Valerie Wade Writing Sample
June 2021
https://vcwade.com/tech

Introduction to REST
REST (REpresentational State Transfer) is an architectural style for providing standards between computer systems on the internet. REST was invented by computer scientist Roy Fielding. A REST API is an application programming interface that conforms to the REST architectural style.

REST is easy to learn once you understand how it's already part of your daily life. Below are some examples of REST APIs you probably use regularly:
- Pokémon's REST API lets developers build mobile applications to retrieve updates on all of the Pokémon characters such as their moves, abilities, and even egg groups.
- Facebook's REST APIs syndicate your new posts to your friends and other users in your network.
- Uber notifies curbside customers with up-to-date information when their driver will arrive at their location using Twilio's SMS REST API.
- Flickr REST APIs help users embed images on their website and social media profiles.

REST is a way for one software system to communicate with another behind the scenes. REST is like using a website to get information, except it doesn't require the website itself. Instead, REST commands are done
using code requests and responses. A request asks to get something back from a resource.

Resources
With REST APIs, a resource is an object with a type. It contains data that's connected to other pieces of data. Resources can have relationships to other resources.

HTTPS
REST communicates using HyperText Transfer Protocol Secure (HTTPS) methods to keep your calls to APIs safe and secure. HTTPS is a combination of
[HTTP](https://en.wikipedia.org/wiki/Hypertext_Transfer_Protocol) + [TLS](https://en.wikipedia.org/wiki/Transport_Layer_Security). Some API providers use HTTP (no S), however, it’s not secure. If you try making calls to our APIs using HTTP, they won’t work. HTTPS sends data between the client and the server using an encrypted SSL certificate to keep all transmissions secure and safe. Typing `https` into a browser's address bar tells the browser to make the HTTPS request to a resource on a given server.
Entity	Description

Client	A computer or cell phone making a request for a data resource.
User-agent	A tool to make requests such as a web browser like Chrome, Firefox, Safari, or Internet Explorer.
Server	One or many computers storing the resource.

HTTP Methods
In the world of REST APIs, HTTP methods are also called verbs. They're used to communicate between your client (for example, your iPhone) and the servers. Below are some common methods:
Method	Description
GET	Requests access to information or a resource.
POST	Creates a new resource in a collection of resources.
PUT	Tells the server to replace a resource.
DELETE	Notifies the server to delete a resource.
PATCH	Updates a resource of collection of resources.

Endpoints
To make a request, you need to know where it's being sent right? That's where endpoints come in. For example, when you send a letter with the United States Postal Service (USPS), the envelope is addressed to someone. Endpoints are the same thing but formatted with a Uniform Resource Locator (URL) like the following example:
```
https://wikipedia.org/wiki
\___/ \____________/ \______/
| | |
scheme root-endpoint path
```

HTTP Headers
HTTP headers validate that the request is coming from an authorized user. They also provide metadata about the message body. The name of the header is separated from the value by a colon.

Common Headers
It can be confusing to understand headers without real-world examples, so let's look at some examples of common headers and what they do.

When you do a Google search for REST APIs, the browser makes a request to a Google server asking it to return the results your browser is looking for in the request header. The `Accept` header tells the Google server what file formats (MIMEtype) the browser wants back to display on the page. For example, the server can send plain text, HTML, JSON, etc. Below is an example of Firefox's Accept header:

GET /page/routing-in-recess-screencast HTTP/1.1

Host: RecessFramework.org

Accept:text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8

The `Content-Type` header asks for details about the media format of the resource like in the example below:

```
Content-Type: text/html; charset=utf-8

Content-Type: multipart/form-data; boundary=something
```

The `Authorization` request header contains the user credentials to authenticate a user agent with an API key like below:

```
GET /something HTTP/1.1

X-API-Key: abcdef12345
```

Requests
A request passes the client information to the server. It's a text record sent by a web browser to a server. It contains details about the information the browser wants returned in the server response. Learn about [the anatomy of an HTTP request](https://gavilan.blog/2019/01/03/anatomy-of-an-http-request/).

Responses
Responses are returned using [JavaScript Object Notation (JSON)](https://www.json.org/json-en.html). JSON a lightweight data-interchange format.