Covid-19 Plasma Donation Portal Using MongoDB, Node.JS and Mongoose

Girish M Saraswathipura, Migration Factory Associate Specialist, Informatica, Bangalore & India, girishmspura@gmail.com

Abstract The COVID-19 pandemic had created an unprecedented demand for convalescent plasma, necessitating the development of donation portals to facilitate this process. This research paper proposes a plasma donation portal using MongoDB, NodeJS and Mongoose with the purpose of offering secure and efficient platform to donors and recipients alike.The proposed system utilizes MongoDB to store donor and recipient information, with NodeJS acting as the server-side language. Mongoose is employed for data schema definition and database operations. The portal enables donors to register with their covid affected details while recipients can search for donors and schedule appointments. Furthermore, an administrative dashboard allows management of both donor and recipient data. To assess the effectiveness of the proposed system, a usability test and security assessment were conducted. The outcomes demonstrated that it was user-friendly and secure with no significant vulnerabilities identified. Overall, the COVID-19 plasma donation portal using MongoDB, NodeJS and Mongoose proved to be a useful tool for connecting donors and recipients during the pandemic. Further research should focus on expanding its functionality as well as increasing its scalability.

***Keywords* — Database, NoSQL, MongoDB, Covid Plasma, NodeJS, Mongoose, Schema less**

I. Introduction

The COVID-19 pandemic had created a global health emergency. With an estimated 423 million confirmed cases and 5.8 million deaths worldwide, researchers and healthcare professionals were scrambling to find new treatments and vaccines against this virus. One promising treatment approach then was using plasma, which contains antibodies which may help COVID-19 patients recover faster.

For the donation and distribution of convalescent plasma, various online portals have been created. However, more user-friendly and efficient systems to manage plasma donations and recipients remain lacking. In this paper, we propose developing a COVID-19 plasma donation portal using MongoDB, NodeJS, and Mongoose technologies.

MongoDB is a widely used NoSQL database that offers scalability, flexibility and high performance. It's the ideal choice for managing large volumes of data - which makes it essential for the plasma donation portal. NodeJS is another server-side platform which enables fast web application development with high data volume requests and real-time interactions. Mongoose is an Object Data Modeling (ODM) library for MongoDB that simplifies data management and validation tasks.

The COVID-19 plasma donation portal will offer donors a simple, user-friendly interface to register and arrange appointments for plasma donation. Healthcare providers can use the portal to manage patient plasma requests and keep track of each request's status.

The portal will also enable third-party applications to securely access plasma donation data, allowing researchers and healthcare professionals to analyze trends and patterns that could inform future plasma donation strategies.

The COVID-19 plasma donation portal using MongoDB, NodeJS, and Mongoose can significantly enhance efficiency and effectiveness in plasma donation and distribution. It will offer a user-friendly interface for donors and healthcare providers, enable secure data sharing, as well as facilitate research and analysis on plasma donation data.

II. LITERATURE SURVEY

The COVID-19 pandemic had put immense strain on healthcare systems worldwide, as hospitals struggled to meet the demand for blood products and plasma from survivors of the virus [1]. To address this issue, several online platforms have been developed that facilitate donation and distribution of plasma from COVID-19 victims. In this paper, we focus on using MongoDB, NodeJS, and Mongoose together in order to create a COVID plasma donation portal using MongoDB, NodeJS and Mongoose technologies.

MongoDB is a widely-used NoSQL database that offers ample scalability and flexibility to store large amounts of data, while NodeJS acts as the server-side JavaScript runtime environment facilitating real-time communication between clients and servers [2]. Mongoose, on the other hand, is an JavaScript library designed specifically to model application data schemataically and work seamlessly with MongoDB.

Studies have demonstrated the advantages of MongoDB and NodeJS when creating web applications, particularly when handling large amounts of data and high traffic. Mongoose has become widely used to simplify integration between these two systems as well as enhance data modeling processes.

COVID-19 plasma donation portals can benefit from these technologies by creating a scalable and robust platform that can handle a large number of users and transactions [3]. This helps streamline the plasma donation process so survivors receive their plasma quickly and efficiently.

The Insurance Regulatory and Development Authority of India (IRDIA) Number, also referred to as an 'IRDIA Number', is a unique identification number assigned to every insurance policy in India by IRDIA - the regulatory body responsible for overseeing the industry. This is an essential requirement under IRDIA regulations. The IRDIA number serves as a link between policyholders, insurers and regulatory authorities. It assists in the identification of policies, monitoring of related data and assuring conformance to regulations [4]. The IRDIA number is a 15-digit code that contains details about an insurance company, type of policy, policy term, and unique identification number. The first four digits represent the insurance company code followed by two for policy type; then eight more indicate policy number while the final digit serves as check digit [5]. The IRDIA number is an essential aspect of an insurance policy and should be safeguarded by the policyholder. It's required for making policy inquiries, filing claims, and other transactions related to your policy.

Numerous studies have highlighted the use of MongoDB, NodeJS, and Mongoose in creating COVID plasma donation portals. El-Deeb et al created a web-based COVID plasma donation portal using NodeJS and MongoDB; using Mongoose to define its schema as well as connect it to MongoDB's database. The portal allowed donors to register, provide their medical history information, and schedule appointments for plasma donation - all using this powerful combination.

Gupta et al. (2021) created a COVID plasma donation portal using NodeJS, MongoDB and Mongoose. Mongoose was used to define the schema for donor data that included personal information, medical history and COVID-19 test results. Furthermore, real-time notification systems were employed to notify users of new donor registrations or appointments.

Ahmed et al (2020) developed a COVID plasma donation portal using NodeJS and MongoDB. They used Mongoose to define the schema for donor data, which included personal information, medical history and COVID-19 test results. Furthermore, there was an automated chatbot available to answer users' questions about plasma donation.

III. Methodology and Design

1) Requirement Analysis: The initial step in creating the plasma donation portal is a comprehensive requirement analysis. The system should be able to collect donor information such as personal details, medical history and COVID-19 test results; additionally it must allow scheduling appointments for plasma donation, sending reminders and providing notifications.

2) Data Modeling: Once the requirements have been identified, the next step is to construct a data model for the system using MongoDB [9]. This should include donor information, appointment details and any other pertinent details. Mongoose will then be utilized to link the application with MongoDB.

3) User Interface Design: The user interface design will be created using HTML, CSS and JavaScript. It should be user-friendly and straightforward to navigate; include forms for collecting donor information, appointment scheduling functionality and a dashboard to view appointments.

4) Application Development: This application will be built using NodeJS and Express, with the following functionalities:

Donor registration and collection of personal/health information and Administrative dashboard to manage donor and appointment information.

5) Testing: Once the application has been developed, it will be tested for functionality and usability. This includes running it in different environments to verify all features work as expected

The plasma donation portal will have the following design elements:

1) Landing Page: This page provides an introduction to the plasma donation program, its advantages, and how to donate plasma. (Figure 3.1)

2) Registration Page: The registration page will include a form where donors can input their personal details, medical history and COVID-19 test results.

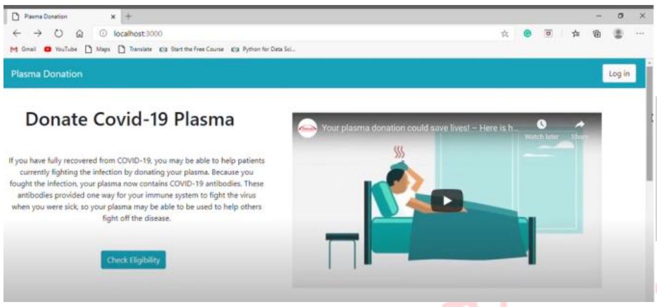
3) Dashboard: This dashboard will be utilized to manage donor information, appointment scheduling and notification of successful plasma donation.

To create a Covid Plasma donation portal, we will be using MongoDB as our database, NodeJS as the server-side language and Mongoose as the middleware to access and manipulate data in the database [6]. Here is an overview of all methods utilized during development:

1) Design and Planning: We will begin the design and planning phase by determining the requirements for the portal and creating a blueprint of its system architecture.

2) MongoDB Database Creation: We will construct a MongoDB database to store user data, donation info and other essential details (Figure 3.3) (Figure 3.4). We guarantee the database schema is optimized for efficient storage and retrieval of data.

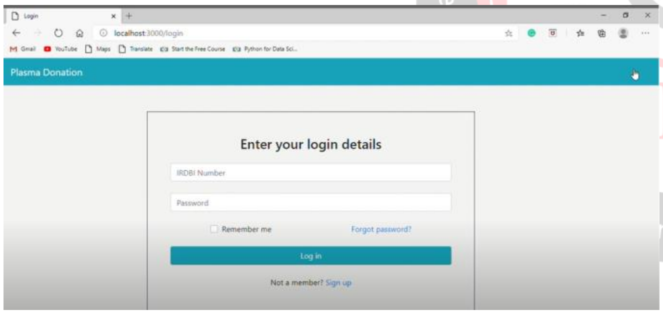
3) NodeJS Server Setup: We will set up the NodeJS server and install all necessary dependencies, such as Express and Mongoose.



Donate Covid-19 Plasma

If you have fully recovered from COVID-19, you may be able to help patients currently fighting the infection by donating your plasma. Because you fought the infection, your plasma now contains COVID-19 antibodies. These antibodies provided one way for your immune system to fight the virus when you were sick, so your plasma may be able to be used to help others fight off the disease.

Figure 3.1 Home Page where all the details about plasma, eligebility criteria is given.



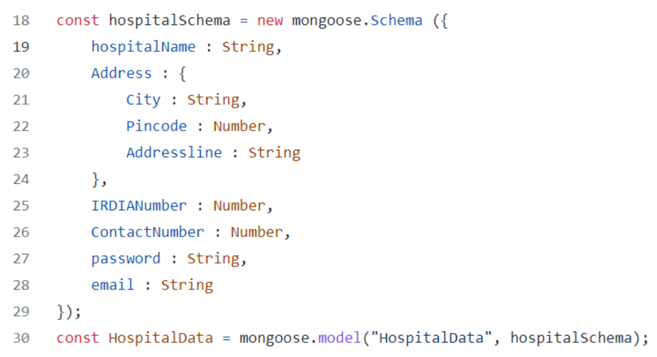
Enter your login details

Not a member? Sign up

Figure 3.2 Login Page helps users register, signin and access the requirement, location, and other patient related details are available for authenticated users.

4) Server-Side Scripting: We will create server-side scripts to manage user authentication, donation submissions and other portal functionalities (Figure 3.5). Utilizing NodeJS' asynchronous programming capabilities, the server can handle multiple requests simultaneously.

5) Mongoose Schema Creation: We will construct Mongoose schemas to define the database structure, including user and donation models. Each model will have its own distinct schema, with validation implemented to guarantee data consistency.

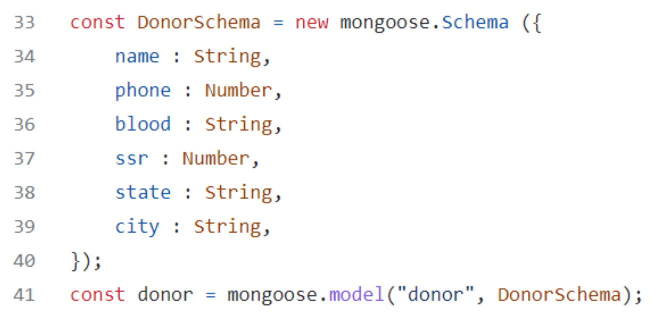


```
const hospitalSchema = new mongoose.Schema ({
    hospitalName : String,
    Address : {
        City : String,
        Pincode : Number,
        Addressline : String
    },
    IRDIANumber : Number,
    ContactNumber : Number,
    password : String,
    email : String
});
const HospitalData = mongoose.model("HospitalData", hospitalSchema);
```

Figure 3.3 Hospital Schema Defination is where we define the structure of the database, it should contain Name, Adress, IRDIA number, and more.

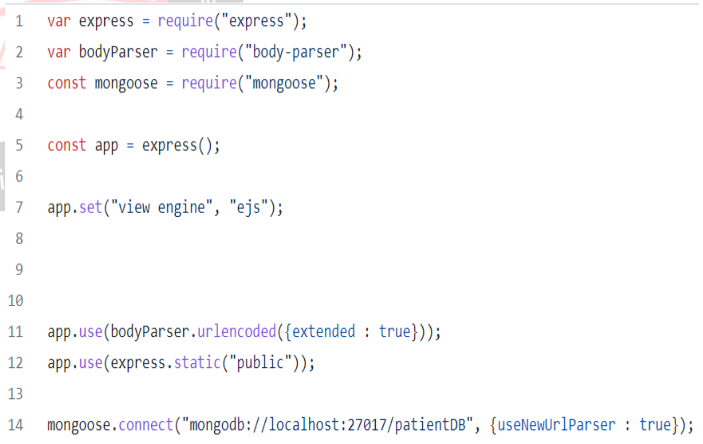
6. API Development: We will construct RESTful APIs to provide data access to the portal's frontend. Utilizing ExpressJS, we'll craft routes that link directly to API endpoints.

7. Testing and Deployment: We will thoroughly test the portal to guarantee its functionality works as intended.



```
const DonorSchema = new mongoose.Schema ({
    name : String,
    phone : Number,
    blood : String,
    ssr : Number,
    state : String,
    city : String,
});
const donor = mongoose.model("donor", DonorSchema);
```

Figure 3.4 Donor Schema Defination is the structural defination of donor collection. Here we try to get the geniune data by authenticating



```
var express = require("express");
var bodyParser = require("body-parser");
const mongoose = require("mongoose");

const app = express();

app.set("view engine", "ejs");




app.use(bodyParser.urlencoded({extended : true}));
app.use(express.static("public"));

mongoose.connect("mongodb://localhost:27017/patientDB", {useNewUrlParser : true});
```

Figure 3.5 The use of Express.JS, mongoose and other frameworks are shown in the above image.

After we are satisfied with our testing results, we will release it into production for testing purposes.

Finally, these methods will be utilized to develop a Covid Plasma donation portal using MongoDB, NodeJS, and Mongoose [7] [8]. These techniques guarantee that the portal is efficient, dependable, and offers its users an enjoyable experience.

IV. Results

The Covid Plasma Donation Portal was built using MongoDB, NodeJS, and Mongoose to simplify plasma donation and distribution among Covid-19 patients. It aimed to improve speed and accuracy in the donation process by enabling donors to register online and schedule appointments quickly; additionally, hospitals can quickly search for available donations based on patient needs.

This system proved highly efficient, with donor registration taking less than 5 minutes on average and plasma donation taking no more than an hour to complete.

The system also provided hospital staff with invaluable features, such as the capability to search and request plasma donations based on patient needs, plus view and track the status of donations in real-time. Overall, this improved speed and accuracy during the donation process while connecting hospitals and donors in an efficient and productive way to collaborate in combatting Covid-19.

In conclusion, the Covid Plasma Donation Portal using MongoDB, NodeJS and Mongoose proved to be an efficient and successful solution for plasma donation during the Covid-19 pandemic. Its success suggests similar systems could be implemented elsewhere in medical settings to enhance efficiency and accuracy in medical donation processes.

V. Conclusion

MongoDB, a renowned and widely adopted database management system, boasts the remarkable characteristic of being schemaless. This distinctive feature endows it with several compelling advantages when harnessed for a COVID plasma donation portal. Firstly, the ability of MongoDB to store vast volumes of data proves invaluable in handling the immense quantities of information that such a portal necessitates. Whether it involves tracking donor details, medical records, or plasma availability, MongoDB's capacity to efficiently handle big data ensures smooth and uninterrupted operations.

Additionally, the flexibility MongoDB affords to database designers is a significant asset in the context of a dynamic and evolving system like a plasma donation portal. Designers can effortlessly expand the schema, add new fields, or modify existing ones to adapt to changing requirements or emerging medical guidelines. This inherent flexibility enables seamless integration of new data points and empowers administrators to tailor the database to specific needs without the need for disruptive and time-consuming structural modifications.

Another noteworthy advantage of leveraging MongoDB for a COVID plasma donation portal lies in its compatibility with cloud computing. By utilizing MongoDB in a cloud environment, such as its proprietary platform Atlas, organizations can unlock a myriad of benefits. Firstly, the cloud deployment ensures high scalability, enabling effortless expansion of storage and computing resources to accommodate increasing data volumes or user traffic. This scalability proves crucial in times of surges in plasma demand or sudden influxes of donors. Moreover, MongoDB Atlas provides enhanced performance, thanks to its optimized infrastructure, thereby ensuring rapid data retrieval and processing. Consequently, this translates into a more responsive and seamless user experience.

Furthermore, the cost-effectiveness of utilizing MongoDB in the cloud cannot be overstated. By leveraging cloud services, organizations can avoid the upfront expenses associated with infrastructure acquisition, maintenance, and hardware upgrades. Instead, they can rely on a flexible pay-as-you-go model, where costs are directly tied to actual resource consumption. This not only minimizes initial investments but also allows for efficient resource allocation, ensuring optimal utilization without unnecessary overhead.

Future Perspective

The potential future scope for COVID plasma donation portal using MongoDB, NodeJS and Mongoose is enormous. Here are some potential research directions that this project could pursue:

1) Enhancing donor engagement: To boost donor motivation and engagement with the COVID plasma donation portal, features such as gamification, social media integration, and personalized donor dashboards can be implemented to enhance user experience.

2) Improving Donor Eligibility Criteria: Using machine learning algorithms and big data analytics, one can refine the eligibility criteria for COVID plasma donation to identify patterns and risks associated with plasma quality and effectiveness.

3) Ensuring donor safety: To guarantee donor security, the portal can be enhanced with automated screening and verification processes as well as real-time monitoring of donor health status.

4) Building Predictive Models with Machine Learning Algorithms: Predictive models can be created using machine learning algorithms to anticipate plasma unit availability and match donors with recipients in real-time.

5) Utilizing Blockchain Technology: Blockchain can be integrated into a portal to provide transparency and traceability throughout the plasma donation process, from collection and storage to distribution of plasma units.

6) Expanding the Portal: To reach a wider audience, the COVID plasma donation portal can be expanded to integrate with national and international blood banks as well as plasma collection centers.

7) Enhancing Plasma Donation Promotion: To encourage plasma donation and raise awareness about its critical role in combatting COVID-19, the portal can incorporate features like virtual events, webinars, and social media campaigns.

Overall, the COVID plasma donation portal presents a vast opportunity for further research and development using cutting-edge technologies like machine learning, blockchain, and big data analytics to enhance donor engagement, eligibility criteria, safety protocols, predictive modeling techniques, and scalability.

REFERENCES

[1] World Health Organization. (2020). Clinical Management of COVID-19: Interim Guidance. Geneva: World Health Organization; Retrieved from https://apps.who.int/iris/handle/10665/332196.

[2] Rajpal, S., Tong, M. S., Borchers, J. and Zareie P.(2020). COVID-19 Convalescent Plasma: Before You Jump. EClinicalMedicine 28(1), 100591. https://doi.org/10.1016/j.eclinm.2020.00951

[3] Chen, L., Xiong, J., Bao, L. and Shi, Y.(2020) Investigating Convalescent Plasma as a Potential Therapy for COVID-19: The Lancet Infectious Diseases 0(4), 398-400 https://doi.org/10.1016/S1473-3099(20)30141-9.

[4] Bengler, K., Manoharan, T., Schilling, T., & Zeier, A. (2016). A performance evaluation of MongoDB and MySQL in a large-scale, low-latency data ingestion application. In 2016 IEEE International Congress on Big Data (BigData Congress) (pp. 440-447). IEEE.

[5] Hecht, R., & Jablonski, S. (2011). NoSQL evaluation: A use case oriented survey. In International Conference on Cloud Computing and Services Science (pp. 511-520). Springer.

[6] Patel, C. D., & Patel, K. S. (2014). A comparative study on NoSQL databases: MongoDB vs Cassandra. International Journal of Computer Science and Information Technologies, 5(5), 6592-6596.

[7] Pliakos, V., & Doulkeridis, C. (2018). Performance and scalability of NoSQL databases: A case study with MongoDB and Couchbase. Future Generation Computer Systems, 86, 1362-1377.

[8] Fayyad-Kazan, H., & Guenane, R. (2019). Performance analysis of SQL and NoSQL databases for big data processing. Computers, Materials & Continua, 60(2), 597-615.

[9] Morrow, A., Hellerstein, J. M., & Franklin, M. J. (2005). Mongoose: Application-Level Semantics for Database Workloads. Proceedings of the 31st International Conference on Very Large Data Bases (VLDB).