API REST

Cruz Arredondo Jose David. Monday November 21, 2023

***Summary*—An API REST is an application programming interface that follows the principles of REST (Representational State Transfer) architecture. REST is a style of designing networked applications that uses HTTP as the main protocol for communication and data exchange. An API REST allows different applications or devices to connect and interact with each other through a uniform interface that uses HTTP methods (such as GET, POST, PUT, DELETE) and supports various data formats (such as XML, JSON, HTML).**

I. INTRODUCTION

An API (Application Programming Interface) is a set of rules and specifications that define how different software components or applications can communicate and exchange data with each other.

REST (Representational State Transfer) is an architectural style for designing networked applications that use HTTP as the main protocol for communication and data exchange.

II. REST CHARACTERISTICS

Resource: A resource is any piece of information or data that can be identified by a unique identifier (URI). A resource can be anything, such as a document, an image, a video, a user, a product, etc. A resource can also have different representations, such as XML, JSON, HTML, etc.

Uniform interface: A uniform interface is a set of rules and constraints that define how the client and the server interact with each other.

Stateless: A stateless communication means that each request from the client to the server is independent and contains all the information needed to process it. The server does not store any information about the client's state or session, which improves scalability and performance.

Cacheable: A cacheable communication means that the client or the server can store the response of a request and reuse it for future requests, which reduces the network traffic and improves efficiency. The server indicates whether a response is cacheable or not by using HTTP headers, such as Cache-Control, Expires, ETag, etc.

Layered system: A layered system means that the client and the server can be separated by intermediate components, such as proxies, gateways, firewalls, etc. These components can provide additional functionality, such as load balancing, security, caching, etc. The client and the server are not aware of the existence of these components, which increases modularity and flexibility.

Code on demand (optional): Code on demand means that the server can send executable code to the client, such as JavaScript, Java applets, etc. This code can extend or customize the functionality of the client, which enhances the user experience.

III. USES OF API REST

There are many guidelines and best practices for designing an API REST, but some of the most important ones are:

1) HTTP Verbs

They are those verbs typical of the HTTP protocol that were taken to define very specific and specific operations on the API resources:

GET: list of resources. Detail of a single resource.

POST: creation of a resource.

PUT: total modification of a resource.

PATCH: partial modification of a resource.

DELETE: deletion of a resource. In many cases it is a soft deletion, that is, a resource is not permanently deleted but is only marked as deleted or deactivated.

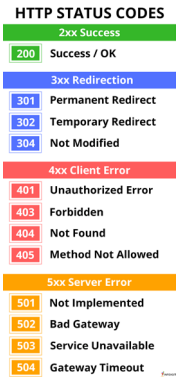
Resource-oriented URL: The definition of API endpoint URLs are resource-oriented, that is, entities that have consistency within the context of the API. For example, in an API for a system that manages books it would be easy to find entities such as books, authors, publishers, collections, etc. [1]

* UTT Cruz Jose

2) HTTP STATUS

They are those response statuses typical of the HTTP protocol that were taken to inform about the result of the requested operation. The most common ones in REST API are:

200 - OK
201 - Created
204 - No Content
400 - Bad Request
401 - Unauthorized
403 - Forbidden
404 - Not Found
500 - Internal Server Error



These statuses are divides in differents layers: Success, Redirecction, Client Error and Server Error.

IV. DIFFERENCE BETWEEN RESTFUL AND RESTLESS

RESTful Web Service:

1) It is a web service that uses REST architecture.
2) It uses request and response type requests with REST principles[1].
3) It supports JSON, HTML, etc format.
4) These services use URL to show business logic.
5) It is more usable and flexible for the users.
6) Less secure as it uses the security layers to communication protocols.
7) It uses a large bandwidth.

RESTless Web Service:

1) It is a web service that does not obey REST architecture.
2) It uses an XML document to send and receive messages.
3) It supports only XML format.
4) These services use service interface to show business logic.
5) It is less usable and flexible for the users.
6) More secure as it designs its own security layer.

V. REST API EXAMPLES

Amazon S3:

With many top-tier companies offering these services, the use of REST APIs for artificial intelligence, data science, and machine learning applications is on the rise. AWS AI Services from Amazon allows developers to incorporate AI functionality into their applications for a more adaptive and intelligent interaction. This can also help to secure data exchange between systems by detecting potential security vulnerabilities.

Twitter (X):

With 450 million monthly active users, Twitter has an enormous reach in the realm of social media. For developers, the Twitter API offers a way to integrate Twitter's functionality and promote their applications through the platform.

Using the Twitter API, developers can streamline the registration process by leveraging Twitter's identification system. The API also enables the display of tweets to users based on criteria such as location or trending hashtags, as well as effective marketing using Twitter's data.

Instagram:

The Instagram Basic Display API provides developers with access to profile data, images, and videos on the platform. This allows developers to build apps that integrate user data from Instagram into their own products. Additionally, Instagram offers a Graph API for professional accounts, enabling users to manage their online activities.

VI. CONCLUSION

Client-server architecture: The client and the server are separate entities that communicate through a standardized interface, such as HTTP. The client is responsible for initiating requests and handling responses, while the server is responsible for processing requests and providing resources.

Statelessness: The server does not store any information about the client's state or session. Each request from the client contains all the necessary information for the server to fulfill it. This improves scalability, performance, and reliability of the web service. [2]

VII. CONCLUSION

REST API is a popular architectural style for building web services. It is based on the HTTP protocol and uses HTTP methods such as GET, POST, PUT, and DELETE to perform operations on resources. RESTful APIs are lightweight, scalable, and easy to maintain. They are widely used in modern web development.

In conclusion, RESTful APIs are a powerful tool for building web services. They are easy to use, scalable, and flexible.

VIII. REFERENCES

[1] Rest api (no date) Red Hat - We make open source technologies for the enterprise. Available at: https://www.redhat.com/es/topics/api/what-is-a-rest-api (Accessed: 21 November 2023).

[2[Team, T.P. (2023) What is a rest api? examples, uses & challenges, Postman Blog. Available at: https://blog.postman.com/rest-api-examples/ (Accessed: 21 November 2023).