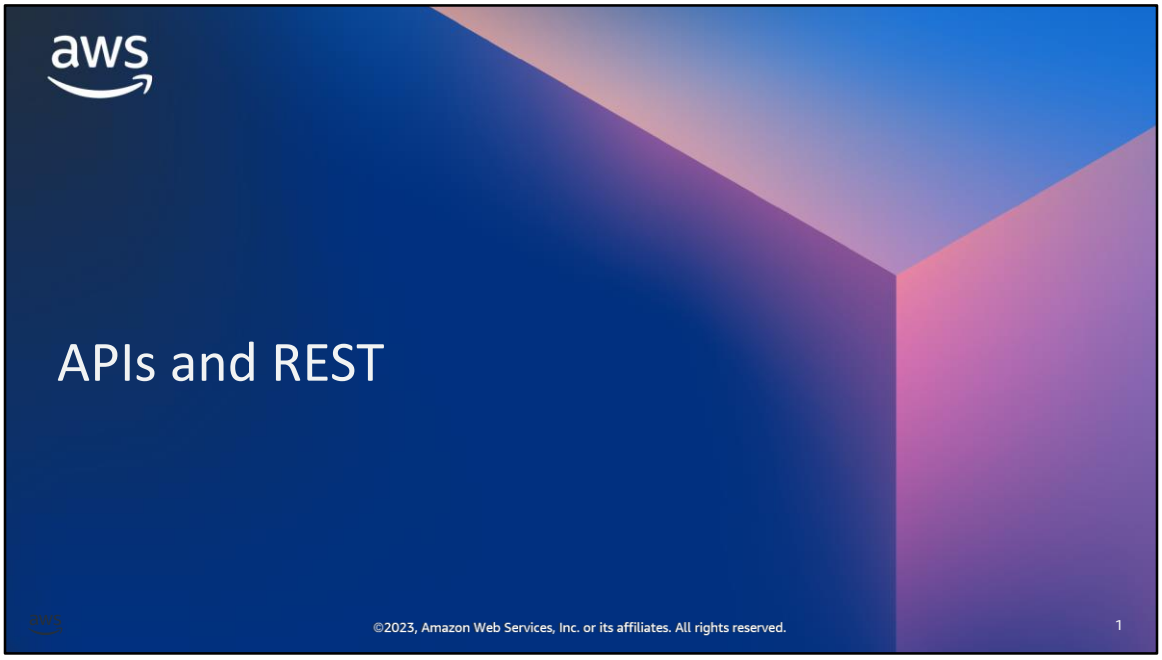


APIs and REST

At the core of the unit

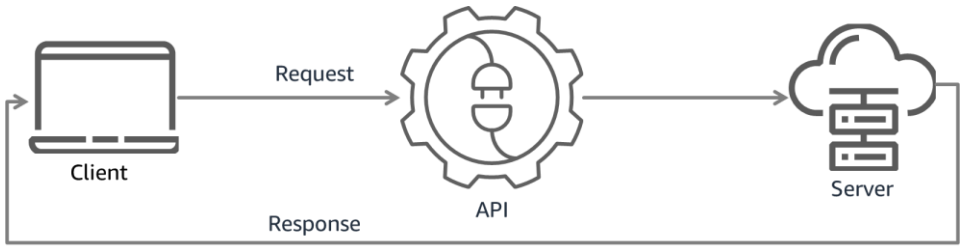
You will learn how to describe the purpose and function of APIs, including REST.

You will learn how to do the following:

- Describe the purpose and function of application programming interfaces (APIs), including Representational State Transfer (REST).

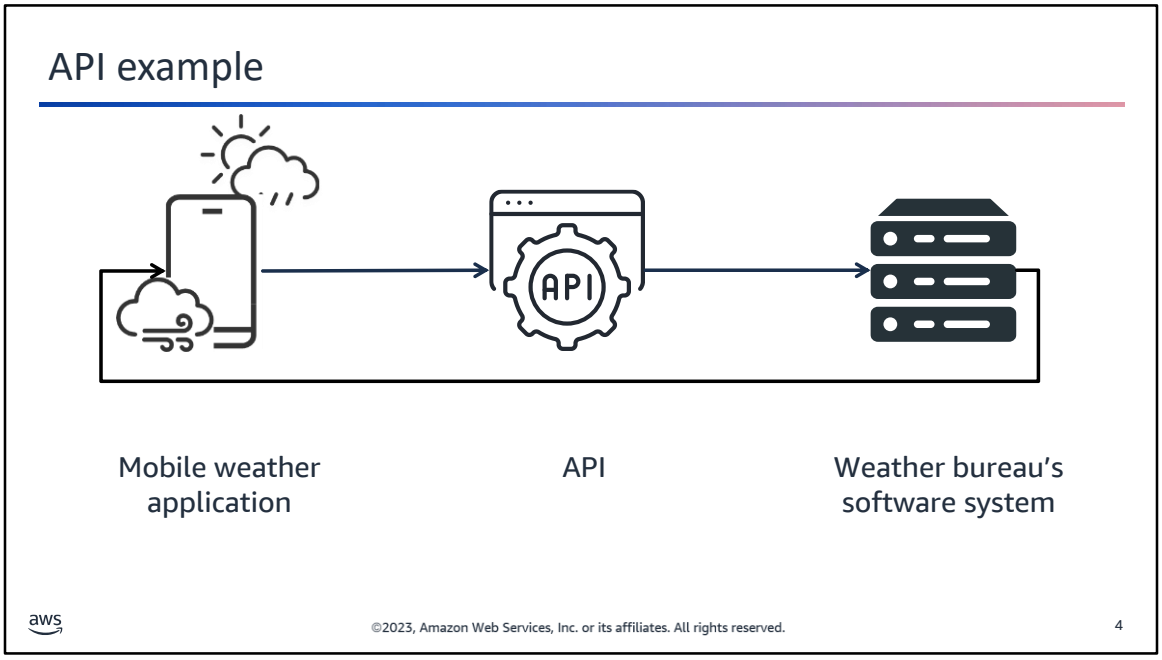
What is an API?

- An API provides programmatic access to an application.
  - The client application sends a request to the server application by using the server application's API.
  - The server application returns a response to the client application.
- Benefits of an API include the following:
  - Provides access to an application's functions without using a graphical user interface (GUI)
  - Provides a consistent way to invoke an application's functions



Most people who use computers are familiar with using a graphical user interface (GUI) to interact with an application. For example, the AWS Management Console is a GUI that a person can use for interacting with AWS services.

When computer programs need to communicate with each other, they use APIs. APIs make computer software accessible to developers through code so that developers can build software programs that interact with other software programs. APIs work as an intermediary so that applications are able to communicate with each other. This communication is initiated with an API call. An API call is a message that is delivered to an application as a request and then returned as a response to the requester. APIs define a standard way of invoking application functions programmatically.

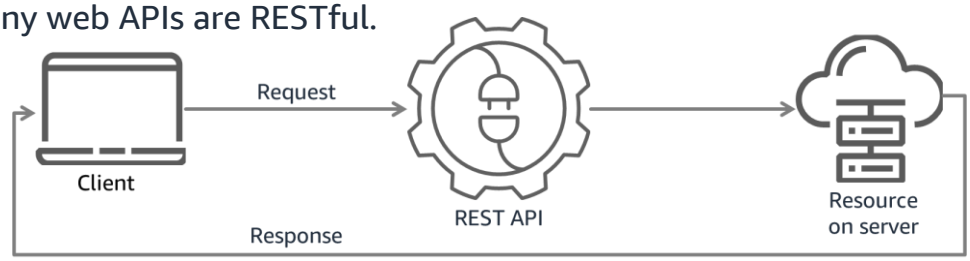


For example, the weather bureau's software system contains daily weather data. The weather app on your phone communicates with this system via APIs and shows you daily weather updates on your phone.

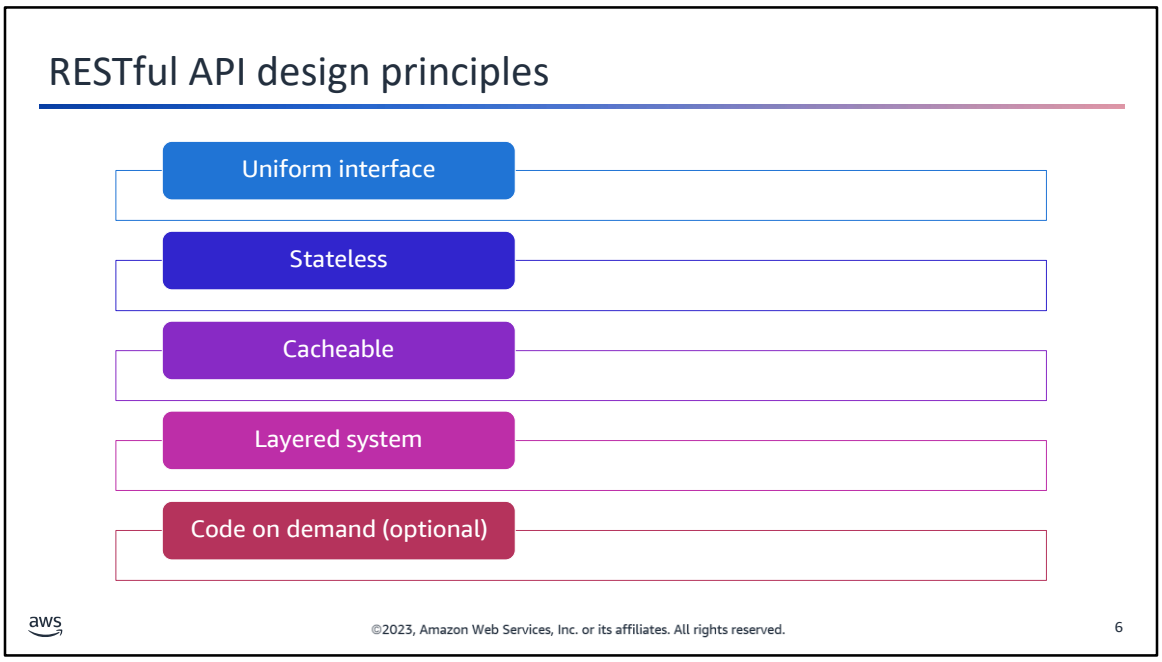
The public APIs of AWS services are another API example. When you use a tool such as the AWS Command Line Interface (AWS CLI), the tool uses the public APIs exposed by the AWS services to invoke service functions.

What are RESTful APIs?

- A RESTful API is an interface that two computer systems use to exchange information securely over the internet.
  - Is designed for loosely coupled network-based applications
  - Communicates over HTTP
  - Exposes resources at specific URIs
- Many web APIs are RESTful.



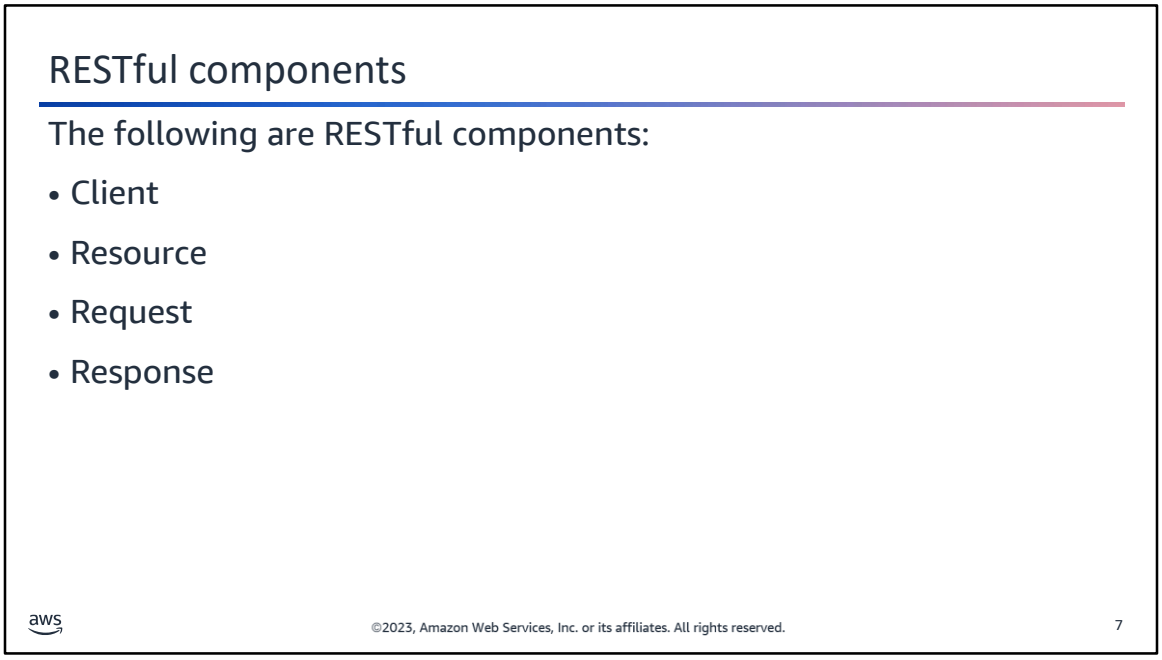
Some APIs follow the principles of REST. REST APIs communicate over HTTP and follow secure, reliable, and efficient software communication standards. REST is a popular API design that has largely replaced Simple Object Access Protocol (SOAP) as the standard for web services. Web-based APIs that adhere to the REST design principles are said to be RESTful.



REST follows five design principles:

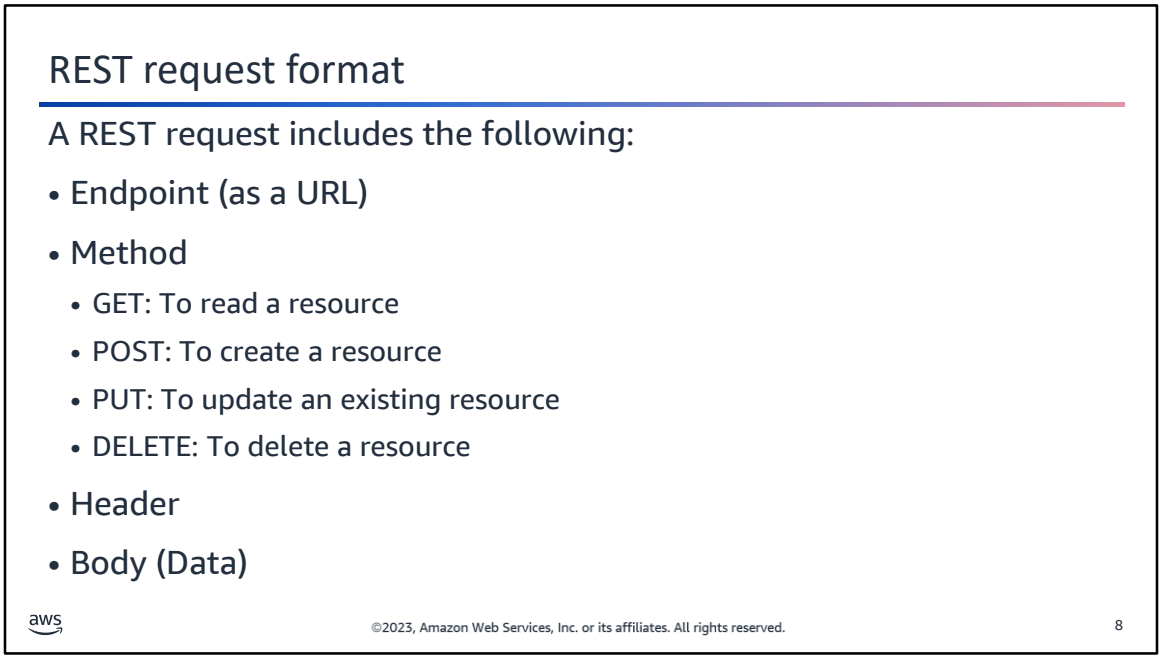
- The client should have a uniform interface to access the server. A request should be made to a single endpoint or URI when it interacts with each distinct resource that is part of the service. The interface does not define the structure of what is returned when the client makes a request.
- RESTful services are also stateless. Stateless means that the server does not track which requests the connecting client has made over time. It also does not keep track of which step the client might have completed in terms of a series of actions. Instead, any session information about the client is known only to the client itself. The server doesn't know about the state of the client, and the client doesn't know about the state of the server.
- REST clients should be able to cache the responses that they receive from the REST server.
- RESTful services support layered systems, where the client might connect to an intermediate server. The REST server can be distributed, which supports load balancing.
- RESTful services can optionally support code on demand. Code on demand means that the server could pass code (that can be run) to the client, such as some JavaScript. This feature extends the functionality of the REST client.

NOTE: The use of code on demand is not widely adopted across developers and is still an optional component of API development.



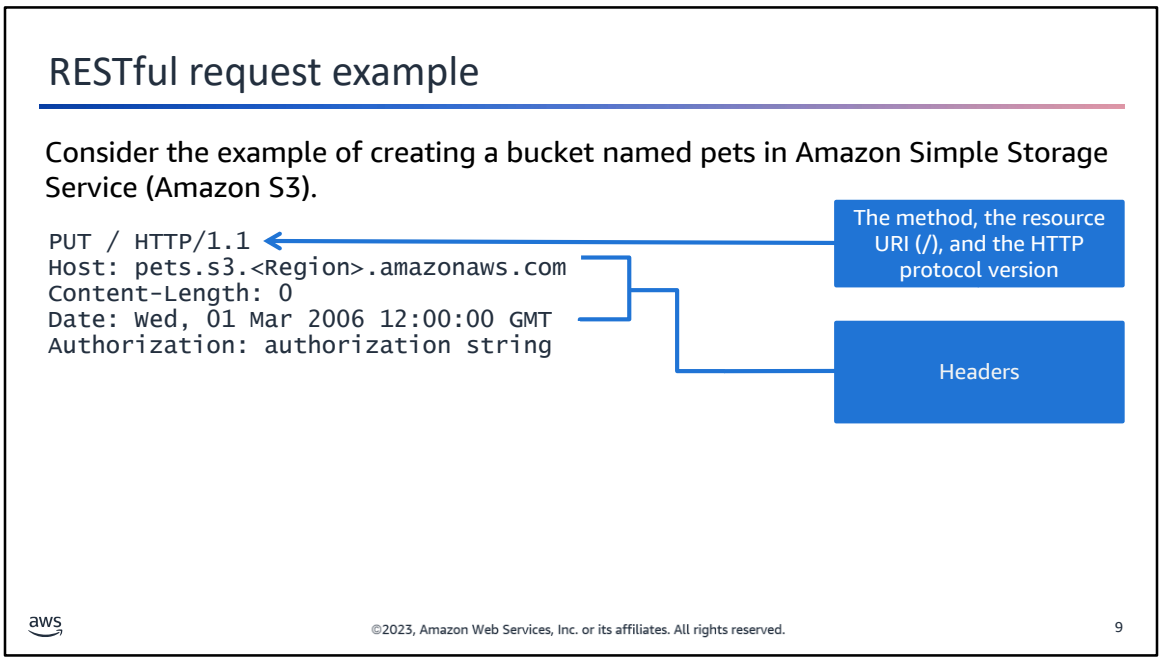
The following are RESTful components:

- Client: Clients are users who want to access information from the web. The client can be a person or a software system that uses the API. For example, developers can write programs that access weather data from a weather system. You can also access the same data from your browser when you visit the weather website directly.
- Resource: Resources are the information that different applications provide to their clients. Resources can be images, videos, text, numbers, or any type of data. The machine that gives the resource to the client is also called the server. Organizations use APIs to share resources and provide web services while maintaining security, control, and authentication. In addition, APIs help them to determine which clients get access to specific internal resources.
- Request: Requests are sent by the client to a server. The request is formatted so that the server will understand.
- Response: The response is what the server sends back to reply to the request from the client. This information includes a status message of success or failure, a message body containing the resource representation, and metadata about the response.



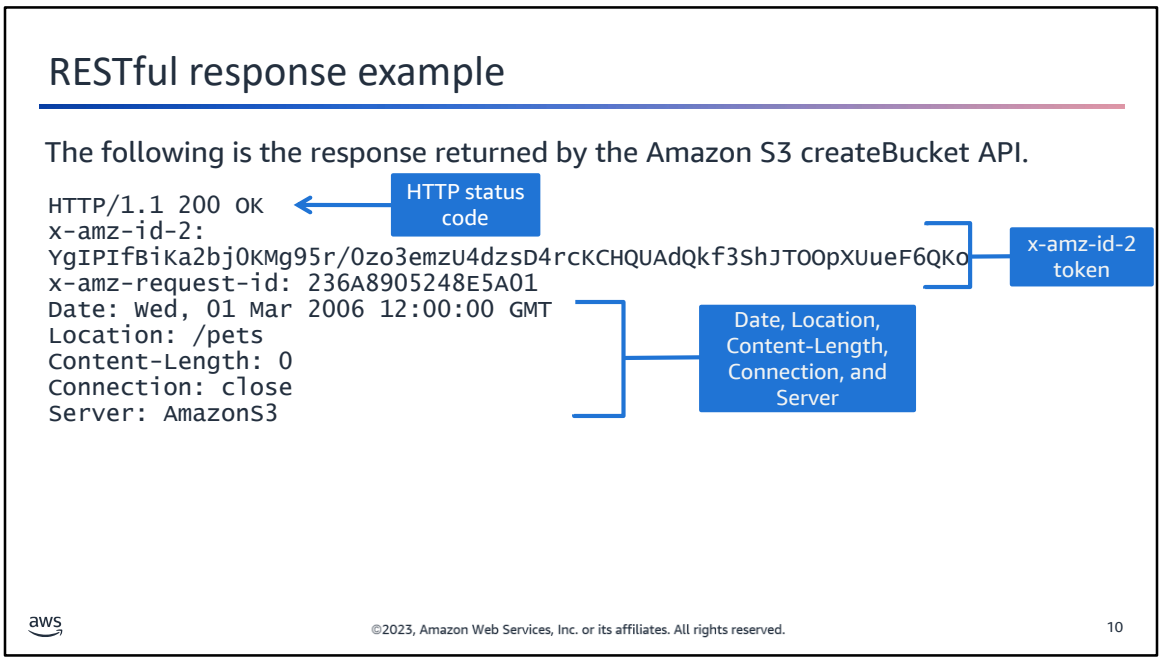
A REST request includes several components:

- Endpoint: When you make a request to a REST server, you must know the endpoint. The endpoint is in the form of a URL. The URL provides a way for the client to notify the server about the resources that it wants to interact with.
- Method: Each resource that is exposed in the REST API supports one or more methods, which include the following:
    - GET: The server is requested to retrieve a resource.
    - POST: The server is requested to create a new resource.
    - PUT: The server is requested to edit or update an existing resource.
    - DELETE: The server is requested to delete a resource.
- Header: The header contains the request's metadata.
- Body of the request: The body of a request is the data that the client sends to the server. A POST or PUT request typically contains a body. For example, if you invoke a PUT method, your request should most likely include details that indicate how an existing resource should be updated. However, GET requests rarely include a body.



Consider creating a bucket named pets in Amazon Simple Storage Service (Amazon S3). Take a look at the code. If you recall, the PUT API method primarily is used to update existing resources and can also be use to create a resource.

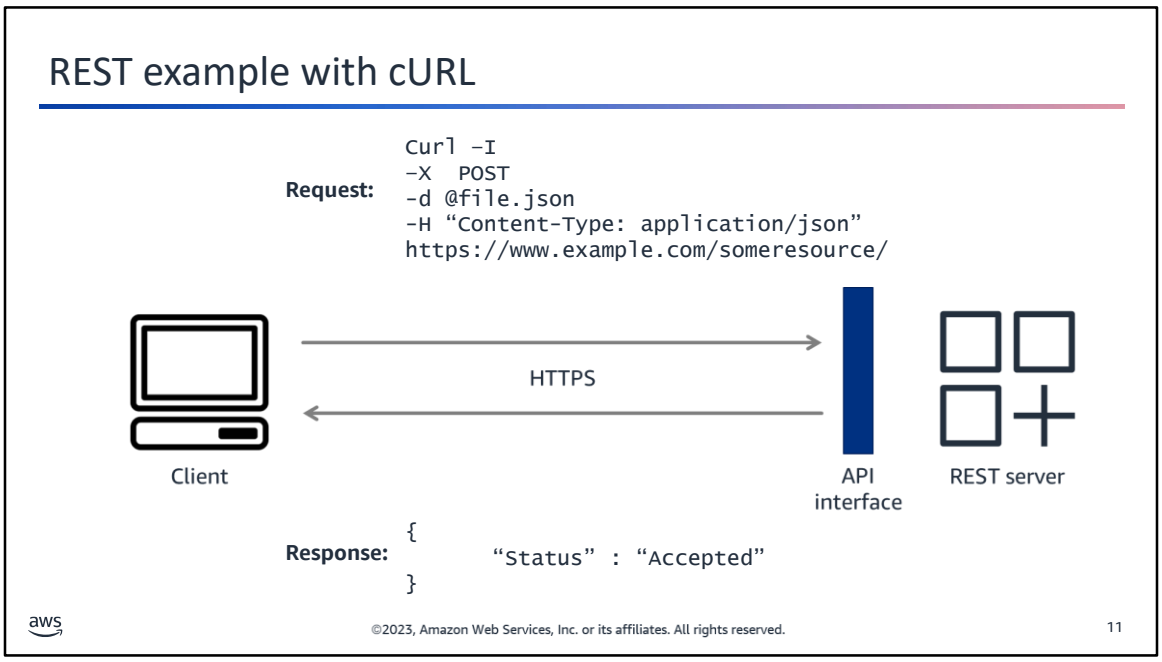
- Bucket name: pets
- Endpoints for Amazon S3 with the follow structure: s3.<Region>.amazonaws.com
- Content-Length: Length of the message (without the headers)
    - In the example, the content length is 0, and the resource URI is blank (/) because the client does not need to send anything to the server to create the bucket.
- Date: The date and time of the request
- Authorization: The credential information to authenticate the request to the server



Take a moment to examine the response header. The following information provides you with details about the response header.

The response shows the following:

- HTTP status code of 200 OK: Indication of success. The server received and accepted the request.
- x-amz-id-2: Special token that is used together with the x-amz-request-id header to help AWS troubleshoot problems.
- Date: Date and time Amazon S3 responded, for example, Wed, 01 Mar 2006 12:00:00 GMT.
- Location: The location of a newly created resource in Amazon S3 (bucket name).
- Content-Length: The length in bytes of the body in the response.
- Connection: Information about whether the connection to the server is open or closed.
- Server: The name of the server that created the response.



Client URL (cURL) is a popular command line tool that you can use to get or send files over HTTP(S). This tool is useful for testing REST API access. In this example, the user is creating a file on a server.

In the example, note the following:

- –i indicates that the response should include the response headers.
- –X indicates the method that you are invoking, for example, POST.
- –d is commonly used with POST or PUT methods and provides the data (body) that is being sent to the REST server. In this case, the data is contained in a file that is called file.json.
- –H passes header information. The header information can contain information, such as the content type, which an acceptable response will include.

A request also must provide credentials for authentication in the header of the request. This example does not include authentication credentials.

The example response is in JSON format.

HTTP status codes

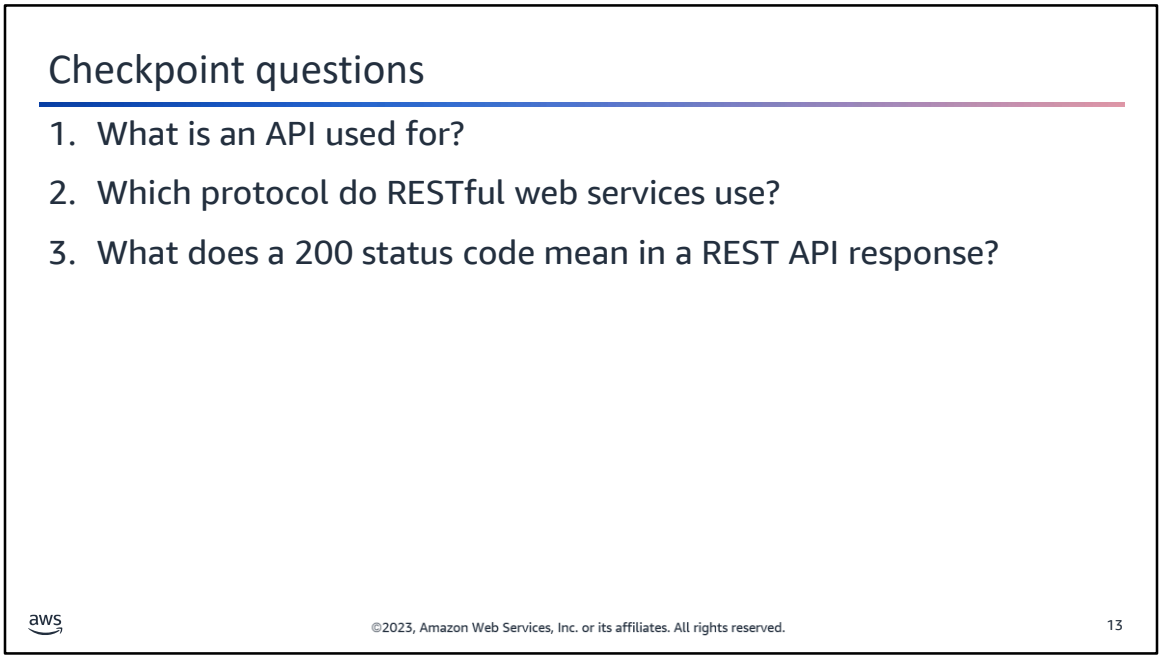
Status code group	Description	Examples
1xx	Informational response	100: Indicates informational response code that everything so far is OK and that the client should continue with the request.
2xx	Success	200: Indicates success. The server received and accepted the request.
3xx	Redirection	307: Indicates that the target resource resides temporarily under a different URI.
4xx	Client errors	401: Indicates a client error. Unauthorized. Authentication is required, but the provided credentials were not accepted, or perhaps no credentials were provided in the request.
5xx	Server errors	503: Indicates that the service is unavailable. Try again later.

An HTTP status code is included in REST API responses. It is helpful to know what the codes mean because the code indicates whether the request was received successfully or if there were errors. Understanding what status codes indicate—especially the error codes—can be helpful when you troubleshoot REST API requests.

This slide includes the groups of status codes for your reference.

Hundreds of status codes exist, but they are grouped so that the first of the three digits will indicate the general nature of the status.

****For accessibility:** HTTP status codes table. The first column provides status code groups. The second column describes the status code. The third column provides an example**. End description.**



The answers to the questions are as follows:

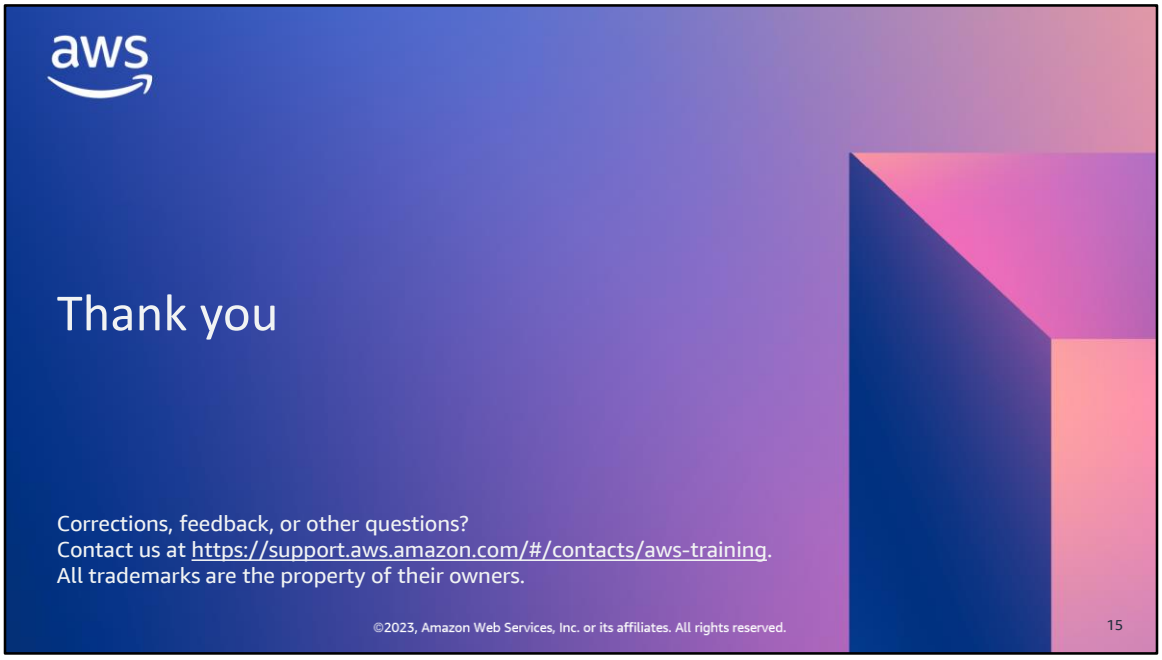
1. What is an API used for?
   You can use APIs to interact programmatically with a separate application or software.

2. Which protocol do RESTful web services use?
   RESTful web services use the HTTP protocol. HTTP standard methods are used to access resources in a RESTful architecture.

3. What does a 200 status code mean in a REST API response?
   A 200 status code indicates success. The server received and accepted the request.

Key Ideas

- APIs provide programmatic access to an application and are often used for programs that communicate with each other.
- The design principles of REST are uniform interface, statelessness, cacheability, layered system, and, optionally, code on demand.
- The components of a REST request are endpoint (URL), method, header, and body.
- HTTP status codes can be helpful in troubleshooting REST API errors.



Thank you

Corrections, feedback, or other questions?
Contact us at https://support.aws.amazon.com/#/contacts/aws-training.
All trademarks are the property of their owners.