Software Architecture and design Pattern

* Essential Element —

(I) Pattern Name —

(II) ~~Re~~ Problem solve

(III) Solution Efficient soln.

* Pattern কখনো definite solution দিবে না) core solution দিবে, কিন্তু কখনো Specific solution দেয় না

* (IV) Consequences.

* Design Pattern in mvc. / ~~তিনটা~~ design pattern use ক

* Pattern — ADB

(I) Pattern name
(II) ~~Intent~~
(III) Also known as
(IV) Motivation
(V) Applicability
(VI) Structure
(VII) Participants
(VIII) collaborations
(IX) Consequences
(X) Implementation
(XI) sample code
(XII) Related Patterns.

* Types of design pattern ——

Creational	Structural (object + class composition)	Behavioral (communication)
→ factory (*)	→ Adapter (*)	→ Interpreter (*)
→ abstract (*)	→ Bridge	→ Template method
→ Builder	→ composite	→ Chain of responsibility
→ Single Item (*)	→ Decorator (*)	→ Iterator
	→ Facade	→ Mediator
	→ Fly weight	→ Observer (* * *)
	→ Proxy	→ State
		→ Strategy
		→ Visitor

* Design pattern এর useful

(I) Finding appropiate objects

(II) Determining object granularity. (object এর scope কতটা করবেন হবে) (বিভিন্ন ভাবে ভাগ করা)

Person → user → student, Teacher, Common user

obstarct class, Interface.

* Specifying object Interface —

object type
object signature

```
Public Interface student{
    Public void setname();      } signature
    Public string getpassword();
}

Public class studentModelDAO implements student{
    Public void setname(){
}}
```

(object Implementation)
class → studentModel DAO (DAO → Data Access object)
Type → student (Interface name)
signature → method definations

* Program to an interface not to an implementation.

* common causes of redesign —

(I) Specifying a class explicitly while creating an object
(II) Dependence on specific operation
(III) Dependence on hardware and software platform.

(iv) Dependence on object representation and implementation.

(v) Algorithm dependencies.

(vi) Tight coupling (একটা class এ change করলে connected সব class এ change করতে হয়)

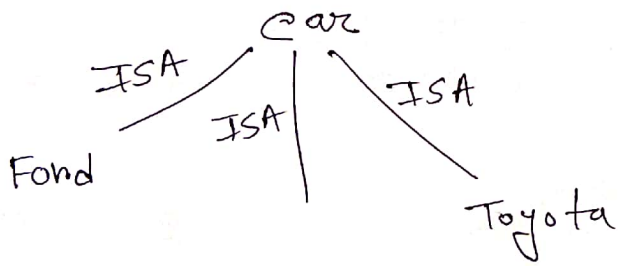
(vii) Extending functionality by subclassing.

(viii) inability to after classes conveniently

Inheritence question —

(i) Is this class a class B?

IS A



(ii) has A. object composition

```
class Institution
    Name;
    Add;
}
```

```
class studen {
    Institution institution;
    institution.name;
    institution.add;
}
```

object composition, delegation.

```
class Rectangle{
    int height, width;
    int getArea(){
    return height * width;
    }
}
class window{
 Rectangle ret;
 int getArea(){
  return rect.getArea();
 }
}
```

```
Public class Tester{
Public static void main(String args[]){
 window window = new window();
 window.ret.height = 10;
 window.ret.width = 10;
 S.O.P (window.getArea());
}
}
```

+92 884 2626

1. software install करना
2. Repository add करना
3. File create, delete, open, copy, remove, rename
3. Folder " " " " or switch, copy

pwd → print working directory.

cp -b /puja 2 Downloads

sudo → admin

apt → advanced packaging tool

get install codeBlocks

apt list | grep

* rename → mv

mv 51

* ls -l → permission list directory

ad min group user

* ~~Chm~~ chmod +x filename command modification
* chmod -x filename

Override — Runtime

Overload — Compile time.

* OPP কাজ করতে SOLID principle maintain করে।

S . O . L . I . D → Principle

- S → Single Responsibility Principle.
- O → Open Close Principle
- L → Liskov Substitution method
- I → Interface Segregation Principle
- D → dependency Inversion principle

Behavioural method →

Template method.

~~Interpreter method.~~

Strategy pattern

Observer ~~Template~~ pattern.

creational
(~~Behavior~~ Pattern)

observer / pattern —

* Single tone class use করা

Static

* creational Pattern

① single ton ~~Parrentn~~. Pattern.

② Factory pattern

↓

এটা fully SOLID কে follow করবে না।

imp → Id device (get button)

iOS