ENUNCIADO PRÁCTICA 2 – NodeJS y Servicios Web

MY WALLAPOP

Instrucciones generales

La práctica consta de dos partes más la documentación, el peso de cada parte es el siguiente:

- Parte 1 "aplicación web" 4/11 puntos.
- Parte 2 "servicios web" 6/11 puntos.
- Documentación 1/11 punto.

Para que el trabajo pueda ser evaluado los requisitos obligatorios deben estar implementados, con sus correspondientes pruebas automatizadas ***(no se evaluarán los casos de uso que no contengan pruebas),*** La puntuación máxima a la que se opta por implementar todos los requisitos obligatorios de forma perfecta es de ***6.5 sobre 11***, mientras que los requisitos opcionales implementados de forma perfecta llegarán hasta ***3.5 puntos sobre 11*** (ver **¡Error! No se encuentra el origen de la referencia.**).

Tabla 1 Detalle de la puntuación de la práctica

DESCRIPCIÓN			Puntos
PARTE 1 - APLICACIÓN WEB			**4.00**
	REQUISITOS OBLIGATORIOS		**3.30**
	W1	Perfil Público: registrarse como usuario	0.30
	W2	Perfil Público: iniciar sesión	0.30
	W3	Usuario Registrado: Fin de sesión	0.30
	W4	Administrador: Listado de usuarios	0.30
	W5	Administrador: Borrado múltiple de usuarios	0.30
	W6	Usuario registrado: Dar de alta una nueva oferta	0.30
	W7	Usuario registrado: Listado de ofertas propias	0.30
	W8	Usuario registrado: Dar de baja una oferta	0.30
	W9	Usuario registrado: Buscar ofertas	0.30
	W10	Usuario registrado: Comprar una oferta	0.30
	W11	Usuario registrado: Ver el listado de ofertas compradas	0.30
	W12	OPTATIVO: Marcar una oferta como destacada	0.70
PARTE 2 - SERVICIOS WEB			**6.00**
	PARTE 2A - IMPLEMENTACIÓN DE LA API DE SERVICIOS WEB REST		**3.00**
	S1	Identificarse con usuario – token	0.50
	S2	Usuario identificado: Mostrar listado de ofertas disponibles (de otros usuarios)	0.50
	S3	Usuario identificado: Enviar mensajes a una oferta	0.50
	S4	Usuario identificado: Obtener los mensajes de una conversación	0.50
	S5	OPTATIVO: Obtener el listado de conversaciones	0.25
	S6	OPTATIVO: Eliminar una conversación	0.25
	S7	OPTATIVO: Marcar mensaje como leído	0.50
	PARTE 2B - CLIENTE REST – APLICACIÓN WEB CON JQUERY		**3.00**
	C1	Autenticación del usuario	0.40
	C2	Mostrar listado de ofertas disponibles	0.40
	C3	Enviar y mostrar los mensajes de una oferta	0.40
	C4	OPTATIVO: Ver el listado de conversaciones	0.40
	C5	OPTATIVO: Eliminar una conversación	0.40
	C6	OPTATIVO: Marcar mensajes como leídos de forma automática	0.50
	C7	OPTATIVO: Mostrar el número de mensajes sin leer	0.50
INFORME OBLIGATORIO			**1.00**
TOTAL			**11.00**
RESUMEN			
REQUISITOS OBLIGATORIOS			**6.50**
REQUISITOS OPCIONALES			**3.50**
INFORME OBLIGATORIO			**1.00**
TOTAL			**11.00**

Evaluación continua:

- **Trabajo en equipo:** Esta práctica se podrá realizar en ***equipos de hasta 2 alumnos*** (pertenecientes al mismo grupo de prácticas) y su entrega es obligatoria.

  Para los alumnos que trabajen en equipo, se tendrá en cuenta el porcentaje de participación de cada miembro y su ***grado de implicación*** durante el desarrollo del proyecto. Esto se hará mediante dos vías:

  - En el informe de entrega se deberá indicar el porcentaje que hay realizado cada uno, así como un apartado explicativo de las tareas realizadas (*Razonamiento de los porcentajes de participación*).
  - El profesor de prácticas realizará un seguimiento de la actividad de cada alumno en el repositorio GitHub de la entrega para contrastarlo con los porcentajes que indiquen los alumnos en el informe.

- **Corrección por pares:** Además, una vez finalizada la ***entrega el 10 de mayo***, se pondrá a disposición de los alumnos a través del campus virtual, una tarea de corrección por pares que será INDIVIDUAL y OBLIGATORIA.

- **Prueba de autoría:** aquellos alumnos que sean requeridos deberán realizar una defensa presencial del trabajo en el laboratorio de prácticas, consistente en la implementación de nuevos casos de uso. La nota de la práctica se verá condicionada a la correcta implementación de esos casos de uso. El listado de alumnos y fecha de dicha prueba se publicará en el CV con suficiente antelación.

Evaluación diferenciada:

- **Trabajo individual:** Los alumnos se hayan acogido a la evaluación diferenciada deberán realizar este trabajo de ***forma individual obligatoriamente*** y su entrega también es obligatoria.

- **Prueba de autoría:** Todos los alumnos que se hayan acogido a la evaluación diferenciada deberán realizar una defensa presencial del trabajo en el laboratorio de prácticas u online, según evolucione la situación sanitaria actual. Esta defensa consistirá en la implementación de nuevos casos de uso. La nota de la práctica se verá condicionada a la correcta implementación de esos casos de uso. El listado de alumnos y fecha de dicha prueba se publicará en el CV con suficiente antelación.

General:

- ***Mongo DB:*** Es requisito indispensable utilizar una base de datos Mongo en la nube (Mongo Cloud).

Parte1 – Aplicación Web

Se trata de desarrollar una aplicación Web de compra-venta entre particulares (al estilo Wallapop) en el que existirán perfiles de usuario de tipo: Público (Anónimo), Usuario Registrado (Administrador y Usuario Estándar), usando NodeJS y Servicios Web. A continuación, se detallan los requisitos:

W1 Público: Registrarse como usuario

Los usuarios deben poder registrarse en la aplicación aportando email, nombre, apellidos y una contraseña (que deberá repetirse dos veces y coincidir entre sí). Además, por defecto, un usuario tendrá una cuenta de dinero que se iniciará con 100 Euros (este dato se gestionará internamente sin necesidad de un campo de formulario).

Consideraciones importantes:

- El contador de dinero de cada usuario deberá mostrarse, en todas las vistas de acceso privado para el usuario, al lado de su email. Si fuese necesario se podrá almacenar el monto disponible en base de datos.
- El email del usuario no podrá estar repetido en el sistema, se debe informar al usuario de los errores en el proceso de registro.
- Es obligatorio realizar las validaciones del lado del servidor.
- Una vez registrado un usuario será autenticado automáticamente redirigiéndole a la vista de opciones de usuario registrado.
- El perfil por defecto cuando un usuario se registre será de "Usuario Estándar".

Pruebas Funcionales

[Prueba1] Registro de Usuario con datos válidos.

[Prueba2] Registro de Usuario con datos inválidos (email, nombre y apellidos vacíos).

[Prueba3] Registro de Usuario con datos inválidos (repetición de contraseña inválida).

[Prueba4] Registro de Usuario con datos inválidos (email existente).

W2 Público: Iniciar sesión

Suministrando su email y contraseña, un usuario registrado podrá autenticarse ante el sistema. Sólo los usuarios que proporcionen correctamente su email y su contraseña podrán iniciar sesión con éxito.

En caso de que el inicio de sesión fracase, será necesario mostrar un mensaje de error indicando el problema.

En caso de que el inicio de sesión sea correcto se debe dirigir al usuario a la vista que contenga las opciones del perfil correspondiente.

Caso 1: Usuario con perfil de administrador

- Sólo existirá un usuario administrador en el sistema con email admin@email.com y contraseña admin.
- En caso de que el inicio de sesión sea correcto se debe dirigir al usuario a la vista que contenga las opciones del ***perfil Administrador (por ejemplo, lista de usuarios)***.

Caso 2: Usuario Estándar

- En caso de que el inicio de sesión sea correcto se debe dirigir al usuario a la vista que contenga con las opciones del ***perfil Usuario Estándar (por ejemplo, listado de ofertas disponibles)***.

Pruebas Funcionales

[Prueba5] Inicio de sesión con datos válidos.

[Prueba6] Inicio de sesión con datos inválidos (email existente, pero contraseña incorrecta).

[Prueba7] Inicio de sesión con datos inválidos (campo email o contraseña vacíos).

[Prueba8] Inicio de sesión con datos inválidos (email no existente en la aplicación).

W3 Usuario Registrado: Fin de sesión

Disponer una opción de menú que permita finalizar la sesión enviando al usuario al formulario de Inicio de sesión. Este botón solo se mostrará si un usuario se ha autenticado previamente.

Pruebas Funcionales

[Prueba9] Hacer click en la opción de salir de sesión y comprobar que se redirige a la página de inicio de sesión (Login).

[Prueba10] Comprobar que el botón cerrar sesión no está visible si el usuario no está autenticado.

W4 Administrador: Listado de usuarios

Un usuario identificado con perfil de administrador debe poder acceder a una lista en la que figuren todos los usuarios de la aplicación. Para cada usuario se mostrará su email, nombre y apellidos.

Consideraciones importantes:

- No es necesario incluir sistema de paginación en este listado.

Pruebas Funcionales

[Prueba11] Mostrar el listado de usuarios y comprobar que se muestran todos los que existen en el sistema.

W5 Administrador: Borrado múltiple de usuarios.

En la lista anterior donde figuran todos los usuarios de la aplicación, se debe poder seleccionar múltiples usuarios (haciendo uso de checkboxes). Mediante un botón **"Eliminar"** se confirmará el borrado de todos aquellos seleccionados. Al pulsar el botón de Eliminar se deben eliminar todos los usuarios, así como toda la información relativa a los mismos (datos, ofertas, conversaciones, etc.).

Consideraciones importantes:

- En este listado no debería aparecer el usuario administrador o en su defecto no debería poder borrarse.
- No es necesario incluir sistema de paginación en este listado.
- Si se incluye paginación hay que tenerlo en cuenta en las pruebas funcionales.

Pruebas Funcionales

[Prueba12] Ir a la lista de usuarios, borrar el primer usuario de la lista, comprobar que la lista se actualiza y dicho usuario desaparece.

[Prueba13] Ir a la lista de usuarios, borrar el último usuario de la lista, comprobar que la lista se actualiza y dicho usuario desaparece.

[Prueba14] Ir a la lista de usuarios, borrar 3 usuarios, comprobar que la lista se actualiza y dichos usuarios desaparecen.

W6 Usuario registrado: Dar de alta una nueva oferta

Un usuario identificado con perfil de Usuario Estándar, debe poder crear una oferta suministrando: título descriptivo de la oferta, detalle textual de la oferta, fecha de alta de la oferta(Esta fecha puede ser la del sistema) y cantidad solicitada en euros. Los tamaños y tipos de estos campos quedan a criterio del alumno.

Consideraciones importantes:

- Es obligatorio realizar las validaciones del lado del servidor.

Pruebas Funcionales

[Prueba15] Ir al formulario de alta de oferta, rellenarla con datos válidos y pulsar el botón Submit. Comprobar que la oferta sale en el listado de ofertas de dicho usuario.

[Prueba16] Ir al formulario de alta de oferta, rellenarla con datos inválidos (campo título vacío y precio en negativo) y pulsar el botón Submit. Comprobar que se muestra el mensaje de campo obligatorio.

W7 Usuario registrado: Listado de ofertas propias

Un usuario identificado con perfil de Usuario Estándar debe poder acceder a una lista en la que figuren todas sus ofertas. Para cada oferta se mostrará: texto descriptivo de la oferta, detalle de la oferta y cantidad solicitada en euros.

Consideraciones importantes:

- No es necesario incluir sistema de paginación en este listado.

Pruebas Funcionales

[Prueba17] Mostrar el listado de ofertas para dicho usuario y comprobar que se muestran todas las que existen para este usuario.

W8 Usuario registrado: Dar de baja una oferta

Un usuario identificado con perfil de Usuario Estándar debe poder dar de baja una oferta sobre el listado de sus ofertas seleccionado un botón/enlace Eliminar para la oferta que desee suprimir. Al pulsar el botón de Eliminar se deben tanto la oferta como toda la información relativa a la misma.

Consideraciones importantes:

- Un usuario no podrá dar de baja a una oferta de otro usuario.
- Un usuario no podrá dar de baja a una oferta que haya vendido.
- No es necesario incluir sistema de paginación en este listado.

Pruebas Funcionales

[Prueba18] Ir a la lista de ofertas, borrar la primera oferta de la lista, comprobar que la lista se actualiza y que la oferta desaparece.

[Prueba19] Ir a la lista de ofertas, borrar la última oferta de la lista, comprobar que la lista se actualiza y que la oferta desaparece.

W9 Usuario registrado: Buscar ofertas

Incluir un sistema que permita realizar una búsqueda de ofertas por su título. El cuadro de búsqueda contendrá un único campo de texto. La búsqueda debe ser ***insensible a mayúscula y minúsculas***. Por ejemplo, si escribimos la cadena "coch" deberá retornar ofertas en los que la cadena "coch", "COCH", "Coch", etc. sea parte de su título. Si la cadena es vacía deberá mostrar un listado completo con todas las ofertas existentes en el sistema.

Para cada oferta se mostrará: *texto descriptivo de la oferta, detalle de la oferta y cantidad solicitada (en euros).* A la derecha de cada oferta, un enlace o botón ***"Comprar"*** si la oferta está disponible para la compra o bien el texto ***"Vendido"*** si la oferta ya ha sido vendida.

El resultado de la búsqueda debe ser una lista que muestre las coincidencias encontradas.

Consideraciones importantes:

- Esta lista debe incluir un sistema de paginación y mostrar 5 ofertas por página

Pruebas Funcionales

[Prueba20] Hacer una búsqueda con el campo vacío y comprobar que se muestra la página que corresponde con el listado de las ofertas existentes en el sistema

[Prueba21] Hacer una búsqueda escribiendo en el campo un texto que no exista y comprobar que se muestra la página que corresponde, con la lista de ofertas vacía.

[Prueba22] Hacer una búsqueda escribiendo en el campo un texto en ***minúscula o mayúscula*** y comprobar que se muestra la página que corresponde, con la lista de ofertas que contengan dicho texto, independientemente que el título esté almacenado en minúsculas o mayúscula.

W10 Usuario registrado: Comprar una oferta

Sobre el listado resultante de una búsqueda de ofertas, un usuario podrá comprar una oferta haciendo click en el botón "Comprar" correspondiente. Sólo se permite la compra de una oferta si el Contador de dinero del Usuario es igual o superior al precio de la misma. Al comprar una oferta, se deberán realizar las siguientes acciones:

- Decrementar el contador de dinero del comprador en el precio que tenga la oferta.
- Marcar la oferta como comprada para evitar que vuelva a ser vendida.

En caso de que se intente comprar una oferta de mayor cantidad a la disponible se deberá indicar que el comprador no tiene saldo suficiente.

Pruebas Funcionales

[Prueba23] Sobre una búsqueda determinada (a elección de desarrollador), comprar una oferta que deja un saldo positivo en el contador del comprobador. Y comprobar que el contador se actualiza correctamente en la vista del comprador.

[Prueba24] Sobre una búsqueda determinada (a elección de desarrollador), comprar una oferta que deja un saldo 0 en el contador del comprobador. Y comprobar que el contador se actualiza correctamente en la vista del comprador.

[Prueba25] Sobre una búsqueda determinada (a elección de desarrollador), intentar comprar una oferta que esté por encima de saldo disponible del comprador. Y comprobar que se muestra el mensaje de saldo no suficiente.

W11 Usuario registrado: Ver el listado de ofertas compradas

Un usuario deberá disponer de una opción que le muestre el listado de ofertas que haya comprado mostrando para cada oferta los datos: *título, detalle, precio y el email del vendedor.*

Consideraciones importantes:

- No es necesario incluir sistema de paginación en este listado.

Pruebas Funcionales

[Prueba26] Ir a la opción de ofertas compradas del usuario y mostrar la lista. Comprobar que aparecen las ofertas que deben aparecer.

W12 OPTATIVO: Marcar una oferta como destacada

Un usuario puede marcar una oferta como Destacada pagando 20 euros (cantidad fija sin fecha de caducidad). Una oferta podrá marcarse como Destacada tanto al crearla (mediante un check en el formulario de creación de una oferta) como mediante un enlace Normal/Destacada en el listado de ofertas propias de un usuario.

Esta oferta aparecerá en una sección destacada en la vista de la página principal de opciones privadas de Usuario con el Botón/enlace de Compra/Vendido. El contador de dinero del usuario se decrementará en 20 euros.

Pruebas Funcionales

[Prueba27] Al crear una oferta marcar dicha oferta como destacada y a continuación comprobar: i) que aparece en el listado de ofertas destacadas para los usuarios y que el saldo del usuario se actualiza adecuadamente en la vista del ofertante (-20).

[Prueba28] Sobre el listado de ofertas de un usuario con más de 20 euros de saldo, pinchar en el enlace Destacada y a continuación comprobar: i) que aparece en el listado de ofertas destacadas para los usuarios y que el saldo del usuario se actualiza adecuadamente en la vista del ofertante (-20).

[Prueba29] Sobre el listado de ofertas de un usuario con menos de 20 euros de saldo, pinchar en el enlace Destacada y a continuación comprobar que se muestra el mensaje de saldo no suficiente.

Parte 2 “API Servicios web REST” + Cliente ligero JQuery/AJAX

Dentro de la aplicación de la Parte 1 se deberán implementar un cliente ligero JQuery/AJAX cuyo backend será una suite de servicios web REST. Este nuevo cliente abordará parte de las funcionalidades de la aplicación anterior, y además incorporará algunas nuevas funcionalidades opcionales tales como un sencillo chat entre usuario para las ofertas. Conceptualmente las funcionalidades, tanto obligatorias como opcionales, en esta parte de la práctica se organizarán de la siguiente manera. Una vez identificado, un usuario dispondrá de las siguientes opciones:

- Ver listado de ofertas de otros usuarios (donde el usuario no es propietario). Desde este listado para cada oferta aparecerá un botón o enlace para **iniciar una conversación** con el propietario de la oferta. Además, se deberá validar en el servidor que un usuario no pueda enviar un mensaje a su propia oferta.
- Ver listado de conversaciones ya iniciadas. Esta opción mostrará las conversaciones en las que el usuario es propietario, así como aquellas en las que es interesado. Para cada conversación se podrá:
    - Enviar mensajes a dicha conversación.
    - Borrar la conversación completa.

Los casos de usos S1al S4 y C1 al C3 son de carácter obligatorio, para que el trabajo pueda ser evaluado. Además, los casos de uso C1-C7 deben incluir sus correspondientes pruebas automatizadas (no se evaluarán los casos de uso que no contengan pruebas). Mientras que los casos de uso S1 – S7 no requerirán pruebas automatizadas.

Para ver la puntuación de cada caso de uso (ver Tabla 1).

Parte 2A – API de Servicios Web REST

S1 Identificarse como usuario vía token

El servicio recibe las credenciales de un usuario (email y contraseña), en caso de que exista coincidencia con algún usuario almacenado en la base de datos retornará un token de autenticación. Sí no hay coincidencia retornará un mensaje de error de inicio de sesión no correcto.

Pruebas Funcionales

- En este caso de uso no será necesario realizar las pruebas funcionales.

S2 Usuario identificado: Mostrar listado de ofertas disponibles (Sólo las ofertas de los otros usuarios)

Realizar un servicio REST que retorne una lista todas las ofertas disponibles en base datos, correspondientes a los usuarios diferentes al usuario identificado.

Para permitir listar las ofertas el usuario debe de estar identificado en la aplicación, por lo tanto, la petición debe contener un token de seguridad válido.

Pruebas Funcionales

- En este caso de uso no será necesario realizar las pruebas funcionales.

S3 Usuario identificado: Enviar mensajes a una oferta

Crear un servicio REST que permita que un usuario pueda enviar mensajes (tipo chat) a una conversación entre dos usuarios (el interesado y el propietario de la oferta).

Se abrirá una conversación por cada oferta entre cada par de usuarios. La primera vez que un usuario envíe un mensaje a una oferta (usuario interesado) se creará dicha conversación y, a partir de ahí, los mensajes enviados por ambos lados (el interesado y el propietario) se irán incorporando a la conversación. Para cada mensaje se mostrará y almacenará: el autor del mensaje, la fecha/hora del mensaje, el texto, leído (true/false).

Consideraciones importantes:

- Por defecto el mensaje se crea como no leído.
- Para permitir enviar un nuevo mensaje, el usuario tiene que estar identificado, por lo tanto, la petición debe contener un token de seguridad válido.
- Sólo los usuarios interesados pueden enviar el primer mensaje a una oferta y será a partir de ese instante cuando al usuario propietario le saldrá esa conversación en su lista de conversaciones.
- Un posible enfoque de almacenamiento de los mensajes, sería crear una colección donde los campos de un mensaje sean: el id del usuario interesado, id del propietario, id de la oferta, texto del mensaje, fecha/hora del mensaje y leído (true/false).

Pruebas Funcionales

- En este caso de uso no será necesario realizar las pruebas funcionales.

S4 Usuario identificado: Obtener los mensajes de una conversación

Crear un servicio REST que dado un usuario identificado y una oferta muestre el listado de los mensajes de una conversación.

Para permitir obtener los mensajes de una conversación el usuario identificado debe ser el interesado o propietario de una oferta, por lo tanto, la petición debe contener un token de seguridad válido.

Pruebas Funcionales

- En este caso de uso no será necesario realizar las pruebas funcionales.

S5 OPTATIVO: Obtener el listado de conversaciones

Crear un servicio REST que dado un usuario identificado retorne todas las conversaciones vinculadas a dicho usuario:

- Por un lado, aquellas conversaciones en las que el usuario aparezca como interesado.
- Y por otro lado, aquellas conversaciones donde el usuario aparezca como propietario.

Para permitir obtener los mensajes de una conversación la petición debe contener un token de seguridad válido.

Pruebas Funcionales

- En este caso de uso no será necesario realizar las pruebas funcionales.

S6 OPTATIVO: Eliminar una conversación

Crear un servicio REST que permita eliminar una conversación. El servicio recibe el identificador de la conversación. Al eliminar una conversación se deberán eliminar todos los mensajes relacionados.

Para eliminar una conversación, el usuario identificado debe ser el *interesado o propietario de los mensajes de dicha conversación,* por lo tanto, la petición debe contener un token de seguridad válido.

Pruebas Funcionales

- En este caso de uso no será necesario realizar las pruebas funcionales.

S7 OPTATIVO: Marcar mensaje como leído

Crear un servicio REST que permita marcar un mensaje como leído, la propiedad leída debe tomar el valor true. El servicio recibe el identificador del mensaje.

Para permitir cambiar el estado de un mensaje, el usuario identificado puede ser tanto un usuario interesado como propietario, por lo tanto, la petición debe contener un token de seguridad válido.

Pruebas Funcionales

- En este caso de uso no será necesario realizar las pruebas funcionales.

Parte 2B – Cliente ligero JQuery/Ajax

En esta parte hay que a desarrollar una aplicación web jQuery-AJAX que consuma los servicios web REST desarrollados en el punto anterior. Esta aplicación debe permitir el intercambio de mensajes en tiempo real.

El cliente también se incluirá dentro del mismo proyecto, en la carpeta "public".

C1 Autenticación del usuario

Haciendo uso de la API REST, la aplicación web debe permitir la autenticación a través de un formulario donde se solicite el correo y la contraseña de usuario, se debe informar si la autenticación no se realiza con éxito.

Pruebas Funcionales

[Prueba30] Inicio de sesión con datos válidos.

[Prueba31] Inicio de sesión con datos inválidos (email existente, pero contraseña incorrecta).

[Prueba32] Inicio de sesión con datos válidos (campo email o contraseña vacíos).

C2 Mostrar listado de ofertas disponibles

Haciendo uso de la API REST, la aplicación web deberá disponer de una opción que le muestre el listado de ofertas disponibles en la Web, mostrando para cada oferta los datos: título, detalle, precio y el email del vendedor.

Se deberán **mostrar sólo las ofertas de otros usuarios**, excluyendo las ofertas realizadas por el usuario identificado.

Consideraciones importantes:

- No es necesario incluir sistema de paginación en este listado.

Pruebas Funcionales

[Prueba33] Mostrar el listado de ofertas disponibles y comprobar que se muestran todas las que existen, menos las del usuario identificado.

C3 Enviar y mostrar los mensajes de una oferta.

Haciendo uso de la API REST y tomando como punto de partida la vista que muestra el listado resultante de una búsqueda de ofertas (punto anterior), se deberá realizar lo siguiente:

1) Al lado de cada oferta añadir un botón o enlace que permita enviar mensajes a la oferta correspondiente. Una vez se pulse el botón o enlace se deberá abrir una nueva vista donde se muestre un "chat".
2) En este "chat" se visualizarán los mensajes del usuario interesado el propietario de la oferta. Esta vista deberá tener al menos:
   a) La lista con todos los mensajes correspondientes al interesado y el propietario de la oferta.
   b) Formulario para enviar un nuevo mensaje a esta conversación.
3) El formulario anterior (2.b) deberá incorporar los nuevos mensajes del usuario identificado a la conversación.

Consideraciones importantes

La lista de mensajes de la oferta que está viendo un usuario, debe actualizarse en tiempo real, sin necesidad de recargar la página ni de pulsar ningún botón. Es decir, debe haber un proceso automático que compruebe cada segundo si hay nuevos mensajes en esa conversación y los incorpore a la lista.

Pruebas Funcionales

[Prueba34] Sobre una búsqueda determinada de ofertas (a elección de desarrollador), enviar un mensaje a una oferta concreta. Se abriría dicha conversación por primera vez. Comprobar que el mensaje aparece en el listado de mensajes.

[Prueba35] Sobre el listado de conversaciones enviar un mensaje a una conversación ya abierta. Comprobar que el mensaje aparece en el listado de mensajes.

C4 OPTATIVO: Ver el listado de conversaciones

Haciendo uso de la API REST un usuario podrá ver un listado de las conversaciones abiertas y podrá reanudar cada una de las conversaciones enviando un nuevo mensaje y viendo los anteriores. El listado mostrará el email del ofertante y el título de la oferta además de un botón/enlace para reanudar la conversación.

Pruebas Funcionales

[Prueba36] Mostrar el listado de conversaciones ya abiertas. Comprobar que el listado contiene las conversaciones que deben ser.

C5 OPTATIVO: Eliminar una conversación

Haciendo uso de la API REST y sobre el listado anterior se podrá borrar una conversación incluyendo un botón/enlace "Eliminar" para cada conversación. Al pinchar sobre el botón/enlace se eliminará dicha conversación y todos los datos relacionados.

Pruebas Funcionales

[Prueba37] Sobre el listado de conversaciones ya abiertas. Pinchar el enlace Eliminar de la primera y comprobar que el listado se actualiza correctamente.

[Prueba38] Sobre el listado de conversaciones ya abiertas. Pinchar el enlace Eliminar de la última y comprobar que el listado se actualiza correctamente.

C6 OPTATIVO: Marcar mensajes como leídos de forma automática

Haciendo uso de la API REST y al entrar en un chat (conversación) todos los mensajes no leídos deben ser marcados como leídos. Junto a cada mensaje mostrado en el chat debe incluirse un <leído> al final (si tiene el estado leído).

Al igual que la lista de mensajes el estado debe actualizarse en tiempo real, sin necesidad de recargar la página ni de pulsar ningún botón. Es decir, debe haber un proceso automático que compruebe cada segundo si hay nuevos mensajes en esa conversación o cambios en los estados leído.

Pruebas Funcionales

[Prueba39] Identificarse en la aplicación y enviar un mensaje a una oferta, validar que el mensaje enviado aparece en el chat. Identificarse después con el usuario propietario de la oferta y validar que tiene un mensaje sin leer, entrar en el chat y comprobar que el mensaje pasa a tener el estado leído.

C7 OPTATIVO: Mostrar el número de mensajes sin leer

Haciendo uso de la API REST se debe mostrar junto a cada conversación cuantos mensajes sin leer hay en esa conversación.

El número de mensajes sin leer debe actualizarse en tiempo real, sin necesidad de recargar la página ni de pulsar ningún botón. Es decir, debe haber un proceso automático que compruebe cada segundo si hay nuevos mensajes.

Pruebas Funcionales

[Prueba40] Identificarse en la aplicación y enviar tres mensajes a una oferta, validar que los mensajes enviados aparecen en el chat. Identificarse después con el usuario propietario de la oferta y validar que el número de mensajes sin leer aparece en su oferta.

Pruebas automatizadas

Se deberá suministrar un proyecto Java **JUnit** con un mínimo de pruebas unitarias empleando el framework Selenium (Se suministrará en el campus virtual un proyecto plantilla de ejemplo).

Las clases de equivalencia mínimas (válidas e inválidas) que deberán realizarse se encuentran al final de cada requisito.

En caso de desear incluir más se deberán incluir al final de la clase de pruebas JUnit.

Todas las pruebas deben incluir el click en las opciones de menú correspondientes, etc. igual que lo haría un usuario real.

Las pruebas realizadas en la primera actividad podrán utilizarse durante el desarrollo de esta actividad.

Informe de entrega y plantilla de casos de prueba

Se deberá entregar un informe técnico en formato PDF, así como un catálogo de casos de prueba en formato XLSX (que a su vez es formulario de autoevaluación). Se suministran dos plantillas que deberán tomarse como base para la elaboración de dichos documentos:

- Una plantilla Word para el informe (sdi-entrega2-n.docx). (Se entregará en formato PDF).
- Una plantilla Excel para los casos de prueba (sdi-entrega2-n.xlsx). (Se entregar en formato Excel).

Ambos documentos deberán ser renombrados cambiando la n por el código ID-GIT de los miembros del equipo (ej. sdi-entrega2-2021-801-2021-804.pdf, sdi-entrega2-2021-801-2021-804.xlsx).

Informe de entrega

El contenido del informe deberá ser el siguiente:

- En la portada se deberá incluir:
    - Los datos personales de los autores: Nombre y apellidos, Emails de ambos, Código ID GIT de cada uno y los porcentajes de participación de cada miembro del equipo.
- Razonamiento de los porcentajes de participación.
- Mapa de Navegación.
- Una descripción clara y detallada aquellos aspectos técnicos y/o diseño que considere relevantes.
    - No se trata de describir qué es NodeJS o Servicios Web.

- Se trata de detallar la toma de decisiones a la hora de implementar la práctica en los aspectos técnicos y/o de diseño relevantes.

- Cualquier otra información necesaria para una descripción razonablemente detallada de lo entregado y su correcto despliegue y ejecución.

Plantilla de casos de prueba

La plantilla suministrada para cumplimentar los casos de prueba realizados presenta dos hojas:

- Una (denominada Pruebas) con una serie de tablas, de las cuales el alumno deberá rellenar aquella indicada en amarillo (Casos de Prueba) y donde deberá reflejar los casos de prueba que haya diseñado, el estado de dicho caso de prueba y una explicación/aclaración sobre el mismo en caso de fallo o no haber rellenado el caso. También en el caso de incluir casos de prueba extra deberá incluir en la columna explicación/aclaración en que consiste dicha prueba.

- Una segunda hoja (Instrucciones) con las instrucciones para cumplimentar la primera hoja.

Corrección por pares

Una vez finalizado el plazo de entrega el 10 de mayo, **será obligatorio completar una tarea por de corrección por pares**.

Cada alumno, deberá rellenar una plantilla de corrección que estará disponible en el Campus Virtual, tal y como se indica a continuación:

- Cada alumno, independientemente de que forme equipo con otro, deberá rellenar de **forma individual** la plantilla aplicándola al ***código que ha entregado.***

- Cada alumno, independientemente de que forme equipo con otro, deberá rellenar de **forma individual** la plantilla aplicándola al ***código de la práctica de otro compañero***. Esta otra práctica, será proporcionada en tiempo y forma por los profesores de la asignatura mediante el Campus Virtual.

El plazo para cumplimentar esta plantilla para los dos casos (práctica propia y práctica de compañero), será indicado por los profesores cuando se envíen los proyectos a evaluar.

Aspecto Generales

Seguridad

Deberán tenerse en cuenta los siguientes aspectos de seguridad:

- Emplear la técnica de autenticación/autorización más adecuada a este contexto.

- En caso necesario comprobar el rol de cada usuario para el acceso a funcionalidades dependientes de rol.

- Registrar el acceso de los usuarios y la actividad en un Logger (por ejemplo, log4js).

Arquitectura

La aplicación deberá estar obligatoriamente diseñada siguiente el patrón arquitectónico visto en clase. Usando siempre de forma correcta controladores, modelos y vistas. La utilización incorrecta de elementos de esta arquitectura ***será fuertemente penalizada***.

Los servicios web REST deben seguir la arquitectura RESTful, se deben utilizar URLs y métodos HTTP adecuados en cada caso.

Otros aspectos que serán evaluados

- Claridad y calidad de la implementación del código JavaScript.
- Calidad de la implementación de las vistas y usando todas las funcionalidades vistas en clase.
- Validaciones en los formularios.
- Diseño de la interfaz de la aplicación.

Forma de entrega y evaluación

El número n de cada trabajo será el ID-GIT del integrante del grupo (en caso de un integrante) o la concatenación de los ID-GIT (en caso de dos ej: 2021-801-2021-804) Según ese número se deberán crear los proyectos NodeJS y las pruebas unitarias (Selenium) con los nombres **sdi-entrega2-n** y **sdi-entrega2-test-n** respectivamente.

Según lo anterior la entrega consistirá en los siguientes tres puntos:

1) Subir los dos proyectos a un repositorio GIT con nombre **sdi-entrega2-n** E invitar al usuario **sdigithubuniovi** como colaborador. Sobre dichos repositorios deberán realizarse actualizaciones frecuentes para que se pueda hacer un seguimiento del ritmo de trabajo.
2) Subir a la tarea del Campus Virtual correspondiente un archivo ZIP (usando el formato ZIP) con el nombre **sdi- entrega2-n.zip** (en minúsculas) y que deberá contener en su raíz:
   - El INFORME OBLIGATORIO en formato PDF con nombre **sdi-entrega2-n.pdf y el archivo Excel con el catálogo de casos de prueba (sdi-entrega2-n.xlsx).**
   - El proyecto NodeJS en formato carpeta (no comprimido) con el nombre **sdi-entrega2-n.**
   - El proyecto Java eclipse con los casos de prueba en formato carpeta (no comprimido) con el nombre **.\sdi-entrega2-test-n**
   - En resumen, el zip deberá contener en su raíz:
     - sdi-**entrega2**-n.pdf (Archivo PDF suministrado y renombrado cambiado la n).
     - sdi-**entrega2**-n.xlsx (Archivo Excel suministrado y renombrado cambiado la n).
     - sdi-**entrega2**-n (Carpeta proyecto NodeJS renombrada cambiado la n)
     - sdi-**entrega2-test**-n (Carpeta proyecto Java eclipse renombrada cambiado la n)

Cuando se abra el plazo para la corrección por pares, será obligatorio realizar la tarea propuesta en el apartado **0. Corrección por pares**

El protocolo de prueba

Para probar cada proyecto el profesor realizará los siguientes pasos:

a) Importar y ejecutar la aplicación NodeJS.
b) Importará y ejecutar el proyecto JUnit en STS.

Fecha máxima de entrega

La fecha de entrega será el ***lunes 10 de mayo del 2021 a las 23:55.***

Consideraciones importantes:

- ***No se aceptará ningún trabajo fuera de plazo.***
- ***Todos los trabajos deben entregarse por el campus virtual.***

Evaluación

La calificación máxima que se puede obtener será de 11 puntos, en base al siguiente reparto: 7.5 puntos pertenecen a la funcionalidad obligatoria, 3.5 puntos a la parte opcional.

Se penalizará:

- Que la compilación del código genere "*warnings*".
- Que se presenten problemas durante el despliegue.