

**North South University**

Department of Electrical and Computer Engineering

**LAB REPORT**

**Topic:** Implementation of a Blood Bank Management System

**Course:** CSE115L

**Section:** 1

**Submitted By:**

**Name:** Fazly Fardin Chowdhury

**ID:** 2322480642

**Abstract:**

This lab report documents the implementation of a Blood Bank Management System using the C programming language. The system allows users to purchase donor, view available donors, remove donor records, search for specific blood group, and edit existing donor records. The implementation utilizes file handling for data storage and retrieval.

**Introduction:**

The Blood Bank Management System is designed to provide a user-friendly interface for purchasing and editing blood donors. The system stores donor information, such as donor name, blood group, donor age, donor hemoglobin level, contact number, and location in a file. Users can perform various operations on the donor records through a main menu.

**Methods:**

The implementation of the Blood Bank Management System involves several key functions. The main components are as follows:

**Structures:** The program defines a structure named "Donor" to store information about each blood donor. The structure contains fields such as name, group, location, contact, age, and hemoglobin.

**Main Menu Function:** The main\_menu function displays the main menu options and handles user input. It presents a menu of actions, such as purchasing donor, adding a donor record, viewing donor records, deleting donor records, editing records or exiting the program. The user's choice is determined using a switch statement.

**Add Function:** The add function allows users to add new donor records. It prompts the user to enter details such as donor name, donor group, donor location, donor contact ,donor age, donor hemoglobin level. It saves the record to a file named "donor.dat" for future reference.

**List Function:** The list function retrieves all rental records from the "donor.dat" file and displays them in a formatted manner. It reads each record line by line, extracts the data, and prints it on the screen in a tabular format.

**Delete Function:** The delete function enables users to remove a specific donor record. The function deletes a donor from the structure donors array and saves it in "donor.dat" file. It is used after purchasing donor and in the delete donor window.

**Search Function:** The search function allows users to search for a specific donor record by blood group. It prompts the user to enter the blood group and performs a search in the structure of donors. If a match is found, it displays the record on the console; otherwise, it informs the user that the record was not found.

**Edit Function:** The edit function provides the functionality to modify existing donor records. The function would prompt the user to enter the donor name of the record to be edited, search for a match, and allow the user to update the record with new information.

**Results and Discussion:**

The implementation of the Blood Bank Management System provides users with a convenient way to purchase donor. The system allows for adding new donor records, viewing existing records, deleting records, searching for specific records, and, with the mentioned edit function, editing records. By utilizing file handling, the system ensures persistent data storage and retrieval.

During testing, the system exhibited the expected behavior. Users were able to perform various operations without encountering any critical issues. The input validation checks helped ensure that only valid data was accepted, enhancing the reliability and integrity of the system.

**Conclusion:**

The implementation of the Blood Bank Management System demonstrates the effective use of C programming concepts and file handling techniques. The system provides essential functionalities for managing donor records, including adding, viewing, deleting, searching, and, with the mentioned edit function, editing donor records. It offers a user-friendly interface and incorporates input validation to ensure data integrity. Further improvements and enhancements can be made based on specific requirements or future development plans.