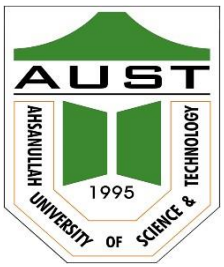

Ahsanullah University of Science and Technology (AUST)
Department of Electrical and Electronic Engineering

Course No.: EEE 1110

Course Title: Programming Language Laboratory

AUST Blood Bank Management System

ID: **20230205166**

Name: **Rajit Roy**

Section: **C2**

For the students of the

Department of Electrical and Electronic Engineering

1st Year, 1st Semester

Overview

This report design and implementation of the **AUST Blood Bank Management System**. The system is intended to manage blood donors' information efficiently. It allows adding, editing, searching, and deleting donor records and facilitates maintaining a structured database of blood donors. Making it a crucial tool for blood banks and health organizations. The system's implementation is written in C++ and involves reading and writing data from/to text files for persistent storage.

Objective

The primary objective of the system is managing blood donors and their associated data, such as their blood group, contact information, and last donation date. Key functionalities include:

- Adding new donors.
- Showing all donor information.
- Searching donors by blood group.
- Editing donor details by their unique ID.
- Deleting donor records by their ID.

Design and Implementation

Here is a breakdown of the system's components:

Structure

- The primary structure that holds the details of each donor:
- name: Stores the donor’s name.
- gender: Stores the donor’s gender (Male/Female).
- age: Stores the donor’s age (16–50).
- blood: Stores the donor’s blood group (e.g., A+,A-, B+,B-,AB+,AB-,O+,O-)

- address: Stores the donor's address.
- date: Stores the last donation date in the format YYYY-MM-DD.
- phone: Stores the donor's phone number.

Functions

- **displayMenu ()**: Displays the main menu with options to add, show, search, edit, or delete donors.
- **isValidGender ()**: Ensures that only valid genders (Male, Female) are entered.
- **isValidAge ()**: Validates the age (between 16 and 50).
- **isValidBloodGroup ()**: Checks if the blood group entered matches the acceptable values (A+, O-, etc.).
- **isValidDate ()**: Validates if the date is in the correct format (YYYY-MM-DD).
- **isValidPhoneNumber ()**: Ensures that the phone number is exactly 11 digits long and contains only digits.
- **AddDonor ()**: Adds a new donor to the file. It checks all validations before saving the record.
- **ShowDonor ()**: Reads and displays all donor records from the file in a neatly formatted table.
- **searchDonor ()**: Searches for a donor based on blood group and displays their information.
- **EditDonor ()**: Edits the last donation date of a donor by their unique ID and updates the file with the new information.
- **DeleteDonor ()**: Removes a donor from the file based on their unique ID and rewrites the remaining records into the file.

Problems During Implementation

Data Validation: Ensuring that user input, such as gender, age, blood group, and dates, is valid before storing it. This required writing validation functions for each type of input.

File Management: Managing donor records in a persistent file required careful consideration, especially when editing or deleting records.

Input Format Consistency: The format of dates and phone numbers had to be standardized to avoid incorrect data storage and ensure ease of future searches.

Future Improvements

The current system is quite basic and could be enhanced in the following ways:

- **Database Integration**: Instead of using text files, a relational database (e.g., MySQL) could be used to store and retrieve donor data more efficiently.
- **Search Improvements**: Add more advanced search functionalities, such as searching by name, age, or multiple blood groups.
- **Data Encryption**: Encrypt sensitive information like phone numbers before storing it in files.
- **GUI Interface**: The system could be enhanced with a graphical user interface (GUI) to make it more user-friendly.

Conclusion

The AUST Blood Bank Management System offers a practical solution for managing blood donor data, focusing on input validation, efficient file handling, and formatted output. While the system meets its basic objectives, there is potential for future expansion and enhancements to make it even more robust. There are opportunities to improve validation, security, and user experience, making the system more robust and user-friendly.