

A Project Report
On
“ONLINE UNUSED MEDICINE DONATION”

Submitted in partial fulfillment of the requirement of

University of Mumbai

For the Degree of

Bachelor of Computer Science

Submitted by

Pranali Raul

Under the Guidance of

Prof. Saba Shaikh

Final Year Computer Science



**Department of BSc. Computer Science Ramniranjan
Jhunjhunwala College of Arts, Science & Commerce**

Affiliated to University of Mumbai.

Mumbai

(2020-2021)



**RAMNIRANJAN JHUNJHUNWALA COLLEGE
GHATKOPAR (W), MUMBAI-400 086**

CERTIFICATE

This is to certify that **Pranali Raul** has successfully completed the project titled

“Online Unused Medicine Donation”

under our guidance towards the partial fulfillment of degree of Bachelors of
Science (Computer Science – SEM V) submitted to Ramniranjan
Jhunjhunwala College, Ghatkopar (W) during the academic year 2020 – 2021
as specified by

UNIVERSITY OF MUMBAI

**Prof. (Ms.) Saba Shaikh
Project Guide
Dept. of Computer Science**

**Prof. Anita Gaikwad
In-Charge
Dept. of Computer Science**

	INDEX	
1.	Acknowledgement	4
2.	Declaration	5
3.	Feasibility Study	6
	• 3.1 Technical Feasibility	6
	• 3.2 Economic Feasibility	6
	• 3.3 Operational Feasibility	6
4.	Scope of the System	7
5.	Existing System and it's Disadvantages	8
6.	Proposed System and Its Advantages	9
7.	Technical Requirements	10
8.	Software Development Model	11-13
9.	System Analysis	14-29
	• 9.1 E-R Diagram	16
	• 9.2 Class Diagram	17
	• 9.3 Object Diagram	19
	• 9.4 Activity Diagram	22
	• 9.5 Sequences Diagram	25
	• 9.6 Use Case Diagram	28
10.	System Design	30-36
	• 10.1 Component Design	30
	• 10.2 Deployment Design	32
	• 10.3 Table Design	34

Acknowledgement

I have great pleasure for representing this project report entitled **“Online Unused Medicine Donation”** and I grab this opportunity to convey my immense regards towards all the distinguished people who have their valuable contribution in the hour of need.

I take this opportunity to thank **“Mr. Himanshu Dawda”**, our Principal of **Ramniranjan Jhunjhunwala College, Ghatkopar (W)** for giving me an opportunity to study in the institute and the most needed guidance throughout the duration of the course.

I would also like to thank **“Mrs. Anita Gaikwad”**, Head of the Department for their timely and prestigious guidance and necessary support

I also like to extend my gratitude to **“Prof. Saba Shaikh”**, for their timely and prestigious guidance and necessary support during each phase of the project.

I also owe to my fellow friends who have been constant source of help to solve the problems and also helped me during the project development phase.

Thanking You
Pranali Raul

Declaration

I declare that this written submission represents my own ideas in my own words and where other's ideas or words have been included, I have adequately cited and referenced the original sources. I also declare that I have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/fact/data/source in my submission. I understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

Pranali Raul

Feasibility Study

Feasibility study is to check the viability of the project under consideration. Theoretically various types of feasibilities are conducted, but I have conducted three types of feasibilities explained as under.

ECONOMIC FEASIBILITY:

1. Development Cost
 - Equipment required for developing the software are easily available.
 - Equipment maintenance is also minimum.
 - Once the required hardware and software requirements get fulfilled there is no need for the users of our system to spend for any additional overhead.
2. Benefits which cannot be measured:
 - Increased customer Loyalty.
 - Increased customer satisfaction.

TECHNICAL FEASIBILITY:

At first it is necessary to check that the proposed system is technically feasible or not and to determine the technology and skill necessary to carry out the project. If they are not available then find out the solution to obtain them.

OPERATIONAL FEASIBILITY:

Proposed system is beneficial only if it can be turned into system that will meet the need of the client's operating requirements. The proposed system is operationally feasible due to the following reasons:

- It is easy to use.
- The website will support all browser.
- It will avoid confusion and resistance by catching the user's attention, as it is presentable.

Scope of the System

This project aims to donate medicines which are unused. The unused medicine can be donated for further utilization by a needy person. This application helps the user to donate unused medicines to NGO. In this system there are five entity namely, Admin, NGO, Donor, Recipient, Volunteer/Executive. Admin will login and manage members by deleting and blocking the users providing improper or expired medicines. Admin also has authority to approve the appointment requested by NGO. Admin has monthly report of medicines which has been donated. NGO can register and login using credentials. NGO manages the stock which helps to maintain the record of the available medicine. In case of security NGO can also change their password. Donors can register and login using credentials. They can donate medicine by providing medicines detail and raising request, further it will be approved by NGO they will schedule the donating date. Members can also check their previous data of medicine transactions. Recipient can register and login and then further request for the required medicine. The NGO will receive a request and then they will assign the Executive/Volunteer who will take the medicine from the donor and then give it to receiver.

Existing System and its Disadvantages:

- The consumer can get expired or counterfeit product.
- The current existing website may not allow the recipient to check the expiry date or the donor can give false information.

Proposed System and its Advantages:

- This website will ensure that the consumer will get the proper requested medicine.
- The consumer can also check the expiry date of the medicine.
- It helps poor people for medication. It will help to maintain records of medicines
- Many needy people will get cure.

Technical Requirements

Programming languages:

- **Front end** : Html , CSS , JS
- **Back end** : Java , SQL

Hardware Requirement:

i3 Processor Based Computer or higher
Memory: 1 GB
Hard Drive: 50 GB
Monitor
Internet Connection

Software Requirement:

Windows 7 or higher
Net Beans
SQL Server
Google Chrome Browser

Software Development Model

V-Model

- The V-model is an SDLC model where execution of processes happens in a sequential manner in a V-shape. It is also known as Verification and Validation model.
- The V-Model is an extension of the waterfall model and is based on the association of a testing phase for each corresponding development stage. This means that for every single phase in the development cycle, there is a directly associated testing phase. This is a highly-disciplined model and the next phase starts only after completion of the previous phase.

Phases of V-Model:

1. Requirement Analysis

This is the first phase in the development cycle where the product requirements are understood from the customer's perspective. This phase involves detailed communication with the customer to understand his expectations and exact requirement. This is a very important activity and needs to be managed well, as most of the customers are not sure about what exactly they need. The acceptance test design planning is done at this stage as the business requirements can be used as an input for acceptance testing.

2. System Design

Once you have clear and detailed product requirements, it is time to design the complete system. The system design will have the understanding and detailing the complete hardware and communication setup for the product under development. The system test plan is developed based on the system design. Doing this at an earlier stage leaves more time for the actual test execution later.

3. Architectural Design

Architectural specifications are understood and designed in this phase. Usually, more than one technical approach is proposed and based on the technical and financial feasibility the final decision is taken. The system design is broken down further into modules taking up different functionality. This is also referred to as High-Level Design (HLD).

The data transfer and communication between the internal modules and with the outside world (other systems) is clearly understood and defined in this stage. With this information,

integration tests can be designed and documented during this stage.

4. Module Design

In this phase, the detailed internal design for all the system modules is specified, referred to as Low-Level Design (LLD). It is important that the design is compatible with the other modules in the system architecture and the other external systems. The unit tests are an essential part of any development process and help eliminate the maximum faults and errors at a very early stage. These unit tests can be designed at this stage based on the internal module designs.

5. Coding Phase

The actual coding of the system modules designed in the design phase is taken up in the Coding phase. The best suitable programming language is decided based on the system and architectural requirements. The coding is performed based on the coding guidelines and standards. The code goes through numerous code reviews and is optimized for best performance before the final build is checked into the repository.

6. Validation Phases

The different Validation Phases in a V-Model are explained in detail below.

i. Unit Testing

Unit tests designed in the module design the phase is executed on the code during this validation phase. Unit testing is the testing at the code level and helps eliminate bugs at an early stage, though all defects cannot be uncovered by unit testing.

ii. Integration Testing

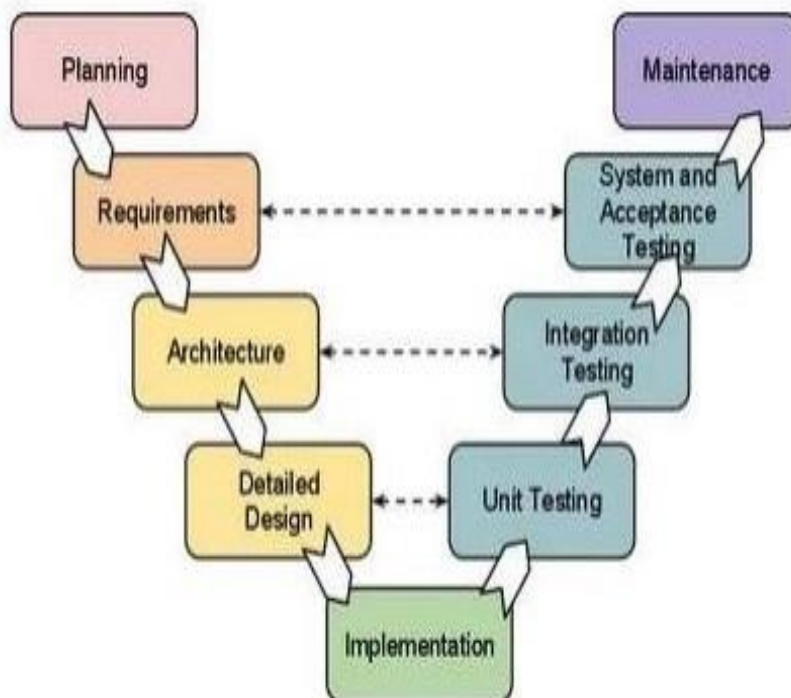
Integration testing is associated with the architectural design phase. Integration tests are performed to test the coexistence and communication of the internal modules within the system.

iii. System Testing

System testing is directly associated with the system design phase. System tests check the entire system functionality and the communication of the system under development with external systems. Most of the software and hardware compatibility issues can be uncovered during this system test execution.

iv. Acceptance Testing

Acceptance testing is associated with the business requirement analysis phase and involves testing the product in the user environment. Acceptance tests uncover the compatibility issues with the other systems available in the user environment. It also discovers the non-functional issues such as load and performance defects in the actual user environment.



GANTT CHART

- A Gantt chart is a useful graphical tool which shows activities or tasks performed against time.
- It is also known as visual presentation of a project where the activities are broken down and displayed on a chart which makes it is easy to understand and interpret.
- On the chart, tasks are shown on the vertical axis while the scheduled time-spend is laid out on the horizontal axis. Each task is represented by a bar that shows the time required for the project. The bar then represents or shows percentage of tasks that have been completed. It also shows dependencies, which simply means the inter-linkages between various activities in the project.
- Understanding the inter-linkages between activities is very important to monitor and Gantt charts help the project manager to do just that. It conveys the information about the completion of other activities in the project. This information is important because of the inter-linkages between various activities and if one activity gets delayed it will have an impact on others.
- Gantt chart is useful tool in planning and scheduling the projects. It keeps the management updated as to when the project will get completed. It also keeps the management informed about any additional resources that are required, and manage dependencies between tasks.
- A Gantt chart is made up of several different elements. Below given are 8 key components to know how to read a Gantt chart:
 - Task list: Runs vertically down the left of the Gantt chart to describe project work and may be organized into groups and sub-groups.
 - Timeline: Runs horizontally across the top of the Gantt chart and shows months, weeks, days, and years.
 - Dateline: A vertical line that highlights the current date on the Gantt chart.
 - Bars: Horizontal markers on the right side of the Gantt chart that represent tasks and show progress, duration, and start and end dates.
 - Milestones: Yellow diamonds that call out major events, dates, decisions, and deliverables.
 - Dependencies: Light gray lines that connect tasks that need to happen in a certain order.
 - Progress: Shows how far along work is and may be indicated by %Complete and/or bar showing.
 - Resource assigned: Indicates the person or team responsible for completing a task.

Title	Start Date	End Date	Duration
REQUIREMENT ANALYSIS			
Preliminary Investigation	04-05-2020	07-05-2020	4
Project Topic Discussion	05-05-2020	05-05-2020	1
Current System Description	06-05-2020	06-05-2020	1
Proposed System Description	06-05-2020	06-05-2020	1
Feasibility Study	07-05-2020	07-05-2020	1
SYSTEM ANALYSIS			
E-R Diagram	11-05-2020	11-05-2020	1
Class Diagram	12-05-2020	12-05-2020	1
Object Diagram	13-05-2020	13-05-2020	1
Activity Diagram	14-05-2020	14-05-2020	1
Sequence Diagram	14-05-2020	14-05-2020	1
Use Case Diagram	15-05-2020	15-05-2020	1
SYSTEM DESIGN			
Component Diagram	08-06-2020	08-06-2020	1

Deployment Diagram	09-06-2020	09-06-2020	1
Table Diagram	10-06-2020	10-06-2020	1
<p align="center">SYSTEM CODING In progress</p>			
<p align="center">SYSTEM TESTING In progress</p>			

REQUIREMENT ANALYSIS



SYSTEM ANALYSIS



SYSTEM DESIGN



System Analysis

E-R Diagram

- The ER or (Entity Relational Model) is a high-level conceptual data model diagram. Entity-Relation model is based on the notion of real-world entities and the relationship between them.
- An Entity Relationship (ER) Diagram is a type of flowchart that illustrates how “entities” such as people, objects or concepts relate to each other within a system.
- ER diagrams are related to data structure diagrams (DSDs), which focus on the relationships of elements within entities instead of relationships between entities themselves. ER modeling is something regarded as a complete approach to design a logical database schema.
- ER Diagrams are composed of entities, relationships and attributes. They also depict cardinality, which defines relationships in terms of numbers.
- Entity
- An entity is an object or component of data. An entity is represented as rectangle in an ER diagram.
- For example: Student and College and these two entities have many to one relationship as many student studies in a single college.
- Weak Entity
- An entity that cannot be uniquely identified by its own attributes and relies on the relationship with other entity is called weak entity. The weak entity is represented by a double rectangle.

Attribute:

An attribute describes the property of an entity. An attribute is represented as Oval in an ER diagram. There are four types of attributes:

Key attribute

Composite attribute

Multivalued attribute

Derived attribute

Relationship:

A relationship is represented by diamond shape in ER diagram, it shows the relationship among

entities. There are four types of relationships:








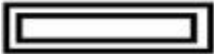

One to One

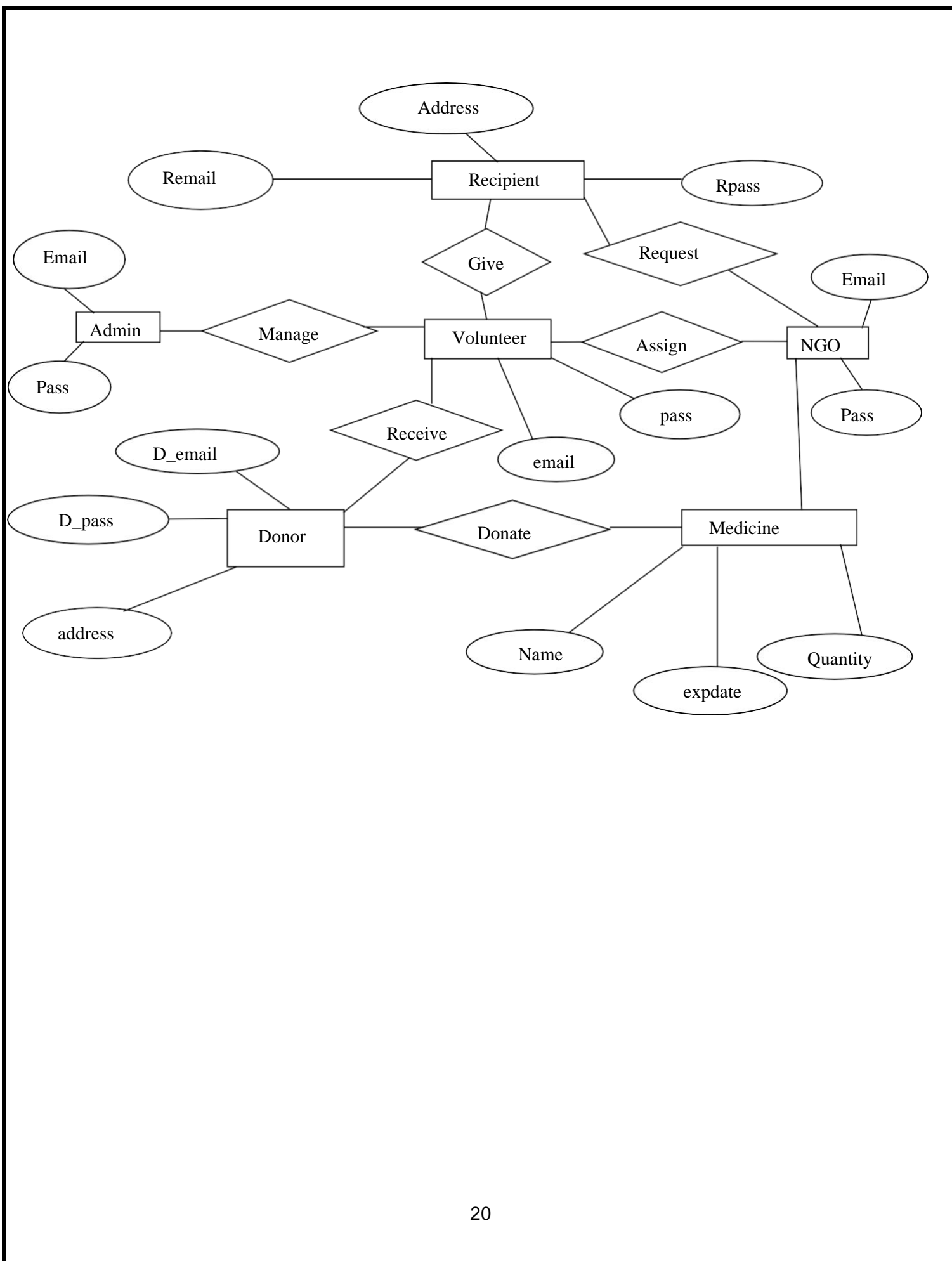
One to Many

Many to One

Many to Many

ER Diagram Symbols

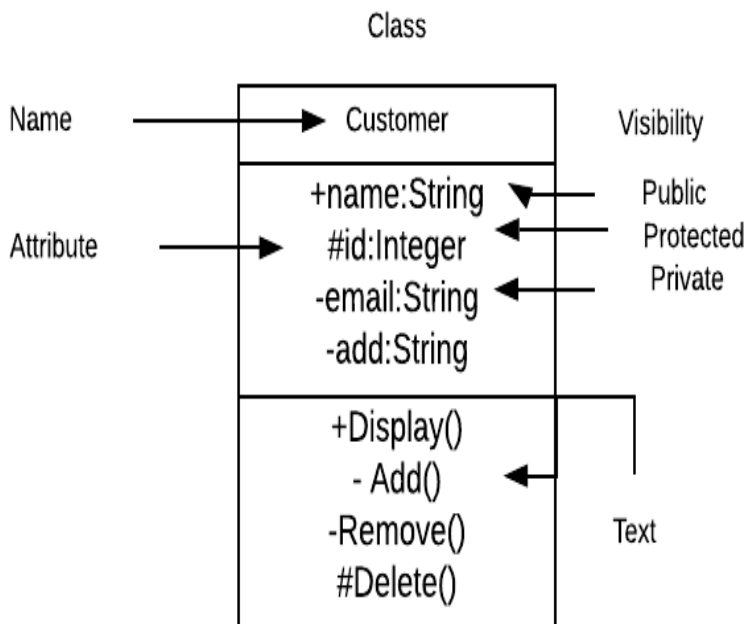
	Represents Entity
	Represents Attribute
	Represents Relationship
	Links Attribute(s) to entity set(s) or Entity set(s) to Relationship set(s)
	Represents Multivalued Attributes
	Represents Derived Attributes
	Represents Total Participation of Entity
	Represents Weak Entity
	Represents Weak Relationships

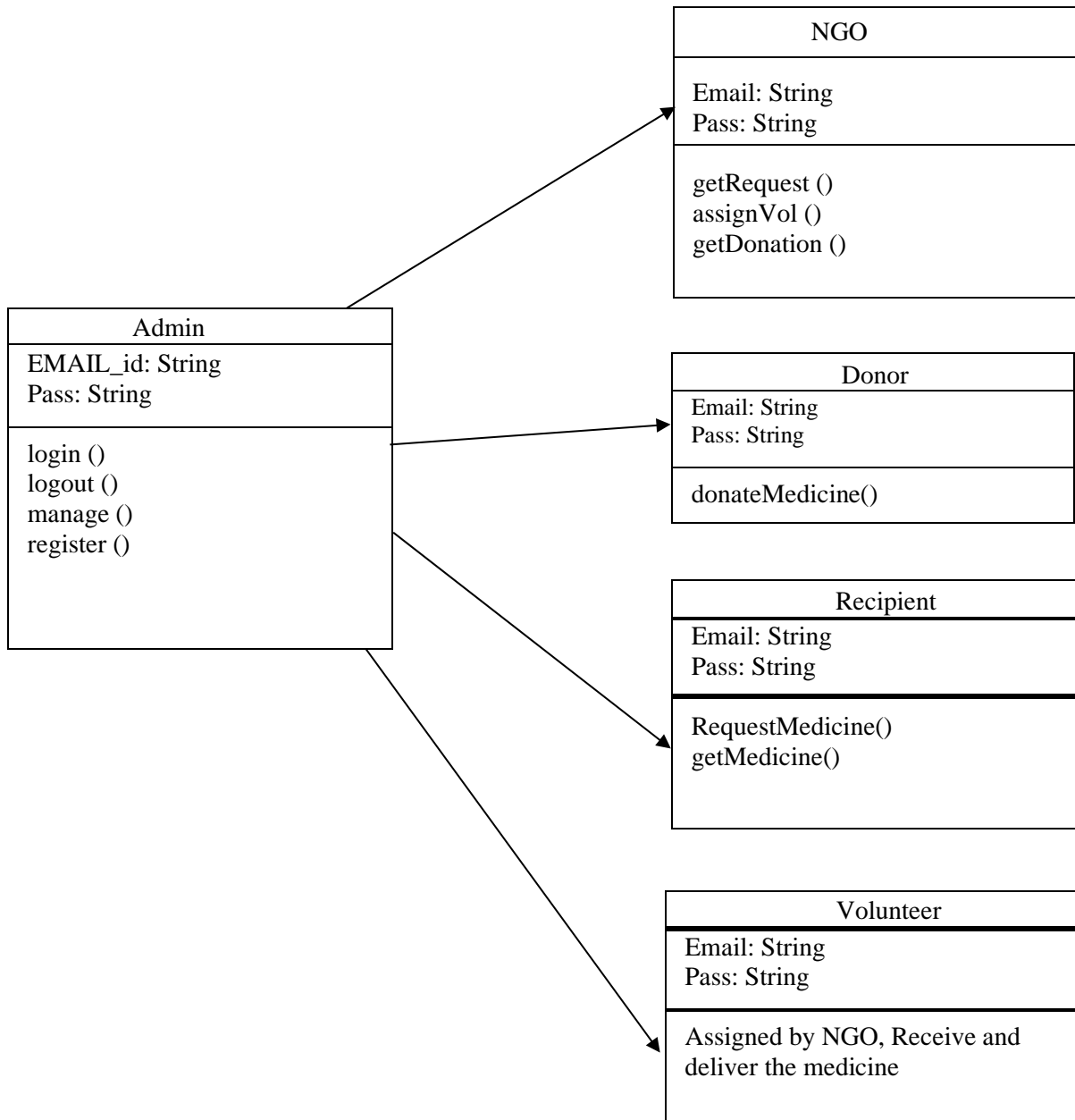


Class Diagram

- It is a model which is used to show the classes constituting a system and their interrelationship. It is based on UML. Only the important attributes and methods are shown in Class diagrams. In the initial period of analysis, the important attributes of the classes, which must be captured and the functionalities provided by the class may not be very clear. As the analysis progresses, the attributes and methods may be added. If more focus is on interrelationships of classes, then the attributes and methods may not be shown in the class diagram.
- The class diagram is used to identify and classify the objects which constitute a system. It also includes the important attributes of the objects which must be captured.

Notation:





Object Diagram

An object diagram is a graph of instances, including objects and data values. A static object diagram is an instance of a class diagram; it shows a snapshot of the detailed state of a system at a point in time.

Object diagrams and class diagrams are closely related and use almost identical notation. Both diagrams are meant to visualize static structure of a system. While class diagrams show classes, object diagrams display instances of classes (objects).

Object diagrams are more concrete than class diagrams. They are often used to provide examples or act as test cases for class diagrams. Only aspects of current interest in a model are typically shown on an object diagram.

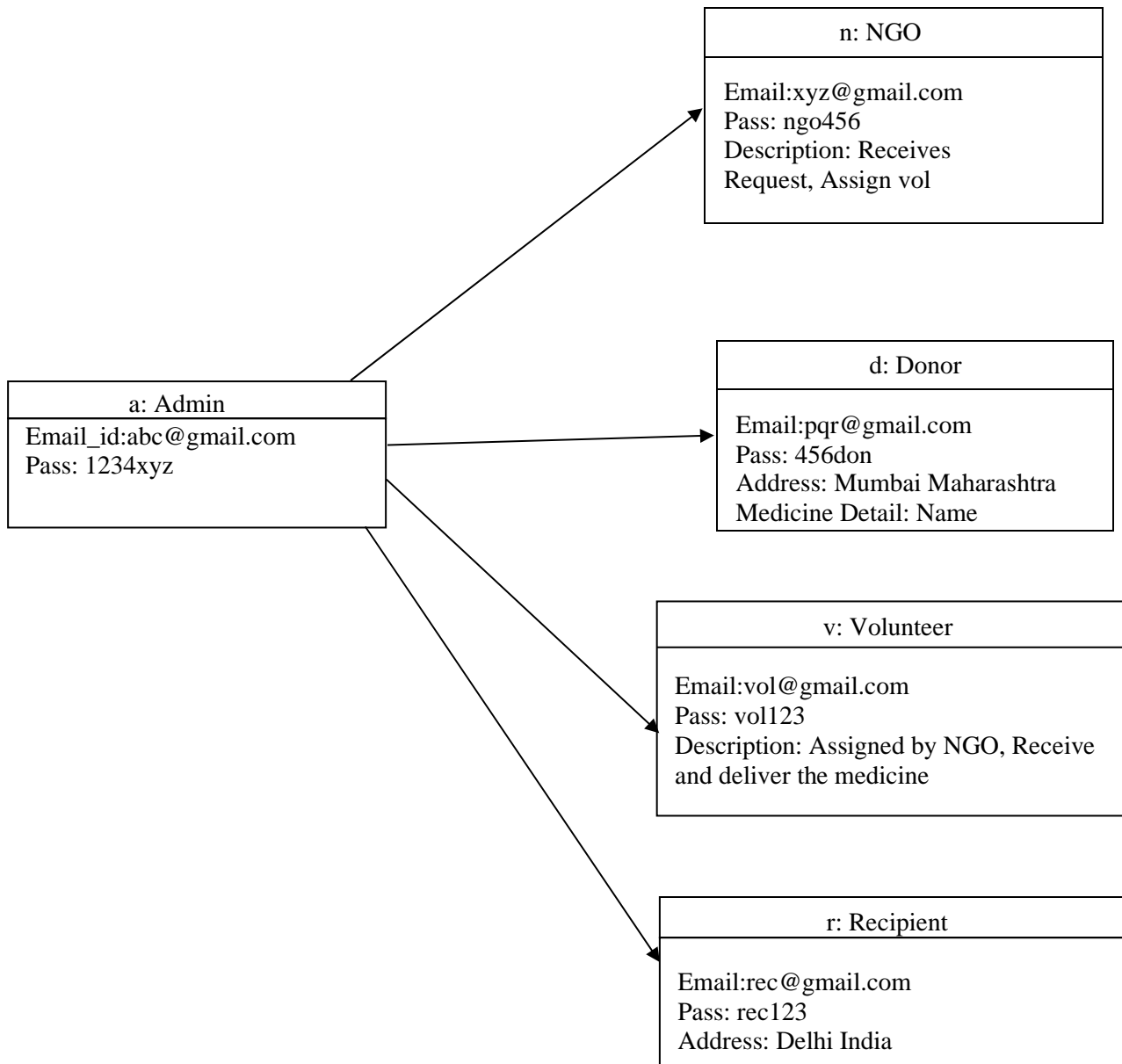
Object diagrams are derived from class diagrams so object diagrams are dependent upon class diagrams.

Purpose of Object Diagrams

- The purpose of a diagram should be understood clearly to implement it practically. The purposes of object diagrams are similar to class diagrams.
- The difference is that a class diagram represents an abstract model consisting of classes and their relationships. However, an object diagram represents an instance at a particular moment, which is concrete in nature.
- It means the object diagram is closer to the actual system behavior. The purpose is to capture the static view of a system at a particular moment.

The purpose of the object diagram can be summarized as –

- Forward and reverse engineering
- Object relationships of a system
- Static view of an interaction.
- Understand object behavior and their relationship from practical perspective.



Activity Diagram

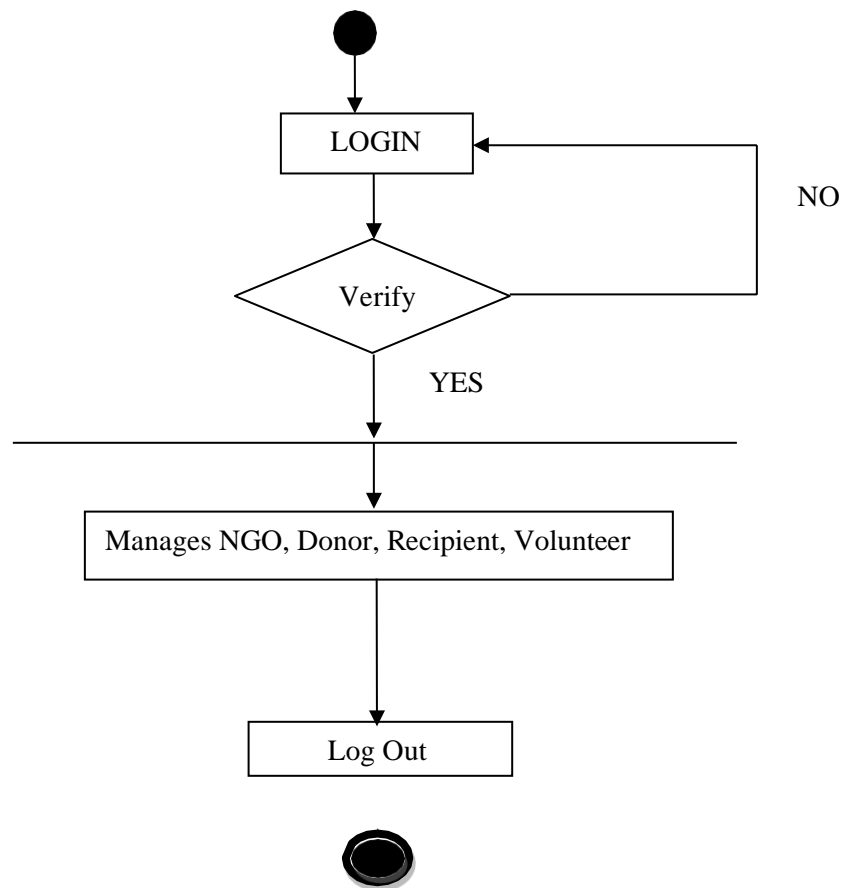
Activity diagrams are graphical representations of workflows of step wise activities and actions with support for choice, iteration and concurrency. In the Unified Modeling Language, activity diagrams can be used to describe the business and operational step-by- step workflows of components in a system. An activity diagram shows the overall flow of control. Activity diagrams are constructed from a limited repertoire of shapes, connected with arrows.

Activity diagrams are constructed from a limited repertoire of shapes, connected with arrows.

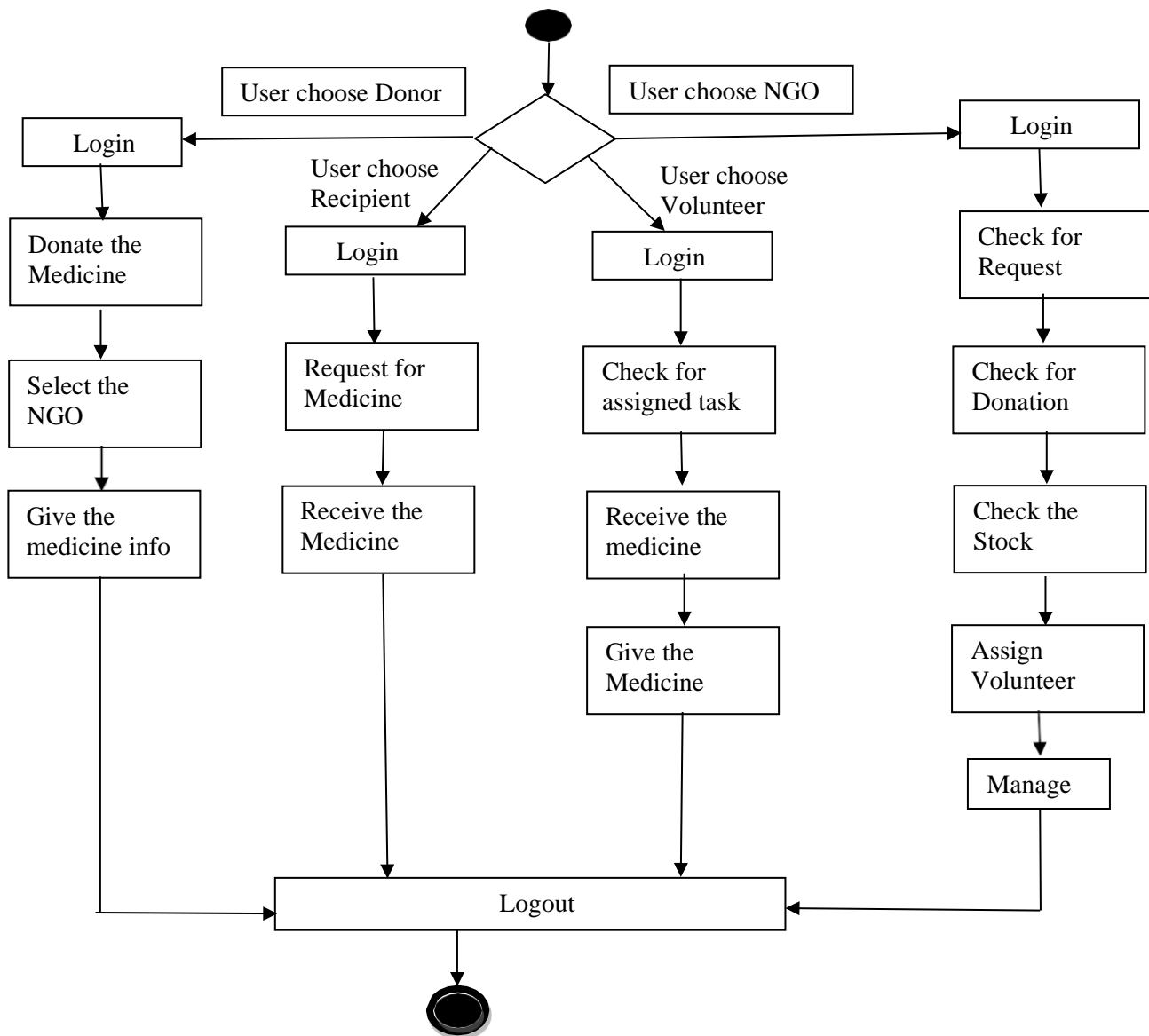
The most important shape types:

- Rounded rectangle represents activities.
- Diamonds represent decisions.
- Bars represent the start (split) or end (join) of concurrent activities.
- A black circle represents the start (initial state) of the workflow.
- An encircled black circle represents the end (final state).
- Arrows run from the start towards the end and represent the order in which activities happen.

Admin side Activity Diagram:



User side Activity Diagram:



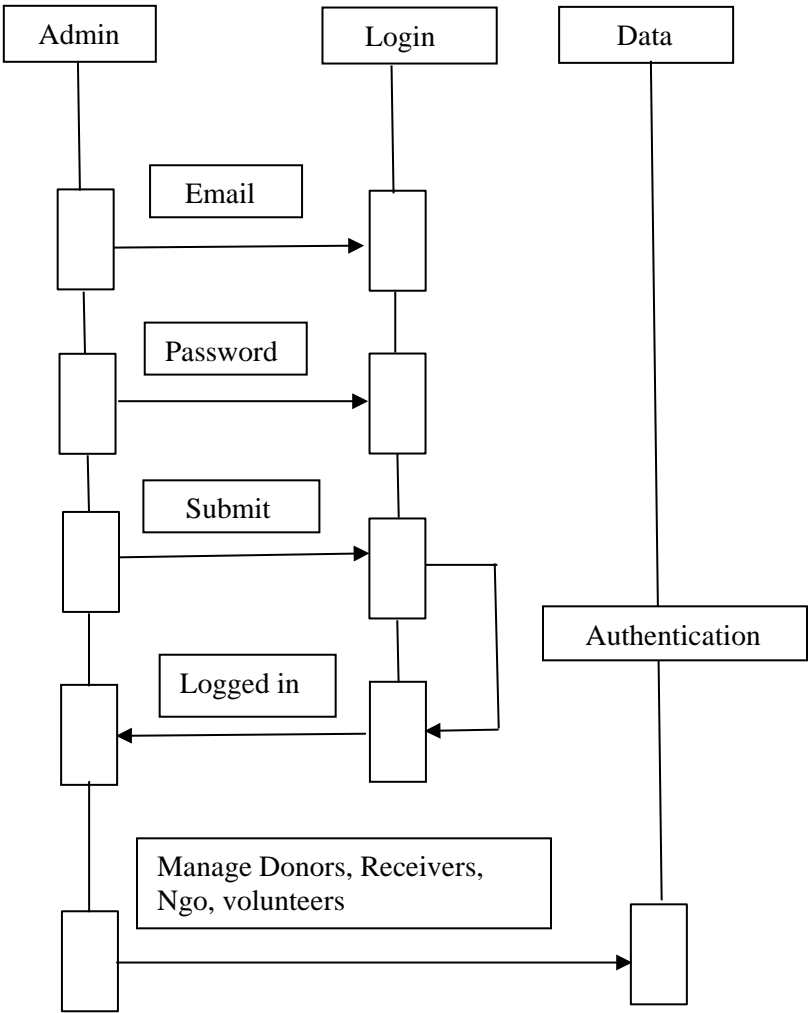
Sequence Diagram

A sequence diagram describes interactions among classes in terms of an “Exchange of message over time”. Sequence diagrams are used to depict the time sequence of message exchanged between objects. Message can correspond to operation on class or an event trigger. Notations of a Sequence Diagram include:

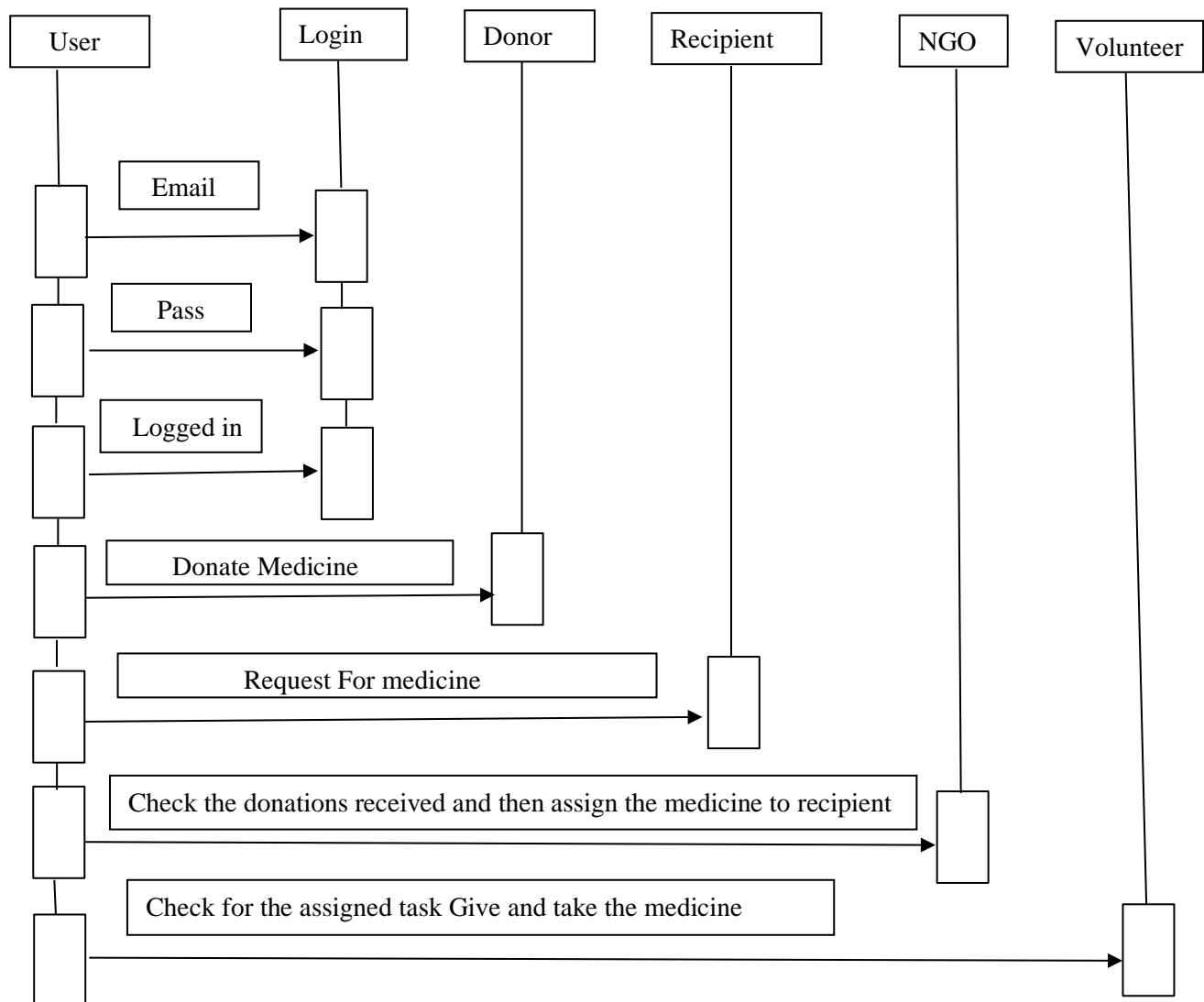
- Lifeline: It is a vertical dashed line that represents the “lifetime” of an object.
- Rows: They indicate flow of message between objects.

Activation: It is a thin rectangle showing period of time, during which an object is performing an action.

Admin side Sequence Diagram:

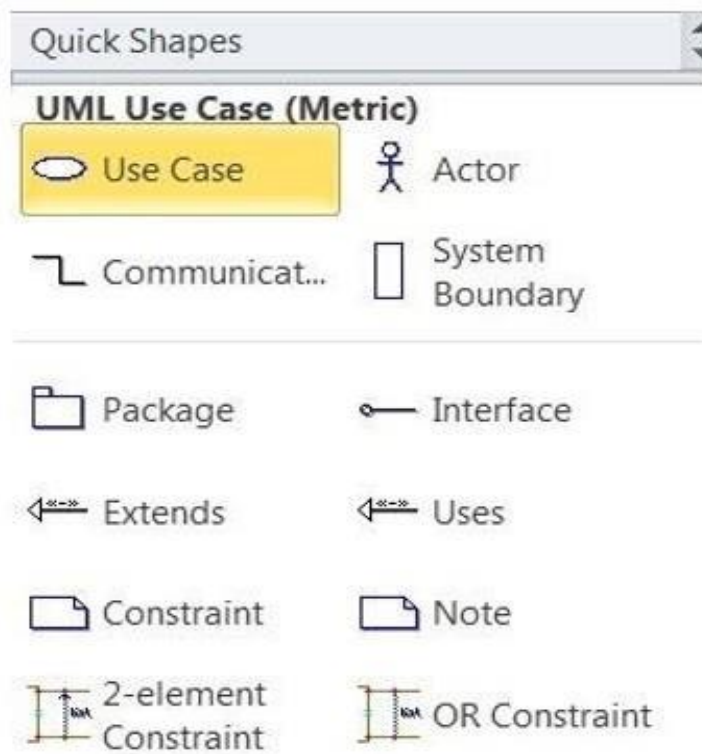


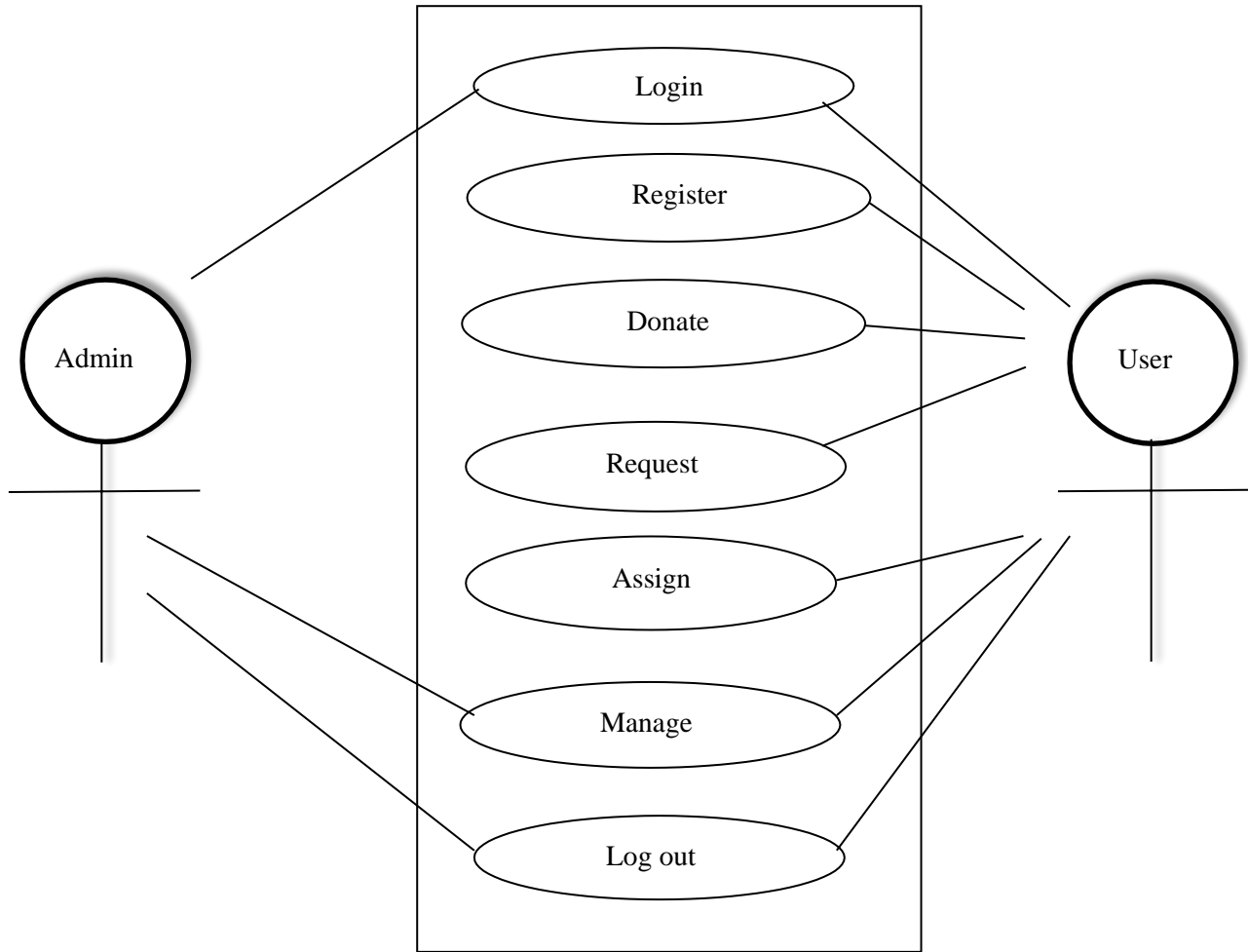
User side Sequence Diagram:



Use Case Diagram

A use case diagram in the unified modeling language is a type of behavioral diagram defined by and created from a Use-Case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases.





System Design

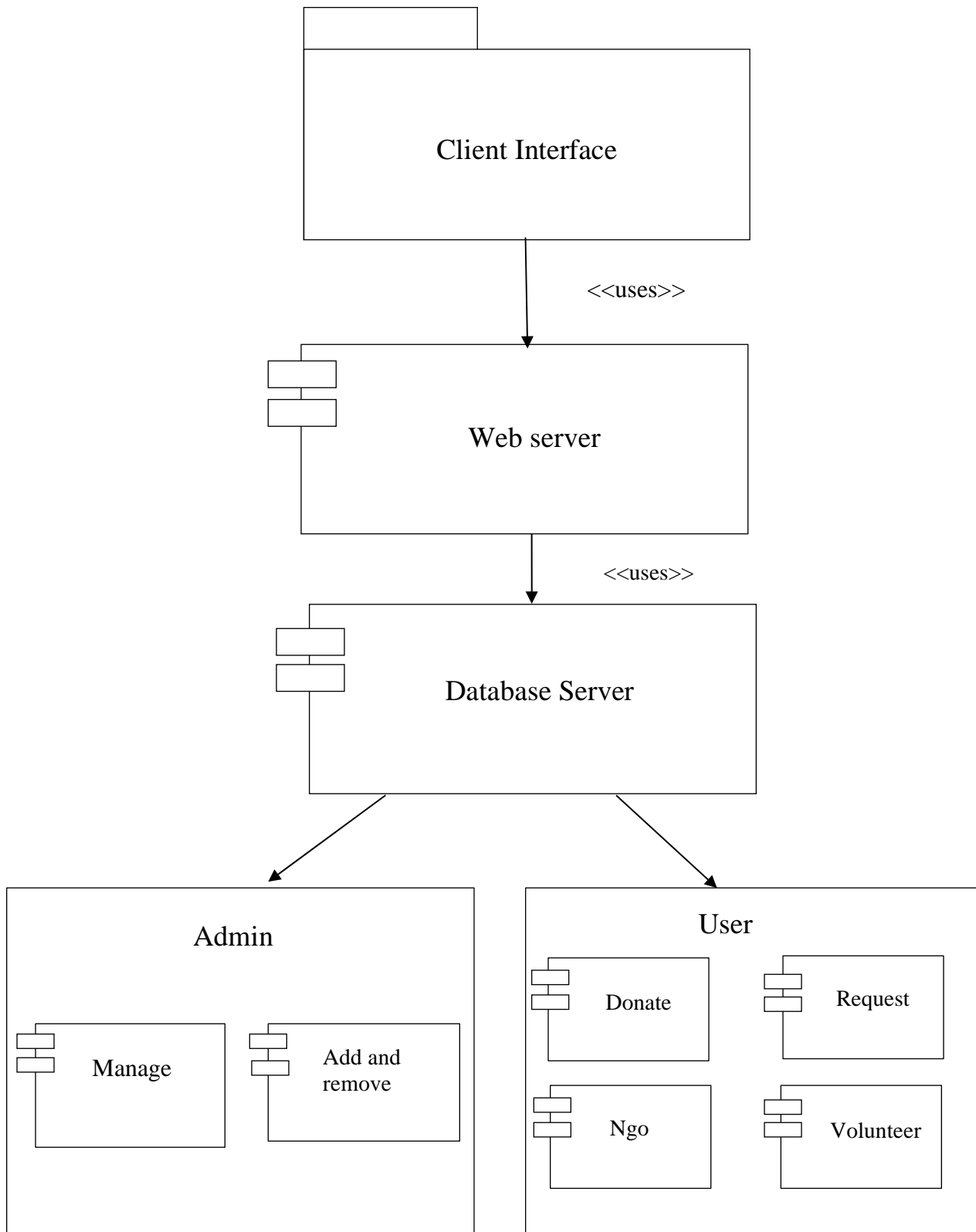
Component Diagram

Component Diagrams describe the organization of components, including source code, run-time (binary) code, and executable. Component Diagrams:

- Give the physical view of the system in terms of implementation aspect. This is important for reusability and performance purpose.
- Constitute the Components, their interfaces and realizations, and dependencies between components.

Component Diagrams are used:

- To depict organizations and dependencies among Component type.
- To show allocation of “Classes” and “objects” to components in the physical design of the system.
- To indicate the “physical layering” and “partitioning” of the system Architecture. A component typically encompasses:
 - Structure and behavior of a “Collaboration of classes” from the system design.
 - Interfaces that describe a group of operations implemented by components



Deployment Diagram

- Deployment diagram is a structure diagram which shows architecture of the system as deployment (distribution) of software artifacts to deployment targets. • Artifacts represent concrete elements in the physical world that are the result of a development process.
- Deployment Diagram is usually represented by a node which is either hardware device or some software execution environment. Nodes could be connected through communication paths to create networked systems of arbitrary complexity.

Note: That component was directly deployed to nodes in UML 1.x deployment diagrams. In UML 2.x artifacts are deployed to nodes, and artifacts could manifest (implement) components. Components are deployed to nodes indirectly through artifacts.

Deployment diagrams could describe architecture at specification level (also called type level) or at instance level (similar to class diagrams and object diagrams).

Specification level deployment diagram shows some overview of deployment of artifacts to deployment targets, without referencing specific instances of artifacts or nodes.

Instance level deployment diagram shows deployment of instances of artifacts to specific instances of deployment targets. It could be used for example to show differences in deployments to development, staging or production.

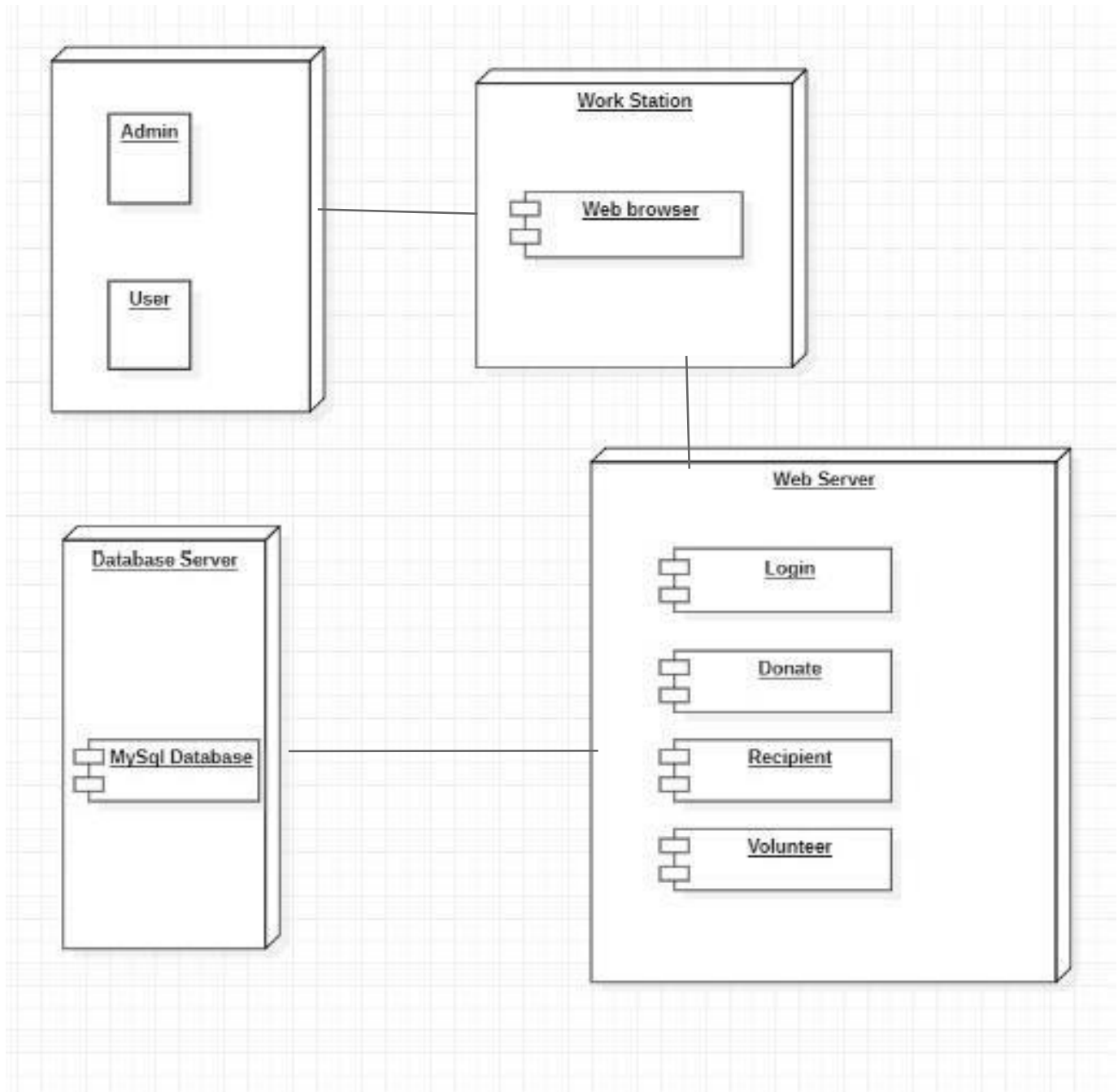


Table Design

Introduction

A table is a collection of related data held in a table format within a database it consists of columns and rows

In relational database and flat file databases a table is a set of data elements (values) using a model of vertical columns (identifiable by name) and horizontal rows, the cell being the unit where a row and column intersect A table has a specified number of columns, but can have any number of rows. Each row is identified by one or more values appearing in a particular column subset. A specific choice of columns which uniquely identify rows is called the primary key.

Table for Admin login:

Sr.No	Column	Datatype	Description
1	Email	Varchar(10)	It stores the Email of admin.
2	Password	Number	It stores the password.

Table for NGO login:

Sr.No	Column	Datatype	Description
1	Email	Varchar(10)	It stores the Email of Ngo.
2	Password	Number	It store the password.

Table for Donor login:

Sr.No	Column	Datatype	Description
1	Email	Varchar(10)	It stores the Email of donor.
2	Password	Number	It stores the password.
3	Address	Varchar(10)	It stores the address.

Table for Recipient login:

Sr.No	Column	Datatype	Description
1	Email	Varchar(10)	It stores the Email of Receiver.
2	Password	Number	It stores the Password
3	Address	Varchar(10)	It stores the Address

Table for Volunteer login:

Sr.No	Column	Datatype	Description
1	Email	Varchar(10)	It stores the Email of Volunteer.
2	Password	Number	It stores the password.

Table for Medicine information:

Sr.No	Column	Datatype	Description
1	Medicine name	Varchar(10)	It stores the name of medicine.
2	Medicine quantity	Number	It stores the quantity of medicine.
3	Expiry date	Date	It stores the expiry date.