

# JavaScript Basic

### Guía #6: REST Server & MySQL

Esta guía hace parte del curso mencionado en el título y les permitirá estudiar con más detenimiento los conceptos de clase, además de servir como instrumento de la misma. La guía contará con código base, el desarrollo conceptual de los temas y enlaces externos para el estudio independiente.

## 1. Iniciar con web server

Ya anteriormente hemos configurado un web server y partiremos de la misma base:

const express = require('express');  
const path = require('path');  
const app = express();  
const port = 8888;  
  
  
app.get('/', function (req, res) {  
 res.json('REST Index');  
});  
  
app.listen(port, function () {  
 console.log(`Server is running at http://localhost:${port}`);  
});

Ahora las respuestas siempre estarán dadas en JSON.

## 2. Rutas

Centralizaremos todos los controladores en un folder de rutas que tendrá como centro de registro de rutas al archivo routes/index.js:

const express = require('express');  
const app = express();  
  
app.use( require('./user));  
  
module.exports = app;

en este registraremos las rutas de usuario que estarán en routes/user.js

const express = require('express');  
const app = express();  
  
//Traer todos los usuarios  
app.get('/user', (req, res) => {  
 res.json('get Usuario')  
});  
  
//Traer al usuario cuya id sea :id  
app.get('/user:/id', (req, res) => {  
 res.json('get Usuario')  
});  
  
app.post('/user', (req, res) => {  
 res.json('post Usuario')  
});  
  
//Actualizar el usuario :id  
app.put('/user/:id', (req, res) => {  
 res.json('put Usuario')  
});  
  
//Borrar el usuario :id  
app.delete('/user/:id', (req, res) => {  
 res.json('delete Usuario')  
});  
  
module.exports = app;

## 3. Body en peticiones Post y Put

Para poder parsear los cuerpos de las peticiones usaremos [npm bodyparser](https://www.npmjs.com/package/body-parser), así que lo adicionaremos a nuestras dependencias así:

npm i body-parser --save

Y ahora lo instalaremos en nuestro servidor:

const bodyParser = require('body-parser')

además adicionaremos la creación de instancias de parseo para formularios y para JSON:

// parse application/x-www-form-urlencoded  
app.use(bodyParser.urlencoded({ extended: false }))  
// parse application/json  
app.use(bodyParser.json());

Estos dos trabajan como Middlewares que procesarán las peticiones antes de que nosotros trabajemos con ellas, de tal manera que ahora podremos ir a routes/user.js y modificar la respuesta del método post para procesar el cuerpo:

app.post('/user', (req, res) => {  
 let body = req.body;  
 res.json({user:body})  
});

Ahora podremos probar dicha petición desde Postman.

## 4. Códigos de estado

Para toda respuesta de un servicio RESTful, el código de respuesta enviado en la cabecera debe coincidir adecuadamente con la operación realizada o el resultado obtenido.

Ahora podremos editar nuestras respuestas como la siguiente respuesta correcta a un post:

res.status(201).json({  
 ok:false,  
 err  
});

## 5. MySQL

Para poder conectarnos con esta base de datos será necesario instalar el paquete del mismo así:

npm i mysql --save  
npm i mysqljs/mysql --save

Ahora debemos centralizar la configuración y la conexión como tal, para esto crearemos mysql.conf.js:

const mysql = require('mysql');  
exports.connection = mysql.createConnection({  
 host : 'localhost',  
 user : 'root',  
 password : '',  
 database : 'test'  
});

Y exportaremos la conexión para poder usarla en las rutas:

routes/user.js

const connection = require("../mysql.conf").connection;  
  
//Traer todos los usuarios  
app.get('/user', (req, res) => {  
 connection.connect();  
 connection.query('SELECT \* FROM users', (error, results, fields) => {  
 if (error) {throw error};  
 connection.end();  
 res.json(results);  
 });  
});  
  
//Traer al usuario cuya id sea :id  
app.get('/user/:id', (req, res) => {  
 let id = req.params.id;  
 connection.connect();  
 connection.query(`SELECT \* FROM users WHERE id = ${id}`, (error, results, fields) => {  
 if (error) {throw error};  
 connection.end();  
 res.json(results[0]);  
 });  
});