P10. Codificación de la API REST con Node.js

En este capítulo se inicia con el desarrollo de la API REST, por lo que se verá la codificación de cada uno de los archivos que componen el proyecto. Para esto, es importante tener en cuenta dos cosas, la primera es realizar buenas prácticas de codificación, permitiendo que el código quede legible y mucho más limpio al momento de documentar y la segunda, es entender la estructura de directorios del proyecto, la cual va a dar el orden al proceso de búsqueda e indexación de las rutas y recursos que se necesitarán al momento del desarrollo.

1 Buenas prácticas en la codificación de Node.js

El objetivo de seguir estándares de codificación es alcanzar un código mantenible, y se considera mantenible cuando el código es legible, consistente, predictivo y documentado. En la práctica, la corrección de errores y modificación del código desarrollado de un producto software toma demasiado tiempo y se llega algunas veces a que no es posible realizar del todo el desarrollo, por lo que se toma la decisión de iniciar de cero, debido a las malas prácticas

A continuación, se indican las buenas prácticas enunciadas por uno de los mejores servicios de control de versiones y desarrollo de software colaborativo basado en Git, llamada GitLab. Esta comunidad sigue de igual manera estándares como Google JavaScript Style Guide.

2 Desarrollo de la API REST

A continuación, se describen de manera sucinta los pasos requeridos para el desarrollo de la API REST con Node.js.

P11. Testing de una API REST con Postman

1. ¿Qué es Postman?

- Es una herramienta utilizada para **probar APIs REST**.
- Permite enviar solicitudes HTTP como: **GET, POST, PUT, DELETE, PATCH**, etc.
- Facilita la **automatización de pruebas**, la **documentación** y el **manejo de colecciones**.
- Muy usado por desarrolladores, QA testers y equipos DevOps.

---

2. ¿Qué es una API REST?

Una **API REST** es un servicio que permite la comunicación entre aplicaciones usando el protocolo HTTP.
Sus características principales:

- Utiliza métodos HTTP estándar.
- Intercambia datos normalmente en **JSON**.
- Es stateless (no guarda estado entre peticiones).

---

3. Beneficios de usar Postman para pruebas de API

- Interfaz visual fácil de usar.
- Permite **probar endpoints rápidamente**.
- Soporta **variables globales, de entorno y colección**.
- Se pueden crear **scripts de prueba** con JavaScript.
- Permite **automatizar pruebas con colecciones**.
- Importar y exportar pruebas para compartir con el equipo.

---

4. Tipos de pruebas que puedes hacer en Postman

✓ **Pruebas manuales**

- Enviar solicitudes individuales.
- Verificar respuestas: **status code, headers, body y tiempos de respuesta**.

✓ **Pruebas automatizadas**

- Usando **Tests** (scripts en JavaScript).
- Ejecutar colecciones completas con **Collection Runner**.

✓ **Pruebas de regresión**

- Probar nuevamente endpoints cada vez que se actualiza la API.

✓ **Pruebas de integración**

- Validar flujo entre varios servicios conectados.

P12.5. Elementos clave en Postman

📌 **1. Colecciones**

Organizan tus peticiones en carpetas.

📌 **2. Environments (Entornos)**

Permiten cambiar variables fácilmente:
{{url}}, {{token}}, {{idUsuario}}

📌 **3. Request (Solicitud)**

Incluye:

- URL del endpoint
- Método HTTP
- Headers

- Body (JSON)

📌 **4. Test Scripts**

Código en JavaScript para validar la respuesta.
Ejemplo:

```
pm.test("Código 200 OK", function () {
  pm.response.to.have.status(200);
});
```

---

6. Pruebas básicas que debes realizar a una API en Postman

Tipo de prueba	**¿Qué se valida?**
Status Code	200, 201, 400, 404, 500…
Estructura JSON	Campos esperados
Headers	Content-Type, Authorization
Tiempo de respuesta	< 1s o según SLA
Autenticación	Tokens, API Keys, JWT
Errores	Respuestas de error controladas

---

7. Ejemplo de Prueba en Postman — GET

Solicitud

GET https://api.ejemplo.com/usuarios

Script de prueba

```
pm.test("Status es 200", () => pm.response.to.have.status(200));
pm.test("El cuerpo tiene usuarios", () => {
  const body = pm.response.json();
  pm.expect(body).to.be.an("array");
});
```

P13.8. Testing Automatizado con Collection Runner

Permite:

- Ejecutar toda la colección.

- Validar cientos de peticiones.
- Exportar reportes de ejecución.
- Integración con CI/CD.

Ideal para QA.

---

9. Buenas prácticas para probar APIs

- Usar entornos (dev, qa, prod)
- Separar pruebas por módulos.
- Crear variables reutilizables.
- Usar scripts pre-request y tests.
- Validar casos exitosos y fallidos.
- Documentar cada endpoint.

---

10. Conclusión

Postman es una herramienta completa que facilita el **testing manual y automatizado** de APIs REST, permitiendo validar su funcionamiento, rendimiento y seguridad de manera eficiente.

P14.Testing de la API REST con Postman

Con el fin de revisar y evaluar el funcionamiento de la API se utilizará la herramienta Postman, que es una herramienta que se utiliza especialmente para el testing de API REST, aunque también admite otras funcionalidades que se salen de lo que engloba el testing de este tipo de tecnologías.

Con Postman, además de testear, consumir y depurar API REST, se puede monitorearlas y escribir pruebas automatizadas para ellas, documentarlas y simularlas. Esta herramienta es favorable para equipos con poca experiencia en programación y para hacer testing de todo tipo en general de API REST (López, 2021).

La instalación de esta herramienta es rápida y sencilla; son pocos los pasos para su configuración, al respecto se puede instalar en el equipo de manera persistente o también como una extensión de Google Chrome (Chrome, 2021).

Para finalizar, conozca la instalación de la herramienta Postman.

P20. Servicios web con PHP

Los servicios web, también conocidos como *web services*, son un conjunto de protocolos que facilitan la comunicación entre dispositivos, permitiendo

intercambiar información: “datos”. Un servicio web tiene una interfaz que oculta los detalles de implementación, para que se pueda utilizar independientemente de la plataforma de *hardware* o *software* en la que se implementa, e independientemente del lenguaje de programación en el que está escrito.

los servicios web se han convertido en la respuesta a esta necesidad ya que son el conjunto de protocolos que permiten la interacción entre dispositivos servidores y computadores tabletas smartphones etcétera, en una red la interacción ocurre cuando se envían solicitudes y respuestas a través de un cliente, dispositivo y un servidor. además, existen muchos lenguajes de programación sistemas operativos y tecnologías por ello es importante encontrar una comunicación estándar entre estos y es necesario utilizar un

P21. "Servicios web con PHP" se refiere a la creación y uso de servicios web utilizando el lenguaje de programación PHP.

◈ ¿Qué es un servicio web?

Un **servicio web** es una aplicación que permite la comunicación y el intercambio de datos entre diferentes sistemas a través de la web. Se basa en estándares como **REST** y **SOAP**, y usa formatos como **JSON** y **XML** para la transmisión de datos.

◈ ¿Cómo se implementan con PHP?

En PHP, los servicios web se pueden desarrollar utilizando varias herramientas y enfoques, como:
✅ **RESTful APIs**: Se implementan con frameworks como **Laravel, Slim o Lumen**, o simplemente con código PHP nativo usando $_GET y $_POST.
✅ **SOAP**: Se pueden crear servicios web SOAP con la extensión SoapServer y SoapClient.
✅ **Frameworks**: Laravel facilita la creación de APIs con recursos como **Laravel Passport** o **Laravel Sanctum** para autenticación.

P22. Métodos o funciones

los métodos o funciones permiten segmentar el trabajo que hace un programa en subtareas o tareas más pequeñas, enfocadas en un fin específico, y se pueden utilizar cuantas veces se necesite; estas están separadas del programa principal, pero aportan a su objetivo.

El uso de funciones da la capacidad de agrupar varias instrucciones de código bajo un solo nombre, y con el nombre de la función, es posible invocarlo las veces que se necesite, permitiendo usar esta función las veces que sean necesarias.

Ejemplo 1:

La función se define con la siguiente sintaxis:

Nota: el símbolo “//” quiere decir que no hace parte del código, se lo utiliza para hacer comentarios sobre el código, por lo cual se pone en color azul.

```
1 <?php

2 function media_aritmetica($a, $b){ //Aquí definimos la función y la llamamos media_arimetica y las variables a y b
3 $media=($a+$b)/2; //escribimos la fórmula de la media, en este caso de 2 números divididos entre 2
4 return $media; // aquí la función retorna el valor de la media
5
}
```

Tipos de visibilidad:

🔷 **Encapsulamiento**: Controla el acceso a las propiedades y métodos con public, private y protected.
🔷 **Herencia**: Permite que una clase herede atributos y métodos de otra.
🔷 **Polimorfismo**: Una función puede comportarse de manera diferente en distintas clases.
🔷 **Abstracción**: Define clases base que otras clases deben implementar.

P23. Clases y objetos

En PHP, las **clases** y **objetos** son fundamentales en la Programación Orientada a Objetos (**POO**).

🏗️ ¿Qué es una Clase?

Una **clase** es una plantilla o molde que define las propiedades (**atributos**) y comportamientos (**métodos**) de un objeto.

🎭 ¿Qué es un Objeto?

Un **objeto** es una instancia de una clase. Representa una entidad con características y acciones específicas.

Las clases son contenedores de información basados en atributos y en métodos de construcción, los cuales pueden modificar y consultar; son suficientes para representar objetos con los cuales se procesa información, y se usan para hacer objetos que tienen un mismo comportamiento, estado e identidad.

Ejemplo 2:

Se tiene una clase “personas” (Mario, Margarita y Alejandro), donde el comportamiento de las personas es leer, trabajar, jugar, etc.; pueden estar en estado despierto o dormido; sus propiedades pueden ser color de ojos, género, estado civil, etc.

Su sintaxis sería así:

```
class Persona {
```

# Propiedades
# Métodos
}

Objeto: el objeto es una instancia de una clase, la cual puede crear varias instancias de la misma clase.
Ejemplo: existe una sola clase “Persona”, pero muchos objetos de tipo persona pueden ser instancias de esta clase.

P24. A continuación, observe la diferencia entre clases y objetos con una analogía:

Analogía

Una clase

Es como el plano de un apartamento, este define la forma del apartamento sobre el papel, con las relaciones entre las diferentes partes claramente definidas y planeadas, a pesar de que el apartamento no exista.

Un objeto

El apartamento real construido de acuerdo a ese modelo; es decir, un objeto es una instancia de la clase.

Ejemplo

En el siguiente ejemplo, se observará el código de un objeto.

El símbolo “//” quiere decir que no hace parte del código, se lo utiliza para hacer comentarios sobre el código, por lo cual, se pone en color azul.

Ejemplo

```
1  <?php
2          class Persona // Se define la clase Persona
3          {
4
5              private $nombre; // Declaración del campo. Este
6  debe ser de carácter privado.
7
8           //Aquí adelante se utiliza un constructor, que es un método
9  que lleva el nombre de la clase y que será el responsable de
10 construir el objeto.
11             function Persona($nom) {
12                $this->nombre = $nom;// this hace referencia a
13 la instancia actual, y se le asigna el valor al constructor->
14             }
15
16
17             function setNombre($n) { // Métodos accesorios
18                $this->nombre = $n;     // Métodos de escritura
```

Resultado

Si se corriera el código, daría este resultado.



P25. Integración de aplicaciones (XML, SOAP, WSDL y UDDI)

Los servicios web permiten la nueva generación de aplicaciones basadas en Internet. Estos servicios admiten la comunicación de Internet de aplicación a aplicación, es decir, las aplicaciones en diferentes ubicaciones de la red se pueden integrar para funcionar como si fueran parte de un único sistema de *software* grande. Ejemplos de aplicaciones que los servicios web hacen posibles incluyen transacciones comerciales automatizadas y acceso directo (sin navegador) a dispositivos de escritorio y de mano, a las reservas, el comercio de acciones y los sistemas de seguimiento de pedidos.

3.1 XML “Extensible Markup Language”

XML (Extensible Markup Language) es un lenguaje de marcas que define un conjunto de reglas para la codificación de documentos, se utiliza para estructurar información en cualquier fichero que contenga texto y es muy usado en el medio, debido a que es un estándar abierto y libre, creado por el consorcio *World Wide Web*, W3C.

3.2 SOAP “Simple Object Access Protocol”

SOAP es un acrónimo de Protocolo Simple de Acceso a Objetos. Es un protocolo de mensajería, basado en XML, para intercambiar información entre computadoras, y es una aplicación de la especificación XML.

3.3 WSDL “Lenguaje de Descripción de Servicios Web”

“es un lenguaje por medio del cual un servicio web describe, entre otras cosas, qué hace o qué funcionalidad implementa. Es el lenguaje de la interfaz pública para los servicios web. Es una descripción basada en XML de los requisitos funcionales necesarios para establecer una comunicación con los servicios web”.

3.4 UDDI “Universal Description, Discovery, and Integration”

UDDI (Universal Description, Discovery, and Integration) “define un modo de publicar y encontrar información sobre servicios web, incluye un esquema XML para mensajes SOAP que define un conjunto de documentos para describir información de empresas y servicios, un conjunto común de API para consultar y publicar información en los directorios y una API para duplicar entradas de directorio entre nodos UDDI iguales”.

3.5 REST “Representational State Transfer”

El protocolo REST (Representational State Transfer) es una arquitectura cliente-servidor, en la cual un servicio es visto como un recurso que es identificado a través de una dirección URL, por medio de la cual puede ser accedido o consumido. Para acceder a estos servicios web, se hace uso de mensajes en formato simple, los cuales se intercambian entre cliente y servidor.

3.6 JSON “JavaScript Object Notation”

JSON (JavaScript Object Notation) es un formato ligero de intercambio de datos, es de fácil lectura y escritura para los usuarios, fácil de analizar y generar por parte de los dispositivos, además, se basa en un subconjunto del lenguaje de programación JavaScript; es compatible con la gran mayoría de lenguajes de programación, lo que lo hace el más popular y usado en el *web service*.

para hacer uso de esta comunicación, se hace a través de estos lenguajes o formatos: XML, HTML, JSON; este es el tipo de mensajes más difundido en diferentes servicios propios de redes sociales (*Facebook* y *Twitter*) y comunidades en Internet. Cada mensaje intercambiado contiene la información necesaria para el funcionamiento del servicio, de tal forma que, para cada servicio, el cliente y el servidor conocen el formato o protocolo interno de los mensajes.

P26. Creación de servicios

Indican que es una tecnología que establece unos estándares y protocolos que se utilizan para intercambiar datos entre distintas aplicaciones, plataformas y con diferentes o iguales lenguaje de programación. Por lo general, se utilizan estándares abiertos, lo que comúnmente se llama interoperabilidad; las organizaciones OASIS y W3C son los comités responsables de la arquitectura y la reglamentación de los servicios web.

su función es permitir que los dispositivos interactúen entre sí y presenten información dinámica al usuario, con base en una arquitectura de referencia estándar, que hace posible su combinación para realizar integraciones entre las mismas aplicaciones.

Conexiones a SQL

Para iniciar con ese tema, es preciso recordar que el SQL (Structured Query Language) es un lenguaje de consulta estructurado, definido como la base de datos relacionales con el lenguaje de alto nivel estándar.

el SQL agrupa tres tipos de sentencias con objetivos particulares, en los siguientes lenguajes:

1. **(DDL) Lenguaje de Definición de Datos**
2. **(DML) Lenguaje de Manipulación de Datos**
3. **(DCL) Lenguaje de Control de Datos**

P27.Servicios en PHP

PHP es uno de los lenguajes de código abierto más utilizados y adecuados para el desarrollo de aplicaciones y servicios web, este puede ser introducido en HTML. Por el hecho de ser un lenguaje gratuito, millones de páginas y portales web están creados con PHP. Su función es muy sencilla: el usuario utiliza aplicaciones o navegadores para realizar una solicitud a un servidor, el servidor recibe la petición, organiza la información que le solicitan y responde enviando la consulta por medio de una página web o aplicación “normal”, pero su creación es “dinámica”.

Aplicación SOAP UI

la aplicación SOAP UI esta es una herramienta muy versátil, que sirve para probar, simular y obtener el código de aplicaciones *web service* y la transferencia de estado representacional. Esta herramienta está generada en código java, funciona sobre SOAP, JMS, JDBC, API de REST, GraphQL y muchos servicios más, además de ser muy útil para que los futuros desarrolladores

Cómo usar Postman para Testear APIs (Práctico) | #3
https://www.youtube.com/watch?v=AtRBTki2oeY
Pagina de API para hacer prubas con postman
https://airportgap.com/docs#authentication

10 Primeros Pasos en Pruebas de API desde Cero con Postman

https://www.youtube.com/watch?v=wu5XOzAXNFE