Arthur Silva

Softex - Recife

**ATIVIDADE 4 – MÓDULO WEB SERVICES**

Explique quais são as seis regras de uma API REST.

*“APIs são mecanismos que permitem que dois componentes de software se comuniquem usando um conjunto de definições e protocolos. REST define um conjunto de funções como GET, PUT, DELETE e assim por diante, que os clientes podem usar para acessar os dados do servidor”.* Texto retirado do site: aws.amazon.com

Roy Fielding é o criador da REST e fez seis regras que a compõe:

1. Sem estado;
2. Cliente-servidor;
3. Interface uniforme;
4. Armazenamento em cache;
5. Sistema em camadas; e
6. Código sob demanda

**Sem estado**: de acordo com este princípio, cada solicitação enviada do cliente ao servidor deve incluir todas as informações necessárias para compreender a solicitação. As APIs são denominadas “Sem Estado” porque cada chamada é independente umas das outras.

**Cliente-servidor**:este princípio trabalha com o conceito de que cliente e servidor devem ser isolados um do outro e ter permissão para desenvolver de forma independente. Ao separar as preocupações de interface de usuário do armazenamento de dados, é possível melhorar a portabilidade através de múltiplas plataformas, simplificar os componentes do servidor, mas principalmente, permitir a evolução de forma independente uma vez que um não tem dependência do outro.

**Interface uniforme**:a característica da arquitetura REST é uma interface uniforme entre o cliente e servidor. Como o cliente e servidor compartilham esta interface, deve-se ter um “contrato” bem definido para comunicação entre os lados. Há quatros princípios que devem ser seguidos para obter uma interface uniforme: Identificação dos Recursos, Representação dos recursos, Mensagens auto-descritivas e Hypermedia.

**Armazenamento em cache**: como uma API sem estado pode aumentar sobrecarga de solicitação gerenciando grandes cargas de chamadas de entrada e saída, o cache ajuda a melhorar a performance, a escalabilidade e eficiência uma vez que reduz o tempo de resposta médio quando comparado entre uma série de interações cliente-servidor.

**Sistema em camadas**: a arquitetura REST inclui várias camadas que operam juntas para construir uma hierarquia que contribui para que a aplicação seja escalável e flexível. É recomendável que o cliente nunca se conecte diretamente no servidor de aplicação e que uma camada de balanceamento de carga seja adicionada entre cliente-servidor, isso ajuda na segurança e restringe o desempenho do componente. de modo que cada componente não pode "ver" além da camada imediata com a qual está se misturando.

**Código sob demanda**: código sob demanda define a possibilidade de serviços baseados em REST responderem como representação de um recurso informação executável pelo cliente. Esta possibilidade de entrega de funcionalidades, no entanto deve ser suportada pelo cliente criando um ambiente de execução propício para sua execução. Na maioria das vezes, um servidor retorna representação estática de recursos no formato XML ou JSON. Porém, quando necessário, os servidores podem fornecer código executável ao cliente.