

**INTELLIGENT BLOOD BANK MANAGEMENT  
SYSTEM WITH INTEGRATED DECISION MAKING  
SYSTEM**

**A PROJECT REPORT**

*Submitted by*

**AYUSH SINGH [Reg No: RA1511008010221]  
ADITYA RAJPUT [Reg No: RA1511008010237]  
REEZ PATEL [Reg No: RA1511008010255]  
MONISH DE [Reg No: RA1511008010261]  
SHREYA DHAGA [Reg No: RA1511008010295]**

*Under the guidance of*

**Mr. K. Navin**

(Asst. Professor, Department of Information Technology)

*in partial fulfillment for the award of the degree*

*of*

**BACHELOR OF TECHNOLOGY**

in

**INFORMATION TECHNOLOGY**

of

**FACULTY OF ENGINEERING AND TECHNOLOGY**



S.R.M. Nagar, Kattankulathur, Kancheepuram District

**APRIL 2019**

# SRM UNIVERSITY

(Under Section 3 of UGC Act, 1956)

## BONAFIDE CERTIFICATE

Certified that this project report titled “**INTELLIGENT BLOOD BANK MANAGEMENT SYSTEM WITH INTEGRATED DECISION MAKING SYSTEM**” is the bonafide work of “**AYUSH SINGH [Reg No: RA1511008010221], ADITYA RAJPUT [Reg No: RA1511008010237], REEZ PATEL [Reg No: RA1511008010255], MONISH DE [Reg No: RA1511008010261], SHREYA DHAGA [Reg No: RA1511008010295]**”, who carried out the project work under my supervision. Certified further, that to the best of my knowledge the work reported herein does not form any other project report or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.

**SIGNATURE**

Mr. K. Navin  
**GUIDE**  
Asst. Professor  
Dept. of Information Technology

Signature of the Internal Examiner

**SIGNATURE**

Dr. G. Vadivu  
**HEAD OF THE DEPARTMENT**  
Dept. of Information Technology

Signature of the External Examiner

## **ABSTRACT**

This project is requested by The Rotary Blood Bank and classifies as an m-health project. This project is aimed to help blood banks to manage their inventory systems using modern solutions. Our project has been developed keeping in view of the blood banks and the people that work for them, we aim to provide for excellent user experience. Intelligent data entry and retrieval classifies are the essentials for every modern inventory system. The digital system design provides a context-dependent user interface that makes it easier for the transition. Using our system the blood stored within the blood banks can easily be tracked audited and used. To fetch and maintain audits is a big task in the blood banks, our system addresses this concern with a smart search solution. Managing data is easier and efficient and organized. This system provides you with basic analytics on the basis of your data. Duplication of data and the rising concern of illicit blood transfer in the red market is also addressed. There is the security of data and rights that are only given to the administrator, this system has access to specifiers and a distributed database system that makes it easier to expand for the future. We hope that this project can grow to become a standard for blood bank inventory systems across various regions.

## **ACKNOWLEDGEMENTS**

I would like to express my deepest gratitude to, Mr Rajagopal Aravindan and the Rotary Blood bank for providing us with an opportunity to work in this healthcare project, and to my guide, Mr K. Navin for his valuable guidance, consistent encouragement, personal caring, timely help and providing me with an excellent atmosphere to accomplish this project. All through the work, in spite of his busy schedule, he has extended cheerful and cordial support to me for completing this work.

**Author**

# TABLE OF CONTENTS

<b>ABSTRACT</b>	<b>iii</b>
<b>ACKNOWLEDGEMENTS</b>	<b>iv</b>
<b>LIST OF TABLES</b>	<b>viii</b>
<b>LIST OF FIGURES</b>	<b>x</b>
<b>ABBREVIATIONS</b>	<b>xi</b>
<b>LIST OF SYMBOLS</b>	<b>xii</b>
<b>1 INTRODUCTION</b>	<b>1</b>
1.1 Insight to Database Management Systems . . . . .	1
1.2 Types of Database Management Systems . . . . .	2
1.2.1 On the basis of the data model . . . . .	2
1.2.2 On basis of users . . . . .	5
1.2.3 On the basis of sites over which network is distributed . . .	6
1.3 Blood Bank Inventory Management . . . . .	7
1.4 Importance of shift . . . . .	7
1.5 Accomplishment . . . . .	8
<b>2 LITERATURE SURVEY</b>	<b>9</b>
2.1 Blood for sale: India's illegal 'red market' . . . . .	9
2.2 Database Lifecycle . . . . .	9
2.3 Concepts of Database Management Systems . . . . .	10
2.4 The Ultimate Guide to SDLC . . . . .	10
2.5 ReactJS . . . . .	11
2.6 What Happens to Donated Blood . . . . .	11
2.7 MongoDB . . . . .	11

2.8	Foundations of Software Testing . . . . .	12
2.9	Box Detection Algorithm . . . . .	12
2.10	Offline Handwriting Recognition Using a Generic Algorithm . . . . .	13
<b>3</b>	<b>System Analysis</b>	<b>14</b>
3.1	Problem Definition . . . . .	14
3.2	Proposed System . . . . .	14
3.2.1	Advantages of Proposed System . . . . .	15
3.3	Requirements . . . . .	15
3.3.1	Hardware Requirements . . . . .	15
3.3.2	Software Requirements . . . . .	15
3.4	Issues in existing methodology . . . . .	16
3.5	New methodology . . . . .	17
3.5.1	Gathering data from donors . . . . .	17
3.5.2	Process partition . . . . .	17
3.5.3	Analytics . . . . .	17
<b>4</b>	<b>System Design</b>	<b>18</b>
4.1	Physical Design . . . . .	18
4.1.1	User Interface Design . . . . .	18
4.1.2	Database Design . . . . .	20
4.1.3	Modules . . . . .	24
4.2	Logical Design . . . . .	31
4.3	RESTful API . . . . .	31
4.3.1	UML Diagrams . . . . .	32
<b>5</b>	<b>Testing</b>	<b>35</b>
5.1	Testing Objectives . . . . .	35
5.2	Types of tests . . . . .	35
5.2.1	Unit Testing . . . . .	35
5.2.2	Integration Testing . . . . .	36
5.2.3	Functional Testing . . . . .	36
5.2.4	Acceptance Testing . . . . .	36

5.2.5	System Testing . . . . .	37
5.2.6	White Box Testing . . . . .	37
5.2.7	Black Box Testing . . . . .	37
5.2.8	Alpha Testing . . . . .	37
5.2.9	Beta Testing . . . . .	38
5.3	Tests Strategy and Approach . . . . .	38
5.3.1	Test Objective . . . . .	38
5.3.2	Features to be tested . . . . .	38
<b>6</b>	<b>Conclusion</b>	<b>39</b>
<b>7</b>	<b>Future Enhancement</b>	<b>40</b>
7.1	Exception rate . . . . .	40
7.2	Donor location . . . . .	40
7.3	Banks closet . . . . .	40
7.4	Notifications . . . . .	41
7.5	Register to digital . . . . .	41
<b>A</b>	<b>Otsu Binarization Thresholding</b>	<b>42</b>
A.1	The Mathematics . . . . .	42
A.2	The algorithm that can be implemented . . . . .	43
A.3	Availability . . . . .	44

## LIST OF TABLES

4.1	NoSQL vs SQL . . . . .	21
-----	------------------------	----



## LIST OF FIGURES

1.1	Fields . . . . .	2
1.2	Record . . . . .	3
1.3	Table . . . . .	3
1.4	Object Oriented Database . . . . .	3
1.5	Object Relational Database . . . . .	4
1.6	Hierarchial Database . . . . .	4
1.7	Network Database . . . . .	5
1.8	Centralized Database System . . . . .	6
1.9	Distributed Database System . . . . .	7
1.10	Blood Bank Inventory Management . . . . .	7
4.1	Wireframe 1 . . . . .	19
4.2	Wireframe 2 . . . . .	19
4.3	Wireframe 3 . . . . .	20
4.4	Input Excel Sheet . . . . .	22
4.5	Donor Consent Form . . . . .	23
4.6	MongoDB Compass showing various collections and documents . .	24
4.7	Address Data Model . . . . .	25
4.8	Blood Bank Data Model . . . . .	25
4.9	User Data Model . . . . .	26
4.10	Blood Discard Data Model . . . . .	26
4.11	Blood Donation Data Model . . . . .	27
4.12	Blood Input Record Data Model . . . . .	27
4.13	Donor Rejection Data Model . . . . .	27
4.14	Stock Register Data Model . . . . .	28
4.15	Blood Bags Data Model . . . . .	29
4.16	Consumables Data Model . . . . .	29

4.17 Chemicals Data Model . . . . .	30
4.18 Kits Data Model . . . . .	30
4.19 Serums Data Model . . . . .	31
4.20 Use Case Diagram . . . . .	32
4.21 Sequence Diagram . . . . .	33
4.22 Flow Diagram . . . . .	34

## ABBREVIATIONS

<b>HIV</b>	Human immunodeficiency virus
<b>HEP B</b>	hepatitis B
<b>HEP C</b>	hepatitis C
<b>MP</b>	Mucopolysaccharidosis
<b>VDRL</b>	Venereal Disease Research Laboratory
<b>ACID</b>	Atomicity, Consistency, Isolation, Durability
<b>SQL</b>	Structured Query Language
<b>NN</b>	Neural Network
<b>MERN</b>	MongoDB, Express, React, Node
<b>PUS</b>	Purchased, Used and Stock
<b>KPI</b>	Key Performance Indicators
<b>DOB</b>	Date of birth
<b>JSON</b>	JavaScript Object Notation
<b>API</b>	Application Program Interface
<b>REST</b>	Representational State Transfer
<b>HTTP</b>	Hyper Text Transfer Protocol

## LIST OF SYMBOLS

$\omega$	Probability
$\sigma$	Standard Deviation
$\mu$	Mean
$\Sigma$	Summation

# **CHAPTER 1**

## **INTRODUCTION**

Blood donation is a voluntary procedure. A person can give their blood to someone who may need a transfusion. Millions of people need blood transfusions each year. Some may need blood during surgery. When we go to give blood we will be asked to perform simple tests that will check our health state. When there is any problem in our health the test will discover it early and you can then have suitable health care and reassurance. After this state, the donation authorities take in the donor details and screen the blood. In the screening process, the donor's blood type is identified and it is checked for infections like Human immunodeficiency virus (HIV), hepatitis B (HEP B), hepatitis C (HEP C), Mucopolysaccharidosis (MP), and Venereal Disease Research Laboratory (VDRL). The blood is usually discarded if any of these tests are positive. After the initial screening, the blood is taken from the candidate and stored in the banks in optimal conditions. This whole system including the storage and feeding the data is done through registers manually.

This is a time taking process and often leads to many errors due to wrong blood transfusions or incorrect samples being sent to the hospitals from the blood banks. This also leads to accounting mismanagement and illegal trafficking of blood in the red market reports according to BBC.

### **1.1 Insight to Database Management Systems**

In the current world, we have a lot of raw data that can be processed and kept in a systematic manner to retrieve information. A Database is this type of collection of information. Databases help in making the project easily accessible, manageable and update-able. Data is organized into tables, each table has rows and columns. Data is indexed to find relevant information. Database processes help to create and update them through the use of queries. A query is an action of retrieving data from a database.

Computer databases contain data of large records or filed that can indefinitely expand provided the hardware is available.

Typically, a database manager provides the user with the ability to control some part of the data, these are known as access levels or user permissions. User Permits are put in place to keep data secure and abstract a portion of the database. A good database offers Atomicity, Consistency, Isolation, Durability (ACID) compliance to guarantee that data is consistent. The Database Management Systems serves an interface between the database and end users or application programmers, to ensure that data is easily accessible. A database is useful for providing a central view of data that is organized and quantified, as well as ensuring there is no redundancy in the database.

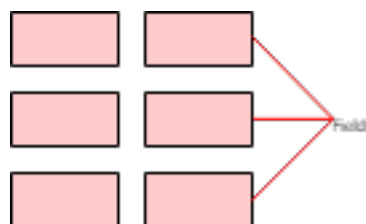
## 1.2 Types of Database Management Systems

Databases Management Systems can be done using various languages and procedures. Following give the basic types of classifications.

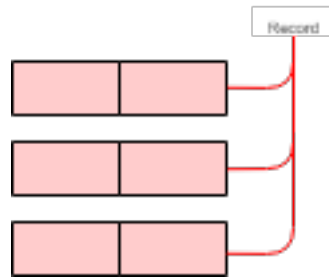
### 1.2.1 On the basis of the data model

#### a. Relational Database

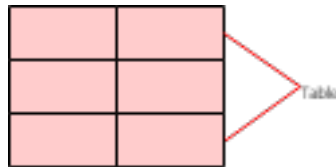
A relational database is a database that has a collective set of many datasets organized using tables, records and columns. These databases ensure that there is a well-defined relation between different databases tables. These tables share information which facilitates search, organization and reporting. These Databases are backed by usage of Structured Query Language (SQL), which is usually a standard.



**Figure 1.1:** Fields



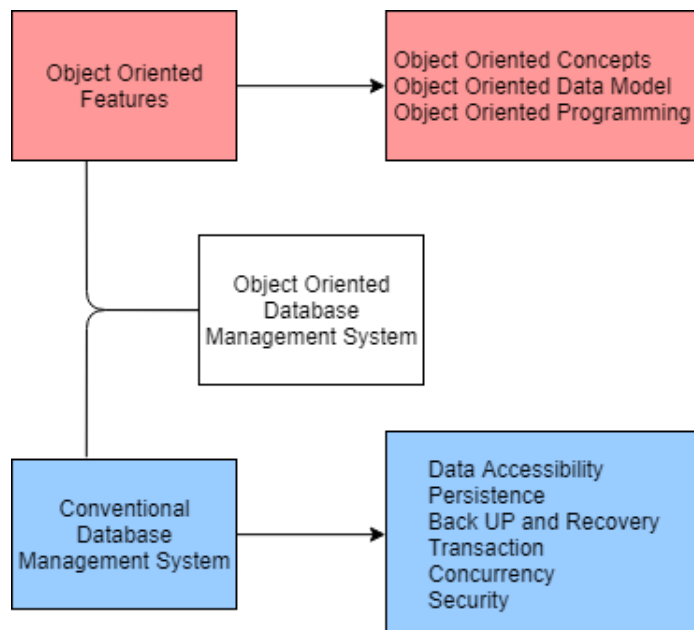
**Figure 1.2: Record**



**Figure 1.3: Table**

## b. Object Oriented Database

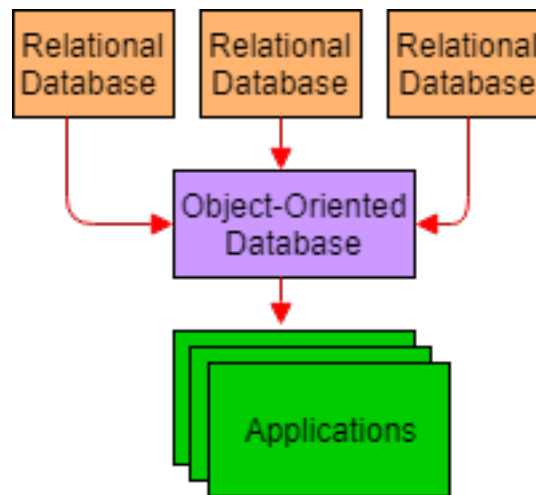
An object-oriented database is a database that allows the definition of objects, which is different from normal database objects. These systems subscribe to the model with the information shown by objects. Objects refer to the ability to develop a product and then define and name it, this enables the object to be referenced as a unit without having complex system involvement.



**Figure 1.4: Object Oriented Database**

### c. Object Relational Database

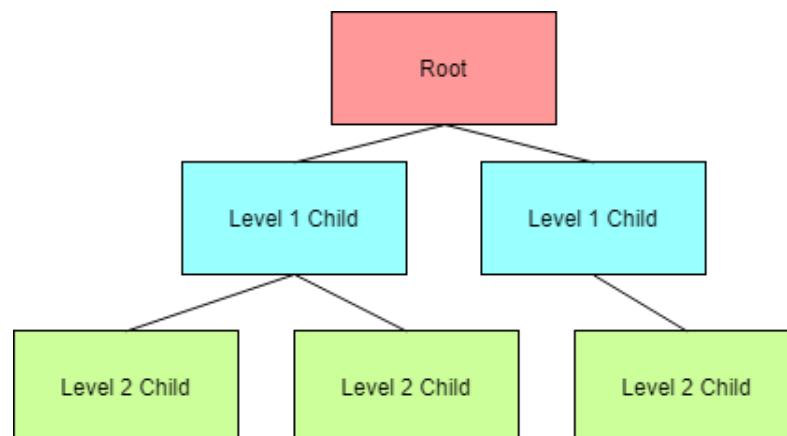
An Object Relational Database is a database management system that has components of both relational and object-oriented components. Object Relational Databases support the basic components of an object-oriented database model in schema and queries. It takes characteristics from both the Relational and Object Oriented Models.



**Figure 1.5:** Object Relational Database

### d. Hierarchical Database

A hierarchical database is a database that shows a one-to-many relationship of a database and therefore is in a tree-like representation. These links elements to a parent/primary record.

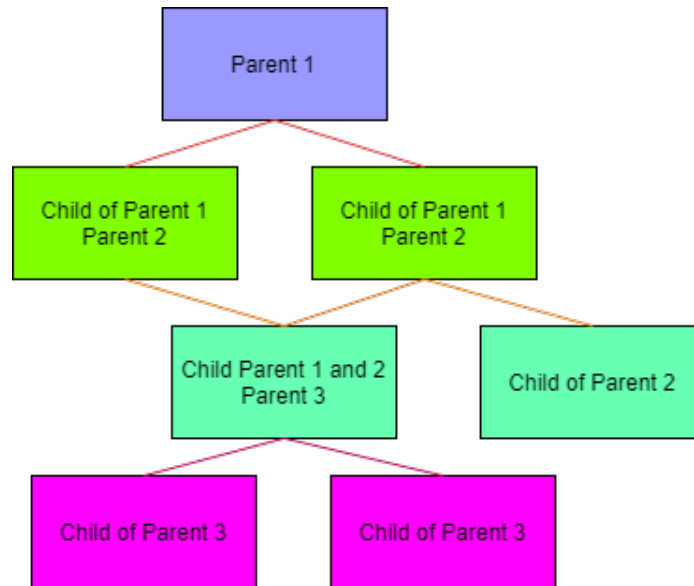


**Figure 1.6:** Hierarchical Database



### e. Network Database

A Network Database is a database that shows many-to-many relations and therefore allows multiple parent multiple child records in the database. Essentially it forms a net-like structure.



**Figure 1.7:** Network Database

## 1.2.2 On basis of users

### a. Single User Database

A Single User Database allows only one connection to the database at a given point of time. This means that only one user is allowed to connect to the database throughout the server at a given moment of time.

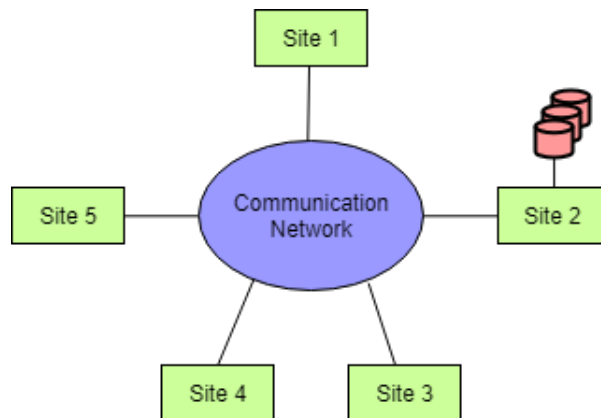
### b. Multiple Users Database

A Multiple User Database allows more than one connection to the database at a given point if time.

### 1.2.3 On the basis of sites over which network is distributed

#### a. Centralized Database System

A centralized database system has a single processor with its storage devices and other peripherals, due to this it is physically confined to a location. Data can be accessed from many sites using a single network while the maintenance is on a single site.



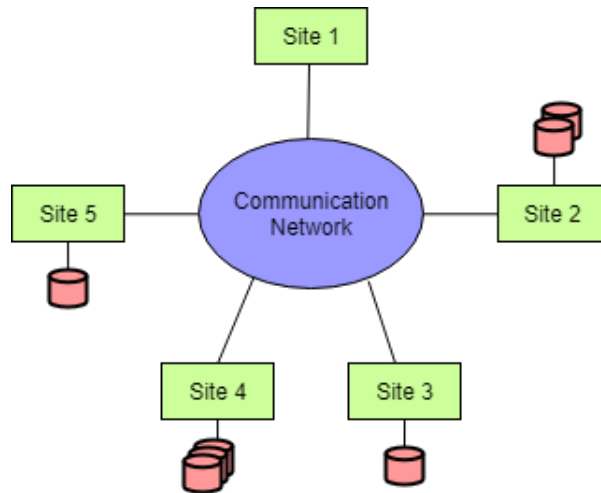
**Figure 1.8:** Centralized Database System

#### b. Parallel Network Database System

A parallel database system has multiple processing units and disks storage is in parallel. This improves the input/output speed and therefore are used in the application that has to query large transaction rates.

#### c. Distributed Database System

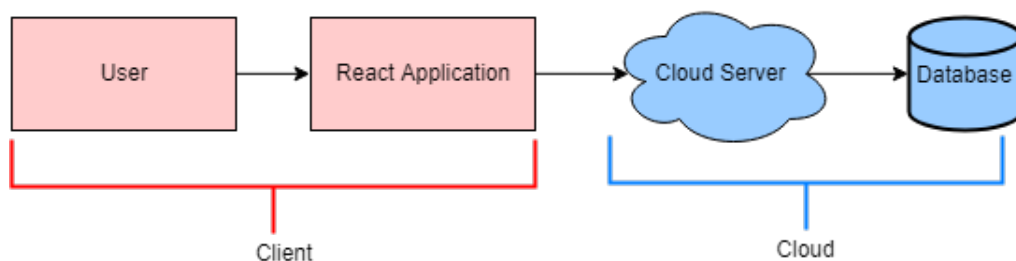
A distributed database system is a system that is spread over different sites. These locations don't share physical components and are accessed globally by all users. Although it should be noted that for the user it must look as if it were one single database. It is classified as Homogeneous Database and Heterogeneous Database.



**Figure 1.9:** Distributed Database System

### 1.3 Blood Bank Inventory Management

In this project, we are using a centralized database system that helps keep a single database. The data first goes from the blood banks to the front end application developed through react framework. After basic validation tests, it goes through cloud-based support on Amazon Web Services and finally it is passed to the database.



**Figure 1.10:** Blood Bank Inventory Management

### 1.4 Importance of shift

There has been a trend in the increase in the gap of the demand-supply ratio of blood and organ donation. According to the articles and reports, this change in the trend is attributed to the high accounting management in the blood banks and selling of quints of blood in the red market. To fraud a few quints of donated blood and then changing the given blood by taking more blood from the same donor but not registering them,

such people manage to sell blood.

There is no security of users personal data. Audits are often slow tedious and have errors. In a survey, it shows that there is an 0.2% error rate of wrong blood. While 0.2 may not seem a large number out of every 100 people, 2 people die just because they could not receive the blood they needed. These are present because of the manual system and can be eliminated with the use of a digital system.

## **1.5 Accomplishment**

This project encourages the use of Human-Computer-Interface. The user interface and the user experience with the system have seen a positive response. Along with increased precision, less storage space and ease of audits, we have created a simple and easy to use interface keeping the users in minds.

## **CHAPTER 2**

### **LITERATURE SURVEY**

#### **2.1 Blood for sale: India's illegal 'red market'**

According to the World Health Organization, which states the standards for blood reserves to be 1% for every country states that there is a chronic short supply in India. India needs 12 million blood units annually instead ends up a collection of only 9 million units only. This accounts for a 25% deficit. A trend shows that this deficit percent rises to 50% leading to a spur of desperate patients. The lack of a central accounting along with a reluctance to donate blood accounts for the shortage. In 1996 Supreme Court running banned unlicensed blood banks but a little has changed from those times. Demands still outstrip supply. The illicit market has simply moved underground. A large sum of Blood is supplied to these markets because records are accessible to manipulate and change to everyone and not secure. A digital inventory will ensure that this accounting mismanagement does not happen. Anand (2015)

#### **2.2 Database Lifecycle**

The database life cycle comprises of six phases, each of which is essential to the making of a stable database and organize information curated to individual needs. Phase 1 is the Initial study which is to analyze, define the problem and list the objective scope and boundaries of the problem. Phase 2 is the Database Design phase, this is loosely related to designing the larger architecture, this is to figure out where the data will fit in the grander scheme of the project. Phase 3 is the Implementation and Loading of the database this phase includes the creation of tables, attributes, views, domains, indexes and many other constraints including security and performance guidelines. Phase 4 is the Testing and Evaluation here the project is run and tested, all current errors and cases

are handled. Phase 5 is Maintenance and Evolution this includes developing means for backup and recovery, Active maintenance of new and old users. Millar (2018)

## **2.3 Concepts of Database Management Systems**

Database Management System is a platform for a lot of functions and services that it has to perform, which guarantee the integrity and consistency of the data that is provided. This inventory ensures that we provide management for permanent storage. The inside schema decides the process of storage management in the physical hardware of the system. The full task list of any database management system includes the basic ability to retrieve, update, delete and store data into respective databases. Our system also accepts external, internal and conceptual schemas with all associated mappings like all other database management systems. Every database management system must provide catalogue management to the user. The end users request for database access are transmitted, we ensure that there is no unauthorized access in this process. A part of our inventory management is focused on backup and recovery as it is an essential feature for every database management system. All database management systems provide an interface for applications to interact with them, we have used NoSQL. Using queries that have Data definition and data manipulation parts to use and access the database we have ensured that the current system has all the functionalities of a standard database management system. Naik (2013)

## **2.4 The Ultimate Guide to SDLC**

Software Development Lifecycle is a series of definitive set of tasks performed to make software. It is followed by every development team that is making a software solution. There are several models that follow but with five essential steps. These are Planning, Defining, Designing, Testing, and Deploying. Many times a seven-step model is seen that includes recovery and maintenance in addition to the above five-step process. A variation of these steps is seen in models like the Waterfall model, V model, Incremental model, Iterative model, Spiral model, and the Big bang modal. Jr. (2012)

## **2.5 ReactJS**

React is a complete javascript library that is used to make user interfaces. It is used to make the frontend of our application. Developed by Facebook, react is a component-based system and thus has more modules and concepts of objects. This factor makes react easier to debug and makes the code more predictable. ReactJS (2019) NodeJS (2019)

## **2.6 What Happens to Donated Blood**

This is an article that shows the insight upon how blood donation systems work and the requirements of an inventory. There are a total of six steps contributing to blood being successfully donated and used. First, you arrive at the donation center and your physical and medical checkup is cleared then health history is also noted your donated sample is made to undergo preliminary screening. Then in the second step blood is taken to the processing center where whole blood donations are spun in centrifuges. Due to spinning, we get three components that are Red cells, Platelets and Plasma. Each component is packaged as a “unit”. Alongside the second step, the third step will take place a dozen tests will be performed on your blood and even if one of them turns out to be positive your blood is discarded. The test results arrive in the fourth step and the units are stored according to the prescribed conditions. Step five is the distribution of blood where it is necessary. Finally, the blood is transfused in the patient. RedCross (2019)

## **2.7 MongoDB**

Mongo DB allows an easy organization, use and enrichment of data anywhere. It provides complete flexibility to the developer and seamless data migration. Mongo DB is a document database with a lot of scalability. The data is retrieved, manipulated and entered through the use of queries. Since the model maps application in application code it makes out retrieval and searching quick and efficient. It is a distributed database

at its core so it is easier for horizontal searching. Moreover, Mongo DB is free to use. MongoDB (2019)

## **2.8 Foundations of Software Testing**

Software Testing checks whether the actual results match the expected outcomes thereby ensuring that the application is defect free. It can also be called as Verification of Application Under Test. there are three broad classifications of testing mainly Functional testing, Non-functional testing, and Maintenance testing. There are seven principles to software testing which are keenly observed and implemented during the course. Software Testing Life Cycle goes hand in hand in with the development lifecycle and therefore is a part of the waterfall and the v model. This includes the Requirement analysis, Test Planning, Test case Development, Environment Setup, Test Execution, and Test Cycle Closure. Each of these stages has entry criteria and exit criteria. Testing can be done manually and automatically depending upon the application and the test cases it covers. Apache Jmeter is used for automatic testing. It is a pure Java open source. It is mainly used on performance testing and to analyze all tests. Jmeter simulates a bunch of user requests to the server that is targeted and returns information by analyzing these results.

## **2.9 Box Detection Algorithm**

This algorithm enables us to detect any table format and extract the data from it using simple morphological operations. To do this for any image we first use preprocessing, in this project preprocessing means to extract the location of where our box is located. Then we can subject this image through the algorithm. To extract each cell one by one and detect the numbers we will apply the Machine Learning model to do the recognition. First, we convert the image to grayscale and then we perform thresholding and inversion. We then have two kernels operate on the same image. One is used to detect the horizontal lines the other is used to detect vertical lines. These two complete our basic morphological detection and the end result is that we have two images. We then



add these two images to recreate the box, this removes all the noise present in the image preventing false box detection. After finding all boxes we will sort these using containers then crop them and put them in a folder or subject them to further processes. Vyas (2018)

## **2.10 Offline Handwriting Recognition Using a Generic Algorithm**

The handwritten document is scanned as input to obtain individual characters. These are written in a text file and are later read back. After this, they are passed back to Neural Network (NN). The scanned grayscale image is read into a matrix which is converted to a monochromatic image matrix with pixel values of 0 for black and 255 for white points. A process of row-wise searching is done from the point (0, 0) to find out the first black point. This is the upper point. After this point, all points are connected to this black value of 999. Post this the values having 999 are connected to recreate the word. The lowest point is now found out and the rightmost and leftmost points are obtained. Then we determined that no letters are missing. Word is searched for the number of cuts on a row-wise and column-wise manner. These are stitched again to recreate the word. Mathur (2008)

# **CHAPTER 3**

## **SYSTEM ANALYSIS**

### **3.1 Problem Definition**

Blood Bank Inventory management systems are challenging but necessary changes that need to be shifted to the digital platform. There is a need for setting a standard procedure to acquire and store blood in every blood bank. There is no effective digital blood banks management system that is currently being used in the blood banks. Whereas it is necessary to make this system sustainable and reliable. Proto.io (2017)

### **3.2 Proposed System**

Through this project, we focus on helping blood bank employees and make their tasks easy while giving them analytics about their blood bank, patients as well as the clients which are the hospitals, thereby helping the process become faster and secure. We use a massive amount of real data to design our system. This is provided by the rotary blood bank including the test cases that come along with the system.

The donor's data is analyzed carefully and upon different parameters, predictions are given to various centers. We have used all modern technologies in order to make sure easy maintenance and usage. We have made the system with a simple design keeping in mind that the end user may not be well versed in computers.

This system plans to shift the current system to a digital platform from the manual records that are maintained. Intelligent data entry and retrieval supported by decision-making system for an integrated blood bank management is a web-based application that is scalable and sustainable to the current and future needs.

### **3.2.1 Advantages of Proposed System**

The employees at the blood bank through this system will be able to perform their operational requirements more easily and efficiently thereby increasing their overall productivity. At the same time it will also bring a huge impact to the society and the patients who actually need blood, because it's about saving a life through technology which is an enormous break through towards humanity.

Considering the real time requirements, enormous data and complexities of the system, we used a client server architecture where the systems deployed at the blood bank will be directly communication with a centralized server which will can be deployed on Amazon Web Services. On this server our Mongo Server will also be up and running all the time to fulfill the client's need.

## **3.3 Requirements**

### **3.3.1 Hardware Requirements**

CPU - Dual Core Intel Core i3 or Higher.

Ram - 2GB or Higher.

HDD - 30GB or Higher

Input Devices - Keyboard, Mouse.

Monitor - 15" (1366\*768) or Higher

### **3.3.2 Software Requirements**

#### **a. Server**

Ubuntu/Linux

MongoDB

Node

Express

React  
TypeScript  
VS Code

#### **b. Client**

Windows 8 or Later

### **3.4 Issues in existing methodology**

In the current system, records are maintained in registers, these registers are organized by the years, monthly donor data in huge racks and other records. This system is not only redundant and slow but also consumes a lot of space and makes it tough to maintain huge amounts of data. The current system cannot sustain the urgency of blood donation and management in the current times. An ideal Blood Bank system must be quick and efficient. The importance of time is critical and crucial in a blood bank system, such systems must be quick and efficient.

The existing model does not cover the basic ACID properties of a database management system along with management to the Purchased, Used and Stock (PUS) inventory management. Inventories in modern times are expected to maintain and record the blood samples. This process is also maintained in registers along with the test results of every sample for various tests that are conducted in the blood sample, thus making it more prominent to more manual errors. There are serious consequences to the delay in this process.

Finally, the Input and Output modules are the means to handle the influx of blood, classify, store and give it accurately to the requested hospital. Such important modules should not have any manual errors. Overall the current system is slow, tedious and gives more room to error.

## **3.5 New methodology**

### **3.5.1 Gathering data from donors**

Depending upon the needs we need to feed the data manually to the system. To make this easier we have autofill feature that sets some fields. This is presented as the Input Module in the system. The input module provides the system with the donor data that is necessary. This system can validate the donor information as a first part of screening the system.

### **3.5.2 Process partition**

The donated blood is taken through several procedures and each of them should be documented with care. We have established several simple forms that have automatically filling process to make and document different tests. if the results of any one test are positive then the blood is disposed. The blood is split into three categories each is stored under different conditions for different amounts of time. This is documented along with the date in which the blood is stored. This is achieved through the purchase, stock and use model in the current system.

### **3.5.3 Analytics**

The analytics section makes predictions based on the fundamentals of probabilities that play in an area. There are several different key index points upon which these are selected. This provides an insight into the bigger picture and can be used for analysis.

# CHAPTER 4

## SYSTEM DESIGN

### 4.1 Physical Design

#### 4.1.1 User Interface Design

Our objective with user interface design was to provide a simplistic design with meaningful user experience. For which we used Google's Material Design system. Material Design has physical edges and surfaces. Shadows and seams which provide meaning about how we touch and interact with things. They state that its design is based on ink and paper but the implementation takes place in an advanced manner. Material Design is used as it's very scalable in case we wish to expand on the mobile platform.

The design language will remain the same across all the devices and provides a more friendly and user-friendly approach. This means that we can save time because we don't have to develop our own visual design and can avoid common design pitfalls. Wikipedia (2018)

#### Wireframing

Through wireframing, we can approach with an effective and quick way to identify usability issues early on the design process. Website framework can also be known as a screen blueprint or a page schematic which is a visual guide representing the skeletal framework for the website. It's done for the purpose of arranging elements to accomplish a particular purpose.

As per the given information by the Blood Bank, the people who will be operating on the computer their purpose will be solely to enter data. These individuals are not very educated and have minimal computer knowledge. Also, the individuals are changed weekly and monthly so it's not necessary that the worker's job duration is very

Wikipedia (2019) The first wireframe shows the view for the various modules in-

**Figure 4.1: Wireframe 1**

NAME OF BLOOD BANK		LOGO		HEADING FOR OPTION SELECTED											
ICON	DONOR LOOKUP														
ICON	DONOR SEARCH														
ICON	DASHBOARD/ANALYTICS														
ICON	PURCHASE STOCK UNIT <b>v</b>														
ICON	INPUTS RECORD <b>v</b>														
ICON	OUTPUT RECORD <b>v</b>														
	OPTION 1														
	OPTION 2														
	OPTION 3														
	OPTION 4														
	OPTION 5														

### VIEW FOR ADD DONOR/UPDATE DONOR

FORM WITH REQUIRED INPUT IN THE FIRST HALF

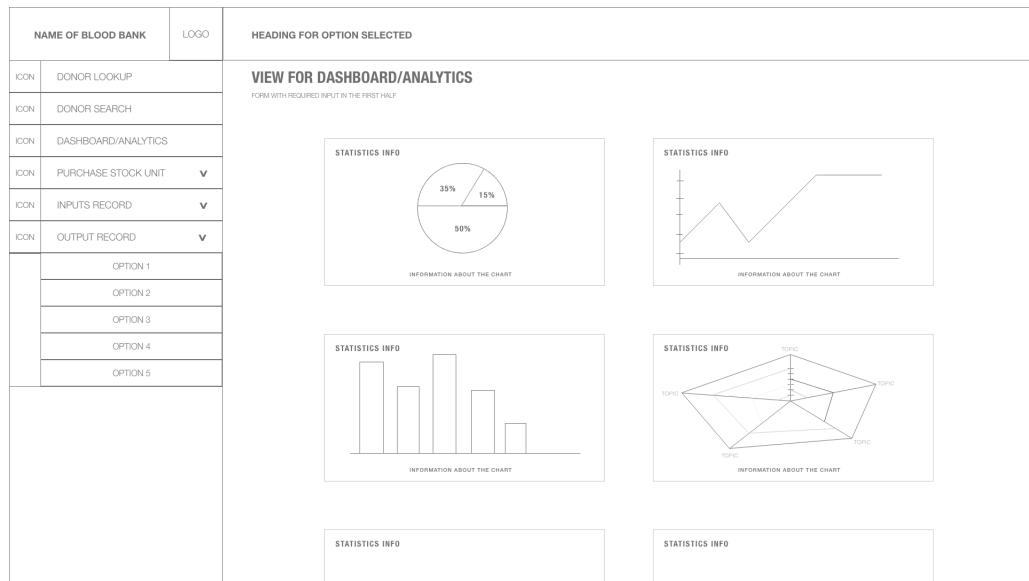
INPUT: <input type="text"/>	INPUT: <input type="text"/>	INPUT: <input type="text"/>	INPUT: <input type="text"/>	INPUT: <input type="text"/>	INPUT: <input type="text"/>
INPUT: <input type="text"/>	INPUT: <input type="text"/>	INPUT: <input type="text"/>	INPUT: <input type="text"/>	INPUT: <input type="text"/>	INPUT: <input type="text"/>
INPUT: <input type="text"/>	INPUT: <input type="text"/>	INPUT: <input type="text"/>	INPUT: <input type="text"/>	INPUT: <input type="text"/>	INPUT: <input type="text"/>
INPUT: <input type="text"/>	INPUT: <input type="text"/>	INPUT: <input type="text"/>	INPUT: <input type="text"/>	INPUT: <input type="text"/>	INPUT: <input type="text"/>
INPUT: <input type="text"/>	INPUT: <input type="text"/>	INPUT: <input type="text"/>	INPUT: <input type="text"/>	INPUT: <input type="text"/>	INPUT: <input type="text"/>
INPUT: <input type="text"/>	INPUT: <input type="text"/>	INPUT: <input type="text"/>	INPUT: <input type="text"/>	INPUT: <input type="text"/>	INPUT: <input type="text"/>
INPUT: <input type="text"/>	INPUT: <input type="text"/>	INPUT: <input type="text"/>	INPUT: <input type="text"/>	INPUT: <input type="text"/>	INPUT: <input type="text"/>
INPUT: <input type="text"/>	INPUT: <input type="text"/>	INPUT: <input type="text"/>	INPUT: <input type="text"/>	INPUT: <input type="text"/>	INPUT: <input type="text"/>
INPUT: <input type="text"/>	INPUT: <input type="text"/>	INPUT: <input type="text"/>	INPUT: <input type="text"/>	INPUT: <input type="text"/>	INPUT: <input type="text"/>
INPUT: <input type="text"/>	INPUT: <input type="text"/>	INPUT: <input type="text"/>	INPUT: <input type="text"/>	INPUT: <input type="text"/>	INPUT: <input type="text"/>
INPUT: <input type="text"/>	INPUT: <input type="text"/>	INPUT: <input type="text"/>	INPUT: <input type="text"/>	INPUT: <input type="text"/>	INPUT: <input type="text"/>
INPUT: <input type="text"/>	INPUT: <input type="text"/>	INPUT: <input type="text"/>	INPUT: <input type="text"/>	INPUT: <input type="text"/>	INPUT: <input type="text"/>
INPUT: <input type="text"/>	INPUT: <input type="text"/>	INPUT: <input type="text"/>	INPUT: <input type="text"/>	INPUT: <input type="text"/>	INPUT: <input type="text"/>
INPUT: <input type="text"/>	INPUT: <input type="text"/>	INPUT: <input type="text"/>	INPUT: <input type="text"/>	INPUT: <input type="text"/>	INPUT: <input type="text"/>

SUBMIT

CLEAR

**Figure 4.2: Wireframe 2**

19



**Figure 4.3: Wireframe 3**

The third wireframe showcases the dashboard/analytics page of the Blood Bank which will be used to view various Key Performance Indicators (KPI)(s) of the Blood Bank. This module incorporates various charts and graphs and the objective was to provide a user interface in which an individual could take a glance of the whole system with just a click of the button.

#### 4.1.2 Database Design

We were given real register record pages courtesy of Rotary Blood Bank. Using these records, identification of database tables and their fields was done. The objective was to reduce redundancy and remove any fields which can be automatically filled such as calculation of age from the Date of birth (DOB) of the donor.

Looking at the table and their fields, they have a tendency to change over time with newer fields as per the needs of a Blood Bank. Using a SQL database could have been a very tedious and time taking work, also resolving issues would be very difficult in such a complex system.



## NoSQL Database

NoSQL which stands for "Not only SQL" is a non-relational database that allows for storage and retrieval of data. NoSQL database includes simplicity of style, less complicated horizontal scaling to clusters of machines and finer management over handiness. the information structures utilized by NoSQL databases are totally different from those utilized by default in relative databases that makes some operations quicker in NoSQL. The quality of a given NoSQL information depends on the matter it ought to solve. Knowledge structures utilized by NoSQL databases are typically conjointly viewed as additional versatile than on-line database tables. The choice of NoSQL database was MongoDB because of it's versatile nature and better community. Ayusharma0698 (2018)

S.No.	NoSQL	SQL
1.	Non relational database.	Relational database.
2.	Document based.	Table based.
3.	Dynamic schema for unstructured data.	Predefined schema for structured data.
4.	Horizontally scalable.	Vertically scalable.

Table 4.1: NoSQL vs SQL

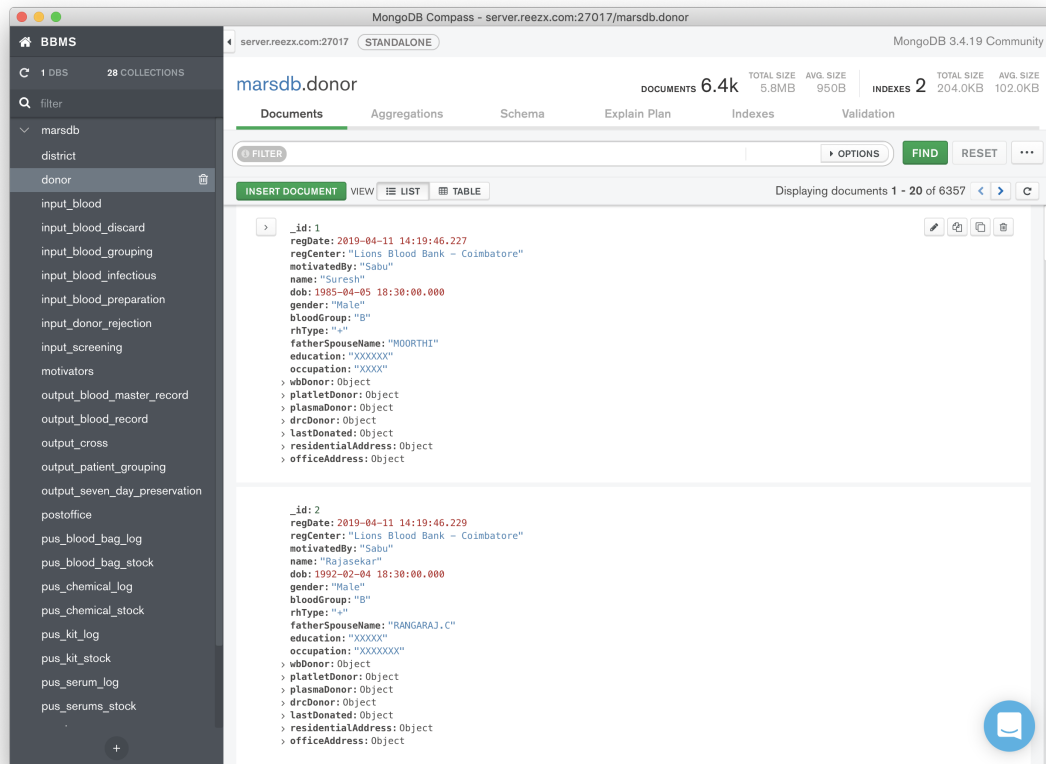
## Creation of collections

In MongoDB a table is called a collection. Using the given record we identified each column for the database and created an excel sheet out of it. These excel sheets were then validated by the personnel at the Blood Bank, upon their validation, we proceeded with the creation of the actual collection for the database.



	DONOR CONSENT FORM <b>ROTARY BLOOD BANK - NANGANALLUR</b> DESIGNATED BLOOD BANK (Approved by Tamil Nadu State Blood Transfusion Council) No. 128, Medavakkam Main Road, Ullagaram-Nanganallur, Chennai - 600 091 License No : 306 Dated : 15-10-2009	<b>Blood Group</b>    <b>Date</b>																										
<b>DONOR DETAILS</b> NAME : ..... DOB: ..... AGE: ..... SEX: M / F FATHER NAME / SPOUSE NAME : ..... EDUCATION ..... OCCUPATION: ..... PHONE / CELL: ..... EMAIL ID : ..... <div style="text-align: center;"><b>ADDRESS (RESIDENCE / OFFICE)</b></div> BUILDING NAME : ..... LAND MARK : ..... DOOR NO : ..... STREET OR ROAD ..... AREA : ..... VILLAGE / TOWN / CITY ..... TALUK ..... DISTRICT : ..... RECENTLY BLOOD DONATED : YES / NO. LBD DT: ..... WILLING TO DONATE BLOOD REGULARLY : YES / NO																												
<b>BLOOD DONOR QUESTIONNAIRE</b> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr><td>1. Are you at present in good health ?</td><td style="text-align: right;">Yes / No</td></tr> <tr><td>2. Are you on fasting during the last 4 hours? (when did you eat last?)</td><td style="text-align: right;">Yes / No</td></tr> <tr><td>3. Are you taking any medicine ?</td><td style="text-align: right;">Yes / No</td></tr> <tr><td>4. Have you been vaccinated or immunized recently ?</td><td style="text-align: right;">Yes / No</td></tr> <tr><td>5. Have you ever suffered an epileptic fit, convulsions or mental disorder ?</td><td style="text-align: right;">Yes / No</td></tr> <tr><td>6. Have you ever had Jaundice or Hepatitis ?</td><td style="text-align: right;">Yes / No</td></tr> <tr><td>7. Have you ever been positive for HBV or HCV ?</td><td style="text-align: right;">Yes / No</td></tr> <tr><td>8. Have you been in contact with a person suffering from Jaundice (Hepatitis) during the last 6 months ?</td><td style="text-align: right;">Yes / No</td></tr> <tr><td>9. Do you know about AIDS ?</td><td style="text-align: right;">Yes / No</td></tr> <tr><td>10. Have you (if male) had sex with another man ?</td><td style="text-align: right;">Yes / No</td></tr> <tr><td>11. Have you had unsafe sex with an individual at increased risk for AIDS ?</td><td style="text-align: right;">Yes / No</td></tr> <tr><td>12. Have you lost significant weight in last 6 months ?</td><td style="text-align: right;">Yes / No</td></tr> <tr><td>13. Have you ever been positive for HIV ?</td><td style="text-align: right;">Yes / No</td></tr> </table>			1. Are you at present in good health ?	Yes / No	2. Are you on fasting during the last 4 hours? (when did you eat last?)	Yes / No	3. Are you taking any medicine ?	Yes / No	4. Have you been vaccinated or immunized recently ?	Yes / No	5. Have you ever suffered an epileptic fit, convulsions or mental disorder ?	Yes / No	6. Have you ever had Jaundice or Hepatitis ?	Yes / No	7. Have you ever been positive for HBV or HCV ?	Yes / No	8. Have you been in contact with a person suffering from Jaundice (Hepatitis) during the last 6 months ?	Yes / No	9. Do you know about AIDS ?	Yes / No	10. Have you (if male) had sex with another man ?	Yes / No	11. Have you had unsafe sex with an individual at increased risk for AIDS ?	Yes / No	12. Have you lost significant weight in last 6 months ?	Yes / No	13. Have you ever been positive for HIV ?	Yes / No
1. Are you at present in good health ?	Yes / No																											
2. Are you on fasting during the last 4 hours? (when did you eat last?)	Yes / No																											
3. Are you taking any medicine ?	Yes / No																											
4. Have you been vaccinated or immunized recently ?	Yes / No																											
5. Have you ever suffered an epileptic fit, convulsions or mental disorder ?	Yes / No																											
6. Have you ever had Jaundice or Hepatitis ?	Yes / No																											
7. Have you ever been positive for HBV or HCV ?	Yes / No																											
8. Have you been in contact with a person suffering from Jaundice (Hepatitis) during the last 6 months ?	Yes / No																											
9. Do you know about AIDS ?	Yes / No																											
10. Have you (if male) had sex with another man ?	Yes / No																											
11. Have you had unsafe sex with an individual at increased risk for AIDS ?	Yes / No																											
12. Have you lost significant weight in last 6 months ?	Yes / No																											
13. Have you ever been positive for HIV ?	Yes / No																											
<p style="text-align: center;"><b>இந்தத்தானம் செய்பவரின் ஒப்புதல்</b></p> <p>நான் அறிந்தவன் அல்லது நான் இந்த விஷயங்களை, முற்றிலும் உண்மை எனவும், எனது மனப்பான்மை அறிவு தெரிவு என, கொடுக்கப்பட்டுள்ளது என உறுதிப்படுத்துகிறேன். இந்த படிவத்தில் நான்கு பக்கத்திலும் மேல்க்கப்பட்டுள்ள கேள்விகள் எனது பரிசுத்தமயத்தானமும், இந்தத்தானம் செய்து கொள்வோமீன் மனதுகொடுத்தவனாகவும் மேல்க்கப்பட்டு என அறிவிக்கிறது. நான் கீழ்க்கண்ட ஒப்புதல் அளிக்கிறேன்.</p> <ol style="list-style-type: none"> <li>இந்தத்தானம் செய்பவருக்கான நான் எடுத்திருக்கின்ற வேறுதலில் பெற்றிருக்கின்ற எண்ணம்.</li> <li>எந்தவித கைமாறும், வெகுமதியோ இந்தத்தானம் செய்வதற்காக நான் எதிர்பார்க்கவில்லை.</li> <li>என் விருப்பப்படி இந்தத்தானம் செய்வதற்கு ஒப்புதல் அளிக்கிறேன். இதன் பரிசுத்தமய நிபந்தனையை நான்.</li> <li>இந்தத்தானம் செய்வதன் மூலம் நேரத்திற்கு எந்தவிதமான கைதீமைகளும் இயக்கவாய்ப்புகள் எனது உறுதிப்படுத்துகிறேன்.</li> <li>இந்தத்தானம் செய்வதின் அல்ல மூலம் நேரத்திற்கு அல்லதுவரின் கைதீமைகளும் உண்டாகாது.</li> </ol> <p style="text-align: right;">_____</p> <p style="text-align: center;">இந்தத்தானம் செய்பவரின் கையொப்பம்</p>	<p style="text-align: center;"><b>DONOR CONSENT</b></p> <p>The Medical history and information furnished by me are true and correct to the best of my knowledge.</p> <p>I understand that the question asked are for my protection as well as to protect the recipient of my blood. I hereby assure that</p> <ol style="list-style-type: none"> <li>I am not motivated for gift of any form to donate blood.</li> <li>I am donating blood without expecting any reward in return.</li> <li>I have voluntarily come forward with out any compulsion to donate blood and with my clear self decision I am donating Blood.</li> <li>I agree not to ride / drive for 1 hour after donating Blood.</li> <li>I agree to be available for observation (½ an hour) after Donating Blood.</li> </ol> <p style="text-align: right;">_____</p> <p style="text-align: center;">Date : _____ Donor's Signature</p>																											

Figure 4.5: Donor Consent Form



**Figure 4.6:** MongoDB Compass showing various collections and documents

The data is stored in the form of JavaScript Object Notation (JSON), in MongoDB. All the data in collections is known as documents and unlike SQL we can have independent fields.

### 4.1.3 Modules

#### a. Search

The donor's info is stored in registers and looking for donors when needed is very time taking. By storing the donor's info and details into a centralized database we can easily search using donor id or donor details. If upon search a donor is not found we can add the donor using the same details as well, which is very convenient and fast for the blood bank. Figure 4.7, 4.8 represents the data models for the user's address and blood bank document stored in the database.

Address <span>✎ A-Z ⊕</span>		
<i>Address Object Inside Collection</i>		
<b>_id</b>	{ objectId }	ID
<b>house</b>	{ string }	House / Flat Number
<b>addressLineOne</b>	{ string }	Address Line 1
<b>addressLineTwo</b>	{ string }	Address Line 2
<b>Locality</b>	{ string }	Locality
<b>pincode</b>	{ int }	Pincode
<b>city</b>	{ string }	City
<b>state</b>	{ string }	State

**Figure 4.7:** Address Data Model

Blood Bank <span>✎ A-Z ⊕</span>		
<i>An Institute to store Blood</i>		
<b>_id</b>	{ objectId }	ID
<b>name</b>	{ string }	Hospital Name
<b>address</b>	{ object }	Address
<b>consumables</b>	{ array }	List of Consumables
<b>staffs</b>	{ array }	List of Staff in the Hospital

**Figure 4.8:** Blood Bank Data Model

## b. Lookup

The donor's info is stored in registers and looking for donors when needed is very time taking. By storing the donor's info and details into a centralized database we can easily search using advanced query. In this module, the implementation to print the data of the query result is also a very convenient way to present the lists of donors. The advanced query include - 'Blood group, Rh factor, Gender, Donor type, Area, etc'. Figure 4.9 represents the data model for the user document stored in the database.

User		
<i>A User is any Person that can login into the System</i>		
<b>_id</b>	{ string }	Username
<b>password</b>	{ string }	Password
<b>firstName</b>	{ string }	First Name
<b>middleName</b>	{ string }	Middle Name
<b>lastName</b>	{ string }	Last Name
<b>DOB</b>	{ date }	Date of Birth
<b>contact</b>	{ long }	Moble Number
<b>email</b>	{ string }	Email ID
<b>address</b>	{ object }	Address
<b>roles</b>	{ array }	donor   blood_bank_staff   admin
<b>institute</b>	{ object }	Name of Blood Bank in which this person Works




**Figure 4.9:** User Data Model

### c. Input




This module is to maintain details of blood donated to the blood bank by donors and the various tests that are carried out after a donation, prior to storing the blood. A lot of registers and logs are to be maintained for the same which is a very tiring and tedious work to do and auditing for the same is also not possible. Through our system and database, everything can be simplified and easy to manage and view everything. The goal here is to reduce all the manual entry and simplify the process of data entry and retrieval. Figure 4.10 - 4.13 represents various data models used for this module's document storage in the database.

Blood Discard Records		
<i>Record of Discarded Blood</i>		
<b>_id</b>	{ objectId }	ID
<b>discardDate</b>	{ date }	Discard Date
<b>collectionDate</b>	{ date }	Date of Collection
<b>donorId</b>	{ objectId }	Donor ID
<b>bagNo</b>	{ int }	Bag Number
<b>reason</b>	{ string }	Reason for Discarding Blood
<b>technicianName</b>	{ string }	Name of Technician




**Figure 4.10:** Blood Discard Data Model

Blood Donation Record   		
<b>_id</b>	{ objectId }	ID
<b>timestamp</b>	{ date }	Time Stamp
<b>donor</b>	{ objectId }	Donor ID
<b>staff</b>	{ string }	Staff ID
<b>bloodType</b>	{ objectId }	Blood Type
<b>bloodBank</b>	{ objectId }	ID of Blood Bank
<b>consumables</b>	{ array }	List of Consumables Used

**Figure 4.11:** Blood Donation Data Model

Blood Input Record   		
<i>Record of Blood Collected</i>		
<b>_id</b>	{ objectId }	ID
<b>quantity</b>	{ int }	Quantity of Blood Collected
<b>type</b>	{ string }	Type of Blood
<b>hospitalName</b>	{ string }	Name of Hospital
<b>issuedDate</b>	{ date }	Date of Issue
<b>collectionDate</b>	{ date }	Date of Collection
<b>expiryDate</b>	{ date }	Date of Expiry

**Figure 4.12:** Blood Input Record Data Model

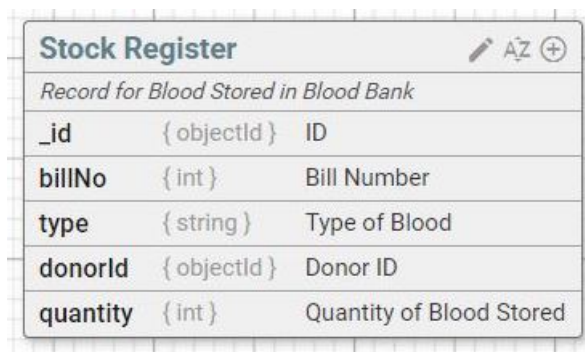
Donor Rejection Record   		
<i>Record for all donor that are rejected</i>		
<b>_id</b>	{ objectId }	ID
<b>place</b>	{ string }	Place of Rejection
<b>name</b>	{ string }	Name of Donor
<b>date</b>	{ date }	Date of Rejection
<b>address</b>	{ string }	address of donor
<b>phone</b>	{ long }	Contact Number
<b>duration</b>	{ string }	"PERMANENTLY"   "TEMPORARILY"
<b>reason</b>	{ string }	Reason for Rejection

**Figure 4.13:** Donor Rejection Data Model



#### d. Output

This module is to maintain details of blood issued by the blood bank to patients and the various tests that are carried out on the stored the blood, prior to an issue. A lot of registers and logs are to be maintained for the same which is a very tiring and tedious work to do and auditing for the same is also not possible. Through our system and database, everything can be simplified and easy to manage and view everything. The goal here is to reduce all the manual entry and simplify the process of data entry and retrieval. Figure 4.14 represents the stock register data models used for the document storage in the database.

The diagram shows a 'Stock Register' data model. It has a title bar with the text 'Stock Register' and icons for edit, sort, and add. Below the title bar is a subtitle 'Record for Blood Stored in Blood Bank'. The main body of the model is a table with six rows, each representing a field in the document. The fields are: '\_id' (type { objectId }, description ID), 'billNo' (type { int }, description Bill Number), 'type' (type { string }, description Type of Blood), 'donorId' (type { objectId }, description Donor ID), and 'quantity' (type { int }, description Quantity of Blood Stored).

Stock Register		
Record for Blood Stored in Blood Bank		
_id	{ objectId }	ID
billNo	{ int }	Bill Number
type	{ string }	Type of Blood
donorId	{ objectId }	Donor ID
quantity	{ int }	Quantity of Blood Stored

**Figure 4.14:** Stock Register Data Model

#### e. PUS

This module is to maintain the stock status of various consumables used in a blood bank like a blood bag, needles, syringes, cotton, chemicals etc. There is no digital way of getting to know the exact quantity of all the items. Our system offers a method of inventory management in which we provide digitally all the present items available and offer interactive screens in which we can add/remove new items. Auto logging of all the transactions of adding and removal of items is done in the backend. Saving time and reducing overhead work. Figure 4.15 - 4.19 represents various data models used for this module's document storage in the database.



Blood Bags		
<i>Different kinds of Blood Bags are used to different types and different quantity of Blood</i>		
<b>_id</b>	{ objectId }	ID
<b>type</b>	{ string }	"DOUBLE_BLOOD_BAG", "PENTAL_BLOOD_BAG", "QUADRUPLE_BLOOD_BAG", "SINGLE_BLOOD_BAG", "TRANSFER_BLOOD_BAG", "TRIPLE_BLOOD_BAG"
<b>purchaseDate</b>	{ date }	Date of Purchase
<b>usedDate</b>	{ date }	Date of Use
<b>manufacturingDate</b>	{ date }	Date of Manufacturing
<b>manufacturerName</b>	{ string }	Name of Manufacturer
<b>purchasedFrom</b>	{ string }	Name of Supplier
<b>billNo</b>	{ int }	Bill Number
<b>quantity</b>	{ int }	Available Quantity
<b>batchNo</b>	{ int }	Batch Number
<b>expiryDate</b>	{ date }	Date of Expiry
<b>status</b>	{ string }	"PURCHASED"   "USED"   "DISCARDED"
<b>donorId</b>	{ objectId }	Donor ID
<b>technicianName</b>	{ string }	Name of Technician

**Figure 4.15:** Blood Bags Data Model

Consumables		
<i>Consumables are the materials used to draw, preserve or test Blood</i>		
<b>_id</b>	{ objectId }	ID
<b>name</b>	{ string }	Name of Consumable
<b>count</b>	{ int }	Total Item in Inventory

**Figure 4.16:** Consumables Data Model

Chemical		
Chemical are used to perform test on Patient's Blood		
_id	{ objectId }	ID
type	{ string }	"COPPER_SULPHATE", "DISTILLED_WATER", "LEISHMAN_STAIN", "NORMAL_SALINE"
purchaseDate	{ date }	Date of Purchase
usedDate	{ date }	Date of Use
manufacturingDate	{ string }	Date of Manufacturing
manufacturerName	{ string }	Name of Manufacturer
batchNo	{ int }	Batch Number
expiryDate	{ date }	Date of Expiry
status	{ string }	"PURCHASED"   "USED"   "DISCARDED"
testCount	{ int }	Number of Test Performed
missedRepeatedCount	{ int }	Number of Test Missed or Repeated
technicianName	{ string }	Name of Technician

**Figure 4.17: Chemicals Data Model**

Kits		
Kits are complete test set that are used to perform specific test		
_id	{ objectId }	ID
type	{ string }	"HBSAG_RAPID_KIT", "HCV_ELISA_KIT", "HCV_RAPID_KIT", "HIV_1_2_RAPID_KIT", "HIV_ELISA_KIT", "HSS_AG_ELISA_KIT", "VDRL_RAPID_KIT"
purchaseDate	{ date }	Date of Purchase
usedDate	{ date }	Date of Use
manufacturingDate	{ date }	Date of Manufacturing
manufacturerName	{ string }	Name of Manufacturer
batchNo	{ int }	Batch Number
expiryDate	{ date }	Date of Expiry
status	{ string }	"PURCHASED"   "USED"   "DISCARDED"
donorId	{ objectId }	Donor ID
testCount	{ int }	Number of Test Performed
technicianName	{ string }	Name of Technician

**Figure 4.18: Kits Data Model**

Serum		
<i>Serums are used to different type of bloods, to determine its type</i>		
<b>_id</b>	{ objectId }	ID
<b>type</b>	{ string }	"A_CELLS", "ANTI_A", "ANTI_A1_LLECTIN", "ANTI_AB", "ANTI_B", "ANTI_BOVINE_ALBUMIN", "ANTI_D", "ANTI_H_LLECTIN", "B_CELLS", "C_CELLS", "COOMBS", "O_CELLS"
<b>purchaseDate</b>	{ date }	Date of Purchase
<b>usedDate</b>	{ date }	Date of Use
<b>manufacturingDate</b>	{ date }	Date of Manufacturing
<b>manufacturerName</b>	{ string }	Manufacturer Name
<b>batchNo</b>	{ int }	Batch Number
<b>expiryDate</b>	{ date }	Date of Expiry
<b>status</b>	{ string }	"PURCHASED"   "USED"   "DISCARDED"
<b>use</b>	{ string }	"DONOR"   "PATIENT"
<b>donorId</b>	{ string }	Donor ID
<b>testCount</b>	{ int }	Number of test performed
<b>patientName</b>	{ string }	Name of Patient
<b>technicianName</b>	{ string }	Name of Technician

**Figure 4.19:** Serums Data Model

## 4.2 Logical Design

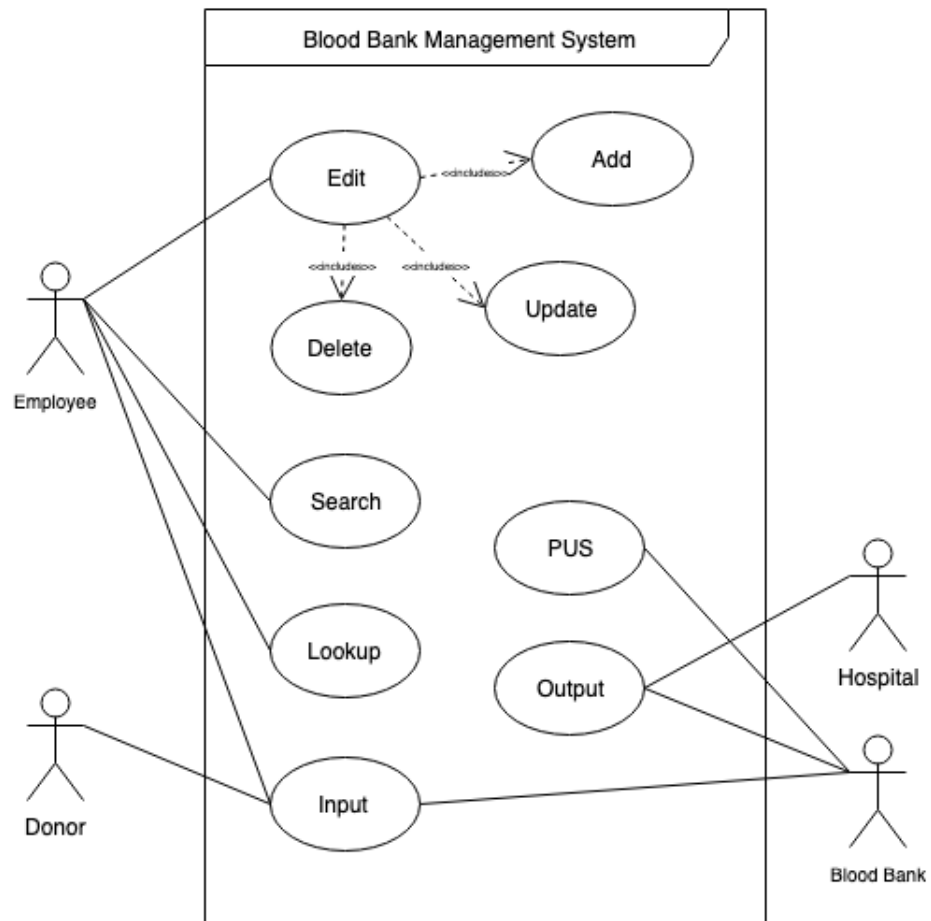
## 4.3 RESTful API

RESTful API is a piece of program which is an Application Program Interface (API) which uses Hyper Text Transfer Protocol (HTTP) to GET, PUT, DELETE and POST data. They are usually know as RESTful web service which is based on Representational State Transfer (REST) technology, which is an approach of architectural style for communication often used in web development.

### 4.3.1 UML Diagrams

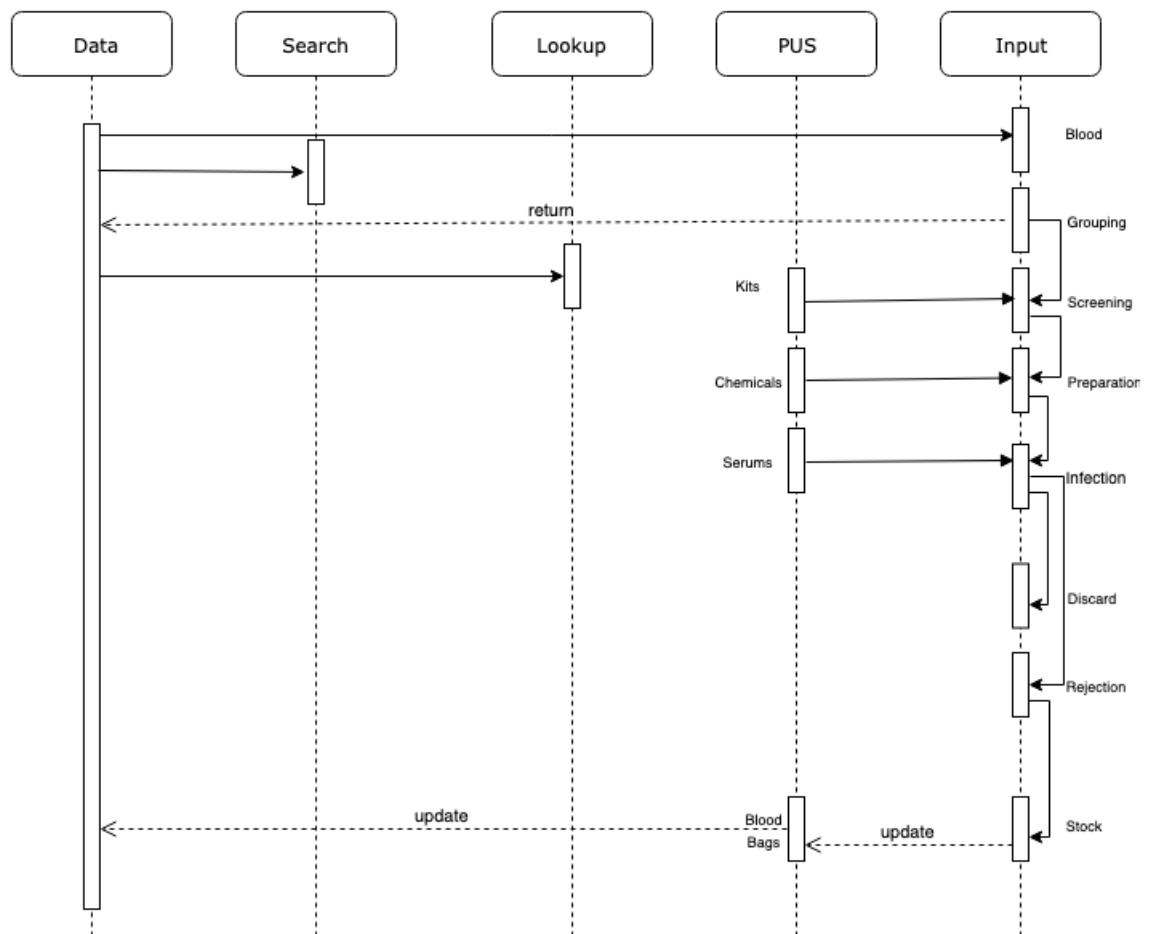
Ceta (2018)

#### a. Use Case Diagram



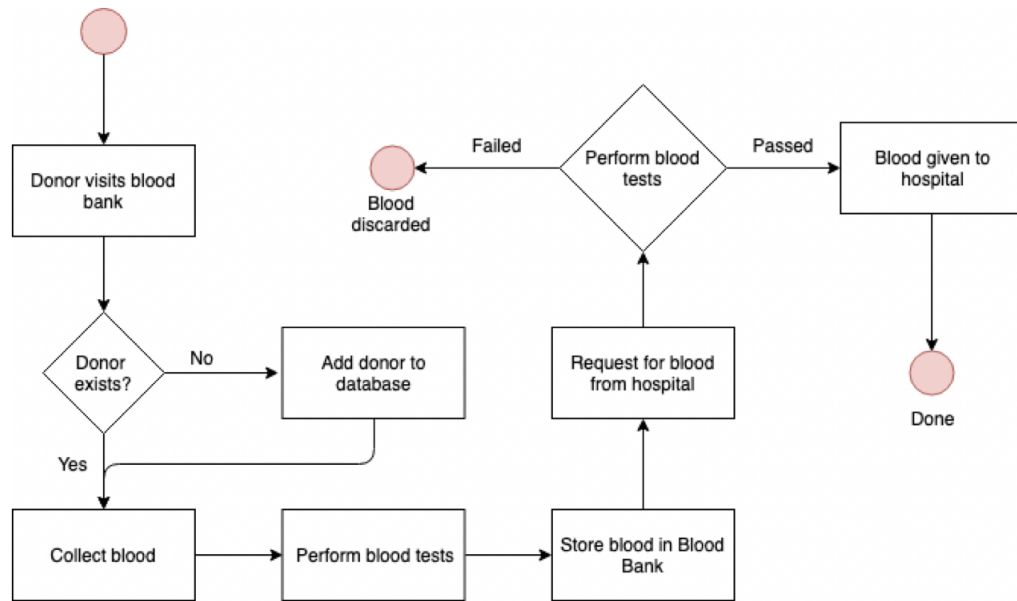
**Figure 4.20:** Use Case Diagram

## b. Sequence Diagram



**Figure 4.21:** Sequence Diagram

### c. Flow Diagram



**Figure 4.22:** Flow Diagram

# **CHAPTER 5**

## **TESTING**

### **5.1 Testing Objectives**

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a product. It provides a way to check the functionality of components, sub-assemblies, assemblies and/or a finished product. It is the process of exercising software with the intent of ensuring that the software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement. Graham (2018)

### **5.2 Types of tests**

#### **5.2.1 Unit Testing**

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application. It is done after the completion of an individual unit before integration. This is structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at the component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

### 5.2.2 Integration Testing

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event-driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

### 5.2.3 Functional Testing

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals. Functional testing is centered on the following items:

**Valid Input:** Identified classes of valid input must be accepted.

**Invalid Input:** Identified classes of invalid input must be rejected.

**Functions:** Identified functions must be exercised.

**Output:** Identified classes of application outputs must be exercised.

**Procedures:** Interfacing systems or procedures must be invoked

Organization and preparation of functional tests are focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify business process flows, data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

### 5.2.4 Acceptance Testing

User Acceptance Testing is a critical phase of any software and it requires significant participation by the end user. It also ensures that the system meets all the functional requirements.



### **5.2.5 System Testing**

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

### **5.2.6 White Box Testing**

White Box Testing is a testing in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is a purpose. It is used to test areas that cannot be reached from a black box level.

### **5.2.7 Black Box Testing**

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, like most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box you cannot “see” into it. The test provides inputs and responds to outputs without considering how the software works.

### **5.2.8 Alpha Testing**

In software development, the alpha test will be a test among the teams to confirm that your product works. Originally, the term alpha test meant the first phase of testing in a software development process. The first phase includes unit testing, component testing, and system testing. It also enables us to test the product on the lowest common denominator machines to make sure download times are acceptable and preloads work.

### **5.2.9 Beta Testing**

In software development, a beta test is the second phase of software testing in which a sampling of the intended audience tries the product out.

## **5.3 Tests Strategy and Approach**

As we can see the importance of testing in an application. The following test cases were kept in mind keeping the user at the center.

### **5.3.1 Test Objective**

1. Maintain the application's integrity and robustness.
2. Database transactions are complete and there is no data loss or redundancy.
3. Proper prompt and error messages are displayed.
4. The data from the forms are validated properly before storing in the database.

### **5.3.2 Features to be tested**

1. All the modules serve their purpose as intended.
2. All the forms have all the specified fields required for data collection.
3. Verify the entries are in correct format and there are no duplicate entries.
4. Navigation across the application is meaningful and states are properly maintained.

## **CHAPTER 6**

### **CONCLUSION**

This project is made with a sense of doing better towards the problems that are faced by the blood bank management inventories and helping society. A web application is made to ensure that this system is malleable and thus can sustain for a long time. The MongoDB, Express, React, Node (MERN) technology stack is used to achieve these goals. To achieve the goals we have been working for the rotary blood bank and getting their feedback to improve our system the end users. Apart from the basic features of a database i.e Add, Search, Update, and Delete, we are providing the offline mode and analytics. The extensive amount of data that is handled, this gives precise analytics. We have made the process of auditing less cumbersome and reduced data redundancy.

The analytics are made through several key index points that are available some of them are the total donor count, classification on the basis of blood count, classification on the basis of the region where the blood was donated, the ratio of male to the female population, and motivation.

A standardized time-efficient and sustainable inventory management system has been provided to the Rotary Blood Bank.

# **CHAPTER 7**

## **FUTURE ENHANCEMENT**

The current project can be enhanced for future use cases with the following pointers that are available.

### **7.1 Exception rate**

The future analysis can include the error count of the people and their causes of death. An example would be to check if there was a delay in searching or delivery of the blood or whether the blood was not available. This analysis will help the system evaluate its most prominent flaws and tackle them.

### **7.2 Donor location**

The application can be extended to the donors and if the need for blood arises users in the vicinity must be informed and allowed to donate blood. The users must be filtered on the basis of requirements and only a select number of candidates must get the message. This can also have logistics of whether the blood reached to patient or not. This will help the system to complete the cycle of blood donation.

### **7.3 Banks closet**

The system can expand to accommodate the hospital and allow them to look upon the blood bank near them and request blood at all times. This can include the ambulances to help track the nearest blood bank and request for blood in case of vital accidents and emergency situations.

## **7.4 Notifications**

The hospital if included must have a facility to notify the donors and the blood bank about its requirements and the priority level of the blood that is to be used. The blood banks must also know all the hospitals that are near to their hospital.

## **7.5 Register to digital**

An obvious part is to make the input easier to shift from the registers to the digital format. This can be achieved using Machine Learning and Image Processing. We can use the Otsu Binarization Technique along with a NN to recognize handwriting.

# APPENDIX A

## OTSU BINARIZATION THRESHOLDING

Otsu Binarization is a technique that is accustomed used to convert grayscale images to monochrome is one of the many binarization methods. This technique aims to find the threshold value where the sum of spread is minimum. It minimizes the grey levels of an image. Otsu's thresholding technique involves iterating all possible values of threshold and calculating the measure of the spread for pixel level that falls in foreground and background.

This method requires the exhaustive scratch of values that maximize the interclass variance, that is the weighted sum of variances between two classes. The objective is to minimize the interclass variance.

The algorithm assumes that the image contains two classes of pixels following bi-modal histogram, it then calculates the optimum threshold separating the two classes so that their combined spread is minimal.

### A.1 The Mathematics

We exhaustively search for the threshold, defined as a weighted sum of variances of the two classes:

$$\sigma_{\omega}^2(t) = \omega_0(t)\sigma_0^2(t) + \omega_1(t)\sigma_1^2(t) \quad (\text{A.1})$$

Weights  $\omega_0$  and  $\omega_1$  are the probabilities of the two classes separated by a threshold  $t$ , and  $\sigma_0^2$  and  $\sigma_1^2$  are variances of two classes. The class probability  $\omega_{0,1}(t)$  is computed from the  $L$  bins of the histogram:

$$\omega_0(t) = \sum_{i=1}^{t-1} p(i) \quad (\text{A.2})$$

$$\omega_1(t) = \sum_{i=t}^{L-1} p(i) \quad (\text{A.3})$$

Otsu demonstrates that minimizing the intra-class variance is the same as maximizing inter-class variance as shown by the equation :

$$\sigma_b^2(t) = \omega_0(t)\omega_1(t)[\mu_0(t) - \mu_1(t)]^2 \quad (\text{A.4})$$

That is expressed in terms of probabilities  $\omega$  and means  $\mu$  . Keeping the class mean as  $\mu_{0,1,T}(t)$  is:

$$\mu_0(t) = \frac{\sum_{i=0}^{t-1} ip(i)}{\omega_0(t)} \quad (\text{A.5})$$

$$\mu_1(t) = \frac{\sum_{i=t}^{L-1} ip(i)}{\omega_1(t)} \quad (\text{A.6})$$

$$\mu_T = \sum_{i=0}^{L-1} ip(i) \quad (\text{A.7})$$

These can be verified:

$$\omega_0 + \omega_1 = 1 \quad (\text{A.8})$$

The class probabilities and class means should be computed at all turns.

## A.2 The algorithm that can be implemented

There is a four-step algorithm.

**Step 1:** Make a histogram and probabilities for each intensity

**Step 2:** Set up the initial  $\omega_i(0)$  and  $\mu_i(0)$ .

**Step 3:** Step through all possible thresholds from  $t=1 \dots$  maximum intensity.

**Step 3.1:** Update  $\omega_i$  and  $\mu_i$ .

**Step 3.2:** Compute  $\sigma_b^2(t)$ .

**Step 4:** The desired threshold corresponds to the maximum of  $\sigma_b^2(t)$ .

### **A.3 Availability**

The implementation can be done in any language which facilitates mathematics. It is advised to use Matlab, Octave or Python for implementation. The OpenCV library has an inbuilt implementation of Otsu Binarization technique.



## REFERENCES

1. Anand, A. (2015). "Blood for sale: India's illegal 'red market'". <https://www.bbc.com/news/business-30273994>. [Online; accessed 16-April-2019].
2. Ayusharma0698 (2018). "Introduction to NoSQL". <https://www.geeksforgeeks.org/introduction-to-nosql/>. [Online; accessed 16-April-2019].
3. Ceta, N. (2018). "All You Need to Know About UML Diagrams". <https://tallyfy.com/uml-diagram/>. [Online; accessed 16-April-2019].
4. Graham, D. (2018). *Foundations of Software Testing*. Thompson, 3rd edition.
5. Jr., V. M. F. (2012). *The Ultimate Guide to the SDLC*. FrontLine, 1st edition.
6. Mathur, S. (2008). "Offline handwriting recognition using genetic algorithm".
7. Millar, K. (2018). *Data Management Life Cycle*. McGraw-Hill, 1st edition.
8. MongoDB (2019). "MongoDB Manual". <https://docs.mongodb.com/manual/tutorial/getting-started/>. [Online; accessed 16-April-2019].
9. Naik, S. (2013). *Concepts of Database Management Systems*. Pearson India, 7th edition.
10. NodeJS (2019). "NodeJS Docs". <https://nodejs.org/en/docs/>. [Online; accessed 16-April-2019].
11. Proto.io (2017). "Why Wireframes Are Important in the Design Process.". <https://link.medium.com/vcQizFe7YV>. [Online; accessed 16-April-2019].
12. ReactJS (2019). "ReactJS Docs". <https://reactjs.org/docs/getting-started.html>. [Online; accessed 16-April-2019].

13. RedCross (2019). "What Happens to Donated Blood". <http://fw.to/CRqd40b>. [Online; accessed 16-April-2019].
14. Vyas, K. (2018). "A Box detection algorithm for any image containing boxes.". <https://link.medium.com/CtKT4OwToV>. [Online; accessed 16-April-2019].
15. Wikipedia (2018). "Systems design". [https://en.wikipedia.org/wiki/Systems\\_design](https://en.wikipedia.org/wiki/Systems_design). [Online; accessed 16-April-2019].
16. Wikipedia (2019). "Website wireframe". [https://en.wikipedia.org/wiki/Website\\_wireframe](https://en.wikipedia.org/wiki/Website_wireframe). [Online; accessed 16-April-2019].