# A Project Report On

# “Blood bank Management System”

## TABLE OF CONTENTS

| Sr.No | Topic | Remark |
|---|---|---|
| 1 | Abstract | |
| 2 | Existing System | |
| 3 | Proposed System | |
| 4 | Purpose of the System | |
| 5 | Hardware/Software Specifications | |
| 6 | Project Screenshot and Code | |
| 7 | Future Scope | |
| 8 | Conclusion | |
| 9 | References | |

# ABSTRACT

Human blood is an essential element of human life with no substitute. Voluntaryblood donors are the cornerstone of a safe and adequate supply of blood and blood products. The safest blood donors are voluntary, non-remunerated blood donors from low-risk populations. Blood donors in India, as across the globe,are of three types; voluntary donors, replacement donors and professional donors. Most donations are as a result of replacement donations, which are non- remunerated donations, provided by the relatives of patients. Professional donors are those who donate blood in exchange for money. Replacement and professional donors may be compelled to donate blood, though their health conditions are unsuitable to donate blood. They do not help maintaining a stock of blood for emergency situations. Moreover, they do not provide rare blood groups. These points are indicating the need and importance of voluntary blood donations. The success of blood donation camp depends on people who organize the camp and blood bank team. Aim: The aim of this report is to discuss about the importance of voluntary blood donation and to guide the camporganizer, medical officer, and also other team members involving in voluntary blood donation camp to organize the camp efficiently.

The Language used in this project are: JAVA, SQLDATABASE.

# EXISTING SYSTEM

The existing system and some of their drawbacks which force us to plan this whole ideaof developing online Blood bank management system.

- Suppose there is some patient who needs blood urgently then how you come toknow about this condition, you will not even able to reach to the patient.

- People will go from one blood bank to another to get the blood which is time-consuming and sometimes not able to reach at the time. In some blood banks,sometimes we get the blood but that is not sufficient so we need to search for another blood bank for more blood.

- If some person wants to donate the blood he/she need to come to the bank and need to fill the form then the first doctor will check his blood group after that hewill allow donating.

## DRAWBACKS OF THIS OF BLOOD DONATION CAMP SYSTEM

- This process is so much time consuming

- There is a threat that person who is donating blood may be doing for the moneyand donating several times which will cause weakness.

- There is no proper way of getting to know about that which blood group is presentwhere and in how much quantity.

- It might be the case that manager can take a commission as well as extra chargesto arrange the blood for rescue.

- So, above this is the whole process of the conventional way of donating blood tothe blood banks.

- Let us look the different aspects which we have designed to make the websitework properly.

- getting blood initially which has been improved by our system we can make it alittle bit more simple and fast process to automate it.

- We can automate this process by creating the application which will allow you touse these things in a fully functional way and the application will include the following entities (an entity is a real world object).

- The automated application help in following way:

- A person who wants to donate blood can register himself directly on the application and generate donate it as well as time, date and venue through themail.

- If the donator is not aware of policies then admin will make him contact with thisapplication and help him to complete the process.

- Admin will maintain the record and security by so that no user can donate morethan once in a month otherwise it will harm the person and cause weakness.

- Donator need not bother about his/her blood and without moving from his place person can donate blood as we will send our representative near to your location.

- No need of filling so many forms you just need to log in and take an appointment.

- Donator can check new requirements of blood which has been uploaded by theAdmin.

- The whole process of getting work done will become faster than before.

# PROPOSED SYSTEM

The proposed system (Blood Bank Management System) is designed to help the Blood Bank administrator to meet the demand of Blood by sending and/or serving the request for Blood as and when required.The proposed system gives the procedural approach of how to bridge the gap between Recipient, Donor, and Blood Banks. This Application will provide a common ground for all the three parties (i.e. Recipient, Donor, and BloodBanks) and will ensure the fulfillment of demand for Blood requested by Recipient and/or Blood Bank. The proposed system consists of the following goals and has the scope as follows:

a) Goals:

- To ease the process of blood donation and reception.
- To improve the existing system.
- To develop a scalable system.
- To be highly available

b) Scope:

- Ensure that all the functionalities of a manual blood bank are covered
- To include all the blood banks at least within a city.
- Make sure the program is simple and easy to use.

**Users of the system:**

- Admin
- Donor
- Seeker

## Modules:

The system is proposed to have the following modules along with thefunctional requirements.

## Admin module:

Admin will use this application to give the access to the person who wants to donate blood for the public welfare, the even admin will add new requirements regarding the blood group so that the user can easily access it and donate according to the demand.Whenever you want to donate the blood you just need to send the request to the admin and he will approve you according to the requirement and inform you the locationand contact number

## Donor:

This option will appear when user login into his account. So that he/she can donate the blood if want. For donating the blood user need to go for the following procedures whichare as follows:

Send the request to the admin by click on the donating blood

If you are a first-time user you need to fill the information otherwise we will extract thedata from your profile which you have created.

After getting the information you will get the message and email in which location andtime would be mentioned.

## Seeker :

This option of the website will help the user to get login and check the notifications if there is any regarding donating the blood not only this user can also donate the blood inadvance if he/she want to donate here user just need to take the appointment that wherehe wants to donate the blood, his request move to the admin and admin decide the location according to the need of blood.

# PURPOSE OF THE SYSTEM

- The main goal of the Blood Bank and Donor Management System project is to monitor Blood Bank data, Blood cells, Blood stock, DonorList.

- It manages all the Blood Bank, Donor, Blood stock data. The project isentirely administrative and therefore access is guaranteed only to the administrator.

- The project's aim is to develop an application system to minimize themanual work for Blood Bank, Donor, Blood Group management.

- It monitors all of the Blood Group information, Blood cells, Bloodsupply and Donor list

## SOFTWARE SPECIFICATIONS:

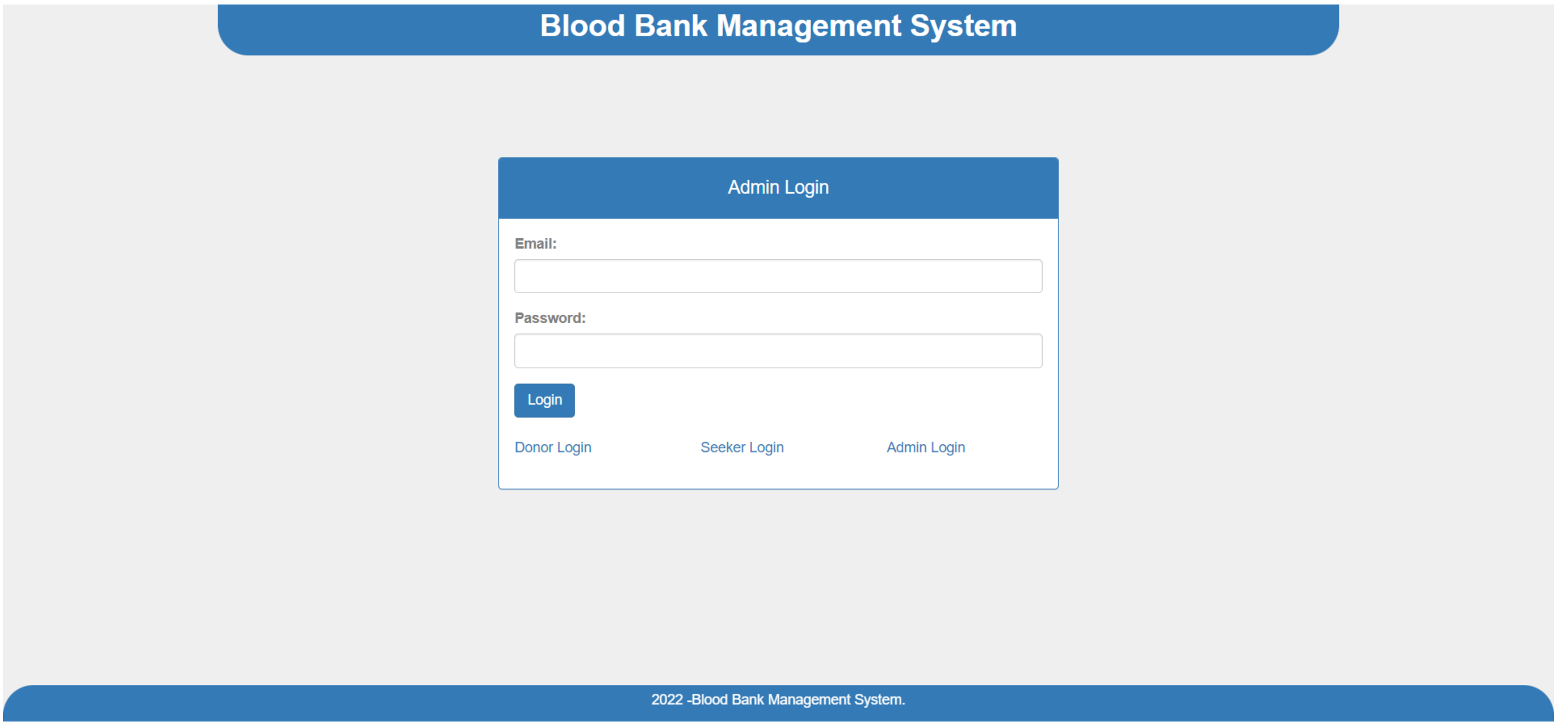
- ECLIPSE
- SQL DATABASE
- TOMCAT
- LANGUAGE: JAVA, MYSQL

## HARDWARE SPECIFICATIONS:

- CPU Intel Core 2 Duo E7300
- RAM- 1 GB (MIN)
- Hard Disk-160GB
- Keyboard: 108 Standard
- Mouse: Optical
- Monitor: 15’’ Colour Monitor

# PROJECT SCREENSHOT AND CODE

# ADMIN SCREENSHOTS



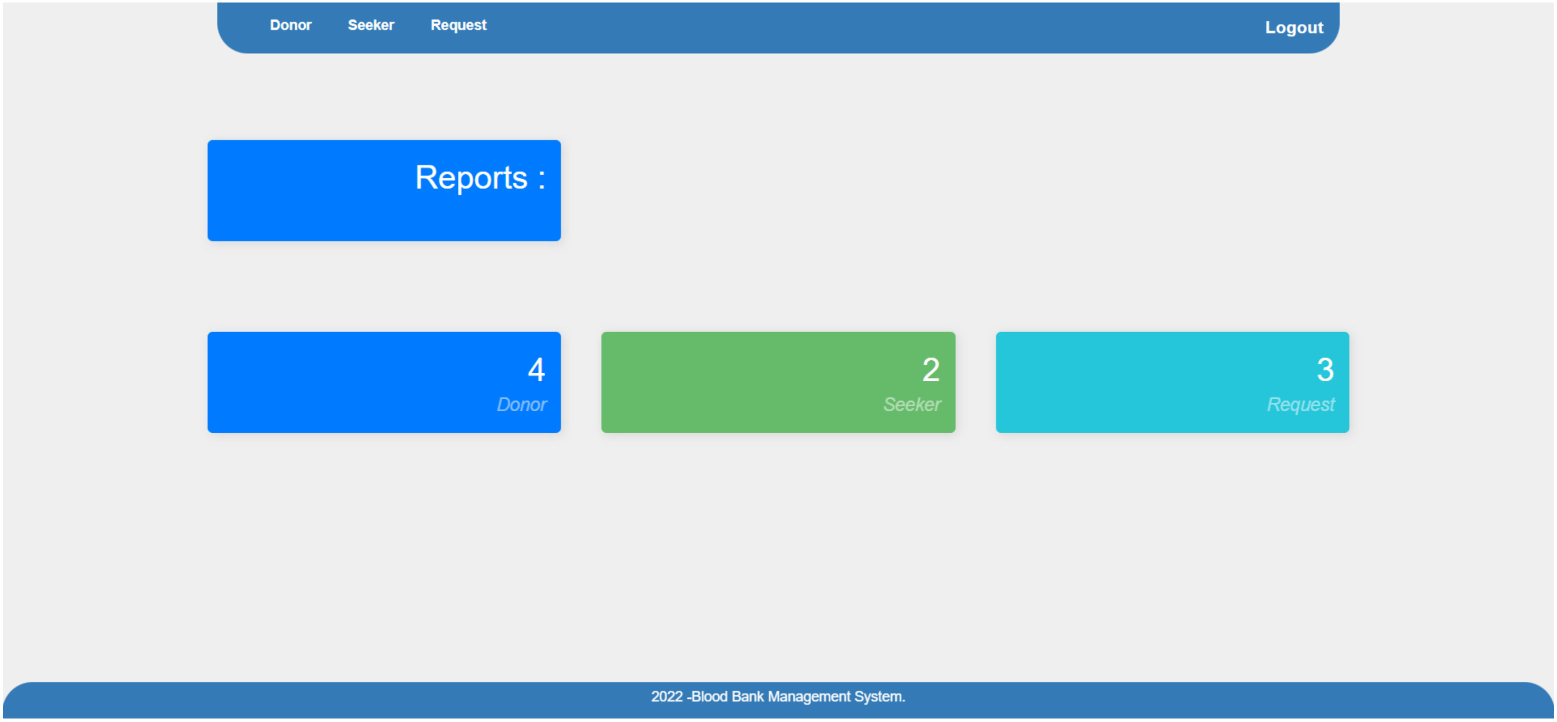
Blood Bank Management System

Admin Login

Email:

Password:

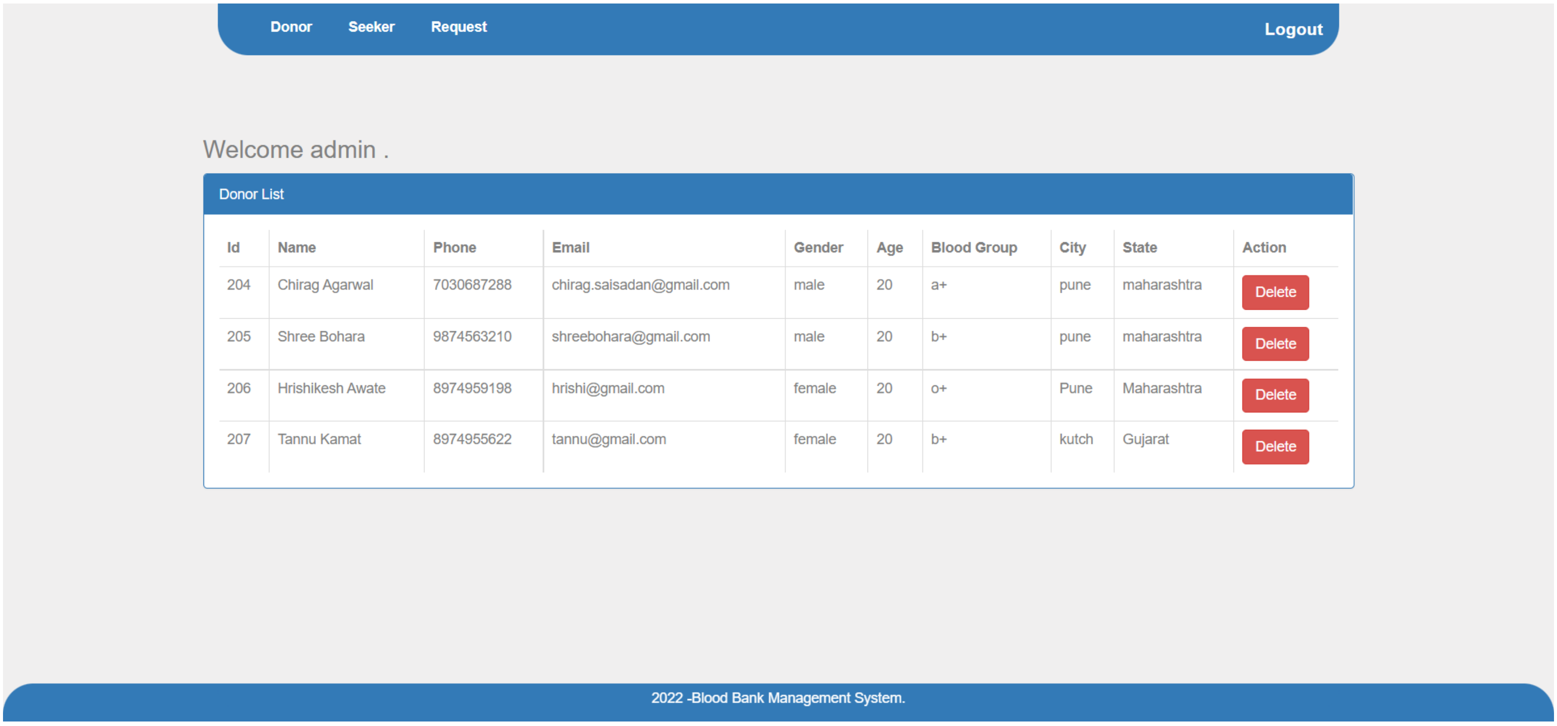
2022 -Blood Bank Management System.



Reports :

| | | |
|---|---|---|
| 4<br>Donor | 2<br>Seeker | 3<br>Request |

2022 -Blood Bank Management System.

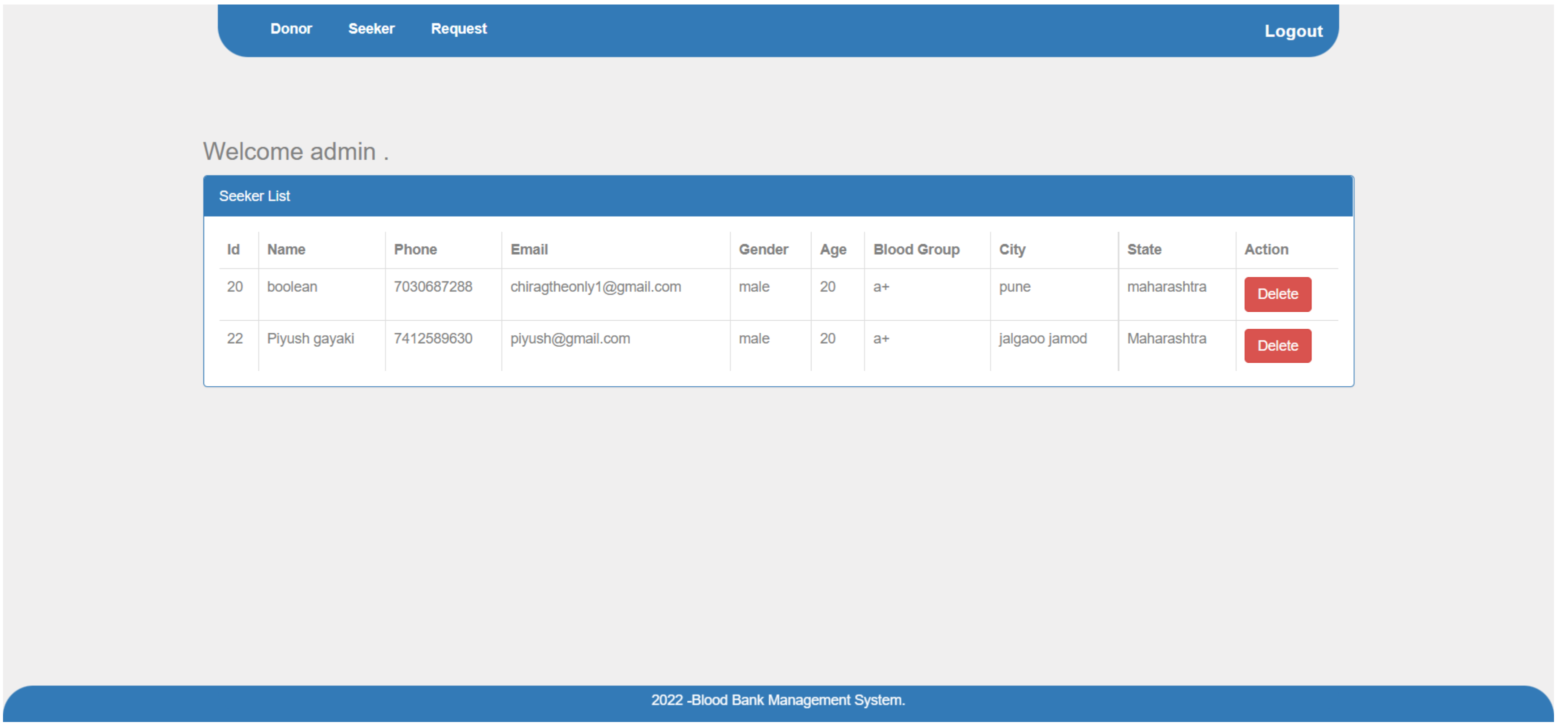


Welcome admin .

Donor List

| Id | Name | Phone | Email | Gender | Age | Blood Group | City | State | Action |
|---|---|---|---|---|---|---|---|---|---|
| 204 | Chirag Agarwal | 7030687288 | chirag.saisadan@gmail.com | male | 20 | a+ | pune | maharashtra | Delete |
| 205 | Shree Bohara | 9874563210 | shreebohara@gmail.com | male | 20 | b+ | pune | maharashtra | Delete |
| 206 | Hrishikesh Awate | 8974959198 | hrishi@gmail.com | female | 20 | o+ | Pune | Maharashtra | Delete |
| 207 | Tannu Kamat | 8974955622 | tannu@gmail.com | female | 20 | b+ | kutch | Gujarat | Delete |

2022 -Blood Bank Management System.

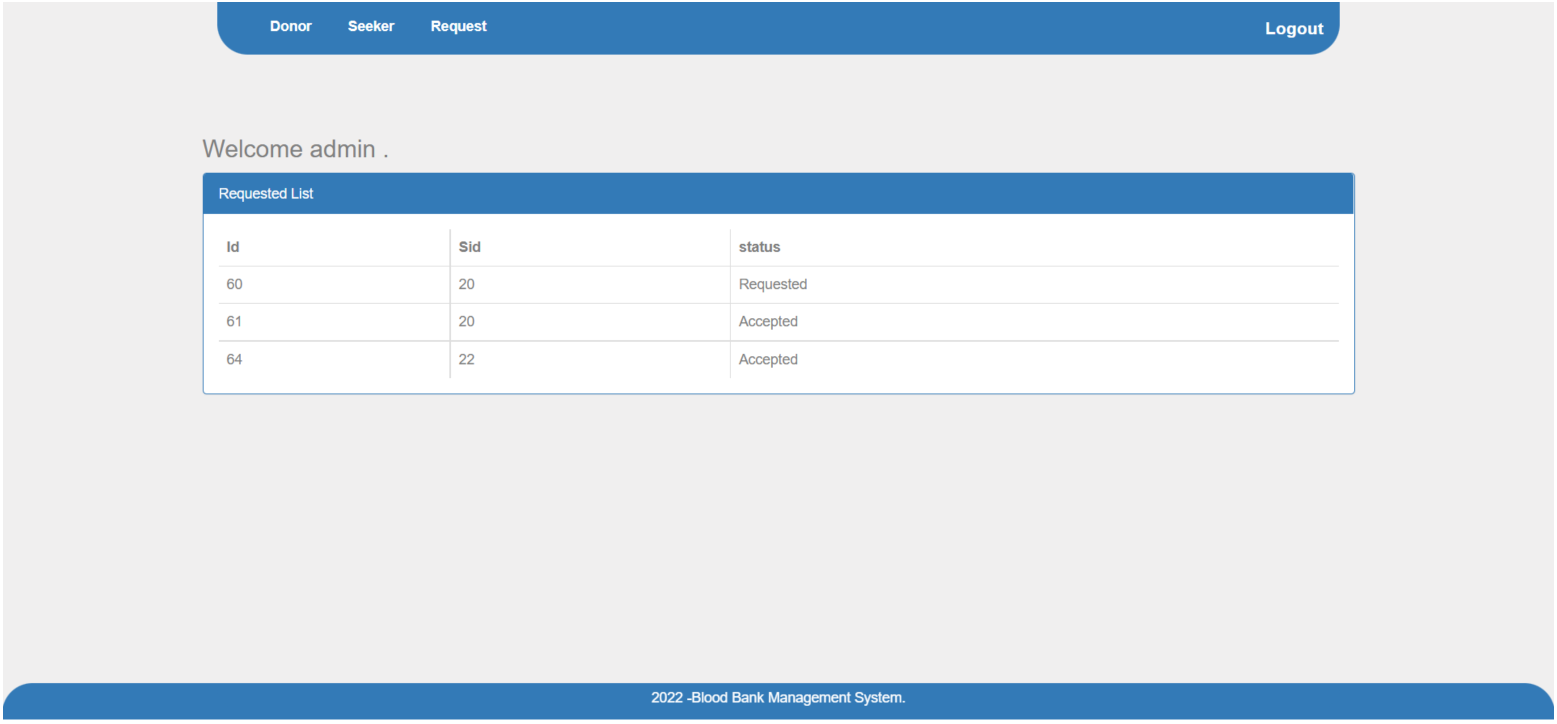


Welcome admin .

Seeker List

| Id | Name | Phone | Email | Gender | Age | Blood Group | City | State | Action |
|---|---|---|---|---|---|---|---|---|---|
| 20 | boolean | 7030687288 | chiragtheonly1@gmail.com | male | 20 | a+ | pune | maharashtra | Delete |
| 22 | Piyush gayaki | 7412589630 | piyush@gmail.com | male | 20 | a+ | jalgaoo jamod | Maharashtra | Delete |

2022 -Blood Bank Management System.



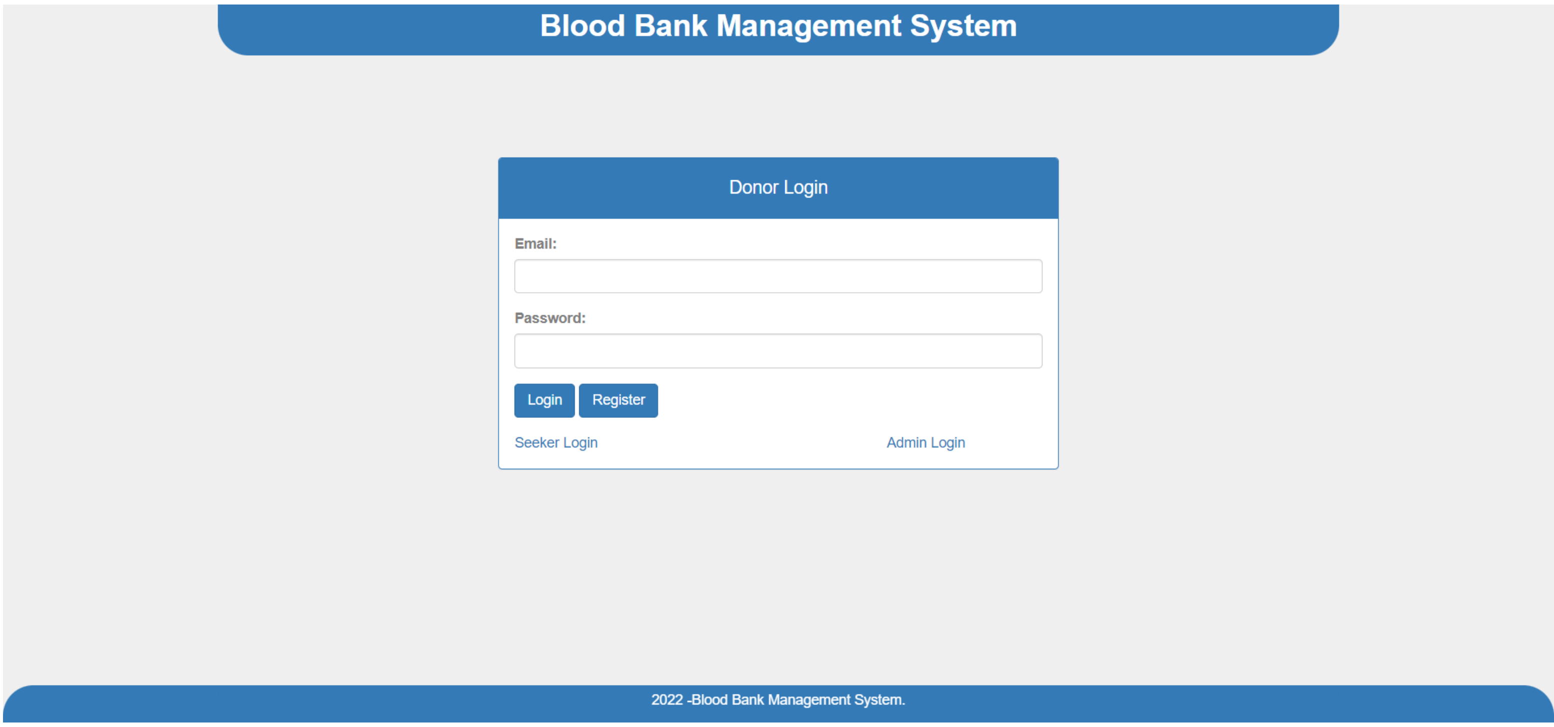
Welcome admin .

Requested List

| Id | Sid | status |
|---|---|---|
| 60 | 20 | Requested |
| 61 | 20 | Accepted |
| 64 | 22 | Accepted |

2022 -Blood Bank Management System.

# DONOR SCREENSHOTS



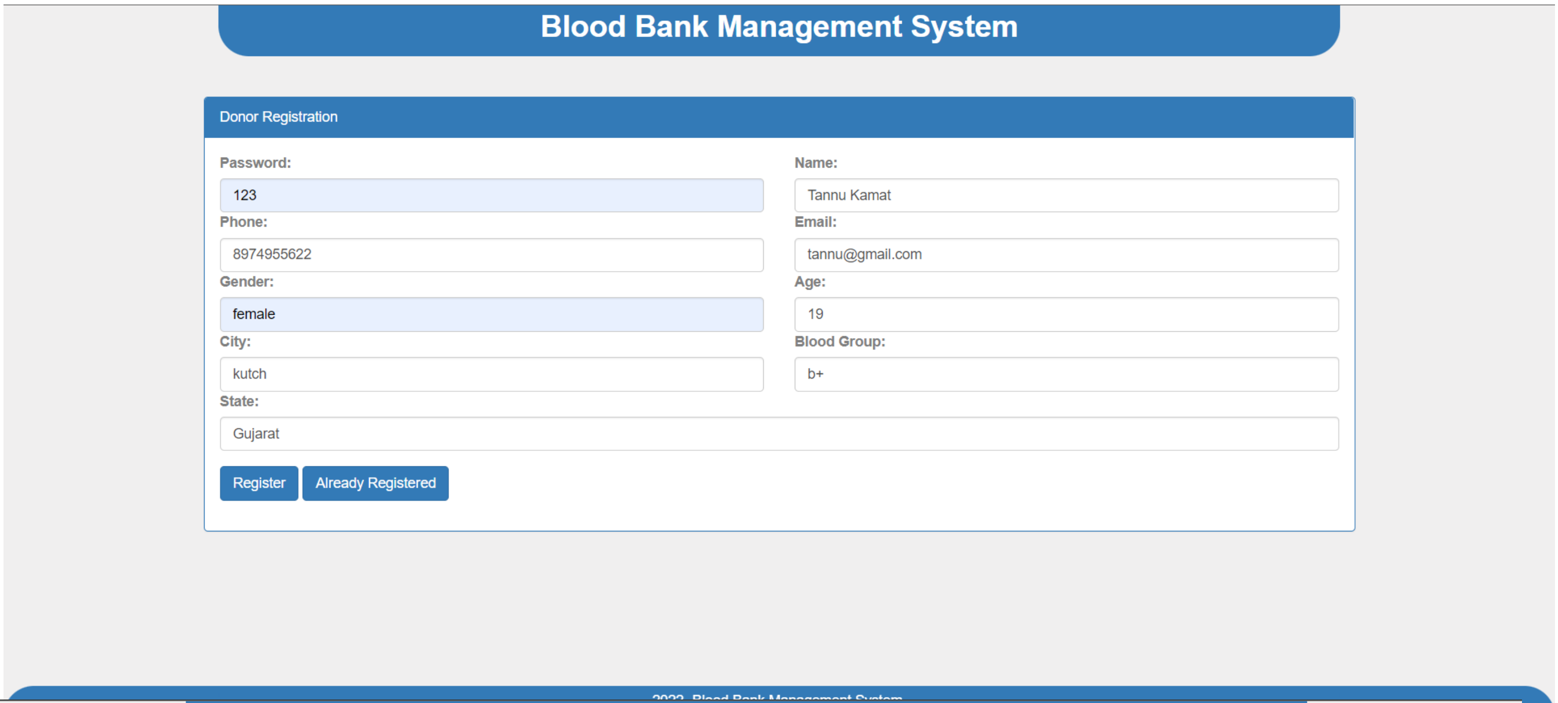
Blood Bank Management System

Donor Login

Email:

Password:

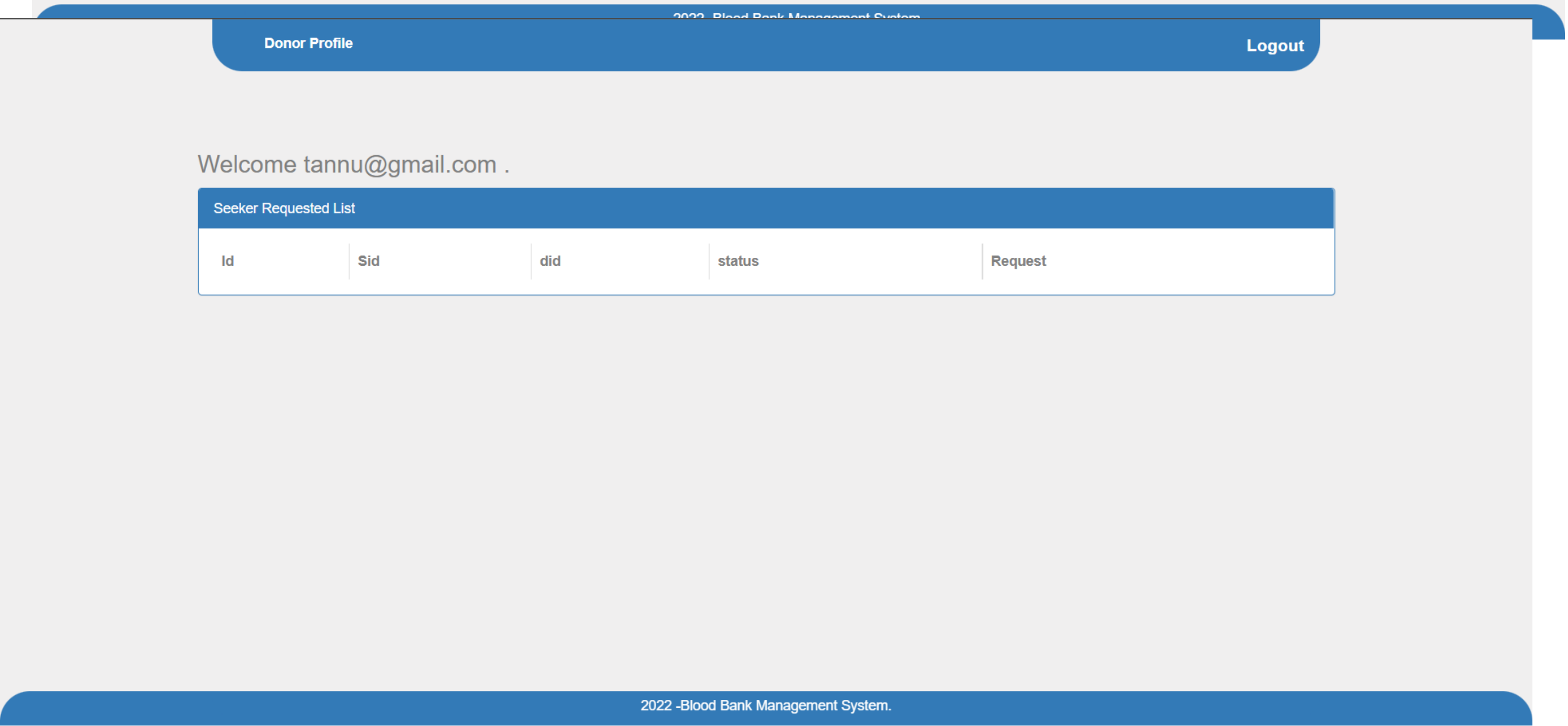
2022 -Blood Bank Management System.



Blood Bank Management System

Donor Registration

| | |
|---|---|
| Password: 123 | Name: Tannu Kamat |
| Phone: 8974955622 | Email: tannu@gmail.com |
| Gender: female | Age: 19 |
| City: kutch | Blood Group: b+ |
| State: Gujarat | |

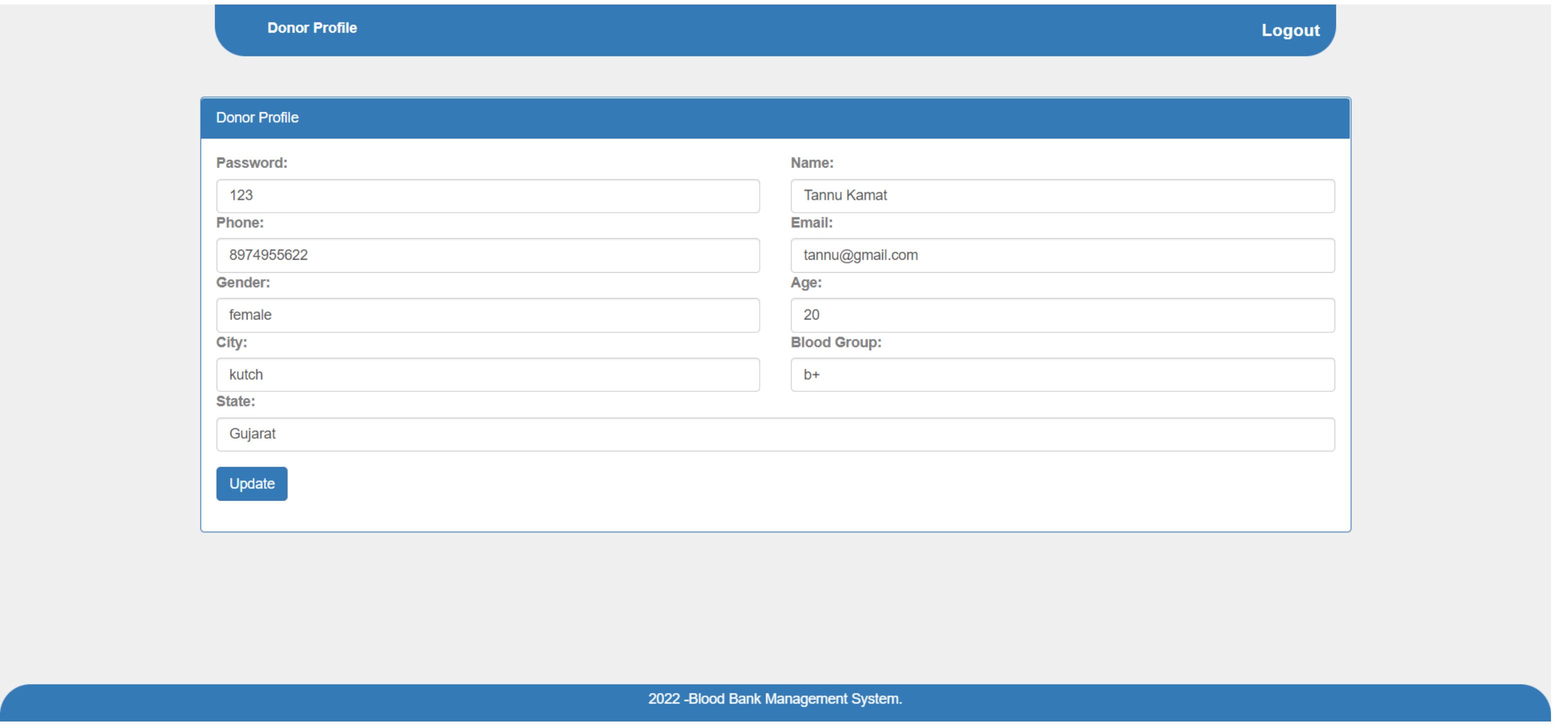


Welcome tannu@gmail.com .

Seeker Requested List

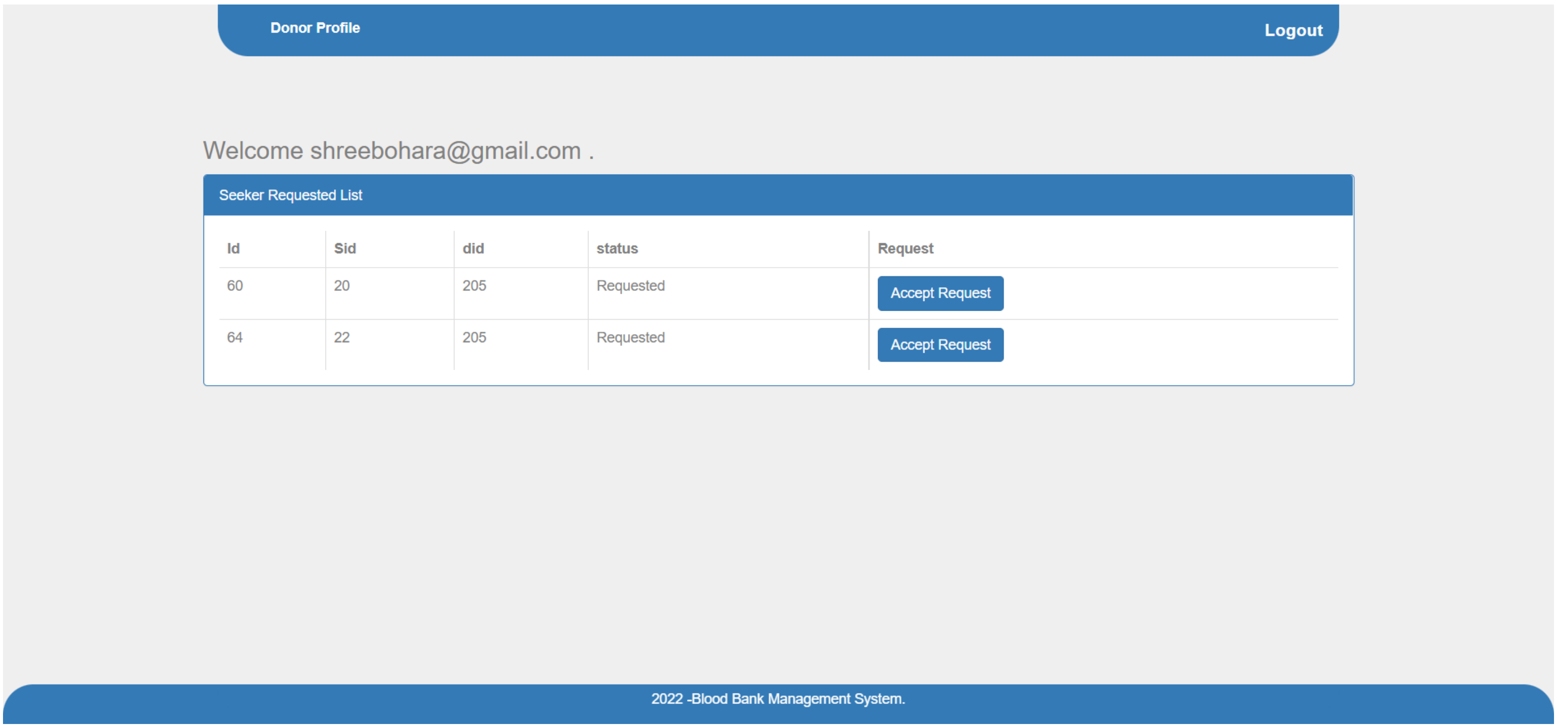
| Id | Sid | did | status | Request |
|---|---|---|---|---|

2022 -Blood Bank Management System.



Donor Profile

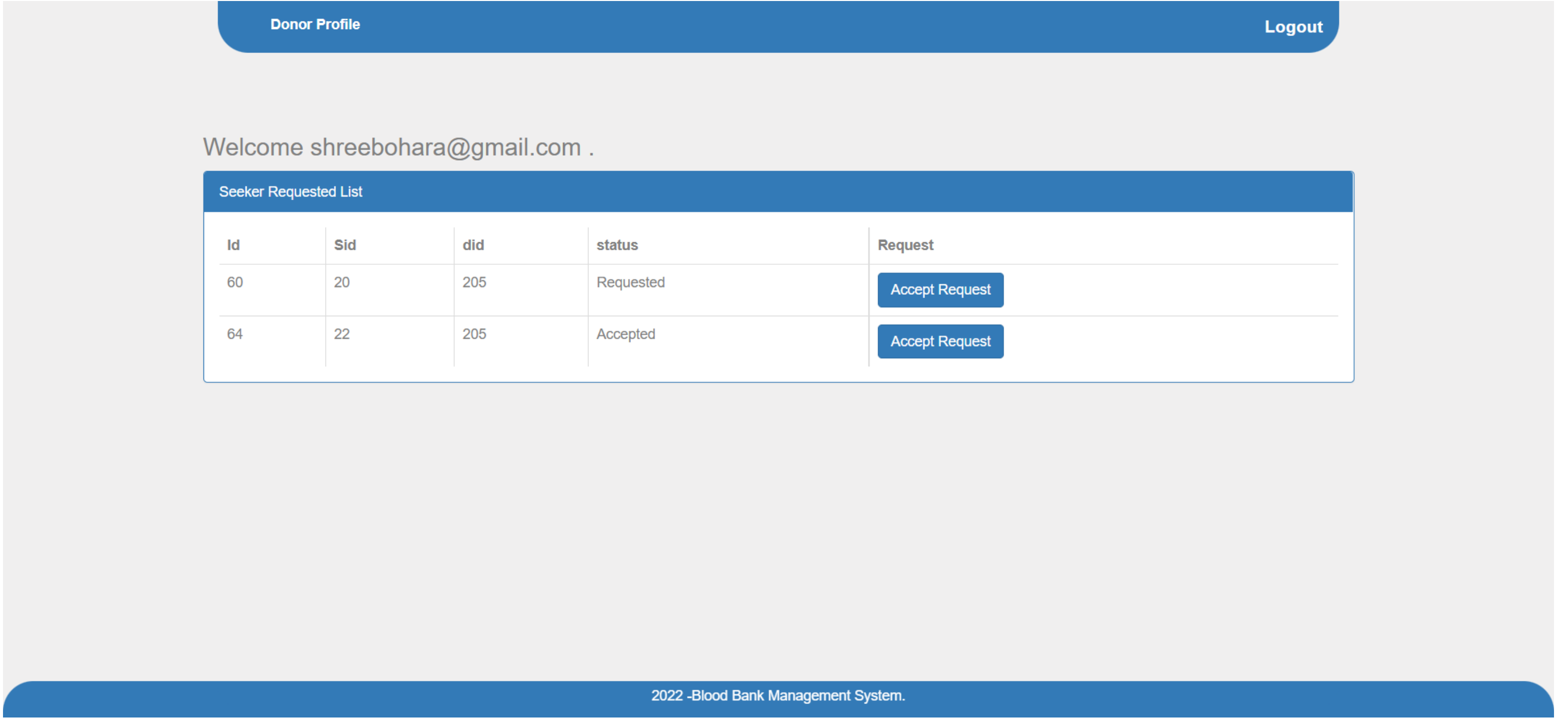
| | |
|---|---|
| Password: 123 | Name: Tannu Kamat |
| Phone: 8974955622 | Email: tannu@gmail.com |
| Gender: female | Age: 20 |
| City: kutch | Blood Group: b+ |
| State: Gujarat | |



Welcome shreebohara@gmail.com .

Seeker Requested List

| Id | Sid | did | status | Request |
|---|---|---|---|---|
| 60 | 20 | 205 | Requested | Accept Request |
| 64 | 22 | 205 | Requested | Accept Request |

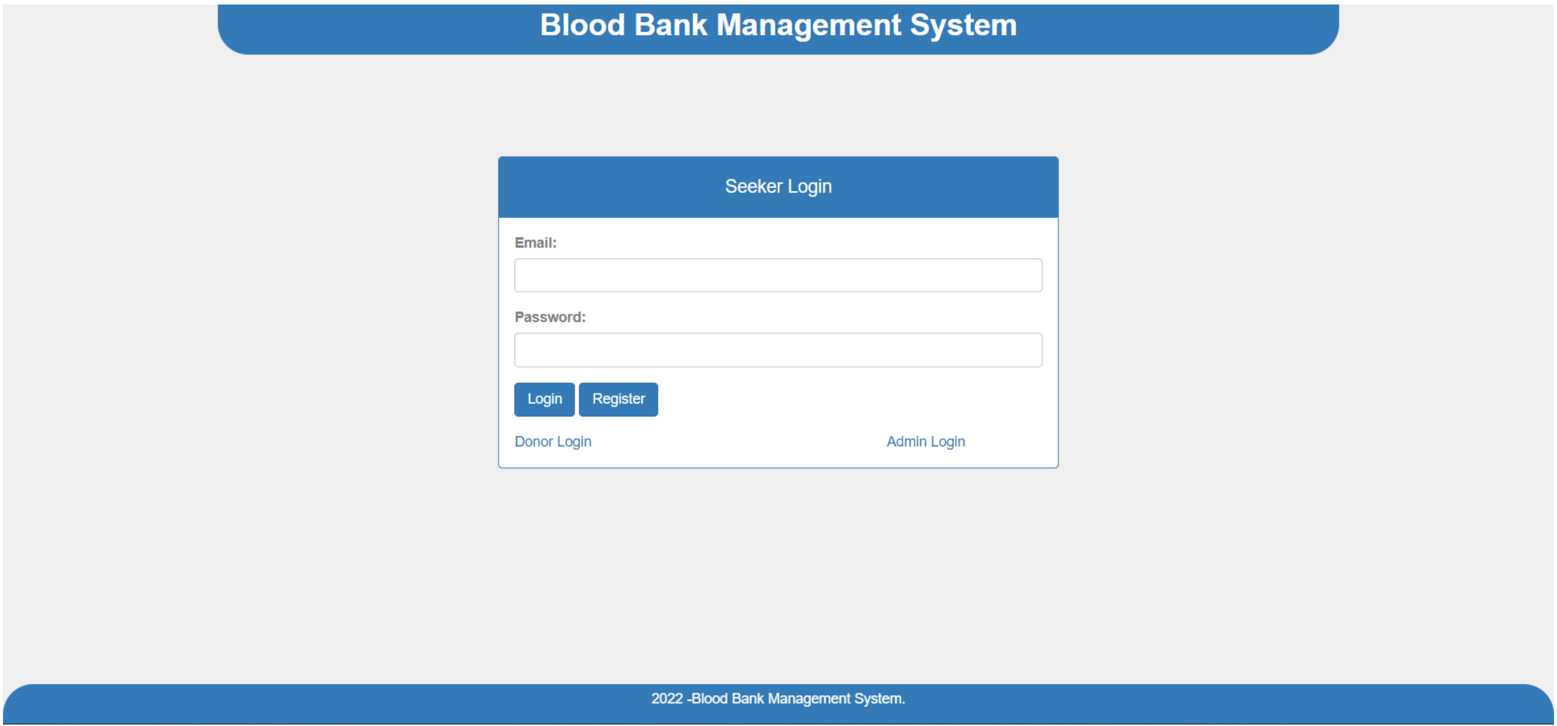


Welcome shreebohara@gmail.com .

Seeker Requested List

| Id | Sid | did | status | Request |
|---|---|---|---|---|
| 60 | 20 | 205 | Requested | Accept Request |
| 64 | 22 | 205 | Accepted | Accept Request |

# SEEKER SCREENSHOTS



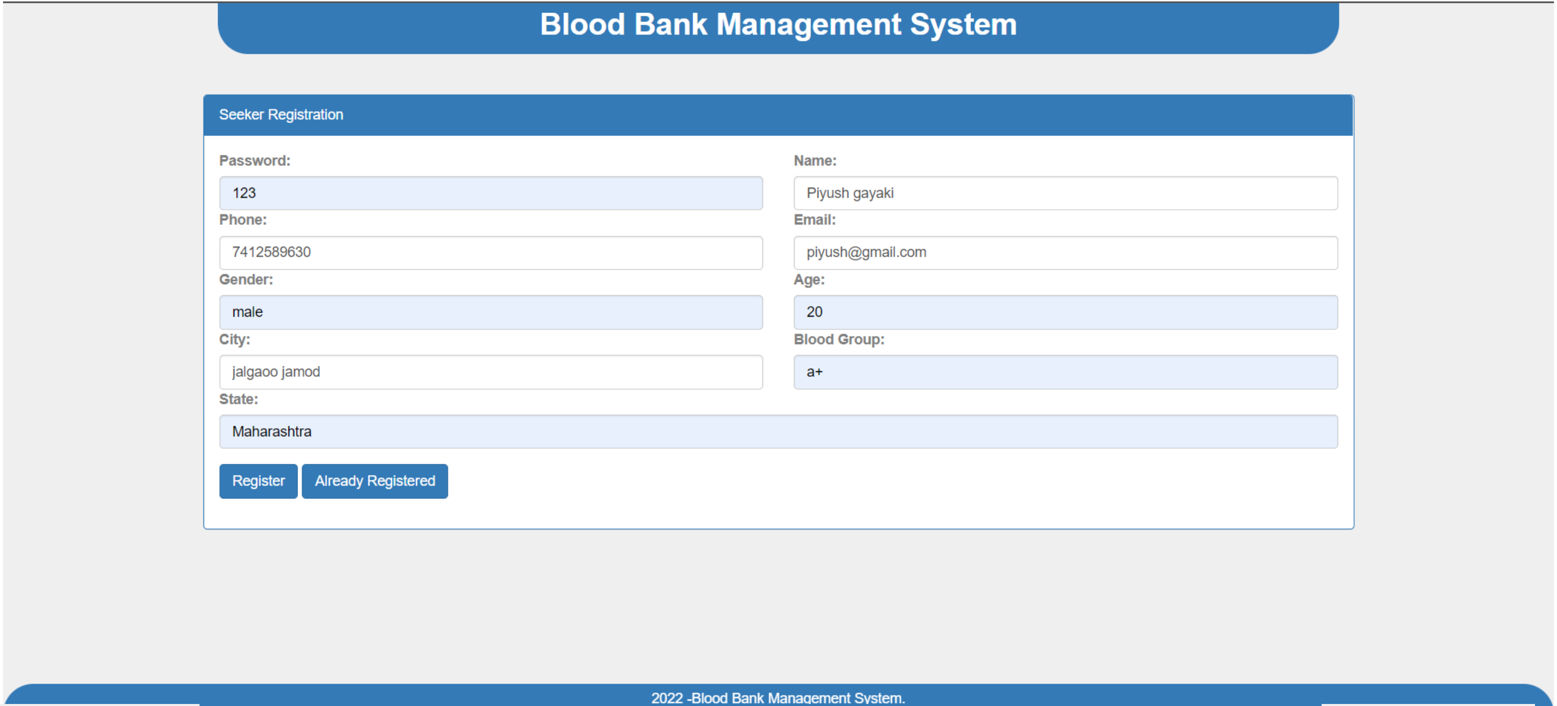
Blood Bank Management System

Seeker Login

Email:

Password:

Login
Register
Donor Login
Admin Login
2022 -Blood Bank Management System.

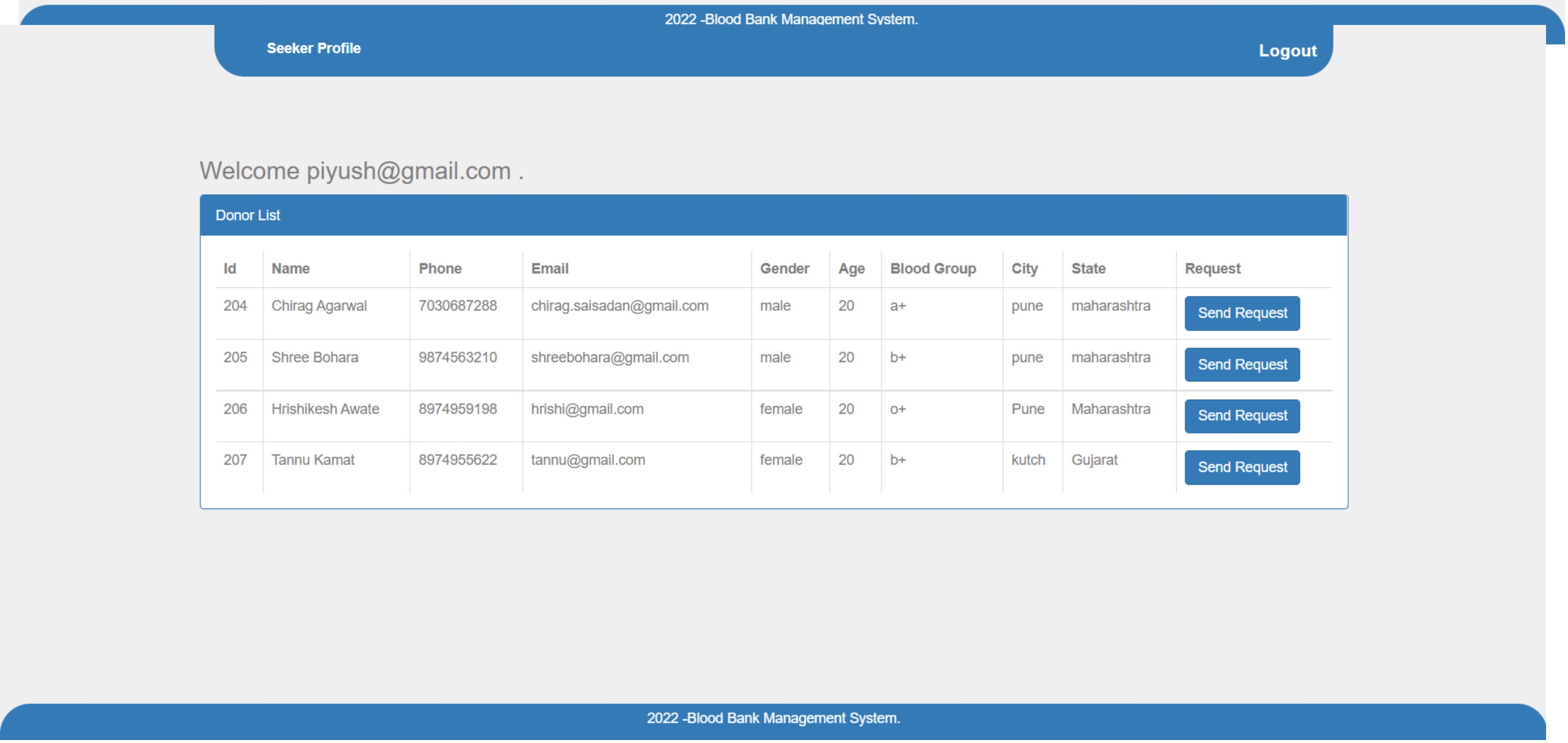


Blood Bank Management System

Seeker Registration

| | |
|---|---|
| Password: 123 | Name: Piyush gayaki |
| Phone: 7412589630 | Email: piyush@gmail.com |
| Gender: male | Age: 20 |
| City: jalgaoo jamod | Blood Group: a+ |
| State: Maharashtra | |

Register
Already Registered
2022 -Blood Bank Management System.



Welcome piyush@gmail.com .

Donor List

| Id | Name | Phone | Email | Gender | Age | Blood Group | City | State | Request |
|---|---|---|---|---|---|---|---|---|---|
| 204 | Chirag Agarwal | 7030687288 | chirag.saisadan@gmail.com | male | 20 | a+ | pune | maharashtra | Send Request |
| 205 | Shree Bohara | 9874563210 | shreebohara@gmail.com | male | 20 | b+ | pune | maharashtra | Send Request |
| 206 | Hrishikesh Awate | 8974959198 | hrishi@gmail.com | female | 20 | o+ | Pune | Maharashtra | Send Request |
| 207 | Tannu Kamat | 8974955622 | tannu@gmail.com | female | 20 | b+ | kutch | Gujarat | Send Request |

# CODE:

## Admin Servlet:-

```
package com.bloodbank.controller;


import java.io.IOException;
import java.util.List;

import javax.servlet.RequestDispatcher;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;

import com.bloodbank.model.Donor;
import com.bloodbank.model.Request;
import com.bloodbank.model.Seeker;

import dao.AdminDAOImpl;
import dao.AdminDao;
import dao.DonorDAO;
import dao.DonorDAOImpl;
import dao.RequestDAO;
import dao.RequestDAOImpl;
import dao.SeekerDAO;
import dao.SeekerDAOImpl;
```

```
@WebServlet({ "/AdminServlet",
"/AdminServlet/login","/AdminServlet/LogoutServlet","/AdminServlet/Donor","/AdminServlet/Seeker","/AdminSer
vlet/request","/AdminServlet/delete","/AdminServlet/Undo" })


public class AdminServlet extends HttpServlet {
	private static final long serialVersionUID = 1L;
	DonorDAO donorDAO = null;
	SeekerDAO seekerDAO = null;
	RequestDAO requestDAO=null;
	AdminDao adminDao=null;
    /**
     * @see HttpServlet#HttpServlet()
     */
    public AdminServlet() {
        super();

        donorDAO = new DonorDAOImpl();
        seekerDAO=new SeekerDAOImpl();
        requestDAO=new RequestDAOImpl();
        adminDao=new AdminDAOImpl();
    }


	/**
	 * @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)
	 */
	@Override
	protected void doGet(HttpServletRequest request, HttpServletResponse response) throws
ServletException, IOException {
		// TODO Auto-generated method stub
		java.lang.String action = request.getServletPath();
		  HttpSession session = request.getSession();
		switch (action) {
		case "/AdminServlet":
			Integer donor=adminDao.DonorCount();
			Integer seeker=adminDao.SeekerCount();
```

```
                            Integer rr=adminDao.ReqCount();
                            request.setAttribute("donor", donor);
                            request.setAttribute("seeker", seeker);
                            request.setAttribute("rr", rr);
                            RequestDispatcher dispatcher =
request.getRequestDispatcher("view/admin/Welcome.jsp");


                            dispatcher.forward(request, response);


                            break;


                  case "/AdminServlet/login":
                             dispatcher = request.getRequestDispatcher("../view/admin/login.jsp");
                            dispatcher.forward(request, response);
                            break;




                  case "/AdminServlet/LogoutServlet":
                             session = request.getSession();
                             session.setAttribute("drEmail",null);
                             session.setAttribute("srEmail", null);
                             session.setAttribute("adEmail", null);
                                     response.sendRedirect("http://localhost:9090/BloodBank/AdminServlet/login");
                            break;


    case "/AdminServlet/Donor":


           List<Donor> ListDonor = donorDAO.getDonor();


                            request.setAttribute("list", ListDonor);




                     dispatcher = request.getRequestDispatcher("../view/admin/Donor.jsp");
                            dispatcher.forward(request, response);
                            break;
```

```
case "/AdminServlet/Seeker":

         List<Seeker> ListSeeker = seekerDAO.getSeeker();

         request.setAttribute("list", ListSeeker);

         dispatcher = request.getRequestDispatcher("../view/admin/Seeker.jsp");

                  dispatcher.forward(request, response);

                              break;

case "/AdminServlet/request":

         List<Request> ListRequest = requestDAO.getRequest();

         System.out.println(ListRequest);

         request.setAttribute("list", ListRequest);

         dispatcher = request.getRequestDispatcher("../view/admin/Request.jsp");

                  dispatcher.forward(request, response);

                              break;


case "/AdminServlet/delete":

                              java.lang.String d_id = request.getParameter("d_id");

                  donorDAO.delete(d_id);

                              response.sendRedirect("http://localhost:9090/BloodBank/AdminServlet/Donor");

                              break;


case "/AdminServlet/Undo":

                  System.out.println("reuest1");

                              java.lang.String sid = request.getParameter("sid");

                  System.out.println("reuest1"+sid);

                  seekerDAO.delete(sid);

                  System.out.println("reuest2");

                  response.sendRedirect("http://localhost:9090/BloodBank/AdminServlet/Seeker");

                              break;


                  default:

                              break;


      }

      }
```

```
	/**
	 * @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse response)
	 */
	@Override
	protected void doPost(HttpServletRequest request, HttpServletResponse response) throws
ServletException, IOException {
		// TODO Auto-generated method stub
		 HttpSession session = request.getSession();
			java.lang.String password = request.getParameter("password");
			java.lang.String name = request.getParameter("name");
			java.lang.String phone = request.getParameter("phone");

			java.lang.String email = request.getParameter("email");
			java.lang.String gender = request.getParameter("gender");

			java.lang.String age = request.getParameter("age");



			java.lang.String blood_group = request.getParameter("blood_group");
			java.lang.String city = request.getParameter("city");
			java.lang.String state = request.getParameter("state");

			java.lang.String action = request.getServletPath();
			System.out.println(action);
			switch (action) {
				case "/AdminServlet":
					System.out.println("POST");
					break;
				case "/AdminServlet/login":

					if (email.equals("admin") && password.equals("123"))
				{
					session.setAttribute("adEmail", email);
					response.sendRedirect("../AdminServlet");
```

```
                }
                else {
                    request.setAttribute("message", "Login Failed");
                    System.out.println("Login Invalid");
                    doGet(request, response);
                }



                    break;

                case "/AdminServlet/edit":


                    break;
                case "/AdminServlet/delete":
                    break;
                default:
                    break;
            }
    }

}
```

## Donor Servlet:-

```
package com.bloodbank.controller;


import java.io.IOException;


import javax.servlet.RequestDispatcher;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;
import com.bloodbank.model.Donor;

import com.bloodbank.model.Request;

import com.bloodbank.model.Seeker;


import java.util.List;


import dao.DonorDAO;

import dao.DonorDAOImpl;

import dao.RequestDAO;

import dao.RequestDAOImpl;

import dao.SeekerDAO;

import dao.SeekerDAOImpl;


/**

 * Servlet implementation class DonorServlet

 */




@WebServlet({ "/DonorServlet", "/DonorServlet/create",

"/DonorServlet/login","/DonorServlet/edit","/DonorServlet/LogoutServlet","/DonorServlet/request" })

public class DonorServlet extends HttpServlet {

	private static final long serialVersionUID = 1L;

	DonorDAO donorDAO = null;

	SeekerDAO seekerDAO = null;

	RequestDAO requestDAO=null;

    /**

     * @see HttpServlet#HttpServlet()

     */

    public DonorServlet() {

        super();


        donorDAO = new DonorDAOImpl();

        seekerDAO=new SeekerDAOImpl();

        requestDAO=new RequestDAOImpl();
```

```
   }


    /**
     * @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)
     */
    protected void doGet(HttpServletRequest request, HttpServletResponse response) throws
ServletException, IOException {
        String action = request.getServletPath();
        switch (action) {
        case "/DonorServlet":


            List<Seeker> ListSeeker = seekerDAO.getSeeker();
              HttpSession session = request.getSession();
             Integer ddid=  (Integer) session.getAttribute("d_id");
            List<Request> ListRequest = requestDAO.getData(ddid);
            request.setAttribute("list", ListSeeker);
            request.setAttribute("list1", ListRequest);
            System.out.println(ListRequest);
            System.out.println("ListSeeker"+ListRequest);
            RequestDispatcher dispatcher =
request.getRequestDispatcher("view/donor/WelcomeDonor.jsp");
            dispatcher.forward(request, response);


            break;


        case "/DonorServlet/login":


            dispatcher = request.getRequestDispatcher("../view/donor/login.jsp");
            dispatcher.forward(request, response);


            break;


        case "/DonorServlet/create":
            dispatcher = request.getRequestDispatcher("../view/donor/register.jsp");
            dispatcher.forward(request, response);


            break;
    case "/DonorServlet/edit":
             session = request.getSession();
            String email=(String)session.getAttribute("drEmail");
            List<Donor> ListDonor = donorDAO.DonorId(email);


                    request.setAttribute("list", ListDonor);
                    System.out.println(ListDonor);
                     dispatcher = request.getRequestDispatcher("../view/donor/edit.jsp");
                    dispatcher.forward(request, response);
            break;

    case "/DonorServlet/delete":
            break;
    case "/DonorServlet/LogoutServlet":
             session = request.getSession();
             session.setAttribute("drEmail",null);
             session.setAttribute("srEmail", null);
             response.sendRedirect("http://localhost:9090/BloodBank/DonorServlet/login");
            break;


    case "/DonorServlet/request":
            System.out.println("reuest");
            Request rr=new Request();
      session = request.getSession();
            String  id= request.getParameter("id");
            System.out.println("reuest1"+id);
            requestDAO.update(id);


             response.sendRedirect("http://localhost:9090/BloodBank/DonorServlet");
            break;
```

```
        default:
            break;


    }


}
    /**
     * @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse response)
     */
    protected void doPost(HttpServletRequest request, HttpServletResponse response) throws
ServletException, IOException {
        // TODO Auto-generated method stub
        //String pt_id = request.getParameter("pt_id");
         HttpSession session = request.getSession();
        String password = request.getParameter("password");
        String name = request.getParameter("name");
        String phone = request.getParameter("phone");

        String email = request.getParameter("email");
        String gender = request.getParameter("gender");

        String age = request.getParameter("age");



        String blood_group = request.getParameter("blood_group");
        String city = request.getParameter("city");
        String state = request.getParameter("state");

        String action = request.getServletPath();
        System.out.println(action);
        switch (action) {
            case "/DonorServlet":
                System.out.println("POST");
                break;
            case "/DonorServlet/login":
```

```
    Donor drLogin = new Donor();
    System.out.println(email);
    System.out.println(password);
    drLogin.setEmail(email);
    drLogin.setPassword(password);

    String status = donorDAO.DonorLogin(drLogin);

    if (status.equals("true")) {
        session.setAttribute("drEmail", drLogin.getEmail());
        System.out.println(drLogin.getEmail());
        System.out.println("IN SERVLET SET");
        System.out.println(drLogin.getEmail());
        int d_id=donorDAO.didDonor(drLogin);
        System.out.print(d_id);
        session.setAttribute("d_id", d_id);

        response.sendRedirect("../DonorServlet");
    } else if (status.equals("false")) {
        request.setAttribute("message", "Login Failed");
        System.out.println("Login Invalid");
        doGet(request, response);
    } else {
        request.setAttribute("message", "Login Error");
        System.out.println("Error");
        doGet(request, response);
    }

    break;
case "/DonorServlet/create":

    System.out.println("POST" + action);
```

```
                    Donor dr = new Donor();


                    dr.setPassword(password);

                    dr.setName(name);

                    dr.setPhone(phone);


                    dr.setEmail(email);

                    dr.setGender(gender);

                    dr.setAge(age);

dr.setState(state);

dr.setCity(city);

                    dr.setBlood_group(blood_group);






                    if (donorDAO.save(dr)) {

                         request.setAttribute("message", "Donor Details Inserted Successfully.");

                         response.sendRedirect("../DonorServlet/login");

                    } else {

                         request.setAttribute("message", "Error somewhere !");

                         doGet(request, response);

                    }


                    break;

               case "/DonorServlet/edit":

                     session = request.getSession();

                 dr = new Donor();


                    dr.setPassword(password);

                    dr.setName(name);

                    dr.setPhone(phone);


                    dr.setEmail(email);
```

```
                                        dr.setGender(gender);

                                        dr.setAge(age);

            dr.setState(state);

            dr.setCity(city);

                                        dr.setBlood_group(blood_group);

                                        dr.setd_id((Integer) session.getAttribute("d_id"));


                                        donorDAO.update(dr);

                                        System.out.println(donorDAO.update(dr));

                                        response.sendRedirect("../DonorServlet/edit");

                                        break;

                              case "/DonorServlet/delete":

                                        break;

                              default:

                                        break;

                    }

          }



}
```

## Seeker Servlet:-

```
package com.bloodbank.controller;


import java.io.IOException;

import java.util.List;


import javax.servlet.RequestDispatcher;

import javax.servlet.ServletException;

import javax.servlet.annotation.WebServlet;

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import javax.servlet.http.HttpSession;
import com.bloodbank.model.Donor;
import com.bloodbank.model.Request;
import com.bloodbank.model.Seeker;

import dao.DonorDAOImpl;
import dao.RequestDAO;
import dao.RequestDAOImpl;
import dao.DonorDAO;
import dao.SeekerDAO;
import dao.SeekerDAOImpl;


@WebServlet({ "/SeekerServlet", "/SeekerServlet/create",
"/SeekerServlet/login","/SeekerServlet/edit","/SeekerServlet/LogoutServlet","/SeekerServlet/request" })

public class SeekerServlet extends HttpServlet {
            private static final long serialVersionUID = 1L;
            SeekerDAO seekerDAO = null;
            DonorDAO donorDAO = null;
            RequestDAO requestDAO=null;
   public SeekerServlet() {
      super();
      System.out.println("listseeker");
      seekerDAO=new SeekerDAOImpl();
      donorDAO = new DonorDAOImpl();
      requestDAO=new RequestDAOImpl();
      List<Seeker> ListSeeker = seekerDAO.getSeeker();


                        System.out.println(ListSeeker);
   }
```

```
        protected void doGet(HttpServletRequest request, HttpServletResponse response) throws
ServletException, IOException {

                String action = request.getServletPath();
                switch (action) {
                case "/SeekerServlet":



        List<Donor> ListDonor = donorDAO.getDonor();


                        request.setAttribute("list", ListDonor);
                        System.out.println(ListDonor);

                        RequestDispatcher dispatcher =
request.getRequestDispatcher("view/seeker/WelcomeSeeker.jsp");
                        dispatcher.forward(request, response);

                        break;

                case "/SeekerServlet/login":

                        dispatcher = request.getRequestDispatcher("../view/seeker/login.jsp");
                        dispatcher.forward(request, response);

                        break;

                case "/SeekerServlet/create":

                        dispatcher = request.getRequestDispatcher("../view/seeker/register.jsp");
                        dispatcher.forward(request, response);

                        break;
                case "/SeekerServlet/edit":
```

```
    System.out.println("inside");
     HttpSession session = request.getSession();
        String email=(String)session.getAttribute("srEmail");
        System.out.println("inside"+email);
        List<Seeker> ListSeeker = seekerDAO.SeekerId(email);
        System.out.println("inside1");
            request.setAttribute("list", ListSeeker);
            System.out.println(ListSeeker);
            System.out.println("inside1"+ListSeeker);
             dispatcher = request.getRequestDispatcher("../view/seeker/edit.jsp");
            dispatcher.forward(request, response);
    break;

case "/SeekerServlet/delete":
    break;

case "/SeekerServlet/LogoutServlet":
    session = request.getSession();
     session.setAttribute("drEmail",null);
     response.sendRedirect("http://localhost:9090/BloodBank/SeekerServlet/login");
    break;

case "/SeekerServlet/request":
    System.out.println("reuest");
    Request rr=new Request();
 session = request.getSession();
    String  did= request.getParameter("d_id");
    System.out.println("reuest1");
    rr.setDid(did);
    System.out.println("reuest2");
    rr.setStatus("Requested");
    rr.setSid((Integer) session.getAttribute("sid"));
    System.out.println("reuest3");
    requestDAO.save(rr);
```

```
                        response.sendRedirect("http://localhost:9090/BloodBank/SeekerServlet");



                        break;
                default:
                        break;


        }
        }


        /**
         * @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse response)
         */
        protected void doPost(HttpServletRequest request, HttpServletResponse response) throws
ServletException, IOException {

                String password = request.getParameter("password");
                String name = request.getParameter("name");
                String phone = request.getParameter("phone");

                String email = request.getParameter("email");
                String gender = request.getParameter("gender");

                String age = request.getParameter("age");

                String blood_group = request.getParameter("blood_group");
                String city = request.getParameter("city");
                String state = request.getParameter("state");

                String action = request.getServletPath();
                System.out.println(action);
                switch (action) {
                        case "/SeekerServlet":
                                System.out.println("POST");
                                break;
```

```
case "/SeekerServlet/login":

    HttpSession session = request.getSession();

    Seeker srLogin = new Seeker();
    System.out.println(email);
    System.out.println(password);
    srLogin.setEmail(email);
    srLogin.setPassword(password);

    String status = seekerDAO.SeekerLogin(srLogin);

    if (status.equals("true")) {
        session.setAttribute("srEmail", srLogin.getEmail());
        System.out.println("IN SERVLET SET");
        System.out.println(srLogin.getEmail());
        int sid=seekerDAO.sidSeeker(srLogin);
        System.out.print(sid);
        session.setAttribute("sid", sid);
        response.sendRedirect("../SeekerServlet");
    } else if (status.equals("false")) {
        request.setAttribute("message", "Login Failed");
        System.out.println("Login Invalid");
        doGet(request, response);
    } else {
        request.setAttribute("message", "Login Error");
        System.out.println("Error");
        doGet(request, response);
    }

    break;
case "/SeekerServlet/create":

    System.out.println("POST" + action);
```

```
                Seeker sr = new Seeker();


                sr.setPassword(password);

                sr.setName(name);

                sr.setPhone(phone);


                sr.setEmail(email);

                sr.setGender(gender);

                sr.setAge(age);

                sr.setState(state);

                sr.setCity(city);

                sr.setBlood_group(blood_group);



                if (seekerDAO.save(sr)) {

                        request.setAttribute("message", "Seeker Details Inserted Successfully.");

                        response.sendRedirect("../SeekerServlet/login");

                } else {

                        request.setAttribute("message", "Error somewhere !");

                        doGet(request, response);

                }


                break;

        case "/SeekerServlet/edit":


                 session = request.getSession();

                   sr = new Seeker();


                   sr.setPassword(password);

                   sr.setName(name);

                   sr.setPhone(phone);


                   sr.setEmail(email);

                   sr.setGender(gender);

                   sr.setAge(age);
```

```
                  sr.setState(state);

                  sr.setCity(city);

                  sr.setBlood_group(blood_group);

                  sr.setSid((Integer) session.getAttribute("sid"));


                  seekerDAO.update(sr);

                        System.out.println(seekerDAO.update(sr));

                        response.sendRedirect("../SeekerServlet/edit");

                break;

            case "/SeekerServlet/delete":

                break;

            default:

                break;

        }

    }


}
```

## logout Servlet:-

```
package com.bloodbank.controller;

import java.io.IOException;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import javax.servlet.http.HttpSession;

/**
 * Servlet implementation class LogoutServlet
 */
@WebServlet("/LogoutServlet")
public class LogoutServlet extends HttpServlet {
    private static final long serialVersionUID = 1L;

   /**
    * @see HttpServlet#HttpServlet()
    */
   public LogoutServlet() {
      super();


   }

    /**
```

```
	 * @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)
	 */
	protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException
{
		// TODO Auto-generated method stub
		 HttpSession session = request.getSession();
		 session.setAttribute("drEmail",null);
		 session.setAttribute("srEmail", null);
		 session.setAttribute("adEmail", null);
		 response.sendRedirect("DonorServlet/login");
	}

	/**
	 * @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse response)
	 */
	protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException
{
		// TODO Auto-generated method stub
		doGet(request, response);
	}

}
```

## } Donor:-

```
package com.bloodbank.model;



public class Donor {

	private Integer d_id;
	private String password;
	private String name;
	private String phone;
	private String email;
	private String gender;
	private String age;
	private String state;
	private String blood_group;
	private String dob;
	private String city;



	@Override
	public String toString() {
		// TODO Auto-generated method stub
		return "Donor [d_id=" + d_id + ", password=" + password + ", name=" + name + ", phone=" + phone + ", email="
		+ email + ", gender=" + gender + ", age=" + age + ", city=" + city + ", blood_group="
		+ blood_group + ", state=" + state + ",city=" + city + "]";
}

	public Integer getd_id() {
		return d_id;
	}
	public void setd_id(Integer d_id) {
		this.d_id = d_id;
	}
	public String getPassword() {
		return password;
	}
	public void setPassword(String password) {
		this.password = password;
```

```
    }
    public String getName() {
        return name;
    }
    public void setName(String name) {
        this.name = name;
    }
    public String getPhone() {
        return phone;
    }
    public void setPhone(String phone) {
        this.phone = phone;
    }
    public String getEmail() {
        return email;
    }
    public void setEmail(String email) {
        this.email = email;
    }
    public String getGender() {
        return gender;
    }
    public void setGender(String gender) {
        this.gender = gender;
    }
    public String getAge() {
        return age;
    }
    public void setAge(String age) {
        this.age = age;
    }
    public String getState() {
        return state;
    }
    public void setState(String state) {
        this.state = state;
    }
    public String getBlood_group() {
        return blood_group;
    }
    public void setBlood_group(String blood_group) {
        this.blood_group = blood_group;
    }

    public String getCity() {
        return city;
    }
    public void setCity(String city) {
        this.city = city;
    }




}
```

## Request:-

```
package com.bloodbank.model;



public class Request {

    private Integer id;
    private String rdid;
    private Integer sid;
    private String status;


    @Override
    public String toString() {
        // TODO Auto-generated method stub
        return "Request [id=" + id + ", sid=" + sid + ", rdid=" + rdid + ", status=" + status + "]";
}



    public Integer getId() {
        return id;
    }
    public void setId(Integer id) {
        this.id = id;
    }
    public String getDid() {
        return rdid;
    }
    public void setDid(String rdid) {
        this.rdid = rdid;
    }
    public Integer getSid() {
        return sid;
    }
    public void setSid(Integer sid) {
        this.sid = sid;
    }
    public String getStatus() {
        return status;
    }
    public void setStatus(String status) {
        this.status = status;
    }



}
```

## Seeker:-

```
package com.bloodbank.model;



public class Seeker {

    private Integer sid;
    private String password;
    private String name;
    private String phone;
    private String email;
    private String gender;
    private String age;
    private String state;
    private String blood_group;
    private String city;



    @Override
    public String toString() {
        // TODO Auto-generated method stub
        return "Seeker [sid=" + sid + ", password=" + password + ", name=" + name + ", phone=" + phone + ", email="
        + email + ", gender=" + gender + ", age=" + age + ", city=" + city + ", blood_group="
        + blood_group + ", state=" + state + ",city=" + city + "]";
}
```

```
	public Integer getSid() {
		return sid;
	}
	public void setSid(Integer sid) {
		this.sid = sid;
	}
	public String getPassword() {
		return password;
	}
	public void setPassword(String password) {
		this.password = password;
	}
	public String getName() {
		return name;
	}
	public void setName(String name) {
		this.name = name;
	}
	public String getPhone() {
		return phone;
	}
	public void setPhone(String phone) {
		this.phone = phone;
	}
	public String getEmail() {
		return email;
	}
	public void setEmail(String email) {
		this.email = email;
	}
	public String getGender() {
		return gender;
	}
	public void setGender(String gender) {
		this.gender = gender;
	}
	public String getAge() {
		return age;
	}
	public void setAge(String age) {
		this.age = age;
	}
	public String getState() {
		return state;
	}
	public void setState(String state) {
		this.state = state;
	}
	public String getBlood_group() {
		return blood_group;
	}
	public void setBlood_group(String blood_group) {
		this.blood_group = blood_group;
	}

	public String getCity() {
		return city;
	}
	public void setCity(String city) {
		this.city = city;
	}


}
```

# FUTURE SCOPE

As there was a little number of contact person's information given, some people may face difficulty in getting blood fast. So i like to gather more information regarding thecontact persons in other cities as well as villages and will provide much more servicesfor the people and help everyone with humanity.

# CONCLUSION

The growing quality demand in the hospital sector makes it necessary to exploit the whole potential of stored data efficiently, not only the clinical data in order toimprove diagnoses and treatments but also on management in order to minimizecosts and improve the care given to the patients.

This system will be beneficial for the Admin, the Doctors and the Patients.

# REFERENCES

BIBLIOGRAPHY

- Learning PHP,MySQL, JavaScript, CSS and HTML 5, Robin Nixon,O'Reilly publication.
- Learning Java, 4th Edition by Patrick Niemeyer, Daniel Leuck.
- High Performance MYSQL by Baron Schwartz.