Práctica 1 Implementación

Cambios en el diseño y especificación del API

Luis Vidal Rico

➢ Cambios introducidos respecto a la fase de diseño

A lo largo de la implementación del API se han modificado algunos aspectos de este, que se detallan a continuación:

- Se ha cambiado el concepto de 'UpVote' y 'DownVote'. Ahora esta entidad se ve recogida en una única tabla llamada 'Vote', que tendrá un atributo 'valor' que indicará si se trata de un voto positivo o negativo.
- Se ha añadido la ruta POST **/api/comunidad/usuario/:name**, que se encarga de vincular una comunidad a un usuario. Esta ruta se corresponde con la acción de 'seguir' una comunidad.
- Se ha añadido la ruta PATCH **/api/comunidad/:name**, que se encarga de modificar los datos de una comunidad.
- Se han cambiado el nombre de las rutas de voto a posts y comentarios. Estas antes acababan con **/vote** y ahora acaban con **/voto**. Este cambio se ha introducido para mantener la coherencia con el idioma en las rutas.

➢ Implementación de requisitos adicionales

- **Persistencia del API:** Para la implementación de la persistencia se ha utilizado el gestor de base de datos PostgreSQL. El paquete node utilizado para realizar la conexión con la base de datos ha sido Sequelize.
- **Testing:** Los tests se han realizado utilizando Postman. Estos se encuentran en la carpeta *test* del directorio raíz del proyecto.

  Para ejecutarlos, se recomienda utilizar el servidor en modo test. Para esto habría que cambiar la variable **Test** del archivo *.env* a **true**, e indicar una base de datos donde se ejecutarán los tests.

  De esta forma, cada vez que se inicie el servidor se reiniciará la base de datos, y se podrán ejecutar los tests de forma correcta.

- **Documentación del API**: La documentación del API se encuentra ya compilada en el directorio raíz del proyecto, en formato **html**. Para realizarla se ha utilizado la sintaxis de **API Bluepoint**. El documento a partir del cual se genera el archivo html se encuentra en la carpeta *doc*.

  En caso de querer generar la documentación otra vez, habría que instalar **Aglio** con el comando **npm i -g Aglio** y ejecutar el comando **npm run doc**.

- **Archivos Binarios:** Se ha habilitado la subida de archivos binarios cuando se crea un usuario y cuando se crea un post. Para ello se ha utilizado la librería **express-fileupload**. Los archivos se guardan en las carpeta public.

- **Paginado**: Se ha implementado paginado en el API en los **GET** de Comunidad (en los posts que se reciben de esta) y Post (en los comentarios que se reciben de este). Para utilizar el paginado se ha usado un **offset**, que indica la 'página' en la que nos encontramos y un límite, que limita el número de instancias a devolver.

  Estos valores se pasan en la URL de la llamada, y si no se indican se utilizan valores por defecto.