)** sends audio data to the Speech-to-Text API, performs recognition on that data, and returns results after all audio has been processed. Synchronous recognition requests are limited to audio data of 1 minute or less in duration.
- **Asynchronous Recognition (REST and gRPC)** sends audio data to the Speech-to-Text API and initiates a Long Running Operation. Using this operation, you can periodically poll for recognition results. Use asynchronous requests for audio data of any duration up to 480 minutes.
- **Streaming Recognition (gRPC only)** performs recognition on audio data provided within a gRPC bi-directional stream. Streaming requests are designed for real-time recognition purposes, such as capturing live audio from a microphone. Streaming recognition provides interim results while audio is being captured, allowing result to appear, for example, while a user is still speaking.

In this project, Synchronous Recognition is used.

1.1 PROJECT PROBLEM DEFINITION

Medical data are sent to third party organization for medical transcription. Since data are passed through third party, there is a possibility of data theft. Chances of causing errors in the medical record is high. It is also expensive and time-consuming task.

1.2 PROJECT OBJECTIVE

To develop a web application to automate medical transcription through Google Speech Recognition API and to securely store the medical transcription record using Blockchain.

1.3 PROJECT OVERVIEW

The major modules present in this project are

- Medical Transcription process
- Storing the medical record
- Accessing the medical record

Medical Transcription process

As soon as the web application is loaded, the user has to upload an audio file and click Transcribe button. When Transcribe button is clicked, the audio file is converted to text and stored in a text file using Google Speech-Recognition API.

Storing the medical record

After the medical transcription process, the medical record is securely stored using Blockchain across the distributed network.

Accessing the medical record

After the medical record is stored in the Blockchain, a password is generated and it is mailed to the patient. Patient can view his/her medical record by entering the password.

1.4 ABOUT ORGANIZATION

Organization Profile: Uniq Technologies

Head Office: Bharathinagar 1st, North Usman Road, T.Nagar, Chennai

Branch Office Locations: Coimbatore, Tirunelveli, Bangalore, Tirupati

E-Mail Address: info@uniqtechnologies.co.in

About: Uniq Technologies is a software services company focusing on Consulting, Enterprise Solutions, Internet Applications, IT Services, System Software, Networking and Telecom and Software Testing, Verification and Validation. At Uniq, we combine business and technical knowledge based on the requirements of the client and ensure maximum Customer Satisfaction.

Specialization: Network Solutions, Software Solutions, Testing Services, Web Designing, Domain Registration, IT Services, Civil Services, Engineering Services, Mechanical Services.

CHAPTER – 2

SYSTEM SPECIFICATION

2.1 HARDWARE SPECIFICATION

Processor	- Intel® Core™ i5-2450M CPU@2.50GHz
Installed Memory (RAM)	- 4.00GB
System Type	- 64-bit OS, x64-based processor
Storage	- 500GB HDD

2.2 SOFTWARE SPECIFICATION

Operating System	- Microsoft Windows 10 Pro
Editor	- Microsoft Visual Studio Code

2.3 TOOLS, FRAMEWORKS, LIBRARIES

- **Python**

  Python is an interpreted, high-level, general-purpose programming language. Python is dynamically-typed and garbage-collected. It supports multiple programming paradigms, including procedural, object-oriented and functional programming. It supports variety of frameworks and libraries to work with data. And it also the programming language for the Blockchain along with Solidity. It will serve as backend along with Flask.

- **Flask**

  Flask is a micro web framework written in Python. It is classified as a microframework because it does not require particular tools or libraries. It has no database abstraction layer, form validation, or any other components where pre-existing third-party libraries provide common functions. Using Flask, the web application is deployed to the internet.

- **Speech Recognition API**

  Speech Recognition API is a library for performing speech recognition, with support for several engines and APIs, online and offline. Recognizer() class is initialized in order to recognize the speech. In this project, Google Speech Recognition and Synchronous Recognition method is used.

- **SQLite3**

  SQLite is a C-language library that implements a small, fast, self-contained, high-reliability, full-featured, SQL database engine. As such, it belongs to the family of embedded database. It is the most widely deployed database engine, as it is used by several of the top web browsers, operating systems, mobile phones, and other embedded systems.

- **HTML5**

  Hypertext Markup Language (HTML) is the standard markup language for documents designed to be displayed in a web browser. Web browsers receive HTML documents from a web server and render the documents into multimedia web pages. It will serve as frontend along with CSS and Bootstrap.

- **CSS3**

  Cascading Style Sheets (CSS) is a style sheet language used for describing the presentation of a document written in a markup language like HTML. CSS is designed to enable the separation of presentation and content, including layout, colors, and fonts.

- **Bootstrap**

  Bootstrap is a free and open-source CSS framework directed at responsive, mobile-first front-end web development. It contains CSS and (optionally) JavaScript-based design templates interface components.

CHAPTER – 3

SYSTEM STUDY

3.1 EXISTING SYSTEM WITH LIMITATIONS

Medical transcription (MT) is the manual processing of voice reports dictated by physicians and other healthcare professionals into text format. Medical data are sent to third party organization for medical transcription. Since data are passed through third party, there is a possibility of breach in medical records. Humans make errors, mistakenly misunderstand some words and it may affect the quality of the medical report. Medical transcription companies are paid extra to complete the process as soon as possible.

Limitations

- Voice reports that are sent to third party organization for medical transcription process has high risk of confidentiality breach.
- Usually, it takes hours of typing to complete one medical record.
- This is an expensive process and a time-consuming task.

3.2 PROPOSED SYSTEM WITH ADVANTAGES

To reduce time and money, the medical transcription process can be automated using Google Speech Recognition API, which converts voice reports into text format. The goal of blockchain is to allow digital information to be recorded and distributed, but not edited. In this way, a blockchain is the foundation for immutable ledgers, or records of transactions that cannot be altered, deleted, or destroyed. The medical record is stored across the distributed network as blocks and combined together. A credential ID is generated automatically and mailed to the patient and using that ID the patient can view their medical report.

This solution is deployed as web application using Flask web micro framework. This process is completed within 5 minutes thus saving enormous amount of time and money.

Advantages

- ✓ Since Speech Recognition API is used to automate the medical transcription, time and money can be reduced.
- ✓ Medical records are securely stored in Blockchain thus making it impossible for any confidentiality breach.
- ✓ It will be deployed as a web application, so it makes it easy for the doctor and patient to access the medical record whenever required.

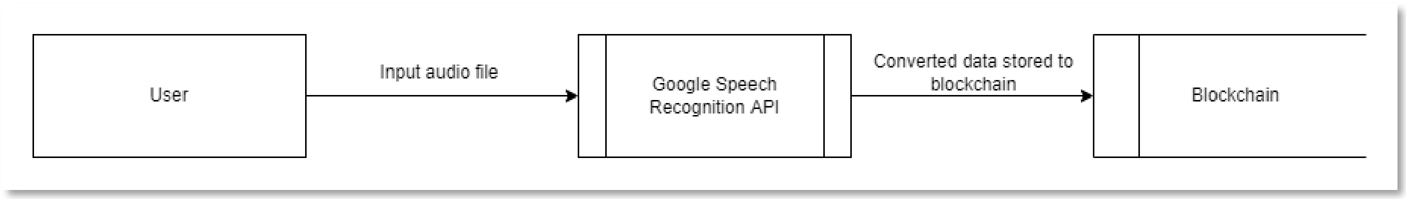
CHAPTER – 4

SYSTEM DESIGN

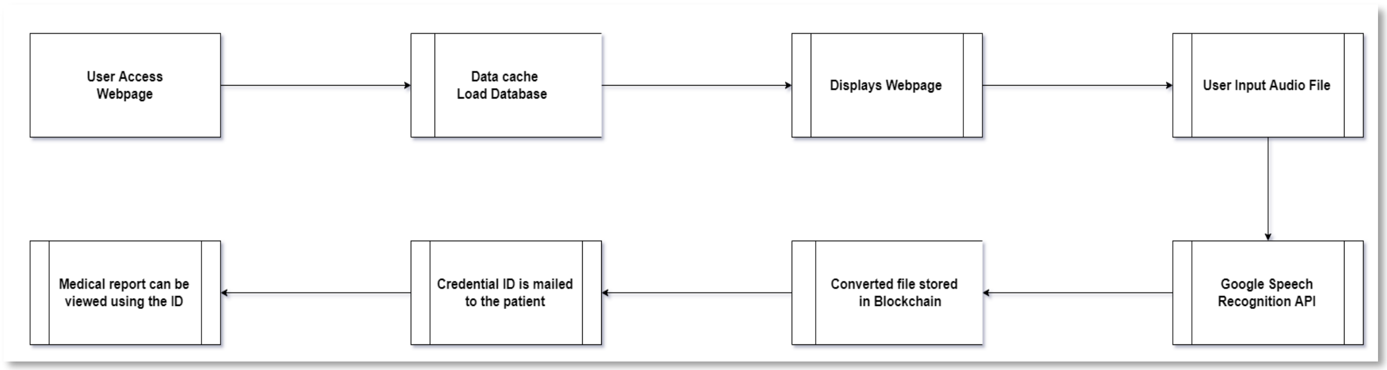
4.1 DATAFLOW DIAGRAM

A data-flow diagram (DFD) is a way of representing a flow of a data of a process or a system (usually an information system). The DFD also provides information about the outputs and inputs of each entity and the process itself.

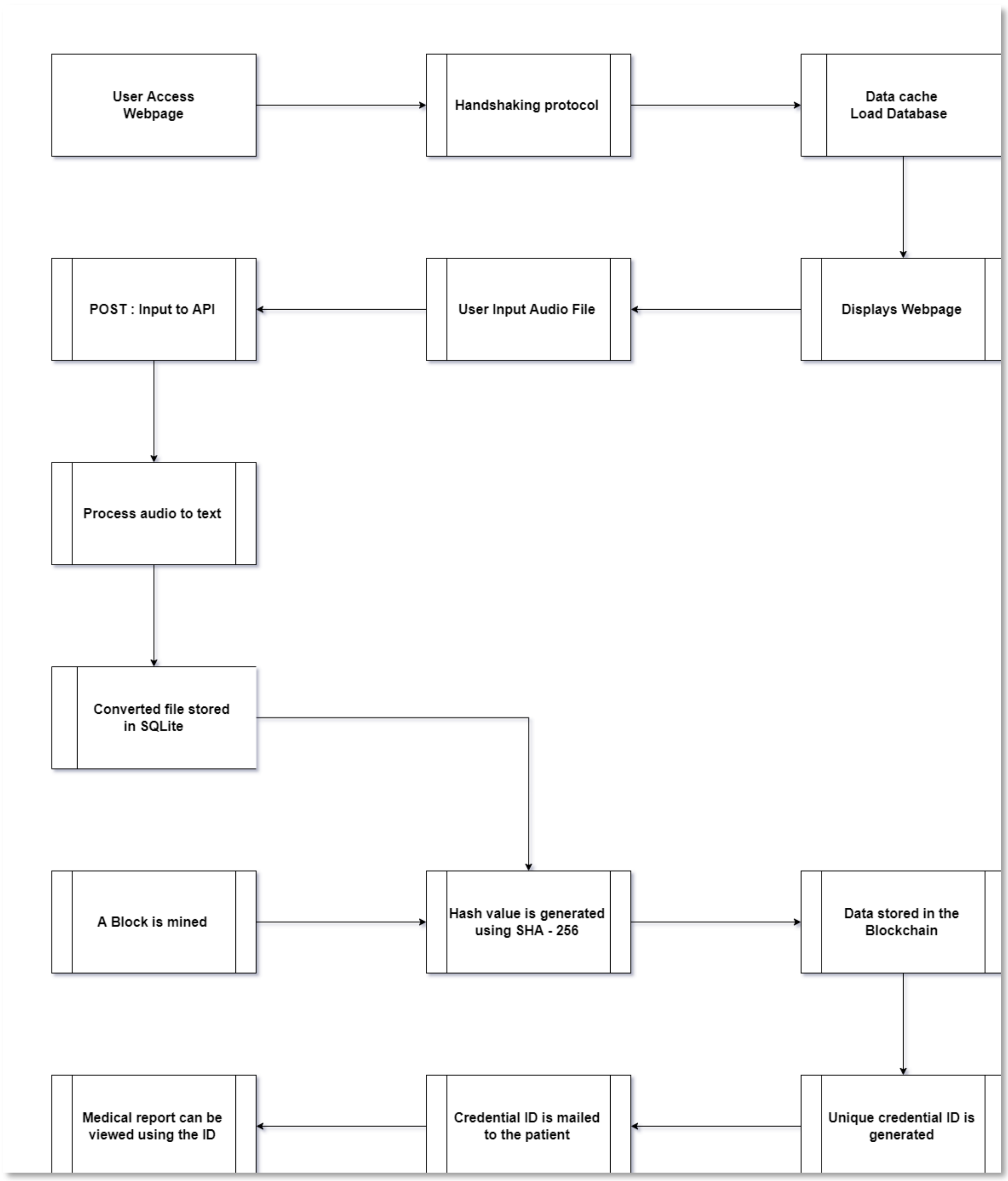
Level 0 DFD


User
Input audio file
Google Speech Recognition API
Converted data stored to blockchain
Blockchain

Level 1 DFD

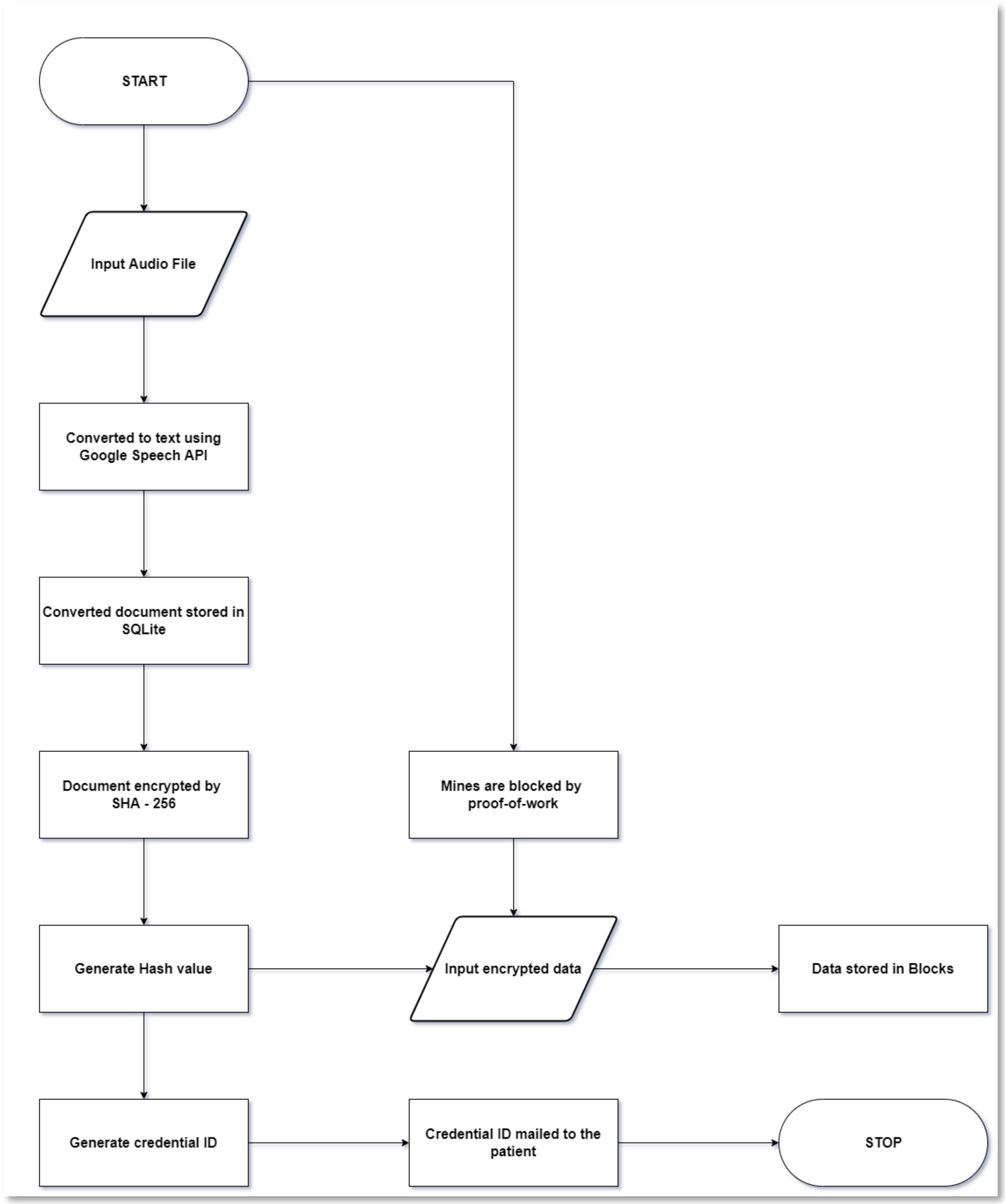

User Access Webpage
Data cache Load Database
Displays Webpage
User Input Audio File
Google Speech Recognition API
Converted file stored in Blockchain
Credential ID is mailed to the patient
Medical report can be viewed using the ID

Level 2 DFD


User Access Webpage
Handshaking protocol
Data cache Load Database
Displays Webpage
User Input Audio File
POST : Input to API
Process audio to text
Converted file stored in SQLite
Hash value is generated using SHA - 256
A Block is mined
Data stored in the Blockchain
Unique credential ID is generated
Credential ID is mailed to the patient
Medical report can be viewed using the ID

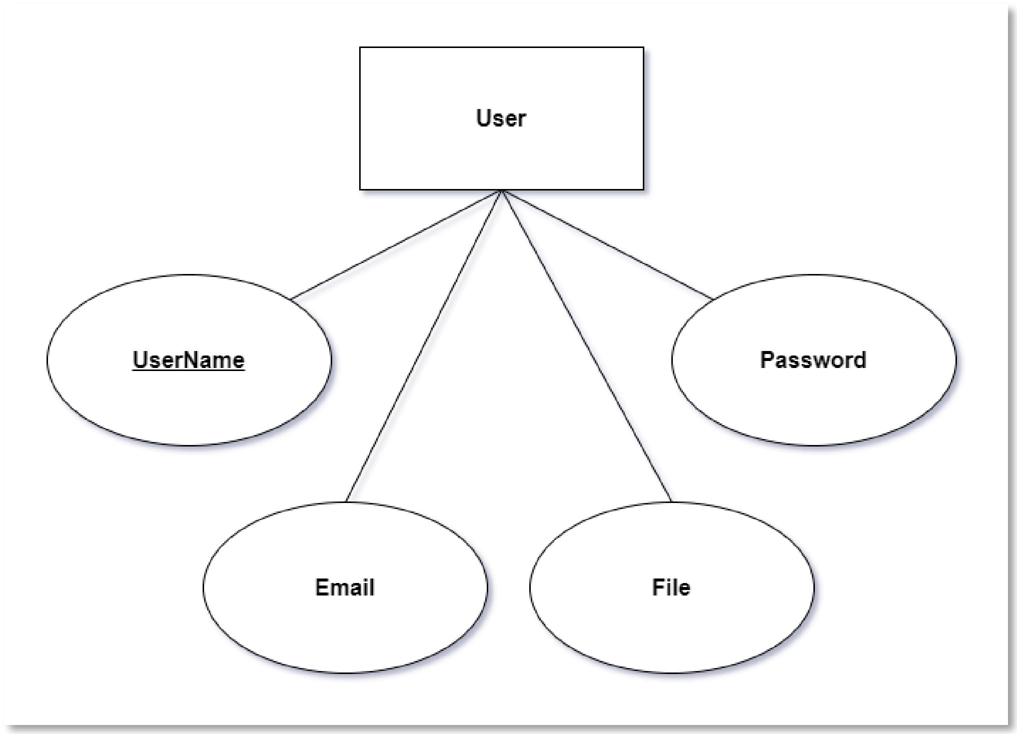
SYSTEM FLOW DIAGRAM

A system flowchart is a visual representation of processes, decisions, inputs and outputs that together form a system. System flowcharts are a way of displaying how data flows in a system and how decisions are made to control events.


START
Input Audio File
Converted to text using Google Speech API
Converted document stored in SQLite
Document encrypted by SHA - 256
Generate Hash value
Generate credential ID
Mines are blocked by proof-of-work
Input encrypted data
Data stored in Blocks
Credential ID mailed to the patient
STOP

ENTITY RELATIONSHIP DIAGRAM

ER Model stands for Entity Relationship Model is a high-level conceptual data model diagram. ER model helps to systematically analyze data requirements to produce a well-designed database. The ER Model represents real-world entities and the relationships between them.



4.2 INPUT DESIGN

Input Design is the process of converting a user-oriented description of the input into a computer-based system. Input is the raw data that is processed to produce output.

Objective of Input Design

The objectives of input design are

- To design data entry and input procedures
- To reduce input volume
- To design source documents for data capture or devise other data capture methods
- To design input data records, data entry screens, user interface screens, etc.
- To use validation checks and develop effective input controls.

Some of the popular data input methods are

- Batch input method (Offline data input method)
- Online data input method
- Computer readable forms
- Interactive data input

In this project, audio file upload form and access medial report form are the primary input modules. Both are designed using CSS and Bootstrap with minimal design and functionality.

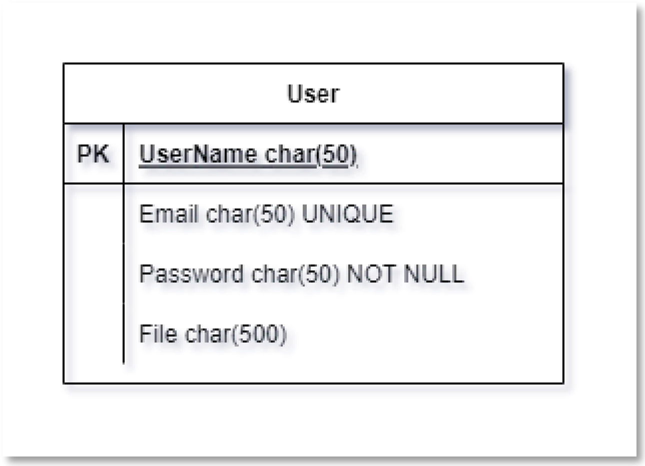
4.3 DATABASE DESIGN

Database design is the organization of data according to a database model. The designer determines what data must be stored and how the data elements interrelate. It helps produce database systems.

- That meet the requirements of the users
- Have high performance

The main objectives of database designing are to produce logical and physical designs models of the proposed database system. The logical model concentrates on the data requirements and the data to be stored independent of physical considerations. It does not concern itself with how the data will be stored or where it will be stored physically.

In this project, SQLite3 RDBMS is used. A database called *mt* is created and one entity called *User* is created with four fields – UserName, Email, Password, File. UserName is the primary key.



4.4 OUTPUT DESIGN

The design of output is the most important task of any system. During output design, developers identify the type of outputs needed, and consider the necessary output controls.

Objectives of Output Design

- To develop output design that serves the intended purpose and eliminates the production of unwanted output.
- To develop the output design that meets the end users' requirements.
- To deliver the appropriate quantity of output.
- To form the output in appropriate format and direct it to the right person.
- To make the output available on time for making good decisions.

In this project, in the Access Document page, a card component is provided for displaying the medical report which is placed under the form. To view the Blockchain, URL is passed through Postman in GET method to view it in JSON format.

CHAPTER-5

SYSTEM TESTING

5.1 TESTING METHODOLOGIES

Software Testing Methodology is defined as strategies and testing types used to certify that the Application Under Test meets client expectations. Example testing methodologies include Unit Testing, Integration Testing, Functional Testing, System Testing, Acceptance Testing

5.2 UNIT TESTING

In Unit testing, we have to test the programs making up the system. For this reason, Unit testing is sometimes called as program testing. The software units in a system are the modules and routines that are assembled and integrated to perform a specific function. Unit testing focuses first on the modules, independently of one another, to locate errors. This enables to detect errors in coding and logic that are contained within the module alone

The testing was carried out during programming stage itself. In the testing step, each module is found to be working satisfactorily as regards to the expected output from the module. E.g. Checking whether the anchor tag for downloading books works properly.

Numbers of input/output operations, global and local variables, scope and validity, call to other modules, file attributes and database exceptions and expressions are also tested in each and every module.

5.3 INTERGRATION TESTING

Integrated testing is proceeded with bottom-up approach. In bottom-up integration testing, an individual module is first tested from a test harness. Once a set of individual modules has been tested, they are then combined into a collection of modules, known as builds, which are then tested by a second harness. This process can combine until the build consists of the entire application.

5.4 FUNCTIONAL TESTING

Functional testing is a type of software testing whereby the system is tested against the functional requirements/specifications. Functions (or features) are tested by feeding them input and examining the output. Functional testing ensures that the requirements are properly satisfied by the application. This type of testing is not concerned with how processing occurs, but rather, with the results of processing. It simulates actual system usage but does not make any system structure assumptions.

5.5 SYSTEM TESTING

System testing is a level of testing that validates the complete and fully integrated software product. The purpose of a system test is to evaluate the end-to-end system specifications. Usually, the software is only one element of a larger computer-based system. Ultimately, the software is interfaced with other software/hardware systems. System Testing is actually a series of different tests whose sole purpose is to exercise the full computer-based system.

5.6 ACCEPTANCE TESTING

Acceptance testing involves planning and executing of functional tests, performance tests and stress test in order to demonstrate that the implemented system satisfies its requirements. Functional test causes involve excising the code with nominal input values for which expected results are known. Giving different input values tests it.

Performance testing determines the amount of executing time spend in various paths of the program unit, program throughput, the response time and device the utilization by the program unit. With respect to the system performance testing is based on the maximum volume of existing data, which the system can handle with an effective throughput, and efficient utilization of the system resources.

Software system is developed in the above manner is one that satisfies the user needs, confirms to its requirement and design specification, and exhibits an absence of errors. The final process should be a software audit where the complete software project is checked to ensure that it meets production management requirement.

CHAPTER – 6

SYSTEM IMPLEMENTATION AND MAINTENANCE

Implementation is the stage of the project where the theoretical design is turned into a working system. At this stage the main work load, the greatest upheaval and the major impact on the existing system shifts to the user department. If the implementation is not carefully planned and controlled it can cause chaos and confusion.

Implementation includes all those activities that take place to convert from the old system to the new one. The new system may be totally new, replacing an existing manual or automated system or it may be a major modification to an existing system. Proper implementation is essential to provide a reliable system to meet the organization requirements.

The web application is developed and deployed using Flask web framework in the localhost for testing purposes. The web pages are loaded properly and all the navigation links are working as expected. The input form fields for uploading audio file and entering patient details are working properly. When the submit type button is clicked, the backend process for that function is triggered and expected output is achieved. Blockchain result can be viewed in Postman. The system can be implemented only after thorough testing is done and if it is found to be working according to the specifications. The system personnel check the feasibility of the system.

Successful implementation may not guarantee improvement in the organization using the new system, but improper installation will prevent it.

The implementation stage involves following tasks.

• Careful planning.

• Investigation of system and constraints.

• Design of methods to achieve the changeover.

• Training of the staff in the changeover phase.

• Evaluation of the changeover method.

Maintenance

The maintenance phase of the software cycle is the time in which a Software product performs useful work. After a system is successfully implemented, it should be maintained in a proper manner. System maintenance is an important aspect in the software development life cycle.

The first maintenance activity occurs because it is unreasonable to assume that software testing will uncover all latent errors in a large software system. During the use of any large program, errors will occur and be reported to the developer. The process that includes the diagnosis and correction of one or more errors is called corrective maintenance.

The second activity that contributes to a definition of maintenance occurs because of the rapid change that is encountered in every aspect of computing. Therefore, adaptive maintenance-an activity that modifies software to properly interfere with a changing environment is both necessary and common place.

The third activity that may be applied to a definition of maintenance occurs when a software package is successful. To satisfy requests in this category, perceptive maintenance is performed.

The fourth maintenance activity occurs when software is changed to improve future maintainability or reliability, or to provide a better basis for future enhancements. Often called preventive maintenance, this activity is characterized by reverse engineering and re-engineering techniques.

The web application is maintained on a regular basis. Navigation links and load speed are checked regularly. Web pages are checked for 404 errors and SEO, meta titles are reviewed. Speech API is updated frequently to provide best result to the user. And security checks are done regularly.

Feasibility Study

A feasibility study is an analysis that takes all of a project's relevant factors into account including economic, technical, legal, and scheduling considerations — to ascertain the likelihood of completing the project successfully.

The goals of the feasibility study are as follows.

- To understand thoroughly all aspects of a project, concept, or plan
- To become aware of any potential problems that could occur while implementing the project
- To determine if, after considering all significant factors, the project is viable—that is, worth undertaking

For example, an automobile prototype is a tool for the feasibility study, an experiment on rats to develop a new medicine is a procedure of feasibility analysis, checking the configuration and features before purchasing a laptop resembles feasibility tests.

Feasibility tests are done for the Blockchain and found to be reliable. Google Speech API is checked with multiple audio files. Web application is deployed using Flask on localhost and checked with sample inputs.

CHAPTER – 7
CONCLUSION

Medical transcribers deal with sensitive health information and they have specific obligations that are often protected by the law. Breachers in medical records can refer to a wide range of security issues that endanger a patient's confidentiality and trust in an organization. Many medical transcription companies offer tiered payments to complete the work faster, ignoring the quality that rushes reports may produce. So, these issues gave rise to the idea of usage of Blockchain. Medical Transcription process makes use of Google Speech-Recognition API which automates the process by converting audio files into text. Then the file is stored in the Blockchain. This solution is developed as web application where the user uploads the audio file and it is converted into text format and stored in Blockchain. A unique credential ID is generated and mailed to the patient. Using that the medical report can be accessed.

CHAPTER – 8

FUTURE ENHANCEMENTS

The future developments which can be made in this project are:

- Process for converting audio files to text will be further improved to identify two-person conversation into text.
- Process for converting audio files to text will be further improved to convert large audio files within minimum amount of time.
- More cryptographic techniques will be probed to incorporate in Blockchain to make it even more secure.

BIBLIOGRAPHY

Journal references:

- Estuar, Ma. Regina Justina E., Ph.D., Towards the development of a blockchain-enabled voice-to-text transcriber plugin in an electronic medical record for doctor-patient conversations, Ateneo de Manila University, (2019). https://archium.ateneo.edu/theses-dissertations/119/
- Dylan Yaga, Peter Mell, Nik Roby, Karen Scarfone, Blockchain Technology Overview, NISTIR 8202, October, 2018.
- Patel Nikunjkumar Sureshbhai, Pronaya Bhattacharya, Sudeep Tanwar, A Blockchain-Based Sentiment Analysis Framework for Fraud Cryptocurrency Schemes, IEEE International Conference on Communications Workshops (ICC Workshops), June (2020).

Web references:

- http://transcription411.com/ - Medical Dictation samples
- https://olympus.co.uk/ - Sample audio files
- https://codelabs.developers.google.com/codelabs/cloud-speech-text-python3#0
- https://flask.palletsprojects.com/en/2.1.x/quickstart/#routing – Flask routes

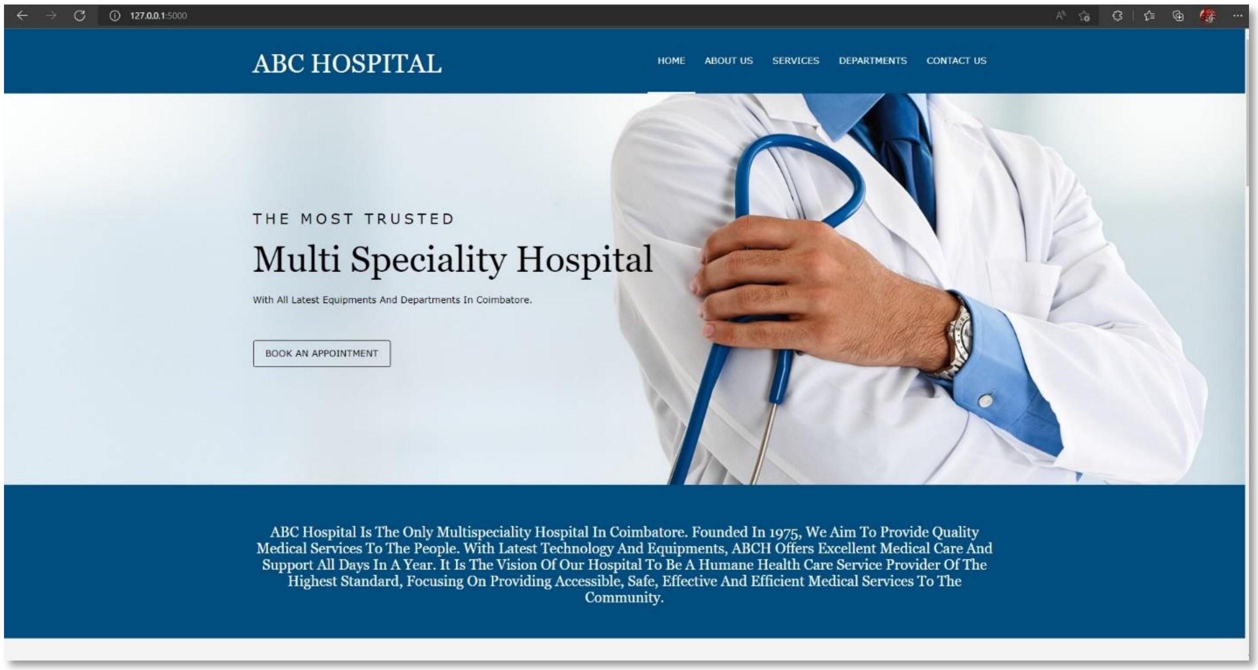
Book references:

- Learn SQLite with Python in 24 hours For Beginners – S. Basu - May 20, 2021
- Using SQLite: Small. Fast. Reliable. Choose Any Three – Jay A. Kreibich – O'Reilly – August 10, 2010

APPENDIX

A. Sample screenshots

Home page

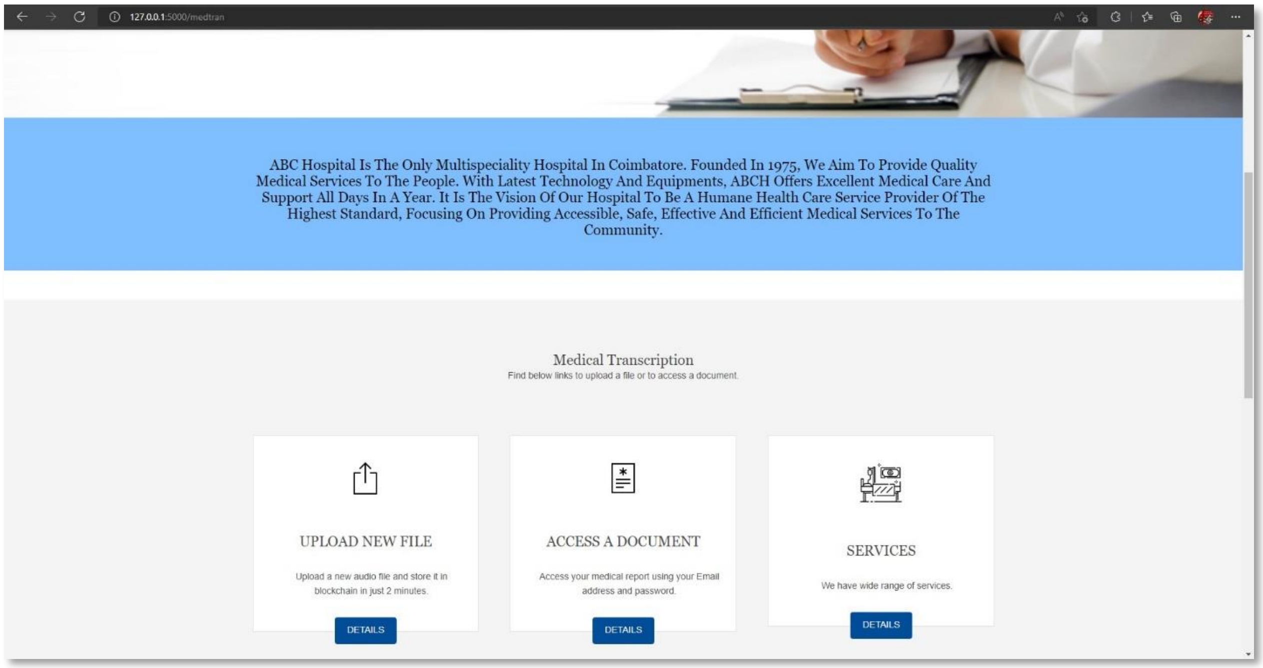


THE MOST TRUSTED

Multi Speciality Hospital

With All Latest Equipments And Departments In Coimbatore.

ABC Hospital Is The Only Multispeciality Hospital In Coimbatore. Founded In 1975, We Aim To Provide Quality Medical Services To The People. With Latest Technology And Equipments, ABCH Offers Excellent Medical Care And Support All Days In A Year. It Is The Vision Of Our Hospital To Be A Humane Health Care Service Provider Of The Highest Standard, Focusing On Providing Accessible, Safe, Effective And Efficient Medical Services To The Community.



ABC Hospital Is The Only Multispeciality Hospital In Coimbatore. Founded In 1975, We Aim To Provide Quality Medical Services To The People. With Latest Technology And Equipments, ABCH Offers Excellent Medical Care And Support All Days In A Year. It Is The Vision Of Our Hospital To Be A Humane Health Care Service Provider Of The Highest Standard, Focusing On Providing Accessible, Safe, Effective And Efficient Medical Services To The Community.

Medical Transcription

Find below links to upload a file or to access a document.

UPLOAD NEW FILE

Upload a new audio file and store it in blockchain in just 2 minutes.

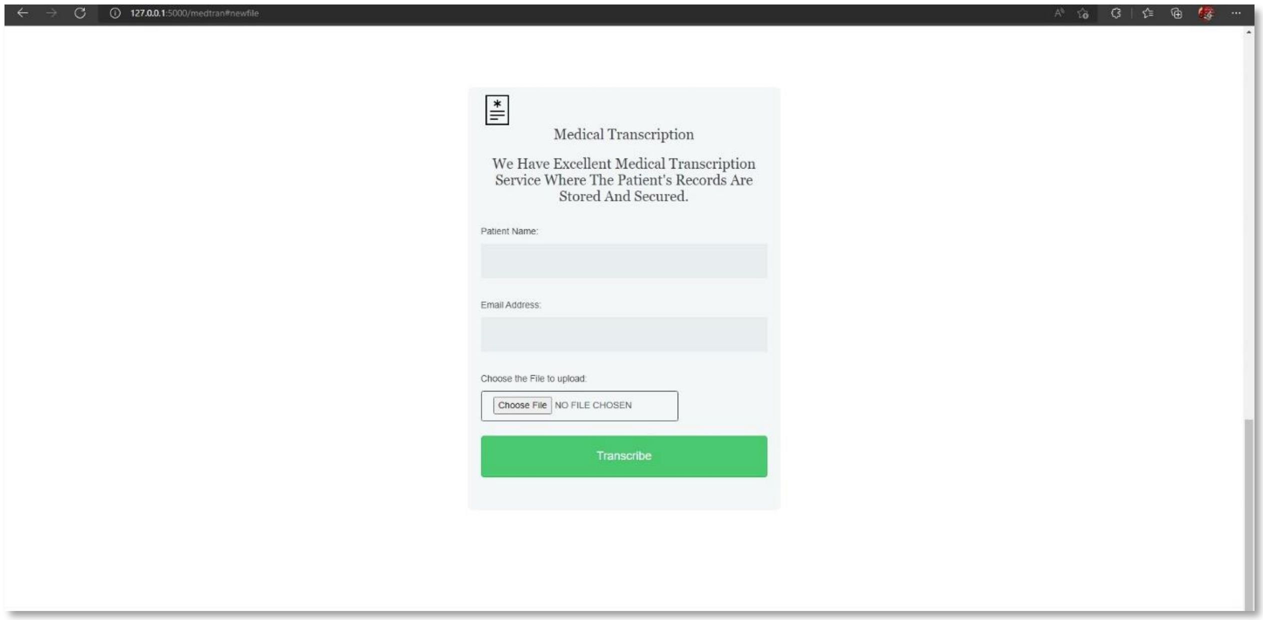
ACCESS A DOCUMENT

Access your medical report using your Email address and password.

SERVICES

We have wide range of services.

Audio file upload form



Medical Transcription

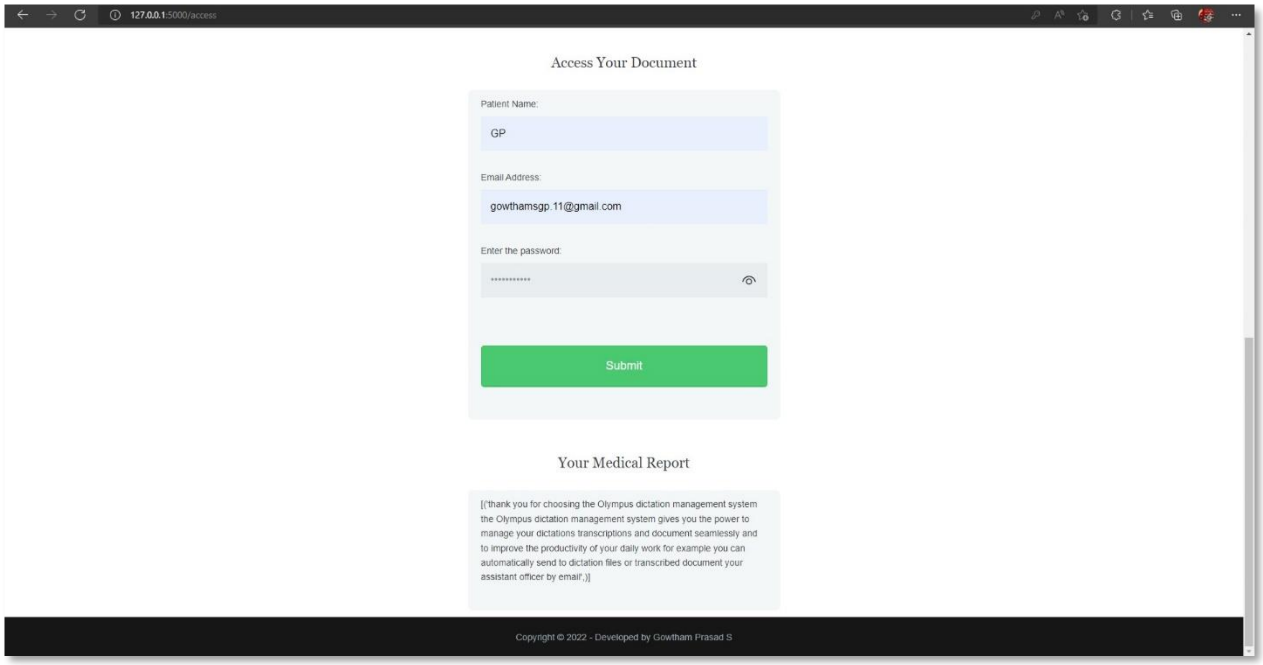
We Have Excellent Medical Transcription Service Where The Patient's Records Are Stored And Secured.

Patient Name:

Email Address:

Choose the File to upload:

Accessing medical report



Access Your Document

Patient Name: GP

Email Address: gowthamsgp.11@gmail.com

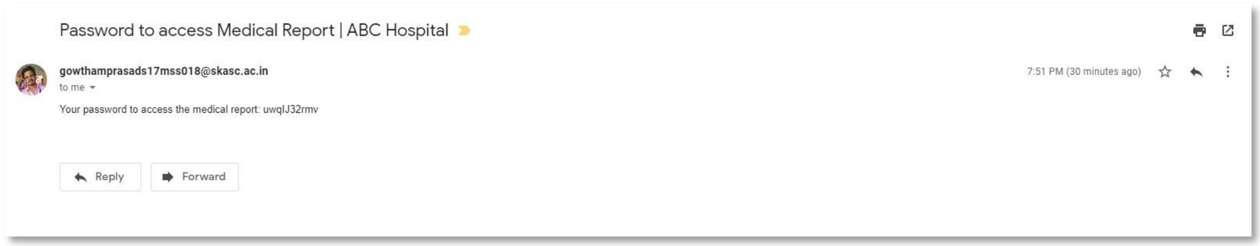
Enter the password: ***********

Your Medical Report

[('thank you for choosing the Olympus dictation management system the Olympus dictation management system gives you the power to manage your dictations transcriptions and document seamlessly and to improve the productivity of your daily work for example you can automatically send to dictation files or transcribed document your assistant officer by email',)]

Copyright © 2022 - Developed by Gowtham Prasad S

Password mailed to patient



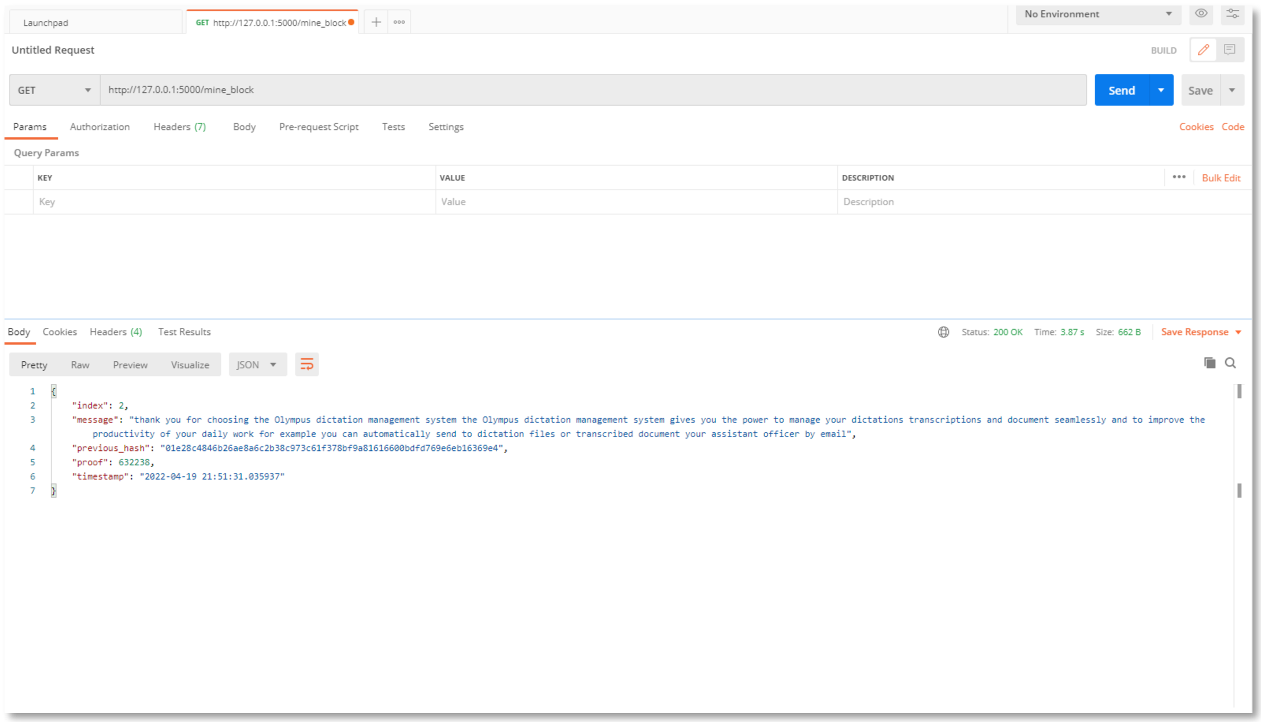
Password to access Medical Report | ABC Hospital

gowthamprasads17mss018@skasc.ac.in
to me

7:51 PM (30 minutes ago)

Your password to access the medical report: uwqlJ32rmv

Blockchain viewed in Postman



```
{
    "index": 2,
    "message": "thank you for choosing the Olympus dictation management system the Olympus dictation management system gives you the power to manage your dictations transcriptions and document seamlessly and to improve the
        productivity of your daily work for example you can automatically send to dictation files or transcribed document your assistant officer by email",
    "previous_hash": "01e28c4846b26ae8a6c2b38c973c61f378bf9a81616600bdfd769e6eb16369e4",
    "proof": 632238,
    "timestamp": "2022-04-19 21:51:31.035937"
}
```

B. Sample source code

```
# IMPORT REQUIRED PACKAGES AND LIBRARIES

import numpy as np

import string # FOR WORKING WITH STRINGS

import random # FOR PASSWORD GENERATION

from flask import Flask, flash, request, redirect, url_for,
jsonify, render_template # FOR FLASK AND RELATED FEATURES

import pickle

import speech_recognition as sr # GOOGLE SPEECH RECOGNITION API

import datetime # FOR TIMESTAMP

import hashlib # CALCULATING THE HASH IN ORDER TO ADD DIGITAL
FINGERPRINTS TO THE BLOCKS

import json # TO STORE DATA IN BLOCKCHAIN

import sqlite3 as sql # TO STORE THE DATA

import smtplib # TO SEND EMAIL

from email.message import EmailMessage # TO SEND EMAIL


# BLOCKCHAIN CLASS

class Blockchain:

     # This function is created to create the very first block
and set it's hash to "0"

    def __init__(self):

        self.chain = []
        self.create_block(message='alternate', proof=1,
previous_hash='0')


     # This function is created to add further blocks into the
chain

    def create_block(self, message, proof, previous_hash):

        block = {'index': len(self.chain) + 1,

                    'timestamp': str(datetime.datetime.now()),

                    'proof': proof,

                    'previous_hash': previous_hash,

                'message': message}

        self.chain.append(block)

        return block


     # This function is created to display the previous block

    def print_previous_block(self):

        return self.chain[-1]


     # This is the function for proof of work and used to
successfully mine the block

    def proof_of_work(self, previous_proof):

        new_proof = 1

        check_proof = False
        while check_proof is False:

            hash_operation = hashlib.sha256(

                str(new_proof**2 -
previous_proof**2).encode()).hexdigest()

            if hash_operation[:5] == '00000':

                check_proof = True

            else:

                new_proof += 1



        return new_proof



    def hash(self, block):

        encoded_block = json.dumps(block,
sort_keys=True).encode()

        return hashlib.sha256(encoded_block).hexdigest()



    def chain_valid(self, chain):

        previous_block = chain[0]

        block_index = 1



        while block_index < len(chain):

            block = chain[block_index]
            if block['previous_hash'] !=
self.hash(previous_block):
                return False

            previous_proof = previous_block['proof']
            proof = block['proof']
            hash_operation = hashlib.sha256(
                    str(proof**2 -
previous_proof**2).encode()).hexdigest()

            if hash_operation[:5] != '00000':
                return False
            previous_block = block
            block_index += 1

        return True

# This Function used to generate password
characters = list(string.ascii_letters + string.digits +
"!@#$%^&*()")

def generate_random_password():
     length = 10
     # shuffling the characters
     random.shuffle(characters)


     password = []
     for i in range(length):
          password.append(random.choice(characters))


     random.shuffle(password)


     # converting the list to string and printing the list
     return("".join(password))
#----------------------------------------------------------------
----------------------------#
# FLASK APP #
# Flask app initialization
app = Flask(__name__)
app.secret_key = "super secret key"


# Rendering Home page - index.html
@app.route('/')
def home():
    return render_template('index.html')
```

```
# Rendering Medical Transcription page - sample.html

@app.route('/medtran')

def med():

    return render_template('sample.html')


# Medical Transcription process route

@app.route('/transcribe',methods=['POST','GET'])

def transcribe():

    global text

    # Initialize recognizer class (for recognizing the speech)

    r = sr.Recognizer()


    # Reading Audio file as source, listening the audio file and
store in audio_text variable


    if request.method == 'POST':

        myfile = request.files['file']


    with sr.AudioFile(myfile) as source:

        audio_text = r.listen(source)
        # recoginize_() method will throw a request error if the
API is unreachable, hence using exception handling

        try:

            # using google speech recognition

            text = r.recognize_google(audio_text)

            print('Converting audio transcripts into text ...')

            # print(text)


        except:

             print('Sorry...run again...')


    # write transcribed text to a text file

    with open("test.txt", "w") as fo:

        fo.write(text)


    # Insert patient name, password and transcribed text into
'mt.db'

    user_name=None

    password=None

    email=None

    password=generate_random_password() # Calling Password
Generation function

    if request.method == 'POST':
        user_name=request.form['name']

        email=request.form['email']

        con = sql.connect('mt.db')

        print("Connected successfully")

        cur = con.cursor()

        cur.execute('INSERT INTO
User(UserName,Password,File,Email) VALUES
(?,?,?,?)',(user_name,password,text,email))

        con.commit()

        print("Data inserted successfully")

        con.close()


    # Send password to patient email address

    # Initialise EmailMessage()

    msg = EmailMessage()

    # message to be sent

    msg.set_content('Your password to access the medical report:
'+password)


    msg['Subject'] = 'Password to access Medical Report | ABC
Hospital'

    msg['From'] = "gowthamprasads17mss018@skasc.ac.in"

    msg['To'] = email
    # creates SMTP session

    server = smtplib.SMTP_SSL('smtp.gmail.com', 465)

    # Authentication

    server.login("gowthamprasads17mss018@skasc.ac.in", "good
luck bro")

    server.send_message(msg)

    server.quit()

    print("Password mailed successfully")




    return render_template('sample.html',transcribed_text=text)

    #return redirect(url_for('mine_block',text=text))


# Rendering Access Document page - read.html

@app.route('/read')

def read():

    return render_template('read.html')


@app.route('/access',methods=['POST','GET'])

def access():

    if request.method == 'POST':

        user_name=request.form['name']

        password=request.form['password']
        con = sql.connect('mt.db')
        print("Connected successfully")
        cur = con.cursor()

        res=con.execute('select UserName from user where
UserName=? and Password=?', (user_name,password)).fetchall()
        res1=con.execute('select File from user where UserName=?
and Password=?', (user_name,password)).fetchall()
        return render_template('read.html',
transcribed_text=res1)

    # else:
    #     flash('Invalid password or username!')
    #     return render_template('read.html')


# OBJECT CREATION FOR BLOCKCHAIN CLASS
blockchain = Blockchain()

# Mining a new block
@app.route('/block/', methods=['GET','POST'])
def mine_block():
    msg = text
    previous_block = blockchain.print_previous_block()

    previous_proof = previous_block['proof']

    proof = blockchain.proof_of_work(previous_proof) # Calling
proof_of_work function

    previous_hash = blockchain.hash(previous_block) # Calling
hash function

    message = msg

    block = blockchain.create_block(message, proof,
previous_hash) # Calling create_block function with parameters


    response = {'message': block['message'],

                'index': block['index'],

                'timestamp': block['timestamp'],

                'proof': block['proof'],

                'previous_hash': block['previous_hash']}


    return jsonify(response), 200


# Display blockchain in json format

@app.route('/get_chain', methods=['GET'])

def display_chain():

    response = {'chain': blockchain.chain,

                'length': len(blockchain.chain)}
    return jsonify(response), 200


# Check validity of blockchain

@app.route('/valid', methods=['GET'])

def valid():

    valid = blockchain.chain_valid(blockchain.chain) # Calling
the chain_valid function


    if valid:

        response = {'message': 'The Blockchain is valid.'}

    else:

        response = {'message': 'The Blockchain is not valid.'}

    return jsonify(response), 200




# EXECUTION OF FLASK APP

if __name__ == "__main__":

    app.run(debug=True)
```