**Chapter 1 OBJECTIVE AND SCOPE OF THE PROJECT**

India's blood banking system has serious shortcomings. The gap between demand and supply of blood is continuously widening. India has an annual requirement of approximately, 5.0 million units of blood. The actual collection is only approximately 3.50 million units. A study conducted by the National AIDS Control Organization (NACO), regarding blood banking services in India has revealed many shortcomings, including the decentralized nature of blood services, a shortage of human, technological and financial resources and a deficit in the availability of blood, especially from voluntary donors. Paradoxically, very few blood banks are operating to their full capacity. Inappropriate use of blood and wastage is not an uncommon occurrence. Even during an emergency, the onus is on the patient's relatives to arrange for replacement of blood.

Blood Bank Management System, the portal bridges the gap between the demand and supply of blood. This portal aims to bring blood donors and recipients under a common on-line platform. Donors can register themselves on the system after going through the basic requirements for donating blood. This portal also has useful information regarding blood donation.

**Chapter 2 THEORETICAL BACKGROUND AND PROBLEM DEFINATION**

Understanding the problem in the existing system & finding requested solution is the most important activity while planning the project. Hence the developing a new system we must get through problem associated with the current system.

* Donors do not have any record of their donations or information related to their blood diseases.
* For the number of Donors, recipients it should be difficult to maintain the data in no. of registers and handling the process of blood donation and sales manually**.**
* Also the calculations part that is for selling and buying the Blood is also handled manually in previous system.

The manpower required for this kind of transaction and maintenance of data is higher than the actual requirement.

**Chapter 3 SYSTEM ANALYSIS**

**3.1 REQUIREMENT ANALYSIS**

**3.1.1 FUNCTIONAL REQUIREMENTS**

**a) Strong Data Validation:**

There is possibility that user might enter wrong data and wrong data may cause inconsistency to the database and hence to the system. To avoid this, data should be validated whenever entered.

**b) Automatic updating of the database:**

After any transaction is performed, it is necessary that the updating should be reflected in the database without any inconsistency.

**c) Provide efficiency querying based on user requests:**

The major purpose is to generate efficient reports on any user request. This will be done by our query processing system, which should be able to process any combination of queries will be done dynamically at run time depending on the user

**3.1.2 EXTERNAL INTERFACE REQUIREMENTS**

**a) User friendly interface:**

The interface should be developed in such a manner that it is very user friendly, this not only improve interaction but also saves data entry time.

**b) Making well designed forms for capturing data:**

The forms for capturing the data should be well-designed using pop-down menus and drag & drop facilities, which reduce the data entry effort on the part of the user.

**3.1.3 PERFORMANCE REQUIREMENTS**

**a) Security:**

All users are not allowed to access the database. Hence there is a need to check authority of every user. Username and Password validation helps to deny unauthorized access to the system.

There are 2 main types of users who will be using the software

They are:-

1) Admin

2) User

Each user is given the specific rights to access the data in Read only, Read Write, Delete.

**.2 ERD**

Maintains

Maintains

tbl\_Login

Maintains

tbl\_Stock

tbl\_DonorDetails

Register

Donor

tbl\_RecipientDetails

Register

Recipient

Blood bank Management System

Maintains

tbl\_DestroyedBlood

Maintains

tbl\_ExpiredBlood

I

Manages

tbl\_DonationDetails

I

tbl\_Sales

A

* 1. **DATA FLOW DIAGRAM**

Search and delete the Data

Admin

Check User Data

Details

Generate Bill

Add Recipient

User

Add Donor

Enter User Detail

User User Profile

Details

Donor Donor Record

Details

Recipient Recipient Details

Details

Sales Bill Record

Details

* 1. **CONTEXT LEVEL DAIGRAM**

5. Donation Details

Admin

Donor

11.Blood Destroyed

10.Blood Expiry

8.Generate Bill

6.Sales Details

4.Donation Details

3.Recipient Details

2. Donor Details

1.Login

7. Sales Details

9. Bill Details

Recipient

* 1. **Methodology adopted, system implementation and details of hardware and software used**

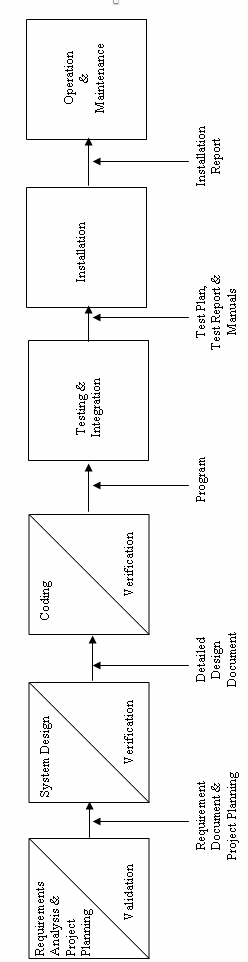
**3.5.1 PROCESS MODEL – SOFTWARE ENGINEERING**

In the development of software we have used **Waterfall Model**, the linear sequential mode. This model encompasses the following activities:

**a) Analysis Phase:**

System Analysis:-

This refers to the gathering of system requirements, with the goal of determining how these requirements will be accommodated in the system.



**b) System Design Phase:**

This is actually a multistep process. In this we tried to focus on some distinct attributes of a program like data structure, software architecture, interface representations and algorithmic detail. In this we tried to translate requirements into representation of the software which can be assessed for quality before coding begins. In the verifications, I have tried to ensure that the design is satisfying the requirements and is of good quality. I have tried to find out if there is any misinterpretation of specified any requirements.

**c) Code Generation Phase:**

In this phase, we translated design of a system into code which can be compiled and executed. In this phase we have done actual coding for all forms. In this we tried to produce simple program which are clear to understanding and modify.

We have used dynamic method to verify the code. We have executed program on some test data and output of the program examined to determine if there are any error present. I have read the code carefully to detect any discrepancies between the design specification and the actual implementation.

**d) Testing:**

Testing plays a critical role in quality assurance for software. Due to limitations of the verification methods for the previous phase, design and requirement faults also appear in the code. Testing is used to detect these errors, in addition to the errors introduced during the coding phase.

# 3.5.2 TOOLS/ENVIRONMENT USED

# SOFTWARE / HARDWARE REQUIREMENTS SPECIFICATION

**PLATFORM:**  Windows XP Professional

**FRONT END:** Visual Studio 2008.

**BACK END:** SQL Server 2005

**HARDWARE REQUIREMENTS:**

Intel Pentium III 733 MHz or Higher.

256 MB RAM or Higher.

**3.5.3 SYSTEM DESIGN**

# DATA STRUCTURE

**1) Table Name: tbl\_DonorDetails**

|  |  |  |  |
| --- | --- | --- | --- |
| **FIELD** | **SIZE** | **TYPE** | **DESCRIPTION** |
| Donor\_ID | 50 | varchar(50) | Donor ID |
| Donor\_Name | 50 | varchar(50) | Donor Name |
| Donor\_Gender | 50 | varchar(50) | Gender |
| Donor\_Address |  | varchar(MAX) | Address |
| Donor\_Contact | 50 | varchar(50) | Contact Number |
| Donor\_Group | 50 | varchar(50) | Blood Group |
| Donor\_Type | 50 | varchar(50) | Donor Type |
| Donor\_HemoglobinEST | 50 | varchar(50) | Hemoglobin Test |
| Donor\_HIVStatus | 50 | varchar(50) | HIV Status |
| Donor\_PhysicalExam | 50 | varchar(50) | Physical Problems if any |
| Donor\_Age | 50 | varchar(50) | Age |

**2) Table Name: tbl\_RecipientDetails**

|  |  |  |  |
| --- | --- | --- | --- |
| **FIELD** | **SIZE** | **TYPE** | **DESCRIPTION** |
| Recipient\_ID | 50 | varchar(50) | Recipient ID |
| Recipient\_Name | 50 | varchar(50) | Name |
| Recipient\_Gender | 50 | varchar(50) | Gender |
| Recipient\_Address |  | varchar(MAX) | Address |
| Recipient\_Age | 50 | varchar(50) | Age |
| Recipient\_Contact | 50 | varchar(50) | Contact Number |
| Recipient\_BloodGroup | 50 | varchar(50) | Blood Group |

**3) Table Name: tbl\_Login**

|  |  |  |  |
| --- | --- | --- | --- |
| **FIELD** | **SIZE** | **TYPE** | **DESCRIPTION** |
| username | 50 | varchar(50) | Username |
| password | 50 | varchar(50) | Password |

**4) Table Name: tbl\_DonationDetails**

|  |  |  |  |
| --- | --- | --- | --- |
| **FIELD** | **SIZE** | **TYPE** | **DESCRIPTION** |
| Serial\_No | 50 | varchar(50) | Serial Number |
| Purchase\_Date |  | smalldatetime | Donation Date |
| Blood\_Group | 50 | varchar(50) | Blood Group |
| Donor\_Name | 50 | varchar(50) | Donor Name |
| Purchase\_Quantity | 50 | varchar(50) | Blood Quantity |
| Blood\_BagNo | 50 | varchar(50) | Bag Number |
| Blood\_ExpiryDate |  | smalldatetime | Expiry Date |

**5) Table Name: tbl\_Sales**

|  |  |  |  |
| --- | --- | --- | --- |
| **FIELD** | **SIZE** | **TYPE** | **DESCRIPTION** |
| Serial\_No | 50 | varchar(50) | Serial Number |
| Sales\_Date | 50 | smalldatetime | Sales Date |
| Patient\_Name | 50 | varchar(50) | Recipient Name |
| Blood\_Group | 50 | varchar(50) | Blood Group |
| Sales\_Quantity | 50 | varchar(50) | Quantity |
| Blood\_ExpiryDate | 50 | smalldatetime | Expiry Date |
| Sales\_Amount | 50 | varchar(50) | Amount |
| Sales\_BagNo | 50 | Varchar(50) | Blood Bag No |

**6) Table Name: tbl\_Stock**

|  |  |  |  |
| --- | --- | --- | --- |
| **FIELD** | **SIZE** | **TYPE** | **DESCRIPTION** |
| Blood\_Group | 50 | varchar(50) | Blood Group |
| Blood\_Quantity |  | numeric | Available Quantity |

**7) Table Name: tbl\_Product**

|  |  |  |  |
| --- | --- | --- | --- |
| **FIELD** | **SIZE** | **TYPE** | **DESCRIPTION** |
| Product\_SR | 50 | varchar(50) | Serial Number |
| Product\_Name | 50 | varchar(50) | Product Name |
| Product\_Rate | 50 | varchar(50) | Product Rate |

**8) Table Name: tbl\_Purchases**

|  |  |  |  |
| --- | --- | --- | --- |
| **FIELD** | **SIZE** | **TYPE** | **DESCRIPTION** |
| Serial\_No | 50 | varchar(50) | Serial Number |
| Purchase\_Date |  | smalldatetime | Purchase Date |
| Blood\_Group | 50 | varchar(50) | Blood Group |
| Donor\_Name | 50 | varchar(50) | Donor Name |
| Purchase\_Quantity | 50 | varchar(50) | Purchase Quantity |
| Blood\_BagNo | 50 | varchar(50) | Blood Bag No |
| Blood\_ExpiryDate |  | smalldatetime | Expiry Date |

**9) Table Name: tbl\_ProductSales**

|  |  |  |  |
| --- | --- | --- | --- |
| **FIELD** | **SIZE** | **TYPE** | **DESCRIPTION** |
| Sales\_SR | 50 | varchar(50) | Serial Number |
| Sales\_Date |  | smalldatetime | Sales Date |
| Sales\_Patient | 50 | varchar(50) | Patient Name |
| Sales\_Product | 50 | varchar(50) | Product Name |
| Sales\_Rate | 50 | varchar(50) | Rate |
| Sales\_Quantity | 50 | varchar(50) | Quantity |
| Sales\_Amount | 50 | varchar(50) | Amount |

**3.5.4 TESTING**

Software testing is a critical element of software quality assurance and represents the ultimate review of specification, design, coding.

**Testing objectives**

1. Testing is a process of executing a program with the intent of finding an error.

2. A good test case is the one that has high portability of finding an as-yet undiscovered error.

3. A successful test is one that uncovers an as-yet discovered error.

The main objective here will be to design test cases to uncover different classes of errors and to do so with minimum amount of time and efforts. If testing is conducted successfully it will uncover errors in the software. Another advantage is that it demonstrates that software functions appear to be working according to the specifications and performance requirement have been met.

But Testing cannot show the absence of defects it can show only that software errors are present.

**Strategies used for software testing**

The software engineering process is viewed as spiral as shown:

**Unit testing** begins at the vortex of the spiral and concentrates at the each unit of the software as implemented in the source code. Unit testing assures each module tested individually functions properly as a unit.

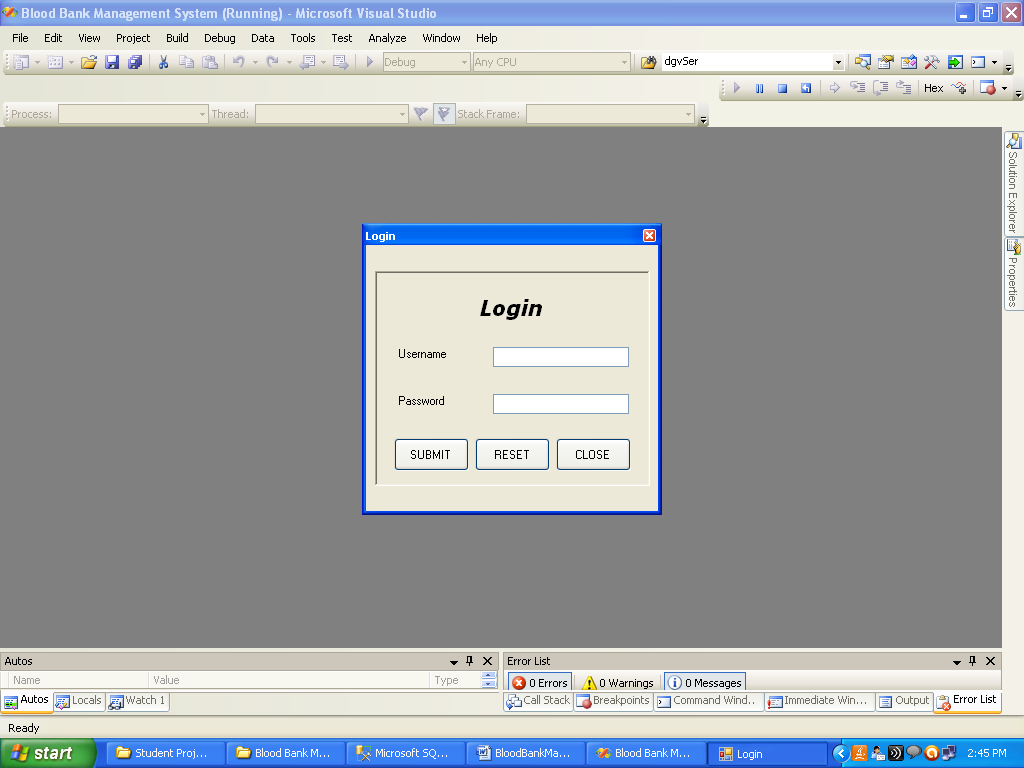
**Integration testing** focuses on the design and construction of the software architecture. It is a systematic technique for constructing a program structure while conducting tests to uncover errors associated with interfacing. The objective is to take unit-tested module and build a program structure that has been specified as design.

**Validation testing** takes care of the requirements established as part of the software requirements analysis are validated against the software that has been constructed. It is said to be successful when the software functions in a manner that can be reasonably expected by the customer.

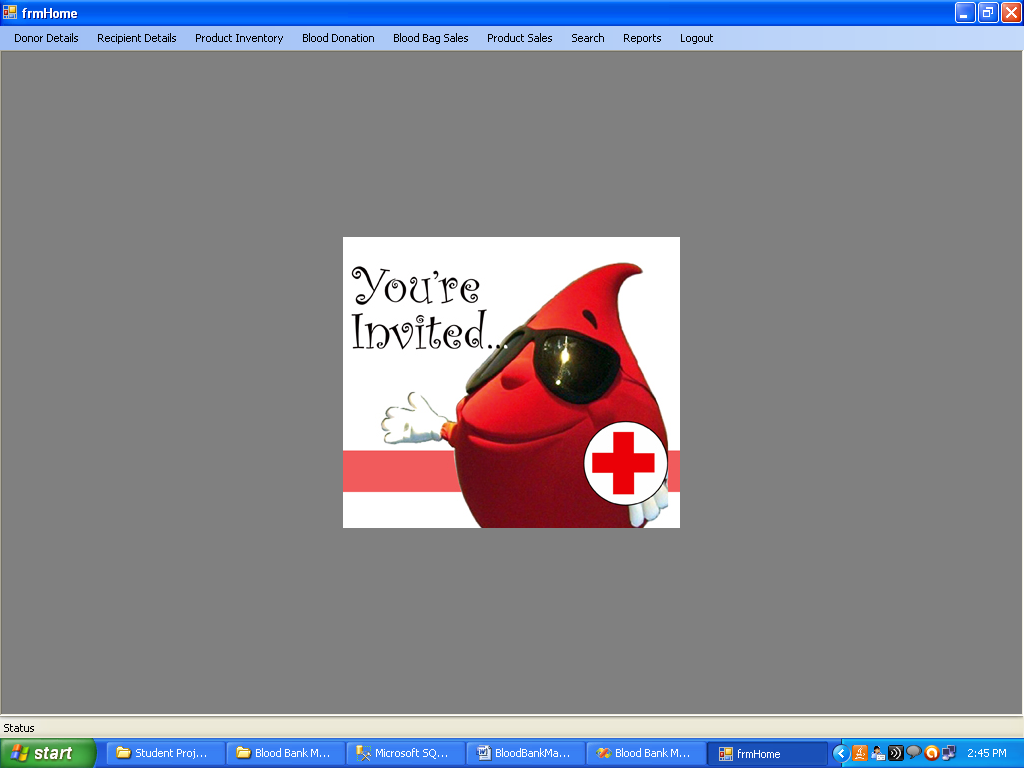
**System testing** tests software and other system elements as a whole. These tests fall outside the scope of software engineering process and are not conducted solely by the software developer.

**3.5.5 DATA MODULES AND THEIR DESCRIPTION**

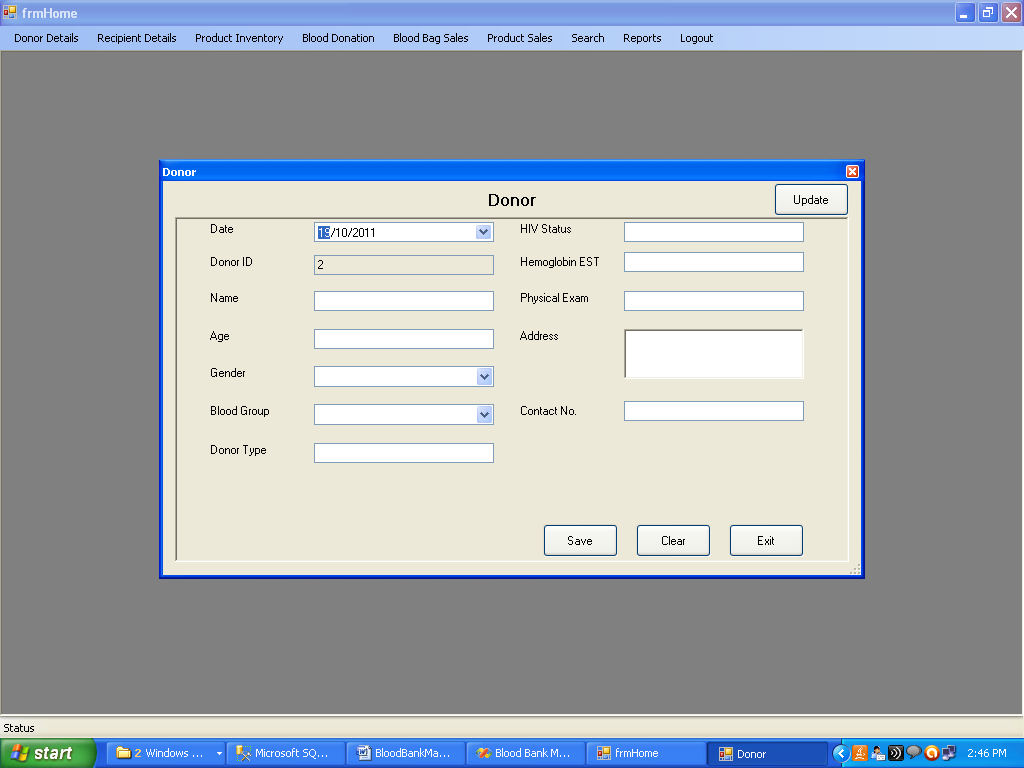
1. **Login:**

****

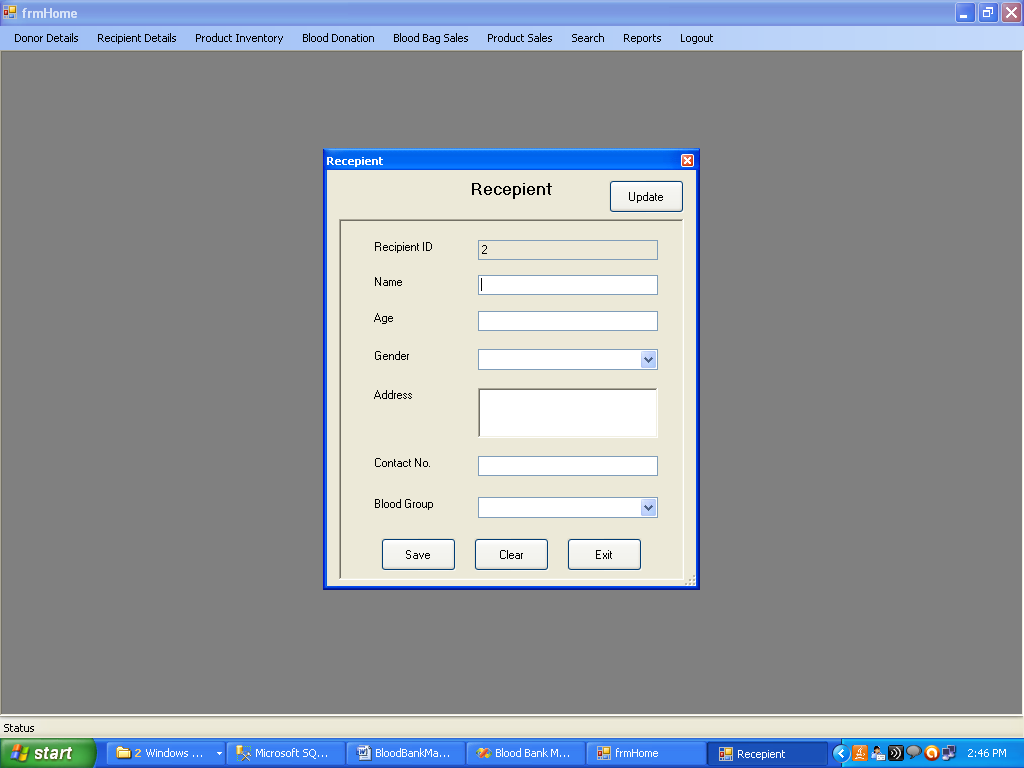
1. **Home Page:**

****

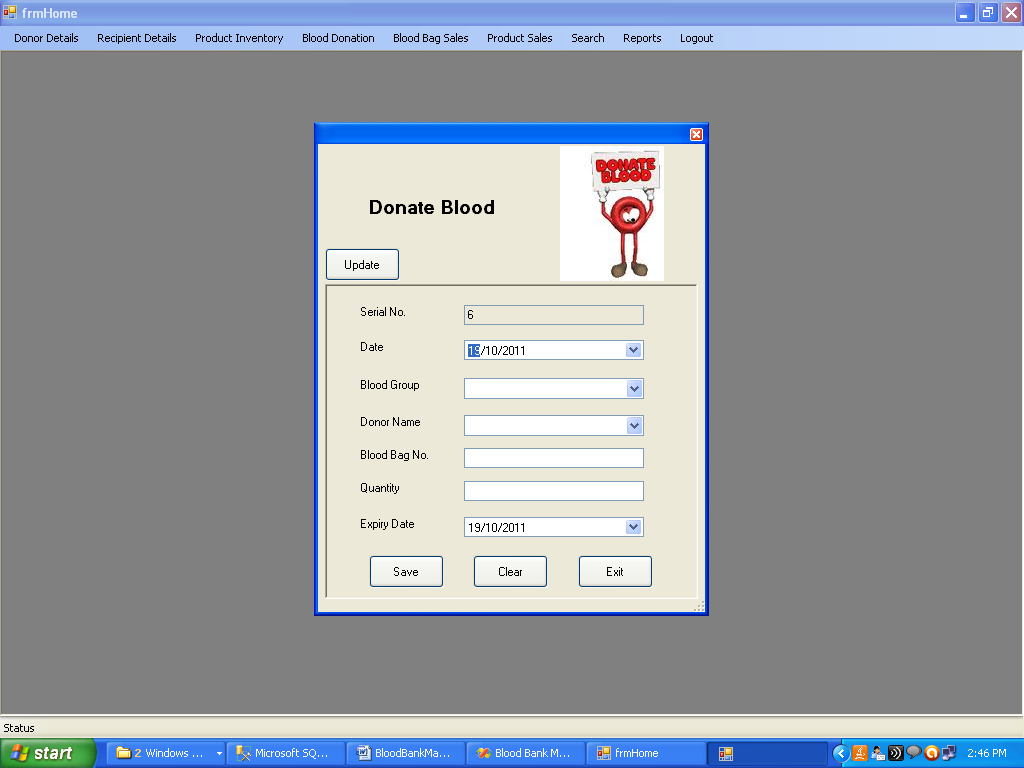
1. **Donor Details:**

****

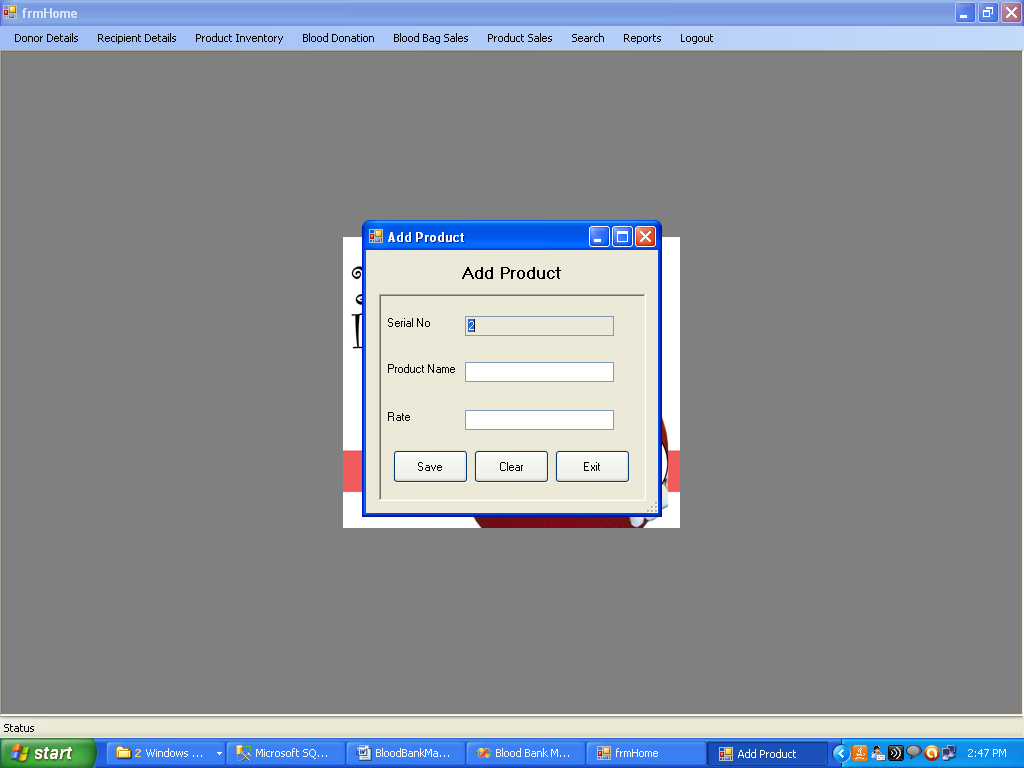
1. **Recipient Details:**

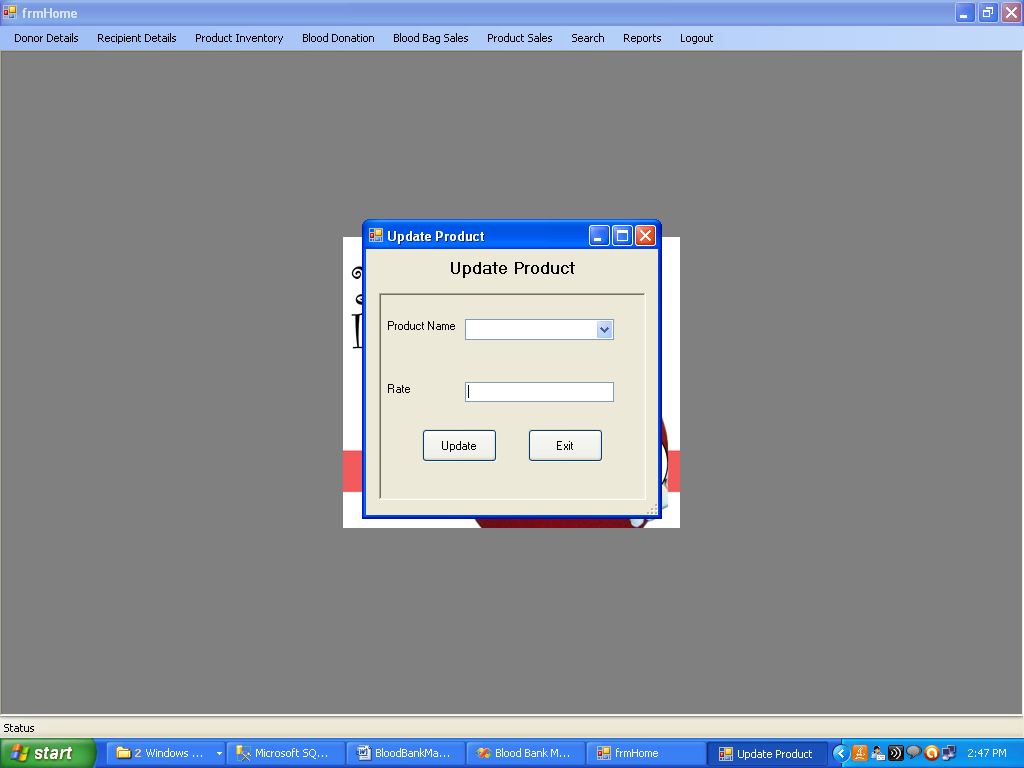
****

1. **Donation:**

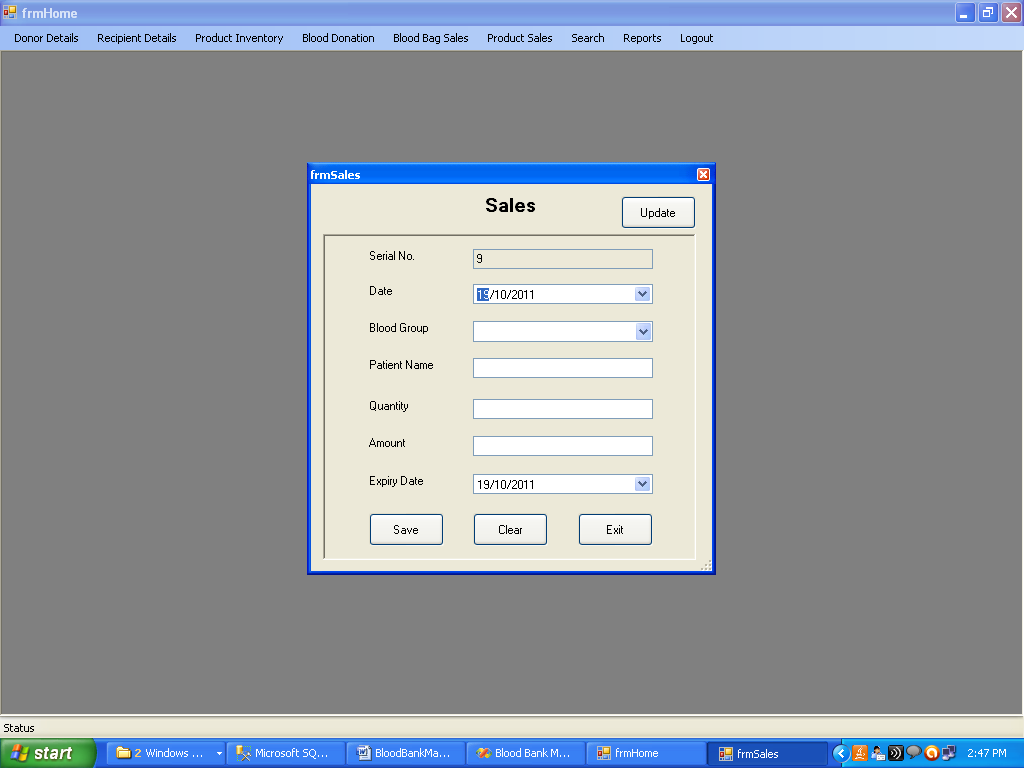
****

1. **Product:**

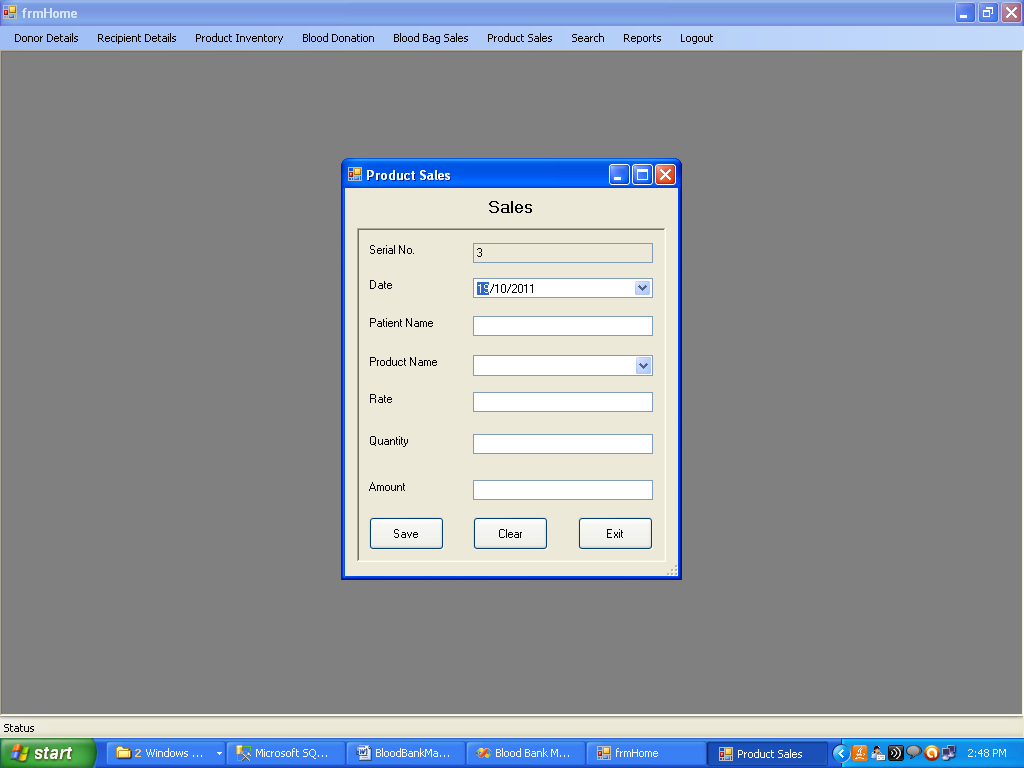
****

****

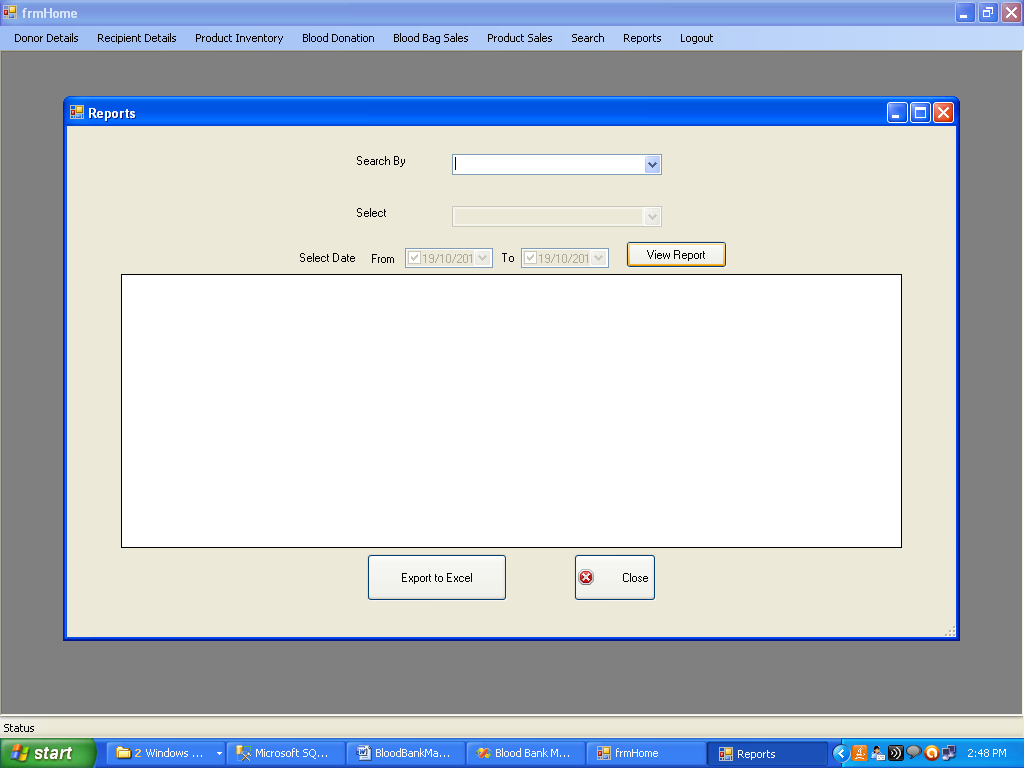
1. **Blood Bag Sales:**

****

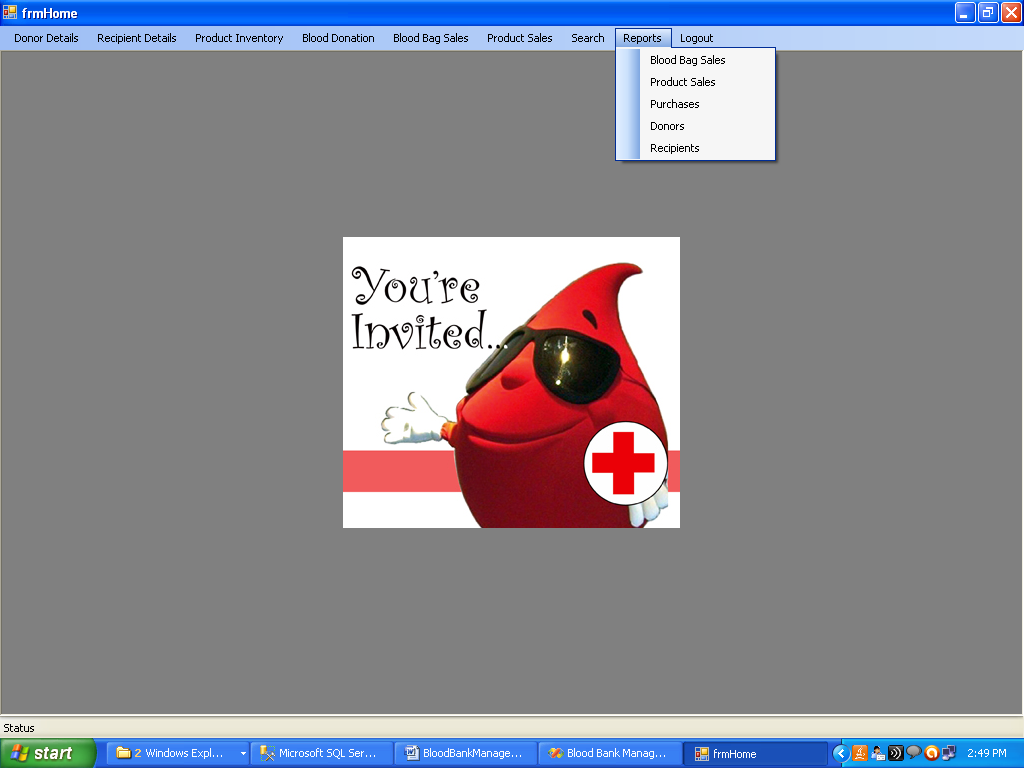
1. **Product Sales**

****

1. **Search**

****

1. **Reports**

****