**ONLINE BLOOD BANK**

**Mini Project Report**

submitted in partial fulfilment for the award of the degree of

**BACHELOR OF TECHNOLOGY**

**in**

**COMPUTER SCIENCE** and **ENGINEERING**

**By**

CH. KUSUMA YADAV - 20L31A0554

B. ARJUN KUMAR - 20L31A0514

E. DEEKSHITA - 20L31A0563

CH. POORNA MANOHAR - 20L31A0552

**Under the Guidance of**

**Mr. CH. SEKHAR**

**ASSOC. PROFESSOR**



**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

**VIGNAN’S INSTITUTE OF INFORMATION TECHNOLOGY** (Autonomous) Affiliated to JNTU Kakinada & Approved by AICTE, New Delhi

Re-Accredited by NBA & NAAC (CGPA of 3.41/ 4.00)

ISO 9001:2008, ISO 14001:2004, OHSAS 18001:2007 Certified Institution

VISAKHAPATNAM – 530 039

**VIGNAN’S INSTITUTE OF INFORMATION TECHNOLOGY**

**Department of Computer Science and Engineering**



**CERTIFICATE**

This is to certify that the project report entitled **“ONLINE BLOOD BANK”** is the bona fide record of project work carried out under my supervision by **Ch.** **Kusuma (20L31A0554), B. Arjun (20L31A0514), E. Deekshita (20L31A0563),** and **Ch. Poorna Manohar (20L31A0552)**, during the academic year 2017-2018, in partial fulfillment of the requirements for the award of the degree of Bachelor of Technology in Computer Science and Engineering of Jawaharlal Nehru Technological University, Kakinada. The results embodied in this project report have not been submitted to any other University or Institute for the award of any Degree or Diploma.

**Head of the Department** **Signature** **of Project Guide**

Dr.DEBNATH BHATTACHARYYA Mr.CH. SEKAR

PROFESSOR ASSOCIATE

PROFESSOR

**DECLARATION**

We hereby declare that the project report entitled “Online Blood Bank” has been written by us and has not been submitted either in part or whole for the award of any degree, diploma or any other similar title to this or any other university.

CH. KUSUMA YADAV - 20L31A0554

B. ARJUN KUMAR - 20L31A0514

E. DEEKSHITA - 20L31A0563

CH. POORNA MANOHAR - 20L31A0552

Date:

Place:

**ACKNOWLEDGEMENT**

It gives us a great sense of pleasure to acknowledge the assistance and cooperation we have received from several persons while undertaking this B. Tech. Final Year Project. We owe a special debt of gratitude to **Mr. CH. SEKHAR,** Associate Professor Department of Computer Science &amp; Engineering, for his/her constant support and guidance throughout the course of our work. His/her sincerity, thoroughness and perseverance have been a constant source of inspiration for us.

We also take the opportunity to acknowledge the contribution of Dr .**DEBNATH BHATTACHARYA,** Head, Department of Computer Science Engineering, for his/her full support and assistance during the development of the project.

We also do not like to miss the opportunity to acknowledge the contribution of all faculty members of the department for their kind assistance and cooperation during the development of our project. Last but not the least, we acknowledge our friends for their contribution in the completion of the project.

**ABSTRACT**

“ The number of persons who are in need of blood are increasing in large number day by day. In order to help people who are in need of blood, our Online Blood Bank can be used effectively for getting the details of blood donors having the same blood group and with in the same city. With the help of my Online Blood Bank people who are having the thought of donating blood gets registered in my Online Blood Bank giving his/her total details. A person who likes to donate blood gives his entire details i.e., fill in the registration form and can create a username with a password by which he can modify his details if at all there are any changes in his information given before. If at all the people find any difficulty in getting blood from the contact persons an admin is appointed to monitor all the transactions. In this way donating and getting blood for needy becomes easy. ”

#### CONTENT

**CERTIFICATE**

**ACKNOWLEDGEMENT**

**ABSTRACT**

**CONTENTS**

**CHAPTER 1:** INTRODUCTION

* 1. **INTRODUCTION TO FRONT END** 
     1. INTRODUCTION TO HTML
     2. INTRODUCTION TO CSS
     3. INTRODUCTION TO SASS
     4. INTRODUCTION TO BOOTSTRAP
     5. INTRODUCTION TO JAVA SCRIPT

**1.2 INTRODUCTION TO BACKEND**

1.2.1 INTRODUCTION TO PYTHON

1.2.2 INTRODUCTION TO DJANGO

**CHAPTER 2:** PROBLEM STATEMENT

**CHAPTER 3:** PROPOSED SYSTEM

* + - 1. INTRODUCTION
      2. PROPOSED ONLINE BLOOD BANK
      3. MODULE OF THE ONLINE BLOOD BANK
      4. SYSTEM REQUIREMENTS

**CHAPTER 4:** FLOW DIAGRAMS

1. USE CASE DIAGRAM

**CHAPTER 5:** IMPLEMENTATION AND OUTPUT SCREENS

**CHAPTER 6:** SOURCE CODE

**CHAPTER 7:** CONCLUSION

**Chapter 1**

**INTRODUCTION**

1. **INTRODUCTION**

The main object of this project is to help people who are in need of blood, our Online Blood Bank can be used effectively for getting the details of blood donors having the same blood group and with in the same city. With the help of my Online Blood Bank people who are having the thought of donating blood gets registered in my Online Blood Bank giving his/her total details. A person who likes to donate blood gives his entire details i.e., fill in the registration form and can create a username with a password by which he can modify his details if at all there are any changes in his information given before. If at all the people find any difficulty in getting blood from the contact persons an admin is appointed to monitor all the transactions. In this way donating and getting blood for needy becomes easy.

**1.1 INTRODUCTION TO FRONT-END**

Front-end web development is the development of the graphical user interface of a website, through the use of HTML, CSS, and JavaScript, so that users can view and interact with that

website.

Tools used for front-end development There are several tools and platforms, such as

WordPress, Joomla, and Drupal, available that can be used to develop the front end of a

website.

**1.1.1 INTRODUCTION TO HTML**

The Hypertext Markup Language or HTML is the standard markup language for documents designed to be displayed in a web browser. It can be assisted by technologies such as Cascading

Style Sheets (CSS) and scripting languages such as JavaScript.

Web browsers receive HTML documents from a web server or from local storage and render

the documents into multimedia web pages. HTML describes the structure of a web page semantic

-ally and originally included cues for the appearance of the document.

HTML elements are the building blocks of HTML pages. With HTML constructs, images

and other objects such as interactive forms may be embedded into the rendered page. HTML

provides a means to create structured documents by denoting structural semantics for text such as headings, paragraphs, lists, links, quotes and other items. HTML elements are delineated by tags, written using angle brackets. Tags such as <img/> and <input/> directly introduce content into the page. Other tags such as <p> surround and provide information about document text and may

include other tags as sub-elements. Browsers do not display the HTML tags but use them to inter

-pret the content of the page.

HTML can embed programs written in a scripting language such as JavaScript, which affects

the behavior and content of web pages. Inclusion of CSS defines the look and layout of content.

The College/universities/schools Wide Web Consortium (W3C), former maintainer of the HTML and current maintainer

of the CSS standards, has encouraged the use of CSS over explicit presentational HTML since 1997.[2] A form of HTML, known as HTML5, is used to display video and audio, primarily

using the element, in collaboration with javascript.

In 1980, physicist Tim Berners-Lee, a contractor at CERN, proposed and prototyped

ENQUIRE, a system for CERN researchers to use and share documents. In 1989, Berners-Lee

wrote a memo proposing an Internet-based hypertext system.[3] Berners-Lee specified HTML

and wrote the browser and server software in late 1990. That year, Berners-Lee and CERN data systems engineer Robert Cailliau collaborated on a joint request for funding, but the project was

not formally adopted by CERN. In his personal notes[4] from 1990 he listed[5] "some of the

many areas in which hypertext is used" and put an encyclopedia first.

The first publicly available description of HTML was a document called "HTML Tags",

first mentioned on the Internet by Tim Berners-Lee in late 1991.[6][7] It describes 18 elements

comprising the initial, relatively simple design of HTML. Except for the hyperlink tag, these

were strongly influenced by SGML guid , an in-house Standard Generalized Markup Language (SGML)-based documentation format at CERN. Eleven of these elements still exist in HTML

4.[8]

HTML is a markup language that web browsers use to interpret and compose text, images,

and other material into visual or audible web pages. Default characteristics for every item of

HTML markup are defined in the browser, and these characteristics can be altered or enhanced by

the web page designer's additional use of CSS. Many of the text elements are found in the 1988

ISO technical report TR 9537 Techniques for using SGML, which in turn covers the features of

early text formatting languages such as that used by the RUNOFF command developed in the

early 1960s for the CTSS (Compatible Time-Sharing System) operating system: these formatting commands were derived from the commands used by typesetters to manually format documents. However, the SGML concept of generalized markup is based on elements (nested annotated

ranges with attributes) rather than merely print effects, with also the separation of structure and markup; HTML has been progressively moved in this direction with CSS.

Berners-Lee considered HTML to be an application of SGML. It was formally defined as

such by the Internet Engineering Task Force (IETF) with the mid-1993 publication of the first proposal for an HTML specification, the "Hypertext Markup Language (HTML)" Internet Draft

by Berners-Lee and Dan Connolly, which included an SGML Document type definition to define

the grammar.[9][10] The draft expired after six months, but was notable for its acknowledgment

of the NCSA Mosaic browser's custom tag for embedding in-line images, reflecting the IETF's philosophy of basing standards on successful prototypes. Similarly, Dave Raggett's competing Internet-Draft, "HTML+ (Hypertext Markup Format)", from late 1993, suggested standardizing already-implemented features like tables and fill-out forms.[11]

After the HTML and HTML+ drafts expired in early 1994, the IETF created an HTML

Working Group, which in 1995 completed "HTML 2.0", the first HTML specification intended to

be treated as a standard against which future implementations should be based.[12]

Further development under the auspices of the IETF was stalled by competing interests.

Since 1996, the HTML specifications have been maintained, with input from commercial software vendors, by the College/universities/schools Wide Web Consortium (W3C).[13] However, in 2000, HTML also became

an international standard (ISO/IEC 15445:2000). HTML 4.01 was published in late 1999, with

further errata published through 2001. In 2004, development began on HTML5 in the Web Hyper

text Application Technology Working Group (WHATWG), which became a joint deliverable with

the W3C in 2008, and completed and standardized on 28 October 2014.

## 

## 1.1.2 INTRODUCTION TO CSS

Cascading Style Sheets (CSS) is a style sheet language used for describing the presentation

of a document written in a markup language such as HTML or XML (including XML dialects

such as SVG, MathML or XHTML).[1] CSS is a cornerstone technology of the College/universities/schools Wide Web, alongside HTML and JavaScript.[2]

CSS is designed to enable the separation of presentation and content, including layout, colors,

and fonts.[3] This separation can improve content accessibility; provide more flexibility and

control in the specification of presentation characteristics; enable multiple web pages to share formatting by specifying the relevant CSS in a separate .css file, which reduces complexity and repetition in the structural content; and enable the .css file to be cached to improve the page load speed between the pages that share the file and its formatting.

Separation of formatting and content also makes it feasible to present the same markup page

in different styles for different rendering methods, such as onscreen, in print, by voice (via speech-based browser or screen reader), and on Braille-based tactile devices. CSS also has rules for

alternate formatting if the content is accessed on a mobile device.[4]

The name cascading comes from the specified priority scheme to determine which style rule applies if more than one rule matches a particular element. This cascading priority scheme is predictable.

The CSS specifications are maintained by the College/universities/schools Wide Web Consortium (W3C). Internet media type (MIME type) text/css is registered for use with CSS by RFC 2318 (March 1998). The W3C operates a free CSS validation service for CSS documents.

**1.1.3 INTRODUCTION TO BOOTSTRAP**

Bootstrap is an open-source and free CSS framework, which helps in directing a responsive device-friendly mobile-first front-end web page development tool. Bootstrap includes the CSS (Cascading Style Sheets), and an optional JavaScript supported design template (plug-ins) that deals with typography, implementation of buttons, forms, and various other components user interface. This framework helps in faster web development and supports developers in creating responsive web pages faster.

**Twitter Blueprint** was the first name of Bootstrap and was developed at Twitter by **Mr. Mark Otto** and **Jacob Thornton**. It got released as an open-source product in **August 2011 on GitHub**. The framework was mainly designed for encouraging the reliability and uniformity of web pages across internal tools. Before Bootstrap's existence, for making responsive sites and interface development, various external libraries were used, which brought inconsistency and gave rise to the heavy maintenance burden.

**1.1.4 INTRODUCTION TO SASS (Style Sheet Language)**

**Sass** (short for ***syntactically awesome style sheets***) is a [preprocessor](https://en.wikipedia.org/wiki/Preprocessor) [scripting language](https://en.wikipedia.org/wiki/Scripting_language) that is [interpreted](https://en.wikipedia.org/wiki/Interpreted_language) or [compiled](https://en.wikipedia.org/wiki/Compiled_language) into [Cascading Style Sheets](https://en.wikipedia.org/wiki/Cascading_Style_Sheets) (CSS). SassScript is the scripting language itself.

Sass consists of two [syntaxes](https://en.wikipedia.org/wiki/Syntax_(programming_languages)). The original syntax, called "the indented syntax," uses a syntax similar to [Haml](https://en.wikipedia.org/wiki/Haml). It uses [indentation](https://en.wikipedia.org/wiki/Indent_style) to separate [code blocks](https://en.wikipedia.org/wiki/Block_(programming)) and [newline](https://en.wikipedia.org/wiki/Newline) characters to separate rules. The newer syntax, "SCSS" (Sassy CSS), uses block formatting like that of CSS. It uses braces to denote code blocks and semicolons to separate rules within a block. The indented syntax and SCSS files are traditionally given the [extensions](https://en.wikipedia.org/wiki/Filename_extension) .sass and .scss, respectively.

[CSS3](https://en.wikipedia.org/wiki/CSS#CSS_3) consists of a series of selectors and pseudo-selectors that group rules that apply to them. Sass (in the larger context of both syntaxes) extends CSS by providing several mechanisms available in more traditional [programming languages](https://en.wikipedia.org/wiki/Programming_language), particularly [object-oriented languages](https://en.wikipedia.org/wiki/Object-oriented_programming), but that are not available to CSS3 itself. When SassScript is interpreted, it creates blocks of CSS rules for various selectors as defined by the Sass file. The Sass interpreter translates SassScript into CSS.

Alternatively, Sass can monitor the .sass or .scss file and translate it to an output .css file whenever the .sass or .scss file is saved.

The indented syntax is a metalanguage. SCSS is a [nested metalanguage](https://en.wikipedia.org/wiki/Metalanguage#Nested), as valid CSS is valid SCSS with the same [semantics](https://en.wikipedia.org/wiki/Semantics_of_programming_languages).

Sass Script provides the following mechanisms: [variables](https://en.wikipedia.org/wiki/Variable_(programming)), [nesting](https://en.wikipedia.org/wiki/Nesting_(computing)#In_programming), [mixins](https://en.wikipedia.org/wiki/Mixin), and selector [inheritance](https://en.wikipedia.org/wiki/Inheritance_(computer_science))

**1.1.5 INTRODUCTION TO JAVA SCRIPT**

**JavaScript** often abbreviated **JS**, is a [programming language](https://en.wikipedia.org/wiki/Programming_language) that is one of the core technologies of the [College/universities/schools Wide Web](https://en.wikipedia.org/wiki/World_Wide_Web), alongside [HTML](https://en.wikipedia.org/wiki/HTML) and [CSS](https://en.wikipedia.org/wiki/CSS). As of 2022, 98% of [websites](https://en.wikipedia.org/wiki/Website) use JavaScript on the [client](https://en.wikipedia.org/wiki/Client_(computing)) side for [webpage](https://en.wikipedia.org/wiki/Web_page) behavior, often incorporating third-party [libraries](https://en.wikipedia.org/wiki/Library_(computing)).[[13]](https://en.wikipedia.org/wiki/JavaScript#cite_note-lib_usage-13) All major [web browsers](https://en.wikipedia.org/wiki/Web_browser) have a dedicated [JavaScript engine](https://en.wikipedia.org/wiki/JavaScript_engine) to execute the [code](https://en.wikipedia.org/wiki/Source_code) on [users](https://en.wikipedia.org/wiki/User_(computing))' devices.

JavaScript is a [high-level](https://en.wikipedia.org/wiki/High-level_programming_language), often [just-in-time compiled](https://en.wikipedia.org/wiki/Just-in-time_compilation) language that conforms to the [ECMAScript](https://en.wikipedia.org/wiki/ECMAScript) standard It has [dynamic typing](https://en.wikipedia.org/wiki/Dynamic_typing), [prototype-based](https://en.wikipedia.org/wiki/Prototype-based_programming) [object-orientation](https://en.wikipedia.org/wiki/Object-oriented_programming), and [first-class functions](https://en.wikipedia.org/wiki/First-class_function). It is [multi-paradigm](https://en.wikipedia.org/wiki/Programming_paradigm), supporting [event-driven](https://en.wikipedia.org/wiki/Event-driven_programming), [functional](https://en.wikipedia.org/wiki/Functional_programming), and [imperative](https://en.wikipedia.org/wiki/Imperative_programming) [programming styles](https://en.wikipedia.org/wiki/Programming_paradigm). It has [application programming interfaces](https://en.wikipedia.org/wiki/Application_programming_interface) (APIs) for working with text, dates, [regular expressions](https://en.wikipedia.org/wiki/Regular_expression), standard [data structures](https://en.wikipedia.org/wiki/Data_structure), and the [Document Object Model](https://en.wikipedia.org/wiki/Document_Object_Model) (DOM).

The ECMAScript standard does not include any [input/output](https://en.wikipedia.org/wiki/Input/output) (I/O), such as [networking](https://en.wikipedia.org/wiki/Computer_network), [storage](https://en.wikipedia.org/wiki/Data_storage), or [graphics](https://en.wikipedia.org/wiki/Computer_graphics) facilities. In practice, the web browser or other [runtime system](https://en.wikipedia.org/wiki/Runtime_system) provides JavaScript APIs for I/O.

JavaScript engines were originally used only in web browsers, but are now core components of some [servers](https://en.wikipedia.org/wiki/Server_(computing)) and a variety of [applications](https://en.wikipedia.org/wiki/Application_software). The most popular runtime system for this usage is [Node.js](https://en.wikipedia.org/wiki/Node.js).

Although [Java](https://en.wikipedia.org/wiki/Java_(programming_language)) and JavaScript are similar in name, [syntax](https://en.wikipedia.org/wiki/Syntax_(programming_languages)), and respective [standard libraries](https://en.wikipedia.org/wiki/Standard_library), the two languages are distinct and differ greatly in design.

**1.2 INTRODUCTION TO BACK-END**

In the computer college/universities/schools, the "backend" refers to any part of a website or software program that users do not see. It contrasts with the frontend, which refers to a program's or website's user

interface. In programming terminology, the backend is the "data access layer," while the frontend

is the "presentation layer."

Most modern websites are dynamic, meaning webpage content is generated on-the-fly. A dynamic page contains one or more scripts that run on the web server each time the page is

accessed. These scripts generate the content of the page, which is sent to the user's web browser. Everything that happens before the page is displayed in a web browser is part of the backend

**1.2.1 INTRODUCTION TO PYTHON**

### Python is an interpreted, object-oriented, high-level programming language with dynamic semantics. Its high-level built in data structures, combined with dynamic typing and dynamic binding, make it very attractive for Rapid Application Development, as well as for use as a scripting or glue language to connect existing components together. Python's simple, easy to learn syntax emphasizes readability and therefore reduces the cost of program maintenance. Python supports modules and packages, which encourages program modularity and code reuse. The Python interpreter and the extensive standard library are available in source or binary form without charge for all major platforms, and can be freely distributed.

### Often, programmers fall in love with Python because of the increased productivity it provides. Since there is no compilation step, the edit-test-debug cycle is incredibly fast. Debugging Python programs is easy: a bug or bad input will never cause a segmentation fault. Instead, when the interpreter discovers an error, it raises an exception. When the program doesn't catch the exception, the interpreter prints a stack trace. A source level debugger allows inspection of local and global variables, evaluation of arbitrary expressions, setting breakpoints, stepping through the code a line at a time, and so on. The debugger is written in Python itself, testifying to Python's introspective power. On the other hand, often the quickest way to debug a program is to add a few print statements to the source: the fast edit-test-debug cycle makes this simple approach very effective.

### Why choose python?

Python is currently one of the most popular high-level languages. Guido van Rossum developed the language at the National Research Institute for Mathematics and Computer Science in the late 1980s.

It has the advantage of being both powerful as well as simple to work with. It is especially famous for being shorter and easier to comprehend. This ease in turn helps reduce the time taken to write the code. The compactness and simplicity of the code help have a lower error rate.

Another major advantage is that it helps you focus on the solution to the problem and not the structure or syntax of the problem.

In today’s data age, python has become one of the major leading languages to work with. It is not only used to connect to various database systems but also for making accessing, working and modifying them easy.

It also can further handle large quantities of data and help visualize the data. You can then further perform complex mathematics on this data.

Another feature of python allows you to connect to a server. This feature is helpful in cases where web applications are being developed. It is also great for creating workflows alongside the software.

**1.2.2 INTRODUCTION TO DJANGO**

Django is a Python framework that makes it easier to create web sites using Python.

Django takes care of the difficult stuff so that you can concentrate on building your web applications.

Django emphasizes reusability of components, also referred to as DRY (Don't Repeat Yourself), and comes with ready-to-use features like login system, database connection and CRUD operations (Create Read Update Delete).

## How does Django Work?

Django follows the MVT design pattern (Model View Template).

* Model - The data you want to present, usually data from a database.
* View - A request handler that returns the relevant template and content - based on the request from the user.
* Template - A text file (like an HTML file) containing the layout of the web page, with logic on how to display the data.

## Model

The model provides data from the database.

In Django, the data is delivered as an Object Relational Mapping (ORM), which is a technique designed to make it easier to work with databases.

The most common way to extract data from a database is SQL. One problem with SQL is that you have to have a pretty good understanding of the database structure to be able to work with it.

Django, with ORM, makes it easier to communicate with the database, without having to write complex SQL statements.

The models are usually located in a file called models.py

## Django History

Django was invented by Lawrence Journal-World in 2003, to meet the short deadlines in the newspaper and at the same time meeting the demands of experienced web developers.

Initial release to the public was in July 2005.

Latest version of Django is 4.0.3 (March 2022).

**Chapter 2**

**PROBLEM**

**STATEMENT**

Blood is considered as a living force of our body. Blood is require in urgent basis like accident or for anticipated transfusion during planned surgeries. Blood transfusion has been responsible for saving life. Ideally there should be enough blood units in a blood bank for everyday requirement for optimum functioning of the health care system. In present system people have to go to the blood bank to find their required blood unit which is unhandy. In most of the time we can see the shortage of blood units in the blood bank and patient has to take help from the relatives, friends and family members. It is very difficult to get blood donor in the time of emergency and urgency.

The main object of this project is to help people who are in need of blood, our Online Blood Bank can be used effectively for getting the details of blood donors having the same blood group and with in the same city. With the help of my Online Blood Bank people who are having the thought of donating blood gets registered in my Online Blood Bank giving his/her total details. A person who likes to donate blood gives his entire details i.e., fill in the registration form and can create a username with a password by which he can modify his details if at all there are any changes in his information given before. If at all the people find any difficulty in getting blood from the contact persons an admin is appointed to monitor all the transactions. In this way donating and getting blood for needy becomes easy.

**CHAPTER 3**

**PROPOSED**

**SYSTEM**

This proposed system is very much attainable and usable. This online system is provided with the awareness regarding blood donation which will increase the number of voluntary blood donors. It consists of central repository which gives the information regarding various blood donation camps available along with the details associated. Voluntary donors can register in the system to post their donation and customer in need of blood can register online and find suitable donor, hence making the whole process friendly and handy.

## ****Modules****

1. **User:**There are two type of user in this project donor and customer. Donor can register to the web site by giving his personal details like name, blood group, address, contact details etc. Once registration is done he can post his donation. Customer can register to the portal and view list of donors who are interested to donate blood. He can also search list of donors based on the blood group. Once he gets suitable donor he can directly contact the donor.
2. **Admin:** Admin is responsible to post all the information related to blood donation. Admin can view list of voluntary blood donors and help the needy by providing their information to blood bank and patients. Admin can send a blood request to other blood bank in case of emergency or shortage of blood.

**SYSTEM REQUIREMENTS**

**Software Requirements**

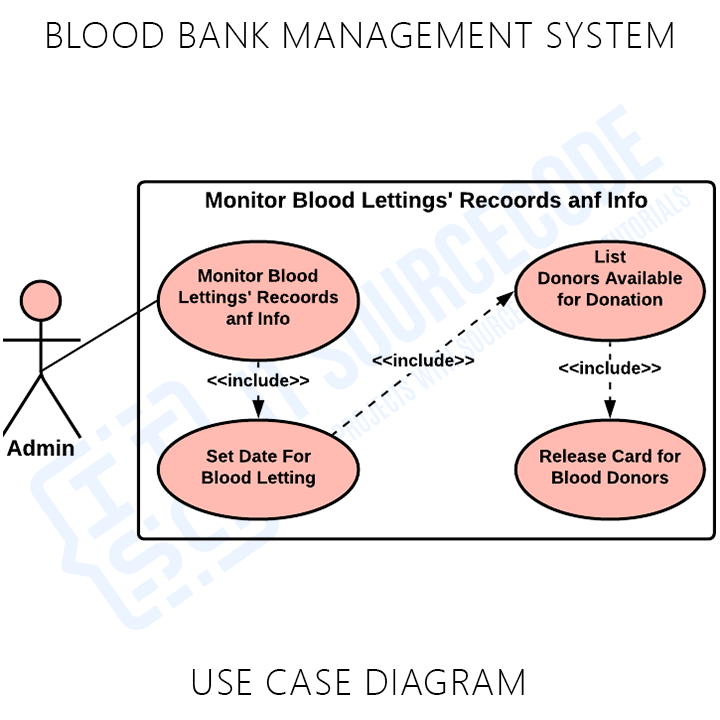
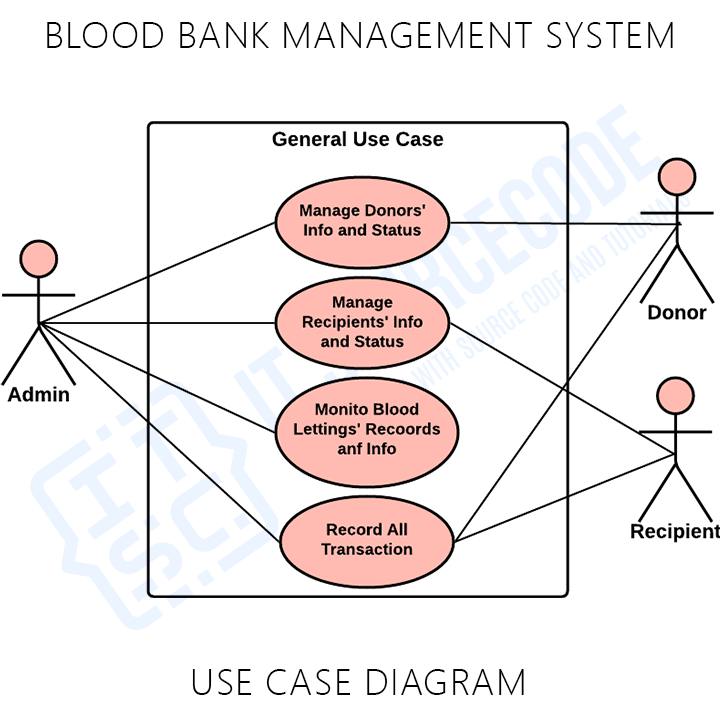
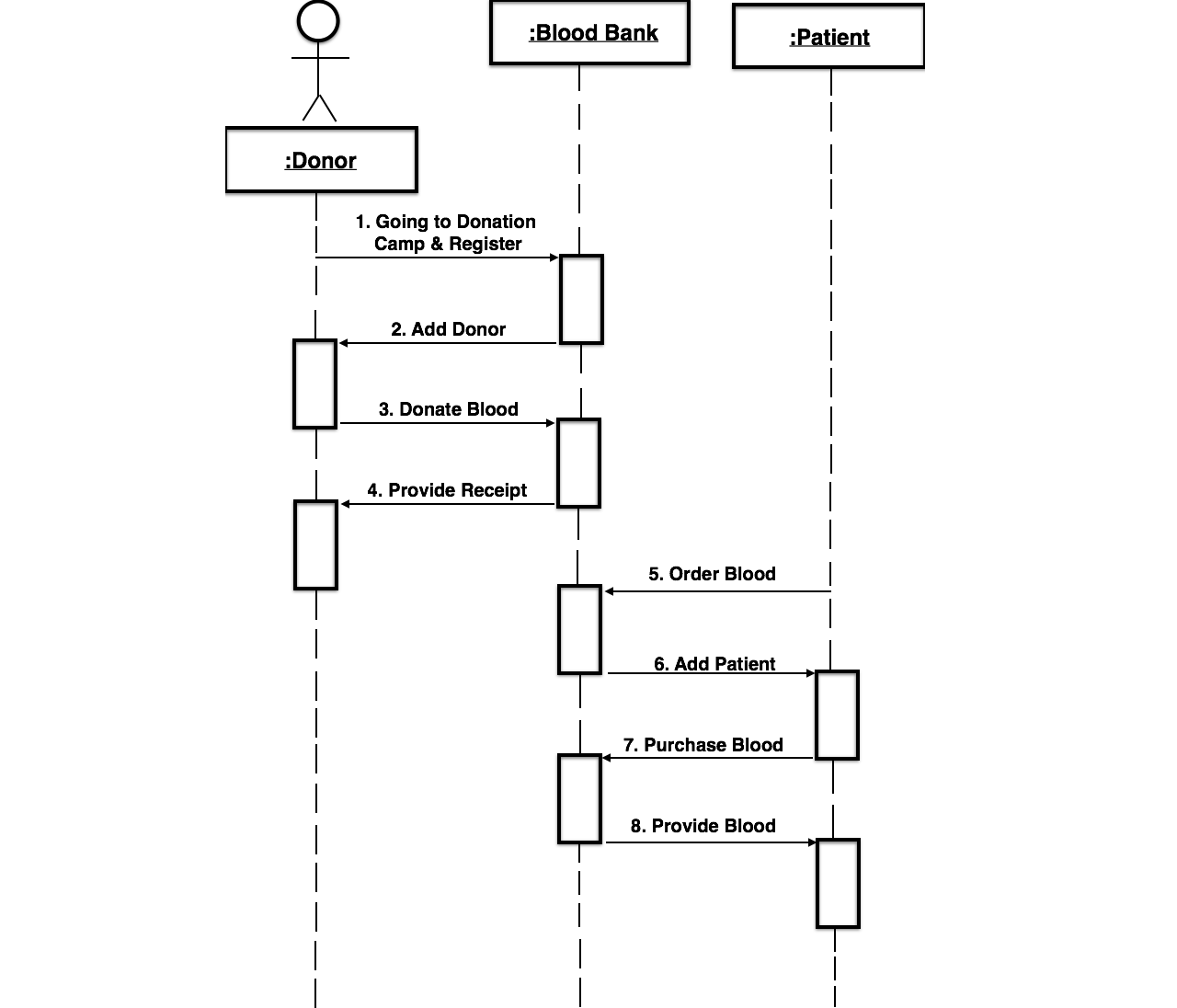
* **Any operating system**
* **Visual studio code or notepad++**
* **Python**
* **SQL**

## ****Hardware Requirements****

• Processor – i3 / AMD  
• Hard Disk – 2 GB  
• Memory – 1GB RAM

**CHAPTER 4**

**FLOW DIAGRAMS**

****

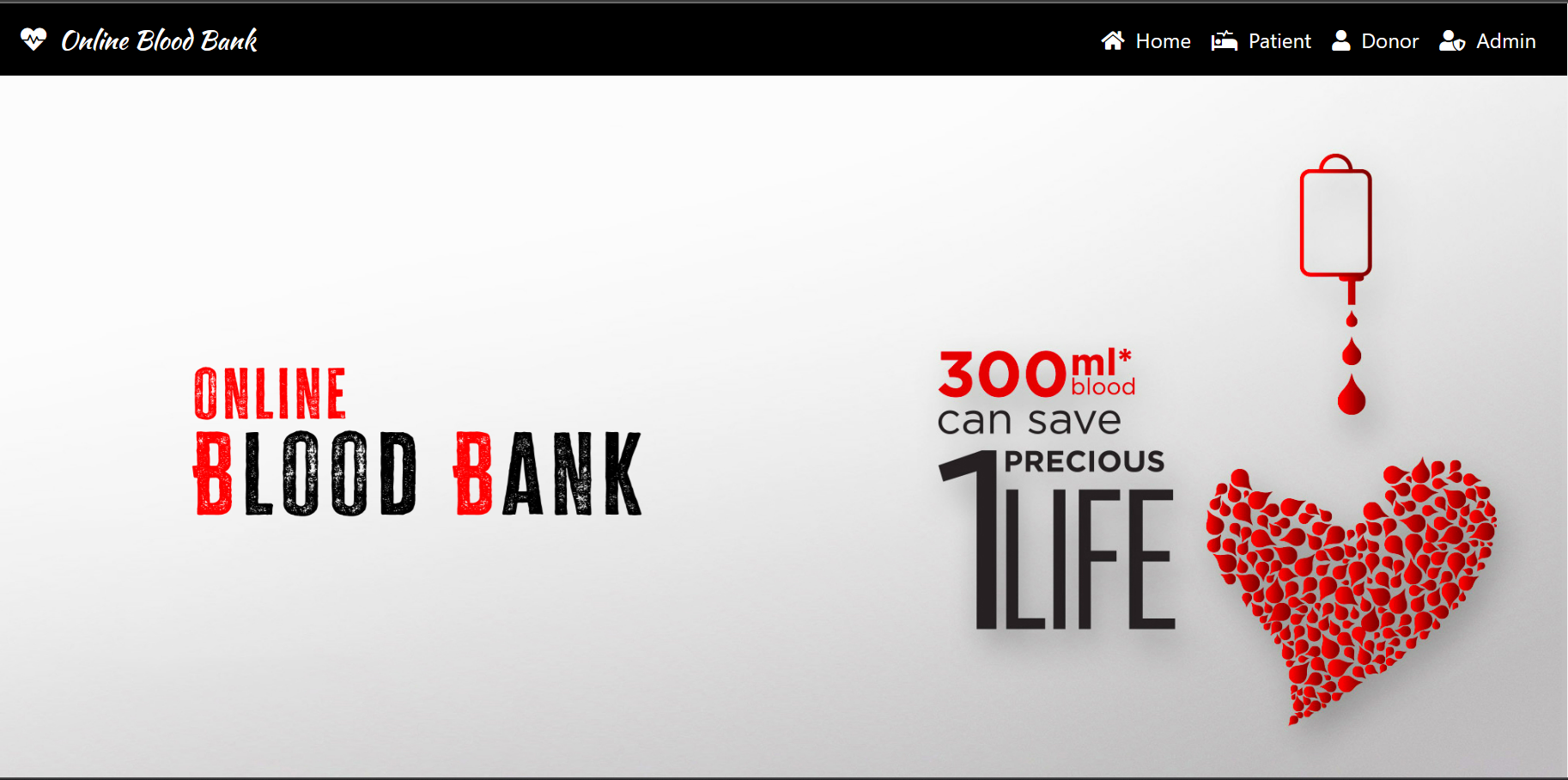
**CHAPTER 5**

**IMPLEMENTATION**

**AND**

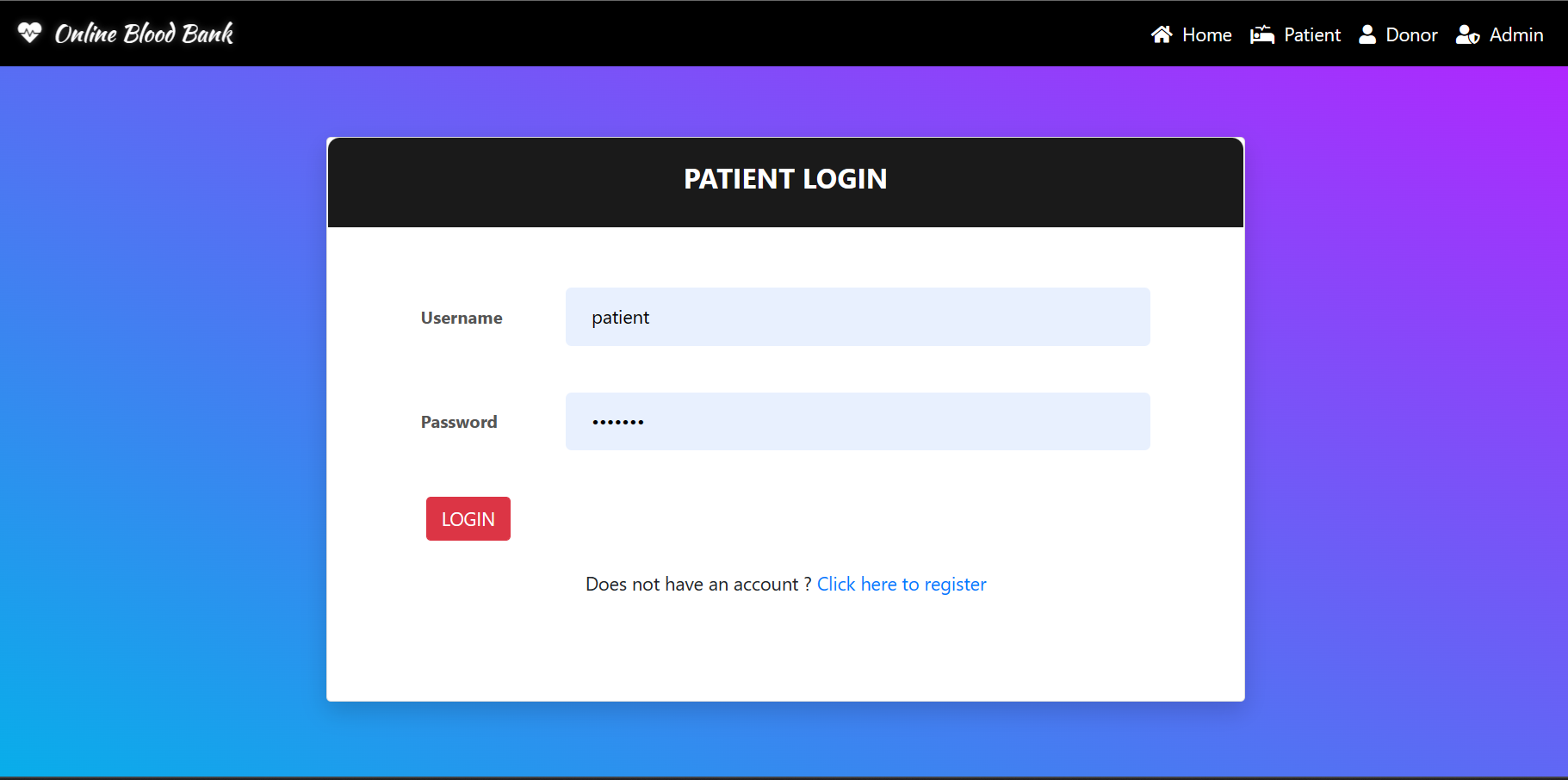
**OUTPUT SCREENS**

**Landing page ( screen 1 )**

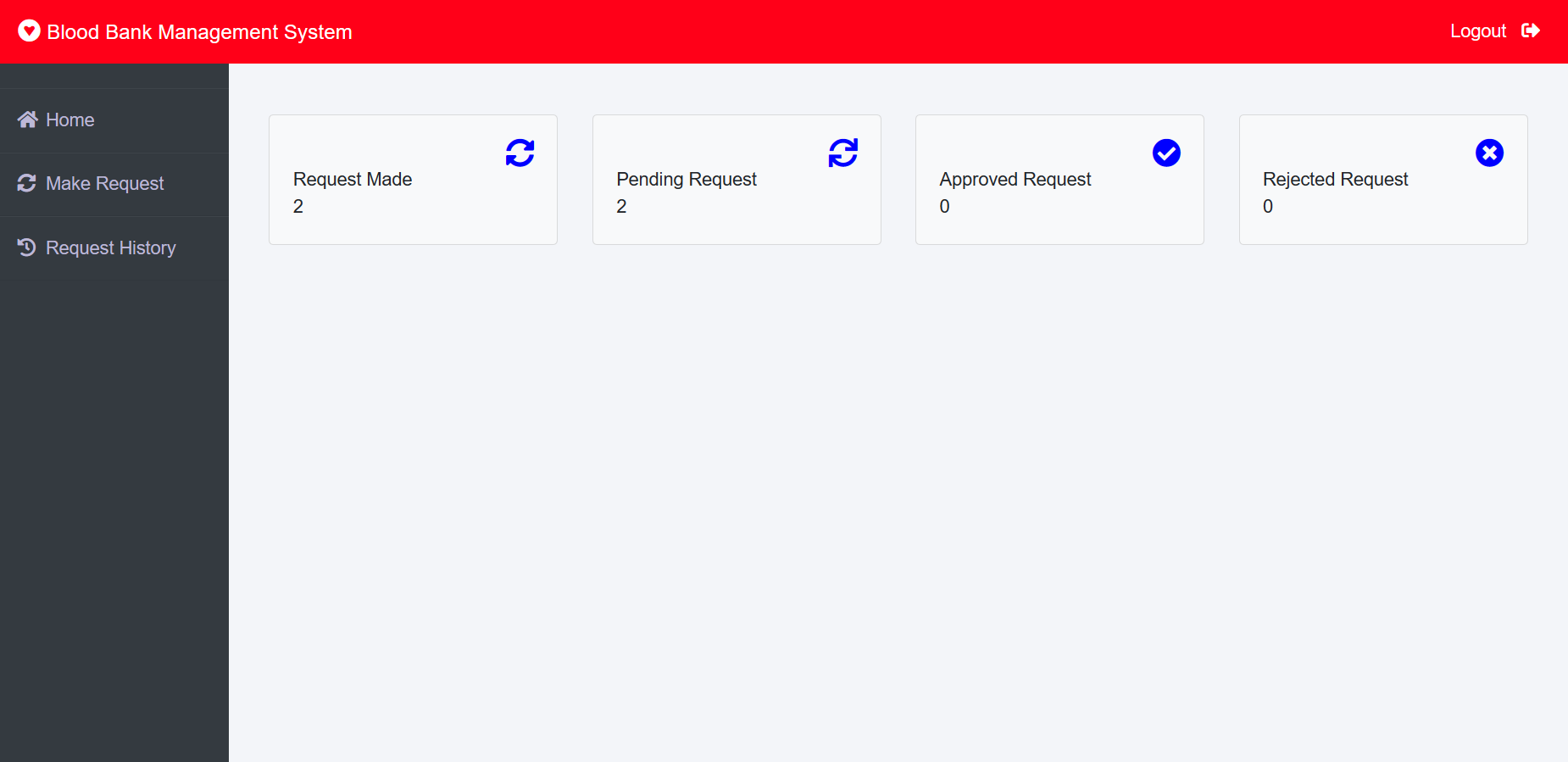
****

This is the first page viewed by the user before login

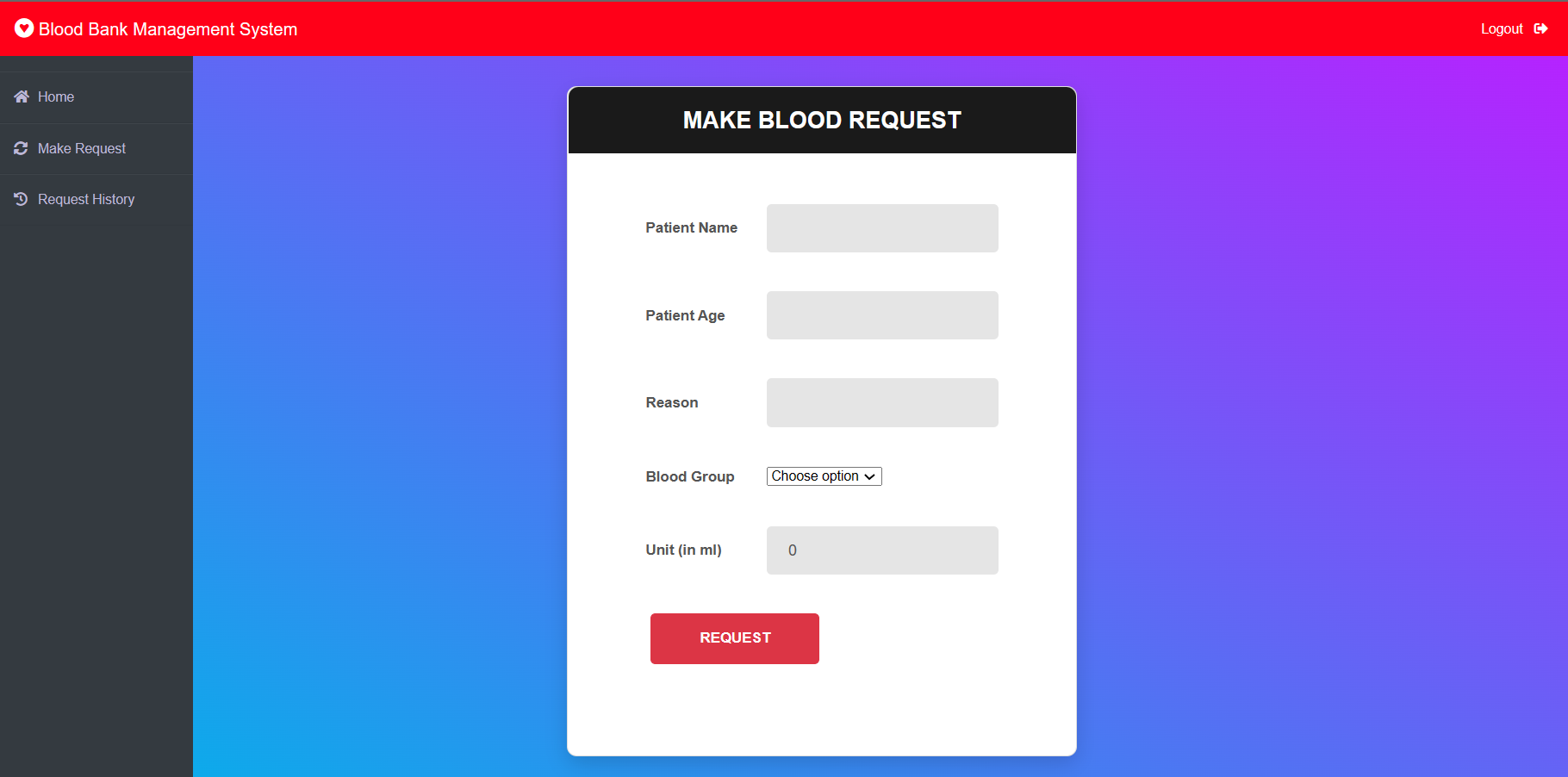
**Patient Login**

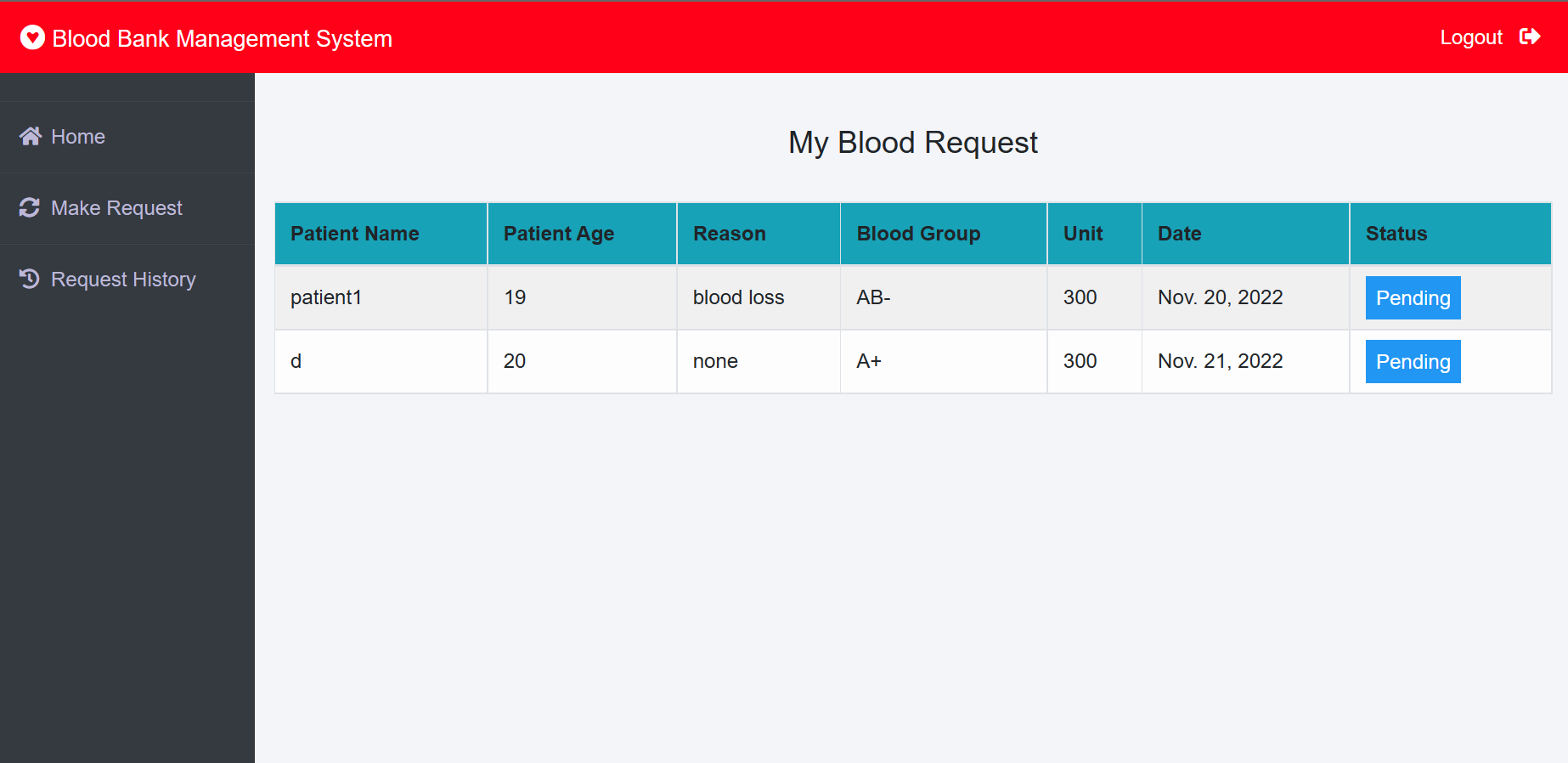


Login form for Patient

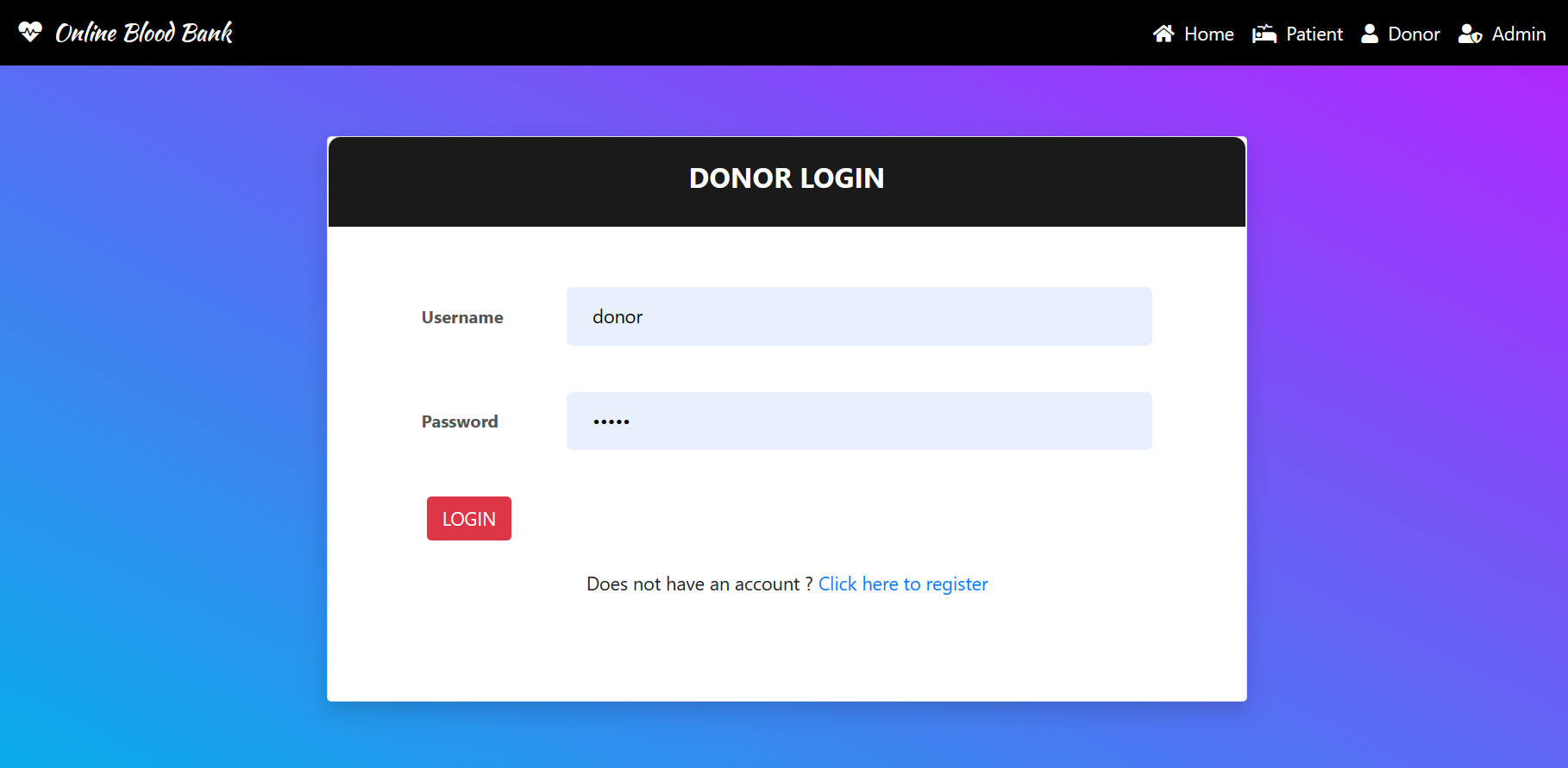
**Patient Dashboard**

**Patient request form**

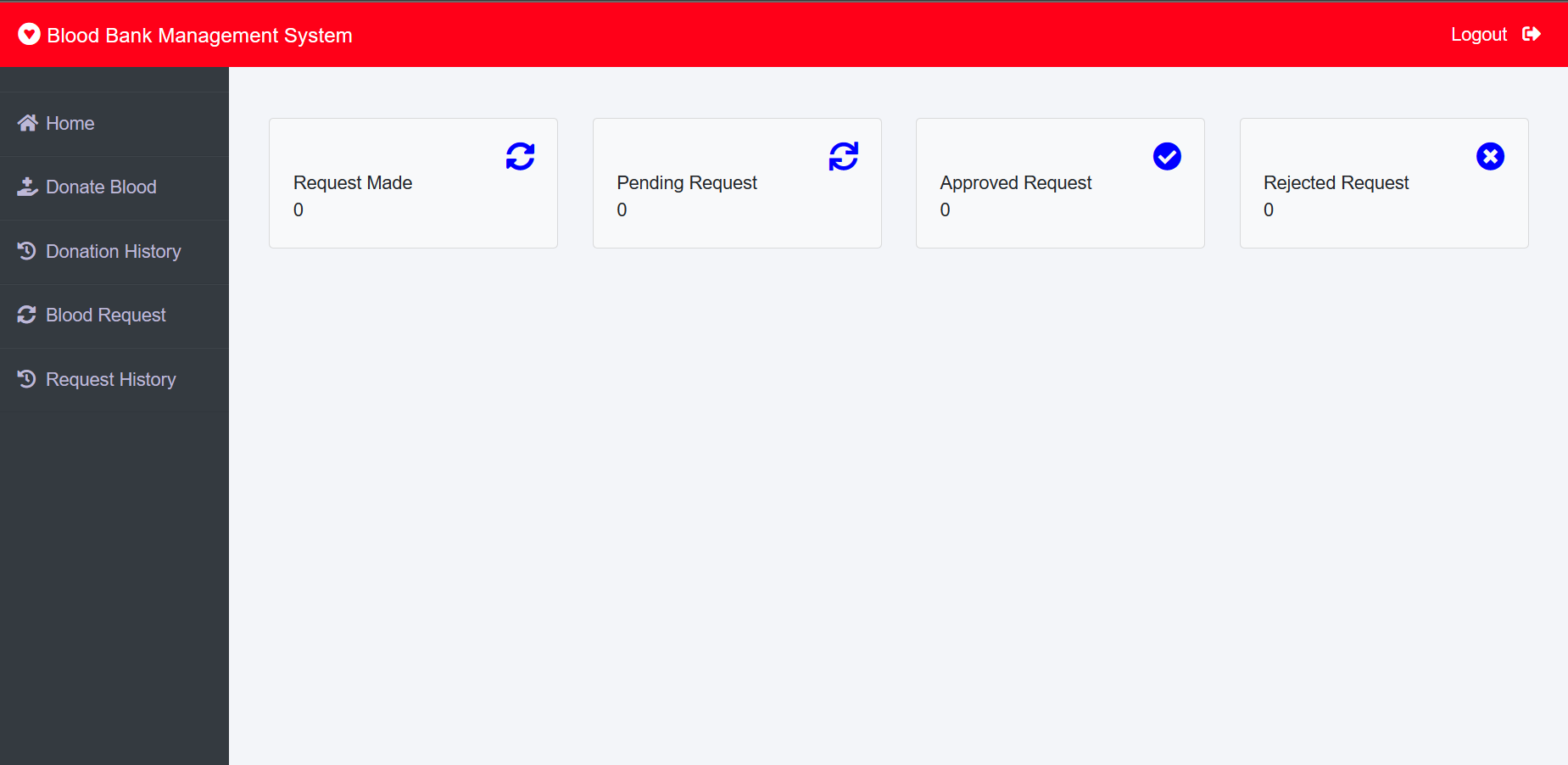


**Request history** 

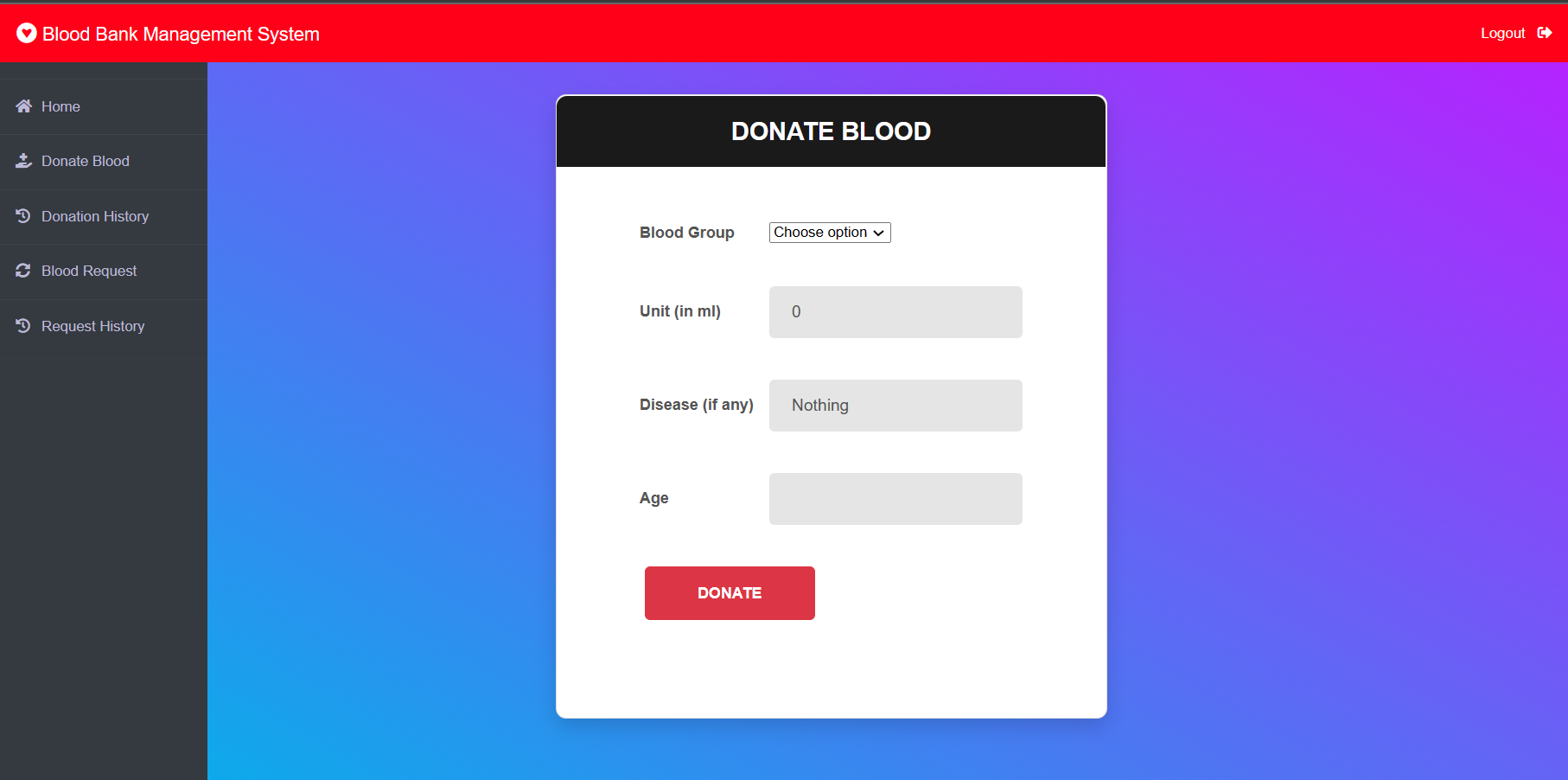
**Donor login**

****

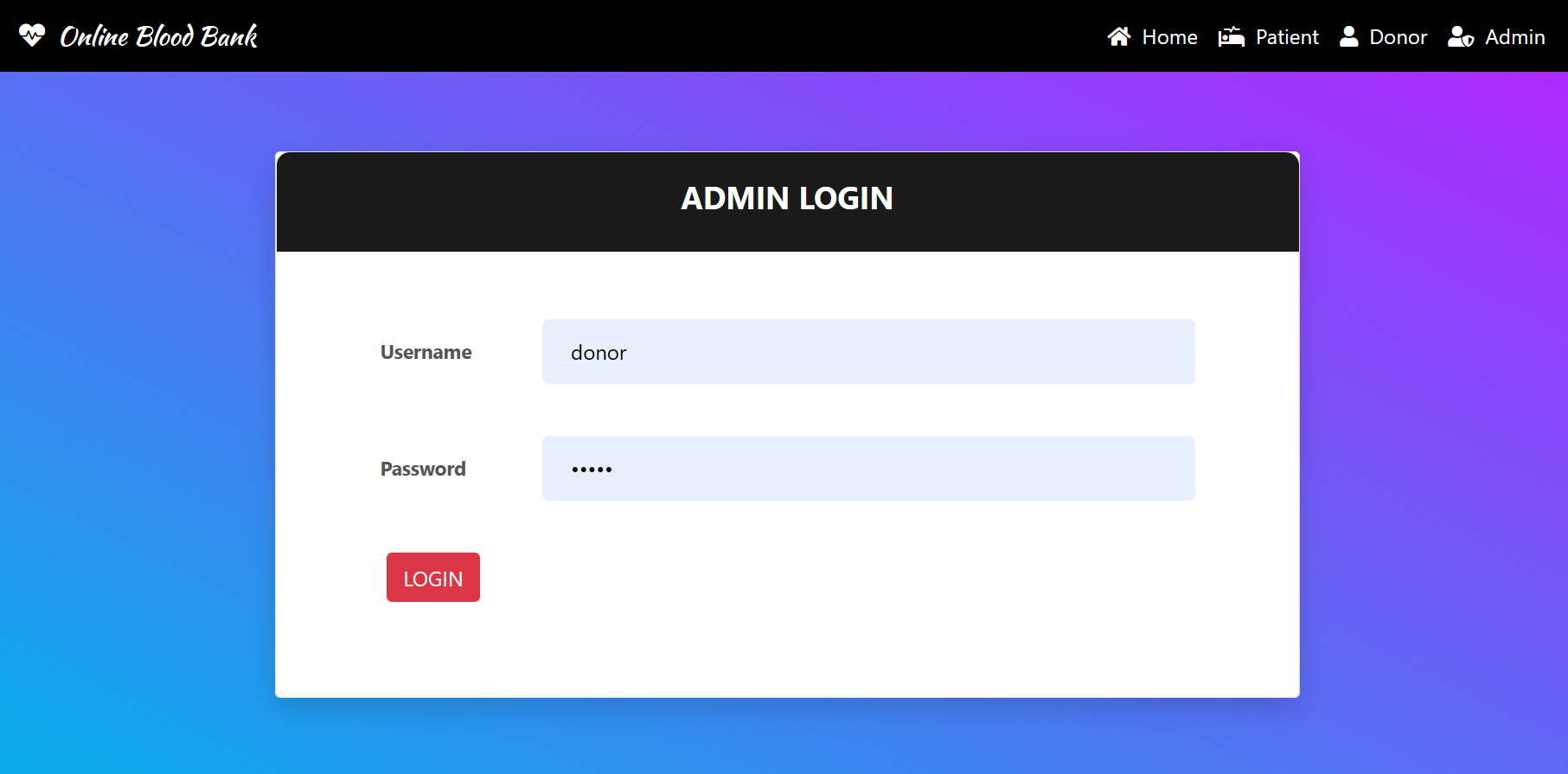
**Donor Dashboard**

****

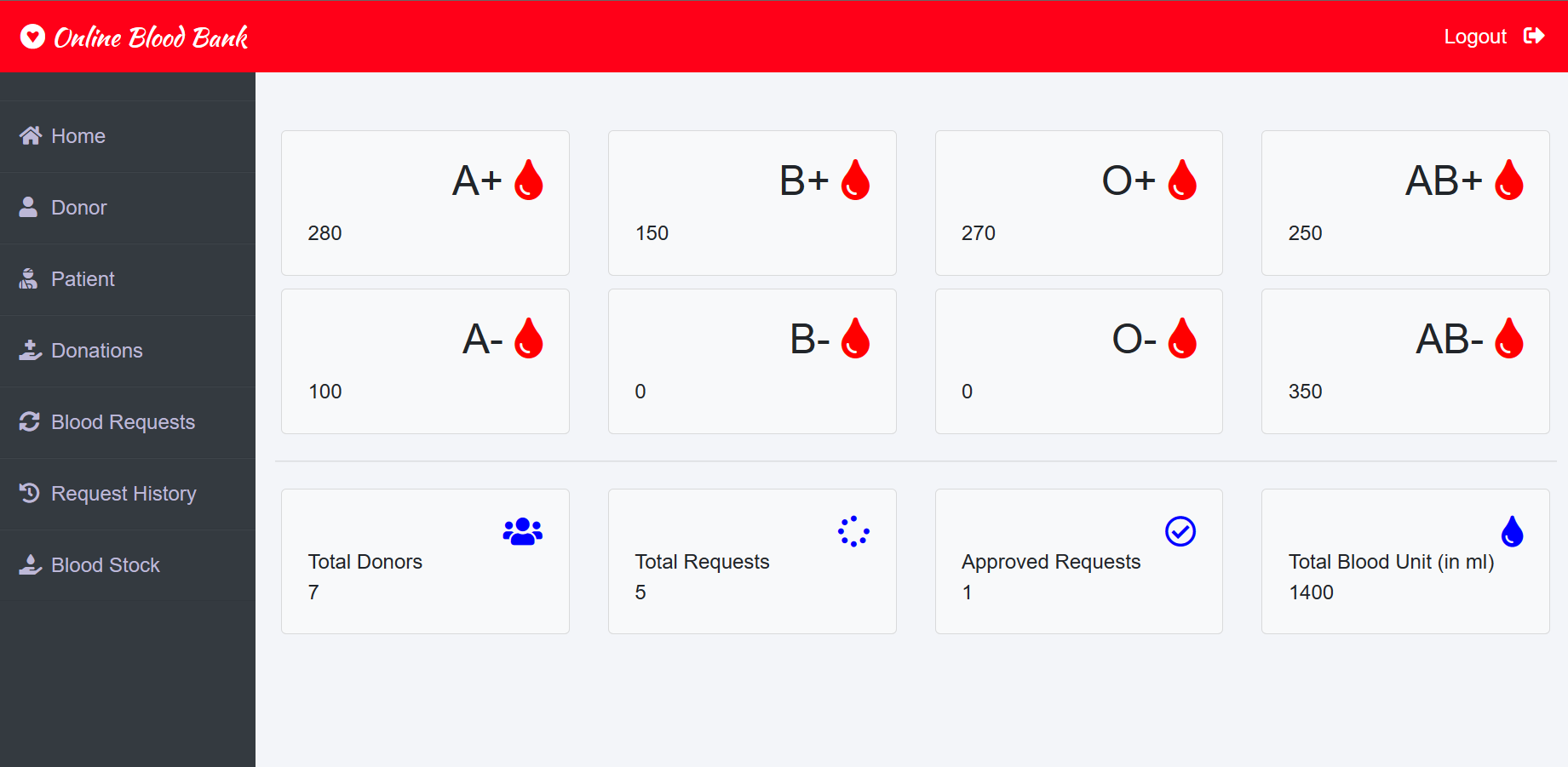
**Donor form**

****

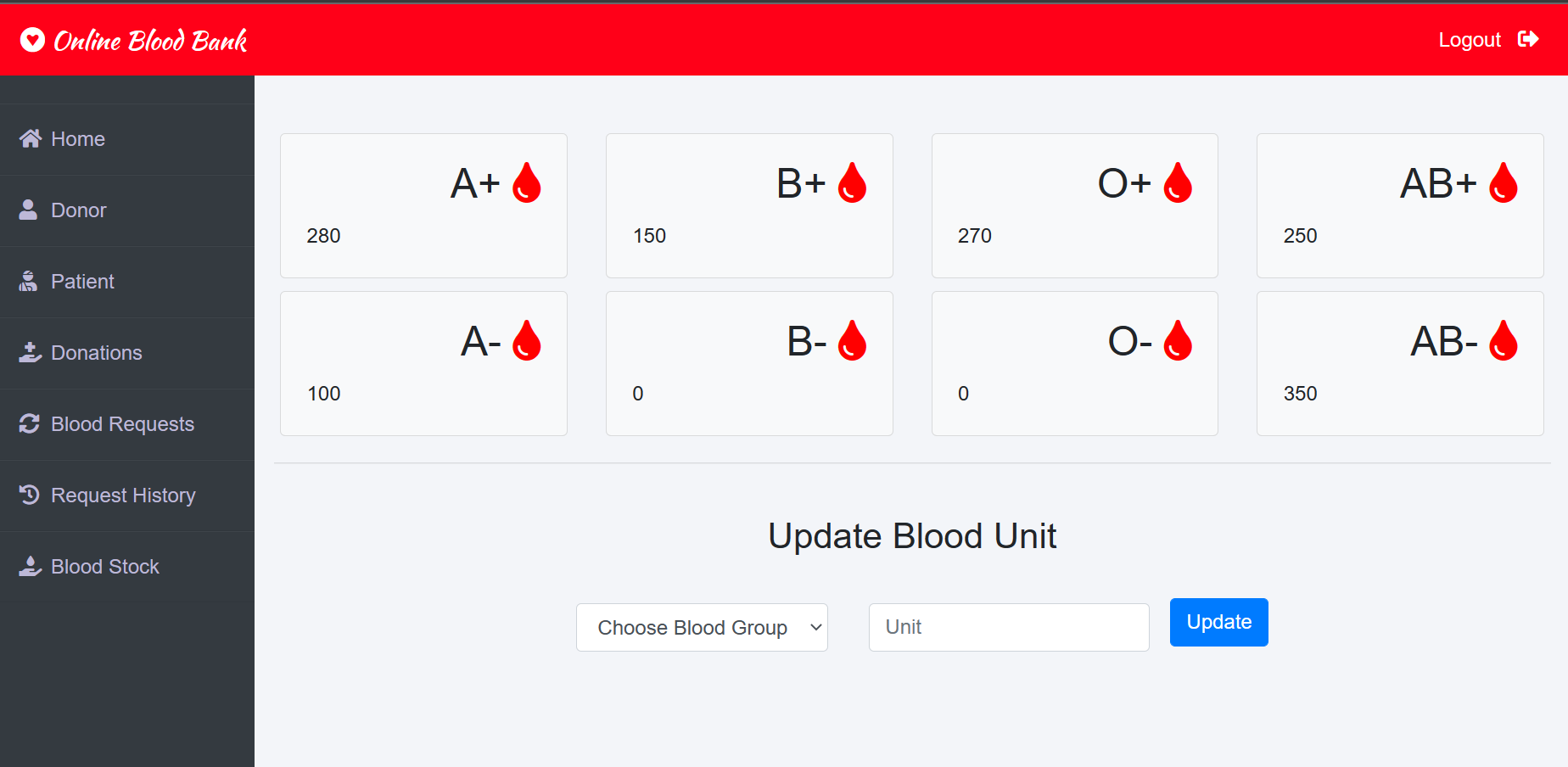
**Admin login**

****

**Admin Dashboard**

****

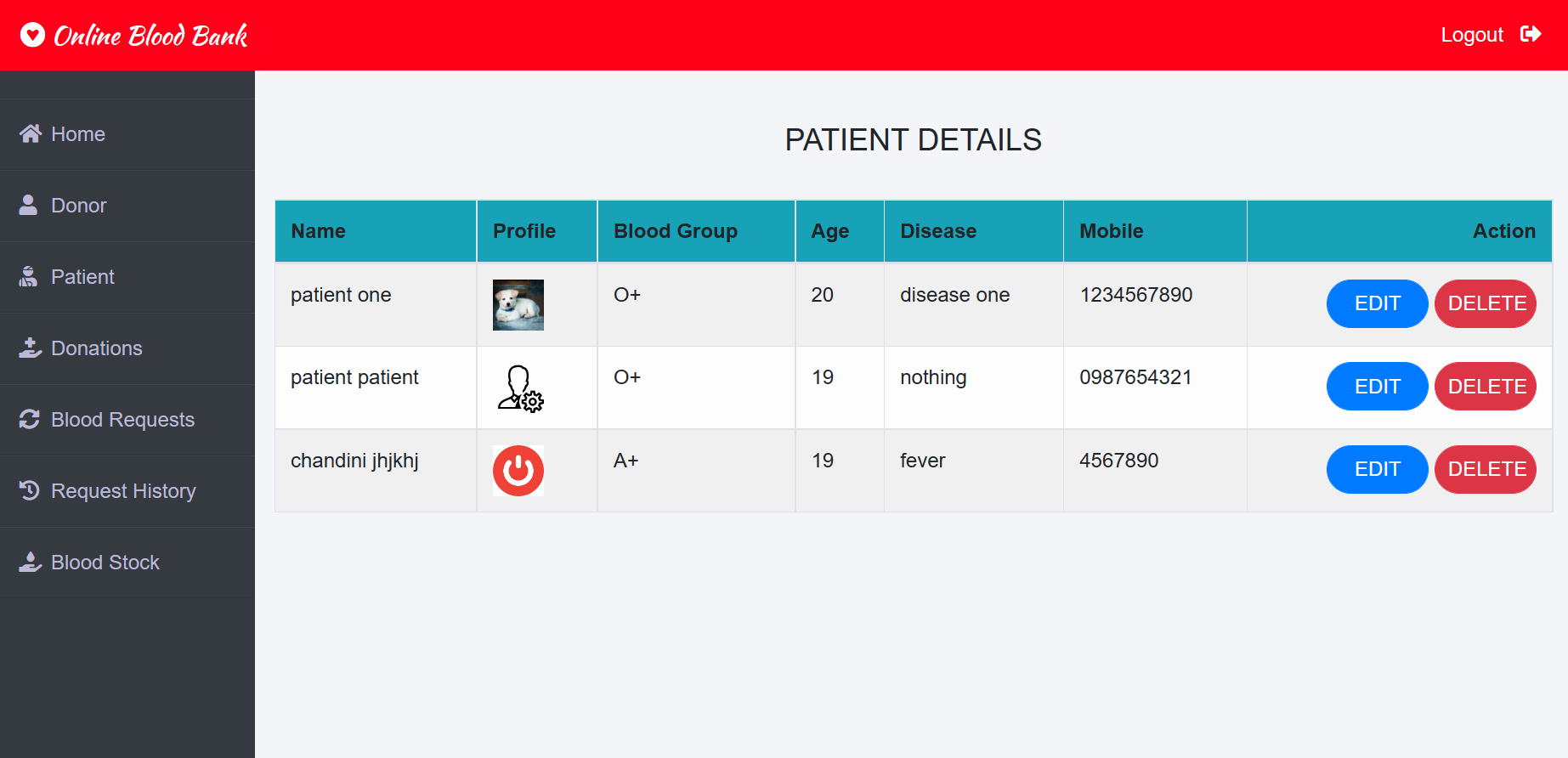
**Blood Stock**

****

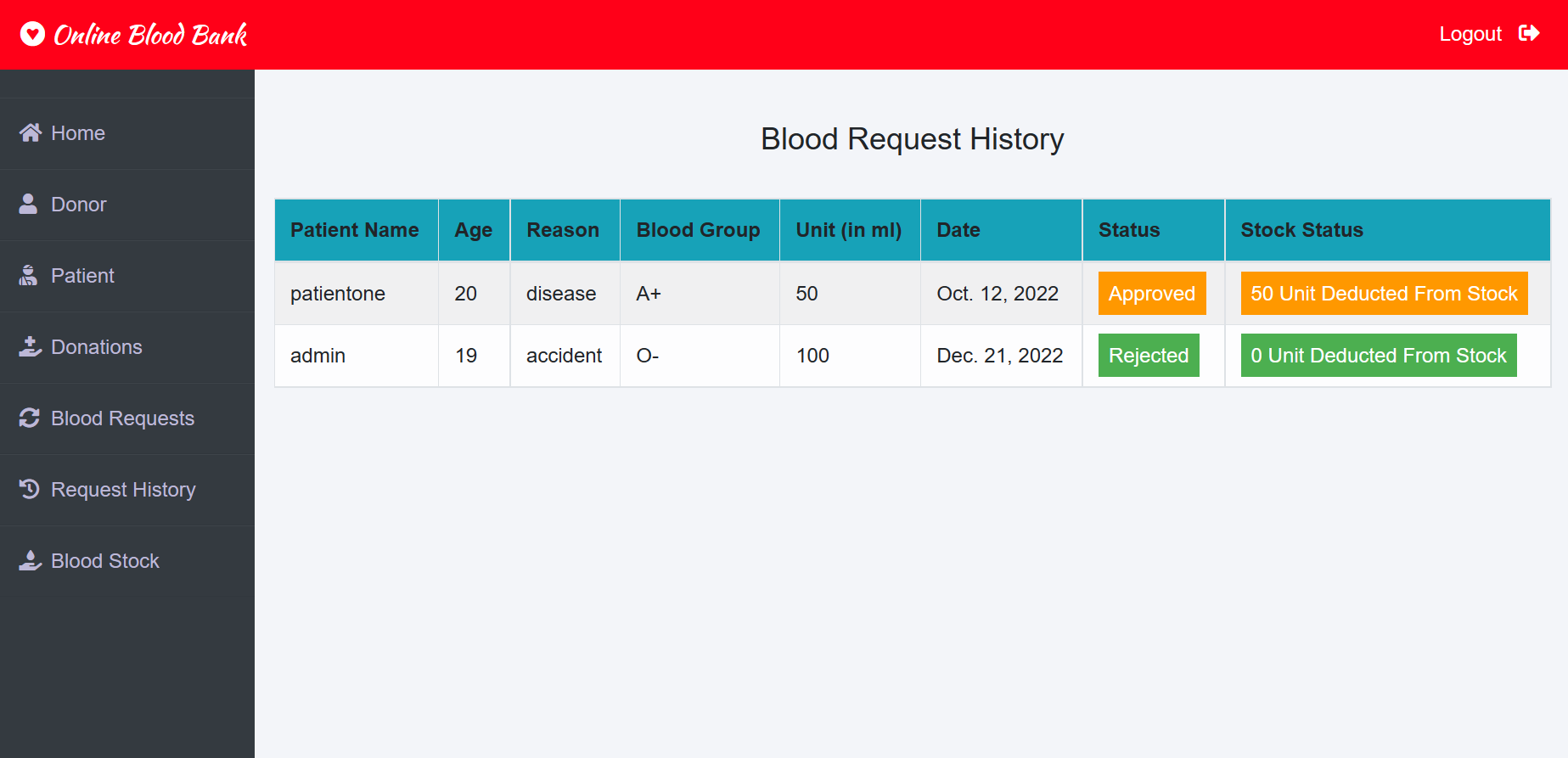
**Donor details**

****

**Patient Details**

****

**Blood Request History**

****

**CHAPTER 6**

**SOURCE CODE**

#init.py

from django.conf import settings

from django.db import migrations, models

import django.db.models.deletion

class Migration(migrations.Migration):

    initial = True

    dependencies = [

        migrations.swappable\_dependency(settings.AUTH\_USER\_MODEL),

    ]

    operations = [

        migrations.CreateModel(

            name='Donor',

            fields=[

                ('id', models.BigAutoField(auto\_created=True, primary\_key=True, serialize=False, verbose\_name='ID')),

                ('profile\_pic', models.ImageField(blank=True, null=True, upload\_to='profile\_pic/Donor/')),

                ('bloodgroup', models.CharField(max\_length=10)),

                ('address', models.CharField(max\_length=40)),

                ('mobile', models.CharField(max\_length=20)),

                ('user', models.OneToOneField(on\_delete=django.db.models.deletion.CASCADE, to=settings.AUTH\_USER\_MODEL)),

            ],

        ),

        migrations.CreateModel(

            name='BloodDonate',

            fields=[

                ('id', models.BigAutoField(auto\_created=True, primary\_key=True, serialize=False, verbose\_name='ID')),

                ('disease', models.CharField(default='Nothing', max\_length=100)),

                ('age', models.PositiveIntegerField()),

                ('bloodgroup', models.CharField(max\_length=10)),

                ('unit', models.PositiveIntegerField(default=0)),

                ('status', models.CharField(default='Pending', max\_length=20)),

                ('date', models.DateField(auto\_now=True)),

                ('donor', models.ForeignKey(on\_delete=django.db.models.deletion.CASCADE, to='donor.donor')),

            ],

        ),

    ]

#urls.py

from django.urls import path

from django.contrib.auth.views import LoginView

from . import views

urlpatterns = [

    path('donorlogin', LoginView.as\_view(template\_name='donor/donorlogin.html'),name='donorlogin'),

    path('donorsignup', views.donor\_signup\_view,name='donorsignup'),

    path('donor-dashboard', views.donor\_dashboard\_view,name='donor-dashboard'),

    path('donate-blood', views.donate\_blood\_view,name='donate-blood'),

    path('donation-history', views.donation\_history\_view,name='donation-history'),

    path('make-request', views.make\_request\_view,name='make-request'),

    path('request-history', views.request\_history\_view,name='request-history'),

]

#admins dashboard

{% extends 'blood/adminbase.html' %}

{% block content %}

{% load widget\_tweaks %}

<style>

            .xyz{

      display: table;

      margin-right: auto;

      margin-left: auto;

            }

</style>

<br><br>

<div class="container">

    <div class="row">

      <div class="col-sm-3">

        <div class="card bg-light">

          <div class="card-body">

              <div class="blood">

                  <h2>A+ <i class="fas fa-tint"></i></h2>

              </div><br><br>

              <div>

                  {{A1.unit}}

              </div>

          </div>

        </div>

      </div>

      <div class="col-sm-3">

          <div class="card bg-light">

              <div class="card-body">

                  <div class="blood">

                      <h2>B+ <i class="fas fa-tint"></i></h2>

                  </div><br><br>

                  <div>

                    {{B1.unit}}

                  </div>

              </div>

            </div>

      </div>

      <div class="col-sm-3">

          <div class="card bg-light">

              <div class="card-body">

                  <div class="blood">

                      <h2>O+ <i class="fas fa-tint"></i></h2>

                  </div><br><br>

                  <div>

                    {{O1.unit}}

                  </div>

              </div>

            </div>

        </div>

        <div class="col-sm-3">

          <div class="card bg-light">

              <div class="card-body">

                  <div class="blood">

                      <h2>AB+ <i class="fas fa-tint"></i></h2>

                  </div><br><br>

                  <div>

                    {{AB1.unit}}

                  </div>

              </div>

            </div>

        </div>

    </div>

    <div class="row">

      <div class="col-sm-3">

        <div class="card bg-light">

          <div class="card-body">

              <div class="blood">

                  <h2>A- <i class="fas fa-tint"></i></h2>

              </div><br><br>

              <div>

                {{A2.unit}}

              </div>

          </div>

        </div>

      </div>

      <div class="col-sm-3">

          <div class="card bg-light">

              <div class="card-body">

                  <div class="blood">

                      <h2>B- <i class="fas fa-tint"></i></h2>

                  </div><br><br>

                  <div>

                    {{B2.unit}}

                  </div>

              </div>

            </div>

      </div>

      <div class="col-sm-3">

          <div class="card bg-light">

              <div class="card-body">

                  <div class="blood">

                      <h2>O- <i class="fas fa-tint"></i></h2>

                  </div><br><br>

                  <div>

                    {{O2.unit}}

                  </div>

              </div>

            </div>

        </div>

        <div class="col-sm-3">

          <div class="card bg-light">

              <div class="card-body">

                  <div class="blood">

                      <h2>AB- <i class="fas fa-tint"></i></h2>

                  </div><br><br>

                  <div>

                    {{AB2.unit}}

                  </div>

              </div>

            </div>

        </div>

    </div>

<hr>

<br>

<h3 class="text-center">Update Blood Unit</h3><br>

<div class="xyz">

    <form class="form-inline" method="POST">

        {% csrf\_token %}

        <div class="form-group mx-sm-3 mb-6">

            <select name="bloodgroup" class="form-control">

                <option disabled="disabled" selected="selected">Choose Blood Group</option>

                <option>O+</option>

                <option>O-</option>

                <option>A+</option>

                <option>A-</option>

                <option>B+</option>

                <option>B-</option>

                <option>AB+</option>

                <option>AB-</option>

            </select>

          </div>

        <div class="form-group mx-sm-3 mb-6">

          <input type="number" class="form-control" name="unit" placeholder="Unit">

        </div>

        <button type="submit" class="btn btn-primary mb-2">Update</button>

      </form>

</div>

</div>

{% endblock content %}

#manage.py

"""Django's command-line utility for administrative tasks."""

import os

import sys

def main():

    """Run administrative tasks."""

    os.environ.setdefault('DJANGO\_SETTINGS\_MODULE', 'bloodbankmanagement.settings')

    try:

        from django.core.management import execute\_from\_command\_line

    except ImportError as exc:

        raise ImportError(

            "Couldn't import Django. Are you sure it's installed and "

            "available on your PYTHONPATH environment variable? Did you "

            "forget to activate a virtual environment?"

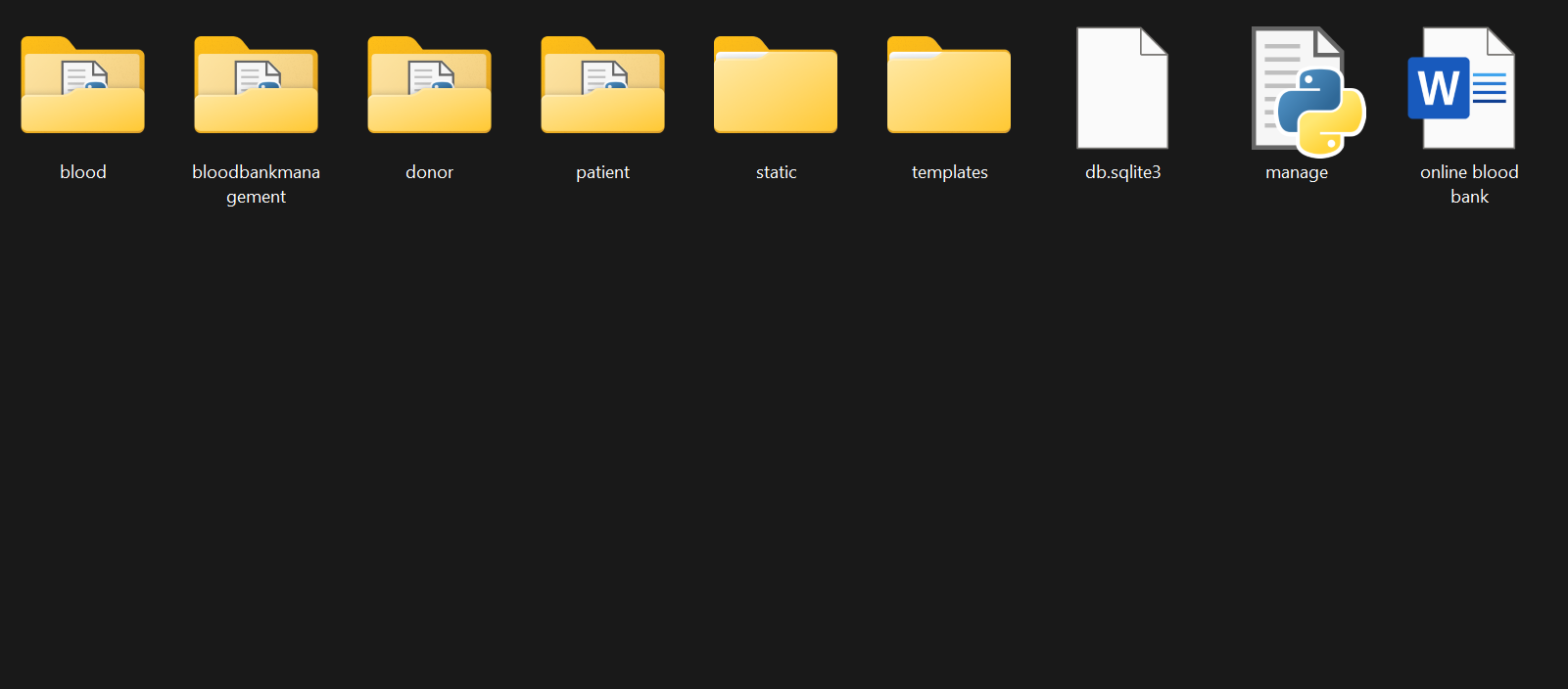
        ) from exc

    execute\_from\_command\_line(sys.argv)

if \_\_name\_\_ == '\_\_main\_\_':

    main()

**File Structure**

****

**CHAPTER 7**

**CONCLUSION**

To conclude about the description of this project. The project is developed using html, css , js , Django and SQL server.

The main aim of this project is to establish an online blood bank management system which helps in managing blood bank operations efficiently.

This proposed system is very much attainable and usable. This online system is provided with the awareness regarding blood donation which will increase the number of voluntary blood donors. It consists of central repository which gives the information regarding various blood donation camps available along with the details associated. Voluntary donors can register in the system to post their donation and customer in need of blood can register online and find suitable donor, hence making the whole process friendly and handy.

By considering these problems of blood bank and patients we have developed this online project which is going to be very much helpful and friendly for the patients in finding required blood unit.Some people are lucky enough to have vital amount of blood in their body, where some are not. This project gives the opportunity to donors to gift a life by donating blood to the needy.  
This crucial and supportive project is developed using Django and SQL