Segundo-parcial.pdf

IngenieroLloron

Ingeniería Web

4º Grado en Ingeniería del Software

Escuela Técnica Superior de Ingeniería Informática
Universidad de Málaga

Reservados todos los derechos.
No se permite la explotación económica ni la transformación de esta obra. Queda permitida la impresión en su totalidad.

Parcial 2. Servicios Web

En este ejercicio desarrollarás servicios REST para *Kalendas*, una aplicación web de gestión de agenda similar a *Google Calendar*, en la que los usuarios pueden gestionar su agenda y compartir o invitar a eventos a sus contactos.

1. Diseño de la base de datos (2 puntos)

En *Kalendas* distinguiremos dos tipos de entidades:

- ***Usuarios.*** Representa los usuarios de la aplicación, con los siguientes atributos:
  - ***email.*** Dirección de *email* del usuario. Funcionará como identificador de la entidad.
  - ***nombre.*** Nombre del usuario, tal como se presenta a otros usuarios de la aplicación.
  - ***contactos.*** Lista de contactos del usuario, representado cada uno por su *email*.
- ***Eventos.*** Representa los eventos de agenda, con los siguientes atributos:
  - ***identificador.*** Clave o identificador del evento, propia de la base de datos elegida.
  - ***anfitrión.*** Dirección de *email* del usuario anfitrión del evento (el que crea el evento).
  - ***descripción.*** Título o descripción breve (hasta 50 caracteres) del evento.
  - ***inicio.*** *Timestamp* con la fecha y hora de inicio del evento (en tramos de 15 minutos).
  - ***duración.*** Duración del evento (en tramos de 15 minutos).
  - ***invitados.*** Lista de invitados al evento, representado cada uno por su *email* y el estado de la invitación (*aceptada*/*pendiente*).

Para el almacenamiento de información, utilizarás una base de datos no relacional. Por tanto, estas entidades del modelo conceptual pueden dar lugar a un número mayor o menor de entidades de base de datos, con más o menos atributos, dependiendo de la base de datos elegida y el diseño creado para representarlos.

2. Servicios REST básicos (2 puntos)

Una vez diseñada la estructura de la base de datos, deberás desarrollar un proyecto que despliegue localmente un servidor REST cuyas operaciones devolverán sus resultados en JSON (opcionalmente, también en XML). Se valorará especialmente que los *endpoints* del servidor y sus posibles parámetros estén definidos de acuerdo a las normas de estilo propias de los servicios REST.

El servidor ofrecerá las siguientes operaciones básicas:

- CRUD de usuarios: GET ALL, GET, POST, PUT, DELETE.
- CRUD de contactos de un usuario: GET ALL (muestra todos sus contactos), POST (añade un contacto a la lista), DELETE (elimina un contacto de la lista).
- CRUD de eventos: GET ALL, GET, POST, PUT, DELETE.

3. Servicios REST adicionales (3 puntos)

Ampliarás la funcionalidad del servidor con las siguientes operaciones:

- Buscar entre los contactos de un usuario, identificado por su *email*, a partir de una cadena con parte del nombre del contacto, devolviendo una lista de contactos (*emails* y nombres).

Reservados todos los derechos. No se permite la explotación económica ni la transformación de esta obra. Queda permitida la impresión en su totalidad.

- Invitar a un evento a un contacto de un usuario, identificado por su *email*. El contacto invitado se incluirá en la lista de invitados del evento, con la invitación en estado *pendiente*.
- Aceptar una invitación a un evento. El estado de la invitación pasará a *aceptada*.
- Reprogramar un evento ya pasado, indicando cuánto tiempo se desplaza (un número de días determinado, una semana, un mes, o un año). Se creará un nuevo evento, con la nueva fecha y el resto de valores iguales a los del evento reprogramado.
- Obtener la agenda de un usuario, representada por una lista de eventos, tanto propios como invitados, por orden ascendente de inicio.

Se valorará que en todas estas operaciones se realicen los controles de errores necesarios para evitar corromper la información de la base de datos (fechas/horas inexistentes o incorrectas, invitar a un evento a alguien que no sea contacto de su anfitrión, aceptar un evento al que no se ha sido invitado, etc.)

4. Descripción y pruebas del servicio (3 puntos)

Todas las operaciones del servidor deberán poder probarse de forma independiente. Las pruebas pueden definirse a partir de una especificación *OpenAPI* o un conjunto de pruebas *Postman*, que acompañarán a la entrega del examen. Tanto en un caso como en otro deben de ser lo suficientemente descriptivas (incluyendo los comentarios y ejemplos que consideres necesarios) como para permitir la prueba del servidor de forma sencilla.

Entrega

Este ejercicio se entregará a través del campus virtual, mediante un archivo comprimido que contenga:

- Una memoria en la que describas:
  - el diseño de la base de datos (apartado 1) describiendo su estructura (entidades y atributos) y su URI o credenciales de acceso (por ejemplo, para MongoDB Atlas o Firestore). Si la base de datos es local, deberás adjuntar en la entrega del examen una descarga o *backup* de la misma.
  - las tecnologías utilizadas (lenguaje de programación, *frameworks*, etc.), y los *scripts* e instrucciones de instalación y despliegue (si son necesarias), así como el puerto de despliegue del servidor.
  - una relación de las limitaciones de la solución propuesta y los problemas encontrados durante su desarrollo (si procede).
- Las fuentes completas (no basta con una referencia a un repositorio GitHub) del servidor REST con las operaciones desarrolladas (apartados 2 y 3).
- La especificación OpenAPI del servidor, o un conjunto de pruebas Postman del servicio (apartado 4).

Reservados todos los derechos. No se permite la explotación económica ni la transformación de esta obra. Queda permitida la impresión en su totalidad.