SDI – Sistemas Distribuidos e Internet

ENUNCIADO PRÁCTICA 2 – NodeJS - SW

INFORME

Grupo 2021-1005

Nombre:	Samuel
Apellidos:	Moreno Vincent
Email:	UO266321@uniovi.es
Cód. ID GIT	2021-1005
% Participación	100%
Repositorio GitHub	https://github.com/samuelmorenov/sdi-entrega2-2021-1005

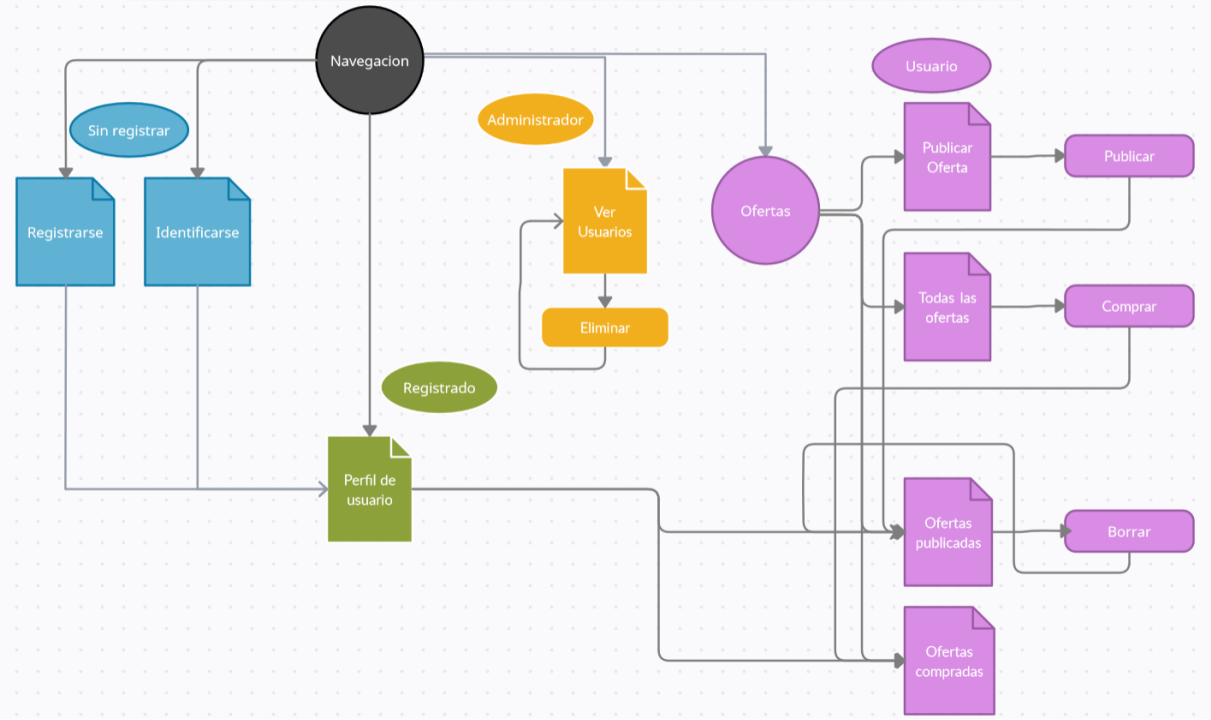
Índice

Introducción

Para la realización de este proyecto se ha utilizado como base el proyecto de Node Js realizado en el desarrollo de las clases prácticas de la asignatura, siguiendo los guiones de las sesiones de la 7 a la 10. Así pues, se ha mantenido toda la apariencia visual que tenía dichas prácticas. Además, se ha utilizado parte del proyecto de My Social Network entregado en la convocatoria adelantada en enero, utilizándolo únicamente como guía en caso de alguna duda para una mayor rapidez a la hora de realizarlo.

Mapa de navegación

Parte de aplicación web



Leyenda:

Página: Muestra cada una de las paginas existentes en la aplicación.

Permisos: Muestra los permisos requeridos para esa zona de la aplicación (código de colores explicados a continuación)

Acción: Posibles acciones a realizar en cada página.

Menú de navegación: Menú o submenú de navegación que se encuentra en la parte superior de la aplicación.

Flecha rellena: Indica un cambio de página accionado por el usuario.

Flecha vacía: Indica una redirección automática de la aplicación.

Código de colores:
Negro: Menú de navegación superior visible desde cualquier parte de la aplicación.
Azul: Opciones públicas para el registro e inicio de sesión del usuario.
Verde: Página común a todos los usuarios autentificados en la aplicación.
Amarillo: Página y acciones exclusivas del administrador.
Morado: Página y acciones exclusivas de usuarios.

Explicación del mapa

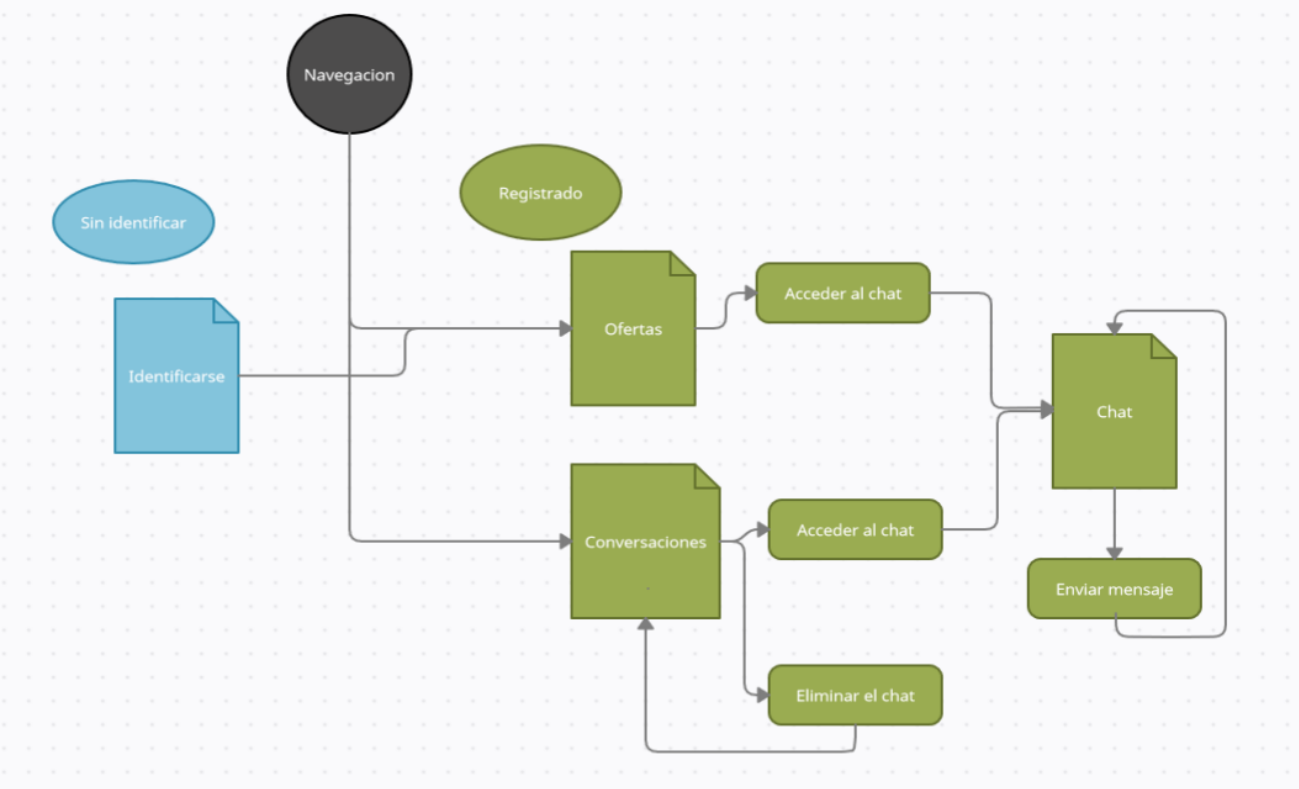
Al acceder a la aplicación el usuario no estará autentificado en la aplicación, por lo que desde el menú "Navegación" solo podrá acceder a las páginas de registro e identificación. Por defecto la aplicación le redirigirá a la pantalla de identificación, pero si no tiene un registro previo puede acceder a la pantalla de registro desde el menú. Una vez autentificado (ya sea como registrado o identificado) la aplicación redirige automáticamente a la página de perfil de usuario, que es común para todos los tipos de usuario.

Si el usuario es administrador en la pantalla de perfil de usuario únicamente verá un mensaje de bienvenida, y en la barra de navegación tendrá la opción de ir a la página de usuarios registrados. En esta página el administrado puede marcar los usuarios de la lista que quiera y eliminarlos del sistema, cuando haga esto verá como los usuarios desaparecen de la lista.

Si el usuario es un usuario estándar en la pantalla de perfil de usuario le aparecerá su nombre, la cantidad de dinero que tiene, y dos enlaces para ver sus ofertas publicadas y compradas. Además, desde el menú de navegación podrá acceder a cuatro opciones:

- Publicar oferta: En esta pantalla el usuario podrá rellenar el formulario para la creación de una nueva oferta. Una vez relleno, al dar a publicar, se le redirigirá a la pantalla de ofertas publicadas automáticamente.
- Todas las ofertas: En esta pantalla por defecto se realiza una búsqueda de todas las ofertas existentes en el sistema, pudiendo filtrarlas con la barra de búsquedas. En el listado de las ofertas se muestra el título, la descripción, la fecha de publicación y las opciones de compra. Estas opciones estarán bloqueadas si la oferta ya ha sido comprada o si la oferta esta publicada por el usuario que está navegando, y estará habilitada la opción de compra en caso contrario. Si el usuario compra la oferta se redirigirá automáticamente a la pantalla de ofertas compradas.
- Ofertas publicadas: En esta pantalla el usuario podrá ver todas las ofertas que ha publicado, y podrá borrar cualquiera de ellas. Cuando borre alguna podrá ver la lista de nuevo en la que no aparecerá esa oferta.
- Ofertas compradas: En esta pantalla el usuario podrá ver una lista de todas las ofertas que ha comprado.

Parte de Servicios Web



Leyenda:

Página: Muestra cada una de las paginas existentes en la aplicación.

Permisos: Muestra los permisos requeridos para esa zona de la aplicación (código de colores explicados a continuación)

Acción: Posibles acciones a realizar en cada página.

Menú de navegación: Menú o submenú de navegación que se encuentra en la parte superior de la aplicación.

Flecha rellena: Indica un cambio de página accionado por el usuario.

Flecha vacía: Indica una redirección automática de la aplicación.

Código de colores:
Negro: Menú de navegación superior visible desde cualquier parte de la aplicación.
Azul: Opciones públicas para el registro e inicio de sesión del usuario.
Verde: Página común a todos los usuarios autentificados en la aplicación.

Explicación del mapa

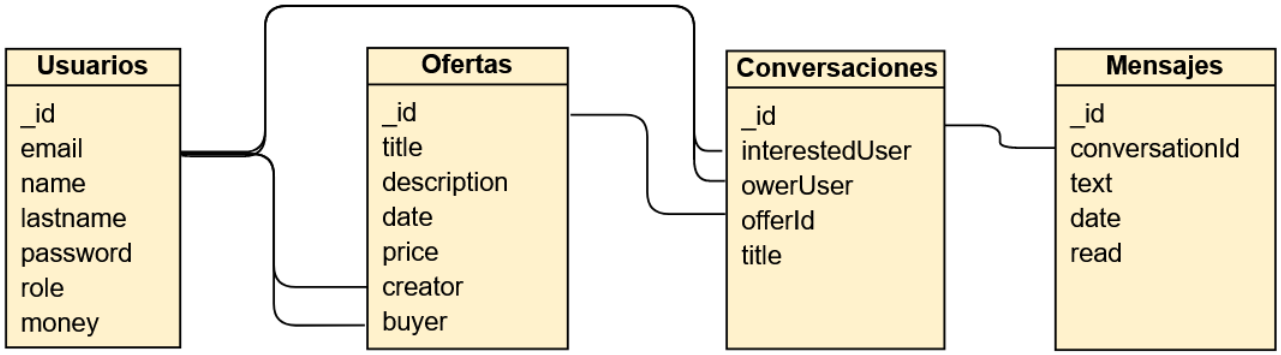
Al acceder a la aplicación el usuario no estará autentificado en la aplicación, por lo que por defecto la aplicación le redirigirá a la pantalla de identificación Una vez autentificado la aplicación redirige automáticamente a la página de ofertas. En la barra de navegación aparecerá siempre la opción de oferta y conversaciones, pero si el usuario no esta registrado le redirigirán directamente a la pantalla de identificación.

En la pantalla de ofertas aparecerá un listado con todas las ofertas existentes en la aplicación a excepción de aquellas de las que el usuario es el creador. Desde aquí podrá acceder a la una pantalla de chat con cada una de las ofertas. Si la conversación no existía antes, se creará una conversación vacía al pulsar este botón, de manera que los dos usuarios implicados podrán enviar mensajes.

En la pantalla de conversaciones aparecerá un listado con todas las conversaciones del usuario. Para cada conversación aparecerá un botón de acceso al chat de esa conversación y un botón de borrado de esta.

Aspectos técnicos y de diseño relevantes

Modelo de dominio de la base de datos



El modelo de dominio es bastante sencillo, únicamente consta de Usuarios, Ofertas, Conversaciones y Mensajes.

Los usuarios tienen los atributos _id (identificador), email (correo electrónico, que no puede estar repetido), name (nombre del usuario), lastname (apellidos del usuario), password (contraseña del usuario que se guardara cifrada), role (si es administrador o usuario estándar) y money (dinero actual que tiene el usuario).

Las ofertas tienen los atributos _id (identificador), title (título de la oferta), description (breve descripción de la oferta), date (fecha de alta de la oferta), price (precio en euros de la oferta), creator (email del usuario que publica la oferta) y buyer (email del usuario que compra la oferta).

Las conversaciones tienen los atributos _id (identificador), interestedUser (email del usuario que está interesado en la oferta), ownerUser (email del usuario propietario de la oferta), offerId (_id de la oferta relacionada a la conversación) y title (título de la oferta relacionada a la conversación).

Los mensajes tienen los atributos _id (identificador), conversationId (_id de la conversacion a la que pertenece), text (texto del mensaje enviado por el usuario), date (fecha en la que se envio el mensaje) y read (booleano que indica si se ha leído el mensaje o no).

Decisiones de implementación

Para el acceso a la base de datos hay que escribir la contraseña dentro del código. Esto me pareció un gran problema de seguridad. Para solventarlo la contraseña se guarda dentro de un archivo pass.txt que queda fuera de repositorio. En una aplicación real se debería cifrar dicho archivo.

Se decidió que el administrador no tenga acceso a la parte de creación y visualización de las ofertas, únicamente a la gestión de los usuarios.

En el ejercicio W9 Usuario registrado: Buscar ofertas, se decidió añadir la opción de buscar por título y/o descripción para una mejor usabilidad de los usuarios. Además, la página por defecto muestra la lista completa de todas las ofertas, como si se hubiera realizado una búsqueda vacía, para que el usuario no se encuentre únicamente con el campo de búsqueda, en consecuencia, esta parte se llamará "Todas las ofertas" dentro de la aplicación.

En lugar de estar todos los casos de pruebas apilados en una única clase se han separado en 13 clases distintas correspondientes a cada uno de los ejercicios. Todos ellos se ejecutan de forma consecutiva desde la clase "SdiEntrega220211005ApplicationTests.java". Esto conlleva el inconveniente de no poder ejecutar una prueba de forma individual a priori, ya que no ejecutaría el método "BeforeClass". Para solucionar este problema, y además hacer la programación de las pruebas más fluida se ha implementado un patrón Singleton para el driver "WebDriver" de selenium, de manera que ahora todos los test llaman a la misma instancia del driver y se inicializa una única vez. Este patrón se puede ver en la clase DriverSingleton.java. Además, todas las clases de pruebas heredan de una clase BaseTests o BaseTestApi (dependiendo de si es la parte web o la parte de servicios web) que establecen el WebDriver, navega hasta la url de inicio antes de cada prueba y limpia las cookies al finalizarlas.

Se decidió crear una tabla en la base de datos conversación a parte de la tabla de mensajes para su mejor manejo a la hora de visualizar la lista de conversaciones y poderlas borrar.

Existe un enrutador llamado rTest.js que tiene un método get para /test/resetDB. Este metodo se ejecuta antes de cada prueba y sirve para reiniciar la base de datos.

Información necesaria para el despliegue y ejecución

Al tener una base de datos mongodb no es necesario arrancarla ya que está siempre activa.

Para arrancar tanto la parte de aplicación web como la de servicios web únicamente es necesario arrancar la aplicación de node app.js. Es importante destacar que, como se explicó en las decisiones de implementación, se necesita el archivo pass.txt en la misma dirección que el archivo app.js. Este archivo contendrá la contraseña. Por motivos de seguridad este archivo no se sube a GitHub, pero si estará presente en la entrega del campus virtual de la asignatura.

La url de acceso a la parte de la aplicación web es https://localhost:8081/ y la de servicios web es https://localhost:8081/cliente.html

Para ejecutar las pruebas se deberá abrir el proyecto correspondiente con STS y ejecutar el archivo SdiEntrega220211005ApplicationTests.java. Para que las pruebas funcionen debe estar arrancada la aplicación.

La aplicación está preparada para ejecutar en modo de desarrollo, ya que tienen la parte del enrutador rTest.js que se menciono en la parte de decisiones de implementación. Para ejecutar bien la aplicación en un entorno de producción este archivo debería ser borrado y modificada su referencia en la aplicación app.js.

Conclusión

Aunque no he podido realizar todas las partes opcionales del proyecto por falta de tiempo, he aprendido mucho acerca de Node Js realizando estas prácticas y me parece un complemento estupendo junto a Spring Boot. Además, la parte de servicios Web es realmente útil, habría querido hacer una parte así también en Spring Boot ya que es lo que estoy haciendo en las prácticas de empresa.