Grado en TIS

Sistemas Distribuidos

Curso 2021-2022

Práctica 2

Estilo de vida saludable

Crear una plataforma web distribuida mediante estándares (HTML, CSS y Javascript) para promover el estilo de vida saludable. La aplicación permitirá a un profesional médico realizar un seguimiento de distintas variables asociadas al paciente para evaluar si realiza un estilo de vida saludable. Para ello se realizarán dos aplicaciones web, una para el médico y otra para el paciente.

El sistema distribuido estará formado por tres módulos:

- Un servidor basado en Javascript mediante la plataforma NodeJS.
- Una aplicación web para el médico
- Una aplicación web para el paciente

Para la realización de la práctica se utilizarán distintos mecanismos de comunicación: Servicios Web tipo REST, RPC y WebSockets

Servidor y sistema de información

Toda la información del sistema estará centralizada en el servidor. Las dos aplicaciones obtendrán toda la información del servidor, utilizando para ello las distintas tecnologías que se proponen. Solo existirá un UNICO servidor con todos los datos y que implementará las TRES tecnologías.

En el servidor los datos serán almacenados como colecciones en memoria, simples arrays de objetos de javascript (no será necesario utilizar una base de datos), por lo que cada vez que se reinicie el servidor se perderán los cambios realizados.

Los datos a gestionar son (cada uno es un array):

- **medicos**: Conjunto de médicos dados de alta en el sistema. Por cada médico se tendrán los siguientes dstos:
  - **id**: Identificador del médico, número único por cada médico (1,2,3…)
  - **nombre**: Nombre del médico
  - **login**: Texto utilizado para realizar la entrada al sistema

    - **password**: Contraseña asociada al login para realizar la entrada al sistema
- **pacientes**: Conjunto de pacientes que el médico irá dando de alta. Cada paciente sólo está asignado a un médico.
    - **id**: Identificador numérico único para cada paciente (1,2,3…)
    - **nombre**: Nombre del paciente
    - **fecha_nacimiento**: Fecha de nacimiento del paciente
    - **genero**: Género del paciente
    - **medico**: identificador (ID) del médico
    - **codigo_acceso**: Código de acceso alfanumérico para este paciente (creado por el médico)
    - **observaciones**: Texto de observaciones realizadas por su médico
- **variables**: Conjunto de variables sobre el estilo de vida que se monitorizarán (peso, metros andados, metros corridos, minutos de gimnasia realizados…)
    - **id**: Identificado numérico único para casa variable (1,2,3…)
    - **nombre**: Nombre de la variable
- **muestras**: Conjunto de valores para una variable y paciente en un momento dado
    - **id**: identificador numérico único de cada valor (1,2,3…)
    - **paciente**: Identificador (id) del paciente asociado a la muestra
    - **variable**: Identificador (id) de la variable asociada a la muestra
    - **fecha**: Fecha y hora en la que se tomó la muestra
    - **valor**: Valor (numérico) correspondiente a esta muestra

A continuación, se describen los requerimientos funcionales de cada uno de los módulos

Parte 1: Gestión del médico

La primera parte de la práctica consiste en la realización de la aplicación de gestión para el médico.

Se trata de una aplicación web que está alojada y trabaja con el servidor, e implementada con servicios web de tipo **REST asíncronos**.

Requerimientos funcionales

Los requerimientos funcionales de la aplicación del médico son:

- El médico entrará al sistema mediante un acceso por usuario y contraseña. Sólo si la combinación usuario/cotraseña es correcta podrá continuar.
- Una vez dentro del sistema se le mostrar un mensaje de bienvenida con su nombre y podrá realizar las siguientes operaciones:
    - Nada más entrar verá un **listado con los pacientes** que tiene asignados.
    - Tendrá una opción que le permitirá **salir** del sistema (la aplicación vuelve a la pantalla de login).
    - Tendrá una opción que le permitirá **agregar un nuevo paciente**. Esta opción le solicitará los datos del paciente, incluido un código de acceso que tendrá que dar al paciente para acceder a su aplicación. Al introducir un nuevo paciente se volverá a la lista y se actualizará automáticamente con el nuevo paciente introducido.
    - Si pulsa sobre un paciente de la lista **consultará el expediente** de ese paciente donde verá:
        - **Datos del paciente** (se pueden **modificar** sus datos).
        - Listado de **muestras de las variables del paciente**. Los valores se **filtran por cada variable** para ver la evolución temporal de cada una de ellas.

Esta gestión se implementará mediante la tecnología de Servicios Web tipo **REST asíncronos** entre el servidor y la aplicación del paciente.

Servidor REST

Para que la aplicación del médico obtenga toda la información necesaria, el servidor ofertará un conjunto de servicios REST con Node/Express, para acceder a la información almacenada en los arrays del servidor.

Los servicios a implementar son (SOLO IMPLEMENTAR ESTOS, **NO SE PUEDE IMPLEMENTAR OTROS)**:

URL	Método	Descripción
/api/variable	GET	Obtiene un array con todas las variables
/api/medico/login	POST	Realiza un login para medico. En body se indican las credenciales. Por ejemplo: `{"login": "xxx","password": "secreto"}` Si va bien se obtiene el id del medico: 5
/api/paciente/:id	GET	Obtiene los datos del paciente indicado (no devolver el código de

		acceso)
		Por ejemplo, si se realiza
		`GET /api/paciente/5`
		Se obtiene
		`{ "id": 5, "nombre": "Diego", "medico": 2, "observaciones": "" }`

/api/medico/:id	GET	Obtiene los datos del médico (**no devolver la contraseña**)
/api/medico/:id/pacientes	GET	Obtiene un array con los datos de sus pacientes
/api/medico/:id/pacientes	POST	Crea un nuevo paciente
/api/paciente/:id	PUT	Actualiza los datos de un paciente
/api/paciente/:id/muestras	GET	Obtiene todas las muestras de ese paciente

Todos los servicios retornarán como código de estado:

- 200: Si todo ha ido bien
- 201: Si un servicio POST ha creado correctamente un registro
- 404: Si se solicita un elemento que no existe (p.e. GET /api/medico/7, y no existe un médico con código 7)
- 403: Si la autenticación (login) no es correcta

Parte 2: Aplicación del paciente

La segunda parte de la práctica consiste en la realización de la aplicación para el paciente.

Se trata de una aplicación web que está alojada y trabaja con el servidor, e implementada con RPC de forma **asíncrona**. Existirá un único servidor, que implementará ambos mecanismos (REST y RPC) y que por lo tanto compartirán los mismos datos (los arrays con la información).

Requerimientos funcionales

Los requerimientos funcionales de la aplicación del paciente son:

- El paciente entrará al sistema mediante un acceso por código de acceso (solo se requerirá un dato, el código de acceso). Solo si el código es correcto (corresponde a un paciente) podrá continuar.

- Una vez dentro del sistema se le mostrar un mensaje de bienvenida con:
  - Su nombre,
  - El nombre de su médico
  - Las observaciones que el médico le ha escrito.
- El paciente podrá realizar las siguientes operaciones:
  - Tendrá una opción que le permitirá salir del sistema (la aplicación vuelve a la pantalla de login).
  - Nada más entrar verá un listado con los valores que ha introducido de todas las variables.
  - Podrá filtrar esos valores para ver sólo los de una única variable (por ejemplo, ver la evolución del peso).
  - Tendrá una opción para añadir un nuevo valor para una variable. Para ello se le solicitará la fecha (que por defecto será la actual) y el valor numérico que tiene esa variable en esa fecha.
  - Tendrá una opción para eliminar un valor incorrecto.

Esta gestión se implementará mediante la tecnología de RPC entre el servidor y la aplicación del paciente.

Servidor RPC

Para que la aplicación del paciente obtenga toda la información necesaria, el servidor ofertará un conjunto de procedimientos. Estos procedimientos trabajarán con los arrays definidos al inicio de la práctica.

Procedimiento	Valor devuelto	Descripción
`login(codigoAcceso)`	Un objeto con todos los datos del paciente o NULL si el código no es correcto	Realiza un login para paciente.
`listadoVariables()`	Array con todas las variables	Obtiene un listado de las variables.
`datosMedico(idMedico)`	Objeto con los datos del médico (excepto login y password). Si no existe devuelve NULL.	Obtiene los datos del médico indicado.
`listadoMuestras(idPaciente)`	Un array con todos las muestras introducidos por ese paciente.	Obtiene un listado de los valores de variables del paciente
`agregarMuestra(idPaciente, idVariable, fecha, valor)`	Id de la nueva muestra	Añade una nueva muestra.
`eliminarMuestra(idValor)`	Boleano indicando si ha ido bien	Elimina una muestra.

Solo se podrán implementar estos procedimientos en el servidor.

Parte 3: Socialización

Para motivar a los pacientes a mejorar su estado de salud, se pretende incluir en el sistema una funcionalidad de socialización, que permita a un paciente compartir en tiempo real el valor de una muestra. De esta forma, un paciente que haya conseguido un día correr 10Km, podrá hacer saber ese valor a otro usuario del sistema.

La comunicación de esta información se realizará en tiempo real, es decir, en el mismo momento que una persona comparta un valor, el o los usuarios a los que ha compartido el valor verán un mensaje informando sobre ello. Es por ello por lo que para su implementación se utilizará la tecnología WebSocket.

Requerimientos funcionales

Desde la aplicación del paciente, en la zona donde se introducen los valores de las muestras, un paciente podrá decidir compartir uno de esos valores (por ejemplo mediante un botón de “Compartir”. Cuando se comparta el sistema le preguntará al usuario con quien quiere compartir el valor:

- Con su médico
- Con todos los pacientes
- Con un solo paciente (por ejemplo, un amigo). En este caso se dejará seleccionar un paciente de los pacientes que tenga asignado su médico.

Esta selección se podrá realizar, por ejemplo, con un SELECT o con botones radio.

Al o los destinatarios (siempre que estén conectados) les llegará un mensaje del tipo: “Juan López ha compartido contigo que el día 12/02/2022 realizó la actividad ‘Correr’ y obtuvo un valor de 12”.

Estas comparticiones NO se almacenarán en la BD, por lo que, si en ese momento el usuario no está conectado, no recibirá el mensaje.

Servidor WebSocket

El servidor recibirá las solicitudes de compartición de un paciente y se las enviará a los destinatarios que estén conectados en ese momento.

Utilizar el módulo de node “websocket”.

El servidor de WebSocket estará integrado con el resto de los servicios, y utilizará los mismos arrays de datos.

La práctica se podrá realizar de forma individual o en parejas.

Entregar todos los archivos utilizados para realizar la práctica.