Assignment - 5

PAGE NO.:
DATE: / /

1. Explain different types of change over methods.

→ As technologies change, many business find themselves needing to change over their computer information systems.

- Upgrading these systems helps them optimize their efficiency and remain competitive.
- Common changeover areas include security systems, database systems, accounting systems and managerial information systems. Deciding which changeover technique will work best for a particular company depends on the type of changeover and degree of risk for the company.
- Types of change over :-
  (1) Data Conversion
  (2) Application / system changeover :-
  (3) parallel change over
  (4) Direct change over

(1) Data Conversion :-

- It is important part of system installation process.
- During data conversion, existing data is loaded into new system.
- It can be done before, after or during operational environment is complete.
- We should develop data conversion plan as early as possible and the conversion process should be tested when tested environment is developed
- When new system replace an existing system, you should automate the data conversion process.
- The old system might be capable of exporting data in an acceptable format.

- Data Conversion is more difficult when the new system replace manual system because all data must be entered manually.

(2) Application / System changeover :-

- changeover or Conversion is the process of changing from the old system, which is currently running in the organization, to the newly built system.
- There are four methods of handling the system Conversion which are :
  - parallel system method
  - Dual system method
  - Direct Cutover method
  - Pilot Approach method

(3) parallel changeover :-

- It require both old and new system operate fully for a specified period.
- Data is input in both system and output generated by new system is compared with the equivalent output from the old system.
- when user, management and IT group are satisfied that the new system operates correctly, old system is terminated.
- Advantages :-
  - lower risk : If the new system does not work correctly, company can use old system as backup.
  - Easier to varify new system : Output of new system is compare with old system and varified during parallel operation.

- Disadvantages :
  - Costly : Company pay for both System because both System are in full operation.
  - Processing Delay : Running both System might place a burden on operating environment and cause processing delay.
  - Work and data are duplicated.
  - Time Consuming.

(4) Direct changeover :-
- Direct changeover, also referred to as immediate replacement, tends to be the least favorite of the changeover technique.
- In a direct changeover, the entire System is replaced in an instant.
- This involves taking the old System offline and putting the new System online within a day or over a weekend or holiday period.
- It is least expensive because IT group has to operate and maintain only one system at a time.
- There are no parallel activities and there is on falling back to old System.
- It is only choice if the operating environment cannot support both the old and new System or if the old and new System ~~or if the old an~~ are incompatible.
- Most organization use direct cutover only for noncritical situations.
- Timing is important when using direct changeover strategy.

- Advantages:
  - Minimal cost: IT group has to operate and maintain only one system at a time.
  - Minimize workload.
- Disadvantages:
  - More Risk: Because if changeover fail stop entire organization.
  - Require careful planning: If something goes wrong, reverting back to the old system usually is impossible.

(5) Phase In / Dual changeover:
- The phased changeover technique is considered a compromise between parallel and direct changeover.
- In a phased changeover, the new system is implemented one stage at a time.
- phased operation works in different phases or stages.
- Implementation of new system in modules or stages is phased operation.
- In this method, new system is gradually phased out while the new one is being phased in.
- In phase operation the risk of errors or failures is limited to the implemented module only and also phased operation is less expensive than the full parallel operation.
- The actually conversion from the old parts of the system to the new parts may be either parallel or direct.
- As an example, consider a company working towards installing a new financial system. Implementing the

new system one department at a time, the company converts accounts receivable, accounts payable, payroll, and so on.

- This method is used when it is not possible to install a system throughout an organization all at once.

- Advantages:
  - less risky because implemented in phase so no danger of total breakdown.
  - Any problem should be in one area other area operations are unaffected.

- Disadvantages:
  - Can take long time to achieve total changeover.
  - Interface between parts of the system may make this impractical.

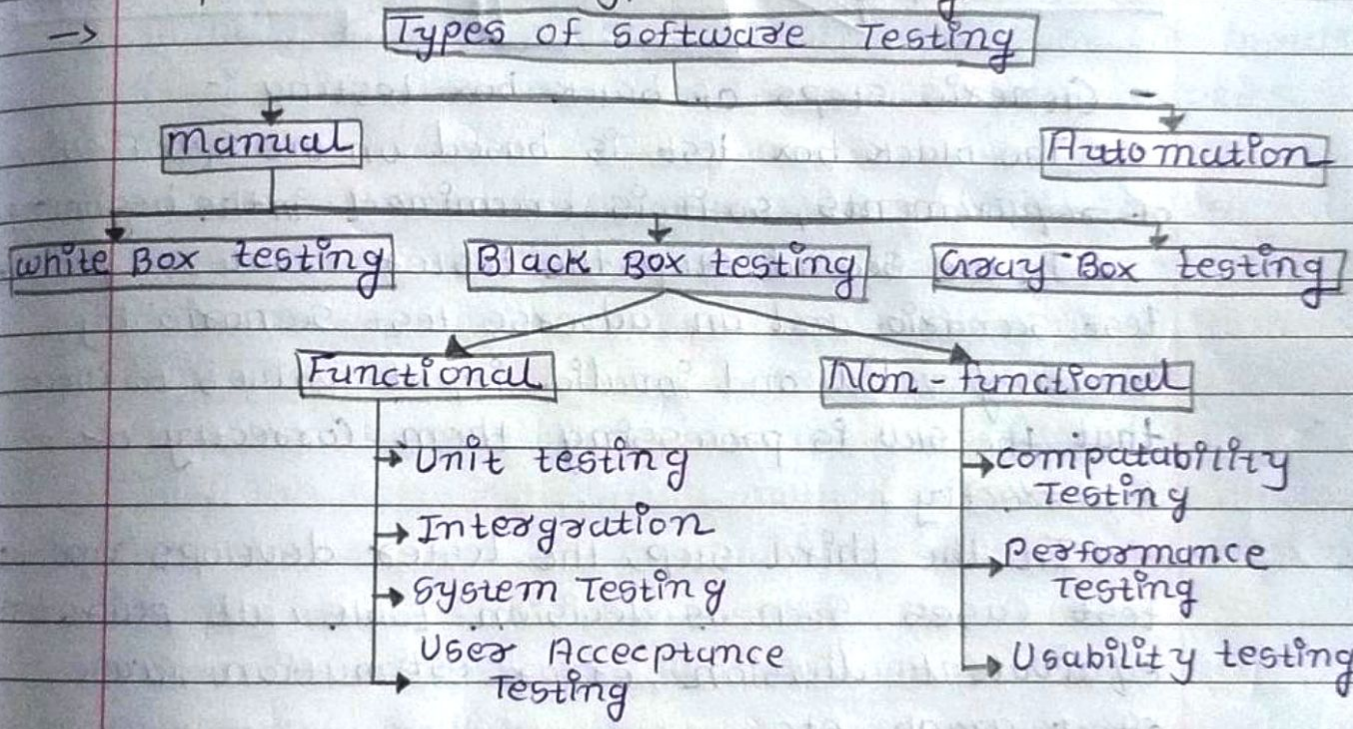
(6) Pilot changeover:-

- In the pilot Approach method, a complete new version of the system is implemented in just one location or site of the organization.
- For example, a bank may first test the system at one of its branches. This branch is refered to as the pilot, or beta, site for the program. The old system continues to operate for the entire organization including the pilot site.
- After the system ~~continues to operate for~~ proves successful at the pilot site, it is implemented in the rest of the organization, usually using direct cutover method.

- Pilot operation is combination of parallel operation and direct cutover methods.
  - Advantages:
    - less risky
    - Less costly
  - Disadvantages:
    - Can take a long time to achieve total changeover.

2. Explain different types of testing.

→



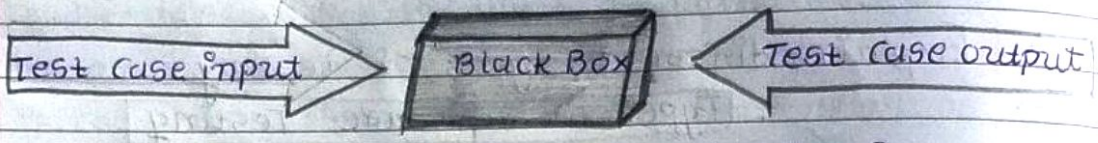
[1] BLACK BOX TESTING :-

- This method is named Black box because the sw program, in the eyes of the tester, is like a black box; inside which one cannot see.
- It is a sw testing method in which the internal structure / design / implementation of the item being tested is not known to the tester.

DATE: / /

- Test design techniques include:
  - Equivalence partitioning
  - Boundary value Analysis
  - Cause Effect Graphing
- Also known as functional testing.

Black Box Testing



- Generic steps of black box testing :-
  - The black box test is based on the specification of requirements, so it is examined in the beginning.
  - In the second step, the tester creates a positive test scenario and an adverse test scenario by selecting valid and invalid input values to check that the s/w is processing them correctly or incorrectly.
  - In the third step, the tester develops various test cases such as decision table, all pairs test, equivalent division, error estimation, cause-effect graph, etc.
  - The fourth phase includes the execution of all test cases.
  - In the fifth step, the tester compares the expected output against the actual output.
  - In the sixth and final step, if there is any flaw in the s/w, then it is cured and tested again.

- Advantages :
  - Tester can be non-technical.
  - Used to verify contradictions in actual system and the specification.
  - Test cases can be designed as soon as the functional specifications are complete.
- Disadvantages :-
  - The test inputs needs to be from large sample space.
  - It is difficult to identify all possible inputs in limited testing time. So writing test cases is slow and difficult.
  - chances of having unidentified paths during this testing.

[2] WHITE BOX TESTING :-

- It is a software testing method in which the internal structure / design / implementation of the item being tested is know to the tester.
- Also known as : clear box testing, glass box testing, structural testing.
- It tests internal coding and infrastructure of a s/w focus on checking of predefined inputs against expected and desired outputs. It is based on inner workings of an application and revolves around internal structure testing.
- In this testing programming skills are required to design test cases. The primary goal of white box testing is to focus on the flow of inputs and outputs through the s/w and strengthening the security of the s/w.

- Developers do white box testing. In this, the developers will test every line of the code of the program. The developers perform the white-box testing and then send the application or the s/w to the testing team, where they will perform the black box testing and verify the application along with the requirements and identify the bugs and sends it to the developer.
- The developer fixes the bugs and does one round of white box testing and sends it to the testing team. Here, fixing the bugs implies that the bug is deleted, and the particular feature is working fine on the application.
- Test design techniques include:
  1. Control flow testing
  2. Data flow testing
  3. Branch testing
  4. path testing
  5. Code coverage testing

- Generic Steps of white box testing:-
  - Design all test scenarios, test cases and prioritize them according to high priority number.
  - This step involves the study of code at runtime to examine the resource utilization, not accessed areas of the code, time taken by various methods and operations and so on.
  - In this step testing of internal subroutines takes place, Internal subroutines such as nonpublic methods, interfaces are able to handle all types of data appropriately or not.
  - This step focuses on testing of control statements like loops and conditional statements to

check the efficiency and accuracy for different data inputs.
- In the last step white box testing includes security testing to check all possible security loopholes by looking at how the code handles security.

- Purpose of white box testing :-
  - Testing all loops, testing Incorrect assumptions.
  - Testing Basis paths, testing Conditional statemen
  - Testing data structure, Testing logic Errors

- Advantages :-
  - It optimizes code so hidden errors can be identified.
  - Test cases of white box testing can be easily automated
  - This testing is more thorough than other testing approaches as it covers all code paths.
  - It can be started in the SDLC phase even without GUI.

- Disadvantages :-
  - white box testing is too much time consuming when it comes to large-scale programming applications.
  - It is much expensive and complex.
  - It can lead to production error because it is not detailed by the developers.
  - It needs professiond programers who have a detailed knowledge and understanding of programming language and implemention.

[3] UNIT TESTING :-

- In focuses on the Smallest unit of SW design. In this, we test an individual unit or group of interrelated units.

It is often done by the programmer by using sample input and observing its corresponding outputs.

- Examples:

a) In a program we are checking if loop, method or function is working fine.

b) Misunderstood or incorrect, arithmetic precedence.

c) Incorrect initialization.

- It involves the testing of each unit or an individual component of the S/W application. It is the first level of functional testing. The aim behind unit testing is to validate unit components with its performance.
- A unit is a single testable part of a S/W system and tested during the development phase of the application software.
- The purpose of unit testing is to test the correctness of isolated code. A Unit component is an individual function or code of the application. white box testing approach used for unit testing and usually done by the developers.
- Whenever the application is ready and given to the test engineer, he/she will start checking every component of the module or module of the application independently or one by one, and this process is known as Unit testing or Components testing.
- Unit testing helps tester and developers to understand the base of code that makes them able to change defect causing code quickely.
- Unit testing helps in the documentation.
- Unit testing fixes defects very early in the development phase that's why there is a possibility to occur a smaller number of defects

in upcoming testing levels.
- It helps with code reusability by migrating code and test cases.

[4] INTEGRATION TESTING :-

- The objective is to take unit tested components and build a program structure that has been dictated by design. Integration testing is testing in which a group of components is combined to product output.
- Integration testing is of four types:
  (i) Top-down (ii) Bottom-up (iii) Sandwich (iv) Big-Bang

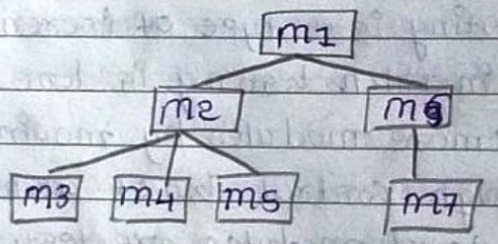
- (i) Top-down :-
  -> Top-down testing is a type of incremental integration testing approach in which testing is done by integrating or joining two or more modules by moving down from top to bottom through control flow of architecture structure.
  -> In this high-level modules are tested first, and then low-level modules are tested.
  -> Following are the steps that are needed to be followed during processing :
  ① Test drivers represents main control module also known as a high-level modules and Stubs are generally used for all low-level modules that directly subordinate (present or rank below another) to high-level modules.
  ② In this, testing takes place from to bottom, so, top or high-level modules are tested first in isolation.
  ③ After this, low-level modules or subordinated modules or stubs replace high-level module one by one at a time. This can be done using methods like depth-first or breadth search.

④ The process is repeated until each module is integrated and tested.

⑤ Another stub replaces present real or control module after completion of each set of tests. These stubs act as a temporary replacement for a called module and give same result or output as actual product gives.

⑥ To check if there is any defect or any error occurred or present, regression testing is done and it's important to reduce any side effect that might be caused due to errors occurred.

- Example : In the top-down testing, if depth-first approach is adopted then we will start integration from module M1. The we will integrate one by one



Program structure

- Advantages :-
  - There is no need to write drivers.
  - Interface errors are identified at an early stage and fault localization is also easier.
  - How-level utilities that are not important are not tested well and high-level testers are tested well in an appropriate manner.
  - Representation of test cases is easier and simple once Input-Output functions are added.

- Disadvantages :-
  - It requires a lot of stubs and mock objects.

- Representation of test cases in stubs can be not easy and might be difficult before Input-Output functions are added.
- How-level utilities that are important are also not tested well.

(ii) Bottom-up :-

→ Bottom-up testing is a type of incremental integration testing approach in which testing is done by integrating or joining two or more modules by moving upward from bottom to top through control flow of architecture structure. In these, how level modules are tested first, and then high-level modules are tested.

→ It is also known as inductive reasoning and is used as a synthesis synonym in many cases. It is user friendly testing.

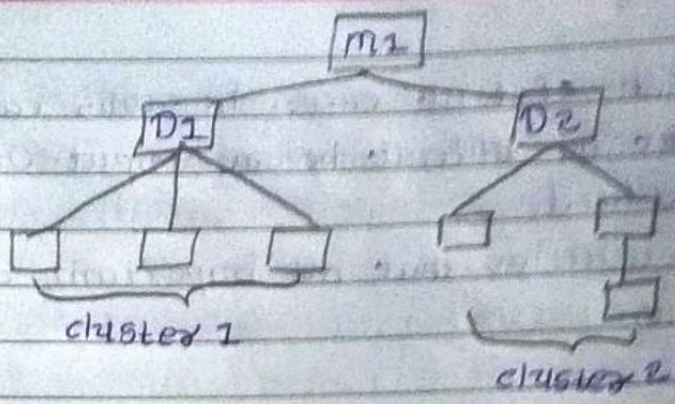
→ process :-

① clusters are formed by merging or combining low-level modules or elements. These clusters are also known as builds that are responsible for performing the certain secondary or subsidiary function of a SW.

② It is important to write a control program for testing. These control programs are also known as drivers or high-level modules. It simply coordinates input and output of a test case.

③ Testing is done of entire build or cluster containing how-level modules.

④ At lastly, control program or drivers or high levels modules are removed and clusters are integrated by moving upward from bottom to top in program structure with help of control flow.



(Bottom Up Integration Testing)

- Advantages :-
  - It is easy and simple to create and develop unit tests.
  - It is also easy to observe test results.
  - It is not necessary to know about the details of the structural design.
  - low-level utilities are also tested well and are also compatible with the object-oriented structure.

- Disadvantages :-
  - Towards top of the Hierarchy, it becomes very complex.
  - There is no concept regarding early skeletal System.
  - There will be an impact on sibling and higher-level unit tests due to changes.

(5) SYSTEM TESTING :-

→ This s/w is tested such that it works fine for the different operating systems. It is covered under the black box testing technique. In this, we just focus on the required input and output without focusing on internal working.
→ In this, we have security testing recovery testing, stress testing, and performance testing.
→ It is carried out on the whole System in the context of either System requirement specifications or functional requirement specifications or in the context of both. System testing tests the design and behavior of the system

and also the expectations of the customer.

→ Examples: This include functional as well as non functional testing.

→ System Testing process:

① Test Environment setup: create testing environment for the better quality testing.

② Create Test Case: Generate test case for the testing process.

③ Create test data: Generate the data that is to be tested.

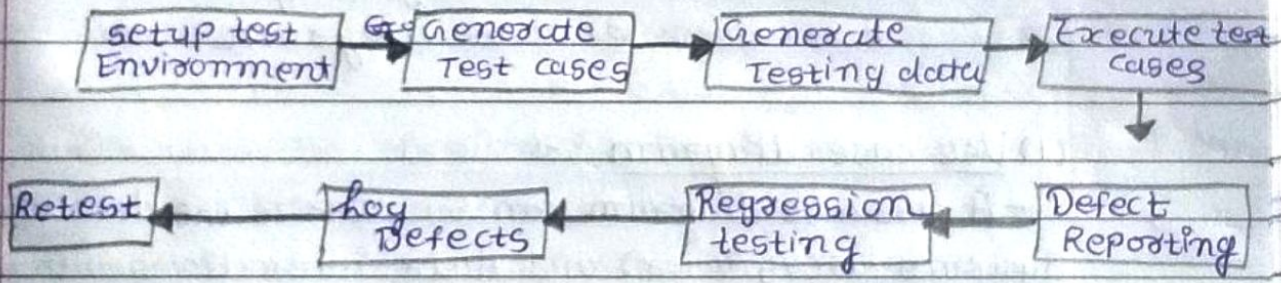
④ Execute test case: After the generation of the test case and the test data, test cases are executed.

⑤ Defect Reporting: Defects in the system are detected.

⑥ Regression Testing: It is carried out to test the side effects of the testing process.

⑦ log: Defects are fixed in this step.

⑧ Retest: If the test is not successful then again test is performed.



→ Types of testing :-

[1] Performance Testing: It is a type of sw testing that is carried out to test the speed, scalability, stability and reliability of the sw product or application.

[2] load: load Testing is a type of sw testing which is carried out to determine the behavior of a system or sw product under extreme load.

[3] Stress testing: it is a type of sw testing performed to check the robustness of the system under the varying loads.

PAGE NO.:
DATE: / /

[4] Scalability testing : It is a type of SW testing which is carried out to check the performance of a SW application or System in terms of its capability to scale up or scale down the number of user request load.

3. What is UML ? Explain use case diagram and Class diagram in detail.

→ UML stands for Unified Modeling language. It is general-purpose, developmental, modeling language in the field of SW engineering that is intended to provide a standard way to visualize the design of a system.

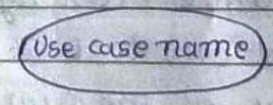
- UML Diagrams : (1) Use case diagram
(2) Activity diagram
(3) Class diagram

(1) Use case diagram :-

- A use case diagram can summarize the details of your System's users (actor) and their interactions with the system

- Symbols of use case diagram :

= Actor



= Use case

= Generalization symbol used between actors and between use cases.

_____ = Association between actor and use case

<<include>> = Include relationship between use cases

.....→

PAGE NO.:
DATE: / /

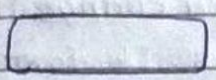
<<extend>> ⟵ = Extend relationship between use case

(2) Activity diagram:-

- It is basically a flowchart to represent the flow from one activity to another activity.
- Symbols of Activity diagram:-

 = initial-state

 = decision-box

 = action-box

 = final-state

(3) class diagram:-

- It represents the types of objects residing in the system and the relationships between them.
- It is used to visualize, describe, document various different aspects of the system, and also construct executable s/w code. It shows the attributes, classes, functions, and relationship to give an overview of the s/w system.
- purpose of class diagram:-
  - The main purpose of class diagrams is to build a static view of an application.
  - It describes the major responsibilities of a system.
  - It is a base for component and deployment diagrams.
  - It incorporates forward and reverse engineering.
- Benefits of class diagram:-
  - It can represent the object model for complex system
  - It reduces the maintenance time by providing an overview of how an application is structured before coding.
  - It provides a general schematic of an application for better understanding.

PAGE NO.:
DATE: / /

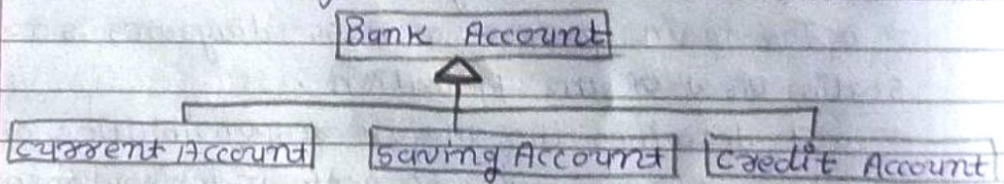
- It represents a detailed chart by highlighting the desired code, which is to be programmed.
- It is helpful for the stakeholders and developers.

- The class diagram is made up of three sections: className, attributes, methods.

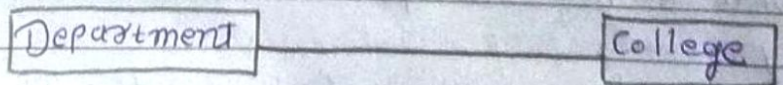
- Relationships:

① Dependency: A dependency is a semantic relationship between two or more classes where a change in one class cause changes in another class. It forms a weaker relationship. In the following example, Student_Name is dependent on the Student Id.

② Generalization: It is a relationship between a parent class and a child class (subclass). In this, the child class is inherited from the parent class.
Ex: The current Account, Saving Account and credit Account are the generalized form of Bank Account.



③ Association: It describes a static or physical connection between two or more objects. It depicts how many objects are there in the relationship. for example, a department is associated with the



④ Composition: The composition is a subset of aggregation. It portrays the dependency between the parent and its child, which means if one part is deleted, then the other part also gets discarded. it represents a whole part relationship.
- A contact book consists of multiple contacts, and if you delete the contact book, all the contacts will be lost.

→ How to draw a class diagram?
(1) to describe a complete aspect of the system, it is suggested to give a meaningful name to the class diagram
(2) The objects and their relationships should be acknowleged in advance.
(3) The attributes and methods of each class must be known.
(4) A minimum number of desired properties should be specified as more number of the unwanted property will lead to a complex diagram.
(5) Notes can be used as and when required by the developer to describe the aspects of a diagram.
(6) The diagrams should be redrawn and reworked as many times to make it correct before producing its final version.

4. What is DFD? Explain DFD with Data Dictionary and process specification.

→ Through a structured analysis technique called data flow diagrams (DFD), the System analyst compiles together a graphical representation of data processes throughout the system.

- The data flow approach emphasizes the logic underlying the System.
- DFD is a graphical tool which can be used by the analyst to clarify his understanding of the System to the user. DFD can be easily converted into a structure chart which can be used in design phase.
- There are two types of DFD: (1) pysical DFD
(2) logical DFD

(1) pysical Data flow Diagrams: An implementation dependent view of the current System, showing what tasks are carried out and how they are performed. pysical characteristics can include:
o Name of people, form and document names or numbers, Names of departments, master and transaction files, Equipment and devices used, locations, Names of procedures.

(2) logical Data flow Diagram: An implementation independent view of the a System, focusing on the flow of data between processes without regard for the specific devices, Storage locations of people in the system. order processing, ~~System~~ Account verification. login Data verificat

- Symbols used in DFD:

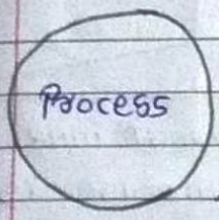
Entity

=> An external entity is a source or destination of a data flow which is outside the area of study. Only those entities which originate or receive data are represented on a data flow diagram.

PAGE NO.:
DATE: / /

Data flow

=> A data flow shows the flow of information from its source to its destination. A data flow is represented by a line, with arrowheads showing the direction of flow.



=> A process shows a transformation or manipulation of data flows within the system, hence the data flow leaving a process is always labeled differently from the one entering it. processes represent work being performed within the system.

Data store

=> A datastore can be thought of as data at rest. It is typically a database but it could be a filling cabinet, a notebook or a datafile. It could also be a permanent or a temporary store. It is represented by an open ended narrow rectangle.

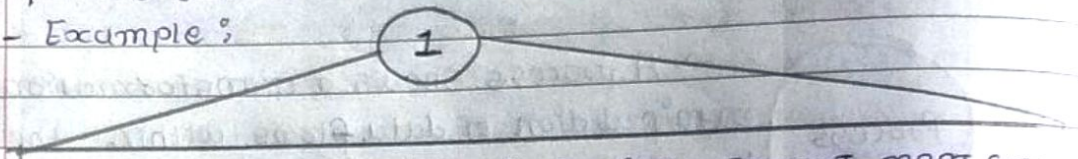
- Data Dictionary :- data dictionary is a collection of data about data. It maintains information about the definition, structure, and use of each data element that an organization uses.

Basic elements of data dictionary :

- Name : the primary name of the data or control item, the data store or an external entity.
- Alias : other names used for the first entry.
- Where-used / how-used : a listing of the processes that use the data or control item and how it is used.
- Description : other information about data types, present values, restrictions or limitations, and so on.

- Process Specification :- A process specification describes the purpose of processes depicted in the DFD, the inputs to the process, and the output. It describes what the process should do and not how it should do it.

- Example :

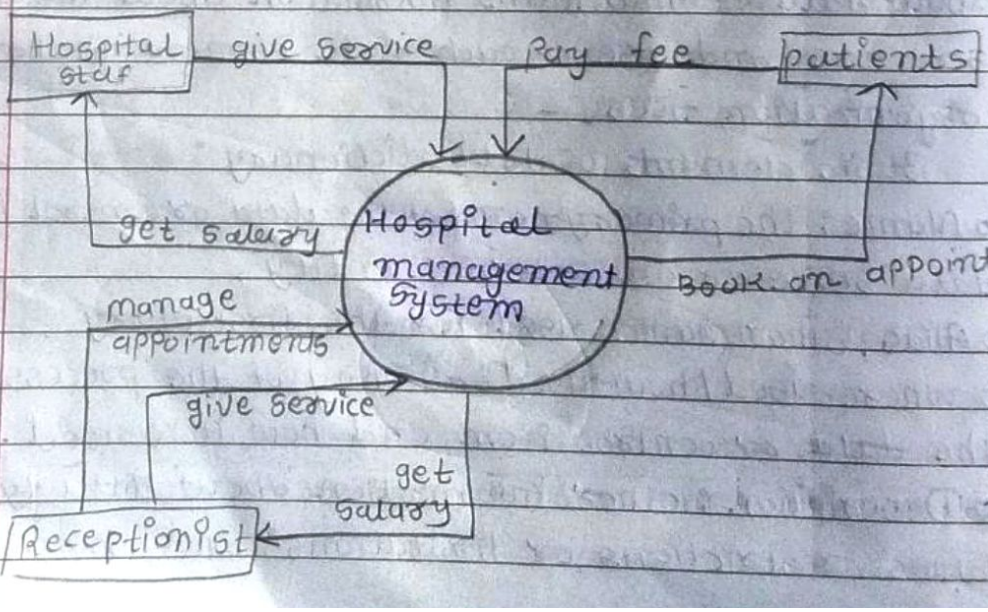


This process is used to login into INVEST MART system. Username and password entered by the user is checked against login mstr Dataset to check whether the user/administrator has entered valid username and password. If valid then user privileges will be provided by the System else the system will prompt for valid username and password.

5. Draw DFD, USECASE diagram, Activity diagram and class diagram of Hospital management system.

- * DFD for Hospital management System :-

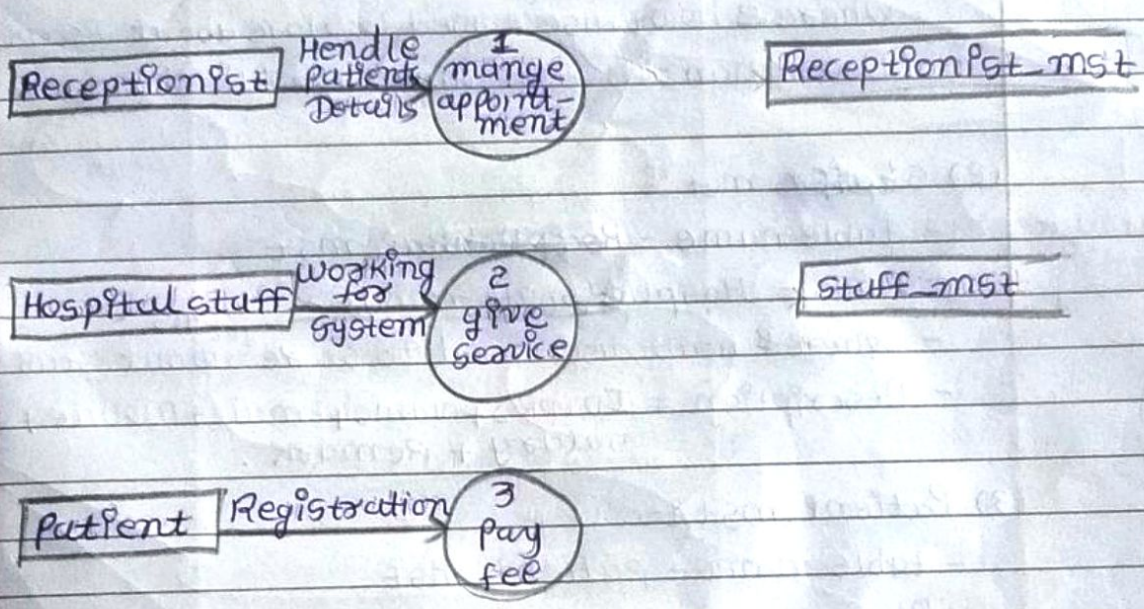
(1) Context or zero level DFD :-



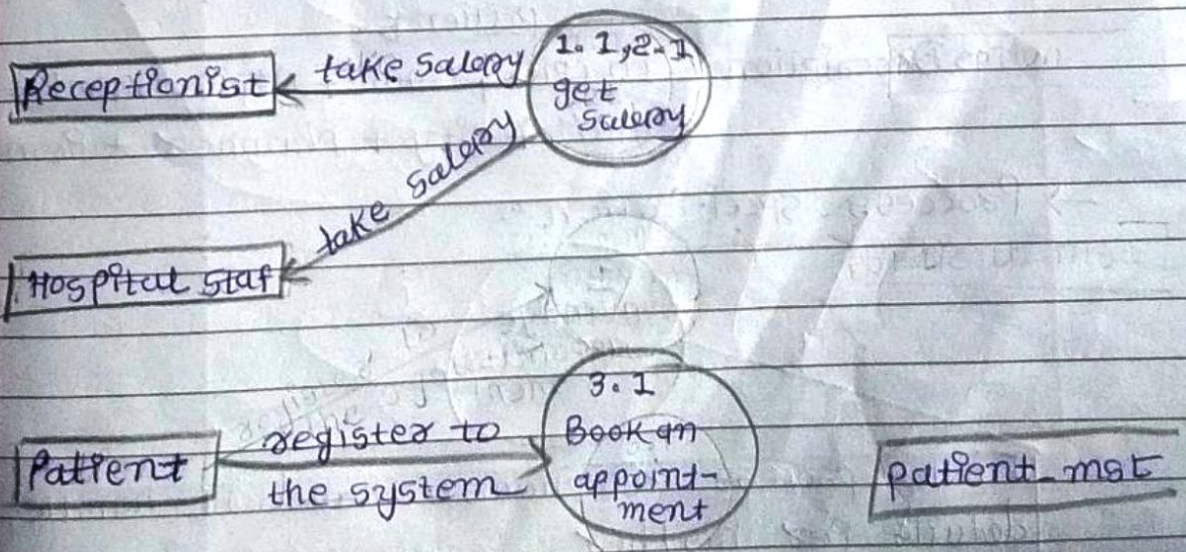
(2) Entity table :

Sr. no:	Entity Name	Level1	Level2	Level3
1	Receptionist	manage appointments	get salary	give service
2	Hospital staf	give service	get salary	-
3	Patients	Pay fee	Book an appointment	

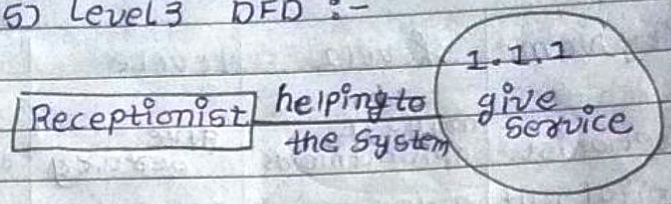
[3] Level 1 - DFD



[4] Level2 - DFD



(5) Level 3 DFD :-



→ Data Dictionary

(1) Receptionist_mst :-

- Table name = Receptionist_mst
- Alias = Receptionist master
- where & how to use = used to store about Receptionist details
- Description = ID(PK) + Name + Email + Mobile + Remark

(2) ~~St~~aff_mst :-

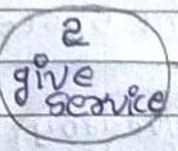
- table name = ~~Hospital~~staf_mst
- Alias = Hosptial staff master
- where & how to use = used to store details about staff member
- Description = ID (PK) + Name + Email + Mobile + ~~Remark~~ salary + Remark .

(3) Patient_mst :-

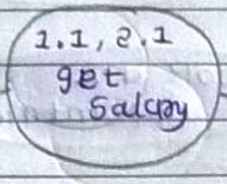
- table name - patient_mst
- Alias = patient master
- where & how to use = used to store details about patients.
- Description = ID (PK) + NAME + Appointment_number + Email + city + Number + Remark

→ Process specification :-

In this process receptionist ~~hand~~ will handle patient details per his/her requirements in the system

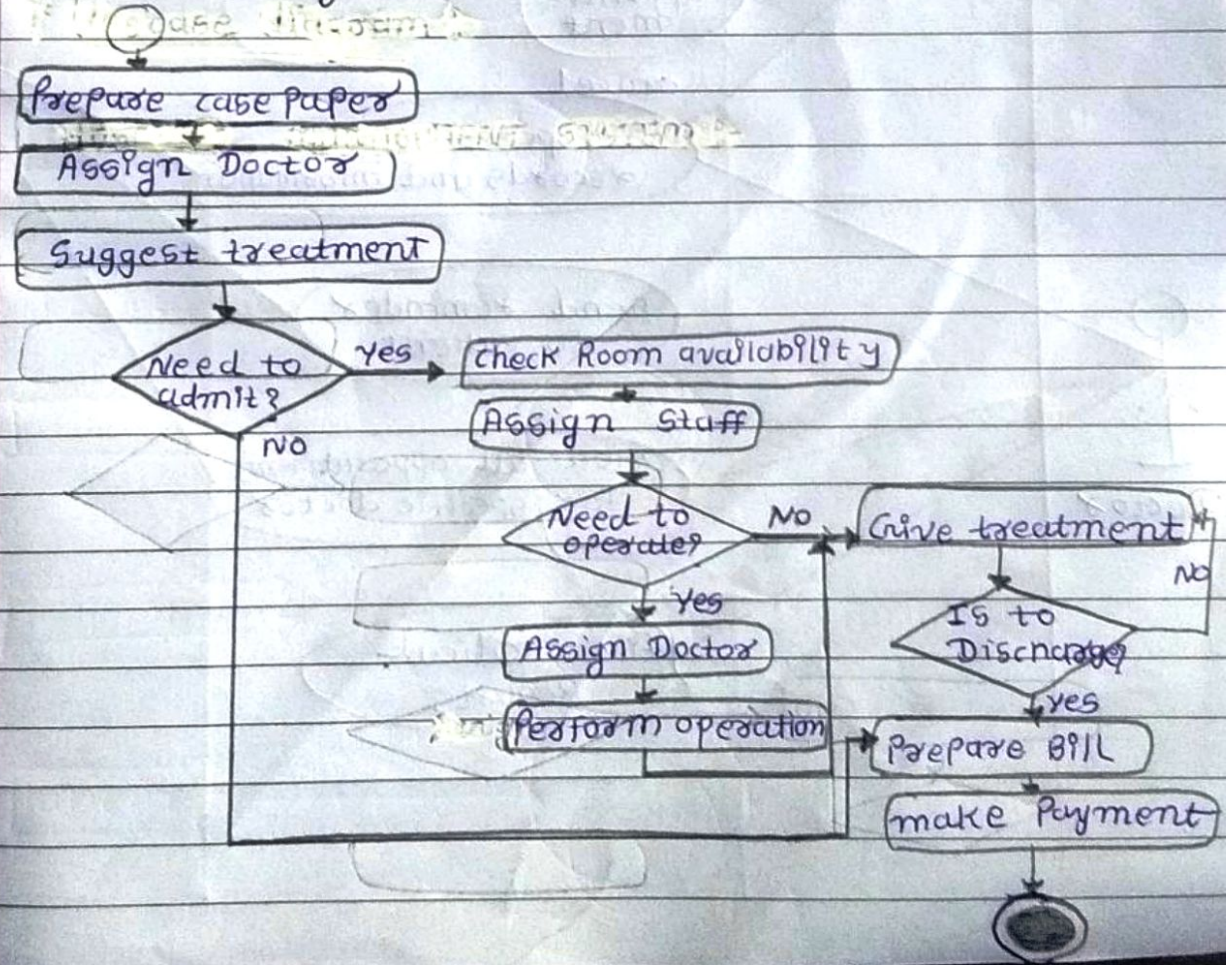


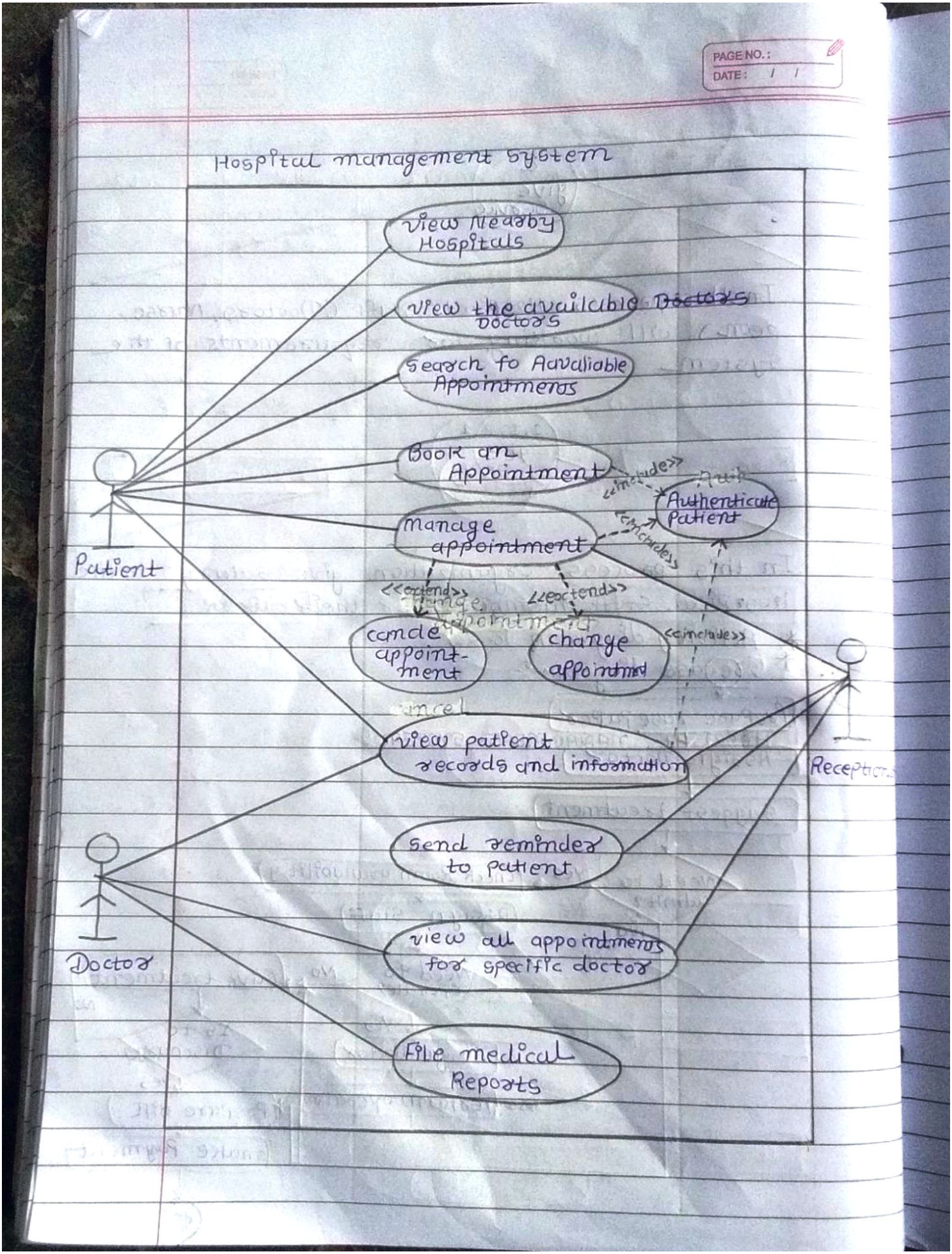
In this process Hospital Staff (Doctors, Nurse, peon) will working for requirements of the System.



In this process organization give salary to the hospital staff members for their work.

* Activity diagram fc




Hospital management system
View Nearby Hospitals
View the available Doctors
Search to Available Appointments
Book an Appointment
Manage appointment
Authenticate Patient
<<include>>
<<include>>
Patient
<<extend>>
<<extend>>
cancle appoint-ment
change appointment
<<include>>
view patient records and information
Send reminder to patient
view all appointments for specific doctor
File medical Reports
Doctor