Desarrollo de aplicaciones web
Curso 2023/24
Documentación Proyecto DAWEB

Autores:
Carlos Cruzado Esteban - 49308284X - 1.2
Eduardo Gallego Nicolás - 48844607J - 1.2

Índice:

1. Introducción

Para esta práctica hemos tenido que desarrollar un proyecto front-end para nuestro back-end de la asignatura de ARSO. Este proyecto requiere una comprensión profunda de la lógica del servidor, las API y los datos manejados por el back-end, así como la habilidad para transformar estos elementos en una experiencia de usuario intuitiva y eficiente. La meta principal es asegurar que la comunicación entre el front-end y el back-end sea fluida y eficaz, permitiendo a los usuarios interactuar con el sistema de manera coherente y sin interrupciones.

El primer paso en este proceso fue familiarizarnos con la estructura y funcionalidad del back-end. Esto incluyó revisar las peticiones en postman, analizar el código existente e incluso realizar algunos cambios respecto al codigo entregado en la asignatura de ARSO. Un aspecto crucial fue identificar los puntos de integración, donde el front-end necesitaba conectarse con los endpoints del back-end para obtener y enviar datos, y poder tener las funcionalidades que requería el proyecto.

La siguiente fase fue diseñar la interfaz de usuario (UI). Este diseño tuvo que ser no solo estéticamente agradable, sino también funcional y accesible. Se tuvieron en cuenta principios de usabilidad y diseño responsivo para asegurar que la aplicación fuera accesible en una variedad de dispositivos y tamaños de pantalla. Herramientas y tecnologías como HTML, CSS, JavaScript y frameworks modernos como React fueron fundamentales en esta etapa para crear componentes reutilizables y eficientes.

La implementación del front-end también implicó manejar la lógica de negocios en el cliente, validar las entradas de usuario, y asegurar la seguridad de los datos transmitidos. Se implementaron mecanismos de autenticación y autorización para proteger las transacciones y mantener la integridad de los datos. Además, se realizaron pruebas exhaustivas para identificar y corregir errores, optimizando el rendimiento y asegurando una experiencia de usuario sin inconvenientes.

En resumen, el desarrollo de un front-end para un back-end existente es un proceso integral que combina análisis, diseño, implementación y pruebas. Este esfuerzo conjunto no solo mejora la interacción del usuario con el sistema, sino que también optimiza la funcionalidad y seguridad de la aplicación web en su conjunto.

2. Descripción de la arquitectura usada

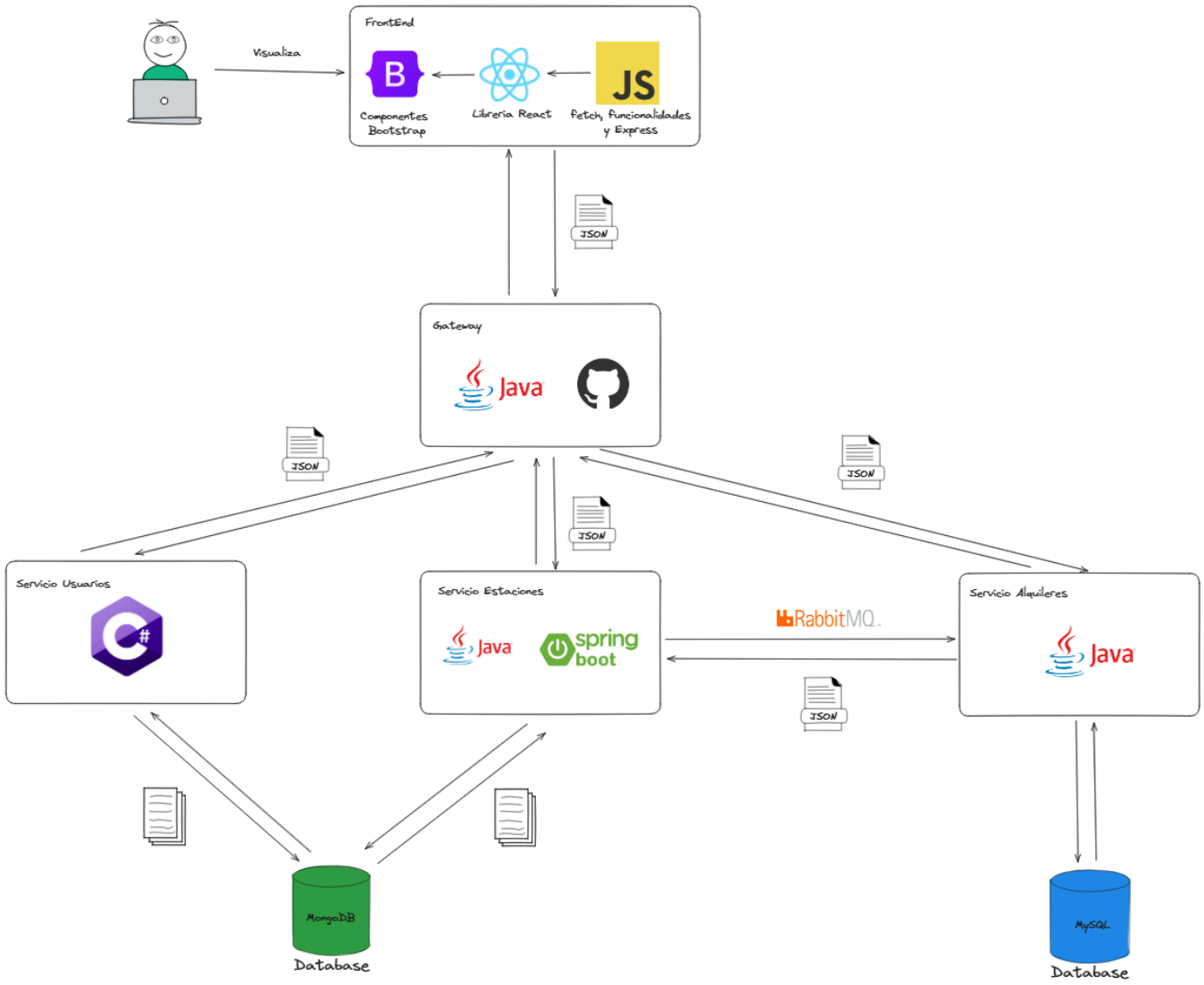

Figura 1: Diagrama de la arquitectura usada

Para realizar el front-end hemos usado el back-end siguiente. Tenemos un gateway que redirecciona las peticiones a los otros servicios, y además realiza el login por defecto y el oauth2 a github. Para registrarse el gateway redirige la petición al servicio Usuarios para ejecutar la función de registro.

El servicio Usuarios se ha creado con el lenguaje c# y ofrece la funcionalidad de registrar, verificar las credenciales de un login o de un login oauth2. Usa una base de datos no relacional, MongoDB, para guardar los usuarios.

El servicio Estaciones ha sido creado con SpringBoot y java. SpringBoot nos ha ayudado con facilidad a crear la ApiRest. Este servicio nos ofrece la funcionalidad de consultar las estaciones de forma paginada y las bicicletas de esta estación, también se puede modificar, eliminar y crear una Estación, y agregar una bicicleta en una estación. Por último, mencionar que usa una base de datos no relacional de MongoDB, para guardar las estaciones y las bicicletas.

El servicio Alquileres ha sido creado con java y la ApiRest se ha creado de forma manual. Este servicio nos ofrece consultar el historial de un usuario, reservar una bicicleta, alquilar una bicicleta y por último, devolver una bicicleta en una estación. Este servicio hace uso de una base de datos relacional de MySQL.

El servicio Estación y Alquileres se comunican con retrofit y mediante mensajes asíncronos de RabbitMQ.

El front-end se comunica con el back-end con peticiones fetch con archivos JSON. Las peticiones se hacen todas al gateway. Además para el registro se ha hecho uso de un servidor express que se comunica con el gateway exclusivamente para el registro.

Por último aclarar que se ha hecho uso del back-end de la asignatura de ARSO.

2. Descripción de la funcionalidad del front-end

La aplicación Citybike es una herramienta de movilidad por las ciudades de España, se pueden reservar y alquilar bicicletas de las estaciones para moverse de un sitio a otro.

Cuando entramos a la aplicación de primeras nos encontramos con la página de presentación la cual su única funcionalidad son los botones de iniciar sesión y registro:

- Iniciar Sesión: el botón te lleva a una página que tiene un formulario para introducir los datos de email y contraseña, si son correctos, el backend hace su trabajo y el frontend completa la ejecución llevando al usuario a la página principal de la aplicación, también hay un botón para iniciar sesión con GitHub, si es que existe un usuario en la BD con estas características y dos botones más, uno para ir al registro y otro para volver atrás.

- Registro: este botón también te lleva a una página formulario en la que hay que rellenar los datos como nombre, dni, fecha de nacimiento, teléfono, email y la contraseña. Si todo va bien y no hay problema como por ejemplo, un email ya registrado, al darle al botón registrar te lleva a la página de presentación para que puedas iniciar sesión, también hay un botón para volver atrás.

Hay dos tipos de usuarios al iniciar sesión, el Cliente y el Gestor, el segundo tiene funcionalidad extra y la página se visualiza diferente para que el usuario sepa en todo momento que él ha iniciado cuenta con un Gestor, empecemos por explicar la funcionalidad del usuario Cliente.

Cuando inicia sesión se abre la página principal, la cual tiene 3 pestañas (Estaciones, Reservas y Alquileres) y un botón para cerrar sesión en el Header, el Footer no cambia. Redirige directamente a la pestaña de Estaciones donde el cliente puede visualizar todas las estaciones del sistema y filtrarlas por un buscador, en el que puede filtrar por nombre, código postal y número de puestos libres, la parte del nombre y código postal usa expresiones regulares y puede realizar búsquedas como por ejemplo “Murcia30007” ó “Murcia 30007” retornando resultados en los que el nombre tenga algo de la palabra “Murcia” y tenga codigo postal 30007 la estación. Cada estación tiene un botón para ver una tabla con todas las bicis estacionadas que hay en ella, pudiendo reservarlas o alquilarlas en caso de estar disponibles.

En la pestaña de Reservas aparecerán la reserva activa y caducadas que tenga el usuario, las reservas activas tendrán un botón para alquilar, cuando se alquilan se elimina la reserva y se crea un alquiler activo en la pestaña alquileres.

En la pestaña de Alquileres aparecerán el alquiler activo y terminados que tenga el usuario, los alquileres activos tendrán un botón para aparcar la bicicleta, la cual se estacionará en la estación seleccionada.

El usuario Gestor sabe en todo momento que él es un gestor porque el Header cambia de color, el gestor tiene funcionalidad adicional al cliente, así que solo voy a explicar la nueva funcionalidad.

En la pestaña de Estaciones aparece un nuevo botón para añadir una nueva estación abriendo un formulario que pide nombre, número de puestos, código postal, latitud y longitud. A parte de este botón, cada estación tiene dos botones extra, uno para modificar y otro para eliminar la estación, además al ver las bicis de una estación también se pueden dar de baja.

Hay una pestaña Bicis que solo pueden ver los gestores para poder visualizar todas las bicis de golpe con todos sus datos.

3. Manual de Usuario

Antes de desplegar el proyecto hay que tener instalado docker desktop en nuestra maquina.

Para desplegar el proyecto hay que seguir los siguientes pasos:

1. Abrir una terminal en la carpeta **ci** del back-end.

2. Ejecutamos **docker-compose build** (En caso de tener otros contenedores ya habrá que eliminar con **docker-compose down**) para construir los contenedores.

3. Seguidamente ejecutamos **docker-compose up -d** para ejecutar los contenedores.

4. Una vez hemos ejecutado el back-end de ARSO, procedemos a ejecutar el front-end. Desde la **carpeta src** del front-end deberemos ejecutar en una terminal nueva **npm -i**, para instalar todas las dependencias.

5. En la terminal anterior después ejecutamos **node server.js** para iniciar el servidor Express.

6. Y por último, abrimos una terminal nueva y ejecutamos desde la **carpeta src** el front-end con **npm start**.

Ya con los pasos a seguir podremos desplegar, ejecutar y probar el proyecto creado para esta asignatura.

4. Conclusiones

En conclusión, el proyecto de desarrollo front-end realizado con React y complementado con Express y componentes de Bootstrap ha sido una experiencia enriquecedora y significativa en el ámbito de la programación y el desarrollo web. Este proyecto no solo ha permitido aplicar conceptos teóricos aprendidos durante el curso como es el caso de la asignatura de INTU, sino también adquirir habilidades prácticas esenciales para la creación de aplicaciones web modernas, funcionales y responsivas.

La combinación de React para el front-end y Express ha demostrado ser altamente efectiva para construir aplicaciones web robustas. React ha proporcionado una manera eficiente de gestionar la interfaz de usuario con componentes reutilizables y un manejo del estado optimizado. Por su parte, Express ha sido usado para la funcionalidad de registro la cual redirige la petición a nuestro back-end.

La interacción con el back-end desarrollado en la asignatura ARSO ha sido crucial para el éxito del proyecto. A través de los distintos servicios y gracias al gateway, se logró una comunicación eficiente entre el cliente y el servidor, permitiendo la obtención y manipulación de datos de manera dinámica. Esta integración ha resaltado la importancia de una arquitectura bien definida y de la interoperabilidad entre distintos módulos de software.

La implementación del proyecto ha contribuido significativamente al desarrollo de habilidades técnicas en varias áreas: desarrollo con React, incluyendo una comprensión profunda de componentes, estado, props, y manejo de eventos; configuración con Express para gestionar rutas y solicitudes en el servidor; y manejo de solicitudes asíncronas y procesamiento de respuestas para actualizar dinámicamente la interfaz de usuario.

Además, la realización de este proyecto ha puesto en práctica habilidades de gestión de proyectos, incluyendo la planificación, división de tareas y colaboración efectiva dentro del equipo. El uso de control de versiones con Git ha sido indispensable para mantener un flujo de trabajo organizado y colaborativo.

Este proyecto ha subrayado la importancia de la sinergia entre el front-end y el back-end en el desarrollo de aplicaciones full stack. La experiencia adquirida no solo se limita a conocimientos técnicos, sino que también abarca habilidades de resolución de problemas, adaptabilidad y trabajo en equipo.

A medida que avanzamos en nuestra formación académica y profesional, este proyecto servirá como una base sólida sobre la cual construir y enfrentar nuevos desafíos en el mundo del desarrollo web y más allá. La combinación de React, Express y el back-end de ARSO ha demostrado ser una elección acertada, proporcionando una plataforma versátil y potente para el desarrollo de aplicaciones web.