

# **PROJECT DOCUMENT: BLOOD MANAGEMENT SYSTEM**

## **TABLE OF CONTENTS**

1. Introduction
  - 1.1. Purpose of the Document
  - 1.2. Project Overview
  - 1.3. Scope of the Project
2. System Requirements
  - 2.1. Functional Requirements
3. Architecture
  - 3.1. High-Level Architecture
  - 3.2. Class Diagram
  - 3.3. Sequence Diagram
4. User Interface
5. Technologies Used
6. Testing
  - 6.1. Test Cases
  - 6.2. Unit Testing
7. Conclusion
8. References

## **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

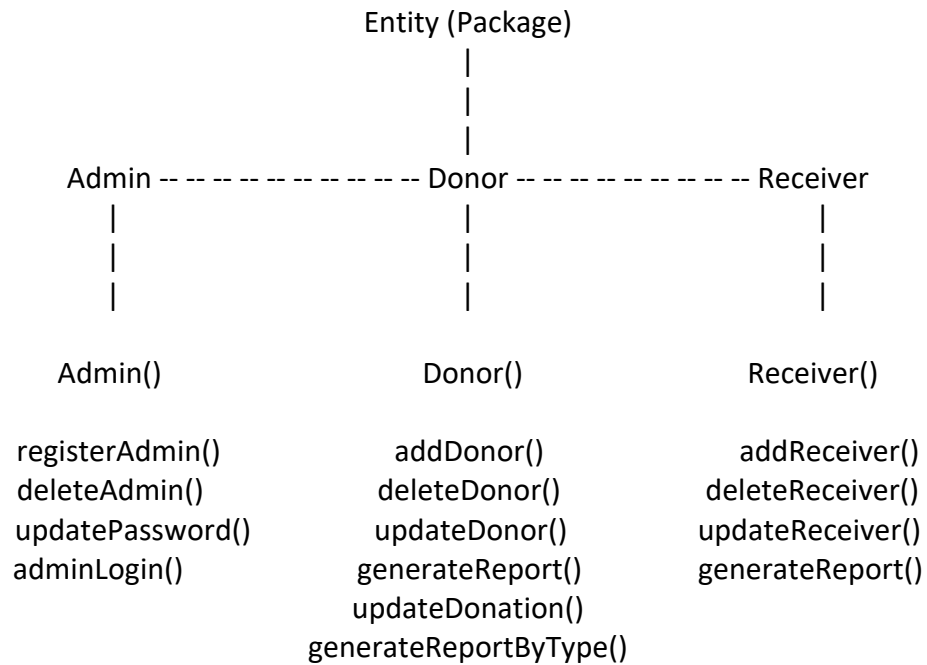
- ~ Register/Delete Admin: Admin Credentials and Database Login (Management) Credentials are required.
- ~ Admin Login/Logout: Admin Credentials are required to login to the system.
- ~ Change Admin Password: Current and New Admin Passwords are required.
- ~ Donor Portal: Add, Update or Delete Donors.
- ~ 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

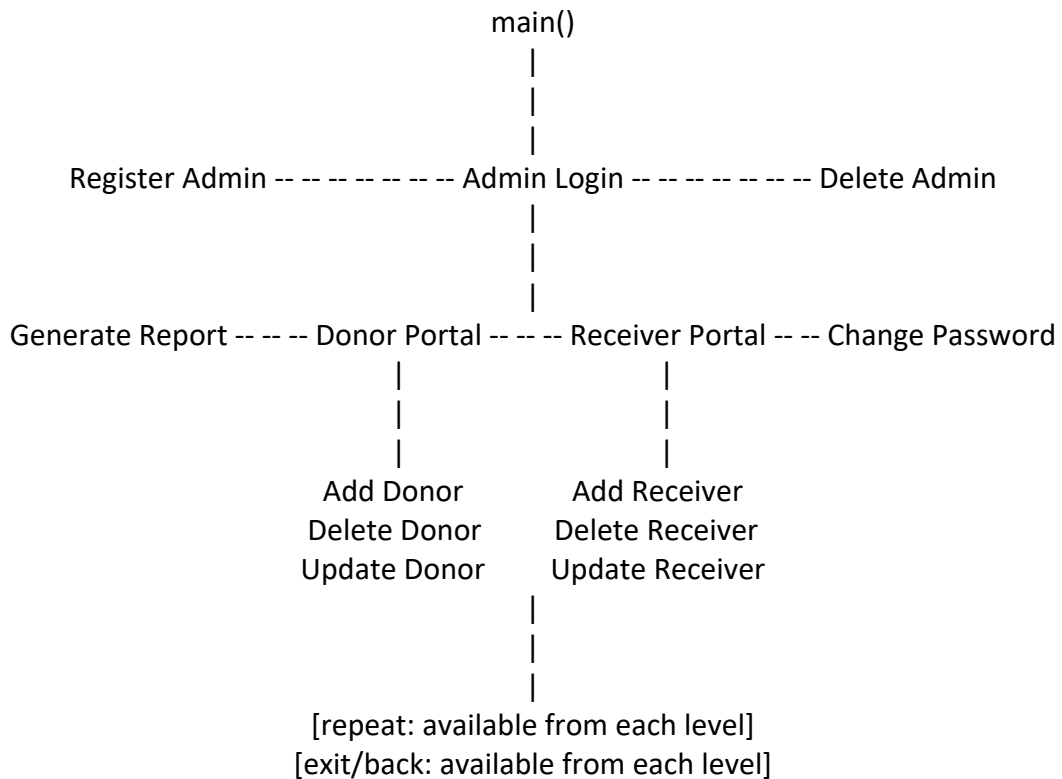
### High-Level Architecture

- ~ 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



### Sequence Diagram



## 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/>