Evaluación: JAVA

Desarrolle una aplicación que exponga una API RESTful de creación de usuarios, entiéndase por esto, la implementación de los verbos GET, POST, PUT, PATCH Y DELETE .

Todos los endpoints deben aceptar y retornar solamente JSON, inclusive para los mensajes de error.

Todos los mensajes de error deben seguir el siguiente formato:

```
{"mensaje": "mensaje de error"}
```

Lógica de negocio para la creación de usuarios

- Este endpoint deberá recibir un usuario (correspondiente al recurso a trabajar) con los campos "nombre", "correo", "contraseña", más un listado de objetos "teléfonos", respetando el siguiente formato:

```
{
    "nombre": "Juan Rodriguez",
    "correo": "juan@rodriguez.org",
    "contraseña": "hunter2",
    "telefonos": [
        {
            "numero": "1234567",
            "codigoCiudad": "1",
            "codigoPais": "57"
        }
    ]
}
```

- Debe responder el código de status HTTP adecuado, dependiendo de la casuística.
- En caso de éxito, debe retornar los siguientes campos:
    - **id:** id del usuario (puede ser lo que se genera por la base de datos, deseable un UUID)
    - **creado:** fecha de creación del usuario
    - **modificado:** fecha de la última actualización de usuario
    - **ultimoLogin**: fecha del último ingreso (en caso de nuevo usuario, va a coincidir con la fecha de creación)
    - **token**: token de acceso de la API, usando JWT. El presente token, le permitirá al usuario creado poder realizar acciones sobre la API.
    - **activo**: Indica si el usuario sigue habilitado dentro del sistema.

- Si el “correo” existe en la base de datos, deberá retornar un error "El correo ya está registrado". Siguiendo el formato de error especificado en un principio.
- El “correo” debe seguir una expresión regular para validar que el formato sea el correcto. (aaaaaaa@dominio.cl), en caso de error manejar según lo indicado en un principio.
- La “contraseña” debe seguir una expresión regular para validar que el formato sea el correcto. (El valor de la expresión regular debe ser configurable), en caso de error manejar según lo indicado en un principio.
- El token deberá ser persistido junto con el usuario

Requisitos

- Plazo: 4 días, si tienes algún inconveniente con el tiempo comunicate con nosotros
- Base de datos en memoria. Ejemplo: HSQLDB o H2.
- Proceso de build vía Gradle o Maven.
- Persistencia con JPA. Ejemplo: EclipseLink, Hibernate u OpenJPA.
- Framework SpringBoot.
- Java 8+
- Entrega en un repositorio público (github o bitbucket) con el código fuente y script de creación de BD.
- Readme explicando cómo probarlo.
- Diagrama de la solución.

Requisitos opcionales

- Pruebas unitarias
- Swagger