**Architectural Styles in Software Design (13 marks)**

An **architectural style** is a **set of principles and patterns** that provide a structure for organizing a software system. Each style defines a unique way of **dividing responsibilities**, **organizing components**, and **managing communication** between them.

Below are the key architectural styles used in software design:

**1. Layered Architecture (2 marks)**

* **Structure:** The system is divided into layers, each with a specific role (e.g., presentation, business logic, data access).
* **Features:** Each layer only communicates with the layer directly below it.
* **Use cases:** Common in enterprise applications and operating systems.
* **Advantage:** Improves **modularity** and **maintainability**.

**2. Client-Server Architecture (2 marks)**

* **Structure:** Divides the system into **clients** (requesting services) and **servers** (providing services).
* **Example:** Web browsers (clients) and web servers.
* **Use cases:** Web applications, database systems.
* **Advantage:** Supports distributed systems and centralised control.

**3. Model-View-Controller (MVC) (2 marks)**

* **Structure:** Divides application into:
  + **Model:** Manages data and business rules.
  + **View:** Handles UI and presentation.
  + **Controller:** Processes user input and updates model/view.
* **Use cases:** Web frameworks (e.g., Django, Rails), GUI applications.
* **Advantage:** Separates concerns, improves **testability** and **scalability**.

**4. Microservices Architecture (2 marks)**

* **Structure:** System is broken into **independent, loosely coupled services**, each responsible for a specific business capability.
* **Communication:** Services interact via APIs (e.g., REST, gRPC).
* **Use cases:** Large-scale enterprise systems, cloud-native apps.
* **Advantage:** Enables **scalability**, **flexible deployment**, and **technology diversity**.

**5. Event-Driven Architecture (2 marks)**

* **Structure:** Components communicate by **producing and responding to events**.
* **Types:** Publisher-subscriber model or event stream processing.
* **Use cases:** Real-time systems, IoT, e-commerce.
* **Advantage:** High responsiveness and **asynchronous processing**.

**6. Pipe-and-Filter Architecture (1 mark)**

* **Structure:** Data flows through a sequence of processing elements (filters) connected by pipes.
* **Use cases:** Compilers, data processing tools.
* **Advantage:** Easy to maintain and reuse filters.

**7. Service-Oriented Architecture (SOA) (1 mark)**

* **Structure:** Applications are built using **reusable, loosely coupled services**.
* **Communication:** Often uses XML/SOAP or REST.
* **Use cases:** Integration of different systems, enterprise apps.
* **Advantage:** Promotes **interoperability** and **reusability**.