Documentación

API REST

OpenEvents

Artur Alcoverro

Desarrollo de Servicios en Línea

GTAS 2020-21

Índice

Introducción

Esta práctica se basa en la creación de una API REST para la red social OpenEvents, la cual forma parte de un proyecto transversal junto a las asignaturas de Desarrollo de aplicaciones móviles y Desarrollo en entornos web.

Para crear esta API usaremos Node.js junto con el framework Express i otras librerías que obtendremos a través del gestor de paquetes npm.

La API debe servir a los distintos clientes todos los datos necesarios sobre los usuarios, eventos, amistades y mensajes. Todos estos datos estarán en una base de datos MySQL, la cual deberemos diseñar y conectar con el servicio.

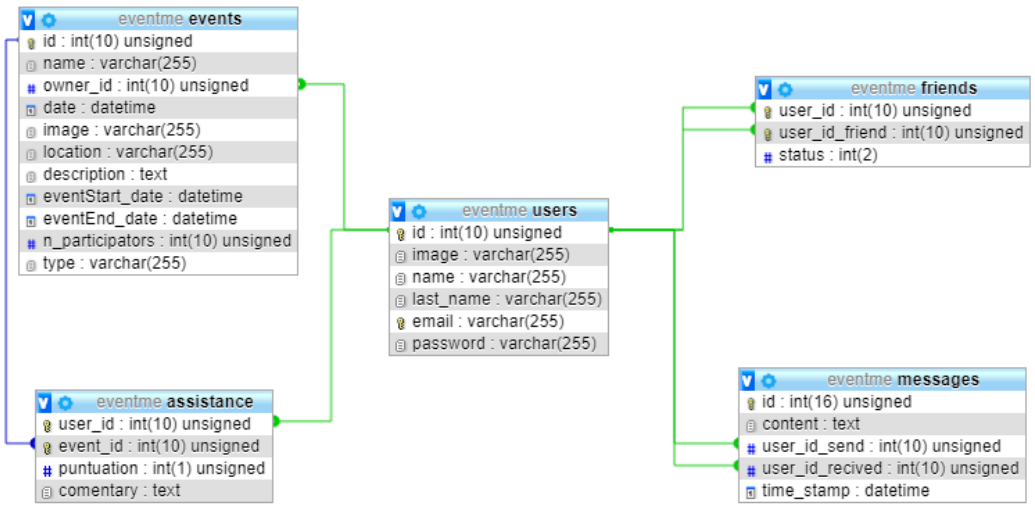
En la mayoría de los casos se accederá a esta información a través de un JSON Web Token, al cual solo se podrá obtener una vez registrado y autenticado.

Para realizar esta práctica se nos ha facilitado los distintos campos que tendrá cada tabla de la base de datos y la lista de endpoints o funcionalidades que debe tener la API.

Persistencia de datos

Antes de empezar a programar la API debemos crear la base de datos. Para hacer-lo he cogido el diseño y las representaciones que decidimos en clase y partir de estas he creado una base de datos MySQL a través del phpmyadmin que nos proporciona el paquete XAMPP.

Este es el resultado:



Para conectar la API a la base de datos he utilizado el módulo npm mysql2, a través del cual haré todas lanzaré todas las llamadas.

Estructura del proyecto

Una vez tenía la base de datos preparada ya podía empezar con la implementación de la API.

Para hacer-lo es muy importante que el código esté organizado siguiendo una estructura clara y coherente. De esta manera hacemos que el código sea mucho más fácil de mantener y/o escalar.

Carpeta raíz

En esta carpeta, aparte de las demás carpetas, es donde encontramos el fichero index.js, el cual se encarga de iniciar los routers, cargar las variables de entorno e iniciar el servicio.

Para las variables de entorno he utilizado el módulo npm.

En esta carpeta también encontramos el módulo que se encarga de autenticar el JSON Web Token y el módulo que guarda las imágenes subidas al servidor a través del módulo npm express-fileupload.

Carpeta /routes

En esta carpeta encontramos la declaración de todos los routers, los cuales se encargan de manejar las diferentes rutas y conectar-las con los controladores.

En estos ficheros es donde definimos los diferentes verbos HTTP para cada ruta y donde separamos las rutas que requieren de una autentificación de las que no.

También es aquí donde cargo las distintas validaciones de datos a través de middlewares.

Carpeta /controllers

En esta carpeta encontramos a los controladores de cada router.

Es en estos controladores donde están implementados todos los endpoints y funcionalidades de la API, y por lo tanto donde se hacen las llamadas a la base de datos a través del módulo npm mysql2.

Estos métodos son los encargados de enviar la información final al cliente y por lo tanto también controlan algún caso de error.

Carpeta /database

En esta carpeta es donde se encuentra el módulo que nos permite hacer llamadas a la base de datos, el cual es llamado desde los distintos controladores. Aquí también he puesto 2 ficheros .sql, uno que sirve para inicializar la base de datos y otro que además de eso carga datos mock.

Carpeta /validation

En esta carpeta se encuentran las validaciones de datos de cada router, las cuales se han realizado con el módulo npm Joi.

En estas validaciones compruebo los bodys, las querys y los parámetros y no solo me aseguro de que estén todos los campos obligatorios, también compruebo el tipo de dato y su estructura si es necesario.

Carpeta /uploads

En esta carpeta es donde se guardan todas las imágenes que se suben al servidor.

Módulos y herramientas de desarrollo utilizados

El proyecto está hecho con Node.js y los módulos que he instalado a través de npm son:

- **Bcrypt.** Para la encriptación y comprobación de contraseñas.
- **Dotenv.** Para tener variables de entorno.
- **Express.** Como framework para Node.js.
- **express- fileupload.** Para controlar las peticiones multi-part y así permitir la subida de imágenes.
- **Joi.** Para validar los datos de las peticiones
- **Jsonwebtoken.** Para la autenticación de los usuarios.
- **mysql2.** Para conectar con la base de datos.
- **Uuid.** Para generar IDs únicas. Lo he usado únicamente para nombrar las imágenes que se suben al servidor.
- **Nodemon.** Dependencia de desarrollo que vuelve a ejecutar el programa con cada modificación del código.

Como de editor de texto he utilizado Visual Studio Code y como programa de control de versiones Git.

Conclusión

Creo que este proyecto ha sido muy útil para entender el potencial que tienen las API REST, y esto en parte es gracias a que este proyecto es transversal con otras dos asignaturas.

En cada una de estas asignaturas creamos un cliente distinto, pero ambos se conectan a esta misma API y comparten datos sin entrar en conflicto en ningún momento.

Otro valor importante que tiene esta práctica es el contacto con todo el ecosistema de Node.js y la experiencia que nos llevamos de JavaScript.

Costes temporales

Para realizar este proyecto he tardado de horas, las cuales se han repartido de la siguiente forma: