**ASSIGNMENT 1:**

**Principles in Practice - Draft a one-page scenario where you apply Microservices Architecture and Event-Driven Architecture to a hypothetical e-commerce platform. Outline how SOLID principles could enhance the design. Use bullet points to indicate how DRY and KISS principles can be observed in this context.**

SOLUTION:

**1.Microservices Architecture for User Management**

Microservices Application:

* User Service: Manages user profiles, including registration, authentication, and authorization. It handles user-related data and provides endpoints for user management operations.
* Order Service: Processes orders placed by users. It handles order validation, payment processing, and order tracking.
* Review Service: Allows users to leave reviews for products they have purchased. It manages review submissions, updates, and retrievals.
* Recommendation Service: Provides personalized product recommendations based on user behavior and preferences. It processes data from various services to generate relevant recommendations.

**2.Event-Driven Architecture:**

* User Registered Event: When a new user registers, the User Service emits an event. This event can trigger actions in other services, such as sending a welcome email or updating recommendation algorithms.
* Order Placed Event: When an order is completed, the Order Service emits an event. This can trigger inventory updates, recommendation adjustments, and order confirmation notifications.
* Review Submitted Event: When a user submits a review, the Review Service emits an event. This can be used to update product ratings and influence recommendations.
* Recommendation Update Event: The Recommendation Service listens for various events (e.g., user registrations, order placements, review submissions) to update and refine its recommendations.

**3.Applying SOLID Principles:**

* Single Responsibility Principle (SRP): Each service is responsible for a specific aspect of user management. For example, the User Service handles user profiles, while the Review Service deals with product reviews.
* Open/Closed Principle (OCP): Services can be extended with new features, such as additional review types or recommendation algorithms, without modifying existing functionality.
* Liskov Substitution Principle (LSP): Different implementations of user authentication can be used interchangeably, allowing for flexibility in security mechanisms.
* Interface Segregation Principle (ISP): Each service provides a focused set of endpoints. For example, the Order Service offers endpoints related to order processing but does not expose user profile management functions.
* Dependency Inversion Principle (DIP): Services depend on abstract interfaces rather than specific implementations, allowing for easier substitution and extension.

**4.Observing DRY and KISS Principles:**

* DRY (Don't Repeat Yourself): Common functionalities, such as logging and error handling, are implemented in shared libraries and reused across services.
* KISS (Keep It Simple, Stupid): Each service is designed to handle a specific task, keeping the overall system architecture simple and manageable.

**ASSIGNMENT 2:**

**Design Pattern Explanation - Prepare a one-page summary explaining the MVC (Model-View-Controller) design pattern and its two variants. Use diagrams to illustrate their structures and briefly discuss when each variant might be more appropriate to use than the others.**

**MVC OVERVIEW:**

The MVC design pattern is a widely-used architectural pattern that helps in separating concerns within an application, promoting organized and maintainable code.

* Model: Contains the core functionality and data of the application. It directly manages the data, logic, and rules.
* View: Displays the data and interacts with the user. It sends user actions to the Controller.
* Controller: Handles user inputs and updates the Model and View accordingly.

Diagram: MVC Basic Structure

| Model | <----> | Controller | <----> | View |

**MVC Pattern Variants**

**1.Model-View-Presenter (MVP)**

* Presenter: Acts as a mediator that fetches data from the Model and applies it to the View. The View is passive and only displays data as instructed by the Presenter.
* Use Case: Suitable for applications with complex UIs where the separation of UI logic from the view elements is needed.

**Diagram: MVP Structure**

| Model | <----> | Presenter | <----> | View |

**2.Model-View-ViewModel (MVVM)**

* ViewModel: Manages the state of the View and facilitates data binding. It serves as a binder that decouples the View from the Model, providing a more manageable structure for complex UIs.
* Use Case: Ideal for applications using data-binding frameworks, enhancing the responsiveness and manageability of the UI.

**Diagram: MVVM Structure**

| Model | <----> | ViewModel | <----> | View |

**ASSIGNMENT 3:**

**Trends and Cloud Services Overview - Write a three-paragraph report covering: 1) the benefits of serverless architecture, 2) the concept of Progressive Web Apps (PWAs), and 3) the role of AI and Machine Learning in software architecture. Then, in one paragraph, describe the cloud computing service models (SaaS, PaaS, IaaS) and their use cases**

1. Serverless architecture offers several advantages for modern software development. It allows developers to execute code without the hassle of managing infrastructure, leading to increased focus on application logic. This architecture is cost-effective since billing is based on actual usage rather than pre-allocated resources. Furthermore, serverless solutions inherently provide automatic scaling to handle varying loads, ensuring optimal performance without manual intervention.
2. Progressive Web Apps (PWAs) bridge the gap between web and native applications by leveraging modern web technologies. They deliver app-like experiences through web browsers, providing benefits such as offline functionality, quick loading times, and the ability to send push notifications. Users can install PWAs on their home screens, and they run independently of the internet, ensuring consistent and reliable performance. This combination of web accessibility and native-like features makes PWAs a versatile choice for developers aiming for broad reach and improved user engagement.
3. AI and Machine Learning (ML) play pivotal roles in enhancing software architecture. These technologies enable software to learn from data, making intelligent decisions and predictions. AI and ML can automate complex tasks, personalize user experiences, and provide insights through data analysis. In software architecture, they improve efficiency, accuracy, and adaptability by optimizing processes, detecting anomalies, and enhancing security measures. The integration of AI and ML leads to more sophisticated, responsive, and robust software systems.
4. Cloud computing encompasses three primary service models: Software as a Service (SaaS), Platform as a Service (PaaS), and Infrastructure as a Service (IaaS). SaaS provides users with access to software applications over the internet, such as email services or office suites, eliminating the need for local installation. PaaS offers a platform that supports the development, testing, and deployment of applications, allowing developers to focus on coding without worrying about underlying infrastructure. IaaS supplies virtualized computing resources, including servers, storage, and networking, enabling businesses to build and manage their own IT environments. Each model caters to different business requirements, providing flexibility and scalability in cloud service utilization.