Aluno: João Pedro Rodrigues Leite
RA: a2487055

1. Explique o conceito de Web API. Estabeleça uma relação entre aplicações Web convencionais e Web APIs. Discuta as semelhanças e diferenças entre essas duas classes de aplicações.

R: O conceito de Web API (Interface de programação de aplicações para a web) nada mais é do que uma aplicação que está disponibilizando serviços na Web assim como uma aplicação Web convencional, geralmente elas utilizam o mesmo protocolo para a comunicação que é o Hiper Text Transfer Protocol (HTTP).

Entretanto a principal diferença entre ela e uma aplicação Web convencional está em para quem elas estão fornecendo os serviços e trocando dados e informações. Na aplicação Web convencional os serviços são disponibilizados para as pessoas, onde o usuário pode interagir com a página com uma interface de usuário (UI), então forma de representar esse conteudo e as informações é muito mais voltado a para os usuários humanos, que seja de fácil entendimento e interação, a forma mais utilizada é através de páginas HTML com CSS e JS, como sites, e-commerce, redes sociais, onde é possível representar informações hipermídia como links que direcionam para outras páginas, mídia, como fotos, vídeos e áudios etc. O que já não acontece com uma Web API.

Uma aplicação Web API faz interações, trocando informações e dados com outras aplicações através de endpoints, que são como endereços na Web que podem ser utilizados para receber(GET), alterar(PUT), enviar(POST) e deletar(DELETE) informações, e por isso a forma de representar as informações já não é mais o HTML, as informações como por exemplo o nome, sobrenome e idade de uma pessoa seriam facilmente compreendidas por seres humanos, mas seriam apenas mais um texto comum que não tem significado nenhum no contexto de Web API pois a comunicação é máquina com máquina, então são utilizados formatos padronizados como JSON e XML para representação das informações, esses formatos utilizados geralmente garantem um desempenho superior em relação a aplicações Web convencionais que utilizam o HTML.

2. Quais são os princípios arquiteturais REST? Descreva com detalhes cada um deles. Faça um paralelo entre os princípios arquiteturais REST e as suas restrições impostas. Discuta sobre os benefícios que uma aplicação Web possui quando adota os princípios arquiteturais REST.

R:O REST(Representational State Transfer) foi proposto por Roy Fielding em 2000, e possui vários princípios e também restrições arquiteturais sendo eles:

- **Paradigma cliente/Servidor:** O REST utiliza do principio da separação entre cliente e servidor, onde o servidor será responsável pelo processamento dos dados enquanto o cliente fica responsável pela interface e experiencia do usuário. Dessa forma facilita a evolução e escalabilidade de ambos, pois alterações no cliente não afetam o servidor e vice-versa.

- **Comportamento stateless do servidor:** O comportamento stateless significa que a API REST armazena o estado de sessões do cliente, ou seja, todas as requisições devem ter todas as informações necessárias para a execução, em uma API que tenha autenticação, o usuário toda vez que precisar utilizar um recurso

terá de enviar os dados para que a autenticação seja feita. Sendo assim, então não é esperado que o servidor mantenha dados na sessão do usuário, isso também facilita na escalabilidade do servidor . A aplicação cliente que terá a responsabilidade de manter o estados dos recursos para o usuário final, ficando o servidor com a resposabilidade de disponibilizar representações de recursos e alterar seus estados.

- **Interface uniforme:** A manipulação dos recuros em uma API REST é feita atravé de endereços que tem um identificador único denominado URI(Uniform Resource Identifier). Os recursos podem ser qualquer informação, uma lista de boletins de ocorrencia de furtos de veículos, uma lista de suspeitos, ou até mesmo uma página de um usuário em uma rede social. E esses recursos podem ser representados de diferentes formas, HTML, PDF, XML, e jamais é acessado diretamente, sempre é através de uma representação, a representação mais utilizadas em Web Services REST é o JSON. Estabelecer uma interface uniforme entre cliente e servidor proporciona uma arquitetura simplificada e desacoplada e para isso é necessário respeitar a semântica do protocolo usado. O protocolo mais utilizado é o HTTP, e seus verbos são GET(obter representações de um recurso), POST(Criar um novo recurso), PUT(Alterar um recurso) e DELETE(Remover um recurso). Todas as requisições tratadas pelo servidor recebem um código de status, que informa ao cliente o resultado da requisição, e é muito importante utilizá-los corretamente. Os códigos de status possuem tamanho fixo de três dígitos e estão organizados da seguinte forma:

  ● 1XX - Informações;
  ● 2XX - Sucessos;
  ● 3XX - Redirecionamentos;
  ● 4XX - Erros causados pelo cliente;
  ● 5XX - Erros causados no servidor.

- **Utilização de controles hipermídia:** HATEOAS - Hypermedia as the Engine of Application State**.** Esse princípio define que as mudanças de estado devem ser feitas através de controler hipermídia, os controles hipermídia podem assumir a forma de links, que guiam a navegação entre diferentes recursos. Além disso Web Services REST podem ser projetados para interligar recursos, não só apenas responder as requisições e transportar dados.

Adotar os princípios arquiteturais REST nos dá inúmeras vantagens para aplicações Web, sendo elas a independência entre cliente e servidor, o que facilita a evolução e a manutenção de ambos, o comportamento stateless ajuda significativamente na escalabilidade e a complexidade é reduzida, permitindo que servidores lidem com um grande número de requisições de maneira eficiente. Além disso, a comunicação padronizada e desacoplada, utilizando protocolos como HTTP, facilita a interoperabilidade entre diferentes aplicações.

Embora existam limitações, como a necessidade de gerenciar a autenticação e a segurança sem manter estados, os benefícios em termos de simplicidade, flexibilidade, e facilidade de integração superam essas dificuldades, acredito o REST atende a maior parte de aplicações Web e ao seguir os princípios REST, as aplicações ganham uma base muito importante para construir sistemas distribuídos, confiáveis e escaláveis.

3. Argumente sobre o princípio arquitetural REST que implica no uso de controles hipermidia. Quais são os obstáculos para a sua utilização em Web APIs? Discuta quais são as alternativas para utilizar controles hipermidia em Web APIs.

R: No contexto REST, entende-se por aplicação um conjunto de recursos e seus respectivos estados manipulados por um cliente, e o princípio HATEOAS(Hypermedia as the Engine of Aplication State) define que as mudanças de estado da aplicação devem ser guiadas através de controles hipermídia, portanto os controler hipermidias podem assumir formas de links, que guiam a navegação entre diferentes recursos, um exemplo disso é quando um cliente vai fazer uma compra em um site e-commerce, o cliente irá selecionar o produto e se estiver disponível em estoque terá um botão para comprar e depois irá clicar em um botão para ir para o carrinho, caso não tenha em estoque terá uma opção para notificar quando o produto tiver estoque, ou seja, o cliente foi guiado através de links que representam recursos para determinadas ações de acordo com aquele estado. Outro exemplo é um cliente que acessa sua conta bancária, se ele tiver saldo terá um link disponível para que ele faça o saque do dinheiro, mas se não tiver saldo o link não estará disponível, todo esse controle de hipermídia é HATEOAS. Além disso Web Services REST podem ser projetados de modo que os recursos são interligados e se relacionam com outros recursos através de links.

Obstáculos: A complexidade para implementar o HATEOAS é maior, requer que os desenvolvedores criem e mantenham um sistema de links que representam o estado da aplicação e as transições possíveis, essa complexidade pode ser um obstáculo em projetos menores que requerem um desenvolvimento rápido. A performance da API também pode ser afetada, pois o envio de controles hipermídia em todas as respostas pode aumentar o tamanho dos payloads das respostas das API's. É necessário também que os clientes da API se adaptem para interpretar e reagir aos links de hipermídia, e isso podem exibir mudanças significativas nos clientes existentes.

Alternativas: Dado essas dificultados de implementação do HATEOAS, existem algumas alterativas, como o uso da HATEOAS com HAL(Hypertext Application Language), que é um formato de mídia para JSON que é simples de implementar e bem suportado por bibliotecas de diferentes linguagens de programação. É possível utilizar HATEOAS apenas para recursos críticos da aplicação que realmente seriam importantes, essa abordagem híbrida pode ser uma alterativa. Também pode-se utilizar ferramentas como Swagger/OpenAPI para fornecer a documentação detalhada e interativa que podem ajudar os desenvolvedores a interagir com a API sem os controles hipermídia.

4. Explique o conceito de Web semântica. Qual a relação entre Web semântica e dados conectados? Descreva as principais tecnologias utilizadas na Web semântica. Qual é a relação entre Web semântica e aplicações Web?

R: A Web Semântica é uma extensão da Web atual onde os dados possuem um significado associado, criando a visão de uma rede de informações interligadas, ou seja, a Web Semântica retira dos seres humanos a exclusividade de interpretação das informações. Associar um significado às informações possibilita que os computadores e os seres humanos trabalhem juntos, isso proporciona integração de reusabilidade de dados entre aplicações através do compartilhamento global de informações.

A Web original é chamada também de Web de Hipertextos que são informações como textos, imagens, vídeos, áudios e permite a ligação com outros hipertextos através de hiperlinks. A evolução disso é a Web de Dados, onde a informação é estruturada, permitindo que agentes de software interpretem as informações.

A Web semântica tem uma relação direta com os dados conectados (Linked Data), uma das tecnologias utilizadas é o RDF(Resource Description Framework), um modelo para representar as informações sobre os recursos na web, usando triplas (sujeito, predicado, objeto) para descrever a relação entre eles. Temos também o JSON-LD (JavaScript Object Notation for Linked Data) que permite criar contextos compostos por classes e termos de vocabulários controlados que são usados como fontes de metadados que descrevem o conteudo dos recursos. O JSON-LD permite associar as propriedades com termos de diferentes fontes semânticas, além da possibilidade de associar o documento a conjunto de classes que vincula um significado para o documento como um todo.

Vocabulários compartilhados como schema.org e FOAF(Friend of a Friend) são fundamentais na Web Semântica. O shema.org oferece um conjunto padronizado de vocabulários para marcar dados estruturados em paǵinas web, facilitando a compreensão desses dados por outras aplicações. O FOAF é mais utilizado para descrever informações sobre pessoas e suas relações sociais, permite a criação de reddes sociais descentralizadas e compreensíveis por máquinas. Esses vocabulários compartilham um conhecimento comum que melhora a interoperabilidade e a troca de dados entre diferentes sistemas.

Com a Web Semântica diferentes aplicações podem compartilhar e compreender dados sem a necessidade de um formato comum pré definido, promovendo maior interoperabilidade, ela também pode automatizar extração e processamento dos dados de diferentes fontes, facilitando a integração de dados.