

**BLOODSHARE HUB**

**P20CAP2702 - MINI PROJECT REPORT**

***Submitted by***

**SRIRAM S**

# Register No: 23MCA058

*in partial fulfilment for the award of the degree*

*of*

# MASTER OF COMPUTER APPLICATIONS

***in***

**COMPUTER APPLICATIONS**

# KUMARAGURU COLLEGE OF TECHNOLOGY

(An Autonomous Institution Affiliated to Anna University, Chennai)

**JUNE 2024**

**KUMARAGURU COLLEGE OF TECHNOLOGY**

(An Autonomous Institution Affiliated to Anna University, Chennai)

**COIMBATORE -641 049**

Department of Computer Applications

**MINI PROJECT WORK JUNE 2024**

This is to certify that the project entitled

**Bloodshare hub**

is the bonafide record of project work done by

## SRIRAM S 23MCA058

of MCA (Computer Applications) during the year 2023-2024.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

## Project Guide Head of the Department

**DECLARATION**

I affirm that the project work titled **Bloodshare hub** being submitted in partial fulfilment for the award of Master of Computer Application is the original work carried out by me. It has not formed the part of any other project work submitted for award of any degree or diploma, either in this or any other University.

(Signature of the Candidate)

## ABSTRACT

"Bloodshare Hub" is a vital web application bridging the gap between blood donors and hospitals. Hospitals can send notifications when they require blood, and donors can easily respond to these requests. The platform facilitates seamless communication and coordination between donors and hospitals, ensuring timely and efficient blood donations. Features include real-time notifications, donor profiles, and hospital requests, creating a user-friendly experience for both donors and hospitals. "Blood Share Hub" is a crucial role in improving access to blood donations, saving lives, and supporting healthcare systems.

Technology Used For:

**Web Application:**

Front-end: HBS, CSS, JavaScript

Back-end: Express.js, Node.js, MongoDB

IDE: Visual Studio Code

## PROJECT DESCRIPTION

### MODULE DESCRIPTION

* Visit Profile
* Request donor.
* Respond to request.
* Conduct blood camps.

**Visit Profile:**

Donor profile: Displays donor information, including blood type, contact details, and donation history. Donors can update their profiles.

Admin profile: Displays administrator information and allows admins to update donor profiles.

**Request Donor:**

Donor request submission: Allows hospital to submit requests for blood donations, specifying the required blood type and quantity.

Donor matching: Matches donor profiles with the specified request criteria, aiding in quick donor selection and coordination.

**Respond to Request:**

Donor response form: Allows donors to respond to blood donation requests, indicating availability and willingness to donate.

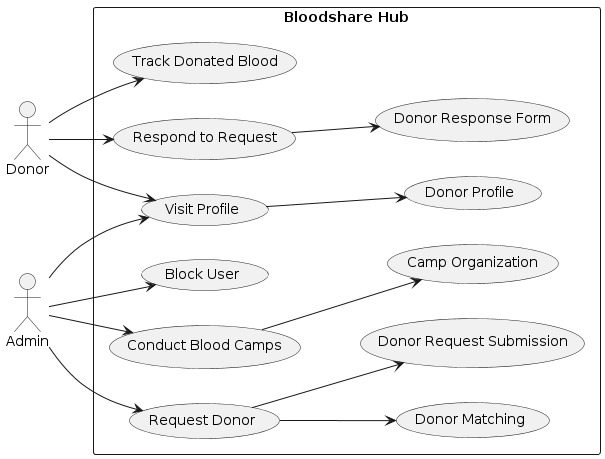
Confirmation notification: Sends a confirmation to the hospital once a donor agrees to donate, providing contact information for coordination.

**Conduct Blood Camps:**

Camp organization: Lets administrators schedule and organize blood donation camps, specifying location, date, and time.

## DATAFLOW DIAGRAM

**USE CASE DIAGRAM:**

****

## DATABASE DESIGN

* **Document Name**: Hospital

{

\_id: ObjectId (Primary Key),

email: String,

password: String,

H\_name: String,

appointments: [

{

id: String,

date: String,

name: String,

bloodGroup: String,

status: String

} ],

place: String

}

* **Document Name**: Request

{

\_id: ObjectId (Primary Key),

hospitalName: String,

hospitalAddress: String,

hospitalDistrict: String,

patientId: String,

phoneNumber: Number,

dateNeeded: String,

bloodGroup: String,

purpose: String,

responseId: String

}

* **Document Name**: Hospital Inventory

{

\_id: ObjectId (Primary Key),

hospitalId: String,

bloodInventory: {

'O+': [

{

id: String,

status: String,

expiryDate: Date

}

],

'Others': [

{

bloodGroup: String,

id: String,

status: String,

expiryDate: Date

}]}

}

* **Document Name**: Donor Details

{

\_id: ObjectId (Primary Key),

name: String,

email: String,

phno: String,

bloodGroup: String,

password: String,

age: Number,

district: String,

donatedTimes: Number,

last\_donate: String,

weight: Number,

status: Number,

last\_edit: Date,

image: String

}

* **Document Name**: Camp Details

{

\_id: ObjectId (Primary Key),

date: String,

district: String,

phoneNumber: String,

time: String,

place: String,

response: {

Morning: [String],

Afternoon: [String],

Evening: [String]

},

startTime: String,

endTime: String,

campId: Number

}

## FORM DESIGN

**SCREENSHOT**

A screenshot of a computer

Description automatically generated

Fig 1 Profile

A screenshot of a medical form

Description automatically generated

Fig 2 Profile

A screenshot of a phone number

Description automatically generated

Fig 3 New Camp

## TESTING

### PERFORMANCE TESTING

The "Bloodshare Hub" platform is committed to evaluating its speed, reliability, and scalability. Through rigorous testing methodologies like unit testing, integration testing, load testing, and stress testing, the application's performance under various conditions is thoroughly assessed. Continuous monitoring and optimization of the infrastructure ensure that the platform maintains optimal performance even during peak traffic periods. By prioritizing performance testing, Blood share Hub aims to deliver a seamless user experience, enhance customer satisfaction, and drive its mission of bridging blood donors with hospitals for critical needs.

### SYSTEM TESTING

In system testing for the Bloodshare Hub project, integration testing is conducted on components such as reporting issues, adding camps, user registration, custom fields, and tags. These components are taken as input to test the system elements, including its environment and behaviour. The goal is to ensure that the administrator system efficiently fulfils its objectives with minimal system resource usage and in a user-friendly manner. Software is employed to manage system resources effectively, enabling the administrator to handle processes efficiently and with ease.

### REGRESSION TESTING

Regression testing is a critical part of the development process for the Bloodshare Hub project. Whenever a change is made to the application, such as fixing a bug or adding a new feature, there is a possibility that it could affect other areas of the application.

The purpose of regression testing is to verify that the changes made do not inadvertently introduce new issues or violate any existing functionality or business rules. It ensures that the application continues to function correctly even after changes have been made. In the context of the Blood share Hub project, regression testing is performed whenever bugs are fixed, or new features are added to ensure that these changes do not negatively impact other modules of the application. This helps maintain the overall stability and reliability of the blood share Hub platform, ensuring a seamless experience for blood donors and hospitals.

### UNIT TESTING

Unit testing is a level of software testing where individual units/components of software are tested. It is necessary to test each module by giving them different kind of inputs. This ensures the stability and consistency of the application.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **TEST CASE ID** | **TEST DESCRIPTION** | **EXPECTED RESULT** | **ACTUAL RESULT** | **RESULT** |
| T1 | Login | Check if user credentials are valid | User is successfully logged in | PASS |
|  |  |  | Invalid credentials provided | FAIL |
| T2 | Adding a new Camp | Ensure all required fields are filled correctly | Camp is successfully added to the inventory | PASS |
|  |  |  | Missing information leads to failed | FAIL |
| T3 | Editing a profile | Verify changes are correctly updated | User details are successfully updated | PASS |
|  |  |  | Failed to update user details | FAIL |
| T4 | Blocking a User | Confirm successful blocked user | User is Blocked | PASS |
|  |  |  | Failed to block the user | FAIL |
| T5 | Adding a new Request for blood | Process request without errors | Requests are successfully processed | PASS |
|  |  |  | Errors encountered during request processing | FAIL |
| T6 | Taking blood from inventory | Blood was successfully taken from inventory | Blood removed from inventory | PASS |
|  |  |  | Failed to remove blood from inventory | FAIL |

## CONCLUSION AND FUTURE ENHANCEMENT

### CONCLUSION

### Here's a conclusion for your project, Bloodshare Hub:

### "In conclusion, Bloodshare Hub stands as a testament to user-centric design and meticulous development. The platform's user-friendly interface ensures ease of use for donors and administrators alike. Through rigorous testing and adherence to coding standards, the system has been proven to be operational and reliable. While the current iteration marks a significant milestone, there remains ample room for enhancement and refinement. Future iterations could focus on further streamlining user interactions, improving backend processes for even greater efficiency, and expanding features to cater to evolving needs. Bloodshare Hub is not just a project; it's a commitment to continuous improvement and a testament to the power of technology to facilitate life-saving endeavours."

### FUTURE ENCHANCEMENT

In the future the system will have the following enhancements,

1. Implement predictive analytics for blood usage based on existing data.
2. Enhance the matching algorithm for more accurate donor matches.
3. Improve the user interface for a more intuitive experience.