

DESIGN A MODEL OF APPLICATION FOR BLOOD BANK MANAGEMENT SYSTEM

Ahin Omar Fatah

Bachelor's thesis
2024



Tomas Bata University in Zlín
Faculty of Applied Informatics

Tomas Bata University in Zlín
Faculty of Applied Informatics
Department of Informatics and Artificial Intelligence

Academic year: 2023/2024

ASSIGNMENT OF BACHELOR THESIS

(project, art work, art performance)

| | |
|-------------------------------|---|
| Name and surname: | Ahin Omar Fatah |
| Personal number: | A19898 |
| Study programme: | B0613A140021 Software Engineering |
| Type of Study: | Full-time |
| Work topic: | Návrh modelu aplikace pro krevní banku |
| Work topic in English: | Designing a Model of an Application for a Blood Bank Management System |

Theses guidelines

1. Prepare the literature review of the thesis topic.
2. Briefly describe the technologies that will be used.
3. Describe and analyse the requirements for the solution.
4. Create the HTML prototype of the proposed application.
5. Describe the user controls of the HTML prototype.

Form processing of bachelor thesis: **printed/electronic**

Recommended resources:

1. ARLOW, Jim; NEUSTADT, Ila. UML 2 and the unified process: practical object-oriented analysis and design. Pearson Education, 2005. JOHNSON, Glenn. Programming in HTML5 with JavaScript and CSS3: training guide. Redmond, Wash.: Microsoft, 2013. ISBN 978-0735674387.
2. JOHNSON, Glenn. Programming in HTML5 with JavaScript and CSS3: training guide. Redmond, Wash.: Microsoft, 2013. ISBN 978-0735674387.
3. UNHELKAR, Bhuvan. Software engineering with uml Auerbach Publications, 2017.
4. LETT, Jacob. Bootstrap 4 Quick Start: A Beginners Guide to Building Responsive Layouts with Bootstrap 4 Bootstrap Creative, 2018.
5. AKOBUS, Benjamin. Mastering Bootstrap 4: Master the latest version of Bootstrap 4 to build highly customized responsive web apps Packt Publishing Ltd, 2018.
6. BEN-GAN, Itzik; DAVIDSON, Louis; VARGA, Stacia. MCSA SQL Server 2016 Database Development Exam Ref 2-pack: Exam Refs 70-761 and 70-762 Microsoft Press, 2017.

Supervisors of bachelor thesis: **doc. Ing. Petr Šilhavý, Ph.D.**
Department of Computer and Communication Systems

Date of assignment of bachelor thesis: **November 5, 2023**

Submission deadline of bachelor thesis: **May 13, 2024**



doc. Ing. Jiří Vojtěšek, Ph.D. m.p.
Dean

prof. Mgr. Roman Jašek, Ph.D., DBA m.p.
Head of Department

In Zlín January 5, 2024

I hereby declare that:

- I understand that by submitting my Bachelor's Thesis, I agree to the publication of my work according to Law No. 111/1998, Coll., On Universities and on changes and amendments to other acts (e.g. the Universities Act), as amended by subsequent legislation, without regard to the results of the defence of the thesis.
- I understand that my Bachelor's Thesis will be stored electronically in the university information system and be made available for on-site inspection, and that a copy of the Bachelor's Thesis will be stored in the Reference Library of the Faculty of Applied Informatics, Tomas Bata University in Zlín, and that a copy shall be deposited with my Supervisor.
- I am aware of the fact that my Bachelor's Thesis is fully covered by Act No. 121/2000 Coll. On Copyright, and Rights Related to Copyright, as amended by some other laws (e.g. the Copyright Act), as amended by subsequent legislation; and especially, by §35, Para. 3.
- I understand that, according to §60, Para. 1 of the Copyright Act, TBU in Zlín has the right to conclude licensing agreements relating to the use of scholastic work within the full extent of §12, Para. 4, of the Copyright Act.
- I understand that, according to §60, Para. 2, and Para. 3, of the Copyright Act, I may use my work - Bachelor's Thesis, or grant a license for its use, only if permitted by the licensing agreement concluded between myself and Tomas Bata University in Zlín with a view to the fact that Tomas Bata University in Zlín must be compensated for any reasonable contribution to covering such expenses/costs as invested by them in the creation of the thesis (up until the full actual amount) shall also be a subject of this licensing agreement.
- I understand that, should the elaboration of the Bachelor's Thesis include the use of software provided by Tomas Bata University in Zlín or other such entities strictly for study and research purposes (i.e. only for non-commercial use), the results of my Bachelor's Thesis cannot be used for commercial purposes.
- I understand that, if the output of my Bachelor's Thesis is any software product(s), this/these shall equally be considered as part of the thesis, as well as any source codes, or files from which the project is composed. Not submitting any part of this/these component(s) may be a reason for the non-defence of my thesis.

I herewith declare that:

- I have worked on my thesis alone and duly cited any literature I have used. In the case of the publication of the results of my thesis, I shall be listed as co-author.
- That the submitted version of the thesis and its electronic version uploaded to IS/STAG are both identical.

In Zlín; dated: 13/05/2024

Ahin Omar Fatah m.p.
Student's Signature

ABSTRAKT

V této práci je navržen model aplikace pro systém řízení krevní banky. Práce začíná teoretickou částí, která obsahuje informace o internetu, webových stránkách a základních technologiích používaných pro tvorbu webových stránek. V jedné kapitole jsou také rozebrány možnosti zabezpečení webových stránek. V analytické části jsou zobrazeny a popsány UML diagramy. Poté je uvedena část s HTML prototypem webu, s prezentací frontendu a stručným popisem ukázky kódu. HTML prototyp a UML diagramy jsou také uloženy v elektronické příloze. Na závěr jsou prezentovány a diskutovány výsledky práce.

Klíčová slova: krevní banka, návrh, HTML prototyp, model, UML

ABSTRACT

In this thesis, model of application for blood bank management system is designed. The thesis starts with the theoretical part which contains information about Internet, websites, and basic technologies used for web design. Website security options are also discussed in one chapter. In the analysis part, UML diagrams are shown and described. The section with the website HTML prototype follows with frontend presentation and the brief description of a code sample. The HTML prototype and UML diagrams are also stored in the electronic attachment. Finally, results of the thesis are presented and discussed.

Keywords: blood bank, design, HTML prototype, model, UML

Contents

| | |
|--|-----------|
| I INTRODUCTION..... | 8 |
| II I. 10 | |
| III THEORY..... | 10 |
| 1.1 EXISTING BLOOD MANAGEMENT SOLUTIONS..... | 11 |
| 1.1.1 WELLSKY BLOOD BANK SOFTWARE | 11 |
| 1.1.2 EZOVION HEALTHCARE’S BLOOD BANK MANAGEMENT SYSTEM..... | 11 |
| 1.2 HTML | 11 |
| 1.2.1 HTML TAGS | 12 |
| 1.2.2 HTML ATTRIBUTES | 12 |
| 1.2.3 HTML ELEMENTS | 13 |
| 1.2.4 IN GENERALS | 13 |
| 1.2.5 SEMANTIC HTML | 13 |
| 1.2.6 HTML STRUCTURE | 14 |
| 1.2.7 HEAD..... | 14 |
| 1.2.8 BODY | 15 |
| 1.2.9 ADVANTAGES OF HTML..... | 15 |
| 1.3 CSS..... | 15 |
| 1.3.1 CSS SELECTORS | 16 |
| 1.3.2 SELECTOR LIST | 16 |
| 1.3.3 TYPES OF SELECTORS | 16 |
| 1.4 GRID VS FLEXBOX | 17 |
| 1.5 CSS BOX MODEL | 18 |
| 1.5.1 BLOCK AND INLINE BOXES..... | 18 |
| 1.5.2 MARGIN | 18 |
| 1.5.3 PADDING | 19 |
| 1.5.4 BORDERS..... | 19 |
| 1.6 RESPONSIVE WEB DESIGN | 19 |
| 1.7 JAVASCRIPT | 20 |
| 1.7.1 LIBRARIES VS FRAMEWORKS IN JAVASCRIPT | 20 |
| 1.8 SERVER-SIDE PROGRAMMING | 20 |
| 1.9 JAVASCRIPT IMPACT ON SITE PERFORMANCE | 21 |
| 1.10 JAVASCRIPT DEBUGGING TECHNIQUES | 21 |
| 1.10.1 BUGS | 22 |
| 1.10.2 DEBUGGING..... | 22 |
| 1.11 JAVASCRIPT FUNCTIONS | 22 |
| 1.12 ADVANTAGES OF JAVASCRIPT | 23 |
| 1.13 BOOTSTRAP..... | 23 |

| | |
|--|------------|
| 1.13.1 WHY USE BOOTSTRAP | 24 |
| 1.13.2 ADVANTAGES OF BOOTSTRAP | 24 |
| 1.13.3 BOOTSTRAP CONTAINER | 25 |
| 1.13.4 BOOTSTRAP GRID SYSTEM | 25 |
| 2 SECURITY OPTIONS FOR WEB APPLICATION | 26 |
| 2.1 IMPORTANCE OF WEB APPLICATION SECURITY | 28 |
| 2.2 WEB APPLICATION FIRE WALLS (WAF) | 29 |
| IV II. | 30 |
| V ANALYSIS | 30 |
| 3 DESIGN OF A BLOOD BANK SYSTEM | 31 |
| 3.1 REQUIREMENT MODELS | 31 |
| 3.2 USE CASE MODEL | 43 |
| 3.2.1 USE CASES SPECIFICATIONS | 45 |
| 3.3 CLASS MODEL | 68 |
| 3.4 SEQUENCE MODEL | 70 |
| 4 VISUAL BLUEPRINT OF THE APP - HTML PROTOTYPE | 86 |
| 5 WEBSITE PRESENTATION | 89 |
| 6 IMPLEMENTATION OF A BLOOD BANK SYSTEM WEBSITE | 96 |
| 6.1 SAMPLE OF THE HTML | 96 |
| 6.2 SAMPLE OF THE CSS | 102 |
| 6.3 SAMPLE OF THE JAVASCRIPT | 104 |
| 7 USER GUIDE FOR APPLICATION FEATURES AND FUNCTIONALITIES ... | 113 |
| 8 FUTURE ENHANCEMENTS AND IMPROVEMENTS | 126 |
| 8.1 FRONT-END IMPROVEMENTS | 126 |
| 8.2 BACK-END IMPROVEMENTS | 126 |
| 8.3 SECURITY IMPROVEMENTS | 127 |
| VI CONCLUSION | 128 |
| VII BIBLIOGRAPHY | 129 |
| VIII LIST OF ABBREVIATIONS | 132 |
| IX LIST OF FIGURES | 133 |
| X LIST OF TABLES | 135 |
| XI APPENDICES | 137 |

ACKNOWLEDGEMENTS

I am delighted to acknowledge the contributions of the many individuals who have helped shape this work and whose teachings have greatly influenced my professional development. I want to ex-press my deepest appreciation to Mr. Petr Silhavy, who has served as my teacher, mentor, and lead-er. I am truly grateful for all he has done for me.

Finally, I would like to extend my gratitude to God, as well as to my parents and friends, who have been a tremendous source of encouragement and support throughout my life, helping me to improve my skills and grow as a student.

I hereby declare that the print version of my bachelor's thesis and the electronic version of my thesis deposited in the IS/STAG system are identical.

INTRODUCTION

Implementing efficient, reliable, and user-friendly systems is essential for saving lives and improving health outcomes in the critical healthcare domain, specifically within blood bank management. This thesis focuses on developing an advanced Blood Bank Management System designed to streamline the processes of blood donation, storage, and distribution processes. By leveraging the latest technology, this project aims to address the challenges faced by current blood bank operations, including inventory management, donor records, and blood matching.

A comprehensive literature review forms the bedrock of this thesis, synthesising existing research, practices, and technological advancements in blood bank management. This review highlights the gaps and opportunities within the current frameworks, setting the stage for the proposed system's contribution to the field.

The technology stack selected for this project is chosen for its robustness, scalability, and compatibility with healthcare applications. The system will be developed using HTML5 and CSS3 for the front end to ensure an intuitive user interface. Due to its reliability in handling complex databases, MSSQL will be employed for data management. An in-depth analysis of these technologies will provide insight into their selection and how they integrate to build a cohesive and efficient management system.

Understanding and defining the system requirements is a pivotal phase of this project, involving meticulous analysis to ensure the Blood Bank Management System meets the specific needs of donors, recipients, and administrators. This section will detail the functional, non-functional, and technical requirements, providing a blueprint for the development process and ensuring the system's effectiveness and efficiency.

Creating an HTML prototype presents a practical visualisation of the proposed system, offering an early look at its functionality and user interface. The development process of this prototype, from conceptual design to its interactive functionalities, will be extensively documented. This prototype serves as a proof of concept and a platform for early user feedback and usability testing.

Lastly, the HTML prototype's user controls and interaction mechanisms will be elaborated upon. This section will dissect the design choices and functionalities provided to the system's users, ensuring the navigation and operations within the system are intuitive and meet

the high standards required for medical applications. The effectiveness of these controls will be evaluated against established usability criteria, ensuring the final system is both practical and user-friendly.

In sum, this thesis aims to contribute to the healthcare management field by delivering a Blood Bank Management System. Through detailed research, technology application, and user-centered design, this project seeks to enhance the efficiency and reliability of blood bank operations, ultimately supporting better health outcomes.

I. THEORY

1.1 Existing blood management solutions

The world of the internet is big, and organizations and business are racing to create their perfect product, there already exists blood management solutions. But of course, each has its own pros and cons.

1.1.1 WellSky blood bank software

WellSky Blood Bank Software is a comprehensive solution designed to enhance the safety and efficiency of blood bank operations. When talking about the pros of such software the software performs more than 60 built-in automated safety checks at critical times throughout the process, ensuring the highest level of safety and accuracy in blood management.

But it also has its cons, the software might be complex to use due to its intense amount of features it has, this might lead to bad user experience.¹

1.1.2 Ezovion Healthcare's Blood bank management system

Ezovion Healthcare provides a Blood Bank Management System that is intended to modernize and streamline the operations of blood banks. This system is part of their larger suite of hospital management solutions, which aims to improve patient care, increase productivity, and ensure cost control through digitalization. The program replaces traditional paperwork with a full digital solution that oversees all elements of blood bank operations, such as donor registration, blood stock management, blood collection, and request management. However, all of these benefits that are considered as pros of the software, Ezovion can be extremely complex, and might need special training for the staff. The software is highly expensive from both sides software and hardware.²

1.2 HTML

HTML, short, for Hypertext Markup Language is a textual markup language that has been carefully designed to structure and organize information within an HTML document. This specialized code acts as a link between a web server and a user's web browser providing instructions on how to display the various elements of a web page including text, images,

¹ Blood Bank Software | WellSky

² Blood Bank Management System | Ezovion Healthcare

multimedia components and more. HTML serves as the foundation of the World Wide Web playing a role in organizing and presenting web content. It functions as a blueprint that guides browsers in creating appealing and coherent representations of content. This ensures that the content appears exactly as intended by web designers or content creators. To definitively identify a file as HTML it must strictly adhere to rules regarding syntax, file format and naming conventions. These rules and standards are crucial for achieving compatibility across web browsers and platforms. They enable users to reliably access and interact with web content. Creating a website using HTML is accessible, to both beginners and experienced web developers. It involves the process of inserting HTML tags into a text document. These tags act as markers that give meaning to parts of the content. These tags, often appearing as keywords enclosed in angle brackets help convey the structure and organization of the content. Once the HTML document is complete it can be easily published online allowing people, from over the world to access the information it contains instantly.

In terms HTML is the core language that empowers the internet by enabling the creation of a range of websites. It provides a way to organize and communicate information, in form making it possible to build anything from basic personal web pages to intricate and interactive web applications. [1][2][3][4][5]

1.2.1 HTML Tags

HTML tags play a role, in crafting and organizing web content. They provide control over text formatting, allowing for options such as bold, italics and underlining. HTML tags also enable the creation of lists, headings, paragraphs and the incorporation of links and images. Developers can effortlessly design tables for tabular data create forms and divide content into sections or containers using <div> tags. Additionally HTML allows for comments, within the code to aid developers. When combined with CSS HTML extends its styling capabilities by separating structure (HTML) from presentation (CSS) making it possible to achieve customized webpage layouts. [6]

1.2.2 HTML Attributes

Attributes are elements of HTML tags that can be included in the opening tag of an element. These attributes, such, as "style " "id," and "class," provide context and functionality to the element. For example the "style" attribute allows for inline styling, which means you can specify properties like colors and fonts within the tag. The "id" and "class" attributes play

roles in selecting and manipulating elements with JavaScript making it easier to create interactions and make modifications, on webpages. Attributes enhance the versatility of HTML elements by enabling them to convey not content but instructions and behavior ultimately improving the interactivity and aesthetics of webpages. [6]

1.2.3 HTML Elements

HTML elements serve as the building blocks of web documents. They consist of tags, characters, content and closing tags. While most elements have both an opening tag and a closing tag there are ones, like the `` tag used for adding images that are considered elements as they don't require a closing tag. It's important to understand that although tags and HTML components have similarities they are not exactly the thing. An HTML element encompasses not the tags. Also includes the content enclosed between the opening and closing tags. This combination forms an unit that defines the structure and functionality of an element, on a webpage.

1.2.4 In Generals

HTML elements consist of tags and attributes. Tags are used to indicate the start. In cases the end of an element defining its boundaries. These tags provide instructions, to web browsers on how to display the content within them. Certain elements, such as the `` tag do not require closing tags since they are self-contained. Attributes on the hand are properties or settings applied to elements and are only placed within the opening tags. Attributes offer information or instructions for the element's behavior. This distinction between tags and attributes is crucial, in HTML as it allows web developers to define both the structure and behavior of web content resulting in visually appealing webpages. [6]

1.2.5 Semantic HTML

Semantic HTML is a practice, in web development that involves using tags to describe the function and meaning of content. Of using tags like `<div>` and `` it's preferable to use semantically descriptive tags such as `<header>`, `<nav>`, `<main>`, `<section>` `<footer>` and `<article>`. This approach improves the clarity and organization of the webpages structure making it easier for both developers and search engines to understand. When search engines encounter content marked up with HTML it becomes simpler for them to index. Categorize the information on the page ultimately improving its SEO ranking.

Not does a website benefit from SEO when using semantic HTML tags but it also enhances the overall quality of the web experience. By using tags with purpose and structure developers provide valuable information about the content. This increased clarity makes websites more flexible, adaptable and easier to maintain or style.

Importantly semantic HTML plays a role in accessibility for users who rely on assistive technologies like screen readers. The descriptive nature of these tags allows screen readers to provide an representation of the contents meaning and context to users, with disabilities.

Ensuring inclusivity, in web content is essential to reach an audience and adhere to the standards, in web development. These standards prioritize both user friendliness and visibility on search engines. [7]

1.2.6 HTML Structure

HTML, also known as Hypertext Markup Language utilizes a hierarchy that consists of elements and tags to construct web documents. At its core an HTML document starts with a declaration called Document Type Declaration (`<!DOCTYPE>`) followed by an `<html>` element. This `<html>` element contains the `<head>` section which holds metadata like character encoding and page title and the `<body>` section where visible content such, as text, images and links are placed. The structure and meaning of the document are defined by elements and tags, within it allowing for nesting to organize elements within one another. Developers use HTML comments (`<!-- -->`) for notes or remarks. This hierarchical arrangement ensures rendering and functionality of web content. [8]

1.2.7 Head

The HTML `<head>` tag acts as a storage container, for information about a webpage, including metadata, titles, links to resources, like stylesheets and JavaScript files and other non visible elements. On the contrary the `<header>` element is used within the content of a website to provide contextual information that may appear at the top of a section or page. This can include headings, logos or navigation menus. Unlike the `<head>` tag, which is typically placed within the `<html>` element and only appears once in a document the `<header>` element can be used times within a webpages content to structure and present various headers or introductory content throughout the site. [8]

1.2.8 Body

The `<body>` tag is a HTML element that typically contains all the content, on a website. Within the `<body>` components like headings (`<h1>`, `<h2>`, etc.) paragraphs (`<p>`) images (``) links (`<a>`) lists (``, ``) and more are represented by HTML tags. These tags provide instructions to the web browser on how to display and render each component. They allow the browser to format and present the content to users based on the HTML structure, styling and any additional CSS (Cascading Style Sheets) rules that may be applied. By organizing content, into HTML elements it ensures that the webpages content is well structured and visually presented. [8]

1.2.9 Advantages of HTML

HTML or Hypertext Markup Language plays a role, in website development due to compelling reasons:

- **Easy to Learn;** HTML is known for its simplicity making it accessible even for beginners. Its straightforward syntax is easy to understand. The fact that its not case further simplifies the learning process.
- **User Friendly;** HTML stands out as one of the user programming languages for creating websites. Websites built with HTML tend to load and have a organized structure, which not only improves user experience but also contributes significantly to search engine optimization (SEO).
- **Cost effective;** One of the standout features of HTML is that it's completely free to use and doesn't require any plugins or paid software. This cost effectiveness makes it an attractive choice for businesses looking to create websites without incurring expenses.
- **Browser Compatibility;** HTML enjoys widespread support across various web browsers, including popular ones like Google Chrome, Safari and Opera. This broad compatibility simplifies the task of optimizing web pages based on HTML to ensure accurate display, across browsers.
- **Lightweight and Fast**HTML code is known for its nature making it easy to download and resulting in speedy webpage loading times. This feature holds value, for businesses aiming to deliver quick loading pages that save bandwidth and ensure an user experience.

1.3 CSS

CSS, also known as Cascading Style Sheets is a powerful design language that enhances the visual presentation of web pages. It provides designers and developers, with the ability to

customize text attributes such as color, font style and spacing. With CSS you can fine tune layouts incorporate background images or colors create designs that work well on devices, control element positioning, add transitions and animations. This level of customization is crucial for aligning with branding and user experience objectives. Moreover CSS promotes browser compatibility and code modularity while allowing for easy reusability. It is a tool, for improving the appeal and functionality of websites.[10]

1.3.1 CSS Selectors

CSS selectors are tools, in web development that allow us to apply styles to elements on our web pages. With a wide range of selectors available we can precisely target the elements we want to style and ensure they receive the desired treatment. Selectors consist of a combination of words and instructions that tell the web browser which HTML elements to identify and how to assign CSS attributes to them. This attention to detail gives designers control over the appearance and layout of their web content resulting in high quality designs and an improved user experience, on websites. 11]

1.3.2 Selector list

A CSS selector list is a tool that allows you to apply the style rule to multiple elements or groups of elements, with just one declaration. By separating selectors with commas you can efficiently consolidate styles for HTML elements making your CSS code more concise and easier to manage. This approach is especially useful when you want to maintain styling across elements that have similar characteristics as it reduces the need for repetitive code and simplifies the process of updating and maintaining styles, on your website.11]

1.3.3 Types of Selectors

In CSS, various types of selectors are available to target distinct elements or sets of elements on a web page

- CSS Selectors for Classes: These selectors target HTML elements with a specific class attribute. They are denoted by a dot (.) followed by the class name (e.g., .my-class).
- CSS Selectors for IDs: ID selectors target a single HTML element with a unique identifier. They are indicated by a hash (#) followed by the ID name (e.g., #my-id).
- CSS Selectors for Elements: Element selectors target HTML elements directly. They use the element's name (e.g., p for paragraphs, h1 for headings) and apply styles to all instances of that element type on the page.

- Attribute selectors in CSS offer the capability to target elements based on their attributes or attribute values. For example, all links with a specific href attribute can be selected using a selector such as `a[href="https://example.com"]``.
- CSS Selector for Pseudo-class: Pseudo-classes target elements based on their state or position. Common examples include `:hover` (for mouse hover), `:focus` (when an element is in focus), and `:nth-child` (for selecting specific children of an element).
- CSS Selectors for Global: Global selectors apply styles to all elements on a webpage. The asterisk (*) is used as a wildcard to target everything (e.g., `* { property: value; }`). [12]

1.4 Grid vs Flexbox

Grid and Flexbox are strong CSS techniques that enable distinctive customization in current web design. Flexbox lets developers design a grid layout flexibility, whereas Grid lets them create complicated, responsive solutions that are straight-forward to manage. These two tools offer unparalleled control over website design and enable the rapid creation of beautiful designs. Developers love them because they make layouts look good and work well, Flexbox and Grid create beautiful, responsive user experiences. Both techniques have advantages and operate differently. Grid creates column-based layouts rapidly, whereas Flexbox lets elements be moved across the page.

The CSS Grid module offers features for layout in a two-dimensional space, making it possible for items to be positioned both vertically and horizontally. On the other-er side, CSS Flexbox offers versatility by enabling components to be arranged along a single axis in either a horizontal or a vertical direction. This can be done both in the vertical and horizontal manners.

CSS Grid & CSS Flexbox are popular approaches to align web page elements. CSS Grid is preferable for two-dimensional layouts and CSS Flexbox for one-dimensional designs. Flexbox employs margins, but CSS Grid uses numerical co-ordinates, this simplifies Flexbox element size adjustments.

Flexbox and Grid equally benefit item management. Flexbox makes manipulating element size, position, and page order easy. Yet, a grid enables more exact two-dimensional element placement and structured layouts utilizing numerical co-ordinates. CSS Grid or CSS Flexbox is deter-mined by the task and desired effect. [13]

1.5 CSS Box Model

The CSS box model is a foundational concept in web design that encompasses several essential components, including the content area, padding, border, and margin, which together surround and define the layout and design of elements in CSS. This model plays a crucial role in creating and customizing the arrangement of various web page components. According to the CSS box model, each element is visually represented as a rectangular prism, with the content area at its core, surrounded by padding, a border, and an optional margin. This model allows designers and developers to control spacing, sizing, and positioning of elements, contributing to the overall structure and aesthetics of web pages. [14]

1.5.1 Block and Inline Boxes

In CSS, there are two primary categories of boxes: Block Boxes and Inline Boxes. The type of box, whether it's a block or inline box, determines its behavior in terms of page flow and its interaction with other boxes on a webpage. Block boxes typically create a new block formatting context, stacking vertically on the page and forcing subsequent content to appear below them. They are commonly used for structural elements like divs, paragraphs, headings, and lists. Inline boxes, on the other hand, do not break the flow and are used for elements that appear within a line of text, such as links or spans. Both the interior and exterior of these boxes can be utilized to display content, giving web developers flexibility in presenting information on webpages. [14]

1.5.2 Margin

In CSS, every box has an invisible gap surrounding it known as the margin, which influences the spacing between the box and other elements on the webpage. Margins can be assigned both positive and negative values, allowing for precise control over the positioning of the box in relation to neighboring elements. When a box is set to have a negative margin along one side, it may overlap with other elements on the page, potentially altering the layout. Importantly, It's worth noting that after determining the dimensions of the displayed box, the margin is consistently included in the box's overall dimensions, regardless of whether the conventional or alternative box model is being used. This consistent inclusion ensures that the margin's impact on the box's placement and spacing is always considered during the rendering of web content. [14]

1.5.3 Padding

Padding is an essential component of the CSS box model, situated between the border and the content area. It serves to create space between the content and the border, essentially pushing the content area further away from the border. Unlike margins, padding cannot have a negative value; it always adds space rather than reducing it. An important aspect to note is that any background applied to an element will be visible beneath the padding, meaning that the background extends through the content area and padding, contributing to the overall visual appearance and design of the element. [14]

1.5.4 Borders

In the CSS box model, the border is positioned between the margin and the padding of the box. When the default box model is employed, the dimensions of the box, encompassing the content, padding, and border, are all taken into consideration in the calculations of the element's width and height. As a consequence, the dimensions of the content box are expanded by an amount equivalent to the measurement of the border. It should be noted that a portion of the available height and width within the element box is absorbed by the border.

In contrast, when the alternative box model is opted for, where borders are included within the box dimensions, the size of the borders directly influences the dimensions of the content box. This can result in a smaller content box, as the border utilizes some of the available space. A clear understanding of and choice between these box models is crucial for achieving precise control over the layout and sizing of elements in CSS. [14]

1.6 Responsive Web Design

Responsive web design (RWD) stands as a design methodology with a primary focus on ensuring that a website adeptly adjusts and responds to user interactions and their device's characteristics, encompassing factors like screen resolution, platform, and orientation. RWD relies on a spectrum of adaptable design techniques, including flexible grids, responsive graphics, and strategic deployment of CSS media queries. The central objective of RWD revolves around the creation of web pages that consistently offer a user-friendly and fully functional experience, regardless of the screen size or resolution employed by website visitors. In essence, RWD represents an approach to web design that places paramount importance on delivering optimal user experiences across diverse devices and screen configurations. [15]

1.7 JavaScript

JavaScript, a versatile programming language, plays a pivotal role in enhancing interactivity and dynamic functionality within applications and webpages. What sets JavaScript apart is its ability to execute directly in web browsers, rather than relying solely on server-side processing. Alongside Hypertext Markup Language (HTML) and Cascading Style Sheets (CSS), JavaScript ranks among the most extensively utilized coding languages on the World Wide Web. It serves as an integral component of the front-end development stack for the majority of websites and online applications, enabling seamless integration of interactivity and user experience enhancements through the harmonious interaction of JavaScript, CSS, and HTML. [16]

1.7.1 Libraries vs Frameworks in JavaScript

JavaScript libraries are invaluable collections of pre-written code modules that developers can access and utilize to efficiently perform commonly recurring tasks. By allowing programmers to reuse code that has been previously created, libraries significantly reduce the time and effort required in software development. In essence, libraries serve as handy tools that streamline coding processes. On the other hand, frameworks encompass a broader spectrum of functionality compared to libraries. They offer a comprehensive toolkit that equips developers with the essentials needed to kickstart a project. In contrast, a library may specialize in specific code snippets for particular website features. Notable examples of JavaScript libraries include jQuery and React, while well-known JavaScript frameworks encompass Vue.js, Node.js, and Angular, all of which are widely adopted for diverse web development tasks. [17]

1.8 Server-side Programming

Server-side programming is a critical aspect of web development that deals with handling client requests originating from web browsers and delivering appropriate responses. This approach empowers developers to create fast, scalable web applications, seamlessly integrate with databases, construct APIs (Application Programming Interfaces), and facilitate real-time communication between clients and servers. A prominent example illustrating the significance of server-side programming is Facebook's newsfeed feature. In a platform as vast as Facebook, creating and serving millions of static pages for every story, post, and

status update would be highly inefficient. Instead, Facebook employs server-side programming to dynamically update both CSS (Cascading Style Sheets) and HTML templates by leveraging JavaScript. This dynamic approach ensures that users receive real-time updates and a personalized experience while conserving server resources and optimizing performance. By utilizing server-side programming, Facebook efficiently manages the immense flow of content and interactions within its platform. [16]

1.9 JavaScript Impact on Site Performance

Incorporating JavaScript into a website's components can have an impact on its performance. However, the benefits of using JavaScript judiciously and sparingly can outweigh the potential drawbacks to performance. Typically, a page's performance tends to decrease in relation to the amount of code and scripts it contains. Each JavaScript script encountered by a user's browser must be downloaded, parsed, and executed in the order specified in the HTML. Additionally, elements like image carousels and embedded videos can contribute to longer page loading times. Complex themes and plugins can also have a notable impact on the website's performance.

The primary thread, responsible for parsing and executing JavaScript, functions as a queue of tasks that must be completed sequentially. When there's an excessive number of JavaScript tasks running on this main thread, it can potentially slow down the loading of other content or images. If essential components, such as the website's content, take an unacceptable amount of time to load, it can negatively affect the user experience, leading users to leave the page. This, in turn, can have consequences for search engine optimization efforts, website traffic, and ultimately the business's success. Additionally, it's crucial to consider that not all users have access to high-performance devices and fast internet connections, leading to varying webpage loading times among users.[18]

1.10 JavaScript Debugging Techniques

The internet provides access to a wide range of debugging tools that can be chosen to align with the selected development methodology. Major libraries and frameworks often offer publicly available debugging practices and tools tailored to their ecosystems. These resources play a crucial role in diagnosing and resolving issues, ensuring the seamless operation of software applications and websites. Developers can utilize these resources to simplify the debugging process and improve the quality and reliability of code.

1.10.1 Bugs

Bugs in software programs are essentially flaws that lead to unintended behaviors or results. These issues often arise when a specific part of the system fails to meet its intended requirements or expectations. Software programs operate based on predefined rules and conditions, and if these conditions are not satisfied, the program may produce unexpected outcomes. Additionally, bugs can occur due to conceptual or syntactic errors in the source code itself. For example, in the JavaScript programming language, an open curly bracket must be followed by a closing curly bracket to properly indicate the end of a code block. Failure to include the closing bracket can result in a syntax error and potentially yield different results than intended. This scenario is exemplified when a missing closing curly bracket leads to an application displaying an error code in the terminal. Debugging is the process of identifying and resolving such issues to ensure the software behaves as expected. [19]

1.10.2 Debugging

Debugging is the process of identifying and resolving issues in a computer program, enhancing its functionality, and ensuring future reliability. Even if a program is well-structured in terms of syntax and logic, unexpected behaviors can still occur. These issues may arise due to incorrect dependencies, setup, or settings, rather than errors within the code itself. To address and rectify such problems, debugging skills are crucial. For instance, consider a scenario where a well-written code fails to produce the expected outcome because the JavaScript file isn't correctly linked to the file required by the browser to display the desired result. In such cases, debugging plays a pivotal role in troubleshooting and resolving the issue, ultimately improving the program's performance and reliability..[19]

1.11 JavaScript Functions

In JavaScript, functions hold a fundamental role as they encapsulate action logic. A function can be thought of as a procedure that executes to perform a task or produce an outcome. However, for it to qualify as a function, it must take input and provide a result. For example, a function designed to calculate a value should accept input and return the calculated result. Essentially, a function comprises a set of statements that carry out a specific operation. By repeatedly invoking the same function in a program, code duplication can be avoided, a practice referred to as modular coding. These functions are purpose-built for specific tasks,

allowing for the division of larger programs into more manageable functional units. Functions can either be included as part of the system or created by users. [20]

1.12 Advantages of JavaScript

JavaScript, being primarily executed within the client's browser, generally boasts impressive speed. It operates efficiently without significant slowdowns caused by interactions with a back-end server, unless external resources are required. Notably, JavaScript doesn't necessitate compilation prior to execution; it seamlessly works across major browsers and is compiled just in time. Its syntax, originally inspired by Java, remains straightforward and easily accessible for learners. JavaScript's growing popularity as a backend language is attributed to its frequent usage and the rising prominence of Node.js. There is a wealth of resources available for learning JavaScript, with an increasing number of JavaScript projects on platforms like Stack Overflow and GitHub. This trend is expected to persist. Unlike some other programming languages, JavaScript can be employed on virtually any website. Its versatility extends to integration with languages such as PHP and Perl. As a client-side language, JavaScript lightens the load on servers, and in certain cases, a server may not even be required for running simple applications. JavaScript's capabilities, including the addition of components like sliders and functionality like drag-and-drop, prove valuable for websites featuring intricate interfaces. Developers enhance website utility by incorporating JavaScript snippets that allow for flexible plug-ins. Node.js servers offer various ways to run JavaScript, enabling the creation of complete applications using Express for Node.js setup, a document database like MongoDB, and JavaScript for front-end user interactions. [21]

1.13 Bootstrap

Bootstrap acts as a front-end programming framework that may be used for free and is open source. It can be used to create web pages and web apps. Bootstrap is a set of syntax that is used for the design of templates. Its primary purpose is to facilitate the responsive building of mobile-first websites, furthermore Bootstrap is a framework that offers the fundamentals for developing responsive websites; therefore, all that is required of developers is to put the code into a pre-defined grid structure. HTML, CSS, and JavaScript are the three primary building blocks of the Bootstrap system. While building websites, web developers that use Bootstrap can do so much more quickly because they don't have to spend as much time thinking about fundamental commands and functions. [22]

1.13.1 Why Use Bootstrap

Bootstrap simplifies the process of website, web app, or mobile app development, especially when faced with tight deadlines. It provides pre-made blocks that can be easily customized, eliminating the need to start from scratch, and offers pre-designed themes and templates. While many users rely on these resources, enhancing a website or app's uniqueness in a competitive field can be achieved by adding creative touches.

Bootstrap's foundation lies in its grid layout, which seamlessly adapts to various screen sizes, particularly important for mobile devices. A clutter-free front-end design is crucial for maintaining visitor trust. Bootstrap's "Mobile-First" grid approach divides screens into 12 columns, ensuring adaptability across different screen sizes. It supports mobile-friendly front-end development, allowing tailored element visibility based on the device.

Bootstrap's CSS exhibits flexibility, rendering it valuable in situations with time constraints or limitations in custom CSS. Bootstrap's themes are open to customization, allowing for the refinement of web projects by eliminating unnecessary plugins and components. Variable values provide added options for template customization, and Bootstrap maintains compatibility with contemporary browsers and systems, while ceasing support for outdated ones.

As an open-source platform, Bootstrap provides customization freedom and continues to evolve with contributions from the developer community. The official Bootstrap website, along with other resources, offers extensive front-end development support, providing clear instructions and editable layouts and themes.

1.13.2 Advantages of Bootstrap

Responsive web design (RWD) offers a multitude of advantages that streamline the web development process and enhance user experiences across various devices and projects. These advantages include:

- **Reusability:** RWD principles allow for the avoidance of redundant code and design elements across multiple projects, promoting code efficiency and consistency.
- **Browser Compatibility:** RWD techniques are compatible with various web browsers, ensuring that websites function correctly and look appealing across different browser platforms.
- **Adaptive Design:** RWD enables designs that fluidly adapt to diverse screen sizes, empowering users to access content seamlessly and control what is displayed or hidden on their devices.

- **Consistency:** When working with multiple development teams on different projects, RWD ensures design coherence and consistency, aligning branding and user experiences effectively.

Prototype Development: RWD facilitates the rapid creation of working prototypes, accelerating the development cycle and allowing for quick testing and refinement of website designs and functionalities. [24]

1.13.3 Bootstrap Container

Bootstrap's foundational component for structuring content is known as a "container." In Bootstrap, web pages' content is enclosed within containers, which serve as essential structural elements crucial to the framework's layout system. These containers are responsible for organizing and aligning the content they contain, adjusting it in accordance with the viewport size or the device being used. The "container" class is the key to defining these containers in Bootstrap. Essentially, the container class determines the width of the layout to accommodate the content effectively. Within these containers, various elements and content are then inserted, allowing for a well-structured and responsive web page layout. [25]

1.13.4 Bootstrap Grid System

A grid is a foundational framework utilized in graphic design, usually on a two-dimensional plane, composed of intersecting straight lines, both vertically and horizontally. Grids are employed to streamline the organized presentation of data, finding extensive use in print design for establishing structured layouts and content arrangements. In web design, the integration of HTML and CSS offers an efficient and time-saving approach for achieving consistent layouts. Bootstrap's Grid System, for instance, provides up to 12 columns that span across a web page, offering flexibility to utilize them individually or combine them into a single, larger column. This grid-based approach enhances web design's structural integrity and responsiveness. [26]

2 SECURITY OPTIONS FOR WEB APPLICATION

Over three-quarters of the whole cybercrime, by most accounts, is directed at applications and their weaknesses. Web application firewalls (WAFs), multi-factor authentication (MFA) for users, using, protecting, and validating cookies to maintain user state and privacy status, and various methods of validating user input to ensure it is not malicious prior to processing are all examples of how web application security products and policies work to protect applications. [27]

The process of locating, testing, possibly setting up, and setting up software patches (also known as updates) on computers is referred to as patch management, a software patch is typically a component of code that is tailored to address any faults or vulnerabilities that are already present in the operating system, add new functionality, or increase the product's security, in general, the process of software patch management consists of checking devices in the system for patches that are missing, testing the patches on a test set of devices, and delivering the fixes either manually or automatically using patch management software. It is essential to do a check and produce reports once software patches are released and installed to ensure an excellent level of patch compliance throughout the network. [29]

A web application firewall, often known as a WAF, filters and monitors the HTTP traffic that moves among a website's app and the Internet to assist in the protection of online applications. Most of the time, it defends online applications against threats like cross-site forged documents, cross-site scripting (also known as XSS), file diversity, and SQL injections, amongst others. A WAF is a defense that operates at protocol layer 7 (in the framework of OSI), and it is not intended to protect against all different kinds of attacks. This technique of combating attacks is typically implemented as part of a larger set of tools that, when used in conjunction with one another, produces an all-encompassing defense against a variety of attack vectors. [30]

On a website or in a web application, the process of handling several messages and inquiries from a single user or entity is referred to as "session management." During these interactions, information about the user will be transmitted back and forth between the web server and the web browser, as well as stored in the browser and even processed by it. For this reason, it is necessary for any web app to effectively manage sessions and provide adequate security. [31]

The concept of the least privilege, often known as POLP, is a principle in security for computers that restricts the access permissions of users to only those things that are necessary for them to perform their duties. Users are only granted authorization to read, write, or execute those files and resources that are essential to the completion of their tasks. [32]

In the field of security, the two terms of authorization and authentication are now the most common terms employed. They may share a similar sound, but they are not even quite comparable to one another. The purpose of authentication is to verify the identity of a person, while the objective of authorization is to grant authority to a user to access a certain resource. Authentication is contrasted with authorization. [33]

When doing validation checks, the data entered by the user is compared to a predetermined list of criteria to ensure that the user is, in fact, entering the correct information. If the validation process is unsuccessful, the submission must be rejected. This is critical not only from the privacy aspect, but also from the perspective of maintaining the consistency and integrity of the data, given that data is typically used across a range of different systems and applications. [34]

The practice of deleting or replacing data that has been provided is referred to as sanitization. In the process of working with data, sanitization is typically a further process that is completed after the appropriate checks for validation are made. This is done to increase the data's safety. [34]

Cross-site scripting, often known as XSS, is a weakness in web security that enables attackers to break into any interactions individuals have with an application that is insecure. This vulnerability is also known as cross-site scripting. It gives an attacker the ability to get around the identical origin policy, that is meant to keep distinct websites apart from one another. Cross-site scripting vulnerabilities allow an attacker to normally pose as a target user, carry out any operations that the user's browser can perform, and gain access to any of the user's data. This is because the vulnerability allows an attacker to impersonate the user. If the victim user of the application has exclusive accessibility within the application, then the attacker may be able to obtain complete control of all the functionality and data contained within the application. [35]

Cross-site request forgery, commonly abbreviated as CSRF, is a web security flaw that enables an attacker to trick users into carrying out actions that they did not want to carry out. Another name for this flaw is clickjacking. It gives an attacker the ability to partially get

around the identical origin policy, that is intended to stop multiple websites from interacting with one another in any way they can. [36]

Server-Side Validation is carried out on the server, which is also where the application is stored and retrieved from. This approval is carried out to protect the application from the customer, who may try to circumvent the approval that is carried out from the customer's perspective and cause harm to the application. Because of this, customers typically attempt to proceed despite the client's side permission, which is carried out using JavaScript. [37]

System logs, specifically Logging and Monitoring, serve the purpose of providing essential information that aids in understanding and identifying issues before they escalate into network threats. Typically, software is responsible for collecting data that reveals activities occurring within the network. This data serves as a valuable source of insights. However, if the pertinent information is buried amidst a deluge of other data, it renders these collections of data meaningless. A reliable security logging and monitoring system can enable an organization to achieve a variety of crucial cybersecurity objectives. [38]

2.1 Importance of Web Application Security

The root cause of security flaws in online applications lies in the vulnerability of sensitive information, such as user credentials or financial transactions, making it a prime target for hackers. Ensuring the security of web applications is crucial for safeguarding data during both transmission and storage on servers. Even a seemingly minor data breach can have significant repercussions for a company's reputation and financial standing.

Today's customers are more tech-savvy than ever, emphasizing the importance of strengthening web application security to prevent breaches that could impact a brand's integrity. Even novice users possess a basic understanding of encryption and other essential web app security measures. A trustworthy web application is one that users confidently recommend to others. Enhancing web app security extends beyond code cleanup; it also enhances a public image, as positive word-of-mouth resulting from robust security practices holds great value.

Hackers exploiting application vulnerabilities can severely impact revenue, making online application security the most critical factor in enhancing financial performance. Direct financial losses can occur when attackers gain access to financial data. Additionally, there are

indirect financial consequences of operating an insecure online service, including users seeking alternatives if an app appears insecure. As new web app security threats emerge, the importance of safeguarding web app data will continue to rise, potentially leading to legal actions from customers and users following a significant security incident involving mishandled data. [28]

2.2 Web Application Firewalls (WAF)

Web application security encompasses various approaches designed to mitigate specific risks and safeguard web applications from a range of threats. One such approach involves the use of Web Application Firewalls (WAFs), which function as protective barriers. WAFs continuously monitor and filter user traffic, providing defense against numerous types of attacks. By configuring WAFs with policies to distinguish between safe and unsafe traffic, they can effectively thwart suspicious communication attempts, ensuring unauthorized access is denied and unauthorized data leakage is prevented. In addition to WAFs, web application security measures include user authentication, which verifies the identity of users, application vulnerability scanning to identify and rectify potential weaknesses, cookies management for safeguarding user data, data visibility to monitor and protect sensitive information, and IP denylists to block access from known malicious IP addresses. Collectively, these security measures contribute to the comprehensive protection of web applications against a wide array of threats and vulnerabilities. [27]

II. ANALYSIS

3 DESIGN OF A BLOOD BANK SYSTEM

In this chapter, the design of a blood bank system is proposed. First, requirement model is presented. After that, use case diagram is described. A Class diagram was created based on the requirements and the use case diagram. At the end, a sequence diagram of one-use case is shown.

3.1 Requirement Models

Donors may be added to the database by authorized users. The various blood types will be monitored and controlled by the system. The blood bank manager is responsible for keeping track of donors, blood types, and blood bag supplies. User accounts may be created, edited, and deleted with the use of the system's features. Blood requests on behalf of hospitals and other healthcare facilities may be submitted by authorized users. Users will be able to generate a wide range of reports on blood bank operations with the use of this technology. These functional criteria highlight the key features and capabilities of the Blood Bank Management System. Donors may be added to the database by authorized users. The various blood types will be monitored and controlled by the system. The blood bank manager is responsible for keeping track of donors, blood types, and blood bag supplies. User accounts may be created, edited, and deleted with the use of the system's features. On behalf of hospitals and other healthcare organizations, authorized users will be able to submit blood requests via the system. Users will be able to generate a wide range of reports on blood bank operations with the use of this technology. These functional criteria highlight the key features and capabilities of the Blood Bank Management System. The system can process large amounts of data quickly and respond to human input. Important processes including enrolling donors, obtaining blood inventory, and processing blood requests should have acceptable response times. The system's graphical user interface is clean and straightforward. Clear and concise instructions or tooltips are provided to aid users in completing tasks. Users have access to a dependable system within the designated working hours. Unexpected system failures, such as network interruptions or power outages, may be mitigated using these measures as shown in Figure 1.

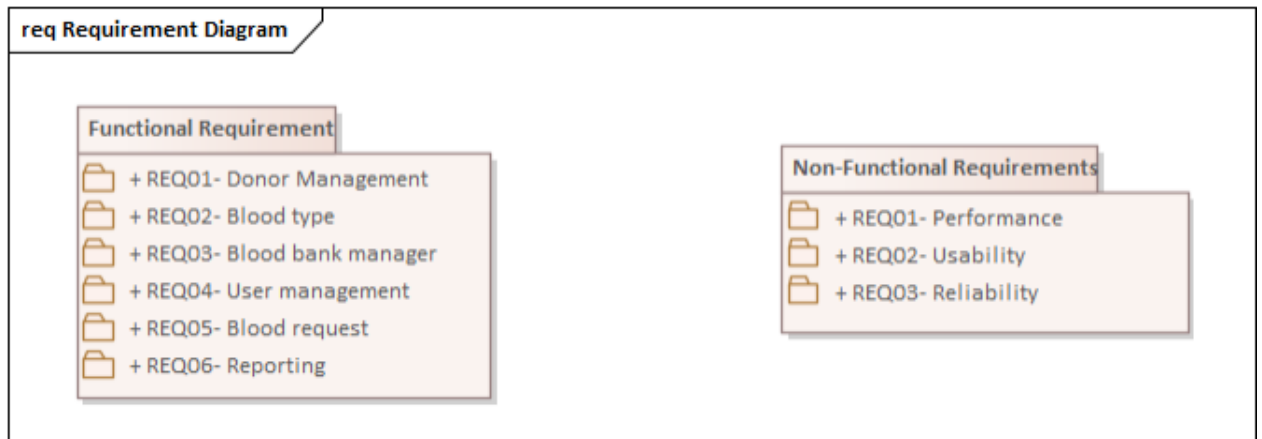


Figure 1 Requirement Model

The donor management aspect plays a pivotal role in ensuring the availability of a steady supply of blood units. The requirement model, specifically focusing on donor management, comprises a set of essential functionalities. These functionalities include "FREQ01- Donor Management," which forms the overarching category, and sub-functions such as "FREQ01_01 Add Donor," "FREQ01_02 Search Donor," "FREQ01_03 Update Donor," "FREQ01-04 View Donor," and "FREQ01-05 Delete Donor." Each of these components contributes significantly to the efficiency and effectiveness of the Blood Bank Management System, enabling the seamless addition, retrieval, modification, viewing, and removal of donor information. This requirement model plays a crucial role when designing the application, ensuring that the critical donor management processes are integrated seamlessly, ultimately contributing to the success of the blood bank's operations and its ability to save lives as shown in Figure 2.

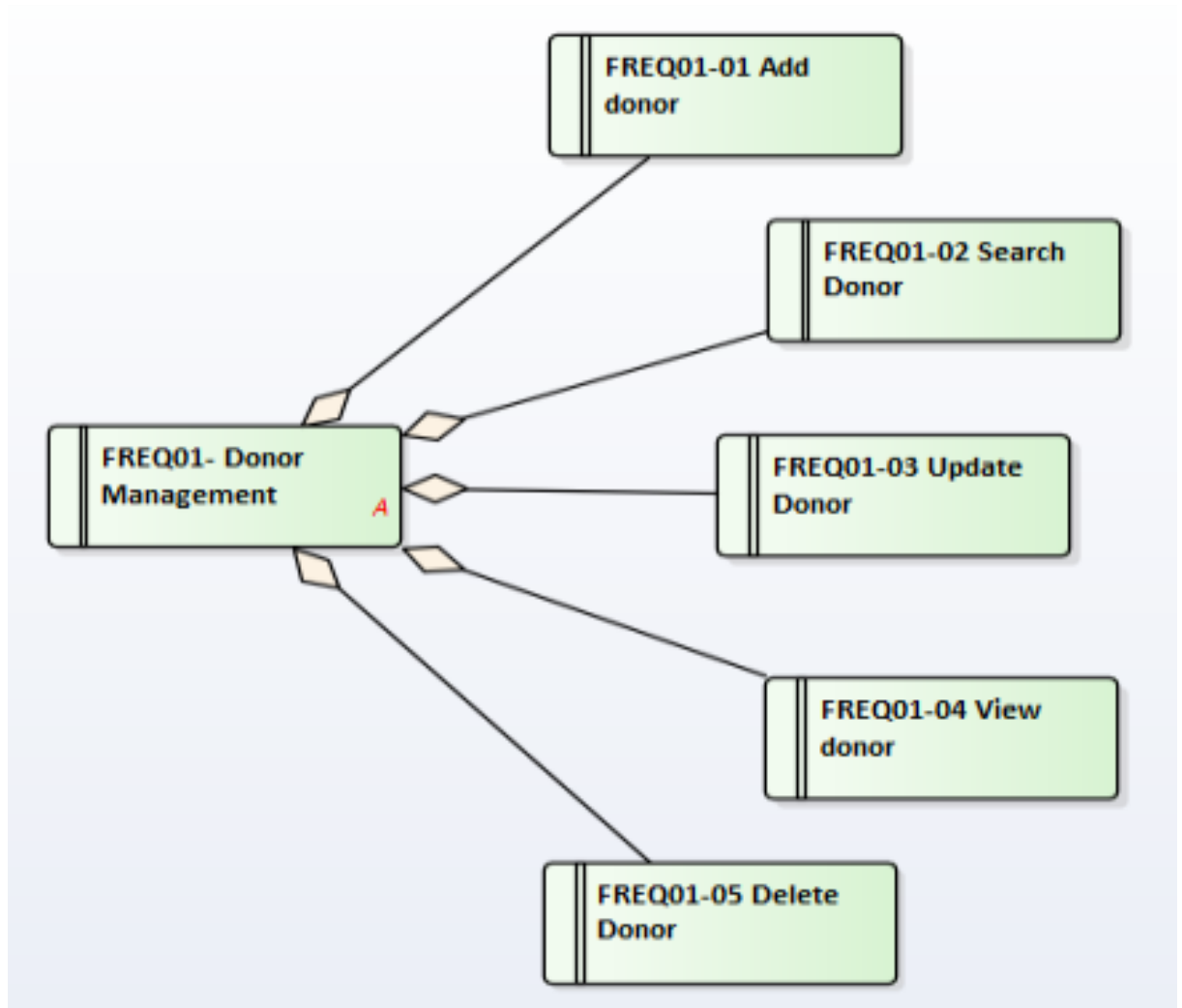


Figure 2 Donor Management Requirement Model

This model encompasses a set of functional requirements, starting with 'FREQ02 - Blood Type' as the foundational requirement, establishing the core need for blood type handling within the system. Subsequently, 'FREQ02_01 - Add Blood Type' allows for the incorporation of new blood types, 'FREQ02_02 - Search Blood Type' facilitates precise retrieval, 'FREQ02_03 - Update Blood Type' enables authorized modifications, 'FREQ02_04 - View Blood Type' provides transparency in displaying information, and 'FREQ02_05 - Delete Blood Type' ensures the removal of obsolete records. Collectively, these requirements form the backbone of the Blood Bank Management System, guaranteeing accurate blood type records, efficient data handling, and transparent operations, thereby enhancing the overall effectiveness and reliability of the blood bank's functioning as observed in Figure 3.

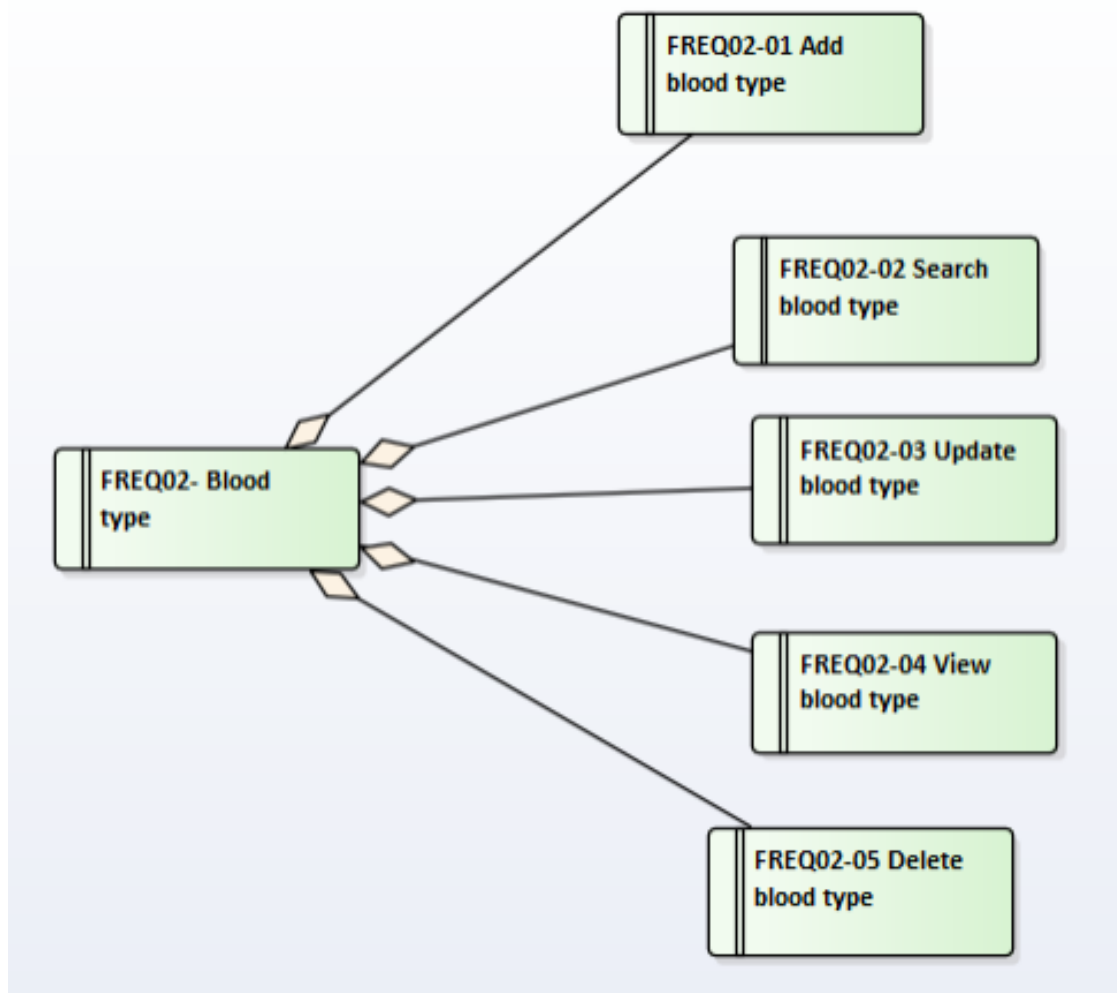


Figure 3 Blood Type Requirement Model

The focus lies on addressing the specific requirements of the Blood Bank Manager, denoted as 'FREQ03' in the requirement model. The Blood Bank Manager plays a pivotal role in ensuring the smooth operation of the blood bank, where every action taken can have critical implications for patients in need. The 'FREQ03' requirement model encompasses a set of essential functionalities that empower the Blood Bank Manager to efficiently manage blood bags within the system.

These functionalities include 'FREQ03_01: Add Blood Bag,' allowing the manager to input new blood donations into the database. 'FREQ03_02: View Blood Bag' enables the manager to access vital information about available blood bags swiftly. 'FREQ03_03: Update Blood Bag' permits necessary modifications to blood bag records when required. 'FREQ03_04: Search Blood Bag' equips the manager with a powerful search tool to quickly locate specific

blood bags among a potentially vast inventory. Lastly, 'FREQ03_05: Delete Blood Bag' provides the capability to remove outdated or unusable blood bags from the system.

These functionalities are not only central to the daily operations of a blood bank but also contribute significantly to maintaining the safety, integrity, and efficiency of the blood inventory. In this thesis, we will delve deeper into the design and implementation of these features within our Blood Bank Management Application, emphasizing their critical role in enabling the Blood Bank Manager to fulfill their responsibilities effectively and, ultimately, save lives through the efficient management of this invaluable resource as shown in Figure 4.

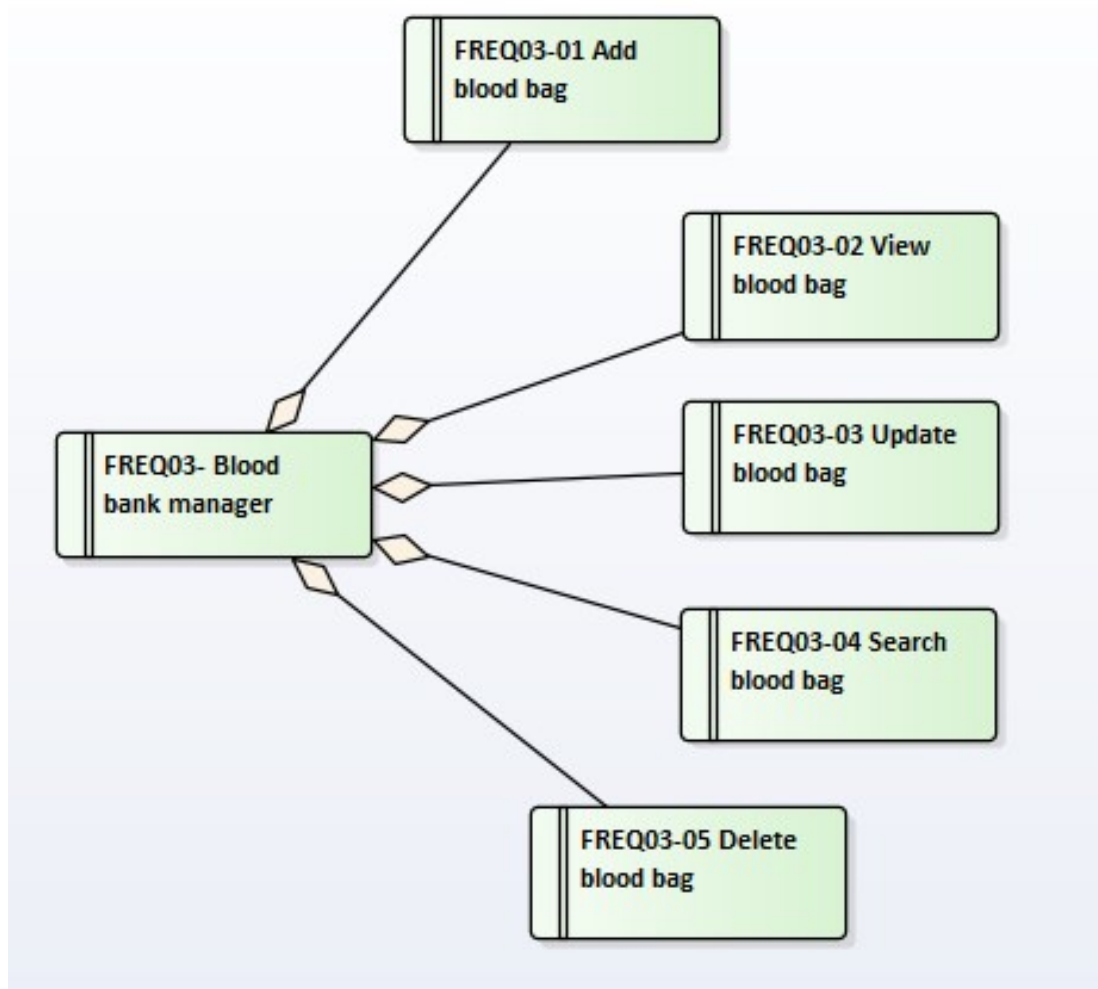


Figure 4 Blood Bank Manager Requirement Model

An essential component is the User Management Requirement Model, which plays a pivotal role in ensuring the security, accessibility, and efficient operation of the application. This model encompasses various user-centric functionalities, as outlined in the requirements.

Firstly, the 'FREQ04_01 registration' requirement highlights the need for a robust user registration process, which is vital for onboarding donors, recipients, and authorized personnel into the system. This ensures that the application maintains an accurate and up-to-date user database.

Secondly, the 'FREQ04_02 login' requirement emphasizes the significance of secure user authentication, allowing registered users to access the system securely while safeguarding sensitive medical and personal information.

The 'FREQ04_03 password reset' requirement addresses the need for a user-friendly mechanism to reset passwords, enhancing user convenience and security.

Lastly, the 'FREQ04_04 role-based access control' requirement introduces a crucial element of role management. This functionality ensures that users have access only to the features and data pertinent to their roles, maintaining data integrity and system security.

Collectively, the User Management Requirement Model lays the foundation for a Blood Bank Management System that is user-centric, secure, and tailored to the specific roles and responsibilities within a blood bank facility. This model's successful implementation will contribute significantly to the overall efficiency and effectiveness of the application as observed in Figure 5.

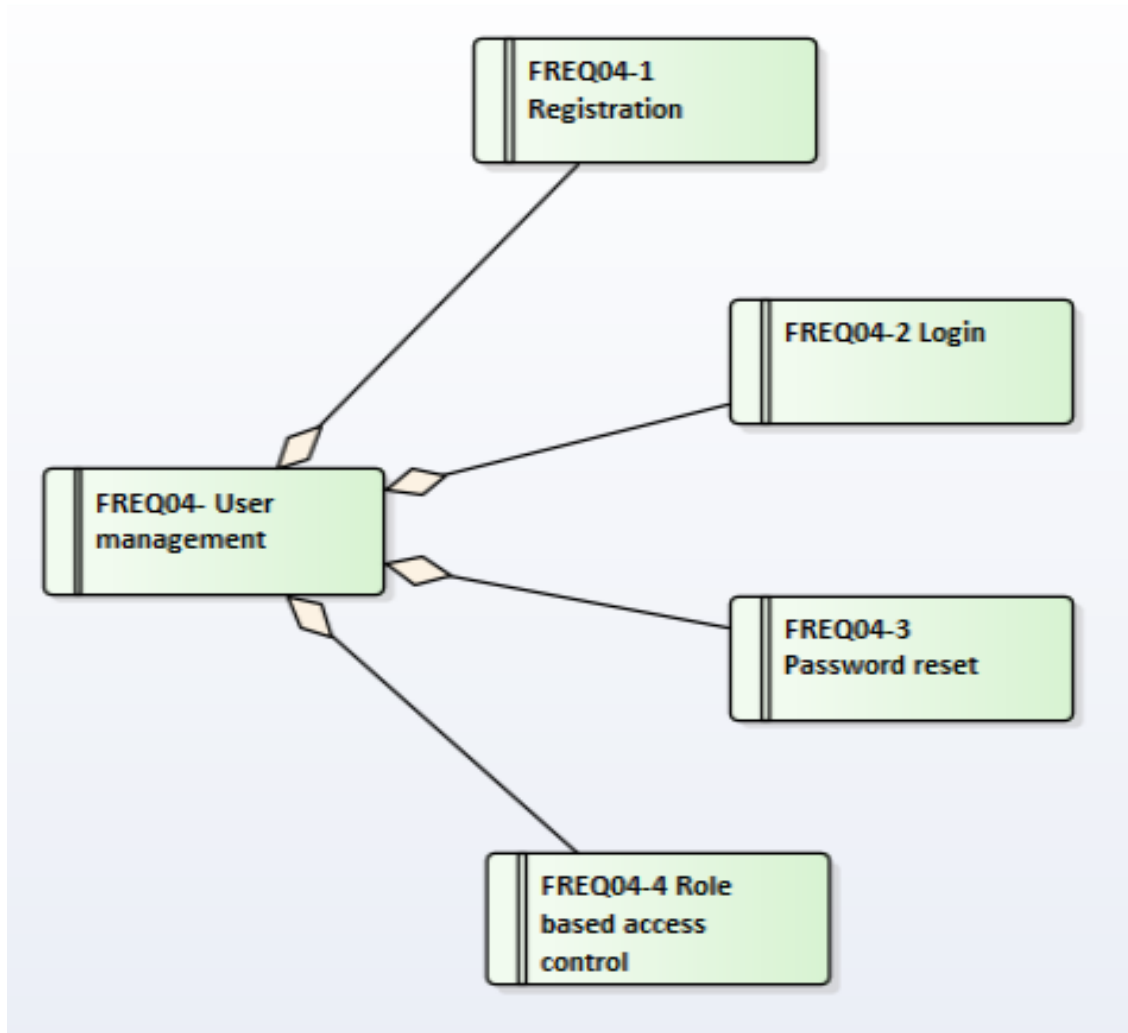


Figure 5 User Management Requirement Model

Another pivotal requirements identified is the 'FREQ05 - Blood Request' module. This requirement encompasses several critical sub-modules, each playing a vital role in the efficient functioning of the system.

Firstly, 'FREQ05_01 - Create Request' allows authorized users, such as medical professionals or hospital staff, to initiate a blood request seamlessly. This feature ensures a streamlined process for generating requests, reducing delays in accessing life-saving blood products.

Secondly, 'FREQ05_02 - Verify Recipient Eligibility' is a crucial component that verifies the eligibility of the recipient to receive the requested blood type. This verification process ensures that the right blood type is matched with the recipient, promoting patient safety and adherence to medical protocols.

Lastly, 'FREQ05_03 - Record Insurance' plays a pivotal role in maintaining a comprehensive patient database by recording insurance-related information. This data capture not only simplifies administrative processes but also facilitates potential reimbursement procedures for the healthcare institution.

The 'FREQ05 - Blood Request' requirement model, with its associated sub-modules, underscores the system's commitment to enhancing the overall efficiency, accuracy, and patient-centric approach of the Blood Bank Management System. It ensures that the process of requesting, verifying, and documenting blood requests is not only streamlined but also aligned with the highest standards of healthcare and patient safety.⁶ as illustrated in Figure 6.

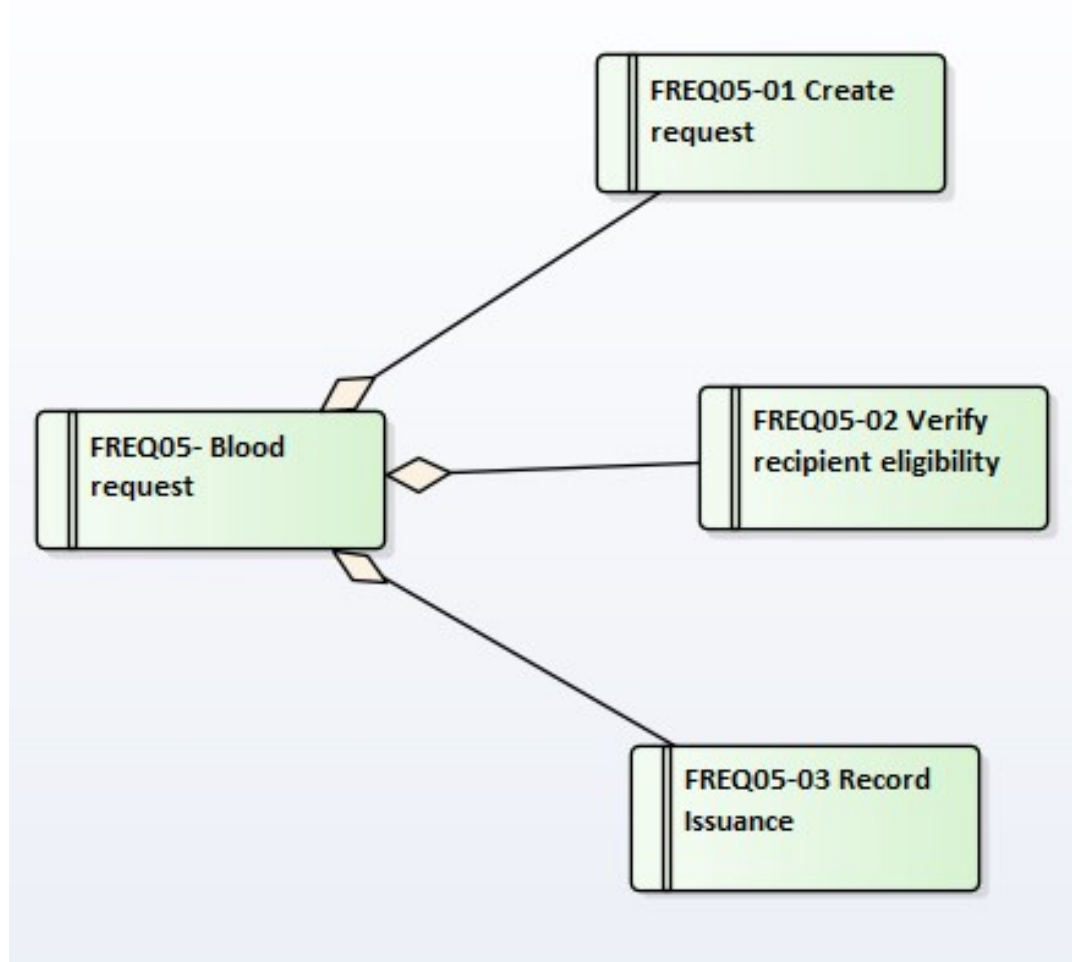


Figure 6 Blood Request Requirement Model

This model encompasses several essential reporting functionalities, identified as FREQ06. These functionalities include FREQ06_01 Donor Report, FREQ06_02 Blood Inventory Report, FREQ06_03 Blood Donation Report, and FREQ06_04 Blood Request Report. Each of

these reporting elements serves a crucial purpose within the system, enabling efficient tracking, management, and decision-making processes. The Donor Report ensures a transparent overview of donor information and their contributions, while the Blood Inventory Report provides real-time insights into the available blood supply. Additionally, the Blood Donation Report and Blood Request Report facilitate the monitoring of blood donations and requests, streamlining the coordination of life-saving resources. This reporting requirement model constitutes an integral part of the proposed Blood Bank Management System, enhancing its functionality and effectiveness in serving the needs of both donors and recipients as shown in Figure 7.

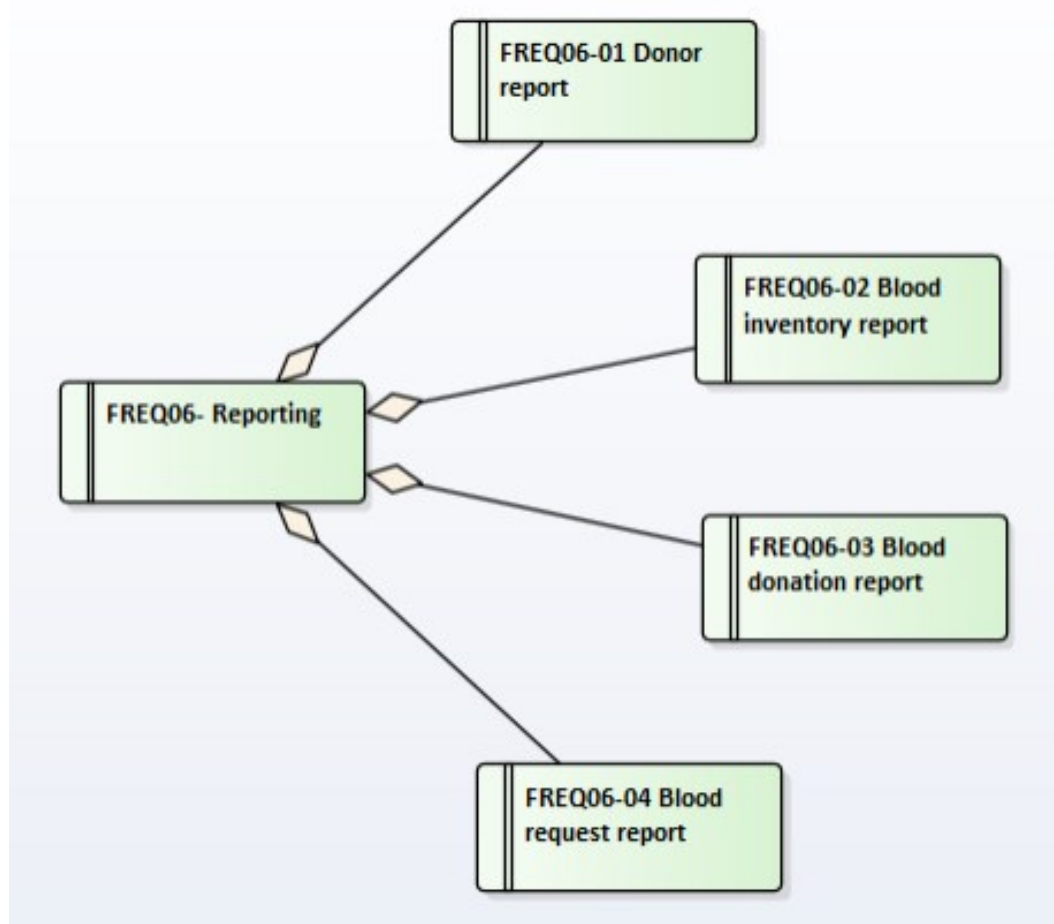


Figure 7 Reporting Requirement Model.

Addressing performance requirements is of paramount importance to ensure the efficient and reliable operation of the application. These performance requirements, outlined as FREQ01, encompass several key aspects. FREQ01_01, which focuses on response time, emphasizes the need for the system to respond swiftly to user requests, ensuring a seamless

and user-friendly experience. Scalability, denoted as `FREQ01_02`, is another critical consideration, highlighting the system's ability to handle increasing volumes of data and users without compromising performance.

Furthermore, `FREQ01_03`, related to availability, underscores the necessity for the application to be consistently accessible, particularly in critical situations where immediate access to blood supply information is vital. Finally, `FREQ01_04` emphasizes load balancing, demonstrating the importance of evenly distributing workloads across the system's components to prevent bottlenecks and maintain optimal performance.

These performance requirements collectively form the foundation upon which the Blood Bank Management System will be designed and developed, ensuring that it not only meets but exceeds the expectations of its users, ultimately contributing to more effective and efficient blood bank operations as illustrated in Figure 8.

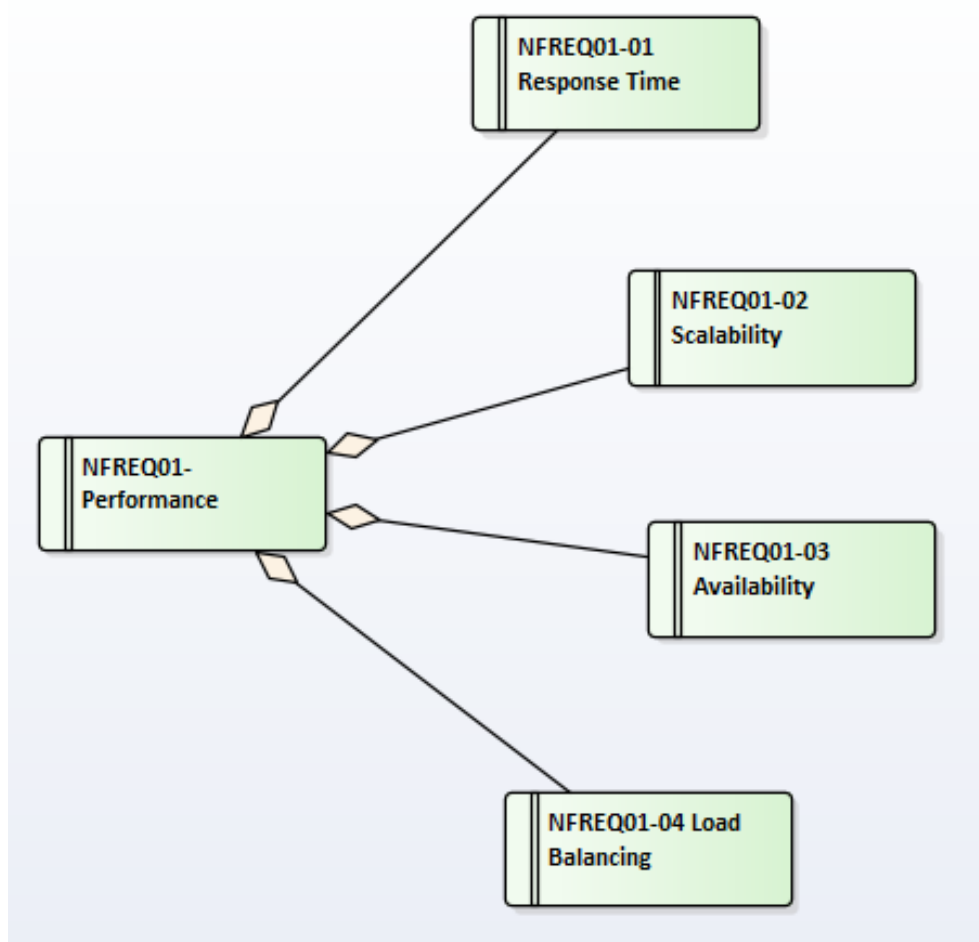


Figure 8 Performance Requirement Model

Usability requirements play a pivotal role in ensuring the effectiveness and accessibility of the application. These usability requirements are encapsulated under the overarching objective of "FREQ02 - Usability," encompassing three critical facets: "FREQ02_01 - Intuitive User Interface," "FREQ02_02 - Multilingual Support," and "FREQ02_03 - Accessibility Compliance." The first requirement, intuitive user interface, emphasizes the significance of crafting an interface that is user-friendly and easy to navigate, thus enhancing the overall user experience. Multilingual support, as specified in the second requirement, recognizes the diverse linguistic needs of potential users, ensuring inclusivity and adaptability for a wider audience. Lastly, the third requirement, accessibility compliance, underscores the importance of adhering to accessibility standards, making the application usable by individuals with disabilities. These usability requirements collectively contribute to the development of a Blood Bank Management System that not only efficiently manages vital resources but also embraces user-centric design principles, fostering its utility, and accessibility for a diverse user base as illustrated in Figure 9.

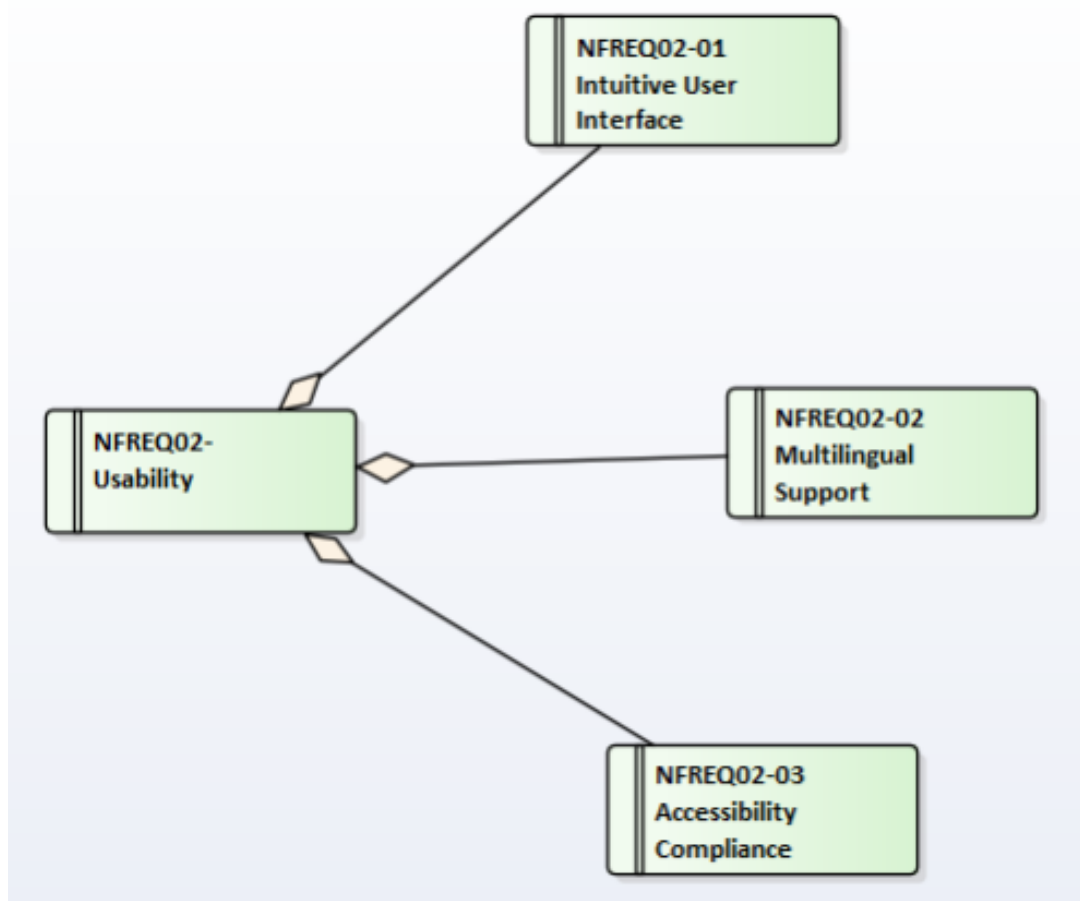


Figure 9 Usability Requirement Model

The reliability requirement model, encapsulated in the FREQ03 category, delves into crucial facets that ensure the system's dependability and uninterrupted functionality. FREQ03_01 emphasizes high availability, ensuring that the application remains accessible and operational at all times, especially during critical moments when blood donations and requests are paramount. FREQ03_02 addresses the vital aspect of backup and recovery mechanisms, ensuring that in the event of unforeseen data loss or system failures, the system can swiftly restore its functionality, safeguarding critical blood bank data. FREQ03_03 focuses on error handling and logging, which play a pivotal role in tracking and resolving issues as they arise, ensuring seamless and efficient system performance. These reliability requirements collectively underscore the commitment to developing a Blood Bank Management System that can be trusted to operate flawlessly, even in challenging scenarios, thereby facilitating the noble cause of blood donation and distribution Figure 10.

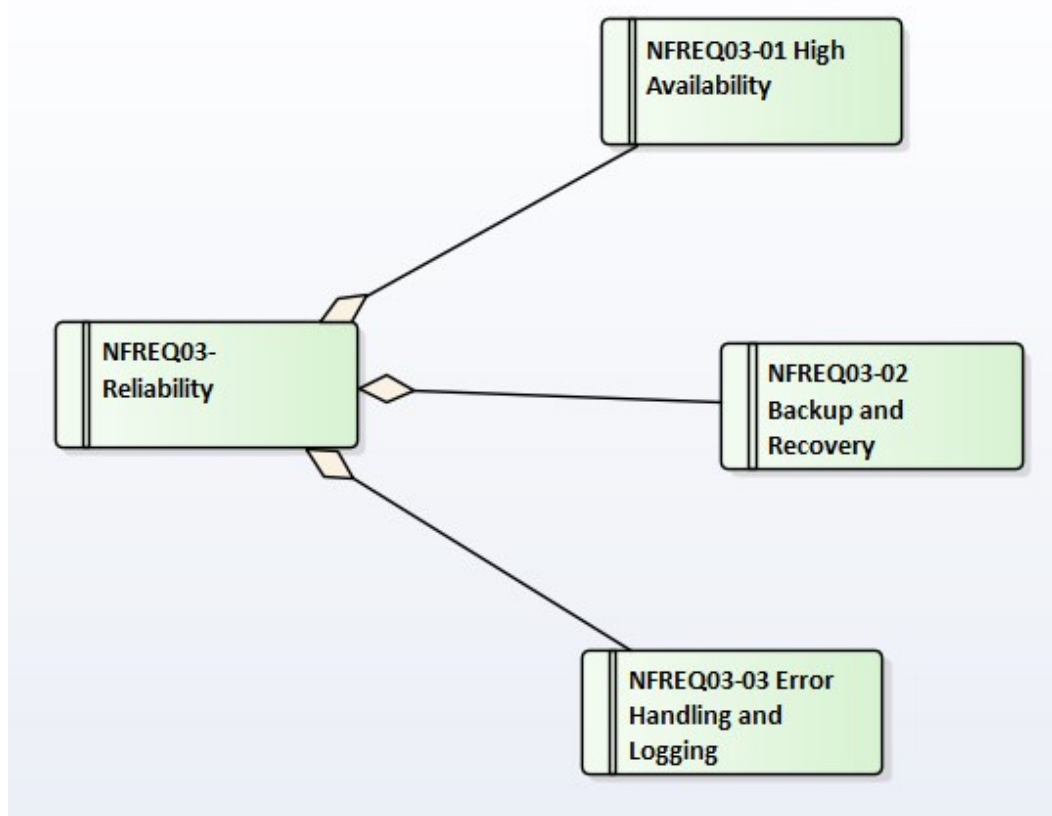


Figure 10 Reliability Requirement Model

3.2 Use Case Model

The use case model for a blood bank administration system depicts the various actors and their interactions and features as illustrated in Figure 11. Each actor in the system is authorized to do a limited set of actions to carry out their functions. Use cases offer a blueprint for further system analysis, design, and implementation by detailing the system's functionality from the perspectives of different actors.

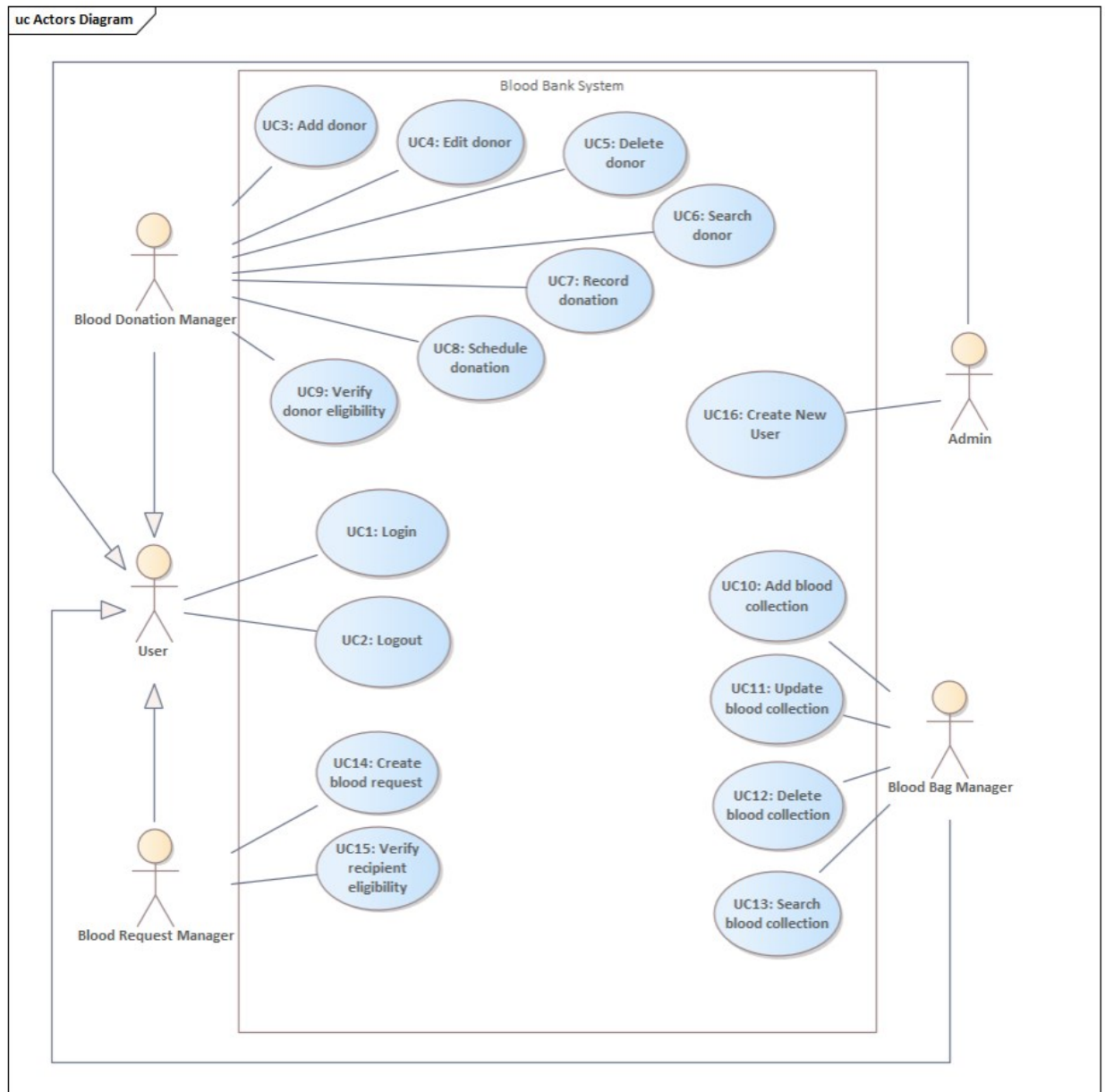


Figure 11 Use Case of Blood Bank Administration System

There are 5 actors in the system: *User*, *Blood Donation Manager*, *Blood Request Manager*, *Blood Bag Manager*, and *Admin*. The User actor simplifies the diagram because every actor can login and logout (User is the generalization of each actor). The admin actor can create new user to the system. Blood Bag Manager manages blood bags; may add, update, delete, and search blood bags. Blood Request Manager manages requests and issuances of the blood bags. Finally, Blood Donation Manager manages donors and donations of blood.

3.2.1 Use cases specifications

In this section, the specifications for each use case identified in the design of the blood bank management system are detailed. Each use case specification is presented in a structured format, outlining the actors involved, the actions performed, and the system's responses. These specifications serve to illustrate the functional requirements of the system, ensuring clarity in the roles and interactions necessary for the system's operation. Tables are used to concisely present these details, offering a clear and organized view of the functionalities each actor can perform within the system. This structured approach aids in understanding the system's capabilities and ensures comprehensive coverage of all intended functionalities.

Table 1. UC1 Login use case specification.

| | |
|------------------------|--|
| Use Case ID: | UC1 |
| Use Case Name: | Login |
| Actor: | User, Blood Donation Manager, Blood Request Manager, Admin, Blood Bag Manager |
| Description: | This use case allows actors (User, Blood Donation Manager, Blood Request Manager, Admin, Blood Bag Manager) to authenticate themselves in the system to access their respective functionalities and data. |
| Preconditions: | <ol style="list-style-type: none"> 1. The actor must have an existing account with a username and password. 2. The system is operational and accessible. |
| Postconditions: | <ol style="list-style-type: none"> 1. If authentication is successful, the actor gains access to their designated functionalities and information based on their role. 2. If authentication fails, the system remains on the login page, and the actor does not gain access. |
| Priority: | High - This use case is crucial as it controls access to the system and ensures that only authorized users can perform operations. |
| Frequency of Use: | Very High - This use case will be executed by all actors each time they wish to access the system. |
| Primary Scenario: | <ol style="list-style-type: none"> 1. The actor enters their username and password. 2. The system validates the credentials. 3. Upon successful validation, the system directs the actor to their respective dashboard based on their role. |
| Alternative Scenarios: | <p>Incorrect credentials:</p> <ol style="list-style-type: none"> 1. The actor enters an incorrect username or password. 2. The system displays an error message indicating incorrect credentials. |
| Exceptions: | <ul style="list-style-type: none"> - System maintenance - Network issues |
| Includes: | None |

Table 2. UC2 Logout use case specification.

| | |
|------------------------|---|
| Use Case ID: | UC2 |
| Use Case Name: | Logout |
| Actor: | User, Blood Donation Manager, Blood Request Manager, Admin, Blood Bag Manager |
| Description: | This use case enables all system actors to securely log out from the system, ensuring their session is properly closed and their credentials are protected from unauthorized access. |
| Preconditions: | <ol style="list-style-type: none"> 1. The actor must be logged in to the system. 2. The system is operational and accessible. |
| Postconditions: | <ol style="list-style-type: none"> 1. The system is operational and accessible. 2. The session is terminated, and the actor is redirected to the login page. |
| Priority: | High - This use case is critical for maintaining security and data integrity by ensuring that users can effectively log out when they have finished their tasks. |
| Frequency of Use: | Very High - This use case will be executed by all actors each time they finish their session in the system. |
| Primary Scenario: | <ol style="list-style-type: none"> 1. The actor selects the logout option in the system interface. 2. The system ends the actor's session. 3. The system redirects the actor to the login page, confirming that they have been logged out. |
| Alternative Scenarios: | None |
| Exceptions: | <ul style="list-style-type: none"> - System maintenance - Network issues |
| Includes: | None |

Table 3. UC3 Add donor use case specification.

| | |
|------------------------|---|
| Use Case ID: | UC3 |
| Use Case Name: | Add donor |
| Actor: | Blood Donation Manager |
| Description: | This use case allows the Blood Donation Manager to register new donors into the system. The process includes capturing essential donor information such as personal details, health history, and blood type to ensure that donors are properly documented and managed within the system. |
| Preconditions: | <ol style="list-style-type: none"> 1. The actor must be logged into the system as a Blood Donation Manager. 2. The system is operational and accessible. |
| Postconditions: | <ol style="list-style-type: none"> 1. The donor's information is recorded in the system. |
| Priority: | High - Proper registration of donors is crucial for maintaining an up-to-date donor database and facilitating efficient blood donation management. |
| Frequency of Use: | Moderate - The frequency of this use case depends on the rate of donor recruitment activities. |
| Primary Scenario: | <ol style="list-style-type: none"> 1. The Blood Donation Manager navigates to the "Donor" section in the system and clicks on "Add Donor". 2. The system presents a form to enter donor details, including name, age, contact number, blood type, last donation date. 3. The Blood Donation Manager fills in the necessary information and submits the form. 4. The system validates the provided information for completeness and correctness. 5. Upon successful validation, the system records the donor's information and generates a unique donor ID. |
| Alternative Scenarios: | - Incomplete form submission => Display warning messages |
| Exceptions: | <ul style="list-style-type: none"> - System maintenance - Network issues |
| Includes: | None |

Table 4. UC4 Edit donor use case specification.

| | |
|------------------------|--|
| Use Case ID: | UC4 |
| Use Case Name: | Edit donor |
| Actor: | Blood Donation Manager |
| Description: | This use case allows the Blood Donation Manager to modify existing donor records in the system. It involves updating personal details, contact information, health history, and other relevant data to ensure that the donor profiles are kept current and accurate. |
| Preconditions: | <ol style="list-style-type: none"> 1. The actor must be logged into the system as a Blood Donation Manager. 2. The donor record to be edited must already exist in the system. 3. The system is operational and accessible. |
| Postconditions: | <ol style="list-style-type: none"> 1. The selected donor's information is updated in the system. |
| Priority: | High - Keeping donor information accurate and up to date is essential for the integrity of the donor management process and compliance with health regulations. |
| Frequency of Use: | Moderate - The frequency depends on the need for updates in donor information, which may vary. |
| Primary Scenario: | <ol style="list-style-type: none"> 1. The Blood Donation Manager navigates to the "Donor". 2. The system presents a table of which it contains all donors, each donor has it is own actions (Modify, Delete). 3. The Blood Donation Manager modify a record and submit. 4. The Donor inforamtion is updated. |
| Alternative Scenarios: | - Incomplete form submission => Display warning messages |
| Exceptions: | <ul style="list-style-type: none"> - System maintance - Network issues |
| Includes: | None |

Table 5. UC5 Delete donor use case specification.

| | |
|------------------------|--|
| Use Case ID: | UC5 |
| Use Case Name: | Delete donor |
| Actor: | Blood Donation Manager |
| Description: | This use case allows the Blood Donation Manager to delete existing donor records in the system. |
| Preconditions: | <ol style="list-style-type: none"> 1. The actor must be logged into the system as a Blood Donation Manager. 2. The donor record to be deleted must already exist in the system. 3. The system is operational and accessible. |
| Postconditions: | <ol style="list-style-type: none"> 1. The selected donor is deleted from the system's database. |
| Priority: | Medium - This use case is important for maintaining the accuracy and relevance of the donor database but is less frequent than adding or editing donor records. |
| Frequency of Use: | Low - This use case is expected to be used infrequently, as the deletion of donor records should be a controlled and rare event. |
| Primary Scenario: | <ol style="list-style-type: none"> 1. The Blood Donation Manager navigates to the "Donor" section in the system and clicks on "Add Donor". 2. The system presents a table of which it contains all donors, each donor has it is own actions (Modify, Delete). 3. The Blood Donation Manager delete a record. 4. The system prompts the Blood Donation Manager to confirm the deletion to prevent accidental data loss. 5. Upon confirmation, the system permanently removes the donor's record. |
| Alternative Scenarios: | - Cancellation of deletion |
| Exceptions: | <ul style="list-style-type: none"> - System maintance - Network issues |
| Includes: | None |

Table 6. UC6 Search donor use case specification.

| | |
|------------------------|---|
| Use Case ID: | UC6 |
| Use Case Name: | Search donor |
| Actor: | Blood Donation Manager |
| Description: | This use case allows the Blood Donation Manager to search for specific donor by name. |
| Preconditions: | <ol style="list-style-type: none"> 1. The actor must be logged into the system as a Blood Donation Manager. 2. The system contains donor records that can be searched |
| Postconditions: | <ol style="list-style-type: none"> 1. The system displays the search results based on the criteria provided by the Blood Donation Manager. 2. If no records match the search criteria, the system informs the Blood Donation Manager. |
| Priority: | High - This use case is essential for effective donor management, allowing quick and easy access to donor records. |
| Frequency of Use: | High - Given the need to manage donor information regularly, this use case will be used frequently by Blood Donation Managers. |
| Primary Scenario: | <ol style="list-style-type: none"> 1. The Blood Donation Manager accesses the donor search function in the system, available in the donors page. 2. The system presents a search interface where the manager can search by name. 3. The Blood Donation Manager inputs the desired search name and submits the search request. 4. The system processes the search and retrieves records that match the criteria. 5. The system displays the search results to the Blood Donation Manager. |
| Alternative Scenarios: | - No result found. |
| Exceptions: | <ul style="list-style-type: none"> - System maintenance - Network issues |
| Includes: | None |

Table 7. UC7 Record donation use case specification.

| | |
|-------------------|---|
| Use Case ID: | UC7 |
| Use Case Name: | Record donation |
| Actor: | Blood Donation Manager |
| Description: | This use case involves the Blood Donation Manager recording details of a blood donation event into the system. It covers capturing information such as blood type, donor's name, contact number, and quantity. |
| Preconditions: | <ol style="list-style-type: none"> 1. The actor must be logged into the system as a Blood Donation Manager. 2. The donor's record is updated with the latest donation. |
| Postconditions: | <ol style="list-style-type: none"> 1. The donation details are accurately recorded in the system. 2. If no records match the search criteria, the system informs the Blood Donation Manager. |
| Priority: | High - Accurate and timely recording of donations is critical for maintaining the integrity of the blood inventory and donor records. |
| Frequency of Use: | High - This use case will be executed whenever a donation is made, which could be daily depending on the size and activity level of the blood bank. |
| Primary Scenario: | <ol style="list-style-type: none"> 1. The Blood Donation Manager navigates to the blood list page in the system. 2. The manager selects the option to add blood, prompting the donation form to appear. 3. The Blood Donation Manager fills out the donation form with details including blood type, donor's name, contact number, and quantity. 4. The Blood Donation Manager submits the form. 5. The system validates the entered information for completeness and correctness. 6. Upon successful validation, the system records the donation in the donor's profile and updates the blood inventory. |

| | |
|------------------------|---|
| Alternative Scenarios: | <p>Incomplete or Incorrect Information:</p> <ul style="list-style-type: none">- If the information provided is incomplete or fails validation checks, the system alerts the Blood Donation Manager.- The Blood Donation Manager is prompted to correct the information and resubmit. |
| Exceptions: | <ul style="list-style-type: none">- System maintenance- Network issues |
| Includes: | None |

Table 8. UC8 Schedule donation use case specification.

| | |
|------------------------|--|
| Use Case ID: | UC8 |
| Use Case Name: | Schedule donation |
| Actor: | Blood Donation Manager |
| Description: | This use case allows the Blood Donation Manager to schedule future blood donation appointments for donors. It involves inserting donor's name, age, contact number, blood type, last donation date and an additional message. |
| Preconditions: | 1. The actor must be logged into the system as a Blood Donation Manager. |
| Postconditions: | 1. A donation appointment is successfully scheduled in the system. |
| Priority: | High - Scheduling donations efficiently helps manage the blood supply and ensures that donor appointments are planned according to need and donor availability. |
| Frequency of Use: | High - Depending on the number of active donors and organizational needs, this use case might be used multiple times daily. |
| Primary Scenario: | <ol style="list-style-type: none"> 1. The Blood Donation Manager accesses the "Schedule Donation" function within the system. 2. The system displays a form to enter appointment details, including the donor's ID, preferred date and time, and location. 3. The Blood Donation Manager fills out the form based on the donor's availability and the blood bank's scheduling needs. 4. The Blood Donation Manager submit the form 5. Upon successful, the appoitment is successfully registered in the system. |
| Alternative Scenarios: | <ul style="list-style-type: none"> - Unavailable Slot - Donor Reschedules |
| Exceptions: | <ul style="list-style-type: none"> - System maintance - Network issues |
| Includes: | None |

Table 9. UC9 Verify donor eligibility use case specification.

| | |
|-------------------|--|
| Use Case ID: | UC9 |
| Use Case Name: | Verify donor eligibility |
| Actor: | Blood Donation Manager |
| Description: | This use case involves the Blood Donation Manager manually verifying the eligibility of potential donors based on pre-defined health criteria and donation history before accepting a donation. This manual process ensures that all donors meet safety and health regulations, which is crucial for the safety of both donors and recipients. |
| Preconditions: | <ol style="list-style-type: none"> 1. The actor has access to the potential donor's detailed health and donation history. 2. The potential donor has been identified and is ready to be assessed for donation. |
| Postconditions: | <ol style="list-style-type: none"> 1. The donor's eligibility for donation is confirmed or denied based on medical and regulatory criteria.. 2. Documentation is updated to reflect the verification outcome |
| Priority: | High - Ensuring that only eligible donors participate in blood donation is critical for maintaining the safety and quality of the blood supply. |
| Frequency of Use: | High - This task is performed each time a new or returning donor is considered for donation. |
| Primary Scenario: | <ol style="list-style-type: none"> 1. The Blood Donation Manager receives information about a potential donor who needs to be assessed for eligibility. 2. The manager accesses the donor's medical and donation history records, which may be available in electronic or physical format. 3. The Blood Donation Manager reviews the donor's records focusing on critical factors such as age, weight, medical conditions, medications, recent travel, and previous donation intervals. 4. The manager compares the donor's information against the established eligibility criteria to determine if the donor can safely donate blood. 5. The outcome of the eligibility verification is documented in the donor's profile, noting whether the donor is approved or denied for donation. |

| | |
|------------------------|--|
| | 6. The Blood Donation Manager updates any necessary systems or records to reflect the verification status. |
| Alternative Scenarios: | <p>Additional Information Required:</p> <ol style="list-style-type: none">1. If the available information is insufficient to verify eligibility, the Blood Donation Manager contacts healthcare providers or the donor for additional necessary data.2. The verification remains pending until the required information is provided |
| Exceptions: | <ul style="list-style-type: none">- System maintenance- Network issues- Documentation issues |
| Includes: | None |

Table 10. UC10 Add blood collection use case specification.

| | |
|------------------------|--|
| Use Case ID: | UC10 |
| Use Case Name: | Add blood collection |
| Actor: | Blood Bag Manager |
| Description: | This use case allows the Blood Bag Manager to add details of newly collected blood units into the system. It involves recording information such as blood type, donor name, donor's contact number and the quantity to ensure proper tracking and management of the blood inventory. |
| Preconditions: | <ol style="list-style-type: none"> 1. The actor must be logged into the system as a Blood Bag Manager. 2. The blood collection must have been completed and all relevant information is ready for entry. |
| Postconditions: | <ol style="list-style-type: none"> 1. The new blood unit's details are accurately recorded in the system. 2. The blood inventory is updated to reflect the addition of the new unit. |
| Priority: | High - Accurate and timely entry of blood unit information is critical for managing the blood bank's inventory and ensuring availability for patients in need. |
| Frequency of Use: | High - This use case is executed each time a new blood unit is collected and needs to be added to the inventory. |
| Primary Scenario: | <ol style="list-style-type: none"> 1. The Blood Bag Manager accesses the "Register Blood Collection" function in the system (Blood list => Add blood). 2. The system presents a form to enter details of the new blood unit, including blood type, donor name, donor's contact number and the quantity. 3. The Blood Bag Manager fills out the form with the necessary information and submits it. 4. The system validates the provided information for completeness and correctness. 5. Upon successful validation, the system updates the blood inventory and logs the new blood unit. |
| Alternative Scenarios: | - Incomplete form submission => Display warning messages |

| | |
|-------------|---|
| Exceptions: | <ul style="list-style-type: none">- System maintance- Network issues |
| Includes: | None |

Table 11. UC11 Update blood collection use case specification.

| | |
|------------------------|--|
| Use Case ID: | UC11 |
| Use Case Name: | Update blood collection |
| Actor: | Blood Bag Manager |
| Description: | This use case allows the Blood Bag Manager to update existing records of blood collections in the system. It involves modifying details such as the blood type, donor name, contact number and quantity of the collected blood. |
| Preconditions: | <ol style="list-style-type: none"> 1. The actor must be logged into the system as a Blood Bag Manager. 2. The blood collection record to be updated must already exist in the system. 3. The system is operational and accessible. |
| Postconditions: | <ol style="list-style-type: none"> 1. The specified blood collection record is updated with the new details. 2. The system reflects the changes accurately in the blood inventory. |
| Priority: | High - Keeping blood collection records accurate and up to date is essential for effective blood inventory management and ensuring the quality and safety of the blood supply. |
| Frequency of Use: | Medium - This use case is used as needed, depending on the frequency of changes or corrections to blood collection records. |
| Primary Scenario: | <ol style="list-style-type: none"> 1. The Blood Bag Manager navigates to the "Blood list". 2. The system presents a table of which it contains all blood collections, each collection has it is own actions (Modify, Delete). 3. The Blood Bag Manager modify a blood collection and submit. 4. The blood collection information is updated. |
| Alternative Scenarios: | - Incomplete form submission => Display warning messages |
| Exceptions: | <ul style="list-style-type: none"> - System maintenance - Network issues |
| Includes: | None |

Table 12. UC12 Delete blood collection use case specification.

| | |
|------------------------|--|
| Use Case ID: | UC12 |
| Use Case Name: | Delete blood collection |
| Actor: | Blood Bag Manager |
| Description: | This use case allows the Blood Bag Manager to permanently remove a blood collection record from the system. |
| Preconditions: | <ol style="list-style-type: none"> 1. The actor must be logged into the system as a Blood Bag Manager. 2. The blood collection record to be deleted must already exist in the system. 3. The system is operational and accessible. |
| Postconditions: | <ol style="list-style-type: none"> 1. The selected blood collection is deleted from the system's database. |
| Priority: | Medium - This use case is important for maintaining the accuracy and relevance of the blood inventory but is less frequent than adding or editing blood collection records. |
| Frequency of Use: | Low - This use case is expected to be used infrequently, as the deletion of blood collection records should be a controlled and rare event. |
| Primary Scenario: | <ol style="list-style-type: none"> 1. The Blood Bag Manager navigates to the "Blood list" page. 2. The system presents a table of which it contains all blood collections, each blood collection has it is own actions (Modify, Delete). 3. The Blood Bag Manager delete a record. 4. The system prompts the Blood Bag Manager to confirm the deletion to prevent accidental data loss. 5. Upon confirmation, the system permanently removes the blood collection record. |
| Alternative Scenarios: | - Cancellation of deletion |
| Exceptions: | <ul style="list-style-type: none"> - System maintance - Network issues |
| Includes: | None |

Table 13. UC13 Search blood collection use case specification.

| | |
|------------------------|---|
| Use Case ID: | UC13 |
| Use Case Name: | Search blood collection |
| Actor: | Blood Bag Manager |
| Description: | This use case enables the Blood Bag Manager to search for records of blood collections in the system using blood type. This functionality facilitates efficient management and tracking of the blood inventory. |
| Preconditions: | <ol style="list-style-type: none"> 1. The actor must be logged into the system as a Blood Bag Manager. 2. The system contains blood collection records that can be searched. |
| Postconditions: | <ol style="list-style-type: none"> 1. The system displays the search results based on the criteria provided by the Blood Bag Manager. 2. If no records match the search criteria, the system informs the Blood Bag Manager. |
| Priority: | High - Efficient searching of blood collection records is essential for managing the blood inventory and ensuring that blood units are tracked and utilized effectively. |
| Frequency of Use: | High - This use case will likely be used frequently to manage inventory and respond to requests for specific blood types or volumes. |
| Primary Scenario: | <ol style="list-style-type: none"> 1. The Blood Bag Manager accesses the blood collection search function in the system, available in the donors page. 2. The system presents a search interface where the manager can search by blood type. 3. The Blood Bag Manager inputs the desired blood type and submits the search request. 4. The system processes the search and retrieves records that match the criteria. 5. The system displays the search results to the Blood Donation Manager. |
| Alternative Scenarios: | - No result found. |
| Exceptions: | <ul style="list-style-type: none"> - System maintenance - Network issues |
| Includes: | None |

Table 14. UC14 Create blood request use case specification.

| | |
|-------------------|--|
| Use Case ID: | UC14 |
| Use Case Name: | Create blood request |
| Actor: | Blood Request Manager |
| Description: | This use case allows the Blood Request Manager to create a new request for blood units within the system. It involves specifying the receiver's name, age, contact number, requested blood type, last received date, and medical conditions. This functionality is crucial for managing the distribution of blood inventory to meet hospital or clinic needs. |
| Preconditions: | <ol style="list-style-type: none"> 1. The actor must be logged into the system as a Blood Request Manager. 2. There must be sufficient understanding of the current blood inventory levels to ensure realistic requests. |
| Postconditions: | <ol style="list-style-type: none"> 1. A new blood request is successfully created and recorded in the system. 2. The request details are logged for inventory management and future tracking.. |
| Priority: | High - This use case is critical for ensuring that blood products are available where and when they are needed, supporting effective patient care. |
| Frequency of Use: | High - Depending on the needs of the healthcare facilities served, this use case may be used multiple times daily. |
| Primary Scenario: | <ol style="list-style-type: none"> 1. The Blood Request Manager accesses the "Schedule blood rrequest" page in the system. 2. The system presents a form to enter details of the new request. 3. The Blood Request Manager fills out the form with the necessary information and submits it. 4. The system validates the provided information for completeness and correctness. 5. Upon successful validation, the system updates the list of the blood requests. |

| | |
|------------------------|--|
| Alternative Scenarios: | - Incomplete form submission => Display warning messages |
| Exceptions: | - System maintance - Network issues |
| Includes: | None |

Table 15. UC15 Verify recipient eligibility use case specification.

| | |
|-------------------|---|
| Use Case ID: | UC15 |
| Use Case Name: | Verify Recipient Eligibility |
| Actor: | Blood Request Manager |
| Description: | This use case involves the Blood Request Manager manually verifying the eligibility of a recipient to receive a blood transfusion. This process includes checking medical history, existing conditions, and compatibility of the blood type, ensuring that all safety and health regulations are met before approving the transfusion. |
| Preconditions: | <ol style="list-style-type: none"> 1. The actor must have access to the recipient's detailed medical records. 2. The recipient has been identified as a potential transfusion recipient. |
| Postconditions: | <ol style="list-style-type: none"> 1. The recipient's eligibility for blood transfusion is confirmed or denied based on medical criteria. 2. Documentation is updated to reflect the verification outcome. |
| Priority: | High - This manual verification is crucial for ensuring the safety of blood transfusions and compliance with medical standards. |
| Frequency of Use: | Moderate - This task is performed whenever a new recipient is identified for a blood transfusion. |
| Primary Scenario: | <ol style="list-style-type: none"> 1. The Blood Request Manager receives a request for a blood transfusion for a recipient. 2. The manager accesses the recipient's medical records, either electronically or in physical form. 3. The Blood Request Manager reviews the recipient's medical history, focusing on factors like blood type compatibility, allergies, past transfusions, and any potential medical conditions that might affect transfusion safety. 4. The manager assesses all information to determine the recipient's eligibility based on established health and safety guidelines. |

| | |
|------------------------|---|
| | 5. The Blood Request Manager documents the outcome of the eligibility verification, updating the recipient's records with notes on the decision and any relevant observations. |
| Alternative Scenarios: | <p>Additional information required:</p> <ol style="list-style-type: none">1. If the information available is insufficient to make a determination, the Blood Request Manager contacts healthcare providers or the recipient's doctor to gather additional necessary data.2. The verification process is paused until the required information is received. |
| Exceptions: | <ul style="list-style-type: none">- System maintance- Network issues- Documentation Issues |
| Includes: | None |

Table 16. UC16 Create new user use case specification.

| | |
|-------------------|---|
| Use Case ID: | UC16 |
| Use Case Name: | Create New User |
| Actor: | Admin |
| Description: | This use case enables the admin to create new user accounts in the system. This includes setting up profiles for new employees or system users such as Blood Donation Managers, Blood Request Managers, Blood Bag Managers, and other relevant staff. The process involves assigning roles, setting permissions, and providing login credentials. |
| Preconditions: | <ol style="list-style-type: none"> 1. The actor must be logged into the system as an Admin. 2. The Admin has the necessary permissions to create user accounts. |
| Postconditions: | <ol style="list-style-type: none"> 1. A new user account is successfully created in the system. 2. The new user receives their login credentials (username and password) and role-specific access. |
| Priority: | High - Efficient management of user accounts is crucial for maintaining system security and ensuring that all personnel have the appropriate access to perform their duties. |
| Frequency of Use: | Low to Moderate - Depending on hiring rates and changes within the organization. |
| Primary Scenario: | <ol style="list-style-type: none"> 1. The Admin accesses the "Create New User" function in the system. 2. The system displays a form for entering new user details, including name, role, department, and contact information. 3. The Admin fills out the form with all required information. 4. The Admin submits the form to create the new user account. 5. The system validates the data to ensure it meets all criteria for a new user. 6. Upon successful validation, the system creates the new user account, assigns the specified role and permissions, and generates login credentials. |

| | |
|------------------------|---|
| | <ol style="list-style-type: none">7. The system sends the login credentials to the new user via a secure method.8. The system displays a confirmation message to the Admin about the successful creation of the new user account. |
| Alternative Scenarios: | <p>Invalid data entered:</p> <ol style="list-style-type: none">1. If the Admin enters invalid or incomplete information, the system displays an error message.2. The Admin is prompted to correct the details and resubmit the form. |
| Exceptions: | <ul style="list-style-type: none">- System maintenance- Network issues |
| Includes: | None |

3.3 Class Model

The class model is a representation of the classes' properties and methods as shown in Figure 12. The '-' symbol denotes non-public exposure, whereas the '+' sign denotes full transparency. Linking classes together based on their characteristics can establish associations. For example, blood donation may be connected to donor.

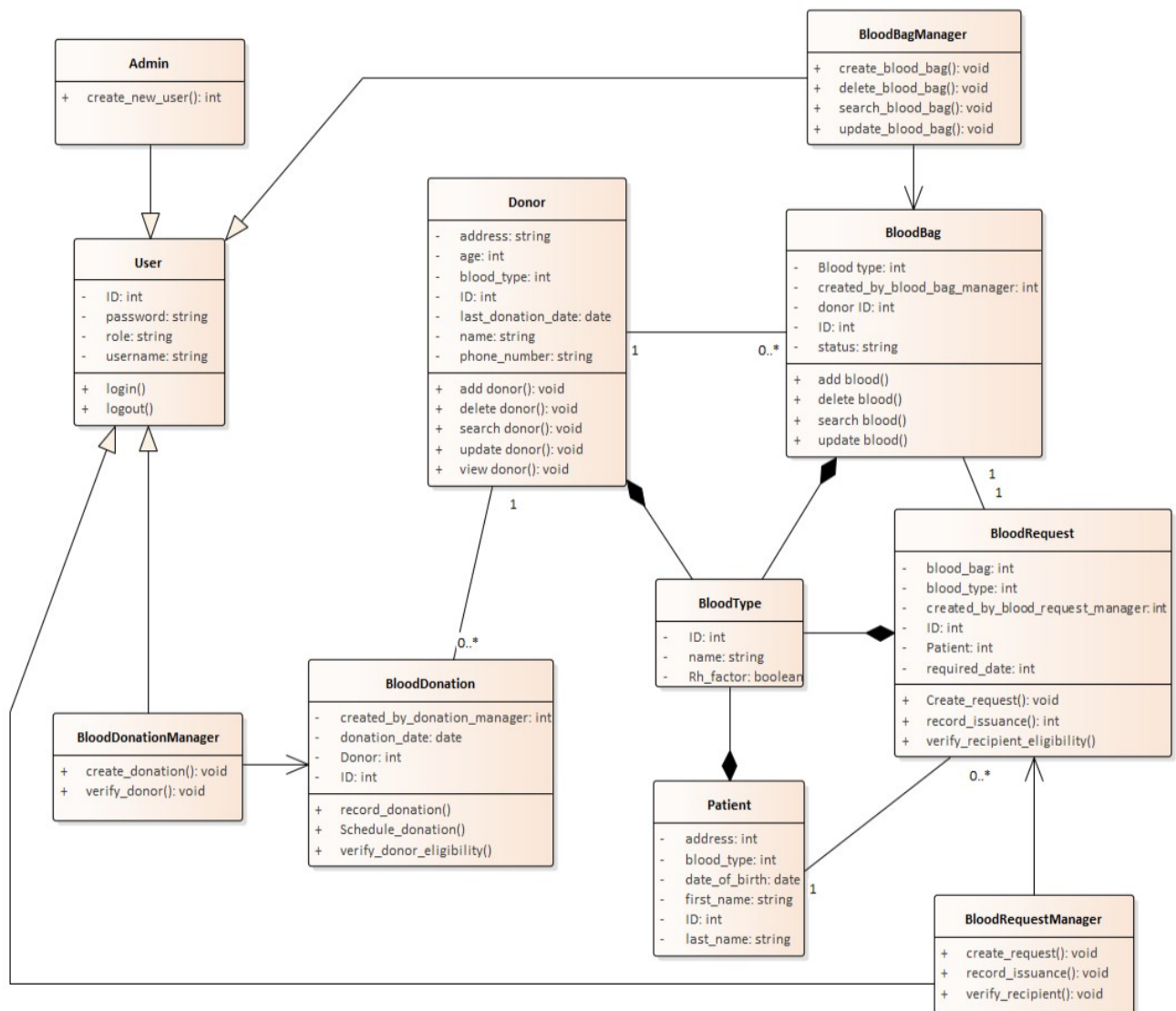


Figure 12 Class Model of Blood Bank Administration System

The proposed system is based on five main classes: Blood Donation, Donor, BloodBag, BloodRequest, and User.

As described in the use case chapter, there are 4 types of users: BloodDonationManager, BloodRequestManager, BloodBagManager, and Admin. The User class generalizes those four classes.

The patient is a person who requests the blood bag. The patient is not a user in the system, the Patient class exists just to store information about a blood recipient. The BloodType class is a composite part of the Donor, BloodBag, BloodRequest, and Patient. It means that these classes are not complete without the information about blood type. This condition is applied because this system is focused on the blood and therefore, those classes cannot work without this information.

In the diagram, we may see an association between BloodRequest and Patient: one patient may have zero or more requests. Between BloodRequest and BloodBag, there is an association one to one. It might happen that the patient needs more blood bags but in such a case, the patient must have more requests to obtain more bags. Each BloodBag is associated to a donor who can donate blood more than once and therefore, the donor may have many donations and many blood bags associated.

3.4 Sequence Model

This section explores sequence diagrams, which are essential tools in software engineering for visualizing interactions within systems over time. These diagrams illustrate the sequence of events that occur as different components of a system, such as users and system functionalities interactions. Each interaction is detailed, showing how processes unfold and the order in which they occur.

For the blood bank management system, sequence diagrams will be provided for each of the 16 use cases. These diagrams will delineate the roles involved, such as the Blood Donation Manager and Admin, and the parts of the system they interact with, like databases or servers. Accompanying each diagram, a brief narrative will describe the sequence of events, enhancing understanding of each interaction's purpose and mechanism.

Sequence diagrams are invaluable for visualizing the dynamic behavior of systems, aiding in the identification and resolution of potential issues during the design phase.

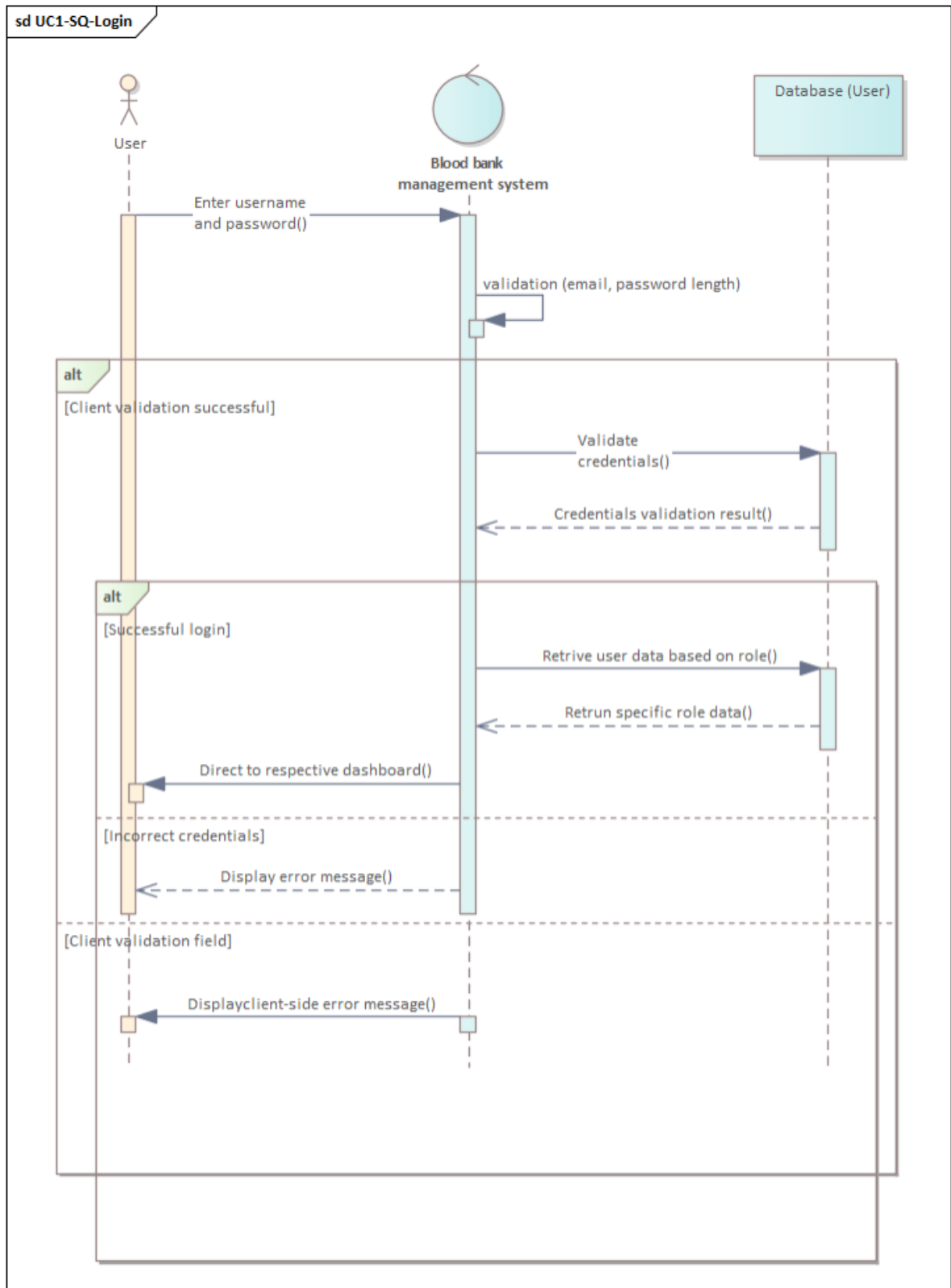


Figure 13. UC1 Login sequence diagram.

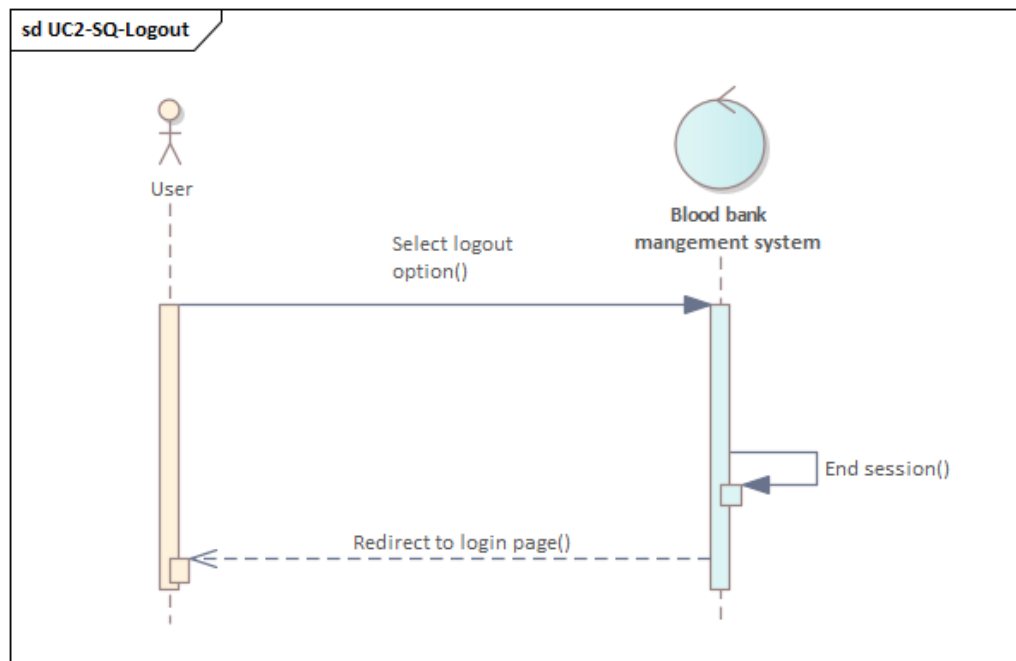


Figure 14. UC2 Logout sequence diagram.

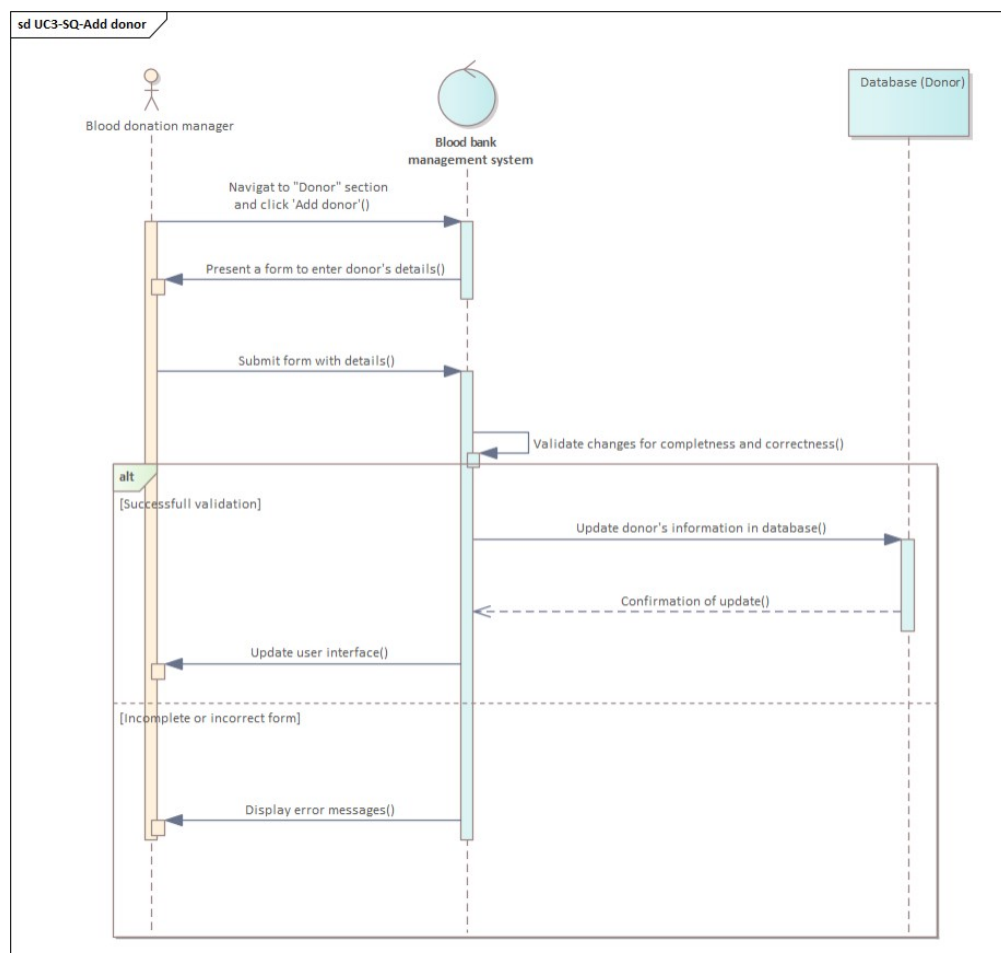


Figure 15. UC3 Add donor sequence diagram.

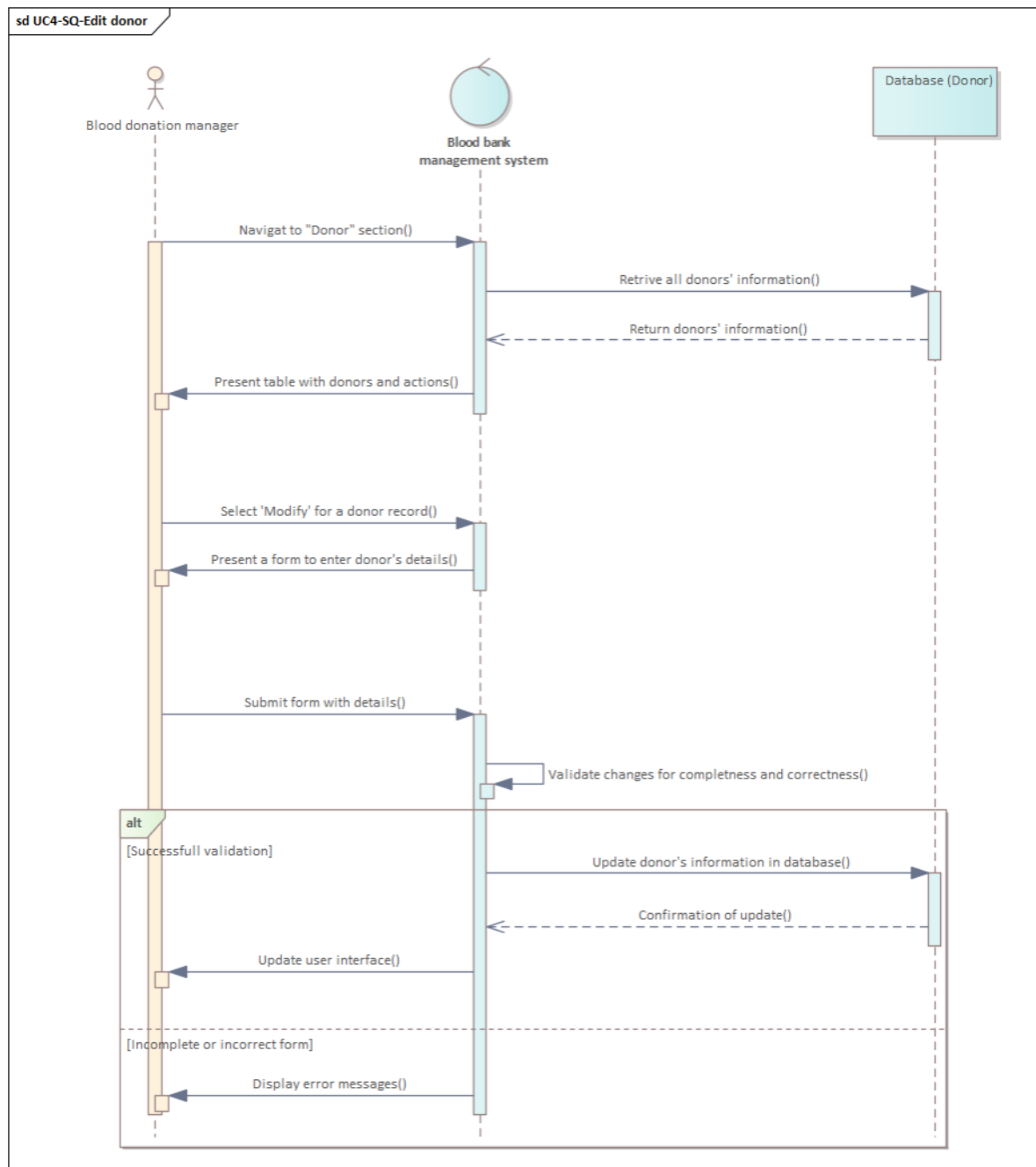


Figure 16. UC4 Edit donor sequence diagram.

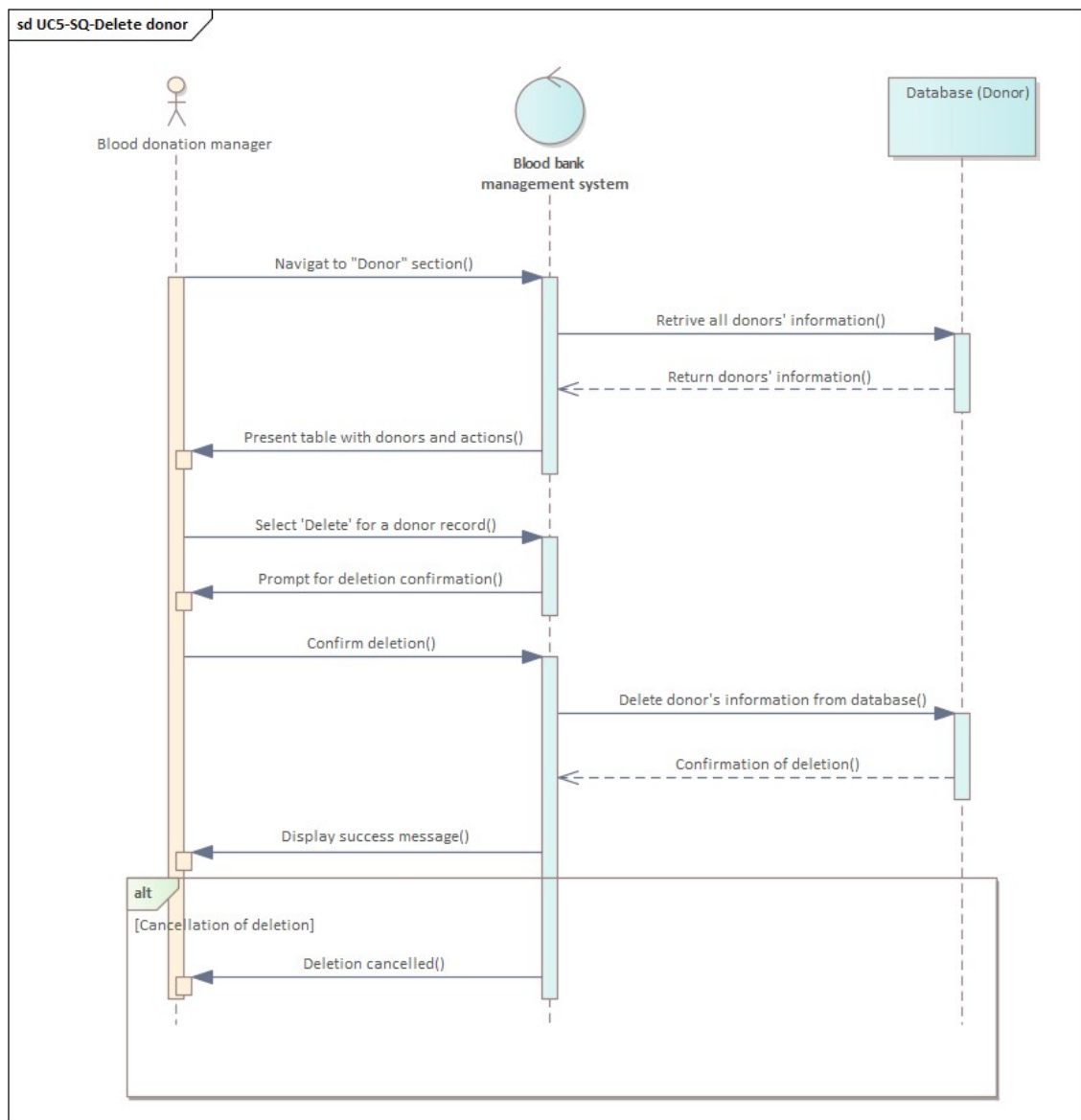


Figure 17. UC5 Delete donor sequence diagram.

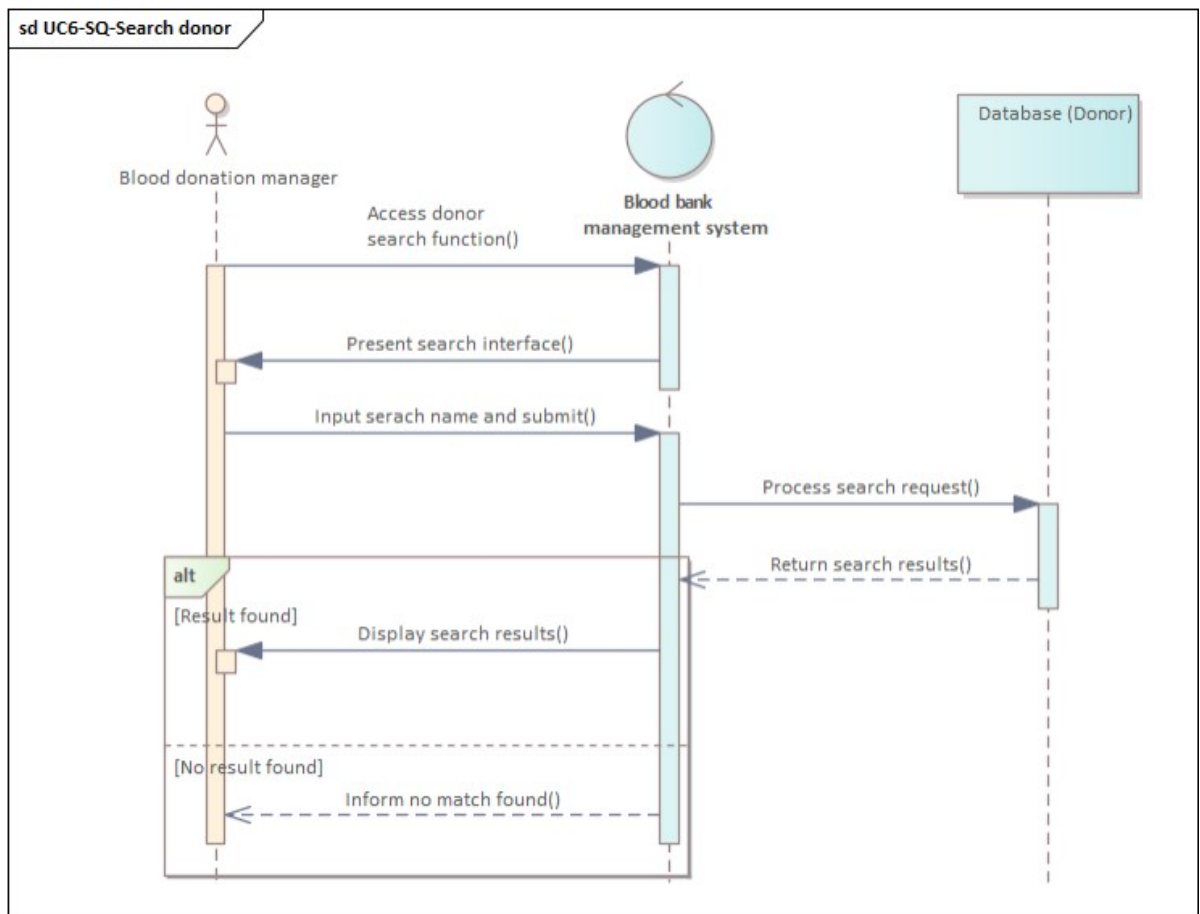


Figure 18.UC6 Search donor sequence diagram.

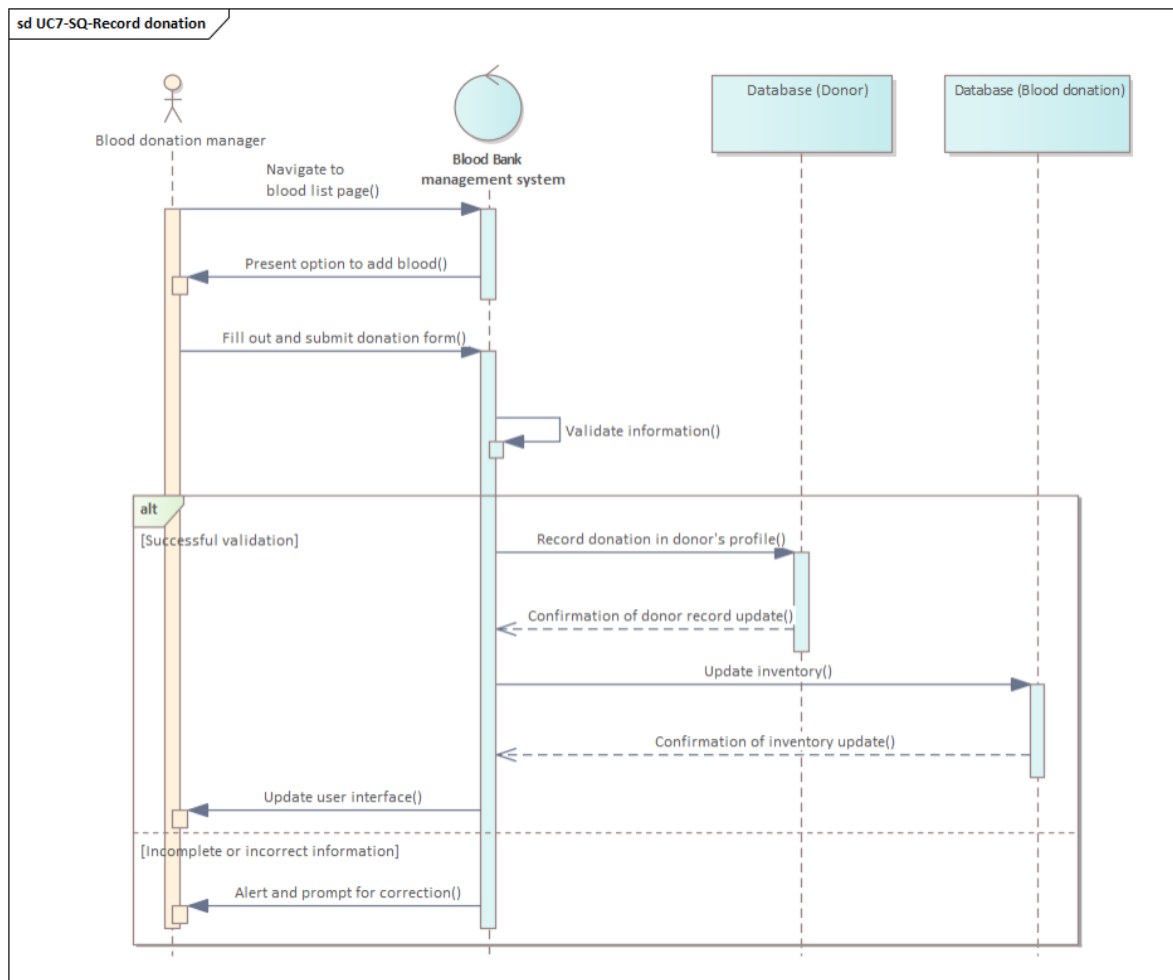


Figure 19. UC7 Record donation sequence diagram.

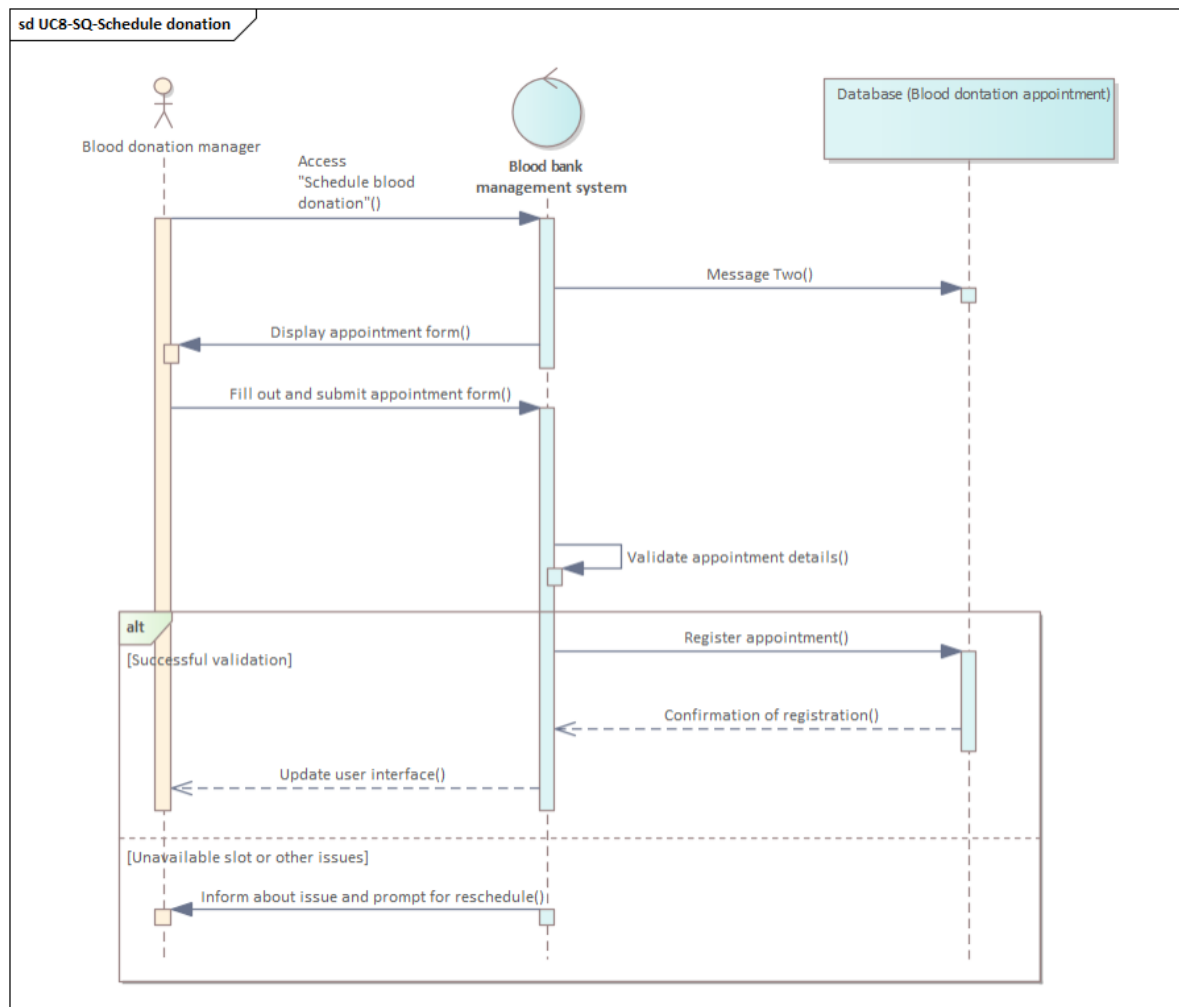


Figure 20. UC8 Schedule donation sequence diagram.

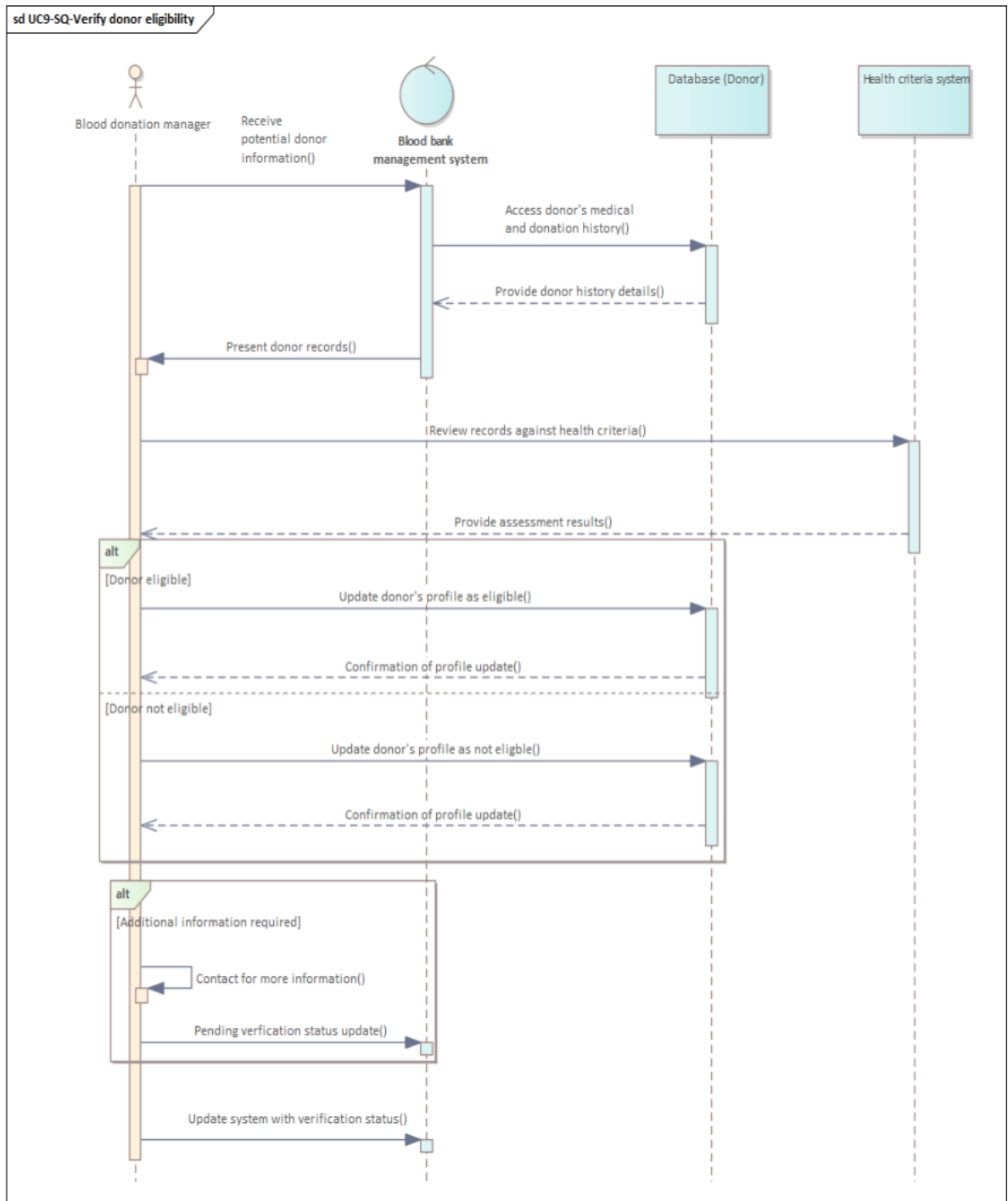


Figure 21. UC9 Verify donor eligibility sequence diagram.

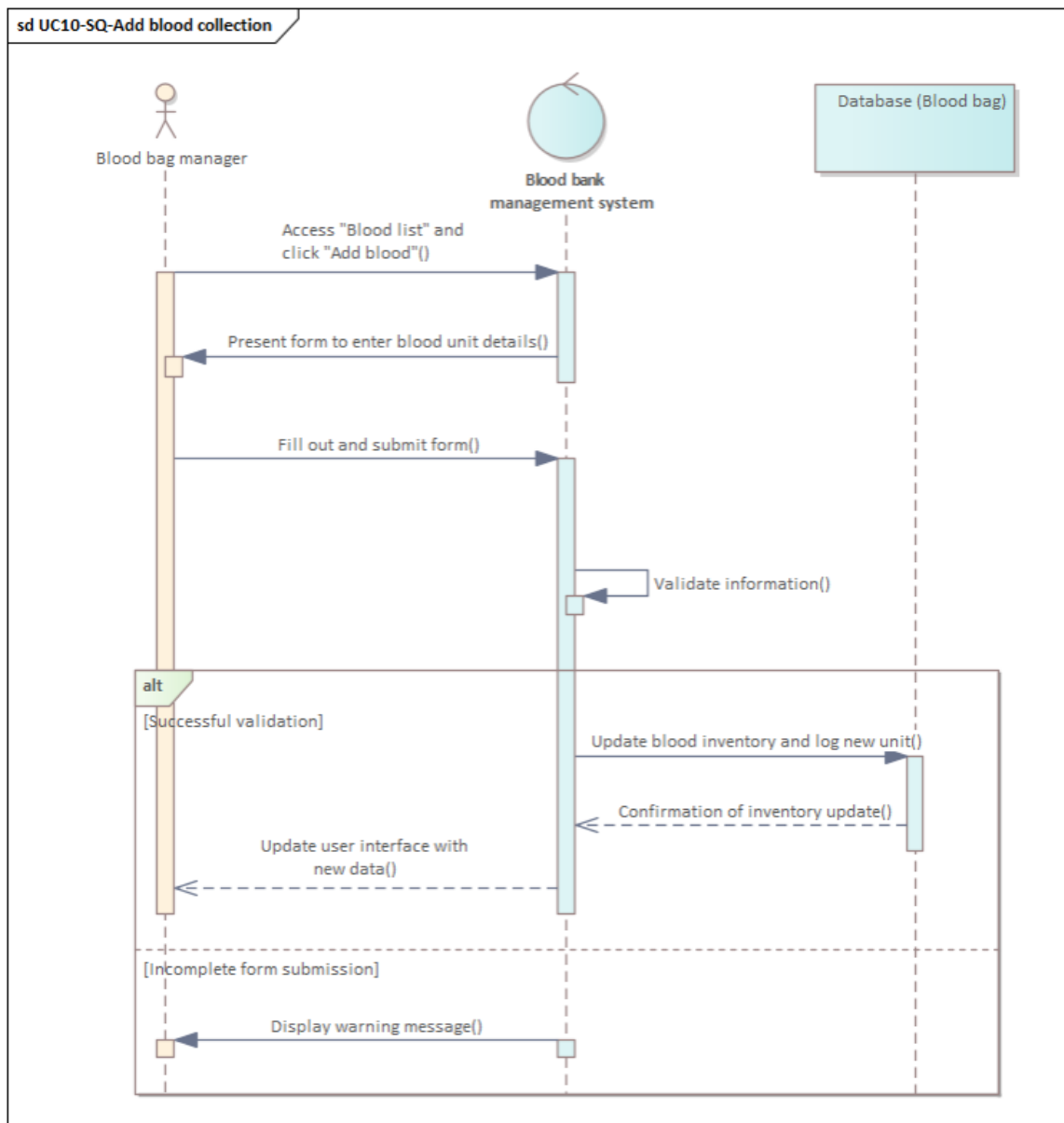


Figure 22. UC10 Add blood collection sequence diagram.

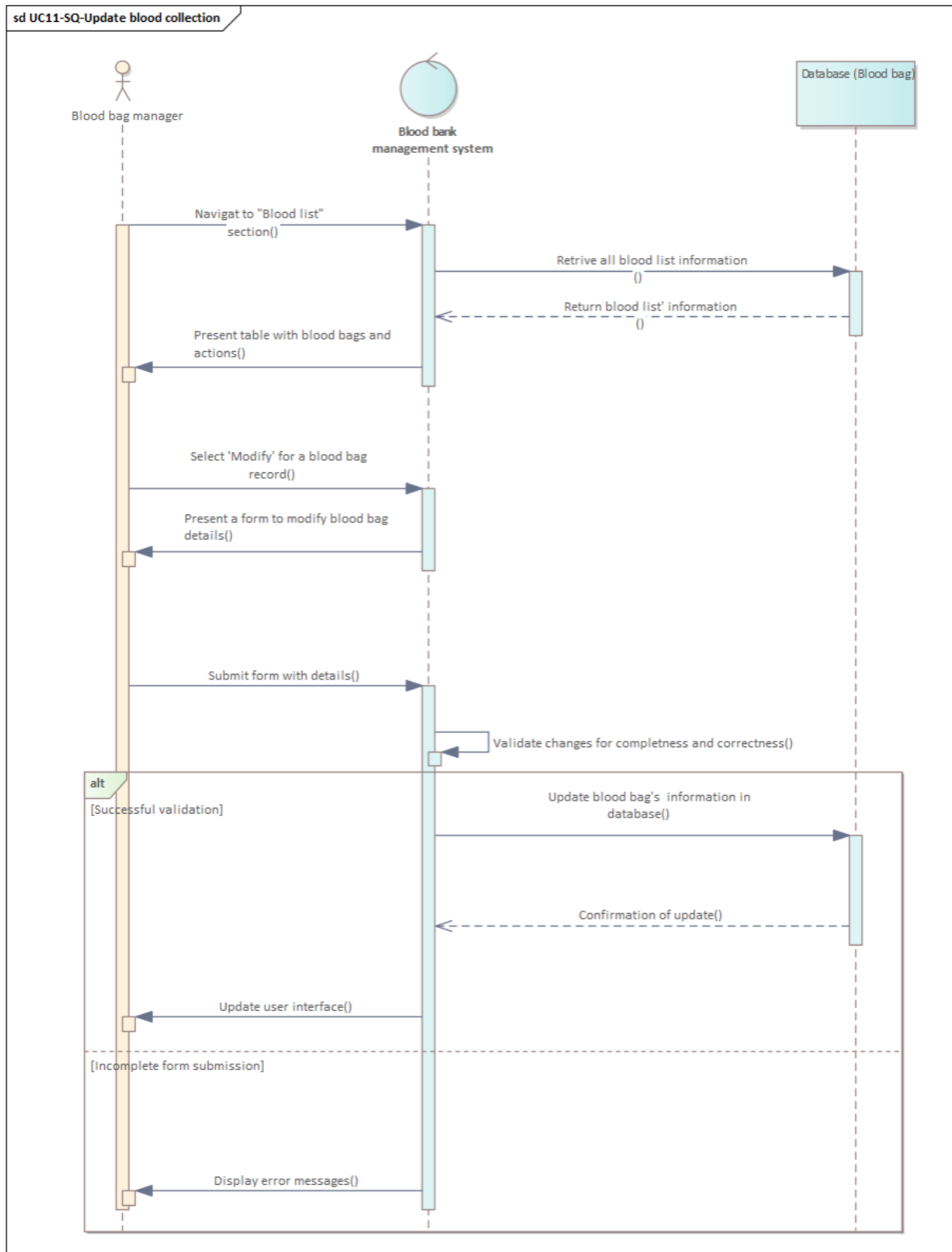


Figure 23. UC11 Update blood collection sequence diagram.

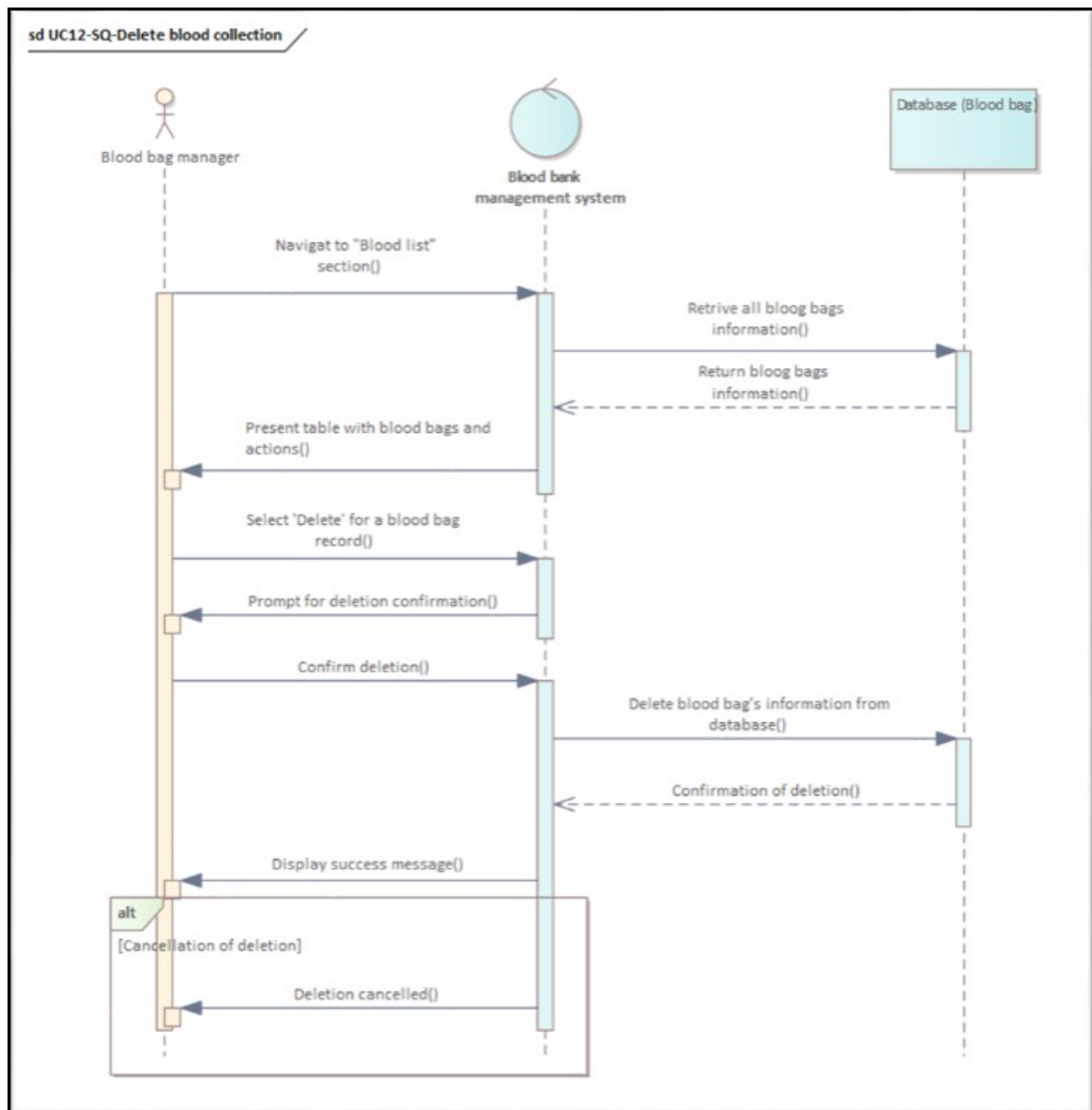


Figure 24. UC12 Delete blood collection sequence diagram.

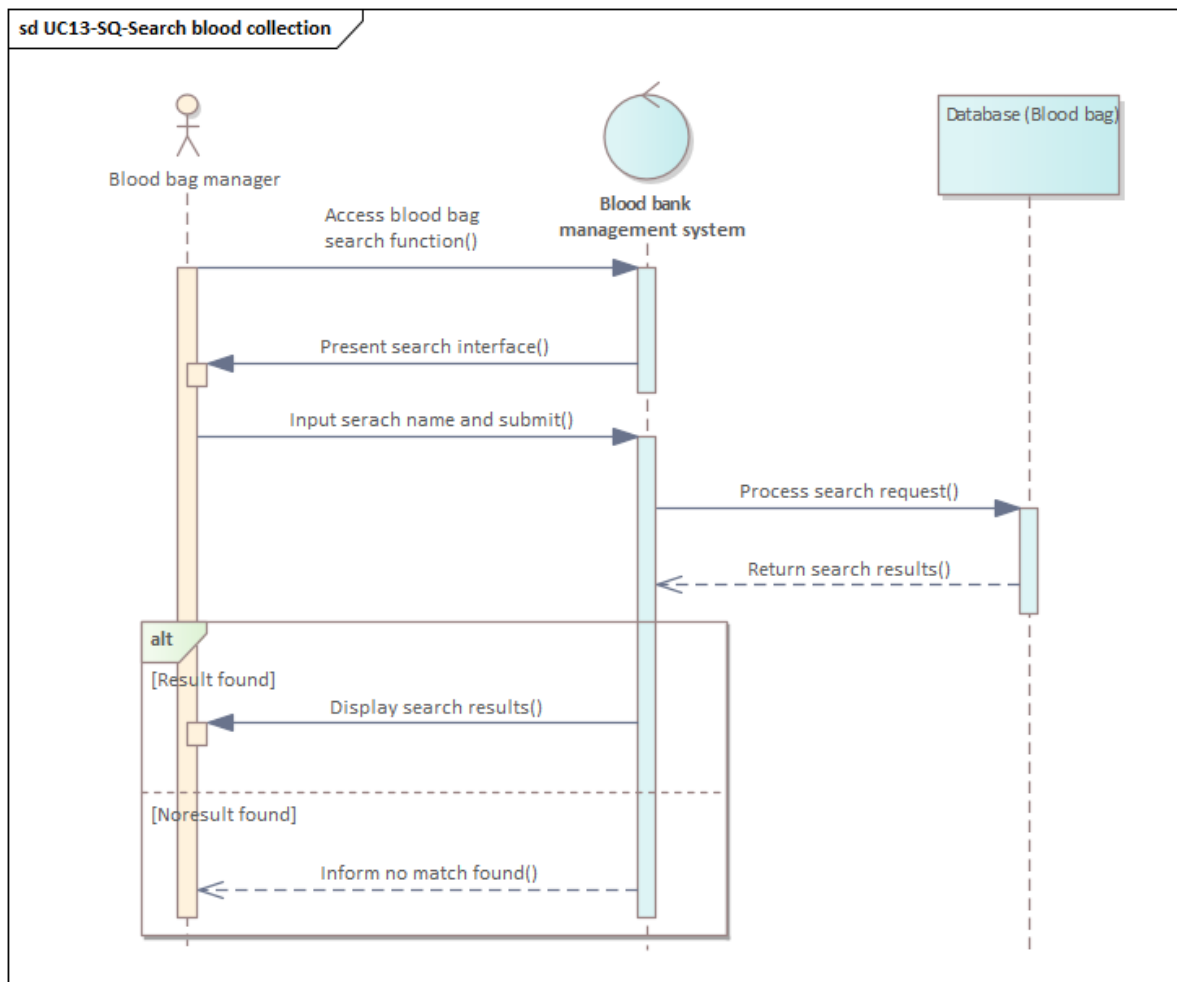


Figure 25. UC13 Search blood collection sequence diagram

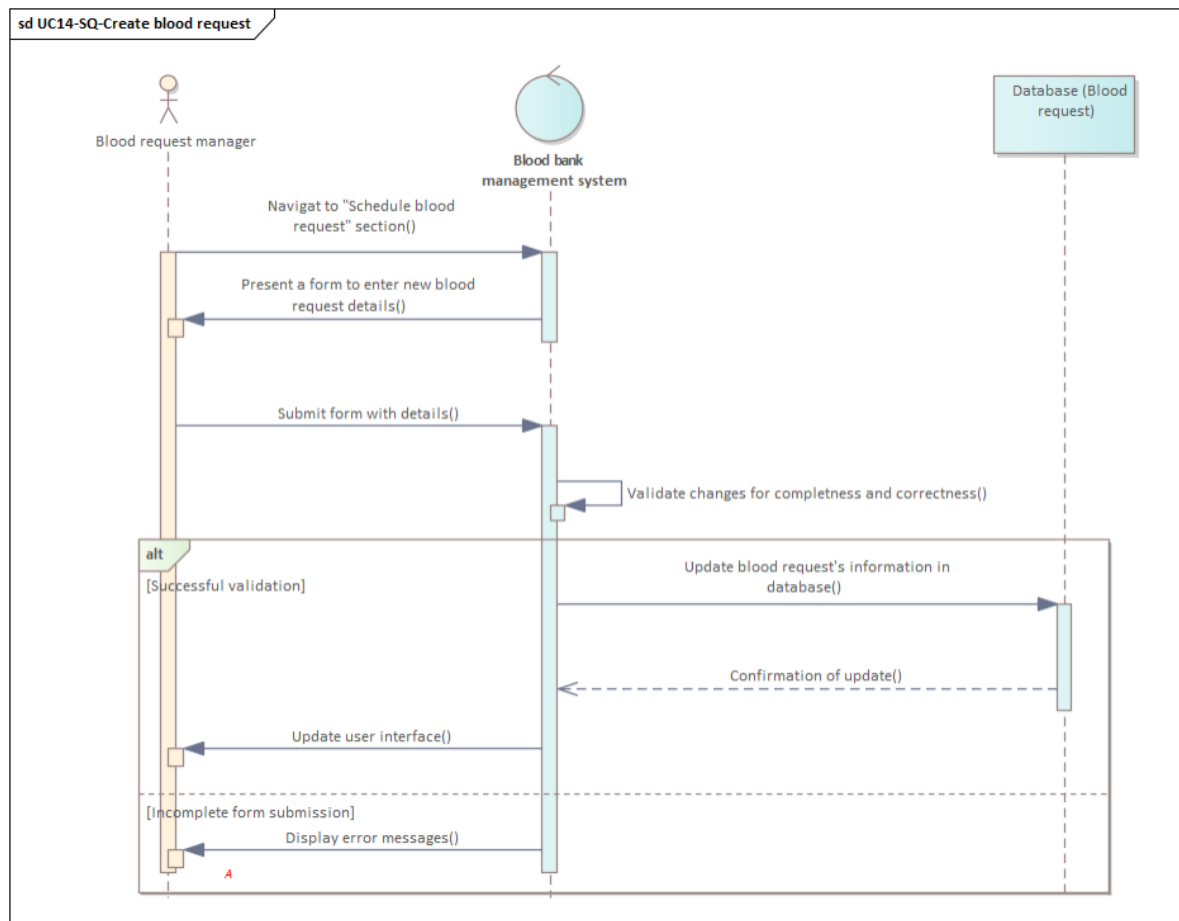


Figure 26. UC14 Create blood request sequence diagram

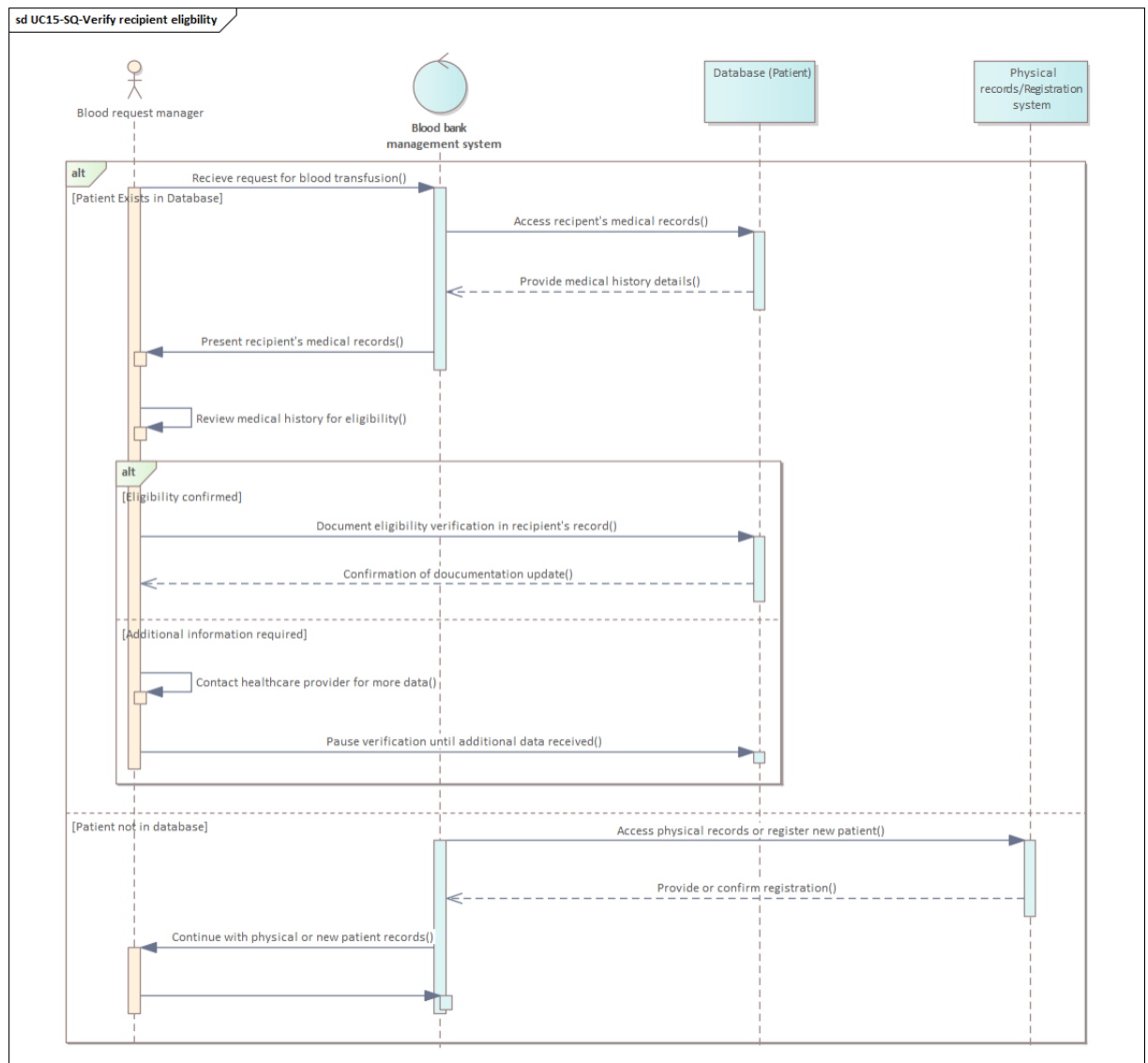


Figure 27. UC15 Verify recipient eligibility sequence diagram.

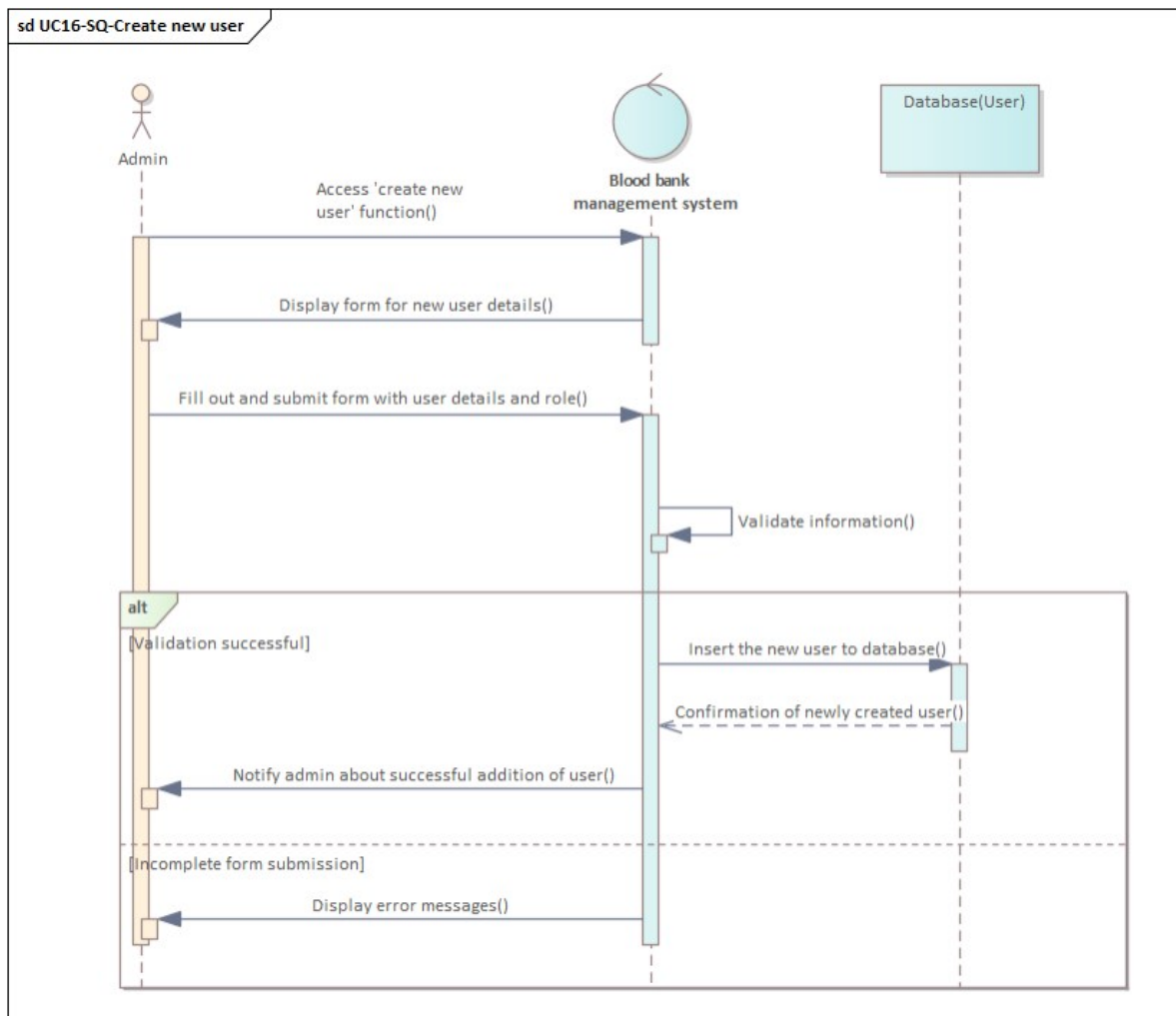


Figure 28. UC16 Create new user sequence diagram.

4 VISUAL BLUEPRINT OF THE APP - HTML PROTOTYPE

When building a system, that it is sure it will contain an interactive user interface, it is important to create the blueprint of the user interface, such a process is not that hard, but it plays a very crucial role when designing and building user interface, making sure that everything aligns together and fit together, such designs and prototypes are called wireframes.

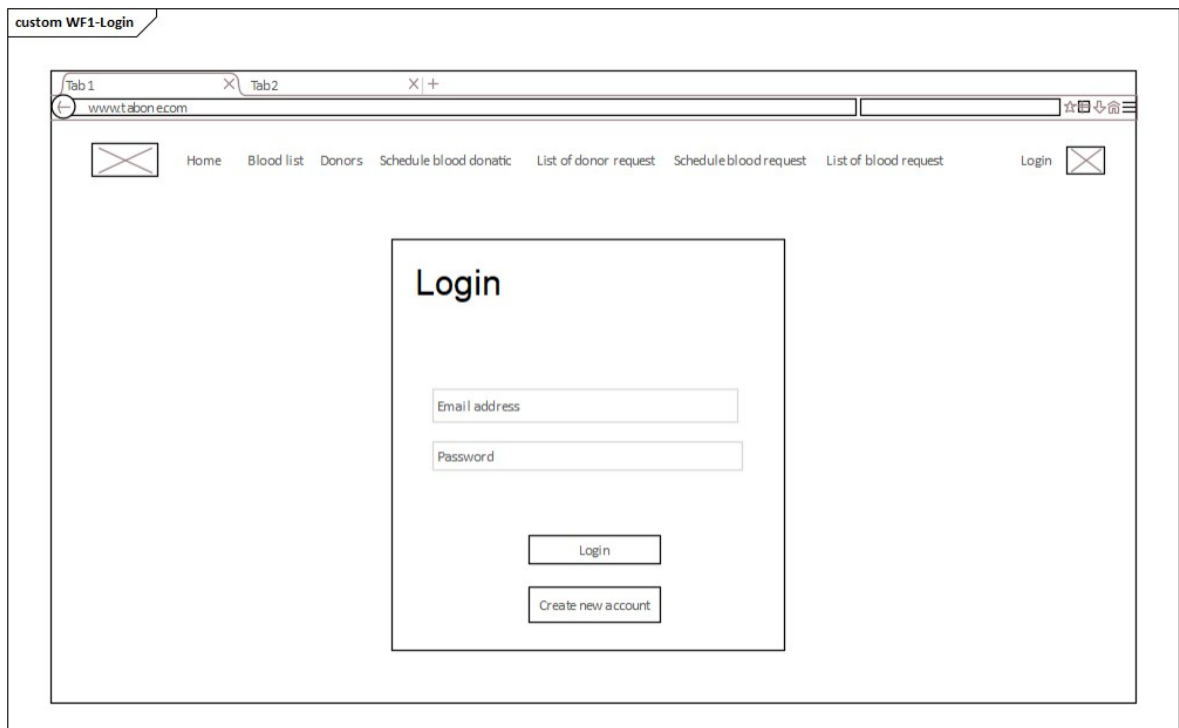


Figure 29. Login page wireframe.

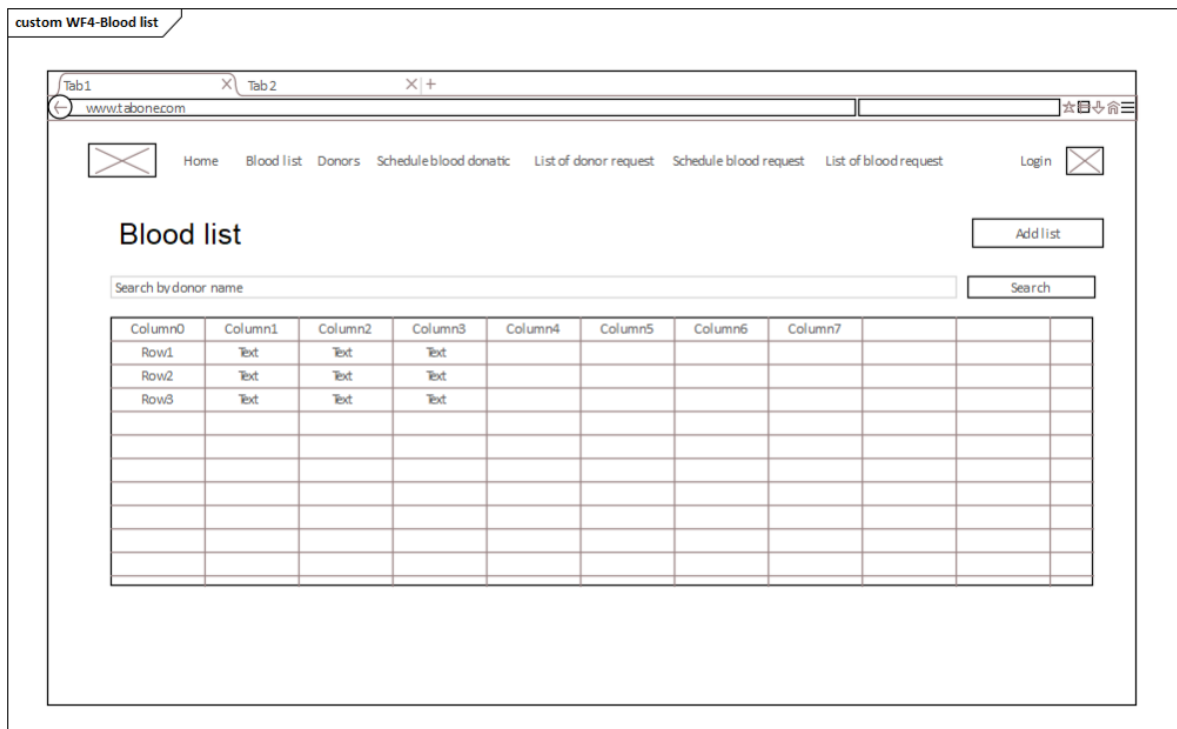


Figure 32. Blood list wireframe

5 WEBSITE PRESENTATION

In the event of a medical emergency, our blood bank maintains a comprehensive blood list to ensure that a variety of blood types are readily available for prompt medical interventions, thereby allowing healthcare professionals to administer life-saving transfusions quickly and efficiently to patients in need as demonstrated in Figure 33.

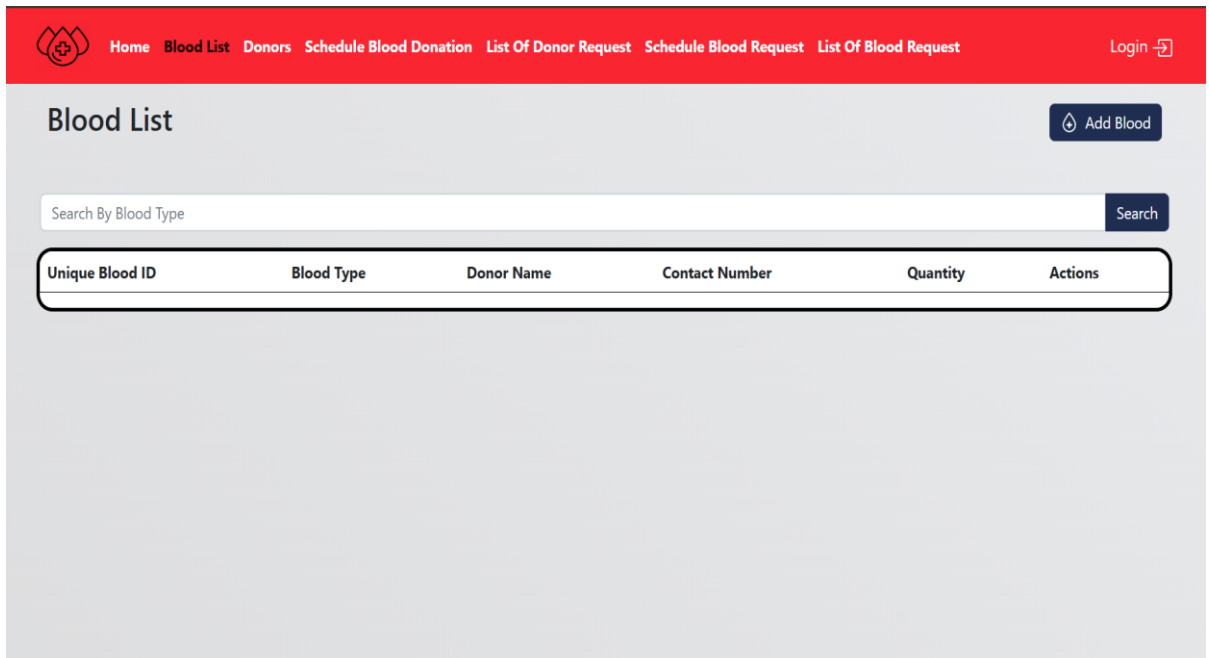


Figure 33 Blood list page.

Donors must be scheduled, thanked, and kept in good standing, all of which may be accomplished with the help of the donor list page as can be seen in Figure 34. It saves lives by guaranteeing a steady supply of good blood for transfusions in emergencies.

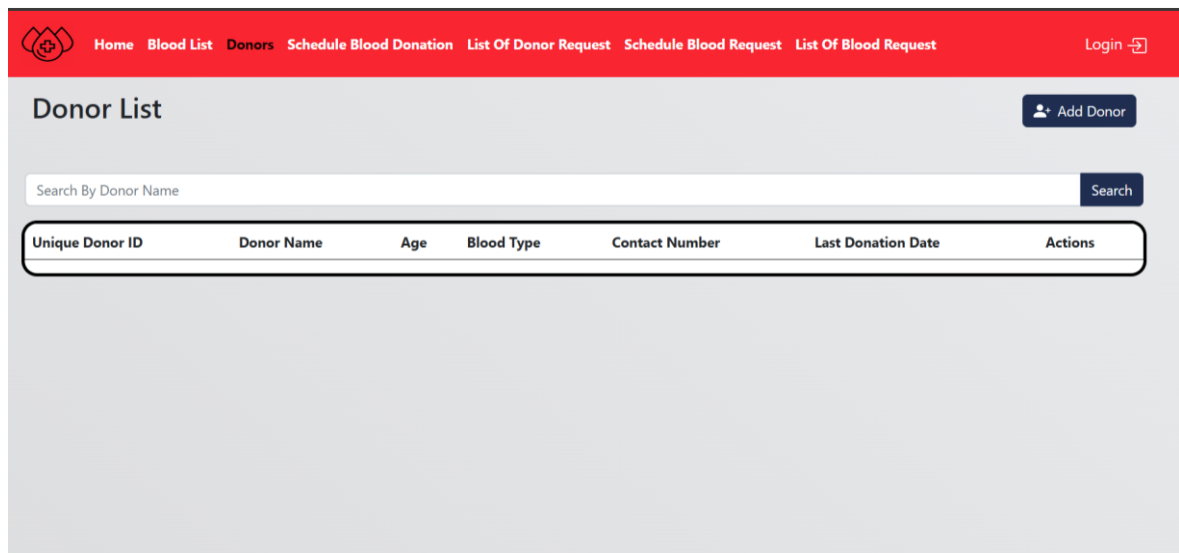


Figure 34 Donor's list page.

The homepage of a blood bank management system is designed to be informative as represented in

Figure 35. It stresses the significance of blood donation, describes the services provided by the blood bank, and urges site visitors to act in favor of this life-saving cause.

Discover a Lifesaving Range of Blood Types

At our blood bank system, we pride ourselves in providing a diverse and comprehensive range of blood types to cater to the urgent needs of patients. With the unwavering support of our dedicated team of donors and stringent testing protocols, we ensure a safe and reliable supply of blood for life-saving transfusions.

Our inventory boasts an extensive selection of blood types, including A, B, AB, and O, ensuring that we can effectively meet the demands of various medical scenarios. Furthermore, we offer both Rh-positive and Rh-negative blood options, ensuring compatibility for patients with specific requirements.

From emergency situations to critical surgeries, our blood bank system is committed to playing a vital role in healthcare by providing the right blood types to those in need. By leveraging advanced technologies and adhering to the highest standards of quality control, we strive to make a difference in the lives of patients and contribute to their well-being.

[Discover Bloods](#)

Become a Hero, Donate Blood

Blood donation is a selfless act that can save lives and make a significant impact on the community. By donating blood, you have the power to make a difference and be a hero for those in need. Your contribution can help patients undergoing surgeries, facing medical emergencies, or battling life-threatening conditions. Every drop of donated blood has the potential to bring hope and healing to individuals and families, providing them with a chance at a brighter future.

At our blood bank, we are committed to making the donation process simple, safe, and convenient for every donor. Our experienced and compassionate team is dedicated to ensuring your comfort and well-being throughout the donation process. We maintain the highest standards of quality and follow strict protocols to ensure the safety of both donors and recipients. Whether you are a first-time donor or a regular donor, your generosity is greatly appreciated and invaluable to our mission of saving lives.

Donating blood is a noble gesture that only takes a short amount of your time but can have a long-lasting impact. By giving just a pint of blood, you can provide hope, healing, and renewed chances at life for those in need. The positive effects of your blood donation can ripple through families and communities, creating a healthier and more vibrant society. Join us in our mission to save lives and be a part of the life-saving journey that starts with a simple act of kindness.

[Save Our Champions](#)

Schedule Your Blood Donation

Thank you for your interest in donating blood and making a difference in the lives of others. Your generous act of donating blood can provide hope, healing, and renewed chances at life for those in need.

Scheduling your blood donation is a simple and convenient step that takes less than 10 minutes. By scheduling your appointment in advance, you allow us to adequately prepare for your donation and optimize our resources to serve the community.

Our convenient online booking system makes it easy to schedule your blood donation appointment. We can select a date and time that best fits your schedule, ensuring a hassle-free experience when you arrive at our donation center. Prior to your appointment, it's important to take care of yourself. Get a good night's sleep, stay hydrated, and have a light meal to ensure you're in the best possible condition to donate. On the day of your donation, please remember to bring a valid identification document for identification purposes.

By scheduling your blood donation and following these guidelines, you play a crucial role in helping us maintain a steady and reliable blood supply for those in need. Your commitment and generosity are deeply appreciated by the patients and the entire community.

[Schedule An Appointment](#)

Helping You Get the Blood You Need

At our blood bank, we are dedicated to helping individuals in need of blood by providing a streamlined process for requesting and receiving the blood they require. We understand the critical importance of timely access to safe and compatible blood, especially in emergency situations and for patients with specific medical conditions.

If you or your loved one is in need of blood, our experienced team is here to assist you. Simply fill out our easy online blood request form, providing essential details such as the blood type required, the quantity needed, and any other relevant information. Our dedicated staff will promptly review your request and do their utmost to fulfill it.

Rest assured that our blood bank follows rigorous testing and screening procedures to ensure the safety and quality of the donated blood. We maintain a diverse inventory of blood types to cater to a wide range of patient needs. Our compassionate team understands the urgency and sensitivity of these situations and is committed to providing the support and care you deserve.

We strive to make the process as smooth as possible, working closely with healthcare professionals to facilitate the efficient delivery of blood to the specified location. Our goal is to contribute to the well-being and recovery of patients by helping them obtain the vital blood they require, ultimately saving lives and making a positive impact.

[Request Blood](#)

Transforming lives through blood donations. Meet experts, schedule appointments, and save lives with ours. Your generosity ensures a steady supply of safe blood products and us to making a lasting impact on communities together. We'll create a brighter future. Be the hero that brings hope and healing.

[Home](#)
[Blood List](#)
[Donors](#)
[Schedule Blood Donation](#)
[Schedule Blood Request](#)

Figure 35 Home page of the website.

Authorized users of a blood bank management system may access the system's capabilities and data through the login section of the website as shown in Figure 36.

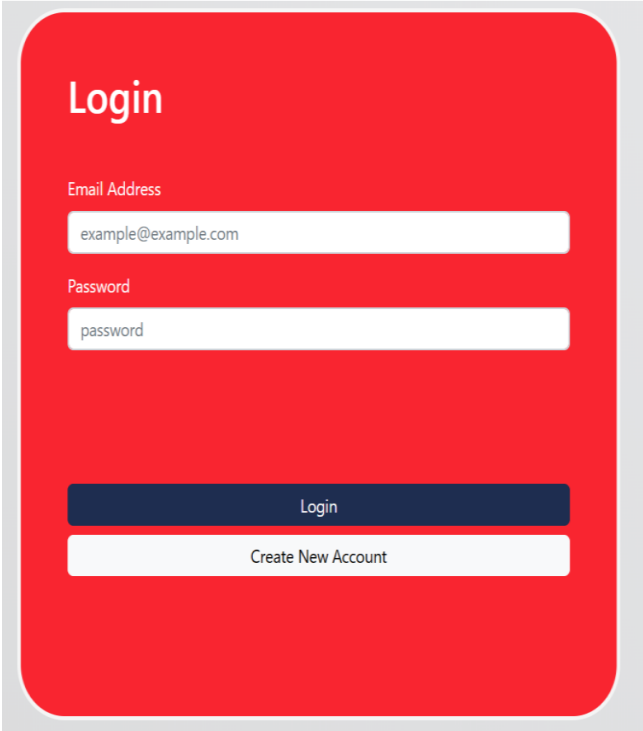

The image shows a login form with a red background and rounded corners. At the top left, the word "Login" is written in white. Below it, there are two input fields. The first is labeled "Email Address" in red text and contains the text "example@example.com". The second is labeled "Password" in red text and contains the text "password". Below these fields are two buttons: a dark blue button labeled "Login" and a white button labeled "Create New Account".

Figure 36 Login section of the website

Using the registration menu, new users may sign up for the system and establish an account for themselves as shown in Figure 37.

A sign-up form with a red background and rounded corners. The title "Sign Up" is in white. Below it are three white input fields with red labels: "Full Name" (containing "John Wick"), "Email Address" (containing "example@example.com"), and "Password" (containing "password"). At the bottom are two buttons: a dark blue "Sign Up" button and a white "Login" button with a dark blue border.

Sign Up

Full Name
John Wick

Email Address
example@example.com

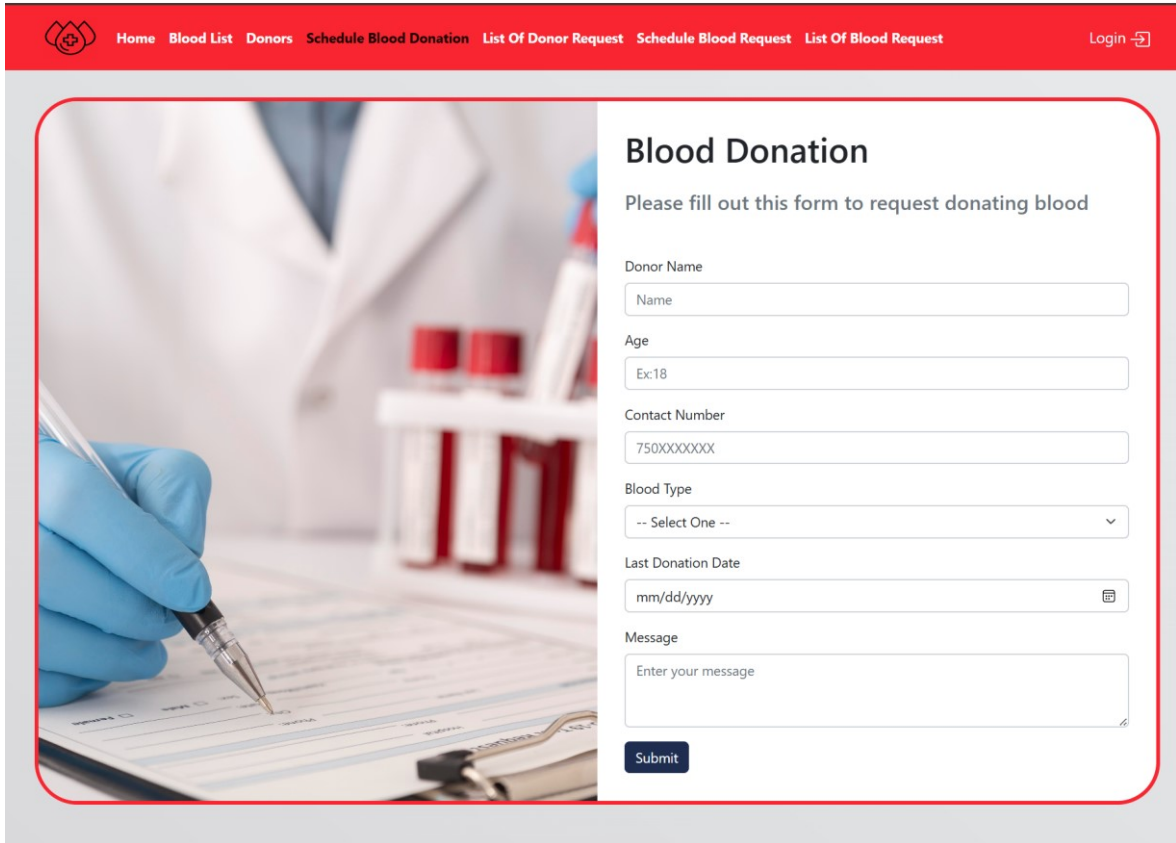
Password
password

Sign Up

Login

Figure 37 Sign up section of the website.

Donors may book their sessions in less time thanks to the scheduling feature included in the blood bank administration system. It increases efficiency, decreases wait times, and helps the blood bank better manage the flow of donors, guaranteeing a steady supply of blood for those in need as observed in Figure 38.



Blood Donation

Please fill out this form to request donating blood

Donor Name

Age

Contact Number

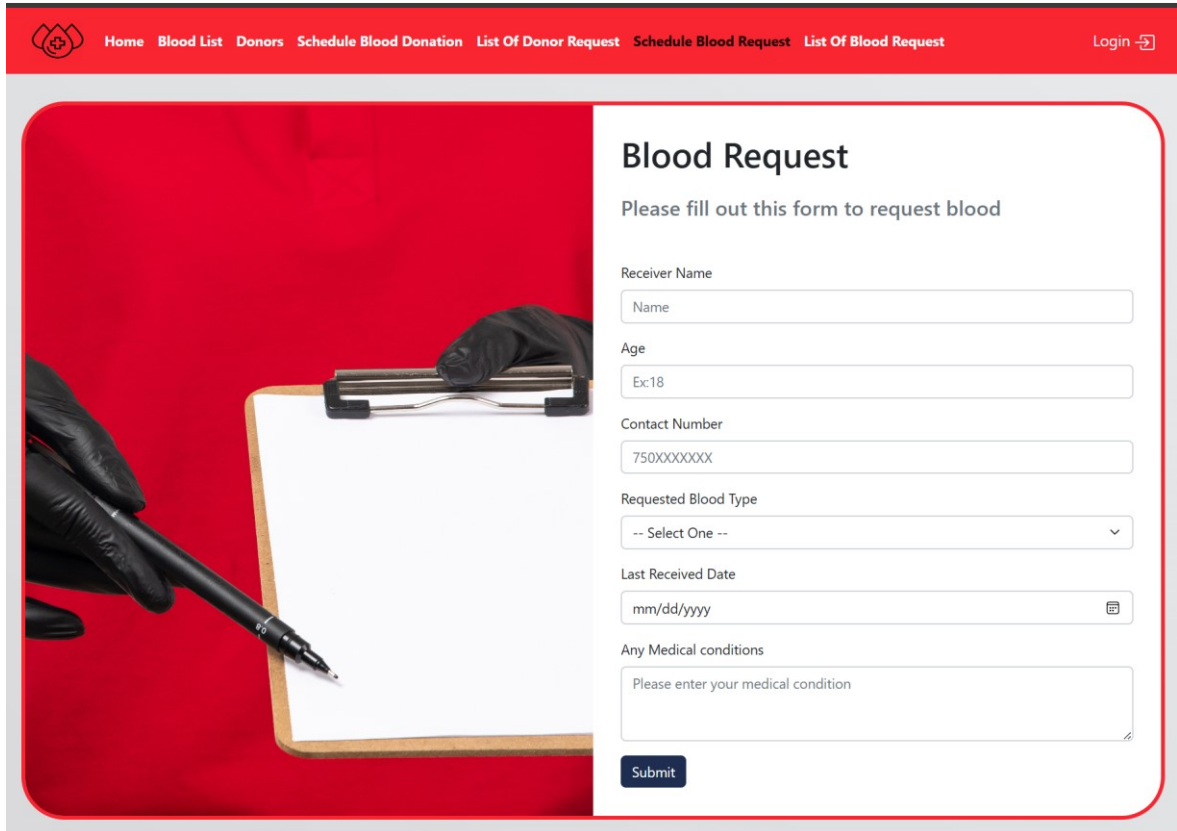
Blood Type

Last Donation Date

Message

Figure 38 Registering for blood donation section.

The blood bank management system accelerates the process of connecting donors, scheduling appointments, and guaranteeing prompt blood supply to patients in need by including a scheduling component for blood requests as shown in Figure 39. It increases productivity, quickens responses, and aids in the blood bank's efficient management of the blood inventory.



Blood Request

Please fill out this form to request blood

Receiver Name

Age

Contact Number

Requested Blood Type

Last Received Date

Any Medical conditions

Figure 39 Registering for requesting blood section.

6 IMPLEMENTATION OF A BLOOD BANK SYSTEM WEBSITE

The goal of this project is to provide a web-based platform for an existing blood bank system. The HTML5, CSS3, and Bootstrap frameworks were used to create a mobile-friendly, aesthetically pleasing website. A responsive layout, a navigation bar, and other web components will need to be designed and implemented using HTML markup and CSS style for this project to be successful. Bootstrap is used to make the site more responsive and ensure that the design is consistent throughout. The goal of this project is to leverage these technologies to build a blood bank management platform that is both user- and donor-friendly.

6.1 Sample of the HTML

The *!DOCTYPE html*> declaration says this is an HTML5 page. The *html lang="en"*> element designates English as the document's language. The *meta charset="UTF-8"*> element indicates that UTF-8 is to be used as the character encoding. To force Internet Explorer to utilize the most up-to-date rendering engine, use the *meta http-equiv="X-UA-Compatible" content="IE=edge"*> tag. For responsive web design to work, the viewport attributes need to be specified using the *meta name="viewport" content="width=device-width, initial-scale=1.0"*> tag. By using the *title>Blood Bank System/title*> tag, we may specify that this page's title will be "Blood Bank System." Using the *link rel="icon" type="image/x-icon" href="/assets/images/logo.png"*> tag. The favicon, a small symbol visible in the tab or bookmarked sites of a web browser, can be identified. The "logo.png" image file may be found in the *"/assets/images/"* folder. External CSS files are referenced in the *link*> tags that come after the favicon declaration. They use both a content delivery network (CDN) and locally stored CSS files. Bootstrap, Bootstrap Icons, and customized styles for the site's navigation and primary content are all included in these CSS files as can be seen in Figure 40.

```

<!DOCTYPE html>
<html lang="en">

<head>
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Blood Bank System</title>
  <link rel="icon" type="image/x-icon" href="/assets/images/logo.png">
  <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.3/dist/css/bootstrap.min.css" rel="stylesheet"
    integrity="sha384-rbsA2VBKQhggwzxH7pPCaAqO46MgnOM80zW1RWuH61DGLWZJEdK2Kadq2F9CUG65" crossorigin="anonymous">
  <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/bootstrap-icons@1.10.2/font/bootstrap-icons.css">
  <link rel="stylesheet" href="/assets/css/navbar.css">
  <link rel="stylesheet" href="/assets/css/index.css">
</head>

```

Figure 40 Structure and Styling.

The HTML document's viewable content is located inside the body section, which is denoted by the *body*> element. The *nav*> element specifies the location of the site's menu at the top of the page. To have the navigation bar stay to the top of the page and expand on bigger displays, use the *class="navbar sticky-top navbar-expand-lg"* property. Using the *div class="container"> tag*, a container element may be generated around the navbar's contents. The navigation bar's branding/logo portion is denoted by a *class="navbar-brand" href="/index.html"> element*. The logo is shown inside an image element, and an anchor tag links to the *"index.html"* page. A mobile device's navigation bar's on/off switch is denoted by the button *class="navbar-toggler"> tag*. It's for making the menu bigger or smaller on mobile devices. The *div class="collapse navbar-collapse d-lg-flex" the contents of the collapsible navigation bar*, including the navigation menu, are included in the *div align="content-between align-items-center" id="navbarNav"> tag*. The navigational elements are shown in an unordered list denoted by the *ul class="navbar-nav"> tag*. There is one *li class="nav-item"> tag* for each menu item. Links in a menu are indicated by a *class="nav-link"> tags*. 3The href property of a link indicates the target URL. The login link may be found in the menu as the last a *class="nav-link__login"> element*. There's a graphical icon and a descriptive label as observed in Figure 41.

```

<body>
  <nav class="navbar sticky-top navbar-expand-lg">
    <div class="container">
      <a class="navbar-brand" href="./index.html">
        
      </a>
      <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-target="#navbarNav"
        aria-controls="navbarNav" aria-expanded="false" aria-label="Toggle navigation">
        <span class="navbar-toggler-icon"></span>
      </button>
      <div class="collapse navbar-collapse d-lg-flex justify-content-between align-items-center" id="navbarNav">
        <ul class="navbar-nav">
          <li class="nav-item">
            <a class="nav-link active" aria-current="page" href="./index.html">Home</a>
          </li>
          <li class="nav-item">
            <a class="nav-link" href="./blood-list.html">Blood List</a>
          </li>
          <li class="nav-item">
            <a class="nav-link" href="./donors.html">Donors</a>
          </li>
          <li class="nav-item">
            <a class="nav-link" href="./schedule-blood-donation.html">Schedule Blood Donation</a>
          </li>
          <li class="nav-item">
            <a class="nav-link" href="./Listofdonorrequest.html">List Of Donor Request</a>
          </li>
          <li class="nav-item">
            <a class="nav-link" href="./schedule-blood-request.html">Schedule Blood Request</a>
          </li>
          <li class="nav-item">
            <a class="nav-link" href="./Listofbloodrequest.html">List Of Blood Request</a>
          </li>
        </ul>
        <a class="nav-link__login d-flex align-items-center gap-1 mb-2 mt-2 mb-lg-0 mt-lg-0" href="login.html">
          <span id="nav-login-text">Login</span>
          
        </a>
      </div>
    </div>
  </nav>

```

Figure 41 Navigation Bar Implementation

The primary content of a web page is denoted by the main `class="main content">` element. A section dealing with blood quantity or quality is denoted by the section `class="blood-section mt-5">` tag. It has a 5-point margin at the top. By using the `div class="row d-flex">` element, a row may be created to separate the text into columns. On medium-sized screens, the `div class="image-container col-md-6 flex-grow-1 align-self-center">` tag produces a column with a width of 6 units. An illustration from the "blood" chapter is included. The blood section picture is shown using the `img src="./assets/images/blood-section.jpg" alt="blood section" class="img-fluid">` tag. The image file "blood-section.jpg" from the `./assets/images/` folder is referred to and a fluid responsiveness class is used. The second `div class="col-md-6">` tag produces a column 6 units wide on screens of medium size. The content of the blood section is enclosed in the `div class="section-content">` element. To

emphasize its importance, the heading "Discover a Lifesaving Range of Blood Types" has been formatted using the `h1>` element. Paragraphs detailing the blood bank system, its wide variety of blood types, its commitment to safety, and its devotion to giving patients with the appropriate blood types may be found inside the following `p>` tags. A button titled "Discover Bloods" is generated by the closing a `class="btn btn-lg btn-primary mt-4" href="/blood-list.html">` tag. The "blood-list.html" page is the target of the link as illustrated in Figure 42.

```
<main class="main-content">
  <section class="blood-section mt-5">
    <div class="row d-flex">
      <div class="image-container col-md-6 flex-grow-1 align-self-center">
        
      </div>
      <div class="col-md-6">
        <div class="section-content">
          <h1>Discover a Lifesaving Range of Blood Types</h1>
          <p>At our blood bank system, we pride ourselves in providing a diverse and
            comprehensive range of blood types to cater to the urgent needs of patients. </p>
          <p>From emergency situations to critical surgeries</p>
          <a class="btn btn-lg btn-primary mt-4" href="/blood-list.html">Discover Bloods</a>
        </div>
      </div>
    </div>
  </section>
```

Figure 42 Blood Bank System Overview.

To indicate a section about blood volume, use the section `class="blood-section mt-5">` tag with a *margin-top of 5* units. To create columns of content, use the `div class="row d-flex">` element to generate a row. On medium-sized screens, the `div class="col-md-6">` tag generates a column that is 6 units wide. The content of the blood section is enclosed in the `div class="section-content">` element. "Become a Hero, Donate Blood" is the heading that is shown in the `h1>` element. Paragraphs describing the significance of blood donation, the dedication of the blood bank to providing a secure and hassle-free donation procedure, and the benefits to donors are included below. Click the "See Our Donors" button generated by the closing a `class="btn btn-lg btn-primary mt-4 mb-3" href="/donors.html">` tag. The "donors.html" page is linked to. For a second column with a width of 6 units on medium-sized screens, use the `div class="image-container col-md-6 flex-grow-1 align-self-center">` tag. For example, to show a picture about blood donation, use the `img src="/assets/images/donate-section.jpg" alt="blood section" class="img-fluid">` tag. It uses a fluid responsiveness

class and links to the "donate-section.jpg" image file in the `./assets/images/` folder as shown in Figure 43.

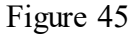
```
<section class="blood-section mt-5">
  <div class="row d-flex">
    <div class="image-container col-md-6 flex-grow-1 align-self-center">
      
    </div>
    <div class="col-md-6">
      <div class="section-content">
        <h1>Discover a Lifesaving Range of Blood Types</h1>
        <p>At our blood bank system, we pride ourselves in providing a diverse and comprehensive range of blood types to cater to the urgent needs of patients. With the unwavering support of our dedicated team of donors and stringent testing protocols, we ensure a safe and reliable supply of blood for life-saving transfusions.</p>
        <p>Our inventory boasts an extensive selection of blood types, including A, B, AB, and O, ensuring that we can effectively meet the demands of various medical scenarios. Furthermore, we offer both Rh-positive and Rh-negative blood options, ensuring compatibility for patients with specific requirements.</p>
        <p>From emergency situations to critical surgeries, our blood bank system is committed to playing a vital role in healthcare by providing the right blood types to those in need. By leveraging advanced technologies and adhering to the highest standards of quality control, we strive to make a difference in the lives of patients and contribute to their well-being.</p>
        <a class="btn btn-lg btn-primary mt-4" href="./blood-list.html">Discover Bloods</a>
      </div>
    </div>
  </div>
</section>
```

Figure 43 Blood Donation Section.

The footer of the page is indicated by the footer `class="footer p-3 pb-0">` element, which includes padding and margin classes. By using the `div class="container">` tag, a container element is generated to hold the footer. Using the `div class="row">` and `div class="col-md-4">` tags, we've separated the footer into two columns. The blood donation effort is described in a paragraph and the logo is included in an anchor tag in the first column (`col-md-4`). Two additional columns, `col-md-8` and `col-md-4` are included inside the second main column. An unordered list (`ul class="navbar-nav">`) depicting Home, Blood List, Donors, Schedule Blood Donation, and Schedule Blood Request may be found in the nesting column (`col-md-8`) of the table. A login anchor tag and a row of Facebook, Instagram, and Twitter icons are enclosed in anchor tags in the nested column (`col-md-4`). The social media icons are laid out in a column using a flex-column class and a justified text style as can be observed in Figure 44

Figure 44 Website Footer Navigation and Social Icons.

6.2 Sample of the CSS

Styles for various page components are included in the given CSS code. For the main content section to fill the whole viewport, the *'main-content'* class must provide a minimum height. The *'table-container'* class makes a container element's borders and corners black and its background white. Classes like *'table'* and *'table tr'* are used to do things like add padding to table cells and change the color of table row borders, respectively. Border widths for form controls and select elements are specified by the *'form-control'* and *'form-select'* classes. When a form control or select element receives attention, it takes on the visual style set for it in its focus style sheet. Background colors, border colors, and text colors may all be altered for main buttons using the *'btn-primary'* class, with extra styles provided for the hover, active, and focus-visible states. The appearance and behavior of a web page's components are heavily influenced by the CSS styles used to format them as shown in  Figure 45

```

.main-content {
  min-height: 100vh;}
.table-container {
  border:4px solid ■black;
  border-radius: 18px;
  background-color: □white;}
.table {
  padding-top: 4px;
  padding-left: 4px;}
.table tr {
  border-color: ■black;}
.form-control, .form-select {
  border-width: 2px; }
.form-control:focus{
  border-color: var(--bs-navbar-active-color);
  box-shadow: 0.5px 0.5px 2px var(--bs-navbar-active-color);}
.form-select:focus, .form-select:focus-visible {
  border-color: ■black;
  box-shadow: 0.5px 0.5px 2px ■black; }
.btn-primary {
  background-color: var(--primary-text);
  border-color: var(--primary-text);
  color: var(--secondary-background); }
.btn-primary:hover {
  background-color: ■black;
  border-color: ■black; }
.btn-primary:active {
  background-color: ■rgba(0, 0, 0, 0.9) !important;
  border-color: ■rgba(0, 0, 0, 0.9) !important;
  box-shadow: none; }
.btn-primary:focus-visible {
  background-color: ■rgba(0, 0, 0, 0.9);
  border-color: ■rgba(0, 0, 0, 0.9);
  box-shadow: 0 0 0 0.25rem ■rgba(0, 0, 0, 0.349); }

```

Figure 45 CSS Styling for Form and Table Elements.

6.3 Sample of the JavaScript

In the Blood Bank System application, efficient user interaction is paramount for managing donor-related activities effectively. To facilitate seamless interaction with the system's interface, we've meticulously selected key elements for manipulation. Each selector targets essential components within the application, enabling precise control and interaction with donor data as observed in Figure 27..

Selectors for User Interaction:

- Table Body Selector (tableBody):

This selector identifies the table body element where donor data is displayed. It serves as the canvas for rendering donor information in a structured format.

- Add Donor Button Selector (addDonorBtn):

Targeting the add donor button, this selector allows users to initiate the process of adding new donors to the system. It acts as a catalyst for expanding the donor database seamlessly.

- Donor Form Selector (donorForm):

With this selector, we pinpoint the donor form element, where users input essential details of new donors. It serves as the interface for capturing and submitting donor information efficiently.

- Donor Modal Selector (donorModal):

Targeting the donor modal element, this selector facilitates user interaction within the modal interface. It ensures a smooth experience when editing or adding donor information.

- Modal Title Selector (modalTitle):

This selector focuses on the modal title element, providing users with context and guidance within the donor modal interface. It enhances clarity and usability during donor data manipulation.

- Blood Type Select Selector (bloodTypeSelect):

Identifying the blood type select element, this selector enables users to specify the blood type of donors conveniently. It streamlines the process of capturing vital donor information.

- Donor Name Input Selector (donorNameInput):

Targeting the donor name input element, this selector allows users to input the names of donors accurately. It ensures precision and consistency in donor data entry.

- Contact Number Input Selector (contactNumberInput):

With this selector, users can input contact numbers for donors seamlessly. It facilitates communication and outreach efforts within the donor management process.

- Age Input Selector (ageInput):

Targeting the age input element, this selector enables users to input the ages of donors accurately. It ensures data accuracy and consistency in donor profiles.

- Last Donation Date Input Selector (lastDonationDateInput):

This selector focuses on the last donation date input element, allowing users to input the dates of donors' last blood donations. It ensures up-to-date and relevant donor information.

- Search Form Selector (searchForm):

Identifying the search form element, this selector empowers users to search for specific donors efficiently. It enhances accessibility and navigation within the application.

```
// SELECTORS
const tableBody = document.querySelector("#table-body");
const addDonorBtn = document.querySelector("#add-donor-btn");
const donorForm = document.querySelector("#donor-form");
const donorModal = document.querySelector("#donor-modal");
const modalTitle = document.querySelector(".modal-title");
const bloodTypeSelect = document.querySelector("#blood-type");
const donorNameInput = document.querySelector("#donor-name");
const contactNumberInput = document.querySelector("#contact-number");
const ageInput = document.querySelector("#age");
const lastDonationDateInput = document.querySelector("#last-donation-date");
const searchForm = document.querySelector("#search-form");
```

Figure 27 DOM Element Selectors.

To ensure seamless data management within the Blood Bank System application, we've implemented a robust mechanism for handling donor data. Upon initialization, the system intelligently fetches donor data from the local storage. If no data is found, an empty array is utilized, guaranteeing a smooth user experience regardless of the initial state as observed in Figure 28.

Functions for Data Manipulation:

- Format Date to YYYY-MM-DD:

This function ensures uniformity in date representation by formatting date objects to the YYYY-MM-DD format. It optimizes data consistency and readability across the application.

- Save Donor Data to Local Storage:

To preserve user input and maintain data integrity, this function efficiently stores donor data in the local storage. By serializing the data to JSON format, it ensures compatibility and accessibility.

- Render Table Rows Based on Donor Data:

Leveraging the fetched donor data, this function dynamically generates table rows, presenting essential donor information in a structured format. Each row contains details such as donor ID, name, age, blood type, contact number, and last donation date, facilitating comprehensive data visualization.

```

// Fetch donor data from local storage or use an empty array if it doesn't exist
let donorData = JSON.parse(localStorage.getItem("donorData")) || [];

✓ // FUNCTIONS
// Formats a date object to YYYY-MM-DD format
✓ function formatDateToYYYYMMDD(date) {
  const year = date.getFullYear();
  const month = String(date.getMonth() + 1).padStart(2, "0");
  const day = String(date.getDate()).padStart(2, "0");
  return `${year}-${month}-${day}`;
}
// Saves donor data to local storage
✓ function saveDonorDataToLocalStorage() {
  localStorage.setItem("donorData", JSON.stringify(donorData));
}
// Renders table rows based on donor data
✓ function renderTableRows() {
  let tableRows = "";
  ✓ donorData.forEach((donor) => {
  ✓   tableRows += `
    <tr id="row" data-row-number="${donor.id}">
      <th scope="row">${donor.id}</th>
      <td>${donor.donorName}</td>
      <td>${donor.age}</td>
      <td>${donor.bloodType}</td>
      <td>${donor.contactNumber}</td>
      <td>${new Date(donor.lastDonationDate).toLocaleDateString("en-US")}</td>
      <td>
        <div class="d-flex gap-1">
          <button id="edit-donor-btn" data-donor-id="${donor.id}" type="button" class="btn btn-primary btn-sm">
            <button id="delete-donor-btn" data-donor-id="${donor.id}" type="button" class="btn btn-danger btn-sm">
        </div>
      </td>
    </tr>
  `;
  });
  tableBody.innerHTML = tableRows;
}

```

Figure 28 Donor Data Management Module.

The submission of donor form data is a pivotal aspect of the Blood Bank System's functionality, ensuring the seamless addition or modification of donor information. This function orchestrates the process, validating user input and executing appropriate actions to maintain data accuracy and integrity as observed in Figure 28..

- Submission of Donor Form Data:

Upon form submission, this function intercepts the default behavior, preventing the page from reloading and allowing for dynamic data processing. It extracts relevant information from the form fields, including blood type, donor name, contact number, age, and last donation date, facilitating comprehensive donor data capture.

- Data Validation and Management:

Before proceeding with data manipulation, the function rigorously validates the input to ensure completeness and accuracy. If all required fields are filled and a valid last donation date is provided, the function determines whether to add a new donor entry or update an existing one based on the presence of a donor ID in the form.

- Updating or Adding Donor Information:

In case of an existing donor ID, the function identifies the corresponding donor entry within the data array, updates the relevant fields with the new information, and triggers the storage of updated data in the local storage. Conversely, if no donor ID is found, indicating a new donor entry, the function appends the new donor information to the data array and saves it to the local storage.

```

// Submits donor form data
function submitDonor(e) {
  e.preventDefault();
  const formData = new FormData(e.target);
  const bloodType = formData.get("blood-type");
  const donorName = formData.get("donor-name");
  const contactNumber = formData.get("contact-number");
  const age = formData.get("age");
  const lastDonationDate = new Date(formData.get("last-donation-date"));

  if (bloodType !== "" && donorName !== "" && contactNumber !== "" && age !== "" && lastDonationDate) {
    // EDIT Donor IF ID EXISTS ON FORM
    if (e.target.dataset.donorId) {
      const donorIndex = donorData.findIndex((donor) => donor.id === parseInt(e.target.dataset.donorId));
      if (donorIndex !== -1) {
        donorData[donorIndex] = {
          id: parseInt(e.target.dataset.donorId),
          donorName,
          age,
          bloodType,
          contactNumber,
          lastDonationDate,
        };
        saveDonorDataToLocalStorage();
        renderTableRows();
      }
    } else {
      // ADD Donor
      const lastDonorId = donorData.length > 0 ? donorData[donorData.length - 1].id : 0;
      const newDonor = {
        id: lastDonorId + 1,
        donorName,
        age,
        bloodType,
        contactNumber,
        lastDonationDate,
      };
      donorData.push(newDonor);
      saveDonorDataToLocalStorage();
      renderTableRows();
    }

    const modal = new bootstrap.Modal(donorModal);
    modal.hide();
    resetForm();
  }
}

```

Figure 29 Figure 1: submitDonor() Function Overview.

Efficient management of donor entries and user interactions is critical to the functionality and usability of the Blood Bank System application. The following functions play a pivotal role in facilitating smooth user experiences, from deleting donor entries to filtering and rendering table rows based on search input as illustrated in Figure 30..

- Deletion of Donor Entry:

This function enables users to delete donor entries from the system with confidence. Upon triggering the deletion action, users are prompted with a confirmation dialog to ensure the accuracy of their decision. If confirmed, the function removes the specified donor entry from the data array, updates the local storage with the modified data, and refreshes the table to reflect the changes.

- Resetting the Donor Form:

To streamline the process of adding new donor entries or editing existing ones, this function resets the donor form to its default state. It clears all form fields, resets the modal title to "Add Donor," removes any existing donor ID attribute, and prepares the form for a fresh entry.

- Performing Actions Based on User Interaction:

This function serves as the central hub for handling user interactions within the donor management interface. It dynamically responds to user actions such as editing or deleting donor entries, ensuring a seamless and intuitive user experience. Whether editing donor information or initiating deletion, this function executes the appropriate actions based on user intent.

- Filtering and Rendering Table Rows:

In response to user input in the search field, this function filters donor entries based on the provided search value. By converting the search input to lowercase and comparing it with donor names, the function generates a filtered result set. Subsequently, it renders table rows based on the filtered data, providing users with relevant and targeted information.

```

// Deletes a donor entry
function deleteDonor(id) {
  const isConfirmed = confirm("Are You Sure?");
  if (isConfirmed) {
    donorData = donorData.filter((donor) => donor.id !== id);
    saveDonorDataToLocalStorage();
    renderTableRows();
  }
}

// Resets the donor form
function resetForm() {
  modalTitle.textContent = "Add Donor";
  donorForm.removeAttribute("data-donor-id");
  bloodTypeSelect.value = "";
  donorNameInput.value = "";
  contactNumberInput.value = "";
  ageInput.value = "";
  lastDonationDateInput.value = "";
}

// Performs actions based on user interaction
function performActions(e) {
  if (e.target.dataset.donorId !== undefined) {
    if (e.target.id === "edit-donor-btn" || e.target.id === "edit-donor-icon") {
      const donorId = parseInt(e.target.dataset.donorId);
      const selectedDonor = donorData.find((donor) => donor.id === donorId);
      if (selectedDonor) {
        modalTitle.textContent = "Edit Donor";
        donorNameInput.value = selectedDonor.donorName;
        ageInput.value = selectedDonor.age;
        bloodTypeSelect.value = selectedDonor.bloodType;
        contactNumberInput.value = selectedDonor.contactNumber;
        lastDonationDateInput.value = formatDateToYYYYMMDD(selectedDonor.lastDonationDate);
        donorForm.setAttribute("data-donor-id", donorId);
      }
    } else {
      deleteDonor(parseInt(e.target.dataset.donorId));
    }
  }
}

// Filters and renders table rows based on search input
function filterAndRenderTableRows(e) {
  e.preventDefault();
  const searchValue = document.getElementById("search").value.trim().toLowerCase();
  if (searchValue === "") {
    renderTableRows();
    return;
  }
  const result = donorData.filter((donor) => donor.donorName.toLowerCase().includes(searchValue));
  renderTableRows(result);
}

```

Figure 30 Donor Management Functions Overview.

Efficient execution and user interaction lie at the core of the Blood Bank System's functionality. Through meticulously designed event listeners and an initialization function, the system ensures smooth operation and seamless user engagement as can be observed in Figure 31.

Execution and Event Listeners:

- Renders Initial Table Rows:

The system initiates by rendering the initial set of table rows, providing users with immediate access to donor information upon loading the application. This proactive approach enhances user experience and expedites data interaction.

- Event Listeners for User Interactions:

The system incorporates event listeners to capture user interactions effectively. By responding to actions such as adding a donor, submitting a donor form, searching for specific donors, or performing actions on donor entries, the system facilitates intuitive user engagement and data manipulation.

Initialization Function:

- Initialization Function:

Central to the system's functionality, the initialization function sets the stage for seamless operation. By orchestrating the execution of essential tasks and ensuring the readiness of key components, this function lays the groundwork for a responsive and user-centric experience.

```
// EXECUTION
// Renders initial table rows
renderTableRows();
// Event listeners for user interactions
addDonorBtn.addEventListener("click", resetForm);
donorForm.addEventListener("submit", submitDonor);
searchForm.addEventListener("submit", filterAndRenderTableRows);
tableBody.addEventListener("click", performActions);

// Initialization function
initializePage();
```

Figure 31 Execution and User Interaction Workflow.

7 USER GUIDE FOR APPLICATION FEATURES AND FUNCTIONALITIES

Welcome to the User Guide for Application Features and Functionalities (Control User) chapter of the Blood Bank System application! Here, we provide you with a comprehensive guide to navigating through the various features and functionalities designed to empower you in managing blood donation activities effectively.

The `index.html` serves as your gateway to a user-friendly interface meticulously crafted to offer clear instructions and seamless access to essential actions within the Blood Bank System. From navigating different sections to scheduling blood donations or requesting blood, every feature is tailored to enhance your experience.

Features and Functionalities:

1. Navigation Bar:

- Allows seamless access to different sections of the application and links to Home, Blood List, Donors, Schedule Blood Donation, List of Donor Requests, Schedule Blood Request, and List of Blood Requests also provides a "Login" button for user authentication.

2. Banner:

- Greets users with personalized messages and Displays the next donation date to personalize the experience.

3. Blood Section:

- Offers insights into the blood bank's services, show cases the variety of blood types available, Encourages exploration of available blood types with a call-to-action button.

4. Donation Section:

- Motivates users to become blood donors. Emphasizes the significance of blood donation in saving lives and offers a call-to-action button to explore existing donors.

5. Schedule Blood Donation Section:

- Guides users through scheduling their blood donation appointments, Stresses the importance of scheduling for a streamlined donation process and encourages users to schedule an appointment with a call-to-action button.

6. Blood Request Section:

- Simplifies the process of requesting blood, educates users about the blood request process and its importance, provides a call-to-action button to request blood.

7. Footer:

- Reinforces the application's mission and purpose, offers quick links to essential pages like Home, Blood List, Donors, Schedule Blood Donation, and Schedule Blood Request, and includes social media icons for engagement and connectivity.

Common Tasks:

1. Navigating the Application:

- Users can effortlessly explore different sections using the navigation bar, links provide direct access to pages like Blood List, Donors, and various scheduling options.

2. Discovering Blood Types:

- Users can learn about available blood types and their significance in the "Blood Section." Moreover, exploration is facilitated through the "Discover Bloods" button.

3. Donating Blood:

- Users interested in donating blood can find information in the "Donation Section."

Moreover, they can explore existing donors by clicking the "See Our Donors" button.

4. Scheduling Blood Donation:

- Users willing to donate blood can schedule appointments via the "Schedule Blood Donation Section." and scheduling is encouraged through the "Schedule an Appointment" button.

5. Requesting Blood:

- Users in need of blood can request it through the "Blood Request Section." And they can initiate requests by clicking the "Request Blood" button.

2. LOGIN.HTML

Experience a straightforward interface for user authentication or new account registration with the login.html section. Whether you're logging in to access your account or creating a new one, our navigation bar ensures easy access, while the login form prioritizes security and ease of use.

Features and Functionalities:

1. Navigation Bar:

- Consistently present across the application for easy navigation and provides links to Home, Blood List, Donors, Schedule Blood Donation, List of Donor Requests, Schedule Blood Request, and List of Blood Requests, it also Includes a "Login" button with a login icon for authentication.

2. Login Form:

- Allows users to input their email address and password for authentication, email and password fields include appropriate placeholders, email field uses type "email" for validation, password field masks input for security and requires both fields for form submission.

3. Buttons:

- Login Button: Submits the form for user authentication, redirecting to the homepage (`.index.html`) upon success, and it create New Account Button: Redirects users to the signup page (`.signup.html`) for account registration.

Common Tasks:

1. Logging In:

- Users input their credentials into the respective fields, after filling in, they click the "Login" button to submit the form, upon successful authentication, users are redirected to the homepage (`.index.html`).

2. Creating a New Account:

- Users without an account can click the "Create New Account" button, this action redirects them to the signup page (`.signup.html`) for registration.

User Interaction:

1. Input Validation:

- Email field ensures valid email format, and both fields (email and password) are required, indicated by the "required" attribute.

2. Error Handling:

- Implementation of error messages for incorrect credentials or server errors is recommended for smoother user experience.

3. Responsive Design:

- Login form adapts to various screen sizes for usability across devices.

3. SIGNUP.HTML

Registering for a new account is made simple with the Signup.html section. Providing your information to create a new account is a breeze, with placeholders guiding you through the process. Responsive design ensures a seamless experience across devices.

Features and Functionalities:

1. Navigation Bar:

- Consistent navigation across the application, it links to Home, Blood List, Donors, Schedule Blood Donation, and Schedule Blood Request and includes a "Login" button for accessing the login page.

2. Sign Up Form:

- Enables users to register for a new account by providing full name, email address, and password, fields include placeholders for full name, email, and password, email field uses type "email" for validation, password field masks input for security and it requires all fields for form submission.

3. Buttons:

- Sign Up Button: Submits the form for user registration, redirecting to the homepage ('./index.html').
- Login Button: Redirects users to the login page ('login.html') if they have an account.

Common Tasks:**1. Registering a New Account:**

- Users input their information into the form fields, after filling in, they click the "Sign Up" button to submit the form and create a new account.

2. Logging In:

- Users with an existing account can click the "Login" button and this action directs them to the login page ('login.html') for authentication.

User Interaction:**1. Input Validation:**

- Email field ensures valid email format and all fields (full name, email, and password) are required.

2. Responsive Design:

- Sign-up form adjusts to various screen sizes for enhanced usability.

4. BLOOD-LIST.HTML

Manage blood donations effortlessly with the Blood-List.html section. From viewing a detailed list of blood donations to adding new entries or searching for specific blood types, this feature-rich interface empowers you to stay organized and informed.

Features and Functionalities:**1. Navigation Bar:**

- Offers consistent navigation throughout the application and provides links to Home, Blood List, Donors, Schedule Blood Donation, and Schedule Blood Request, it also includes a "Login" button for user authentication.

2. Blood List Display:

- Presents a comprehensive list of blood donations with details such as unique blood ID, blood type, donor name, contact number, and quantity, and displays blood donations in a tabular format for easy comprehension.

3. Add Blood Button:

- Allows users to add a new blood donation entry with ease, and clicking the button opens a modal window for inputting details like blood type, donor name, contact number, and quantity.

4. Search Functionality:

- Empowers users to search for specific blood types using the search input field, and after entering a query, users can click the "Search" button to filter the blood list accordingly.

5. Modal for Adding Blood Donation:

- Facilitates the process of adding a new blood donation entry and offers fields for blood type (dropdown), donor name, contact number, and quantity within a modal window.

Common Tasks:**1. Viewing Blood Donations:**

- Users can easily browse through the list of blood donations, and the tabular format ensures clear presentation of donation details.

2. Adding a New Blood Donation:

- Users can initiate the addition of a new blood donation entry by clicking the "Add Blood" button and completing the required details in the modal window.

3. Searching for Blood Types:

- The search functionality enables users to quickly locate specific blood types within the list.

User Interaction:

1. Input Validation:

- The modal form requires input for fields like blood type, donor name, contact number, and quantity, and the blood type field offers a dropdown selection to ensure accurate input, also the contact number field is validated as type "tel" for data integrity.

2. Responsive Design:

- The layout adapts seamlessly to various screen sizes, enhancing accessibility.

5. DONORS.HTML

Discover the power of efficient blood donor management with the Donors.html section. From adding new donors to browsing through a detailed list or searching for specific individuals, this feature ensures streamlined interaction with blood donors.

Features and Functionalities:

1. Navigation Bar:

- Consistent navigation across the application, offering links to Home, Blood List, Donors (active), Schedule Blood Donation, and Schedule Blood Request, it also includes a "Login" button for authentication.

2. Donor List Display:

- Presents a detailed list of blood donors featuring unique donor ID, donor name, age, blood type, contact number, and last donation date and organized in a tabular format for clarity and ease of browsing.

3. Add Donor Button:

- Enables users to add a new donor entry seamlessly and the clicking the button opens a modal window for inputting donor details like name, age, contact number, blood type, and last donation date.

4. Search Functionality:

- Empowers users to search for specific donors by name using the search input field, and the "Search" button triggers filtering of the donor list based on the entered donor name.

5. Modal for Adding Donor:

- Simplifies the process of adding a new donor entry, and it provides fields for donor name, age, contact number, blood type (dropdown), and last donation date within a modal window.

Common Tasks:

1. Viewing Donors:

- Users can effortlessly browse through the list of blood donors, viewing essential details, the tabular layout enhances readability and comprehension.

2. Adding a New Donor:

- Users can initiate the addition of a new donor entry by clicking the "Add Donor" button and completing the required details in the modal window.

3. Searching for Donors:

- The search functionality facilitates quick retrieval of specific donors by name, streamlining the user experience.

User Interaction:**1. Input Validation:**

- The modal form mandates input for fields such as donor name, age, contact number, blood type, and last donation date, it also has numeric validation for age and contact number ensures data accuracy and dropdown selection for blood type enhances precision.

2. Responsive Design:

- Ensures optimal usability across various devices and screen sizes, enhancing accessibility.

6. SCHEDULE-BLOOD-REQUEST.HTML

Initiate blood donation requests seamlessly with the Schedule-Blood-Request.html section. Whether scheduling a donation or managing existing requests, this feature provides a user-friendly interface for efficient blood donation management.

Features and Functionalities:**1. Navigation Bar:**

- Consistent navigation throughout the application, offering links to Home, Blood List, Donors, Schedule Blood Donation, List of Donor Requests, Schedule Blood Request (active), and List of Blood Requests and it also includes a "Login" button for user authentication.

2. Schedule Blood Request Form:

- Enables users to submit requests for blood donations seamlessly, and form fields include Receiver Name, Age, Contact Number, Requested Blood Type, Last Received Date, also any Medical Conditions, and displays a toast notification upon successful submission, confirming the request.

3. List of Blood Requests:

- Presents a comprehensive list of blood donation requests for reference and management, also table columns feature Receiver Name, Age, Blood Type, Contact Number, Last Received Date, Medical Conditions, and possibly Actions for request management.

Step-by-Step Instructions:

1. Scheduling Blood Request:

- Navigate to the "Schedule Blood Request" section, and fill out the form with the required details, including Receiver Name, Age, Contact Number, Requested Blood Type, Last Received Date, and Any Medical Conditions, then submit the form finally await the toast notification to confirm successful request submission.

2. Viewing List of Blood Requests:

- Access the "List Of Blood Requests" section, and explore the table displaying the list of blood donation requests, then review details such as Receiver Name, Age, Blood Type, Contact Number, Last Received Date, and Medical Conditions, finally utilize available actions for managing each request, if applicable.

Common Tasks:

1. Requesting Blood Donation:

- Users can utilize the "Schedule Blood Request" form to submit blood donation requests by providing relevant details.

2. Viewing List Of Blood Requests:

- Users can refer to the "List of Blood Requests" section to view a consolidated list of blood donation requests, facilitating efficient management and response.

3. Managing Blood Donation Requests:

- Admins or authorized users can perform actions on blood donation requests, such as approval, rejection, or updating details, based on application functionality and user permissions.

7. SCHEDULE-BLOOD-DONATION.HTML

Simplify blood donation scheduling with the Schedule-Blood-Donation.html section. From submitting donation requests to managing donor requests, this feature ensures a hassle-free process for both donors and recipients.

Features and Functionalities:

1. Navigation Bar:

- Provides consistent navigation across the application, offering links to Home, Blood List, Donors, Schedule Blood Donation (active), List of Donor Requests, Schedule Blood Request, and List of Blood Requests and it includes a "Login" button for user authentication.

2. Blood Donation Scheduling Form:

- Allows users to submit requests for blood donation effortlessly, form fields include Donor Name, Age, Contact Number, Blood Type, Last Donation Date, and Message, and upon submission, a toast notification confirms the successful request.

3. List of Donor Requests:

- Displays a comprehensive list of donor requests for easy reference and management, also table columns feature Donor Name, Age, Blood Type, Contact Number, Last Donation Date, Message, and possibly Actions for request management.

4. Responsive Design:

- Ensures the application is accessible and user-friendly across various devices and screen sizes.

Step-by-Step Instructions:

1. Schedule Blood Donation:

- Navigate to the "Schedule Blood Donation" section. It completes the form with the required details, including Donor Name, Age, Contact Number, Blood Type, Last Donation Date, and Message, and then submit the form, finally wait the toast notification to confirm successful request submission.

2. View List of Donor Requests:

- Access the "List of Donor Requests" section, and then review the table displaying the list of donor requests, after that Examine details such as Donor Name, Age, Blood Type, Contact Number, Last Donation Date, and Message and finally utilize available actions, if applicable, to manage donor requests.

Common Tasks:

1. Scheduling Blood Donation:

- Users can utilize the "Schedule Blood Donation" form to submit blood donation requests by providing relevant details.

2. Viewing List of Donor Requests:

- Users can refer to the "List of Donor Requests" section to view a consolidated list of donor requests, facilitating efficient management and response.

8 FUTURE ENHANCEMENTS AND IMPROVEMENTS

The system now, have most of the business requirements accomplished, however, there always can be improvements and enhancements for such a large system, from front-end to back-end to user interface to even for the business requirements.

8.1 Front-end improvements

Currently the system utilizes pure JavaScript code, which is considered really fast and functional, however when the application's functionalities grow, the usage of modern web development frameworks should be present, such as ReactJS, VueJS or AngularJS, which will make the development of business requirements and user interface much faster, thanks to the already built libraries and packages provided by such frameworks and ecosystems. Also, the possibilities of using CSS frameworks such as Tailwind CSS can be present, which will make the lifecycle of designing and building the user interface much faster.

Adding a dark/light theme toggle to the system is a great feature to add, that provides nice and interesting user experience.

8.2 Back-end improvements

The system currently utilizes JavaScript local storage API, to store the information in the browser storage, such a way of development can provide fast read/write operations when in development environment, however such feature or way of development can be limited, due to the limited storage of the browser memory, when the applications grows, and in a real production environment the usage of a relational database can be used such as MySQL or MS SQL Server or any other relational database, and in order to deal with the database a server-side programming language can be used to construct a useful API to interact safely with the database, such as C# or Java or PHP.

8.3 Security improvements

When mentioning security, the focus on the roles and authorization, authentication processes should be highlighted, security plays a critical role in any system, specifically in health organizations' systems, because the information of the patients and the employees should be protected and secured from outside access. The system currently does not have a guard for unauthorized access, due to the ease of development and caring more about the core functionality, however the business requirements were built in a way that it cares about these features.

When talking about the protection of the system from outside access we are talking about the authorization process, means we need to identify that the current user is authorized and registered in the system to view/use the system. In the world of web development and system development, there are several ways to implement such a feature, it can be done using session-cookie authentication process, or using secret token that is provided by the back end.

Since the system has different roles, we care about the authentication of each user in the system, meaning that each user/actor will have specific actions that he can do and specific sections that he can view, as was discussed in the business requirements and modeling of the system.

So, these functionality and authentication/authorization processes can be implemented in the future when used or deployed in a production environment.

CONCLUSION

In conclusion, a blood bank management system not only offers great value but is also necessary for the efficient and risk-free administration of blood supplies. Because of this advancement in technology, data relating to patients, blood donors, and supply levels can now be accessible and stored in a single, centralized system.

Content-rich, highly interactive, and aesthetically pleasing web apps can be built with the help of JavaScript, HTML, CSS, and Bootstrap. They allow for a consistent user experience across a broad variety of platforms and devices by virtue of their flexibility, scalability, and browser compatibility.

This journey via research has not only increased my knowledge and capabilities as a researcher, but it has also ingrained in me a greater interest. Resilience and intellect have both been nurtured because of the challenges that were encountered throughout the course of the study, and I am convinced that the lessons learnt will continue to affect my future academic endeavors.

BIBLIOGRAPHY

- [1] “World Wide Web Vs Internet - What’s the Difference?” BBC Newsround, www.bbc.co.uk/newsround/47523993.
- [2] “Web Development - Wikipedia.” Web Development - Wikipedia, 1 Dec. 2012, en.wikipedia.org/wiki/Web_development.
- [3] “HTTP Responses.” HTTP Responses, www.ibm.com/docs/en/cics-ts/5.2?topic=protocol-http-responses.
- [4] SeniorQuant. “Website Categorization.” Medium, 6 Feb. 2023, medium.com/website-categorization/website-categorization-api-ca6c3e0f6c4d.
- [5] “What Is HTML and How Does Hypertext Markup Language Work?” TheServer-Side.com, 1 Feb. 2020, www.theserverside.com/definition/HTML-Hypertext-Markup-Language.
- [6] & rarr;, View Archive. “Learn What Are the Elements, Tags, and Attributes in HTML With Ex-amples.” Dynamic Web Training Blog, 24 Dec. 2019, www.dynamic-webtraining.com.au/blog/element-tag-attributes-in-html.
- [7] “Semantic HTML: What It Is and How to Use It Correctly.” Semrush Blog, www.semrush.com/blog/semantic-html5-guide.
- [8] “HTML Structure | Webflow University.” HTML Structure | Webflow University, university.webflow.com/lesson/html-structure#:~:text=The%20two%20primary%20structural%20components,and%20settings%20of%20a%20webpage
- [9] Singhal, Piyush. “What Are the Advantages of HTML?- Scaler Topics.” Scaler Topics, 17 Oct. 2022, www.scaler.com/topics/advantages-of-html.
- [10] “What Is CSS?” What Is CSS?, www.tutorialspoint.com/css/what_is_css.htm.
- [11] “CSS Selectors - Learn Web Development | MDN.” CSS Selectors - Learn Web Development | MDN, developer.mozilla.org/en-US/docs/Learn/CSS/Building_blocks/Selectors.
- [12] “CSS Selectors - Complete List - Dofactory.” CSS Selectors - Complete List - Dofactory, www.dofactory.com/css/ref/selectors#class-selectors.
- [13] “CSS Grid Vs Flexbox: A Tutorial to Understand the Key Differences.” Simplilearn.com, www.simplilearn.com/tutorials/css-tutorial/css-grid-vs-flexbox.

- [14] “The Box Model - Learn Web Development | MDN.” The Box Model - Learn Web Development | MDN, developer.mozilla.org/en-US/docs/Learn/CSS/Building_blocks/The_box_model.
- [15] “Responsive Web Design: What It Is and How to Use It — Smashing Magazine.” Smashing Magazine, 12 Jan. 2011, www.smashingmagazine.com/2011/01/guidelines-for-responsive-web-design.
- [16] “What Is JavaScript? - Learn Web Development | MDN.” What Is JavaScript? - Learn Web Development | MDN, developer.mozilla.org/en-US/docs/Learn/JavaScript/First_steps/What_is_JavaScript.
- [17] “JavaScript Library Vs JavaScript Frameworks - the Differences.” JavaScript Library Vs JavaScript Frameworks - the Differences, www.microverse.org/blog/javascript-library-vs-javascript-frameworks-the-differences.
- [18] “Web Performance | MDN.” Web Performance | MDN, developer.mozilla.org/en-US/docs/Web/Performance.
- [19] Njah, Faith. “JavaScript Debugging Techniques.” Sweetcode.io, 19 Apr. 2022, sweetcode.io/javascript-debugging-techniques.
- [20] Ugochi, Ukpai. “How JavaScript Works: The Different Ways of Declaring a Function + 5 Best Practices.” Medium, 19 Nov. 2021, blog.sessionstack.com/how-javascript-works-the-different-ways-of-declaring-a-function-5-best-practices-8a0324c06fe2.
- [21] “The Advantages and Disadvantages of JavaScript.” freeCodeCamp.org, 5 Dec. 2019, www.freecodecamp.org/news/the-advantages-and-disadvantages-of-javascript.
- [22] Bidve, Pratibha. “What Is Bootstrap in Programming?” What Is Bootstrap in Programming?, www.clariontech.com/blog/what-is-bootstrap-in-programming.
- [23] Prabhu, John. “Why Should You Use Bootstrap? | Responsive Front-end Development.” Tech Blogs by TechAffinity, 26 May 2020, techaffinity.com/blog/why-use-bootstrap-for-frontend-design.
- [24] “Benefits of Using Bootstrap for Web Design.” Benefits of Using Bootstrap for Web Design, 6 May 2020, www.clarity-ventures.com/blog/benefits-of-using-bootstrap-for-web-design.
- [25] Jacob Thornton, and Bootstrap contributors, Mark Otto. “Containers.” Containers · Bootstrap v5.0, getbootstrap.com/docs/5.0/layout/containers.

-
- [26] “Bootstrap Grid - JavaTpoint.” www.javatpoint.com, www.javatpoint.com/bootstrap-grid.
- [27] “What Is Web Application Security?” What Is Web Application Security? | F5, www.f5.com/glossary/web-application-security.
- [28] ProtectOnce. “Why Is Web Application Security Important | ProtectOnce.” ProtectOnce, 19 July 2022, protectonce.com/appsec-academy-for-developer/why-is-web-application-security-important.

LIST OF ABBREVIATIONS

| | |
|------|------------------------------|
| BBMS | Blood Bank Management System |
| CMS | Content Management System |
| CSS | Cascading Style Sheets |
| HTML | Hyper Text Markup Language |
| HTTP | Hypertext Transfer Protocol |
| JS | JavaScript |
| MFA | Multi-Factor Authentication |
| RWD | Responsive Web Design |
| SEO | Search Engine Optimization |
| WAFs | Web Application Firewalls |

LIST OF FIGURES

| | |
|---|----|
| Figure 1 Requirement Model | 32 |
| Figure 2 Donor Management Requirement Model | 33 |
| Figure 3 Blood Type Requirement Model | 34 |
| Figure 4 Blood Bank Manager Requirement Model | 35 |
| Figure 5 User Management Requirement Model | 37 |
| Figure 6 Blood Request Requirement Model | 38 |
| Figure 7 Reporting Requirement Model. | 39 |
| Figure 8 Performance Requirement Model | 40 |
| Figure 9 Usability Requirement Model | 41 |
| Figure 10 Reliability Requirement Model | 42 |
| Figure 11 Use Case of Blood Bank Administration System..... | 43 |
| Figure 12 Class Model of Blood Bank Administration System..... | 68 |
| Figure 13. UC1 Login sequence diagram. | 71 |
| Figure 14. UC2 Logout sequence diagram. | 72 |
| Figure 15. UC3 Add donor sequence diagram. | 72 |
| Figure 16. UC4 Edit donor sequence diagram. | 73 |
| Figure 17. UC5 Delete donor sequence diagram. | 74 |
| Figure 18. UC6 Search donor sequence diagram. | 75 |
| Figure 19. UC7 Record donation sequence diagram. | 76 |
| Figure 20. UC8 Schedule donation sequence diagram. | 77 |
| Figure 21. UC9 Verify donor eligibility sequence diagram. | 78 |
| Figure 22. UC10 Add blood collection sequence diagram. | 79 |
| Figure 23. UC11 Update blood collection sequence diagram. | 80 |
| Figure 24. UC12 Delete blood collection sequence diagram..... | 81 |
| Figure 25. UC13 Search blood collection sequence diagram | 82 |
| Figure 26. UC14 Create blood request sequence diagram..... | 83 |
| Figure 27. UC15 Verify recipient eligibility sequence diagram. | 84 |
| Figure 28. UC16 Create new user sequence diagram. | 85 |
| Figure 29. Login page wireframe..... | 86 |
| Figure 30. Donor list wireframe..... | 87 |
| Figure 31. Schedule blood donation wireframe. | 87 |
| Figure 32. Blood list wireframe | 88 |

| | |
|--|-----|
| Figure 33 Blood list page. | 89 |
| Figure 34 Donor's list page..... | 90 |
| Figure 35 Home page of the website. | 91 |
| Figure 36 Login section of the website..... | 92 |
| Figure 37 Sign up section of the website..... | 93 |
| Figure 38 Registering for blood donation section..... | 94 |
| Figure 39 Registering for requesting blood section. | 95 |
| Figure 40 Structure and Styling. | 97 |
| Figure 41 Navigation Bar Implementation | 98 |
| Figure 42 Blood Bank System Overview. | 99 |
| Figure 43 Blood Donation Section..... | 100 |
| Figure 44 Website Footer Navigation and Social Icons. | 101 |
| Figure 45 CSS Styling for Form and Table Elements. | 103 |

LIST OF TABLES

| | |
|--|----|
| Table 1. UC1 Login use case specification..... | 46 |
| Table 2. UC2 Logout use case specification..... | 47 |
| Table 3. UC3 Add donor use case specification. | 48 |
| Table 4. UC4 Edit donor use case specification. | 49 |
| Table 5. UC5 Delete donor use case specification..... | 50 |
| Table 6. UC6 Search donor use case specification. | 51 |
| Table 7. UC7 Record donation use case specification..... | 52 |
| Table 8. UC8 Schedule donation use case specification..... | 54 |
| Table 9. UC9 Verify donor eligibility use case specification. | 55 |
| Table 10. UC10 Add blood collection use case specification..... | 57 |
| Table 11. UC11 Update blood collection use case specification. | 59 |
| Table 12. UC12 Delete blood collection use case specification. | 60 |
| Table 13. UC13 Search blood collection use case specification..... | 61 |
| Table 14. UC14 Create blood request use case specification. | 62 |
| Table 15. UC15 Verify recipient eligibility use case specification. | 64 |
| Table 16. UC16 Create new user use case specification..... | 66 |

APPENDICES

APPENDIX P I: APPENDIX TITLE