Date .....

Unit - 1

1) → OOAD
2) Functional Programming
3) Structural Programming
4) Properties of OOAD a) object
5) Diff b/w Structural & OOAD programming b) class
c) Inheritance
6) Advantages of OOAD d) Polymorphism
7) Characteristics of OOAD e) Information Hiding.
f) Association
8) SDLC g) Aggregation.
9) SDLC Models. h) Composition
(a) Waterfall (b) Iterative (c) spiral (d) RUP.
Imp. Imp

Unit-2

1) Use case Modeling
2) Use Case Diagrams
3) Components of UCD
4) Types of Actors
5) Type of Relationship b/w UC
6) Steps of to draw UCD.
7) Examples of UCD.
8) Requirement Engineering

Unit-3.

Imp 1) Sequence Diagrams
2) Notation for SD
Imp 3) Collaboration Diagram
4) Collaboration D V/s SD
5) Eg of SD & CD.
6) UP Artifacts

Unit-4

Imp 1) Domain Modelling
2) Steps To create Domain

Object Oriented Analysis & Design

Book - Object oriented Modelling & Design - James Rambough

Unit - 1

Object oriented analysis & design is a technique for understanding the problem to make & represent the requirements of a software system in terms of problem domain objects and interaction between theese objects.

In earlier times we used to have functional and structured programming.

(1) Functional Programming: The Analyst maps the problem domain into functions and sub functions which are difficult to construct & are highly volatile.

(2) Structured Programming: Analyst maps the problem domain into different process & flow of data between those process.

(3) OOAD: It decomposes problem domain in terms of interactive objects.

Properties of OOAD-

* Object - The Things with which we can interact through a well defined interface or object is an entity, which has a state & a set of well defined operations that operate on that state.

Date

class - Objects can be classified into classes based on their similarity & dissimilarity.
eg car, truck, cycle can be clubed in vehcile class. Debit & Credit account can be clubed into account class.

* Inheritance: The inheritance permits reusability. The classes can be organised into hierarchy of classes & sub-classes. The new class is called derived class & the old class is called parent class.

* Polymorphism: This property allows a single func^n at conceptual level to take different forms depending on the object type or which it is acting. ie. different classes support same behaviour with diffent arguments.

* Information Hiding: Information hiding is supported through the concept of public, private & protected members of a class. Objects in a class through a well defined interface provide services to the users & hides implementation details.

* Association: It is a kind of relationship where all the objects have their own life cycle & there is no owner.
eg: Teacher-student relationship.
A Teacher can have multiple students & a student

can have multiple teachers. & both have their own lifecycle.

* Aggregation: It is a one to one relation In this all the objects have their own life cycle but there is a ownership. A child object cannot belong to another parent. eg: Teacher & department. One teacher can belong to one departmen but if the department is deleted the teacher remains.

Composition: It is a special form of aggregation. In this a child does not have its life cycle & if the the parent is deleted the child is also deleted. eg: A house can have multiple rooms b if the house is destroyed the rooms ar automatically destroyed.

Diffrences between structured & object oriented programming.

structured programming has following properties

1) subsystem
2) System boundary
3) System relationship
4) Modules
5) LifeCycle
6) Cohession
7) open/close system
8) Dependency.

1. Properties of OOAD

(1) Class
(2) Object
(3) Inheritance
(4) Polymorphism
(5) Information Hiding
(6) association
(7) aggregation
(8) Composition

2. Structured programming focuses on :- process & flow of data between the process.

2. OOAD programming focuses on objects & the interaction among the objects.

- Advantages of OOAD :

(1) Easier maintainance :
Objects may be understood as stand alone entity.

(2) Objects are appropriate, reusable components.

(3) For some system there may be obvious mapping from real world entity to system object.

Software Development Life Cycle (SDLC):

SDLC is a process used by software industry to design, develop and test high quality software. It aims to produce a high quality s/w that meets customer expectations. It basically represents the steps and timeline of software development model.

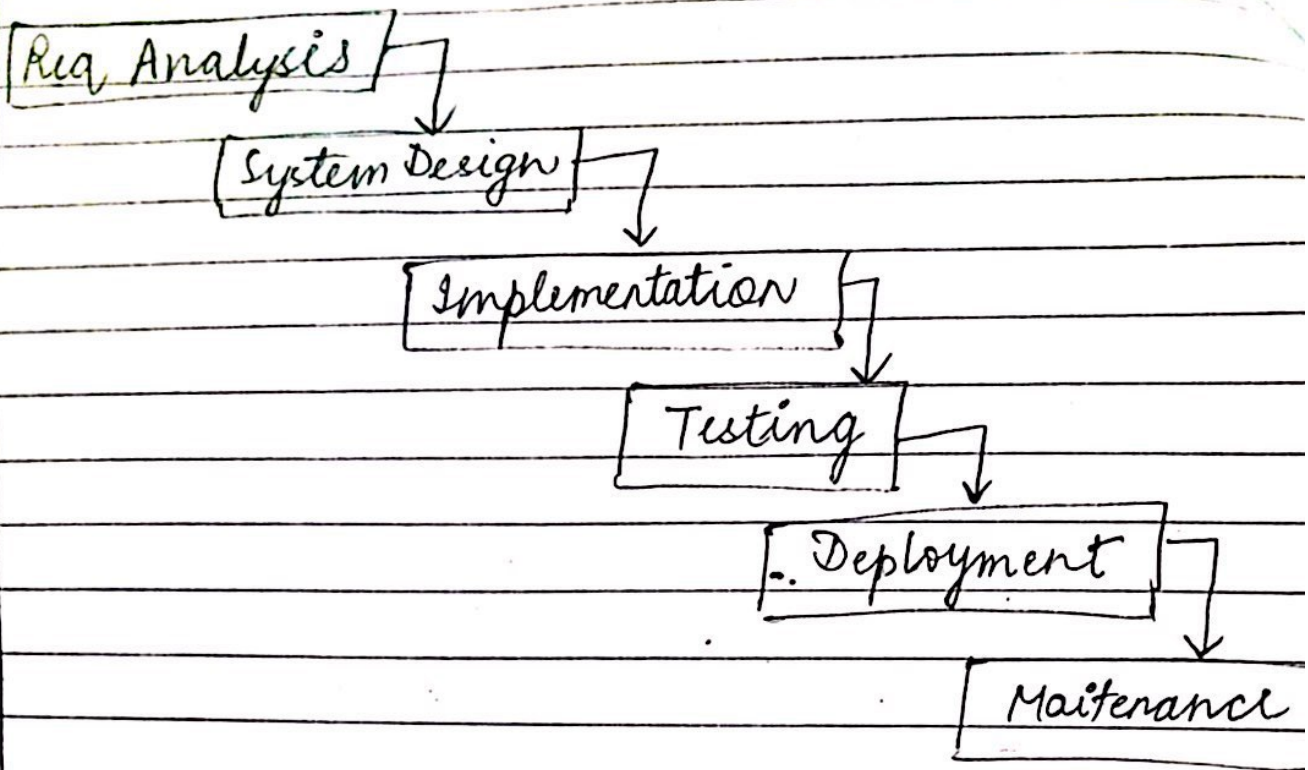
Steps of SDLC:

1. System study
   - Requirement specification
   - Req. Elicitation
   - Problem statement
2. Analysis & Design
3. Development
   - Algorithm
   - Coding
   - System / Testing.
4. Implementation
   - Deployment
   - Maintance.

SDLC Models:

(1) Waterfall Model
(2) Iteration Model
(3) Spiral Model
(4) RUP.

* Waterfall Model



Waterfall model is simple to use and understand. In this model each phase must be completed before the next phase begins and there is no overlapping b/w the phase

Advantages:

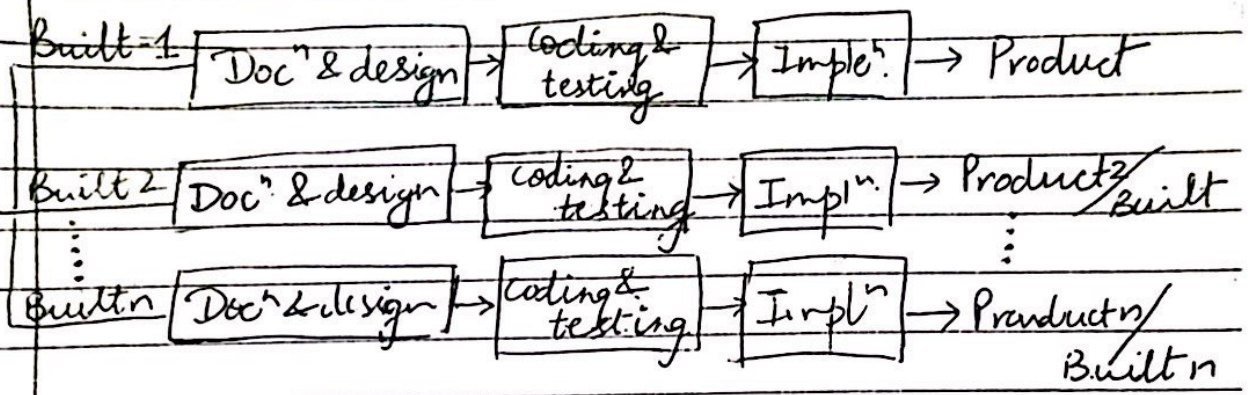
→ simple to use and easy to understand
→ easy to manage
→ phases are completed and processed one by one
→ works well for small projects.

Disadvantages:

→ No working s/w until late.
→ High uncertainity and high risk.
→ Cannot accomodate change in the req't's.

Date.....

(2) Iterative Model:



* It is a way of breaking down the s/w development of large applications in small chunks. In iterative development the featured code is designed developed & tested untill there is a fully functional s/w application body to be deployed to customers. Large project would be broken into small chunks.

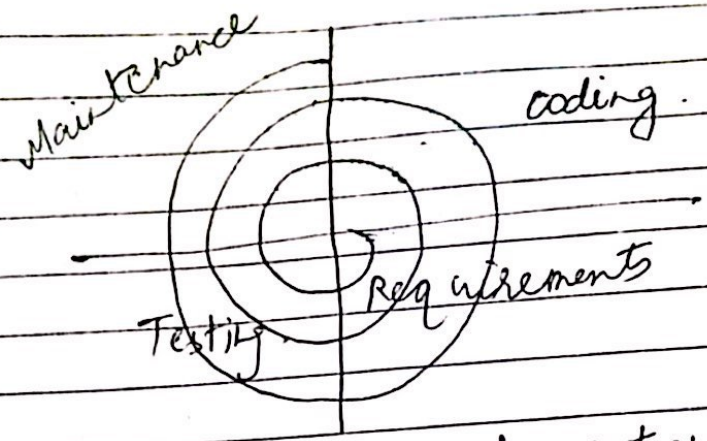
Advantages:

→ Results are obtained easily & early.
→ testing and debugging is easy.
→ risk identification are reserved during each iteration.

Disadvantages:

→ Not suitable for small projects & high complexity.

Spiral Model: (Iterative + Waterfall Model).



It is a combination of waterfall model & iterative model. It takes place in form of four iterations that are: identification → design → construct → evaluation & risk analysis in spiral form.

Advantages:
User can see working system early.
changing req't's can be accomodated.

Disadvantages

It works on large system only.

* RUP → Rational Unified Process:

The phases of RUP model are:

(1) Inception
(2) Elaboration
(3) Construction
(4) Transition

RUP is a software development process from rational, a division of IBM. RUP is an object oriented and web enabled program development methodology.

Phases of RUP:

(1) **Inception:** In Inception the idea of the project is taken to development then team & it determines if the project is worth pursuing and what resources will be needed.

(2) **Elaboration:** The project is in architecture and required resources are further evaluated. Developer considers posible application of the software and the cost associated with the development.

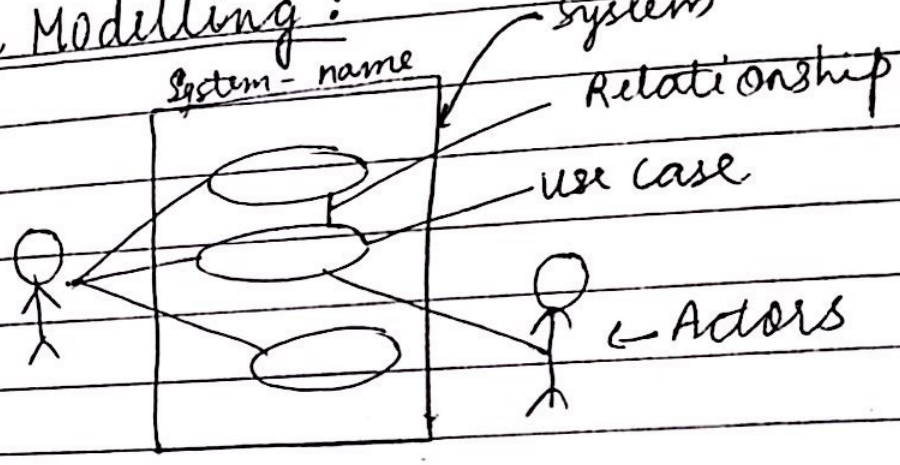
(3) **Construction:** If the project is developed & completed. The project is designed, constructed and tested.

(4) **Transition:** The software is released to public. Final adjustment or update are based from feedback of end users.

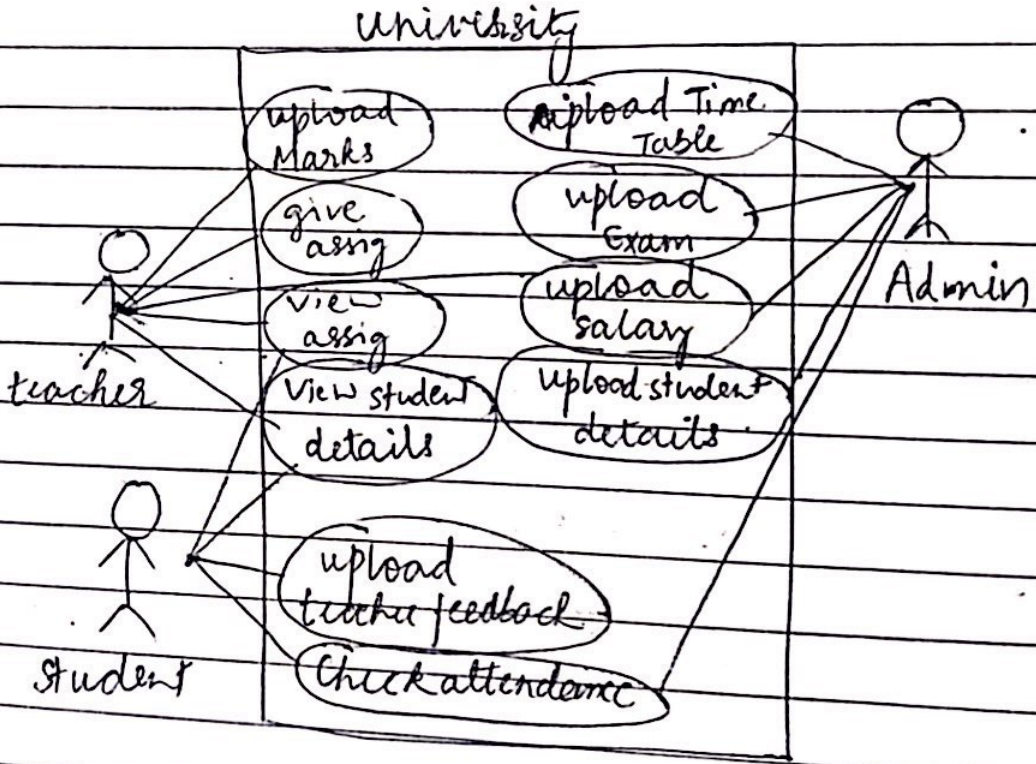
Date.....

Unit - 2

Use Case Modelling :



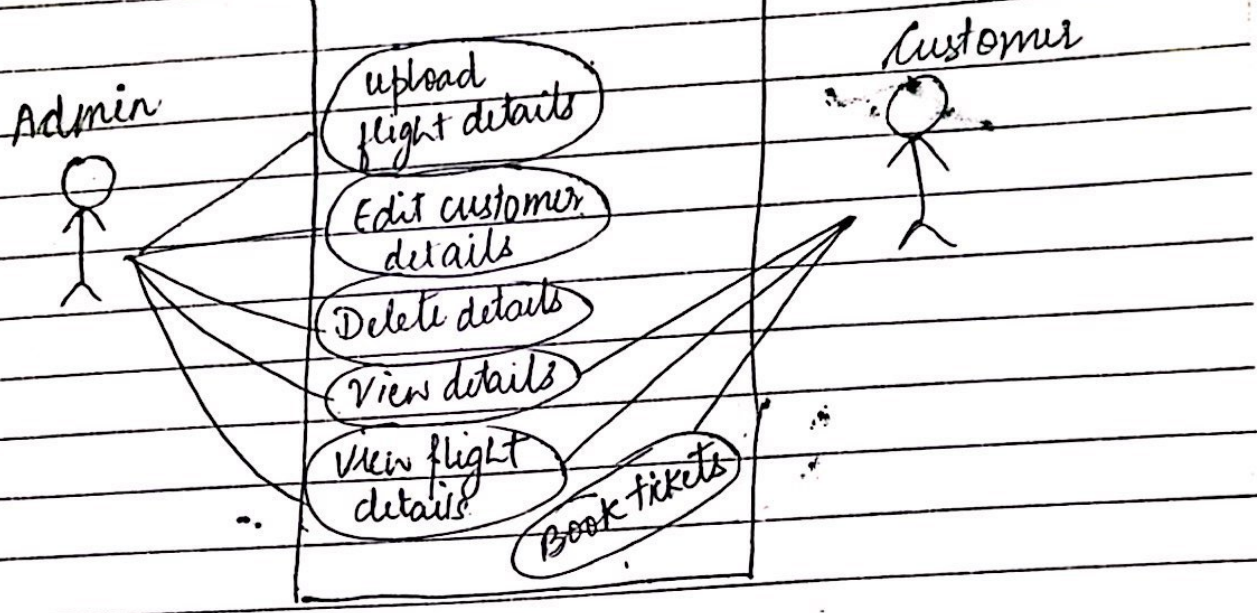
* University Management



Ques Draw a Use Case diagram for Online Air ticket Booking System.

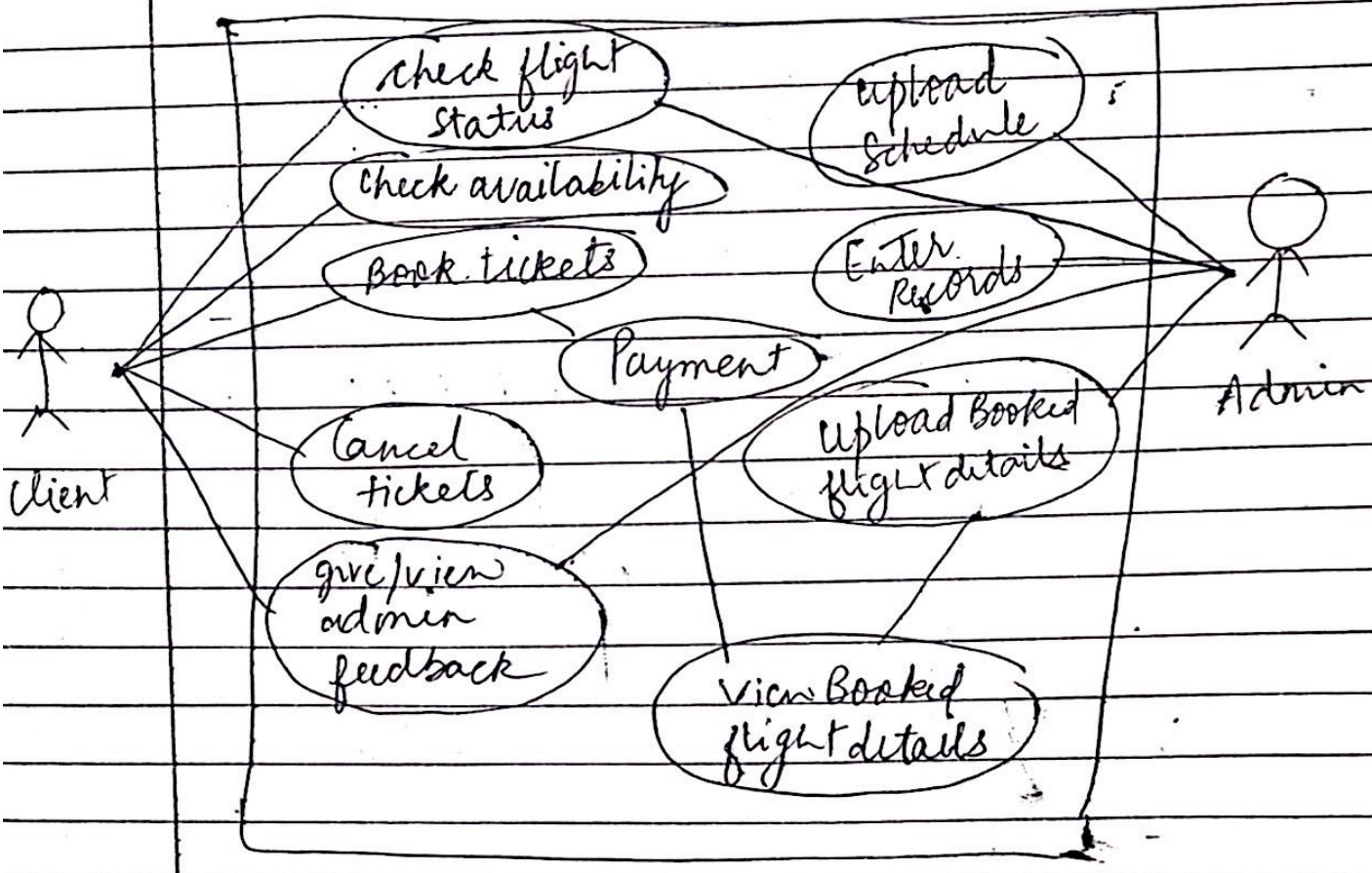
Date.....

Airlines Reservation System



OR

Airlines Reservation System



Unit-2

Date.....

A use case Diagram describes the system from user point of view partitioning the system functionality into transactions that are meaningful to the user.

The UCD's are used for the following purpos

- → To gather reqt's of the system.
- → To get outside view of the system.
- → To identify external & internal factors influencing the system.
- → To show interaction among actors.

A UCD specifies events of a system and their flow but never describes how they are implemented. A UCD can be imagined as a black box where only the inputs & outputs and funct's of the Black-box are known. A UCD has following components::

1. System: A system shows the boundary of the organisation to be build and is represented by a rectangle.

2. Actors: An actor is a role played by a person, organisation & other system devices which interacts with a system. An actor can be called as the one who initiates some events which in turn triggers the activities of the organisation. It is shown by

Date.....

3. Use cases:

It shows the required functionality of the system without specifying how it will be achieved. It is shown as ⬭.

Relationships: It shows which actor interacts with which use case.

★ Types of Actors:

(1) Primary Actors: It is the one who initiates a use-case. Eg Admin.

(2) Secondary Actors: It is the one who supports the use case but doesn't initiate it. Eg: students s/w which runs for the system.

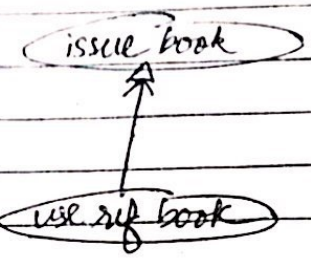
(3) Offline Actors:
The actors such as govt. or ISP who have no direct role in the system, but are used for a particular use case.

★ Types of Relationships b/w use case:

(1) Associations ⟷ or ——

(2) dependencies → include MC —«include»→ EC
→ extends MC —«extends»→ EC

(3) Generalization:

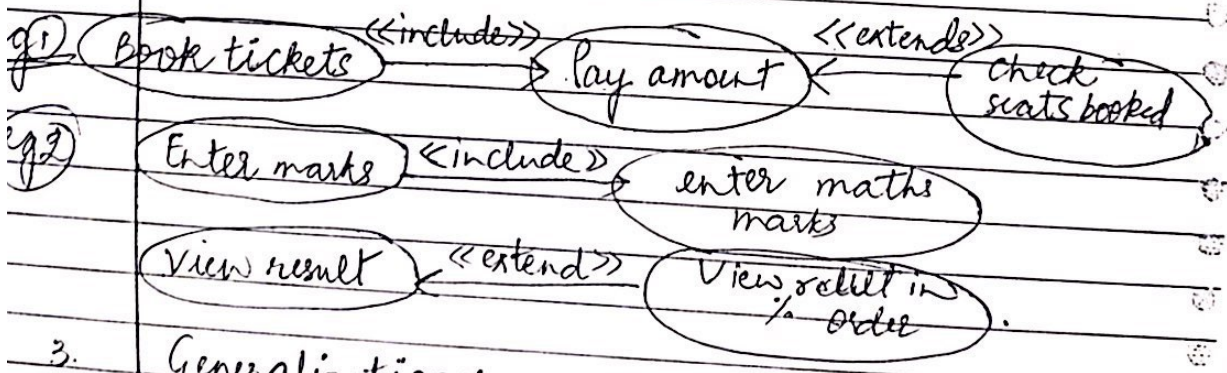


Date.....

1. **Association:** It shows communication of actors and are represented by straight lines connecting actors & use cases.

2. **dependencies:**
They are used to show relationships between use-cases (a) <<include>>
It says that the execution of one use-case includes the working of another use-case always. The arrow direction is from the main use case to the one which is required by it.

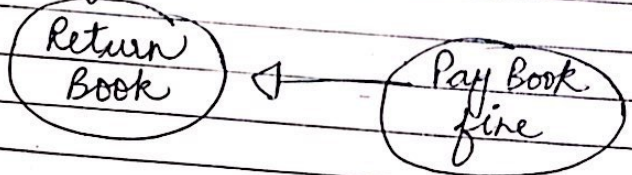
(b) <<extends>>
It says that one use-case may or may not use another usecase. The arrow is from the extended use-case to the main use case.



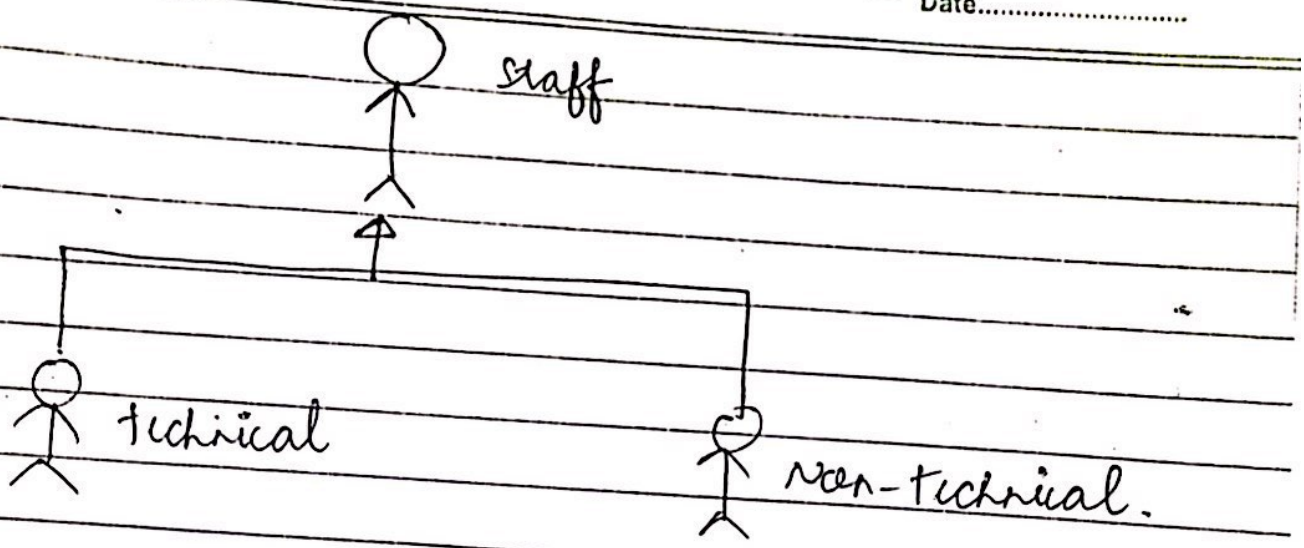
3. **Generalization:**

It is used to show the concept of inheritance among uses cases & actors.

Eg 1



q2. — Date.....

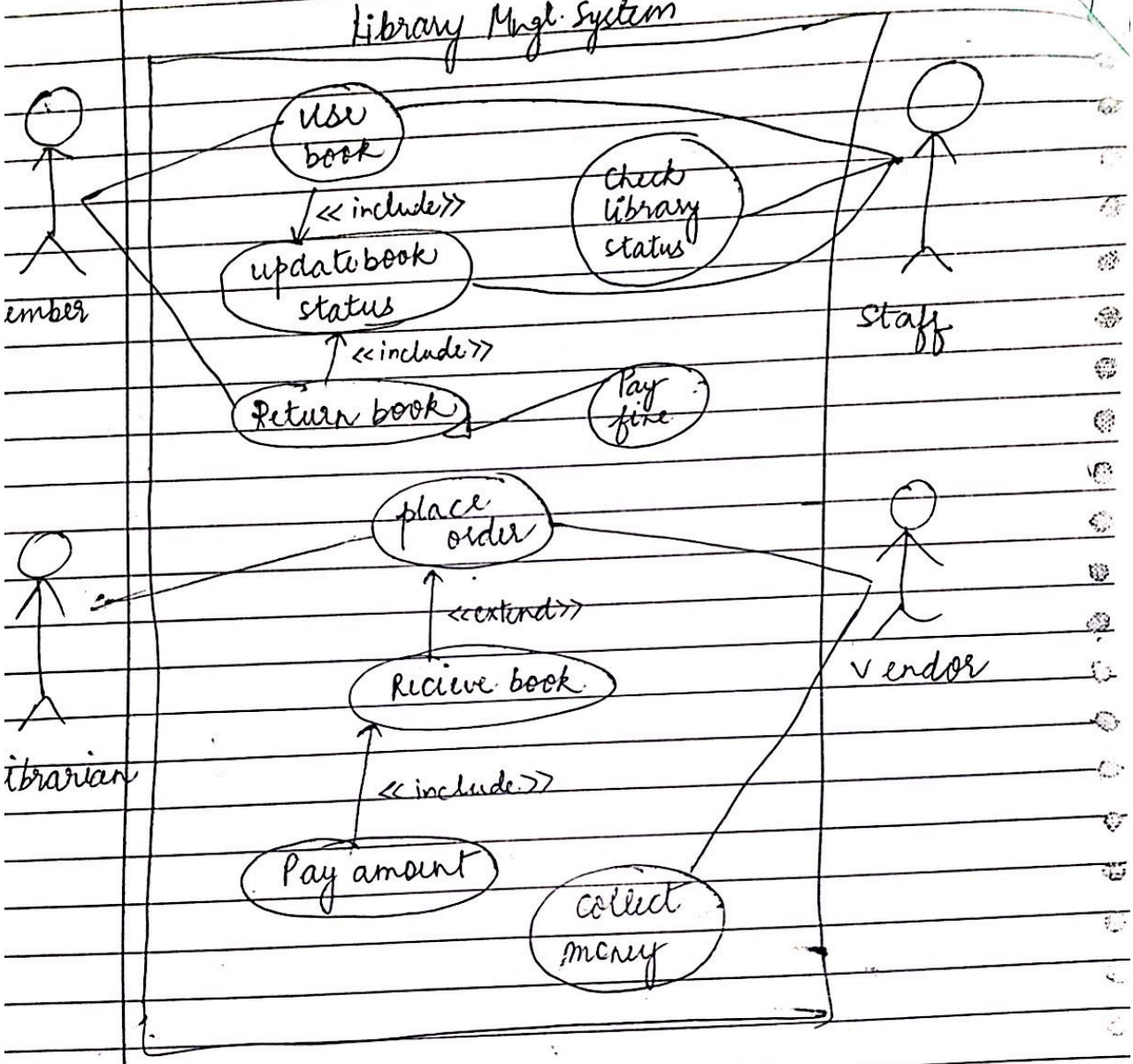


* Steps to draw a use-case:

1. Set the boundary of systems
2. Identify the actors.
3. Identify the use-cases.
4. Establish association b/w actors & use-cases.
5. Identify include & extend relationship b/w use cases.
6. Identify the generalization b/w actors & use-cases.
7. Define / draw the diagram.

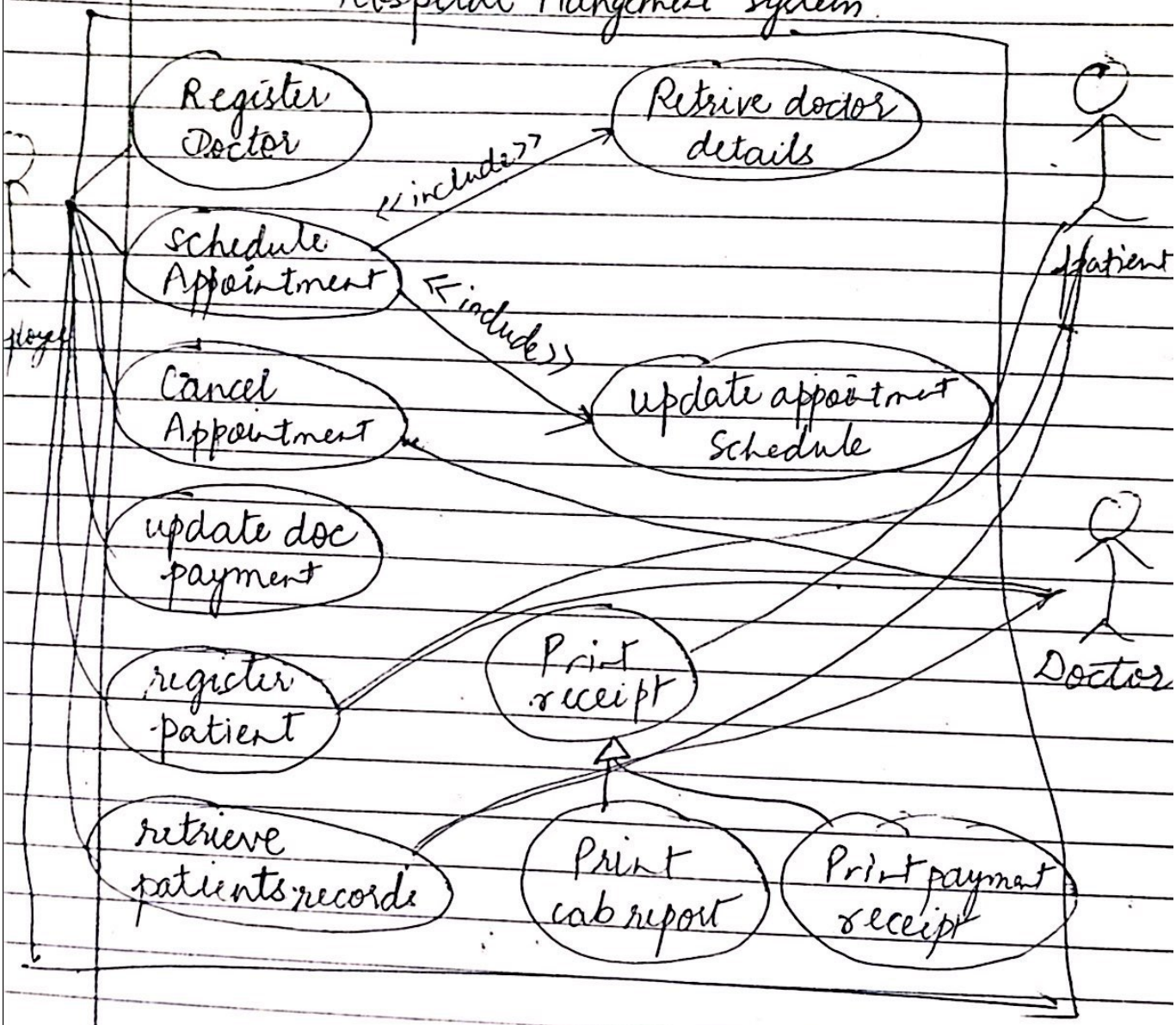
Date.....

* Draw a UCD for Library Mngt. System.



Date.....

Ques Draw a UCD for Hospital Mangt. System.



Date.....

Requirement Engineering :

Steps of Requirement Engineering :

(1) Inception (2) Elicitation (3) Elaboration
(4) Negation (5) * specification (6) validation
(7) Req. Mangt.

Requirement Engineering is the process of collecting the S/W requirements from the client, and understanding, evaluating & documenting it, is called Requirement Engineering.

Steps :

(1) Inception : It is the task where the requirement engineer ask the set of questionniare to establish a S/W process.
- In this task it understands the problem & evaluates the proper solution.
- It colaborates the relationship b/w customer & developer.
- The developer & customer describes the overall
(2) Etc scope and nature of the question.

(2) Elicitation : It deals with finding the requirements from anybody. It includes requirements in terms of problem of scope, problem of understanding, problem of validating.

(3) Elaboration : In this task information taken from Elicitation & Inception are expanded and refined.
It's main task is to build a border of the S/W

Date.....

using function, features & constraint of a s/w.

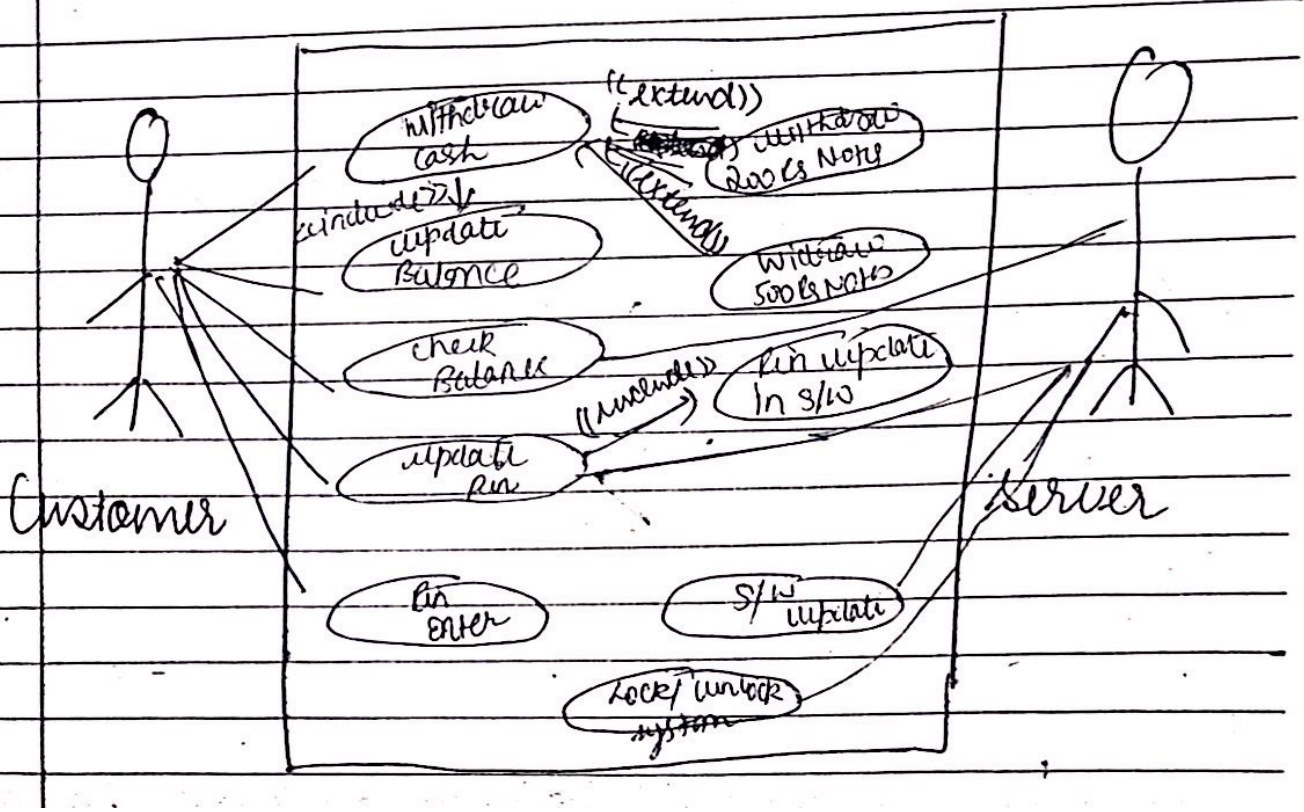
(4) Negotiation: In this try to achieve minimum business resources and requirement for the project to be completed.

(5) Specification: It constructs the final work product of requirements in a particular manner called s/w requirement specification.

(6) Validation: In this the s/w is accessed for quality through both the developer & client party.

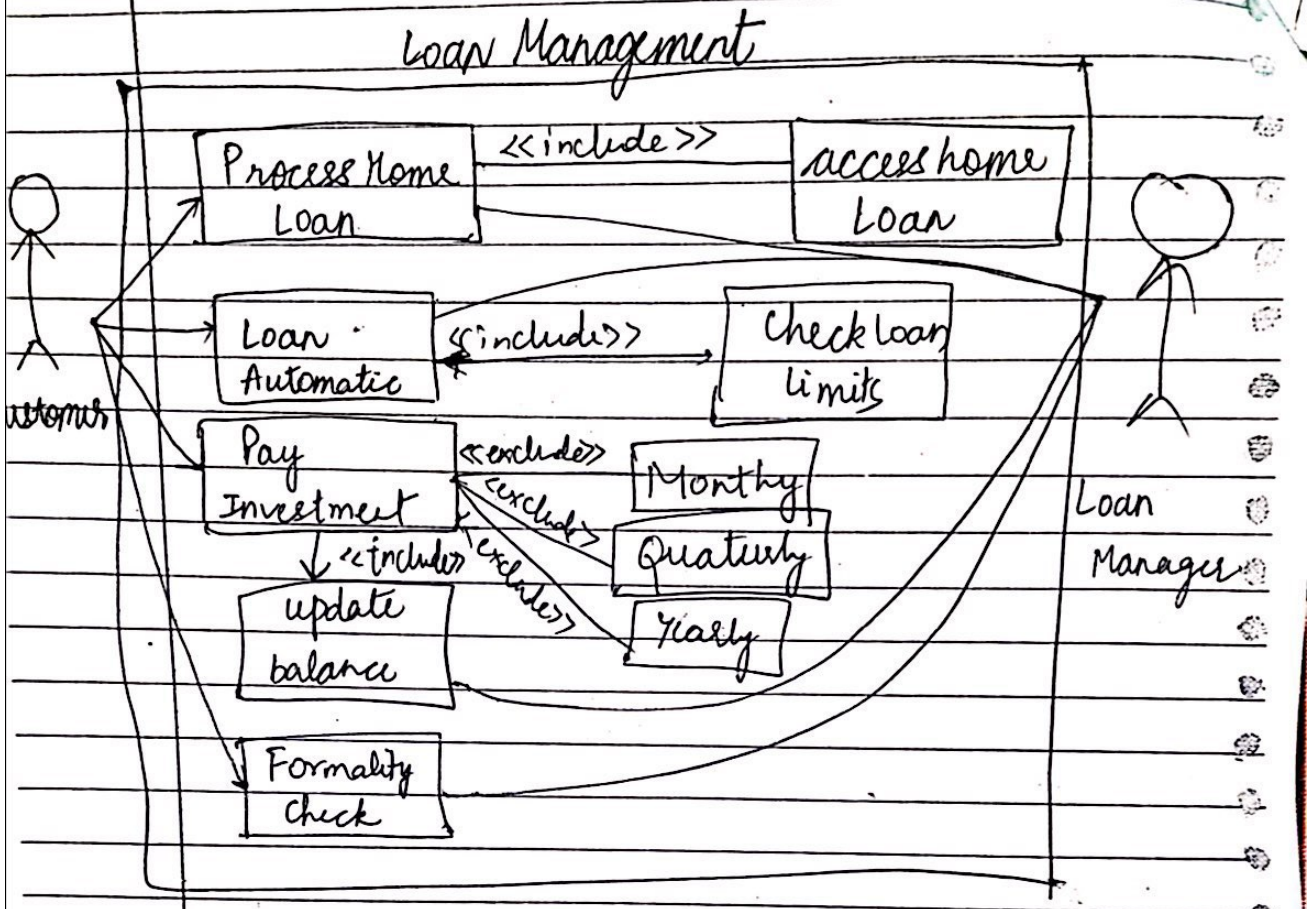
(7) Requirement Management: In this the actual resources are checked for the process.

Ques Draw a use case diagram for ATM a machine.



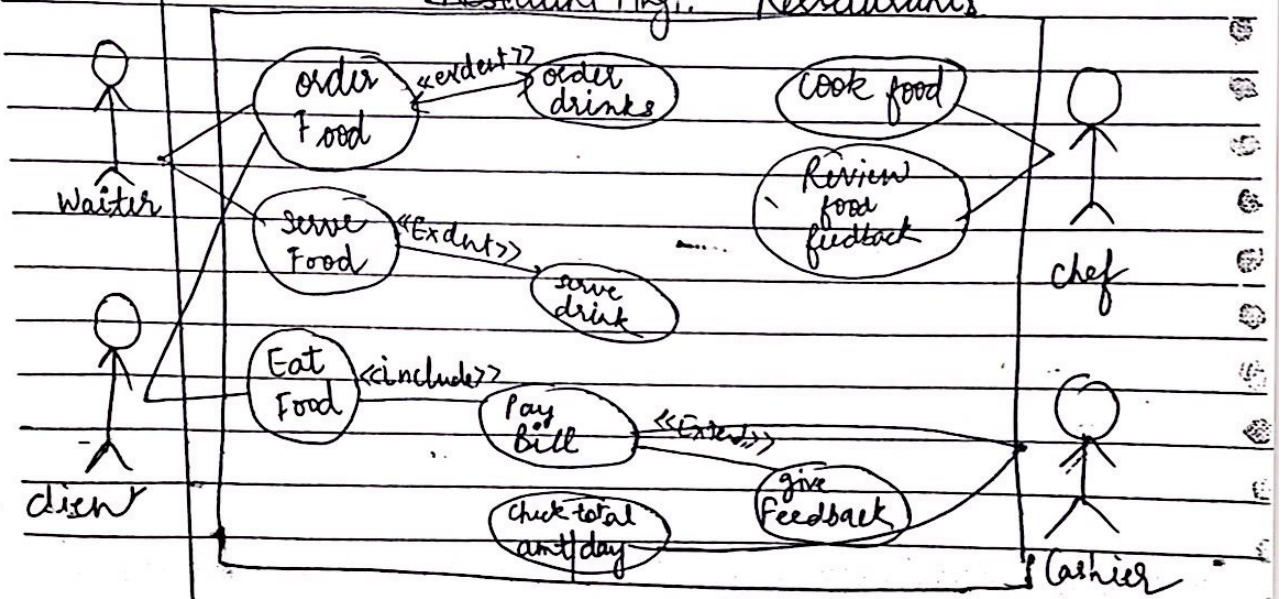
Date.....

Q.5 Draw a use case diagram for loan Management system.



Q.6 Draw a use case Diagram for restaurant Mangt. System

Restaurant Mgt. Restaurants



Date.....

Types of Requirements:

1) Customer Requirement: The customer reqt. defines the expectations in terms of mission, object, environment & constraint.

2) Functional Requirement: They explain what has to be done. It identifies necessary actions or activity and task.
It is used as a top level func" for func" analysis.

3) Non-functional Requirements: It deals with requirements related to behaviours of a system rather than the operation.

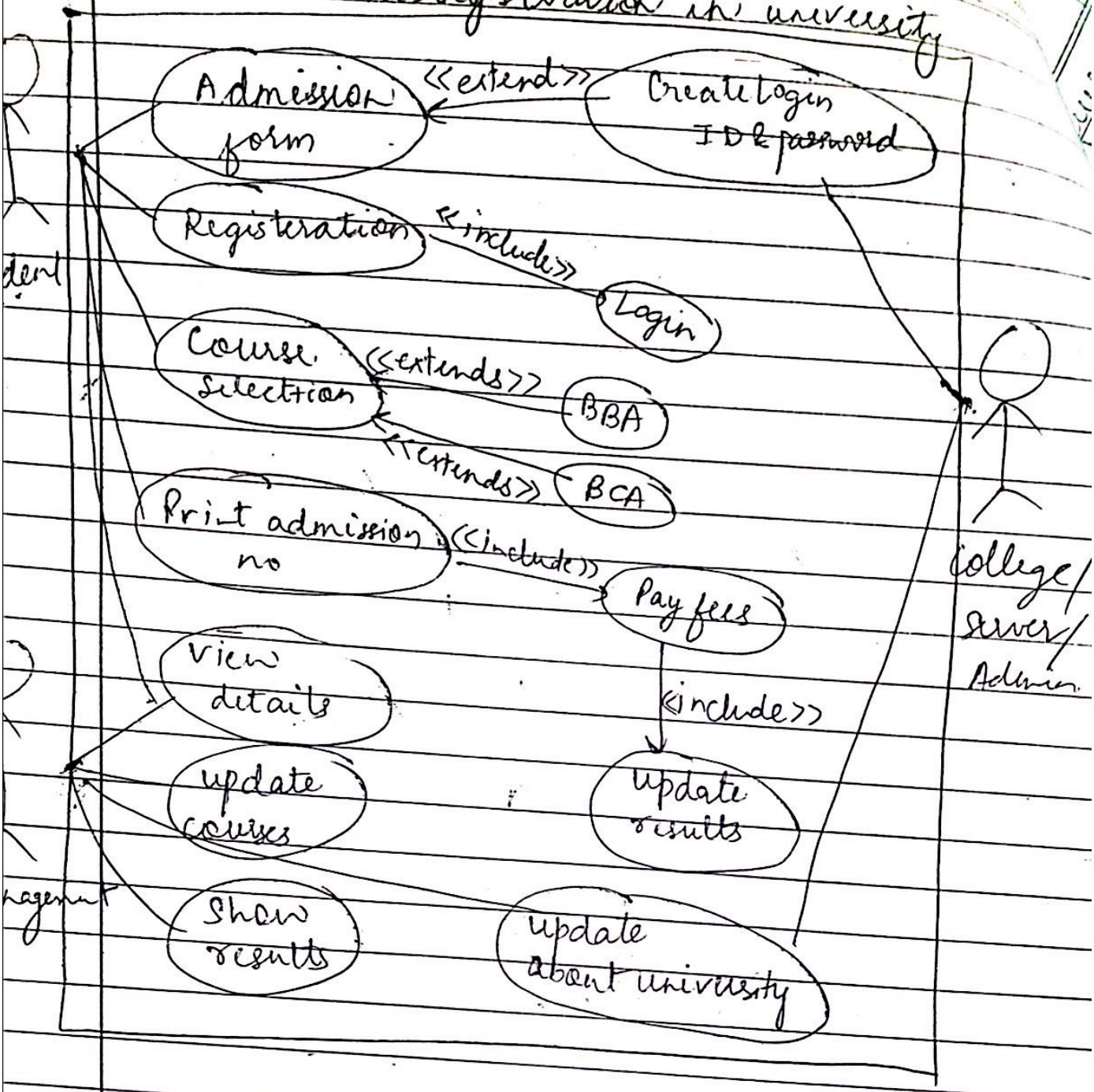
4) Performance Reqt's: It deals with what mission or func". must be executed.

Ws Draw a VCD for online registration for a course in a university.

→

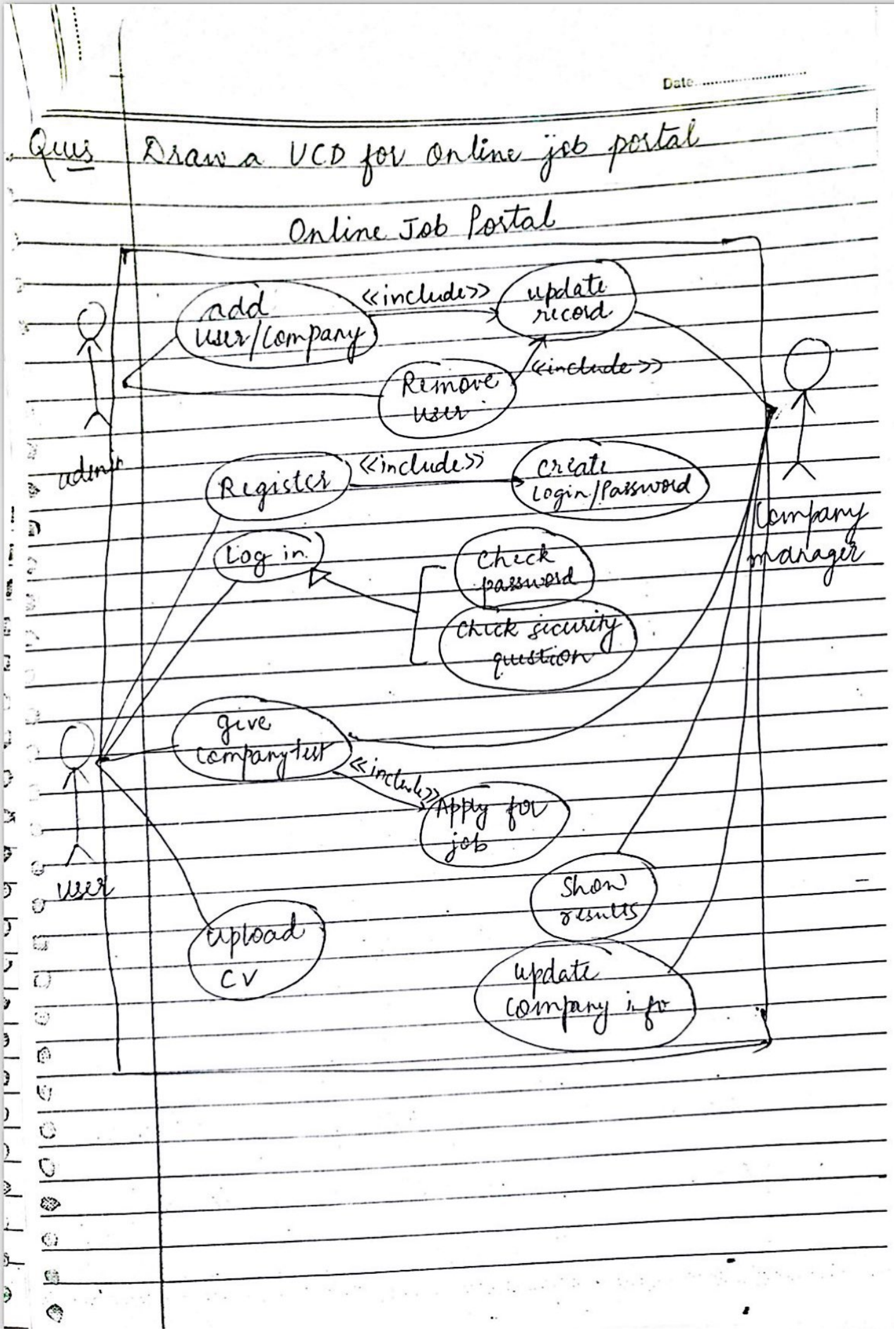
Date.....

Online registeration in university



Date .....

Ques Draw a UCD for online job portal



Date.....

Unit-3
Sequence Diagram

A sequence diagram basically depends upon the sequence of actions that occur in a system. It captures the invoke of the messages or methods of objects and order of invokation. It basically represents the dynamic or functional behaviour of the system.

To model the interaction b/w the objects in a single use-case, they illustrate how different parts of the system interact with each other. To carry out the functions and the order in which the interaction occurs. when a particular use case is executed. For each particular Use Case Diagram we can draw a sequence diagram.

* Notation in Sequence Diagram.

1. objects [Name]
It is any real world entity associated with a use-case.

2. Life-Line | or ↓ ; It is basically used to show timeline of an object.
No two lifelines can overlap with each other.

3. Action Bar | ▯ | It is based on life-line & is used to indicate when an object is active during an interaction b/w two objects. The length of rectangle indicates the

Date.....

duration of the object staying active. The interactions b/w the two objects occur when one object sends a message to another object.

4. Message Arrows

An arrow of the message from the caller to the receiver specifies a message in the sequence diagram.

- Synchronous Message ——▶.

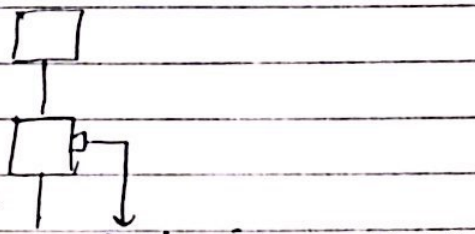
It is used when the sender waits for the receiver to process the message and return before carrying one with another message.

- Asynchronous Message ——▷.

It is used when the message caller does not wait for the receiver to process the message and return before sending other messages to other objects within the system.

- Reflexive Message

It is a message sent by object to its cell.

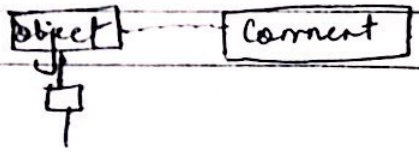


- Lost message ——→⊙

- Found message ←——○

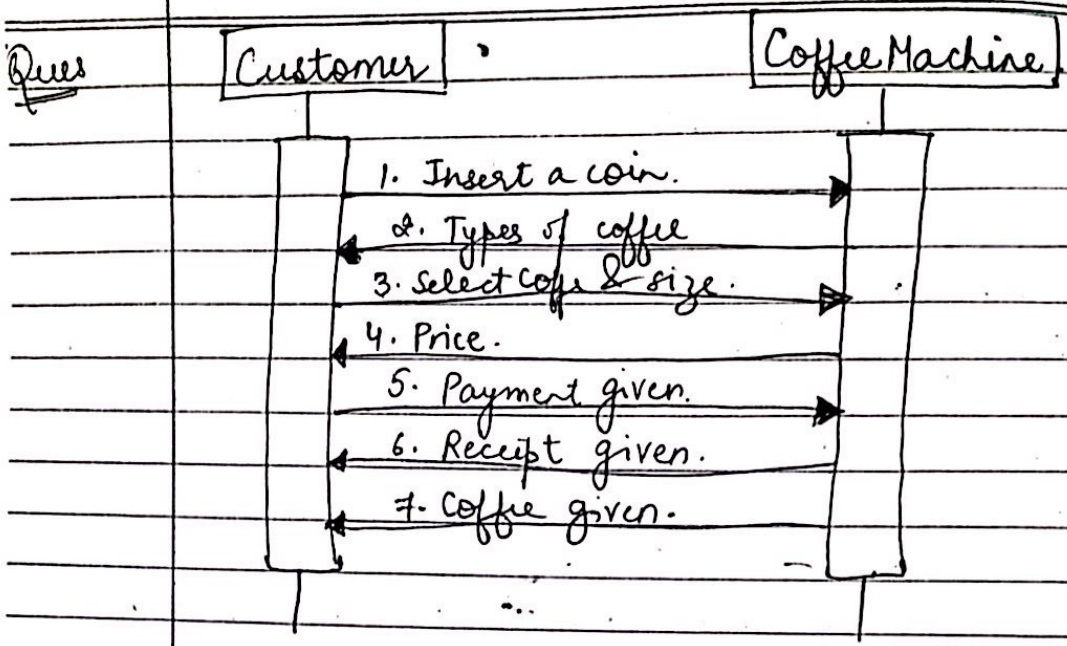
- Delete message ——→X

→ Comments message

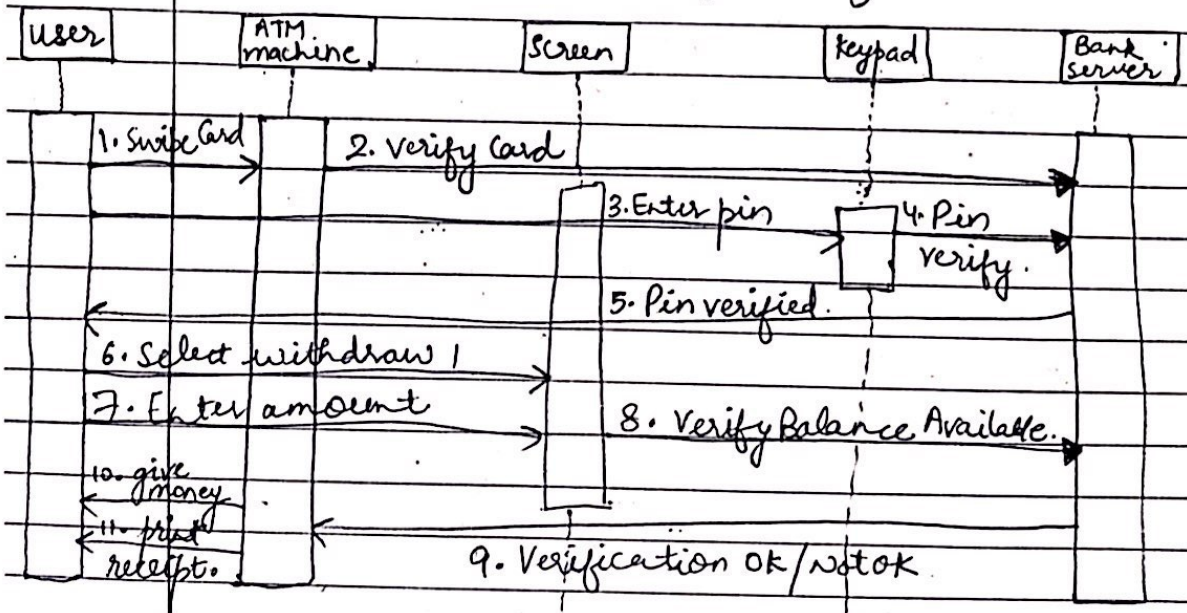


Date.....

Ques

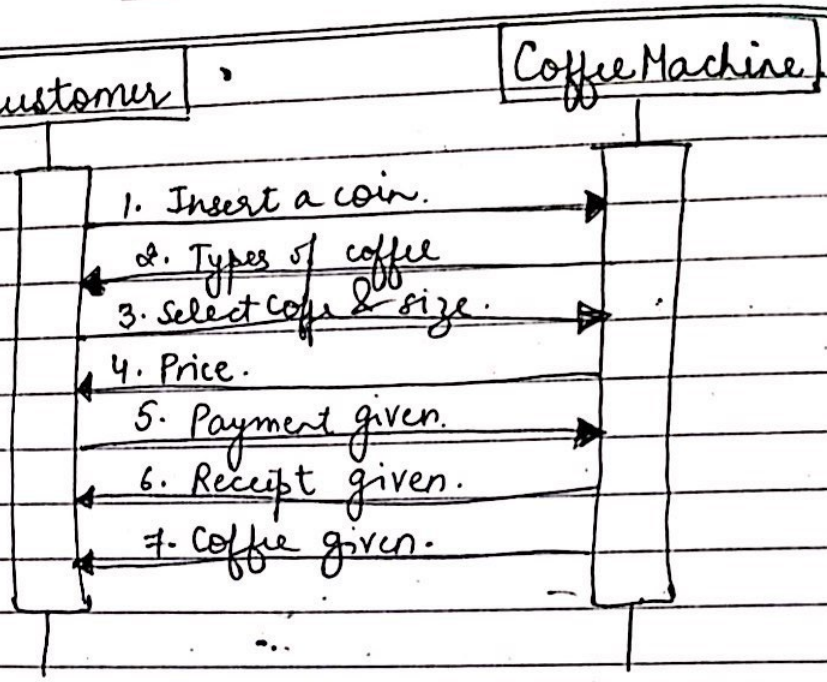


Ques. Draw a sequence diagram for ATM machine.

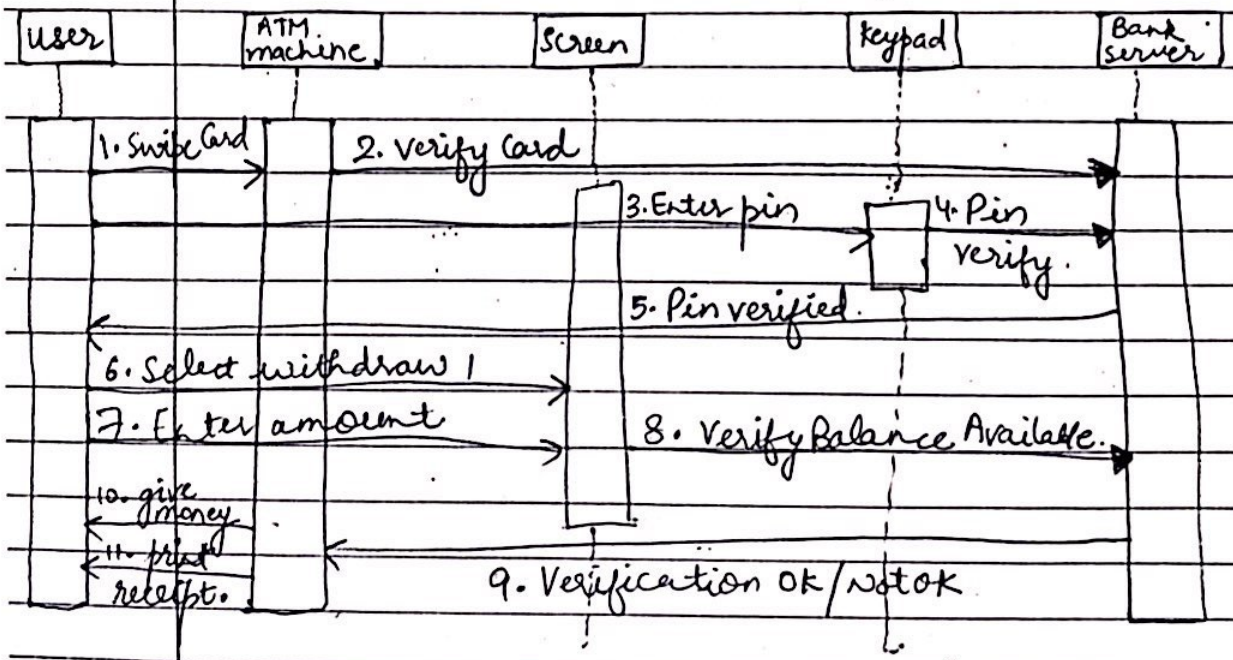


Date.....

Ques



Ques. Draw a sequence diagram for ATM machine.



Date.....

* Following are the steps to draw a sequence diagram:

(1) Draw a use case diagram.
(2) Focus on one particular use case
(3) Identify the actors/objects that would be involved in that particular situation
(4) Write a detailed description on what the use case does.
(5) Figure out the interactions taking place in that situation.
(6) Identify the messages exchange.
(7) Draw the sequence diagram.

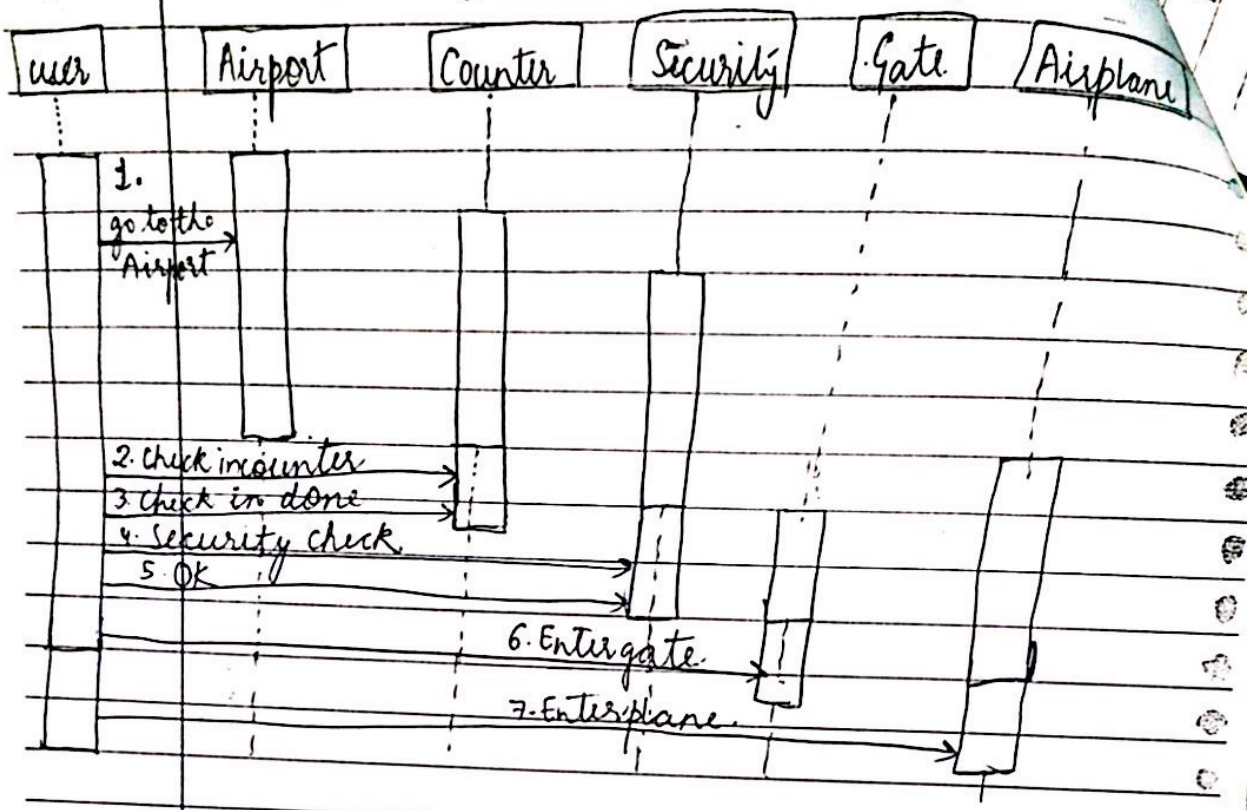
* Common Mistakes while drawing the sequence diagram.

(1) Adding too much details
(2) Ab Obsolete & out of date sequence diagrams that are irrelevant when compared to the interfaces
(3) Not considering the origin of message arrows carefully.

Ques: Draw a sequence diagram for Airport Authority.

Date.....

Sequence Diagram for Airport Authority



* Uses of Sequence Diagram:

1. They are used to model and visualize the logic behind a sophisticated function, operation or procedure.
2. They are used to ~~not~~ show the details of the UML use case diagram.
3. They are used to understand the detailed functionality of the current or future systems.
4. They are used to visualize how messages and task move between objects & components in a system.

* Strengths of Sequence Diagram:

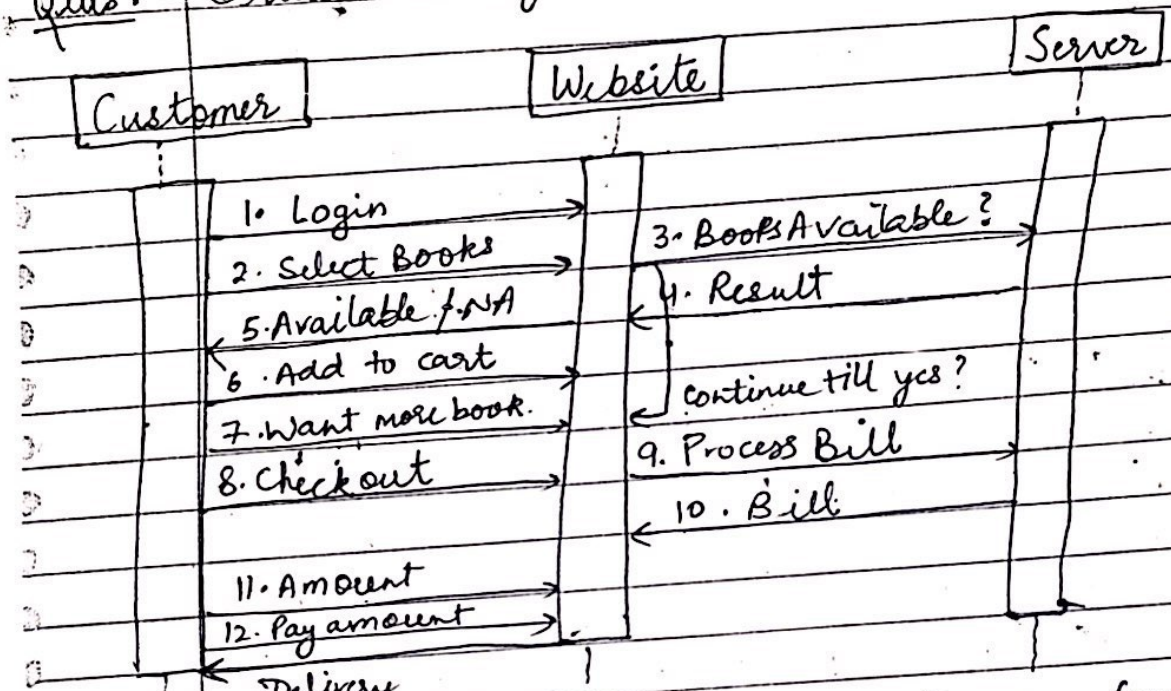
1. The sequence diagram clearly shows sequence or time or ordering of messages.
2. In SD there are a large set of detailed notation options.

Date.....

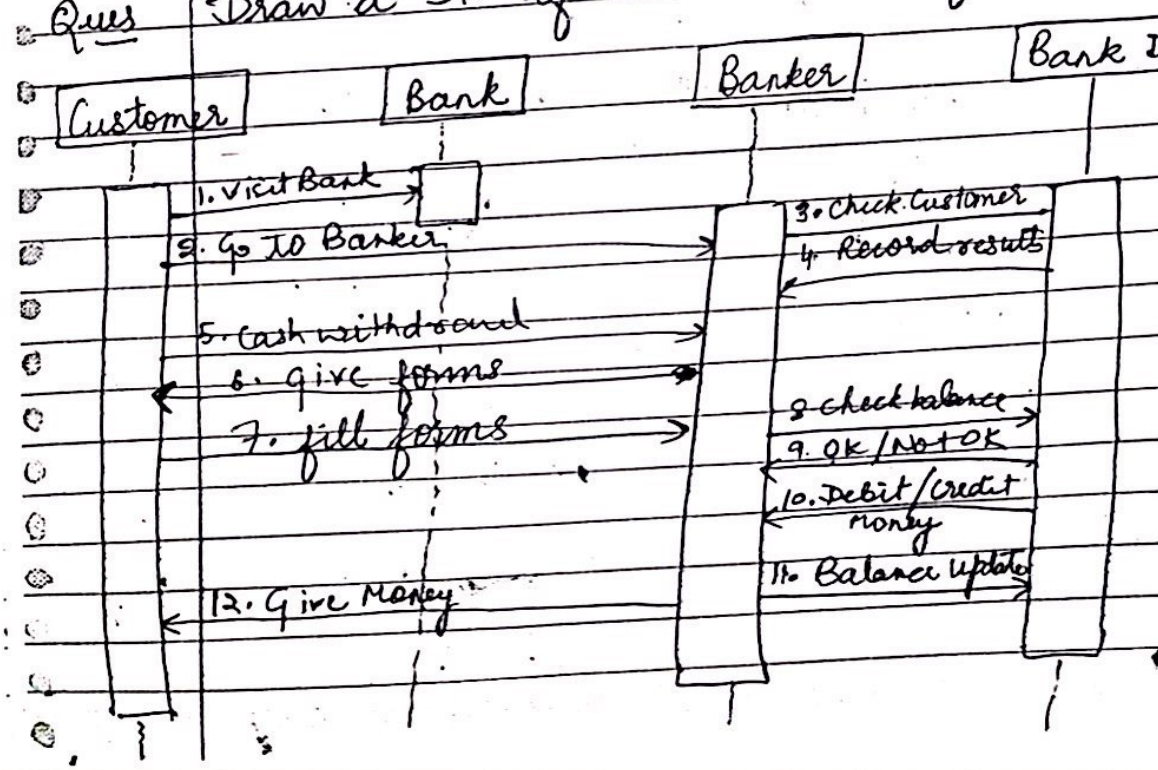
* Weakness of SD:

1. They are forced to extend to the right when adding new objects, horizontal spacing.

Ques: Draw a SD for online e-book purchasing

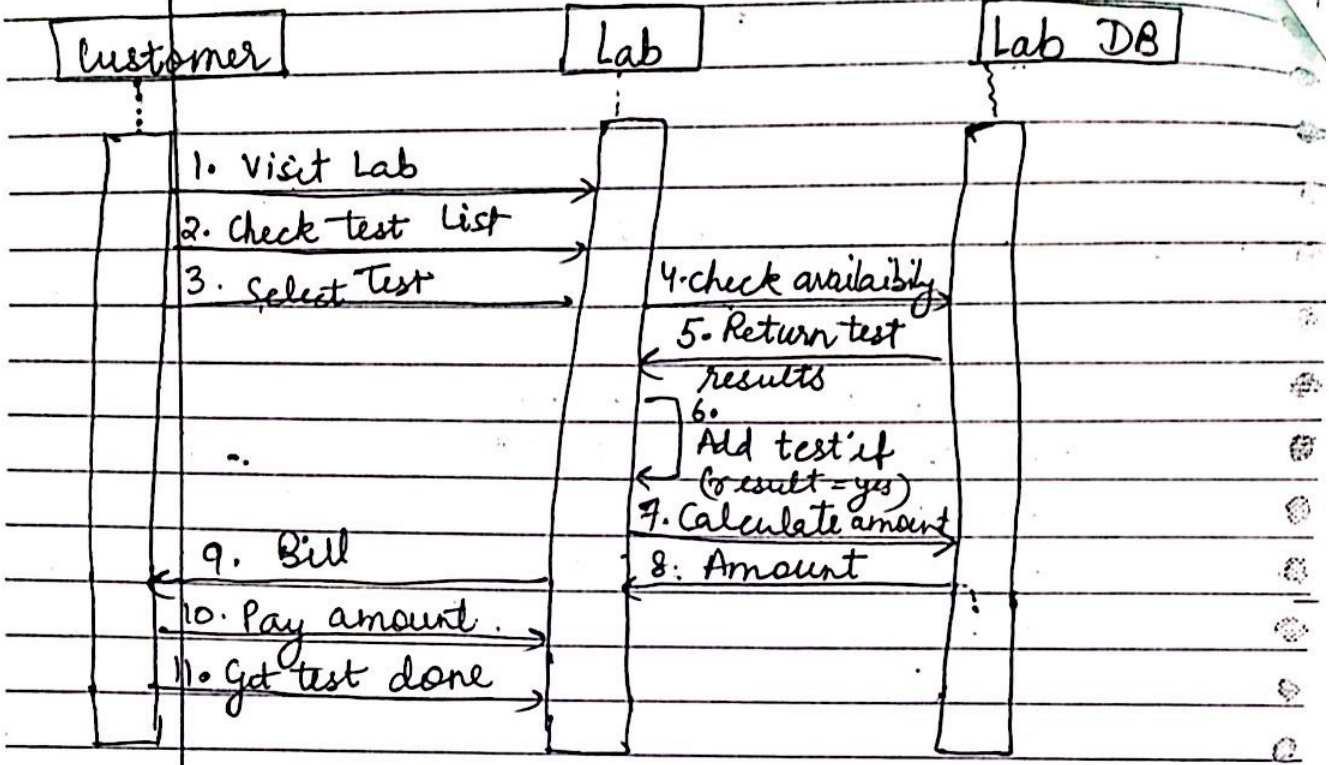


Ques Draw a SD of cash widrawl from bank.

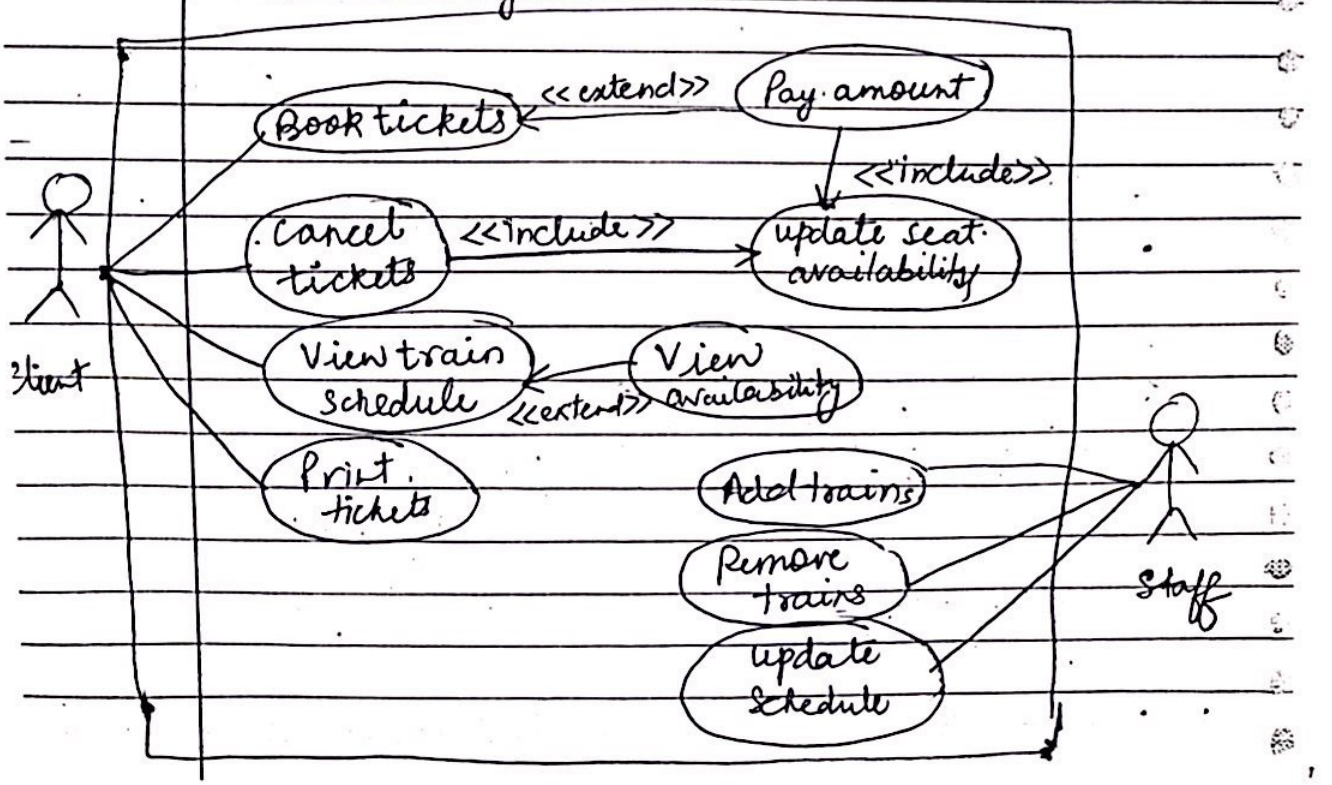


Date.....

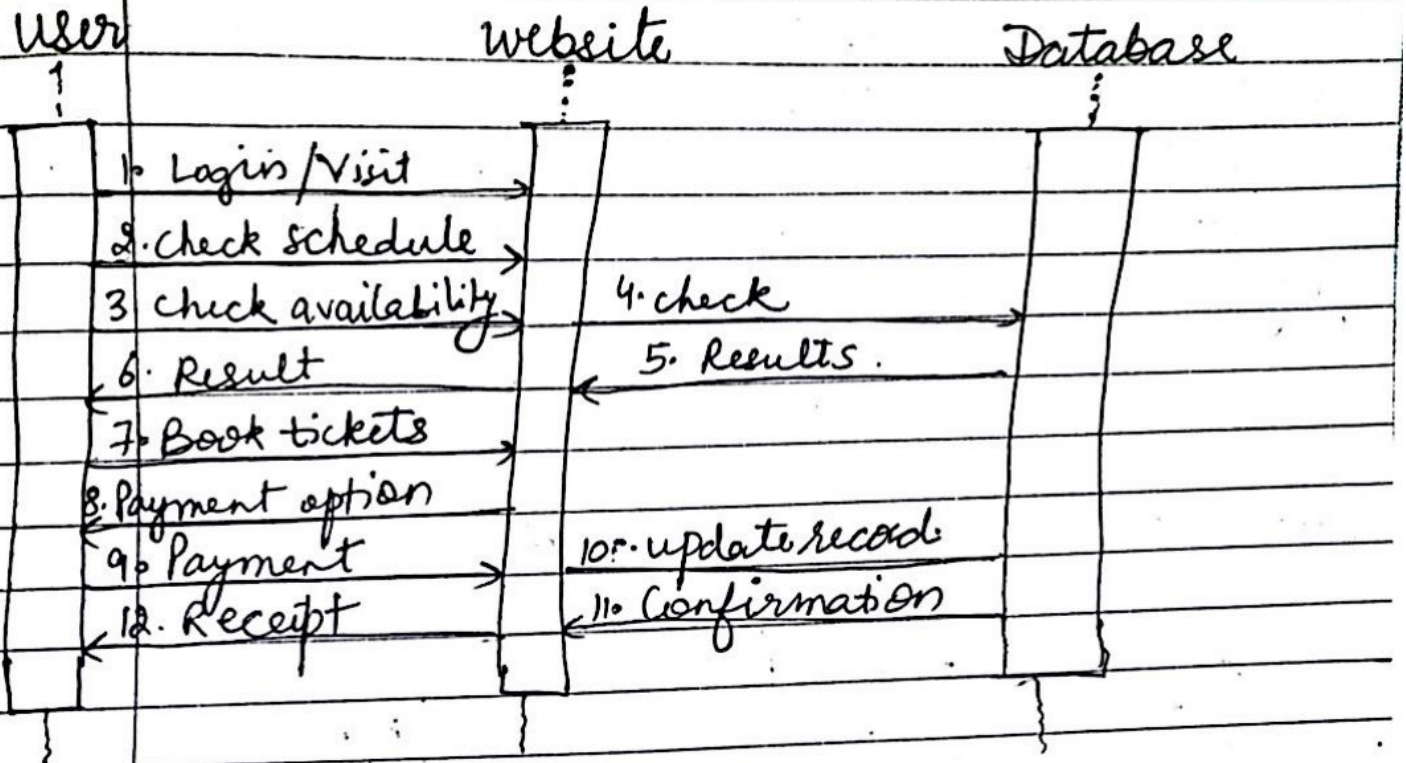
Ques Draw a SD for bill and lab diagnostic centre.



Ques Draw a use case diagram & SD for railway reservation system.



Sequence Diagram

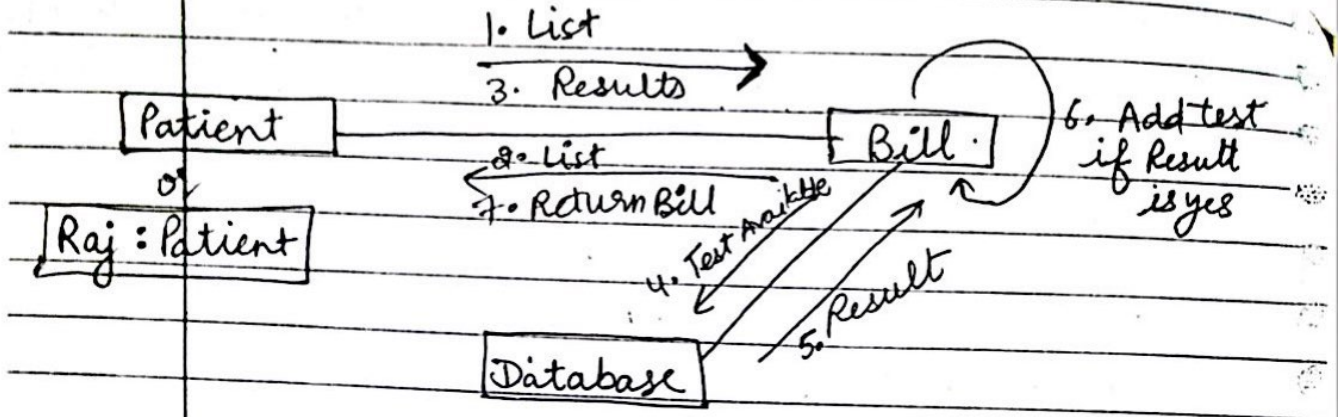


Collaboration Diagram

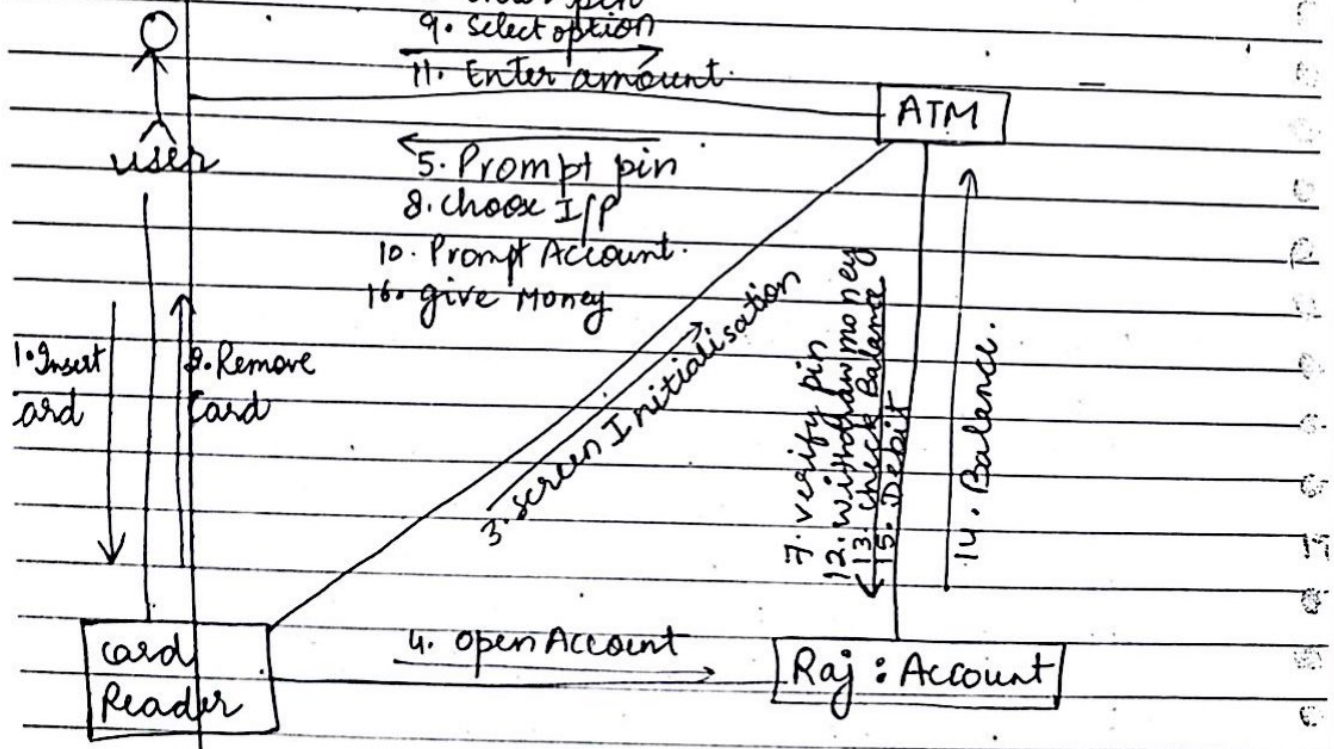
The collaboration diagram models the object interaction like sequence diagram but instead of focusing on the type of messages that are sent, it focusses on interactions with respect to object relations. The object can be placed anywhere in the diagram. In CD the method call is indicated by the numbering b/w which indicates which number is called one after another. We can decide which diagram to draw (sequence or collaboration diagram) depending upon the type of requirement. If the time sequence is important the SD is used and if organisation is important then CD is used.

Date.....

Ques Draw a CD for billing at diagnostic centre

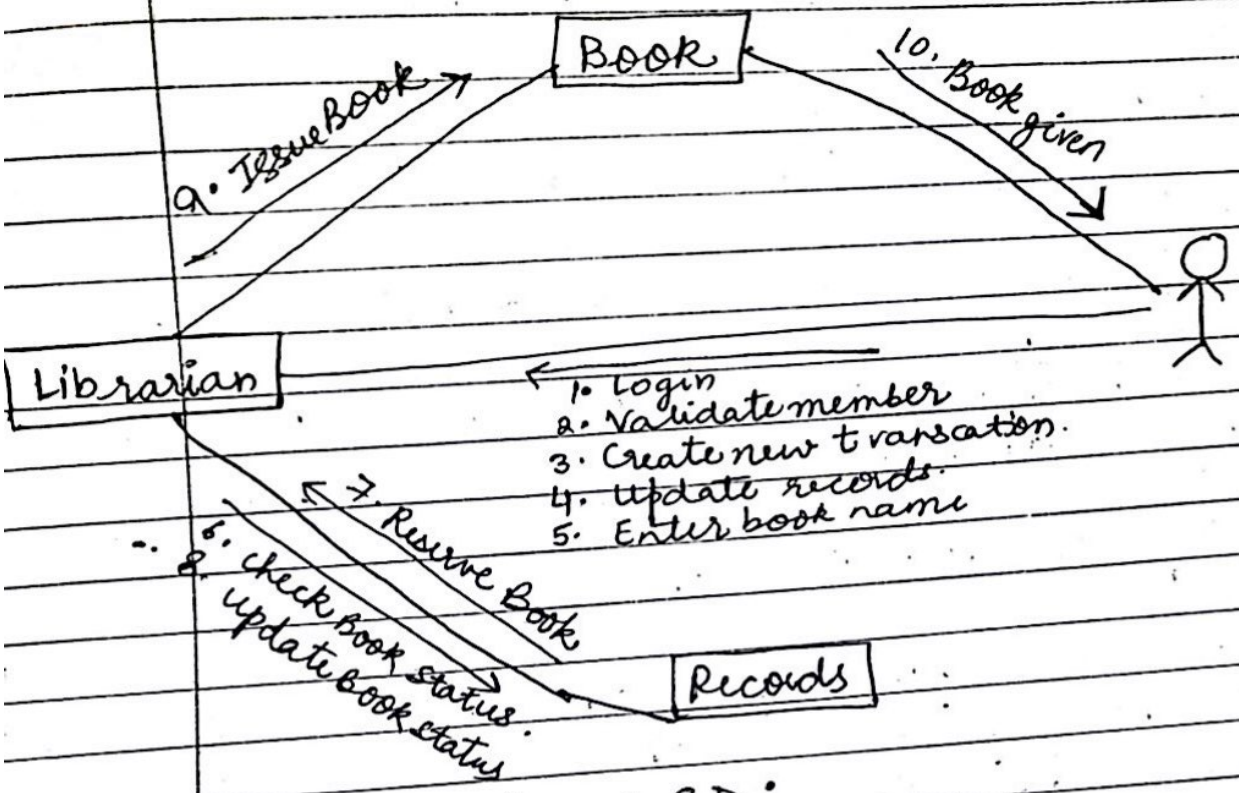


Ques Draw a CD for ATM



Date.....

Ques Draw a CD for library Managment System

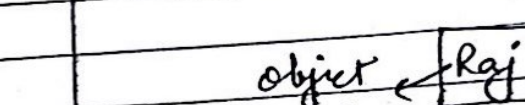


* Steps to draw CD:

(1) scope of diagram
(2) objects
(3) Initial values
(4) Links b/w the objects
(5) Messages with each link
(6) Sequence no.

* Contents of collaboration Diagram:

(1) Object: An object will be represented by a rectangle showing the name of the object and its class separated by a colon.



Date.....

Actor: An actor instance occur in CD as the interaction with other actors.

Link: A link is a relationship among objects b/w which messages can be exchanged.

4. **Message:** Messages are communication b/w objects that convey information expected with the activity. The messages are shown by a labelled arrow via a link.

* **Steps to draw a CD**

(1) Determine the scope of the diagram. As with SD the scope of CD can be a use case.
(2) Place the objects that participate in the collaboration on the Diagram. Remember to place the most important objects towards the center of diagram.
(3) If a particular object has a property or maintains a state that is important to the collaboration, set the initial value of the property or state.
(4) Create links b/w the objects.
(5) Create messages associated with each link
(6) Add sequence number to each message corresponding to the timely ordering of messages in the collaboration.

CD

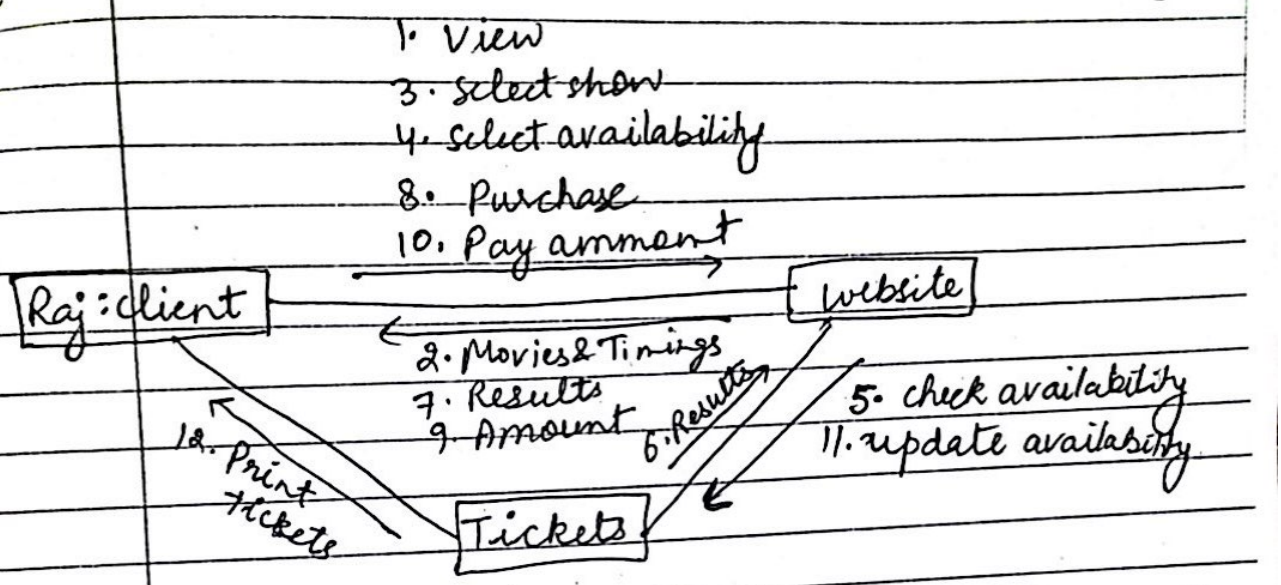
Date.....

Ques Draw a CD for online movie ticket booking

1. View
3. Select show
4. Select availability
8. Purchase
10. Payment

Date.....

Ques Draw a CD for online movie ticket booking



★ Sequence Diagram V/S Colaboration Diagram

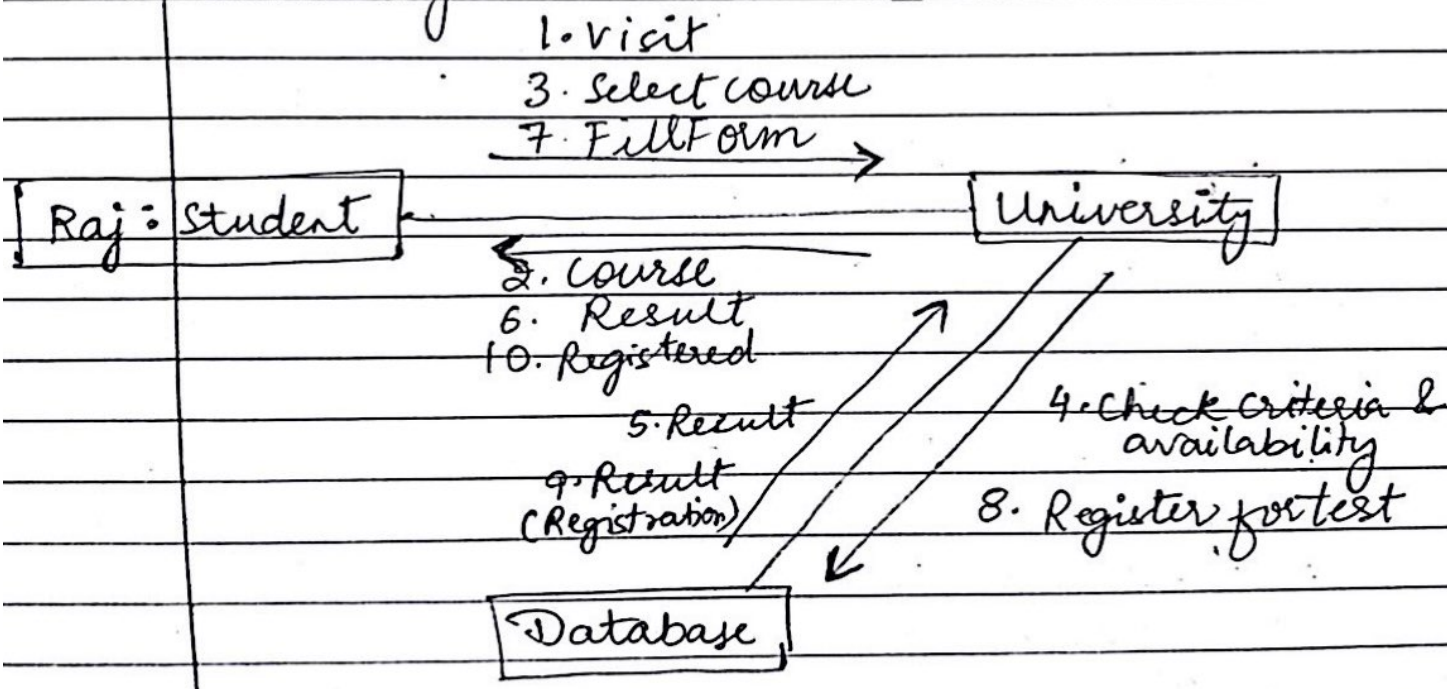
→ A SD shows synchronous as well as asynchronous message.
∞ A CD shows only synchronous message.
→ A SD shows overall system events in a given use-case.
A CD shows how objects interacts with each other
→ A SD is good for real time specification & complex scenarios.
A CD does not show time and the objects are arranged in a graph or network format.
→ In SD it is difficult to find responsibility of the object.
In CD it is easy to detect responsibility of the object.
→ SD are less spatial whereas CD are more spatial.
→ The main difference b/w SD & CD is that SD is time based while CD shows how objects are associated with each other.

Date.....

Collaboration Diagram:

CD is very similar to SD. In other words it shows dynamic interaction of the object in a system. A CD is easily represented by modelling objects in a system. CD are used to show how objects interact to perform the behaviour of a particular use-case. Unlike SD, CD show interaction among objects.

ues Draw a CD to book a course in a university.



Date.....

UP Artifacts:

An artifact is a unified modelling language. It is the specification of a physical piece of ion ie used or produced by a S/W development or by deployment and operation of a system. Example of artifacts includes model files, source files, scripts and binary executable files. A word processing document or a mail message. Artifacts are the physical entities that are deployed on nodes that is devices & execution environment. Other UML elements such as classes and components are first manifisted into artifacts and instances of these artifacts are then deployed.
Artifacts can also be composed of an another artifact.

Draw a ~~sequence~~ collaboration diagram for electriticity bill payment: