**Curso de API REST con PHP**

**Api Rest**

**¿Qué es un API y para que sirve?**

API: Application Programming Interface, conjunto de reglas que definen como se van a comunicar dos aplicaciones, por ejemplo qué tipos de request y response se van a utilizar.

En **POO** una API son todos los elemento públicos que están disponibles.

**Conocer conceptos principales de Rest**

**¿Qué es y cómo funciona el protocolo HTTP?**

HTTP: Hytpertext Transfer Protocol, conjunto de reglas en las que se van a comunicar dos entidades.

Protocolo: conjunto de reglas.

**¿Qué significa Rest?**

REST: Representational State Transfer o bien transferencia de estado representacional.

API RESTful es una API diseñada con los conceptos de REST.

* Recurso: todo dentro de una Api RESTFul debe ser un recurso.
* URI: Todos los recursos se manipulan a partir de una URL (Identificadores Universales de Recursos).
* Acction: todas las peticiones de un Api Rest debe de estar asociado con un verbo HTTP:
  + Get: para obtener recursos.
  + Post: para escribir un recurso.
  + Put: Para editar un recurso.
  + Delete: para eliminar un recurso.

**Métodos de autenticación**

**HTTP**

* Mediante HTTP auth user, debe de ir en el encabezado authorization.

**HMAC**

* HMAC: (Hash Message Authorizarion Code) para lo cual se necesita 3 elementos, Basado en Hash de Mensajes:
  + Función hash: Difícil de romper y que sea conocida por el cliente y el servidor.
  + Clave Secreta: para corroborar el HASH
  + UID: Id de usuario será utilizado dentro del HASH junto con la clave secreta y el timestamp

Es mucho más seguro que la autenticación vía HTTP.

**Acceso mediante Tokens**

Roles del servidores:

* Uno se encarga de la autenticación.
* Otro se encarga de desplegar los recursos del API.

Flujo.

* Petición para la solicitud de Token
* El servidor devuelve token
* El usuario hace una petición para pedir recurso.
* El servidor con los recursos hace una petición al servidor de autenticación para verificar token sea valido.
* Una vez verificado el token, el servidor devuelve los recursos al cliente.

**Manejo de Errores en servicio REST**

Para las respuestas con error se utilizaran los siguientes códigos: en los encabezados HTTP.

* 400: Bad Request, que el servidor no puede procesar la petición.
* 404: Not Found, el servidor no encuentra el recurso buscado.
* 500 Invernal Error Server, la petición no se pudo procesar por un error en el servidor.

**Buenas prácticas**

* Utilizar sustantivos para nombrar los recursos.
* Añadir los nombres en plural para las URLs
* Las modificaciones se deben de hacer con su verbo HTTP correspondiente.
* Para devolver recursos asociados se debe incorporar sub-recursos

autos/1/choferes

* Navegabilida vía vínculos
* Cuando devuelva colecciones deben de ser filtrables, ordenables y paginables.
* Versionar las api

**Instalación de chocolatey**

Se instala este programa para posteriormente instalar el **curl**

 **Abre el símbolo del sistema como administrador**:

* Busca "cmd" en el menú Inicio.
* Haz clic derecho en "Símbolo del sistema" y selecciona "Ejecutar como administrador".

 **Ejecuta el comando de instalación de Chocolatey**:

* Copia y pega el siguiente comando en la ventana del símbolo del sistema y presiona Enter:

@"%SystemRoot%\System32\WindowsPowerShell\v1.0\powershell.exe" -NoProfile -InputFormat None -ExecutionPolicy Bypass -Command "Set-ExecutionPolicy Bypass -Scope Process -Force; [System.Net.ServicePointManager]::SecurityProtocol = [System.Net.ServicePointManager]::SecurityProtocol -bor 3072; iex ((New-Object System.Net.WebClient).DownloadString('https://community.chocolatey.org/install.ps1'))" && SET "PATH=%PATH%;%ALLUSERSPROFILE%\chocolatey\bin"

**Verifica la instalación**:

* Una vez completada la instalación, escribe choco -v en el símbolo del sistema y presiona Enter para verificar que Chocolatey está instalado correctamente. Esto debería mostrar la versión instalada.

choco –v

**Instalar Curl**

Chocolatey facilita la instalación. Simplemente ejecuta el siguiente comando en un símbolo del sistema con permisos de administrador:

**choco install curl –y**

* Cuando te pregunte, selecciona la opción Y para continuar la instalación
* Verifica tu instalación ejecutando el comando curl <https://platzi.com>

**Inastalar JQ**

Una vez dentro de la consola copia y pega este comando:

choco install jq

Verifica tu instalación ejecutando el comando

jq

¡Todo listo! Ya tienes instaladas las herramientas básicas para trabajar con los ejemplos del curso de REST API.

**COMANDOS**

Ver la comunicación completa con el servidor mediante la api usa –v

**curl "http://localhost:8080/cursoApiRest/server.php?" –v**

\* Host localhost:8080 was resolved.

\* IPv6: ::1

\* IPv4: 127.0.0.1

\* Trying [::1]:8080...

\* Connected to localhost (::1) port 8080

> GET /cursoApiRest/server.php HTTP/1.1

> Host: localhost:8080

> User-Agent: curl/8.9.1

> Accept: \*/\*

>

< HTTP/1.1 200 OK

< Date: Wed, 19 Mar 2025 17:56:58 GMT

< Server: Apache/2.4.25 (Win32) OpenSSL/1.0.2j PHP/7.1.1

< X-Powered-By: PHP/7.1.1

< Content-Length: 0

< Content-Type: text/html; charset=UTF-8

Ver la colección de libros

**curl “http://localhost:8080/cursoApiRest/server.php?resource\_type=books”**

Ver la colección de libros pero con el jq

**curl "http://localhost:8080/cursoApiRest/server.php?resource\_type=books" | jq**

Consultar un libro en especifico

**curl “http://localhost:8080/cursoApiRest/server.php?resource\_type=books&resource\_id=1”**

Consultar los libros pero con urls amigables desde el route.php

**curl http://localhost:8080/cursoApiRest/router.php/books**

Con un id de libro en especifico

**curl http://localhost:8080/cursoApiRest/router.php/books/1**

Como mandar un post con curl –X es para indicar que usaremos una petición de tipo POST

El –d para indicar la información

**curl -X "POST" http://localhost:8080/cursoApiRest/router.php/books -d "{ \"titulo\":\"Nuevo Libro\",\"id\_autor\": 1,\"id\_genero\": 2}"**

Como actualizar un dato con PUT es indicar “PUT”

**curl -X "PUT" http://localhost:8080/cursoApiRest/router.php/books/1 -d "{ \"titulo\":\"Nuevo Libro\",\"id\_autor\": 1,\"id\_genero\": 2}"**

Para eliminar es parecido, hay que cambiar a DELETE

**curl -X "DELETE" http://localhost:8080/cursoApiRest/router.php/books/1**

**AUTENTIFICACIÓN POR HTTP**

Para consultarlo en el curl hay que especificar el usuario:password dentro de la liga

curl http://**admin:1234@**localhost:8080/cursoApiRest/router.php/books

agregar el sig código

if (array\_key\_exists('PHP\_AUTH\_USER', $\_SERVER)) {

    $user = $\_SERVER['PHP\_AUTH\_USER'];

} else {

    $user = '';

}

if (array\_key\_exists('PHP\_AUTH\_PW', $\_SERVER)) {

    $pwd = $\_SERVER['PHP\_AUTH\_PW'];

} else {

    $pwd = '';

}

if ($user != 'admin' || $pwd != '1234') {

    die;

}

**METODO HMAC**

Código de autorización basado en Hash de infomación, se basa en un hash difícil de romper y sea conocida por el cliente y el servidor, palabra secreta que esta es compartida por el servidor y el cliente

Ejecutar el archivo para generar el hash y el timestamp

C:\AppServ\www\cursoApiRest>**php generate\_hash.php 1**

Time: **1742491311**

Hash: 5a940710c34c0c7a04b4580ab0485b6205bfd821

Y luego el sig comando

C:\AppServ\www\cursoApiRest>curl http://localhost:8080/cursoApiRest/router.php/books/1 -H "X-HASH: **5a940710c34c0c7a04b4580ab0485b6205bfd821**" -H "X-UID: 1" -H "X-TIMESTAMP: **1742491311**"

**METODO POR ACCESS TOKEN**

Es por tokens temporales de acceso, este método parte en dos el servidor una parte destinada para la autentificación

curl http://localhost:8080/cursoApiRest/auth\_server.php -X "POST" -H "X-Client-id:1" -H "X-Secret:SuperSecreto!"

**d2d957cfdbd61dd965ec9e727ace9e2c35361b43**

curl http://localhost:8080/cursoApiRest/router.php/books/1 -H "X-Token:**d2d957cfdbd61dd965ec9e727ace9e2c35361b43**"

**MANEJO DE ERRORES**

Crear un archivo de cliente para hacer la petición a http://localhost:8080/cursoApiRest/router.php/books/1

C:\AppServ\www\cursoApiRest>php client.php http://localhost:8080/cursoApiRest/router.php/books/1

También del lado del servidor se tienen que declarar el manejo de errores

if(!in\_array($resourceType, $allowedResourceTypes)){

    // avisar del codigo de estado 400 Bad Request

    http\_response\_code(400);

    die;

}

EL POST se necesitan madnar toda la colección de datos

EL DELETE es indicar el id del recurso

La autenticación vía HTTP tiene dos problemas:

Es poco segura: las credenciales se envían en cada request anteponiendo el usuario y contraseña en la url, por ejemplo: user:password@platzi.com.

Es ineficiente: la autenticación se debe realizar en cada request.