**SDA ASSIGNMENT # 3**

**K20-1052**

**BSE-4B**

**ARCHITECTURE PATTERNS**

Architecture patterns provide a reusable solution to the commonly encountered structural design problems like how the different layers within the software should interact with each other. The software development process can get complicated if the correct architectural pattern is not used.

**1- MVC (Model View Controller) Architecture**

It has three components Model, View and Controller, and each of them has a specific responsibility. Model is used to deal with the data, the database is connected so we can easily handle data logically. Since the controller never directly communicates with the database so model responds to the controller request. View component is used for data representation, showing the interface of the data provided by the controller. The controller is the main component that acts as a communicator between model and view. After getting all the data from the view, it processes the data and sends it to the view for representation. Here components are reusable and easy to maintain.

**2- Layered Architecture**

In this pattern, the components are organized in a horizontal layer. Although the components are connected, they are not dependent on each other. The four layers include presentation, business, persistence and database layer. The presentation layer has components for representation, the business layer has components for all business logic, the persistence layer uses object-oriented mapping and the database layer is where the database is stored. Additional layers can also be added to this pattern. It is easy to test since components are specified to each category of the layer. Changes to any layer mean that it has to be deployed all over again in a single unit.

**3- Client Server Architecture**

In this model, the server host, delivers, and most of the services and resources are consumed by the client. Many clients are connected to a centralized network and all the requests and services are delivered over a network. Since the components are doing their work independently of each other so distributed computing system is formed. It also has a frontend, for interacting with users. The application server that connects to the database is the backend. And database server where the data is being managed and operations and performed. It is very flexible and can be used with other patterns. Modules can be updated without changing the client-server service.

**4- Repository Architecture**

In this architecture pattern, several independent components operate on central data structures. It is very important for data integration including software development. The central repository is passive and the components as active. It is suitable for the applications in which central issues are likely to happen and we have to maintain a complex central body of information. This architecture triggers change to computation in a way that data structure is being changed. It is a very efficient way of storing and securing a large amount of data and restoring data can also be done easily.

**5- Pipe and Filter Architecture**

The independent entities in this architecture, filters, perform the transformation on the data and process when the input has been received. Pipes act as a linker between the stream of data that has been transformed and each component connected next in the pipeline. It can connect the output of one application with the input of another using pipe on the shell. It can also consist of more than one data source. This design is ideal if you have a large number of changes to execute and need to be able to use them in a variety of ways while maintaining their robustness. The architecture also allows for a recursive method, in which a filter is made up of a series of pipe filters.