

MINI PROJECT REPORT

A report submitted in partial fulfillment of the requirements of

**BACHELOR OF TECHNOLOGY
in
APPLIED COMPUTATIONAL SCIENCE AND ENGINEERING
By**

Name: Animesh (2101921520031)

**Under the Supervision of
Mr. Mohit Kumar**



**G.L. BAJAJ INSTITUTE OF TECHNOLOGY & MANAGEMENT,
GREATER NOIDA
Affiliated to**



DR. A.P.J. ABDUL KALAM TECHNICAL UNIVERSITY,LUCKNOW

2023-2024

**G.L. BAJAJ INSTITUTE OF TECHNOLOGY & MANAGEMENT,
GREATER NOIDA
DEPARTMENT OF APPLIED COMPUTATIONAL SCIENCE AND ENGINEERING**



CERTIFICATE

This is to certify that the “**Mini Project Report**” entitled “**Blood Bank App**” is being done by Animesh Sinha(2101921520031) in partial fulfillment of the requirements for the award of the degree of **BACHELOR OF TECHNOLOGY in Applied Computational Science and Engineering** for the academic session 2023-2024. They have completed their Summer Mini Project from GL BAJAJ INSTITUTE OF TECHNOLOGY AND MANAGEMENT in GREATER NOIDA.

Declaration

We hereby declare that the project work presented in this report entitled **“BLOOD BANK APP”**, in partial fulfilment of the requirement for the award of the degree of Bachelor of Technology in Applied Computational Science & Engineering, submitted to A.P.J. Abdul Kalam Technical University, Lucknow, is based on our own work carried out at Department of Applied Computational Science & Engineering, G.L. Bajaj Institute of Technology & Management, Greater Noida. The work contained in the report is original and project work reported in this report has not been submitted by me/us for award of any other degree or diploma.

Signature:

Name: Animesh

Roll No: 2101921520031

ACKNOWLEDGEMENT

I would like to express my sincere gratitude to **Mr. Mohit Kumar** for his invaluable guidance and support throughout the duration of the mini project on "**Blood Bank App**". His expertise, encouragement, and dedication significantly contributed to the success of this project.

I would also like to extend my thanks to the college for providing the necessary resources and environment conducive to learning and innovation.

I pay special thanks to the Dean of the department **Prof.(Dr.) Naresh Kumar** for his constructive criticism throughout my mini project.

I am extremely grateful to my department staff members and friends who helped me in successful completion of this mini project.

Name: - Animesh Sinha

Roll No: - 2101921520031



Mini Project Feedback Form

Student Name: _____

Company Name: _____

Contact Number: _____

Title: _____

Duration of Internship: _____

Email: _____

Summary of student's activities and responsibilities:

Performance Areas	Excellent	Above Average	Average	Below Average
Basic Engineering Knowledge				
Problem Analysis				
Design/Development Skills				
Solving complex problems using research-based techniques				
Familiarity with Modern Tools				
Engineer and Society				
Awareness on Environment and Sustainability				
Professional Ethics for Engineering Practice				
Individual and Teamwork				
Communication on complex engineering activities.				
Project Management in multidisciplinary environments				
Life-Long Learning				
Use Problem solving skill to develop efficient algorithmic solution				
Develop a solution in the area for AI, Data Analytics, Computer Vision and IoT etc.				

Based on the above observations, we put forward few suggestions regarding this internship program:

Supervisor's Signature:

Name:

Designation:

Official Email_Id...

TABLE OF CONTENT

Declaration.....	(ii)
Certificate	(iii)
Acknowledgement	(iv)
Feedback	(v)
Table of Content.....	(vi)
List of Figures	(viii)
List of Tables	(ix)

Chapter 1. Introduction Pg. No.

- 1.1 Problem Definition.....
- 1.2 Internship Overview / Specifications.....
- 1.3

Chapter 2. Motivation/Problem Statement..... Pg.No

- 2.1 Introduction
- 2.2 Existing System.....
- 2.3

Chapter 3. Plan of work Pg.No

- 3.1 Tools and Technology used
- 3.2

Chapter 4. Methodology..... Pg.No

- 4.1
- 4.2

Chapter 5. Result & Discussion Pg.No

- 5.1

Chapter 6. Conclusion, Limitation & Future Scope Pg.No

- 6.1.
- 6.2.

REFEERNCES

LIST OF FIGURES

Name	Page No.
Figure 5.1.1 Output a	15
Figure 5.1.2 Output b	15
Figure 5.1.3 Output c	16
Figure 5.1.4 Output d	16
Figure 5.2.1 Express code	17
Figure 5.2.2 Mongoose code	17
Figure 5.2.3 JavaScript code	18
Figure 5.2.4 JavaScript code	18

Chapter 1

Introduction

Problem Statement:

The blood donation ecosystem often faces challenges related to inefficiencies in blood inventory management, communication gaps between donors, organizations, hospitals, and the need for a secure and user-friendly platform. Traditional methods for tracking blood donations and managing inventory lack the efficiency required in emergency situations. Additionally, ensuring data security and privacy is paramount, considering the sensitive nature of health-related information. In response to these challenges, the Blood Bank App aims to address these critical issues by leveraging modern web technologies to create a comprehensive and secure solution.

This initiative aims to streamline blood donation processes, improve inventory management, and establish a secure role-based system for donors, organizations, hospitals, and administrators, contributing to enhanced transparency and efficiency in the blood donation ecosystem.

Chapter 2

MOTIVATION

The motivation behind the Blood Bank Project stems from a deeply rooted commitment to transforming the landscape of blood donation and inventory management. Recognizing the critical gaps in existing systems, there is a compelling drive to harness the power of modern web technologies to create a solution that is not only efficient but also user centric. The vision is to revolutionize the way blood donation processes are conducted, making them more accessible, transparent, and responsive to the urgent needs of hospitals and organizations.

This endeavor is motivated by the belief that advancements in technology can significantly impact healthcare systems, and by addressing the complexities of blood donation, we can make a meaningful difference in the lives of those in need. By providing a secure, role-based platform, the project aims to instill confidence in donors, streamline communication between stakeholders, and ultimately contribute to saving lives.

EXISTING SYSTEM

Manual Processes:

Current blood donation systems heavily rely on manual processes for donor registration, blood inventory tracking, and communication.

The manual nature of these processes can lead to delays, errors, and challenges in maintaining real-time data.

Limited Responsiveness:

Traditional systems often lack the agility required to respond swiftly to emergency situations, impacting the timely availability of blood units.

The inefficiencies in responsiveness can hinder hospitals and organizations in urgent need of blood supplies.

Communication Gaps

Communication between donors, organizations, and hospitals is often fragmented, making it challenging to coordinate and fulfill blood requests efficiently.

Existing systems struggle to establish seamless and transparent communication channels among stakeholders.

Data Security Concern:

Sensitive health information is handled within the existing systems, raising concerns about data security and privacy. The need for secure handling of donor and patient information becomes crucial in the context of healthcare-related platforms.

Inefficient Inventory Management:

Inventory management of blood units is often cumbersome, with limitations in tracking real-time quantity, types, and expiration dates.

The inefficiencies in inventory management can result in challenges in maintaining an optimal and readily available blood supply.

Adaptability Challenges:

Existing systems may struggle to adapt to changing requirements and emergency scenarios, hindering their effectiveness in critical situations.

The inability to quickly modify and update processes can impede the overall efficiency of blood donation systems.

Chapter 3

Plan Of Work

TOOLS AND TECHNOLOGY NEEDED

- **MongoDB:** A NoSQL database for flexible and scalable data storage.
- **Express.js:** A web application framework for building robust and scalable server-side applications.
- **React.js:** A JavaScript library for building user interfaces, providing a dynamic and interactive frontend.
- **Bootstrap:** A front-end framework for responsive and visually appealing user interfaces.
- **Redux:** A state management library for maintaining application state in a predictable manner, enhancing data flow and consistency.
- **Postman:** A comprehensive API testing and documentation tool to ensure accurate and reliable data communication between different components of the application.
- **Axios:** A promise-based HTTP client for making asynchronous HTTP requests, enhancing the speed and reliability of data retrieval and updates.
- **Visual Studio Code:** A lightweight and powerful code editor for efficient development.
- **Git:** A distributed version control system for tracking changes in the source code and facilitating collaboration among developers.
- **JSON Web Tokens (JWT):** A standard for secure communication between parties, used for user authentication and authorization.
- **bcrypt:** A cryptographic hashing function for securing passwords and sensitive data stored in the database.

Chapter 4

METHODOLOGY

1. Requirement Gathering

Conducting thorough discussions with potential users, organizations, and healthcare providers, to gather detailed requirements and expectations.

2. Conduct a Competitive Analysis:

Research existing Blood Bank systems and technologies to identify strengths, weaknesses, and gaps in the market.

Explore opportunities for innovation and differentiation in terms of accuracy, real-time performance, and adaptability.

3. Define Project Goals and Objectives:

Clearly outline the goals of the Blood bank App project, such as enhancing communication between user and hospitals or improving inventory management systems.

4. Design Prototypes:

Create wireframes and prototypes to visually articulate the user interface (UI). Solicit early feedback from stakeholders to refine the design. Establish the system architecture and database schema based on identified requirements.

5. Implementation:

Execute the development phase using the MERN stack technologies for both front-end and back-end components. Incorporate user authentication and authorization mechanisms to fortify system security.

6. User Interface Design:

Develop a user-friendly interface that emphasizes efficient data entry, retrieval, and user feedback to streamline the process of blood donation, inventory management, and user interactions within the application.

7. Testing Procedures:

Implement rigorous testing methodologies, encompassing unit testing, integration testing, and user acceptance testing (UAT). Leverage tools such as Postman for API testing and ensure robust error handling.

8. User Training and Onboarding:

Develop comprehensive user manuals and conduct structured training sessions for end-users, organizations, and administrators. Streamline the onboarding process to ensure user proficiency.

9. Deployment:

Deploy the Blood bank App on cloud platform such as Heroku, ensuring accessibility for the target audience.

Consider integration with communication devices or applications for practical use.

10. Monitoring and Optimization:

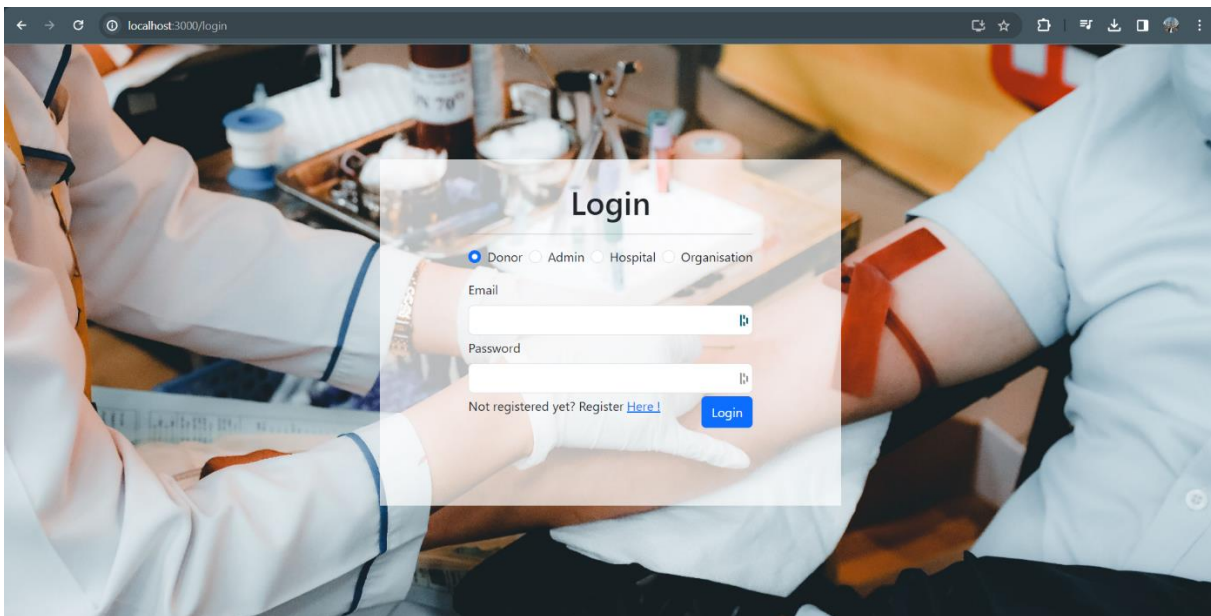
Implementing routine maintenance procedures and updates to proactively address any issues that may arise post-deployment. Additionally, incorporating user feedback systematically to enhance the application's performance and user experience.

.

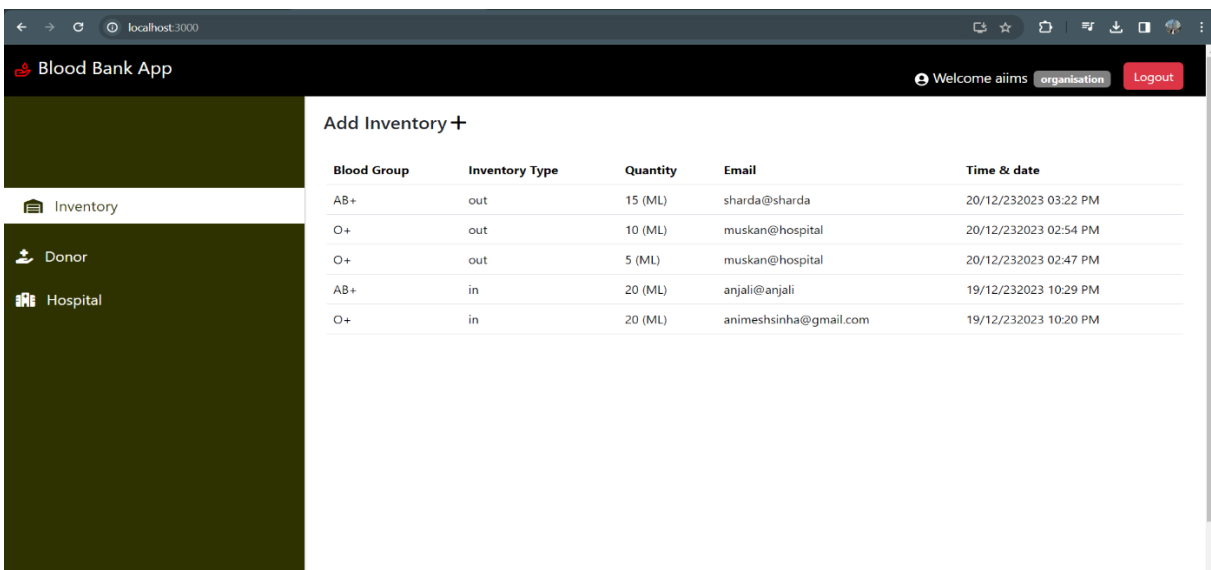
Chapter 5

RESULT & DISCUSSION

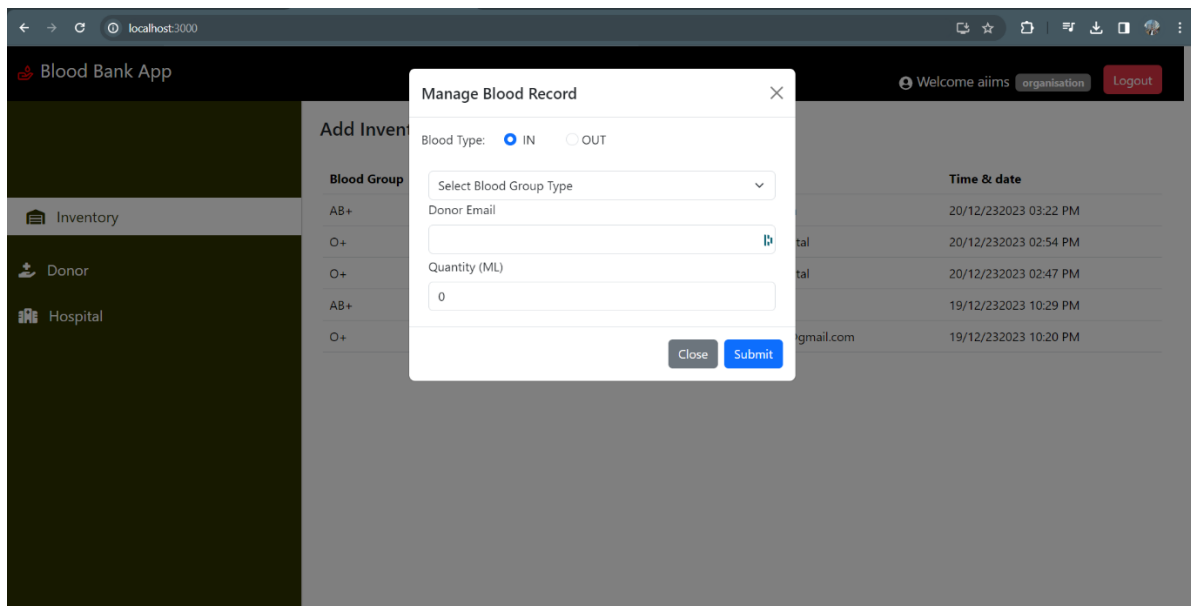
Result



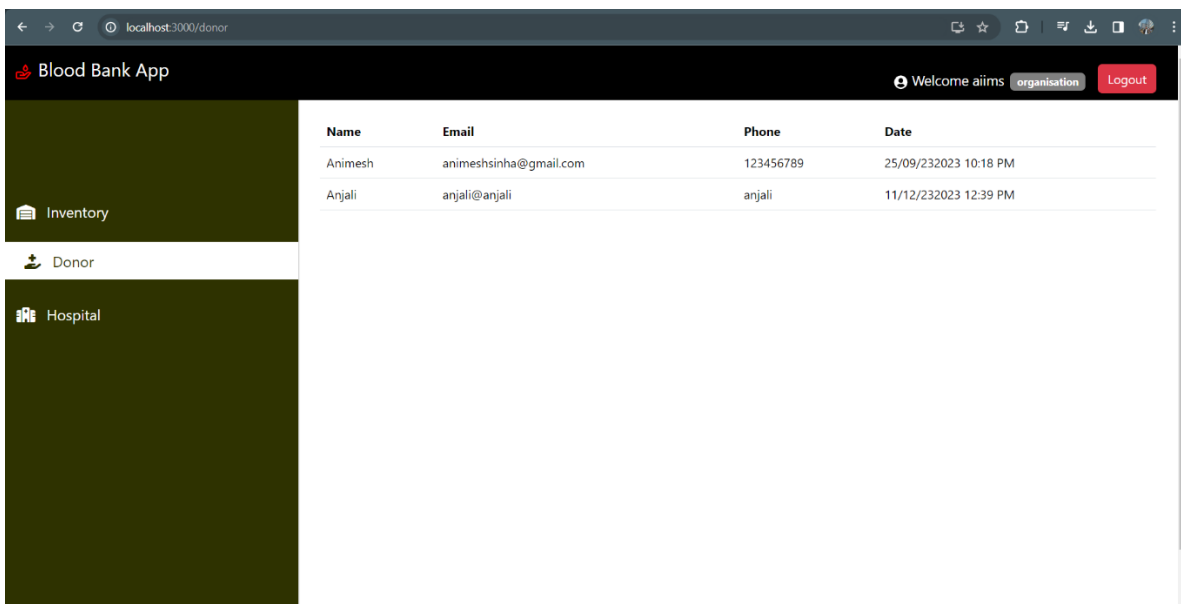
(A)



(B)



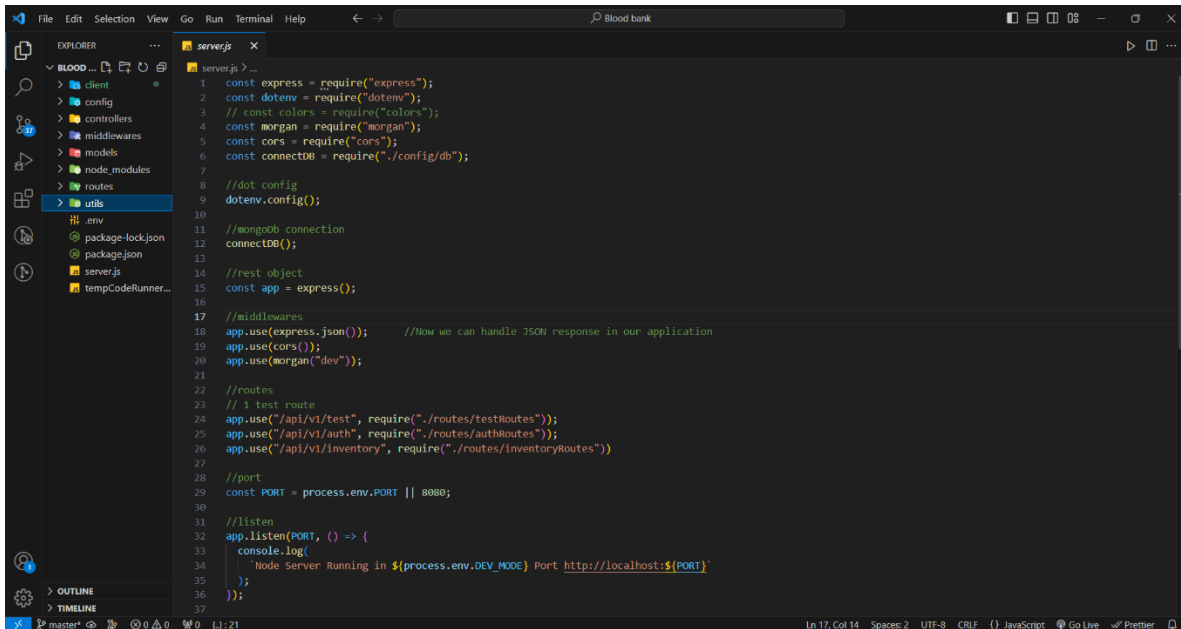
(C)



(d)

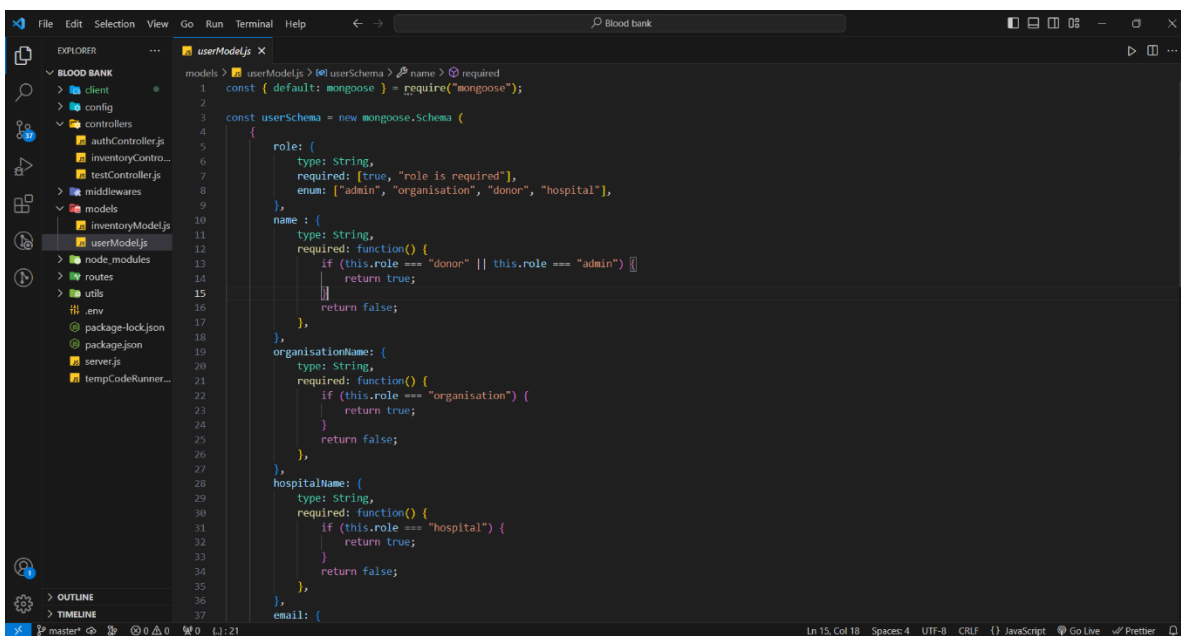
The user-friendly interface and intuitive design are expected to enhance user engagement, fostering active participation from donors, organizations, hospitals, and administrators. With streamlined processes for blood donation, inventory tracking, and request management, the project aims to optimize the efficiency of blood banks and related organizations.

DISCUSSIONS



The screenshot shows the VS Code editor with the 'Blood bank' project open. The Explorer sidebar on the left shows the project structure, with 'server.js' selected under the 'utils' folder. The main editor displays the content of 'server.js', which is a Node.js application setup. The code includes imports for express, dotenv, colors, morgan, cors, and mongoose. It sets up a config directory, connects to a MongoDB database, and defines routes for testing, authentication, and inventory. The application is configured to listen on port 8080.

```
1 const express = require("express");
2 const dotenv = require("dotenv");
3 // const colors = require("colors");
4 const morgan = require("morgan");
5 const cors = require("cors");
6 const connectDB = require("./config/db");
7
8 //dot config
9 dotenv.config();
10
11 //mongodb connection
12 connectDB();
13
14 //rest object
15 const app = express();
16
17 //middlewares
18 app.use(express.json()); //Now we can handle JSON response in our application
19 app.use(cors());
20 app.use(morgan("dev"));
21
22 //routes
23 // 1 test route
24 app.use("/api/v1/test", require("./routes/testRoutes"));
25 app.use("/api/v1/auth", require("./routes/authRoutes"));
26 app.use("/api/v1/inventory", require("./routes/inventoryRoutes"));
27
28 //port
29 const PORT = process.env.PORT || 8080;
30
31 //listen
32 app.listen(PORT, () => {
33   console.log(
34     `Node Server Running in ${process.env.DEV_MODE} Port http://localhost:${PORT}`
35   );
36 });
```



The screenshot shows the VS Code editor with the 'Blood bank' project open. The Explorer sidebar on the left shows the project structure, with 'userModel.js' selected under the 'models' folder. The main editor displays the content of 'userModel.js', which defines a Mongoose schema for a user. The schema includes fields for role, name, organisationName, hospitalName, and email. The role field has a required function that checks if the role is 'donor' or 'admin'. The organisationName, hospitalName, and email fields also have required functions that check if the role is 'organisation' or 'hospital'.

```
1 const { default: mongoose } = require("mongoose");
2
3 const userSchema = new mongoose.Schema (
4   {
5     role: {
6       type: String,
7       required: [true, "role is required"],
8       enum: ["admin", "organisation", "donor", "hospital"],
9     },
10    name: {
11      type: String,
12      required: function() {
13        if (this.role === "donor" || this.role === "admin") {
14          return true;
15        }
16        return false;
17      },
18    },
19    organisationName: {
20      type: String,
21      required: function() {
22        if (this.role === "organisation") {
23          return true;
24        }
25        return false;
26      },
27    },
28    hospitalName: {
29      type: String,
30      required: function() {
31        if (this.role === "hospital") {
32          return true;
33        }
34        return false;
35      },
36    },
37    email: {
```



```
client > src > App.js > App
You, 10 hours ago | 1 author (You)
1 import './App.css';
2 import { Routes, Route } from 'react-router-dom';
3 import HomePage from './pages/HomePage';
4 import Login from './pages/auth/Login';
5 import Register from './pages/auth/Register';
6 import { ToastContainer } from 'react-toastify';
7 import 'react-toastify/dist/ReactToastify.css';
8 import ProtectedRoute from './components/Routes/ProtectedRoute';
9 import PublicRoute from './components/Routes/PublicRoute';
10 import Donor from './pages/Dashboard/Donor';
11 import Hospital from './pages/Dashboard/Hospital';
12 import Organisation from './pages/Dashboard/Organisation';
13
14 function App() {
15   return (
16     <>
17       <Routes>
18         <Route
19           path="/"
20           element={
21             <ProtectedRoute>
22               <HomePage />
23             </ProtectedRoute>
24           }
25         />
26         <Route
27           path="/login"
```

```
[0] GET /api/v1/inventory/get-hospitals 304 60.559 ms - -
[0] GET /api/v1/auth/current-user 304 31.328 ms - -
[0] GET /api/v1/inventory/get-hospitals 304 54.813 ms - -
[0] GET /api/v1/auth/current-user 304 38.631 ms - -
[0] GET /api/v1/inventory/get-inventory 304 74.266 ms - -
[0] GET /api/v1/auth/current-user 304 31.767 ms - -
[0] GET /api/v1/inventory/get-inventory 304 70.319 ms - -
```

```
client > src > components > shared > Form > Form.js > ...
24 </>
25 <h1 className='text-center'>{formTitle}</h1>
26 <hr />
27
28 <div className='d-flex mb-3' style={{
29   justifyContent: 'center',
30 }}>
31
32 <div className='form-check'>
33   <label className='form-check-label' htmlFor='donorRadio'>
34     Donor
35   </label>
36   <input className='form-check-input' type='radio' name='role' id='donorRadio' value='donor' onChange={(e) => setRole(e.target.value)} defaultChecked />
37 </div>
38
39 <div className='form-check ms-2'>
40   <input className='form-check-input' type='radio' name='role' id='adminRadio' value='admin' onChange={(e) => setRole(e.target.value)} />
41   <label className='form-check-label' htmlFor='adminRadio'>
42     Admin
43   </label>
44 </div>
45
46 <div className='form-check ms-2'>
47   <input className='form-check-input' type='radio' name='role' id='hospitalRadio' value='hospital' onChange={(e) => setRole(e.target.value)} />
48   <label className='form-check-label' htmlFor='hospitalRadio'>
49     Hospital
50   </label>
51 </div>
52
53 <div className='form-check ms-2'>
54   <input className='form-check-input' type='radio' name='role' id='organisationRadio' value='organisation' onChange={(e) => setRole(e.target.value)} />
55   <label className='form-check-label' htmlFor='organisationRadio'>
56     Organisation
57   </label>
58 </div>
59
60 </div>
```

Engaging in discussions about the challenges faced during the development process has led to effective problem-solving and continuous improvement.

Challenges related to integration complexity, data security, and user adoption have prompted thoughtful discussions on solutions and best practices.

Chapter 6

CONCLUSION, LIMITATION & FUTURE SCOPE

Conclusion

In conclusion, The Blood bank App stand at the forefront of technological innovation. The combination of MongoDB, Express.js, React.js, and Node.js has empowered the development of a robust and user-centric platform. By prioritizing requirements gathering, intuitive user interface design, and iterative development, the project aims to address the dynamic needs of donors, organizations, hospitals, and administrators.

The deployment on a cloud platform ensures widespread accessibility, contributing to the project's overarching goal of optimizing the efficiency of blood banks. Real-time features, continuous user feedback integration, and proactive maintenance practices collectively anticipate a positive user experience and sustained system reliability.

As the project unfolds, the commitment to compliance, security, and community impact underscores its significance in promoting organized, responsive, and technologically advanced blood donation practices. This endeavor aligns with the broader ethos of leveraging technology to create meaningful solutions for healthcare and community welfare.

Limitations

Technological Constraints:

The project's success is contingent on the reliability and scalability of the chosen technologies. Unforeseen technological challenges may impact performance and scalability.

Dependency on Internet Connectivity:

Accessibility to the application relies on internet connectivity, potentially limiting usage in areas with inconsistent or limited network access.

User Adoption Challenges:

The success of the project relies on user adoption. Overcoming potential resistance or hesitation from users and stakeholders may pose a challenge.

Data Privacy Concerns:

Despite robust security measures, concerns related to data privacy may arise, especially when handling sensitive health information. Compliance with data protection regulations is paramount.

Resource Constraints:

Limited resources, including time and personnel, may impact the project's ability to implement all desired features or respond rapidly to evolving user needs.

Continuous Maintenance Demand:

Ongoing maintenance demands consistent attention. The need for routine updates and bug fixes may impose resource requirements and impact the project's long-term sustainability.

Future Scope

Search Functionality:

Incorporating an advanced search functionality to enable users to discover specific blood types, donors, or nearby hospitals more efficiently.

Mobile Application Development:

Expanding accessibility by developing a mobile application, allowing users to interact with the system seamlessly from their smartphones, enhancing convenience and reach.

Integration with Health Systems:

Exploring opportunities for integration with existing health systems to streamline data sharing and enhance coordination between blood banks and healthcare providers.

Predictive Analytics Enhancement:

Enhancing predictive analytics capabilities to provide more accurate forecasts regarding blood inventory needs, aiding in proactive management and reducing shortages.

Continuous User Feedback Integration:

Establishing mechanisms for continuous user feedback integration to inform iterative development, ensuring the project evolves in alignment with user expectations.

References

- YouTube.
- MongoDB Documentation. (2023). [Online]. Available: <https://docs.mongodb.com/>
- Node.js Documentation. (2023). [Online]. Available: <https://nodejs.org/en/docs/>
- Express.js Documentation. (2023). [Online]. Available: <https://expressjs.com/>
- React.js Documentation. (2023). [Online]. Available: <https://reactjs.org/docs/getting-started.html>