

# **EA1: API REST NodeJs**

POR

Camilo Andres Olea Aguirre

Maria Catalina Alvarez Henao

Juan Camilo Jimenez Vasquez

PRESENTADO A

Jorge Armando Julio Cruz

Docente Ingeniería Web II

Grupo 140

INSTITUCIÓN UNIVERSITARIA DIGITAL DE ANTIOQUIA

Medellín - Septiembre 12 de 2022

**INTRODUCCIÓN**

En esta actividad de aprendizaje grupal se pretende Diseñar y construir una API REST con NodeJS que permita llevar el control del inventario de todos los equipos electrónicos con los que cuenta actualmente la Universidad Digital de Antioquia en un aplicativo web.

De manera introductoria se tomarán textualmente algunas definiciones y conceptos de consideración para nuestro proyecto, estos apuntes se encuentran en el artículo publicado en el año 2019 por la Revista Ibérica de Sistemas y Tecnologías de la Información, titulado Desarrollo Backend para aplicaciones web, servicios web restful: Node, js vs spring boot y escrito por Haro, E.,Guarda, T., Peñaherrera A. Y son los siguientes:

REST es un tipo de arquitectura orientada a recursos para el desarrollo de servicios web, con la que se expone el servicio web para ser consumido por los clientes sin la necesidad de programación exclusiva para una plataforma determinada o limitada por el lenguaje o hardware que usa la web como una plataforma de procesamiento distribuido

Los servicios web basados en la arquitectura REST (Representational State Transfer), son una alternativa más simple a SOAP (Simple Object Access Protocol) y a WSDL (Web Services Description Language) y permite transmitir los datos sobre el protocolo estandarizado HTTP utilizando una URI. Varias empresas como Google, Facebook y Yahoo! son casos de éxito al migrar sus servicios a esta tecnología.

Node JS es un entorno diseñado para construir aplicaciones de red escalables, es un framework que permite ejecutar código en JavaScript de parte del servidor que al combinarse con Express.js, permite generar enrutamiento para trabajar sobre el protocolo HTTP.

Para realizar la parte práctica de la implementación de API REST mediante el uso de Node.js utilizaremos como IDE de desarrollo Visual Studio Code, como lenguaje de programación JavaScript en la plataforma de Node.js, como gestor de dependencias utilizaremos NPM y finalmente como gestor de base de datos MongoDB.

Esperamos cumplir con los objetivos planteados para esta actividad práctica de desarrollo de software perteneciente a la materia Ingeniería Web II y agradecemos la oportunidad que se nos brinda de ampliar nuestros conocimientos e implementarlos a nivel grupal.

***EXPLICACIÓN DE CADA SERVICIO REALIZADO***

1. Instalación de Node Js, Visual Studio Code, Yarn y MongoDB y Postman API.
2. Se crean las carpetas del proyecto de Inventarios IUD, sobre la carpeta principal se crea el archivo gestor de dependencias (package.json) utilizando el comando npm en la terminal, lo que permitirá el uso y gestión de librerías que se requieran.
3. Dentro del archivo ***package.json*** se instalan las siguientes librerías: ***express*** que sirve para crear y gestionar el servidor web con el que funcionará la aplicación, ***cors*** que nos sirve para conectarnos vía internet a nuestra aplicación desde cualquier punto o dispositivo, ***dotenv***, librería o dependencia que permite crear y gestionar variables de entorno y ***Mongoose*** para gestionar la creación de la base de datos y también generar los distintos modelos que requiere nuestro proyecto.
4. En MongoDB se crea usuario, se establece que el acceso a la red sea público por ser un ejercicio académico, nos conectamos al cluster de NodeJs versión 2.2.12 o superior por ser esta la que menos conflictos genera.
5. En Visual Studio Code sobre la raíz del proyecto creamos carpeta llamada ***Db*** y dentro de esta creamos el archivo ***db.connectionmongo.js*** y sobre este archivo realizamos la importación de mongoose mediante el método .connect el cual recibe los parámetros para establecer la coneccion con la base de datos.
6. En la raíz del proyecto creamos el archivo ***index.js*** como el archivo base, el cual inicializará nuestra aplicación web e importamos la función get connection que permite conectarnos a nuestra base de datos.
7. Sobre la raíz del proyecto creamos una nueva carpeta llamada ***models*** y dentro de esta **5** archivos **.js** (***usuario, marca, tipo equipo, estadoequipo e inventario***), cada uno de ellos tendrá establecido los atributos con sus respectivas características de los datos que se utilizaran en determinado modelo.
8. En la raíz del proyecto creamos la carpeta ***router*** que contendrá las rutas asociadas a nuestros distintos servicios rest que podrá tener nuestro servidor de backend.

***LINK -Repositorio en el Github***

El trabajo está comprimido en un archivo llamado “inventarios”

https://github.com/Athea44/IngenieriaWeb2

***CONCLUSIONES***

El desarrollo de esta actividad nos permite concluir que con Node Js se puede generar páginas con contenido dinámico, crear, abrir, leer, escribir, eliminar archivos en el servidor, es fácil y sencillo de aprender e implementar, además es considerado el mejor framework para el desarrollo de API REST y permite la facilidad de integración con bases de datos No SQL.

Además comprendimos que REST es cualquier interfaz entre sistemas que use HTTP para obtener datos o generar operaciones sobre esos datos en todos los formatos posibles; su uso tiene varias ventajas como la separación entre el cliente y el servidor lo que hace más escalable, más visible y fiable el sistema, adicionalmente La API REST siempre es independiente del tipo de plataformas o lenguajes siempre y cuando las respuestas a las peticiones se hagan siempre en el lenguaje de intercambio de información usado, normalmente XML o JSON.

La base de datos MongoDB es una de las mejores aplicaciones en los microservicios, además de que es fácil de usar tiene diferentes integraciones con los lenguajes de programación que existen en la actualidad.

**BIBLIOGRAFÍA**

Haro, E., Guarda, T., Peñaherrera, A. O. Z., & Quiña, G. N. (2019). Desarrollo backend para aplicaciones web, servicios web restful: Node. js vs spring boot. *Revista Ibérica de Sistemas e Tecnologias de Informação*, (E17), 309-321.