Prova Teórica

1-

O modelo 2 camadas é composto por uma camada de interface do usuário e uma camada de dados. O programa é instalado em cada cliente, este programa acessa dados em um banco de dados que fica em um servidor.

Prós:

* É uma arquitetura mais fácil e simples de entender
* Desempenho: como não temos uma camada intermediária de processamento, temos acesso direto aos dados.

Contras:

* Escalabilidade limitada: podendo ter dificuldade com o aumento de complexidade e volume de dados.
* Dificuldade na manutenção

No Modelo 3 camadas são retiradas as regras de negócio do lado do cliente e centralizadas em uma camada de servidor de aplicação também conhecida por camada lógica, o qual faz o acesso ao banco de dados.

Assim temos a camada de apresentação, onde encontramos a interface do sistema, camada que o usuário interage com o sistema. A camada lógica, onde temos as regras de negócio, que determinam a maneira que os dados serão utilizados. E a camada de dados, onde temos o banco de dados com todas as informações necessárias para o funcionamento da aplicação.

Prós

* Problemas de manutenção foram diminuídos e facilitando a evolução do sistema, pois as responsabilidades são separadas entre as camadas resultando em um código mais organizado.
* Escalabilidade e reutilização de código e camadas, proporcionando maior flexibilidade e adaptação facilitada.

Contras:

* O sistema tem uma complexidade maior, exigindo mais esforço e tempo para desenvolvimento.
* Pode ter impacto no desempenho por conta de ter uma camada a mais.

2-

SOAP é um protocolo baseado em xml, para a troca de mensagens entre sistemas. Usa envelopes xml para encapsular e transportar os dados. Permite integração com sistemas legados e heterogêneos. Tende a ser mais pesado em termos de tamanho de mensagem e processamento por conta de sua estrutura mais complexa.

REST: é um estilo arquitetural mais flexível e leve, que utiliza os princípios do protocolo HTTP para permitir a comunicação entre sistemas. Usa os verbos HTTP como GET, POST, PUT, DELETE para realizar operações nos recursos. O padrão REST facilita a interoperabilidade e a integração com a infraestrutura da web. Devido à sua simplicidade e uso de formatos de dados mais leves, como o JSON, o REST geralmente tem um desempenho melhor em termos de tamanho da mensagem e processamento.

3-

Facilita a troca de informações entre diferentes partes de um sistema, permitindo que os aplicativos se comuniquem e compartilhem dados.

É possível criar aplicativos cliente/servidor, onde um aplicativo atua como o servidor, buscando informações no banco de dados e aplicando as regras de negócio e fornecendo esses dados e informações ao cliente, e outros aplicativos atuam como clientes que se conectam ao servidor para acessar essas informações.

4-

Todos os hotéis teriam um servidor local, onde teriam um banco de dados e uma redundante ao banco de dados do data center, onde em situações emergenciais de falta de conexão seriam salvos os dados e quando a conexão fosse restabelecida os dados fossem transportados ao data center.

5-

A POO organiza o código em torno de objetos. Os objetos são criados a partir de modelos chamados classes, e interagem entre si por meio de trocas de mensagens.

Os três conceitos fundamentais são:

* Encapsulamento: É o conceito que permite agrupar os dados e os métodos relacionados em um objeto. Pode utilizar de métodos públicos e privados.
* Herança: A herança permite criar novas classes a partir de classes existentes, aproveitando características e comportamentos já definidos.
* Polimorfismo: um objeto pode assumir diferentes formas e comportamentos, dependendo do contexto em que é utilizado. O polimorfismo facilita a reutilização de código e a flexibilidade na implementação.

Três benefícios são:

* Permite que o código seja organizado em módulos independentes, ou seja, classes e objetos. Facilitando a manutenção, o entendimento e a reutilização do código
* Por meio da herança e do polimorfismo, permite-se reutilizar código já existente.
* A abstração permite representar objetos de forma simplificada, é possível modelar objetos complexos de maneira mais compreensível e criar uma representação mais próxima do domínio do problema.