**Examen Teórico**

1. ¿Qué es la arquitectura MVC y cuál es su propósito en el desarrollo de aplicaciones web?

La arquitectura modelo-vista-controlador, es un patrón de diseño de software. Su propósito es separar la lógica de la aplicación de la interfaz de usuario, es decir, el modelo se encarga de la lógica de negocio, se define como la aplicación interactúa con la base de datos y como se procesan dichos datos. La vista es la interfaz de usuario y presenta los datos que vienen del modelo. Y el controlador es el encargado de procesar las solicitudes del usuario desde la vista.

2. Explica qué es la programación orientada a objetos (POO) y menciona algunos conceptos clave asociados a ella.

Es un paradigma de programación que organiza el software en objetos, es decir, instancias de una clase. Cada objeto puede tener tanto atributos como métodos, lo que permite modelar un sistema complejo de una manera más estructurada. De entre los conceptos asociados a este, se encuentran las Clases, las cuales son moldes de lo que un objeto será, definen sus atributos y sus métodos. Los Objetos son instancias de las clases, cada objeto tiene sus propios valores para los atributos de su clase. El encapsulamiento es el principio de ocultar los detalles internos de un objeto y acceder a ellos solo mediante métodos (getters y setters). La herencia permite que una clase herede atributos y métodos de otra clase, para reutilizar código y crear una jerarquía de clases. El polimorfismo es la capacidad de que un método pueda comportarse de manera diferente en función del contexto, este puede ser por sobrecarga o sobreescritura. Finalmente la abstracción permite solo mostrar los aspectos esenciales de un objeto y oculta los detalles complejos, se logra a través de clases abstractas o interfaces definiendo comportamientos genéricos.

3. Menciona algunos patrones de diseño J2EE y explica brevemente su propósito.

Patrón Front Controller: Centraliza las solicitudes de los usuarios en un mismo controlador, todas las solicitudes pasan por el controlador antes de ser delegadas a otros componentes.

Patrón Transfer Object (DTO): Transporta datos entre capas de una aplicación, minimiza las llamadas remotas para obtener cada atributo de un objeto, mediante objetos planos se agrupan varios atributos.

Patrón Compose Entity: Simplifica la gestión de entidades relacionadas, utilizando un objeto compuesto para representar varios objetos dependientes, evita la duplicación de datos y facilita la gestión de entidades complejas.

Patrón Session Facade: Proporciona una interfaz simplificada para la lógica de negocio, agrupa múltiples operaciones en un solo servicio. Facilita la interacción entre la capa de presentación y los servicios.

4. ¿Qué son las APIs, sockets y webservices? ¿Cuál es su propósito en el desarrollo de aplicaciones?

APIs: es un conjunto de definiciones y protocolos que permiten que una aplicación interactúe con otra, estas proporcionan métodos que los desarrolladores pueden utilizar para acceder a funcionalidades o datos de otra aplicación, su propósito principal es permitir la comunicación e integración entre diferentes sistemas o aplicaciones.

Sockets: son un mecanismo de comunicación que permite el intercambio de datos entre dispositivos o aplicaciones en una red. Su principal uso es la comunicación en tiempo real ya que permiten la comunicación bidireccional.

WebService: es un conjunto de funciones o servicios para su uso a través de internet mediante protocolos estándar como HTTP, XML, SOAP y REST. Estos permiten la integración de aplicaciones distribuidas y heterogéneas, facilita la interoperabilidad en aplicaciones y servicios remotos.

5. ¿Qué significa REST y SOAP en el contexto de servicios web? ¿Cuáles son las principales diferencias entre ellos?

REST: es un estilo arquitectónico para diseñar servicios web que se basa en los principios de transferencia de estado y recursos. Utiliza métodos estándar HTTP como GET, POST, PUT, DELETE para realizar operaciones sobre los recursos. Su formato de datos suele ser de tipo JSON o XML, aunque JSON es mas utilizado. Se utiliza comúnmente en aplicaciones web y móviles modernas por su flexibilidad y uso eficiente de recursos.

SOAP: es un protocolo mas formal y estandarizado para el intercambio de mensajes entre aplicaciones a diferencia de REST. Este siempre utiliza XML para formatear mensajes lo cual los hace mas pesados, pero mas estructurados y seguros. Este es preferido en entornos empresariales donde se requieren transacciones complejas, mayor seguridad y consistencia.

6. ¿Qué es Kubernetes y cuál es su función en el despliegue y gestión de aplicaciones en contenedores?

Es una plataforma de orquestación de contenedores, diseñada para automatizar el despliegue, la gestión y la escalabilidad de aplicaciones. De sus principales funciones, se destaca la automatización del despliegue, ya que kubernetes facilita el despliegue y simplifica el ciclo de vida de aplicaciones en contenedores. Permite que las aplicaciones en contenedores se ejecuten en diferentes infraestructuras, ya sean nubes, servidores locales o ambos.

De entre sus componentes se encuentran, los nodos, es una maquina que ejecuta los contenedores. Un clúster, es un conjunto de nodos que ejecutan aplicaciones en contenedores. Un Pod, es la unidad mas pequeña de Kubernetes y contiene uno o mas contenedores que comparten recursos ya sea almacenamiento o red. Un Controlador, gestiona el ciclo de vida de los pods y las aplicaciones. Un Service, es una abstracción que define como acceder a un conjunto de pods. El Deployment, es una descripción de como debe ejecutarse y actualizarse las aplicaciones en un clúster.