Sistemas Distribuidos e Internet

Curso 2022/2023

PRÁCTICA 2 DE ENTREGA – Node.js y Servicios web

MY WALLAPOP

INSTRUCCIONES GENERALES **2**

PARTE1 – APLICACIÓN WEB **4**

W1 Público: Registrarse como usuario *4*
W2 Público: Iniciar sesión *4*
W3 Usuario Registrado: Fin de sesión *5*
W4 Administrador: Listado de usuarios del sistema *5*
W5 Administrador: Borrado múltiple de usuarios. *5*
W6 Usuario registrado: Dar de alta una nueva oferta *6*
W7 Usuario registrado: Listado de ofertas propias *6*
W8 Usuario registrado: Dar de baja una oferta *6*
W9 Usuario registrado: Buscar ofertas *7*
W10 Usuario registrado: Comprar una oferta *7*
W11 Usuario registrado: Ver el listado de ofertas compradas *8*
W12 Usuario registrado: Marcar una oferta como destacada *8*
W13 Seguridad y auditoria de la aplicación *9*

PARTE 2 "API SERVICIOS WEB REST" + CLIENTE LIGERO JQUERY/AJAX **10**

PARTE 2A – API DE SERVICIOS WEB REST **11**

S1 Identificarse como usuario vía token *11*
S2 Usuario identificado: Obtener el listado de ofertas disponibles (Sólo las ofertas de los otros usuarios) *11*
S3 Usuario identificado: Enviar mensajes a una oferta *11*
S4 Usuario identificado: Obtener los mensajes de una conversación *12*
S5 Usuario identificado: Obtener el listado de conversaciones *12*
S6 OPTATIVO: Eliminar una conversación *13*
S7 OPTATIVO: Marcar mensaje como leído *13*

PARTE 2B – CLIENTE LIGERO JQUERY/AJAX **13**

C1 Autenticación del usuario *13*
C2 Usuario registrado: Mostrar listado de ofertas disponibles *13*
C3 Usuario registrado: Establecer conversación en una oferta *14*
C4 Usuario registrado: Ver el listado de conversaciones *14*
C5 OPTATIVO - Usuario registrado: Eliminar una conversación *15*
C6 OPTATIVO - Usuario registrado: Marcar mensajes como leídos de forma automática *15*
C7 OPTATIVO - Usuario registrado: Mostrar el número de mensajes sin leer *15*

PRUEBAS AUTOMATIZADAS **15**

DATOS DE PRUEBA Y TIEMPO DE EJECUCIÓN DE PRUEBAS **16**

INFORME DE ENTREGA Y PLANTILLA DE CASOS DE PRUEBA **16**

ASPECTOS TRANSVERSALES EN LA EVALUACIÓN **17**

ASPECTOS GENERALES **17**

Arquitectura *17*
Otros aspectos que serán evaluados *17*

PROCESO DE ENTREGA Y PROTOCOLO DE PRUEBA **17**

El protocolo de prueba *18*
Fecha máxima de entrega *18*

Instrucciones generales

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

- Parte 1 "aplicación web" 4.5/11 puntos.
- Parte 2 "servicios web" 5.5/11 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 ***8.0 sobre 11*** (ver **¡Error! No se encuentra el origen de la referencia.**). La p untuación establecida para cada requisito es una puntuación máxima, **pudiendo llegar a ser negativa**, afectando a la puntuación total de la práctica**.** La **responsabilidad** de que no se cumpla lo especificado en este párrafo es, **en cualquier tipo de caso, grupal y no individual,** en evaluación continua.

Los **requisitos opcionales permiten alcanzar una puntuación máxima de 11.0 puntos** (ver **¡Error! N o se encuentra el origen de la referencia.**), siempre y cuando se implementen todos los casos de uso, a la perfección a nivel de funcionalidad y pruebas automatizadas. Perfección hace referencia a aspectos tales como: validación, manejo de errores, documentación del código, etc. **La puntuación mínima de un ejercicio opcional será de 0.0 puntos**.

Tabla 1 Detalle de la puntuación de la práctica

DESCRIPCIÓN			Puntos
PARTE1 - APLICACIÓN WEB			**4.50**
	REQUISITOS OBLIGATORIOS		**4.50**
	W1	Perfil Público: registrarse como usuario	0.25
	W2	Perfil Público: iniciar sesión	0.25
	W3	Usuario Registrado: Fin de sesión	0.25
	W4	Administrador: Listado de usuarios	0.25
	W5	Administrador: Borrado múltiple de usuarios	0.50
	W6	Usuario registrado: Dar de alta una nueva oferta	0.25
	W7	Usuario registrado: Listado de ofertas propias	0.25
	W8	Usuario registrado: Dar de baja una oferta	0.25
	W9	Usuario registrado: Buscar ofertas	0.25
	W10	Usuario registrado: Comprar una oferta	0.25
	W11	Usuario registrado: Ver el listado de ofertas compradas	0.25
	W12	Usuario registrado: Marcar una oferta como destacada	0.50
	W13	Seguridad y auditoria de la aplicación	1.00
PARTE2 - SERVICIOS WEB			**5.50**
	PARTE2A - IMPLEMENTACIÓN DE LA API DE SERVICIOS WEB REST		**2.50**
	S1	Identificarse con usuario – token	0.25
	S2	Usuario identificado: Obtener el listado de ofertas disponibles (de otros usuarios)	0.25
	S3	Usuario identificado: Enviar mensajes a una oferta	0.50
	S4	Usuario identificado: Obtener los mensajes de una conversación	0.25
	S5	Usuario identificado: Obtener el listado de conversaciones	0.25
	S6	OPTATIVO: Eliminar una conversación	0.50
	S7	OPTATIVO: Marcar mensaje como leído	0.50
	PARTE2B - CLIENTE REST – APLICACIÓN WEB CON JQUERY		**3.00**
	C1	Autenticación del usuario	0.25
	C2	Usuario registrado: Mostrar listado de ofertas disponibles	0.25
	C3	Usuario registrado: Establecer conversación en una oferta	0.25
	C4	Usuario registrado: Ver el listado de conversaciones	0.25
	C5	OPTATIVO-Usuario registrado: Eliminar una conversación	0.50
	C6	OPTATIVO-Usuario registrado: Marcar mensajes como leídos de forma automática	0.75
	C7	OPTATIVO - Usuario registrado: Mostrar el número de mensajes sin leer	0.75
INFORME OBLIGATORIO			**1.00**
TOTAL			**11.00**

RESUMEN	
REQUISITOS OBLIGATORIOS	**7.00**
REQUISITOS OPCIONALES	**3.00**
INFORME OBLIGATORIO	**1.00**
TOTAL	**11.00**

Evaluación continua:

- **Trabajo en equipo:** Esta práctica se deberá realizarse en equipo (los equipos formados en los grupos de prácticas) y su entrega es obligatoria.
- **Coevaluación holística:** La calificación final asignada a cada integrante de un equipo (Nota Individual del Estudiante, NIE) se calculará mediante el método denominado coevaluación holística: coevaluación, porque cada alumno va a valorar el rendimiento y comportamiento de sus compañeros, así como de uno mismo, y holístico porque esa valoración deberá ser una medida global (actitud, cumplimiento de plazos, …) de sus compañeros en lo que compete al desarrollo de esta práctica.

  Con el fin de simplificar ese cálculo cada alumno asignará una valoración entre varios niveles de posibles de Ciudadanía de Equipo (CE) a sus compañeros. Los valores de la escala Likert de CE van desde:

  - "EXCELENTE" (100%): Contribución muy destacada y constante en el trabajo de equipo, con un rendimiento sobresaliente, hasta
  - NO MOSTRADO (0%): No jugó un papel efectivo en el trabajo en equipo y/o asistencia y compromisos virtualmente inexistentes.

  De esta forma cada alumno emitirá un listado de N valoraciones siendo un equipo de N miembros. Con la matriz de NxN valores de un equipo, el profesor correspondiente calculará para cada alumno de dicho equipo el Factor Individual de Coevaluación (FIC). De esta forma se calculará la Nota Individual de cada Estudiante (NIE) a partir del FIC y de la Nota obtenida por el Equipo en la práctica entregada (NE):

  $$NIE = FIC * NE$$

  Se enviará un mensaje desde el CV indicando cual será el procedimiento tanto para enviar los niveles de CE. Además, en clase se explicará con algo más de detalle cómo se realizará el cálculo del FIC, en caso de ser necesario.

- **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. 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 local, debido a la latencia que hay Mongo en la nube (Mongo Cloud). Esto último podría ocasionar fallos y errores en las pruebas automatizadas.

Parte1 – Aplicación web

El objetivo de la práctica es desarrollar una aplicación web basada en la compraventa de artículos, similar a Wallapop. Existirán perfiles de usuario de tipo: Público (Anónimo), Usuario Registrado (Administrador y Usuario Estándar), usando Node.js 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, fecha de nacimiento (DD/MM/AAAA), 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 (en la barra de navegación). 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 "listado de ofertas propias" [Requisito Obligatorio W2].
- La fecha de nacimiento deberá ser una fecha válida.
- La contraseña se deberá guardar cifrada en la base de datos.
- 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, apellidos y fecha de nacimiento 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"**. Se puede crear un usuario administrador de prueba e insertarlo desde código o directamente en la base de datos.

- En caso de que el inicio de sesión sea correcto se debe redirigir al usuario a la vista: "listado de todos los usuarios de la aplicación" [Requisito Obligatorio W4].

Caso 2: Usuario Estándar

- En caso de que el inicio de sesión sea correcto se debe redirigir al usuario a la vista "listado de ofertas propias" [Requisito Obligatorio W7].

Pruebas Funcionales

[Prueba5] Inicio de sesión con datos válidos (administrador).

[Prueba6] Inicio de sesión con datos válidos (usuario estándar).

[Prueba7] Inicio de sesión con datos inválidos (usuario estándar, email existente, pero contraseña incorrecta).

[Prueba8] Inicio de sesión con datos inválidos (campo email o contraseña vacíos).

W3 Usuario Registrado: Fin de sesión

Incluir en el menú de navegación una opción 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 del sistema

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, apellidos.

Consideraciones importantes:

- Es necesario incluir sistema de paginación en este listado.

Pruebas Funcionales

[Prueba11] Mostrar el listado de usuarios. Comprobar que se muestran todos los que existen en el sistema, contabilizando al menos el número de usuarios.

W5 Administrador: Borrado múltiple de usuarios.

En la vista del [Requisito Obligatorio W4] donde figuran todos los usuarios de la aplicación se debe poder seleccionar múltiples usuarios (con un checkbox) y se dispondrá de un botón **"Eliminar"** para confirmar el borrado de todos aquellos seleccionados. Al pulsar el botón de Eliminar se deben eliminar todos los usuarios seleccionados, así como toda la información relativa a los mismos, véase: datos, ofertas, conversaciones, etcétera. Un usuario Administrador no podrá borrarse a sí mismo.

Consideraciones importantes:

- En este listado no debería aparecer el usuario administrador o en su defecto no debería poder borrarse (validación tanto en lado cliente como en lado servidor).
- Es necesario incluir sistema de paginación en este listado y 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.

[Prueba15] Intentar borrar el usuario que se encuentra en sesión y comprobar que no ha sido borrado (porque no es un usuario administrador o bien, porque, no se puede borrar a sí mismo, si está autenticado).

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 publicación 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 para cada uno de los siguientes valores: título, detalle y precio.

Pruebas Funcionales

[Prueba16] 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.

[Prueba17] 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 inválido.

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: título descriptivo de la oferta, detalle de la oferta, fecha de publicación de la oferta y cantidad solicitada en euros.

Consideraciones importantes:

- Es necesario incluir sistema de paginación en este listado y tenerlo en cuenta en las pruebas funcionales.

Pruebas Funcionales

[Prueba18] 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

En el listado de ofertas propias [Requisito Obligatorio W7], un Usuario Estándar podrá dar de baja una oferta. Para cada una de éstas, se presentará un botón/enlace "Eliminar" que, al hacer clic, eliminará la oferta y 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.

Pruebas Funcionales

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

[Prueba20] Ir a la lista de ofertas, borrar la última oferta de la lista, comprobar que la lista se actualiza y que la oferta desaparece.

[Prueba21] Ir a la lista de ofertas, borrar una oferta de otro usuario, comprobar que la oferta no se borra.

[Prueba22] Ir a la lista de ofertas, borrar una oferta propia que ha sido vendida, comprobar que la oferta no se borra.

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: título descriptivo de la oferta, detalle de la oferta, fecha de publicación 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
- Al hacer click en una de las páginas debe mantenerse el parámetro de búsqueda.

Pruebas Funcionales

[Prueba23] 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

[Prueba24] 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.

[Prueba25] 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.

Consideraciones importantes:

- Sólo se permite la compra de una oferta si:
  - El contador de dinero del Usuario es igual o superior al precio de la misma.
  - La oferta que desea comprar no está ya vendida.
  - La oferta que desea comprar no es suya.

- Al comprar una oferta, se deberán realizar las siguientes acciones:
  - Comprobar en el servidor que se cumplen las condiciones anteriores.
    - Si no se cumple alguna de las condiciones anteriores, se mostrará un mensaje de error identificativo al usuario.
  - 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.

Pruebas Funcionales

[Prueba26] 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.

[Prueba27] 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.

[Prueba28] 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:

- Es necesario incluir sistema de paginación en este listado.

Pruebas Funcionales

[Prueba29] Ir a la opción de ofertas compradas del usuario y mostrar la lista. Comprobar que aparecen las ofertas que deben aparecer.

W12 Usuario registrado: 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

[Prueba30] 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 (comprobar saldo antes y después, que deberá diferir en 20€).

[Prueba31] 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 (comprobar saldo antes y después, que deberá diferir en 20€).

[Prueba32] 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.

W13 Seguridad y auditoria de la aplicación

Deberá diseñarse adecuadamente la política de seguridad de la aplicación para que no existan situaciones de vulnerabilidad en el acceso a recursos/acciones de usuarios registrados. Además, se evaluará en este apartado que:

- Se emplea la técnica de autenticación/autorización más adecuada a este contexto.
- En caso de que sea 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). Incluyendo: la fecha, la acción y la información que el grupo considere relevante para cada caso:
    - Registro de todas y cada una de las peticiones recibidas en el controlador/router correspondiente.
    - Registro de las altas de usuarios en el sistema.
    - Registro de los inicios de sesión tanto exitosos como fallidos.
    - Registro de los cierres de sesión.
- Almacenar en una colección de la base de datos la información de los cuatro tipos de logs indicados en el punto anterior (peticiones, altas, inicio de sesión con éxito y sin éxito y cierres de sesión.). La información mínima que se deberá almacenar en la base de datos será:
    - Para cada petición:
        - Tipo de log: “PET”.
        - Fecha-Hora: Fecha y hora en que se registró el log en formato timestamp.
        - Texto descriptivo: Un texto que incluya el mapping del controlador que recibe la petición, el método http, y los parámetros recibidos si los hubiera.
    - Para cada alta:
        - Lo mismo que para cada petición, pero con tipo de log = “ALTA”. En este caso el log aparecerá duplicado: uno por la petición y otro por el alta.
    - Para cada inicio de sesión con éxito:
        - Tipo de log: “LOGIN-EX”.
        - Fecha-Hora: Fecha y hora en que se realizó el login en formato timestamp.
        - Texto descriptivo: el email del usuario que ha entrado en sesión.
    - Para cada inicio de sesión sin éxito:
        - Tipo de log: “LOGIN-ERR”.
        - Fecha-Hora: Fecha y hora en que se realizó el intento de login en formato timestamp.
        - Texto descriptivo: el email del usuario que intentó entrar en sesión.
    - Para cada vez que el usuario sale de sesión:
        - Tipo de log: “LOGOUT”.

- Fecha-Hora: Fecha y hora en que se realizó el logout en formato timestamp.
- Texto descriptivo: el email del usuario que salió de sesión.

- El usuario administrador podrá visualizar todos los logs ordenados por fecha-hora de más reciente a más antiguo. El listado podrá ser filtrado por cada tipo de log indicado arriba y el resultado siempre será ordenado por fecha de más reciente a más antiguo.
- El usuario administrador podrá borrar todos los logs con una opción de menú.

Pruebas Funcionales

[Prueba33] Intentar acceder sin estar autenticado a la opción de listado de usuarios. Se deberá volver al formulario de login.

[Prueba34] Intentar acceder sin estar autenticado a la opción de listado de conversaciones [REQUISITO OBLIGATORIO S5]. Se deberá volver al formulario de login.

[Prueba35] Estando autenticado como usuario estándar intentar acceder a una opción disponible solo para usuarios administradores (Añadir menú de auditoria (visualizar logs)). Se deberá indicar un mensaje de acción prohibida.

[Prueba36] Estando autenticado como usuario administrador visualizar todos los logs generados en una serie de interacciones. Esta prueba deberá generar al menos dos interacciones de cada tipo y comprobar que el listado incluye los logs correspondientes.

[Prueba37] Estando autenticado como usuario administrador, ir a visualización de logs, pulsar el botón/enlace borrar logs y comprobar que se eliminan los logs de la base de datos.

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.

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

[Prueba38] Inicio de sesión con datos válidos.

[Prueba39] Inicio de sesión con datos inválidos (email existente, pero contraseña incorrecta).

[Prueba40] Inicio de sesión con datos inválidos (campo email o contraseña vacíos).

Comprobar en las tres pruebas el estado HTTP retornado y la estructura JSON retornada por el servicio REST de identificación. El token no es necesario comprobarlo ya que no es viable. Emplear la API RestAssured (ver ejemplos en el proyecto plantilla compartido).

S2 Usuario identificado: Obtener el 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

[Prueba41] Mostrar el listado de ofertas para dicho usuario y comprobar que se muestran todas las que existen para este usuario. Esta prueba implica invocar a dos servicios: S1 y S2.

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).
- Deberá validarse:
    - El usuario propietario no podrá iniciar una conversación en una oferta propia.
    - No será posible enviar un mensaje vacío para iniciar una conversación.
    - Los únicos usuarios que pueden participar en una conversación son: el usuario que inició la conversación y el propietario de la oferta.

Pruebas Funcionales

[Prueba42] Enviar un mensaje a una oferta. Esta prueba consistirá en comprobar que el servicio almacena correctamente el mensaje para dicha oferta. Por lo tanto, el usuario tendrá que identificarse (S1), enviar un mensaje para una oferta de id conocido (S3) y comprobar que el mensaje ha quedado bien registrado (S4).

[Prueba43] Enviar un primer mensaje una oferta propia y comprobar que no se inicia la conversación. En este caso de prueba, el propietario de la oferta tendrá que identificarse (S1), enviar un mensaje para una oferta propia (S3) y comprobar que el mensaje no se almacena (S4).

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

[Prueba44] Obtener los mensajes de una conversación. Esta prueba consistirá en comprobar que el servicio retorna el número correcto de mensajes para una conversación. El ID de la conversación deberá conocerse a priori. Por lo tanto, se tendrá primero que invocar al servicio de identificación (S1), y solicitar el listado de mensajes de una conversación de id conocido a continuación (S4), comprobando que se retornan los mensajes adecuados.

S5 Usuario identificado: 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

[Prueba45] Obtener la lista de conversaciones de un usuario. Esta prueba consistirá en comprobar que el servicio retorna el número correcto de conversaciones para dicho usuario. Por lo tanto, se tendrá primero que invocar al servicio de identificación (S1), y solicitar el listado de conversaciones a continuación (S5) comprobando que se retornan las conversaciones adecuadas.

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

[Prueba46] Eliminar una conversación de ID conocido. Esta prueba consistirá en comprobar que se elimina correctamente una conversación concreta. Por lo tanto, se tendrá primero que invocar al servicio de identificación (S1), eliminar la conversación ID (S6) y solicitar el listado de conversaciones a continuación (S5), comprobando que se retornan las conversaciones adecuadas.

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

[Prueba47] Marcar como leído un mensaje de ID conocido. Esta prueba consistirá en comprobar que el mensaje marcado de ID conocido queda marcado correctamente a true como leído. Por lo tanto, se tendrá primero que invocar al servicio de identificación (S1), solicitar el servicio de marcado (S7), comprobando que el mensaje marcado ha quedado marcado a true como leído (S4).

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

[Prueba48] Inicio de sesión con datos válidos.

[Prueba49] Inicio de sesión con datos inválidos (email existente, pero contraseña incorrecta).

[Prueba50] Inicio de sesión con datos inválidos (campo email o contraseña vacíos).

C2 Usuario registrado: 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, fecha de publicación 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

[Prueba51] Mostrar el listado de ofertas disponibles y comprobar que se muestran todas las que existen, menos las del usuario identificado.

C3 Usuario registrado: Establecer conversación en una oferta

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

1) Al lado de cada oferta añadir un botón o enlace "Conversación" 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". Las conversaciones son únicas para cada oferta. Es decir, el usuario A tendrá múltiples conversaciones con el usuario B, siempre y cuando sean para ofertas diferentes.
2) Una propuesta de interfaz sería un pequeño formulario donde escribir el mensaje y una tabla donde se muestran los mensajes de uno y otro.
3) 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.

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.
- Deberá tenerse en cuenta las validaciones realizadas en el servidor (REQUISITO OBLIGATORIO S3):
  - El usuario propietario no podrá iniciar una conversación en una oferta propia.
  - No será posible enviar un mensaje vacío para iniciar una conversación.
  - Los únicos usuarios que pueden participar en una conversación son: el usuario que inició la conversación y el propietario de la oferta.

Pruebas Funcionales

[Prueba52] Sobre listado de ofertas disponibles (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.

[Prueba53] Sobre el listado de conversaciones enviar un mensaje a una conversación ya abierta. Comprobar que el mensaje aparece en el listado de mensajes.

C4 Usuario registrado: 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

[Prueba54] Mostrar el listado de conversaciones ya abiertas. Comprobar que el listado contiene la cantidad correcta de conversaciones.

C5 OPTATIVO - Usuario registrado: 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

[Prueba55] Sobre el listado de conversaciones ya abiertas. Pinchar el enlace Eliminar en la primera y comprobar que el listado se actualiza correctamente.

[Prueba56] Sobre el listado de conversaciones ya abiertas. Pinchar el enlace Eliminar en la última y comprobar que el listado se actualiza correctamente.

C6 OPTATIVO - Usuario registrado: 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

[Prueba57] 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 - Usuario registrado: 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

[Prueba58] 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 un proyecto plantilla de ejemplo**.

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

El grupo puede añadir más pruebas además de las planteadas en el enunciado, pero deberán incluirla en los ficheros correspondientes y en la documentación.

El desarrollo de las pruebas estará basado en el framework Selenium y JUnit5 vistos durante la asignatura.

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

Datos de prueba y tiempo de ejecución de pruebas

Se deberá poblar la base datos al arrancar la aplicación con un mínimo de datos que permite ejecutar las pruebas solicitadas:

- Un único usuario administrador con las siguientes credenciales:
  - Login: **admin@email.com**
  - Password: **admin**
- Usuarios registrados: al menos 15 (para que permitan al menos un listado de usuarios con 3 páginas de 5 usuarios por páginas). Seguir el patrón de nombrado siguiente:
  - Login user01@email.com, user02@email.com, ….
  - Password: user01, user02, ..
- Ofertas por usuario: al menos 10 ofertas por usuario que permitan un listado con páginas de 5 ofertas.

Un aspecto a tener en cuenta es que las pruebas no deben depender del resultado las anteriores. Por ello, debe pensarse en cómo poblar adecuadamente la base de datos para que cada prueba parta de un conjunto de datos conocido. Por otro lado, debe buscarse un compromiso entre este aspecto y el tiempo de ejecución de las pruebas.

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). **Todos estos ficheros son OBLIGATORIOS.** Se suministran dos plantillas que deberán tomarse como base para la elaboración de dichos documentos:

- Una plantilla Word para el informe (sdi2223-entrega2-n.docx). (Entregar en formato PDF).
- Una plantilla Excel para los casos de prueba sdi2223-entrega2-n.xlsx). (Entregar en formato Excel).

Ambos documentos deberán ser renombrados cambiando la n por el código del grupo asignado al equipo (ej. sdi2223-entrega2-22.pdf, sdi2223-entrega2-22.xlsx).

El contenido del informe deberá ser el siguiente:

- En la portada se deberá incluir:
  - Los datos personales de los autores: Nombre y apellidos, IDGIT, Emails de UNIOVI, Código ID GIT de cada uno y código del equipo.
- Diagrama de navegabilidad con texto explicativo. Un diagrama de navegabilidad no se compone de capturas de pantalla de la interfaz. Pueden usarse estas como apoyo en el texto explicativo, pero nunca como sustitutivo.

- Una descripción clara y detallada aquellos aspectos técnicos y/o diseño que considere relevantes. No se trata de describir qué es Node.js 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. **Obligatorio usar las versiones de los softwares usados en clase.**
- Conclusión individual de cada miembro del grupo. Esto incluye la aportación en el desarrollo de la práctica, así como las dificultades encontradas durante el mismo. También qué ventajas y desventajas se han detectado a nivel técnico y a nivel de trabajo en grupo. Una conclusión individual incompleta, repercutirá negativamente en la nota individual del alumno.

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.

Aspectos transversales en la evaluación

La calificación máxima que se puede obtener será de 11 puntos, en base al siguiente reparto: 8.0 puntos pertenecen a la funcionalidad obligatoria, 3.0 puntos a la parte opcional

Si bien existen aspectos que se evaluarán de forma transversal a todos los requisitos establecidos en el presente enunciado. **Como normal general se considera que, si no se cumplen, penalizan a la nota obtenida en la práctica por el grupo.**

Aspectos generales

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 y los distintos endpoints de los servicios web.

Proceso de entrega y protocolo de prueba

Según el número asignado a cada equipo, se deberán crear los proyectos Node.js y las pruebas unitarias (Selenium) con los nombres **sdi2223-entrega2-n** y **sdi2223-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 **sdi2223-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 **sdi2223-entrega2-n.zip** (en minúsculas) y que deberá contener en su raíz:
   - El INFORME OBLIGATORIO en formato PDF con nombre **sdi2223-entrega2-n.pdf y el archivo Excel con el catálogo de casos de prueba (sdi2223-entrega2-n.xlsx).**
   - El proyecto Node.js en formato carpeta (no comprimido) con el nombre **sdi2223-entrega2-n.**
   - El proyecto Java eclipse con los casos de prueba en formato carpeta (no comprimido) con el nombre **.\sdi2223-entrega2-test-n**
   - En resumen, el zip deberá contener en su raíz:
     - **sdi2223-entrega2**-n.pdf (Archivo PDF suministrado y renombrado cambiado la n).
     - **sdi2223-entrega2**-n.xlsx (Archivo Excel suministrado y renombrado cambiado la n).
     - **sdi2223-entrega2**-n (Carpeta proyecto Node.js renombrada cambiado la n)
     - **sdi2223-entrega2-test**-n (Carpeta proyecto Java eclipse renombrada cambiado la n)

El protocolo de prueba

Para probar cada proyecto el profesor realizará los siguientes pasos:

a) Importar y ejecutar la aplicación Node.js.
b) Importará y ejecutar el proyecto JUnit en Intellij IDEA.

Nota importante: Es requisito indispensable utilizar una base de datos Mongo en local y que se generen automáticamente todos los datos necesarios para ejecutar las pruebas.

Fecha máxima de entrega

La fecha de entrega será el ***Domingo 07/05/2023 a las 23:55.***

Consideraciones importantes:

- ***No se aceptará ningún trabajo fuera de plazo.***
- ***Todos los trabajos deben entregarse por el campus virtual.***
- ***No se aceptará ningún trabajo que sea entregado por una vía diferente a la especificada en este documento.***