**Reactive Programming**

Reactive programming is a programming paradigm that deals with asynchronous data streams (sequences of events) and the specific propagation of change, which means it implements modifications to the execution environment (context) in a certain order.

Graphical user interface, text

Description automatically generatedGraphical user interface, website

Description automatically generated

A picture containing diagram

Description automatically generatedGraphical user interface, application

Description automatically generatedGraphical user interface, text

Description automatically generatedGraphical user interface, text

Description automatically generatedGraphical user interface, text

Description automatically generatedGraphical user interface

Description automatically generated with low confidenceGraphical user interface, text

Description automatically generated

Graphical user interface, text, application

Description automatically generated

A picture containing diagram

Description automatically generated

**Event Driven Architecture:**

An event-driven architecture is an architectural paradigm that uses events to trigger and communicate between loosely coupled applications are services. These events needs to be published to an event broker platform and be sent over to consuming applications.

Components of Event Driven Architecture

1) Producers

2) Router

3) Consumers

Producers: Producers are apps or services that publish events to event broker platform

Router: The event router or broker which is a platform or middleware that takes in events and routes them to their respective consuming applications and of course the consumer which is another app or service that consumes to a particular topic in an event router, receives the event or stream of events and does something with it like save the state to the persistent storage or for analytics.

*When design an event driven architecture there is usually two models that we could implement*

1. **Pub/Sub Model**: Events are published to a broker topic and sent over to one or more subscribers. Once received the event cannot be backtraced or reread again and new subscribers do not see the event.

2. **Event Streaming**: Events are written to a log and orderes in a partition. A client application can read from any part of the stream anytime and replay the events.

Benefits of Event-Driven Architecture

1. Asynchronous

2. Scalable and Failure Independent.

3. Auditing and Point-int-time recovery.

**Apache Kafka**

Apache Kafka is an open-source, distribute event streaming platform.