Examen Teórico

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

Es un patrón de arquitectura el cual se encarga de dividir una aplicación en tres capas diferentes, las cuales son

Modelo: Representa la lógica de negocio de la aplicación, los datos y las reglas para manipular esos datos. Es el "cerebro" de la aplicación.

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

La programación orientada a objetos es un paradigma de programación el cual nos permite representar objetos de la vida real abstrayéndolos directamente en codigo (clases) con la cual nos facilita el entendimiento e interacción entre clases (objetos), algunos conceptos clave son: encapsulamiento, abstracción, polimorfismo y herencia

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

Algunos de los patrones más importantes y su utilidad son:

DAO (Data Access Object): Su objetivo es separar por completo la lógica de acceso a la base de datos del resto de la aplicación. Así, la capa de negocio no se contamina con detalles de cómo se guardan o leen los datos, facilitando el cambio de tecnología de persistencia sin afectar el núcleo del sistema.

DTO (Data Transfer Object): Un DTO es un objeto simple que usamos para pasar datos entre las diferentes capas de la aplicación (por ejemplo, del servicio al controlador o al frontend). La idea es transferir solo la información necesaria, optimizando la comunicación y evitando exponer datos internos que no deberían ser visibles.

Fachada (Facade): Este patrón nos ayuda a simplificar la interacción con un subsistema complejo. Es como crear una interfaz más amigable y de alto nivel que esconde la complejidad interna, lo que hace el sistema más fácil de usar y mantener.

Singleton: Garantiza que una clase solo tenga una única instancia en toda la aplicación y proporciona un punto de acceso global a esa instancia. Es útil para recursos compartidos y que son costosos de inicializar, como un gestor de conexiones a una base de datos o una configuración centralizada.

Por mencionar algunos

3.-¿Qué son las APIs, sockets y webservices? ¿Cuál es su propósito en el desarrollo de Aplicaciones?

APIs es el término más amplio. Es la definición de cómo interactuar con un programa. Un Web Service es un tipo de API.

Los Sockets son la capa para la comunicación en red, permitiendo un control directo sobre el flujo de datos. Su enfoque es principalmente para la comunicación en tiempo real.

Los Web Services son un tipo de API basada en estándares web (HTTP), diseñada para la comunicación entre sistemas distribuidos, siendo REST y SOAP los estilos más comunes.

El propósito de los tres conceptos es básicamente: intercambiar información

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

Rest: Representational State Transfer

Representational State Transfer (REST) significa que los clientes interactúan con los servicios web transfiriendo representaciones del estado de los recursos, y esta transferencia se realiza de manera sin estado.

SOAP: (Simple Object Access Protocol)

El término SOAP originalmente era un acrónimo para Simple Object Access Protocol. Sin embargo, con el tiempo y las evoluciones del estándar, el significado de ese acrónimo se volvió menos preciso y la especificación más moderna lo ha eliminado, simplemente refiriéndose a él como "SOAP".

La principal diferencia entre REST y SOAP en servicios web es que REST es un estilo arquitectónico ligero y sin estado que se basa en HTTP y JSON/XML, mientras que SOAP es un protocolo estricto y con estado basado en XML que puede usar varios transportes.

5.-¿Qué es Kubernetes y cuál es su función en el despliegue y gestión de aplicaciones en Contenedores?

No he usado Kubernetes, sin embargo, hasta donde he visto, funciona como un orquestador y administrador de contenedores