

## **Documentación técnica Woowup - Challenge**

### **Ambiente de desarrollo:**

- Sistema operativo: GNU/Linux Pop OS 22.04
- Memoria RAM: 16GB
- Procesador: Intel i7 8va generación.

### **Tecnologías utilizadas:**

- Java 8
- Spring Boot
- Visual Studio Code
- Thunder Client para pruebas
- Mockito

### **Endpoints para probar:**

Una vez ejecutado el programa, tanto con Postman como Thunder Client, es posible probar la respuesta de los endpoints, los cuales son los que se mencionan a continuación:

#### **AlertaController:**

Endpoint: POST /crearAlerta

Descripción: Crea una nueva alerta utilizando la información proporcionada en el cuerpo de la solicitud.

Endpoint: POST /bajaAlerta

Descripción: Da de baja una alerta según la información proporcionada en el cuerpo de la solicitud.

Endpoint: GET /consultarAlertas

Descripción: Consulta y devuelve la lista de alertas existentes.

#### **TemaController:**

Endpoint: POST /crearTema

Descripción: Crea un nuevo tema utilizando la información proporcionada en el cuerpo de la solicitud.

#### **UsuarioController:**

Endpoint: POST /suscripcionAlerta

Descripción: Suscribe a un usuario a una alerta según la información proporcionada en el cuerpo de la solicitud.

Endpoint: GET /consultarSuscripciones

Descripción: Consulta las suscripciones del usuario según la información proporcionada en el cuerpo de la solicitud.

Endpoint: GET /listarNotificacionesUsuario

Riveros Sebastian  
DNI: 42974713

Descripción: Lista las notificaciones del usuario especificadas en el cuerpo de la solicitud.

Endpoint: POST /marcarAlertaLeida

Descripción: Marca una notificación como leída según la información proporcionada en el cuerpo de la solicitud.

Endpoint: POST /registrarUsuario

Descripción: Registra un nuevo usuario utilizando la información proporcionada en el cuerpo de la solicitud.

Se adjunta el link del archivo JSON con los cuerpos de cada solicitud para facilitar la prueba del programa:

[https://drive.google.com/file/d/1XTaLMMS\\_I1xTBubzVSZ8pDwJx\\_IGINv/view?usp=sharing](https://drive.google.com/file/d/1XTaLMMS_I1xTBubzVSZ8pDwJx_IGINv/view?usp=sharing)

### Diagrama de clases utilizado para resolver el reto:

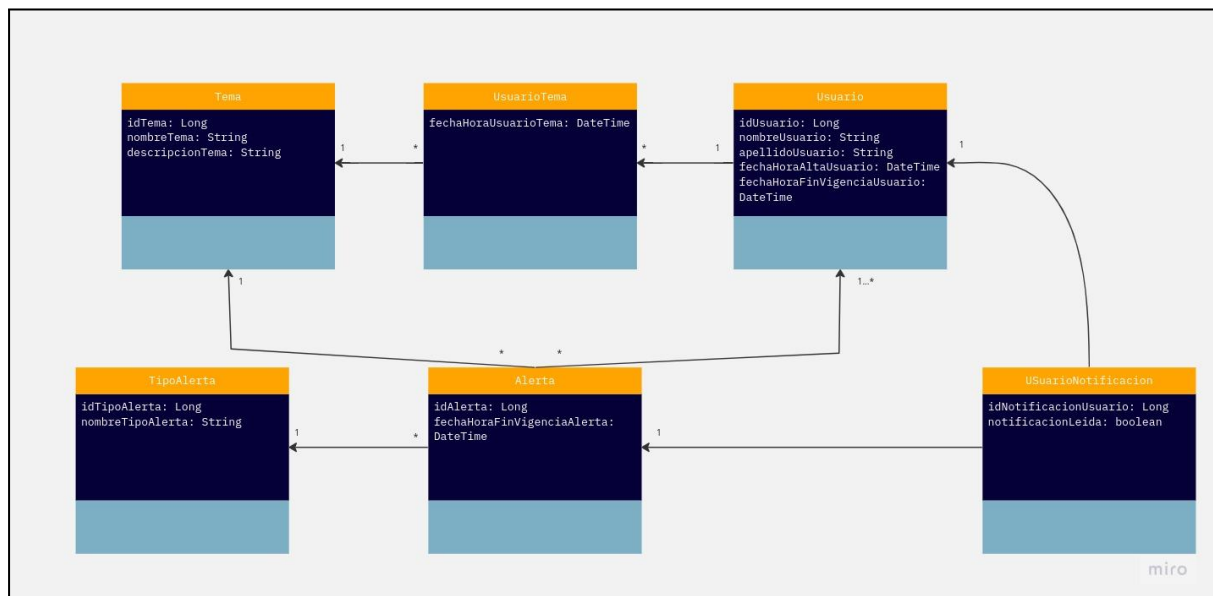


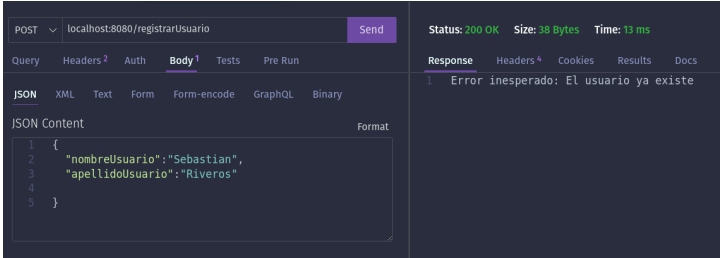
Figura 1: Diagrama de clases

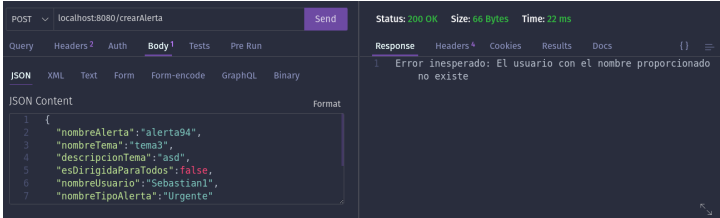
### Consulta SQL:

```
SELECT C.ID, C.Nombre, C.Apellido
FROM Clientes C
JOIN Ventas V ON C.ID = V.Id_cliente
WHERE V.Fecha >= DATEADD(MONTH, -12, GETDATE())
GROUP BY C.ID, C.Nombre, C.Apellido
HAVING SUM(V.Importe) > 100000;
```

Riveros Sebastian  
DNI: 42974713

Casos de prueba:

Nombre del caso	Alta usuario ya existente
Descripción	Se realiza el alta a un usuario con un nombre de usuario ya existente
Datos de prueba	nombreUsuario: Sebastian
Resultado esperado	Se espera que NO se dé el alta del usuario y que lance una excepción
Resultado obtenido	No se da de alta al usuario y se lanza una excepción
Evidencias	
RESULTADO DE PRUEBA	ÉXITO

Nombre del caso	Crear una alerta con usuario NO existente
Descripción	Se realiza el alta de una alerta, asignando un usuario que no existe a la misma
Datos de prueba	nombreUsuario: Sebastian
Resultado esperado	Se espera que NO se dé el alta de la alerta y que lance una excepción
Resultado obtenido	No se da de alta la alerta y lanza una excepción
Evidencias	
RESULTADO DE PRUEBA	ÉXITO

Nombre del caso	Leer notificación de usuario
Descripción	Se realiza la lectura de una notificación para que no aparezca mas como SIN VER
Datos de prueba	nombreUsuario: Sebastian idNotificación: 1
Resultado esperado	Se espera que al leer la notificación, deje de aparecer en la lista de notificaciones del usuario.
Resultado obtenido	El usuario lee la notificación y la misma deja de aparecer en la lista de notificaciones.
Evidencias	<div>1- Se crea la Alerta para Sebastian</div> <div></div> <div>2-Se listan las notificaciones para Sebastian</div> <div></div> <div>3-Ejecutamos el método de leer notificación e ingresamos el idNotificacion</div> <div></div> <div>4-Volvemos a listar las notificaciones</div> <div></div>
RESULTADO DE PRUEBA	ÉXITO