**Low Level Design**

**Blood Bank Management System Revision Number: 1.3.2**

**Last date of revision: 26/01/2023 Team 1**

|  |  |  |
| --- | --- | --- |
|  | **CONTENTS** |  |
| **SL.NO** | **CHAPTER NAME** | **PAGE NO.** |
| 1. | Introduction | 01 |
| 2. | Block Diagram | 01 |
| 3. | System Architecture | 02 |
| 4. | Dataflow Diagram | 03 |
|  | * 1. Data Flow Diagram Notations   2. DFD’s for Blood Bank Management System   3. DFD for Admin Login   4. DFD for Account Creation | 04 |
| 05 |
| 06 |
| 07 |
| 5. | Tables Structure | 07 |
|  | * 1. Tables structure for admin login   2. Tables structure for user login | 07 |
| 07 |
| 6. | State Transition Diagram | 10 |
| 7. | ER Diagram | 10 |
|  | 7.1 Relationship Diagram Notations | 11 |
| 8. | Conclusion | 12 |

|  |  |  |
| --- | --- | --- |
|  | **TABLE CONTENTS** |  |
| **SL NO.** | **TABLE NAME** | **PAGE NO.** |
| 5.1 | Tables structure for admin login | 08 |
| 5.2 | Tables structure for user login | 08 |
| 1 | Donor details | 08 |
| 2 | Requester | 08 |
| 3 | Blood inventory | 08 |
| 4 | donate blood | 08 |
| 5 | Blood master | 09 |

**Low Level Design for Blood Bank Management System**

# Abstract

When the work related to all the estimation and planning is been completed the next step which is actual designing of the product is to be done. After successfully listing and gaining the resources required the process of constructing the elements with accomplished of required features is to be done. And the definition of the architecture, modules, components and different interfaces of those components in the system design phase.

# Introduction:

Systems design is the process of defining the architecture, modules, components and data for a system to satisfy specified requirements. Systems design could be seen as the application of systems theory to product development. There is some overlap with the disciplines of systems analysis, systems architecture and systems engineering. High level design identifying the system processes, functional components and their interfaces. Derived from system requirements, provides an overview of the project

* + Define the components that are needed.
  + Establish how components “communicate” with other components.
  + Determine how to modularize the project into discrete work package.
  + Identify critical interfaces that must be well defined.
  + Used to provide initial cost, scheduled & resource estimates. Usually little or few implementation details.
  + As system design is refined, and lower level subsystems are included, implementation

issues may need to be addressed.

# Block Diagram:

* + A block diagram is a diagram of a system in which represented by blocks connected by lines that show the relationships of blocks. They are heavily used in engineering in hardware design, electronic design, software design, and process flow diagrams.
  + Block diagrams are typically used for higher level, less detailed descriptions that are intended to clarify overall concepts without concern for the details of implementation.

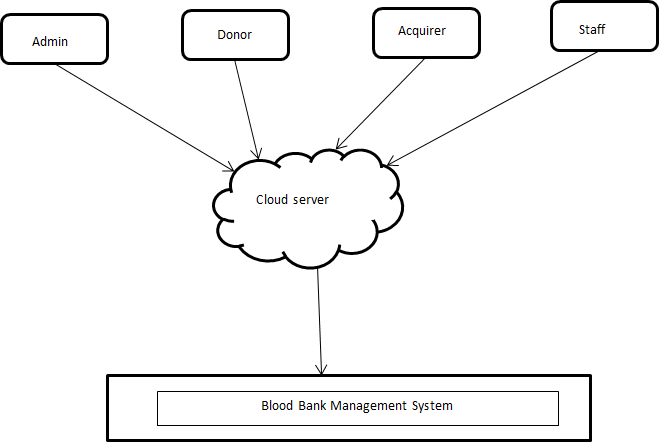


Fig 1: Block diagram of blood bank system

# System Architecture:

A system architecture or systems architecture is the conceptual model that defines the structure, behavior, and more views of a system. An architecture description is a formal description and representation of a system, organized in a way that supports reasoning about the structures and System architecture can comprise system components, the externally visible properties of those components, the relationships (e.g. the behavior) between them. It can provide a plan from which products can be procured, and systems developed, that will work together to implement the overall system.

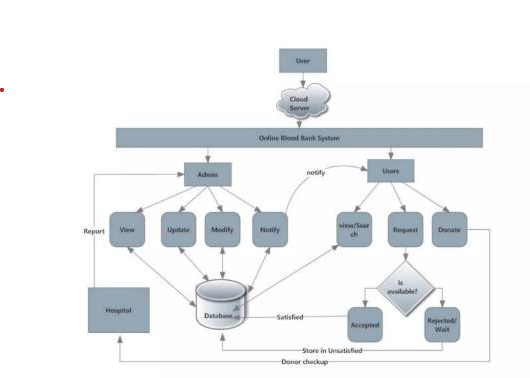


Fig 2: System architecture of blood bank system

# Dataflow Diagram

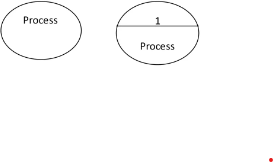
A data flow diagram is a graphical representation that depicts flow and the transforms that are applied ass data move from input to output. The basic form of a data flow diagram, also known as a data flow graph or a bubble chart. The data flow diagram may be used to represent a system or software at any level of abstraction. In fact, DFDs may be partitioned into levels that represent increasing information flow and functional detail. Therefore, the DFD provides a mechanism for functional modeling as well as flow modeling.

A context diagram is a top level (also known as Level 0) data flow diagram. It only contains one process node (process 0) that generalizes the function of the entire system in relationship to external entities. The first level DFD shows the main processes within the system.

## : Data Flow Diagram Notations

You can use two different types of notations on your data flow diagrams :

## 1: Process Notations

Process: A person transforms incoming data flow outgoing data flow.

## 2: Data Store Notations

Data Store: Data stores are repositories of data in the system. They are sometimes also referred to files.

## 3: Dataflow Notations

Dataflow: Dataflow are pipelines through which packets of information flow. Label the arrows with the name of the data moves through it.



**4: External Entity**: External entities are objects outside the systems, with which the system communicates. External entities are sources and destination of the system’s inputs and outputs.



# DFD’s for Blood Bank Management System

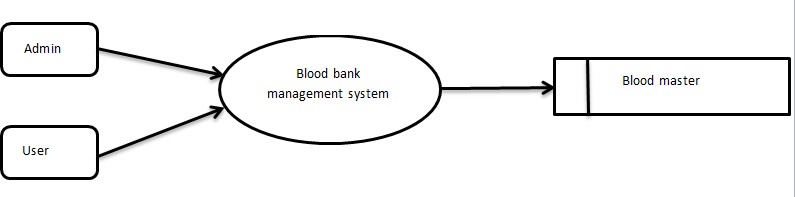


Fig 4.1: Level 0 DFD- Blood bank system

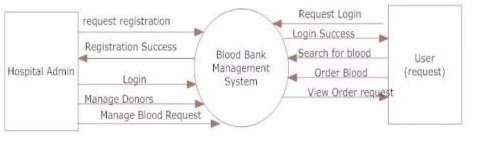


Fig 4.2: Level 1 DFD- Blood bank system

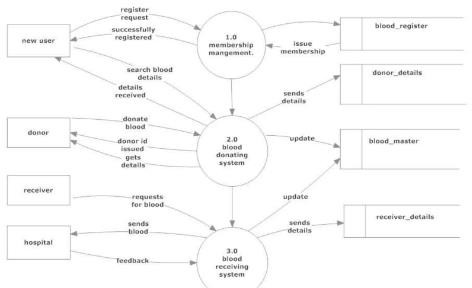


Fig 4.3: Level 2 DFD- Blood bank system

# DFD for Admin Login:

After entering to the home page of the website, admin can chose the admin login options where they are asked to enter username and password and if he/she is a valid admin login page will be displayed.

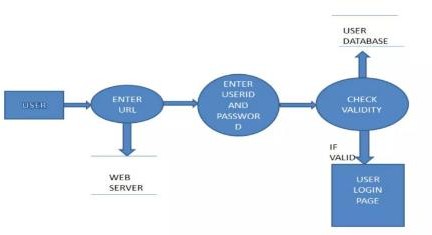
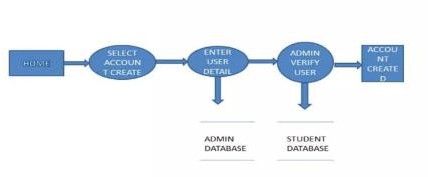


Fig 4.1.1: DFD for user login

# DFD for Account Creation:

After the home page there will be an option for CREATE an account where after entering the details if all the fields are filled then a request will be send to a librarian who will approve him/her as a registered member of the website



# Tables Structure:

Fig 4.2.1: DFD for account creation

In computer programming, a table is a data structure used to organize information, just as it is on paper. There are many different types of computer-related tables, which work in a number of different ways. The following are examples of the more common types.

1. In a data processing, a table (also called an array) is organized of fields. Tables may store relatively permanent data, or may be frequently updated.
2. In a relational database, a table (sometimes called a file) organizes the information about a single topic into rows and columns.
3. A decision table (often called a truth table), which can be computer-based or simply drawn upon paper, contains a list of decisions should be listed, and the action to take in each situation should be specified.
4. An HTML table is used to organize Web page elements spatially or to create a structure for data that is best displayed in tabular form, such as lists or specifications.

## Tables structure for admin login

|  |  |  |
| --- | --- | --- |
| **Name** | **Type** | **Allow Null** |
| Hospital\_id | VARCHARz | No |
| Password | VARCHAR | No |

* 1. **Tables structure for user login**

|  |  |  |
| --- | --- | --- |
| **Name** | **Type** | **Allow Null** |
| User\_id | VARCHAR | No |
| Password | VARCHAR | No |

## Table 1: Donor details

|  |  |  |
| --- | --- | --- |
| **Name** | **Type** | **Null** |
| Id | Int(10) | No |

|  |  |  |
| --- | --- | --- |
| Name | VARCHAR | No |
| Location | VARCHAR | No |
| Contact No. | Int(10) | No |
| Age | Int(2) | No |
| Location | VARCHAR | Yes |

**Table 2: Requester**

|  |  |  |
| --- | --- | --- |
| **Name** | **Type** | **Allow Null** |
| Id | Int(10) | No |
| Name | VARCHAR | No |
| Age | Int(2) | Yes |
| Location | VARCHAR | No |
| Blood type | VARCHAR | No |

## Table 3: Blood inventory

|  |  |  |
| --- | --- | --- |
| **Name** | **Type** | **Allow Null** |
| Id | Int(10) | No |
| City | Int(10) | No |
| Location | VARCHAR | Yes |
| Blood type | VARCHAR | No |

**Table 4: donate blood**

|  |  |  |
| --- | --- | --- |
| **Name** | **Type** | **Allow null** |
| Id | Int(10) | No |
| Name | VARCHAR | No |
| Location | VARCHAR | No |
| Blood type | VARCHAR | No |
| Contact | Int(10) | No |
| Sugar | VARCHAR | No |
| Address | VARCHAR | No |

**Table 5: Blood master**

|  |  |  |
| --- | --- | --- |
| **Name** | **Type** | **Allow null** |
| Id | Int(10) | No |
| Group | VARCHAR | No |

# State Transition Diagram:

The state transition diagram represents the behavior of a system by depicting its states and the events that cause the system to change state. In addition, the STD indicates what actions (e.g., process activation) are taken as a consequence of a particular event. A state transition diagram indicates how the system moves from state to state. State transition diagrams have been used right from the beginning in object-oriented modeling. The basic idea is to define a machine that has a number of states (hence the term finite state machine). The machine receives events from the outside world, and each event can cause the machine to transition from one state to another.

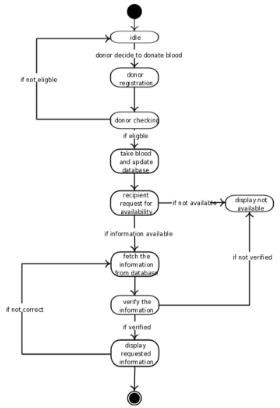


Fig 1.6: State transition diagram

## ER Diagram:

The entity / relationship diagram enables a software engineering to fully the data objects that are input and output from a system, the attributes that define the properties of these objects, and their relationships. Like most elements of the analysis model, the ERD is constructed in an iterative manner.

## Relationship Diagram Notations

* + 1. **Entity**: an entity is an object or concept about which you want to store information.
    2. **Weak Entity**: A weak entity is an entity that must defined by a foreign key relationship with another entity as it cannot be uniquely identified by its own attributes alone.
    3. **Key Attribute:** a key attribute is the unique, distinguish by characteristics of the entity. For example, an employee’s social security number might be the employee’s key attribute.
    4. Multi Valued Attribute: Multi valued attribute can have more than one value. For example, an employee entity can have multiple skill values.
    5. Relationships: Relationships illustrate how two entities share information in the database structure.

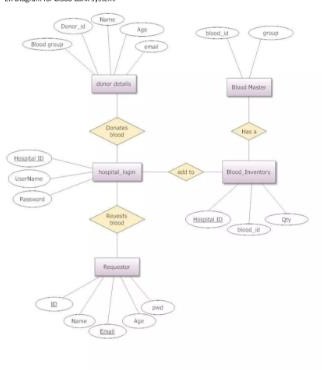


Fig 7.1: ER Diagram

## 8. Conclusion:

In this chapter we have define the components that are needed, and specified how components ‘communicate’ with other components. We have modularized the project into discrete work packages, identified critical interfaces that must be well defined. We have designed the block diagram, they are typically used for higher level, less detailed descriptions. After that we have designed system architecture, a system architecture is the conceptual model that defines the structure, behavior, and more views of a system. A data flow diagram is a graphical representation that depicts information flow and the transforms that are applied as data move from input to output. A table is a data structure used to organize information. A state transition diagram indicates how the system moves from state to state. E- R diagrams fully specify data objects, properties of the objects and the relationship.