Object Oriented System

Java is a programming language and a platform
Java is high level, robust, object oriented and secured language.
Developed by Sun microsystem in 1995

Helloworld in Java

```
class Simple {
    public static void main (String args[]) {
        System.out.println ("Hello word");
    }
}
```

Application
- Desktop application – Developed using (AWT, Swing)
- web application – " " (Servlet, JSP, JSF, Spring)
- mobile application (Android)
- games, robotics.
- Enterprise application

features - Simple
* Object Oriented (OOPS)
Platform independent
Robust
Secured
* Multithread (
Dynamic

Java code compilation - At compile time, Java file is converted to bytecode.

Javacode → compiler → Bytecode.

Runtime steps

classfile → classloader → Bytecode verified → Interpreter → Runtime

classloader: loads classfiles.
Bytecode verified: check code
Runtime: hardware

JVM - Java virtual machine is abstract machine that provide runtime environment in which Java byte code can be executed.

JRE - Java runtime Environment, it used to provide run time environment. It is implementation of JVM. It contain libraries + other file that JVM use at runtime.

JDK - Java development kit - is a software development environment which is used to develop Java applications.

Java variable -

Variable is container that contains some value.

Type - local variable (declared inside method)
instance variable (declared inside class but outside method)
static variable (static variable have single copy)

Datatype - Primitive - int, char, long, float, double, byte.
Non-primitive - class, interface, Array.

* char take 2byte.
* byte take 1byte.

operator - Unary, Assignment
Arithmetic, Bitwise
logical, relational.
shift
Ternary

Java oops concept

Object - means a real world entity such as chair table.

class - class is user defined data type that define how object look like.

oops concept - inheritance, polymorphism, Abstraction, Encapsulation.

Class	Object
Class is collection of similar object.	object is instance of class.
No memory is allocated for class.	Each object has memory.
class is template declared once	object is real. object can be created many times.

Advantage of OOPs over Procedural programming

1) OOPs make development and maintainance easier. compared to PP.
2) OOPs provide data hiding
3) Adding new data is easy. in OOPs.
4) Overloading is possible.
5) It is more real and create complex system.

PP — C, Fortran

OOP — C++, Java

Class - It is template or blueprint of object.

A class contain - field
method
constructor
Nested class, Interface
Block

Advantage of class - code reusability
code optimisation.

Ex- class Rectangle

```
{ int l;   — field
  int w;
  void insert (int ll, int wl)  — method
    { l = ll;
      w = wl;
    }
}.
```

Inheritance - It is mechanism by which one class is allowed to inherit feature of another class

super class - Parent class

sub class - child class

types of inheritance -

single inheritance, —— A ↓ B

multilevel —— A ↓ B → C

hierarchical —— A with B C D

Multiple inheritance - Java do not support but can be achieved using interface —→ A B ↓ ↓ C

Hybrid - (through interface) A, B, C, D

```
class one {
    public void print()
    {
        su —
    }
}
class two extends one {
    public void print()
    {
        ——
    }
}
```

two inherits one.

Java polymorphism - Polymorphism means many forms

Method overloading - multiple function with same name but different parameter.

Ex
```
void abc()
{
}
```
==

```
void abc(int a)
{
}
```
==

Method overriding - when ~~to~~ sub class have same function name as parent class.

Java abstraction - Process of hiding the implementation detail and only showing necessary functionality achieved through class (abstract)

Abstract class - A class declared with abstract keyword is abstract class it have abstract or non abstract method it needs to extended and can not be instantiated.

Java Encapsulation - wrapping up of data under a single unit it is mechanism that bind data achieved through class.

Interface in Java - An interface is blueprint of class that contain static constant and abstract methods. It is used to achieve achieve abstraction. Only has public specifier.

```
interface player
{
    int id = 10;
    int move();
}
```

Java Package - group of similar type of class, interface and sub package.

Some build in package - swing, util, sql.

use - Preventing name conflict
making search/locate use of class/interface easy.
Package consider as data encapsulation.

Type
- User defined
- In built Package. (Java.lang, Java.io, Java.util)

Ex - import java.util.*

Exception handling - mechanism to handle to runtime error.

reason for exception - invalid input
code error
unavailable file.

Types of Exception

checked Exception - They are checked at compile time.

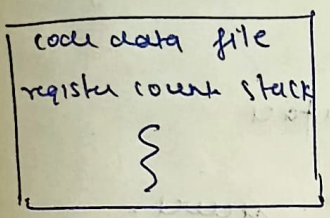
unchecked Exception - compiler will not check them.

user defined exception -

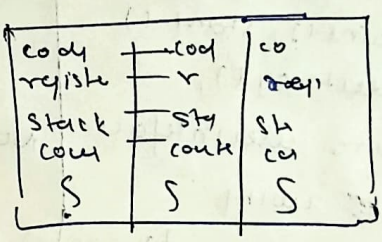
try, catch, throw, throws, finally.

Multithreading - in Java feature allows concurrent execution of two or more part of a program for maximum utilization.

single thread



multi thread.



Java string handling - String in Java are object that are backed by char array.

Example - String str= "ABC";

or String s = new String ("ABC");

taking input in Java

```
import Java.util.Scanner;
class Main {
    public static void main (String args[])
    { Scanner sc = new Scaner (System.in);
String ⟶ String str= sc.next();
Int ⟶ int a = sc.nextInt();
float ⟶ float v = sc.nextFloat();
}
```

Java Event handling - handle any type of event like mouse in, mouse out, key up, key down.

Java.awt.event provide many event class and listner for event handling.

Java applet - special type of program embedded to web page to generate dynamic content work at client side.

Java Servlet - technology to create web server side web application and generate dynamic web content.

Applet	Servlet
• executed on client side	server side.
lifecycle - init(), stop(), paint(), start() destroy(),	lifecycle - init(), service(), destroy()
Applet are use interface class like swing	No interface
Applet prone to risk	more secure.
more network bandwidth.	less network bandwidth.

Awt - Abstract window toolkit - Used to develop GUI - Java.awt contain class for Textfield, label, text-area, button etc.

JSP - Java server page - create web application like servlet but has more feature like expression language and JSTL.

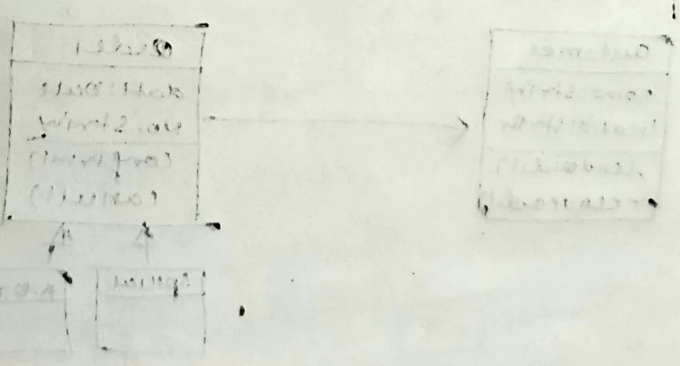
~~Disadvantage~~ Adv - Addition feature
Easy to maintain
fast development
less code than servlet

Java beans - Java bean ~~is a Java class~~ are classes that encapsulates many objects into single object (bean) It follow following convention

- must implement serializable
- public no-arg constructor
- all properties must be private with public getter and setter.

Java swing - Used to create GUI built on top of awt. Unlike awt swing has platform independent and lightweight component.

Swing	Awt
platform independent	dependent
lightweigh component	heavy weight component
powerful and more component	less component
follow MVC	do not follow MVC



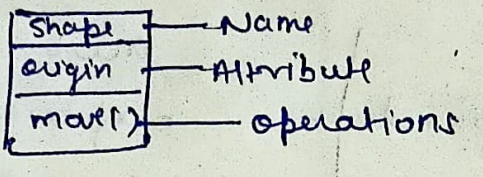
Basic structural modelling -

structural modelling captures the static feature of system it consist of -

class diagram
object diagram
Deployment diagram
Package diagram
composite structure diagram
component diagram.

structural model represent framework for system where all component are shown

class - description of object.



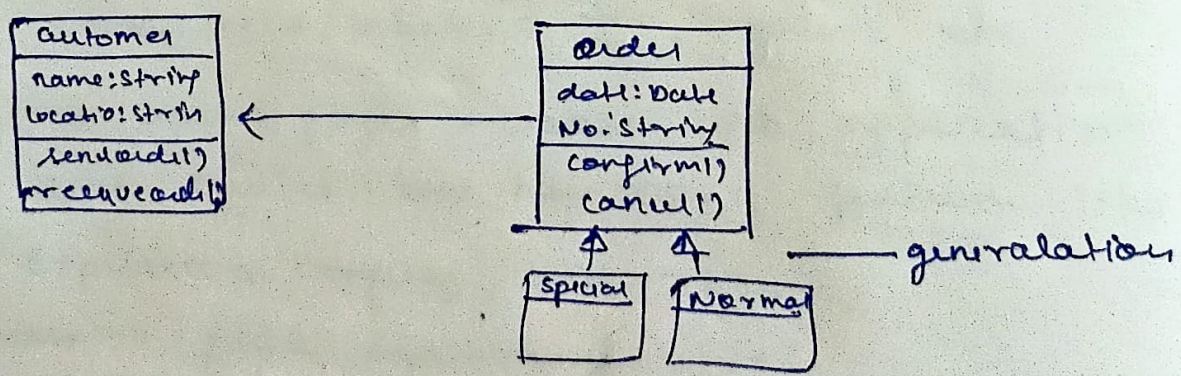
class diagram - A class diagram is set of classes interfaces and collaboration and their relationships. It show static view of system.

Purpose

1) show static structure of a system
2) Provide basic notation for other diagram in UML
3) Helpful for developers & team.
4) Business analyst can use class diagram to model system.

class diagram made of - classes and their relation.

class diagram for order -

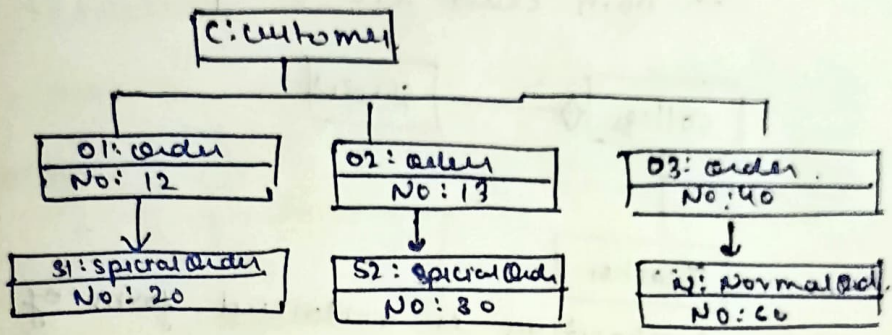


Object diagram - derived from class diagram and depends on class diagram.

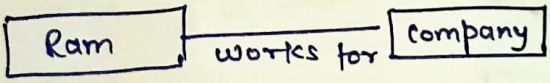
Object diagram represent instance of class diagram so also represent static view of system at particular moment.

Purpose -
forward and reverse engineering
object relationship of system
static view
understand object behavior and relationship

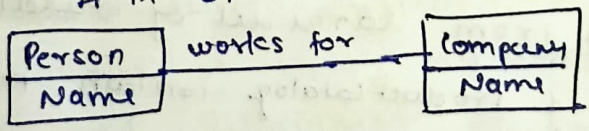
object diagram for order management.



Link and association -
In object modelling, links provide a relationship between object. Represent using ——

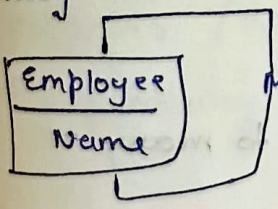


Association - It is the relationship among classes.
It is bidirectional.



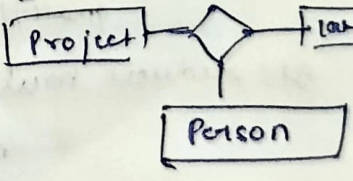
degree of association - No: of classes connected by association.

a) Unary



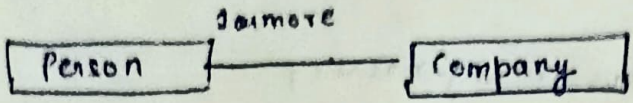
Binary

Ternary -



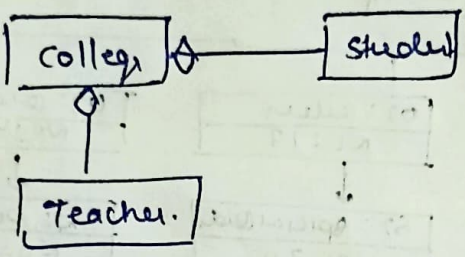
Multiplicity - multiplicity denotes the cardinality of association three common type are -

- one to one
- one to many
- Many to Many

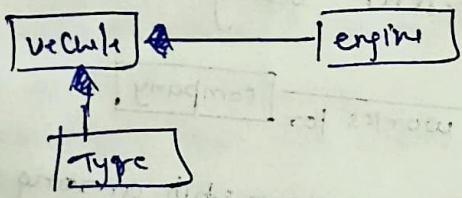


Aggregation - A plain association between two class having property -

- ~~Unidirectional~~ Unidirectional
- both class are independent.

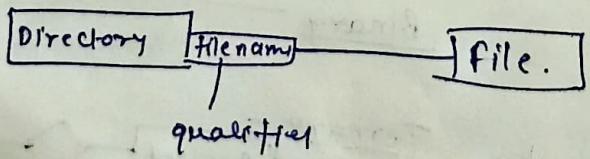


Composition - composition is restricted form of aggregation two entity are highly dependent on each other.
composed object do not have its own entity.



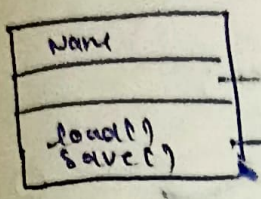
Qualified Association - It has a qualifier that is used to select object from large set of related object. Ex - if Productcatalog contain many Product and each is can be selected by itemID.

Ex



It reduces multiplicity of one to many or many to many.

Interface - collection of operation to specify service.
specify a behaviour of an element.
described by abstract operation
use to model system.

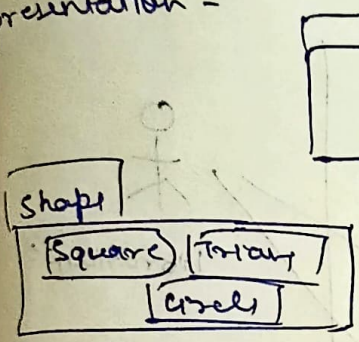


- representation

- participate in generalization, association, dependency realization.

Package - General purpose mechanism for organising modelling elements into groups.
- organise element in large chunks

representation -



well structured packages are loosely coupled and very cohesive.

Use case modelling -

A use-case model is a model how different user interact with system to solve a problem.
describe goal of user, interaction b/w user and system and behaviour of system.

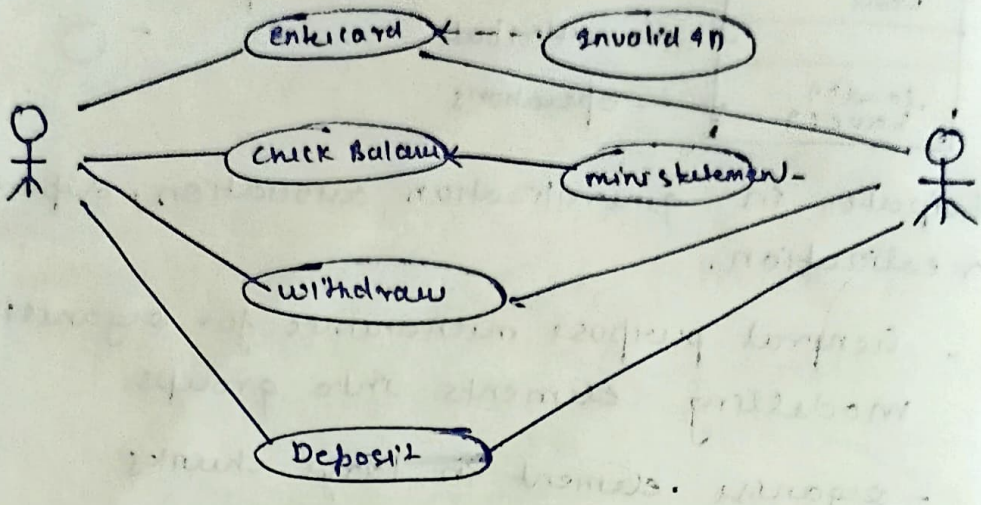
Use diagram contain
- use case (how actor use system) - ⬭
- Actors (People involved with system) - 🧍
- Dependency, generalization, association / (relationship)

→

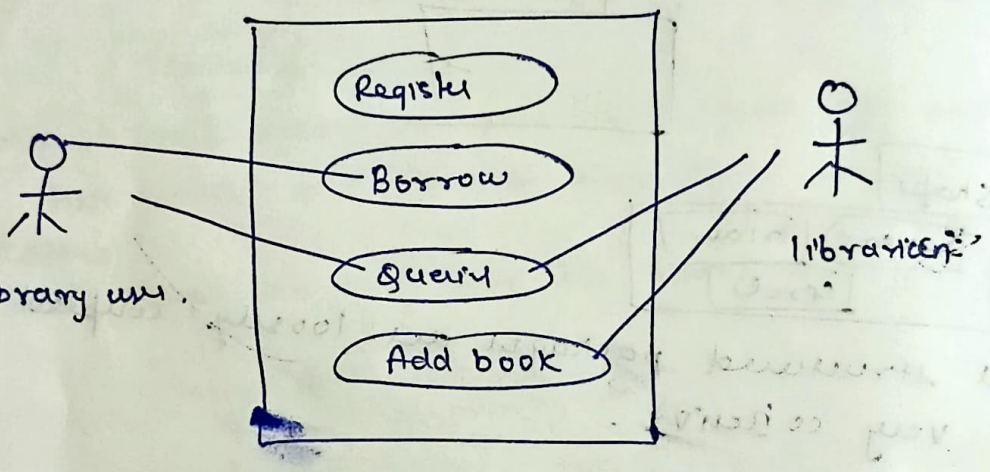
Use of Use case diagram

- To model context of system
- To model requirement of system.

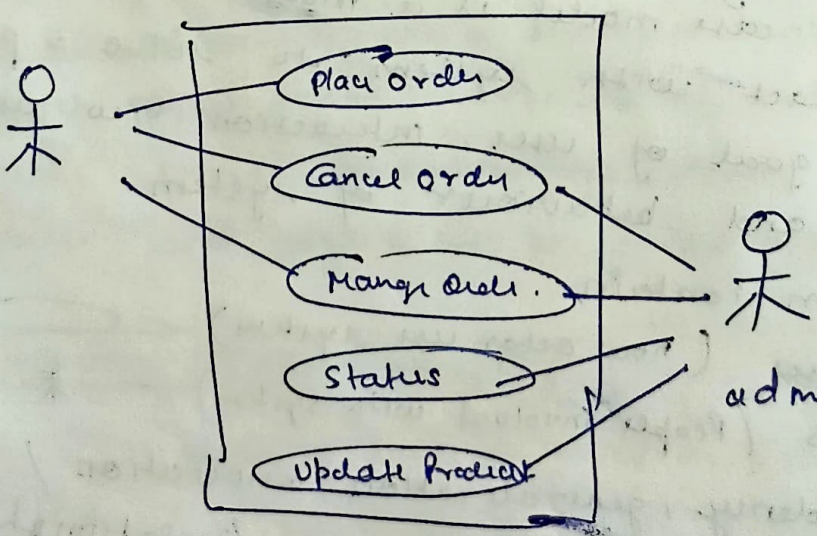
Use case for ATM



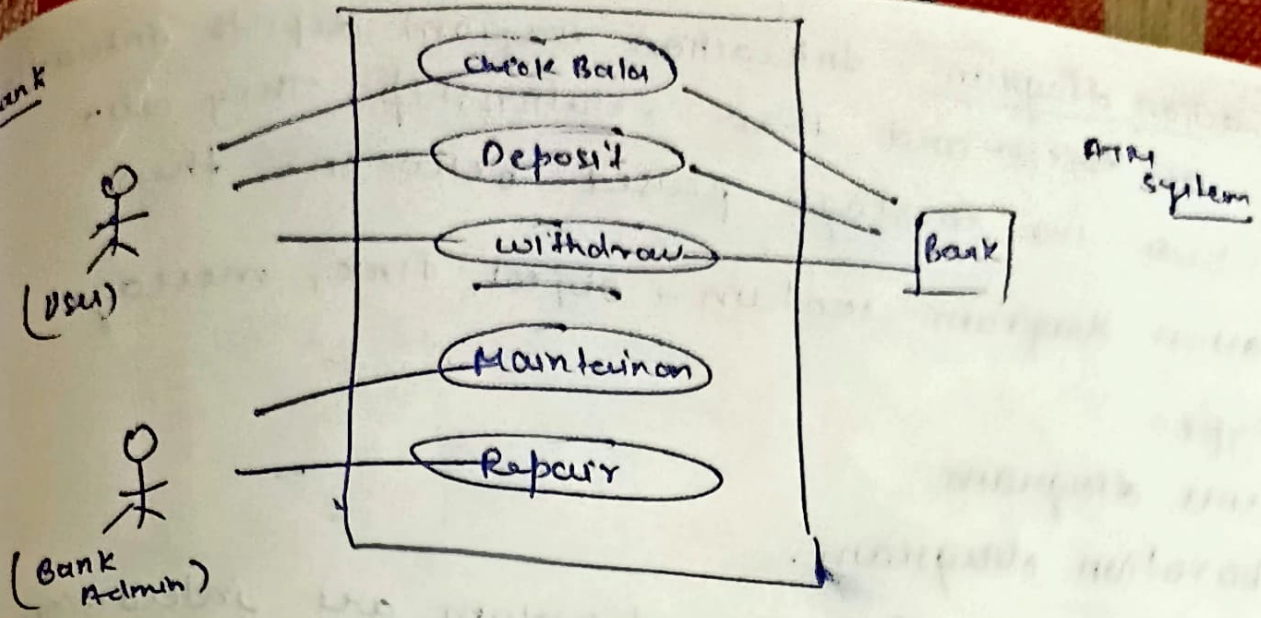
Use case for library



Online store



Bank


Check Balan
Deposit
withdraw
Maintenance
Repair
Bank
(User)
(Bank Admin)
ATM System

Interaction diagram - Interaction diagram depicts interaction of objects and their relationship. They also include the message passed between them.

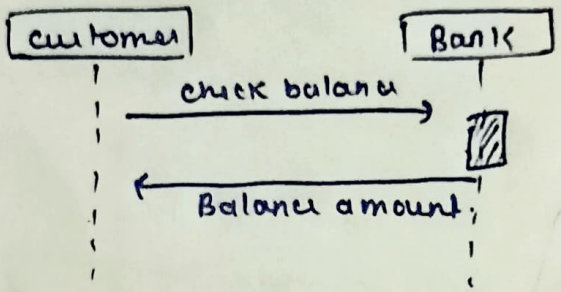
Interaction diagram contain - object, link, message.

Two type -
Sequence diagram
collaboration diagram.

Sequence diagram - Sequence diagram are interaction diagram that illustrate the ordering of message according to time.

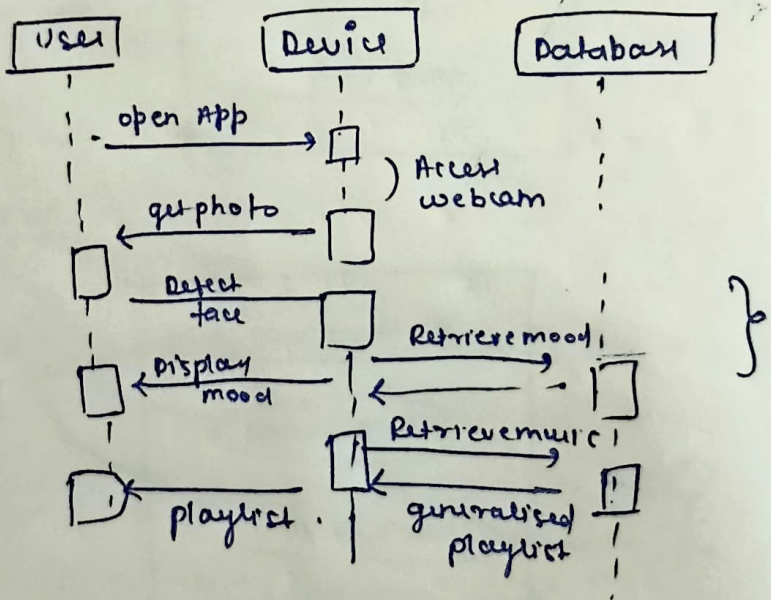
Components - Actors, lifelines, messages.

Ex -



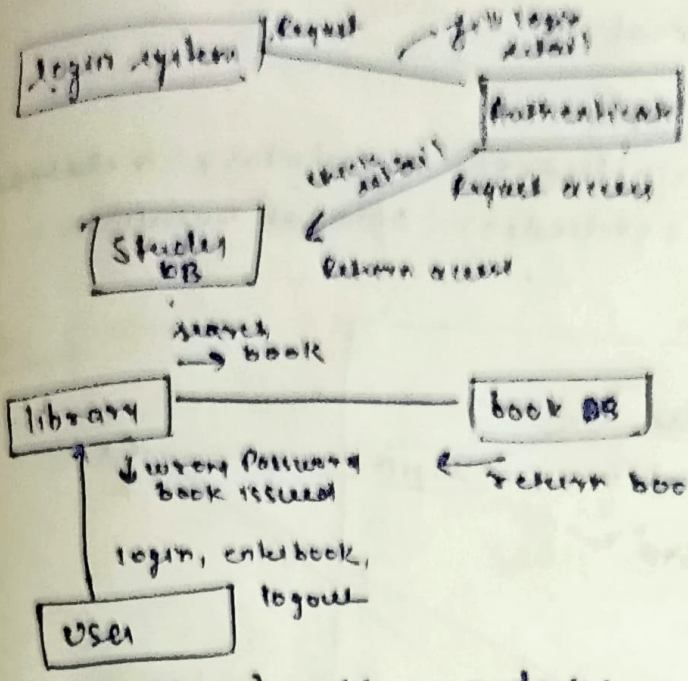
Sequence diagram.

Sequence diagram for emotion based music player.



collaboration diagram - Collaboration diagram illustrate structure of object that send and receive message. It is used to understand object architecture within system rather than flow of message.

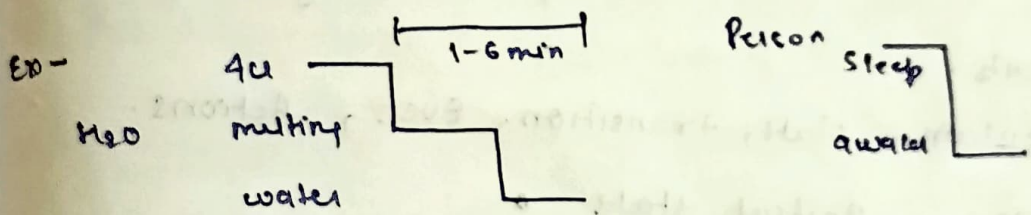
relation - Actors, object, link, message



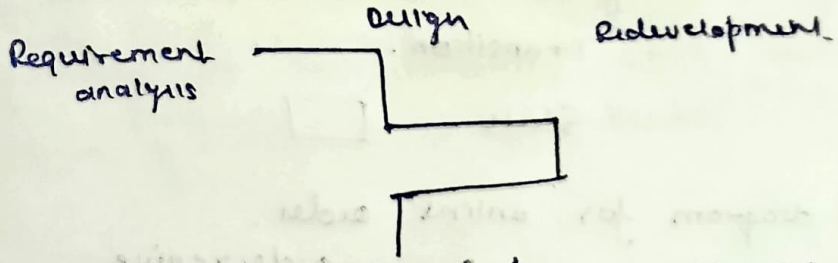
Use - model flow by time ordering
model flow by organization.

Timing diagram - Part of interaction diagram it consist of graph, waveform that depicts state of lifeline at specific point of time.

timing diagram depict how object change from one form to other.



for software development



Use case realization - It represents how a use case will be implemented in term of collobarating objects.

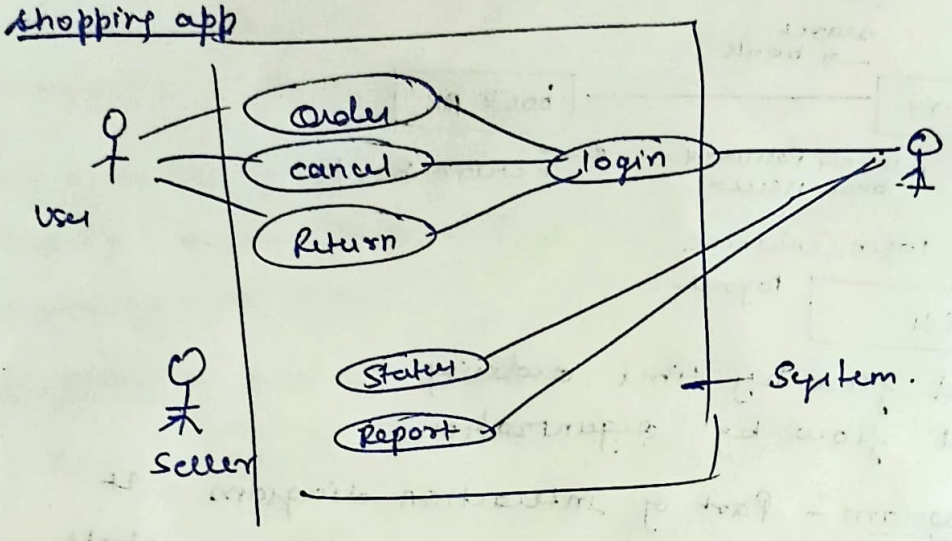
Use case is represented using ⬭

—— connector

——→ generalization

- - - -→ stereotype.

Relationship - <<include>> mandatory relation.
<<exclude>> optional relation



State chart diagram - A state chart diagram show a state machine that depicts control flow of an object from one state another. It potray sequence of states which an object undergoes due to events.

It contain - state, transition, even, Actions.

Notation - Initial state •

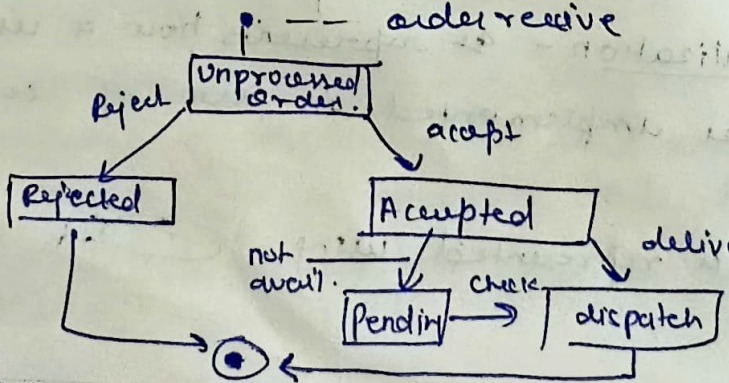
final state ⊙

Decision box - ◇

Transition. ——

State - ▭

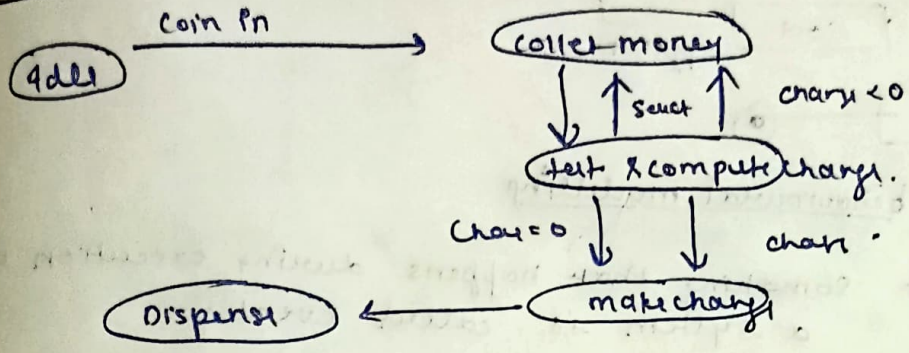
state diagram for online order.



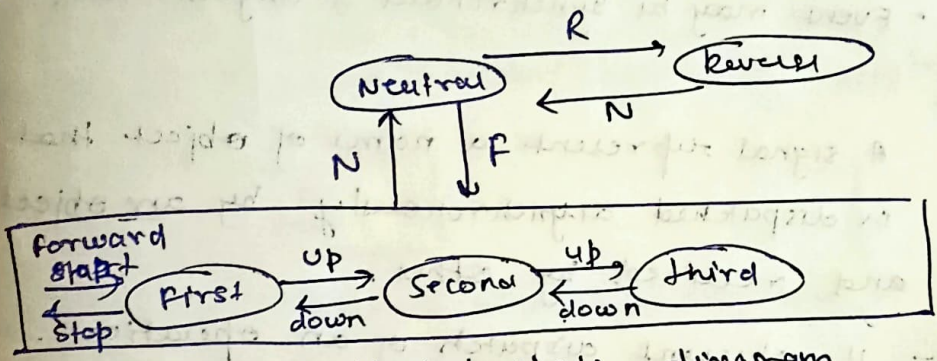
Nested state diagram - It is used to model complex system as regular state diagram is inadequate. It is advanced modelling concept.

complex system can be restructured by implementing sub diagrams by expanding state.

Vending machine



Nested state car transmission.



Nested state diagram.

Activity diagram - An activity diagram is behavioural diagram i.e depicts behaviour of system. It portrays control from a start point to finish point showing various decision path that exist while activity is being executed.

Notations

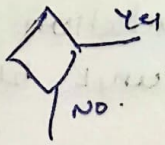
Initial state - •

final state - ◉

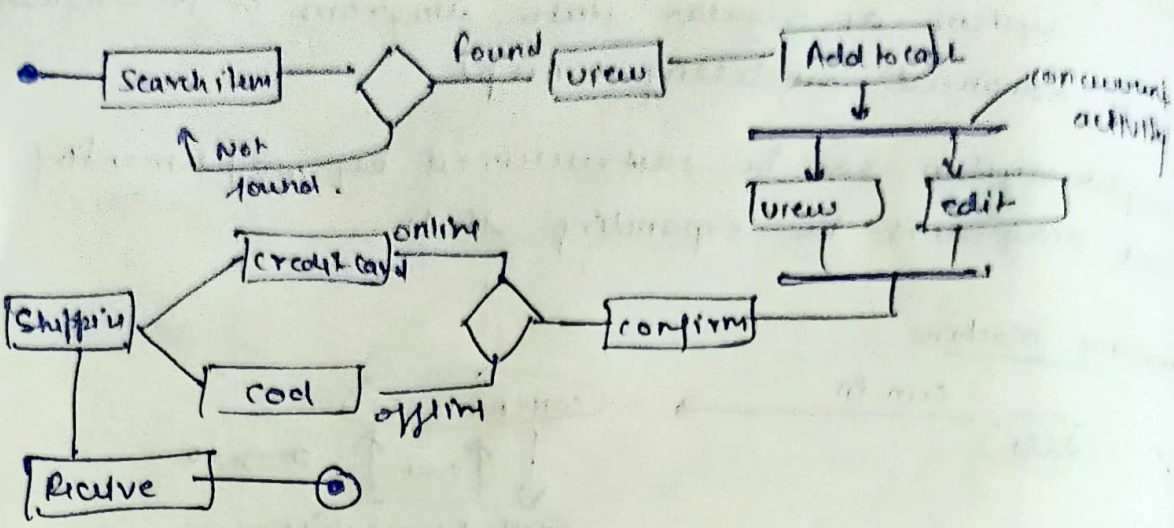
Activity - ▭

flow control - ⟶

decision box, condition Branch.



Activity diagram -



Advanced behavioural modelling

Events - Something that happens during execution of a system is called event.

- It may include signal, calls, change in state.
- Event may be synchronous & asynchronous.

4 kind

Signal - A signal represent a name of object that is dispatched asynchronously by one object. and received by other.

call events - it represent dispatch of an operation. this trigger a state transition. It is synchronous.

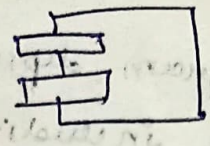
Time and change event - It represent passage of time.

Sending and receiving event - one object sends the signals or invoke operation and other, to which signal is directed.

Architectural modelling -

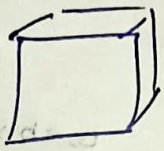
Component - component is replacable and executable piece of system whose implementation details are hidden. It is logical unit block of system.

representation



Deployment - A node is a physical element that exists at runtime and represents a computational resource generally having atleast some memory and processing capability.

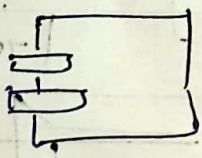
Representation :-



Component diagram - It is used to break large OOS into small components to make it more manageable.

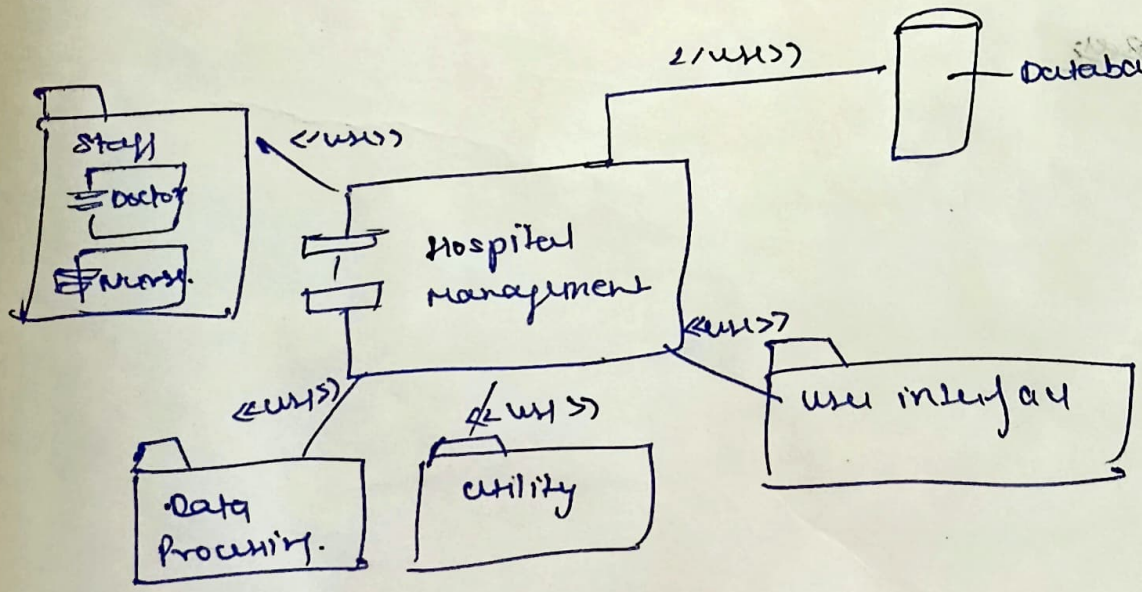
A component diagram describes the organisation of physical components in system.

Component - Physical building block of system.



Interface - An interface describes a group of operation created by component.

dependencies - Represent relations.



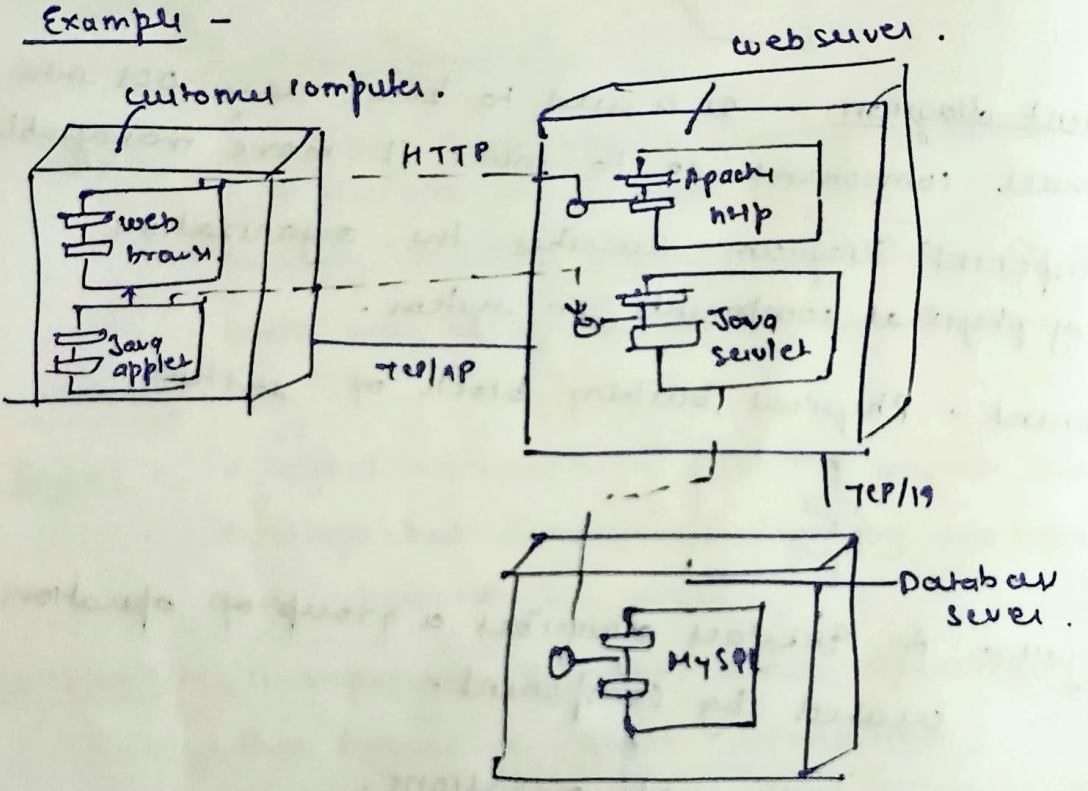
Deployment diagram-

Deployment diagram depicts the physical resources in a system including: node, component, connection. Portrays deployment view of system.

Node - A node is a physical resource that executes code component.

Association - connection.

Example -



Components: