Term Paper
Annapurne Krishna
Roll No: 180101009

Generic introduction of the Design Patterns:

Design patterns are the excellent practices used in software development that offer answers to issues confronted with the aid of using software developers. They are fashions of code that clear up a number of the traditional issues. They are acquired with the aid of using trial and error with the aid of using numerous software developers over a protracted time, which saves effort and time all th tharough the implementation of the code. Design patterns can accelerate the improvement procedure with the aid of using imparting tested, established improvement paradigms. Design patterns are language-neutral, so that they may be carried out to any language that helps object orientation. They are to be used only whilst it's necessary. It is crucial to understand diverse design patterns due to the factt each pattern may be used for a particular situation.

The main three types of design patterns are-

1. Creational Patterns: These design patterns offer a manner to create objects whilst hiding the creation logic, as opposed to instantiating objects without delay with the use of a brand new operator. This offers this system extra flexibility in determining which objects are required to be created for a given use case.
Following are creational design patterns:
Abstract Factory, Builder, Factory Method, Object Pool, Prototype, Singleton

2. Structural Patterns: It provides a way to outline relationships between classes or objects. These design patterns are related to class and object composition. Inheritance is used to compose interfaces and outline approaches to compose objects to attain new functionalities. They arrange different classes and objects to shape large structures.
Following are structural design patterns:
Adapter, Bridge, Composite, Decorator, Facade, Flyweight, Proxy

3. Behavioral Patterns: These design patterns are in particular involved with communication or interplay between objects.
Chain of Responsibility, Command, Interpreter, Iterator, Mediator, Memento, Observer, State, Strategy, Template, Method, Visitor

Design patterns useful in my project(Inter IIT Tournament management system)

1. Creational:

a. Singleton: In this pattern, we ensure that each class has a single instance that may be accessed with the use of a global point. We have three types of users in our design - participants, team manager, and admin. According to the singleton pattern, only one object of a particular class is ever created and since there can exist only one "admin" who has access to all the features of the app, it is a singleton pattern.

2. Structural:

a. Facade: The homepage of the design acts as a unified interface to a set of interfaces in a subsystem. It is a higher-level interface that makes the subsystem easy to use. From the homepage, users can access all the main subsystems like viewing the leaderboard, viewing match results, and viewing the schedule of events. This pattern hides the complexities of the system and provides a simpler interface to users.

3. Behavioral:

a. Observer: The observer pattern is used to permit an item to put up adjustments to its state.
We have a feature for users to bookmark sports, so whenever there is a change in the event of the particular sport, the participant is notified automatically. Whenever the admin makes any changes in an event like the timing or venue of an event, the observer is notified of those changes. The participants act as observers here.

b. Command: In this sample, we encapsulate a command request as an object. The command sample is used to explicit a request, consisting of the decision to be made and all of its required parameters, in a command object.
The user commands the interface to give him event details, match results, or leaderboard. The interface then passes the request to the database. The interface acts as an invoker here and the database as the receiver of the command. The user is then provided with the results according to the request.