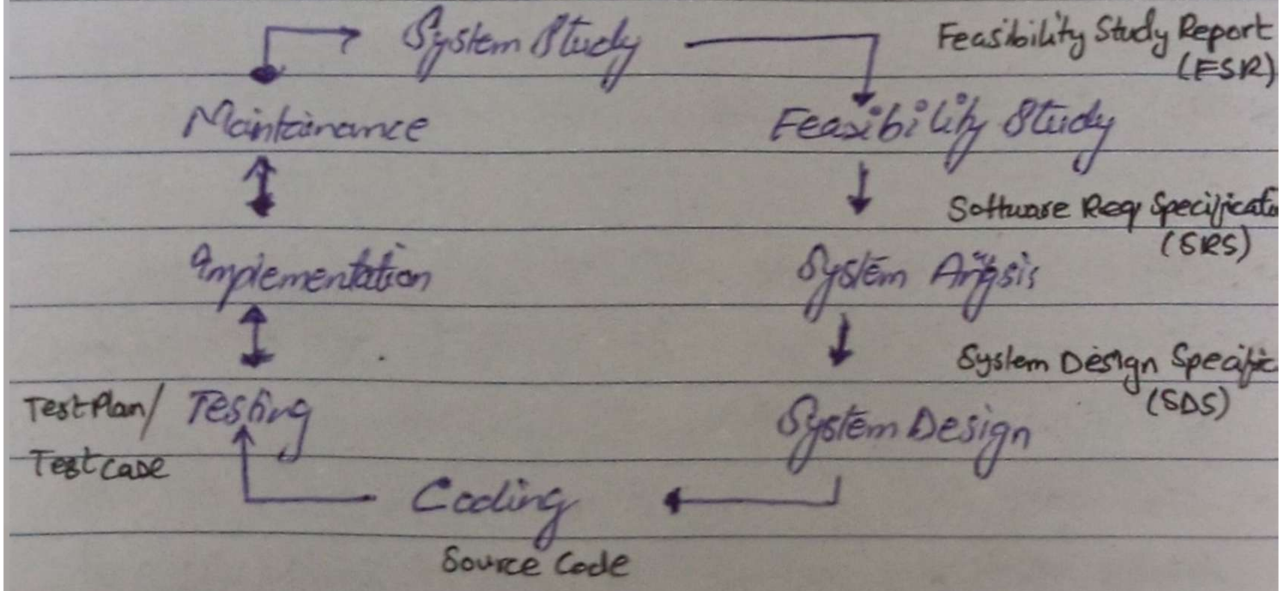
Date ____________ Day ____________

# " SYSTEM DESIGN & ANAYLISIS "

## " SOFTWARE DEVELOPMENT LIFE CYCLE "

→ SDLC is an organizational process of system develo
& maintainance

→ Following are the phases of software DLC.



i) System Study:
↳ Understanding system req. Stakeholders (clients, end-us
managers) explain their needs.
↳ If there isn't any system, then market research.
→ Identifying existing problems in current system.
↳ Clear picture of what actually the physical system is

2)

SRS → Software Req. Specification Document

Date ______ Day ______

② Feasiblity Study :

→ This study is conducted to check the financial, technical, operational viability of software.

→ Whether it is cost-effective and econimically feasible or not?

→ Costs and benifts are estimated with greater accuracy.

③ System Analysis :

→ Determination of functional & non-functional analysis.

→ Identification of possible challenges, solution and constraints.

→ Use-Cases are also defined.

→ ~~SRS is also made~~ Requirements get approval from client, market analyst or stakeholders.

→ Then a SRS is made which contains all those things that are needed to create during the entire project cycle.

→ Data Collection: for files, decision points & transactions for present system

→ Tools for System Analysis: Interviews, on-site observations and questionaire.

→ Defining the boundary of the new system (walkout pros & cons of new system)

→ Includes sub-dividing of complex process,

Date ______ Day ______

What a system should do

→ Functional Requirments : User Auth, CRUD Oper, Payment Processing, Notificati-

→ Non-Functional Requirements : Performance, Scalibility, Security, Reliablity etc

↳ How system performs it's functions.

④ System Design :

→ Involves designing the architecture of the system.

→ Includes Designing

- ↳ Database
- ↳ User Interface
- ↳ System Modules
- ↳ Data flow Diagrams

→ Design have two stages

↳ Perliminary or general Design (High Level)

↳ Structure or Detailed design (Abstraction level or low level)

1. Flow chart

2. Data Flow Diagrams

3. Data Dictionary → Database design

4. Structured English ↳ use of English with syntax of structured Programing Easy to communicate design to non-technical persons.

5. Decision Tables

Date ______ Day ______

(5) Coding:

→ The actual code for the development of software is written based on Design Document.

→ Framework req. to meet design specifications

→ Must be in modular nature.

Development → Standard Coding → Scalable Code → Version Control → Code Review

(6) Testing:

→ Testing the developed software to identify bugs & errors.

→ Following are the various types of testing.

(1) Unit Testing:

It involves testing individual units or components of the software in isolation.

Like testing user registration, login etc.

(2) Integration Testing:

Focuses on the interaction b/w diff units or modules of the software.

All the components are tested as a group to ensure that they are working acc to requirements.

Page No.

(3) System Testing:

Determines if the entire system satisfies the basic req & performs as intended.

(a) Smoke Testing: Check the happy flow after new feature is added.

Date ______ Day ______

(3) Alpha Testing: ↗ sys should satisfy basic req of end-users.

Type of User Acceptance testing (UAT) that occurs in development env.

It is conducted by internal teams (eg: developers, & testers (SQA)).

(4) Beta Testing:

It involves realising the software to a limited no. of external users who use the sys in real-world env.

The purpose is to gather feedback & identify bugs.

(7) Implementation: (Deployment)

Software is deployed to the production env. where end-users can start ~~testing it~~ using it.

Parallel Run: Both old & new sys runs ||ly

Pilot Run: New modules are installed in parts.

(8) Mantainance:

Eliminate errors in system during it's working life

Adding new features, as req changes & ensures that sys runs smoothly over time.

Page No.

## "ENVIORINMENT IN SD"

① Development:
↳ First env. in SD which acts as a workspace for developers. Like VS Code (IDE)
↳ No Client Data involved.

② Testing:
↳ Used by Quality Assurance Engineers.
↳ this env. is created by allocating storage, computing & other resource need for testing.
↳ No Client Data involved.

③ Staging Enviornment:
↳ Used by QA and/or clients for UAT
↳ Limited Production data
↳ You reveal the software to immediate owner but not the users

④ Pre-Production:
↳ It is the copy of production env.
↳ It allows you to test & catch bugs in your code before pushing it to production.

⑤ Production:
↳ When an end-user uses the software, it's running on production env.
↳ Tests can be carried out while production and new features can be introduced.
↳ Full Production data

⑥ Mirror:
↳ Replica of production env
↳ Developers & QA performs bug fixes or testing that would be risky in production.

## "OBJECT ORIENTED ANALYSIS & DESIGN"

→ Ability to thoroughly represent complex relationships
→ OOAD systems development life cycle:

① Analysis Phase → identifying entities/objects & their relationship
- Model of the real world application is developed showing its imp properties
- Model specifies the functional behaviour of the system independent of implementation details.
↳ Defining the functions & attributes (User: name, id, getAuthenticated)

② Design Phase
- Analysis is refined & adapted to make
- ▹ System Design: Concerned with overall sys architecture
- ▹ Object Design: Implementation details are added to sys design.

Date Day

(5) Implementation Phase

- Design is implemented thru programing lang or OBMS.

## WEEK # 02

- INHERITANCE:

```
class Add {
    int my;
    int by;
    void setmyby(int xy, int hy){
        my = xy;
        by = hy;
    }
}
class sub extends Add {
    // more functionality can be added here if needed.
}

public class inheritance {
    public static void main (String[], args) {

        Sub subtraction = new sub();
        subtraction.setmyby(10, 20);
```

Page No.

Date Day

```
        System.out.println("my - by:" + (subtraction.my - subt.by));
    }
}
```

- Constructor:

```
public class Main {
    int a;
    public Main() {
        a = 3*3;
    }
    public static void main(String[] args){
        Main myObj = new Main();
        System.out.println(myObj.a);
    }
}
```

- Abstract:

```
abstract class Animal {                        ↗ Abstract Method
    public abstract void animalSound();
    public void sleep() {
        System.out.println("Zzzz");
    }
}
```

Page No.

Date ______ Day ______

```
// Sub class inherit from Animal
class Cow extends Animal {
    public void animalSound () {
        System.out.println(" Mooo ");
    }
}
```

"PROCESS MODELS"

① WATERFALL MODEL :

→ It's a basic SDLC. It's very simple but idealistic.
→ Useful when the project requirements are well-defined and project goals are clear.
→ Used for large scale projects with long timeline.
→ Little room for error.
→ Stakeholders need to have highlevel of confidence in the outcome

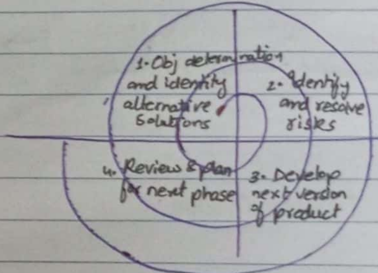
$$\begin{tikzcd}\boxed{\text{Requirements}} \arrow[r] & \boxed{\text{Design}} \arrow[r] & \boxed{\text{Development}} \arrow[r] & \boxed{\text{Testing}} \arrow[r] & \boxed{\text{Deploy}} \arrow[r] & \boxed{\text{Maintenance}}\end{tikzcd}$$

② EVOLUTIONARY MODELS :
→ These are iterative type models.
▷ SPIRAL MODEL :
→ Combination of waterfall and iterative model.
→ Provides support for Risk handling.

Page No.

Date ______ Day ______

→ Each loop of spiral is called a phase of the software development process.



1. Obj determination and identify alternative solutions
2. Identify and resolve risks
3. Develop next version of product
4. Review & plan for next phase

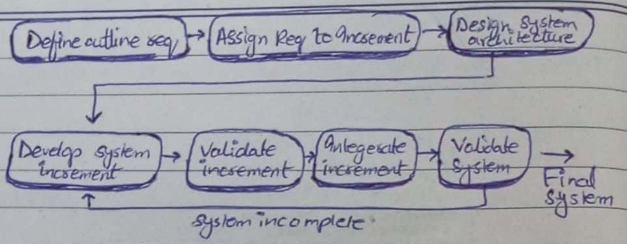
→ It is complex, expensive, Difficulty in time management.

▷ INCREMENTAL MODEL :
→ First, a simple working system implementing only a few basic features is built and then that is delivered to the customer.
→ Then ~~after~~ many sucessive iteration/versions are implemented to reach desired design
→ Useful when,
↳ req are known up-front
↳ projects have lengthy development shedule
↳ Total cost is not lower (bcz of continuous iteration) cost increases

Page No.

→ Iterations are steps in the model & increments are growth, of the product.

Date ______ Day ______



" RATIONAL UNIFIED PROCESS MODEL "

→ (RUP) is a framework for software eng process.
→ It is SDLC for Object oriented models.
→ UML driven iterative process model.

▸ Phases of RUP:

① Inception:
↳ Communication & Planning
↳ Identifies scope of project using a use-case model allowing manager to estimate costs and time requir.
↳ Customer's requirements are identified
↳ Project Plan, goals, risks, use-case model and description are made.
↳ Project is checked against mileston criteria, if (not pass criteria)
Reject or redisign

Page No.

Date ______ Day ______

② Elaboration:
↳ Planning and modeling
↳ A detailed evaluation & development plan is carried out which reduces risks.
↳ Revise or redefine usecase model (approx 80%), and risks
↳ Again milestone checking

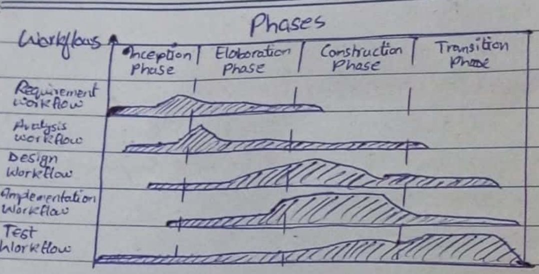
③ Construction:
↳ The project is developed & completed
↳ System or source code is created
↳ Testing
↳ Coding takes place

④ Transition:
↳ Final project released to the public
↳ Transit the project from development into production.
↳ Update project documentation
↳ Beta Testing
↳ Defects are remove upon public feedback

⑤ Production:
↳ Keeps system useful/productive after deployment to customer.

Page No.

Date ______ Day ______



▷ AGILE MODEL:

↳ The main aim of agile model is to facilitate quick project completion.

↳ It is a project mangement framework that breaks project down into several dynamic phases, known as sprints

↳ This framework is an iterative methodology

↳ After every sprint, team reflects and look back to see if there was any thing that could be improved so they can adjust their strategy for next sprint

→ Following are 4 pillars of Agile

1. Individuals over processes & tools
2. Working software over Comprehensive documentation
3. Customer collab over contract negotiation
4. Responding to change over following a plan.

Page No.

Date ______ Day ______

→ Following are 12 principles

1. Customer Satisfaction
2. Early & Continuous Delivery
3. Embrace Change
4. Frequent Delivery
5. Collab of business & developers
6. Motivated individuals
7. Face-to-face conversat
8. Technical excellence
9. Simplicity
10. Self-organized Teams
11. Functional Products
12. Regulation, Reflection, and adjustment.

• SCRUM:

→ It is one of the agile methodologies.

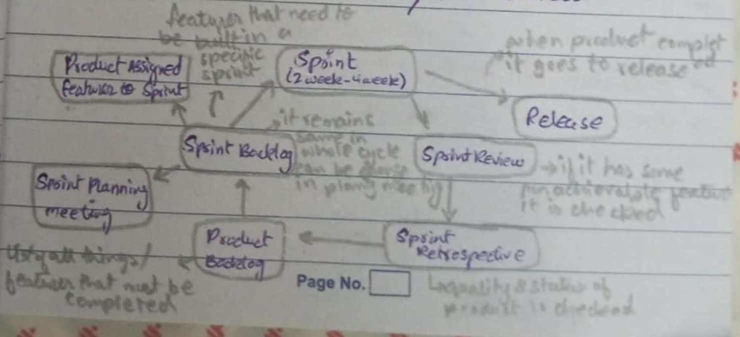
→ Subset of Agile. It is a lightweight framework.

→ Used primarily for software dev projects with the goal of delivering new software every 2-4 week

→ The team is led by Scrum Master.

→ Sprint Planning: This event kicks off the sprint. Outlines what can be delivered in sprint

→ Sprint Retrospective: This reccuring meeting ~~as~~ acts as a sprint review



features that need to be built in a specific sprint
Product Assigned features to Sprint
Sprint (2 week-4 week)
when product completed it goes to release
Release
it remains same in whole cycle can be discuss in planing meeting
Sprint Backlog
Sprint Review
if it has some [illegible] feature it is checked
Sprint Planning meeting
Product Backlog
Sprint Retrospective
list of all things/ features that must be completed
Legality & status of product is checked

Page No.

# "DOMAIN MODEL"

- User Stories:

→ It is a simple, concise description of a software feature or functionality from perspective of end-users.

Key Components:

1. User Role: Who is using the system?
2. Goal: What does the user want to achieve?
3. Benefit: Why does the users need this.

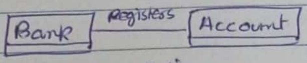
"As a [user role], I want to [goal], so that [benefit]."

Ex: As an admin, I want to assign trainers to members, so that each member has a trainer to guide him.

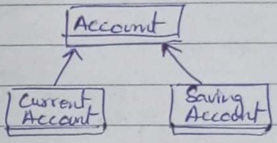
- Domain Model:

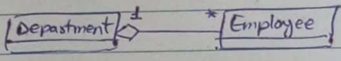
→ Captures the most imp types of obj in a sys.
→ Describing objects, classes, interfaces, package or a subsystem and relation b/w them.
→ Includes attributes
→ Doesnot include operations/ func.

- Relationships:

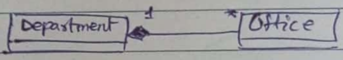


▲ Association

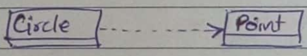




▲ Aggregation

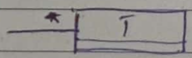


▲ Composition

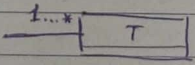


▲ Dependency

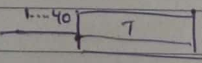
- Multiplicity:



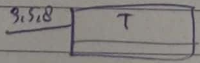
zero or more



one or more

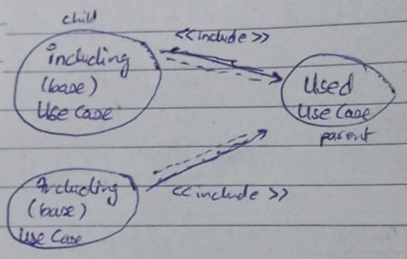


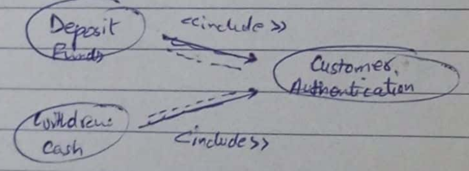
one to 40



exactly 3,5,8 or 8

## "USE CASE DIAGRAMS"





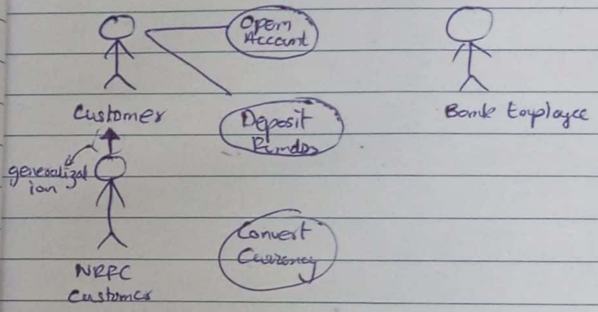
Deposit & withdraw cash use cases includes Customer Auth use case.

<<extend>> is used to define optional functionality.
parent se child par arrow jayega. Arrow head on ~~child~~ parent

<<uses>> One use case uses another

- Relationships in Use Case Diagram:

① Generalization of an Actor:


Open Account
Customer
Deposit Funds
Bank Employee
generalization
NRFC Customer
Convert Currency