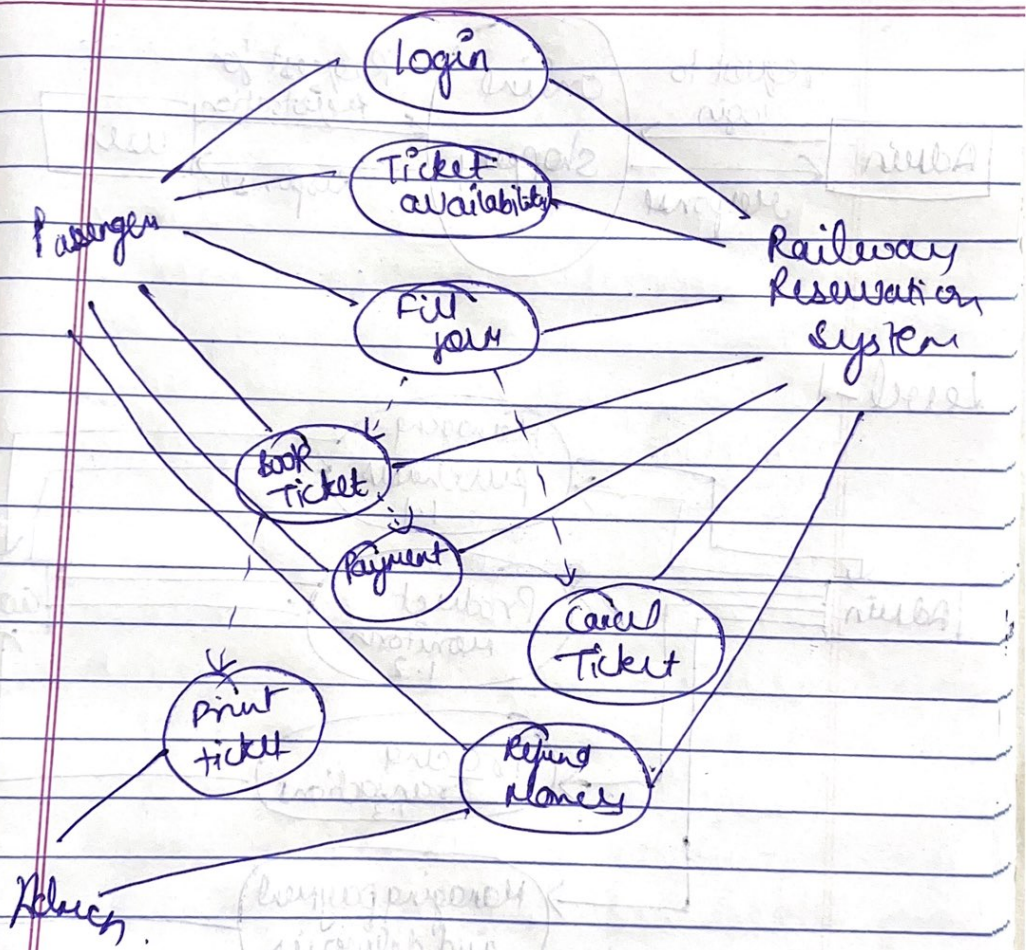
use case diag

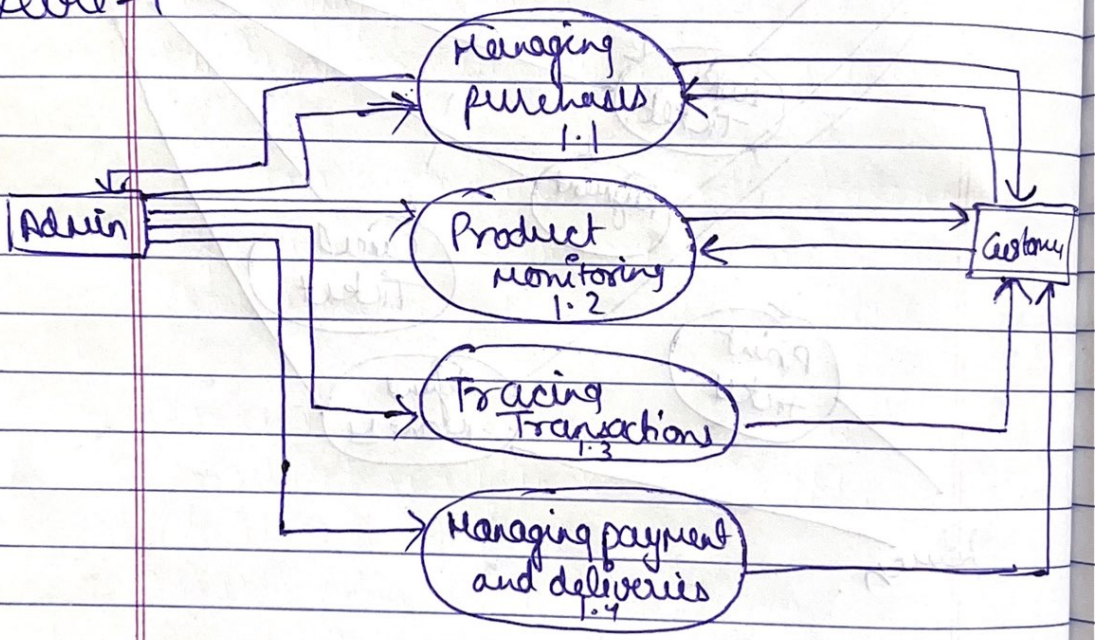
Date. ____
Page No. ____



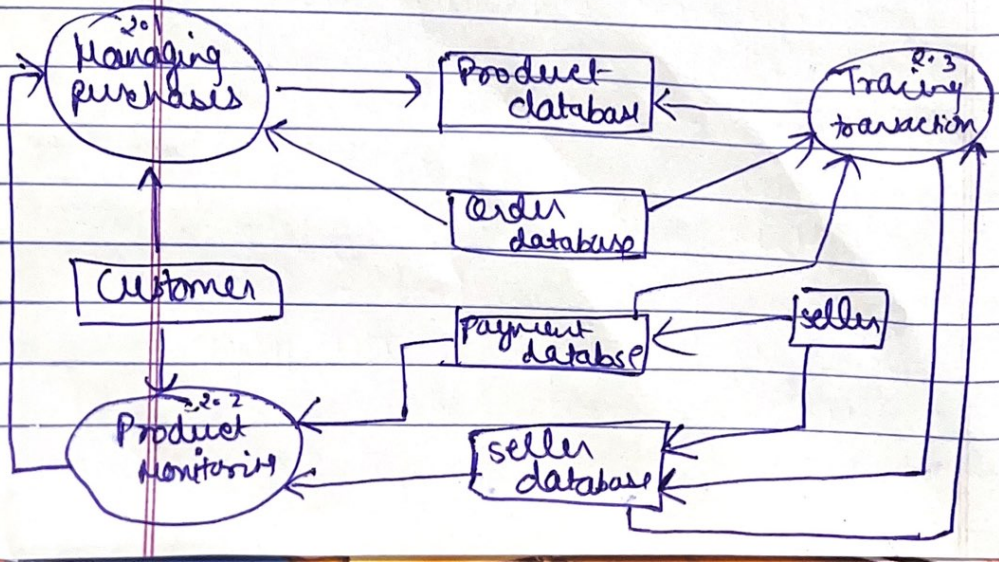
Date.
Page No.

Level-0

Level-1

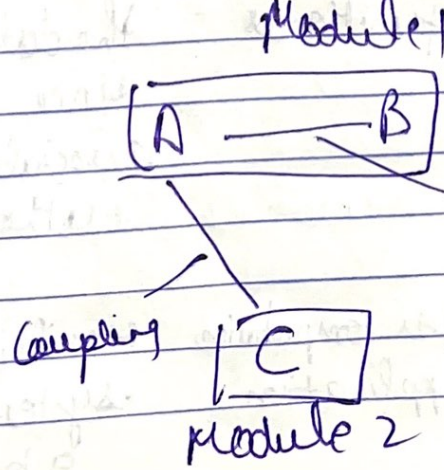


Level-2



Date. _____
Page No. _____

Cohesion	Coupling
concept of intra-module	concept of inter-module
represent relationship within a module	represent relationship blw modules.
↑ig cohesion is good for software	↓ow coupling is good avoided for software.
represents functional strength of modules	represents independence among modules.
Highly cohesive gives the best software	Loosely coupling gives the best software
Modules focus on single thing	modules are connected to other modules.



Date.
Page No.

Function Oriented Design	Object Oriented design
It is a top down approach	It is a bottom up approach
we decompose in function/procedure level	we decompose in class level
the state information is represented in centralized shared memory	the state information is not represented in centralized shared memory
carried out using Structured analysis & structured design i.e data flow	carried out using UML
Functions are group together by which a higher level function is obtained	Functions are grouped together on the basis of the data they operate since classes are associated with their method
mainly used for computation sensitive application	mainly used for evolving system which mimics a business or business case.

Date.
Page No.

Top Down Approach	Bottom-Up Approach
In this approach we focus on breaking up the problem into smaller parts	In this approach we solve smaller problems and integrate it as a whole and complete the solution
Mainly used by structured programming language, C, COBOL etc	mainly used by object oriented programming language, C++, C#
Each part is programmed separately ∴ contain redundancy	Redundancy is minimized by using data encapsulation & data hiding
communication is less in modules	Modules must have communication
It is used in debugging, module documentation etc	It is used in testing.
decomposition takes place	Composition takes place

Date.
Page No.

Abstraction

It simply means to hide the implementation details to reduce the complexity i.e. ~~its~~ the efficiency or quality. Different levels of abstraction are necessary and must be applied at each stage of its design process.
The solution should be described in broad ways that covers a wide range of different things at a higher level of abstraction and a more detailed description of a solution of software should be given at a lower level of abstraction.

Modularity

It simply means diving the system or project into smaller parts to reduce the complexity of the system or project.
These smaller parts can be created independently and than we use these part in different system to perform different function. Monolithic software is hard to grasp for software engineers. If we are able to divide the software into modules it is cheaper & easier to develop.

Date.
Page No.

Verification	Validation
includes checking documents, design, codes & programs	includes testing and validating actual product
It is static testing	It is dynamic testing
does not include the execution of code	includes the execution of code.
Methods used are - reviews, walkthrough, inspection & desk-checking	Methods used are Black Box Testing, White Box Testing & non-functional Testing
can find bugs at early stage of process	can only find bug which can not found by verification
goal: application & software architecture & specification	goal is an actual product
comes before validation	comes after verification
consist of checking of document files and is performed by human	consist of execution of program performed by computer
Done by Quality assurance	Done by Testing team.

Date.
Page No.

Program	Software
A computer program is a set of instruction that is used to create a software by using programming language	Software is a set of programs that enables the hardware to perform a specific task.
do not have further categorization	Three types: system, application & programming
does not have a user interface	provide user interface
size Kb to Mb	size Mb to Gb
take less time to develop	take more time to develop
less features & functionalities	more features & functionalities
lack of documentation	properly documented
development approach is unorganized, unplanned, unprocedural	organized, planned procedural
eg video games, malware	Adobe Photoshop, Google chrome.

Date.
Page No.

Classical waterfall model

Feasibility study → Requirement analysis → Design → Coding and unit testing → system testing and integration → Maintenance

cdw
base model
Simple & easy
small projects

disadv.
no feedback
no experiment
no parallelism
high risk
60% efforts maintenance.

Requirements Elicitation:

It is related to the various ways used to gain knowledge about the project domain and requirements. The various sources of domain knowledge include customers, business manuals, the existing software of same type, standards and other stakeholders of the project.

The techniques used for requirements elicitation include interviews, brainstorming, task analysis, Delphi technique, prototyping, etc.

1. Interviews:

Objective of conducting an interview is to understand the customer's expectations from the software.

It is impossible to interview every stakeholder hence representatives from groups are selected based on their expertise and credibility.

Interviews maybe be open-ended or structured.

1. In open-ended interviews there is no pre-set agenda. Context free questions may be asked to understand the problem.
2. In structured interview, agenda of fairly open questions is prepared. Sometimes a proper questionnaire is designed for the interview.

Brainstorming Sessions:

- It is a group technique
- It is intended to generate lots of new ideas hence providing a platform to share views
- A highly trained facilitator is required to handle group bias and group conflicts.
- Every idea is documented so that everyone can see it.
- Finally, a document is prepared which consists of the list of requirements and their priority if possible.

Prototyping

One of the most important phases of the requirements elicitation process, prototyping enables business owners and end-users to visualize realistic models of applications before they are finally developed. Prototyping helps generate early feedback, and it boosts stakeholder participation in requirements elicitation.