DRUCICARE PHARMACY PROJECT

[CSC824 Group Project]

AKPODONOR Annabel
BEN-IZUAGIE Roseline
ERIAMIANTOE Jeffrey I.
GUOBADIA Timothy
IDEHEN Ernest
[Group 1]

04-May-21

DRUCICARE PHARMACY, OVBIOGIE COMMUNITY, BENIN CITY.

System Chosen: Medium-Sized Pharmacy **Location:** Ovbiogie Community.

Upon receipt of the assignment, the team met and decided to swing into action on the same day as we had a very limited time.

The pharmacy (**Drucicare**) located at Ovbiogie community; Benin City was visited at about 4:20pm on Tuesday 9th of February, 2021.

On arrival at the pharmacy, we inquired about the overall head (owner of the pharmacy) and he was said to be unavailable. One of the girls (a worker) we met also claimed not to be in the position to answer our questions/enquiries.

We however chose to stick with the **observation fact-finding technique**.

In this fact finding technique, the analyst visits the organization, observes and tries to understand the workflow, the existing system, the system users etc. This method is best carried out by the analyst himself as he knows which points should be noted and highlighted.

OVERVIEW OF THE EXISTING SYSTEM:

From our observation, we observed that the pharmacy have four (4) members of staff consisting of three (3) female and one (1) male.

When customers come to purchase drugs, any free/available staff attends to the customer.

The customer tells the staff what he/she wants and the staff goes to the shelf (where drugs are arranged), if the drug is available, it is brought to the customer and he/she makes payment for the drug (to the same staff who attended to him/her). If the drug is not available on the shelf, the staff may go into the store (like a mini warehouse where the drugs are kept, from where they are arranged in

the shelf) and check for the drug. If available, it is brought to the customer and payment is also made to the staff.

The staff that made the sale then goes to drop the money in a particular carton/box where they drop monies realized from sales.

MAIN PROBLEMS OBSERVED:

- **OVERALL CUSTOMER TIME AND PHARMACY'S PRODUCTIVE TIME WASTAGE:**

  We observed a case when the staff didn't remember the exact location/shelf a particular drug is kept. It took some minutes to locate this drug before the transaction was finally sealed with the customer.

  In another case, the staff had to go into the store to check for the drugs and eventually the drug wasn't available.

  On the side of the customer, it is very frustrating to waste fruitful time in waiting for a drug and in the end it is reported not to be available.

  On the side of the pharmacy, time and efforts spent in searching for unavailable drugs in the store would have been channeled into attending to other customers.

  There was also a case of one of the staff not knowing the exact price/cost of a particular drug; she had to seek the attention of the other girls before she attended to the already waiting customer.

- **POOR ACCOUNTABILITY/RECORD KEEPING:**

  In the time spent observing this pharmacy, there was no form of record taking on a logbook or whatsoever. This may create futuristic problems as regard stocks and even customer return policy.

  As long as anybody in charge of anything, nobody can really be held accountable to a particular issue.

Members of staff may even make sales without dropping the money in the allocated box and the particular staff can/may not be fished out.

RECOMMENDED SOLUTIONS:

- For the issue of time wastage, we recommend the design of a system that has the complete list of drugs tagged to their residing shelves and points out availability.

  For every drug inquired/demanded by the customers, the staff runs a quick check on the system for the drug and the price is displayed, alongside the availability and exact location of the drug say A3. At this point, the shelf must have been tagged/labeled accordingly.
- For the issue of accountability/record taking, each staff should be designated to a specific role. Let's say two staff assigned to the systems, one assigned to retrieving the drugs from the shelf and one assigned to the finance (receiving and confirming payments in case of cashless payments).

  Each staff will have his/her login and every transaction can be traced to a specific staff.

OVERVIEW OF THE RECOMMENDED SYSTEM:

In eradicating the problems observed and unobserved, we totally recommend a new system which will reduce overall time spent on one customer and also increase the productive time of the pharmacy.

The recommended software will be designed to allow for the taking of stocks from the backend by the supervisor or overall boss or anybody he assigns to the role.

When goods are supplied to the pharmacy, the stock is taken and entered into the system (software). On every sale, the number of a particular drug sold will be deducted from the available stock. This can easily point to the staff when a

particular drug is out of stock, and in turn reduce the time spent searching for unavailable drugs which is a waste to the customers and also increase the productive time of the pharmacy staff, as they now have more time to attend to other customers.

Every staff will have a unique login ID, every transaction done will be tagged to their unique ID, which will bring out accountability and also erase the issue of poor record taking.

When a customer comes to buy a drug, the staff keys in the name and selects the drug (it will show availability, location and price), the staff prints out an invoice which the customer takes to the finance staff (cashier) for payment and the cashier endorses it when payment is made/confirmed. While the customer is away with the cashier, the staff on the system signals the staff in charge of the shelf to bring the specific drug. After payment, the customer tenders his/her payment slip and goes with his/her drug.

All financial transactions can only be done with the cashier and no other person.

This will bring about a sense of proper documentation or record keeping as daily/weekly/monthly records can be checked from the backend.

With this recommended system, members of staff will hardly have a loophole to cheat the pharmacy as in the case of the existing system.

FEASIBILITY STUDY

A feasibility study is an analysis that takes all of a project's relevant factors into account/consideration – including the economic, technical, legal, operational considerations, to ascertain the likelihood of completing the project successfully.

Technical Feasibility:

In the technical feasibility study, we analyze and determine whether the existing system can support the proposed solution or not, or the availability of necessary facilities/requirements to effectively run the proposed system.

From our study, the current/existing system lacks the necessary requirement in terms of technical facilities. The pharmacy will have to purchase or however, make available the needed facilities which include;

i. Two (2) Desktop Computers *(Minimum of 4GB RAM and 500GB ROM)*
ii. One (1) thermal Printer
iii. One (1) Point of sale (POS) machine

The pharmacy already has 4 staff, which will be enough to run the proposed system. However, the staff will have to undergoing some training at the emergence of the new system.

Operational Feasibility:

In the operational feasibility study, we study how well the proposed system will solve the existing problems and how well the proposed system fits into the existing business environment.

From our study of the existing system and a critical analysis of the proposed system, we can say the proposed system will solve the identified problems effectively.

Economic Feasibility:

In the economic feasibility, we identify the cost and the benefits associated with a development project. The purpose of cost benefit analysis is to measure the costs associated with the new system and compare it with the benefits attached to implementing the system.

We apply the COCOMO model of cost estimation in this project. COCOMO stands for constructive cost estimation model. This model is based on the lines of code (LOC). Where project estimation is done based on the total lines of codes required to develop the system i.e. size of the system define the cost. The COCOMO model was developed by Barry W. Boehm in 1981. The model is used to estimate the effort cost, development time, average staff size, productivity, etc.

Basic COCOMO equations

Effort = a(KLOC)^{b}

Development time = c(Effort)^{d}

Average staff size = Effort/Development time

Productivity = KLOC/Effort

[Where a, b, c, d are constants/coefficients based on the mode of development]

Semi-Detached: KLOC = 60 [a=3.0, b=1.12, c=2.5, d=0.35]

(gotten from the coefficient table)

Effort = a(KLOC)^{b}

= 3.0(60)$^{1.12}$

=294.205 person-months

Dev. Time = c(Effort)^{d}

= 2.5(294.205)$^{0.35}$

=18.27 months

BENEFITS OF THE PROPOSED/RECOMMENDED SYSTEM

- There will be improved/better documentation of stock(s)
- Better/improved record of sales/transactions

- Reduced stress on the staff *(based on the reduced time spent on searching for drugs, especially when they are unavailable and the staff wasn't aware)*
- Shorter waiting time for customers

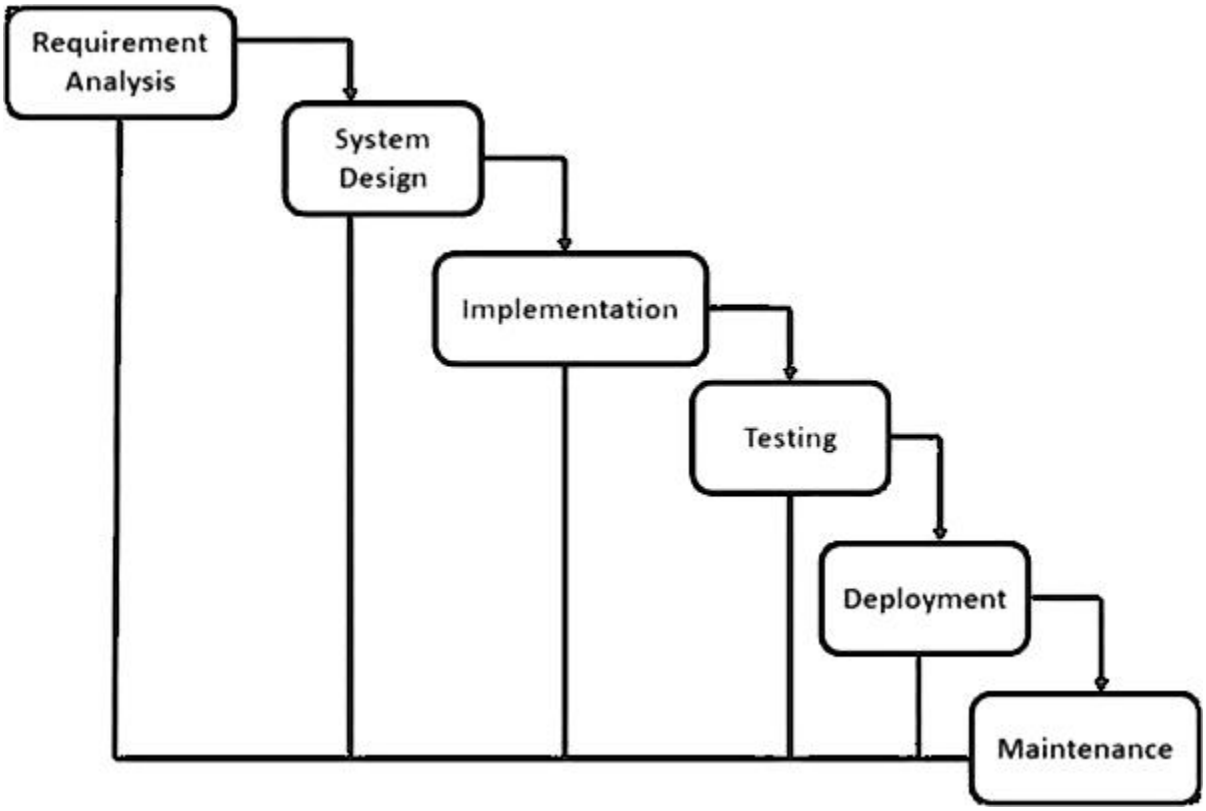
SYSTEM DEVELOPMENT LIFE CYCLE – Waterfall Model

It is also referred to as a linear-sequential life cycle model. It is very simple to understand and use. In a waterfall model, each phase must be completed before the next phase can begin and there is no overlapping in the phases.

The Waterfall model is the earliest SDLC approach that was used for software development.

The waterfall Model illustrates the software development process in a linear sequential flow. This means that any phase in the development process begins only if the previous phase is complete. In this waterfall model, the phases do not overlap.

The following illustration is a representation of the different phases of the Waterfall Model.



The sequential phases in Waterfall model are:

- **Requirement Gathering and** analysis − All possible requirements of the system to be developed are captured in this phase and documented in a requirement specification document.
- **System Design** − the requirement specifications from first phase are studied in this phase and the system design is prepared. This system design helps in specifying hardware and system requirements and helps in defining the overall system architecture.
- **Implementation** − with inputs from the system design, the system is first developed in small programs called units, which are integrated in the next phase. Each unit is developed and tested for its functionality, which is referred to as Unit Testing.

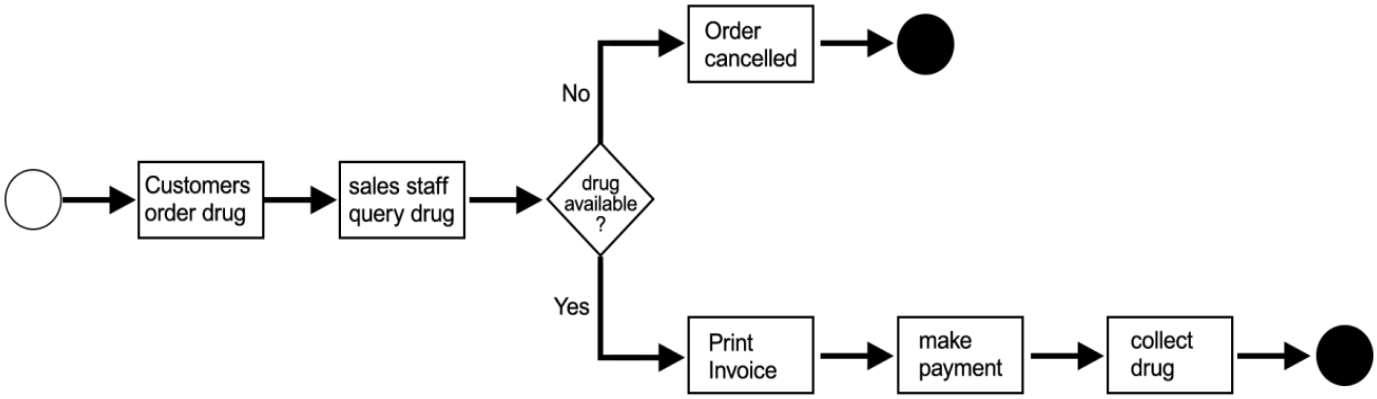
- **Integration and Testing** − All the units developed in the implementation phase are integrated into a system after testing of each unit. Post integration the entire system is tested for any faults and failures.
- **Deployment of system** − Once the functional and non-functional testing is done; the product is deployed in the customer environment or released into the market.
- **Maintenance** − There are some issues which come up in the client environment. To fix those issues, patches are released. Also to enhance the product some better versions are released. Maintenance is done to deliver these changes in the customer environment.

All these phases are cascaded to each other in which progress is seen as flowing steadily downwards (like a waterfall) through the phases. The next phase is started only after the defined set of goals are achieved for previous phase and it is signed off, so the name "Waterfall Model". In this model, phases do not overlap.

Business Process Model

Business process modeling (or) process modeling is the analytical representation or put simply an illustration of an organization's business processes. Modeling processes is a critical component for effective business process management.

Process modeling a project gives an analytical representation of the processes in an organization.



Business Process Model

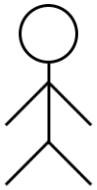
The Use Case Diagram

A use case diagram is a technique of summarizing details of a system and the users within that system. It is generally shown as a graphic depiction of interactions among different elements in a system. Use case diagram specifies the events in a system and how those events flow.

Use case diagrams consist of actors (some internal or external agents for making the interaction), use cases and their relationships. The diagram is used to model the

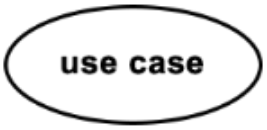
system/subsystem of an application. A single use case diagram captures a particular functionality of a system.

Actors: they are users that interact with the system for the project to achieve certain goals. They can be persons, organizations, external devices, etc. Actors are represented outside the system and there can be multiple actors in a use case diagram. Actors are represented with the symbol below;

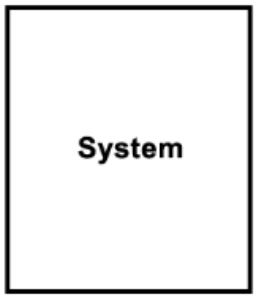


Use Case: A use case is an action or task typically defining the interaction between an actor and a system to produce a result, the actor can be humans or other external systems. They are tasks executed in the system.

It is represented with the symbol below;



The System: The system is the actual project being represented diagrammatically.



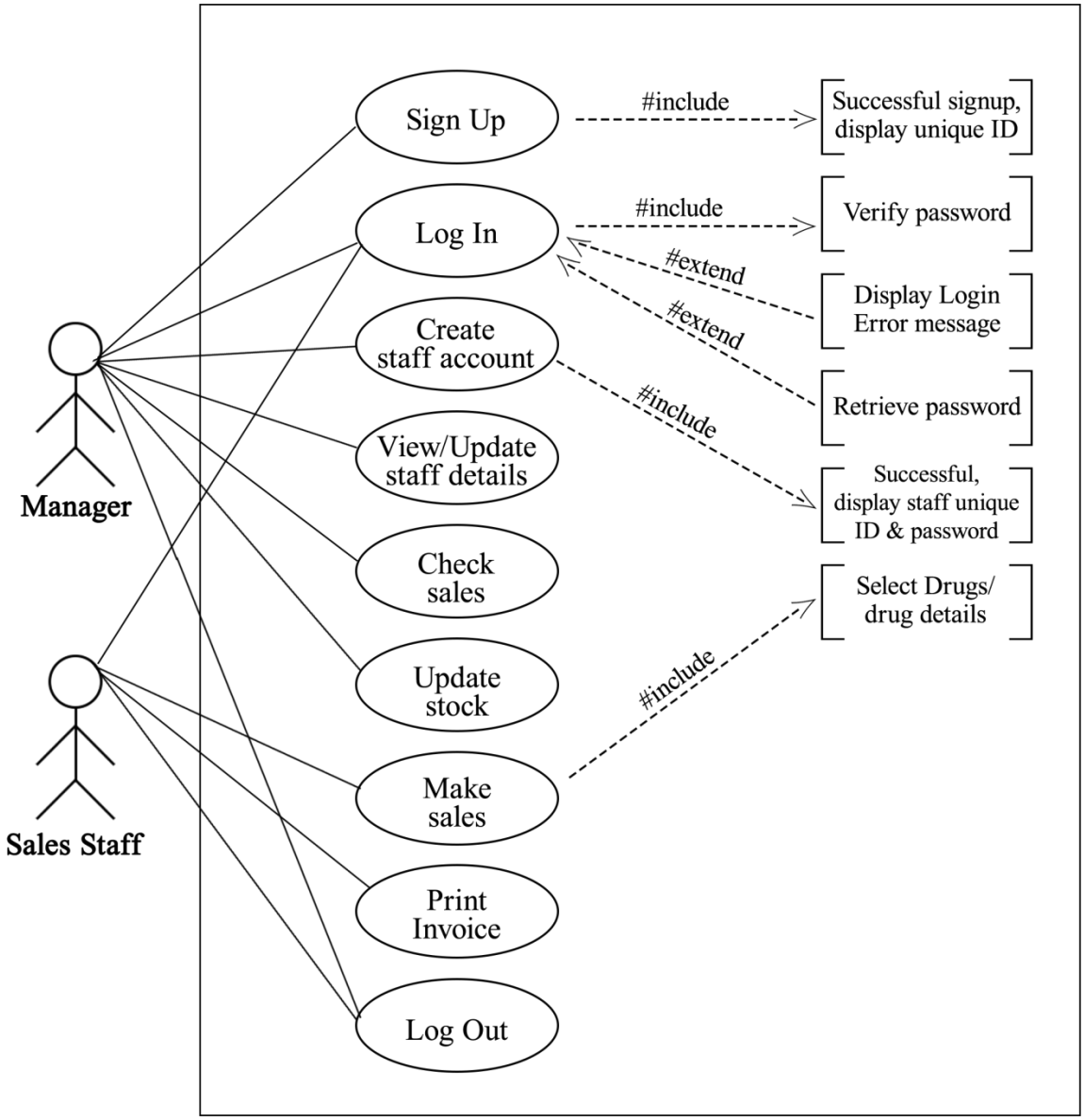
The system can be a website, banking application, e-learning platform, etc.

Systems are represented with a vertically erect rectangle as shown below;

Relationship: There are 4 forms of relationships;

- **The Association Relationship:** This is when an actor communicates with a use case; this is referred to as a communication relationship or association.
- **The Include Relationship:** The include relationship links a primary or base use case with one or more secondary use cases. It is useful to model an include relationship if parts of the use case occur in several use cases.

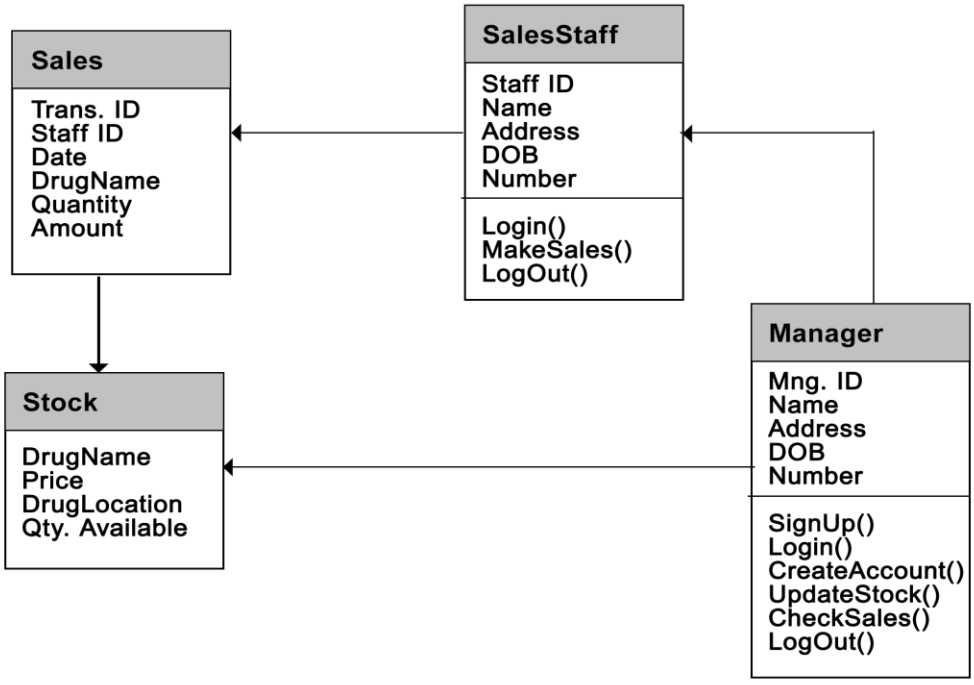
  There are 5 aspects to include relationship;
  - An included use case is always executed when the included use case is executed.
  - An included use case can also be executed separately.
  - The primary and secondary use case should be described at similar levels of abstraction.
  - Descriptions can be used as often as desired by include relationships.
  - In contrast to generalization, no properties are inherited.
- **The Extend Relationship:** If parts of a use case are only executed under defined conditions, the extend relationship adds a function to the basic use case.



Class Diagram

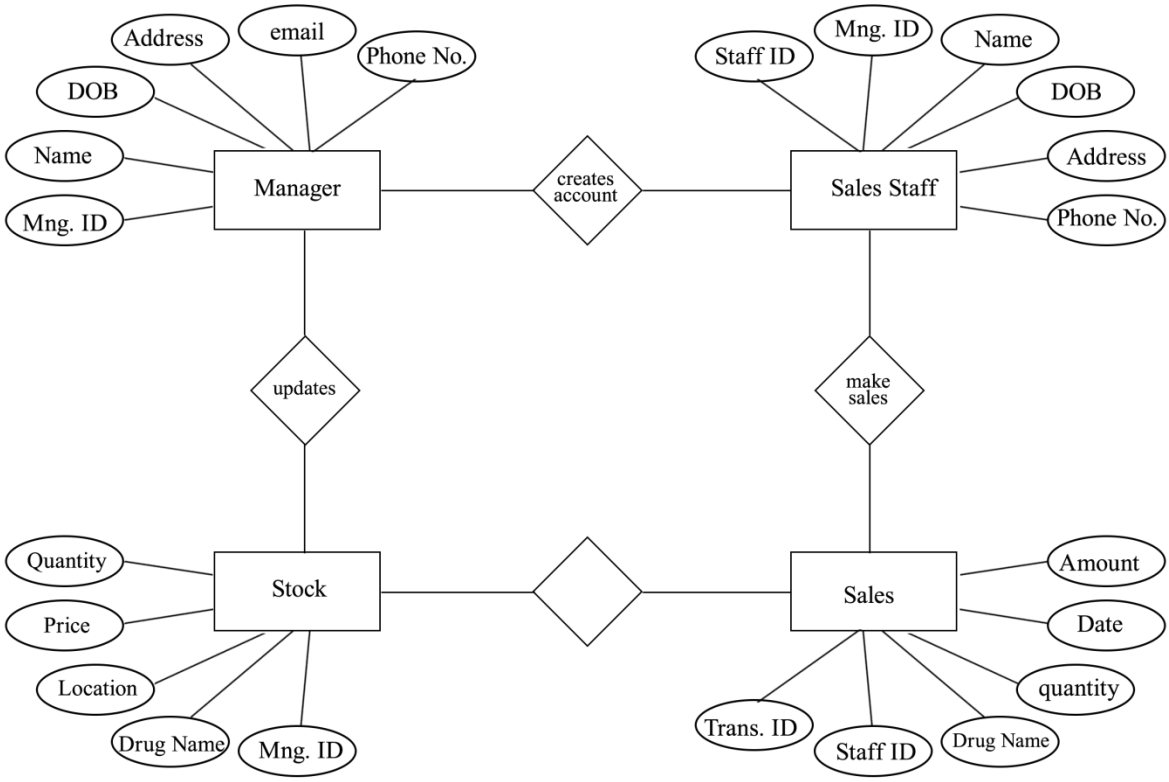
Class diagram describes the attributes and operations of a class and also the constraints imposed on the system. The class diagrams are widely used in the modeling of object-oriented systems because they are the only UML diagrams, which can be mapped directly with object-oriented languages.

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



Entity-Relationship Diagram

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 most often used to design or debug relational databases in the fields of software engineering, business information systems, education and research. Also known as ERDs or ER Models, they use a defined set of symbols such as rectangles, diamonds, ovals and connecting lines to depict the interconnectedness of entities, relationships and their attributes



TOOLS

- **HTML (Hypertext Markup Language):** This is the latest and the most enhanced version of HTML which is also a standard markup language for tagging text files to achieve font, colour, graphics and hyperlink effects on the World Wide Web pages.
- **PHP (Hypertext Preprocessor):** This is a server-side scripting language that is used to enhance webpages; it was conceived in 1994 and was originally the work of Rasmus Lerdorf. It was adopted by other talented people and has gotten through the four major rewrite to bring in the broad mature product that's seen today. PHP is an open source project which means a person can have access to source code and ultimately use, alter and

redistribute it without charge. Creation of pages like signup, log in, can be done with PHP.

- **CSS (Cascading Style Sheet):** This is a style sheet language used for describing the look and formatting a document written in HTML and XHTML, the language can be applied to any kind of XML document, along with HTML CSS, JavaScript.
- **SQL (Structured Query Language):** SQL is used to communicate with a database. According to ANSI (American National Standards Institute), it is the standard language for relational database management systems.

CHANGEOVER TECHNIQUE

The recommended changeover technique is Parallel running. Parallel running is one of the ways to change from an existing system to a new one. It is one of the strategies for system implementation, in which both the old and new systems will be used side by side until the users are certain that the new system is error free. After a period of time, when the system has been proven to work correctly, the old system will be removed completely and users will depend solely on the new system.

Why parallel running?

Parallel running allows the results of the old and new system to be properly compared to ensure the new system is error free. If errors are found, we can refer to the old system to resolve the problem and make modifications to the new system. Thus, operations will not be suspended as a result of errors in the new system rather, the operations will continue with the old system till the new system is rectified.

Parallel running also provides the opportunity for staff to be trained with the new system and help them gain confidence they desire to use the new system.