Chapter 2: Analysis

2.1 Introduction

The term Analysis in software development means the process to inspect of data and documentation in detail to understand the system for providing basis to solve problem. In this phase, first step is to gather the necessary requirements for the system, which is the foundation for the proposed project. For the analysis phase, various diagrams are used such as use case diagram and class diagram to visualize the gathered requirements. This process of visualizing the requirements is known as system modelling.

2.2 Feasibility Study

Feasibility study is performed to determine whether the solution considered to accomplish the requirements is practical and workable in the software.

Technical feasibility

It deals with technical resource of the project such as type of technology is used to develop project, process or technique.

Economic feasibility

It deals with the budget require for the development of the project.

Operational feasibility

It is use to identify the importance of certain problem and how to solve. It takes action accordingly to track that problem.

Legal Feasibility

It helps to know the country constitution provide permission to create such website or not.  By the help of that software, it tracks personal detail or not.

Time

When client provides software to build, the time is firstly fixed. It is our major concern to finish the development of the software in time. If due to some difficulties, project will not be able to finish in time, the developer team has to pay fine to client based on agreement. Therefore, the company major concern is to make proper plan and develop the software in time.

Social

Social factor is also matter to concern while creating software. Does it affect people directly/indirectly or not? While creating the project it also focuses that how does it affect a particular community, races, and castes.

Therefore, proper analysis is done when the developer is creating project and client must focus on that parts.

2.4 SRS (Software Requirement Specification):

Software Requirement Specification also known as requirements document is a detailed description of a software system to be developed with its functional and non-functional requirements. It is an agreement between customer and contractor. This may include the use cases of how the user is going to interact with software system. A good SRS defines how software system will interact with all internal modules, hardware, communication with other program and human interaction with wide range of real-life scenarios.

2.4.1 Functional requirements:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ID | Title | Description | Rational | Dependency |
| FR01 | Registration | If new to the system the user can create a new account with this function | To create a new personal account. | N/A |
| FR02 | Login | The user can login to the system with their username and password. | To login to their account. | FR01 |
| FR03 | Logout | With this function a user can logout of their account after login. | To logout of their account. | FR02 |
| FR04 | Edit profile | After logging in the user can edit any personal information. And admin can edit all information . | To edit information on their account. | FR02 |
| FR05 | Search | Search donors | Search donor with their name. |  |
| FR06 | Delete | Users or admin can delete any information or even their own account in the system with this function. | To delete information and account in the system. | FR02 |
| FR07 | Request Blood | This function allows users to request the blood needed. | Request for the blood needed |  |
| FR08 | View Requests | This function allows donor to view the list of requested blood. | View list from the database. | FR07 |
| FR09 | View Donor Details | This function allows users to view list of donors with contact and blood group. | View list from the database. |  |

2.4.2 Non-functional requirements:

a) **Performance**: here the system must have a good response time, throughput, utilization etc. These are the things that determine the performance of the system, in order to run the software smoothly a good performance of the system is very crucial.

b) **Usability**: the software cannot be fully useable if It isn’t user friendly. This means the software must make users feel comfortable and be simple and easy to use.

c) **Reliability**: users have to trust the system, even after using it for a long time. In other words, the users must be able to rely on the software for a long time.

d) **Security**: the software must be able to provide high security along with providing the services, as it is suppose to the users. Example a secure system can be gained by the authorization and authentication during login.

e) **Supportability**: the software/ website should support in all kind of devices or screen size of the users. I.e. the website must be responsive or supportable in all kind of screen resolution.

f) **Availability**: the system should be available whenever the user wants to user it. The system must be able to run in every platform as well.

2.4.3 MOSCOW prioritization:

I have down the **MoSCoW prioritization** for all the functional requirements that I have listed above. The definition of MoSCoW gives more insight to the importance and priority of the requirements. The full form and description of MoSCoW:

**Must Have**: must have means that the function is guaranteed to be in the software.

**Should Have**: this means that the function/ requirement is important but is not vital.

**Could Have**: this means that the requirement is decided to be in the system but is not much important even if it’s not delivered.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ID | Requirement | MOSCOW | Rational | Remarks |
| M01 | Registration | Must have | To create a new personal account. |  |
| M02 | Login | Must have | To login to their account. |  |
| M03 | Logout | Must have | To logout of their account. |  |
| M04 | User role check | Could have | To check their role in the organization. |  |
| M05 | Edit profile | Could have | To edit information on their account. |  |
| M06 | Search | Must have | Can search donors. |  |
| M07 | Delete | Could have | To delete information and account in the system. |  |
| M08 | Request Blood | Must Have | Request Blood needed. |  |
| M09 | View Requests | Must Have | To view the requested blood by donor. |  |
| M10 | View Donor Details | Must Have | To view the list of donors with contact information. |  |

2.4.4 Hardware specification:

Employee management system or EMS should be able to work on a computer with the following minimum hardware specifications:

OS: Windows XP/Vista/7/8 and Linux

CPU: Pentium III (700MHz) and above

Memory: 128 MB and above

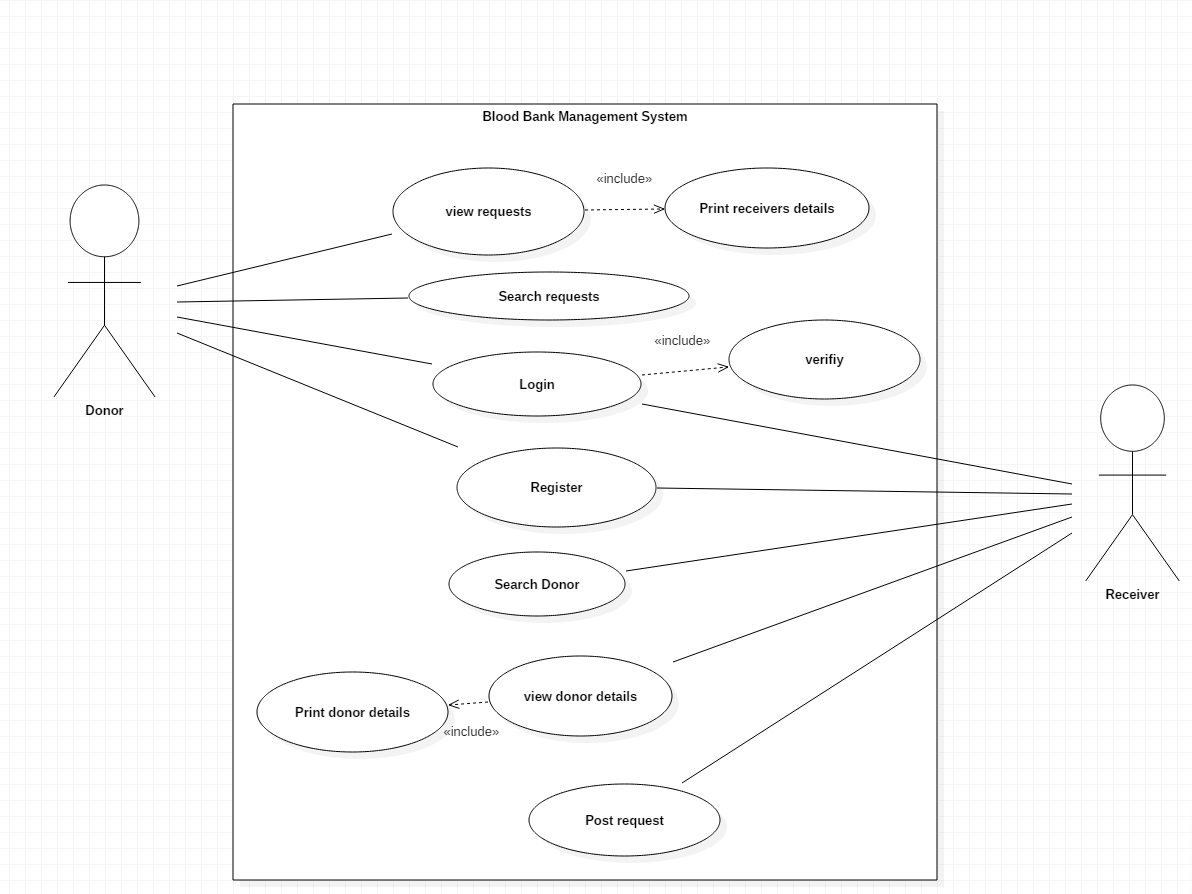
Capacity: 4GB of hard drive

Others: Network interface card, mouse, keyboard, and monitor

2.4.5 Software specification:

Since the Blood Bank management system is a web-based application, there must be internet connection.

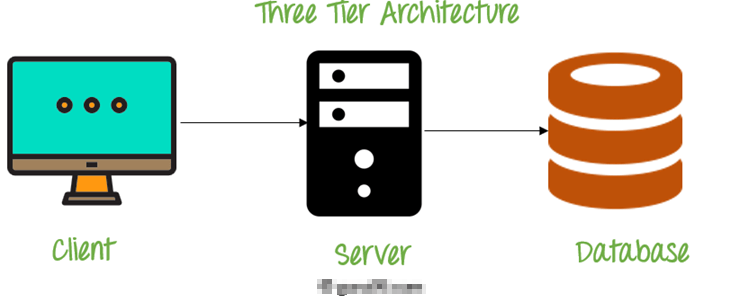
2.5 Use Case Diagram



2.6 System Architecture

System Architecture

This system used the three tier-architecture. 3-tier application is described as the software architecture that contains the three major logical computing. This architecture is very flexible and provides many benefits in the developmental environments because of the independent relationship between all three tiers. For example, the user interface can be changed without affecting the functional and data access logic.



**Client** **Tier**: The presentation layer is the front-end layer in the system architecture. This contains the User Interface (UI). This is the graphical view that can be accessed with web browser or web-based application, which displays the contents, that interacts with the user.

This tier is mostly written in web technology like HTML, JavaScript, CSS, or web frameworks.

**Server Tier**: This layer contains all the functional and business logic that helps to run the system. This tier is developed using core programming languages such as Java, .Net, C++, and many more.

**Data Tier**: This tier comprises of database. This tier is responsible for managing all the data and make them accessible whenever required by the system. MySQL, Oracle and many other databases are some of the example of this tier. (Anonymous, 2019)

Design Pattern

Design pattern is widely used by the software developer as a way of solving the problem that occurred during the development phase. There are many design patterns and they are categorized according to their behavior.

Bike Showroom Management System uses MVC pattern. Since the system is web-based application and uses 3-tier architecture, this pattern will help a lot in characterizing between the components used in the system. MVC stands for **M**odel **V**iew **C**ontroller. These three components communicate with each other to give fully functional application. This pattern is very loosely coupled and reliable to use.



**Model:**  It contains pure application data that does not concern UI or presentation. It has the main business logic.

**View:** View represents the graphical components of the system which user sees.

**Controller:**  This act both as model and as view. It controls the flow of data into the model and updates the view as per the logic. View is updated when model changes and model is updated with user view manipulation.