Documento del Microservicio de Gestión de Usuarios

Hecho por: Vicente Cambrón Tocados (20622913D)

# 1. Nivel de Acabado

El nivel de acabado del microservicio es funcional, es decir, cubre todas las funcionalidades necesarias para gestionar usuarios en una aplicación. Se ha implementado el registro, inicio de sesión, administración de usuarios por parte de un administrador, y la verificación de roles. Adicionalmente, incluye la implementación de auditoría de acciones realizadas sobre los usuarios.

# 2. Descripción de la Aplicación

La aplicación es un sistema de gestión de usuarios que incluye un microservicio para el manejo de usuarios con funcionalidades de autenticación, administración de roles, auditoría y gestión de usuarios (creación, eliminación y actualización). Se utiliza un frontend que interactúa con este microservicio mediante APIs REST.

Además, el microservicio utiliza **SendGrid** para enviar correos electrónicos en las operaciones de recuperación de contraseñas. Actualmente, se está utilizando el **plan gratuito de SendGrid**, que permite un número limitado de correos electrónicos al mes, suficiente para gestionar las pruebas y operaciones de un entorno de desarrollo o baja escala. Si el uso aumenta, se evaluará la posibilidad de escalar a un plan de pago para garantizar un servicio continuo y sin interrupciones.

# 3. Descomposición en Microservicios

La aplicación está descompuesta en varios microservicios, de los cuales hemos implementado el microservicio de gestión de usuarios. Este microservicio permite la creación, actualización, autenticación y eliminación de usuarios, así como la gestión de auditorías relacionadas con los cambios realizados por los administradores.

# 4. Customer Agreement (CA)

El Customer Agreement (CA) de la aplicación se basa en los siguientes principios:

* Descripción del servicio: El microservicio de gestión de usuarios proporciona un conjunto de API REST para el registro, autenticación y gestión de usuarios, incluyendo la creación de roles y auditoría de actividades. Además, se utiliza SendGrid para enviar correos electrónicos en las operaciones de recuperación de contraseñas y notificaciones importantes a los usuarios.
* Términos de uso: Los usuarios tienen la responsabilidad de usar sus credenciales de manera segura, y los administradores son responsables de gestionar de forma adecuada las cuentas de usuario y roles. El servicio de correo electrónico es proporcionado por SendGrid, sujeto a las limitaciones del plan gratuito, lo que podría influir en la cantidad de correos electrónicos que se envían al mes. En caso de necesitar un mayor volumen de correos, se evaluará la migración a un plan de pago.
* Planes de precios: No aplica para este proyecto, pero el uso de SendGrid está limitado por el plan gratuito, que tiene restricciones sobre el número de correos electrónicos mensuales.
* Acuerdos de Nivel de Servicio (SLAs): La aplicación garantiza un 99% de disponibilidad, con un sistema de auditoría para verificar las acciones de los usuarios y administradores. Se han implementado medidas para asegurar la escalabilidad del sistema y evitar tiempos de inactividad. El servicio de correo electrónico proporcionado por SendGrid está sujeto a la disponibilidad de este proveedor y a las limitaciones del plan gratuito.

# 5. Descripción del API REST del Microservicio

El microservicio sigue el estilo arquitectónico REST, utilizando los métodos HTTP adecuados (GET, POST, PUT, DELETE) para interactuar con los recursos.

* **GET /api/v1/admin/users**
  + Descripción: Permite al administrador obtener la lista de todos los usuarios registrados en la aplicación.
  + Requiere Autorización: Solo usuarios con rol de administrador.
  + Respuesta: Devuelve una lista con información de los usuarios.
* **DELETE /api/v1/admin/users/{id}**
  + Descripción: Permite al administrador eliminar un usuario específico utilizando su ID.
  + Requiere Autorización: Solo usuarios con rol de administrador.
  + Respuesta: Confirmación de que el usuario ha sido eliminado con éxito.
* **GET /api/v1/admin/audits**
  + Descripción: Devuelve una lista de todas las auditorías registradas de las acciones administrativas realizadas en la aplicación.
  + Requiere Autorización: Solo usuarios con rol de administrador.
  + Respuesta: Lista de registros de auditoría, con detalles de las acciones realizadas.
* **POST /api/v1/admin/users/create-admin**
  + Descripción: Permite crear un nuevo usuario con rol de administrador.
  + Requiere Autorización: Solo usuarios con rol de administrador.
  + Respuesta: Confirmación de la creación exitosa del nuevo administrador junto con el token JWT.
* **GET /api/v1/admin/users/check-admin**
  + Descripción: Verifica si el usuario autenticado tiene privilegios de administrador.
  + Requiere Autorización: Debe ser un usuario autenticado.
  + Respuesta: Devuelve un booleano indicando si el usuario es administrador.
* **PUT /api/v1/admin/users/{id}/role**
  + Descripción: Permite al administrador actualizar el rol de un usuario específico.
  + Requiere Autorización: Solo usuarios con rol de administrador.
  + Respuesta: Confirmación de que el rol del usuario ha sido actualizado con éxito.
* **GET /api/v1/admin/users/{id}/audits**
  + Descripción: Obtiene los registros de auditoría para un usuario específico utilizando su ID.
  + Requiere Autorización: Solo usuarios con rol de administrador.
  + Respuesta: Lista de auditorías relacionadas con el usuario específico.
* **POST /api/v1/users/register**
  + Descripción: Permite registrar un nuevo usuario en la aplicación proporcionando la información necesaria como nombre, email y contraseña.
  + Requiere Autorización: No.
  + Respuesta: Confirmación del registro junto con el token JWT del usuario.
* **POST /api/v1/users/login**
  + Descripción: Permite a un usuario autenticarse y obtener un token JWT utilizando su correo electrónico y contraseña.
  + Requiere Autorización: No.
  + Respuesta: Retorna un token JWT que puede ser utilizado para acceder a los recursos protegidos.
* **GET /api/v1/users/profile**
  + Descripción: Permite al usuario autenticado obtener la información de su perfil.
  + Requiere Autorización: Sí, con token JWT.
  + Respuesta: Devuelve los detalles del perfil del usuario autenticado.
* **PUT /api/v1/users/profile**
  + Descripción: Permite a un usuario autenticado actualizar su perfil.
  + Requiere Autorización: Sí, con token JWT.
  + Respuesta: Confirmación de que los datos del perfil han sido actualizados exitosamente.
* **PUT /api/v1/users/password**
  + Descripción: Permite al usuario cambiar su contraseña actual.
  + Requiere Autorización: Sí, con token JWT.
  + Respuesta: Confirmación de que la contraseña ha sido actualizada con éxito.
* **DELETE /api/v1/users/delete**
  + Descripción: Permite a un usuario autenticado eliminar su cuenta.
  + Requiere Autorización: Sí, con token JWT y verificación de contraseña.
  + Respuesta: Confirmación de que la cuenta ha sido eliminada con éxito.
* **POST /api/v1/users/forgot-password**
  + Descripción: Permite a un usuario solicitar el restablecimiento de su contraseña proporcionando su correo electrónico.
  + Requiere Autorización: No.
  + Respuesta: Confirmación de que se ha enviado el correo de recuperación.
* **POST /api/v1/users/reset-password/{token}**
  + Descripción: Permite al usuario restablecer su contraseña utilizando un token de recuperación recibido por correo electrónico.
  + Requiere Autorización: No.
  + Respuesta: Confirmación de que la contraseña ha sido restablecida con éxito.

# 6. Justificación de Requisitos Implementados

- Autenticación de usuarios con JWT: Utilizamos jsonwebtoken para generar y verificar tokens JWT, almacenando la identidad del usuario y permitiendo que cada petición incluya este token. El código de implementación se encuentra en controllers/userController.js, en las funciones de login y protección de rutas mediante middleware.  
- Gestión de roles: Los administradores pueden modificar los roles de los usuarios utilizando la API. Esta funcionalidad está implementada en adminController.js en la función updateUserRole.  
- Auditoría: Se registra cada acción administrativa importante en un sistema de auditoría para asegurar la trazabilidad. Las auditorías se implementan en adminController.js y auditModel.js.  
- Gestión de usuarios: Se permite la creación, actualización y eliminación de usuarios a través de las rutas definidas en userRoutes.js y adminRoutes.js.