SOLID principle:

1. Single responsibility principle
   1. There should never be more than one single reason for a class to change
      1. Means focused single functionality, address specific concern
2. Open Closed principle
   1. Software entities (classes, modules, methods) should be open for extension but closed to modification
3. Liskov substitutive principle
   1. We should be substitute base class objects with child class objects and this should not alter the behavior of the program
4. Interface Segregation principle
   1. Clients should not be forced to depend upon interfaces that they don’t use
      1. Interface pollution
         1. Large interfaces with unrelated methods
         2. Signs of pollution
            1. Classes have emty method implementation
            2. Method implementation throws unsupportedOperation Exception
5. Dependency inversion principle
   1. High level modules should not depend upon low level modules both should depends on abstractions
   2. Abstractions should not depend on details. Details should depends on abstractions

Types

1. Creational
   1. Builder
      1. Will solve the problem of immutable classes with desired constructors. Objects that needs other objects or parts to construct them
      2. If we have complex process to construct an object having multiple steps then builder pattern will help us
      3. In builder we remove the logic related to object construction from client code and abstract it in other class
      4. Diagram

         Description automatically generated with low confidence
      5. Steps of builder implementation
         1. We start by creating builder
            1. Identify the parts of the product and provide methods to those parts
            2. It should provide method to assemble the product/object
            3. It must provide a way/method to get fully build object out. Optionally builder can keep reference to an product it has to build so the same can be return again in future
            4. A director can be separate class or client can play the role of director
      6. Diagram

         Description automatically generated
      7. String builder is partial example for build and Calender in java8 is full
   2. Simple Factory
   3. Factory method
   4. Prototype
   5. Abstract Factory
   6. singleton
2. Structural
3. Behavioral