Architecture pattern

Software architecture is the blueprint of building software. It shows the overall structure of the software, the collection of components in it, and how they interact with one another while hiding the implementation.

Software architecture patterns are predefined ways of organizing components in software architecture. These patterns have been tried and tested, successfully solving various problems by organizing components differently for specific software architecture problems.

Different Software Architecture Patterns:

- Layered Pattern
- Client-Server Pattern
- Event-Driven Pattern
- Microkernel Pattern
- Microservices Pattern

Layered Pattern:

Components(code) in this pattern are separated into layers of subtasks and they are arranged one above another.

Each layer has unique tasks to do and all the layers are independent of one another. Since each layer is independent, one can modify the code inside a layer without affecting others.

It is the most commonly used pattern for designing the majority of software. This layer is also known as ‘N-tier architecture’. Basically, this pattern has 4 layers.

(1) Presentation layer (The user interface layer where we see and enter data into an application.)
(2) Business layer (this layer is responsible for executing business logic as per the request.)

(3) Application layer (this layer acts as a medium for communication between the ‘presentation layer’ and ‘data layer’.

(4) Data layer (this layer has a database for managing data.)

Ideal for:

E-commerce web applications development like Amazon.

Client-Server Pattern :

The client-server pattern has two major entities,Server and multiple clients. Here the server has resources(data, files or services) and a client requests the server for a particular resource. Then the server processes the request and responds back accordingly.

Examples of software developed in this pattern:

- Email.
- WWW.
- File sharing apps.
- Banking.

Event-Driven Pattern :

Event-Driven Architecture is an agile approach in which services (operations) of the software are triggered by events.

Well, what does an event mean?

When a user takes action in the application built using the EDA approach, a state change happens and a reaction is generated that is called an event.

Eg: A new user fills the signup form and clicks the signup button on Facebook and then a FB account is created for him, which is an event.

Ideal for:

Building websites with JavaScript and e-commerce websites in general.

Microkernel Pattern :

Microkernel pattern has two major components, Core system and plug-in modules.

The core system handles the fundamental and minimal operations of the application.

The plug-in modules handle extended functionalities (like extra features) and customized processing.

Let's imagine, you have successfully built a chat application. And the basic functionality of the app is that you can text with people across the world without an internet connection. After some time, you would like to add a voice messaging feature to the application, then you are adding the feature successfully. You can add that feature to the already developed application because the microkernel pattern facilitates you to add features as plug-ins.

Microkernel pattern is ideal for:

Product-based applications and scheduling applications. We love new features that keep giving dopamine boost to our brain. Such as Instagram reels, YouTube Shorts and a lot more that feasts us digitally. So this pattern is mostly preferred for app development.

Microservices Pattern :

The collection of small services that are combined to form the actual application is the concept of microservices pattern. Instead of building a bigger application, small programs are built for every service (function) of an application independently. And those small programs are bundled together to be a full-fledged application.

So, adding new features and modifying existing microservices without affecting other microservices are no longer a challenge when an application is built in a microservices pattern.

Modules in the application of microservices patterns are loosely coupled. So they are easily understandable, modifiable and scalable.

Example Netflix is one of the most popular examples of software built-in microservices architecture. This pattern is most suitable for websites and web apps having small components.