**BLOOD BANK MANAGEMENT SYSTEM**

* A solution to the ever growing requirements of blood due to the road accidents and some health issues

**A MINI PROJECT REPORT**

**Submitted By**

**DEEBAN N**

**KARVENDHAN K**

**PRASHANTH P**

**SAM JOSEPH J**

**VIJAY PRSANTH A**

## ABSTRACT

To build a solution to the ever growing requirement of blood due to the accidents and the various health problems. We have developed a system called “Blood Bank Management System” for accessing the information about various blood banks and hospitals and their blood stock in their respective blood banks. The blood bank management system will provide complete information about the blood donors, and their respective blood demanded person’s information and everything about the process. Donors are provided with registration process to maintain their information for future donations as well as to make their information available to search easily if that donor’s blood group is in need. Person can place the order of the blood group in the blood bank management system during the need of blood for surgeries and for any other medical issues. The blood bank management system will provide the information about the donors who are all near by the location of the hospital, who needs blood. This is how our blood bank management system will work which will mostly use during accident times in order to avoid a life loss due to the unavailability of required blood type.

**HARDWARE REQUIREMENTS**

* Minimum 32 bit (\*86 processor) needed, but 64 bit (\*64 processor) recommended
* Minimum 512 GB needed, but 1 TB recommended
* 4 GB Minimum Ram Required, but 8 GB RAM recommended
* 1024 \* 768 display needed, but 1280 \* 1024 display is recommended

**SOFTWARE REQUIREMENTS**

* IBM Rational Rose Enterprise Edition
* Code blocks for Java Programming (Forward Engineering and Reverse Engineering)
* Operating System : Windows 8 and Later

**ARCHITECTURAL DIAGRAMS**

**UML DIAGRAM**

1. **CLASS DIAGRAM**

****

1. **UseCase Diagram:**

****

1. **Sequence Diagram:**

****

1. **Collaboration Diagram:**

****

1. **State – Chart Diagram:**

****

1. **Deployment Diagram:**

****

1. **Component Diagram:**

****

**MODULE DESCRIPTION**

This module maintains details about the donors and recipients. The module is linked to other modules in the software for wards and OT in the hospital, whereby, any and all blood requirement during surgeries etc. that happens in the hospital are known to the bank. Important information and parameters such as availability of blood, cross-matching between donor’s and recipient’s blood groups and blood transfusion reactions are recorded. Also, the interactions with other blood banks within a hospital or outside and delivery/receipts of blood bags between these banks or hospitals are recorded and maintained.

**Need of Blood Bank Management System:**

Blood Bank donation system can collect blood from many donators in short from various sources and distribute that blood to needy people who require blood.

**Roles:**

1. Admin

2. Blood bank

3. Donor

4. Patient

**ADMIN**

* Manage Registration for user
* Manage Blood bank information like (update, delete)
* Manage Donor Request for Donor
* Manage patient Request for needy people
* Manage Inquiry form for Appropriate Reply
* Manage feedback for Appropriate Reply

**BLOOD BANK**

* Blood bank information view/update
* View Donor information
* Manage Patient Blood Request.

**DONOR**

* Manage Donor information(profile)
* Add new Donation for Blood

**PATIENT**

* Manage patient information(profile)
* Give the Request for patient for blood

**CLASSES**

Class diagram is a static diagram. It represents the static view of an application. Class diagram is not only used for visualizing, describing, and documenting different aspects of a system but also for constructing executable code of the software application.

Class diagram describes the attributes and operations of a class and also the constraints imposed on the system. The class diagrams are widely used in the modeling of object oriented systems because they are the only UML diagrams, which can be mapped directly with object-oriented languages.

Class diagram shows a collection of classes, interfaces, associations, collaborations, and constraints. It is also known as a structural diagram.



Blood Class is used to do the operations of add and subtract the count of blood types i.e. If someone has taken the blood or donated the blood we have to subtract the number of count or Add the number of count of the blood type to the database. This can be done by the Inventory Manager.



Lab Technician is used to test the type of blood and also check the blood is infected or not if suppose the blood is infected then they will not use those blood to the other persons but they will use those blood for the experiments. If the blood is well and good then they will accept the blood if not they will reject the blood. After the Blood is accepted then they will given to the Blood Class to count the blood.



Organization admin’s work is to hire the Person to act as a staff in their respective places like Nurse, Doctors, Lab Technicians, and Receptionists and so on.



Nurse work is to check the patients and take the blood from the patients and give to the lab technicians.



Inventory Manager has to keep track of the blood in the database. If anyone blood type is reduced or finished off or needed more the inventory manager has to order for the blood type from the nearby blood bank and process the blood type. They can accept the blood or also they can reject the blood from the donors or from nearby blood bank.



This module deals with two primary functions Register Donor(), and Receptionist(). The Register Donor function is used to collect the information from the donor one who donates the blood for the first time, that data will be stored in the database. And for future emergency of blood we can use the database to gather the information about the donor.

And the second function named Receptionist is used by the receptionist people in the hospital for the purpose of ordering the blood type, purchasing the blood etc, during accident and surgical time.



This module will hold two functions are Donor() and Donate(). The Donor function will collect the information like name, blood group, age, address, phone number etc, in order to use that person’s blood for the future needs.

The second module Donate function is used to inform to the blood donor by the hospital receptionist during emergency need of blood.



This module hold three functions Order blood(), Purchase blood() and hospital(). The hospital receptionist will order the blood by using Order blood function. And the ordered blood information will go to the database and send this information to the donor who gets matched with the requested blood group.

If the donor accepts the request for donating the blood to that hospital the purchase function will intimate back to the hospital that the donor is ready to donate the blood.

And the hospital function will hold the details of both the donor details as well as the patient detail who receive the blood from the donor.

1. **Forward Engineering:**

**Output:**

private class Blood

{

private int Blood\_Type;

private int Code;

private int Price;

private char Info;

public Lab\_Technician theLab\_Technician;

/\*\*

\* @roseuid 5CA231270147 \*/

public Blood()

{

}

/\*\*

\* @roseuid 5CA0EA8F036A

\*/

public void Add()

{

}

/\*\*

\* @roseuid 5CA0EA9500A1

\*/

public void Subtract()

{

}

}

private class Donor

{

private char Name;

private int Age;

private char Email;

private int Phone\_No;

private char Address;

/\*\*

\* @roseuid 5CA231270163

\*/

public Donor()

{

}

/\*\*

\* @roseuid 5CA0EA1700B9 \*/

public void Donate()

{

}

/\*\*

\* @roseuid 5CA0EA1B023C

\*/

public void Register\_For\_Camp()

{

}

}

private class Hospital

{

private char Name;

private int Phone;

private char Fax;

private char Founded;

public Inventory\_Manager theInventory\_Manager;

/\*\*

\* @roseuid 5CA231270184

\*/

public Hospital()

{

}

/\*\*

\* @roseuid 5CA0EA2A036E

\*/

public void Order\_Blood()

{

}

/\*\*

\* @roseuid 5CA0EA380021 \*/

public void Purchase\_Blood()

{

}

}

private class Inventory\_Manager

{

private char Name;

private int Phone\_No;

private char Email;

private char Address;

public Blood theBlood;

/\*\*

\* @roseuid 5CA2312701A0

\*/

public Inventory\_Manager()

{

}

/\*\*

\* @roseuid 5CA0EA4700E6

\*/

public void Process\_Orders()

{

}

/\*\*

\* @roseuid 5CA0EA520076

\*/

public void Accept\_Blood()

{

}

/\*\*

\* @roseuid 5CA0EA5C01A6 \*/

public void Reject\_Blood()

{

}

}

private class Lab\_Technician

{

private char Name;

private int Phone\_No;

private char Email;

private char Address;

public Blood theBlood;

/\*\*

\* @roseuid 5CA2312701BC

\*/

public Lab\_Technician()

{

}

/\*\*

\* @roseuid 5CA0EA9D034F

\*/

public void Test\_Blood()

{

}

/\*\*

\* @roseuid 5CA0EAA30109

\*/

public void Accept\_Blood()

{

}

/\*\*

\* @roseuid 5CA0EAAA02C0 \*/

public void Reject\_Blood()

{

}

}

private class Nurse

{

private char Name;

private int Phone\_No;

private char Address;

/\*\*

\* @roseuid 5CA2312701D0

\*/

public Nurse()

{

}

/\*\*

\* @roseuid 5CA0EAB600B5

\*/

public void Barcode\_Blood()

{

}

}

private class Organization\_admin

{

private char Name;

private int License\_No;

public Nurse theNurse;

public Inventory\_Manager theInventory\_Manager;

/\*\*

\* @roseuid 5CA2312701F8

\*/

public Organization\_admin()

{

}

/\*\*

\* @roseuid 5CA0EA6800F5

\*/

public void Register\_Staff()

{

}

}

private class Receptionist

{

private char Name;

private int Phone\_No;

private char Email;

private char Address;

public Donor theDonor;

public Hospital theHospital;

/\*\*

\* @roseuid 5CA231270223

\*/

public Receptionist()

{

}

/\*\*

\* @roseuid 5CA0EA00016B

\*/

public void Register\_Donor()

{

}

} /\*\* \* void Donor.get(){ \* } \*/

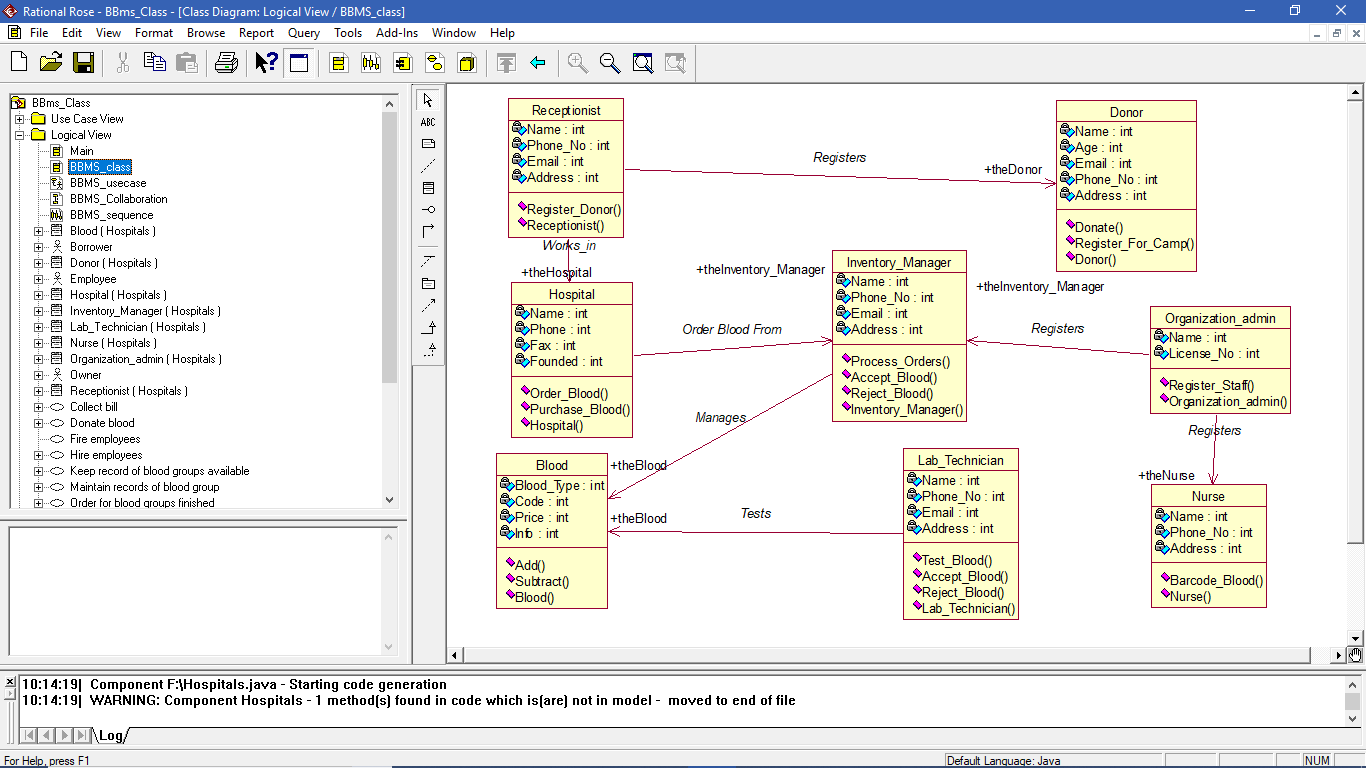
1. **Reverse Engineering:**

**Output:**

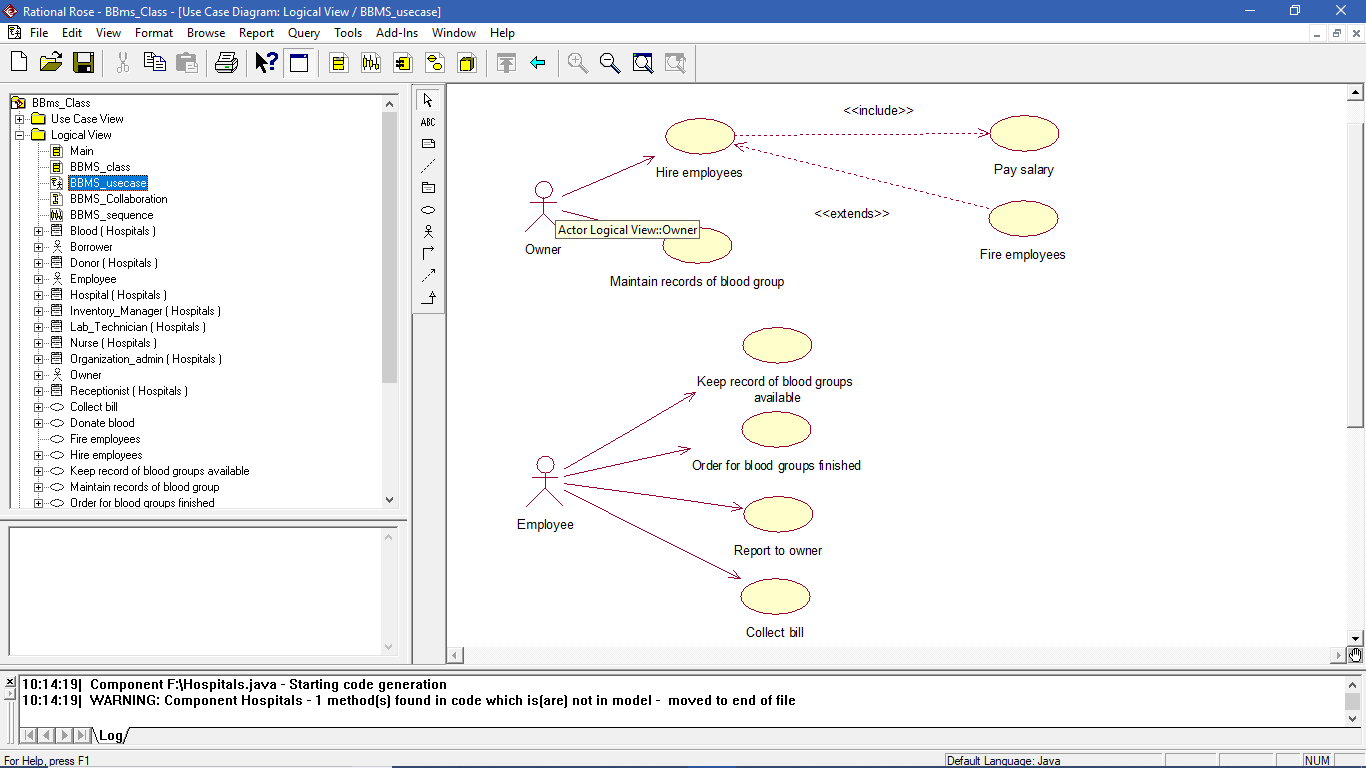
****

**SAMPLE SNAP SHOTS**

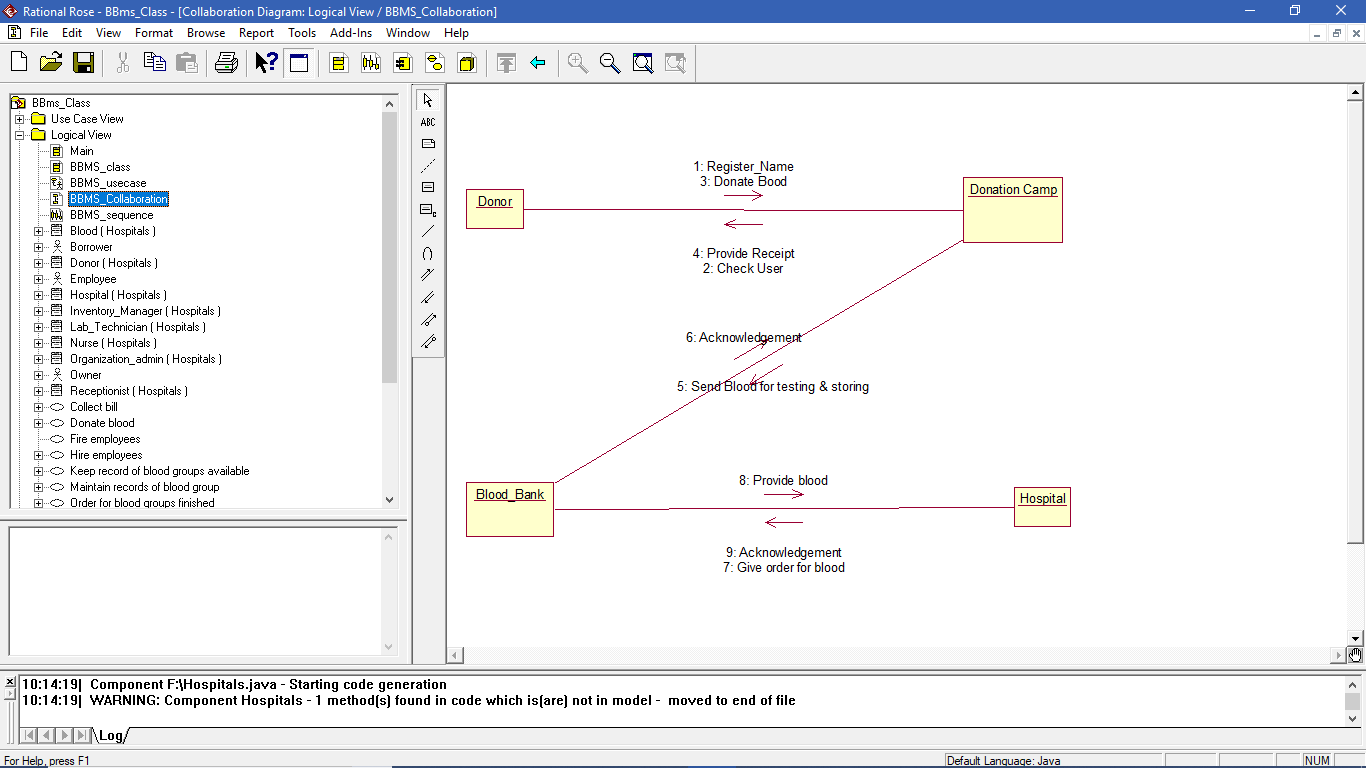
1. **Class Diagram**

****

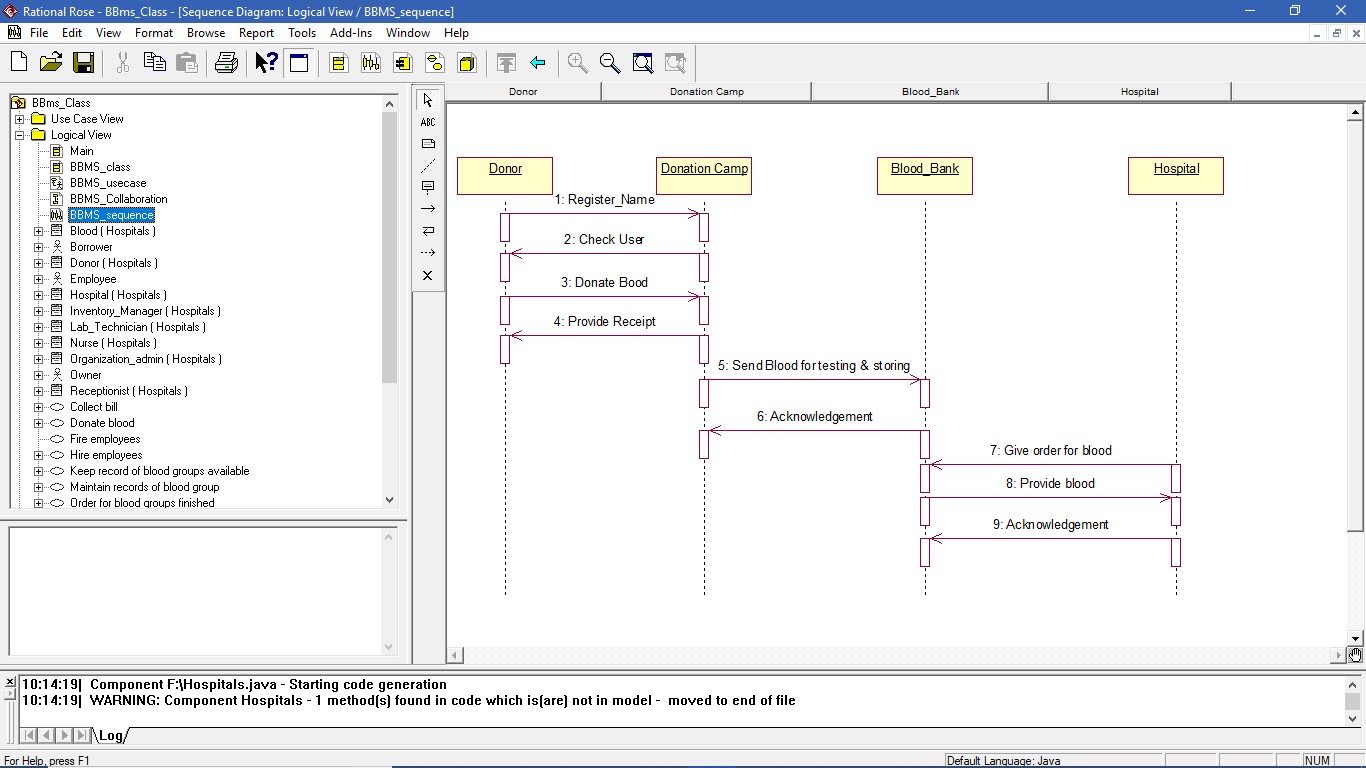
1. **Use case Diagram**

****

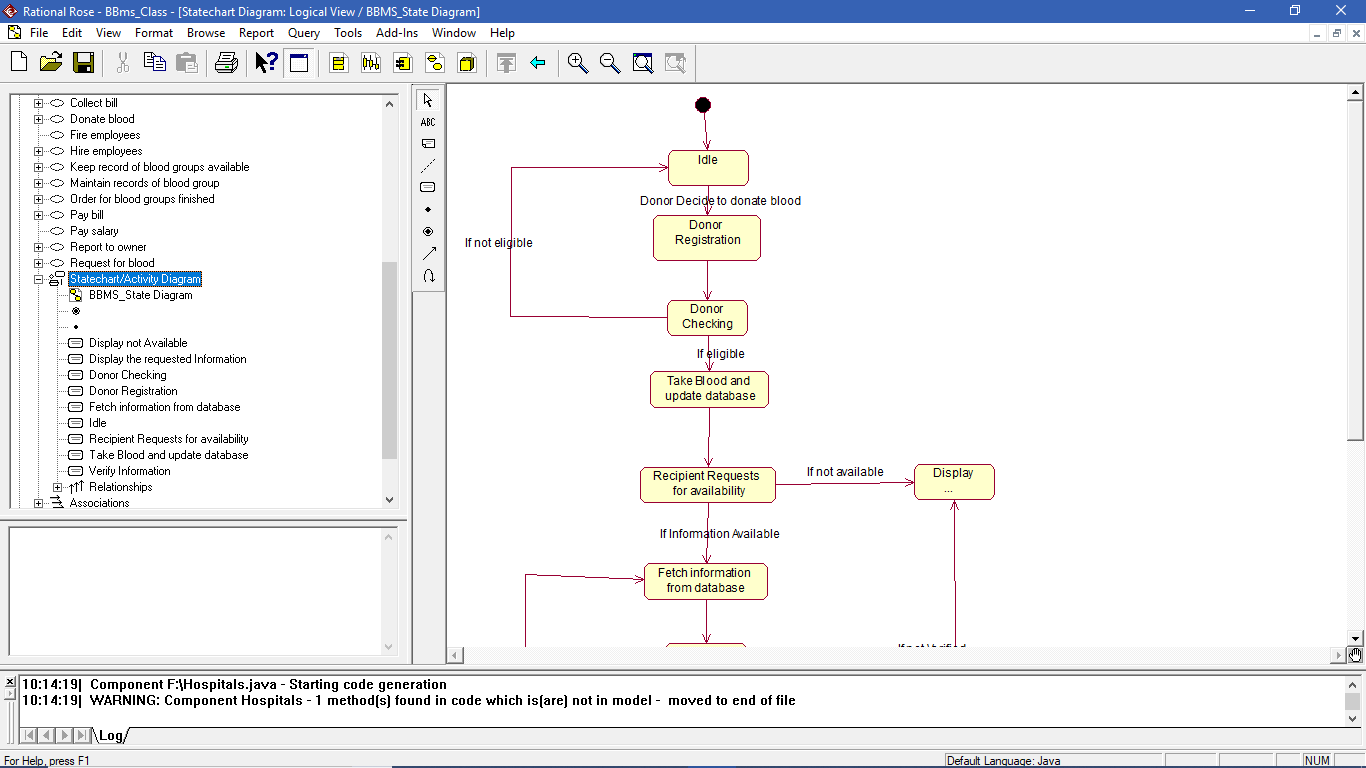
1. **Collaboration Diagram**

****

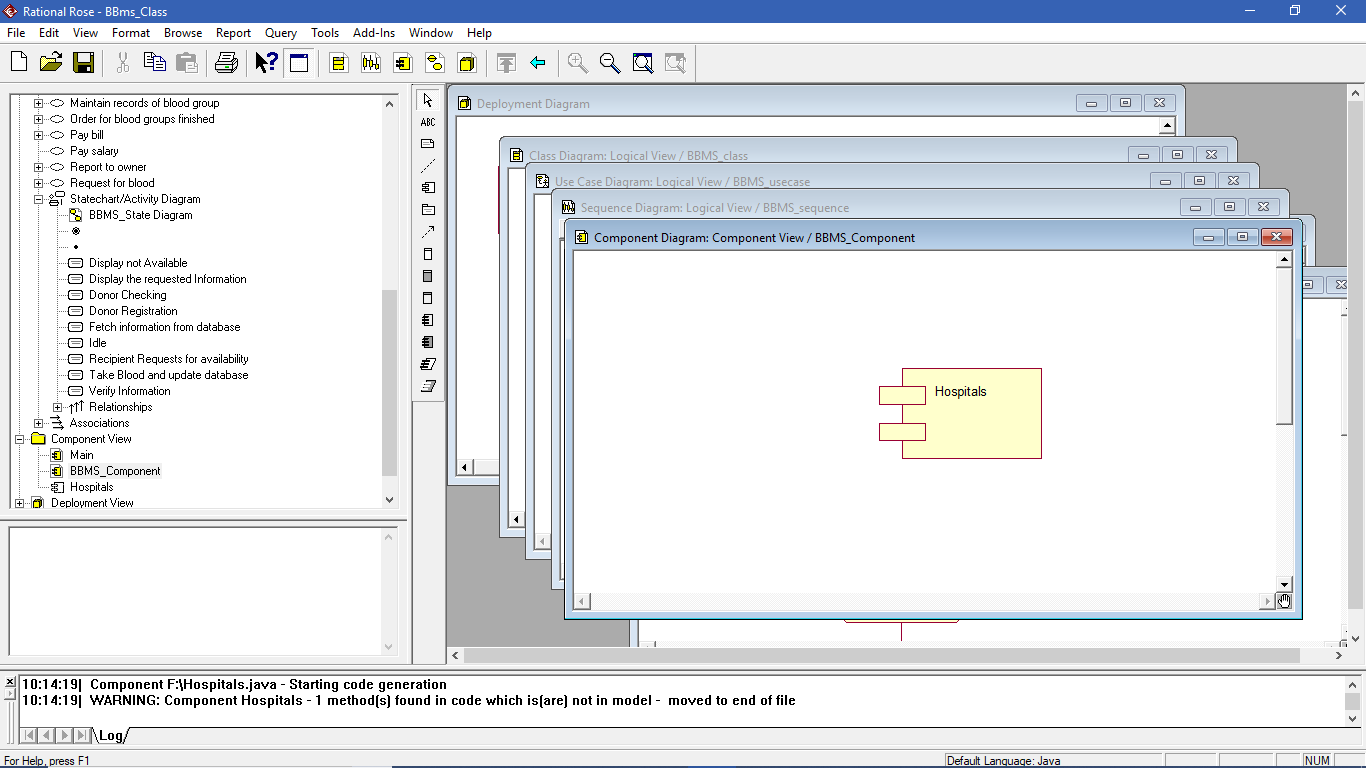
1. **Sequence Diagram**

****

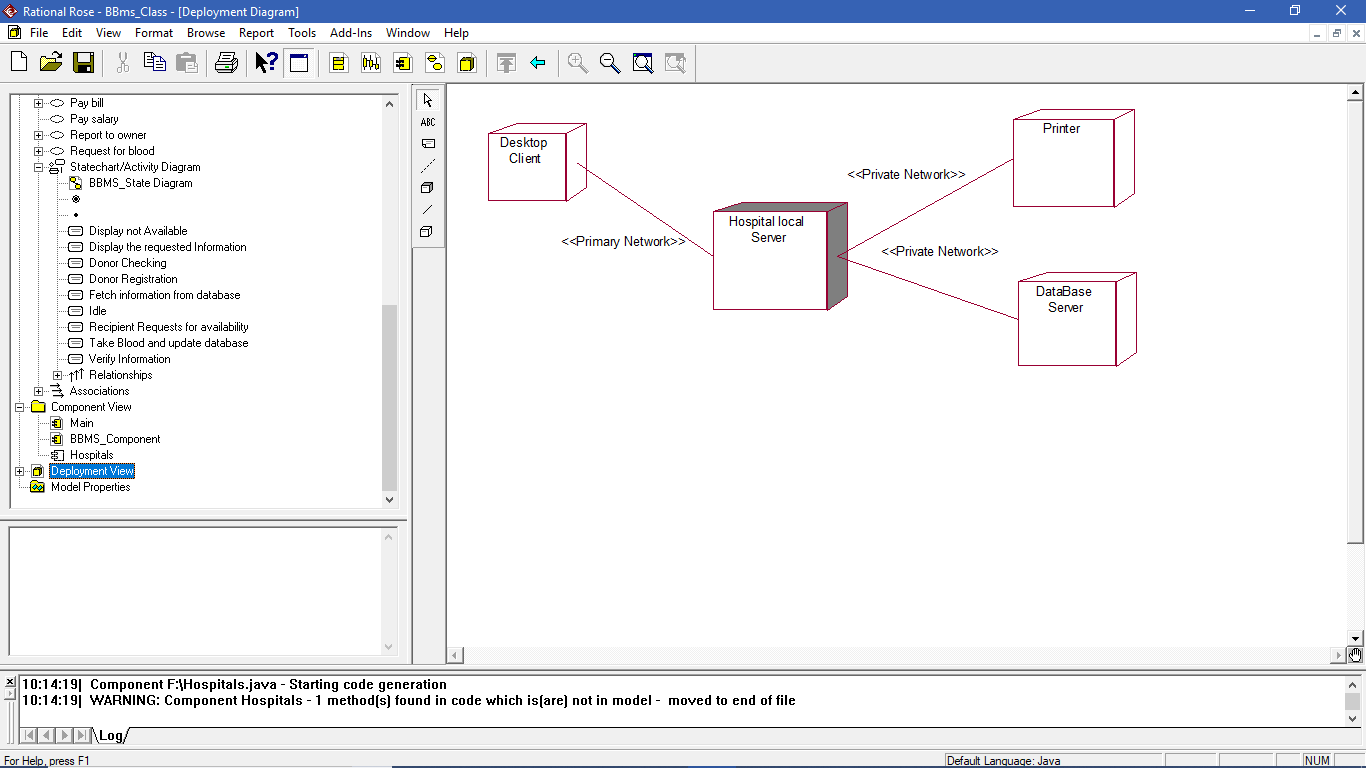
1. **State Diagram**

****

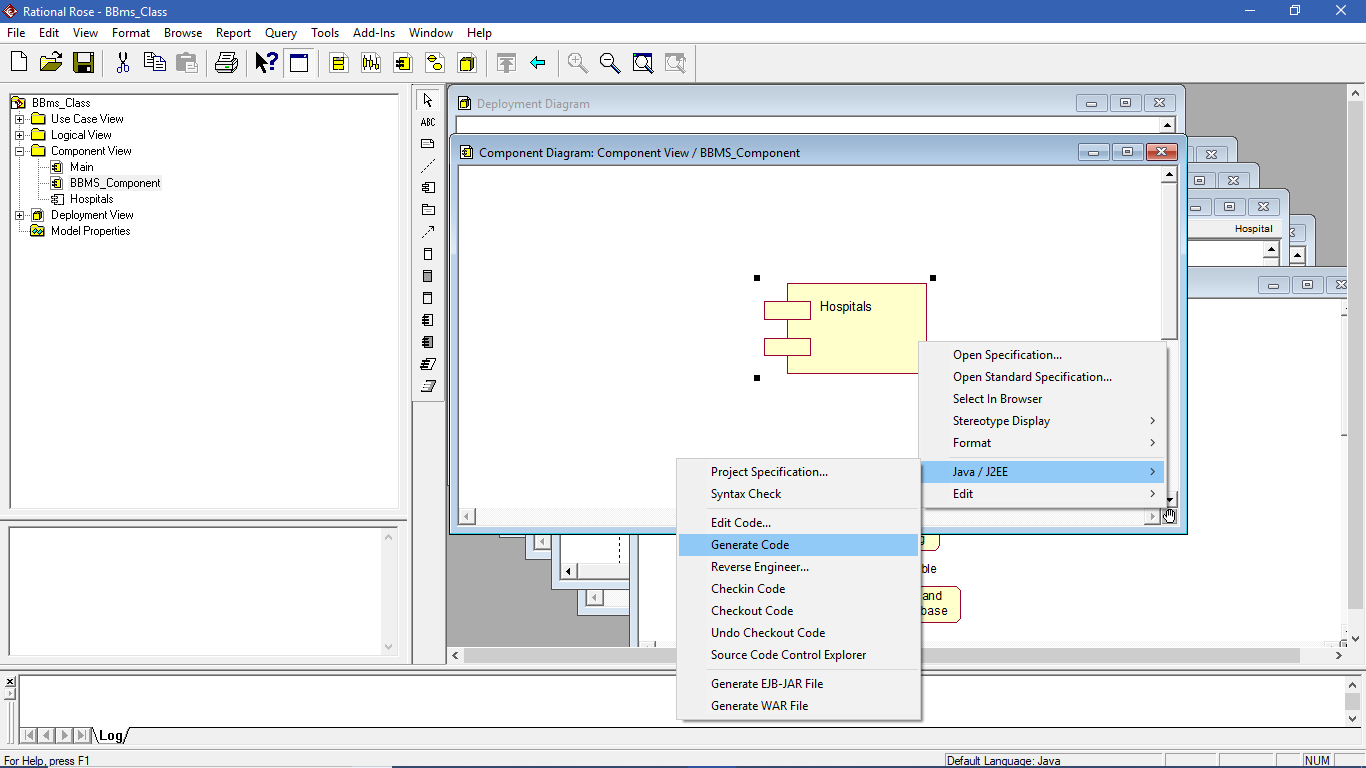
1. **Component Diagram**

****

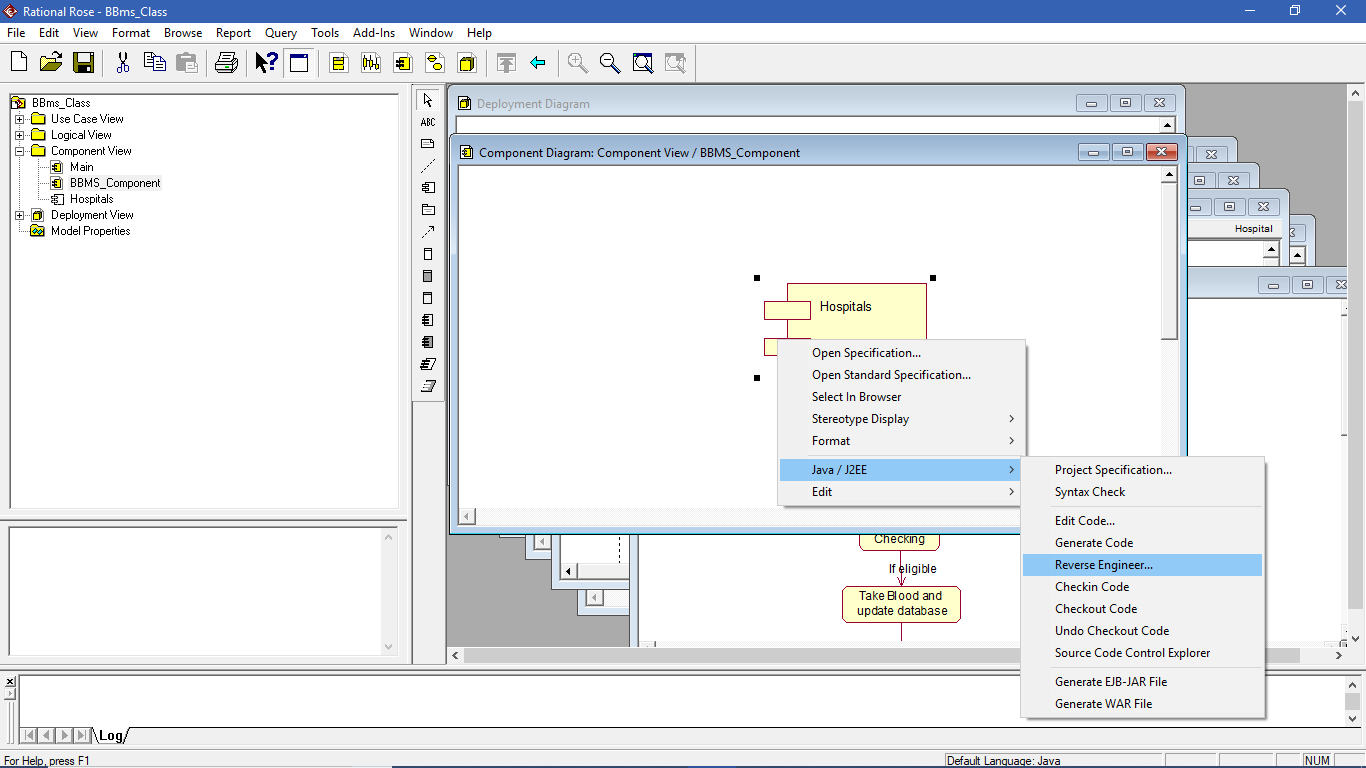
1. **Deployment Diagram**

****

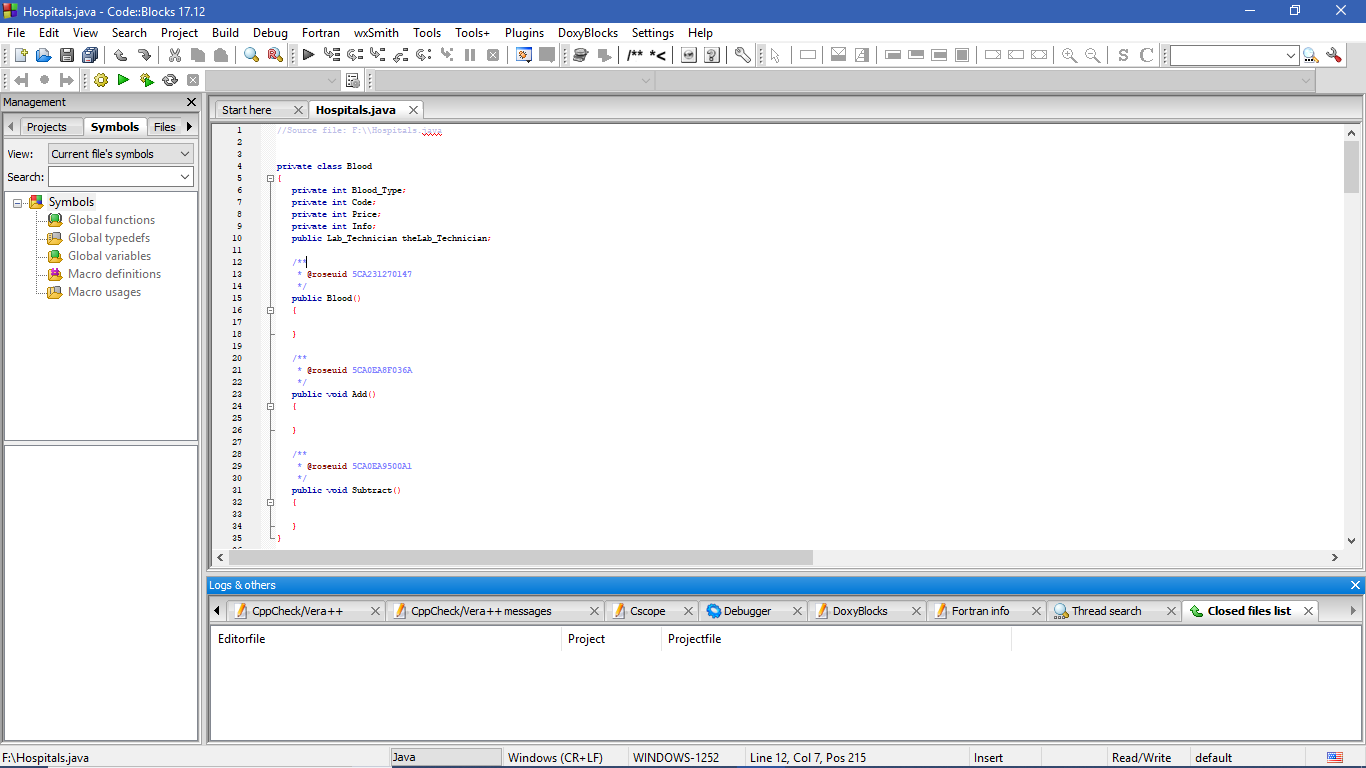
1. **Generate code (Forward Engineer)**



1. **Reverse Engineer**

****

1. **Code for the drawn Class Diagram**

****

**CONCLUSION**

This system computerizes the blood transaction process in any blood bank. It manages the different operations in blood bank such as blood collection process, storage and uses through a scientific manner during an emergency situation. It has the searching option along with the managing capability. New features and modules can be added into the system as per user requirement. The proposed system can be used to reduce the time required to deliver required blood to the needy in cases of emergency.