UTTARA UNIVERSITY

BLOOD DONATION MANAGEMENT SYSTEM

BY

MD. HAMIDUL HAQUE HAMID	2183081026	46 (A DAY)
SHOULY HOSSAIN JYOTI	2183081049	46 (A DAY)
KAZI MAHBUB RAHMAN	2183081009	46 (A DAY)
AFROZA AKTER	2183081001	46 (B DAY)
FATAMATUJ JOHORA SHANTU	2183081042	46 (B DAY)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
SCHOOL OF SCIENCE AND ENGINEERING
UTTARA UNIVERSITY

SUMMER 2022

UTTARA UNIVERSITY

BLOOD DONATION MANAGEMENT SYSTEM

BY

MD. HAMIDUL HAQUE HAMID	**2183081026**	**46 (A DAY)**
SHOULY HOSSAIN JYOTI	**2183081049**	**46 (A DAY)**
KAZI MAHBUB RAHMAN	**2183081009**	**46 (A DAY)**
AFROZA AKTER	**2183081001**	**46 (B DAY)**
FATAMATUJ JOHORA SHANTU	**2183081042**	**46 (B DAY)**

A project submitted in partial fulfillment of the requirement for the degree of Bachelor of Science in Computer Science and Engineering.

Department of Computer Science and Engineering
School of Science and Engineering
Uttara University, Uttara, Dhaka, Bangladesh

SUMMER 2022

ABSTRACT

A blood donation management system is created for the benefit of people in order to manage the blood donation process from the blood request through the assignment of donors, ensuring that no one would suffer a blood-related death. Anyone can request for blood using this web-based tool. This approach benefits welfare organizations, blood banks, hospitals, and individuals from every sector. This system can be used effortlessly and without any hesitation even through donors.The goal of this project is to create a web-based free of cost blood donation management system that will assist in managing the blood donation system in a sophisticated and digital manner. It has been created in a clear and fluid manner. Our main objective is to create a website with a simple appearance that included all necessary features. Therefore, we haven't concentrate too much on the design but rather always tried to offer it a straight forward appearance with appropriate functionality. This system is designed to keep track of blood donations and documentation, thereby saving the lives of individuals in need of blood. By employing this approach, blood donors of a certain blood group can be assigned as needed, and blood seekers can place emergency blood requests. Anyone without a login or registration can request blood. This technique protects the confidentiality of blood donors. This system will offer a variety of useful features that will reduce costs and maintain the report digitally current through automated notifications and a pdf report generator.

Keywords: *Blood; Donation; BDMS; Appication; CSE; Management; Welfare; Seeker; System; Notifications;* Automated; Verified; Donors;

APPROVAL

I certify that I have supervised this project and read this manuscript. In my opinion, it conforms to acceptable standards of scholarly presentation and is fully adequate in scope and quality, as a report for the degree of B.Sc in Computer Science and Engineering.

Ratul Sikder
Senior Lecturer, Dept. of CSE
Supervisor

I certify that I have read this study. In my opinion, it conforms to acceptable standards of scholarly presentation and is fully adequate in scope and quality as a project for the degree of B.Sc in Computer Science and Engineering.

MD. Torikur Rahman
Assistant Professor, Dept. of CSE
Coordinator / Internal Examiner

This project report was submitted to the Department of Computer Science and Engineering and is accepted as a fulfillment of the requirement for the degree of B.Sc in Computer Science and Engineering.

A.H.M Saifullah Sadi
Chairman, Dept. of CSE

DECLARATION

We here by declare that this report is the result of our own investigations, except where otherwise stated. We also declare that it has not been previously or concurrently submitted as a whole for any other degrees at Uttara University or any other institutions. We also declare that the formatting of the manuscript is same as the provided template. We also do not have any objections for the further use of the manuscript as Uttara University has all the rights to update, publish or conduct further systems of the submitted work.

Student Names	Student IDs	Signature	Date
MD. Hamidul Haque Hamid	2183081026	_______________	_______________
Shouly Hossain Jyoti	2183081049	_______________	_______________
Kazi Mahbub Rahman	2183081009	_______________	_______________
Afroza Akter	2183081001	_______________	_______________
Fatamatuj Johora Shantu	2183081042	_______________	_______________

UTTARA UNIVERSITY

DECLARATION OF COPYRIGHT AND AFFIRMATION OF FAIR USE OF UNPUBLISHED SYSTEMS

BDMS (BLOOD DONATION MANAGEMENT SYSTEM)

We declare that the copyright holders of this dissertation are jointly owned by the students and Uttara University (UU).

No part of this unpublished systems may be reproduced, stored in a retrieval system or transmitted in any form or by any means, electronic, mechanical, photocopying, recording or otherwise without prior written permission of the copyright holder except as provided below.

1. Any material contained in or derived from this unpublished systems may be used by others in their writing with due acknowledgement.

2. UU or its library will have the right to make and transmit copies (print or electronic) for institutional and academic purposes.

3. The UU library will have the right to make store in a retrieved system and supply copies of this unpublished systems if requested by other universities and systems libraries.

By signing this form, we acknowledged that we have read and understand the UU Intellectual Property Right and Commercialization policy.

Affirmed by **MD. Hamidul Haque Hamid**

……..……………………………………… ………………………..
Signature (on behalf of the team) Date

DEDICATION

We dedicate this report to
our honorable parents and our younger brother(s)
for their meticulous support, continuous inspiration and unconditional love
till the very end of this journey.

ACKNOWLEDGEMENTS

A project without proper guidance is like a ship without a navigator. A successful story is incomplete without paying tribute to those who inspired it. We would like to express our gratitude towards all those people who guided us in preparing this project which was a great learning process for us.

We are heartily indebted to **Senior *Lecturer, Ratul Sikder sir*** Department of Computer Science and Engineering of Uttara University who guided us throughout the project and gave up valuable suggestions and encouragement.

Once again, we would like to thank all those people who gave their valuable support by helping us in this project which is a great learning experience.

Above all we would like to thank the Almighty, our family, supervisor and friends for their encouragement and unending motivation.

TABLE OF CONTENT

LIST OF TABLES

LIST OF FIGURES

LIST OF SYMBOLS/ACRONYMS

HTML	Hyper Text Markup Language
CSS	Cascading Style Sheets
JS	JavaScript
PHP	Hypertext Preprocessor
SQL	Structured Query Language
DB	Database
PDF	Portable Document Format
ERD	Entity Relationship Diagram
DFD	Data Flow Diagram
CSE	Computer Science and Engineering
FYP	Final Year Project

CHAPTER ONE
INTRODUCTION

1.1 OVERVIEW

BDMS (Blood Donation Management System) is a web-based application by which anyone can request for blood. A blood request then appears on the admin panel, an admin verifies the request. After that admin assign donors for a particular blood-seeking request by a particular blood group. Donors are then notified by automated notification through email. After accepting any of his requests, donor can download the informations of seeker. Donor can also notify blood was donated or not. If blood was donated donor becomes inactive for 3 months. If blood wasn't donated donor have to mention the reason. Here, anyone can signup for becoming donor. Admin and super-admin role can be given if he or she is registered as donor. Monthly, yearly and total report of application and donation histories of donors can be downloaded as pdf.

1.2 PROBLEM STATEMENTS

There are some significant problems in these three websites [1], [2], [3] that we have taken as our case study. And also, there are some improvements that should be done. These are discussed below:

1.2.1 MESSY USER INTERFACE

The three websites [1],[2],[3] have a very messy User Interface (UI) which makes a user very confused about where to begin for a simple registration. All of them provides some unnecessary informations, articles and so on.

1.2.2 SIGNUP FOR BLOOD SEEKERS

In our case study period, we noticed that blood seeker registration is a must. It dosen’t make any sense that a blood seeker have to register at their system just for giving a request for blood. So we took a decision to remove this feature and will them a better experience with ‘Guest Mode’. It means anyone can make a request for blood with some verified informations and admin panel will verify and accept the request. By this, no-one have to register for just giving a blood seeking request.

1.2.3 LONG TIME TO RESPONSE

This is a major problem we found in case study with those three websites. Whenever some one request for blood or a donor requests for sign up, they took too much time to approve them. Blood seeking is a sensitive case , it can not be delayed. On the other hand, the second website [2] we studied it takes much time to load and slower to do some operations.

1.2.4 NOTIFICATIONS

Among these three websites [1],[2],[3] , two of them [2],[3] are not sending any real time notifications. Notications are very important for this kind of website by which anyone can be notified what is the decision admin panel taken to requests or what is the status right now. They can use email for notify their user or others but they didn’t. Notifactions are so much reliable to keeping records and giving updates. So, we have choosen this features to be included in our system.

1.3 PROJECT OBJECTIVES

Our main goal is to develop a simple UI based login - registration system, banners to influence donors, blood request in guest mode, verified users by secure user verification, secure data & privacy for the user, automated notifications and hastle free donor assign, events, generating and downloading reports in pdf format and also responsive web-application in this Final Year Project. We took a time frame so that we can develop the web-application within the time. We didn't want to make this website a deal breaker thing, rather we focused to keep things as simple as possible. We started to work on this project after the very 1st semester of this Final Year (Fall 2021). We learned the necessary skills we needed to implement on this project. We expected to finish this project within July 2022 to the first week of August 2022. Our main objective is to develop a web application that is very easy to understand, saves time and stores data while managing the blood donation system.

1.3.1 ELEGANT USER INTERFACE

We planned our web-applicaton to provide a simple and easier user experience. That is why we designed our website as simple as possible. We would like to give users a better experience with so much informations. We have tried to make interface informative. There would be many kinds of users we will have in future.This web-application should not make anyone confused from where to begin. Dashboard is also providing informations according to user roles.

1.3.2 BLOOD REQUEST IN GUEST MODE

Unlike others, we planned to give users much beneficial and time saving features. Guest Mode is one of them. Actually we named this ‘Guest Mode’ just because a blood seekers does not need to register into the system just to put a blood request. Just because of time saving and making our database light we are offering this to our seekers. Admins will be assigned to maintain and verifying this process.

1.3.3 AUTOMATED NOTIFICATIONS

Notification is always an important feature. By sending notifications we can easily notify users or seekers about their blood request status , can notify admin that new blood requests or new donor signup requests has made, can notify donors that they have assign to donate blood and so on. In our system, we used mail to send notifications.

1.3.4 DONOR REGISTRATION AND BLOOD REQUEST VERIFICATION

We designed our blood request and donor registration form to provide all the data by which we can verify donor registration and the blood-seeking request is genuine. For donor registration, a donor has to provide his/her original photo and NID card copy. Admin will find the users data from NID number and will cross the process.On the other hand, seeker has to provide an official document from hospital that the blood is needed for the patient. Admins will verify the requests.

1.3.5 HASSLE FREE DONOR ASSIGN

After verifying the blood request, then comes the donor assign proccess for corresponding blood request. Here, system will provide same blood group donors whom had donated blood 3 months ago. Admin can select multiple donors based on location or donation period or so on. It is a hastle free process for admins to manage requests.

1.3.6 GENERATING AND DOWNLOADING REPORTS

Our web application will generate reports as pdf format. It will generate reports of donor’s donation history, last month activities of application, last year activities of application and whole activities of application from our existing database which can be downloaded easily.

1.4 SCOPE

Project Scope Management includes three processes:

● **Planning** – The project is defined and the work (or processes) needed to deliver the project is determined. For this we have already determined the objectives we need to deliver.

● **Controlling** – This involves tracking, managing and monitoring the progress of a project, including tracking documentation, scope creeps tracking the work during each phase and disapproving/approving any changes along the way.For this, we have created documentation such as Gantt charts to keep track of our progress.

● **Closing** – This is the “wrap-up” part of the process, which involves an audit of the project deliverables and accessing the results of the final product against the original defined plan. The closing will be completed after we have completed all the phases of the development cycle of our project.

1.5 PROJECT METHODOLOGY

The methodology is the theoretical and methodical study of the methods, procedures or processes used to attain the project's objectives and goals. It consists primarily of a paradigm, a theoretical model, stages and quantitative or qualitative methodologies. For our software development life cycle, we employed the Agile methodology. This section will go through agile methods and a block diagram for our website.

1.5.1 AGILE METHODOLOGY

Agile technique is used to manage projects by segmenting them into phases. Stakeholders are constantly co-operated with throughout the process and improvements are made on a regular basis. When the job begins, teams go through a cycle of planning, execution and evaluation. Collaboration is essential both with team members and project stakeholders. The six stages of the agile life cycle are as follows:

i) **Concept**

The concept phase comes first. A product owner will decide on the scope of their project here. If there are several initiatives, the most important ones will be prioritized. The product owner will meet with the client to discuss important needs and develop documentation outlining them, including which features will be supported and the planned end outcomes. It is best to make the requirements as simple as possible because they can be expanded later. This extensive study will assist them in determining whether a project is feasible before beginning work.

ii) **Inception**

As soon as the concept has been outlined, it is time to assemble the software development team. In addition to providing the necessary tools and resources, the product owner checks the availability of their colleagues and chooses the best people for the project. They can then start the design process. An interface mockup and a project architecture will be designed by the team. In the inception stage, stakeholder input is gathered to flesh out the product requirements and determine its functionality.

ii) **Iteration**

Next up is the iteration phase also referred to as construction. It tends to be the longest phase as the bulk of the work is carried out here. The developers will work with UX designers to combine all product requirements and customer feedback, turning the design into code. The goal is to build the bare functionality of the product by the end of the first iteration or sprint. Additional features and tweaks can be added in later iterations. This stage is a cornerstone of Agile software development, enabling developers to create working software quickly and make improvements to satisfy the client.

iv) **Release**

The product is almost ready for distribution. However, the quality assurance team must first do several tests to guarantee that the software is completely functional. These Agile team members will test the system to ensure that the code is clean; if any problems or flaws are discovered, the developers will address them as soon as possible.This phase will also include user training, which will necessitate additional documentation. When all of this is finished, the product's final iteration can be put into production.

v) **Maintenance**

Customers will now be able to access the software after it has been fully delivered. This operation puts it in the maintenance mode. During this phase, the software development team will give continuing assistance to ensure that the system runs properly and that any new defects are resolved. They will also be available to provide further training to users and ensure that they understand how to utilize the product. New iterations might occur throughouttime to refresh the existing product with updates and new features.

vi) **Retirement**

There are two reasons why a product will enter the retirement phase: either it is being replaced with new software or the system itself has become obsolete or incompatible with the organization over time. The software development team will first notify users that the software is being retired. If there is a replacement, the users will be migrated to the new system. Finally, the developers will carry out any remaining end-of-life activities and remove support for the existing software. Each phase of the Agile life cycle contains numerous iterations to refine deliverables and deliver great results. Let's take a look at how this iteration workflow works within each phase: The Agile iteration workflow.

Agile iterations are usually between two and four weeks long with a final completion date. The workflow of an Agile iteration will typically consist of five steps:

i) Plan requirements

ii) Develop product

iii) Test software

iv) Deliver iteration

v) Incorporate feedback

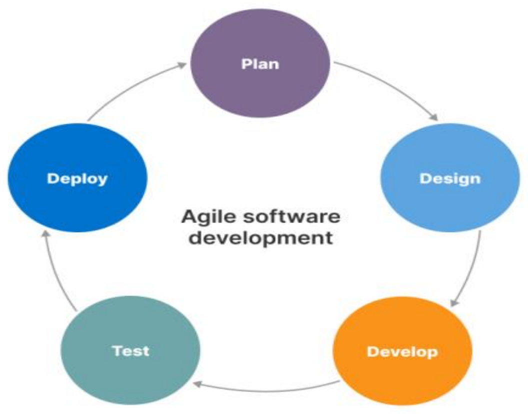


Figure 1.1 Flow diagram (Agile)of the activities in the development process of the project.

1.5.2 BLOCK DIAGRAM

We have created a block diagram to show preliminary architecture for our blood donation management system. In our block diagram, we have to show how seekers will be able to submit request for bloods, how donor sign up requests are made all kinds of things related to bdms. These are the front-end parts of our website. On the other hand, in the back-end part of website maintenance, how admin approves requests and how admin assign donors to a particular blood request, download pdf reports. MySQL is used to create our database to handle all the data in our website.

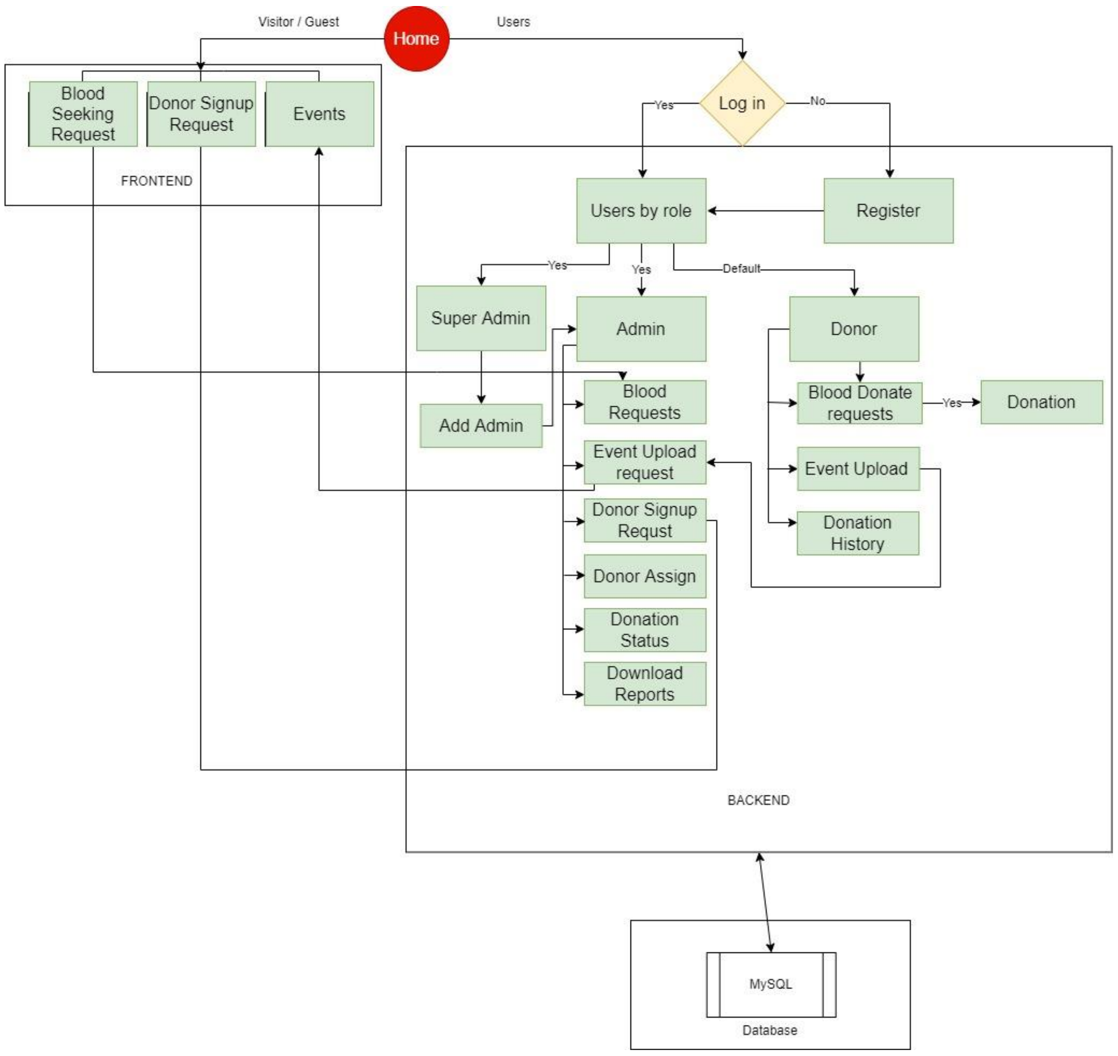
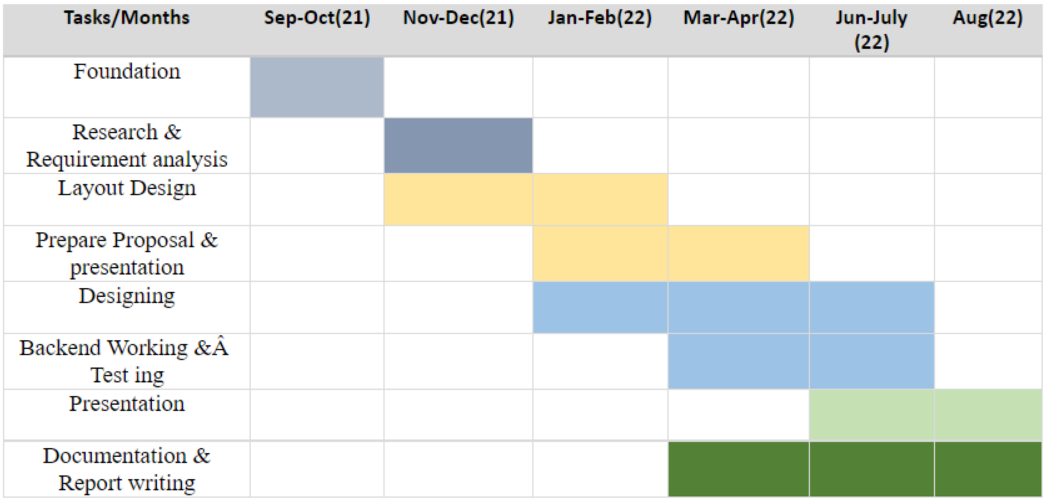


Figure 1.2 Block Diagram of the BDMS project.

1.6 GANTT CHART AND SYSTEMS MILESTONE



Tasks/Months	Sep-Oct(21)	Nov-Dec(21)	Jan-Feb(22)	Mar-Apr(22)	Jun-July (22)	Aug(22)
Foundation	▨					
Research & Requirement analysis		▨				
Layout Design		🟨	🟨			
Prepare Proposal & presentation			🟨	🟨		
Designing			🟦	🟦	🟦	
Backend Working &Â Test ing				🟦	🟦	
Presentation					🟩	🟩
Documentation & Report writing				🟩	🟩	🟩

Figure 1.3 Gantt Chart of the proposed project activities.

SL NO	Milestone	Date
1	Completion of background study and comparative analysis.	02 May 2022
2	Completion of formatting document done.	17 May 2022
3	Completion of explaining all the points	03 June 2022
4	Completion of checking and reviewing the overall documents	25 July 2022
5	Completion of publishing the document template	01 August 2022

Table 1.4 Project Milestones and dates.

1.7 SUMMARY AND OUTLINE

Chapter 1: Chapter one presents the motivational and introductory statements of the project. The chapter also presents problem statements, objectives, methodology and Gantt chart of the project.

Chapter 2: This chapter presents in depth Literature review or Background study of the conducted systems or project.

Chapter 3: In this chapter, detail analysis of the project/thesis, modeling and design of the overall system or conducted project or systems are presented.

Chapter 4: This chapter presents a detail description about system setup, implementation and testing of the system/prototype.

Chapter 5: Result analysis and benchmarking of the conducted systems or project are presented in Chapter 5.

Chapter 6: Conclusion and Recommendations are presented in this chapter.

CHAPTER TWO
BACKGROUND STUDY

2.1 OVERVIEW

Background study is a kind of systems, study and evaluation of available system’s features within the area of chosen or proposed topic or title. It documents the important points related to the proposed field and makes comparisons among the related systems to provide a clear view of the available strategies, their advantages, disadvantages, applications and potential or future works.

2.2 BACKGROUND STUDY

A reach study on systems or backgrounds aids in the generation of project philosophy, hypothesis, systems inquiries and most crucially, recent system challenges. Recent difficulties will lead to the definition of problem statements and the identification of project objectives. For completing our background of existing websites like we studied three of them, which are listed below.

i) BADHAN

ii) Bloodworks Northwest

iii) American Red Cross

2.2.1 FIRST CASE STUDY

In order to perform our case study, we have chosen "https://badhan.org"[1].BADHAN is a platform which allows many of welfare tasks blood donation is one of them. Badhan can manage Donor Details Information, Blood Components Management, Search-Based on Donor Registration ID and Donor Name, Add and Delete Donor Information and enable /disable a Donor.

Throughout these features badhan also has some limitations.

i) No proper conditions between different Applications and Users.

ii) Cannot upload and download the latest updates at the right time.

iii) Anyone can search for a donor which does not maintain the donor's privacy.

iv) Have to be a registered user for making a blood-seeking request.

2.2.2 SECOND CASE STUDY

In our second case study, we took "Bloodworks Northwest" [2]. Bloodworks Northwest is a non-profit organization harnessing donor gifts to provide a safe, life saving blood supply. It is a very well-designed website to donate blood and so on.Bloodworks Northwest's limitations:

i) No monitoring system over private blood transfusion.

ii) Have to make an appointment for making a blood-seeking request.

iii) Takes too much time to approve requests.

iv) No privacy for donors.

v) They do not provide notifications.

vi) The website takes too much time to load.

2.2.3 THIRD CASE STUDY

Our last and final case study was "American Red Cross" [3]. This website has so many features such as scheduling an appointment, viewing donation history, searching donors, blood donation process and donation process overview. Despite these, it has some limitations that we had noticed:

i) No option for requesting blood for an ordinary man. Only blood banks can request.

ii) The user interface of the American Red Cross is so much tough to understand.

iii) Doesn’t maintain data in a better way.

iv) It does not have any options to download reports.

2.3 SUMMARY

In this chapter, we finished our background research for three cases in order to determine the common characteristics and limitations of blood donation management systems from current websites.

2.4 CONCLUSION

We hope to fulfill our objectives by creating a web application with the functionalities required to handle the blood donation system. The general public will benefit from this web application. Our website will also have a user-friendly interface. Our website will be built using common features and will eliminate the constraints discovered during our background study.

CHAPTER THREE
SYSTEM ANALYSIS / MODELING & DESIGN

3.1 OVERVIEW

We have examined and analyzed the design process of the Blood Donation Management System in this chapter. Before the development process begins, the requirements are fully described and an overview of the system needs is provided. The system model improved the efficiency and efficacy of attaining system goals.

3.2 SYSTEM ANALYSIS

Systems analysis is defined as "The multi-disciplinary problem-solving activity developed by analysts to deal with technical system difficulties." A system has been developed to implement all of the required functionality for the blood donation management system web application. To understand the major requirements of the blood donation management system web application, a requirement analysis was performed. Following the discovery of the needs, the website was designed to be efficient and useful in meeting those requirements. System analysis has made use of requirement analysis. The process of determining user expectations for a new website or program being constructed or upgraded is known as requirement analysis, sometimes known as requirement engineering. Many key functional and non-functional requirements were discovered as a result of the analysis. Non-functional properties assisted in recognizing the properties needed to improve the quality of service given by the blood donation management system online application, while functional needs helped to grasp the major functions of the website. These requirements have been fulfilled in order for the website to function properly.

3.3 FUNCTIONAL REQUIREMENTS

A functional requirement defines a system or its component. It describes the functions the website must perform. The following functional requirements are needed for the BDMS web application.

3.3.1 BLOOD REQUEST

Blood request is actually blood seeking request. The seeker does not need to register into the system for uploading a blood-seeking request. But he/she has to provide all the information in the blood request form such as patient pieces of information, blood requirements, hospital pieces of information and verified documents.

3.3.2 DONOR REGISTRATION

Donor registration is a must for participating blood donation activates through this web application. Actually, donors are the registred users in this application. Donor informations like NID, date of birth, valid mobile number and email address, blood group, address etc will be the parts of this component. Donors can create an account and after becomes verified he/she is able to donate blood.

3.3.3 ADMINISTRATOR

An administrator(s) should be able to check and verify the information about donor registration, blood requests and event uploads. He/she can also donate blood. An admin after verifying blood-seeking requests can assign donors on the request. Also, he can download reports of the total activities. Admin can view, edit or delete the donor, blood requests, and events.

3.4 NON-FUNCTIONAL REQUIREMENTS

In systems engineering and requirements engineering, a non-functional requirement (NFR) is a requirement that specifies criteria that can be used to judge the operation of a system, rather than specific behaviors. They are contrasted with functional requirements that define specific behavior or functions. The following non-functional requirements are needed for the BDMS web application.

3.4.1 EFFICIENCY REQUIREMENTS

Websites will make it easy for donors to register and donate blood. Seekers can also request for blood without losing their time on registrations. New donors will find this website efficient for donating blood without any worries. Donors can quickly accept verified blood request by knowing details and addresses. Seekers will be mailed the information of donors. Blood donation process can be monitored by admin panel. As the system is web-based, anyone can use this system remotely.

3.4.2 RELIABILITY REQUIREMENTS

The system must perform accurately towards different input requests. For example, when the users updated profile details, the website is able to update the database to show the change. The website will also have a validity check to check the input to prevent wrong data type or wrong information.

3.4.3 SECURITY REQUIREMENTS

This website must be highly secure with registration and login. Passwords must be secured to provide security of the account. We used hash make technology to make the password encrypted. No sql injection can not be one. Login panel is so much secure that too much attempt on wrong login can not be made. Database can only be viewed by the administrators. Deleting an account is the administrator's only ability.

3.4.4 USABILITY REQUIREMENT

This system is designed with a user-friendly and easy-to-use path so that the users can perform their process easily. It must have clean instructions to guide the user through the system. Besides that, the description of the error message should be clear.

3.5 DESIGN

One of the most significant phases of website development is designing. It necessitates significant preparation and consideration on the part of the system designer. Data flow diagrams and use case diagrams are generated to aid in the development of the website. Data flow design will assist developers in demonstrating how data flows from one end of the website to the other. The development was carried out in accordance with the data flow diagram from beginning to end. The use case diagram aids in identifying the primary actions of website visitors. This aids in the establishment of processes that must be put in place to facilitate those actions. Users of the website will be unable to access their functions unless this is done. As a result, if the use case diagram is correctly designed, the website will be useful and productive.

3.5.1 DATA FLOW DIAGRAM

The data flow diagram depicts how data flows from one end to the other when people access the website. The figure depicts the website's preliminary architecture. On the other hand, the backend of website upkeep handles admin and contribution activity. MySQL is used to establish a database that will house all of the website's data.

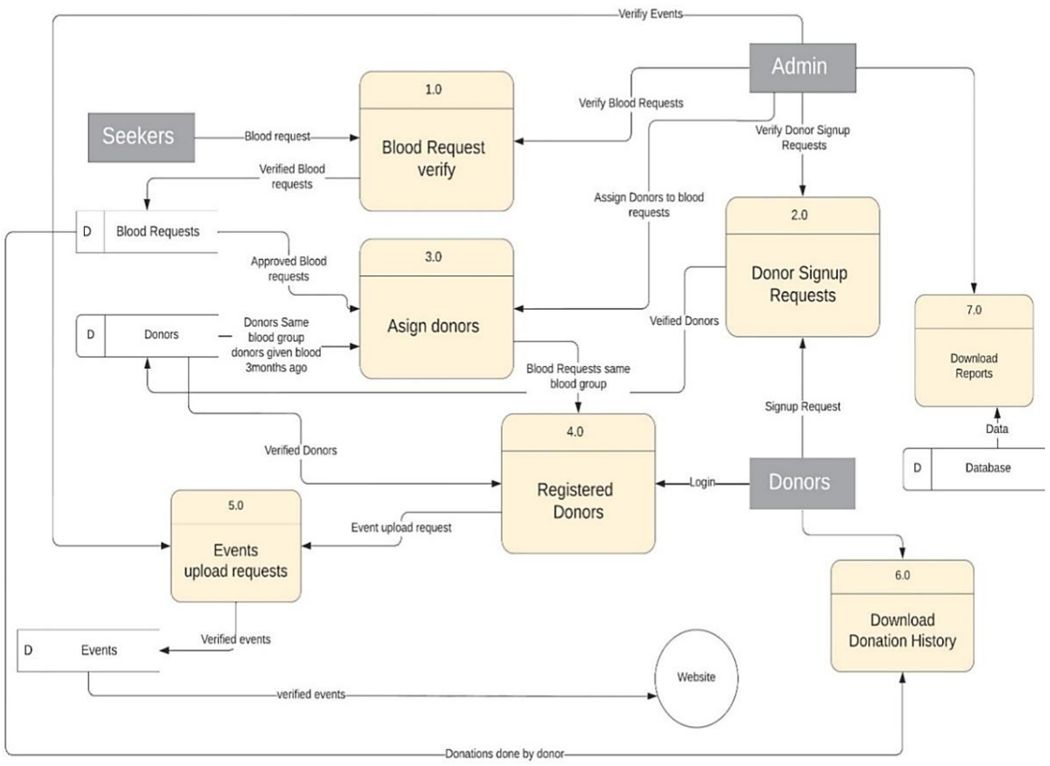


Figure 3.1 Data flow diagram of Blood Donation Management System.

3.5.2 USE CASE DIAGRAM

A use case diagram in UML is used to demonstrate the different ways in which a user can interact with a system. Use case diagram of Blood donation management system:

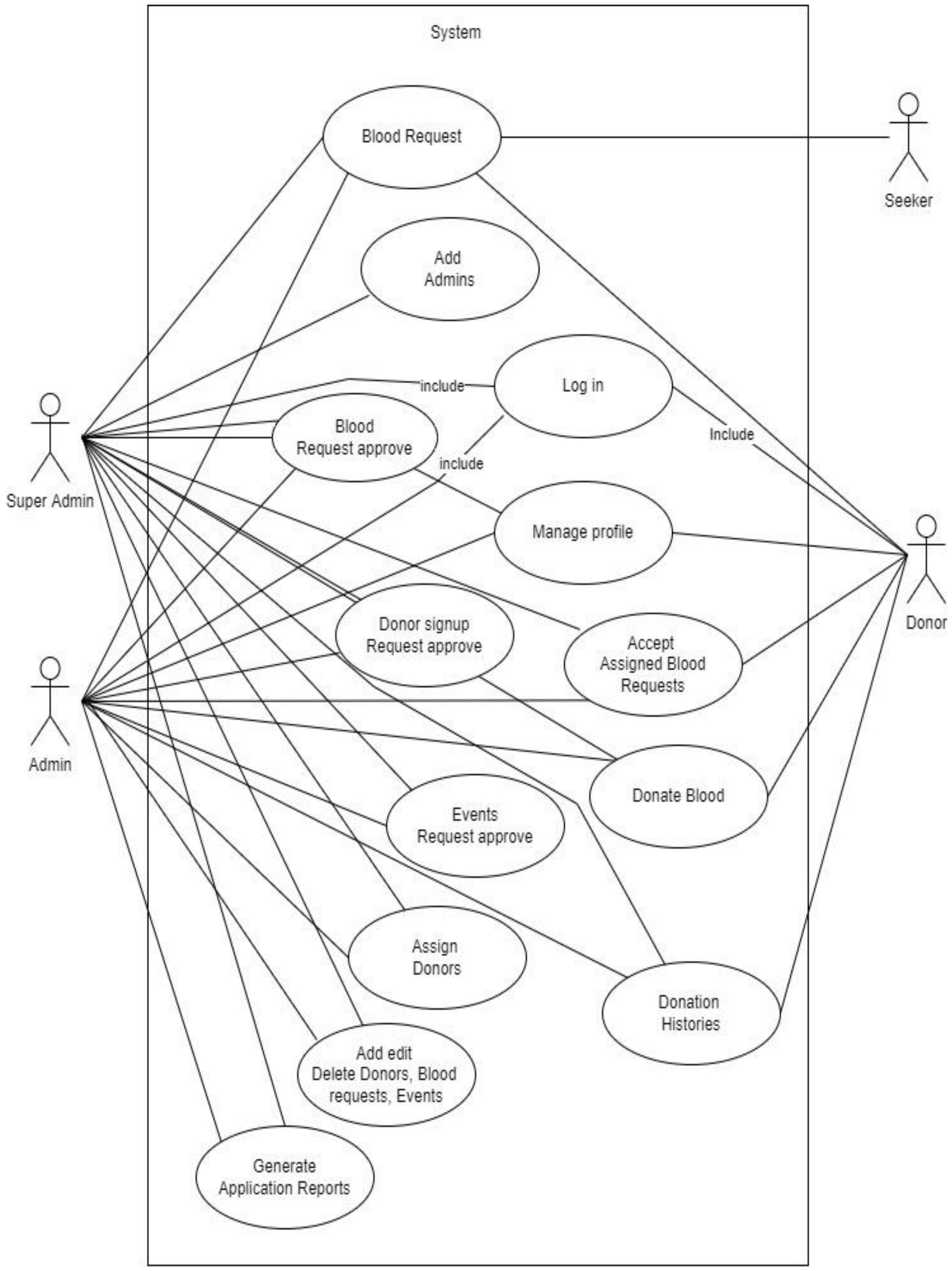


Figure 3.2 Use case diagram of Blood Donation Management System.

3.5.3 ER -DIAGRAM

An entity relationship diagram (ERD), also defined as an entity relationship model is a graphical depiction of the relationships between people, objects, places, concepts and events in an information technology (IT) system. The ER diagram below depicts the relationship between the website's primary elements.

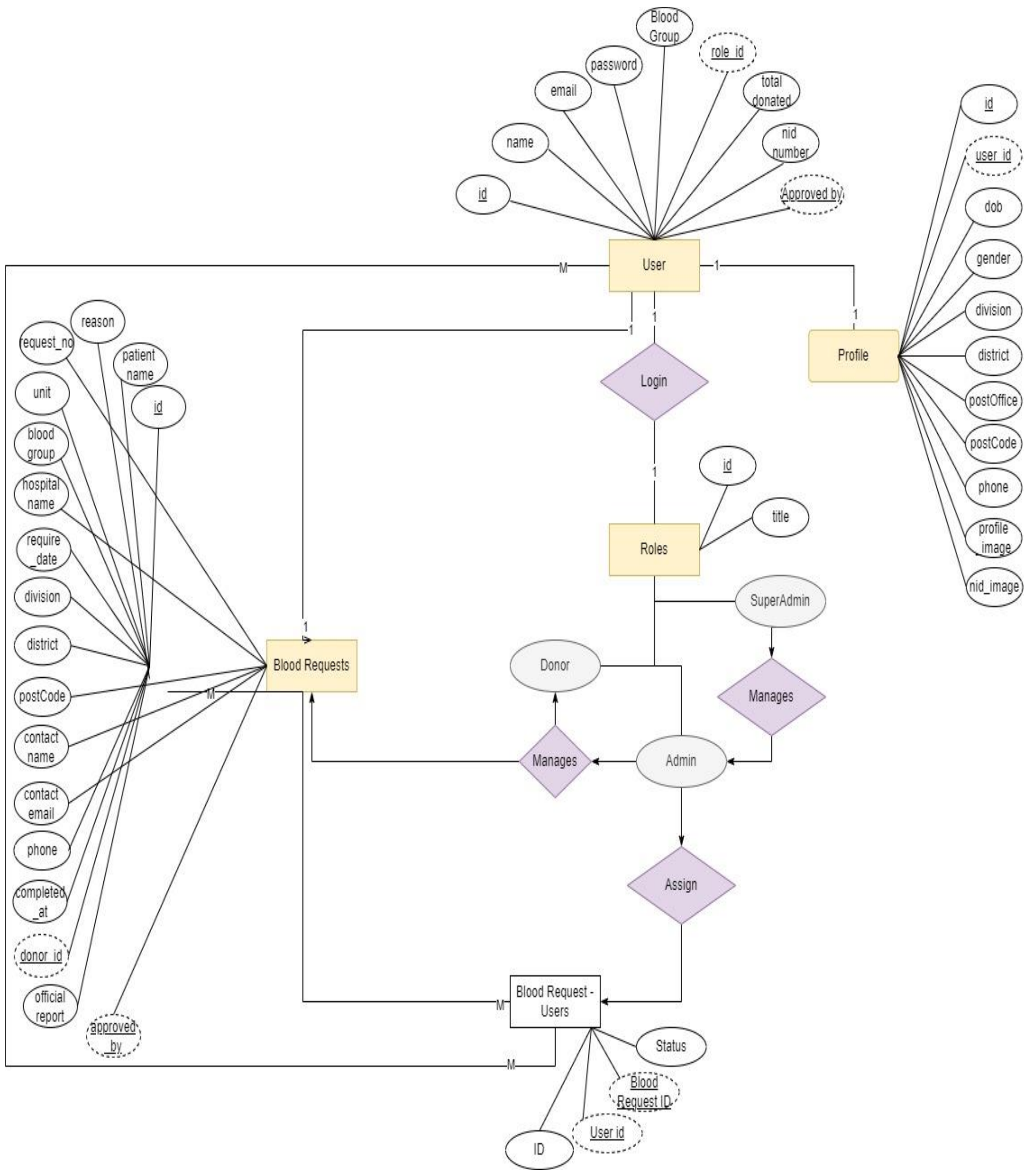


Figure 3.3 ER-diagram of Blood Donation Management System.

3.6 SUMMARY

The system analysis and design of the blood donation management system have been described in this chapter. The DFD data flow diagram depicts how data flows through the system. The use case diagram depicts the acting and tasks of several roles. ER-Diagram, on the other hand describes database flows, migrations and management.

CHAPTER FOUR
SYSTEM SETUP, IMPLEMENTATION & TESTING

4.1 OVERVIEW

System testing of software and hardware is performed on a system that is a complete integrated system that functions as a whole. System testing is a vital testing method that software engineers must perform before releasing the system. Several test cases will be run during the testing phase to assess the system's dependability and accuracy. A system testing report will be prepared based on the test cases for further analysis in order to identify system flaws and make appropriate modifications.

4.2 SYSTEM SETUP

A computer with sufficient processing capability is necessary for this project. The computer is essential for the developer to complete project development tasks such as database creation and updating. Several hardware requirements must be met for the entire system to function. To begin a database is required for the entire system to store and retrieve data. A laptop is also required to host the database and manipulate the entire system.

HARDWARE

i) Intel ® Core(TM) i3- 3220 M CPU @3.30 GHz / Any desipline

ii) 4 GB RAM

iii) 150GBstorage

iv) Internet connectivity

SOFTWARE

i) **HTML 5**

HTML 5 is the most recent iteration of the HTML standard. It is a new version of the HTML language, including new elements, properties and behaviors as well as a broader range of technologies.

ii) **CSS 3**

Cascading Style Sheets (CSS) is a style sheet language that is used to describe the display of a document produced in a markup language such as HTML. CSS, like HTML and JavaScript, is a foundational technology of the World Wide Web.

iii) **Bootstrap 5**

The frontend UI library we used is Bootstrap. We used several UI components from such package, such as a navigation bar, input fields, Vertical or Horizontal Steppers and data tables.

iv) **Javascript**

JavaScript, sometimes known as JS, is a computer language that adheres to the ECMAScript specification. JavaScript is multi-paradigm, high-level and often just-in-time compiled. It supports curly-bracket syntax, dynamic typing, prototype-based object-oriented programming and first-class functions.

v) **PHP**

PHP (Hypertext Preprocessor) is a scripting language that may be used to create dynamic and interactive web pages. It was one of the first server-side languages that could be embedded in HTML, making it easy to add functionality to web sites without requiring data from external files.

vi) **Laravel 9**

Laravel is a full-stack open-source PHP web application framework that is both sturdy and simple to use. It adheres to the model-view-controller design pattern. Laravel creates web applications by combining existing components from several industry frameworks just for recognizing by MVC.

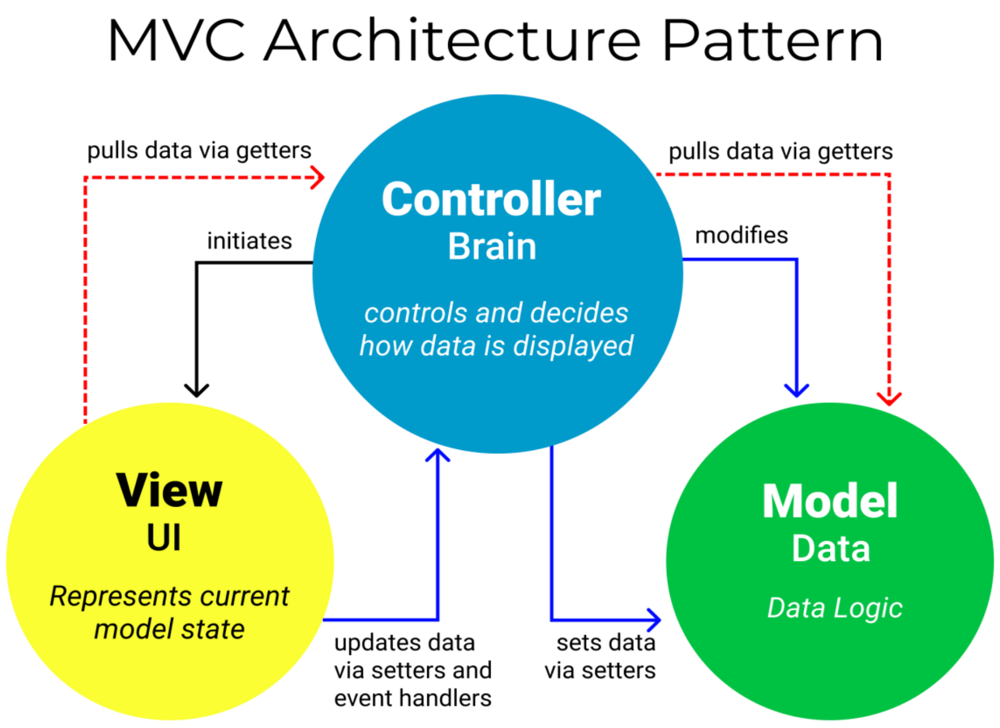


Figure 4.1 Model-View-Controller pattern of laravel 9.

vii) **My SQL**

MySQL is a relational database management system that is free and open source. Its name is a mix of "My", co-founder Michael Widenius's daughter's name and

"SQL" the abbreviation for Structured Query Language. We will utilize MySQL to store data in our database using phpmyadmin.

viii) **Local Server**

Local server like xampp, laragon will be used to host our application locally.

4.3 IMPLEMENTATION

In systems engineering, implementation (or project execution) is the phase where visions and plans become reality. This is the logical conclusion after evaluating, deciding, visioning, planning, applying for funds and finding the financial resources of a project. Technical implementation is one part of executing a project.

4.3.1 IMPLEMENTATION ISSUES & CHALLENGES

During the system implementation phase, several challenges need to be confronted because it involves end users to test the production system with various situations. The possible challenges may face are as following:

4.3.1.1 USERS WITHOUT BASIC COMPUTER SKILLS

The users are required to have basic knowledge of how to operate a computer system and online browser. It can be difficult to train the user for this type of knowledge. On the other hand, this knowledge is available online.

4.3.1.2 SERVER PERFORMANCE

During real time website implementation, there would be a huge number of blood requests and also a huge number of users that access the server to register for blood donations, download reports at the same time. Therefore, it might slow down the connection and performance of the system and even cause the server to shut down if the traffic goes way beyond the level of capacity.

4.3.2 DEVELOPMENT TOOLS

4.3.2.1 DATABASE ENVIRONMENT

The BDMS system has a database system to support in order to store the huge amount of data. MySQL database system is chosen to support the BDMS system because MySQL is a relational database management system (RDBMS) developed by Oracle that is based on structured query language (SQL).

4.3.2.2 WEB TECHNOLOGY

The BDMS system is an online based system. Therefore, it is required to implement web services in order to support the Android mobile phone client device to fetch data from the database server and store data back to the database server. In this project, MySQL Server is used to support the issue that was mentioned earlier.

4.3.2.3 DIAGRAM CREATION TOOL

In this project, “Lucidchart”, “Visual Paradigm” and “draw.io” have been used to create documents for several system planning diagrams such as Use-Case diagram, Activity diagram, etc. By doing so, software developers are able to visualize the system and communicate information much more concisely and clearly.

4.4 TESTING

The development process then went on to the system testing step after the system had been developed. During the system testing step, the produced system must be installed on appropriate testing devices. Following the completion of the system installation, the system testing duty will be conducted by various user roles such as super-admin, admin role, and user role. System testing is used to discover and assess the degree of system stability. Simultaneously, it provides an opportunity for developers to identify faults or bugs that were not raised or experienced during the system development phase. Any problems or bugs discovered during system testing will be addressed before the system is released. Each and every testing process prior to system testing is really tested by the system developer. As a result of the system developer's knowledge of the system software logic some biases toward testing may result, causing the Chapter 5 System Testing Result to be inappropriate. The developed system will be tested using four forms of testing: unit testing, functional testing, system testing and acceptance testing.

4.4.1 UNIT TESTING

First and foremost, unit testing will be the first testing approach utilized to validate the developed system. It is made up of testing operations that test the system module by module because it has not been integrated as a whole. Unit testing allows developers to readily uncover faults and problems because the error and bugs are found through a unit section of the system rather than the entire system. Furthermore, the developer will test the system's unit portion for validity and data value correctness. Valid and invalid information will be entered to test and ensure that the system operations work as planned. So far, the components such as account creation, blood seeking request, donor assign and download reports have been successfully tested and functioned as the developers expected.

4.4.2 FUNCTIONAL TESTING

Following unit testing, functional testing assesses the produced system. Functional testing ensures that system application processes operate as planned and in compliance with design and specifications. During functional testing, the primary system application functions will be tested with a variety of test cases to ensure that the entire system performs as a whole and completes tasks with the expected results. Logins with multiple users, blood-seeking requests, assigning donors, profile updates, event uploads, authorizing requests and downloading reports have all been tested.

TESTING 1: ENROLL WITH DIFFERENT USER (BY EMAIL)

Testing Objective: To ensure users with different roles login according to restricted system features.

NO	Event	Attribute & Value	Expected Result	Result
1	Login As “SUPER ADMIN”	Login with super admin information	Successfully logged in	PASS
2	Login As “ADMIN”	Login with super admin information	Successfully logged in	PASS
3	Login As “DONOR”	Login with Donor information	Successfully logged in	PASS

Table 4.3 Login with different user roles.

4.4.3 SYSTEM TESTING

This required testing the system to identify and remedy mistakes or problems. System testing entails putting the entire website to the test to see whether or not the system's functional requirements have been efficiently and effectively integrated and satisfied. Following system testing, it is determined that the website is operating as intended.

4.4.4 ACCEPTANCE TESTING

Finally, acceptance testing, also known as user acceptability testing, is the ultimate testing technique used to test the built software system. The testing activities in acceptance testing differ from those in prior testing activities since the tester who tests the system is the ultimate user who does not understand the system logic. If a final user experiences an error while using the system, system developers must maintain it as quickly as feasible and issue a new patch for the existing system to recover from the error. If the final user detects no defects while using the system for an extended length of time, the developer's development task is regarded complete and the system is a final system product.

4.5 SUMMARY

Throughout this chapter, we described how we implemented our system. Our system installation and configuration went without a hitch. Our supervisor has gone through our system with us thoroughly and determined that the deployment is a success.

CHAPTER FIVE
RESULT ANALYSIS

5.1 OVERVIEW

The blood donation management system website allows seekers to post blood requests and donors to use the platform so that they can donate blood. The donors are restricted to donate blood whether the last donation period has not overcome three months also they can download their last donation histories.

5.2 RESULT ANALYSIS

The BDMS website is designed and developed to solve all the problem statements which are stated in chapter one of this report. Using the website, it eliminates all the problems involving complex navigation systems. In addition, the developed website allows the admin to take necessary actions as needed. Furthermore, the project objectives which were mentioned regarding providing convenience for both users and admins and user-friendly UI have been achieved. In a nutshell, the developed website has fulfilled all the significant outcomes according to all the problem statements and project objectives that were stated.

Home page: The homepage of the blood donation management system website is shown in the picture.

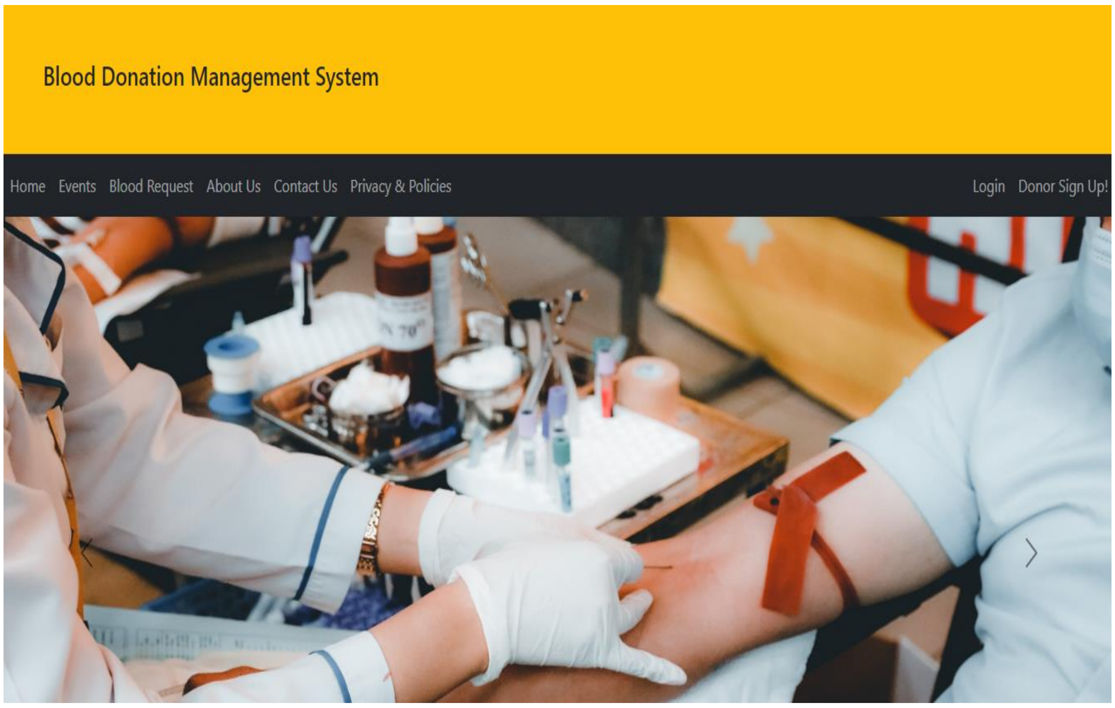
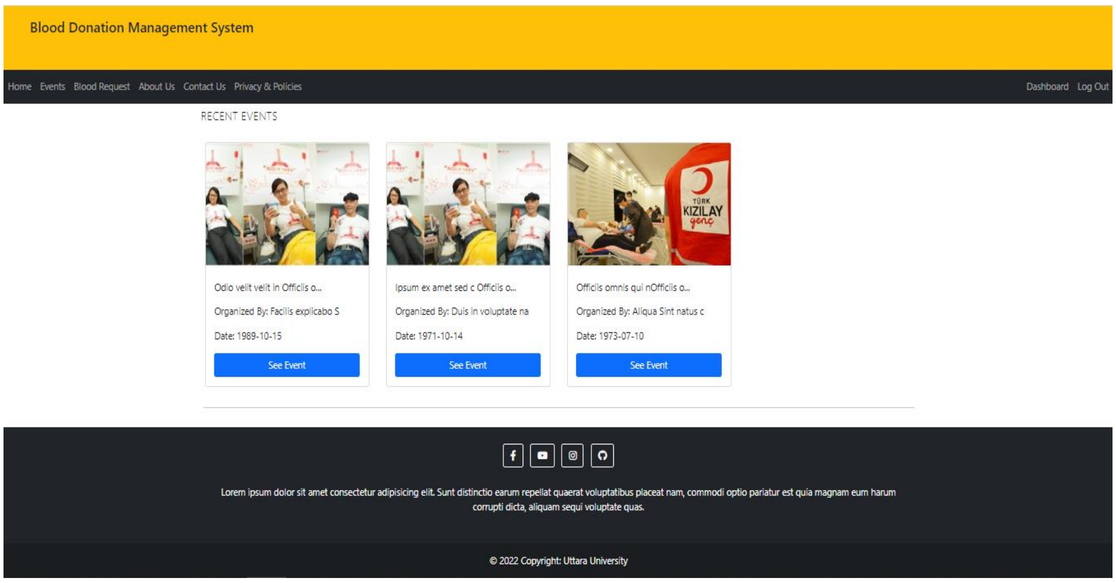

Blood Donation Management System
Home Events Blood Request About Us Contact Us Privacy & Policies
Login Donor Sign Up!

FIGURE 5.1 Homepage of blood donation management system.

Events page: The events page of the blood donation management system website is shown in the picture.


Blood Donation Management System
Home Events Blood Request About Us Contact Us Privacy & Policies
Dashboard Log Out
RECENT EVENTS

Odio velit velit in Officiis o...
Organized By: Facilis explicabo S
Date: 1989-10-15

Ipsum ex amet sed c Officiis o...
Organized By: Duis in voluptate na
Date: 1971-10-14

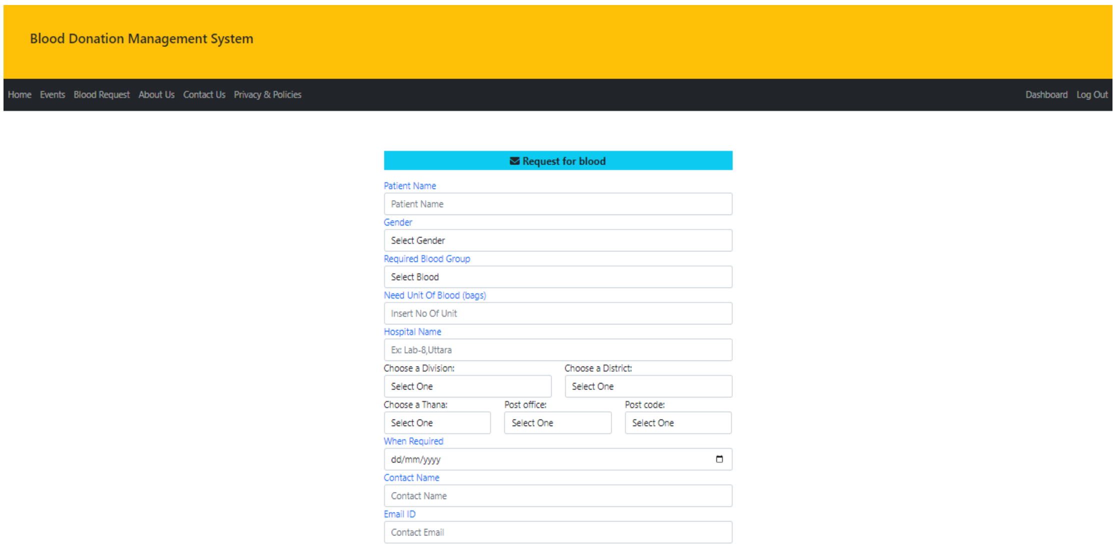
Officiis omnis qui nOfficiis o...
Organized By: Aliqua Sint natus c
Date: 1973-07-10

Lorem ipsum dolor sit amet consectetur adipisicing elit. Sunt distinctio earum repellat quaerat voluptatibus placeat nam, commodi optio pariatur est quia magnam eum harum corrupti dicta, aliquam sequi voluptate quas.

© 2022 Copyright: Uttara University

FIGURE 5.2 Events page of blood donation management system.

Blood request page: The Blood request page of the blood donation management system website is shown in the picture.



Blood Donation Management System

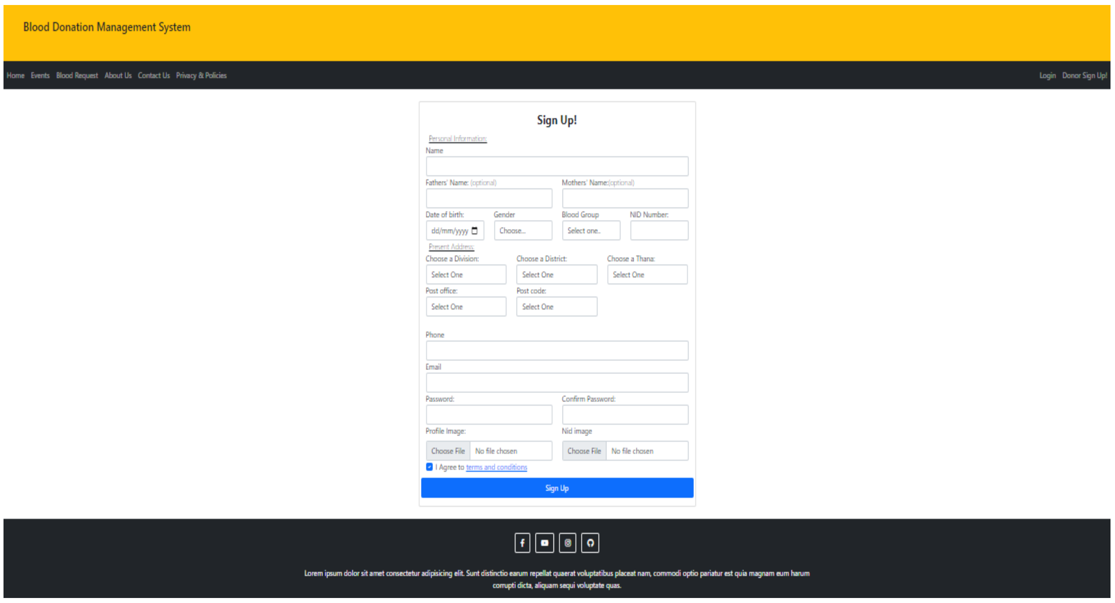
Home Events Blood Request About Us Contact Us Privacy & Policies Dashboard Log Out

Request for blood

Patient Name
Patient Name
Gender
Select Gender
Required Blood Group
Select Blood
Need Unit Of Blood (bags)
Insert No Of Unit
Hospital Name
Ex: Lab-8,Uttara
Choose a Division: Select One
Choose a District: Select One
Choose a Thana: Select One
Post office: Select One
Post code: Select One
When Required
dd/mm/yyyy
Contact Name
Contact Name
Email ID
Contact Email

FIGURE 5.3 Blood request page of blood donation management system.

Donor signup: The Donor signup page of the blood donation management system website is shown in the picture.



Blood Donation Management System

Home Events Blood Request About Us Contact Us Privacy & Policies Login Donor Sign Up!

Sign Up!

Personal Information:
Name
Fathers' Name: (optional)
Mothers' Name:(optional)
Date of birth: dd/mm/yyyy
Gender: Choose...
Blood Group: Select one..
NID Number:
Present Address:
Choose a Division: Select One
Choose a District: Select One
Choose a Thana: Select One
Post office: Select One
Post code: Select One
Phone
Email
Password:
Confirm Password:
Profile Image: Choose File No file chosen
Nid image: Choose File No file chosen
☑ I Agree to terms and conditions
Sign Up

Lorem ipsum dolor sit amet consectetur adipisicing elit. Sunt distinctio earum repellat quaerat voluptatibus placeat nam, commodi optio pariatur est quia magnam eum harum corrupti dicta, aliquam sequi voluptate quas.

FIGURE 5.4 Donor signup page of blood donation management system.

Login page: This is the login page for admins and donors. After verifying data users will see the dashboard which is limited by his role.

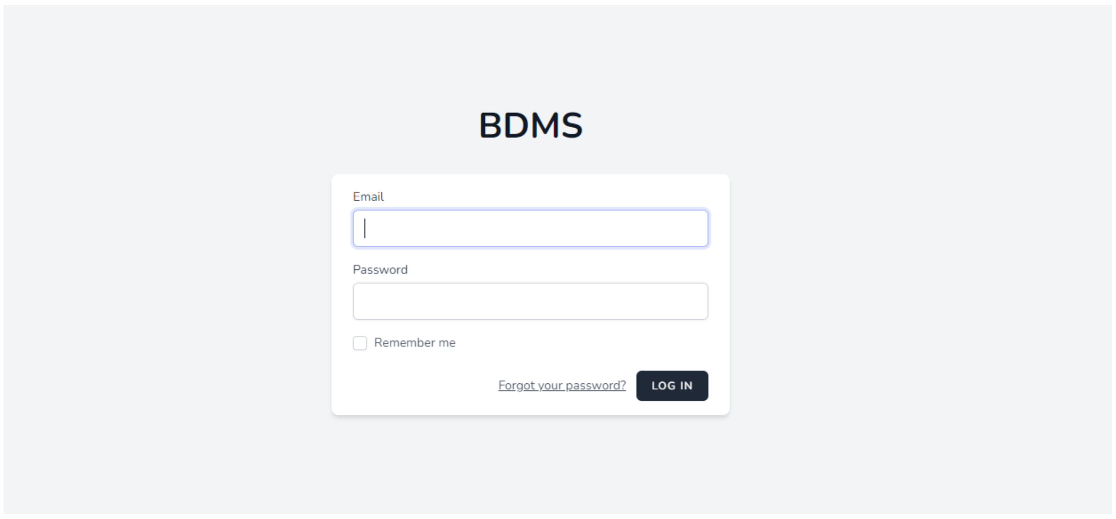
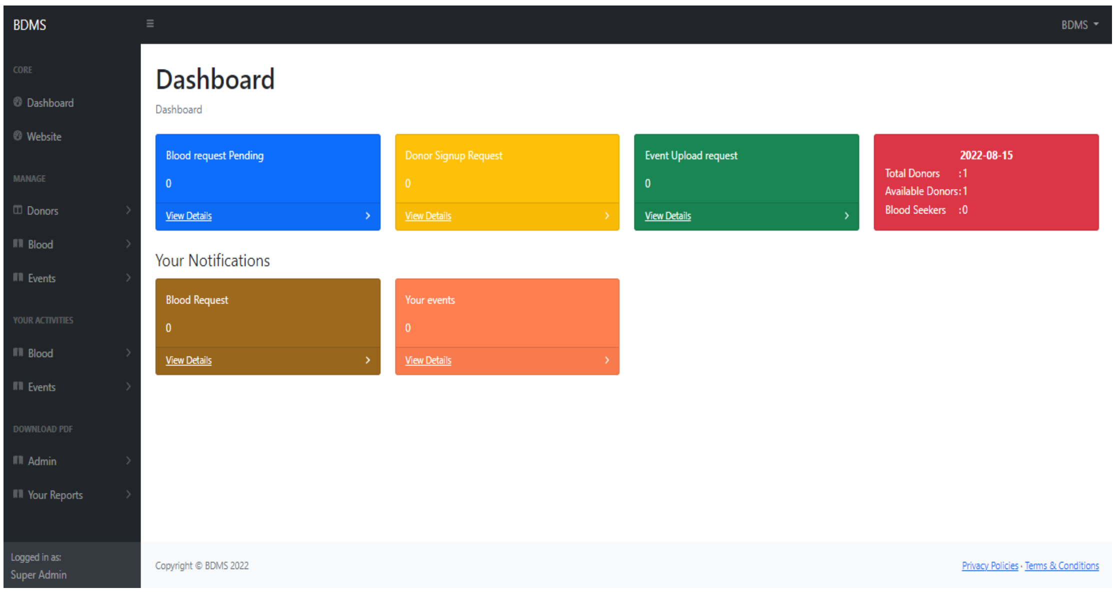


FIGURE 5.5 Log in page of blood donation management system.

Admin Dashboard: This is the admin dashboard page for admins. If user role is admin this panel will be visible.



Dashboard

Dashboard

Blood request Pending
0
View Details

Donor Signup Request
0
View Details

Event Upload request
0
View Details

2022-08-15
Total Donors :1
Available Donors:1
Blood Seekers :0

Your Notifications

Blood Request
0
View Details

Your events
0
View Details

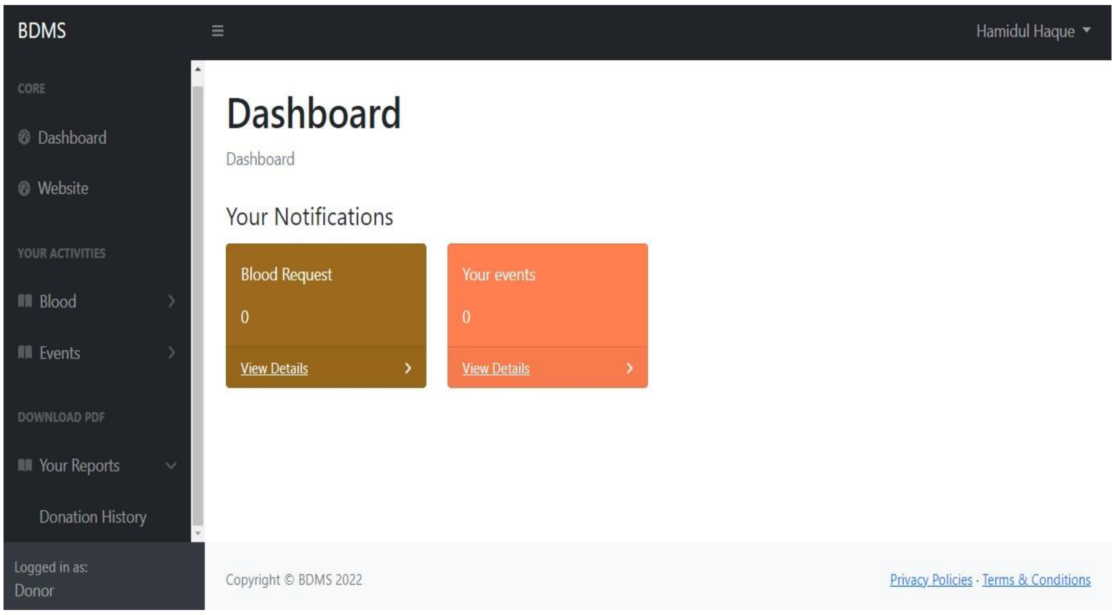
Logged in as:
Super Admin

Copyright © BDMS 2022

Privacy Policies · Terms & Conditions

FIGURE 5.6 Admin dashboard of blood donation management system.

Donor Dashboard: This is the donor dashboard page for donors. If user role is donor this panel will be visible.



Dashboard

Dashboard

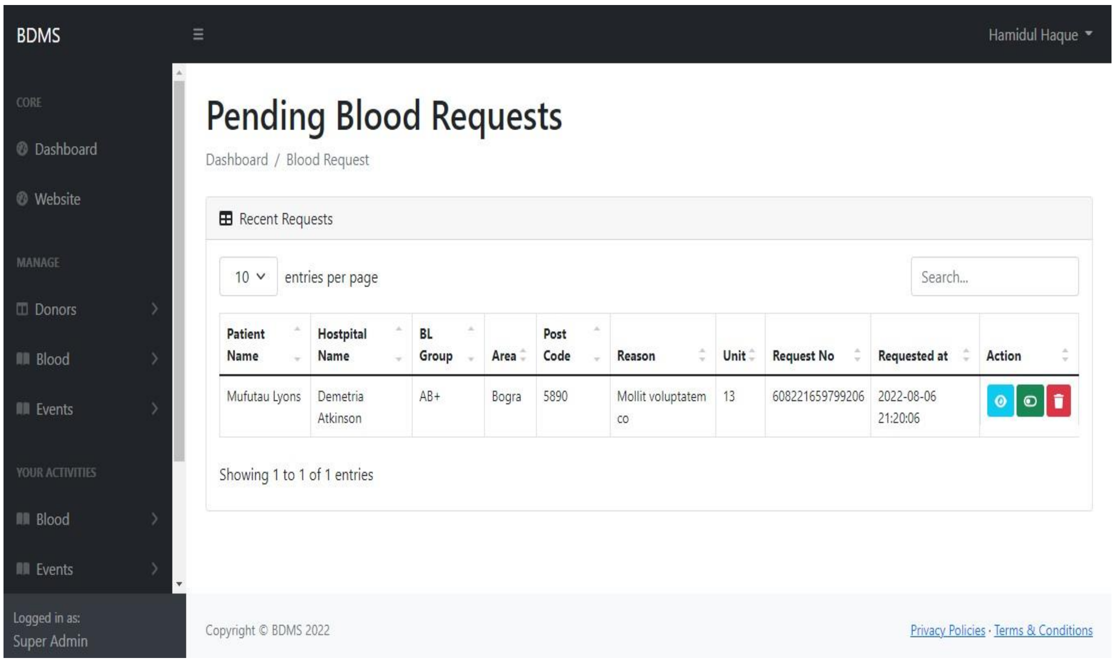
Your Notifications

Blood Request
0

Your events
0

Copyright © BDMS 2022

FIGURE 5.7 Donor dashboard of blood donation management system.



Pending Blood Requests

Dashboard / Blood Request

Recent Requests

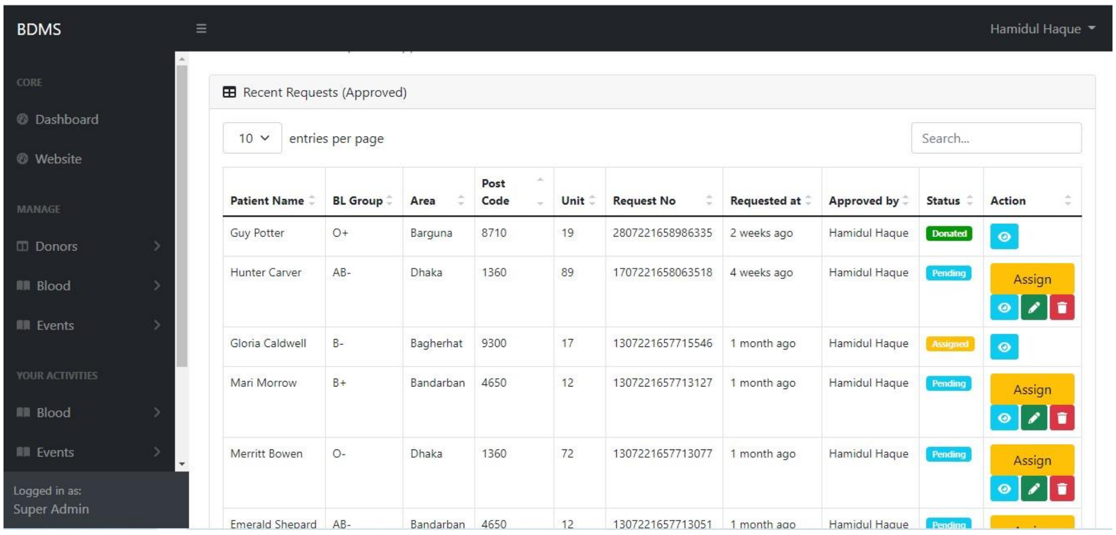
10 entries per page

Patient Name	Hostpital Name	BL Group	Area	Post Code	Reason	Unit	Request No	Requested at	Action
Mufutau Lyons	Demetria Atkinson	AB+	Bogra	5890	Mollit voluptatem co	13	608221659799206	2022-08-06 21:20:06	

Showing 1 to 1 of 1 entries

Copyright © BDMS 2022

FIGURE 5.8 Blood requests management from admin panel.



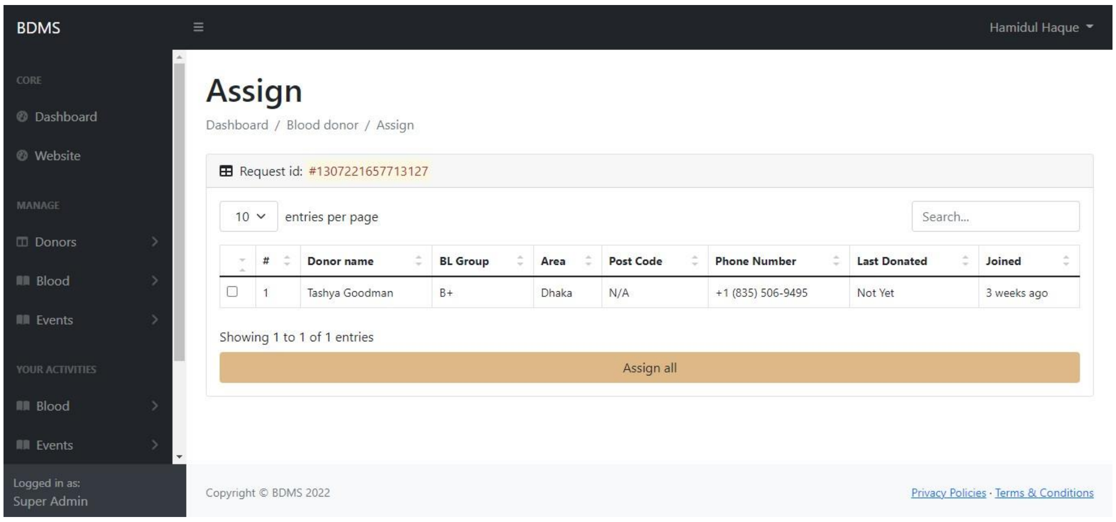
Recent Requests (Approved)

10 entries per page

Search...

Patient Name	BL Group	Area	Post Code	Unit	Request No	Requested at	Approved by	Status	Action
Guy Potter	O+	Barguna	8710	19	2807221658986335	2 weeks ago	Hamidul Haque	Donated	
Hunter Carver	AB-	Dhaka	1360	89	1707221658063518	4 weeks ago	Hamidul Haque	Pending	Assign
Gloria Caldwell	B-	Bagherhat	9300	17	1307221657715546	1 month ago	Hamidul Haque	Assigned	
Mari Morrow	B+	Bandarban	4650	12	1307221657713127	1 month ago	Hamidul Haque	Pending	Assign
Merritt Bowen	O-	Dhaka	1360	72	1307221657713077	1 month ago	Hamidul Haque	Pending	Assign
Emerald Shepard	AB-	Bandarban	4650	12	1307221657713051	1 month ago	Hamidul Haque	Pending	

FIGURE 5.9 Approved blood requests management from admin panel.



Assign

Dashboard / Blood donor / Assign

Request id: #1307221657713127

10 entries per page

Search...

	#	Donor name	BL Group	Area	Post Code	Phone Number	Last Donated	Joined
☐	1	Tashya Goodman	B+	Dhaka	N/A	+1 (835) 506-9495	Not Yet	3 weeks ago

Showing 1 to 1 of 1 entries

Assign all

Copyright © BDMS 2022

Privacy Policies · Terms & Conditions

FIGURE 5.10 Assign donors to blood requests from admin panel.

5.3 SUMMARY

The BDMS website is created and built to address all of the problem descriptions. We reached our aim after putting our thoughts and ideas into action. As a result, the outcomes are quite well defined. The outcome has been described and shown in this chapter. Please visit our GitLab repository and follow the instructions to install the project. [GITLAB, gitlab.com/hamidulhaque]

CHAPTER SIX
CONCLUSION & RECOMMENDATIONS

6.1 PROJECT OUTCOMES

The website has been functional and can be used by the users. But the website might face some issues in the future to upgrade. The outcomes of the developed are:

i) Smart and simple design of the website.

ii) User friendly interface of the website.

iii) No useless features.

iv) Blood requests in 'Guest mode'.

v) Notifications by email.

vi) Donor assign.

vii) PDF download.

6.2 LIMITATIONS OF THE PROJECT

The system is not compatible with IOS mobile devices and Android operating devices. Therefore, for IOS mobile phones and Android users may not be able to use the site in their mobile phone and experience the system. The IOS and The Android mobile phones users have to access the website using mobile phone supported browsers. In the future we'll develop the application for both mobile operating systems IOS and Android. Users must need the internet on their device like desktop, laptop and iPad etc. and a web browser to access the website on their device.

6.3 RECOMMENDATIONS

For future enhancement, there are a few suggestions to improve the website system. Systems should be analyzed and designed based on current situation demands and design trends of then. Some recommendations can be done in future:

i) Geo location can be added to bring actual locations.

ii) Donors request criteria can be updated.

iii) Phone numbers can be verified by OTP.

iv) More admin tools can be added.

v) Charity Donations (money donations from organization/individual) can be added.

vi) Donors review can be added

vii) Blood banks, Organization can be added to the roles.

REFERENCES

BADHAN - A Voluntary Blood Donor's Organization.(n.d.).BADHON. Retrieved August 18, 2022, from https://badhan.org/

Bloodworks Northwest | Donate Blood. (n.d.).Bloodworks Northwest. Retrieved August 18, 2022, from https://www.bloodworksnw.org/

American Red Cross.(n.d.). Help Those Affected by Disasters. Retrieved August 18, 2022, from https://www.redcross.org/

BDMS_GROUP, B. D. M. S. (n.d.). *HamidulHaque / BDMS-Blood Donation Management-system* ·.BDMS-GITLAB. Retrieved August 18, 2022, from https://gitlab.com/hamidulhaque/bdms-blood-donation-management-system

APPENDIX A
EXAMPLE CODES

CODES FOR LOGIN PAGE

```php
<?php

namespaceApp\Http\Controllers\Auth;

use App\Http\Controllers\Controller;
use App\Http\Requests\Auth\LoginRequest;
use App\Providers\RouteServiceProvider;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\Auth;

classAuthenticatedSessionControllerextendsController
{

    publicfunctionstore(LoginRequest$request)
    {
        $request->authenticate();

        $request->session()->regenerate();

        returnredirect()->intended(RouteServiceProvider::HOME);
    }


    publicfunctiondestroy(Request$request)
    {
        Auth::guard('web')->logout();

        $request->session()->invalidate();

        $request->session()->regenerateToken();

        returnredirect('/');
    }
}
```

CODES FOR REGISTRATION PAGE

```php
<?php

namespaceApp\Http\Controllers\Auth;

use App\Http\Controllers\Controller;
use App\Models\User;
use App\Providers\RouteServiceProvider;
use Illuminate\Auth\Events\Registered;
use Illuminate\Http\Request;
use Illuminate\Support\Facades\Auth;
use Illuminate\Support\Facades\Hash;
use Illuminate\Validation\Rules;
useImage;
useAlert;

classRegisteredUserControllerextendsController
{

    publicfunctioncreate()
    {
        returnview('auth.register');
    }
publicfunctionstore(Request$request)
    {
        $request->validate([
            'name' =>'required|string|max:255',
            'email' =>'required|string|email|max:255|unique:users',
            'password' => ['required', 'confirmed', Rules\Password::defaults()],
            'blood_group' =>'required',
            'nid_number'=>'required|numeric|unique:users',
            'phone'=>'required',
            'dob'=>'required|date|before:18 years ago',
            'gender'=>'required',
            'division'=>'required',
            'district'=>'required',
            'thana'=>'required',
            'postOffice'=>'required',
            'postCode'=>'required|numeric',
            'terms'=>'required',
            'profile_image' =>'required|mimes:jpg,png|min:5|max:2048',

            'nid_image' =>'required|mimes:jpg,png|min:5|max:512',
        ]);
        $default_role = 3;
        $requestData = $request->all();

        if ($request->hasFile('profile_image') &&$request->hasFile('nid_image')) {

            $profile_image = $request->file('profile_image');
            $nid_image = $request->file('nid_image');


            $nid_image_Name = time() . '.' . $nid_image-
>getClientOriginalExtension();
            $profile_image_Name = time() . '.' . $profile_image-
>getClientOriginalExtension();
            Image::make($request->file('profile_image'))
            ->resize(300, 200)
            ->save(storage_path() . '/app/public/users/profile/'
.$profile_image_Name);

            Image::make($request->file('nid_image'))
                ->resize(300, 200)
                ->save(storage_path() . '/app/public/users/nid/' .$nid_image_Name);
            $requestData['profile_image'] = $profile_image_Name;
            $requestData['nid_image'] = $nid_image_Name;
        }
    $user = User::create([
        'name' =>$request->name,
        'email' =>$request->email,
        'password' =>Hash::make($request->password),
        'blood_group'=>$request->blood_group,
        'role_id' =>$default_role,
        'nid_number' =>$request->nid_number,
        'total_donated'=>0,

    ]);
    $user->profile()->create([
        'phone' =>$request->phone,
        'phone2' =>$request->phone2,
        'father'=>$request->father,
        'mother'=>$request->mother,
        'dob'=>$request->dob,
        'gender'=>$request->gender,
        'division'=>$request->division,
        'district'=>$request->district,
        'thana'=>$request->thana,
        'postOffice'=>$request->postOffice,
        'postCode' =>$request->postCode,
        'profile_image' =>$requestData['profile_image'],
        'nid_image' =>$requestData['nid_image'],
    ]);

    event(newRegistered($user));
    auth()->login($user);
    Auth::login($user);
    returnredirect('verify-email');
    returnredirect()->back()->withMessage('Your signup request is submitted! We
will mail you soon');
    //   return redirect()->route('verification.notice');
    }
}
```

CODES FOR BLOOD SEEKING REQUEST CONTROLLER

```
<?php

namespaceApp\Http\Controllers;

use App\Models\BloodRequest;
use App\Http\Requests\StoreBloodRequestRequest;
use App\Http\Requests\UpdateBloodRequestRequest;
use App\Models\User;
use Carbon\Carbon;
use Illuminate\Http\Request;

use Illuminate\Support\Facades\Auth;
use Illuminate\Support\Facades\Date;
use Illuminate\Support\Facades\Mail;

classBloodRequestControllerextendsController
{

    privatefunctiondonor()
    {
        $role_id = 3;
        if (Auth::user()->role_id == $role_id) {
            returntrue;
        } else {
            returnfalse;
        }
    }
    protectedfunctionprotect()
    {
        if ($this->donor()) {
            abort(404);
        };
    }
publicfunctionpending()
    {
        $this->protect();
        $bloodRequests = BloodRequest::whereNull('approved_by')-
>whereNull('status')->whereNull('not_donated_reason')->whereNull('rejected_by')-
>latest()->get();
        returnview('backend/blood/not-approved', compact('bloodRequests'));
    }
  publicfunctionapprove($id)
    {
        $this->protect();
        $bloodRequest = BloodRequest::findOrFail($id);
        // $bloodRequest['approved_by'] = auth()->user()->id;
        $done = $bloodRequest->update([
            'approved_by' =>Auth::id()
        ]);
        // sending mail to seekers;
        if ($done) {
            $data = [
                'request_no' =>$bloodRequest['request_no'],
                'patient_name' =>$bloodRequest['patient_name'],
                'blood_group' =>$bloodRequest['blood_group'],
                'hospital_name' =>$bloodRequest['hospital_name'],
                'contact_name' =>$bloodRequest['contact_name'],
            ];
            $user['to'] = $bloodRequest['email'];
            Mail::send('mail.request-approve', $data, function ($messages) use
($user) {
                $messages->to($user['to']);
                $messages->subject('Your Request has been Approved');
            });
        }

        returnredirect()->back()->withMessage('Successfully Approved! To assign
donor please visit approved blood requests');
    }
publicfunctionreject(Request$request)
    {
        $this->protect();
        $request->validate(
            [
                'reject_reason' => ['required']
            ]
        );
        if ($request->reject_reason || $request->reject_reason2) {
            if($request->reject_reason2){
                $request->reject_reason = $request->reject_reason2;
            }
            $bloodRequest = BloodRequest::findOrFail($request->id);
            // $bloodRequest['approved_by'] = auth()->user()->id;
            $done = $bloodRequest->update([
                'rejected_by' =>Auth::id(),
                'reject_reason' =>$request->reject_reason
            ]);

            //sending mail
            if ($done) {
                $data = [
                    'request_no' =>$bloodRequest['request_no'],
                    'patient_name' =>$bloodRequest['patient_name'],
                    'blood_group' =>$bloodRequest['blood_group'],
                    'hospital_name' =>$bloodRequest['hospital_name'],
                    'contact_name' =>$bloodRequest['contact_name'],
                ];
                $user['to'] = $bloodRequest['email'];
                Mail::send('mail.request-reject', $data, function ($messages) use
($user) {
                    $messages->to($user['to']);
                    $messages->subject('Your Request has been Rejected');
                });
            }
            returnredirect()->back()->withMessage('Rejected!');
        } else {
            returnredirect()->back()->withErrors('NO REASON SPECIFIED!PLEASE MARK A
REASON TO REJECT');
        }
    }
  publicfunctionallRequests()
    {
        $this->protect();
        $bloodRequests = BloodRequest::whereNotNull('approved_by')->latest()-
>get();
        returnview('backend/blood/all', compact('bloodRequests'));
    }

        publicfunctionassignIndex(BloodRequest$bloodRequest)
    {
        $this->protect();

        $donationAvail = Carbon::parse(Carbon::now()->subMonths(3));

        $blood_group = $bloodRequest['blood_group'];
        // dd($blood_group);

        $donors = User::where([
            ['blood_group', $blood_group],
            ['last_donated', '<=', $donationAvail],
        ])
            ->orWhere([
                ['blood_group', $blood_group],
                ['rejected_by', null],
                ['last_donated', null]
            ])
            ->whereNull('status')
            ->latest()
            ->get();
        returnview('backend.blood.assign-index', compact('donors',
'bloodRequest'));
    }
    publicfunctionassignDonor(Request$request, BloodRequest$bloodRequest)
    {
        $this->protect();

        $bloodRequest->donors()->attach($request->donor_ids);
        $done = $bloodRequest->update([
            'status' =>1
        ]);

        if ($done) {
            $donors = User::where('id', $request->donor_ids)->get();
            foreach ($donors as $donor) {
                $data = [
                    'donor_name' =>$donor['name'],
                ];
                Mail::send('mail.request-donor', $data, function ($messages) use
($donor) {
                    $messages->to($donor['email']);
                    $messages->subject('DONOR:You have new blood requests');
                });
            };

            // sending mail to seekers
            $data = [
                'request_no' =>$bloodRequest['request_no'],
                'contact_name' =>$bloodRequest['contact_name'],
            ];
            Mail::send('mail.request-assigned', $data, function ($messages) use
($bloodRequest) {
                $messages->to($bloodRequest['email']);
                $messages->subject('Your request is assigned to donors');
            });
        }



        returnredirect()->route('blood-request-all')->withMessage('Successfully
Assigned!');
    }


    publicfunctionshow(BloodRequest$bloodRequest)
    {
        $this->protect();

        returnview('backend/blood/view', compact('bloodRequest'));
    }

    publicfunctionedit(BloodRequest$bloodRequest)
    {
        $this->protect();
        //
    }

    publicfunctionupdate(UpdateBloodRequestRequest$request,
BloodRequest$bloodRequest)
    {
        $this->protect();
        //
    }


    publicfunctiondestroy(BloodRequest$bloodRequest)
    {
        $this->protect();
        //
    }
}
```

PDF DOWNLOAD CONTROLLER

```
<?php

namespaceApp\Http\Controllers;

use App\Models\BloodRequest;
use App\Models\Event;
use App\Models\User;
use Carbon\Carbon;
use Illuminate\Http\Response;
use Illuminate\Support\Facades\Auth;
use Illuminate\Support\Facades\DB;
usePDF;
classPDFControllerextendsController
{


    //protecting datas
    privatefunctiondonor()
    {
        $role_id = 3;
        if (Auth::user()->role_id == $role_id) {
            returntrue;
        } else {
            returnfalse;
        }
    }
    protectedfunctionprotect()
    {
        if ($this->donor()) {
            abort(404);
        };
    }
    publicfunctionseekerProfile($id)
    {
        $profile = DB::table('blood_requests')->where('id', $id)->get()->first();
        $pdf = PDF::loadView('pdf.seeker', compact('profile'));
        return$pdf->download('seekerprofile.pdf');
    }
    //monthly report
    publicfunctionmonthReport()
    {
        $this->protect();
        $date = \Carbon\Carbon::now();

        $lastMonth=  $date->subMonth()->format('m');
        $lastMonthName=  $date->subMonth()->format('m/Y');
        $bloodReports = BloodRequest::whereMonth('created_at', '=', $lastMonth)
            ->whereNotNull('approved_by')
            ->orWhereNotNull('rejected_by')->get();
        // a+
        $aP_blood = BloodRequest::whereMonth('created_at', '=', $lastMonth)

            ->where('blood_group', 'A+')
            ->get();
        // b+
        $bP_blood = BloodRequest::whereMonth('created_at', '=', $lastMonth)

            ->where('blood_group', 'B+')
            ->get();
        // o+
        $oP_blood = BloodRequest::whereMonth('created_at', '=', $lastMonth)

            ->where('blood_group', 'O+')
            ->get();
        // ab+
        $abP_blood = BloodRequest::whereMonth('created_at', '=', $lastMonth)
            ->whereNotNull('approved_by')
            ->orWhereNotNull('rejected_by')
            ->where('blood_group', 'AB+')
            ->get();
        // a-
        $aN_blood = BloodRequest::whereMonth('created_at', '=', $lastMonth)

            ->where('blood_group', 'A-')
            ->get();
        // b-
        $bN_blood = BloodRequest::whereMonth('created_at', '=', $lastMonth)

            ->where('blood_group', 'B-')
            ->get();
        // o-
        $oN_blood = BloodRequest::whereMonth('created_at', '=', $lastMonth)

            ->where('blood_group', 'O-')
            ->get();
// ab-
        $abN_blood = BloodRequest::whereMonth('created_at', '=', $lastMonth)

            ->where('blood_group', 'AB-')
            ->get();

        // a+
        $aP_donor = User::whereMonth('created_at', '=', $lastMonth)

            ->where('blood_group', 'A+')
            ->get();
        // a+
        $bP_donor = User::whereMonth('created_at', '=', $lastMonth)
            ->where('blood_group', 'A+')
            ->get();
        // ab+
        // o+
        $oP_donor = User::whereMonth('created_at', '=', $lastMonth)

            ->where('blood_group', 'O+')
            ->get();
        // ab+
        $abP_donor = User::whereMonth('created_at', '=', $lastMonth)

            ->where('blood_group', 'AB+')
            ->get();
        // a-
        $aN_donor = User::whereMonth('created_at', '=', $lastMonth)

            ->where('blood_group', 'A-')
            ->get();
        // b-
        $bN_donor = User::whereMonth('created_at', '=', $lastMonth)

            ->where('blood_group', 'B-')
            ->get();
        // o-
        $oN_donor = User::whereMonth('created_at', '=', $lastMonth)

            ->where('blood_group', 'O-')
            ->get();


        // ab-
        $abN_donor = User::whereMonth('created_at', '=', $lastMonth)

            ->where('blood_group', 'AB-')
            ->get();


        $eventReports = Event::whereNotNull('approved_by')-
>whereMonth('created_at', '=', $lastMonth)->get();
        $donorReports = User::whereMonth('created_at', '=', $lastMonth)
            ->whereNotNull('approved_by')
            ->orWhereNotNull('rejected_by')->get();
        //generating pdf
        $pdf = PDF::loadView('pdf.monthly-report', compact(
            'bloodReports', 'aP_blood','bP_blood', 'oP_blood',
            'abP_blood', 'aN_blood', 'bN_blood', 'oN_blood',
            'abN_blood', 'aP_donor', 'bP_donor', 'oP_donor',
            'abP_donor', 'aN_donor', 'bN_donor', 'oN_donor',
            'abN_donor', 'donorReports', 'eventReports', 'lastMonthName'
        ));

        $pdf->setPaper('A4', 'portait');
        $name = 'bdms-report' .$date->subMonth()->format('F') . $date->format('Y')
. '.pdf';
        return$pdf->download($name);
```

```
    }
    //total report
    publicfunctiontotalReport()
    {
        $this->protect();
        $bloodReports = BloodRequest::all();
        $donorReports = User::all();
        $eventReports = Event::all();
        $pdf = PDF::loadView('pdf.total-report',
compact('bloodReports','eventReports', 'donorReports'));
        $name = 'bdms-total-report'.Carbon::now()->format('d/m/Y').'.pdf';
        return$pdf->download($name);
    }
    //yearly report
    publicfunctionyearReport()
    {
        $date = \Carbon\Carbon::now();
        $lastYear=  $date->subYear()->format('Y');
        $bloodReports = BloodRequest::whereYear('created_at', '=', $lastYear)
            ->whereNotNull('approved_by')
            ->orWhereNotNull('rejected_by')->get();
        $donorReports = User::whereYear('created_at', '=', $lastYear)
            ->whereNotNull('approved_by')
            ->orWhereNotNull('rejected_by')->get();
        $eventReports = Event::whereYear('created_at', '=', $lastYear)
            ->whereNotNull('approved_by')
            ->orWhereNotNull('deleted_by')->get();

        $name = 'bdms-year-report' .$lastYear . '.pdf';
        $pdf = PDF::loadView('pdf.yearly-report',
compact('bloodReports','eventReports', 'donorReports','lastYear'));
        return$pdf->download($name);
    }
    //my donation report
    publicfunctionmyDonation()
    {
        $date = Carbon::now()->format('d-m-Y');
        $user = Auth::user();
        $user_id = Auth::id();
        $donations = BloodRequest::where('donor_id',$user_id)->get()->all();
        $pdf = PDF::loadView('pdf.myprofile-report', compact('user','donations'));
        $name = Auth::user()->name.'.pdf';
        return$pdf->download($name);
    }
}
```

B.Sc in CSE

BLOOD DONATION MANAGEMENT SYSTEM

Fall - 2021

SPINE TEXT

Spine text should contain:

1) BSc/MSc in CSE

2) 10 blank spaces

3) Title of the project

4) UU, Semester & year

Formatting of the texts:

1) Font: Calibri (Body)

2) Bold

3) UPPER CASE

4) Font size: 16 points or less to adjust within the spine.

BSc in CSE

BLOOD DONATION MANAGEMENT SYSTEM

UU, SUMMER 22