28-4-2022

Documento de decisiones arquitectónicas dede_es1b

Lucía García Sopeña (uo271080)
Sergio Salgueiro Monforte (uo276843)
Héctor Lavandeira Fernández (uo277303)
Juan Domínguez Álvarez (uo277646)
Maria González Otero (uo270235)

ÍNDICE

1 INTRODUCCIÓN

En este documento se van a detallar todas las decisiones arquitectónicas que el grupo dede-es1b han ido decidiendo desde el comienzo de la asignatura.

Aviso: en este documento no se tienen en cuenta los requisitos obligatorios impuestos por el profesorado de la asignatura.

2 DECISIONES RESTAPI

En cuanto a la parte del backend se ha decidido utilizar lo siguiente:

- Como gestor de base de datos hemos decidido utilizar **mongodb** ya que tiene una buena conexión con JavaScript.

- Utilizamos el servicio **MongoDB Atlas** que es un servicio de Cloud Database que nos permite poder administrar y consultar la base de datos desde cualquier lugar del mundo de manera rápida y eficiente.

- Hemos utilizado la librería **dotenv** que nos permite configurar de manera concisa las variables de entorno y así poder ocultar varios datos del github que son de alta importancia como la url de mongodb.

- Utilizamos el **JSON web Token** para que nos ayude a propagar la identidad por toda la aplicación, de tal forma que una vez se haga login, no se pierda esa identidad hasta que se haga logout.

- En cuanto a la organización o estructura del código hemos decidido crear un paquete para cada tipo de datos (usuario,productos, …) en donde dentro de cada uno se tendrán tres clases, un modelo (define los esquemas con los atributos del tipo de datos), un controlador(define las consultas a la base de datos) y un router(define las rutas donde se van a utilizar las consultas get y post).

3 DECISIONES WEBAPP

En cuanto a la parte del frontend se ha decidido utilizar lo siguiente:

- Utilizamos la librería **mui** ya que proporciona una librería bastante robusta de componentes básicos para la aplicación y hace que la programación de la aplicación react sea bastante más rápida.

- Para solid utilizamos las dependencias **solid-ui-react** y **solid-ui-client** ya que son las más sencillas de utilizar para el uso que queremos hacer de solid.

- En cuanto a estructura de paquetes tenemos la conexión con el back en la clase api.ts y el resto de las clases (tsx y css) de diseño las tenemos en el paquete pages ordenadas en sus respectivos paquetes

4 OTRAS DECISIONES

4.1 DECISIONES DEPLOY

En cuanto el deploy decidimos utilizar lo siguiente:

- Decidimos hacerlo con heroku porque consideramos que es más fácil de usar.

- Creamos dos aplicaciones una con restapi y otra con webapp ambas son realizadas a partir de las imágenes creadas por sus DockerFile correspondientes.

- En cuanto al despliegue de la aplicación webapp está hecho con integración continua con github, es decir, cualquier actualización del main se actualizará en heroku también.

- En cuanto al despliegue de la aplicación restapi decidimos no hacer lo mismo, ya que al vincularla con github saldría en el apartado de enviroments y no queremos que se sepa su url por protección de datos ya que con ella se podría hacer cualquier cambio en la base de datos. Para solucionar este problema creamos una imagen en local y la subimos directamente a heroku.

4.2 DECISIONES IMÁGENES

En cuanto a las imágenes decidimos utilizar cloudinary que es un repositorio que nos permite subir nuestras imágenes y gracias a ello no necesitaríamos tener las imágenes en local.