O que é a API RESTful?

A API RESTful é uma interface que dois sistemas de computador usam para trocar informações de forma segura pela internet. A maioria das aplicações de negócios precisa se comunicar com outras aplicações internas e de terceiros para executar várias tarefas. Por exemplo, para gerar contracheques mensais, seu sistema interno de contas precisa compartilhar dados com o sistema bancário de seu cliente a fim de automatizar o faturamento e se comunicar com uma aplicação interna de planilha de horas. As APIs RESTful suportam essa troca de informações porque seguem padrões de comunicação de software seguros, confiáveis e eficientes.

O que é uma API?

Uma interface de programação de aplicação (API) define as regras que você precisa seguir para se comunicar com outros sistemas de software. Os desenvolvedores expõem ou criam APIs para que outras aplicações possam se comunicar com suas aplicações programaticamente. Por exemplo, a aplicação de planilha de horas expõe uma API que solicita o nome completo de um funcionário e um intervalo de datas. Ao receber essas informações, processa internamente a planilha de horas do funcionário e retorna o número de horas trabalhadas nesse intervalo de datas.

Você pode pensar em uma API da Web como um gateway entre clientes e recursos na Web.

Clientes

Clientes são usuários que desejam acessar informações da Web. O cliente pode ser uma pessoa ou um sistema de software que usa a API. Por exemplo, desenvolvedores podem escrever programas que acessam dados meteorológicos de um sistema meteorológico. Ou você pode acessar os mesmos dados do seu navegador ao visitar o site meteorológico diretamente.

Recursos

Recursos são as informações que diferentes aplicações fornecem aos seus clientes. Os recursos podem ser imagens, vídeos, textos, números ou qualquer tipo de dado. A máquina que fornece o recurso ao cliente também é chamada de servidor. As organizações usam APIs para compartilhar recursos e fornecer serviços da Web, mantendo a segurança, o controle e a autenticação. Além disso, as APIs ajudam a determinar quais clientes têm acesso a recursos internos específicos.

O que é REST?

A representational state transfer (REST – transferência de estado representacional) é uma arquitetura de software que impõe condições sobre como uma API deve funcionar. A REST foi criada inicialmente como uma diretriz para gerenciar a comunicação em uma rede complexa como a internet. Você pode usar a arquitetura baseada em REST para possibilitar a comunicação confiável e de alta performance em escala. Você pode implementá-la e modificá-la facilmente, trazendo visibilidade e portabilidade entre plataformas para qualquer sistema de API.

Os desenvolvedores de API podem projetar APIs usando várias arquiteturas diferentes. As APIs que seguem o estilo de arquitetura REST são chamadas de APIs REST. Os serviços da Web que

implementam a arquitetura REST são chamados de serviços da Web RESTful. O termo API RESTful geralmente se refere a APIs da Web RESTful. No entanto, você pode usar os termos API REST e API RESTful de forma intercambiável.

Veja a seguir alguns dos princípios do estilo de arquitetura REST.

Interface uniforme

A interface uniforme é fundamental para o design de qualquer serviço da Web RESTful. Indica que o servidor transfere informações em formato-padrão. O recurso formatado é chamado de representação em REST. Esse formato pode ser diferente da representação interna do recurso na aplicação do servidor. Por exemplo, o servidor pode armazenar dados como texto, mas enviá-los em um formato de representação HTML.

A interface uniforme impõe quatro restrições arquitetônicas:

1. As solicitações devem identificar recursos. Elas fazem isso usando um identificador de recurso uniforme.
2. Os clientes têm informações suficientes na representação do recurso para modificar ou excluir o recurso caso queiram. O servidor atende a essa condição enviando metadados que descrevem melhor o recurso.
3. Os clientes recebem informações sobre como processar ainda mais a representação. O servidor faz isso enviando mensagens autodescritivas que contêm metadados sobre como o cliente pode usá-los melhor.
4. Os clientes recebem informações sobre todos os outros recursos relacionados de que precisam para concluir uma tarefa. O servidor faz isso enviando hiperlinks na representação para que os clientes possam descobrir dinamicamente mais recursos.

Ausência de estado

Na arquitetura REST, a ausência de estado refere-se a um método de comunicação no qual o servidor completa cada solicitação do cliente independentemente de todas as solicitações anteriores. Os clientes podem solicitar recursos em qualquer ordem, e cada solicitação é sem estado ou isolada de outras solicitações. Essa restrição de design da API REST implica que o servidor possa entender completamente e atender à solicitação todas as vezes.

Sistema em camadas

Em uma arquitetura de sistema em camadas, o cliente pode se conectar a outros intermediários autorizados entre o cliente e o servidor e ainda receber respostas do servidor. Os servidores também podem passar solicitações para outros servidores. Você pode projetar seu serviço da Web RESTful para ser executado em vários servidores com diversas camadas, como segurança, aplicação e lógica de negócios, trabalhando juntos para atender às solicitações do cliente. Essas camadas permanecem invisíveis para o cliente.

Capacidade de armazenamento

Os serviços da Web RESTful permite o armazenamento em cache, que é o processo de armazenar algumas respostas no cliente ou em um intermediário para melhorar o tempo de resposta do servidor. Por exemplo, digamos que você visite um site que tenha imagens comuns de cabeçalho e rodapé em todas as páginas. Toda vez que você visita uma nova página do site, o servidor deve reenviar as mesmas imagens. Para evitar isso, o cliente armazena em cache ou armazena essas imagens após a primeira resposta e, em seguida, usa as imagens diretamente do cache. Os serviços da

Web RESTful controlam o cache usando respostas de API que se definem como armazenáveis ou não em cache.

Código sob demanda

No estilo de arquitetura REST, os servidores podem estender ou personalizar temporariamente a funcionalidade do cliente, transferindo o código de programação de software para o cliente. Por exemplo, quando você preenche um formulário de registro em qualquer site, seu navegador imediatamente destaca os erros cometidos, como números de telefone incorretos. Ele pode fazer isso devido ao código enviado pelo servidor.

Quais são os benefícios das APIs RESTful?

APIs RESTful incluem os seguintes benefícios:

Escalabilidade

Os sistemas que implementam APIs REST podem ser escalados com eficiência, porque a REST otimiza as interações entre cliente e servidor. A ausência de estado remove a carga do servidor, porque o servidor não precisa reter informações de solicitações anteriores do cliente. O cache bem gerenciado elimina parcial ou completamente algumas interações entre cliente e servidor. Todos esses recursos permitem a escalabilidade sem causar gargalos de comunicação que reduzem a performance.

Flexibilidade

Os serviços da Web RESTful permitem a separação total do cliente do servidor. Eles simplificam e desacoplam vários

componentes do servidor para que cada parte possa evoluir independentemente. Mudanças de plataforma ou tecnologia na aplicação do servidor não afetam a aplicação do cliente. A capacidade de camadas de funções de aplicações aumenta ainda mais a flexibilidade. Por exemplo, os desenvolvedores podem fazer alterações na camada de banco de dados sem reescrever a lógica da aplicação.

Independência

APIs REST são independentes da tecnologia usada. Você pode escrever aplicações de cliente e servidor em várias linguagens de programação sem afetar o design da API. Também é possível alterar a tecnologia subjacente em ambos os lados sem afetar a comunicação.

Como funcionam as APIs RESTful?

A função básica de uma API RESTful é a mesma de navegar na internet. O cliente entra em contato com o servidor usando a API quando requer um recurso. Os desenvolvedores de API explicam como o cliente deve usar API REST na documentação da API da aplicação do servidor. Estas são as etapas gerais para qualquer chamada de API REST:

1. O cliente envia uma solicitação ao servidor. O cliente segue a documentação da API para formatar a solicitação de modo que o servidor entenda.
2. O servidor autentica o cliente e confirma que o cliente tem o direito de fazer essa solicitação.
3. O servidor recebe a solicitação e a processa internamente.

4. O servidor retorna uma resposta ao cliente. A resposta contém informações que indicam ao cliente se a solicitação foi bem-sucedida. A resposta também inclui informações solicitadas pelo cliente.

Os detalhes de solicitação e resposta da API REST variam um pouco, dependendo de como os desenvolvedores da API projetam a API.

O que contém a solicitação do cliente da API RESTful?

As APIs RESTful exigem que as solicitações contenham os seguintes componentes principais:

Identificador de recurso exclusivo

O servidor identifica cada recurso com identificadores de recursos exclusivos. Para serviços REST, o servidor normalmente realiza a identificação de recursos usando um uniform resource locator (URL – localizador de recurso uniforme). O URL especifica o caminho para o recurso. Um URL é semelhante ao endereço do site que você digita no navegador para visitar qualquer página da Web. O URL também é chamado de endpoint de solicitação e especifica claramente ao servidor o que o cliente requer.

Método

Os desenvolvedores geralmente implementam APIs RESTful usando o Hypertext Transfer Protocol (HTTP – Protocolo de Transferência de Hipertexto). Um método HTTP informa ao servidor o que ele precisa fazer com o recurso. A seguir, estão quatro métodos HTTP comuns:

GET

Os clientes usam GET para acessar recursos localizados no URL especificado no servidor. Eles podem armazenar em cache solicitações GET e enviar parâmetros na solicitação da API RESTful para instruir o servidor a filtrar dados antes de enviar.

POST

Os clientes usam POST para enviar dados ao servidor. Eles incluem a representação de dados com a solicitação. Se enviarem a mesma solicitação POST várias vezes, criarão o mesmo recurso várias vezes.

PUT

Os clientes usam PUT para atualizar recursos existentes no servidor. Ao contrário do POST, o envio da mesma solicitação PUT várias vezes em um serviço da Web RESTful tem o mesmo resultado.

DELETE

Os clientes usam a solicitação DELETE para remover o recurso. Uma solicitação DELETE pode alterar o estado do servidor. No entanto, se o usuário não tiver a autenticação apropriada, a solicitação falhará.

Cabeçalhos HTTP

Os cabeçalhos de solicitação são os metadados trocados entre o cliente e o servidor. Por exemplo, o cabeçalho da solicitação indica o formato da solicitação e da resposta, fornece informações sobre o status da solicitação e assim por diante.

Dados

As solicitações da API REST podem incluir dados para POST, PUT e outros métodos HTTP para funcionarem com êxito.

Parâmetros

As solicitações da API RESTful podem incluir parâmetros que fornecem ao servidor mais detalhes sobre o que precisa ser feito. A seguir, estão alguns tipos diferentes de parâmetro.

- Parâmetros de caminho que especificam detalhes do URL.
- Parâmetros de consulta que solicitam mais informações sobre o recurso.
- Parâmetros de cookies que autenticam clientes rapidamente.

O que são métodos de autenticação da API RESTful?

Um serviço da Web RESTful deve autenticar solicitações antes de enviar uma resposta. Autenticação é o processo de verificação de uma identidade. Por exemplo, você pode provar sua identidade mostrando uma carteira de identidade ou carteira de motorista. Da mesma forma, os clientes do serviço RESTful devem provar a identidade ao servidor para estabelecer confiança.

A API RESTful tem quatro métodos de autenticação comuns:

Autenticação de HTTP

O HTTP define alguns esquemas de autenticação que você pode usar diretamente ao implementar a API REST. Estes são dois desses esquemas:

Autenticação básica

Na autenticação básica, o cliente envia o nome de usuário e a senha no cabeçalho da solicitação. Ele os codifica com base64, que é uma técnica de codificação que converte o par em um conjunto de 64 caracteres para transmissão segura.

Autenticação do portador

O termo autenticação do portador se refere ao processo de dar controle de acesso ao portador do token. O token do portador é normalmente uma cadeia de caracteres criptografada que o servidor gera em resposta a uma solicitação de login. O cliente envia o token nos cabeçalhos de solicitação para acessar recursos.

Chaves de API

As chaves de API são outra opção para autenticação da API REST. Nessa abordagem, o servidor atribui um valor gerado exclusivo a um cliente iniciante. Sempre que o cliente tenta acessar recursos, ele usa a chave de API exclusiva para verificar a si mesmo. As chaves de API são menos seguras porque o cliente precisa transmitir a chave, o que a torna vulnerável a roubo de rede.

OAuth

OAuth combina senhas e tokens para acesso de login altamente seguro a qualquer sistema. Primeiro, o servidor solicita uma senha e, depois, um token adicional para concluir o processo de autorização. Ele pode verificar o token a qualquer momento e também ao longo do tempo com escopo e longevidade específicos.

O que contém a resposta do servidor da API RESTful?

Os princípios da REST exigem que a resposta do servidor contenha os seguintes componentes principais:

Linha de status

A linha de status contém um código de status de três dígitos que comunica o sucesso ou a falha da solicitação. Por exemplo, os códigos 2XX indicam êxito, mas os códigos 4XX e 5XX indicam erros. Os códigos 3XX indicam redirecionamento de URL.

A seguir, estão alguns códigos de status comuns.

- 200: resposta genérica de êxito
- 201: resposta de êxito do método POST
- 400: solicitação incorreta que o servidor não pode processar
- 404: recurso não encontrado

Corpo da mensagem

O corpo da resposta contém a representação do recurso. O servidor seleciona um formato de representação apropriado com base no que os cabeçalhos da solicitação contêm. Os clientes podem solicitar informações nos formatos XML ou JSON, que definem como os dados são escritos em texto simples. Por exemplo, se o cliente solicitar o nome e a idade de uma pessoa chamada John, o servidor retornará uma representação JSON da seguinte forma:

'{"name":"John", "age":30}'

Cabeçalhos

A resposta também contém cabeçalhos ou metadados sobre a resposta. Elas fornecem mais contexto sobre a resposta e incluem informações como servidor, codificação, data e tipo de conteúdo.

Como a AWS pode ajudar com o gerenciamento de API RESTful?

O Amazon API Gateway é um serviço totalmente gerenciado que facilita aos desenvolvedores criar, publicar, manter, monitorar e proteger APIs em qualquer escala. Usando o API Gateway, você pode criar APIs RESTful para aplicações de comunicação bidirecional em tempo real:

Com o API Gateway, é possível:

- Fornecer aos usuários performance de alta velocidade para solicitações e respostas de API.
- Autorizar o acesso às suas APIs com o AWS Identity and Access Management (IAM) e o Amazon Cognito; ambos fornecem suporte OAuth nativo.
- Executar várias versões da mesma API simultaneamente com o API Gateway para iterar, testar e lançar rapidamente novas versões.
- Monitorar métricas de performance e informações sobre chamadas de API, latência de dados e taxas de erro do API Gateway.

Comece a usar o API Gateway usando nosso guia detalhado e criando uma conta da AWS hoje mesmo.