

A PROJECT REPORT ON

***Blood Bank Management System***

SUBMITTED BY:

# Miss.Rasika Santosh chavan

(2124UCEF1001)

SUBJECT: **C++** PROGRAMMING

Under the guidance of

**Prof. Ishwari Tirse**

Department of **Computer Science and Engineering**

Sanjivani Rural Education Society’s SANJIVANI UNIVERSITY

KOPARGAON – 423603 (2024-2025)

# INDEX

|  |  |  |
| --- | --- | --- |
| Sr No. | Content | Page No. |
| 1 | INTRODUCTION | 3 |
| 2 | CODE | 4 |
| 3 | OUTPUT | 9 |
| 4 | CONCLUSION | 12 |

# INTRODUCTION

*Blood Bank Management System*

Blood Bank Management System is a simple C++ application for the management and organization of blood donors and requests for blood. It is an application meant to register donors, view all the names of the registered donors, search for a particular donor according to the blood group, and request for blood. All information regarding the donors will be saved in a file so that it will not get lost when multiple sessions have been done.

Key features:

1. Registration of new blood donors with his / her name, blood group, contact details, and age.

2. View the donor list view showing registered donors and full details regarding the donor.

3. Search for donors based on a desired specific blood group in order to provide blood.

This system will help a hospital or blood bank manage the information regarding donors efficiently, and in case of an emergency, it will find donors with ease.

# CODE

# #include <iostream>

# #include <fstream>

# #include <string>

# using namespace std;

# class Donor {

# string name;

# string bloodGroup;

# string contact;

# int age;

# public:

# // Register Donor Method

# void registerDonor() {

# cout << "Enter Donor Name: ";

# cin.ignore();

# getline(cin, name);

# cout << "Enter Blood Group (A+/A-/B+/B-/O+/O-/AB+/AB-): ";

# cin >> bloodGroup;

# cout << "Enter Contact Number: ";

# cin >> contact;

# cout << "Enter Age: ";

# cin >> age;

# }

# // Display Donor Method

# void displayDonor() const {

# cout << "Name: " << name << "\nBlood Group: " << bloodGroup << "\nContact: " << contact << "\nAge: " << age << endl;

# }

# string getBloodGroup() const {

# return bloodGroup;

# }

# string getName() const {

# return name;

# }

# // Method to save donor data to file

# void saveToFile(ofstream &outFile) const {

# outFile << name << "\n" << bloodGroup << "\n" << contact << "\n" << age << "\n";

# }

# // Method to load donor data from file

# void loadFromFile(ifstream &inFile) {

# getline(inFile, name);

# inFile >> bloodGroup;

# inFile >> contact;

# inFile >> age;

# inFile.ignore(); // Ignore newline character after age

# }

# };

# // Save donor data to file

# void saveDonor(const Donor& donor) {

# ofstream outFile("donors.dat", ios::app); // Open file in append mode

# if (outFile.is\_open()) {

# donor.saveToFile(outFile);

# outFile.close();

# cout << "Donor registered successfully!" << endl;

# } else {

# cout << "Error saving donor information!" << endl;

# }

# }

# // Display all donors

# void displayAllDonors() {

# Donor donor;

# ifstream inFile("donors.dat");

# if (inFile.is\_open()) {

# while (!inFile.eof()) {

# donor.loadFromFile(inFile);

# if (inFile.eof()) break; // Avoids printing last empty record

# donor.displayDonor();

# cout << "---------------------------" << endl;

# }

# inFile.close();

# } else {

# cout << "Error opening file!" << endl;

# }

# }

# // Search donor by blood group

# void searchDonorByBloodGroup(const string& bloodGroup) {

# Donor donor;

# bool found = false;

# ifstream inFile("donors.dat");

# if (inFile.is\_open()) {

# while (!inFile.eof()) {

# donor.loadFromFile(inFile);

# if (inFile.eof()) break; // Avoids printing last empty record

# if (donor.getBloodGroup() == bloodGroup) {

# donor.displayDonor();

# found = true;

# cout << "---------------------------" << endl;

# }

# }

# inFile.close();

# if (!found) {

# cout << "No donor found with Blood Group: " << bloodGroup << endl;

# }

# } else {

# cout << "Error opening file!" << endl;

# }

# }

# // Request blood (search for available donors)

# void requestBlood() {

# string bloodGroup;

# cout << "Enter Blood Group needed: ";

# cin >> bloodGroup;

# searchDonorByBloodGroup(bloodGroup);

# }

# int main() {

# int choice;

# Donor donor;

# while (true) {

# cout << "\n\*\*\* Blood Bank Management System \*\*\*\n";

# cout << "1. Register Donor\n2. Display All Donors\n3. Search Donor by Blood Group\n4. Request Blood\n5. Exit\n";

# cout << "Enter your choice: ";

# cin >> choice;

# switch (choice) {

# case 1:

# donor.registerDonor();

# saveDonor(donor);

# break;

# case 2:

# displayAllDonors();

# break;

# case 3: {

# string bloodGroup;

# cout << "Enter Blood Group to search: ";

# cin >> bloodGroup;

# searchDonorByBloodGroup(bloodGroup);

# break;

# }

# case 4:

# requestBlood();

# break;

# case 5:

# cout << "Exiting the system..." << endl;

# return 0;

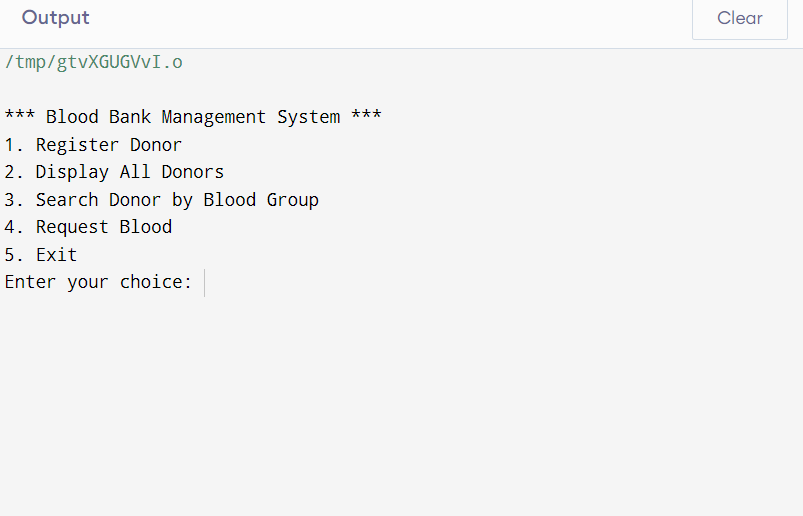
# default:

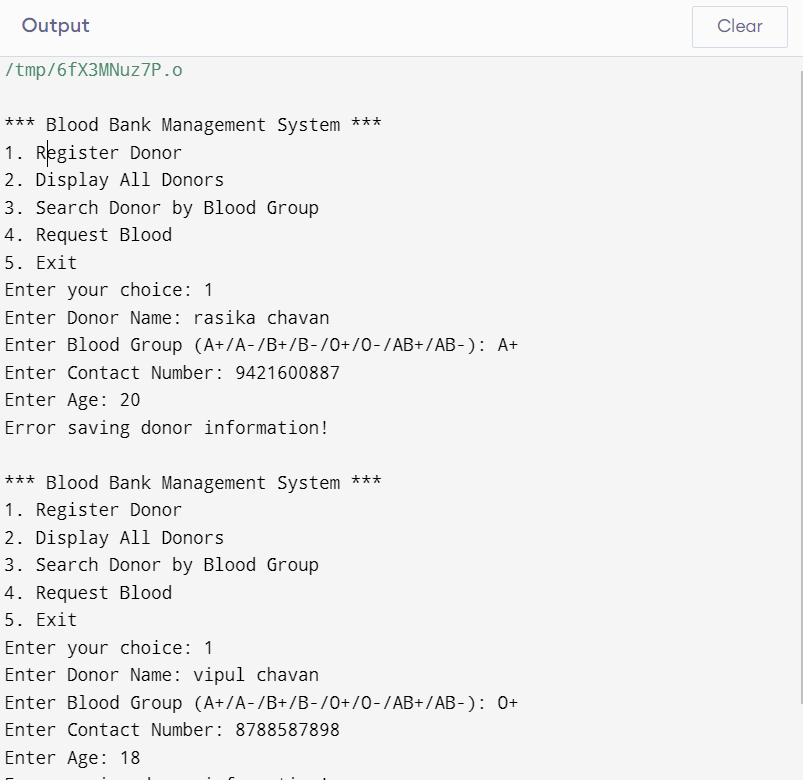
# cout << "Invalid choice! Please try again." << endl;

# }

# }

# }

**OUTPUT**



**CONCLUSION**

**Conclusion:**

The given C++ program serves as an efficient means of managing a basic Blood Bank Management System by including the following essential features: donor registration, display of data of all donors, search for a donor by blood group, and requisition of blood. It uses file handling to ensure storage and retrieval of data related to donors so that the data remains persistent over several runs of the program.

**Strength:**

The program is simple and hence easy to understand and does not pose much of a problem to be used by a basic user for blood donor management.

**Last Words:**

Overall, this program is a good starting point for a basic blood bank system but will definitely require improving validation and error handling with an upgrade to more advanced database management for better scalability.