PROJECT DOCUMENT: BLOOD MANAGEMENT SYSTEM

TABLE OF CONTENTS

INTRODUCTION

Purpose of the Document

The purpose of the document is to provide an overview of the Blood Management System project developed using Core Java. It outlines the system's requirements, architecture, features, user interface and testing procedures.

Project Overview

The Blood Management System is a console-based application that allows Administrators to manage various records of the blood donated by different Donors, and Receivers who receive the donated blood.

Scope of the Project

The scope of the Blood Management System project include the following functionalities :

~ Register/Delete Admin
~ Admin Login/Logout
~ Change Admin Password

~ Donor Portal
~ Receiver Portal
~ Generate Report

SYSTEM REQUIREMENTS

Functional Requirements

~ Register/Delete Admin: Admin Credentials and Database Login (Management) Credentials are required.

~ Admin Login/Logout: Admin Credentials are required to login to the system.

~ Change Admin Password: Current and New Admin Passwords are required.

~ Donor Portal: Add, Update or Delete Donors.

~ Receiver Portal: Add, Update or Delete Receivers.

~ Generate Report: Display details of all Donors and Receivers, along with total blood remaining for each blood type.

ARCHITECTURE

High-Level Architecture

~ Separate classes for Admins, Donors and Receivers inside Entity package.

~ Each class has its own Add, Delete and Update methods.

~ Respective classes connect to Admin, Donor and Receiver tables inside Entity database.

~ Driver code and database entry format checks written inside main().

Class Diagram

```
                         Entity (Package)
                                |
                                |
                                |
  Admin -- -- -- -- -- -- -- -- -- -- Donor -- -- -- -- -- -- -- -- -- Receiver
    |                           |                            |
    |                           |                            |
    |                           |                            |

  Admin()                    Donor()                     Receiver()

 registerAdmin()            addDonor()                  addReceiver()
 deleteAdmin()             deleteDonor()              deleteReceiver()
 updatePassword()          updateDonor()              updateReceiver()
 adminLogin()             generateReport()            generateReport()
                          updateDonation()
                       generateReportByType()
```

Sequence Diagram

```
                              main()
                                |
                                |
                                |
Register Admin -- -- -- -- -- -- -- Admin Login -- -- -- -- -- -- -- Delete Admin
                                |
                                |
                                |
Generate Report -- -- -- Donor Portal -- -- -- Receiver Portal -- -- Change Password
                           |                   |
                           |                   |
                           |                   |
                       Add Donor          Add Receiver
                      Delete Donor       Delete Receiver
                      Update Donor       Update Receiver
                                |
                                |
                                |
                [repeat: available from each level]
               [exit/back: available from each level]
```

USER INTERFACE

This project follows a Console-Based, Event-Driven approach. Hence, the UI is the System CLI.

TECHNOLOGIES USED

~ Core Java
~ MySQL
~ Visual Studio Code
~ Git

TESTING

Test Cases (Admin)

1. username: ronaldo1 password: halamadrid1
2. username: ronaldo2 password: halamadrid2
3. username: ronaldo1 password: halamadrid3
4. username: ronaldo3 password: halamadrid1

Test Cases (Donor)

1. donor_id: 1234567890 donor_name: Ronaldo1 donor_type: O- amount_donated: 20
2. donor_id: 1234567891 donor_name: Ronaldo2 donor_type: A- amount_donated: 19
3. donor_id: 1234567891 donor_name: Ronaldo3 donor_type: B- amount_donated: 47
4. donor_id: 1234567892 donor_name: Ronaldo2 donor_type: AB+ amount_donated: 9
5. donor_id: 12345678923 donor_name: Ronaldo2 donor_type: A+ amount_donated: 7

Test Cases (Receiver)

1. rceivr_id: 9876543214 rceivr_name: Ronaldo5 rceivr_type: O+ amount_required: 23
2. rceivr_id: 9876543215 rceivr_name: Ronaldo7 rceivr_type: B+ amount_required: 17
3. rceivr_id: 9876543215 rceivr_name: Ronaldo6 rceivr_type: A+ amount_required: 59
4. rceivr_id: 9876543216 rceivr_name: Ronaldo7 rceivr_type: AB- amount_required: 47
5. rceivr_id: 98765432167 rceivr_name: Ronaldo7 rceivr_type: B- amount_required: 91

Unit Testing

The above test cases are sufficient for Unit Testing, and are used for the same.

CONCLUSION

Programming tools learned during the development of the Blood Management System project are as follows :

~ Core Java (Packages, Classes, Static Variables and Methods, Database Connectivity with JDBC)
~ MySQL
~ Visual Studio Code
~ Git

REFERENCES

~ https://dev.mysql.com/downloads/connector/j/