1. INTRODUCTION

Organ failure or damage occurs due to an injury or a disease. It affects the quality of life and, in some cases, leads to death. Donating an organ is one of humanity's most honorable actions to save the lives of patients through organ transplantation. For a successful transplant, the organ must be in acceptable working conditions with donor-recipient matching, and its removal should not pose a life-threatening risk to the donor .The first successful organ donation occurred with a kidney transplant between twin brothers in 1954 . Since then, the annual number of transplants has steadily increased. However, the demand for organ donations still exceeds the number of donors . In fact, while waiting for an organ transplant, twenty people die every day, and a new patient is added to the waiting list in every ten minutes . More importantly, accessing the organ donation waiting list is a basic requirement for organ allocation. Referral for transplantation can be affected by both geographical and socioeconomic factors. Therefore, the allocation process on the waiting list should not discriminate against certain groups of patients.

Organ donation is conducted in two different ways, including deceased donation and living donation. Illustrates the typical flow chart for donating an organ and transplanting it to a patient. First, the donor is examined by the hospital transplant team, and if the donor is deceased, a brain death test is performed. Meanwhile, if the donor is still alive, doctors examine the donor and ensure that the donor is fit for live donation. Then, all medical records are reported to the procurement organizer. The procurement organizer is responsible for evaluating the donor's condition to decide if he is a fit donor and ensuring that the donor is properly registered in the medical system. Next, if the evaluation shows that the donor is eligible for donation, the procurement organizer sends all the data to the organ transplantation organizer. This step can be performed only if the donor gives consent to donate to an anonymous person. After that, the matching process between the available donors and patients on the waiting list is performed by the organ transplantation organizer. As a result, a ranked list is generated as an output and provided to the transplantation surgeons. Next, the transplant surgeon decides whether the organ is appropriate for the patient based on various considerations, such as the donor's medical records and the current health of the prospective recipient. Later, when a transplant surgeon accepts the

donated organ, the donor's surgeon is notified to remove the donated organ. Finally, the donated organ is transported to the patient's hospital and received by the transplant surgeon. However, suppose the situation is for a live donor and it has been planned to donate to a known person by name. In that case, the data will go directly to the transplant surgeon to start the surgery of removing and transplanting the donated organ.

In the past, when a patient died or was near death, the organ procurement organization and hospital worked together to do an initial medical test to decide if the patient could be an organ donor. This call takes around 15 minutes, and only 6% of these calls result in possible organ donors' being identified. Over the years, this phone call has been replaced by an instant message generated by central computer systems that store all the data required for this process . However, the core issue with this strategy is that the security and validity of such data are entirely dependent on the transplantation centers' ability to keep their systems secure and identify potential harm to donors and recipients. The accuracy of the wait-list data is largely dependent on people's faith and trust in these centers' ability to keep it secure from hackers and fraudulent employees . Moreover, transparency is another challenge affecting the success of the organ donation process. According to World Health Organization (WHO) reports, up to 10% of transplanted organs may have been obtained unethically via organ trafficking, but the exact numbers are unknown . The lack of transparency in the current system among participants leads to illegal organ trade and purchases and medical professionals engaging in unethical practices . Moreover, there are hospitals that take advantage of the patient's need for the organ and offer the opportunity to transfer the organ to those who pay a higher amount to the hospital while ignoring the patient with the highest priority on the waiting list . In addition, current transplant systems are also frequently slow, which is unacceptable in such a critical and life-threatening scenario. Such systems are hardly up to date with the minimum security standards. So far, there has recently been a surge in security breaches affecting user privacy and system integrity. In general, modern systems manage data through the use of standard databases; however, most hospitals, health ministries, and other medical facilities lack a standardized data communication system.

1.1 Motivation

The motivation for proposing a private Ethereum blockchain-based solution for organ donation and transplantation management stems from the need to address critical challenges present in current systems. These challenges include the lack of transparency and trust, inefficiencies in matching donors with recipients, concerns about patient data privacy and security, and the need for traceability and auditability throughout the process. By leveraging blockchain technology, we aim

to decentralize control, enhance security through cryptographic protocols, improve transparency and traceability, automate processes with smart contracts, ensure patient privacy, and conduct rigorous performance evaluations. Ultimately, our goal is to create a fair, efficient, secure, and trustworthy system that prioritizes patient experience, ethical considerations, and regulatory compliance in organ donation and transplantation.

1.2 Problem Defination

The problem at hand involves the complex landscape of organ donation and transplantation systems, which face multifaceted challenges across several key stages. Firstly, there is the issue of registration, where the system must efficiently and accurately onboard donors and recipients while safeguarding their data privacy. Secondly, the critical task of donor-recipient matching demands precise algorithms and criteria to ensure compatibility, urgency, and ethical considerations are met. The process of organ removal must then be carefully coordinated, encompassing medical evaluations, consent management, and logistical arrangements. Subsequently, organ delivery poses logistical challenges, necessitating secure and timely transportation methods that preserve organ viability. The transplantation phase involves intricate surgical procedures, pre and post-operative care, and ongoing monitoring. Throughout these stages, legal, clinical, ethical, and technical constraints loom large, requiring adherence to regulations, clinical best practices, ethical guidelines, and overcoming technical hurdles related to data management, security, and interoperability. In essence, the problem is to design a comprehensive organ donation and transplantation

management system that navigates these challenges, ensures fairness, efficiency, and patient-centered care, and fosters trust among all stakeholders involved in the process.

1.3 Objectives of the Project

The main objectives are diverse, aiming to tackle the intricate challenges of organ donation and transplantation systems using a private Ethereum blockchain approach. It seeks to establish decentralization for transparency and accessibility, enhance security through cryptographic protocols, ensure traceability for trust and accountability, automate processes via smart contracts to improve efficiency, prioritize privacy with encryption and controlled access, evaluate performance rigorously, adhere to ethical and regulatory standards, engage stakeholders for feedback and support, and continually refine the system based on real-world usage. Overall, the project aims to create a robust, efficient, secure, and ethical organ donation and transplantation management system that benefits patients and stakeholders alike.

2.Literature Survey

“Organ donation decentralized application using blockchain technology,”

L. A. Dajim, S. A. Al-Farras, B. S. Al-Shahrani, A. A. Al-Zuraib, and R. Merlin Mathew

The proposed system is an organ donation decentralized app using blockchain technology. It would be a web application for patients to register their information-most importantly medical ID, blood type, organ type and state. The system would work on a first-in, first-out basis unless a patient is in critical condition.

“Using blockchain technology for the organ procurement and transplant network,”

U. Jain,

The organ donation system in the United States is centralized and difficult to audit by the general public. This centralized approach may lead to data integrity issues in the future. The Organ Procurement and Transplant Network (OPTN) was built and maintained by a non-governmental organization called the United Network for Organ Sharing (UNOS) under its proprietary UNet(SM) umbrella platform. This platform is made up of proprietary closed source software and does not provide the general public easy access to the organ transplant data for auditing. This study investigates the feasibility, challenges, and advantages of a blockchain-based OPTN.

A prototype of a blockchain-based OPTN was created using the Hyperledger Fabric framework. The policies and guidelines issued by the United States Department of Health and Human Services for UNOS and the OPTN were used as the basis of this prototype. Four factors were identified to have a direct effect on the performance of this system, viz. max batch time out, max block size, endorsement policy, and transaction rate. Additionally, two variants of the blockchain chaincode were also developed. The first variant performed the organ-candidate matching inside the blockchain (Scheme A), and the second variant performed it outside the blockchain (Scheme B).

Analysis of these data showed that Scheme A outperformed Scheme B in all experiments for write-operations. However, the read operations remained unaffected by any of the experiment variables in the given environment.
Based on these results, it is recommended to perform the organ-candidate matching on the blockchain with the max batch time out close to the transaction rate.

"Use of forensic DNA testing to trace unethical organ procurement and organ trafficking practices in regions that block transparent access to their transplant data,"

M. He, A. Corson, J. Russo, and T. Trey

This study proposes an approach to track unethically procured organs in particular in countries or regions where investigations cannot be performed by utilizing forensic DNA methodology. Using China as an example, previous research has concluded that organs in China are in part unethically and extra-legally procured (so called "forced organ harvesting") from living prisoners of conscience without consent. Using forensic DNA-analysis, we propose building a DNA data bank from missing prisoners of conscience in China and comparing these results with DNA from donor organs in patients who received transplants in China. Biological materials collected in China will provide DNA directly or indirectly from potential victims of forced organ harvesting. Archival biopsies from transplant recipients' donor organs will provide DNA profiles of donors. Verified match between DNA profiles of transplanted organs and missing victims will establish proof of such connection, thus provides evidence despite a lack of transparency.

"Decentralised and distributed system for organ/tissue donation and transplantation,"

P. Ranjan, S. Srivastava, V. Gupta, S. Tapaswi, and N. Kumar

In today's era of digitisation, many technologies have evolved that every manual work can be digitally automatized. In the digital automatizing process, security and privacy are the most important and highly demanding aspects. Blockchain offers many

features that can be used in almost every sphere of life. Features like decentralisation, transparency, privacy makes it an extremely useful technology. Therefore, by making use of all these features, several problems in healthcare sector can be solved like removing complex network of third parties and lack of traceability of transactions. This paper presents a decentralised, secure and transparent organ and tissue transplant web application (also called DApp), which not only nullifies the role of any third party involved in the organ transplantation, but also is a cost effective solution that saves the patient's from high cost of transplantation. The details and Electronic Medical Record(EMR) are hashed using the IPFS(a distributed file server), which reduces the cost of upload to a great extent as shown in the results section of this paper.

"A systematic review of the use of blockchain in healthcare,"

M. Hölbl, M. Kompara, A. Kamišalić, and L. N. Zlatolas,

Blockchain, a form of distributed ledger technology has attracted the interests of stakeholders across several sectors including healthcare. Its' potential in the multi-stakeholder operated sector like health has been responsible for several investments, studies, and implementations. Electronic Health Records (EHR) systems traditionally used for the exchange of health information amongst healthcare stakeholders have been criticised for centralising power, failures and attack-points with exchange data custodians. EHRs have struggled in the face of multi-stakeholder and system requirements while adhering to security, privacy, ethical and other regulatory constraints. Blockchain is promising amongst others to address the many EHR challenges, primarily trustless and secure exchange of health information amongst stakeholders. Many blockchain-in-healthcare frameworks have been proposed; some prototyped and/or implemented. This study leveraged the PRISMA framework to systematically search and evaluate the different models proposed; prototyped and/or implemented. The bibliometric and functional distribution of all 143 articles from this study were presented. This study evaluated 61 articles that discussed either prototypes or pilot or implementations. The technical and architectural analysis of these 61 articles for privacy, security, cost, and performance were detailed. Blockchain was

found to solve the trust, security and privacy constraints of traditional EHRs often at significant performance, storage and cost trade-offs.

3. Analysis

3.1 Existing System

The existing systems for organ donation and transplantation, despite their innovative approaches, have some notable disadvantages. One major drawback is the lack of blockchain implementation, leading to concerns about security and communication between hospitals and donors. Without blockchain technology, these systems may be more vulnerable to data breaches and unauthorized access, potentially compromising patient confidentiality and system integrity. Additionally, the absence of an automated matching process based on specific criteria using smart contracts is a significant limitation. This manual matching process can be time-consuming, prone to errors, and may not optimize the allocation of organs based on critical factors such as organ quality and recipient needs. Overall, these limitations highlight the need for advancements in leveraging blockchain and smart contract technologies to enhance security, efficiency, and fairness in organ donation and transplantation systems.

- The system is not implemented blockchain based organ donation which leads less security and less communication between hospitals and donors..
- The system is not implemented an auto-matching process between the donor and recipient through a smart contract based on certain criteria.

3.2 Proposed System

The advantages of the proposed Blockchain Based Management for Organ Donation and Transplantation are as follows:

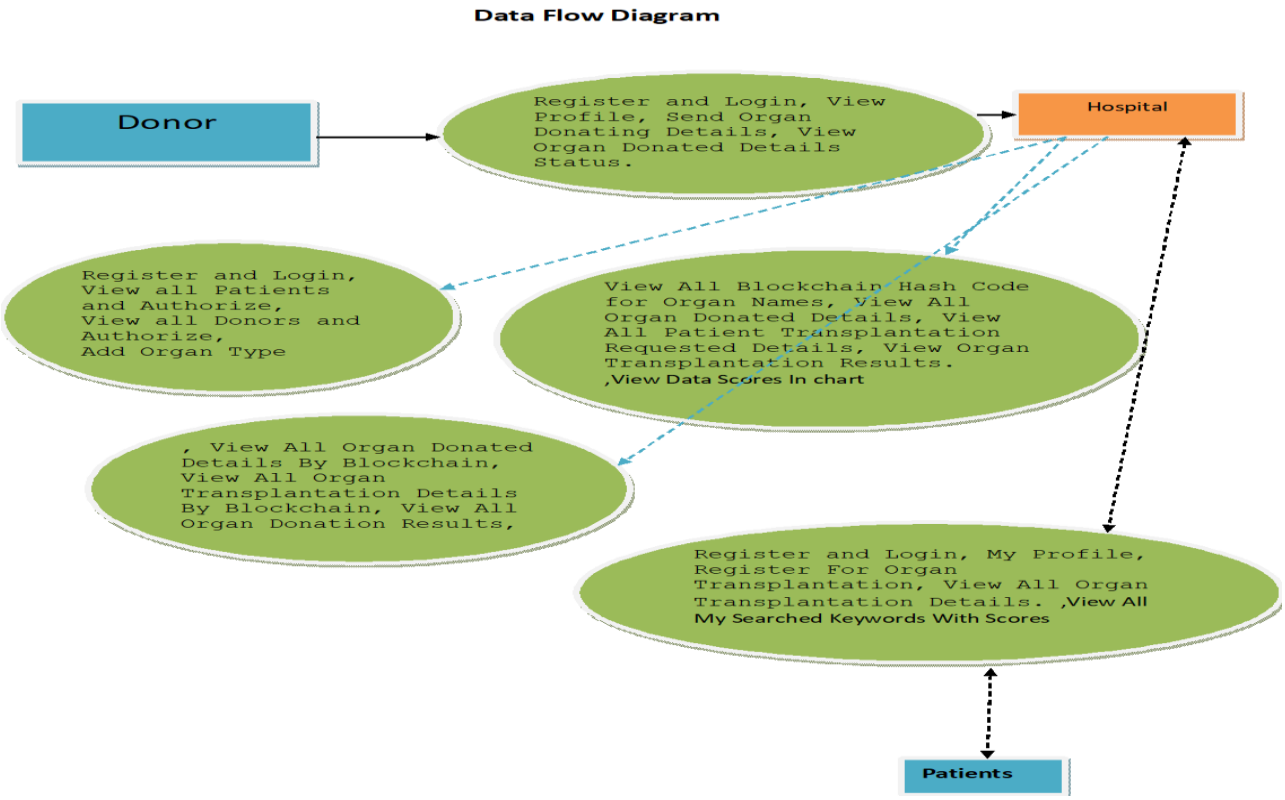

Fig:3.1: Flow Chart

- **Decentralized and Secure Management**:Leveraging a private Ethereum blockchain-based solution ensures decentralized, secure, reliable, and trustworthy management of organ donation and transplantation processes.
- **Smart Contracts for Data Provenance**:The development of smart contracts registers actors and ensures data provenance through event tracking, enhancing transparency and accountability in the system.

- **Automated Matching Process**:Implementation of an auto-matching process between donors and recipients through smart contracts based on specific criteria reduces manual efforts and potential errors, improving efficiency.
- **Security Analysis and Comparison**:Conducting security analysis to identify and mitigate common security attacks and vulnerabilities ensures the system's resilience and readiness for deployment. Comparison with existing solutions demonstrates its novelty and effectiveness.

3.3 Software Requirement Specification

3.3.1 Purpose

The purpose of the Blockchain Based Management for Organ Donation and Transplantation
is to introduce the challenges and requirements faced by today's organ donation and transplantation systems, highlighting the need for an end-to-end solution that addresses these challenges while ensuring fairness, efficiency, patient experience enhancement, and trust in the process. The paragraph also outlines the proposed solution using a private Ethereum blockchain-based approach, emphasizing its key attributes such as decentralization, security, traceability, auditability, privacy, and trustworthiness. Additionally, it mentions the development of smart contracts, algorithms, and the evaluation criteria for the proposed solution, setting the stage for the rest of the paper.

3.3.2 Scope

The scope of this Blockchain Based Management for Organ Donation and Transplantation
encompasses a comprehensive examination of the challenges and requirements within organ donation and transplantation systems. This includes registration procedures, donor-recipient matching protocols, organ removal and delivery logistics,

as well as the transplantation process itself. Legal, clinical, ethical, and technical constraints are considered throughout, reflecting the multifaceted nature of managing organ donation and transplantation.

The proposed solution focuses on developing a private Ethereum blockchain-based system that addresses these challenges. Key aspects such as decentralization, security, traceability, auditability, privacy, and trustworthiness are central to the proposed solution. The development of smart contracts and algorithms plays a crucial role in optimizing the system's functionality and efficiency.

3.3.3 Overall Description

H/W System Configuration: -

- Processor - Pentium –IV
- RAM - 4 GB (min)
- Hard Disk - 20 GB
- Key Board - Standard Windows Keyboard
- Mouse - Two or Three Button Mouse
- Monitor - SVGA

Software Requirements:

- Operating System - Windows XP
- Coding Language - Java/J2EE (JSP, Servlet
- Front End - J2EE
- Back End - MySQL

4.IMPLEMENTATION

Donors

In this module, the Donor will register and login then uploads their organ donor data to the Hospital and will do the following operations such as View Profile, Send Organ Donating Details, View Organ Donated Details Status.

Patients

In this module, patients logs in by using his/her user name and password. After Login User will do so

me operations such as My Profile, Register For Organ Transplantation, View All Organ Transplantation Details.

Hospital

The Hospital manages Hospital records to provide organ storage service for donation and transplantation and also performs the following operations such as View all Patients and Authorize, View all Donors and Authorize,Add Organ Type, View All Blockchain Hash Code for Organ Names,View All Organ Donated Details,View All Patient Transplantation Requested Details,View All Organ Donated Details By Blockchain,View All Organ Transplantation Details By Blockchain, View All Organ Donation Results,View Organ Transplantation Results.

Front End and User Interface Design:

The entire user interface is planned to be developed in browser specific environment with a touch of Intranet-Based Architecture for achieving the Distributed Concept.

The browser specific components are designed by using the HTML standards, and the dynamism of the designed by concentrating on the constructs of the Java Server Pages.

Java Language Choice:

Java, selected as the language for the project, stands out due to its platform independence, cohesiveness, and consistency. Originally named "Oak," Java emerged in 1995 with a primary motivation for a platform-independent, architecture-neutral language suitable for embedded software in various consumer electronic devices. Java offers full control to programmers within the constraints of the Internet environment. Its significance in Internet programming is likened to C's role in system programming.

Java's Impact on the Internet:

Java has left an indelible mark on the Internet, expanding the universe of objects that can move freely in cyberspace. In networked environments, two categories of objects—passive information and dynamic active programs—are transmitted between servers and personal computers. Java applets, small programs dynamically downloaded across the network, introduced a new form of executable programs in cyberspace. Java addresses security concerns associated with downloading executable programs, ushering in a new era of program development known as Applets.

Java Architecture:

The architecture of Java provides a portable, robust, and high-performance environment for development. Java's uniqueness lies in its bytecode, an optimized set of instructions executed by the Java Virtual Machine (JVM). The JVM, integral to Java technology, acts as an interpreter for bytecode, ensuring portability across

various platforms. The development process involves writing Java source code, compilation into bytecode using the Java compiler (javac), and execution in the JVM.

Java Database Connectivity (JDBC):

JDBC, short for Java Database Connectivity, represents a critical Java API designed for executing SQL statements. More than just a set of classes and interfaces, JDBC provides a standard API for tool and database developers, enabling the creation of database applications using pure Java. The three primary functions of JDBC include establishing a connection with a database, sending SQL statements, and processing the results.

Two-Tier and Three-Tier Models:

JDBC supports both the two-tier and three-tier models for database access. In the two-tier model, a Java applet or application communicates directly with the database. This requires a JDBC driver capable of interfacing with the specific database management system. The three-tier model introduces a middle tier, where commands are sent to services that communicate with the database. The three-tier architecture is favored for its enhanced control over access and updates to corporate data, offering performance advantages and a simplified API.

JDBC Driver Types:

Several JDBC driver types exist, categorized into four groups: JDBC-ODBC bridge plus ODBC driver, Native-API partly-Java driver, JDBC-Net pure Java driver, and Native-protocol pure Java driver. Each type caters to specific requirements, with considerations for security, robustness, and ease of use. The choice of JDBC driver depends on the project's needs and the target database.

Java Server Pages (JSP):

Java Server Pages, a technology for creating dynamic-content web pages, emerges as a simple yet powerful tool based on the Java programming language. This architecture facilitates the separation of content generation from presentation, allowing different team members to focus on their expertise. Java Server Pages offer proven portability,

open standards, and a mature re-usable component model, making them a preferred choice for building dynamic web applications.

Tomcat 6.0 web server

Tomcat is an open source web server developed by Apache Group. Apache Tomcat is the servlet container that is used in the official Reference Implementation for the Java Servlet and Java Server Pages technologies. The Java Servlet and Java Server Pages specifications are developed by Sun under the Java Community Process. Web Servers like Apache Tomcat support only web components while an application server supports web components as well as business components (BEAs Web logic, is one of the popular

4.1 List of program files

Data Trust Framework Using Blockchain.txt: This text file likely describes a framework for establishing trust in data sharing using blockchain technology. It could detail the components, functionalities, and benefits of the framework.

Database: A database file storing actual data. (Further context needed for confirmation)

A text file containing information about a database, such as its schema, connection details, or tables.

Gallery/images (directory): This directory likely stores image files. It could be used for a web application's image gallery or for storing other visual content.

sources (directory): This directory likely contains source code files, potentially related to the project's functionalities.

WEB-INF (directory): This directory typically holds configuration files for a web application. These files define settings for the application's behavior and operation.

Attack.java, Attack1.java, Attack2.java: These text files could contain information about security vulnerabilities or specific attack methods identified in the system.

auditor.java, auditormain.java: These Java source code files likely implement functionalities related to auditing the system's activities or data access.

authentication.java, authentication1.java: These Java source code files likely handle user authentication mechanisms for the system, ensuring secure access.

chart_bg.jpg: This GIF image file is likely used as a background image for charts or data visualizations within the system.

connect.java: This Java source code file likely establishes connections to a database, allowing the system to interact with and retrieve data.

csprovider.java: This Java source code file's purpose is related to a csprovider, which might be a data provider component or a database connection provider.

DeleteFile.java, DeleteFile1.java: These Java source code files likely provide functionalities for deleting files within the system, following proper authorization procedures.

Download.java, Download1.java, Download2.java: These Java source code files likely handle functionalities for downloading files from the system, ensuring secure and authorized access.

endusermain.java, EndUsers.java: These Java source code files are likely related to the end-user interface or functionalities specific to end-users interacting with the system.

FileRequest.java, FileRequest1.java: These Java source code files likely handle functionalities related to user requests for accessing or downloading files within the system.

images: This could be a directory containing images or a file related to images.

images1: This could be an image file or a directory containing images.

index: This file is main program index file which is loaded as the main page on web.

Insertdata.java: This could be a script or program for inserting data into a database or system.

Register.java: This could be a program or script related to user registration or account creation.

OViewFiles: This could be a program or script used for viewing files within the system.

ownermain: This could be the main program file for managing or overseeing ownership rights within the system.

Process: This is a general term and could refer to a program file or script related to a specific process within the system. More context is needed to determine the exact functionality.

recover: This could be a program or script used for data recovery purposes.

register: This could be a program or script related to user registration or account creation.

Results: This is a general term and could refer to a file containing output or data resulting from a process. More context is needed to determine the specific content.

Results 1: This could be a specific file containing output or data resulting from a process. More context is needed to determine the specific content.

servermain: This is the main program file for the server-side functionality of the system.

TD Results: This is a file containing results related to the Data Trust framework, potentially showing outcomes or analysis.

TPA Attackers: This is a file containing information about attackers or attack methods related to Trust Protection Agreements (TPAs).

TPA Hash Table: This is a file or data structure used to store information related to Trust Protection Agreements (TPAs), potentially using a hash table for efficient access.

TPA Results: This is a file containing results related to Trust Protection Agreements (TPAs), potentially showing outcomes or analysis.

TPA Transaction: This is a file containing information about transactions secured by Trust Protection Agreements (TPAs).

tpavb: The purpose of this file is unclear without more context. It could be related to Trust Protection Agreements (TPAs) but the specific meaning of "tpavb" is unknown.

TPT Results: This could be a file containing results related to the Trust Protection Technology (TPT), potentially showing outcomes or analysis.

UpdateBlock: This could be a program or script used to update a block of data within the system.

UpdateBlock1, UpdateBlock2, UpdateBlock3: These could be program or script files used to update specific blocks of data within the system, potentially in a sequence.

Upload: This could be a program or script used for uploading files to the system.

verifyblock, verifyblock2, verifyblock3, verifyblock8: These files likely contain data related to verification blocks, which could be a unit of data on a blockchain used to store information about a transaction or record.

usermain: This file could be the main user file for the system.

vb, vb1: These is the abbreviation variations of "verifyblock".

VerifyBlock, VerifyBlock1: These could be program files related to the VerifyBlock application.

ViewAllDataOwners,ViewAttackers,ViewFileRequest,ViewFileResponse, ViewHashTable, ViewResults, ViewResults1, ViewResults2, View Transaction: These files likely contain functions used for viewing data or system results. For instance, "ViewAllDataOwners" could return a list of all data owners in the system.

wronglogin: This file name suggests it might be used to store information about incorrect login attempts.

4.2 List of Libraries

Java Database Connectivity (JDBC):

JDBC is used to connect to a MySQL database. It provides a standard interface for Java applications to interact with relational databases.

Bouncy Castle (org.bouncycastle):

Bouncy Castle is a cryptographic library that provides a variety of cryptographic algorithms and protocols. In this context, it may be used for encryption and decryption operations related to security.

Apache Tomcat:

Tomcat is a servlet container and web server that implements the Java Servlet and JavaServer Pages (JSP) technologies. It provides a runtime environment for Java web applications.

Cufon (cufon-yui.js and cufon-aller.js):

Cufon is a font replacement technology that uses a combination of JavaScript and vector graphics to enable custom fonts on web pages. It enhances the visual appearance of text.

jQuery:

jQuery is a fast and lightweight JavaScript library. It simplifies HTML document traversal and manipulation, event handling, and animation. In the provided code, it is used for various UI-related tasks.

Java Cryptography Architecture (JCA):

JCA provides a framework for working with cryptography in Java. It includes APIs for key generation, encryption, decryption, and secure random number generation.

Java Naming and Directory Interface (JNDI):

JNDI is a Java API for directory services. It provides a way for Java software clients to discover and look up data and objects via a name.

Java Servlet API:

The Java Servlet API provides the means to define HTTP-specific servlet classes. Servlets are used to extend the capabilities of servers, such as Tomcat, to respond to network requests.

5. EXPERIMENTAL RESULT

5.1 Experiment setup

Setting up the experimental environment for a project involves preparing the necessary software, databases, and dependencies.

Prerequisites:

Java Development Kit (JDK):

Install the latest version of JDK on the server where your web application will run. You can download it from the official Oracle or OpenJDK website.

Apache Tomcat:

Download and install Apache Tomcat, which will serve as the servlet container for your web application. Configure Tomcat to work with the installed JDK.

MySQL Database:

Install and set up MySQL database server. Create a database named 'dtfu' and a user with appropriate privileges. Update the database connection details in the project's JDBC code.

Bouncy Castle Library:

Download the Bouncy Castle cryptographic library JAR files. Include these JARs in your project's classpath. You can often find them on the official Bouncy Castle website or through Maven repositories.

Web Application Files:

Deploy your web application files (JSP pages, HTML, CSS, JavaScript) into the appropriate directories in the Tomcat webapps folder.

Configuration:

Database Connection Configuration:

Update the database connection details in the JSP files where JDBC connections are established. This includes the URL, username, and password.

Tomcat Configuration:

Adjust Tomcat configuration files if necessary. Specifically, you may need to configure the context for your web application in the server.xml file.

Security Configuration:

Implement necessary security measures such as HTTPS, user authentication, and authorization based on the requirements of your project.

Running the Experiment:

Start MySQL Database:

Ensure that the MySQL database server is running.

Deploy Web Application:

Start the Tomcat server and deploy your web application by placing the WAR file or the project directory in the webapps folder.

Access the Application:

Open a web browser and navigate to the URL where Tomcat is running (e.g., http://localhost:8090/DataTrustFramework/). Access the different roles in the application (Owner, End User, Cloud Service Provider) and perform actions as specified in your project.

Monitoring and Logging:

Implement logging mechanisms to capture relevant events and errors. Monitor Tomcat logs, database logs, and application-specific logs to track the behavior of the system.

Performance Testing (Optional):

If performance evaluation is part of your experiment, consider using tools like Apache JMeter to simulate concurrent users and analyze the system's performance under different loads.

Security Auditing:

Perform security audits using tools like OWASP ZAP or other security scanners to identify vulnerabilities in your application.

5.2 Parameters with Formulas

Data Integrity:

Parameter: Hash Value (e.g., SHA-256)

Formula: hash_value = SHA-256(data)

Blockchain Transaction Validation:

Parameter: Transaction Validity

Formula: Validation rules based on blockchain consensus algorithms (e.g., Proof of Work, Proof of Stake)

Adaptive Transaction Validation:

Parameter: Adaptive Threshold

Formula: Adjusting validation requirements dynamically based on network conditions or security policies.

Security Audit Score:

Parameter: Audit Score

Formula: Calculate a score based on the results of security audits and checks.

Cloud Storage Resource Utilization:

Parameter: Storage Utilization

Formula: utilization_percentage = (used_storage / total_storage) * 100

Blockchain Network Performance:

Parameter: Transaction Throughput

Formula: throughput = (total_transactions / total_time)

Data Ownership Verification:

Parameter: Ownership Verification Score

Formula: Calculate a score based on the accuracy of data ownership verification mechanisms.

Blockchain Consensus Efficiency:

Parameter: Consensus Efficiency

Formula: efficiency = (successful_consensus / total_consensus_attempts) * 100

User Authentication Success Rate:

Parameter: Authentication Success Rate

Formula: success_rate = (successful_authentications / total_authentication_attempts) * 100

Adaptive Security Parameter (e.g., for dynamic adjustments based on threat levels):

Parameter: Security Threshold

Formula: Adjusting security parameters based on threat assessments and risk levels.

Transaction Latency:

Parameter: Latency

Formula: Calculate the time delay between transaction initiation and its confirmation on the blockchain.

Blockchain Block Size:

Parameter: Block Size

Formula: The size of each block in the blockchain, measured in bytes.

Resource Availability in Cloud Storage:

Parameter: Resource Availability

Formula: Calculate the availability of cloud storage resources over a specific period.

Data Deletion Verification:

Parameter: Deletion Verification Score

Formula: Score based on the effectiveness of mechanisms ensuring secure data deletion.

6.3 Sample Code

Connect:

```
<title>Block chain based management for organ donation and transplantation

<%@ page import="java.sql.*"%>

<%@ page import="java.util.*" %>

<%

        Connection connection = null;

        try {

        Class.forName("com.mysql.jdbc.Driver");
        Connection=DriverManager.getConnection("jdbc:mysql://localhost:3306/organ
_donation","root","root");

}

        catch(Exception e)

        {

                System.out.println(e);

        }

%>
```

Owner:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN""http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml">

<head>

<title>Hospital Main Page</title>

<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
```

```
<link href="css/style.css" rel="stylesheet" type="text/css" />
<link rel="stylesheet" type="text/css" href="css/coin-slider.css" />
<script type="text/javascript" src="js/cufon-yui.js"></script>
<script type="text/javascript" src="js/cufon-aller.js"></script>
<script type="text/javascript" src="js/jquery-1.4.2.min.js"></script>
<script type="text/javascript" src="js/script.js"></script>
<script type="text/javascript" src="js/coin-slider.min.js"></script>
<style type="text/css">
<!--
.style1 {font-size: 14px}
.style2 {font-size: 24px}
-->
</style>
</head>
<body>
<div class="main">
  <div class="header">
    <div class="header_resize">
      <div class="menu_nav">
        <ul>
          <li class="active"><a href="index.html"><span>Home Page</span></a></li>
                <li><a href="HospitalLogin.jsp">Hospital</a></li>
                <strong></strong>
```

```
        <li><a href="CompanyLogin.jsp">Donor</a></li>
        <li><a href="UserLogin.jsp"><span>Patients</span></a></li>
      </ul>
    </div>
    <div class="logo">
      <h1 class="style1"><a href="index.html" class="style2">Blockchain Based Management for Organ Donation and Transplantation</a></h1>
    </div>
    <div class="clr"></div>
    <div class="slider">
      <div id="coin-slider"> <a href="#"><img src="images/slide1.jpg" width="970" height="305" alt="" /> </a> <a href="#"><img src="images/slide2.jpg" width="970" height="305" alt="" /> </a> <a href="#"><img src="images/slide3.jpg" width="970" height="305" alt="" /> </a> </div>
    </div>
    <div class="clr"></div>
  </div>
 </div>
 <div class="content">
  <div class="content_resize">
   <div class="mainbar">
    <div class="article">
     <h2>Welcome To Hospital Main..!</h2>
     <p class="infopost"> </p>
```

```
<div class="clr"></div>
<div class="post_content">
<p> </p>
<p><img src="images/Hospital.jpg" width="620" height="487" /></p>
</div>
<div class="clr"></div>
</div>
</div>
<div class="sidebar">
<div class="searchform">
<form id="formsearch" name="formsearch" method="post" action="#">
<span>
<input name="editbox_search" class="editbox_search" id="editbox_search" maxlength="80" value="Search our ste:" type="text" />
</span>
<input name="button_search" src="images/search.gif" class="button_search" type="image" />
</form>
</div>
<div class="clr"></div>
<div class="gadget">
<h2 class="star"><span>Hospital</span> Menu</h2>
```

```
<div class="clr"></div>

<ul class="sb_menu">

  <li><strong><a
href="Hospital_Main.jsp">Home</a></strong></li>

  <li><strong><a href="A_AuthorizeUsers.jsp">View all Patients
and Authorize</a></strong></li>

        <li><strong><a
href="A_AuthorizeDonorUsers.jsp">View all Donors and
Authorize</a></strong></li>

        <li><strong><a href="A_Organ_Type.jsp">Add Organ
Type</a></strong></li>

        <li><strong><a
href="A_View_All_Organ_Names.jsp">View All Blockchain Hash Code
for Organ Names</a></strong></li>

        <li><strong><a
href="A_View_All_Organ_Donated_Details.jsp">View All Organ
Donated Details</a></strong></li>

        <li><strong><a
href="A_View_All_Patient_Transplantation_Requested_Details.jsp">Vie
w All Patient Transplantation Requested Details</a></strong></li>

        <li><strong><a
href="A_View_All_Organ_Donated_Details_By_Blockchain.jsp">View
All Organ Donated Details By Blockchain</a></strong></li>

        <li><strong><a
href="A_View_All_Organ_Transplantation_Details_By_Blockchain.jsp"
>View All Organ Transplantation Details By
Blockchain</a></strong></li>
```

```
                    <li><strong><a href="A_View_Results.jsp">View All Organ Donation Results</a></strong></li>

                    <li><strong><a href="A_View_Results1.jsp">View Organ Transplantation Results </a></strong></li><li><strong><a href="HospitalLogin.jsp">Logout</a></strong></li>

          </ul>

        </div>

      </div>

      <div class="clr"></div>

    </div>

  </div>

  <div class="fbg">

    <div class="fbg_resize">

      <div class="clr"></div>

    </div>

  </div>

  <div class="footer">

    <div class="footer_resize">

      <div style="clear:both;"></div>

    </div>

  </div>

</div>

<div align=center></a></div></body>

</html>
```

Enduser:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml">

<head>

<title>Donor User Login</title>

<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />

<link href="css/style.css" rel="stylesheet" type="text/css" />

<link rel="stylesheet" type="text/css" href="css/coin-slider.css" />

<script type="text/javascript" src="js/cufon-yui.js"></script>

<script type="text/javascript" src="js/cufon-aller.js"></script>

<script type="text/javascript" src="js/jquery-1.4.2.min.js"></script>

<script type="text/javascript" src="js/script.js"></script>

<script type="text/javascript" src="js/coin-slider.min.js"></script>

<style type="text/css">

<!--

.style1 {font-size: 14px}

.style2 {font-size: 24px}

.style4 {color: #000000}

-->

</style>

</head>

<body>
```

```
<div class="main">
  <div class="header">
    <div class="header_resize">
      <div class="menu_nav">
        <ul>
          <li class="active"><a href="index.html"><span>Home Page</span></a></li>
                <li><a href="HospitalLogin.jsp">Hospital</a></li>
                <strong></strong>
          <li><a href="DonorLogin.jsp">Donor</a></li>
          <li><a href="UserLogin.jsp"><span>Patients</span></a></li>
        </ul>
      </div>
      <div class="logo">
        <h1 class="style1"><a href="index.html" class="style2">Blockchain Based Management for Organ Donation and Transplantation</a></h1>
      </div>
      <div class="clr"></div>
      <div class="slider">
        <div id="coin-slider"> <a href="#"><img src="images/slide1.jpg" width="970" height="305" alt="" /> </a> <a href="#"><img src="images/slide2.jpg" width="970" height="305" alt="" /> </a> <a href="#"><img src="images/slide3.jpg" width="970" height="305" alt="" /> </a> </div>
      </div>
```

```
    <div class="clr"></div>
   </div>
  </div>
  <div class="content">
   <div class="content_resize">
    <div class="mainbar">
     <div class="article">
      <h2>Welcome To Donor User Login..!</h2>
      <p align="center" class="infopost"><img src="images/Login.jpg" width="222" height="222" /></p>
      <div class="clr"></div>
            <div class="post_content">
            <form id="form1" name="form1" method="post" action="Authentication.jsp?type=<%="cuser"%>">
<table width="464" border="0" cellspacing="2" cellpadding="2"
<tr>
<td width="200" height="46" align="center"><div align="justify"><span class="style34"><label for="name"><span class="style4">Donor Username (required)</span></label></span></div> </td>
<td width="253"><input id="name" name="cuserid" class="text" /></td>
</tr>
 <tr>
  <td height="30" align="center"><div align="justify"><span class="style34 style4"><label for="pass">Password (required)</label>
            </span></div></td>
```

```
                <td><input type="password" id="pass" name="pass" class="text" /></td>
                </tr>
                <tr>
                  <td> </td>
                  <td> </td>
                </tr>
                <tr>
                  <td> </td>
                  <td><span class="style16">
                    <input name="imageField" type="submit" class="LOGIN" id="imageField" value="Login" />
                    <strong> New User?</strong></span><a href="DonorUserRegister.jsp" class="style30"> Register </a></td>
                </tr>
                <tr>
                  <td> </td>
                  <td> </td>
                </tr>
              </table>
              </form>
                </div>
        <div class="clr"></div>
      </div>
    </div>
```

```
<div class="sidebar">
 <div class="searchform">
  <form id="formsearch" name="formsearch" method="post" action="#">
   <span>
   <input name="editbox_search" class="editbox_search" id="editbox_search" maxlength="80" value="Search our ste:" type="text" />
   </span>
   <input name="button_search" src="images/search.gif" class="button_search" type="image" />
  </form>
 </div>
 <div class="clr"></div>
 <div class="gadget">
  <h2 class="star"><span>Sidebar</span> Menu</h2>
  <div class="clr"></div>
  <ul class="sb_menu">
   <li><a href="index.html">Home</a></li>
   <li><a href="#"></a></li>
  </ul>
 </div>
</div>
<div class="clr"></div>
</div>
```

```
  </div>
  <div class="fbg">
    <div class="fbg_resize">
      <div class="clr"></div>
    </div>
  </div>
  <div class="footer">
    <div class="footer_resize">
      <div style="clear:both;"></div>
    </div>
  </div>
</div>
<div align=center></a></div></body>
</html>
```

Authentication:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<title>Authentication Page</title>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<link href="css/style.css" rel="stylesheet" type="text/css" />
<link rel="stylesheet" type="text/css" href="css/coin-slider.css" />
```

```
<script type="text/javascript" src="js/cufon-yui.js"></script>
<script type="text/javascript" src="js/cufon-aller.js"></script>
<script type="text/javascript" src="js/jquery-1.4.2.min.js"></script>
<script type="text/javascript" src="js/script.js"></script>
<script type="text/javascript" src="js/coin-slider.min.js"></script>
<style type="text/css">
<!--
.style1 {font-size: 14px}
.style2 {font-size: 24px}
.style3 {
        color: #FF0000;
        font-weight: bold;
}
-->
</style>
</head>
<body>
<div class="main">
  <div class="header">
    <div class="header_resize">
      <div class="menu_nav">
        <ul>
          <li class="active"><a href="index.html"><span>Home
Page</span></a></li>
```

```
                <li><a href="HospitalLogin.jsp">Hospital</a></li>
                <strong></strong>
          <li><a href="CompanyLogin.jsp">Donor</a></li>
          <li><a href="UserLogin.jsp"><span>Patients</span></a></li>
        </ul>
      </div>
      <div class="logo">
        <h1 class="style1"><a href="index.html" class="style2">Blockchain Based Management for Organ Donation and Transplantation</a></h1>
      </div>
      <div class="clr"></div>
      <div class="slider">
        <div id="coin-slider"> <a href="#"><img src="images/slide1.jpg" width="970" height="305" alt="" /> </a> <a href="#"><img src="images/slide2.jpg" width="970" height="305" alt="" /> </a> <a href="#"><img src="images/slide3.jpg" width="970" height="305" alt="" /> </a> </div>
      </div>
      <div class="clr"></div>
    </div>
  </div>
  <div class="content">
    <div class="content_resize">
      <div class="mainbar">
        <div class="article">
```

```
        <h2>Authentication..!</h2>
        <p class="infopost"> </p>
        <div class="clr"></div>

        <div class="post_content">
          <%@ include file="connect.jsp" %>
<%
        String type=request.getParameter("type");
    try{
            if(type.equalsIgnoreCase("Hospital"))
{
        String Hospitalname=request.getParameter("Hospitalid");
        String Password=request.getParameter("pass");
        application.setAttribute("Hospital",Hospitalname);
        String sql="SELECT * FROM Hospital where Hospitalname='"+Hospitalname+"' and password='"+Password+"'";
  Statement stmt = connection.createStatement();
   ResultSet rs =stmt.executeQuery(sql);
        if(rs.next())
                        {
                                response.sendRedirect("Hospital_Main.jsp");
                        }
                        else
                        {
```

```
                response.sendRedirect("HospitalRe-Login.jsp");
            }
        }
         else if(type.equalsIgnoreCase("user"))
        {
            String username=request.getParameter("userid");
        String Password=request.getParameter("pass");
            application.setAttribute("user",username);
          String sql="SELECT * FROM user where username='"+username+"' and password='"+Password+"'";
            Statement stmt = connection.createStatement();
            ResultSet rs =stmt.executeQuery(sql);
           if(rs.next())
            {
                String sql1="SELECT * FROM user where username='"+username+"' and status='Authorized'";
                Statement stmt1 = connection.createStatement();
                ResultSet rs1 =stmt1.executeQuery(sql1);

                if(rs1.next())
              {
                response.sendRedirect("User_Main.jsp");
                }
                Else
```

```
{
  %>
 <br/><h3><p align="left" class="style3"> </p><p align="left" class="style4" style="color:#000000">You are not the Authorized User, Please wait!! </p></h3><br/><br/><a href="UserLogin.jsp">Back</a>
<%
}
}
 else
{
        response.sendRedirect("UserRe-Login.jsp");
}
                }
                else if(type.equalsIgnoreCase("cuser"))
{

 String company=request.getParameter("company");
 String cusername=request.getParameter("cuserid");
 String Password=request.getParameter("pass");
  application.setAttribute("cname",company);
  application.setAttribute("cuser",cusername);
String sql="SELECT * FROM cuser where (cusername='"+cusername+"' and password='"+Password+"') ";
Statement stmt = connection.createStatement();
ResultSet rs =stmt.executeQuery(sql);
```

```
                    if(rs.next())
                    {
                    String sql1="SELECT * FROM cuser where cusername='"+cusername+"' and status='Authorized'";
Statementmn stmt1 = connection.createStatement();
ResultSet rs1 =stmt1.executeQuery(sql1);
if(rs1.next())
{

        response.sendRedirect("DonorUser_Main.jsp");
}
else
{
   %>
  <br/><h3><p align="left" class="style3"> </p>
   <p align="left" class="style4" style="color:#000000">You are not the Authorized User, Please wait!! </p></h3>
   <br/><br/><a href="DonorLogin.jsp">Back</a>
   <%
     }
}
else
   {
```

```
                    response.sendRedirect("DonorUserRe-Login.jsp");
                }
            }
            else{

                }
        }
        catch(Exception e)
        {
            out.print(e);
        }
%>
        </div>
        <div class="clr"></div>
      </div>
    </div>
    <div class="sidebar">
      <div class="searchform">
        <form id="formsearch" name="formsearch" method="post" action="#">
          <span>
          <input name="editbox_search" class="editbox_search" id="editbox_search" maxlength="80" value="Search our ste:" type="text" />
          </span>
```

```
        <input name="button_search" src="images/search.gif"
class="button_search" type="image" />
      </form>
    </div>
    <div class="clr"></div>
    <div class="gadget">
      <h2 class="star"><span>Sidebar</span> Menu</h2>
      <div class="clr"></div>
      <ul class="sb_menu">
        <li><a href="index.html">Home</a></li>
        <li><a href="#"></a></li>
      </ul>
    </div>
  </div>
  <div class="clr"></div>
 </div>
</div>
<div class="fbg">
 <div class="fbg_resize">
  <div class="clr"></div>
 </div>
</div>
<div class="footer">
 <div class="footer_resize">
```

```
      <div style="clear:both;"></div>
    </div>
  </div>
</div>
<div align=center></a></div></body>
</html>
```

6.Testcases

Table 6.1 Test cases for the proposed system

Test case ID	**INPUT**	**Expected Output**	**Actual Output**	**Rate**
1.	Uploading data	Data encrypted successfully and Data uploaded successfully.	Data uploaded successfully.	success
2.	Requesting data	Data Request sent to authentication successfully.	Data Request sent to authentication successfully.	success
3.	Authentication provides response to the requested data.	Responded successfully.	Responded successfully.	success
4.	Creating Hash code for uploaded and requested data	Data viewed in code format successfully	Data viewed in code format successfully	success
5.	The authenticated block can change the state by owner successfully	Changed successfully.	Changed successfully.	success

7.Screenshots

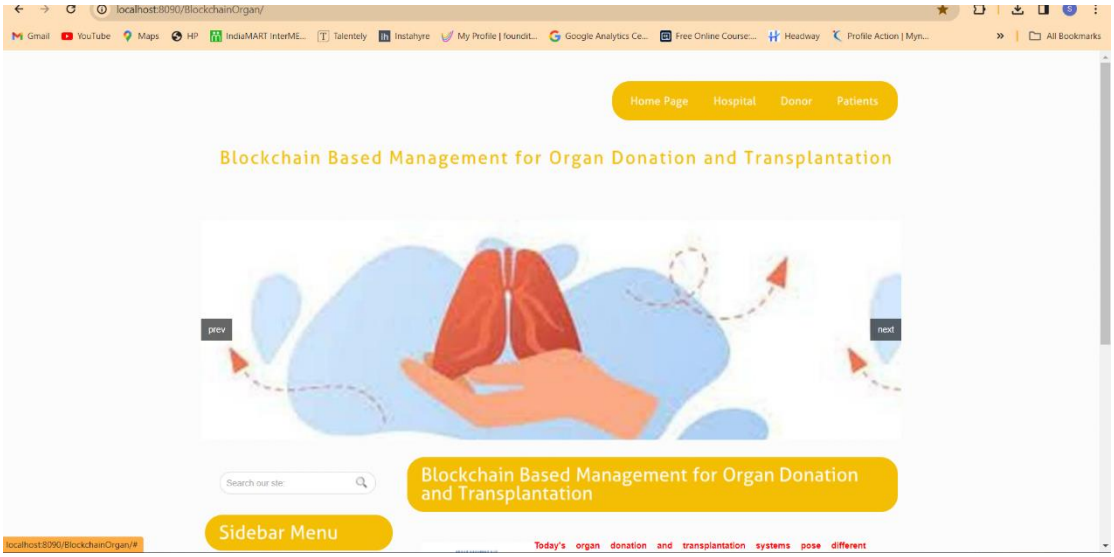

Figure 7.1: A Trustworthy Data Sharing Framework Home Page

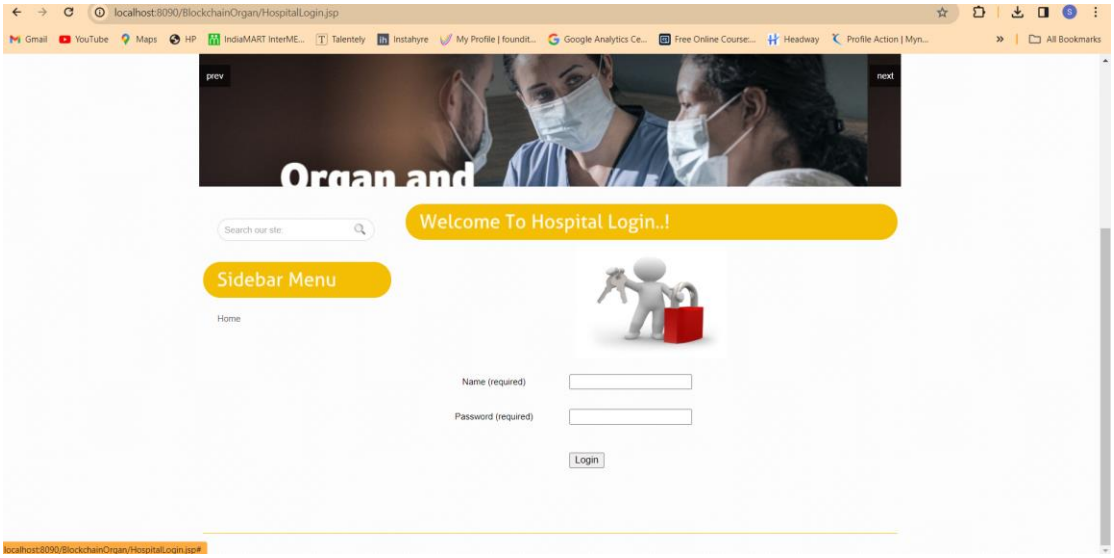

Figure 7.2 showcases a data trust framework login page integrating blockchain technology, ensuring secure and transparent data transactions.

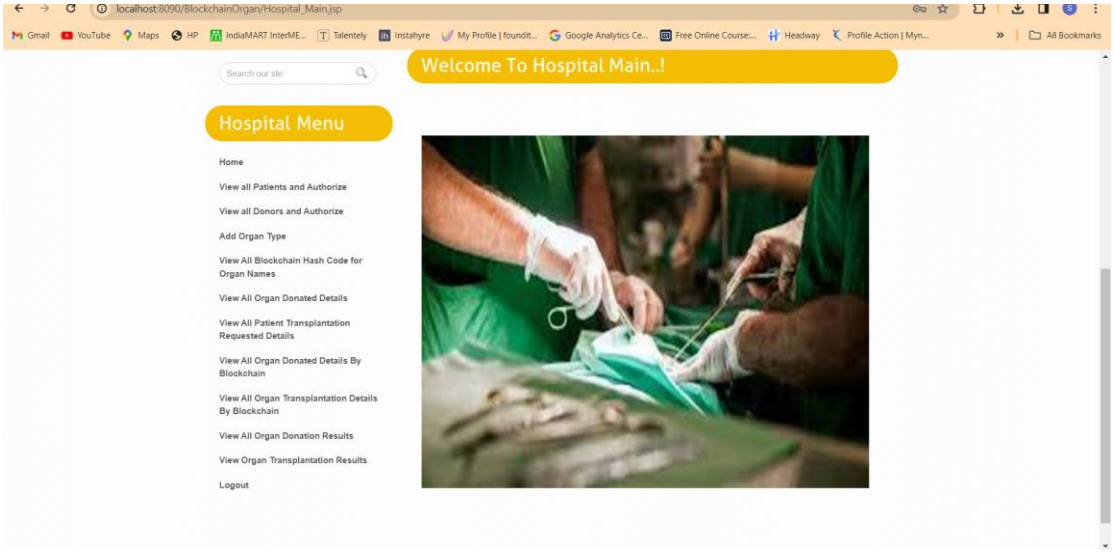

Figure 7.3 illustrates the owner main page, providing a secure gateway for authenticated access to the data trust framework.

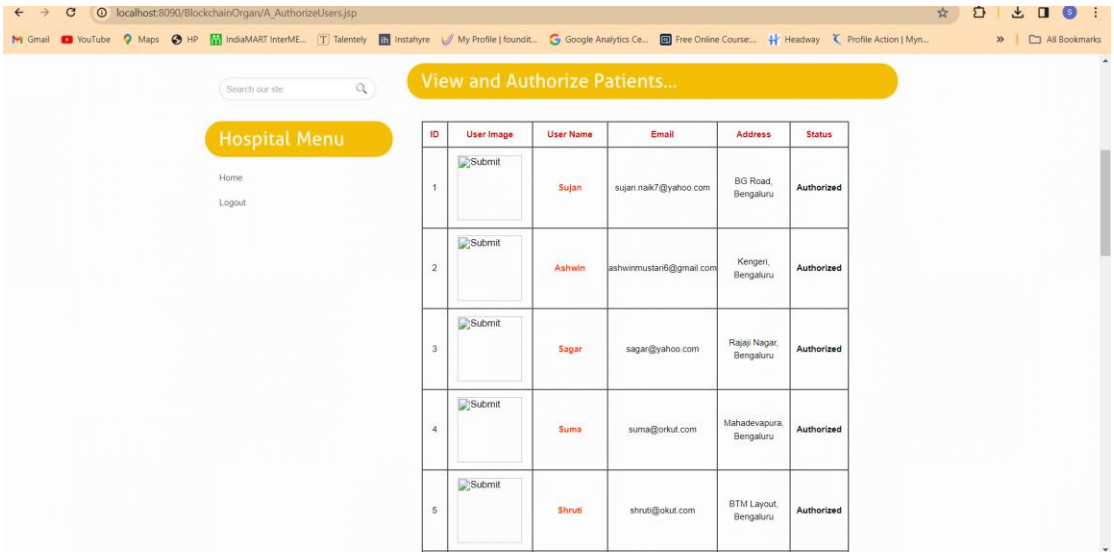

Figure 7.4 illustrates the owner viewing the registered patients page which is going to update the status from waiting to authorization.

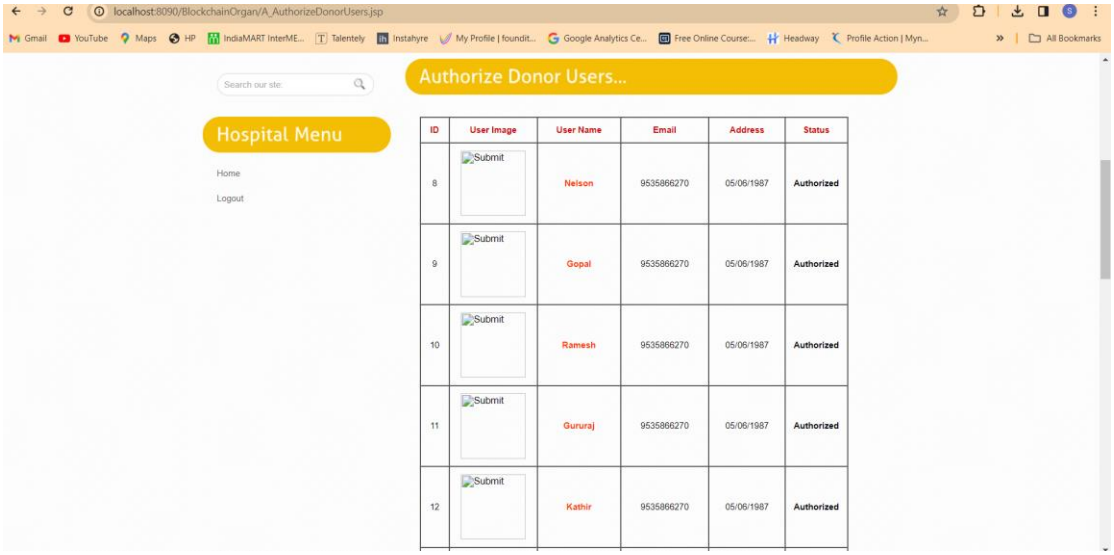

Figure 7.5 illustrates the owner viewing the registered donors page which is going to update the status from waiting to authorization.

Figure 7.6 illustrates the owner to make an add on feature to the organ donation and transplantations.

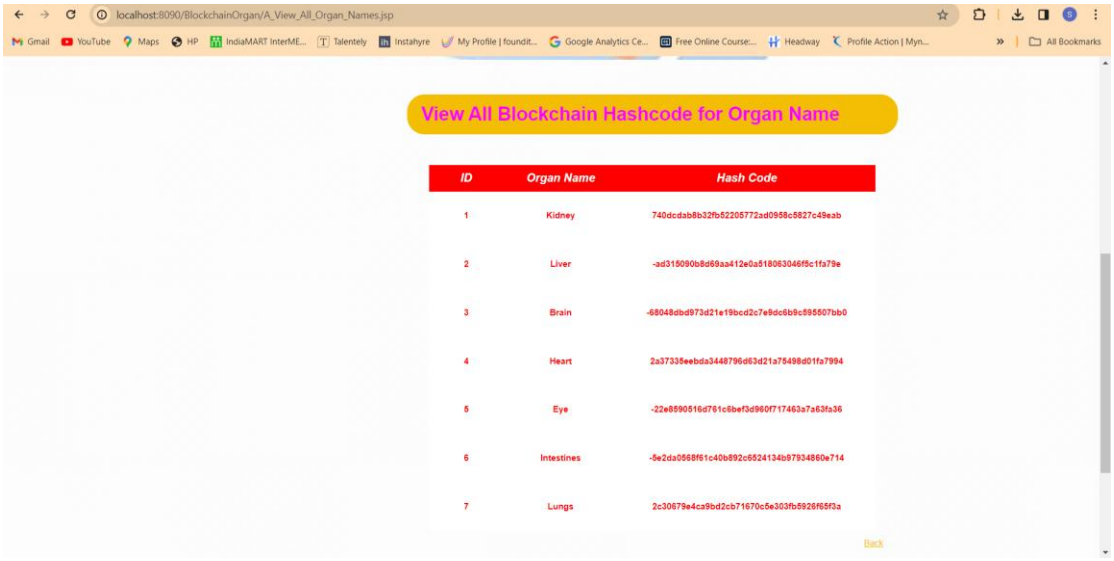

Figure 7.7 Creating an hash code for the registered organs in the list.

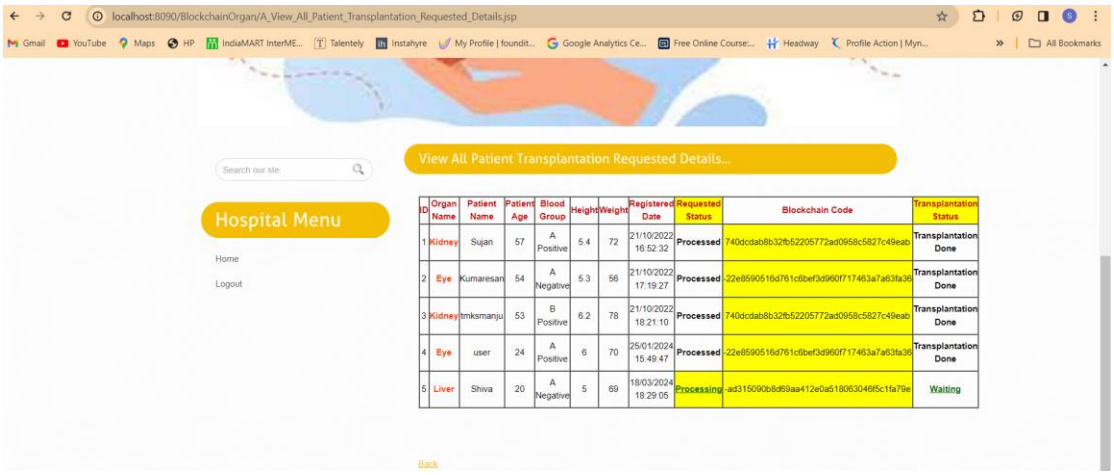

Figure 7.8 owners data, where all the patients and donors with their registered data for updating a successful transplantation.

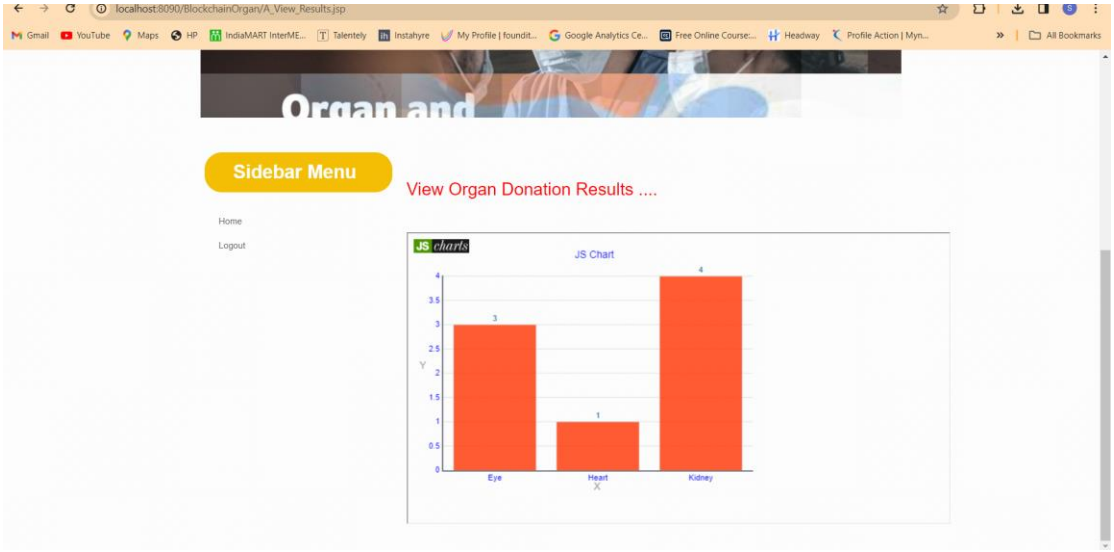

Figure 7.9 showcases the result of the donation, offering a centralized platform for managing and facilitating secure data transactions within the data trust framework.

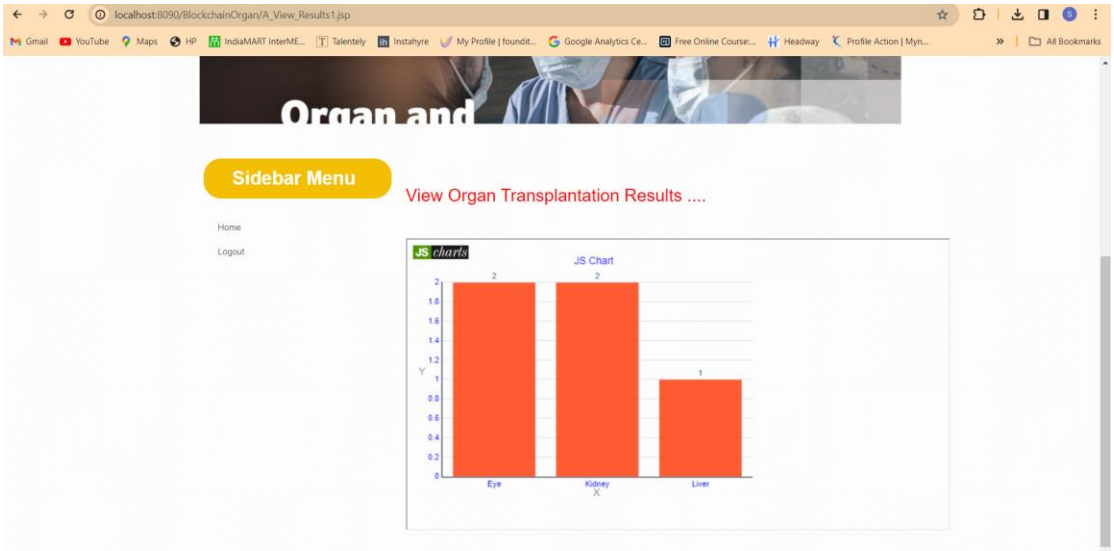

Figure 7.10 showcases the result of the transplantation, offering a centralized platform for managing and facilitating secure data transactions within the data trust framework.

8.Conclusion

In this paper, we have proposed a private Ethereum blockchain-based solution that manages organ donation and transplantation in a decentralized, accountable, auditable, traceable, secure, and trustworthy manner. We developed smart contracts that ensure the data provenance by recording events automatically. We present algorithms with their implementation, testing, and validation details. We analyze the security of the proposed solution to guarantee that smart contracts are protected against common attacks and vulnerabilities. We compare our solution to other blockchain-based solutions that are currently available. We discuss how our solution can be customized with minimal effort to meet the needs of other systems experiencing similar problems. In the future, our solution can be improved by developing an end-to end DApp. Furthermore, the smart contracts can be deployed and tested on a real private Ethereum network. Finally, the Quorum platform can provide better confidentiality because transactions among entities can only be viewed by specific participants and nobody else, which is not the case in our solution, where transactions between two participants are viewed by other actors authorized in the private blockchain.

9.Future Enhancement

Future enhancements of a Blockchain Based Management for Organ Donation and Transplantation and adaptive transaction validation can focus on several key areas to further improve the system:

Integration with Emerging Technologies: Explore how the proposed solution could integrate with emerging technologies such as AI and IoT for improved data analysis, real-time monitoring of organ transportation, and predictive modeling for better donor-recipient matching.

Scalability and Interoperability: Discuss strategies to enhance the scalability of the blockchain network to handle a larger volume of transactions and ensure interoperability with other healthcare systems and databases for seamless data exchange.

Enhanced Privacy and Data Security: Develop advanced encryption techniques and privacy-preserving algorithms to enhance the privacy of patient data and strengthen the security measures against potential cyber threats and data breaches.

Regulatory Compliance: Address the regulatory compliance aspects by incorporating features that ensure adherence to legal requirements, ethical standards, and healthcare regulations governing organ donation and transplantation.

User Experience Optimization: Implement user-friendly interfaces, mobile applications, and personalized dashboards for donors, recipients, healthcare providers, and administrators to enhance their experience and facilitate efficient communication and decision-making.

Community Engagement and Education: Integrate community engagement initiatives and educational resources within the platform to raise awareness about organ donation, transplantation processes, and ethical considerations, fostering a more informed and supportive ecosystem.

10. BIBLIOGRAPHY

[1] A. Powell. (Mar. 18, 2019). **A Transplant Makes History. Harvard Gazette.** https://news.harvard.edu/gazette/story/2011/09/atransplantmakes-history/

[2] Organ Donation Facts and Info: **Organ Transplants. Accessed:** Apr. 18, 2021 https://my.clevelandclinic.org/health/articles/11750-organ-donation-and-transplantation

[3] (Mar. 21, 2019). **Facts and Myths About Transplant**. Accessed: Apr. 21, 2021 https://www.americantransplantfoundation.org/about-transplant/facts-and-myths/

[4] Organ Procurement and Transplantation Network. Accessed: Apr. 18, 2021 https://optn.transplant.hrsa.gov/resources/ethics/ethical-principles-in-the-allocation-of-humanorgans/

[5] How Donation Works. Accessed: Jan. 7, 2022 https://www.organdonor.gov/learn/process

[6] UFO Themes. (Aug. 1, 2017). **Organ Donation and Transplantation in Germany. Plastic Surgery Key** https://plasticsurgerykey.com/organ-donation-and-transplantation-in-germany/