Date

SE
MID2 NOTES.

Syllabus:-

1. Delphi Techniques
2. Design Concepts
3. Architecture Design
4. UI Design

Delphi Techniques. (Estimating Time for completion of a software)

Brief:

Ye aik time estimate karne ki technique hai jismain har member ko input dene ka moka milta hai (chahe beginner ho ya expeienced ho). Aur initially sab ki id anonymous rehti hai take koi bhi kisi beginner ki estimation pe hase na. Aur is ke liye estimation forms ka istemaal kiya jata hai.

Let's Start:-

Estimation: Project manager must set expectations about the time required to complete the software.

The expectations of time must be realistic and practical. (Aisa na hoke 1 mahine ka project hai aur uske liye 10din estimate kiye hon)! ← Masla hojaega ☹

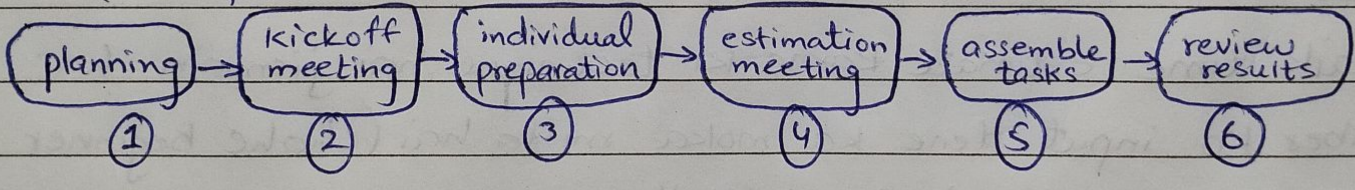
Date ____________

How to do a good/realistic estimate:-

For that the project manager must have

1. WBS (work break down structure)
2. Effort estimate for each task
3. List of related assumptions for project.
4. Consensus among the project team that the estimate is accurate. (Sub members agree hone chahiyen)

Wideband Delphi : (A process to generate an estimate)
(6 steps)



Bht easy, self-explanatory 6 steps hain Lkn phir bhi main inki description dedeta hoon apko for better understanding.

1. Planning. choose a 3-7 members project/estimation team
   Choose a moderator (moderator cannot take part in estimation, he'll only manage process)

* project manager should not be a moderator, he must take part in estimation.

2. Kickoff meeting: to make understand delphi process and project background and needs.

Date

The team : ① brainstorms and writes assumptions
② generates a WBS of 10-20 tasks
③ agrees on a unit for estimation (eg. days or weeks)

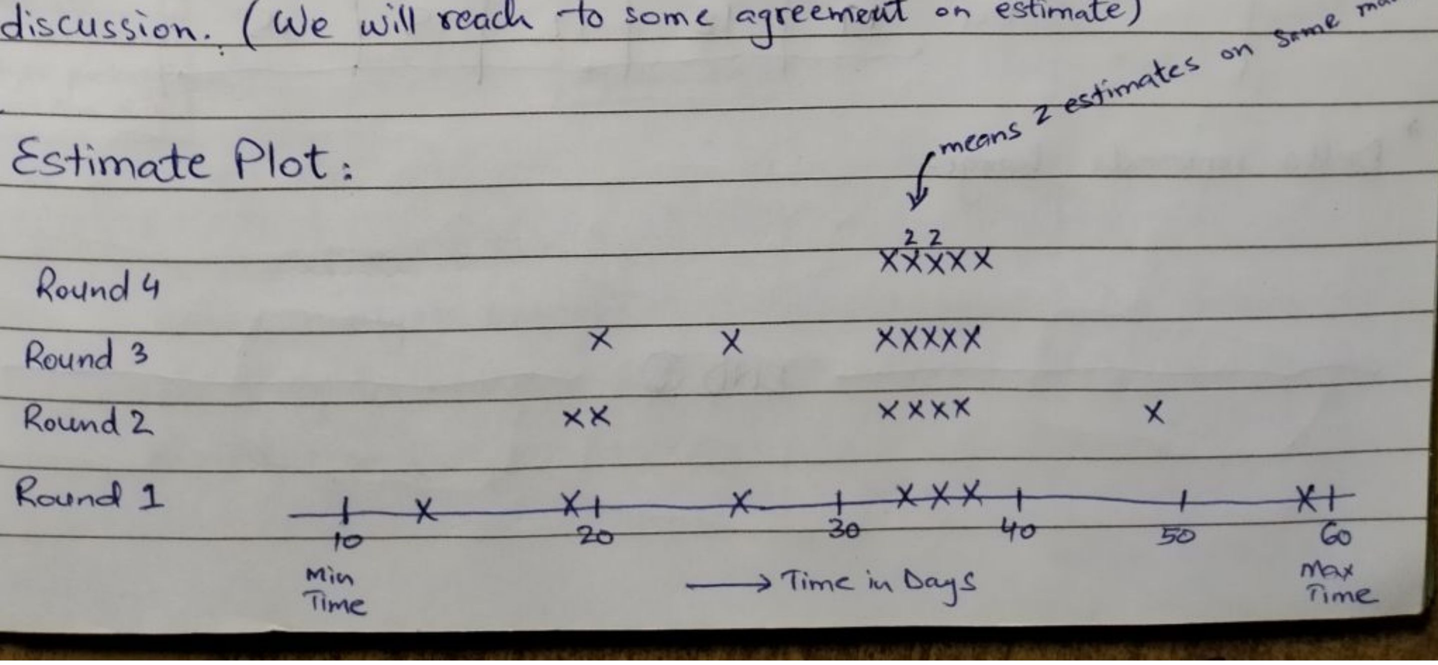
③ Individual Preparation: (Har member individually har task ke liye estimates lagata hai!)

④ Estimation Session: (3-4 revision sessions)

- each member fills out the estimation form
- moderator collects forms, and the outlier time (min time & max time) are discussed. (Jisne max estimate kiya hai, wo apne reasons batata hai ke kyoon itna zyada time estimate kiya aur min wala bhi)
- iterate this process 3-4 times (fill estimation form, discuss)
- revise estimations in each iteration considering the reasons given in discussion. (We will reach to some agreement on estimate)

Imp:

Estimate Plot:



⑤ Assemble Tasks:

Project Manager compiles the final task list, estimate, assumptions

⑥ Review Results:

Project Manager & estimate team reviews final task list.

	Task Name	Est	Delta 1	Delta 2	Delta 3	Delta 4
	⋮		⋮			⋮
1	Abc	3	+2	0	0	0
2	Efg	6	-2	0	+1	0
3	Xyz	10	-1	-1	-3	0
	⋮	⋮	⋮	⋮	⋮	⋮
	Delta	▨	-1	-1	-2	0
	Total	19	18	17	15	15

* Delta represents change.

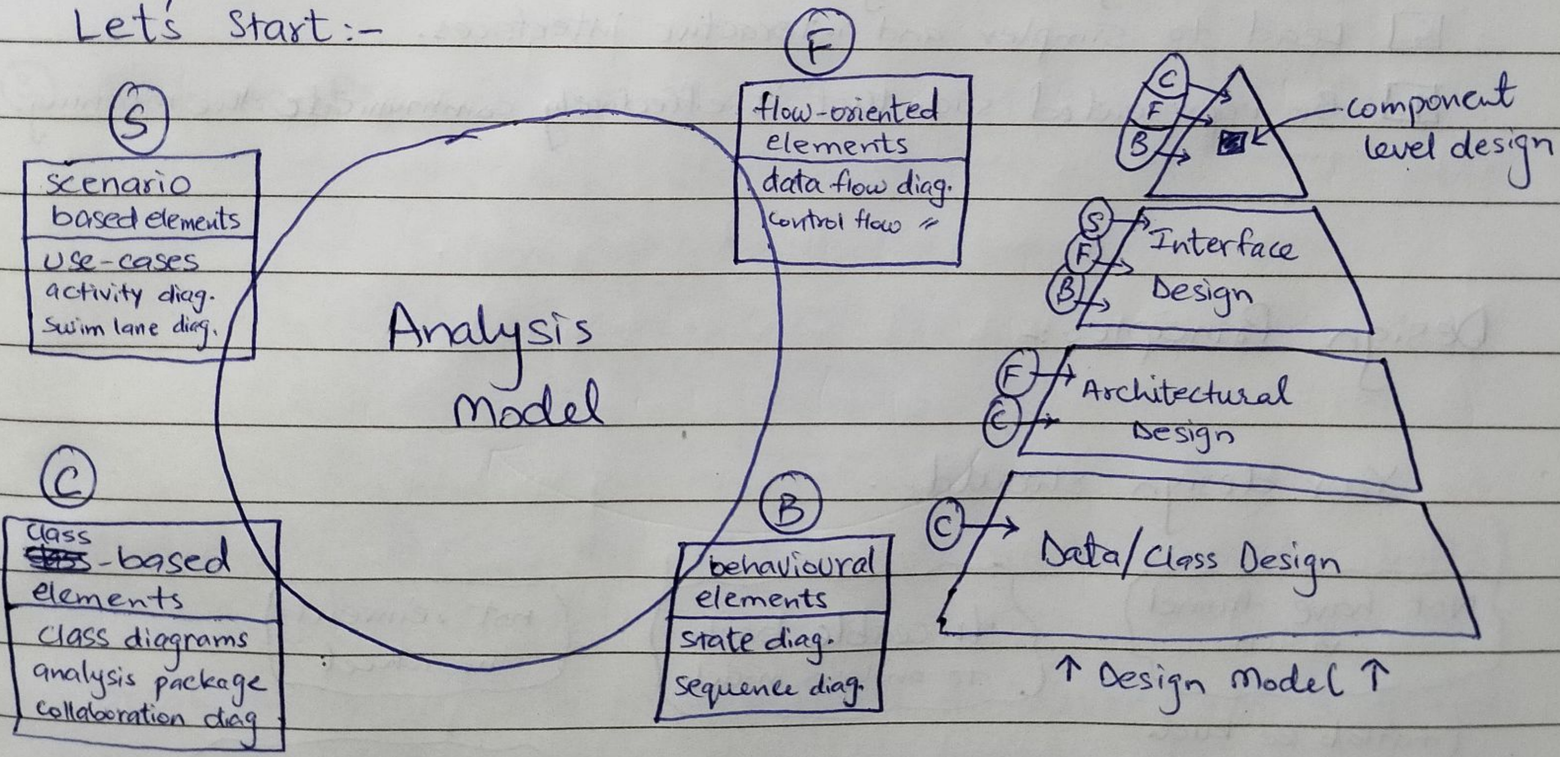
END ☺

Date ____________

Design Concepts

Brief: Zahir si baat hai aik acha software bana rahe hain to uski designing dhang se ki ho, it should have firmness (no bugs in functions), commodity (fulfil the purpose), delight (pleasurable experience ho [slate nahin banana hamein]) ☺ Design aur quality ke principles dekhen ge.

Let's Start:-



↑ Design Model ↑

↑ is model ko follow karke analysis karoge aur phir us analysis ki madad se ---

↑ --- is model ko follow karke design banao ge.

KESA DIYA?

Date

* Quality Guidelines:-

Your design should...

1- Exhibit an architecture

2- Be Modular (module mattlab chote chote tukre)

3- Contain distinct representation of data, architecture, interface and components.

4- Lead to appropriate data structures for the classes

5- Independantly functioning components

6- Lead to simpler and interactive interfaces.

7- Be represented such that it effectively communicate the meaning.

Design Principles:-

Your design should...

Not have tunnel vision

(mattlab ke kuch elements ko consider kar rahe ho doosron ko nahin)

traceable back to analysis model

not reinvent the wheel

exhibit uniformity

be close to real-world problem

be able to accomodate change

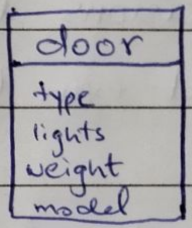
design is not coding and vice versa.

be accessed for quality while creation not after it!

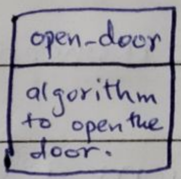
minimize errors

Fundamental concepts: (you would be familiar to most of them if you cleared OOP ☺)
(Total 10)

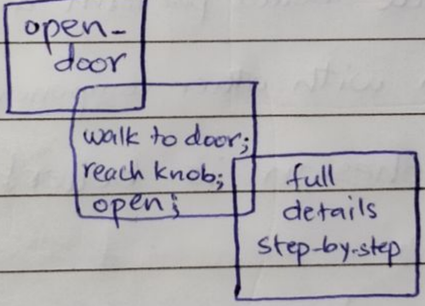
① Data Abstraction: (hide the details of a thing e.g door)



② Procedural Abstraction:
(only show the procedure name like "open door"
hide the details like "the steps taken to open the door)



③ Stepwise Refinement:



④ Modularity: (Break the software into smaller manageable chunks)

Break the design into many modules for easier understand and implementation. So that cost could be reduced.

* Use your brain to decide the optimal no. of modules. If so many modules are made then integrating all of them will become a big overhead so decide on an average no.

⑤ Separation of concerns: (Separate the features or behaviours that are different and independant. Work on them easily now.)

Date ____________

⑥ Information Hiding:-

Client ko sirf workable interface do, baaki information chupa ke rakho (its the secret of your firm), jeseke kya design decisions liye gaye hain, resources ko allocate kese kiya hai and even the whole code.

⑦ Functional Independance:

→ two concepts cohesion and coupling.

Cohesion: a cohesive module should perform a single task and require least interaction with other components.

(So, High Cohesion is better!)

Coupling: relative interdependance among modules.

(jitni interdependancies zyada hongi system banana ~~to~~ utna hi complex hojaega!)

(So, LOOSE COUPLING is better!)

⑧ Refactoring: Changing the software system to improve its internal structure but not altering its external behaviour.

YAAD RAKHNA!

control class, entity aur boundary classes se communicate karti hai, wo dono aapas main communicate nhn karsakte!

Date

⑨ OO Design concepts

- Design classes (ECB) ← entity, control, boundary classes.
- Inheritance
- Messages
- Polymorphism (a characteristic that greatly reduces the effort required to extend the design)

⑩ Design Classes:-

- analysis classes refined to make entity classes.
- boundary classes developed to create the interface.
- controller classes designed to manage
  1. creation and updation of entity
  2. instantiate boundary objects as they obtain info from entity objects.
  3. communication b/w these objects and validate data.
  4.

END ☺

Date

Architectural Design

Brief: Architecture aapke poore software system ka aik fundamental design hota hai jo batata hai apne kya elements daale hain aur wo aapas main kese related hain, unmain functions konse hain, basically it is the big picture of your system.
Hum dekhen ge ke kese architecture select karte hain, unke patterns, unke views aur unke features.

Let's start!

Architecture "in the small" is concerned with architecture of individual programs and how it is decomposed into further components. Eg. Function call in C program.

and "in the large" it is concerned with the architecture of bigger complex enterprise systems that include further systems.

We use architectural models to facilitate discussion about the system design with the stakeholders and team.
(kyoonke usmai bht zyada details nahin hotin!)
aur poora system bhi show horaha hota hai...

Date ____________

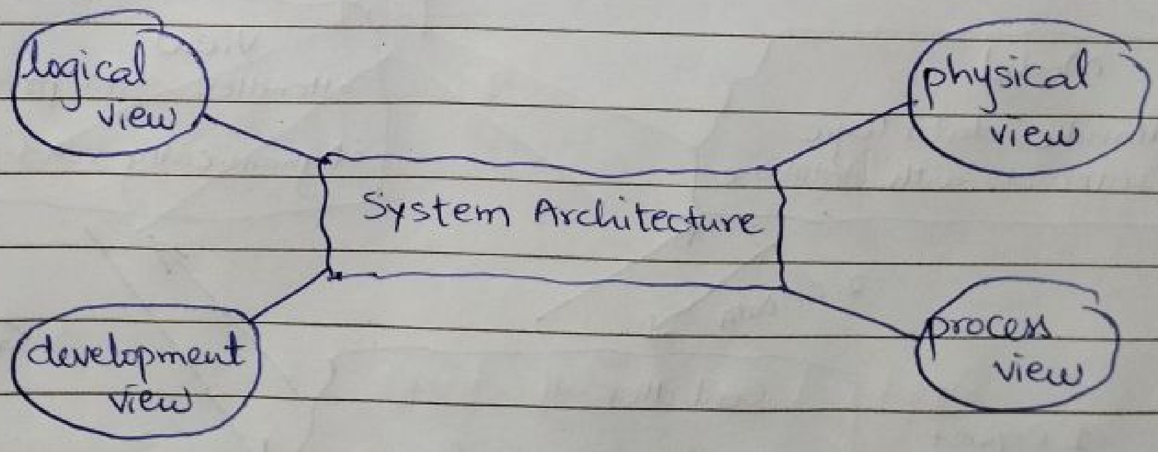
① Architecture design Desicions:-

- * performance (localize critical operations, minimize communication
- * security (layered architecture) * availability (fault tolerant)
- * safety (safety-critical features) * maintainability (use fine-grain and replacable component

② Architectural views:-

Imp

* each architectural view shows only one view or perspective of the system.



↑ 4+1 view model

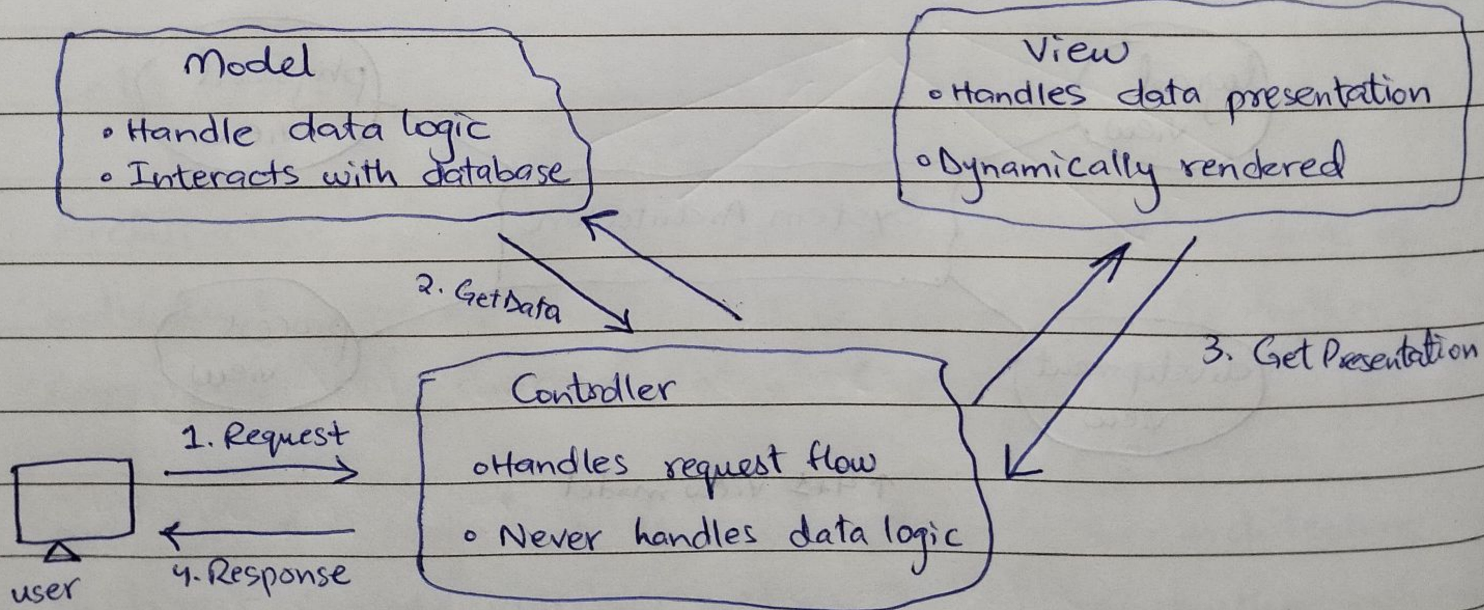
Date

③ Architectural patterns :-

→ Patterns are a means of representing, sharing and reusing knowledge.

→ should include info. of when they are and when they are not useful.

① MVC pattern: (Model, View, Controller)



e.g. you want some cat data, so you request to controller, it gets the data from Model, it gets the layout (HTTP+CSS) of how to present cats, from View and then responds to the user. Happy Life ☺

Date

② Repository Architecture:

→ Data is stored in a central shared repository

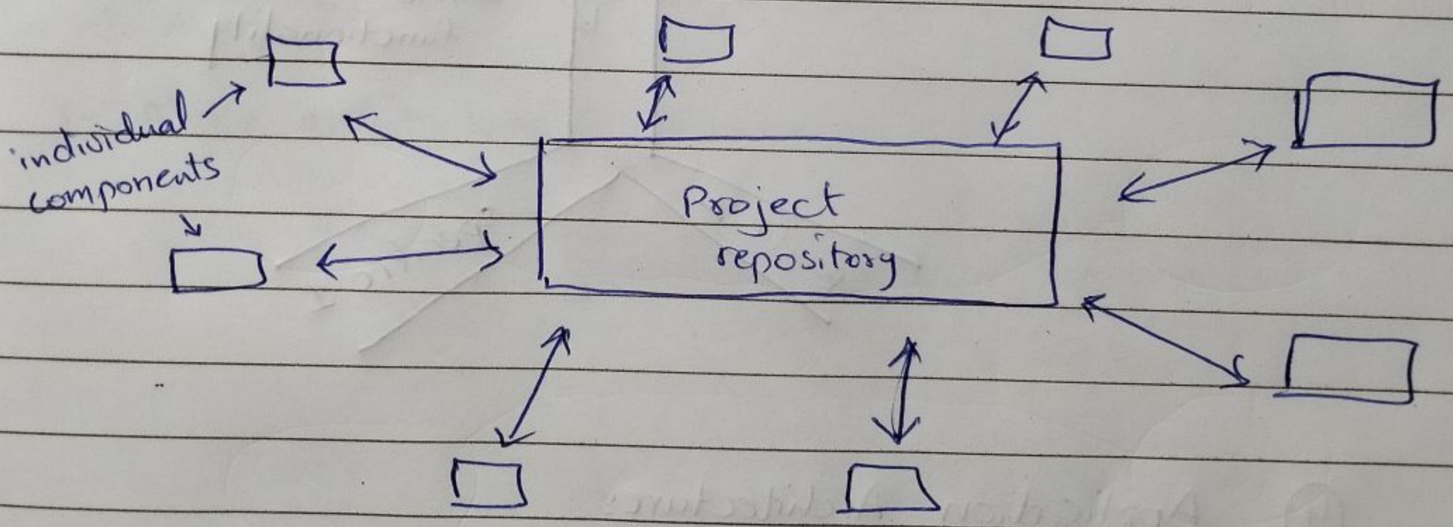
→ Components interact through repository only. (so consistent)

Adv:

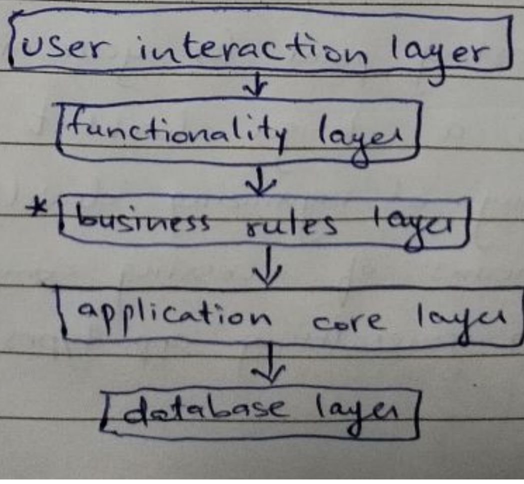
→ Saare components individually independant honge

→ aur aapas main communicate nhn karenge (to no overhead)

→ aur data bhi consistent rahega)



Layered Architecture:



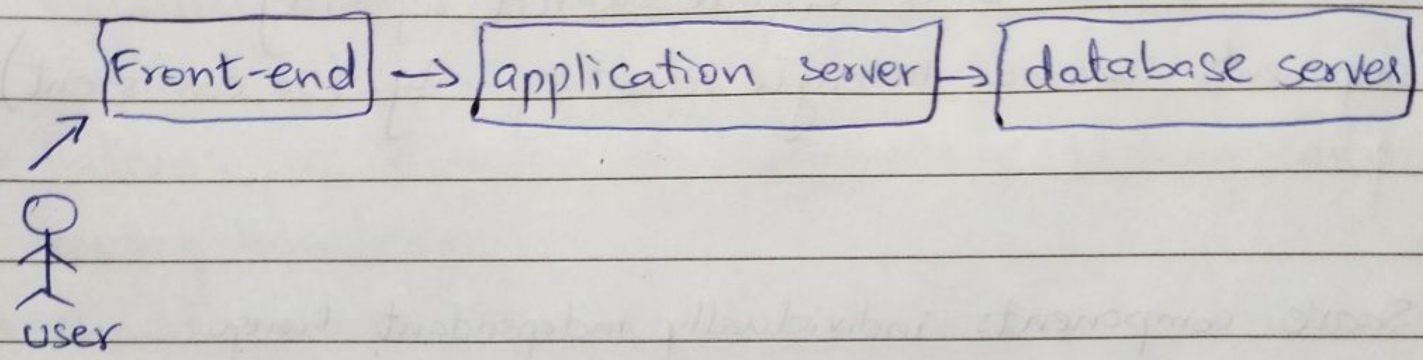
* rules set for your system to work

explanation:-

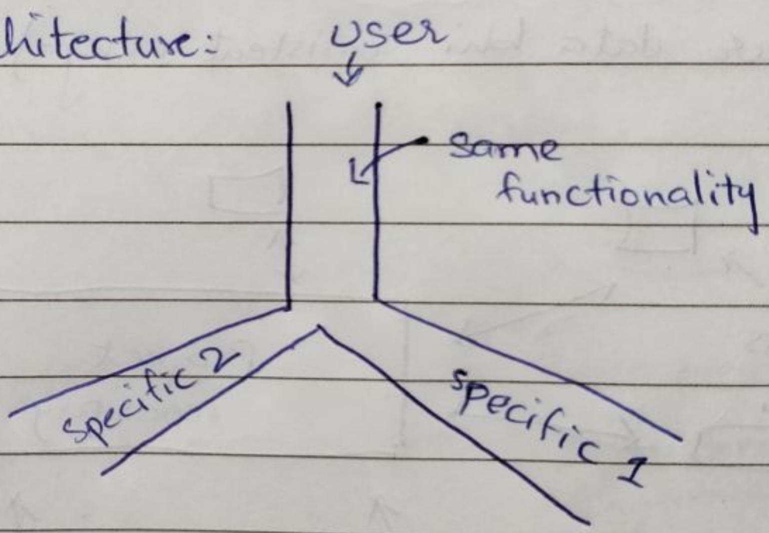
user ↓ interface ↓ functions ↓ behaviour ↓ programming layer ↓ data

Date ________

Client-server architecture :-



Pipe and Filter Architecture:



④ Application Architectures

why use application architectures?

1- starting point for your architecture

2- as a design checklist

3- way of organizing development work.

4- means of accessing components for reuse.

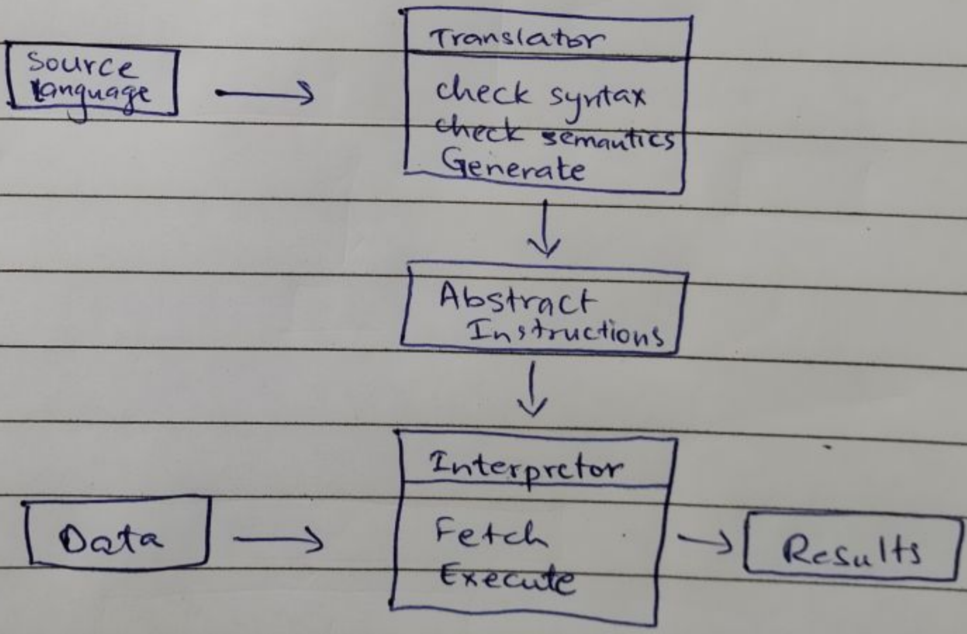
5- for generalising app. types.

Examples of Application Types:-

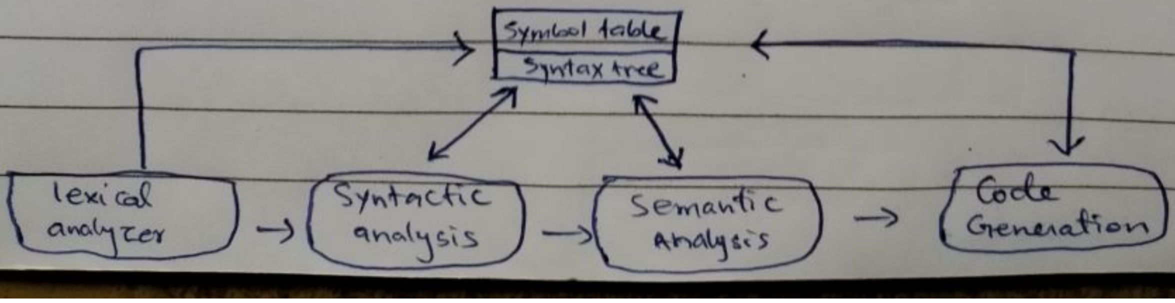
- Data processing Applications
- Transaction 〃 〃 (eg. Ecommerce systems)
- Event 〃 〃
- Language 〃 〃 (eg. Compilers)

Real World Example:- (Compiler)

1- Make architecture for compiler:-



2. Make pipe & filter arch. for compiler:


Symbol table
Syntax tree
lexical analyzer
Syntactic analysis
Semantic Analysis
Code Generation