#### BLOOD BANK MANAGEMENT SYSTEM

#### REPORT

**BACHELOR OF TECHNOLOGY IN**

**INFORMATION & TECHNOLOGY ENGINEERING**

**Course Name: Java Programming Course Code: 19PC2IT03**

**Submitted By**

|  |  |
| --- | --- |
| K. SHARWANI | Reg. No. 19071A1290 |
| N. S. AKHILA | Reg. No. 19071A1298 |
| P. VINAYA MADHULIKHA | Reg. No. 19071A12A4 |
| P. NIPPUN | Reg. No. 19071A12A5 |
| V. HARVENI | Reg. No. 19071A12B9 |

**UNDER THE SUPERVISION OF**

**G. RAJESH KUMAR**

**VNRVJIET**



**VALLURUPALLI NAGESWARA RAO VIGNANA JYOTHI INSTITUTE OF**

**ENGINEERING & TECHNOLOGY**

NBA Accredited CE, EEE, ME, ECE, CSE, EIE, IT -B. Tech Programs Approved by AICTE, New Delhi, Affiliated to JNTUH



VIGNANA JYOTHI NAGAR, BACHUPALLY, NIZAMPET (S.O.), HYDERABAD 500090. TELANGANA, INDIA.

**BLOOD BANK MANAGEMENT SYSTEM**

**ABSTRACT**

Blood Bank management system is to provide an ability to maintain the online blood inventory, where the donors can able to fill the details to get the slot for donation and needed people can search the inventory for the availability of specific blood group in the given area, submit the request to reserve it.

The Blood Bank Management System (BBMS) is a desktop application that is designed to store, process, retrieve and analyze information concerned with the administrative and inventory management within a blood bank. This project aims at maintaining all the information pertaining to blood donors, different blood groups available in each blood bank and help them manage in a better way. The donors who are interested in donating blood must register in the system.

The requirement of the blood must be requested, and we give the information of the donor. The administrator can update their status whether they are available or not. It is developed in a manner that is easily manageable, time saving and relieving one from manual work.

**INTRODUCTION**

The population of the world is multiplying with each coming year and so are the diseases and health issues. With an increase in the population there is an increase in the need of blood. The growing population of the world results in a lot of potential blood donors. But in spite of this not more than 10% of the total world population participates in blood donation. With the growing population and the advancement in medical science the demand for blood has also increased. Due to the lack of communication between the blood donors and the blood recipients, most of the patients in need of blood do not get blood on time and hence lose their lives. There is a dire need of synchronization between the blood donors and hospitals and the blood banks. This improper management of blood leads to wastage of the available blood inventory. Improper communication and synchronization between the blood banks and hospitals leads to wastage of the blood available. These problems can be dealt with by automating the existing manual blood bank management system. A high-end, efficient, highly available and scalable system has to be developed to bridge the gap between the donors and the recipients and to reduce the efforts required to search for blood donors.

Blood Bank Management System (BBMS) is a management system that assists the information of blood bag during its handling in the blood bank. With this system, the user can look in the result of blood test that has been conducted to each of the blood bag received by the blood bank. The result of test will indicate whether the blood bag can be delivered to patient or not. From this system, there are several types of report that can be generated such as blood stock report, donor’s gender report and the total of blood donation according to months and year. The system also can give the information to the donor about blood analysis test result for each time the donor makes contribution. Hence, BBMS will make the blood bank stock more systematic and manageable.

* Blood bank management system is a web-based platform to provide an easy and handy to search for a blood donor according to the location.
* It has 3 different modules
  + Blood bank to monitor the demand and supply
  + Donors can register voluntarily by providing their details and availability.
  + Blood recipients can search for the availability and submit their request.
* System will have ability to send the message to relevant donors based on the demand.

**EXISTING SYSTEM:**

The existing Blood Bank Management System follows the following process starting from the donation of blood from the donor to its transfusion to the patient:

Step One: The Donation

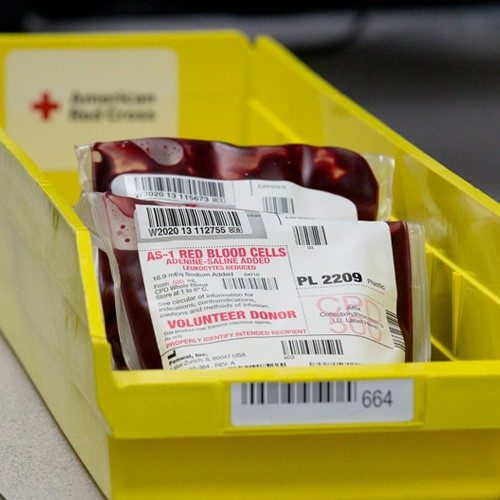
You arrive for your blood donation appointment. For a whole blood donation, about 1 pint of blood is collected; several small test tubes of blood are also collected for testing. The test tubes go to the lab.

Step Two: Processing

At the processing center, information about the donation is scanned into a computer database. Each component is packaged as a “unit,” a standardized amount that doctors will use when transfusing a patient.

Step Three: Testing

* In parallel with Step 2, the test tubes arrive at a testing laboratory, test for infectious diseases are performed and Test results are transferred electronically to the processing center within 24 hours

Step Four: Storage

When test results are received, units suitable for transfusion are labelled and stored.

 Step Five: Distribution

Blood is available to be shipped to hospitals 24 hours a day, 7 days a week.

Step Six: Transfusion

Physicians determine whether a patient requires a transfusion and, if so, which type.

Blood transfusions are given to patients in a wide range of circumstances, including serious injuries (such as in a car crash) surgeries, child birth, anemia, blood disorders, cancer treatments, and many others.

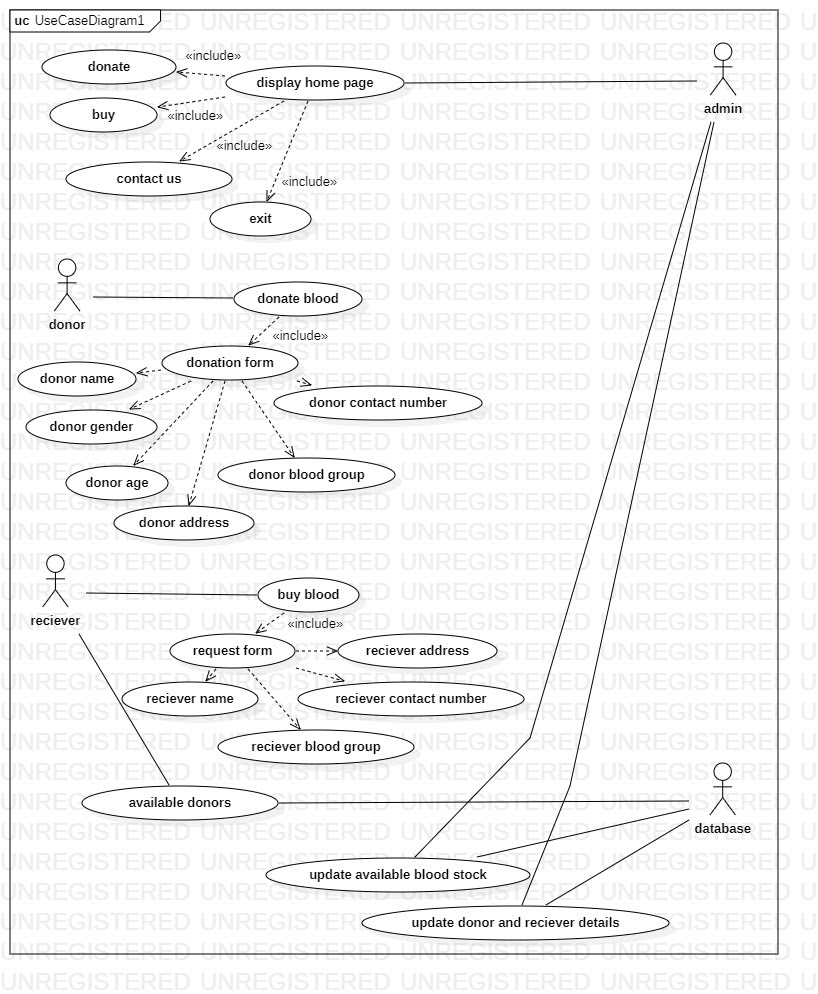
**RELATED STUDY**

* Initiated discussions with blood bank to identify the minimum set of attributes needed in donor form, requestor form.
* Finalized the dashboard needed to monitor the availability and requests.
* Identified technologies as Java to develop the front end and oracle to store the data in the backend.
* Initial version of the platform works in desktop mode with Java Swing. The same can be extended to Java/JSP to web version.

**PROPOSED ARCHITECTURE**

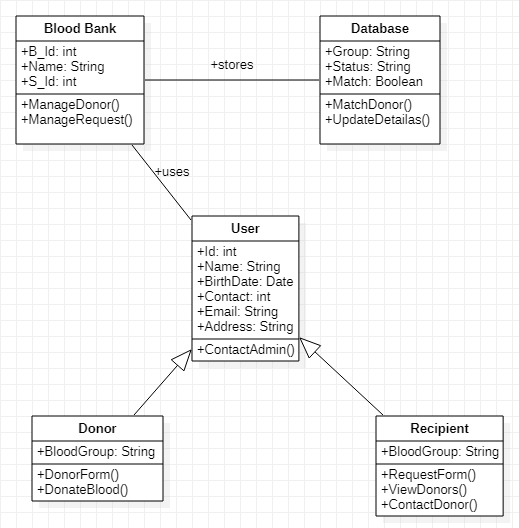
* Blood Bank Management system consists of five screens which are home page, donor form, search inventory, request form and dashboard.
* Home page appears on screen as soon as the application launches.
* If the user wants to donate blood, then he/she can click on donate button which directs to the donor form where he/she can register to donate blood by filling the details and availability.
* The user can click on Search Inventory button to see the availability of blood for the given blood group, which will show the list of available packets and immediate available donors.
* Once User see the availability, they can submit the request for particular group and put a reservation on it. The user can exit the application by pressing exit button.
* We have developed the following UML diagrams as part of our design phase.
  + Use case diagram
  + Class Diagram
  + Sequence and Collaboration diagram for donor
  + Sequence and Collaboration diagram for recipient

**Use case Diagram**

****

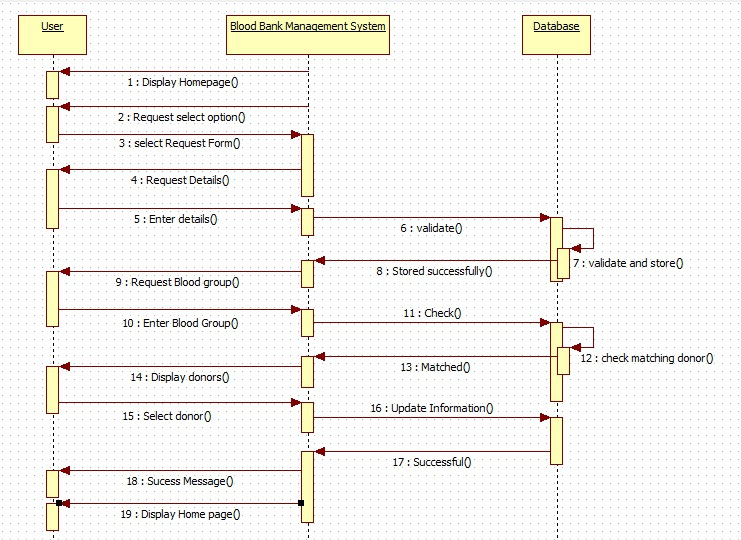
A use case diagram is used to represent the dynamic behavior of a system. It encapsulates the system's functionality by incorporating use cases, actors, and their relationships. The above use case diagram has the donor, admin, recipient, database as actors who communicate among themselves to carry out the functions mentioned in the use cases. It depicts what all the actor can do in the system like donor can donate blood by filling the donate form and recipient can request and receive blood by filling the request form.

**Class Diagram**



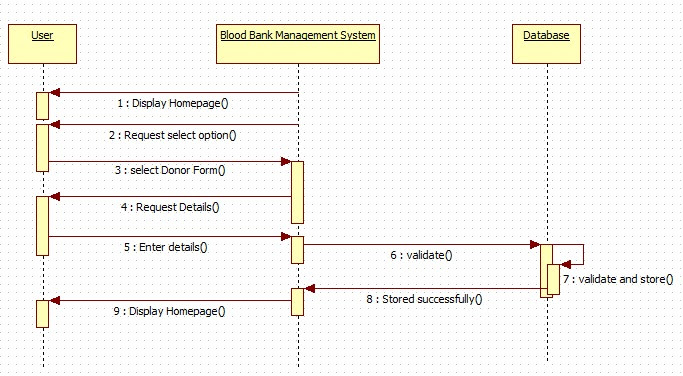
The class diagram depicts a static view of an application. A class consists of its objects, and also it may inherit from other classes. This class diagram contains User, Blood Bank System, Database as classes. The user can be further categorized into donor and recipient. Both the users have similar attributes but the functions they perform are different. The donor submits the Donate Form, thereby providing his acceptance and details for donation of blood. When the recipient requests blood, the database and the system checks if there are donors of matching blood groups, and if so, provide the details of donor to the recipient.

**Sequence Diagram for donor**



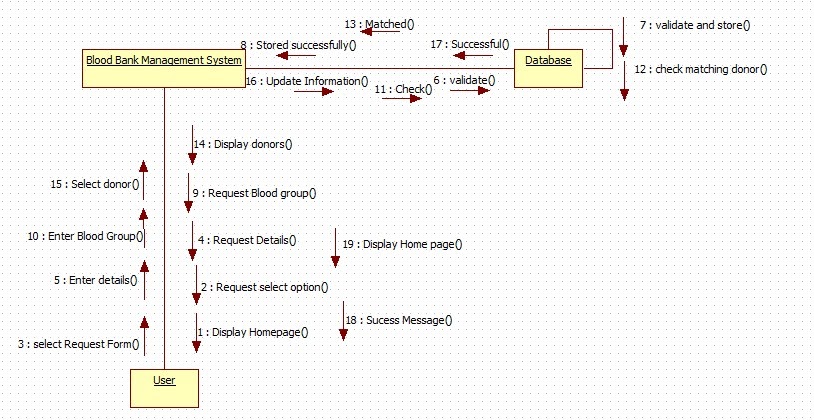
A sequence diagram represents the flow of messages in the system. This is the sequence diagram which represents the sequence of operations performed by the donor who is willing to donate his/her blood to recipient. The donor opens the application, and the homepage with different options is display. He/She clicks on the donate option which redirects him/her to a donate form where the donor needs to fill the details like name, address, blood group and accept the terms and conditions before submitting. This information is stored in the database and used for further blood transfusions. This process is clearly expressed in the sequence diagram.

**Sequence Diagram for Recipient**

****

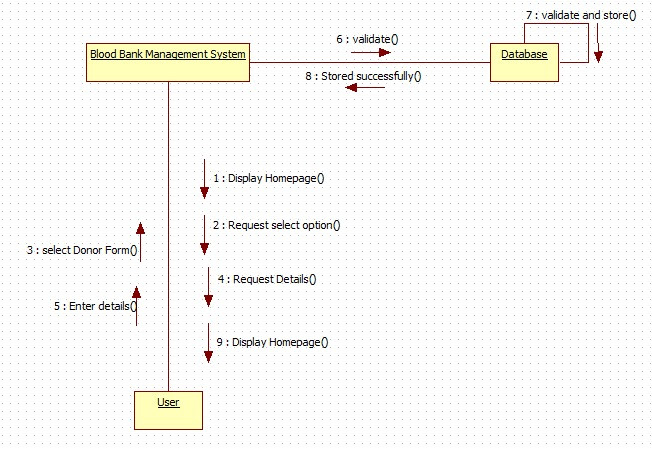
The sequence diagram for Recipient depicts flow of messages between the system and recipient who is in need of blood. The recipient selects the “Request blood” option in homepage which redirects to the request form and the required details should be filled by recipient which will be stored in the database. Then Recipient can click on show list which shows the donors with same blood group. The details of donors are made available and hence may contact any person in the list which will be intimated to donor automatically by clicking contact button**.**

**Collaboration diagram for donor**

****

The collaboration diagram is used to show the relationship between different objects in a system. Instead of showing the flow of messages like in sequence diagram, it depicts the architecture of the object residing in the system. The collaboration diagram for the donor depicts all the steps or operations performed by one entity on other. For example, it shows all the functions performed by the user on the blood bank management system.

**Collaboration diagram for Recipient**

****

The collaboration diagram for recipient shows the flow of messages requested by recipient and sent by database. It depicts the process of how the recipient can receive blood and actions performed by recipient and database as messages. The recipient can also contact the donor through this process.

**IMPLEMENTATION**

**We have implemented our project using Java Swing. We have also used Java AWT in some parts.**

AWT and Swing are used to develop window-based applications in Java. AWT is an abstract window toolkit that provides various component classes like Label, Button, TextField, etc., to show window components on the screen. All these classes are part of the Java.awt package.

On the other hand, Swing is the part of JFC (Java Foundation Classes) built on the top of AWT and written entirely in [Java](https://www.javatpoint.com/java-tutorial). The javax.swing API provides all the component classes like JButton, JTextField, JCheckbox, JMenu, etc.

AWT components are platform dependent whereas swing components are platform independent.

The hierarchy of java swing API is given below

**JButton**

The JButton class is used to create a labeled button that has platform independent implementation. The application results in some action when the button is pushed. It inherits the AbstractButton class.

**JLabel**

The object of the JLabel class is a component for placing text in a container. It is used to display a single line of read only text. The text can be changed by an application but a user cannot edit it directly. It inherits the JComponent class.

**JTextField**

The object of a JTextField class is a text component that allows the editing of a single line text. It inherits the JTextComponent class.

# JTextArea

The object of a JTextArea class is a multi-line region that displays text. It allows the editing of multiple line text. It inherits JTextComponent class

**JPasswordField**

The object of a JPasswordField class is a text component specialized for password entry. It allows the editing of a single line of text. It inherits the JTextField class.

# JCheckBox

The JCheckBox class is used to create a checkbox. It is used to turn an option on (true) or off (false). Clicking on a CheckBox changes its state from "on" to "off" or from "off" to "on''. It inherits the JToggleButton class.

# JRadioButton

The JRadioButton class is used to create a radio button. It is used to choose one option from multiple options. It is widely used in exam systems or quizzes.

It should be added in ButtonGroup to select one radio button only.

# JComboBox

The object of Choice class is used to show a popup menu of choices. Choice selected by the user is shown on the top of a [menu](https://www.javatpoint.com/java-jmenuitem-and-jmenu). It inherits the JComponent class.

# JTable

The JTable class is used to display data in tabular form. It is composed of rows and columns.

# JFrame

The javax.swing.JFrame class is a type of container which inherits the java.awt.Frame class. JFrame works like the main window where components like labels, buttons, text fields are added to create a GUI.

**Java AWT** (Abstract Window Toolkit) is *an API to develop Graphical User Interface (GUI) or windows-based applications* in Java.

The java.awt [package](https://www.javatpoint.com/package) provides [classes](https://www.javatpoint.com/object-and-class-in-java) for AWT API such as [TextField](https://www.javatpoint.com/java-awt-textfield), [Label](https://www.javatpoint.com/java-awt-label), [TextArea](https://www.javatpoint.com/java-awt-textarea), RadioButton, [CheckBox](https://www.javatpoint.com/java-awt-checkbox), [Choice](https://www.javatpoint.com/java-awt-choice), [List](https://www.javatpoint.com/java-awt-list) etc.

**The Java ActionListener** is notified whenever you click on the button or menu item. It is notified against ActionEvent. The ActionListener interface is found in java.awt.event [package](https://www.javatpoint.com/package).

It has only one method: actionPerformed().

The actionPerformed() method is invoked automatically whenever you click on the registered component.

**CODE:**

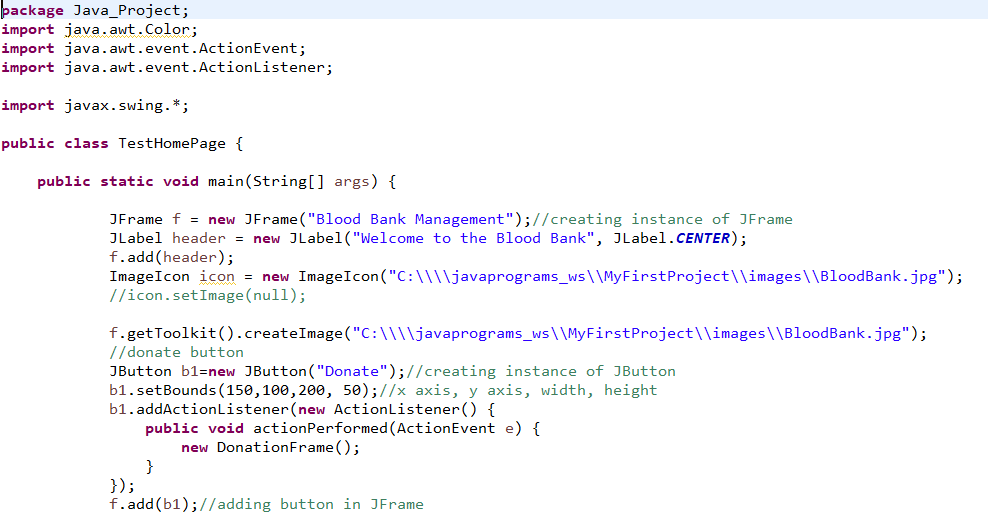
These are the snippets of some interfaces of our project.

**HomeScreen**

This is the code snippet of home screen. This is the first screen visible to the user which displays different options such as

* Request Blood
* Donate Blood
* Dashboard
* Exit

The Donate Blood redirects the user willing to donate blood to fill the donate form. The Request Blood option helps the recipients in need of blood to find the suitable donors for their blood group and thereby contact them. The dashboard gives an overview of the system and provides contact details of the Blood bank management system. addActionListener of AWT is used to invoke different screens when the buttons are clicked.

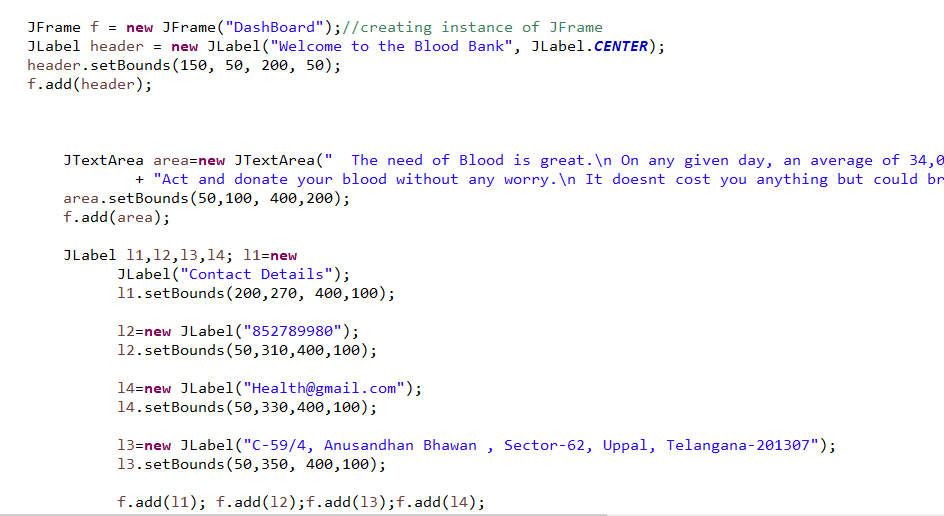




**Dashboard**

This is the code snippet of Dashboard. This is provides the information to the user about the system and its functionality. It also provides its contact details to the users in case of any queries or to provide feedback.

This is implemented using Swing where all the description about the system is provided using a TextArea and all the contact details are provided using JLabel.



**Donate Form**

This system automates the blood donation process by releasing a donation form to the donors.

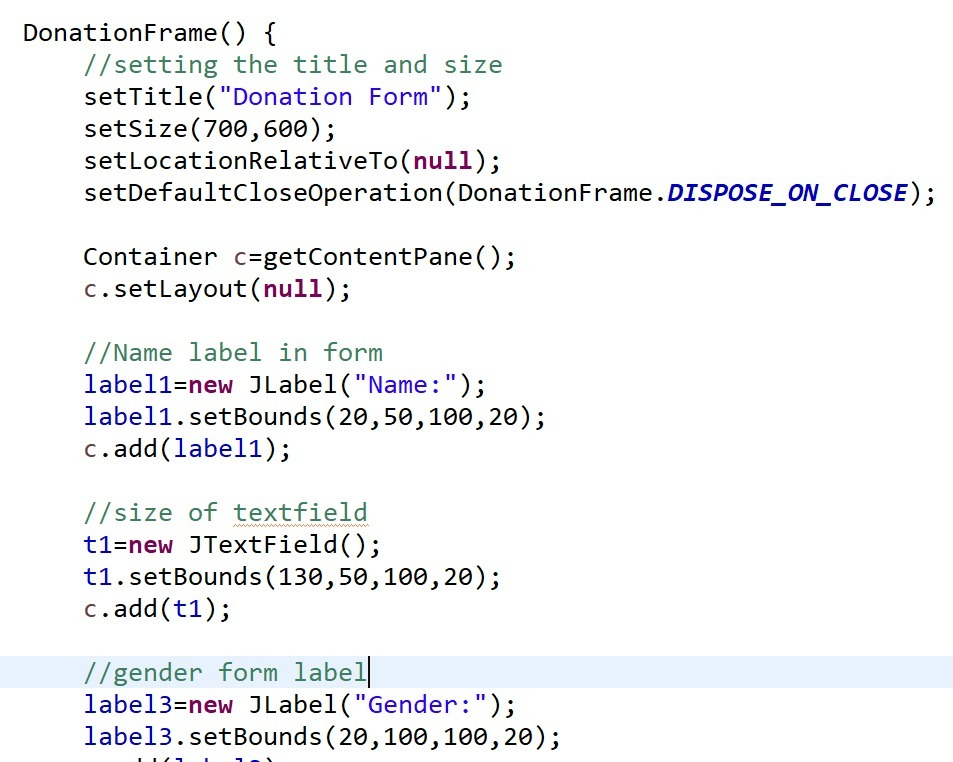
To donate blood, the donor must register his details in the donation form. There are some factors that are needed to be considered in order to accept blood from a person. One of the factor is age and health issues. The donation form asks the donor to enter his/her name, gender, age, contact number, address, and his/her blood group and submit the form by accepting the terms and conditions. The details are not registered unless the donor accepts the terms and conditions. Once the details are submitted, they are made available to the recipients.

The *donation form* is created using a java class that extends the JFrame class and implements ActionListener interface.

The labels ‘Name’, ’Gender’, ’Age’, ’Contact’, ’Address’, ’Blood Group’ displayed on the donation form are created using JLabel().

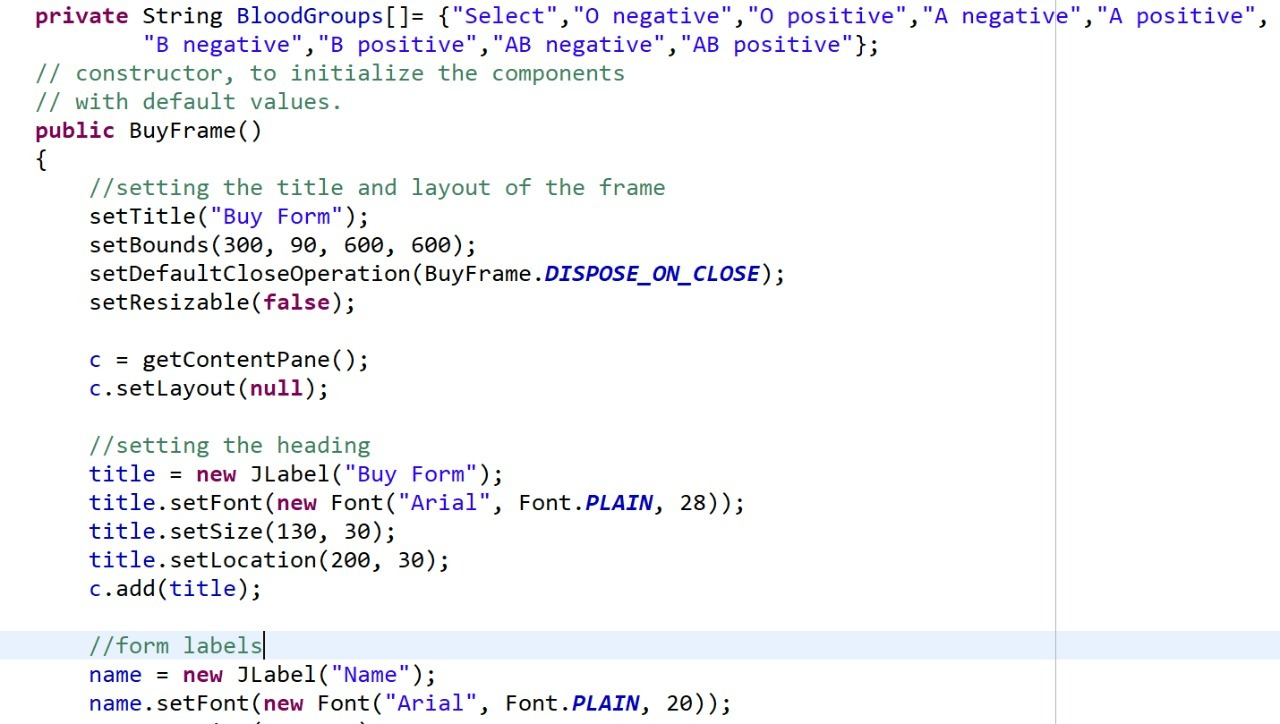
The donor can enter his/her details in the fields displayed on donation form.

Fields associated with ‘Name’, ‘Age’, ‘Contact’ labels are created using JTextField().



**Request Form**

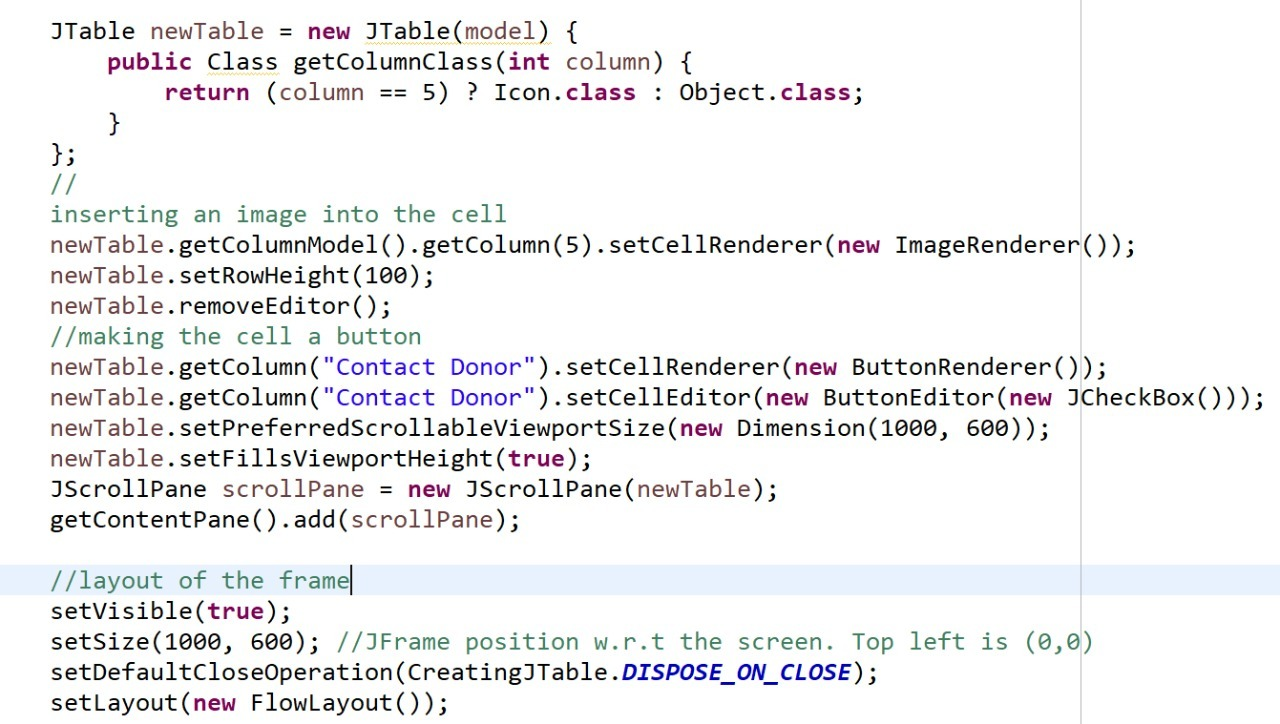
The request form enables the users in need of blood to find the donors of suitable blood group and thereby contact them. This consists of fields like name, blood group, address which are required to be filled by the user. There is a button named showlist which shows the available donors list of that particular blood group along with their contacts. The submit button submits the request. An addActionListener is added to connect the home page.



**Donor Table Implementation:**

In the implementation of the table, JTable class was used to display the table in tabular form. It comprises rows and columns that intersect to form cells that store data. JTable does not contain or cache data; it is simply a view of your data. The advantage of the constructors used is that they are easy to use.

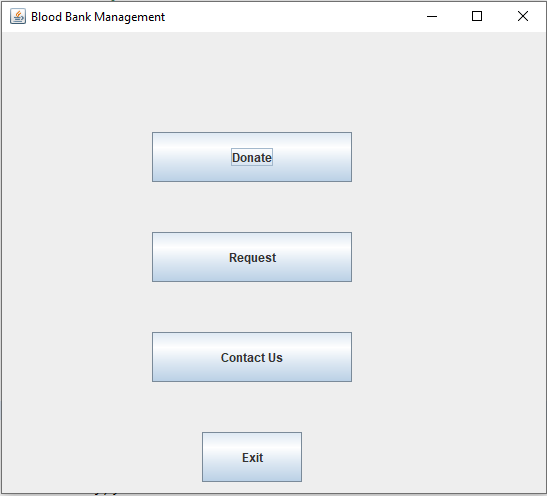
This table displays the list of users along with their details whenever a recipient fills in his/ her details.



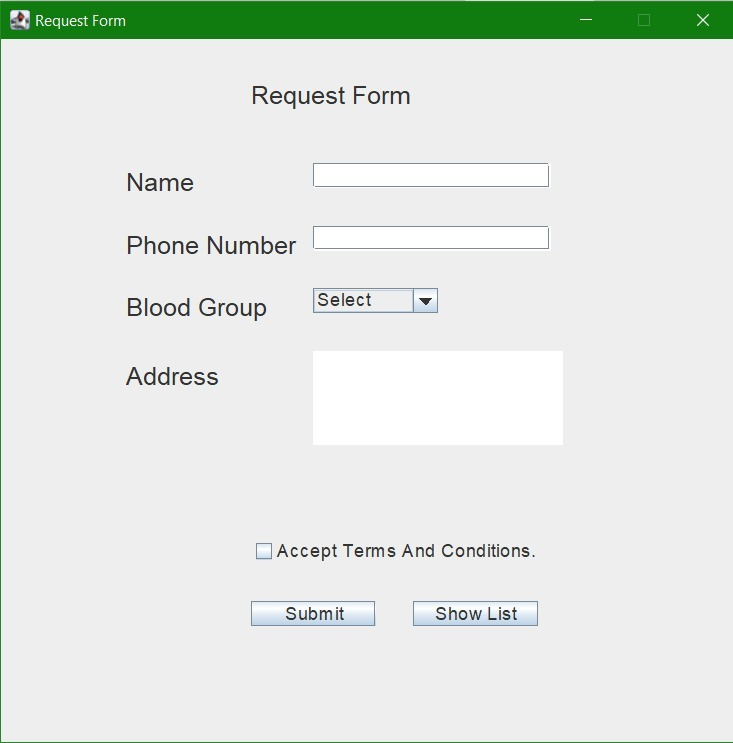
**RESULTS AND DISCUSSION:**

The following are some output screens of our project which includes home page, donate form, request form, registered users page and dashboard.

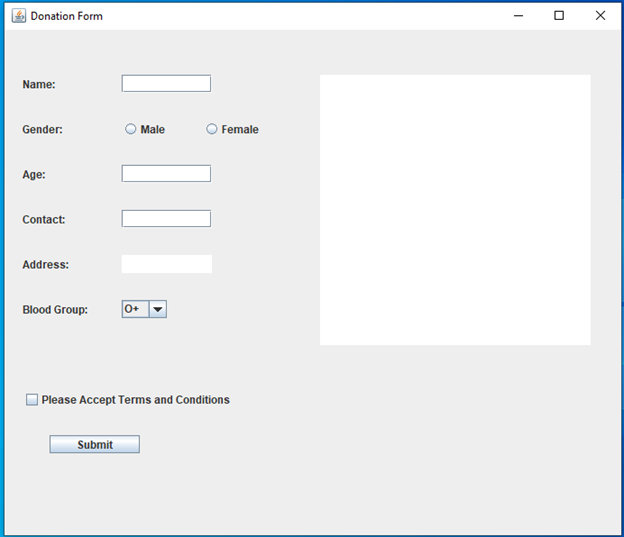
**Home Screen**



**Request Form**



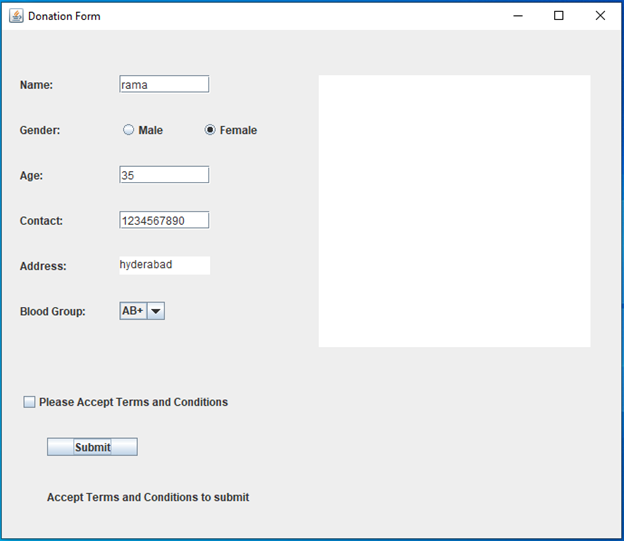
**Donation Form**



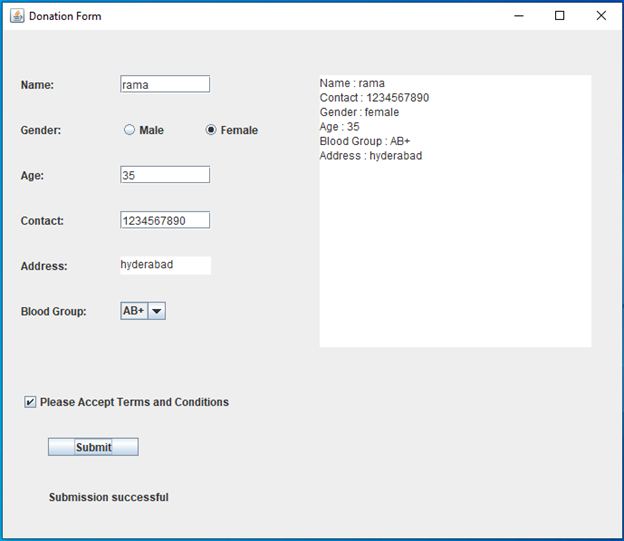
The donor submits his/her details by clicking on the ‘Submit’ button created using JButton.

The system gets notified when the submit button is clicked as it is associated with the addActionListener() method.

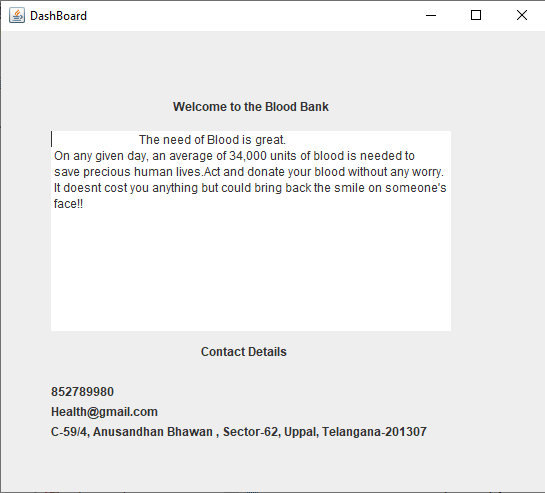
If the donor submits his/her details without accepting the terms and conditions a message ‘Accept Terms and Conditions to submit’ is displayed on the donation form.



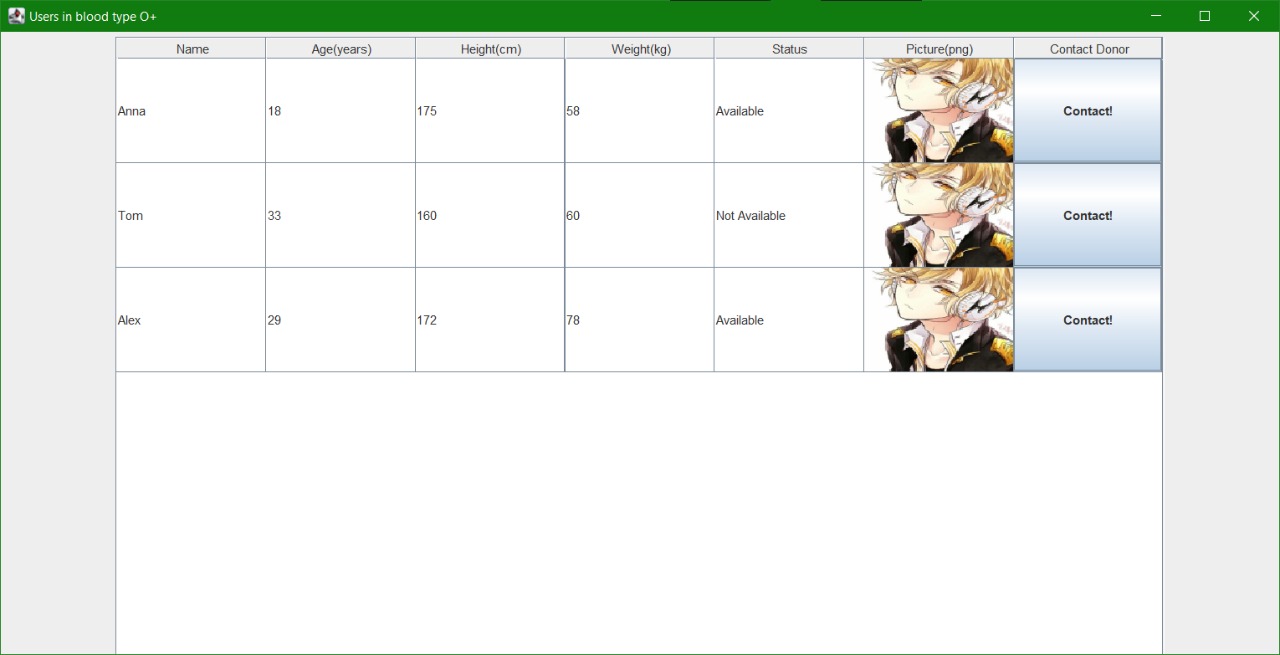
On successful submission a message ‘Submission successful’ along with the registered donor details are displayed on the donation form.



Dashboard



Registered Users in blood group O+



**CONCLUSION:**

* In this project, we have developed a simple Blood Bank management system that aims to provide an ability to maintain the online blood inventory.
* Here the donors can able to fill the details to get the slot for donation and needed people can search the inventory for the availability of specific blood group in the given area.
* Current system works in desktop mode only and is available at the blood bank.
* In order to use it remotely, we need to make it as a web-based application. Even we can extend it to mobile related platforms.

**REFERENCES:**

* [**https://www.javatpoint.com/java-swing**](https://www.javatpoint.com/java-swing)
* [**https://www.freeprojectz.com/**](https://www.freeprojectz.com/)
* [**https://docs.oracle.com/javase/8/docs/api/index.html?javax/swing/package-summary.html**](https://docs.oracle.com/javase/8/docs/api/index.html?javax/swing/package-summary.html)