LLD - 2

UML diagrams

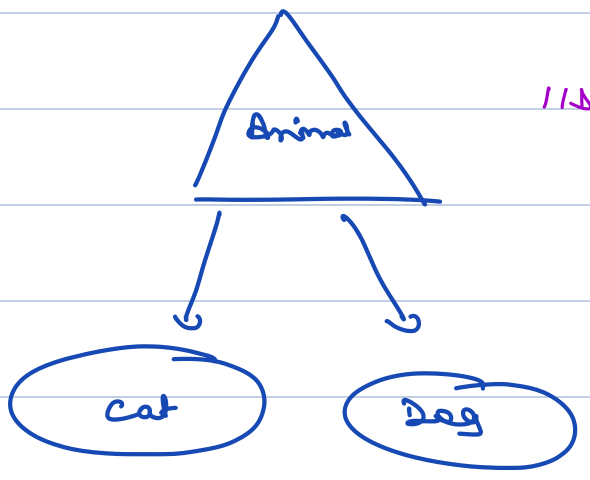
→ Use case Diagram
→ class diagram

/ product
/ project manager } understand reqn demos
clients
Team Leads } growth + day to day job
Architects } design Approval
Business stakeholders } req, demos

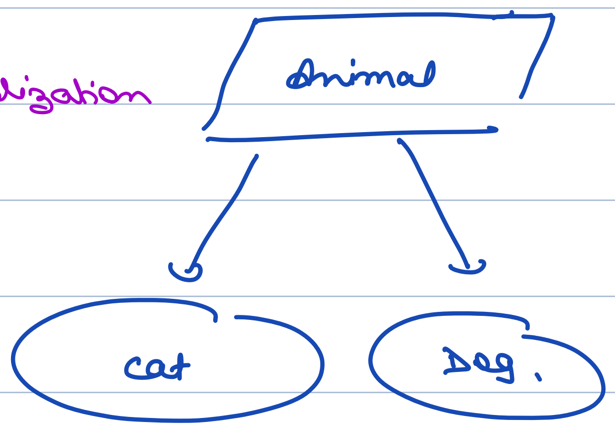
ways

① words → emails → meeting
1) misunderstanding
2) Ambiguity.

flowcharts / diagrams / images.



//Standardization



→ standard to represent diff. s/w concepts. in a diagram.

UML (Unified Modelling Lang).

$$\begin{tikzcd} & \text{UML} \arrow[dl, bend right] \arrow[dr, bend left] & \\ \text{Structural} & & \text{Behavioural} \end{tikzcd}$$

Structural	Behavioural
Deals with structure of your code base.	working of system & its features.
→ Class diagram	→ use-case
→ Package diagram	→ Activity
→ Component diag.	→ sequence
	→ State.

use Case - Diagram

it tells what are features and functionalities → diff features / functionalities

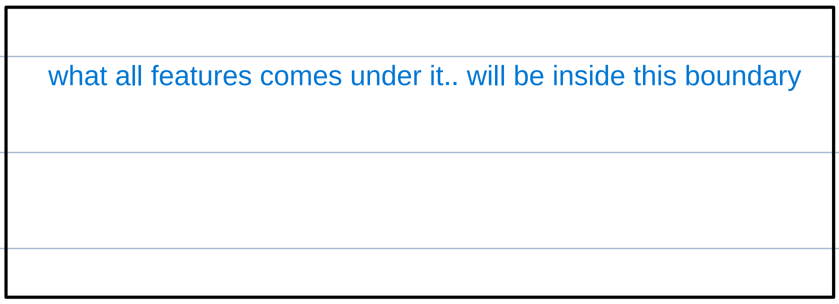
→ who are the users.

5 keywords

keyword ① System Boundary.

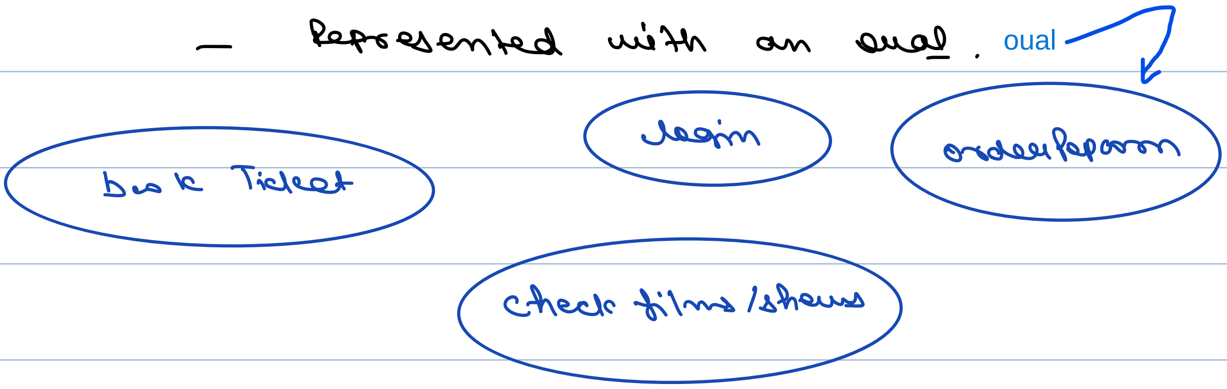
→ represents scope of your system

→ It doesn't include outside feature


what all features comes under it.. will be inside this boundary

keyword ⓘⓘ Use - Case

- features / Actions
- must always be a verb
- Represented with an oval. oual



raise an HR
Request Recording

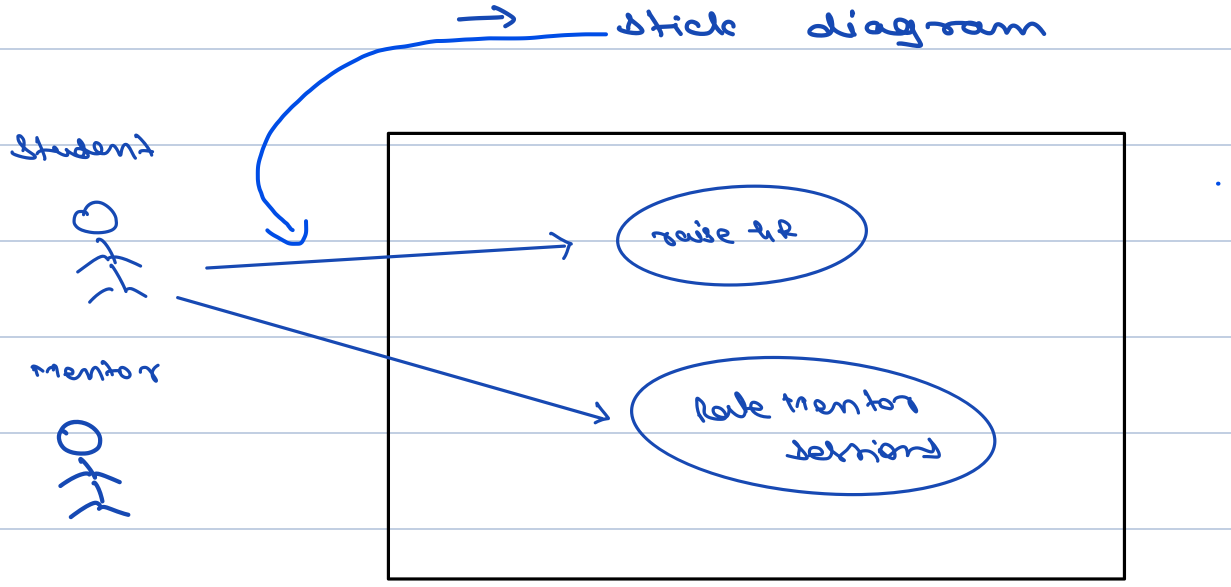
keyword

(III) Actors

→ People who use a particular use - case.

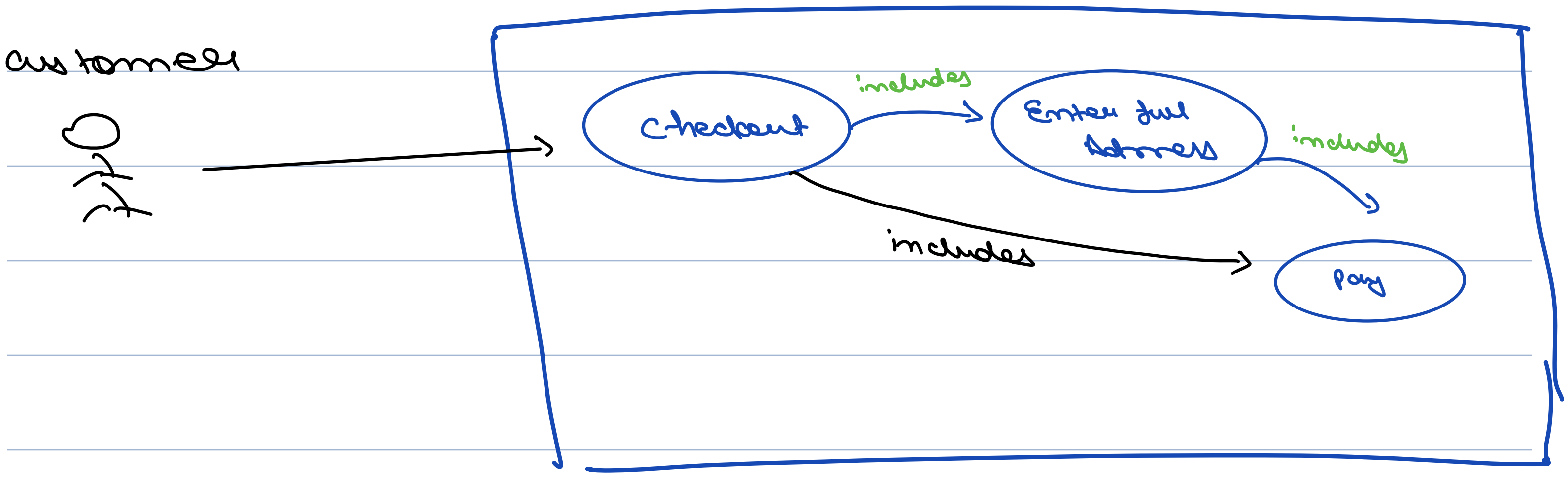
→ must be noun

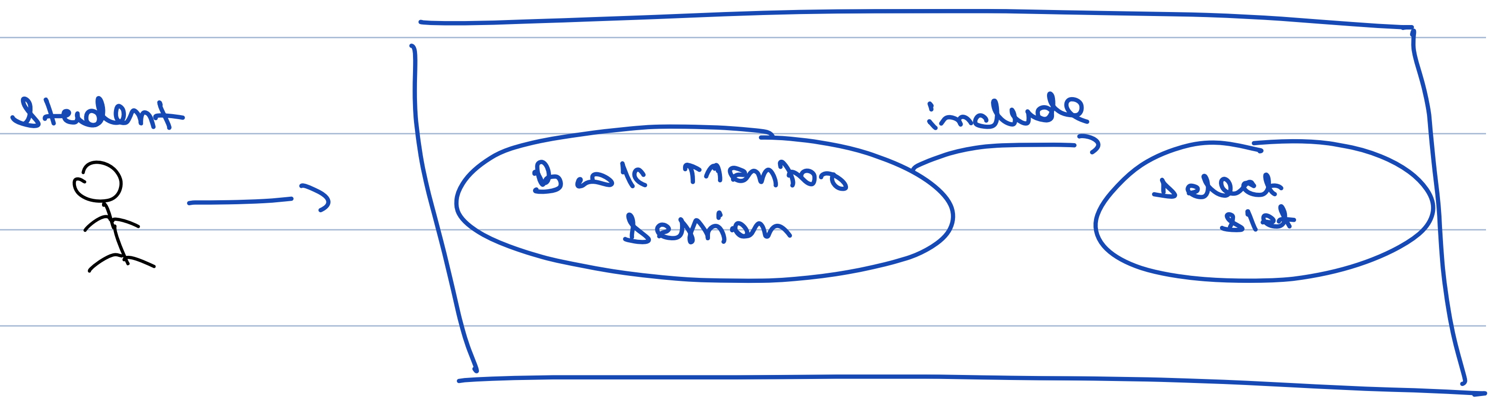
→ stick diagram



keyword

(IV) Includes

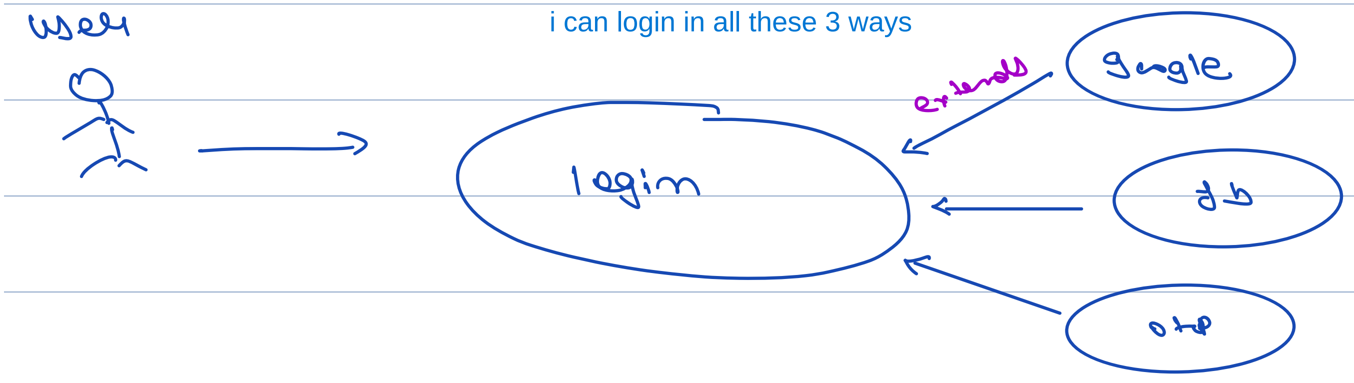


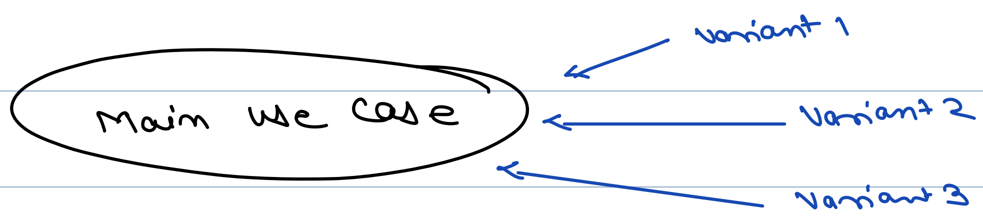


keyword

(V) Extends

→ if one feature has multiple variants.





Draw a use-case Diagram

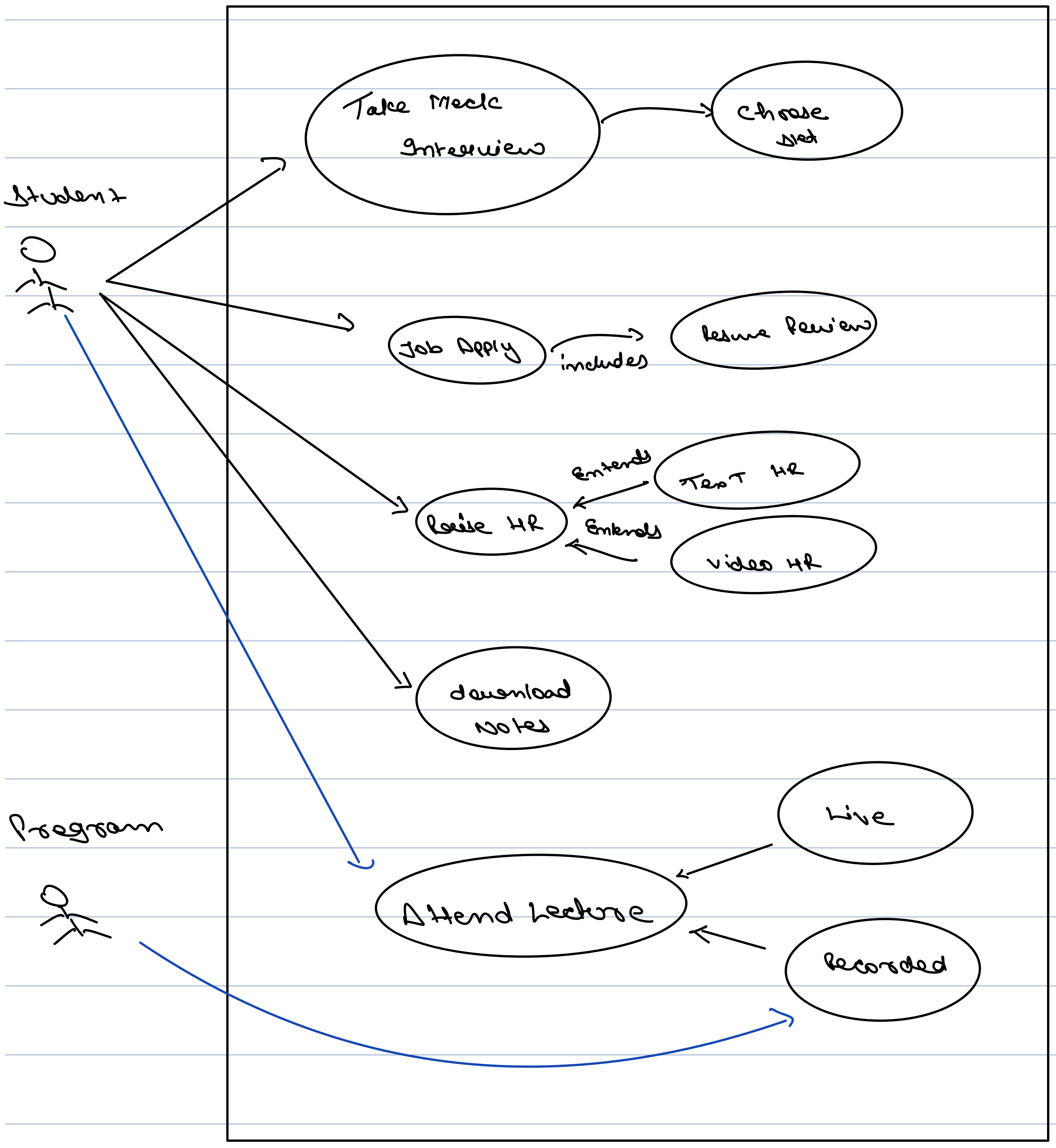
1) 5 use-cases

2) 2 Actors

3) 1 use case: includes

4) 1 use case: Extends

9:41pm – 9:47pm



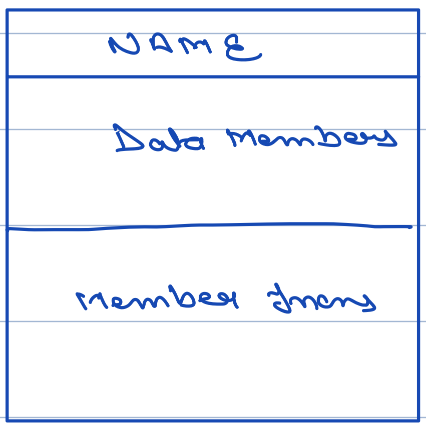
Student
Program
Take Mock Interview
Choose slot
Job Apply
includes
Resume Review
extends
Test HR
Raise HR
Extends
Video HR
download Notes
Live
Attend Lecture
Recorded

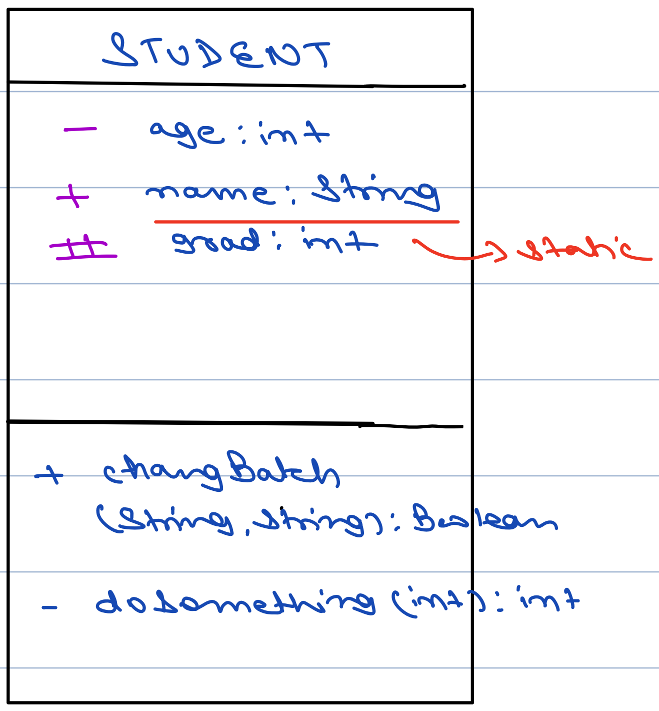
Class Diagram

- → represent diff entities.
  - → class
  - → Interface
  - → Abstract class
  - → Enums.

- → represent reln b/w the entities.
  - → implements interface
  - → extends
  - → having another class as an attribute.

class

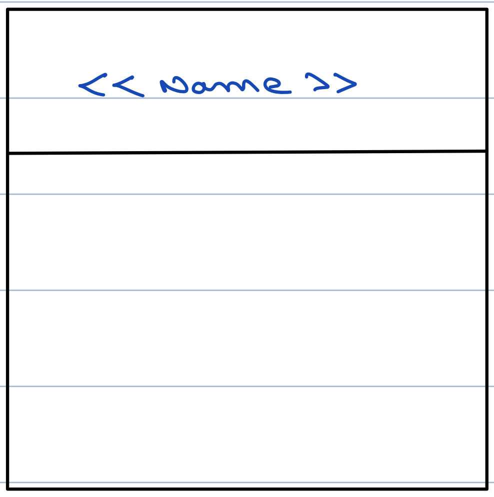



STUDENT
– age : int
+ name : String
# grad : int
static
+ changBatch (String, String) : Boolean
- doSomething (int) : int

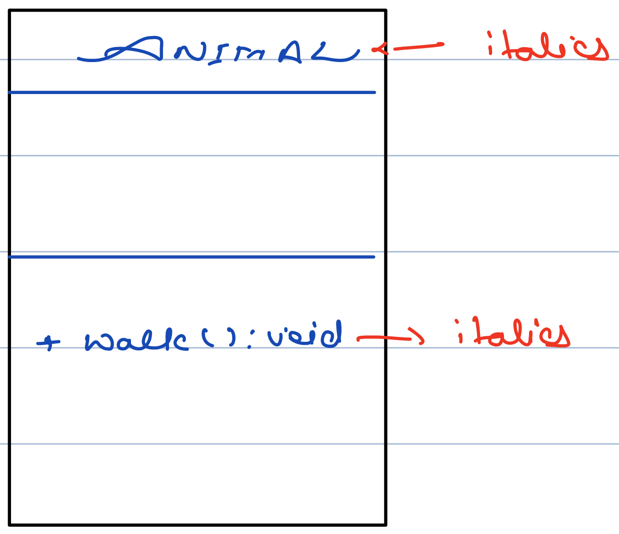
access modifier name : datatype

public → +
Protected → #
Private → -

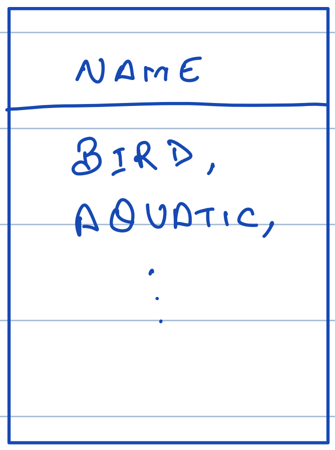
Interfaces



Anything Abstract italics



Enums



Relationship

1) is A

(Extends or implements)

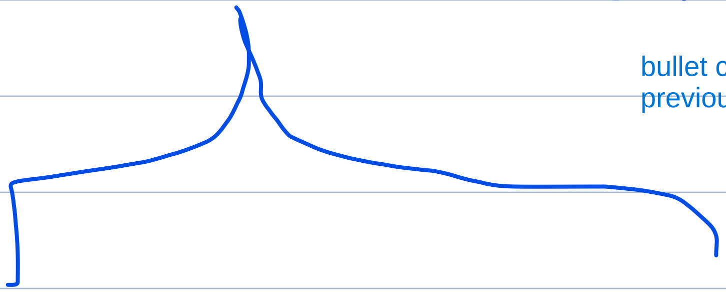
$$\begin{tikzcd} \boxed{\text{Parent}} & \boxed{\text{child}} \arrow[l] \end{tikzcd}$$

$$\begin{tikzcd} \text{Animal} & \text{<<flyable>>} \\ \text{Dog} \arrow[u] & \text{Sparrow} \arrow[u] \end{tikzcd}$$

2) Association : having another class as an attribute.

(has a)

bullet class and flying bullet class ka e.g. just in previous session



Aggregation

-week association
-Contained entity can exist indendently

e.g.
```
car{

driver d;    if this is another class
             data member

}
```

if car class is deleted then driver can still exist

e.g. if in any theatre any show cancelled.. it doesn't mean movie doesn't exist

Composition

if outer thing looses then inner part will lost its existence

-Strong association

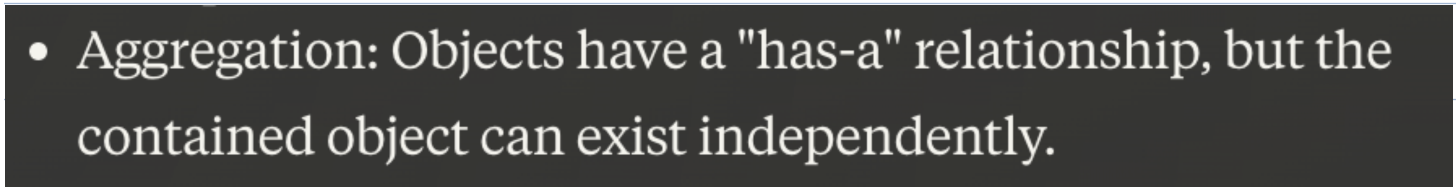
e.g.
```
post{

comments c; its this class object

}
```

now here if post losses then comment will lost its existence

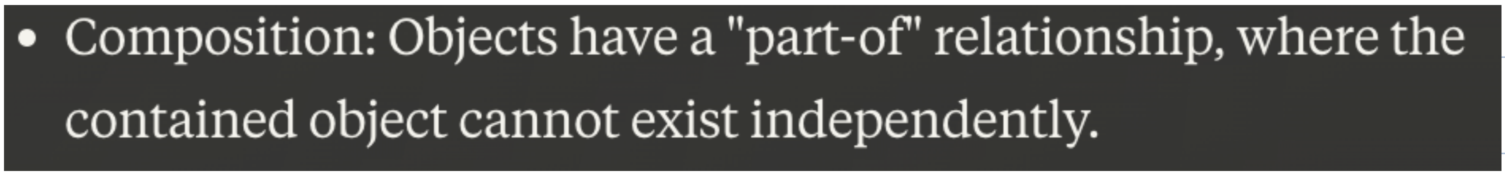
comments can not survive independently

but here if show got cancelled then ticket also doesn't have any existence

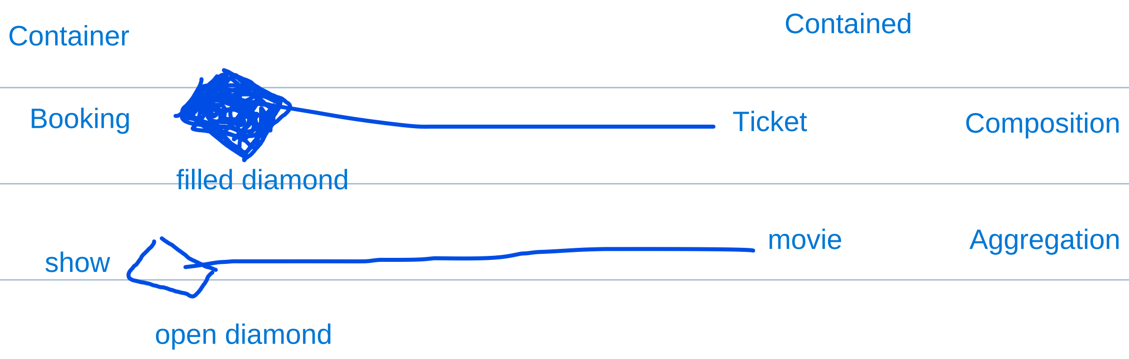
its not like a class can be only aggregation or composition.. mostly its both



- Aggregation: Objects have a "has-a" relationship, but the contained object can exist independently.



- Composition: Objects have a "part-of" relationship, where the contained object cannot exist independently.


Container
Contained
Booking
Ticket
Composition
filled diamond
show
movie
Aggregation
open diamond