SDI – Sistemas Distribuidos e Internet

ENUNCIADO PRÁCTICA 2 – NodeJS & Servicios Web

INFORME
1920-508

Nombre1:	Aitor
Apellidos1:	Llanos-Irazola
Email1:	UO264476@uniovi.es
Cód. ID GIT	1920-508

Índice

Introducción

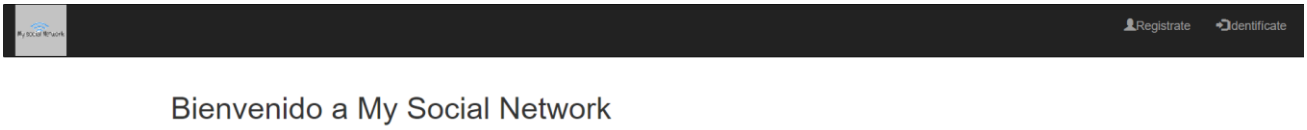
Este trabajo realizado para la asignatura de Sistemas Distribuidos e Internet consiste en una red social muy básica en la cual se puede agregar amigos mandándole peticiones de amistad previa y una red de mensajería muy básica a su vez también.

Es un trabajo para demostrar el conocimiento sobre el framework NodeJS para la realización de esta tarea.

Mapa de navegación

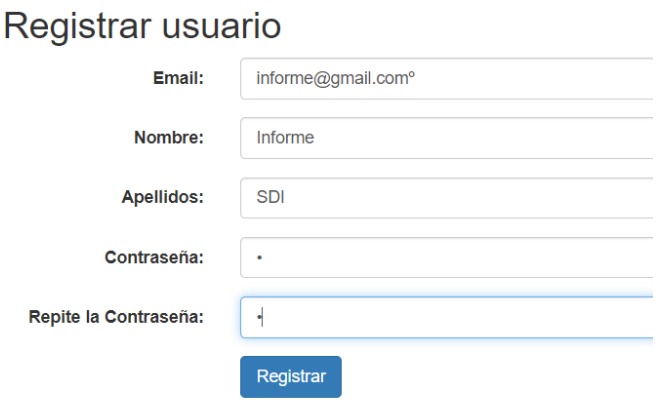
En este mapa de navegación se va a mostrar una ejecución de la aplicación y posteriormente se aclarará como se ha realizado los distintos apartados de la práctica.

Al ejecutar la aplicación nos dirigimos a la url https://localhost:8081 ya que está desplegada en local en el puerto 8081, y nos encontramos con la pantalla principal.



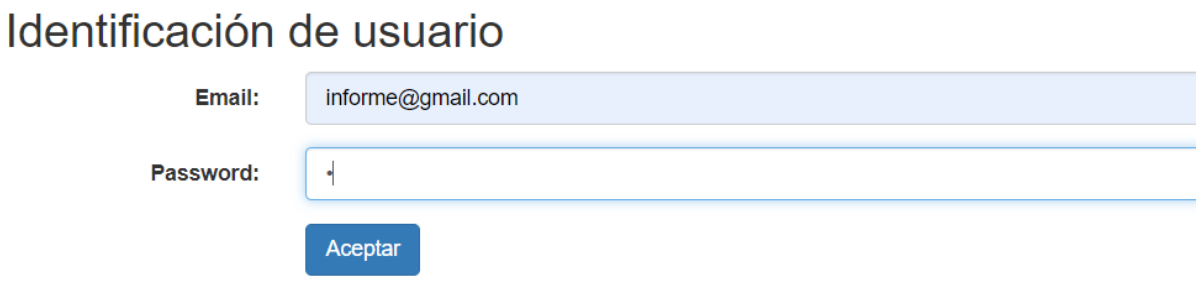
Bienvenido a My Social Network

Ahora de manera habitual procederíamos a registrarnos en la aplicación, para ello seleccionamos el botón superior derecho "Registrarse".

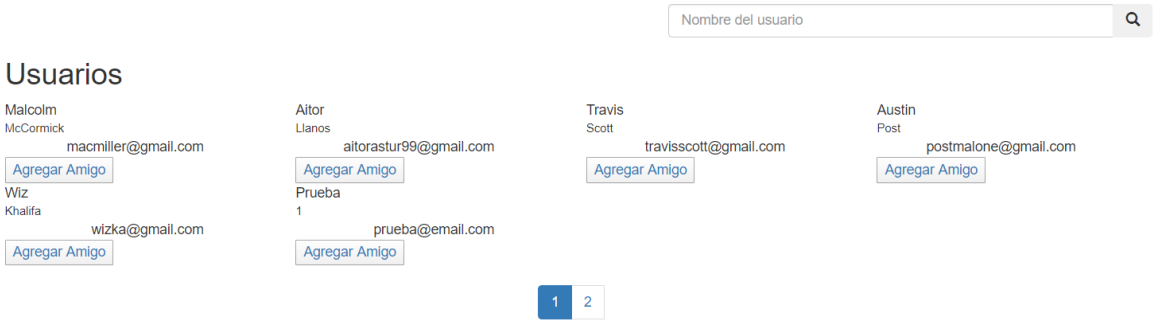


En este apartado introduciremos los datos pertinentes y si ocurriese algún inconveniente el usuario serio avisado.

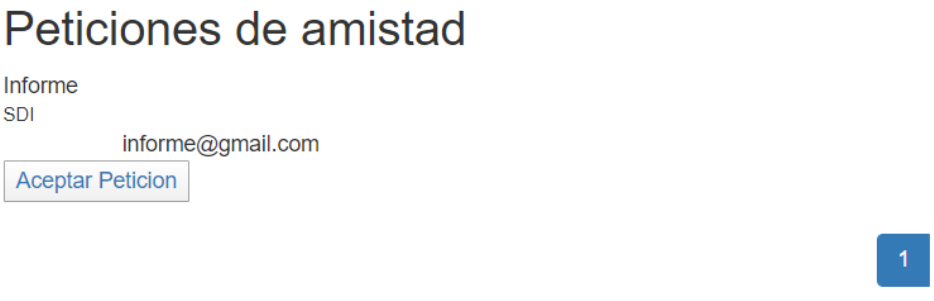
Posteriormente a registrarnos, nos identificaremos en el botón superior derecho "Identifícate" con la cuenta recientemente creada.



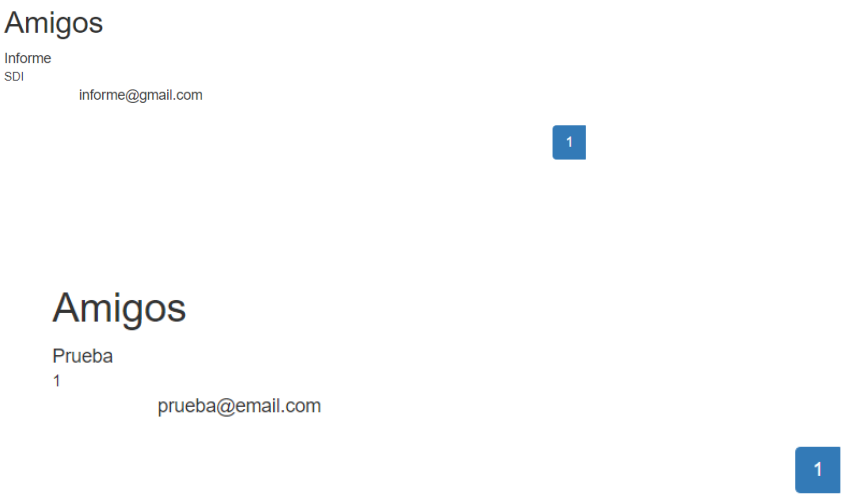
Al registrarnos correctamente, nos redirigirá a la vista donde se muestran todos los usuarios, excepto usted claramente, para poder enviarles una petición de amistad si lo deseamos.



En esta ventana se puede buscar al usuario que desees mandarle una petición ya sea buscándolo con la parte superior derecha, en donde introducirías su nombre, apellido o email o incluso ciertas partes de este esta aceptado, o moviéndote entre las páginas y buscándolo visualmente. Al encontrarlo podrás enviarle una solicitud de amistad clickando en el botón "Agregar Amigo", este no estará visible si ya has enviado una petición de amistad, te la ha enviado a ti o sois ya amigos, entonces al seleccionar el botón se enviará la petición de amistad, lo que nos lleva a la siguiente ventana.



En la ventana anterior se ha enviado una petición de amistad al usuario prueba@email.com y para visualizar esta petición debemos iniciar sesión con esa cuenta. Aquí veremos como se muestra la petición de amistad y tenemos la posibilidad de aceptarla, en este momento no se permite rechazarla por lo que si no se desea aceptarla siempre se puede dejar sin contestar la petición, en cuanto se acepte la petición ambos pasan a ser amigos y la petición de borra automáticamente.



En cuanto a la navegación por la api o servicios Rest se trata únicamente de enviar las peticiones como en cualquier otro api y al carecer de interfaz se explicarán los pasos de manera procedimental.

1. Se debe iniciar sesión con unos valores ya existentes en la aplicación, esta devolverá un token el cual se debe copiar en la cabecera de las siguientes peticiones.
2. En cualquiera de las siguientes peticiones se debe enviar el token en la cabecera de la petición y posteriormente dependiendo de las diferentes opciones otros valores en el cuerpo de la petición.
    - En cuanto al listado de amigos únicamente se debe incluir la cabecera como cualquier otro método de esta lista.
    - En cuanto a la creación de un mensaje además del token en el cuerpo de la petición de debe incluir en un fichero json el apartado "para" el receptor del mensaje, "texto" el contenido del mensaje a enviar.
    - En cuanto a la visualización de mensajes con el token es suficiente, se le mostraran cualquiera de los mensajes de los cuales sea emisor y/o receptor.
    - Por último, marcar un mensaje como leído únicamente necesita el identificador del mensaje el cual se le proporcionara en su visualización.

Cabe destacar que cualquiera de los métodos mencionados previamente no se podrá realizar si no se realiza la identificación correcta, para que la aplicación devuelva un token.

La ultima parte de esta aplicación consiste en un apartado realizado con JQuery, con similitudes a la primera parte de esta aplicación desarrollada toda en si en NodeJS.

Ahora se pasa a explicar cómo se ha realizado cada uno de los apartados como se pide en el documento de la práctica.

1. Parte 1: "Aplicación Web"

1.1 Registrarse como usuario

Para la realización de este apartado se ha empleado la misma técnica que en las prácticas de la asignatura

1.2 Iniciar Sesión

Para la realización de este apartado se ha empleado la misma técnica que en las prácticas de la asignatura

1.3 Fin de Sesión

Únicamente consta de un botón que se muestra únicamente si un usuario está en sesión y un método que deja el usuario en sesión como si no hubiese nadie y muestra el mensaje de desconexión.

1.4 Listar todos los usuarios de la aplicación

Este apartado muestra todos los usuarios de la aplicación menos el usuario en sesión o usuario que esta usando la aplicación, se muestra una paginación de 5 usuarios por páginas y un botón de "Agregar Amigo" en cada uno de ellos, el cual no se mostrara si tiene una petición de amistad pendiente tanto enviada y/o recibida o es su amigo ya.

1.5 Buscar entre todos los usuarios de la aplicación

Consiste en un campo de texto donde se introduce una parte del nombre, apellido o email del usuario a buscar y lo mostrara en la misma página con una paginación de 5 usuarios por página.

1.6 Enviar una invitación de amistad a un usuario

Este apartado consiste en la utilización del botón "Agregar Amigo" para enviar la petición de amistad a un amigo y todas las comprobaciones explicadas en el punto 1.4.

1.7 Listar las invitaciones de amistad recibidas

Muestra una vista donde se muestran los usuarios que le han enviado una petición de amistad donde puede aceptarla para convertirse en su amigo.

1.8 Aceptar una invitación recibida

Esta es la parte del botón del punto anterior donde hace que el usuario que envió la petición y el usuario que la acepto se conviertan en amigos en la aplicación.

1.9 Listar los usuarios amigos

Este apartado como el 1.4 y 1.7 realizan una muestra de una lista, de amigos en este caso.

1.10 Seguridad

Se han implementado diferentes mecanismos de seguridad en la aplicación. Se monitoriza el acceso a cualquier URL de la aplicación que no sea el registro e identificación y se redirige a esta ultima porque no se permite el acceso sin una cuenta. A su vez los únicos botones en la barra de navegación que puede utilizar el usuario no registrado son los asociados a esas funciones. Tambien se ha implementado el uso de un logger para tener constancia de los pasos de los usuarios.

2. Parte 2: "Servicios Web"

2.1 Servicios Rest

2.1.1 Identificarse como usuario – token

Tal y como se ha realizado en clase, responde con un token a un usuario identificado y con un mensaje de error si no lo está, todo ello realizado en json.

2.1.2 Usuario Identificado: listar todos los amigos

Lista los amigos de un usuario por medio de su token y la muestra de manera completa en json en una respuesta

2.1.3 Usuario Identificado: Crear un mensaje

Permite la creación de un mensaje a un destinatario con un texto si el usuario tiene un token valido.

2.1.4 Usuario Identificado: Obtener mis mensajes de una "conversación"

Permite la visualización de todos los mensajes de los cuales es emisor o receptor.

2.1.5 Usuario Identificado: Marcar mensaje como leído

Marca el mensaje deseado como leído por medio de su identificador obtenido con el punto 2.1.4.

2.2 Aplicación JQuery

2.2.1 Autentificación del Usuario

Se realiza la autentificación y se muestra un error en caso de que no lo este, a su vez se redirige a su listado de amigos.

2.2.2 Mostrar la lista de amigos

Se realiza un listado de amigos haciendo una petición al api y se muestran en una tabla donde se permite su filtrado por nombre en un cuadro de texto.

2.2.3 Mostrar los mensajes

La tabla anterior muestra un enlace para obtener todos los mensajes entre el usuario que está en sesión y el usuario que se ha escogido de la tabla. Estos mensajes se cargan de manera automática cada 5 segundos.

2.2.4 Crear mensaje

En la misma vista que el punto 2.2.3 se permite la inclusión de un mensaje a esa persona, mediante un cuadro de texto donde se envía el mensaje y un botón para enviar el mismo.

2.2.5 Marcar mensajes como leídos de forma automática

En este apartado marca los mensajes como leídos únicamente los que son para el usuario en sesión, es decir si el usuario envía un mensaje este no será leído hasta que el usuario receptor entre en la conversación, y posteriormente cada 5 segundos por si se enviasen de manera simultánea mensajes.

Aspectos técnicos y de diseño relevantes

Se ha seguido las pautas con las que se ha trabajado en las clases de prácticas, tanto en el diseño de la aplicación, usando Bootstrap, como en el código del servidor. Las pruebas han sido diseñadas como se ha enseñado en la Practica 5 de la asignatura de Sistemas Distribuidos e Internet, usando Selenium y Java para ellos. Cabe destacar que se ha decidido para no tener numerosos métodos que realicen mínimas pruebas, realizar diferentes pruebas adicionales para el correcto funcionamiento de la aplicación en una misma, es decir, no se ha utilizado la convención de PR06_1 si se ha deseado hacer algunas comprobaciones adicionales no dictadas por la práctica para el buen funcionamiento de la aplicación.

Para el uso de logs se ha utilizado la recomendación dada por los profesores y se ha utilizado Log4js para realizar un seguimiento del usuario por la aplicación.

Información necesaria para el despliegue y ejecución

Realmente no se ha realizado ninguna adición o modificación de lo habitual en cuanto al uso de la aplicación realizada en prácticas. El puerto 8081 debe estar liberado de cualquier acción ya que la aplicación usa ese puerto para la conexión con el servidor, de momento se carece de alojamiento en cualquier lugar por lo que el acceso será mediante localhost. El controlador de conexión con la base de datos ya viene proporcionado en el proyecto, así como el controlador de las pruebas de funcionalidad. También se debería revisar la url introducida en las pruebas, tanto del geckodriver como del Mozilla, ya que puede haber variaciones.

En cuanto a la ejecución de pruebas, cabe destacar que como diversas pruebas hacen interacciones con la base de datos la cual es persistente y no se crea cada vez que la base de datos, la cual en la primera ejecución esta de manera optima para ejecutar, al realizar modificaciones en ella en las pruebas estas pueden fallar en la segunda ejecución, por ejemplo, intentar mandar una petición a un amigo cuando este ya tiene uno

porque se hizo en la primera ejecución. Si es necesario para el profesor volver a ejecutar las pruebas por algún motivo siempre puede contactar al email de la portada y se efectuaran las tareas para que pueda hacerlo.

Tambien se ha tomado la decisión, como se aclaro en el mapa de navegación de que el propio usuario no solo no pudiera enviarse una petición a sí mismo, como decía la práctica, sino que a su vez tampoco saliese en el listado de usuarios, ya que, como cuyos fines era solicitar una petición de amistad con otros usuarios y como esa tarea no puede realizarla seria algo contraproducente mantenerle en ese listado.

Por último, se permite en el apartado de Rest que un usuario se mande un mensaje a si mismo, esto ha sido pensado así ya que numerosas páginas de mensajería, véase Gmail, Outlook…, permiten esta tarea y se ha querido mantener esa idea, también en cuanto a marcar los mensajes de manera automática se marcaran únicamente los que son para el usuario en sesión ya que los cuales él es el emisario no tiene sentido marcarlos como leídos ya que lo escribió él.

Conclusión

Este proyecto no solo me ha servido para ponerme a prueba como programador y mis conocimientos adquiridos en la asignatura de Sistemas Distribuidos en Internet sobre el framework NodeJS y otras herramientas como Mongodb, sino ha propiciado un entorno de simulación de un trabajo al tener que realizar diversos procesos, alguno de ellos más complicado que otro, pero en si realistas. Y no solo ateniéndose a la base de programación, también nos han hecho realizar otras tareas como la de pruebas o testing, y la realización de este informe en sí.

Por todo esto y por el trabajo que he realizado, me considero contento con el proyecto final y con el producto realizado.